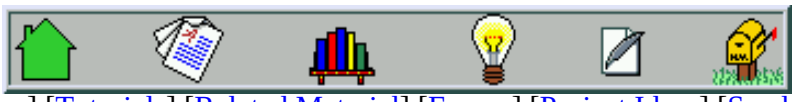


v1.1.1

Multi-Threaded Programming With POSIX Threads

Table Of Contents:

Before We Start...

This tutorial is an attempt to help you become familiar with multi-threaded programming with the POSIX threads (pthreads) library, and attempts to show how its features can be used in "real-life" programs. It explains the different tools defined by the library, shows how to use them, and then gives an example of using them to solve programming problems. There is an implicit assumption that the user has some theoretical familiarity with paralell programming (or multi-processing) concepts. Users without such background might find the concepts harder to grasp. A seperate tutorial will be prepared to explain the theoreticl background and terms to those who are familiar only with normal "serial" programming.

I would assume that users which are familiar with asynchronous programming models, such as those used in windowing environments (X, Motif), will find it easier to grasp the concepts of multi-threaded programming.

When talking about POSIX threads, one cannot avoid the question "Which draft of the POSIX threads standard shall be used?". As this threads standard has been revised over a period of several years, one will find that implementations adhering to different drafts of the standard have a different set of functions, different default values, and different nuances. Since this tutorial was written using a Linux system with the kernel-level LinuxThreads library, v0.5, programmers with access to other systems, using different versions of pthreads, should refer to their system's manuals in case of incompatibilities. Also, since some of the example programs are using blocking system calls, they won't work with user-level threading libraries (refer to our parallel programming theory tutorial for more information). Having said that, i'd try to check the example programs on other systems as well (Solaris 2.5 comes to mind), to make it more "cross-platform".

What Is a Thread? Why Use Threads

A thread is a semi-process, that has its own stack, and executes a given piece of code. Unlike a real process, the thread normally shares its memory with other threads (where as for processes we usually have a different memory area for each one of them). A Thread Group is a set of threads all executing inside the same process. They all share the same memory, and thus can access the same global variables, same heap memory, same set of file descriptors, etc. All these threads execute in parallel (i.e. using time slices, or if the system has several processors, then really in parallel).

The advantage of using a thread group instead of a normal serial program is that several operations may be carried out in parallel, and thus events can be handled immediately as they arrive (for example, if we have one thread handling a user interface, and another thread handling database queries, we can execute a heavy query requested by the user, and still respond to user input while the query is executed).

The advantage of using a thread group over using a process group is that context switching between threads is much faster then context switching between processes (context switching means that the system switches from running one thread or process, to running another thread or process). Also, communications between two threads is usually faster and easier to implement then communications between two processes.

On the other hand, because threads in a group all use the same memory space, if one of them corrupts the contents of its memory, other threads might suffer as well. With processes, the operating system normally protects processes from one another, and thus if one corrupts its own memory space, other processes won't suffer. Another advantage of using processes is that they can run on different machines, while all the threads have to run on the same machine (at least

normally).

Creating And Destroying Threads

When a multi-threaded program starts executing, it has one thread running, which executes the main() function of the program. This is already a full-fledged thread, with its own thread ID. In order to create a new thread, the program should use the `pthread_create()` function. Here is how to use it:

```
#include <stdio.h>       /* standard I/O routines                 */
#include <pthread.h>     /* pthread functions and data structures */

/* function to be executed by the new thread */
void*
do_loop(void* data)
{
    int i;

    int i;                      /* counter, to print numbers */
    int j;                      /* counter, for delay        */
    int me = *((int*)data);     /* thread identifying number */

    for (i=0; i<10; i++) {
        for (j=0; j<500000; j++) /* delay loop */
            ;
        printf("'%d' - Got '%d'\n", me, i);
    }

    /* terminate the thread */
    pthread_exit(NULL);
}

/* like any C program, program's execution begins in main */
int
main(int argc, char* argv[])
{
    int        thr_id;         /* thread ID for the newly created thread */
    pthread_t  p_thread;       /* thread's structure                     */
    int        a         = 1;  /* thread 1 identifying number            */
    int        b         = 2;  /* thread 2 identifying number            */

    /* create a new thread that will execute 'do_loop()' */
    thr_id = pthread_create(&p_thread, NULL, do_loop, (void*)&a);
    /* run 'do_loop()' in the main thread as well */
    do_loop((void*)&b);

    /* NOT REACHED */
    return 0;
}
```

A few notes should be mentioned about this program:

1. Note that the main program is also a thread, so it executes the `do_loop()` function in parallel to the thread it creates.

2. `pthread_create()` gets 4 parameters. The first parameter is used by `pthread_create()` to supply the program with information about the thread. The second parameter is used to set some attributes for the new thread. In our case we supplied a NULL pointer to tell `pthread_create()` to use the default values. The third parameter is the name of the function that the thread will start executing. The forth parameter is an argument to pass to this function. Note the cast to a 'void*'. It is not required by ANSI-C syntax, but is placed here for clarification.
3. The delay loop inside the function is used only to demonstrate that the threads are executing in parallel. Use a larger delay value if your CPU runs too fast, and you see all the printouts of one thread before the other.
4. The call to `pthread_exit()` Causes the current thread to exit and free any thread-specific resources it is taking. There is no need to use this call at the end of the thread's top function, since when it returns, the thread would exit automatically anyway. This function is useful if we want to exit a thread in the middle of its execution.

In order to compile a multi-threaded program using `gcc`, we need to link it with the pthreads library. Assuming you have this library already installed on your system, here is how to compile our first program:

```
gcc pthread_create.c -o pthread_create -lpthread
```

Note that for some of the programs later on this tutorial, one may need to add a '-D_GNU_SOURCE' flag to this compile line, to get the source compiled.

The source code for this program may be found in the pthread_create.c file.

Synchronizing Threads With Mutexes

One of the basic problems when running several threads that use the same memory space, is making sure they don't "step on each other's toes". By this we refer to the problem of using a data structure from two different threads.

For instance, consider the case where two threads try to update two variables. One tries to set both to 0, and the other tries to set both to 1. If both threads would try to do that at the same time, we might get with a situation where one variable contains 1, and one contains 0. This is because a context-switch (we already know what this is by now, right?) might occur after the first tread zeroed out the first variable, then the second thread would set both variables to 1, and when the first thread resumes operation, it will zero out the second variable, thus getting the first variable set to '1', and the second set to '0'.

What Is A Mutex?

A basic mechanism supplied by the pthreads library to solve this problem, is called a mutex. A mutex is a lock that guarantees three things:

1. Atomicity - Locking a mutex is an atomic operation, meaning that the operating system (or threads library) assures you that if you locked a mutex, no other thread succeeded in locking this mutex at the same time.
2. Singularity - If a thread managed to lock a mutex, it is assured that no other thread will be able to lock the thread until the original thread releases the lock.
3. Non-Busy Wait - If a thread attempts to lock a thread that was locked by a second thread, the first thread will be suspended (and will not consume any CPU resources) until the lock is freed by the second thread. At this time, the first thread will wake up and continue execution, having the mutex locked by it.

From these three points we can see how a mutex can be used to assure exclusive access to variables (or in general critical code sections). Here is some pseudo-code that updates the two variables we were talking about in the previous section, and can be used by the first thread:

```
lock mutex 'X1'.
set first variable to '0'.
set second variable to '0'.
unlock mutex 'X1'.
```

Meanwhile, the second thread will do something like this:

```
lock mutex 'X1'.
set first variable to '1'.
set second variable to '1'.
unlock mutex 'X1'.
```

Assuming both threads use the same mutex, we are assured that after they both ran through this code, either both variables are set to '0', or both are set to '1'. You'd note this requires some work from the programmer - If a third thread was to access these variables via some code that does not use this mutex, it still might mess up the variable's contents. Thus, it is important to enclose all the code that accesses these variables in a small set of functions, and always use only these functions to access these variables.

Creating And Initializing A Mutex

In order to create a mutex, we first need to declare a variable of type `pthread_mutex_t`, and then initialize it. The simplest way it by assigning it the `PTHREAD_MUTEX_INITIALIZER` constant. So we'll use a code that looks something like this:

```
pthread_mutex_t a_mutex = PTHREAD_MUTEX_INITIALIZER;
```

One note should be made here: This type of initialization creates a mutex called 'fast mutex'. This means that if a thread locks the mutex and then tries to lock it again, it'll get stuck - it will be in a deadlock.

There is another type of mutex, called 'recursive mutex', which allows the thread that locked it, to lock it several more times, without getting blocked (but other threads that try to lock the mutex now will get blocked). If the thread then unlocks the mutex, it'll still be locked, until it is unlocked the same amount of times as it was locked. This is similar to the way modern door locks work - if you turned it twice clockwise to lock it, you need to turn it twice counter-clockwise to unlock it. This kind of mutex can be created by assigning the constant `PTHREAD_RECURSIVE_MUTEX_INITIALIZER_NP` to a mutex variable.

Locking And Unlocking A Mutex

In order to lock a mutex, we may use the function `pthread_mutex_lock()`. This function attempts to lock the mutex, or block the thread if the mutex is already locked by another thread. In this case, when the mutex is unlocked by the first process, the function will return with the mutex locked by our process. Here is how to lock a mutex (assuming it was initialized earlier):

```
int rc = pthread_mutex_lock(&a_mutex);
if (rc) { /* an error has occurred */
    perror("pthread_mutex_lock");
    pthread_exit(NULL);
```

```
}
/* mutex is now locked - do your stuff. */
.
.
```

After the thread did what it had to (change variables or data structures, handle file, or whatever it intended to do), it should free the mutex, using the `pthread_mutex_unlock()` function, like this:

```
rc = pthread_mutex_unlock(&a_mutex);
if (rc) {
    perror("pthread_mutex_unlock");
    pthread_exit(NULL);
}
```

Destroying A Mutex

After we finished using a mutex, we should destroy it. Finished using means no thread needs it at all. If only one thread finished with the mutex, it should leave it alive, for the other threads that might still need to use it. Once all finished using it, the last one can destroy it using the `pthread_mutex_destroy()` function:

```
rc = pthread_mutex_destroy(&a_mutex);
```

After this call, this variable (a_mutex) may not be used as a mutex any more, unless it is initialized again. Thus, if one destroys a mutex too early, and another thread tries to lock or unlock it, that thread will get a `EINVAL` error code from the lock or unlock function.

Using A Mutex - A Complete Example

After we have seen the full life cycle of a mutex, lets see an example program that uses a mutex. The program introduces two employees competing for the "employee of the day" title, and the glory that comes with it. To simulate that in a rapid pace, the program employs 3 threads: one that promotes Danny to "employee of the day", one that promotes Moshe to that situation, and a third thread that makes sure that the employee of the day's contents is consistent (i.e. contains exactly the data of one employee).
Two copies of the program are supplied. One that uses a mutex, and one that does not. Try them both, to see the differences, and be convinced that mutexes are essential in a multi-threaded environment.

The programs themselves are in the files accompanying this tutorial. The one that uses a mutex is employee-with-mutex.c. The one that does not use a mutex is employee-without-mutex.c. Read the comments inside the source files to get a better understanding of how they work.

Starvation And Deadlock Situations

Again we should remember that `pthread_mutex_lock()` might block for a non-determined duration, in case of the mutex being already locked. If it remains locked forever, it is said that our poor thread is "starved" - it was trying to acquire a resource, but never got it. It is up to the programmer to ensure that such starvation won't occur. The pthread library does not help us with that.

The pthread library might, however, figure out a "deadlock". A deadlock is a situation in which a set of threads are all waiting for resources taken by other threads, all in the same set. Naturally, if all threads are blocked waiting for a

mutex, none of them will ever come back to life again. The pthread library keeps track of such situations, and thus would fail the last thread trying to call `pthread_mutex_lock()`, with an error of type `EDEADLK`. The programmer should check for such a value, and take steps to solve the deadlock somehow.

Refined Synchronization - Condition Variables

As we've seen before with mutexes, they allow for simple coordination - exclusive access to a resource. However, we often need to be able to make real synchronization between threads:

- In a server, one thread reads requests from clients, and dispatches them to several threads for handling. These threads need to be notified when there is data to process, otherwise they should wait without consuming CPU time.
- In a GUI (Graphical User Interface) Application, one thread reads user input, another handles graphical output, and a third thread sends requests to a server and handles its replies. The server-handling thread needs to be able to notify the graphics-drawing thread when a reply from the server arrived, so it will immediately show it to the user. The user-input thread needs to be always responsive to the user, for example, to allow her to cancel long operations currently executed by the server-handling thread.

All these examples require the ability to send notifications between threads. This is where condition variables are brought into the picture.

What Is A Condition Variable?

A condition variable is a mechanism that allows threads to wait (without wasting CPU cycles) for some even to occur. Several threads may wait on a condition variable, until some other thread signals this condition variable (thus sending a notification). At this time, one of the threads waiting on this condition variable wakes up, and can act on the event. It is possible to also wake up all threads waiting on this condition variable by using a broadcast method on this variable.

Note that a condition variable does not provide locking. Thus, a mutex is used along with the condition variable, to provide the necessary locking when accessing this condition variable.

Creating And Initializing A Condition Variable

Creation of a condition variable requires defining a variable of type `pthread_cond_t`, and initializing it properly. Initialization may be done with either a simple use of a macro named `PTHREAD_COND_INITIALIZER` or the usage of the `pthread_cond_init()` function. We will show the first form here:

```
pthread_cond_t got_request = PTHREAD_COND_INITIALIZER;
```

This defines a condition variable named 'got_request', and initializes it.

Note: since the `PTHREAD_COND_INITIALIZER` is actually a structure, it may be used to initialize a condition variable only when it is declared. In order to initialize it during runtime, one must use the `pthread_cond_init()` function.

Signaling A Condition Variable

In order to signal a condition variable, one should either the `pthread_cond_signal()` function (to wake up a only one thread waiting on this variable), or the `pthread_cond_broadcast()` function (to wake up all threads waiting on this variable). Here is an example using signal, assuming 'got_request' is a properly initialized condition variable:

```
int rc = pthread_cond_signal(&got_request);
```

Or by using the broadcast function:

```
int rc = pthread_cond_broadcast(&got_request);
```

When either function returns, 'rc' is set to 0 on success, and to a non-zero value on failure. In such a case (failure), the return value denotes the error that occured (`EINVAL` denotes that the given parameter is not a condition variable. `ENOMEM` denotes that the system has run out of memory.

Note: success of a signaling operation does not mean any thread was awakened - it might be that no thread was waiting on the condition variable, and thus the signaling does nothing (i.e. the signal is lost).
It is also not remembered for future use - if after the signaling function returns another thread starts waiting on this condition variable, a further signal is required to wake it up.

Waiting On A Condition Variable

If one thread signals the condition variable, other threads would probably want to wait for this signal. They may do so using one of two functions, `pthread_cond_wait()` or `pthread_cond_timedwait()`. Each of these functions takes a condition variable, and a mutex (which should be locked before calling the wait function), unlocks the mutex, and waits until the condition variable is signaled, suspending the thread's execution. If this signaling causes the thread to awake (see discussion of `pthread_cond_signal()` earlier), the mutex is automagically locked again by the wait funciton, and the wait function returns.

The only difference between these two functions is that `pthread_cond_timedwait()` allows the programmer to specify a timeout for the waiting, after which the function always returns, with a proper error value (ETIMEDOUT) to notify that condition variable was NOT signaled before the timeout passed. The `pthread_cond_wait()` would wait indefinitely if it was never signaled.

Here is how to use these two functions. We make the assumption that 'got_request' is a properly initialized condition variable, and that 'request_mutex' is a properly initialized mutex. First, we try the `pthread_cond_wait()` function:

```
/* first, lock the mutex */
int rc = pthread_mutex_lock(&a_mutex);
if (rc) { /* an error has occurred */
    perror("pthread_mutex_lock");
    pthread_exit(NULL);
}
/* mutex is now locked - wait on the condition variable.             */
/* During the execution of pthread_cond_wait, the mutex is unlocked. */
rc = pthread_cond_wait(&got_request, &request_mutex);
if (rc == 0) { /* we were awakened due to the cond. variable being signaled */
               /* The mutex is now locked again by pthread_cond_wait()      */
    /* do your stuff... */
    .
}
/* finally, unlock the mutex */
pthread_mutex_unlock(&request_mutex);
```

Now an example using the `pthread_cond_timedwait()` function:

```
#include <sys/time.h>     /* struct timeval definition           */
#include <unistd.h>       /* declaration of gettimeofday()       */
```

```
struct timeval  now;            /* time when we started waiting        */
struct timespec timeout;        /* timeout value for the wait function */
int             done;           /* are we done waiting?                */

/* first, lock the mutex */
int rc = pthread_mutex_lock(&a_mutex);
if (rc) { /* an error has occurred */
    perror("pthread_mutex_lock");
    pthread_exit(NULL);
}
/* mutex is now locked */

/* get current time */
gettimeofday(&now);
/* prepare timeout value */
timeout.tv_sec = now.tv_sec + 5
timeout.tv_nsec = now.tv_usec * 1000; /* timeval uses microseconds.         */
                                      /* timespec uses nanoseconds.         */
                                      /* 1 nanosecond = 1000 micro seconds. */

/* wait on the condition variable. */
/* we use a loop, since a Unix signal might stop the wait before the timeout */
done = 0;
while (!done) {
    /* remember that pthread_cond_timedwait() unlocks the mutex on entrance */
    rc = pthread_cond_timedwait(&got_request, &request_mutex, &timeout);
    switch(rc) {
        case 0:  /* we were awakened due to the cond. variable being signaled */
                 /* the mutex was now locked again by pthread_cond_timedwait. */
            /* do your stuff here... */
            .
            .
            done = 0;
            break;
        case ETIMEDOUT: /* our time is up */
            done = 0;
            break;
        default:        /* some error occurred (e.g. we got a Unix signal) */
            break;      /* break this switch, but re-do the while loop.   */
    }
}
/* finally, unlock the mutex */
pthread_mutex_unlock(&request_mutex);
```

As you can see, the timed wait version is way more complex, and thus better be wrapped up by some function, rather then being re-coded in every necessary location.

Note: it might be that a condition variable that has 2 or more threads waiting on it is signaled many times, and yet one of the threads waiting on it never awakened. This is because we are not guaranteed which of the waiting threads is awakened when the variable is signaled. It might be that the awakened thread quickly comes back to waiting on the condition variables, and gets awakened again when the variable is signaled again, and so on. The situation for the un-awakened thread is called 'starvation'. It is up to the programmer to make sure this situation does not occur if it implies bad behavior. Yet, in our server example from before, this situation might indicate requests are coming in a

very slow pace, and thus perhaps we have too many threads waiting to service requests. In this case, this situation is actually good, as it means every request is handled immediately when it arrives.

Note 2: when the mutex is being broadcast (using pthread_cond_broadcast), this does not mean all threads are running together. Each of them tries to lock the mutex again before returning from their wait function, and thus they'll start running one by one, each one locking the mutex, doing their work, and freeing the mutex before the next thread gets its chance to run.

Destroying A Condition Variable

After we are done using a condition variable, we should destroy it, to free any system resources it might be using. This can be done using the `pthread_cond_destroy()`. In order for this to work, there should be no threads waiting on this condition variable. Here is how to use this function, again, assuming 'got_request' is a pre-initialized condition variable:

```
int rc = pthread_cond_destroy(&got_request);
if (rc == EBUSY) { /* some thread is still waiting on this condition variable */
    /* handle this case here... */
    .
    .
}
```

What if some thread is still waiting on this variable? depending on the case, it might imply some flaw in the usage of this variable, or just lack of proper thread cleanup code. It is probably good to alert the programmer, at least during debug phase of the program, of such a case. It might mean nothing, but it might be significant.

A Real Condition For A Condition Variable

A note should be taken about condition variables - they are usually pointless without some real condition checking combined with them. To make this clear, lets consider the server example we introduced earlier. Assume that we use the 'got_request' condition variable to signal that a new request has arrived that needs handling, and is held in some requests queue. If we had threads waiting on the condition variable when this variable is signaled, we are assured that one of these threads will awake and handle this request.

However, what if all threads are busy handling previous requests, when a new one arrives? the signaling of the condition variable will do nothing (since all threads are busy doing other things, NOT waiting on the condition variable now), and after all threads finish handling their current request, they come back to wait on the variable, which won't necessarily be signaled again (for example, if no new requests arrive). Thus, there is at least one request pending, while all handling threads are blocked, waiting for a signal.

In order to overcome this problem, we may set some integer variable to denote the number of pending requests, and have each thread check the value of this variable before waiting on the variable. If this variable's value is positive, some request is pending, and the thread should go and handle it, instead of going to sleep. Further more, a thread that handled a request, should reduce the value of this variable by one, to make the count correct.
Lets see how this affects the waiting code we have seen above.

```
/* number of pending requests, initially none */
int num_requests = 0;
.
.
/* first, lock the mutex */
int rc = pthread_mutex_lock(&a_mutex);
```

```
if (rc) { /* an error has occurred */
    perror("pthread_mutex_lock");
    pthread_exit(NULL);
}
/* mutex is now locked - wait on the condition variable */
/* if there are no requests to be handled.              */
rc = 0;
if (num_requests == 0)
    rc = pthread_cond_wait(&got_request, &request_mutex);
if (num_requests > 0 && rc == 0) { /* we have a request pending */
        /* do your stuff... */
        .
        .
        /* decrease count of pending requests */
        num_requests--;
    }
}
/* finally, unlock the mutex */
pthread_mutex_unlock(&request_mutex);
```

Using A Condition Variable - A Complete Example

As an example for the actual usage of condition variables, we will show a program that simulates the server we have described earlier - one thread, the receiver, gets client requests. It inserts the requests to a linked list, and a hoard of threads, the handlers, are handling these requests. For simplicity, in our simulation, the receiver thread creates requests and does not read them from real clients.

The program source is available in the file thread-pool-server.c, and contains many comments. Please read the source file first, and then read the following clarifying notes.

1. The 'main' function first launches the handler threads, and then performs the chord of the receiver thread, via its main loop.
2. A single mutex is used both to protect the condition variable, and to protect the linked list of waiting requests. This simplifies the design. As an exercise, you may think how to divide these roles into two mutexes.
3. The mutex itself MUST be a recursive mutex. In order to see why, look at the code of the 'handle_requests_loop' function. You will notice that it first locks the mutex, and afterwards calls the 'get_request' function, which locks the mutex again. If we used a non-recursive mutex, we'd get locked indefinitely in the mutex locking operation of the 'get_request' function.
   You may argue that we could remove the mutex locking in the 'get_request' function, and thus remove the double-locking problem, but this is a flawed design - in a larger program, we might call the 'get_request' function from other places in the code, and we'll need to check for proper locking of the mutex in each of them.
4. As a rule, when using recursive mutexes, we should try to make sure that each lock operation is accompanied by a matching unlock operation in the same function. Otherwise, it will be very hard to make sure that after locking the mutex several times, it is being unlocked the same number of times, and deadlocks would occur.
5. The implicit unlocking and re-locking of the mutex on the call to the `pthread_cond_wait()` function is confusing at first. It is best to add a comment regarding this behavior in the code, or else someone that reads this code might accidentally add a further mutex lock.
6. When a handler thread handles a reuqest - it should free the mutex, to avoid blocking all the other handler threads. After it finished handling the request, it sohuld lock the mutex again, and check if there are moer requests to handle.

"Private" thread data - Thread-Specific Data

In "normal", single-thread programs, we sometimes find the need to use a global variable. Ok, so good old teach' told us it is bad practice to have global variables, but they sometimes do come handy. Especially if they are static variables - meaning, they are recognized only on the scope of a single file.

In multi-threaded programs, we also might find a need for such variables. We should note, however, that the same variable is accessible from all the threads, so we need to protect access to it using a mutex, which is extra overhead. Further more, we sometimes need to have a variable that is 'global', but only for a specific thread. Or the same 'global' variable should have different values in different threads. For example, consider a program that needs to have one globally accessible linked list in each thread, but note the same list. Further, we want the same code to be executed by all threads. In this case, the global pointer to the start of the list should be point to a different address in each thread.

In order to have such a pointer, we need a mechanism that enables the same global variable to have a different location in memory. This is what the thread-specific data mechanism is used for.

Overview Of Thread-Specific Data Support

In the thread-specific data (TSD) mechanism, we have notions of keys and values. Each key has a name, and pointer to some memory area. Keys with the same name in two separate threads always point to different memory locations - this is handled by the library functions that allocate memory blocks to be accessed via these keys. We have a function to create a key (invoked once per key name for the whole process), a function to allocate memory (invoked separately in each thread), and functions to de-allocate this memory for a specific thread, and a function to destroy the key, again, process-wide. we also have functions to access the data pointed to by a key, either setting its value, or returning the value it points to.

Allocating Thread-Specific Data Block

The `pthread_key_create()` function is used to allocate a new key. This key now becomes valid for all threads in our process. When a key is created, the value it points to defaults to NULL. Later on each thread may change its copy of the value as it wishes. Here is how to use this function:

```
/* rc is used to contain return values of pthread functions */
int rc;
/* define a variable to hold the key, once created.         */
pthread_key_t list_key;
/* cleanup_list is a function that can clean up some data   */
/* it is specific to our program, not to TSD                */
extern void* cleanup_list(void*);

/* create the key, supplying a function that'll be invoked when it's deleted. */
rc = pthread_key_create(&list_key, cleanup_list);
```

Some notes:

1. After `pthread_key_create()` returns, the variable 'list_key' points to the newly created key.
2. The function pointer passed as second parameter to `pthread_key_create()`, will be automatically invoked by the pthread library when our thread exits, with a pointer to the key's value as its parameter. We may supply a NULL pointer as the function pointer, and then no function will be invoked for key. Note that the function will be invoked once in each thread, even thought we created this key only once, in one thread.
   If we created several keys, their associated destructor functions will be called in an arbitrary order, regardless of the order of keys creation.

3. If the `pthread_key_create()` function succeeds, it returns 0. Otherwise, it returns some error code.
4. There is a limit of `PTHREAD_KEYS_MAX` keys that may exist in our process at any given time. An attempt to create a key after `PTHREAD_KEYS_MAX` exits, will cause a return value of EAGAIN from the `pthread_key_create()` function.

Accessing Thread-Specific Data

After we have created a key, we may access its value using two pthread functions: `pthread_getspecific()` and `pthread_setspecific()`. The first is used to get the value of a given key, and the second is used to set the data of a given key. A key's value is simply a void pointer (void*), so we can store in it anything that we want. Lets see how to use these functions. We assume that 'a_key' is a properly initialized variable of type `pthread_key_t` that contains a previously created key:

```
/* this variable will be used to store return codes of pthread functions */
int rc;

/* define a variable into which we'll store some data */
/* for example, and integer.                          */
int* p_num = (int*)malloc(sizeof(int));
if (!p_num) {
    fprintf(stderr, "malloc: out of memory\n";
    exit(1);
}
/* initialize our variable to some value */
(*p_num) = 4;

/* now lets store this value in our TSD key.    */
/* note that we don't store 'p_num' in our key. */
/* we store the value that p_num points to.     */
rc = pthread_setspecific(a_key, (void*)p_num);

.
.
/* and somewhere later in our code... */
.
.
/* get the value of key 'a_key' and print it. */
{
    int* p_keyval = (int*)pthread_getspecific(a_key);

    if (p_keyval != NULL) {
        printf("value of 'a_key' is: %d\n", *p_keyval);
    }
}
```

Note that if we set the value of the key in one thread, and try to get it in another thread, we will get a NULL, since this value is distinct for each thread.

Note also that there are two cases where `pthread_getspecific()` might return NULL:

1. The key supplied as a parameter is invalid (e.g. its key wasn't created).
2. The value of this key is NULL. This means it either wasn't initialized, or was set to NULL explicitly by a previous call to `pthread_setspecific()`.

Deleting Thread-Specific Data Block

The `pthread_key_delete()` function may be used to delete keys. But do not be confused by this function's name: it does not delete memory associated with this key, nor does it call the destructor function defined during the key's creation. Thus, you still need to do memory cleanup on your own if you need to free this memory during runtime. However, since usage of global variables (and thus also thread-specific data), you usually don't need to free this memory until the thread terminate, in which case the pthread library will invoke your destructor functions anyway.

Using this function is simple. Assuming list_key is a `pthread_key_t` variable pointing to a properly created key, use this function like this:

```
int rc = pthread_key_delete(key);
```

the function will return 0 on success, or EINVAL if the supplied variable does not point to a valid TSD key.

A Complete Example

None yet. Give me a while to think of one..... sorry. All i can think of right now is 'global variables are evil'. I'll try to find a good example for the future. If you have a good example, please let me know.

Thread Cancellation And Termination

As we create threads, we need to think about terminating them as well. There are several issues involved here. We need to be able to terminate threads cleanly. Unlike processes, where a very ugly method of using signals is used, the folks that designed the pthreads library were a little more thoughtful. So they supplied us with a whole system of canceling a thread, cleaning up after a thread, and so on. We will discuss these methods here.

Canceling A Thread

When we want to terminate a thread, we can use the `pthread_cancel` function. This function gets a thread ID as a parameter, and sends a cancellation request to this thread. What this thread does with this request depends on its state. It might act on it immediately, it might act on it when it gets to a cancellation point (discussed below), or it might completely ignore it. We'll see later how to set the state of a thread and define how it acts on cancellation requests. Lets first see how to use the cancel function. We assume that 'thr_id' is a variable of type `pthread_id` containing the ID of a running thread:

```
pthread_cancel(thr_id);
```

The `pthread_cancel()` function returns 0, so we cannot know if it succeeded or not.

Setting Thread Cancellation State

A thread's cancel state may be modified using several methods. The first is by using the `pthread_setcancelstate()` function. This function defines whether the thread will accept cancellation requests or not. The function takes two arguments. One that sets the new cancel state, and one into which the previous cancel state is stored by the function. Here is how it is used:

```
int old_cancel_state;
```

```
pthread_setcancelstate(PTHREAD_CANCEL_DISABLE, &old_cancel_state);
```

This will disable canceling this thread. We can also enable canceling the thread like this:

```
int old_cancel_state;
pthread_setcancelstate(PTHREAD_CANCEL_ENABLE, &old_cancel_state);
```

Note that you may supply a NULL pointer as the second parameter, and then you won't get the old cancel state.

A similar function, named `pthread_setcanceltype()` is used to define how a thread responds to a cancellation request, assuming it is in the 'ENABLED' cancel state. One option is to handle the request immediately (asynchronously). The other is to defer the request until a cancellation point. To set the first option (asynchronous cancellation), do something like:

```
int old_cancel_type;
pthread_setcanceltype(PTHREAD_CANCEL_ASYNCHRONOUS, &old_cancel_type);
```

And to set the second option (deferred cancellation):

```
int old_cancel_type;
pthread_setcanceltype(PTHREAD_CANCEL_DEFERRED, &old_cancel_type);
```

Note that you may supply a NULL pointer as the second parameter, and then you won't get the old cancel type.

You might wonder - "What if i never set the cancellation state or type of a thread?". Well, in such a case, the `pthread_create()` function automatically sets the thread to enabled deferred cancellation, that is, `PTHREAD_CANCEL_ENABLE` for the cancel mode, and `PTHREAD_CANCEL_DEFERRED` for the cancel type.

Cancellation Points

As we've seen, a thread might be in a state where it does not handle cancel requests immediately, but rather defers them until it reaches a cancellation point. So what are these cancellation points?

In general, any function that might suspend the execution of a thread for a long time, should be a cancellation point. In practice, this depends on the specific implementation, and how conformant it is to the relevant POSIX standard (and which version of the standard it conforms to...). The following set of pthread functions serve as cancellation points:

- `pthread_join()`
- `pthread_cond_wait()`
- `pthread_cond_timedwait()`
- `pthread_testcancel()`
- `sem_wait()`
- `sigwait()`

This means that if a thread executes any of these functions, it'll check for deferred cancel requests. If there is one, it will execute the cancellation sequence, and terminate. Out of these functions, `pthread_testcancel()` is unique - it's only purpose is to test whether a cancellation request is pending for this thread. If there is, it executes the cancellation

sequence. If not, it returns immediately. This function may be used in a thread that does a lot of processing without getting into a "natural" cancellation state.

Note: In real conformant implementations of the pthreads standard, normal system calls that cause the process to block, such as `read()`, `select()`, `wait()` and so on, are also cancellation points. The same goes for standard C library functions that use these system calls (the various printf functions, for example).

Setting Thread Cleanup Functions

One of the features the pthreads library supplies is the ability for a thread to clean up after itself, before it exits. This is done by specifying one or more functions that will be called automatically by the pthreads library when the thread exits, either due to its own will (e.g. calling `pthread_exit()`), or due to it being canceled.

Two functions are supplied for this purpose. The `pthread_cleanup_push()` function is used to add a cleanup function to the set of cleanup functions for the current thread. The `pthread_cleanup_pop()` function removes the last function added with `pthread_cleanup_push()`. When the thread terminates, its cleanup functions are called in the reverse order of their registration. So the the last one to be registered is the first one to be called.

When the cleanup functions are called, each one is supplied with one parameter, that was supplied as the second parameter to the `pthread_cleanup_push()` function call. Lets see how these functions may be used. In our example we'll see how these functions may be used to clean up some memory that our thread allocates when it starts running.

```
/* first, here is the cleanup function we want to register.        */
/* it gets a pointer to the allocated memory, and simply frees it. */
void
cleanup_after_malloc(void* allocated_memory)
{
    if (allocated_memory)
        free(allocated_memory);
}

/* and here is our thread's function.      */
/* we use the same function we used in our */
/* thread-pool server.                     */
void*
handle_requests_loop(void* data)
{
    .
    .
    /* this variable will be used later. please read on...         */
    int old_cancel_type;

    /* allocate some memory to hold the start time of this thread. */
    /* assume MAX_TIME_LEN is a previously defined macro.          */
    char* start_time = (char*)malloc(MAX_TIME_LEN);

    /* push our cleanup handler. */
    pthread_cleanup_push(cleanup_after_malloc, (void*)start_time);
    .
    .
    /* here we start the thread's main loop, and do whatever is desired.. */
    .
    .
```

```
    .

    /* and finally, we unregister the cleanup handler. our method may seem */
    /* awkward, but please read the comments below for an explanation.     */

    /* put the thread in deferred cancellation mode.      */
    pthread_setcanceltype(PTHREAD_CANCEL_DEFERRED, &old_cancel_type);

    /* supplying '1' means to execute the cleanup handler */
    /* prior to unregistering it. supplying '0' would     */
    /* have meant not to execute it.                      */
    pthread_cleanup_pop(1);

    /* restore the thread's previous cancellation mode.   */
    pthread_setcanceltype(old_cancel_type, NULL);
}
```

As we can see, we allocated some memory here, and registered a cleanup handler that will free this memory when our thread exits. After the execution of the main loop of our thread, we unregistered the cleanup handler. This must be done in the same function that registered the cleanup handler, and in the same nesting level, since both `pthread_cleanup_pop()` and `pthread_cleanup_pop()` functions are actually macros that add a '{' symbol and a '}' symbol, respectively.

As to the reason that we used that complex piece of code to unregister the cleanup handler, this is done to assure that our thread won't get canceled in the middle of the execution of our cleanup handler. This could have happened if our thread was in asynchronous cancellation mode. Thus, we made sure it was in deferred cancellation mode, then unregistered the cleanup handler, and finally restored whatever cancellation mode our thread was in previously. Note that we still assume the thread cannot be canceled in the execution of `pthread_cleanup_pop()` itself - this is true, since `pthread_cleanup_pop()` is not a cancellation point.

Synchronizing On Threads Exiting

Sometimes it is desired for a thread to wait for the end of execution of another thread. This can be done using the `pthread_join()` function. It receives two parameters: a variable of type `pthread_t`, denoting the thread to be joined, and an address of a `void*` variable, into which the exit code of the thread will be placed (or `PTHREAD_CANCELED` if the joined thread was canceled).
The `pthread_join()` function suspends the execution of the calling thread until the joined thread is terminated.

For example, consider our earlier thread pool server. Looking back at the code, you'll see that we used an odd `sleep()` call before terminating the process. We did this since the main thread had no idea when the other threads finished processing all pending requests. We could have solved it by making the main thread run a loop of checking if no more requests are pending, but that would be a busy loop.

A cleaner way of implementing this, is by adding three changes to the code:

1. Tell the handler threads when we are done creating requests, by setting some flag.
2. Make the threads check, whenever the requests queue is empty, whether or not new requests are supposed to be generated. If not, then the thread should exit.
3. Make the main thread wait for the end of execution of each of the threads it spawned.

The first 2 changes are rather easy. We create a global variable named 'done_creating_requests' and set it to '0' initially. Each thread checks the contents of this variable every time before it intends to go to wait on the condition variable (i.e. the requests queue is empty).
The main thread is modified to set this variable to '1' after it finished generating all requests. Then the condition variable is being broadcast, in case any of the threads is waiting on it, to make sure all threads go and check the 'done_creating_requests' flag.

The last change is done using a `pthread_join()` loop: call `pthread_join()` once for each handler thread. This way, we know that only after all handler threads have exited, this loop is finished, and then we may safely terminate the process. If we didn't use this loop, we might terminate the process while one of the handler threads is still handling a request.

The modified program is available in the file named thread-pool-server-with-join.c. Look for the word 'CHANGE' (in capital letters) to see the locations of the three changes.

Detaching A Thread

We have seen how threads can be joined using the `pthread_join()` function. In fact, threads that are in a 'join-able' state, must be joined by other threads, or else their memory resources will not be fully cleaned out. This is similar to what happens with processes whose parents didn't clean up after them (also called 'orphan' or 'zombie' processes).

If we have a thread that we wish would exit whenever it wants without the need to join it, we should put it in the detached state. This can be done either with appropriate flags to the `pthread_create()` function, or by using the `pthread_detach()` function. We'll consider the second option in our tutorial.

The `pthread_detach()` function gets one parameter, of type `pthread_t`, that denotes the thread we wish to put in the detached state. For example, we can create a thread and immediately detach it with a code similar to this:

```
pthread_t a_thread;   /* store the thread's structure here             */
int rc;               /* return value for pthread functions.           */
extern void* thread_loop(void*); /* declare the thread's main function. */

/* create the new thread. */
rc = pthread_create(&a_thread, NULL, thread_loop, NULL);

/* and if that succeeded, detach the newly created thread. */
if (rc == 0) {
    rc = pthread_detach(a_thread);
}
```

Of-course, if we wish to have a thread in the detached state immediately, using the first option (setting the detached state directly when calling `pthread_create()` is more efficient.

Threads Cancellation - A Complete Example

Our next example is much larger then the previous examples. It demonstrates how one could write a multi-threaded program in C, in a more or less clean manner. We take our previous thread-pool server, and enhance it in two ways. First, we add the ability to tune the number of handler threads based on the requests load. New threads are created if the requests queue becomes too large, and after the queue becomes shorter again, extra threads are canceled.

Second, we fix up the termination of the server when there are no more new requests to handle. Instead of the ugly sleep we used in our first example, this time the main thread waits for all threads to finish handling their last requests, by joining each of them using `pthread_join()`.

The code is now being split to 4 separate files, as follows:

1. requests_queue.c - This file contains functions to manipulate a requests queue. We took the `add_request()` and `get_request()` functions and put them here, along with a data structure that contains all the variables previously defined as globals - pointer to queue's head, counter of requests, and even pointers to the queue's mutex and condition variable. This way, all the manipulation of the data is done in a single file, and all its functions receive a pointer to a 'requests_queue' structure.

2. handler_thread.c - this contains the functions executed by each handler thread - a function that runs the main loop (an enhanced version of the 'handle_requests_loop()' function, and a few local functions explained below). We also define a data structure to collect all the data we want to pass to each thread. We pass a pointer to such a structure as a parameter to the thread's function in the `pthread_create()` call, instead of using a bunch of ugly globals: the thread's ID, a pointer to the requests queue structure, and pointers to the mutex and condition variable to be used.
3. handler_threads_pool.c - here we define an abstraction of a thread pool. We have a function to create a thread, a function to delete (cancel) a thread, and a function to delete all active handler threads, called during program termination. we define here a structure similar to that used to hold the requests queue, and thus the functions are similar. However, because we only access this pool from one thread, the main thread, we don't need to protect it using a mutex. This saves some overhead caused by mutexes. the overhead is small, but for a busy server, it might begin to become noticeable.
4. main.c - and finally, the main function to rule them all, and in the system bind them. This function creates a requests queue, creates a threads pool, creates few handler threads, and then starts generating requests. After adding a request to the queue, it checks the queue size and the number of active handler threads, and adjusts the number of threads to the size of the queue. We use a simple water-marks algorithm here, but as you can see from the code, it can be easily be replaced by a more sophisticated algorithm. In our water-marks algorithm implementation, when the high water-mark is reached, we start creating new handler threads, to empty the queue faster. Later, when the low water-mark is reached, we start canceling the extra threads, until we are left with the original number of handler threads.

After rewriting the program in a more manageable manner, we added code that uses the newly learned pthreads functions, as follows:

1. Each handler thread created puts itself in the deferred cancellation mode. This makes sure that when it gets canceled, it can finish handling its current request, before terminating.
2. Each handler thread also registers a cleanup function, to unlock the mutex when it terminates. This is done, since a thread is most likely to get canceled when calling `pthread_cond_wait()`, which is a cancellation point. Since the function is called with the mutex locked, it might cause the thread to exit and cause all other threads to 'hang' on the mutex. Thus, unlocking the mutex in a cleanup handler (registered with the `pthread_cleanup_push()` function) is the proper solution.
3. Finally, the main thread is set to clean up properly, and not brutally, as we did before. When it wishes to terminate, it calls the 'delete_handler_threads_pool()' function, which calls `pthread_join` for each remaining handler thread. This way, the function returns only after all handler threads finished handling their last request.

Please refer to the source code for the full details. Reading the header files first will make it easier to understand the design. To compile the program, just switch to the thread-pool-server-changes directory, and type 'gmake'.

Exercise: our last program contains some possible race condition during its termination process. Can you see what this race is all about? Can you offer a complete solution to this problem? (hint - think of what happens to threads deleted using 'delete_handler_thread()').

Exercise 2: the way we implement the water-marks algorithm might come up too slow on creation of new threads. Try thinking of a different algorithm that will shorten the average time a request stays on the queue until it gets handled. Add some code to measure this time, and experiment until you find your "optimal pool algorithm". Note - Time should be measured in very small units (using the `getrusage` system call), and several runs of each algorithm should be made, to get more accurate measurements.

Using Threads For Responsive User Interface Programming

One area in which threads can be very helpful is in user-interface programs. These programs are usually centered around a loop of reading user input, processing it, and showing the results of the processing. The processing part may

sometimes take a while to complete, and the user is made to wait during this operation. By placing such long operations in a seperate thread, while having another thread to read user input, the program can be more responsive. It may allow the user to cancel the operation in the middle.

In graphical programs the problem is more severe, since the application should always be ready for a message from the windowing system telling it to repaint part of its window. If it's too busy executing some other task, its window will remain blank, which is rather ugly. In such a case, it is a good idea to have one thread handle the message loop of the windowing systm and always ready to get such repain requests (as well as user input). When ever this thread sees a need to do an operation that might take a long time to complete (say, more then 0.2 seconds in the worse case), it will delegate the job to a seperate thread.

In order to structure things better, we may use a third thread, to control and synchronize the user-input and task-performing threads. If the user-input thread gets any user input, it will ask the controlling thread to handle the operation. If the task-performing thread finishes its operation, it will ask the controlling thread to show the results to the user.

User Interaction - A Complete Example

As an example, we will write a simple character-mode program that counts the number of lines in a file, while allowing the user to cancel the operation in the middle.

Our main thread will launch one thread to perform the line counting, and a second thread to check for user input. After that, the main thread waits on a condition variable. When any of the threads finishes its operation, it signals this condition variable, in order to let the main thread check what happened. A global variable is used to flag whether or not a cancel request was made by the user. It is initialized to '0', but if the user-input thread receives a cancellation request (the user pressing 'e'), it sets this flag to '1', signals the condition variable, and terminates. The line-counting thread will signal the condition variable only after it finished its computation.

Before you go read the program, we should explain the use of the `system()` function and the 'stty' Unix command. The `system()` function spawns a shell in which it executes the Unix command given as a parameter. The `stty` Unix command is used to change terminal mode settings. We use it to switch the terminal from its default, line-buffered mode, to a character mode (also known as raw mode), so the call to `getchar()` in the user-input thread will return immediatly after the user presses any key. If we hadn't done so, the system will buffer all input to the program until the user presses the ENTER key. Finally, since this raw mode is not very useful (to say the least) once the program terminates and we get the shell prompt again, the user-input thread registers a cleanup function that restores the normal terminal mode, i.e. line-buffered. For more info, please refer to stty's manual page.

The program's source can be found in the file line-count.c. The name of the file whose lines it reads is hardcoded to 'very_large_data_file'. You should create a file with this name in the program's directory (large enough for the operation to take enough time). Alternatively, you may un-compress the file 'very_large_data_file.Z' found in this directory, using the command:

```
uncompress very_large_data_file.Z
```

note that this will create a 5MB(!) file named 'very_large_data_file', so make sure you have enough free disk-space before performing this operation.

Using 3rd-Party Libraries In A Multi-Threaded Application

One more point, and a very important one, should be taken by programmers employeeing multi-threading in their programs. Since a multi-threaded program might have the same function executed by different threads at the same time, one must make sure that any function that might be invoked from more then one thread at a time, is MT-safe (Multi-Thread Safe). This means that any access to data structures and other shared resources is protected using mutexes.

It may be possibe to use a non-MT-safe library in a multi-threaded programs in two ways:

1. Use this library only from a single thread. This way we are assured that no function from the library is executed simultanously from two seperate threads. The problem here is that it might limit your whole design, and might force you to add more communications between threads, if another thread needs to somehow use a function from this library.
2. Use mutexes to protect function calls to the library. This means that a single mutex is used by any thread invoking any function in this library. The mutex is locked, the function is invoked, and then the mutex is unlocked. The problem with this solution is that the locking is not done in a fine granularity - even if two functions from the library do not interfere with each other, they still cannot be invoked at the same time by seperate threads. The second thread will be blocked on the mutex until the first thread finishes the function call. You might call for using seperate mutexes for unrelated functions, but usually you've no idea how the library really works and thus cannot know which functions access the same set of resources. More then that, even if you do know that, a new version of the library might behave differently, forcing you to modify your whole locking system.

As you can see, non-MT-safe libraries need special attention, so it is best to find MT-safe libraries with a similar functionality, if possible.

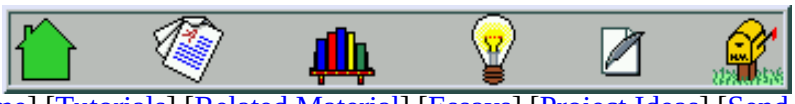
Using A Threads-Aware Debugger

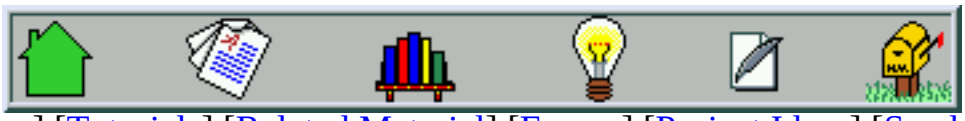
One last thing to note - when debugging a multi-threaded application, one needs to use a debugger that "sees" the threads in the program. Most up-to-date debuggers that come with commercial development environments are thread-aware. As for Linux, gdb as is shiped with most (all?) distributions seems to be not thread-aware. There is a project, called 'SmartGDB', that added thread support to gdb, as well as a graphical user interface (which is almost a must when debugging multi-threaded applications). However, it may be used to debug only multi-threaded applications that use the various user-level thread libraries. Debugging LinuxThreads with SmartGDB requires applying some kernel patches, that are currently available only for Linux kernels from the 2.1.X series. More information about this tool may be found at http://hegel.ittc.ukans.edu/projects/smartgdb/. There is also some information about availability of patches to the 2.0.32 kernel and gdb 4.17. This information may be found on the LinuxThreads homepage.

Side-Notes

water-marks algorithm

> An algorithm used mostly when handling buffers or queues: start filling in the queue. If its size exceeds a threshold, known as the high water-mark, stop filling the queue (or start emptying it faster). Keep this state until the size of the queue becomes lower then another threshold, known as the low water-mark. At this point, resume the operation of filling the queue (or return the emptying speed to the original speed).



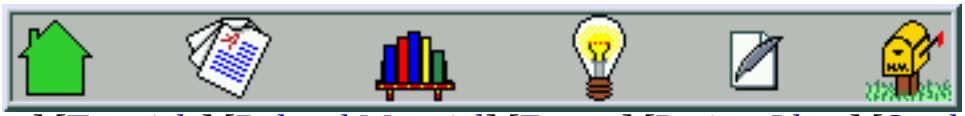


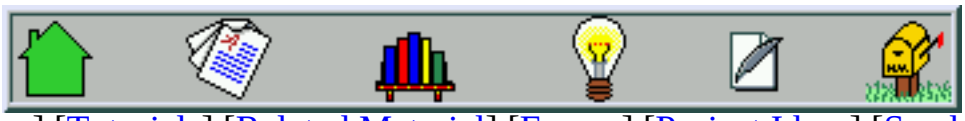
Essays About Computer Programming

This part will be used to write essays about computer programming issues i find interest in. Feel free to comment about these essays - some of them are written in purpose in a way that will attract reactions - hopefully constructive reactions.

Essays Index (currently there's only one, but more are in the works):

1. Software Engineering
    1. Why Do Universities Fail Teaching Software Engineering?



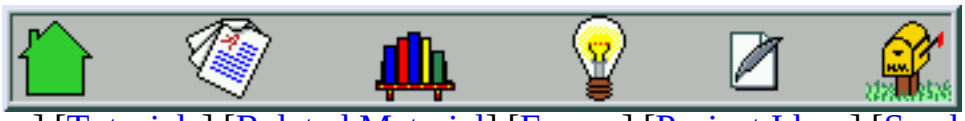


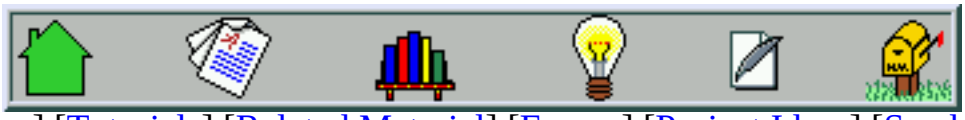
Ideas For Little Learning Projects

Reading tutorials is not enough. One needs to experiment with example programs, and write new programs of their own, in order to really learn. The following is a short list of ideas for little learning projects, that you could try to implement. The proposals are indexed in a fashion similar to that of the tutorials, so it will be possible to try them out after reading the relevant tutorial. Each proposal include a goal, some background, and a design description. The designs in the beginners section are very detailed, and they get less detailed in the more advanced levels.

Project Proposals Index:

1. Unix Beginners
    1. Simple Configuration File Parser (file operations)
    2. Web Server "Visitor Tracking" Log Analyzer (file operations, algorithms)
    3. Log File Rotator (file and directory operations)
2. Intermediate Level
    1. Implementing Hash Tables (algorithms)
    2. Writing A VT100-Based PacMan Game (signals, timers)
    3. Implementing A Simple File-Transfer Client And Server (sockets)
3. Advanced Topics
    1. Tiny Multi-Threaded HTTP Server (pthreads, sockets)

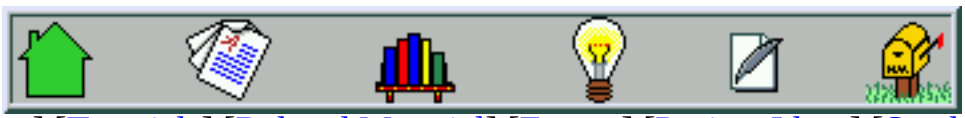


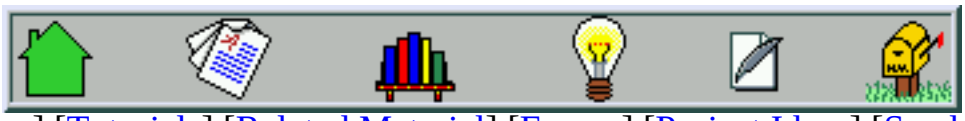


Related Sites And Material

The following list is split into two parts. The first contains links (non-broken, i hope) to sites on the Internet containing material related to Unix programming. The second contains references to other source of wisdom, mostly books.

- Related Sites
  - A Unix Programming FAQ
  - The Unix Socket Programming FAQ
  - Raw IP Networking FAQ
  - The comp.programming.threads FAQ
  - The LinuxThreads Homepage
- Related Material
  - Unix System Programming Books:
    - Advanced Programming in the Unix Environment - W. Richard Stevens. a through coverage of Unix system calls related to I/O, file system access, process control, signals and inter-process communication. ~750 pages.
    - Unix Network Programming, volume 1 - W. Richard Stevens. a great resource that covers almost every aspect of Unix network programming, and does not hide the flaws of the system. A very heavy book (~1000 pages).
  - General Programming Languages Books:
    - The C Programming Language, 2nd ed. - Brian Kernighan and Dennis Ritchie.
      the "bible" for C programmers, written by the creators of the language. A small-sized book (~270 pages), but strait to the point.
    - The C++ Programming Language, 3rd ed. - Bjarne Stroustroup.
      Likewise, for the C++ language. contains updates for the final C++ language draft. A very heavy book (~700 pages).





Tutorials

The following set of tutorials reflects an effort to give Unix programmers and programmers wanna-be a chance to get familiar with various aspects of programming on Unix-like systems, without the need to buy an expensive set of books and spending a lot of time in understanding lots of technical material. The one assumption common to all tutorials (unless stated otherwise) is that you already know C programming on any system.

The general intention is to allow someone to get familiar with a subject rather quickly, so they can start experimenting with it, and allow them to read a more thorough user manual or reference manual after they got over the initial "fear". By no means will these tutorials suffice to turn anyone into a proficient professional, but one needs to start somewhere and then again, why not do it for free?

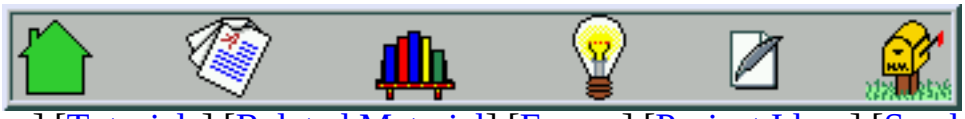
Tutorials Index (note - each tutorial may be browsed online, or downloaded as a .tar.gz archive). Size of each tutorial is given in ammount of screen-pages when viewed using the lynx text-based web browser (assuming 25 lines per page):

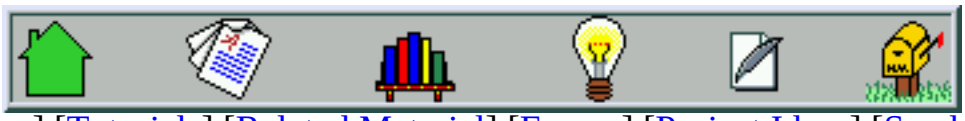
1. Unix Beginners
    1. Compiling C/C++ Programs On Unix (archive) (~15 lynx pages)
    2. Debugging With "gdb" (archive) (~11 lynx pages)
    3. Automating Program Compilation Using Makefiles (archive) (~13 lynx pages)
    4. Manipulating Files And Directories In Unix (archive) (~50 lynx pages)
2. Intermediate Level
    1. Creating And Using C Libraries (archive) (~18 lynx pages)
    2. Unix Signals Programming (archive) (~27 lynx pages)
    3. Internetworking With Unix Sockets (archive) (~21 + ~44 lynx pages)
    4. Accessing User Information On A Unix System (archive) (~38 lynx pages)
    5. Graphics Programming
        1. Basic Graphics Programming With The Xlib Library (archive) (~59 + ~44 lynx pages)
3. Advanced Topics
    1. Parallel Programming - Basic Theory For The Unwary (archive) (~29 lynx pages)
    2. Multi-Threaded Programming With The Pthreads Library (archive) (~60 lynx

pages)

3. Multi-Process Programming Under Unix (archive) (~80 lynx pages)

Note: The levels mentioned here are just to give one a basic idea of what they are sticking their head into, before they delve into the tutorial. Your experience will vary based on your theoretical background, and your experience (for example, an experienced programmer coming from another platform will probably find most of these tutorials to be rather easy).





Little Unix Programmers Group (LUPG)'s Little Site

(Check the change log for the latest changes to this site.)

The LUPG's little site is dedicated to people who want to learn about Unix programming. Is is intended to form a little community of people who will focus on learning various programming topics, and then teach others.

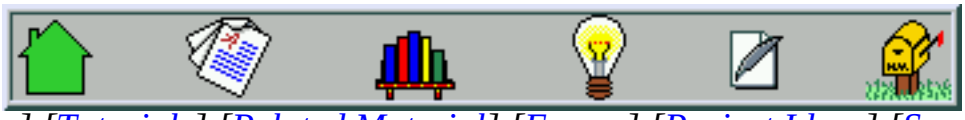
The majority of the site is dedicated to a set of tutorials, each dedicated to one issue in Unix programming. The tutorials are divided to small groups of inter-related categories, and all can be either browsed online, or downloaded to your computer and viewed locally.

Other sections of the site include project ideas and a new section with essays about software engineering issues.

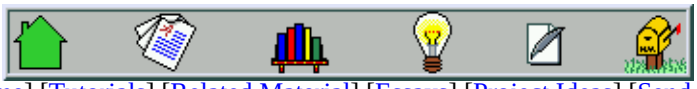
There is also the unavoidable page of info about related sites and material.

If you want to get automatic e-mail notifications about new tutorials or changes to this site, you may use the automatic e-mail notification form.

Note: This site is presented as a spare-time activity, in the hope that it will be useful, but without any expressed or implied warranty. See the standard disclaimer for more info.



This site is generated using the power of vim, perl, and gnu make.



Simple Configuration File Parser

Goal

1. Writing a set of functions that allow a program to read a simple configuration file, check which options exist in the file, and what are their values.
2. Extend the set of functions to allow the program to also modify the contents of the configuration file.

Background

Quite often we write applications that have a large amount of options the user can configure. To ease the user's life, we allow her to write a small configuration file, that our application will read when it starts running, and act upon. If we are not too lazy, we also write some interface that lets the user modify the configuration, and the configuration file is updated automatically.

Details

3. API. The following set of functions should be written, which will be used as an interface for the programmer. Other functions may be added in order to make your code more readable and flexible.

    1. `struct config_data* parse_config(char* file_name);`

        Open the given configuration file (located in the user's home directory), parse its information, and return a structure of type "config_data", or NULL if an error occurred. Print out any error messages, e.g. file not found, file format errors, etc.

    2. `char* get_config_variable(struct config_data* conf_data, char* name);`

        Return the contents of the given variable from the given configuration data, or NULL if no such variable is defined in the configuration data.

    3. `void delete_config_data(struct config_data* conf_data);`

        delete the configuration data found in the given structure, freeing any memory associated with this structure.

Extended API to support configuration file updating. Note that supporting this API would require a more complicated data structure for "struct config_data".

1. `int update_config_variable(struct config_data* conf_data, char* name, char* value);`

   Update the given configuration variable in "conf_data" to have the given contents. Return '1' on success, or '0' if the variable was not found in "conf_data".

2. `int add_config_variable(struct config_data* conf_data, char* name, char* value);`

   Add a new configuration variable to "conf_data", with the given contents. Return '1' on success, or '0' if the variable was already found in "conf_data".

3. `int delete_config_variable(struct config_data* conf_data, char* name);`

   Delete the given configuration variable from "conf_data". Return '1' on success, or '0' if the variable was not found in "conf_data".

4. `int write_config(struct config_data* conf_data, char* file_name);`

   Write the contents of the given configuration data into the given file. If the file does not exist, create it and emit the contents. If the file already exists, only update the new, modified and deleted variables, preserving any comments that might have existed in the file, and preserving the order of its lines. New variables should be appended to the end of the file.

.

```
char* home = getenv("HOME");
if (!home) {
    fprintf(stderr, "Error: Environment variable "HOME" not defined!\n");
    exit(1);
}
```

.

```
    Now variable 'home' points to the contents of the HOME environment
    variable.
```

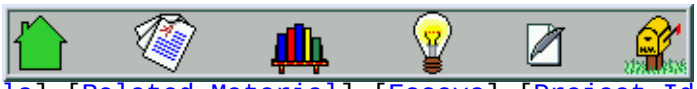
1. The configuration file resides in the home directory of the user running the program. Finding the full path to this directory will be done by reading the environment variable "HOME" using the `getenv()` function, as follows:

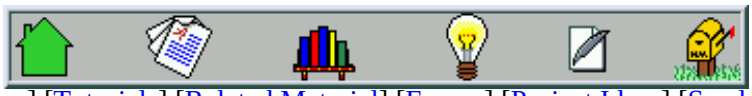
2. Configuration file format. The file is made of a set of independent lines. Each line may have any of the following formats:

   - Lines starting with the '#' sign are comments, and should be ignored.
   - Empty lines should be ignored as well.
   - A line with a format of

     ```
     name = value
     ```

     defines a configuration variable named "name", whose contents is "value". Leading and trailing white space characters (spaces, tabs) on the line should be ignored. The "value" part is optional, and if missing, the "name" variable contains the empty string.
   - If a variable is defined twice in the file, an error message should be printed, stating the number of the faulty line and its contents.
   - A line that is not a comment and does not contain the "=" character should be printed out with a "bad syntax" error message (again, including the line number and the line's contents).
   - To make life easier, assume the file may not contain more then 100 variables (i.e. use arrays to hold the variables, not linked lists or other less trivial data structures).





Web Server "Visitor Tracking" Log Analyzer

Goal

1. Writing a program that parses the contents of a web server's log file, and tries to deduce information about the "visitors", based on groups of "hits" from a single IP address.

Background

Any web server today keeps a log file into which it writes a line for each file that was served to a client. Each log record contains the URL of the file, date and time of the retrieval, the IP address from which the client connected, and possibly some more data.
One of the problems people face when trying to analyze this data, is understanding how many people actually entered their site, and what each visitor exactly looked for in their site.

Of-course, the log file never allows us to get accurate data (sometimes an IP address stands for a proxy server representing many users, sometimes people re-load the same page more then once, and so on), but we may deduce some general data that will give us a "feeling" of what's going on. We can crudely treat all requests coming from the same IP address in a short period of time, as representing a single user, and treat them accordingly. If we see another access from this IP address after a long period of time (say, more then half an hour), we may assume it's a new user that happens to use the same IP address.

Details

1. The log file is kept in the default format that apache uses:

   ```
   192.167.44.5 - - [30/Dec/1996:08:20:52 +0200] www.actcom.co.il "GET
   /~choo/bullet.gif HTTP/1.0" 200 873
   ```

   Each line contains the following fields, separated by spaces:
   1. Client's IP Address.
   2. User name (or '-' if no password was used).
   3. ...
   4. Date, time and time-zone (all between a pair of brackets, contains a space between the time and time-zone).
   5. Address of web server.
   6. Request type (GET or POST), Partial URL and Protocol Version (all between a pair of double-quotes, separated by spaces).
   7. HTTP return code (200 denotes successful request, 3XX denotes a re-located document, 4XX denotes authentication/access restriction errors, 5XX denotes general server configuration error).
   8. Size of data file sent back to the client.

   As an example, here is the LUPG's November 1998 log file, that you may use for testing your program.

2. Program's Output. The program should print out the following data:
   - Number of users that accessed the site.
   - Average number of requests per user.
   - Average total size (in bytes) of retrieved files per user.

- A list of requests made by each user, with the users sorted based on the time of their first request.

3. Log file may be very long. We should not make assumptions in our program regarding the size of the log file, and thus should allocate memory dynamically as required.

4. Data Structures Used. In order to allow easy collection of data, we will define a structure to hold data for a specific client, and an array to hold data for all clients. Here is how our data structures might look:

```
/* constants definitions */
#define MAX_TIME_LEN 26
#define MAX_ADDR_LEN 50
#define MAX_CONCURRENT_CLIENTS 200

/* definition of a single request's data */
struct request {
    char *file;                /* full path to requested data file.      */
    char time[MAX_TIME_LEN];   /* time when request was made.            */
    struct request* next;      /* pointer to next request, NULL if none. */
};

/* definition of a single client's requests list */
struct client {
    char address[MAX_ADDR_LENGTH]; /* address from which client connected */
    struct request* requests;      /* linked list of client's requests   */
};

/* currently active clients array */
int num_clients = 0;
struct client clients[MAX_CONCURRENT_CLIENTS];
```

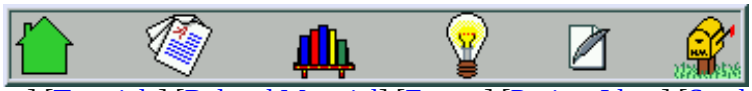
   We will make the assumption that the delay between two consecutive requests of a given client may not be more then half an hour. This allows us to keep a rather small array of clients. We will see how this helps us when we discuss the scanning algorithm. We also make the assumption that an entry in the clients array whose address field is an empty string, is free for re-use.

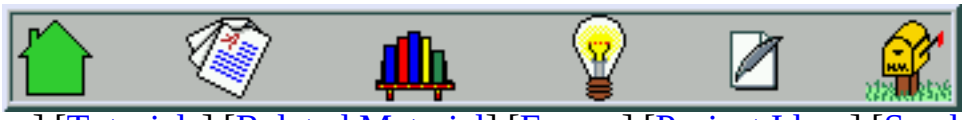
5. Algorithm. The algorithm goes as follows:
   1. Initialize the address fields of all entries in the clients array to an empty string.
   2. While log file contains more lines.
      1. Read next line from file, parse to fields, and store in line structure.
      2. Look for an entry in the clients array whose address matches that of the new request. If there is none - create a new entry at the first empty location in the array.
      3. Create a new request entry, and add it to the beginning of the linked list of the client's requests.
      4. Scan all the entries in the clients array. For each entry where the time field shows a time more then half an hour earlier then that of the request we just read:
         1. Print out the list of requests, from last to first! (since we stored them in reverse order).
         2. Free all memory used by the client's requests, and set the address field of that client to an empty string (for re-use).
   3. Print out data about the remaining clients.

   Note that we print out the data only loosely based on the time of the request. We live it up to you to fix the algorithm to print the data sorted by the time of the first request made by each client.

6. Additions. One problem with our data structure is its slow lookup method. A site having many simultaneous visitors will have its log file parsed very slow by our program. By using a move advanced data lookup method (such as hash tables,

we can speed up the whole program. Another problem is the usage of a lot of memory allocation. Other additions may be better configure-ability (e.g. allow the user to change the "half hour rotation factor" using a command line option, allow to also check how much data transfer each client has made, allow checking only fetches of files of a specific type.).





Log File Rotator

Goal

1. Writing a program that can rotate log files, keeping a given number of old log files.

Background

Many Unix servers and programs tend to maintain a log file, into which they write data about operations they perform and their success or failure. For example, an ftp server will write each login made through it, and possibly which files were erased via the server. The log data can later be used to analyze security problems, or server malfunctions. These log files tend to grow, and unless truncated, might fill up the disk.

The most common way of handling log file growth is via a rotation procedure. We first define how many old copies of the log file we want to create, and the rate at which we want to rotate them (once a day, once a week and so on). When we rotate a given log file, we rename the original file to have a '.1' suffix, rename the old '.1' file to have a '.2' suffix, and so on. The last copy of the log file (with suffix '.n') is being erased. of-course, we do this renaming from last to first, otherwise we will end up with 'n' copies of the current log file. Once done, we create a new, empty log file, with the same access permissions as the original log file.

Often the log file is kept open by the server writing into it, so we need to restart the server, so it will start writing into the new log file. However, we will ignore this issue for this exercise.

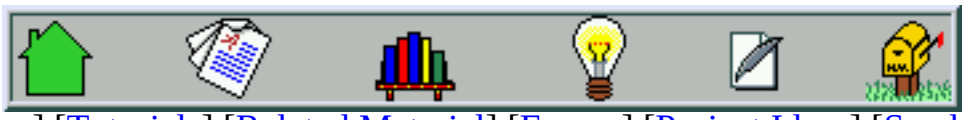
Details

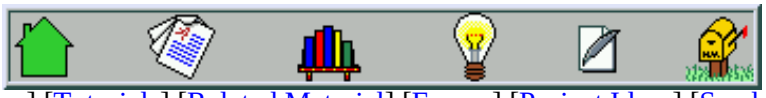
1. Rotation Procedure.

   Write a function named 'rotate', that gets a file name and the number of old copies to maintain as its parameters. This function will do the file renaming according to the algorithm shown earlier. It will assume that the log file is in the current directory. Remember to handle the case where less then 'n' old copies of the log file exist. In such cases, the number of old copies should grow by one.

2. Main Function.

Write the main function, that reads three parameters from the command line: full path of the logs directory, name of the log file to rotate, and number of old copies to maintain. The main function will check that the directory exists, and that we have both read and write access permission to the directory. It will then switch to the logs directory, and invoke the 'rotate' function to do the actual job.





Implementing Hash Tables

Goal

1. Writing a small library that implements a hash table (a structure used for very efficient lookup operations).

Background

Many programs are required to store some data in memory, and later on to look it up. Many types of data structures were formed to handle these operations (such as arrays, linked lists, binary search trees, etc), with varying levels of complexity, and a varying set of operations supported (adding items, deleting items, sorting items, locating items, finding the largest item and so on). One of the genuine data structure used for purposes of very efficient data lookup is the hash table.

A hash table may contain objects that have keys, and values. The hash table is made of an array into which objects may be inserted, and a hashing function. The hashing function takes a key as an argument, and uses it to calculate some index in the range of the table's array. This function is used to map different elements to different arrays in the table.

When we want to insert an element in to the hash table, we calculate its hash value (the result of activating the hash function on the element's key), and then go to the place in the array whose index equals this hash value, and insert the item there. If this location in the table is already in use, we look for the nearest free location and place the element there (there are various other variations for how to handle such "collisions". the method we describe here is called "simple hashing).

When we want to locate an element in the array (given its key), we calculate its hash value, and then start scanning the table at the location whose index matches this hash value. If we find the element, we are done. If we find a different element, we move to the next higher location (moving back to the beginning of the array if we are at the last element). We keep scanning until we either find the element, or find an empty cell. Such an empty cell denotes that the element was not found in the table, and thus the search fails.

When we want to delete an item from the hash table, we first locate it using the search algorithm. When we find it, we mark that location as free, not as empty. Why is that? this will let the search algorithm go past this freed item, instead of stopping there. Lets see an example. Suppose that we have a hash table made of 6 cells, and we insert three items 'A', 'B' and 'C', all of which have '3' as their hash value. The first ('A') will be inserted into cell '3', the second ('B') to cell '4' (since '3' is now in use), and the third ('C') will go into cell '5' (since both cells '3' and '4' are now in use):

```
-------------------------
|   |   | a | b | c |   |
-------------------------
```

If we try to search for element 'C' now, we calculate its hash value, go look for it in location 2 (that contains 'A'), and due to the mismatch start scanning to the right, skipping 'B', and eventually landing on 'C'. Now lets try to remove element 'B'. If we marked its location as empty, the table would now look like this:

```
-------------------------
|   |   | a |   | c |   |
-------------------------
```

If we try to search for 'C' again, we calculate its hash value (2, as you probably remember), find element 'A' (no match) and the next cell is empty, so we wrongly declare that 'C' was not found. Now, if when deleting 'B' we would mark it as 'free' instead of as 'empty', the table would look like this:

```
-------------------------
|   |   | a | * | c |   |
```

```
-------------------------
```

(* denotes a free cell). If we go to search for 'C' again, we land on 'A' (no match), go to the next cell (free cell, so no match, but keep on looking), and finally land on 'C' - a match was found.

One last saying goes to the efficiency of the insert, search and lookup operations on a hash table. This is a function of two factors: the density of the hash table (i.e. what percent of its cells actually contain elements), and the quality of the hashing function, in the sense of how well it spreads the hashing values of its elements. For example, if the hashing function would always return the value '1' for any element, then it's obvious to see that its associated hash table acts just like a linked list, which is not very efficient for search operations (in average, you will need to scan half the table's cells in order to find an item, and as the amount of elements grows, the search gets longer). Now, assuming a good hash function, it was found the if the table is no more then 50% filled up, finding an element in the table takes about a single operation, while finding that a table is not in the table takes about 2 operations (a good hash function will cause every other cell to be empty, and thus the longest sequence of filled cells is.. 1 cell long). Note that this efficiency is affected only by the density of the table, not by the number of elements in the table. Obviously, it is hard to find such a perfect hash function (unless we know something about the nature of the elements' keys in advance), but many good hash functions would lead us to near-optimal results.

Details

1. API. The following set of functions should be written, which will be used as an interface for the programmer. Other functions may be added in order to make your code more readable and flexible.
    1. `struct hash_table* create_hash_table(int size);`

       Create an empty hash table, with "size" cells. Return a pointer to a structure with the table's information, of NULL in case of failure (e.g. out of free memory).

    2. `int hash_insert(struct hash_table* htable, char* key, void* element);`

       Insert the given element, with the given key, to the given hash table. Return '1' on success, '0' if an element with the same key is already found in the table, or '-1' if the table is full.

    3. `void* hash_remove(struct hash_table* htable, char* key);`

       Remove the element with the given key from the hash table. Return a pointer to the removed element, or NULL if the element was not found in the table.

    4. `int hash_num_elements(struct hash_table* htable);`

       Return the number of elements found in the given hash table.

    5. `void destroy_hash_table(struct hash_table* htable);`

       Delete the given hash table, freeing any memory it currently uses (but do NOT free any elements that might still be in it).

    6. `int hash_value(struct hash_table* htable, char* key);`

       Not an external function, but a must. There was a lot of research done regarding "what is a good hashing function?". I'll present a very simple one here, but you should try your own ideas, and test them out on various input sets:

       ```
       Add the ASCII values of all characters in the key, and take the
       result of their sum modulo the table size (in C notation, the '%'
       operator calculates the modulo of a number. "i1 Modulo i2" means
       "the remainder of dividing the i1 by i2).
       ```

2. Table With Varying Size. Since we saw how important it is for the table not to be too dense, we will now implement a hash table of varying size. When an element's insertion causes the table to be too dense (for example, more then 70% filled), we will make the table twice as large (thus making it about 35% filled). When an element's deletion causes the table to be too dense (for example, less then 20% filled), we will cut its size by half (making it about 40% filled). These table expansion and shrinking operations might be very costly - we need to modify the hash function to so that it will generate numbers within the new range, and need to re-insert all the elements into the new table. However, in most operations we win the efficiency of the very sparse table, so the efficiency for a set of insertion, deletion and lookup operations is still kept. An Extended API is required to support this modified table:
    1. `void hash_set_resize_high_density(struct hash_table* htable, int fill_factor);`

       Set the fill density of the table after which it will be expanded. The fill factor is a number between 1 and 100.

    2. `void hash_set_resize_low_density(struct hash_table* htable, int fill_factor);`

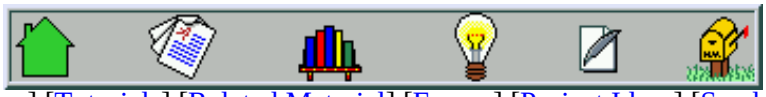
       Set the fill density of the table below which it will be shrank. The fill factor is a number between 1 and 100.

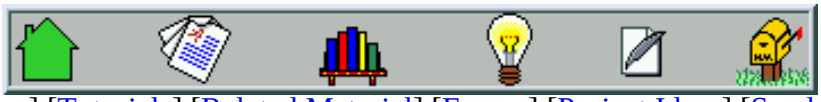
    3. `int hash_get_resize_high_density(struct hash_table* htable);`

       Get the "high" fill density of the table. The fill factor is a number between 1 and 100.

    4. `void hash_get_resize_low_density(struct hash_table* htable);`

       Get the "low" fill density of the table. The fill factor is a number between 1 and 100.





Writing A VT100-Based PacMan Game

Goal

1. Writing a clone of the good-old PacMan game, using ASCII text, on a TV-100 compatible terminal.
2. Learning how to use VT-100 escape sequences to generate full-screen ASCII animations.

Background

We have seen how we can generate timers using the ALRM signal, and we can use them in order to handle some time-based operations. A good example would be writing a simple computer game, and what would be better then trying out the old game of PacMan? It's much simpler then you might think...

The game allows the user to control Pac using the keyboard. Every short period of time (e.g. every two seconds) the game needs to move the computer-controlled ghosts in a semi-intelligent manner (i.e. make it look like they are chasing Pac). In addition to that, we need to set the keyboard into "raw" mode (so we can process key presses as they are made, as opposed to the default line-mode of VT-100 terminals). At the end of the game (either normal end, or brutal Ctrl-C end) we need to reset the keyboard to its normal line mode.

The "graphics" will be done using plain-text, with a 'c' denoting Pac, A '@' denoting a ghost, a '.' denoting a normal point that Pac can eat, and a '%' denotes a power-point, that when ate by Pac, gives him the power to eat the ghosts for a short period. Making the text displayed in specific locations on screen will be handled using something called "VT-100 Escape sequences". These are character sequences that are interpreted specially by the terminal emulation (instead of being printed as-is to the screen) and contains commands such as "place cursor in location X,Y", "clear screen", and so on.

Details

1. VT-100 Escape Sequences.

   All VT-100 escape sequences begin with the Escape character (ASCII code 033, in octal. In a string it is denoted as '\033'). Here are a few useful sequences, and their meanings:

   \033[2J
   : Clear the screen.

   \033[H
   : Place the cursor at the top-left of the screen.

   \033[30;20Hhello world
   : Print the string 'hello world' starting at row '30', column '20' on screen. 'hello world' may be replaced by any other string. The same goes for the co-ordinates

   \033[?25l
   : Disable echoing the cursor on screen (only supported in VT-220 or better emulations)

   \033[?25h

Re-enable echoing the cursor on screen (only supported in VT-220 or better emulations)

2. Double Buffering.

   When displaying animation, one can display all changes directly on the screen. When more then a single change at a time is made, this has the effect of causing the screen to flicker. In order to avoid that, most programs calculate all changes first, and then redraws the modified area in a single operation. This technique is called double-buffering. In our program, we could use a two dimensional array to denote the screen, make all calculations using this array, and then re-print the whole array at once. To be more efficient, we can hold two such arrays. One shows the current state of the screen, the other is used to calculate the next screen-shot. When we paint the new screen, we compare all cells of the two arrays, only redrawing those that differ.

3. Handling Moving Characters.

   There are various methods of handling of moving a character: where is Pac or a ghost allowed to go to, how to make the movement "flow", etc. One method of checking where to go is by checking "what is in there?", and defining a different icon for each type of square - 'W' for a wall, 'w' for the location of the ghosts house (where Pac may not enter), '.' for a normal pill, '%' for a power pill, and a blank for an empty square. We say that Pac is allowed to walk onto places containing anything but 'W' or 'w'. We also define that when Pac steps on a pill, he eats it. However, when a ghost steps on a pill, it does not eat it, so we need to remember the pill was there (and what was its type), in order to place it back there in the ghost's next move.

4. Screen Layout.

   A game screen might look something like this layout. Note that in one place on each side there is a missing wall - this forms the tunnel that allows Pac (as well as the ghosts) to "teleport" from one side of the screen to the other in an instant. In the original game, moving through the tunnel took a short while, and it might be that a ghost would kill Pac while both travel in the tunnel coming from opposite directions.

5. Switching to/from raw mode.

   By default, a Unix terminal is set to line mode (or buffered mode). In this mode, the kernel buffers the user's input until they press the ENTER key, allowing her to do simple editing (backspace over characters, delete the whole line, etc). Once the user presses the ENTER key, the kernel passes the whole line of input to the process reading from the terminal. This means that even if we use a function to read a single character, it won't receive any input until the user presses ENTER. obviously, this is not a good idea for an action game, when the user needs quick response to key presses. Furthermore, when the user presses a key, the system echos it back on screen. This is also not good for an action game (we don't want to see the keys echoed back to us, rather see them causing little Pac to move all over the screen).

   The easy (and lazy) way to make these mode switches is by using the "stty" command. Here is how to enter raw-mode with no echo (while still allowing Ctrl-C, Ctrl-Z and Ctrl-\ to send signals):
   `system("stty raw -echo isig");`
   The `system()` function executes the given Unix command in a sub-shell. In this case - runs stty with "raw -echo isig" as its parameters. In order to restore normal mode (line mode with echo), we can use this:
   `system("stty -raw echo");`
   If you wish, you may read the manual page of "stty" for more info.

6. Setting A Fine-Grained Timer - `setitimer()`.

   When we saw how to set timers using the `alarm()` system call, we were limited to delays of full seconds.

For an action game, this is not sufficient. Thus, we could use the `setitimer()` to generate the ALRM signals. It allows us to specify intervals of micro-seconds, and to even ask the system to reinstate a timer automatically after the first got set off. Note that `setitimer()` is not defined in POSIX, but it appears in both BSD systems and SVR4 systems, so it may be assumed to be portable. Here is how to use it:

```
/* for time measurements, delay between shape movements */
struct itimerval delay;

/* define the first timer to set off after 40,000 micro seconds.  */
h.it_value.tv_sec = 0;
h.it_value.tv_usec = 40000;
/* define the interval between each two timer set-offs ('clicks') */
/* to be 40,000 micro seconds as well.                            */
h.it_interval.tv_sec = 0;
h.it_interval.tv_usec = 40000;

/* finally, start off the timer */
setitimer(ITIMER_REAL,&h,0);
```

After the first call to `setitimer()`, the timer will set off every 40,000 microseconds, invoking whatever handler we set for the SIGALRM signal.

7. Multiplexing User Input And Timers.

   In order to allow the program to handle both user input and alarms, our program will use the `read()` system call in order to read user input, rather then the `getchar()` function. This is because receiving a signal will cause `read()` to return immediately (this assumes the signal handler for SIGALRM was defined using `sigaction()`, not using `signal()`). Thus, the main loop will begin with a `read()` on file descriptor 0 (which is used for standard input), and then a test for the return code. If it is '-1', and `errno` equals `EINTR`, then we probably got a SIGALRM, and thus should call the elements moving function. Otherwise, the user probably pressed a key, and we should check which one, and switch Pac's moving direction accordingly.

8. Structure Of The Program.

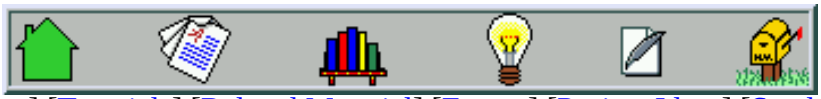
   1. Write two functions to switch screen mode to raw mode and back.
   2. Write a function that accepts 3 parameters: X and Y coordinates, and a string. The function prints the given string in the given screen location (using VT-100 escape sequences). Remember NOT to print a new-line character in the end!
   3. Write a signal handler for SIGINT (Ctrl-C) to restore normal screen mode before exiting. Write another handler for SIGTSTP (Ctrl-Z) to restore normal screen mode and suspend, and to restore raw mode when the process resumes execution.
   4. Write a signal handler for SIGALRM that sets a flag stating an alarm was received.
   5. Write a function that moves Pac in his current direction, blocking him if he bounces to a wall, eating a pill (and increasing score) if the new point contains a pill, setting on "power" mode if he eats a power pill (or setting it off if it was set on, and the time limit for "power" mode has elapsed), killing Pac if he stepped on a ghost (or killing the ghost, if this happens while "power" mode is set on).
   6. Write a function that moves a single ghost, killing Pac if the ghost steps on Pac (or killing the ghost, if "power" mode was on). The function should calculate where the ghost should go to next in order to chase Pac. A simple function will just try to get it closer to Pac. Note that a ghost may never reverse

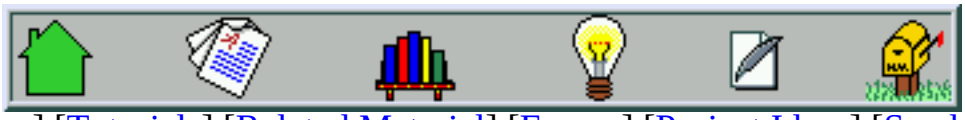
its moving direction. It may choose to continue forward, move left, or move right. This is how it is done in the original game.

7. Write a function that performs all characters movements: moving Pac and moving the ghosts.
8. Write a function that gets two arrays, one describing the current screen layout, another that describes the new (desired) screen layout. The function will perform the drawing, as described in the double-buffering scheme earlier. Note that in order to allow the user a fair game, Pac must move faster then the ghosts initially. As the game proceeds, the ghosts speed should be increased, to make it harder to play. A good way to perform that is to make the timer tick very quickly, and then make Pac move every 50 ticks, while the ghosts move every 100 ticks. As the game proceeds, decrease the number of ticks between every move of the ghosts.
9. Finally, the main function registers the signal handlers, starts the timer (with `setitimer()`) and performs the main loop of waiting for user input (or alarms).

9. Small Enhancements.

- Add levels to the game. The higher the level is, the faster the ghosts move.
- Add magical prizes, like in the original game. On occasion (randomally) they appear just below the ghosts house, and give many points if Pac eats them. If they are left there for a period (for example, 15 seconds) they disappear.
- Enhance the ghosts' chasing algorithm. Make them cooperate (i.e. if one sees another is chasing back, it will try to come in from the other direction). Try to come up with your own ideas.
- Write a "demo" mode, in which Pac is moved by the computer. Try to think of a smart algorithm for Pac that enables him to eat all pills on screen, while keeping away from the ghosts.





Implementing A Simple File-Transfer Client And Server

Goal

1. Implement a simplistic file transfer server and file transfer client.
2. Add support for multiple client connections.
3. Add support for directory commands.

Background

Serving files is one of the most common services of Unix servers. This is done using many different types of protocols, such as FTP, NFS, SMB etc. These protocols are not very easy to implement, so we will limit our exercise to our own proprietary protocol.

Generally, the file server has a given sub-directory on the server from which it serves files. Any file path given to the file server by a client, is interpreted relative to that sub-directory. For example, if the sub-directory on the server is "/export/files", and a client asks for a file whose path is "/data/config.txt", the server will serve the file "/export/files/data/config.txt".

We also need to define some protocol used between the client and the server. The client will send commands (with optional parameter), and the server will send responses. Each message transfered will be preceded by the size of the message, so the receiving side will know how many bytes to read. A message sent by the server will contain a return code, and may contain the output of a command (such as a directory listing), or even a file (for a file transfer request).

Details

1. Protocol Details.

   Each message transfered by either side contains 3 fields:

   1. Message length - 4 bytes number in network byte order.
   2. Message code - 2 bytes number in network byte order.
   3. Optional - message parameter - as large as required.

A client initiates a connection to the server. The server sends a greeting reply (code '1', with a "hello there" parameter). The client may send only the following command: gimme file (code '101', with a parameter containing the file path). The server may respond with any of the given responses:

1. Permission denied (code '201', the error message as the parameter).
2. File not found (code '202', the error message as the parameter).
3. Internal error (code '203', the error message as the parameter).
4. Unrecognized command (code '204', the error message as the parameter).
5. Path is a directory (code '205', the error message as the parameter).
6. Here it is (code '301', the file's data as the parameter).

Once the command's output has been sent to the client, the server terminates the connection.
You should define a constant for each message code, and use a shared header file that is included by both the client and the server's code.

2. Flow Control.

   In order not to abuse the system, every time a message is sent, the sending party should perform proper checking to see if the socket is ready for write operations (using the `select()` system call).

3. Security/Sanity Checks.

   The server must check any parameter it got from the client:

   1. Make sure the file path begins with a '/' character, and does not contain a '..' sequence in it.
   2. Allocate buffers large enough to accept reasonable command sizes. If the client specifies a too long message size, send an "Internal Error" message and terminate the connection.

4. The Client.

   The client will have a command-line interface. It will accept a command and an optional parameter, and send them to the server. The only command it accepts is "get", with a full file path as a parameter.

5. Logging.

   The server should write a one-line message to a log file regarding any of the following operations (with a time and date at the beginning of each message):

   1. A connection to a client is established. Write out the file descriptor of the socket created for this connection.
   2. A command was received from a client. Write out the command code, its parameter, and the file descriptor through which it arrived.
   3. A response was sent to a client. Write out the response code, its parameter, and the file descriptor through which it arrived. The parameter is the message text (in case of an error), or the file path (in case of a file transfer).

6. Advanced Options.

   1. Multiple Clients.

      Add support for multiple clients. The server should keep track of the state of conversation it has with each client, and handle sending data for more then a given amount of time - the connection to the stuck client is terminated.

   2. Extra Commands.

      Add support for the command "list" (with a path to a file or a directory as a parameter, and a message code of '102'). The server will reply with a formated list of the directory's contents (in case of a directory) or the file (if the path is that of a file). The reply's message code will be '302' (if successful).

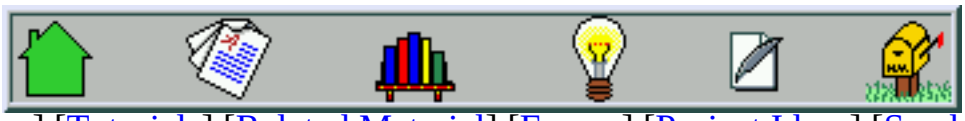
   3. Persistent Connection.

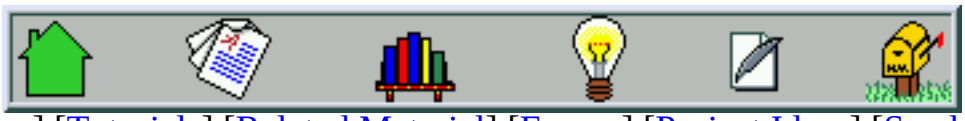
      Add support for a persistent connection (i.e. the TCP connection is kept open after the server sent the response, to allow the client to send more requests). This time it is the client that terminates the connection when done. Add a commands-reading loop to the client side, so the user can type in more then one command. Add a notion of a "current directory" in the client (NOT in the server), so the user can type in partial paths (note that the "cd" command requires asking the server if the new path is really a directory. add a new message type for checking this to the protocol, and a new response type).

   4. Timeout.

      Add a checking for timeout by the server - if it got stuck with sending data for

more then a given amount of time - the connection to the stuck client is terminated. This is tricky, and requires adding timers support on top of the `select()` loop. Think how to implement this, using a delta list of timers, and the timeout parameter of the `select()` system call.





Tiny Multi-Threaded HTTP Server

Goal

1. Writing a most basic multi-threaded HTTP server that can serve off static pages.
2. Testing out various ways of thread pools management.

Background

The HTTP (version 1.0) protocol is a simple protocol that allows a (web) client to requests files from a server. Various types of requests can be sent by the client, the most basic (and the one we'll implement) is the "GET" request, to fetch data files. The checks the request, returns a header containing a result code, with the file attached after the header. Finally, the server closes the connection.

The client's request might actually contain much more data (e.g. the preferred language for the document, what MIME types the client understands etc.). Our server will simply ignore any of these hints. This way, we will be able test our server using commonly used web browsers (lynx, netscape, and even *gasp* internet explorer).

When the server sends a data file to the client, it needs to tell it a few things about the file: the file's size (in bytes), the last modification date of the file (in the same format as returned by the `ctime()` function), and the MIME type of the file. A MIME type is a string made of two parts, separated by a '/' character. The first tells us what type of file we have (text, image, audio and so on). The second part tells us the specific sub-type (gif and jpeg for images, x-wav and mpeg for audio, html or plain for text, and so on). The server usually deduces the MIME type of the file based on its extensions, not its contents, for efficiency reasons. Here are a few common file extensions and their associated mime types:

.html
    text/html
.htm
    text/html
.txt
    text/plain
.gif

image/gif

.jpg

image/jpeg

.qt

video/quicktime

If the server encounters an unknown suffix, it should attach a default mime type to the file (many servers default to 'text/plain' for unknown suffixes, others use 'application/octet-stream', to make sure the client won't try to mess with the file, but rather allow the user to save it to disk).

Details

1. General Server Data. The server will serve files from a directory knows as the "root documents directory". Any file path supplied by the client should be relative to this directory.

2. Requests Acceptor Thread. One thread will work as the requests acceptor thread. It will usually be blocked on the `accept()` system call, and then create a "client structure", and place it on the incoming requests queue, an a fashion similar to that demonstrated in our "multi-threaded requests handler" example.

3. Handler Threads. Each requests will be dequeued by a handler thread. The thread will get the socket from the request, read the data from the client, and act upon it.

4. Request Parsing Module. A module should be written to parse the HTTP request, and extract the relevant data from it (i.e. the request type and the file path). This module will read directly from the socket, and generate a structure containing the parsed data. An HTTP request might look like this:

   ```
   GET / HTTP/1.0
   <more headers come here>
   ```

   Note that an empty line is used to denote the end of the request. If anything other then a 'GET' request is sent, we should eventually reply with an error message.

5. Reply Preparation Module. This module will take a request structure, and prepare a reply structure, containing the result code, data and time, full path the the data file (if any), MIME type of the file and last modification data of the file.

6. MIME-Type Deducing Module. This module is used to deduce the MIME type of a file, based on the file name's extension. It will be made of a single function that gets a file name as a parameter, and returns a string containing its MIME type. Initially we can store all the MIME type mapping information in an array in memory, and search it linearly.

7. Reply Emitting Module. Another module will be responsible for generating a reply. It will be given the reply structure, and will send its data over the socket, in an HTTP reply format. This includes writing the header (result code, server type, date and time, MIME type and size of attached file, if any, and last modification date of the file), followed by the contents of the data file (again, if any). The reply would looks something like this:

```
HTTP/1.0 200 OK
Date: Fri, 13 Nov 1998 16:32:44 GMT
Server: Tiny-Tux/0.1a
Last-Modified: Sun, 01 Nov 1998 12:48:09 GMT
Content-Length: 4782
Content-Type: text/html

<file contents come here>
```

This reply is sent in case of success (i.e. request understood, file found and the server managed to access it). Note the empty line separating the reply's header from the reply's body.

If there was some error, the result code (first line) should be replaced by an appropriate result code, and the file-related fields (e.g. "Last-Modified:", "Content-Length:" and "Content-Type:") removed. Error codes go as follows:

HTTP/1.0 404 File Not Found

The requested file was not found.

HTTP/1.0 403 Forbidden

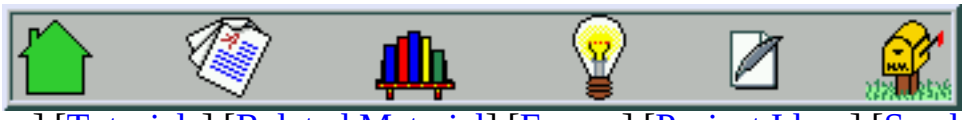
The server cannot open the file for reading.

HTTP/1.0 501 Method Not Implemented

The given request type is not implemented (i.e. something other then GET was requested, such as 'PUT' or something completely illegal).

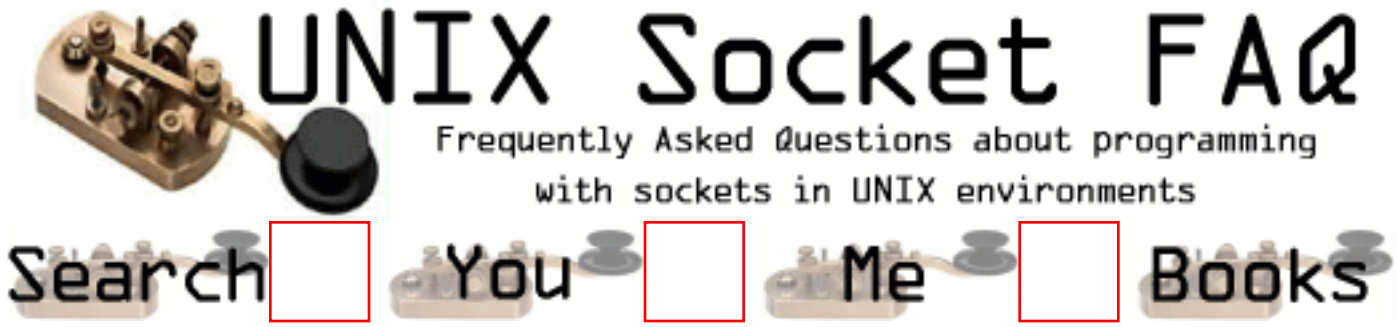
HTTP/1.0 400 Bad Request

The server failed to parse the request (e.g. the file path has an invalid format, the protocol type/version was not specified, etc.).



Unix Programming Frequently Asked Questions

This document was generated on 1 September 2000 using the texi2html translator version 1.51.



Welcome to the home of the UNIX Socket FAQ!

There's some really major "under the hood" changes I've just made to the FAQ software that drives this site. I've seperated the content from the functionality, and I'm releasing the source under the GPL. I've named the software "Self Serve FAQ" because of the way it empowers the users to add questions and provide answers without intervention from the FAQ maintainer. I've created a new site on lcg.org for it at http://www.lcg.org/ssfaq/ and you can find out more about it there.

I've added a couple of new features. First, you can download a single file version of the faq here. Second, the main page now shows the most recent updates. Let me know if you notice any quirks.

I've "upgraded" the sock-faq to the new software, so please let me know if you find any problems by posting to the new ssfaq site. Hopefully I'll get any problems worked out early on.

My mirrors appear to have dissapeared, and it may be because they were difficult to keep up-to-date before. One of the first things I want to work on with the new software is making the whole mirror thing easier, so I hope to have mirrors available soon.

If you are looking for Dr. Charles Campbell's Simple Sockets Library, you can download version 2.09 here.

5 most recent updates:

1. How does it know to send back the right information (Updated July 11, 2001)
2. C function to empty a socket (Updated July 11, 2001)
3. confused with backlog of listen() (Updated July 11, 2001)
4. Problem in Sockets while using Message Queues (Updated July 11, 2001)
5. I loss data when I receive from a unix socket!!! (Updated July 11, 2001)

Uncategorized Questions:

1. tracking socket calls
2. How can I determine the space available in the send buffer of a socket?
3. Socket question
4. How to act as server and client at the same time.

5. What are the implications of reducing receive buffer size?
6. reversed name resolving
7. How can I send packet on a choosen interface ?
8. Fix IP source address ?
9. How do I reject a pending connection?
10. Different Flavors of Pipe Signal
11. how to restart a server immediately
12. How to use sendmsg/recvmsg (or does anyone actually use this "feature"?)
13. Motorola 88k Gotcha
14. Socket question
15. Errors in sockets
16. problem in socket programming
17. client/server environment
18. Sending large (> 50 meg) files (binary + ascii) through sockets
19. Proxy Handling
20. socket and XTI
21. Sending & Receiving data on a socket
22. Stream Sockets, Multithreading, and Windows
23. Subsequent connect call fails (EBADF); why?
24. HTML Client
25. Passing sockets to pre-forked children
26. Unix domain sockets using linux
27. Can port be shared ?
28. Reading from a port
29. java.net objects for reading port
30. Socket Programming
31. How do I get a MAC address using code on Solaris?
32. send object through socket
33. Client/Server mystery (sockets)
34. Connec to different hosts/ports from the same host/port
35. Compiler Difficulty
36. How to know if the sending buffer is fully empty?
37. How do I get the IP Adress to which a client connected to my server ?

70. optimal solution
71. Chat program in C/UNIX
72. how to create second socket on my client application
73.
74. how many simultaneous socket connections can a concurrent server accept?
75. Sending integer values to sockets.
76. can u answer this ??
77. C to unix base by win sock
78. C to unix base by win sock
79. Multiple blocking on select.
80. multiple connections
81. using libpcap
82. buffer limit
83. How to find local IP address in client.c program
84. Swap area
85. Using socket fuctions and debugging
86. how does one measure throughput in TCP and UDP?
87. defnition
88. source-IP equals dest-IP
89. Proxy Server written in C using sockets
90. gethostbyname
91. select() for timing
92. nic card
93. How do I access individual packets being received on a socket?
94. Time out on Recv
95. restart server
96. SO_RCVBUF
97. recv() fails with EGAIN for a blocking socket
98. sending floats over winsock
99. A server on multiple machines behind a Net Address Translation Device
100. qustion about socket
101. how to determine if socket is blocking or non-blocking
102. read() and timeouts

103.
104.
105. XDR
106. Proxy with Sockets - grabbing the http portion
107. How do i send file(s) to server
108. NT sockets VS Unix sockets
109. Blocking/Non-Blocking
110. sockets
111. Why does select not see available buffer space on a loopback socket?
112. socket bufsize recv and send > 4k
113. Blocking writes fail
114. When gethostbyname() doesn´t work...
115. viewing active data over socket
116. why is my client not able to connect to the server after the first time
117. Adding delay between IP package
118. psuedo terminals
119. transmit files via socket
120. Error bind
121. How can I check the evolution of a non blocking write on a socket?
122. discard socket data
123. discard socket data
124. Starting server when disconnected from ethernet fails to run
125. packets lost using SOCK_STREAM
126. packets lost using SOCK_STREAM
127.
128. using select to check if the second end is alive
129. package loss and delay
130. Sockets with Unix
131. Number of Ports? Newbie question
132. binding multiple file descriptors to a single port
133. recv() on unconnected socket ?
134. Sockets, timeout, signals
135. Port in use question?

136. Accept call on unconnected socket
137. Send & Receive using the same port
138. Ports in use
139. Newbie-Question:How to start with a proxy server?
140. Why my ping fails?
141. Final result of setting Timeout for connect() in socket
142. how many clients??
143. raw socket with udp port scan
144. raw socket
145. raw socket
146. How do i program different socket types??..Example: type 8 type 3, etc..
147. How do i program different socket types??..Example: type 8 type 3, etc..
148. TIME_WAIT
149.
150. printf() and select()?
151. Should socket be closed if sendto() or recvfrom() fails ?
152. TIMEOUT USING SELECT
153. CLosing connexion
154. problems in UDP socket programming(recvfrom)
155. Problems in recieving UDP packets
156. UDP socket. receiving broadcast problem.
157. FTP protocol
158. multiple applications on same host using same socket
159. Connecting to a Web server
160. Reading &Writing from UDP Port. (Help needed for Win)
161. Server: accepting and handling multiple clients
162.
163. bind fails with udp socket
164.
165. sending structures through TCPIP sockets
166. I need help, i am a beginner
167. Listening to port traffic?
168. how do i convert binary..

169. how do i convert binary..
170. How can i receive a file which size is unknown?
171. Err#9 EBADF on door_info call
172. Problem with write
173. How can I know my peer is dead(or close the socket) without read or write?
174. sigaction and sigset
175.
176. What is EAGAIN ??
177. Question on select() and send()
178. Problems with setting a timeout on connect
179. setting a timeout on connect
180. read() and MSS
181. unable to establish socket connection
182. exception in java
183. How could I know that the client stopped sending the data and waiting for me to response?
184. recv returns 0!
185. writing into file
186.
187. TCP/IP called address for a ANY_HOST listen
188. VB client to Perl Server
189. How can i make a server act as connection oriented as well as connection less at same time
190. Cannot create UDP socket - port number in use
191. socket c programming
192. question
193. How can i send communication break
194. How can i send communication break
195. Data comes together
196. Link Error.
197. demon.
198. MX lookup
199. hostname

200. accept() hangs
201. Semantics of ioctl with FIONREAD and recv
202. Server Application
203. WWW
204. TCP/IP does not preserve message boundaries
205. MX lookup
206. Using select() for web-server program
207. Ports that Hang
208. read-write problem
209. I am getting connection refused when ftp from UNIX to NT based system
210. can select() notify if a tcp/ip fd is readable?
211. write files to NT
212. TIME_WAIT and EPIPE
213. IPC using -shared memmory and semaphore Vs Sockets
214. Memory corrupted by child process
215. Number of sockets and OPEN_MAX limit
216. SIGPIPE in multi-threaded client/server program
217. How to use select() ?
218. listen fails on OpenServer 5.0.5
219. Authorizations doubts
220. About sharing the same socket/same connection to server for two programs
221. c++ and sockets (problem using close)
222. select/recv: how to read complete message?
223. Blocked I/O Timeout?
224. Securing sockets
225. how to get all listening ports ?
226. If ISocket permission denied
227. many http request via single socket.
228. HOW can I tell protocol?
229. Problems receiving UDP Broadcast...
230. Changing the buffer size of a listening socket
231. read count diff when accessing web server.
232. How to allocate the server buffer effectively?

233. signal(SIGPIPE,SIG_IGN) still doesnt ignore SIGPIPE
234. Socket send guarantee delivery ?
235. Problems with socket UDP
236. Socket Latency
237. Connect to a server
238. RECVFROM and IOCTL Question
239. sockets program to retrieve web content
240. How to bind a socket to multiple known ports?
241. Re: Sending files over (Socket)
242. Why Select is not realizing that there is data on socket to read.
243. How to get ISP assigned IP
244.
245. UDP has connection in linux?
246. UDP has connection in linux?
247. Why UDP is used for the Internet instead of TCP?
248. sockaddr structure oddity
249. Can same port be used by two processes?
250. how to choose my own source port
251. Socket Address Structure
252. IPC
253.
254. XDR
255. accept() error!
256. How to post data using Post method to a servlet
257. my program gets stuck when closing a socket.
258. why does bind() give EADDRINUSE
259. Compilation problem
260. live AP_INET SOCK_STREAM connection
261. TCP/IP issue on HP Unix
262. select() between a UDP socket and a TCP client socket
263. recv gives a segmentation fault.
264. select question
265. Single server with multiple clients

266. How to write socket programming between UNIX & Windows platform
267. CONNECTION ABORT
268.
269. MAC Address from client w/o using ARP?
270. Threads and sockets
271. write() Hanging Program.
272. KCP
273. TCP Send-Q Retransmission
274. How can I do a ftp???
275. reading from buffer
276. How to retrieve data sent by client to server when data sent is stored in a buffer?
277. broken pipe?
278. about recv() call
279. binary file prob
280. how to signal select function
281. Java linux socket problem
282.
283. Web Hostinb
284. Reg send() socket call
285. exit parameter when send call fails
286. Using recv function
287. CLOSE_WAIT
288. CCGI - C socket communication
289. setsockopt with IP_MULTICAST_IF question
290. Socket is not being released
291. Can I bind socket descriptor to a port which bound by others ? How ?
292. How Many Sockets?
293. Questions regarding both Clients and Servers (TCP/SOCK_STREAM)
294. Questions regarding both Clients and Servers (TCP/SOCK_STREAM)
295. How can I bind to port use different name ?
296. how to bind a port that was bound ?
297. Accept method
298. Winsocket><sockets

299. is there any way to determine which process id is running a particular server ?
300. simplest way to send structures via sockets
301. Recv and Send functions Result
302. HOW MANY BITS ARE IN A TCP SOCKET?
303. server host name
304. blocking write
305. TIME_WAIT
306. How to flush the socket ?
307. client /server problem
308. Maximum speed in the socket
309. Socket reliability
310. blocking ip address
311. send a buffer from server to all clients?
312. Accessing non-HTTP servers through a proxy
313. re: socket error
314. Asynchronous I/O
315. Encrypting data on socket
316. RE: Adding packet handler for socket
317. Select hangs randomly
318.
319. tcp server
320. non blocking connect with select
321. socketing data loss between Unix and Windows
322. SCO -> DOS Communication via socket
323.
324. Meaning of UDP "connection"
325. How to delete a route using API calls
326. select() on blocking environment
327. Where can I download QT v2.2.2
328. how to write optimal performance TCP server
329. Client and Servers.
330. SOCKETS AND IMAP SERVER
331. client/server to check a file in a dir

332. recv() - the delay between recv()s
333. OOB msgs
334. timeout on send with SO_SNDTIMEO??
335. Server-Client Design
336. CLOSE_WAIT Error
337.
338. SO_RCVTIMEO question
339. Why does Apache only accept a message after the socket closes?
340. How can I Receive or Deal with Out-of-Band Data
341. SIGURG & OOB problem ?
342. SIGURG & OOB problem ?
343. Broadcast and receive DHCP messages
344. What is Meant By RFCxxxx ?
345. How we can Get Mails From Servers Using Sockets.
346. How we can Get Mails From Servers Using Sockets.
347. Can someone answer to that ??!!
348. select statement times out when there is data
349. sending File Descriptors across Unix Sockets
350. Server servicing multiple sockets (ports)
351. How to use "errorfds
352. How to use "errorfds
353. How to use "errorfds" in select()
354. Select returns EBADF
355. core dump on send/recv
356. socket programming
357. transfer data from winsock to unix by C/C++ socket programming
358. NT Socket to Unix
359. pthread_join
360. sending binary file from a windows client to a unix server
361.
362.
363. using socket to connect : a client to a server and a client to another client ?
364. Send a file and struct

365. configure tcpip/ppp using unix interprise 5.05 sco
366. Determining MAC from IP w/ARP
367. Sockets Intialization Failed!
368. Anonymous FTP
369. Problem with socket-timeout
370. Which langage is better for network programming ?
371. server
372. I need to send 0x00 0x00 0x00 0x1 ?
373. Broken Pipe
374. Open a session
375. Can i Read more than a PAGESIZE in a single read call.....
376. kick off Unix process from Windows 2000/NT
377. Why my recv is too slow?
378. select loop..
379. udp port reuse
380. sockets
381. socket send timeout
382. byte order of data received
383. Searching For: Inline "Hex to Decimal" conversion routine - to be used during execution of Socket Datagram Program
384. How to determine which IP client is calling?
385. How to reconnect a TCP socket?
386. Ada Language Unix-based "Datagram Link Program" Type Error When Porting To And Recompiling On A Linux Machine
387. call to write blocks forever
388. persistent connection to a web server
389. Windows / Unix Socket Communication
390. DANGER : big bug in Select() !!
391. Socket on Unix
392. rebinding bound socket
393. How can a process know its hostname ?
394. Sending info down sockets
395. Sending info down sockets

396. GET Protocol
397. get multiple documents from www server
398. problems receiving UDP Broadcast
399. To change, my IP address.
400. socket and rpc
401. What is "IDMaps" and how to use it in indentifying the closest mirror site?
402. fprintf error code
403. % sign in a file read by my SOCKET server
404.
405. socket 7 processor
406. Need help in compilation error2001
407. Controlling an unexpected socket close.
408. What is a Blocked socket ?
409. server/multiple clients
410. Connection failures
411. Socket Programming
412. problem in client identification by server
413. Connecting a socket throw a proxy with autentication
414. How to test if a server closed my socket?
415. Line down condition
416. Select Read problem
417. How does server control number of client---help
418. how to detect SIGCHLD
419. TCP Socket Read/Write With Two Threads
420. Data Flow from socket call(s) to a network device driver
421. Data Flow from socket call(s) to a network device driver
422. Data Flow from socket call(s) to a network device driver
423. Select on a unconnected socket
424. What is client IP on a multi-IP machine?
425. Can I increment an IP address "IP=IP+1"?
426. SCO Socket
427. How socket calls are get mapped to network device driver functions
428. using C program to get connect to the web server

429. urgently required!!!!!!!1
430. UDP Buffers
431. Read and write irrespective of the order
432. how to use a C client to connect to an RMI port?
433. resource temporarily unavailable - recv problem
434. setting file permissions for sockets
435. Timeout period affecting selects ability to detect data
436. Making sockets working over the internet
437. EINTR Testing
438. Where can I get the converter to convert Windows to Unix?
439. TCP/UDP on same server
440. HOW CAN I open n connections working at the same time ??
441. non blocking socket
442. RST packet ???????
443. Source Code client.c And Server.c for UNIX
444. how to check if a socket is connected or not?
445. NBT packets???
446. TCP UDP on same server
447. i want to implement a natwork game, i want to discover all hosts on a subnet that are participating in it, what shall i do?
448. Using select with recv send
449. communication between a java program and a c program
450. Getting Signal Number 2147482380 While sending data on socket
451. Time Polling
452. UDP BraodCast
453. Recv ( ) in blocking mode
454. Why is my socket not ready for writing?
455. writing to RAW ethernet socket
456. Tcp port configuration
457. Max data that can be send in one send
458. Regarding Non blocking calls
459. Why do I get "accept: Too many open files" error
460. Errors when a remote host crashes

461. using poll for packet timeout
462. How can I prevent from losing packet
463. client to multi-homed server
464. deleting files and directories
465. packet demultiplexing
466. number of connectons
467. Are there Token Ring programmer ?
468. UDP with retransmission?
469. UDP Server and Client on same machine gives Address already in use error
470. problem transferring file through socket
471. cgi and socket with DB
472. question about UDP SOCKET buffering
473. Broadcast a large file
474. recv Socket
475.
476.
477. Telecom call routing
478. why is inet_addr() not working??
479. NFILE parameter increases!
480. Blocking Vs Non-blocking
481. TCP connection re-establishment
482. single program acting as server and client
483. How can a server let other server connect with a client ?
484.
485. CGI problems.
486.
487. which is best Fork () or pthread ?
488. Timestamp
489. Difference between port & socket
490. Client write to closed server
491. How to flush a udp socket?
492. Binding of RAW Sockets
493. Server does not receive message sent by Client

494. problems with sending files
495. Difference between socket and port
496. Forwarding data from one interface to another
497. problem with error "unresolved text symbol" in UDP program using recvfrom & sendto
498. API to get state of socket?
499. Looking for advice on non-blocking sockets
500. how to specify a rang of port numbers to be used
501. Raw socket programming
502. IP-telephony
503. re-establish socket connection
504. Output Queue Gets Full
505. Transfer a file using UDP
506. SO_LINGER socket option does not work
507. Multiple Servers
508. How to terminate the read/recv call..?
509. Error in bindind an UDP socket
510. Invalid argument!?
511. Blocking socket recv with pthread
512. Instant Messaging Server
513. How to detect if process id is running
514. Broadcasting Problem
515. select problem
516. peer-to-peer chat application
517. help on UDP broadcasts
518. select question
519. Socket usage with Unixware 7.1.1
520. How can I send data from server to client?
521. socket source code
522. OPEN_MAX SUN app to Linux
523. multi-threaded connections
524. Open connection!
525. accept file descriptor
526. Unix command to know the ports.

527. Can UDP be used for sms
528. sending http request via proxy
529. Can you help me.?
530. Reuse of socket descriptors in linux
531. Non-blocking client
532. Multiple IP Address in Client, how to....
533. ERROR - CORE DUMP
534. How to create Sockets in UNIX
535. Socket and mutual exclusion
536. Connection testing using select() and recv()
537. testing TCP related APIs!!
538. How can I prevent from losing data ?
539. Disconnecting from modem
540. Receiving packet over a specific interface?
541. buffer question.
542. 6 second delay ...
543. Ho to make blocking send with UDP socket?
544. TCP Sockets with pthreads
545. Unable to catch SIGCHLD on a HP workstation.
546. Unable to catch SIGCHLD on a HP workstation.
547. Unable to catch SIGCHLD on a HP workstation.
548. Try to make a secure conection
549. Error while compiling
550. ftp server bind with REUSEADDR ?
551. Specific concurrent server port
552. socket deadlock
553. Select problem with NT and Solaris sockets
554. Where can I find the source code for server written for broadcasting audio ?
555. download component on Unix
556. UDP server question
557. Regarding connect system call
558. Could someone explain me something strange
559. what is HUP

560. Java Multicast Client with two NICs
561. Creating non_blocking sockets
562. how to write a port scanner on unix using c c++
563.
564. how can i attach file in a email
565. strings
566. Listening for remote socket error using poll
567. accept(); fills sock_addr structre with the same ip forever after one fail
568. Info
569. how to find the memory used by the Stack and the Heap for a function??
570. where is the memory fora local array allocated
571. Async. IO, How to store Clients IP and Port
572. writing to socket descriptor newSD
573. What is UMODE
574. Socket error
575. ENOBUFS Returned from Connect()
576. Broadcast UDP on linux gives ENOPERM
577. TCP programming
578. Program exits with code 141 during recv()
579. recv()?
580. please.... run it always wrong (on linux 7.0)
581. What happens with incoming buffer over-run
582. How does it know to send back the right information
583. C function to empty a socket
584. Windows sockets
585. How to bind Multi-IP to a MAC
586. How can I run a program and set some environment variable when "startx" on RedHat ?
587. How can I run a program and set some environment variable when "startx" on RedHat ?
588. Limit on maximum number of sockets
589. Select problem - read()
590. socket programming
591. can u send a .gif file through a socket. ?
592. About IP Alias

593. Connection refused when using syslog()
594. How can we use the LENGTH in recvfrom() in UDP socket?
595. Multiple clients using select
596. Problem in Sockets while using Message Queues
597. Asynchronous programming
598. confused with backlog of listen()
599. select() problems on DEC Alpha XP900
600. Mapping PDU data over tcp/ip
601. I loss data when I receive from a unix socket!!!
602. aix 4.3 ftp client data socket question (can you help?)
603. question
604. unix for submitting html form to a server
605. RAW Sockets
606. counter-wait system
607. Consulting C++
608. read() hangs w/ binary transfers
609. Asynchronous DNS lookups
610. When does EPROTO occur?

Categorized Questions:

1. General Information and Concepts
    1. What's new?
    2. About this FAQ
    3. Who is this FAQ for?
    4. What are Sockets?
    5. How do Sockets Work?
    6. Where can I get source code for the book [book title]?
    7. Where can I get more information?
    8. Where can I get the sample source code?
2. Questions regarding both Clients and Servers (TCP/SOCK_STREAM)
    1. How can I tell when a socket is closed on the other end?
    2. What's with the second parameter in bind()?
    3. How do I get the port number for a given service?
    4. If bind() fails, what should I do with the socket descriptor?

Add Your Own Question to the FAQ

Raw IP Networking FAQ

The FAQ is maintained by Thamer Al-Herbish shadows@whitefang.com. Feel free to mail in suggestions and additions to any material found on this page.

The FAQ material:

- The FAQ in text.
- The FAQ in HTML.
- The example raw socket source code (0.5).

Related Documents:

- The BSD Packet Filter: A New Architecture for User Level Packet capture.
- Data Link Provider Interface Specification
- How to Use DLPI (SunOS 5.x)

Related Links

- LBNL Network Research Group's FTP archive

Related FAQs

- The UNIX Programming FAQ:
  - Homesite in the UK
  - US Mirror
- The Socket Programming FAQ:
  - Site in the UK
  - US Mirror
- The TCP/IP FAQ:
  - http://www.itprc.com/tcpipfaq/default.htm
- The Secure UNIX Programming FAQ:
  - http://www.whitefang.com/sup/

Also you may want to join the Raw IP Networking mailing list, if you still have trouble.

shadows@whitefang.com

1. Answers to frequently asked questions for comp.programming.threads: Part 1 of 1

Bryan O'Sullivan

Last modified: Wed Sep 3 10:21:39 1997
$Revision: 1.10 $

0. Table of contents

2. Introduction

This posting consists of answers to many of the questions most frequently asked and summaries of the topics most frequently covered on comp.programming.threads, the Usenet newsgroup for discussion of issues in multithreaded programming. The purpose of this posting is to circulate existing information, and to avoid rehashing old topics of discussion and questions. Please read all parts of this document before posting to this newsgroup.

The FAQ is posted monthly to comp.programming.threads, in multiple parts. It is also available on the World-Wide Web, at <URL: http://www.serpentine.com/~bos/threads-faq>. You may prefer to browse the FAQ on the Web rather than on Usenet, as it contains many useful hyperlinks (and tables are readable, which is unfortunately not the case for the text version).

2.1. Reader contributions and comments

Your contributions, comments, and corrections are welcomed; mail sent to <threads-faq@serpentine.com> will be dealt with as quickly as I can manage. Generally, performing a

reply or followup to this article from within your newsreader should do the Right Thing.

While I am more than happy to include submissions of material for the FAQ if they seem appropriate, it would make my life a lot easier if such text were proof-read in advance, and kept concise. I don't have as much time as I would like to digest 15K text files and summarise them in three paragraphs for inclusion here. If you are interested in contributing material, please see the to-do list at the end of part 3 of the FAQ.

2.2. How to read this FAQ

Some headers in this FAQ are preceded by words in parentheses, such as "(POSIX)". This indicates that the sections in question are specific to a particular threads family, or to the implementation provided by a specific vendor.

Wherever it may not otherwise be obvious that a particular section refers only to some families or implementations, you will find one or more of the following key words to help you.

Key	Implementation
DCE	OSF/DCE threads (POSIX draft 4)
OS/2	IBM OS/2 threads
POSIX	POSIX 1003.1c-1995 standard threads
UI	Unix International threads
Unix	Of general relevance to Unix users
WIN32	Microsoft Win32 API threads

2.3. Acknowledgments and caveats

Although this FAQ has been the result of a co-operative effort, any blame for inaccuracies and/or errors lies entirely with my work. I would like to thank the following people for their part in contributing to this FAQ:

Dave Butenhof <butenhof@zko.dec.com>

Bil Lewis <bil@lambdaCS.com>

3. What are threads?

A thread is an encapsulation of the flow of control in a program. Most people are used to writing single-threaded programs - that is, programs that only execute one path through their code "at a time". Multithreaded programs may have several threads running through different code paths "simultaneously".

Why are some phrases above in quotes? In a typical process in which multiple threads exist, zero or more threads may actually be running at any one time. This depends on the number of CPUs the computer on which the process is running, and also on how the threads system is implemented. A machine with *n* CPUs can, intuitively enough, run no more than *n* threads in parallel, but it may give the appearance of running many more than *n* "simultaneously", by sharing the CPUs among threads.

3.1. Why are threads interesting?

A context switch between two threads in a single process is *considerably* cheaper than a context switch between two processes. In addition, the fact that all data except for stack and registers are shared between threads makes them a natural vehicle for expressing tasks that can be broken down into subtasks that can be run cooperatively.

3.2. A little history

If you are interested in reading about the history of threads, see the relevant section of the comp.os.research FAQ at <URL: http://www.serpentine.com/~bos/os-faq>.

4. What are the main families of threads?

There are two main families of threads:

- POSIX-style threads, which generally run on Unix systems.
- Microsoft-style threads, which generally run on PCs.

These families can be further subdivided.

4.1. POSIX-style threads

This family consists of three subgroups:

- "Real" POSIX threads, based on the IEEE POSIX 1003.1c-1995 (also known as the ISO/IEC 9945-1:1996) standard, part of the ANSI/IEEE 1003.1, 1996 edition, standard. POSIX implementations are, not surprisingly, the emerging standard on Unix systems.
    - POSIX threads are usually referred to as Pthreads.
    - You will often see POSIX threads referred to as POSIX.1c threads, since 1003.1c is the section of the POSIX standard that deals with threads.
    - You may also see references to draft 10 of POSIX.1c, which became the standard.
- DCE threads are based on draft 4 (an early draft) of the POSIX threads standard (which was originally named 1003.4a, and became 1003.1c upon standardisation). You may find these on some Unix implementations.
- Unix International (UI) threads, also known as Solaris threads, are based on the Unix International threads standard (a close relative of the POSIX standard). The only major Unix variants that support UI threads are Solaris 2, from Sun, and UnixWare 2, from SCO.

Both DCE and UI threads are fairly compatible with the POSIX threads standard, although converting from either to "real" POSIX threads will require a moderate amount of work.

Those few tardy Unix vendors who do not yet ship POSIX threads implementations are expected to do so "real soon now". If you are developing multithreaded applications from scratch on Unix, you would do well to use POSIX threads.

4.2. Microsoft-style threads

This family consists of two subgroups, both originally developed by Microsoft.

- WIN32 threads are the standard threads on Microsoft Windows 95 and Windows NT.
- OS/2 threads are the standard threads on OS/2, from IBM.

Although both of these were originally implemented by Microsoft, they have diverged somewhat over the years. Moving from one to the other will require a moderate amount of work.

4.3. Others

Mach and its derivatives (such as Digital UNIX) provide a threads package called C threads. This is not very widely used.

5. Some terminology

The terms here refer to each other in a myriad of ways, so the best way to navigate through this section is to read it, and then read it again. Don't be afraid to skip forwards or backwards as the need appears.

5.1. (DCE, POSIX, UI) Async safety

Some library routines can be safely called from within signal handlers; these are referred to as async-safe. A thread that is executing some async-safe code will not deadlock if it is interrupted by a signal. If you want to make some of your own code async-safe, you should block signals before you obtain any locks.

5.2. Asynchronous and blocking system calls

Most system calls, whether on Unix or other platforms, block (or "suspend") the calling thread until they complete, and continue its execution immediately following the call. Some systems also provide asynchronous (or *non-blocking*) forms of these calls; the kernel notifies the caller through some kind of out-of-band method when such a system call has completed.

Asynchronous system calls are generally much harder for the programmer to deal with than blocking calls.

5.3. Context switch

A context switch is the action of switching a CPU between executing one thread and another (or transferring control between them). This may involve crossing one or more protection boundary.

5.4. Critical section

A critical section of code is one in which data that may be accessed by other threads are inconsistent. At a higher level, a critical section can be viewed as a section of code in which a guarantee you make to other threads about the state of some data may not be true.

If other threads can access these data during a critical section, your program may not behave correctly. This may cause it to crash, lock up, produce incorrect results, or do just about any other unpleasant thing you care to imagine.

Other threads are generally denied access to inconsistent data during a critical section (usually through use of locks). If some of your critical sections are too long, however, it may result in your code performing poorly.

5.5. Lightweight process

A lightweight process (also known in some implementations, confusingly, as a *kernel thread*) is a schedulable entity that the kernel is aware of. On most systems, it consists of some execution context and some accounting information (i.e. much less than a full-blown process).

Several operating systems allow lightweight processes to be "bound" to particular CPUs; this guarantees that those threads will only execute on the specified CPUs.

5.6. MT safety

If some piece of code is described as MT-safe, this indicates that it can be used safely within a multithreaded program, *and* that it supports a "reasonable" level of concurrency. This isn't very interesting; what you, as a programmer using threads, need to worry about is code that is *not* MT-safe. MT-unsafe code may use global and/or static data. If you need to call MT-unsafe code from within a multithreaded program, you may need to go to some effort to ensure that only one thread calls that code at any time.

Wrapping a global lock around MT-unsafe code will generally let you call it from within a multithreaded program, but since this does not permit concurrent access to that code, it is not considered to make it MT-safe.

If you are trying to write MT-safe code using POSIX threads, you need to worry about a few issues such as dealing correctly with locks across calls to `fork`(2) (if you are wondering what to do, read about the `pthread_atfork`(3) library call).

5.7. Protection boundary

A protection boundary protects one software subsystem on a computer from another, in such a way that only data that is explicitly shared across such a boundary is accessible to the entities on both sides. In general, all code within a protection boundary will have access to all data within that boundary.

The canonical example of a protection boundary on most modern systems is that between processes and the kernel. The kernel is protected from processes, so that they can only examine or change its internal state in certain strictly-defined ways.

Protection boundaries also exist between individual processes on most modern systems. This prevents one buggy or malicious process from wreaking havoc on others.

Why are protection boundaries interesting? Because transferring control across them is expensive; it takes a lot of time and work.

5.8. Scheduling

Scheduling involves deciding what thread should execute next on a particular CPU. It is usually also taken as involving the context switch to that thread.

6. What are the different kinds of threads?

There are two main kinds of threads implementations:

- User-space threads, and
- Kernel-supported threads.

There are several sets of differences between these different threads implementations.

6.1. Architectural differences

User-space threads live without any support from the kernel; they maintain all of their state in user space. Since the kernel does not know about them, they cannot be scheduled to run on multiple processors in parallel.

Kernel-supported threads fall into two classes.

- In a "pure" kernel-supported system, the kernel is responsible for scheduling all threads.
- Systems in which the kernel cooperates with a user-level library to do scheduling are known as *two-level*, or *hybrid*, systems. Typically, the kernel schedules LWPs, and the user-level library schedules threads onto LWPs.

Because of its performance problems (caused by the need to cross the user/kernel protection boundary twice for *every* thread context switch), the former class has fewer members than does the latter (at least on Unix variants). Both classes allow threads to be run across multiple processors in parallel.

6.2. Performance differences

In terms of context switch time, user-space threads are the fastest, with two-level threads coming next (all other things being equal). However, if you have a multiprocessor, user-level threads can only be run on a single CPU, while both two-level and pure kernel-supported threads can be run on multiple CPUs simultaneously.

6.3. Potential problems with functionality

Because the kernel does not know about user threads, there is a danger that ordinary blocking system calls will block the entire process (this is *bad*) rather than just the calling thread. This means that user-space threads libraries need to jump through hoops in order to provide "blocking" system calls that don't block the entire process.

This problem also exists with two-level kernel-supported threads, though it is not as acute as for user-level threads. What usually happens here is that system calls block entire LWPs. This means that if more

threads exist than do LWPs and all of the LWPs are blocked in system calls, then other threads that could potentially make forward progress are prevented from doing so.

The Solaris threads library provides a reasonable solution to this problem. If the kernel notices that all LWPs in a process are blocked, it sends a signal to the process. This signal is caught by the user-level threads library, which can create another LWP so that the process will continue to make progress.

7. Where can I find books on threads?

There are several books available on programming with threads, with more due out in the near future. Note also that the programmer's manuals that come with most systems that provide threads packages will have sections on using those threads packages.

7.1. POSIX-style threads

David R. Butenhof, *Programming with POSIX Threads*. Addison-Wesley, ISBN 0-201-63392-2.

> This book gives a comprehensive and well-structured overview of programming with POSIX threads, and is a good text for the working programming to work from. Detailed examples and discussions abound.

Steve Kleiman, Devang Shah and Bart Smaalders, *Programming With Threads*. SunSoft Press, ISBN 0-13-172389-8. <URL: http://www.sun.com/smi/ssoftpress/books/Kleiman/Kleiman.html>

> This book goes into considerably greater depth than the other SunSoft Press offering (see below), and is also recommended for the working programmer who expects to deal with threads on a day-to-day basis. It includes many detailed examples.

Bil Lewis and Daniel J. Berg, *Threads Primer*. SunSoft Press, ISBN 0-13-443698-9. <URL: http://www.sun.com/smi/ssoftpress/books/Lewis/Lewis.html>

> This is a good introduction to programming with threads for programmers and managers. It concentrates on UI and POSIX threads, but also covers use of OS/2 and WIN32 threads.

Charles J. Northrup, *Programming With Unix Threads*. John Wiley & Sons, ISBN 0-471-13751-0. <URL: http://www.wiley.com/compbooks/catalog/14/13751-0.html>

> This book details the UI threads interface, focusing mostly on the Unixware implementation. This is an introductory book.

7.2. Microsoft-style threads

Jim Beveridge, Robert Wiener, *Multithreading Applications in Win32*. Addison-Wesley, ISBN 0-201-44234-5. <URL: http://www.aw.com/devpress/titles/44234.html>.

> Seasoned Win32 programmers, neophytes, and programmers being dragged kicking and screaming from the Unix world are all likely to find this book a useful resource. It doubles as primer and reference on writing and debugging robust multithreaded code, and provides a thorough exposition on the subject.

Len Dorfman, Marc J. Neuberger, *Effective Multithreading with OS/2*. Publisher and ISBN unknown.

> This book covers the OS/2 threads API and contains many examples, but doesn't have much by way of concepts.

Thuan Q. Pham, Pankaj K. Garg, *Multithreaded Programming with Windows NT*. Prentice Hall, ISBN 0-131-20643-5. <URL: http://www.prenhall.com/allbooks/ptr_0131206435.html>

> Not surprisingly, this book focuses on WIN32 threads, but it also mentions other libraries in passing. It also deals with some relatively advanced topics, and has a thorough bibliography.

7.3. Books on implementations

If you are interested in how modern operating systems support threads and multiprocessors, there are a few excellent books available that may be of interest to you.

Curt Schimmel, *Unix Systems for Modern Architectures*. Addison-Wesley, ISBN 0-201-63338-8. <URL: http://www.aw.com/cp/schimmel.html>

> This book gives a lucid account of the work needed to get Unix (or, for that matter, more or less anything else) working on a modern system that incorporates multiple processors, each with its own cache. While it has some overlap with the Vahalia book (see below), it has a smaller scope, and thus deals with shared topics in more detail.

Uresh Vahalia, *Unix Internals: the New Frontiers*. Prentice Hall, ISBN 0-13-101908-2. <URL: http://www.prenhall.com/013/101907/10190-7.html>

> This is the best kernel internals book currently available. It deals extensively with building multithreaded kernels, implementing LWPs, and scheduling on multiprocessors. Given a choice, I would buy *both* this and the Schimmel book.

Ben Catanzaro, *Multiprocessor System Architectures*. SunSoft Press, ISBN 0-13-089137-1. <URL: http://www.sun.com/smi/ssoftpress/books/Catanzaro/Catanzaro.html>

> I don't know much about this book, but it deals with both the hardware and software (kernel and user) architectures used to put together modern multiprocessor systems.

7.4. The POSIX threads standard

To order ISO/IEC standard 9945-1:1996, which is also known as ANSI/IEEE POSIX 1003.1-1995 (and includes 1003.1c, the part that deals with threads), you can call +1-908-981-1393. The document reference number in the IEEE publications catalogue is SH 94352-NYF, and the price to US customers is $120 (shipping overseas costs extra).

Unless you are implementing a POSIX threads package, you should not ever need to look at the POSIX threads standard. It is the last place you should look if you wish to learn about threads!

Neither IEEE nor ISO makes standards available for free; please do not ask whether the POSIX threads standard is available on the Web. It isn't.

8. Where can I obtain training on using threads?

Organisation	Contact	Description
Sun Microsystems	<training_seats@Sun.COM> +1-408-276-3630	Classes at Sun and on-site classes
Lambda Computer Science (Bil Lewis)	<URL: http://www.lambdaCS.com> +1-415-328-8952	Seminars and on-site classes
Phoenix Technologies (Chris Crenshaw)	<phnxtech@attmail.com> +1-908-286-2118	
Marc Staveley	<marc@staveley.com>	

9. (Unix) Are there any freely-available threads packages?

- Xavier Leroy <xleroy@inria.fr> has written a POSIX threads implementation for Linux 2.x that uses pure kernel-supported threads. While the package is currently in alpha testing, it is allegedly very stable. For more information, see <URL: http://pauillac.inria.fr/~xleroy/linuxthreads>.
- Michael T. Peterson <mtp@big.aa.net> has written a user-space POSIX and DCE threads package for Intel-based Linux systems; it is called PCthreads. See <URL: http://www.aa.net/~mtp/PCthreads.html> for more information.
- Christopher Provenzano <proven@mit.edu> has written a fairly portable implementation of draft 8 of the POSIX threads standard. See <URL: http://www.mit.edu:8001/people/proven/pthreads.html> for further details. *Note*: as far as I can see, development of this library has halted (at least temporarily), and it still contains many serious bugs.
- Georgia Tech's OS group has a fairly portable user-level threads implementation of the Mach C threads package. It is called Cthreads, and can be found at <URL: ftp://ftp.cc.gatech.edu/pub/groups/systems/Falcon/cthreads_distribution.tar.gz>.
- Frank Müller, of the POSIX / Ada-Runtime Project (PART) has made available an implementation of draft 6 of the POSIX 1003.4a Pthreads specification, which runs under SunOS 4, Solaris 2.x, SCO Unix, FreeBSD and Linux. For more information, see <URL: file://ftp.cs.fsu.edu/pub/PART/PTHREADS/pthreads_ANNOUNCE>.
- Elan Feingold has written a threads package called ethreads; I don't know anything about it, other than that it is available from <URL: ftp://frmap711.mathp7.jussieu.fr/pub/scratch/rideau/misc/threads/ethreads/ethreads.tgz>.
- QuickThreads is a toolkit for building threads packages, written by David Keppel <pardo@cs.washington.edu>. It is available from <URL: ftp://ftp.cs.washington.edu/pub/qt-001.tar.Z>, with an accompanying tech report at <URL: ftp://ftp.cs.washington.edu/tr/1993/05/UW-CSE-93-05-06.PS.Z>. The code as distributed includes ports for the Alpha, x86, 88000, MIPS, SPARC, VAX, and KSR1.

10. (DCE, POSIX, UI) Why does my threaded program not handle signals sensibly?

Signals and threads do not mix well. A lot of programmers start out by writing their code under the mistaken assumption that they can set a signal handler for each thread; this is not the way things work. You can *block* or *unblock* signals on a thread-by-thread basis, but this is not the same thing.

When it comes to dealing with signals, the best thing you can do is create a thread whose sole purpose is to handle signals for the entire process. This thread should loop calling `sigwait`(2); this allows it to deal with signals synchronously. You should also make sure that all threads (*including* the one that calls `sigwait`) have the signals you are interested in handling blocked. Handling signals synchronously in this way greatly simplifies things.

Note, also, that sending signals to other threads within your own process is not a friendly thing to do, unless you are careful with signal masks. For an explanation, see the section on asynchronous cancellation.

Finally, using `sigwait` and installing signals handlers for the signals you are `sigwait`ing for is a bad idea.

11. (DCE?, POSIX) Why does everyone tell me to avoid asynchronous cancellation?

Asynchronous cancellation of threads is, in general, evil. The reason for this is that it is usually (very) difficult to guarantee that the recipient of an asynchronous cancellation request will not be in a critical section. If a thread should die in the middle of a critical section, this will very likely cause your program to misbehave.

Code that can deal sensibly with asynchronous cancellation requests is *not* referred to as async-safe; that means something else (see the terminology section of the FAQ). You won't see much code around that handles asynchronous cancellation requests properly, and you shouldn't try write any of your own unless you have compelling reasons to do so. Deferred cancellation is your friend.

12. Why are reentrant library and system call interfaces good?

There are two approaches to providing system calls and library interfaces that will work with multithreaded programs. One is to simply wrap all the appropriate code with mutexes, thereby guaranteeing that only one thread will execute any such routine at a time.

While this approach mostly works, it provides terrible performance. For functions that maintain state across multiple invocations (e.g. `strtok`() and friends), this approach simply doesn't work at all, hence the existence of "`_r`" interfaces on many Unix systems (see below).

A better solution is to ensure that library calls can safely be performed by multiple threads at once.

12.1. (DCE, POSIX, UI) When should I use thread-safe "_r" library calls?

If your system provides threads, it will probably provide a set of thread-safe variants of standard C library routines. A small number of these are mandated by the POSIX standard, and many Unix vendors provide their own useful supersets, including functions such as `gethostbyname_r`().

Unfortunately, the supersets that different vendors support do not necessarily overlap, so you can only *safely* use the standard POSIX-mandated functions. The thread-safe routines are conceptually "cleaner" than their stateful counterparts, though, so it is good practice to use them wherever and whenever you can.

13. (POSIX) How can I perform a join on any thread?

UI threads allow programmers to join on any thread that happens to terminate by passing the appropriate argument to `thr_join`(). This is not possible under POSIX and, yes, there is a rationale behind the absence of this feature.

Unix programmers are used to being able to call `wait`() in such a way that it will return when "any" process exits, but expecting this to work for threads can cause confusion for programmers trying to use threads. The important thing to note here is that Unix processes are based around a notion of parent and child; this is a notion that is *not* present in most threads systems. Since threads don't contain this notion, joining on "any" thread could have the undesirable effect of having the join return once a completely unrelated thread happened to exit.

In many (perhaps even most) threaded applications, you do not want to be able to join with any thread in your process. Consider, for example, a library call that one of your threads might make, which in its turn might start a few threads and try to join on them. If another of your threads, joining on "any" thread, happened to join on one of the library call's threads, that would lead to incorrect program behaviour.

If you want to be able to join on any thread so that, for example, you can keep track of the number of running threads, you can achieve the same functionality by starting detached threads and having them decrememnt a (suitably locked, of course) counter as they exit.

14. (DCE, UI, POSIX) After I create a certain number of threads, my program crashes

By default, threads are created non-detached. You need to perform a join on each non-detached thread, or else storage will never be freed up when they exit. As an alternative, you can create detached threads, for which storage will be freed as soon as they exit. This latter approach is generally better; you shouldn't create non-detached threads unless you explicitly need to know when or if they exit.

15. Where can I find POSIX thread benchmarks?

I do not know of any benchmarks.

16. Does any DBMS vendor provide a thread-safe interface?

Oracle 7.3 and above provide thread-safe database client interfaces, as do Sybase System 11.1 and above, and Informix ESQL 7 and above.

If you are still using an older release of any of these products, it is possible to hack together a set of intermediate thread-safe interfaces to your database if you really need it, but this requires a moderately large amount of work.

17. Why is my threaded program running into performance problems?

There are many possible causes for performance problems in multithreaded programs. Given the scope for error, all I can do is detail a few common pitfalls briefly, and point you at the section of this FAQ on books about multithreaded programming.

- You can probably discount the performance of the threads package you are using almost straight away, unless it is a user-level package. If so, you may want to try to find out whether your whole process is blocking when you execute certain system calls. Otherwise, you should look at your own code unless you have a very strong reason for believing that there may be a problem with your threads package.
- Look at the granularity of your locks. If a single lock protects too much data for which there is contention, you will needlessly serialise your code. On the other hand, if your locks protect very

small amounts of data, you will spend too much time obtaining and releasing locks. If your vendor is worth their salt, they will have a set of tools available that allow you to profile your program's behaviour and look for areas of high contention for locks. Tools of this kind are invaluable.

18. What tools will help me to program with threads?

- The TNF Tools are a set of extensible tools that allow users to gather and analyse trace information from the Solaris kernel and from user processes and threads. They can be used to correlate user and kernel thread activity, and also to determine such things as lock hold times. They are available for free from Sun; for more information, see <URL: http://opcom.sun.ca/toolpages/tnftools.html>.
- The debugger that comes with the DevPro compiler set from Sun understands threads.
- GDB nominally understands threads, but only supports them (and in a flaky way) under some versions of Irix and a few other systems (mostly embedded machines).

19. What operating systems provide threads?

Vendor	OS version	Threads model	POSIX API	Notes
Digital	Digital UNIX 4.0	mixed	D4, 1c	
	Digital UNIX 3.2	kernel / bound	D4	
	OpenVMS 7.0 (Alpha)	see note 1	D4, 1c	user model without patch kit
	OpenVMS 7.0 (VAX)	user	D4, 1c	
	OpenVMS 6.2	user	D4	
HP	HP/UX 10.20	*?*	*?*	not yet announced
	HP/UX 10.10	user	D4	
IBM	AIX 4.1 & 4.2	kernel	D4, D7	
	AIX 3.5.x	user	DCE	
	OS/2	kernel	DCE	
Linux	Linux 1.2.13 and above	user / kernel	1c, DCE	see free implementations for several packages
	Linux 2.x	kernel	*n/a*	`clone`() interface
Microsoft	Windows NT & 95	kernel	DCE	DCE kits layer on top of WIN32 threads
SGI	Irix 6.2	mixed	see note 2	`sproc`() is still kernel / bound
	Irix 6.1	kernel / bound	*n/a*	`sproc`() interface only
Sun	Solaris 2.5 and above	mixed / system / bound	1c	
	Solaris 2.4	mixed / system / bound	D8	available from Sun only upon request

Solaris 2.x	mixed / system / bound	*n/a*	UI threads supported across all releases
SunOS 4.x	user	*n/a*	LWP only

Threads model	Meaning
user	a purely user-level threads system, with threads multiplexed atop a "traditional" Unix-style process
kernel	threads are "kernel entities" with no context switches taking place in user mode
/ bound	a thread may be explicitly bound to a particular processor
mixed	a mixed-mode scheduler where user threads are multiplexed across some number of LWPs
/ system	threads have "system" contention scope (a user thread may be permanently bound to an LWP)
/ bound	an LWP may be permanently bound to a particular processor

API	Meaning
n/a	no POSIX threads API provided
1c	conforms to the POSIX 1003.1c-1995 threads API
DCE	POSIX 1003.4a draft 4 API is available as part of the OSF DCE kit for the platform
D4	DCE threads (or something similar) is bundled with the system
D7	POSIX 1003.4a draft 7 API (similar to the final 1003.1c standard, but substantially different in some details)
D8	POSIX 1003.4a draft 8 API (identical in most respects to the 1003.1c standard, but with a few "gotchas")

1. OpenVMS 7.0 for Alpha shipped with kernel threads support disabled by default. The "mixed" threads model can be turned on by setting the `MULTITHREAD` `sysgen` parameter. With patch kit, the "mixed" or "user" model is determined by the main program binary (i.e. via the linker or the `threadcp` qualifier) in addition to `MULTITHREAD`.
2. SGI ships the POSIX 1003.1c API as a patch for Irix 6.2, but it will be bundled with future revisions of the OS.

20. What about other threads-related software?

- Douglas C. Schmidt <schmidt@cs.wustl.edu> has written a package called ACE (*Adaptive Communication Environment*). ACE supports multithreading on Unix and WIN32 platforms, and integrates popular IPC mechanisms (sockets, RPC, System V IPC) and a host of other features that C++ programmers will find useful. For details, see <URL: http://www.cs.wustl.edu/~schmidt/ACE.html>.

21. Where can I find other information on threads?

- The most comprehensive collection of threads-related information on the Web is at Sun's threads page, at <URL: http://www.sun.com/sunsoft/Products/Developer-products/sig/threads>.
- IBM has a thorough treatment of AIX 4.1 threads (based on POSIX draft 7) at <URL: http://developer.austin.ibm.com/sdp/library/ref/about4.1/df4threa.html>.
- Digital has a brief overview of threads in Digital UNIX at <URL: http://www.unix.digital.com/unix/smp>.
- A bibliography on threads is available at <URL: http://liinwww.ira.uka.de/bibliography/Os/threads.html>.
- Tom Wagner <wagner@cs.umass.edu> and Don Towsley have written an introductory tutorial on programming with POSIX threads at <URL: http://centaurus.cs.umass.edu/~wagner/threads_html/tutorial.html>

21.1. Articles appearing in periodicals

- An introduction to programming with threads, at <URL: http://www.sun.com/sunworldonline/swol-02-1996/swol-02-threads.html>, from SunWorld Online's February 1996 issue
- An introduction to programming with threads, at <URL: http://developer.austin.ibm.com/sdp/library/aixpert/nov94/aixpert_nov94_intrmult.html>, from AIXpert Magazine's November 1994 issue
- Porting DCE threads to AIX 4.1 (POSIX draft 7), at <URL: http://www.developer.ibm.com/sdp/library/aixpert/aug94/aixpert_aug94_PTHREADS.html>, from AIXpert Magazine's August 1994 issue
- A less thorough introduction to programming with threads, at <URL: http://developer.austin.ibm.com/sdp/library/aixpert/aug95/aixpert_aug95_thread.html>, from AIXpert Magazine's August 1995 issue
- Using signals with POSIX threads, at <URL: http://developer.austin.ibm.com/sdp/library/aixpert/aug95/aixpert_aug95_signal.html>, from AIXpert Magazine's August 1995 issue

22. Notice of copyright and permissions

fitness for a particular purpose or non-infringement. The Author shall not be liable for any damages suffered as a result of using information distributed in These Documents or any derivatives.

These Documents may be reproduced and distributed in whole or in part, subject to the following conditions:

- This copyright and permission notice must be retained on all complete or partial copies of These Articles.
- These Articles may be copied or distributed in part or in full for personal or educational use. Any translation, derivative work, or copies made for other purposes must be approved by the copyright holder before distribution, unless otherwise stated.
- If you distribute These Articles, instructions for obtaining the complete current versions of them free or at cost price must be included. Redistributors must make reasonable efforts to maintain current copies of These Articles.

Exceptions to these rules may be granted, and I shall be happy to answer any questions about this copyright notice - write to Bryan O'Sullivan, PO Box 62215, Sunnyvale, CA 94088-2215, USA or email <bos@serpentine.com>. These restrictions are here to protect the contributors, not to restrict you as educators, professionals and learners.

The LinuxThreads library

LinuxThreads is an implementation of the Posix 1003.1c thread package for Linux.

Unlike other implementations of Posix threads for Linux, LinuxThreads provides **kernel-level** threads: threads are created with the new `clone()` system call and all scheduling is done in the kernel.

The main strength of this approach is that it can take full advantage of multiprocessors. It also results in a simpler, more robust thread library, especially w.r.t. blocking system calls.

For more information, see the LinuxThreads README and the LinuxThreads Frequently Asked Questions.

Warning

LinuxThreads is now closely integrated in the glibc GNU C library. The initial author of LinuxThreads (Xavier Leroy) is no longer active on LinuxThreads; maintenance and further developments are mostly handled by Kaz Kylheku and the glibc team, in particular Ulrich Drepper.

The information on this Web page has not been updated in a while and may not be 100% up to date (except the FAQ, which is still actively maintained). The glibc mailing lists often contain more up-to-date information.

General questions about LinuxThreads and programming with POSIX threads should be posted on the `comp.programming.threads` newsgroup.

Bug reports should be sent directly to the glibc development team using the `glibcbug` script that is installed on your machine along with glibc itself.

Obtaining LinuxThreads

For Linux systems based on glibc 2 (e.g. RedHat 5.0 and later): the `glibc` 2 version of LinuxThreads is distributed along with `glibc` 2 itself. If you have installed a Linux distribution based on `glibc` 2 (such as RedHat 5.0 and later, or indeed most major Linux distributions these days), you already have LinuxThreads on your computer. Congratulations. Otherwise, both `glibc` 2 and the associated LinuxThreads distribution can be found on Cygnus/RedHat source collection, and also on any FTP site that mirrors GNU software (the Cygnus/RedHat site is sometimes more up-to-date than the GNU sites).

For Linux systems based on libc 5 (e.g. old versions of Slackware): The source distribution of LinuxThreads for libc 5 is available at `ftp://ftp.inria.fr/INRIA/Projects/cristal/Xavier.Leroy/linuxthreads.tar.gz`. Please note that the libc 5 version of LinuxThreads is obsolete and no longer maintained. An upgrade to glibc 2 is highly recommended for reliable multithreading programming.

Debugging threaded programs with GDB: Recent versions of gdb and glibc support the debugging of multithreaded programs. For instance, RedHat 6.2 and later come with a thread-aware version of gdb.

For older systems: see H.J.Lu's patches for gdb, which include thread support and more. The very first attempt at multi-threaded debugging for LinuxThreads was the patches for gdb 4.17 developed at The Open Group, Grenoble, but this is now largely obsolete. The Open Group patches are for glibc 2. Kaz Kylheku back-ported them to libc 5: patch for LinuxThreads 0.71/libc5 and patch for GDB 4.17.

Documentation

- Two FAQs from `comp.programming.threads`: O'Sullivan's FAQ and Lewis' FAQ.
- Getting Started With POSIX Threads, by Tom Wagner and Don Towsley.
- Sun's pthread pages.
- LinuxThreads comes with a complete set of man pages.
- Code samples from Programming with POSIX® Threads, D. R. Butenhof, Addison-Wesley.
- Code samples, sources, infos from Multithreading Programming Techniques, S. Prasad, McGraw Hill.

LinuxThreads-related software

- Richard Neitzel's libaio, a library of POSIX.1b compliant asynchronous I/O functions based on LinuxThreads.
- The Adaptive Communication Environment (ACE) contains C++ wrappers for LinuxThreads and much, much more.
- Jim Doyle's DCE Threads Emulation Package for LinuxThreads allows applications written with DCE threads (a.k.a. draft 4 POSIX threads) to run easily on top of LinuxThreads.
- The AOL multithreaded Web server uses LinuxThreads for its Linux version.
- Software AG's port of DCOM for Linux also uses LinuxThreads.
- LinuxThreads is used in MpegTV Player, a commercial/shareware real-time MPEG Video player (with audio/sync) and Video-CD player for Linux.
- Com-One's Video live server runs under Linux and uses LinuxThreads.
- Blackdown's port of the Java Development Kit 1.2 for Linux supports "native" threads built on top of LinuxThreads as an alternative to the JDK's own "green" threads. The Kaffe Java implementation does the same.
- GNAT, the GNU Ada95 compiler, comes with a "native" run-time system that implements Ada tasks using LinuxThreads.
- Ted Baker's FLORIST package is a POSIX.5 library for GNAT that can be used as an Ada95 interface to LinuxThreads and other Linux system calls. It also includes a POSIX conformance test kit.
- Proxy, a threaded IP filtering proxy server.

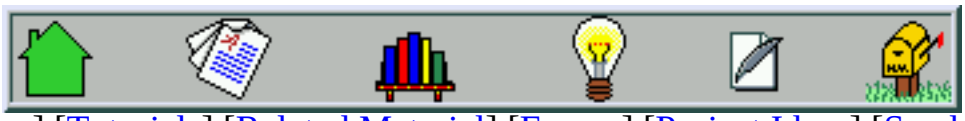
Xavier.Leroy@inria.fr

Lynx

Lynx is a text browser for the World Wide Web. Lynx 2.8.3 runs on Un*x, VMS, Windows 95/98/NT but **not** 3.1 or 3.11, on DOS (386 or higher) and OS/2 EMX. The current developmental version is also available for testing. Ports to Mac are in beta test.

- How to get Lynx, and much more information, is available at Lynx links.
- Many user questions are answered in the online help provided with Lynx. Press the '?' key to find this help.
- If you are encountering difficulty with Lynx you may write to **lynx-dev@sig.net**. Be as detailed as you can about the URL where you were on the Web when you had trouble, what you did, what Lynx version you have (try '=' key), and what OS you have. If you are using an older version, you may well need to upgrade.

Maintained by lynxdev@browser.org.



v1.0.2

Compiling "C" And "C++" Programs On Unix Systems - gcc/g++

Table Of Contents:

Preface - How To Read This Document

This document tries to give the reader basic knowledge in compiling C and C++ programs on a Unix system. If you've no knowledge as to how to compile C programs under Unix (for instance, you did that until now on other operating systems), you'd better read this tutorial first, and then write a few programs before you try to get to gdb, makefiles or C libraries.

If you're already familiar with that, it's recommended to learn about makefiles, and then go and learn other C programming topics and practice the usage of makefiles, before going on to read about C libraries. This last issue is only relevant to larger projects, while makefiles make sense even for a small program composed of but a few source files.

As a policy, we'll stick with the basic features of programming tools mentioned here, so that the information will apply to more then a single tool version. This way, you might find the information here useful, even if the system you're using does not have the GNU tools installed.

In this lovely tutorial, we'll deal with compilation of a C program, using the compiler directly

from the command line. It might be that you'll eventually use a more sophisticated interface (an IDE - Integrated Development Environment) of some sort, but the common denominator you'll always find is the plain command line interface. Further more, even if you use an IDE, it could help you understand how things work "behind the scenes". We'll see how to compile a program, how to combine several source files into a single program, how to add debug information and how to optimize code.

Compiling A Single-Source "C" Program

The easiest case of compilation is when you have all your source code set in a single file. This removes any unnecessary steps of synchronizing several files or thinking too much. Lets assume there is a file named 'single_main.c' that we want to compile. We will do so using a command line similar to this:

```
cc single_main.c
```

Note that we assume the compiler is called "cc". If you're using a GNU compiler, you'll write 'gcc' instead. If you're using a Solaris system, you might use 'acc', and so on. Every compiler might show its messages (errors, warnings, etc.) differently, but in all cases, you'll get a file 'a.out' as a result, if the compilation completed successfully. Note that some older systems (e.g. SunOs) come with a C compiler that does not understand ANSI-C, but rather the older 'K&R' C style. In such a case, you'll need to use gcc (hopefully it is installed), or learn the differences between ANSI-C and K&R C (not recommended if you don't *really* have to), or move to a different system.

You might complain that 'a.out' is a too generic name (where does it come from anyway? - well, that's a historical name, due to the usage of something called "a.out format" for programs compiled on older Unix systems). Suppose that you want the resulting program to be called "single_main". In that case, you could use the following line to compile it:

```
cc single_main.c -o single_main
```

Every compiler I've met so far (including the glorious gcc) recognized the '-o' flag as "name the resulting executable file 'single_main'".

Running The Resulting Program

Once we created the program, we wish to run it. This is usually done by simply typing its name, as in:

```
single_main
```

However, this requires that the current directory be in our PATH (which is a variable telling our Unix shell where to look for programs we're trying to run). In many cases, this directory is not placed in our PATH. Aha! - we say. Then lets show this computer who is smarter, and thus we try:

```
./single_main
```

This time we explicitly told our Unix shell that we want to run the program from the current directory. If we're lucky enough, this will suffice. However, yet one more obstacle could block our path - file permission flags.

When a file is created in the system, it is immediately given some access permission flags. These flags tell the system who should be given access to the file, and what kind of access will be given to them. Traditional Unix systems use 3 kinds of entities to which they grant (or deny) access: The user which owns the file, the group which owns the file, and everybody else. Each of these entities may be given access to read the file ('r'), write to the file ('w') and execute the file ('x').

Now, when the compiler created the program file for us, we became owners of the file. Normally, the compiler would make sure that we get all permissions to the file - read, write and execute. It might be, thought that something went wrong, and the permissions are set differently. In that case, we can set the permissions of the file properly (the owner of a file can normally change the permission flags of the file), using a command like this:

```
chmod u+rwx single_main
```

This means "the user ('u') should be given ('+') permissions read ('r'), write ('x') and execute ('x') to the file 'single_main'. Now we'll surely be able to run our program. Again, normally you'll have no problem running the file, but if you copy it to a different directory, or transfer it to a different computer over the network, it might loose its original permissions, and thus you'll need to set them properly, as shown above. Note too that you cannot just move the file to a different computer an expect it to run - it has to be a computer with a matching operating system (to understand the executable file format), and matching CPU architecture (to understand the machine-language code that the executable file contains).

Finally, the run-time environment has to match. For example, if we compiled the program on an operating system with one version of the standard C library, and we try to run it on a version with an incompatible standard C library, the program might crush, or complain that it cannot find the relevant C library. This is especially true for systems that evolve quickly (e.g. Linux with libc5 vs. Linux with libc6), so beware.

Creating Debug-Ready Code

Normally, when we write a program, we want to be able to debug it - that is, test it using a debugger that allows running it step by step, setting a break point before a given command is executed, looking at contents of variables during program execution, and so on. In order for the debugger to be able to relate between the executable program and the original source code, we need to tell the compiler to insert information to the resulting executable program that'll help the debugger. This information is called "debug information". In order to add that to our program, lets compile it differently:

```
cc -g single_main.c -o single_main
```

The '-g' flag tells the compiler to use debug info, and is recognized by mostly any compiler out there. You will note that the resulting file is much larger then that created without usage of the '-g' flag. The difference in size is due to the debug information. We may still remove this debug information using the `strip` command, like this:

```
strip single_main
```

You'll note that the size of the file now is even smaller then if we didn't use the '-g' flag in the first place. This is because even a program compiled without the '-g' flag contains some symbol information (function names, for instance), that the `strip` command removes. You may want to read `strip`'s manual page (man strip) to understand more about what this command does.

Creating Optimized Code

After we created a program and debugged it properly, we normally want it to compile into an efficient code, and the resulting file to be as small as possible. The compiler can help us by optimizing the code, either for speed (to run faster), or for space (to occupy a smaller space), or some combination of the two. The basic way to create an optimized program would be like this:

```
cc -O single_main.c -o single_main
```

The '-O' flag tells the compiler to optimize the code. This also means the compilation will take longer, as the compiler tries to apply various optimization algorithms to the code. This optimization is supposed to be conservative, in that it ensures us the code will still perform the same functionality as it did when compiled without optimization (well, unless there are bugs in our compiler). Usually can define an optimization level by adding a number to the '-O' flag. The higher the number - the better optimized the resulting program will be, and the slower the compiler will complete the compilation. One should note that because optimization alters the code in various ways, as we increase the optimization level of the code, the chances are higher that an improper optimization will actually alter our code, as some of them tend to be

non-conservative, or are simply rather complex, and contain bugs. For example, for a long time it was known that using a compilation level higher then 2 (or was it higher then 3?) with gcc results bugs in the executable program. After being warned, if we still want to use a different optimization level (lets say 4), we can do it this way:

```
cc -O4 single_compile.c -o single_compile
```

And we're done with it. If you'll read your compiler's manual page, you'll soon notice that it supports an almost infinite number of command line options dealing with optimization. Using them properly requires thorough understanding of compilation theory and source code optimization theory, or you might damage your resulting code. A good compilation theory course (preferably based on "the Dragon Book" by Aho, Sethi and Ulman) could do you good.

Getting Extra Compiler Warnings

Normally the compiler only generates error messages about erroneous code that does not comply with the C standard, and warnings about things that usually tend to cause errors during runtime. However, we can usually instruct the compiler to give us even more warnings, which is useful to improve the quality of our source code, and to expose bugs that will really bug us later. With gcc, this is done using the '-W' flag. For example, to get the compiler to use all types of warnings it is familiar with, we'll use a command line like this:

```
cc -Wall single_source.c -o single_source
```

This will first annoy us - we'll get all sorts of warnings that might seem irrelevant. However, it is better to eliminate the warnings then to eliminate the usage of this flag. Usually, this option will save us more time than it will cause us to waste, and if used consistently, we will get used to coding proper code without thinking too much about it. One should also note that some code that works on some architecture with one compiler, might break if we use a different compiler, or a different system, to compile the code on. When developing on the first system, we'll never see these bugs, but when moving the code to a different platform, the bug will suddenly appear. Also, in many cases we eventually will want to move the code to a new system, even if we had no such intentions initially.

Note that sometimes '-Wall' will give you too many errors, and then you could try to use some less verbose warning level. Read the compiler's manual to learn about the various '-W' options, and use those that would give you the greatest benefit. Initially they might sound too strange to make any sense, but if you are (or when you will become) a more experienced programmer, you will learn which could be of good use to you.

Compiling A Single-Source "C++" Program

Now that we saw how to compile C programs, the transition to C++ programs is rather simple. All we need to do is use a C++ compiler, in place of the C compiler we used so far. So, if our program source is in a file named 'single_main.cc' ('cc' to denote C++ code. Some programmers prefer a suffix of 'C' for C++ code), we will use a command such as the following:

```
g++ single_main.cc -o single_main
```

Or on some systems you'll use "CC" instead of "g++" (for example, with Sun's compiler for Solaris), or "aCC" (HP's compiler), and so on. You would note that with C++ compilers there is less uniformity regarding command line options, partially because until recently the language was evolving and had no agreed standard. But still, at least with g++, you will use "-g" for debug information in the code, and "-O" for optimization.

Compiling A Multi-Source "C" Program

So you learned how to compile a single-source program properly (hopefully by now you played a little with the compiler and tried out a few examples of your own). Yet, sooner or later you'll see that having all the source in a single file is rather limiting, for several reasons:

- As the file grows, compilation time tends to grow, and for each little change, the whole program has to be re-compiled.
- It is very hard, if not impossible, that several people will work on the same project together in this manner.
- Managing your code becomes harder. Backing out erroneous changes becomes nearly impossible.

The solution to this would be to split the source code into multiple files, each containing a set of closely-related functions (or, in C++, all the source code for a single class).

There are two possible ways to compile a multi-source C program. The first is to use a single command line to compile all the files. Suppose that we have a program whose source is found in files "main.c", "a.c" and "b.c" (found in directory "multi-source" of this tutorial). We could compile it this way:

```
cc main.c a.c b.c -o hello_world
```

This will cause the compiler to compile each of the given files separately, and then link them all together to one executable file named "hello_world". Two comments about this program:

1. If we define a function (or a variable) in one file, and try to access them from a second file, we need to declare them as external symbols in that second file. This is done using the C `"extern"` keyword.

2. The order of presenting the source files on the command line may be altered. The compiler (actually, the linker) will know how to take the relevant code from each file into the final program, even if the first source file tries to use a function defined in the second or third source file.

The problem with this way of compilation is that even if we only make a change in one of the source files, all of them will be re-compiled when we run the compiler again.

In order to overcome this limitation, we could divide the compilation process into two phases - compiling, and linking. Lets first see how this is done, and then explain:

```
cc -c main.cc
cc -c a.c
cc -c b.c
cc main.o a.o b.o -o hello_world
```

The first 3 commands have each taken one source file, and compiled it into something called "object file", with the same names, but with a ".o" suffix. It is the "-c" flag that tells the compiler only to create an object file, and not to generate a final executable file just yet. The object file contains the code for the source file in machine language, but with some unresolved symbols. For example, the "main.o" file refers to a symbol named "func_a", which is a function defined in file "a.c". Surely we cannot run the code like that. Thus, after creating the 3 object files, we use the 4th command to link the 3 object files into one program. The linker (which is invoked by the compiler now) takes all the symbols from the 3 object files, and links them together - it makes sure that when "func_a" is invoked from the code in object file "main.o", the function code in object file "a.o" gets executed. Further more, the linker also links the standard C library into the program, in this case, to resolve the "printf" symbol properly.

To see why this complexity actually helps us, we should note that normally the link phase is much faster then the compilation phase. This is especially true when doing optimizations, since that step is done before linking. Now, lets assume we change the source file "a.c", and we want to re-compile the program. We'll only need now two commands:

```
cc -c a.c
cc main.o a.o b.o -o hello_world
```

In our small example, it's hard to notice the speed-up, but in a case of having few tens of files each containing a few hundred lines of source-code, the time saving is significant; not to mention even larger projects.

Getting a Deeper Understanding - Compilation Steps

Now that we've learned that compilation is not just a simple process, lets try to see what is the complete list of steps taken by the compiler in order to compile a C program.

1. `Driver` - what we invoked as "cc". This is actually the "engine", that drives the whole set of tools the compiler is made of. We invoke it, and it begins to invoke the other tools one by one, passing the output of each tool as an input to the next tool.
2. `C Pre-Processor` - normally called "cpp". It takes a C source file, and handles all the pre-processor definitions (#include files, #define macros, conditional source code inclusion with #ifdef, etc.) You can invoke it separately on your program, usually with a command like:

   cc -E single_source.c

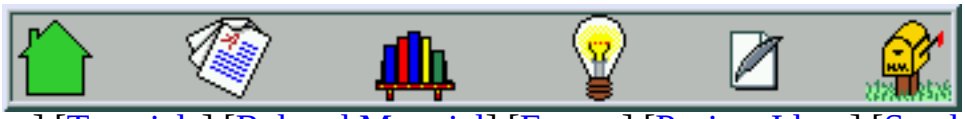
   Try this and see what the resulting code looks like.
3. `The C Compiler` - normally called "cc1". This is the actual compiler, that translates the input file into assembly language. As you saw, we used the "-c" flag to invoke it, along with the C Pre-Processor, (and possibly the optimizer too, read on), and the assembler.
4. `Optimizer` - sometimes comes as a separate module and sometimes as the found inside the compiler module. This one handles the optimization on a representation of the code that is language-neutral. This way, you can use the same optimizer for compilers of different programming languages.
5. `Assembler` - sometimes called "as". This takes the assembly code generated by the compiler, and translates it into machine language code kept in object files. With gcc, you could tell the driver to generated only the assembly code, by a command like:

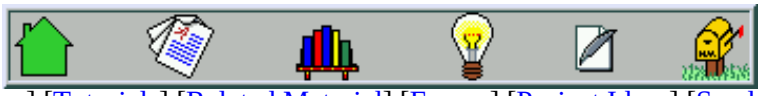
   ```
   cc -S single_source.c
   ```

6. `Linker-Loader` - This is the tool that takes all the object files (and C libraries), and links them together, to form one executable file, in a format the operating system supports. A Common format these days is known as "ELF". On SunOs systems, and other older systems, a format named "a.out" was used. This format defines the internal structure of the executable file - location of data segment, location of source code segment, location of debug information and so on.

As you see, the compilation is split in to many different phases. Not all compiler employs exactly the same phases, and sometimes (e.g. for C++ compilers) the situation is even more complex. But the basic idea is quite similar - split the compiler into many different parts, to give the programmer more flexibility, and to allow the compiler developers to re-use as many modules as possible in different compilers for different languages (by replacing the

preprocessor and compiler modules), or for different architectures (by replacing the assembly and linker-loader parts).





Debugging "C" And "C++" Programs Using "gdb"

Table Of Contents:

Why Use A Debugger?

This might sound silly, but I've heard of many programmers that claim they do not need a debugger. They simply don't create bugs. Well, one thing is sure - either they've no idea what they are saying, or they just never put their code to real test. Or maybe they're indeed as gifted as they claim. Unfortunately, most of us tend to have bugs in our code. We could use printing commands to test our code, or we could use a debugger. Many times our code might seem to work correctly, because we didn't test it under enough scenarios. Other times we know there's a bug, but by just reading the code we don't notice it is there. Thus, we should develop a habit of launching a debugger when we get into trouble. It shouldn't come instead of making an effort to write correct code, to add many tests in the code for invalid function arguments, NULL pointers, etc. But when we're in trouble, it's probably our best shot.

The explanations given here are specific to the "gdb" debugger, since there are no real standards regarding the activation and usage of debuggers, but once you know what features to expect from a debugger, it's not too hard to adapt your knowledge to different debuggers.

Invoking the "gdb" Debugger

Before invoking the debugger. make sure you compiled your program (all its modules, as well as during linking) with the "-g" flag. Otherwise, life will be tough. Lets compile the "debug_me.c" program, and then invoke "gdb" to debug it:

```
gcc -g debug_me.c -o debug_me
gdb debug_me
```

Note that we run the program from the same directory it was compiled in, otherwise gdb won't find the source file, and thus won't be able to show us where in the code we are at a given point. It is possible to ask gdb to search for extra source files in some directory after launching it, but for now, it's easier to just invoke it from the correct directory.

Running A Program Inside The Debugger

Once we invoked the debugger, we can run the program using the command "run". If the program requires command line parameters (like our debug_me program does), we can supply them to the "run" command of gdb. For example:

```
run "hello, world" "goodbye, world"
```

Note that we used quotation marks to denote that "hello, world" is a single parameter, and not to separate parameters (the debugger assumes white-space separates the parameters).

Setting Breakpoints

The problem with just running the code is that it keeps on running until the program exits, which is usually too late. For this, breakpoints are introduced. A break point is a command for the debugger to stop the execution of the program before executing a specific source line.We can set break points using two methods:

1. Specifying a specific line of code to stop in:

   ```
   break debug_me.c:9
   ```

   Will insert a break point right before checking the command line arguments in our program (see the file supplied with this tutorial).
2. Specifying a function name, to break every time it is being called:

   ```
   break main
   ```

   this will set a break point right when starting the program (as the function "main" gets executed automatically on the beginning of any C or C++ program).

Stepping A Command At A Time

So lets see, we've invoked gdb, then typed:

```
break main
run "hello, world" "goodbye, world"
```

Then the debugger gave something like the following:

```
Starting program: /usr/home/choo/work/c-on-unix/debug_me
warning: Unable to find dynamic linker breakpoint function.
warning: GDB will be unable to debug shared library initializers
warning: and track explicitly loaded dynamic code.

Breakpoint 1, main (argc=1, argv=0xbffffba4) at debug_me.c:9
9           if (argc < 2) { /* 2 - 1 for program name (argv[0]) and one for a param.
*/
(gdb)
```

Note that you won't always get the warnings i got - it just goes to show you how lousy my system setup is. In any case, these warnings are not relevant to our code, as we do not intend to debug any shared libraries.

Now we want to start running the program slowly, step by step. There are two options for that:

1. "next" - causes the debugger to execute the current command, and stop again, showing the next command in the code to be executed.

2. "step" - causes the debugger to execute the current command, and if it is a function call - break at the beginning of that function. This is useful for debugging nested code.

Now is your time to experiment with these options with our debug program, and see how they work. It is also useful to read the debuggers help, using the command "help break" and "help breakpoints" to learn how to set several breakpoints, how to see what breakpoints are set, how to delete breakpoints, and how to apply conditions to breakpoints (i.e. make them stop the program only if a certain expression evaluates to "true" when the breakpoint is reached).

Printing Variables And Expressions

Without being able to examine variables contents during program execution, the whole idea of using a debugger is quite lost. You can print the contents of a variable with a command like this:

```
print i
```

And then you'll get a message like:

```
$1 = 0
```

which means that "i" contains the number "0".
Note that this requires "i" to be in scope, or you'll get a message such as:

```
No symbol "i" in current context.
```

For example, if you break inside the "print_string" function and try to print the value of "i", you'll get this message.

You may also try to print more complex expressions, like "i*2", or "argv[3]", or "argv[argc]", and so on. In fact, you may also use type casts, call functions found in the program, and whatever your sick mind could imagine (well, almost). Again, this is a good time to try this out.

Examining The Function Call Stack

Once we got into a break-point and examined some variables, we might also wish to see "where we are". That is, what function is being executed now, which function called it, and so on. This can be done using the `"where"` command. At the gdb command prompt, just type "where", and you'll see something like this:

```
#0  print_string (num=1, string=0xbffffc9a "hello") at debug_me.c:7
#1  0x80484e3 in main (argc=1, argv=0xbffffba4) at debug_me.c:23
```

This means the currently executing function is "print_string", at file "debug_me.c", line 7. The function that called it is "main". We also see which arguments each function had received. If there were more functions in the call chain, we'd see them listed in order. This list is also called "a stack trace", since it shows us the structure of the execution stack at this point in the program's life.

Just as we can see contents of variables in the current function, we can see contents of variables local to the calling function, or to any other function on the stack. For example, if we want to see the contents of variable "i" in function "main", we can type the following two commands:

```
frame 1
print i
```

The "frame" command tells the debugger to switch to the given stack frame ('0' is the frame of the currently executing function). At that stage, any print command invoked will use the context of that stack frame. Of-course, if we issue a "step" or "next" command, the program will continue at the top frame, not at the frame we requested to see. After all, the debugger cannot "undo" all the calls and continue from there.

Attaching To an Already Running Process

It might be that we'll want to debug a program that cannot be launched from the command line. This may be because the program is launched from some system daemon (such as a CGI program on the web), and we are too lazy to make it possible to run it directly from the command line. Or perhaps the program takes very long time to run its initialization code, and starting it with a debugger attached to it will cause this startup time to be much much longer. There are also other reasons, but hopefully you got the point. In order to do that, we will launch the debugger in this way:

```
gdb debug_me 9561
```

Here we assume that "debug_me" is the name of the program executed, and that 9561 is the process id (PID) of the process we want to debug.

What happens is that gdb first tries looking for a "core" file named "9561" (we'll see what core files are in the next section), and when it won't find it, it'll assume the supplied number is a process ID, and try to attach to it. If there process executes exactly the same program whose path we gave to gdb (not a copy of the file. it must be the exact same file that the process runs), it'll attach to the program, pause its execution, and will let us continue debugging it as if we started the program from inside the debugger. Doing a "where" right when we get gdb's prompt will show us the stack trace of the process, and we can continue from there. Once we exit the debugger, It will detach itself from the process, and the process will continue execution from where we left it.

Debugging A Crashed Program

One of the problems about debugging programs, has to do with Murphy's law: *A program will crash when least expected*. This phrase just means that after you take the program out as production code, it will crash. And the bugs won't necessarily be easy to reproduce. Luckily, there is some aid for us, in the image of "core files".

A core file contains the memory image of a process, and (assuming the program within the process contains debug info) its stack trace, contents of variables, and so on. A program is normally set to generate a core file containing its memory image when it crashes due to signals such as SEGV or BUS. Provided that the shell invoking the program was not set to limit the size of this core file, we will find this file in the working directory of the process (either the directory from which it was started, or the directory it last switched to using the `chdir` system call).

Once we get such a core file, we can look at it by issuing the following command:

```
gdb /path/to/program/debug_me core
```

This assumes the program was launched using this path, and the core file is in the current directory. If it is not, we can give the path to the core file. When we get the debugger's prompt (assuming the core file was successfully read), we can issue commands such as "print", "where" or "frame X". We can not issue commands that imply execution (such as "next", or the invocation of function calls). In some situations, we will be able to see what caused the crash.

One should note that if the program crashed due to invalid memory address access, this will imply that the memory of the program was corrupt, and thus that the core file is corrupt as well, and thus contains bad memory contents, invalid stack frames, etc. Thus, we should see the core file's contents as one possible past, out of many probable pasts (this makes core file analysis rather similar to quantum theory. almost).

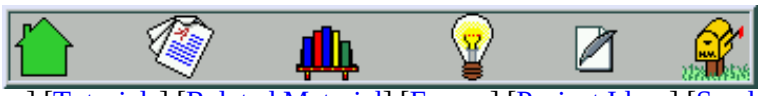
Getting More Info About Debugging

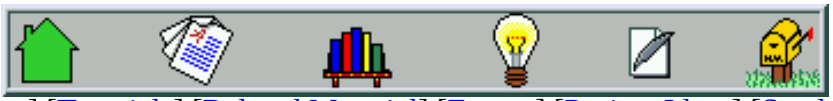
It is now probably time to go play around with your programs and your debugger. It is suggested to try "help" inside gdb, to learn more about its commands. Especially the "dir" command, that enables debugging programs whose source code is split over several directories.

Once you feel that gdb is too limiting, you can try out any of various graphical debuggers. Try to check if you have "xxgdb" installed - this is a graphical interface running on top of gdb. If you find it too ugly, you can try out "ddd". Its main advantage

over xxgdb is that it allows you to graphically view the contents of pointers, linked lists and other complex data structures. It might not be installed on your system, and thus you'll need to download it from the network.

If you're running on a SunOs or Solaris environment, there is a program named "gcore", that allows taking the core of a running process, without stopping it. This is useful if the process is running in an infinite loop, and you want to take a core file to keep aside, or you want to debug a running process without interrupting it for too long.





v1.0.1

Automating Program Compilation - Writing Makefiles

Table Of Contents:

Preface

Compiling a program made of one source file is easy. Compiling one made of few sources is slightly annoying, but may be automated via a simple shell script. Anything larger then that would start to get on your nerves. This is where makefiles are helpful.

A makefile is a collection of instructions that should be used to compile your program. Once you modify some source files, and type the command "make" (or "gmake" if using GNU's make), your program will be recompiled using as few compilation commands as possible. Only the files you modified and those dependent upon them will be recompiled. Of-course, this is not done via usage of magic. You need to supply the rules for compiling various files and file types, and the list of dependencies between files (if file "A" was changed, then files "B", "C" and "D" also need to be re-compiled), but that only has to be done once.

The Structure Of A Makefile

A typical makefile contains lines of the following types:

- Variable Definitions - these lines define values for variables. For example:

```
CFLAGS = -g -Wall
SRCS = main.c file1.c file2.c
CC = gcc
```

- Dependency Rules - these lines define under what conditions a given file (or a type of file) needs to be re-compiled, and how to compile it. For example:

```
main.o: main.c
        gcc -g -Wall -c main.c
```

  This rule means that "main.o" has to be recompiled whenever "main.c" is modified. The rule continues to tell us that in order to compile "main.o", the command `"gcc -g -Wall -c main.c"` needs to be executed.
  Note that each line in the commands list must begin with a TAB character. "make" is quite picky about the makefile's syntax.

- Comments - Any line beginning with a "#" sign, or any line that contains only white-space.

Order Of Compilation

When a makefile is executed, we tell the `make` command to compile a specific target. A target is just some name that appears at the beginning of a rule. It can be a name of a file to create, or just a name that is used as a starting point (such a target is also called a "phony" target).

When `make` Is invoked, it first evaluates all variable assignments, from top to bottom, and when it encounters a rule "A" whose target matches the given target (or the first rule, if no target was supplied), it tries to evaluate this rule. First, it tries to recursively handle the dependencies that appear in the rule. If a given dependency has no matching rule, but there is a file in the disk with this name, the dependency is assumed to be up-to-date. After all the dependencies were checked, and any of them required handling, or refers to a file newer then the target, the command list for rule "A" is executed, one command at a time.

This might appear a little complex, but the example in the next section will make things clear.

Starting Small - A Single-Source Makefile Example

Lets first see a simple example of a makefile that is used to compile a program made of a single source file:

```
# top-level rule to create the program.
all: main

# compiling the source file.
main.o: main.c
        gcc -g -Wall -c main.c

# linking the program.
main: main.o
        gcc -g main.o -o main

# cleaning everything that can be automatically recreated with "make".
clean:
        /bin/rm -f main main.o
```

Few notes about this makefile:

1. Not all rules have to be used in every invocation of `make`. The "clean" rule, for example, is not normally used when building the program, but it may be used to remove the object files created, to save disk space.
2. A rule doesn't need to have any dependencies. This means if we tell make to handle its target, it will always execute its commands list, as in the "clean" rule above.
3. A rule doesn't need to have any commands. For example, the "all" rule is just used to invoke other rules, but does not need any commands of its own. It is just convenient to make sure that if someone runs `make` without a target name, this rule will get executed, due to being the first rule encountered.
4. We used the full path to the "rm" command, instead of just typing "rm", because many users have this command aliased to something else (for example, "rm" aliased to "rm -i"). By using a full path, we avoid any aliases.

Getting Bigger - A Multi-Source Makefile Example

In anything but the simplest programs, we usually have more then one source file. This is where using a makefile starts to pay off. Making a change to one file requires re-compilation of only the modified file, and then re-linking the program. Here is an example of such a makefile:

```
# top-level rule to compile the whole program.
all: prog

# program is made of several source files.
prog: main.o file1.o file2.o
        gcc main.o file1.o file2.o -o prog

# rule for file "main.o".
main.o: main.c file1.h file2.h
        gcc -g -Wall -c main.c

# rule for file "file1.o".
file1.o: file1.c file1.h
        gcc -g -Wall -c file1.c

# rule for file "file2.o".
file2.o: file2.c file2.h
        gcc -g -Wall -c file2.c

# rule for cleaning files generated during compilations.
clean:
        /bin/rm -f prog main.o file1.o file2.o
```

Few notes:

- There is one rule here for each source file. This causes some redundancy, but later on we'll see how to get rid of it.
- We add dependency on included files (file1.h, file2.h) where applicable. If one of these interface-definition files changes, the files that include it might need a re-compilation too. This is not always true, but it is better to make a redundant compilation, then to have object files that are not synchronized with the source code.

Using Compiler And Linker Flags

As one could see, there are many repetitive patterns in the rules for our makefile. For example, what if we wanted to change the flags for compilation, to use optimization (`-O`), instead of add debug info (`-g`)? we would need to go and change this flag for each rule. This might not look like much work with 3 source files, but it will be tedious when we have few tens of files, possibly spread over few directories.

The solution to this problem is using variables to store various flags, and even command names. This is especially useful when trying to compile the same source code with different compilers, or on different platforms, where even a basic command such as "rm" might reside in a different directory on each platform.

Lets see the same makefile, but this time with the introduction of variables:

```
# use "gcc" to compile source files.
CC = gcc
# the linker is also "gcc". It might be something else with other compilers.
LD = gcc
# Compiler flags go here.
CFLAGS = -g -Wall
# Linker flags go here. Currently there aren't any, but if we'll switch to
# code optimization, we might add "-s" here to strip debug info and symbols.
LDFLAGS =
# use this command to erase files.
RM = /bin/rm -f
# list of generated object files.
OBJS = main.o file1.o file2.o
# program executable file name.
PROG = prog

# top-level rule, to compile everything.
all: $(PROG)

# rule to link the program
$(PROG): $(OBJS)
        $(LD) $(LDFLAGS) $(OBJS) -o $(PROG)

# rule for file "main.o".
main.o: main.c file1.h file2.h
        $(CC) $(CFLAGS) -c main.c

# rule for file "file1.o".
file1.o: file1.c file1.h
        $(CC) $(CFLAGS) -c file1.c

# rule for file "file2.o".
file2.o: file2.c file2.h
        $(CC) $(CFLAGS) -c file2.c

# rule for cleaning re-compilable files.
clean:
        $(RM) $(PROG) $(OBJS)
```

Few notes:

- We define many variables in this makefile. This will make it very easy to modify compile flags, compiler used, etc. It is good practice to define even things that might seem like they'll never change. In time - they will.
- We still have a problem with the fact that we define one rule for each source file. If we'll want to change this rule's format, it will be rather tedious. The next section will show us how to avoid this problem.

A Rule For Everyone - Using "File Type" Rules

So, the next phase would be to eliminate the redundant rules, and try to use one rule for all source files. After all, they are all compiled in the same way. Here is a modified makefile:

```
# we'll skip all the variable definitions, just take them from the previous
# makefile
.
.
# linking rule remains the same as before.
$(PROG): $(OBJS)
        $(LD) $(LDFLAGS) $(OBJS) -o $(PROG)

# now comes a meta-rule for compiling any "C" source file.
%.o: %.c
        $(CC) $(CFLAGS) -c $<
```

We should explain two things about our meta-rule:

1. The "%" character is a wildcard, that matches any part of a file's name. If we mention "%" several times in a rule, they all must match the same value, for a given rule invocation. Thus, our rule here means "A file with a '.o' suffix is dependent on a file with the same name, but a '.c' suffix".
2. The "$<" string refers to the dependency list that was matched by the rule (in our case - the full name of the source file). There are other similar strings, such as "$@" which refers to the full target name, or "$*", that refers the part that was matched by the "%" character.

Automatic Creation Of Dependencies

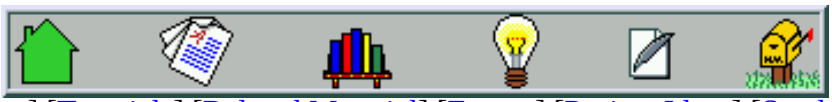
One problem with the usage of implicit rules, is that we lost the full list of dependencies, that are unique to each file. This can be overcome by using extra rules for each file, that only contain dependencies, but no commands. This can be added manually, or be automated in one of various ways. Here is one example, using the "makedepend" Unix program.

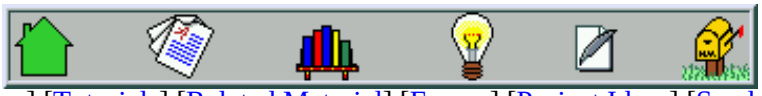
```
# define the list of source files.
SRCS = main.c file1.c file2.c
.
.
# most of the makefile remains as it was before.
# at the bottom, we add these lines:

# rule for building dependency lists, and writing them to a file
# named ".depend".
depend:
```

```
        $(RM) .depend
        makedepend -f- -- $(CFLAGS) -- $(SRCS) > .depend

# now add a line to include the dependency list.
include .depend
```

Now, if we run "make depend", the "makedepend" program will scan the given source files, create a dependency list for each of them, and write appropriate rules to the file ".depend". Since this file is then included by the makefile, when we'll compile the program itself, these dependencies will be checked during program compilation.

There are many other ways to generate dependency lists. It is advised that programmers interested in this issue read about the compiler's "-M" flag, and read the manual page of "makedepend" carefully. Also note that gnu make's info pages suggest a different way of making dependency lists created automatically when compiling a program. The advantage is that the dependency list never gets out of date. The disadvantage is that many times it is being run for no reason, thus slowing down the compilation phase. Only experimenting will show you exactly when to use each approach.





Manipulating Files And Directories In Unix

Who Is This For?

The following tutorial describes various common methods for reading and writing files and directories on a Unix system. Part of the information is common C knowledge, and is repeated here for completeness. Other information is Unix-specific, although DOS programmers will find some of it similar to what they saw in various DOS compilers. If you are a proficient C programmer, and know everything about the standard I/O functions, its buffering operations, and know functions such as `fseek()` or `fread()`, you may skip the standard C library I/O functions section. If in doubt, at least skim through this

section, to catch up on things you might not be familiar with, and at least look at the standard C library examples.

General Unix File System Structure

In the Unix system, all files and directories reside under a single top directory, called root directory, and denoted as "/". Even if the computer has several hard disks attached, they are all combined in a single directories tree. It is up to the system administrator to place all disks on this tree. Each disk is being connected to some directory in the file system. This connection operation is called "mount", and is usually done automatically when the system starts running.

Each directory may contain files, as well as other directories. In addition, each directory also contains two special entries, the entries "." and ".." (i.e. "dot" and "dot dot", respectively). The "." entry refers to the same directory it is placed in. The ".." entry refers to the directory containing it. The sole exception is the root directory, in which the ".." entry still refers to the root directory (after all, the root directory is not contained in any other directory).

A directory is actually a file that has a special attribute (denoting it as being a directory), that contains a list of file names, and "pointers" to these files on the disk.

Besides normal files and directories, a Unix file system may contain various types of special files:

- Symbolic link. This is a file that points to another file (or directory) in the file system. Opening such a file generally opens the file it points to instead (unless special system calls are used).
- Character (or block) special file. This file represents a physical device (and is usually placed in the "/dev" directory). Opening this file allows accessing the given device directly. Each device (disks, printers, serial ports etc) has a file in the "/dev" directory.
- Other special files (pipes and sockets) used for inter-process communications.

Standard "C" File Read And Write

The basic method of reading files and writing into files is by using the standard C library's input and output functions. This works portably across all operating systems, and also gives us some efficiency enhancements - the standard library buffers read and write operations, making file operations faster then if done directly by using system calls to read and write files.

The `FILE` Structure

The `FILE` structure is the basic data type used when handling files with the standard C library. When we open a file, we get a pointer to such a structure, that we later use with all other operations on the file, until we close it. This structure contains information such as the location in the file from which we will read next (or to which we will write next), the read buffer, the write buffer, and so on. Sometimes this structure is also referred to as a "file stream", or just "stream".

Opening And Closing A File

In order to work with a file, we must open it first, using the `fopen()` function. We specify the path to the file (full path, or relative to the current working directory), as well as the mode for opening the file (open for reading, for writing, for reading and writing, for appending only, etc.). Here are a few examples of how to use it:

```
/* FILE structure pointers, for the return value of fopen() */
FILE* f_read;
FILE* f_write;
FILE* f_readwrite;
FILE* f_append;

/* Open the file /home/choo/data.txt for reading */
f_read = fopen("/home/choo/data.txt", "r");
if (!f_read) { /* open operation failed. */
```

```
    perror("Failed opening file '/home/choo/data.txt' for reading:");
    exit(1);
}

/* Open the file logfile in the current directory for writing. */
/* if the file does not exist, it is being created.           */
/* if the file already exists, its contents is erased.        */
f_write = fopen("logfile", "w");

/* Open the file /usr/local/lib/db/users for both reading and writing    */
/* Any data written to the file is written at the beginning of the file, */
/* over-writing the existing data.                                       */
f_readwrite = fopen("/usr/local/lib/db/users", "r+");

/* Open the file /var/adm/messages for appending.       */
/* Any data written to the file is appended to its end. */
f_append = fopen("/var/adm/messages", "a");
```

As you can see, the mode of opening the file is given as an abbreviation. More options are documented in the manual page for the `fopen()` function. The `fopen()` function returns a pointer to a `FILE` structure on success, or a NULL pointer in case of failure. The exact reason for the failure may be anything from "file does not exist" (in read mode), "permission denied" (if we don't have permission to access the file or its directory), I/O error (in case of a disk failure), etc. In such a case, the global variable "errno" is being set to the proper error code, and the `perror()` function may be used to print out a text string related to the exact error code.

Once we are done working with the file, we need to close it. This has two effects:

1. Flushing any un-saved changes to disk (actually, to the operating system's disk cache).
2. Freeing the file descriptor (will be explained in the system calls section below) and any other resources associated with the open file.

Closing the file is done with the `fclose()` function, as follows:

```
if (!fclose(f_readwrite)) {
    perror("Failed closing file '/usr/local/lib/db/users':");
    exit(1);
}
```

`fclose()` returns 0 on success, or `EOF` (usually '-1') on failure. It will then set "errno" to zero. One may wonder how could closing a file fail - this may happen if any buffered writes were not saved to disk, and are being saved during the close operation. Whether the function succeeded or not, the `FILE` structure may not be used any more by the program.

Reading From An Open File

Once we have a pointer for an open file's structure, we may read from it using any of several functions. In the following code, assume f_read and f_readwrite pointers to `FILE` structures returned by previous calls to `fopen()`.

```
/* variables used by the various read operations.           */
int c;
char buf[201];

/* read a single character from the file.                   */
/* variable c will contain its ASCII code, or the value EOF, */
/* if we encountered the end of the file's data.            */
c = fgetc(f_read);

/* read one line from the file. A line is all characters up to a new-line  */
/* character, or up to the end of the file. At most 200 characters will be */
```

```
/* read in (i.e. one less then the number we supply to the function call). */
/* The string read in will be terminated by a null character, so that is   */
/* why the buffer was made 201 characters long, not 200. If a new line     */
/* character is read in, it is placed in the buffer, not removed.          */
/* note that 'stdin' is a FILE structure pre-allocated by the */
/* C library, and refers to the standard input of the process (normally    */
/* input from the keyboard).                                               */
fgets(buf, 201, stdin);

/* place the given character back into the given file stream. The next     */
/* operation on this file will return this character. Mostly used by       */
/* parsers that analyze a given text, and try to guess what the next       */
/* is. If they miss their guess, it is easier to push the last character   */
/* back to the file stream, then to make book-keeping operations.          */
ungetc(c, stdin);

/* check if the read/write head has reached past the end of the file.      */
if (feof(f_read)) {
    printf("End of file reached\n");
}

/* read one block of 120 characters from the file stream, into 'buf'.   */
/* (the third parameter to fread() is the number of blocks to read).    */
char buf[120];
if (fread(buf, 120, 1, f_read) != 1) {
    perror("fread");
}
```

There are various other file reading functions (`getc()` for example), but you'll be able to learn them from the on-line manual.

Note that when we read in some text, the C library actually reads it from disk in full blocks (with a size of 512 characters, or something else, as optimal for the operating system we work with). For example, if we read 20 consecutive characters using `fgetc()` 20 times, only one disk operation is made. The rest of the read operations are made from the buffer kept in the `FILE` structure.

Writing Into An Open File

Just like the read operations, we have write operations as well. They are performed at the current location of the read/write pointer kept in the `FILE` structure, and are also done in a buffered mode - only if we fill in a full block, the C library's write functions actually write the data to disk. Yet, we can force it to write data at a given time (e.g. if we print to the screen and want partially written lines to appear immediately). In the following example, assume that f_readwrite is a pointer to a `FILE` structure returned from a previous call to `fopen()`.

```
/* variables used by the various write operations.   */
int c;
char buf[201];

/* write the character 'a' to the given file.        */
c = 'a';
fputc(c, f_readwrite);

/* write the string "hello world" to the given file. */
strcpy(buf, "hello world");
fputs(buf, f_readwrite);

/* write the string "hi there, mate" to the standard input (screen) */
```

```
/* a new-line in placed in the string, to make the cursor move      */
/* to the next line on screen after writing the string.             */
fprintf(stdout, "hi there, mate\n");

/* write out any buffered writes to the given file stream.          */
fflush(stdout);

/* write twice the string "hello, great world. we feel fine!\n" to 'f_readwrite'. */
/* (the third parameter to fwrite() is the number of blocks to write).            */
char buf[100];
strcpy(buf, "hello, great world. we feel fine!\n");
if (fwrite(buf, strlen(buf), 2, f_readwrite) != 2) {
    perror("fwrite");
}
```

Note that when the output is to the screen, the buffering is done in line mode, i.e. whenever we write a new-line character, the output is being flushed automatically. This is not the case when our output is to a file, or when the standard output is being redirected to a file. In such cases the buffering is done for larger chunks of data, and is said to be in "block-buffered mode".

Moving The Read/Write Location In An Open File

Until now we have seen how input and output is done in a serial mode. However, in various occasions we want to be able to move inside the file, and write to different locations, or read from different locations, without having to scan the whole code. This is common in database files, when we have some index telling us the location of each record of data in the file. Traveling in a file stream in such a manner is also called "random access".

The `fseek()` function allows us to move the read/write pointer of a file stream to a desired location, stated as the number of bytes from the beginning of the file (or from the end of file, or from the current position of the read/write pointer). The `ftell()` function tells us the current location of the read/write header of the given file stream. Here is how to use these functions:

```
/* move the read/write pointer of the file stream to position '30' */
/* in the file. Note that the first position in the file is '0',   */
/* not '1'.                                                        */
fseek(f_read, 29L, SEEK_START);

/* move the read/write pointer of the file stream 25 characters    */
/* forward from its given location.                                */
fseek(f_read, 25L, SEEK_SET);

/* remember the current read/write pointer's position, move it     */
/* to location '520' in the file, write the string "hello world",  */
/* and move the pointer back to the previous location.             */
long old_position = ftell(f_readwrite);
if (old_position < 0) {
    perror("ftell");
    exit(0);
}
if (fseek(f_readwrite, 520L, SEEK_SET) < 0) {
    perror("fseek(f_readwrite, 520L, SEEK_SET)");
    exit(0);
}
fputs("hello world", f_readwrite);
if (fseek(f_readwrite, old_position, SEEK_SET) < 0) {
    perror("fseek(f_readwrite, old_position, SEEK_SET)");
    exit(0);
}
```

Note that if we move inside the file with `fseek()`, any character put to the stream using `ungetc()` is lost and forgotten.

Note: it is ok to seek past the end of a file. If we will try to read from there, we will get an error, but if we try to write there, the file's size will be automatically enlarged to contain the new data we wrote. All characters between the previous end of file and the newly written data will contain nulls ('\0') when read. Note that the size of the file has grown, but the file itself does not occupy so much space on disk - the system knows to leave "holes" in the file. However, if we try to copy the file to a new location using the Unix "cp" command, the new file will have all wholes filled in, and will occupy much more disk space then the original file.

A Complete Example

Two examples are given for the usage of the standard C library I/O functions. The first example is a file copying program, that reads a given file one line at a time, and writes these lines to a second file. The source code is found in the file stdc-file-copy.c. Note that this program does not check if a file with the name of the target already exists, and thus viciously erases any existing file. Be careful when running it! Later, when discussing the system calls interface, we will see how to avoid this danger.

The second example manages a small database file with fixed-length records (i.e. all records have the same size), using the `fseek()` function. The source is found in the file stdc-small-db.c. Functions are supplied for reading a record and for writing a record, based on an index number. See the source code for more info. This program uses the `fread()` and `fwrite()` functions to read data from the file, or write data to the file. Check the on-line manual page for these functions to see exactly what they do.

Accessing Files With System Calls

Usually, reading and writing files is done best using the standard C library functions. However, in various occasions we need a more low-level to the files. For example, we cannot check file permissions or file size using the standard C library. Also, you will see that Unix treats various devices in a similar manner to using files, and using the same functions you can read from a file, from a network connection and so on. Thus, it is useful to learn this generic interface.

The Little File Descriptor That Could

The basic system object used to manipulate files is called a file descriptor. This is an integer number that is used by the various I/O system calls to access a memory area containing data about the open file. This memory area has a similar role to the `FILE` structure in the standard C library I/O functions, and thus the pointer returned from `fopen()` has a role similar to a file descriptor.

Each process has its own file descriptors table, with each entry pointing to a an entry in a system file descriptor table. This allows several processes to share file descriptors, by having a table entry pointing to the same entry in the system file descriptors table. You will encounter this phenomena, and how it can be used, when learning about multi-process programming.

The value of the file descriptor is a non-negative integer. Usually, three file descriptors are automatically opened by the shell that started the process. File descriptor '0' is used for the standard input of the process. File descriptor '1' is used for the standard output of the process, and file descriptor '2' is used for the standard error of the process. Normally the standard input gets input from the keyboard, while standard output and standard error write data to the terminal from which the process was started.

Opening And Closing File Descriptors

Opening files using the system call interface is done using the `open()` system call. Similar to `fopen()`, it accepts two parameters. One containing the path to the file to open, the other contains the mode in which to open the file. The mode may be any of the following:

`O_RDONLY`

Open the file in read-only mode.

`O_WRONLY`

Open the file in write-only mode.

`O_RDWR`

Open the file for both reading and writing.

In addition, any of the following flags may be OR-ed with the mode flag:

`O_CREAT`

If the file does not exist already - create it.

`O_EXCL`

If used together with `O_CREAT`, the call will fail if the file already exists.

`O_TRUNC`

If the file already exists, truncate it (i.e. erase its contents).

`O_APPEND`

Open the file in append mode. Any data written to the file is appended at the end of the file.

`O_NONBLOCK` (or `O_NDELAY`)

If any operation on the file is supposed to cause the calling process block, the system call instead will fail, and errno be set to `EAGAIN`. This requires caution on the part of the programmer, to handle these situations properly.

`O_SYNC`

Open the file in synchronous mode. Any write operation to the file will block until the data is written to disk. This is useful in critical files (such as database files) that must always remain in a consistent state, even if the system crashes in the middle of a file operation.

Unlike the `fopen()` function, `open()` accepts one more (optional) parameter, which defines the access permissions that will be given to the file, in case of file creation. This parameter is a combination of any of the following flags:

S_IRWXU

Owner of the file has read, write and execute permissions to the file.

S_IRUSR

Owner of the file has read permission to the file.

S_IWUSR

Owner of the file has write permission to the file.

S_IXUSR

Owner of the file has execute permission to the file.

S_IRWXG

Group of the file has read,write and execute permissions to the file.

S_IRGRP

Group of the file has read permission to the file.

S_IWGRP

Group of the file has write permission to the file.

S_IXGRP

Group of the file has execute permission to the file.

S_IRWXO

Other users have read,write and execute permissions to the file.

S_IROTH

Other users have read permission to the file.

S_IWOTH

Other users have write permission to the file.

S_IXOTH

Other users have execute permission to the file.

Here are a few examples of using `open()`:

```
/* these hold file descriptors returned from open(). */
int fd_read;
int fd_write;
int fd_readwrite;
int fd_append;

/* Open the file /etc/passwd in read-only mode. */
fd_read = open("/etc/passwd", O_RDONLY);
if (fd_read < 0) {
    perror("open");
    exit(1);
}

/* Open the file run.log (in the current directory) in write-only mode. */
/* and truncate it, if it has any contents.                              */
fd_write = open("run.log", O_WRONLY | O_TRUNC);
if (fd_write < 0) {
    perror("open");
    exit(1);
}

/* Open the file /var/data/food.db in read-write mode. */
fd_readwrite = open("/var/data/food.db", O_RDWR);
if (fd_readwrite < 0) {
    perror("open");
    exit(1);
}

/* Open the file /var/log/messages in append mode. */
fd_append = open("/var/log/messages", O_WRONLY | O_APPEND);
if (fd_append < 0) {
    perror("open");
    exit(1);
}
```

Once we are done working with a file, we need to close it, using the `close()` system call, as follows:

```
if (close(fd) == -1) {
    perror("close");
    exit(1);
}
```

This will cause the file to be closed. Note that no buffering is normally associated with files opened with `open()`, so no buffer flushing is required.

Note: If a file that is currently open by a Unix process is being erased (using the Unix "rm" command, for example), the file is not really removed from the disk. Only when the process (or all processes) holding the file open, the file is physically removed from the disk. Until then it is just removed from its directory, not from the disk.

Reading From A File Descriptor

Once we got a file descriptor to an open file (that was opened in read mode), we may read data from the file using the `read()` system call. This call takes three parameters: the file descriptor to read from, a buffer to read data into, and the number of characters to read into the buffer. The buffer must be large enough to contain the data. Here is how to use this call. We assume 'fd' contains a file descriptor returned from a previous call to `open()`.

```
/* return value from the read() call. */
size_t rc;
/* buffer to read data into.          */
char buf[20];

/* read 20 bytes from the file.       */
rc = read(fd, buf, 20);
if (rc == 0) {
    printf("End of file encountered\n");
}
else if (rc < 0) {
    perror("read");
    exit(1);
}
else {
    printf("read in '%d' bytes\n", rc);
}
```

As you can see, `read()` does not always read the number of bytes we asked it to read. This could be due to a signal interrupting it in the middle, or the end of the file was encountered. In such a case, `read()` returns the number of bytes it actually read.

Writing Into A File Descriptor

Just like we used `read()` to read from the file, we use the `write()` system call, to write data to the file. The write operations is done in the location of the current read/write pointer of the given file, much like the various standard C library output functions did. `write()` gets the same parameters as `read()` does, and just like `read()`, might write only part of the data to the given file, if interrupted in the middle, or for other reasons. In such a case it will return the number of bytes actually written to the file. Here is a usage example:

```
/* return value from the write() call. */
size_t rc;

/* write the given string to the file. */
rc = write(fd, "hello world\n", strlen("hello world\n"));
if (rc < 0) {
    perror("write");
    exit(1);
}
else {
    printf("wrote in '%d' bytes\n", rc);
}
```

As you can see, there is never an end-of-file case with a write operation. If we write past the current end of the file, the file will be enlarged to contain the new data.

Sometimes, writing out the data is not enough. We want to be sure the file on the physical disk gets updated immediately (note that even thought the system calls do not buffer writes, the operating system still buffers write operations using its disk cache). In such cases, we may use the `fsync()` system call. It ensures that any write operations for the given file descriptor that are kept in the system's disk cache, are actually written to disk, when the `fsync()` system call returns to the caller. Here is how to use it:

```
#include <unistd.h>    /* declaration of fsync() */
.
.
if (fsync(fd) == -1) {
```

```
    perror("fsync");
}
```

Note that `fsync()` updates both the file's contents, and its book-keeping data (such as last modification time). If we only need to assure that the file's contents is written to disk, and don't care about the last update time, we can use `fdatasync()` instead. This is more efficient, as it will issue one fewer disk write operation. In applications that need to synchronize data often, this small saving is important.

Seeking In An Open File

Just like we used the `fseek()` function to move the read/write pointer of the file stream, we can use the `lseek()` system call to move the read/write pointer for a file descriptor. Assuming you understood the `fseek()` examples above, here are a few similar examples using `lseek()`. We assume that 'fd_read' is an integer variable containing a file descriptor to a previously opened file, in read only mode. 'fd_readwrite' is a similar file descriptor, but for a file opened in read/write mode.

```
/* this variable is used for storing locations returned by       */
/* lseek().                                                      */
off_t location;

/* move the read/write pointer of the file to position '40'      */
/* in the file. Note that the first position in the file is '0', */
/* not '1'.                                                      */
location = lseek(fd_read, 39L, SEEK_START);

/* move the read/write pointer of the file stream 67 characters  */
/* forward from its given location.                              */
location = lseek(fd_read, 67L, SEEK_SET);
printf("read/write pointer location: %ld\n", location);

/* remember the current read/write pointer's position, move it   */
/* to location '664' in the file, write the string "hello world",*/
/* and move the pointer back to the previous location.           */
location = lseek(fd_readwrite, 0L, SEEK_SET);
if (location == -1) {
    perror("lseek");
    exit(0);
}
if (lseek(fd_readwrite, 663L, SEEK_SET) == -1) {
    perror("lseek(fd_readwrite, 663L, SEEK_SET)");
    exit(0);
}
rc = write(fd_readwrite, "hello world\n", strlen("hello world\n"));
if (lseek(fd_readwrite, location, SEEK_SET) == -1) {
    perror("lseek(fd_readwrite, location, SEEK_SET)");
    exit(0);
}
```

Note that `lseek()` might not always work for a file descriptor (e.g. if this file descriptor represents the standard input, surely we cannot have random-access to it). You will encounter other similar cases when you deal with network programming and inter-process communications, in the future.

Checking And Setting A File's permission modes

Since Unix supports access permissions for files, we would sometimes need to check these permissions, and perhaps also manipulate them. Two system calls are used in this context, `access()` and `chmod()`.

The `access()` system call is for checking access permissions to a file. This system call accepts a path to a file (full or relative), and a mode mask (made of one or more permission modes). It returns '0' if the specified permission modes are granted for the calling process, or '-1' if any of these modes are not granted, the file does not exist, etc. The access is granted or denied based on the permission flags of the file, and the ID of the user running the process. Here are a few examples:

```
/* check if we have read permission to "/home/choo/my_names".       */
if (access("/home/choo/my_names", R_OK) == 0)
    printf("Read access to file '/home/choo/my_names' granted.\n");
else
    printf("Read access to file '/home/choo/my_names' denied.\n");

/* check if we have both read and write permission to "data.db".     */
if (access("data.db", R_OK | W_OK) == 0)
    printf("Read/Write access to file 'data.db' granted.\n");
else
    printf("Either read or write access to file 'data.db' is denied.\n");

/* check if we may execute the program file "runme".                 */
if (access("runme", X_OK) == 0)
    printf("Execute permission to program 'runme' granted.\n");
else
    printf("Execute permission to program 'runme' denied.\n");

/* check if we may write new files to directory "/etc/config".       */
if (access("/etc/config", W_OK) == 0)
    printf("File creation permission to directory '/etc/sysconfig' granted.\n");
else
    printf("File creation permission to directory '/etc/sysconfig' denied.\n");

/* check if we may read the contents of directory "/etc/config".     */
if (access("/etc/config", R_OK) == 0)
    printf("File listing read permission to directory '/etc/sysconfig' granted.\n");
else
    printf("File listing read permission to directory '/etc/sysconfig' denied.\n");

/* check if the file "hello.world" in the current directory exists. */
if (access("hello world", F_OK) == 0)
    printf("file 'hello world' exists.\n");
else
    printf("file 'hello world' does not exist.\n");
```

As you can see, we can check for read, write and execute permissions, as well as for the existence of a file, and the same for a directory. As an example, we will see a program that checks out if we have read permission to a file, and notifies us if not - where the problem lies. The full source code for this program is found in file read-access-check.c.

Note that we cannot use `access()` to check *why* we got permissions (i.e. if it was due to the given mode granted to us as the owner of the file, or due to its group permissions or its word permissions). For more fine-grained permission tests, see the `stat()` system call mentioned below.

The `chmod()` system call is used for changing the access permissions for a file (or a directory). This call accepts two parameters: a path to a file, and a mode to set. The mode can be a combination of read, write and execute permissions for the user, group or others. It may also contain few special flags, such as the set-user-ID flag or the 'sticky' flag. These permissions will completely override the current permissions of the file. See the `stat()` system call below to see how to make modifications instead of complete replacement. Here are a few examples of using `chmod()`.

```
/* give the owner read and write permission to the file "blabla", */
/* and deny access to any other user.                             */
if (chmod("blabla", S_IRUSR | S_IWUSR) == -1) {
```

```
    perror("chmod");
}

/* give the owner read and write permission to the file "blabla", */
/* and read-only permission to anyone else.                       */
if (chmod("blabla", S_IRUSR | S_IWUSR | S_IRGRP | S_IWOTH) == -1) {
    perror("chmod");
}
```

For the full list of access permission flags to use with `chmod()`, please refer to its manual page.

Checking A File's Status

We have seen how to manipulate the file's data (write) and its permission flags (chmod). We saw a primitive way of checking if we may access it (access), but we often need more then that: what are the exact set of permission flags of the file? when was it last changed? which user and group owns the file? how large is the file?
All these questions (and more) are answered by the `stat()` system call.

`stat()` takes as arguments the full path to the file, and a pointer to a (how surprising) 'stat' structure. When `stat()` returns, it populates this structure with a lot of interesting (and boring) stuff about the file. Here are few of the fields found in this structure (for the rest, read the manual page):

umode_t st_mode
: Access permission flags of the file, as well as information about the type of file (file? directory? symbolic link? etc).

uid_t st_uid
: The ID of the user that owns the file.

gid_t st_gid
: The ID of the group that owns the file.

off_t st_size
: The size of the file (in bytes).

time_t st_atime
: Time when the file was last accessed (read from or written to). Time is given as number of seconds since 1 Jan, 1970.

time_t st_mtime
: Time when the file was last modified (created or written to).

time_t st_ctime
: Time when the file was last changed (had its permission modes changed, or any of its book-keeping, but NOT a contents change).

Here are a few examples of how `stat()` can be used:

```
/* structure passed to the stat() system call, to get its results. */
struct stat file_status;

/* check the status information of file "foo.txt", and print its */
/* type on screen.                                               */
if (stat("foo.txt", &file_status) == 0) {
    if (S_ISDIR(file_status.st_mode))
        printf("foo.txt is a directory\n");
    if (S_ISLNK(file_status.st_mode))
        printf("foo.txt is a symbolic link\n");
    if (S_ISCHR(file_status.st_mode))
        printf("foo.txt is a character special file\n");
    if (S_ISBLK(file_status.st_mode))
        printf("foo.txt is a block special file\n");
    if (S_ISFIFO(file_status.st_mode))
        printf("foo.txt is a FIFO (named pipe)\n");
```

```
    if (S_ISSOCK(file_status.st_mode))
        printf("foo.txt is a (Unix domain) socket file\n");
    if (S_ISREG(file_status.st_mode))
        printf("foo.txt is a normal file\n");
}
else { /* stat() call failed and returned '-1'. */
    perror("stat");
}

/* add the write permission to the group owner of file "/tmp/parlevouz", */
/* without overriding any of the previous access permission flags.       */
if (stat("/tmp/parlevouz", &file_status) == -1) {
    perror("stat");
    exit(1);
}
if (!S_IWGRP(file_status.st_mode)) { /* the group has no write permission */
    umode_t curr_mode = file_status.st_mode & ~S_IFMT
    umode_t new_mode = curr_mode | S_IWGRP;

    if (chmod("/tmp/parlevouz", new_mode) == -1) {
        perror("chmod");
        exit(1);
    }
}
```

The last item should be explained better. For some reason, the 'stat' structure uses the same bit field to contain file type information and access permission flags. Thus, to get only the access permissions, we need to mask off the file type bits. The mask for the file type bits is 'S_IFMT', and thus the mask for the permission modes is its logical negation, or '~S_IFMT'. By logically "and"-ing this value with the 'st_mode' field of the 'stat' structure, we get the current access permission modes. We can add new modes using the logical or ('|') operator, and remove modes using the logical and ('&') operator. After we create the new modes, we use `chmod()` to set the new permission flags for the file.
Note that this operation will also implicitly modify the 'ctime' (change time) of the file, but that won't be reflected in our 'stat' structure, unless we `stat()` the file again.

Renaming A File

The `rename()` system call may be used to change the name (and possibly the directory) of an existing file. It gets two parameters: the path to the old location of the file (including the file name), and a path to the new location of the file (including the new file name). If the new name points to a an already existing file, that file is deleted first. We are allowed to name either a file or a directory. Here are a few examples:

```
/* rename the file 'logme' to 'logme.1' */
if (rename("logme", "logme1") == -1) {
    perror("rename (1):");
    exit(1);
}

/* move the file 'data' from the current directory to directory "/old/info" */
if (rename("data", "/old/info/data") == -1) {
    perror("rename (2):");
    exit(1);
}
```

Note: If the file we are renaming is a symbolic link, then the symbolic link will be renamed, not the file it is pointing to. Also, if the new path points to an existing symbolic link, this symbolic link will be erased, not the file it is pointing to.

Deleting A File

Deleting a file is done using the `unlink()` system call. This one is very simple:

```
/* remove the file "/tmp/data" */
if (unlink("/tmp/data") == -1) {
    perror("unlink");
    exit(1);
}
```

The file will be removed from the directory in which it resides, and all the disk blocks is occupied will be marked as free for re-use by the system. However, if any process currently has this file open, the file won't be actually erased until the last process holding it open erases it. This could explain why often erasing a log file from the system does not increase the amount of free disk space - it might be that the system logger process (syslogd) holds this file open, and thus the system won't really erase it until syslogd closes it. Until then, it will be removed from the directory (i.e. 'ls' won't show it), but not from the disk.

Creating A Symbolic Link

We have encountered symbolic links earlier. lets see how to create them, with the `symlink()` system call:

```
/* create a symbolic link named "link" in the current directory, */
/* that points to the file "/usr/local/data/datafile".           */
if (symlink("/usr/local/data/datafile", "link") == -1) {
    perror("symlink");
    exit(1);
}

/* create a symbolic link whose full path is "/var/adm/log",     */
/* that points to the file "/usr/adm/log".                       */
if (symlink("/usr/adm/log", "/var/adm/log") == -1) {
    perror("symlink");
    exit(1);
}
```

So the first parameter is the file being pointer to, and the second parameter is the file that will be the symbolic link. Note that the first file does not need to exist at all - we can create a symbolic link that points nowhere. If we later create the file this link points to, accessing the file via the symbolic link will work properly.

The Mysterious Mode Mask

If you created files with `open()` or `fopen()`, and you did not supply the mode for the newly created file, you might wonder how does the system assign access permission flags for the newly created file. You will also note that these "default" flags are different on different computers or different account setups. This mysteriousness is due to the usage of the `umask()` system call, or its equivalent `umask` shell command.

The `umask()` system call sets a mask for the permission flags the system will assign to newly created files. By default, newly created files will have read and write permissions to everyone (i.e. rw-rw-rw- , in the format reported by 'ls -l'). Using `umask()`, we can denote which flags will be turned *off* for newly created files. For example, if we set the mask to 077 (a leading 0 denotes an octal value), newly created files will get access permission flags of 0600 (i.e. rw-------). If we set the mask to 027, newly created files will get flags of 0640 (i.e. rw-r-----). Try translating these values to binary format in order to see what is going on here.

Here is how to mess with the `umask()` system call in a program:

```
/* set the file permissions mask to '077'. save the original mask */
/* in 'old_mask'.                                                 */
```

```
int old_mask = umask(077);

/* newly created files will now be readable only by the creating user. */
FILE* f_write = fopen("my_file", "w");
if (f_write) {
    fprintf(f_write, "My name is pit stanman.\n");
    fprintf(f_write, "My voice is my pass code. Verify me.\n");
    fclose(f_write);
}

/* restore the original umask. */
umask(old_mask);
```

Note: the permissions mask affects also calls to `open()` that specify an exact permissions mask. If we want to create a file whose permission are less restrictive the the current mask, we need to use `umaks()` to lighten these restrictions, before calling `open()` to create the file.

Note 2: on most systems you will find that the mask is different then the default. This is because the system administrator has set the default mask in the system-wide shell startup files, using the shell's `umask` command. You may set a different default mask for your own account by placing a proper `umask` command in your shell's starup file ("~/.profile" if you're using "sh" or "bash". "~/.cshrc" if you are using "csh" or "tcsh").

A Complete Example

As an example to the usage of the system calls interface for manipulating files, we will show a program that handles simple log file rotation. The program gets one argument - the name of a log file, and assumes it resides in a given directory ("/tmp/var/log"). If the size of the log file is more then 1024KB, it renames it to have a ".old" suffix, and creates a new (empty) log file with the same name as the original file, and the same access permissions. This code demonstrates combining many system calls together to achieve a task. The source code for this program is found in the file rename-log.c.

• Reading The Contents Of Directories

After we have learned how to write the contents of a file, we might wish to know how to read the contents of a directory. We could open the directory and read its contents directly, but this is not portable. Instead, we have a standard interface for opening a directory and scanning its contents, entry by entry.

The `DIR` And `dirent` Structures

When we want to read the contents of a directory, we have a function that opens a directory, and returns a `DIR` structure. This structure contains information used by other calls to read the contents of the directory, and thus this structure is for directory reading, what the `FILE` structure is for files reading.

When we use the `DIR` structure to read the contents of a directory, entry by entry, the data regarding a given entry is returned in a `dirent` structure. The only relevant field in this structure is `d_name`, which is a null-terminated character array, containing the name of the entry (be it a file or a directory). note - the name, NOT the path.

Opening And Closing A Directory

In order to read the contents of a directory, we first open it, using the `opendir()` function. We supply the path to the directory, and get a pointer to a `DIR` structure in return (or `NULL` on failure). Here is how:

```
#include <dirent.h>    /* struct DIR, struct dirent, opendir().. */

/* open the directory "/home/users" for reading. */
```

```
DIR* dir = opendir("/home/users");
if (!dir) {
    perror("opendir");
    exit(1);
}
```

When we are done reading from a directory, we can close it using the `closedir()` function:

```
if (closedir(dir) == -1) {
    perror("closedir");
    exit(1);
}
```

`closedir()` will return '0' on success, or '-1' if it failed. Unless we have done something really silly, failures shouldn't happen, as we never write to a directory using the `DIR` structure.

Reading The Contents Of A Directory

After we opened the directory, we can start scanning it, entry by entry, using the `readdir()` function. The first call returns the first entry of the directory. Each successive call returns the next entry in the directory. When all entries have been read, `NULL` is returned. Here is how it is used:

```
/* this structure is used for storing the name of each entry in turn. */
struct dirent* entry;

/* read the directory's contents, print out the name of each entry.   */
printf("Directory contents:\n");
while ( (entry = readdir(dir)) != NULL) {
    printf("%s\n", entry.d_name);
}
```

If you try this out, you'll note that the directory always contains the entries "." and "..", as explained in the beginning of this tutorial. A common mistake is to forget checking these entries specifically, in recursive traversals of the file system. If these entries are being traversed blindingly, an endless loop might occur.

Note: if we alter the contents of the directory during its traversal, the traversal might skip directory entries. Thus, if you intend to create a file in the directory, you would better not do that while in the middle of a traversal.

Rewinding A Directory For A Second Scan

After we are done reading the contents of a directory, we can rewind it for a second pass, using the `rewinddir()` function:

```
rewinddir(dir);
```

Checking And Changing The Working Directory

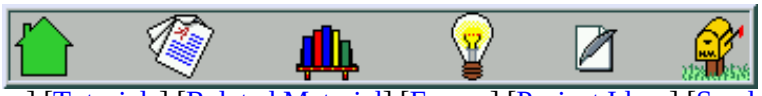
Sometimes we wish to find out the current working directory of a process. The `getcwd()` function is used for that. Other times we wish to change the working directory of our process. This will allow using short paths when accessing several files in the same directory. The `chdir()` system call is used for this. Here is an example:

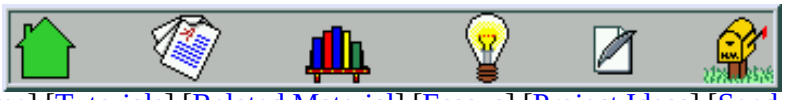
```
/* this buffer is used to store the full path of the current */
/* working directory.                                       */
#define MAX_DIR_PATH 2048;
char cwd[MAX_DIR_PATH+1];
```

```
/* store the current working directory.    */
if (!getcwd(cwd, MAX_DIR_PATH+1)) {
    perror("getcwd");
    exit(1);
}

/* change the current directory to "/tmp". */
if (!chdir("/tmp")) {
    perror("chdir (1)");
    exit(1);
}

/* restore the original working directory. */
if (chdir(cwd) == -1) {
    perror("chdir (2)");
    exit(1);
}
```

A Complete Example

As an example, we will write a limited version of the Unix 'find' command. This command basically accepts a file name and a directory, and finds all files under that directory (or any of its sub-directories) with the given file name. The original program has zillions of command line options, and can also handle file name patterns. Our version will only be able to handle substrings (that is, finding the files whose names contain the given string). The program changes its working directory to the given directory, reads its contents, and recursively scans each sub-directory it encounters. The program does not traverse across symbolic-links to avoid possible loops. The complete source code for the the program is found in the find-file.c file.





v1.0.1

Building And Using Static And Shared "C" Libraries

Table Of Contents:

Building And Using Static And Shared "C" Libraries

One of the problems with developed programs, is that they tend to grow larger and larger, bringing up overall compilation and linking time to a large figure, and polluting out makefile, and the directory where we placed the source files. The first time a program we write reaches this state, is normally when we look for a different way to manage our projects.

It is this point where we start thinking about combining out source code into small units of related files, that can be managed with a separate makefile, possibly by a different programmer (for a multi-programmer project).

What Is A "C" Library? What Is It Good For?

One of the tools that compilers supply us with are libraries. A library is a file containing several object files, that can be used as a single entity in a linking phase of a program. Normally the library is indexed, so it is easy to find symbols (functions, variables and so on) in them. For this reason, linking a program whose object files are ordered in libraries is faster then linking a program whose object files are separate on the disk. Also, when using a library, we have fewer files to look for and open, which even further speeds up linking.

Unix systems (as well as most other modern systems) allow us to create and use two kinds of libraries - static libraries and shared (or dynamic) libraries.

Static libraries are just collections of object files that are linked into the program during the linking phase of compilation, and are not relevant during runtime. This last comment seems obvious, as we already know that object files are also used only during the linking phase, and are not required during runtime - only the program's executable file is needed in order to run the program.

Shared libraries (also called dynamic libraries) are linked into the program in two stages. First, during compile time, the linker verifies that all the symbols (again, functions, variables and the like) required by the program, are either linked into the program, or in one of its shared libraries. However, the object files from the dynamic library are not inserted into the executable file. Instead, when the program is started, a program in the system (called a dynamic loader) checks out which shared libraries were linked with the program, loads them to memory, and attaches them to the copy of the program in memory.

The complex phase of dynamic loading makes launching the program slightly slower, but this is a very insignificant drawback, that is out-weighted by a great advantage - if a second program linked with the same shared library is executed, it can use the same copy of the shared library, thus saving a lot of memory. For example, the standard "C" library is normally a shared library, and is used by all C programs. Yet, only one copy of the library is stored in memory at any given time. This means we can use far less memory to run our programs, and the executable files are much smaller, thus saving a lot of disk space as well.

However, there is one drawback to this arrangement. If we re-compile the dynamic library and try to run a second copy of our program with the new library, we'll soon get stuck - the dynamic loader will find that a copy of the library is already stored in memory, and thus will attach it to our program, and not load the new (modified) version from disk. There are ways around this too, as we'll see in the last section of our discussion.

Creating A Static "C" Library Using "ar" and "ranlib"

The basic tool used to create static libraries is a program called `'ar'`, for 'archiver'. This program can be used to create static libraries (which are actually archive files), modify object files in the static library, list the names of object files in the library, and so on. In order to create a static library, we can use a command like this:

```
ar rc libutil.a util_file.o util_net.o util_math.o
```

This command creates a static library named 'libutil.a' and puts copies of the object files "util_file.o", "util_net.o" and "util_math.o" in it. If the library file already exists, it has the object files added to it, or replaced, if they are newer then those inside the library. The `'c'` flag tells ar to create the library if it doesn't already exist. The `'r'` flag tells it to replace older object files in the library, with the new object files.

After an archive is created, or modified, there is a need to index it. This index is later used by the compiler to speed up symbol-lookup inside the library, and to make sure that the order of the symbols in the library won't matter during compilation (this will be better understood when we take a deeper look at the link process at the end of this tutorial). The command used to create or update the index is called `'ranlib'`, and is invoked as follows:

```
ranlib libutil.a
```

On some systems, the archiver (which is not always `ar`) already takes care of the index, so ranlib is not needed (for example, when Sun's C compiler creates an archive, it is already indexed). However, because `'ar'` and `'ranlib'` are used by many makefiles for many packages, such platforms tend to supply a ranlib command that does nothing. This helps using the same makefile on both types of platforms.

Note: when an archive file's index generation date (stored inside the archive file) is older then the file's last modification date (stored in the file system), a compiler trying to use this library will complain its index is out of date, and abort. There are two ways to overcome the problem:

1. Use `'ranlib'` to re-generate the index.
2. When copying the archive file to another location, use `'cp -p'`, instead of only `'cp'`. The `'-p'` flag tells `'cp'` to keep all attributes of the file, including its access permissions, owner (if "cp" is invoked by a superuser)

and its last modification date. This will cause the compiler to think the index inside the file is still updated. This method is useful for makefiles that need to copy the library to another directory for some reason.

Using A "C" Library In A Program

After we created our archive, we want to use it in a program. This is done by adding the library's name to the list of object file names given to the linker, using a special flag, normally `'-l'`. Here is an example:

```
cc main.o -L. -lutil -o prog
```

This will create a program using object file "main.o", and any symbols it requires from the "util" static library. Note that we omitted the "lib" prefix and the ".a" suffix when mentioning the library on the link command. The linker attaches these parts back to the name of the library to create a name of a file to look for. Note also the usage of the `'-L'` flag - this flag tells the linker that libraries might be found in the given directory ('.', refering to the current directory), in addition to the standard locations where the compiler looks for system libraries.

For an example of program that uses a static library, try looking at our static library example directory.

Creating A Shared "C" Library Using "ld"

The creation of a shared library is rather similar to the creation of a static library. Compile a list of object files, then insert them all into a shared library file. However, there are two major differences:

1. Compile for "Position Independent Code" (PIC) - When the object files are generated, we have no idea where in memory they will be inserted in a program that will use them. Many different programs may use the same library, and each load it into a different memory in address. Thus, we need that all jump calls ("goto", in assembly speak) and subroutine calls will use relative addresses, and not absolute addresses. Thus, we need to use a compiler flag that will cause this type of code to be generated.
   In most compilers, this is done by specifying `'-fPIC'` or `'-fpic'` on the compilation command.
2. Library File Creation - unlike a static library, a shared library is not an archive file. It has a format that is specific to the architecture for which it is being created. Thus, we need to use the compiler (either the compiler's driver, or its linker) to generate the library, and tell it that it should create a shared library, not a final program file.
   This is done by using the `'-G'` flag with some compilers, or the `'-shared'` flag with other compilers.

Thus, the set of commands we will use to create a shared library, would be something like this:

```
cc -fPIC -c util_file.c
cc -fPIC -c util_net.c
cc -fPIC -c util_math.c
cc -shared libutil.so util_file.o util_net.o util_math.o
```

The first three commands compile the source files with the `PIC` option, so they will be suitable for use in a shared library (they may still be used in a program directly, even thought they were compiled with `PIC`). The last command asks the compiler to generate a shared library

---

Using A Shared "C" Library - Quirks And Solutions

Using a shared library is done in two steps:

1. Compile Time - here we need to tell the linker to scan the shared library while building the executable program, so it will be convinced that no symbols are missing. It will not really take the object files from the shared library and insert them into the program.
2. Run Time - when we run the program, we need to tell the system's dynamic loader (the process in charge of

automatically loading and linking shared libraries into the running process) where to find our shared library.

The compilation part is easy. It is done almost the same as when linking with static libraries:

```
cc main.o -L. -lutil -o prog
```

The linker will look for the file 'libutil.so' (`-lutil`) in the current directory (`-L.`), and link it to the program, but will not place its object files inside the resulting executable file, 'prog'.

The run-time part is a little trickier. Normally, the system's dynamic loader looks for shared libraries in some system specified directories (such as /lib, /usr/lib, /usr/X11/lib and so on). When we build a new shared library that is not part of the system, we can use the `'LD_LIBRARY_PATH'` environment variable to tell the dynamic loader to look in other directories. The way to do that depends on the type of shell we use ('tcsh' and 'csh', versus 'sh', 'bash', 'ksh' and similar shells), as well as on whether or not `'LD_LIBRARY_PATH'` is already defined. To check if you have this variable defined, try:

```
echo $LD_LIBRARY_PATH
```

If you get a message such as `'LD_LIBRARY_PATH: Undefined variable.'`, then it is not defined.

Here is how to define this variable, in all four cases:

1. 'tcsh' or 'csh', `LD_LIBRARY_PATH` is not defined:

   ```
   setenv LD_LIBRARY_PATH /full/path/to/library/directory
   ```

2. 'tcsh' or 'csh', `LD_LIBRARY_PATH` already defined:

   ```
   setenv LD_LIBRARY_PATH /full/path/to/library/directory:${LD_LIBRARY_PATH}
   ```

3. 'sh', 'bash' and similar, `LD_LIBRARY_PATH` is not defined:

   ```
   LD_LIBRARY_PATH=/full/path/to/library/directory
   export LD_LIBRARY_PATH
   ```

4. 'sh', 'bash' and similar, `LD_LIBRARY_PATH` already defined:

   ```
   LD_LIBRARY_PATH=/full/path/to/library/directory:${LD_LIBRARY_PATH}
   export LD_LIBRARY_PATH
   ```

After you've defined `LD_LIBRARY_PATH`, you can check if the system locates the library properly for a given program linked with this library:

```
ldd prog
```

You will get a few lines, each listing a library name on the left, and a full path to the library on the right. If a library is not found in any of the system default directories, or the directories mentioned in `'LD_LIBRARY_PATH'`, you will get a 'library not found' message. In such a case, verify that you properly defined the path to the directory inside `'LD_LIBRARY_PATH'`, and fix it, if necessary. If all goes well, you can run your program now like running any other program, and see it role...

For an example of a program that uses a shared library, try looking at our shared library example directory.

Using A Shared "C" Library Dynamically - Programming Interface

One of the less-commonly used feature of shared libraries is the ability to link them to a process anytime during its life. The linking method we showed earlier makes the shared library automatically loaded by the dynamic loader of the system. Yet, it is possible to make a linking operation at any other time, using the 'dl' library. This library provides us with a means to load a shared library, reference any of its symbols, call any of its functions, and finally detach it from the process when no longer needed.

Here is a scenario where this might be appealing: suppose that we wrote an application that needs to be able to read files created by different word processors. Normally, our program might need to be able to read tens of different file formats, but in a single run, it is likely that only one or two such document formats will be needed. We could write one shared library for each such format, all having the same interface (readfile and writefile for example), and one piece of code that determines the file format. Thus, when our program is asked to open such a file, it will first determine its format, then load the relevant shared library that can read and translate that format, and call its readfile function to read the document. We might have tens of such libraries, but only one of them will be placed in memory at any given time, making our application use less system resources. It will also allow us to ship the application with a small set of supported file formats, and add new file formats without the need to replace the whole application, by simply sending the client an additional set of shared libraries.

Loading A Shared Library Using `dlopen()`

In order to open and load the shared library, one should use the `dlopen()` function. It is used this way:

```
#include <dlfcn.h>      /* defines dlopen(), etc.       */
.
.
void* lib_handle;       /* handle of the opened library */

lib_handle = dlopen("/full/path/to/library", RTLD_LAZY);
if (!lib_handle) {
    fprintf(stderr, "Error during dlopen(): %s\n", dlerror());
    exit(1);
}
```

The `dlopen()` function gets two parameters. One is the full path to the shared library. The other is a flag defining whether all symbols refered to by the library need to be checked immediatly, or only when used. In our case, we may use the lazy approach (`RTLD_LAZY`) of checking only when used. The function returns a pointer to the loaded library, that may later be used to reference symbols in the library. It will return `NULL` in case an error occured. In that case, we may use the `dlerror()` function to print out a human-readable error message, as we did here.

Calling Functions Dynamically Using `dlsym()`

After we have a handle to a loaded shared library, we can find symbols in it, of both functions and variables. We need to define their types properly, and we need to make sure we made no mistakes. The compiler won't be able to check those declarations, so we should be extra carefull when typing them. Here is how to find the address of a function named 'readfile' that gets one string parameter, and returns a pointer to a `'struct local_file'` structure:

```
/* first define a function pointer variable to hold the function's address */
struct local_file* (*readfile)(const char* file_path);
/* then define a pointer to a possible error string */
```

```
const char* error_msg;
/* finally, define a pointer to the returned file */
struct local_file* a_file;

/* now locate the 'readfile' function in the library */
readfile = dlsym(lib_handle, "readfile");

/* check that no error occured */
error_msg = dlerror();
if (error_msg) {
    fprintf(stderr, "Error locating 'readfile' - %s\n", error_msg);
    exit(1);
}

/* finally, call the function, with a given file path */
a_file = (*readfile)("hello.txt");
```

As you can see, errors might occur anywhere along the code, so we should be carefull to make extensive error checking. Surely, you'll also check that 'a_file' is not `NULL`, after you call your function.

Unloading A Shared Library Using `dlclose()`

The final step is to close down the library, to free the memory it occupies. This should only be done if we are not intending to use it soon. If we do - it is better to leave it open, since library loading takes time. To close down the library, we use something like this:

```
dlclose(lib_handle);
```

This will free down all resources taken by the library (in particular, the memory its executable code takes up).

Automatic Startup And Cleanup Functions

Finally, the dynamic loading library gives us the option of defining two special functions in each library, namely `_init` and `_fini`. The `_init` function, if found, is invoked automatically when the library is opened, and before `dlopen()` returns. It may be used to invoke some startup code needed to initialize data structures used by the library, read configuration files, and so on.

The `_fini` function is called when the library is closed using `dlclose()`. It may be used to make cleanup operations required by the library (freeing data structures, closing files, etc.).

For an example of a program that uses the 'dl' interface, try looking at our dynamic-shared library example directory.

Getting a Deeper Understanding - The Complete Linking Story

The Importance Of Linking Order

In order to fully understand the way linking is done, and be able to overcome linking problems, we should bare in mind that the order in which we present the object files and the libraries to the linker, is the order in which the linker links them into the resulting binary file.

The linker checks each file in turn. If it is an object file, it is being placed fully into the executable file. If it is a library, the linker checks to see if any symbols referenced (i.e. used) in the previous object files but not defined (i.e. contained) in them, are in the library. If such a symbol is found, the whole object file from the library that contains the symbol - is being added to the executable file. This process continues until all object files and libraries on the command line were processed.

This process means that if library 'A' uses symbols in library 'B', then library 'A' has to appear on the link command before library 'B'. Otherwise, symbols might be missing - the linker never turns back to libraries it has already processed. If library 'B' also uses symbols found in library 'A' - then the only way to assure successful linking is to mention library 'A' on the link command again after library 'B', like this:

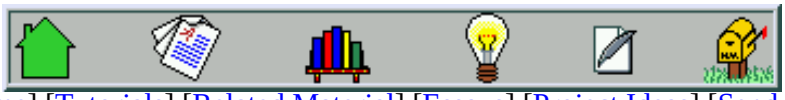
```
$(LD) ..... -lA -lB -lA
```

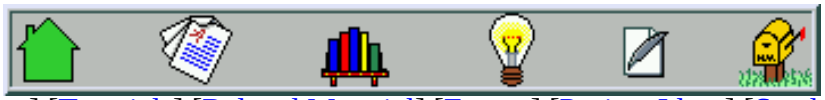
This means that linking will be slower (library 'A' will be processed twice). This also hints that one should try not to have such mutual dependencies between two libraries. If you have such dependencies - then either re-design your libraries' contents, or combine the two libraries into one larger library.

Note that object files found on the command line are always fully included in the executable file, so the order of mentioning them does not really matter. Thus, a good rule is to always mention the libraries after all object files.

Static Linking Vs. Dynamic Linking

When we discussed static libraries we said that the linker will try to look for a file named 'libutil.a'. We lied. Before looking for such a file, it will look for a file named 'libutil.so' - as a shared library. Only if it cannot find a shared library, will it look for 'libutil.a' as a static library. Thus, if we have created two copies of the library, one static and one shared, the shared will be preferred. This can be overridden using some linker flags (`'-Wl,static'` with some linkers, `'-Bstatic'` with other types of linkers. refer to the compiler's or the linker's manual for info about these flags).





Introduction To Unix Signals Programming

Table Of Contents

What Are Signals?

Signals, to be short, are various notifications sent to a process in order to notify it of various "important" events. By their nature, they interrupt whatever the process is doing at this minute, and force it to handle them immediately. Each signal has an integer number that represents it (1, 2 and so on), as well as a symbolic name that is usually defined in the file /usr/include/signal.h or one of the files included by it directly or indirectly (`HUP`, `INT` and so on. Use the command `'kill -l'` to see a list of signals supported by your system).

Each signal may have a signal handler, which is a function that gets called when the process receives that signal. The function is called in "asynchronous mode", meaning that no where in your program you have code that calls this function directly. Instead, when the signal is sent to the process, the operating system stops the execution of the process, and "forces" it to call the signal handler function. When that signal handler function returns, the process continues execution from wherever it happened to be before the signal was received, as if this interruption never occurred.

Note for "hardwarists": If you are familiar with interrupts (you are, right?), signals are very similar in their behavior. The difference is that while interrupts are sent to the operating system by the hardware, signals are sent to the process by the operating system, or by other processes. Note that signals have nothing to do with software interrupts, which are still sent by the hardware (the CPU itself, in this case).

Sending Signals To Processes

Sending Signals Using The Keyboard

The most common way of sending signals to processes is using the keyboard. There are certain key presses that are interpreted by the system as requests to send signals to the process with which we are interacting:

Ctrl-C

Pressing this key causes the system to send an `INT` signal (`SIGINT`) to the running process. By default, this signal causes the process to immediately terminate.

Ctrl-Z

Pressing this key causes the system to send a `TSTP` signal (`SIGTSTP`) to the running process. By default, this signal causes the process to suspend execution.

Ctrl-\

Pressing this key causes the system to send a `ABRT` signal (`SIGABRT`) to the running process. By default, this signal causes the process to immediately terminate. Note that this redundancy (i.e. Ctrl-\ doing the same as Ctrl-C) gives us some better flexibility. We'll explain that later on.

Sending Signals From The Command Line

Another way of sending signals to processes is done using various commands, usually internal to the shell:

`kill`

The kill command accepts two parameters: a signal name (or number), and a process ID. Usually the syntax for using it goes something like:

```
kill -<signal> <PID>
```

For example, in order to send the `INT` signal to process with PID 5342, type:

```
kill -INT 5342
```

This has the same affect as pressing Ctrl-C in the shell that runs that process.
If no signal name or number is specified, the default is to send a `TERM` signal to the process, which normally causes its termination, and hence the name of the `kill` command.

`fg`

On most shells, using the `'fg'` command will resume execution of the process (that was suspended with Ctrl-Z), by sending it a `CONT` signal.

Sending Signals Using System Calls

A third way of sending signals to processes is by using the kill system call. This is the normal way of sending a signal from one process to another. This system call is also used by the `'kill'` command or by the `'fg'` command. Here is an example code that causes a process to suspend its own execution by sending itself the `STOP` signal:

```
#include <unistd.h>     /* standard unix functions, like getpid()       */
#include <sys/types.h>  /* various type definitions, like pid_t         */
#include <signal.h>     /* signal name macros, and the kill() prototype */

/* first, find my own process ID */
pid_t my_pid = getpid();

/* now that i got my PID, send myself the STOP signal. */
kill(my_pid, SIGSTOP);
```

An example of a situation when this code might prove useful, is inside a signal handler that catches the TSTP signal (Ctrl-Z, remember?) in order to do various tasks before actually suspending the process. We will see an example of this later on.

Catching Signals - Signal Handlers

Catchable And Non-Catchable Signals

Most signals may be caught by the process, but there are a few signals that the process cannot catch, and cause the process to terminate. For example, the `KILL` signal (`-9` on all unices I've met so far) is such a signal. This is why you usually see a process being shut down using this signal if it gets "wild". One process that uses this signal is a system shutdown process. It first sends a `TERM` signal to all processes, waits a while, and after allowing them a "grace period" to shut down cleanly, it kills whichever are left using the `KILL` signal.

`STOP` is also a signal that a process cannot catch, and forces the process's suspension immediately. This is useful when debugging programs whose behavior depends on timing. Suppose that process A needs to send some data to process B, and you want to check some system parameters after the message is sent, but before it is received and processed by process B. One way to do that would be to send a `STOP` signal to process B, thus causing its suspension, and then running process A and waiting until it sends its oh-so important message to process B. Now you can check whatever you want to, and later on you can use the `CONT` signal to continue process B's execution, which will then receive and process the message sent from process A.

Now, many other signals are catchable, and this includes the famous `SEGV` and `BUS` signals. You probably have seen numerous occasions when a program has exited with a message such as *'Segmentation Violation - Core Dumped'*, or *'Bus Error - core dumped'*. In the first occasion, a `SEGV` signal was sent to your program due to accessing an illegal memory address. In the second case, a `BUS` signal was sent to your program, due to accessing a memory address with invalid alignment. In both cases, it is possible to catch these signals in order to do some cleanup - kill child processes, perhaps remove temporary files, etc. Although in both cases, the memory used by your process is most likely corrupt, it's probable that only a small part of it was corrupt, so cleanup is still usually possible.

Default Signal Handlers

If you install no signal handlers of your own (remember what a signal handler is? yes, that function handling a signal?), the runtime environment sets up a set of default signal handlers for your program. For example, the default signal handler for the `TERM` signal calls the `exit()` system call. The default handler for the `ABRT` signal calls the `abort()` system call, which causes the process's memory image to be dumped into a file named 'core' in the process's current directory, and then exit.

Installing Signal Handlers

There are several ways to install signal handlers. We'll use the most basic form here, and refer you to your manual pages for further reading.

The `signal()` System Call

The `signal()` system call is used to set a signal handler for a single signal type. `signal()` accepts a signal number and a pointer to a signal handler function, and sets that handler to accept the given signal. As an example, here is a code snippest that causes the program to print the string "Don't do that" when a user presses Ctrl-C:

```
#include <stdio.h>     /* standard I/O functions                          */
#include <unistd.h>    /* standard unix functions, like getpid()         */
#include <sys/types.h> /* various type definitions, like pid_t           */
#include <signal.h>    /* signal name macros, and the signal() prototype */

/* first, here is the signal handler */
void catch_int(int sig_num)
{
    /* re-set the signal handler again to catch_int, for next time */
    signal(SIGINT, catch_int);
    /* and print the message */
    printf("Don't do that");
    fflush(stdout);
}

.
.
.
/* and somewhere later in the code.... */
.
.

/* set the INT (Ctrl-C) signal handler to 'catch_int' */
signal(SIGINT, catch_int);

/* now, lets get into an infinite loop of doing nothing. */
for ( ;; )
    pause();
```

The complete source code for this program is found in the catch-ctrl-c.c file.

Notes:

- the `pause()` system call causes the process to halt execution, until a signal is received. it is surely better then a 'busy wait' infinite loop.
- the name of a function in C/C++ is actually a pointer to the function, so when you're asked to supply a pointer to a function, you may simply specify its name instead.
- On some systems (such as Linux), when a signal handler is called, the system automatically resets the signal handler for that signal to the default handler. Thus, we re-assign the signal handler immediately when entering the handler function. Otherwise, the next time this signal is received, the process will exit (default behavior for `INT` signals). Even on systems that do not behave in this way, it still won't hurt, so adding this line always is a good idea.

Pre-defined Signal Handlers

For our convenience, there are two pre-defined signal handler functions that we can use, instead of writing our own: `SIG_IGN` and `SIG_DFL`.

`SIG_IGN`:

> Causes the process to ignore the specified signal. For example, in order to ignore Ctrl-C completely (useful for programs that must NOT be interrupted in the middle, or in critical sections), write this:
>
> ```
> signal(SIGINT, SIG_IGN);
> ```

`SIG_DFL`:

> Causes the system to set the default signal handler for the given signal (i.e. the same handler the system would have assigned for the signal when the process started running):
>
> ```
> signal(SIGTSTP, SIG_DFL);
> ```

Avoiding Signal Races - Masking Signals

One of the nasty problems that might occur when handling a signal, is the occurrence of a second signal while the signal handler function executes. Such a signal might be of a different type then the one being handled, or even of the same type. Thus, we should take some precautions inside the signal handler function, to avoid races.

Luckily, the system also contains some features that will allow us to block signals from being processed. These can be used in two 'contexts' - a global context which affects all signal handlers, or a per-signal type context - that only affects the signal handler for a specific signal type.

Masking signals with sigprocmask()

the (modern) "POSIX" function used to mask signals in the global context, is the `sigprocmask()` system call. It allows us to specify a set of signals to block, and returns the list of signals that were previously blocked. This is useful when we'll want to restore the previous masking state once we're done with our critical section. `sigprocmask()` accepts 3 parameters:

`int how`

> defines if we want to add signals to the current mask (`SIG_BLOCK`), remove them from the current mask (`SIG_UNBLOCK`), or completely replace the current mask with the new mask (`SIG_SETMASK`).

`const sigset_t *set`

> The set of signals to be blocked, or to be added to the current mask, or removed from the current mask (depending on the 'how' parameter).

`sigset_t *oldset`

> If this parameter is not `NULL`, then it'll contain the previous mask. We can later use this set to restore the situation back to how it was before we called `sigprocmask()`.

Note: Older systems do not support the `sigprocmask()` system call. Instead, one should use the `sigmask()` and `sigsetmask()` system calls. If you have such an operating system handy, please read the manual pages for these system calls. They are simpler to use then sigprocmask, so it shouldn't be too hard understanding them once

you've read this section.

You probably wonder what are these `sigset_t` variables, and how they are manipulated. Well, i wondered too, so i went to the manual page of `sigsetops`, and found the answer. There are several functions to handle these sets. Lets learn them using some example code:

```
/* define a new mask set */
sigset_t mask_set;
/* first clear the set (i.e. make it contain no signal numbers) */
sigemptyset(&mask_set);
/* lets add the TSTP and INT signals to our mask set */
sigaddset(&mask_set, SIGTSTP);
sigaddset(&mask_set, SIGINT);
/* and just for fun, lets remove the TSTP signal from the set. */
sigdelset(&mask_set, SIGTSTP);
/* finally, lets check if the INT signal is defined in our set */
if (sigismember(&mask_set, SIGINT)
    printf("signal INT is in our set\n");
else
    printf("signal INT is not in our set - how strange...\n");
/* finally, lets make the set contain ALL signals available on our system */
sigfillset(&mask_set)
```

Now that we know all these little secrets, lets see a short code example that counts the number of Ctrl-C signals a user has hit, and on the 5th time (note - this number was "Stolen" from some quite famous Unix program) asks the user if they really want to exit. Further more, if the user hits Ctrl-Z, the number of Ctrl-C presses is printed on the screen.

```
/* first, define the Ctrl-C counter, initialize it with zero. */
int ctrl_c_count = 0;
#define CTRL_C_THRESHOLD        5

/* the Ctrl-C signal handler */
void catch_int(int sig_num)
{
    sigset_t mask_set;  /* used to set a signal masking set. */
    sigset_t old_set;   /* used to store the old mask set.   */

    /* re-set the signal handler again to catch_int, for next time */
    signal(SIGINT, catch_int);
    /* mask any further signals while we're inside the handler. */
    sigfillset(&mask_set);
    sigprocmask(SIG_SETMASK, &mask_set, &old_set);

    /* increase count, and check if threshold was reached */
    ctrl_c_count++;
    if (ctrl_c_count >= CTRL_C_THRESHOLD) {
        char answer[30];

        /* prompt the user to tell us if to really exit or not */
        printf("\nRealy Exit? [y/N]: ");
```

```
        fflush(stdout);
        gets(answer);
        if (answer[0] == 'y' || answer[0] == 'Y') {
            printf("\nExiting...\n");
            fflush(stdout);
            exit(0);
        }
        else {
            printf("\nContinuing\n");
            fflush(stdout);
            /* reset Ctrl-C counter */
            ctrl_c_count = 0;
        }
    }
    /* restore the old signal mask */{{/COMMENT_FONT}*/
    sigprocmask(SIG_SETMASK, &old_set, NULL);
}

/* the Ctrl-Z signal handler */
void catch_suspend(int sig_num)
{
    sigset_t mask_set;  /* used to set a signal masking set. */
    sigset_t old_set;   /* used to store the old mask set.   */

    /* re-set the signal handler again to catch_suspend, for next time */
    signal(SIGTSTP, catch_suspend);
    /* mask any further signals while we're inside the handler. */
    sigfillset(&mask_set);
    sigprocmask(SIG_SETMASK, &mask_set, &old_set);

    /* print the current Ctrl-C counter */
    printf("\n\nSo far, '%d' Ctrl-C presses were counted\n\n", ctrl_c_count);
    fflush(stdout);

    /* restore the old signal mask */
    sigprocmask(SIG_SETMASK, &old_set, NULL);
}

.
.
/* and somewhere inside the main function... */
.
.

/* set the Ctrl-C and Ctrl-Z signal handlers */
signal(SIGINT, catch_int);
signal(SIGTSTP, catch_suspend);
.
.
/* and then the rest of the program */
.
.
```

The complete source code for this program is found in the count-ctrl-c.c file.

You should note that using `sigprocmask()` the way we did now does not resolve all possible race conditions. For example, It is possible that after we entered the signal handler, but before we managed to call the `sigprocmask()` system call, we receive another signal, which WILL be called. Thus, if the user is VERY quick (or the system is very slow), it is possible to get into races. In our current functions, this will probably not disturb the flow, but there might be cases where this kind of race could cause problems.

The way to guarantee no races at all, is to let the system set the signal masking for us before it calls the signal handler. This can be done if we use the `sigaction()` system call to define both the signal handler function AND the signal mask to be used when the handler is executed. You would probably be able to read the manual page for `sigaction()` on your own, now that you're familiar with the various concepts of signal handling. On old systems, however, you won't find this system call, but you still might find the `sigvec()` call, that enables a similar functionality.

Implementing Timers Using Signals

One of the weak aspects of Unix-like operating systems is their lack of proper support for timers. Timers are important to allow one to check timeouts (e.g. wait for user input up to 30 seconds, or exit), check some conditions on a regular basis (e.g. check every 30 seconds that a server we're talking to is still active, or close the connection and notify the user about the problem), and so on. There are various ways to get around the problem for programs that use an "event loop" based on the `select()` system call (or its new replacement, the `poll()` system call), but not all programs work that way, and this method is too complex for short and simple programs.

Yet, the operating system gives us a simple way of setting up timers that don't require too much hassles, by using special alarm signals. They are generally limited to one timer active at a time, but that will suffice in simple cases.

The `alarm()` System Call

The `alarm()` system call is used to ask the system to send our process a special signal, named `ALRM`, after a given number of seconds. Since Unix-like systems don't operate as real-time systems, your process might receive this signal after a longer time then requested. Combining this system call with a proper signal handler enables us to do some simple tricks. Lets see an example of a program that waits for user input, but exits if none was given after a certain timeout.

```
#include <unistd.h>    /* standard unix functions, like alarm()          */
#include <signal.h>    /* signal name macros, and the signal() prototype */

char user[40];          /* buffer to read user name from the user */

/* define an alarm signal handler. */
void catch_alarm(int sig_num)
{
    printf("Operation timed out. Exiting...\n\n");
    exit(0);
}

.
.
/* and inside the main program... */
.
```

```
.

/* set a signal handler for ALRM signals */
signal(SIGALRM, catch_alarm);

/* prompt the user for input */
printf("Username: ");
fflush(stdout);
/* start a 30 seconds alarm */
alarm(30);
/* wait for user input */
gets(user);
/* remove the timer, now that we've got the user's input */
alarm(0);

.
.
/* do something with the received user name */
.
.
```

The complete source code for this program is found in the use-alarms.c file.

As you can see, we start the timer right before waiting for user input. If we started it earlier, we'll be giving the user less time then promised to perform the operation. We also stop the timer right after getting the user's input, before doing any tests or processing of the input, to avoid a random timer from setting off due to slow processing of the input. Many bugs that occur using the `alarm()` system call occur due to forgetting to set off the timer at various places when the code gets complicated. If you need to make some tests during the input phase, put the whole piece of code in a function, so it'll be easier to make sure that the timer is always set off after calling the function. Here is an example of how NOT to do this:

```
int read_user_name(char user[])
{
    int ok_len;
    int result = 0; /* assume failure */

    /* set an alarm to timeout the input operation */
    alarm(3); /* Mistake 1 */
    printf("Enter user name: "); /* Mistake 2 */
    fflush(stdout);
    if (gets(user) == NULL) {
        printf("End of input received.\n");
        return 0;               /* Mistake 3 */
    }
    /* count the size of prefix of 'user' made only of characters */
    /* in the given set. if all characters in 'user' are in the   */
    /* set, then ok_len with be equal to the length of 'user'.    */
    ok_len = strspn(user, "abcdefghijklmnopqrstuvwxyz0123456789");
    if (ok_len == strlen(user)) {
        /* check if the user exists in our database */
        result = find_user_in_database(user); /* Mistake 4 */
```

```
    }
    alarm(0);

    return result;
}
```

Lets count the mistakes and bad programming practices in the above function:

1. * Too short timeout:

   3 seconds might be enough for superman to type his user name, but a normal user obviously needs more time. Such a timeout should actually be tunable, because not all people type at the same pace, or should be long enough for even the slowest of users.
2. * Printing while timer is ticking:

   Norty Norty. This printing should have been done before starting up the timer. Printing may be a time consuming operation, and thus will leave less time then expected for the user to type in the expected input.
3. * Exiting function without turning off timer:

   This kind of mistake is hard to catch. It will cause the program to randomally exit somewhere LATER during the execution. If we'll trace it with a debugger, we'll see that the signal was received while we were already executing a completely different part of the program, leaving us scratching our head and looking up to the sky, hoping somehow inspiration will fall from there to guide us to the location of the problem.
4. * Making long checks before turning off timer:

   As if it's not enough that we gave the poor user a short time to check for the input, we're also inserting some database checking operation while the timer is still ticking. Even if we also want to timeout the database operation, we should probably set up a different timer (and a different `ALRM` signal handler), so as not to confuse a slow user with a slow database server. It will also allow the user to know *why* we are timing out. Without this information, a person trying to figure out why the program suddenly exits, will have hard time finding where the fault lies.

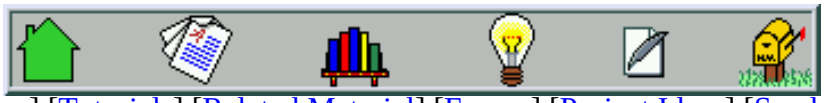
Summary - "Do" and "Don't" inside A Signal Handler

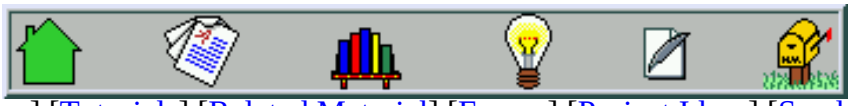
We have seen a few "thumb rules" all over this tutorial, and there are quite many of those, as the area of signals handling is rather tricky. Lets try to summarize these rules of thumb here:

- Make it short - the signal handler should be a short function that returns quickly. Instead of doing complex operations inside the signal handler, it is better that the function will raise a flag (e.g. a global variable, although these are evil by themselves) and have the main program check that flag occasionally.
- Proper Signal Masking - don't be too lazy to define proper signal masking for a signal handler, preferably using the `sigaction()` system call. It takes a little more effort then just using the `signal()` system call, but it'll help you sleep better at night, knowing that you haven't left an extra place for race conditions to occur. Remember - if some bug has a probability of 1/10,000 to occur, it WILL occur when many people use that program many times, as tends to be the case with good programs (you write only good programs, no?).
- Careful with "fault" signals - If you catch signals that indicate a program bug (`SIGBUS`, `SIGSEGV`, `SIGFPE`), don't try to be too smart and let the program continue, unless you know exactly what you are doing (which is a very rare case) - just do the minimal required cleanup, and exit, preferably with a core dump (using the `abort()` function). Such signals usually indicate a bug in the program, that if ignored will most likely cause it to crush sooner or later, making you think the problem is somewhere else in the code.
- Careful with timers - when you use timers, remember that you can only use one timer at a time, unless you also (ab)use the `VTALRM` signal. If you need to have more then one timer active at a time, don't use signals, or devise a set of functions that will allow you to have several virtual timers using a delta list of some sort. If

you've no idea what I'm talking about, you probably don't need several simultaneous timers in the first place.

- Signals are NOT an event driven framework - it is easy to get carried away and try turning the signals system into an event-driven driver for a program, but signal handling functions were not meant for that. If you need such a thing, use some framework that is more suitable for the application (e.g. use an event loop of a windowing program, use a select-based loop inside a network server, etc.).





Network programming under Unix systems

Preface, or - who is this for?

This document is meant to provide people who already have a knowledge of Programming in C, with the knowledge necessary to write Unix programs that use the network (actually, the Internet). It is supposed to save you all the time it took me to learn how to do this, due to lack of decent online documentation about the subject.

The idea is to explain only the really necessary information for writing client and server applications, leaving less "urgent" information for the appendices, and even less important information for the "see also" part. By the way, I'm not providing an Index, because usually indexed documents scare me away, and I want this document to look friendly to *me*, hoping it will also look friendly to you.

OK, lets get down to business.

Table of contents

(yeah, I know I promised to avoid an index, but an un-detailed one is necessary)

Appendices:

---

1. Internet

(Skip this if you know what Internet is, what protocols it uses, what kind of addresses are used over Internet, etc).

Preview

The Internet is a computer communication network. Every computer connected to the Internet is also known as a "host", so we could say that Internet's role is to allow hosts to talk amongst themselves. I assume you are already familiar with Internet, as a user of programs such as 'Telnet', 'Ftp', 'Irc' and others. Lets first discuss Internet addresses a little, before we talk about the Internet protocols, and various programming aspects regarding network uniformity.

Internet Addresses

(Skip this if you know what IP addresses are and what ports in Internet are).

An Internet address (or an IP address, or an IP number) is a number made of 4 bytes (numbers between 0 and 255), written in what is called 'a dot notation'. for example, "128.0.46.1" is a valid Internet address. Such an address identifies a computer which is connected to the Internet. Note that a computer might have more then one such address, if it has more then one physical connections to the Internet (such as having two Ethernet cards connected. What's Ethernet? read the appendices, or ignore this).

However, when a normal human being uses Internet, they usually use human-readable (oh, really?) addresses of the form "uunet.uu.net" or "wuarchive.wustl.edu". A system in the Internet called "The Domain Name System" or DNS for short, is responsible to translate between human-readable addresses and IP addresses (again, read the appendices for information about DNS). You will not have to know anything about DNS in order to use it in your programs, so do not worry about it now.

OK. We said that IP numbers define a computer. Well, usually there is more then one program that wants to use the network, that runs on a given computer. For this purpose, the Internet people made up some extension for an IP address, called a port number. Each communications address is made up of an IP number AND a port number. Port numbers could be any number between 1 and 65535, although for certain reasons, you will use port numbers above 1024, unless you have a superuser privileges on your machine (also stated sometimes as "having root password").

For our purposes, we will use addresses of the form: 114.58.1.6:6072 where the "114.58.1.6" part is the IP number, and the "6072" part is the port number. Remember this for later usage.

Internet protocols

(Skip this if you know what IP, TCP and UDP are).

You probably heard the term "TCP/IP" and wondered, or had some vague idea about what it means. Lets make things clearer:

The Internet is a network. In order to talk on a network, you need some kind of a "language". That language is also called "a protocol". The Internet has many kinds of protocols used to talk on it, in a manner called "layering".

Layering means that instead of defining one protocol that will do everything, which will be very hard to design

and implement, the tasks are divided between several protocols, sitting on top of each other.

What does this mean? Think about it as sending letters: you write your letter on a paper, and then put it in an envelope and write the address on it. The postman doesn't care WHAT you wrote in your letter, as long as you wrote the correct address on the envelope.

The same thing is done when layering protocols. If one protocol contains the data sent and knows how to make sure the data is correct, then a lower protocol will contain the address to be used, and will know how to transfer the data to the correct target. The lower protocol does not understand the format of the data in the upper protocol, and the upper protocol doesn't have to know how to actually transfer the data. This way, each protocol becomes much simpler to design and test. Furthermore, If we will want to use the same protocol for writing the data, but send the data on a different network, we will only need to replace the protocol that knows how to transfer the data over the network, not the whole set of protocols.

(By the way, This sort of packing up several protocols on top of each other is called Encapsulation.)

One other important notion about the Internet is that it forms something known as a "packet switching network". This means that every message sent out is divided into small amounts of information, called packets. The different protocols send the data in packets, which might get divided into smaller packets while it travels to the target host, due to "Electrical" limitations of physical networks. The target machine will eventually combine the small packets (also known as fragments) and build the original message again.

The packets are what allows several connections to use the same physical network simultaneously, in a manner transparent to the network users. No machine will take over a line completely, even if it needs to send a large message. Instead, it will send the message in small fragments, allowing other machines to send their packets too.

Let us now name out some of the Internet protocols, and explain briefly each one of them:

IP

IP is the basic protocol used on the Internet. It is responsible to make it possible to transfer packets from one computer to another, given their IP addresses. Most other protocols are placed on top of IP. As a programmer you will usually not use IP directly.

TCP

TCP is the most useful Internet protocol for a programmer. Most networking programs you use (such as 'Telnet', and 'Ftp') are placed on top of TCP. TCP is placed on top of IP, and adds 3 functionalities:

1. The notion of ports (IP supports only Internet addresses, as mentioned above).
2. The notion of a 'safe' protocol, i.e. no errors will be encountered in packets sent using TCP, so no error correcting mechanisms are required in programs using TCP (well, this is almost true, but lets assume so for now).
3. Connection-mode link. After you make a connection between two programs using TCP, you have a steadily open connection on which you can send data without having to specify who you want the data to be sent to. This works very similar to reading or writing on a pipeline, or on a regular file.

UDP

UDP is another protocol that is placed on top of IP. It is used for services that send small amounts of data between programs that do not require a long-time connection, or that send only little amount of data at a time. The 'talk' program uses the UDP protocol.

UDP adds only port numbers to the functionality IP gives, so the programmer needs to worry about checking for errors in messages (that come due to line noises and such), making sure the data sent using UDP arrives in the right order (which is not automatically achieved, due to IP's nature of not having a

constantly open connection), and such.

There are various other protocols used in conjunction with IP, such as ARP, RARP, ICMP, SNMP and others, but those won't be dealt with in this document. Look at the 'See Also' part to find pointers to articles discussing those protocols.

Network uniformity

(Skip this if you already know what byte ordering means, and what are 'well known ports' across Internet)

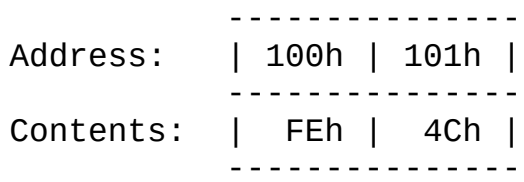
It is important to understand that protocols need to define some low-level details, in order to be able to talk to each other. We will discuss two such aspects here, in order to understand the example programs given later on.

Byte Order

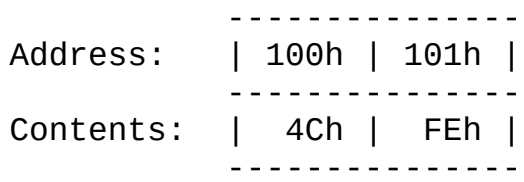
It is an old argument amongst different computer manufacturers how numbers should be kept in a computer.

As all computers divide memory into bytes (or octets) of information, each 8 bit long, there is no problem with dealing with byte-sized numbers. The problem arises as we use larger numbers: short integers (2 bytes long) and long integers (4 bytes long). Suppose we have a short integer number, FE4Ch (that is, FE4C in hexadecimal notation). Suppose also that we say this number is kept in memory address 100h. This could mean one of two things, lets draw them out:

1. Big Endian:



2. Little Endian:



In the first form, also called 'Big Endian', The Most Significant Byte (MSB) is kept in the lower address, while the Least significant Byte (LSB) is kept in the higher address.

In the second form, also called 'Little Endian', the MSB is kept in the higher address, while the LSB is kept in the lower address.

Different computers used different byte ordering (or different endianess), usually depending on the type of CPU they have. The same problem arises when using a long integer: which word (2 bytes) should be kept first in memory? the least significant word, or the most significant word?

In a network protocol, however, there must be a predetermined byte and word ordering. The IP protocol

defines what is called 'the network byte order', which must be kept on all packets sent across the Internet. The programmer on a Unix machine is not saved from having to deal with this kind of information, and we'll see how the translation of byte orders is solved when we get down to programming.

Well Known Ports

When we want two programs to talk to each other across Internet, we have to find a way to initiate the connection. So at least one of the 'partners' in the conversation has to know where to find the other one. This is done by letting one partner know the address (IP number + port number) of the other side.

However, a problem could arise if one side's address is randomly taken over by a third program. Then we'll be in real trouble. In order to avoid that, There are some port numbers which are reserved for specific purposes on any computer connected to the Internet. Such ports are reserved for programs such as 'Telnet', 'Ftp' and others.

These port numbers are specified in the file /etc/services on any decent Unix machine. Following is an excerpt from that file:

```
daytime         13/tcp
daytime         13/udp
netstat         15/tcp
qotd            17/tcp      quote
chargen         19/tcp      ttytst source
chargen         19/udp      ttytst source
ftp-data        20/tcp
ftp             21/tcp
telnet          23/tcp
smtp            25/tcp      mail
```

Read that file to find that Telnet, for example, uses port 23, and Ftp uses port 21. Note that for each kind of service, not only a port number is given, but also a protocol name (usually TCP or UDP). Note also that two services may use the same port number, provided that they use different protocols. This is possible due to the fact that different protocols have what is called different address spaces: port 23 of a one machine in the TCP protocol address space, is not equivalent to port 23 on the same machine, in the UDP protocol address space.

So how does this relate to you? When you will write your very own programs that need to 'chat' over Internet, you will need to pick an unused port number that will be used to initiate the connection.

---

2. Client and Server model

(Skip this if you already know what clients and servers are, and what are the relations between them).

This section will discuss the most common type of interaction across the Internet - the Client and Server model. Note that this discussion is relevant to other types of networks too, and a few examples will be mentioned along the text.

We will first explain what the client-server model is, then detail the roles of the client, the roles of the server, and give examples of some famous servers and clients used.

The Client-Server model

(Skip this if you know what the client-server model basically means)

The client-server model is used to divide the work of Internet programs into two parts. One part knows how to do a certain task, or to give a certain service. This part is called the Server. The other part knows how to talk to a user, and connect that user to the server. this part is called the Client. One server may give service to many different clients, either simultaneously , or one after the other (the server designer decides upon that). on the other hand, a Client talks to a single user at a time, although it might talk to several servers, if it's nature requires that. There are other such complex possibilities, but we will discuss only clients that talk to a single server.

Roles of Clients

(Skip this if you already know what clients are supposed to do)

A client's main feature is giving a convenient User interface, hiding the details of how the server 'talks' from the user. Today, people are trying to write mostly graphical clients, using windows, pop-up-menus and other such fancy stuff. We will leave this to someone else to explain, and concentrate on the networking part. The client needs to first establish a connection with the server, given it's address. After the connection is established, The Client needs to be able to do two things:

1. Receive commands from the user, translate them to the server's language (protocol) and send them to the server.
2. Receive messages from the server, translate them into human-readable form, and show them to the user. Some of the messages will be dealt with by the client automatically, and hidden from the user. This time, the Client designer's choice.

This forms the basic loop a client performs:

```
get the server's address
form a working address that can be used to talk over Internet.
connect to the server
while (not finished) do:
  wait until there's either information from the server, or from the
      user.
  If (information from server) do
    parse information
    show to user, update local state information, etc.
  else {we've got a user command}
    parse command
    send to server, or deal with locally.
done
```

In the end of this tutorial you will be able to write such clients.

Roles of Servers

(Skip this if you already know what servers are supposed to do)

A server main feature is to accept requests from clients, handle them, and send the results back to the clients. We will discuss two kinds of servers: a Single-client server, and a multi-client server.

Single Client Servers

These are servers that talk to a single client at a time. They need to be able to:

1. Accept connection requests from a Client.
2. Receive requests from the Client and return results.
3. Close the connection when done, or clear it if it's broken from some reason.

this forms the main loop a Single-Client Server performs:

```
bind a port on the computer, so Clients will be able to connect
forever do:
  listen on the port for connection requests.
  accept an incoming connection request
  if (this is an authorized Client)
    while (connection still alive) do:
      receive request from client
      handle request
      send results of request, or error messages
    done
  else
    abort the connection
done
```

Multi Client Servers

These are servers that talk to a several Clients at the same time. They need to be able to:

1. Accept new connection requests from Clients.
2. Receive requests from any Client and return results.
3. Close any connection that the client wants to end.

this forms the main loop a Multi-Client Server performs:

```
bind a port on the computer, so Clients will be able to connect
listen on the port for connection requests.
forever do:
  wait for either new connection requests, or requests from existing
          Clients.
  if (this is a new connection request)
    accept connection
    if (this is an un-authorized Client)
      close the connection
  else if (this is a connection close request)
    close the connection
  else { this is a request from an existing Client connection}
```

```
        receive request from client
        handle request
        send results of request, or error messages
    done
```

"Famous" Servers and Clients

(Skip this if you already know about too many server types, or if you're not interested in knowing about them)

In this section we will give short descriptions of some "famous" servers and clients, that are used daily over Internet, and over some other famous kinds of networks. This is simply an illustrative section, that can be safely skipped by a rushing reader.

Internet Clients

FTP

Ftp is a Client program used to transfer files across the Internet. The Client connects to an FTPD Server (see below) and allows a user to scan a tree-structure of files and directories on the remote machine, retrieve and transmit files, etc. It uses two connections - one to exchange commands, and another to exchange data (the files themselves).

TELNET

Telnet is a Client program that enables working on a remote machine, using the keyboard and screen of the local one. It allows connecting from one type of machine to a completely different type. The Telnet Client could be used to connect to any ASCII services, although it might be more convenient to use specialized Clients for that.

TALK

talk is a Client that allows two users across the Internet to carry out a real-time chat. The screen is split into two halves, and the user see what he types in the upper half, and what the other user types in the upper half.

IRC

Another chat Client, which allows connecting to Internet-wide network of users and carry out both private conversations and conference talks.

LYNX

An information-retrieval Client. Lynx knows a wide set of information retrieval protocols, including some hyper-text protocols (which allow a plain text file to contain pointers to relevant information that lynx can follow), ftp, and so on.

Internet Servers

FTPD

The server that normally accepts connections from Ftp Clients. It works off the well-known Ftp port number 21. This server is a Single-Client server, i.e. it handles one connection only, and then terminates. The operating system has to make sure one running copy of the server will be created for each interested client.

TELNETD

The Server that talks to the Telnet client. Uses the well-known Telnet port number 23. TELNETD is a Single-Client Server, i.e. one TELNETD server is used for each Telnet connection request that should be handled.

TALKD

Used to receive talk requests from a remote machine, and notify the user that such a request has

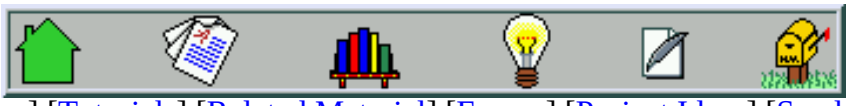
arrived. It has a role only in the initiation of the connection, not during the conversation itself, which is carried directly between the two Clients.

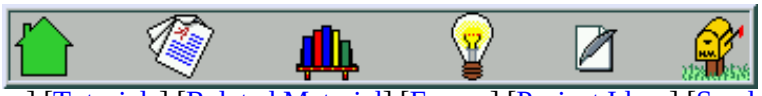
HTTPD

A server that serves information for hyper-text Clients such as Lynx, or graphical-based ones (Netscape, Internet Explorer.... or have you heard of Mosaic?).

IRCD

A server that cooperates with other servers of its kind to form the IRC network. Allows exchanging of information, along with state information about channels used for conferencing, and such.





v1.0.1

Accessing User Information On A Unix System

Table Of Contents:

Preface

One type of information that many programs require, is user information. This includes finding out the home directory of a given user, finding the list of groups a user belongs to, finding out who is logged on the system at a given time and so on.

Most Unix systems keep this data in a set of files spread across the system. In order to allow programs to access this data in a portable manner, some system functions (or API - Application Programming Interface) and data structures are defined, and documented in different manual pages. This tutorial is an attempt at collecting the information that is common across most Unix systems into one location, and at giving a few examples of using this information.

You will note one great problem with these APIs - they are exclusively read-only. You can use them in order to read user information, but not to write out anything (i.e. create new data, update existing data or delete existing data). This makes these APIs useless for writing a program that allows (for example) creation of new users, addition of groups for a user and so on. Some hints will be supplied here as to how these tasks could be accomplished, but should be taken as time-bombs - do NOT rely only on this information if you are trying to write such programs, as you might find your system's data corrupt, your list of users erased, or something similar - DO THIS AT YOUR VERY OWN RISK. Now that we have made this point clear, we can get started.

Accessing Info About User Accounts

The most basic info one would want to know about user accounts is - what user accounts exist on a system? or, is a given user name valid on this system? This information is traditionally kept in the file named "/etc/passwd", and on current Unix systems, in other files as well.

Structure Of The /etc/passwd File

The "/etc/passwd" file contains one line (or record) per user account, and may NOT contain any comments. Each line is made of a set of fields, separated by a colon (':') character. An example of such an line would be:

```
root:UQstWvHV002G1:0:0:root:/root:/bin/tcsh
```

This example shows a "root" (or superuser) account's entry. Most fields serve the same purpose on all Unix systems, and would be as follows:

User Name
: The name used to identify this account in all human-readable printouts, as well as when the user tries to log on to the system.

Password
: The encrypted password of the user (see the manual page of "crypt" for more information on password encryption). on many systems, this field may also serve as a marker that the password is kept elsewhere - see discussion of shadow passwords below.

User ID
: The numeric ID of the given user on the system - this number is used internally by the system instead of the user name, mostly because numbers occupy a fixed size in binary format (usually 2 or 4 bytes), and are much faster to handle, to use as keys in search tables, etc.

User's Group ID
: The numeric ID of a group this user belongs to. Every user belongs to at least one group on the system. Additional groups may be specified for a user using the /etc/group file.

GECOS field
: This field is a general-purpose field in which a comment about this user may be kept. This will often contain the name of the account holder. Some systems use this to store the phone number, address and office room number of the account holder. The contents of this field are never crucial for the account's proper operation (on a non-modified system).

Home Directory
: Full path to the home directory of this user. This will be used by the login program to place this user in this directory when logged on, for example.

Shell.
: The default shell this user will use. This shell is launched for the user when logging on to the system.

The "/etc/passwd" file is readable for all users on the system, but writable only to its owner - root. This means that users cannot directly change the contents of this file - instead, there are specialized programs that allow a user to change her password, default shell and GECOS field. See the manual page for the "passwd" command for more info on these programs.

Getting A Given User's Info

The first API we will introduce is the function for fetching the "/etc/passwd" entry of a given user. There are two functions for doing that - `getpwnam()` and `getpwuid()`.

`getpwnam` is used to fetch a "/etc/passwd" record of a user, given his user name. The function will return a pointer to a `struct passwd`, or NULL, if the user does not exist on the system. This structure is defined as follows:

```
struct passwd {
```

```
    char    *pw_name;       /* user name */
    char    *pw_passwd;     /* user password */
    uid_t   pw_uid;         /* user id */
    gid_t   pw_gid;         /* group id */
    char    *pw_gecos;      /* real name */
    char    *pw_dir;        /* home directory */
    char    *pw_shell;      /* shell program */
};
```

Here is an example of using this function:

```
#include <pwd.h> /* defines 'struct passwd', and getpwnam(). */
#include <sys/types.h> /* defines 'uid_t', etc.               */

/* lets print some information about our root account. */
struct passwd* user_info = getpwnam("root");

/* make sure the root account exists. */
if (!user_info) {
    printf("What?? no root account? this system is broken....\n");
}
else { /* information found - lets print it out. */
    printf("Information for user 'root':\n");
    printf("  User ID: %d\n", user_info->pw_uid);
    printf("  Group ID: %d\n", user_info->pw_gid);
    printf("  Home Directory: '%s'\n", user_info->pw_dir);
    printf("  Default Shell: '%s'\n", user_info->pw_shell);
}
```

Just like we used the `getpwnam()` function, we may also use the `getpwuid()` function, supplying it the UID of the given user. For example, we know that the "root" account always has the UID of '0', so doing the above can be done by:

```
/* get the info of user with ID '0'. */
struct passwd* user_info = getpwuid(0);
/* continue like in the previous case.... */
if (!user_info) {
    printf("What?? no root account? this system is broken....\n");
}
else {
   .
   .
}
```

One important note about these two functions: The structure they return is allocated in static memory. This means that the same structure will be re-filled with new information after each call to the same function, erasing any data it returned in a previous call. If you need to get info about two users and have it stored in memory, you should copy the structure returned from the first call, and store it in a `struct passwd` you allocated (as a local variable, or using `malloc()`). Here is an example:

```
/* this function copies one passwd struct into another. */
void
copy_passwd(struct passwd* from_pw, struct passwd* to_pw)
{
    /* "string" fields need to be pre-allocated first. */
    /* copy the user name field. */
```

```
    to_pw->pw_user = (char*)malloc(strlen(from_pw->pw_user)+1);
    if (!to_pw->pw_user) { /* out of memory?? */
        fprintf(stderr, "copy_passwd: out of memory\n");
        exit(1);
    }
    strcpy(to_pw->pw_user, from_pw->pw_user);
    /* copy the passwd field. */
    to_pw->pw_passwd = (char*)malloc(strlen(from_pw->pw_passwd)+1);
    if (!to_pw->pw_passwd) { /* out of memory?? */
        fprintf(stderr, "copy_passwd: out of memory\n");
        exit(1);
    }
    strcpy(to_pw->pw_passwd, from_pw->pw_passwd);
    to_pw->pw_uid = from_pw->pw_uid;
    to_pw->pw_gid = from_pw->pw_gid;
    /* copy the GECOS field. */
    to_pw->pw_gecos = (char*)malloc(strlen(from_pw->pw_gecos)+1);
    if (!to_pw->pw_gecos) { /* out of memory?? */
        fprintf(stderr, "copy_passwd: out of memory\n");
        exit(1);
    }
    strcpy(to_pw->pw_gecos, from_pw->pw_gecos);
    /* copy the home directory field. */
    to_pw->pw_dir = (char*)malloc(strlen(from_pw->pw_dir)+1);
    if (!to_pw->pw_dir) { /* out of memory?? */
        fprintf(stderr, "copy_passwd: out of memory\n");
        exit(1);
    }
    strcpy(to_pw->pw_dir, from_pw->pw_dir);
    /* copy the default shell field. */
    to_pw->pw_shell = (char*)malloc(strlen(from_pw->pw_shell)+1);
    if (!to_pw->pw_shell) { /* out of memory?? */
        fprintf(stderr, "copy_passwd: out of memory\n");
        exit(1);
    }
    strcpy(to_pw->pw_shell, from_pw->pw_shell);
}

/* here comes the main body of the code... */
/* allocating two local passwd structures. */
struct passwd user1_info;
struct passwd user2_info;

/* get info about account 'john'. */
struct passwd* user_info = getpwnam("john");
if (!user_info) {
    printf("Account 'john' does not exist.n");
    exit(1);
}
copy_passwd(user_info, &user1_info);
/* get info about account 'sarah'. */
user_info = getpwnam("sarah");
if (!user_info) {
    printf("Account 'sarah' does not exist.n");
    exit(1);
}
copy_passwd(user_info, &user2_info);
```

```
/* now do something with the information found... */
```

As you can see, the structure copying was done using a "deep copy" function - i.e. every field in the structure is copied in turn, including desired memory allocation. You ought to know that just copying the structure itself is not sufficient...
Note that in a real program, more testing should be made - what if one of the string fields has a NULL value?

Enumerating All User Accounts

The next API we will encounter is made of a set of functions that allow us to scan the full list of user accounts on our system. A set of functions are used here: `getpwent()`, `setpwent()` and `endpwent()`.

The `getpwent()` function will fetch, on first invocation, the `struct passwd` info of the first user account on the system. On the next invocation it will fetch the info of the second user, and so on. After fetching the last user's info, if we call `getpwent()` again, it will return NULL. Here is a sample of printing out the list of user names on our system:

```
struct passwd* user_info;

printf("List of user names:\n");
for (user_info = getpwent(); user_info; user_info = getpwent() ) {
    printf("  %s\n", user_info->pw_user);
}
```

As you can see, this is a rather simple piece of code.

The `setpwent()` function may be used to re-start a scan in the middle. After calling this function, the next invocation of the `getpwent()` will start scanning the user accounts list from the beginning.

The `endpwent()` function will terminate the current scan of the user accounts list. It is used simply to reduce resource usage. When we first called the `getpwent()` function, it actually opened the "/etc/passwd" file for reading, and kept the file open for the next reads. The `endpwent()` function simply closes this file, freeing any resources used to keep it open. thus, it is a good idea to call it when you are done handling the user's information.

Writing A User Authentication Function

Various types of programs need to authenticate users. This is often done by the user supplying a user name and a password, and the program verifying that it has this user name and password combination in its list of users.

The login program on Unix systems works in a similar fashion, thought a little more secure. When a user logges on, she is prompted for a user name and a password. The login program then uses this password as a key for encrypting some constant text, using the `crypt()` function. This produces a string that is then compared to a string stored in the password field of the user. If the strings match - the user is assumed to be authenticated, and is allowed to log in. The `crypt()` function performs an algorithm known as DES (Data Encryption Standard). One property of this algorithm is that there is no way to find the original password given the encrypted string that `crypt()` generates. This means that if a user got a copy of the "/etc/passwd" file, they won't be able to calculate the original passwords of users using a simple algorithm (this does not preclude using other methods to crack passwords...).

Lets see how one uses the `crypt()` function. It accepts two parameters. The first is the key (normally, the password to encrypt) and the second is called "salt". This is a two-character string that is used to further alter the results of crypt. One must make sure that the same salt is used when storing the password (actually the encrypted result of `crypt()`), and when authenticating the user. The login program often uses the first two characters of the user name as the "salt" string. `crypt()` returns an encrypted string when it returns, 13 characters long. Lets see a program that asks a user for their password, and compares it to the password, as taken from the "/etc/passwd" file.

```
#include <unistd.h>   /* crypt(), etc.          */
#include <pwd.h>      /* getpass(), getpwnam(). */
#include <string.h>   /* strcmp(), etc.         */

/* buffers for reading in the user name and the password. */
```

```
char user[21];
char* password;
/* storing the encrypted password, and the salt. */
char* encrypted_password;
char salt[2];
/* user's "/etc/passwd" entry. */
struct passwd* user_info;

/* prompt the user for a user name. */
printf("User name: ");
fflush(stdout); {COMMENT_FONT}}/* flush the prompt to make sure the user sees it. */
fgets(user, 20, stdin);
/* fgets() stores also the new-line that the user typed in. so we */
/* need to locate the new-line character, and truncate it.        */
if (strchr(user, '\n'))
    (*(strchr(user, '\n'))) = '\0';

/* prompt the user for their password. the getpass() function  */
/* prints the given prompt, turns off echo (so the password    */
/* typed won't be seen on screen), and returns the string that */
/* the user types.                                             */
password = getpass("Password: ");

/* find the user's encrypted password, as stored in "/etc/passwd". */
user_info = getpwnam(user);
if (!user_info) {
    printf("login incorrect.\n");
    exit(1);
}

/* take the salt as stored in the password field of the user. */
strncpy(salt, user_info->pw_passwd, 2);

/* encrypt the given password using the found "salt". */
encrypted_password = crypt(password, salt);

/* compare the results of crypt, with the user's stored password field. */
if (strcmp(user_info->pw_passwd, encrypted_password) != 0) {
    printf("login incorrect.\n");
    exit(1);
}

/* authentication succeeded... */
printf("login successful.\n");
```

The source code of this program is also available in the authenticate-user.c file. Note that on Linux systems based on the GNU C library version 2 (glibc2), there is a need to link this program with the crypt library, by adding the option "-lcrypt" to the compilation command.

Some notes about this code:

- Better checking of the user's input should be performed (e.g. making sure the user actually typed something as a user name, making sure that the user name contains at least 2 characters (for the salt), etc.
- The buffer returned by `getpass()` should be over-written immediately after use by the paranoid programmer, to avoid a possibility of someone somehow tracing it in memory while our process is still running.
- Such an algorithm is useful only if it's done in a server, while the user uses some client program to connect to us. This way, we could terminate the connection in case of authorization failure. An interactive program that uses this code can be cracked easily (for example, the user uses a debugger or a disassembler program to skip the authentication

function's execution altogether).

- We always notify the user of the same error message, no matter what happened. For example, if we said "password incorrect" in the second message, the user would know that the user name by itself is valid - something they might have not known prior to the login attempt.

The (Non-Standard) Handling Of Shadow Passwords

Keeping the encrypted password of a user in the "/etc/passwd" file is considered insecure - even thought the encryption algorithm cannot be reversed (i.e. given the encrypted password, there is no algorithm to find out the original password), a user could copy out the "/etc/passwd" file of a system to their own system, and try running a password cracker that will in turn try to guess passwords for all accounts. On most systems, and with most users, this will allow them to guess passwords of a few accounts at least.

In order to reduce this risk, shadow passwords were invented. On a system that uses shadow passwords, the passwords of the users are kept in a different file, that is only readable to its owner - root. the password entry in the "/etc/passwd" file (for each user) will contain a special mark-up character, defining that the actual password is kept in the shadow file. This mark-up is done differently on different operating systems, and the shadow file's path is also different for different systems. Thus, although shadow passwords exist for at least a decade, there is no standard implementation for them.

When reading a user account's info (using `getpwnam()` for example), the password field in the returned structure will contain the place-holder string, rather then the encrypted password. This makes it impossible to write simple programs that authenticate users using their account's password. The solution to this problem is usually a specialized (non-standard) function supplied by the operating system, to perform such authentication. Such a password will normally accept a user name and an encrypted password, and will return '1' if this encrypted password matches that of the given user, or '0' if it does not. This still allows people to write password crackers, but then they must be ran on the same system whose accounts they are trying to break into, making it much easier to spot this activity and to block it (e.g. by disabling an account if too many authentication attempts are made against it, in a given period of time).

Association Of Extra Groups For Users

On a Unix system, it is possible to associate a user with more then one group. This is done using the "/etc/group" file, together with a set of APIs.

Structure Of The /etc/group File

The "/etc/group" file is a text file that contains a set of lines, each defining membership of a list of users in a given group. Each group may be defined by one or more lines, and the users of this group include all the users defined in all these lines, as well as the users having this group's GID (Group ID) in their "/etc/passwd" entries. Here is an example of a valid "/etc/group" line:

```
bin::1:root,bin,daemon
```

This example shows some information about the "bin" group. This line is composed of a set of fields, separated by a colon (':') character (like in the "/etc/passwd" file). The fields serve the following purposes:

Group Name
: The name of the group this line defines.

Password
: The password for this group. Normally empty, and implying that no password is required for the group.

Group ID.
: The numeric ID of the given group.

Users List
: A list of user names that belong to this group. The names are separating using comma (',') characters.

Thus, the above line states that the "bin" group has a GID of '1', and has users "root", "bin" and "daemon" as members. Again, remember that there may be several lines for the group "bin" in this file, and they should all have the same GID. The

order of such lines is not important.

Finally, note that much like the "/etc/passwd" file, the "/etc/group" file is readable by anyone, and writable only by its owner - root. However, unlike for the "/etc/passwd" file, there are no standard programs that allow changing the contents of the "/etc/group" file.

Listing The Users In A Given Group

The first API we will deal with is used to get the list of users belonging to a given group. This is done using the `getgrnam()` or the `getgrgid()` functions.

The `getgrnam()` function gets a group name as its parameter, and returns a pointer to a `struct group` containing information about the given group. The function will return NULL if the group does not exist on the system. The group structure that is returned is defined as follows:

```
struct group {
    char    *gr_name;    /* group name */
    char    *gr_passwd;  /* group password */
    gid_t   gr_gid;      /* group id */
    char    **gr_mem;    /* group members */
};
```

Note that the `gr_mem` field is made of a list of user names, terminated by a NULL pointer. This group will contain all members of the group - no matter if they are defined on several separate lines. However, users associated with the group only via the GID field in their "/etc/passwd" entries (and not via proper lines in the "/etc/group" file) would NOT be included in this list. This makes this function somewhat broken.
Here is an example of a code that prints out the list of members in the "strange" group:

```
#include <grp.h> /* defines 'struct group', and getgrnam(). */
#include <sys/types.h> /* defines 'gid_t', etc.               */

/* get the information about the "strange" group. */
struct group* group_info = getgrnam("strange");
/* make sure this group actually exists. */
if (!group_info) {
    printf("group 'strange' does not exist.\n");
}
else {
    char** p_member;

    printf("Here are the members of group 'strange':\n");
    for (p_member = group_info->gr_mem; *p_member; p_member++)
        printf("  %s\n", *p_member);
}
```

The `getgrgid()` function works the same as `getgrnam()`, except that its parameter is the numeric GID of the desired group. hopefully you would be able to write your own example of using this function.

One last note about the `getgrnam()` and `getgrgid()` functions is that just like the `getpwnam()` function, they return pointers to statically allocated structures, and thus erase the results of the previous call on each consecutive call.

Enumerating (almost) All Groups

Similarly to the API used to enumerate user accounts, there are APIs used to enumerate "/etc/group" file entries. There are `getgrent()`, `setgrent()` and `endgrent()`. they work in exactly the same way as their counter APIs for the "/etc/group" file. Here is a quick example:

```
/* start the scan of the "/etc/group" file. */
struct group* group_info = getgrent();

/* loop over all group entries, until a NULL pointer is returned. */
printf("Here is a list of all groups found in '/etc/group':\n");
for ( ; group_info; group_info = getgrent() ) {
    printf(" %s\n", group_info->gr_name);
}
```

We won't demonstrate the usage of `setgrent()` and `endgrent()` - that should be obvious by now. If it is not, re-read the user accounts enumeration section.

Groups In /etc/group Vs. Groups In /etc/passwd

As it was mentioned before, groups may be defined in two places: by using the GID field of a user's entry in the "/etc/passwd" file, and by an entry in the "/etc/group" file. A group specified in the "/etc/group" file always has a group name, while a group specified in the "/etc/passwd" file does not have a name (unless there is a group with the same GID defined in the "/etc/group" file).

Another difference (already mentioned before) is that the `getgr*()` functions will show members specified in the "/etc/group" file, but will not show members specified via the "/etc/group" file.

Finally, we will refer the user to the `groups` command. This command will print the names of all groups the currently logged on user belongs to, or all groups a given user belongs to. This list of group will contain all relevant entries, whether found in the "/etc/group" file, or in the "/etc/passwd" file. Read the manual page for the "groups" command for more information.

Determining User Info At Run-Time

One of the things programs often need to know is which user they perform as, what groups they belong to, etc. For this, we should note that each process running on a Unix system, has a specific UID and a specific set of GIDs. These are usually the UID and GIDs of the user running the process, but there are exceptions.

Which User Is My Process Running As?

Determining the UID of a process is done using the `getuid()` system call, and a set of its cousins that will be discussed when we explain about Set UID programs below. This system call gets no parameters, and returns the effective UID of the process. Here is an example that prints out the name of the user executing this process:

```
/* Find the "/etc/passwd" entry of the UID of our process. */
struct passwd* user_info = getpwuid(getuid());

/* Print out the results. */
if (!user_info)
    printf("Error, cannot determine who i am.\n");
else
    printf("A-ha! I am %s\n", user_info->pw_name);
```

As the `getuid()` system call cannot fail, there is no need (and no way) to check its success.

Which Groups Is My Process Associated With?

Sometimes our program might want to know which groups its process (or rather, it's process's user) is part of. For example, consider a file manager that allows us to change the group owning a file. It will present us with a menu containing all groups

which we belong to. For this, the `getgroups()` system call may be used. This system call accepts two parameters: the maximal number of groups IDs we want to receive, and an array into which the GIDs will be stored by the system call. `getgroups()` returns the number of GIDs actually stored in the array. A special case is if we supply '0' in the first parameter. In this case, the array is unmodified, and the return value is the total number of groups associated with the process. Here is how we can receive all the groups in a single call:

```
#include <unistd.h>     /* getgroups(), etc. */

/* this array will be used to store the GIDs. */
gid_t** group_ids;
int num_groups;

/* first, check how many groups our process actually has. */
num_groups = getgroups(0, group_ids);
if (num_groups < 0) {
    perror("getgroups");
    exit(1);
}
if (num_groups > 0) {
    int i;

    group_ids = (gid_t**)malloc(num_groups * sizeof(gid_t));
    getgroups(num_groups, group_ids);
    printf("GIDs of our process:\n");
    for (i=0; i<num_groups; i++)
        printf("%d,", group_ids[i]);
    printf("\n");
}
```

Note that we called `getgroups()` twice - once to find the number of groups (so we could allocate a large enough array), and once to actually get the GIDs. We could then translate the GIDs into group names (using the `getgrgid()` function we saw earlier), and present that to the user (in our file manager menu's example).

The Sick Case Of Set-User-ID (suid) Programs

If you remember, we mentioned earlier that the "/etc/passwd" file is only writable for "root". Yet, there is a program named "passwd" that allows other users to change the contents of certain fields in this file. You might wonder how this is done. The idea is rather simple. If you check out the permissions of the "passwd" program, you will see something like this (using 'ls -l /usr/bin/passwd'):

```
-r-sr-xr-x   1 root     bin         15796 Apr 23  1997 /usr/bin/passwd
```

As you can see, instead of an 'x' for the owner, there is a lower-case 's'. This indicates that this program has the "set user-ID" (or SUID for short) bit - set. When the operating system launches a program that has the SUID bit set, it will give the process running it the identity of the user that owns the file - in this case, "root". This means that this process has the same access permissions as "root" has, rather then as the user that started this process. This way, when user "john" runs the "passwd" program, he may alter the "/etc/passwd" file, that he could not alter directly. It is up to the program to make sure that he only may change fields of his own account, and indeed, many security holes on Unix systems are due to bugs in SUID programs, that allow users that run them to perform operations they shouldn't be able to.

The method by which the operating system handles SUID programs, is by giving each process two identities: the real UID, and the effective UID. The real UID is the UID associated with the process that launched our process. The effective UID is the same as the real UID for normal programs, or it is the owner of the executable file, for SUID programs. These changes may also be done using the `setuid()` and `seteuid()` functions, under various restrictions. These system calls are used (for example) by the "login" program (that runs as user "root" initially), to spawn a shell whose real and effective UID are those of the user that logs in.

To find the real UID of a running process, we may use the `getuid()` system call. To find the effective UID - we use the

`geteuid()` system call. If you remember, when we demonstrated finding the user name of our process earlier, we used the getuid() system call. However, if we wanted to find what actions the process will be actually able to do, we should have checked the effective UID of the process - this is because we need to handle the case of SUID programs - where the effective UID and the real UID are different.

Note: on Unices derived from the BSD system, there is a third type of UID, called the stored UID, that further complicates the rules of setting a process's real and effective UID.

Who Is Currently Logged On?

One thing that might interest our program is - who is currently logged on to our system? For example, the finger program prints that information our to users. Messaging programs (write, talk) use this information to find if the person we try to send a message to is logged on the system or not.

For this purpose, the "login" program writes down information in a file named "utmp" (often residing in the "/var/log", "/var/adm" or "/var/run" directory). This file is a binary file, and each entry is a record of a fixed size. There is no standard API used to read the contents of this file, but there is a data structure, defined in the "utmp.h" include file, that gives us the exact format of a record. Here is how it is defined on a Linux system:

```
struct utmp {
    short ut_type;              /* type of login */
    pid_t ut_pid;               /* pid of process */
    char ut_line[UT_LINESIZE];  /* device name of tty - "/dev/" */
    char ut_id[2];              /* init id or abbrev. ttyname */
    time_t ut_time;             /* login time */
    char ut_user[UT_NAMESIZE];  /* user name */
    char ut_host[UT_HOSTSIZE];  /* host name for remote login */
    long ut_addr;               /* IP addr of remote host */
};
```

This structure may be defined differently on different operating systems, but it will usually contain at least the user name and the time of the login. For our purpose, the login name is all that matters. Here is a sample program that lists the names of the currently logged in users:

```
#include <stdio.h>     /* printf(), etc.    */
#include <utmp.h>      /* struct utmp, etc. */
#include <string.h>    /* strncpy(), etc.   */
#include <unistd.h>    /* open(), etc.      */
#include <fcntl.h>     /* O_RDONLY, etc.    */

/* full path for the UTMP file on our sample system. */
#define UTMP_PATH "/var/run/utmp"

/* buffer to read one utmp record. */
struct utmp utmp_entry;
/* buffer to store a user name. */
char user_name[UT_NAMESIZE+1];
/* file descriptor for the "utmp" file. */
int fd;

/* open the utmp file for reading. */
fd = open(UTMP_PATH, O_RDONLY);
if (fd < 0) {
    perror("open");
    exit(1);
}
```

```
printf("Currently logged-in users:\n");
/* start scanning the file, entry by entry. */
while (1) {
    int rc = read(fd, &utmp_entry, sizeof(utmp_entry));
    if (rc < 0) {
        perror("read");
        exit(1);
    }
    if (rc == 0) /* end of file - exit the reading loop. */
        break;

    /* This is Linux specific - only records whose ut_type field is    */
    /* USER_PROCESS, represent logged in users. the rest are temporary */
    /* records of various types.                                       */
    if (utmp_entry.ut_type != USER_PROCESS)
        continue;

    /* copy the user name field to our user name variable. */
    strncpy(user_name, utmp_entry.ut_name, UT_NAMESIZE);
    /* make sure this string is terminated with a null character. */
    user_name[UT_NAMESIZE] = '\0';

    printf("'%s',", user_name);
}

printf("\n");
close(fd);
```

The source code of this program is also available in the show-logged-users.c file.

A few notes are in place:

- The code above is not portable, as not all systems have the ut_type field in a utmp record.
- String fields (user name, host name) are terminated by a null character ('\0'), unless their contents are as large as the field's size. In such a case, we must add the missing null character before treating the value as a string.
- The code showed here is buggy. just think what happens if in the middle of scanning the file a new user is logged in. To avoid this, some file locking is necessary (e.g. using the `fcntl`). However, this should not be done for a long period of time, or else no user will be able to log in (the login program will get blocked when it tries to update the utmp file) - so try this only at home - not on a production system.
- The same user may have several records in the utmp file - one for each session the user currently has. If we wanted to only show the user names, we should remove these duplicates.
- On some systems, there might be a non-standard API for reading from (and possibly writing to) the the utmp file. It is better to use that API then to use this code, thought it will be even less portable.

Who Has Been Using My System Lately?

Much like the utmp file stores records of currently logged in users, the wtmp file stores historic records of login and logout operations. This can be used to find out when a user was last logged in, or how many hours they have spent on the system in the past week. The wtmp file contains records that are appended by the various login programs. Their format is exactly like the format of utmp record, with two exceptions:

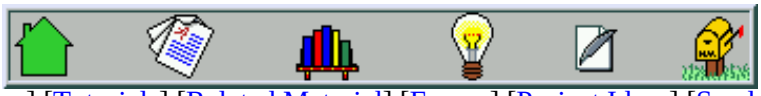
- A null user name in a record indicates a logout record. To find the matching login record, scan the wtmp file backwards, until the first record with the same device name field (ut_line);
- A record with the value '~' in its terminal name, and a value of "shutdown" or "reboot" in its user name field, indicates a shutdown operation or a boot operation, respectively. In such a case, there might be missing logout entries for users that were logged on during the shutdown, and programs that process the wtmp file should take this into account.
- Some other record types might exist - read the manual page about wtmp for info about how your system handles this

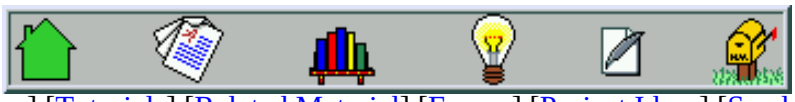
file.

We will leave it up to you to write a program that parses the wtmp file. Note that this file might grow to a very large size on busy systems, so such a program should be very efficient. An example of such a program is the "last" program, that shows you the list of logins to the system from the most recent - backwards.

References To Other System Databases

There are many other files on Unix systems that serve as databases. For example, the "/etc/shells" file defines the list of valid user shells, and is consulted by the "chsh" (change shell) program that allows a user to set their login shell. There are databases to hold mapping of host names into IP addresses ("/etc/hosts"), and so on. Some of these database files are standard on most Unix systems. Some are specific to classes of Unix systems. Often, a specific section of the on-line manual (section 4 or section 5, depending on the system type) documents the structure of these files. You might want to take a look at the files found in directory "/usr/man/man5" or "/usr/man/cat5" to get a clue of what database files exist on your system.





v1.0

Basic Graphics Programming With The Xlib Library

Preface

This tutorial is the first in a series of "would-be" tutorials about graphical programming in the X window environment. By itself, it is useless. A real X programmer usually uses a much higher level of abstraction, such as using Motif (or its free version, lesstiff), GTK, QT and similar libraries. However, we need to start somewhere. More then this, knowing how things work down below is never a bad idea.

After reading this tutorial, one would be able to write very simple graphical programs, but not programs with a descent user interface. For such programs, one of the previously mentioned libraries would be used.

The Client And Server Model Of The X Window System

The X window system was developed with one major goal - flexibility. The idea was that the way things look is one thing, but the way things work is another matter. Thus, the lower levels provide the tools required to draw windows, handle user input, allow drawing graphics using colors (or black and white screens), etc. To this point, a decision was made to separate the system into two parts. A client that decides what to do, and a server that actually draws on the screen and reads user input in order to send it to the client for processing.

This model is the complete opposite of what one is used to when dealing with clients and servers. In our case, the user seats near the machine controlled by the server, while the client might be running on a remote machine. The server controls the screen, mouse and keyboard. A client may connect to the server, request that it draws a window (or several windows), and ask the server to send it any input the user sends to these windows. Thus, several clients may connect to a single X server - one might be running an email software, one running a WWW browser, etc. When input it sent by the user to some window, the server sends a message to the client controlling this window for processing. The client decides what to do with this input, and sends the server requests for drawing in the window.

The whole session is carried out using the X message protocol. This protocol was originally carried over the TCP/IP protocol suite, allowing the client to run on any machine connected to the same network that the server is. Later on the X servers were extended to allow clients running on the local machine more optimized access to the server (note that an X protocol message may be several hundreds of KB in size), such as using shared memory, or using Unix domain sockets (a method for creating a logical channel on a Unix system between two processes).

GUI programming - the Asynchronous Programming Model

Unlike conventional computer programs, that carry some serial nature, a GUI program usually uses an asynchronous programming model, also known as "event-driven programming". This means that that program mostly sits idle, waiting for events sent by the X server, and then acts upon these events. An event may say "The user pressed the 1st button mouse in spot x,y", or "the window you control needs to be redrawn". In order for the program to be responsive to the user input, as well as to refresh requests, it needs to handle each event in a rather short period of time (e.g. less then 200 milliseconds, as a rule of thumb).

This also implies that the program may not perform operations that might take a long time while handling an event (such as opening a network connection to some remote server, or connecting to a database server, or even performing a long file copy operation). Instead, it needs to perform all these operations in an asynchronous manner. This may be done by using various asynchronous models to perform the longish operations, or by performing them in a different process or thread.

So the way a GUI program looks is something like that:

1. Perform initialization routines.
2. Connect to the X server.
3. Perform X-related initialization.
4. While not finished:
    1. Receive the next event from the X server.
    2. handle the event, possibly sending various drawing requests to the X server.
    3. If the event was a quit message, exit the loop.
5. Close down the connection to the X server.
6. Perform cleanup operations.

Basic Xlib Notions

In order to eliminate the needs of programs to actually implement the X protocol layer, a library called 'Xlib' was created. This library gives a program a very low-level access to any X server. Since the protocol is standardized, A client using any implementation of Xlib may talk with any X server. This might look trivial these days, but back at the days of using character mode terminals and proprietary methods of drawing graphics on screens, this looked like a major break-through. In fact, you'll notice the big hype going around thin-clients, windows terminal servers, etc. They are implementing today what the X protocol enabled in the late 80's. On the other hand, the X universe is playing a catch-up game regarding CUA (common user access, a notion made by IBM to refer to the usage of a common look and feel for all programs in order to ease the lives of the users). Not having a common look and feel was a philosophy of the creators of the X window system. Obviously, it had some drawbacks that are evident today.

The X Display

The major notion of using Xlib is the X display. This is a structure representing the connection we have open with a given X server. It hides a queue of messages coming from the server, and a queue of pending requests that our client intends to send to the server. In Xlib, this structure is named 'Display'. When we open a connection to an X server, the library returns a pointer to such a structure. Later, we supply this pointer to any Xlib function that should send messages to the X server or receive messages from this server.

The GC - Graphics Context

When we perform various drawing operations (graphics, text, etc), we may specify various options for controlling how the data will be drawn - what foreground and background colors to use, how line edges will be connected, what font to use when drawing some text, etc). In order to avoid the need to supply zillions of parameters to each drawing function, a graphical context structure, of type 'GC' is used. We set the various drawing options in this structure, and then pass a pointer to this structure to any drawing routines. This is rather handy, as we often needs to perform several drawing requests with the same options. Thus, we would initialize a graphical context, set the desired options, and pass this GC structure to all drawing functions.

Object Handles

When various objects are created for us by the X server - such as windows, drawing areas and cursors - the relevant function returns a handle. This is some identifier for the object that actually resides in the X server's memory - not in our application's memory. We can later manipulate this object by supplying this handle to various Xlib functions. The server keeps a mapping between these handles and the actual objects it manages. Xlib provides various type definitions for these objects (Window, Cursor, Colormap and so on), which are all eventually mapped to simple integers. We should still use these type names when defining variables that hold handles - for portability reasons.

Memory Allocation For Xlib Structures

Various structure types are used in Xlib's interface. Some of them are allocated directly by the user. Others are allocated using specific Xlib functions. This allows the library to initialize properly these structures. This is very handy, since these structures tend to contain a lot of variables, making it rather tedious for the poor programmer to initialize. Remember - Xlib tries to be as flexible as possible, and this means it is also as complex as it can get. Having default values will enable a beginner X programmer to use the library, without interfering with the ability of a more experienced programmer to tweak with these zillions of options.

As for freeing memory, this is done in one of two ways. In cases where we allocated the memory - we free it in the same manner (i.e. use `free()` to free memory allocated using `malloc()`). In case we used some Xlib function to allocate it, or we used some Xlib query method that returns dynamically allocated memory - we will use the `XFree()` function to free this memory block.

Events

A structure of type 'XEvent' is used to pass events received from the X server. Xlib supports a large amount of event types. The XEvent structure contains the type of event received, as well as the data associated with the event (e.g. position on the screen where the event was generated, mouse button associated with the event, region of screen associated with a 'redraw' event, etc). The way to read the event's data depends on the event type. Thus, an XEvent structure contains a C language union of all possible event types (if you're not sure what C unions are, it is time to check your proffered C language manual...). Thus, we could have an XExpose event, an XButton event, an XMotion event, etc.

Compiling Xlib-Based Programs

Compiling Xlib-Based programs requires linking them with the Xlib library. This is done using a compilation command like this:

```
cc prog.c -o prog -lX11
```

If the compiler complains that it cannot find the X11 library, try adding a '-L' flag, like this:

```
cc prog.c -o prog -L/usr/X11/lib -lX11
```

or perhaps this (for a system with release 6 of X11):

```
cc prog.c -o prog -L/usr/X11R6/lib -lX11
```

On SunOs 4 systems, the X libraries are placed in /usr/openwin/lib:

```
cc prog.c -o prog -L/usr/openwin/lib -lX11
```

and so on...

Opening And Closing The Connection To An X Server

An X program first needs to open the connection to the X server. When we do that, we need to specify the address of the host running the X server, as well as the display number. The X window system can support several displays all connected to the same machine. However, usually there is only one such display, which is display number '0'. If we wanted to connect to the local display (i.e. the display of the machine on which our client program runs), we could specify the display as ":0". To connect to the first display of a machine whose address is "simey", we could use the address "simey:0". Here is how the connection is opened:

```
#include <X11/Xlib.h>   /* defines common Xlib functions and structs. */
.
.
/* this variable will contain the pointer to the Display structure */
/* returned when opening a connection.                             */
Display* display;

/* open the connection to the display "simey:0". */
display = XOpenDisplay("simey:0");
if (display == NULL) {
    fprintf(stderr, "Cannot connect to X server %s\n", "simey:0");
    exit (-1);
}
```

Note that is common for X programs to check if the environment variable 'DISPLAY' is defined, and if it is, use its contents as the parameter to the `XOpenDisplay()` function.

When the program finished its business and needs to close the connection the X server, it does something like this:

```
XCloseDisplay(display);
```

This would cause all windows created by the program (if any are left) to be automatically closed by the server, and any resources stored on the server on behalf of the clients - to be freed. Note that this does not cause our client program to terminate - we could use the normal `exit()` function to do that.

Checking Basic Information About A Display

Once we opened a connection to an X server, we should check some basic information about it: what screens it has, what is the size (width and height) of the screen, how many colors it supports (black and white? grey scale? 256 colors? more?), and so on. We will show a code snippet that makes few of these checks, with comments explaining each function as it is being used. We assume that 'display' is a pointer to a 'Display' structure, as returned by a previous call to `XOpenDisplay()`.

```
/* this variable will be used to store the "default" screen of the  */
/* X server. usually an X server has only one screen, so we're only */
/* interested in that screen.                                       */
int screen_num;

/* these variables will store the size of the screen, in pixels.    */
int screen_width;
int screen_height;

/* this variable will be used to store the ID of the root window of our */
/* screen. Each screen always has a root window that covers the whole   */
/* screen, and always exists.                                           */
Window root_window;

/* these variables will be used to store the IDs of the black and white */
/* colors of the given screen. More on this will be explained later.    */
unsigned long white_pixel;
unsigned long black_pixel;

/* check the number of the default screen for our X server. */
screen_num = DefaultScreen(display);

/* find the width of the default screen of our X server, in pixels. */
screen_width = DisplayWidth(display, screen_num);

/* find the height of the default screen of our X server, in pixels. */
screen_height = DisplayHeight(display, screen_num);

/* find the ID of the root window of the screen. */
root_window = RootWindow(display, screen_num);

/* find the value of a white pixel on this screen. */
white_pixel = WhitePixel(display, screen_num);
```

```
/* find the value of a black pixel on this screen. */
black_pixel = BlackPixel(display, screen_num);
```

There are various other macros to get more information about the screen, that you can find in any Xlib reference. There are also function equivalents for some of these macros (e.g. XWhitePixel, which does the same as WhitePixel).

Creating A Basic Window - Our "hello world" Program

After we got some basic information about our screen, we can get to creating our first window. Xlib supplies several functions for creating new windows, one of which is `XCreateSimpleWindow()`. This function gets quite a few parameters determining the window's size, its position, and so on. Here is a complete list of these parameters:

`Display* display`
: Pointer to the Display structure.

`Window parent`
: The ID of an existing window that should be the parent of the new window.

`int x`
: X Position of the top-left corner of the window (given as number of pixels from the left of the screen).

`int y`
: Y Position of the top-left corner of the window (given as number of pixels from the top of the screen).

`unsigned int width`
: Width of the new window, in pixels.

`unsigned int height`
: Height of the new window, in pixels.

`unsigned int border_width`
: Width of the window's border, in pixels.

`unsigned long border`
: Color to be used to paint the window's border.

`unsigned long background`
: Color to be used to paint the window's background.

Lets create a simple window, whose width is 1/3 of the screen's width, height is 1/3 of the screen's height, background color is white, border color is black, and border width is 2 pixels. The window will be placed at the top-left corner of the screen.

```
/* this variable will store the ID of the newly created window. */
Window win;

/* these variables will store the window's width and height. */
int win_width;
int win_height;

/* these variables will store the window's location. */
int win_x;
int win_y;

/* calculate the window's width and height. */
win_width = DisplayWidth(display, screen_num) / 3;
win_height = DisplayHeight(display, screen_num) / 3;

/* position of the window is top-left corner - 0,0. */
```

```
win_x = win_y = 0;

/* create the window, as specified earlier. */
win = XCreateSimpleWindow(display,
                          RootWindow(display, screen_num),
                          win_x, win_y,
                          win_width, win_height,
                          win_border_width, BlackPixel(display, screen_num),
                          WhitePixel(display, screen_num));
```

The fact that we created the window does not mean it will be drawn on screen. By default, newly created windows are not mapped on the screen - they are invisible. In order to make our window visible, we use the `XMapWindow()` function, as follows:

```
XMapWindow(win);
```

To see all the code we have gathered so far, take a look at the simple-window.c program. You'll see two more function not explained so far - `XFlush()` and `XSync()`. The `XFlush()` function flushes all pending requests to the X server - much like the `fflush()` function is used to flash standard output. The `XSync()` function also flushes all pending requests to the X server, and then waits until the X server finishes processing these requests. In a normal program this will not be necessary (you'll see why when we get to write a normal X program), but for now we put it there. Try compiling the program either with or without these function calls to see the difference in its behavior.

Drawing In A Window

Drawing in a window can be done using various graphical functions - drawing pixels, lines, circles, rectangles, etc. In order to draw in a window, we first need to define various general drawing parameters - what line width to use, which color to draw with, etc. This is done using a graphical context (GC).

Allocating A Graphics Context (GC)

As we said, a graphical context defines several attributes to be used with the various drawing functions. For this, we define a graphical context. We can use more then one graphical context with a single window, in order to draw in multiple styles (different colors, different line widths, etc.). Allocating a new GC is done using the `XCreateGC()` function, as follows (in this code fragment, we assume "display" is a pointer to a Display structure, and "win" is the ID of a previously created window):

```
/* this variable will contain the handle to the returned graphics context. */
GC gc;

/* these variables are used to specify various attributes for the GC. */
/* initial values for the GC. */
XGCValues values = CapButt | JoinBevel;
/* which values in 'values' to check when creating the GC. */
unsigned long valuemask = GCCapStyle | GCJoinStyle;

/* create a new graphical context. */
gc = XCreateGC(display, win, valuemask, &values);
if (gc < 0) {
    fprintf(stderr, "XCreateGC: \n");
}
```

Note should be taken regarding the roles of "valuemask" and "values". Since a graphics context has zillions of

attributes, and since often we don't want to define few of them, we need to be able to tell the `XCreateGC()` which attributes we want to set. This is what the "valuemask" variable is for. We then use the "values" variable to specify actual values for the attributes we defined in the "valuesmask". Thus, for each constant used in "values", we'll use the matching constant in "valuesmask". In this case, we defined a graphics context with two attributes:

1. When drawing a multiple-part line, the lines should be joined in a 'Bevelian' style.
2. A line's end-point will be drawn straight (as opposed to ending the line in a round shape, if its width is more then 1 pixel wide).

The rest of the attributes of this GC will be set to their default values.

Once we created a graphics context, we can use it in drawing functions. We can also modify its parameters using various functions. Here are a few examples:

```
/* change the foreground color of this GC to white. */
XSetForeground(display, gc, WhitePixel(display, screen_num));

/* change the background color of this GC to black. */
XSetBackground(display, gc, BlackPixel(display, screen_num));

/* change the fill style of this GC to 'solid'. */
XSetFillStyle(display, gc, FillSolid);

/* change the line drawing attributes of this GC to the given values. */
/* the parameters are: Display structure, GC, line width (in pixels), */
/* line drawing  style, cap (line's end) drawing style, and lines     */
/* join style.                                                        */
XSetLineAttributes(display, gc, 2, LineSolid, CapRound, JoinRound);
```

for complete information on the various attributes available in a graphics context, refer to the manual page of `XCreateGC()`. We will use just a few simple attributes in our tutorial, to avoid over-complicating it.

Drawing Primitives - Point, Line, Box, Circle...

After we have created a `GC`, we can draw on a window using this GC, with a set of Xlib functions, collectively called "drawing primitives". Without much fuss, lets see how they are used. We assume that "gc" is a previously initialized `GC`, and that 'win' contains the handle of a previously created window.

```
/* draw a pixel at position '5,60' (line 5, column 60) of the given window. */
XDrawPoint(display, win, gc, 5, 5);

/* draw a line between point '20,20' and point '40,100' of the window. */
XDrawLine(display, win, gc, 20, 20, 40, 100);

/* draw an arc whose center is at position 'x,y', its width (if it was a     */
/* full ellipse) is 'w', and height is 'h'. Start the arc at angle 'angle1'  */
/* (angle 0 is the hour '3' on a clock, and positive numbers go              */
/* counter-clockwise. the angles are in units of 1/64 of a degree (so 360*64 */
/* is 360 degrees).                                                          */
int x = 30, y = 40;
int h = 15, w = 45;
int angle1 = 0, angle2 = 2.109;
XDrawArc(display, win, gc, x-(w/2), y-(h/2), w, h, angle1, angle2);

/* now use the XDrawArc() function to draw a circle whose diameter */
```

```
/* is 15 pixels, and whose center is at location '50,100'.        */
XDrawArc(display, win, gc, 50-(15/2), 100-(15/2), 15, 15, 0, 360*64);

/* the XDrawLines() function draws a set of consecutive lines, whose    */
/* edges are given in an array of XPoint structures.                    */
/* The following block will draw a triangle. We use a block here, since */
/* the C language allows defining new variables only in the beginning of */
/* a block.                                                             */
  {
    /* this array contains the pixels to be used as the line's end-points. */
    XPoint points[] = {
      {0, 0},
      {15, 15},
      {0, 15},
      {0, 0}
    };
    /* and this is the number of pixels in the array. The number of drawn */
    /* lines will be 'npoints - 1'.                                       */
    int npoints = sizeof(points)/sizeof(XPoint);

    /* draw a small triangle at the top-left corner of the window. */
    /* the triangle is made of a set of consecutive lines, whose   */
    /* end-point pixels are specified in the 'points' array.       */
    XDrawLines(display, win, gc, points, npoints, CoordModeOrigin);
  }

/* draw a rectangle whose top-left corner is at '120,150', its width is */
/* 50 pixels, and height is 60 pixels.                                  */
XDrawRectangle(display, win, gc, 120, 150, 50, 60);

/* draw a filled rectangle of the same size as above, to the left of the  */
/* previous rectangle. note that this rectangle is one pixel smaller then */
/* the previous line, since 'XFillRectangle()' assumes it is filling up   */
/* an already drawn rectangle. This may be used to draw a rectangle using */
/* one color, and later to fill it using another color.                   */
XFillRectangle(display, win, gc, 60, 150, 50, 60);
```

Hopefully, you got the point by now. We will mention a few more functions that may be used in a similar fashion. For example, `XFillArc()` takes the same parameters as `XDrawArc()`, but draws only the inside of this arc (like `XFillRectangle()` does to a rectangle drawn using the `XDrawRectangle()` function). There is also an `XFillPolygon()` function that fills the inside of a polygon. It takes almost the same parameters as `XDrawLines()`. However, if the last point in the array has a different location then the first point in the array, the `XFillPolygon()` function automatically adds another "virtual" lines, connecting these two points. Another difference between the two functions, is that `XFillPolygon()` takes an additional parameters, shape, that is used to help the X server optimize its operation. You can read about it in your manual pages. There are also plural versions for these functions, namely `XFillArcs()` and `XFillRectangles()`.

The source code for a program doing these drawings is found in the file simple-drawing.c.

X Events

In an Xlib program, everything is driven by events. Event painting on the screen is sometimes done as a response to an event - an "expose" event. If part of a program's window that was hidden, gets exposed (e.g. the window was raised above other windows), the X server will send an "expose" event to let the program know it should repaint that part of the window. User input (key presses, mouse movement, etc.) is also received as a set of events.

Registering For Event Types Using Event Masks

After a program creates a window (or several windows), it should tell the X server what types of events it wishes to receive for this window. By default, no events are sent to the program. It may register for various mouse (also called "pointer") events, keyboard events, expose events and so on. This is done for optimizing the server-to-client connection (i.e. why send a program (that might even be running at the other side of the globe) an event it is not interested in?).

In Xlib, we use the `XSelectInput()` function to register for events. This function accepts 3 parameters - the display structure, an ID of a window, and a mask of the event types it wishes to get. The window ID parameter allows us to register for receiving different types of events for different windows. Here is how we register for "expose" events for a window whose ID is 'win':

```
XSelectInput(display, win, ExposureMask);
```

`ExposureMask` is a constant defined in the "X.h" header file. If we wanted to register to several event types, we can logically "or" them, as follows:

```
XSelectInput(display, win, ExposureMask | ButtonPressMask);
```

This registers for "expose" events as well as for mouse button presses inside the given window. You should note that a mask may represent several event sub-types.

Note: A common bug programmers do is adding code to handle new event types in their program, while forgetting to add the masks for these events in the call to `XSelectInput()`. Such a programmer then could sit down for hours debugging his program, wondering "why doesn't my program notice that i released the button??", only to find that they registered for button press events, but not for button release events.

Receiving Events - Writing The Events Loop

After we have registered for the event types we are interested in, we need to enter a loop of receiving events and handling them. There are various ways to write such a loop, but the basic loop looks like this:

```
/* this structure will contain the event's data, once received. */
XEvent an_event;

/* enter an "endless" loop of handling events. */
while (1) {
    XNextEvent(display, &an_event);
    switch (an_event.type) {
      case Expose:
        /* handle this event type... */
        .
        .
        break;
      default: /* unknown event type - ignore it. */
        break;
    }
```

```
}
```

The `XNextEvent()` function fetches the next event coming from the X server. If no event is waiting, it blocks until one is received. When it returns, the event's data is placed in the `XEvent` variable given to the function as the second parameter. After that, the "type" field of this variable specifies what type of event we got. `Expose` is the event type that tells us there is a part of the window that needs to be redrawn. After we handle this event, we go back and wait for the next event to process. Obviously, we will need to give the user some way of terminating the program. This is usually done by handling a special "quit" event, as we'll soon see.

Expose Events

The "expose" event is one of the most basic events an application may receive. It will be sent to us in one of several cases:

- A window that covered part of our window has moved away, exposing part (or all) of our window.
- Our window was raised above other windows.
- Our window mapped for the first time.
- Our window was de-iconified.

You should note the implicit assumption hidden here - the contents of our window is lost when it is being obscured (covered) by other windows. One may wonder why the X server does not save this contents. The answer is - to save memory. After all, the number of windows on a display at a given time may be very large, and storing the contents of all of them might require a lot of memory (for instance, a 256 color bitmap covering 400 pixels by 400 pixels takes 160KB of memory to store. Now think about 20 windows, some much larger then this size). Actually, there is a way to tell the X server to store the contents of a window in special cases, as we will see later.

When we get an "expose" event, we should take the event's data from the "xexpose" member of the `XEvent` structure (in our code example we refer to it as "an_event.xexpose"). It contains several interesting fields:

`count`
: Number of other expose events waiting in the server's events queue. This may be useful if we got several expose events in a row - we will usually avoid redrawing the window until we get the last of them (i.e. until count is 0).

`Window window`
: The ID of the window this expose event was sent for (in case our application registered for events on several windows).

`int x, y`
: The x and y coordinates (in pixels) from the top-left of the window, of the window's region that needs to be redrawn.

`int width, height`
: The width and height (in pixels) of the window's region that needs to be redraw.

In our demo programs, we will tend to ignore the region supplied, and simply re-draw all the window. However, this is very inefficient, and we will try to demonstrate some techniques for drawing only the relevant section of screen later on.

As an example, here is how we will draw a line across our window, whenever we receive "expose" events. Assume this 'case' is part of the event loop's `switch` command.

```
  case Expose:
    /* if we have several other expose events waiting, don't redraw. */
    /* we will do the redrawing when we receive the last of them.    */
    if (an_event.xexpose.count > 0)
        break;
    /* ok, now draw the line... */
    XDrawLine(display, win, gc, 0, 100, 400, 100);
    break;
```

Getting User Input

User input traditionally comes from two sources - the mouse and the keyboard. Various event types exist to notify us of user input - a key being pressed on the keyboard, a key being released on the keyboard, the mouse moving over our window, the mouse entering (or leaving) our window and so on.

Mouse Button Click And Release Events

The first event type we'll deal with is a mouse button-press (or button release) event in our window. In order to register to such an event type, we would add one (or more) of the following masks to the event types we specify for the `XSelectInput()` function:

`ButtonPressMask`
: Notify us of any button that was pressed in one of our windows.

`ButtonReleaseMask`
: Notify us of any button that was released over one of our windows.

The event types to be checked for in our event-loop switch, are any of the following:

`ButtonPress`
: A button was pressed over one of our windows.

`ButtonRelease`
: A button was released over one of our windows.

The event structure for these event types is accessed as "an_event.xbutton", and contains the following interesting fields:

`Window window`
: The ID of the window this button event was sent for (in case our application registered for events on several windows).

`int x, y`
: The x and y coordinates (in pixels) from the top-left of the window, of the mouse pointer, during the click.

`int button`
: The number of mouse button that was clicked. May be a value such as `Button1`, `Button2`, `Button3`.

`Time time`
: time (in millisecond) the event took place in. May be used to calculate "double-click" situations by an application (e.g. if the mouse button was clicked two times in a duration shorter then a given amount, assume this was a double-click).

As an example, here is how we will draw a black pixel at the mouse position, whenever we receive "button press" events, with the 1st mouse button, and erase that pixel (i.e. draw a white pixel) when the 2nd mouse button is pressed. We assume the existence of two GCs, gc_draw with foreground color set to black, and gc_erase, with foreground color set to white.
Assume that the following 'case' is part of the event loop's `switch` command.

```
  case ButtonPress:
    /* store the mouse button coordinates in 'int' variables. */
    /* also store the ID of the window on which the mouse was */
    /* pressed.                                               */
    x = an_event.xbutton.x;
    y = an_event.xbutton.y;
    the_win = an_event.xbutton.window;
```

```
    /* check which mouse button was pressed, and act accordingly. */
    switch (an_event.xbutton.button) {
        case Button1:
            /* draw a pixel at the mouse position. */
            XDrawPoint(display, the_win, gc_draw, x, y);
            break;
        case Button2:
            /* erase a pixel at the mouse position. */
            XDrawPoint(display, the_win, gc_erase, x, y);
            break;
        default: /* probably 3rd button - just ignore this event. */
            break;
    }
    break;
```

Mouse Movement Events

Similar to mouse button press and release events, we also can be notified of various mouse movement events. These can be split into two families. One is of mouse pointer movement while no buttons are pressed, and the second is a mouse pointer motion while one (or more) of the buttons are pressed (this is sometimes called "a mouse drag operation", or just "dragging"). The following event masks may be added in the call to `XSelectInput()` for our application to be notified of such events:

`PointerMotionMask`
: Events of the pointer moving in one of the windows controlled by our application, while no mouse button is held pressed.

`ButtonMotionMask`
: Events of the pointer moving while one (or more) of the mouse buttons is held pressed.

`Button1MotionMask`
: Same as `ButtonMotionMask`, but only when the 1st mouse button is held pressed.

`Button2MotionMask`, `Button3MotionMask`, `Button4MotionMask`, `Button5MotionMask`
: Likewise, for 2nd mouse button, or 3rd, 4th or 5th.

The event types to be checked for in our event-loop switch, are any of the following:

`MotionNotify`
: The mouse pointer moved in one of the windows for which we requested to be notified of such events.

The event structure for these event types is accessed as "an_event.xbutton", and contains the following interesting fields:

`Window window`
: The ID of the window this mouse motion event was sent for (in case our application registered for events on several windows).

`int x, y`
: The x and y coordinates (in pixels) from the top-left of the window, of the mouse pointer, when the event was generated.

`unsigned int state`
: A mask of the buttons (or keys) held down during this event - if any. This field is a bitwise OR of any of the following:
    - `Button1Mask`
    - `Button2Mask`

- Button3Mask
- Button4Mask
- Button5Mask
- ShiftMask
- LockMask
- ControlMask
- Mod1Mask
- Mod2Mask
- Mod3Mask
- Mod4Mask
- Mod5Mask

Their names are self explanatory, where the first 5 refer to mouse buttons that are being pressed, while the rest refer to various "special keys" that are being pressed (Mod1 is usually the 'ALT' key or the 'META' key).

`Time time`

time (in millisecond) the event took place in.

As an example, the following code handles a "draw mode" for a painting program, that is, if the user moves the mouse while the 1st mouse button is being held down, then we 'draw' on the screen. Note that this code has a flow: Since mouse movement may generate many events, it might be that we won't get a mouse motion event for each pixel the mouse moved over. Our program should be able to cope with such a situation. One way to do that would be to remember the last pixel the mouse was dragged over, and draw a line between that position and the new mouse pointer position. Assume that the following 'case' is part of the event loop's `switch` command.

```
  case MotionNotify:
    /* store the mouse button coordinates in 'int' variables. */
    /* also store the ID of the window on which the mouse was */
    /* pressed.                                               */
    x = an_event.xmotion.x;
    y = an_event.xmotion.y;
    the_win = an_event.xbutton.window;

    /* if the 1st mouse button was held during this event, draw a pixel */
    /* at the mouse pointer location.                                   */
    if (an_event.xmotion.state & Button1Mask) {
        /* draw a pixel at the mouse position. */
        XDrawPoint(display, the_win, gc_draw, x, y);
    }
    break;
```

Mouse Pointer Enter And Leave Events

Another type of event that applications might be interested at, is a mouse pointer entering a window the program controls, or leaving such a window. Some programs use these events to show the user that the application is now in focus. In order to register for such an event type, we would add one (or more) of the following masks to the event types we specify for the `XSelectInput()` function:

`EnterWindowMask`

Notify us when the mouse pointer enters any of our controlled windows.

`LeaveWindowMask`

Notify us when the mouse pointer leaves any of our controlled windows.

The event types to be checked for in our event-loop switch, are any of the following:

`EnterNotify`
: The mouse pointer just entered one of our controlled windows.

`LeaveNotify`
: The mouse pointer just left one of our controlled windows.

The event structure for these event types is accessed as "an_event.xcrossing", and contains the following interesting fields:

`Window window`
: The ID of the window this button event was sent for (in case our application registered for events on several windows).

`Window subwindow`
: The ID of the child window from which the mouse entered our window (in an EnterNotify event), or into which the mouse pointer has moved (in a LeaveNotify event), or None, if the mouse moved from outside our window.

`int x, y`
: The x and y coordinates (in pixels) from the top-left of the window, of the mouse pointer, when the event was generated.

`int mode`
: The number of mouse button that was clicked. May be a value such as `Button1`, `Button2`, `Button3`.

`Time time`
: time (in millisecond) the event took place in. May be used to calculate "double-click" situations by an application (e.g. if the mouse button was clicked two times in a duration shorter then a given amount, assume this was a double-click).

`unsigned int state`
: A mask of the buttons (or keys) held down during this event - if any. This field is a bitwise OR of any of the following:
  - `Button1Mask`
  - `Button2Mask`
  - `Button3Mask`
  - `Button4Mask`
  - `Button5Mask`
  - `ShiftMask`
  - `LockMask`
  - `ControlMask`
  - `Mod1Mask`
  - `Mod2Mask`
  - `Mod3Mask`
  - `Mod4Mask`
  - `Mod5Mask`

  Their names are self explanatory, where the first 5 refer to mouse buttons that are being pressed, while the rest refer to various "special keys" that are being pressed (Mod1 is usually the 'ALT' key or the 'META' key).

`Bool focus`
: Set to `True` if the window has the keyboard focus, `False` otherwise.

The Keyboard Focus

There may be many windows on a screen, but only a single keyboard attached to them. How does the X server then

know which window should be sent a given keyboard input? This is done using the keyboard focus. Only a single window on the screen may have the keyboard focus at a given time. There are Xlib functions that allow a program to set the keyboard focus to a given window. The user can usually set the keyboard focus using the window manager (often by clicking on the title bar of the desired window). Once our window has the keyboard focus, every key press or key release will cause an event to be sent to our program (if it registered for these event types...).

Keyboard Press And Release Events

If a window controlled by our program currently holds the keyboard focus, it can receive key press and key release events. In order to register for such events, any of the following masks may be added to the call to `XSelectInput()`:

`KeyPressMask`

Notify our program when a key was pressed while any of its controlled windows had the keyboard focus.

`KeyPressMask`

Notify our program when a key was released while any of its controlled windows had the keyboard focus.

The event types to be checked for in our event-loop switch, are any of the following:

`KeyPress`

A key was just pressed on the keyboard while any of our windows had the keyboard focus.

`KeyRelease`

A key was just released on the keyboard while any of our windows had the keyboard focus.

The event structure for these event types is accessed as "an_event.xkey", and contains the following interesting fields:

`Window window`

The ID of the window this button event was sent for (in case our application registered for events on several windows).

`unsigned int keycode`

The code of the key that was pressed (or released). This is some internal X code, that should be translated into a key symbol, as will be explained below.

`int x, y`

The x and y coordinates (in pixels) from the top-left of the window, of the mouse pointer, when the event was generated.

`Time time`

time (in millisecond) the event took place in. May be used to calculate "double-click" situations by an application (e.g. if the mouse button was clicked two times in a duration shorter then a given amount, assume this was a double-click).

`unsigned int state`

A mask of the buttons (or modifier keys) held down during this event - if any. This field is a bitwise OR of any of the following:

- `Button1Mask`
- `Button2Mask`
- `Button3Mask`
- `Button4Mask`
- `Button5Mask`
- `ShiftMask`
- `LockMask`
- `ControlMask`
- `Mod1Mask`
- `Mod2Mask`

- Mod3Mask
- Mod4Mask
- Mod5Mask

Their names are self explanatory, where the first 5 refer to mouse buttons that are being pressed, while the rest refer to various "special keys" that are being pressed (Mod1 is usually the 'ALT' key or the 'META' key).

As we mentioned, the key code is rather meaningless on its own, and is affected by the specific keyboard device attached to the machine running the X server. To actually use this code, we translate it into a key symbol, which is standardized. We may use the `XKeycodeToKeysym()` function to do the translation. This function gets 3 parameters: a pointer to the display, the key code to be translated, and an index (we'll supply '0' for this parameter). Standard Xlib key codes are found in the include file "X11/keysymdef.h". As an example for using the key press events together with the `XKeycodeToKeysym` function, we'll show how to handle key presses of this sort: Pressing '1' will cause painting the pixel where the mouse pointer is currently located. Pressing the DEL key will cause to erase that pixel (using a 'gc_erase' GC). Pressing any of the letters (a to z, upper case or lower case) will cause it to be printed to standard output. Any other key pressed will be ignored. Assume that the following 'case' is part of the event loop's `switch` command.

```
case KeyPress:
  /* store the mouse button coordinates in 'int' variables. */
  /* also store the ID of the window on which the mouse was */
  /* pressed.                                               */
  x = an_event.xkey.x;
  y = an_event.xkey.y;
  the_win = an_event.xkey.window;
  {
      /* translate the key code to a key symbol. */
      KeySym key_symbol = XKeycodeToKeysym(display, an_event.xkey.keycode, 0);
      switch (key_symbol) {
          case XK_1:
          case XK_KP_1: /* '1' key was pressed, either the normal '1', or */
                        /* the '1' on the keypad. draw the current pixel. */
              XDrawPoint(display, the_win, gc_draw, x, y);
              break;
          case XK_Delete: /* DEL key was pressed, erase the current pixel. */
              XDrawPoint(display, the_win, gc_erase, x, y);
              break;
          default:  /* anything else - check if it is a letter key */
              if (key_symbol >= XK_A && key_symbol <= XK_Z) {
                  int ascii_key = key_symbol - XK_A + 'A';
                  printf("Key pressed - '%c'\n", ascii_key);
              }
              if (key_symbol >= XK_a && key_symbol <= XK_z) {
                  int ascii_key = key_symbol - XK_a + 'a';
                  printf("Key pressed - '%c'\n", ascii_key);
              }
              break;
      }
  }
  break;
```

As you can see, key symbols refer to the physical key on the keyboard in some manner, so one should be careful to properly check all possible cases (as we did for the '1' key in the above example). We also assume that the letter keys have consecutive key symbol values, or else the range checking tricks and the key symbol to ASCII code translations wouldn't have worked.

X Events - A Complete Example

As an example for handling events, we will show the events.c program. This program creates a window, makes some drawings on it, and then enters an event loop. If it gets an expose event - it redraws the whole window. If it gets a left button press (or motion) event, it draws the pixel under the mouse pointer in black color. If it gets a middle button press (or motion) event, it draws the pixel under the mouse pointer in white color (i.e. erases this pixel). Some note should be taken as to how these picture changes are handled. It is not sufficient to just draw the pixel with the appropriate color. We need to make a note of this color change, so on the next expose event we will draw the pixel again with the proper color. For this purpose, we use a huge (1000 by 1000) array representing the window's pixels. Initially, all cells in the array are initialized to '0'. When drawing a pixel in black, we set the matching array cell to '1'. When drawing a pixel in white, we set the matching array cell to '-1'. we cannot just reset it back to '0', otherwise the original drawing we painted on the screen will always be erased. Finally, when the user presses on any on the keyboard, the program exits.

When running this program, you will note that motion events often skip pixels. If the mouse was moved from one point to another in a quick motion, we will not get motion events for each pixel the mouse pointer moved over. Thus, if it was our intention to handle these gaps properly, we would need to remember the location of the last motion event, and then perhaps draw a line between that location and the location of the next motion event. This is what painting programs normally do.

Handling Text And Fonts

Besides drawing graphics on a window, we often want to draw text. Text strings have two major properties - the characters to be drawn, and the font with which they are drawn. In order to draw text we need to first request the X server to load a font. We then assign the font to a GC, and finally we draw the text in a window, using the GC.

The Font Structure

In order to support flexible fonts, a font structure is defined, of type `XFontStruct`. This structure is used to contain information about a font, and is passed to several functions that handle fonts selection and text drawing.

Loading A Font

As a first step to draw text, we use a font loading function, such as `XLoadQueryFont()`. This function asks the X server to load data that defines a font with a given name. If the font is found, it is loaded by the server, and an `XFontStruct` pointer is returned. If the font is not found, or the loading operation fails, a NULL value is returned. Each font may have two names. One is a long string, that specifies the full properties of the font (font family, font size, italic/bold/underline attributes, etc.). The other is a short nickname for the font, configured for the specific X server. As an example, we will try to load a "*-helvetica-*-12-*" font (the '*' characters work the same as wildcard characters in a shell):

```
/* this pointer will hold the returned font structure. */
XFontStruct* font_info;

/* try to load the given font. */
char* font_name = "*-helvetica-*-12-*";
font_info = XLoadQueryFont(display, fond_name);
if (!font_info) {
    fprintf(stderr, "XLoadQueryFont: failed loading font '%s'\n", font_name);
}
```

Assigning A Font To A Graphics Context

After we load the font, we need to assign it to a GC. assuming that 'gc' is a variable of type GC that's already initialized, here is how this is done:

```
XSetFont(display, gc, font_info->fid);
```

The 'fid' field in an `XFontStruct` is an identified used to identify the loaded font in various requests.

Drawing Text In A Window

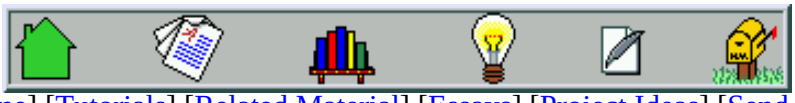
Once we got our wanted font loaded and assigned to our GC, we can draw text in our window, using a function such as `XDrawString()`. This function will draw a given text string at a given location on the window. The location would be that of the lower-left corner of the drawn text string. Here is how it is used:

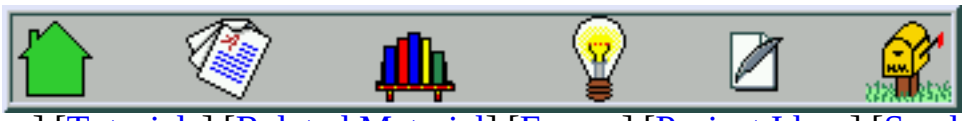
```
/* assume that win_width contains the width of our window, win_height        */
/* contains the height of our window, and 'win' is the handle of our window. */

/* some temporary variables used for the drawing. */
int x, y;

/* draw a "hello world" string on the top-left side of our window. */
x = 0;
y = 0;
XDrawString(display, win, gc, x, y, "hello world", strlen("hello world"));

/* draw a "middle of the road" string in the middle of our window. */
char* text_string = "middle of the road";
/* find the width, in pixels, of the text that will be drawn using */
/* the given font.                                                 */
int string_width = XTextWidth(font_info, text_string, strlen(text_string));
/* find the height of the characters drawn using this font.        */
int fond_height = font_info->ascent + font_info->descent;
x = (win_width - string_width) / 2;
y = (win_width - font_height) / 2;
XDrawString(display, win, gc, x, y, "hello world", strlen("hello world"));
```

Some notes must be made in order to make this code snippet clear:

- The `XTextWidth()` function is used to "predict" the width of a given text string, as it will be drawn using a given font. This may be used to determine where to draw the left end of the string so it will look like it's occupying the middle of the window, for example.
- A font has two attributes named "ascent" and "descent", used to specify the height of the font. Basically, a font's characters are drawn relative to some imaginary horizontal line. part of a character is drawn above this line, and part of it is drawn below this line. the highest letter is drawn at most "font_info->ascent" pixels above this line, while the letter with the lowest part will be drawn at most "font_info->descent" pixels below that line. Thus, the sum of these two numbers determines the height of the font.

The source code for a program drawing these texts is found in the file simple-text.c.





Parallel Programming - Basic Theory For The Unwary

Preface - Why Should You Care?

Writing sequential programs is the what most, if not all, programmers are being initially trained to do. At a certain phase in their life they discover that there are other models for programs. One of them is using parallelism. This means that instead of having your program carried out in a sequence, one instruction at a time, it is being executed by several different entities simultaneously. This can sometimes make the programs simpler to design, and may also run faster then a matching sequential program.

For example, if you have a computer with several processors, they might each be running a part of the program simultaneously, and thus complete it faster then if only one processor would have to run the whole program.

Another example is if you have a program that needs to read data from a disk, and meanwhile make some heavy calculations on it. Since the disk can transfer data to memory without intervention of the CPU, it would make sense to split your program into two parts running in parallel: one handles I/O, and reads data into memory. The other does the heavy calculations.

You could do it all in one sequential part, that majorly does the calculations, and from time to time goes to read another block of data, but it is harder to write it this way, or to be efficient (how will the program know when the last block of data was read, and it is time to read another block?)

Scope Of This Tutorial

This document attempts to illustrate the terms and principles commonly used in parallel systems. It is by no means a replacement for a good parallel programming course. Yet, it may make it easier for people without this background able to read and understand the various parallel programming tutorials, and start writing parallel programs. I'd still advise that they eventually take a course or two about parallel and/or distributed programming. I'd like to hear from you if this tutorial really achieved its purpose for you, if it was too theoretical, too short, or was flawed in a different way.

Part of this tutorial concentrates on the underlying hardware and operating system software used in parallel systems. The last section tries to give a few in-sites as to when to try to make a system parallelic, how to design it properly, and what kind of tools and frameworks one could expect to find that can ease such development.

Entities Of Parallel Systems

A parallel system is a system (software and/or hardware) that allows one to write programs whose different parts are carried out in different threads of execution.
In order to better understand what a parallel (or parallelic) system is, we should check what are the different components such a system is made of.

Processes

A process is an entity that executes a computer program, and manipulates all other resources in order to fulfill the mission of the program. Several (or many) processes may be running on the same computer, or on different computers. Usually a single process is running on a single computer. Also, usually each process has its own address space, as well as a few other private resources (an private execution stack, for example).

Note - what we call here 'process' is a broader entity then what you might know as a Unix process. We should have actually called it a 'runnable', and in practice it could be a Unix process, a Unix thread and so on.

Resources

A resource is an entity that can be used by a process to perform some specific operation. There could be physical resources, as well as logical (or virtual) resources.
A physical resource could be, for example, a disk, which is used for saving files. Or a keyboard, which is used to read data from the user.
A logical resource could be a shared memory area, which several processes may access in order to read data to, or read data from.

Executers

An executer is a special resource that is used to execute a process. Usually, this is a CPU, so we'll use the term CPU from now on. Note that not all CPUs are equal - some are general-purpose, and might carry out the program as a whole. Others are specialized for different tasks - a CPU that only does mathematical floating point operations; A CPU that only handles graphical operations (commonly found on graphical screen cards, and serving as a graphic accelerator) and so on.

Schedulers

A scheduler is an entity specifying when processes will run, on which CPUs (executors) they will run, and in what order. Such a scheduler may be controlling a single computer, or several computers (see clustering below).

Synchronizers

A synchronizer is an entity used to control the access of processes to resources. Some synchronizers are used to make sure only a single process uses a given resource at any one time. Other synchronizers are used to make sure a given resource is accessed in a given order of priority by different processes. There is a host of other types of synchronizers to be dealt with. You will most likely bump into synchronizers such as Mutexes, Semaphores, Monitors, Conditions, etc.

Short Terms Dictionary

There are a few terms used in conjunction with all types of parallel systems. We will describe them here, divided into several categories in some sort of a (hopefully) logical order. For each term, we'll try to give a real-life example making it easier to grasp the concept, and to remember it (I'd call this kind of example a "mantra". Forgive me for abusing this word).

Accessing Resources

Mutual Exclusion

A mechanism used to make sure no two processes are trying to use a resource at the same time. Used to avoid corrupting the internal state state of this resource. (mantra - that great big nurse sitting in front of the doctor's room, making sure no one gets in until the previous person comes out).

Deadlock

A situation in which a group of two or more processes are all waiting for a set of resources that are currently taken by other processes in the same group, or waiting for events that are supposed to be sent from other processes in the group. (mantra - when was the last time you were anxiously dialing to your boyfriend, finding his phone line constantly busy, while he was trying to call you back all this time? Each one of you was making one phone busy, while trying to dial the other's phone's number. If none of you would have let go of his/her phone, you'd be in a deadlock. The phone is, indeed, a valuable resource).

Starvation

A situation where a process that tries to access some resource is never granted access to that resource (mantra - think of the last time you went into your fast-food restaurant, approached the counter, and were always pushed back to the end of the line by the crowd, verbally "starving to death").

Race Condition

A situation in which two (or more) processes are doing some competing operations at the same time, and the results might come up screwed up due to collisions. (mantra - think of two people trying to get into the same door at the same time, jamming each other's way in).

Busy Wait

A situation in which a process that is waiting for a resource to become free, enters a loop of constantly polling the resource in order to find if it is free. In the process it consumes all available CPU power to perform its constant polls. (mantra - when you are waiting for the cable guy to come and connect you new apartment, and keep looking out the window all the time to see when the technician arrives).

Handling Events And Notifications

Asynchronous Notification

A method for one process to notify a second process about some state change, while the second process is executing some unrelated operations. This is usually done using hardware interrupts, or using Unix signals (see the Unix signal programming tutorial). In properly designed programs, this notification method is not commonly used. (mantra - you go downtown to do some business, when suddenly your cellular phone rings, with your dad on the other side of the line, telling you his PC tells him he is "bad and invalid",

and asks what to do...).

Synchronous Notification

A method for one process to notify a second process about some state change, that the second process will receive only when specifically polling for such notification. This is done by using functions such as `select()` (see internetworking with Unix sockets) in some sort of an event loop.

Event Loop

A control loop constantly executed by an event-driven program - its structure is:

1. Wait for a new event, suspending execution until one arrives from another process (or from the operating system).
2. Check which event type we got.
3. Invoke a function that can handle the given event type.
4. Go back to '1'.

(mantra - a technical support technician is sitting by the phone, doing nothing in particular. Every once in a while a phone call arrives - an event - and the technician handles the call, and afterwards goes back to waiting for another call).

Process Scheduling

Process Preemption

An event of the system suspending the execution of a process that is in the middle of putting the CPU to good use, in order to let other processes run. This makes sure no single process will take over the full CPU, even if it gets into an infinite loop and does not make any blocking system calls.

Context Switch

The method by which the operating system makes one CPU suspend the execution of on process, and continue executing a second process. This is called a context switch since the CPU needs to switch to using the full context of the second process - its execution stack, its memory area, the values the registers contained when last executing this process, etc. A context switch may occure when the running process blocks waiting for some resource to become free, when it yields the CPU voluntarily, or when it is being preempted by the operating system's scheduler. (mantra - you talk to your mom on the phone, while trying to watch a very interesting T.V. program - you switch your attention between you T.V. and your mom constantly, always taking a short while to remember where you were before the last switch).

Round-Robin Scheduling

A scheduling algorithm in which several processes are being executed by an executed one after the other in turn, each getting the same amount of time to run. After this time slice passes, a context-switch is made, where the first process on the list gets to run next, while the suspended process is being put back at the end of the list. (mantra - when the ideal teacher in class makes sure every pupil gets a chance to speak in turn - well, as if

that ideal teacher exists...).

Prioritized Scheduling

A scheduling algorithm in which a priority is assigned to each process, and when several processes are waiting for the CPU to get executed, the one with highest priority is being run, usually until it gets blocked waiting for some resource, or until a process with a higher priority wakes up and demands CPU time. (mantra - on a T.V. debate, some people are given higher priority then others, and interrupt everybody else when they have something to say. On Israeli T.V. - see under 'Popolitica').

Load Balancing

A method by which several tasks are scheduled for separate CPUs or computers, based on the current load on each CPU. Sometimes this includes the ability to migrate processes from one CPU (or computer) to another during its execution.

Commands Pipelining

A process in which commands are broken into several smaller steps, that may be executed in sequence using different components of the CPU, meanwhile allowing others components of the CPU to start executing the next command in parallel. For example, suppose that we have a CPU component that can read data from memory, and another that can do mathematical calculations. If we have two consecutive addition commands that don't affect each other (i.e. working on un-related parameters), the CPU could first load the data for the first command using its I/O component, and then while the mathematical component calculates the sum of the data, the I/O component can start fetching the data for the next command in parallel, making overall execution faster. (mantra - when you wake up very late, and find you are very late for school, you start wearing your shirt, and while slipping it on your body with one hand, the other hand rushes to pick up a new pair of socks from the drawer, and so on).

Implementations Of Parallel Systems

Parallel systems implementation may be done in software, in hardware, or as a combination of both. It can also be made using symmetric elements, all of which are capable of performing the same tasks in the same level of efficiency, or using units that are specializing in in different jobs. We will show here several of the commonly used approaches for building a parallel system. In real life cases, we will usually see several methods combined, hopefully due to some good reasoning by the designers, and not by just the driving of marketeers wanting their product to carry the title of a "Parallelic system designed with the Cache-Coherency paradigm, using state-of-the-art virtual multi-threaded scheduling policy". You get the idea.

Implementations In Software

Software implementations of parallel systems usually have to handle the task of letting many processes run on a limited amount of hardware, using it in the most efficient way possible. Most efficient might mean "making sure the CPU is never idle", or better yet "making sure the whole system finishes its tasks as quickly as possible, in human time".

Multi Processes

This time we use the term process to denote an operating system process. All Unix systems, as well as many other operating systems, are multi-process systems. In most of these systems, each process is given its own address space, and they all share the same CPU in turn. The scheduling algorithm might be a simple round-robin algorithm, or one that takes priorities into account (priorities are mostly needed for real-time programs, that must be granted some limits on how long it'll take since an event they need arrives, until they are allowed to execute it).

Multi Threading Kernels

A system similar to the multi-process system, except that here a process may be divided into several threads. All threads share the same data area, and are being scheduled in a similar way to how processes are scheduled. Due to the sharing of data area (and few other resources), a context-switch between two threads of the same process is faster then a context switch between two processes. Also, passing data between threads is easier then between two processes, due to the shared address space. On the other hand, protection of threads from one another is weaker then that given to processes - if one thread corrupts its memory area, all other threads in its process may be directly affected. Threads supported by the kernel are sometimes called 'Light-Weight Processes' (LWP).

Multi Threading Libraries

Similar to the kernel-based threads, except that the operating system's kernel is not aware of these threads - they are created and handled by a library. The advantage is that "context-switching" between them is faster - it does not involve any system call (with the overhead of switching into kernel mode). On the other hand, if one thread gets blocked by the operating system when invoking some system call (e.g. `sleep()`, or waiting for input from a network device), the whole process gets suspended, including all its threads. Thus a multi-threaded library implementation is suitable for calculation-intensive applications, not for I/O intensive applications.

Implementations In Hardware

Implementations of parallel systems often involve using extra hardware - several CPUs that can execute different processes in parallel being the most notable hardware addition.

SMP - Symmetric Multi-Processing

This method is used to contain several CPUs in a single computer, each of which has access to the same set of memory chips, and each working as a general-purpose CPU, that can execute any process in the system. The operating system is responsible for assigning newly created processes to specific processes, using some load-balancing algorithm. Sometimes these systems also handle moving a process from one CPU to another, that just became free.

In the past it was a simple task (after the current CPU finished executing one command, simple copy its registers contents into another CPU, and set it to continue execution from the same position). However, these days a CPU executes several commands of a process simultaneously, using pipelining techniques, and also contains an internal cache containing some data and commands, making such a process migration harder to implement, and more wasteful (all the partially-completed commands in the pipeline have to be flashed, the cache of the second CPU has to be reloaded, etc).

SMP systems exist now on many platforms, including PCs running with Intel and Intel-clone CPUs, Sun SPARC stations, using Ultra-SPARC CPUs (or older SPARC-10 and SPARC-20 CPUs), IBM RS/6000 systems using several PowerPC CPUs, and so on. Support for SMP systems is built into many operating systems running on these types of hardware, usually different by the amount of CPUs supported, as well as various internal implementation parameters.

CC-NUMA - Cache-Coherent Non-Uniform Memory Access

This is a different concept of having a parallel system with several CPUs. sometimes this system uses specialized processes to do different tasks, sometimes the access to just access to different memory parts is done in an asymmetric way (i.e. every CPU (or group of CPUs) have its own memory part, and thus memory is not shared between CPUs). An example of such a system is Silicon Graphic's Origin 2000 system. CC-NUMA systems are usually harder to implement then SMP systems (and thus more expensive), and thus normally not found in low-end or mid-sized systems.

Clustering

Clustering is a technique used to make several computers act as one larger machine, splitting tasks amongst them. They allow one to take several cheap stations, and combine them together to a larger system. It also allows for more redundancy for the system - if one machine in the

cluster dies off, the other machines can cover up for it until the malfunctioning machine is repaired, and all this without bringing the whole system down. This type of setup is thus common in systems that must run 24 hours non-stop.

Clustering is often implemented in software, often using a protocol named PVM to communicate between the different machines. Examples fir such systems are Beowulf, for Linux systems, or the clustering system by Tandem corporation.

Tools And Methods For Parallel Applications Development

Writing a parallel application takes a different approach then writing a sequential program. After we decide what needs to be done, we need to decide who gets to do what, and find points where extra parallelism would be beneficial. We then need to decide how our different runnables are going to communicate with one another - sometimes a whole slew of different communications methods is used in one large parallel application, each of which fits a particular need best. We then come to the art of debugging parallel applications, which requires some techniques not required when debugging sequential applications. You will note that similar techniques are also used when debugging device drivers, and even windowing GUI applications.

Note that we're not trying to teach the whole methodology in a few paragraphs, but rather just to point out a few places where one might search for more information and wisdom.

Designing A Parallel Application

The first step in designing a parallel application, is determining what level or parallelism, if at all, is beneficial to the problem our application tries to solve. In many cases, parallelism would add much more overhead, then benefits. An important factor is the experience of the programmers with parallel systems. This is not a factor when you're trying to learn, of-course, but it is a factor if you want to get something done in an reasonable amount of time. take into account some extra overhead needed to fix hard bugs that stem from timing problems, race conditions, deadlocks and the like.

Once we decided to use parallel programming, we should work on decomposing our system into units that would logically belong to a single runnable. Sometimes we find very natural divisions, other times only experience will help us, or better, looking at other similar applications for which we could find some success record. If we're programming in order to learn, we should mostly experiment, write code, test it, dump bad ideas, and be ready to write again from scratch. If we see our design leads to new complexities, its probably time for a change.

Communications Frameworks

A very important factor for the success or a parallel application, is choosing an appropriate communications framework. There are several such framework in common use, and for anything but simplistic and experimental work, we should consider using one of them. We'll show here a few examples, thought of-course other methods (including methods implemented by various commercial products) exist.

ONC RPC - Remote Procedure Call

Remote Procedure Calls (RPC) are a method originally developed by Sun microsystems©, allows one process to activate a procedure in a second process, passing it parameters, and optionally getting a result back from the call.

The set of procedures supported by a process are defined in a file using notation called 'RPC Language', and is pre-processed by a tool named 'rpcgen', which creates two groups of files forming two 'stubs'. One stub defines functions whose invocation will cause a message to be sent to the remote process, with a request to invoke a certain procedure. This function is invoked by the first (client) process, and returns when we get a reply from the second (server) process, with the value it returned. The second stub contains declarations of functions that need to be implemented by the second (server) process, in order to actually implement the procedures.

During the years, new RPC variants were created, most notably 'DCE RPC', which is part of the 'Distributed Computing Environment', now being maintained by the open group.

OMG's CORBA - Common Object Request Broker Architecture

CORBA (Common Object Request Broker Architecture) was started up as an attempt of few hundreds of companies to define a standard that allows clients to invoke methods on specific objects running in remote servers.

This framework defines a language-neutral protocol to allow processes to communicate even if they are written in different programming languages, or running on different operating systems. A declarative language, named IDL (Interface Definition Language) was defined to allow specifying language-neutral interfaces. Each interface has a name, and contains a set of methods and attributes. The interface is then pre-processed by some tool, and generates client and server stubs, similarly to how it is done with 'rpcgen' for RPC. An entity named 'ORB' (Object Request Broker) is then used to allow these clients and servers to communicate.

Above this basic interface, a set of standard services were defined, that supply features that are commonly required: A naming service, to allow client processes to locate remote objects by name, An event service, to allow different objects to register for events sent by other objects, and so on. These services are collectively known as 'Horizontal CORBA Services'.
Yet other services are being defined for different areas of computing, for instance, services to

be used by medical applications. These are called 'vertical services'.

For more information about CORBA, please refer to the Object Management Group's web site. You may also check the free CORBA page to locate and download various 'free' implementations of CORBA.

Microsoft©'s DCOM - Distributed Component Object Model

I might annoy some of the readers here, but although we are dealing here with Unix programming, we cannot ignore what appears to be the currently more developed distributed objects framework, DCOM (Distributed Component Object Model). DCOM gives us rather similar services to what CORBA does, and people usually argue that their range of services is smaller then what CORBA provides. However, DCOM served as the foundation for the ActiveX set of interfaces, that are used to allow one windows application to activate another one and fully control it. This is most commonly used to allow one application to embed an object created by another application, and allow the user to manipulate the embedded object by invoking the second application when required. Of-course, the ActiveX interface allows much more then that.

There are several reasons why DCOM is important also for Unix programmers:

1. In many cases, one needs to be able to make Unix and Windows programs communicate. For that reason, a standard for a CORBA to DCOM interface has been defined by the OMG, and you, as a Unix programmer, might find yourself in a need to interface with such programs.
2. There are various ideas that exist in DCOM, that are worthy of looking at, and implementing on top of native Unix frameworks (such as CORBA). The rule here is - "don't throw out the baby with the bath water".
3. The Win32 API system is being ported to the Unix environment by several companies, and COM (the non-distributed version of DCOM) is already available for various Unix platforms. DCOM is also being ported to these platforms. When this is done, Unix programmers will be able to employ DCOM without the need to use a Microsoft© operating system.

Third-Party Libraries Supporting Process/Thread Abstractions

Various third-party libraries exist, whose purpose is to ease the development of cross-platform applications. of those, various libraries try to make multi-process and multi-threaded programming easier.

ACE (Adaptive Communications Environment) is a large C++ library, developed at the Washington university in St. Louis. ACE attempts to supply abstractions for a lot of system programming concepts, including sockets, pipes, share memory, processes and threads. These abstractions allow one source-code to be compiled by different compilers on different operating

systems, from PCs running Linux, BSD and windows systems, through most types of Unix for workstations, and up to IBM's MVS open edition, and not to forget several real-time operating systems, such as VxWorks and LynxOS. There is also a version of ACE ported to Java, named JACE.

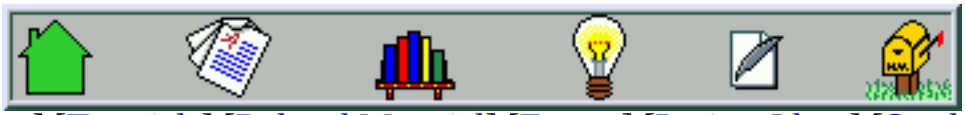
Rogue Wave© is a company known for writing commercial libraries that are used to ease the development of applications. One of their libraries is named 'Threads++', and is used to make multi-threaded programming easier. This library is something to consider when developing a commercial multi-threaded application. Refer to Rogue Wave's home page for more information.

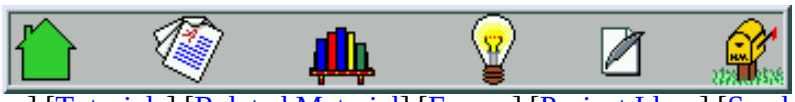
Debugging And Logging Techniques

Last, but not least, comes the host of problems we face when trying to debug parallel applications. The first problem is that things happen in different processes or threads at the same time, and we need a debugger that can follow all relevant runnables simultaneously. most debuggers now can handle processes properly, and on platforms with multi-threading support, usually the native debugger can handle threads as well. Of course, we need some kind of IDE if we want to use these debuggers without loosing our sanity.

However, because of the complex nature of such applications, and their sensitivity to timing issues, stopping the application in order to examine it with a debugger is something we sometimes cannot afford. Thus, our best shot would be at extensive logging, that can be turned on and off while the application is running. Such a logging facility must allow us to see which process or thread wrote each log record, and exactly when, to be able to deduce anything out of this info. We would also need to make the format of the log files easy to parse, in order to locate interesting events. It is advised that such a logging mechanism be devised early in the life of the project, as it can save hours of battling processes later.

One of the first problems we face when looking for the the cause of a bug in a parallel application, is finding the responsible process. Many times several processes are sending messages in a chain, and this chain breaks somewhere along the way. One method that could be used to debug such problems between two interacting processes, is simply suspending both of them. Then running the one that should initiate the message with a debugger, checking that the data it contains is legal and that the message it is about to send contains correct information. Then we can attach a debugger to the second process, set a breakpoint on the function that is supposed to receive the message, and resume the execution of this process. Of-course, this method cannot be employed when the message handling is sensitive to some timing constraints. In that case, only extensive logging will help us.





v1.01

Unix Multi-Process Programming and Inter-Process Communications (IPC)

Table Of Contents:

Preface And Motivation

One of the strong features of Unix-like operating systems, is their ability to run several processes simultaneously, and let them all share the CPU(s), memory, and other resources. Any none-trivial system developed on Unix systems will sooner or later resort to splitting its tasks into more then one process. True, many times threads will be preferred (thought these are candidates for a separate tutorial), but the methods used in both of them tend to be rather similar - how to start and stop processes, how to communicate with other processes, how to synchronize processes.

We'll try to learn here the various features that the system supplies us with in order to answer these questions. One should note that dealing with multi-process systems takes a slightly different approach then dealing with a single-process program - events happen to occur in parallel, debugging is more complicated, and there's always the risk of having a bug cause endless process creation that'll bring your system to a halt. In fact, when i took a course that tried teaching these subjects, at the week we had to deliver one of the exercises, the system hardly managed to survive the bunch of students running buggy programs that create new processes endlessly, or leaving background processes running into endless loop, and so on. OK, lets stop intimidating, and get to see how it's done.

What Is A Unix Process

Before we talk about processes, we need to understand exactly what a process is. If you know exactly what it is, and are familiar with the notion of 'Re-entrancy', you may skip to the next section...

You didn't skip? OK. Lets try to write a proper definition:

Unix Process

> An entity that executes a given piece of code, has its own execution stack, its own set of memory pages, its own file descriptors table, and a unique process ID.

As you might understand from this definition, a process is not a program. several processes may be executing the same computer program at the same time, for the same user or for several different users. For example, there is normally one copy of the 'tcsh' shell on the system, but there may be many tcsh processes running - one for each interactive connection of a user to the system. It might be that many different processes will try to execute the same piece of code at the same time, perhaps trying to utilize the same resources, and we should be ready to accommodate such situations.

This leads us to the concept of 'Re-entrancy'.

Re-entrancy

> The ability to have the same function (or part of a code) being in some phase of execution, more then once at the same time.

This re-entrancy might mean that two or more processes try to execute this piece of code at the same time. it might also mean that a single process tries to execute the same function several times simultaneously. How this may be possible? a simple example is a recursive function. The process starts executing it, and somewhere in the middle (before exiting the function), it calls the same function again. This means that the function should only use local variables to save its state information, for example.

Of-course, with a multi-process code, we don't have conflicts of variables, because normally the data section of each process is separate from that of other processes (so process A that runs program P and process B that runs the same program P, have distinct copies of the global variable 'i' of that program), but there might be other resources that would cause a piece of code to be non-reentrant. For example, if the program opens a file and writes some data to it, and two processes try to run the program at the same time, the contents of the file might be ruined in an unpredictable way. This is why a program must protect itself by using some kind of 'locking' mechanism, that will only allow one process at a time to open the file and write data into the file. An example of such a mechanism is the usage of semaphores, which will be discussed later on.

Process Creation

As you might (hopefully) already know, it is possible for a user to run a process in the system, suspend it (Ctrl-Z), and move it to the background (using the 'bg' command). If you're not familiar with this, you would do best to read the 'Job Control' section of the 'csh' manual page (or of 'bash', if that is the shell you normally use). However, we are interested in learning how to create new processes from within a C program.

The `fork()` System Call

The `fork()` system call is the basic way to create a new process. It is also a very unique system call, since it returns twice(!) to the caller. Sounds confusing? good. This confusion stems from the attempt to define as few systems calls as possible, it seems. OK, lets see:

`fork()`

> This system call causes the current process to be split into two processes - a parent process, and a child process. All of the memory pages used by the original process get duplicated during the `fork()` call, so both parent and child process see the exact same image. The only distinction is when the call returns. When it returns in the parent process, its return value is the process ID (PID) of the child process. When it returns inside the child process, its return value is '0'. If for some reason this call failed (not enough memory, too many processes, etc.), no new process is created, and the return value of the call is '-1'. In case the process was created successfully, both child process and parent process continue from the same place in the code where the `fork()` call was used.

To make things clearer, lets see an example of a code that uses this system call to create a child process that prints (you guessed it) "hello world" to the screen, and exits.

```
#include <unistd.h>     /* defines fork(), and pid_t.      */
#include <sys/wait.h>   /* defines the wait() system call. */

/* storage place for the pid of the child process, and its exit status. */
pid_t child_pid;
int child_status;
```

```
/* lets fork off a child process... */
child_pid = fork();

/* check what the fork() call actually did */
switch (child_pid) {
    case -1:    /* fork() failed */
        perror("fork"); /* print a system-defined error message */
        exit(1);
    case 0:     /* fork() succeeded, we're inside the child process */
        printf("hello world\n");
        exit(0);        /* here the CHILD process exits, not the parent. */
    default:    /* fork() succeeded, we're inside the parent process */
        wait(&child_status);    /* wait till the child process exits */
}
/* parent's process code may continue here... */
```

Notes:

- The `perror()` function prints an error message based on the value of the errno variable, to stderr.
- The `wait()` system call waits until any child process exits, and stores its exit status in the variable supplied. There are a set of macros to check this status, that will be explained in the next section.

Note: `fork()` copies also a memory area known as the 'U Area' (or User Area). This area contains, amongst other things, the file descriptor table of the process. This means that after returning from the `fork()` call, the child process inherits all files that were open in the parent process. If one of them reads from such an open file, the read/write pointer is advanced for both of them. On the other hand, files opened after the call to `fork()` are not shared by both processes. Further more, if one process closes a shared file, it is still kept open in the other process.

Child Process Termination

Once we have created a child process, there are two possibilities. Either the parent process exits before the child, or the child exits before the parent. Now, Unix's semantics regarding parent-child process relations state something like this:

- When a child process exits, it is not immediately cleared off the process table. Instead, a signal is sent to its parent process, which needs to acknowledge it's child's death, and only then the child process is completely removed from the system. In the duration before the parent's acknowledgment and after the child's exit, the child process is in a state called "zombie". (for info about Unix signals, please refer to our Unix signals programming tutorial).

When the parent process is not properly coded, the child remains in the zombie state forever. Such processes can be noticed by running the 'ps' command (shows the process list), and seeing processes having the string "<defunct>" as their command name.

The `wait()` System Call

The simple way of a process to acknowledge the death of a child process is by using the `wait()` system call. As we mentioned earlier, When `wait()` is called, the process is suspended until one of its child processes exits, and then the call return with the exit status of the child process. If it has a zombie child process, the call returns immediately, with the exit status of that process.

Asynchronous Child Death Notification

The problem with calling `wait()` directly, is that usually you want the parent process to do other things, while its

child process executes its code. Otherwise, you're not really enjoying multi-processes, do you? That problem has a solution by using signals. When a child process dies, a signal, SIGCHLD (or SIGCLD) is sent to its parent process. Thus, using a proper signal handler, the parent will get an asynchronous notification, and then when it'll call `wait()`, the system assures that the call will return immediately, since there is already a zombie child. Here is an example of our "hello world" program, using a signal handler this time.

```
#include <stdio.h>      /* basic I/O routines.   */
#include <unistd.h>     /* define fork(), etc.   */
#include <sys/types.h> /* define pid_t, etc.    */
#include <sys/wait.h>  /* define wait(), etc.   */
#include <signal.h>     /* define signal(), etc. */

/* first, here is the code for the signal handler */
void catch_child(int sig_num)
{
    /* when we get here, we know there's a zombie child waiting */
    int child_status;

    wait(&child_status);
    printf("child exited.\n");
}

.
.
/* and somewhere in the main() function ... */
.
.

/* define the signal handler for the CHLD signal */
signal(SIGCHLD, catch_child);

/* and the child process forking code... */
{
    int child_pid;
    int i;

    child_pid = fork();
    switch (child_pid) {
        case -1:         /* fork() failed */
            perror("fork");
            exit(1);
        case 0:          /* inside child process  */
            printf("hello world\n");
            sleep(5);    /* sleep a little, so we'll have */
                         /* time to see what is going on  */
            exit(0);
        default:         /* inside parent process */
            break;
    }
    /* parent process goes on, minding its own business... */
    /* for example, some output...                          */
    for (i=0; i<10; i++) {
        printf("%d\n", i);
        sleep(1);    /* sleep for a second, so we'll have time to see the mix */
```

```
    }
}
```

Lets examine the flow of this program a little:

1. A signal handler is defined, so whenever we receive a SIGCHLD, catch_child will be called.
2. We call `fork()` to spawn a child process.
3. The parent process continues its control flow, while the child process is doing its own chores.
4. When the child calls `exit()`, a CHLD signal is sent by the system to the parent.
5. The parent process' execution is interrupted, and its CHLD signal handler, catch_child, is invoked.
6. The `wait()` call in the parent causes the child to be completely removed off the system.
7. finally, the signal handler returns, and the parent process continues execution at the same place it was interrupted in.

Communications Via Pipes

Once we got our processes to run, we suddenly realize that they cannot communicate. After all, often when we start one process from another, they are supposed to accomplish some related tasks. One of the mechanisms that allow related-processes to communicate is the pipe, or the anonymous pipe.

What Is A Pipe?

One of the mechanisms that allow related-processes to communicate is the pipe, or the anonymous pipe. A pipe is a one-way mechanism that allows two related processes (i.e. one is an ancestor of the other) to send a byte stream from one of them to the other one. Naturally, to use such a channel properly, one needs to form some kind of protocol in which data is sent over the pipe. Also, if we want a two-way communication, we'll need two pipes, and a lot of caution...

The system assures us of one thing: The order in which data is written to the pipe, is the same order as that in which data is read from the pipe. The system also assures that data won't get lost in the middle, unless one of the processes (the sender or the receiver) exits prematurely.

The `pipe()` System Call

This system call is used to create a read-write pipe that may later be used to communicate with a process we'll fork off. The call takes as an argument an array of 2 integers that will be used to save the two file descriptors used to access the pipe. The first to read from the pipe, and the second to write to the pipe. Here is how to use this function:

```
/* first, define an array to store the two file descriptors */
int pipes[2];

/* now, create the pipe */
int rc = pipe(pipes);
if (rc == -1) { /* pipe() failed */
    perror("pipe");
    exit(1);
}
```

If the call to `pipe()` succeeded, a pipe will be created, pipes[0] will contain the number of its read file descriptor, and pipes[1] will contain the number of its write file descriptor.

Now that a pipe was created, it should be put to some real use. To do this, we first call `fork()` to create a child process, and then use the fact that the memory image of the child process is identical to the memory image of the parent process, so the pipes[] array is still defined the same way in both of them, and thus they both have the file descriptors of the pipe. Further more, since the file descriptor table is also copied during the fork, the file descriptors are still valid inside the child process.

Lets see an example of a two-process system in which one (the parent process) reads input from the user, and sends it to the other (the child), which then prints the data to the screen. The sending of the data is done using the pipe, and the protocol simply states that every byte passed via the pipe represents a single character typed by the user.

```
#include <stdio.h>    /* standard I/O routines.                */
#include <unistd.h>   /* defines pipe(), amongst other things. */

/* this routine handles the work of the child process. */
void do_child(int data_pipe[]) {
    int c;      /* data received from the parent. */
    int rc;     /* return status of read().       */

    /* first, close the un-needed write-part of the pipe. */
    close(data_pipe[1]);

    /* now enter a loop of reading data from the pipe, and printing it */
    while ((rc = read(data_pipe[0], &c, 1)) > 0) {
        putchar(c);
    }

    /* probably pipe was broken, or got EOF via the pipe. */
    exit(0);
}

/* this routine handles the work of the parent process. */
void do_parent(int data_pipe[])
{
    int c;      /* data received from the user. */
    int rc;     /* return status of getchar().  */

    /* first, close the un-needed read-part of the pipe. */
    close(data_pipe[0]);

    /* now enter a loop of read user input, and writing it to the pipe. */
    while ((c = getchar()) > 0) {
        /* write the character to the pipe. */
        rc = write(data_pipe[1], &c, 1);
        if (rc == -1) { /* write failed - notify the user and exit */
            perror("Parent: write");
            close(data_pipe[1]);
            exit(1);
        }
    }

    /* probably got EOF from the user. */
    close(data_pipe[1]); /* close the pipe, to let the child know we're done. */
    exit(0);
}
```

```
/* and the main function. */
int main(int argc, char* argv[])
{
    int data_pipe[2]; /* an array to store the file descriptors of the pipe. */
    int pid;       /* pid of child process, or 0, as returned via fork.    */
    int rc;        /* stores return values of various routines.            */

    /* first, create a pipe. */
    rc = pipe(data_pipe);
    if (rc == -1) {
        perror("pipe");
        exit(1);
    }

    /* now fork off a child process, and set their handling routines. */
    pid = fork();

    switch (pid) {
        case -1:        /* fork failed. */
            perror("fork");
            exit(1);
        case 0:         /* inside child process.  */
            do_child(data_pipe);
            /* NOT REACHED */
        default:        /* inside parent process. */
            do_parent(data_pipe);
            /* NOT REACHED */
    }

    return 0;   /* NOT REACHED */
}
```

As we can see, the child process closed the write-end of the pipe (since it only needs to read from the pipe), while the parent process closed the read-end of the pipe (since it only needs to write to the pipe). This closing of the un-needed file descriptor was done to free up a file descriptor entry from the file descriptors table of the process. It isn't necessary in a small program such as this, but since the file descriptors table is limited in size, we shouldn't waste unnecessary entries.

The complete source code for this example may be found in the file one-way-pipe.c.

Two-Way Communications With Pipes

In a more complex system, we'll soon discover that this one-way communications is too limiting. Thus, we'd want to be able to communication in both directions - from parent to child, and from child to parent. The good news is that all we need to do is open two pipes - one to be used in each direction. The bad news, however, is that using two pipes might cause us to get into a situation known as 'deadlock':

Deadlock

> A situation in which a group of two or more processes are all waiting for a set of resources that are currently taken by other processes in the same group, or waiting for events that are supposed to be sent from other processes in the group.

Such a situation might occur when two processes communicate via two pipes. Here are two scenarios that could led to

such a deadlock:

1. Both pipes are empty, and both processes are trying to read from their input pipes. Each one is blocked on the read (cause the pipe is empty), and thus they'll remain stuck like this forever.
2. This one is more complicated. Each pipe has a buffer of limited size associated with it. When a process writes to a pipe, the data is placed on the buffer of that pipe, until it is read by the reading process. If the buffer is full, the `write()` system call gets blocked until the buffer has some free space. The only way to free space on the buffer, is by reading data from the pipe.
   Thus, if both processes write data, each to its 'writing' pipe, until the buffers are filled up, both processes will get blocked on the `write()` system call. Since no other process is reading from any of the pipes, our two processes have just entered a deadlock.

Lets see an example of a (hopefully) deadlock-free program in which one process reads input from the user, writes it to the other process via a pipe. the second process translates each upper-case letter to a lower-case letter and sends the data back to the first process. Finally, the first process writes the data to standard output.

```
#include <stdio.h>    /* standard I/O routines.                 */
#include <unistd.h>   /* defines pipe(), amongst other things.  */
#include <ctype.h>    /* defines isascii(), toupper(), and other */
                      /* character manipulation routines.       */

/* function executed by the user-interacting process. */
void user_handler(int input_pipe[], int output_pipe[])
{
    int c;    /* user input */
    int rc;   /* return values of functions. */

    /* first, close unnecessary file descriptors */
    close(input_pipe[1]); /* we don't need to write to this pipe.  */
    close(output_pipe[0]); /* we don't need to read from this pipe. */

    /* loop: read input, send via one pipe, read via other */
    /* pipe, and write to stdout. exit on EOF from user.   */
    while ((c = getchar()) > 0) {
        /* write to translator */
        rc = write(output_pipe[1], &c, 1);
        if (rc == -1) { /* write failed - notify the user and exit. */
            perror("user_handler: write");
            close(input_pipe[0]);
            close(output_pipe[1]);
            exit(1);
        }
        /* read back from translator */
        rc = read(input_pipe[0], &c, 1);
        if (rc <= 0) { /* read failed - notify user and exit. */
            perror("user_handler: read");
            close(input_pipe[0]);
            close(output_pipe[1]);
            exit(1);
        }
        /* print translated character to stdout. */
        putchar(c);
    }

    /* close pipes and exit. */
```

```
    close(input_pipe[0]);
    close(output_pipe[1]);
    exit(0);
}

/* now comes the function executed by the translator process. */
void translator(int input_pipe[], int output_pipe[])
{
    int c;    /* user input */
    int rc;   /* return values of functions. */

    /* first, close unnecessary file descriptors */
    close(input_pipe[1]); /* we don't need to write to this pipe.  */
    close(output_pipe[0]); /* we don't need to read from this pipe. */

    /* enter a loop of reading from the user_handler's pipe, translating */
    /* the character, and writing back to the user handler.               */
    while (read(input_pipe[0], &c, 1) > 0) {
        /* translate any upper-case letter to lower-case. */
        if (isascii(c) && isupper(c))
            c = tolower(c);
        /* write translated character back to user_handler. */
        rc = write(output_pipe[1], &c, 1);
        if (rc == -1) { /* write failed - notify user and exit. */
            perror("translator: write");
            close(input_pipe[0]);
            close(output_pipe[1]);
            exit(1);
        }
    }

    /* close pipes and exit. */
    close(input_pipe[0]);
    close(output_pipe[1]);
    exit(0);
}

/* and finally, the main function: spawn off two processes, */
/* and let each of them execute its function.               */
int main(int argc, char* argv[])
{
    /* 2 arrays to contain file descriptors, for two pipes. */
    int user_to_translator[2];
    int translator_to_user[2];
    int pid;       /* pid of child process, or 0, as returned via fork.    */
    int rc;        /* stores return values of various routines.            */

    /* first, create one pipe. */
    rc = pipe(user_to_translator);
    if (rc == -1) {
        perror("main: pipe user_to_translator");
        exit(1);
    }
    /* then, create another pipe. */
    rc = pipe(translator_to_user);
```

```
    if (rc == -1) {
        perror("main: pipe translator_to_user");
        exit(1);
    }

    /* now fork off a child process, and set their handling routines. */
    pid = fork();

    switch (pid) {
        case -1:        /* fork failed. */
            perror("main: fork");
            exit(1);
        case 0:         /* inside child process.  */
            translator(user_to_translator, translator_to_user); /* line 'A' */
            /* NOT REACHED */
        default:        /* inside parent process. */
            user_handler(translator_to_user, user_to_translator); /* line 'B' */
            /* NOT REACHED */
    }

    return 0;   /* NOT REACHED */
}
```

A few notes:

1. Character handling: `isascii()` is a function that checks if the given character code is a valid ASCII code. `isupper()` is a function that checks if a given character is an upper-case letter. `tolower()` is a function that translates an upper-case letter to its equivalent lower-case letter.
2. Note that both functions get an input_pipe and an output_pipe array. However, when calling the functions we must make sure that the array we give one as its input pipe - we give the other as its output pipe, and vice versa. Failing to do that, the user_handler function will write a character to one pipe, and then both functions will try to read from the other pipe, thus causing both of them to block, as this other pipe is still empty.
3. Try to think: what will happen if we change the call in line 'A' above to:

   ```
   translator(user_to_translator, user_to_translator); /* line 'A' */
   ```

   and the code of line 'B' above to:

   ```
   user_handler(translator_to_user, translator_to_user); /* line 'B' */
   ```
4. Think harder now: what if we leave line 'A' as it was in the original program, and only modify line 'B' as in the previous question?

The complete source code for this example may be found in the file two-way-pipe.c.

Named Pipes

One limitation of anonymous pipes is that only processes 'related' to the process that created the pipe (i.e. siblings of that process.) may communicate using them. If we want two un-related processes to communicate via pipes, we need to use named pipes.

What Is A Named Pipe?

A named pipe (also called a named FIFO, or just FIFO) is a pipe whose access point is a file kept on the file system. By

opening this file for reading, a process gets access to the reading end of the pipe. By opening the file for writing, the process gets access to the writing end of the pipe. If a process opens the file for reading, it is blocked until another process opens the file for writing. The same goes the other way around.

Creating A Named Pipe With The `mknod` Command

A named pipe may be created either via the 'mknod' (or its newer replacement, 'mkfifo'), or via the `mknod()` system call (or by the POSIX-compliant `mkfifo()` function). To create a named pipe with the file named 'prog_pipe', we can use the following command:

```
mknod prog_pipe p
```

We could also provide a full path to where we want the named pipe created. If we then type 'ls -l prog_pipe', we will see something like this:

```
prw-rw-r--   1 choo     choo            0 Nov  7 01:59 prog_pipe
```

The 'p' on the first column denotes this is a named pipe. Just like any file in the system, it has access permissions, that define which users may open the named pipe, and whether for reading, writing or both.

Opening A Named Pipe For Reading Or Writing

Opening a named pipe is done just like opening any other file in the system, using the `open()` system call, or using the `fopen()` standard C function. If the call succeeds, we get a file descriptor (in the case of `open()`, or a 'FILE' pointer (in the case of `fopen()`), which we may use either for reading or for writing, depending on the parameters passed to `open()` or to `fopen()`.

Reading/Writing From/To A Named Pipe

Reading from a named pipe is very similar to reading from a file, and the same goes for writing to a named pipe. Yet there are several differences:

1. Either Read Or Write - a named pipe cannot be opened for both reading and writing. The process opening it must choose one mode, and stick to it until it closes the pipe.
2. Read/Write Are Blocking - when a process reads from a named pipe that has no data in it, the reading process is blocked. It does not receive an end of file (EOF) value, like when reading from a file. When a process tries to write to a named pipe that has no reader (e.g. the reader process has just closed the named pipe), the writing process gets blocked, until a second process re-opens the named pipe.

Thus, when writing a program that uses a named pipe, we must take these limitations into account. We could also turn the file descriptor via which we access the named pipe to a non-blocking mode. This, however, is out of the scope of our tutorial. For info about how to do that, and how to handle a non-blocking pipe, please refer to the manual pages of 'open(2)', fcntl(2), read(2) and write(2).

Named Pipe - A Complete Example

As an example to an obscure usage of named pipes, we will borrow some idea from a program that allows one to count how many times they have been "fingered" lately. As you might know, on many Unix systems, there is a finger daemon, that accepts requests from users running the "finger" program, with a possible user name, and tells them when this user last logged on, as well as some other information. Amongst other thing, the finger daemon also checks if the user has a file named '.plan' (that is dot followed by "plan") in her home directory. If there is such a file, the finger daemon opens it, and prints its contents to the client. For example, on my Linux machine, fingering my account might show something like:

```
[choo@simey1 ~]$ finger choo
Login: choo                             Name: guy keren
Directory: /home/choo                   Shell: /bin/tcsh
On since Fri Nov  6 15:46 (IDT) on tty6
No mail.
Plan:
- Breed a new type of dogs.
- Water the plants during all seasons.
- Finish the next tutorial on time.
```

As you can see, the contents of the '.plan' file has been printed out.

This feature of the finger daemon may be used to create a program that tells the client how many times i was fingered. For that to work, we first create a named pipe, where the '.plan' file resides:

```
mknod /home/choo/.plan p
```

If i now try to finger myself, the output will stop before showing the 'plan' file. How so? this is because of the blocking nature of a named pipe. When the finger daemon opens my '.plan' file, there is no write process, and thus the finger daemon blocks. Thus, don't run this on a system where you expect other users to finger you often.

The second part of the trick, is compiling the named-pipe-plan.c program, and running it. note that it contains the full path to the '.plan' file, so change that to the appropriate value for your account, before compiling it. When you run the program, it gets into an endless loop of opening the named pipe in writing mode, write a message to the named pipe, close it, and sleep for a second. Look at the program's source code for more information. A sample of its output looks like this:

```
[choo@simey1 ~]$ finger choo
Login: choo                             Name: guy keren
Directory: /home/choo                   Shell: /bin/tcsh
On since Fri Nov  6 15:46 (IDT) on tty6
No mail.
Plan:
I have been fingered 8 times today
```

When you're done playing, stop the program, and don't forget to remove the named pipe from the file system.

Few Words About Sockets

Various sockets-based mechanisms may be used to communicate amongst processes. The underlying communications protocol may be TCP, UDP, IP, or any other protocol from the TCP/IP protocols family. There is also a socket of type 'Unix-domain', which uses some protocol internal to the operating system to communicate between processes all residing on a single machine. Unix-domain sockets are similar to named pipes in that the communicating processes use a file in the system to connect to establish a connection. For more information about programming with sockets, please refer to our tutorial about internetworking with Unix sockets.

System V IPC

Many variants of Unix these days support a set of inter-process communications methods, which are derived from Unix System V release 4, originating from AT&T Bell laboratories. These mechanisms include message queues (used for sending and receiving messages), shared memory (used to allow several processes share data in memory) and

semaphores (used to co-ordinate access by several processes, to other resources). Each of these resource types is handled by the system, and unlike anonymous pipes, may out-live the process that created it. These resources also have some security support by the system, that allows one to specify which processes may access a given message queue, for example.

The fact that these resources are global to the system has two contradicting implications. On one hand, it means that if a process exits, the data it sent through a message queue, or placed in shared memory is still there, and can be collected by other processes. On the other hand, this also means that the programmer has to take care of freeing these resources, or they occupy system resources until the next reboot, or until being removed by hand.

I am going to make a statement here about these communications mechanisms, that might annoy some readers: System V IPC mechanisms are evil regarding their implementation, and should not be used unless there is a very good reason. One of the problem with these mechanism, is that one cannot use the `select()` (or its replacement, `poll()`) with them, and thus a process waiting for a message to be placed in a message queue, cannot be notified about messages coming via other resources (e.g. other message queues, pipes or sockets). In my opinion, this limitation is an oversight by the designers of these mechanisms. Had they used file descriptors to denote IPC resources (like they are used for pipes, sockets and files) life would be easier.

Another problem with System V IPC is their system-global nature. The total number of message queues that may live in the system, for example, is shared by all processes. Worse then that, the number of messages waiting in all messages queues is also limited globally. One process spewing many such messages will break all processes using message queues. The same goes for other such resources. There are various other limitations imposed by API (Application programming interface). For example, one may wait on a limited set of semaphores at the same time. If you want more then this, you have to split the waiting task, or re-design your application.

Having said that, there are still various applications where using system V IPC (we'll call it SysV IPC, for short) will save you a large amount of time. In these cases, you should go ahead and use these mechanism - just handle with care.

Permission Issues

Before delving into the usage of the different System V IPC mechanisms, we will describe the security model used to limit access to these resources.

Private Vs. Public

Each resource in SysV IPC may be either private or public. Private means that it may be accessed only by the process that created it, or by child processes of this process. Public means that it may be potentially accessed by any process in the system, except when access permission modes state otherwise.

Access Permission Modes - The 'ipc_perm' Structure

SysV IPC resources may be protected using access mode permissions, much like files and directories are protected by the Unix system. Each such resource has an owning user and an owning group. Permission modes define if and how processes belonging to different users in the system may access this resource. Permissions may be set separately for the owning user, for users from the owning group, and everyone else. permissions may be set for reading the resource (e.g. reading messages from a message queue), or writing to the resource (e.g. sending a message on a queue, changing the value of a semaphore). A structure of type 'ipc_perm', which is defined as follows:

```
struct ipc_perm
{
  key_t  key;   /* key identifying the resource                    */
  ushort uid;   /* owner effective user ID and effective group ID   */
  ushort gid;
  ushort cuid;  /* creator effective user ID and effective group ID */
  ushort cgid;
```

```
  ushort mode;  /* access modes                                   */
  ushort seq;   /* sequence number                                */
};
```

These fields have the following meanings:

- key - the identifier of the resource this structure refers to.
- uid - effective user ID owning the resource.
- gid - effective group ID owning the resource.
- cuid - effective user ID that created the resource.
- cgid - effective group ID that created the resource.
- mode - access permission modes for the given resource. This is a bit field, with the lowest 9 bits denoting access flags, and are a bit-wise 'or' of the following (octal) values:
  - 0400 - owning user may read from this resource.
  - 0200 - owning user may write to this resource.
  - 0040 - owning group may read from this resource.
  - 0020 - owning group may write to this resource.
  - 0004 - every other user may read from this resource.
  - 0002 - every other user may write to this resource.
- seq - used to keep system-internal info about the resource. for further info, check your kernel's sources (you are working on a system with free access to its source code, right?).

Part of the SysV IPC API allows us to modify the access permissions for the resources. We will encounter them when discussing the different IPC methods.

System Utilities To Administer System-V IPC Resources

Since SysV IPC resources live outside the scope of a single process, there is a need to manage them somehow - delete resources that were left by irresponsible processes (or process crashes); check the number of existing resources of each type (especially to find if the system-global limit was reached), etc. Two utilities were created for handling these jobs: 'ipcs' - to check usage of SysV IPC resources, and 'ipcrm' - to remove such resources.

Running 'ipcs' will show us statistics separately for each of the three resource types (shared memory segments, semaphore arrays and message queues). For each resource type, the command will show us some statistics for each resource that exists in the system. It will show its identifier, owner, size of resources it occupies in the system, and permission flags. We may give 'ipcs' a flag to ask it to show only resources of one type ('-m' for shared Memory segments, -q for message Queues and '-s' for Semaphore arrays). We may also use 'ipcs' with the '-l' flag to see the system enforced limits on these resources, or the '-u' flag to show us usage summary. Refer to the manual page of 'ipcs' for more information.

The 'ipcrm' command accepts a resource type ('shm', 'msg' or 'sem') and a resource ID, and removes the given resource from the system. We need to have the proper permissions in order to delete a resource.

Using Message Queues

One of the problems with pipes is that it is up to you, as a programmer, to establish the protocol. Now, usually this protocol is based on sending separate messages. With a stream taken from a pipe, it means you have to somehow parse the bytes, and separate them to packets. Another problem is that data sent via pipes always arrives in a FIFO order. This means that before you can read any part of the stream, you have to consume all the bytes sent before the piece you're looking for, and thus you need to construct your own queuing mechanism on which you place the data you just skipped, to be read later. If that's what you're interested at, this is a good time to get acquainted with message queues.

What Are Message Queues?

A message queue is a queue onto which messages can be placed. A message is composed of a message type (which is a number), and message data. A message queue can be either private, or public. If it is private, it can be accessed only by its creating process or child processes of that creator. If it's public, it can be accessed by any process that knows the queue's key. Several processes may write messages onto a message queue, or read messages from the queue. Messages may be read by type, and thus not have to be read in a FIFO order as is the case with pipes.

Creating A Message Queue - `msgget()`

In order to use a message queue, it has to be created first. The `msgget()` system call is used to do just that. This system call accepts two parameters - a queue key, and flags. The key may be one of:

- IPC_PRIVATE - used to create a private message queue.
- a positive integer - used to create (or access) a publicly-accessible message queue.

The second parameter contains flags that control how the system call is to be processed. It may contain flags like IPC_CREAT or IPC_EXCL, which behave similar to O_CREAT and O_EXCL in the `open()` system call, and will be explained later, and it also contains access permission bits. The lowest 9 bits of the flags are used to define access permission for the queue, much like similar 9 bits are used to control access to files. the bits are separated into 3 groups - user, group and others. In each set, the first bit refers to read permission, the second bit - to write permission, and the third bit is ignored (no execute permission is relevant to message queues).

Lets see an example of a code that creates a private message queue:

```
#include <stdio.h>     /* standard I/O routines.           */
#include <sys/types.h> /* standard system data types.      */
#include <sys/ipc.h>   /* common system V IPC structures.  */
#include <sys/msg.h>   /* message-queue specific functions. */

/* create a private message queue, with access only to the owner. */
int queue_id = msgget(IPC_PRIVATE, 0600); /* <-- this is an octal number. */
if (queue_id == -1) {
    perror("msgget");
    exit(1);
}
```

A few notes about this code:

1. the system call returns an integer identifying the created queue. Later on we can use this key in order to access the queue for reading and writing messages.
2. The queue created belongs to the user whose process created the queue. Thus, since the permission bits are '0600', only processes run on behalf of this user will have access to the queue.

The Message Structure - `struct msgbuf`

Before we go to writing messages to the queue or reading messages from it, we need to see how a message looks. The system defines a structure named 'msgbuf' for this purpose. Here is how it is defined:

```
struct msgbuf {
    long mtype;     /* message type, a positive number (cannot be zero). */
    char mtext[1];  /* message body array. usually larger then one byte. */
};
```

The message type part is rather obvious. But how do we deal with a message text that is only 1 byte long? Well, we actually may place a much larger text inside a message. For this, we allocate more memory for a msgbuf structure then `sizeof(struct msgbuf)`. Lets see how we create an "hello world" message:

```
/* first, define the message string */
char* msg_text = "hello world";
/* allocate a message with enough space for length of string and */
/* one extra byte for the terminating null character.           */
struct msgbuf* msg =
        (struct msgbuf*)malloc(sizeof(struct msgbuf) + strlen(msg_text));
/* set the message type. for example - set it to '1'. */
msg->mtype = 1;
/* finally, place the "hello world" string inside the message. */
strcpy(msg->mtext, msg_text);
```

Few notes:

1. When allocating a space for a string, one always needs to allocate one extra byte for the null character terminating the string. In our case, we allocated `strlen(msg_text)` more then the size of "`struct msgbuf`", and didn't need to allocate an extra place for the null character, cause that's already contained in the msgbuf structure (the 1 byte of mtext there).
2. We don't need to place only text messages in a message. We may also place binary data. In that case, we could allocate space as large as the msgbuf struct plus the size of our binary data, minus one byte. Of-course then to copy the data to the message, we'll use a function such as `memset()`, and not `strcpy()`.

Writing Messages Onto A Queue - `msgsnd()`

Once we created the message queue, and a message structure, we can place it on the message queue, using the `msgsnd()` system call. This system call copies our message structure and places that as the last message on the queue. It takes the following parameters:

1. `int msqid` - id of message queue, as returned from the `msgget()` call.
2. `struct msgbuf* msg` - a pointer to a properly initializes message structure, such as the one we prepared in the previous section.
3. `int msgsz` - the size of the data part (mtext) of the message, in bytes.
4. `int msgflg` - flags specifying how to send the message. may be a logical "or" of the following:
    - `IPC_NOWAIT` - if the message cannot be sent immediately, without blocking the process, return '-1', and set `errno` to `EAGAIN`.

    to set no flags, use the value '0'.

So in order to send our message on the queue, we'll use `msgsnd()` like this:

```
int rc = msgsnd(queue_id, msg, strlen(msg_text)+1, 0);
if (rc == -1) {
    perror("msgsnd");
    exit(1);
}
```

Note that we used a message size one larger then the length of the string, since we're also sending the null character. `msgsnd()` assumes the data in the message to be an arbitrary sequence of bytes, so it cannot know we've got the null

character there too, unless we state that explicitly.

Reading A Message From The Queue - `msgrcv()`

We may use the system call `msgrcv()` In order to read a message from a message queue. This system call accepts the following list of parameters:

1. `int msqid` - id of the queue, as returned from `msgget()`.
2. `struct msgbuf* msg` - a pointer to a pre-allocated msgbuf structure. It should generally be large enough to contain a message with some arbitrary data (see more below).
3. `int msgsz` - size of largest message text we wish to receive. Must NOT be larger then the amount of space we allocated for the message text in 'msg'.
4. `int msgtyp` - Type of message we wish to read. may be one of:
    - 0 - The first message on the queue will be returned.
    - a positive integer - the first message on the queue whose type (mtype) equals this integer (unless a certain flag is set in msgflg, see below).
    - a negative integer - the first message on the queue whose type is less then or equal to the absolute value of this integer.
5. `int msgflg` - a logical 'or' combination of any of the following flags:
    - `IPC_NOWAIT` - if there is no message on the queue matching what we want to read, return '-1', and set `errno` to `ENOMSG`.
    - `MSG_EXCEPT` - if the message type parameter is a positive integer, then return the first message whose type is NOT equal to the given integer.
    - `MSG_NOERROR` - If a message with a text part larger then 'msgsz' matches what we want to read, then truncate the text when copying the message to our msgbuf structure. If this flag is not set and the message text is too large, the system call returns '-1', and `errno` is set to `E2BIG`.

Lets then try to read our message from the message queue:

```
/* prepare a message structure large enough to read our "hello world". */
struct msgbuf* recv_msg =
     (struct msgbuf*)malloc(sizeof(struct msgbuf)+strlen("hello world"));
/* use msgrcv() to read the message. We agree to get any type, and thus */
/* use '0' in the message type parameter, and use no flags (0).         */
int rc = msgrcv(queue_id, recv_msg, strlen("hello world")+1, 0, 0);
if (rc == -1) {
    perror("msgrcv");
    exit(1);
}
```

A few notes:

1. If the message on the queue was larger then the size of "hello world" (plus one), we would get an error, and thus exit.
2. If there was no message on the queue, the `msgrcv()` call would have blocked our process until one of the following happens:
    - a suitable message was placed on the queue.
    - the queue was removed (and then `errno` would be set to `EIDRM`).
    - our process received a signal (and then `errno` would be set to `EINTR`.

Now that you've seen all the different parts, you're invited to look at the private-queue-hello-world.c program, for the complete program.

Message Queues - A Complete Example

As an example of using non-private message queues, we will show a program, named "queue_sender", that creates a message queue, and then starts sending messages with different priorities onto the queue. A second program, named "queue_reader", may be run that reads the messages from the queue, and does something with them (in our example - just prints their contents to standard output). The "queue_reader" is given a number on its command line, which is the priority of messages that it should read. By running several copies of this program simultaneously, we can achieve a basic level of concurrency. Such a mechanism may be used by a system in which several clients may be sending requests of different types, that need to be handled differently. The complete source code may be found in the public-queue directory.

Process Synchronization With Semaphores

One of the problems when writing multi-process application is the need to synchronize various operations between the processes. Communicating requests using pipes, sockets and message queues is one way to do it. however, sometimes we need to synchronize operations amongst more then two processes, or to synchronize access to data resources that might be accessed by several processes in parallel. Semaphores are a means supplied with SysV IPC that allow us to synchronize such operations.

What Is A Semaphore? What Is A Semaphore Set?

A semaphore is a resource that contains an integer value, and allows processes to synchronize by testing and setting this value in a single atomic operation. This means that the process that tests the value of a semaphore and sets it to a different value (based on the test), is guaranteed no other process will interfere with the operation in the middle.

Two types of operations can be carried on a semaphore: wait and signal. A set operation first checks if the semaphore's value equals some number. If it does, it decreases its value and returns. If it does not, the operation blocks the calling process until the semaphore's value reaches the desired value. A signal operation increments the value of the semaphore, possibly awakening one or more processes that are waiting on the semaphore. How this mechanism can be put to practical use will be explained soon.

A semaphore set is a structure that stores a group of semaphores together, and possibly allows the process to commit a transaction on part or all of the semaphores in the set together. In here, a transaction means that we are guaranteed that either all operations are done successfully, or none is done at all. Note that a semaphore set is not a general parallel programming concept, it's just an extra mechanism supplied by SysV IPC.

Creating A Semaphore Set - `semget()`

Creation of a semaphore set is done using the `semget()` system call. Similarly to the creation of message queues, we supply some ID for the set, and some flags (used to define access permission mode and a few options). We also supply the number of semaphores we want to have in the given set. This number is limited to `SEMMSL`, as defined in file `/usr/include/sys/sem.h`. Lets see an example:

```
/* ID of the semaphore set.     */
int sem_set_id_1;
int sem_set_id_2;

/* create a private semaphore set with one semaphore in it, */
/* with access only to the owner.                           */
sem_set_id_1 = semget(IPC_PRIVATE, 1, IPC_CREAT | 0600);
if (sem_set_id_1 == -1) {
    perror("main: semget");
    exit(1);
```

```
}

/* create a semaphore set with ID 250, three semaphores */
/* in the set, with access only to the owner.           */
sem_set_id_2 = semget(250, 3, IPC_CREAT | 0600);
if (sem_set_id_2 == -1) {
    perror("main: semget");
    exit(1);
}
```

Note that in the second case, if a semaphore set with ID 250 already existed, we would get access to the existing set, rather then a new set be created. This works just like it worked with message queues.

Setting And Getting Semaphore Values With `semctl()`

After the semaphore set is created, we need to initialize the value of the semaphores in the set. We do that using the `semctl()` system call. Note that this system call has other uses, but they are not relevant to our needs right now. Lets assume we want to set the values of the three semaphores in our second set to values 3, 6 and 0, respectively. The ID of the first semaphore in the set is '0', the ID of the second semaphore is '1', and so on.

```
/* use this to store return values of system calls.   */
int rc;

/* initialize the first semaphore in our set to '3'.  */
rc = semctl(sem_set_id_2, 0, SETVAL, 3);
if (rc == -1) {
    perror("main: semctl");
    exit(1);
}

/* initialize the second semaphore in our set to '6'. */
rc = semctl(sem_set_id_2, 1, SETVAL, 6);
if (rc == -1) {
    perror("main: semctl");
    exit(1);
}

/* initialize the third semaphore in our set to '0'.  */
rc = semctl(sem_set_id_2, 2, SETVAL, 0);
if (rc == -1) {
    perror("main: semctl");
    exit(1);
}
```

There are one comment to be made about the way we used `semctl()` here. According to the manual, the last parameter for this system call should be a union of type `union semun`. However, since the `SETVAL` (set value) operation only uses the `int val` part of the union, we simply passed an integer to the function. The proper way to use this system call was to define a variable of this union type, and set its value appropriately, like this:

```
/* use this variable to pass the value to the semctl() call */
union semun sem_val;

/* initialize the first semaphore in our set to '3'. */
```

```
sem_val.val = 0;
rc = semctl(sem_set_id_2, 2, SETVAL, sem_val);
if (rc == -1) {
    perror("main: semctl");
    exit(1);
}
```

We used the first form just for simplicity. From now on, we will only use the second form.

Using Semaphores For Mutual Exclusion With semop()

Sometimes we have a resource that we want to allow only one process at a time to manipulate. For example, we have a file that we only want written into only by one process at a time, to avoid corrupting its contents. Of-course, we could use various file locking mechanisms to protect the file, but we will demonstrate the usage of semaphores for this purpose as an example. Later on we will see the real usage of semaphores, to protect access to shared memory segments. Anyway, here is a code snippest. It assumes the semaphore in our set whose id is "sem_set_id" was initialized to 1 initially:

```
/* this function updates the contents of the file with the given path name. */
void update_file(char* file_path, int number)
{
    /* structure for semaphore operations.   */
    struct sembuf sem_op;
    FILE* file;

    /* wait on the semaphore, unless it's value is non-negative. */
    sem_op.sem_num = 0;
    sem_op.sem_op = -1;   /* <-- Comment 1 */
    sem_op.sem_flg = 0;
    semop(sem_set_id, &sem_op, 1);

    /* Comment 2 */
    /* we "locked" the semaphore, and are assured exclusive access to file.  */
    /* manipulate the file in some way. for example, write a number into it. */
    file = fopen(file_path, "w");
    if (file) {
        fprintf(file, "%d\n", number);
        fclose(file);
    }

    /* finally, signal the semaphore - increase its value by one. */
    sem_op.sem_num = 0;
    sem_op.sem_op = 1;   /* <-- Comment 3 */
    sem_op.sem_flg = 0;
    semop(sem_set_id, &sem_op, 1);
}
```

This code needs some explanations, especially regarding the semantics of the `semop()` calls.

1. Comment 1 - before we access the file, we use `semop()` to wait on the semaphore. Supplying '-1' in `sem_op.sem_op` means: If the value of the semaphore is greater then or equal to '1', decrease this value by one, and return to the caller. Otherwise (the value is 1 or less), block the calling process, until the value of the semaphore becomes '1', at which point we return to the caller.
2. Comment 2 - The semantics of `semop()` assure us that when we return from this function, the value of the

semaphore is 0. Why? it couldn't be less, or else `semop()` won't return. It couldn't be more due to the way we later on signal the semaphore. And why it cannot be more then '0'? read on to find out...

3. Comment 3 - after we are done manipulating the file, we increase the value of the semaphore by 1, possibly waking up a process waiting on the semaphore. If several processes are waiting on the semaphore, the first that got blocked on it is wakened and continues its execution.

Now, lets assume that any process that tries to access the file, does it only via a call to our "update_file" function. As you can see, when it goes through the function, it always decrements the value of the semaphore by 1, and then increases it by 1. Thus, the semaphore's value can never go above its initial value, which is '1'. Now lets check two scenarios:

1. No other process is executing the "update_file" concurrently. In this case, when we enter the function, the semaphore's value is '1'. after the first `semop()` call, the value of the semaphore is decremented to '0', and thus our process is not blocked. We continue to execute the file update, and with the second `semop()` call, we raise the value of the semaphore back to '1'.
2. Another process is in the middle of the "update_file" function. If it already managed to pass the first call to `semop()`, the value of the semaphore is '0', and when we call `semop()`, our process is blocked. When the other process signals the semaphore with the second `semop()` call, it increases the value of the semaphore back to '0', and it wakes up the process blocked on the semaphore, which is our process. We now get into executing the file handling code, and finally we raise the semaphore's value back to '1' with our second call to `semop()`.

We have the source code for a program demonstrating the mutex concept, in the file named sem-mutex.c. The program launches several processes (5, as defined by the NUM_PROCS macro), each of which is executing the "update_file" function several times in a row, and then exits. Try running the program, and scan its output. Each process prints out its PID as it updates the file, so you can see what happens when. Try to play with the DELAY macro (specifying how long a process waits between two calls to "update_file") and see how it effects the order of the operations. Check what happens if you replace the delay loop in the "do_child_loop" function, with a call to `sleep()`.

Using Semaphores For Producer-Consumer Operations With `semop()`

Using a semaphore as a mutex is not utilizing the full power of the semaphore. As we saw, a semaphore contains a counter, that may be used for more complex operations. Those operations often use a programming model called "producer-consumer". In this model, we have one or more processes that produce something, and one or more processes that consume that something. For example, one set of processes accept printing requests from clients and place them in a spool directory, and another set of processes take the files from the spool directory and actually print them using the printer.

To control such a printing system, we need the producers to maintain a count of the number of files waiting in the spool directory and incrementing it for every new file placed there. The consumers check this counter, and whenever it gets above zero, one of them grabs a file from the spool, and sends it to the printer. If there are no files in the spool (i.e. the counter value is zero), all consumer processes get blocked. The behavior of this counter sounds very familiar.... it is the exact same behavior of a counting semaphore.

Lets see how we can use a semaphore as a counter. We still use the same two operations on the semaphore, namely "signal" and "wait".

```
/* this variable will contain the semaphore set. */
int sem_set_id;

/* semaphore value, for semctl().                */
union semun sem_val;

/* structure for semaphore operations.           */
struct sembuf sem_op;

/* first we create a semaphore set with a single semaphore, */
```

```
/* whose counter is initialized to '0'.                    */
sem_set_id = semget(IPC_PRIVATE, 1, 0600);
if (sem_set_id == -1) {
    perror("semget");
    exit(1);
}
sem_val.val = 0;
semctl(sem_set_id, 0, SETVAL, sem_val);

/* we now do some producing function, and then signal the   */
/* semaphore, increasing its counter by one.                */
.
.
sem_op.sem_num = 0;
sem_op.sem_op = 1;
sem_op.sem_flg = 0;
semop(sem_set_id, &sem_op, 1);
.
.
.
/* meanwhile, in a different process, we try to consume the      */
/* resource protected (and counter) by the semaphore.            */
/* we block on the semaphore, unless it's value is non-negative. */
sem_op.sem_num = 0;
sem_op.sem_op = -1;
sem_op.sem_flg = 0;
semop(sem_set_id, &sem_op, 1);

/* when we get here, it means that the semaphore's value is '0'  */
/* or more, so there's something to consume.                     */
.
.
```

Note that our "wait" and "signal" operations here are just like we did with when using the semaphore as a mutex. The only difference is in who is doing the "wait" and the "signal". With a mutex, the same process did both the "wait" and the "signal" (in that order). In the producer-consumer example, one process is doing the "signal" operation, while the other is doing the "wait" operation.

The full source code for a simple program that implements a producer-consumer system with two processes, is found in the file sem-producer-consumer.c.

Semaphores - A Complete Example

As a complete example of using semaphores, we write a very simple print spool system. Two separate programs will be used. One runs as the printing command, and is found in the file tiny-lpr.c. It gets a file path on its command line, and copies this file into the spool area, increasing a global (on-private) semaphore by one. Another program runs as the printer daemon, and is found in the file tiny-lpd.c. It waits on the same global semaphore, and whenever its value is larger then one, it locates a file in the spool directory and sends it to the printer. In order to avoid race conditions when copying files into the directory and removing files from this directory, a second semaphore will be used as a mutex, to protect the spool directory. The complete tiny-spooler mini-project is found in the tiny-spool directory.

One problem might be that copying a file takes a lot of time, and thus locking the spool directory for a long while. In order to avoid that, 3 directories will be used. One serves as a temporary place for tiny-lpr to copy files into. One will be used as the common spool directory, and one will be used as a temporary directory into which tiny-lpd will move the files before printing them. By putting all 3 directories on the same disk, we assure that files can be moved between

them using the `rename()` system call, in one fast operation (regardless of the file size).

Shared Memory

As we have seen, many methods were created in order to let processes communicate. All this communications is done in order to share data. The problem is that all these methods are sequential in nature. What can we do in order to allow processes to share data in a random-access manner?
Shared memory comes to the rescue. As you might know, on a Unix system, each process has its own virtual address space, and the system makes sure no process would access the memory area of another process. This means that if one process corrupts its memory's contents, this does not directly affect any other process in the system.

With shared memory, we declare a given section in the memory as one that will be used simultaneously by several processes. This means that the data found in this memory section (or memory segment) will be seen by several processes. This also means that several processes might try to alter this memory area at the same time, and thus some method should be used to synchronize their access to this memory area (did anyone say "apply mutual exclusion using a semaphore" ?).

Background - Virtual Memory Management Under Unix

In order to understand the concept of shared memory, we should first check how virtual memory is managed on the system.

In order to achieve virtual memory, the system divides memory into small pages each of the same size. For each process, a table mapping virtual memory pages into physical memory pages is kept. When the process is scheduled for running, its memory table is loaded by the operating system, and each memory access causes a mapping (by the CPU) to a physical memory page. If the virtual memory page is not found in memory, it is looked up in swap space, and loaded from there (this operation is also called 'page in').

When the process is started, it is being allocated a memory segment to hold the runtime stack, a memory segment to hold the programs code (the code segment), and a memory area for data (the data segment). Each such segment might be composed of many memory pages. When ever the process needs to allocate more memory, new pages are being allocated for it, to enlarge its data segment.

When a process is being forked off from another process, the memory page table of the parent process is being copied to the child process, but not the pages themselves. If the child process will try to update any of these pages, then this page specifically will be copied, and then only the copy of the child process will be modified. This behavior is very efficient for processes that call `fork()` and immediately use the `exec()` system call to replace the program it runs.

What we see from all of this is that all we need in order to support shared memory, is to some memory pages as shared, and to allow a way to identify them. This way, one process will create a shared memory segment, other processes will attach to them (by placing their physical address in the process's memory pages table). From now all these processes will access the same physical memory when accessing these pages, thus sharing this memory area.

Allocating A Shared Memory Segment

A shared memory segment first needs to be allocated (created), using the `shmget()` system call. This call gets a key for the segment (like the keys used in `msgget()` and `semget()`), the desired segment size, and flags to denote access permissions and whether to create this page if it does not exist yet. `shmget()` returns an identifier that can be later used to access the memory segment. Here is how to use this call:

```
/* this variable is used to hold the returned segment identifier. */
int shm_id;

/* allocate a shared memory segment with size of 2048 bytes,      */
/* accessible only to the current user.                            */
```

```
shm_id = shmget(100, 2048, IPC_CREAT | IPC_EXCL | 0600);
if (shm_id == -1) {
    perror("shmget: ");
    exit(1);
}
```

If several processes try to allocate a segment using the same ID, they will all get an identifier for the same page, unless they defined `IPC_EXCL` in the flags to `shmget()`. In that case, the call will succeed only if the page did not exist before.

Attaching And Detaching A Shared Memory Segment

After we allocated a memory page, we need to add it to the memory page table of the process. This is done using the `shmat()` (shared-memory attach) system call. Assuming 'shm_id' contains an identifier returned by a call to `shmget()`, here is how to do this:

```
/* these variables are used to specify where the page is attached.  */
char* shm_addr;
char* shm_addr_ro;

/* attach the given shared memory segment, at some free position */
/* that will be allocated by the system.                         */
shm_addr = shmat(shm_id, NULL, 0);
if (!shm_addr) { /* operation failed. */
    perror("shmat: ");
    exit(1);
}

/* attach the same shared memory segment again, this time in     */
/* read-only mode. Any write operation to this page using this   */
/* address will cause a segmentation violation (SIGSEGV) signal. */
shm_addr_ro = shmat(shm_id, NULL, SHM_RDONLY);
if (!shm_addr_ro) { /* operation failed. */
    perror("shmat: ");
    exit(1);
}
```

As you can see, a page may be attached in read-only mode, or in read-write mode. The same page may be attached several times by the same process, and then all the given addresses will refer to the same data. In the example above, we can use 'shm_addr' to access the segment both for reading and for writing, while 'shm_addr_ro' can be used for read-only access to this page. Attaching a segment in read-only mode makes sense if our process is not supposed to alter this memory page, and is recommended in such cases. The reason is that if a bug in our process causes it to corrupt its memory image, it might corrupt the contents of the shared segment, thus causing all other processes using this segment to possibly crush. By using a read-only attachment, we protect the rest of the processes from a bug in our process.

Placing Data In Shared Memory

Placing data in a shared memory segment is done by using the pointer returned by the `shmat()` system call. Any kind of data may be placed in a shared segment, except for pointers. The reason for this is simple: pointers contain virtual addresses. Since the same segment might be attached in a different virtual address in each process, a pointer referring to one memory area in one process might refer to a different memory area in another process. We can try to work around this problem by attaching the shared segment in the same virtual address in all processes (by supplying an address as the second parameter to `shmat()`, and adding the `SHM_RND` flag to its third parameter), but this might fail if the

given virtual address is already in use by the process.

Here is an example of placing data in a shared memory segment, and later on reading this data. We assume that 'shm_addr' is a character pointer, containing an address returned by a call to `shmat()`.

```
/* define a structure to be used in the given shared memory segment. */
struct country {
    char name[30];
    char capital_city[30];
    char currency[30];
    int population;
};

/* define a countries array variable. */
int* countries_num;
struct country* countries;

/* create a countries index on the shared memory segment. */
countries_num = (int*) shm_addr;
*countries_num = 0;
countries = (struct country*) ((void*)shm_addr+sizeof(int));

strcpy(countries[0].capital_city, "U.S.A");
strcpy(countries[0].capital_city, "Washington");
strcpy(countries[0].currency, "U.S. Dollar");
countries[0].population = 250000000;
(*countries_num)++;

strcpy(countries[1].capital_city, "Israel");
strcpy(countries[1].capital_city, "Jerusalem");
strcpy(countries[1].currency, "New Israeli Shekel");
countries[1].population = 6000000;
(*countries_num)++;

strcpy(countries[1].capital_city, "France");
strcpy(countries[1].capital_city, "Paris");
strcpy(countries[1].currency, "Frank");
countries[1].population = 60000000;
(*countries_num)++;

/* now, print out the countries data. */
for (i=0; i < (*countries_num); i++) {
    printf("Country %d:\n", i+1);
    printf("  name: %s:\n", countries[i].name);
    printf("  capital city: %s:\n", countries[i].capital_city);
    printf("  currency: %s:\n", countries[i].currency);
    printf("  population: %d:\n", countries[i].population);
}
```

A few notes and 'gotchas' about this code:

1. No usage of `malloc()`.

   Since the memory page was already allocated when we called `shmget()`, there is no need to use `malloc()` when placing data in that segment. Instead, we do all memory management ourselves, by simple pointer arithmetic operations. We also need to make sure the shared segment was allocated enough memory to

accommodate future growth of our data - there are no means for enlarging the size of the segment once allocated (unlike when using normal memory management - we can always move data to a new memory location using the `realloc()` function).

2. Memory alignment.

   In the example above, we assumed that the page's address is aligned properly for an integer to be placed in it. If it was not, any attempt to try to alter the contents of 'countries_num' would trigger a bus error (SIGBUS) signal. further, we assumed the alignment of our structure is the same as that needed for an integer (when we placed the structures array right after the integer variable).

3. Completeness of the data model.

   By placing all the data relating to our data model in the shared memory segment, we make sure all processes attaching to this segment can use the full data kept in it. A naive mistake would be to place the countries counter in a local variable, while placing the countries array in the shared memory segment. If we did that, other processes trying to access this segment would have no means of knowing how many countries are in there.

Destroying A Shared Memory Segment

After we finished using a shared memory segment, we should destroy it. It is safe to destroy it even if it is still in use (i.e. attached by some process). In such a case, the segment will be destroyed only after all processes detach it. Here is how to destroy a segment:

```
/* this structure is used by the shmctl() system call. */
struct shmid_ds shm_desc;

/* destroy the shared memory segment. */
if (shmctl(shm_id, IPC_RMID, &shm_desc) == -1) {
    perror("main: shmctl: ");
}
```

Note that any process may destroy the shared memory segment, not only the one that created it, as long as it has write permission to this segment.

A Complete Example

As a naive example of using shared memory, we collected the source code from the above sections into a file named shared-mem.c. It shows how a single process uses shared memory. Naturally, when two processes (or more) use a single shared memory segment, there may be race conditions, if one process tries to update this segment, while another is reading from it. To avoid this, we need to use some locking mechanism - SysV semaphores (used as mutexes) come to mind here. An example of two processes that access the same shared memory segment using a semaphore to synchronize their access, is found in the file shared-mem-with-semaphore.c.

A Generalized SysV Resource ID Creation - `ftok()`

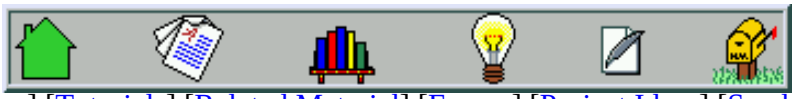
One of the problems with SysV IPC methods is the need to choose a unique identifier for our processes. How can we make sure that the identifier of a semaphore in our project won't collide with the identifier of a semaphore in some other program installed on the system?

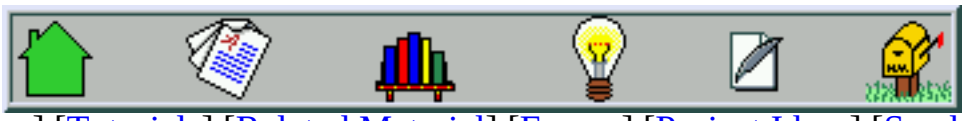
To help with that, the `ftok()` system call was introduced. This system call accepts two parameters, a path to a file and

a character, and generates a more-or-less unique identifier. It does that by finding the "i-node" number of the file (more or less the number of the disk sector containing this file's information), combines it with the second parameter, and thus generates an identifier, that can be later fed to `semget`, `shmget()` or `msgget()`. Here is how to use `ftok()`:

```
/* identifier returned by ftok() */
key_t set_key;

/* generate a "unique" key for our set, using the */
/* directory "/usr/local/lib/ourprojectdir".      */
set_key = ftok("/usr/local/lib/ourprojectdir", 'a');
if (set_key == -1) {
    perror("ftok: ");
    exit(1);
}

/* now we can use 'set_key' to generate a set id, for example. */
sem_set_id = semget(set_key, 1, IPC_CREAT | 0600);
.
.
```

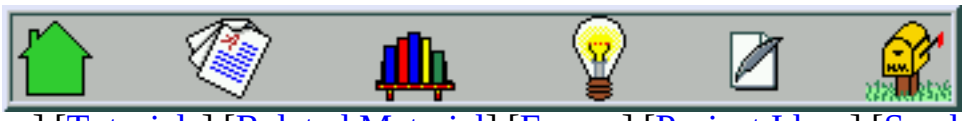
One note should be taken: if we remove the file and then re-create it, the system is very likely to allocate a new disk sector for this file, and thus activating the same `ftok` call with this file will generate a different key. Thus, the file used should be a steady file, and not one that is likely to be moved to a different disk or erased and re-created.

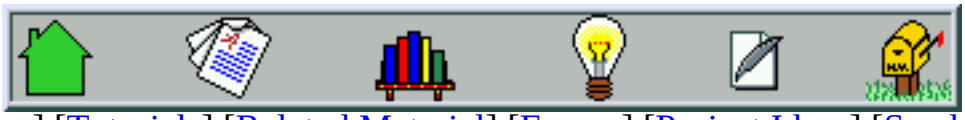




Change Log

- 12-May-2001:
  - Added mutex-unlock call in the various thread pool server examples of the multi-threading tutorial. Without this unlock operation, the hander thread locks out all other handler threads while it services the request, making the pool work serially, and not parallely. (thanks to Mike Hansen for pointing out this crucial bug).
  - Added a note about the need to use '-D_GNU_SOURCE' on current Linux systems, to have the source of some of the programs compiled properly.
  - Add a new essays section to the site.
  - Fixed the 'clear screen' and 'get cursor to top of screen' vt100 sequences in the Writing A VT100-Based PacMan Game project idea.
  - Fixed dangling links in the related material section. (thanks to Olaf for pointing that out).
  - Made some cosmetic/typos fixes to various tutorials (thanks to Markus Wolf for pointing me towards them).





Getting Automatic E-Mail Notification About Site Additions

As another small service, you may choose to get automatic notifications, via e-mail, when new material is added to this site. This will include notifications about new tutorials, new major versions of existing tutorials, new sections added to this web site (naturally, related to Unix programming), etc. To get these notifications (or to stop getting them), fill in the following form. You may use any e-mail address that fits you - it won't be supplied or given to anybody else, and neither will i use it for any purpose other then the one mentioned above.

Important note: - unlike most web sites with similar options, i do intend the UN-subscribe option to work properly. If, for some reason, it does not work for you or you suspect it does not work, please send me e-mail with a request to manually remove you from the list. Don't forget to tell me which e-mail address you are registered under. I will reply to such requests personally.

You want to:

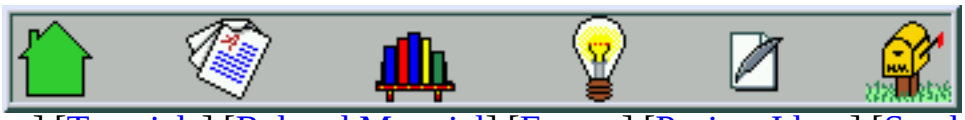
- ○ Be added to the list.
- ○ Have my registration info updated.
- ○ Be removed from this list.

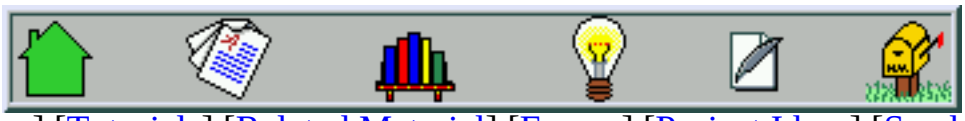
Type of notifications you wish to receive.

- ☐ Any new tutorials added to this site.
- ☐ Any new tutorials as well as major versions of tutorials.
- ☐ New sections added to this web site.
- ☐ Any "interesting" changes made to this site.

Your E-Mail address:

Submit Clear Form



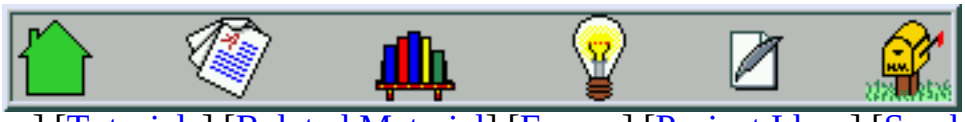


Standard Disclaimer

The material (information and programs) on these web pages are provided AS IS, without any expressed or implied warranty or claim of fitness for a particular purpose. Neither the author nor any contributers shell be liable for any damages done directly or indirectly by using the material presented on this web site, even if they were warned of the possibility of such problems.

Having that said, I still hope this material will be useful for its readers, and help them get on the fast-lane to being productive Unix programmers. If you find a mistake, inaccuracy or other fault with any of the tutorials material, please let me know, via email.

Microsoft© and Microsoft Windows© are registered trademarks of Microsoft Corporation, Ltd. Sun Microsystems© SPARC© and UltraSPARC are registered trademarks, and Java is a trademark of Sun Microsystems, Inc. IBM©, AIX© and RS/6000© are registered trademarks of International Business Machines. All other trademarks mentioned in these pages and tutorials are registered trademarks of their respective owners.



The VIM (Vi IMproved) Home Page

Vim **User** Release	**Vim-5.8**	010531 Everybody please upgrade!
Vim User Release - Patches	Vim-5.8.007	010630
Vim **Developer** Release	Vim-6.0am	010701

[Download Vim] -- [NEWS] -- [**Search** www.vim.org]

What does "Vi IMproved" mean?

FreeBSD: "A vi "workalike", with many additional features."

Linux: "vim - Vi IMproved, a programmers text editor"

VIM is an improved version of the editor "vi", one of the standard text editors on UNIX systems.

VIM adds many of the features that you would expect in an editor: Unlimited undo, syntax coloring, split windows, visual selection, graphical user interface (read: menus, mouse control, scrollbars, text selection), and much much more.

VIM runs on many operating systems:

```
AmigaOS, AtariMiNT, BeOS, DOS, MacOS,
MachTen, OS/2, RiscOS, VMS, and Windows (95/98/NT4/NT5/2000)
```

and, of course, on UNIX in a lot of flavours:

```
A/UX, AIX, BSDI, Convex, DYNIX/ptx, DG/UX, DEC Unix, FreeBSD,
  HPUX, Irix, Linux [Debian, RedHat, Slackware, SuSE,...],
  MacOSX, NetBSD, NEXTSTEP, OpenBSD, OSF, QNX, SCO, Sinix,
    Solaris, SunOS, SUPER-UX, Ultrix, Unixware, Unisys.
```

For those who use Windows - you can use Vim as your editor in other programs, eg many users already use Vim as their editor within VisualStudio. (See the page) For more information see the page on Vim and its OLE interface.

And the best of all: **VIM is FREE**! :-)

Please note: The source code to "vi" is copyrighted. Therefore the system like BeOS, FreeBSD, OpenBSD, and Linux can only give you "clones" of the original vi. So why would you want to use a vi clone? Well, please read about the many reasons to use a vi clone.

By the way, you can discuss customization and development of Vim, and editing with Vim on the Usenet

newsgroup "**comp.editors**" and on several mailing lists.

If you are still unsure about downloading vim then take a look at LinuxCare to read about comments on vim-5.4

[**Search** www.vim.org]

NEWSFLASH

The following is just an excerpt of the News. You can find them all on the News Page.

DATE	NEWS
June 19th	Vim Book Errata List now available: http://www.moolenaar.net/vim_errata.html If you find a mistake in the book that isn't in the list yet, or perhaps an item in the errata list that's wrong, send Bram a message. Preferably use the same format as used on the page.
June 13th	Vim is now a topic on **Wikipedia**, "a collaborative project to produce a complete encyclopedia from scratch": • http://www2.wikipedia.com/wiki.cgi?VIM The idea is that anyone can edit the webpages, so everyone can add info when he feels like it. have fun!
June 4th	Turkish translation of the short description on Vim in six kilobytes. Translated by Kutlay Topatan kutlay@hotmail.com - thanks!
May 25th	Linux Journal, June 2001, "Algorithms in Africa", by Wayne Marshall: http://www2.linuxjournal.com/lj-issues/issue86/4657.html (local copies: HTML and Text) Bram: "It is about the relation between software in third world countries. It's very well written and gives you a good idea of what people in Africa really need. The last section is about Vim in Uganda."
May 20th	Czech translation of the short description on Vim in six kilobytes. Translated by Tomá¹ Znamenáèek tomas.znamenacek@worldonline.cz - thanks!
May 8th	A short overview to the Vim-6.0ae - options and their defaults. Hopefully someone has some use for this.
May 3rd	**An Interview with Vim Authors** - translated to Chinese (GB) by slimzhao@21cn.com
April 23rd	"What is your favorite Tcl programming editor or environment?" Results: • http://www.ajubasolutions.com/poll/poll-results.tcl?poll_id=281
April 20	Vim was added to the "Developer Tools - Editors" section on SoftlandIndia.com, "India's Website for Linux OS and Applications Downloads.": http://www.softlandindia.com/Linux/Editors.htm 
April 11	The **Vim Book** is expected to ship on April 18th. Author: Steve Oualline; ISBN: 0735710015; Pubilisher For more info see page http://www.iccf.nl/click5.html

April 11	Bram: "Vim 6.0 is still a few months away. Don't know how many." [Bram won't release it until he sees that it is good. So stop bugging us about it. ;-)
April 08	To all developers and testers of Vim-6 alpha versions: Bram Moolenaar on vim-dev maillist [010408 15:12]: Although Vim 6.0 is progressing well, there is still a large number of items in the todo list. The question is: Which items should be done before going to a beta release? I would like to hear the opinion of the people on the vim-dev list. Please send me a message with up to five items that you think should still be done before freezing 6.0. Subjects you could consider: - A feature that should really be included in Vim 6.0. - A feature that was included but needs improvement. - Documentation that needs to be written. - Things that were included in a wrong way and need to be changed. Check the todo list ":help todo" to see items that could be included. If you think that Vim 6.0 is ready to be released, just mention that. No need to mention that bugs should be fixed, that will happen anyway. Send the message to me directly. I will send a summary to the list. I must receive the message before May 1 2001. So - take a look at the todo list of todo list of Vim-6.0z, and please limit your list to five items. Then send it to Bram Moolenaar at bram@vim.org.
March 26 March 13	Interesting screenshot: Vim-6.0w with UTF-8 text by Alexey Marinichev lyosha@math.ucr.edu This screenshot shows Vim-6.0w within an xterm from the XFree86-4 distribution. The main window is split vertically, and each subwindow shows text in several fonts/languages: Japanese, Hebrew, Polish, Russian, Thai; English, French, and German.
March 07	"Vim Regular Expressions 101" - a guide to regular expressions (aka "patterns") in Vim. http://physlab.sci.ccny.cuny.edu/%7Eorycc/vim-regex.html written by Oleg Raisky olrcc@scisun.sci.ccny.cuny.edu

February 23	Interesting Macro Set: An (ASCII) Mandelbrot Set Generator written by Linus Akesson lft@df.lth.se: http://www.df.lth.se/~lft/vim/ This macro generates the following output:
	
	`@@@@@@@@@@@@@@@.@@@@@@@@@@@@@@@`
	`@@@@@@@@@MMFFF99.99FFFMM@@@@@@@@`
	`@@@@@@MMMFFFF9fi.if9FFFFMMM@@@@@`
	`@@@@MMMFFFFF9.m...m.9FFFFFMMM@@@`
	`@@@MMMMFFFF9fw.....wf9FFFFMMMM@@`
	`@@MMMMFFF99flw.....wlf99FFFMMMM@`
	`@MMMMMFF99l...........l99FFMMMMM`
	`@MMMMMFfl.............lfFMMMMM`
	`MMMMMMM9.m...........m.9MMMMMM`
	`@MMMMMMF99f...........f99FMMMMMM`
	`@MMMMMMMMFF9fffffffff9FFMMMMMMMM`
	`@@MMMMMMMMMFFFFFFFFFFMMMMMMMMM@`
	`@@@MMMMMMMMMMMMFMMMMMMMMMMMMM@@`
	`@@@@MMMMMMMMMMMMMMMMMMMMMMMMM@@@`
	`@@@@@@MMMMMMMMMMMMMMMMMMMMM@@@@`
	`@@@@@@@@@MMMMMMMMMMMMMMMM@@@@@@@@`

[**Search** www.vim.org]

Frequently Asked Questions (FAQs)

The VIM FAQ will shortly be rewritten. The "style" will be something like this:

FAQ: How to set the default colors for GUI?

[gvim, color setup]

Use the command ":hilight". Example:

```
:hi Normal guibg=white guifg=black
```

FAQ: How do I specify where the gvim window will be placed on my desktop?

[gvim, startup]

Use the "-geom" startup option, specifying the "geometry" as *columns*x*lines*+*offset*+*offset*.

```
gvim -geom 80x65+10+50
```

Comments?

Here are a few FAQs (with answers :-) which you should know:

Q: I have been out of touch with Vim development for a while - what's going on?
A: Take a look at the development page to read about some new features of vim-5.4 or look at the Vim History for a log of all changes.

Q: Whoa - a new version is out - should I test it?
A: Sure, everyone is welcome to test Vim, of course. However, you are expected to know how to handle an alpha/developer version - it is not meant for the faint of heart. If you are just "a user" then you had better leave the testing to others and look for the next release. Yes, these are scarce, but we really do a LOT of testing...

Q: Where's the MacOS version of Vim?
A: You can get the latest binaries for MacOS from the page of Dany St-Amant:

`http://www3.sympatico.ca/dany.stamant/vim/`

For further info look at the page on Vim on MacOS.

Please read all of the VIM FAQ before asking questions. Thanks!

VIM Pages Overview

These pages are currently available:

SEARCH
Search www.vim.org for words.

News and Overview

Short Description -
Vim explained in 6 kilobytes - available in the following languages: Chinese (Mandarin), Chinese (Big5), Czech, Dutch, **English**, Finnish, French, German, Greek, Hungarian, Italian, Japanese, Korean, Polish, Portuguese, Romanian, Russian, Spanish, Swedish, Turkish, Ukrainian.
(Translations welcome! Welsh anyone?)

News
Latest news on Vim, Vim features, and the Vim Pages

Page Tree
An overview to all pages at this site. (Yes, this needs an update...)

Feature Descriptions

Why use Vim?
Why use a vi clone? What sets Vim apart?

Y2K compliancy
Yes, vim *will* work after 1JAN2000. :-)

Distribution

Binaries&Packages
Precompiled versions of vim. (Not for all systems, though.)

Distribution/Downloading
How and where to get the latest Vim.

Mirrors (FTP,WWW)
All FTP and WWW mirrors - for direct access.

Documentation

Documentation Overview
A link list to many (online) documentation about Vim.

Helpfiles (TXT)
Online versions of Vim-5.7's standard documentation (pure ASCII).

Helpfiles (HTML)
Online versions of Vim-5.7's standard documentation converted to HTML for online reading with links.

FAQ
Frequently Asked Questions and Answers.

HOWTOs

Vim Color Editor HOWTO by Alavoor "Al Dev" Vasudevan alavoor@yahoo.com

HowTo Docs on Vim (Index)

Advanced Editing / Pictures

Macros+Mappings
Some useful macro/mapping sets.

Tips+Tricks (Answers)
A collection of answers given by Sven on comp.editors and in emails.

Pictures

Buttons/Icons
Buttons/icons you can use to advertise Vim on your website.

Screenshots
Vim in action on various OSs - and in color!

Communication

Chat Chat with us on IRC!

Mailing Lists and Newsgroup
How and where to discuss Vim and gets answers from.

Development

CVS Server
Latest patches to the developer version.

Developer's Corner
What's going on with current development?

History
The History of Vim - all version release dates.

HOWTO Help
How you can help the Vim community. Feedback, please!

User Pages / Feedback

Users
Links to pages on Vim by Vim users.
Sample config files and current syntax files.

Wishlist
Features that users have requested for future versions.

Miscellaneous

Organization
Who set all of this up? Some background info.

Press
Articles about Vim in the media (magazines and newspapers).

Resources

Languages
Available Language Syntax Files. Links to online versions.

Utilities
Other programs that are useful for coding/editing with Vim

Fun Fun Fun :-)

[Challenges](#)
"Can you do it?" - Some challenges on hacking with Vim.

[Quotes](#)
Quotes of people about Vim.

[**Search** www.vim.org]

VIM Pages - Basic Info

You should know these things about Vim.

Getting started with highly configurable programs usually means that you will have to RTFM a lot and combine possibilities until they give you the setup commands which really make the program *work* for you.

Taking a look at some setup files can usually give you some ideas about the power of the program. So I commented my setup file to let you know about some features of Vim:

Heavily commented setup file to get you started:

```
ca  6K Sven's personal vimrc
ca  6K Sven's personal gvimrc
ca 66K Sven's big vimrc
ca 24K Sven's big vimrc (compressed with gzip)
```

Here's a "minimal setup" of mine:

```
set nocp " :-)
" turn these ON:
set digraph ek hidden ruler sc vb wmnu
" turn these OFF:
set noeb noet nosol
" non-toggles:
set bs=2 fo=cqrt ls=2 shm=at tw=72 ww=<,>,h,l
set comments=b:#,:%,fb:-,n:>,n:)
set list listchars=tab:»·,trail:·
set viminfo=%,'50,\"100,:100,n~/.viminfo
" settings which are the default
" (at least with "nocompatible" anyway):
" set smd sw=8 ts=8
" mappings:
map K     <NUL>
map <C-Z> :shell
map ,F :view    $VIMRUNTIME/filetype.vim
map ,SO :source $VIMRUNTIME/syntax/
map ,V  :view   $VIMRUNTIME/syntax/
" autocommands:
au FileType mail set tw=70
```

```
" some colors:  "white on black"
hi normal   ctermfg=white  ctermbg=black guifg=white  guibg=black
hi nontext  ctermfg=blue   ctermbg=black guifg=blue   guibg=black
" syntax coloring!! :-)
syn on
```

It is also a good idea to print out the "Vim Guide" (by Oleg Raisky):

- http://physlab.sci.ccny.cuny.edu/%7Eorycc/vim-main.html

Let's Talk about Vim

You can chat with us on Vim.
See the Vim Chat Page for more info.

But if you are interested in helping others or
in discussing Vim's current development
then you can join one of the following mailing lists:

VIM Mailing Lists:

```
vim-announce@vim.org          Vim Announcements
         vim@vim.org          Vim Help List
     vim-dev@vim.org          Vim Development List
     vim-mac@vim.org          Vim Development on the Macintosh
vim-multibyte@vim.org         Vim Development of Multibyte Support
vim-fr@yahoogroups.com        Vim for French speaking users
```

NOTE: You cannot send to these mailing lists without subscription.
All lists require subscription
for posting!
And the vim-announce list is merely for announcements of new versions
(so it is pretty much "low volume").

Mailing List Archives:
However, if you do not like subscribing to mailing lists
or if you simply want to take a look at them
to see what's currently going on then you can search the
maillist archives on egroups.com.

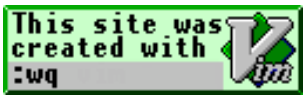
See the page on mailing lists for more info!

News/Usenet:
However, I prefer "talking" about Vim on News (aka Usenet).
It is much easier to use a newsreader to subscribe and
unsubscribe to *newsgroups* than it is with mailing lists.
Furthermore, your messages are automatically archived on
deja.com which
has a powerful search interface for the medium.

Besides, comp.editors has been known for years
and is read by lots of people - especially Vim users.

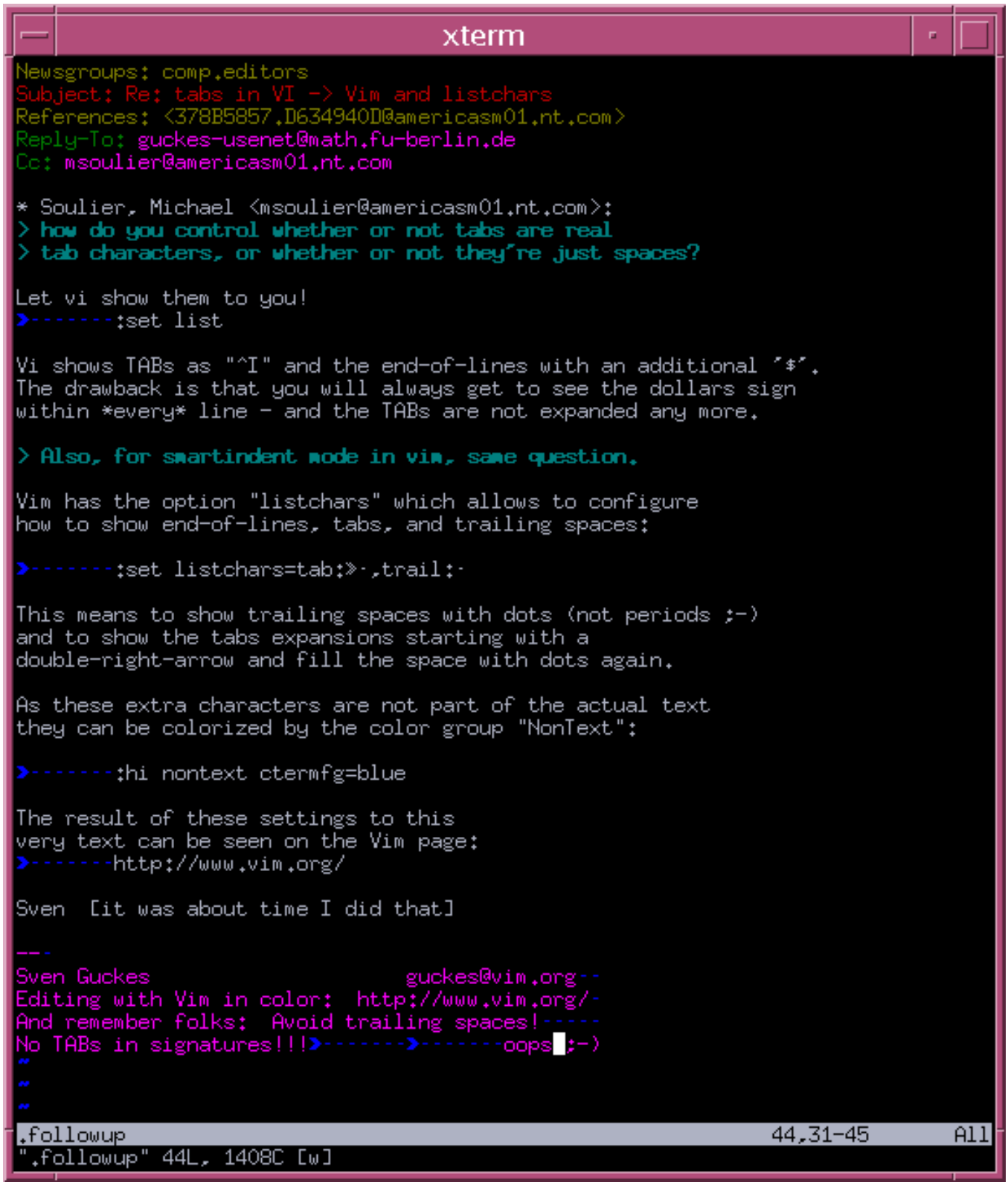
VIM Usenet Newsgroups:

```
        comp.editors :-)
```



Editing Example

Vim colorizes texts according to the *syntax* that defines the "language" the text was written in. Here's an example:



Here I am editing a post to the newsgroup comp.editors that explains the use of the "listchars" option which show you TABs, trailing spaces, and the end-of-line. (Why doesn't every editor allow you to show these?)

The colorization is done automatically by vim (using the description of the syntax file "mail.vim").

By the way, my newsreader is slrn which also works on color terminals (see some pictures).

Would you like to see more screen-shots of editing with Vim? Let me know!

[**Search** www.vim.org]

A Note on Mirrors

The current home address of these pages is

```
http://www.math.fu-berlin.de/~guckes/vim/
```

All other sites are **mirrors**!

So if you cannot see any changes on **http://www.vim.org/** that's because it **is a mirror** which is updated only once a day (well, actually at 2am MET).

A complete list of mirrors is given on

- Distribution Page

If you want to mirror these pages then you are very welcome to do so. Please take a look at the

- HowTo Mirror Page.

[**Search** www.vim.org]

Webpages with VIM

All pages on www.vim.org have been crafted with Vim. The HTML is *very* simple so all browsers should be able to display them - even the text browsers lynx and w3m.

Most HTML is inserted by abbreviations - see my html.vim

[**Search** www.vim.org]

SEE ALSO

There are more sites about Vi and its clones:

- Vi-Editor.org (aka "Vi Pages")

This site has everything else that applies to Vi - not only to Vim. It gives info about everything else that applies to Vi, including other Vi clones. There are lots of links to other documentation about Vi (intro, tips and tricks, the Vi FAQ) and also specialized info eg on the substitution command (aka search+replace).

[**Search** www.vim.org]

Send Feedback!


```
From:     Sven Guckes <guckes@vim.org>
To:       You <reader@vim.org>
Subject:  Feedback, please!

Comments?  Corrections?  Praise?  Typos?
Please send them to me - thanks!  :-)

Sven

--.                                          Powered
Sven Guckes guckes@vim.org
"The wonderful thing about Tiggers
 Is Tiggers are wonderful chaps
 They're loaded with vim and with vigor..."
```

A small note on the signature...

Downloads
Documentation
CPAN
FAQs
Training
Resources
Article Archive
Books
Search
Register/Log in

Columns
P5P Digest
P6P Digest
Off the Wall

O'Reilly Network Technologies

Mailing list report covering July 2 - 9

What goes on in the mind of this Perl Porter?

This Fortnight in Perl 6 (17 - 30 June 2001)
A detailed summary of a recent Perl vs. Java battle, a discussion on the internal API for strings, and much more. [Perl News and Features from www perl com]

Why Not Translate Perl to C?
Mark-Jason Dominus explains why it might not be any faster to convert your code to a C program rather than let the Perl interpreter execute it. [Perl.com]

This Week on p5p 2001/07/02
Module versioning and testing, regex capture-to-variable, and much more. [Perl.com]

Yet Another YAPC Report: Montreal
Schuyler Erle gives a detailed report of all the exciting events at this year's Yet Another Perl Conference in Montreal. By his account, it appears to be an exciting time to be involved with the development of Perl. [Perl.com]

Parse::RecDescent Tutorial

what's new

Perl Runs Sweden's Pension System

Intended only as a quickly developed backup for Sweden's Premium Pension System, the Perl-based application proved to be faster and more promising than the original commercial version--at a fraction of the cost. Read about it here. There are sure to be more Perl success stories, not to mention the State of the Onion Address and Internet Quiz Show, at the Perl Conference 5, July 23-27, 2001, in San Diego, California.

weblogs

news

.NET
Apache
BSD
ONJava.com
Javascript and CSS
Linux
Mac
Mozilla
ONLamp.com
Policy
openp2p.com
Perl
PHP
Python
RSS
Wireless
XML

Parse::RecDescent is a recursive descent parser generator designed to help to Perl programmers who need to deal with any sort of structured data,from configuration files to mail headers to almost anything. It's even been used to parse other programming languages for conversion to Perl. [Perl.com]

The Beginner's Attitude of Perl: What Attitude?
Robert Kiesling says that the Perl Community's attitude towards new users is common fare for Internet development and compared to other lists Perl is downright civil. [Perl.com]

Using CGI::Application
The Common Gateway Interface may well be the backbone of many web applications, but sometimes it can feel dry and monotonous to work with. If you're fed up with `"my $query = CGI->new()"`, Jesse Erlbaum presents a kinder, gentler alternative. [Perl.com]

Resources

A B C
Advocacy Binaries Biology Books and Magazines Business C and Perl CGI Communications Community Compiler Conversion CORBA Core Documentation Courses and Training CPAN

D E F
Data Structures Databases Debugging Documentation Editors Files Finance Functions

G H I
Games Gear Geographical Graphics HTTP Installation

J K L
Java Language Development Larry Wall Lingua Linux Lists

M N 0
Macintosh Mail and USENET News Makefiles Materials Science Mathematics Modules mod_perl Music Net

Parrot? Well, sort of.
Remember Parrot, the April Fool's Joke, where I had Perl and Python joining forces to create a new language? Well, there was a serious side to that little prank. [O'Reilly Network Weblogs]

The Direction of SOAP
SOAP is evolving into new areas of functionality and interoperability. Paul Kulchenko, a speaker at the upcoming Open Source Software Convention, has taken time out to show us where SOAP is going. [O'Reilly Network Weblogs]

recipe of the day

Wednesday: You have a date, either in Epoch seconds or as distinct year, month, etc. values. You want to find out what week of the year, day of the week, day of the month, or day of the year that the date falls on. What to do?

From the Perl Cookbook

Damian Conway in Portland 31 Jul - 3 Aug
[Perl News]

Env-Modulecmd-1.0
[CPAN Uploads]

Apache-ASP-2.19
[CPAN Uploads]

HTML-Macro.1-13
[CPAN Uploads]

DBIx-SearchBuilder-0.40
[CPAN Uploads]

Tie-RangeHash-0.60
[CPAN Uploads]

Newsgroups NeXT Objects Oddities Open Source

P Q R S

Palm Pilot PCL Perl 6 Perl Internals Porting Programming Regular Expressions Releases School Screen I/O Security Sets Solaris Sorting Sound and Audio Statistics Style Guides Sysadmin

T U V

Text Tools Time Tools Troubleshooting Tutorials UNIX User Groups User Interfaces Version Control Systems VMS

W X Y Z

Web Admin Web Development Web/CGI Win32 XML Y2K

off the wall

Quotations from Larry:

Of course, this being Perl, we could always take both approaches. :-)

Perl Versions:
Stable is 5.6.1. Devel is 5.7.1.

GNU Make

What Is GNU Make?

Make is a tool which controls the generation of executables and other non-source files of a program from the program's source files.

Make gets its knowledge of how to build your program from a file called the *makefile*, which lists each of the non-source files and how to compute it from other files. When you write a program, you should write a makefile for it, so that it is possible to use Make to build and install the program.

Capabilities of Make

- Make enables the end user to build and install your package without knowing the details of how that is done -- because these details are recorded in the makefile that you supply.
- Make figures out automatically which files it needs to update, based on which source files have changed. It also automatically determines the proper order for updating files, in case one non-source file depends on another non-source file.

  As a result, if you change a few source files and then run Make, it does not need to recompile all of your program. It updates only those non-source files that depend directly or indirectly on the source files that you changed.
- Make is not limited to any particular language. For each non-source file in the program, the makefile specifies the shell commands to compute it. These shell commands can run a compiler to produce an object file, the linker to produce an executable, `ar` to update a library, or TeX or Makeinfo to format documentation.
- Make is not limited to building a package. You can also use Make to control installing or deinstalling a package, generate tags tables for it, or anything else you want to do often enough to make it worth while writing down how to do it.

Make Rules and Targets

A *rule* in the makefile tells Make how to execute a series of commands in order to build a *target* file from source files. It also specifies a list of *dependencies* of the target file. This list should include all files (whether source files or other targets) which are used as inputs to the commands

in the rule.

Here is what a rule looks like:

```
target:   dependencies ...
          commands
          ...
```

When you run Make, you can specify particular targets to update; otherwise, Make updates the first target listed in the makefile. Of course, any other target files needed as input for generating these targets must be updated first.

Make uses the makefile to figure out which target files ought to be brought up to date, and then determines which of them actually need to be updated. If a target file is newer than all of its dependencies, then it is already up to date, and it does not need to be regenerated. The other target files do need to be updated, but in the right order: each target file must be regenerated before it is used in regenerating other targets.

Advantages of GNU Make

GNU Make has many powerful macro features for use in makefiles, beyond what other Make versions have. It can also regenerate, use, and then delete intermediate files which need not be saved.

GNU Make also has a few simple features that are very convenient. For example, the `-o` *`file`* option which says ``pretend that source file *file* has not changed, even though it has changed." This is extremely useful when you add a new macro to a header file. Most versions of Make will assume they must therefore recompile all the source files that use the header file; but GNU Make gives you a way to avoid the recompilation, in the case where you know your change to the header file does not require it.

However, the most important difference between GNU Make and most versions of Make is that GNU Make is free software.

About GNU Make

Full documentation for GNU make is found in the Make manual, which is included in the GNU Make distribution. You can also buy printed copies of the Make manual from the Free Software Foundation; the money we raise goes to fund free software development.

Alternatively you can browse the online documentation available at www.gnu.org/manual/make/index.html

You can download GNU `make` from the GNU FTP site ftp.gnu.org or one of its mirrors. You can also obtain it by buying the GNU Source Code CD-ROM from the Free Software Foundation; this too helps fund free software development.

GNU `make` was written by Richard Stallman and Roland McGrath.

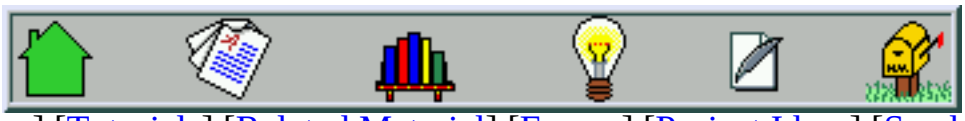
Makefiles And Conventions

We have developed conventions for how to write Makefiles, which all GNU packages ought to follow. It is a good idea to follow these conventions in your program even if you don't intend it to be GNU software, so that users will be able to build your package just like many other packages, and will not need to learn anything special before doing so.

These conventions are found in the chapter ``Makefile conventions'' (147 k characters) of the GNU Coding Standards (147 k characters).

Return to GNU's home page.

Updated: 6 Sep 2000 neelakanth



v1.0.0

Why Do Universities Fail Teaching Software Engineering?

Table Of Contents:

Abstract

Many universities and other institutions try to teach software engineering concepts to their students, either as part of their regular programming courses, in courses designed to teach just software engineering, or as a set of such courses.

In this short essay, I'll supply a loose definition of "Software Engineering", show why it is so hard (or impossible) teaching it to inexperienced programmers, and try to illustrate a few methods that may be used to still put some sense of engineering into inexperienced programmers.

What Is Software Engineering?

Software engineering is a methodological process of writing software, in a manner that can be repeated for various software projects in a high level of accuracy, and produce good software, under given time and budget constraints.

Lets explore this definition a bit. The methodology means that the software creation process is

broken down into several steps, each of which is self-contained, and can be described easily, without using "magical" terms, as is common in the software industry ("i didn't know where to start, so i just hacked down something, and somehow it worked").

For example, one may break software creation into steps such as "requirements specification", "architectural design", "interface design", "detailed design", "implementation", "integration" and "testing". Each of these steps can be properly defined (e.g. "requirements specification is a step in which we define the set of functionalities our software will support, without defining yet how it will support them, or how any kind of user-interface would look like").

Good software would imply that the software does what it is supposed to do, that it has few bugs, that it can be easily maintained, and that it is relatively cheap to extend and add new features without introducing many new bugs.

The software doing what it is supposed to do might sound trivial, but in many cases the original requirements get massively cut down because the software fails to perform various operations properly. easy maintenance would mean that it would be easy for a new programmer to learn how the software works and how to fix bugs in it, and that fixing bugs does not require rewriting complete parts of the software. Easy extension would mean, for example, that the software can be easily ported to new platforms, can easily have its user interface replaced (text GUI to graphical GUI, or web GUI) without rewriting the rest of the system, and so on.

Finally, every software development project has some limits on the amount of time until it has to be ready, and the amount of money that can be spent on the software. The engineering process should make it possible to supply a rather accurate forecast for the time and effort it will take to develop different modules of the software, using different designs. The time limits often come on expense of future maintainability and expend-ability of the software, and these issues need to be weighed against each other, and weighted appropriately, depending on the software requirements.

Software Engineering And Students

From the above description, one can see that software engineering introduces a lot of overhead to software development, and that this overhead might only be worthwhile for large software development. Using such process for small class exercises of few hundreds lines of code is an overkill and makes students despise or even ridicule the idea of software engineering.

Another problem with software engineering, is that it can't be practiced without understanding that small decisions have big implications on large programs. For example, suppose that a method of a communications class returns a pointer to an object it has allocated. Who should be in charge of freeing that memory? The caller of the method, or the object that initially allocated the memory? What happens if in one place its easier to have the caller free the memory, and in another place its easier to make the allocating object free the memory? an inexperienced programmer would most likely do the easier thing in each case, claiming that the simplicity of the code is important, and would overlook the importance of consistency. For a small program,

it won't do much harm - since the program is written in a short time, the programmer can remember who should free the memory in each case. Even if they don't, and have a memory leak, they are not going to run this program long enough to notice - once it works, it gets dumped.

The same little decision would have a much bigger impact on a large software, that is supposed to last. A large software tends to have hundreds or thousands of classes, and not each one of them is used very often. If you had consistent rules (e.g. "the object allocating memory is always in charge of freeing it"), it will be easy to remember when to free allocated memory, and thus having less memory leaks and memory corruptions to chase.

More On Students

As we saw, software engineering requires large projects to make sense. It also requires experience to make sense. In particular, bad experience - if you write software in the right way, you don't get to see how wrong badly written software can get, and thus don't learn to appreciate the right ways. Thus, part of learning software engineering is achieved by seeing how lack of it can hurt large software projects. For students "a large project" might be something they have written in one semester, in which they also studied a few other courses. Given a software made of a few thousands lines of code, or a few tens of classes, inexperienced programmers consider the software to be a large project. Later on, when they get out to work in the industry, they will begin to realize those were in fact rather small projects. In such small projects, lack of proper engineering won't usually reveal the problems it causes, since those projects only need to work once or twice (when demonstrating them to the teacher) and don't get developed further afterwards.

Another aspect of software engineering is designing the software to be easy to maintain. Since most student projects don't get tested by end-users users, many of their bugs remain hidden, and thus the programmer isn't exposed to the amount of time it would take to debug and fix it until it becomes on the level of usable release software - something which often would take more then the amount of time it took to originally write the code and make it run. The original developer of the code tends not to stress-test their code in its weak spots, since they subconsciously know it'll cause the software to crash or malfunction - something which will require more work, that isn't really appreciated when the project is graded. Thus, the programmer cannot see how their software design affected the ability to isolate bugs in the code, and get the false impression that their design generated bug-free software.

Another aspect of learning about software engineering, is seeing how it affects the life cycle of a software project. Sometimes, experienced software engineers make decisions that look like small neat-peaking to the naked eye. Only after a while, as the project evolves to different directions, these original decisions begin to make sense to an inexperienced programmer. Some people will claim that this ability to see the need for those small decisions in advance comes with experience - and yet, this is what software engineering strives for - the ability to make these decisions using a methodological process, that generates repeated success.

Finally, there are students that understand the rules of thumb, and believe that they might be useful, but still prefer not using them for their exercises on the grounds that understanding is enough, and when they get involved in large projects, they will know how to deal with them. The problem is that by the time they get to working on large projects, they might gather bad habits and find it hard to free themselves of these bad habits. They also overlook the little things that can only be achieved with real practice, and thus will stay behind fellow students, who have spent a few years of their undergraduate studies to actually apply these rules, and gained some experience, and thus more insight.

So the conclusion we come to is that you need good familiarity and hands-on involvement in large and lasting software projects, both successful and failures, in order to grasp the essence of software engineering, and appreciate it. And in university environments (or various programming courses, for that matter) the ability to participate in such activities is rather limited.

More On Universities

What do universities do in order to try and teach software engineering? One thing they do is try to teach it as a set of rules of thumb, hoping that students will follow them, even if only because they are being enforced somehow when grading exercises and exams. The problem with this approach is that often the people who do the actual grading are graduate students, who themselves haven't had the chance to grab the concept of software engineering (especially if they entered graduate school directly after finishing undergraduate school).

Even if some of the teachers, or teaching assistants, do have experience with large and lasting software projects, often their students don't know about that, and hence don't trust them. When you don't trust someone, you don't listen to advise they give you if you cannot see an immediate benefit (and in software engineering, there is no benefit for very small exercises). Thus, the students actually tend to ignore their teachers' rules of thumb, seeing them as a burden, and this causes more damage then if those rules were never given in the first place.

At other times, there might be a good software engineering teacher, that indeed has experience in the field, and tries to show some real life examples to their students. These efforts, however, might be lost if the students don't get their own hands-on experience with such software. Learning software engineering on a theoretical bases, and using small code examples (since there is no time to really delve into large code examples) makes no sense, except for people who already understand software engineering in the first place - kind of a chicken-and-egg problem.

How To Make It Work?

After seeing what is needed to make students appreciate software engineering, we might as well spell out a few things that will make teaching it possible. First, we need to have accredited teachers. These teachers may either be people with past or current experience in the industry, that can use it to back their claims to students. They could also be people who participated in large academic projects, that got enough credit for being large and lasting. A good example would be the work done in the MIT university, and their Athena project (see also A review of - MIT project Athena: a model for distributed campus computing). Another good example is the work done at the Washington university in Seattle, by the Distributed Object Computing (DOC) group. There exist various other such examples. The important factor is that they are large projects, involve quite a few staff members (including graduate students), and last for quite a few years, and thus carry a scope similar to that of large industrial projects.

It is also important that the students will know that their teachers have that experience. This is not to be used as a method of bragging, but rather to assure the students that their teacher is not just talking about theoretical software engineering methods; That the teacher has actually applied them, and can show them good, real-life examples, of why these methods are useful, and should be practiced even for smaller projects.

Carrying out large projects by university staff members is also good as it allows graduate students to participate in such projects, and thus be more credible to serve as teaching assistants in software engineering related courses. With good project management, it is also possible to allow undergraduate students to take part in such projects, and witness, from first hand, the complexity of such a project. When they have to delve into code created by other programmers, possibly code that is 2-3 years old, they will learn to appreciate how hard it is to get into code parts that weren't properly engineered, and how relatively easy it is to get into parts that were properly engineered. And having specific parts of the code that are badly engineered on purpose, would serve the teaching goal quite well.

Of-course, getting students involved in large software projects should be done gradually. At first, they may be introduced to small modules, learn them, and be guided in making small changes to them. It is a very useful quality to be able to delve into existing source bases, and inexperienced programmers often find it hard to do. At later phases, these students will be able to write new classes or modules. Getting credit for such projects will be more desirable then letting these students design their own software in a software project course, that will turn out to be small (they don't have enough time to work on it, and usually only 2-3 of them work on the code of such project) and having to create software that will most likely not last, and not be a part of a lasting project.

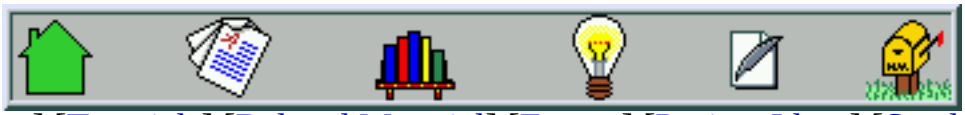
Financing Academic Software Projects

One of the major problems with carrying large software projects in universities is financing them. You need more equipment then with theoretical research, more system management staff, and more researchers then for a theoretical research.

The equipment needed is often not so hard to get as a donation from large hardware manufacturers and resellers. They already have such donation relationship with universities, sometimes donating full labs for students to work on, in a hope that these students will get used to their development environments, and endorse them when they get some influence in their future working place.

Another option is carrying software research projects that are useful for large industrial companies. Showing these companies how this research can help them, it is possible to convince them to sponsor such projects. The fact that financing a project in which graduate students and undergraduate students perform large parts of the work, would be cheaper then financing it inside the industry, it might look appealing to industrial companies. The TAO project carried by the DOC group at the Washington university and university of California, is a good example of such a relationship.

Another major problem is attracting good software engineers that both wish to carry out research work in their field, and have the skills to manage large projects at the same time. The success of such a project in fact often relies on one or more such enthusiast leaders, that carries a good reputation in the academic field, as well as in the industry. It would be easier to attract such people to the academy, if they know they will get a supportive environment, and financing to projects that will seem feasible and are personally appealing to them. Sometimes, it does not require paying them better then in the industry. The fact that they get more freedom, and without the pressure of marketing personnel, would be enough to attract a few of them to moving to the academic world.



MIT Project Athena

In May 1983, MIT announced the establishment of a five-year program to explore new, innovative uses of computing in the MIT curriculum. The MIT faculty was concerned that too little was being done to integrate the new computational technology into the undergraduate educational experience. Project Athena, as the program was called, arose from this concern.

Project Athena's mandate was to explore diverse uses of computing and to build the base of knowledge needed for a longer term strategic decision about how computers fit into the MIT curriculum. In January of 1988, Project Athena was granted a three-year extension to the original five-year program, and on June 30, 1991, Project Athena came to an end. But the fruit of Project Athena -- the Athena service environment -- was adopted as MIT's academic computing infrastructure, with plans to extend it beyond the educational sphere, into the research and administrative activities of the Institute.

The product of eight years of research, the Athena environment provides computing resources to nearly 14,000 users across the MIT campus through a vast system of 1,300 computers in more than 40 clusters, private offices, and machine rooms, all connected to the campus network.

Athena users have access to software to help them write papers, create graphs, analyze data, communicate with their colleagues, play games, and perform countless other tasks, as well as software designed specifically for classwork.

Athena has pervaded campus life. At last count, 99% of MIT undergraduates and 81% of MIT graduate students had Athena accounts. On a typical day, over 6,000 different users access their personal files and various software packages on the system.

A review of -

MIT project Athena: a model for distributed campus computing

by George A. Champine
Digital Press, 1991. 282 pages

Distributed computing is one of those "year" concepts in computing. There was the year of the personal computer (now in its second decade) and the year of desktop publishing. I remember about five years of the local area network. Like these phenomena, distributed computing is one that lots of computer people think will become a landmark. If 1992 isn't named the year of distributed computing I'll be surprised.

What is distributed computing? Here's the general idea. You can build a computing environment in three ways. First, you can have a bunch of big mainframe computers in an air-conditioned machine room somewhere and let everybody connect to them from dumb terminals. This is called time sharing because you share the computer's time and attention with a few, a few tens, or a few hundreds of other users. The year of time sharing was somewhere back in the early '70s. Time sharing is no fun because you don't have much control over those mainframes and they're too distracted by other users to give you the service you need. You do time sharing on the mainframe's terms, not your own.

The second way to compute is with desktop computers. You give everyone a computer and she or he gets its undivided attention. It's great. One user, one computer; you pick the flavor you want and you compute more-or-less on your own terms. As I mentioned, the year of the personal computer started big in 1980 and hasn't quit yet. Personal computing is all right, but it's lonely. If you want to do something collaborative on the computer you have to make all kinds of awkward arrangements, like exchanging floppy disks or keying in arcane commands to do file transfer over a network. To get some human interaction into your computing, through electronic mail and conferencing, you may even wind up squandering the power of your PC by using it as a terminal to a time sharing system!

Enter distributed computing, option three. It's got the best of both worlds. You put powerful personal computers on every desk, but link them with a high-speed network. For every ten or a hundred desktop machines, depending on what you're doing, you add a server computer to the network. The servers each give many users a place to park shared software, data files, text files, mail, and so on, but nobody computes there. You compute on your desktop machine, whatever kind you choose. The server computers spend their time keeping track of who owns what and making sure your stuff is there when you need it. Now you've got computing on your own terms, the power's on your desktop under your control, you're actually using it all, and you're connected to other people, other software, and other information.

Champine's book gives us a window into the development of Athena, the distributed computing

environment at MIT. Project Athena has been in the works since 1982. It resulted from MIT's need to improve instructional computing on campus, and its desire to become a national leader in applying technology to scholarship. By 1990 ($100 million later) Athena was serving 10,500 users on 1,300 workstations. Ninety-six percent of undergraduates and 60% of graduates used Athena, each for an average of eight hours a week.

If you're thinking of developing a distributed computing environment (DCE), and you're not already plugged into developments in Project Athena, you really ought to take a look at this book. It will introduce you to many of the complexities of DCE development and suggest tools and other resources for coping with them. It includes a hefty bibliography, though many of the sources cited are unpublished MIT reports. It would have been helpful (and fitting) if these reports had been put online and Champine had directed us to them.

The book is in four sections: Development, Pedagogy, Technology, and Administration. Each has a distinct flavor. Development briefly describes how MIT got IBM and Digital Equipment Corporation to commit to a total of $50 million in grants and to collaborate with MIT - and to do so cooperatively, with a minimum of corporate competition. And it explains the Athena vision of an environment in which a student can find a desktop computer anywhere on campus, identify herself to it, and have full access to all the software, information, and services she needs, including her own personal text and data files and software. In his book, Champine calls this "coherence." Among DCE computer wizards this vision is summed up in a line from the 1984 cult film The Adventures of Buckaroo Banzai Across the Eighth Dimension: "...wherever you go, there you are!"

The Pedagogy section contains descriptions of many instructional software systems written for the Athena environment. These systems, the Athena courseware, were written by MIT faculty or by students under their supervision. The cost was staggering. Faculty estimated it took 100 to 200 hours of programming for each hour of classroom use. The courseware content ranges from creative writing to thermodynamics. Champine's descriptions emphasize the value of the Athena workstations - high- powered IBM or Digital desktop computers - in presenting information to students in an interactive, graphical way. With careful reading, these descriptions are compelling; one can only wish that Champine had made this part of his book more valuable by including a few printed graphics. The entire book is under-illustrated, but this section suffers most.

The Technology section is where most of the action is. The writing is ponderously descriptive, much like a dry prose flowchart, and is backed up only by simple block diagrams - boxes and arrows. Nevertheless these chapters give the technically inclined reader a glimpse of the problems MIT, DEC, and IBM faced in integrating dissimilar computers into a seamless network, providing a uniform set of tools across all workstation types, handling data security while ensuring easy access, and scaling the system for tens of thousands of users. The section also includes a discussion of Athena's still aborning multi- media workstations.

In the Administration section Champine details the financial and administrative keys to Athena's success. He also devotes a chapter to assessment of that success from many points of view. This information is telling. Athena serves most MIT students, but only a fraction of MIT faculty. Those it serves it serves well; they can't imagine life without it. Those it doesn't serve

seem appalled at the system's expense; many millions of dollars of MIT money went into the project. Worse, many instructors can't use Athena because, in one currency or another, they can't afford to develop courseware for it. One reaction common to user and non-user alike is that Athena would be much more useful if it ran on everyday microcomputers, like IBM PS/2s and Macintoshes, instead of expensive, inscrutable UNIX workstations.

All in all, this is a sober book - what you might expect from Digital Press, house organ of Digital Equipment Corporation. But Athena has many wildly successful components. Athena's Kerberos authentication server is becoming a standard element of distributed computing environments. The Athena File System is an attractive alternative for many DCE developers. And, as an appendix describes, five other universities have successfully adapted Athena to their needs. Champine views the project as goddess Athena herself might have, grey-eyed and aloof (mostly). But don't let the book's dark-suited tone fool you. Athena is worth studying. Whether it touches you directly or not, MIT's Project Athena (and its half-mortal progeny) will change the way you compute in years to come.

Reviewed by Mark C. Sheehan, published in the February 1992 issue of Database magazine, a publication of Online Inc. (Note: This is a copy of my original manuscript. Minor changes made by the publisher are not reflected here.)

Distributed Object Computing (DOC) Group

[Electrical and Computer Engineering University of California, Irvine]

616E Engineering Tower
University of California, Irvine
Irvine, CA 92697-2625
TEL (949) 824-1901
FAX (949) 824-3408

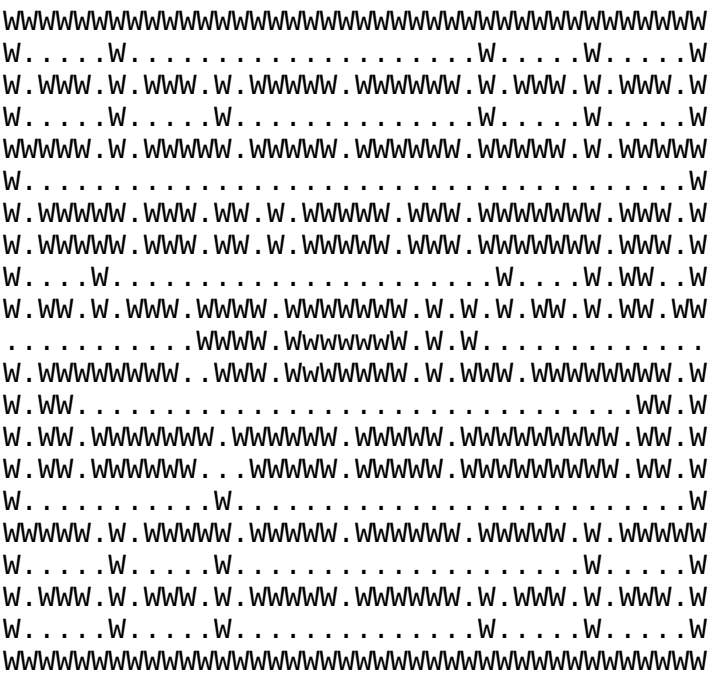
[Department of Computer Science Washington University]

Bryan Hall, Room 503
One Brookings Drive
[St. Louis, Missouri] 63130-4899
TEL (314) 935-4215
FAX (314) 935-7302

The Distributed Object Computing (DOC) Group is a distributed research consortium consisting of the Center for Distributed Object Computing in the Computer Science department at Washington University and the Laboratory for Distributed Object Computing in the Electrical and Computer Engineering department at the University of California, Irvine. In addition, the DOC Group also includes members at Siemens ZT in Munich, Germany, Bell Labs in Murray Hill, New Jersey, and OCI in St. Louis, MO. The purpose of the DOC group is to support advanced R&D on distributed object computing middleware using an open source software development model. This model allows academics, developers, and end-users to participate in leading-edge R&D projects driven by the free market of ideas, requirements, and resources.

Back to Douglas C. Schmidt's home page.

Last modified 19:09:22 CST 07 January 2001



About this FAQ

$Id: rawfaq.texi,v 1.37 2000/09/01 06:34:57 andrew Exp $

This FAQ was originally begun by Patrick Horgan in May 1996; I took it over after it had been lying idle for several months. I've reorganised it a bit and added some stuff; I still regard it as `under development'. Comments, suggestions, additions, corrections etc. should be sent to the maintainer at: <andrew@erlenstar.demon.co.uk>.

A hypertext version of this document is available on the WWW. The home site is located at http://www.erlenstar.demon.co.uk/unix/faq_toc.html. A US mirror site is available at http://www.whitefang.com/unix/faq_toc.html.

This document is available by FTP from the news.answers archives at rtfm.mit.edu and its many mirror sites worldwide. The official archive name is `unix-faq/programmer/faq'. Sites which also archive *.answers posts by group should also carry the file under the `comp.unix.programmer' directory.

Other sources of information are not listed here. You can find pointers to other FAQs, books, source code etc. in the regular [READ ME FIRST] posting that should appear weekly in comp.unix.programmer. Administrivia regarding newsgroup conduct, etc., are also found there; I want to reserve this document specifically for technical Q's and A's. All contributions have been edited by the maintainer, therefore any errors or omissions are my responsibility rather than that of the contributor.

This FAQ is now maintained as Texinfo source; I'm generating a raw text version for Usenet using the `makeinfo` program, and an HTML version using `texi2html`.

List of contributors in no particular order:

Andrew Gierth	`<andrew@erlenstar.demon.co.uk>`
Patrick J. Horgan	*withheld*
Stephen Baynes	`<stephen.baynes@soton.sc.philips.com>`
James Raynard	*withheld*
Michael F. Quigley	*withheld*

Ken Pizzini	*withheld*
Thamer Al-Herbish	*withheld*
Nick Kew	<nick.kew@pobox.com>
Dan Abarbanel	*withheld*
Billy Chambless	<billy@cast.msstate.edu>
Walter Briscoe	<walter@wbriscoe.demon.co.uk>
Jim Buchanan	<jbuchana@buchanan1.net>
Dave Plonka	<plonka@doit.wisc.edu>
Daniel Stenberg	*withheld*
Ralph Corderoy	<ralph@inputplus.demon.co.uk>
Stuart Kemp	*withheld*
Sergei Chernev	<ser@nsu.ru>
Bjorn Reese	*withheld*
Joe Halpin	<jhalpin@nortel.ca>
Aaron Crane	<aaronc@pobox.com>
Geoff Clare	<gwc@root.co.uk>

Go to the first, previous, next, last section, table of contents.

---

1. Process Control

1.1 Creating new processes: fork()

1.1.1 What does fork() do?

```
#include <sys/types.h>
#include <unistd.h>

pid_t fork(void);
```

The `fork()` function is used to create a new process from an existing process. The new process is called the child process, and the existing process is called the parent. You can tell which is which by checking the return value from `fork()`. The parent gets the child's pid returned to him, but the child gets 0 returned to him. Thus this simple code illustrate's the basics of it.

```
pid_t pid;

switch (pid = fork())
{
case -1:
    /* Here pid is -1, the fork failed */
    /* Some possible reasons are that you're */
    /* out of process slots or virtual memory */
    perror("The fork failed!");
    break;

case 0:
    /* pid of zero is the child */
    /* Here we're the child...what should we do? */
    /* ... */
    /* but after doing it, we should do something like: */
    _exit(0);

default:
    /* pid greater than zero is parent getting the child's pid */
    printf("Child's pid is %d\n",pid);
}
```

Of course, one can use `if()... else...` instead of `switch()`, but the above form is a useful idiom.

Of help when doing this is knowing just what is and is not inherited by the child. This list can vary

depending on Unix implementation, so take it with a grain of salt. Note that the child gets *copies* of these things, not the real thing.

Inherited by the child from the parent:

- process credentials (real/effective/saved UIDs and GIDs)
- environment
- stack
- memory
- open file descriptors (note that the underlying file positions are shared between the parent and child, which can be confusing)
- close-on-exec flags
- signal handling settings
- nice value
- scheduler class
- process group ID
- session ID
- current working directory
- root directory
- file mode creation mask (umask)
- resource limits
- controlling terminal

Unique to the child:

- process ID
- different parent process ID
- Own copy of file descriptors and directory streams.
- process, text, data and other memory locks are NOT inherited.
- process times, in the tms struct
- resource utilizations are set to 0
- pending signals initialized to the empty set
- timers created by timer_create not inherited
- asynchronous input or output operations not inherited

1.1.2 What's the difference between fork() and vfork()?

Some systems have a system call `vfork()`, which was originally designed as a lower-overhead version of `fork()`. Since `fork()` involved copying the entire address space of the process, and was therefore quite expensive, the `vfork()` function was introduced (in 3.0BSD).

However, since `vfork()` was introduced, the implementation of `fork()` has improved drastically, most notably with the introduction of `copy-on-write', where the copying of the process address space is transparently faked by allowing both processes to refer to the same physical memory until either of them modify it. This largely removes the justification for `vfork()`; indeed, a large proportion of systems now lack the original functionality of `vfork()` completely. For compatibility, though, there may still be a

`vfork()` call present, that simply calls `fork()` without attempting to emulate all of the `vfork()` semantics.

As a result, it is *very* unwise to actually make use of any of the differences between `fork()` and `vfork()`. Indeed, it is probably unwise to use `vfork()` at all, unless you know exactly *why* you want to.

The basic difference between the two is that when a new process is created with `vfork()`, the parent process is temporarily suspended, and the child process might borrow the parent's address space. This strange state of affairs continues until the child process either exits, or calls `execve()`, at which point the parent process continues.

This means that the child process of a `vfork()` must be careful to avoid unexpectedly modifying variables of the parent process. In particular, the child process must **not** return from the function containing the `vfork()` call, and it must **not** call `exit()` (if it needs to exit, it should use `_exit()`; actually, this is also true for the child of a normal `fork()`).

1.1.3 Why use _exit rather than exit in the child branch of a fork?

There are a few differences between `exit()` and `_exit()` that become significant when `fork()`, and especially `vfork()`, is used.

The basic difference between `exit()` and `_exit()` is that the former performs clean-up related to user-mode constructs in the library, and calls user-supplied cleanup functions, whereas the latter performs only the kernel cleanup for the process.

In the child branch of a `fork()`, it is normally incorrect to use `exit()`, because that can lead to stdio buffers being flushed twice, and temporary files being unexpectedly removed. In C++ code the situation is worse, because destructors for static objects may be run incorrectly. (There are some unusual cases, like daemons, where the *parent* should call `_exit()` rather than the child; the basic rule, applicable in the overwhelming majority of cases, is that `exit()` should be called only once for each entry into `main`.)

In the child branch of a `vfork()`, the use of `exit()` is even more dangerous, since it will affect the state of the *parent* process.

1.2 Environment variables

1.2.1 How can I get/set an environment variable from a program?

Getting the value of an environment variable is done by using `getenv()`.

```
#include <stdlib.h>

char *getenv(const char *name);
```

Setting the value of an environment variable is done by using `putenv()`.

```
#include <stdlib.h>

int putenv(char *string);
```

The string passed to putenv must *not* be freed or made invalid, since a pointer to it is kept by `putenv()`. This means that it must either be a static buffer or allocated off the heap. The string can be freed if the environment variable is redefined or deleted via another call to `putenv()`.

Remember that environment variables are inherited; each process has a separate copy of the environment. As a result, you can't change the value of an environment variable in another process, such as the shell.

Suppose you wanted to get the value for the `TERM` environment variable. You would use this code:

```
char *envvar;

envvar=getenv("TERM");

printf("The value for the environment variable TERM is ");
if(envvar)
{
    printf("%s\n",envvar);
}
else
{
    printf("not set.\n");
}
```

Now suppose you wanted to create a new environment variable called `MYVAR`, with a value of `MYVAL`. This is how you'd do it.

```
static char envbuf[256];

sprintf(envbuf,"MYVAR=%s","MYVAL");

if(putenv(envbuf))
{
    printf("Sorry, putenv() couldn't find the memory for %s\n",envbuf);
    /* Might exit() or something here if you can't live without it */
}
```

1.2.2 How can I read the whole environment?

If you don't know the names of the environment variables, then the `getenv()` function isn't much use. In this case, you have to dig deeper into how the environment is stored.

A global variable, `environ`, holds a pointer to an array of pointers to environment strings, each string in the form `"NAME=value"`. A `NULL` pointer is used to mark the end of the array. Here's a trivial program to print the current environment (like `printenv`):

```
#include <stdio.h>

extern char **environ;

int main()
```

```
{
    char **ep = environ;
    char *p;
    while ((p = *ep++))
        printf("%s\n", p);
    return 0;
}
```

In general, the `environ` variable is also passed as the third, optional, parameter to `main()`; that is, the above could have been written:

```
#include <stdio.h>

int main(int argc, char **argv, char **envp)
{
    char *p;
    while ((p = *envp++))
        printf("%s\n", p);
    return 0;
}
```

However, while pretty universally supported, this method isn't actually defined by the POSIX standards. (It's also less useful, in general.)

1.3 How can I sleep for less than a second?

The `sleep()` function, which is available on all Unixes, only allows for a duration specified in seconds. If you want finer granularity, then you need to look for alternatives:

- Many systems have a function `usleep()`
- You can use `select()` or `poll()`, specifying no file descriptors to test; a common technique is to write a `usleep()` function based on either of these (see the comp.unix.questions FAQ for some examples)
- If your system has itimers (most do), you can roll your own `usleep()` using them (see the BSD sources for `usleep()` for how to do this)
- If you have POSIX realtime, there is a `nanosleep()` function

Of the above, `select()` is probably the most portable (and strangely, it is often much more efficient than `usleep()` or an itimer-based method). However, the behaviour may be different if signals are caught while asleep; this may or may not be an issue depending on the application.

Whichever route you choose, it is important to realise that you may be constrained by the timer resolution of the system (some systems allow very short time intervals to be specified, others have a resolution of, say, 10ms and will round all timings to that). Also, as for `sleep()`, the delay you specify is only a *minimum* value; after the specified period elapses, there will be an indeterminate delay before your process next gets scheduled.

1.4 How can I get a finer-grained version of alarm()?

Modern Unixes tend to implement alarms using the `setitimer()` function, which has a higher resolution and more options than the simple `alarm()` function. One should generally assume that `alarm()` and `setitimer(ITIMER_REAL)` may be the same underlying timer, and accessing it both ways may cause confusion.

Itimers can be used to implement either one-shot or repeating signals; also, there are generally 3 separate timers available:

`ITIMER_REAL`
: counts real (wall clock) time, and sends the `SIGALRM` signal

`ITIMER_VIRTUAL`
: counts process virtual (user CPU) time, and sends the `SIGVTALRM` signal

`ITIMER_PROF`
: counts user and system CPU time, and sends the `SIGPROF` signal; it is intended for interpreters to use for profiling.

Itimers, however, are not part of many of the standards, despite having been present since 4.2BSD. The POSIX realtime extensions define some similar, but different, functions.

1.5 How can a parent and child process communicate?

A parent and child can communicate through any of the normal inter-process communication schemes (pipes, sockets, message queues, shared memory), but also have some special ways to communicate that take advantage of their relationship as a parent and child.

One of the most obvious is that the parent can get the exit status of the child.

Since the child inherits file descriptors from its parent, the parent can open both ends of a pipe, fork, then the parent close one end and the child close the other end of the pipe. This is what happens when you call the `popen()` routine to run another program from within yours, i.e. you can write to the file descriptor returned from `popen()` and the child process sees it as its stdin, or you can read from the file descriptor and see what the program wrote to its stdout. (The mode parameter to `popen()` defines which; if you want to do both, then you can do the plumbing yourself without too much difficulty.)

Also, the child process inherits memory segments mmapped anonymously (or by mmapping the special file `` `/dev/zero' ``) by the parent; these shared memory segments are not accessible from unrelated processes.

1.6 How do I get rid of zombie processes?

1.6.1 What is a zombie?

When a program forks and the child finishes before the parent, the kernel still keeps some of its information about the child in case the parent might need it -- for example, the parent may need to check the child's exit status. To be able to get this information, the parent calls `wait()`; when this happens, the kernel can discard the information.

In the interval between the child terminating and the parent calling `wait()`, the child is said to be a `zombie'. (If you do `ps', the child will have a `Z' in its status field to indicate this.) Even though it's not running, it's still taking up an entry in the process table. (It consumes no other resources, but some utilities may show bogus figures for e.g. CPU usage; this is because some parts of the process table entry have been overlaid by accounting info to save space.)

This is not good, as the process table has a fixed number of entries and it is possible for the system to run out of them. Even if the system doesn't run out, there is a limit on the number of processes each user can run, which is usually smaller than the system's limit. This is one of the reasons why you should always check if `fork()` failed, by the way!

If the parent terminates without calling wait(), the child is `adopted' by `init`, which handles the work necessary to cleanup after the child. (This is a special system program with process ID 1 -- it's actually the first program to run after the system boots up).

1.6.2 How do I prevent them from occuring?

You need to ensure that your parent process calls `wait()` (or `waitpid()`, `wait3()`, etc.) for every child process that terminates; or, on some systems, you can instruct the system that you are uninterested in child exit states.

Another approach is to `fork()` *twice*, and have the immediate child process exit straight away. This causes the grandchild process to be orphaned, so the init process is responsible for cleaning it up. For code to do this, see the function `fork2()` in the examples section.

To ignore child exit states, you need to do the following (check your system's manpages to see if this works):

```
    struct sigaction sa;
    sa.sa_handler = SIG_IGN;
#ifdef SA_NOCLDWAIT
    sa.sa_flags = SA_NOCLDWAIT;
#else
    sa.sa_flags = 0;
#endif
    sigemptyset(&sa.sa_mask);
    sigaction(SIGCHLD, &sa, NULL);
```

If this is successful, then the `wait()` functions are prevented from working; if any of them are called, they will wait until *all* child processes have terminated, then return failure with `errno == ECHILD`.

The other technique is to catch the SIGCHLD signal, and have the signal handler call `waitpid()` or `wait3()`. See the examples section for a complete program.

1.7 How do I get my program to act like a daemon?

A **daemon** process is usually defined as a background process that does not belong to a terminal session. Many system services are performed by daemons; network services, printing etc.

Simply invoking a program in the background isn't really adequate for these long-running programs; that does not correctly detach the process from the terminal session that started it. Also, the conventional way of starting daemons is simply to issue the command manually or from an rc script; the daemon is expected to put *itself* into the background.

Here are the steps to become a daemon:

1. `fork()` so the parent can exit, this returns control to the command line or shell invoking your program. This step is required so that the new process is guaranteed not to be a process group leader. The next step, `setsid()`, fails if you're a process group leader.
2. `setsid()` to become a process group and session group leader. Since a controlling terminal is associated with a session, and this new session has not yet acquired a controlling terminal our process now has no controlling terminal, which is a Good Thing for daemons.
3. `fork()` again so the parent, (the session group leader), can exit. This means that we, as a non-session group leader, can never regain a controlling terminal.
4. `chdir("/")` to ensure that our process doesn't keep any directory in use. Failure to do this could make it so that an administrator couldn't unmount a filesystem, because it was our current directory. [Equivalently, we could change to any directory containing files important to the daemon's operation.]
5. `umask(0)` so that we have complete control over the permissions of anything we write. We don't know what umask we may have inherited. [This step is optional]
6. `close()` fds 0, 1, and 2. This releases the standard in, out, and error we inherited from our parent process. We have no way of knowing where these fds might have been redirected to. Note that many daemons use `sysconf()` to determine the limit `_SC_OPEN_MAX`. `_SC_OPEN_MAX` tells you the maximun open files/process. Then in a loop, the daemon can close all possible file descriptors. You have to decide if you need to do this or not. If you think that there might be file-descriptors open you should close them, since there's a limit on number of concurrent file descriptors.
7. Establish new open descriptors for stdin, stdout and stderr. Even if you don't plan to use them, it is still a good idea to have them open. The precise handling of these is a matter of taste; if you have a logfile, for example, you might wish to open it as stdout or stderr, and open `` `/dev/null' `` as stdin; alternatively, you could open `` `/dev/console' `` as stderr and/or stdout, and `` `/dev/null' `` as stdin, or any other combination that makes sense for your particular daemon.

Almost none of this is necessary (or advisable) if your daemon is being started by `inetd`. In that case, stdin, stdout and stderr are all set up for you to refer to the network connection, and the `fork()`s and session manipulation should *not* be done (to avoid confusing `inetd`). Only the `chdir()` and `umask()` steps remain as useful.

1.8 How can I look at process in the system like ps does?

You really *don't* want to do this.

The most portable way, by far, is to do `popen(pscmd, "r")` and parse the output. (pscmd should be something like `"ps -ef"' on SysV systems; on BSD systems there are many possible display options: choose one.)

In the examples section, there are two complete versions of this; one for SunOS 4, which requires root permission to run and uses the `kvm_*' routines to read the information from kernel data structures; and another for SVR4 systems (including SunOS 5), which uses the `/proc' filesystem.

It's even easier on systems with an SVR4.2-style `/proc'; just read a psinfo_t structure from the file `/proc/PID/psinfo' for each PID of interest. However, this method, while probably the cleanest, is also perhaps the least well-supported. (On FreeBSD's `/proc', you read a semi-undocumented printable string from `/proc/PID/status'; Linux has something similar.)

1.9 Given a pid, how can I tell if it's a running program?

Use `kill()` with 0 for the signal number.

There are four possible results from this call:

- `kill()` returns 0
  - this implies that a process exists with the given PID, and the system would allow you to send signals to it. It is system-dependent whether the process could be a zombie.
- `kill()` returns @math{-1}, `errno == ESRCH`
  - either no process exists with the given PID, or security enhancements are causing the system to deny its existence. (On some systems, the process could be a zombie.)
- `kill()` returns @math{-1}, `errno == EPERM`
  - the system would not allow you to kill the specified process. This means that either the process exists (again, it could be a zombie) or draconian security enhancements are present (e.g. your process is not allowed to send signals to *anybody*).
- `kill()` returns @math{-1}, with some other value of `errno`
  - you are in trouble!

The most-used technique is to assume that success or failure with `EPERM` implies that the process exists, and any other error implies that it doesn't.

An alternative exists, if you are writing specifically for a system (or all those systems) that provide a `/proc' filesystem: checking for the existence of `/proc/PID' may work.

1.10 What's the return value of system/pclose/waitpid?

> The return value of `system()`, `pclose()`, or `waitpid()` doesn't seem to be the exit value of my process... or the exit value is shifted left 8 bits... what's the deal?

The man page is right, and so are you! If you read the man page for `waitpid()` you'll find that the return code for the process is encoded. The value returned by the process is normally in the top 16 bits, and the rest is used for other things. You can't rely on this though, not if you want to be portable, so the suggestion is that you use the macros provided. These are usually documented under `wait()` or `wstat`.

Macros defined for the purpose (in `` `<sys/wait.h>' ``) include (stat is the value returned by `waitpid()`):

`WIFEXITED(stat)`
: Non zero if child exited normally.

`WEXITSTATUS(stat)`
: exit code returned by child

`WIFSIGNALED(stat)`
: Non-zero if child was terminated by a signal

`WTERMSIG(stat)`
: signal number that terminated child

`WIFSTOPPED(stat)`
: non-zero if child is stopped

`WSTOPSIG(stat)`
: number of signal that stopped child

`WIFCONTINUED(stat)`
: non-zero if status was for continued child

`WCOREDUMP(stat)`
: If `WIFSIGNALED(stat)` is non-zero, this is non-zero if the process left behind a core dump.

1.11 How do I find out about a process' memory usage?

Look at `getrusage()`, if available.

1.12 Why do processes never decrease in size?

When you free memory back to the heap with `free()`, on almost all systems that *doesn't* reduce the memory usage of your program. The memory `free()`d is still part of the process' address space, and will be used to satisfy future `malloc()` requests.

If you really need to free memory back to the system, look at using `mmap()` to allocate private

anonymous mappings. When these are unmapped, the memory really is released back to the system. Certain implementations of `malloc()` (e.g. in the GNU C Library) automatically use `mmap()` where available to perform large allocations; these blocks are then returned to the system on `free()`.

Of course, if your program increases in size when you think it shouldn't, you may have a `memory leak' -- a bug in your program that results in unused memory not being freed.

1.13 How do I change the name of my program (as seen by `ps')?

On BSDish systems, the `ps` program actually looks into the address space of the running process to find the current `argv[]`, and displays that. That enables a program to change its `name' simply by modifying `argv[]`.

On SysVish systems, the command name and usually the first 80 bytes of the parameters are stored in the process' u-area, and so can't be directly modified. There may be a system call to change this (unlikely), but otherwise the only way is to perform an `exec()`, or write into kernel memory (dangerous, and only possible if running as root).

Some systems (notably Solaris) may have two separate versions of `ps`, one in `` `/usr/bin/ps' `` with SysV behaviour, and one in `` `/usr/ucb/ps' `` with BSD behaviour. On these systems, if you change `argv[]`, then the BSD version of `ps` will reflect the change, and the SysV version won't.

Check to see if your system has a function `setproctitle()`.

1.14 How can I find a process' executable file?

This would be a good candidate for a list of `Frequently Unanswered Questions', because the fact of asking the question usually means that the design of the program is flawed. `:-)`

You can make a `best guess' by looking at the value of `argv[0]`. If this contains a `/', then it is probably the absolute or relative (to the current directory at program start) path of the executable. If it does not, then you can mimic the shell's search of the `PATH` variable, looking for the program. However, success is not guaranteed, since it is possible to invoke programs with arbitrary values of `argv[0]`, and in any case the executable may have been renamed or deleted since it was started.

If all you want is to be able to print an appropriate invocation name with error messages, then the best approach is to have `main()` save the value of `argv[0]` in a global variable for use by the entire program. While there is no guarantee whatsoever that the value in `argv[0]` will be meaningful, it is the best option available in most circumstances.

The most common reason people ask this question is in order to locate configuration files with their program. This is considered to be bad form; directories containing executables should contain *nothing* except executables, and administrative requirements often make it desirable for configuration files to be located on different filesystems to executables.

A less common, but more legitimate, reason to do this is to allow the program to call `exec()` *on itself*; this is a method used (e.g. by some versions of `sendmail`) to completely reinitialise the process (e.g. if a daemon receives a `SIGHUP`).

1.14.1 So where do I put my configuration files then?

The correct directory for this usually depends on the particular flavour of Unix you're using; `/var/opt/PACKAGE', `/usr/local/lib', `/usr/local/etc', or any of several other possibilities. User-specific configuration files are usually hidden `dotfiles' under $HOME (e.g. `$HOME/.exrc').

From the point of view of a package that is expected to be usable across a range of systems, this usually implies that the location of any sitewide configuration files will be a compiled-in default, possibly using a `--prefix' option on a configure script (Autoconf scripts do this). You might wish to allow this to be overridden at runtime by an environment variable. (If you're not using a configure script, then put the default in the Makefile as a `-D' option on compiles, or put it in a `config.h' header file, or something similar.)

User-specific configuration should be either a single dotfile under $HOME, or, if you need multiple files, a dot-subdirectory. (Files or directories whose names start with a dot are omitted from directory listings by default.) Avoid creating multiple entries under $HOME, because this can get very cluttered. Again, you can allow the user to override this location with an environment variable. Programs should always behave sensibly if they fail to find any per-user configuration.

1.15 Why doesn't my process get SIGHUP when its parent dies?

Because it's not supposed to.

SIGHUP is a signal that means, by convention, "the terminal line got hung up". It has nothing to do with parent processes, and is usually generated by the tty driver (and delivered to the foreground process group).

However, as part of the session management system, there are exactly two cases where SIGHUP is sent on the death of a process:

- When the process that dies is the session leader of a session that is attached to a terminal device, SIGHUP is sent to all processes in the foreground process group of that terminal device.
- When the death of a process causes a process group to become orphaned, and one or more processes in the orphaned group are *stopped*, then SIGHUP and SIGCONT are sent to all members of the orphaned group. (An orphaned process group is one where no process in the group has a parent which is part of the same session, but not the same process group.)

1.16 How can I kill all descendents of a process?

There isn't a fully general approach to doing this. While you can determine the relationships between processes by parsing ps output, this is unreliable in that it represents only a snapshot of the system.

However, if you're lauching a subprocess that might spawn further subprocesses of its own, and you want to be able to kill the entire spawned job at one go, the solution is to put the subprocess into a new process group, and kill that process group if you need to.

The preferred function for creating process groups is setpgid(). Use this if possible rather than

`setpgrp()` because the latter differs between systems (on some systems `setpgrp();' is equivalent to `setpgid(0,0);', on others, `setpgrp()` and `setpgid()` are identical).

See the job-control example in the examples section.

Putting a subprocess into its own process group has a number of effects. In particular, unless you explicitly place the new process group in the foreground, it will be treated as a background job with these consequences:

- it will be stopped with `SIGTTIN` if it attempts to read from the terminal
- if `tostop` is set in the terminal modes, it will be stopped with `SIGTTOU` if it attempts to write to the terminal (attempting to change the terminal modes should also cause this, independently of the current setting of `tostop`)
- The subprocess will not receive keyboard signals from the terminal (e.g. `SIGINT` or `SIGQUIT`)

In many applications input and output will be redirected anyway, so the most significant effect will be the lack of keyboard signals. The parent application should arrange to catch at least `SIGINT` and `SIGQUIT` (and preferably `SIGTERM` as well) and clean up any background jobs as necessary.

Go to the first, previous, next, last section, table of contents.

---

2. General File handling (including pipes and sockets)

See also the Sockets FAQ, available at:

http://www.lcg.org/sock-faq/

2.1 How to manage multiple connections?

> I have to monitor more than one (fd/connection/stream) at a time. How do I manage all of them?

Use `select()` or `poll()`.

Note: `select()` was introduced in BSD, whereas `poll()` is an artifact of SysV STREAMS. As such, there are portability issues; pure BSD systems may still lack `poll()`, whereas some older SVR3 systems may not have `select()`. SVR4 added `select()`, and the Posix.1g standard defines both.

`select()` and `poll()` essentially do the same thing, just differently. Both of them examine a set of file descriptors to see if specific events are pending on any, and then optionally wait for a specified time for an event to happen.

[Important note: neither `select()` nor `poll()` do anything useful when applied to plain files; they are useful for sockets, pipes, ptys, ttys & possibly other character devices, but this is system-dependent.]

There the similarity ends....

2.1.1 How do I use select()?

The interface to `select()` is primarily based on the concept of an `fd_set`, which is a set of FDs (usually implemented as a bit-vector). In times past, it was common to assume that FDs were smaller than 32, and just use an int to store the set, but these days, one usually has more FDs available, so it is important to use the standard macros for manipulating fd_sets:

```
fd_set set;
FD_ZERO(&set);      /* empties the set */
FD_SET(fd,&set);    /* adds FD to the set */
FD_CLR(fd,&set);    /* removes FD from the set */
FD_ISSET(fd,&set)   /* true if FD is in the set */
```

In most cases, it is the system's responsibility to ensure that fdsets can handle the whole range of file descriptors, but in some cases you may have to predefine the `FD_SETSIZE` macro. *This is system-dependent*; check your `select()` manpage. Also, some systems have problems handling more than 1024 file descriptors in `select()`.

The basic interface to select is simple:

```
int select(int nfds, fd_set *readset,
                     fd_set *writeset,
                     fd_set *exceptset, struct timeval *timeout);
```

where

`nfds`
: the number of FDs to examine; this must be greater than the largest FD in any of the fdsets, *not* the actual number of FDs specified

`readset`
: the set of FDs to examine for readability

`writeset`
: the set of FDs to examine for writability

`exceptfds`
: the set of FDs to examine for exceptional status (note: errors are *not* exceptional statuses)

`timeout`
: NULL for infinite timeout, or points to a timeval specifying the maximum wait time (if `tv_sec` and `tv_usec` both equal zero, then the status of the FDs is polled, but the call never blocks)

The call returns the number of `ready' FDs found, and the three fdsets are modified in-place, with only the ready FDs left in the sets. Use the `FD_ISSET` macro to test the returned sets.

Here's a simple example of testing a single FD for readability:

```
int isready(int fd)
{
    int rc;
    fd_set fds;
    struct timeval tv;

    FD_ZERO(&fds);
    FD_SET(fd,&fds);
    tv.tv_sec = tv.tv_usec = 0;

    rc = select(fd+1, &fds, NULL, NULL, &tv);
    if (rc < 0)
      return -1;

    return FD_ISSET(fd,&fds) ? 1 : 0;
}
```

Note that we can pass `NULL` for fdsets that we aren't interested in testing.

2.1.2 How do I use poll()?

`poll()` accepts a pointer to a list of `struct pollfd`, in which the descriptors and the events you wish to poll for are stored. The events are specified via a bitwise mask in the events field of the structure. The instance of the structure will later be filled in and returned to you with any events which occured. Macros

defined by `poll.h' on SVR4 (probably older versions as well), are used to specify the events in the field. A timeout may be specified in milliseconds, only the type provided is an integer which is quite perplexing. A timeout of 0 causes poll() to return immediately; a value of @math{-1} will suspend poll until an event is found to be true.

```
struct pollfd {
    int fd;         /* The descriptor. */
    short events;  /* The event(s) is/are specified here. */
    short revents; /* Events found are returned here. */
};
```

A lot like select(), the return value if positive reflects how many descriptors were found to satisfy the events requested. A zero return value is returned if the timeout period is reached before any of the events specified have occured. A negative value should immediately be followed by a check of errno, since it signifies an error.

If no events are found, revents is cleared, so there's no need for you to do this yourself.

The returned events are tested to contain the event.

Here's an example:

```
/* Poll on two descriptors for Normal data, or High priority data.
   If any found call function handle() with appropriate descriptor
   and priority. Don't timeout, only give up if error, or one of the
   descriptors hangs up. */

#include <stdlib.h>
#include <stdio.h>

#include <sys/types.h>
#include <stropts.h>
#include <poll.h>

#include <unistd.h>
#include <errno.h>
#include <string.h>

#define NORMAL_DATA 1
#define HIPRI_DATA 2

int poll_two_normal(int fd1,int fd2)
{
    struct pollfd poll_list[2];
    int retval;

    poll_list[0].fd = fd1;
    poll_list[1].fd = fd2;
    poll_list[0].events = POLLIN|POLLPRI;
    poll_list[1].events = POLLIN|POLLPRI;
```

```
    while(1)
    {
        retval = poll(poll_list,(unsigned long)2,-1);
        /* Retval will always be greater than 0 or -1 in this case.
           Since we're doing it while blocking */

        if(retval < 0)
        {
            fprintf(stderr,"Error while polling: %s\n",strerror(errno));
            return -1;
        }

        if(((poll_list[0].revents&POLLHUP) == POLLHUP) ||
           ((poll_list[0].revents&POLLERR) == POLLERR) ||
           ((poll_list[0].revents&POLLNVAL) == POLLNVAL) ||
           ((poll_list[1].revents&POLLHUP) == POLLHUP) ||
           ((poll_list[1].revents&POLLERR) == POLLERR) ||
           ((poll_list[1].revents&POLLNVAL) == POLLNVAL))
          return 0;

        if((poll_list[0].revents&POLLIN) == POLLIN)
          handle(poll_list[0].fd,NORMAL_DATA);
        if((poll_list[0].revents&POLLPRI) == POLLPRI)
          handle(poll_list[0].fd,HIPRI_DATA);
        if((poll_list[1].revents&POLLIN) == POLLIN)
          handle(poll_list[1].fd,NORMAL_DATA);
        if((poll_list[1].revents&POLLPRI) == POLLPRI)
          handle(poll_list[1].fd,HIPRI_DATA);
    }
}
```

2.1.3 Can I use SysV IPC at the same time as select or poll?

No. (Except on AIX, which has an incredibly ugly kluge to allow this.)

In general, trying to combine the use of `select()` or `poll()` with using SysV message queues is troublesome. SysV IPC objects are not handled by file descriptors, so they can't be passed to `select()` or `poll()`. There are a number of workarounds, of varying degrees of ugliness:

- Abandon SysV IPC completely. `:-)`
- `fork()`, and have the child process handle the SysV IPC, communicating with the parent process by a pipe or socket, which the parent process can `select()` on.
- As above, but have the child process do the `select()`, and communicate with the parent by message queue.
- Arrange for the process that sends messages to you to send a signal after each message. **Warning:** handling this right is non-trivial; it's very easy to write code that can potentially lose messages or deadlock using this method.

(Other methods exist.)

2.2 How can I tell when the other end of a connection shuts down?

If you try to read from a pipe, socket, FIFO etc. when the writing end of the connection has been closed, you get an end-of-file indication (`read()` returns 0 bytes read). If you try and write to a pipe, socket etc. when the reading end has closed, then a `SIGPIPE` signal will be delivered to the process, killing it unless the signal is caught. (If you ignore or block the signal, the `write()` call fails with `EPIPE`.)

2.3 Best way to read directories?

While historically there have been several different interfaces for this, the only one that really matters these days the the Posix.1 standard `` `<dirent.h>' `` functions.

The function `opendir()` opens a specified directory; `readdir()` reads directory entries from it in a standardised format; `closedir()` does the obvious. Also provided are `rewinddir()`, `telldir()` and `seekdir()` which should also be obvious.

If you are looking to expand a wildcard filename, then most systems have the `glob()` function; also check out `fnmatch()` to match filenames against a wildcard, or `ftw()` to traverse entire directory trees.

2.4 How can I find out if someone else has a file open?

This is another candidate for `Frequently Unanswered Questions' because, in general, your program should never be interested in whether someone else has the file open. If you need to deal with concurrent access to the file, then you should be looking at advisory locking.

This is, in general, too hard to do anyway. Tools like `fuser` and `lsof` that find out about open files do so by grovelling through kernel data structures in a most unhealthy fashion. You can't usefully invoke them from a program, either, because by the time you've found out that the file is/isn't open, the information may already be out of date.

2.5 How do I `lock' a file?

There are three main file locking mechanisms available. All of them are `advisory'[*], which means that they rely on programs co-operating in order to work. It is therefore vital that all programs in an application should be consistent in their locking regime, and great care is required when your programs may be sharing files with third-party software.

[*] Well, actually some Unices permit mandatory locking via the sgid bit -- RTFM for this hack.

Some applications use lock files -- something like `` `FILENAME.lock' ``. Simply testing for the existence of such files is inadequate though, since a process may have been killed while holding the lock. The method used by UUCP (probably the most notable example: it uses lock files for controlling access to modems, remote systems etc.) is to store the PID in the lockfile, and test if that pid is still running. Even this isn't enough to be sure (since PIDs are recycled); it has to have a backstop check to see if the lockfile is old, which means that the process holding the lock must update the file regularly. Messy.

The locking functions are:

```
flock();
lockf();
fcntl();
```

`flock()` originates with BSD, and is now available in most (but not all) Unices. It is simple and effective on a single host, but doesn't work at all with NFS. It locks an entire file. Perhaps rather deceptively, the popular Perl programming language implements its own `flock()` where necessary, conveying the illusion of true portability.

`fcntl()` is the only POSIX-compliant locking mechanism, and is therefore the only truly portable lock. It is also the most powerful, and the hardest to use. For NFS-mounted file systems, `fcntl()` requests are passed to a daemon (`rpc.lockd`), which communicates with the lockd on the server host. Unlike `flock()` it is capable of record-level locking.

`lockf()` is merely a simplified programming interface to the locking functions of `fcntl()`.

Whatever locking mechanism you use, it is important to sync all your file IO while the lock is active:

```
lock(fd);
write_to(some_function_of(fd));
flush_output_to(fd); /* NEVER unlock while output may be buffered */
unlock(fd);
do_something_else;   /* another process might update it */
lock(fd);
seek(fd, somewhere); /* because our old file pointer is not safe */
do_something_with(fd);
...
```

A few useful `fcntl()` locking recipes (error handling omitted for simplicity) are:

```
#include <fcntl.h>
#include <unistd.h>

read_lock(int fd)   /* a shared lock on an entire file */
{
    fcntl(fd, F_SETLKW, file_lock(F_RDLCK, SEEK_SET));
}

write_lock(int fd)  /* an exclusive lock on an entire file */
{
    fcntl(fd, F_SETLKW, file_lock(F_WRLCK, SEEK_SET));
}

append_lock(int fd) /* a lock on the _end_ of a file -- other
                       processes may access existing records */
{
    fcntl(fd, F_SETLKW, file_lock(F_WRLCK, SEEK_END));
}
```

The function file_lock used by the above is

```
struct flock* file_lock(short type, short whence)
{
    static struct flock ret ;
    ret.l_type = type ;
    ret.l_start = 0 ;
    ret.l_whence = whence ;
    ret.l_len = 0 ;
    ret.l_pid = getpid() ;
    return &ret ;
}
```

2.6 How do I find out if a file has been updated by another process?

This is close to being a Frequently Unanswered Question, because people asking it are often looking for some notification from the system when a file or directory is changed, and there is no portable way of getting this. (IRIX has a non-standard facility for monitoring file accesses, but I've never heard of it being available in any other flavour.)

In general, the best you can do is to use `fstat()` on the file. (Note: the overhead on `fstat()` is quite low, usually much lower than the overhead of `stat()`.) By watching the mtime and ctime of the file, you can detect when it is modified, or deleted/linked/renamed. This is a bit kludgy, so you might want to rethink *why* you want to do it.

2.7 How does the `du' utility work?

`du` simply traverses the directory structure calling `stat()` (or more accurately, `lstat()`) on every file and directory it encounters, adding up the number of blocks consumed by each.

If you want more detail about how it works, then the simple answer is:

> Use the source, Luke!

Source for BSD systems (FreeBSD, NetBSD and OpenBSD) is available as unpacked source trees on their FTP distribution sites; source for GNU versions of utilities is available from any of the GNU mirrors, but you have to unpack the archives yourself.

2.8 How do I find the size of a file?

Use `stat()`, or `fstat()` if you have the file open.

These calls fill in a data structure containing all the information about the file that the system keeps track of; that includes the owner, group, permissions, size, last access time, last modification time, etc.

The following routine illustrates how to use `stat()` to get the file size.

```
#include <stdlib.h>
```

```
#include <stdio.h>

#include <sys/types.h>
#include <sys/stat.h>

int get_file_size(char *path,off_t *size)
{
  struct stat file_stats;

  if(stat(path,&file_stats))
    return -1;

  *size = file_stats.st_size;
  return 0;
}
```

2.9 How do I expand `~' in a filename like the shell does?

The standard interpretation for `~' at the start of a filename is: if alone or followed by a `/', then substitute the current user's home directory; if followed by the name of a user, then substitute that user's home directory. If no valid expansion can be found, then shells will leave the filename unchanged.

Be wary, however, of filenames that actually start with the `~' character. Indiscriminate tilde-expansion can make it very difficult to specify such filenames to a program; while quoting will prevent the shell from doing the expansion, the quotes will have been removed by the time the program sees the filename. As a general rule, do not try and perform tilde-expansion on filenames that have been passed to the program on the command line or in environment variables. (Filenames generated within the program, obtained by prompting the user, or obtained from a configuration file, are good candidates for tilde-expansion.)

Here's a piece of C++ code (using the standard string class) to do the job:

```
string expand_path(const string& path)
{
    if (path.length() == 0 || path[0] != '~')
      return path;

    const char *pfx = NULL;
    string::size_type pos = path.find_first_of('/');

    if (path.length() == 1 || pos == 1)
    {
        pfx = getenv("HOME");
        if (!pfx)
        {
            // Punt. We're trying to expand ~/, but HOME isn't set
            struct passwd *pw = getpwuid(getuid());
            if (pw)
              pfx = pw->pw_dir;
```

```
        }
    }
    else
    {
        string user(path,1,(pos==string::npos) ? string::npos : pos-1);
        struct passwd *pw = getpwnam(user.c_str());
        if (pw)
          pfx = pw->pw_dir;
    }

    // if we failed to find an expansion, return the path unchanged.

    if (!pfx)
      return path;

    string result(pfx);

    if (pos == string::npos)
      return result;

    if (result.length() == 0 || result[result.length()-1] != '/')
      result += '/';

    result += path.substr(pos+1);

    return result;
}
```

2.10 What can I do with named pipes (FIFOs)?

2.10.1 What is a named pipe?

A **named pipe** is a special file that is used to transfer data between unrelated processes. One (or more) processes write to it, while another process reads from it. Named pipes are visible in the file system and may be viewed with `ls' like any other file. (Named pipes are also called **fifo**s; this term stands for `First In, First Out'.)

Named pipes may be used to pass data between unrelated processes, while normal (unnamed) pipes can only connect parent/child processes (unless you try *very* hard).

Named pipes are strictly unidirectional, even on systems where anonymous pipes are bidirectional (full-duplex).

2.10.2 How do I create a named pipe?

To create a named pipe interactively, you'll use either `mknod` or `mkfifo`. On some systems, mknod will be found in /etc. In other words, it might not be on your path. See your man pages for details.

To make a named pipe within a C program use `mkfifo()`:

```
/* set the umask explicitly, you don't know where it's been */
umask(0);
if (mkfifo("test_fifo", S_IRUSR | S_IWUSR | S_IRGRP | S_IWGRP))
{
    perror("mkfifo");
    exit(1);
}
```

If you don't have `mkfifo()`, you'll have to use `mknod()`:

```
/* set the umask explicitly, you don't know where it's been */
umask(0);
if (mknod("test_fifo",
           S_IFIFO | S_IRUSR | S_IWUSR | S_IRGRP | S_IWGRP,
          0))
{
    perror("mknod");
    exit(1);
}
```

2.10.3 How do I use a named pipe?

To use the pipe, you open it like a normal file, and use `read()` and `write()` just as though it was a plain pipe.

However, the `open()` of the pipe may block. The following rules apply:

- If you open for both reading and writing (`O_RDWR`), then the open will not block.
- If you open for reading (`O_RDONLY`), the open will block until another process opens the FIFO for writing, unless `O_NONBLOCK` is specified, in which case the open succeeds.
- If you open for writing `O_WRONLY`, the open will block until another process opens the FIFO for reading, unless `O_NONBLOCK` is specified, in which case the open fails.

When reading and writing the FIFO, the same considerations apply as for regular pipes and sockets, i.e. `read()` will return EOF when all writers have closed, and `write()` will raise `SIGPIPE` when there are no readers. If `SIGPIPE` is blocked or ignored, the call fails with `EPIPE`.

2.10.4 Can I use a named pipe across NFS?

No, you can't. There is no facility in the NFS protocol to do this. (You may be able to use a named pipe on an NFS-mounted filesystem to communicate between processes on the same client, though.)

2.10.5 Can multiple processes write to the pipe simultaneously?

If each piece of data written to the pipe is less than `PIPE_BUF` in size, then they will not be interleaved. However, the boundaries of writes are not preserved; when you read from the pipe, the read call will return as much data as possible, even if it originated from multiple writes.

The value of `PIPE_BUF` is guaranteed (by Posix) to be at least 512. It may or may not be defined in `` `<limits.h>' ``, but it can be queried for individual pipes using `pathconf()` or `fpathconf()`.

2.10.6 Using named pipes in applications

How can I implement two way communication between one server and several clients?

It is possible that more than one client is communicating with your server at once. As long as each command they send to the server is smaller than `PIPE_BUF` (see above), they can all use the same named pipe to send data to the server. All clients can easily know the name of the server's incoming fifo.

However, the server can not use a single pipe to communicate with the clients. If more than one client is reading the same pipe, there is no way to ensure that the appropriate client receives a given response.

A solution is to have the client create its own incoming pipe before sending data to the server, or to have the server create its outgoing pipes after receiving data from the client.

Using the client's process ID in the pipe's name is a common way to identify them. Using fifos named in this manner, each time the client sends a command to the server, it can include its PID as part of the command. Any returned data can be sent through the appropriately named pipe.

Go to the first, previous, next, last section, table of contents.

---

3. Terminal I/O

3.1 How can I make my program not echo input?

> How can I make my program not echo input, like login does when asking for your password?

There is an easy way, and a slightly harder way:

The easy way, is to use `getpass()`, which is probably found on almost all Unices. It takes a string to use as a prompt. It will read up to an `EOF` or newline and returns a pointer to a static area of memory holding the string typed in.

The harder way is to use `tcgetattr()` and `tcsetattr()`, both use a `struct termios` to manipulate the terminal. The following two routines should allow echoing, and non-echoing mode.

```
#include <stdlib.h>
#include <stdio.h>

#include <termios.h>
#include <string.h>

static struct termios stored_settings;

void echo_off(void)
{
    struct termios new_settings;
    tcgetattr(0,&stored_settings);
    new_settings = stored_settings;
    new_settings.c_lflag &= (~ECHO);
    tcsetattr(0,TCSANOW,&new_settings);
    return;
}

void echo_on(void)
{
    tcsetattr(0,TCSANOW,&stored_settings);
```

```
    return;
}
```

Both routines used, are defined by the POSIX standard.

3.2 How can I read single characters from the terminal?

> How can I read single characters from the terminal? My program is always waiting for the user to press RETURN.

Terminals are usually in canonical mode, where input is read in lines after it is edited. You may set this into non-canonical mode, where you set how many characters should be read before input is given to your program. You also may set the timer in non-canonical mode terminals to 0, this timer flushs your buffer at set intervals. By doing this, you can use `getc()` to grab the key pressed immediately by the user. We use `tcgetattr()` and `tcsetattr()` both of which are defined by POSIX to manipulate the `termios` structure.

```
#include <stdlib.h>
#include <stdio.h>

#include <termios.h>
#include <string.h>

static struct termios stored_settings;

void set_keypress(void)
{
    struct termios new_settings;

    tcgetattr(0,&stored_settings);

    new_settings = stored_settings;

    /* Disable canonical mode, and set buffer size to 1 byte */
    new_settings.c_lflag &= (~ICANON);
    new_settings.c_cc[VTIME] = 0;
    new_settings.c_cc[VMIN] = 1;

    tcsetattr(0,TCSANOW,&new_settings);
    return;
}

void reset_keypress(void)
```

```
{
    tcsetattr(0,TCSANOW,&stored_settings);
    return;
}
```

3.3 How can I check and see if a key was pressed?

> How can I check and see if a key was pressed? On DOS I use the `kbhit()` function, but there doesn't seem to be an equivalent?

If you set the terminal to single-character mode (see previous answer), then (on most systems) you can use `select()` or `poll()` to test for readability.

3.4 How can I move the cursor around the screen?

> How can I move the cursor around the screen? I want to do full screen editing without using curses.

Seriously, you probably **don't** want to do this. Curses knows about how to handle all sorts of oddities that different terminal types exhibit; while the termcap/terminfo data will tell you whether any given terminal type possesses any of these oddities, you will probably find that correctly handling all the combinations is a **huge** job.

However, if you insist on getting your hands dirty (so to speak), look into the `termcap` functions, particularly `tputs()`, `tparm()` and `tgoto()`.

3.5 What are pttys?

Pseudo-teletypes (pttys, ptys, other variant abbreviations) are pseudo-devices that have two parts: the **master** side, which can be thought of as the `user', and the **slave** side, which behaves like a standard tty device.

They exist in order to provide a means to emulate the behaviour of a serial terminal under the control of a program. For example, `telnet` uses a pseudo-terminal on the remote system; the remote login shell sees the behaviour it expects from a tty device, but the master side of the pseudo-terminal is being controlled by a daemon that forwards all data over the network. They are also used by programs such as `xterm`, `expect`, `script`, `screen`, `emacs`, and many others.

3.6 How to handle a serial port or modem?

The handling of serial devices under Unix is heavily influenced by the traditional use of serial terminals. Historically, various combinations of ioctls and other hacks were necessary to control the precise behaviour of a serial device, but fortunately this is one of the areas that POSIX made some efforts to standardise.

If you're using a system that doesn't understand `<termios.h>`, `tcsetattr()` and related functions, then you'll have to go elsewhere for information (or upgrade your system to something less archaeological).

There are still significant differences between systems, however, mainly in the area of device names, handling of hardware flow control, and modem signalling. (Whenever possible, leave the device driver to do all the handshaking work, and don't attempt to manipulate handshaking signals directly.)

The basic steps for opening and initialising a serial device are:

- `open()` the device; this may require the use of certain flags:

  `O_NONBLOCK`

  > Opening a dial-in or modem-controlled device will block until carrier is present, unless this flag is used. A nonblocking open gives you the opportunity to disable the modem controls (see CLOCAL below) if necessary.

  `O_NOCTTY`

  > On 4.4BSD-derived systems this is redundant, but on other systems it controls whether the serial device can become a control terminal for the session. In most cases you probably *don't* want to acquire a control terminal, and should therefore specify this flag, but there are exceptions.
- Use `tcgetattr()` to retrieve the current device modes. While one will often ignore most or all of the initial settings thus obtained, it's still a convenient way of initialising a `struct termios`.
- Set suitable values for `c_iflag`, `c_oflag`, `c_cflag`, `c_lflag`, and `c_cc` in the termios structure. (See below.)
- Use `cfsetispeed()` and `cfsetospeed()` to set the desired baud rate. Very few systems allow you to set differing input and output speeds, so as a general rule you should set both to your desired speed.
- Use `tcsetattr()` to set the device modes.
- You may wish, if you used `O_NONBLOCK` when opening the port, to use `fcntl()` to ensure that `O_NONBLOCK` is turned off again. Systems seem to differ as to whether a nonblocking open on a tty will affect subsequent `read()` calls; better to be explicit.

Once you have opened and set up the port, you can then use `read()` and `write()` normally. Note that the behaviour of `read()` will be controlled by the flag settings you gave to `tcsetattr()`.

`tcflush()`, `tcdrain()`, `tcsendbreak()` and `tcflow()` are additional useful functions that you should be aware of.

When you're done with the port, and want to close it, be aware of a very nasty little hazard on some systems; if there's any pending output waiting to be written to the device (e.g. if output flow is stopped by hardware or software handshaking), your process can hang **unkillably** in the `close()` call until the output drains. Calling `tcflush()` to discard any pending output is probably a wise move.

(Blocked output on tty devices is by far the most common cause of "unkillable" processes in my experience.)

3.6.1 Serial device names and types

The device names used for serial port devices vary quite widely between systems. Some examples from different systems are:

- `` `/dev/tty[0-9][a-z]' `` for direct access devices, and `` `/dev/tty[0-9][A-Z]' `` for modem control devices (e.g. SCO Unix)
- `` `/dev/cua[0-9]p[0-9]' `` for direct access devices, `` `/dev/cul[0-9]p[0-9]' `` for dial-out devices and `` `/dev/ttyd[0-9]p[0-9]' `` for dial-in devices (e.g. HP-UX)
- `` `/dev/cua[a-z][0-9]' `` for dial-out devices and `` `/dev/tty[a-z][0-9]' `` for dial-in devices (e.g. FreeBSD)

The precise interaction between the device name used, and the effect on any hardware handshake lines is system-, configuration- and hardware-dependant, but will usually follow approximately these rules (assuming that the hardware is RS-232 DTE):

- A successful open of any device should assert DTR and RTS
- A blocking open of a modem-control or dial-in device will wait for DCD (and possibly also DSR and/or CTS) to be raised, usually after asserting DTR/RTS.
- An open of a dial-out device while an open call to the corresponding dial-in device is blocked waiting for carrier *may or may not* cause the open of the dial-in port to complete. Some systems implement a simple sharing scheme for dial-in and dial-out ports whereby the dial-in port is effectively "put to sleep" while the dial-out port is in use; other systems do not do this, and sharing the port between dial-in and dial-out on such systems requires external cooperation (e.g. use of UUCP lockfiles) to avoid contention problems.

3.6.2 Setting up termios flags

Some hints on setting up the termios flags when using a serial device that you've opened yourself (as opposed to using your existing control tty):

3.6.2.1 c_iflag

You probably want to set **all** the bits in `c_iflag` to 0, unless you want to use software flow control (ick) in which case you set `IXON` and `IXOFF`.

3.6.2.2 c_oflag

Most of the bits of `c_oflag` are hacks of one sort or another to make output to slow terminals work, and as such some newer systems have dropped almost all of them as obsolete (especially all the gory output-padding options). As with `c_iflag`, setting everything to 0 is reasonable for most applications.

3.6.2.3 c_cflag

When setting the character size, remember to mask using `CSIZE` first; e.g. to set 8-bit characters, use:

```
attr.c_cflag &= ~CSIZE;
attr.c_cflag |= CS8;
```

Other important flags found in `c_cflag` that you probably want to turn **on** and `CREAD` and `HUPCL`.

If you need to generate even parity, then set `PARENB` and clear `PARODD`; if you need to generate odd parity then set both `PARENB` and `PARODD`. If you don't want parity at all, then make sure `PARENB` is clear.

Clear `CSTOPB` unless you actually need to generate two stop bits.

Flags for enabling hardware flow control may also be found in `c_cflag`, but they aren't standardised (pity).

3.6.2.4 c_lflag

Most applications will probably want to turn off `ICANON` (canonical, i.e. line-based, input processing), `ECHO`, and `ISIG`.

`IEXTEN` is a more complex issue. If you don't turn it off, the implementation is allowed to do nonstandard things (like define additional control characters in `c_cc`) that might cause unexpected results, but you might need to leave `IEXTEN` enabled on some systems to get useful features like hardware flow control.

3.6.2.5 c_cc

This is an array of characters that have special meanings on input. These characters are given names like `VINTR`, `VSTOP` etc.; the names are indexes into the array.

(Two of these "characters" are not really characters at all, but control the behaviour of `read()` when `ICANON` is disabled; these are `VMIN` and `VTIME`.)

The indexes are often referred to as though they were actual variables, e.g. "set VMIN to 1" actually means "set c_cc[VMIN] to 1". The shorthand is useful and only occasionally confusing.

Many of the slots in `c_cc` are only used if some other combination of flags is set:

Used only if `ICANON` is set

> `VEOF`, `VEOL`, `VERASE`, `VKILL` (and also `VEOL2`, `VSTATUS` and `VWERASE` if defined and `IEXTEN` is set)

Used only if `ISIG` is set

> `VINTR`, `VQUIT`, `VSUSP` (and also `VDSUSP` if defined and `IEXTEN` is set)

Used only if `IXON` or `IXOFF` is set

> `VSTOP`, `VSTART`

Used only if `ICANON` is **not** set

> `VMIN`, `VTIME`

Implementations may define additional entries in `c_cc`. It may be prudent to initialise all the entries to `_POSIX_VDISABLE` (the constant `NCCS` gives the array size) before setting the specific values you wish to use.

`VMIN` and `VTIME` (which may share slots with `VEOF` and `VEOL` respectively, depending on the implementation) have the following meaning. The value of `VTIME` is (if not 0) always interpreted as a timer in tenths of seconds.

`c_cc[VMIN] > 0, c_cc[VTIME] > 0`

> `read()` will return when either VMIN bytes of input are available, or if at least one character has been read and VTIME has expired between characters, or if interrupted by a signal.

`c_cc[VMIN] > 0, c_cc[VTIME] == 0`

> `read()` will return when VMIN bytes of input are available, or if interrupted. Otherwise it will wait indefinitely.

`c_cc[VMIN] == 0, c_cc[VTIME] > 0`

> `read()` will return as soon as any input is available; if VTIME expires with no data arriving, it will return with no characters read. (This conflicts slightly with the end-of-file indication received in the event of modem hangup; using 1 for VMIN and either `alarm()` or `select()` for a timeout avoids this particular problem.)

`c_cc[VMIN] == 0, c_cc[VTIME] == 0`

> `read()` will always return immediately; if no data is available it will return with no characters read (with the same problem as above).

Go to the first, previous, next, last section, table of contents.

---

4. System Information

4.1 How can I tell how much memory my system has?

This is another `Frequently Unanswered Question'. In most cases, you should not even *attempt* to find out.

If you really must, then it can usually be done, but in a highly system-dependent fashion. For example, on Solaris, you can use `sysconf(_SC_PHYS_PAGES)` and `sysconf(_SC_PAGESIZE)`; on FreeBSD, you can use `sysctl()`; on Linux you can read and parse `` `/proc/meminfo' `` (being careful to allow any of the historically valid formats for this file); other systems may have their own methods. I'm not aware of any more portable methods.

For HP-UX (9 and 10), the following code has been contributed:

```
struct pst_static pst;

if (pstat_getstatic(&pst, sizeof(pst), (size_t) 1, 0) != -1)
{
    printf(" Page Size: %lu\n", pst.page_size);
    printf("Phys Pages: %lu\n", pst.physical_memory);
}
```

4.2 How do I check a user's password?

4.2.1 How do I get a user's password?

Traditionally user passwords were kept in the `` `/etc/passwd' `` file, on most UNIX flavours. Which is usually of this format:

username:password:uid:gid:gecos field:home directory:login shell

Though this has changed with time, now user information may be kept on other hosts, or not necessarily in the `` `/etc/passwd' `` file. Modern implementations also made use of `shadow' password files which hold the password, along with sensitive information. This file would be readable only by privileged users.

The password is usually not in clear text, but encrypted due to security concerns.

POSIX defines a suite of routines which can be used to access this database for queries. The quickest way to get an individual record for a user is with the `getpwnam()` and `getpwuid()` routines. Both return a pointer to a struct passwd, which holds the users information in various members. `getpwnam()` accepts a string holding the user's name, `getpwuid()` accepts a uid (type `uid_t` as defined by POSIX). Both return NULL if they fail.

However, as explained earlier, a shadow database exists on most modern systems to hold sensitive information, namely the password. Some systems only return the password if the calling uid is of the superuser, others require you to use another suite of functions for the shadow password database. If this is the case you need to make use of `getspnam()`, which accepts a username and returns a struct spwd. Again, in order to successfully do this, you will need to have privileges. (On some systems, notably HP-UX and SCO, you may need to use `getprpwnam()` instead.)

4.2.2 How do I get shadow passwords by uid?

> My system uses the getsp* suite of routines to get the sensitive user information. However I do not have `getspuid()`, only `getspnam()`. How do I work around this, and get by uid?

The work around is relatively painless. The following routine should go straight into your personal utility library:

```
#include <stdlib.h>
#include <stdio.h>

#include <pwd.h>
#include <shadow.h>

struct spwd *getspuid(uid_t pw_uid)
{
  struct spwd *shadow;
  struct passwd *ppasswd;

  if( ((ppasswd = getpwuid(pw_uid)) == NULL)
      || ((shadow = getspnam(ppasswd->pw_name)) == NULL))
    return NULL;

  return shadow;
}
```

The problem is, that some systems do not keep the uid, or other information in the shadow database.

4.2.3 How do I verify a user's password?

The fundamental problem here is, that various authentication systems exist, and passwords aren't always what they seem. Also with the traditional one way encryption method used by most UNIX flavours (out of the box), the encryption algorithm may differ, some systems use a one way DES encryption, others like the international release of FreeBSD use MD5.

The most popular way is to have a one way encryption algorithm, where the password cannot be decrypted. Instead the password is taken in clear text from input, and encrypted and checked against the encrypted password in the database. The details of how to encrypt should really come from your man page for `crypt()`, but here's a usual version:

```
/* given a plaintext password and an encrypted password, check if
 * they match; returns 1 if they match, 0 otherwise.
 */

int check_pass(const char *plainpw, const char *cryptpw)
{
    return strcmp(crypt(plainpw,cryptpw), cryptpw) == 0;
}
```

This works because the salt used in encrypting the password is stored as an initial substring of the encrypted value.

WARNING: on some systems, password encryption is actually done with a variant of crypt called `bigcrypt()`.

Go to the first, previous, next, last section, table of contents.

---

5. Miscellaneous programming

5.1 How do I compare strings using wildcards?

The answer to *that* depends on what exactly you mean by `wildcards'.

There are two quite different concepts that qualify as `wildcards'. They are:

Filename patterns

These are what the shell uses for filename expansion (`globbing').

Regular Expressions

These are used by editors, `grep`, etc. for matching text, but they normally *aren't* applied to filenames.

5.1.1 How do I compare strings using filename patterns?

Unless you are unlucky, your system should have a function `fnmatch()` to do filename matching. This generally allows only the Bourne shell style of pattern; i.e. it recognises `*', `[...]' and `?', but probably won't support the more arcane patterns available in the Korn and Bourne-Again shells.

If you don't have this function, then rather than reinvent the wheel, you are probably better off snarfing a copy from the BSD or GNU sources.

Also, for the common cases of matching actual filenames, look for `glob()`, which will find all existing files matching a pattern.

5.1.2 How do I compare strings using regular expressions?

There are a number of slightly different syntaxes for regular expressions; most systems use at least two: the one recognised by `ed`, sometimes known as `Basic Regular Expressions', and the one recognised by `egrep`, `Extended Regular Expressions'. Perl has it's own slightly different flavour, as does Emacs.

To support this multitude of formats, there is a corresponding multitude of implementations. Systems will generally have regexp-matching functions (usually `regcomp()` and `regexec()`) supplied, but be wary; some systems have more than one implementation of these functions available, with different interfaces. In addition, there are many library implementations available. (It's common, BTW, for regexps to be compiled to an internal form before use, on the assumption that you may compare several separate strings against the same regexp.)

One library available for this is the `rx' library, available from the GNU mirrors. This seems to be under active development, which may be a good or a bad thing depending on your point of view :-)

5.2 What's the best way to send mail from a program?

There are several ways to send email from a Unix program. Which is the best method to use in a given situation varies, so I'll present two of them. A third possibility, not covered here, is to connect to a local SMTP port (or a smarthost) and use SMTP directly; see RFC 821.

5.2.1 The simple method: /bin/mail

For simple applications, it may be sufficient to invoke `mail` (usually `` `/bin/mail' ``, but could be `` `/usr/bin/mail' `` on some systems).

WARNING: Some versions of UCB Mail may execute commands prefixed by `~!' or `~|' given in the message body even in non-interactive mode. This can be a security risk.

Invoked as `mail -s 'subject' recipients...' it will take a message body on standard input, and supply a default header (including the specified subject), and pass the message to `sendmail` for delivery.

This example mails a test message to `root` on the local system:

```
#include <stdio.h>

#define MAILPROG "/bin/mail"

int main()
{
    FILE *mail = popen(MAILPROG " -s 'Test Message' root", "w");
    if (!mail)
    {
        perror("popen");
        exit(1);
    }

    fprintf(mail, "This is a test.\n");

    if (pclose(mail))
    {
        fprintf(stderr, "mail failed!\n");
        exit(1);
    }
}
```

If the text to be sent is already in a file, then one can do:

```
    system(MAILPROG " -s 'file contents' root </tmp/filename");
```

These methods can be extended to more complex cases, but there are many pitfalls to watch out for:

- If using system() or popen(), you must be very careful about quoting arguments to protect them from filename expansion or word splitting

- Constructing command lines from user-specified data is a common source of buffer-overrun errors and other security holes
- This method does not allow for CC: or BCC: recipients to be specified (some versions of /bin/mail may allow this, some do not)

5.2.2 Invoking the MTA directly: /usr/lib/sendmail

The `mail` program is an example of a **Mail User Agent**, a program intended to be invoked by the user to send and receive mail, but which does not handle the actual transport. A program for transporting mail is called an **MTA**, and the most commonly found MTA on Unix systems is called `sendmail`. There are other MTAs in use, such as `MMDF`, but these generally include a program that emulates the usual behaviour of `sendmail`.

Historically, `sendmail` has usually been found in `` `/usr/lib' ``, but the current trend is to move library programs out of `` `/usr/lib' `` into directories such as `` `/usr/sbin' `` or `` `/usr/libexec' ``. As a result, one normally invokes `sendmail` by its full path, which is system-dependent.

To understand how `sendmail` behaves, it's useful to understand the concept of an **envelope**. This is very much like paper mail; the envelope defines who the message is to be delivered to, and who it is from (for the purpose of reporting errors). Contained in the envelope are the **headers**, and the **body**, separated by a blank line. The format of the headers is specified primarily by RFC 822; see also the MIME RFCs.

There are two main ways to use `sendmail` to originate a message: either the envelope recipients can be explicitly supplied, or `sendmail` can be instructed to deduce them from the message headers. Both methods have advantages and disadvantages.

5.2.2.1 Supplying the envelope explicitly

The recipients of a message can simply be specified on the command line. This has the drawback that mail addresses can contain characters that give `system()` and `popen()` considerable grief, such as single quotes, quoted strings etc. Passing these constructs successfully through shell interpretation presents pitfalls. (One can do it by replacing any single quotes by the sequence single-quote backslash single-quote single-quote, then surrounding the entire address with single quotes. Ugly, huh?)

Some of this unpleasantness can be avoided by eschewing the use of `system()` or `popen()`, and resorting to `fork()` and `exec()` directly. This is sometimes necessary in any event; for example, user-installed handlers for SIGCHLD will usually break `pclose()` to a greater or lesser extent.

Here's an example:

```
#include <sys/types.h>
#include <sys/wait.h>
#include <unistd.h>
#include <stdlib.h>
#include <fcntl.h>
#include <sysexits.h>
/* #include <paths.h> if you have it */

#ifndef _PATH_SENDMAIL
#define _PATH_SENDMAIL "/usr/lib/sendmail"
```

```
#endif

/* -oi means "dont treat . as a message terminator"
 * remove ,"--" if using a pre-V8 sendmail (and hope that no-one
 * ever uses a recipient address starting with a hyphen)
 * you might wish to add -oem (report errors by mail)
 */

#define SENDMAIL_OPTS "-oi","--"

/* this is a macro for returning the number of elements in array */

#define countof(a) ((sizeof(a))/sizeof((a)[0]))

/* send the contents of the file open for reading on FD to the
 * specified recipients; the file is assumed to contain RFC822 headers
 * & body, the recipient list is terminated by a NULL pointer; returns
 * -1 if error detected, otherwise the return value from sendmail
 * (which uses <sysexits.h> to provide meaningful exit codes)
 */

int send_message(int fd, const char **recipients)
{
    static const char *argv_init[] = { _PATH_SENDMAIL, SENDMAIL_OPTS };
    const char **argvec = NULL;
    int num_recip = 0;
    pid_t pid;
    int rc;
    int status;

    /* count number of recipients */

    while (recipients[num_recip])
        ++num_recip;

    if (!num_recip)
        return 0;    /* sending to no recipients is successful */

    /* alloc space for argument vector */

    argvec = malloc((sizeof char*) * (num_recip+countof(argv_init)+1));
    if (!argvec)
        return -1;

    /* initialise argument vector */

    memcpy(argvec, argv_init, sizeof(argv_init));
    memcpy(argvec+countof(argv_init),
           recipients, num_recip*sizeof(char*));
```

```
    argvec[num_recip + countof(argv_init)] = NULL;

    /* may need to add some signal blocking here. */

    /* fork */

    switch (pid = fork())
    {
    case 0:   /* child */

        /* Plumbing */
        if (fd != STDIN_FILENO)
            dup2(fd, STDIN_FILENO);

        /* defined elsewhere -- closes all FDs >= argument */
        closeall(3);

        /* go for it: */
        execv(_PATH_SENDMAIL, argvec);
        _exit(EX_OSFILE);

    default:  /* parent */

        free(argvec);
        rc = waitpid(pid, &status, 0);
        if (rc < 0)
            return -1;
        if (WIFEXITED(status))
            return WEXITSTATUS(status);
        return -1;

    case -1:  /* error */
        free(argvec);
        return -1;
    }
}
```

5.2.2.2 Allowing sendmail to deduce the recipients

The `-t' option to `sendmail` instructs `sendmail` to parse the headers of the message, and use all the recipient-type headers (i.e. `To:`, `Cc:` and `Bcc:`) to construct the list of envelope recipients. This has the advantage of simplifying the `sendmail` command line, but makes it impossible to specify recipients other than those listed in the headers. (This is not usually a problem.)

As an example, here's a program to mail a file on standard input to specified recipients as a MIME attachment. Some error checks have been omitted for brevity. This requires the `` `mimencode' `` program from the `metamail` distribution.

```
#include <stdio.h>
```

```
#include <unistd.h>
#include <fcntl.h>
/* #include <paths.h> if you have it */

#ifndef _PATH_SENDMAIL
#define _PATH_SENDMAIL "/usr/lib/sendmail"
#endif

#define SENDMAIL_OPTS "-oi"
#define countof(a) ((sizeof(a))/sizeof((a)[0]))

char tfilename[L_tmpnam];
char command[128+L_tmpnam];

void cleanup(void)
{
    unlink(tfilename);
}

int main(int argc, char **argv)
{
    FILE *msg;
    int i;

    if (argc < 2)
    {
        fprintf(stderr, "usage: %s recipients...\n", argv[0]);
        exit(2);
    }

    if (tmpnam(tfilename) == NULL
        || (msg = fopen(tfilename,"w")) == NULL)
        exit(2);

    atexit(cleanup);

    fclose(msg);
    msg = fopen(tfilename,"a");
    if (!msg)
        exit(2);

    /* construct recipient list */

    fprintf(msg, "To: %s", argv[1]);
    for (i = 2; i < argc; i++)
        fprintf(msg, ",\n\t%s", argv[i]);
    fputc('\n',msg);

    /* Subject */
```

```
    fprintf(msg, "Subject: file sent by mail\n");

    /* sendmail can add it's own From:, Date:, Message-ID: etc. */

    /* MIME stuff */

    fprintf(msg, "MIME-Version: 1.0\n");
    fprintf(msg, "Content-Type: application/octet-stream\n");
    fprintf(msg, "Content-Transfer-Encoding: base64\n");

    /* end of headers -- insert a blank line */

    fputc('\n',msg);
    fclose(msg);

    /* invoke encoding program */

    sprintf(command, "mimencode -b >>%s", tfilename);
    if (system(command))
        exit(1);

    /* invoke mailer */

    sprintf(command, "%s %s -t <%s",
            _PATH_SENDMAIL, SENDMAIL_OPTS, tfilename);
    if (system(command))
        exit(1);

    return 0;
}
```

Go to the first, previous, next, last section, table of contents.

---

6. Use of tools

6.1 How can I debug the children after a fork?

Depending on the tools available there are various ways:

Your debugger may have options to select whether to follow the parent or the child process (or both) after a `fork()`, which may be sufficient for some purposes.

Alternatively, your debugger may have an option which allows you to attach to a running process. This can be used to attach to the child process after it has been started. If you don't need to examine the very start of the child process, this is usually sufficient. Otherwise, you may wish to insert a `sleep()` call after the `fork()` in the child process, or a loop such as the following:

```
{
    volatile int f = 1;
    while(f);
}
```

which will hang the child process until you explicitly set `f` to 0 using the debugger.

Remember, too, that actively using a debugger isn't the only way to find errors in your program; utilities are available to trace system calls and signals on many unix flavours, and verbose logging is also often useful.

6.2 How to build library from other libraries?

Assuming we're talking about an archive (static) library, the easiest way is to explode all the constituent libraries into their original objects using `ar x' in an empty directory, and combine them all back together. Of course, there is the potential for collision of filenames, but if the libraries are large, you probably don't want to be combining them in the first place....

6.3 How to create shared libraries / dlls?

The precise method for creating shared libraries varies between different systems. There are two main parts to the process; firstly the objects to be included in the shared library must be compiled, usually with options to indicate that the code is to be position-independent; secondly, these objects are linked together to form the library.

Here's a trivial example that should illustrate the idea:

```
/* file shrobj.c */

const char *myfunc()
{
    return "Hello World";
}

/* end shrobj.c */

/* file hello.c */

#include <stdio.h>

extern const char *myfunc();

main()
{
    printf("%s\n", myfunc());
    return 0;
}

/* end hello.c */

$ gcc -fpic -c shrobj.c
$ gcc -shared -o libshared.so shrobj.o
$ gcc hello.c libshared.so
$ ./a.out
Hello World
```

By far the best method if you want the library and build procedure to be anything approaching portable is to use GNU Libtool. This is a small suite of utilities which know about the platform-dependent aspects of building shared libraries; you can distribute the necessary bits with your program, so that when the installer configures the package, he or she can decide what libraries to build. Libtool works fine on systems which do not support shared libraries. It also knows how to hook into GNU Autoconf and GNU Automake (if you use those tools to manage your program's build procedure).

If you don't want to use Libtool, then for compilers other than gcc, you should change the compiler options as follows:

AIX 3.2 using xlc (unverified)

> Drop the `-fpic', and use `-bM:SRE -bE:libshared.exp' instead of `-shared'. You also need to create a file `libshared.exp' containing the list of symbols to export, in this case `myfunc'. In addition, use `-e _nostart' when linking the library (on newer versions of AIX, I believe this changes to `-bnoentry').

SCO OpenServer 5 using the SCO Development System (unverified)

Shared libraries are only available on OS5 if you compile to ELF format, which requires the `-belf' option. Use `-Kpic' instead of `-fpic', and `cc -belf -G' for the link step.

Solaris using SparcWorks compilers

Use `-pic' instead of `-fpic', and use `ld -G' instead of `gcc -shared'.

(Submission of additional entries for the above table is encouraged.)

Other issues to watch out for:

- AIX and (I believe) Digital Unix don't require the -fpic option, because all code is position independent.
- AIX normally requires that you create an `export file', which is a list of symbols to be exported from the shared library. Some versions of the linker (possibly only the SLHS linker, svld?) have an option to export all symbols.
- If you want to refer to your shared library using the conventional `-l' parameter to the linker, you will have to understand how shared libraries are searched for at runtime on your system. The most common method is by using the `LD_LIBRARY_PATH` environment variable, but there is usually an additional option to specify this at link time.
- Most implementations record the expected runtime location of the shared library internally. Thus, moving a library from one directory to another may prevent it from working. Many systems have an option to the linker to specify the expected runtime location (the `-R' linker option on Solaris, for example, or the `LD_RUN_PATH` environment variable).
- ELF and a.out implementations may have a linker option `-Bsymbolic' which causes internal references within the library to be resolved. Otherwise, on these systems, all symbol resolution is deferred to the final link, and individual routines in the main program can override ones in the library.

6.4 Can I replace objects in a shared library?

Generally, no.

On most systems (except AIX), when you link objects to form a shared library, it's rather like linking an executable; the objects don't retain their individual identity. As a result, it's generally not possible to extract or replace individual objects from a shared library.

6.5 How can I generate a stack dump from within a running program?

Some systems provide library functions for unwinding the stack, so that you can (for example) generate a stack dump in an error-handling function. However, these are highly system-specific, and only a minority of systems have them.

A possible workaround is to get your program to invoke a debugger *on itself* -- the details still

vary slightly between systems, but the general idea is to do this:

```
void dump_stack(void)
{
    char s[160];

    sprintf(s, "/bin/echo 'where\ndetach' | dbx -a %d", getpid());
    system(s);

    return;
}
```

You will need to tweak the commands and parameters to dbx according to your system, or even substitute another debugger such as `gdb`, but this is still the most general solution to this particular problem that I've ever seen. Kudos to Ralph Corderoy for this one :-)

Here's a list of the command lines required for some systems:

Most systems using dbx

```
    "/bin/echo 'where\ndetach' | dbx /path/to/program %d"
```

AIX

```
    "/bin/echo 'where\ndetach' | dbx -a %d"
```

IRIX

```
    "/bin/echo 'where\ndetach' | dbx -p %d"
```

Go to the first, previous, next, last section, table of contents.

Examples

Catching SIGCHLD

```
#include <sys/types.h>  /* include this before any other sys headers */
#include <sys/wait.h>   /* header for waitpid() and various macros */
#include <signal.h>     /* header for signal functions */
#include <stdio.h>      /* header for fprintf() */
#include <unistd.h>     /* header for fork() */

void sig_chld(int);     /* prototype for our SIGCHLD handler */

int main()
{
    struct sigaction act;
    pid_t pid;

    /* Assign sig_chld as our SIGCHLD handler */
    act.sa_handler = sig_chld;

    /* We don't want to block any other signals in this example */
    sigemptyset(&act.sa_mask);

    /*
     * We're only interested in children that have terminated, not ones
     * which have been stopped (eg user pressing control-Z at terminal)
     */
    act.sa_flags = SA_NOCLDSTOP;

    /*
     * Make these values effective. If we were writing a real
     * application, we would probably save the old value instead of
     * passing NULL.
     */
    if (sigaction(SIGCHLD, &act, NULL) < 0)
    {
        fprintf(stderr, "sigaction failed\n");
        return 1;
    }

    /* Fork */
    switch (pid = fork())
    {
```

```
    case -1:
        fprintf(stderr, "fork failed\n");
        return 1;

    case 0:                         /* child -- finish straight away */
        _exit(7);                   /* exit status = 7 */

    default:                        /* parent */
        sleep(10);                  /* give child time to finish */
    }

    return 0;
}

/*
 * The signal handler function -- only gets called when a SIGCHLD
 * is received, ie when a child terminates
 */
void sig_chld(int signo)
{
    int status, child_val;

    /* Wait for any child without blocking */
    if (waitpid(-1, &status, WNOHANG) < 0)
    {
        /*
         * calling standard I/O functions like fprintf() in a
         * signal handler is not recommended, but probably OK
         * in toy programs like this one.
         */
        fprintf(stderr, "waitpid failed\n");
        return;
    }

    /*
     * We now have the info in 'status' and can manipulate it using
     * the macros in wait.h.
     */
    if (WIFEXITED(status))              /* did child exit normally? */
    {
        child_val = WEXITSTATUS(status); /* get child's exit status */
        printf("child's exited normally with status %d\n", child_val);
    }
}
```

Reading the process table -- SUNOS 4 version

```
#define _KMEMUSER
#include <sys/proc.h>
#include <kvm.h>
#include <fcntl.h>

char regexpstr[256];
#define INIT            register char *sp=regexpstr;
#define GETC()          (*sp++)
#define PEEKC()         (*sp)
#define UNGETC(c)       (--sp)
#define RETURN(pointer) return(pointer);
#define ERROR(val)
#include <regexp.h>

pid_t
getpidbyname(char *name,pid_t skipit)
{
    kvm_t *kd;
    char **arg;
    int error;
    char *p_name=NULL;
    char expbuf[256];
    char **freeme;
    int curpid;
    struct user * cur_user;
    struct user myuser;
    struct proc * cur_proc;


    if((kd=kvm_open(NULL,NULL,NULL,O_RDONLY,NULL))==NULL){
        return(-1);
    }
    sprintf(regexpstr,"^.*/%s$",name);
    compile(NULL,expbuf,expbuf+256,'\0');

    while(cur_proc=kvm_nextproc(kd)){
        curpid = cur_proc->p_pid;
        if((cur_user=kvm_getu(kd,cur_proc))!=NULL){
            error=kvm_getcmd(kd,cur_proc,cur_user,&arg,NULL);
            if(error==-1){
                if(cur_user->u_comm[0]!='\0'){
                    p_name=cur_user->u_comm;
                }
            }
            else{
                p_name=arg[0];
```

```
            }
        }
        if(p_name){
            if(!strcmp(p_name,name)){
                if(error!=-1){
                    free(arg);
                }
                if(skipit!=-1 && ourretval==skipit){
                    ourretval=-1;
                }
                else{
                    close(fd);
                    break;
                }
                break;
            }
            else{
                if(step(p_name,expbuf)){
                    if(error!=-1){
                        free(arg);
                    }
                    break;
                }
            }
        }
        if(error!=-1){
            free(arg);
        }
        p_name=NULL;
    }
    kvm_close(kd);
    if(p_name!=NULL){
        return(curpid);
    }
    return (-1);
}
```

Reading the process table -- SYSV version

```
pid_t
getpidbyname(char *name,pid_t skipit)
{
    DIR  *dp;
    struct dirent *dirp;
    prpsinfo_t retval;
    int fd;
    pid_t ourretval=-1;
```

```
    if((dp=opendir("/proc"))==NULL){
        return -1;
    }
    chdir("/proc");
    while((dirp=readdir(dp))!=NULL){
        if(dirp->d_name[0]!='.'){
            if((fd=open(dirp->d_name,O_RDONLY))!=-1){
                if(ioctl(fd,PIOCPSINFO,&retval)!=-1){
                    if(!strcmp(retval.pr_fname,name)){
                        ourretval=(pid_t)atoi(dirp->d_name);
                        if(skipit!=-1 && ourretval==skipit){
                            ourretval=-1;
                        }
                        else{
                            close(fd);
                            break;
                        }
                    }
                }
                close(fd);
            }
        }
    }
    closedir(dp);
    return ourretval;
}
```

Reading the process table -- AIX 4.2 version

```
#include <stdio.h>
#include <procinfo.h>

int getprocs(struct procsinfo *, int, struct fdsinfo *,
             int, pid_t *, int);

pid_t getpidbyname(char *name, pid_t *nextPid)
{
  struct procsinfo  pi;
  pid_t             retval = (pid_t) -1;
  pid_t             pid;

  pid = *nextPid;

  while(1)
  {
    if(getprocs(&pi, sizeof pi, 0, 0, &pid, 1) != 1)
      break;
```

```
    if(!strcmp(name, pi.pi_comm))
    {
      retval = pi.pi_pid;
      *nextPid = pid;
      break;
    }
  }

  return retval;
}

int main(int argc, char *argv[])
{
  int   curArg;
  pid_t pid;
  pid_t nextPid;

  if(argc == 1)
  {
    printf("syntax: %s <program> [program ...]\n",argv[0]);
    exit(1);
  }

  for(curArg = 1; curArg < argc; curArg++)
  {
    printf("Process IDs for %s\n", argv[curArg]);

    for(nextPid = 0, pid = 0; pid != -1; )
      if((pid = getpidbyname(argv[curArg], &nextPid)) != -1)
        printf("\t%d\n", pid);
  }
}
```

Reading the process table using popen and ps

```
#include <stdio.h>      /* FILE, sprintf, fgets, puts */
#include <stdlib.h>     /* atoi, exit, EXIT_SUCCESS */
#include <string.h>     /* strtok, strcmp */
#include <sys/types.h>  /* pid_t */
#include <sys/wait.h>   /* WIFEXITED, WEXITSTATUS */

char *procname(pid_t pid)
{
   static char line[133], command[80], *linep, *token, *cmd;
   FILE *fp;
   int status;

   if (0 == pid) return (char *)0;
```

```
   sprintf(command, "ps -p %d 2>/dev/null", pid);
   fp = popen(command, "r");
   if ((FILE *)0 == fp) return (char *)0;

   /* read the header line */
   if ((char *)0 == fgets(line, sizeof line, fp))
   {
      pclose(fp);
      return (char *)0;
   }

   /* figure out where the command name is from the column headings.
    * (BSD-ish machines put the COMMAND in the 5th column, while SysV
    * seems to put CMD or COMMAND in the 4th column.)
    */
   for (linep = line; ; linep = (char *)0)
   {
      if ((char *)0 == (token = strtok(linep, " \t\n")))
      {
         pclose(fp);
         return (char *)0;
      }
      if (0 == strcmp("COMMAND", token) || 0 == strcmp("CMD", token))
      { /*  we found the COMMAND column */
         cmd = token;
         break;
      }
   }

   /* read the ps(1) output line */
   if ((char *)0 == fgets(line, sizeof line, fp))
   {
      pclose(fp);
      return (char *)0;
   }

   /* grab the "word" underneath the command heading... */
   if ((char *)0 == (token = strtok(cmd, " \t\n")))
   {
      pclose(fp);
      return (char *)0;
   }

   status = pclose(fp);
   if (!WIFEXITED(status) || 0 != WEXITSTATUS(status))
     return (char *)0;

   return token;
}
```

```
int main(int argc, char *argv[])
{
   puts(procname(atoi(argv[1])));
   exit(EXIT_SUCCESS);
}
```

Daemon utility functions

```
#include <unistd.h>
#include <stdlib.h>
#include <fcntl.h>
#include <signal.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <errno.h>

/* closeall() -- close all FDs >= a specified value */

void closeall(int fd)
{
    int fdlimit = sysconf(_SC_OPEN_MAX);

    while (fd < fdlimit)
      close(fd++);
}

/* daemon() - detach process from user and disappear into the background
 * returns -1 on failure, but you can't do much except exit in that case
 * since we may already have forked. This is based on the BSD version,
 * so the caller is responsible for things like the umask, etc.
 */

/* believed to work on all Posix systems */

int daemon(int nochdir, int noclose)
{
    switch (fork())
    {
        case 0:  break;
        case -1: return -1;
        default: _exit(0);          /* exit the original process */
    }

    if (setsid() < 0)               /* shoudn't fail */
      return -1;

    /* dyke out this switch if you want to acquire a control tty in */
```

```
    /* the future -- not normally advisable for daemons */

    switch (fork())
    {
        case 0:  break;
        case -1: return -1;
        default: _exit(0);
    }

    if (!nochdir)
      chdir("/");

    if (!noclose)
    {
        closeall(0);
        open("/dev/null",O_RDWR);
        dup(0); dup(0);
    }

    return 0;
}

/* fork2() -- like fork, but the new process is immediately orphaned
 *            (won't leave a zombie when it exits)
 * Returns 1 to the parent, not any meaningful pid.
 * The parent cannot wait() for the new process (it's unrelated).
 */

/* This version assumes that you *haven't* caught or ignored SIGCHLD. */
/* If you have, then you should just be using fork() instead anyway.  */

int fork2()
{
    pid_t pid;
    int rc;
    int status;

    if (!(pid = fork()))
    {
        switch (fork())
        {
          case 0:  return 0;
          case -1: _exit(errno);    /* assumes all errnos are <256 */
          default: _exit(0);
        }
    }

    if (pid < 0 || waitpid(pid,&status,0) < 0)
      return -1;
```

```
    if (WIFEXITED(status))
      if (WEXITSTATUS(status) == 0)
        return 1;
      else
        errno = WEXITSTATUS(status);
    else
      errno = EINTR;  /* well, sort of :-) */

    return -1;
}
```

An example of using the above functions:

```
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <stdio.h>
#include <stdlib.h>
#include <syslog.h>
#include <errno.h>

int daemon(int,int);
int fork2(void);
void closeall(int);

#define TCP_PORT 8888

void errexit(const char *str)
{
    syslog(LOG_INFO, "%s failed: %d (%m)", str, errno);
    exit(1);
}

void errreport(const char *str)
{
    syslog(LOG_INFO, "%s failed: %d (%m)", str, errno);
}

/* the actual child process is here. */

void run_child(int sock)
{
    FILE *in = fdopen(sock,"r");
    FILE *out = fdopen(sock,"w");
    int ch;

    setvbuf(in, NULL, _IOFBF, 1024);
    setvbuf(out, NULL, _IOLBF, 1024);

    while ((ch = fgetc(in)) != EOF)
```

```
      fputc(toupper(ch), out);

    fclose(out);
}

/* This is the daemon's main work -- listen for connections and spawn */

void process()
{
    struct sockaddr_in addr;
    int addrlen = sizeof(addr);
    int sock = socket(AF_INET, SOCK_STREAM, 0);
    int flag = 1;
    int rc = setsockopt(sock, SOL_SOCKET, SO_REUSEADDR,
                        &flag, sizeof(flag));

    if (rc < 0)
      errexit("setsockopt");

    addr.sin_family = AF_INET;
    addr.sin_port = htons(TCP_PORT);
    addr.sin_addr.s_addr = INADDR_ANY;

    rc = bind(sock, (struct sockaddr *) &addr, addrlen);
    if (rc < 0)
      errexit("bind");

    rc = listen(sock, 5);
    if (rc < 0)
      errexit("listen");

    for (;;)
    {
        rc = accept(sock, (struct sockaddr *) &addr, &addrlen);

        if (rc >= 0)
          switch (fork2())
          {
            case 0:  close(sock); run_child(rc); _exit(0);
            case -1: errreport("fork2"); close(rc); break;
            default: close(rc);
          }
    }
}

int main()
{
    if (daemon(0,0) < 0)
    {
        perror("daemon");
```

```
        exit(2);
    }

    openlog("test", LOG_PID, LOG_DAEMON);

    process();

    return 0;
}
```

Modem handling example

```
/* issue some simple modem commands
 * requires the name of a serial device (preferably a dial-out device,
 * or a non-modem-control device) as its only parameter.
 * If you don't have functional dial-out devices, then move CLOCAL
 * to CFLAGS_TO_SET instead.
 */

#include <stdio.h>
#include <stdlib.h>
#include <fcntl.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/time.h>
#include <sys/ioctl.h>   /* maybe; system-dependent */
#include <termios.h>
#include <errno.h>
#include <string.h>
#include <ctype.h>

#define CFLAGS_TO_SET (CREAD | HUPCL)
#define CFLAGS_TO_CLEAR (CSTOPB | PARENB | CLOCAL)

enum flowmode { NoFlow, HardFlow, SoftFlow };

/* system-dependent */
#define CFLAGS_HARDFLOW (CRTSCTS)

#define EXAMPLE_BAUD B19200
#define EXAMPLE_FLOW HardFlow

static void die(const char *msg)
{
    fprintf(stderr, "%s\n", msg);
    exit(1);
}
```

```
static int close_and_complain(int fd, const char *msg, int err)
{
    fprintf(stderr, "%s: %s\n", msg, strerror(err));
    if (fd >= 0)
        close(fd);
    errno = err;
    return -1;
}

int open_port(const char *name, speed_t baud, enum flowmode flow)
{
    int flags;
    struct termios attr;

    int fd = open(name, O_RDWR | O_NONBLOCK | O_NOCTTY);

    if (fd < 0)
        return close_and_complain(-1, "open", errno);

    /* set vaguely sensibe settings */

    if (tcgetattr(fd, &attr) < 0)
        return close_and_complain(fd, "tcgetattr", errno);

    /* no special input or output processing */

    attr.c_iflag = (flow == SoftFlow) ? (IXON | IXOFF) : 0;
    attr.c_oflag = 0;

    /* set 8-bit character size and miscellanous control modes */

    attr.c_cflag &= ~(CSIZE | CFLAGS_TO_CLEAR | CFLAGS_HARDFLOW);
    attr.c_cflag |= (CS8 | CFLAGS_TO_SET);
    if (flow == HardFlow)
        attr.c_cflag |= CFLAGS_HARDFLOW;

    /* local modes */

    attr.c_lflag &= ~(ICANON | ECHO | ECHOE | ECHOK | ISIG);

    /* special characters -- most disabled by prior settings anyway */

    {
        int i;
#ifdef _POSIX_VDISABLE
        attr.c_cc[0] = _POSIX_VDISABLE;
#else
        attr.c_cc[0] = fpathconf(fd, _PC_VDISABLE);
#endif
        for (i = 1; i < NCCS; i++)
```

```
            attr.c_cc[i] = attr.c_cc[0];
    }

    attr.c_cc[VSTART] = 0x11;
    attr.c_cc[VSTOP] = 0x13;

    /* timing controls for read() */

    attr.c_cc[VMIN] = 1;
    attr.c_cc[VTIME] = 0;

    /* baud rate */

    cfsetispeed(&attr, baud);
    cfsetospeed(&attr, baud);

    /* write settings */

    if (tcsetattr(fd, TCSANOW, &attr) < 0)
        return close_and_complain(fd, "tcsetattr", errno);

    /* turn off O_NONBLOCK if the device remembered it */

    flags = fcntl(fd, F_GETFL, 0);
    if (flags < 0)
        return close_and_complain(fd, "fcntl(GETFL)", errno);
    if (fcntl(fd, F_SETFL, flags & ~O_NONBLOCK) < 0)
        return close_and_complain(fd, "fcntl(SETFL)", errno);

    return fd;
}

/* some simple timing utilities */

/* add SECS and USECS to *TV */

static void timeradd(struct timeval *tv, long secs, long usecs)
{
    tv->tv_sec += secs;
    if ((tv->tv_usec += usecs) >= 1000000)
    {
        tv->tv_sec += tv->tv_usec / 1000000;
        tv->tv_usec %= 1000000;
    }
}

/* Set *RES = *A - *B, returning the sign of the result */

static int timersub(struct timeval *res,
                    const struct timeval *a, const struct timeval *b)
```

```
{
    long sec = a->tv_sec - b->tv_sec;
    long usec = a->tv_usec - b->tv_usec;

    if (usec < 0)
        usec += 1000000, --sec;

    res->tv_sec = sec;
    res->tv_usec = usec;

    return (sec < 0) ? (-1) : ((sec == 0 && usec == 0) ? 0 : 1);
}

/* this doesn't try and cope with pathological strings (e.g. ababc)
 * timeout is in millisecs
 * A more usual approach to this is to use alarm() for the timeout.
 * This example avoids signal handling for simplicity and to illustrate
 * an alternative approach
 */

int expect(int fd, const char *str, int timeo)
{
    int matchlen = 0;
    int len = strlen(str);
    struct timeval now,end,left;
    fd_set fds;
    char c;

    gettimeofday(&end, NULL);
    timeradd(&end, timeo/1000, timeo%1000);

    while (matchlen < len)
    {
        gettimeofday(&now, NULL);
        if (timersub(&left, &end, &now) <= 0)
            return -1;

        FD_ZERO(&fds);
        FD_SET(fd, &fds);
        if (select(fd+1, &fds, NULL, NULL, &left) <= 0)
            return -1;

        if (read(fd, &c, 1) != 1)
            return -1;

        if (isprint((unsigned char)c) || c == '\n' || c == '\r')
            putchar(c);
        else
            printf("\\x%02x", c);
```

```
        if (c == str[matchlen])
            ++matchlen;
        else
            matchlen = 0;
    }

    return 0;
}

int main(int argc, char **argv)
{
    int fd;
    unsigned char c;

    if (argc < 2)
        die("no port specified");

    setvbuf(stdout, NULL, _IONBF, 0);

    fd = open_port(argv[1], EXAMPLE_BAUD, EXAMPLE_FLOW);
    if (fd < 0)
        die("cannot open port");

    write(fd, "AT\r", 3);
    if (expect(fd, "OK", 5000) < 0)
    {
        write(fd, "AT\r", 3);
        if (expect(fd, "OK", 5000) < 0)
        {
            tcflush(fd, TCIOFLUSH);
            close(fd);
            die("no response to AT");
        }
    }

    write(fd, "ATI4\r", 5);
    expect(fd, "OK", 10000);

    putchar('\n');

    tcflush(fd, TCIOFLUSH);
    close(fd);

    return 0;
}
```

Job Control example

```
/* functions to spawn foreground/background jobs */

#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
#include <fcntl.h>
#include <signal.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <errno.h>

/* Some of the functions below fail if the control tty can't be
 * located, or if the calling process isn't in the foreground. In the
 * first case, we are assuming that a foreground process will have the
 * ctty open on either stdin, stdout or stderr, and we return ENOTTY
 * if it isn't. In the second case, we return EPERM if a
 * non-foreground process attempts to put something into the
 * foreground (probably overly paranoid) except for the special case
 * of foreground_self().
 */

/* assign the terminal (open on ctty) to a specific pgrp. This wrapper
 * around tcsetpgrp() is needed only because of extreme bogosity on the
 * part of POSIX; conforming systems deliver STGTTOU if tcsetpgrp is
 * called from a non-foreground process (which it almost invariably is).
 * A triumph of spurious consistency over common sense.
 */

int assign_terminal(int ctty, pid_t pgrp)
{
    sigset_t sigs;
    sigset_t oldsigs;
    int rc;

    sigemptyset(&sigs);
    sigaddset(&sigs,SIGTTOU);
    sigprocmask(SIG_BLOCK, &sigs, &oldsigs);

    rc = tcsetpgrp(ctty, pgrp);

    sigprocmask(SIG_SETMASK, &oldsigs, NULL);

    return rc;
}
```

```
/* Like fork(), but does job control. FG is true if the newly-created
 * process is to be placed in the foreground. (This implicitly puts
 * the calling process in the background, so watch out for tty I/O
 * after doing this.) PGRP is -1 to create a new job, in which case
 * the returned pid is also the pgrp of the new job, or specifies an
 * existing job in the same session (normally used only for starting
 * second or subsequent process in a pipeline).  */

pid_t spawn_job(int fg, pid_t pgrp)
{
    int ctty = -1;
    pid_t pid;

    /* If spawning a *new* foreground job, require that at least one
     * of stdin, stdout or stderr refer to the control tty, and that
     * the current process is in the foreground.
     * Only check for controlling tty if starting a new foreground
     * process in an existing job.
     * A session without a control tty can only have background jobs
     */

    if (fg)
    {
        pid_t curpgrp;

        if ((curpgrp = tcgetpgrp(ctty = 2)) < 0
            && (curpgrp = tcgetpgrp(ctty = 0)) < 0
            && (curpgrp = tcgetpgrp(ctty = 1)) < 0)
            return errno = ENOTTY, (pid_t)-1;

        if (pgrp < 0 && curpgrp != getpgrp())
            return errno = EPERM, (pid_t)-1;
    }

    switch (pid = fork())
    {
        case -1: /* fork failure */
            return pid;

        case 0: /* child */

            /* establish new process group, and put ourselves in
             * foreground if necessary
             * unclear what to do if setpgid fails ("can't happen")
             */

            if (pgrp < 0)
                pgrp = getpid();

            if (setpgid(0,pgrp) == 0 && fg)
```

```
                assign_terminal(ctty, pgrp);

            return 0;

        default: /* parent */

            /* establish child process group here too. */

            if (pgrp < 0)
                pgrp = pid;

            setpgid(pid, pgrp);

            return pid;
    }

    /*NOTREACHED*/
}

/* Kill job PGRP with signal SIGNO */

int kill_job(pid_t pgrp, int signo)
{
    return kill(-pgrp, signo);
}

/* Suspend job PGRP */

int suspend_job(pid_t pgrp)
{
    return kill_job(pgrp, SIGSTOP);
}

/* Resume job PGRP in background */

int resume_job_bg(pid_t pgrp)
{
    return kill_job(pgrp, SIGCONT);
}

/* resume job PGRP in foreground */

int resume_job_fg(pid_t pgrp)
{
    pid_t curpgrp;
    int ctty;

    if ((curpgrp = tcgetpgrp(ctty = 2)) < 0
        && (curpgrp = tcgetpgrp(ctty = 0)) < 0
        && (curpgrp = tcgetpgrp(ctty = 1)) < 0)
```

```
        return errno = ENOTTY, (pid_t)-1;

    if (curpgrp != getpgrp())
        return errno = EPERM, (pid_t)-1;

    if (assign_terminal(ctty, pgrp) < 0)
        return -1;

    return kill_job(pgrp, SIGCONT);
}


/* put ourselves in the foreground, e.g. after suspending a foreground
 * job
 */

int foreground_self()
{
    pid_t curpgrp;
    int ctty;

    if ((curpgrp = tcgetpgrp(ctty = 2)) < 0
        && (curpgrp = tcgetpgrp(ctty = 0)) < 0
        && (curpgrp = tcgetpgrp(ctty = 1)) < 0)
        return errno = ENOTTY, (pid_t)-1;

    return assign_terminal(ctty, getpgrp());
}

/* closeall() - close all FDs >= a specified value */

void closeall(int fd)
{
    int fdlimit = sysconf(_SC_OPEN_MAX);

    while (fd < fdlimit)
        close(fd++);
}

/* like system(), but executes the specified command as a background
 * job, returning the pid of the shell process (which is also the pgrp
 * of the job, suitable for kill_job etc.)
 * If INFD, OUTFD or ERRFD are non-NULL, then a pipe will be opened and
 * a descriptor for the parent end of the relevent pipe stored there.
 * If any of these are NULL, they will be redirected to /dev/null in the
 * child.
 * Also closes all FDs > 2 in the child process (an oft-overlooked task)
 */

pid_t spawn_background_command(const char *cmd,
```

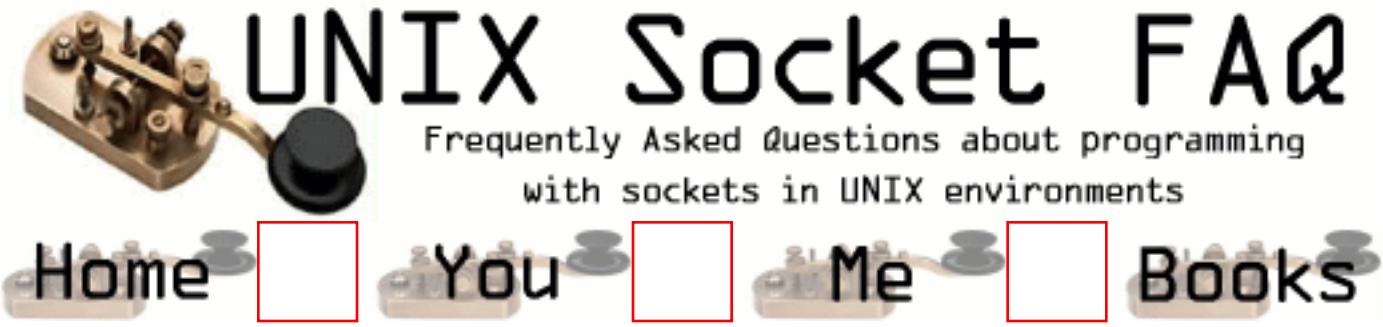
```
                                int *infd, int *outfd, int *errfd)
{
    int nullfd = -1;
    int pipefds[3][2];
    int error = 0;

    if (!cmd)
        return errno = EINVAL, -1;

    pipefds[0][0] = pipefds[0][1] = -1;
    pipefds[1][0] = pipefds[1][1] = -1;
    pipefds[2][0] = pipefds[2][1] = -1;

    if (infd && pipe(pipefds[0]) < 0)
        error = errno;
    else if (outfd && pipe(pipefds[1]) < 0)
        error = errno;
    else if (errfd && pipe(pipefds[2]) < 0)
        error = errno;

    if (!error && !(infd && outfd && errfd))
    {
        nullfd = open("/dev/null",O_RDWR);
        if (nullfd < 0)
            error = errno;
    }

    if (!error)
    {
        pid_t pid = spawn_job(0, -1);
        switch (pid)
        {
            case -1: /* fork failure */
                error = errno;
                break;

            case 0: /* child proc */

                dup2(infd ? pipefds[0][0] : nullfd, 0);
                dup2(outfd ? pipefds[1][1] : nullfd, 1);
                dup2(errfd ? pipefds[2][1] : nullfd, 2);
                closeall(3);

                execl("/bin/sh","sh","-c",cmd,(char*)NULL);

                _exit(127);

            default: /* parent proc */

                close(nullfd);
```

```
                if (infd)
                    close(pipefds[0][0]), *infd = pipefds[0][1];
                if (outfd)
                    close(pipefds[1][1]), *outfd = pipefds[1][0];
                if (errfd)
                    close(pipefds[2][1]), *errfd = pipefds[2][0];

                return pid;
            }
        }

        /* only reached if error */

        {
            int i,j;
            for (i = 0; i < 3; ++i)
                for (j = 0; j < 2; ++j)
                    if (pipefds[i][j] >= 0)
                        close(pipefds[i][j]);
        }

        if (nullfd >= 0)
            close(nullfd);

        return errno = error, (pid_t) -1;
    }

    /*-------------------------------------------------------------------*/
    /* This bit is a somewhat trivial example of using the above.        */

    pid_t bgjob = -1;
    volatile int signo = 0;

    #ifndef WCOREDUMP
     /* If WCOREDUMP is missing, you might want to supply a correct
      * definition for your platform (this is usually (status & 0x80) but
      * not always) or punt (as in this example) by assuming no core dumps.
      */
    # define WCOREDUMP(status) (0)
    #endif

    int check_children()
    {
        pid_t pid;
        int status;
        int count = 0;

        while ((pid = waitpid(-1, &status, WNOHANG | WUNTRACED)) > 0)
        {
            if (pid == bgjob && !WIFSTOPPED(status))
```

```
            bgjob = -1;

        ++count;

        if (WIFEXITED(status))
            fprintf(stderr,"Process %ld exited with return code %d\n",
                    (long)pid, WEXITSTATUS(status));
        else if (WIFSIGNALED(status))
            fprintf(stderr,"Process %ld killed by signal %d%s\n",
                    (long)pid, WTERMSIG(status),
                    WCOREDUMP(status) ? " (core dumped)" : "");
        else if (WIFSTOPPED(status))
            fprintf(stderr,"Process %ld stopped by signal %d\n",
                    (long)pid, WSTOPSIG(status));
        else
            fprintf(stderr,"Unexpected status - pid=%ld, status=0x%x\n",
                    (long)pid, status);
    }

    return count;
}

void sighandler(int sig)
{
    if (sig != SIGCHLD)
        signo = sig;
}

int main()
{
    struct sigaction act;
    int sigcount = 0;

    act.sa_handler = sighandler;
    act.sa_flags = 0;
    sigemptyset(&act.sa_mask);
    sigaction(SIGINT,&act,NULL);
    sigaction(SIGQUIT,&act,NULL);
    sigaction(SIGTERM,&act,NULL);
    sigaction(SIGTSTP,&act,NULL);
    sigaction(SIGCHLD,&act,NULL);

    fprintf(stderr,"Starting background job 'sleep 60'\n");
    bgjob = spawn_background_command("sleep 60", NULL, NULL, NULL);
    if (bgjob < 0)
    {
        perror("spawn_background_command");
        exit(1);
    }
    fprintf(stderr,"Background job started with id %ld\n", (long)bgjob);
```

```
    while (bgjob >= 0)
    {
        if (signo)
        {
            fprintf(stderr,"Signal %d caught\n", signo);
            if (sigcount++)
                kill_job(bgjob, SIGKILL);
            else
            {
                kill_job(bgjob, SIGTERM);
                kill_job(bgjob, SIGCONT);
            }
        }

        if (!check_children())
            pause();
    }

    fprintf(stderr,"Done - exiting\n");
    return 0;
}
```

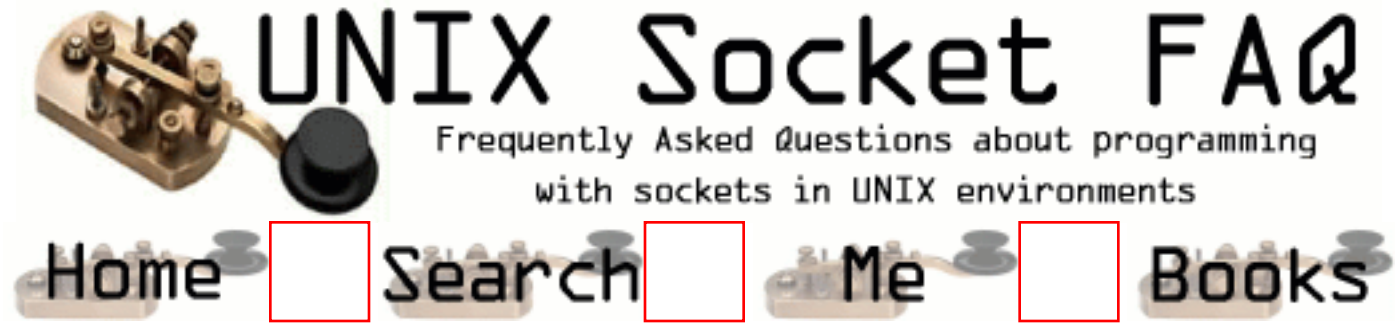
Go to the first, previous, next, last section, table of contents.



Search for: Search

This will search for words or phrases in the contents of the FAQ. Multiple keywords are not allowed. There are no special characters, and searches are not case sensitive.

| FAQ Home || Your User Profile || About Me || Books |



User Profile

Setting up a user profile simply sets some cookies in your browser so that your name and email address come up automatically when you post comments. Cookies are also used to notify you which questions are new to you. Nothing about you (other than your comments) is stored on the web server, so if you change browsers, or loose your cookie file, the settings will dissapear.

I know a lot of people don't like cookies, and that is just fine. This feature is provided for those who want it as a convenience. If you don't like cookies, don't fill out the form, and they won't be sent to you.

Your Name:

Your Email:

Create Profile

| FAQ Home || Search FAQ || About Me || Books |

My name is Vic Metcalfe, and I maintain the UNIX Socket FAQ, and this web site. I started learning how to write network applications when I wanted to create a MS-Windows GUI for CVS. I don't run Windows anymore, but I'm grateful to Microsoft for re-creating the need for UNIX.

I run mostly Linux, and I also have an old SparcStation 2 running Solaris 2.6, and another machine running Solaris x86 2.5.

Outside of computers I have a beautiful wife, Angela, a tiny baby girl named Carolyn, and a dog named Bjarne after Mr. Stroustrup of C++ fame. I hope he wouldn't take any offense; he's a good dog.

I also have an interested in television. I use to direct a local cable show, and I've taken six courses at the Second City in Toronto on improv. I highly recommend their classes to anyone.

| FAQ Home || Search FAQ || Your User Profile || Books |

UNIX Socket FAQ

Frequently Asked Questions about programming with sockets in UNIX environments

IN ASSOCIATION WITH amazon.com EARTH'S BIGGEST BOOKSTORE

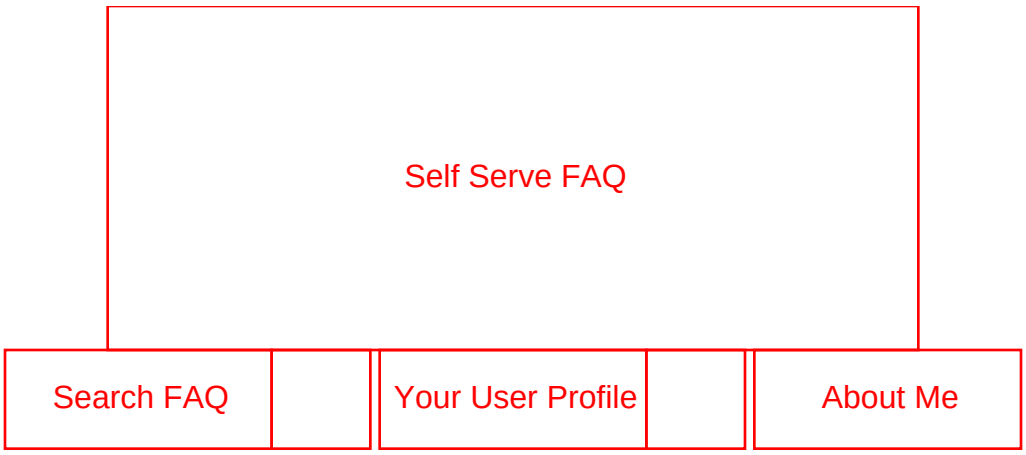
You will find reviews and comments on these books from amazon.com, where you can also buy the books. When you do, I get a small cut of the action. It has worked out to about $100/year. Thanks to all of you who have purchased books here! It is nice to receive a little compensation for the work done on the faq.

Advanced Programming in the Unix Environment By W. Richard Stevens ISBN: 0-201-56317-7	This is the most often recommended book about programming in the Unix environment. If you take the subject seriously and don't have a copy, I strongly suggest you pick this one up. It contains explanations of many advanced topics, complete with commented example code. This book rocks.
Unix Network Programming : Networking APIs: Sockets and Xti By W. Richard Stevens ISBN: 0-13-490012-X	If you are focusing on writing Network applications for Unix, this is also a must have book. This is the first, and only volume currently available of a three volume series. The other two books are in development. You can also still get the original single volume Unix Network Programming, as featured in the motion picture "Wayne's World 2".
UNIX Network Programming Volume 2: Interprocess Communications By W. Richard Stevens ISBN: 0130810819	Volume 2 of the popular UNIX Network Programming series
POSIX Programmer's Guide By Donald Lewine ISBN: 0-937175-73-0	I used this book as an introduction to programming in the Unix environment. It was great because it focuses on writing portable code, and contains a lot a good general programming advice. If you are interested in portability, you will find this book a good read.

TCP/IP Illustrated, Volume 1 : The Protocols By W. Richard Stevens ISBN: 0201633469	The casual browser might notice that almost all of my recommended books are written by Stevens. This is because I have found his books to be the most useful. This book in particular is a great help when a network application isn't behaving the way you would expect it to. It will help you to understand what is happening at the protocol level so that you can solve problems at the application level.

As an added convenience, you can now search the Amazon site from here.

| FAQ Home || Search FAQ || Your User Profile || About Me |



Download the Self Serve FAQ 0.2 Here. (Zip file Here)

This is not the UNIX Socket FAQ. Questions not related to the Self Serve FAQ software will be deleted. For UNIX network programming questions, please use the UNIX Socket FAQ.

Welcome to the home of the "Self Serve FAQ"! The "Self Serve FAQ" is software to help maintain lists of questions and answers using the web. It is available under the GNU GPL at no cost with source code.

The software is a set of PHP scripts, and requires an Apache server with mod_php4 or mod_php3 built with MySQL support.

Uncategorized Questions:

14. tring it NOW
15. rt
16. Microsoft is super!
17. Content Management
18. test
19. This is Mike's test
20. Subjects can't have single quote without \
21. prova
22. fucker, fucker
23. Socket Implementation.
24. wella
25. arrrèt
26. ’†•§
27.
28. test de faq
29.
30. bug with other character sets
31. c source code to get connect to the internet and get the contents
32. c source code to get connect to the internet and get the contents
33. pop
34. stst
35.
36. Bugs
37. Using this SSFAQ
38. ƒeƒXƒg
39.
40. gdfs
41. Guess what??
42. try
43. Hola, estoy haciendo una pregunta
44. no listen
45.
46. Anything Goes

47. ÇáÖáÇã Úáíßã
48. Bug in header when you view the questions...
49. ƒeƒXƒg
50. ÓÄÇá
51. Prova III
52. who is the man ?
53. The bototm linee
54. Configuration
55. dfdfdf
56. problem
57. query about socket
58. This is a test
59. Where will this question end up?
60. How to categorise questions?
61. Running create.sql
62. Is this a good idea?
63. A new q from me.
64. How to distinguish?
65. dfsgdfgggggggggggggggggg
66. test
67.
68. teste
69. dfd
70. test
71. Test
72. comment faire un essai
73. tertre
74. Should you give a blind man a digital watch?
75.
76. En liten provtur
77. if my client encounter the firewall....
78. Eesti keeles polegi

Categorized Questions:

1. General Questions
    1. What is the 'Self Serve FAQ'?
    2. What are the system requirements?
    3. Where can I get it?
2. From the Users Perspective
    1. Place test comments here
    2. Can users specify which section they want to use?
    3. What is beind the different comment types ?
3. Administering the Self Serve FAQ
    1. Why does it look like this? How can I customize it?
    2. Why MySQL? Can I use another database?
    3. Creating the database and user for MySQL
    4. Problem with a proxy server?
4. Requests for Features
    1. Some possible additions maybe ?
    2. Faq layout
5. Bugs
    1. Cannot get to the save or delete buttons
    2. Static version of FAQ contains blank entries
    3. PHP4 Installed, now the FAQ is not working.
    4. Problem with magic_quotes and addslashes()

Add Your Own Question to the FAQ

| Search FAQ || Your User Profile || About Me |

Unix Socket FAQ

Welcome to the home of the UNIX Socket FAQ!

There's some really major "under the hood" changes I've just made to the FAQ software that drives this site. I've seperated the content from the functionality, and I'm releasing the source under the GPL. I've named the software "Self Serve FAQ" because of the way it empowers the users to add questions and provide answers without intervention from the FAQ maintainer. I've created a new site on lcg.org for it at http://www.lcg.org/ssfaq/ and you can find out more about it there.

I've added a couple of new features. First, you can download a single file version of the faq here. Second, the main page now shows the most recent updates. Let me know if you notice any quirks.

I've "upgraded" the sock-faq to the new software, so please let me know if you find any problems by posting to the new ssfaq site. Hopefully I'll get any problems worked out early on.

My mirrors appear to have dissapeared, and it may be because they were difficult to keep up-to-date before. One of the first things I want to work on with the new software is making the whole mirror thing easier, so I hope to have mirrors available soon.

If you are looking for Dr. Charles Campbell's Simple Sockets Library, you can download version 2.09 here.

Categorized Questions:

1. General Information and Concepts

1. What's new?

You can find out what is new by looking at the main page for questions that have been recently submitted. You can also set up a user profile for yourself that will allow the main page to tell you what questions have been added, and which questions have new comments since your last visit.

2. About this FAQ

This FAQ is maintained by Vic Metcalfe (vic@acm.org), with lots of assistance from Andrew Gierth (andrew@erlenstar.demon.co.uk). I am depending on the true wizards to fill in the details, and correct my (no doubt) plentiful mistakes. The code examples in this FAQ are written to be easy to follow and understand. It is up to the reader to make them as efficient as required. I started this faq because after reading comp.unix.programmer for a short time, it became evident that a FAQ for sockets was needed.

The FAQ is available at the following locations:

Usenet: (Posted on the 21st of each month)

news.answers, comp.answers, comp.unix.answers, comp.unix.programmer

FTP:

ftp://rtfm.mit.edu/pub/usenet/news.answers/unix-faq/socket

WWW:

http://www.ibrado.com/sock-faq
http://kipper.york.ac.uk/~vic/sock-faq
http://www.ntua.gr/sock-faq

Please email me if you would like to correct or clarify an answer. I would also like to hear from you if you would like me to add a question to the list. I may not be able to answer it, but I can add it in the hopes that someone else will submit an answer. Every hour I seem to be getting even busier, so if I am slow to respond to your email, please be patient. If more than a week passes you may want to send me another one as I often put messages aside for later and then forget about them. I'll have to work on dealing with my mail better, but until then feel free to pester me a little bit.

3. Who is this FAQ for?

This FAQ is for C programmers in the Unix environment. It is not intended for WinSock programmers, or for Perl, Java, etc. I have nothing against Windows or Perl, but I had to limit the scope of the FAQ for the first draft. In the future, I would really like to provide examples for Perl, Java, and maybe others. For now though I will concentrate on correctness and completeness for C.

This version of the FAQ will only cover sockets of the AF_INET family, since this is their most common use. Coverage of other types of sockets may be added later.

4. What are Sockets?

Sockets are just like "worm holes" in science fiction. When things go into one end, they (should) come out of the other. Different kinds of sockets have different properties. Sockets are either connection-oriented or connectionless. Connection-oriented sockets allow for data to flow back and forth as needed, while connectionless sockets (also known as datagram sockets) allow only one message at a time to be transmitted, without an open connection. There are also different socket families. The two most common are `AF_INET` for internet connections, and `AF_UNIX` for unix IPC (interprocess communication). As stated earlier, this FAQ deals only with `AF_INET` sockets.

5. How do Sockets Work?

The implementation is left up to the vendor of your particular unix, but from the point of view of the programmer, connection-oriented sockets work a lot like files, or pipes. The most noticeable difference, once you have your file descriptor is that `read()` or `write()` calls may actually read or write fewer bytes than requested. If this happens, then you will have to make a second call for the rest of the data. There are examples of this in the source code that accompanies the faq.

6. Where can I get source code for the book [book title]?

Here is a list of the places I know to get source code for network programming books. It is very short, so please mail me with any others you know of.

Title: Unix Network Programming
Author: W. Richard Stevens (rstevens@kohala.com)
Publisher: Prentice Hall, Inc.
ISBN: 0-13-949876-1
URL: http://www.kohala.com/~rstevens

Title: Power Programming with RPC
Author: John Bloomer
Publisher: O'Reilly & Associates, Inc.
ISBN: 0-937175-77-3
URL: ftp://ftp.uu.net/published/oreilly/nutshell/rpc/rpc.tar.Z

Recommended by: Lokmanm Merican (lokmanm#pop4.jaring.my@199.1.1.88)
Title: UNIX PROGRAM DEVELOPMENT for IBM PC'S Including OSF/Motif
Author: Thomas Yager
Publisher: Addison Wesley, 1991

ISBN: 0-201-57727-5

7. Where can I get more information?

I keep a copy of the resources I know of on my socks page on the web. I don't remember where I got most of these items, but some day I'll check out their sources, and provide ftp information here. For now, you can get them at http://www.ibrado.com/sock-faq.

There is a TCP/IP FAQ which can be found at http://www.dc.net/ilazar/tcpipfaq/default.htm

8. Where can I get the sample source code?

The sample source code is no longer included in the faq. To get it, please download it from one of the unix-socket-faq www pages:

> http://www.ibrado.com/sock-faq
> http://kipper.york.ac.uk/~vic/sock-faq
> http://www.ntua.gr/sock-faq

If you don't have web access, you can ftp it with ftpmail by following the following instructions.

To get the sample source by mail, send mail to ftpmail@decwrl.dec.com, with no subject line and a body like this:

```
reply
connect ftp.zymsys.com
binary
uuencode
get pub/sockets/examples.tar.gz
quit
```

Save the reply as examples.uu, and type:

```
% uudecode examples.uu
% gunzip examples.tar.gz
% tar xf examples.tar
```

This will create a directory called socket-faq-examples which contains the sample code from this faq, plus a sample client and server for both tcp and udp.

Note that this package requires the gnu unzip program to be installed on your system. It is very common, but if you don't have it you can get the source for it from:

> ftp://prep.ai.mit.edu/pub/gnu/gzip-1.2.4.tar

If you don't have ftp access, you can obtain it in a way similar to obtaining the sample source. I'll leave the exact changes to the body of the message as an excersise for the reader.

2. Questions regarding both Clients and Servers (TCP/SOCK_STREAM)

1. How can I tell when a socket is closed on the other end?

From Andrew Gierth (andrew@erlenstar.demon.co.uk):

AFAIK:

If the peer calls `close()` or exits, without having messed with `SO_LINGER`, then our calls to `read()` should return 0. It is less clear what happens to `write()` calls in this case; I would expect `EPIPE`, not on the next call, but the one after.

If the peer reboots, or sets `l_onoff = 1, l_linger = 0` and then closes, then we should get `ECONNRESET` (eventually) from `read()`, or `EPIPE` from `write()`.

I should also point out that when `write()` returns `EPIPE`, it also raises the `SIGPIPE` signal - you never see the

EPIPE error unless you handle or ignore the signal.

If the peer remains unreachable, we should get some other error.

I don't think that write() can legitimately return 0. read() should return 0 on receipt of a FIN from the peer, and on all following calls.

So yes, you **must** expect read() to return 0.

As an example, suppose you are receiving a file down a TCP link; you might handle the return from read() like this:

```
rc = read(sock,buf,sizeof(buf));
if (rc > 0)
{
    write(file,buf,rc);
    /* error checking on file omitted */
}
else if (rc == 0)
{
    close(file);
    close(sock);
    /* file received successfully */
}
else /* rc < 0 */
{
    /* close file and delete it, since data is not complete
       report error, or whatever */
}
```

2. What's with the second parameter in bind()?

The man page shows it as "struct sockaddr *my_addr". The sockaddr struct though is just a place holder for the structure it really wants. You have to pass different structures depending on what kind of socket you have. For an AF_INET socket, you need the sockaddr_in structure. It has three fields of interest:

sin_family

Set this to AF_INET.

sin_port

The network byte-ordered 16 bit port number

sin_addr

The host's ip number. This is a struct in_addr, which contains only one field, s_addr which is a u_long.

3. How do I get the port number for a given service?

Use the getservbyname() routine. This will return a pointer to a servent structure. You are interested in the s_port field, which contains the port number, with correct byte ordering (so you don't need to call htons() on it). Here is a sample routine:

```
/* Take a service name, and a service type, and return a port number.  If the
   service name is not found, it tries it as a decimal number.  The number
   returned is byte ordered for the network. */
int atoport(char *service, char *proto) {
  int port;
  long int lport;
  struct servent *serv;
  char *errpos;
```

```
  /* First try to read it from /etc/services */
  serv = getservbyname(service, proto);
  if (serv != NULL)
    port = serv->s_port;
  else { /* Not in services, maybe a number? */
    lport = strtol(service,&errpos,0);
    if ( (errpos[0] != 0) || (lport < 1) || (lport > 5000) )
      return -1; /* Invalid port address */
    port = htons(lport);
  }
  return port;
}
```

4. If bind() fails, what should I do with the socket descriptor?

If you are exiting, I have been assured by Andrew that all unixes will close open file descriptors on exit. If you are not exiting though, you can just close it with a regular `close()` call.

5. How do I properly close a socket?

This question is usually asked by people who try `close()`, because they have seen that that is what they are supposed to do, and then run netstat and see that their socket is still active. Yes, `close()` is the correct method. To read about the TIME_WAIT state, and why it is important, refer to 2.7 Please explain the TIME_WAIT state..

6. When should I use shutdown()?

From Michael Hunter (mphunter@qnx.com):

`shutdown()` is useful for deliniating when you are done providing a request to a server using TCP. A typical use is to send a request to a server followed by a `shutdown()`. The server will read your request followed by an `EOF` (read of 0 on most unix implementations). This tells the server that it has your full request. You then go read blocked on the socket. The server will process your request and send the necessary data back to you followed by a close. When you have finished reading all of the response to your request you will read an `EOF` thus signifying that you have the whole response. It should be noted the TTCP (TCP for Transactions -- see R. Steven's home page) provides for a better method of tcp transaction management.

S.Degtyarev (deg@sunsr.inp.nsk.su) wrote a nice in-depth message to me about this. He shows a practical example of using shutdown() to aid in synchronization of client processes when one is the "reader" process, and the other is the "writer" process. A portion of his message follows:

Sockets are very similar to pipes in the way they are used for data transfer and client/server transactions, but not like pipes they are bidirectional. Programs that use sockets often `fork()` and each process inherits the socket descriptor. In pipe based programs it is strictly recommended to close all the pipe ends that are not used to convert the pipe line to one-directional data stream to avoid data losses and deadlocks. With the socket there is no way to allow one process only to send data and the other only to receive so you should always keep in mind the consequences.

Generally the difference between `close()` and `shutdown()` is: `close()` closes the socket id for the process but the connection is still opened if another process shares this socket id. The connection stays opened both for read and write, and sometimes this is very important. `shutdown()` breaks the connection for all processes sharing the socket id. Those who try to read will detect `EOF`, and those who try to write will reseive `SIGPIPE`, possibly delayed while the kernel socket buffer will be filled. Additionally, `shutdown()` has a second argument which denotes how to close the connection: 0 means to disable further reading, 1 to disable writing and 2 disables both.

The quick example below is a fragment of a very simple client process. After establishing the connection with the server it forks. Then child sends the keyboard input to the server until `EOF` is received and the parent receives answers from the server.

```
/*
 *      Sample client fragment,
 *      variables declarations and error handling are omitted
```

```
*/
        s=connect(...);

        if( fork() ){   /*      The child, it copies its stdin to
                                        the socket              */
                while( gets(buffer) >0)
                        write(s,buf,strlen(buffer));

                close(s);
                exit(0);
                }

        else {          /* The parent, it receives answers  */
                while( (l=read(s,buffer,sizeof(buffer)){
                        do_something(l,buffer);

                /* Connection break from the server is assumed  */
                /* ATTENTION: deadlock here                     */
                wait(0); /* Wait for the child to exit          */
                exit(0);
                }
```

What do we expect? The child detects an `EOF` from its `stdin`, it closes the socket (assuming connection break) and exits. The server in its turn detects `EOF`, closes connection and exits. The parent detects `EOF`, makes the `wait()` system call and exits. What do we see instead? The socket instance in the parent process is still opened for writing and reading, though the parent never writes. The server never detects EOF and waits for more data from the client forever. The parent never sees the connection is closed and hangs forever and the server hangs too. Unexpected deadlock! (any deadlock is unexpected though :-)

You should change the client fragment as follows:

```
                if( fork() ) {  /* The child                    */
                        while( gets(buffer) }
                                write(s,buffer,strlen(buffer));

                                shutdown(s,1); /* Break the connection
        for writing, The server will detect EOF now. Note: reading from
        the socket is still allowed. The server may send some more data
        after receiving EOF, why not? */
                        exit(0);
                        }
```

I hope this rough example explains the troubles you can have with client/server syncronization. Generally you should always remember all the instances of the particular socket in all the processes that share the socket and close them all at once if you whish to use close() or use shutdown() in one process to break the connection.

7. Please explain the TIME_WAIT state.

Remember that TCP guarantees all data transmitted will be delivered, if at all possible. When you close a socket, the server goes into a TIME_WAIT state, just to be really really sure that all the data has gone through. When a socket is closed, both sides agree by sending messages to each other that they will send no more data. This, it seemed to me was good enough, and after the handshaking is done, the socket should be closed. The problem is two-fold. First, there is no way to be sure that the last ack was communicated successfully. Second, there may be "wandering duplicates" left on the net that must be dealt with if they are delivered.

Andrew Gierth (andrew@erlenstar.demon.co.uk) helped to explain the closing sequence in the following usenet posting:

Assume that a connection is in ESTABLISHED state, and the client is about to do an orderly release. The client's

sequence no. is Sc, and the server's is Ss. The pipe is empty in both directions.

```
Client                                                  Server
======                                                  ======
ESTABLISHED                                             ESTABLISHED
(client closes)
ESTABLISHED                                             ESTABLISHED
              ------->>
FIN_WAIT_1
             <<--------
FIN_WAIT_2                                              CLOSE_WAIT
             <<--------   (server closes)
                                                        LAST_ACK
             , ------->>
TIME_WAIT                                               CLOSED
(2*msl elapses...)
CLOSED
```

Note: the +1 on the sequence numbers is because the FIN counts as one byte of data. (The above diagram is equivalent to fig. 13 from RFC 793).

Now consider what happens if the last of those packets is dropped in the network. The client has done with the connection; it has no more data or control info to send, and never will have. But the server does not know whether the client received all the data correctly; that's what the last ACK segment is for. Now the server may or may not *care* whether the client got the data, but that is not an issue for TCP; TCP is a reliable rotocol, and *must* distinguish between an orderly connection **close** where all data is transferred, and a connection **abort** where data may or may not have been lost.

So, if that last packet is dropped, the server will retransmit it (it is, after all, an unacknowledged segment) and will expect to see a suitable ACK segment in reply. If the client went straight to CLOSED, the only possible response to that retransmit would be a RST, which would indicate to the server that data had been lost, when in fact it had not been.

(Bear in mind that the server's FIN segment may, additionally, contain data.)

DISCLAIMER: This is my interpretation of the RFCs (I have read all the TCP-related ones I could find), but I have not attempted to examine implementation source code or trace actual connections in order to verify it. I am satisfied that the logic is correct, though.

More commentarty from Vic:

The second issue was addressed by Richard Stevens (rstevens@noao.edu, author of "Unix Network Programming", see 1.6 Where can I get source code for the book [book title]?). I have put together quotes from some of his postings and email which explain this. I have brought together paragraphs from different postings, and have made as few changes as possible.

From Richard Stevens (rstevens@noao.edu):

If the duration of the TIME_WAIT state were just to handle TCP's full-duplex close, then the time would be much smaller, and it would be some function of the current RTO (retransmission timeout), not the MSL (the packet lifetime).

A couple of points about the TIME_WAIT state.

- The end that sends the first FIN goes into the TIME_WAIT state, because that is the end that sends the final ACK. If the other end's FIN is lost, or if the final ACK is lost, having the end that sends the first FIN maintain state about the connection guarantees that it has enough information to retransmit the final ACK.
- Realize that TCP sequence numbers wrap around after 2**32 bytes have been transferred. Assume a connection between A.1500 (host A, port 1500) and B.2000. During the connection one segment is lost and retransmitted. But the segment is not really lost, it is held by some intermediate router and then re-injected into the network. (This is called a "wandering duplicate".) But in the time between the packet being lost & retransmitted, and then reappearing, the connection is closed (without any problems) and then another connection is established between

the same host, same port (that is, A.1500 and B.2000; this is called another "incarnation" of the connection). But the sequence numbers chosen for the new incarnation just happen to overlap with the sequence number of the wandering duplicate that is about to reappear. (This is indeed possible, given the way sequence numbers are chosen for TCP connections.) Bingo, you are about to deliver the data from the wandering duplicate (the previous incarnation of the connection) to the new incarnation of the connection. To avoid this, you do not allow the same incarnation of the connection to be reestablished until the TIME_WAIT state terminates. Even the TIME_WAIT state doesn't complete solve the second problem, given what is called TIME_WAIT assassination. RFC 1337 has more details.

- The reason that the duration of the TIME_WAIT state is 2*MSL is that the maximum amount of time a packet can wander around a network is assumed to be MSL seconds. The factor of 2 is for the round-trip. The recommended value for MSL is 120 seconds, but Berkeley-derived implementations normally use 30 seconds instead. This means a TIME_WAIT delay between 1 and 4 minutes. Solaris 2.x does indeed use the recommended MSL of 120 seconds.

A wandering duplicate is a packet that appeared to be lost and was retransmitted. But it wasn't really lost ... some router had problems, held on to the packet for a while (order of seconds, could be a minute if the TTL is large enough) and then re-injects the packet back into the network. But by the time it reappears, the application that sent it originally has already retransmitted the data contained in that packet.

Because of these potential problems with TIME_WAIT assassinations, one should *not* avoid the TIME_WAIT state by setting the `SO_LINGER` option to send an RST instead of the normal TCP connection termination (FIN/ACK/FIN/ACK). The TIME_WAIT state is there for a reason; it's your friend and it's there to help you :-)

I have a long discussion of just this topic in my just-released "TCP/IP Illustrated, Volume 3". The TIME_WAIT state is indeed, one of the most misunderstood features of TCP.

I'm currently rewriting "Unix Network Programming" (see 1.6 Where can I get source code for the book [book title]?). and will include lots more on this topic, as it is often confusing and misunderstood.

An additional note from Andrew:

Closing a socket: if `SO_LINGER` has not been called on a socket, then `close()` is not supposed to discard data. This is true on SVR4.2 (and, apparently, on all non-SVR4 systems) but apparently **not** on SVR4; the use of either `shutdown()` or `SO_LINGER` seems to be required to guarantee delivery of all data.

8. Why does it take so long to detect that the peer died?

From Andrew Gierth (andrew@erlenstar.demon.co.uk):

Because by default, no packets are sent on the TCP connection unless there is data to send or acknowledge.

So, if you are simply waiting for data from the peer, there is no way to tell if the peer has silently gone away, or just isn't ready to send any more data yet. This can be a problem (especially if the peer is a PC, and the user just hits the Big Switch...).

One solution is to use the `SO_KEEPALIVE` option. This option enables periodic probing of the connection to ensure that the peer is still present. **BE WARNED:** the default timeout for this option is **AT LEAST 2 HOURS**. This timeout can often be altered (in a system-dependent fashion) but not normally on a per-connection basis (AFAIK).

RFC1122 specifies that this timeout (if it exists) must be configurable. On the majority of Unix variants, this configuration may only be done globally, affecting all TCP connections which have keepalive enabled. The method of changing the value, moreover, is often difficult and/or poorly documented, and in any case is different for just about every version in existence.

If you must change the value, look for something resembling `tcp_keepidle` in your kernel configuration or network options configuration.

If you're *sending* to the peer, though, you have some better guarantees; since sending data implies receiving ACKs from the peer, then you will know after the retransmit timeout whether the peer is still alive. But the retransmit timeout is designed to allow for various contingencies, with the intention that TCP connections are not dropped simply as a result of minor network upsets. So you should still expect a delay of several minutes before getting notification of the failure.

The approach taken by most application protocols currently in use on the Internet (e.g. FTP, SMTP etc.) is to implement read timeouts on the server end; the server simply gives up on the client if no requests are received in a given time period (often of the order of 15 minutes). Protocols where the connection is maintained even if idle for long periods have two choices:

1. use `SO_KEEPALIVE`
2. use a higher-level keepalive mechanism (such as sending a null request to the server every so often).

9. What are the pros/cons of select(), non-blocking I/O and SIGIO?

Using non-blocking I/O means that you have to poll sockets to see if there is data to be read from them. Polling should usually be avoided since it uses more CPU time than other techniques.

Using `SIGIO` allows your application to do what it does and have the operating system tell it (with a signal) that there is data waiting for it on a socket. The only drawback to this soltion is that it can be confusing, and if you are dealing with multiple sockets you will have to do a `select()` anyway to find out which one(s) is ready to be read.

Using `select()` is great if your application has to accept data from more than one socket at a time since it will block until any one of a number of sockets is ready with data. One other advantage to `select()` is that you can set a time-out value after which control will be returned to you whether any of the sockets have data for you or not.

10. Why do I get EPROTO from read()?

From Steve Rago (sar@plc.com):

`EPROTO` means that the protocol encountered an unrecoverable error for that endpoint. `EPROTO` is one of those catch-all error codes used by STREAMS-based drivers when a better code isn't available.

And an addition note from Andrew (andrew@erlenstar.demon.co.uk):

Not quite to do with `EPROTO` from `read()`, but I found out once that on some STREAMS-based implementations, `EPROTO` could be returned by `accept()` if the incoming connection was reset before the accept completes.

On some other implementations, accept seemed to be capable of blocking if this occured. This is important, since if `select()` said the listening socket was readable, then you would normally expect *not* to block in the `accept()` call. The fix is, of course, to set nonblocking mode on the listening socket if you are going to use `select()` on it.

11. How can I force a socket to send the data in its buffer?

From Richard Stevens (rstevens@noao.edu):

You can't force it. Period. TCP makes up its own mind as to when it can send data. Now, *normally* when you call `write()` on a TCP socket, TCP will indeed send a segment, but there's no guarantee and no way to force this. There are *lots* of reasons why TCP will not send a segment: a closed window and the Nagle algorithm are two things to come immediately to mind.

(Snipped suggestion from Andrew Gierth to use `TCP_NODELAY`)

Setting this only disables one of the many tests, the Nagle algorithm. But if the original poster's problem is this, then setting this socket option will help.

A quick glance at tcp_output() shows around 11 tests TCP has to make as to whether to send a segment or not.

Now from Dr. Charles E. Campbell Jr. (cec@gryphon.gsfc.nasa.gov):

As you've surmised, I've never had any problem with disabling Nagle's algorithm. Its basically a buffering method; there's a fixed overhead for all packets, no matter how small. Hence, Nagle's algorithm collects small packets together (no more than .2sec delay) and thereby reduces the amount of overhead bytes being transferred. This approach works well for rcp, for example: the .2 second delay isn't humanly noticeable, and multiple users have their small packets more efficiently transferred. Helps in university settings where most folks using the network are using standard tools such as rcp and ftp, and programs such as telnet may use it, too.

However, Nagle's algorithm is pure havoc for real-time control and not much better for keystroke interactive applications

(control-C, anyone?). It has seemed to me that the types of new programs using sockets that people write usually do have problems with small packet delays. One way to bypass Nagle's algorithm selectively is to use "out-of-band" messaging, but that is limited in its content and has other effects (such as a loss of sequentiality) (by the way, out-of-band is often used for that ctrl-C, too).

More from Vic:

So to sum it all up, if you are having trouble and need to flush the socket, setting the `TCP_NODELAY` option will usually solve the problem. If it doesn't, you will have to use out-of-band messaging, but according to Andrew, "out-of-band data has its own problems, and I don't think it works well as a solution to buffering delays (haven't tried it though). It is *not* 'expedited data' in the sense that exists in some other protocols; it is transmitted in-stream, but with a pointer to indicate where it is."

I asked Andrew something to the effect of "**What promises does TCP make about when it will get around to writing data to the network?**" I thought his reply should be put under this question:

Not many promises, but some.

I'll try and quote chapter and verse on this:

References:

RFC 1122, "Requirements for Internet Hosts" (also STD 3)
RFC 793, "Transmission Control Protocol" (also STD 7)

1. The socket interface does not provide access to the TCP PUSH flag.
2. RFC1122 says (4.2.2.2): A TCP MAY implement PUSH flags on SEND calls. If PUSH flags are not implemented, then the sending TCP: (1) must not buffer data indefinitely, and (2) MUST set the PSH bit in the last buffered segment (i.e., when there is no more queued data to be sent).
3. RFC793 says (2.8): When a receiving TCP sees the PUSH flag, it must not wait for more data from the sending TCP before passing the data to the receiving process. [RFC1122 supports this statement.]
4. Therefore, data passed to a `write()` call must be delivered to the peer within a finite time, unless prevented by protocol considerations.
5. There are (according to a post from Stevens quoted in the FAQ [earlier in this answer - Vic]) about 11 tests made which could delay sending the data. But as I see it, there are only 2 that are significant, since things like retransmit backoff are a) not under the programmers control and b) must either resolve within a finite time or drop the connection.

The first of the interesting cases is "window closed" (ie. there is no buffer space at the receiver; this can delay data indefinitely, but only if the receiving process is not actually reading the data that is available)

Vic asks:

OK, it makes sense that if the client isn't reading, the data isn't going to make it across the connection. I take it this causes the sender to block after the recieve queue is filled?

The sender blocks when the socket send buffer is full, so buffers will be full at both ends.

While the window is closed, the sending TCP sends window probe packets. This ensures that when the window finally does open again, the sending TCP detects the fact. [RFC1122, ss 4.2.2.17]

The second interesting case is "Nagle algorithm" (small segments, e.g. keystrokes, are delayed to form larger segments if ACKs are expected from the peer; this is what is disabled with `TCP_NODELAY`)

Vic Asks:

Does this mean that my tcpclient sample should set TCP_NODELAY to ensure that the end-of-line code is indeed put out onto the network when sent?

No. tcpclient.c is doing the right thing as it stands; trying to write as much data as possible in as few calls to `write()` as is feasible. Since the amount of data is likely to be small relative to the socket send buffer, then it is likely (since the connection is idle at that point) that the entire request will require only one call to `write()`, and that the TCP layer will

immediately dispatch the request as a single segment (with the PSH flag, see point 2.2 above).

The Nagle algorithm only has an effect when a second `write()` call is made while data is still unacknowledged. In the normal case, this data will be left buffered until either: a) there is no unacknowledged data; or b) enough data is available to dispatch a full-sized segment. The delay cannot be indefinite, since condition (a) must become true within the retransmit timeout or the connection dies.

Since this delay has negative consequences for certain applications, generally those where a stream of small requests are being sent without response, e.g. mouse movements, the standards specify that an option must exist to disable it. [RFC1122, ss 4.2.3.4]

Additional note: RFC1122 also says:

[DISCUSSION]:

> When the PUSH flag is not implemented on SEND calls, i.e., when the application/TCP interface uses a pure streaming model, responsibility for aggregating any tiny data fragments to form reasonable sized segments is partially borne by the application layer.

So programs should avoid calls to `write()` with small data lengths (small relative to the MSS, that is); it's better to build up a request in a buffer and then do one call to `sock_write()` or equivalent.

The other possible sources of delay in the TCP are not really controllable by the program, but they can only delay the data temporarily.

Vic asks:

By temporarily, you mean that the data will go as soon as it can, and I won't get stuck in a position where one side is waiting on a response, and the other side hasn't recieved the request? (Or at least I won't get stuck forever)

You can only deadlock if you somehow manage to fill up all the buffers in both directions... not easy.

If it is possible to do this, (can't think of a good example though), the solution is to use nonblocking mode, especially for writes. Then you can buffer excess data in the program as necessary.

12. Where can I get a library for programming sockets?

There is the Simple Sockets Library by Charles E. Campbell, Jr. PhD. and Terry McRoberts. The file is called ssl.tar.gz, and you can download it from this faq's home page. For c++ there is the Socket++ library which is on ftp://ftp.virginia.edu/pub/socket++-1.11.tar.gz. There is also C++ Wrappers. The file is called ftp://ftp.huji.ac.il/pub/languages/C++/C++_wrappers.tar.gz. Thanks to Bill McKinnon for tracking it down for me! From http://www.cs.wustl.edu/~schmidt you should be able to find the ACE toolkit. Another C++ library called libtcp++ is also available at http://www.sashanet.com/internet/download.html. PING Software Group has some libraries that include a sockets interface among other things. It seems to be all Java stuff now. You can find their stuff at http://www.nerosworld.com/ping/. Thanks to Reid Judd for hunting that down for us!

Philippe Jounin has developed a cross platform library which includes high level support for http and ftp protocols, with more to come. You can find it at http://perso.magic.fr/jounin-ph/P_tcp4u.htm, and you can find a review of it at http://www6.zdnet.com/cgi-bin/texis/swlib/hotfiles/info.html?fcode=000H4F

I don't have any experience with any of these libraries, so I can't recomend one over the other.

13. How come select says there is data, but read returns zero?

The data that causes select to return is the EOF because the other side has closed the connection. This causes read to return zero. For more information see 2.1 How can I tell when a socket is closed on the other end?

14. Whats the difference between select() and poll()?

From Richard Stevens (rstevens@noao.edu):

The basic difference is that `select()`'s `fd_set` is a bit mask and therefore has some fixed size. It would be possible for the kernel to not limit this size when the kernel is compiled, allowing the application to define `FD_SETSIZE` to

whatever it wants (as the comments in the system header imply today) but it takes more work. 4.4BSD's kernel and the Solaris library function both have this limit. But I see that BSD/OS 2.1 has now been coded to avoid this limit, so it's doable, just a small matter of programming. :-) Someone should file a Solaris bug report on this, and see if it ever gets fixed.

With `poll()`, however, the user must allocate an array of `pollfd` structures, and pass the number of entries in this array, so there's no fundamental limit. As Casper notes, fewer systems have `poll()` than `select`, so the latter is more portable. Also, with original implementations (SVR3) you could not set the descriptor to -1 to tell the kernel to ignore an entry in the `pollfd` structure, which made it hard to remove entries from the array; SVR4 gets around this. Personally, I always use `select()` and rarely `poll()`, because I port my code to BSD environments too. Someone could write an implementation of `poll()` that uses `select()`, for these environments, but I've never seen one. Both `select()` and `poll()` are being standardized by POSIX 1003.1g.

15. How do I send [this] over a socket

Anything other than single bytes of data will probably get mangled unless you take care. For integer values you can use `htons()` and friends, and strings are really just a bunch of single bytes, so those should be OK. Be careful not to send a pointer to a string though, since the pointer will be meaningless on another machine. If you need to send a struct, you should write `sendthisstruct()` and `readthisstruct()` functions for it that do all the work of taking the structure apart on one side, and putting it back together on the other. If you need to send floats, you may have a lot of work ahead of you. You should read RFC 1014 which is about portable ways of getting data from one machine to another (thanks to Andrew Gabriel for pointing this out).

16. How do I use TCP_NODELAY?

First off, be sure you really want to use it in the first place. It will disable the Nagle algorithm (see 2.11 How can I force a socket to send the data in its buffer?), which will cause network traffic to increase, with smaller than needed packets wasting bandwidth. Also, from what I have been able to tell, the speed increase is very small, so you should probably do it without `TCP_NODELAY` first, and only turn it on if there is a problem.

Here is a code example, with a warning about using it from Andrew Gierth:

```
int flag = 1;
int result = setsockopt(sock,            /* socket affected */
                        IPPROTO_TCP,     /* set option at TCP level */
                        TCP_NODELAY,     /* name of option */
                        (char *) &flag,  /* the cast is historical
                                            cruft */
                        sizeof(int));    /* length of option value */
if (result < 0)
   ... handle the error ...
```

`TCP_NODELAY` is for a *specific* purpose; to disable the Nagle buffering algorithm. It should only be set for applications that send frequent small bursts of information without getting an immediate response, where timely delivery of data is required (the canonical example is mouse movements).

17. What exactly does the Nagle algorithm do?

It groups together as much data as it can between ACK's from the other end of the connection. I found this really confusing until Andrew Gierth (andrew@erlenstar.demon.co.uk) drew the following diagram, and explained:

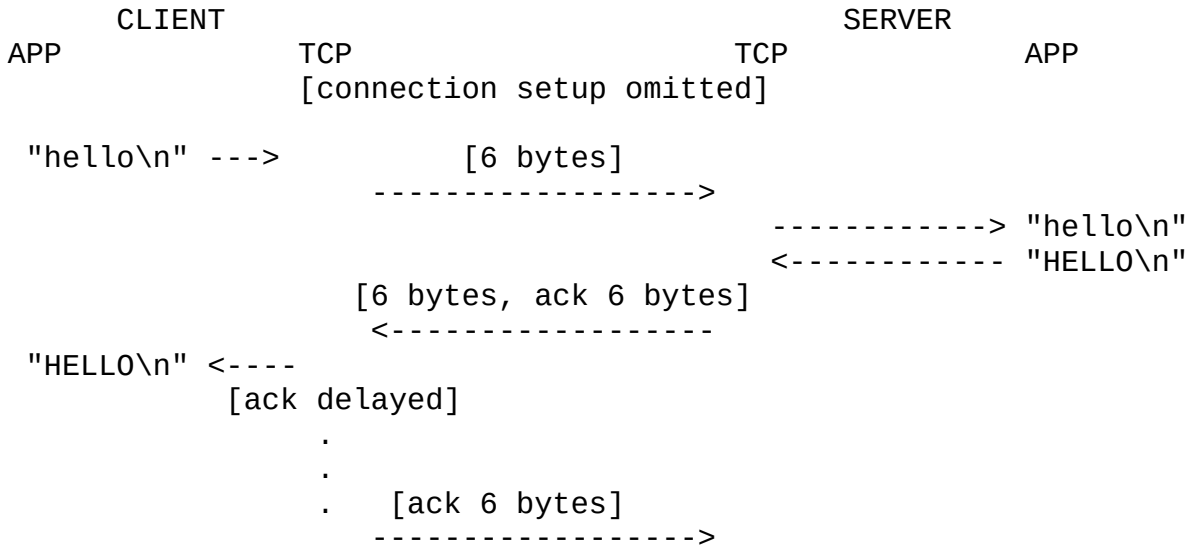
This diagram is not intended to be complete, just to illustrate the point better...

Case 1: client writes 1 byte per `write()` call. The program on host B is tcpserver.c from the FAQ examples.

```
        CLIENT                                  SERVER
    APP             TCP                  TCP             APP
                    [connection setup omitted]
```

Total segments: 5. (If `TCP_NODELAY` was set, could have been up to 10.) Time for response: 2*RTT, plus ack delay.

Case 2: client writes all data with one `write()` call.



Total segments: 3.

Time for response = RTT (therefore minimum possible).

Hope this makes things a bit clearer...

Note that in case 2, you *don't* want the implementation to gratuitously delay sending the data, since that would add straight onto the response time.

18. What is the difference between read() and recv()?

From Andrew Gierth (andrew@erlenstar.demon.co.uk):

`read()` is equivalent to `recv()` with a `flags` parameter of 0. Other values for the `flags` parameter change the behaviour of `recv()`. Similarly, `write()` is equivalent to `send()` with `flags` == 0.

It is unlikely that send()/recv() would be dropped; perhaps someone with a copy of the POSIX drafts for socket calls can check...

Portability note: non-unix systems may not allow `read()/write()` on sockets, but `recv()/send()` are usually ok. This is true on Windows and OS/2, for example.

19. I see that send()/write() can generate SIGPIPE. Is there any advantage to handling the signal, rather than just ignoring it and checking for the EPIPE error?

From Andrew Gierth (andrew@erlenstar.demon.co.uk):

In general, the only parameter passed to a signal handler is the signal number that caused it to be invoked. Some systems have optional additional parameters, but they are no use to you in this case.

My advice is to just ignore `SIGPIPE` as you suggest. That's what I do in just about all of my socket code; errno values are easier to handle than signals (in fact, the first revision of the FAQ failed to mention `SIGPIPE` in that context; I'd got so used to ignoring it...)

There is one situation where you should *not* ignore `SIGPIPE`; if you are going to `exec()` another program with stdout redirected to a socket. In this case it is probably wise to set `SIGPIPE` to `SIG_DFL` before doing the `exec()`.

Jesse Norell has pointed out that if you are using SO_KEEPALIVE to test the connection, and you aren't doing reads or writes very frequently, you might want to leave SIGPIPE enabled so that your server process gets signalled when the system determines your link is dead. Normally though you will just check returns from `read()/write()` and act appropriately.

20. After the chroot(), calls to socket() are failing. Why?

From Andrew Gierth (andrew@erlenstar.demon.co.uk):

On systems where sockets are implemented on top of Streams (e.g. all SysV-based systems, presumably including Solaris), the `socket()` function will actually be opening certain special files in /dev. You will need to create a /dev directory under your fake root and populate it with the required device nodes (only).

Your system documentation may or may not specify exactly which device nodes are required; I can't help you there (sorry). (Editors note: Adrian Hall (adrian@hottub.org) suggested checking the man page for ftpd, which should list the files you need to copy and devices you need to create in the chroot'd environment.)

A less-obvious issue with `chroot()` is if you call `syslog()`, as many daemons do; `syslog()` opens (depending on the system) either a UDP socket, a FIFO or a Unix-domain socket. So if you use it after a `chroot()` call, make sure that you call `openlog()` *before* the chroot.

21. Why do I keep getting EINTR from the socket calls?

This isn't really so much an error as an exit condition. It means that the call was interrupted by a signal. Any call that might block should be wrapped in a loop that checkes for `EINTR`, as is done in the example code (See 1.8. Sample Source Code).

22. When will my application receive SIGPIPE?

From Richard Stevens (rstevens@noao.edu):

Very simple: with TCP you get `SIGPIPE` if your end of the connection has received an RST from the other end. What this also means is that if you were using `select` instead of write, the select would have indicated the socket as being readable, since the RST is there for you to read (read will return an error with `errno` set to `ECONNRESET`).

Basically an RST is TCP's response to some packet that it doesn't expect and has no other way of dealing with. A common case is when the peer closes the connection (sending you a FIN) but you ignore it because you're writing and not reading. (You should be using `select`.) So you write to a connection that has been closed by the other end and the other end's TCP responds with an RST.

23. What are socket exceptions? What is out-of-band data?

Unlike exceptions in C++, socket exceptions do not indicate that an error has occured. Socket exceptions usually refer to the notification that out-of-band data has arrived. Out-of-band data (called "urgent data" in TCP) looks to the application like a separate stream of data from the main data stream. This can be useful for separating two different kinds of data. Note that just because it is called "urgent data" does not mean that it will be delivered any faster, or with higher priorety than data in the in-band data stream. Also beware that unlike the main data stream, the out-of-bound data may be lost if your application can't keep up with it.

24. How can I find the full hostname (FQDN) of the system I'm running on?

From Richard Stevens (rstevens@noao.edu):

Some systems set the hostname to the FQDN and others set it to just the unqualified host name. I know the current BIND FAQ recommends the FQDN, but most Solaris systems, for example, tend to use only the unqualified host name.

Regardless, the way around this is to first get the host's name (perhaps an FQDN, perhaps unaualified). Most systems support the Posix way to do this using `uname()`, but older BSD systems only provide `gethostname()`. Call `gethostbyname()` to find your IP address. Then take the IP address and call `gethostbyaddr()`. The `h_name` member of the `hostent{}` should then be your FQDN.

25. How do I monitor the activity of sockets?

From: Matthias Rabast (matthias.rabast@ubs.com)

How can I find out,

- which sockets have highest throughput ?
- how big is the tcp window size for each socket ?
- how often does a special socket block and go again ?

For monitoring throughput there are tools such as IPAudit that will monitor throughput. I can't remember which tool I used to use for this purpose, but a quick search found IPAudit. I haven't tried it, so let me know if it works, or if you know some better tools.

You can use `netstat -a` under solaris and look at the Swind and Rwind columns for send and recieve window sizes.

I'm not aware of any tools for monitoring how often a socket blocks. Someone please add a comment if you have any suggestions for this.

You could parse the output of snoop/tcpdump to get some of this information. Let me know if you know a good parser and I'll list it here.

3. Writing Client Applications (TCP/SOCK_STREAM)

1. How do I convert a string into an internet address?

If you are reading a host's address from the command line, you may not know if you have an aaa.bbb.ccc.ddd style address, or a host.domain.com style address. What I do with these, is first try to use it as a aaa.bbb.ccc.ddd type address, and if that fails, then do a name lookup on it. Here is an example:

```
/* Converts ascii text to in_addr struct.  NULL is returned if the
   address can not be found. */
struct in_addr *atoaddr(char *address) {
  struct hostent *host;
  static struct in_addr saddr;
```

```
    /* First try it as aaa.bbb.ccc.ddd. */
    saddr.s_addr = inet_addr(address);
    if (saddr.s_addr != -1) {
      return &saddr;
    }
    host = gethostbyname(address);
    if (host != NULL) {
      return (struct in_addr *) *host->h_addr_list;
    }
    return NULL;
  }
```

2. How can my client work through a firewall/proxy server?

If you are running through separate proxies for each service, you shouldn't need to do anything. If you are working through sockd, you will need to "socksify" your application. Details for doing this can be found in the package itself, which is available at:

ftp://coast.cs.purdue.edu/pub/tools/unix/socks/

3. Why does connect() succeed even before my server did an accept()?

From Andrew Gierth (andrew@erlenstar.demon.co.uk):

Once you have done a `listen()` call on your socket, the kernel is primed to accept connections on it. The usual UNIX implementation of this works by *immediately* completing the SYN handshake for any incoming valid SYN segments (connection attempts), creating the socket for the new connection, and keeping this new socket on an internal queue ready for the `accept()` call. So the socket is fully open *before* the accept is done.

The other factor in this is the 'backlog' parameter for `listen()`; that defines how many of these completed connections can be queued at one time. If the specified number is exceeded, then new incoming connects are simply ignored (which causes them to be retried).

4. Why do I sometimes lose a server's address when using more than one server?

From Andrew Gierth (andrew@erlenstar.demon.co.uk):

Take a careful look at struct hostent. Notice that almost everything in it is a pointer? *All* these pointers will refer to statically allocated data.

For example, if you do:

```
struct hostent *host = gethostbyname(hostname);
```

then (as you should know) a subsequent call to `gethostbyname()` will overwrite the structure pointed to by 'host'.

But if you do:

```
struct hostent myhost;
struct hostent *hostptr = gethostbyname(hostname);
if (hostptr) myhost = *host;
```

to make a copy of the `hostent` before it gets overwritten, then it *still* gets clobbered by a subsequent call to `gethostbyname()`, since although `myhost` won't get overwritten, all the data it is pointing to will be.

You can get round this by doing a proper 'deep copy' of the `hostent` structure, but this is tedious. My recommendation would be to extract the needed fields of the `hostent` and store them in your own way.

Robin Paterson (etmrpat@etm.ericsson.se) has added:

It might be nice if you mention MT safe libraries provide complimentary functions for multithreaded programming. On

the solaris machine I'm typing at, we have `gethostbyname` and `gethostbyname_r` (`_r` for reentrant). The main difference is, *you* provide the storage for the hostent struct so you always have a local copy and not just a pointer to the static copy.

5. How can I set the timeout for the connect() system call?

From Richard Stevens (rstevens@noao.edu):

Normally you cannot change this. Solaris does let you do this, on a per-kernel basis with the ndd `tcp_ip_abort_cinterval` parameter.

The easiest way to shorten the connect time is with an `alarm()` around the call to `connect()`. A harder way is to use `select()`, after setting the socket nonblocking. Also notice that you can only shorten the connect time, there's normally no way to lengthen it.

From Andrew Gierth (andrew@erlenstar.demon.co.uk):

First, create the socket and put it into non-blocking mode, then call connect(). There are three possibilities:

- connect succeeds: the connection has been successfully made (this usually only happens when connecting to the same machine)
- connect fails: obvious
- connect returns -1/EINPROGRESS. The connection attempt has begun, but not yet completed.

If the connection succeeds:

- the socket will select() as writable (and will also select as readable if data arrives)

If the connection fails:

- the socket will select as readable *and* writable, but either a read or write will return the error code from the connection attempt. Also, you can use getsockopt(SO_ERROR) to get the error status - but be careful; some systems return the error code in the result parameter of getsockopt, but others (incorrectly) cause the getsockopt call *itself* to fail with the stored value as the error.

6. Should I bind() a port number in my client program, or let the system choose one for me on the connect() call?

From Andrew Gierth (andrew@erlenstar.demon.co.uk):

**** Let the system choose your client's port number ****

The exception to this, is if the server has been written to be picky about what client ports it will allow connections from. Rlogind and rshd are the classic examples. This is usually part of a Unix-specific (and rather weak) authentication scheme; the intent is that the server allows connections only from processes with root privilege. (The weakness in the scheme is that many O/Ss (e.g. MS-DOS) allow anyone to bind any port.)

The `rresvport()` routine exists to help out clients that are using this scheme. It basically does the equivalent of `socket()` + `bind()`, choosing a port number in the range 512..1023.

If the server is not fussy about the *client's* port number, then don't try and assign it yourself in the client, just let `connect()` pick it for you.

If, in a client, you use the naive scheme of starting at a fixed port number and calling `bind()` on consecutive values until it works, then you buy yourself a whole lot of trouble:

The problem is if the server end of your connection does an active close. (E.G. client sends 'QUIT' command to server, server responds by closing the connection). That leaves the client end of the connection in CLOSED state, and the server end in TIME_WAIT state. So after the client exits, there is no trace of the connection on the client end.

Now run the client again. It will pick the same port number, since as far as it can see, it's free. But as soon as it calls `connect()`, the server finds that you are trying to duplicate an existing connection (although one in TIME_WAIT). It is perfectly entitled to refuse to do this, so you get, I suspect, `ECONNREFUSED` from `connect()`. (Some systems may sometimes allow the connection anyway, but you *can't* rely on it.)

This problem is *especially* dangerous because it doesn't show up unless the client and server are on *different* machines. (If they are the same machine, then the client *won't* pick the same port number as before). So you can get bitten well into the development cycle (if you do what I suspect most people do, and test client & server on the same box initially).

Even if your protocol has the client closing first, there are still ways to produce this problem (e.g. kill the server).

7. Why do I get "connection refused" when the server isn't running?

The `connect()` call will only block while it is waiting to establish a connection. When there is no server waiting at the other end, it gets notified that the connection can not be established, and gives up with the error message you see. This is a good thing, since if it were not the case clients might wait for ever for a service which just doesn't exist. Users would think that they were only waiting for the connection to be established, and then after a while give up, muttering something about crummy software under their breath.

8. What does one do when one does not know how much information is comming over the socket? Is there a way to have a dynamic buffer?

This question asked by Niranjan Perera (perera@mindspring.com).

When the size of the incoming data is unknown, you can either make the size of the buffer as big as the largest possible (or likely) buffer, or you can re-size the buffer on the fly during your read. When you `malloc()` a large buffer, most (if not all) varients of unix will only allocate address space, but not physical pages of ram. As more and more of the buffer is used, the kernel allocates physical memory. This means that malloc'ing a large buffer will not waste resources unless that memory is used, and so it is perfectly acceptable to ask for a meg of ram when you expect only a few K.

On the other hand, a more elegant solution that does not depend on the inner workings of the kernel is to use realloc() to expand the buffer as required in say 4K chunks (since 4K is the size of a page of ram on most systems). I may add something like this to sockhelp.c in the example code one day.

9. How can I determine the local port number?

From: Fajun Shi (fajun@cs.msstate.edu):

Hi, my question is: When I write a client, how can I know the port number that the socket bound in my machine?

4. Writing Server Applications (TCP/SOCK_STREAM)

1. How come I get "address already in use" from bind()?

You get this when the address is already in use. (Oh, you figured that much out?) The most common reason for this is that you have stopped your server, and then re-started it right away. The sockets that were used by the first incarnation of the server are still active. This is further explained in 2.7 Please explain the TIME_WAIT state., and 2.5 How do I properly close a socket?.

2. Why don't my sockets close?

When you issue the `close()` system call, you are closing your interface to the socket, not the socket itself. It is up to the kernel to close the socket. Sometimes, for really technical reasons, the socket is kept alive for a few minutes after you close it. It is normal, for example for the socket to go into a TIME_WAIT state, on the server side, for a few minutes. People have reported ranges from 20 seconds to 4 minutes to me. The official standard says that it should be 4 minutes. On my Linux system it is about 2 minutes. This is explained in great detail in 2.7 Please explain the TIME_WAIT state..

3. How can I make my server a daemon?

There are two approaches you can take here. The first is to use inetd to do all the hard work for you. The second is to do all the hard work yourself.

If you use inetd, you simply use `stdin`, `stdout`, or `stderr` for your socket. (These three are all created with `dup()` from the real socket) You can use these as you would a socket in your code. The inetd process will even close the socket for you when you are done. For more information on setting this up, look at the man page for inetd.

If you wish to write your own server, there is a detailed explanation in "Unix Network Programming" by Richard Stevens

(see 1.6 Where can I get source code for the book [book title]?). I also picked up this posting from comp.unix.programmer, by Nikhil Nair (nn201@cus.cam.ac.uk). You may want to add code to ignore SIGPIPE, because if this signal is not dealt with, it will cause your application to exit. (Thanks to ingo@milan2.snafu.de for pointing this out).

```
I worked all this lot out from the GNU C Library Manual (on-line
documentation).  Here's some code I wrote - you can adapt it as necessary:


#include
#include
#include
#include
#include
#include
#include

/* Global variables */
...
volatile sig_atomic_t keep_going = 1; /* controls program termination */


/* Function prototypes: */
...
void termination_handler (int signum); /* clean up before termination */


int
main (void)
{
  ...

  if (chdir (HOME_DIR))         /* change to directory containing data
                                   files */
   {
     fprintf (stderr, "`%s': ", HOME_DIR);
     perror (NULL);
     exit (1);
   }

   /* Become a daemon: */
   switch (fork ())
     {
     case -1:                    /* can't fork */
       perror ("fork()");
       exit (3);
     case 0:                     /* child, process becomes a daemon: */
       close (STDIN_FILENO);
       close (STDOUT_FILENO);
       close (STDERR_FILENO);
       if (setsid () == -1)      /* request a new session (job control) */
         {
           exit (4);
         }
       break;
     default:                    /* parent returns to calling process: */
```

```
        return 0;
      }

    /* Establish signal handler to clean up before termination: */
    if (signal (SIGTERM, termination_handler) == SIG_IGN)
      signal (SIGTERM, SIG_IGN);
    signal (SIGINT, SIG_IGN);
    signal (SIGHUP, SIG_IGN);

    /* Main program loop */
    while (keep_going)
      {
        ...
      }
    return 0;
}

void
termination_handler (int signum)
{
  keep_going = 0;
  signal (signum, termination_handler);
}
```

4. How can I listen on more than one port at a time?

The best way to do this is with the `select()` call. This tells the kernel to let you know when a socket is available for use. You can have one process do i/o with multiple sockets with this call. If you want to wait for a connect on sockets 4, 6 and 10 you might execute the following code snippet:

```
fd_set socklist;

FD_ZERO(&socklist); /* Always clear the structure first. */
FD_SET(4, &socklist);
FD_SET(6, &socklist);
FD_SET(10, &socklist);
if (select(11, NULL, &socklist, NULL, NULL) < 0)
  perror("select");
```

The kernel will notify us as soon as a file descriptor which is less than 11 (the first parameter to `select()`), and is a member of our `socklist` becomes available for writing. See the man page on `select()` for more details.

5. What exactly does SO_REUSEADDR do?

This socket option tells the kernel that even if this port is busy (in the TIME_WAIT state), go ahead and reuse it anyway. If it is busy, but with another state, you will still get an address already in use error. It is useful if your server has been shut down, and then restarted right away while sockets are still active on its port. You should be aware that if any unexpected data comes in, it may confuse your server, but while this is possible, it is not likely.

It has been pointed out that "A socket is a 5 tuple (proto, local addr, local port, remote addr, remote port). SO_REUSEADDR just says that you can reuse local addresses. The 5 tuple still must be unique!" by Michael Hunter (mphunter@qnx.com). This is true, and this is why it is very unlikely that unexpected data will ever be seen by your server. The danger is that such a 5 tuple is still floating around on the net, and while it is bouncing around, a new connection from the same client, on the same system, happens to get the same remote port. This is explained by Richard Stevens in 2.7 Please explain the TIME_WAIT state..

6. What exactly does SO_LINGER do?

On some unixes this does nothing. On others, it instructs the kernel to abort tcp connections instead of closing them

properly. This can be dangerous. If you are not clear on this, see 2.7 Please explain the TIME_WAIT state..

7. What exactly does SO_KEEPALIVE do?

From Andrew Gierth (andrew@erlenstar.demon.co.uk):

The `SO_KEEPALIVE` option causes a packet (called a 'keepalive probe') to be sent to the remote system if a long time (by default, more than 2 hours) passes with no other data being sent or received. This packet is designed to provoke an ACK response from the peer. This enables detection of a peer which has become unreachable (e.g. powered off or disconnected from the net). See 2.8 Why does it take so long to detect that the peer died? for further discussion.

Note that the figure of 2 hours comes from RFC1122, "Requirements for Internet Hosts". The precise value should be configurable, but I've often found this to be difficult. The only implementation I know of that allows the keepalive interval to be set per-connection is SVR4.2.

8. 4.8 How can I bind() to a port number < 1024?

From Andrew Gierth (andrew@erlenstar.demon.co.uk):

The restriction on access to ports < 1024 is part of a (fairly weak) security scheme particular to UNIX. The intention is that servers (for example rlogind, rshd) can check the port number of the client, and if it is < 1024, assume the request has been properly authorised at the client end.

The practical upshot of this, is that binding a port number < 1024 is reserved to processes having an effective UID == root.

This can, occasionally, itself present a security problem, e.g. when a server process needs to bind a well-known port, but does *not* itself need root access (news servers, for example). This is often solved by creating a small program which simply binds the socket, then restores the real userid and `exec()`s the real server. This program can then be made setuid root.

9. How do I get my server to find out the client's address / hostname?

From Andrew Gierth (andrew@erlenstar.demon.co.uk):

After `accept()`ing a connection, use `getpeername()` to get the address of the client. The client's address is of course, also returned on the `accept()`, but it is essential to initialise the address-length parameter before the accept call for this will work.

Jari Kokko (jkokko@cc.hut.fi) has offered the following code to determine the client address:

```
int t;
int len;
struct sockaddr_in sin;
struct hostent *host;

len = sizeof sin;
if (getpeername(t, (struct sockaddr *) &sin, &len) < 0)
        perror("getpeername");
else {
        if ((host = gethostbyaddr((char *) &sin.sin_addr,
                                  sizeof sin.sin_addr,
                                  AF_INET)) == NULL)
            perror("gethostbyaddr");
        else printf("remote host is '%s'\n", host->h_name);
}
```

10. How should I choose a port number for my server?

The list of registered port assignments can be found in STD 2 or RFC 1700. Choose one that isn't already registered, and isn't in /etc/services on your system. It is also a good idea to let users customize the port number in case of conflicts with

other un-registered port numbers in other servers. The best way of doing this is hardcoding a service name, and using `getservbyname()` to lookup the actual port number. This method allows users to change the port your server binds to by simply editing the /etc/services file.

11. What is the difference between SO_REUSEADDR and SO_REUSEPORT?

`SO_REUSEADDR` allows your server to bind to an address which is in a TIME_WAIT state. It does not allow more than one server to bind to the same address. It was mentioned that use of this flag can create a security risk because another server can bind to a the same port, by binding to a specific address as opposed to `INADDR_ANY`. The `SO_REUSEPORT` flag allows multiple processes to bind to the same address provided all of them use the `SO_REUSEPORT` option.

From Richard Stevens (rstevens@noao.edu):

This is a newer flag that appeared in the 4.4BSD multicasting code (although that code was from elsewhere, so I am not sure just who invented the new `SO_REUSEPORT` flag).

What this flag lets you do is rebind a port that is already in use, but only if all users of the port specify the flag. I believe the intent is for multicasting apps, since if you're running the same app on a host, all need to bind the same port. But the flag may have other uses. For example the following is from a post in February:

From Stu Friedberg (stuartf@sequent.com):

> `SO_REUSEPORT` is also useful for eliminating the try-10-times-to-bind hack in ftpd's data connection setup routine. Without `SO_REUSEPORT`, only one ftpd thread can bind to TCP (lhost, lport, `INADDR_ANY`, 0) in preparation for connecting back to the client. Under conditions of heavy load, there are more threads colliding here than the try-10-times hack can accomodate. With `SO_REUSEPORT`, things work nicely and the hack becomes unnecessary.

I have also heard that DEC OSF supports the flag. Also note that under 4.4BSD, if you are binding a multicast address, then `SO_REUSEADDR` is condisered the same as `SO_REUSEPORT` (p. 731 of "TCP/IP Illustrated, Volume 2"). I think under Solaris you just replace `SO_REUSEPORT` with `SO_REUSEADDR`.

From a later Stevens posting, with minor editing:

Basically `SO_REUSEPORT` is a BSD'ism that arose when multicasting was added, even thought it was not used in the original Steve Deering code. I believe some BSD-derived systems may also include it (OSF, now Digital Unix, perhaps?). `SO_REUSEPORT` lets you bind the same address *and* port, but only if all the binders have specified it. But when binding a multicast address (its main use), `SO_REUSEADDR` is considered identical to `SO_REUSEPORT` (p. 731, "TCP/IP Illustrated, Volume 2"). So for portability of multicasting applications I always use `SO_REUSEADDR`.

12. How can I write a multi-homed server?

The original question was actually from Shankar Ramamoorthy (shankar@viman.com):

> *I want to run a server on a multi-homed host. The host is part of two networks and has two ethernet cards. I want to run a server on this machine, binding to a pre-determined port number. I want clients on either subnet to be able to send broadcast packates to the port and have the server receive them.*

And answered by Andrew Gierth (andrew@erlenstar.demon.co.uk):

Your first question in this scenario is, do you need to know which subnet the packet came from? I'm not at all sure that this can be reliably determined in all cases.

If you don't really care, then all you need is one socket bound to `INADDR_ANY`. That simplifies things greatly.

If you *do* care, then you have to bind multiple sockets. You are obviously attempting to do this in your code as posted, so I'll assume you do.

> *I was hoping that something like the following would work. Will it? This is on Sparcs running Solaris 2.4/2.5.*

I don't have access to Solaris, but I'll comment based on my experience with other Unixes.

[Shankar's original code omitted]

What you are doing is attempting to bind all the current hosts unicast addresses as listed in hosts/NIS/DNS. This may or may not reflect reality, but much more importantly, neglects the broadcast addresses. It seems to be the case in the majority of implementations that a socket bound to a unicast address will *not* see incoming packets with broadcast addresses as their destinations.

The approach I've taken is to use `SIOCGIFCONF` to retrieve the list of active network interfaces, and `SIOCGIFFLAGS` and `SIOCGIFBRDADDR` to identify broadcastable interfaces and get the broadcast addresses. Then I bind to each unicast address, each broadcast address, *and to* `INADDR_ANY` *as well*. That last is necessary to catch packets that are on the wire with `INADDR_BROADCAST` in the destination. (`SO_REUSEADDR` is necessary to bind `INADDR_ANY` as well as the specific addresses.)

This gives me very nearly what I want. The wrinkles are:

- I don't assume that getting a packet through a particular socket necessarily means that it actually arrived on that interface.
- I can't tell anything about which subnet a packet originated on if its destination was INADDR_BROADCAST.
- On some stacks, apparently only those with multicast support, I get duplicate incoming messages on the INADDR_ANY socket.

13. How can I read only one character at a time?

This question is usually asked by people who are testing their server with telnet, and want it to process their keystrokes one character at a time. The correct technique is to use a psuedo terminal (pty). More on that in a minute.

According to Roger Espel Llima (espel@drakkar.ens.fr), you can have your server send a sequence of control characters: `0xff 0xfb 0x01 0xff 0xfb 0x03 0xff 0xfd 0x0f3`, which translates to `IAC WILL ECHO IAC WILL SUPPRESS-GO-AHEAD IAC DO SUPPRESS-GO-AHEAD`. For more information on what this means, check out std8, std28 and std29. Roger also gave the following tips:

- This code will suppress echo, so you'll have to send the characters the user types back to the client if you want the user to see them.
- Carriage returns will be followed by a null character, so you'll have to expect them.
- If you get a `0xff`, it will be followed by two more characters. These are telnet escapes.

Use of a pty would also be the correct way to execute a child process and pass the i/o to a socket.

I'll add pty stuff to the list of example source I'd like to add to the faq. If someone has some source they'd like to contribute (without copyright) to the faq which demonstrates use of pty's, please email me!

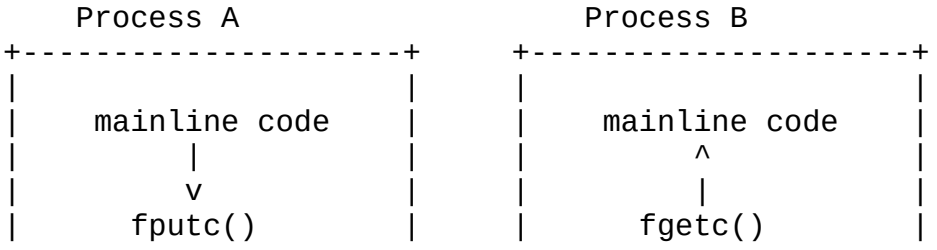
14. I'm trying to exec() a program from my server, and attach my socket's IO to it, but I'm not getting all the data across. Why?

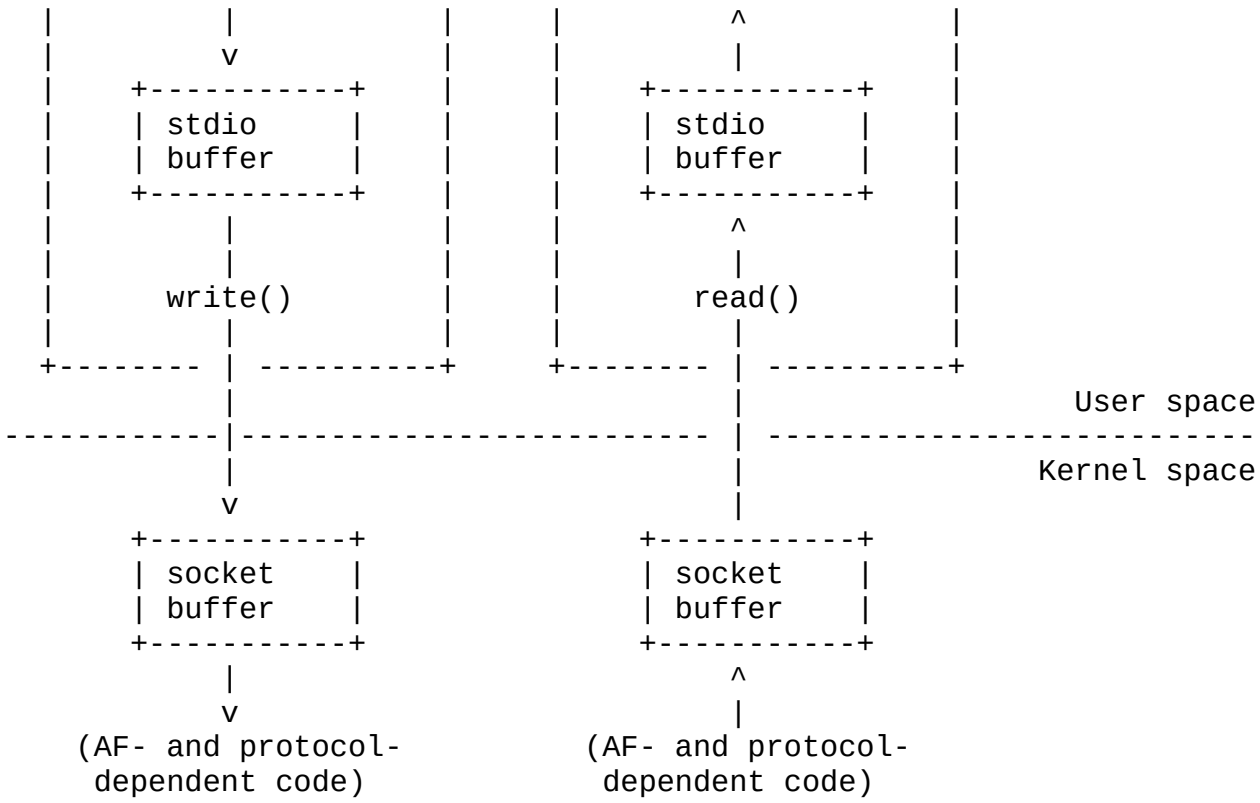
If the program you are running uses `printf()`, etc (streams from `stdio.h`) you have to deal with two buffers. The kernel buffers all socket IO, and this is explained in section 2.11. The second buffer is the one that is causing you grief. This is the stdio buffer, and the problem was well explained by Andrew:

(The short answer to this question is that you want to use a pty rather than a socket; the remainder of this article is an attempt to explain why.)

Firstly, the socket buffer controlled by `setsockopt()` has *absolutly nothing* to do with stdio buffering. Setting it to 1 is guaranteed to be the Wrong Thing(tm).

Perhaps the following diagram might make things a little clearer:





Assuming these two processes are communicating with each other (I've deliberately omitted the actual comms mechanisms, which aren't really relevent), you can see that data written by process A to its stdio buffer is completely inaccessible to process B. Only once the decision is made to flush that buffer to the kernel (via `write()`) can the data actually be delivered to the other process.

The only guaranteed way to affect the buffering within process A is to change the code. However, the default buffering for stdout is controlled by whether the underlying FD refers to a terminal or not; generally, output to terminals is line-buffered, and output to non-terminals (including but not limited to files, pipes, sockets, non-tty devices, etc.) is fully buffered. So the desired effect can usually be achieved by using a pty device; this, for example, is what the 'expect' program does.

Since the stdio buffer (and the `FILE` structure, and everything else related to stdio) is user-level data, it is not preserved across an `exec()` call, hence trying to use `setvbuf()` before the exec is ineffective.

A couple of alternate solutions were proposed by Roger Espel Llima (espel@drakkar.ens.fr):

If it's an option, you can use some standalone program that will just run something inside a pty and buffer its input/output. I've seen a package by the name pty.tar.gz that did that; you could search around for it with archie or AltaVista.

Another option (***warning, evil hack***) , if you're on a system that supports this (SunOS, Solaris, Linux ELF do; I don't know about others) is to, on your main program, `putenv()` the name of a shared executable (*.so) in LD_PRELOAD, and then in that .so redefine some commonly used libc function that the program you're exec'ing is known to use early. There you can 'get control' on the running program, and the first time you get it, do a `setbuf(stdout, NULL)` on the program's behalf, and then call the original libc function with a `dlopen()` + `dlsym()`. And you keep the `dlsym()` value on a static var, so you can just call that the following times.

(Editors note: I still haven't done an expample for how to do pty's, but I hope I will be able to do one after I finish the non-blocking example code.)

5. Writing UDP/SOCK_DGRAM applications

1. When should I use UDP instead of TCP?

UDP is good for sending messages from one system to another when the order isn't important and you don't need all of

the messages to get to the other machine. This is why I've only used UDP once to write the example code for the faq. Usually TCP is a better solution. It saves you having to write code to ensure that messages make it to the desired destination, or to ensure the message ordering. Keep in mind that every additional line of code you add to your project in another line that could contain a potentially expensive bug.

If you find that TCP is too slow for your needs you may be able to get better performance with UDP so long as you are willing to sacrifice message order and/or reliability.

Philippe Jounin would like to add...

> In chapter 5.1 you say UDP allows more throughput than TCP. It is rarely the case if you have to pass several routers.
>
> For instance, if you connect two LANs via X25 (a common way in Europe!), every UDP datagram will :
>
> - establish a Virtual Channel (VC)
> - send the data
> - close the VC,
>
> whereas the VC remains during a TCP dialog.

UDP must be used to multicast messages to more than one other machine at the same time. With TCP an application would have to open separate connections to each of the destination machines and send the message once to each target machine. This limits your application to only communicate with machines that it already knows about.

2. What is the difference between "connected" and "unconnected" sockets?

From Andrew Gierth (andrew@erlenstar.demon.co.uk):

If a UDP socket is unconnected, which is the normal state after a `bind()` call, then `send()` or `write()` are not allowed, since no destination address is available; only `sendto()` can be used to send data.

Calling `connect()` on the socket simply records the specified address and port number as being the desired communications partner. That means that `send()` or `write()` are now allowed; they use the destination address and port given on the connect call as the destination of the packet.

3. Does doing a connect() call affect the receive behaviour of the socket?

From Richard Stevens (rstevens@noao.edu):

Yes, in two ways. First, only datagrams from your "connected peer" are returned. All others arriving at your port are not delivered to you.

But most importantly, a UDP socket must be connected to receive ICMP errors. Pp. 748-749 of "TCP/IP Illustrated, Volume 2" give all the gory details on why this is so.

4. How can I read ICMP errors from "connected" UDP sockets?

If the target machine discards the message because there is no process reading on the requested port number, it sends an ICMP message to your machine which will cause the next system call on the socket to return `ECONNREFUSED`. Since delivery of ICMP messages is not guarenteed you may not recieve this notification on the first transaction.

Remember that your socket must be "connected" in order to receive the ICMP errors. I've been told, and Alan Cox has verified that Linux will return them on "unconnected" sockets. This may cause porting problems if your application isn't ready for it, so Alan tells me they've added a `SO_BSDCOMPAT` flag which can be set for Linux kernels after 2.0.0.

5. How can I be sure that a UDP message is received?

You have to design your protocol to expect a confirmation back from the destination when a message is received. Of course is the confirmation is sent by UDP, then it too is unreliable and may not make it back to the sender. If the sender does not get confirmation back by a certain time, it will have to re-transmit the message, maybe more than once. Now the receiver has a problem because it may have already received the message, so some way of dropping duplicates is required. Most protocols use a message numbering scheme so that the receiver can tell that it has already processed this message and return another confirmation. Confirmations will also have to reference the message number so that the

sender can tell which message is being confirmed. Confused? That's why I stick with TCP.

6. How can I be sure that UDP messages are received in order?

You can't. What you can do is make sure that messages are processed in order by using a numbering system as mentioned in 5.5 How can I be sure that a UDP message is received?. If you need your messages to be received and be received in order you should really consider switching to TCP. It is unlikely that you will be able to do a better job implementing this sort of protocol than the TCP people already have, without a significant investment of time.

7. How often should I re-transmit un-acknowleged messages?

The simplest thing to do is simply pick a fairly small delay such as one second and stick with it. The problem is that this can congest your network with useless traffic if there is a problem on the lan or on the other machine, and this added traffic may only serve to make the problem worse.

A better technique, described with source code in "UNIX Network Programming" by Richard Stevens (see 1.6 Where can I get source code for the book [book title]?), is to use an adaptive timeout with an exponential backoff. This technique keeps statistical information on the time it is taking messages to reach a host and adjusts timeout values accordingly. It also doubles the timeout each time it is reached as to not flood the network with useless datagrams. Richard has been kind enough to post the source code for the book on the web. Check out his home page at http://www.kohala.com/~rstevens.

8. How come only the first part of my datagram is getting through?

This has to do with the maximum size of a datagram on the two machines involved. This depends on the sytems involved, and the MTU (Maximum Transmission Unit). According to "UNIX Network Programming", all TCP/IP implementations must support a minimum IP datagram size of 576 bytes, regardless of the MTU. Assuming a 20 byte IP header and 8 byte UDP header, this leaves 548 bytes as a safe maximum size for UDP messages. The maximum size is 65516 bytes. Some platforms support IP fragmentation which will allow datagrams to be broken up (because of MTU values) and then re-assembled on the other end, but not all implementations support this.

This information is taken from my reading of "UNIX Netowrk Programming" (see 1.6 Where can I get source code for the book [book title]?).

Andrew has pointed out the following regarding large UDP messages:

Another issue is fragmentation. If a datagram is sent which is too large for the network interface it is sent through, then the sending host will fragment it into smaller packets which are reassembled by the receiving host. Also, if there are intervening routers, then they may *also* need to fragment the packet(s), which greatly increases the chances of losing one or more fragments (which causes the entire datagram to be dropped). Thus, large UDP datagrams should be avoided for applications that are likely to operate over routed nets or the Internet proper.

9. Why does the socket's buffer fill up sooner than expected?

From Paul W. Nelson (nelson@thursby.com):

In the traditional BSD socket implementation, sockets that are atomic such as UDP keep received data in lists of mbufs. An mbuf is a fixed size buffer that is shared by various protocol stacks. When you set your receive buffer size, the protocol stack keeps track of how many bytes of mbuf space are on the receive buffer, not the number of actual bytes. This approach is used because the resource you are controlling is really how many mbufs are used, not how many bytes are being held in the socket buffer. (A socket buffer isn't really a buffer in the traditional sense, but a list of mbufs).

For example: Lets assume your UNIX has a small mbuf size of 256 bytes. If your receive socket buffer is set to 4096, you can fit 16 mbufs on the socket buffer. If you receive 16 UDP packets that are 10 bytes each, your socket buffer is full, and you have 160 bytes of data. If you receive 16 UDP packets that are 200 bytes each, your socket buffer is also full, but contains 3200 bytes of data. `FIONREAD` returns the total number of bytes, not the number of messages or bytes of mbufs. Because of this, it is not a good indicator of how full your receive buffer is.

Additionaly, if you receive UDP messages that are 260 bytes, you use up two mbufs, and can only recieve 8 packets before your socket buffer is full. In this case, only 2080 bytes of the 4096 are held in the socket buffer.

This example is greatly simplified, and the real socket buffer algorithm also takes into account some other parameters. Note that some older socket implementations use a 128 byte mbuf.

6. Advanced Socket Programming

1. How would I put my socket in non-blocking mode?

From Andrew Gierth (andrew@erlenstar.demon.co.uk):

Technically, `fcntl(soc, F_SETFL, O_NONBLOCK)` is incorrect since it clobbers all other file flags. Generally one gets away with it since the other flags (`O_APPEND` for example) don't really apply much to sockets. In a similarly rough vein, you would use `fcntl(soc, F_SETFL, 0)` to go back to blocking mode.

To do it right, use `F_GETFL` to get the current flags, set or clear the `O_NONBLOCK` flag, then use `F_SETFL` to set the flags.

And yes, the flag can be changed either way at will.

2. How can I put a timeout on connect()?

Andrew Gierth (andrew@erlenstar.demon.co.uk) has outlined the following procedure for using `select()` with `connect()`, which will allow you to put a timeout on the `connect()` call:

First, create the socket and put it into non-blocking mode, then call `connect()`. There are three possibilities:

- connect succeeds: the connection has been successfully made (this usually only happens when connecting to the same machine)
- connect fails: obvious
- connect returns -1/`EINPROGRESS`. The connection attempt has begun, but not yet completed.

If the connection succeeds:

- the socket will `select()` as writable (and will also select as readable if data arrives)

If the connection fails:

- the socket will select as readable *and* writable, but either a read or write will return the error code from the connection attempt. Also, you can use `getsockopt(SO_ERROR)` to get the error status - but be careful; some systems return the error code in the result parameter of `getsockopt()`, but others (incorrectly) cause the `getsockopt` call *itself* to fail with the stored value as the error.

Sample code that illustrates this can be found in the file `<http://www.lcg.org/sock-faq/connect.c>`.

3. How do I complete a read if I've only read the first part of something, without again calling select()?

4. How to use select routine

5. RAW sockets

6. Restricting a socket to a given interface

7. Receiving all incoming traffic through a RAW-socket?

8. Multicasting

9. getting IP header of a UDP message

10. To fork or not to fork?

7. Sample Source Code

1. Looking for a good C++ socket library

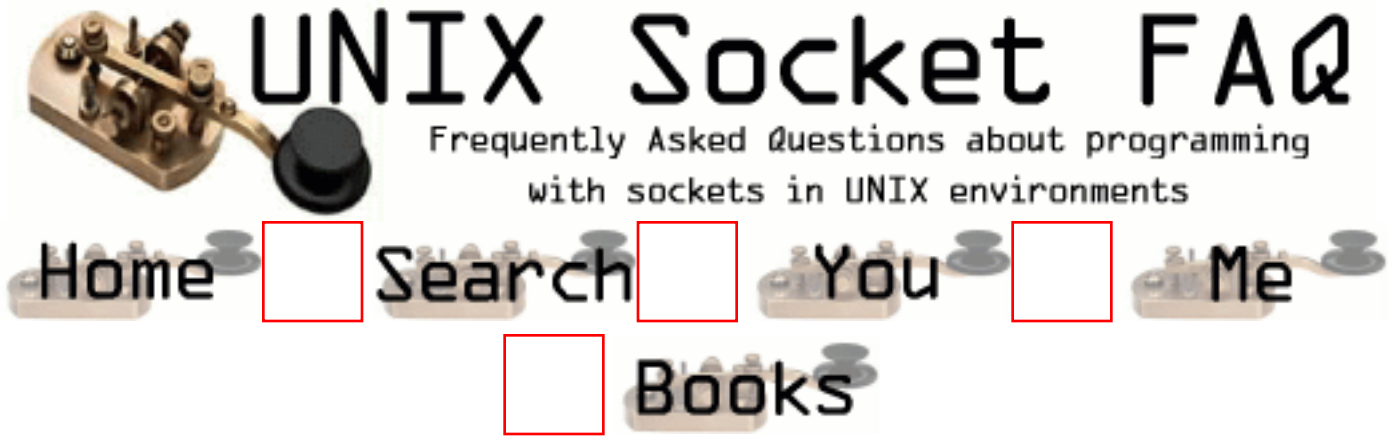
2. perl examples of source code

3. Where is the source code from Richard Stevens' books?

8. Bugs and Strange Behaviour

1. send() hangs up when sending to a switched off computer

2. Error when using inetd



New Questions: How does it know to send back the right information

From: Dwayne

If you have two or more programs sending information to a server and both requesting information back, how does it know that this program should get this and this program should get this. To me its like a walkie talkie everyone is sending on the same channel, should they get everbody response.

Note: Keep in mind the programs sending the inforamtion are started from CGI scripts in different process but there all use the same socket.

From: zhan dong

when you connect to a server using socket, your system
will distritute a unused port like 8922 to you,
you use this port to connect to server (www server, 80)
now, 80 talks to you(8922)
if you start another program, your system will give you another port number like 8923...

From: Dwayne

Thanks I thought that a socket was just a IP address and a port. I never knew that a port is one to many relatioship with sockets

---

From: manikandan

Hi,

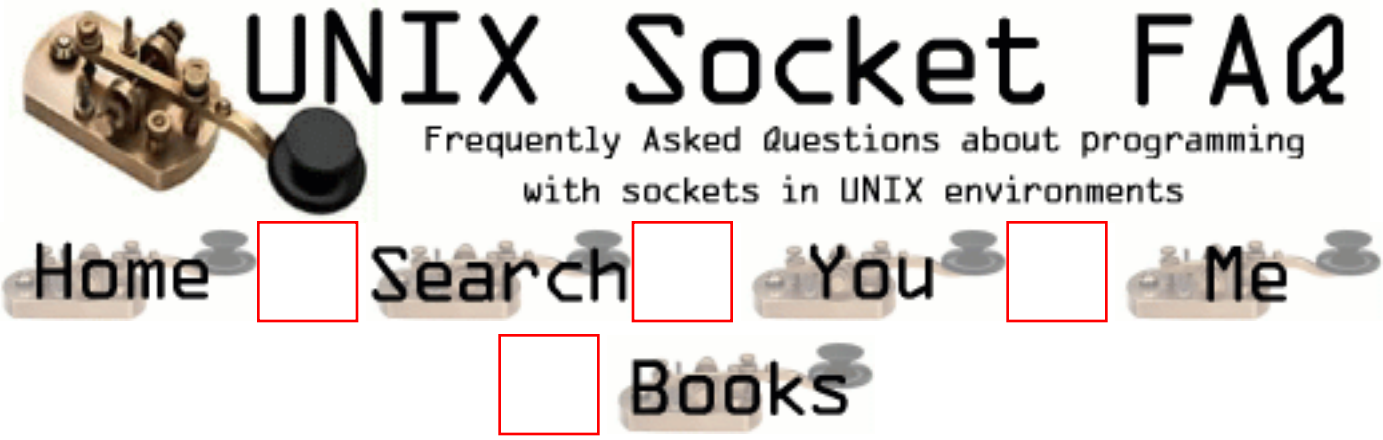
When multiple clients communicate with a server. the server is designed in such a way that for each connection or a request a instance of the server is created, and connection is established with the client and the server instance.

remember new requests come thru socket() system call and the subsequent requests comes thru accept() system call, where both will give socket descriptors.

I hope this is clear

Regards
Mani.

Add a Comment



New Questions: C function to empty a socket

From: JAIME SIEIRO

I am sending some requests on a server and I force a 'shutdown connection' after x seconds if I don't obtain an answer from the server. So, after this moment (of the shutdown), the server is answering with a delay. For exemple: I am sending a request1 and I force a timeout (we don't receive answer1), after that I send a request2 and server answer with answer1 not answer2!!!!! and the continue with this delay of the answer always.... I need some C function (I think) to empty the socket just after the timeout.Could someone help me. Thanks in advance.
Jaime

From: kolp

my english is not well! maybe i understand you wrong !!!

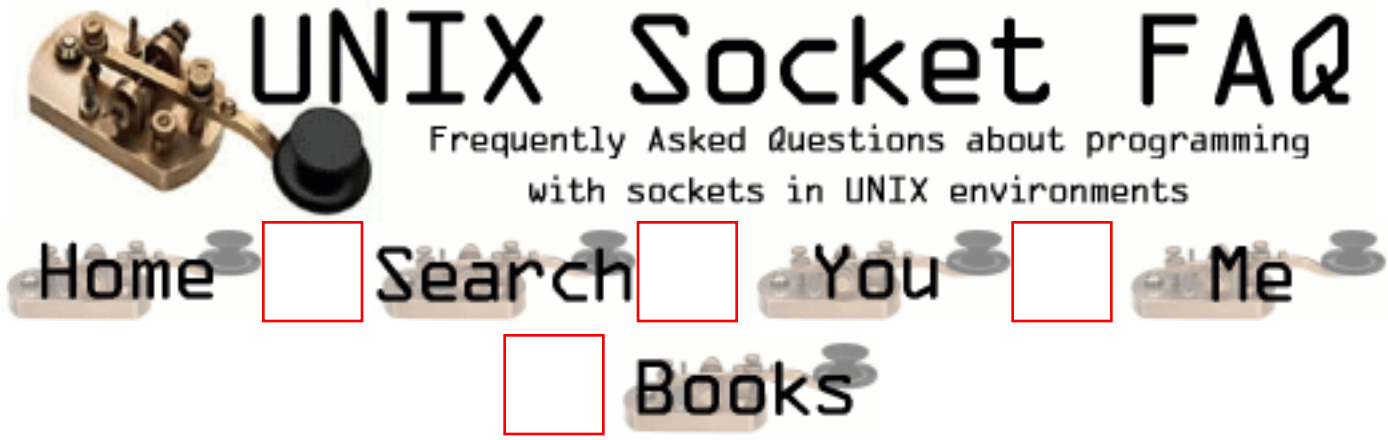
maybe use memset() to clear the struct

From: manikandan

Hi,
to emptry a socket or to clear the sockfd from the descriptor table u can just close(sockfd). if u want to clear the buffer use SO_LINGER by using setsockopt, but if u use this option there wont be any TIMEWAIT state for IP packet.

Regards
Manikandan

Add a Comment



New Questions: confused with backlog of listen()

From: peng cheng

what is the meaning of listen()'s second parameter, backlog?
I have read Rechard's <<unix network programming>>. At first he said "backlog" specified the maximum for the sum of incomplete connection queue and completed connection queue. However, in the next section he said it indicated the maximum for the entry of complieted queue.
I got confused, and I need help.
thanks in advance!

From: Loco

It is the length of the "pending connections" queue.

From: Loco

I'm sorry, i forgot something:

It is the maximum length the queue can grow to. If the queue is full and a new connection arrives then it might get a ECONNREFUSED error.

After you accept a connection it is taken out of that queue.

From: manikandan

Hi,

It the sum of incompelte connection queue and the connection completed queue (SYN_RCVD state,ESTABLISHED state respectivily)

It the queue if full the incoming request if dropped.

Regards
Mani

Add a Comment

New Questions: Problem in Sockets while using Message Queues

From: Girish Mohan

I am using System V message queues along with sockets and everytime I read my Message Queue, I get some activity (Junk) on the socket connection. I am using select() system call to schedule my socket connection.

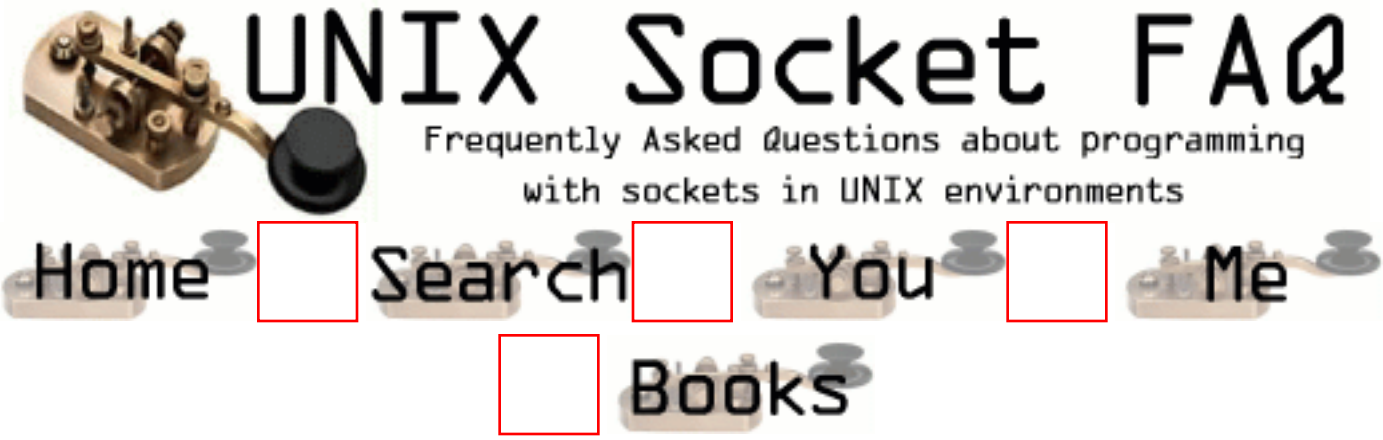
What could be the problem?

Any help is appreciated.

From: manikandan

Hi,
There should not be any problem while using socket with message queues b'cause socket and a MQ are different entities . Can u send some piece of code or some detail explanation.
Mani

Add a Comment



New Questions: I loss data when I receive from a unix socket!!!

From: Sevil Demir

Please Help ME !!!!!!!!!!!!!!!!!!!!!!!!

```
#define MAXDATSIZE 8192
char buf[MAXDATASIZE+1];
int i=0;

socket(...);

bind (...);

listen(....);

new_sock=accept(.....);

i=recv(new_sock,buf,MAXDATASIZE,0);
```

I test this program SCO Unix...At the same machine the result is true, But when I run the program at a different machine ,i= returns 1460 or 536 everytime.... But buf length is longer then i...

So my problem is that , I can`t receive the whole buffer. What should I do ?

From: Extirpater

8192 var ya onu 4096'a dusur hatta 1024 ve 512'yi dene. Bu yavaslatir dogal olarak ama test amacli. Belki kerneldaki max. paket boyutu bi nedenden dolayi daha kucuge ayarlanmis olabilir. ayni kod linux'ta falan calisiyo mu?

From: Extirpater

ha bi de $u var ki. Bunu nasil test ettin? yani buna bilgi gonderen ek bi program yazdiysan sorun ikisinin anlasamamasida olabilir. Ornegin biri cok bilgi gonderiyodur digeri yetisemedigi icin alamiyo butun bilgiyi. her paketten sonra aldim anlamina gelen bisi gondersen bu data bekleyen kisim... fikir sadece...

From:

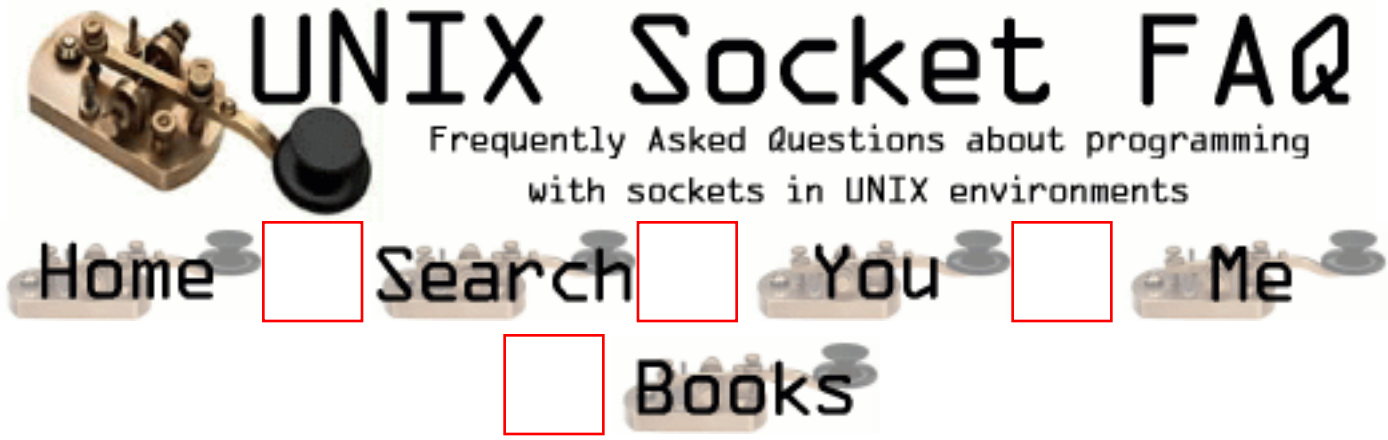
From: manikandan

Hi,

Try to increase the receive buffer using setsockopt with SO_RCVBUF option.

Regards
Mani

Add a Comment



New Questions: tracking socket calls

From: Venkat Raghavan

Hi, I need to track socket calls any application makes and emulate them in my own protocol stack implementation. If there already exists tools to do this for either linux or bsd I'd appreciate any pointers to this. thanks in advance -venkat raghavan_at_csl_dot_sr_dot_com

---

From: Venkat Raghavan

Oops my address should be raghavan_at_csl_dot_sri_dot_com trying my hardest not to get spammed... :-)

From: Hector Lasso

You can trace avery system call by using the "strace" tool in Linux.
If you specify the "-e trace=network" only the network related system calls will be displayed.
You can even trace system calls of already running processes by using the -p option (-p PID_OF_THE_PROCESS).

For more information about strace read the man page.

From: banda

banda maagia

From: Jens Meissner

Hallo,

How can I get Data from the Netwrok (TCP/IP-Packets) directly from the Ehernetcard ?
Is There a Port or socket ?
Is there a request to the Kernel or an C-Programm which gives me the Information into an File where I can get all Binäries ?

Where can I get this Information is there a RFC ?
I use SUSE 7.0

Best regardes JM

From: Deepak Kotian

For hpux-11, you can find tusc command, which is downloadable from the net .This can also all system calls
for a running process or execute a process.

On SunOS -> truss
On AIX -> it is sctrace, this is not freely downloadable
but has to be bought from AIX kernel group.

Thanks and Regards
Deepak

From: xs

Hi,
This can be achieved quite simply under Linux, however I am not sure as to the portability of this method.
You create a shared object (eg, wrap-myprotocol.so) which contains functions with the same prototypes as those of the socket calls. These functions may then call the normal socket functions by loading the library containing those calls using dlopen(3), and then dlsym(3) to get the address of that function in memory.
Once you have created this you can then override the libc/socket lib calls by exporting
LD_PRELOAD="/path/to/libdl.so /path/to/yourlib.so"
and then running the required program.
I've managed to add SSL to virtually any program using this method. It works quite well :)

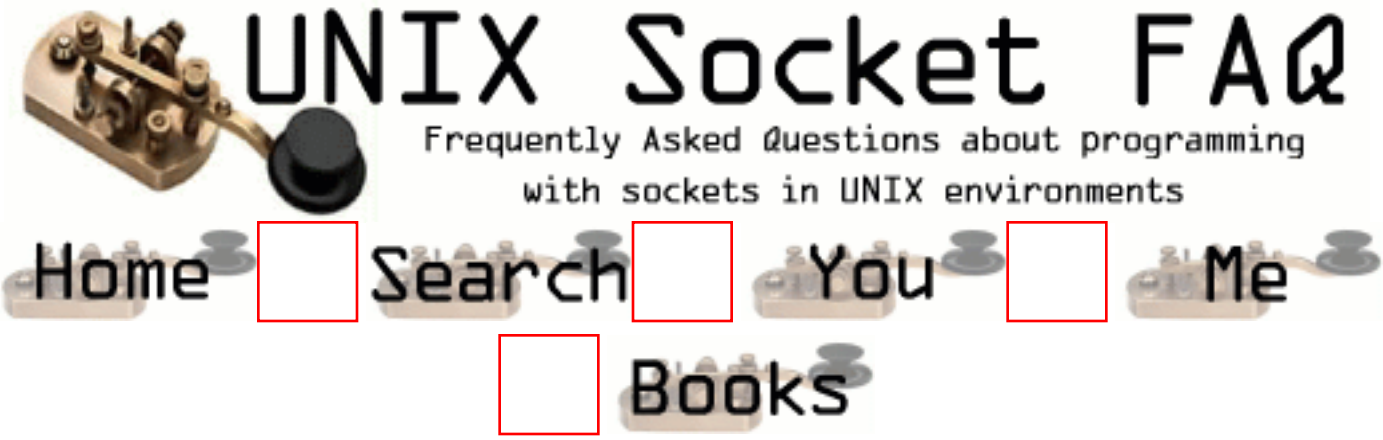
From: bimalendu

hi,
One way I can suggest is that use PF_PACKET type socket
(for linux at least) to get the raw packets directly from
the device driver.
open a socket
sockfd = socket(PF_PACKET,int socket_type,int protocol)

use htons(ETH_P_ALL) for protocol ,then you can get the data for
for all protocols (like IP, arp etc)
and use socket_type to SOCK_RAW to get raw packets including
link level header is received.
and then read in an unsigned char buffer, the arriving data.

read(sockfd,buffer,sizeof(buffer))
But you need a root to use PF_PACKET socket

Add a Comment



New Questions: How can I determine the space available in the send buffer of a socket?

From: Lyman Neuschaefer

I want to write a function which attempts to write a message to socket. Either it succeeds, writing the full content of the message to the send buffer or it fails, writing none (0 bytes) to the send buffer. Querying the socket to determine the available space in the send buffer would enable such a function. Do you have any ideas how to proceed? Thanks, Lyman Neuschaefer

---

From: Daniel Kiracofe

Well, what might work for you would be to set the SO_SNDLOWAT option to setsockopt to the size of your packet, and then set the socket to non-blocking. This way, either the whole packet gets processed, or none of it does.
I do not know of any standard way to determine the available send buffer space of a socket (although I did hack up a patch to linux 2.2 get that information).

From: Warren Nash

sizeof() or even strlen() may assist, before you write to a socket
e.g.
len = sizeof(buff);
write(1, len, sizeof(len));
write(csock, buff, sizeof(buff));

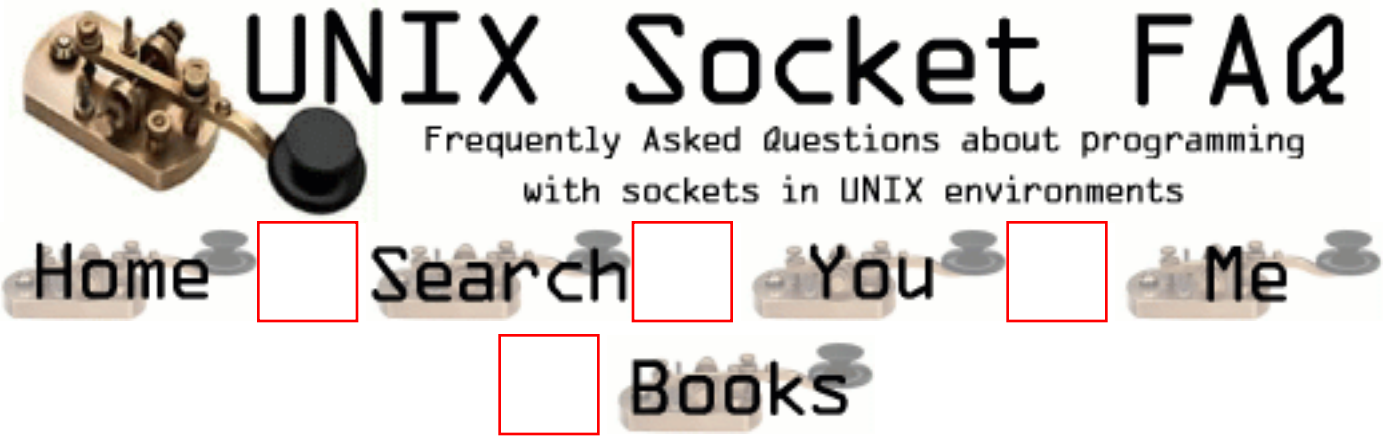
From: James Kassabian

There are two possible solutions which can, and should, be used simultaneously for your purposes, assuming a SOL_SOCKET type.
1. Set the O_NONBLOCK option on the socket using fcntl(2) and use sendmsg(3XN) to write the message to the socket.
If there is insufficient space at the sending socket to hold the message, and O_NONBLOCK is

set, sendmsg(3XN) will fail.
2. Set the size of the send buffer yourself using setsockopt(3XN) (command argument: SO_SNDBUF), then the application will know
the size of the buffer.

Warren Nash's suggestion may be based on a misunderstanding of the question. write(2) is not guaranteed to write the full
contents of the byte buffer in one operation. It only ATTEMPTS to write the number of bytes specified by the third argument.
It is guaranteed not to write more than this number of bytes. If successfull, the return value of write(2) indicates the number
of bytes actually written. It is the responsibility of the programmer to keep track of this number and adjust the byte buffer
offset, and number of bytes to write, on subsequent calls to write(2) in order to ensure that the entire contents of the buffer
are written. Since this may require several calls it does not meet your specifications, and it is not a viable solution for your
purposes.

Add a Comment



New Questions: Socket question

From: Alexey

I open TCP connection by the function:

connect(sockwww, (struct sockaddr *)&saddr, sizeof(saddr));

and after opening this connection I need to know the current meanings SEQ and ACK. How it is possible them to find out?

From: Hifny

My english is not so good.
Do you have this Information in german, please ?

From: Bret Watson

user babelfish.altavista.com and set it to translate from english to german

---

From: Vinn

try using 'tcpdump' command

From: devnull

hello Alexey,
for obtaining SEQ and ACK you should use SOCK_RAW socket,
but connect(2) is only for SOCK_STREAM and SOCK_DGRAM.
so, you should create SOCK_RAW socket, and use recvfrom(2) & sendto(2)
routines to obtain packets with TCP headers.
you will probably need root to do this.

From: neal

what alternatives exist to using a socket?

From: Jatin Patel

I want to knwo that how can i change the program of TCP to UDP. I mean what changes i have to made?

From: rgbulusu

i am a beginner and i want to know wheteher we can simulate network programminng on stand alone systems want you to send reply to rgbulusu@usa.net

---

From: alfredo

tengo un problema,necesito transferir archivos de pc a
unix(solaris), y viseversa, pero de manera automatica y
ciclica, he intentado con varios software, pero no he encontrado nada que satisfaga mis requerimientos, un amigo me siguirio utilizar sockets, pero nunca lo he utilizado.
¿puede alguien ayudarme?, !! ESTOY DESESPERADO!!.
de antemano muchas gracias, mi e-mail es : jose.landeros@iusacell.com.mx

---

From: amit Nagwanshi

i have seen set of all the questions but unfortunately i could not found answer of them, will you please help me

---

From: LIEN

Can you give more information about the soket?

Add a Comment

UNIX Socket FAQ

Frequently Asked Questions about programming with sockets in UNIX environments

New Questions: How to act as server and client at the same time.

From: Theo Wilhelm

I want to write a program, that contacts the same program
running at a different machine. The programs create server
and client sockets, that should contact each other:
server1 writes to client2
client1 reades from server2
and vers a visa.

The programs should be identically. I don't want to have a
server and a client version.

How can I menage to do this?

From: Cee

Very simple way would be to set a flag of some sort
and use it to execute functions depanding on the
it.. ex. if FLAG=server than Init_server()
FLAG=client than Init_client()

There are better ways.

From: Paul Mamin

Use on of the following ways:
- by select() or poll()
- fcntl()
- by non-bloking sockets

From: gs

make a call to client_clt_call() and server_clt_call() with if logic

---

From: Michael Pede

I did a 'man select' in Linux and derived this handy function from the program in the man pages :

```
int avail(int sock)
      {
         fd_set rfds;
         struct timeval tv;
         int retval;

         FD_ZERO(&rfds);
         FD_SET(sock, &rfds);
         /* Wait up to five seconds. */
         tv.tv_sec = 0;
         tv.tv_usec = 1;

         retval = select(sock+1, &rfds, NULL, NULL, &tv);
         /* Don't rely on the value of tv now! */
       return retval;
       }
```

if this isn't good, then return FD_ISSET(sock,&rfds); I think... :)
but since we only FD_SET one thing, we can expect select to only return based on our one socket...

Please note : avail(int) works for file descriptors, which are basically sockets returned from the socket(...) command.

From: franz

From: Alberto Mejias Martos

I think a better option (at least in Windows it works)
is to use a WaitForMultipleEvents with an event for writing
to the socket, and another for reading from it. The server
(the one who accepts the connection) should make an accept()
and then make the WaitFor..., read whenever is needed or
write if necessary.

If you need code, I might be able to send it to you.

From: Travis M. landry

Write you socket to recieve. it can also send data. The big thing to consider, is to make sure you call the right subroutines with the apropriate data.
You could make the program multi-threaded with the server in one thread,
or multiple servers in multiple threads. and the client executing in
a thread

From: frankv

It is pretty easy if you are already familiar with socket programming and concepts.

You can have different port numbers for the servers on different machines, but why bother complicating the situation.

Your program can be multi threaded or not - but it doesn't matter.
If you only want two hosts to connect to each other there will only be one connected socket anyway. You can use select() to handle multiple sockets at the same time. ie your listening and connected sockets.

Your program should pretty much be based on a server program - open a socket and listen on it. Then you add some code to try and connect to each of the other hosts at the start of the program. If the connection fails you just clean up the failed fd and wait for the other host(s) to connect to you instead.

In the rare scenario that both programs try to connect to each other at the same time you may need to add some special code. If the network is down when both application starts you may want to do periodic re-attempts to connect unestablished connections.
In the program I wrote, I created a function that checked a list of all required connections - and if it was down it would try and connect it. If all connections are up, it simply returns.
The same function can be used anytime. I use it at startup of the application, once a minute, whenever a socket closes, and triggered after a signal was handled.
This list of connections would be updated after accept()ing a connection or a successful connect().
The list is also used to determine which fd's to include in my select() statement.

If using c, your code should look something like this.

```
main()
{
  struct appdata app;
  app.port = 8888;

  app.fd = create_listen_fd( app.port );
  connectToOthers( &app );
```

```
  while( 1 ) checkSockets( &app );
}
```

Your checkSockets() routine should do a select() on your listening port and all connected sockets.
Connected sockets include both initiated and accepted sockets.
When you accept a socket you can tell where it is from by using the following code ..

```
void acceptConnection( int app_fd )
{
  struct sockaddr_in sa;
  int sal = sizeof( sa );
  int ns;

  ns = accept( app_fd, (struct sockaddr *)&sa, &sal );
  if( ns >= 0 )
  {
    char *ip_addr;

    ip_addr = inet_ntoa( sa.sin_addr );

    printf( "Connection from %s:%d", ip_addr, ntohs(sa.sin_port) );
  }
}
```

You can use memcmp on sa.sin_addr (from accept) and sin.sin_addr (used in connect) to see where this connection is from. This is more useful when there are more then one possible clients.

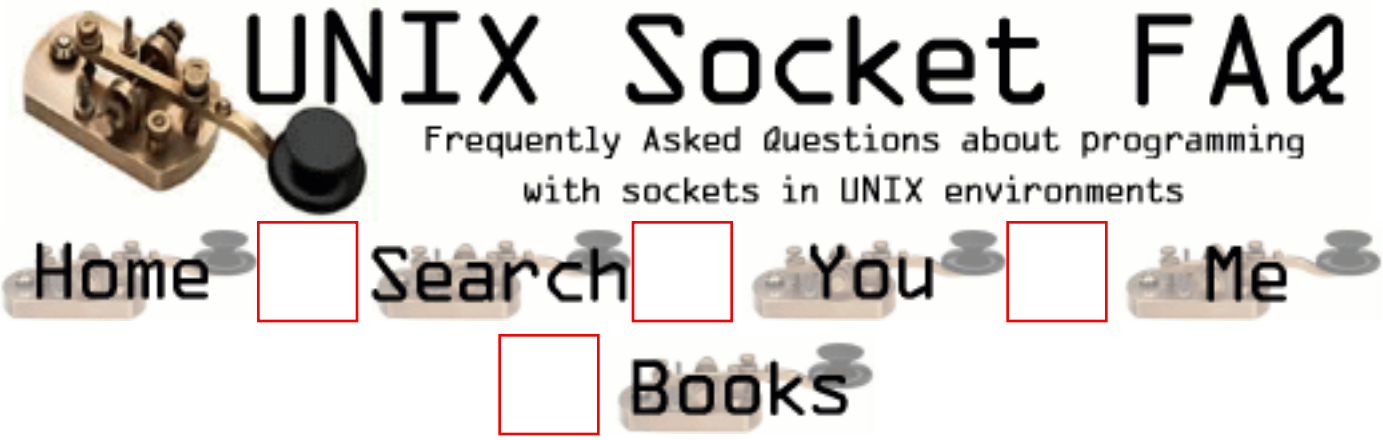
```
  struct sockaddr_in sin;
  struct hostent *hp;

  if( (hp=gethostbyname(host)) == 0 )
  {
    fprintf( stderr, "Unknown host '%s'\n", host );
    return;
  }
  memcpy( &sin.sin_addr, hp->h_addr, hp->h_length );
```

Comparing an accept()ed connections sa.sin_addr with sin.sin_addr should allow you to discriminate the connection. If the connection is unwanted you can close() the fd afterwards. If it is one you are expecting, you now know the fd for the socket and away you go.

Depending on how robust or complicated your app needs to be, you may want to just write it so that it tries to do a connect() like a client first, and if it fails become a server instead. When you finally accept a connection you can close the listening fd, and your two applications are connected either way.

FV.

Add a Comment



New Questions: What are the implications of reducing receive buffer size?

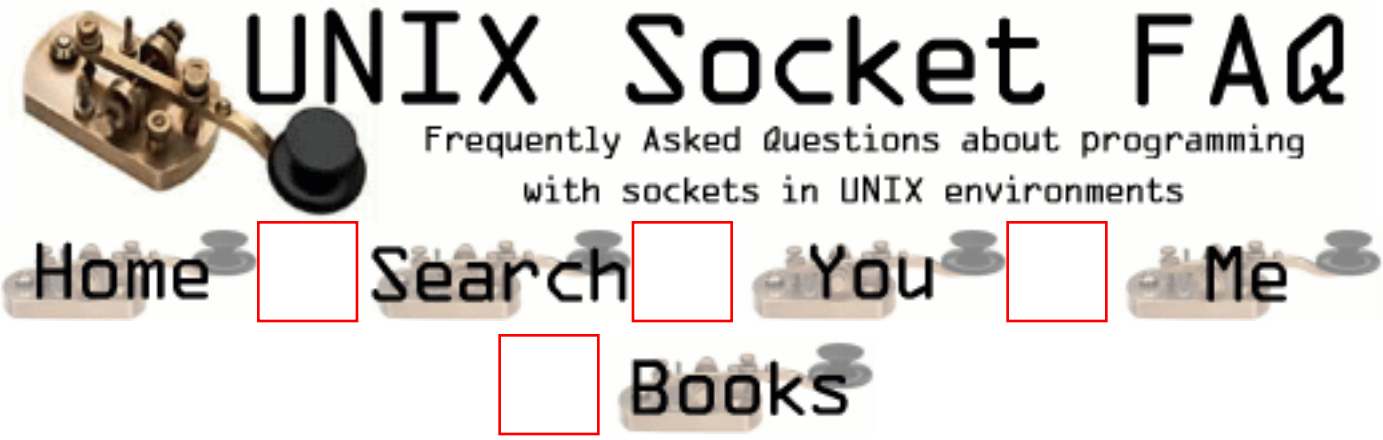
From: Sateesh Potturu

Does reducing receive buffer size cause more datagrams to be dropped?

From: gs

depending how you code the sender.

Add a Comment



New Questions: reversed name resolving

From: Efrat Meir

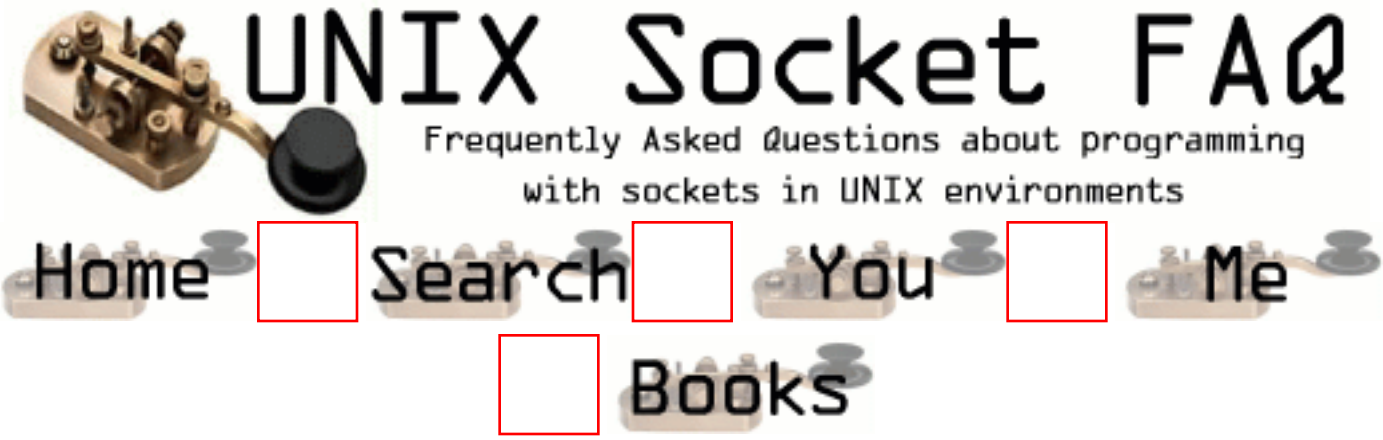
I'm using gethostbyname() to get a host address and then
gethostbyaddr() to check the hostname of the address I just
resolved.
1. Will the result always be identical ?
   (Will the 2 calls return the same 'struct hostent' ?)
2. The same question about other domain types
   (e.g.: getservbyname(), getservbyaddr())

From: Vic Metcalfe

Keep in mind that a host can have multiple A records, as well as multiple PTR records. These functions return lists of aliases, and IP addresses. With multiple A records, most DNS servers will return a different IP in the first slot each time a request is made. This is to make it easy to do cheap "load balancing".

You should always get back the same information (until the DNS records change), but not always in the same order.

Add a Comment



New Questions: How can I send packet on a choosen interface ?

From: Frederic Pont

I'd like to send UDP packet on a choosen interface. How can
I do that ?
It looks like I should use MSG_DONTROUTE but I really don't
understand how it work (if the packets are not routed, what
MAC address is choosed,as ARP won't work if the IP dest is not
on the same network).

From: Jason DeStefano

Take a look at the function sendto(). I've used this function with raw sockets and specified my prefered interface, such as eth1, in the struct sockaddr *to. Its tricky at first, but it works.

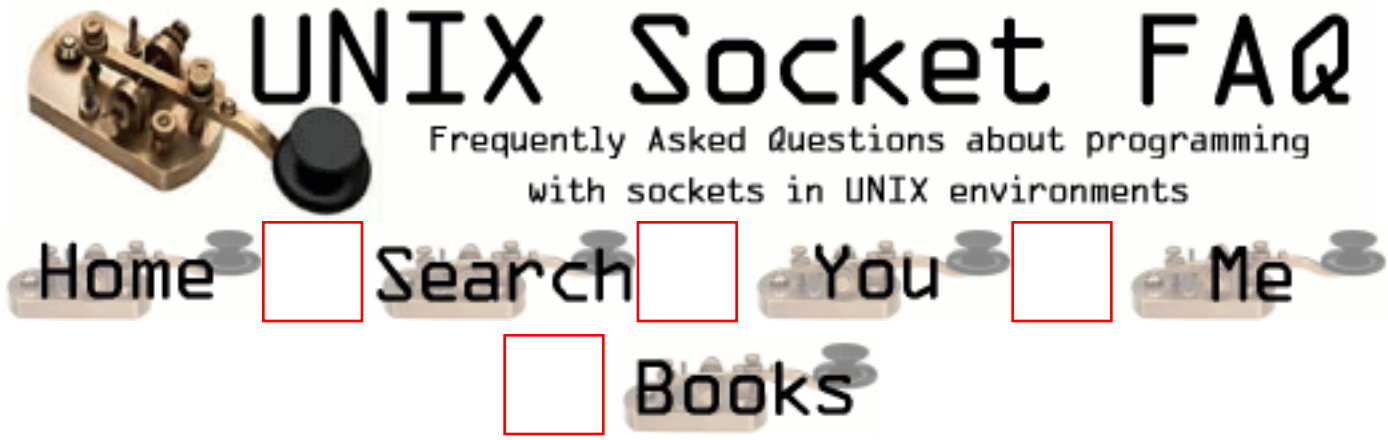
From: Nikhil Dalal

There is one more way for doing this. you can use the set socket options at the ip protocol level(IPPROTO_IP). In this u can set the IP_BROADCAST_IF to the address of the interface.

From: Arthur Lung

Off the top of my head, you can probably easily do this by
bind()'ing your socket to something other than INADDR_ANY
before you start calling send. INADDR_ANY as the local address
of the socket tells the kernel to select whatever interface
it wants that will get the job done.

Add a Comment



New Questions: Fix IP source address ?

From: Frederic Pont

I'like to send UDP packet using a specific source IP address
that can be different from the host's IP address(es). If I fix
myAddr.sin_addr.s_addr=htonl(inet_addr("123.123.123.123)),
I get an error message from bind.
Is that OS dependent ?

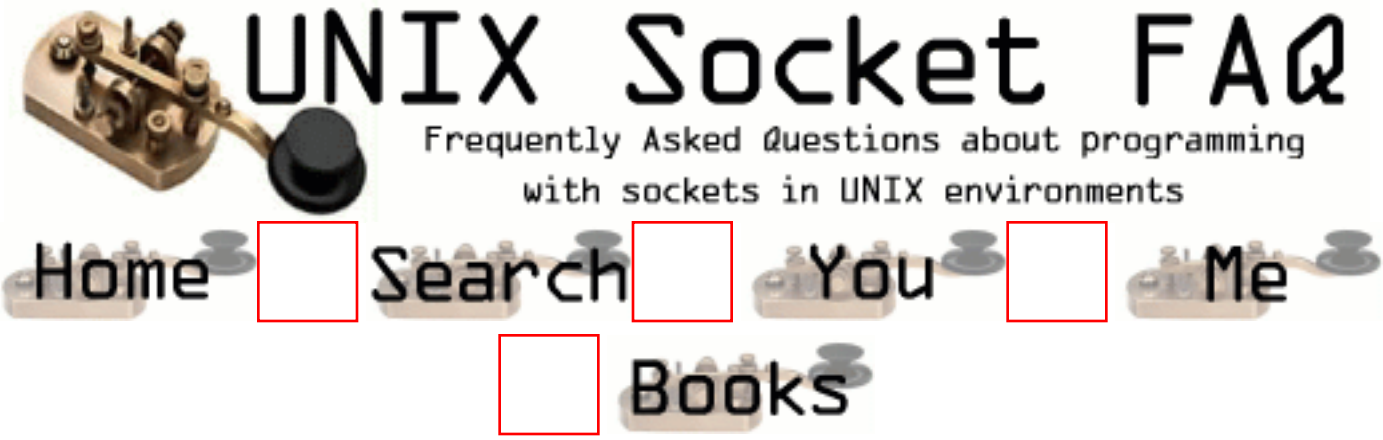
From: Sowmya N.V.

Yes It should give an error message 'cos addresses are resolved by the DNS server and we cannot use IP addresses to our fancy.It should be cofigured during installn:

From:

If you wish to set the source address of a udp packet to something that is not an interface on the machine; the best method is to obtain a library such as libnet and create a raw packet.

Add a Comment



New Questions: How do I reject a pending connection?

From: Joaquin M Lopez

With a nonblocking socket set to listen() mode,
when select() tells me there's a pending connection, seems
all I can do is accept() it; I'd like to be able to
reject it as well, but can't figure out how. Of course
I could accept the connection and immediately release it,
but that doesn't look like a pleasant solution to me.

From: Bruce M. Simpson

Under Solaris, with the XTI/TLI programming models, you
can use the t_snddis() function to reject a pending
connection request.
I can't think off the top of my head how you'd do this using
the BSD socket API, but I suppose you could try shutdown()
on the listener.

From: MaTTy`

Heyo :)

why dont you try make a fucntion that retrieves the ip adress / host of the user trying to connect... then parse the ip addresses. and only accept the ones that you specify..
thats ideally what you need.
but is there a way of seeing who was "trying " to connect to your socket?
thats the big question i guess

seeya laytah

From: Chris Leisman

AFAIK the connection has already been "accepted" by the OS
at this point (the three-way handshake is complete). So
accepting and then calling close() shouldn't be a real issue.

That being said - I wonder if there is a way (other than
packet firewalls) to tell the system to not even try and
three-way handshake with a particular host...

From: Ozz Nixon

Per previous reply - Under Unix(es) and some other OS's the listen() queue has already accepted the connection request into the queue list (pending an accept). So your choice to only accept, compare remote address if you really want to let them in - then continue your normal operation, otherwise close the socket. This also free's up the queue list (which is limited to 5 pending or less on most operating systems).

Ozz Nixon
www.dxsock.com

From: Juhan Aslak Näkkäläjärvi

Supposing I have root privileges, what would I then have to
do to accomplish such functionality - to pick addresses
and ports I want to listen to, and to decide whether I want
to accept only when the connection is coming in? My specific
use does not need to know /who/ is connecting, but *when*
(and to which address/port) is critical. Do I have to hack the
kernel tcp/ip stack or something?

What about win32? Before I realized there is no refuse() in
BSD sockets, I was going to compile with cygwin. Since changing
win32 tcp/ip stack could be just a *little* harder, do I
have even hope of being able to do this?

From: Arthur Lung

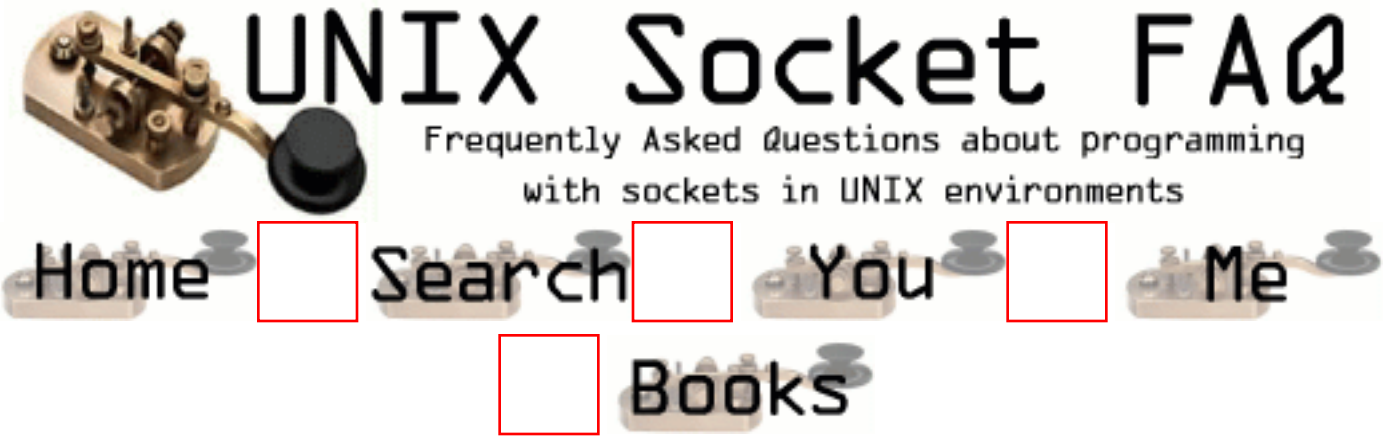
I think what you want to do is a sequence similiar to the
following (putting in actual valid parameters is left as an
exercise to the reader :)

```
accept ()
getpeername ()
myMagicCheckPeerAndDecideIfIWantThisConnection ()
close () // Only if you don't want this connection
```

You don't need to be root to do this, and it should work
under WinXX as well as all Unixen (that use BSD sockets anyways).

If you're looking for a way to just not have connections you don't want get into the listen queue, you want to buy yourself some sort of firewall...

Add a Comment



New Questions: Different Flavors of Pipe Signal

From: Naresh Motwani

Is there a way to differentiate Pipe signal received by
client as CONNECT or DISCONNECT or ERROR on the pipe?

I am writing the code in perl 5.004 on AIX.
The client sends request to the server using unix domain
socket and waits for respond from server. If the server
goes down while client is waiting for respond , how can
client detect the server when down. I tried using PIPE
signal but pipe signal is to general. I get pipe signal
when connect to server. Is there another signal that client
could be scanning for?

From: James Kassabian

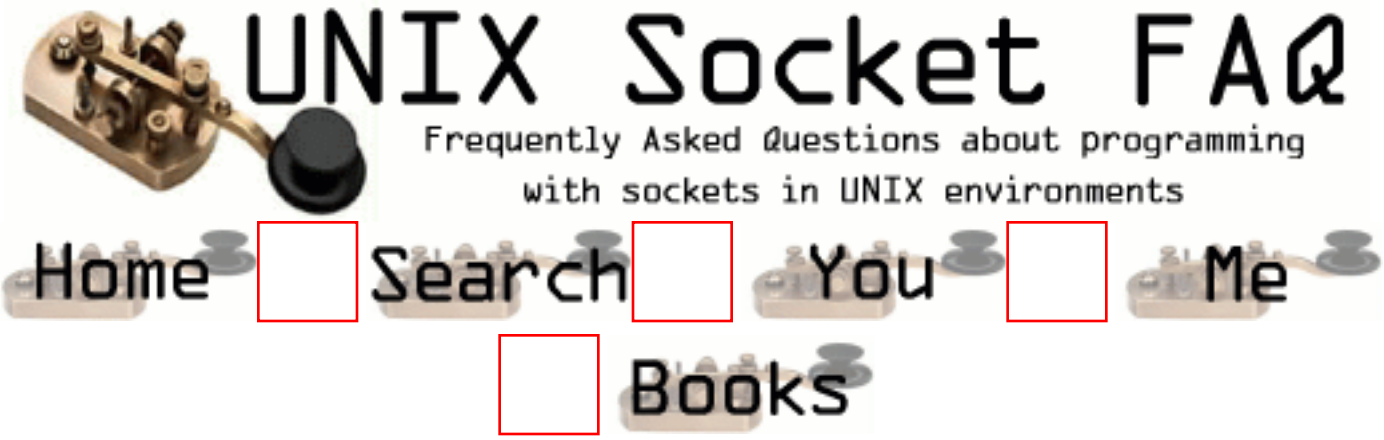
If your client program receives a SIGPIPE while trying to read from a socket
it means the client is attempting to read from a broken connection.

Any program reading from a socket should use the 4 argument select() function
to test if the file handle representing the socket has data which is ready to
be read. If so, only then should the program attempt to read data from it. You
set the timeout on select(), say 15 seconds, so that if the file handle has no
data ready to be read in 15 seconds, you can close the connection to avoid a
possible SIGPIPE. Similarly, a program writing to a socket should use the 4
argument select() function to test that the socket is ready to be written to.
It is still possible for the program to receive a SIGPIPE, and you should set
up a signal handler to catch it so that the program can handle it cleanly.

The reason for the SIGPIPE should be in the Perl built in variable $!. In

string context $! will hold a string descripton of the error. In numeric context it yields the the corresponding error number.

Add a Comment



New Questions: how to restart a server immediately

From: jerry

When I had a concurrent server and a client running, I killed the server by Cont-C then killed the client by cont-C too. It turns out that I can't start the server immediately after that. I need to wait a coupld of minutes to successfully restart the server. When I killed the client first then the server, there's no problem restarting the server immediately. Can anyone tell me why? I guess it's caused by not closing sockets correctly.

From: Vic Metcalfe

Check out 2.7. Please explain the TIME_WAIT state. This is the question that prompted me to start this faq!

From: Richard Foletta

Install a signal handler in your program to catch SIGINT. This signal handler will set a flag (rcvd_sigint = 1) when
the program receives a SIGINT from someone pressing CTL-C. Somewhere in your program you should be checking this flag and if set you should exit your program gracefully by shutting down your sockets.

From: Nick

Actually, this is because the server socket has entered the TIME_WAIT state after it was closed by the server when you sent it the SIGINT signal with Cntrl-C. you can see this by typing "netstat" after you close the server. You will see TIME_WAIT under the State column.
To fix this, simply set the socket option to re-use the address. Every concurrent server should do this between the calls to socket and bind!

Here is the line that you want:
setsockopt(sockfd, SOL_SOCKET, SO_REUSEADDR, &on, sizeof(on));
where on is an int with value 1. Hope this helps.

-Nick-

From: Mourad

You're absolutely right, Nick. I've been struggling with that problem for a long time (not being able to bind to the same port immediately after an ungraceful exit because it is in a state of TIME_WAIT according to netstat).
Anyway, you helped a great deal. Thanks.

From: Andrey

May be you dont make shutdown(your_bind_socket,2) ?
In this case in two start server failed of bind().

Sorry for my poor english.

From: Alisha

I tried setsockopt(sockfd, SOL_SOCKET, SO_REUSEADDR, &on, sizeof(on)); and I recieved a "bad file number" error. Any ideas?

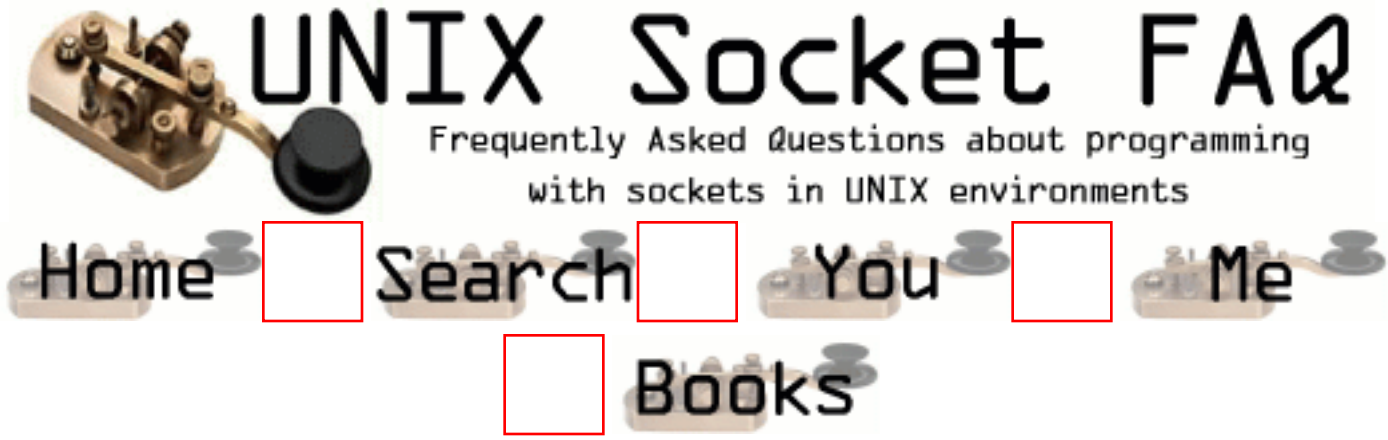
From: Aeka Abdullah

Remember that the socket option should be set BEFORE the call to bind.

From: Suresh

Hai Nick,
The information u gave is really useful and it really worked. Thanks a lot.

Add a Comment



New Questions: How to use sendmsg/recvmsg (or does anyone actually use this "feature"?)

From: Tom Emerson

The subject pretty much sums it up -- I'm looking into writing a "custom" application to send arbitrary-length records over a network, and it appears sendmsg/recvmsg will do the blocking/deblocking for me so I get discrete records rather than fragmented/recombined records on the read/recv call.

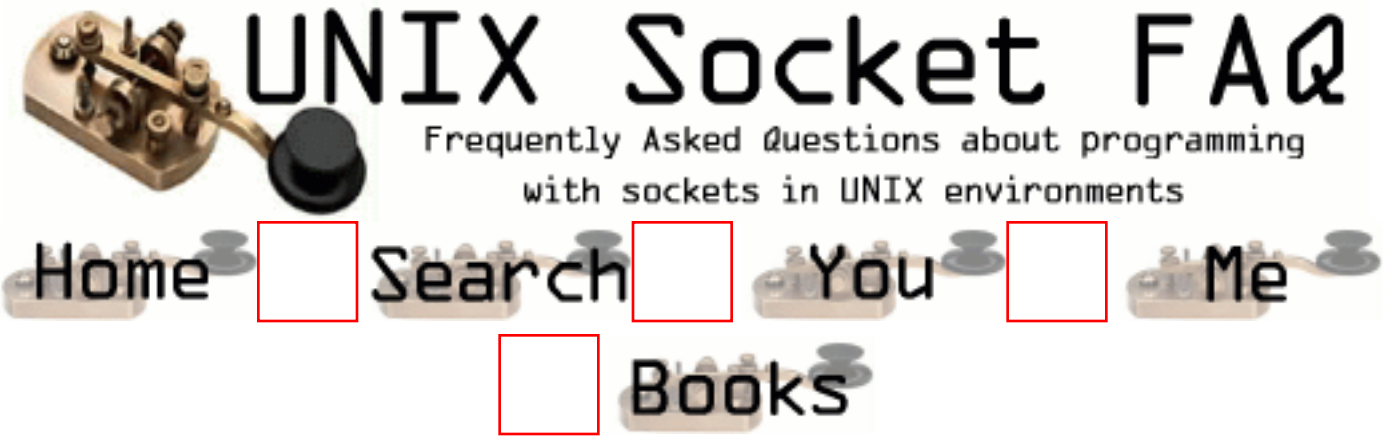
1) does anyone actually use these calls?

2) if you do use them, could you provide an example (that works)

3) is the "accessrights" string really neccessary (and if so, what do I put into this string?)

4) what is the purpose of the "alternate address" and how should/would it be used?

Add a Comment



New Questions: Motorola 88k Gotcha

From: Alan Peakall

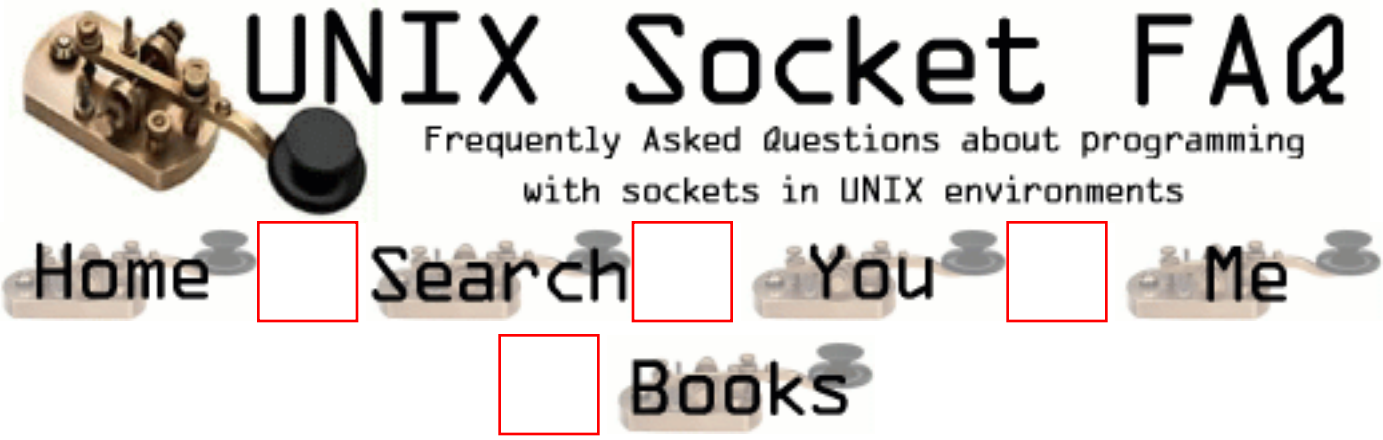
Question:
My use of a non-blocking socket for a connect with a specified timeout fails on Motorola 88k.

Answer:
Using the code outlined in the FAQ for doing a connect with a specified timeout, may fail on Motorola 88k Unix. The problem is that this system uses errno==EAGAIN as the return condition from `connect' where other systems use EINPROGRESS.

The solution is to augment your code as follows:-

```
#include <errno.h>

#if defined(m88k) || defined(__m88k__)
#undef EINPROGRESS
#endif

#ifndef EINPROGRESS
#define EINPROGRESS EAGAIN
#endif
```

Add a Comment



New Questions: Socket question

From: Steph

I have a program that acts as the server and writes alarm
messages to a socket. There is a client that exists
on another machine across the network that connects up to
the socket and reads the messages and forwards them on.
The client does a poll and if POLLIN is set in the revents
then it does a read. Everything works okay for awhile. But
after a time when the poll is done, the POLLIN wouldn't
be set in the revents anymore even though the server is
sending messages on (if you telnet to the port you can see
them). Would anybody know what is happening?

From:

From: Baroda Sankar Banerjee

Hello!
I am into the telecom field,i think i can extend some help
Set me the code with exact problem encountered.
B.S.Banerjee

From: sanjukta

The SocketServer is receiving a SIG-SIGALARM on a reading socket when the client is trying to writ the first attempt of a socket connection.

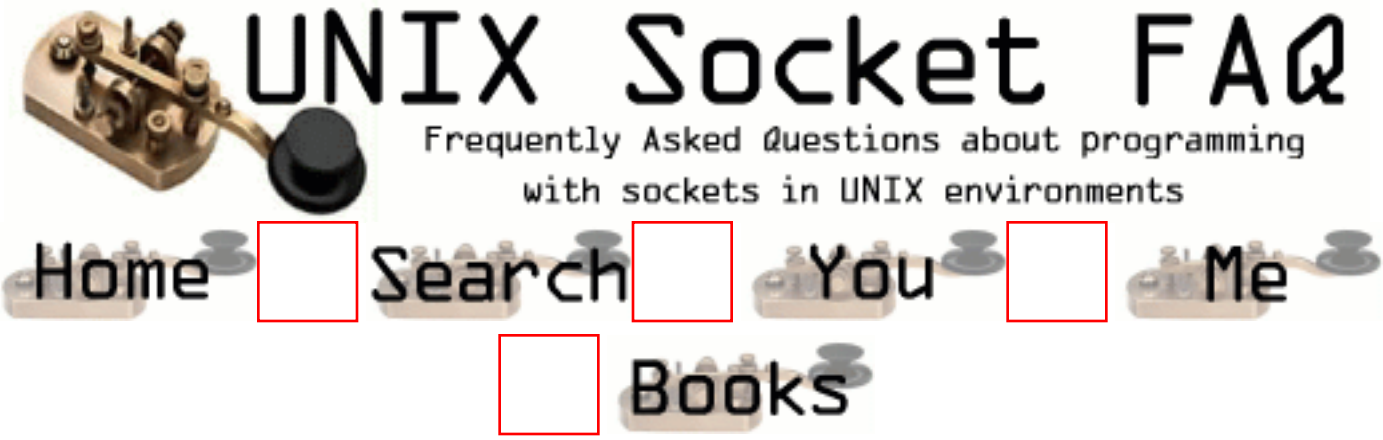
That is where it is killing 30sec.

Any idea why it is receiving a SIG-SIGALARM ?

Thanks in advance

Sanjukta

Add a Comment



New Questions: Errors in sockets

From: Alejandro Botello

Why some code that work fine in many UNIX operating systems (like Solaris, AIX or Irix) not work properly under LINUX (RedHat 4.2)?. I run a simple server using sockets and get the following error mesages in recv(): "Broken pipe" and "Socket operation on non-socket". What code I have to add to this work well?..Someone could helpme?

From:

From: xidong wang

please post your code.

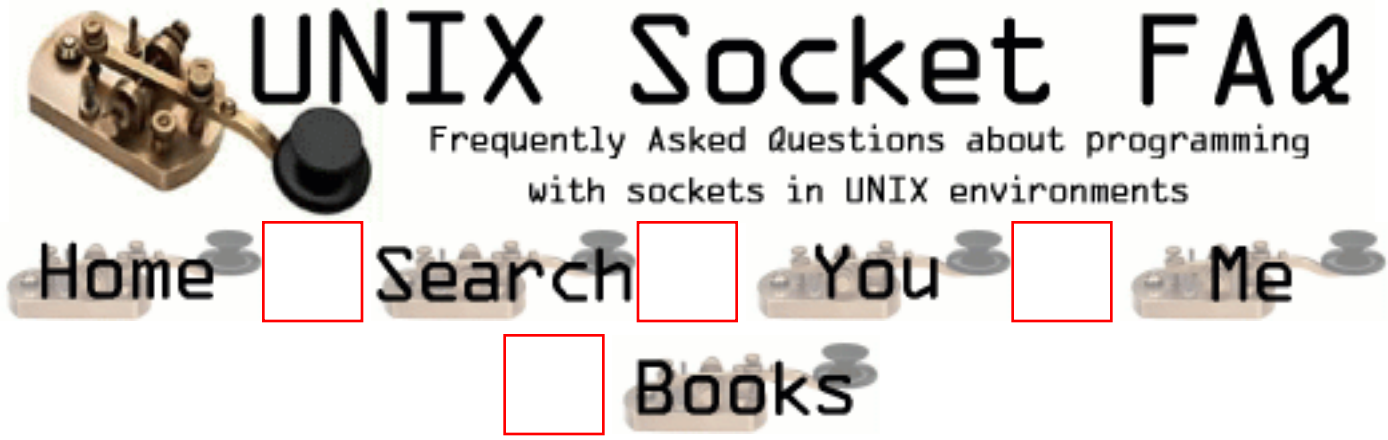
From: Michael H. Ward

Make sure you have your IPC (inter process communication)
option turned on in your kernel configuration.

From: roshan

Q1.Why is that a server process running in the linux gives socket error?
Q2.how is the socket is linked in linux as we compile as
<cc server.c -lsocket serv > in unix

Add a Comment



New Questions: problem in socket programming

From: azizan mohamad

I wrote a simple program to tansmit message from client to server. I use send and recv function. The problem is the server did not receive the message because it return -1.Why this is happened??

my sample code for client:

...
...

```
n = strlen(s1);
cout << "sending message"<<s1<<"\..";
cout <<((send(sock,s1,n,0)==-1)?"Error:"Done")<<endl;
```

Note: s1 is declare is char *

my sample code for server:

```
while(1)
 if (recv(nsock,s3,sizeof(s3),0)==-1)
   {
    cout << "Error";
    break;
   }
  else
   {
```

```
 cout << " message" << s3;
}
```

From: Tarkan Alptekin DURMUS

You must use for <char *> to <void *>.
And use the max length of the incoming value(sampl 1024)
if you know.

From: satish

Am new to socket programming ,what could be the possible reasons for getting a bad socket descriptor while issuing a connect call and how am get over it ,

Can anyone shed some light on it ,

From: Hector Lasso

Hi,

Your problem is simple: you are using sizeof(s3) which returns the size of the pointer to a char (if s3 is a char array). You must use the number of characters your array can hold or the maximun number of chars you are expecting (whichever is less)

To satish:
Are you using socket() before calling connect()???
If you post your code it would be easier...

From: vinay pole

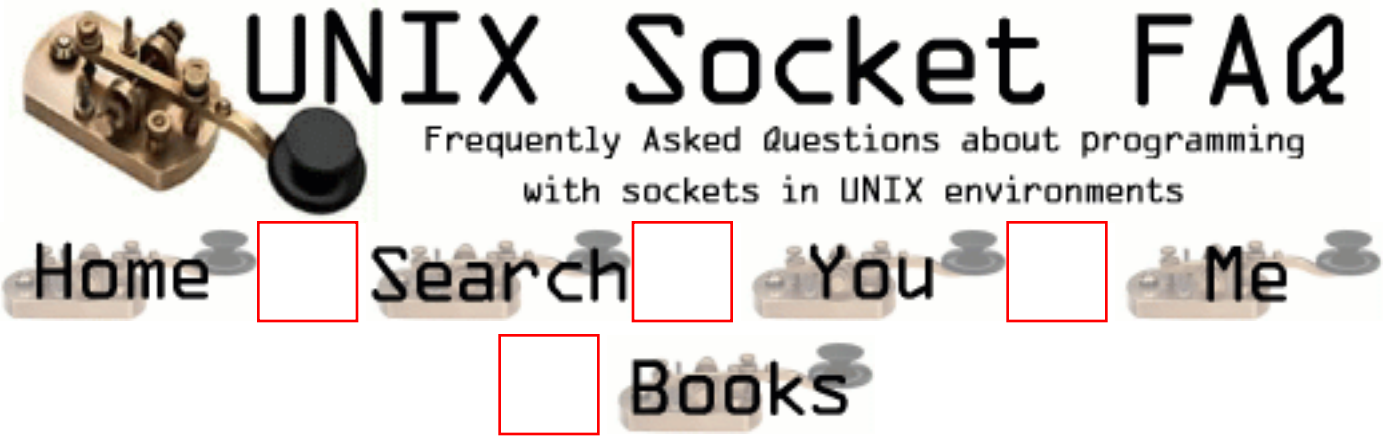
From: Raja

My server is not able to receive the message sent by the client which established connection to server thru sockets.

From: abed

I wrote one client_server program but I did a few process between each of sending and recieving message, my program only worked for first send and recieve.Remainder of send and receive messages weren`t done.Please guide me.

Add a Comment



New Questions: client/server environment

From: ezzete

Suppose i want to build a simple email system which involve
one sender and one receiver. Of course i need to create a socket
to establish the connection. What is the step should i do? Do i
need to establish the client/server connection first or i wait for
the input message from user, break them into a small units.Only after
this i establish the connection then transfer the message ?

From: Warren Nash

I think its best you buy Richard Stevens book - as it describes everything you need
e.g.all the functions.

From: mn

will you please advice me on which chapter of richard stevens
i should refer to , to get the correct & direct answer
to the question that i asked?

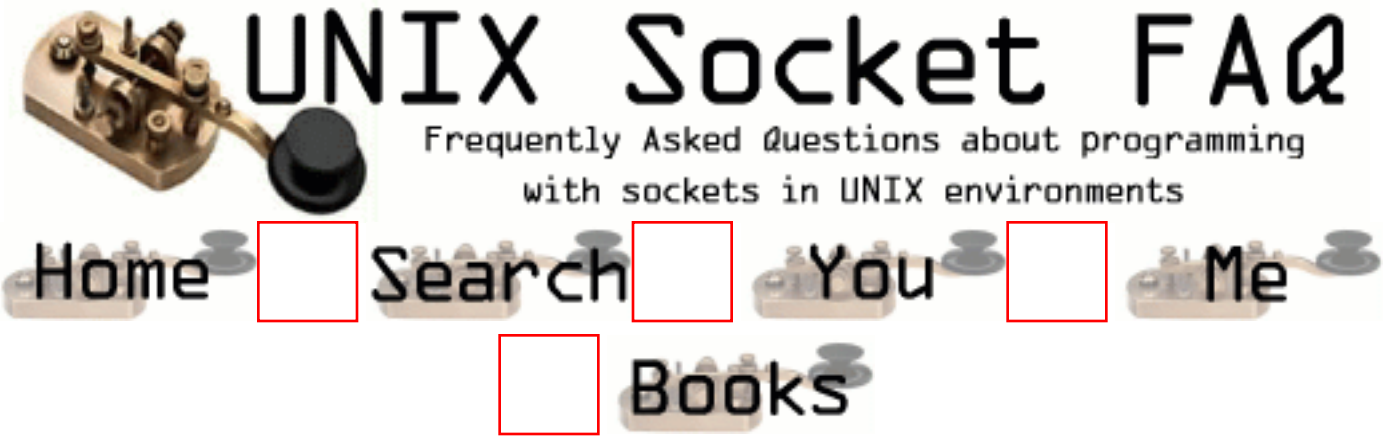
From: lm0re

u also should read the rfc 821,then u'll know what u should do.

From: camran

heey mn;
your problem is laziness ; read that book twice then u will understand good, dont ask for chapters, I M sure u will find your answer.

Add a Comment



New Questions: Sending large (> 50 meg) files (binary + ascii) through sockets

From: Randy Randleson

I want to send large files that would exceed the send and
receive buffer size limitations. What would be the best
method of doing this? Any code out there that gets at this
would be much appreciated.

From: Randy Randleson

Never mind -- I figured this out on my own

From: Landabaso

The easiest way to do this would be to invent yourself a easy protocol,
such sending first the amount of data you are going to send and
then the size of little packets.
Then you only would have to divide your data in these packets
and reconstructing them in the other side of the comunication

---

From: Mikael Chambon

Don't care
do
{
 brec += send (.....)
}
while(brec < 50 megs) ;

You can do it like that, it's works !!
Hope that helps !!

From: Jagadeesh

Hello,
Could any one tel me how to re-assemble the packets once received.

From: THoK

Just write a small protocol thingy. Use four bytes for
the packet number, split your data up into 10K packets, and
send each packet one after another. Each packet starts with
a four byte number. First packet has a value of "0", next
10K packet has "1", and so forth. That way it's easy to
reassemble.

From: Paul Beardsell

Why bother with all this "devise your own protocol"
rubbish? SOCK_STREAM guarantees the packets will
arrive in order. By the way: Even if you are only
sending 100 bytes (not 50MB) the packet can still be broken
up and arrive in more than one lump.

From: mrgibson

the best way i know to do it is to open a file in binary mode and put into it each packets you receive (fixed size or not, who care) and close the file after the transfert.

Do not think about a fat variable containing 50 Megs.
Imagine how you computer must swap? :)

From: Ozz Nixon

Correct Answer:

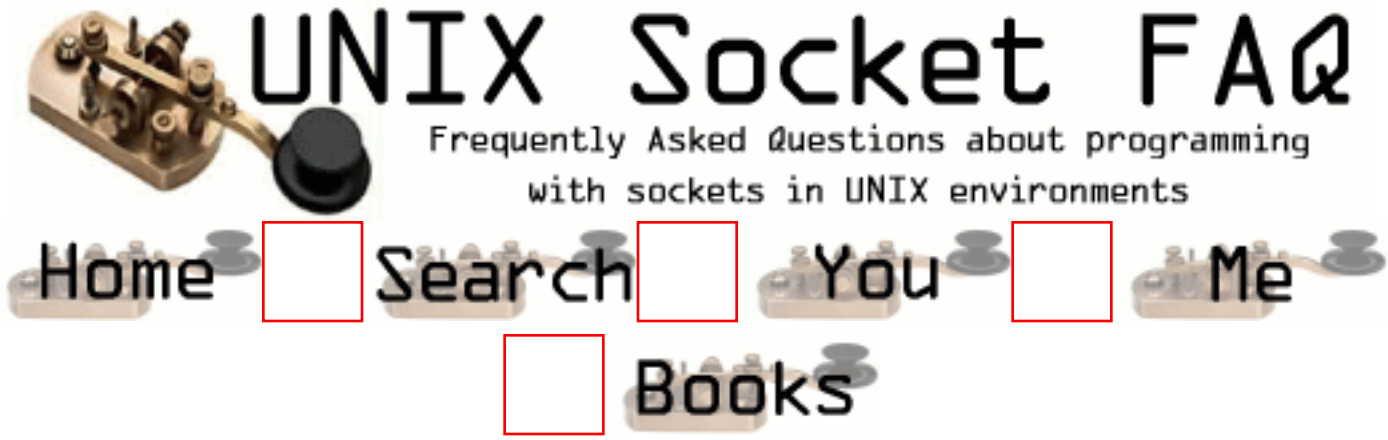
Yes a protocol solution is viable, but you still have the problem of overrunning the outbound socket buffer, or the receiver's socket buffer (especially when they are on a slower connection).

Making the assumption your packet will get there is down right stupid! I hear this from customers all the time, it is not going to get there, if the remote buffer is full - duh! It gets dropped.

The correct answer is to develop calculations of delivery performance. There are ICMP messages the remote will send to say "SLOW DOWN - YOU ARE TOO DAMN FAST!". This is done at transparent to the receiver's code - which is simply doing a read loop. So you must implement support for handling these. I have not seen anyone release a good example of doing this correctly, but my staff is building an example for a project we are developing. Which requires the ability to throttle out socket suite - doing bulk delivery.

Ozz Nixon
www.dxsock.com

Add a Comment



New Questions: Proxy Handling

From: Sankalp Upadhyay

Where can I get the method to see webpages through a proxy??
that is, I should know the request syntax to be sent to proxy server.

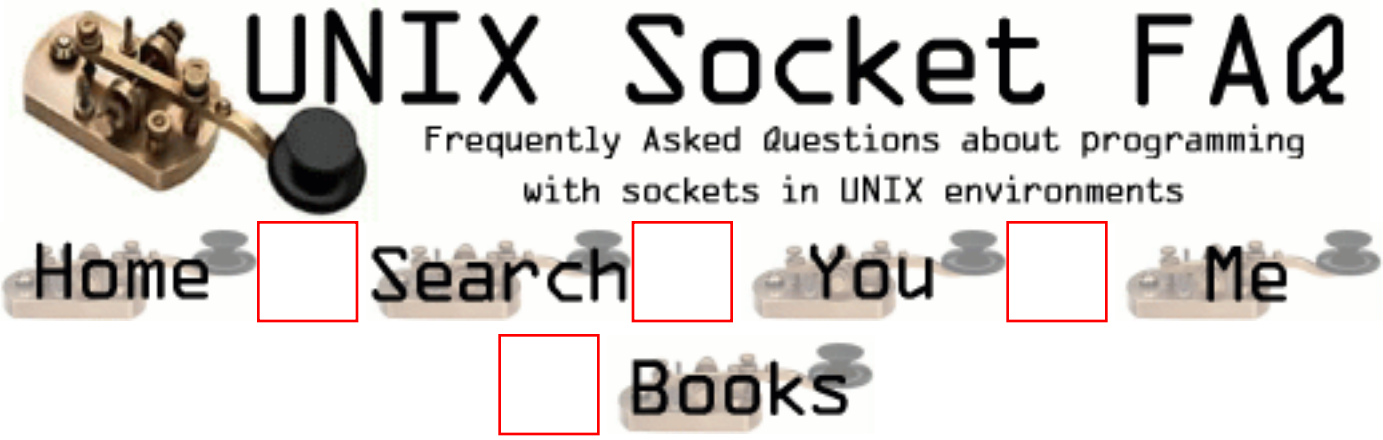
From: Alex Povolotsky

The simpliest variant is to send full URL, like

GET http://www.lcg.org/sock-faq/ HTTP/1.0

to proxy site:port. Maybe you'd like to use LWP::UserAgent PERL module, it already have proxy interface.

Add a Comment



New Questions: socket and XTI

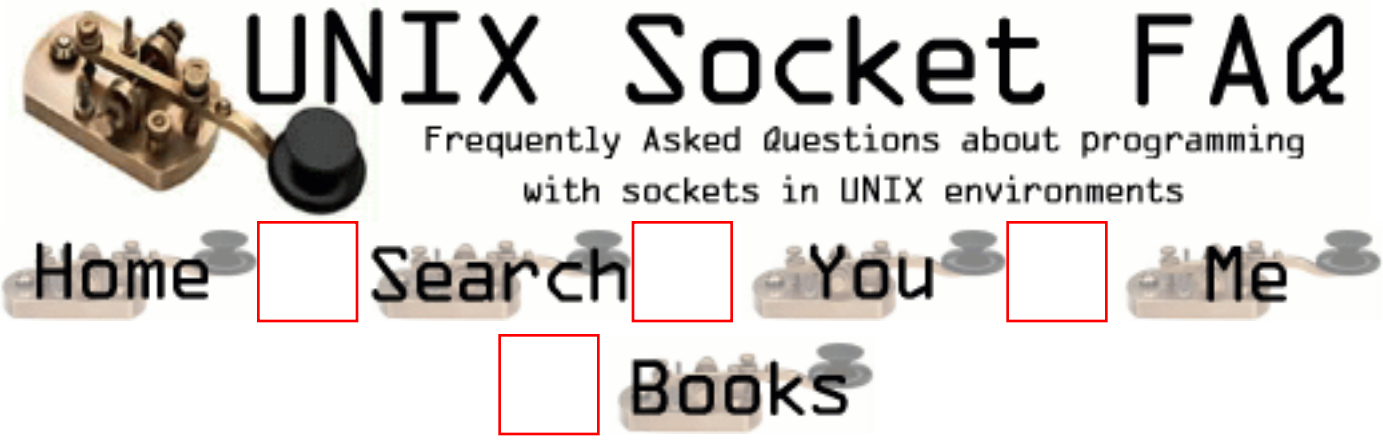
From: guyot patrick

Is It possible to communicate between an application with API XTI and
an application with API socket ?

From: Tommy Irizarry

Yes, it is possible.
It is the application protocol (HTTP, Telnet, amd so on)
and the transport layer (TCP or UDP) that determine interoperability. The API we use to write the code makes
no difference.

Add a Comment



New Questions: Sending & Receiving data on a socket

From: Eileen

I'm sending 10 x 2048byte messages from a server to a client
over a socket connection. Although the call to write returns
successfully at the server end, when I do a select for read on
the client end the socket does not flag as ready to read. I
don't get to check if FD_ISET because select never returns > 0.
This problem is intermittent, the messages are sent and receceived
successfully about 60% of the time. Does anyone have any idea why
the call to select doesn't pick up the socket as ready to read when
the server has written the messages to the socket successfully. I have
have got a 10sec timeout on the select so I'm waiting plenty of time.
I need to sort this problem urgently so please help if you can.
Thanks.

From: Vic Metcalfe

If you're running UDP packets could be getting dropped. In any case, I'd suggest running tcpdump or whatever sniffer you like and checking to see that things look right on the network.

From: Eileen

Thanks Vic. I'll give that a go.

From: Bruce Edgerton

Eileen, I am having a similar problem when using the Tools++ Professional library. I am waiting on select on the client side, and only some of the messages get through. I mproved it a little by flushing the buffers, but it still didn't work 100%. Interestingly, when I killed the server side, suddenly all of the messages appeared on the client side before it shut down with a read error. Have you found a reason, and better still, a solution?

From: Mahafuz

I'm not 100% sure by my suggestion is try setting
TCP_NODELAY socket option. This will make the system to send out any packet immidiately without waiting for ACK for previous packet.

I assume that you are using TCP (STREAM) socket.

From: Arun Kumar

Hai Eileen, While using select method to poll for the
message , you have to set the readset every time.
That is , FD_SET(Handles, &readset).

From: franky

If you are using UDP it is theoretically possible. Less likely if the server and client are on the same machine.

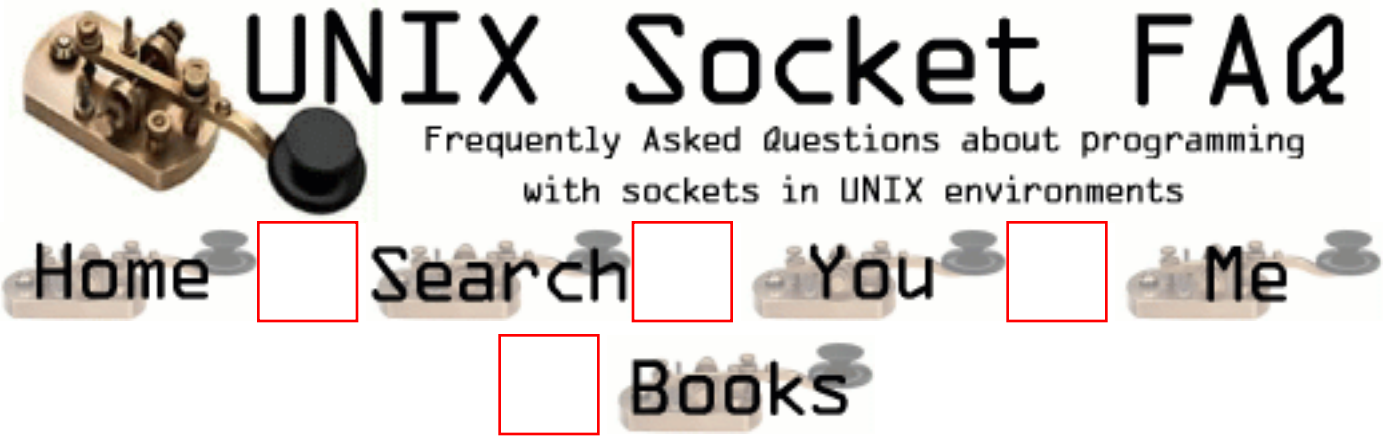
Non-unix systems used to be really crap at handling UDP and dropped packets left right and centre unless you wrote a handler to read packets asap and buffer them manually yourself. I don't know (or care) if it has been improved since then.

If you are worried about losing info you should really use tcp/ip SOCK_STREAM.

If you are using SOCK_STREAM (which I assume anyway) there is either something seriously wrong here, or it is just a silly simple oversight.
The previous suggestion about using FD_SET everytime is important. If select gets an error your set remains unmodified and you may believe there is info available.
The same goes vise-versa - if your fd gets cleared from the set somehow it needs to be added again. Remember that the call to select will modifiy your sets. Is it possible you are using the same readset reference somewhere else?
From your message it seems that select is returning 0. Is it returning before the ten seconds timelimit?? Is it possible that signals are interupting your select?
If select is returning -1, check errno for more info.

fv.

Add a Comment



New Questions: Stream Sockets, Multithreading, and Windows

From: Daniel Morgan

What is the best way to implement a graphical client using
a GUI toolkit such as GTK+, multi-threading using POSIX threads,
and internet stream sockets?

I currently I have basic basic understanding and can do the above mentioned.
It is just rather hard putting all three together to form something useful.

Let's say I wanted to implement a GUI POP3 client, GUI IRC client, or a GUI FTP client.

How would I do this?

From: dodi

i have problem with my irc client
the warning is identd no istalled in your client
but in client i have istall it

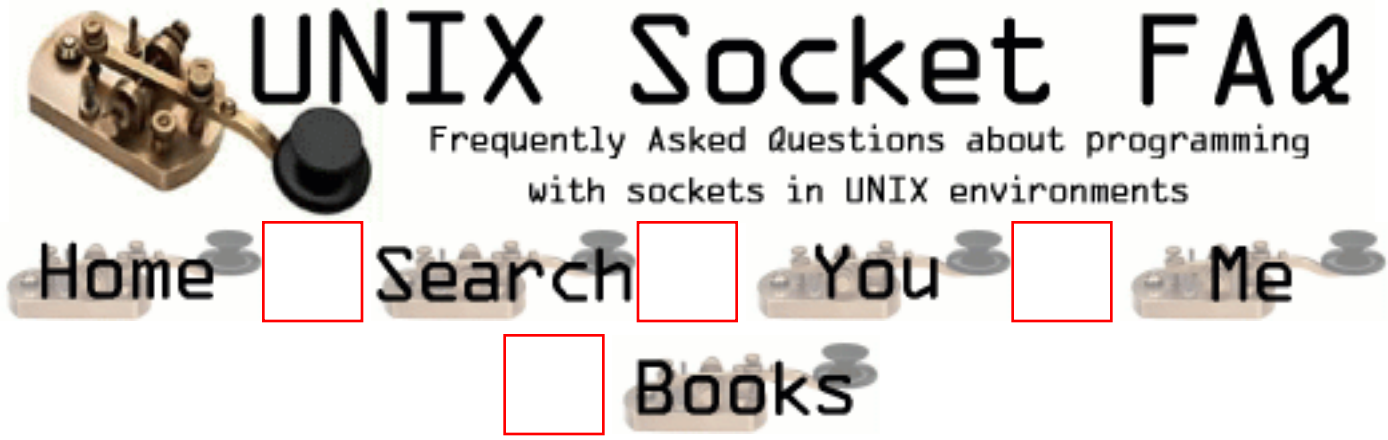
From: Rang

I wnt to know
if i want to create program for connct server with c++ in UNIX and Windows System
if you have an example coding program with c then can you sent to me for learning
Thanks you

From: lost

From:

Add a Comment



New Questions: Subsequent connect call fails (EBADF); why?

From: arp

On Linux (Redhat 5.0 not that it matters):
0. int s = socket (PF_INET, ...);
1. connect (s, ...);
2. write (s...);
3. read(s, ...);
4. close (s);
5. connect (s, ...);

(where i've left out the error checking and extraneous stuff).
The second connect (step 5) fails with errno == EBADF. This
happens even after adding a 'shutdown' before step 4.

Is this correct? The man pages & GNU libc.info don't indicate
that close does anything strange to the socket descriptor.
Why do i need to create another one with socket (...)? Or
is this a Linux bug (sorry, i don't have access to other
UNICES to test against)?

If this is addressed in the FAQ, slap me with a wet noodle and point
me in the correct direction please.

From: Vic Metcalfe

Close frees all the kernel resources associated with the file descriptor, so yes, you do need to call socket() again.

And no, that wasn't in the FAQ, so no wet noodle for you!

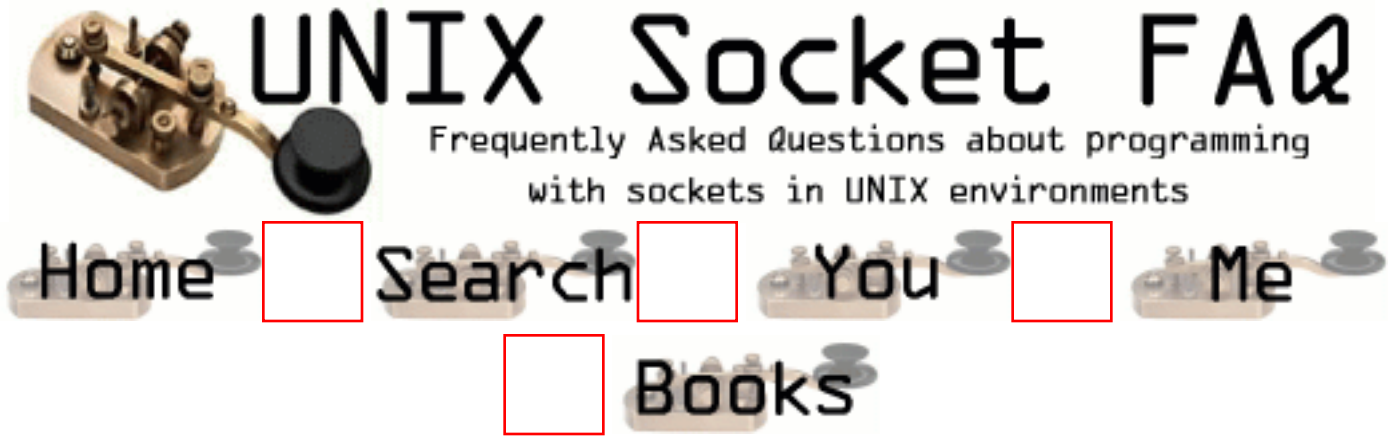
Take care,

Vic.

From: Andy Kanch

Thanks for the info. I was alos facing the same problem in CICS. Now I know what to do.

Add a Comment



New Questions: HTML Client

From: Craig Matsuura

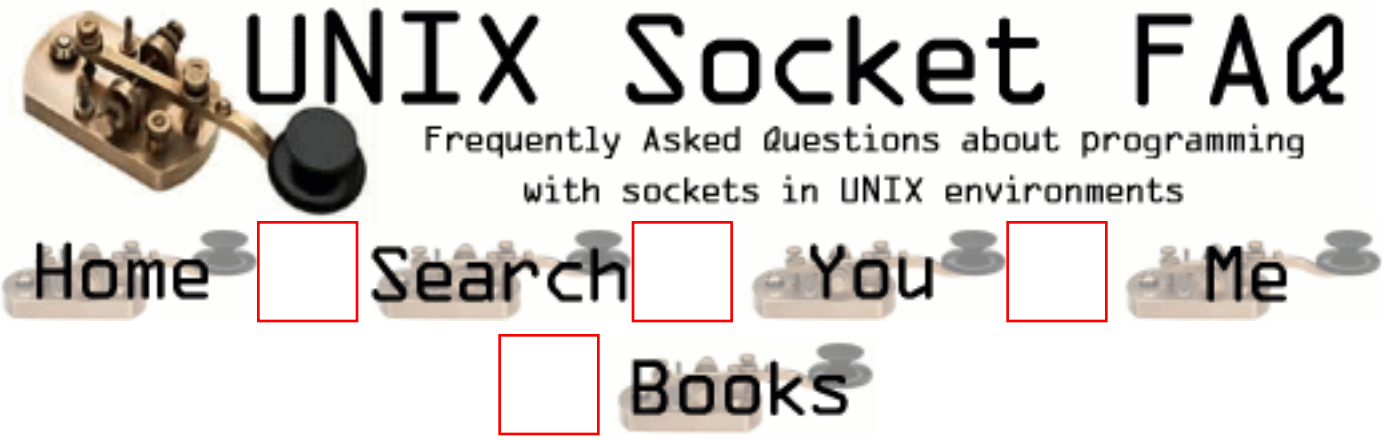
Where can I find example code for writing a simple HTML Client
(Web Browser). In C.

From: Carlos M. Gutierrez

Try looking up the source code to Lynx, a text-only web browser. Also check libwww at www.w3c.org

---

Add a Comment



New Questions: Passing sockets to pre-forked children

From: Carlos M. Gutierrez

If I have server (like Apache) and that server "pre-forks" several children, how are incoming socket connections passed to the pre-forked children? Because they are "pre-forked", they can't inherit directly from the parent's file descriptors...

From: Josef Pojsl

You can do socket(), bind() and listen() in the original process and then fork() as many children as you like and do select() on the socket in each forked child. After select(), use accept(), serve the call and return to the loop beginning with select(). This works for me on FreeBSD, at least.

From: Jim Rogers

One method is to have all incoming connection requests handled by the parent. The parent generates an event, passes the incoming socket info to the child, and closes the socket. The child catches the event and establishes its own socket connection with the client.

From: Chris Wood

Unless you are on a MVS system this is not possible.

From: Rob Seace

What's not possible? Passing FDs to child processes? Sure
it is, on most OS's I've seen... It's usually just not
documented very well... ;-) But, for instance, you can do
it under Linux with an AF_UNIX socket to the process you
want to pass FDs (including socket FDs) to; you call
sendmsg() with a control message of type SCM_RIGHTS, and
filled with an array of FDs to transfer to the other
process...

From: Wes

You are all misunderstanding the question.

The question Carlos asked was how to pass a socket to a pre-forked child.

Pre-forking the child means that the child does not know what do with the integer we normally call a "file descriptor", because this integer really doesn't describe anything; it is an index into an array we don't normally see as programmers. So, we need to pass the underlying information to the child somehow.

The general technique for solving this problem is to set up a method of IPC between child and parent before the inbound connection arrives. I like to use AF_UNIX domain sockets for this. Then, when the parent needs to send a socket to the child, it constructs a msghdr with an iovec containing the underlying information. The receiving process will almost certainly have a difference file descriptor number than the sending process does, for the same open file (socket).

There is an example in W. Richard Stevens' TCP/IP Illustrated Volume I of how to do this. (If not this book, maybe the first Unix IPC book? I know it's there somewhere..)

Also, Apache does this, as Carlos pointed out. Why not read the source?

From: Wes

..I found that reference:

Chapter 27.9, "TCP Preforked Server, Descriptor Passing", on page 746 of UNIX Network Programming, Volume 1, Second Edition. W. Richard Stevens, ISBM 0-13-490012-X.

Add a Comment

UNIX Socket FAQ

Frequently Asked Questions about programming with sockets in UNIX environments

New Questions: Unix domain sockets using linux

From: All

I'm using Redhat 6.0. I'm trying to implement a client and aserver model using unix domain sockets.
I'm using GNOM windows manager. I want the client to communicate with the server that is located in a defferent user acount
so I'm using two Xtermials to run each of the client and the server. since they are in different directory I gave the client the right path to bind. But When I run the client the system gives me an error message
saying can not connect. if any one knows the answere please let me know.

From: Jacqueline Guerrero

Are you sure that your server is actually waiting to receive messages on the socket that your client is sending to?

From: S.M.Krish

Hai Friend
If You Receive "Connection Refused" Error message
Probably The Server May not be in Ready Condition while
you run your client. And Also Check the Directory
Path Permissions between Your client and Servers.
And the "Socket File's" Access permission. If You Receive
Error Still, Post your error message clearly. Thanks and bye

From: soupa

Also including the code on the post won't hurt.
It sounds as if your server is not running, and if it is running it is not on the specified port

---

From: soupa

Also including the code on the post won't hurt.
It sounds as if your server is not running, and if it is running it is not on the specified port

---

From: anil kumar

is it possible to run a client /server model on linux without network card?
if yes how?

From: Michael Jarvis

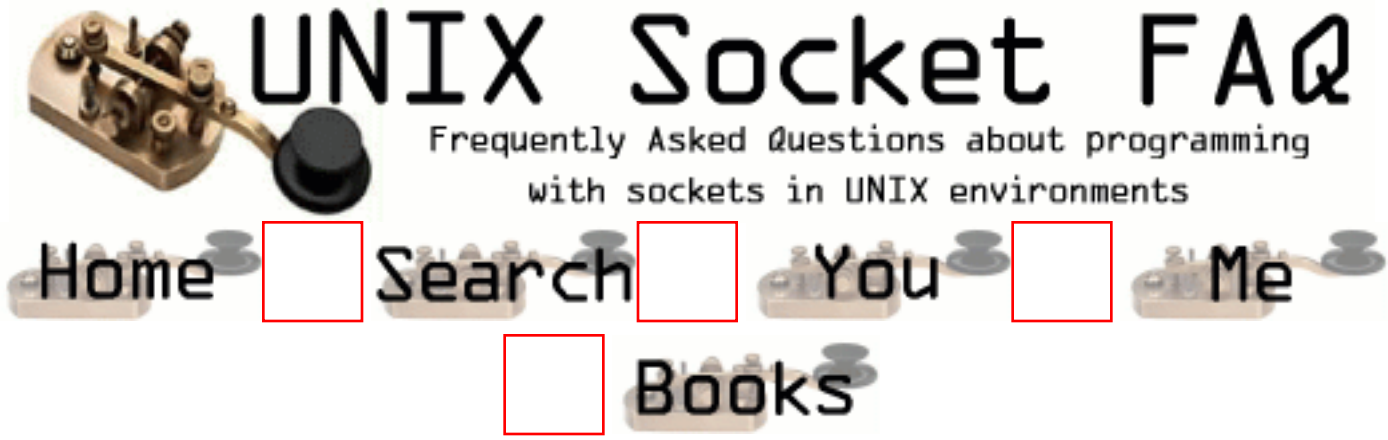
You can do client/server programming over the local loopback interface (lo -- 127.0.0.1). It will allow one program to talk to another program on the same machine over a socket, without actually needing a real network card.

From: Sougat

Hi,
I am receiving an error when I try to start Websphere Commerce Suite on Aix4.3.3.
CMN0310E: The socket name is already in use.
CMN0519E: Server controller TCP/IP service 'newcom' is already in use, an instance of server controller may already be running.

The problem gets resolved when we reboot the machine. Could someone tell as to how to resolve this problem without reboot.
Thanks,
Sougat

Add a Comment



New Questions: Can port be shared ?

From: Kim Dong Ho

When two hosts, client C and server S, make connection each other.
In this situation, a new host H want to communicate with client C through the port - ex, port 1000 - which is assigned
to communicate with server S.
Of course, there is no connection with client C and new host H.
Can the new host H send packet to the client's opened port, port 1000.
And, Can the client receive the packet from new host H through the port, port 1000, which is opened to
server S.

Thanks for reading

From: Vic Metcalfe

Lets look at a specific real life example. Lets say that joe@hostA.net and jane@hostB.net send email to one another at the same time. Here's what happens...

- Joe's email client sends his message to his smtp server which happens to be mail.hostA.net.
- Jane's email client sends her message to her smtp server which happens to be mail.hostB.net.
- The SMTP server at mail.hostA.net does an MX lookup on hostB.net which returns mail.hostB.net
- The SMTP server at mail.hostB.net does an MX lookup on hostA.net which returns mail.hostA.net
- The SMTP server at mail.hostA.net connects to port 25 of mail.hostB.net
- The SMTP server at mail.hostB.net connects to port 25 of mail.hostA.net
- The SMTP server at mail.hostA.net sends Joe's email to Jane at mail.hostB.net
- The SMTP server at mail.hostB.net sends Jane's email to Joe at mail.hostA.net

You'll notice that mail.hostA.net connects to mail.hostB.net on port 25 at *the same time* that mail.hostB.net connects to mail.hostA.net on the same port! The reason is that connections are identified by five things: The source IP, the destination IP, the source port, the target port and the IP protocol. The connections look like this:

Source IP	Dest IP	Source Port	Dest Port	IP Protocol
mail.hostA.net	mail.hostB.net	25	55851	TCP
mail.hostB.net	mail.hostA.net	25	55840	TCP

Each connection is unique, and there is no conflict. Otherwise mail servers wouldn't be able to communicate with one another at the same time, you couldn't run a web browser from a web server, etc.

I hope this helps,
Vic.

From: Jay

Hi,
It cant be possible because when u sent mesg u will connect and connect will fail bcoz connect intern call bind and u cant bind single port to two sockets.

From: caspre

Jay: You're right that you can't bind two sockets to the same port, but when you accept a connection from a listening socket on a port using accept(), a new socket is created on the same port. This means that you now have the listening socket, and a new socket, both on the same port

---

From: Arthur Lung

Two servers running on the same computer can't listen on the same port for connections. Like if you had two web servers, they couldn't both use port 80. When you try to start the second server, it notices the port is already in use and you get a big fat error.

On the other hand, two clients running on the same computer can connect to the same server and the same port simultaneously. The reason for that is because the "local port" that represents the client side endpoint of the communication is different for both clients.

So to summarize:

2 telnet servers on server Bob listening on port 23 = no way

2 telnet connections to server bob connected to port 23 = happy land

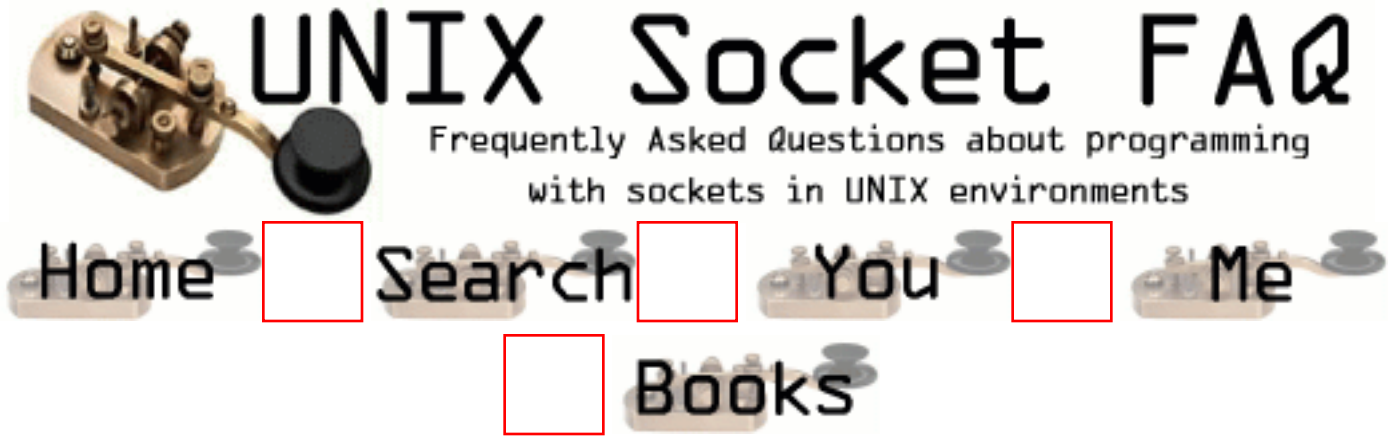
---

From: Tim Jones

Actually, Arthur, that's not exactly it either. If your computer has MORE than one IP address, you can configure your server processes to listen to a particular IP and PORT combination. I have a colocated Linux box with IPs x.y.z.7 and x.y.z.8. Apache listens to x.y.z.7:80 and Tomcat (another webserver) listens to x.y.z.8:80.

You can also configure stuff to only listen to 127.0.0.1, which is a common way to secure stuff from the outside world without turning it off completely.

Add a Comment



New Questions: Reading from a port

From: Niraj Shah

Hi,
I have been successfull in establishing client / server socket connection in UNIX. I want to read from port 5000 of my local system (192.168.9.23). How can I go about this ?

Thanking you in anticipation
Niraj

From: Mark Lucas

Extremely basic server program follows:

```
#include <stdio.h>
#include <sys/types.h>
#include <unistd.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netdb.h>


#define PORT 1066

void main(void) {
  int sock, newsock, i = 0;
  char c, line[BUFSIZ];
  struct sockaddr_in server;
  int addrsize = sizeof server;

  printf("Server\n");
```

```
  /* Step 1: Create a socket */
  /***************************/
  printf("Creating unbound socket...\n");
  if ((sock = socket(AF_INET, SOCK_STREAM, 0)) < 0) {
    printf("Can't create socket\n");
    exit(1);
  }

  /* Step 2: Set up the address */
  /******************************/
  server.sin_family = AF_INET;
  server.sin_port = htons((short)PORT);
  server.sin_addr.s_addr = INADDR_ANY;

  /* Step 3: Bind address to the port */
  /************************************/
  printf("Binding to port...\n");
  if (bind(sock, (struct sockaddr *)(&server), addrsize) == -1) {
    printf("Can't bind address to port.\n");
    exit(1);
  }

  /* Step 4: Listen for and accept connections */
  /*********************************************/
  printf("Ready for connections!\n");
  listen(sock, 5);
  if ((newsock = accept(sock, (struct sockaddr *)(&server), &addrsize)) == -1) {
    printf("Can't accept connection\n");
    close(newsock);
    newsock = -1;
  }
  else printf("Accepted connection\n");

  /* Step 5: Read the Data */
  /*************************/
  while(read(newsock, &c, 1)) {
    line[i++] = c;
    if (c == 0) {
      line[i] = 0;
      printf("Data: %s\n", line);
      i = 0;
    }
  }
}
```

From: Terrance

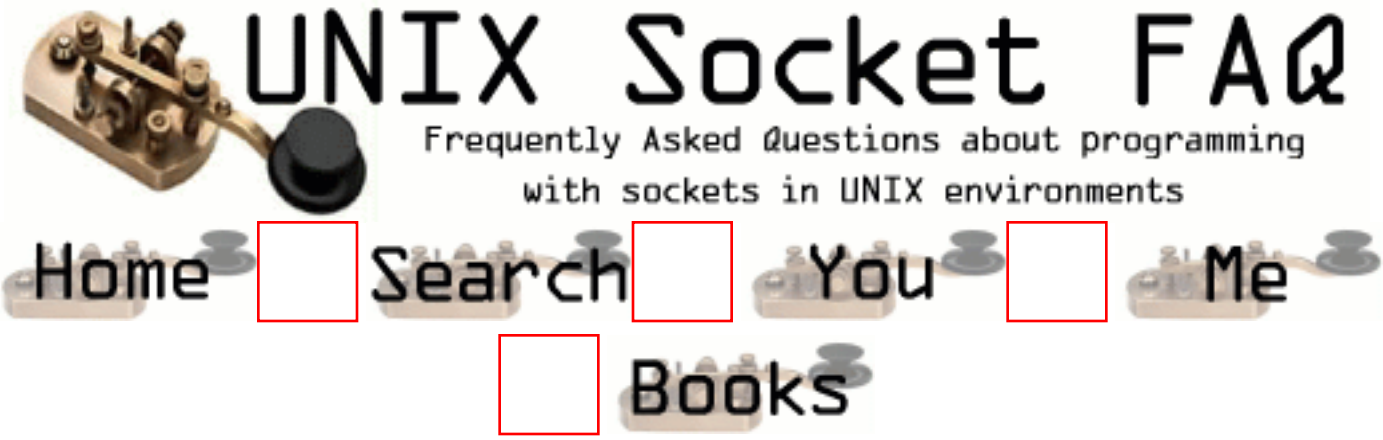
Hi,

I am doing a Web client to load a test server but what I find in using the same read loops is that I am getting different number of bytes each time I receive a page.

I think - something else might be needed but not sure what at the moment - still looking into it.

Cheers,
Terrance

Add a Comment



New Questions: java.net objects for reading port

From: Niraj Shah

Hi,
I want to know what are the equivalent functions in UNIX socket programming for Datainputstream and Outputstream objects of java.net package for reading a port.

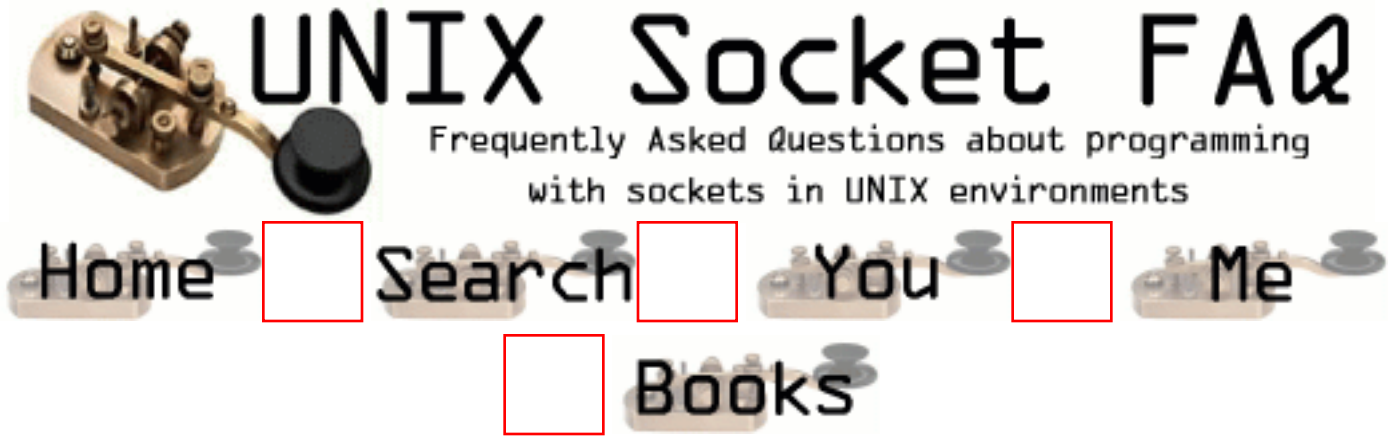
From: Thierry¨PAIN

Send() and recv(). Consult the man of unix, please.

From: Mark Lucas

or read() and write()

Add a Comment



New Questions: Socket Programming

From: Veerendra Moodbidri

Hi everyone,

I have written a "C" program to create a socket in Unix system. After creating, I am able to "bind" this socket to a local host system. My requirement is to "bind" this socket to a Remote Unix system. When i try this I am getting "bind failed" error. I have mentioned the IP address of the remote system in hexadecimal in the C program. After binding i need to read the 5000 port of the Remote unix system (Intranet).

Thanks in advance
Veeru

From: Aner Gusic

Read some man-pages, for example:
bind(2), connect(2), listen(2), select(2), socket(2),
accept(2)

From: venkat

Try printing the actual error code after your bind and see the description in
/usr/include/sys/errno.h
I suspect you need to use "setsockopt" for this problem, after opening the socket.

From: Thierry PAIN

Search perror() in the man too.

From: Jayaprasad Rao K.

When the client's port address does not match with server's
port address, then you will get "bind error". When you get a bind error,
server takes some time to release the port. If you try before

that to execute again, you will get bind error. So if you
get "bind error", change the port number in both client and
server then compile both programs and execute.

From: Hector Lasso

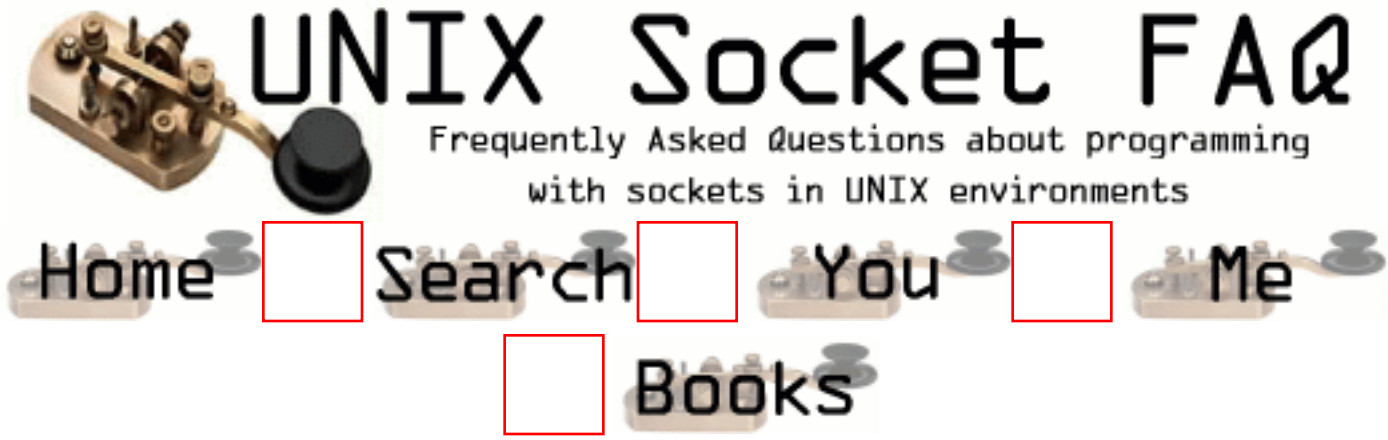
You can only bind sockets on local interfaces (at least it works that way for TCP/IP)

From: bao nguyen

remote ports require you to "connect()", not bind(). bind()
is only for local interfaces.

bao

Add a Comment



New Questions: How do I get a MAC address using code on Solaris?

From: xin

From: S.Chandrasekaran

Hi
You may have to use DLPI primitives to get that information (also have super user access). You may
have to open /dev/le (or /dev/hme) and then use DLPI
primitives to get the information. See man dlpi, man le
and also the file /usr/include/sys/dlpi.h for more
information

From: Heba M Naguib

you ca use the unix command ifconfig -a to show thw mac address of your card,or use arp command

---

From: Ozz Nixon

look at sys/ioctl.h and bits/ioctls.h

You will see in there, you have the ability to get down and dirty with the NIC. (Assuming newer Solaris 2.6 or newer)

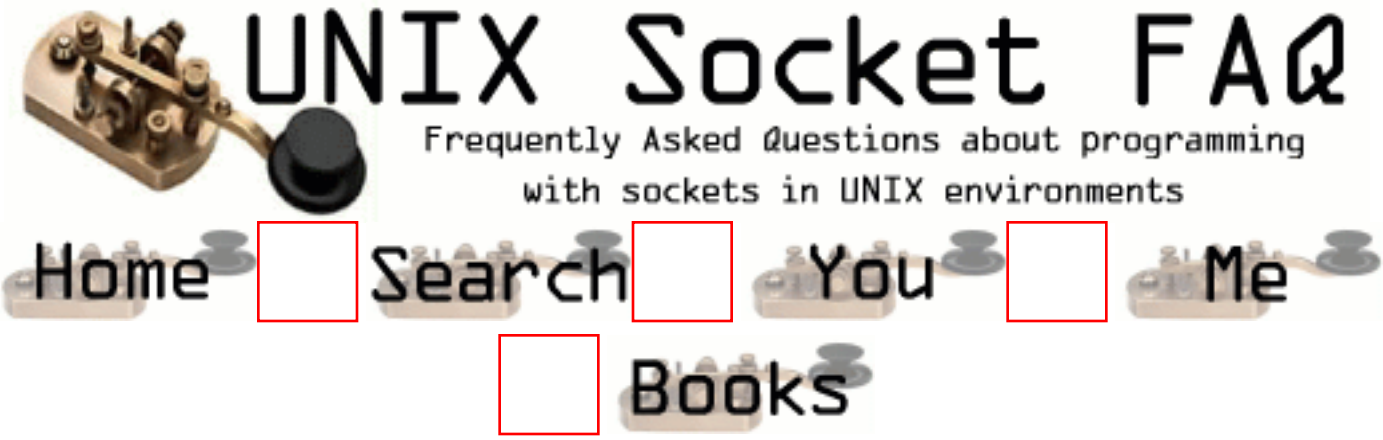
Ozz Nixon
www.dxsock.com

From: Wes

Those ioctls are also available on 2.5.1 (at least on recent patch clusters). Do a truss on ifconfig -a for more clues.

Wes

Add a Comment



New Questions: send object through socket

From: Yan Xu

Now I successfully send strings, integers and float point numbers through socket. Everything works very well after I take care of the byte order of different hosts.

Is it possible to send C++ object through socket? Is it possible to send C++ code through socket? I know Java can do this and this feature was used in mobile agents. But how can I do this in C++?

From: Serge

It's strongly recommended to send structures (and objects) item-by-item, writing its members individually (and taking care of byte order,etc ;)
The reason is difference in allocation customs (alignment,padding,etc) of different compilers even on the same machine/OS.
So if I have to send objects, I have to write methods for writing its members to the socket and for extracting them. Anyway, you have to initialize
the object-receiver, but not simply extract it from the socket.
As for passing code, I can't understand the need to do it. Maybe you think of RPC?

From: Hector Lasso

Hello,

Java uses serialization to create a binary representation of the objects before sending them via opened connections.
This procedure makes possible to send object containers with their content (other objects). And even send references to Remote objects if any (the stub is sent).

To write such code is a relly hard task, so you should try defining some protocol for your set of

objects (you're not going to send any kind of object, are you?) and then using this protocol to exchange the contents/state of the objects.

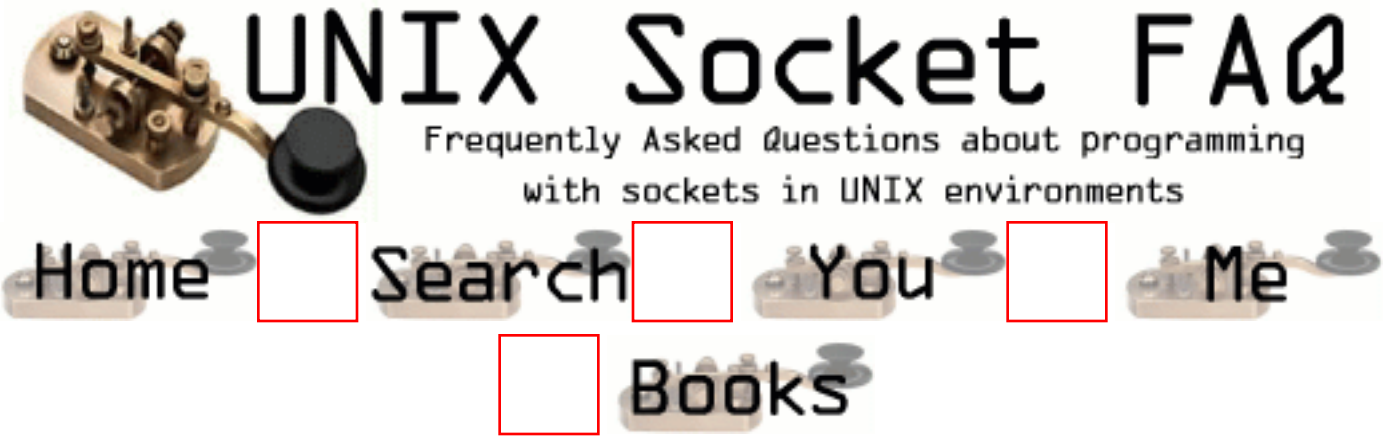
Hope this helps

From: Ozz Nixon

Moving Objects across a socket - can be done. We developed a component for our socket suite that makes this a snap. Basically what we do is push the object into a stream, then push the stream across the socket using pointers. (That is the 2 second description).

Ozz Nixon
www.dxsock.com

Add a Comment



New Questions: Client/Server mystery (sockets)

From: Tara King

I've created a client/server pair of processes on a Unix machine. I edited /etc/services and inetd.conf with the appropriate information so that the client retrieves the server's port number for his connect. I also restarted inetd with the -c option to pick up the modification. This connect also kicks off the server (which was not previously executing.)

By using logging in both processes, I've established that the connections occurs with 0 return codes (that includes checks for the value in errno.) However, the server then just "hangs" on his accept statement. Meanwhile, my client has gone on to his read statement...since the server's hung on accept that's where we stay...hung.

Here's the interesing part...if I begin the server FIRST, and then kick off the client, not only do all the connections occur successfully, but all my test data gets passed back and forth successfully (no "hanging" here.)

My group has used the method I mentioned at the top successfully. Anyone have any idea what I'm missing? (The person who used to do this stuff in my group is no longer working here!)

Thanks,
Tara King

(703) 676-5543
tara.l.king@saic.com

From: venkat

Situation I understand is like this.
If you start your client first and server there after you are facing probelm
Similar kind of proble I faced long back, you can try this
Once after finding that the client is failed to connect to the server first time try for 2 more times then in case of failure still close the socket there and reopen it again and try to connect.
I think it will help you for time being.

From: Jacob O'Reilly

I am not sure if I understand your situation well enough, but it seems to me that you should not have any sockets code in your server program if you are using inetd. inetd handles this for you.

-- Jacob.

From: Taylor

I too had this problem, without the inetd part -- just two
socket apps that wouldn't talk when I ported them to Unix.
I would have to start the server first. Putting a call to
"close" on the socket ID on the client side, after a failed
connect call, seems to do the trick. Oh, followed by another
call to "socket" to initialize it.

Note: theoretically, for a non-blocking socket, the "connect"
call can return -1 with errno set to either EWOULDBLOCK or
EINPROGRESS (heard conflicting reports of which one). This
means that connect will succeed in a moment, and it has been
said that you need to check using "select" for when it finally
does. I haven't witnessed this, but I put things the following
way just in case:

```
// loop of some sort
connect(...);
if (connect succeded)
   break;
else if (errno == EWOULDBLOCK)
   sleep(some time); // let it finish
   break;
else
   close(sock);
   sock = socket(...);
end
```

Taylor

From: ilgar hidayatov

Het men please write some example programms about sockets !
And please if you can send me FAQ about select() how it works with simple words !

From: Ashutosh Sapre

Hi !!!!

Use infinite loop in the client program to connect to
the Server . If the Server is not up , Let the client sleep for 2 sec/Minutes and then try connecting again . Once the connection is established then do the binding etc

---

From: algoud ahmed

i recherche a programme with C, for the socket

Add a Comment

New Questions: Connec to different hosts/ports from the same host/port

From: Fabio Bettoni

I know that 2 sockets in a system can't have the same
protocol/local_host/local_port/remote_host/remote_port.
At least one of them must be different.
It's possible to connect from the same local_host/local_port to different
remote_hosts/remote_ports.
If it's possible, how can I do a bind from differents sockets on the same local_host/local_port ?

P.S. I assume host = host IP address

From: Baswaraj

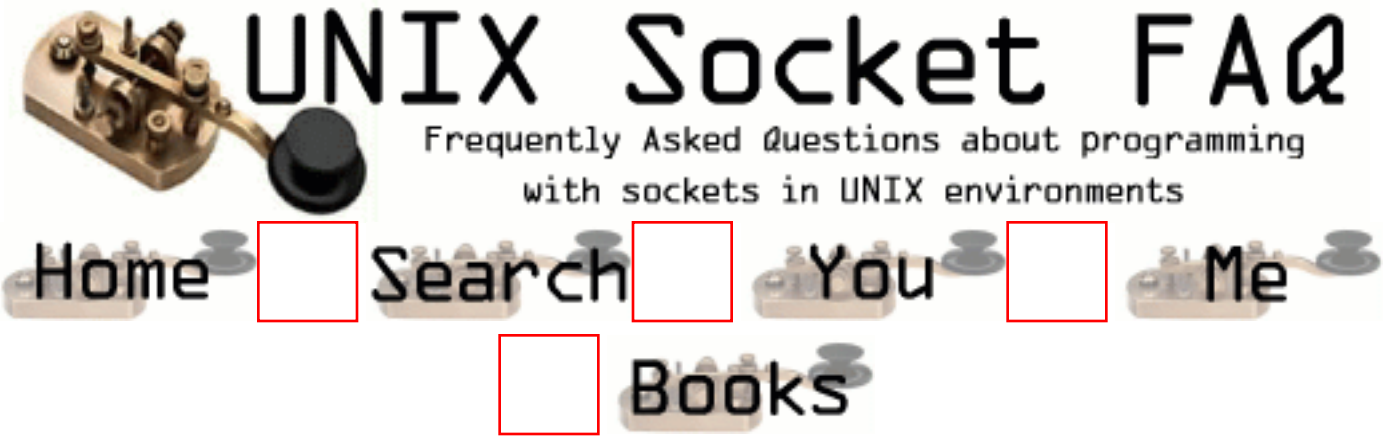
It's nit possible to bind two differernt sockets on the same
machine using same port.

Single socket bindes on remote machine can accept more than one connection from
the other Machine.

From: Harshit

It is possible to bind 2 sockets with same port and host IP,
as long as protocols are different. I bet that !!

Add a Comment



New Questions: Compiler Difficulty

From: Han Guowen

How to compile the programs.

```
/* socket_server.c */

#include <sys/types.h>
#include <sys/socket.h>
#include <sys/un.h>
#include <stdio.h>

#define NSTRS 3 /* number of strings */
#define ADDRESS "mysocket" /* addr to connect */

/* Strings send to the client */

char *strs[NSTRS] = {
  "This is the first string from the server.\n",
  "This is the second string from the server.\n",
  "This is the third string from the server.\n"};

main()
{
  char c;
  FILE *fp;
  int fromlen;
  register int i, s, ns, len;
  struct sockaddr_un saun, fsaun;

  /* Get a socket to work with. This socket will be in the UNIX
    domain, and will be a stream socket. */
```

```
if ((s = socket(AF_UNIX, SOCK_STREAM, 0)) < 0){
  perror("server: socket");
  exit(1);
}

/* Create the address binding to */
saun.sun_family = AF_UNIX;
strcpy(saun.sun_path, ADDRESS);

/* Try to bind the address to the socket
   Unlink the name first so that the bind won't fail.
   The third argument indicates the "length" of the structure,
   not just the length of the socket name */
unlink(ADDRESS);
len = sizeof(saun.sun_family)+strlen(saun.sun_path);

if (bind(s, (struct sockaddr *)&saun, len) < 0){
  perror("server: bind");
  exit(1);
}

/* Listen on the socket */
if (listen(s, 5) < 0){
  perror("server: listen");
  exit(1);
}

/* Accept connections.
   When we accept one, ns will be connected to the client.
   fsaun will contain the address of the client */
if ((ns = accept(s, (struct sockaddr *)&fsaun, &fromlen)) < 0){
  perror("server: accept");
  exit(1);
}

/* We'll use stdio for reading the socket */
fp = fdopen(ns, "r");

/* First, send some strings to the client */
for (i=0; i<NSTRS; i++)
  send(ns, strs[i], strlen(strs[i]), 0);
  /* Then, read some strings from the client */
for (i=0; i<NSTRS; i++){
  while ((c = fgetc(fp)) != EOF){
    putchar(c);
```

```
      if (c == '\n')
        break;
    }
  }

  /* Terminate the connection */
  close(s);

  exit(0);
}

/* socket_client.c */

#include <sys/types.h>
#include <sys/socket.h>
#include <sys/un.h>
#include <stdio.h>
#include <stdlib.h>

#define NSTRS 3 /* number of strings */
#define ADDRESS "mysocket" /* addr to connect */

/* Strings send to the client */

char *strs[NSTRS] = {
  "This is the first string from the client.\n",
  "This is the second string from the client.\n",
  "This is the third string from the client.\n"};

main()
{
  char c;
  FILE *fp;
  register int i, s, len;
  struct sockaddr_un saun;

  /* Get a socket to work with. This socket will be in the UNIX
    domain, and will be a stream socket. */
  if ((s = socket(AF_UNIX, SOCK_STREAM, 0)) < 0){
    perror("client: socket");
    exit(1);
  }

  /* Create the address connecting to */
  saun.sun_family = AF_UNIX;
```

```
  strcpy(saun.sun_path, ADDRESS);

  /* Try to connect to the address. For this to succeed, the server
     must already have bound this address, and must have issued a
     listen() request. The third argument indicates the "length" of
     the structure, not just the length of the socket name. */
  len = sizeof(saun.sun_family) + strlen(saun.sun_path);
  if (connect(s,(struct sockaddr *)&saun, len) < 0){
    perror("client: connect");
    exit(1);
  }
    /* Use stdio for reading the socket */
  fp = fdopen(s, "r");

  /* First read some strings from the server */
  for (i=0; i<NSTRS; i++){
    while((c = fgetc(fp)) != EOF){
      putchar(c);
      if (c == '\n')
        break;
    }
  }

  /* Then send some strings to the server */
  for (i=0; i< NSTRS; i++)
    send(s, strs[i], strlen(strs[i]), 0);

  /* Terminate the connection */
  close(s);
  exit(0);
}
```

From: David W. Huang

I also have this problem. I am a starter in socket programming. When I first copied someone's program, and using "gcc filename" command to compile. But the link always failed because the compiler does not know socket, connect, bind, etc even I put all the relevant header files in my program.
I also want to find out how to compile socket program. There must be some easy tips!!!

From: Sam

You should compile it as:

gcc filename -lnsl -lsocket
it will work

From: ccwork

It should be noted that in linux we don't need
the -lsocket and -lnsl

From: Xi Chen

-lnsl -lsocket is generally needed on Sys V based systems which put BSD specific functions into separate libraries from its C library. If -lnsl -lsocket is needed, you probably also need -lucb for other BSD functions. You need these options for Solaris.

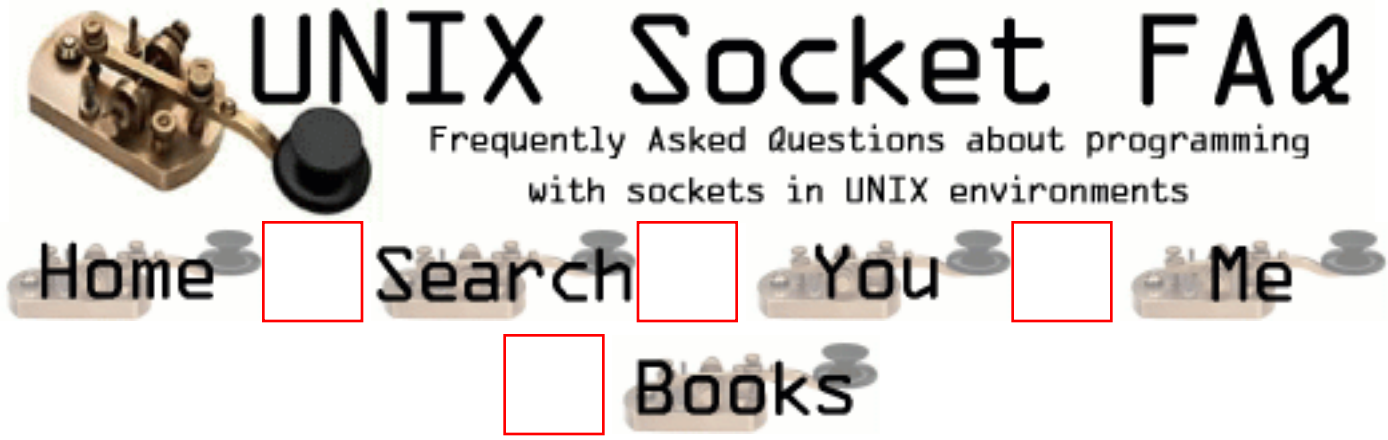
From: roshan

do anyone please tell if we can have many clients in the server program above without the fork option,ie if there has to be inetraction between the clients of the same network..

From: Norvin

The linker options -lsocket and -lnsl don't seem to work on HPUX. Any special issues here?

Add a Comment



New Questions: How to know if the sending buffer is fully empty?

From: Paul Mamin

How could I know that the sending buffer is fully empty (i.e. all data were sent or almost sent to the peer)?
By select() I could only know that there's some space in the sending buffer and nothing more.

From: Jill

Hi Paul,
This is how I've checked my data. Since it's ANSI C you can use it on both Unix & PC platforms.

```
int send_chk;

/* this returns number of bytes sent */
send_chk=send(sock, szMsg, strlen(szMsg), 0);

If (send_chk != strlen(szMsg))
{
   printf("send_chk = %d\tszMsg = %d\n\n", \
   send_chk, strlen(szMsg));

   printf("Error=> data sent not equal to message size\n");
   exit(0);
}
```

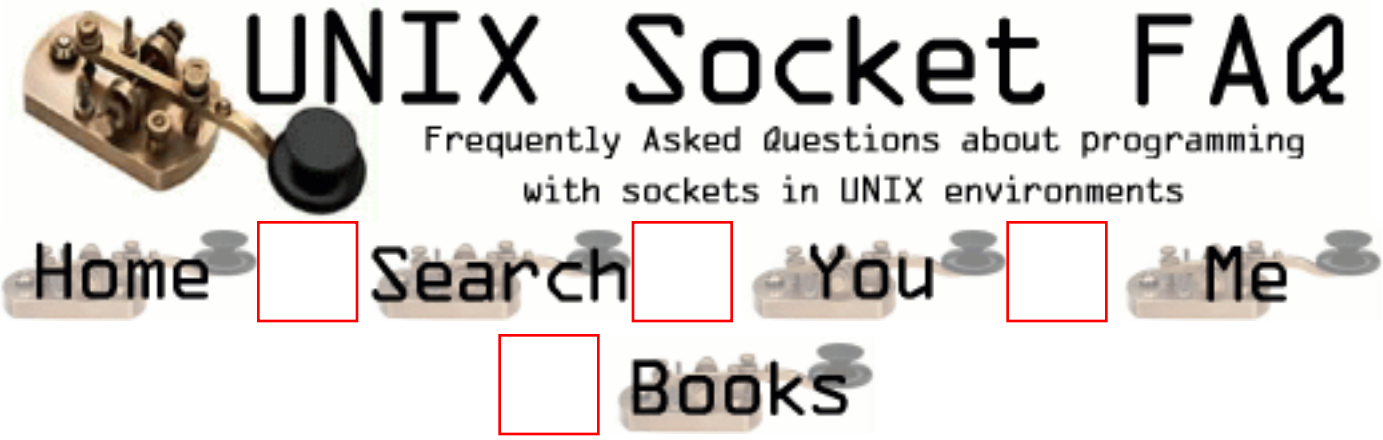
From: Hector Lasso

Hi,

I haven't found a way to check the number of bytes available in the send or receive buffers. getsockopt() with SO_RCVBUF/SO_SNDBUF return the size of the buffers however.

Do you need to know the available size of the buffers just for information or to do something special. Maybe there is another way to do it without having the available size.

Add a Comment



New Questions: How do I get the IP Adress to which a client connected to my server ?

From: Jens Thiele

I wrote a TCP Socket Server and if I have a incoming connect I
can determine the IP of the client. But how do I get my IP
(The one he connected to) ?

From: Bernie Boudet

If you bind to INADDR_ANY rather than a specific address then your server will listen() on all the host's IP's.

After you accept() the new connection use getpeername() on the new socket fd to get the address and port the client connected from.

In the same manner, call getsockname() on the new socket fd to get the address and port the client connected to.

You can also use getsockname() on the original listening socket fd after you bind() it to find out which port it will be listening on (in case you bound to port 0).

From: Paresh Mehta

Hi,

I think after you accept your client using
accept() function, thereafter if you use
inet_ntop(1,2,3,4) function
where 1,2,3,4 are
1. AF_INET
2. &struct sockaddr_in.sin_addr <- this is the accepted client
in accept function.

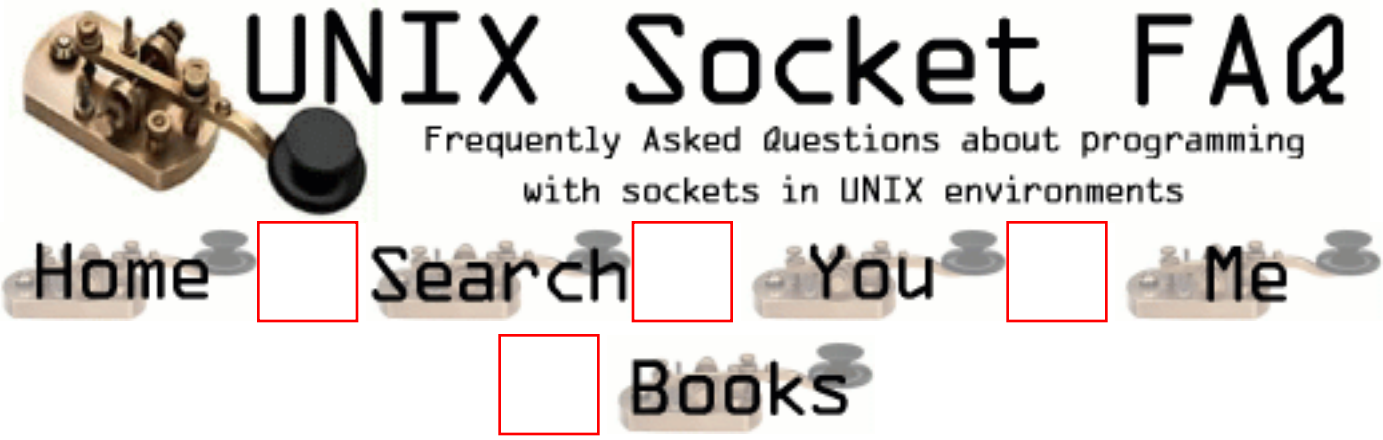
3. any character array or a character pointer.
4. sizeof(character array you have defined)

for refering the inet_ntop() function you can refer to the
unix network programing vol 1 by Stevens.

From: bob

bang

Add a Comment



New Questions: client server program

From: Raghu

Hi i want to write a program of client and server in which
i have to send a file from client to server and when server
receive the file it send back message to client. Similarly client
want to retrive a file from server how can i do this.

From: Vic Metcalfe

I think I need to add a section called "Homework"... ;)

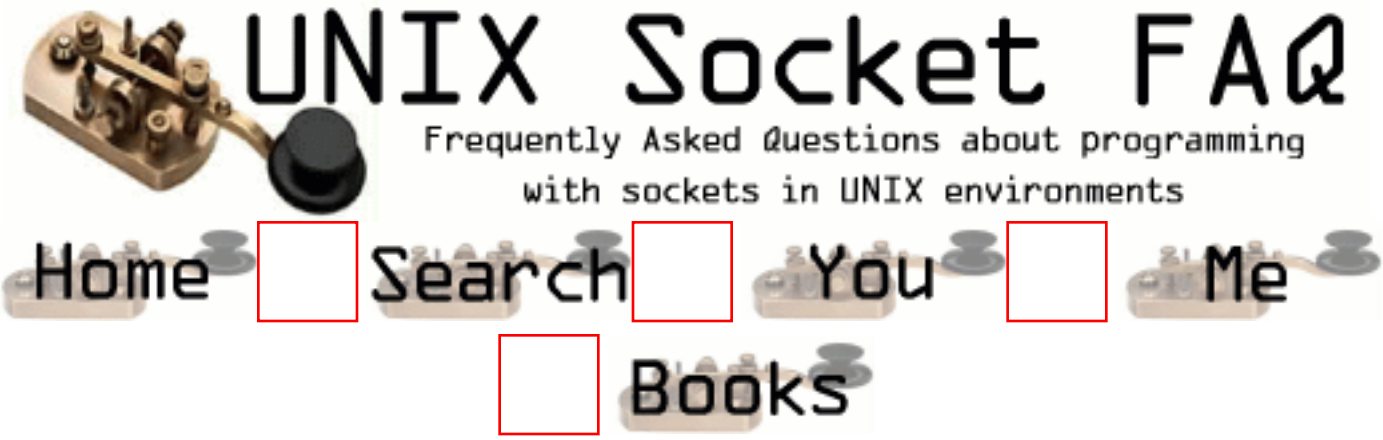
From: rajiv

gethostbyaddr{}write me information about it

From: Erric

I also want this sample program.But this program should be work even I take out the cable and until I plug in back.

Add a Comment



New Questions: What are the limits on the number of sockets?

From: Michael Friedman

What are the limits on the number of sockets from a single server?

I'm looking at an application where a single server will have relatively light weight connections to over 100,000 clients. I strongly suspect that I cannot dedicate one socket per client, but I'm not sure.

Are there any OSes with higher limits on the number of possible sockets?

From: Gavin

How can you increase the FD_SETSIZE to allow more then X amount of connections that the OS defaults to.

From: Ville

I am trying to accomplish a similar feat, so far some test code has shown that Win2k can do about 8000 connections, dropping with lower versions of NT. Win9x can only do between 80 and 120, so my hope was Linux would be the answer. So far I've missed it, being able to get 1000 connections - some have pointed me to multi-threading or multi-processing, but as Im new to both topics I have not been able to get anywhere. Any advice would be appreciated.

From: Rob Seace

Well, there are a few different things that will limit you from having ridiculously huge numbers of sockets... One is the OS-imposed limits on number of file descriptors; both on a per-process basis, and system-wide... Another is your libc's "fd_set" size ("FD_SETSIZE")... Another is simple exhaustion of system resources (each open FD sucks up some

amount of RAM in your process, and likely in your kernel)...

But, if you're determined to have such insanely large numbers of sockets, you generally CAN... Some systems let you pre-defined "FD_SETSIZE" prior to #include'ing <sys/select.h>, to give it any arbitrary size... I've never had a need to do so, so I can't say how well it works, or which systems support this... But, as a last resort, you could also directly change your system headers to define your new larger value, and recompile your libc (and, anything and everything else which uses it)... ;-)

As for OS-imposed open files limits, the ways to overcome those are pretty OS specific... Per-process limits can generally be changed by the process (if it has the perms to do so) with setrlimit()... System-wide limits vary greatly in how to change, if they're even changable at all... Of course, if you have kernel source, you can always recompile with any arbitrarily large size you like... But, many OS's let you change the values on the fly, as well... Eg: with Linux, you can change this via various files under "/proc/sys/fs/" ("file-max", "inode-max", etc.; see "proc.txt" under the "Documentation" subdir of your Linux source tree for more info; it also tells you how to change the fixed per-process upper-limit, which unfortunately requires a kernel recompile)...

But, before you go and do all this nonsense, maybe you really ought to rethink your design... I can think of no sane reason why any process should ever need more than 1024 open files... In fact, I can't think why one would ever need more than about 100 or so... Even a heavily busy server, doing TONS of stuff for millions of clients... The only situation I can envision is something like a massive MUD/MUSH/MUCK/chat-server type thing, where you wish to support a ton of simultaneous users, and each one maintains a persistent TCP connection all the while they are using the system... If I were going to attack such a problem, I would change the design: make it such that the clients do NOT hold open TCP connections all the time; make them connect and disconnect after every message/command/whatever; or, better yet, use UDP messages for all communication, a la Quake servers... If you must maintain persistent state for all 'connected' clients, it does become a bitch to deal with, though... And, granted, real persistent TCP connections

would be the ideal solution... But, the overhead is just
going to kill you... Having that many open file
descriptors, even if you hack your system to allow it, is
just going to suck performance-wise...

From: Derek Becker

Actually, if it's the server you're talking about, then you can use on port. Sockets are determined by the 4-tuple of <destaddr,srcaddr,dstport,srcport> so if you had 100,000 clients then the srcaddr and srcport would be different for every client. The only way you would run into problems is if the 100,000 clients are all on the same machine.

Derek

From: Andy Ning

Another way to support huge number of simultaneous TCP connection
is to split the connections among several processes, say, one
process handle 512 connections. The Parent process listen
and accept connections and pass the accepted sockets to child
processes, and the child process handle the actual reading
and writing.

I have test the method and it works well. Only thing I'm not
sure and don't know how to figure out is the performance.
Say, one process handle 512 connections versus two process
each handle 256 connectons, which one is more efficient?

Andy

From: Xuemei Zhang

Hi, all

Any one knows how to change the TCP connection's
TIME_WAIT time? My program will process many incoming requests all the time, and it will create one TCP connection for each request, and after processing it, the connection is closed. However, seems the closed connection will stay there for more than 1 minutes, there will be hundreds of TIME_WAITing connections there. So, I want to reduce the TIME_WAIT time to smaller one like 10s. How to do it, is the small delay good for all applications?

Thanks in advance for kind help!

Xuemei

From: Jose Luis Tur

The amount of time a TIME_WAIT connection spends is stored
as miliseconds in the parameter TCP_TIME_WAIT_INTERVAL.
Usually is equal to 240000 (4 minutes); but you could easily
set to 60000 (1 minute). In Solaris (the OS that I'm working for)
it's very easy, you only need tune the parameter TCP_CLOSE_
WAIT_INTERVAL (it works like T_T_W_I). You can read it with :

ndd /dev/tcp tcp_close_wait_interval

and you can set a new value with :

ndd -set /dev/tcp tcp_close_wait_interval nn (where nn it's the value)

Only newly created connections will work with the new configuration.

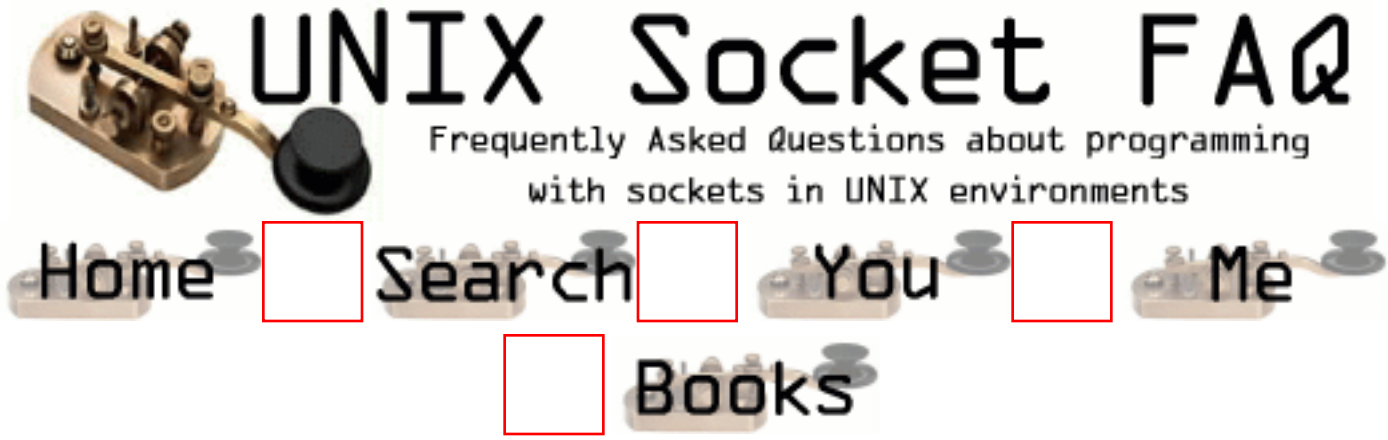
From: Frank

I might be missing something, but if you're trying to maintain 100,000 connections at once, that's impossible with TCP/IP - you can only have 64,000ish ports connected at the same time because that's all that TCP can handle...

From: Willy

For an extremely large number of light weight connections (i.e. few bytes transmitted at large...by computer standards...time intervals), why not use a connectionless protocol like datagrams and do the error-retransmit logic internally?

Add a Comment



New Questions: How much new Bytes are in the Buffer for Inread ?

From: Stefan Schnuderl

How can I get the number of new bytes in the buffer, which have been received from the Server since the last call of read(socket, *buffer, len) ?

/\
||
I need to know this len.

From: Gerrit van Dyk

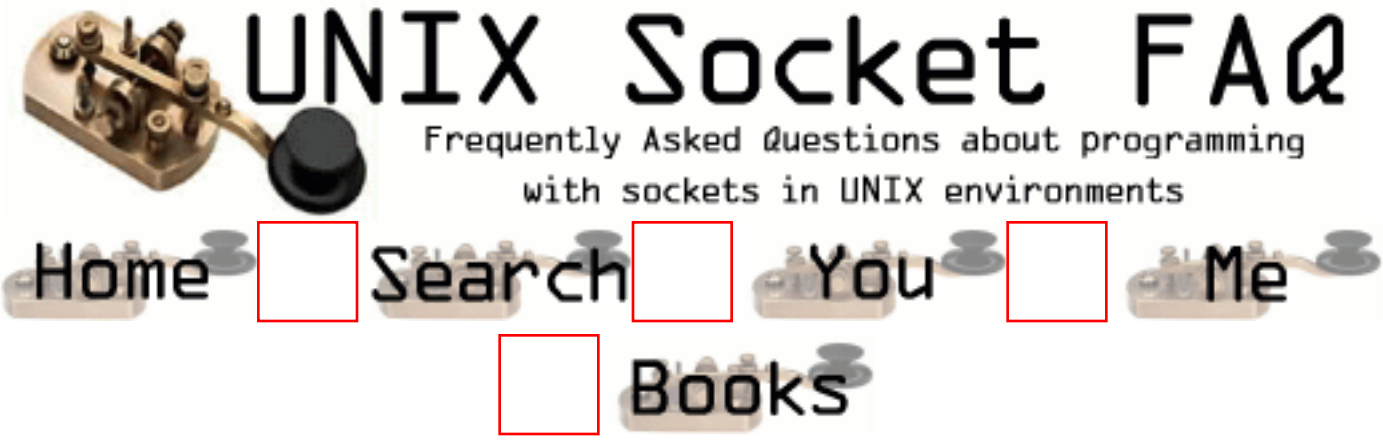
You can do it as follows:

```
unsigned availBytesOnDescriptor(int aDescriptor)
{
  unsigned availBytes;

  if (ioctl(aDescriptor,FIONREAD,&availBytes) != -1)
    return availBytes;
  else
  {
    perror("availableBytes");
    return 0;
  }
}
```

From: Daniel Liljebladh

Fantastic ! It works excellent :)

Add a Comment



New Questions: Socket and thread

From: Mario

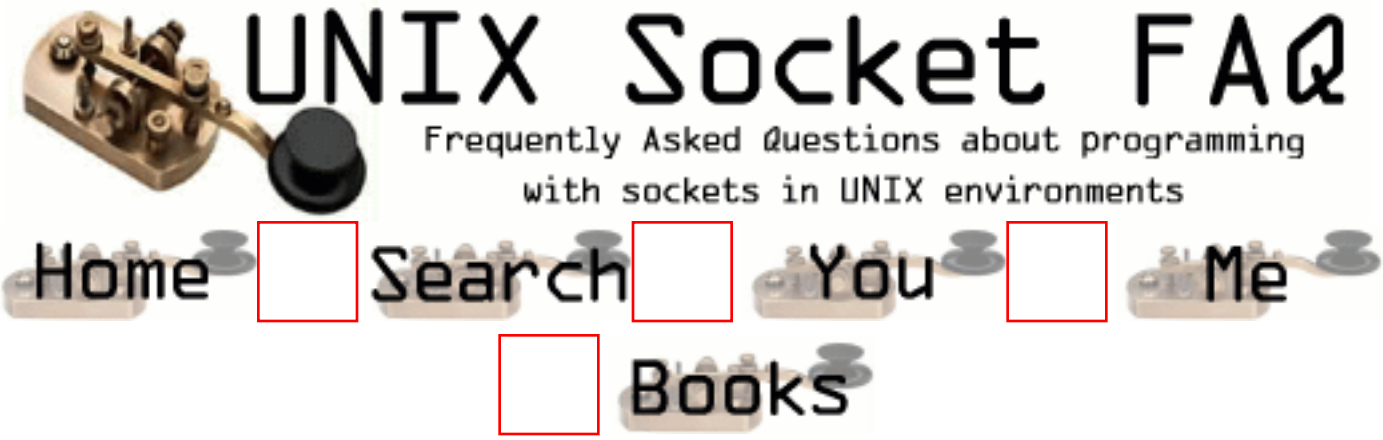
How can I do to that one application work with sock and thread sametime?

From: Mark

go see this address:
http://www.cis.temple.edu/~ingargio/old/cis307s96/readings/threads-server.html

---

Add a Comment



New Questions: Undefined Symbol Problem

From: Eric Johnston

I have been able to compile, but not link, a client-side socket program, and have been consistently receiving the following error messages:

Undefined first referenced
 symbol in file
socket testsocket.o
inet_addr testsocket.o
connect testsocket.o

All are defined in socket.h, which is included in the source code.

This only appears to occur on Solaris 5.6 (SPARC) -- Which is my target platform.
I have not received this error message on Linux 6.0 (Intel) -- where the test program runs properly.

Does anybody know what could be/is causing this?

From: Brent Hendricks

Under Solaris (and possibly other platforms?) you need to
explicitly link with -lsocket and -lnsl

From: rad

Thanks for the tip.
This linker information should be more easily obtainable.

From: Thierry PAIN

You don't need to search very far. The man ;-))

From: Kalaitzoglou Panayotis

Thank you very much for this information. It is
not easy to find I think.

From: James Kassabian

The information you were looking for is the first thing listed in the SYNOPSIS section of the socket(3N) or socket(3XN) man pages.

From: Anurag Mishra

This undefined symbol can be removed by properly linking the header file.
If its compiling then do like this & try.

cc filename -L/usr/lib -lsocket -lnsl

try & reply

From: sofarsoclose

I have been same problem and I haven't get the answer for this
question.I can't find words to say how thankful I am.

This is a great place to get information.

Thank you so much !

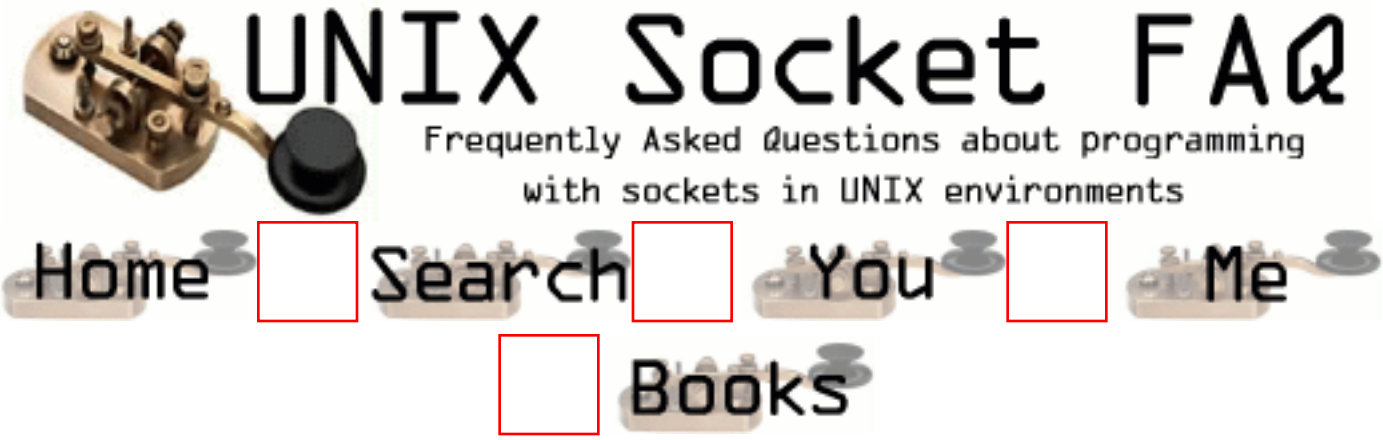
From: jason

thanks for help.

i meet another problem, i use method above to compile my program, i use the inet_aton method in program, when compile i meet the "Undefined symbol inet_aton". how can do it. Thank you for advance.

jason

Add a Comment



New Questions: synchronous and asynchronous communication using sockets

From: Paresh Mehta

Hello sir/s,

Can you please tell me how does the synchronous and
asynchronous communication differ from each other
with reference to sockets?

From: Gangadhar

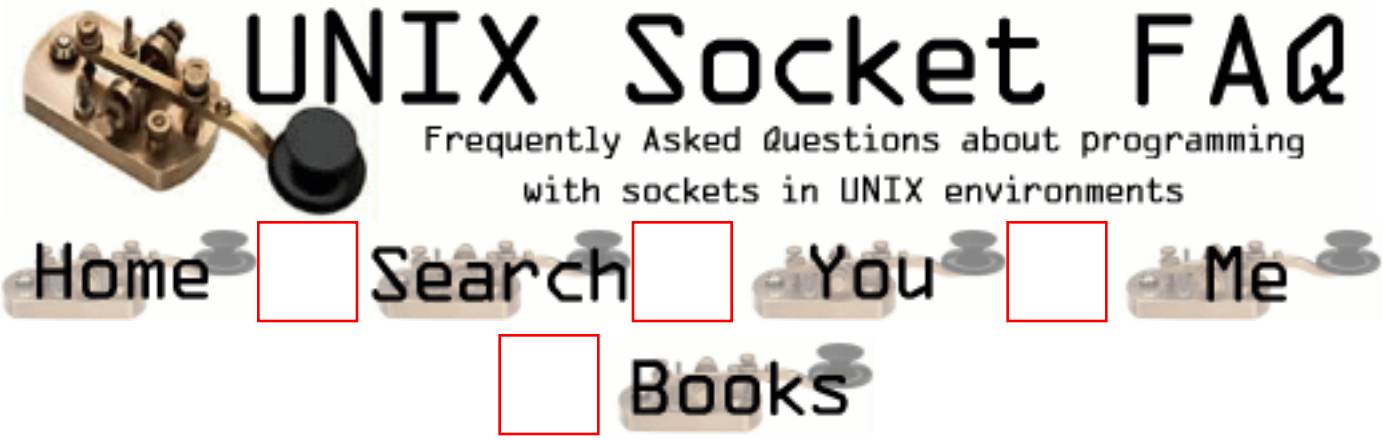
Hi,
Can you please tell me how I can do asynchronous socket programming in UNIX.

thanks
Gangadhar

From: James Kassabian

Use the select(3C) function from the standard C library for synchronous I/O multiplexing. You pass 3 pointers to file descriptor sets (fd_set objects), among other arguments, to the function. An fd_set object is just a bit vector. One fd_set object is for those file descriptors to be tested for reading, one fd_set object is for those file descriptors to be tested for writing, and one fd_set object will indicate which file descriptors have pending error conditions.
Initialze the fd_set objects with the FD_SET macro. select(3C) modifies the fd_set objects passed as argument on output. If a file descriptor is not ready, select(3C) clears the bit for the file descriptor in the corresponding fd_set object. You use the FD_ISSET macro to test if the bit for a file descriptor has been cleared or not. If it was not cleared it is ready. Always test the same bit in the error fd_set object first. If the bit is set in the error fd_set
object you should test the file descriptor for errors instead of using it.

Add a Comment



New Questions: PORT NOT RESPONDING

From: VENKAT

Dear All,

I got windows client and Unix server.
Some times client will not connect to the server it returns 710 error.
If I start and stop the server instance or if I try with another port at
client side it works.

Could you please suggest me what could be the possible problem.
Hanging up of the port is not very clear to me.

Thanks
-Venkat

From: Asha

Hi,

We are facing the same problem. if you figure it out, could you please let us know.

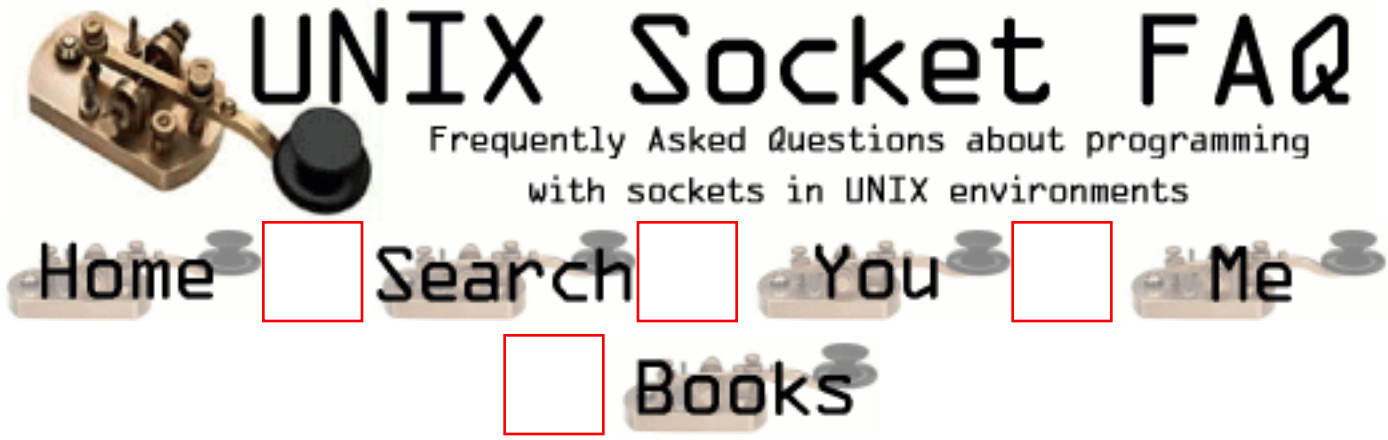
Asha

From: Rob Seace

You say that if you change the port that the client is
using, it works? Then, my guess would be that the problem
is the old connection is still in the TIME_WAIT state, and
hadn't yet timed out, so you wouldn't be allowed to connect
with the same exact source port and IP to the same exact
destination port and IP... See the other sections in the
main FAQ here for more details about the TIME_WAIT state...

But, I think you already came up with the logical solution to the problem: use a different source port on the client for every connection... That's what you SHOULD be doing, anyway... In general, unless there is some very special need for a specific source port, all clients should let the system assign them an unused port, and should NOT specify their own... I believe that's mentioned here in the FAQ somewhere, too... Ie: your client should simply not bother with calling bind() at all, and just let the connect() handle it for you; or, if you must call bind() for some reason, just set "sin_port" to 0 in the sockaddr_in struct, and then bind() will generate a new port# for you, and fill it into "sin_port" for you...

Add a Comment



New Questions: How....

From: Bob Andrews

Does anyone have any idea how to create a client using sockets that does the following,

Follows a starting URL, parses that file for url links, then puts those links into a log file. It then verifies each of the links and returns their status using HTTP responses??.

Bob

From: Thierry PAI

From: Thierry PAIN

From: Thierry PAIN

You don't want, I bring to you a coffee too ;-))

From: Michael Huber

Try opening a socket to port 80 of the web server.
Use telnet or nc to see what the results will look like.
You can get nc (NetCat) at:
http://www.l0pht.com/~weld/netcat/index.html

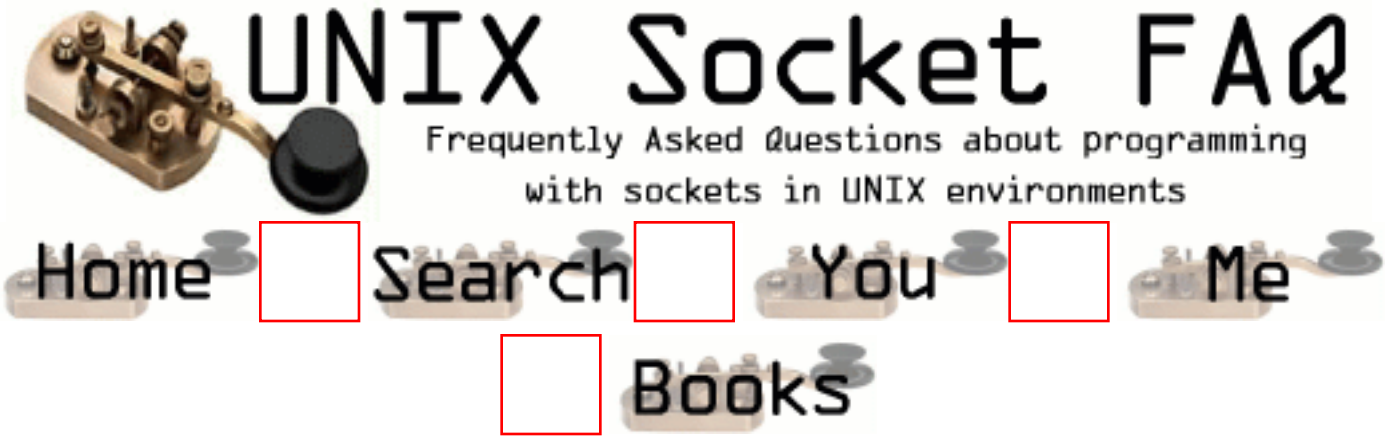
Try getting a page from some server you know of...
nc www.google.com 80 (or telnet www.google.com 80)
then type
GET /index.html
It should return the source code for "http://www.google.com/index.html"
Google has a really short timeout, so you have to type fast (or try a different server, like yahoo).
The GET command on port 80 also works for CGI scripts.

I'm sure you can rig something up that searches for "<a href=", and fwrites the URL into a log file.
You'll probably want to keep track of the current path (server and directories), and write that in the log file before the relative URL's (like if a link just takes you to "mypics.html" or "pics/michael.html" or even through the root directory like "/cgi-bin/myscript.cgi")

You probably don't even need to use sockets...just call nc from your C program.
system("nc www.google.com 80 > tempfile");
<OR>
FILE *fp = popen("nc www.google.com 80", "r");
loop using fgets(buffer, bufsize, fp);
pclose(fp);

popen() does something like re-route stdout from the system call to FILE *fp
...check the man pages.

Add a Comment



New Questions: Asynchronous client

From: jagadeesh

Hello,
I need to send more than one request to server to get the respective response message. I need to develop an Asynchronous client. pl. send me some details .
Thanks
Jagadeesh

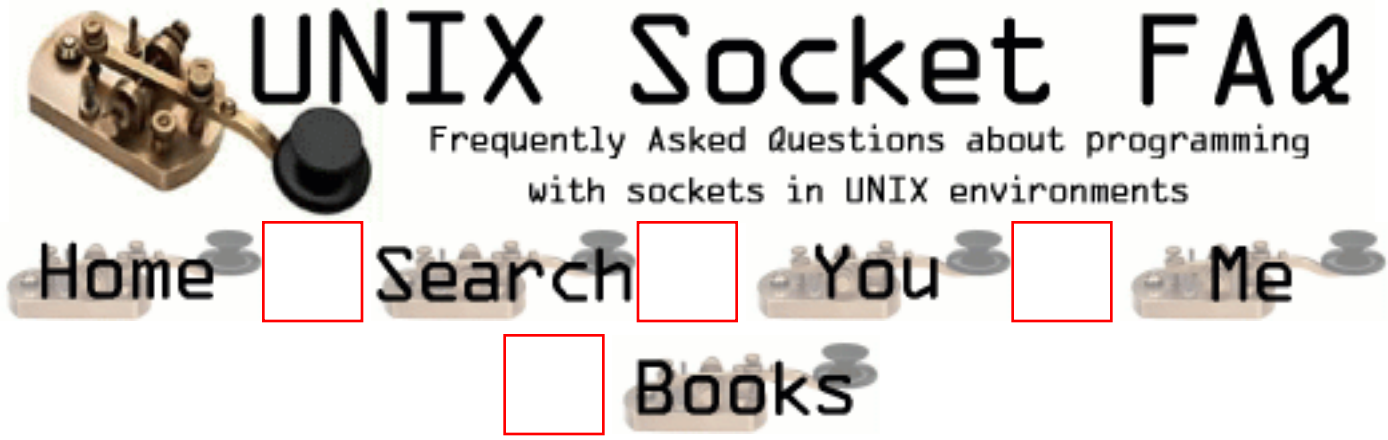
From: Mahafuz

Pls try creating thread where every thread will send a request to server & get the respective reply.

or

open several connection to the erver. send request
to server. then use select() on the socket set
to procress response.

Add a Comment



New Questions: What do you do to constantly poll a file on the client side (Unix) and send it to server (NT) ?

From: Navin Agrawal

Can someone help me URGENT...??

Q : In short I am developing a program which transfers data from Unix box to NT. Now NT acts as server and Unix box is the client. I have developed the connectivity between the 2. My problem is that this Unix Box has to constantly keep on polling on one HEX file (which may reach a size of 35GB) and any NEW changes (just the changes) in this file should be sent to the NT side where the necessary processing will be done.

Now can someone give me the proper code on the Unix side. And this is URGENT because I am really short of time and have to complete this in a day or two. I hope I made myself clear about my question.

From: John

Your client on unix should simply read the file and
output this to the server over the socket. When it gets
to the end of the file, the process should wait a short
interval (using sleep() or select() functions) and then attempt to read more. This is exactly what is done by
the unix command "tail" with the -f option.

Since the client process never knows when more is going to be added to the file, and since using select on the file
won't help you, in this case you must simply poll the file
by repeatedly attempting to read it.

I hope this answers your question.

From: jay

John has given the solution but I think u should poll the file for updatation means u have to check when file is updated if the time is greating the the last check u can transfer the file to unix. For file updation u can use standard C library function.

From: Jay

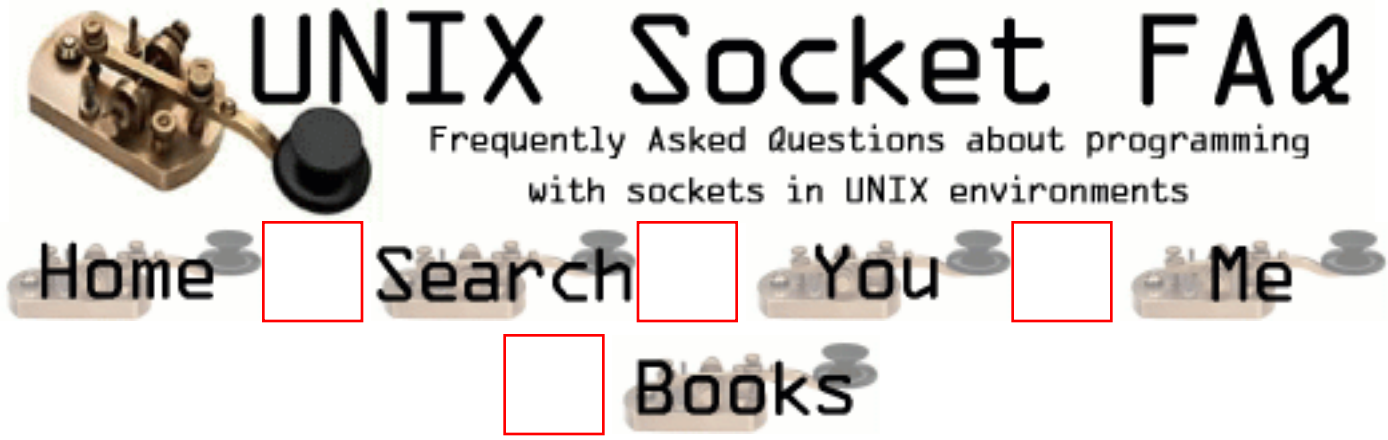
Sorry transfer to the NT..... My mistake

From: Wes

The simple is answer is: don't re-invent the wheel! You've described pretty much exactly what rsync() does. It is open source, and can be made to run on Windows.

Wes

Add a Comment



New Questions: Asynchronous client ?

From: jagadeesh

Hello every one !
How can I send more than one request to the server. How is the client know the status of each connection ? Does the client is creating a separate sockect for each request?

Thanks

From: VP

One possible methodology for Async Client :

Async client :

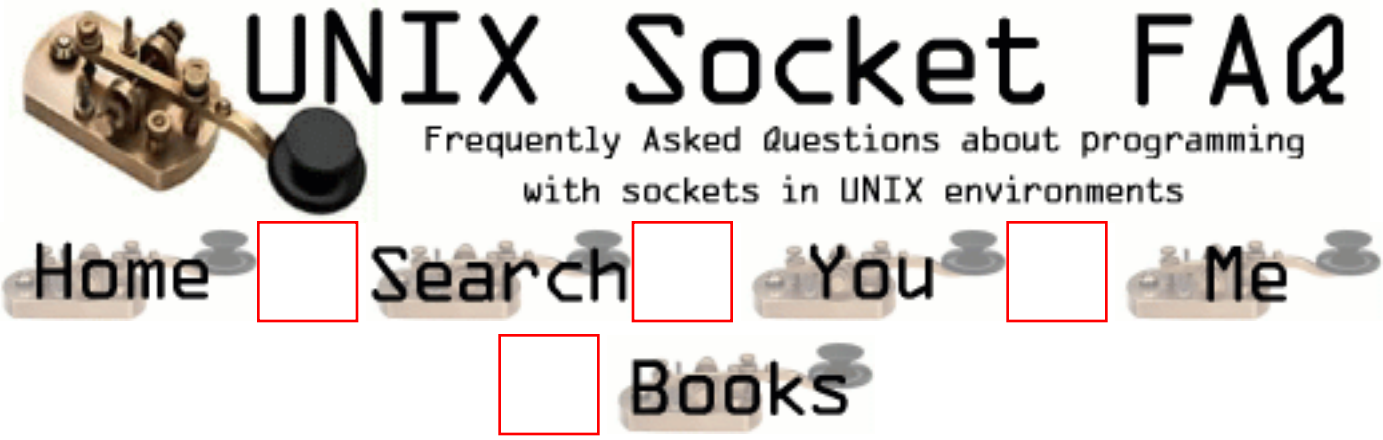
. Open socket
. Establish connection
. set it async (can use fcntl() F_GETFL & F_SETFL options)
. fork ()
. Child :
    for(;;)
        Take/Read input messages/data - say user_query
        use select() for 'writable' check/wait
        Send across
. Parent :
    for(;;)
        use select() for 'readable' check
        Receive processed output/data
        Do the needful - say, answer user query,
        update DB etc

The basic idea is to getaway with BLOCKING calls of
sockets() and as shown above, I think we can use select()
for achieving the same purpose and the same can be extended

to server side also.

--
VP

Add a Comment



New Questions: writeset in select

From: AK

I have not been able to understand the purpose of the
writeset in the select() system call. Can anyone explain
with an example in what cases it is useful ?

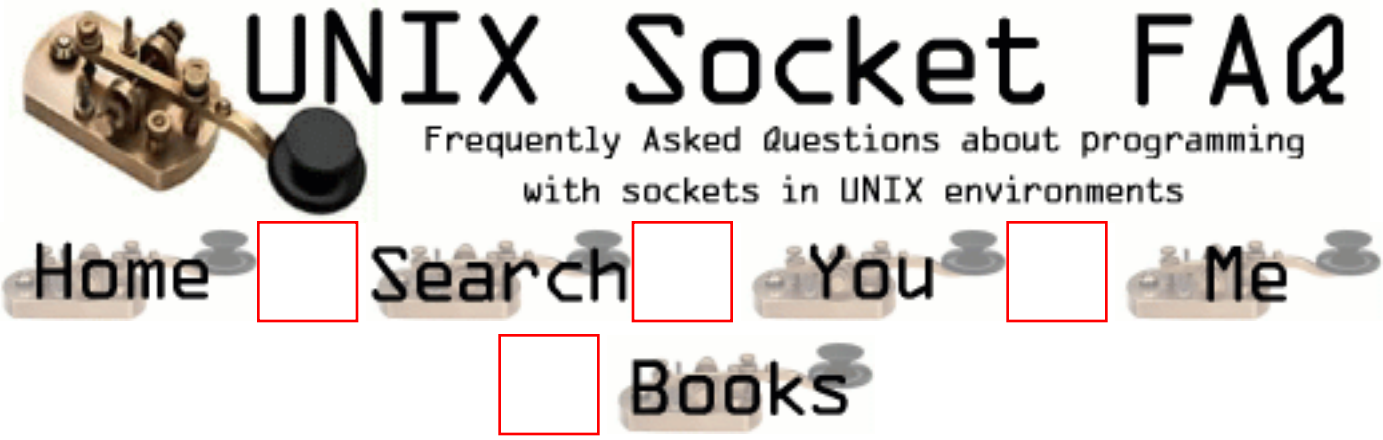
TIA,
AK

From: John

The write set allows you to test when you can write to a socket. This is useful if you are writing a lot of data to a socket and the buffer fills up. The select will then indicate that you cannot write on the socket, so your process can do something else, instead of blocking while trying to write until the buffer empties.

If you are not writing much to a socket, of have a program that doesn't have anything else it can usefully do instead of writing, then you can let the program block on write and not bother with the write set in the select().

Add a Comment



New Questions: no client server setup !

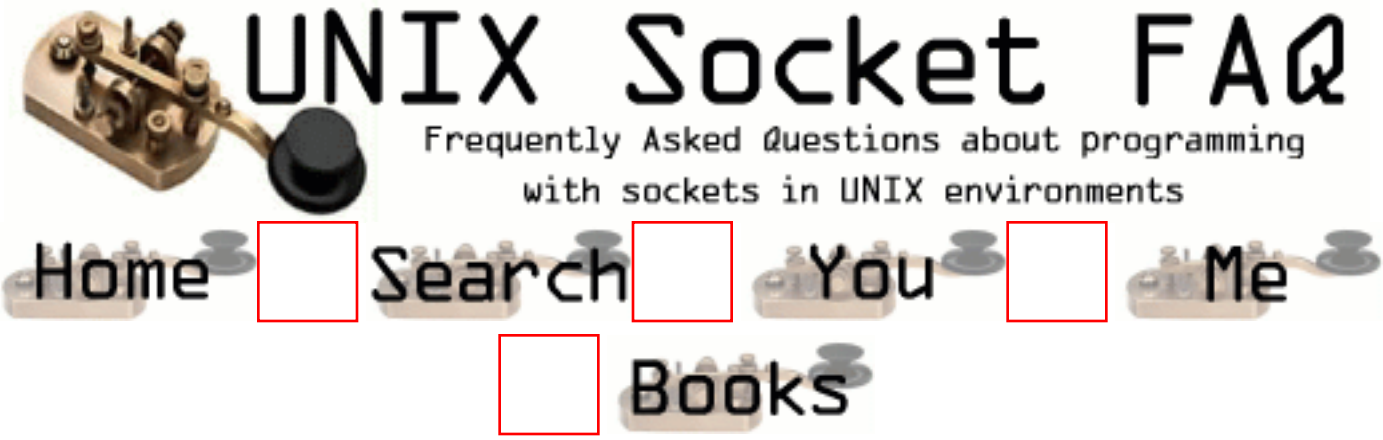
From: Sameer kamat

I want to know some kind of implementation detail of how I could have only a client client setup and have a commmunication between them. I want to one client to communicate with a server of another client and vice versa. Please clear this doubt !
-Sameer

From: Jay

Question is not clear ...
Want to communicate with the server of another client means what...

Add a Comment



New Questions: Autoflush

From: Rino Morin

I Have 2 Question.

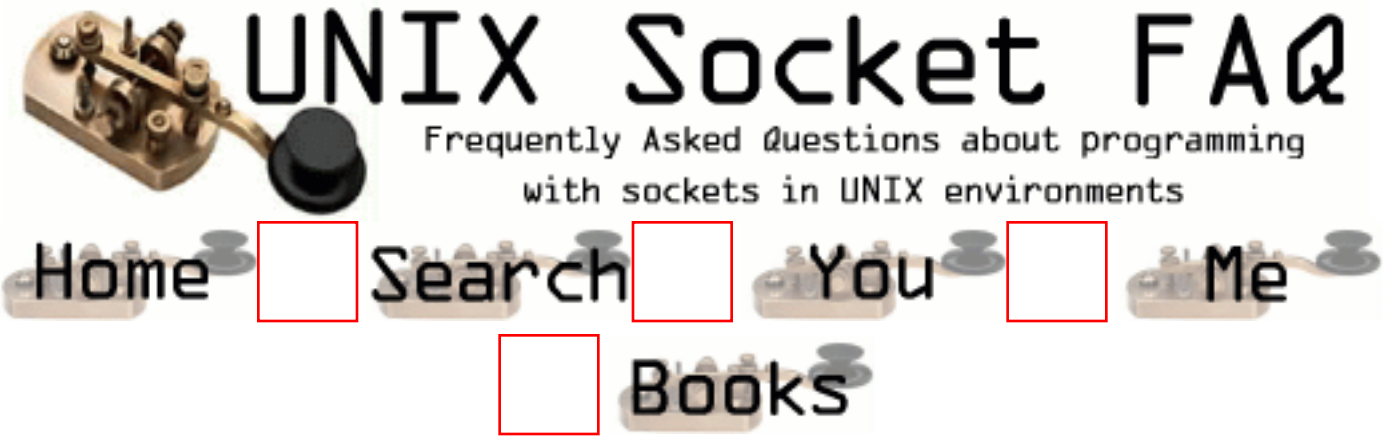
1. In perl how do you set the socet to be Blocking or (non polling).

2. In C. how do you do autoflush, to clear the buffer. OS/2 and Windows automatically flushes it, but in unix. ther has to be a way to flush the buffer.?

From: James Kassabian

1. Use fcntl().
2. Use setbuf(3S).

Add a Comment



New Questions: using loopback interface for testing socket programs

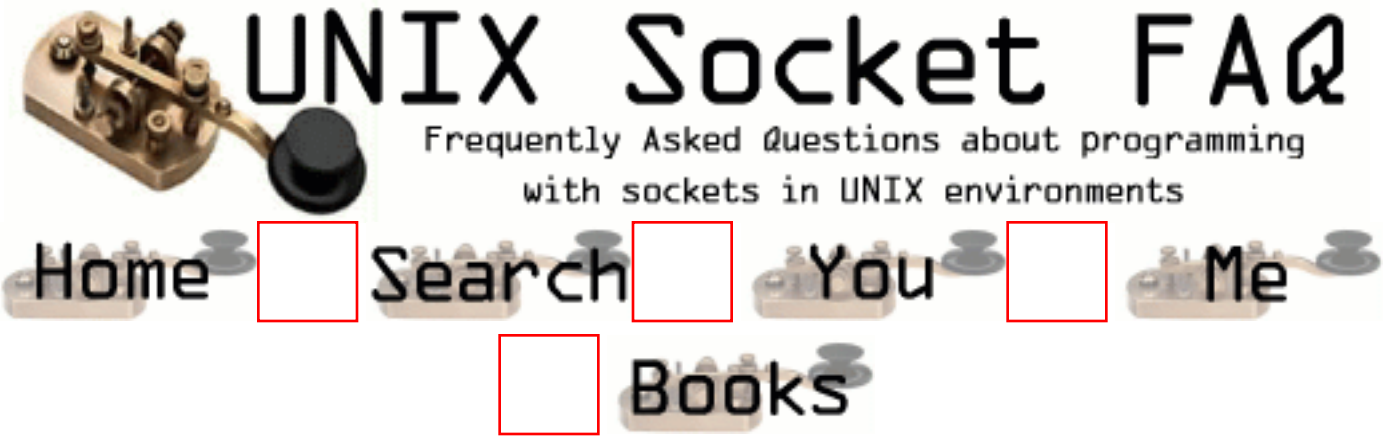
From: Anup Ochani

Can the loopback interface(127.0.0.1)
be used to test socket programs???

From: Nick

yup

Add a Comment



New Questions: Check readset and writeset at the same time

From: Winnie

i'm wondering whether i can check the readset and writeset at the same time (within a select()) call?

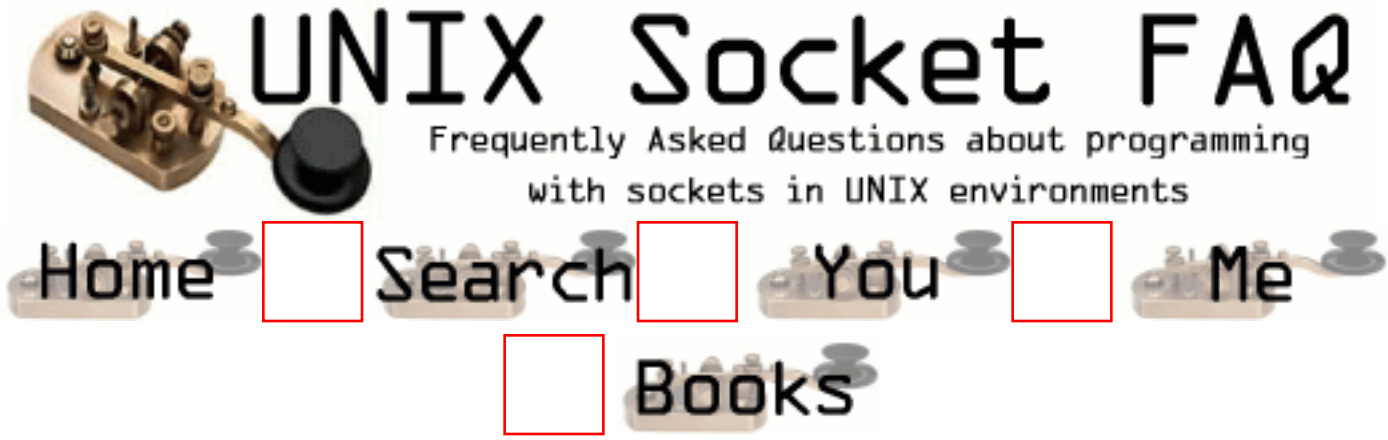
From: John Vincent

Yes you can.

From: gayathri

can you explain me how readset and writeset can be done at one time.can you give the eg code.

Add a Comment



New Questions: How to bind using struct sockaddr_ll..?

From: joelle teh

struct sockaddr_spkt is obselete in Kernel 2.2 and struct sockaddr_ll should be used.

but i'm facing problem in specifying the socket to bind to the particular NIC using this soscket address structure.

what is the value to be assigned to the element of this structure..?

how to get the macaddress of my PC to be copy into the structure..?

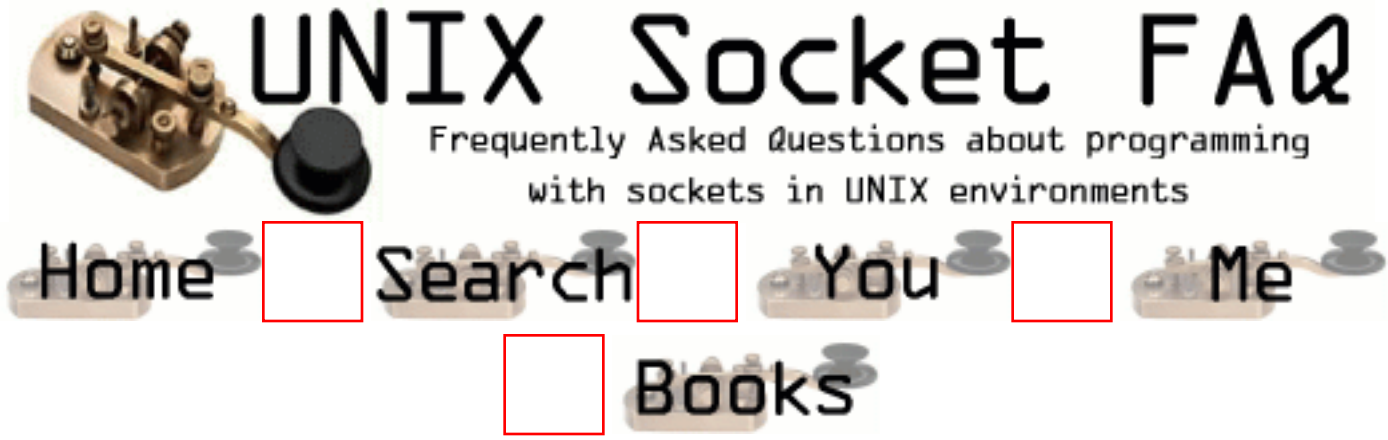
thank u..

From: Lars Westerhoff

The sockaddr_ll is not only used for binding but also for
returning informations about a received packed (recvfrom(2)).
For bind() you only need to fill sll_family (AF_PACKET),
sll_protocol (the link-layer protocol, e.g. ETH_P_IP)
and sll_ifindex. See packet(7) for details.

To get the interface index you can use the SIOCGIFINDEX
ioctl (see netdevice(7)).

Add a Comment



New Questions: Opening a socket on a random port number

From: Tom

Dear all,

Does anybody know how to bind a socket to a random port number (>1024), by letting the system assign a currently available port to your socket? I'll briefly explain why : it's an FTP client, and I need to create a listening (client-side) socket for the data connection, and then notify the server of the port number with the PORT FTP command.

Alternatively, is there an easy way to return the port number of a currently available socket (ie not just guessing one and testing it's availability)?

Thanks for your help,

Tom

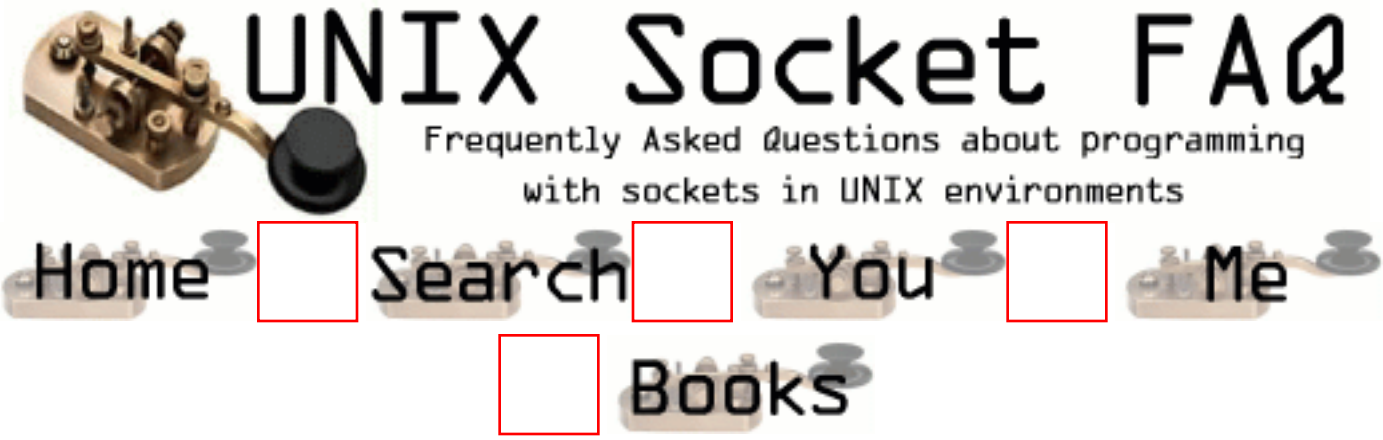
From: Richard Foletta

You tell the system to give you a port number by setting the SOCKADDR_IN.sin_port to 0. Then do the bind(). The bind() will give you a socket fd. use the getsockname() function to fill in a new SOCKADDR_IN structure. the sin_port member will now hold the value of the port that was given to by the system. Following is some sample code.

```
//----------------------------------------
// bind the queue_addr to the socket
//----------------------------------------
  rc = bind(queue_fd, (SOCKADDR_IN*) &queue_addr, sizeof(queue_addr));

  if ( rc<0 ) {
    error_it(__FILE__,__LINE__,"Could not bind to socket, errno=%d
%s\n",errno,strerror(errno));
```

```
    return Q_BINDERR;
    }
//----------------------------------------
// If the qname.port was set to zero then
// the system picked the port for us on the
// bind call above. use getsockname on the
// queue_fd to find out what the port is
// that the system gave us so we can make
// it available to the user of this object.
//----------------------------------------
  if(atoi(qname.port) == 0) {
    SOCKADDR_IN test_addr;
    int len = sizeof(test_addr);
    memset(&test_addr,0,len);
    getsockname(queue_fd,(SOCKADDRPTR) &test_addr,&len);
    ushort portnum = ntohs(test_addr.sin_port);
    sprintf(qname.port,"%d",portnum);
    }
```

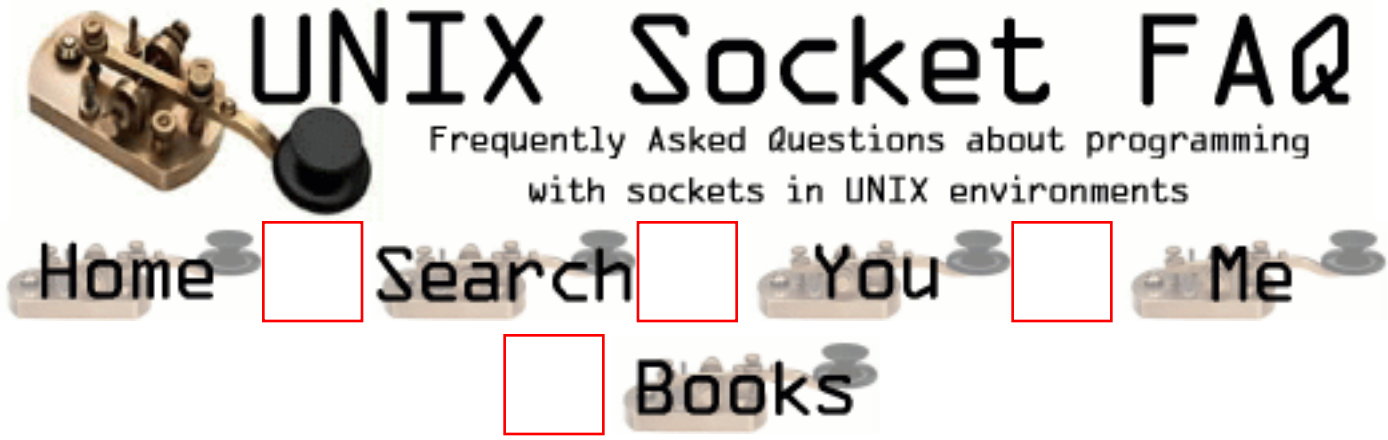
Add a Comment



New Questions: What is the meaning of the IDLE state in netstat command

From: JES

Add a Comment



New Questions: How can the client know, if the server has accepted the connection?

From: Virat Patel

As long as the server is listening , if the client make a connect - it succeds immediately. It is important for me that I know when the server has accepted the connection. One thing I can do is once I make a connect , I can do a blocking read and let the server send a message when it accepts the connection. But I am sure there can be another way, I believe TCP certain internal message to be send to the client - when the server does an accept.

appreciate every one's input in this case

thank you

From: Ahmed

Connect system call returns as successful only if the connection had been established between the two systems.. TCP handshaking messages are exchanged internally..

From: Warren Nash

Another way - not as efficient as other responses. Is on the server
side wait till accept completes and verify the client address details.
It is a paramter of accept() that gets filled in once accept returns.

From: acidblosoom #kumkum

i will uses wsaasyncselect() function to ask notify about completed connection.. for example :
after i use mysocket = connect();
i write wsaasyncselect(mysocket,mywinhandle,wm_getconnect(*where its my custom message),FD_CONNECT);

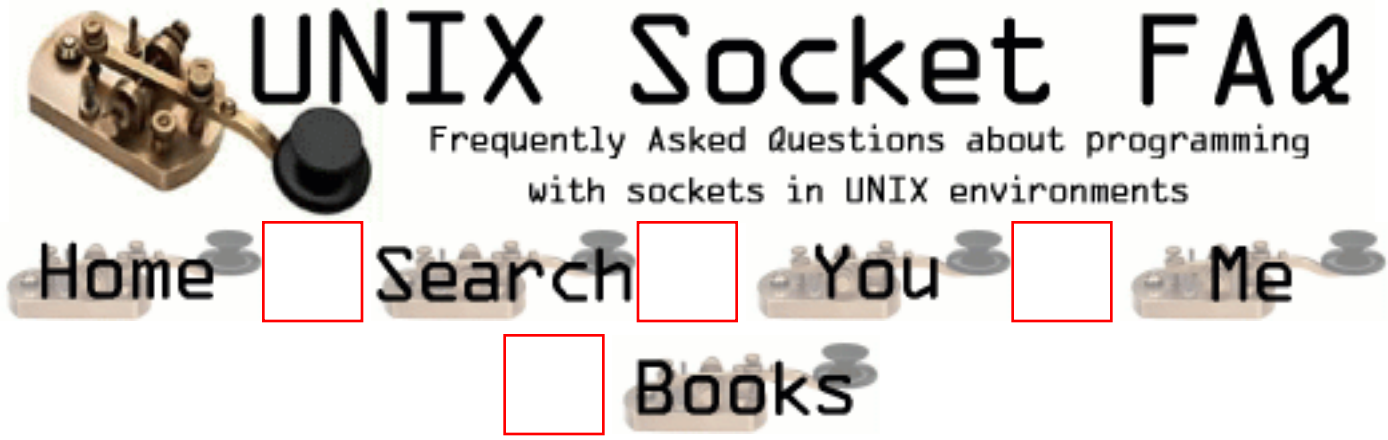
so thats after mysocket complete connection (or same as accepted by server) winsock will send

message wm_getconnect to my progie where i have to write what the process will do after.

From: Acidblosoom #kumkum

I beg ur pardon cauze what i explain above is uses in windows environment. try to use select() instead wsaasyncselect().

Add a Comment



New Questions: How can the client know, if the server has accepted the connection?

From: Virat Patel

As long as the server is listening , if the client make a connect - it succeds immediately. It is important for me that I know when the server has accepted the connection. One thing I can do is once I make a connect , I can do a blocking read and let the server send a message when it accepts the connection. But I am sure there can be another way, I believe TCP certain internal message to be send to the client - when the server does an accept.

appreciate every one's input in this case

thank you

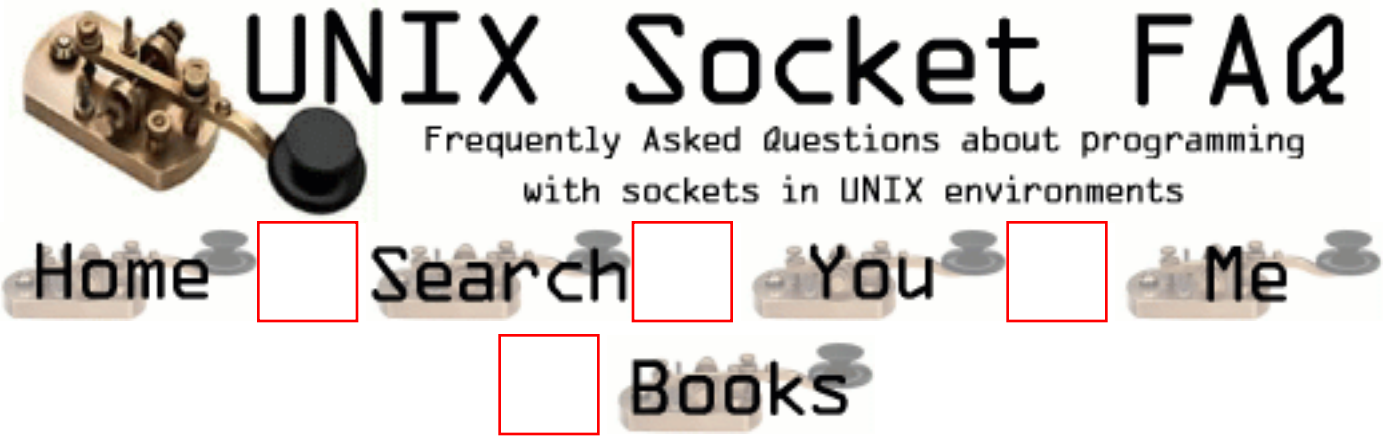
From: jeff dennison

1) open socket
2) use ioctl to set up non-blocking socket
3) set socket options SOL_SOCKET, SO_REUSEADDR
4) issues the connect() from within loop and check the errno returned by connect(). if connect returns 0 or connect return < 0 AND errno = EISCONN then break the loop
5) You are connected

From: jeff dennison

1) open socket
2) use ioctl to set up non-blocking socket
3) set socket options SOL_SOCKET, SO_REUSEADDR
4) issues the connect() from within loop and check the errno returned by connect(). if connect returns 0 or connect return < 0 AND errno = EISCONN then break the loop
5) You are connected

Add a Comment

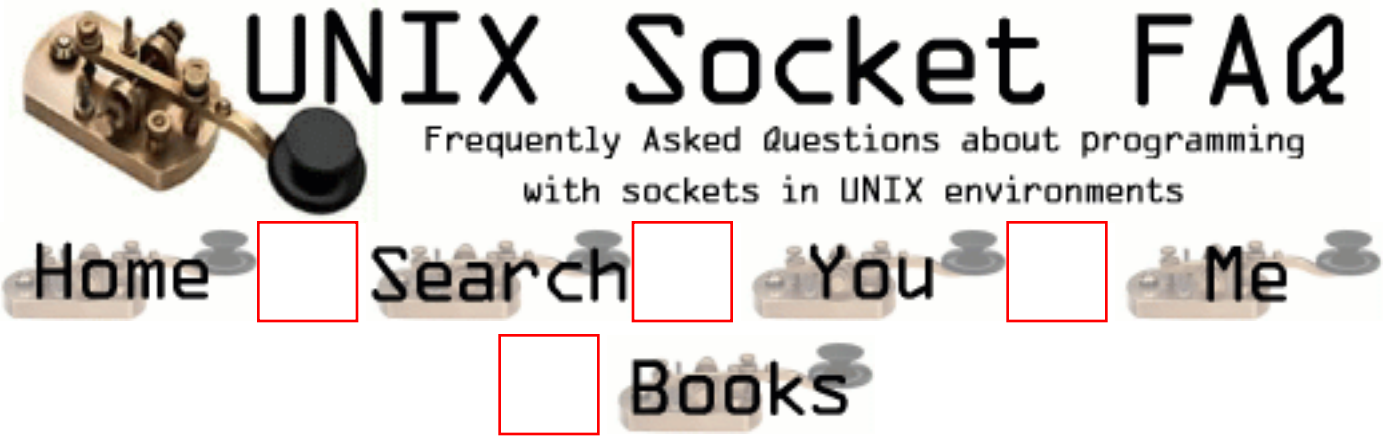


New Questions: Which error detection method to use?

From: Jonathan Rynd

Is there a standard way of determining when to use the errno variable, and when to use getsockopt() ?

Add a Comment



New Questions: concurrent server !

From: jaya wijethilake.

Hi,

This is the first time I am writing to you. This site is very
useful and good.

I am final year student of engineering faculty of university
of moratuwa. I am instrested in socket programming and have
reas comer's and steffen coffin's books.

I want to write a concurrent server which support { let say
three differet service connections between main server which I am
connected.That mean three socket links. To connect to main server
I can use a raw socket . My server must be concurrent server. lets say
one I got connection for one sevice I must keep that connection continuosly
unless I purposely stop the connection or link drops. Also
I should use my server raw socket for accepting the first connection to the service
and must fork() child process to handle multiple connections to that service. When should
I use accept() , fork() , select().

Also Do I have to use event table to keep track of all sockets and how I create
a event table. When to close child socket and master sockets?. And specialy my connetion
to main server must be allways up. So onece the first request accepted for a one serivce
that must be up forever { for all 3 services). Mulltiple connections for all three
services I must handle.

Plese what kind of server and how to do it.

best regards,

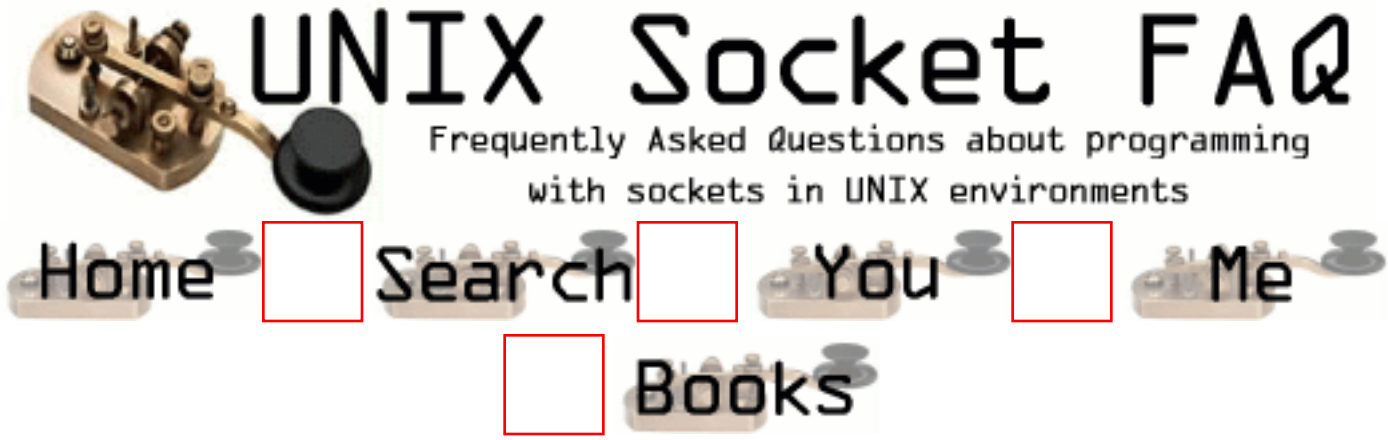
Jaya.

From: Warren Nash

Jaya,

Richard's book has everything you need - even the source code
is available at his web page.

Add a Comment



New Questions: Discarding unwanted socket data

From: Tom

Hi,

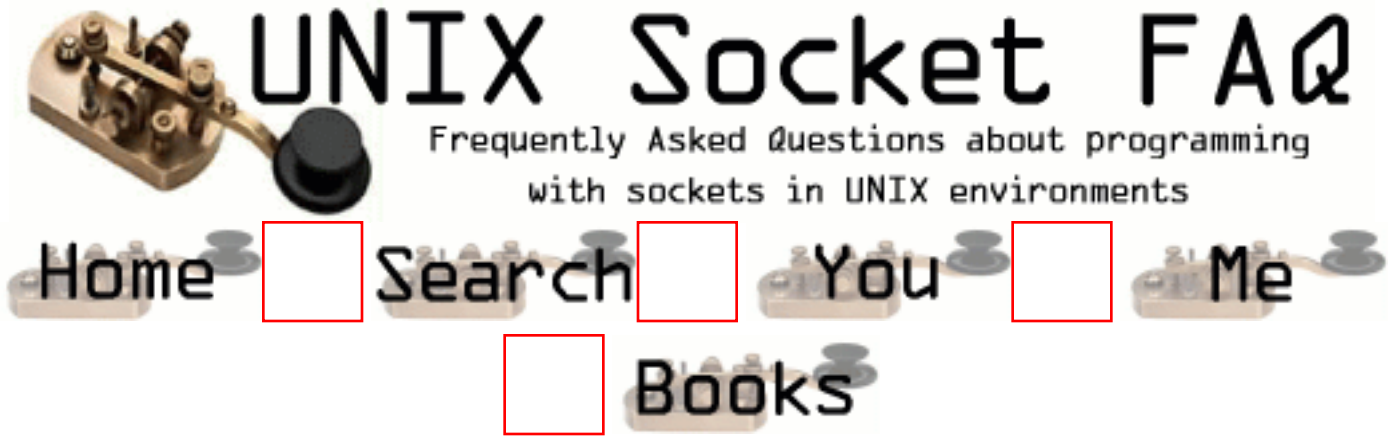
is there a way to throw away n bytes of data from the receive buffer of a socket, without RECVing it into a local buffer and then throwing the returned data away?

I would like to be able to retreive a single line of text from a socket at a time. I had planned to do this by RECVing say 2048 bytes of data with the MSG_PEEK flag, searching this data for the end-of-line character, calculating the number of bytes that the single line of text consumes, and then discarding that number of bytes from the socket recieve buffer. It seems ineffecient to RECV the data twice (first peeking and then reading), so is there a way to discard socket data?

Thanks for your help,

Tom

Add a Comment



New Questions: Options besides concurrent servers

From: Yishai

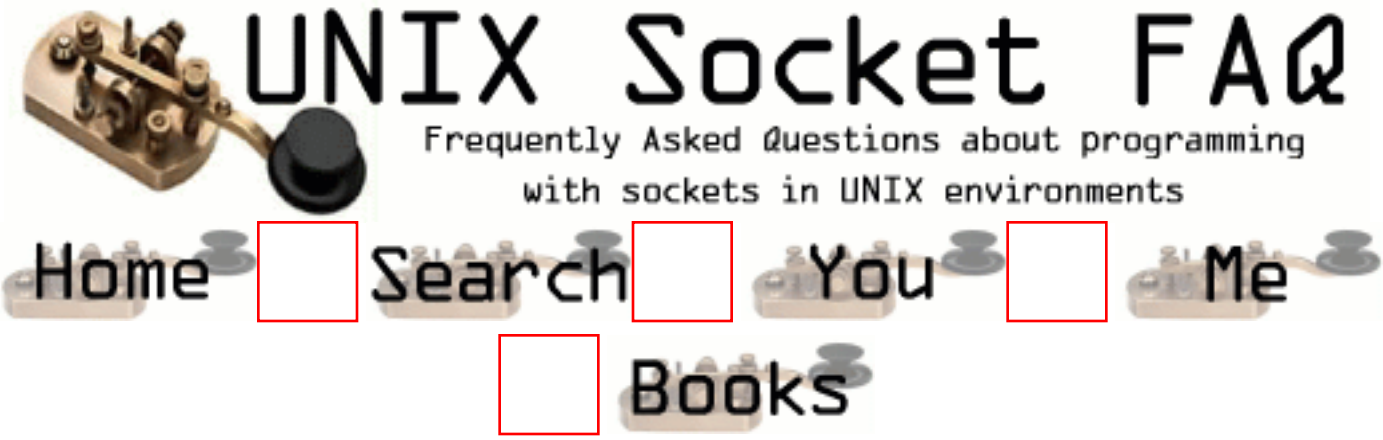
I'm writing a TCP server but instead of spawning off a child process for each connection that comes in, I just want to handle the request with any process from a pool of pre_created processes. I need to do this as the processes have a large memory overhead which would make a fork() prohibitvely expensive.

Anyone have any suggestions on how to do this.

From: Warren Nash

Use fork() or threads - pooling is also expensive, but you could use the select() function to block on a set of process id's or even use threads to minimise the use of memory. You will have to ensure that none of the pooled processes are being used by an exitsing application.

Add a Comment



New Questions: read write continuosly!!!

From: jaya

hi,

I want to write a server which connect to a switch through
a one socket (raw socket) and accept a connection . that is
this socket used only to accept a connection. then server must
create a nother process and this process must handle this connection
and it must read ,write cotinuosly.That mean it should not
close the socket unless purposely closed or link down. If
socket closed server must again able to get new connection from
raw socket.

Plese tell me how do it in concurrent way if possible or
iterative way. Non stop handling of connection (read,write)
is very important.

thanks in advance !!!!!!!!

jaya.

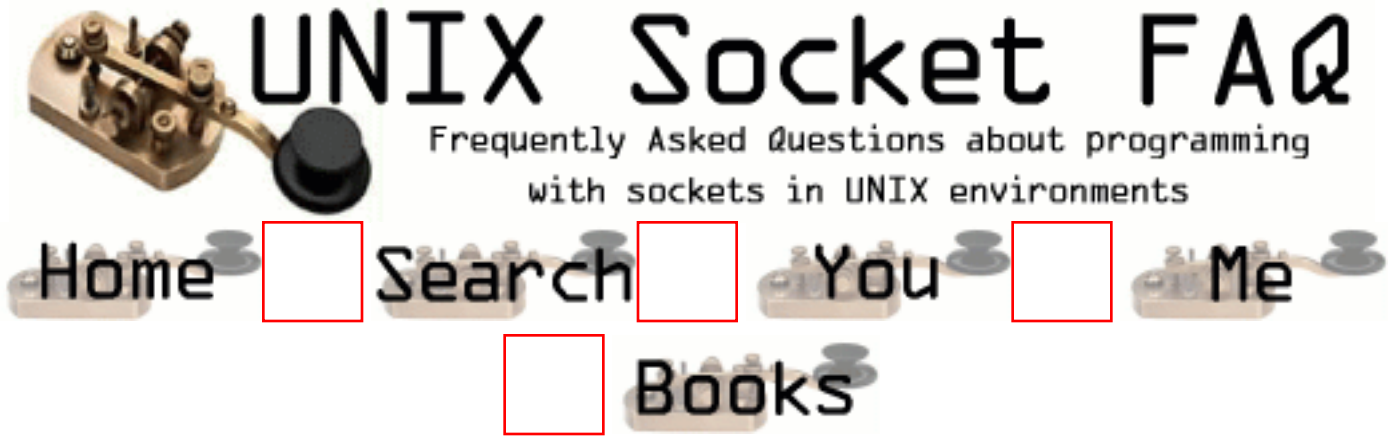
From: jagadeesh

send and receive some junk message to keep connection alive at certain interval of time.
Identify type of message and process it. If the message is not with respect to your requrements
discard and send and send some junk message. Do the othe end alos in the same way.
-jsh

From: manohar

u have to use non blocking read/write

Add a Comment



New Questions: Can I determine the number of pending connection request in the listen queue?

From: Umberto D'Agostini

I would prefer to increase the number of pre-forked children of socket based server, depending on the number of pending connection in the listen queue. There's a way to determine this number ? Thanks in advance.

Add a Comment

UNIX Socket FAQ

Frequently Asked Questions about programming with sockets in UNIX environments

New Questions: How can i get the IP address form name domain?

From: Mario Rugeles

How can i get the IP address form name domain?

ej.

char *cad = "www.yahoo.com"

IP = 204.71.202.160

Thank you very much.

From: Drew Vogel

I believe you're looking for gethostbyname(). Note however, that I have never been successful in understanding the hostent structure. Any in-depth explanation of the structure you find would help me greatly.

From: Rob Seace

Why not look at the question dedicated to this very issue,
right in this here FAQ:
"http://www.lcg.org/sock-faq/detail.php3?id=32"? ;-)

As I comment there, I think getaddrinfo() is conceptually a
better replacement for gethostbyname(), these days... But,
there is the issue of it perhaps not being available on all
systems, yet... Or, it not functioning quite right (I've
seen some somewhat annoying quirks with the Linux/glibc
version, for instance; it really likes to do a lot more DNS
queries than it should), since it's still a relatively new

function... But, ideally, getaddrinfo() is at least CONCEPTUALLY nicer than gethostbyname(), IMHO... And, one big advantage it has is that it's thread-safe, unlike old gethostbyname()... (Though, some systems may have a gethostbyname_r() version you can use, if you need a thread-safe version... That's what I eventually ended up resorting to, at least for now, under Linux for all my hostname lookups, just because I didn't like the odd behavior of the current getaddrinfo() implementation (which I reported on "bugzilla.redhat.com", BTW; still have heard nothing about it yet, though)...) Another big advantage is that it's built to handle both IPv6 and IPv4 nicely, whereas gethostbyname*() can also deal with IPv6 at an API level just fine, but really only wants to deal with one or the other at a time, based on resolver options, and is only capable of returning either one or the other on any given lookup (whereas, getaddrinfo() can theoretically return BOTH varieties for a single host lookup, if the host has both an IPv4 and IPv6 addr)... Basically, I just think getaddrinfo() is just much nicer/cleaner; but, unfortunately, it probably can't realistically be used quite yet on any widespread basis... So, gethostbyname*() will probably continue to be used for quite some time yet, unfortunately... I'd love to see it be replaced, though...

As for a discussion of the "hostent" struct, you local man page for gethostbyname() should have details on that... But, the main fields of importance are "h_name", the host name, "h_addr" (or "h_addr_list[0]"), the primary address (in raw, binary, network order format), "h_addrtype", the address family (eg: AF_INET or AF_INET6) of the address, and "h_length", the length of the raw address(es) returned... So, eg: if "h_addrtype" is AF_INET, then "h_addr_list[*]" contains pointers to "struct in_addr"s, and if it's AF_INET6 then they are pointers to "struct in6_addr"s...

From: Danny shaul

```
phostinfo = gethostbyname(query);
if (phostinfo!=NULL)
{
IP = inet_ntoa (*(struct in_addr *) *phostinfo->h_addr_list);
return IP;
}
```

From: Danny shaul

```
phostinfo = gethostbyname(query);
if (phostinfo!=NULL)
{
IP = inet_ntoa (*(struct in_addr *) *phostinfo->h_addr_list);
return IP;
}
```

From: Danny shaul

use this :

```
phostinfo = gethostbyname(query);
if (phostinfo!=NULL)
{
IP = inet_ntoa (*(struct in_addr *) *phostinfo->h_addr_list);
return IP;
}
```

From: Danny shaul

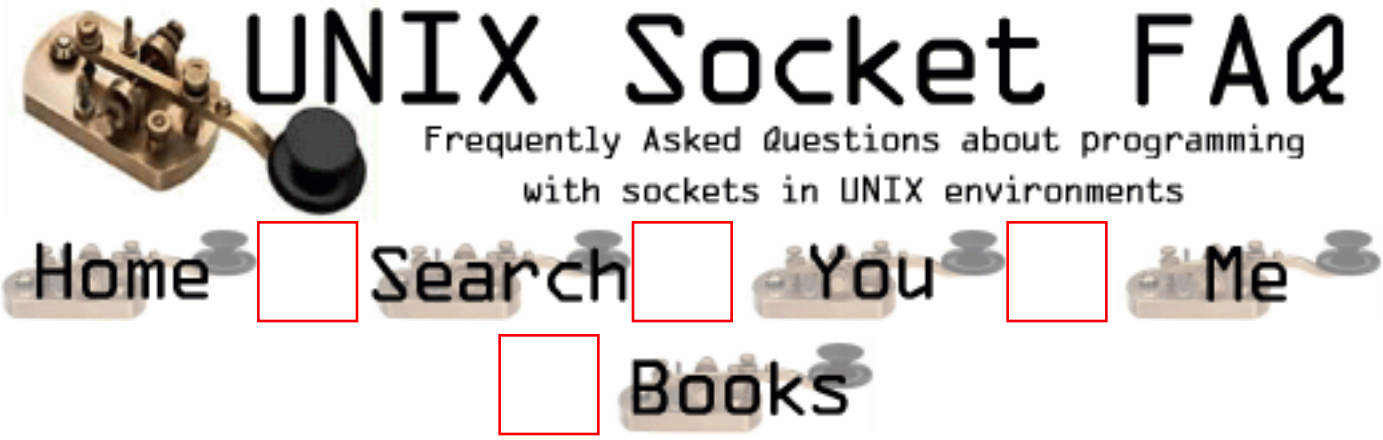
use this :

```
phostinfo = gethostbyname(query);
if (phostinfo!=NULL)
{
  IP = inet_ntoa (*(struct in_addr *) *phostinfo->h_addr_list);
  return IP;
}
```

From: sam

go to hell

Add a Comment



New Questions: How do I put my NIC in promiscuous mode?

From: Scott

Hello,

I am trying to write an application for automatic IP adress allocation and have been tring to find out how to place a NIC in promiscuous mode.

Thanks,

Scott

From: Rob Seace

It depends on your particular system... Get ahold of
libpcap ("ftp://ftp.ee.lbl.gov/libpcap.tar.Z"), and see
what it does for your system... But, eg., for Linux, you
can call ioctl() with SIOCSIFFLAGS, and OR-in IFF_PROMISC...

From: blorf

try using the ifconfig command.
#ifconfig eth0 +promisc
or something like that. Also simply use a program like
TCPDUMP or SNOOP that will set it into PROMISC mode.
See man ifconfig. Solaris is wacky about telling you about
promisc mode.

From: k.durga

sir,
we are doing project on network traffic monitory...
so we need to set nic of a host in a ethernet in
promiscuous mode... we have got NIC of D_LINK company...
please help us ... I will always be grateful to u..
thanking you
DURGA (INDIA)

From: Shreyas Ranade

We are doing Project Packet Sniffer in Linux.
We want to know how to put NIC in promiscous mode.
We also want to know about Sites and books that we could refer.

Thanking you,
Waiting for Your Reply...

Add a Comment

New Questions: What port do I broadcast messages to?

From: Steven

My server needs to broadcast a datagram (UDP) message to several clients. What port, on the HPUX machine, should i send to?

Thanks,

Steven

From: Hunter

Well, i'm not familiar with HPUX but the question should be:
"What ADDRESS should I broadcast messages to?"
I suppose that if you send a message (say ICMP ECHO) to the
broadcast address of your network, all machines will answer
to it....
the port should be the port your clients are listening on...
I hope i have helped....

Regards,
Hunter

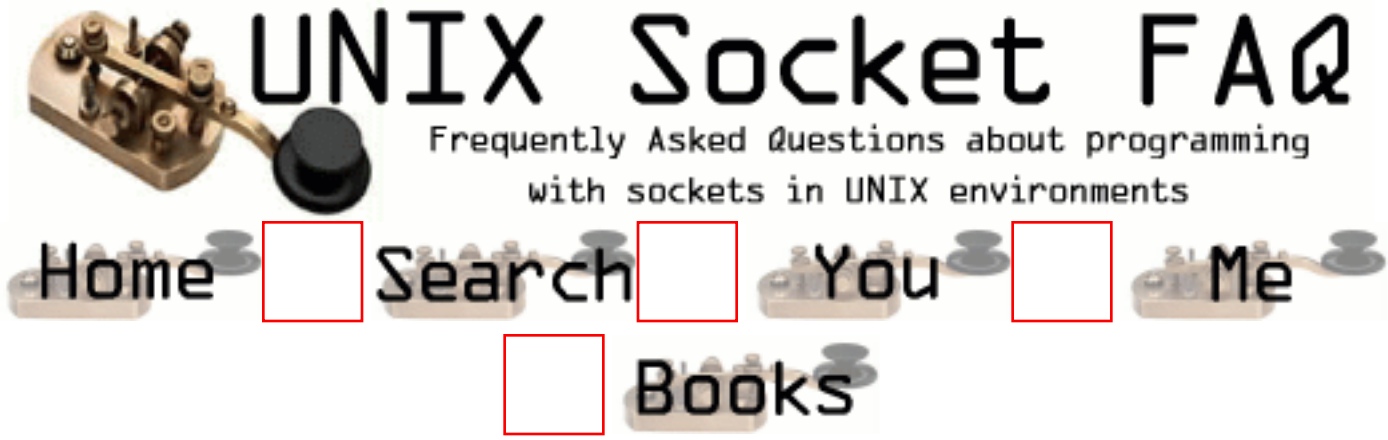
Add a Comment

New Questions: sendto/recvfrom errors on Solaris

From: Neil Milani

I decided to get the first and second volumes of Stephens TCP/IP series in hopes learning more about this stuff. I'm starting with example program in 1.5 and I'm already having problems. I'm using gcc on a Sun running Solaris 4.2 and both sendto() and recvfrom() are returning error values.
I compile and link but no real results - help!

Add a Comment



New Questions: Unix socket programming vs NT window socket programming

From: Patrick

I wrote a network game. The server is in Unix. And I tested that client could run in win98 and win95. However, when I
tested the client running in NT. The server seems that cannot bind the socket. Why? Pls. help me. Thanks a lot

---

From: Duda

maybe the programm written on win98 is not compatible on NT
...u know...things of libraries...

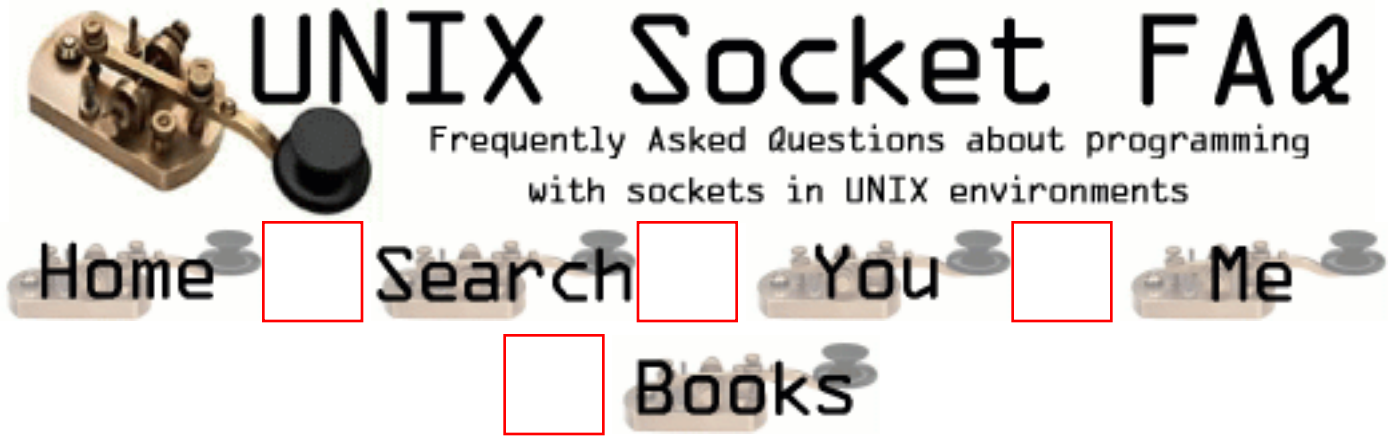
From: wilson

hi,the socket on windows are more complicated.for you will have to initilize the socket library.and use a type as SOCKET instead of int.

From:

1.What does Windows ahve to do with it, if the server (where bind() problem) run un UNIX ?
2. Check tcp number port translation (htons(...))

Add a Comment



New Questions: optimal solution

From: Eliahu

If I should build a server that must give a fast answer
to a client witch makes a lot of connections(descriptors)
what solution is an optimal ?(select,accept with multithreading,...)

From: Vic Metcalfe

You might want to consider UDP. TCP requires the three way handshake to set up each connection, so for quick messages between numerous hosts there is a lot of overhead.

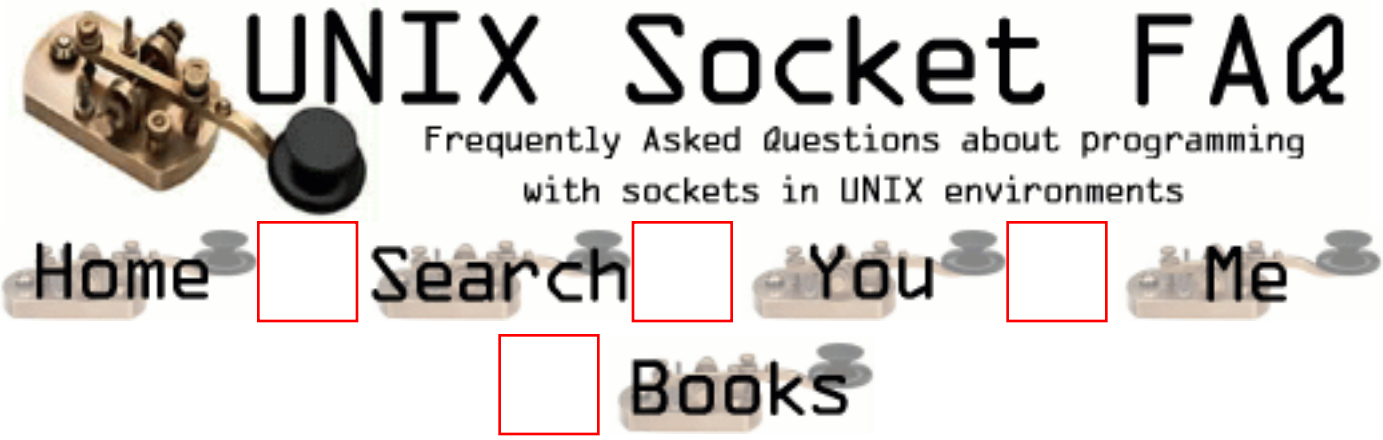
DNS for example uses UDP for most of its communications.

Of course once you get a TCP connection established it isn't so bad, and does take care of re-transmits, etc. It all depends on whether you want to keep all these connections alive or prefer to stay connectionless.

From: Jay

Use multithreaded program which is faster then the server who use forking. Bcoz creating a thread use less OS operation and u dont have to use much IPC.
So ur server can be fast.

Add a Comment



New Questions: Chat program in C/UNIX

From: EY

Where can I get the source code(client/server) for chat program written in
C/Unix ?

From: MANOJ KARANTH

are yaar ,
instead of putting the question here,why don't u find it your self
good luck,
;-)

From: Akintayo

There is a talk program Ytalk, whose source is available

From: Shawn Elliott

Take a look at Jabber
http://jabber.org
it's an open source program so you might be able to get in on the developement after you played with it awhile
:*

From: zorro

Can any one tell me where can i find the source code for the chat program written in c on unix

---

From: sudhakar

how can i get the source code for chat program using socket programming

---

From: charlie

where can i get a souce code for Internet browser on UNIX environment?

From: kiran

where do get the chat program eithr in C or UNIX

From: Mahafuz

All these programs are inclided in linux (standard package)with codes. so pls look in linux for these codes. You'll get a lot of them.

From: lost

If u want source code to a simple client\server chat program then email me...ill be glad to send u the one i made

---

From: Arindam Malakar

Hi,

Please send me the source code for the chat program and
also the source code for file transfer through ftp if
these are avilable with you.

I am awaiting for your earliest reply.

With Regards,

Arindam Malakar

From: Andy

If you are wondering about chat programs called 'talkers' there there's a variaty of sources for them, depending on the type you want. Here are a few places to check out to get source code:

http://amnuts.talker.com/
http://pgplus.ewtoo.org/
http://asteroid-b612.org/ncohafmuta/index.shtml
http://www.ogham.demon.co.uk/nuts.html

You can also find a bunch more on:

http://download.ewtoo.org/
http://www.talker.com/libtalkercodes.html

Andy

From: Banjo

how can i get the source code for chat program using socket programming

---

From: tito

1cplusplusstreet.com search for chat

From: gauravchopra

From: Dan

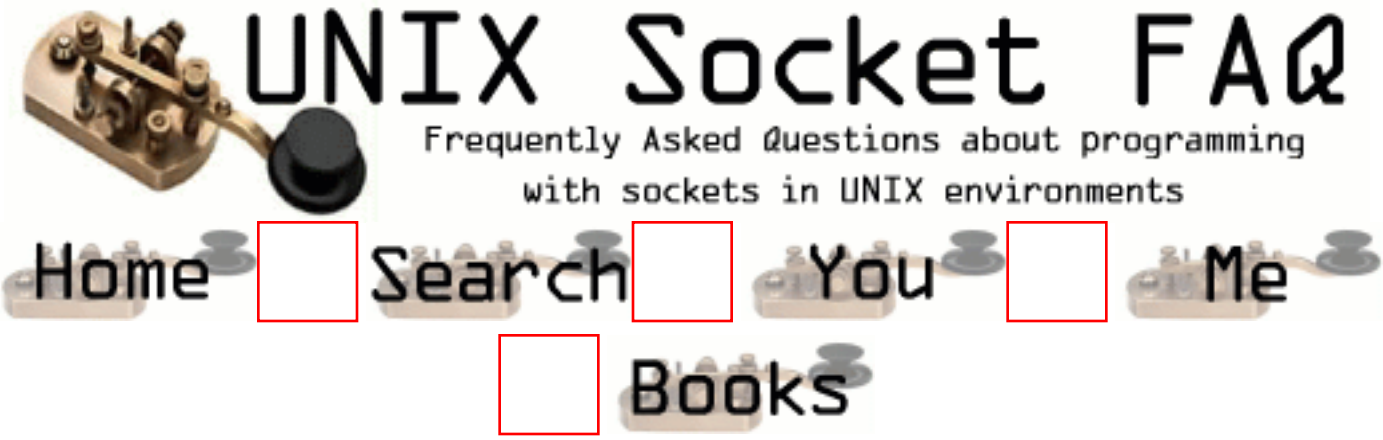
Come on people.. Look around some.. look in the code examples, perhaps, in the main part of the faq??

It's all right there under yer noses..

From: Aaron Sethman

Well if you are looking for a irc server, hybrid(which is used on a good portion of efnet servers) is at http://www.ircd-hybrid.org/

Add a Comment



New Questions: how to create second socket on my client application

From: kurumlu aliye

I have two client applications to communicate one unique Server. For the first application I establish a session by using socket() and connect()functions. But for the second conncetion when I do the same thing for my second connection, Server refused my connection. I guess reason for it is to use same socket Id for both connections. My question is how I can create another socket in my second client application

---

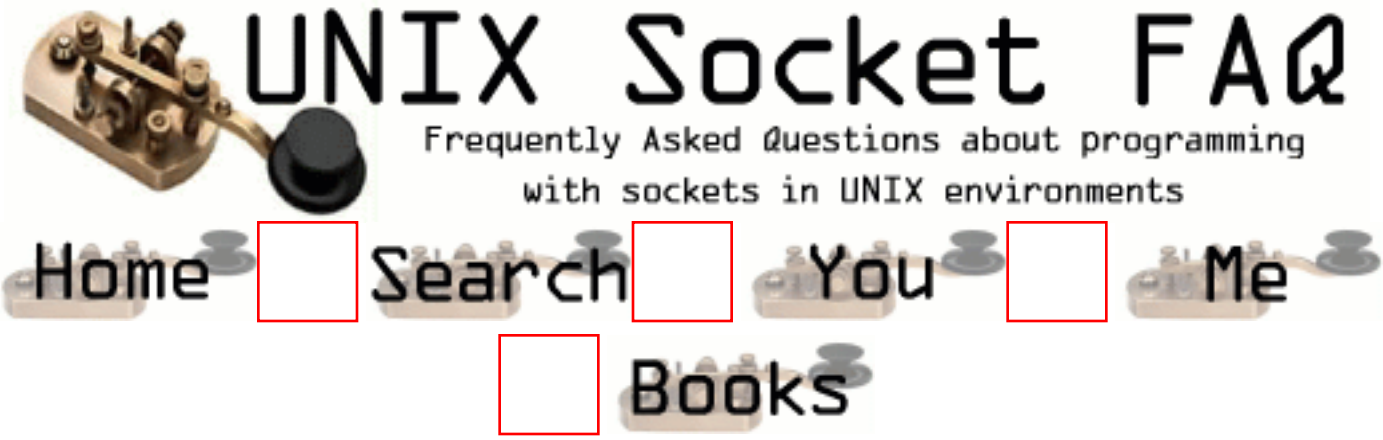
From: Warren Nash

do socket() and connect() again - but best to make it dynamic
and not static. Remember server will need to be told what
port, address and socket to talk to on - accept() in the
server.

From: jagadeesh

Hello,
create one more process and again call connect and send and recv function.
bye
jsh

Add a Comment



New Questions: how many simultaneous socket connections can a concurrent server accept?

From: Jaeyoun Chung

Hello,
I'm writing a TCP concurrent server which should accept from many many clients and keep the connections -- from time to time, it sends back some informations to the client. I used select call in the while loop body so that it can service concurrently. But the accept() call fail when there's about 60 client connections there. Should I fork a new server and listen() and accept() again after closing all the descriptors that have been opened?

From: Xidong Wang

hmm, each process has its own upper limit of opened file decriptor, maybe 60 is your program' limit.
bye the way, maybe you can check the retrun value of accept and errno value, they can tell you more about the situation.
-wxd

From: AK

Call setrlimit() in your main function to increase the number
of open file descriptors that your server process can have.
Thro' this system call, you can increase the limit upto 1024.

From: Michael H. Ward

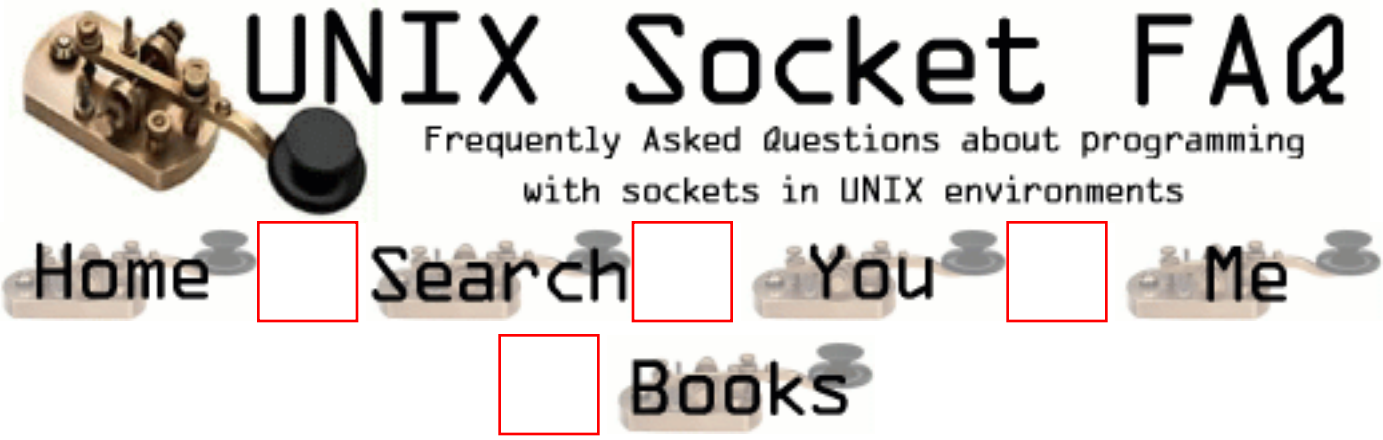
Look at the value of SOMAXCONN defined in <sys/socket.h>.

Typicaly it is 5 on SunOs and LynxOs and 128 for Linux.

From: Ville Koskela

How would you get more than 1024 connections? Would you need to use multiple processes, or could multi-threading work?

Add a Comment



New Questions: Sending integer values to sockets.

From: skuzzlebutt

I am having problems sending integers to sockets.
I am using HP-UX 10.20. Which tells me host and network byte order are the same.
Yet I keep getting garbage when I recv for integers. I have tried using the hto...
routines but it still doesn't work. Can anyone help ?

Skuz.

From: Rob Seace

Can you post some of your malfunctioning code? Because, I
see no reason why you shouldn't just be able to do something
on one end like:

```
int i = 42;
write (sockfd, &i, sizeof (int));
```

And, then on the other end:

```
int i;
read (sockfd, &i, sizeof (int));
```

As long as both ends are on the same architecture (or, at
least share a common byte ordering), it should work fine...
In fact, I often do such things (eg: to send the size of an
upcoming fixed-sized message to be read, so the other end
will know exactly how much to read at once)... And, I've
never seen any problems...

Of course, the 'correct' way of doing it would be to use htonl()/htons() on the sending end, and ntohl()/ntohs() on the receiving end, to allow platforms with different byte orders to communicate as well... Or, better yet, conver the number to a text string, and send that, eliminating the whole mess of dealing with transfering raw binary data... But, for dealing with two ends known to be on the same platform, I often use this approach, and see nothing wrong with it... *shrug*

From: skuzzlebutt

Thanks. I tried your method and it works fine, however I am only testing the code on the same machine and I will be looking to using different machines.

My code looks something like this

SENDING

```
int id;
id = 10;
if ((send(sockfd, id, sizeof(int), 0)) == -1)
   {
   /* error handling */
   }
```

RECEIVING

```
int id;
if ((recv(sockfd, id, sizeof(int),0)) == -1)
   {
    /* error handling */
   }
else
   {
    printf ("The recv'd id is %d \n", id);
   }
```

I have tried both htonl/ntohl & htons/ntohs
but I never receive anything like the data I am expecting.

Skuz.

From: skuzzlebutt

Cheers Rob, I've got it sorted, your example put me in the right direction.

Skuz.

From: Rob Seace

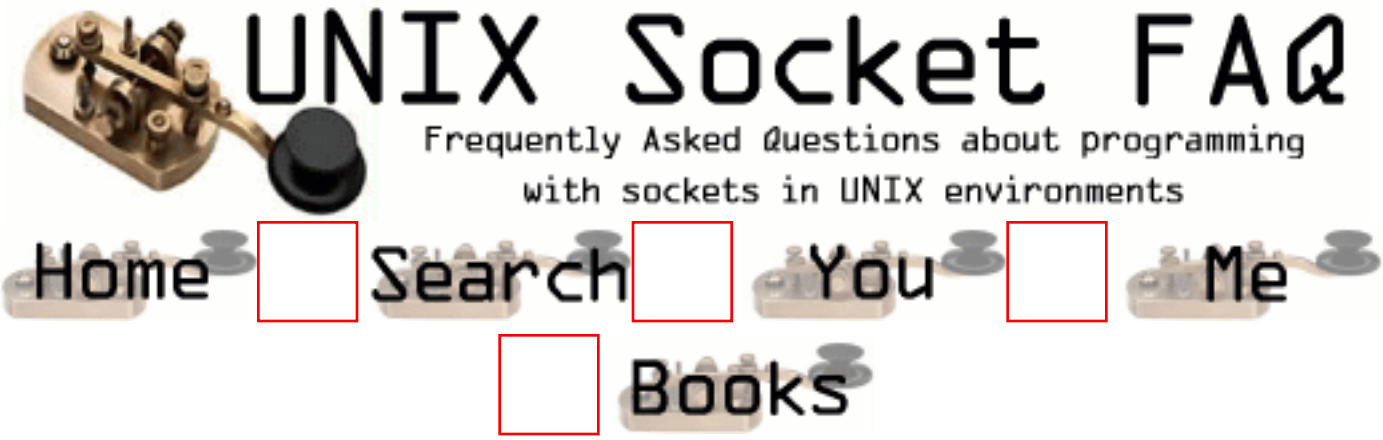
I assume you found the error in the send()/recv() based code you posted too, then? Unless it was merely a typo here, the main problem looked to be the fact that you were passing the actual int ("id") rather than its address ("&id") to the send()/recv() calls... So, I would've expected such code to either seg-fault or fail with EFAULT or EINVAL or something similar... But, maybe it just happened to not do so, for some reason (I'm not too familiar with HP-UX, and how it tends to deal with invalid memory references in programs), and so the value you were printing out was just some random value off the stack (which also was the address of wherever the value really got stored; though, the value that gets actually sent would similarly be random crap from whatever was in address "10")...

But, if it was merely a typo in the code here, and you really were passing "&id" correctly, I see no reason why that send()/recv() approach shouldn't work, either; it's just that I personally generally prefer to stick with the old write()/read() approach, myself... But, it should still WORK just fine either way, I think...

From: skuzzlebutt

Yep, found the error.
Cheers.

Add a Comment



New Questions: can u answer this ??

From: ganesh

Can we write a C-program which writes to a file for which all the permission bits are disabled ?

From: Wang Xidong

as a super user, you can. or not

From: Mourad

As a SU, you turn on the write bit, write to the file, then turn it back off again

---

From: S.Krishna

Are you super user or normal user ? it depends on that.

Add a Comment

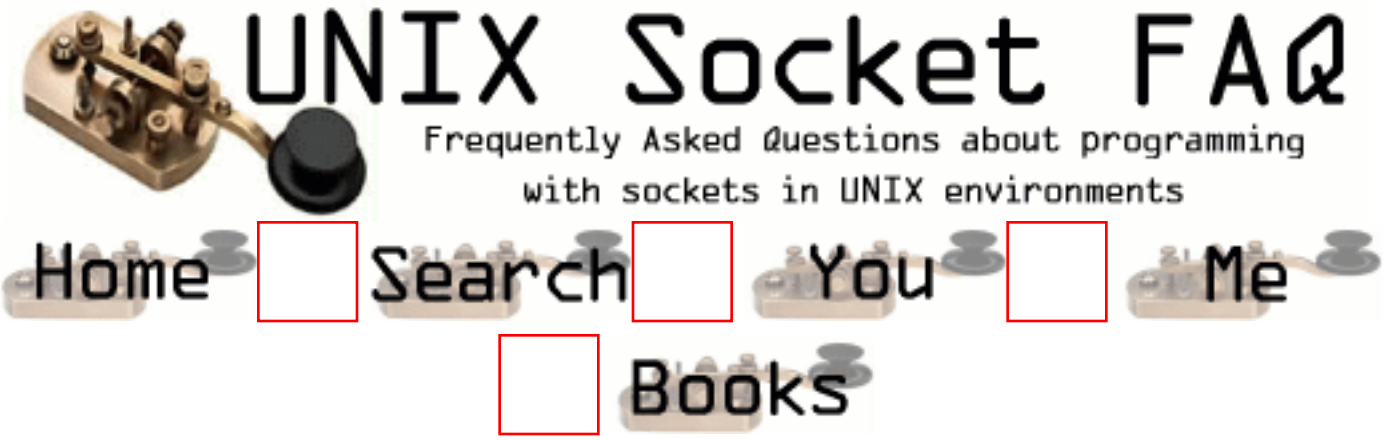
UNIX Socket FAQ

Frequently Asked Questions about programming with sockets in UNIX environments

New Questions: C to unix base by win sock

From: yeow choon hong

Add a Comment

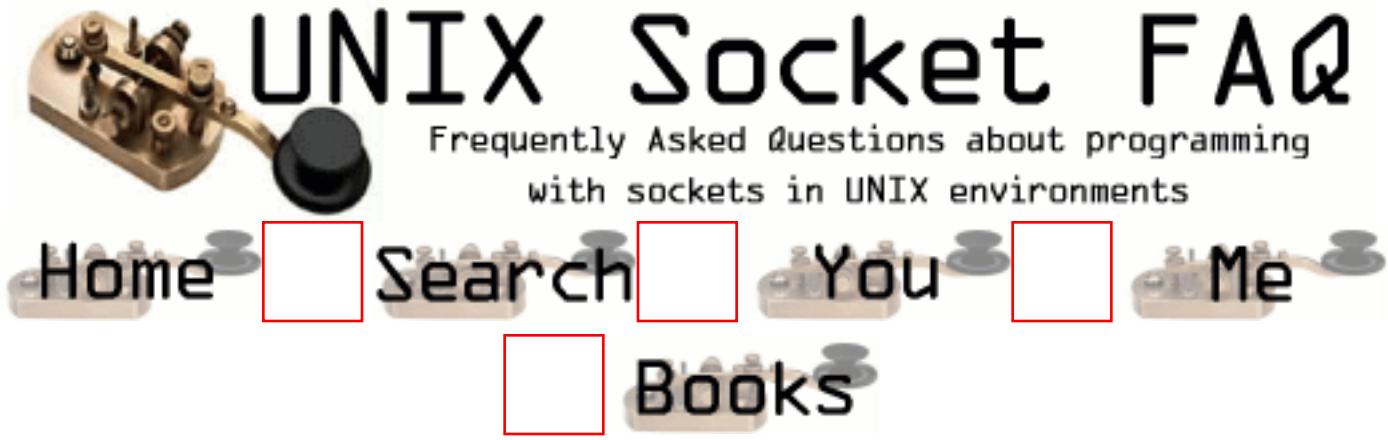


New Questions: C to unix base by win sock

From: yeow choon hong

how to send a character from c++ language to unix by winsock?
the server winsock program will be embedded into another program to let it listen to the char sent continuously.

Add a Comment



New Questions: Multiple blocking on select.

From: chris orthner

How do I use select() within a server application to comunicate to multiple sockets within a looping structure?
It seems after the first itteration the select() call no longer blocks, it just returns -1. It seems to be an issue with the read, write and error bitmasks and the FD_SET/FD_ZERO/etc. macros.

Obviously it can be done, I'm just not sure how to use the macros properly.

Can anyone give a quick sample of select() used within a loop?

From: skuzzlebutt

Zero all the fdsets before the select.
Add your socket descriptor to the exception set and
the read and/or write sets using FD_SET.
You can then check if your socket has had an exception or
is ready for reading/writing using FD_ISSET.
Select amends the fdsets depending on what events have occured.
If a socket is ready to read it will set the appropriate bit
in the fdset for that socket, similarly for writes and exceptions.
I've only started with sockets so this isn't a great explanation, but
I hope it helps.

```
while(1){
        FD_ZERO(&readfd);
        FD_ZERO(&writefd);
        FD_ZERO(&exceptfd);
        FD_SET(listenSock, &readfd);
        FD_SET(listenSock, &exceptfd);

        irc = select(MAX_SOCKETS, &readfd, &writefd, &exceptfd, NULL);
```

```
        if (FD_ISSET(listenSock, &exceptfd) != IZERO)
           {
            /* exception occured deal with it */
           }
        if (FD_ISSET(listenSock, &readfd) != IZERO)
           {
            /* Socket is ready to read */
            /* Deal with it. */
           }

}
```

From: skuzzlebutt

For multiple sockets, you need an array of descriptors to use
and loop around these doing the FD_ZERO, FD_SET and FD_ISSET
macros, I think. The approach I have taken is to pre-fork
children for each socket and use each child for a select
on a single socket.

From: Rob Seace

Well, that (only select()'ing on a single FD at a time) kind
of defeats the whole point of select(): to multiplex I/O
between a whole bunch of different FDs... ;-)

As for the required "array of descriptors", you've already
got a nice datatype designed exactly for that purpose, and
which you're already using if you're using select() at all:
"fd_set"... Simply keep a fixed fd_set of the FDs which you
are interested in outside the main loop, and before every
single select() call, copy that whole fd_set into another
temp fd_set to pass to select()... That was likely the
problem that the original poster was having: he was failing
to reset his fd_set's at the top of the loop, prior to the
select() call, failing to realize that select() modifies
them... So, your suggested loop code fragment should be
almost exactly the sort of thing he'll want to do (except,
if dealing with multiple FDs at a time, rather than go
through the FD_ZERO()/FD_SET() calls you show, he can just
memcpy() from an external, fixed fd_set into temp ones)...

From: Pier

same problem here
how to copy the mem?

```
 FD_ZERO(&readfds);
 FD_SET(STDIN, &readfds);
 FD_SET(fd, &readfds);
while(1) {
memcpy(&rfdstmp, &readfds, sizeof(readfds));

... }
```
doesnt do the trick, what's the size i got to use?
it blocks once, but not twice ...

greets
Pieter

From: Rob Seace

Hmmmm... What you post looks about right to me, so I don't know why it wouldn't be working for you... I usually use "sizeof (fd_set)", but sizeof() one of the vars that is an "fd_set" should be the same thing... *shrug* Another thing you can try, if you have any sort of decent compiler that supports structure copying via assignment, is to just replace the memcpy() with "readfdstmp = readfds;"...

Hmmmm... Wait, I notice you don't actually show your select() call itself, in your snippet of code... Are you sure you're passing in "&readfdstmp" to select(), rather than "&readfds"?? Basically, you'll want code something like this:

```
fd_set permfds, tmpfds;
int maxfd, fd, newfd, cnt;

FD_ZERO (&permfds);
FD_SET (listenfd, &permfds);
maxfd = listenfd + 1;

for (;;) {
    tmpfds = permfds;
    /* or memcpy (&tmpfds, &permfds, sizeof (fd_set)); */
    cnt = select (maxfd, &tmpfds, NULL, NULL, NULL);
    if (cnt > 0) {
        for (fd = 0; fd < maxfd; fd++) {
            if (FD_ISSET (fd, &tmpfds)) {
                if (fd == listenfd) {
                    /* it's an incoming new client */
```

```
                newfd = accept (listenfd, NULL, NULL);
                if (newfd < 0) {
                   /* error; die or whatever */
                } else {
                   FD_SET (newfd, &permfds);
                   if (newfd >= maxfd)
                      maxfd = newfd + 1;
                }
             } else {
                /* it's a client connection; read() */
                /* from it, or write() to it */
             }
          }
       }
    }
}
```

From: Pier

i tried your suggestions, but it didnt work. i included a larger snippet of code. I did use &rdfstmp (i changed it to tmpfds). I dont have a clue, btw: im running this on a 2.4.0-test1 i686 linux machine, could that be the problem?
kind regards
Pieter

```
  FD_ZERO(&readfds);
  FD_SET(fd, &readfds);
  FD_SET(STDIN, &readfds);

  while(1) {
        // memcpy(&tmpfds, &readfds, sizeof(fd_set));
        tmpfds = readfds; //same result as with memcpy
        tv.tv_sec = 2;
        tv.tv_usec = 500000;

         /* don't care about writefds and exceptfds: */
         if (select(fd+1, &tmpfds, NULL, NULL, &tv) <0) {
           printf("ERROR select\n"); }
         else {
           if (FD_ISSET(STDIN, &tmpfds)) {
              printf("A key was pressed!\n");
              // if i pressed the enter he keeps printing "A key ..."
              // like below, he keeps entering this if part, but he may not !! how comes?
           }
           else if (FD_ISSET(fd, &tmpfds)) {
```

```
            re = recv(fd,buff, bufflen,0);
            printf("%d:%s", re, buff);
            // when there is no data he prints: 0:buffer from last read
            // but he may never enter this if part if there is no data ! ????
          }
          else
            printf("Timed out.\n");
        }
    }
```

From: Rob Seace

I think I see your problem... You do an initial check for
a select() return of LESS THAN 0, but you don't check for
an actual return of 0 itself... Which will happen if you
timeout... And, you ARE using a timeout value, so that is
likely what is happening... And, when a timeout occurs, the
value of your fd_set can NOT be relied upon, so the fact
that the FD_ISSET() test succeeds on it is not all that
surprising...

From: Pier

hi
now i check for a zero return value, but it still doesnt work cause it returns 1, but there is no data (the read returns 0). The select also doesnt wait the hole 2.5 sec, so it doesn't time out.

i start the program, wait a few seconds, then press enter and in 1 sec i get thousends of messages below (cause its in a loop)
this is my output:

UIT: 0
UIT: 0
UIT: 0

UIT: 1
A key was pressed!
UIT: 1
A key was pressed!
UIT: 1
A key was pressed!

code snippet:

```
 uit=select(fd+1, &tmpfds, NULL, NULL, &tv);
```

```
 printf("UIT: %d\n", uit);
   if (uit < 0) {
      printf("ERROR select\n"); }
   else if (uit = 0) {
      printf("Timed out\n");
   }
   else {
      if (FD_ISSET(STDIN, &tmpfds)) {
         printf("A key was pressed!\n");
      }
      else if (FD_ISSET(fd, &tmpfds)) {
      re = recv(fd,buff, bufflen,0);
      printf("READ:%d:%s", re, buff);
   }
 }
```

greets
Pieter

From: Rob Seace

Ah, well, because you're never actually READING the data
from stdin, are you?? So, of course, it remains readable,
because the data is still there to read...

From: pier

but im having the same problem with fd (which i do read)
and now i read STDIN, but i have exactly the same problem.
i replaced recv by read and now it works for STDIN but not for fd, the select still returns 1 while no data is read.

code:

```
uit=select(fd+1, &tmpfds, NULL, NULL, &tv);
printf("UIT: %d\n", uit);
   if (uit < 0) {
      printf("ERROR select\n"); }
   else if (uit == 0) {
      printf("Timed out\n");
   }
   else {
      if (FD_ISSET(STDIN, &tmpfds)) {
```

```
        re = read(STDIN,buff, bufflen);
        printf("A key was pressed:%s\n",buff);
      }
      else if (FD_ISSET(fd, &tmpfds)) {
        re = read(fd,buff, bufflen);
        printf("READ:%d:%s", re, buff);
      }
    }
```

output:

```
UIT: 1
READ:0:Login:...
UIT: 1
READ:0:Login:...
```

uit = return of select
READ = number of bytes read from fd (which shouldnt happen because theres nothing to read)

regards
Pier

From: Rob Seace

Well, if read() on a socket returns 0, that means the other end has closed the socket... It's perfectly valid for the socket to select() as readable in such a case, because what is there to read() is the EOF (ie: the FIN packet from the remote end)... And, when you get a read() return of 0, you can't continue trying to read from socket any more, because nothing more will ever be coming... Your only option is to close() it, really... (Technically, in some cases, it may be valid to just shutdown() the read half, and continue writing data, if the remote end is cooperating the same way... But, in general, the remote probably just close()'d it itself, rather than just doing a half-close...) So, once you get the 0 return, you should "close (fd);", then remove it from the active fd_sets...

Now, if you don't think the socket should be closing, that's a different issue... Because, I'm pretty sure it IS, for some reason...

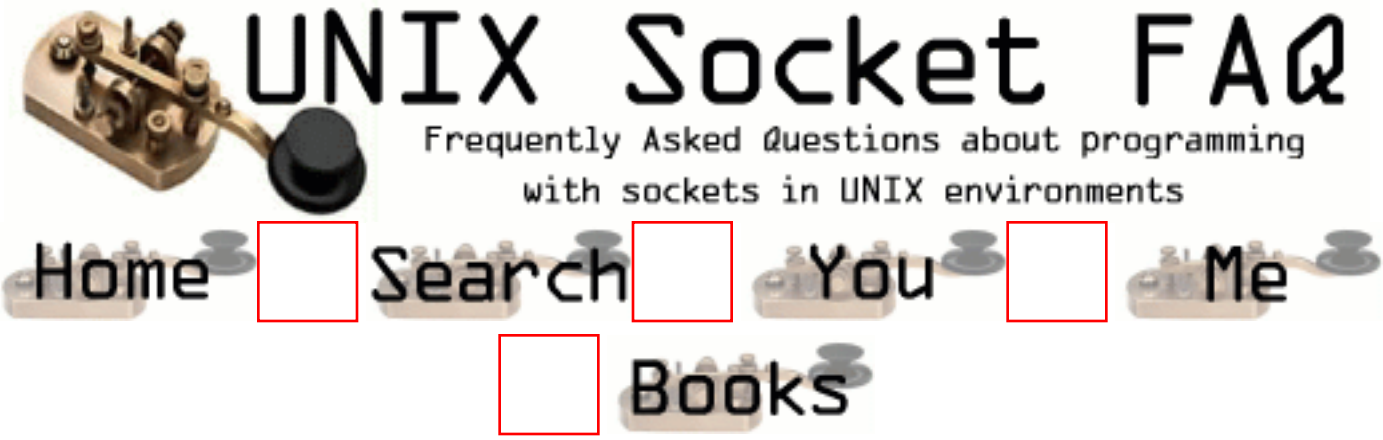
From: pier

i tested it with fingerd (127.0.0.1 79) so i guess its closed by fingerd.

Thanks a lot for your help.

regards
Pieter

Add a Comment



New Questions: multiple connections

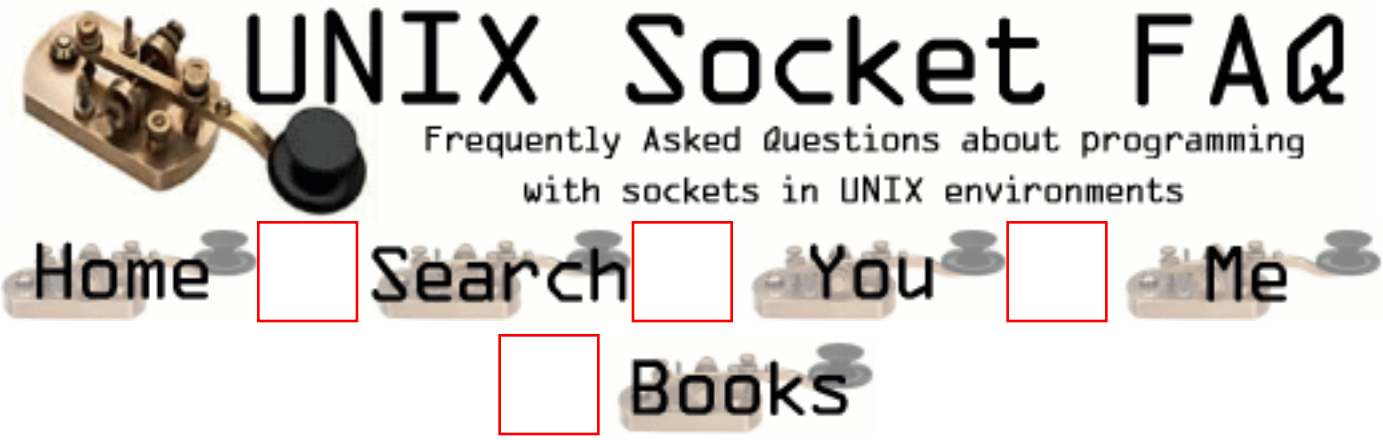
From: yhel

How can I filter the connection to a limited number of clients. I tried to accept connection but only 1 client
can be accepted. Does it has something to do with the format of the parameter I passed in the accept() function.

From: vinay

u Can use threads or
fork just before accepting the connection and close the socket id corresponding to the child.
To limit the number of connections in case if threads u can use limited array of threads and in case of fork keep a count.

Add a Comment



New Questions: using libpcap

From: Marc

Hello all,
I am using Net::RawIP and libpcap for Perl. My question is,
when i use the program on my LAN i can capture packets. But,
when i try this on the internet i get nothing. I am on a
cable modem, and I have heard that this prevents packet
capturing. Is this true? Anyone know about this or away
around it?
Thanks,
-Marc

From: Andreas Forsgren

True, but you should still be able to capture packets going in/out from your local interface.

From: Johnathan Ingram

Hi

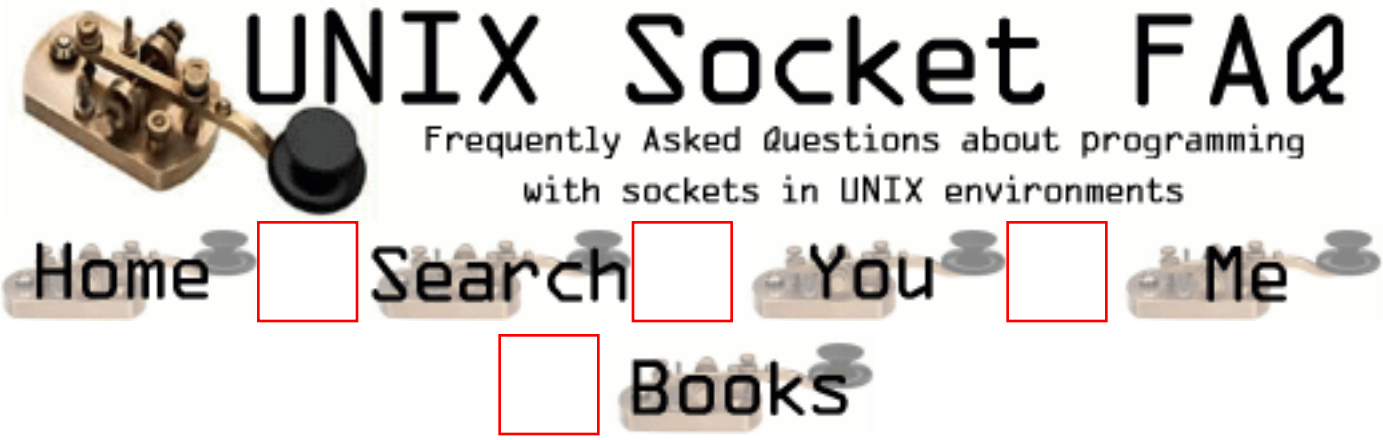
This is a limitation on the networking level.
If you are connected to a normal hub the hub acts a a repeater repeating the ethernet packets to all port on the hub. This is why you can "sniff" the packets as your nic recieves packets even if they are not destined for your nic.

Now if you where connected to a switch or your modem, the ethernet packets are not repeated accross all ports on the switch. You cannot "sniff" packet as the packet will never reach you nic unless it was destined for it.

This is why we have switches. To reduce traffic on lans and to prevent packets from been sniffed.

L8r

Add a Comment



New Questions: buffer limit

From: Anup

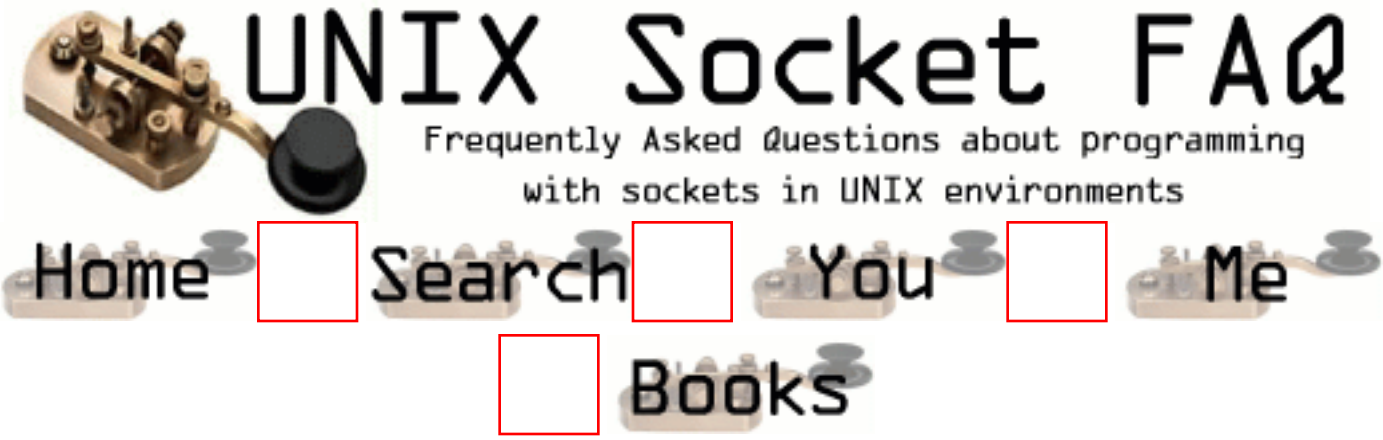
While writing a huge amount of data as a single block into
the socket is there any limitation to the size of the block?
if so will the socket block until some amount of data
is read and what is its size?

From: skuzzlebutt

I think default buffer size is 4096 bytes.
Socket buffer sizes can be set using the setsockopt system call.
However this doesn't mean you can write this in one go and receive
it in one go. So you will need to loop around your read/recv until
you get all the data. Say you are writing to a TCP socket with the buffer
set to 20K, some of the data will get buffered up and sent, while
waiting for an ack more data gets buffered, when the ack is received
the buffered data gets sent and while waiting for the next ack more data gets buffered
and so on until all the data is sent. Hence a single read/recv won't do.
So loop until all the data is recvd.

I have used sockets with 250k buffers without problems.
Be sure to size the sending and receiving buffers, strange
things happen when data that's greater than buffer size arrives.
Like the sending process being killed instead or returning
after the write/send.

Add a Comment



New Questions: How to find local IP address in client.c program

From: Warren Nash

Trying to find find local IP address - would I use an environment variable to do this. Or is it done in one of the functions e.g. socket().

From: Rob Seace

There are a couple common methods... Probably the most common that I've seen is to call either gethostname() or uname() to get the local hostname, then just look that up with gethostbyname() or getaddrinfo() or whatever you prefer... Another method I've seen is to hunt through the list of local network interfaces, and just grab the addr of the first up, non-loopback interface (which will generally be an ethernet card or a PPP interface or whatever your main connective interface is)... This can be tricky unless you already know how to get such info... It typically involves a lot of obscure ioctl() calls... I'd say the gethostname() method is probably the easiest/best way to do it...

Or, if you mean you already have an established connection somewhere, then it's really simple: just do a getsockname() on the connected socket, and it'll give you the IP of the exact interface that you are routing out through for this particular connection (just in case you happen to have multiple IPs for this host)...

From: Warren Nash

Rob,

Thanks for the reply - but I have tried to use the getsocketname()
function and I receive a segmentation error. Here is part of my code.

```
getsockname(sockfd, (struct sockaddr *) &cliaddr, &len);
printf("%ld %d \n\n", cliaddr.sin_addr, cliaddr.sin_port);
/* inet_ntop(AF_INET, &cliaddr.sin_addr, str, sizeof(str) );*/
/* printf("local address is %s\n", *str); */

printf("My Port Number is %d\n", cliaddr.sin_port);
inet_ntop(AF_INET, &cliaddr.sin_addr, str, sizeof(str) );
printf("My IP Address is %s\n", *str);
```

Lastly, I was wondering if anyone out there has a proper client.c and server.c
e.g. Robert Stevens that has "printf" debugging through - so
So I could see waht gets passed / updated / returned by all the
functions that could be used in client.c and server.c

King Regards

Warren Nash

From: Rob Seace

Well, I'm going to guess it's probably just the printf()'s
causing the seg-fault... In fact, unless that was a just a
typo, I suspect the major problem is passing "*str" instead
of just "str" to the printf() that prints the IP string...
Or, it could be passing the whole "sin_addr" struct as "%ld"
too; even though the struct only contains a single long, I
still don't like the looks of that... Better to dereference
it further as "cliaddr.sin_addr.s_addr"... But, the basic
details of what you are trying to do look ok to me... No
reason it shouldn't work...

From: Pandurang

Give the actual array size of str instead calling the sizeof(str), which gives you only 4 bytes result

---

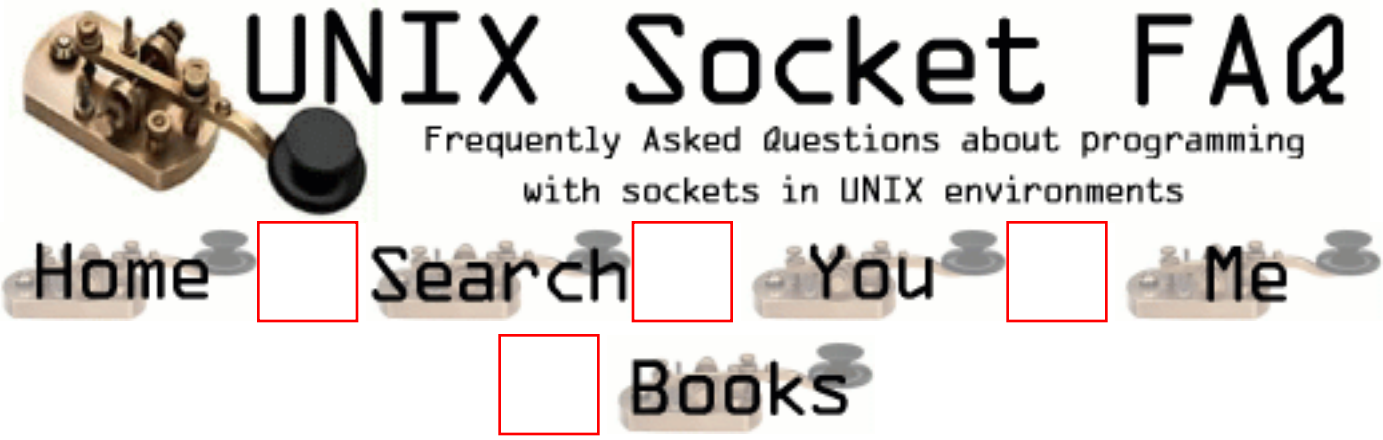
From: Jim Patterson

When you call getsockname(), you need to initialize len to the size of the socket address structure you pass in before calling it. That might be why your code crashed.

From: Anil Maipady

MIB II have an entry which stores the ipaddress of the host that is "iso.org.dod.internet.mgmt.mib-2.ip.ipAddrTable.ipAddrEntry.ipAddress" you can query this to find the IP address of local host, since socket calls will always do a look up.

Add a Comment



New Questions: Swap area

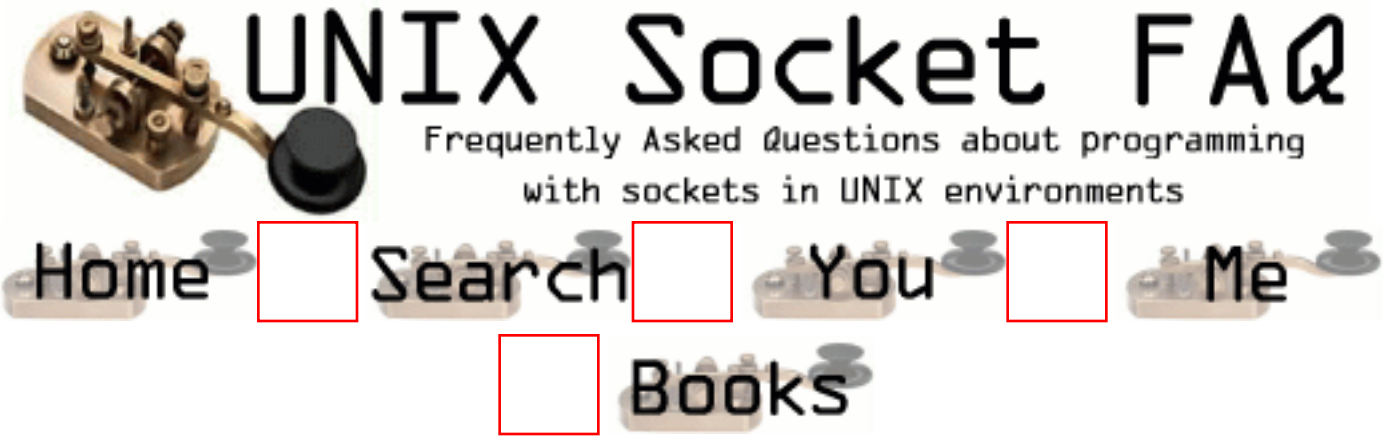
From: Anup

Is there any relation between communication through sockets
and swap area? While writing continuously a large buffer to
a socket and reading from it(the memory for read is allocated dynamically
using malloc and making it free after processing in a loop)
the swap area gradually decreases and finally system shows
runtime exception. Is this a general behaviour of unix?

From: joe

If it were, it would be hard to understand why some servers running under Unix can process millions of requests every day and stay up for months and even years without being restarted. It sounds to me as if something you believe you deallocate, probably a buffer, you don't. A good investment for a programmer is a debugging heap allocator, the equivalent of something like Smartheap in the Windows world. Many Unix libc's come with one, eg. GNU. Best to exhaust this possibility before assuming something horrible about the operating system.

Add a Comment



New Questions: Using socket fuctions and debugging

From: Warren Nash

Rob,

Thanks for the reply - but I have tried to use the getsocketname()
function and I receive a segmentation error. Here is part of my code.

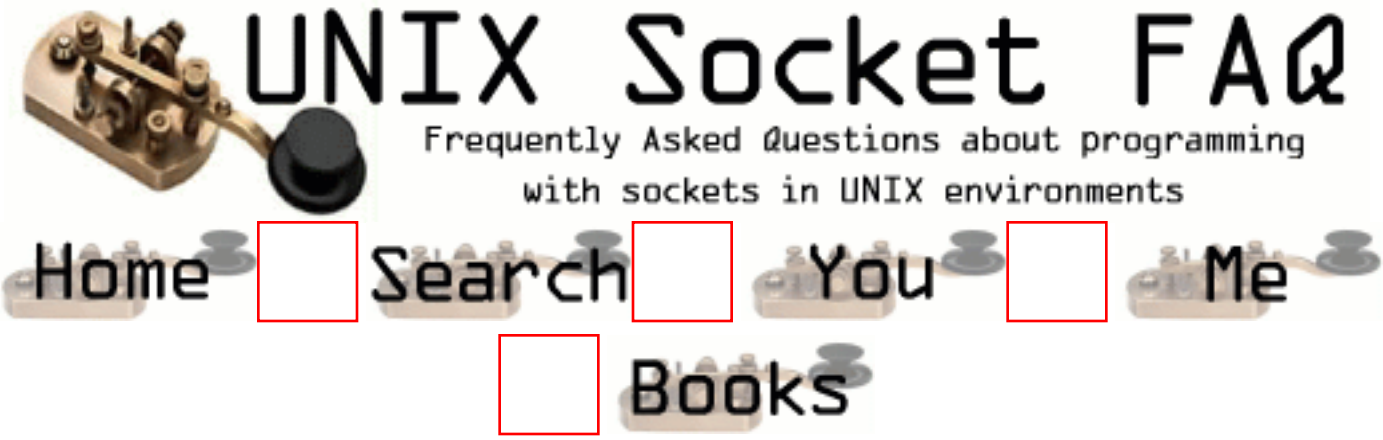
```
getsockname(sockfd, (struct sockaddr *) &cliaddr, &len);
printf("%ld %d \n\n", cliaddr.sin_addr, cliaddr.sin_port);
/* inet_ntop(AF_INET, &cliaddr.sin_addr, str, sizeof(str) );*/
/* printf("local address is %s\n", *str); */

printf("My Port Number is %d\n", cliaddr.sin_port);
inet_ntop(AF_INET, &cliaddr.sin_addr, str, sizeof(str) );
printf("My IP Address is %s\n", *str);
```

Lastly, I was wondering if anyone out there has a proper client.c and server.c
e.g. Robert Stevens that has "printf" debugging through - so
So I could see waht gets passed / updated / returned by all the
functions that could be used in client.c and server.c

King Regards

Warren Nash

Add a Comment



New Questions: how does one measure throughput in TCP and UDP?

From: Abhinav Keswani

If I want to measure throughput of different sized messages
using TCP and UDP - can anyone suggest a good place to find
C code that does this? Alternatively, can someone suggest some
tips for writing my own code - specifically how to time arrival
of packets at receivers end. Thanks.

From: Rob Seace

Maybe take a look at bing... It actually seems to use ICMP to do its measurements, rather than TCP/UDP, but it may give you some ideas...

From: Andy

download an IP sniffer. A good one will give you readings on num of packets, size, bytes, etc, as well as source & destination address and port.

From: Anton Hendriks

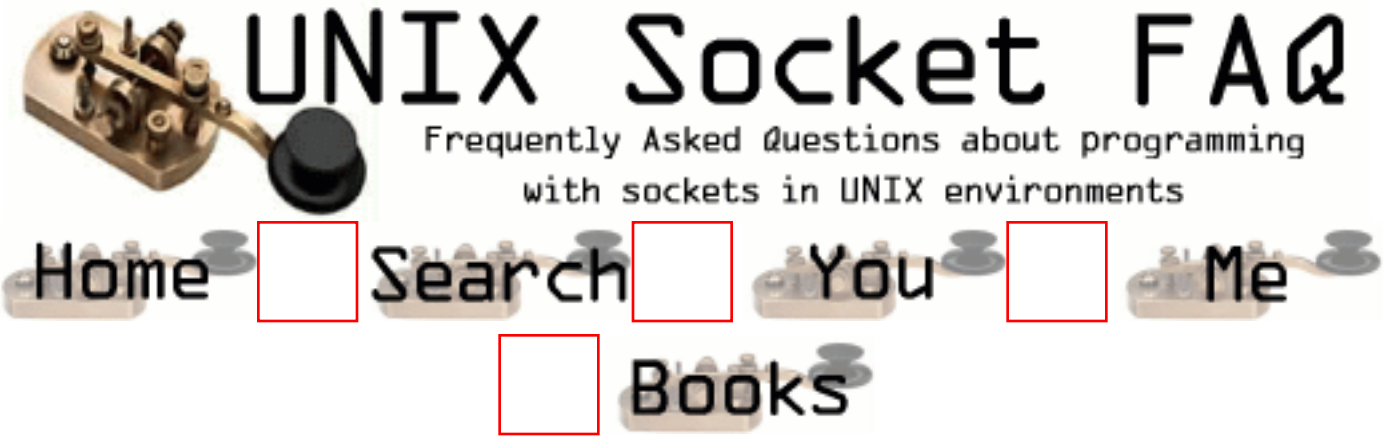
Take a look at LMbench at www.bitmover.com/lmbench
This is a benchmark suite, which includes a bechmark for TCP bandwidth.
Measuring UDP bandwidth is very dificult, if not impossible, due to the nature of UDP.

From: Arvo Koppel

A question for Andy, or anybody. Please,Please, can you suggest a byte-counting/logging by IP program that is not $13,000, or could anybody suggest where to start on this. Capturing 'snoop' output and parsing it down simply does not cut it, due to traffic levels.

What is the underlying mechanism used by UNIX 'snoop' command (source code?) so that I can butch it, get rid of the 'content' and simply catch the byte size of each packet?

Add a Comment



New Questions: defnition

From: anitha

what is socket?

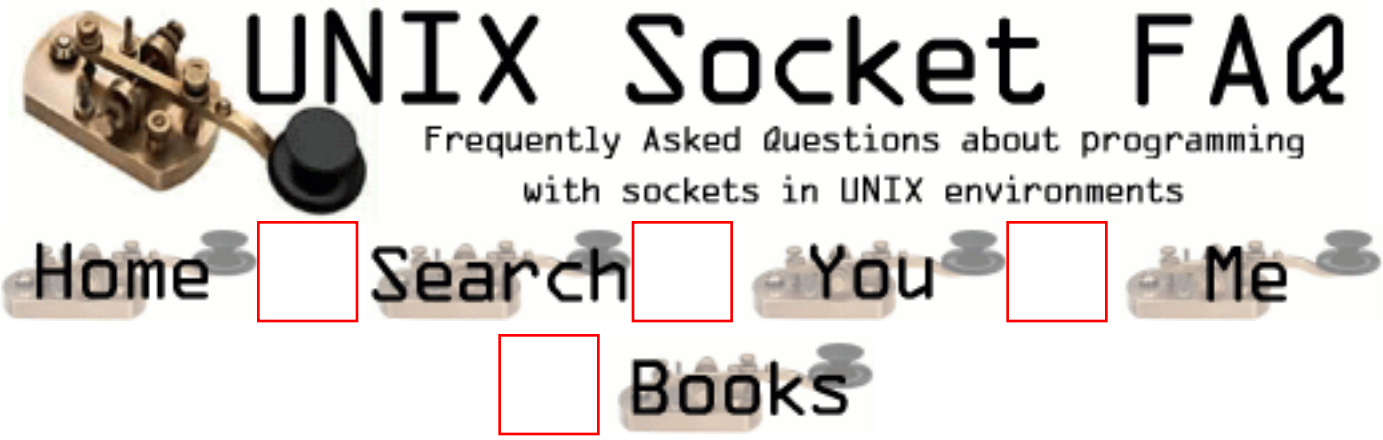
From: Mahafuz

In plain words, socket is the communication tunnel/channel
endpoint used by your program.

Technically, Socket is the file system like interface
(file desriptor) provided by the OS, wrapping your communication point. Through socket you send / receive data to other programs (usually over network).

pardon my english.

Add a Comment



New Questions: source-IP equals dest-IP

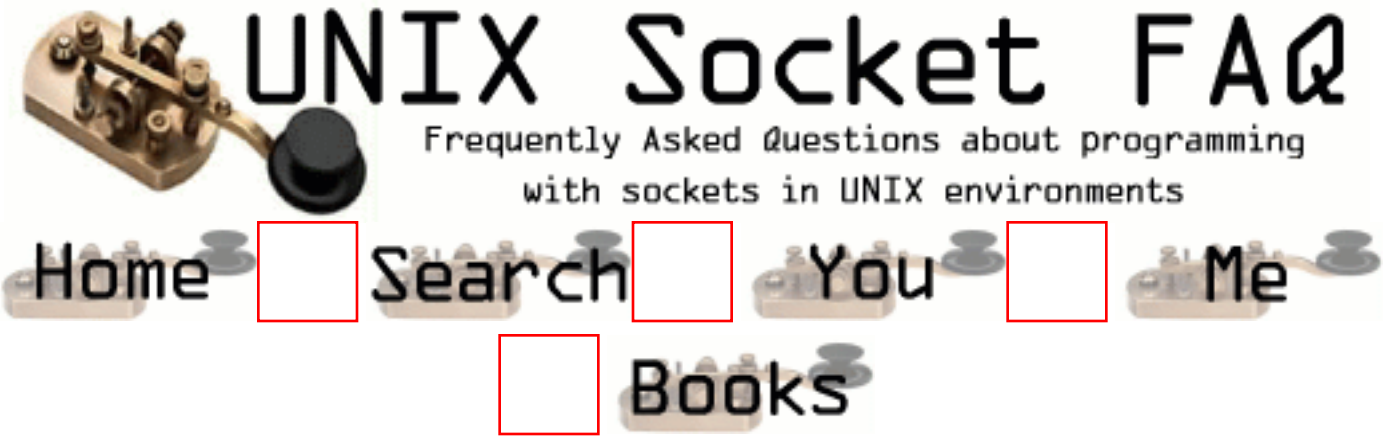
From: Lukas

why does the structure 'ip' tell me that ip.ip_src
and ip.ip_dst are the same? i'm using RAW sockets
in linux.

please mail me any suggestions

From:

Add a Comment



New Questions: Proxy Server written in C using sockets

From: Warren Nash

Does someone have an example of how to write a proxy - while using the example(s) in Richard's Unix Network Programming Book.

From: Vaishali

Excellent!!!!

From: Wang Xidong

I have implemented one socks proxy and one http proxy, is there anyone who want to see it?

From: Warren Nash

Yes I would like too see the HTTP proxy - as I am at the moment not reeiving all data back from the web server, I would be very interested how someone else may have written there proxy server.

Warren Nash
wnash@dme.qld.gov.au

From: Antonio

I would love to see the SOCKS proxy...I would also like to know how to make one of my own (always in C), or make a TCP Mapping service... if anyone can help please email me

---

From: Stephen Tsang

I would like to see the Proxy Server.

From: nu

I really need to see the proxy sever (HTTP proxy)i try to implement one and i face many problem.

From: Satya Prakash Prasad

Great Work !!!

From: ziyouma

I want to see your proxy server. Thanks.

From: Srikanth

I would like to see the Proxy Server.

From: vivek

i have got a big assignment in network programming
can any one help if yes please mail me
vivekupadhyay@hotmail.com

From: emlyn

yup please e-mail me with the code

From: Jason

I'd like to see your proxy server code too ;)

From: Parthiban

Currently, I'm developing proxy, i would like to see u'r code.

From: srini

From: srini

sir,

i am trying to shift unix c, c++ programming
i would like to see your http and socket proxy code please
try to send me.

From: sarala.M

I would want to know where/when and why to use SOCKET Programming.

From: h_hun.C

I'd like to see your proxy server code too.
please send me email with code~~

From: Vertinski

And for me too, please..

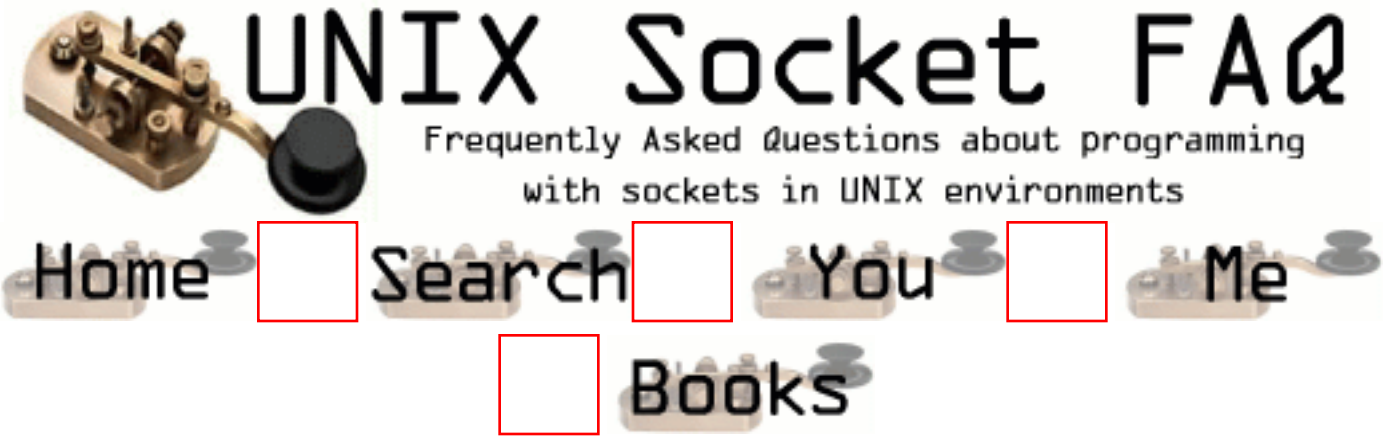
From: Luksz

I would like to see the Proxy Server.
Thanks

From: tomas

if anyone have the code
please send it to me
thank you ^^

Add a Comment



New Questions: gethostbyname

From: hanaa al-ostad

I'm trying to write a client program that uses gethostbyname to look up the machine names at my site .
it's a simple program but when I complied it it gave me an error that the gethostbyname is not defined
although I have included all necessary file and I wrote exactly as in my book .

From: Warren Nash

I believe you may need to verify the formal parameters and
actual parameters that you are dealing with.

Remember it returns a pointer to a hostent structure. If you have
Richard Stevens book on Network programming - sockets and XTI
- see page 244 for implementation details

From: Kumaran M

Hai
  i guess u have to include the library option -lxnet for including those library.It should work by that way.

Kumaran M

From: Kumaran M

Hai
  i guess u have to include the library option -lxnet for including those library.It should work by that way.

Kumaran M

From: Arthur Lung

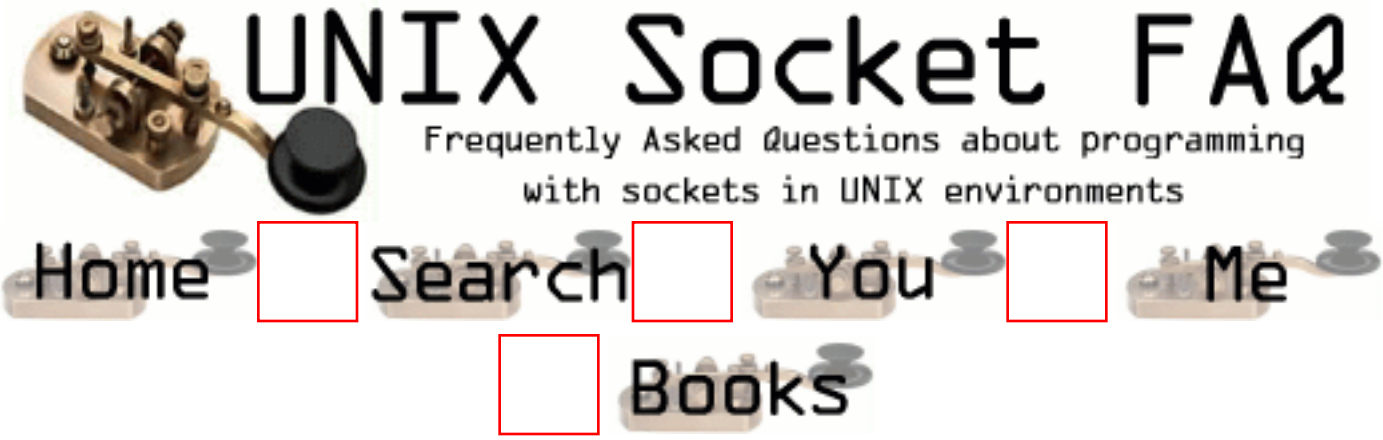
If it's a compile-time error, and you're sure that you have all of the correct include files included, and you're sure you didn't spell it wrong, it could be that your include directory path is wrong.

If it's a link time error, and you're sure that you have all of the correct libraries included, and you're sure you didn't spell it wrong, it could be that your library path is wrong.

I wouldn't assume that the library/include paths given in a book reflect your reality at any given time (assuming that you're also using a sample makefile or something out of the book as well).
If you're sure you have all of the correct include files being included,

---

Add a Comment



New Questions: select() for timing

From: Johanna

Why does this function call return the error "No such file or descriptor": select(0,NULL,NULL,NULL,&tval) where tval is set to a value greater than 0? This works as a timer on one machine environment without Forte running and does not work on another machine environment with Forte running. Is it Forte or is it some other environment setting?

From: johanna

After further testing, the select statements work well when the calls to Forte are commented out. As soon as I call ForteStartup, I get system interrupts to my select statement. The errno value was not valid since I was calling select within a pthread which has it's own errno, correct?

From: Warren Nash

the first actual paramater is the maximum number of file
descriptors - if set to zero I can only assume that you are
not selecting any file descriptors to read from.

Would this be true - the max i believe is somewhere areound 65000.
the value in the first actual parameter should be the maximum value of
your fd's / sockets on your program + 1.

From: Xidong Wang

the first parameter should be zero to inform
the system: that select call is only used for delaying
some time.
In Linux, Solaris, AIX, I test the same operation and it succeed. So I think
maybe it is the "Forte" causing trouble. can you explain what
the "Forte" is?
--wxd

From: Johanna

Forte is a Sun Microsystems object-oriented network application. See website www.forte.com
You are correct, Xidong. I am not trying to read from any sockets, I am just trying to use select for timing purposes.

From: Johanna

I solved my timer problem by using pthread_cond_timedwait instead of select(). But then the select I used for reading the sockets kept returning "Interrupted system call" (as perror showed), but the errno value was still "No such file or directory". So now I check for ENOENT instead of EINTR and call select again and everything works fine. Strange!

From: Corey

I set a timeout value at 60 seconds to perform some additional checking to see if the connection was closed on the other end do to some abnormal closure. However, once to 60 seconds elasped the select statement hangs. If there is activity to "recv" that happens in less than 60 seconds, it works fine. Any ideas?

From: Corey

I figured it out. I had the accept function prior to the select. Once the select informs the program that there is something ready to read the program should perform the accept and then the recv. Works great!

From: franky

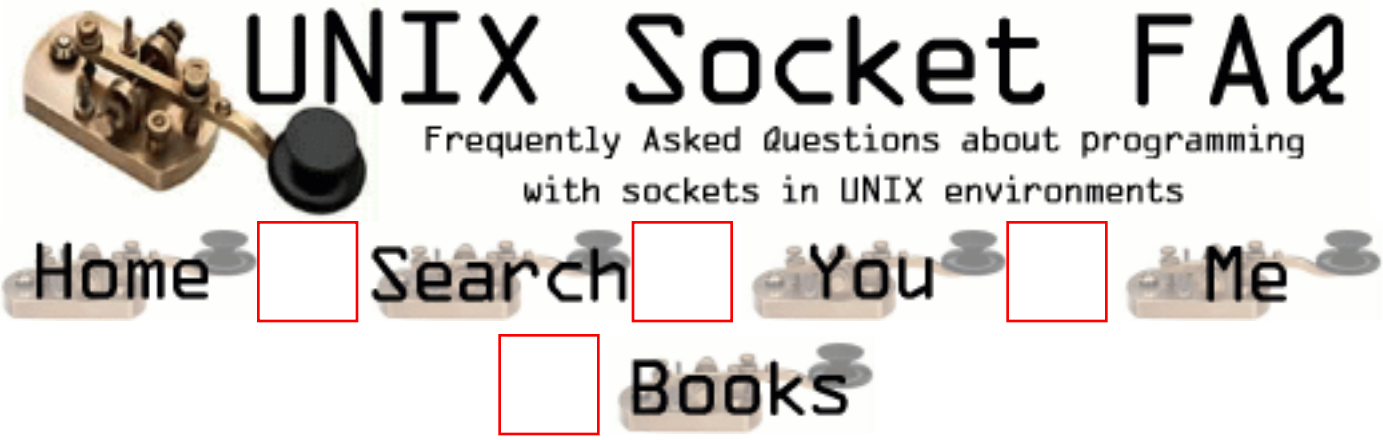
you may also be getting some issues with linking multi-threaded and non-threaded libraries together.
Or more specifically, having _REENTRANT defined before #including <errno.h>
- _REENTRANT affects the way errno works - which sometimes causes the wrong errno to be referenced.

Also, you should be able to pass all NULL sets, but there is no reason that you can't pass a set which has no FDs set in it (ie an FD_ZEROed set). There is also no reason to say maxfd is zero, as long as your sets are NULL or empty. This may make it work. Other then that, only signals may be stuffing up your select.

cond_timedwait is a pretty good solution for timing because it can be cut short by another thread at leisure. Great for indicating that you do not need to wait any longer.

fv.

Add a Comment

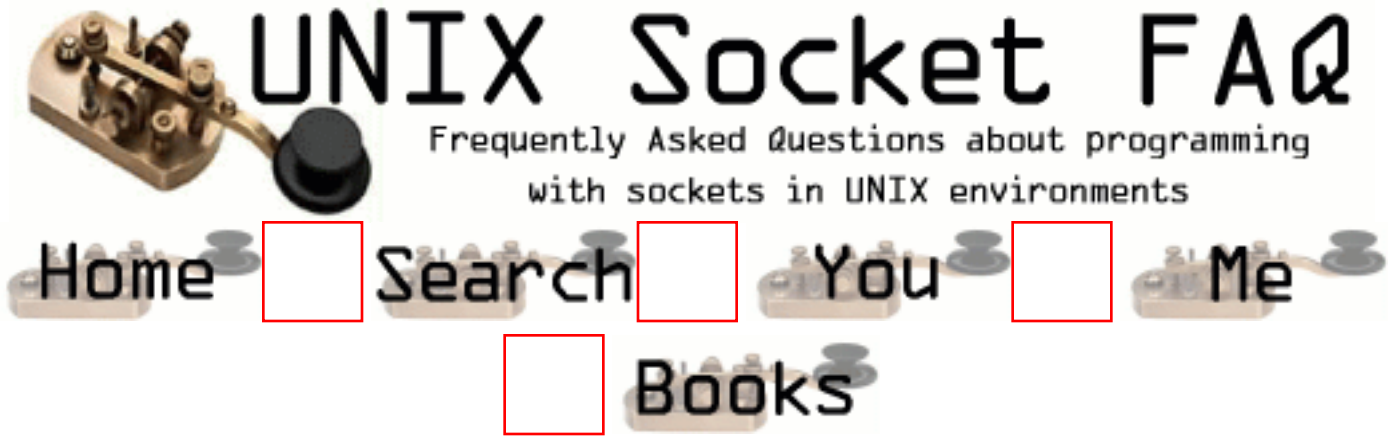


New Questions: nic card

From: asheesh

hello
iam making a simple program to capture frames. can someone tell me how to get the address of the
buffer in my system memory where the buffer is created? Can someone tell is there a version of tcpdump for winnnt?
(iam rather new to net. prog. ; therefore the question might look
trivial to some.
regrads
asheesh.

Add a Comment



New Questions: How do I access individual packets being received on a socket?

From: Tym

Hi,
Is there a way to read/write one packet at a time on a socket? Idealy I'd like to call something like readpacket(sock, buff), and I would get back the data contained in the first packet that is read from the socket, as well as how many bytes were read. Similar thing for writing (ie. write a packet at a time). Is this possible with sockets, and if not, how would I do this in UNIX. Do I need to use Raw Sockets, and/or libpcap?

thanks for any help at all,

tym

From: Bruce Edgerton

Tym, a simple way is to write the length of the packet first and then at the other end do one read to get the length, followed by another read for the actual packet.

From: Jacques Richer

Also: be _very_careful_ about memory allocation and received (network) data... If you allow network data to overwrite the end of a buffer, you may have a security problem with your application.

Regards,
Jacques

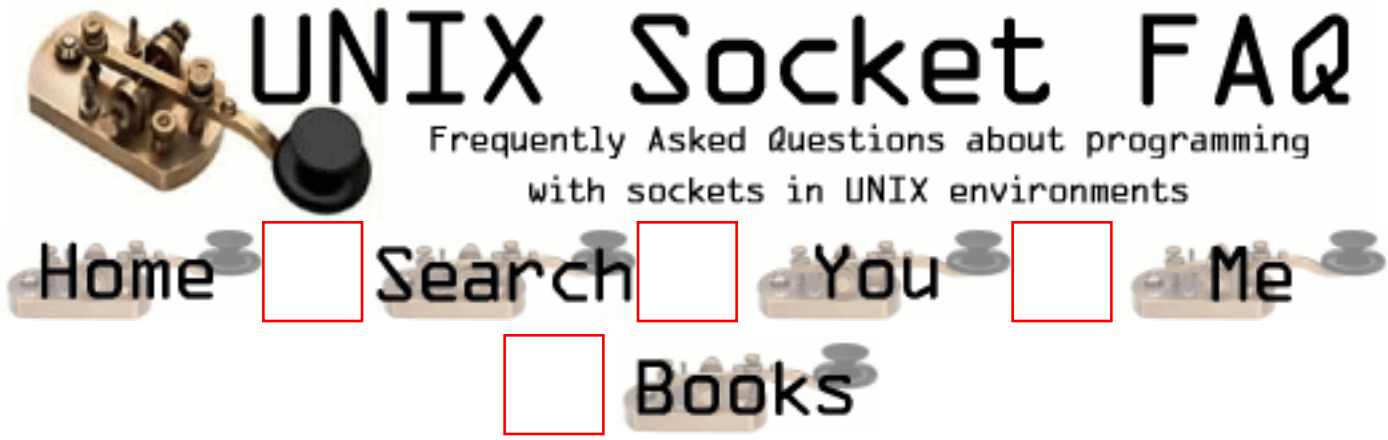
From: kapil

Hi !

I think u can do it by using the writen and readn functions given in the richers stevens book. hope this helps

---

Add a Comment



New Questions: Time out on Recv

From: Krishnakanth

I am using select to time out on a recv. This socket is a UDP socket. Sometimes even after select returns, FD_ISSET is not set. When i tested by ignoring this recv call gives me an error "Bad file descriptor:" .

Can somebody explain this

Thanx in advance

Krishna

From: Rob Seace

Well, without seeing the code, one can only guess... But,
possible causes are: the timeout is being exceeded; or, you
are getting hit with a signal of some sort which interrupts
select()... What is "errno" set to? What exactly is the
return value from select()?

From: sapnakaiya

Hi Krisnakanth,

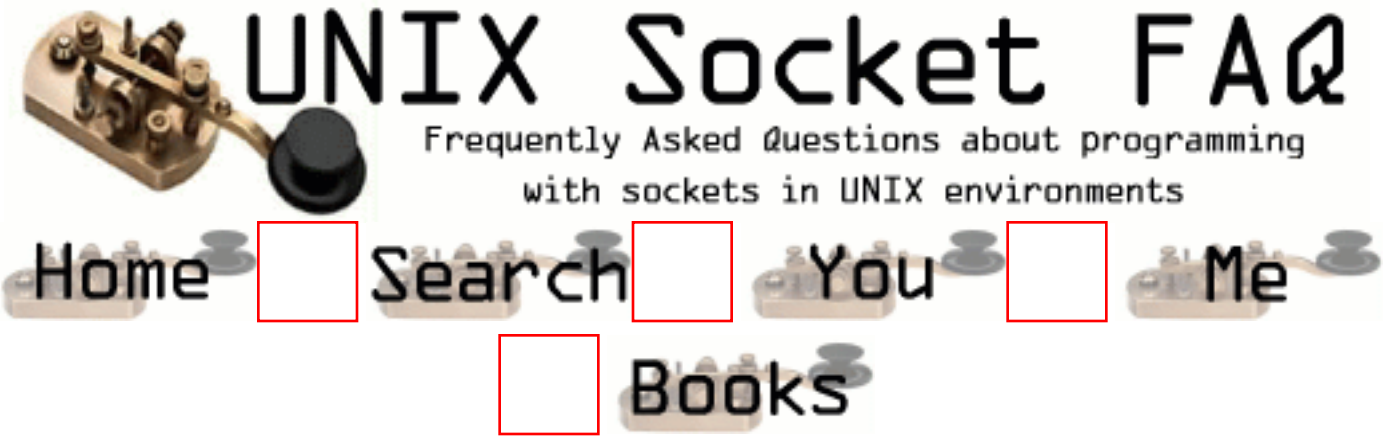
I am facing a similar problem. I have a udp file transfer application.
In this, my server is faster than the client. So if my
server sends 400 packets ,the client receives somewhere around 380 packets
and then hangs. I want to come out of this state and want to see how many packets r
lost.The server exits out properly but the client instead of exiting just hangs.

I made the socket non_blocking so that if there r no msgs on
socket ,it will exit. But in this case it doesnot recv any packets at all.

Can neone explain this to me. I would be greateful if u could mail me abt the same.

Thanks,
sapna.

Add a Comment



New Questions: restart server

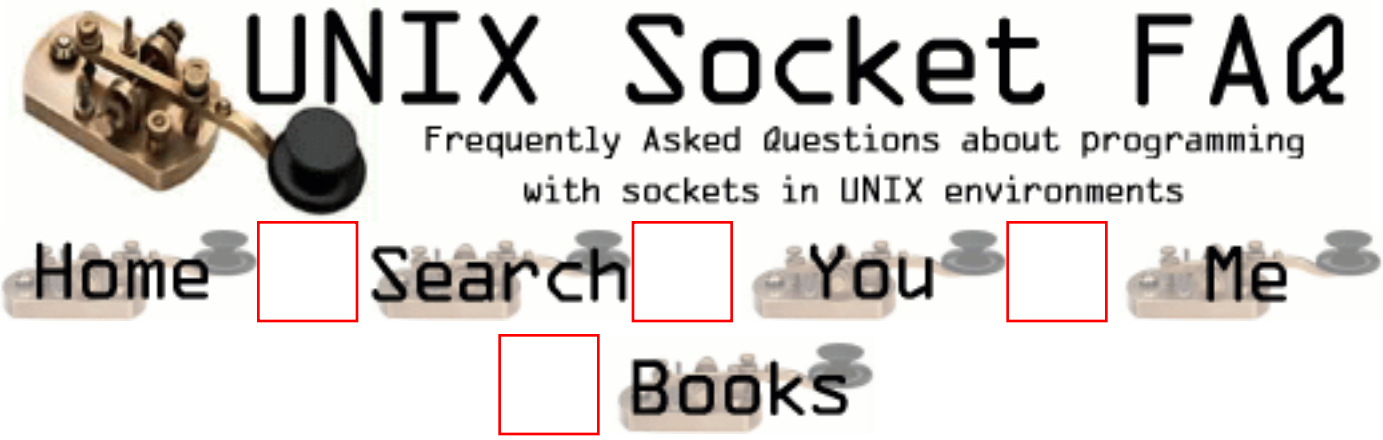
From: V.C.Mahadevan

Sir,
I am now engaged in a Multithreaded distributed fault tolerant software development.When the user wants to change the mode from distributed to non-distributed mode (or viceversa),he can give a command,in which case the application spawns a thread and communicates with the user.Obviously, its neccessary for the application to be restarted in other mode.But while restarting I am unable to bind a new server, as the old socket is still in TIME_WAIT state.What should I do, for avoiding this unnecessary wait state?

From: Rob Seace

Well, you should probably start by actually looking through the existing questions in the FAQ to make sure your question isn't already answered before asking it... Specifically, this issue is covered in Question 7, Section 2, and Question 5, Section 4... In short, you'll need to use the SO_REUSEADDR socket option...

Add a Comment



New Questions: SO_RCVBUF

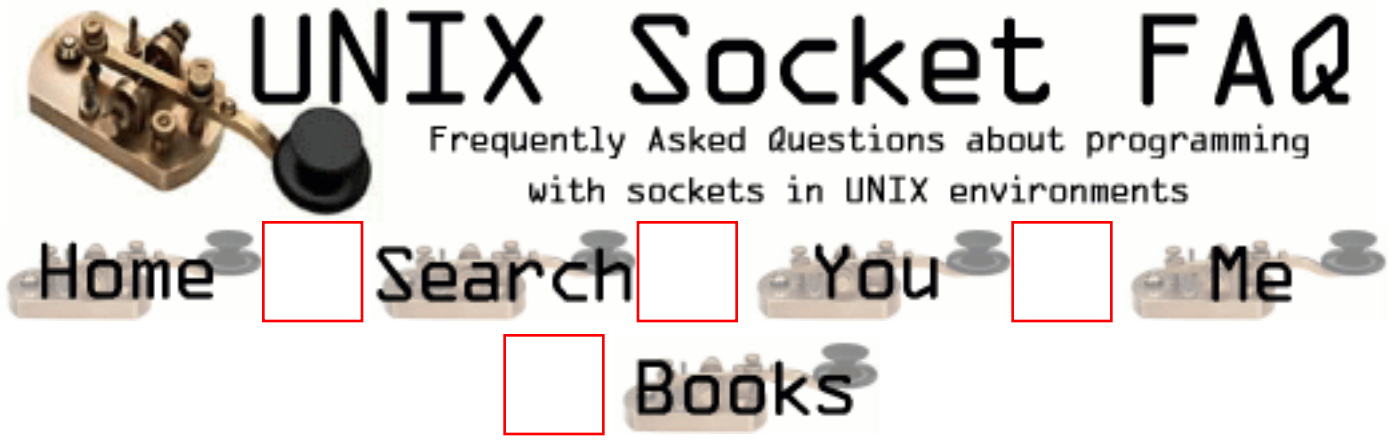
From: jagadeesh

Any body can asnwer this.
Why my socket is not receiving even after setting of SO_SNDBUF and SO_RCVBUF to 32768. Sender is sending but the receiver is receiving only 4380 bytes.
Thanks
Jagadish

From: jsh

I found out myself.

Add a Comment



New Questions: recv() fails with EGAIN for a blocking socket

From: AK

Hi,

I have to read TPKT packets from the network. For this I am
using recv() with MSG_PEEK option to first peek at the packet
length in the TPKT header and then using recv with flags=0
to actually read the packet. The socket in question has
been set to blocking using the foll. code:

```
 const char on=0;

 if(ioctl(sockfd,FIONBIO, &on))
 {
   printf("ioctl failed\n"));
   close(sockfd);
 }
 printf("socket %d has been set to blocking\n", sockfd));
```

I'm facing the foll. problem: the packet header says there
are a finite no. of bytes to be read in the packet.
But when i go on to actually read those no. of bytes,
recv() is failing with error EGAIN (Resource temporarily
unavailable).
What does this error mean and why is read
returning at all if the socket is blocking ??

From: Terrance

Got this error too - but what I did was to go into a wait state (e.g. sleep(10)) and try the socket connection again. If this does not work - it will loop and wait a number of times before reporting it as an error.

Usually for my testing it is not more than 2 sleep(10) when the system free the resource enough for my program to continue.

Hope this helps.

From: Arthur Lung

EAGAIN means:
Your socket is set to non-blocking mode, and the receive operation would block (because there's no data, or you tried to read more than there is), OR, you set a receive timeout, and the timeout expired before any data became available.

From: sielim

Yes, that it is. See man pages about poll and select functions.
And how to use nonblocking sockets in this faq.

Add a Comment

UNIX Socket FAQ

Frequently Asked Questions about programming with sockets in UNIX environments

New Questions: sending floats over winsock

From: vikram

Hi,
I have been trying to pass a float buffer through a windows stream socket.
the send() method only accepts char* ! conversion of float into char is one method i thought of.
But this kills the resolution of the data.
Could anyone please help??
Thanks in advance
Vik

From: Shawn Elliott

I'd try to write a conversion function on both ends to convert with more detail. I've had to do this on some UNIX systems where some functions were missing.

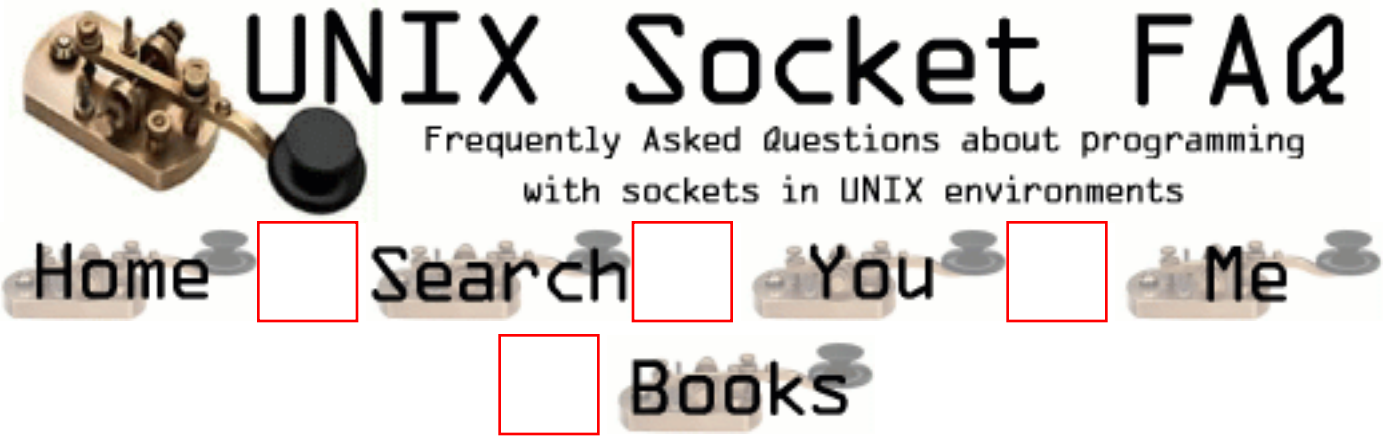
From: Phil Frisbie

You don't need to cast the floats to chars, you just need to cast your float * to a char *. And when you receive, cast the char * back to a float *.

From: Rob Seace

Note that Phil's method will only work if both the client and server are exactly the same platform, or just happen to share the same binary storage method for floats (not to mention byte ordering)... In general, sending raw binary data over sockets is a no-no... But, if you KNOW both ends are going to be on the same platform, you can get away with it... (I do it myself all the time, actually... But, that doesn't make it any less of a bad idea, in general... ;-))

Add a Comment



New Questions: A server on multiple machines behind a Net Address Translation Device

From: Richard Foletta

We have a requirement to use a Network Address Translation Device to provide internet connectivity to multiple machines on a local net using a dynamic IP from the ISP. Host A and host B have local IP addrs not known to the internet. The nat device has a dynamic IP addr. There is a server listening on port 55565 running on A and also running on B. I could manually configure the nat to say that its port 55565 is mapped to A.55565 (system A port 55565) and its port 55566 is mapped to B.55565 but I don't want this requirement. What I want is to be able to have A and B make a socket connection through the nat to a third host C. This host C will do a getpeername() to find out what the IP and port is of the connecting system and store it on C. A would connect to C and C would log A's address as nats IP.55565. B would connect to C and C would log B's address as nats IP.55566. This way some other system could ask C what the IP and port is for A and make a socket connection directly to A.

The question is how do I get the nat to map one of its ports to A's 55565. I think maybe the first thing my server should do is bind to its port 55565 and make the connection to C allowing the nat to map the port. Then close the connection and then listen on its port 55565. But wont the nat drop the mapping when the connection to C is closed. I need the map to be set up and stay set up as long as the nat is connected to its ISP.

Another thought was for a serparte program using a different port to make a connection to C and send some data containing A's IP and port 55565 through the socket in some protocol that the nat will recognize and do the mapping. When C reads the data the port in the data will have been changed by the nat to that of its port that it had mapped to A's 55565.

Hope this was not too confusing.

Add a Comment

New Questions: qustion about socket

From: DHAVAL

hello sir
i have a question that can i change the address of a socket
why and how?

From: Warren Nash

are you talking about using pointers - it would be difficult,
but possible - I suppose you would need few structure declarartions
to handle each socket address movement (rememeber a socket is an
endpoint).

From: Warren Nash

Or are you talking about IP address - see previous for possible answer.

From: Raja

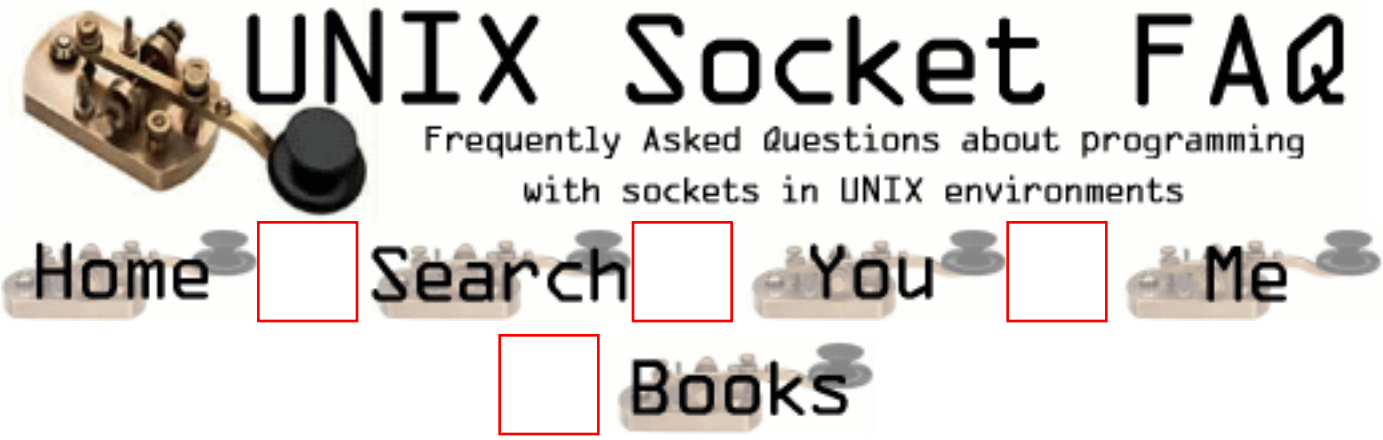
Actually theres a server which was executing some flow.Now i want that server to accept requests from other server and based on the message it has to perform the actions apart from the normal flow.So i was trying to create new thread apart from normal flow of execution and that thread keeps listening at one portnumber and IPaddress so that whenever a request saying "STATUS" comes it sends back "UP" and if the message is "SHUTDOWN" it shud close the thread and also the normal flow.So i was using the sockets concept where it accepts messages and sends back the respective message.Thats it of my execution.So pls give me some suggestions on that.I'm able to send message from the other server but this server is not able to catch though its listening one the same port to which the other server is establishing a cnnection for sending message.pls respond me quickly.

From: raju

Are you doing a connect from the second server to the first server?
You do the same things that you do from a client socket, connect, send in your

second server.

Add a Comment



New Questions: how to determine if socket is blocking or non-blocking

From: TR

Hi,

Can anyone tell me how I can determine
whether a socket is blocking or non-blocking ? getsockopt()
does not seem to have any options for querying this. I am
working on Sun Solaris platform.

TIA,
TR.

From: Rob Seace

It's not a socket option, but rather just a generic file
descriptor option... So, use fcntl() with F_GETFL, and look
whether or not O_NONBLOCK is set...

From: alwyn

Say the socket is blocking, does it mean that write/read will
be blocking?

From: Wes

Yes, if the socket is blocking, the read/write will be blocking.

As to how to set the option, be sure you do a GET_FL, or in O_NONBLOCK, and then do the SET_FL. IIRC, there is a code snippet for doing this in the categorized section of the FAQ.

From: kapil

Hi !

I think read is a blocking system call but write is not a blocking system call.

Add a Comment

New Questions: read() and timeouts

From: Chris Parr

Hello,

I'm communicating over sockets and need some way to timeout the read() call after x seconds.

Can anyone show me how? or even show me using recv()?

Thanks in advance

Chris Parr

From: Rob Seace

Well, personally, I usually just use an alarm() based method
to timeout read()'s/write()'s... Some people might object
to that, especially if dealing with multi-threaded code...
(However, I HAVE managed to successfully use alarm() based
timeouts in a multi-threaded app before, too... It's just
very tricky... ;-)) A common alternative is to just use
select() to handle your timeouts, and don't even bother
starting the read() until the socket has select()'ed as
readable... But, depending on a variety of conditions, the
read() may or may not still end up blocking, even after
select()'ing as readable... So, to be really sure, you'd
need to set the socket to non-blocking mode, as well... As
I said, I've just always found it easier to do straight
alarm() based timeouts, and they've always seemed to work

well enough for me... Just set your alarm(), sigsetjmp() to save your state and check for the timeout condition, and then start read()'ing; and in your SIGALRM handler, just siglongjmp() to the saved state buffer, which will return control back to the sigsetjmp() in your code, with a non-zero return value...

From: Nick

I recommend the following snippet. I use it in all of my programs and it performs better than a SIGALRM.

```
int readableTimeout(int fd, int sec, int usec)
{
  struct timeval tv;
  fd_set rset;

  tv.tv_sec = sec;
  tv.tv_usec = usec;

  FD_ZERO(&rset);
  FD_SET(fd, &rset);

  return (select(fd+1, &rset, NULL, NULL, &tv));
}
```

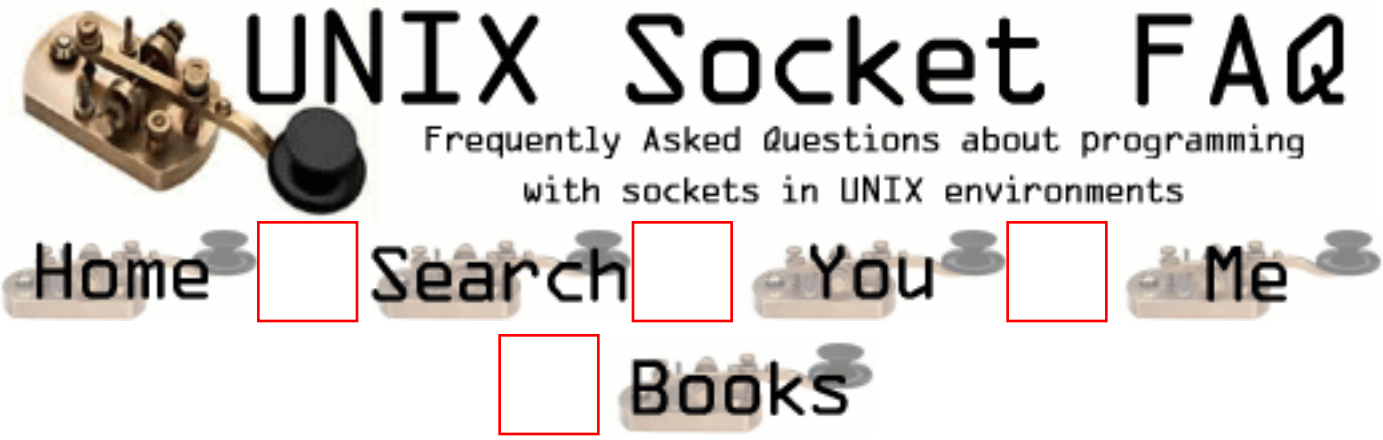
From: Phil

Take a look at:

http://cvs.sourceforge.net/cgi-bin/cvsweb.cgi/DroneUtil/src/ReadableTimeOut.c?rev=1.9&content-type=text/x-cvsweb-markup&cvsroot=droneutil

---

Add a Comment



New Questions: XDR

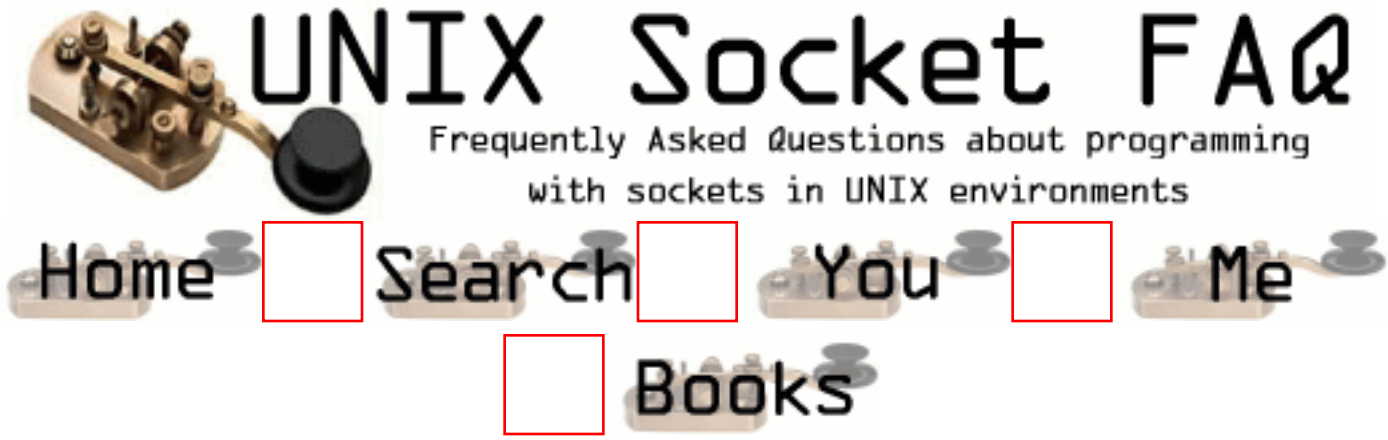
From: Warren Nash

How to use XDR with sockets - do I need to transfer all
the data to XDR before sending e.g. (int, char, etc).

From: issam qasas

you only need to use XDR when the machines in your site
are different in architecture (intell , matorolla ,...)

Add a Comment

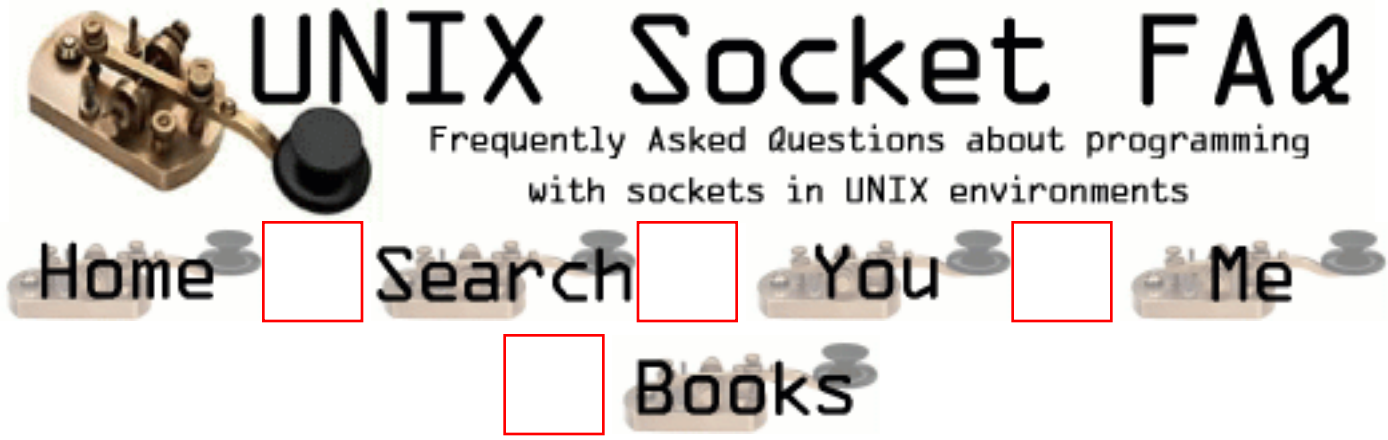


New Questions: Proxy with Sockets - grabbing the http portion

From: Warren Nash

How can I grab the GET or HEAD part of a URL web page, including the pictures URL's. This is used to limit / stop access to certain web pages. I am building an additional proxy to the ones already available in linux.

Add a Comment



New Questions: How do i send file(s) to server

From: ezzete

I use send/recv to send and receive data from/to client/server but i don't
know how to send file(s) using send command since the format for send command is
int send (int sockfd, char * buff,int bytes,int flags).
Anyone please help!!

From: Hanan

You can read the data from the file , to a buffer , and then send the data using sockets functions.

From: issam qasas

go and read this site
http://www.ecst.csuchico.edu/~beej/guide/net/

this site is suggested by the freind sebastian.

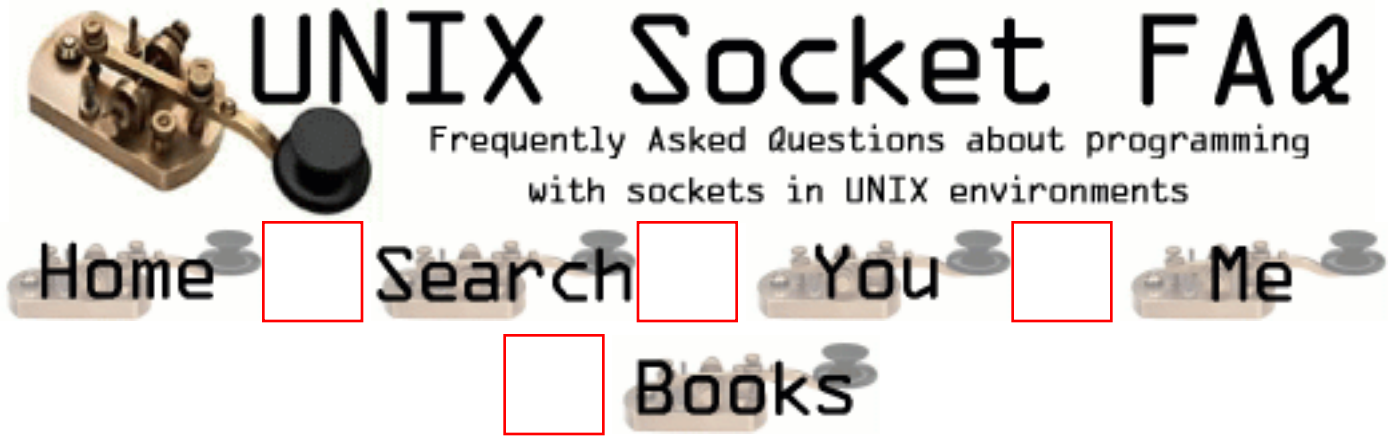
From: Alisha

How do you send a file to a server with an associated file name associated with the buffer. How do you store the file once transfered in the current working directory?

From: Rajiv

You can send the filename firstly, and then send the file contents.

Add a Comment



New Questions: NT sockets VS Unix sockets

From: sheenu Chopra

I have a project in Windows NT which I have to port unix.
Where can I find corresponding Unix API's to NT APIs.

Eg : WSASocket,WSACleanup,WSAStartup,WSAGetLastError

From:

unix sockets(BSD style) i believe, are actually much more easy to use than windows sockets. There is no WSAStartup or WSACleanup. For unix doesn't use dlls. You just need to create your socket with a socket() call and manipulate it from there. To control your i/o polling you need to use either select() or poll(). Select is easier to use but i like poll better. WSAAsyncSelect() is no more. Also in a unix enviroment there is a command called 'man', it is your friend. type 'man socket' and poof it will bring up the socket section of the manual. 'man 2 socket' means find the sockets manual section 2. Often written as socket(2).

From: Johnathan Ingram

Hi

Use errno if a socket functions returns -1 in order to determine the error that occured. As stated above.. man is your best friend and will give you the error codes.

You can also use strerror(errno) to convert the error to a string readable error.

L8r

From: Johnathan Ingram

Hi

Use errno if a socket functions returns -1 in order to determine the error that occured. As stated

above.. man is your best friend and will give you the error codes.

You can also use strerror(errno) to convert the error to a string readable error.

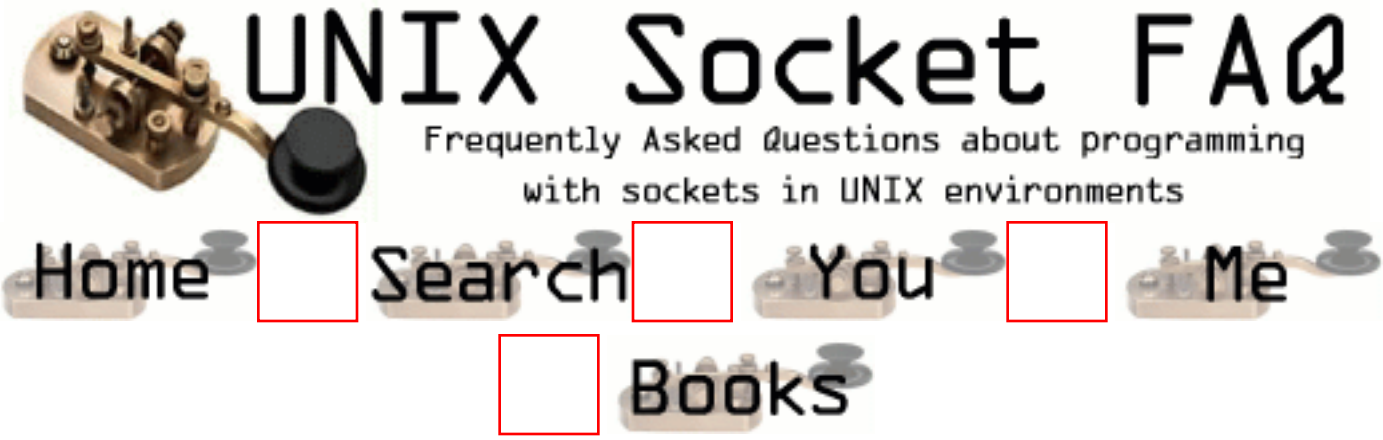
L8r

From: dzhan

i'd like to use select better, because no 'poll' in windows
enviroment.

From:

Add a Comment



New Questions: Blocking/Non-Blocking

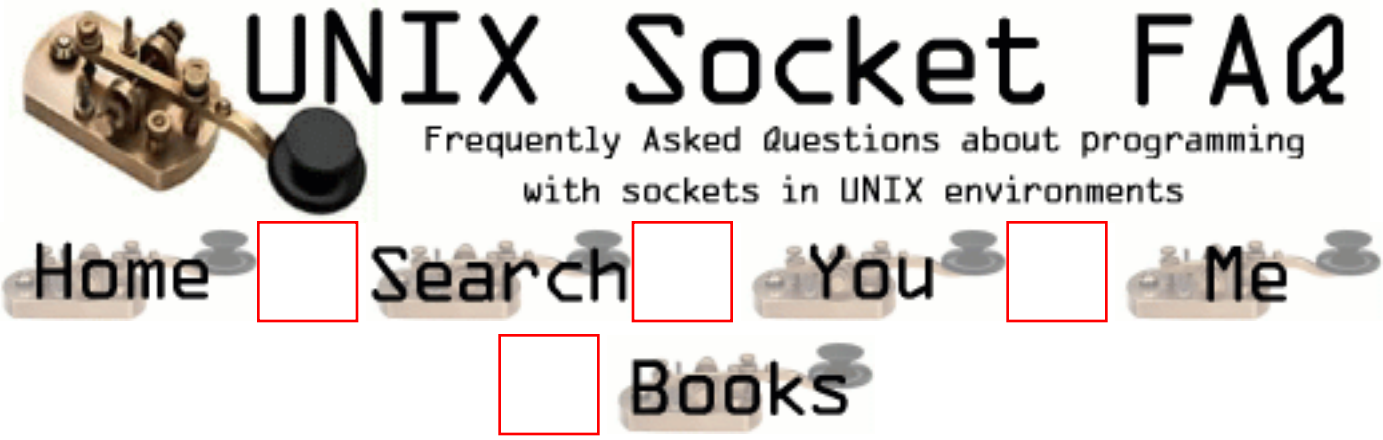
From: Marc Soda

What is the difference between blocking and nonblocking
sockets?

From: Wait - don't tell me

Well - in a nut shell blocking means "waiting for the responce"
Non blocking means - if you have the data sitting in the network card's buffer - get it, else go on and do not wait.

Add a Comment



New Questions: sockets

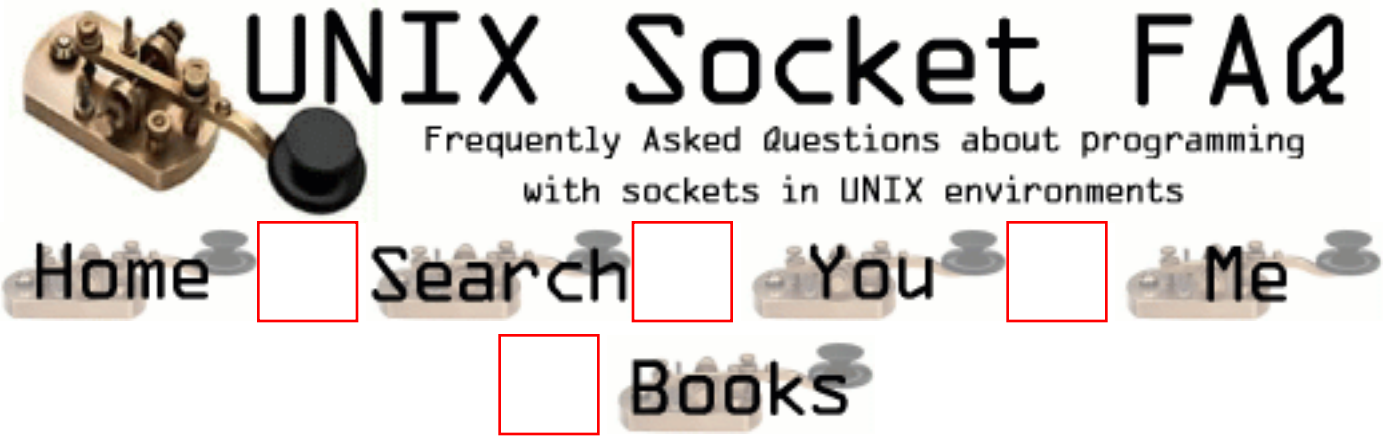
From: mary

What happens when you kill a socket, what are the methods.

From: mathur

one doesnt 'kill' a socket, one 'closes' it by the close() system call.
for a TCP socket it does not lead to an immediate 'disconnection' but the
kernel keeps sending the data in the send buffer.this is followed by sending a
'close connection' request to the peer.then we wait for an ACK and then send a return ACK
then the logical connection is considered closed. now the socket descriptor can be discarded.

Add a Comment



New Questions: Why does select not see available buffer space on a loopback socket?

From: Alastair Rae

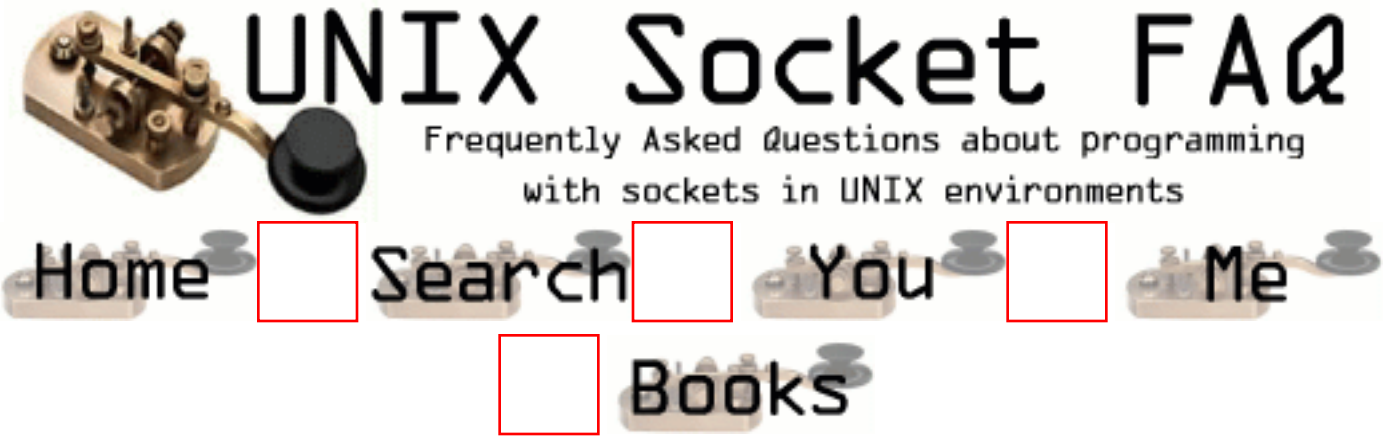
Two processes on a Solaris box are joined by an AF_INET connection.
One loops doing read(). The other loops writing to a non-blocking socket.
If the write fails with EWOULDBLOCK, the writer waits on a select().
However, the writer does not come off the select until the reader has drained ALL its messages.
Is this a bug? Why does the socket not become writable as soon as the reader has read some of its data?

From: Kieran

Are the two tasks running at the same priority?

You could try sleep()ing in the reader and writer tasks for
a short while after you send each message. There is
a possibility that the socket is ready for more writes but
the select has not yet been "woken".

Add a Comment



New Questions: socket bufsize recv and send > 4k

From: jagadeesh

How can receive more than 4k bytes of data in a recv system call?
I need to receive at least 8k of bytes of data at a time. I'm using the sockoption (SNDBUF) system call to set the buffer size in server side and also RECVBUF option is set the buffer size in client side. I'm haven't get any errors but the data is receiving only 4k. Can you anybody helps to get at least 8 k bytes of data.

Thanks
Jagadeesh

From: URAL YALCIN

use this
recv(newsockfd,buff, size,MSG_WAITALL);

yo won't get a problem anymore .....
Or use safe read command s on the richard's book.

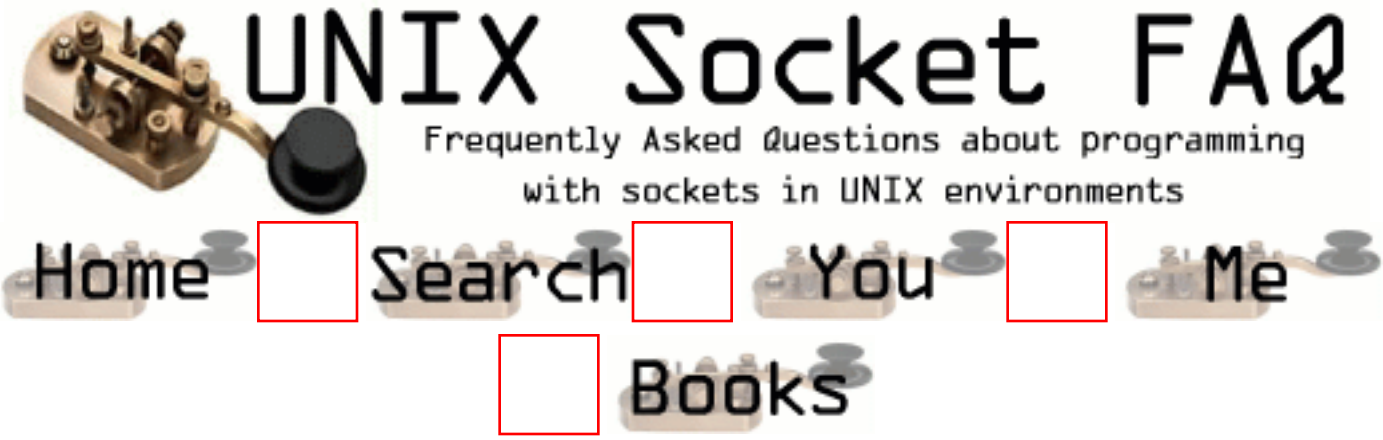
From: jagadish

Hi,
I've got solution,I'm just looping till to buffered all the data.
Thank you for your solution. I'll try this also.
-jsh

From:

hi,
we have the same situation. can u show me the solution for this.

Add a Comment



New Questions: Blocking writes fail

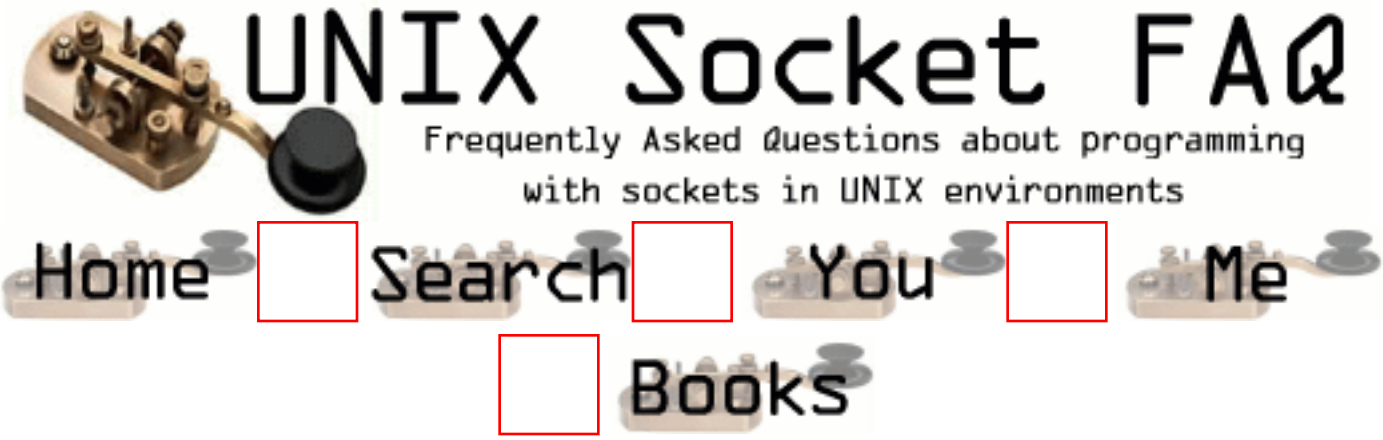
From: Mike Tarnowski

I am working with an API interface that does raw BSD socket routines. Occasionally, we get a blocking write failure. What can cause the writes to fail, and how should they be handled to stop API failure?

From: Alastair Rae

Assuming the connection is intact, the only thing that can cause a block write to fail a blocking write is a signal (EINTR) and no data will have been transferred. You should retry the write.

Add a Comment



New Questions: When gethostbyname() doesn´t work...

From: Flávio

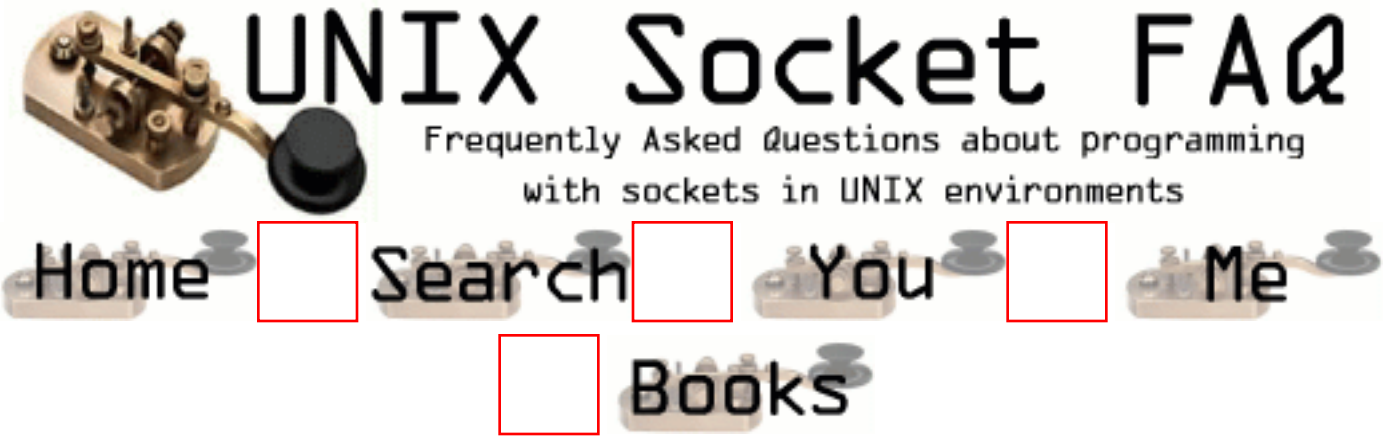
I´m programing to the BeOS and I´m having a problem to get the IP of an adress.
The host->h_addr_list[] have a hundred numbers of IP and I just have to know wich one is the right Ip when I debug the program. In a case, the right IP was the host->h_addr_list[82] in another the host->h_addr_list[86].
Everybody write to use the host->h_addr_list[0] but it is not working.
I´m using the BeOS in the University and I am behind a Firewall.

Is there a logic?
Is the problem with the Firewall?
How can I do?

Sorry for my bad English...

Thanks.

Add a Comment



New Questions: viewing active data over socket

From: Nash

I'm trying to build a simple web server and was wondering what functions I would
be using to actively look at the data being sent over my socket.
Basically how to interpret the HTTP headers so I know when to stop reading
(ie when I receive a CRLF on its own). I'm coding in C

Thanks for your help

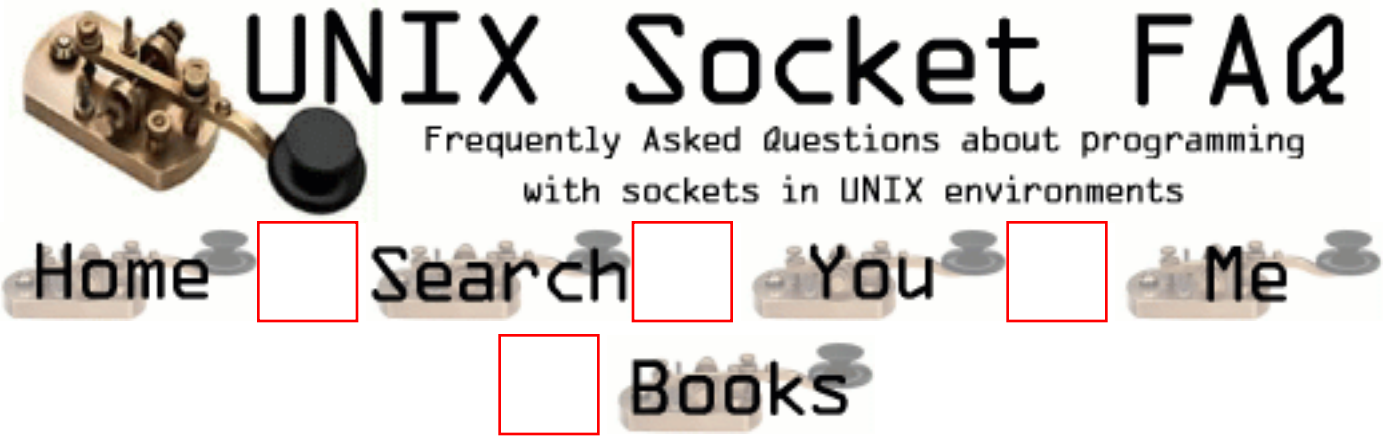
From: Warren Nash

writen() and readn().

From: Nick

I wrote a simple HTTP server that compiled to 13K. If you email me I can get it to you.
Otherwise, go to www.acme.com and look at the HTTP server examples there.

Add a Comment



New Questions: why is my client not able to connect to the server after the first time

From: latha

i have written a server and a client prog in c for unix socket communication.
when i run it for the first time everything works well.for the second time
the client program gives a 'connect' error(cannot connect).
please suggest me a solution or an alternative
thank you in advance
latha

From: skuzzlebutt

It's possible you are closing the socket on the server when
your first run of the client disconnects. It may be that you are
not disconnecting properly hence your second run won't be able
to connect as the server socket will still be in use.

From: Crich

Hello ,

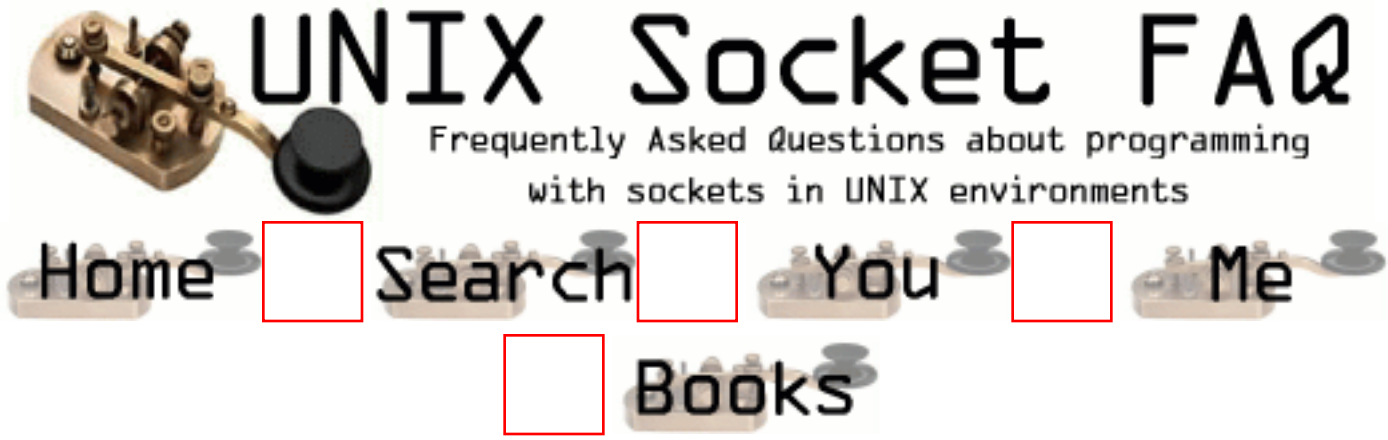
maybe you forgot to set the reuseaddress option in the setsockopt() - call .. i had a problem like this too ..

From: Michael H. Ward

You need to set the SO_REUSEADDR option flag to 1.

optval = 1;
status = setsockopt(listenSocket, SOL_SOCKET, SO_REUSEADDR, &optval, sizeof(optval));

Add a Comment



New Questions: Adding delay between IP package

From: Loui

Hi all

I am trying to se the result of delay in a small network.

I have therefore made two client-server program to send data from client to the server. One of them using TCP socket and the other using UDP socket (In Unix platform).

I want to ask

-How could I add some delay in between each IP package going to the server ?

Any help will be appreciated Loui

From: Rico Mandes

Try using .....void sleep(int); function.e.g sleep(1);the argument is in seconds.

From: ornaz

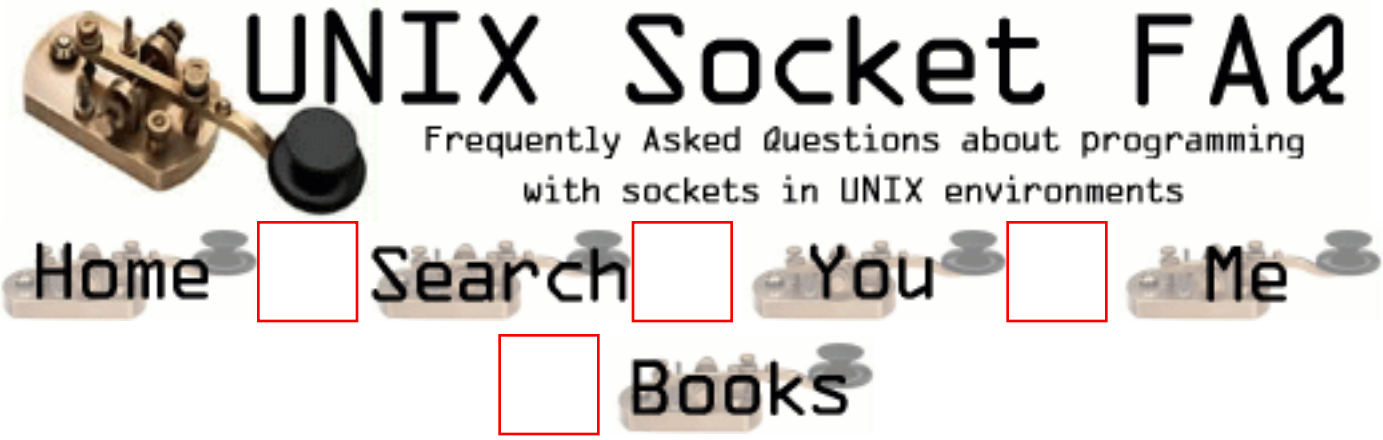
if u wish to do delay in micro seconds do the following

#define DELAY 0.005
#define USEC_PER_SEC 1000000

struct timeval tv;
....

```
 tv.tv_sec = 0;
 tv.tv_usec = DELAY * USEC_PER_SEC;

select(0,NULL,NULL,NULL,&tv);
```

Add a Comment



New Questions: psuedo terminals

From: Barel Gil

I need to write a program that illustrates the login process
under TCP/IP on Red Hat 6.1

A client does telnet server port and the server program on
the server, which listens to this port opens a connection.

I need to verify the username and password of the user trying
to use the server.

In unix this is done by allocating a psuedo terminal,
connecting it to the remote client and running by exec the
login program.

How can you connect it through the socket ?
Do you have a sample code for that ? if you do, I would
appreciate it if you could send it to me.

Thank you.

Add a Comment

UNIX Socket FAQ

Frequently Asked Questions about programming with sockets in UNIX environments

New Questions: transmit files via socket

From: ezzete

so far i had succesfully transmit integer and character throug socket using send and recv command. Can anyone tell me how to transmit files using send and recv command or is there any way else to do this?

From: David McReynolds

You'll have to define what type of file you wish to transmitt (ascii or binary). Are you planning on using your transmission technique in a heterogenous computing environment? Or, will the sender and the receiver be the same type of platform? Why not just use FTP?

From: tybins

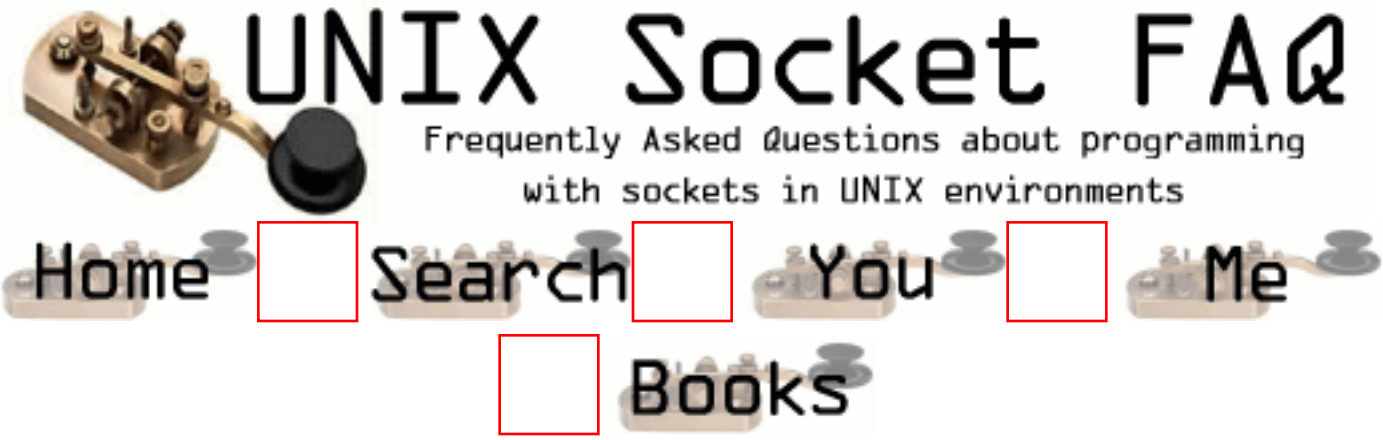
I need to exchange binary files between clients and servers, I Would like to know if it would be possible to invoke and use the ftp program from my server and client code, and just let ftp handle this work without having to reinvent the wheel.

From: Alisha

I would also like to know how to transfer entire files from my server to my client and also once I transfer files to my client how I would store the entire file with the associated file name in my current working directory.

Alisha

Add a Comment



New Questions: Error bind

From: ezzete

I'd succesfully run my simple email system on Unix but i got
error bind when i run it on Linux.
Besides i got warning: passing `int *' as argument 3 of
`accept(int,sockaddr *,socklen_t *)' changes signedness after
compiling the program

I call the accept function as following

struct sockaddr sa;
int sz;
sz = sizeof(sa)

newsockfd = accept(ssock,(struct sockaddr *)&sa,&sz);

From: S.M.Krish

Hai ezeete
Why are you finding sizeof() and all. you simply pass
any integer address only.
just as below

int sz;
struct sockaddr_in addr;
newfd = accept(sockfd,(struct sockaddr *)&addr,&sz);
In my Linux Machine its coming with out any error.
On return The structure will be filled with the client
address.

Thanks and bye.

From: Rob Seace

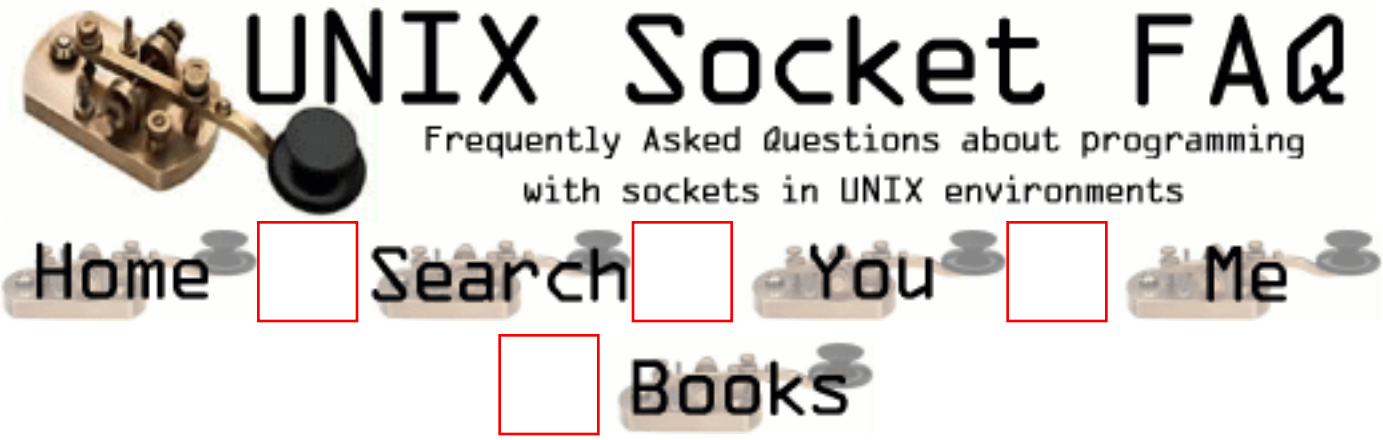
No, that's not true... The third arg to accept() is a standard value-result parameter... Ie: you must initially set it to the max length available for storage in the sockaddr pointed to by the second arg; and, on return, it will be set to the actual length filled in (no more than the given max)... By not setting the value before calling accept(), you pass in random junk off the stack (whatever value the automatic variable happens to be set to), which is used as the max length of your sockaddr... In practice, most of the time, that'll probably just happen to work, as the odds favor the random junk being greater than or equal to sizeof(struct sockaddr_in)... But, it's not something you should rely on, certainly... (And, pray you never encounter an IPv6 addr, or anything larger than sockaddr_in, or else you'll overflow, and trash memory...)

No, I'd say the first person's problem mainly looks to be the fact that they are using just a plain "struct sockaddr" rather than "struct sockaddr_in" (or "sockaddr_in6", or whatever they really wanted)... You should never use a real "struct sockaddr" for any actual storage/use; it's just a generic placeholder, whose only real purpose is to allow all the real sockaddrs to be casted to it, for passing around to socket functions...

Actually, re-reading the original problem, you say it's an error from bind(), not accept()? So, where is your call to bind(), then? That sounds like what's at fault, not the accept(), anyway...

And, I don't know why you are getting an error/warning from the compiler about int vs. socklen_t... I do that all the time without warning... But, then, I use a very specific set of GCC switches to adjust warning level, as well... GCC tends to be a bit anal about warnings, sometimes... ;-)

Add a Comment



New Questions: How can I check the evolution of a non blocking write on a socket?

From: Irina

I want to do a write on a non blocking socket, and later
on in the program check if the data (or how much data) has
been copied from the application's buffer to the socket's
buffer. How could I do this?

Thanks.

From: Kieran

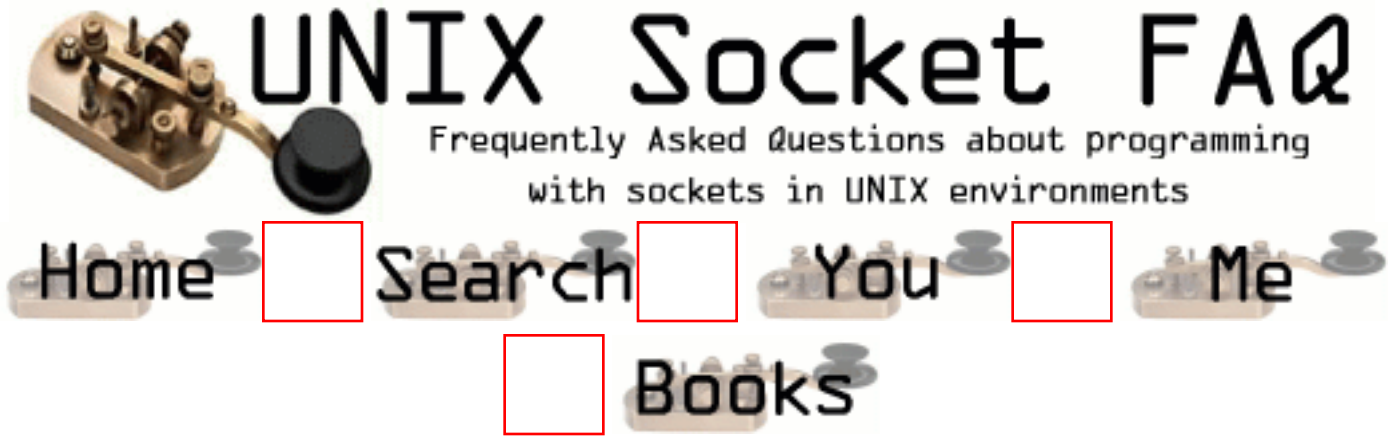
AFAIK, all "write"s on sockets are non-blocking, in the sense that
they never wait for confirmation from the remote end.

When a write/send/sendto on a socket returns, the data has
already been copied into the socket's buffer.

From: igor

if in client you call closesocket on blocking socket right after calling send it is not guaranteed that the recv call on the other peer succeeds.

Add a Comment



New Questions: discard socket data

From: Beli

---

30 10.55496 193.152.37.57 -> SSMRRO5 ETHER Type=0800 (IP), size = 61 bytes
30 10.55496 193.152.37.57 -> SSMRRO5 IP D=195.235.104.13 S=193.152.37.57 LEN=47, ID=19716
30 10.55496 193.152.37.57 -> SSMRRO5 TCP D=7047 S=15357 Ack=1583335949 Seq=2845681306 Len=7 Win=4380

0: b1b1 b1b1 b1b1 ffffffb1 .....

ASCEND MANDA 1 1 1 1 1 1 1

---

31 10.58554 193.152.37.57 -> SSMRRO5 ETHER Type=0800 (IP), size = 61 bytes
31 10.58554 193.152.37.57 -> SSMRRO5 IP D=195.235.104.13 S=193.152.37.57 LEN=47, ID=19972
31 10.58554 193.152.37.57 -> SSMRRO5 TCP D=7047 S=15357 Ack=1583335949 Seq=2845681313 Len=7 Win=4380

0: b1b1 b1b1 b1b1 ffffffb1 .....

ASCEND MANDA 1 1 1 1 1 1 1

---

32 10.58557 SSMRRO5 -> 193.152.37.57 ETHER Type=0800 (IP), size = 54 bytes
32 10.58557 SSMRRO5 -> 193.152.37.57 IP D=193.152.37.57 S=195.235.104.13 LEN=40, ID=22549
32 10.58557 SSMRRO5 -> 193.152.37.57 TCP D=15357 S=7047

Ack=2845681320 Seq=1583335949 Len=0 Win=9112

---

33 10.61476 193.152.37.57 -> SSMRRO5 ETHER Type=0800 (IP), size = 61 bytes
33 10.61476 193.152.37.57 -> SSMRRO5 IP D=195.235.104.13 S=193.152.37.57 LEN=47, ID=20228
33 10.61476 193.152.37.57 -> SSMRRO5 TCP D=7047 S=15357 Ack=1583335949 Seq=2845681320 Len=7 Win=4380

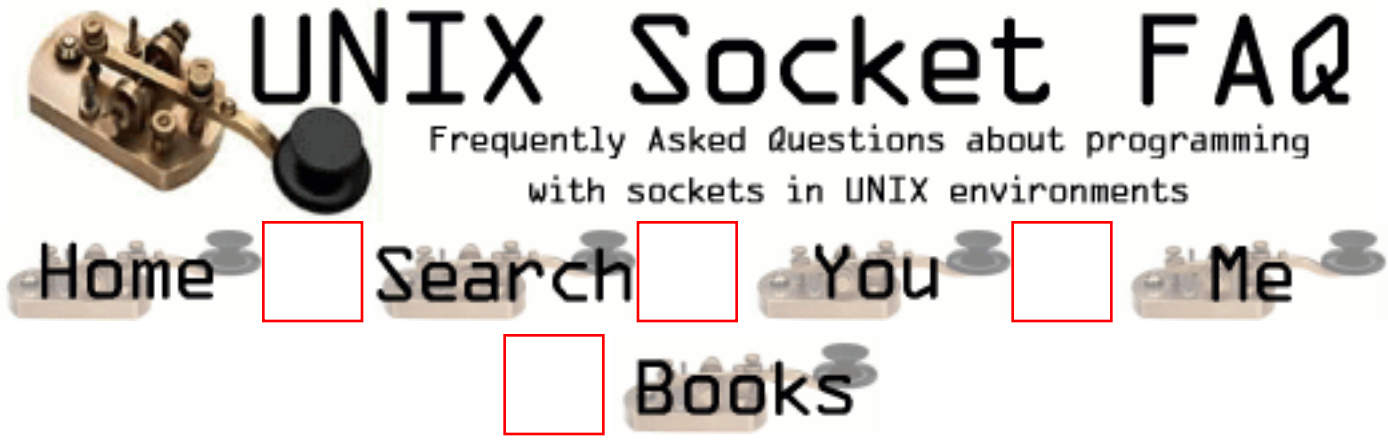
0: b1b1 b1b1 b1b1 ffffff8d .....

ASCEND MANDA 1 1 1 1 1 1 CR

From: Beli

Sorry this question is wrong

Add a Comment



New Questions: discard socket data

From: Beli

I have traced the next sequence in the machine named SSMRRO5

---

30 10.55496 193.152.37.57 -> SSMRRO5 ETHER Type=0800 (IP), size = 61 bytes
30 10.55496 193.152.37.57 -> SSMRRO5 IP D=195.235.104.13 S=193.152.37.57 LEN=47, ID=19716
30 10.55496 193.152.37.57 -> SSMRRO5 TCP D=7047 S=15357 Ack=1583335949 Seq=2845681306 Len=7 Win=4380

0: b1b1 b1b1 b1b1 ffffffb1 .....

---

31 10.58554 193.152.37.57 -> SSMRRO5 ETHER Type=0800 (IP), size = 61 bytes
31 10.58554 193.152.37.57 -> SSMRRO5 IP D=195.235.104.13 S=193.152.37.57 LEN=47, ID=19972
31 10.58554 193.152.37.57 -> SSMRRO5 TCP D=7047 S=15357 Ack=1583335949 Seq=2845681313 Len=7 Win=4380

0: b1b1 b1b1 b1b1 ffffffb1 .....

---

32 10.58557 SSMRRO5 -> 193.152.37.57 ETHER Type=0800 (IP), size = 54 bytes
32 10.58557 SSMRRO5 -> 193.152.37.57 IP D=193.152.37.57 S=195.235.104.13 LEN=40, ID=22549
32 10.58557 SSMRRO5 -> 193.152.37.57 TCP D=15357 S=7047 Ack=2845681320 Seq=1583335949 Len=0 Win=9112

33 10.61476 193.152.37.57 -> SSMRRO5 ETHER Type=0800 (IP), size = 61 bytes
33 10.61476 193.152.37.57 -> SSMRRO5 IP D=195.235.104.13 S=193.152.37.57 LEN=47, ID=20228
33 10.61476 193.152.37.57 -> SSMRRO5 TCP D=7047 S=15357 Ack=1583335949 Seq=2845681320 Len=7 Win=4380

0: b1b1 b1b1 b1b1 ffffff8d .....

When I read this characters I can't read it all with one recv, I have to do it three times, firs time I read

packet 30 b1b1 b1b1 b1b1 ffffffb1

Second time:

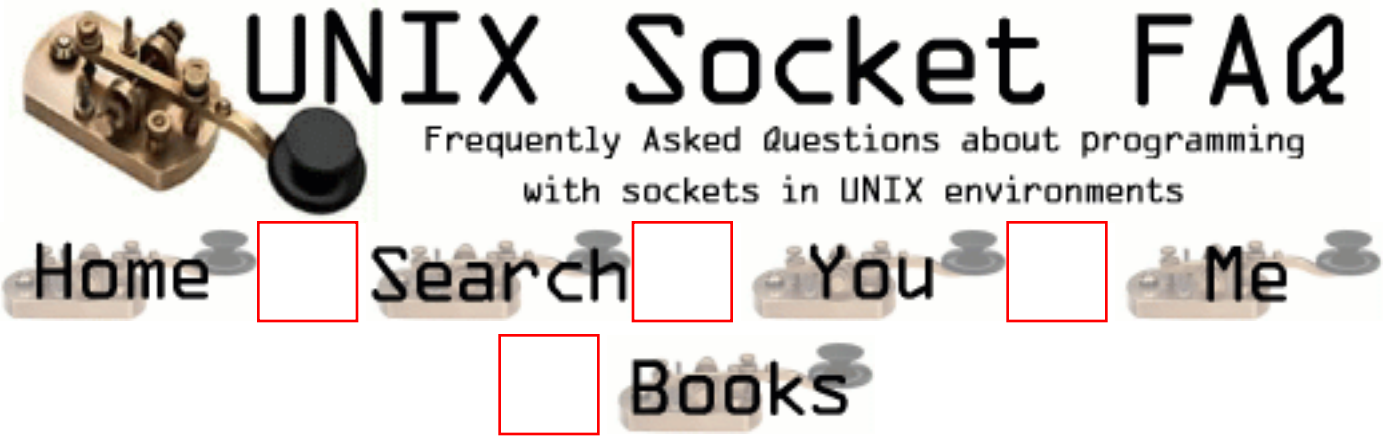
packet 31 b1b1 b1b1 b1b1 ffffffb1

and the third:

packet 32 b1b1 b1b1 b1b1 ffffff8d

I would like to read all with only one recv, would I?

Add a Comment



New Questions: Starting server when disconnected from ethernet fails to run

From: Andy

I have written a WINSOCK based server that runs infallably except under one set of circumstances:
1) The server is running
2) The ethernet cable is pulled
3) The server is shut down and re-started
4) The cable is plugged in.

While disconnected, the server gets up and running and establishes a socket in the listening state (with a blocking select call). After plugging the cable back in, when I try to connect to it, I get WSAError 10061 (connection forcefully rejected). Running the server under any other circumstances, a connect causes select to return so that a connection can be accepted (expected behavior).

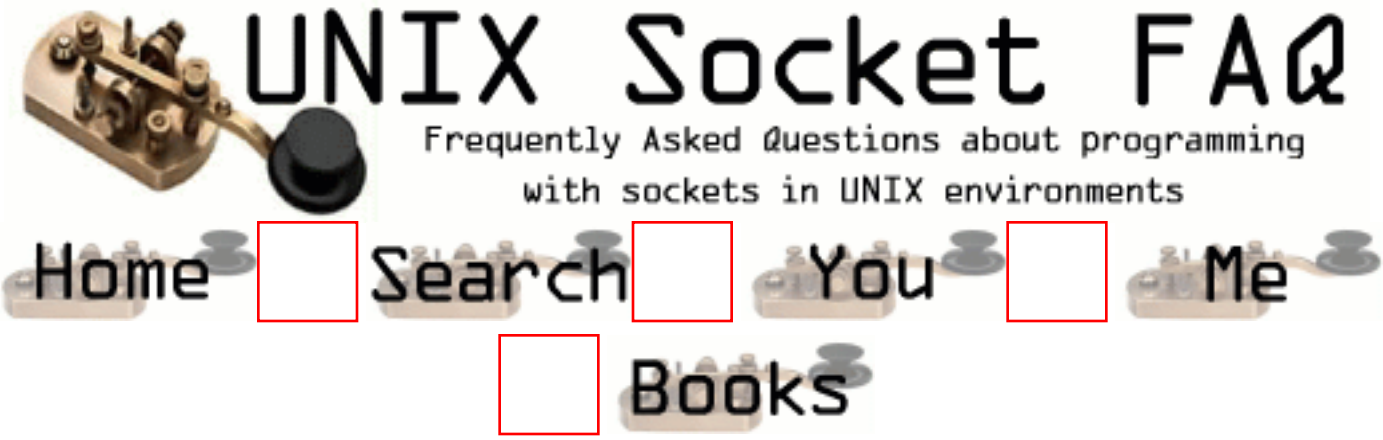
Any thoughts on this wierd problem?? Thanks in advance.

Andy

From:

Sorry, actually the WINSOCK error 10061 is ConnRefused, not ConnForcefullyRejected. But the problem persists.

Add a Comment

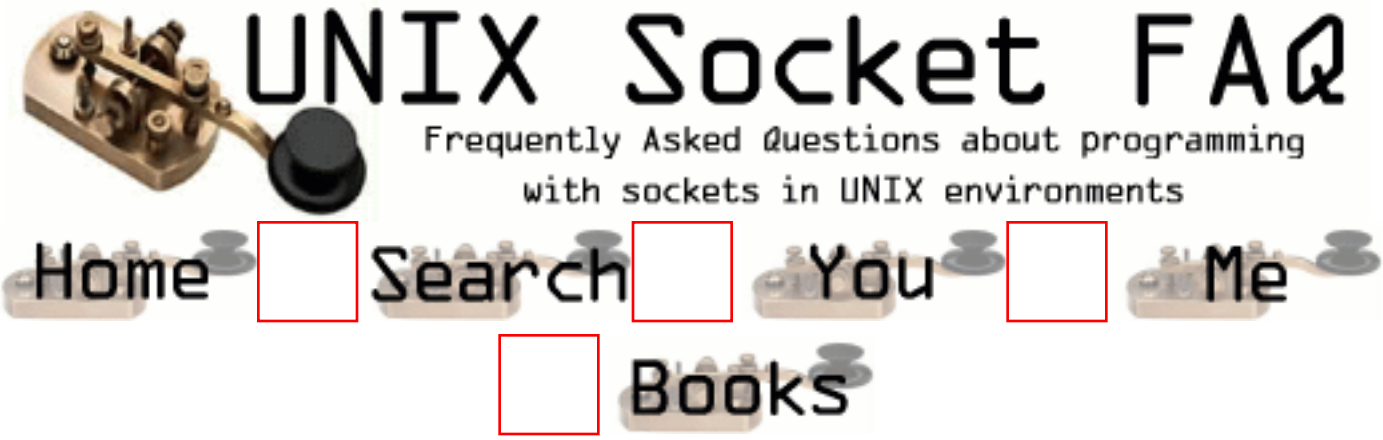


New Questions: packets lost using SOCK_STREAM

From: David

what should I do to get all the packets? here is the client: #include #include #include #include #include #include #include main(int argc, char *argv[]) { int rc, s, i; struct hostent *hen; struct sockaddr_in sa; char str[1024]; hen=gethostbyname("ai-way.local"); if (NULL==hen) { perror("hostname error\n"); return 0; } memset(&sa, 0, sizeof(sa)); sa.sin_port=htons(1200); sa.sin_family=AF_INET; memcpy(&sa.sin_addr.s_addr, hen->h_addr_list[0], hen->h_length); s=socket(AF_INET, SOCK_STREAM, 0); if (-1==s) { perror("socket open error\n"); return 0; } rc=connect(s, &sa, sizeof(sa)); if (-1==rc) { perror("connect error\n"); return 0; } for (i=1; argc>i; i++) { sprintf(str, "%d\t%s\n", i, argv[i]); rc=write(s, str, strlen(str)+1); printf("%d\t>>\t", rc); printf("%s", str); } close(s); return 0; } and here is the server: #include #include #include #include #include #include main() { int rc, s, csa_size, cs; struct sockaddr_in sa, csa; char str[1024]; memset(&sa, 0, sizeof(sa)); sa.sin_port=htons(1200); sa.sin_family=AF_INET; /* sa.sin_addr.s_addr=inet_addr("127.0.0.1"); */ sa.sin_addr.s_addr=INADDR_ANY; s=socket(AF_INET, SOCK_STREAM, 0); if (-1==s) { perror("socket error\n"); return 1; } rc=bind(s, &sa, sizeof(sa)); if (-1==rc) { perror("bind error\n"); return 1; } csa_size=sizeof(csa); rc=listen(s, 5); if (-1==rc) { perror("listen error\n"); return 1; } while (1) { cs=accept(s, &csa, &csa_size); if (cs<0) continue; while (0<(rc=read(cs, str, 1024))) { printf("%s", str); } printf("%d\n", rc); close(cs); } return 0; }

Add a Comment

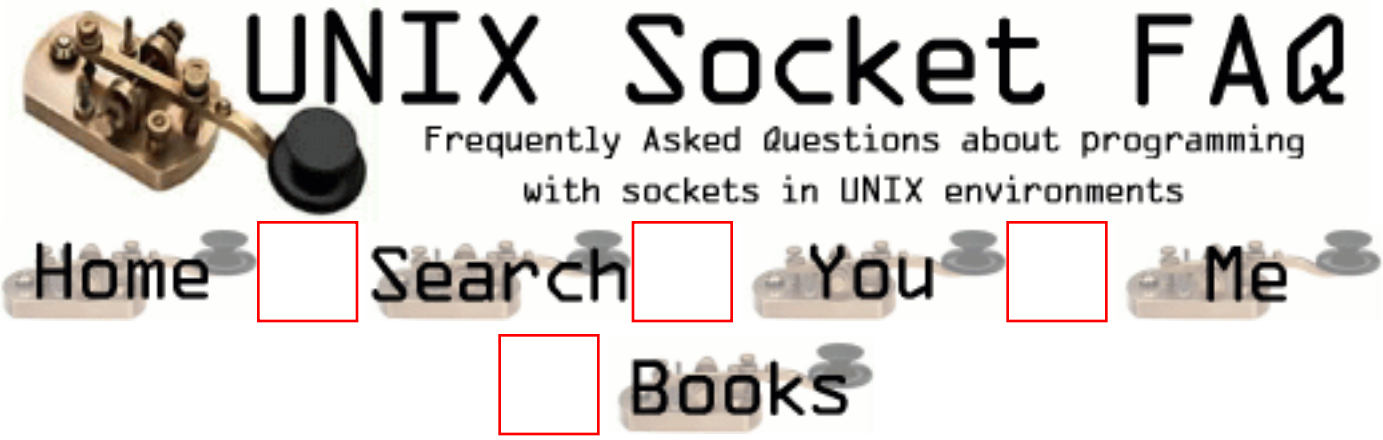


New Questions: packets lost using SOCK_STREAM

From: David

what should I do to get all the packets? here is the client: #include #include #include #include #include #include #include main(int argc, char *argv[]) { int rc, s, i; struct hostent *hen; struct sockaddr_in sa; char str[1024]; hen=gethostbyname("ai-way.local"); if (NULL==hen) { perror("hostname error\n"); return 0; } memset(&sa, 0, sizeof(sa)); sa.sin_port=htons(1200); sa.sin_family=AF_INET; memcpy(&sa.sin_addr.s_addr, hen->h_addr_list[0], hen->h_length); s=socket(AF_INET, SOCK_STREAM, 0); if (-1==s) { perror("socket open error\n"); return 0; } rc=connect(s, &sa, sizeof(sa)); if (-1==rc) { perror("connect error\n"); return 0; } for (i=1; argc>i; i++) { sprintf(str, "%d\t%s\n", i, argv[i]); rc=write(s, str, strlen(str)+1); printf("%d\t>>\t", rc); printf("%s", str); } close(s); return 0; } and here is the server: #include #include #include #include #include #include main() { int rc, s, csa_size, cs; struct sockaddr_in sa, csa; char str[1024]; memset(&sa, 0, sizeof(sa)); sa.sin_port=htons(1200); sa.sin_family=AF_INET; /* sa.sin_addr.s_addr=inet_addr("127.0.0.1"); */ sa.sin_addr.s_addr=INADDR_ANY; s=socket(AF_INET, SOCK_STREAM, 0); if (-1==s) { perror("socket error\n"); return 1; } rc=bind(s, &sa, sizeof(sa)); if (-1==rc) { perror("bind error\n"); return 1; } csa_size=sizeof(csa); rc=listen(s, 5); if (-1==rc) { perror("listen error\n"); return 1; } while (1) { cs=accept(s, &csa, &csa_size); if (cs<0) continue; while (0<(rc=read(cs, str, 1024))) { printf("%s", str); } printf("%d\n", rc); close(cs); } return 0; }

Add a Comment



New Questions: using select to check if the second end is alive

From: Helena

Sorry, forgot to fill the subject before... I'm trying to check the avaliability of the second end socket with the select(), but even when the second end socket disconnected, the select() indicates as ready to read/write... How can I check that the second end (whatever client or server) was disconnected?
I must say that I read almost all the FAQs here, they are realy helpfull, but anyhow I'm not succeeding in this simple task.

From: Rob Seace

See Question 1, Section 2 and Question 13, Section 2... Basically, you really need to try to read(), and get the 0 return in order to know for sure... (Unless you happen to know that no data should be getting sent by the remote end at this time, in which case you can just select() for readability, and if it becomes readable, then you know it must be the EOF/FIN waiting to be read...)

From: Rajiv

you can try this,
if you are using blocking sockets then perform
a recv() with MSG_PEEK option. if its returns 0, then
you have received a FIN and that indicated that the other
end closed the connection,or the peer process crashed. < 0 (-1) indicates error that is possible because of several reasons (ETIMED_OUT, EHOST_UNREACH etc.). > 1 i.e. above the low-watermark threshold indicates that there is active data to be read from the socket

for non-blocking sockets, the treatment is more or less the same.

From: Senthil

I am having a thread A. Doing a fork inside it. I am
running an executable in the child process using 'execvp'.
Now creating a socket in the parent and communication
happens between the child executable and the parent.
  The problem in this is that after the communication is
over, the socket is not getting closed even though i am doing
it properly in the code and the program hangs.
What could be the reason ?

Debugger shows me that '_poll' does not return.
What is wrong in my approach ?

Add a Comment

UNIX Socket FAQ

Frequently Asked Questions about programming with sockets in UNIX environments

New Questions: package loss and delay

From: Sam

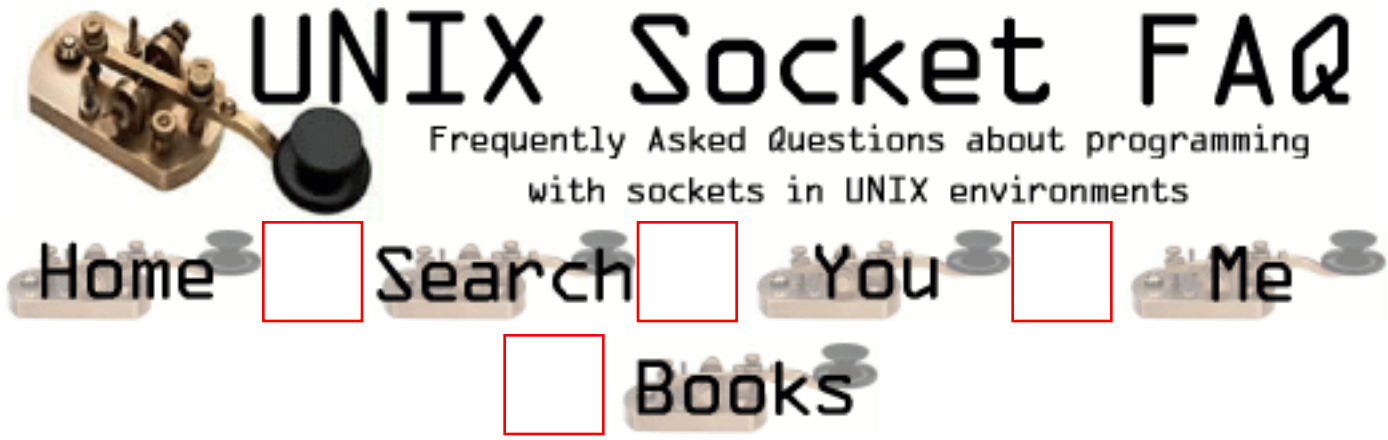
Hi all
I was wandering when we get package loss and delay in a network ? what is the reason for that ? are there any information abut these topic available in the net ?

thanks in advance Sam

From: wilson

hi,if you are working on a very crowded network.with a lot of communication works to do.then you may get collision error.and lose your package sometime.

Add a Comment



New Questions: Sockets with Unix

From: Sylvain Nadeau

I am new to programming with sockets.

I want to write an application (in Visual Basic if possible) that will interface with a Unix system.

I want to send and receive files, execute scripts on the unix server, get the contents of the stdout of the script and display it on the client station.

I want to do all this with a minimun of opened connection between the client and the server.

I don't want to open a connection and close it with each file transfer or script execution because there will by many developpers that will use my application.

I have tried to use wsock32.dll and mswinsck.ocx but I don't know how to use them properly.

Can you help me ?

Do you have code samples that could help me build my application (Visual Basic if possible, else C/C++ will do) ?

Do you know about a good reference book that could help me ?

Thanks

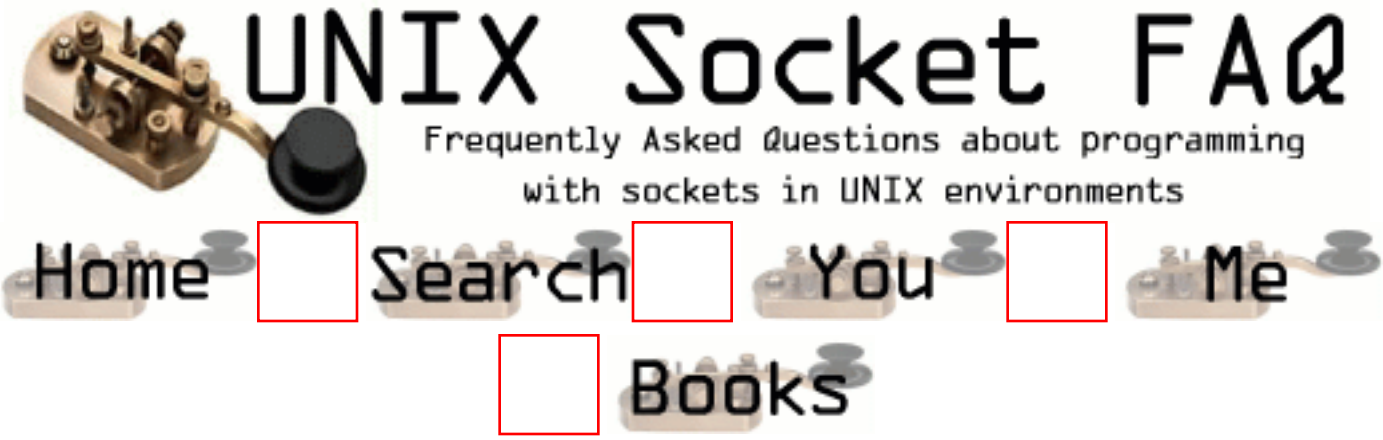
From: gates Bill

it's impossible to do that because there is no interraction
between Microsoft visual basic and unix system but you can
use of course visual c++

From: Mike

Can anybody tell me why a TCP socket number can not be immediately reused?

Add a Comment



New Questions: Number of Ports? Newbie question

From: Agent

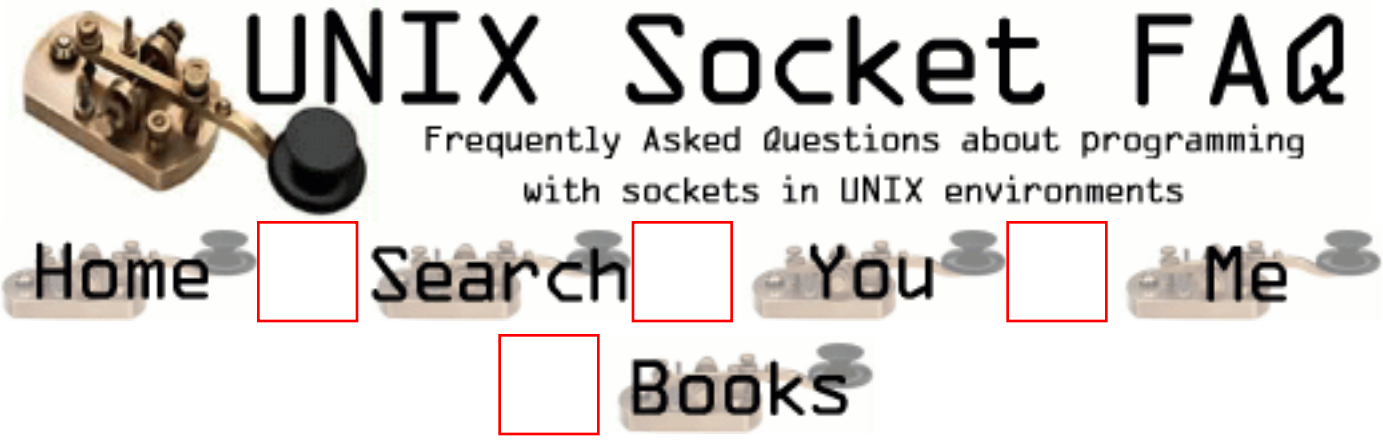
hi,
Is there an upper limit to the number of ports available?

regards

From: Rob Seace

Well, there's a theoretical limit of 64K different port#'s, since they're stored in an unsigned short... But, in practice, you're likely to run into OS open file descriptor limits before you ever manage to bind sockets to all 64K port#'s, anyway...

Add a Comment

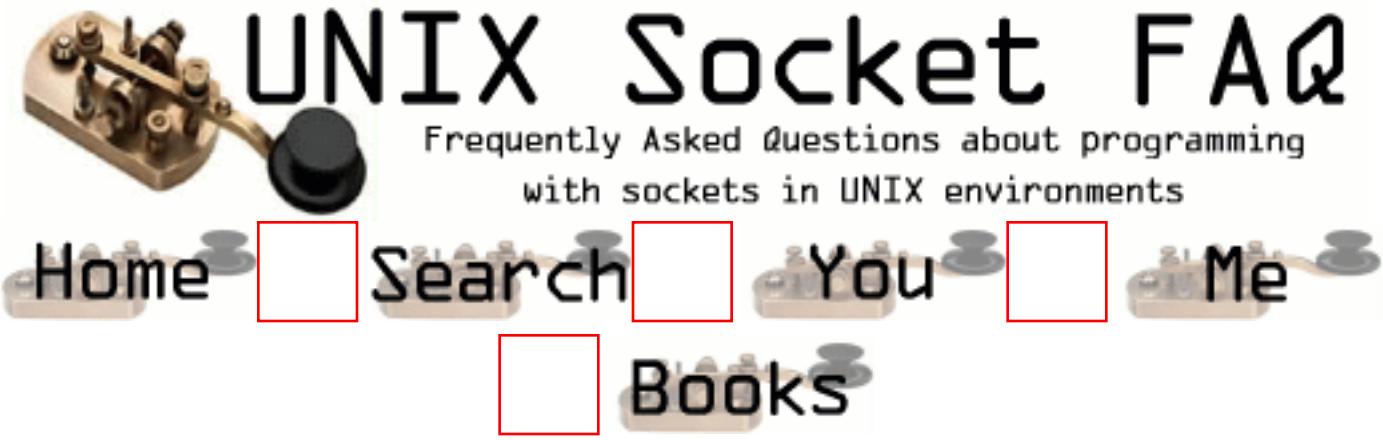


New Questions: binding multiple file descriptors to a single port

From: Brent

Are there any issues I should be concerned about in binding two file descriptors to the same port?

Add a Comment



New Questions: recv() on unconnected socket ?

From: AK

Hi,

What does recv() return when used for a socket that is not connected? Does it return 0 or <0.
If <0, then what is the errno that it sets ?

TIA.

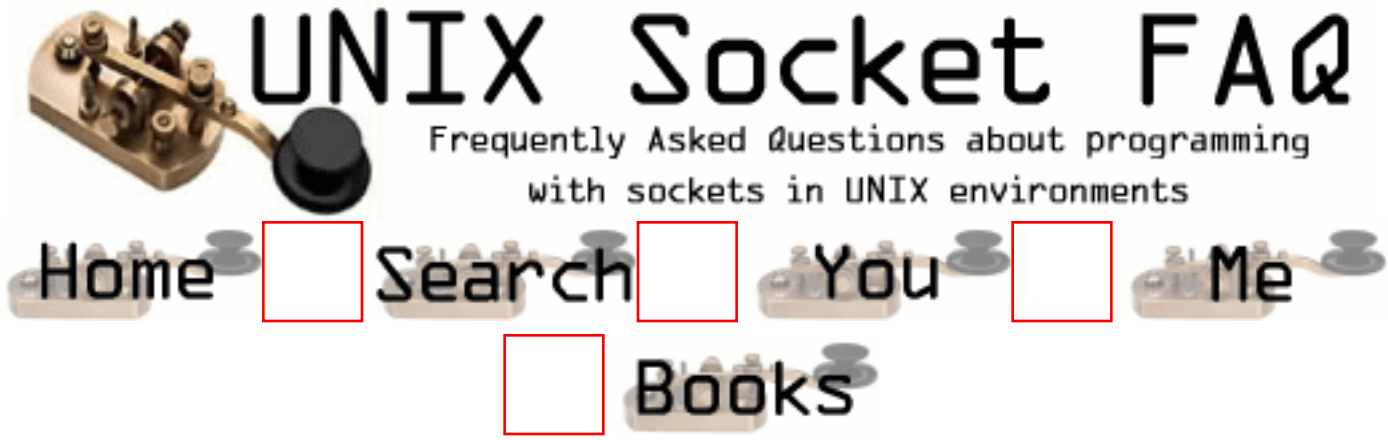
From: Arthur Lung

Assuming that it's a TCP socket:

recv returns -1 to signal an error, and errno is set to ENOTCONN.

It took about 2 1/2 second to look that up in the manual.
Perhaps you need a copy of the socket api documentation
in some language other than english to help you find such
mind numbingly obvious answers?

Add a Comment



New Questions: Sockets, timeout, signals

From: Gejzir

Hi all!

I've written a complex data distribution system using Linux servers and both Win32 and unix clients, and Stream sockets.

Everything works fine, except one thing.
My server automatically can recognise when a client disconnects or the client connection is broken, except when the client computer itself goes down.(someone pushes reset, or disconnects from the power supply)

I've found a following solution for this problem:
I've written a signal handler function, wich receives a signal after a broken connection's timeout period.

When a client connection goes into broken state, I can see, that the timer for the socket is on, and counting.
(netstat -to)
After a counter reaches a value (where can I set the limit?), my application receives a signal, and terminates.

BUT. This is working fine on a 4-5 Linux machine I've tested, but is NOT working on the final machines of the customer. The counter starts, I can see it in netstat, but it counts forever, my application just doesn't get a signal.

I've checked that the kernel versions are the same, the .so libraries are the same. I've checked it with many kernel versions. On the most machines this is working well, but on the others it doesn't.

Questions:

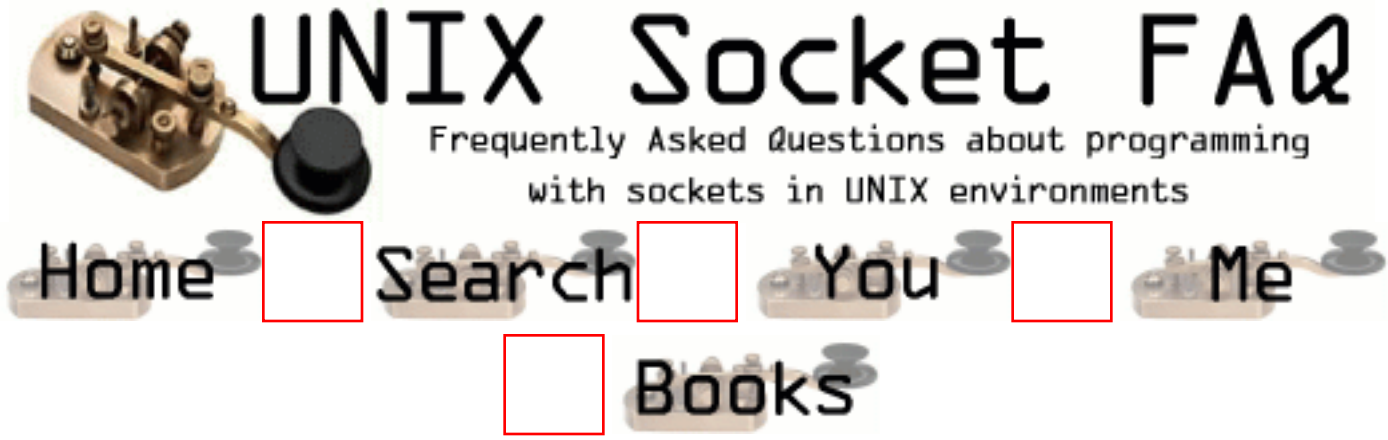
1. How can I set the value of the timeout period when a connection brakes.
2. What could be the difference beetween the two configuration?

Regards: Gejzir

From: Shawn Elliott

I had a similar problem. What I did was write a cleanup function that ran through all of the Socket Descriptors and sends a * through. If the write gives me an error then the socket is closed and I proceed to kill the Threaded process.

Add a Comment



New Questions: Port in use question?

From: Paul Chafin

On UNIX (HP-UX 10.2 specifically) I have a TCP server program
that binds a socket to a port and waits for incoming connections.

If we stop the server program and restart it we get the message "Can't bind
socket port already in use".

Question how to disconnect the client connections from this port?

From: Andreas Forsgren

Try setting SO_REUSEADDR with the setsockopt() function.

From: Data64

Look up SO_LINGER. Depending on your application, you may be able to set it to false.

From: Muslim Moiz Ghadiali

kill the client first then kill the server...disable server if port in use by first saying 'ps' in unix and 'kill ##' commands.

From: Sudhakar

in case that port no is already taken up by other process, how to determine which is using port. Is there any unix command to identify the process

---

From: nitin

u can do 'netstat -an |grep <port #>

From: nitin

u can do 'netstat -an |grep <port #>

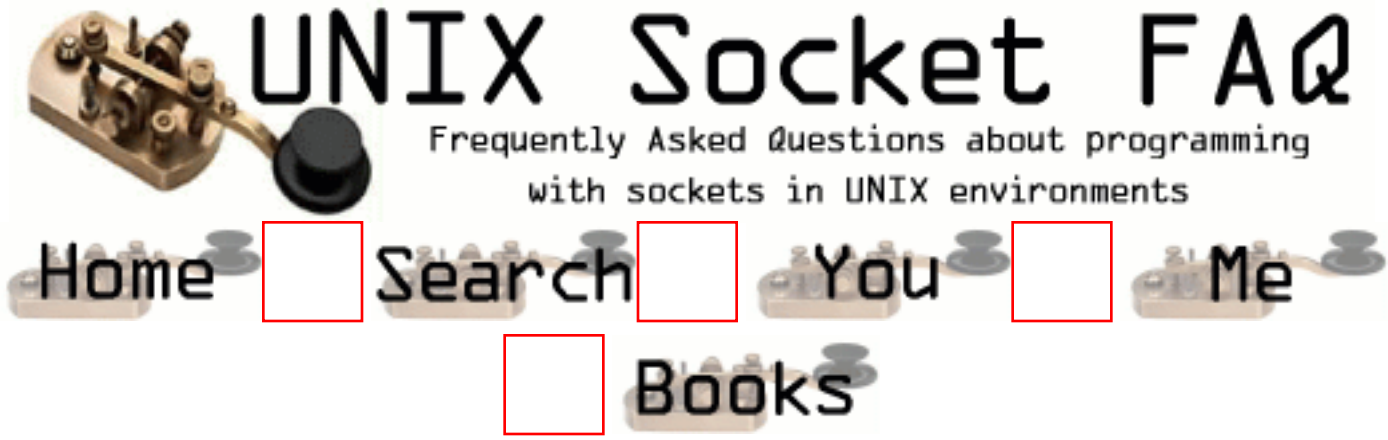
From: jdev

Use lsof command .it will give u the process which is using the specifed port.

From: Suresh

Hai Paul,
U can see the solution in the question number 11
"how to restart a server immediately"
That worked out for me.
Try.
Suresh

Add a Comment



New Questions: Accept call on unconnected socket

From: Beth Rourke

As the server, in a client-server code setup, I want to test for a connection
from the client. If the client is not ready to connect, go on and do something else
my problem is that I do not want to wait for the connection to be made, which is what the accept does.
When I try to do non-blocking the client side fails when if finally trys to connect.
I have also tried using the select call and run into the same problem. If I start the server code, and
start the client code when I get to the accept call, everything behaves nicely. I have looked pretty much
everywhere for a solution to no avail. Any suggestions? I am working on SUN ultra-sparcs running Solaris.

From: Kieran

This should work (pseudo-code):

server = socket()

/* set into non-blocking mode with call for your system
  (fncl or ioctl or ioctlsocket) */

listen(server, 1);

From: Kieran

Whoops, posted that a bit too soon.

<continuation>
FD_SET(server, &readfds);

```
timeval.secs = timeval.usecs = 0; // check struct member names!
select(1, &readfds, NULL, NULL, timeval);
```

If select fails initially, call it periodically in
between doing your "other stuff". select()ing for a read
on a server socket allows you to be reasonably confident
accept() will succeed.

I suspect what you were missing was the bind() and/or the
listen() call; some accept() calls do a bind() and
listen() implicitly.

From: Kieran

Speaking of which, my first code should have a bind() in
there...and forget what I said about implicit bind()s -
wouldn't make much sense for a server.

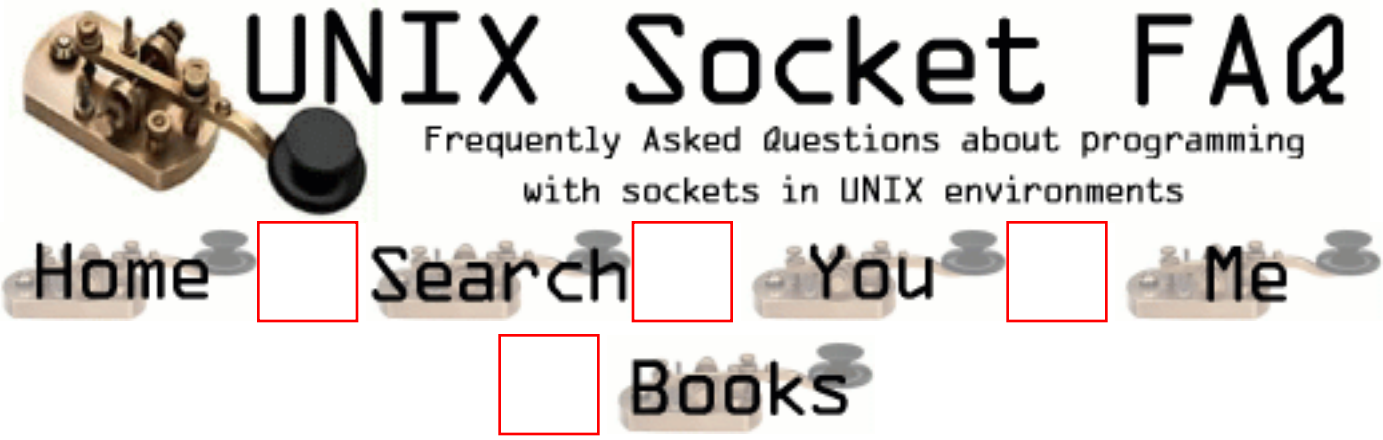
For clarity here is what I should have posted originally:

```
fd_set readfds;
struct timeval timer;
server = socket(...);

bind(server, ...); // specify the port in the sockaddr struct

listen(server, 1); // crucial if clients are to connect()

timer.secs = timer.usecs = 0;

FD_ZERO(&readfds);
FD_SET(server, &readfds);

if (select(1, &readfds, NULL, NULL, timer)) {
   connection = accept();
} else {
  // do some other stuff and try again later;
}
```

From: Magnus Boden

should the first argument to select be the value of the server sockfd+1 in you example?

Add a Comment



New Questions: Send & Receive using the same port

From: Michele Piccini

I use the same port and the same socket to send & receive, i wait for input with a bocking select(socket,fdread,NULL,NULL,timeout) but whenever i write to the socket the select wait for input returns.
How can read and write to the same port without having my select() returned whenever i write?

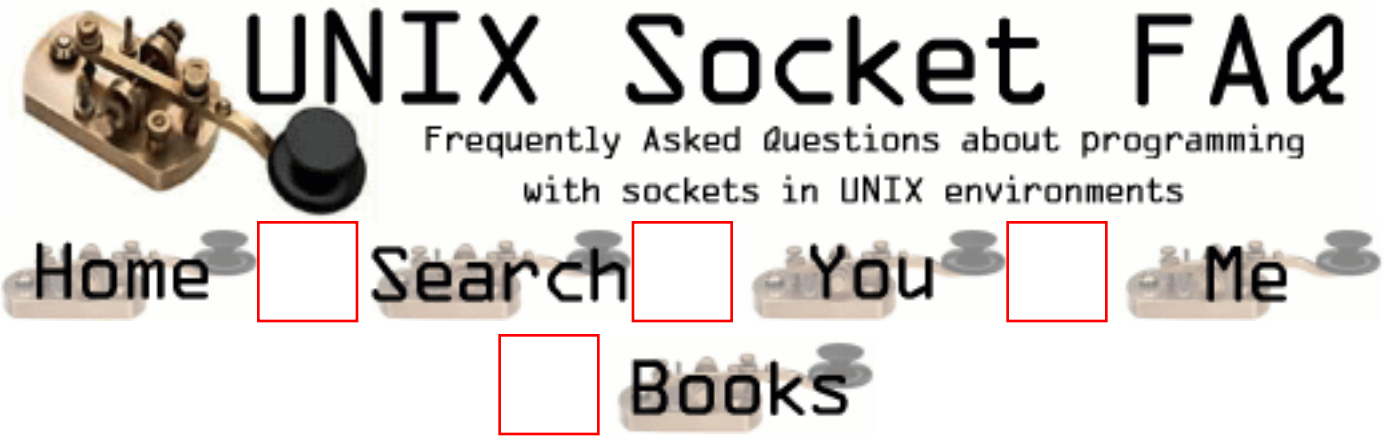
From: Eric J Krejci

I am having the same problem i think. I block & then read
in one thread, and write from another. And if i send data fast enough the thing segfaults.
I can supply code sample if you send email.
Any ideas ?
Thanks Eric

From: tony zalatan

I am trying to create a C program that will form a loopback
using either one serial port or two serial ports. I am having
trouble reading consistangly what I am writing. any clues
would be appreciated....

Add a Comment



New Questions: Ports in use

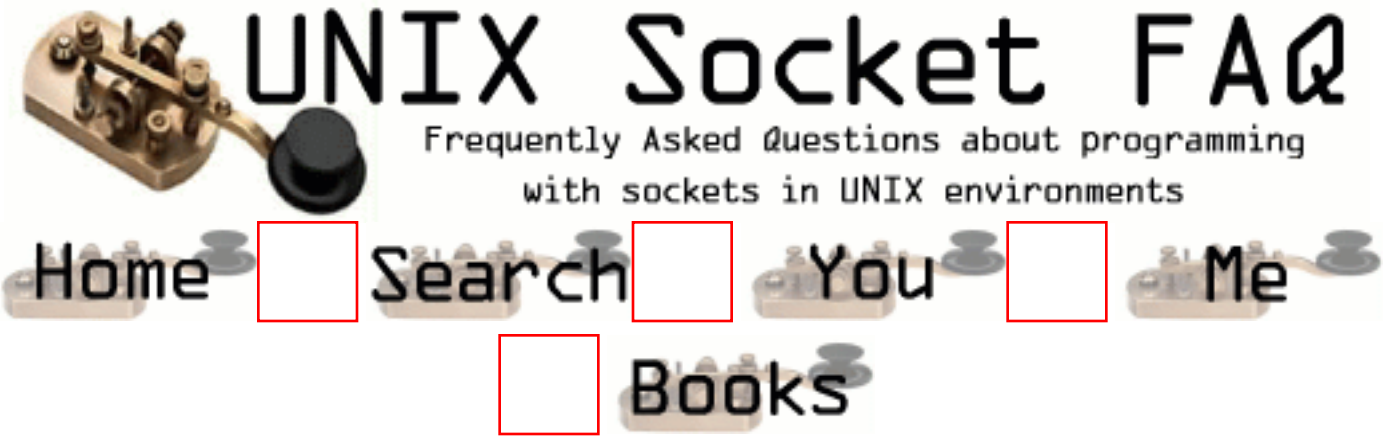
From: Valentin Chartier

How do I know which socket ports are in use on a Unix statin ?
How do I know by which process a socket port is in use ?
Thanks

From: Rob Seace

Try "netstat -a" to show you the ports in use (add "-n" for raw numeric ports, rather than service names)... Or, alternatively, you could use a port scanner like "nmap"... (Which can show you the ports in use not just on the local host, but on remote hosts as well...) As for what processes are bound to those ports, how to get at that info depends greatly on your OS... I can tell you how to do so on Linux or QNX; other than that, I can't help you... On Linux, "netstat" has a "-p" switch to list the programs using each listed port... (Or, alternatively, you can use something like "lsof", or manually dig through "/proc" and find the data yourself... Actually, the "lsof" idea will probably be useful on many other Unix-alikes, as well; I believe that's widely ported...) On QNX, a "sin fd" should give you the info...

Add a Comment



New Questions: Newbie-Question:How to start with a proxy server?

From: Joerg

Hello everybody!

I am quite new to socket programming but not new to (V)C++ and I am trying to program an app, which (should)check, which URL a user typed in into any webbrowser on the system. I want to do this, because, I want to block certain websites and make them not viewable for the user. I soon figured out, I need to program a proxy server but I have no idea how to do that. Is there something on the web, you can recommedn as a turorial for proxy programming or just give me an idea how to set a proxy up (not detailed of course :)).

Any help is apreciated!
Thanks in advance!
Bye,
joerg

From: Shawn Elliott

I'd use a FindTest API on the Window of the browser. Find what site they are on that way. Alot easier

---

From: Ram

hi,
i would like ask the question as that of joerg.if anyone has tips.please respond

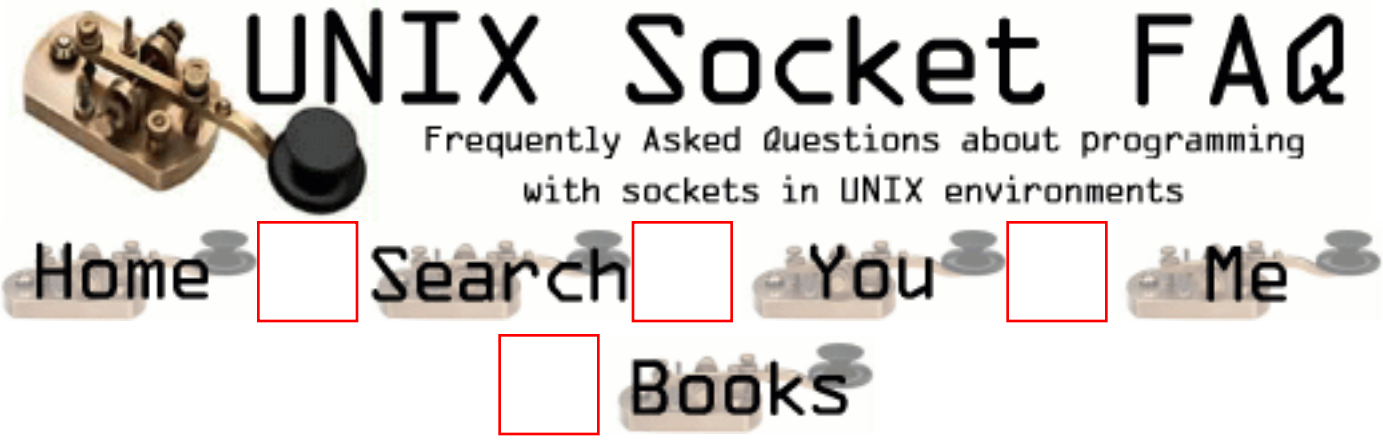
---

From: Mukhben Singh

Download squid caching proxy server from http://squid.nlanr.net/ and follow the instructions. It is the best and most popular proxy server around. You will have to configure it on a machine which has direct net access and block other machines from using the net directly at ur firewall.

I hope it helps

Add a Comment



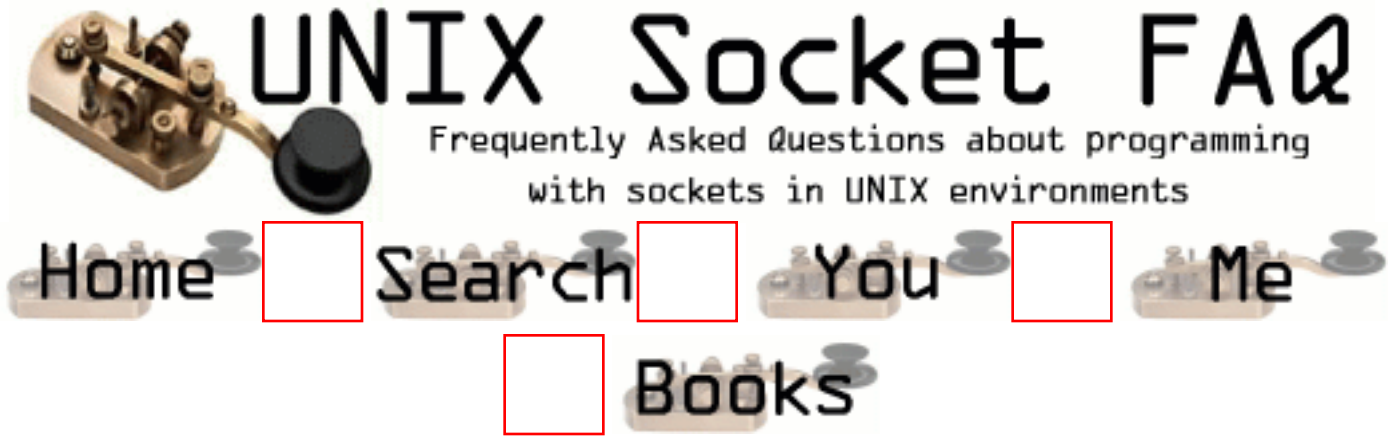
New Questions: Why my ping fails?

From: sandman

that's not realy a socket question, but maybe somebody knows the answer. i wrote a ping prog (icmp socket). in the id field of the icmp header i write some id (let's say 14). when the answer arrives the id field is zero, but it should be 14. i just don't know, whats wrong!

sandman

Add a Comment



New Questions: Final result of setting Timeout for connect() in socket

From: Ramzan Ul Haq

Then what would be the final thing to set the timeout for connect() call in socket programming? Please do answer clearly, with code preferably.
Thanks

From: Mukhben Singh

u can use SIGALRM signal. set an alarm with alarm function. set the signal handler before the call to conenct. and after the call unset the alarm and install the previous signal handler. for details consult UNIX NETWORK PROGRAMMING
by richard stevens.
--mukhben

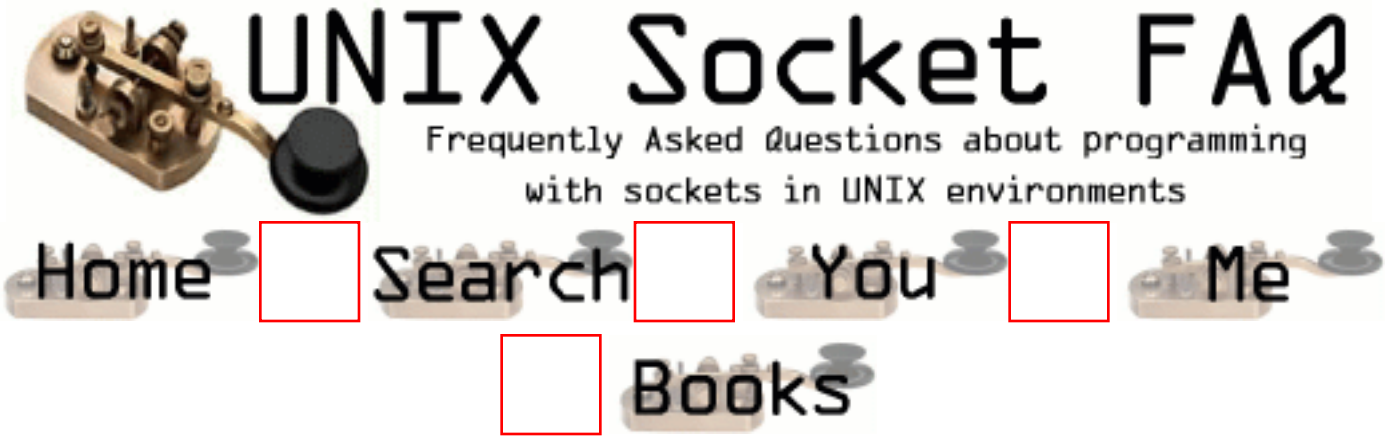
From: franky

If you have multiple connects going at the same time (multi-threaded) you can't use sigalarm. See the man page for details.

Try using a non-blocking connect instead and use select to find out when it is ready(/connected).

fv.

Add a Comment



New Questions: how many clients??

From: rikin

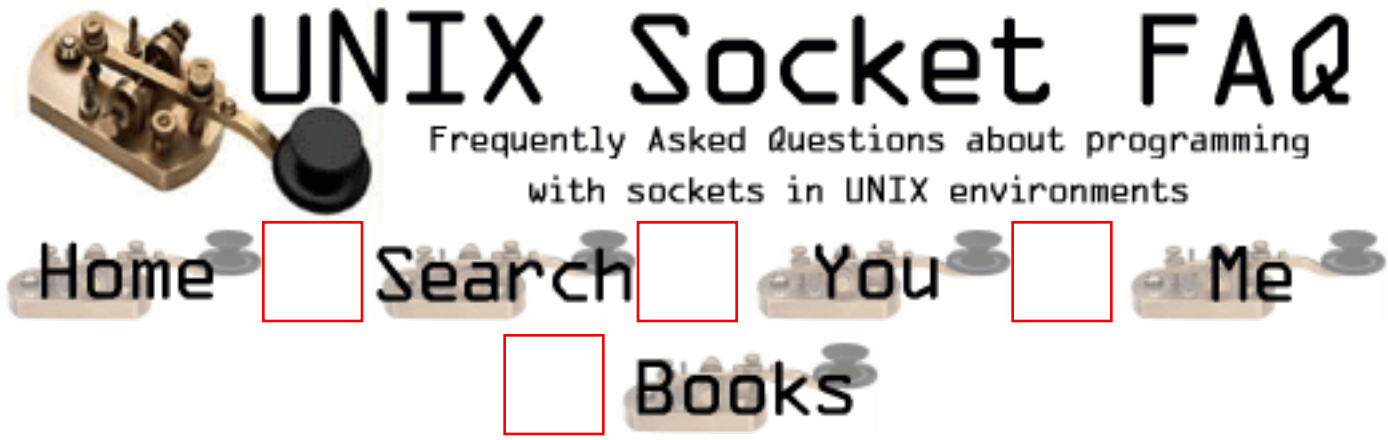
How can i determine how many clients are conencted to my server. I've tried everthing, using pointers, trying to get an array, , but nothing seems to work. if anyone knows, please email me back. thanks
Rikin

From: Michael H. Ward

The way I do it is to increament a global counter each
time I accept() a client then in the server. I then
decreament the counter when I detect the SIGCHLD signal.

Simple, but effective.

Add a Comment

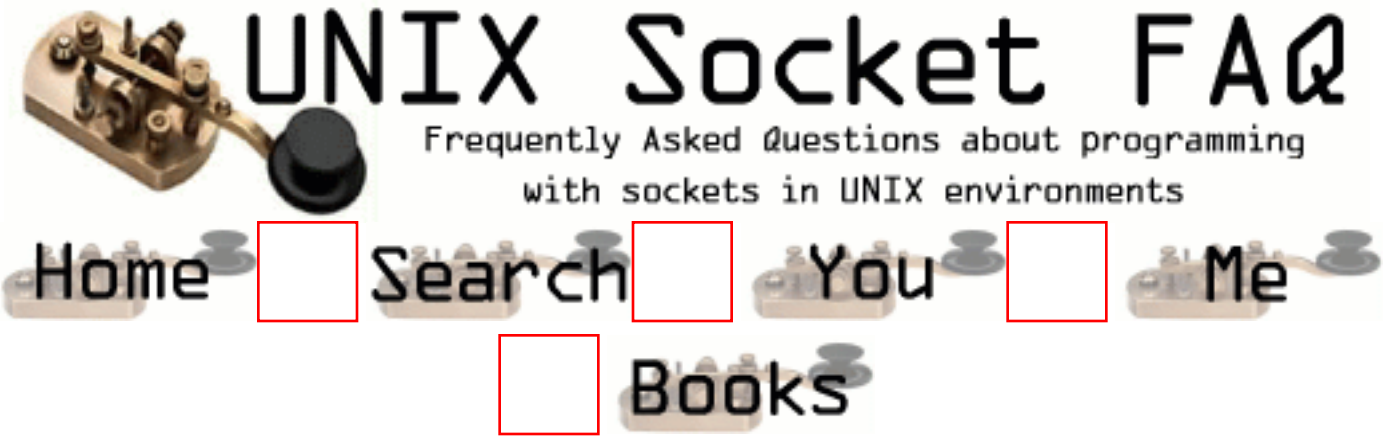


New Questions: raw socket with udp port scan

From: Danny

I try to use raw socket to check if UDP port is open.
I am using udp socket with sendto() and then I use select() with raw socket as expected fd and I always get zero as return value which means time out.
If the udp port is close I expect getting ICMP error.

Add a Comment

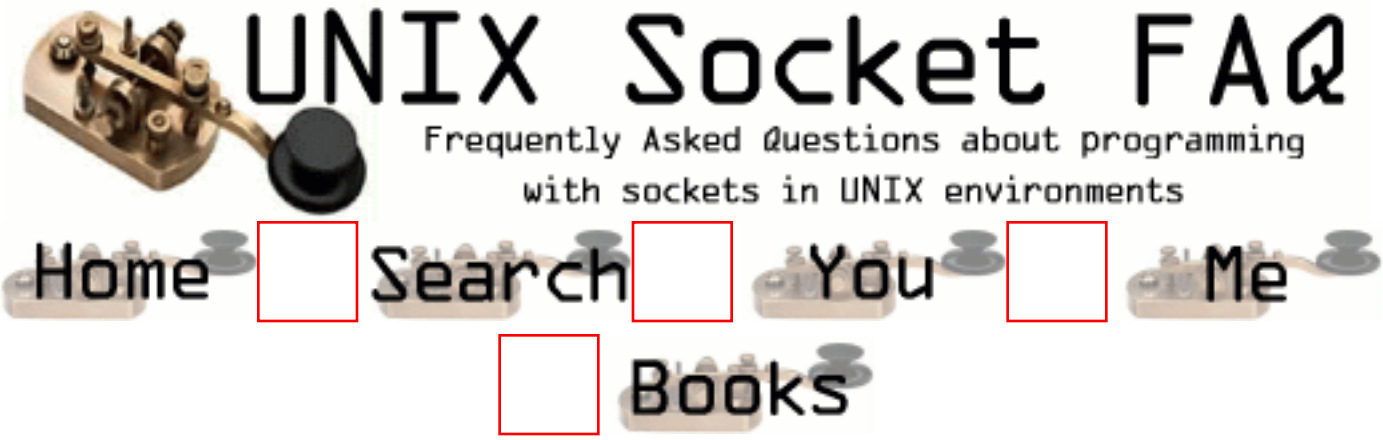


New Questions: raw socket

From: Danny shaul

how can I read ICMP destination port unreachbale with raw socket.
dose raw socket has to be configured with setsockopt() befor trying to recv this message.

Add a Comment



New Questions: raw socket

From: Danny shaul

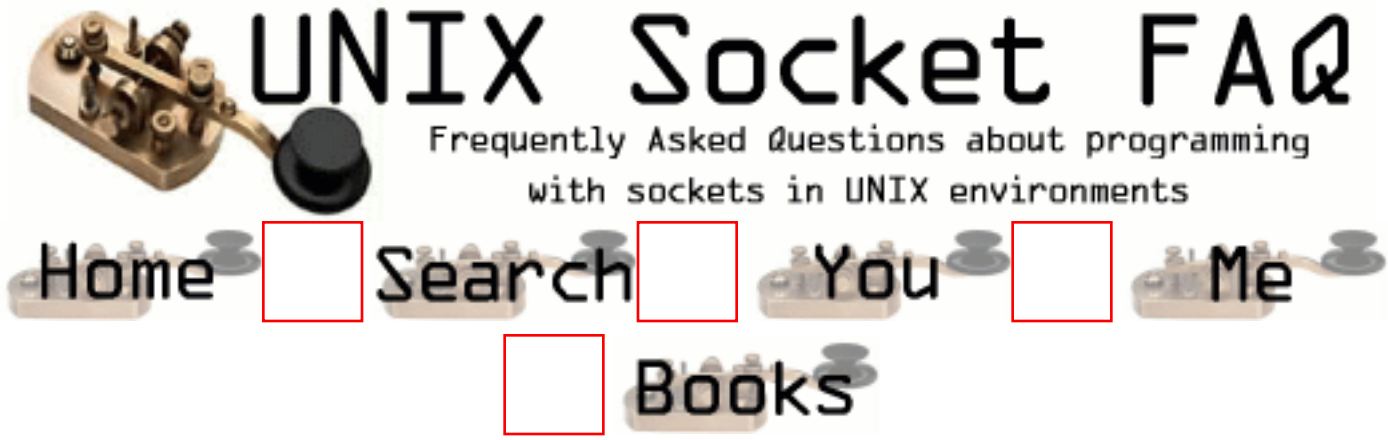
how can I read ICMP destination port unreachbale with raw socket.
dose raw socket has to be configured with setsockopt() befor trying to recv this message.

Add a Comment

New Questions: How do i program different socket types??..Example: type 8 type 3, etc..

From: hermes haley

Add a Comment



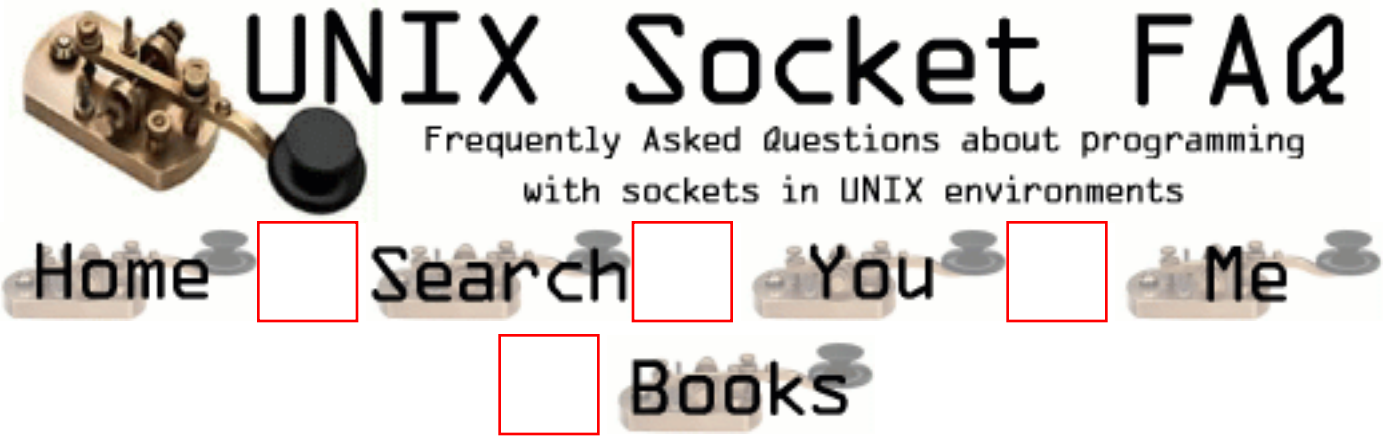
New Questions: How do i program different socket types??..Example: type 8 type 3, etc..

From: hermes haley

i would like to program raw socket types.. i'm not sure how..i would also appreciate if you explained to me how
a firewall constantly has access to your tcp stack and can interfere with it, as in block your packet i/o, thnx - hermes haley

---

Add a Comment



New Questions: TIME_WAIT

From: Michael Hecht

Hallo,
I want to connect with a client to a server every about 5
seconds, send data and disconnect. If I do so, the connection
impossible after about 195 connections on my SGI O2 and after
about 100 connections on a Siemens RM600. If I call netstat
there are all 195 (100) socket connections in the
CLOSE_WAIT status (from client and in FIN_WAIT_2 status
from server [both are running on the same computer at the moment]).
I know that this problem is related to the TIME_WAIT behaviour
of TCP/IP, but what is to do to avoid my problem?

Thanks in advance

Michael

From: Rob Seace

No, actually, the problem you describe is NOT related to
the TIME_WAIT state issue, I don't think... I think what
your problem is, is that the end sitting in the CLOSE_WAIT
state never actually issues a close() call of its own, so
the socket remains in this undead state, never fully
shutting down... The states you describe (FIN_WAIT_2 on
one end, and CLOSE_WAIT on the other) would seem to indicate
a situation in which only ONE end (the FIN_WAIT_2 end) has
close()'d their socket... Once the other end (the
CLOSE_WAIT end) close()'s the socket, the FIN_WAIT_2 should
change to TIME_WAIT for a period (which is normal, and
desirable), while the CLOSE_WAIT should just go to CLOSED,
and disapear...

From: Michael Hecht

Thank you for your comment. Before I added shutdown() to every
close() statement I had TIME_WAIT, and the same problem as before.
Afer adding shutdown() and hoping, that now the sockets are closed
"faster" I got the described behaviour.

From: Michael Hecht

Hello Rob,

you was right!! I tried now a simple example without any
complex addings, and it worked!! So I know now what to look
for. Thank you again!

Michael

From: Helios de Creisquer

Hi !
I've got the same problem for basic client/Server talk
of about 1 sec, with a TIME_WAIT persistent during 1 minute
After reading the previous comments, I added a
shutdown(sock, SHUT_RDWR) before the close(sock)
to both server&client (one's never know ;o).
But the TIME_WAIT status is always here for about 1 minute
and neither 'listen' or 'connect' are possible on this port

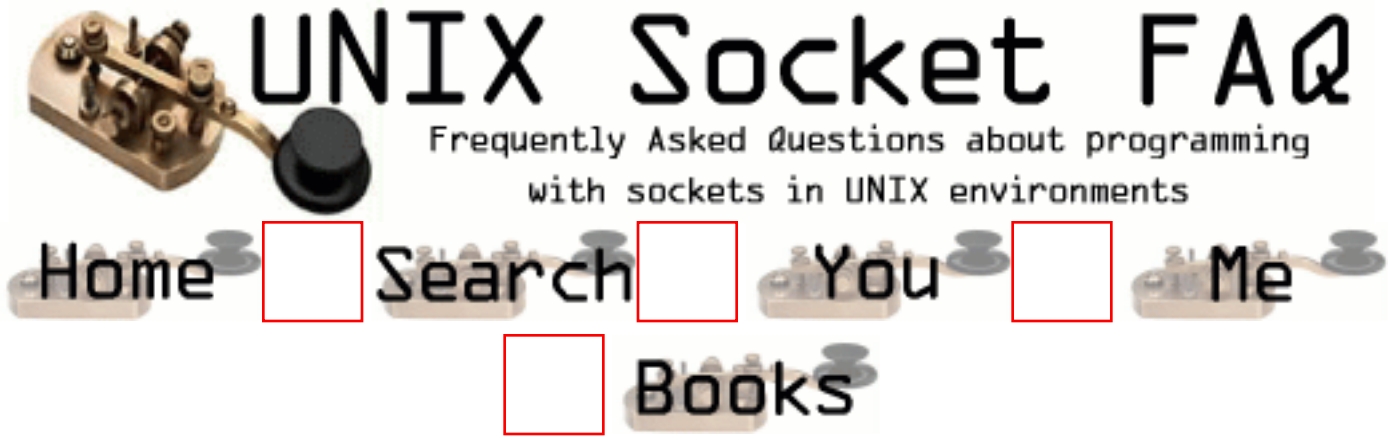
I'm very new to socket programming, Sorry in advance if
the answer to this question is evident ;-)

Thanks in advance.

From: Rob Seace

No, TIME_WAIT is NOT a problem; it's normal, desirable behavior... Don't try to get around it; it's there for a good reason... See question 2.7 in the main FAQ...

Add a Comment



New Questions: printf() and select()?

From: Andy

Using W Richard Stevens tcpservselect01.c, I'm trying to change it to write anything received to ALL sockets.
I thought that changing the Writen() call to something like:

```
for (j =0; j < maxi; j++ {
   if (client[j] < 0)
      continue;
   Writen(client[j], line, n);
}
```

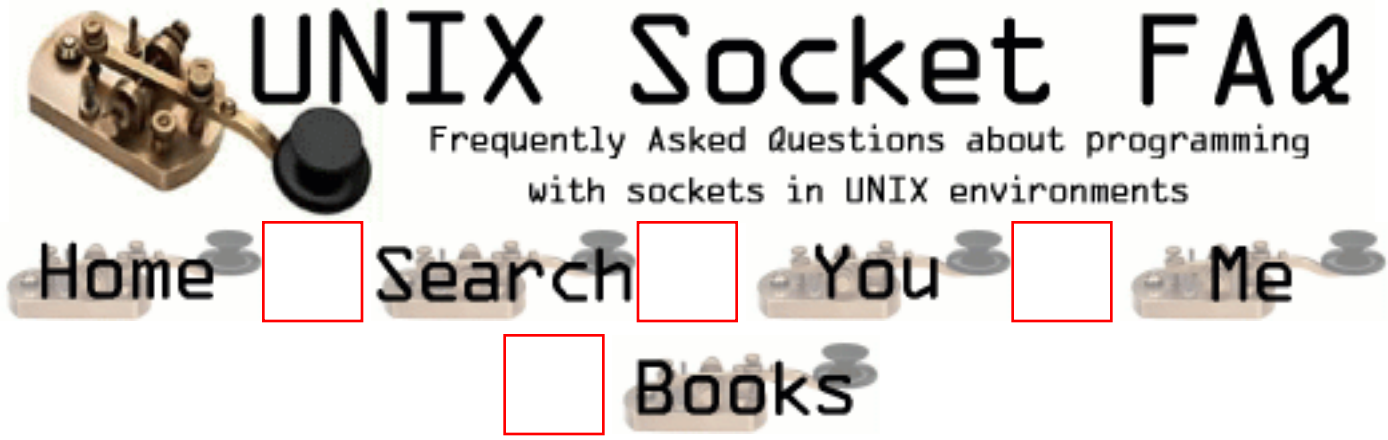
would work, but it isn't ??
Also I can't get any output from printf() after the select() call.

Any Ideas?

From: Andy

Please ignore above, printf does work, it's just that for some reason, the client is not connecting to the server. Why I do not know - it appears to, and echoes back to itself but it doesn't.

Add a Comment



New Questions: Should socket be closed if sendto() or recvfrom() fails ?

From: AK

Hi,

the subject line basically says it all. I wish to know whether it is necessary to close down the socket if a sendto()or recvfrom() call
for a UDP socket fails ? In case of a TCP socket, a failure could mean that the socket is no longer connected, so it should be closed down
in that case. But since in case of UDP there is no concept of "connections", i feel read/write failures should not mean socket must be closed
down. Am i right in thinking this ?

Thanks in advance,
AK.

From: Frank Binder

Hi,
you only have to close an UDP-Socket if you called a bind() on it, so that you can use the recv() and send() calls. Else the port won't be available anymore, unless your application is exiting. If you don't use bind(), you don't need to close the socket to free the port, but to free the socket descriptor, I would suppose. Does anybody disagree?

Greetings

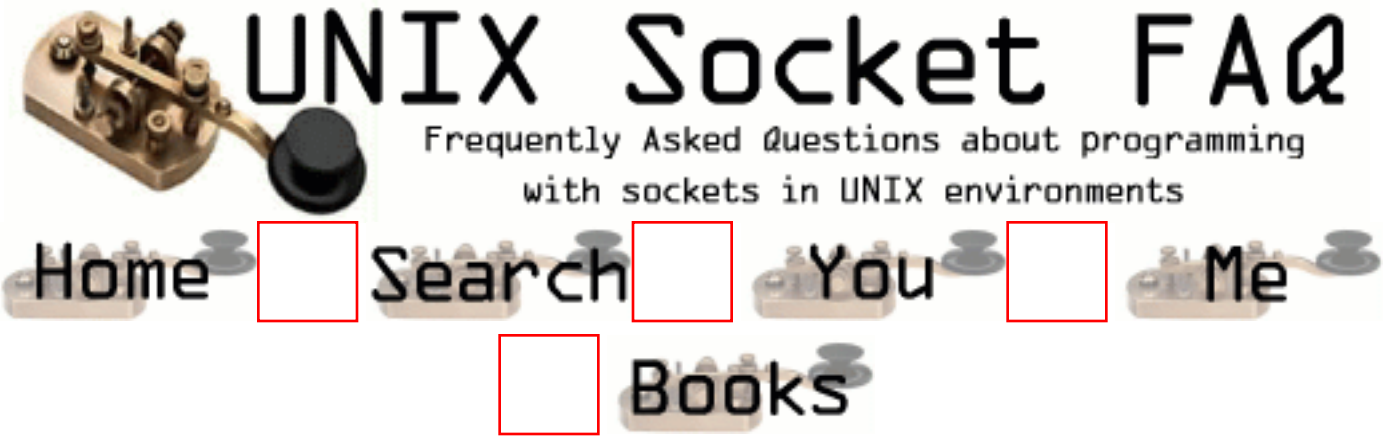
Frank

From:

Hi,
The UDP socket still needs to be closed as it is a file descriptor that has been allocated. Calling

or not calling bind() should have nothing to do with this.

Add a Comment



New Questions: TIMEOUT USING SELECT

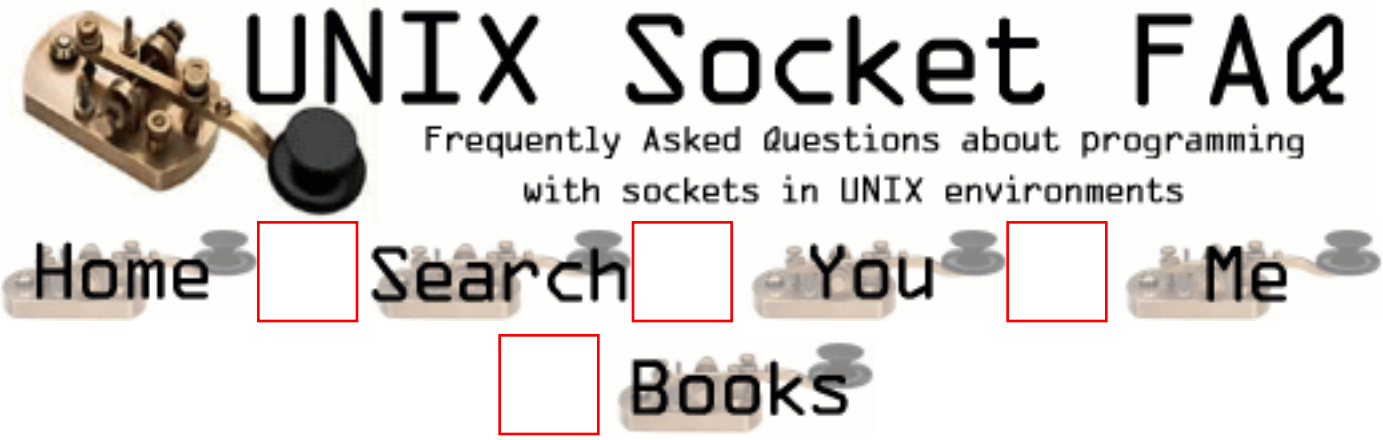
From: Mikael Chambon

Hello, I am trying to see what is the best way to set a timeout
for my client/server app. Until now I am using blocking socket,
is there a way to set a timeout or shoult I use non blocking socket??
If yes, could you explain me How to do it with select ??
Thx for all..

From: devnull

you should use either select
or setsockopt routines.

Add a Comment



New Questions: CLosing connexion

From: Mikael Chambon

I am writing a client/server application, the server has
a while (1) loop waiting for connexion with accept and then
I do a fork for the client process. But the problem is when
the server child do a close on the socket, the client connexion is not
closing , do you know why ?? Is is because of the fork ??

From: Mikael Chambon

No I've found why, after the fork, I have to close the fd
in the father process and let it in the child one ..

Add a Comment

New Questions: problems in UDP socket programming(recvfrom)

From: Azza

I made client/server application (UDP) but I faced this problem:
 *client sends data broadcast correctly using sendto()
  but server couldn't recieve using recvfrom() and it's
  blocked forever
what can I do?

Note:
  this is sample of my code:
  client.c

  {
   struct sockaddr_in broadcast_addr;
   struct sockaddr_in local_addr;
   int RREQ_Sock;
   int sent,bound;
   RREQ *RouteReq;
   int optval;
   int optlen;

   if((RREQ_Sock=socket(AF_INET,SOCK_DGRAM,0))==-1)
      { //for test
printf("can't create socket\n"); //for test
return -1;
      } //for test

   broadcast_addr.sin_family=AF_INET;
   broadcast_addr.sin_port=BROADCAST_PORT;

```
   optlen=sizeof(int); //for test
   optval=1; //for test
   setsockopt(RREQ_Sock,SOL_SOCKET,SO_BROADCAST,(char *)&optval,optlen);
   broadcast_addr.sin_addr.s_addr=inet_addr("255.255.255.255");

   sent=sendto(RREQ_Sock,(char *) RouteReq,sizeof(RREQ), 0 ,(struct sockaddr *)
&broadcast_addr,sizeof(broadcast_addr));
   close(RREQ_Sock);
   printf(" should have sent %d bytes\n", sent); //test
   return 0; //normal exit
  }
**********************************************************
  server.c
  --------
   {
   if((RCVRREP_Sock=socket(AF_INET,SOCK_DGRAM,pptr->p_proto))==-1)
      {
printf("ERROR Creating Socket\n");
return -1;
      }
    bound=bind(RCVRREP_Sock,(struct sockaddr *) &local_addr,sizeof(local_addr));
    if(bound==-1)
      {
printf("\n BIND ERROR");
return -1;
      }
    recieved=recvfrom(RCVRREP_Sock,(char *) RouteRep,sizeof(RREP),0,(struct sockaddr *)
&dest_addr,(int *)sizeof(dest_addr));
    if(recieved==-1)
    {
        printf("\n ERROR recieving data");
    }
    else
    {
      printf("\n I have recieved %d bytes ",recieved);
    }
    }
    close(RCVRREP_Sock);

**************************************************
```

these were samples of programs

when I run these programs & I used a sniffer to know what

happened

I saw that
client sent data correctly to server
but server sent to client an ICMP packet that:
- Destination unreachable
- Port unreachable
and server was blocked forever

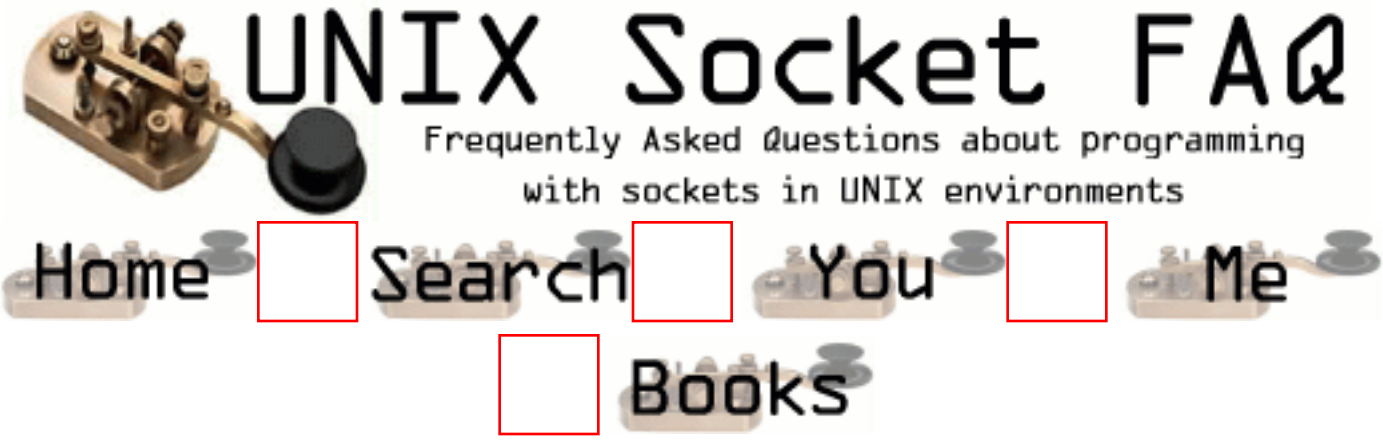
what is a problem?

please,I need your help as fast as possible

From: devnull

you should not send to ff.ff.ff.ff,
you should send to yours net broadcast.
suppose you have C net 197.7.7.123, mask ff.ff.ff.0,
then the broadcast should be 197.7.7.255

Add a Comment



New Questions: Problems in recieving UDP packets

From: Azza

How can I recieve UDP packets(broadcast&unicast)on the same
port?

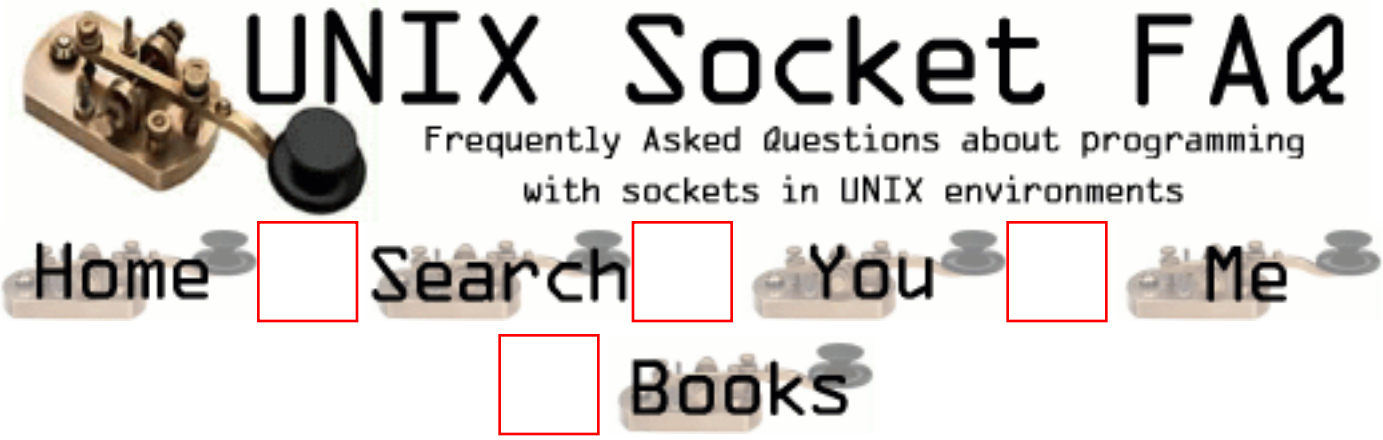
or

A server bounded to certain port and needs to recieve UDP packets
from different clients(some of them sends broadcast and others send unicast)
how can it recieves these two types of packets on the same port?

From: devnull

UDP socket bounded on some port , will receive *both*
unicast and broadcast packets.

Add a Comment



New Questions: UDP socket. receiving broadcast problem.

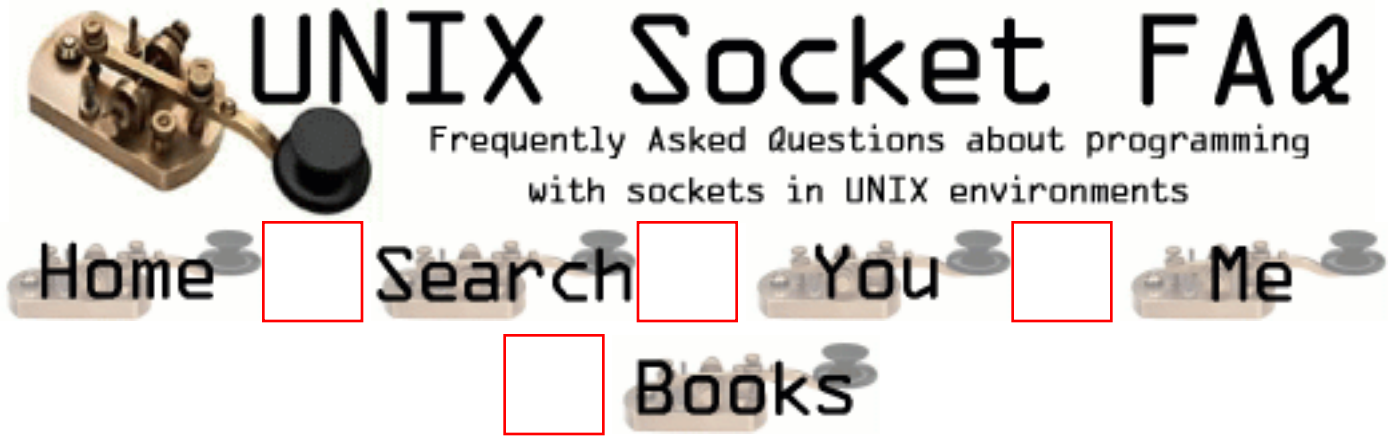
From: devnull

i have a problem receiving broadcast udp packet.
program does recvfrom() in a loop,
but recvfrom() get one message two times !
when i send one broadcast message, the program get this message
twice from a socket. Whats wrong ?
it's all OK with unicast.

From: devnull

hello devnull, it's me, devnull.
i found out a solution by mysqelf.
you bind you socket to INADDR_ANY on a multihomed computer with
2 interfaces :-)

Add a Comment



New Questions: FTP protocol

From: kaler

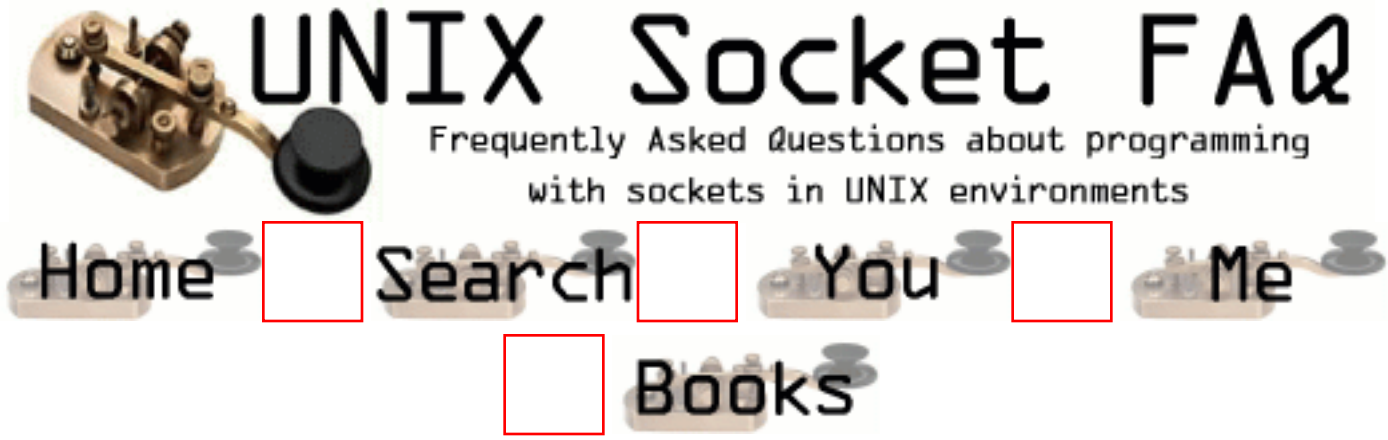
Please help me to find the detail of FTP protocol and some samples for client. thank you

---

From: Rob Seace

FTP is detailed in RFC 959 (which you can find many places, including here)... As for sample implementations, there should be several to choose from: the ones distributed with Linux distros should have source available, as should those that come with *BSD...

Add a Comment



New Questions: multiple applications on same host using same socket

From: John Peterman

Hi!

I hope this hasn't been covered somewhere, and I just overlooked it....

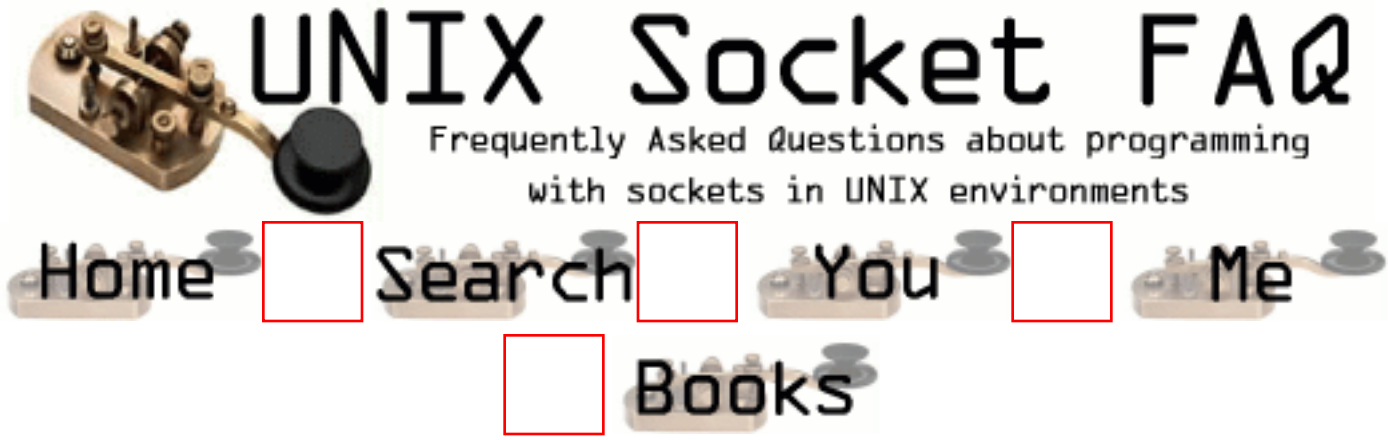
Here's the situation I'm looking at. Host A will be sending messages through a socket to Host B. Application(s) on Host B will need to acknowledge messages from Host A, and also to send messages through the socket to Host A. Basically, what I'm trying to figure out is this: Can two separate applications running on Host B use the same socket for communicating with Host A. Or can only one application on Host B use that socket? If two applications can use the socket, then I can have one program acknowledge incoming messages, and another program send messages to host A.

Thanks for your help!

From: Bruce Edgerton

John, as long as Host A is acting as a server and binds to a particular port (see socket, bind, listen functions), it can then listen for and accept any number of connections on that port from other client hosts (see gethostbyname, socket, setsockopt and connect functions). I suggest getting a copy of Unix Network Programming Volume 1 by W. Richard Stevens, it reveals all. The Tools++ Professional class library from RogueWave (if you are using C++) greatly simplifies the task. If you are using C, there is quite a bit of work.

Add a Comment



New Questions: Connecting to a Web server

From: Iztok Polanic

Hi!

How can I connect to a Web server and then get it's context (of the page)?

From: devnull

hello !
you should create SOCK_STREAM socket,
connect it to ip address / port of a web server,
and then send() him a HTTP request:
char *msg= "GET http://server/you/want HTTP/1.0";
send(s, (void*)msg, strlen(msg), 0);
you cat put various HTTP headers in a message:
char *msg= "GET http://<url> HTTP/1.1\0xd\x0a"\
"User-Agent: some agent\0x0d\x0a";
and so so
after that get a reply !
that's all.

From: Sudipto Basu

You can look after my code listed below it is doing the same thing.

```
#include <sys/types.h>
#include <sys/socket.h>
#include <stdio.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <netdb.h> /* for struct hostent */
```

```
#include <strings.h> /* for bcopy() */
#include <unistd.h> /* for read() */
#include <stdlib.h> /* for exit() */

#define SERVER_PORT 80
#define MAXHOSTNAMELEN 256
/* #define SERV_HOST_ADDR "203.163.50.62" */
/* #define SERV_HOST_ADDR "128.163.2.27" */
#define SERV_HOST_ADDR "www.uky.edu"

void str_cli(FILE *,int);

/* global variable */
struct hostent *servInfo;

int readline(int fd, char *ptr,int maxlen)
{
int n,rc;
char c;

for(n=1;n<maxlen;n++)
{
if ( (rc = read (fd,&c,1)) == 1 )
  {
*ptr++=c;
if ( c =='\n') break;
} else if ( rc ==0 ) {
if ( n==1) return(0);
else break;
} else return(-1);

}
*ptr = 0;
return(n);
}


int writen(int fd, char * ptr,int nbytes)
{
int nleft, nwritten;

nleft = nbytes;
while (nleft > 0) /* keep writing until the entire request is sent */
  {
nwritten = write (fd, ptr, nleft);
```

```
if ( nwritten <= 0 ) return(nwritten);
nleft -= nwritten;
ptr += nwritten;
}
return(nbytes - nleft);
}

void str_cli(fp,sockfd)
FILE *fp;
int sockfd;
{
int n;
char sendline[80], recvline[81];

/* get input from the stdin and put it in 'sendline'*/
while(fgets(sendline, 80,fp) != NULL )
{
n = strlen(sendline);
/* send the request to the server, ex: GET / */
if (writen(sockfd,sendline,n) != n )
printf("written error on socket\n");

/* once the request has been send, receive the response */
/* receive only 80 charactars and put them in recvline */
n = readline(sockfd,recvline,80);
if ( n<0) printf("readline error\n");
recvline[n] = 0;
fputs(recvline,stdout); /* print <=80 chars. to the stdout */
/* after receiving and printing <=80 chars, then??.....
'while' calls fgets(...), which waits for the input from
the stdin, (remember you haven't read all the charactars
from the sockfd, just 80 chars.). so even before you read
all the charactars from the sockfd, you are asking for an
input from the stdin, ofcouse if you just press 'enter',
it'll send nothing to the webserver, then again you call the
readline(...) which reads the next 80 chars and then waits
for the input from stdin. the process continues...
So the point is, read all the characters from the sockfd
(not just 80 chars) before you wait for an input.

when it prints 'connection established' to the stdout,
give the input as 'GET /' and keep the enter key pressed.
hope this helps.
*/
```

```
}
if (ferror(fp)) printf("error reading file\n");

}

int main(int argc, char *argv[])
{
int sd;
char serverName[MAXHOSTNAMELEN];
struct sockaddr_in serv_addr;

      printf("\n Enter name or IP of the web server (eg.
www.yahoo.com): ");
scanf("%s", serverName);
/* need to clear the stdin since you are taking the http request from
* the stdin, otherwise it will send the '\n' character */
fflush(stdin);

/* initialize the memory */
bzero((char *) &serv_addr, sizeof(serv_addr));

/* fill in the server address structure */
serv_addr.sin_family = AF_INET;
serv_addr.sin_port = htons(SERVER_PORT);

/* get the server information from the DNS, returns a pointer to
* structure hostent */
if((servInfo = gethostbyname(serverName)) == NULL)
{
fprintf(stdout, "Unknown server");
exit(0);
}

/* copy the server address returned by 'gethostbyname' into server
* address structure */
bcopy(servInfo->h_addr, (char *) &(serv_addr.sin_addr),
servInfo->h_length);

/* create a socket stream using TCP */
sd = socket(AF_INET, SOCK_STREAM, IPPROTO_TCP);

if ( sd < 0 )
{
printf("cannot open the socket");
exit(1);
```

```
}

/* establish a connection to the web server */
if (connect(sd, (struct sockaddr *) &serv_addr, sizeof(serv_addr)) < 0
)
printf("cannot be connected\n");
else {
printf("connection established");
}

/* now we are connected to the web server */
str_cli(stdin,sd);
close(sd);
        return 0;
}
```

From: jagan

when i try to compile the source code in linux it showing errors

parse error

please help me to complie the code

From: jagan

when i try to compile the source code in linux it showing errors

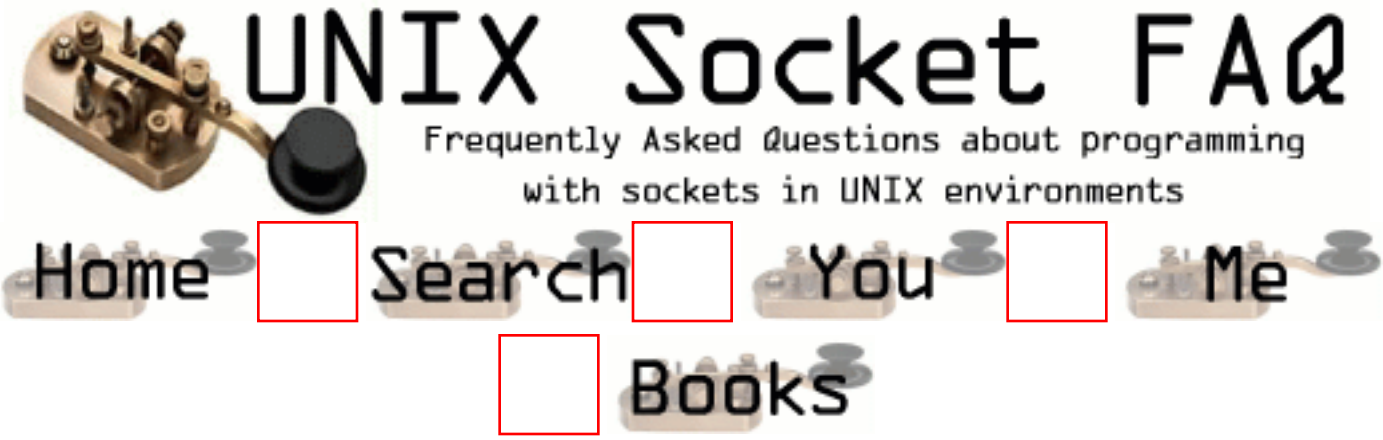
parse error

please help me to complie the code

From: Arthur Lung

Give me your address and I'll drive over to your house and
hold your hand while you do it...

Add a Comment



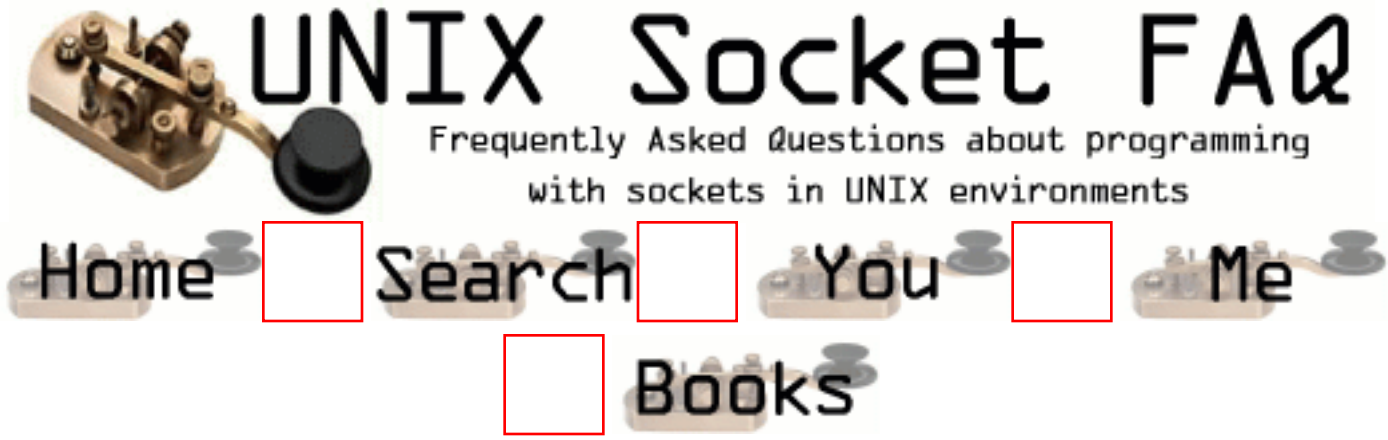
New Questions: Reading &Writing from UDP Port. (Help needed for Win)

From: Mathew T

Hi,
I want to write a procedure in Perl for writing & reading data from UDP port(Windows). I would like to check the port for messages & if messages are there i want them & if not i would like to wait a specified time & then check again.....
But i am not sure how?..... I am new to perl & if anyone could suggest some tutorial for perl-socket programming or some source codes.... that would be really helpful.

Any help is apreciated!
Thanks in advance!
Bye,
Mathew

Add a Comment



New Questions: Server: accepting and handling multiple clients

From: G.Gabriele

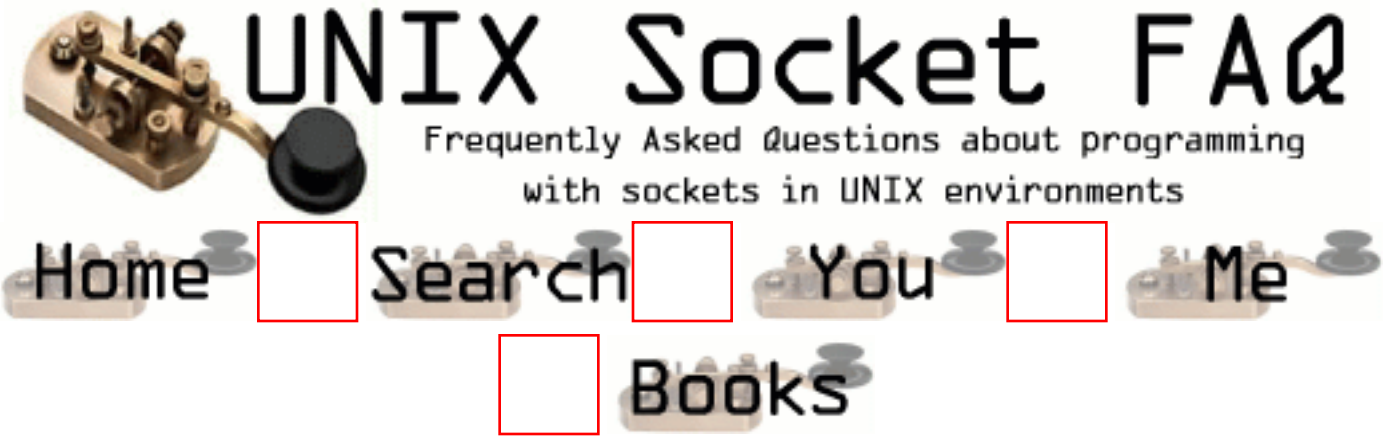
Hi , I'm writing a chat program (client/server) a kind of talk,
but for GNOME. Anyway the server should accept MAX number of
clients and then get the input from a client (when its available)
and send it to all the connected socket.
The problem I'm having is that when only the first client
it connect is able to send data, the other are only reciving data.

General tips on doing that would be appriciated, anyway if want to
take a look at the code:

```
struct sockaddr_in clientname;
size_t size;

fd_set active_fd_set, read_fd_set;

      /* this one set up a binding socket on port 9000 */
server_socket = gpe_server_socket_new (9000);

   FD_ZERO (&active_fd_set);
   FD_SET (server_socket, &active_fd_set);

   while (1)
    {

     read_fd_set = active_fd_set;
     select (FD_SETSIZE, &read_fd_set, NULL, NULL, NULL);
```

```
     for (si = 0; si < max_users_number; ++si)
     if (FD_ISSET (si, &read_fd_set))
{
if (si == server_socket)
{

printf("got connection.\n");
client_socket[client_n] = gpe_server_socket_accept(server_socket);
FD_SET (client_socket[client_n], &active_fd_set);
client_sockets_status[client_n] = 1;
++client_n;


}
else
{

char buffer[200];

int size;

size = gpe_socket_read(si, buffer);
if (size == 0)
{
close(si);
}

else {
int i;
for (i = 0; i < max_users_number; ++i)
{
if (client_sockets_status[i] == 1)
{
gpe_socket_write(client_socket[i], buffer);
}
}
}
   }

}
    }
 }
```

Add a Comment



New Questions: bind fails with udp socket

From: Marc Summers

Hello, I am working on a simple UDP socket program and
when I try to bind, I get the error
Bind Failed : Can't assign requested address

Any idea at all why I get this error, and what I can do
to fix it?
Also do I need to put any entry in the /etc/services file?
I am programming in C on a Unix system HP-UX 10.20
Thanks

From: Rob Seace

Well, it's difficult to say for sure, without seeing the
source to see exactly what you're trying to do... But, at
a pure guess, maybe you're trying to bind to a port below
1024, so you need to be root? Or, perhaps you're supplying
invalid arguments to bind(), somehow? I THINK I've seen a
similar error message when I screwed up the "addrlen" arg
to bind()... Or, perhaps that port is simply already in
use? (Though, I would expect a different errno than you
describe, for that... But, every system behaves a little
differently, so who knows...)

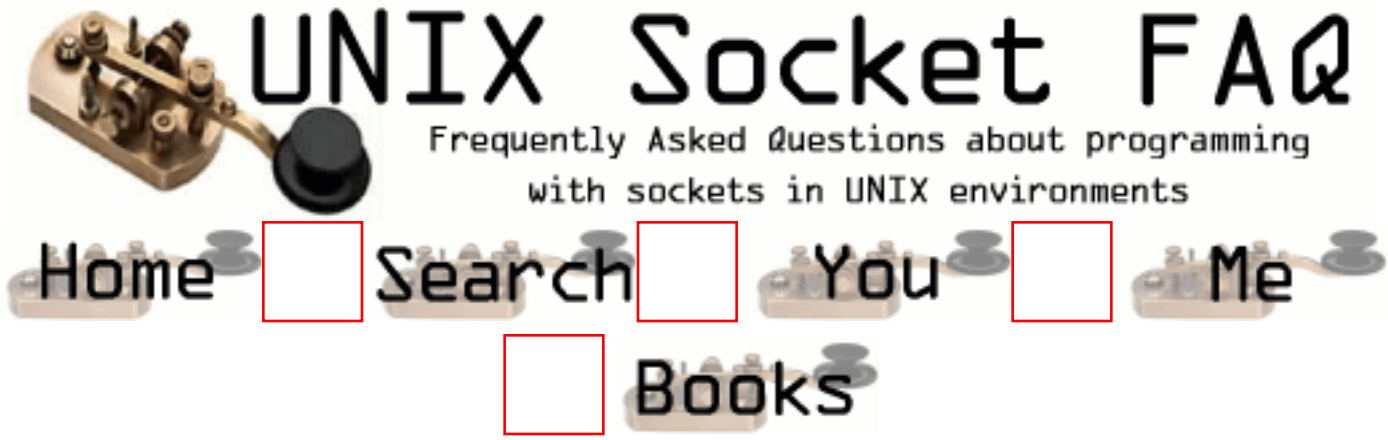
From: Igor

You can miss bind() to assign a port to a socket.
Just make a connect() the OS itself assign a local port to your socket
and then you can use send() and rcv()
instead of sendto() and rcvfrom() functions. ;))

From: Loco

I would like to see the code.
It might be that you are trying to bind to a specific IP address, but it isn't one of your interfaces addresses.
Another possibility is that you are using the number of the port directly without first converting it to network byte order, which would give a different port number, which in turn could be an invalid/in use/privileged port number (see Rob's answer)

Add a Comment



New Questions: sending structures through TCPIP sockets

From: Sebastian

i have a server prog and a client prog, both running on the same machine. I try to send structures with send(sockfd, &strct, sizeof(struct STRUC), 0). The client should receive data but it only receive the first sent structure correctly, the next ones get messed up.
The client uses recv(sockfd, &st, sizeof(struct STRUC), 0).
I would be grateful if someone would help.

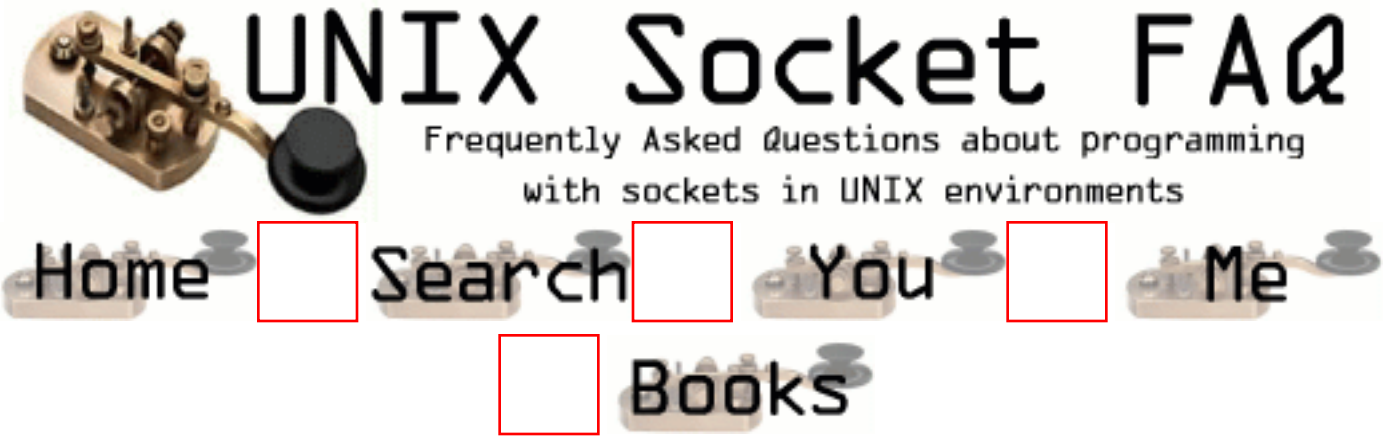
From: Jianxin Ge

You'd better to use RPC to transfer a structure. Or, you can serialize the structure into a byte stream with the XDR library and then send the byte stream to the peer.

From: Charles Campbell

Transferring structures via sockets is often dicey. Unless the two machines are the same kind, have the same o/s, and are using the same compiler, the transfer may not work. Some of the reasons: byte/word/etc alignment restrictions get satisfied (by "holes") differently, differing floating point formats, different integer "endian"ness. RPC uses XDR, and you may learn about both from "Unix System Programming using C++" by Terrence Chan.

Add a Comment



New Questions: I need help, i am a beginner

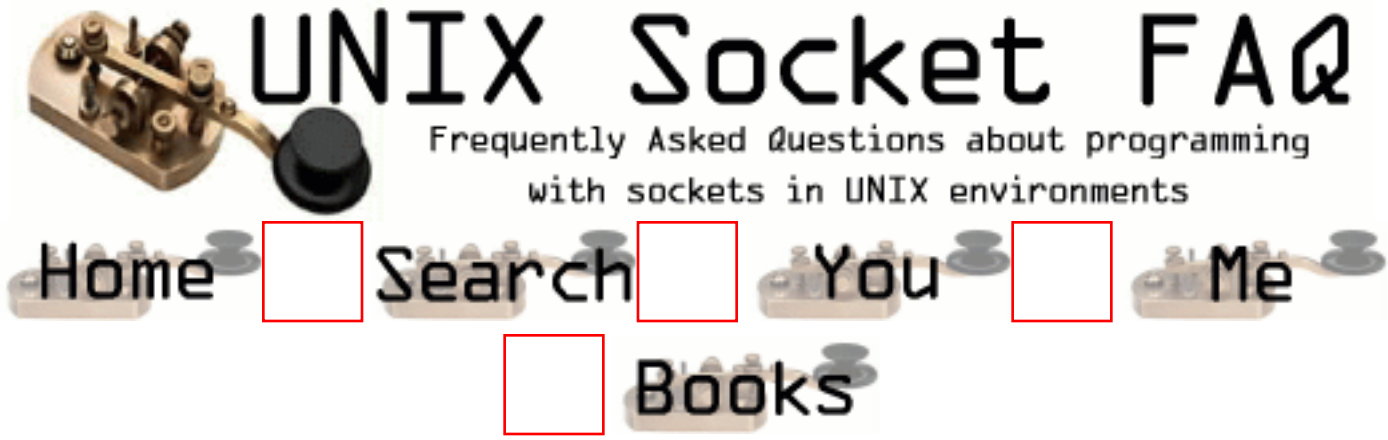
From: Angel Valencia

How do i use the function connect(), i need to know what is the socket descriptor and what type of variable is it.

From: Sebastian

go to
http://www.ecst.csuchico.edu/~beej/guide/net/
this is where i started learning (3days ago).

Add a Comment



New Questions: Listening to port traffic?

From: smitty

I'm trying to figure out how to intercept the incomming and outgoing traffic on a particular local port, hopefully without interrupting the traffic. Could anyone tell me how to do this, or point me in the direction of an article to read?

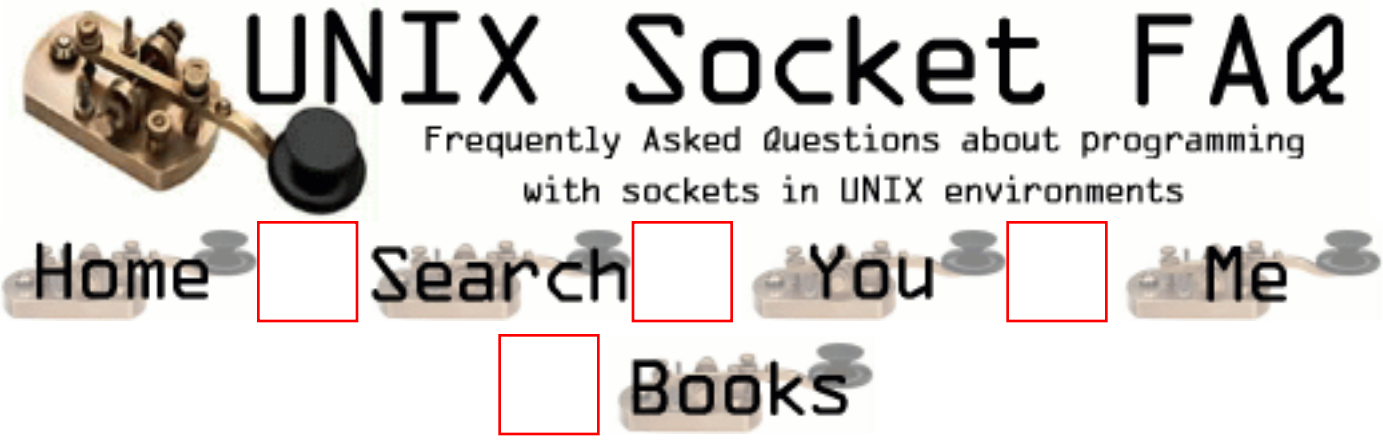
From: Rob Seace

It sounds like you want to get yourself a packet sniffer... You didn't say what OS you were using, so I can't recommend one... But, assuming a Unix-like OS, tcpdump is the standard workhorse command-line sniffer... But, my personal favorite is a GUI sniffer called Ethereal, which is just excellent... Another popular one is sniffit... If you're on a Windoze system though, your choices are much more limited (unless you're willing to shell out some bucks)... Someone ported "tcpdump" to Windoze, and called it windump... Other than that, the only decent free sniffer I've seen is an old Back Orifice plug-in, by the bizarre name of "Butt-Sniffer"... ;-) (which I can sadly no longer find an official page for, but you can easily find a copy on several less than trustworthy pages via search engine, if you really must have a copy... I'd say "windump" is your best bet, though... Or, better yet, get yourself a free copy of Linux and install that; Red Hat even ships with "tcpdump" installed, standard... ;-))

From: Jacques Richer

Ethereal has been ported to Win32. You can find a pointer
to the binaries off the ethereal homepage.

Add a Comment



New Questions: how do i convert binary..

From: lguo

Add a Comment

New Questions: how do i convert binary..

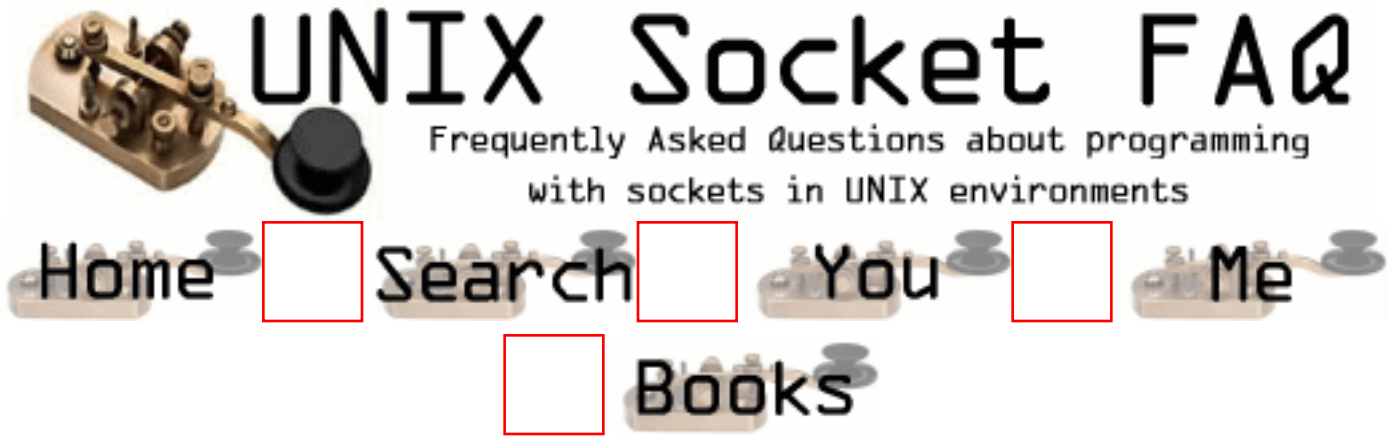
From: lguo

hi, i have created a server socket, and the client sent me a access file which is binary. i save it to a file, when i open a file, it is like this
"^A~^^A~\^A~\^A~\^A~X^A~X^A~X^A~X^A~X^A~X^A~X^A~X^Ap^Al^Af^AD^A6^A$^Aö....."
how can i convert them to charater???
thanks alot

From: soullace

Binary data just means there are bytes that are not printable, using a standard character set like ASCII. When you open up a file(in a text editor) it assumes that the bytes in the file are printable (i.e. characters). Using the ASCII standard, there are 'characters' which are not printable. So the text editor trys to display some of them as ^A, ^X, ^M, and so forth, the rest it doesn't display. If you tried opening up the file in a HexEditor, it might make more since, or it might not.

Add a Comment

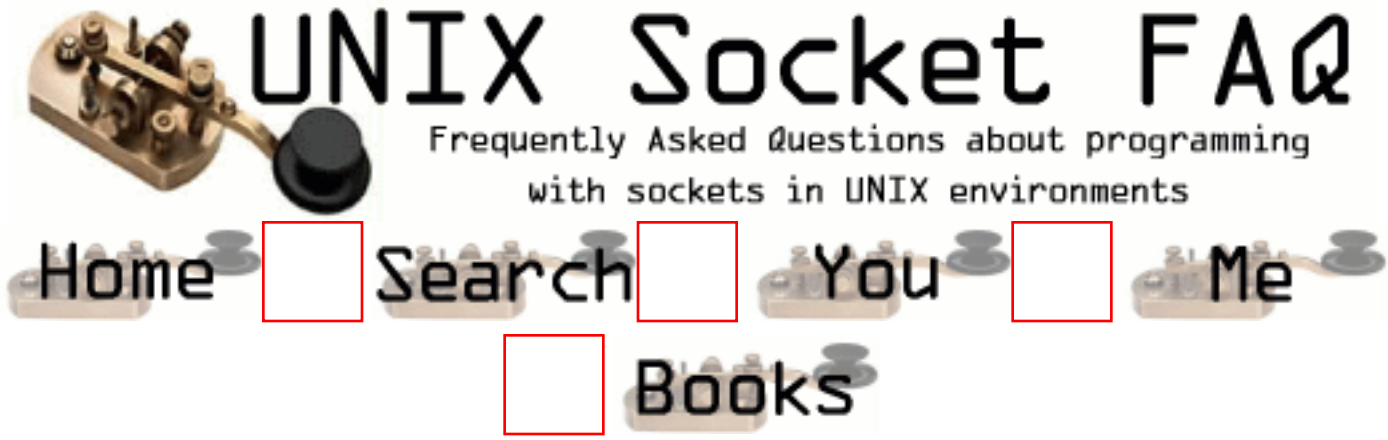


New Questions: How can i receive a file which size is unknown?

From: gary

hi, i wrote a server socket which can receive a string which size has been set up. i want to know how i can receive a database or whatever other file which size in no known and how i can store the whole file, and seperate each data? is it anything related to packet?
thanks

Add a Comment



New Questions: Err#9 EBADF on door_info call

From: Chris Milner

Hi,

I'm running PHP4, Apache, Oracle8i on Solaris 2.7. I have an intermitment TNS-12154 Cannot resolve service name in the browser when trying to connect to the db server. A PHP developer has looked into this and has fixed the bug, (some sort of double closure of the Oracle socket was occuring in one the PHP Apache modules). BUT, the problem came back after applying the patch.

I've truss'd the httpd processes and in one of the truss output, an error frequently appears which concerns me. The error is...

read(33, " # T N S N A M E S . O".., 8192) = 910
read(33, 0x0026BB8C, 8192) = 0
llseek(33, 0, SEEK_CUR) = 910
close(33) = 0
getuid() = 60001 [60001]
open64("/etc/.name_service_door", O_RDONLY) = 33
fcntl(33, F_SETFD, 0x00000001) = 0
door_info(33, 0xFEABA200) Err#9 EBADF

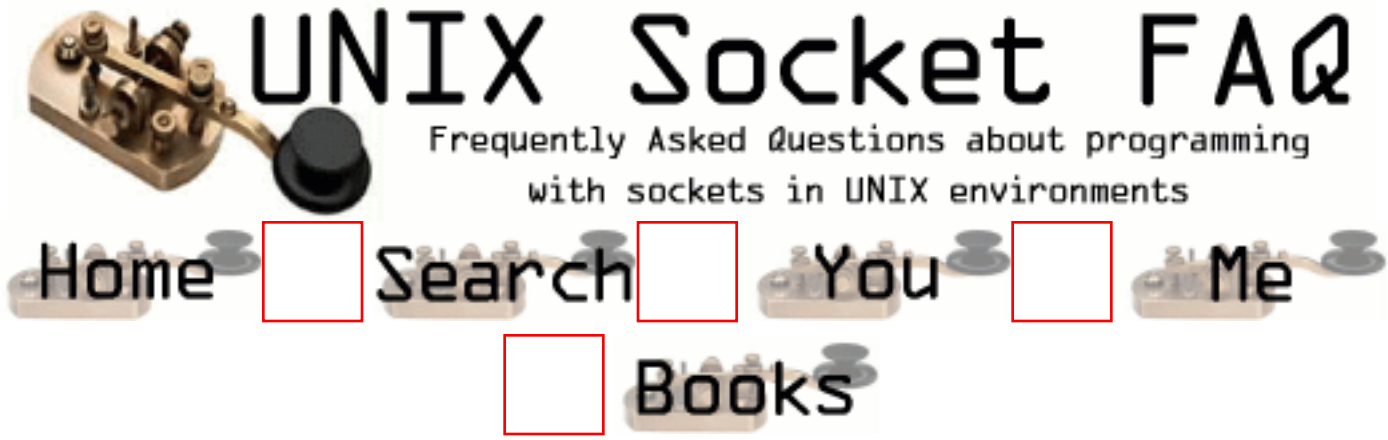
The PHP developer reckons there's now another problem, in that something, somewhere is closing the Oracle socket??? Is this what this error suggests. I know that TNS-12154 definately suggests that it cannot find the tnsnames.ora file.

Have you got any suggestions for us on how to proceed...can we trace this further....maybe we need to truss the processes from the start to see it working, then watch it fail and hopefully see why it failed.

We're desparate now...any help is greatly appreciated.

Chris

Add a Comment



New Questions: Problem with write

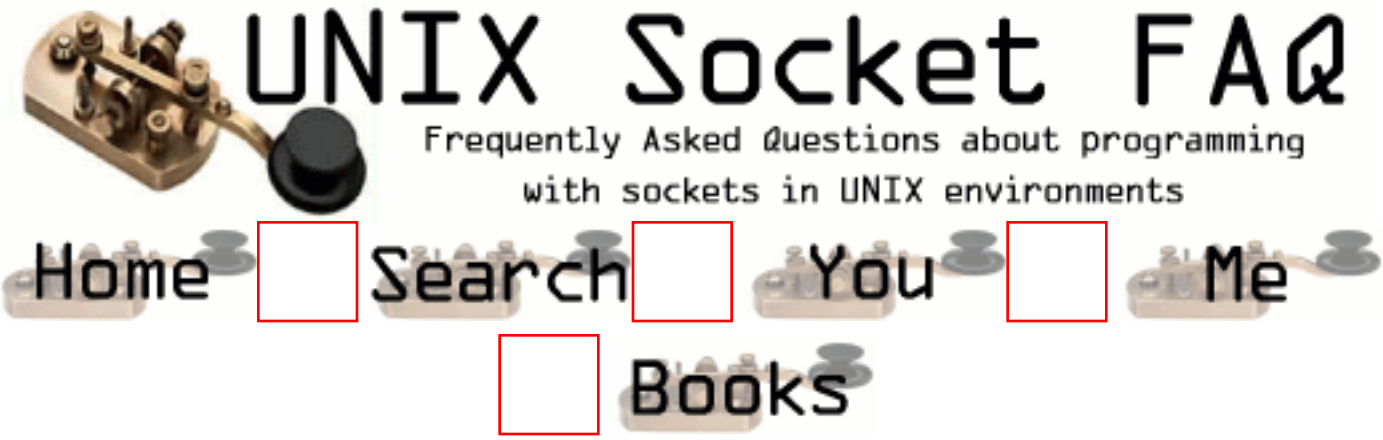
From: Ram

I am developing a multithreaded socketserver which accepts connections from client and also connects back to clients.(inessence a server talking to another server). When I tried to write to other server to which I connected to and if the socket is invalid my program exits whitout giving me any message. I tried catching SIGPIPE but in vain.
Has anyone encountered this error before.

From: Matthias Blankenhaus

I had a similiar problem. So I simplified my code to one process that contains two threads: client/server. After the connections is successfuly established (netstat -an) I try to send data from the client to the server. As a result I get a SIGPIPE. This is only true for blocking I/O. If I use non-blocking I/O in combination with select() I can avoid the problem altogether.

Add a Comment



New Questions: How can I know my peer is dead(or close the socket) without read or write?

From: sung-jin,kim

Hi. Glad to meet you.
Thanks for reading my question.

That is,
my server want to know whether my peer(client) is dead or close the connection without read or write the socket-descriptor.

I've tried to this using select + ioctl(FIONREAD), because if r_set is true in select, and read buffer count is zero,
I can recognize that client was dead.
but, sometimes the read buffer count is not zero with client death situation.

is there any method to know about client death?
or, how can I know about socket-descriptor state?

maybe, I can do this if I would know about server socket-descriptor state with CLOSE_WAIT.

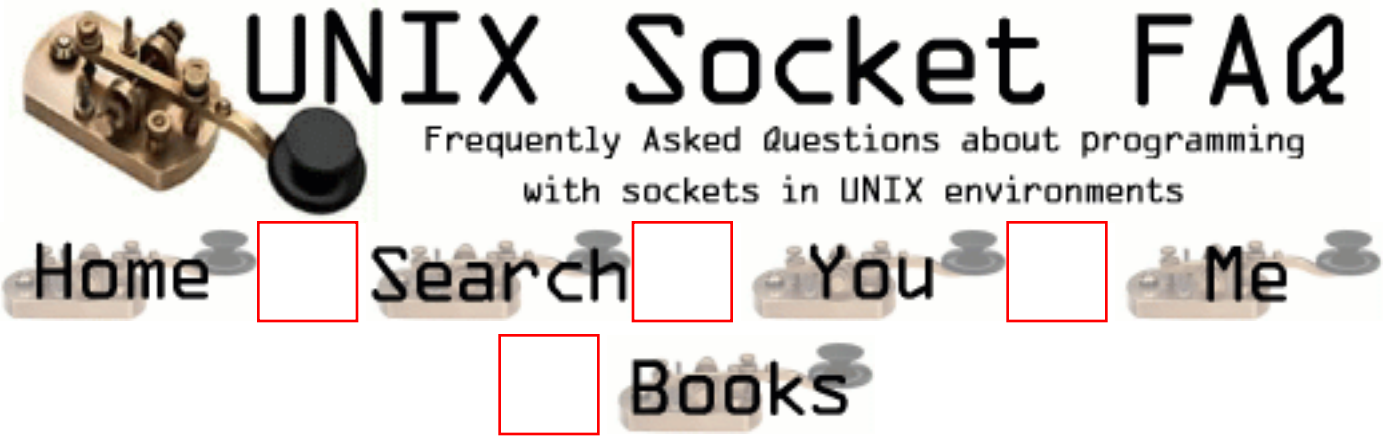
bye

From: Sam Horrocks

The only way to tell when the client is dead is when the
read returns 0 bytes. If the client dies and there is still
some unread data on the socket, you will see a read with a
non-zero value, but the next read after that will return 0
bytes.

Also, if your client dies, it may not shut down the tcp connection properly, and your server will never be informed. You may want to use the socket option "SO_KEEPALIVE" so that you'll be informed if the client is no longer responding.

Add a Comment



New Questions: sigaction and sigset

From: Ram

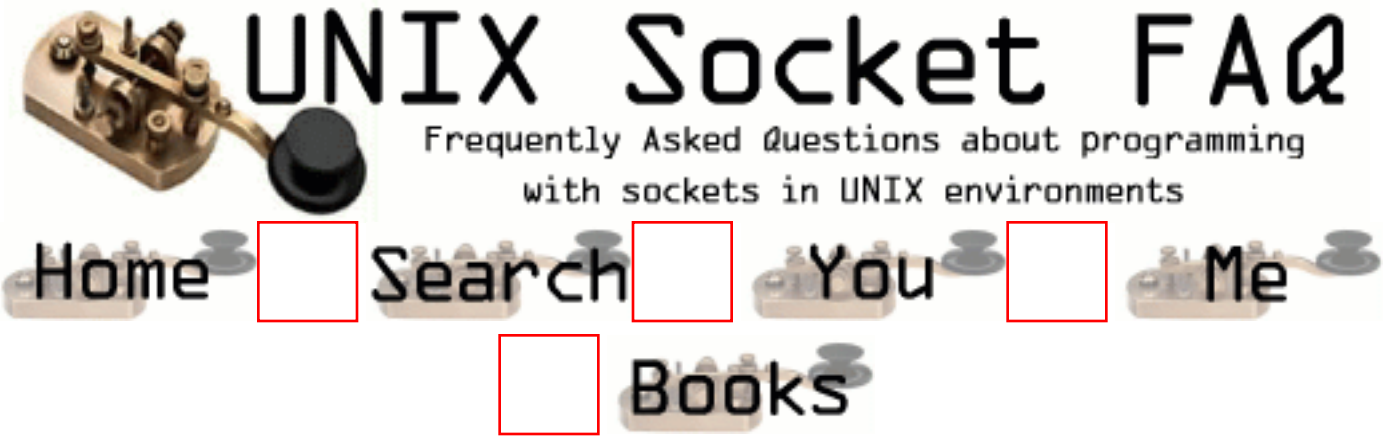
Hi,

Can anyone tell me when to use sigaction() and when to use sigset()

From: Ninja

sigaction is the generic signal handler on Linux, ported and improved from BSD standards. However, it does have some problems, since most signals perform default actions anyway and the signal handler is not always cleanly unloaded from memory. sigset avoids this problem. Think of sigset as a sigaction (or sigvec, if you are a BSD lover:D) that just doesn't mess up anything (almost). The safest choice if you do not need to much of refinement is using the standard ANSI library function signal, that is built on top of sigaction but works most of the time without too much trouble.

Add a Comment



New Questions: What is EAGAIN ??

From: AK

Hi,

I would like to know what the error EAGAIN means in the case of <<blocking>> sockets?
Also, what is the cause of this error occurring - does it have anything to do with the
load on that particular server ?

Thanx in advance,
AK.

From: Chad

The only time I have seen the EAGAIN is when I write to socket that is full. In our situation the reader was doing too much processing before getting back to the select while our sender was sending a steady stream of data.

From: Michael H. Ward

The EAGAIN (or EWOULDBLOCK under the new POSIX standard (1g))
basically says that if you are in NONBLOCKing mode, and made
a call to accept(), there is nothing there are no clients there
waiting to connect. The accept() will return a -1, and the
errno will be set to EWOULDBLOCK. So if you are wanting to
let various client machines connect to your server and you
don't want your server to hang on the accept() function (i.e.
you may need the server routine to run off and do other tasks)
you set the O_NOBLOCK option on the socket, then set up a
loop that executes an accept() command. If the accept()
returns -1 and errno == WOULDBLOCK, then there is really no

error, just nobody to connect to so your server is free to go about it's tasks. If accept() returns something else, then you have a client trying to connect, so you need to process that connection.

Hope this helps.

From:

If you have a nonblocking socket, then EAGAIN is returned in any context in which what you tried to do would have to block to succeed. Examples are:

accept a connection when nobody is trying to connect to you
read some data when there isn't any to read
write some data when the socket's buffers are full

It can also mean something like you specified a timeout for the operation to take place, but the timeout expired before whatever it was happened, and so you should try it again. For example, if you have a receive timeout of 20 seconds set, you call recv, but no data arrives for 20 seconds.

Add a Comment

New Questions: Question on select() and send()

From: Chong

Hi,
I generate a 15475 bytes of data in client (n1) and
my server (n2) uses select() to read the data. But why
the select() don't receive the whole chunk of data at the
same time? You can see the output from the server as following,
and the result from tcpdump at the client(n1) side.

Thank you for any suggestion,
-Chong

```
/**********************************
******* Client Result *********
**********************************/

   Time:[964847266.90611], generate 15475 chars
[Fri Jul 28 23:07:46 2000] send 15475 chars


/**********************************
******** Server Code *********
**********************************/
While(1){
     waitT->tv_sec = 0;
     waitT->tv_usec = 500;
     n = select (maxfd + 1, &rfds, NULL, NULL, NULL);
```

```
     .....
}//while

/*********************************
******* Server Result ********
*********************************/
            GW:receive 2896 chars
            GW:receive 1448 chars
            GW:receive 1448 chars
            GW:receive 2896 chars
            GW:receive 2896 chars
            GW:receive 3891 chars
/*********************************
***** tcpdump (n1) Result ****
*********************************/

22:56:02.521416 > n1.1360 > n2.qos: S 4073992885:4073992885(0) win 32120 <mss
1460,sackOK,timestamp 65547544 0,nop,wscale 0> (DF)
22:56:02.521596 < n2.qos > n1.1360: S 4082996984:4082996984(0) ack 4073992886 win
32120 <mss 1460,sackOK,timestamp 151588220 65547544,nop,wscale 0> (DF)
22:56:02.521645 > n1.1360 > n2.qos: . 1:1(0) ack 1 win 32120 <nop,nop,timestamp 65547544
151588220> (DF)
22:56:02.662159 > n1.1360 > n2.qos: P 1:1449(1448) ack 1 win 32120 <nop,nop,timestamp
65547558 151588220> (DF)
22:56:02.662219 > n1.1360 > n2.qos: P 1449:2897(1448) ack 1 win 32120 <nop,nop,timestamp
65547558 151588220> (DF)
22:56:02.662611 < n2.qos > n1.1360: . 1:1(0) ack 1449 win 31856 <nop,nop,timestamp
151588234 65547558> (DF)
22:56:02.662697 > n1.1360 > n2.qos: P 2897:4345(1448) ack 1 win 32120 <nop,nop,timestamp
65547558 151588234> (DF)
22:56:02.662713 > n1.1360 > n2.qos: P 4345:5793(1448) ack 1 win 32120 <nop,nop,timestamp
65547558 151588234> (DF)
22:56:02.663094 < n2.qos > n1.1360: . 1:1(0) ack 4345 win 31856 <nop,nop,timestamp
151588234 65547558> (DF)
22:56:02.663127 > n1.1360 > n2.qos: P 5793:7241(1448) ack 1 win 32120 <nop,nop,timestamp
65547558 151588234> (DF)
22:56:02.663142 > n1.1360 > n2.qos: P 7241:8689(1448) ack 1 win 32120 <nop,nop,timestamp
65547558 151588234> (DF)
22:56:02.663157 > n1.1360 > n2.qos: P 8689:10137(1448) ack 1 win 32120
<nop,nop,timestamp 65547558 151588234> (DF)
22:56:02.663525 < n2.qos > n1.1360: . 1:1(0) ack 7241 win 31856 <nop,nop,timestamp
151588234 65547558> (DF)
22:56:02.663555 > n1.1360 > n2.qos: P 10137:11585(1448) ack 1 win 32120
<nop,nop,timestamp 65547558 151588234> (DF)
22:56:02.663569 > n1.1360 > n2.qos: P 11585:13033(1448) ack 1 win 32120
```

<nop,nop,timestamp 65547558 151588234> (DF)
22:56:02.663582 > n1.1360 > n2.qos: P 13033:14481(1448) ack 1 win 32120
<nop,nop,timestamp 65547558 151588234> (DF)
22:56:02.663777 < n2.qos > n1.1360: . 1:1(0) ack 10137 win 30408 <nop,nop,timestamp
151588234 65547558> (DF)
22:56:02.663821 > n1.1360 > n2.qos: P 14481:15476(995) ack 1 win 32120
<nop,nop,timestamp 65547558 151588234> (DF)
22:56:02.664076 < n2.qos > n1.1360: . 1:1(0) ack 13033 win 30408 <nop,nop,timestamp
151588234 65547558> (DF)
22:56:02.666204 < n2.qos > n1.1360: . 1:1(0) ack 15476 win 31856 <nop,nop,timestamp
151588235 65547558> (DF)
22:56:02.700956 > n1.1360 > n2.qos: F 15476:15476(0) ack 1 win 32120 <nop,nop,timestamp
65547562 151588235> (DF)
22:56:02.701110 < n2.qos > n1.1360: . 1:1(0) ack 15477 win 31856 <nop,nop,timestamp
151588238 65547562> (DF)
22:56:02.701323 < n2.qos > n1.1360: F 1:1(0) ack 15477 win 31856 <nop,nop,timestamp
151588238 65547562> (DF)
22:56:02.701353 > n1.1360 > n2.qos: . 15477:15477(0) ack 2 win 32120 <nop,nop,timestamp
65547562 151588238> (DF)

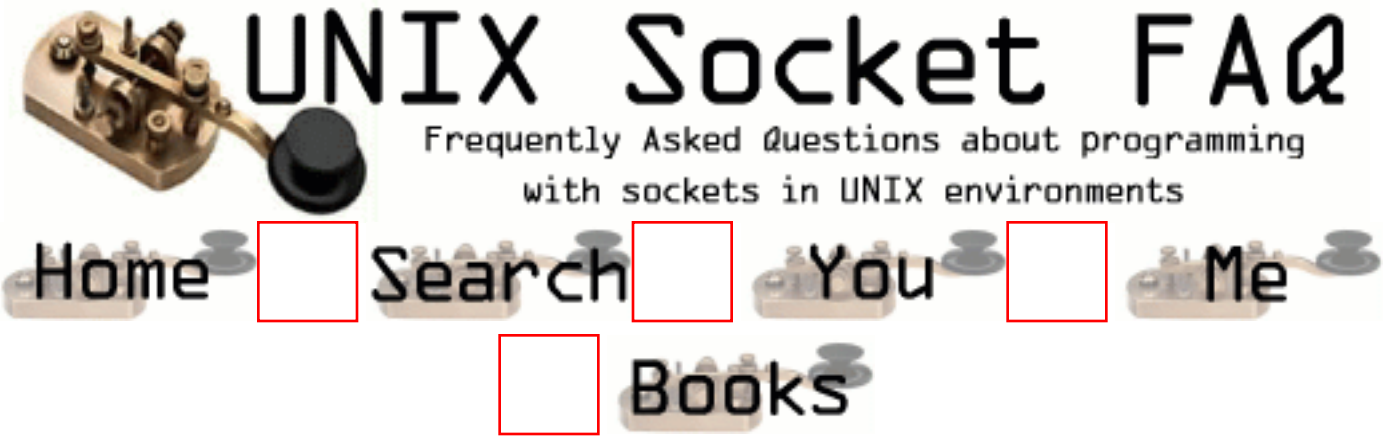
From: Rob Seace

That's not unusual... Your data is going to arrive in
separate packets, of a size based on the smallest MTU
between the source and destination... That's just the way
it works... If you want to get it all in one chunk, use a
blocking read of some sort that waits for the whole amount
to arrive... (Eg: recv() with the MSG_WAITALL flag, or a
local read() cover that loops until it has the full len...
This assumes you KNOW the length of the whole message on
the receiving end, too... If not, well, you'll need to
communicate that to the receiver, somehow...)

From: Chad

We effectively use 2 reads, one is for an int which is the size of the data to follow and the next read blocks waiting for data of that size.

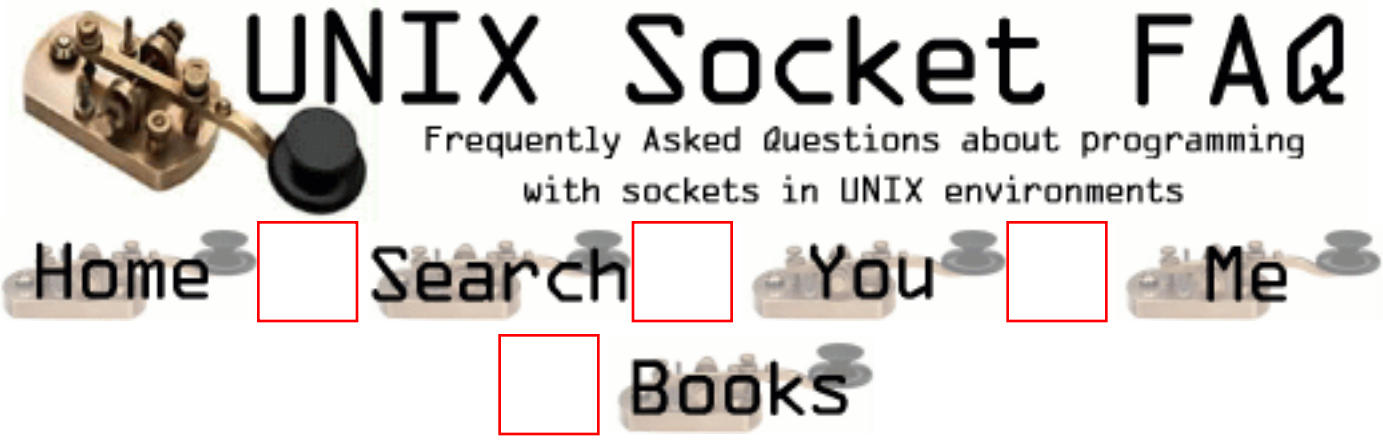
Add a Comment



New Questions: Problems with setting a timeout on connect

From: jb

Add a Comment



New Questions: setting a timeout on connect

From: jb

how would i go about doing this i've heard so many ways, 0block then use select, that didn't work for me, then
using signals and whatnot, can someone please set me straight heh

---

From: poop`

ok heres what ya do...first you open the socket and then u put your millimeter peter in it...OR...you can go buy a fufme and put your finger in one and have the other go up yer ass

---

From: jb's pimp

yeah i pimp jb for a living

From: jb's bigger pimp

na...u aint got nuthin on the grid....

From: jb's pimp pimp

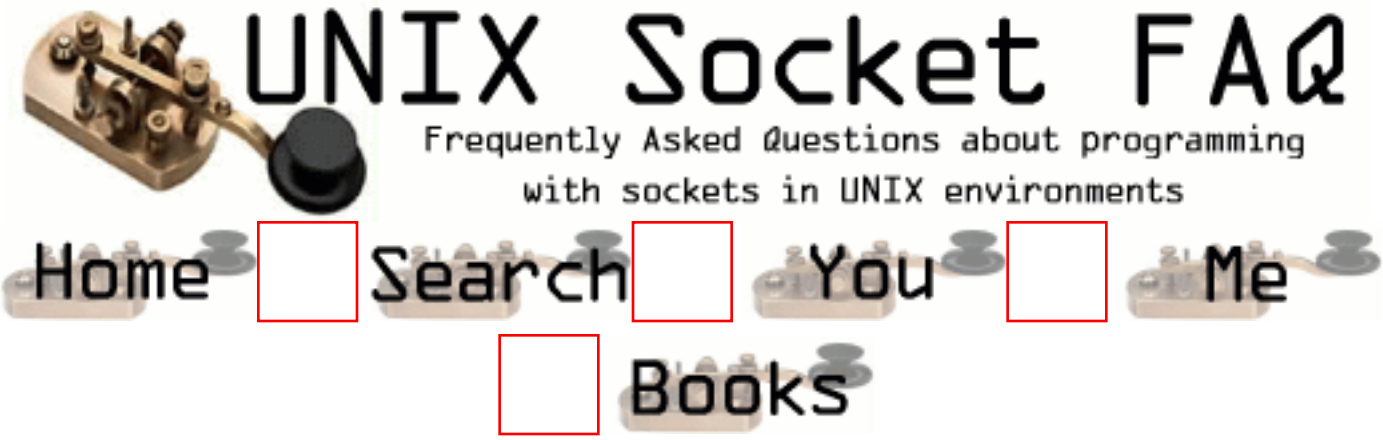
I P1Mp j00 ALL 4 l1f3 suck4'5!!!!!!!

From: lost

This is a unix faq board...not some lame highschool board..how about u answere his question u faggets

---

Add a Comment



New Questions: read() and MSS

From: chong

I find that my client write() two packets with 722 and 1028
bytes respectively, but the server read() two buffers
with 1448 and 302 bytes respectively. I think 1448 is the
MSS in the ethernet (1500-20-20-12), but why the server
reveive the two packets at the different pattern?

Thank you,
-Chong

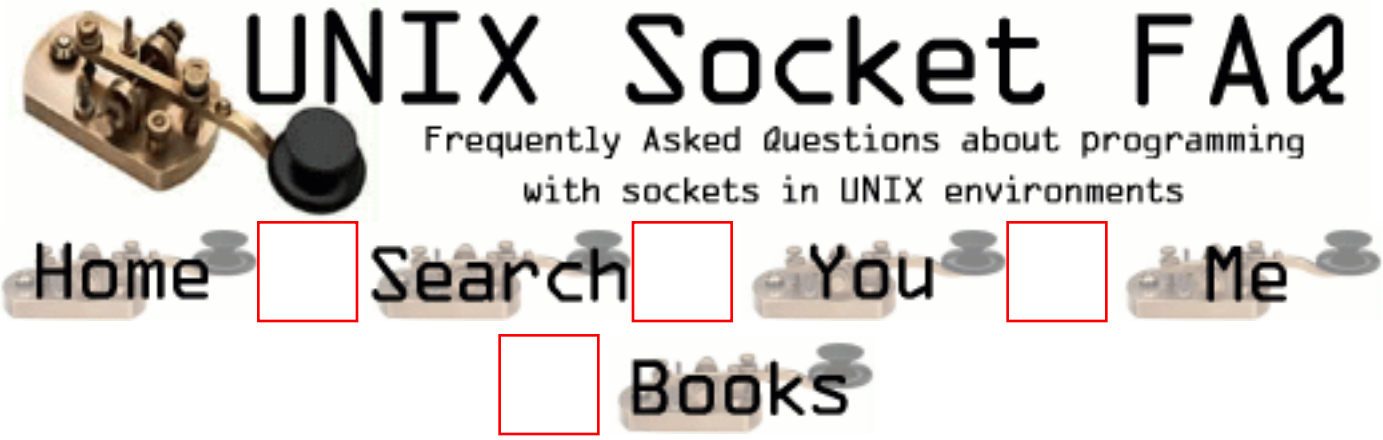
From: chong

I use select(fd, &rfds, NULL, NULL, NULL) at the server
side and use read() to read packet if the data available
from an existed socket.

From: Rob Seace

Probably, the client write()'s the two messages close enough
together in time that the TCP stack combines them, and then
maxes out the packets, for efficiency... This is normal...
You can't rely on receiving the same length message that was
sent, with TCP... It's a stream based protocol, not a
packet/datagram based one... If you need to know the size
of the message sent, you'll have to send that separately,
prior to each message, or something...

Add a Comment



New Questions: unable to establish socket connection

From: Sarah

Hi, I'm trying to connect to my unix server(I'm brand new to this) and I keep getting the error, unable to establish socket connection. Any ideas? Thanks

---

From: Sarah

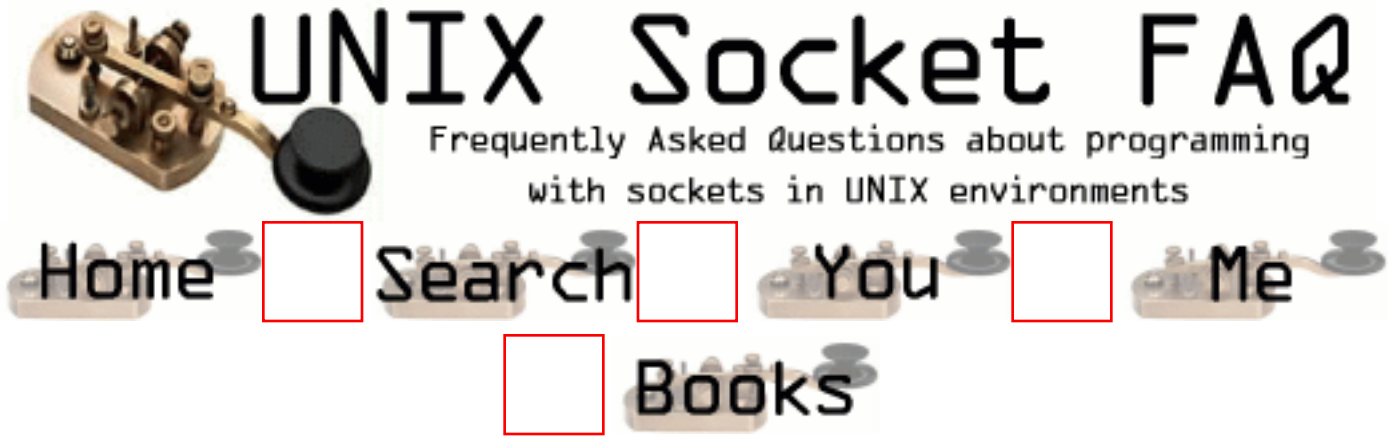
I fixed it thanks

From: issam qasas

how do you fix it ?

thanks

Add a Comment



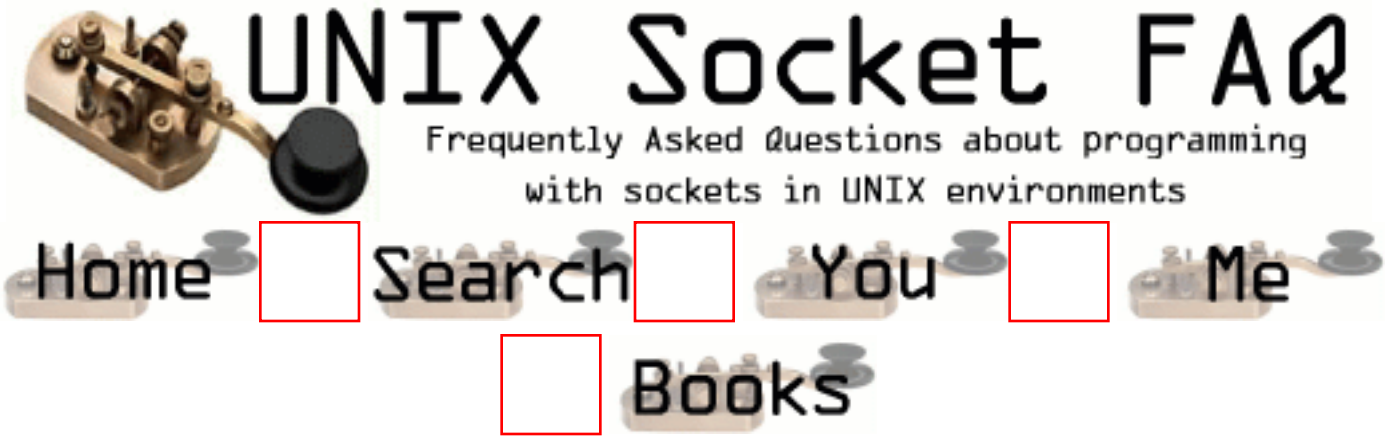
New Questions: exception in java

From: frank

Is it possible that socketexception in java shuts down the entire webserver. Below you can find the exception message I received:

HANDLER THREAD PROBLEM: java.net.SocketException: Connection reset by peer
java.net.SocketException: Connection reset by peer
com.sun.web.core.DefaultServlet: init
at java.net.SocketInputStream.read(Compiled Code)
at java.net.SocketInputStream.read(Compiled Code)
at com.sun.web.server.ServletInputStreamImpl.read(Compiled Code)
at javax.servlet.ServletInputStream.readLine(Compiled Code)
at
at
at com.sun.web.server.ConnectionHandler.run
(Compiled Code)
Endpoint ServerSocket[addr=0.0.0.0/0.0.0.0,port=0,localport=8080] shutdown due to exception: java.net.SocketException: No buffer space available
endpoint down: localhost/127.0.0.1:8080

Add a Comment



New Questions: How could I know that the client stopped sending the data and waiting for me to response?

From: Dimitry

What is the reliable way to know that the client stopped sending the data and waiting for me(server) to response?

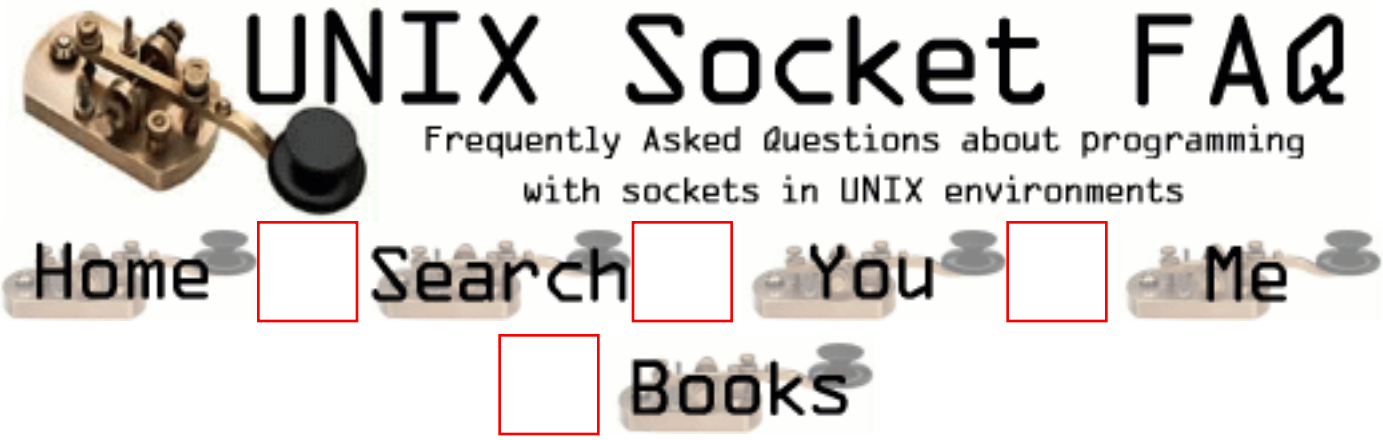
Does the POLLOUT event on the server side means that that
the client stopped sending the data and blocked in recv()/select()/poll()?

Thank you!

From: K E V I N

Also, what kind of function can catch those events such as when data is comming?? so that I can call recv() or send()...
how do I do to let my socket client and server can communicate "simultaneously"?? I mean when data is comming, it will automatically calls recv(); and when I input, it calls send()....

Add a Comment



New Questions: recv returns 0!

From: Dimitry

Hi there!

oes the receiving zero form recv() means that the client closed the connection?
Does it 100% reliable?

I mean - can I receive zero, but the client does not close the connection?

TIA

From: g.jagadish

Yes, you are received 0 bytes, the Client and Server connection is alive.
-jsh

From: Rob Seace

No, I don't think that's true at all... I can think of no
circumstances where it would ever be possible for a read()
(or recv()) to return 0, unless the other end of the socket
were closed... A read() return of 0 ALWAYS means EOF, as
far as I know... Unless you know something I don't... If
so, please share...

From: jayant jain

Well you can receive a read of 0 and the error number will be set to EAGAIN..

Youhave to check for the error number before you come to conclusions.

happens all the while..

From: Hector Lasso

When a socket is non-blocking, and you call recv(), it will return -1 (indicating an error) and errno will be set to EAGAIN when there is no data ready to be read for the socket. I think Jayant is wrong about this.

The recv() documentation doesn't say that you should expect 0 when you get the EOF, however, read() does return 0 when EOF.

From: David Bruhn

On a blocking socket, recv() returns zero if the peer closed the connection.
On a non blocking socket, recv() returns zero if the peer closed the connection or if there is no data to read.
FYI I include this quote from W. Richard Stevens, UNIX Network Programming, vol.1, 2nd edition, chapter 15:
"Traditionally, System V has returned the error EAGAIN for a nonblocking I/O operation that cannot be satisfied while Berkley-derived implementations have returned the error EWOULDBLOCK. To confuse things even more, Posix.1 specifies that EAGAIN is used while Posix.1g specifies that EWOULDBLOCK is used. Fortunately, most current systems (including SVR4 and 4.4BSD) define these two error codes to be the same (check your system's <sys/errno.h> header), so it doesn't matter which one we use."

From: Rob Seace

Are you sure about recv() returning 0 for a non-blocking socket, when no data is ready?? That goes against all documention I've ever seen... Just for example, to quote from my current system's (Red Hat 6.2 Linux) man page for recv(), recvmsg(), and recvfrom():

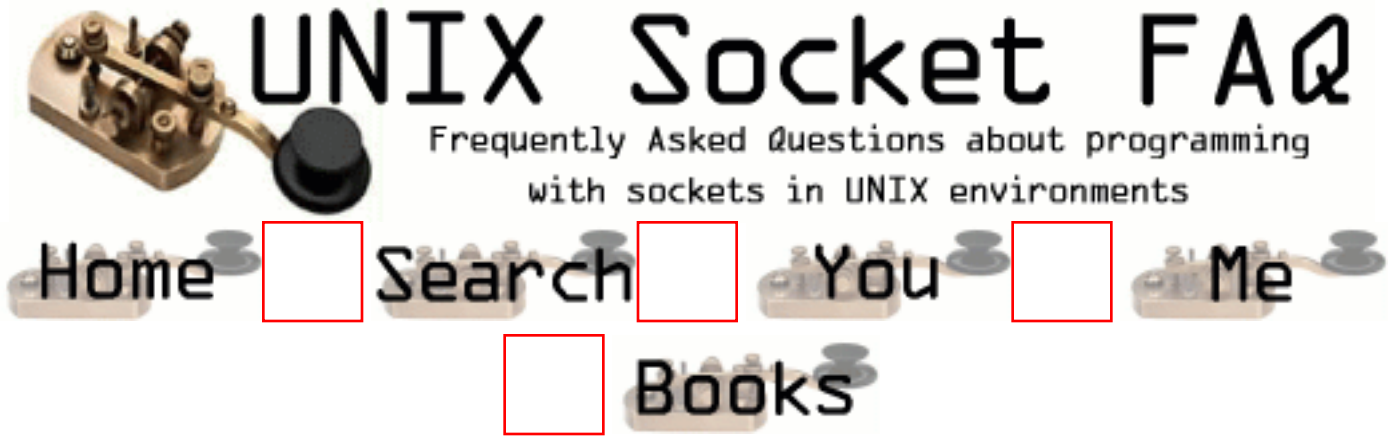
"If no messages are available at the socket, the receive calls wait for a message to arrive, unless the socket is nonblocking (see fcntl(2)) in which case the value -1 is returned and the external variable errno set to EAGAIN."

So, I stand by my initial statement: a return of 0 from either read() or recv() ALWAYS indicates EOF, which on a socket means the peer closed the connection... Whether the FD is blocking or non-blocking... I'd say any system where this isn't true is broken...

From: David Bruhn

Rob, you are correct. I unfortunately posted behavior of a Roguewave implementation of recv() on a nonblocking socket. Thanks for clearing this up. I'm rewriting my code to use the socket APIs.

Add a Comment



New Questions: writing into file

From: Pratibha

Hi
My server application reads from and updates a text file. How di I synchronise the file operations so as to enable mutliple clients to access the file simultaneously?

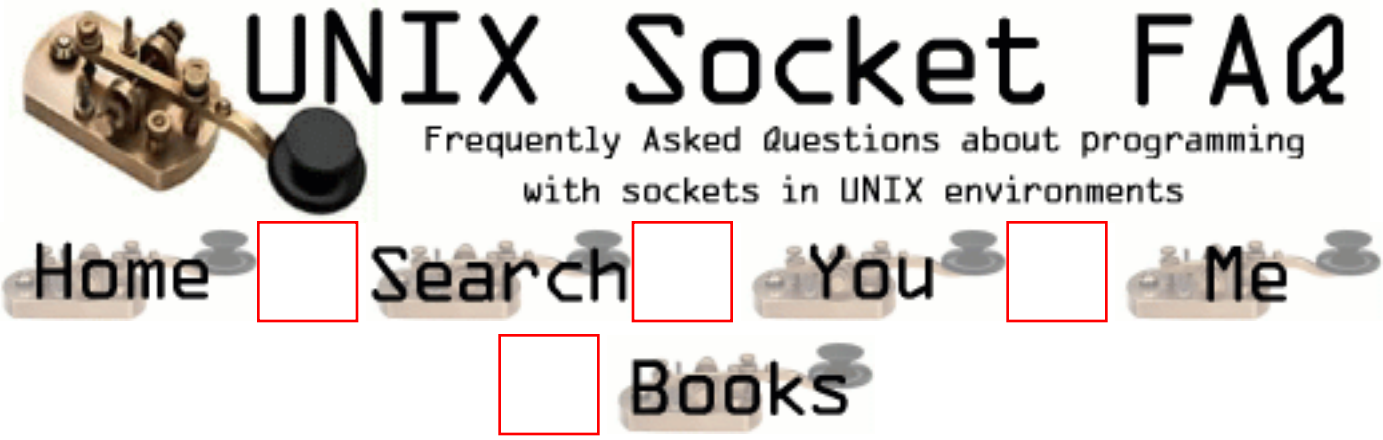
From: Muslim Moiz Ghadiali

put your program in a continuous while loop and keep forking new children each time a client gets connected.remember that the accept function comes outside the loop and after forking the new child close the old socket and use the new one created by 'accept'.

From: ornaz

use mutex - A mutex is a MUTual EXclusion device,
and is useful for protecting shared data structures
from concurrent modifications, and implementing critical sections and monitors.
see man f.ex. on pthread_mutex_init

Add a Comment



New Questions: TCP/IP called address for a ANY_HOST listen

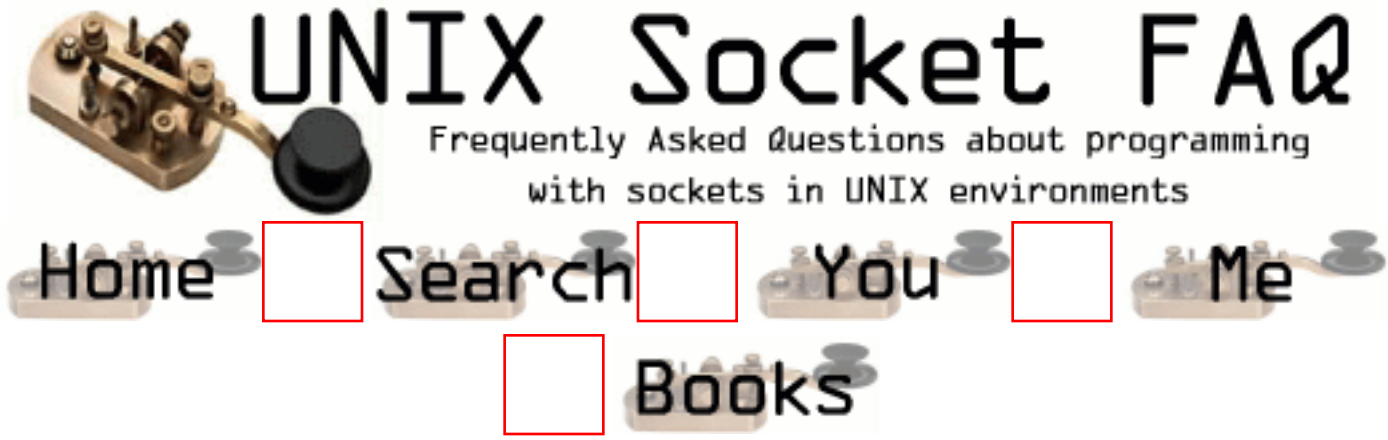
From: Leclercq

Is it possible to know the called ip address on a multi ip address machine when the the listen has been done on ANY_HOST ?

From: Rob Seace

Yeah, I believe just doing a getsockname() on the connected
socket (the one returned from accept()) should give you the
local IP that it came in on...

Add a Comment



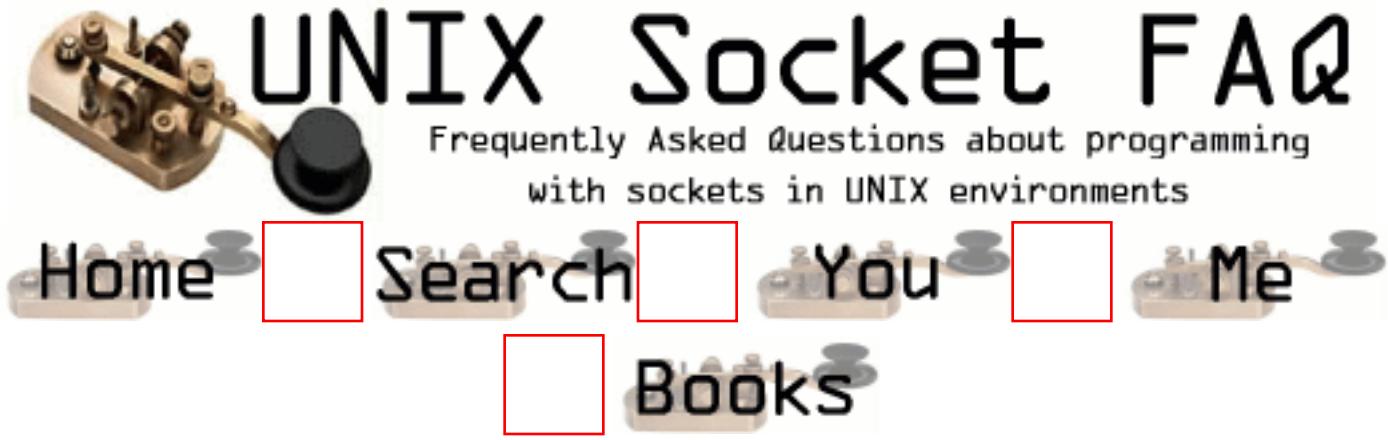
New Questions: VB client to Perl Server

From: Lee Turner

How do you send data to a local port using VB and have a perl server accept the connection. I am trying to use Winsock in Vb but the 'Accept' command in Perl does not work in conjuncion with VB's 'Connect' command

---

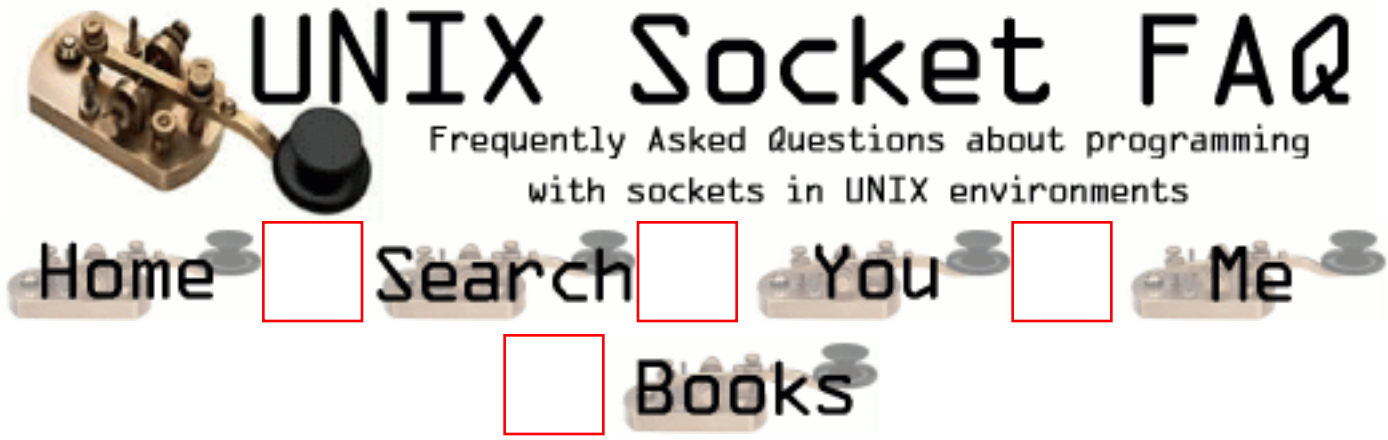
Add a Comment



New Questions: How can i make a server act as connection oriented as well as connection less at same time

From: Eswar

Add a Comment

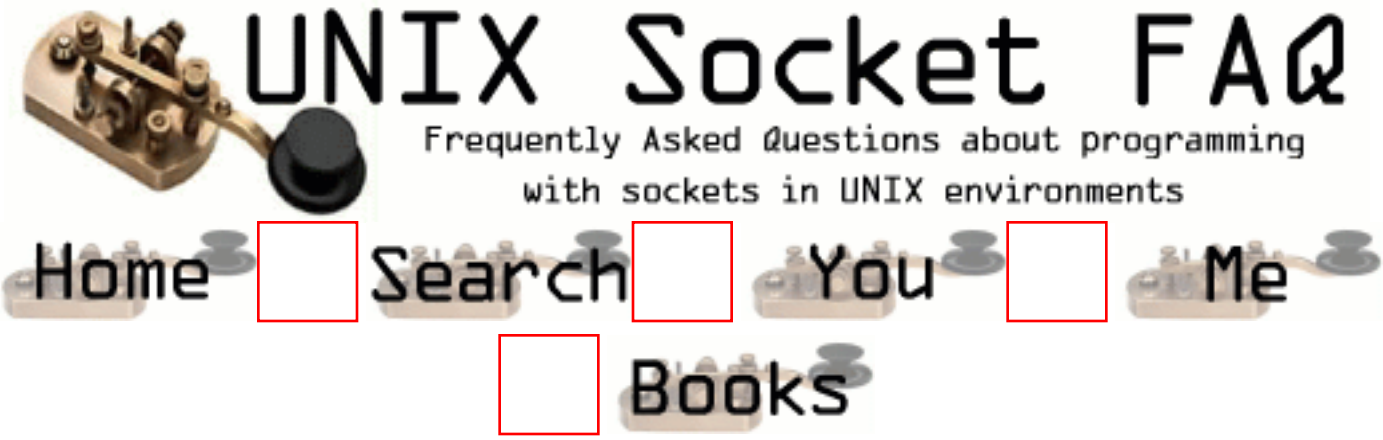


New Questions: Cannot create UDP socket - port number in use

From: David Friedman

I have an application (in Java, JDK 1.2 under Linux) that runs on multiple machines. They communicate using datagram sockets with a fixed port number (say, 29000). Occasionally the program will abort (I'm still debugging it) without closing the socket. THe next time I try to start the program, I get a Java socket exception: address in use. When I use netstat, I see the socket listed with my port number. The only way I've been able to kill it (because the object reference is gone) is to reboot. Is there a way I can delete the old socket (e.g., via a Linux command) before trying to create a new one? Alternately, can I set the "re-use address" on the Java datagram socket without making it a multicast socket? (Java automatically sets the re-use address property on multicast sockets, or so says the JDK 1.2 documentation.)

Add a Comment



New Questions: socket c programming

From: matt

Hi,
I am writing a program that does a bunch of stuff with MQSeries and it needs to be compatible with UNIX and NT. My question is: Is there a C command that I can get the "hostname" of the system w/o having to go to the command prompt and do it a command?

Thanks,
matt

From: Rob Seace

Either gethostname() or uname() should do the job for you...
The former is pretty common on all Unixes I've seen, but the
latter is only one that's POSIX compliant, I believe...

Add a Comment

UNIX Socket FAQ

Frequently Asked Questions about programming with sockets in UNIX environments

New Questions: question

From: allen

I need to get a file size (C language)

Add a Comment

New Questions: How can i send communication break

From: Arul Anand.J

I am unable to send a communication break in TruUNIX & SCO open server i tried with the following functions

1) ioctl with TCSBRK (arg. 0)
2) ioctl with TCXONC

Add a Comment

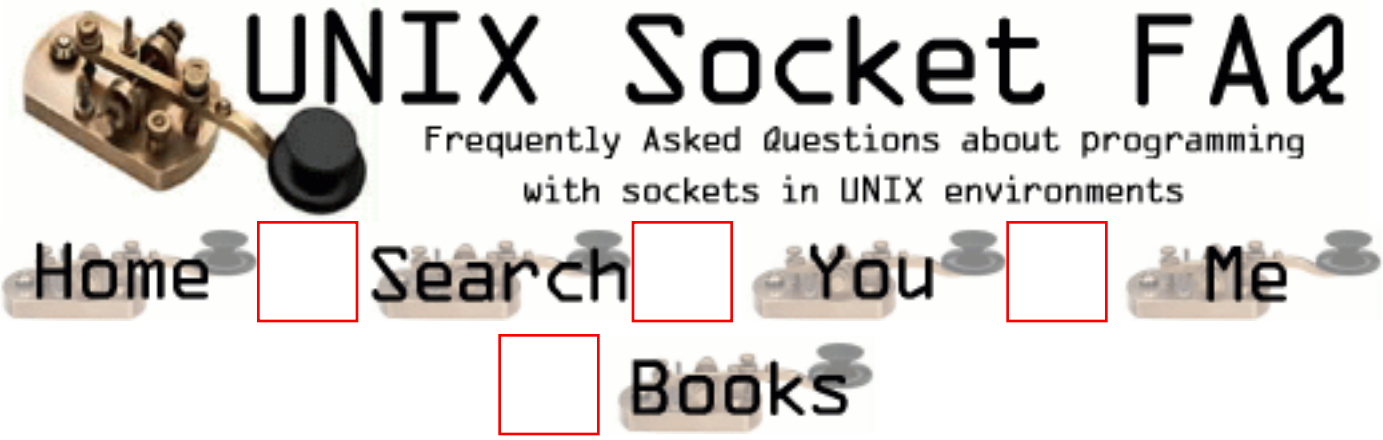
New Questions: How can i send communication break

From: Arul Anand.J

I am unable to send a communication break in TruUNIX & SCO open server i tried with the following functions

1) ioctl with TCSBRK (arg. 0)
2) ioctl with TCXONC (arg. 0 , some delay & arg. 1)
3) tcsendbreak

Add a Comment



New Questions: Data comes together

From: Rafael Rocha

I have an aplication that consists of a Linux server and a Vb client.

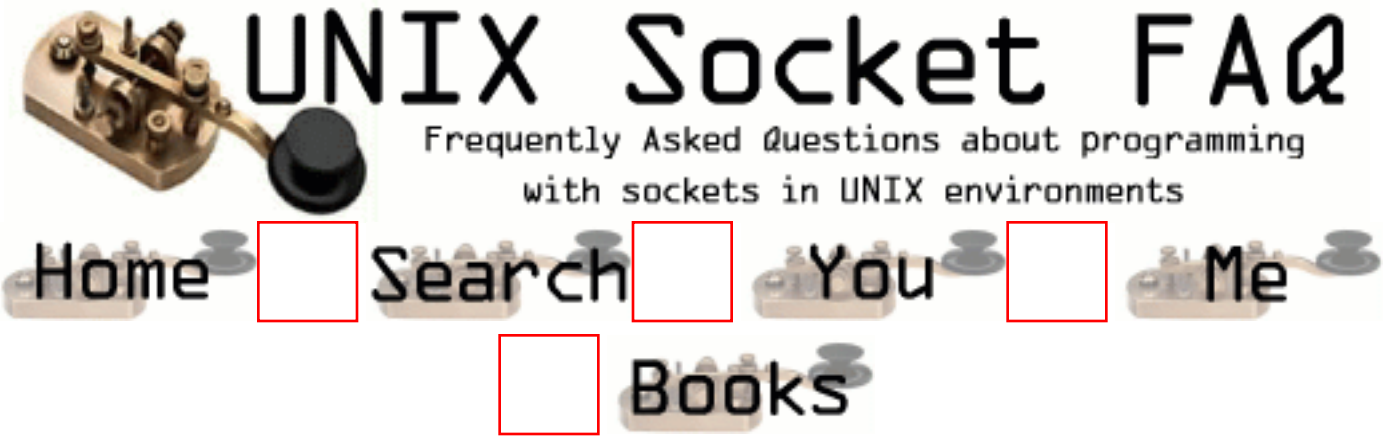
I am doing a stream to the client. It is workinf fine.

The problem is that when the connection is thought a modem, sometimes data arrives together.

No data is being lost, but the VB aplication thing so. Because it receives more than one data at the ssame time.

Thanks in advance.
Rafael.

Add a Comment



New Questions: Link Error.

From: Carlos A

I've read your question at Unix Socket FAQ and noticed that you have some experience programming sockets on Solaris. Currently I'm programming a client-server app using sockets and when compiling I get this:

[netflow]:root> gcc radser.c -o radser
Undefined first referenced
 symbol in file
socket /var/tmp/cca003Jq1.o
gethostbyname /var/tmp/cca003Jq1.o
bind /var/tmp/cca003Jq1.o
sendto /var/tmp/cca003Jq1.o
inet_ntoa /var/tmp/cca003Jq1.o
recvfrom /var/tmp/cca003Jq1.o
ld: fatal: Symbol referencing errors. No output written to radser

Do you know how I can solve this?

Thanks,
Carlos A

Add a Comment

New Questions: demon.

From: Facundo

Hi!

Do you know, how i can do a demon from my program in c,unix.

Fac.

From: Rob Seace

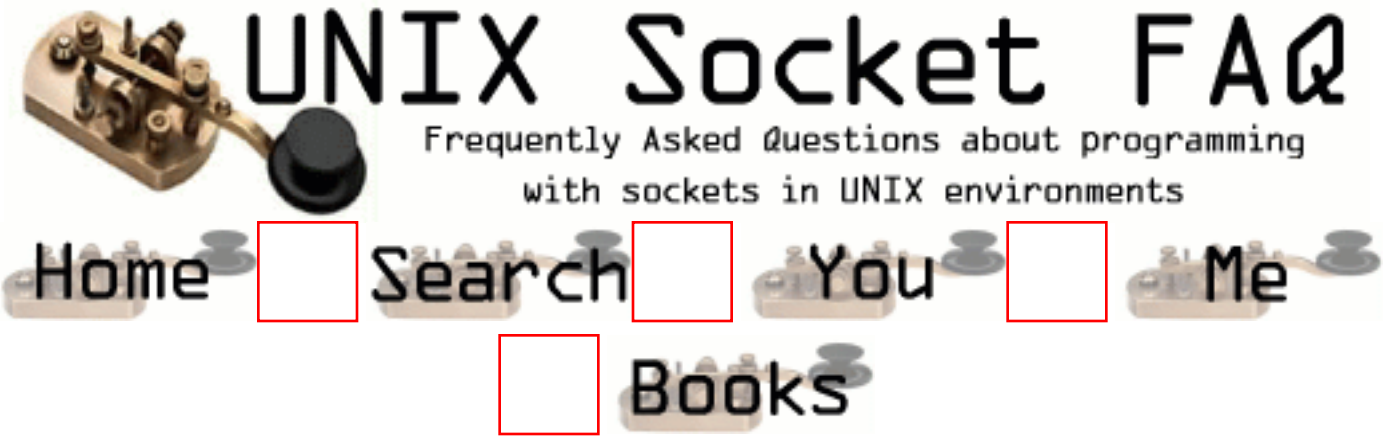
Well, I'll assume by "do a demon", you mean "become a
daemon"... ;-) This is actually covered already in the FAQ,
in <A HREF="http://www.lcg.org/sock-faq/detail.php3?id=42">
section 4, question 3</A>... But, I don't particularly like
how that code just close()'s stdin/stdout/stderr, either; I
always like to have valid, open descriptors for 0, 1, and 2,
just in case some library functions try to write to them, so
they aren't trashing anything else you may have opened,
which just happened to use those FDs... So, I always open
"/dev/null" for stdin, stdout, and stderr, in my daemons...
I also like to actually fork() TWICE, once after the
setsid(), just to abandon session-leader status, and assure
that we can never regain a controlling tty... Roughly, and
without error checking, I like to do essentially this:

```
signal (SIGHUP, SIG_IGN);
if (fork())
    exit (0);
setsid ();
if (fork())
    exit (0);
freopen ("/dev/null", "r", stdin);
```

```
freopen ("/dev/null", "w", stdout);
freopen ("/dev/null", "w", stderr);
chdir ("/");
signal (SIGINT, SIG_IGN);
signal (SIGQUIT, SIG_IGN);
signal (SIGPIPE, SIG_IGN);
signal (SIGCHLD, zombie_reaper);
```

Where zombie_reaper() is just a simple signal handler to clean up any zombie child processes:

```
void zombie_reaper (int sig)
{
   while (waitpid (-1, NULL, WNOHANG) > 0)
      ; /* don't fear the reaper ;-) */
}
```



New Questions: MX lookup

From: Mikael Chambon

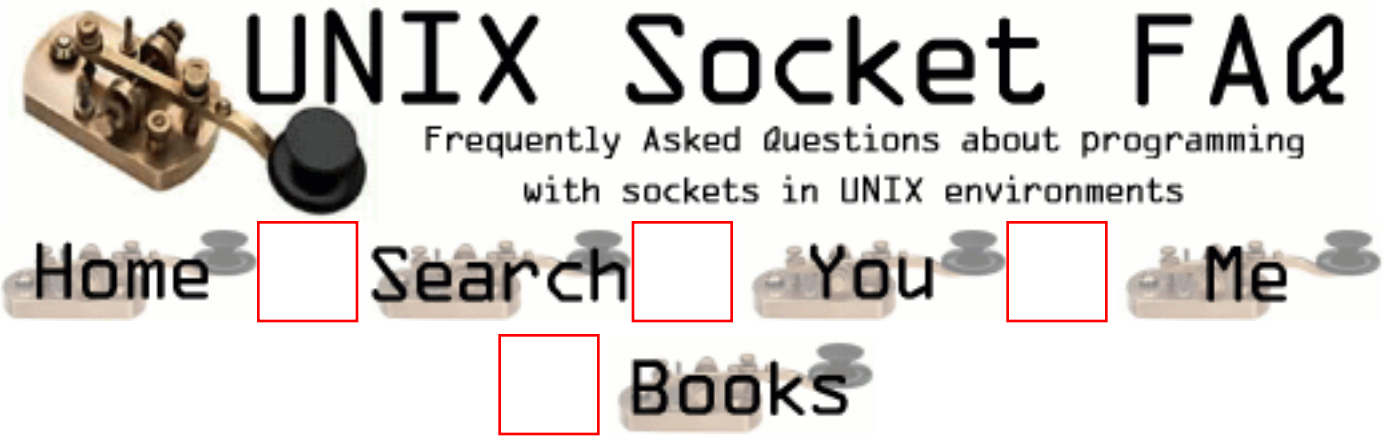
Hello, does someone know how to do a MX lookup in C ??
I want to get the server mail adress from a domain name ..

Thx

From: zhan dong

try download source code "bind"
there should be a sample program dnslookup or other
can help you.

Add a Comment



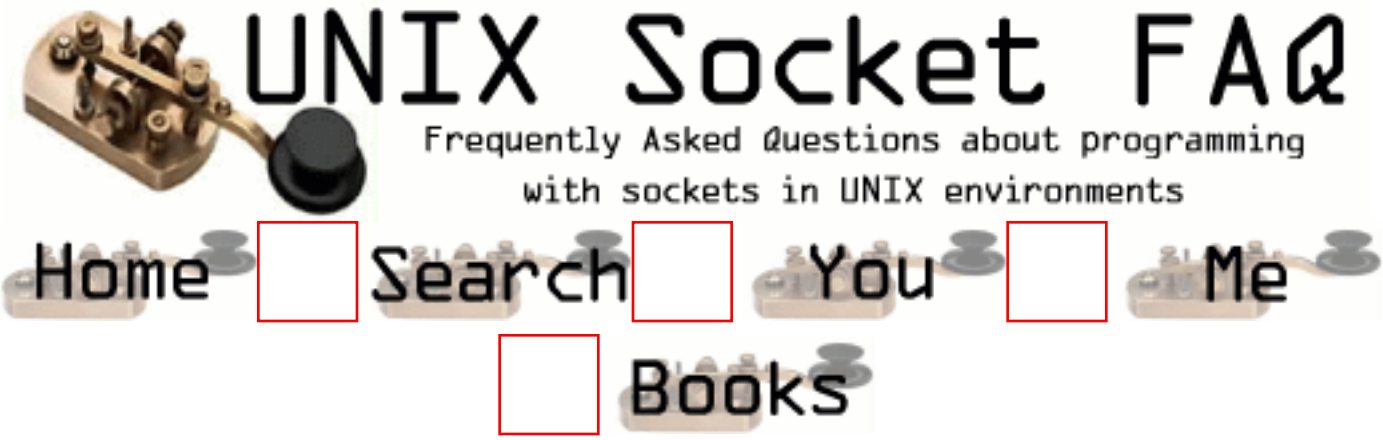
New Questions: hostname

From: ethan hunt

Hello.
I need a command (used in a c program) that works for unix and nt, that gets the hostname. gethostname() and uname() work for unix but not for nt. does anyone know if there is a command that can be used for nt, and aix and solaris boxes?

thanks.
ethan.

Add a Comment



New Questions: accept() hangs

From: Haim Lichaa

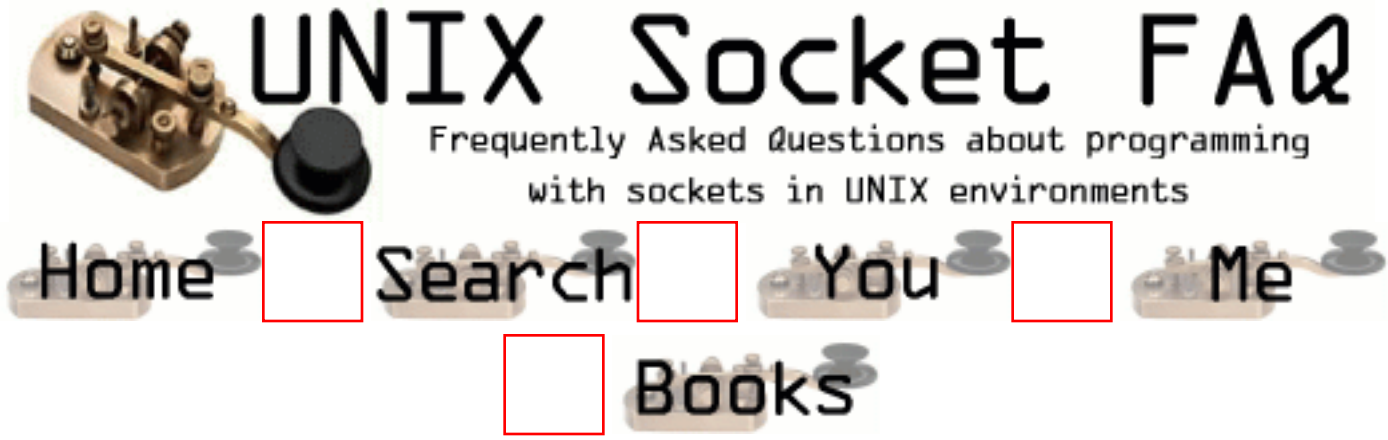
I wrote a small client/server application,upon server initialization and after successfully preforming bind() and listen() my application hangs on accept() and won't return from this socket function. Can anyone tell me why?

From: Haim

From: Haim Lichaa

I think I found the problem.
I used the getservbyname() function which accepts the service and returns an address to the port via a structure that is passed to it (the same return value as htons() when passing a port number to it), and passed the getservbyname() modified structure field to htons().
In summary, when using getservbyname() there is no need to use htons(). Use the value in the structure directly when opening a socket.

Add a Comment



New Questions: Semantics of ioctl with FIONREAD and recv

From: Yannis Pavlidis

Hi,

I am facing the following situation. Select is triggered and is informing me that there is data in the socket to be read. Before going to read data I call ioctl(fd, FIONREAD, &nLen) to determine how much to read. The ioctl returns 0 (based on the man page it is correct) and the value of nLen is zero. When I call the recv of course it will return zero. Based on the semantics of the recv it means that the socket was closed gracefully by the other end.

Is this right?
My second question is why does the server says that the socket was closed? I am 100% that the client does not do something like this. Which are the causes. The platform is redhat 6.2

Any help would be highly appreciated.

Thanks a lot for your time.

Yannis.

From: Rob Seace

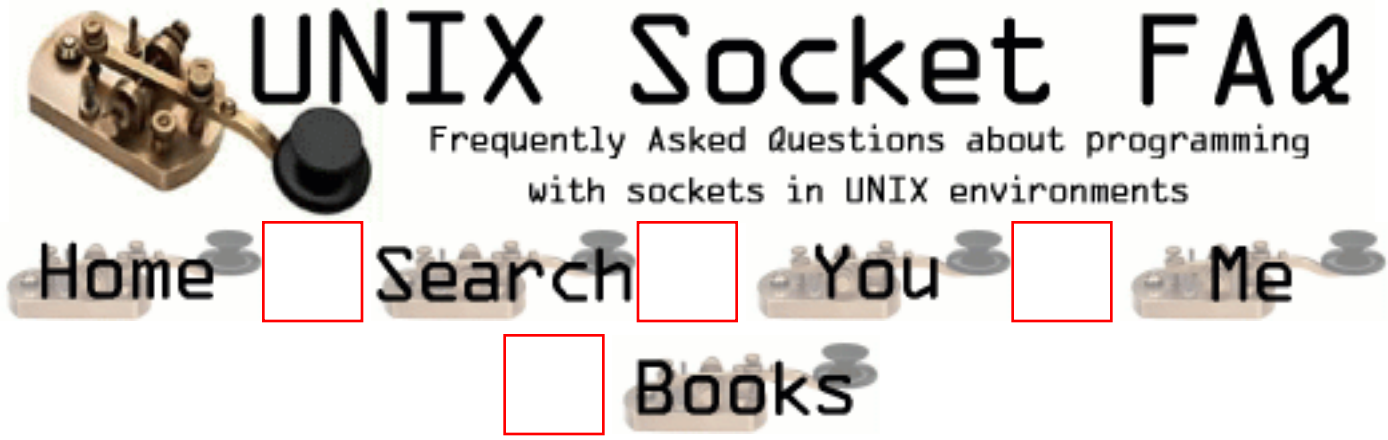
Are you actually calling the read()/recv() to verify that it really DOES return 0? Because, I can't testify to the accuracy of that ioctl() call you mention; I've never used it, myself... But, yes, when read()/recv() returns 0, I think the only way that should ever happen is if the socket has been closed (or, at least half of it has)... Now, if you're sure the remote end isn't closing the socket, you may want to run both the server and the client under "strace"

and/or "ltrace" (or, similar programs that will trace system
and/or library calls), and see exactly what's really going
on... I bet you'll find the socket IS getting closed,
somehow... Or, perhaps it's a network problem, and your
connection between the server and client is going away?
Running a packet sniffer (Red Hat comes with "tcpdump") may
also yield useful information, as to what is happening...
If you see an incoming FIN from the client, it IS closing
the connection... If you see something like an incoming
RST, maybe the network is flaking out, or some firewall or
something between server and client is screwing with the
traffic... Etc... There are a many possibilities, so it's
difficult to really guess exactly what's going on...

From: Florin

I'm having the same problem with ioctl.
What do I do when SOCKET_ERROR!=ioctl(socket, FIONREAD, &nbytes) and nbytes is 0 after the call? If I try to recv 0 bytes, recv will return 0 and I have to throw (in C++ :-).
I thought that this might mean that no data has arrived, so I select() the socket, which returns 1 (meaning my (only) socket is readable) and call ioctl() again, which gives me nbytes=0 again; so I loop forever and I don't like it.
How do you really do this, after all?

Add a Comment



New Questions: Server Application

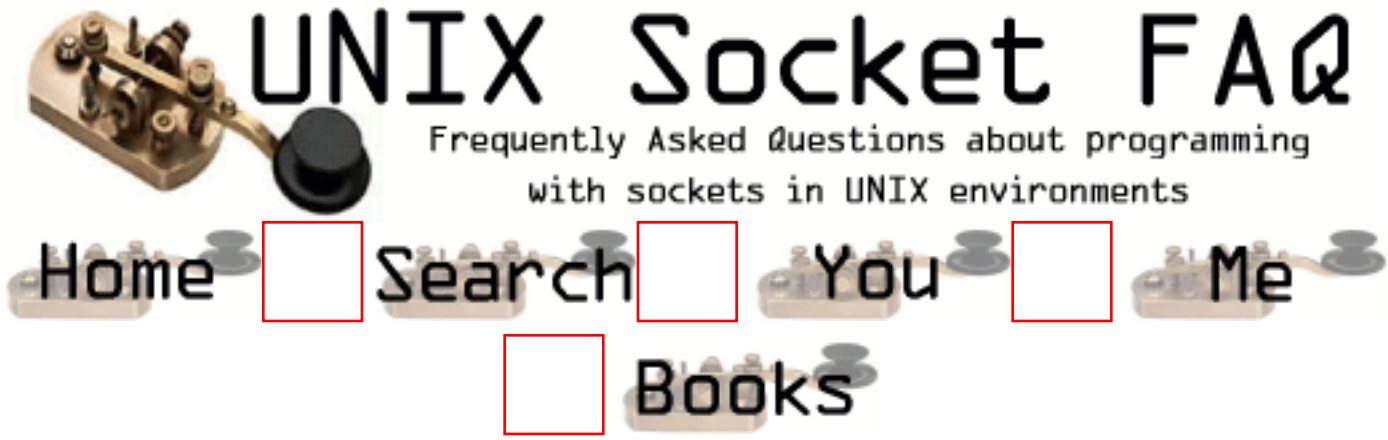
From: Yigal Petreanu

Is there a way to pass a socket from a server to another process (which is not its child!), without the client knowing about it ?

From: Rob Seace

Possibly, but I don't believe it's standardized, so it's probably just dependent on whether or not your OS supports it... But, one method that seems to be supported fairly widely is passing file descriptors (including socket FDs) to other processes via a Unix domain socket, using sendmsg()/recvmsg(), with eg. "SCM_RIGHTS" as the control message type, and the FD(s) as ancillary data... Perhaps, check the man pages on your system for "sendmsg", "recvmsg", and maybe "cmsg" (on my Red Hat box, the latter gives the most helpful info, with a bit of sample code on how to construct a control message for passing FDs)...

Add a Comment

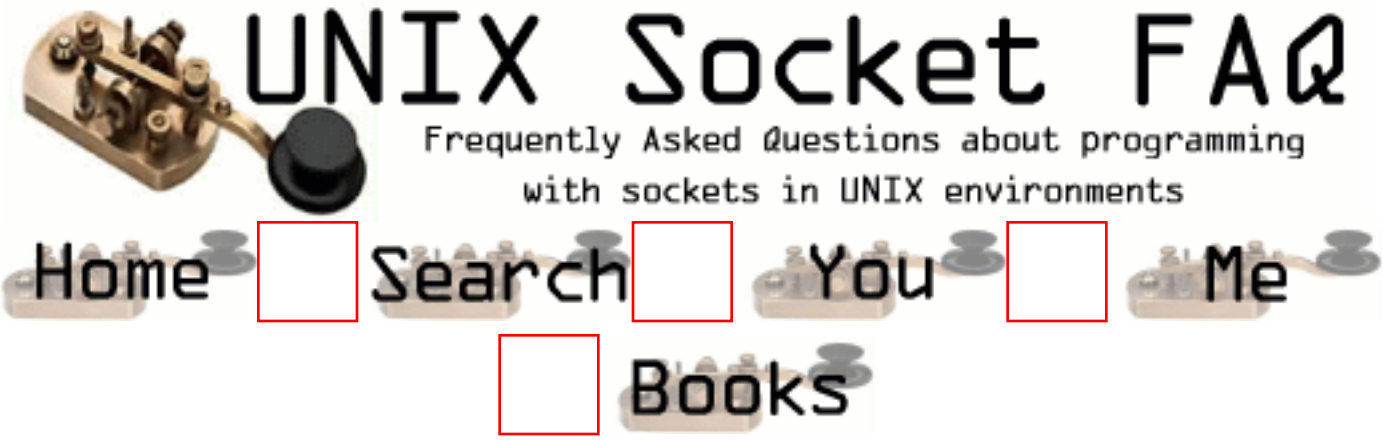


New Questions: WWW

From: Daniel

My unix workstation is unable to surf the net or ping and external IP address after reinstallation. Could someone help me to solve the problem? Thanks a million!

Add a Comment



New Questions: TCP/IP does not preserve message boundaries

From: CECE

Hello,

I have a problem. As you all know, tcp does not preserve message boundaries. But, in my project, I have to separate each single message from the others. And I can't use a header (which may include the real length of the message) for the messages sent and received, because the senders are not in control of me. If any of you know something to handle this condition, please give me some detailed explanation.

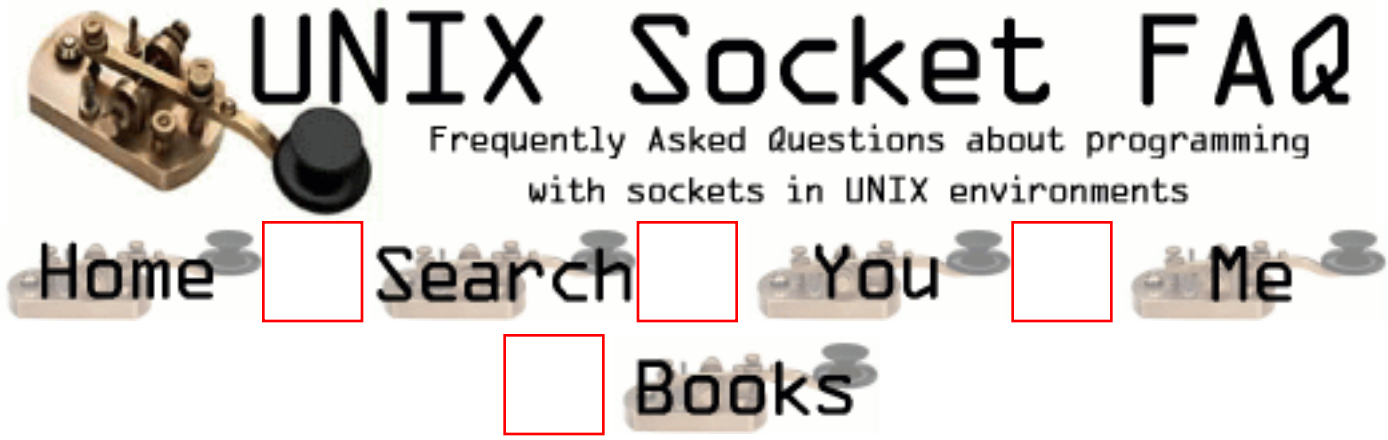
From: Rob Seace

Well, if you can't prefix the messages with a length header, the only other real choice would be to have some "end of message" marker, which you could read at the end of each message... (Eg: a newline character, or a null character, or something like that...) But, it sounds like you are not in control of the data being sent to you? If that's the case, you probably can't force them to terminate the messages correctly, any more than you could force them to prefix them with a length header... So, in such a case, I really don't think there's anything you CAN do, really... Aside from just using UDP, which will keep the messages as distinct, individual messages... (But, then, you're stuck with the inherent unreliability of UDP too, which you probably don't want...) But, TCP is stream based, so there's just not a lot you can do about it, unless you can control the format of the data you are receiving...

From: Arthur Lung

I'd say that if you have no control over what's being sent to you, and you need to preserve message boundaries, but the people sending it won't do it for you, then you either don't need to know message boundaries or someone needs to be smacked in the head with a stick for screwing you over.

Add a Comment



New Questions: MX lookup

From: Mikael Chambon

Hello, can someone give me an example for a Mx lookup.

I Want to get the mail server ip from a domain name,
I know that I should use res_query and then dn_expand function

but I seems complicated to make it works?

When I do a res_query, I got an answer from the DNS but
It's seems to be a compressed answer and then I try to uncompress
it with dn_expand but it doesn't work, if someone
could give me an exmaple, it would be sosososos great..

Thx for all.

From: zhan dong

```
#include <sys/types.h>
#include <sys/socket.h>

#include <netinet/in.h>
#include <arpa/nameser.h>
#include <arpa/inet.h>

#include <errno.h>
#include <netdb.h>
#include <resolv.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
```

```
#include <unistd.h>


extern int errno;
extern int h_errno;
extern char *h_errlist[];


int
ns_sprintrrf1(const u_char *msg, size_t msglen,
const char *name, ns_class class, ns_type type,
u_long ttl, const u_char *rdata, size_t rdlen,
const char *name_ctx, const char *origin,
char *buf, size_t buflen);
static size_t prune_origin(const char *name, const char *origin);
static int charstr(const u_char *rdata, const u_char *edata,
char **buf, size_t *buflen);
static int addname(const u_char *msg, size_t msglen,
const u_char **p, const char *origin,
char **buf, size_t *buflen);
static void addlen(size_t len, char **buf, size_t *buflen);
static int addstr(const char *src, size_t len,
char **buf, size_t *buflen);
static int addtab(size_t len, size_t target, int spaced,
char **buf, size_t *buflen);


struct __res_state res;

char*
getmxname(char* name1) {
u_char *msgbase;size_t msglen;
u_char *rdata;size_t rdlen;
u_long ttl;
char name[MAXDNAME];

u_char answer[8*1024];
int c, n, i = 0;
u_int32_t ul;
int nameservers = 0, class, type, len;
struct in_addr q_nsaddr[MAXNS];
struct hostent *q_nsname;
HEADER *hp;
int stream = 0, debug = 0;
ns_msg handle;
```

```
int success, proto_type;

int sflag, rrnum;
static char buf[2048];
ns_opcode opcode;
ns_rr rr;
int ret;


/* set defaults */
len = MAXDNAME;
gethostname(name, len);
class = C_IN;
type = T_ANY;

strcpy(name, name1);

proto_type = sym_ston(__p_type_syms,
"MX", &success);
if (success)
type = proto_type;
else {
return NULL;
}

len = sizeof(answer);

if (!(res.options & RES_INIT))
if (res_ninit(&res) == -1) {
fprintf(stderr, "res_ninit() failed\n");
return NULL;
}

  res.options |= RES_USEVC;

if (nameservers != 0) {
res.nscount = nameservers;
for (i = nameservers - 1; i >= 0; i--) {
res.nsaddr_list[i].sin_addr.s_addr = q_nsaddr[i].s_addr;
res.nsaddr_list[i].sin_family = AF_INET;
res.nsaddr_list[i].sin_port = htons(NAMESERVER_PORT);
}
}

if (name[strlen(name) - 1] == '.') {
```

```
n = res_nquery(&res, name, class, type, answer, len);
if (n < 0) {
fprintf(stderr, "Query failed (h_errno = %d) : %s\n",
h_errno, h_errlist[h_errno]);
return NULL;
}
} else if ((n = res_nsearch(&res, name, class, type,
answer, len)) < 0) {
fprintf(stderr, "Query failed (h_errno = %d) : %s\n",
h_errno, h_errlist[h_errno]);
return NULL;
}

if( 0 != ns_initparse(answer, n, &handle)){
fprintf(stderr, ";; ns_initparse: %s\n", strerror(errno));
return NULL;
}

if (ns_parserr(&handle, ns_s_an, 0, &rr) == 0){

n = ns_sprintrrf1(ns_msg_base(handle), ns_msg_size(handle),
ns_rr_name(rr), ns_rr_class(rr), ns_rr_type(rr),
ns_rr_ttl(rr), ns_rr_rdata(rr), ns_rr_rdlen(rr),
NULL, NULL, buf, sizeof buf);
if(n<0){
fprintf(stderr, ";; ns_sprintrr: %s\n",
strerror(errno));
return NULL;
}
return buf;
}
return NULL;
}

static size_t
prune_origin(const char *name, const char *origin) {
const char *oname = name;

while (*name != '\0') {
if (origin != NULL && ns_samename(name, origin) == 1)
return (name - oname - (name > oname));
while (*name != '\0') {
if (*name == '\\') {
name++;
if (*name == '\0')
```

```
break;
} else if (*name == '.') {
name++;
break;
}
name++;
}
}
return (name - oname);
}

static int
charstr(const u_char *rdata, const u_char *edata, char **buf, size_t *buflen) {
const u_char *odata = rdata;
size_t save_buflen = *buflen;
char *save_buf = *buf;

if (addstr("\"", 1, buf, buflen) < 0)
goto enospc;
if (rdata < edata) {
int n = *rdata;

if (rdata + 1 + n <= edata) {
rdata++;
while (n-- > 0) {
if (strchr("\n\"\\", *rdata) != NULL)
if (addstr("\\", 1, buf, buflen) < 0)
goto enospc;
if (addstr((const char *)rdata, 1,
buf, buflen) < 0)
goto enospc;
rdata++;
}
}
}
if (addstr("\"", 1, buf, buflen) < 0)
goto enospc;
return (rdata - odata);
 enospc:
errno = ENOSPC;
*buf = save_buf;
*buflen = save_buflen;
return (-1);
}
```

```
static int
addname(const u_char *msg, size_t msglen,
const u_char **pp, const char *origin,
char **buf, size_t *buflen)
{
size_t newlen, save_buflen = *buflen;
char *save_buf = *buf;
int n;

n = dn_expand(msg, msg + msglen, *pp, *buf, *buflen);
if (n < 0){
printf("dn_expand no space\n");
goto enospc;
}
newlen = prune_origin(*buf, origin);
if ((origin == NULL || origin[0] == '\0' || (*buf)[newlen] == '\0') &&
(newlen == 0 || (*buf)[newlen - 1] != '.')) {
if (newlen + 2 > *buflen){
printf("dn_expand no space 1\n");
goto enospc;
}
(*buf)[newlen++] = '.';
(*buf)[newlen] = '\0';
}
if (newlen == 0) {
if (newlen + 2 > *buflen){
printf("dn_expand no space 2\n");
goto enospc;
}
(*buf)[newlen++] = '@';
(*buf)[newlen] = '\0';
}
*pp += n;
addlen(newlen, buf, buflen);
**buf = '\0';
return (newlen);
 enospc:
errno = ENOSPC;
*buf = save_buf;
*buflen = save_buflen;
return (-1);
}

static void
```

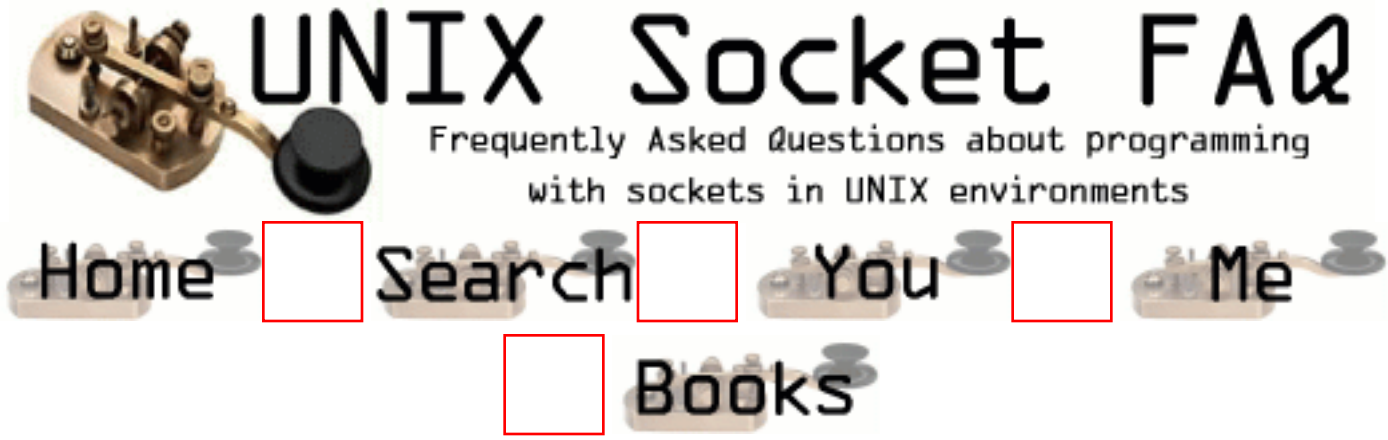
```
addlen(size_t len, char **buf, size_t *buflen) {
*buf += len;
*buflen -= len;
}


static int
addstr(const char *src, size_t len, char **buf, size_t *buflen) {
if (len >= *buflen) {
errno = ENOSPC;
return (-1);
}
memcpy(*buf, src, len);
addlen(len, buf, buflen);
**buf = '\0';
return (0);
}

static int
addtab(size_t len, size_t target, int spaced, char **buf, size_t *buflen) {
size_t save_buflen = *buflen;
char *save_buf = *buf;
int t;

if (spaced || len >= target - 1) {
addstr(" ", 2, buf, buflen);
spaced = 1;
} else {
for (t = (target - len - 1) / 8; t >= 0; t--)
if (addstr("\t", 1, buf, buflen) < 0) {
*buflen = save_buflen;
*buf = save_buf;
return (-1);
}
spaced = 0;
}
return (spaced);
}

int min(int a,int b)
{
if(a<=b)return a;
return b;
}
```

```
int
ns_sprintrrf1(const u_char *msg, size_t msglen,
const char *name, ns_class class, ns_type type,
u_long ttl, const u_char *rdata, size_t rdlen,
const char *name_ctx, const char *origin,
char *buf, size_t buflen)
{
const char *obuf = buf;
const u_char *edata = rdata + rdlen;
int spaced = 0;

const char *comment;
char tmp[100];
int len, x;

/*
* Owner.
*/
if (name_ctx != NULL && ns_samename(name_ctx, name) == 1) {
addstr("\t\t\t", 3, &buf, &buflen);
} else {
len = prune_origin(name, origin);
if (len == 0) {
// addstr("@\t\t\t", 4, &buf, &buflen);
} else {
// addstr(name, len, &buf, &buflen);
/* Origin not used and no trailing dot? */
if ((!origin || !origin[0] || name[len] == '\0') &&
name[len - 1] != '.') {
// addstr(".", 1, &buf, &buflen);
len++;
}
// spaced = addtab(len, 24, spaced, &buf, &buflen);
}
}

/*
* TTL, Class, Type.
*/
// x = ns_format_ttl(ttl, buf, buflen);
// addlen(x, &buf, &buflen);
// len = sprintf(tmp, " %s %s", p_class(class), p_type(type));
// addstr(tmp, len, &buf, &buflen);
// spaced = addtab(x + len, 16, spaced, &buf, &buflen);
```

```
/*
* RData.
*/
switch (type) {
case ns_t_mx:
case ns_t_afsdb:
case ns_t_rt: {
u_int t;

if (rdlen < NS_INT16SZ)
goto formerr;

t = ns_get16(rdata);
rdata += NS_INT16SZ;
len = sprintf(tmp, "%u ", t);
// addstr(tmp, len, &buf, &buflen);

addname(msg, msglen, &rdata, origin, &buf, &buflen);
break;
}


default:
comment = "unknown RR type";
goto hexify;
}
return (buf - obuf);
 formerr:
comment = "RR format error";
 hexify: {
int n, m;
char *p;

len = sprintf(tmp, "\\#(\t\t; %s", comment);
addstr(tmp, len, &buf, &buflen);

while (rdata < edata) {
p = tmp;
p += sprintf(p, "\n\t");
spaced = 0;
n = min(16, edata - rdata);
for (m = 0; m < n; m++)
p += sprintf(p, "%02x ", rdata[m]);
addstr(tmp, p - tmp, &buf, &buflen);
if (n < 16) {
```

```
addstr(")", 1, &buf, &buflen);
addtab(p - tmp + 1, 48, spaced, &buf, &buflen);
}
p = tmp;
p += sprintf(p, "; ");
for (m = 0; m < n; m++)
*p++ = (isascii(rdata[m]) && isprint(rdata[m]))
? rdata[m]
: '.';
addstr(tmp, p - tmp, &buf, &buflen);
rdata += n;
}
return (buf - obuf);
    }
}
```

Add a Comment



New Questions: Using select() for web-server program

From: Alvin Chang

I've got a question of how to write a code which uses select instead of just fork each child process to deal with different request. i.e, instead of

```
if (!fork())
{
   recv(request);
   send(response);
}
```
but to use select() to go through a list of connect_id and fds...

Thank u!

From: Magnus Boden

This is in pseudo code.

```
fd_set readfds;

while(1){
FD_ZERO(&readfds);
FD_SET(socket1,&readfds); // and one of these for every socket
select(maxfds,&readfds,NULL,NULL,0);
if(FD_ISSET(socket1,&readfds)){ // you should make some
readrequest(); // kind of loop to check
writerequest(); // all sockets
}
}
```

If you intend to make a proper app then you might want
to keep a state of the sockets so you know when to readrequest()
and when to writerequest(); and use a writefds to for all sockets
so you know when you can write to them, otherwise just because
you might be able to read from them it doesn't necessarily
mean you can write to them.

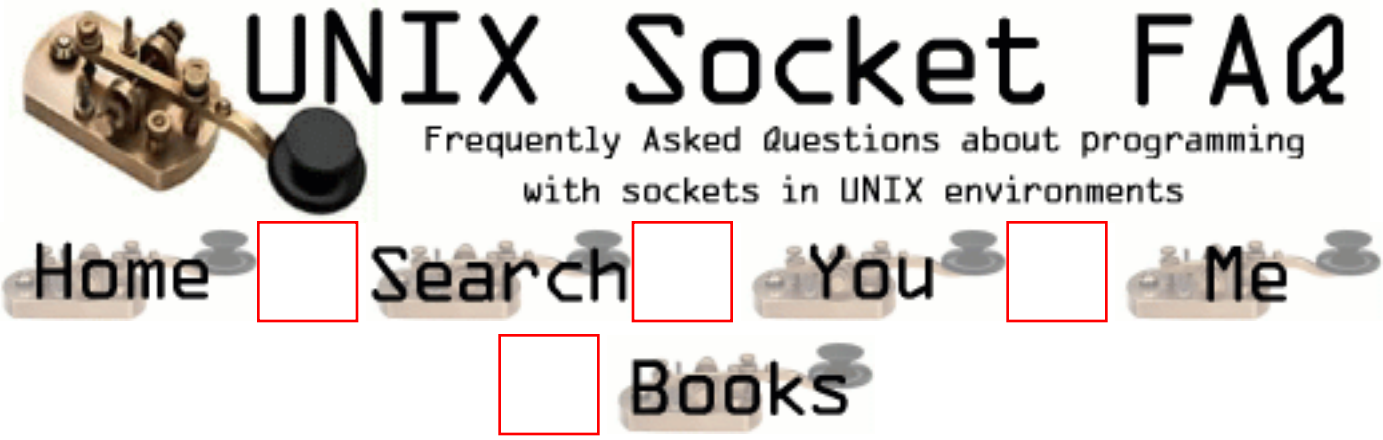
Although I think you will get by so you can se it work without
the writefds but when it is used a lot it will probably fail
sometimes if you don't.

P.S I am not a veteran socket programmer so I might have missed
a lot of stuff D.S

From: gangoor@yahoo.com

Good try

Add a Comment



New Questions: Ports that Hang

From: Casey Milford

Hello,
I have a situation that I hope someone can help me with. We have several UNIX (Digital) boxes that run our online applications. We also have an SDK that uses said applications. If one of the applications should fail, the ports open by the SDK users will hang. The client software does not recognise the application is down, so it holds the ports for approximately 15 minutes. Of course, the server side application cannot restart if the ports are being used. So, aside from writing the client to free its connection if the application is down, how can I _kill_ the ports so that our downtime is minimal? I am sorry for the long story, but I felt it necessary to tell the whole story.

Thank you for any help you can offer.

From: Magnus Boden

Is the sockets in TIME_WAIT state?
You can check this with netstat.

If so. Read the question about TIME_WAIT state in this faq.

From: Casey Milford

No. The socket will move to a TIME_WAIT until the program quits, then it moves to a FIN_WAIT2. Then the FIN_WAIT2 hangs until the client releases the port.

Thank you again for your help.

From: freeze

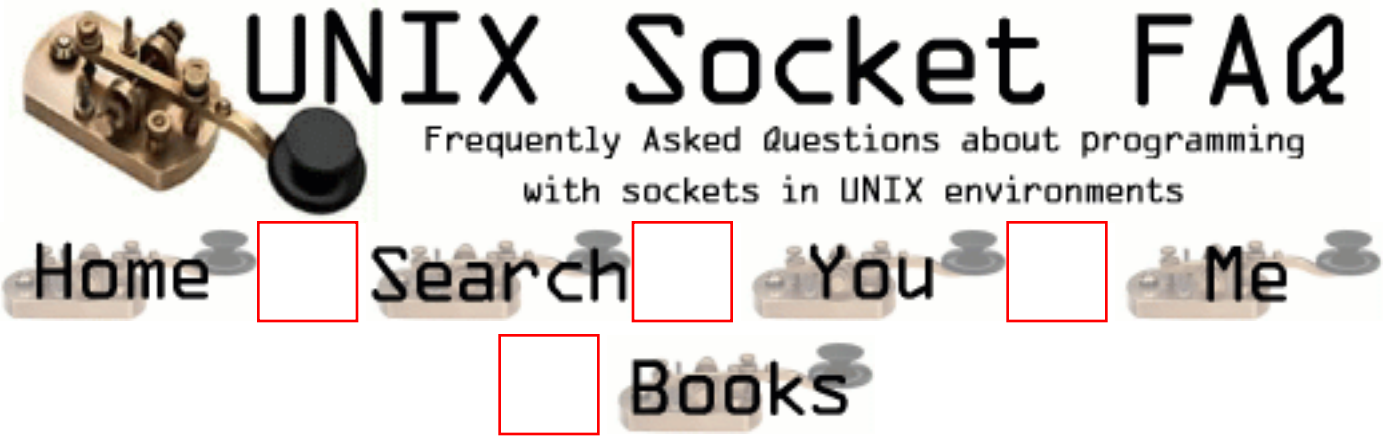
The FIN_WAIT2 state means that the connection is closed, and the socket is waiting for a shutdown from the remote end

later,
freeze

From: Baskar K

Try to use shutdown socket call instead of close.This should solve your problem.

Add a Comment



New Questions: read-write problem

From: Ben

I am writing a pretty elementary socket program. Without getting into the details, it writes an HTTP request to a socket, then reads back the response and parses it in a certain way. I have two routines, both write to the socket then read back the response (they parse it differently). If I call them individually, they both work exactly as they should. However, if I call them consecutively, the 1st one called works, and the 2nd one will not write to the socket. I am getting a -1 returned from my call to write(). So, basically, I can write, then read, but I can't then do another write. Anyone have a clue?

Thanks,
Ben

From: Rob Seace

What is "errno" set to when the second write() fails? Since you mention it's HTTP requests you're dealing with, my first reaction is that it's probably just the HTTP server that you are talking to closing down the socket on you, because I think HTTP is a one-shot protocol, by default... Ie: you connect, issue a single command, and get back a single response... There is a way to get a persistent connection, but I'm not sure of the details... Go check the appropriate RFCs...

From: Magnus Boden

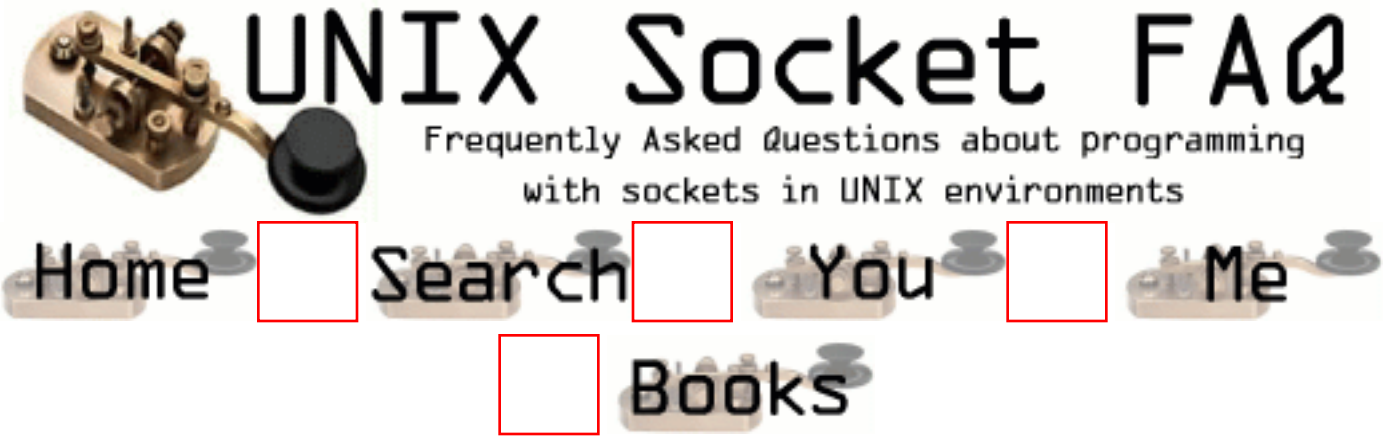
If you make an http1.1 request the connection is kept alive for more requests.

Here is how you make an http1.1 request. I have done it with telnet lots of times.

telnet www.someweb.com 80
GET / HTTP/1.1
Host: www.someweb.com:80

You have to press return twice at the last line.

Add a Comment



New Questions: I am getting connection refused when ftp from UNIX to NT based system

From: Somin

Dear friends,
I have a utility which ftp a file from UNIX system to the NT server .
but when I use the UNIX utility (HP_UX) , it throws back an error of connection refused.
I thought may be the Nt system requires IIS .
similarly when I create a TCP connection , It throws error no 127 : Connection Refused .
Please help me out .
Regards and Thanks ,
Somin

From: Arthur Lung

Trying to ftp from unix to nt? Make sure there's an ftp server running on the nt box, that it works, that it's using the correct ftp port, that it's actually running, that you have a route to get to it from your unix box.

Trying to ftp from nt to unix? Make sure the ftp server is running, works (connect to it from some other place), that network routes are all groovy.

If this is a case of it seeming to connect but take a really long time before it gives you a login prompt (probably to not even get that far), you may want to attempt to give the

unix machine the ability to resolve the window's machines IP address (or make sure that works). We've noticed that if it can't do that, it gets a little pissy sometimes.

From: ravi kumar

Whenever u want to ftp from one system to another system then u should have ftp server daemon running on the other end and the ftp client daemon running on ur end. Then only u can ftp to that system. Normally on NT systems there is no ftp server daemon running. It only has ftp client daemon. So u can't ftp to any of the NT systems. The Network Neighbourhood utility is used to connect from a windows system to another windows system. But u can't connect from UNIX to NT. But U can ftp from NT to UNIX, because it has ftp client daemon running.

From: Loco

There is no ftp client daemon... It's just an ftp client application. You just run it and can connect to any ftp server your machine can access.

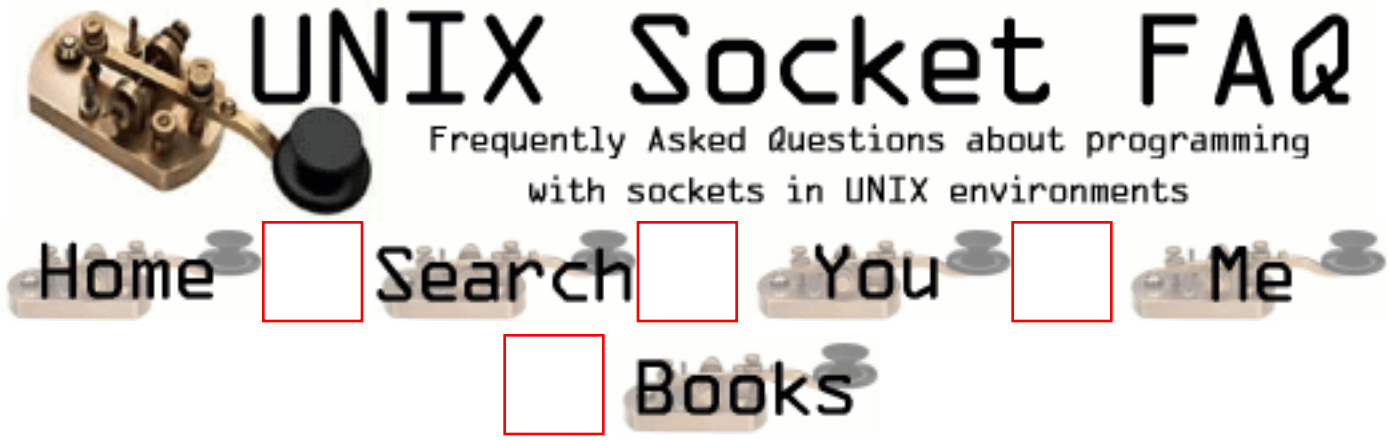
You can configure the FTP server on NT using IIS. YOu don't need to configure the HTTP server under IIS if you don't want to, they are separate services of IIS.

You can also download a lightweight ftp server for windows from any of the download sites in the web.

Unix normally comes with an FTP server in its distribution. It doesn't mean it is running, so you have to check it.
A simple "ps -ef | grep ftp" might be enough to find out if a server is running, however the best way to check it is to connect to the 21 port using telnet and see the response:
"telnet localhost 21" (use whatever address your host or any remote host has)
a response like: "telnet: Unable to connect to remote host: Connection refused" means you cannot connect to that specific port on the host which could be caused by many things, one of them is that the service is not running

You can share files from windows systems with unix systems, look for information on NFS and SAMBA.

Add a Comment



New Questions: can select() notify if a tcp/ip fd is readable?

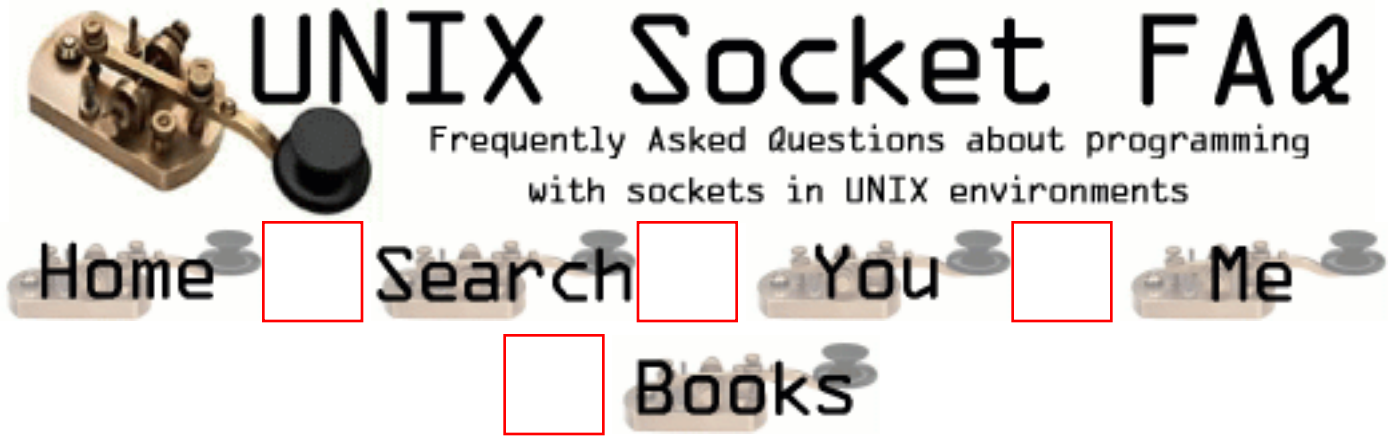
From: sreekanth

is it possible to wait for a certern amount of time for reading in a tcp/ip socket? i am not sure if select()is the function to use for this.

From: Rob Seace

Yes, select() should handle that just fine...

Add a Comment



New Questions: write files to NT

From: Jonas

I have a problem, the code is written in a oracle db. And now we problem with files in the db cause we have started to use a Unix server, earlier we have used a NT-SERVER.
Files that I earlier could be placed by giving path in the application ex \\computername\test\
Have stopt working. The files can be stored on the UNIX server.
Is it possible to write the files from the unix server to your localy NT-computer ?
(the files are about 100kb)

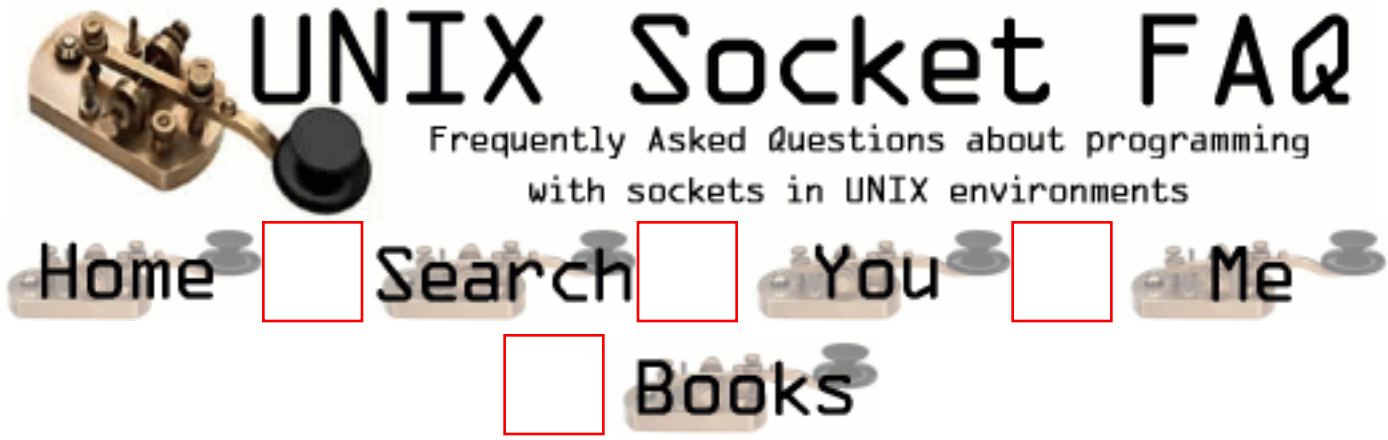
/Jonas

From: Dema_Lamer

=))) what problem ?
write simple program (for example ... using perl)
and copy all data (before copy struct =))

(DBI, DBDOracle)

Add a Comment

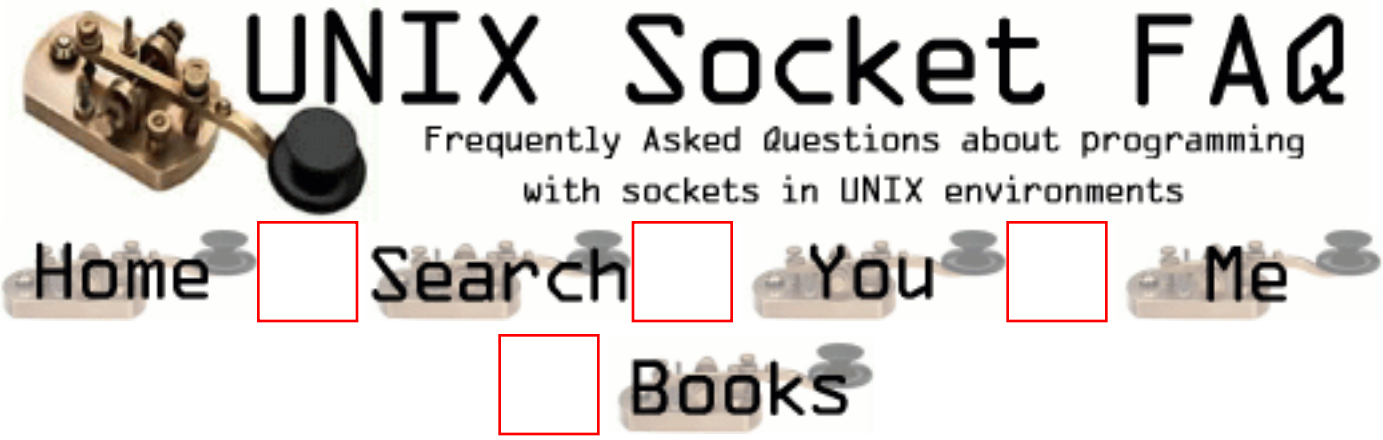


New Questions: TIME_WAIT and EPIPE

From: kevin

I have an app (IPCReader) on a Unixware system that opens a socket on an NT Server and write msgs it gets from an IPC queue. The IPCReader is forked from a parent who's job it is to read msgs from the socket and send it back to the IPC owner. Works fine, except under load testing. Each socket is only open for a couple millisecs, it finishes and closes its socket. During load testing this process to repeated every 5 secs while each of 48 IPC owners send data thru the socket. After several hundred successes, the sockets in TIME_WAIT are pilling up and then each new IPC msg is able to create/connect to a new socket, but the first send results in an EPIPE! Is this normal? I bumped my open fds up to 150, but 48 is the highest number of socks I'll ever have....

Add a Comment



New Questions: IPC using -shared memmory and semaphore Vs Sockets

From: Sridhar S

Could anyone please elaborate as to which is better in terms
of perfomance, memmory etc for an IPC implementation
vis a vis
i) using shared memmory and semaphores
ii) sockets

From: Rob Seace

Well, it would depend greatly on your OS of choice, and its implementation of each, of course... But, I think it's probably going to be fairly universal that shared memory is likely to give you better performance than sockets... But, at the same time, there's a performance vs. ease of use and functionality trade-off, too... In general, coding with sockets is going to be a lot simpler and less prone to coding errors and race conditions than a shared memory approach will be... And, if in the future you decide to extend whatever communication you're doing to talk between remote hosts, you'll already have the basic framework in place to do it, if you go with a sockets approach; it will just be a matter of changing from using AF_UNIX sockets to AF_INET ones, basically... And, on most systems, I think you'll find the speed of AF_UNIX sockets are going to be plenty fast enough for most uses; I've always been happy with them, anyway... Unless you really need pure, absolute maximum speed possible, for whatever reason... *shrug*

Of course, I love sockets programming too, so take my advice

with a grain of salt... ;-) If there are any shared memory
fans, I'm sure they will extol upon its virtues, too...
(Though, given that this is the Unix Sockets FAQ, I suspect
you're likely to find more sockets fans like myself, rather
than any major shared memory proponents... ;-))

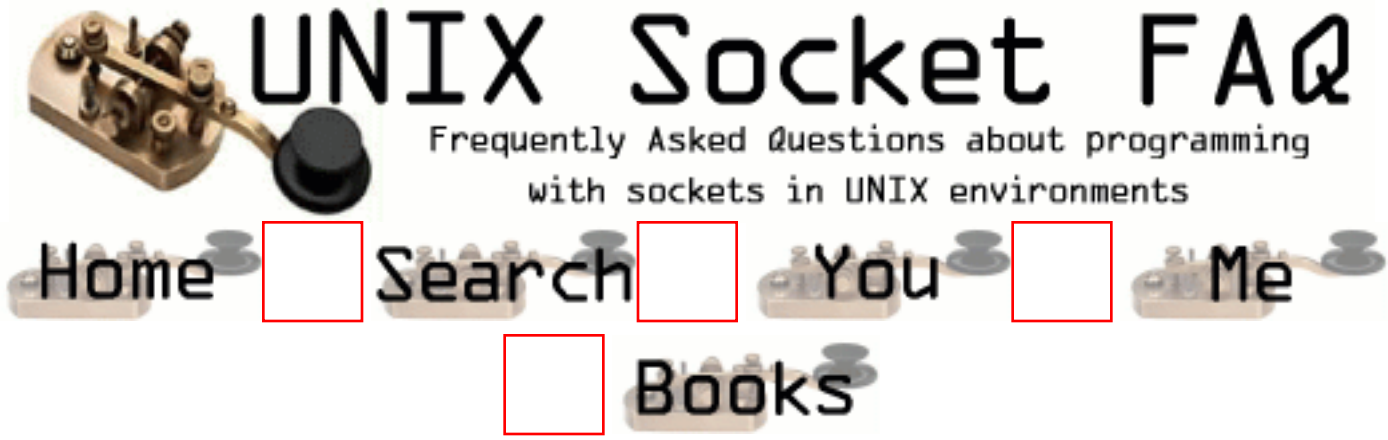
From: M Dhanraj Ganapathrao

please add some more detail information about sockets.

From: www

ghfh

Add a Comment



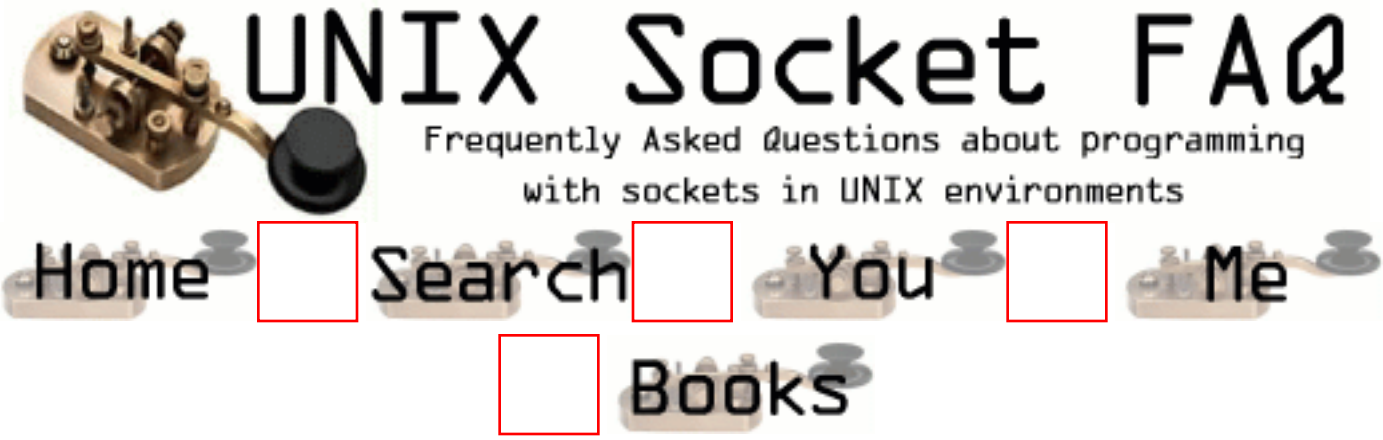
New Questions: Memory corrupted by child process

From: KBS

I have a global variable initiated by the parent process. This variable is suppose to share by all the child processes after forked.
In my program, I forked out 2 child processes, one to do some calculation work (let's call it "pCal") and the other to send and receive message from another server("pHost"). Everything run very smoothly until one day the other server went down, somehow it affect my pCal process and the variable data being currupted. One thing to highlight is the variable is never being used in the pHost process and the variable data in the parent process is perfectly ok even after this.

Can someone help?

Add a Comment



New Questions: Number of sockets and OPEN_MAX limit

From: Kevin F. Webb

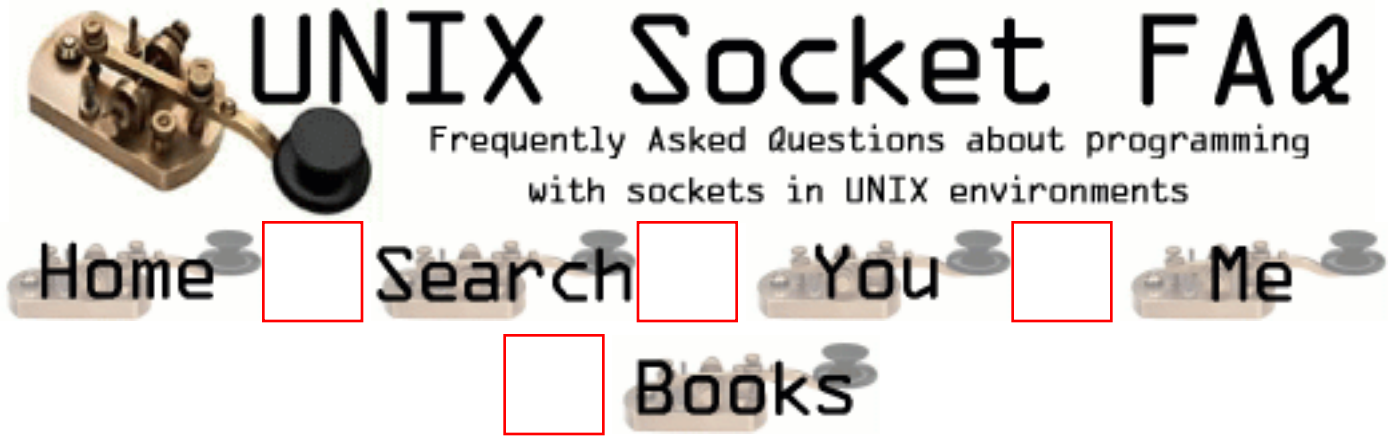
Using setrlimit() I have increased the max allowable open file descriptors to 1024 for a process, verified at runtime using /usr/bin/pfiles <pid>. However, via some other mechanism the system is limiting the number of sockets I can create in my process. Additional testing has yielded a limit of 256 sockets per process (sockets only), but if I create any FIFO's before I create the sockets then I can't even create 256 sockets.
Any ideas?

Running Solaris 2.8.

From: Sudhir Nelvagal

I think there are some hard limits that are defined. These are called Kernel Parameters. I am not sure the exact way you go about changing kernel parameters on Solaris. But on HP-UX you can use SAM (system administration tool) to change these parameters which will let you bump up max number of file descriptors, number of processes and so on..
basically the whole gamut of parameters that the kernel uses.
After this you will be required to re-compile the kernel (which may be done transparently by the system admin tool).
This way you can get over the hard limits you are seeing currently.

Add a Comment



New Questions: SIGPIPE in multi-threaded client/server program

From: Matthias Blankenhaus

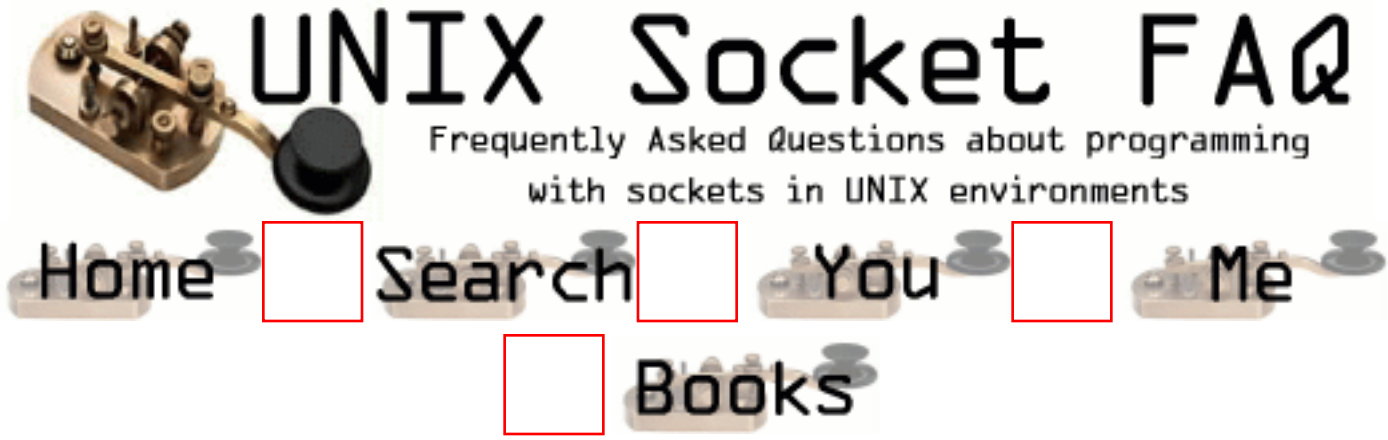
Ok, here is the problem: I am using blocking socket I/O between a client and a server which are different Posix threads within the same process. My server-thread binds and then blocks in its accept stmt. My client thread comes up and tries to connect to the server. I am doing the conntect() call within a loop, therefore calling connect() as often as it takes to establish successfuly a connection.
So after connect() indicated a successful execution I try to send data to the server (The server blocks in a read after accepting the client connection request.). The client's send() stmt triggers a SIGPIPE.
Does anyone has a clue whats going on ? BTW, I am using Solaris 2.6.

Thanx in advance,
Matthias

From: Matthias Blankenhaus

I got the solution. Looking closely at the man page of connect()
shows that in case of a ECONNECTREFUSED you should close the socket,
create a new socket and then try to establish the connection again.
I should have rtfm. Basically after a failed connect() the socket
is in a not defined state. If I add that to my code, then it just
works fine !

Add a Comment



New Questions: How to use select() ?

From: Antonio Vilei

Hi,
a multicast ftp server has to send a file (divided into blocks) to a multicast IP address through a DATAGRAM socket and it has to receive (just a few) NACK packets from the receivers (the say which data blocks were lost). The server should go on sending data blocks without stopping while receiving nacks: i.e. no stop and wait.

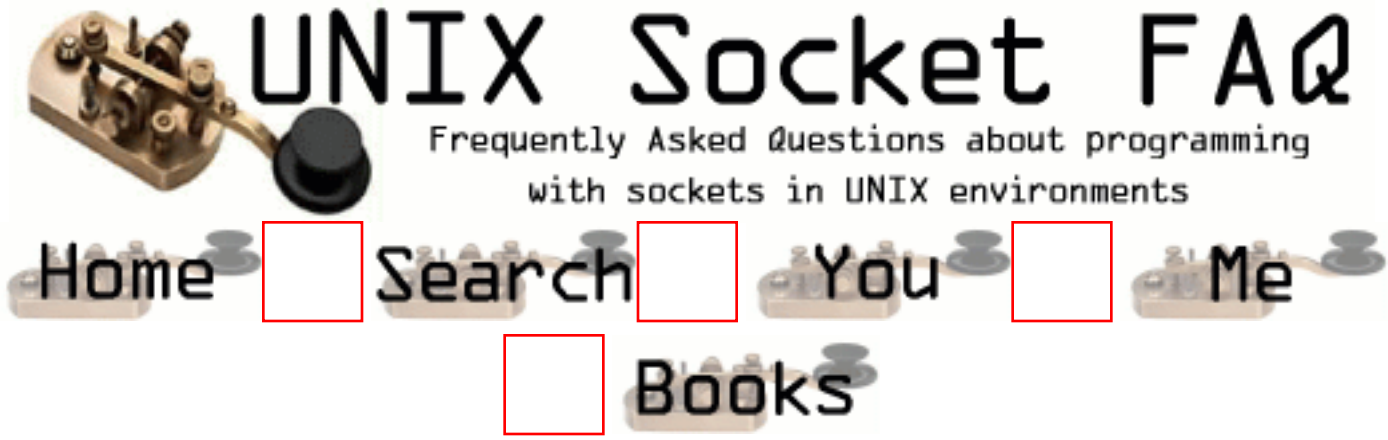
The question is:
can I use select() using as read set a SINGLE socket which reads the nacks, and as write set the socket which sends datablock? Do I risk to lose any nacks from the receivers?

Thanks in advance.

Antonio Vilei

From:

Add a Comment



New Questions: listen fails on OpenServer 5.0.5

From: Bob Hepple

I have some code which runs on Linux/Solaris to create a listening socket - it's really vanilla socket/bind/listen stuff... but I can't get it working on SCO OpenServer 5.0.5

I get this error message from the sample code below:

StartSocket:pid 685: Socket Listen Failed errno=122 (Operation not supported on transport endpoint)

... which is a weird errno (not in errno.h) ...

any ideas what I am doing wrong???? - I have tried running as root and using different ports, all to no avail. I've tried putting 12396 into /etc/services - do I need to register that port anywhere?????

Cheers

Bob

```
// port = 12396

int
StartSocket(short port, SOCKET * pListener)
{
```

```
	int err;
	SOCKADDR_IN localAddr;
	SOCKET listener;

	if ((listener = socket(AF_INET, SOCK_STREAM, 0)) < 0)
		return -1;
	/*
	 * Bind our server to the agreed upon port number. See
	 * commdef.h for the actual port number (default 12396)
	 */
	memset(&localAddr, 0, sizeof (localAddr));
	localAddr.sin_port = htons (port);
	localAddr.sin_family = AF_INET;
	localAddr.sin_addr.s_addr = INADDR_ANY;

	err = bind(listener, (PSOCKADDR) &localAddr, sizeof(localAddr));
	if (err < 0) {
		debugErr(logerr (FN"pid %d: Socket Bind Failed\n", _getpid());)
		return -1;
	}

	/*
	// Prepare to accept client connections. Allow some pending
	// connections.
	*/
	err = listen(listener, 2); // .... returns -1 !!!!!!!
	if (err != 0) {
		logerr (FN"pid %d: Socket Listen Failed errno=%d (%s)\n", _getpid(),
errno, sys_errlist[errno]);
		return -1;
	}

	*pListener = listener;

	return 0;
}
```

From: Rob Seace

Odd... I don't see anything obviously wrong with your code, at all... And, if it works on other systems (as you mention it does), I'd have to say that it looks like something is wrong with SCO, not with you... ;-) My only guess would be to try playing with the backlog count passed to listen(); maybe it doesn't like only 2 being specified? But, the

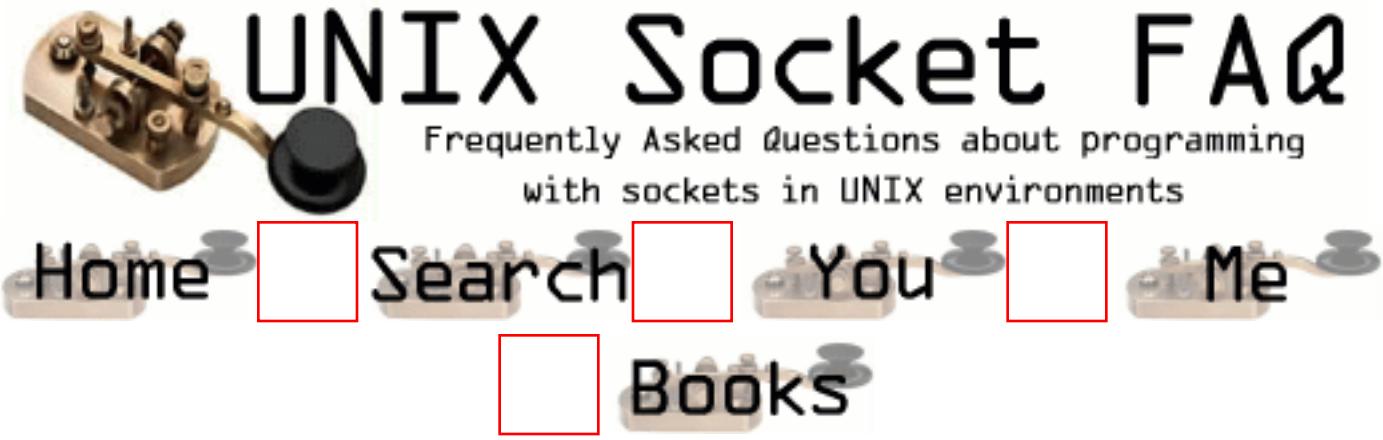
quoted error message makes it sound like it doesn't even
think the socket is a valid SOCK_STREAM socket, on which
listen() is allowed... So, I'm not sure if that would
really help at all... I assume you've dug through the
local SCO man pages already, and they don't mention any odd
requirements for doing a listen() that differs from the rest
of the Unix world? Or, they should at least mention what
condition should cause that errno to be generated... *shrug*
I just don't know SCO at all, so I can't really be of any
help... But, I'd say it seems almost certain that the
problem lies in something OS-specific, and not with your
code in general...

From: Bob Hepple

Here's the solution - don't put -I/usr/include in the compile string.
Sheesh! I was using the compatibility compiler /udk/usr/ccs/cc

---

Add a Comment

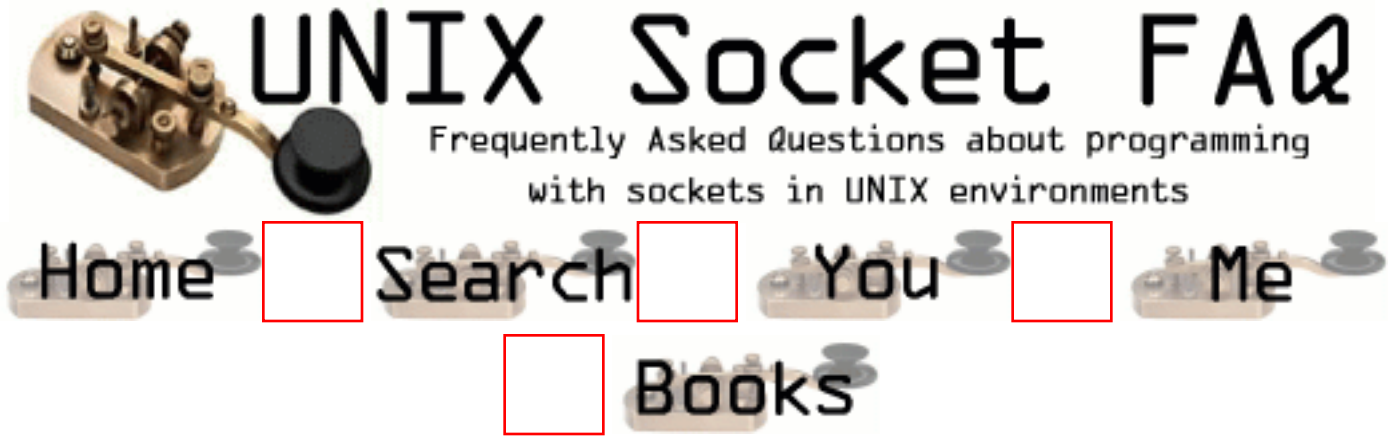


New Questions: Authorizations doubts

From: Daniele

How can I make my server not to accept requests from clients that are not in an authorizated machines list?

Add a Comment



New Questions: About sharing the same socket/same connection to server for two programs

From: angus

After finish running c program, the socket will be automatically closed. How can I prevent this situation occurs?
Also, if the first program create the connection. how can the second program can use same connection that created by first program

---

Add a Comment

New Questions: c++ and sockets (problem using close)

From: James

I am trying to compile a cpp file on a solaris machine
using g++. I get a compiler error wherever I try
to close a socket. eg: close(sd); What am I doing wrong.
This is the error.

implicit declaration of function `int close(...)'

Thanks
James Skinner

From: Rob Seace

You're not including the header file that close() is defined
in... (I would guess it's probably in <unistd.h>... But,
check your local man pages, or just grep through the headers
for it, yourself...)

Add a Comment

New Questions: select/recv: how to read complete message?

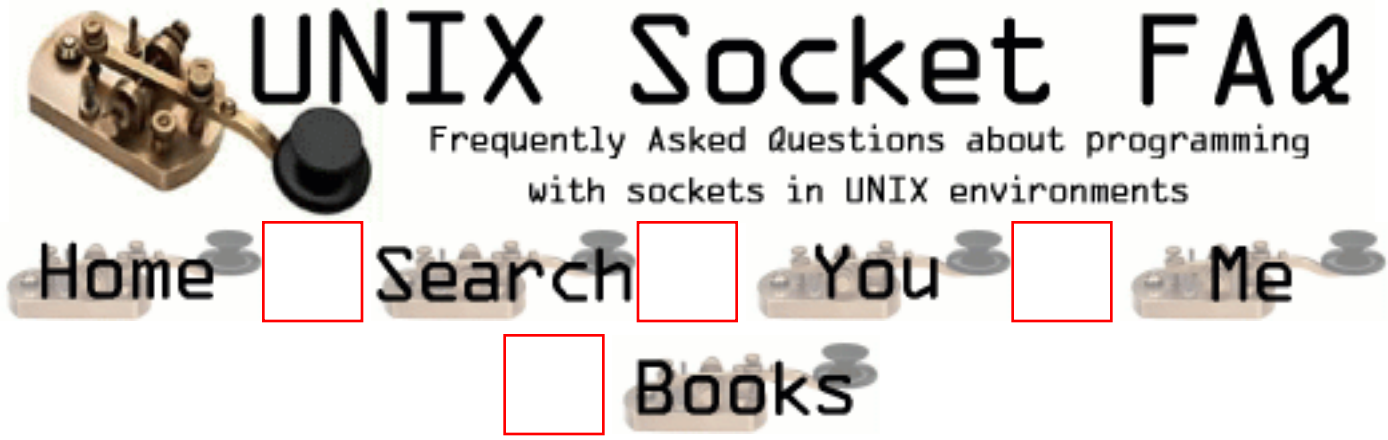
From: manu

my idea:
- on a select() return, test by recv(MSG_PEEK) or ioctl(FIONREAD) if the message is complete
- if complete, read it by recv(0)
- if not, let it in the buffer and go on select()ing
The problem:
select() continues to tell there is something to read in the buffer even if there is no new byte in it
Any idea?

From: Arthur Lung

Keep reading data and adding it to a buffer. After every read, check to see if you've got the whole message or not. You could do that by knowing how big a message is, examining the data you read, or some other method.

Add a Comment



New Questions: Blocked I/O Timeout?

From: Tom

Is there a way to force a return from a blocked write call?

Say the receiving program quits reading data with connection open. The write call eventually blocks. What mechanisms exist to return control out of this blocked write call and close the port.

From: Rob Seace

About the only way to do such a thing would be with a
timeout... (Or, use non-blocking IO mode on the socket,
and call select() as appropriate...) There are other
questions already in this FAQ for how to do timeouts, but
basically the standard method is to use alarm(), and set up
a SIGALRM signal handler... (You can either siglongjmp()
back into the code from the handler, or just return, and
let the read()/write() be interrupted and fail with EINTR,
if your system works that way... Some will restart a
read()/write() that gets interrupted by a signal, which
forces you to use the siglongjmp() method...) Or, if your
system supports socket-level timeout settings, via
setsockopt() with SO_SNDTIMEO, you could use that instead...
But, the alarm() approach is the most portable...

From: Mark Busslinger

I would not use alarm() to timeout a write.
I use select on a blocking socket!!
Try this:

```
static int Send (sock_typ *Socket,
                 const unsigned char* Buffer, int Size,
                 int TimeoutSeconds)
{
  int BytesSent;
  fd_set fd;
  struct timeval tv;

  FD_ZERO (&fd);
  FD_SET(Socket->m_Socket, &fd); // adds sock to the file descriptor set

  tv.tv_sec = TimeoutSeconds;
  tv.tv_usec = 0;

  assert (Socket);
  assert (Buffer);


  if (0 == Socket->m_Socket)
    return (0); // Allready closed

  // The return value is smaller than 'Size'
  // when the Client breakes the operation down.
  if (select (Socket->m_Socket + 1, NULL, &fd, NULL, &tv) == 0)
  {
    UpdateSocket (Socket);
    Socket->m_LastFunction = "Send Timeout";
    return (0);
  }

  BytesSent = send (Socket->m_Socket, (char *) Buffer, Size, 0);
  if (BytesSent == SOCKET_ERROR)
  {
    BytesSent = 0;
    UpdateSocket (Socket);
  }
  Socket->m_LastFunction = "Send";

  return (BytesSent);

} // End of Send
```

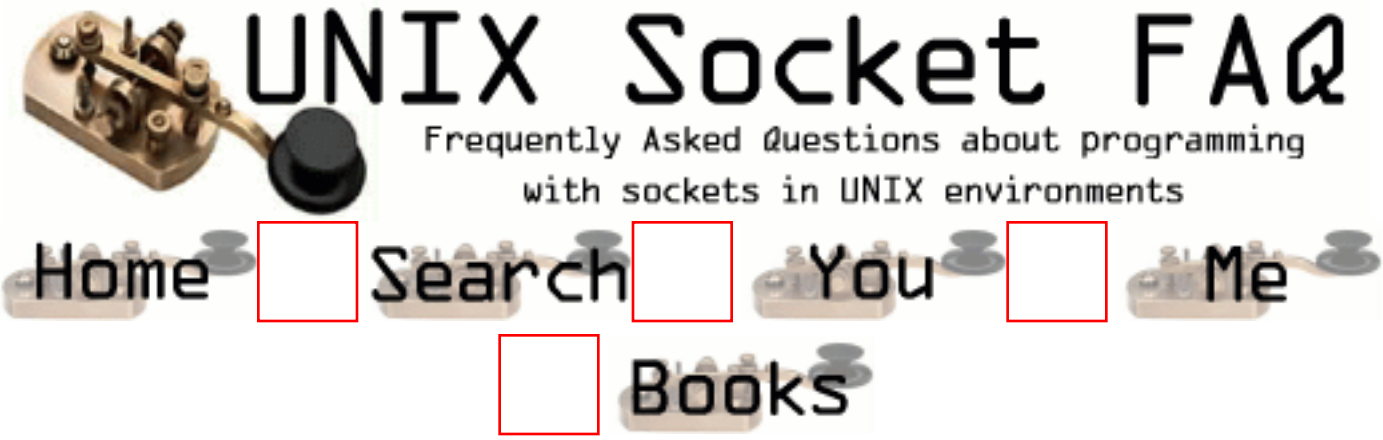
From: richard nathan

I am relatively new to socket programming...
I have however written a client and server routines that work fine, but I was asked to send an image file larger than
1MB and the receiver always received ~132KBytes maximum.
How can I set the socketoptions or use select (something) to increase the amount of data received.
I am working on a Sun-Solaris 2.6. using ANSI C.

From: sielim

You should just loop until the read returns 0. All data can be
read this way. Just like in file.

Add a Comment



New Questions: Securing sockets

From: Pete

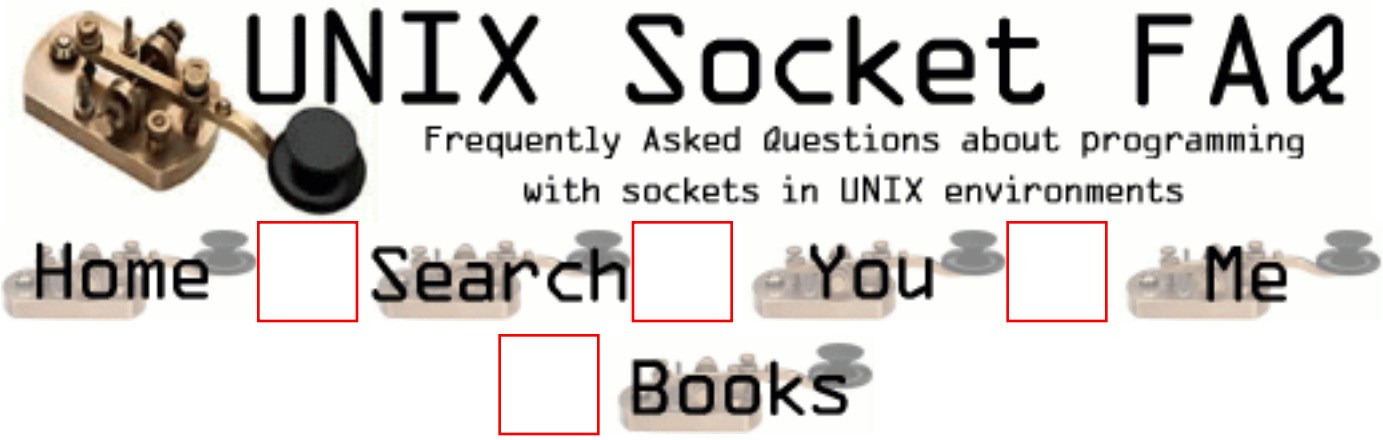
How do I secure socket communications using SSL?

From: Danny

I would have thought that your first point of call would be to look into
Eric Young's OpenSSL library. www.openssl.org.

hope this helps.

Add a Comment

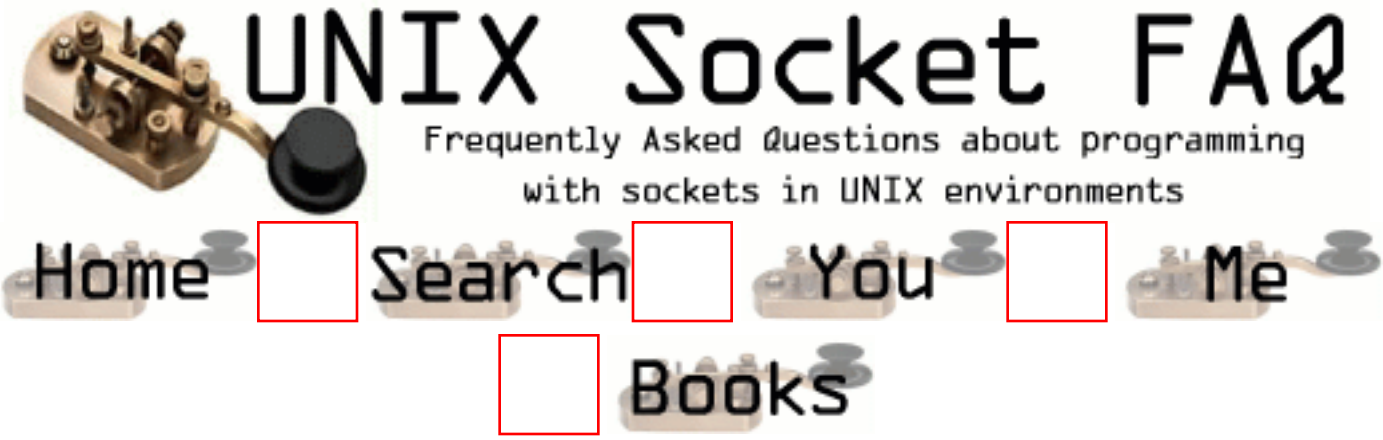


New Questions: how to get all listening ports ?

From: dummyNode

How to implement someting like netstat ?

Add a Comment



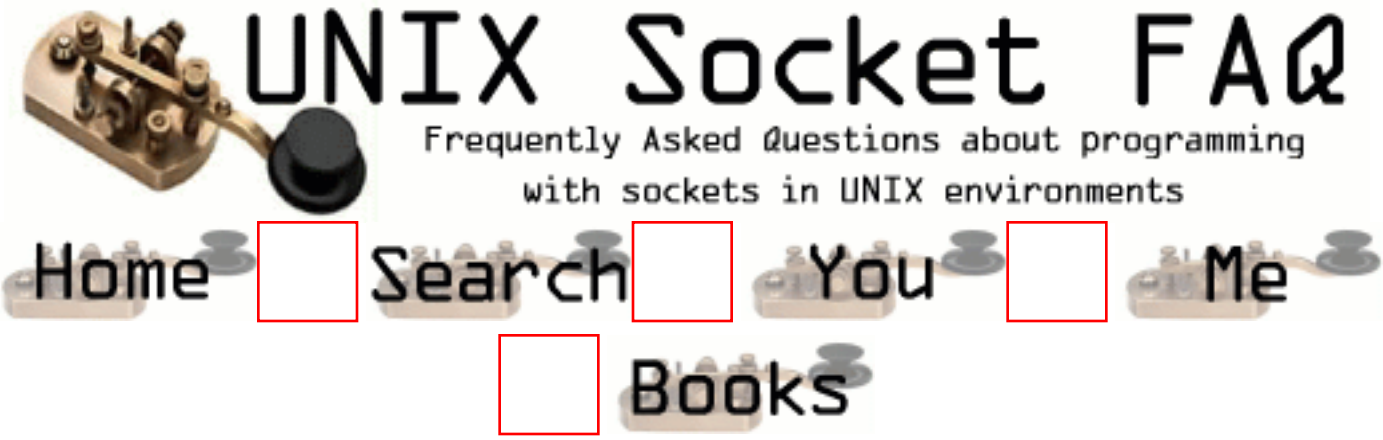
New Questions: If ISocket permission denied

From: vishwanath

Hi,

If I try to connect one IRIX machine to another or if I ping it it gives an error message of socket permission denied. Can anybody aanswer my query?

Add a Comment



New Questions: many http request via single socket.

From: hallian

hi all,

I was wondering:
Has anyone created own packets to send many HTTP request from a single socket,
so that it does not consume system resources as if I open one socket for each http
request. Also, each time a packet is sent, the source address should be different.

Any help towards this objective would be great!!!

thanks,
hallian

From: Gautam

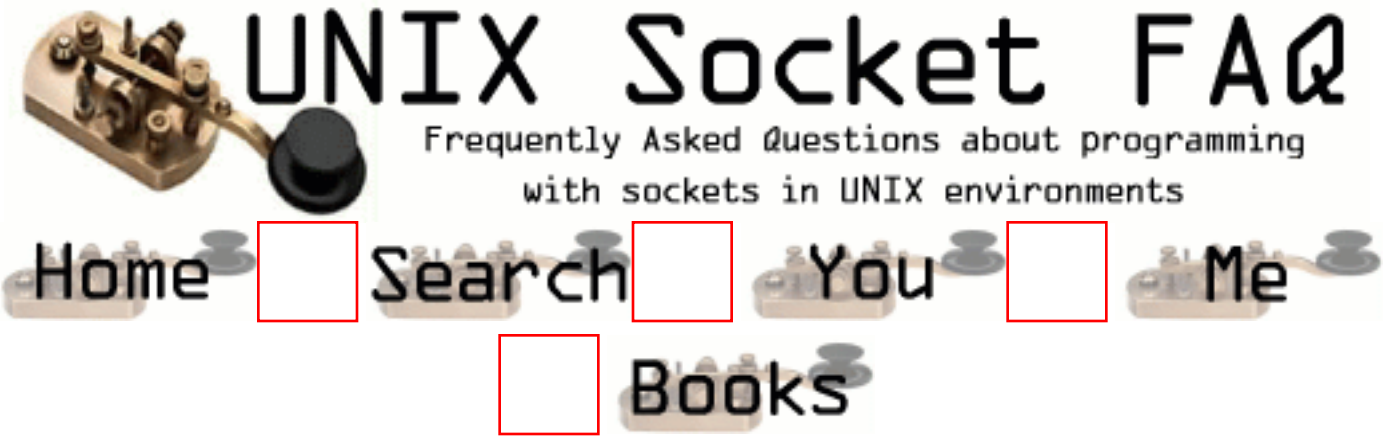
I doubt because http is stateless protocol. So u serve the request & the connection is lost. For next request another socket..& so on. There is a runway ..u catch the stream & don't leave it unless the client leaves (but a very crude way) also this is possible through clients that are written so as to request through same socket....standered browsers will not!!! so choice is to write ur own browser ;) or write a applet..

Cheers,
Gautam

From: Bill Klein

If I'm understanding your question correctly, the answer is yes. In HTTP/1.1 you can have persistent connections (and in fact it is the default) and can thus make multiple requests in the same session. See http://www.faqs.org/rfcs/rfc2616.html for more info.

Add a Comment



New Questions: HOW can I tell protocol?

From: Sridhar

I'm writing a port scanner program which scans all ports or a range of ports on a system. The program connects to each port and checks for any initial response and reports whether it can connect to that port or not. The problem is, How can I tell which protocol the port is implementing. If I use tcp to connect to all sockets, will I be able to read from UDP or ICMP sockets as well? Before anyone suggests me to get the protocol from /etc/services, let me ask: What if the port number is different? for example, in /etc/services in my computer, udp/13 is set for the daytime protocol. What if in some other host, the daytime protocol is udp/14? How can I know what protocol 14 is under their computer?

Please help!!!!
I would greatly appreciate it if You could contact me by e-mail at sridhar@coolmail.net

-Sridhar

From: Mark

If you connect with a wrong protocol, you will get an error.
Maybe you need a protocol scanner with try and error as well.

From: Rob Seace

There's really NO way you can tell what is running on a remote machine, on any given port#... The best you can do is hope they are sticking to the standard port# assignments, and base your assumptions on that... However, why create a port scanner? There are several good ones already in existence... Or, is this just a project to recreate the wheel, for a fun learning experience? Even so, you may wish to check out nmap... It's a superb free port-scanner, with full source code available...

From:

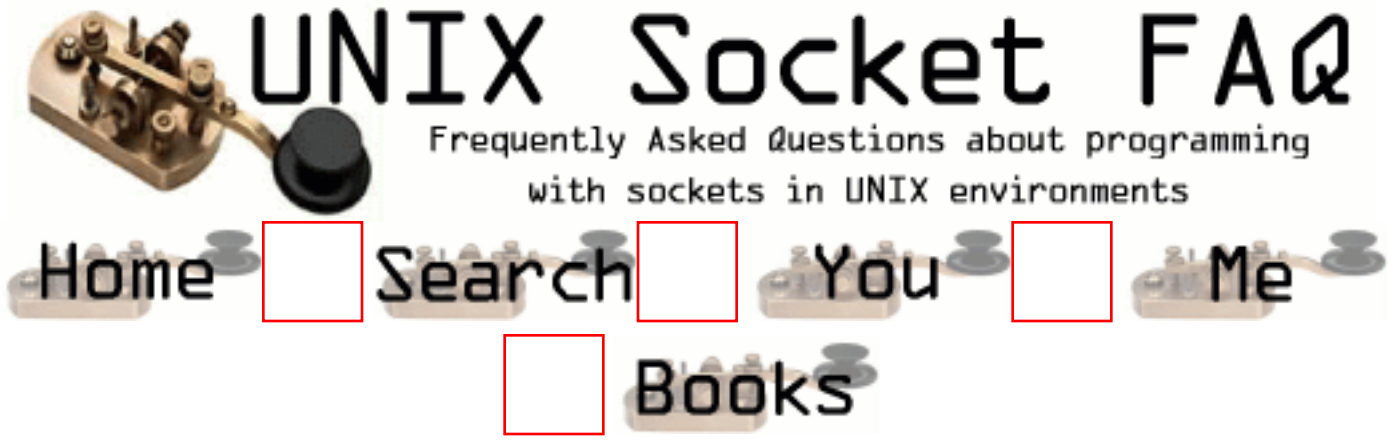
One universal way. Connect to each port a bajillion times.

Each time, try to talk to it using one of the bajillion different protocols. Hopefully, eventually it will seem like it's working, and you can hope that the protocol is what you think it is and not something similiar.

Most of the "popular" protocols, like telnet,ftp,www,and so on use a specific port always, for just the very reason that it would be impossible to determine what port to use otherwise.

So like the dude above this said, I'd stick with assuming anything listening on port 23 is either a telnet server, or one of those stupid programs designed to tell you when you're being scanned.

Add a Comment

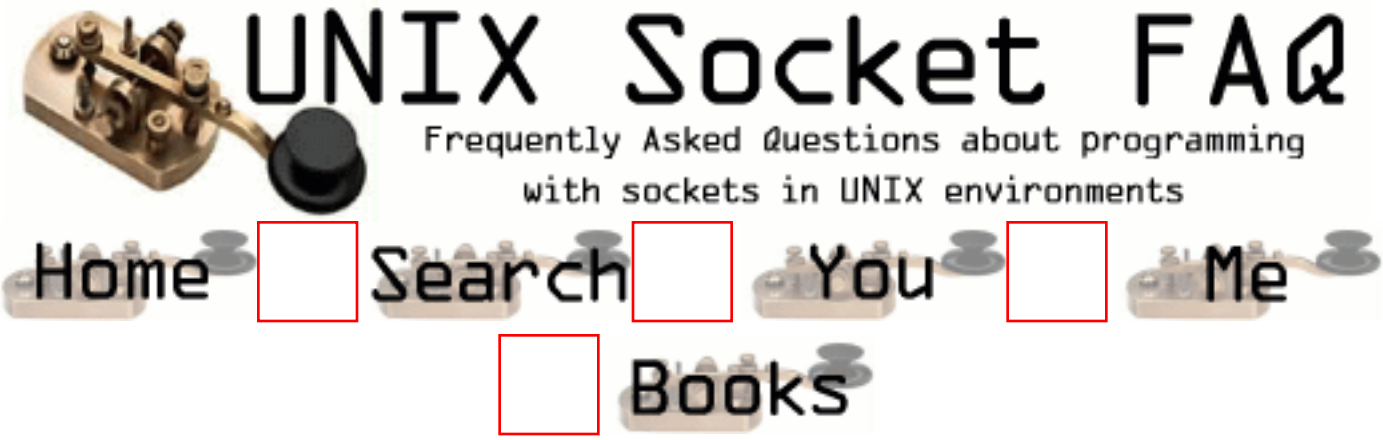


New Questions: Problems receiving UDP Broadcast...

From: Todd Gruben

I have a simple python application that basically forwards a UDP broadcast through a tcp socket accross the internet. This app worked great on NT. On my linux box, is see that the broadcast is occuring via the tcpdump command but for some reason my sockets app is not seeing those packets. Is this a sockets problem or a network config problem?

Add a Comment



New Questions: Changing the buffer size of a listening socket

From: Irina

Does somebody know if the modification of the size of the buffer of a listening socket will affect the size of all the connected sockets to that socket?

For example, if "accept" will generate several connected sockets for all the clients which succeed to connect to the listening socket, will every buffer of each connected socket have the new size (say 32K) of the listening socket? Or should I make "setsockopt" to every connected socket in order for this to be true?

Thanks.

From: Arthur Lung

The documentation says that it copies "most" of it's socket options to the newly created socket. I'm not sure what constitutes "most" though. I know the new socket will inheret stuff like wether it's a blocking socket or not. About the only way to be sure would be to accept a connection and then examine it to see if what you're looking for is already set.

It's not clear if the "most" options are the same across platforms.

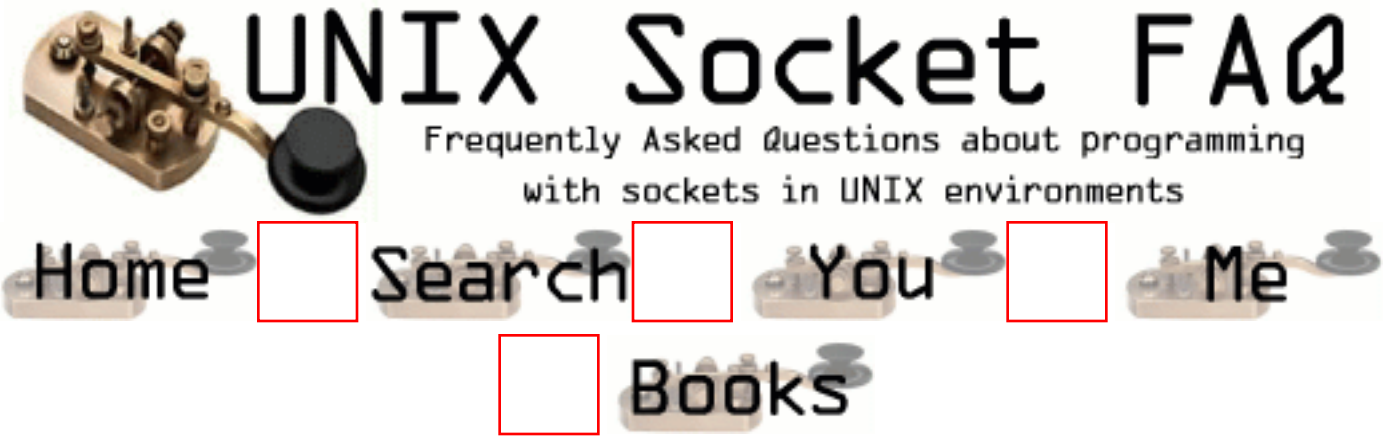
Either way, the gist is that when accept creates a new socket, that socket inherits settings. But if you later change the listening socket's options, they don't carry over to the

sockets that have already been accepted.

I beleive that when accept creates a new socket, it sets all of the existing options (like blocking or not, for example) to be the same as those options on itself. I'm not sure if the receive buffer is one of them

---

Add a Comment



New Questions: read count diff when accessing web server.

From: Terrance

Hi,
When I access a web site using the following loop, I usually
get different number of bytes. And when I print it out it
is very different from what I see when I use the View Source
in Netscape. How do the browsers always get the page
correctly ?

```
char c;
int i = 0;

while(read( sockid, &c, 1))
{
  cout << c;
  i++;
}


cout << "Word count : [" << i << "]" << endl;
```

Thanks !
Terrance

Add a Comment

New Questions: How to allocate the server buffer effectively?

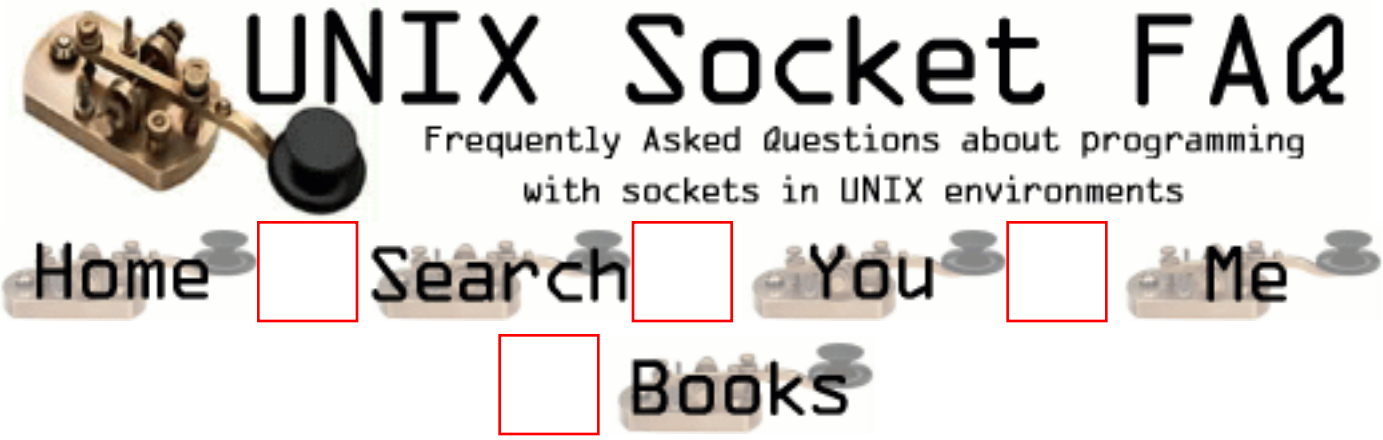
From: biju

Dear sir,

I have made one POP3 and SMTP server.
It is correctly woriking with mail clients.
It takes much time to get a large image from mail client. I have used CSocketFile and CArchive in MFC to aid in my programming. I am reading data from the CArchive variable character by character to an array. When this array limit reaches I am adding the content of this array to a CString variable, the size of which is entirely dynamic.

Could you please give me a way to solve the problem?

Add a Comment



New Questions: signal(SIGPIPE,SIG_IGN) still doesnt ignore SIGPIPE

From: Per Arneng

I do this call but the process still terminates on a SIGPIPE signal..
i have tried to do it with sigaction but it still doesnt work. I also
check the handler of SIGPIPE after i have done signal(SIGPIPE,SIG_IGN)
and the handler is not SIG_IGN.

best regards Per

From: Rob Seace

Odd, I've never seen signal() fail before... But, that's
what it sounds like is happening to you... Check the return
value from the signal() call, and verify that it's really
failing... And, if so, see what the "errno" is set to...
Maybe that will give you some clues as to the problem...

From: Per Arneng

signal does not return SIG_ERR. here is the code where i check:

```
 if(signal(SIGPIPE,SIG_IGN)==SIG_ERR) {
   //it doesnt come here.. but if i change SIGPIPE
   //to a random number it reports an error
   // so this step seems ok
   perror("signal");
 }

 struct sigaction oldSigpipeAction;
 sigaction(SIGPIPE,NULL,&oldSigpipeAction);
```

```
  if(oldSigpipeAction.sa_handler == SIG_IGN) {
    cout << "SIGPIPE = SIG_IGN" << endl;
  } else {
    cout << "SIGPIPE != SIG_IGN" << endl;
  }

//-------------------------------------
```

i compile and run it on debian 2.2
g++ -v = gcc version 2.95.2 20000220 (Debian GNU/Linux)

best regars Per

From: Rob Seace

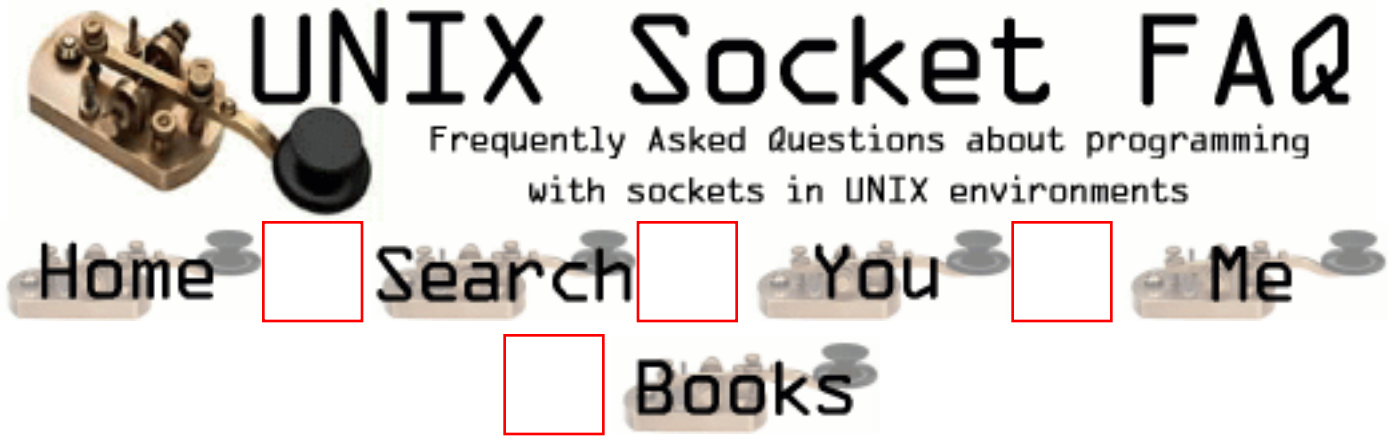
That's really odd... I just cut+pasted your little snippet of code (and, enclosed it in a main(), and added a few #include's), and compiled it on my Red Hat 6.2 box (g++ -v reports "gcc version egcs-2.91.66 19990314/Linux (egcs-1.1.2 release)"), and running it output the "SIGPIPE = SIG_IGN" success message... So, I don't know what's so different between the two systems that could be at fault... I wouldn't think it would just be the compiler version differences; this isn't really a compiler issue in any rational sense that I can think of... The code should function the same regardless of compiler or OS, for that matter... It just really seems like something totally odd is going on with your system, refusing to ignore signals... Does it behave this way for signals other than SIGPIPE? Eg: can you ignore SIGINT, SIGQUIT, etc.? What about if you set it to an actual signal-handler function, instead of SIG_IGN; does it similarly refuse to instate your handler function?

From: Per Arneng

Seems very odd. I have done handlers for othe signals like SIGINT and others,, and they work fine.. its just sigpipe. Well im getting a new harddrive tomorrow and im going to reinstall debian .. the i will compile again and see if i get the same error. Well this is really a strange error.. or mayby a bug. Could it be that some special things happen just with my code. i compile with -D_REENTRANT amongst other things.. (threaded webserver). If anyone want i can send the code..

best regards Per

Add a Comment



New Questions: Socket send guarantee delivery ?

From: Roy

Using SOCK_STREAM i'm using send and checking the return value. send only checks that the data has been successfully passed to the local transport provider though.
I need to be absolutely certain the data has been read by the server program (not running on same machine).
Whats the best way to do this. I realise i could implement my own acknowledgement but can't TCP/IP guarantee the successful delivery of my message ?

From: Rob Seace

TCP does "guarantee" delivery of data... If it is at all possible to get it there... Ie: if TCP can't get it there, there's not a lot YOU are going to be able to do about it, via any other method, instead...

Is your requirement that you just need to know for sure WHEN the remote side has successfully received the data you sent? If so, personally, I would make the design such that the remote end had to send some message back to confirm receipt, as you mentioned (but, didn't like)... But, if that were not doable, for some reason, the only choices I can see are: 1) set a small send buffer on your socket, preferably exactly the size of your outgoing message, and then monitor the socket with select() after your write(), and watch for it to become writable again, which means the buffer has room in it again; 2) use a raw socket to monitor all inbound TCP traffic, and watch for the incoming ACK for your outgoing message... #2 is probably overly complicated for any normal usage... (But, now that you mention it, it does seem a shame that the standard sockets API doesn't define a way to be notified of when your packets get ACK'd... I can see how that could certainly be handy...) And, I really can't guarantee the accuracy of #1... But, it sure SOUNDS like it should work, to me... The message shouldn't leave the send buffer until the remote side ACKs it, I wouldn't think... So, I don't know why it shouldn't work... But, this is all just theory; I've never had a need to try anything like that before...

You may want to check out the following sections of the FAQ here, as well, which may or may

not be helpful to your situation: 2.11, 2.16, 2.17...

From: Roy

Thanks Rob for your comments.
(As you gathered its a guarantee of state of delivery i'm interested in)

According to Stevens book "when TCP sends data it requires an acknowledgement in return. If an acknowledgement is not received TCP retransmitts the data, and after some retransmissions TCP will give up - time spent retrying depending on implementation (typically between 4 & 10 mins)".
Trouble with this is i can see no way of detecting that TCP has given up with a particular data segment. (Hence still no guarantee of delivery/detection)

Also in Stevens "only when ACK received from peer can TCP discard the data from the socket send buffer"
If i have interpreted this correctly then setting the buffer size to be the length of my message will mean that i can send no other data (it will block) while that packet is being re-sent. While this enables detection that the data hasn't recieved an ACK whilst in the buffer, it also means i'm blocked sending any other data for the given 4 to 10 mins!.

Also regarding performance of a smaller buffer under normal working cicumstances, if i do reduce the buffer size (to my actual message size) the buffer will be full until the ACK is received and so my sends would block. How long does this ACK response take ? (With testing it seems approx 200ms) Setting TCP_NODELAY does not seem to improve this. Would this be my throughput limit ?

From: Rob Seace

No, you don't need to block with another write(); you can
just call select() after the first write(), and when the
socket select()'s as writable again, presumably that should
mean the other side has ACK'd the first message... However,
this assumes you want/need to know the state of every
message immediately after it's sent... If, instead, you
just want to blast out a bunch of messages, and then go back
after, and find out for certain whether or not they've all
(or, any particular one) has made it, it may be harder...

And, of course yes, it is likely to reduce your throughput a
fair bit as well, since you won't be able to send nearly as
fast as you could otherwise... But, if you're concerned
about whether or not a particular message made it to the
remote side at a particular time, then presumably you would
not be interested in sending any further messages anyway,
would you? I'm just not sure exactly what it is you are

trying to accomplish, so I'm not really sure the best way to
go about it...

From: sielim

But there's some other question: How to set buffer size
to the minimal value ? At redhat 6.2:
```
  int x=0;
  setsockopt(sock_fd,SOL_SOCKET,SO_SNDBUF,&x,sizeof(x));
```
Unfortunatelly the buffer seems to be still of size 1KB.
It is not enough. How can I set it to lower value ?
Is there a way to flush the buffer ?
I tried also to use
```
    int err=ioctl(sd, SIOCGSTAMP, &timestamp);
```
to determine time of last data sent (exactly as in man pages)
but it returns error: No such file or directory.

Could you help me ?

From: sielim

But there's some other question: How to set buffer size
to the minimal value ? At redhat 6.2:
```
  int x=0;
  setsockopt(sock_fd,SOL_SOCKET,SO_SNDBUF,&x,sizeof(x));
```
Unfortunatelly the buffer seems to be still of size 1KB.
It is not enough. How can I set it to lower value ?
Is there a way to flush the buffer ?
I tried also to use
```
    int err=ioctl(sd, SIOCGSTAMP, &timestamp);
```
to determine time of last data sent (exactly as in man pages)
but it returns error: No such file or directory.

Could you help me ?

From: Rob Seace

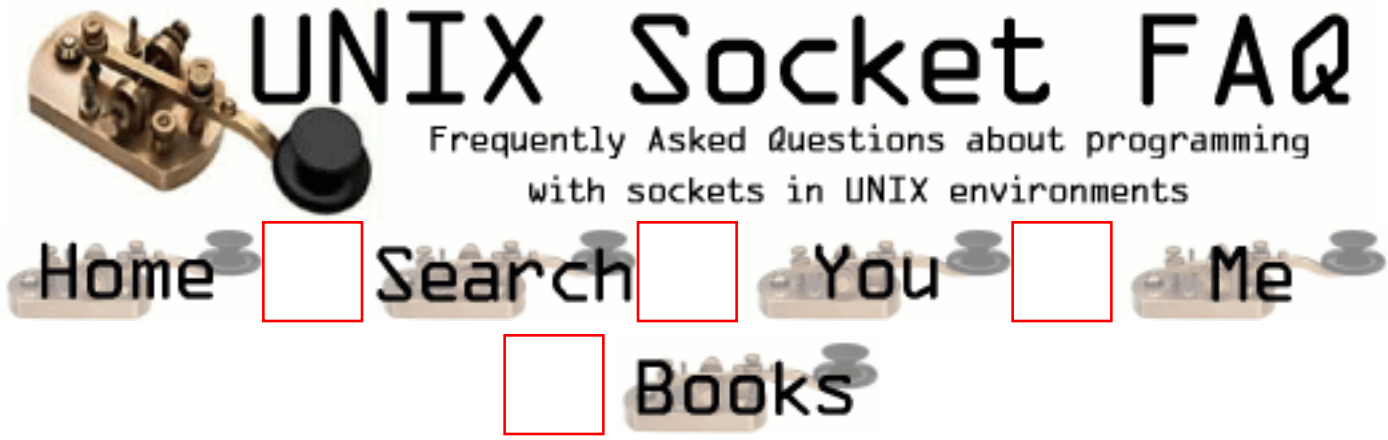
There's definitely a lower limit to the socket buffer size,
that you can't go beyond... It needs some minimum amount
just for everything to actually work properly...

There's no way to "flush" a socket buffer, if by that you
mean to force it to deliver the buffered data now... TCP
makes up its own mind about when to deliver the data... The
best you can do is coax it along, by doing stuff like

disabling the Nagle algorithm, which generally ensures
almost immediate sends of any output data...

I'm not sure why your SIOCGSTAMP isn't working... *shrug*

Add a Comment



New Questions: Problems with socket UDP

From: Jose Luis

I have an application client/server UDP. If the server this up, the client works correctly.
However, if there is not server, the client he sends correctly but when staying
to he listens to it, the function recvfrom returns the error "Refused Connection." To that can be due?
Because this happens when there is not connection settled down to be UDP?. Thank you.

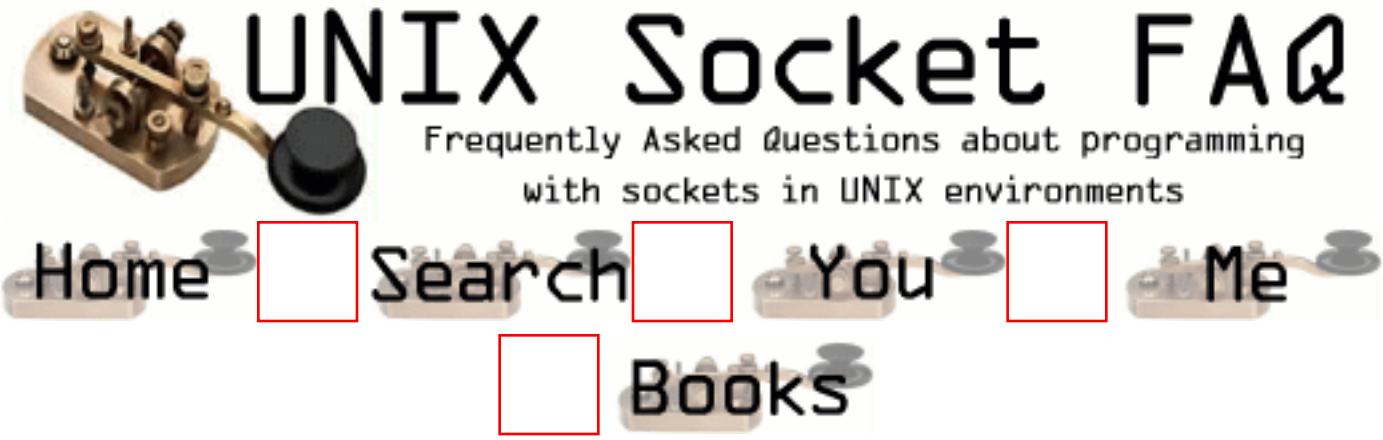
Client.c

```
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netdb.h>
#include <stdio.h>

#define DATOS "que tal estas"
#define T_MENSAJE 100

main(int argc,char *argv[])
{
int soc, lon,datos;
struct sockaddr_in dir_local,dir_remota;
struct hostent *h_puntero;
char buf[T_MENSAJE];

memset(buf,0,T_MENSAJE);
soc=socket(AF_INET,SOCK_DGRAM,0);
if (soc < 0)
{
perror("Error en socket\n");
```

```
exit(1);
}
if ((h_puntero=gethostbyname("JOSE_PC"))==NULL)
{
perror("Error en gethostbyname\n");
exit(1);
}
memcpy(&dir_local.sin_addr,h_puntero->h_addr,h_puntero->h_length);

dir_local.sin_family=AF_INET;
//dir_local.sin_addr.s_addr=INADDR_ANY;
dir_local.sin_port=htons(1040);
if (bind(soc,(struct sockaddr *)&dir_local,sizeof(dir_local))<0)
{
perror("Error en bind\n");
exit(1);
}
lon=sizeof(dir_local);
if(getsockname(soc,(struct sockaddr *)&dir_local,&lon)<0)
{
perror("Error en getsockname\n");
exit(1);
}
printf ("\nPuesto asignado: %d\n",ntohs(dir_local.sin_port));
if ((h_puntero=gethostbyname("JOSE_PC"))==NULL)
{
perror("Error en gethostbyname\n");
exit(1);
}
memcpy(&dir_remota.sin_addr,h_puntero->h_addr,h_puntero->h_length);
dir_remota.sin_family=AF_INET;
dir_remota.sin_port=htons(atoi(argv[1]));
if ((datos=sendto(soc,DATOS,sizeof(DATOS),0,(struct sockaddr
*)&dir_remota,sizeof(dir_remota)))<0)
{
perror("Error en sendto\n");
exit(1);
}
fflush(soc);
printf ("\ndatos enviados\n");

printf ("\nEsperamos respuesta\n");
memset(&dir_remota,0,sizeof(dir_remota));
lon=sizeof(dir_local);
if((datos=recvfrom(soc,buf,T_MENSAJE,0,NULL,NULL))<0)
```

```
{
perror("Error en recvfrom\n");
close(soc);
exit(1);
}
buf[datos]='\0';
printf ("\nRecibidos %d datos: %s\n",datos,buf);
close(soc);
}
```

Add a Comment

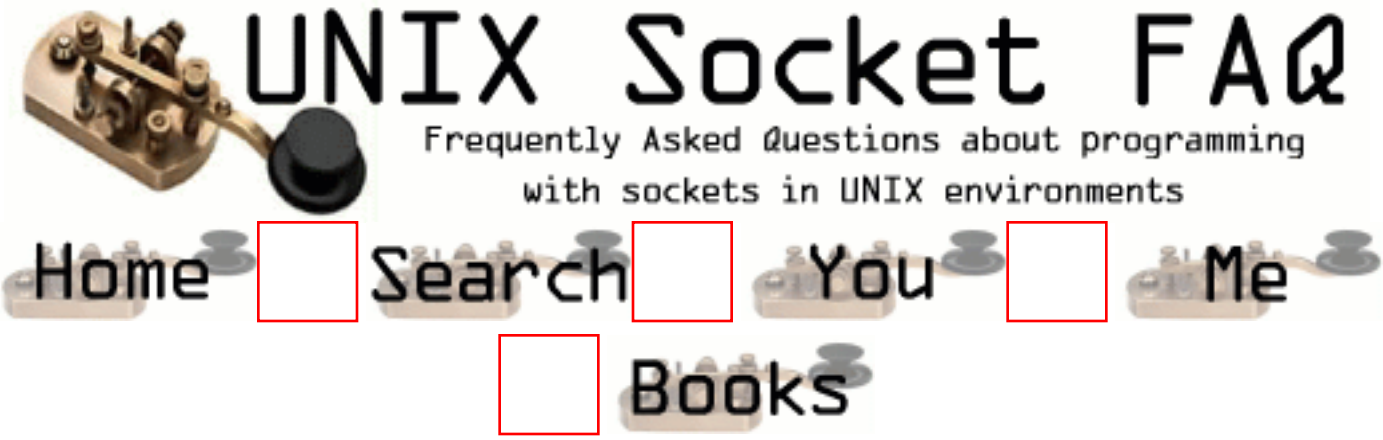


New Questions: Socket Latency

From: Nitin

What is the meaning of Socket Latency? How can i control it?

Add a Comment



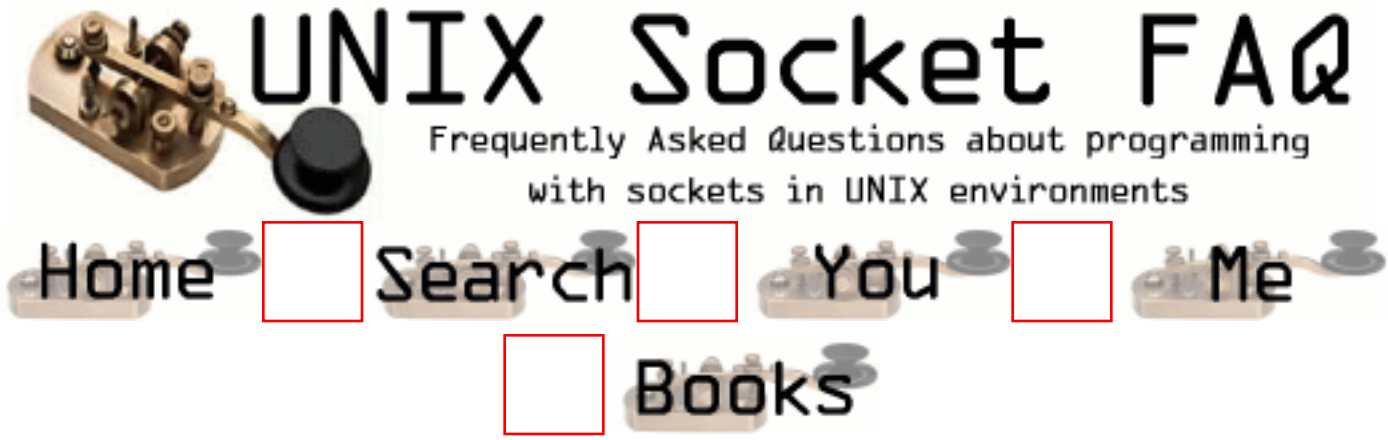
New Questions: Connect to a server

From: antonatos spyros

I am new to socket programming and I would like to make
an application in C which connects to www.quios.com and send
sms through my command line.I don't know how to send the package
needed neither the form that package has.I have made the
basic code which takes the message and the phone number from stdin
but that's it.

Thank you in advance.

Add a Comment



New Questions: RECVFROM and IOCTL Question

From: Tammy

I have a client setup on a different machine than the server. The client sends acknowledgements to the server and waits 5 seconds for a response. When the server goes into the function to read the socket, it does this:

```
stat = ioctl(fd, FIONREAD, &size);
addr_len = sizeof(source_addr);
count = recvfrom(fd, buf, size,0,
  (struct sockaddr *)&source_addr, &addr_len);
```

The Acknowledgement that the client sent was 12 bytes long, and that is what count equals after the recvfrom statement is executed. However, size is equal to 28 bytes. What can be causing this? We have another program that does a very similar thing that works correctly.

Any help would be appreciated.

Add a Comment

UNIX Socket FAQ

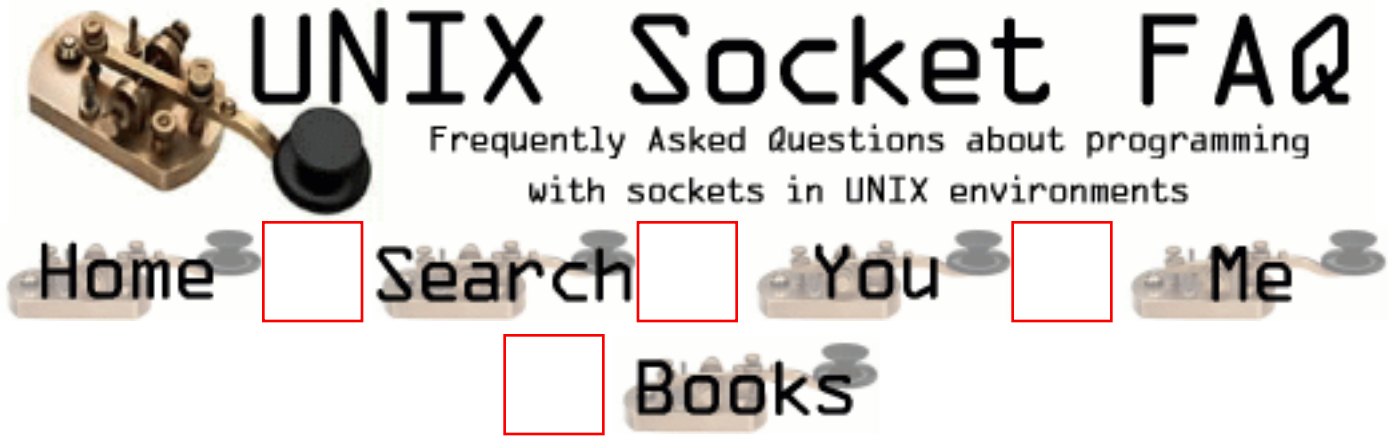
Frequently Asked Questions about programming with sockets in UNIX environments

New Questions: sockets program to retrieve web content

From: Madhu Garlanka

Is there anyone who has sample code to retrieve web content using sockets programming?
Thanks,
Madhu

Add a Comment



New Questions: How to bind a socket to multiple known ports?

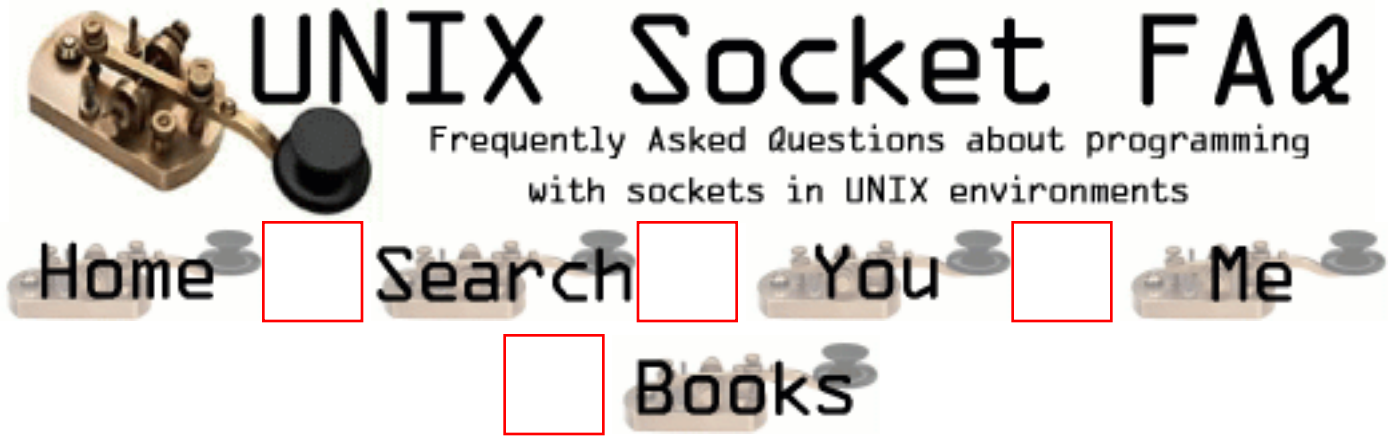
From: ed

Hi! I want to bind my server socket to more than one ports, so that it can listen to more ports. I don't know whether it is possible. Thank you!

From: Rob Seace

No, you can't bind a single socket to multiple ports... You need to just have multiple sockets, one for each port you want to listen on... Then, you can simply multiplex among them using select()...

Add a Comment



New Questions: Re: Sending files over (Socket)

From: Bryan Lee

How do i send files over socket, what commands and functions can i use?

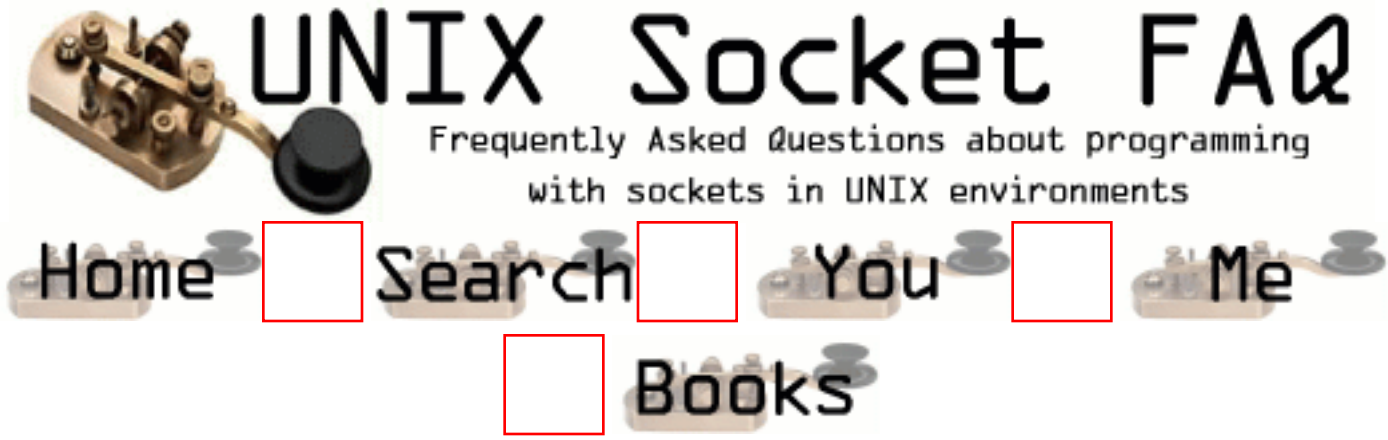
From: Mahafuz

I guess you want to transfer a file over your socket
connection. Note that socket is a very low level/crude thing.
It does not provide you with nice interfaces like FTP.
Basically FTP is built over socket connection. So you need
to crudely write the file you want to send into your socket
(standard write()) buffer by buffer and you need to read at the other end
of socket (standard read()) buffer by buffer.

I'm sure there is nothing like standard library function
like send_file(...) or recv_file(...) that will work for
socket.

You need to buld up your own.

Add a Comment



New Questions: Why Select is not realizing that there is data on socket to read.

From: Muhammad Mobeen

I have written the following code for reading data from sockets using Select but the problem is.. there is data sent by client but the select doesnot see it. If i write the read statement before the select it is recieving the data and outputting it, but if i remove it then the select waits upto the time out and then just exiting.

The code is here

```
listenfd=socket(....)
bind(....)
listen(listenfd,...)


FD_ZERO(&rset);
FD_ZERO(&allset);
FD_SET(listenfd,&allset);

for (;;)
{

  rset=allset;
  nready=select(maxfd,&rset,NULL,NULL,NULL);
  if (FD_ISSET(listenfd,&rset))
   {
     // Accepting the request from client
       connfd=accept(listenfd,.....);
       FD_SET(connfd,&allset);
       printf("Connection and select succesful\n");
```

```
    //read the data from socket and proces....
    ......
  }
```

This is v simple and i think the select should work but it is not working, it is just blocking on select.

Looking for some reply
Thanx

From: Mahafuz

You are doing select on the wrong set. you have added your
file descriptor in wrong set.

insted of

```
FD_ZERO(&rset);
FD_ZERO(&allset);
FD_SET(listenfd,&allset);
```

Do:

```
FD_ZERO(&rset);
FD_ZERO(&allset);
FD_SET(listenfd,&rset);
```

actually your allset is useless. The rule is

1. you declare a file desc set.
2. Add your desc to that set.

Since you are selecting only for read event , you should
add your file descriptor to read set.

From: gustavo

The set he is using is correct. check the line
rset=allset.
The problem is that you don't necessarily have data to read on the new socket connfd. You should call select again for this connection

---

From: Rob Seace

Actually, my guess is that it's the "maxfd" setting that's
causing the problems... I see nothing is the posted code
snippet that shows "maxfd" getting set, or updated... So,
when you accept() a new connection and add its FD to the
"allset", you need to make sure to reset "maxfd" to be one
greater than whatever the highest FD so far is... Eg:

```
if (connfd >= maxfd)
   maxfd = confd + 1;
```

If you're really already doing this, but just didn't show
it here, then I'm not sure what else might be the cause...

From: nagendra

From: Nagendra M

the solution is:

```
sockfd=socket();
fd_set read_fds;
FD_CLR(&read_fds);
FD_SET(sockfd,&read_fds);

select(maxfd,&read_fds,NULL,NULL,NULL);
```
use this

From: Hector Lasso

I think that the problem is that you are trying to copy "allset" into "rset" before calling select() like this: "rset=allset;". This shouldn't work because fd_set is a complex structure. Use "memcpy(rset, allset, sizeof(rset));" instead.

Also, Rob points another issue, in your code i can't see any reference to maxfd. I guess you are tracking the greatest file descriptor plus one in this variable.
If the memcpy() doesn't work, try a loop to fill rset:

```
FD_ZERO(&rset);
for (i=0;i<maxfd;i++)
  if (FD_ISSET(i, &allset))
    FD_SET(i, &rset);
```

I've used memcpy() before and it has worked just fine, however, looking at the fd_set structure i have found it very complex, so i don't feel very comfortable about it.

I'll just take a closer look at what the FD_XXXX macros do and how the sructure is filled before using memcpy() again.

Good luck

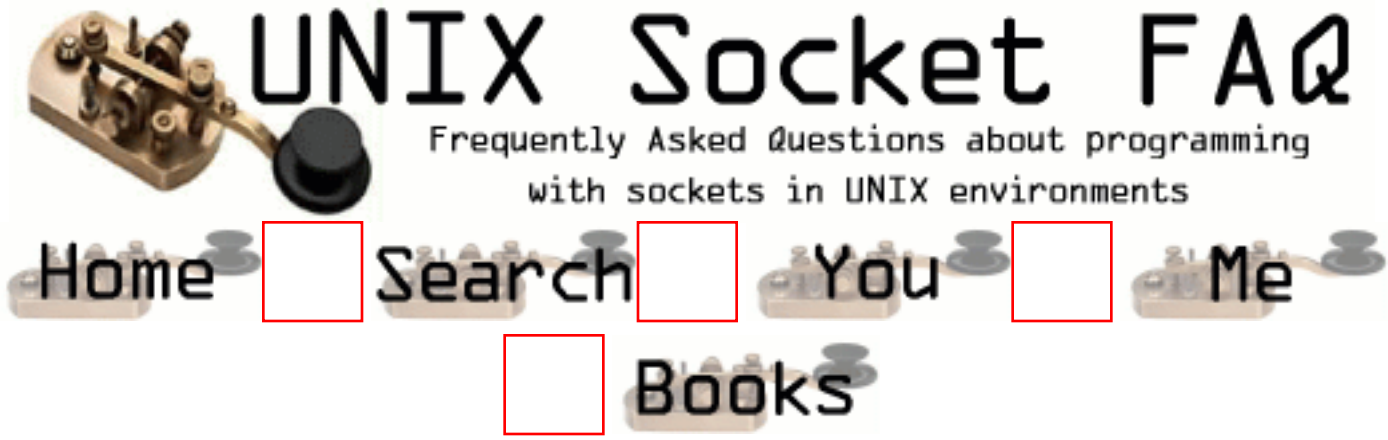
From: Rob Seace

Actually, most modern compilers will allow you to do structure assignments like that, just fine... But, I personally don't like it too well either, and generally use the memcpy() approach, too... (And, you forgot the "&"s in your memcpy() call, to take the address of the fd_sets...)

From: Prakash

After select the read,write and exception sets gets cleared , so we require to set all again.
Hence all the FD_ZERO and FD_SET should be placed inside 'for' loop.
Best Regards,
Satya

Add a Comment



New Questions: How to get ISP assigned IP

From: Mahafuz

Hi,

I'm writing a TCP/IP server app and i want to bind() my server to a socket with the IP address dynamically assigned by my ISP. Assume that i'm connected to internet while i'm starting up my server. My problem is, how can I find out (programmatically) the ip address assigned by my ISP.

-- Mahafuz

From: Rob Seace

First of all, do you REALLY have a need to bind() directly to the IP? Most servers simply bind to INADDR_ANY (ie: 0.0.0.0), and then they can receive incoming packets on any local interface, regardless of IP...

However, if you really do need to get the IP, you can... First, you'll need to know what the name of the interface is... If the IP is being dynamically assigned, I'll assume it's probably a PPP connection, yes? If so, depending on your exact flavor of Unix, the interface is likely to be named "ppp0", or something similar... (Just do "ifconfig" from a command line when you're connected, to find its name; and, the current IP, for that matter...) Then, you do something like:

struct ifreq ifr;

```
int fd;

fd = socket (AF_INET, SOCK_DGRAM, 0);
if (fd < 0)
   /* fail here */

strcpy (ifr.ifr_name, "ppp0");
if (ioctl (fd, SIOCGIFADDR, &ifr) == 0) {
   /* it worked, and "ifr.ifr_addr" is the raw IPv4 */
   /* inside a "struct sockaddr"; copy to a "struct */
   /* sockaddr_in", then extract "sin_addr"... */
} else {
   /* it failed */
}

close (fd);
```

Of course, if you have a hostname, you can just do a reverse name lookup on it, which would be a lot easier... But, since you mention it's dynamically allocated, there's a good chance you don't have one...

From: Mahafuz

Thanks Rob. Great help.

Though i've found the way but you helped me a great with the ioctl constant name. I was using the wrong constant (destination interface address- SIOGCIFDSTADDR) which was giving me the ip of othre end. I've found it from ifconfig source.

Now i want to query my system for available interfaces. I've seen that ifconfig does it by reading proc file system. It seemed to complex code to me. I have little knowledege about proc files. So i couldn't understand very much.

I'd appriciate help about querying my system for available interface & proc file structures.

Also where do i get a very good documentation about various system defined constant (like POSIX constants) and other constants (fclt,ioctl etc).

Thakns again for help.

-- Soyuz

From: Rob Seace

Well, on my system (Red Hat Linux 6.2), there's a set of glibc functions available for easily getting at the list of local interfaces: if_nameindex() is the main function that returns the list of interfaces... (It's defined in <net/if.h>... It returns a list of structs that contain the interface name and index... It's terminated by a struct with a NULL name...) I don't know if such a function exists on your system, or not... But, there are ways to walk through the list of intefaces with other ioctl() calls, as well... SIOCGIFCONF is the main one of interest... But, it seems rather complex to use... You can find an example of use in the "libpcap" library, in "inet.c"...

As for a list of ioctl() calls, again on my system, I can do "man ioctl_list", to receive a fairly comprehensive list of the available ones, with minimal info about each one...

From: Mahafuz

My system is RedHat 6.1.

I Couldn't find if_nameindex() in man pages. But its there in net/if.h. Thanks.

From: Lost

just for anyones reference, on linux, you can just use get_myaddress() like this

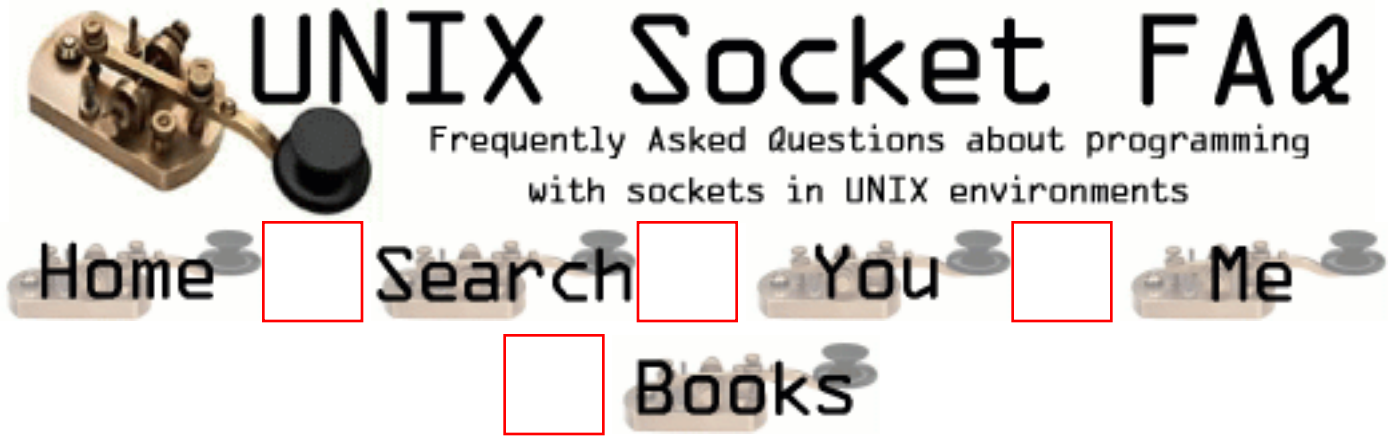
struct sockaddr_in my_addr;

get_myaddress(&my_addr);
printf("Your ip is %s\n", inet_ntoa(my_addr.in_addr);

From: lost

add another) to the end of that printf :)

Add a Comment

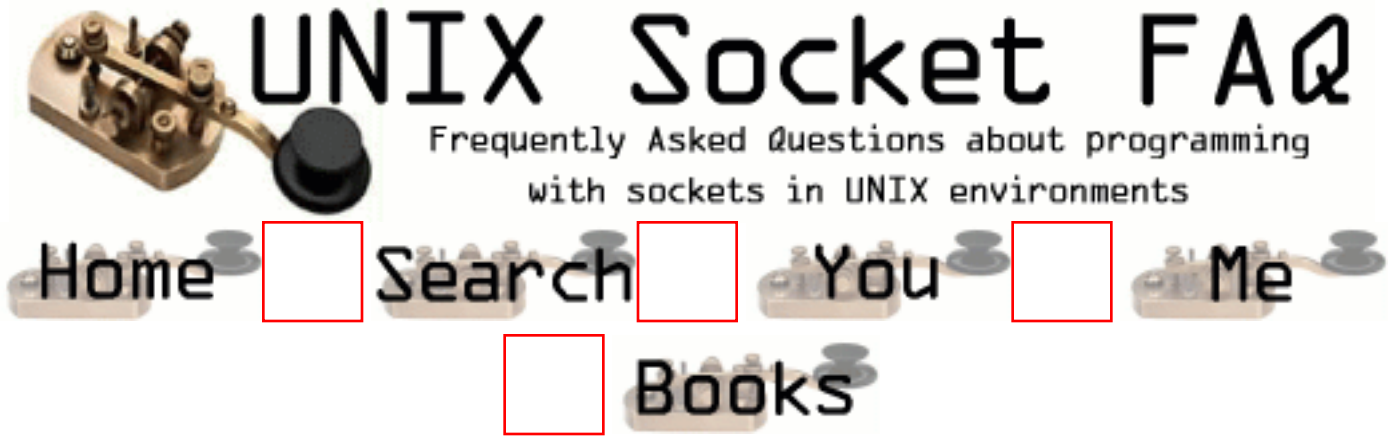


New Questions: UDP has connection in linux?

From: meng

I sent message from host A to B useing UDP in linux. When B close the socket, in A I got a "connection refuse" system warnning.In a few minute, even if B bind to the UDP port again,from A still cann't send message to B.Usaging tcpdump on A, I found a ICMP package of "network unrecahable".
How can I resend message after B is restart quickly?
Is there virtual conection in linux for UDP?
In Winnt, this will not happen?
Who can help me?

Add a Comment

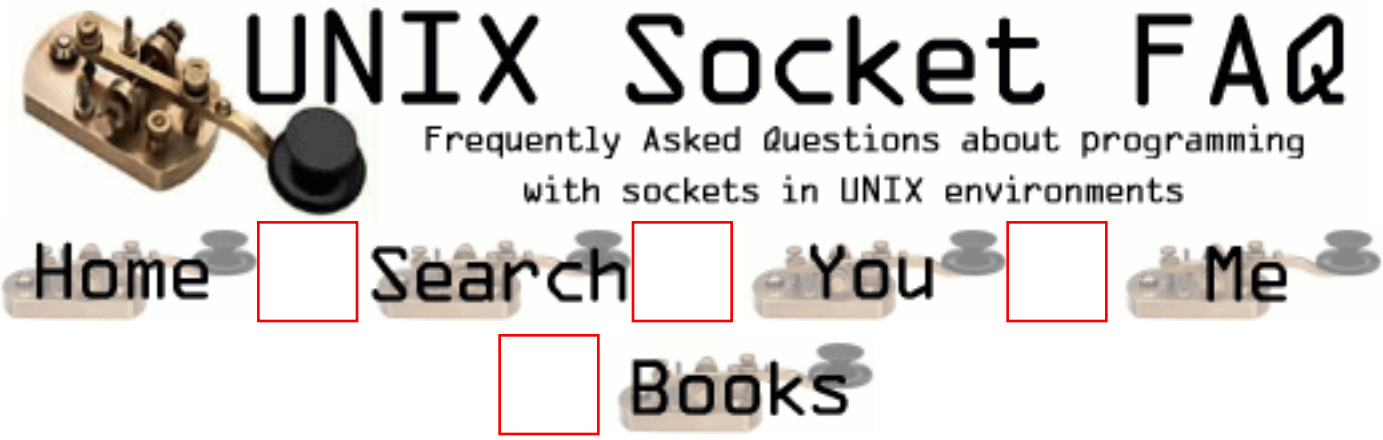


New Questions: UDP has connection in linux?

From: meng

I sent message from host A to B useing UDP in linux. When B close the socket, in A I got a "connection refuse" system warnning.In a few minute, even if B bind to the UDP port again,from A still cann't send message to B.Usaging tcpdump on A, I found a ICMP package of "network unrecahable".
How can I resend message after B is restart quickly?
Is there virtual conection in linux for UDP?
In Winnt, this will not happen?
Who can help me?

Add a Comment



New Questions: Why UDP is used for the Internet instead of TCP?

From: K.Kumar

Why UDP(unreliable protocol) is being chosen for the Internet instead of TCP which is reliable?

From: Rob Seace

Um, what exactly are you talking about?? I'd say TCP is
used FAR more widely on the Net than UDP is... But, UDP
does have its uses... (Eg: Quake is a perfect example: you
need the absolute blazingest fast speed possible, with 0
overhead, and also don't really care if you drop a few
packets here and there...)

From: K.Kumar

U mean that the http uses connection-oriented protocol (TCP) (maintaining session)? Explain about Quake.

From: Mahafuz

Don't mix up underlying IP protocol with UDP.
Both UPD & TCP are implemented using IP protocol (low level). IP is connectionless & unreliable. TCP is based on IP but provides you with connection oriented & reliable communication. It uses its own technique to make a reliable
connection. On the otherhand UDP is high level IP (not exactly true) in plain words. Its connectionless & unreliable. Both of them have their uses according to the need. It seems that TCP is always better for use but as you
see rob mentioned UDP uses by quake. Its basically reliability Vs performane issue.

But at low level both are implemented using IP - like UDP connetionless & unreliable.

--- Mahafuz

From: Magnus Boden

And you can't multicast or broadcast with tcp.

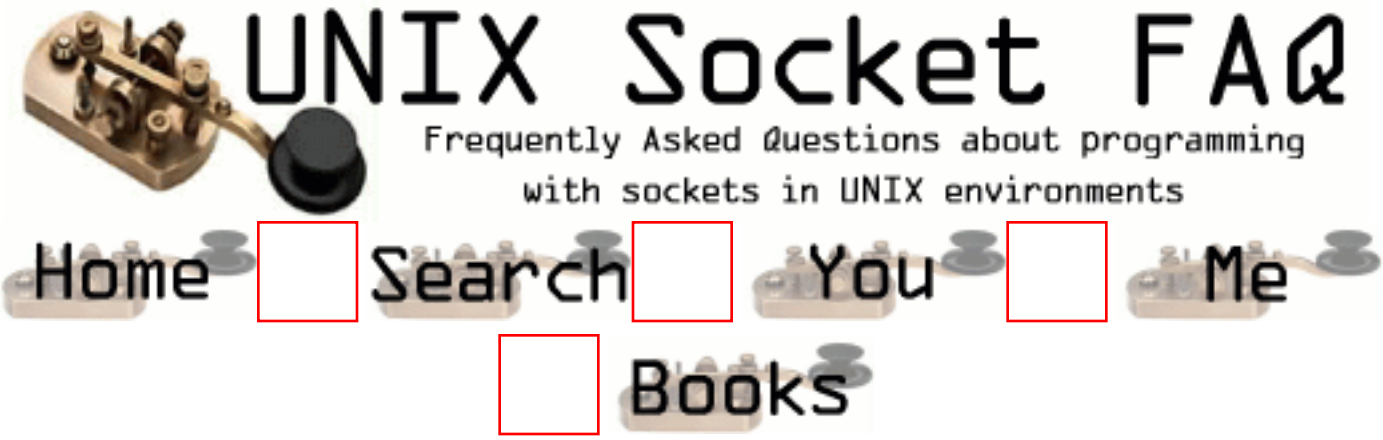
From: Stephen Silvey

At the following link, click on the Networks Section under "Table of Contents" for a good overview of Network Protocols:

http://www.sftw.umac.mo/resources/LDP/LDP/tlk/tlk.html

Happy computing!

Add a Comment

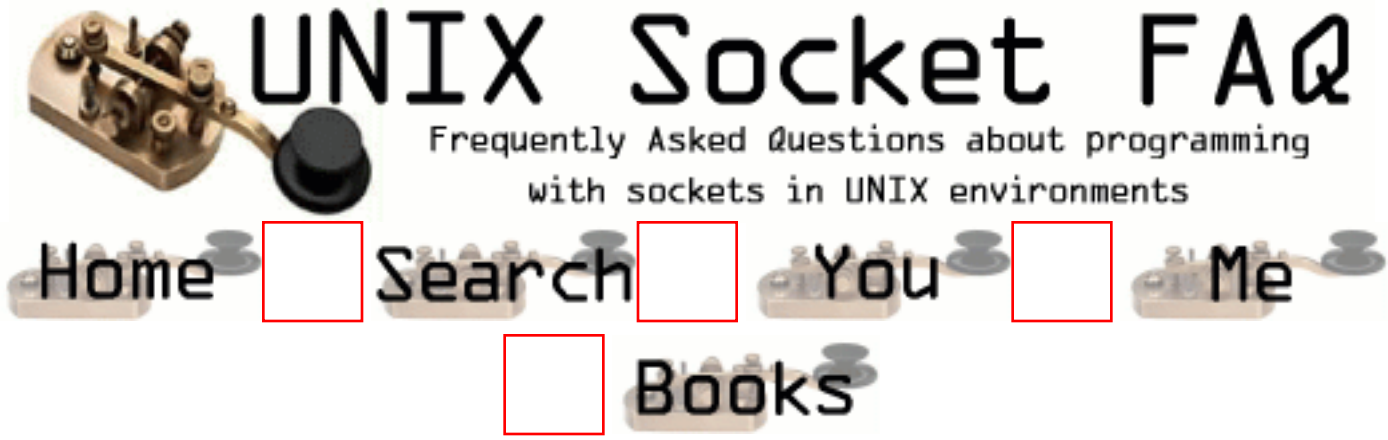


New Questions: sockaddr structure oddity

From: ak

Why do the generic socket address and the protocol specific socket addresses have the first 2 fields common (unit8_t and sa_family_t) ?

Add a Comment



New Questions: Can same port be used by two processes?

From: Mohan Rao Kakumani

Hi all,

Is it possible, for two processes in the same system to use the same port i.e while issueing socket(..) system call, can we give same port number in both the processes?

However, one process only uses the socket(port) for sending messages, while other uses it only for receiving messages.

Is there any ways to get around of the above problem. Normally when we issue socket() system call with the same port number, it gives error. But, i want to open sockets in the same port, in two processes of same machine, one used for reading while the other for sending.

Please through up your ideas to solve this problem.

Thanks for patient reading

From: Bruce Edgerton

Mohan, yes. Go to this link:

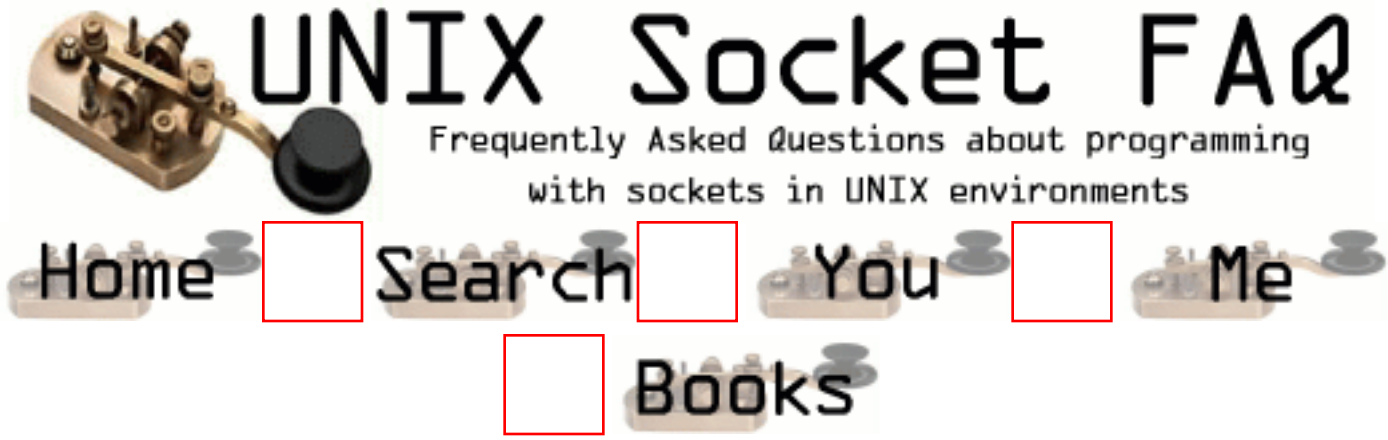
http://www.ecst.csuchico.edu/~beej/guide/net/

It reveals all.

From: eric

yes it's possible. use setsockopt SO_REUSEADDR...

Add a Comment



New Questions: how to choose my own source port

From: ziv

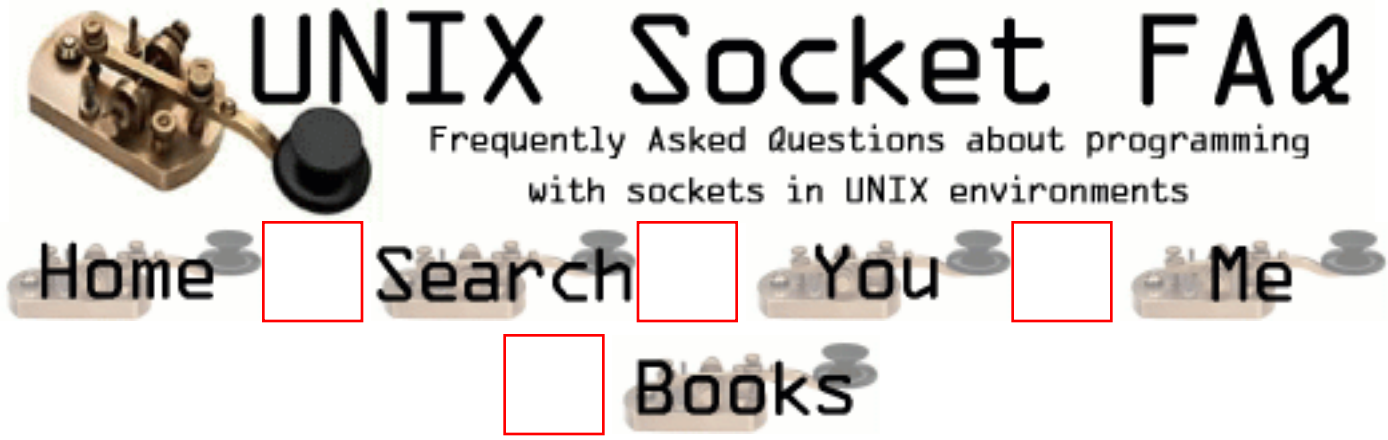
Hi, the problem we're facing is that when we want to send a udp message, the operating system chooses a source port for the packets. Problem is, we want to packets through a source port chosen by us. How?
ziv
p.s. we're using sendto() for the sending.

From: Mahafuz

After creating the socket and before using sendto(...)
use bind(...) function to bind the socket to your chosen
interface & port using the sockaddr_in structure.

note: don't assign the port number = htons(0)
This means that OS will assign a port for you.
use for example: htons(1025) for port number 1025.

Add a Comment



New Questions: Socket Address Structure

From: Kidd Smith

Why is it necessary that the first two fields (unit8_t and
sa_family_t) be defined in the same order for in both
the generic sockaddr structure and protocol
specific sockaddr structure?

The Generic Socket Address Structure:
Struct sockaddr
{
unit8_t sa_len;
sa_family_t sa_family;
char sa_data[14];
}
Ipv6 Socket Address Structure
Struct sockaddr_in6
{
unit8_t sin6_len;
sa_family_t sin6_family;
in_port_t sin6_port;
uint32_t sin6_flowinfo;
struct in6_addr sin6_addr;
}

From: Mahafuz

By using the first filed, who to interpret the rest of the address is determined. Since there are various protocol specific address structues and the fucntions like
connect(...) are protocol independent(they take pointer to generic address structure) , they uses the first field to determine the type of the address structure and uses the rest of the strucure.

From: David Johnson

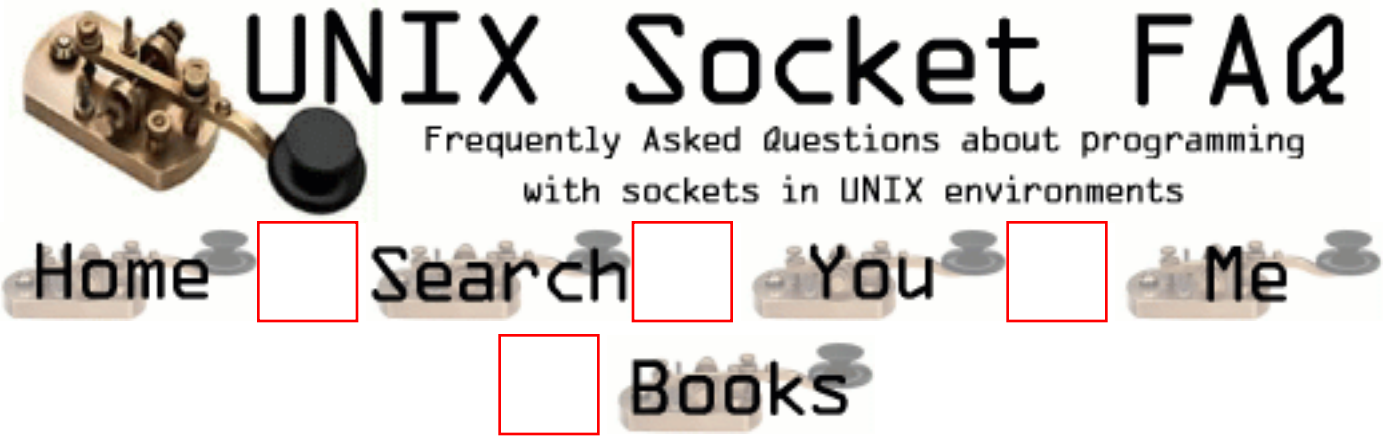
i think it is mainly for compatability.
only the first two fields are really necessary to
satisfy the two main requirement: identification and
length limit. usually the protocol specific address
structures tend to have more details(usually not extremely
essential). so if we were to implement something that
is protocol independent, we only need to cast the protocol
specific address to the generic one, and no essential
information is lost.

From: Mahafuz

For each protocol there is a protocol specific socket
address strucutre. But net functions like connect(...) are
protocol independent (which they should be). To make this work, there is a protocol independent socket structure 'sockaddr'. Protocol specefic socket structure like 'sockaddr_in' for TCP/IP is passed as sockaddr since these function expect them to be type generic sockaddr. Now first field help these functions to determine how many bytes there are in the structure & second field helps to determine the type of protocol famaly. After that the rest of the structure is interpreted accocding to the value of protocol family. So all the protocol specific socket structure has these two fields common with the generic
socket structure.

Without this convention there has to be protocol specific functions. ie. connect_XXXX(...) where XXXX is the protocol family.

Add a Comment



New Questions: IPC

From: Kidd Smith

What would it take to extend primarily local IPC
mechanisms such as pipes, FIFO, and message queue
to network (IPC mechanisms)?
Which one would be easiest and why?

From: Rob Seace

Hmmmm, well I'd guess that pipes would probably be the most
straightforward to convert over to sockets... Mainly,
because the I/O API remains the same (they're still just
file descriptors, which you can read()/write()/whatever)...
However, there are some notable differences: mainly, sockets
are bi-directional (full duplex), while pipes are only
uni-directional... So, what you needed two pipes to do
before, can really be done with a single socket... But, if
you wanted to get by with the least code change possible,
you could just kluge around that by dup()'ing the socket FD,
to create an artificially separate 'reading' vs. 'writing'
end...

Of course, the easiest local IPC method to extend to network
IPC would be Unix domain (AF_UNIX) sockets... ;-) If you're
designing something new to communicate locally, which you
expect to later extend to network communication, I would
definitely recommend going with AF_UNIX sockets... It's
generally got a BIT more overhead than plain pipes/FIFOs,
but it's usually not enough to matter for most uses, unless
you need absolute max throughput and/or lowest possible
latency... Personally, I've been quite happy using Unix
domain sockets for most things...

From: Jennifer Baker

shared memory is out.
pipe ok if given a name and this is fifo so fifo is ok
message queue: since kernel persistent, ok since has
so that other object can referenence the queue. there is
a queue identifier and a permission structure maintained
for each queu.

Add a Comment

UNIX Socket FAQ

Frequently Asked Questions about programming with sockets in UNIX environments

New Questions: XDR

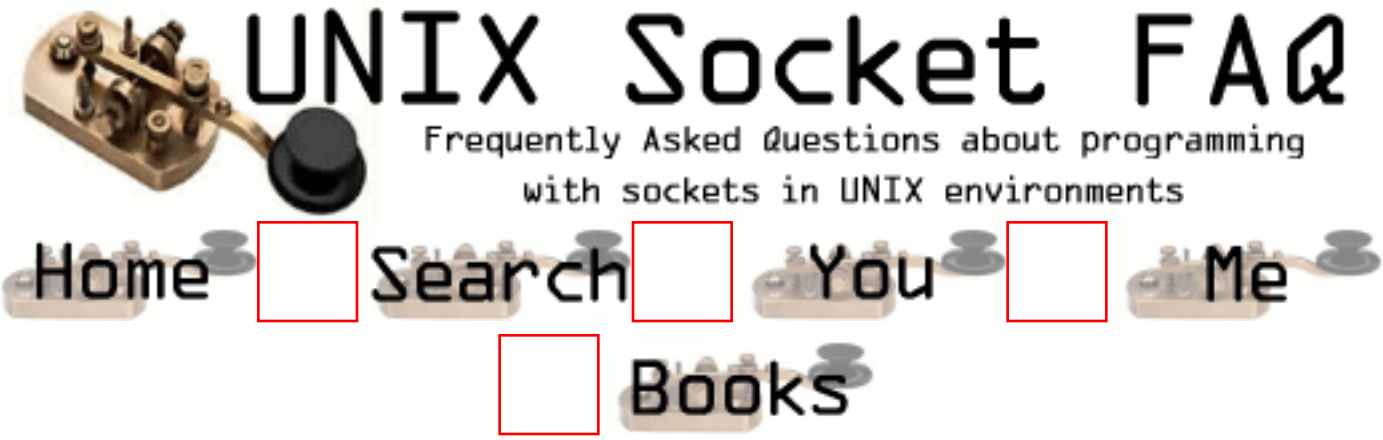
From: Kidd Smith

When dealing with heterogeneous machines (i.e. different
platforms such as Little Endian and Big Endian),
we have primarily two ways of dealing with format
conversion. One way, known as the
external data representation, has the sender mapped
its format to the network format and the receiver
converted from the network format into its own format.
Apollo implemented the other approach.
The sender here has to include a tag
indicating what its format is. Upon receipt,
the receiver consults a format conversion table
and does the appropriate conversion.
Besides being a little less efficient
(in the first approach there are two
unnecessary conversions: from sender
format to network and from network to receiver),
the first implementation is comparable to the second.
When (under what conditions or circumstances)
 then is it more appropriate to use one over
another and why?

From: Jennifer Baker

universal standard as in the case of XDR is always better.
everyone knows what to expect and what's expected of him/her.

Add a Comment



New Questions: accept() error!

From: zhangg

urgent need help!!!

```
/*
explain: I use accept() function to wait Client connect ,
but accept() error when client connect number more than 19.
errno is :bad file number .
*/

/*following is my server code: */

struct sockaddr_in ServerAddr,ClientAddr ;
int SockAddrLen;
char buff[100];

if((s = socket( AF_INET ,SOCK_STREAM , 0 ))==-1)
{
exit(0);
}

bzero((char *)&ServerAddr, sizeof(ServerAddr)) ;
ServerAddr.sin_family = AF_INET ;
ServerAddr.sin_port = htons( 9999);
ServerAddr.sin_addr.s_addr=htonl(INADDR_ANY);

if (bind(s,(struct sockaddr*)&ServerAddr,sizeof(ServerAddr))==-1)
{
exit(0);
}
```

```
if ( listen(s, 1 )== -1 )
{
exit(0);
}

if( (AcceptPID=fork())==0 )
{
//accept loop
for( ; ; )
{
SockAddrLen=sizeof(ClientAddr);

     /*accept() error when client connect number more than 19. */
new_s=accept(s,(struct sockaddr*)&ClientAddr,&SockAddrLen);

if ( new_s== -1)
{
//accept error!
//errno is : bad file number .
                        exit(0);
}

/*do something */
}//continue wait connect
}
```

From: Mahafuz

After you are done (after /* do something */)
close the socket you got from accept(...)

For your code:

/*do something */

From: Mahafuz

Please try the following:

After you are done (after /* do something */)
close the socket you got from accept(...)

For your code:

```
/*do something */
close(new_s);
```

From: Mahafuz

sorry misunderstood your problem.

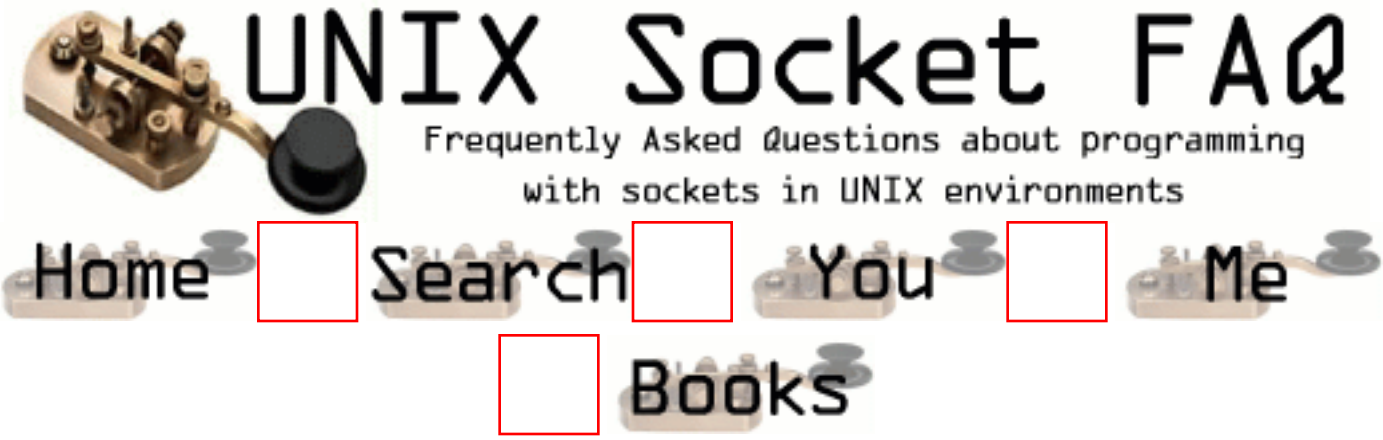
I advised the wrong thing.

Pls make sure you are not closing the socket descriptor - "s" on which you are acceping, somewhere in
/* do something*/

From: zhangg

Thanks for Mahafuz,
This Question have resolved!

Add a Comment



New Questions: How to post data using Post method to a servlet

From: Sudipto Basu

Hi,

I am facing a problem of how to send data to a servlet which is in a webserver.

Look, the servlet have some text fields like name and id. and i want to post this two parameters from my client program.

The servlet is situated in a webserver which I can access through the program and getting the Data out of it using GET / method. And the servlet support both GET and POST methods.

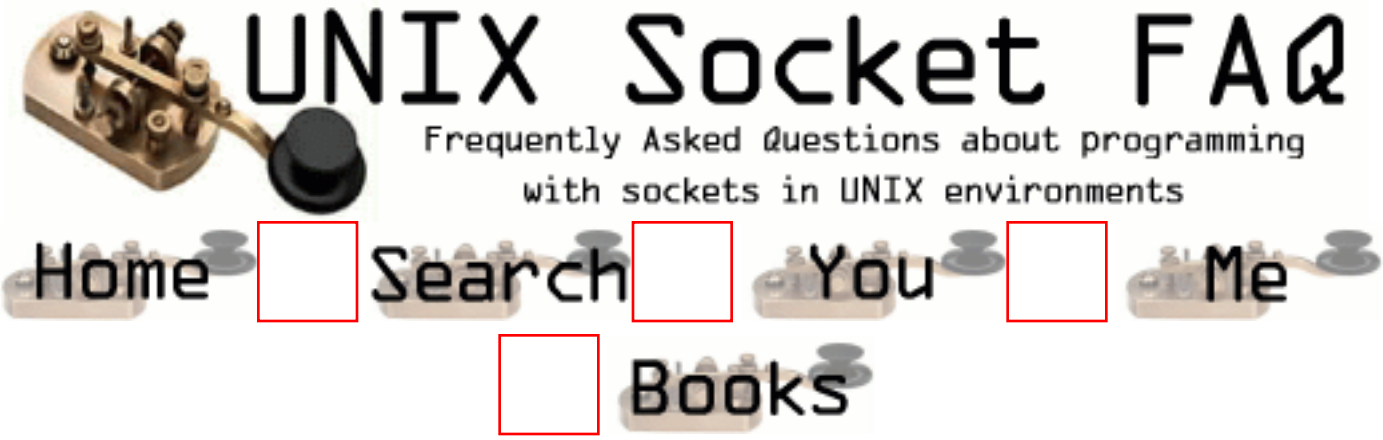
Have you got any answer please let me know. I will be thank ful to you for this

with regards,
Sudipto Basu,India

From: Amit Kumar

U wanna see this link :
http://www.ibiblio.org/javafaq/javafaq.html#xtocid900061

Add a Comment



New Questions: my program gets stuck when closing a socket.

From: Ma'ayan

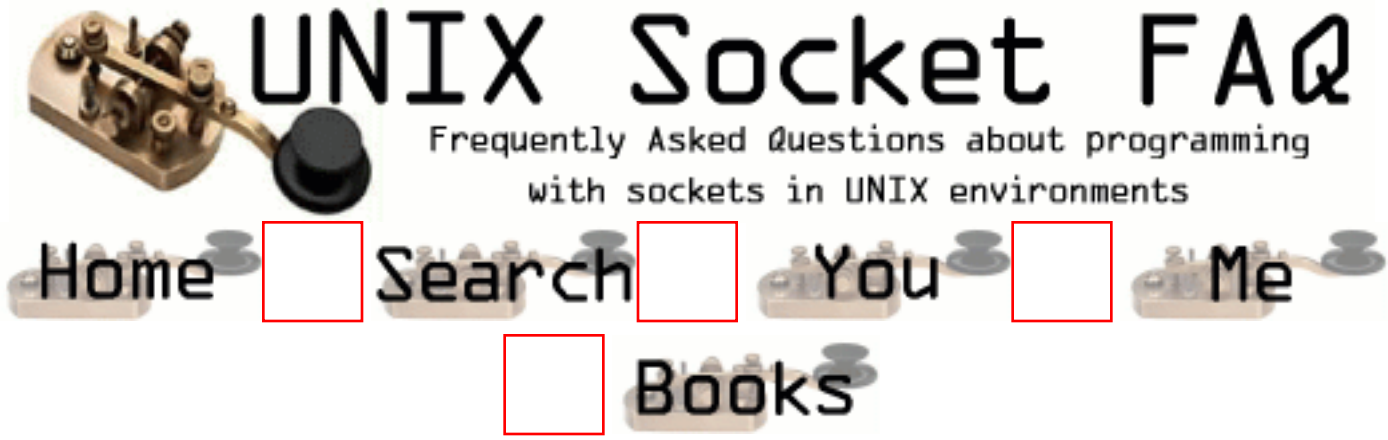
Hi,

I have a problem with closing a socket.
My program gets stuck when it tries to close the socket.
I've tried to use a non-blocking socket but it doesn't help.

Does anyone have an idea what can cause this ?

thanks in advance,
Ma'ayan.

Add a Comment



New Questions: why does bind() give EADDRINUSE

From: Duane Ellis

I have a fork()ing server application, that I need the ability to quickly kill and restart. If there are no clients attached at the time the server goes down and comes right back up all is well.

If there is a client attached to a fork()ed instance of the server, and the server goes down (ie: during a restart) the client dies. When the server comes back up and trys to re-bind()to the same port bind() fails with EADDRINUSE.

If I wait some period of time (more then 30 seconds) it works. Why?

ANSWER: When the server dies, the open socket enters the TIME_WAIT state, basically TIME_WAIT is suppose to handle any floating packets still enroute between the two machines. During that time, the port number you where using is 'IN USE' (so that it can safely ignore the lost packets that majically re-appear)

To let your server come back up quickly, you need to call the function setsockopt() turning on the option "SO_REUSEADDR" like this:

s = socket(....);

int flag; flag = 1;

setsockopt(s, SOL_SOCKET, SO_REUSEADDR, &flag, sizeof(flag));

... then ...

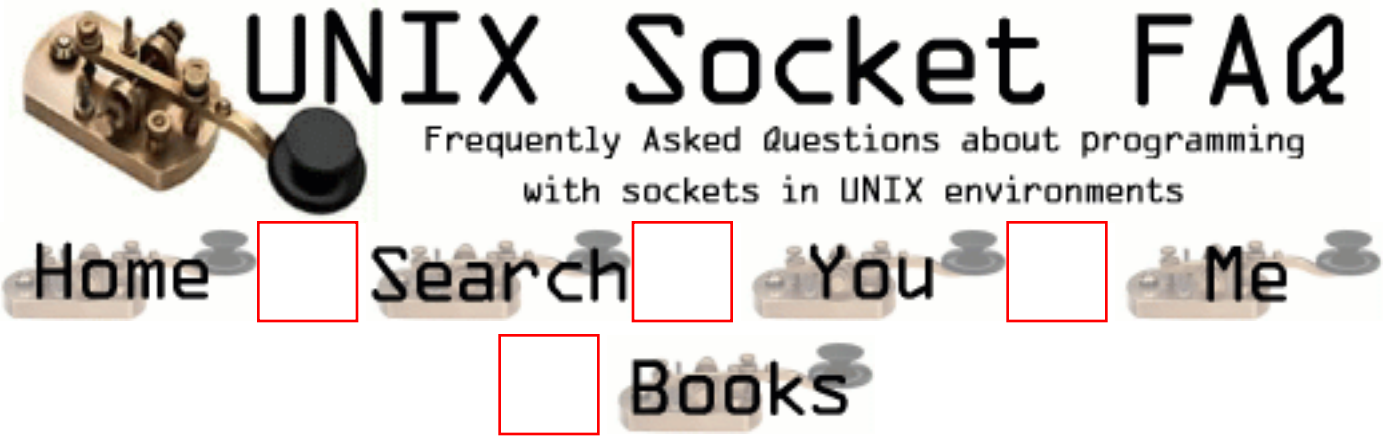
bind(s, ...);

This took quite a while for me to find, and might well be the source of alot of the TIME_WAIT confusion.

From: Rob Seace

Well, that's true, but this exact issue is already covered quite extensively in the FAQ proper: 4.1, 4.5, and 1.7... ;-)

Add a Comment



New Questions: Compilation problem

From: Minko Govedarov

Hi, I have a problem compiling a client application.
When #include <netdb.h>, which I need for the struct sockaddr_in
I get a compilation error:
...
... #error "No DATAMODEL_NATIVE specified"

I work on Solaris on alpha
Any help would be appreciated.

Thanks,

Minko

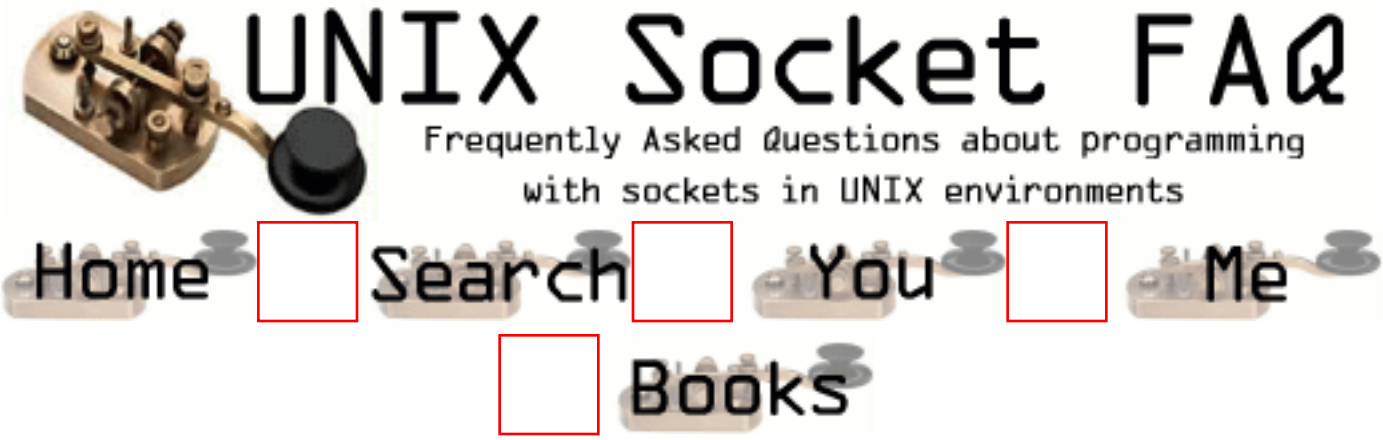
From: Minko Govedarov

... i fixed it. I forgot I was on a 64 bit architecture :)

regards,

Minko

Add a Comment



New Questions: live AP_INET SOCK_STREAM connection

From: Flawd95

what steps do you need to take in makng a connection that constantly reads from a server? I've managed to get it connected, but would like some help in how I can "monitor" the socket for incomming data without having to recv 100 times.

From: flawd95

sorry for the double post. an example of what I want to know would be something like an IRC client. as the data comes, it comes up in the window. I'm brand new to socket programming. I'm sure I can manage do display the information (I hope :)) I'm guessing that there's a loop somewheres?

From: Rob Seace

Well, you have a few choices, depending on your needs:

1. You can just do normal read()/recv() on standard blocking sockets, and display the data as it comes in... This is really only good if dealing with a single client at a time, and you don't need to do anything else while waiting for the client to send you data...

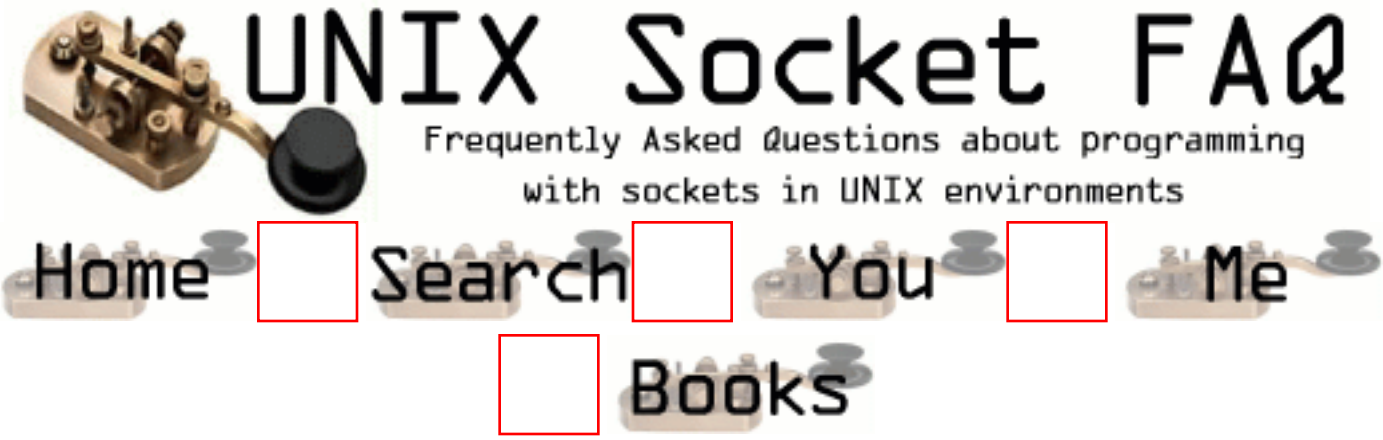
2. You can set the socket(s) to non-blocking mode, and then periodically try to read from them, and just go back to what you were doing, if they've got nothing to send you... Some people like this, but I've never been a fan of non-blocking I/O, myself...

3. You can use select() to check for incoming data that is ready to read... You can have it use a configurable timeout

wait for the data to arrive, or wait forever, or not wait at all (just do a simple poll)... And, it easily handles many simulataneous sockets, allowing you to easily multiplex between several connected clients, servicing whichever ones are ready... I'm a big fan of select()... ;-)

For something like an IRC server, I think you'd definitely want to use a select() approach... You select() for readability from any of the connected clients, and then after reading their available data, you send it off to all the others...

Add a Comment



New Questions: TCP/IP issue on HP Unix

From: ravinder Singh

The issue is that I have 2 demon process both for reading and writing into the same socket in TCP/IP. The first process creates the socket, but second process gets the socket id from a file as first process stores the same into this file. But the second process is not able to write into that socket.

Is there some constrain for the same or do we need to have some privilege for the same. The error that I get is : Value after write function [-1]<send_msg> write buffer fail - errno: <4>

If I run these programs separately, both run perfectly.

If you have any ideal please reply immediately.

From: Magnus Boden

Are both sockets supposed to use the same socket?
The socket number 5 is not the same as socket 5 in another socket.
You can send a filedescriptor (same as socket) using sendmsg.
You have to have an pf_unix connection to send the other descriptor over.
You can read about that in Unix Network Programming Volume 1 page 381 by Richard Stewens.

From: Magnus Boden

LINE 1: Are both proccesses supposed to use the same socket?

Correction from previous comment.

From: ravinder Singh

yes both process r suppose to use same socket

From: Btvrao

.

Sir, file descriptor is an index to a table of files that are opened associated with perticular process.

So, with different processes same file having different descriptors.

here two processes sharing common socket is not possible.

this my level best.

From: Btvrao

From: Btvrao

.

Sir,

so solution to above problem is simple by creating
stream pipe. a stream pipe is a full duplex(bidirectional)
for pasing file descriptorsmbetween two unrelated processes like two demons. these stream pipes are eqivalent to UNIX domain sockets. here we have to use loop around driver.

Add a Comment

New Questions: select() between a UDP socket and a TCP client socket

From: oemar

Hi all,

I'm having this problem with select. You see I have a TCP client socket (connected to a tcp server) and a UDP socket running together inside a loop.

this is what I do (in psuedo):

```
 create tcp client socket;
 connect that socket to tcp server;
 /* here the tcp client socket successfully connected */
 create udp socket;
 /* here the udp socket successfully bound to a port */
 set time out for select();
 now use FD_ZERO()
 now set my FD_SET();
 while(1){
    rc = select (between the FD_SET);
    if rc>0{
       if FD_ISSET (from UDP)
       {data = read(UDP);
        pass data to TCP socket}
       if FD_ISSET (from TCP)
       {data = read(TCP)
        pass data to UDP socket)
    }
    etc...
}
```

/*end of psuedo */

The size of data travelling between these sockets was set to 4k, time out in select() is set to 1 second.

Now the problem.. when I send data from the tcp server, select() doesn't return anything (still blocking). When I send data from a peer to the UDP port the same thing happened, select still blocks. I checked the select FD_SET and it's waiting on the right sockets.

I need your opinion on this:
- Is what I'm doing here(mixing TCP and UDP under the same select function) acceptable (can be done)?
- Does my 4k data size is the one causing problem? FYI, the tcp server and udp peer are running inside the same address as the client.
- can you do a select() with TCP clients?

that's all.. Thanks a million.. :-)

From: Aaron Stein

I am having almost the same problem. I added a TCP socket fd and a UDP socket fd to a read set. Then I use select on the read set. My situation is a little different in that when a tcp connection is made, select returns correctly, however, when a udp broadcast is received, the select doesn't return. Through netstat I noticed that there is data in the recv queue of the udp socket. One other weird thing - my FD_ISSET() check on the udp socket actually is successful ONLY when a tcp connection is made (after there is already queued data for the udp socket)...weird

any help??

From: orna

maybe it will sound simple
but do u both add one to the largest descriptor number
and set it in the select first parm function

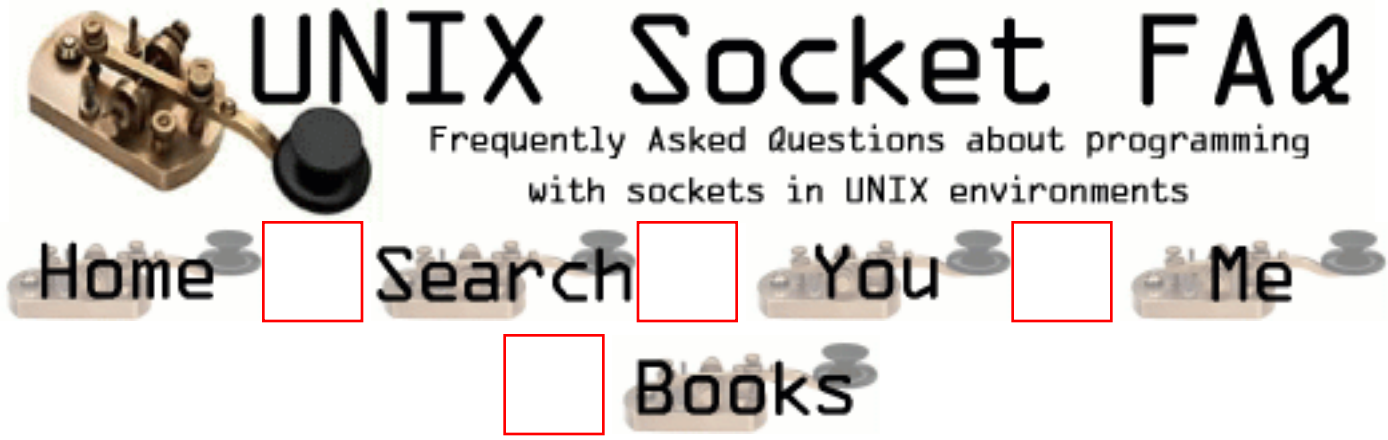
From: Hector Lasso

Hello,

It's not clear for me something:
Are you FD_SETting() inside the loop?
If you are not, remember that select() modifies the fd_set struct to let you know which file descriptors are ready for read/write/except... (I just can't be completely positive about this because it is not written in the documentation... however, it says: "... On success, select returns the number of descriptors contained in the descriptor sets, which may be zero if the timeout expires before anything interesting happens..." - man 2 select)

I can tell you that select() works on file descriptors, whether they to sockets (of any kind) or to files.

Add a Comment



New Questions: recv gives a segmentation fault.

From: Bharat

Hi,

I am getting a segmentation fault with the recv system call.
(I know u r thinking "So, what?" :-> but pl. read on..)

I setup a TCP conection between the server and 10 clients(yaah some people do petty things and still get stuck in it:->). The client is able to send a message to the server and the server receives it, but when the server sends it a message, the client gets a segmentation fault in its recv call.

I would believe that if there was some problem with the way recv is called, it would return -1 setting the pertinent errorno. But, that doesn't seem to happen. I do assign the buffer of the required size(in case smbdy has a doubt about that).

I check the connection using getsockname() and getpeername() before the recv, and everything looks in order.

Can anybody come up with a scenario wherein the recv will give you a segmentation fault? Would really appreciate it if anybody could give me a clue, 'coz I am completely clueless(:->) about this.

Thanks
Bharat

From: Magnus Boden

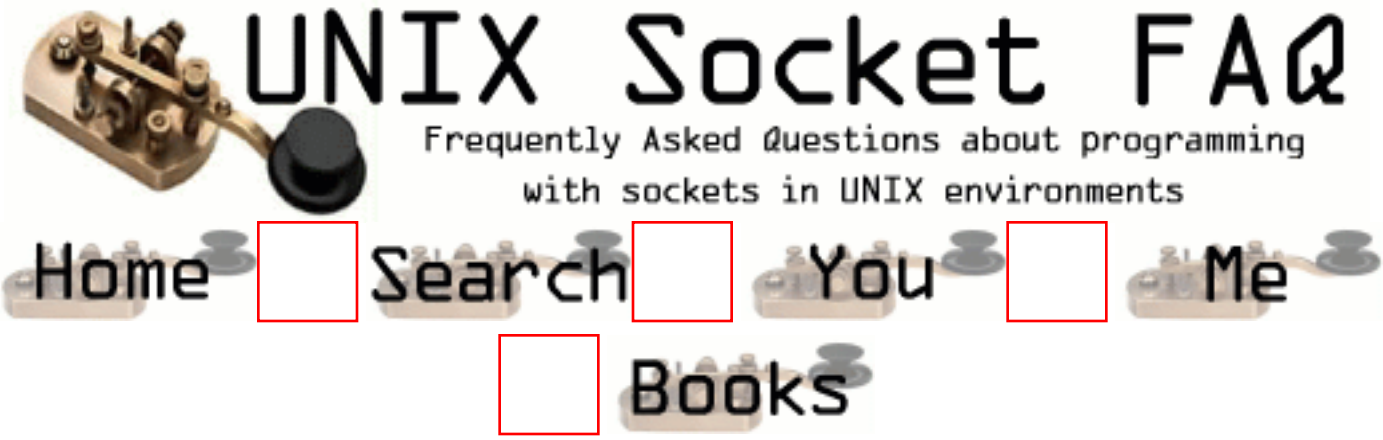
Please post your code.
At least the buffer declaration and the recv call.

From: Arthur Lung

Off the top of my head, if anything ever is giving you a segmentation fault, it's probably in your best interests to make sure that any pointer related parameter is correct.

For example, maybe you're not passing a pointer where you should, or passing a pointer where you shoudln't, or you forgot to malloc some memory, or you forgot to check that the malloc worked, or you're passing in a size that's larger than the actual buffer itself, or you're passing NULL where something valid is required.

Add a Comment



New Questions: select question

From: ornaz

is there a diffrent way to check readabilty on multiply sockets
because after the select wake on read socket
i need to search for this socket
and it is very very heavy loop when there are allot of sockets

thanks in advance
ornaz

From: Rob Seace

Not sure what you mean by "very heavy loop", exactly...
But, just at a guess, are you perhaps just trying to read()
from EVERY socket after select() returns? You don't need
to do that; select() modifies the passed in fd_set, to only
contain the sockets that are readable (or, writable, or have
errors, depending on how you are using it), so you can
simply examine the fd_set, and only read() from those that
are ready to be read... I'm not sure how anything could
possibly be more efficient than that...

From: ornaz

i will explain my self

my applications use i.e: hundreds of file descriptors,
and suffer greatly from the poor scalability of select().
after the select wake up i need to figure out by the
FD_ISSET which descriptor is ready
and in case of hundrds descriptor it is a very "heavy" loop

is there an alternative way to do that ?
or did i miss somthing?

thanks
Orna

From: Rob Seace

I just don't know what you think could be a more efficient method than the select() one... I mean, no matter WHAT you do, you're still going to have all of your FDs to deal with, somehow... Or, are you envisioning an approach that would just return an array/list of only the FDs that are ready, so you wouldn't need to find them within the set of FDs, via looping and calling FD_ISSET()? I know of no such function that works that way... You could look into poll(), but I think its approach will require the same amount of looping work... I just don't think there's much you can do, when working with a huge number of FDs, like that... Being able to multiplex among that many different FDs is going to cause a bit of performance hit, and there's just not a lot you can do about it, really...

Now, if there really did exist a function that fit the API I mentioned, where it only returned a small array/list of the FDs that were ready for processing, that MAY improve efficiency a bit... I say MAY, because in order to have such an API, the array/list it returned would likely need to be dynamically allocated on the fly, so you would have the overhead of dynamic memory allocation/freeing, in place of the extra looping/bitmask-testing... You'd just be shifting the inefficiency into the library function, rather than your code... So, I really don't think there's much of any way to make such a scenerio as efficient as you'd probably like it to be... *shrug*

One method to possibly persue though, which you may or may not already have thought to do: select() returns the total number of FDs that are ready... So, you can use that to reduce your looping, as well... Eg:

```
cnt = select (nfds, &rfds, NULL, NULL, NULL);

for (i = 0; i < nfds && cnt > 0; i++) {
   if (FD_ISSET (i, &rfds)) {
```

```
        cnt--; /* this is the key */
        /* read(), or whatever you need to do, here */
    }
}
```

Then, "cnt" will go to 0 after you read from the last FD in the set, and you'll break out of the loop early... Of course, this won't help you in the case that the very last FD in your set is one of the ones ready... But, probably on average, it'll save you half of your loop iterations, assuming an even distribution of readable FDs in your set...

From: legend hacker

You have discovered the very reason why select is not used. Its sequential, like the programming we did in the 80s. Use threads.

From: Rob Seace

Why select() is not used?? News to me... I use it ALL the time, as do many, MANY others... I think you're engaging in some dreaming/wishful-thinking there, in saying that it's not used anymore... select() is still an extremely useful function, and can be quite efficient if used correctly, and in the right circumstances... Just like any OTHER tool, it shouldn't be used for ALL jobs... However, to suggest that it should NEVER be used is equally foolish...

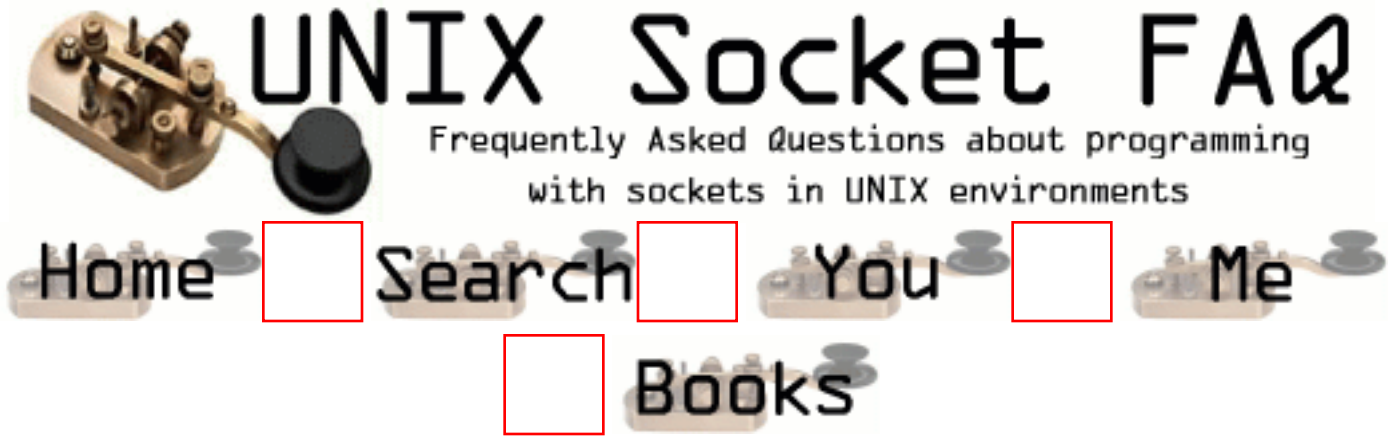
Threads are all well and good, but have several problems and limitations... If you propose to have a single thread for handling each and every FD, you're talking about a LOT of overhead, for the hundreds of FDs being discussed... And, what exactly makes you think your OS is going to time-share between all your threads, such that it turns out you are responding to incoming data more efficiently than you would be by doing a simple search through an fd_set for a set bit? And, even if that were the case, is it worth the overhead incurred by such an approach (extra memory taken up by all the threads, extra processing time eaten up swapping between each thread (depending on your local implementation of threads, anyway), and added code complexity)? In some cases, absolutely, I'm sure... I've certainly used such an approach, where it makes sense to do so... But, it's not the end-all, be-all, one-and-only solution, either... There are many cases where it just doesn't make sense to use, at all...

From: Loco

Rob is right. Threads are useful and very powerful tools to use in some cases, in others they can be a real pain in the ass.
What i understand is that you want a more efficient way than just loop through all the file descriptors checking that each of them is set or not in the structure. I don't see it as a real problem. This kind of loops are very frequent, and as long as the code it executes is "light" then it will be bearable. If you put lots of code inside the loop then you will have a problem.
Usually the effort spent optimizing code is lost in parts that really won't have any impact on the overall performance. So i suggest you try the simple select() model, and see it working, when you have seen it then find the code most repeated and most "expensive" and optimize it. After that continue until you reach your desired performance. I am almost sure the loop you mention won't be of any importance to your overall performance.

:D (HAL)

Add a Comment



New Questions: Single server with multiple clients

From: Stephen Callan

I have a problem with a server/client program that im using at the moment.

When I just use 1 client the program runs properly, and you can send data from the server to the client.

However when you try to use a server to send data to 2 clients on the same port, with the 2 clients on different locations, you get a connection refused message on both client stations.

Why might this happen...

From: Btvrao

.

Sir, i think that in ur server program whenever connection comes u didn't fork the loop . if u fork the loop that child will take care of new connection.

this is my with level best.

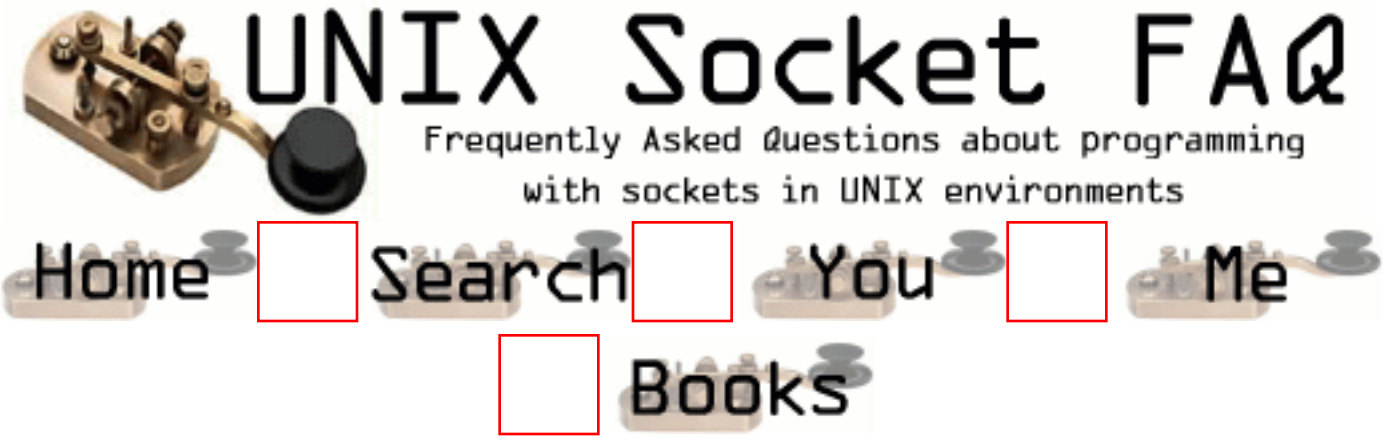
From: Btvrao

.

Sir, i think that in ur server program whenever connection comes u didn't fork the loop . if u fork the loop that child will take care of new connection.

this is my with level best.

Add a Comment



New Questions: How to write socket programming between UNIX & Windows platform

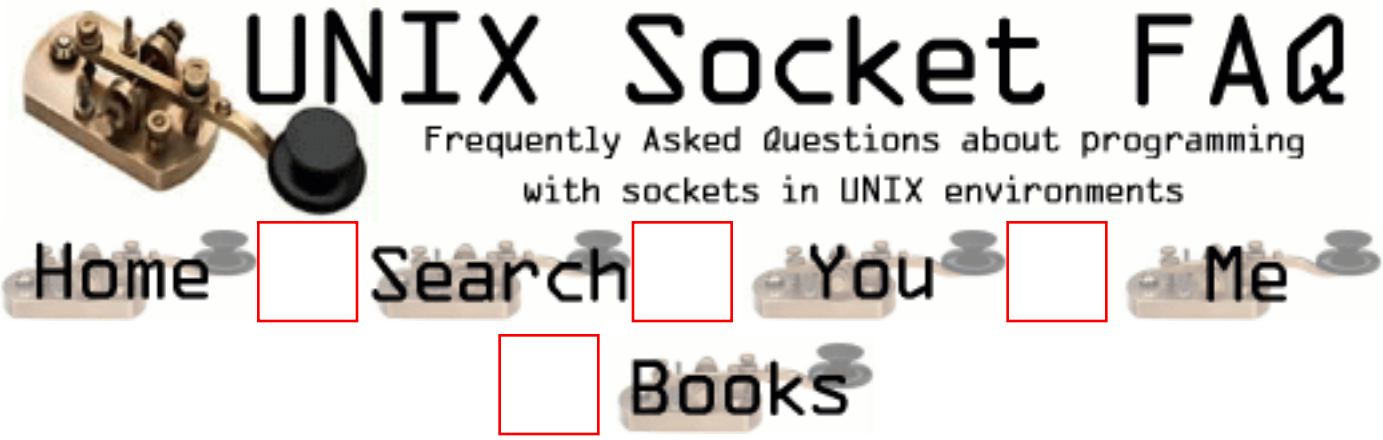
From: Mohd Sanie

I want write a socket programming between UNIX and Windows (95/98/NT) using C on UNIX and Visual Basic or Visual C++ on Windows. How could I integrate this two since they are in two different platform? In Windows it is most likely to use winsock. Can anyone help me on this or suggest any website to refer to? Thank.

From: Todd Stout

The winsock API is modeled after the BSD socket API that
appeared many years ago on UNIX platforms. The MS
winsock API is not 100% BSD but it's close. There
are a few major differences such as error codes and
the fact that Windows requires an explicit initialization
of the TCP/IP stack (it's not part of the kernel) However,
I have found that it is relativiely easy to write C/C++
code that compiles and runs on UNIX and Windows without
an excessive amount of #ifdefs in the code.
One drawback to this however is that most unices are
moving away from the BSD API in favor of the POSIX API.
The POSIX api is similar to BSD but different enough
to require code changes. I have no idea if MS is planning
on supporting a POSIX socket API in the future. Since
open standards are against Microsoft's business plan,
I'd doubt they will ever support it.

Add a Comment

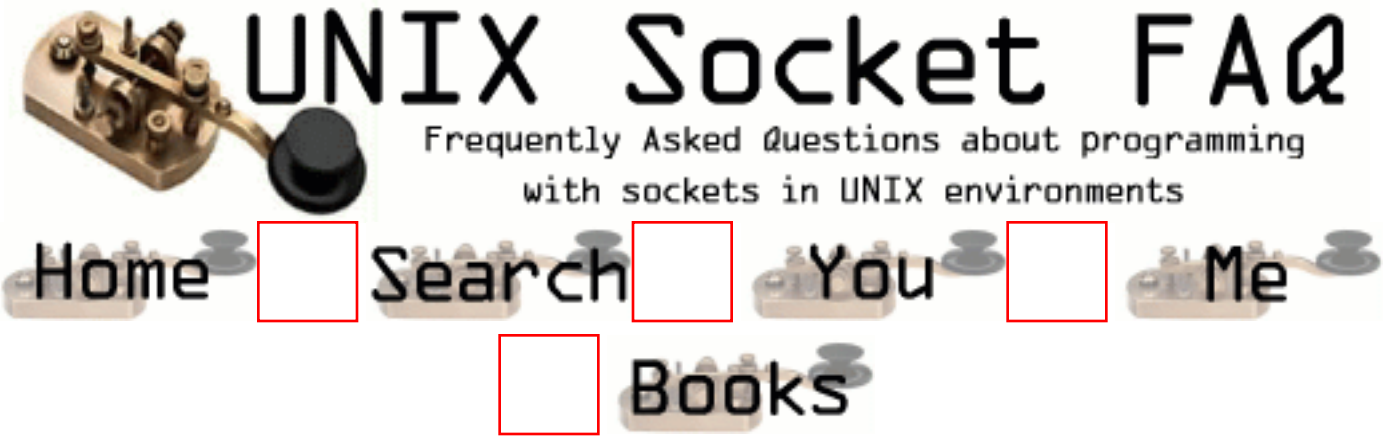


New Questions: CONNECTION ABORT

From: Erin

Periodically, when a client connects to my server process, I'll get Connection Aborted errors on my server. Is this a "catch-all" for many different things that could go wrong - or does anyone know of (even a small number) of reasons why this would occur?

Add a Comment



New Questions: MAC Address from client w/o using ARP?

From: supafuz

Hey, im looking for some help with this issue im having ...
w/o reinventing the wheel.
I would like to find the MAC address of clients which sends
a UDP packet to my server. I see two ways to do it:
1) use ioctrl(sockfd, SIOCGARP, &pointer)
2) sniff at the MAC layer and wait for something destined to me

the first 1) screws me up everytime Linux flushes out its arp
cache by returning an error. I have the client send out UDP
packets every 200-2000 msecs to this server. Im receiving several (1000s) of
packets and about every 5-8 minutes the arp tables are flushed.
The next packet received will not have an arp entry, even though
it is still receiving packets. I guess the arp flush is another
issue? Extending the duration of cache flushes is not an option
since this program will be run constantly. While in use several
different computers will be sending info ... not just the one
im testing. (for those wandering why i need to get the source MAC
from EACH packet)

number 2) seems like too much work for the kernel/CPU to do for
what I need. But, it may be the only solution?

I was hoping for some struct similar to sockaddr_ll where
the sender's MAC info could be placed on arrival of each
packet. Im using this struct, sockaddr_ll, to send frames
but am unable to use it to store Received frames.

Im at a stand still right now. Ive look in Richard Steven's UNIX Network Programing w/o too much luck for this prob. (He helps so much in other issues, though!)

So, what do you thinK?

-imallears

From: Rob Seace

What is the errno set to when the ioctl() fails? Have you tried simply retrying the ioctl() after the failure? Maybe it's simply a temporary/transient failure, of some sort...

The only other choice I can think of would indeed be to use raw/packet sockets, to get at the link layer... A bit of a pain, certainly...

Add a Comment

New Questions: Threads and sockets

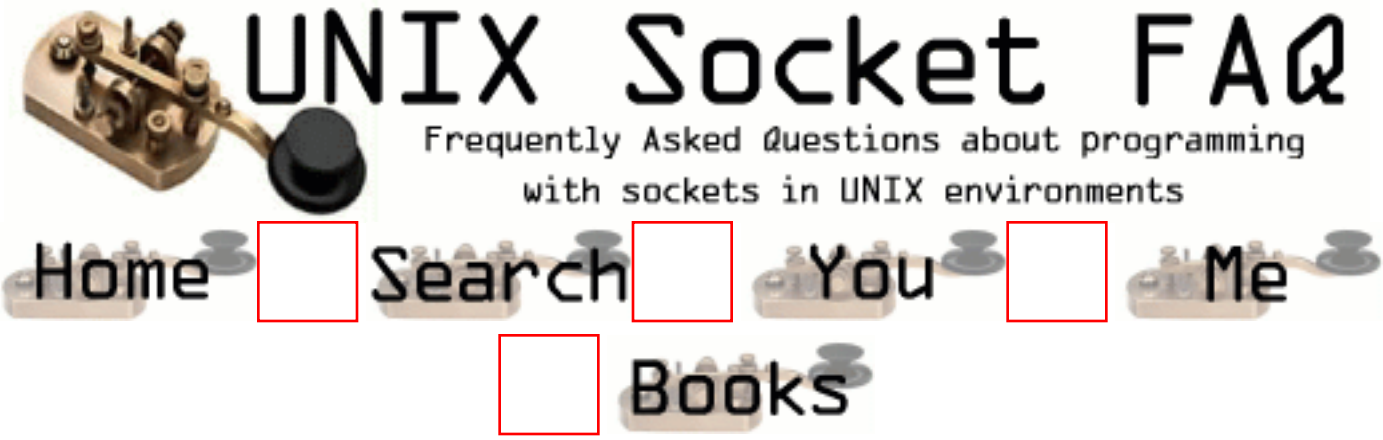
From: John Arcade

I'm writing a sliding window protocol using UDP, and the problem I have is I cannot recvfrom the client. I have a thread that sends the data to the client, and another thread that is supposed to constantly listen to the client. The code below is the listening thread.

```
void *ThreadListen(void *threadArgs)
{
        int sock, size, rsize,s;
        struct sockaddr_in fromAddr;
        Msg rcvb;
        SwpState *rstate;
        fd_set sset;

        size=sizeof(fromAddr);
        rsize=1;
        sock=((struct ThreadArgs *) threadArgs) -> clntSock;
    rstate=((struct ThreadArgs *) threadArgs) -> lswps;
    free(threadArgs);
printf("\n in Thread Listen... recvfrom");

        while (rsize !=0)
 {
            FD_SET(sock,&sset);
            printf("\nin while loop\n");
            if(select(sock+1, &sset, NULL, NULL, NULL)==0)
                ErrHandle("listen select err ");
            else
            {
                if ((recvfrom(sock,rcvb.data, PKT_SIZE, 0,
                        (struct sockaddr *) &fromAddr, &size)) < 0 )
                {
                        if (errno != EWOULDBLOCK)
                                ErrHandle("\n ERR:  could not handle recv");
                }
                else
                {
                        rsize=recvSWP(rstate,&rcvb);
                        printf("\n rsize=%d\n",rsize);
```

```
                }
            }
printf("\n TL:  finished recvfrom");
        }
        return (NULL);
}
```

Add a Comment



New Questions: write() Hanging Program.

From: Twilight

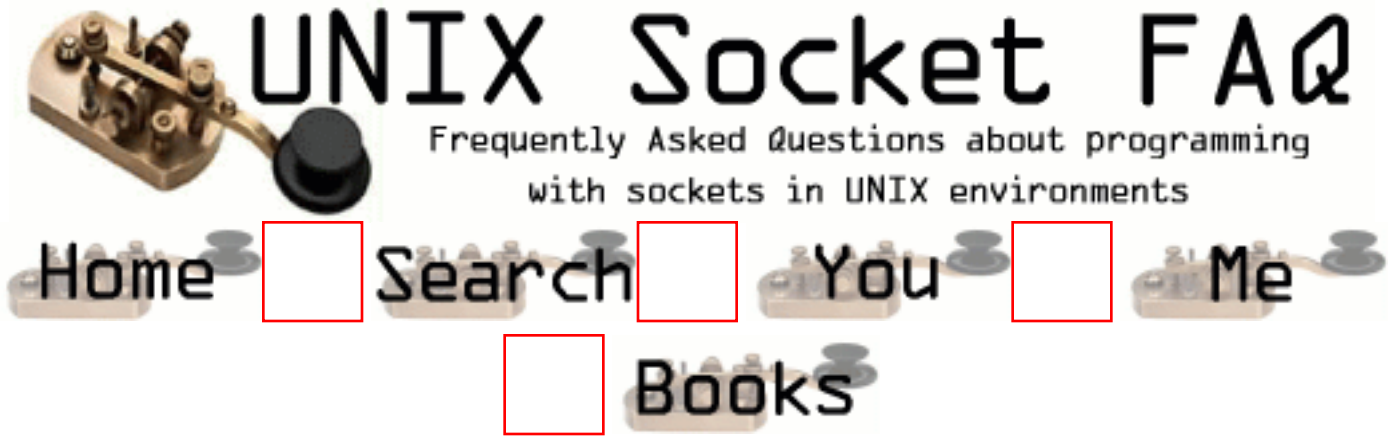
I am using write() to send data back to sockets. The problem is that every few hours the server hangs on the write() command. Should I use some other command? or check to make sure the fd is still active some how?

Any feedback would be great.

From: sielim

This happens... Write (in BLOCKING MODE) waits until there's
some free space in local data buffers. And there might not be
because of the second party is not able to receive data and buffers
are filled up (somebody has kicked off the network plug, for exaple...).
Try to use NON BLOCKING MODE.

Add a Comment

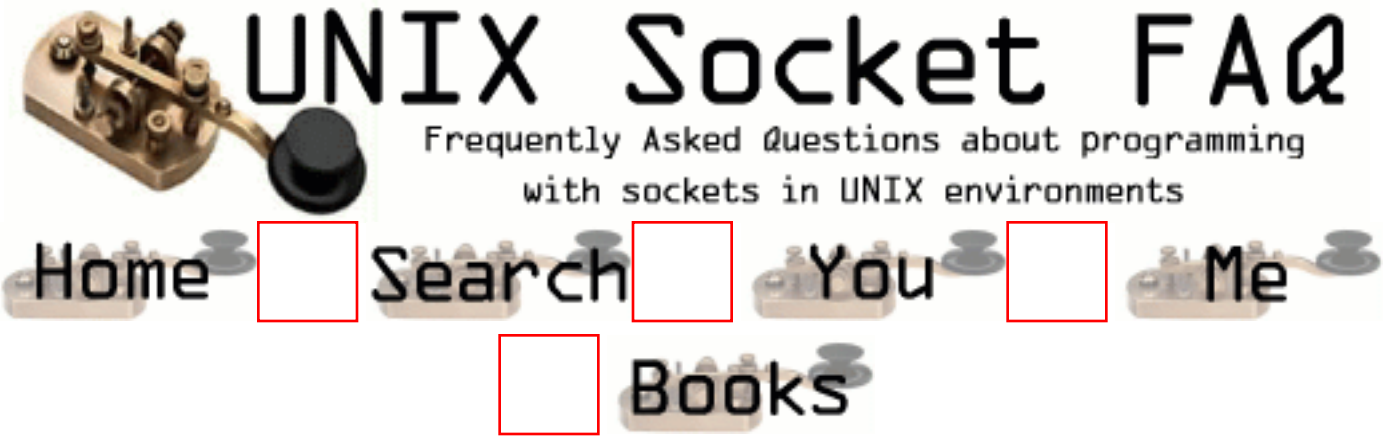


New Questions: KCP

From: K.Kumar

In which site can I find the detailed documentation of KCP(Kiva Communication Protocol)?
Thanks in advance.

Add a Comment



New Questions: TCP Send-Q Retransmission

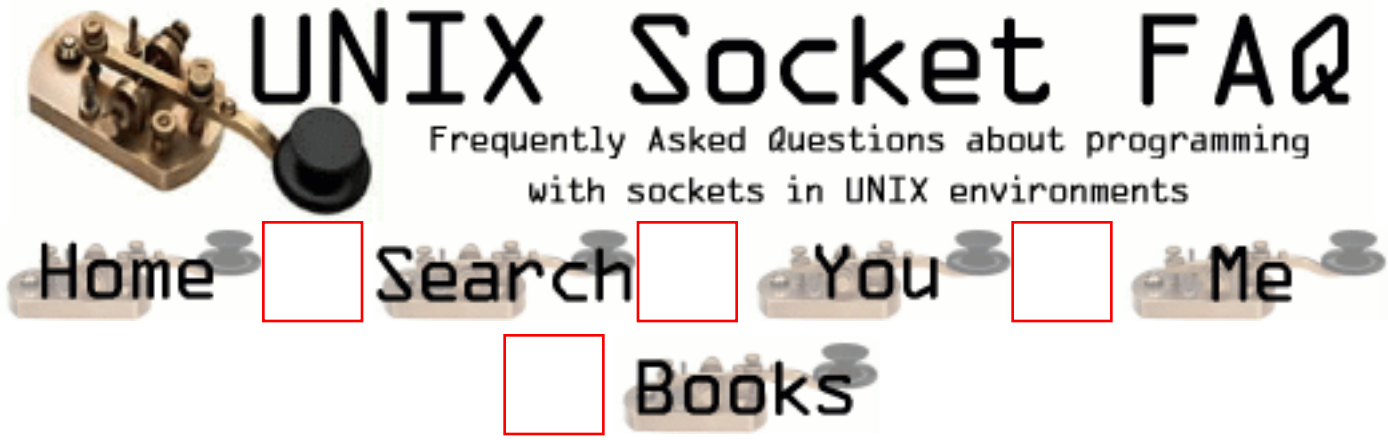
From: Claes

Hi, I have a problem with a Linux game server. The clients just don't get the TCP Send-Q Retransmissions. Is there any way to improve the performance of the retransmission of not Ack'ed tiny datagrams?

From: Rob Seace

Not sure if this is exactly what you're looking for, but it sounds similar: try disabling the Nagle algorithm... Though, that is really related to delays with small packets initially, not retransmission of them... But, it still may be of use, if you're not already using it... But, it may just be that for a game server, you want to be using UDP rather than TCP... (Just like Quake...) Then, you don't need to worry about the vagaries of the TCP stack and how it retransmits and other such things... You can just code your own retransmission behavior as you like... (If you need any at all... You could just do like Quake does, and EXPECT packets to be dropped, and not consider that a problem... Just let them drop, and move on... It requires a protocol that can recover from such a situation though... Ie: one that is able to catch up on any missed info, with later packets that DO make it though...)

Add a Comment



New Questions: How can I do a ftp???

From: Sofy and Miriam

Hello friends!!!

Somebody can help us, how can to do a socket that works
like a FTP,I mean the information that I send to the server
have to be in a file. Our teacher told us that we can to use
the functions get and put.
We know that this question is so simple for us we are principiant
in perl!!
Or do you know some reference about sockets that we can read???

Thanks a lot!!!

From: Luca

I'd like to know how to do too.(in C language).

From: TRC

SOCKETS in C
www.ecst.csuchico.edu/~beej/guide/net

From: Hector Lasso

Hi,

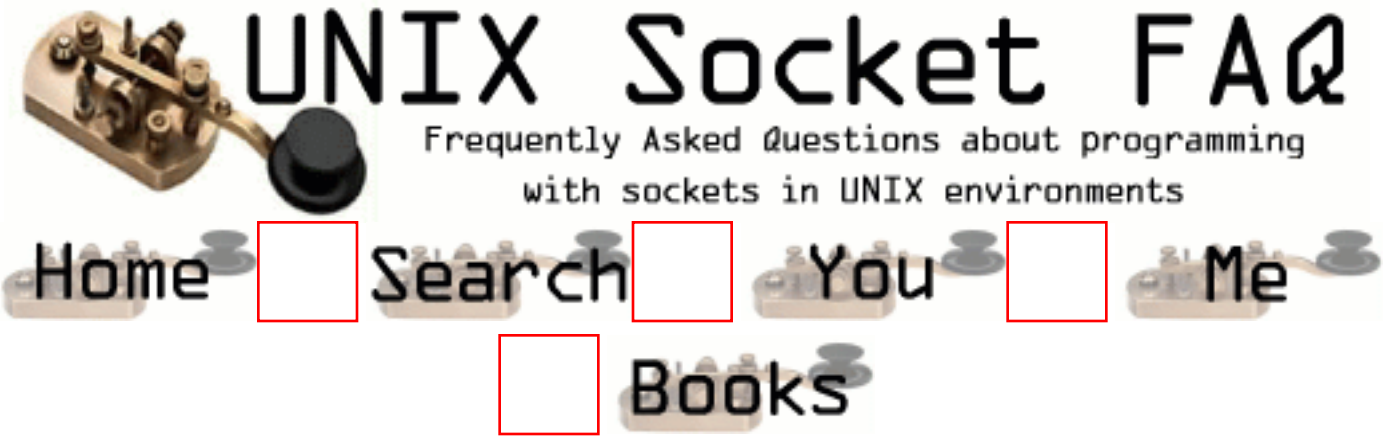
This is not a simple matter. FTP is a protocol, its specification is written in RFC 959 (which you can find easily on the web - ftp://nic.merit.edu/documents/rfc/rfc0959.txt).

It uses TCP connections from client to server to transfer files and do some other operations

(such as listing files).
When the client connects to the server it creates a TCP connection that is used to authenticate the user, to send commands to server and to receive status notifications about the commands. This connection is called the control connection. However, when data is transfered a temporary connection must be used (the data connection), which is created from the server to the client, or from the client to the server. This temporary connection is used to send data and need not exist throughout the session (i mean, it can be closed when there is no data going from one side to the other...)

I hope this helps...

Add a Comment



New Questions: reading from buffer

From: larry

How can data(char datatype) passed on from client program to the server through sockets be retrieved?

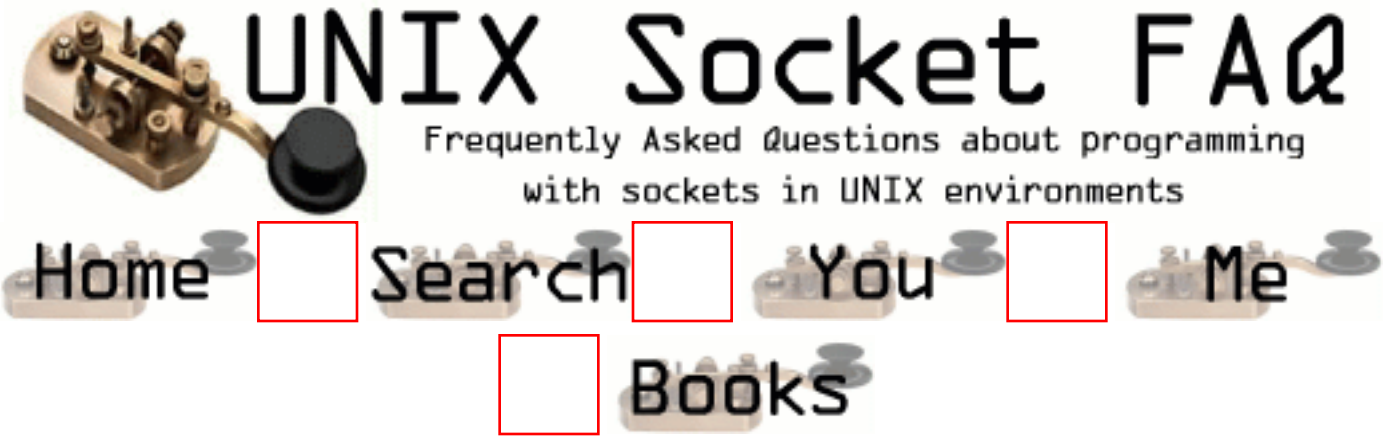
From: Andreas Forsgren

send() and write().

From: suresh

how could i read text from a buffer

Add a Comment



New Questions: How to retrieve data sent by client to server when data sent is stored in a buffer?

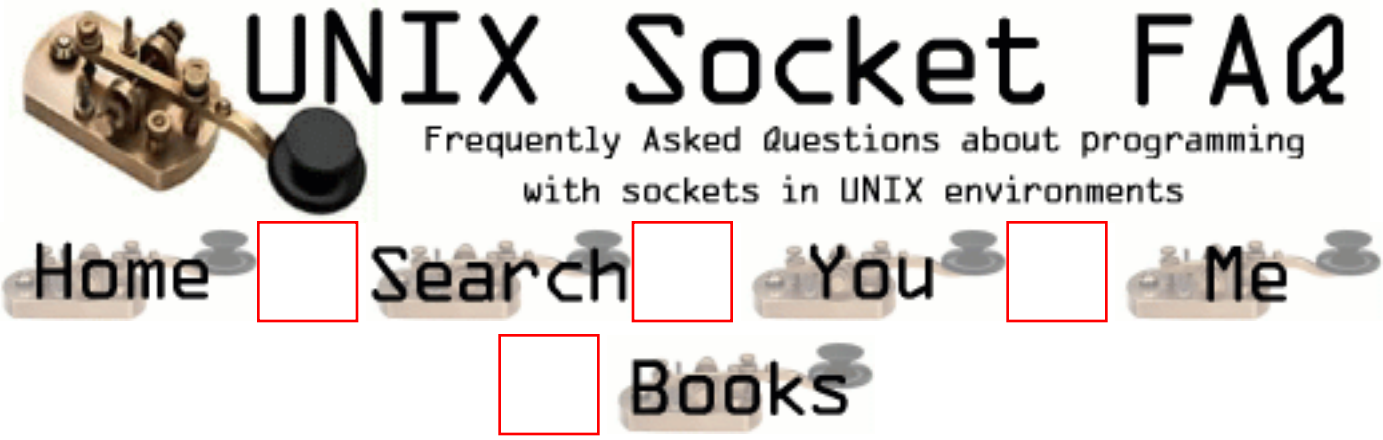
From: Lara

hi

I've written a client program which sends data, input by the user of type char[],the server basically gets data sent by client and prints it at the output which works fine.My problem is how do i sort out the data sent by the client.
The buffer where in i get data from client contains a series of values,eg:date,name,folder,string value etc..
Kindly help

regards

Add a Comment



New Questions: broken pipe?

From: irene

hi

Why do i get the run time error 'broken pipe' when i run my client and server programs

---

From: Rob Seace

See questions 2.22, and 2.19...

Basically, it's normal behavior which you should expect in any sockets program... I always ignore SIGPIPE, and then just check for an error return from read()/write() (it'll set errno to EPIPE)...

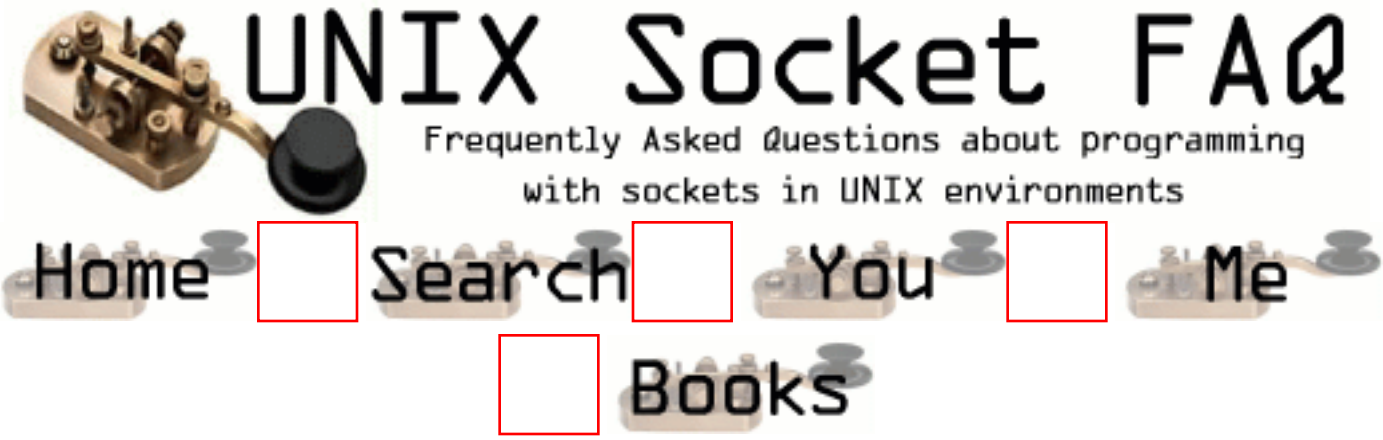
From: Muslim Moiz Ghadiali

It basically means your connection between the server and client sockets is been terminated for some reason.Happens mostly in programs running in continous loops at both ends.

From: Rajesh

When the server to which the client wants to connect is not running, this happens. run the client only, yule get broken pipe.

Add a Comment



New Questions: about recv() call

From: dana

I am trying to created a connection oriented socket, & I find my accept call is succeeding & returning a valid socket descriptor, but my recv() call is returning 0 when it receives a packet. I dont know what is happening. could anyone help.
thankx
dana

From: Magnus Boden

Post your code here.

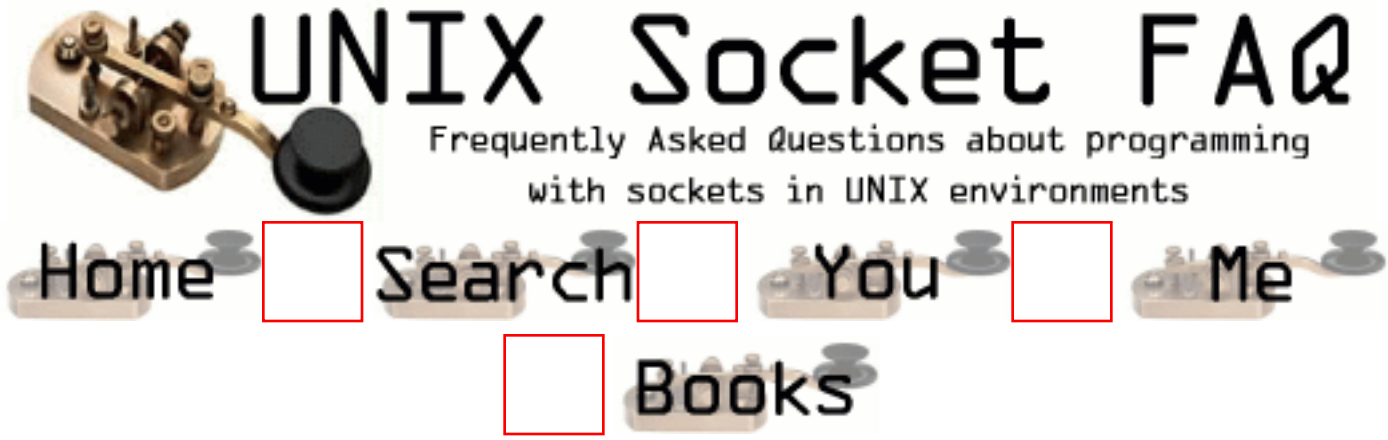
From: Sebastien Metrot

We have the exact same problem under win32 on certain configuration only. (win2K AND win98...)
Can anyone help?

From: Topo

I´m having the exact same problem!
Please somebody answer...

Add a Comment



New Questions: binary file prob

From: supafuz

Here is what I want to do:

Send a Binary file to an awaiting UDP socket

My problem:

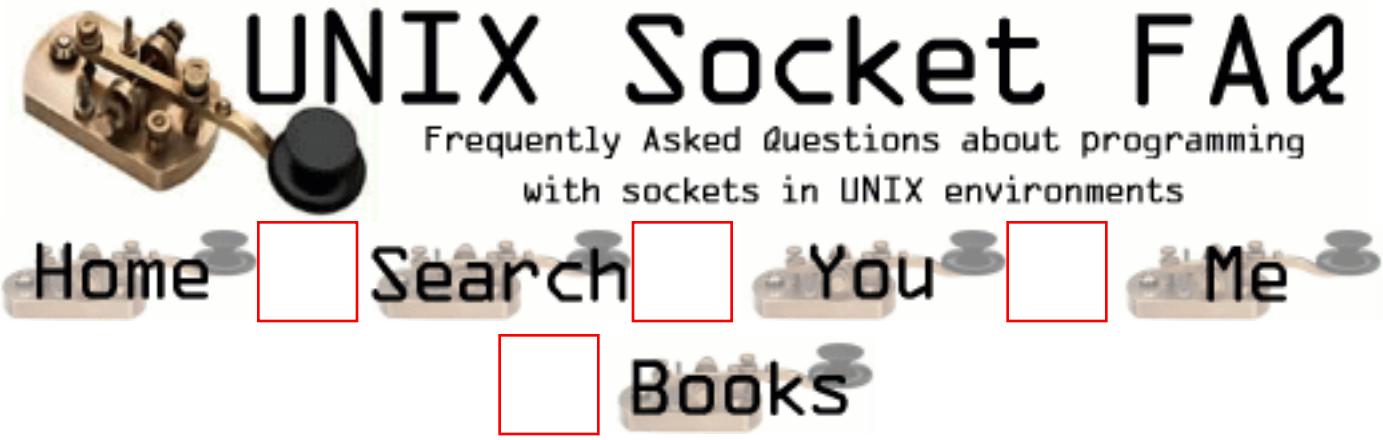
creating a PROPER file descriptor to the source file

I have:
```
length = fileLength; // this is done in a separate function
                     // with fseek ...
fp = fopen(file,"rb");
file_stuff = malloc(length);
fread(file_stuff, length, 1, fp);
...
bind() ...
...
rc = sendto(sd, filechunk, sizeof(filechunk), 0,
(struct sockaddr *) &remoteServAddr,
sizeof(remoteServAddr));
```

Im using gcc with linux RH 6.1
Everything works fine with ASCII files.
The first 0x00 encountered terminates sendto().
When I try to compile the same code with gcc on Winnt
using cygwin (www.cygnus.com), which provides a Unix
working environment, both ASCII and BINARY files work fine.

Im really confused and have been stuck on this for two weeks.
You know anything about this?

Add a Comment



New Questions: how to signal select function

From: ornaz

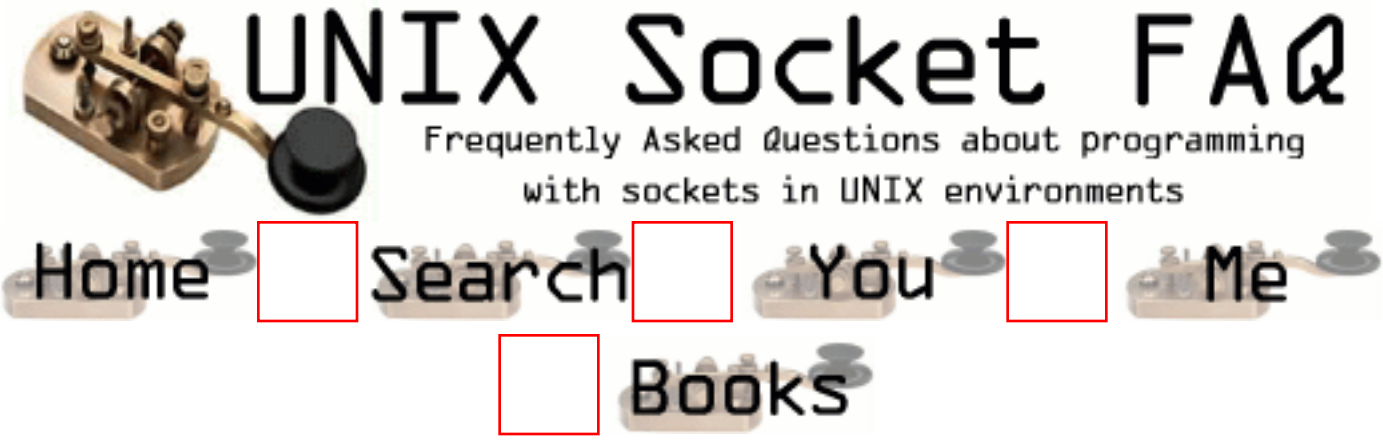
i have 2 threads:
1. connect_thread that responsable to reconnect close sockets.
2. read_thread that responsable to do select on multi active
sockets and do recv.
i have the following problem:
the read_thread is waiting on select on certain sockets
but meanwhile there are new active sockets
so how do i signal the read_thread - the select to return
for adding the new sockets?

waiting for reply,
thanks in advance
Orna

From: Nospam

Create a pipe and put the read end of the pipe in the
read set of the select. Writes to the write end of the
pipe will wake up the select. If the pipe fd is readable,
then you know to check for whatever internal condition you may have. This is generally how multi-threaded (or multi-processed) applications signal each other when one is blocked in select.

Add a Comment



New Questions: Java linux socket problem

From: sagi

I'm trying to open 4000 sockets. It seems to work fine
but right about socket number 3950 or so I get the message
" errno 11 ,Resource temporarily unavailable to fd NNNN "
(NNNN seems to be the fd number). I've increased the number of
filedescriptors in the right place so I'm sure it doesn't
concern that (plus the error message can appear when
NNNN=3976 or NNNN=3870). I ran the examination on linux
2.2.14 with IBM's java 2.
Thanks.

From: tomo

I had the same problem.
I think your system is running out of sockets.
Every time a ServerSocket accepts a socket
a new socket is created. Once you close this socket,
however, it does not get closed immediately but stays
in the TIME_WAIT mode for some time (you can check this
using netstat) In c you can set SO_REUSEADDR option
in order to disable this but I haven't found a way
of doing this in java

Hope this clears it up a bit.

Tom

Add a Comment

UNIX Socket FAQ

Frequently Asked Questions about programming with sockets in UNIX environments

New Questions: Web Hostinb

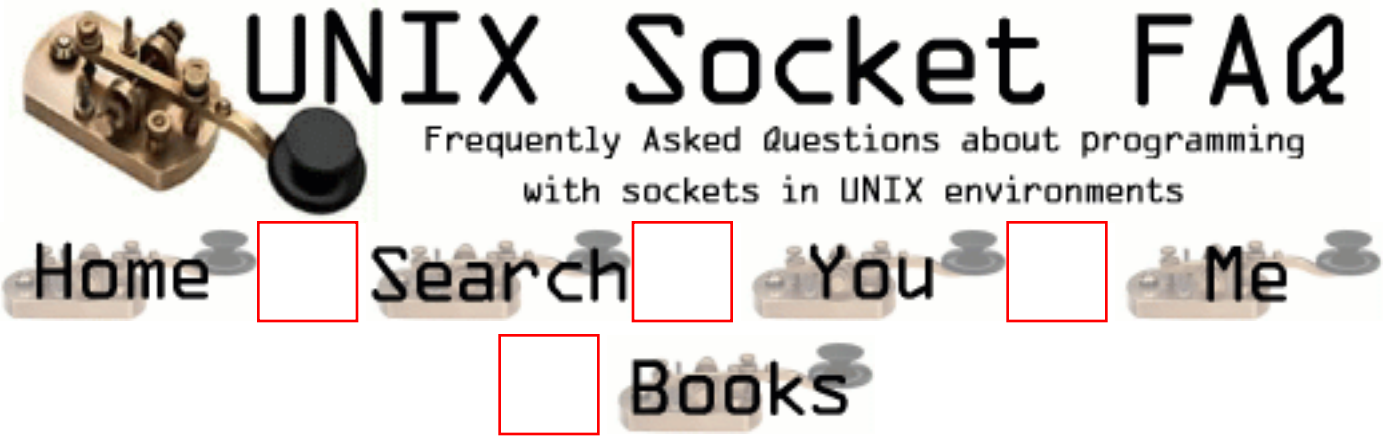
From: Bombino

where can I get the source code for a web-hosting application server/client) in C/Unix?

From: Hector Lasso

Download apache source code. www.apache.org

Add a Comment



New Questions: Reg send() socket call

From: dana

I have established a connection between the server & client.
I am sending packets from server using a send() call to client. Is there a way to stop server from sending packets when i stop the client from running.
cud anyone give me hints???

-dana

From: Hector Lasso

If the client process is terminated, the server will detect that the socket was closed the next time you use send() and when you use send() again it will generate a SIGPIPE signal that by default will terminate the sending process unless you handle the signal or ignore it. If you ignore the signal (by using sigaction() or alike), the send() function will return -1 and errno will be EPIPE. You can also get the same behaviour if you use the MSG_NOSIGNAL value in the flag parameter of send().
However, detection of disconnection is not assured, and the documentation clearly states that "No indication of failure to deliver is implicit in a send. Locally detected errors are indicated by a return value of -1." (man 2 send)

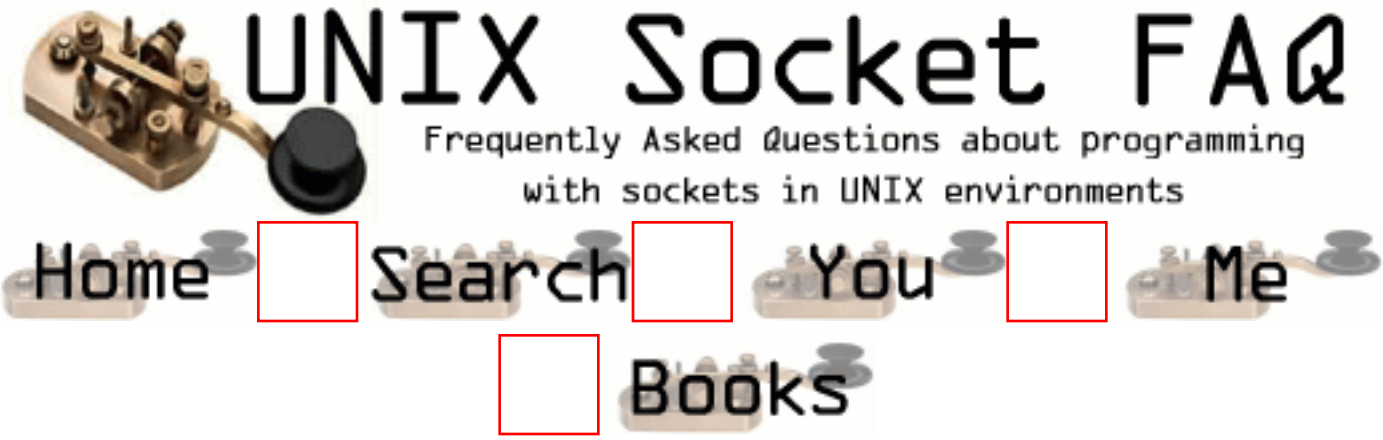
I think you should use some kind of acknowledgement or message from the client to let know that it received the last message, or that it won't be able to receive further messages.

Good luck

From: dana

Thanks for response. I could really find a solution to my problem.

Add a Comment



New Questions: exit parameter when send call fails

From: dana

When my send call fails, I want to terminate my process by calling exit call.What parameter should i pass for exit either exit(0) or exit(1)?

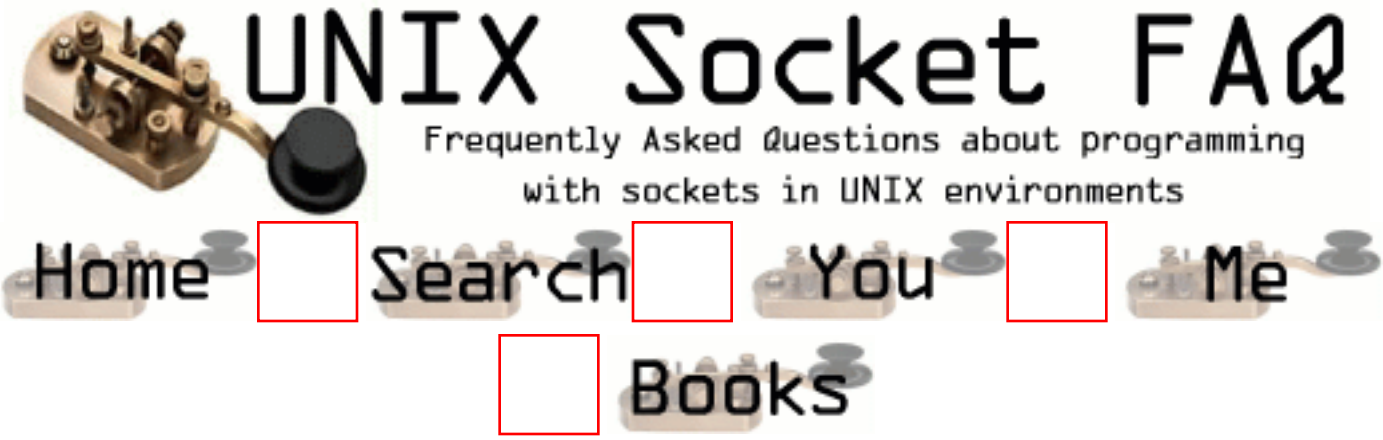
From: Hector Lasso

You can use whatever value you want.
The value you use will be returned to the parent process which may use for whatever it wants to, or not use it at all...
The parent can retrieve the exit status of the child by using the wait() or waitpid() functions.
If the parent process is a shell, and the process runs in foreground, the exit status will be stored in the $? especial variable.
For example, to see your process exit status yous could write something like in the shell:
your_process_name_here; echo "This is the exit status: $?"
I usually use 0 as a normal termination exit status, and some other value to indicate the program finished because of a situation out of control.

From: Hector Lasso

You can use whatever value you want.
The value you use will be returned to the parent process which may use for whatever it wants to, or not use it at all...
The parent can retrieve the exit status of the child by using the wait() or waitpid() functions.
If the parent process is a shell, and the process runs in foreground, the exit status will be stored in the $? especial variable.
For example, to see your process exit status yous could write something like in the shell:
your_process_name_here; echo "This is the exit status: $?"
I usually use 0 as a normal termination exit status, and some other value to indicate the program finished because of a situation out of control.

Add a Comment



New Questions: Using recv function

From: Gabi Adler

I'm looking for an example of how to use the "recv" function. Specifically, I'm trying to use it to enable my code to deal with interrupts, and retry a system call that is interrupted:

```
do
{
   sysret = recv(...);
} while (sysret < 0 && errno == EINTR);
```

My question is: what parameters do I fill in where the suspension marks are? Does anyone have a specific example of recv usage in this context?

Any help would greatly appreciated.

--Gabi

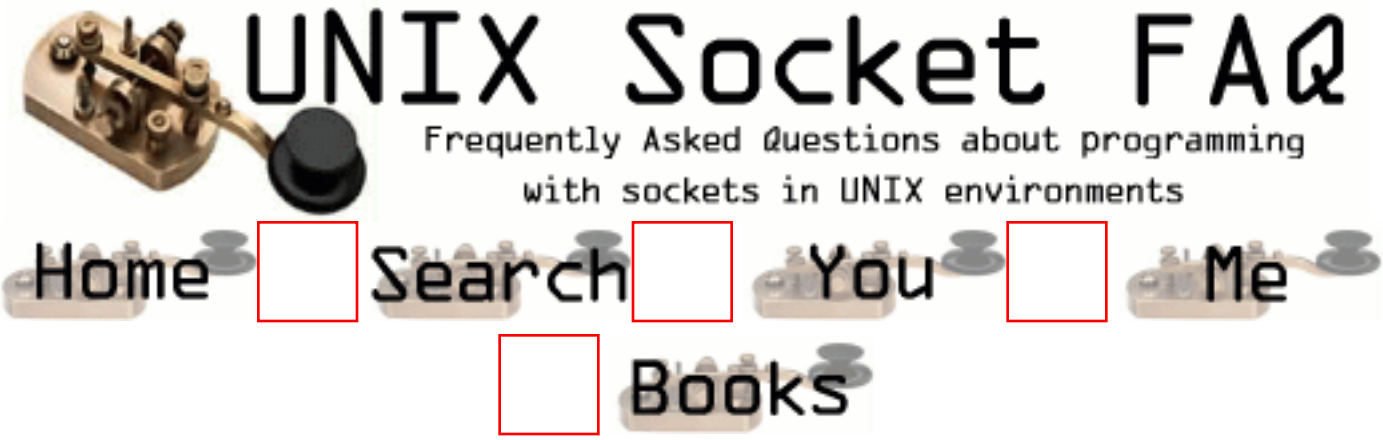
From: Rob Seace

Well, the basic prototype of recv() is:

int recv(int s, void *buf, size_t len, int flags);

Where, "s" is your socket FD, "buf" is your buffer to read
into, "len" is the max length to read, and "flags" can have
several different values, but is generally just 0... But,
possible other choices are "MSG_OOB", "MSG_PEEK", etc...
See your local man page for full details...

Add a Comment



New Questions: CLOSE_WAIT

From: Prabin

How do i free the port that is in CLOSE_WAIT status?

Thank you.

From: Mike Temelkovski

Anybody got the answer for that, except rebooting the machine?

From: Michael H. Ward

If you look at the port via netstat, look to see if there anything in Recv-Q for the port. If so, that means there is some leftover data waiting at the port to be read (even if you have already shutdown the port). A final read() to the filedescriptor of the port should clear it.

From: Jacob Harris

Hello,

Here's a question. Suppose I have a server whose sockets are set to DONTLINGER. I have a bunch of clients who are still set to linger. Are there any situations where the server would close sockets but get stuck in CLOSE_WAIT? Is it expecting more data from the client in that case (despite the DONTLINGER), and is the way to fix this to make the clients DONTLINGER, or is it something else? I should also mention the server is NT-based. Anybody seen this before?

Thanks,
Jake

From: Juan Forero

Isn´t there a way to set a time out to close sockets in CLOSE_WAIT state?

Thx

Add a Comment

New Questions: CCGI - C socket communication

From: subrahmanyam

hi friends

Client is C socket
server is C - CGI program

how to send data from C socket to C-CGI program
Is there any interface required if required please send me some piece of code.

Along with this How i can reflect environment variables values while sending data to cgi program.
every thimg should be in C language
If any person knows please guide me

From: Rex

I'd also like to see how this is done.

From: Hector Lasso

Hi,

I assume your CGI is correctly installed and configured in the web server software.
To communicate the server and client you must use the HTTP protocol from the client, especially the methods GET and PUT.
The server will get the information from the client using its stdin, and will output results or whatever to stdout.

The client will have to connect to the web server (usually port 80), send a POST or GET command with the appropriate URL, and the information it wants to send to the CGI program

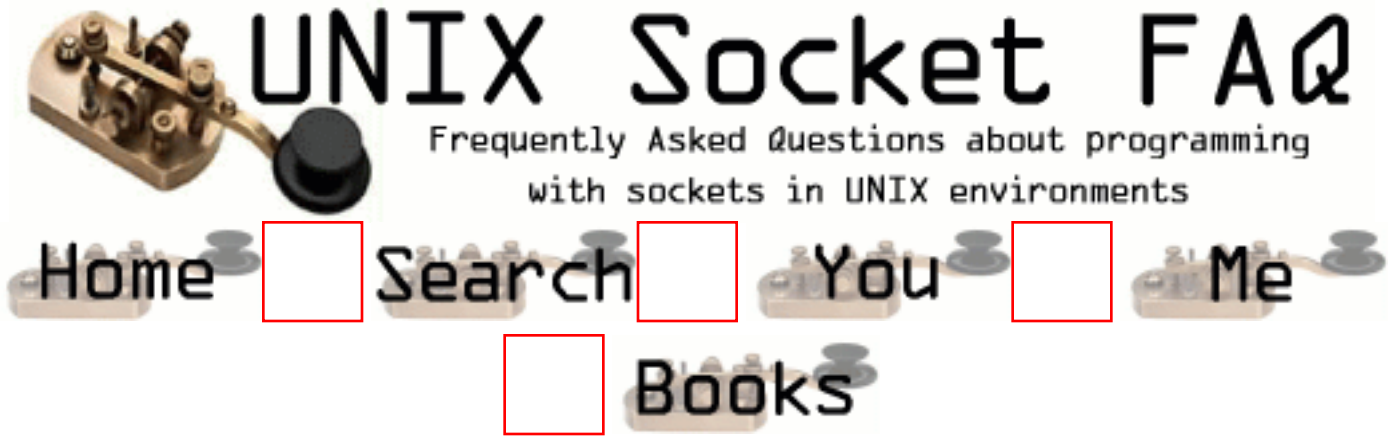
Example:
// You have already conected to port 80
// The CGI is in /cgi-bin and is named my_cgi
// I will use the POST method
send(my_socket, "POST /cgi-bin/my_cgi\n", 22, 0);
send(my_socket, "Content-type: text/plain\n", 26,0);
// You will send n bytes, so tell the server so
n = strlen(data_buffer);
sprintf(some_buffer, "Content-length: %d\n", n);
send(my_socket, some_buffer, strlen(some_buffer),0);
send(my_socket, data_buffer, n, 0);

// Receive the response from the web server
recv(my_socket, buffer, m, 0);
// ...

You will have to study HTTP a little bit more to make this code work, but it isn't much problem. It's been a lot of time since i wrote some code for this, so i hope i haven't missed something.

Good luck

Add a Comment



New Questions: setsockopt with IP_MULTICAST_IF question

From: Antonio Vilei

Hi there,
In Linux I need to choose a sending interface (192.168.0.1) from 2 interfaces when sending to a multicast group.
I tried setsockopt() with IP_MULTICAST_IF but it failed saying that address cannot be assigned. It fails even if I try to bind the socket before calling setsockopt(), but I get the same error.
Here is the code:

```
   int sockfd;
   struct sockaddr_in their_addr; // multicast receivers' group information
   struct in_addr interface_addr;

   if ((sockfd = socket(AF_INET, SOCK_DGRAM, 0)) < 0)
      {
      perror("cannot create socket");
      exit(1);
      }

   interface_addr.s_addr = inet_addr("192.168.0.1");
   if (setsockopt (socket, IPPROTO_IP, IP_MULTICAST_IF, &interface_addr,
sizeof(interface_addr)) < 0)
      {
       perror("setsockopt failed");
       exit(1);
       }
```

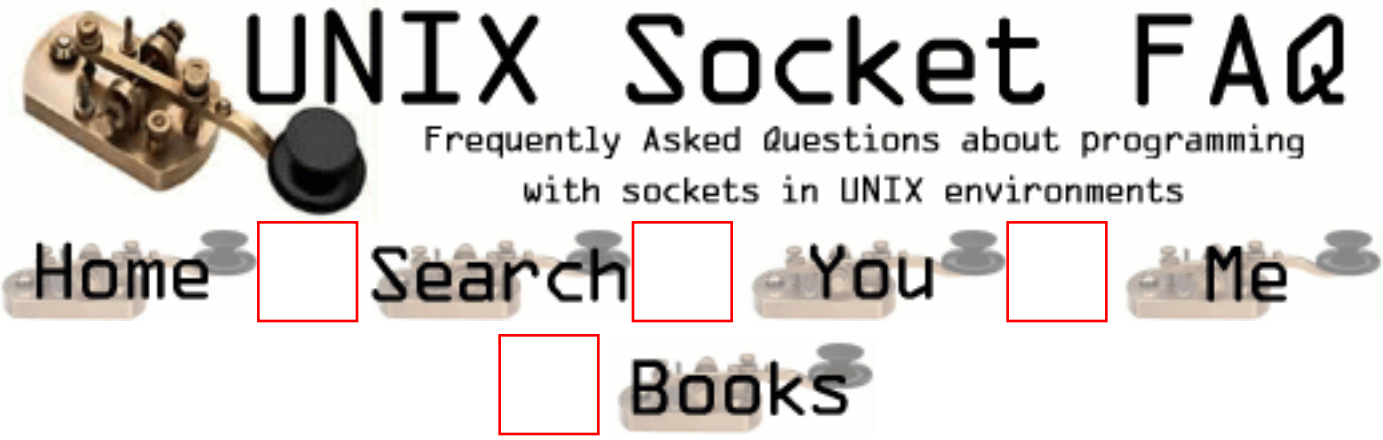
Anyone knows what to do?

Thanks in advance.
Antonio Vilei

From: Rob Seace

I don't know much about multicast stuff, however looking
at "man 7 ip", the info about the IP_MULTICAST_IF socket
option seems to indicate that it needs a "struct ip_mreqn"
as an arg, not a "struct in_addr" like you're passing it...
Perhaps that's the problem?

Add a Comment



New Questions: Socket is not being released

From: jagadeesh

Hi,
I've a Java tcp client, it is conneted to java server which is running on solaries. the socket is not being released soon after the network cable is pulled out. Any one can answer me how can i re-establish the connection immediately in the same java tcp client to the server.

Thanks
Jagadeesh

Add a Comment

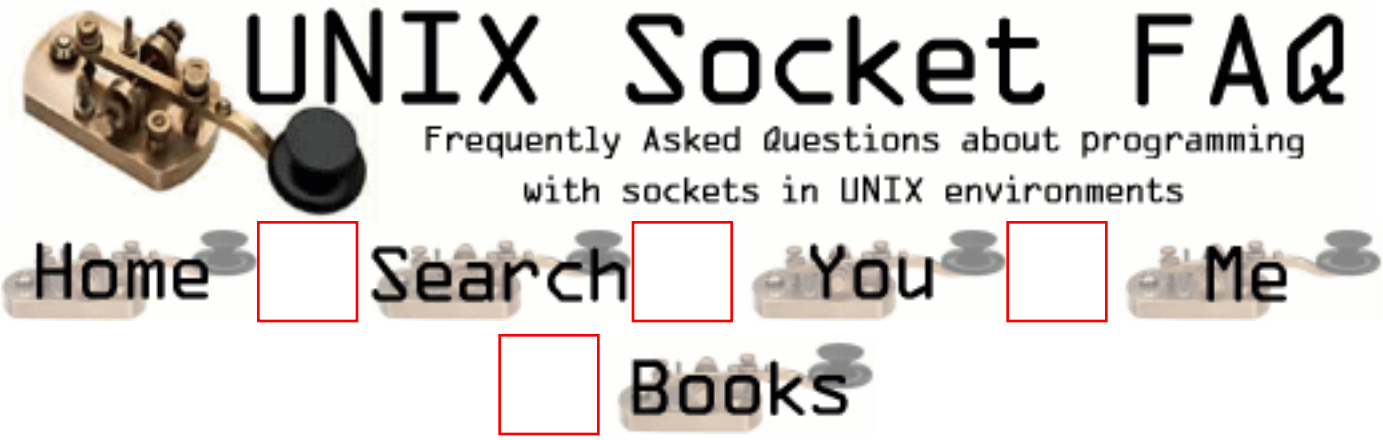
New Questions: Can I bind socket descriptor to a port which bound by others ? How ?

From: Jack

From: Rob Seace

Well, it depends on your OS... Some allow you to steal a bound address away from a daemon listening on INADDR_ANY, by instead binding to a specific IP, for the same port... Some also support SO_REUSEPORT, which will do it, as long as all programs use it... (See question 4.11 in the main FAQ for more info on that...) In general though, I sure wouldn't think you would really WANT to do such a thing...

Add a Comment



New Questions: How Many Sockets?

From: JiK

How can you tell how many sockets your system will allow you to have open concurrently?

From: Rob Seace

Easiest way would be to just write a program that keeps
trying to open as many as it can, until it fails...

Short of that, the best way is to check your kernel source
(if you have access to it)...

Short of that, you can call getrlimit()... Or, as a rough
guess, check the value of FD_SETSIZE on your system... That
specifies the max number of FDs you can select() on, so it
should be a safe absolute max... (Though, many systems let
you redefine its value, and thereby increase the number of
select()'able FDs... Meaning, the actual max # of possible
FDs you can open MAY be higher... Though, you generally
have to go through magic to increase that, too...)

From: Amit

if you write a program to open sockets it will fail at the file descriptor limit for a process and not the operating system.

Is there a way to set the file descriptor limit to 'unlimited'?

the command: "limit descriptors unlimited" sets it to 1024

From: Rob Seace

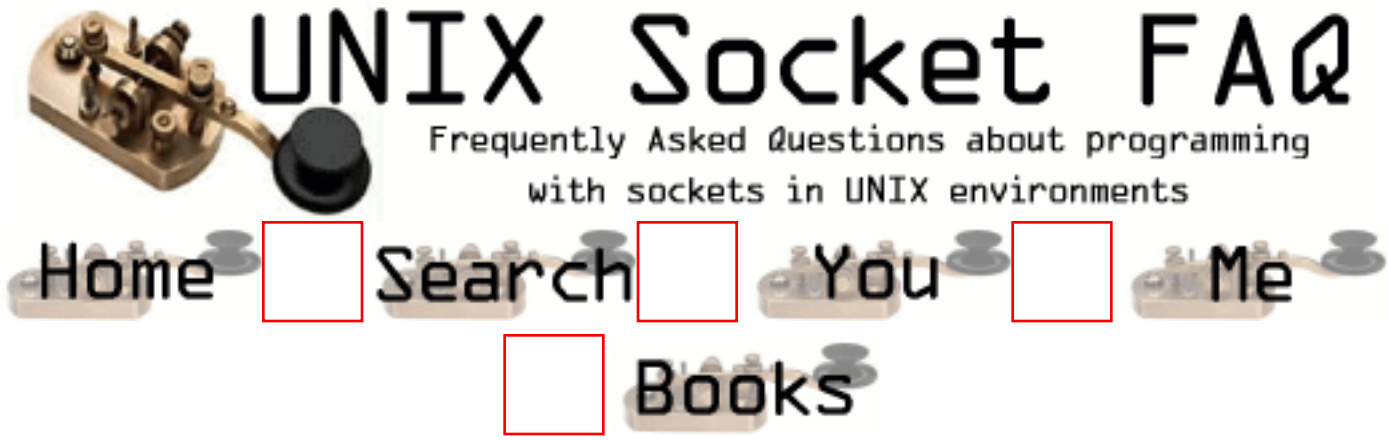
I thought that's what was wanted: the per-process limit...

If you want the hard OS limit for ALL processes, well, I think you'll just have to peruse your kernel source (or, at least, some low-level docs)...

And, there's no such thing as truly "unlimited"... If nothing else, it will at least have a limit just based on whatever size the datatype used to hold the value is... However, I don't think you'll likely be allowed to get nearly that many open files in any real world situation... But, some OS's do allow an essentially "unlimited" number of open files, limited only by available RAM (and, of course, the datatype limits I mentioned)... Presumably, on such an OS, "root" would be allowed to call setrlimit(), to increase the max open files for a process to some utterly insane limit... I don't know, though...

Personally, I think if you have that many open files in a single app, that you're running into the default per-process limit (unless it's insanely small), you're doing something wrong... I just can't imagine needing more than 1024 FDs simultaneously open in any app... *shrug*

Add a Comment



New Questions: Questions regarding both Clients and Servers (TCP/SOCK_STREAM)

From: Naim Far

how can connect() and write() in the client succeed while the server's accept() doesn't accept any message from the client ?!

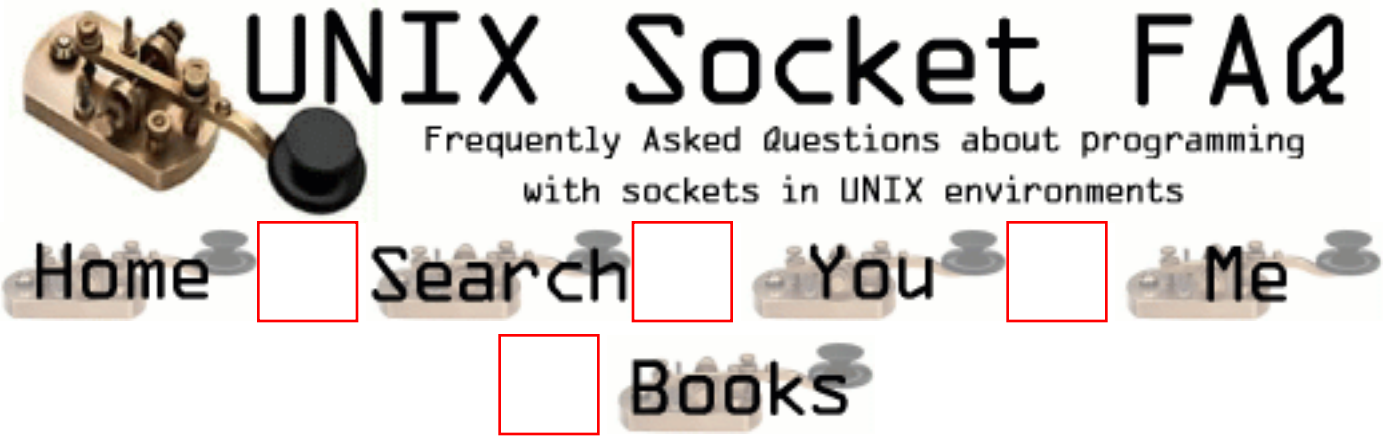
From: Rob Seace

See question 3.3 in the main FAQ...

In short, any listening socket will pre-accept a number of clients, and keep them queued, until the server accept()'s them...

And, write() will just fill the outgoing socket buffer, and the data will be sent later, when it can be... (Or, it may even pre-buffer it on the receiving side... Not sure...)

Add a Comment

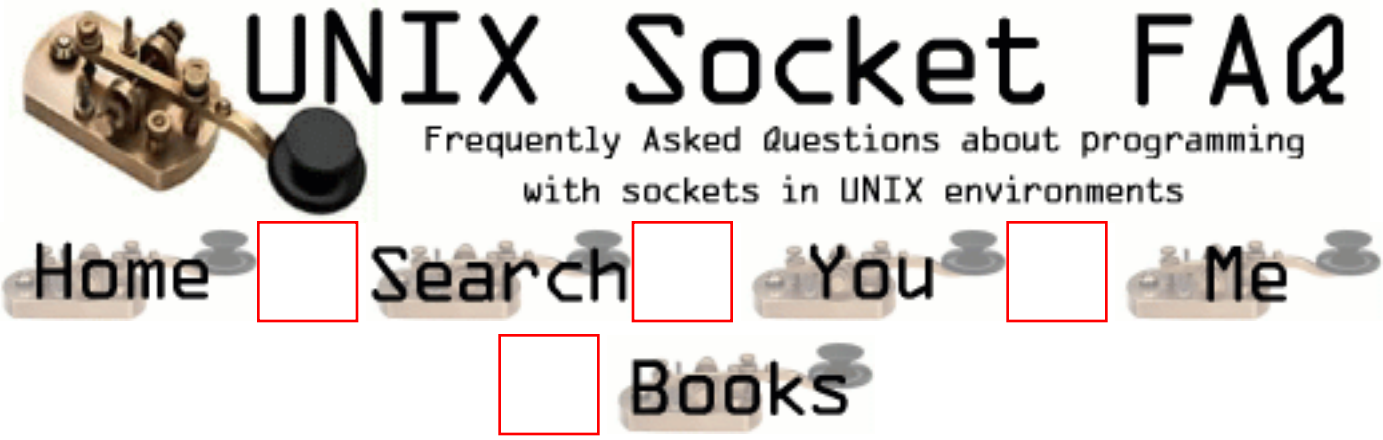


New Questions: Questions regarding both Clients and Servers (TCP/SOCK_STREAM)

From: Naim Far

how can the client's connect() fails while the server has his accept() running?!

Add a Comment

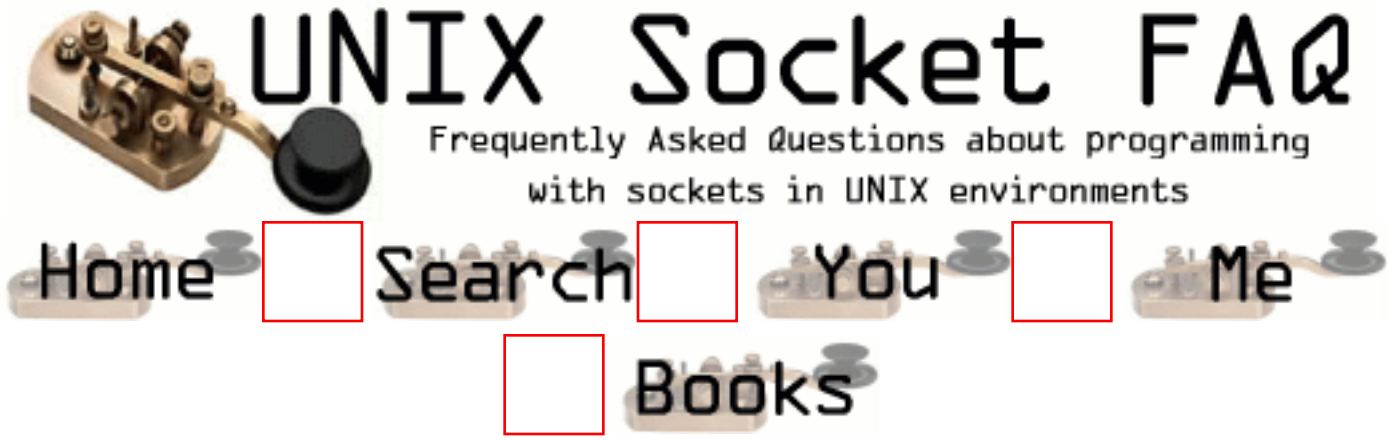


New Questions: How can I bind to port use different name ?

From: Jack

A socket, a IP address and a port was bound, I want to bind another socket to this port now. I know, I must use a different IP address, but how ?

Add a Comment



New Questions: how to bind a port that was bound ?

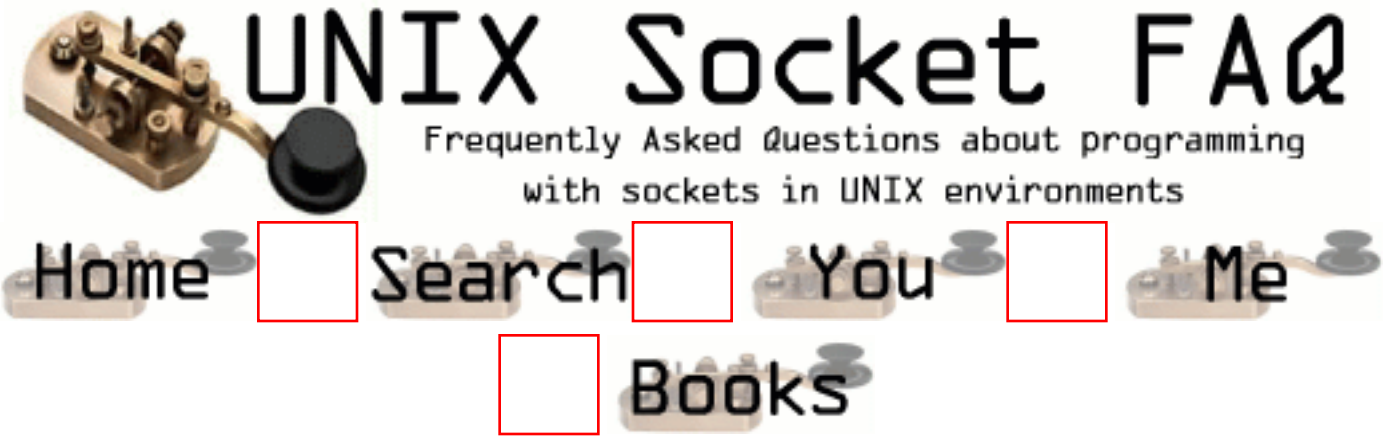
From: Jack

A socket, a IP address and a port was bound, I want to bind another socket to this port now. Can I use same address and same port ? And I can not use UDP.

From: Rob Seace

Didn't you ask this, and I already respond to it, the other day, already? Yep, here it is...

Add a Comment



New Questions: Accept method

From: Nir

When waiting for a client to connect via accept method - is there any default timeout set for accept to wait ?
The problem appeared after the server waits on a client to coonect and after an uncertain time period it hangs.
Any advices ?
Thanx in advance.

From: Rob Seace

No, assuming normal blocking I/O mode on your socket FD, then accept() should just block forever, until someone connects... (Or, until an error occurs, or a signal is caught, interrupting the call...)

What do you mean by it "hangs"? You mean it sits there blocked on accept(), but actually fails to really accept new connections? That sounds odd... Or, do you mean it fails, returning an error? If so, what is "errno" set to?

Add a Comment

UNIX Socket FAQ

Frequently Asked Questions about programming with sockets in UNIX environments

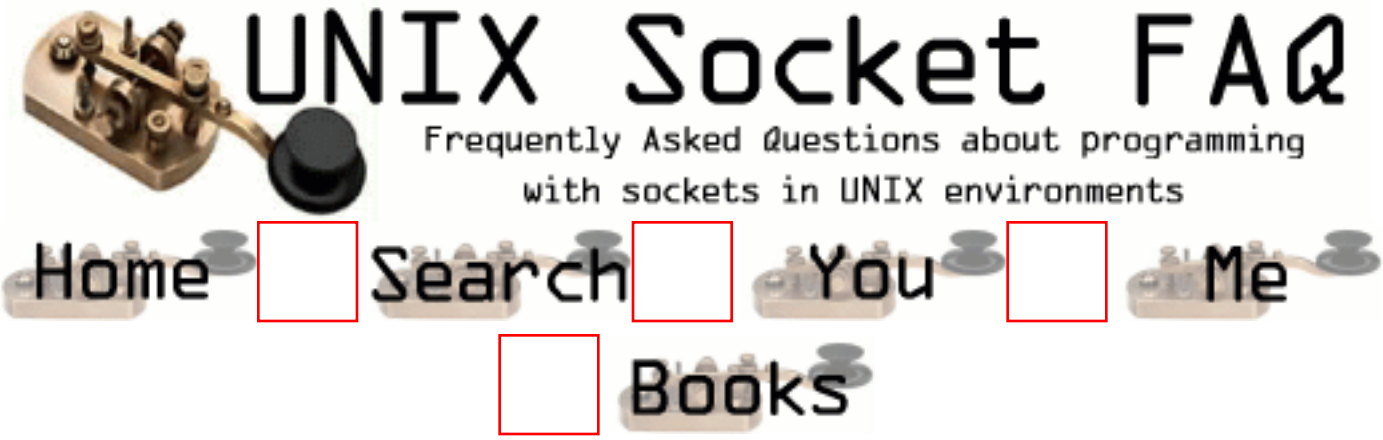
New Questions: Winsocket> From: Vadim

Hi...im very freash at this area...so maybe you could help me.
Q: Can unix comunicate throug sockets to winsocket..(or onother win-based socket) with TCP/IP
thanks!

From: mike

Yes, of course they can. The good thing about TCP/IP is that it's a standard protocol, and all systems can communicate with each other.

Add a Comment



New Questions: is there any way to determine which process id is running a particular server ?

From: AK

Hi all,

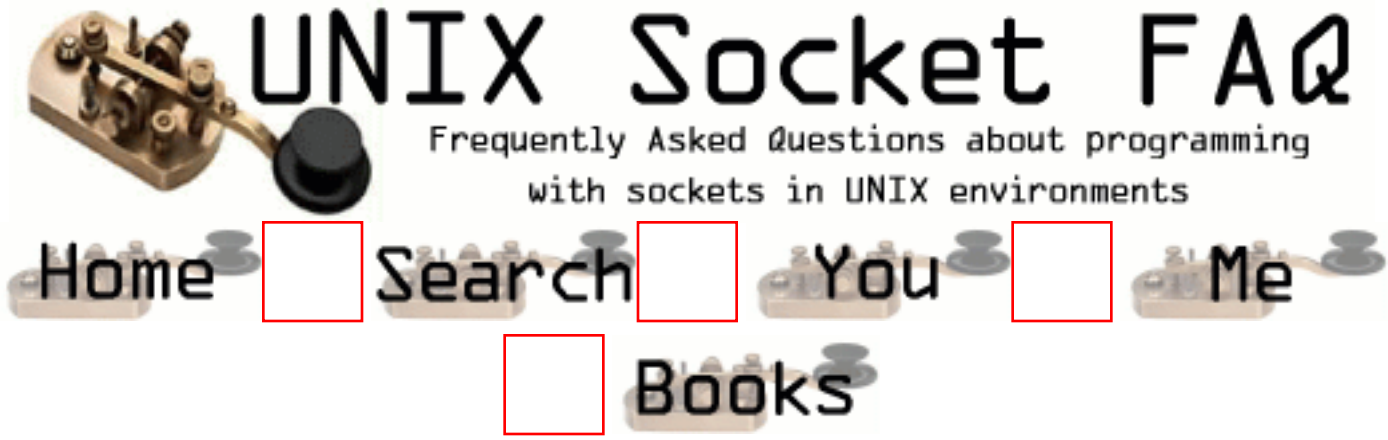
Is there any C system call in Solaris by which it can be determined
which process id is running a server on a particular port ?

Thanx and regards.

From: Rob Seace

Well, I don't know much about Solaris, but I tend to doubt
there is such a standard lib function available... However,
the program "lsof" has been ported to Solaris, I believe,
and that can give you the info you want... And, if you
examine its source code, you can see exactly how it does
it (it would vary greatly, depending on OS), and then copy
it in your own program, if you want... The main distro site
is "ftp://vic.cc.purdue.edu/pub/tools/unix/lsof/"...

Add a Comment



New Questions: simplest way to send structures via sockets

From: Gregory

I am very new to socket programming but loving it by the day. I'm trying to write client /server programs which send various info between each other. How can i send this info in structures. All the read /write functions for sockets seem to require string arrays. I've told a can use RPC. What is it and where can i find it.

From: Glenn

One implementation is to have some message codes that tell the other side of the socket what kind of structure to use, then do a memcpy from the byte array into the structure.

From: sulfy

some example code would be great, since sending structures are giving me and my compiler a headache =)

From: Loco

The sender code would look like:

```
// struct definition
struct mystruct {
   char somestring[SIZE];
   int someint;
   //etc...
};
...
// Sending data
struct mystruct mydata;
// fill struct
```

send(mysocket, (void*)&mydata, sizeof(struct mystruct), 0);

The receiver would do something similar:

struct mystruct recdata;

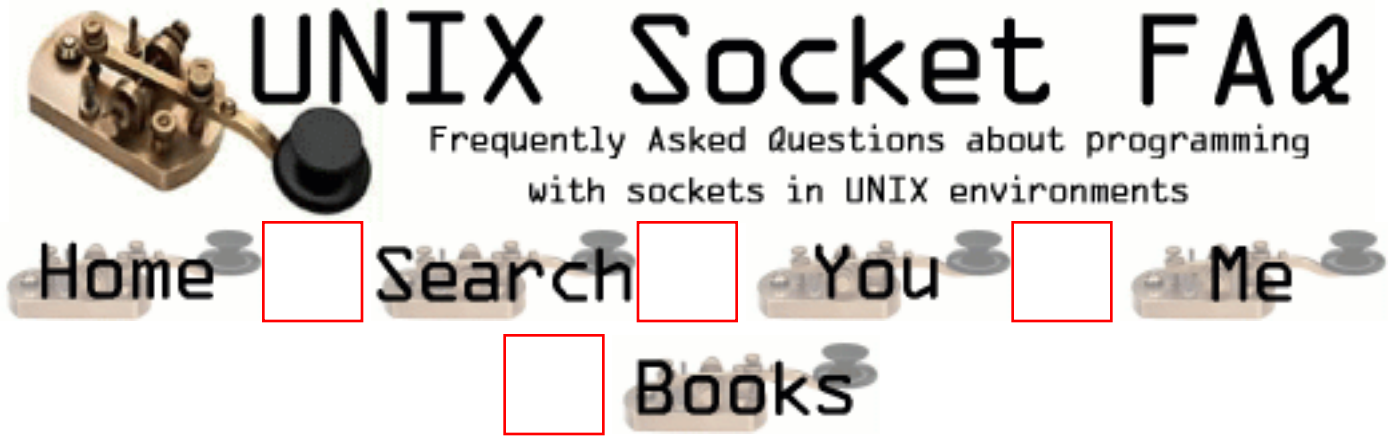
recv(mysocket, (void*)&recdata, sizeof(struct mystruct), 0);

The struct definition must be the same for both parties. You must be aware about datatypes in different platforms.

htons(), htonl(), ntohs() and ntohl() may help you send integers in a consistent format.
Do not send pointers (they are useless out of the context of your process) neither file descriptors.

:) (HAL)

Add a Comment



New Questions: Recv and Send functions Result

From: Ismail Civgaz

How I can create OnBufferReceived/OnBufferSent events in my app.Needed realtime results of Recv/Send functions in non-blocking mode.

Add a Comment

UNIX Socket FAQ

Frequently Asked Questions about programming with sockets in UNIX environments

New Questions: HOW MANY BITS ARE IN A TCP SOCKET?

From: Tommy

?

From: John Doe

In excess of 4.

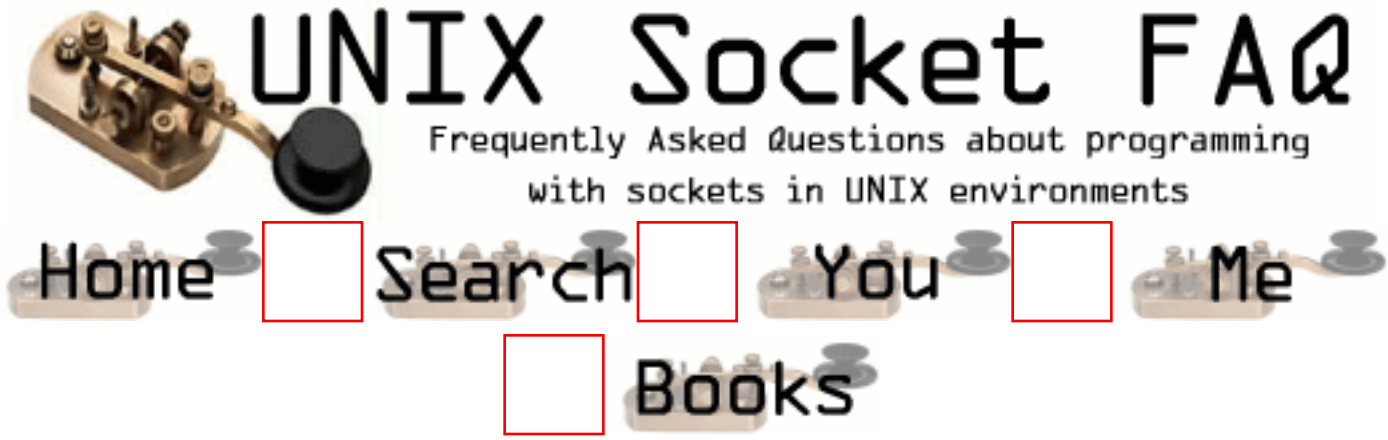
From: dave

yes

From: George

Fish.

Add a Comment



New Questions: server host name

From: Vaidhyanathan

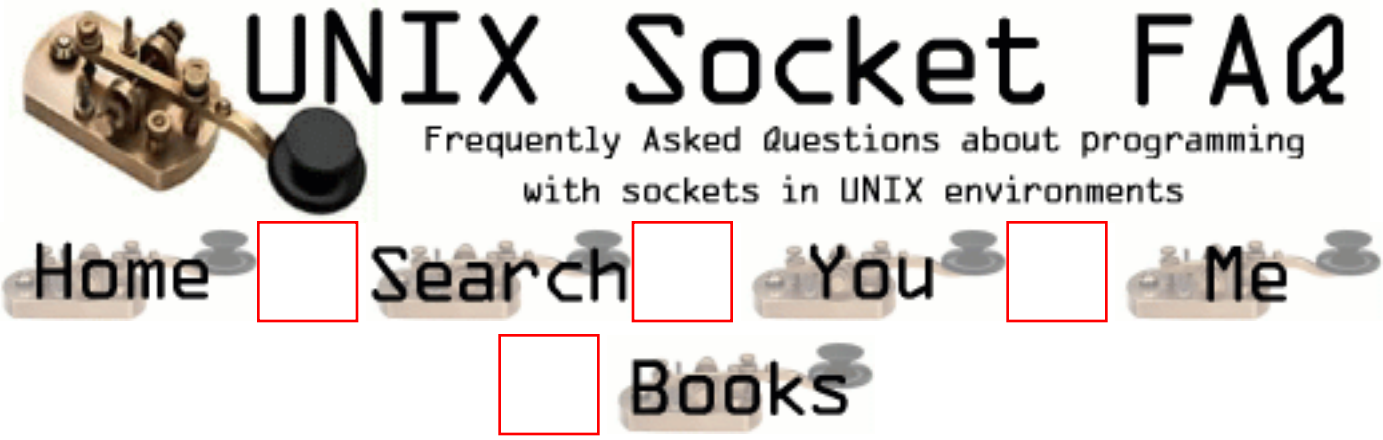
i have made a client/server program.the command line arguements to server program are portnumber and hostname.if hostname is not given i am using INADDR_ANY to find the host name.client's command line arguements are server hostname and portnumber.how can i find the server host name.
i have written the program in c under unix environment.

From: mahesh

client has to know servers name to contact the server, it
doesn't matter if it is in the numbers and dot notation
or words and dot notation. you can get the servers address
structure using gethostbyname() function.

-mahesh

Add a Comment



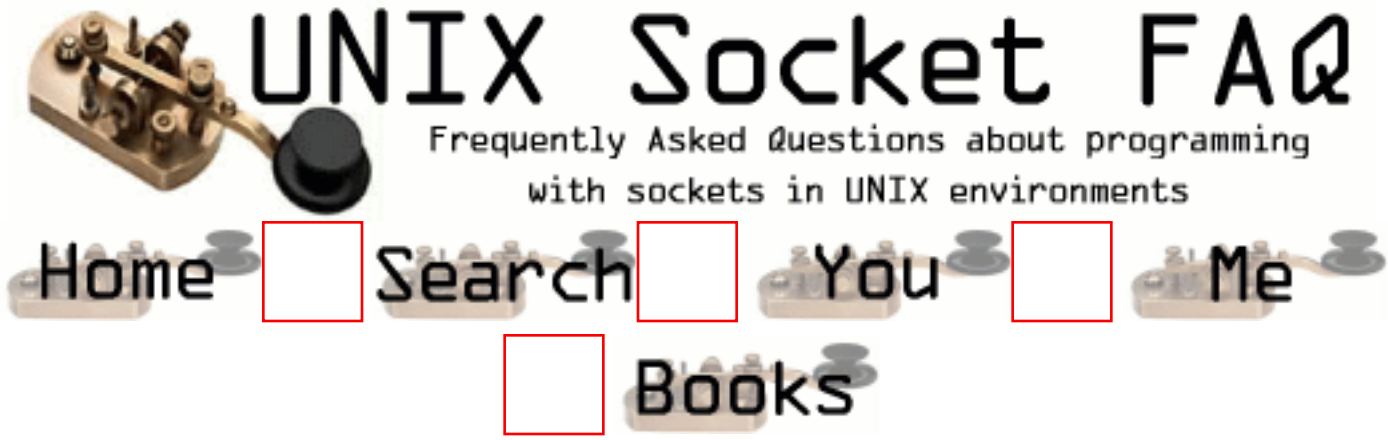
New Questions: blocking write

From: orna

my application works that
- in recv operation it's in non-blocking mode
  and in send opreation it's in blocking mode
so after setting to non-blocking recv i need to set the write to blocking
but i could find a CONST flag to this (like O_NONBLOCK)
i do work around that save the socket's flags before settting the socket to non-blocking
is there an eazy way?

thanks in advance
orna

Add a Comment



New Questions: TIME_WAIT

From: mahesh

hi,

i'm trying to write a proxy server, (browser contacts me, i forward the request to webserver, webserver replies to me, i forward the reply to browser). i'm facing problem with the TIME_WAIT states. a site like www.cnn.com seems to be opening about 20 connections at one time and if i continue to access it in quick successions the no. of connections with TIME_WAIT state increases and my programs gives an error (# 24) too many files open. i know that TIME_WAIT state remains for 2*MSL or 120 sec. one of the solutions i found after going through several articles was to send RST message instead of or after closing the socket. i don't know how to send a RST message, if you knwo it please let me know. if you have faced this problem before and have been able to solve this problem, please share it with me.

i think the client/server that initiates the close (sends the FIN) reaches the TIME_WAIT state. in my program, i close the connection once i write the data to the proxy client. so i make as many no. of closes as the no. of connections the proxy client makes, and hence my program ends up accumulating the TIME_WAIT states. i think if the browser closes the connection before i close, i might end up with CLOSE_WAIT state, but the problem is how do i tell the browser (i'm using netscape for my proxy, HTTP 1.0) to close the connection. or is there a way around to avoid accumulating the TIME_WAIT states.

thanks a lot for any suggestions/hints.
-mahesh

From: mahesh

i found the problem, and the problem was i'd missed to close one of the file descriptors which kept on growing until it gave the error "too many files open".

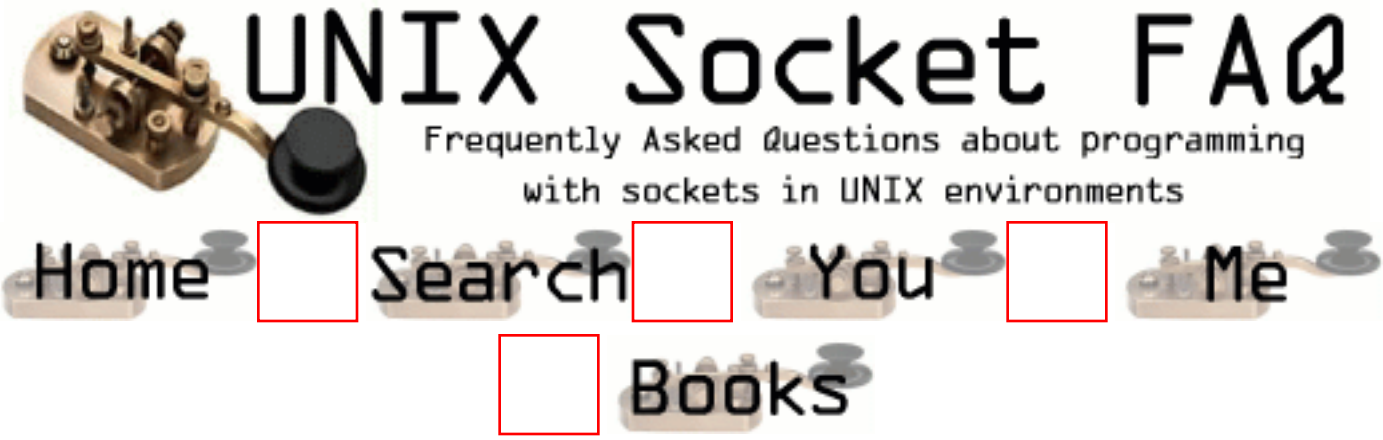
but TIME_WAIT state does accumulate very fast. connecting to www.ccn.com, within a min i got 317 TIME_WAIT states

---

From: Rob Seace

Yeah, the build up of sockets in a TIME_WAIT state really shouldn't cause any harm... You'd have to manage to churn through about 64K of different sockets in the short timeout period (2-4 minutes, generally), in order for you to really run out of available ports to use... (Well, I guess it depends on how your OS assigns ephemeral port#'s... It may only choose them from a limited range, with far less than 64K to choose from... But, still, I think it would generally be a fairly large amount, and more than you'd ever go through in the space of the TIME_WAIT timeout...)

Add a Comment



New Questions: How to flush the socket ?

From: acheng

when I write a chat server with socket ,i encount such a problem-- i cannot flush a socket efficiently as do with a file stream. who can help me? Any help will be greatly appreciated.

From: Rob Seace

First, I'll assume you're not using stdio file pointers for your sockets... If you are, that could lead to problems, because you're then going through two sets of buffers (the stdio buffers, then the socket buffers)...

See questions 2.11, and 2.16 in the main FAQ section... Disabling the Nagle algorithm might be what you're looking for...

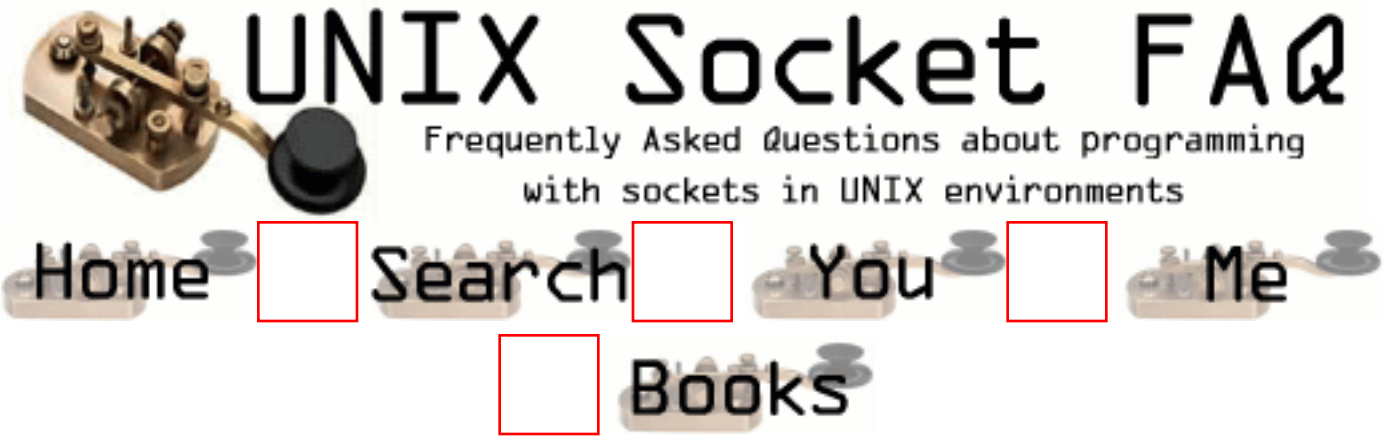
From: acheng

But I use SCO unix, the OS has no TCP_NODELAY option , i can not disable Nagle algorithm . what should I do next?

From: Rob Seace

Um, change OS's, maybe? ;-) Seriously, I don't know...
But, I find it really odd that SCO would not have a
TCP_NODELAY sockopt... That's really a very standard thing,
and is used by a lot of software out there... If it doesn't
have something so basic and common, I'm not sure I'd put
much trust in its sockets handling at all, anyway... *shrug*

Add a Comment



New Questions: client /server problem

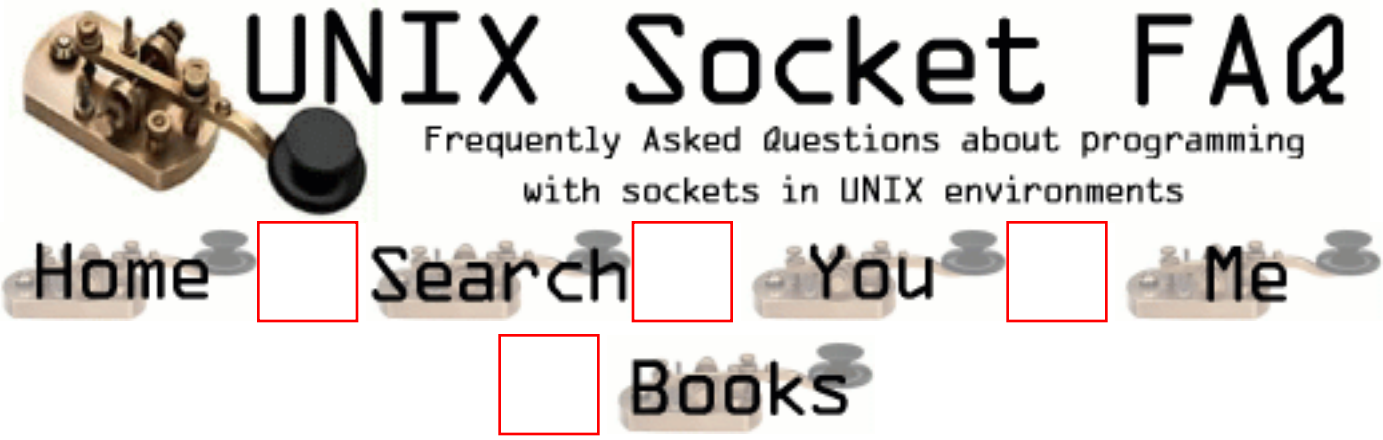
From: nu

How can I implemenr a client/server program that do the following:
1. The client requests a file from the server.
2. The server search for the requested file and send it back to the client.
3. If the server do not have the file he will connected to another server to search for the file.

From: Mahafuz

pls see Gnutella protocol.

Add a Comment



New Questions: Maximum speed in the socket

From: juanuker

I would like to know which is the maximum speed that data
can reach using sockets.

Have you got some kind of performance results ?

Thanks in advance.

From: Rob Seace

Your speed would be limited by your network connection's
bandwidth, for the most part... You should be able to get
close to the theoretical maximum transport bandwidth... Of
course, there's a bit of overhead for IP and TCP headers
(plus, physical transport headers, such as ethernet), so
actual data throughput will always be a little less than the
max... But, it should be able to get close... That's
assuming you are fully maxing out all your packets, and are
on a stable, reliable connection, so you don't have to do
a lot of retransmission and such, too...

From: juanuker

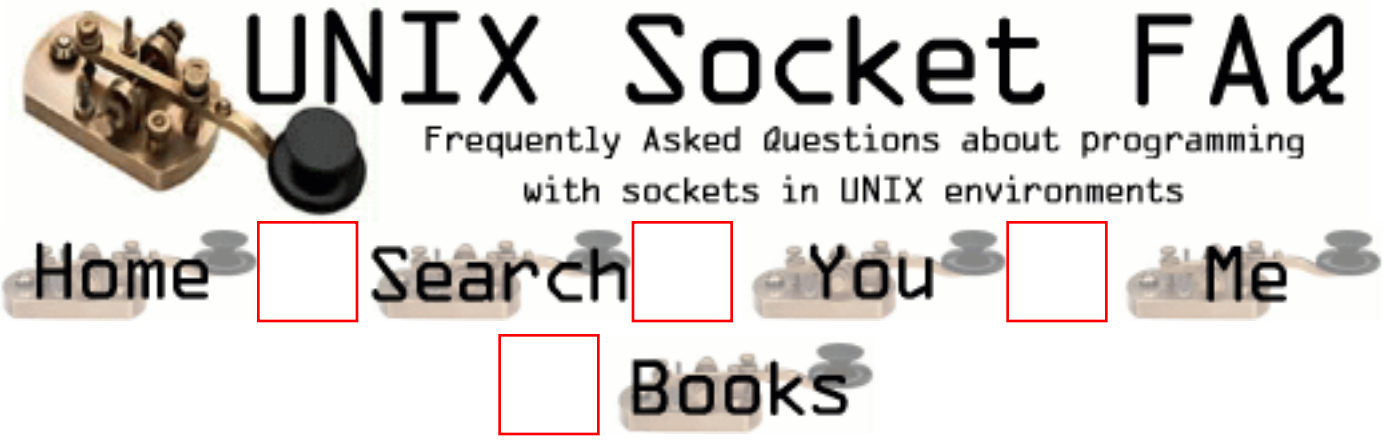
Suppossing that I have an unlimited bandwidth,
is there any other limitation in the socket implementation ?

Thanks.

From: alex

Your speed for connections (other than localhost) are limited to the speed of light.

Add a Comment



New Questions: Socket reliability

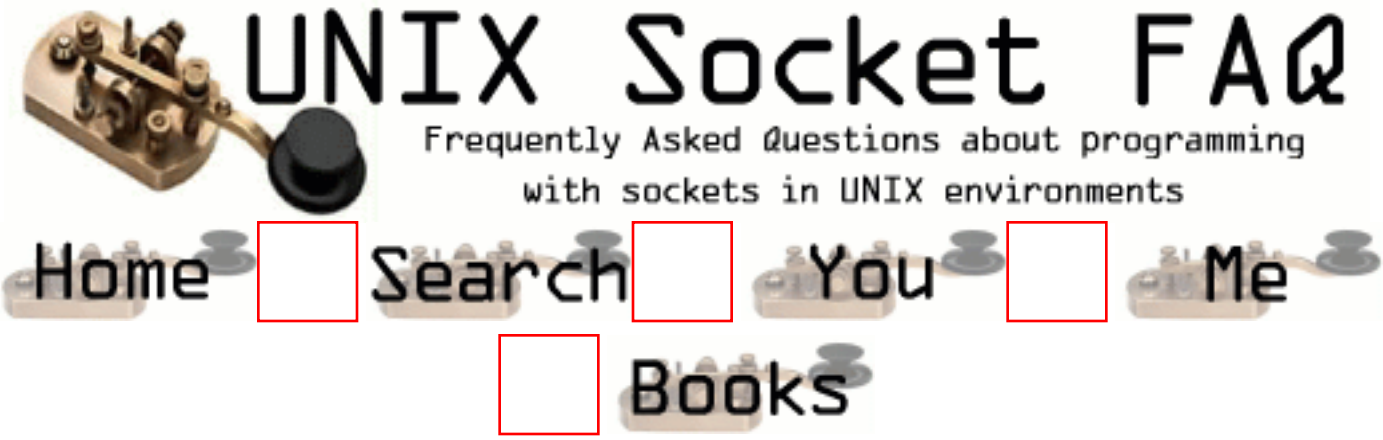
From: Ankur

Using socket API, how do we know that the peer end has received the data without an acknowledgement from that side. I hope the write system call return the number of bytes written in local TCP buffer not at the peer buffer.

From: Mahafuz

When you are using a connection oriented protocol (like TCP) it is guranteed that what you write to your socket buffer is received by the peer 100% error free and reliably.

Add a Comment

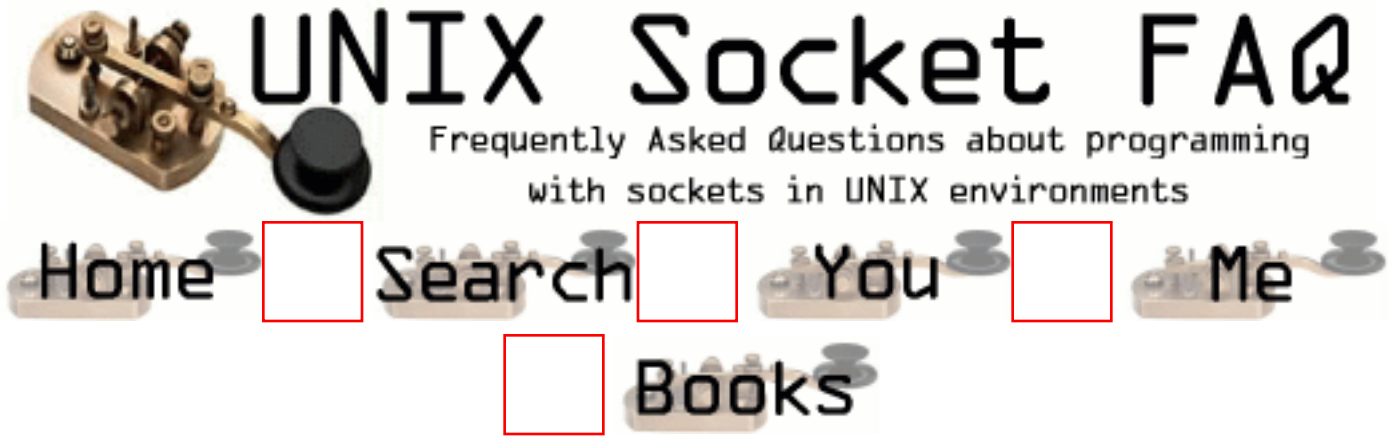


New Questions: blocking ip address

From: Donny S.

how do i block a user from telneting to a certain ip address?

Add a Comment



New Questions: send a buffer from server to all clients?

From: cabbe

Im writing a chat program and a server to learn about sockets and threads. How can i send the recieved(server) buffer to the sockets in all threads? (server -> all clients). can that be done?

Add a Comment

New Questions: Accessing non-HTTP servers through a proxy

From: YMS

Hello,

I am experimenting with web proxy servers. For testing, I telnet to my web proxy and type:

"GET http://BLAH.BLAH HTTP/1.0" <crlf> <crlf>

to access a web page outside of my firewall . I am now trying to access a server outside of my firewall that is not HTTP-based; I want to use a protocol that I developed myself. Is this possible via a web proxy? What kind of string to I need to pass to the web proxy?

Thanks,
YMS

From: Rob Seace

I think you want something like httptunnel...

From: Hector Lasso

Hi,

If your proxy allows the CONNECT command, you may use it to connect to some other server like this:

CONNECT host/address:your_port_number HTTP/1.0<cr><cr>
and voila, you have stablished your connection.

I tested it on my proxy (enabling access control for some ports) and it worked just fine!!!

Example

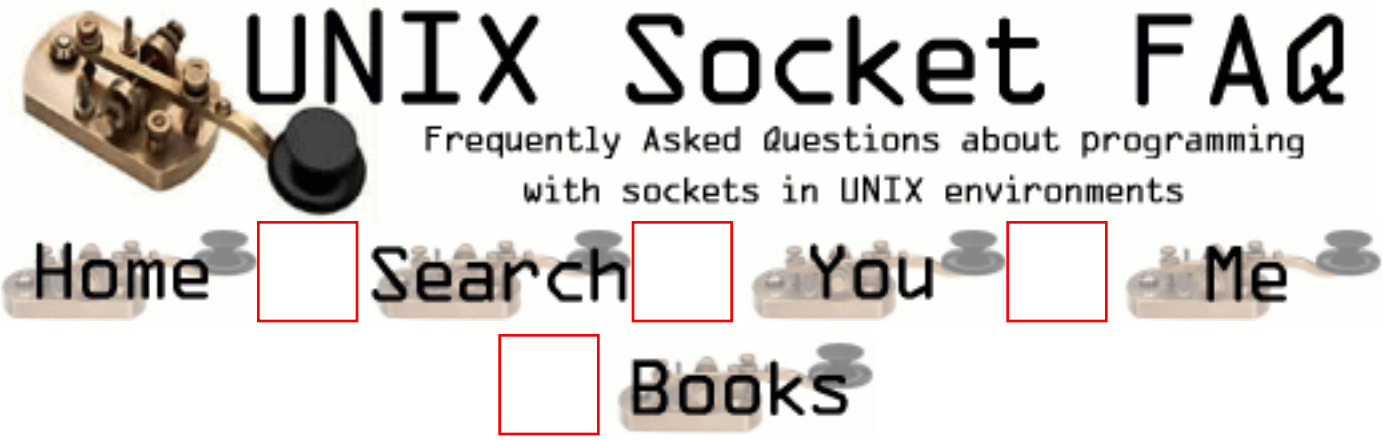
CONNECT 192.168.0.1:25 HTTP/1.0

HTTP/1.0 200 Connection established

220 myserver.mydomain.com ESMTP Sendmail 8.11.0/8.8.7; Thu, 14 Dec 2000 17:21:22 -0500

Good luck

Add a Comment

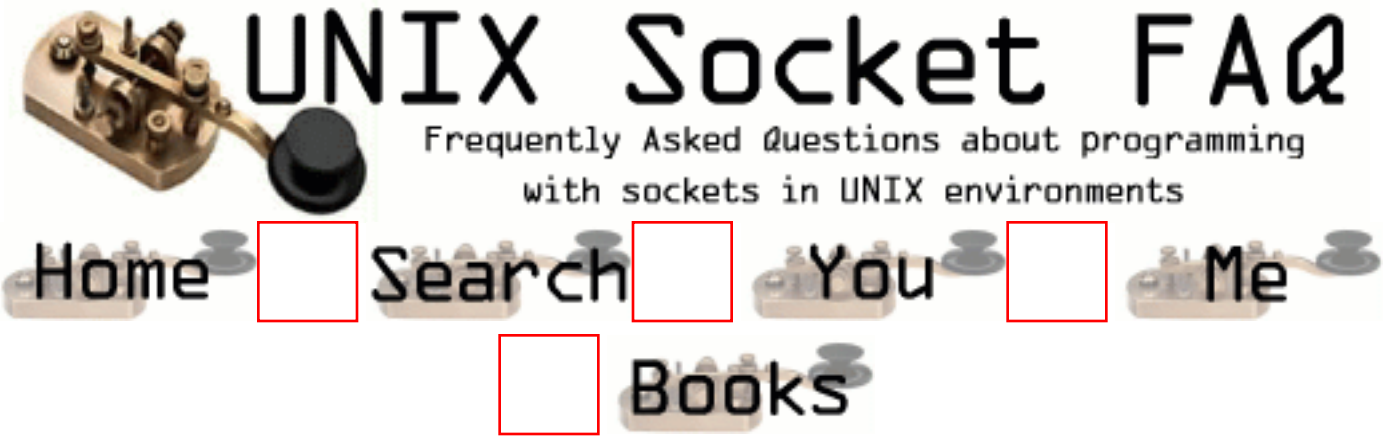


New Questions: re: socket error

From: Emily

I can login to my novell network successfully, but there is a problem when i open outlook express and IE 5.5.... the error message is "server not found" and "socket error". pls advise....

Add a Comment

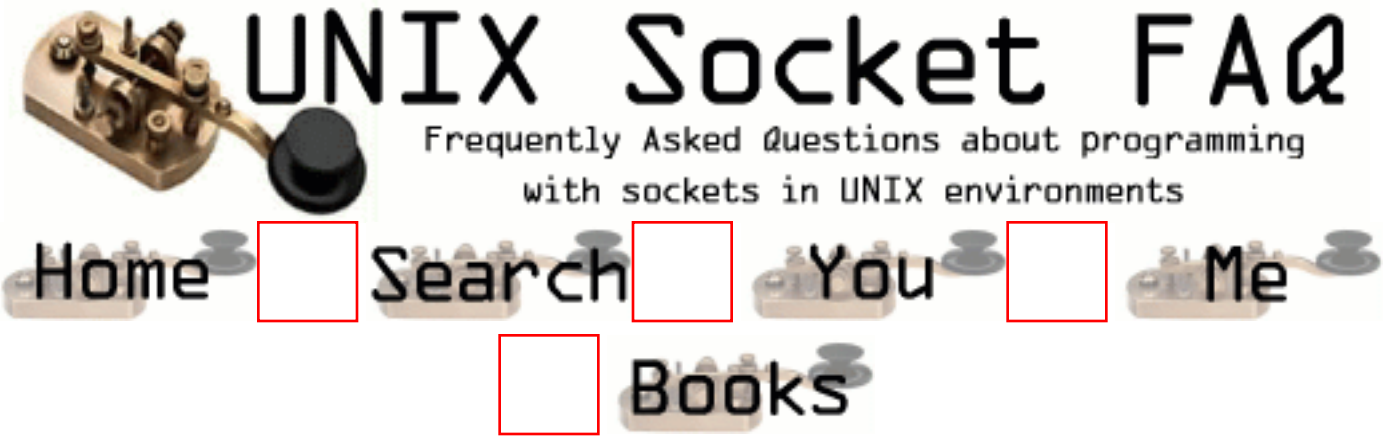


New Questions: Asynchronous I/O

From: Christos

I want to implemment a client program to send and recieve data from a TCP/IP connection. Also the client wants to receive data from another connection established by the server. The second communication channel should be an asynchronous one. I have define a function to establish a protocol when the SIGIO sent but I can not return to the main control of my program.

Add a Comment



New Questions: Encrypting data on socket

From: Roy

I need to protect the data i'm sending on a TCP socket on unix (Our own client and server programs).
Suggestions i've had are
1) Using secure sockets
2) secure shell on unix
3) DES encrypt message in our code before sending

I'm don't know very much about first 2 options (is option 2 valid?) to compare against option 3.

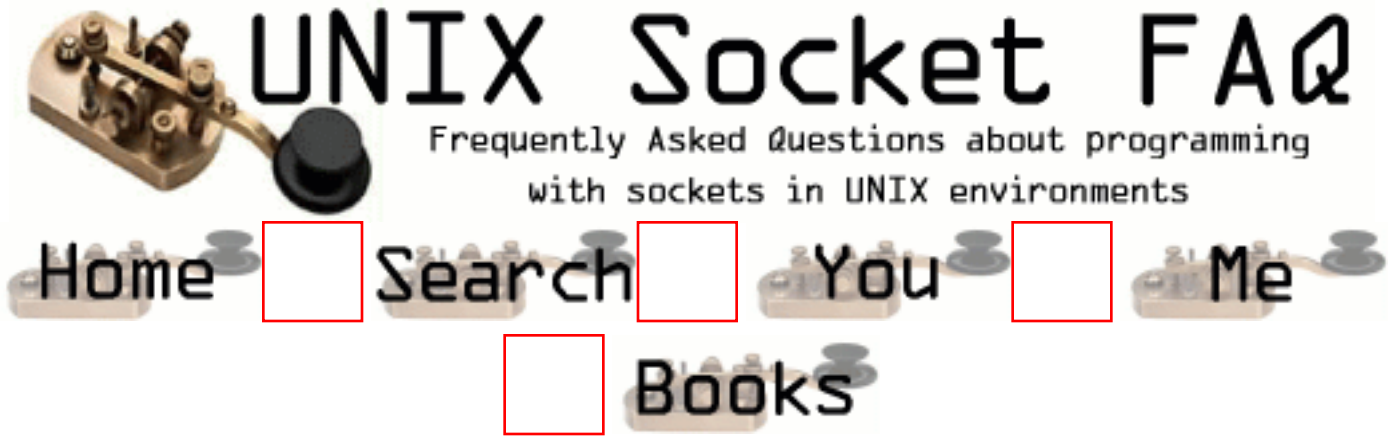
Any comments welcome.

From: Rob Seace

Well, I'd certainly recommend going with something standard and already widely in use, such as SSL, rather than rolling your own encryption method... Just much less chance of screwing something up, and much higher likelihood that the data will be secured well... There is a freely available implementation of SSL, which you can use, if you like: OpenSSL...

Your option #2 (tunnelling through SSH) is also certainly a possibility... SSH is also highly trusted, as far as encryption strength/integrity goes, and it's easily able to tunnel any TCP traffic you like... However, you then have to deal with setting up the tunnel, completely outside of your own app... Which MAY be ideal, depending on your circumstances... If you wish to be able to operate in the clear, in some cases, it may be best to let some external thing like SSH worry about all the encryption stuff, and let your app just do everything the same way, no matter what... But, if you always want to encrypt everything, it's probably best to code it right into your app, by using SSL calls throughout, rather than rely on an external tunnelling app...

Add a Comment

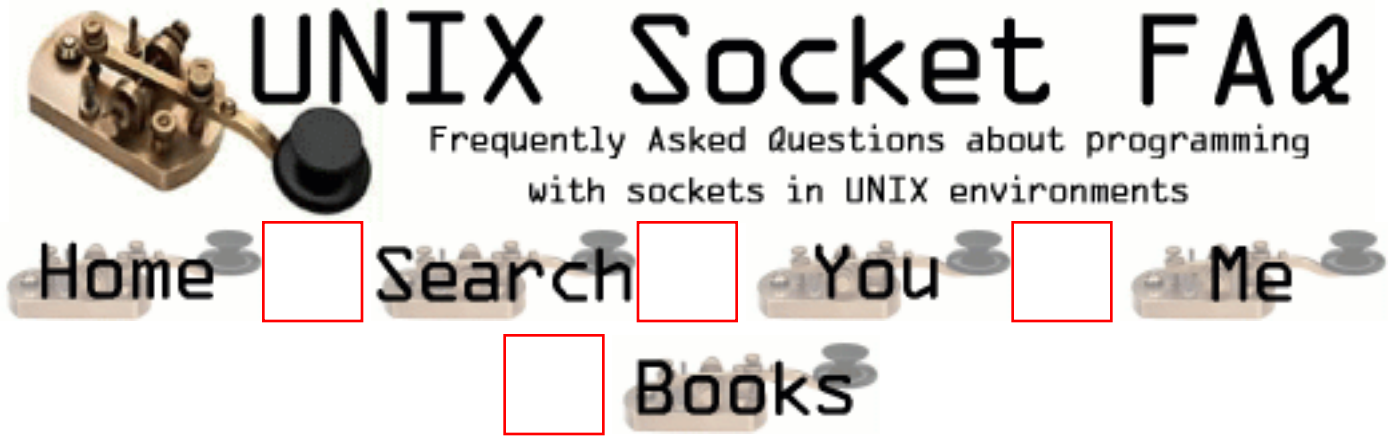


New Questions: RE: Adding packet handler for socket

From: Harshad

Other than using dev_add_packet to add packet handler for a packet type is there any other way to do the same thing ?
Trying to install my code handler to do some pre-processing before passing packets to IPv4 handler.
Any help appreciated.
Thanx.

Add a Comment



New Questions: Select hangs randomly

From: David Mack

I have an application that will periodically hang on the select statement for no apparent reason. It works great and then every once in a while it will just hang.

This is a server application that uses forks to handle the incoming connections. Any ideas can be emailed to me.
I have checked the file descriptors are being zeroed out each time but it hangs. This is a proxy server by the way

---

From: cma

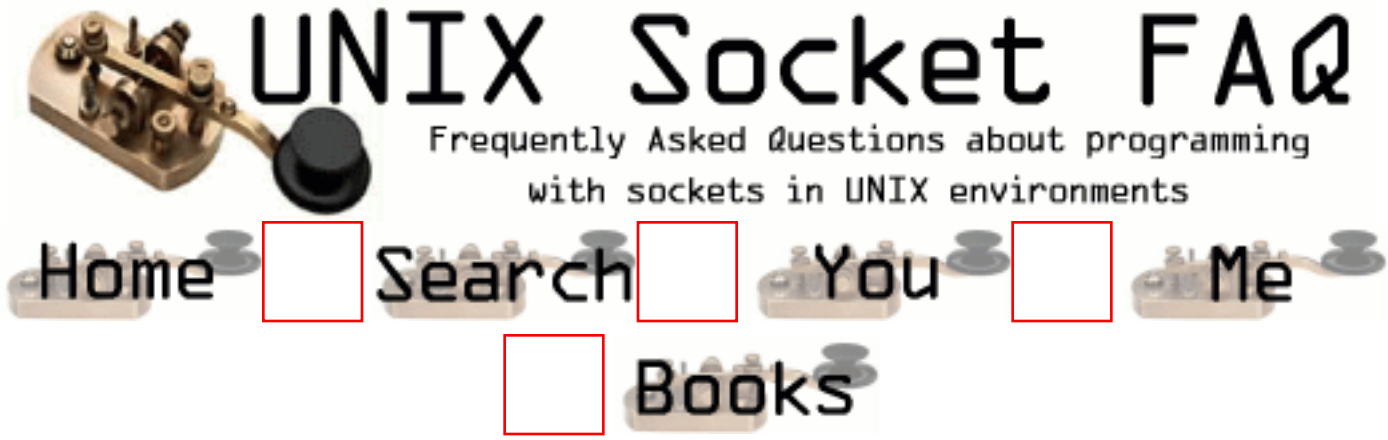
me too.... mine uses select() for the tcp socket and rpcs for other calls, and it just hangs after about 1 hour.... used truss, but still cant tell exactly whats happening

---

From: Loco

Post code!!!!

Add a Comment

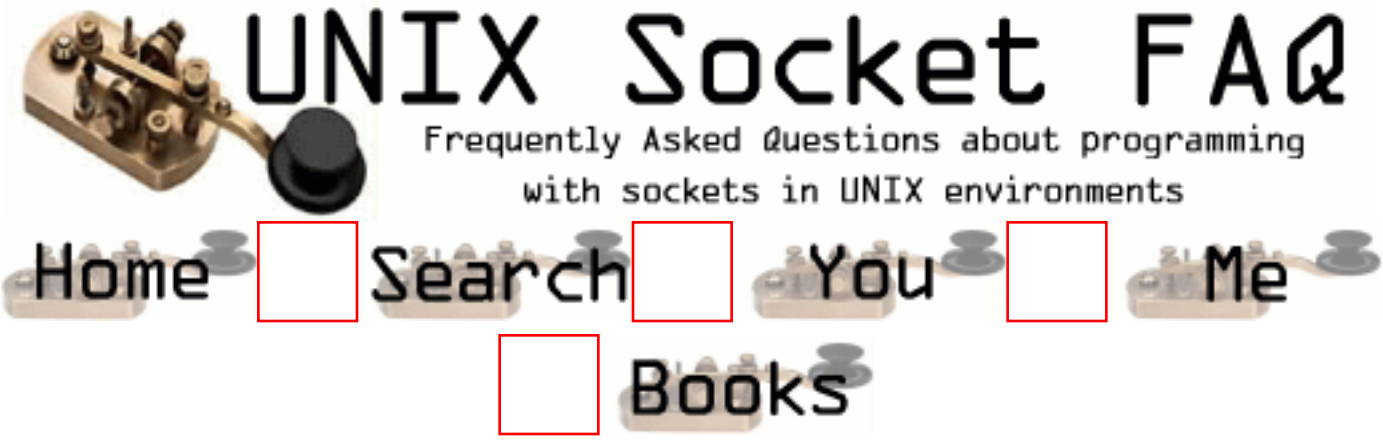


New Questions: tcp server

From: Tom Thorpe

hi. i have used SO_REUSEADDR to bind more than one socket to a port. my problem is now using select to get incoming connection requests on them - no matter how many clients try to connect, select only returns 1. a look at netstat shows that the sockets are being opened correctly. help ?

Add a Comment



New Questions: non blocking connect with select

From: ornaz

i write a program that need to to do multi connection.

i do it as a non blocking connect and use the select
for checking connection successfully or not.

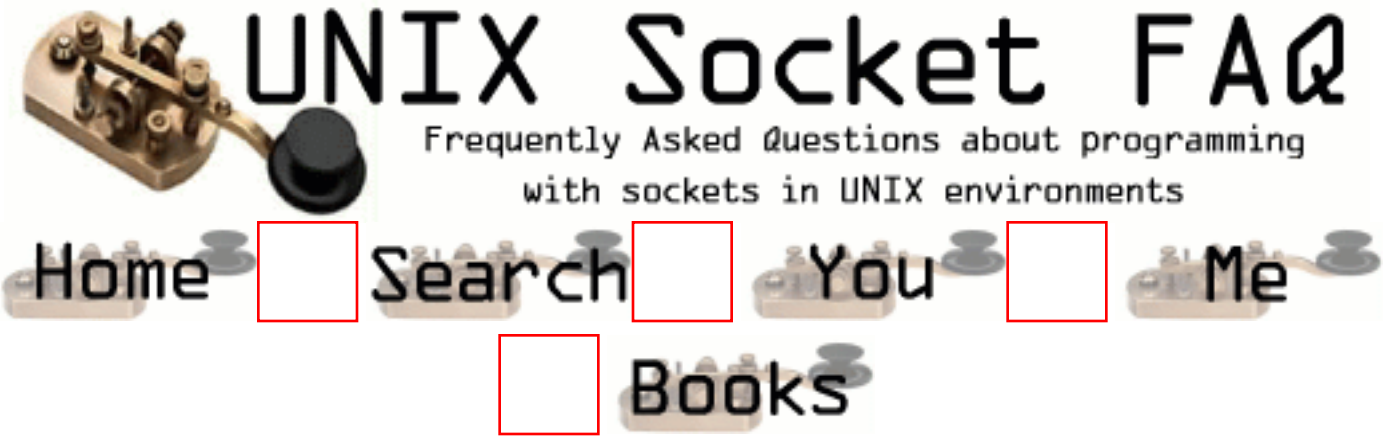
but from a reason some sockets are not waking up
i see that my select wake up on timeout only
after i do select again but again.....

i do check if i set the descriptors to the readset and to writeset
and i do check the max socket i set in select func

so do u think on diffrent problem?

waiting for your answer
tanks in advance
orna

Add a Comment



New Questions: socketing data loss between Unix and Windows

From: John R. C.

Folks,

I have an application which is running under Windows and another application running under UNIX that are connected via a socket. The Windows application is the server and the UNIX application is the client. When I send data thru the socket, from Windows to UNIX, data is lost. Everything seems fine for a short time (a few hundred characters) and then data begins to get lost.

Any ideas? Thanks.

From: Alexander Pevzner

This is a bug in your code. Check it again.

The most probably reason for such a problem is that your program doesn't verify values returned by send/recv/write/read functions.

These calls may transmitt less bytes that requested by your program.
So the command

```
  rc = send( sock, buffer, sizeof( buffer ), 0 );
```

may actually send less bytes that sizeof(buffer), even on blocking socket. The count of actually transmitted bytes will be returned as a function value.

This is normal behavior, it's not the error. You program must verify values returned by send/receive calls and call these functions again if required.

--
Wishes, Pzz

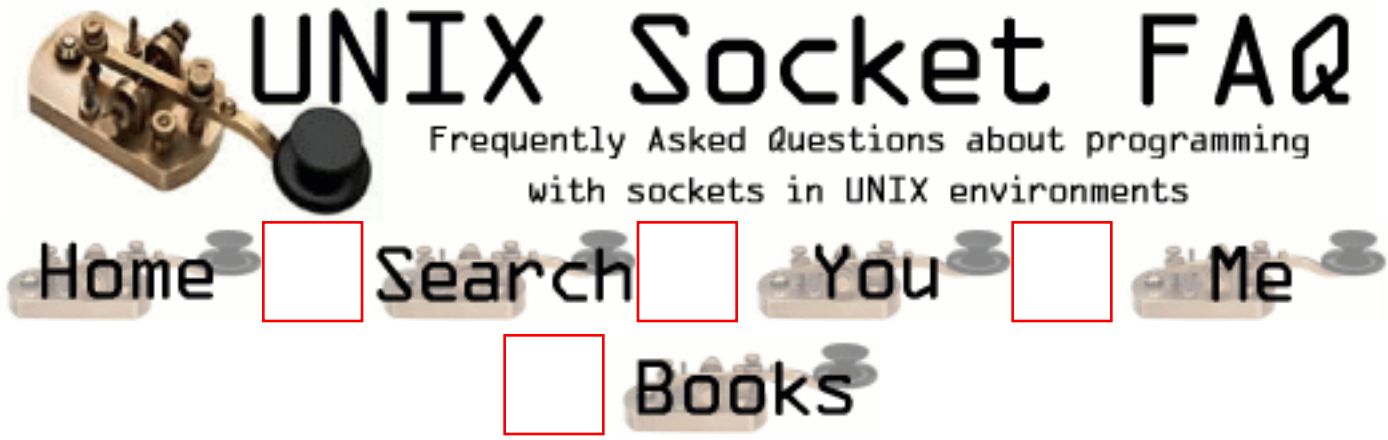
From: Raul Sanchez

Hi, friends
I'm Raul from Ecuador, I'm working in a university project and I read about conexion between windows and linux by a sockest, wher I could get programs, source code or something about it!

For your help!

Tanks

Add a Comment

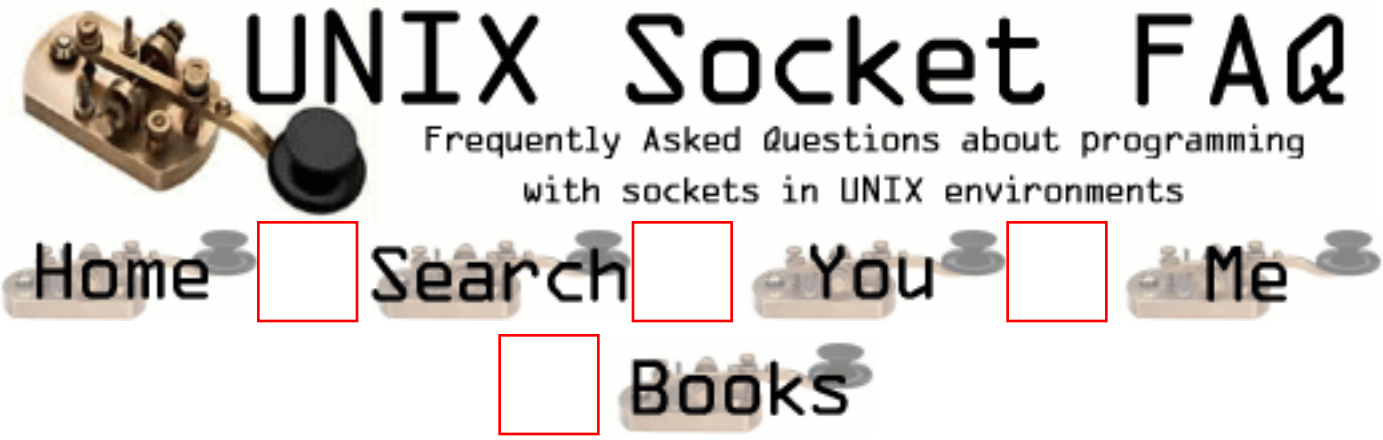


New Questions: SCO -> DOS Communication via socket

From: Edmund Schurnik

For the first - please excuse my bad english, i hope, you understand whats my problem:
I have a P166-Server with SCO Unix and some DOS-Terminals (Siemens ES235). The Siemens-Terminals are connectet via TCP/IP to the Unix-Server. All works correctly.
Now i take a new Server (HP-E800 Pentium III 866). Now the socket-server says "read 0 byte" and the communicaten to the Siemens-Terminals failed. The software for the socket-server is been recompiled on the new server, but it dosn't work.
The network-parameters are all the same like the old server, i can "ping" the siemens terminals without errors.
At two times - i don't know why - the connection was ok, i can send data from terminal to the server, i receive answers from the server. The next login failed.
Can there be some timing-problems with the new, fast PIII-866 Server ????
I hope, somebody can help me ! It's very, very urgent !!!!!

Add a Comment



New Questions: Meaning of UDP "connection"

From: Sanjay

I have another question: When I "connect" a UDP socket to a remote addr and port, is there a corresponding "accept" at the other end for my "connect" to go through? I know that when a TCP socket is connected, the connection happens when the server "accept"s the connection. What is the equivalent thing for UDP sockets?

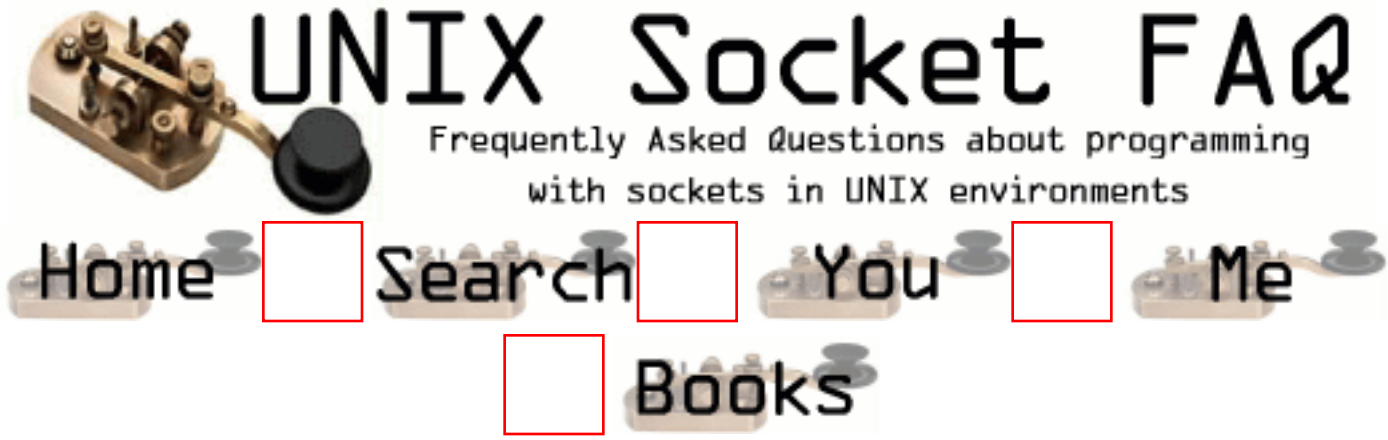
From: Hironimo

There is no equivalent. UDP is connectionless, therefore
no connect and accept. You just create socket, bind and then send or recieve data.

From: Arthur Lung

You can call connect on a UDP socket if you want to. If you
connect () a UDP socket, then you can read/write to it using
send/recv instead of sendto/recvfrom. There is no corrosponding
accept on the other side. All the connect () really does
is tell the kernel that from now on, when you send data
using this socket, it should go to that particular address.
This is supposed to enhance performance as well, but I'm not
sure if that's still the case now, or if it was only true
back when Stevens wrote the networking bible.

Add a Comment



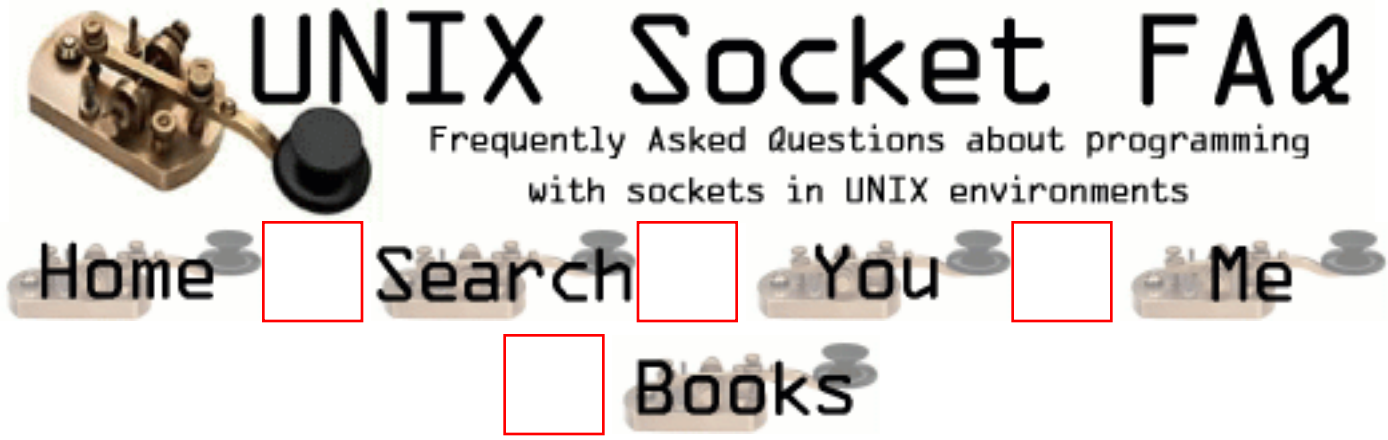
New Questions: How to delete a route using API calls

From: reena

I want to delete route between two machines on a network using socket programming & WIN API calls.It is posible to delete it manually.But i want to create an application which does this task as soon as i switch on the machine.

From:

Add a Comment



New Questions: select() on blocking environment

From: datawar

I got a set of 5 sockets one of them is the listening socket
when i detect a read/write operation on the listening socket
i bind a new port. At this port other clients can connect, when
a new connection arrives and a client sends something i print
out a msg but after that for some reason FD_ISSET keeps thinking
that there is a change on the socket. I attach the loop, note
that new_sock just makes a blocking socket and binds it to
a local port and hostl is a structure with a remote,local sock

Code:

```
while(1){
    FD_ZERO(&rs);
    FD_SET(sock, &rs);
    /* add all sockets to the read set */
    for(n=0;n<=i;n++)
        FD_SET(hostl[n].rsock, &rs);

    to.tv_sec = 0;
    to.tv_usec = 100;
    sel = select(FD_SETSIZE+1, &rs, NULL, NULL, &to);
    switch(sel){
        case 0:
            printf("yo timeout\n");
            fflush(stdout);
            break;
        case -1:
            printf("yo error\n");
```

```
                fflush(stdout);
                break;
        }
        if(FD_ISSET(sock, &rs)){
            if(i == MAX_CON) continue;
            slen = sizeof(hostl[i].r);
            hostl[i].sock = accept(sock, &hostl[i].r, &slen);
            hostl[i].rsock = new_sock();
            i++;
            continue;
        }
        for(n=0;n<=i;n++){
            if(FD_ISSET(hostl[n].rsock, &rs)){
                printf("i got a msg\n", buf);
                fflush(stdout);
            }
        }
    }
```

From: Hector Lasso

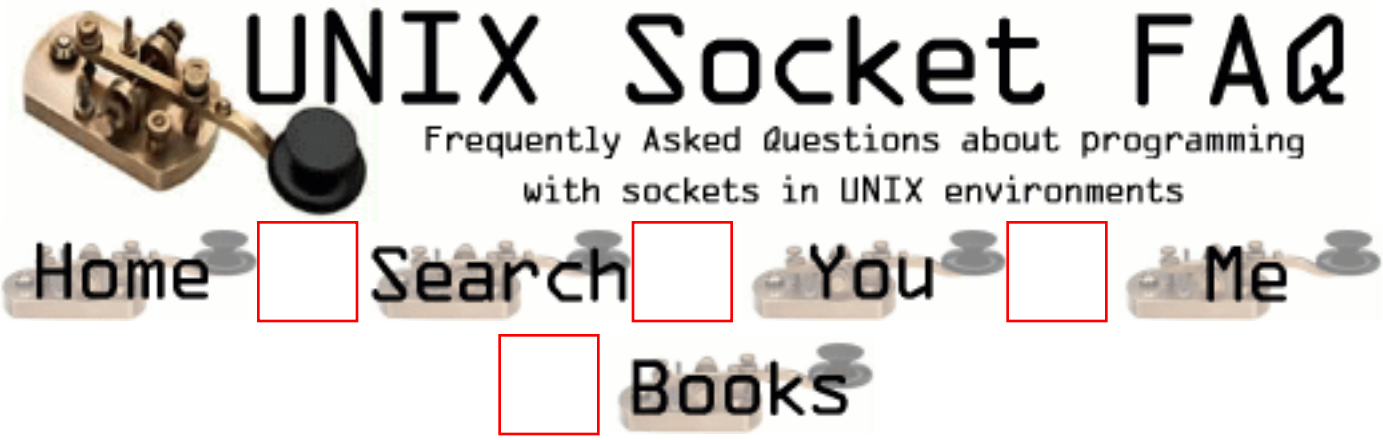
Hi,

As every new connection creates and binds other sockets (and i don't see any accept() for sockets different to the main socket, i assume it is telling you that you have not "accepted" any of the connections to the other sockets...

From: datawar

Thanks dude that was it. i forgot to do an accept!!!

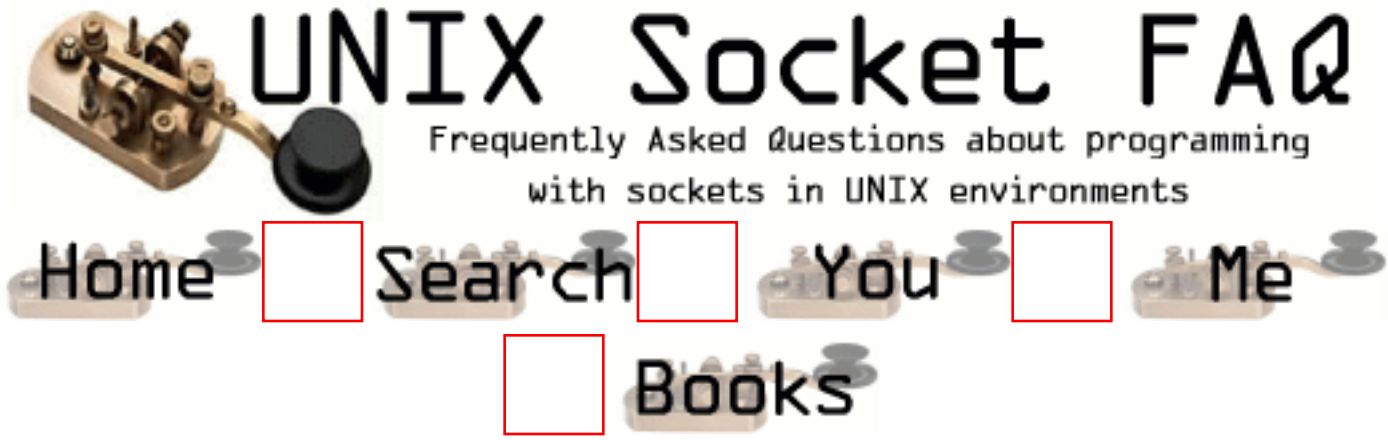
Add a Comment



New Questions: Where can I download QT v2.2.2

From: Jack

Add a Comment



New Questions: how to write optimal performance TCP server

From: Mahafuz

I'm writing a srever that will serve almost 1000 client.
The client/server nature is, client will send a command to
server and the server will send a reply to that client after
executing the command. There is no need for state saving since
the commands are independent of each other. My worse case scenerio
is all the clients are sending command at once.

My question is what architecture should i follow to have optimal
performance.

1. I can fork() server process and have one process server one command.
   After serving the request the child ends.

or
2. I can create a thread to server the client request from server
   process. After serving the request the thread ends.

or is there any better way to do it?

It will also be helpful to know pro/con of each architecture.

I tried to find out what architecture Apache follows but i couldn't
find any good documentation. What i only know is it maintains a number
of pre-forked process to server http request. But i need to know in detail. I'm sure that the architecture followed by apache is a very
good one. reference to any document will help a lot too.

From: Hector Lasso

Hi,

Apache does something like this (which is, i think, a very good approach):

The server process creates a socket, binds it, and then executes "listen()".
After doing this, it forks as many children as the configuration file tells it to. And waits on a infinite loop for events such as a dead child, or not enough servers available, the parent process will never serve the clients.

Every child inherits the socket, so every one "accept()s" on the socket (one at a time using a lock file to serialize) . As soon as a connection is available, the process that accept'ed it serves the client through the connection. When the task is done, the process closes the peer socket (not the one created by the parent process) and returns to the loop where it will (eventually) accept another connection.
While the process is serving a connection, it won't accept any other, however, the other processes will be there to receive the connections.
Every children has a counter or something similar, and will die after a fixed number of connections (which can be configured through httpd.conf or similar).

As they explain, this architecture is very practical:
- The parent process is very simple, and it only is responsible for mantaining enough children available.
- The children processes are very complex (they load modules and do fancy things that will likely crash the process). If a child does something wrong, the process will go down, only affectting his clients (in fact only his client), while the other connections won't notice the problem.

If you need more information see www.apache.org. It has a lot of information regarding their decision to use preforked processes.

Best regards

From: Mahafuz

Thanks to Rob Seace. He has given me a very good reply.
I post a portion of his reply here for others.

Rob Seace wrote >>>>>

I'm not positive what Apache does, either... Your best bet would

be to just download the source code and look at what it does... (That's available from "www.apache.org", among other places...)

But, the approach to choose depends on a lot of variables... I've used each different approach for different cases, before... It really depends on what makes sense for your particular server's needs...

The pros of a "fork() a separate child as clients connect" approach are: each client has its own separate server process, so it can't tie up any other clients; also, if the client somehow manages to crash the server due to a bug, it will only crash the separate process serving that client, and not affect the main server or any other clients; and, it's conceptually very easy to understand, and very easy to implement, and relatively easy to debug... The cons are: you need a whole separate process for each and every connecting client, which leads to a lot of overhead; your OS limits on number of simultaneous processes will come into play to effectively limit your number of simultaneous clients; and, other limits may come into effect even sooner (eg: you may run out of memory, with all the separate server processes running, before you run into the limit on number of processes); and it's generally pretty slow and innefficient to fork() a whole new process on most OS's... (The exception to the last con being Linux, which employs "copy-on-write" to GREATLY improve the efficiency of fork()'ing a child process... It's as efficient as creating a new thread... But, under most OS's that's not the case...)

The pros of the "pre-fork() a pool of children to handle incoming clients" are: each client still has their own separate process, so all of the above pros apply; but, it's much more efficient than the above, because all the process creation time is done up front, and is limited to however many you choose as the max... The cons are: you need to be able to signal the children processes that they should service a given client, which means the added complexity of something like semaphores and shared memory (or, each child must try to do accept() on the main listening socket itself, and grab them as they come; which introduces the stampede effect, and lends itself to fairly poor distribution of clients among available servers); and, you are still really limited to a fixed number of simultaneous clients, so you MAY see some clients forced to wait, if you exhaust your pool; and, the added complexity of course makes things harder to understand, code, and debug...

The pros of the "create a new thread as clients connect" approach are: almost all of the same ones as for the first one, but it's also much more efficient (on most OS's) to create a new thread than it is to fork() a whole new process... The cons are: you lose the "crash

protection" advantage of a separate process (ie: if a client can crash one thread, it'll generally take down all other related threads, too); and, multi-threaded apps are notoriously hard to understand, code, and debug, too...

The pros of the "pre-create a pool of threads to handle incoming clients" approach are: almost all of the same ones as for the "pre-fork()" approach, but you get the added efficiency of threads; plus, it's actually easier for threads to intercommunicate and share data (eg: file descriptors) than it is for separate processes, since they are designed for sharing data... The cons are: the same as for the above normal multi-threaded case, plus the added complexity of inter-thread communication/data-sharing...

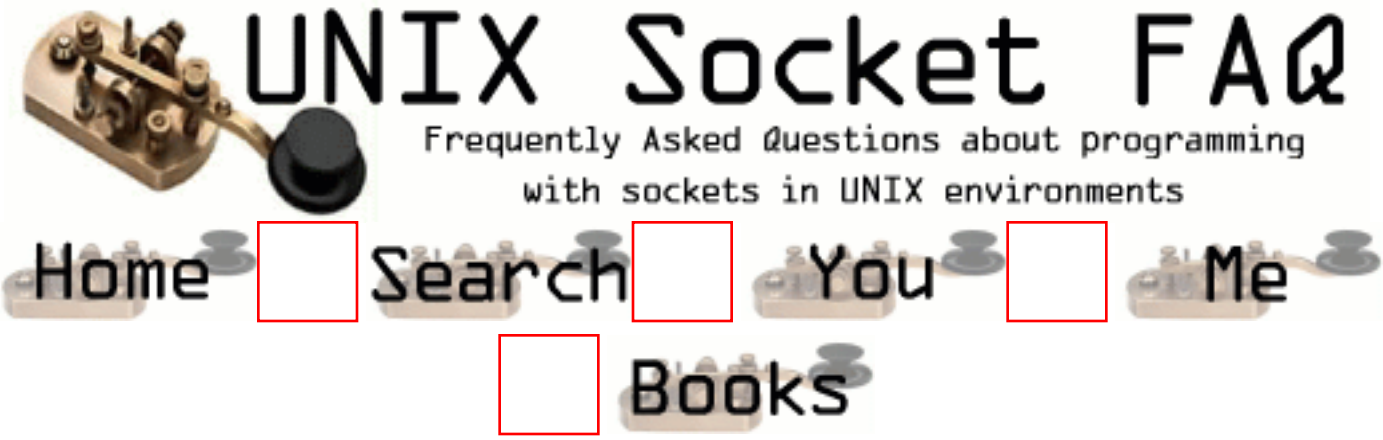
Basically, I've roughly listed the above in order from "easiest, but least efficient" to "hardest, but most efficient"... Which you choose really depends on your specific needs... I've seen legitimate cases for any and all of the above... (As well as the "single server which handles all clients" approach, which I outlined earlier, too...)

From: Justin Flude

Just to add my two cents ...

I believe that Apache can be built to either fork a child process to process an HTTP request or create a thread to do so. As I understand it, the architecture is kept flexible because threads are a relatively new addition to Unices, and not all versions support eg. the POSIX model, while fork() is very old and is universally portable. I don't dispute that spinning off a new thread for each accepted connection is faster than forking a new child - even on Linux - but the significance of this really depends on what the server serves up to its clients. A (traditional) HTTP server closes its connection as soon as it has fulfilled the client's request to be sent a page, hence the business of establishing the connection takes up a considerable part of the total time to process a request. But with a different kind of server, eg. one serving up long-lasting TELNET connections (I'm thinking of MTREK :), the overhead of establishing the connection is miniscule in comparison to the total time that the client is connected. In this case, it seems to me, a traditional forking server is much more attractive.

Add a Comment



New Questions: Client and Servers.

From: Aman

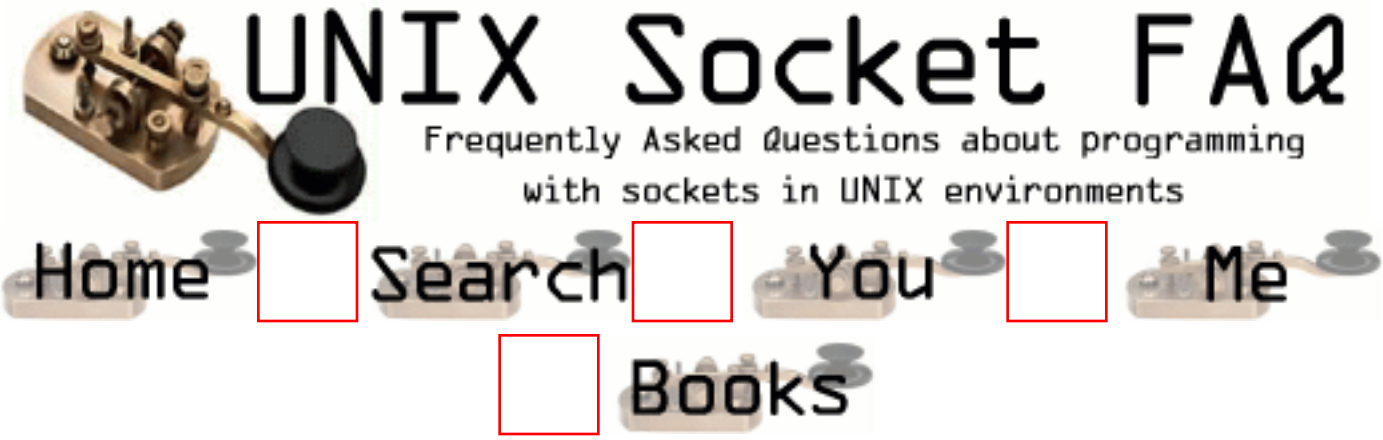
Hi,

Could you please direct me as to how the following can be done.

I want to have a server that listens at a port X for some connection from a client. Once the client is connected this server should start another server that talks with this client at yet another port Y.

Aman.

Add a Comment



New Questions: SOCKETS AND IMAP SERVER

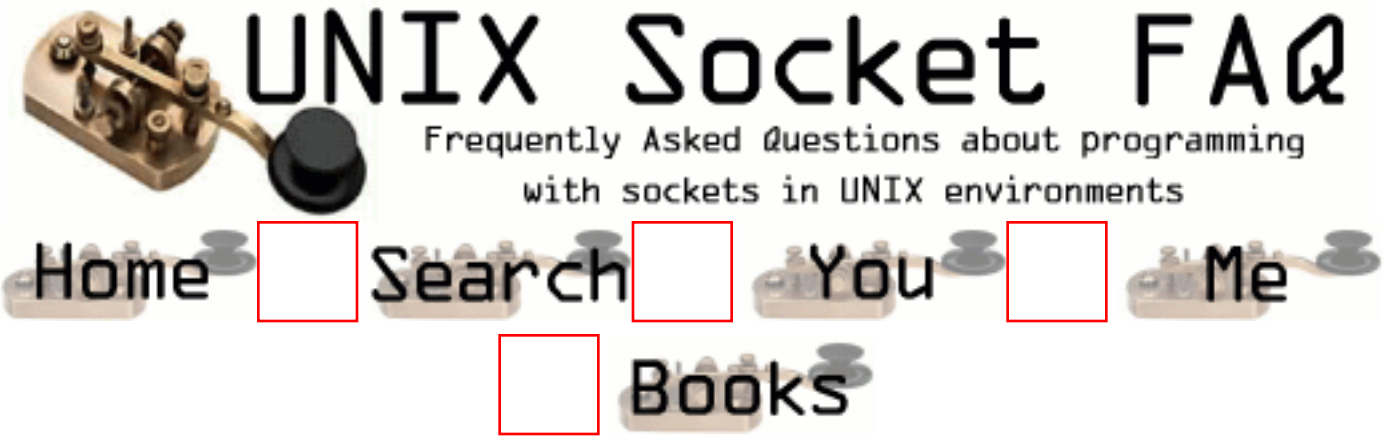
From: Marty

Hi,

I have a problem. I have a client program connecting to an IMAP server on port 143. I can log-on to the server but cannot send anything thereafter. In fact, once connected any SEND or RECV still has the original response from the connect command in the buffer of the RECV command. Very Strange!!! Can anyone help.

Thanks
Marty

Add a Comment



New Questions: client/server to check a file in a dir

From: Tomaz

Hi.. i am trying to write a client and server program,in which i want to check if a file exists in a directory,if it exists it returns ok,if not returns an error.can somebody help me how to start writing this program thanks in advance.

From: datawar

by writting the program for you it will only do bad, but
i can give u some hints on how it can work:

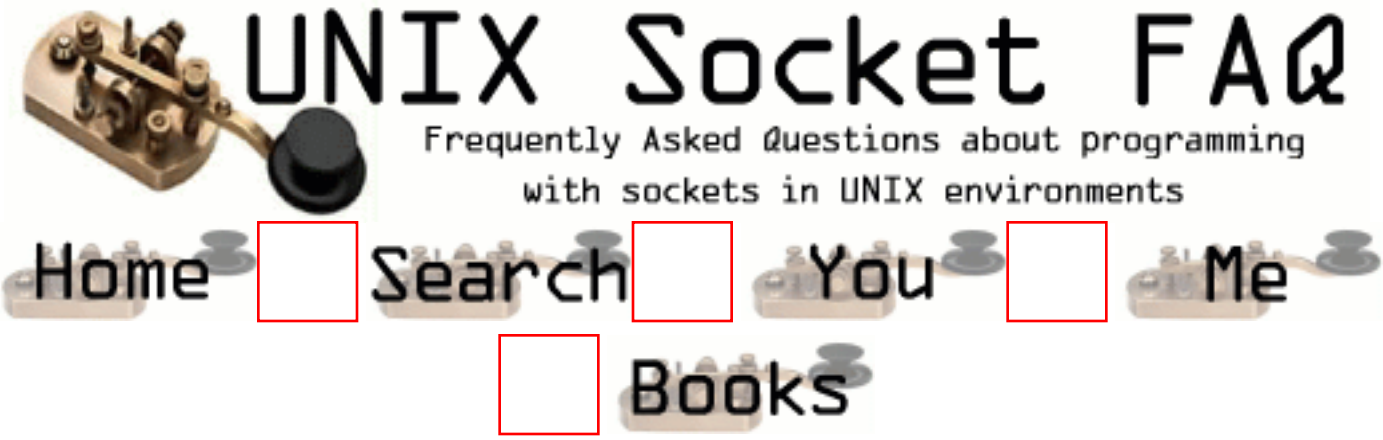
Server
---
bind a socket to a port and start listening, then u can
detect the msg file with a recv() call from the client
and do a stat() on the dir requested (use chdir() for that).
stat() will do the work for you.

Client

Connect to the server on the listening port and just
send the filename with send() the server will detect
and send you back a reply, u can use select() to check
if the server responds.

--datawar

Add a Comment



New Questions: recv() - the delay between recv()s

From: Erik Mattsson

Ive created a test program that opens up a tcp stream socket
and sends a few messages over it. Then later down in the program I
I receive the sent messages. BUT sometime there is this small
0.1 secs delay in recv(). The strange thing is that it seems
only to occur for smaller messages (<3000) but not for larger
messages.

0.1 secs are a way to long time when sending messages between
localhosts. Why is there the 0.1 secs delay sometimes, and
is there ANY way of removing it ? (kernel/etc?)

Testoutput:

As you can see sometimes it doesnt fill the buffer (row 2), and
another recv() must be called. (there is no problem with
the send()). You can also see the delay in the third recv()
BUT since the message is sent in the same thread, then it should
not be any problem of recv() a full buffer. Any idea why ?

Receiving. Size to read : 1024 Actual read size :1024 Total size read :1024 Total size :2833 time=2.5e-05
Receiving. Size to read : 1024 Actual read size :436 Total size read :1460 Total size :2833 time=3.7e-05
Receiving. Size to read : 1024 Actual read size :1024 Total size read :2484 Total size :2833 time=0.098295
Receiving. Size to read : 349 Actual read size :349 Total size read :2833 Total size :2833 time=8.2e-05

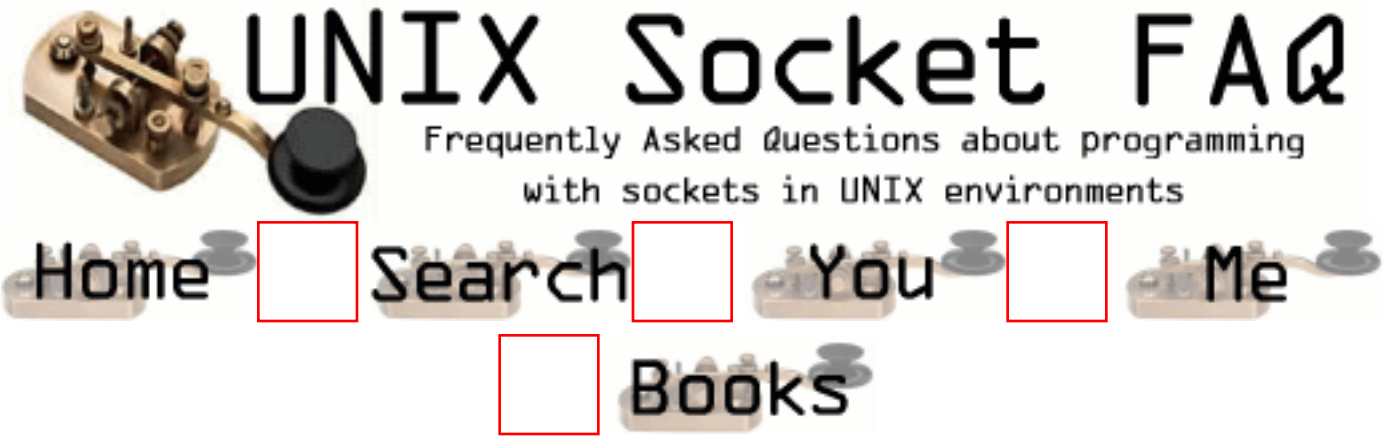
Code:
::recv(m_socket, (char *) buf, size, 0);

I tried to change the flag to MSG_WAITALL, BUT then the delay showed up some other place. So that didnt solve my problem.

Any ideas or suggestions ?

Im runnning on a FreeBsd 4.1, using g++.

Add a Comment



New Questions: OOB msgs

From: datawar

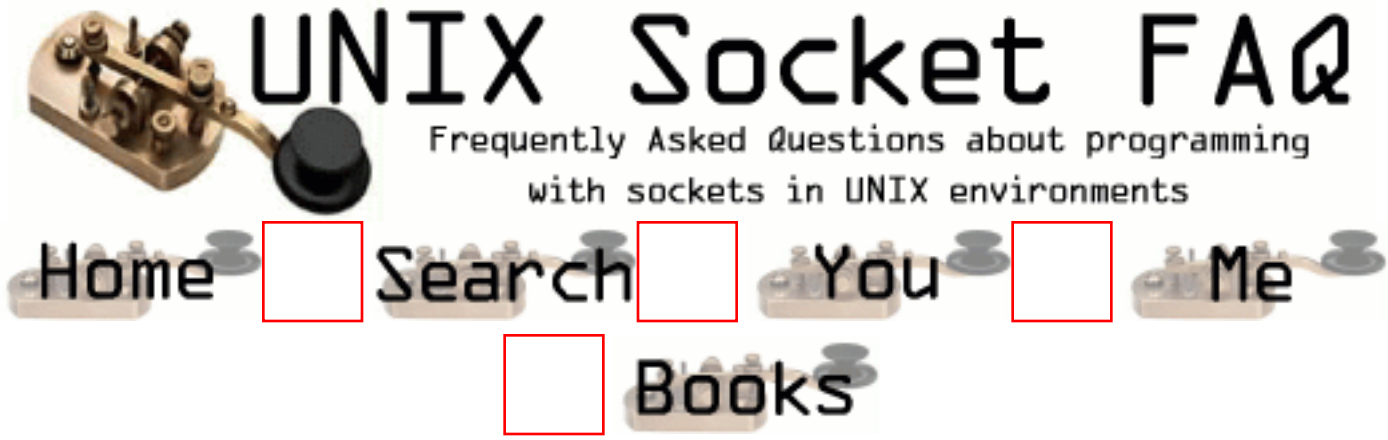
Can someone give me a small loop that will detect which
msg is type OOB and which is normal. I tried doing it with
select() by setting the exception set but for some reason
it didnt work.

Thanks

From: datawar

I figure it out how it can be done with select() I am trying
to get it work with URG signals now.

Add a Comment



New Questions: timeout on send with SO_SNDTIMEO??

From: ornaz

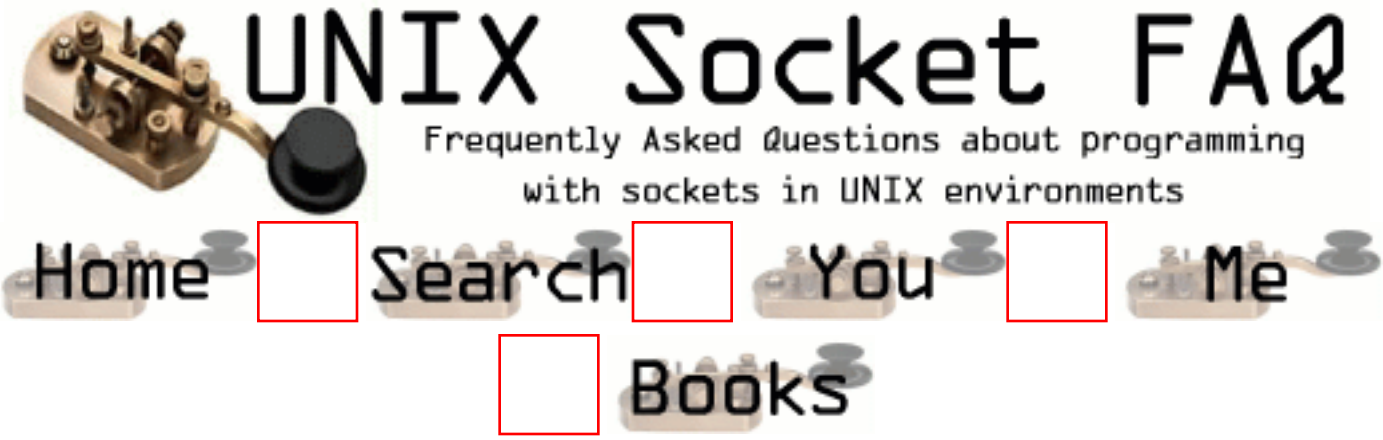
I want to use the socket-level timeout settings,
via setsockopt() with SO_SNDTIMEO,
but how do i know about the timeout?
is the select
  or the recv
 wake up on some signal ?
if yes which ?
and are both operations(select and recv) catch this signal?
is the time i set reinitlize per each i/o operation like if i recv somthing
or in my next send?

waiting for a replay
tnx in advance
orna

From: ornaz

addition to the above questions
i have another question
if the errno is EWOULDBLOCK after getting socket timeout
what can i do because my recv is nonblocking?

Add a Comment



New Questions: Server-Client Design

From: Lonnie Cumberland

Hello,

I am looking for a small example code for a server running
as a daemon and a client system.

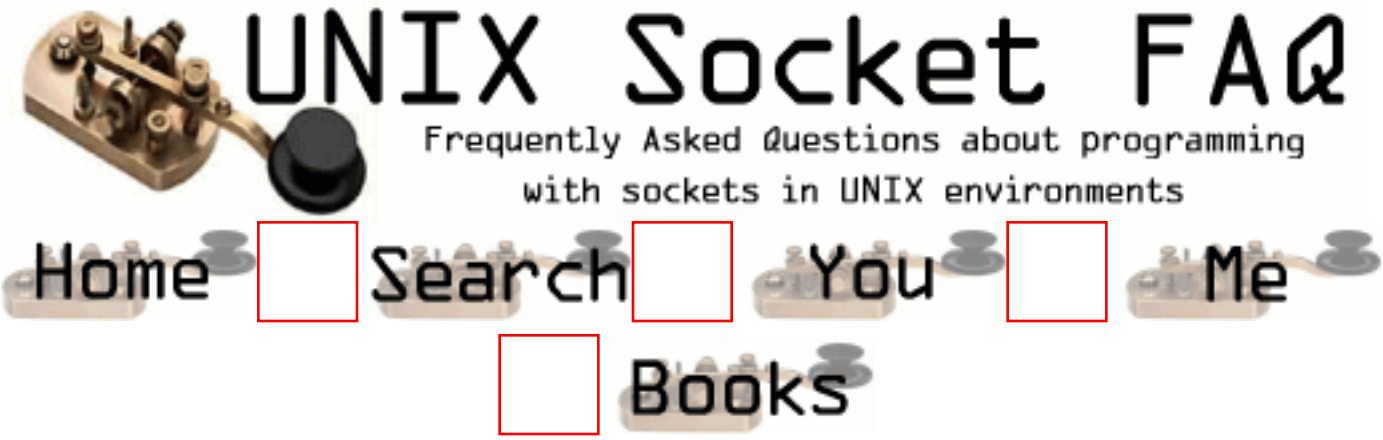
What I am trying to do is this.

I have a few servers which will be running this daemon and a
master machine which will receive an initial connection from
a user.

I want to be able to send the IP address of the user to each
of the servers and have them do a ping to that user and return
the time to me on the master server. Then based upon the fastest
ping time, the user will be re-directed to that particular
server.

Any ideas on how to best accomplish this task?
Regards,
Lonnie

Add a Comment



New Questions: CLOSE_WAIT Error

From: Yogesh Malik

Hi,

i have a problem.

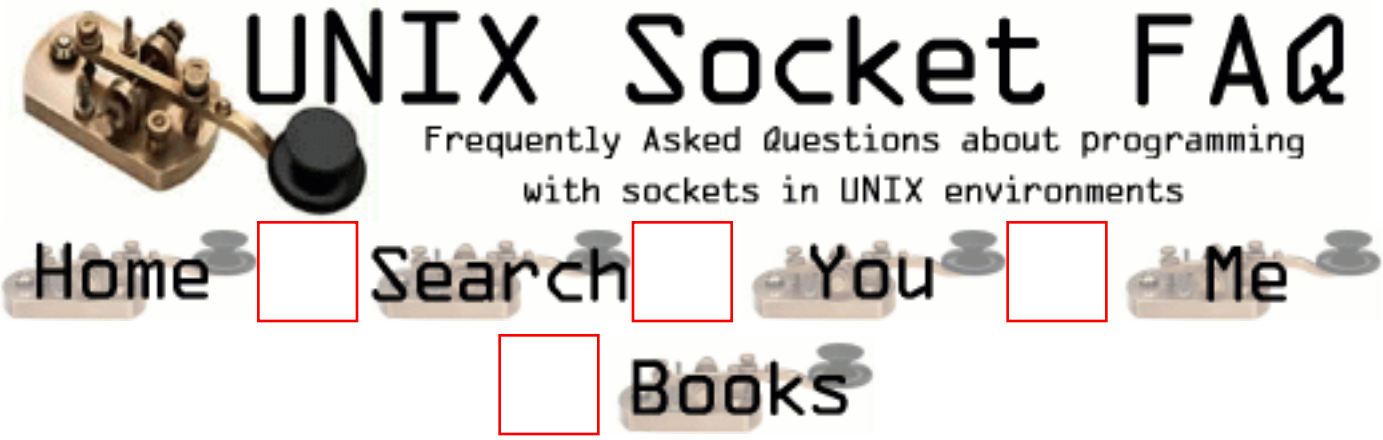
I have a nshttps server listening at port 80...it works fine until at some point of time sudennly all the port starts to go into CLOSE_WAIT state and with time number of ports in CLOSE_WAIT state increase until i recycle the server process...can anyone giv me clue tothis problem execpt recycling the server process

thanks
YM

From: Wen Hu

I am having the exact problem here. Can someone please help?

Add a Comment



New Questions: SO_RCVTIMEO question

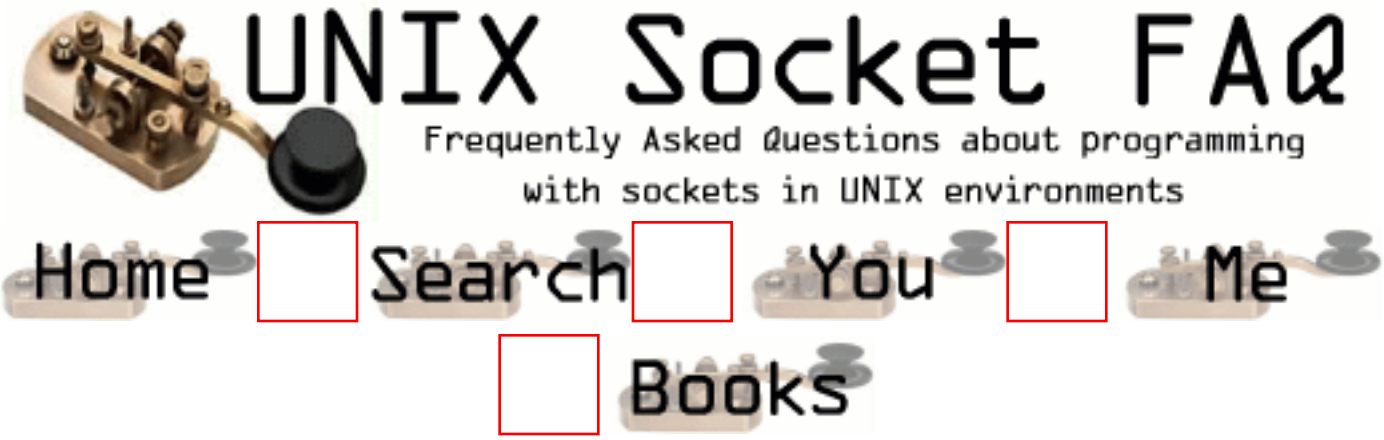
From: ornaz

I want to use the socket-level timeout settings,
via setsockopt() with SO_RCVTIMEO,
but how do i know about the timeout?
is the select
  or the recv
 wake up on some signal ?
if yes which ?
and are both operations(select and recv) catch this signal?
is the time i set reinitlize per each i/o operation like if in my next recv?

if the errno is EWOULDBLOCK after getting socket timeout
what can i do in case my recv is nonblocking?

tnx
Orna

Add a Comment



New Questions: Why does Apache only accept a message after the socket closes?

From: Donncha

Hi there,
I am a complete newbie to sockets although questios on this site and other have helped me along a lot. My task is to run a CGI script on a remote computer from a C++ program. Normally the CGI is run from a webpage, but I need to interface directly (it's part of the product I'm working on). So, my plan is to connect to the webserver port, and send a HTTP request to the script (GET or POST operation) to run it with the appropriate data. I'm testing my program with a plain Apache webserver, but it seems as though Apache only processes the message after the socket is closed (by which time the client has timed out). Is this how webservers usually work (I've tried my client with a simple example server and it works fine, not timing out, server processes message and replies immediately) or is there something wrong with my socket:
sock = socket(AF_INET, SOCK_STREAM, 0);

Thanks for any replies, sorry if this is an excessively newbie question :)

Donncha Mac Craith

From: Donncha

Aaargh!
I can't believe how stupid my mistake was!
Here's my command line for running the little client program (it's trying to download my home page from the machine called selma)

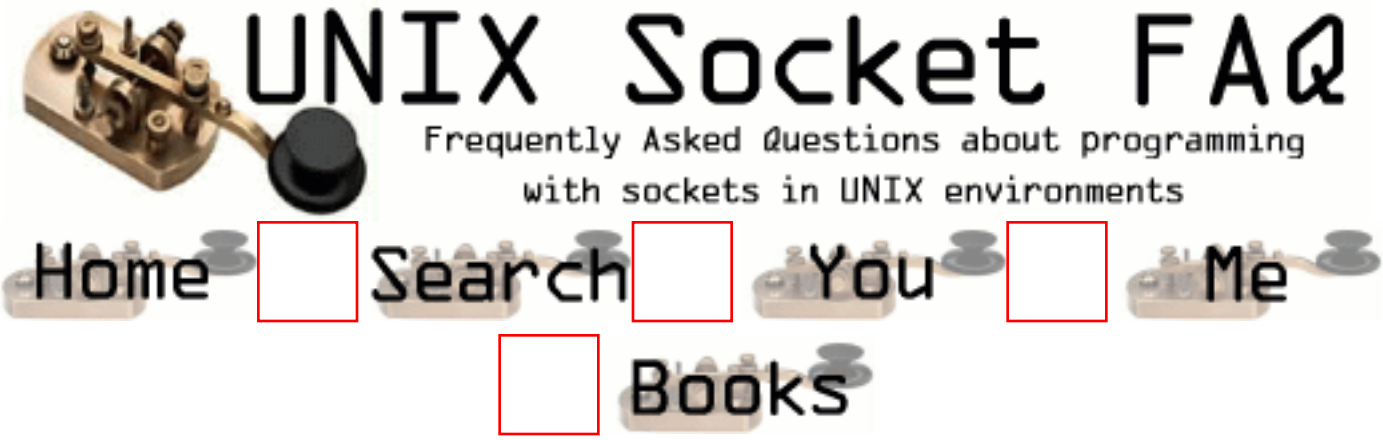
a.out -h selma -p 80 "GET /~dmccraith/index.html HTTP/1.0"

Spot what's wrong with the GET command - it's not followed by two line feeds/carriage returns! Needless to say, you just leave out the last quote, hit <return> three or four times, the put in the last quote (you're basically adding the LFs by hand) and then it works fine, spitting back the raw

HTML page.
Anyway, that's sorted that out, I'll just put this here in case any other struggling socket programmers run into the same thing!

Happy New Year,
Donncha

Add a Comment



New Questions: How can I Receive or Deal with Out-of-Band Data

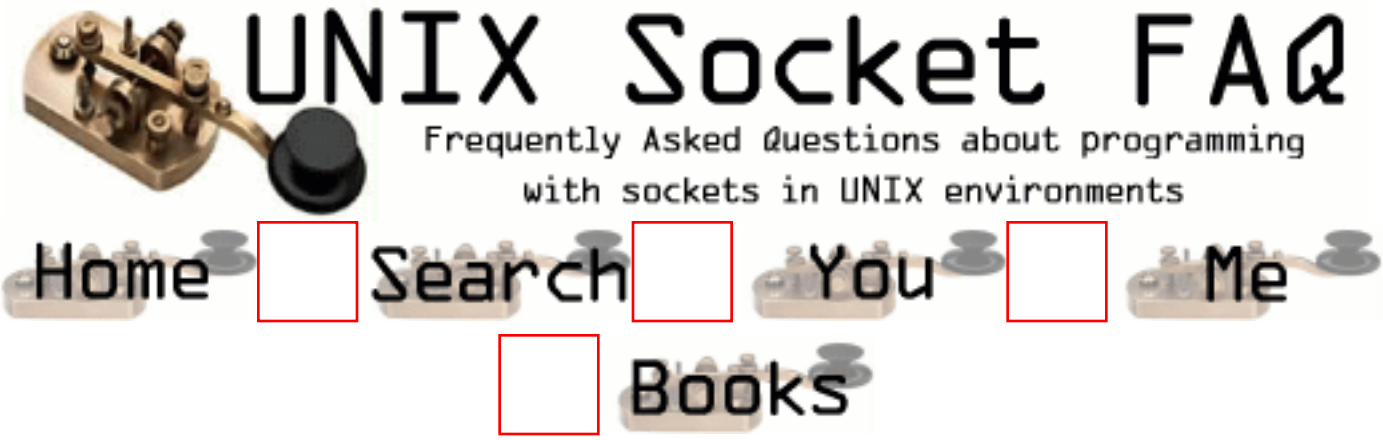
From: Krish

I know Out-of-Band data to the application is a seperate
stream which is ('urgent Data') and we have to deal with
it properly. I don't Know When I receive Out of Band Data
and How to deal with it ? Please any one tell me.
Thanks -Krish

From: datawar

1)SIGURG
2)select
3)look the stream for the OOB mark

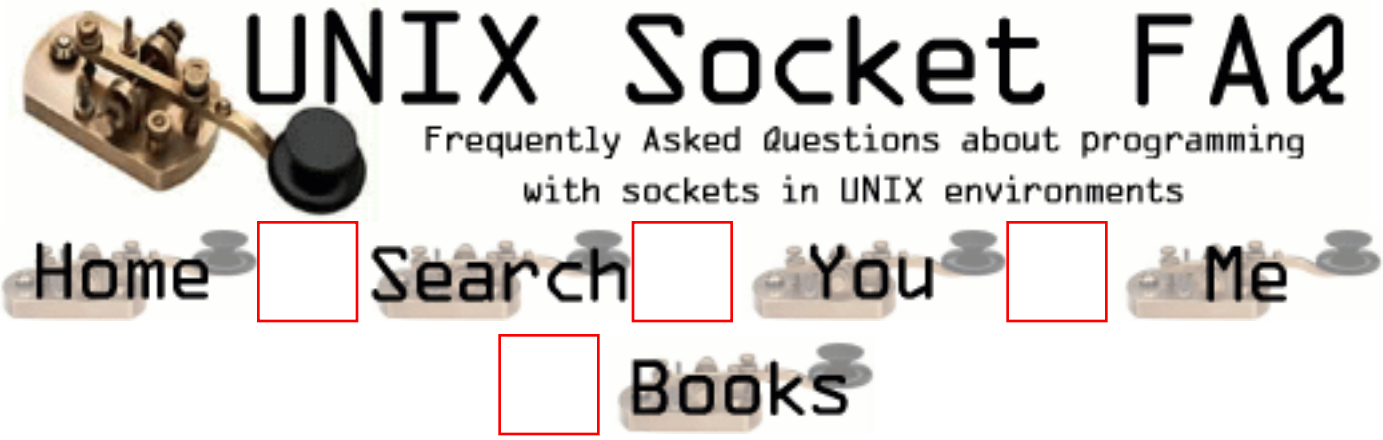
Add a Comment



New Questions: SIGURG & OOB problem ?

From: datawar

Add a Comment



New Questions: SIGURG & OOB problem ?

From: datawar

is there a way to find which socket has sent an OOB
message causing the URG signal to be set. On a normal
case when we deal with only 1 socket its easy but if
we got a list being monitored and the URG signal is
set there is no way to detect which end sent the OOB msg ?

From: Hector Lasso

Hi,

I've worked with OOB data in X.25 domain sockets, but only with one socket per process...
I suggest you try the following:
1) Place the sockets in non-blocking mode
2) When the SIGURG signal is caught, use the recv() function with MSG_OOB and check to see if the return value is not -1... I guess if there is no OOB data the recv() would return -1 and errno would be set to EAGAIN or EWOULDBLOCK...

If it works please let me know

Good luck

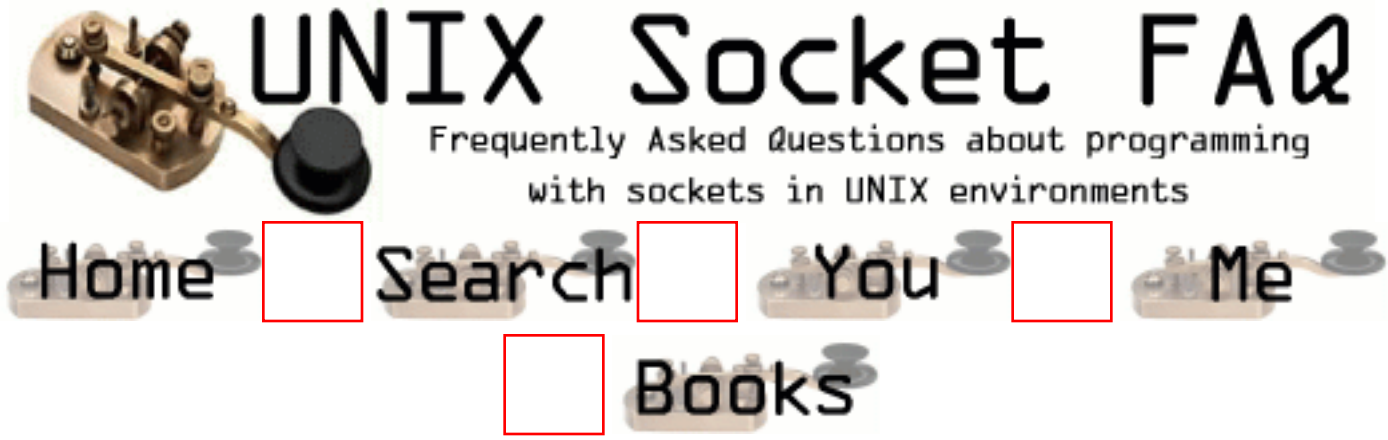
From: datawar

Thanx for answering to my question on SIGURG & OOB. Since then
I have been working on this problem and I came around the following:
You set a signal handler for SIGURG and when its set it means that
an OOB msg has been received. To deal with a set of sockets i simply
execute select() on the signal handler to see which socket has sent
the msg. I also didnt include the fcntl() call to set the proc. id.
owner to each new socket. Another way i came around it was to make

a new set on select() the exception set and check each socket if it has changed the exception set changes status on OOB msgs. I havent tried your way yet but it seems that its going to work with a small modification that i will need to do a recv() on all the currect sockets since the server might serve more than 1 clients.

Thanks,
DatawaR

Add a Comment



New Questions: Broadcast and receive DHCP messages

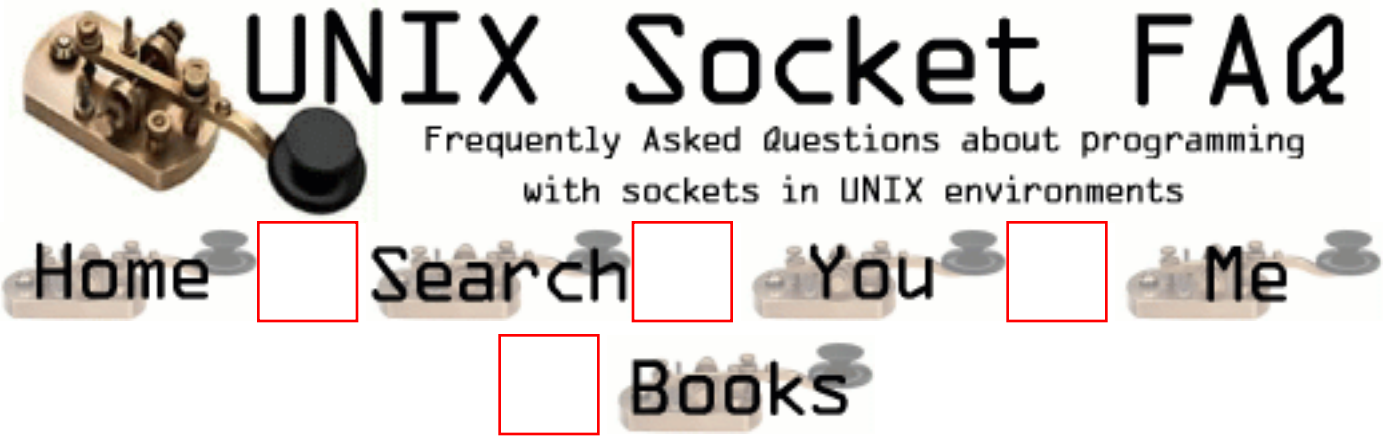
From: Zhang Wei

I am a beginner of socket programming. I want to write a "partial" DHCP code so that I can broadcast a DHCPDISCOVER and receive a DHCPOFFER message because I want to extract information from the DHCPOFFER message. I know that DHCP works on UDP and it must be able to broadcast and receive messages even without an IP. That's where I encounter problems. I tried writing a code using a normal UDP socket. Although it can send("sendto") a broadcast message when the interface has an IP, it cannot send messages without an IP, and it can never receive("recvfrom") the message from DHCP server. (I think it's because the message is not sent to the address of my interface that is bound by the kernel)
I just want to have some idea of how I can send a broadcast message(with port number specified, e.g. 67) even without an IP, and how I can receive a reply that is directed to my hardware address? Shall I use raw socket? Or datalink access? Can someone just give me some idea of what is the right thing to do? I work on Linux. Thank!

From: Nils O. Selåsdal

I suggest you get the source code for an existing dhcp
client and look. Source code for e.g. pump is pretty small
and you should find it at www.freshmeat.net or www.rpmfind.net.

Add a Comment



New Questions: What is Meant By RFCxxxx ?

From: S.Murali Krishna

In Many places of this FAQ Iam often encountering one
such word like RFCxxxx , See RFC 1738 ,etc..
So What is meant by RFC and Where we can see that.
Please any one tell me Detail about that.
Thanks -Krish

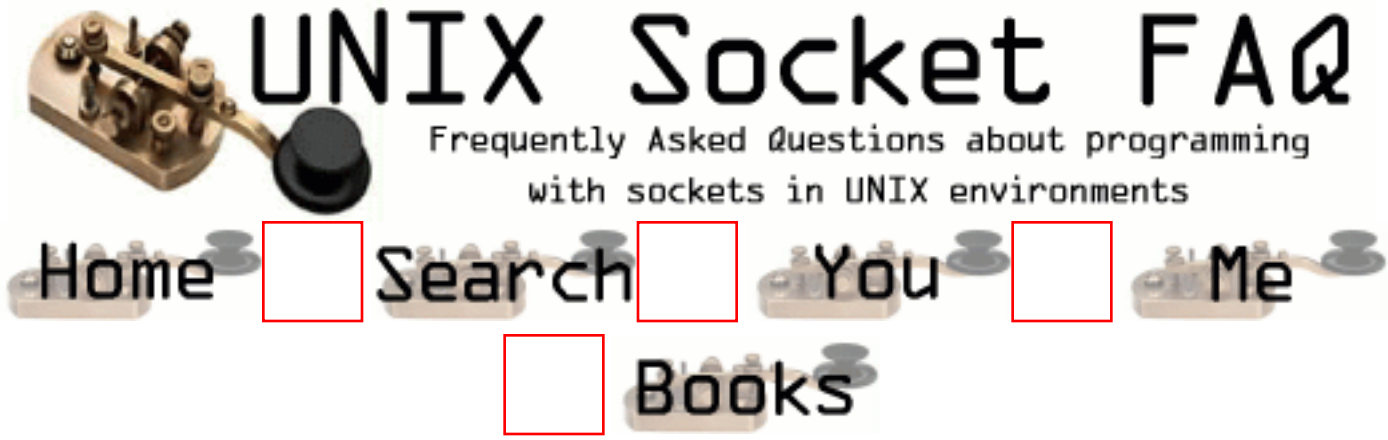
From: Rob Seace

RFC = Request For Comments... They are just documents written up by various people, describing certain Internet standards/protocols/etc... They are generally looked upon as the definitive guides to how things should officially be done... You can find all of them online many places... Faqs.org has a good RFC section... The IETF is probably the most official source... But, you can find them all over the place...

From: I gwana

porn

Add a Comment

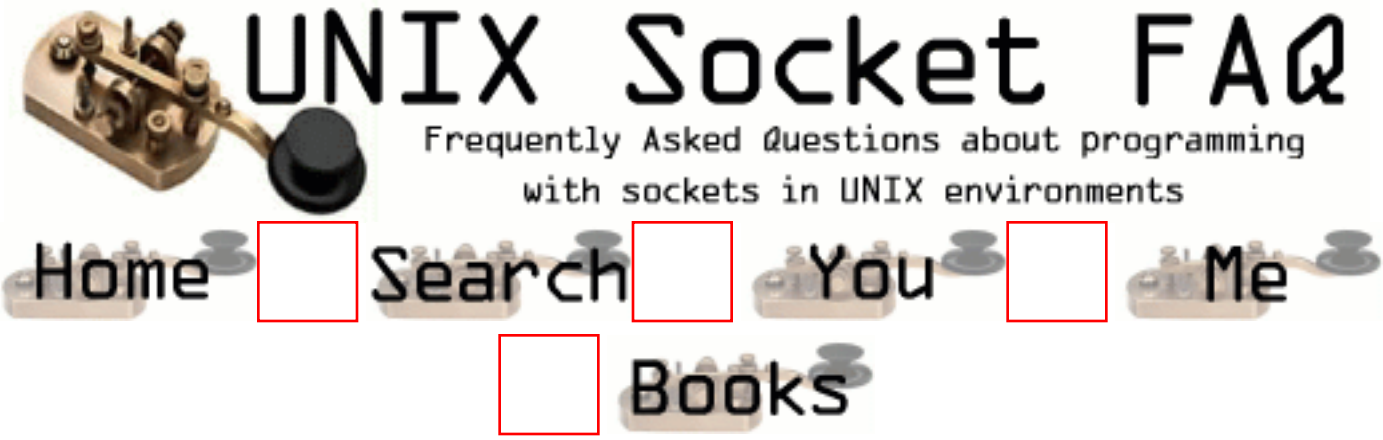


New Questions: How we can Get Mails From Servers Using Sockets.

From: S.Murali Krishna

The Internet Domain socket consists of ADDRESS and PORT
So Once we have the domain name and mail Port. (25)
Can't we access our mails from another servers ?
If Yes Means Please tell me some clue to do that.
If no Means Please specify the reason clearly.

Add a Comment



New Questions: How we can Get Mails From Servers Using Sockets.

From: S.Murali Krishna

The Internet Domain socket consists of ADDRESS and PORT
So Once we have the domain name and mail Port. (25)
Can't we access our mails from another servers ?
If Yes Means Please tell me some clue to do that.
If no Means Please specify the reason clearly.
Thanks -Krish

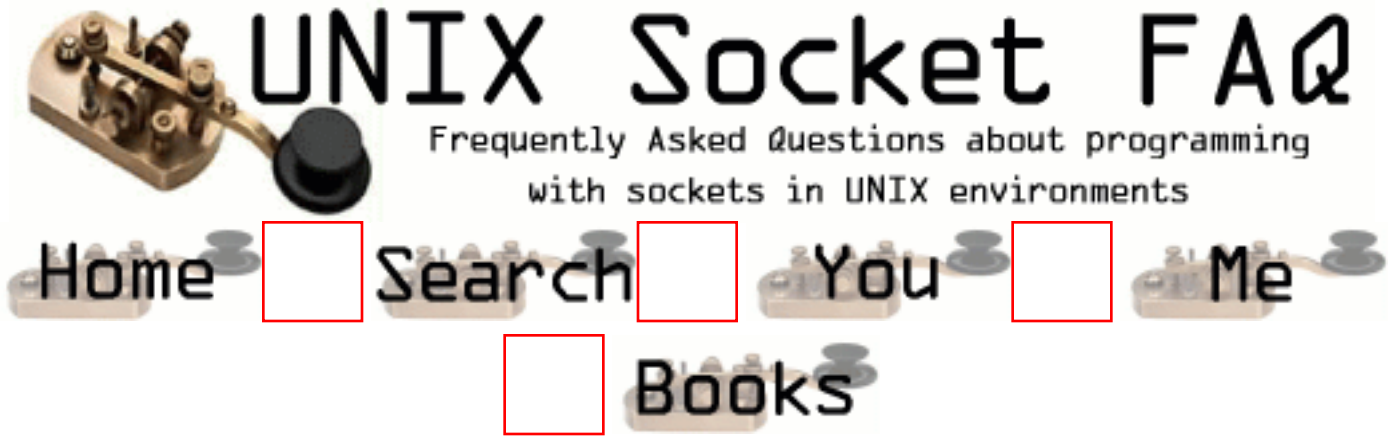
From: Mahafuz

With mail server IP address and Port you can send/recive mail. Thats how its done.

First you need to know about POP3 and SMTP protocol.
After that you connect to mail server and do the things
need by the mail server . Basically to send mail you'll
need to follow POP3 protocol. And to send mail you'll
need to follow SMTP protocol.

Pls see RFCs for these protocols and you'll know what to do.

Add a Comment



New Questions: Can someone answer to that ??!!

From: ornaz

I want to use the socket-level timeout settings,
via setsockopt() with SO_RCVTIMEO,
but how do i know about the timeout?
is the select
or the recv
wake up on some signal ?
if yes which ?
and are both operations(select and recv) catch this signal?
is the time i set reinitlize per each i/o operation like if in my next recv?

if the errno is EWOULDBLOCK after getting socket timeout
what can i do in case my recv is nonblocking?

tnx
Orna

From: Mahafuz

As far as i understood,
according to richard Stevens book - "Unix Network Programming" the socket option
SO_RECVTIMEO &
SO_SNDTIMEO are meaningless. I mean they don't
do anything ("don't currently affects timers").

Is this true??

From: Rob Seace

Well, I don't know if it's universally true across all OS's,
but *I* certainly would never rely on SO_{SND,RCV}TIMEO to
impliment my timeouts... Better to just do it yourself with
alarm() or select() and non-blocking sockets... Because,
there certainly are systems where those socket options don't
do anything... (I don't have my copy of Stevens with me at
the moment, so I can't see exactly what he says about it...
But, if he really says they aren't implemented anywhere,
then I'd probably believe him... ;-)) Just taking a quick
look at the 2.2.x Linux kernel source, it looks to me like
at least Linux doesn't implement those options, anyway...
(Rather, they exist, but getsockopt() returns 0's, and
setsockopt() fails with ENOPROTOOPT...)

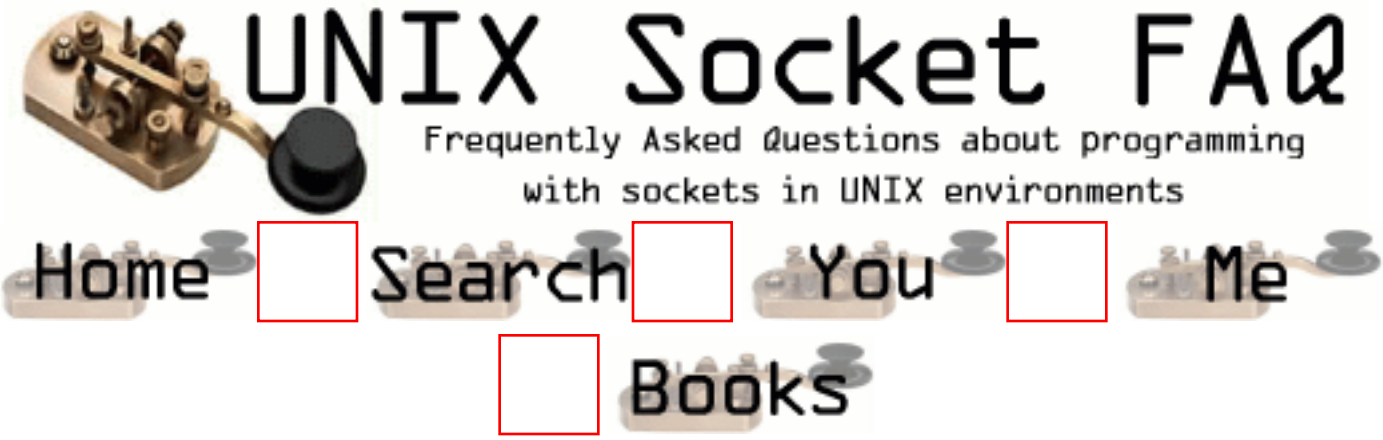
From: Hector Lasso

Hi,

The timeouts affected by these two options "... are fixed to a protocol specific setting in Linux and cannot be read or written. ..." according to the man page in Linux.

I would check with your OS's manual pages...
I prefer to use select() or alarm() when i don't want to write too much code...

Add a Comment



New Questions: select statement times out when there is data

From: David Mack

I have a program that for proxy program that for some reason will timeout. I am using FD_ISSET to check the file descriptors to see which ones have data and I expect them to have data but the timeout still happens on the select. It's almost like it's stalled for some reason. Any help is appreciated.

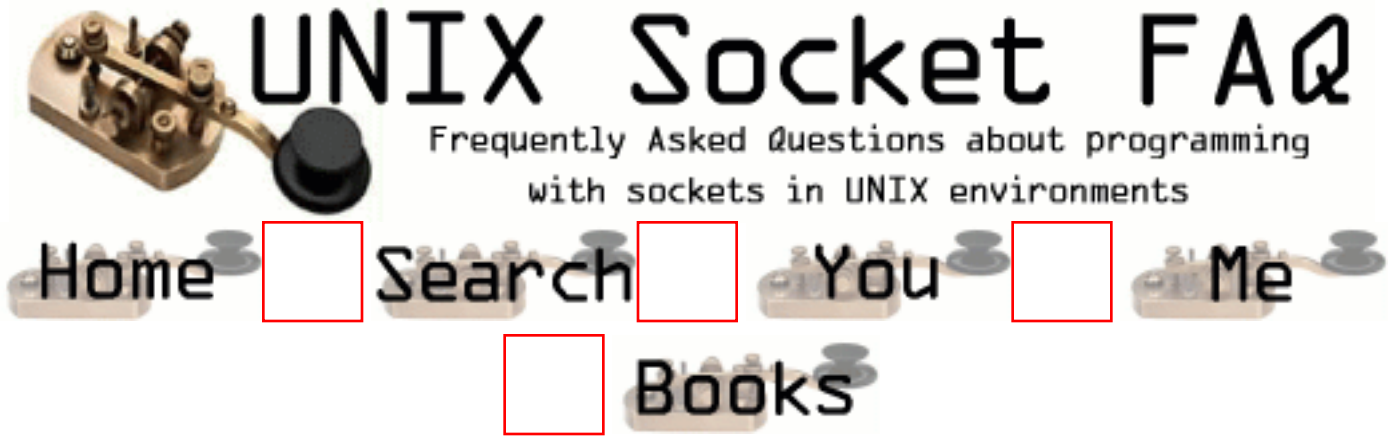
From: Hector

Post the code

From: ancienthero

Maybe you need to check the first parameter in select, it should be the highest number of all file descriptors in every fd_sets.

Add a Comment



New Questions: sending File Descriptors across Unix Sockets

From: Andy Bitton

I am trying to send an open file descriptor from one process to another over a Unix domain socket.

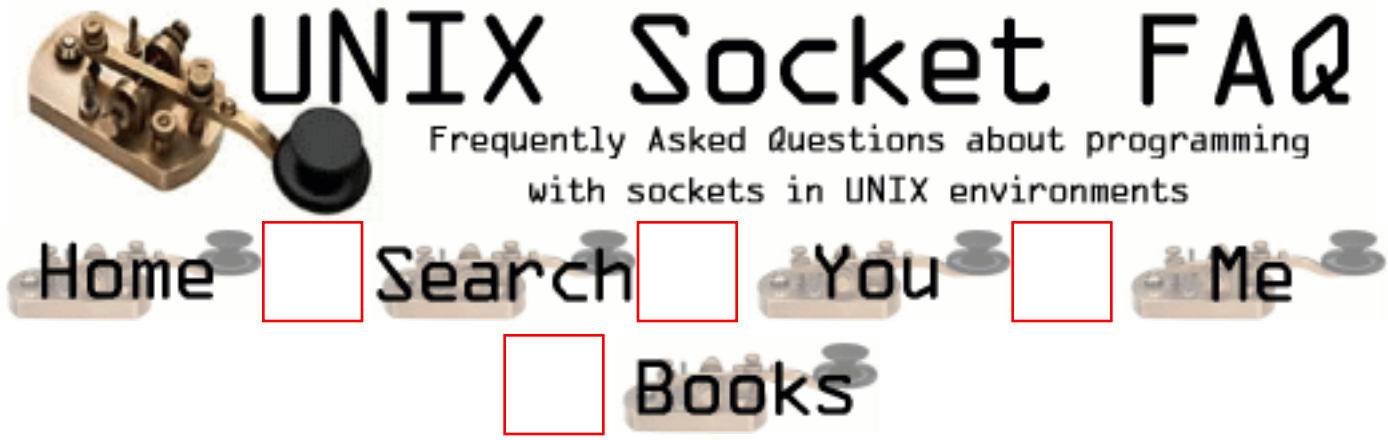
I am using sendmsg/recvmsg functions to do this.

The problem is that the sendmsg call fails with a EBADF error even though the socket will send and receive data before and after the call.

I am implemting this under SCO open server 5.05.

Any ideas ??

Add a Comment



New Questions: Server servicing multiple sockets (ports)

From: Dave Zavetz

I am writing an application in which a single server needs to service 5 different sockets. Here's the catch - each socket must reside on it's own unique port. I'm sure that select() is involved somehow or other but could someone give me a clue on how to get started.

Also, to further complicate matters, the server (perhaps via a 2nd application) needs to monitor a 6th socket (port) for SNMP (UDP) traffic.

Dave

From: Hector Lasso

Hello,

I think you need to create 5 different sockets, bind each one to a different port and then listen on every one of them using "select()" to know which one is receiving a new connection (the file descriptor is set in readfds).

I still don't know why you need 5 different ports as one port can do the trick -> accept 5 connections on the same port.

Best regards

Add a Comment

UNIX Socket FAQ

Frequently Asked Questions about programming with sockets in UNIX environments

New Questions: How to use "errorfds

From: Mark Johnson

Add a Comment

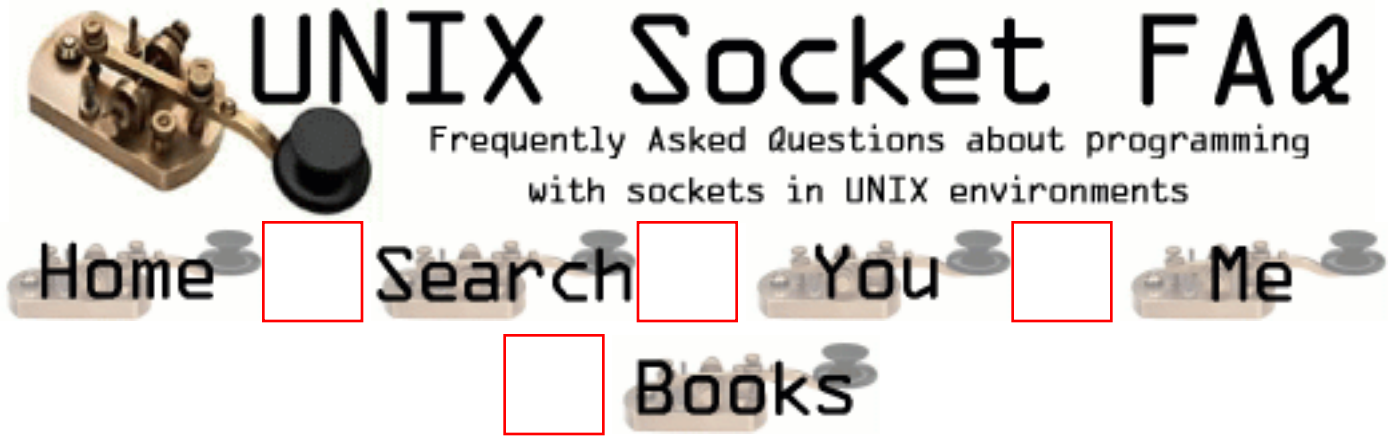
UNIX Socket FAQ

Frequently Asked Questions about programming with sockets in UNIX environments

New Questions: How to use "errorfds

From: Mark Johnson

Add a Comment



New Questions: How to use "errorfds" in select()

From: Mark Johnson

When select() returns -1 and error is set to EBADF
I need to figure out which socket descriptor (sd)
to clear from the fd_set. I don"t want to call recv()
on each sd at this moment because it will be in an
inapproriate place for the application. Instead, I
would like to use the "errorfds" argument of select()
to know which sd is the culprit. But I guess I do
not understand how this works because calling FD_ISSET()
on the returned "errorfds" always returns 1 when
select() returns EBADF:

```
fd_set read;
FD_ZERO(&read);

// set active sd"s for "read" set

fd_set errorfds;
FD_ZERO( &errorfds );
memcpy( &errorfds, &read, sizeof(fd_set) );

//
// i assume that select will modify the appropriate
// entries in "errorfds" to reflect pending error
//
int rv = select( nfds, &read, NULL, &errords, NULL );
if( rv == -1 )
{
```

```
  switch( errno )
  {
  case EBADF:
     // find the culprit
     for( int i = 0; i < nfds; ++i )
     {
        int flag = FD_ISSET(SdList[i], &errorfds);
        if( flag )
        {
           FD_CLR( SdList[i], read );
           // clean up SdList and bad sd...
        }
      }
break;
default:
// etc..
}
}
```

How do I use the "errorfds" correctly with select()?

From: Rob Seace

As far as I know, all of the fd_sets are undefined when select fails (returns -1), and you can't rely on their value... The "errorfds" are actually for catching socket errors/exceptions, not general errors like a bad file descriptor... A bad FD is just not something that should ever show up under normal circumstances, or ever have to normally be dealt with like you seem to want to deal with it... If you're seeing it, it probably indicates an error in your code, somewhere... And, looking at the code posted above, it looks to ME like you may be misunderstanding how select() and fd_sets actually work... You seem to reference an "SdList" array, which you index from 0 upto the "nfds" arg you pass to select(), and each element of which you seem to be treating as an FD... That just seems wrong to me... The "nfds" passed to select() is not actually supposed to be a count of how many FDs you are using; it's supposed to be set to one greater than the highest FD that you are using; which is a subtle, but important, difference... And, there is no need to maintain an array of FDs for anything; the "i" in your loop, counting up from 0 to "nfds" gives you the actual FDs you want to be checking for... (You'll

just want to keep around an extra source fd_set that holds
all your currently used FDs, and never gets passed to
select(), but is what you use to fill the "readfds" and
"errorfds"...) Without seeing all of your source in full,
I can't tell if maybe this all makes sense in context, but
based on the above snippet, I'd say it doesn't look right...

From: Hector Lasso

Hi,

I partially agree with Rob:
- When there is an error in the execution of select() it returns -1 and the sets and timeout value become undefined. -> You cannot use them for anything else. About the nfds parameter, Rob's right, it is one greater to the highest FD you are using...
- You need to keep the FDs in some kind of array or structure, because you could be using non-contiguos file descriptors, so Rob's approach won't work (for example the first three FDs are stdin, stdout & stderr). You should recreate your three sets before calling select() every time (it could be done with a loop and FD_SET() or just memcpy()ing from a set of the file descriptors. This set could be initialized as empty, and every time you receive or create a new connection you can add descriptors to it by using FD_SET() for that single descriptor, and everytime you detect disconnections just FD_CLEAR() the descriptor - this is the extra source fd_set Rob tells you about).

Good luck

From: Rob Seace

But, if you have the extra source fd_set, that serves the
same function as having them in an array... That's all I
was saying: you can make your life simpler, and save some
overhead, by replacing the array of FDs with just another
fd_set... Then, it's ready for easy memcpy()'ing into the
fd_sets you want to pass to select(), AND you can still do
any checking of it that you want to, to see which FDs from
0 to "nfds" are currently in use... My point was you don't
want to be treating "i" in the above loop as an array index,
but rather as a file descriptor, itself... (One which may
or may not actually be in use by you... But, if it's not in
use by you, FD_ISSET() won't return true, so you're all set
there...)

From: Hector Lasso

Rob,

That is a good one...

I just couldn't catch what you really meant...

From: frankv

As far as I can understand, the error set is used for other reasons - exception handling. Dont ask me what that is supposed to mean, but it is not going to tell you which fd is bad.

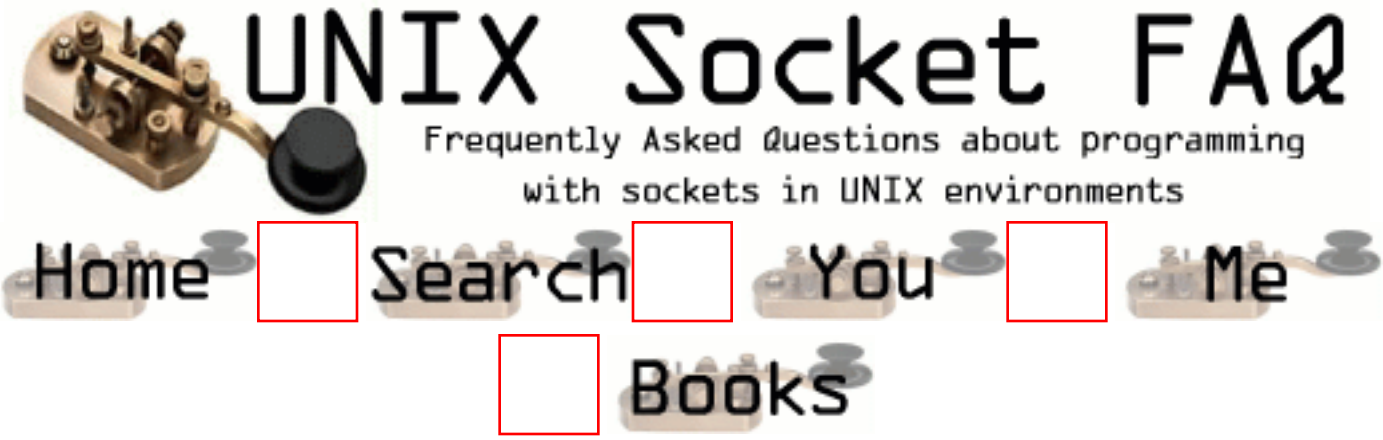
The only solution I have managed to come up with is to rip through all the fd's one by one using select (with zero timeout) until I have got the one causing an error - usually it means the socket has closed, so all you need to do is get rid of it. It works, but the method isn't very satisfactory. Does anyone know the proper way of finding out the culprit??

fv.

From: Rob Seace

I don't know what OS you people are using, but I really
don't think it should be normal behavior to be having
select() failing with EBADF (or, really, anything aside
from EINTR, and EAGAIN/EWOULDBLOCK for non-blocking FDs)...
It should NOT happen when the remote end of a socket gets
closed; I don't know where you get that from... The socket
should select as readable, in that case, and when you read
it, you'll get EOF (ie: no data to read)... No error
condition, at all... I still maintain that if you're
getting select() failing with EBADF, you have a bug in your
code somewhere... It's is NOT normal behavior, and you
shouldn't be looking for ways to kluge around it like this,
but rather tracking down the bug, so that you no longer get
this error in the first place... I mean, I suppose I can
understand it, if you're trying to design an absolutely
bullet-proof, mission-critical, "it can't fail even if the
server is on fire!" type of app... But, at the very least,
you should log the anomoly, so you can debug your code
later...

Add a Comment



New Questions: Select returns EBADF

From: Andra Fralin

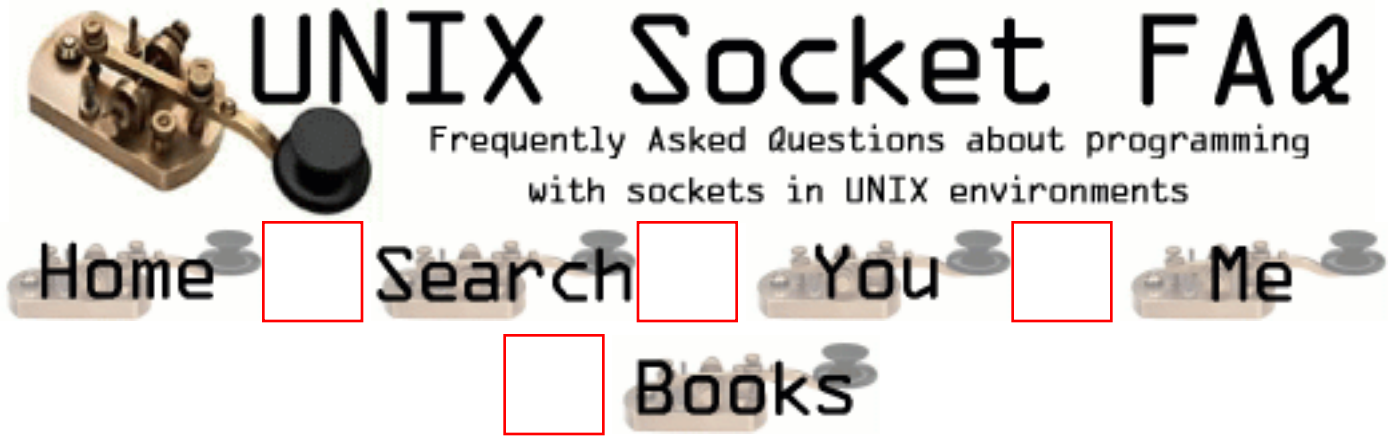
I have a server that runs fine most of the time however sometimes it returns EBADF on a select statement and I don't understand why. I have three file descriptors in one set and when I do select every once in a while select will return a number < 0 and then errno will equal EBADF. Any help is appreciated

---

From: Rob Seace

Is one of your FDs getting closed, but not removed from the
fd_set, maybe? It's difficult to guess what the problem
might be without seeing the code in question...

Add a Comment



New Questions: core dump on send/recv

From:

my application coredumps when sending/receiving data that exceeds the buffer size (definitely). how can the client/server receive/send indefinite amount of data through send()/recv(). need source code.

From: Hector Lasso

Hi,

You will have to use a buffer large enough to hold the data, or you will have to receive the data your buffer can hold, move it to another location/process the data and continue receiving in the same buffer.

The source code is straight-forward:

```
do {
  // Be sure you have data to read, or else the function
  // will block until data becomes available
  // Use select() for example
  r = recv(soc, buffer, buffer_capacity, 0);
  if (r > 0) {
    // Received r bytes, do soething with them
    do_something_with_the_data(buffer, r);
  }
  else if (r < 0) {
    // Error
    do_error_handling();
  }
  else {
    // Your socket has been closed...
    shutdown(soc,2);
```

```
    close(soc);
  }
} while (1);
```

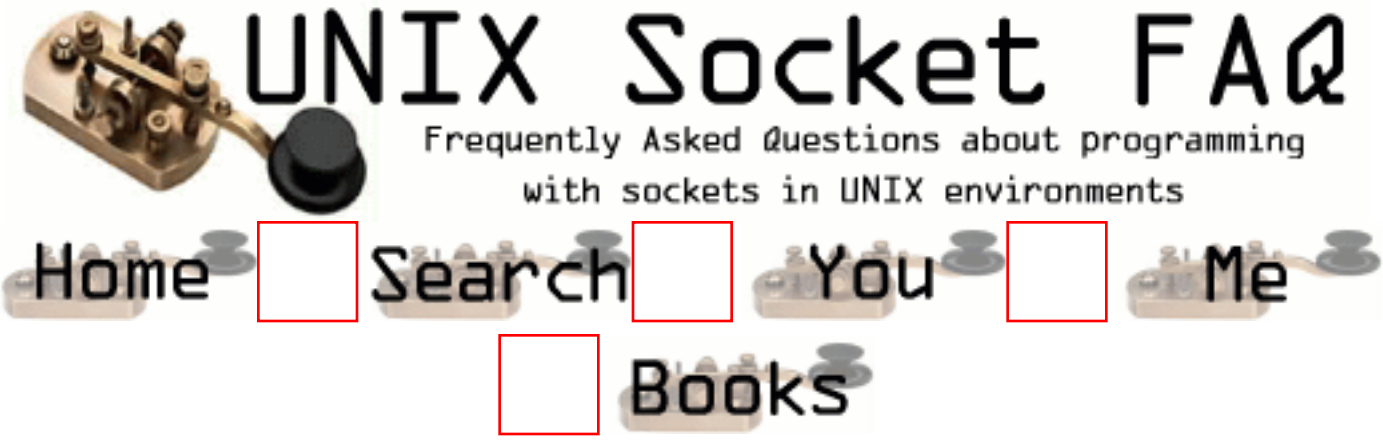
From:

thanks hector, i am now done with the sending part. but what if the receiver side does not know the capacity of the buffer. initially i set the buffer_size to 1024 (for the receiving side). but the sender sends more than 1024 and the receiver definitely will get only 1024. how can i the receive receive all the data coming from the sender?

From: Rob Seace

This same basic question is addressed in the main FAQ: question 3.8...

Add a Comment

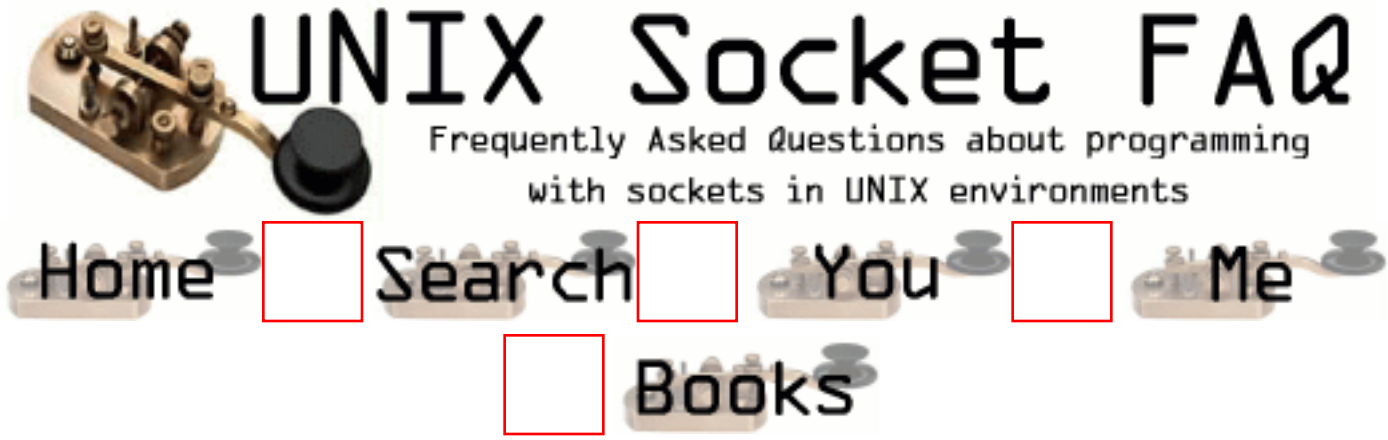


New Questions: socket programming

From: Mathin

How can we transfer data from winsock application to unix by C/C++ socket programming? Is there software about it? Please suggest me.

Add a Comment



New Questions: transfer data from winsock to unix by C/C++ socket programming

From: Mathin

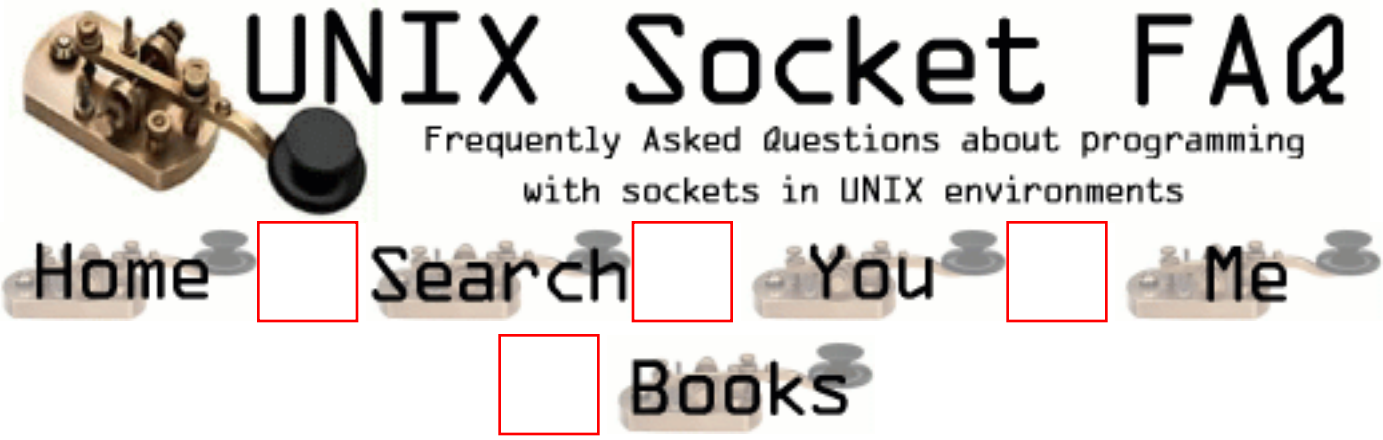
How can we transfer data from winsock application to unix by C/C++ socket programming? Is there software about it? Please suggest me.

From: Hector Lasso

Are you planning to code both sides (Unix and Winsock)?

Programming with Winsock is very much like programming sockets under C for Unix. Anyway, both sides can communicate regardless of the interface used to program, what is important is that they use the same protocol, and that data is exchanged in a consistent format.

Add a Comment



New Questions: NT Socket to Unix

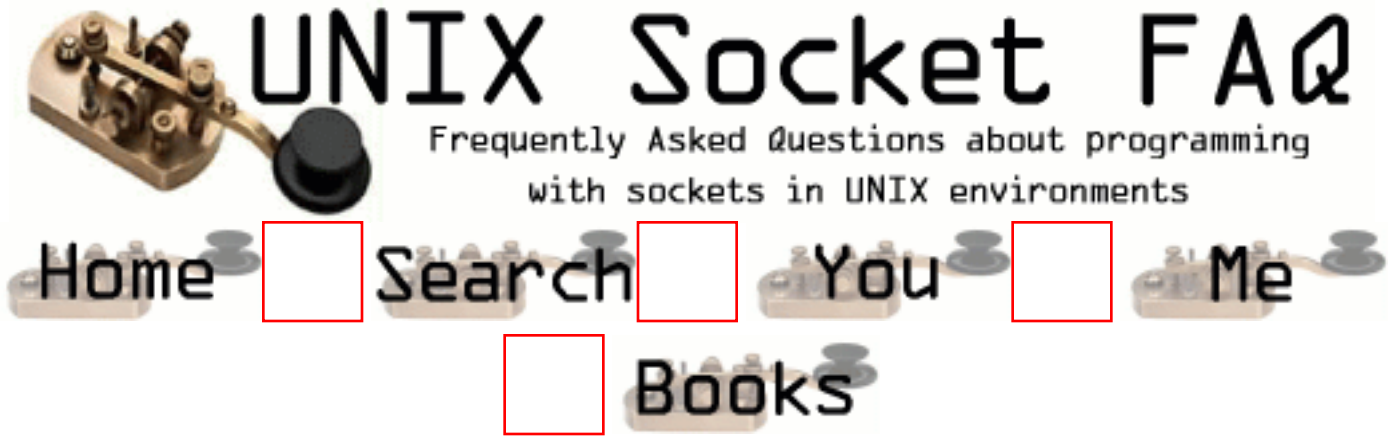
From: Amit

Hello:
I have A client program in C++ written using Winsock and I need to port this to UNIX. Can anybody help me in telling me whats the way to do . I dont have any unix idea. The client in NT is a simple V c++ program.

From: Hector Lasso

Did you use MFC or just plain old Winsocks calls?

Add a Comment



New Questions: pthread_join

From: Arul Ananad

Hi,

I have some doubt in pthread_join,

my doubts is,

can i join to the same thread (pthread_t) more than one time.

bye
Arul

From: Someone

pthread_join() suspends the calling thread until the thread identified by the first parameter terminates (either by calling pthread_exit or by being cancelled) That means that if you run again pthread_join() using the same first parameter it will fail and the error will be ESRCH, unless you have created another thread that uses the same thread identifier (is it possible?)

You can call pthread_join() many times from the same thread joining different threads

---

Add a Comment

New Questions: sending binary file from a windows client to a unix server

From: Aleksey

Hi

I'm having a problem with subj.

Here is the setup:
On the unix side, server is using a SOCK-STREAM socket in blocking mode w/ receive buffer size of 4k (default), and periodically calls select to check if there is data waiting to be read.

Windows client sends a message to tell the server it wants to transmit a file (including file size and desired name), and then immediately dumps the entire file to the socket.

The problem is, whenever header+file size is over 4k (receive buffer size on the server), after reading the first buffer-full of data, select gets confused and says there is no more. But, when I do netstat, I can see that there is data in the Recv-Q on that port, and if file is big enough, I can see even see Recv-Q grow over the default 4k.

Any ideas why this happens and/or how to fix this?

Should I force the client send the file in smaller chunks and have it wait for an ACK message from the server before sending more?

From: Rob Seace

I would guess that the call to select() is just wrong in some way, but without seeing the code, it's hard to say for sure... There's no reason that select() shouldn't see more data to read, if there truly is some there... It shouldn't make the slightest bit of difference what size chunks you

send the data in...

From: Aleksey

It's a very simple call and works for everything else...

```
FD_ZERO(&ReadSockSet);
FD_SET(m_nSock,&ReadSockSet);
result=select(m_nSock+1, &ReadSockSet, NULL, NULL, &tv);

if (result == 0)
  printf("Timeout\n");
else if (result == -1)
  perror("select error\n");
else
{
  // perform read
}
```

From: Rob Seace

What's happening, exactly? Is select() timing out? I don't see your code that initializes the "tv" info... I'm assuming you do that every time before every call to select() as well, right? Because, select() will modify the contents to reflect the time remaining (at least on some systems)...

From: Aleksey

After windows client dumps the whole file to the socket, select successfully detects something has arrived, and the first 4k is read, all successive calls to select return 0 (timeout).

tv is initialized to 0,500000 (.5 sec) and is reset before every call to select (though as far as I'm aware SCO UNIX doesn't modify the value of tv).

Perhaps the error is not in the select call but in how sockets work in the OS.

From: Hector Lasso

Do you reset the ReadSocketSet struct in the loop before calling select()?

It would help a lot if you show us the complete loop.

From: Mahafuz

I think you are missing one thing. The purpose of select(...) is to notify you if the descriptor *changes status*.
So when you are doing select(...) on descriptors and data arrives on your socket (status changes), you get the event and go to read from the socket. You read less amount of data from the socket than the amount of data available to read. after first read you again to do select. But this time
there is no event change on the descriptor. (There is no new arrival of data). So your select(...) times out. But data is indeed there to read.

If you are doing two writes in one end of socket that doesn't mean that there will be two read in other end.
TCP/IP send data by its own buffer size (fragmentation/ defragmentation occurs).

My suggestion is, after first read from socket you know the amount of data read and from the file size sent from client side, you can easily calculate the rest amount of data. Just normally read again from the socket buffer by buffer until you get the rest amount of data. Its a simle & stright recv(...)/read(...) loop. Pls don't go for select again.

From: Aleksey

I'm not sure where you got that information from. Both the select() man page on my system, and Steven's book on network programming say select() returns the total number of ready descriptors in all sets, 0 for timeout, or -1 for error.

If this really is the case, it would indeed explain why the problem is occurring, but this leaves me with another problem. I'm stuck with a blocking socket and can't do blind reads.

From: Mahafuz

I don't think what i said contradicts/disagrees with man pages of select(...). Select is associated with the concept of asynchronous i/o. It helps you to get evevts instead of blocking polling. Anyway, refering to your problem what i can suggested that,

whenever you get read event in select(..) you should start a thread/fork the process to go on reading incoming data. The main process again goes to select(...). This way your reading is not

blocked and you can serve may clients at the same time.

From: Mahafuz

I don't think what i said contradicts/disagrees with man pages of select(...). Select is associated with the concept of asynchronous i/o. It helps you to get evevts instead of blocking polling. Anyway, refering to your problem what i can suggested that,

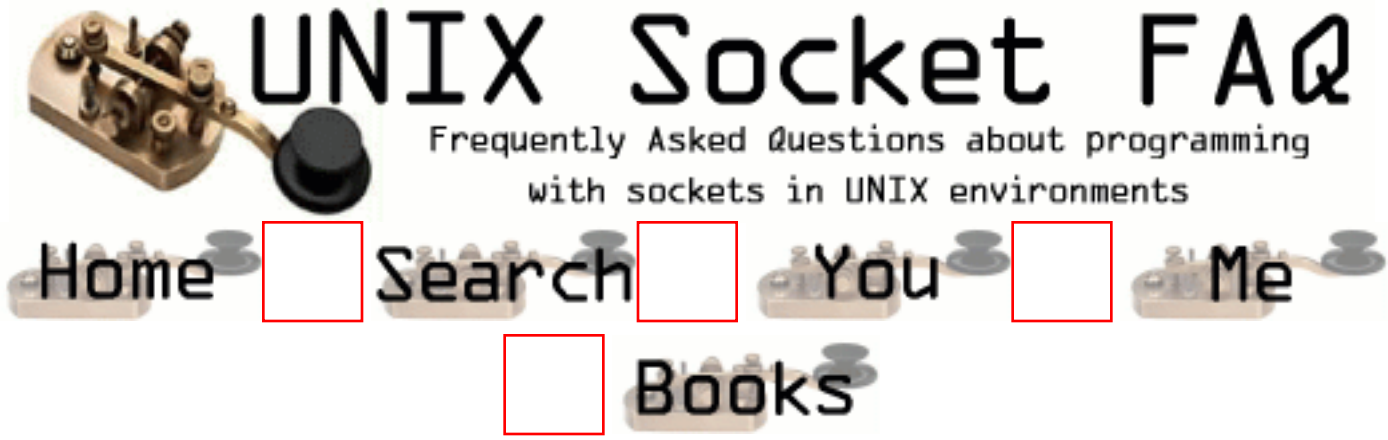
whenever you get read event in select(..) you should start a thread/fork the process to go on reading incoming data. The main process again goes to select(...). This way your reading is not blocked and you can serve may clientÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ

From: Rob Seace

Yeah, I don't think Mahafuz's description of select() is really correct, either... As far as *I* have always been aware, select() will notify you of ready sockets, even if it has already previously notified you of their readiness in a prior call... So, as long as there is still data to read, I think select()'ing for read should still notify you of the socket's readiness, even if you have already read PART of the data previously... There's still some to read, so it's still a readable socket, so select() should find it...

Now, as to what the problem might be then, I'm still not sure... It could be an OS issue... Or, it could be some flaw in your code... It's often difficult to spot flaws in code from just a few choice snippets... If you'd care to post (or E-mail) the entirety of your code, or at least all of the parts that deal with any socket calls (initial socket setup, any sockopts you set, all socket API calls, and all I/O calls on sockets), I'd be happy to take a look to see if I can see anything obviously wrong... But, you also might want to try just writing a very simple, stripped-down server that just sits in a select() loop like your own, and see if it exhibits the same anomolous behavior... If it does, then you've got a simpler test/example case to show people (and, have them try on their systems, to see if it's an OS issue); and if it doesn't, then you'll know the fault is somewhere else within your own original code...

Add a Comment



New Questions: using socket to connect : a client to a server and a client to another client ?

From: Floriane

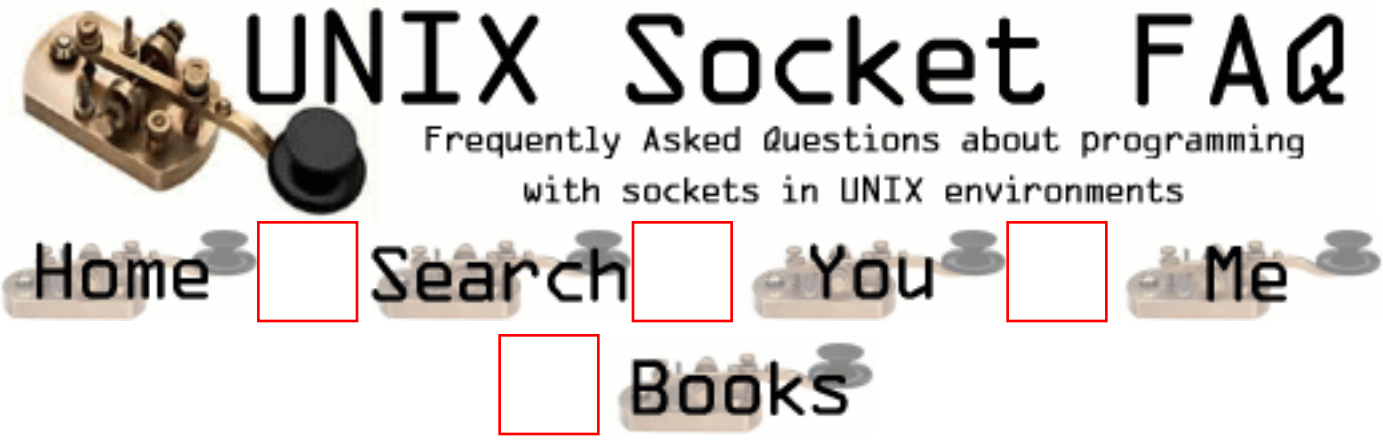
I would like to know if it is possible to have 2 clients, connected thanks to sockets to the same server, open a socket with each other, so that they don't pass through the server anymore?
And if yes, how can I do that ?
Thanks !

From: Hector Lasso

One way to do it would be:

Both clients connect to the same server. One of the clients sends through the server a message to the other client telling it its address, and probably a port where it will be listening. The second client connects to the address and port given, and the communication between both clients continues without the server.

Add a Comment



New Questions: Send a file and struct

From: Luca

How can I send a fiel using sockets?

And a Structure?

From: Hector Lasso

To send a file through a socket you could just read the file and send it through the socket using write()/send(). However, you should tell the receiving program something about the file, for example file size, in order for the receiver to know how many bytes coming from the socket it should expect.

To send structs is a little bit trickier. If you are using the same platform in both ends you just cast your struct to char*:

```
struct my_struct data;
// ... do something with struct
send(my_socket, (char*)(&data), sizeof(my_struct), 0);
```

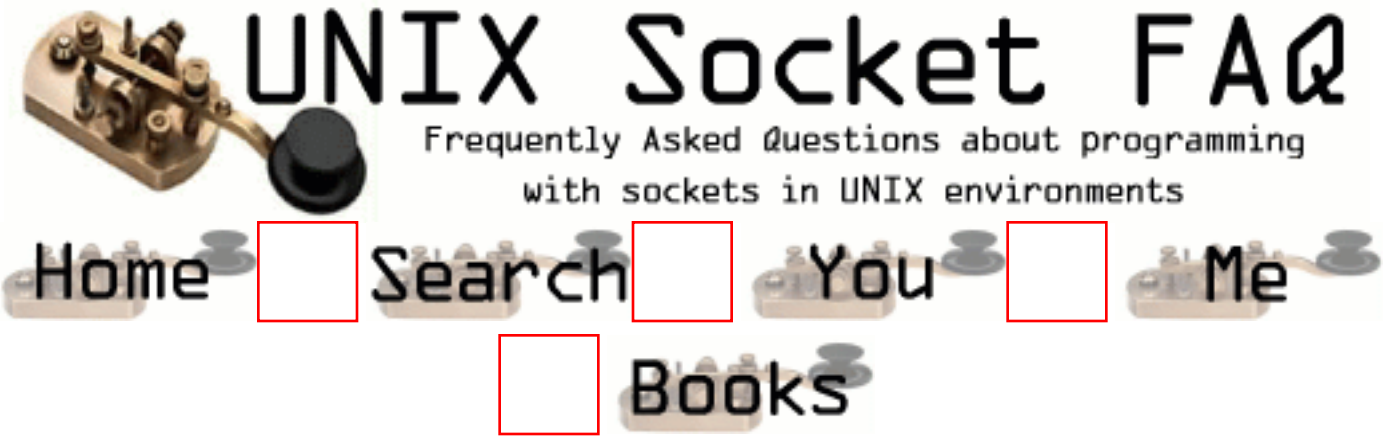
The receiver should do something like:

```
struct my_struct remote_data;
recv(my_socket, (char*)&remote_data, sizeof(my_struct), 0);
```

From: Hector Lasso

You can find more details in the main FAQ: Question 2.15

Add a Comment

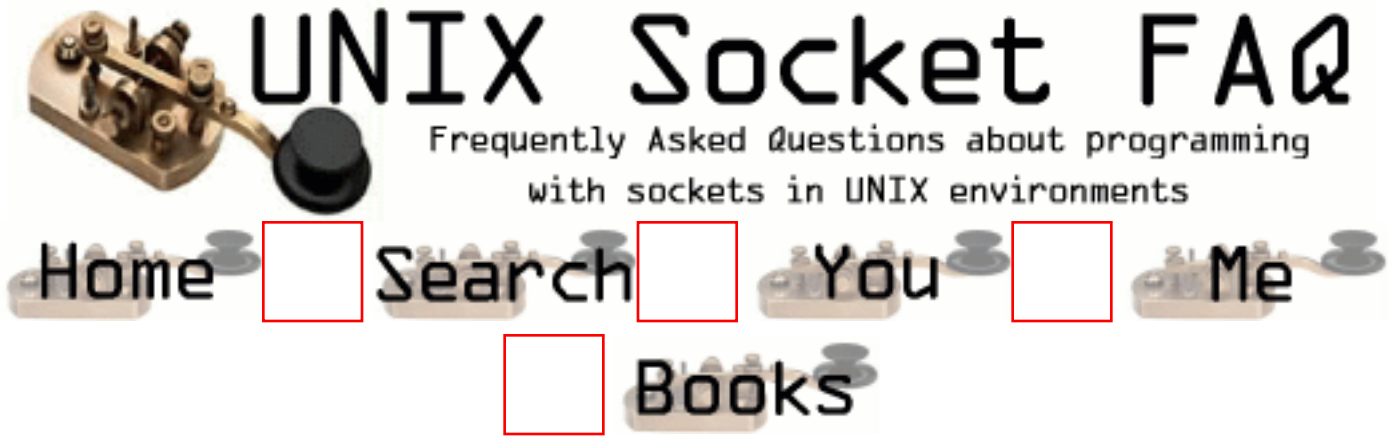


New Questions: configure tcpip/ppp using unix interprise 5.05 sco

From: BRYANT BORDERS

How do I configure unix remote dial-up to use tc/pip or ppp?

Add a Comment



New Questions: Determining MAC from IP w/ARP

From: Todd

I'm on a UnixWare 7.1 platform. I need to look up a MAC address in the ARP cache from a known IP address. I foudn the following example source but it does not work. What am I doing wrong?

```
#include <stdlib.h>
#include <stdio.h>
#include <sys/sockio.h>
#include <sys/socket.h>
#include <net/if.h>
#include <net/if_arp.h>
#include <sys/ioctl.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <errno.h>

int main(int argc, char *argv[])
{
  struct sockaddr_in sin = { 0 };
  struct arpreq myarp = { { 0 } };
  int sockfd;
  unsigned char *ptr;

  if(argc!=2) {
    printf("usage: %s <IP address>\n",argv[0]);
    exit(0);
  }
```

```
 sin.sin_family = AF_INET;
 if(inet_aton(argv[1], &sin.sin_addr)==0) {
   printf("%s: IP address '%s' not valid\n",argv[0],argv[1]);
   exit(0);
 }

 memcpy(&myarp.arp_pa, &sin, sizeof myarp.arp_pa);

 /*strcpy(myarp.arp_dev, "eth0");*/

 if ((sockfd = socket(AF_INET, SOCK_DGRAM, 0)) == -1) {
   printf("%s: cannot open socket\n",argv[0]);
   exit(0);
 }

 errno = 0;
 if (ioctl(sockfd, SIOCGARP, &myarp) == -1) {
   printf("%s: no entry in arp_cache for '%s', errno = %d\n",argv[0],argv[1],
          errno);
   exit(0);
 }

  ptr = &myarp.arp_ha.sa_data[0];
  printf("%s: MAC address for '%s' is : ",argv[0],argv[1]);
  printf("%x:%x:%x:%x:%x:%x\n",*ptr, *(ptr+1),*(ptr+2),
     *(ptr+3),*(ptr+4),*(ptr+5));

  return 1;
}
```

From: Todd

I almost forgot, the ioctl sets errno = 6, no such address or device.

From: Min Hsieh

I compiled the code on Solaris 2.7 and it works fine.

But there are some places you have to make adjustment
though, like "inet_ntoa" --> "inet_addr".

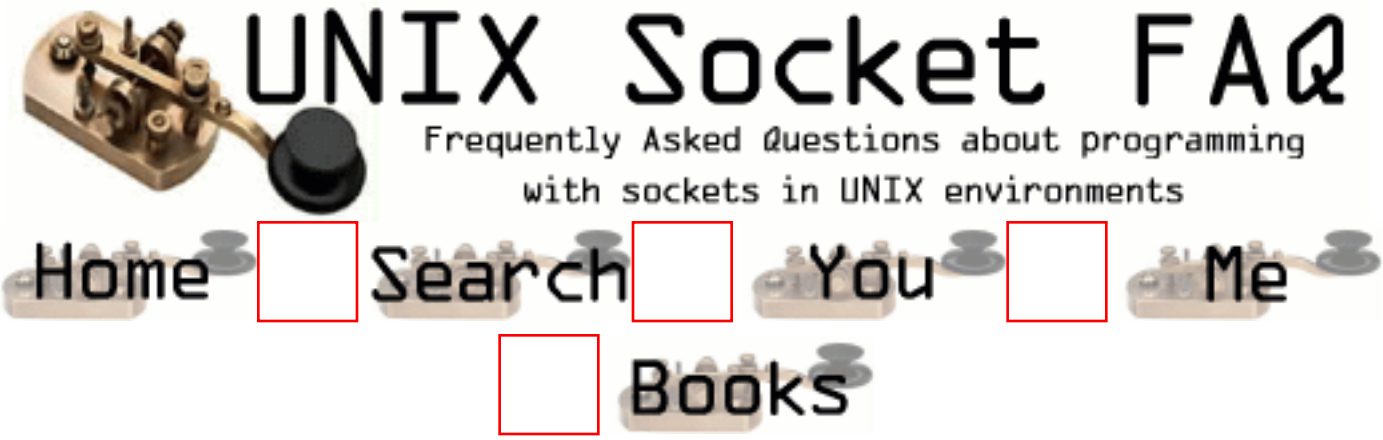
It's working, but it can only get IP<->MAC mapping
from currently existing ARP table. Not very useful.

Cheers.

From: Todd

I found the problem. The example code was using a socket's file descriptor, which would work on a connected socket. For my purposes I had to open /dev/arp and use that file descriptor.

Add a Comment



New Questions: Sockets Intialization Failed!

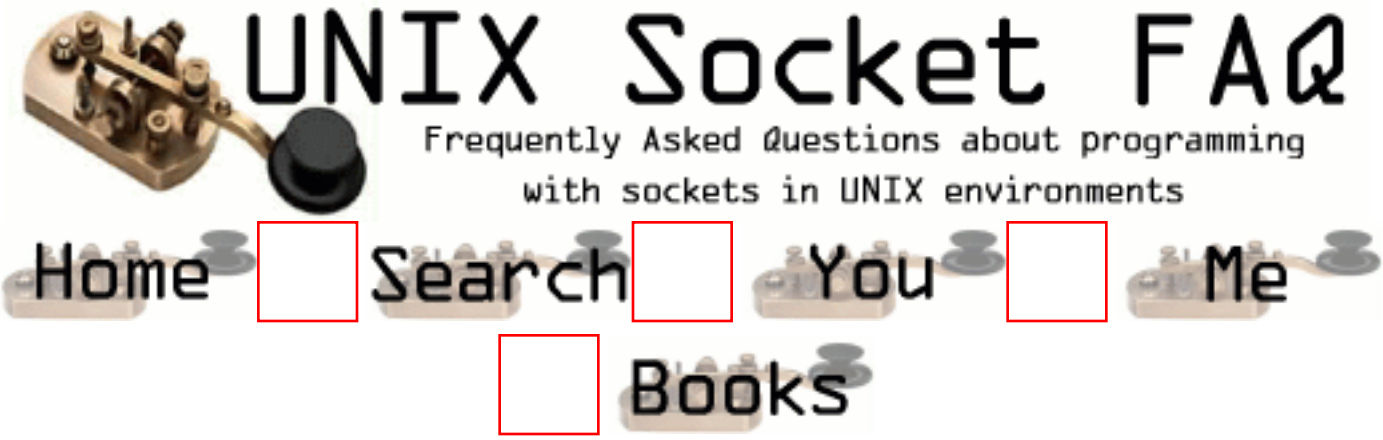
From: Richie

Out of the blue my computer tells me that "Sockets initialization Failed!" when i try to run an internet related application, eg IE5 or ICQ. I can connect to my ISP fine but thats as far as things go!
HELP ME!!
please reply.

From: Rob Seace

Um... IE5?? It doesn't sound to me like you're running
Unix... (Take a look at the name of this site, again...)
This is a site about writing programs using Unix sockets,
not a Microsoft TCP/IP problems help desk...

Add a Comment



New Questions: Anonymous FTP

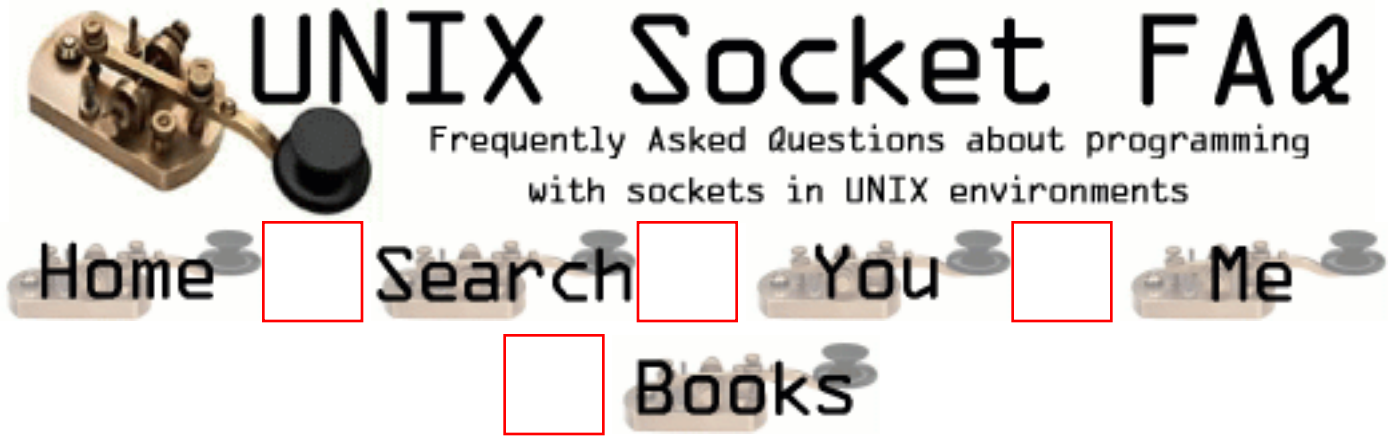
From: Johannes Herseni

How can I get if a FTP-Server allows anonymous FTP?

From: Hector Lasso

You should read the documentation that comes with your FTP server.

Add a Comment



New Questions: Problem with socket-timeout

From: Carsten Schaudel

Hello together !
I have the following problem:
When I establish an TCP-Connection (in my case client and server are running on the same PC) the connection seems to have an timeout if I don't transmit cyclic data in both directions. Suddenly (after about 30 Min) I get an error ETIMEOUT on the client when I try to receive data. Just before this error it was no problem to send data from the server to the client.

Any ideas how to switch of this timeout ?

(using Linux 2.2.18)

From: Aleksey

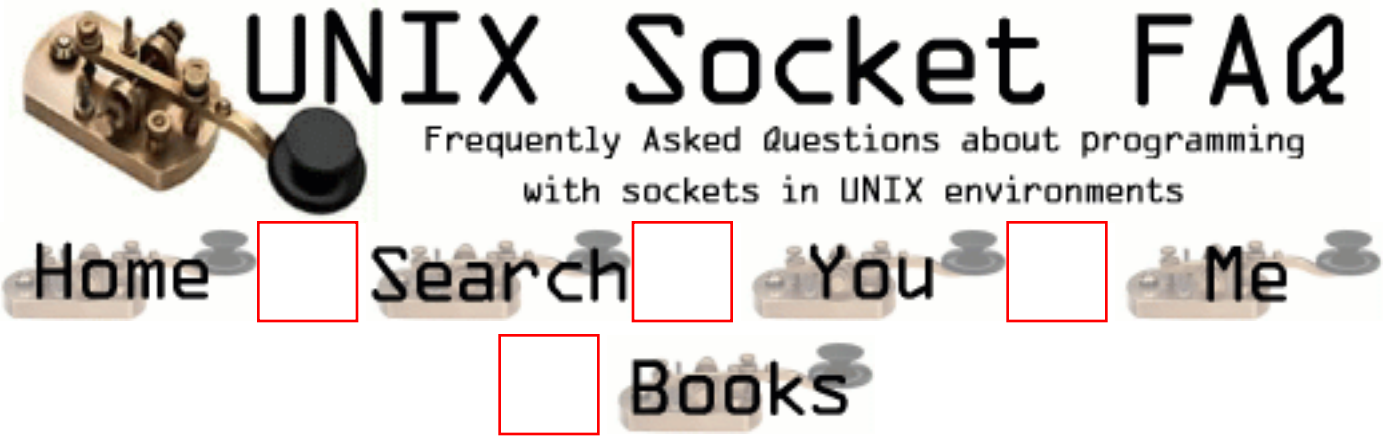
Try setting SO_KEEPALIVE with setsockopt().

From: Carsten Schaudelo

Hi !

I allready tried setting SO_KEEPALIVE, but that hasn't changed anything regarding this problem.

Add a Comment



New Questions: Which langage is better for network programming ?

From: Francky

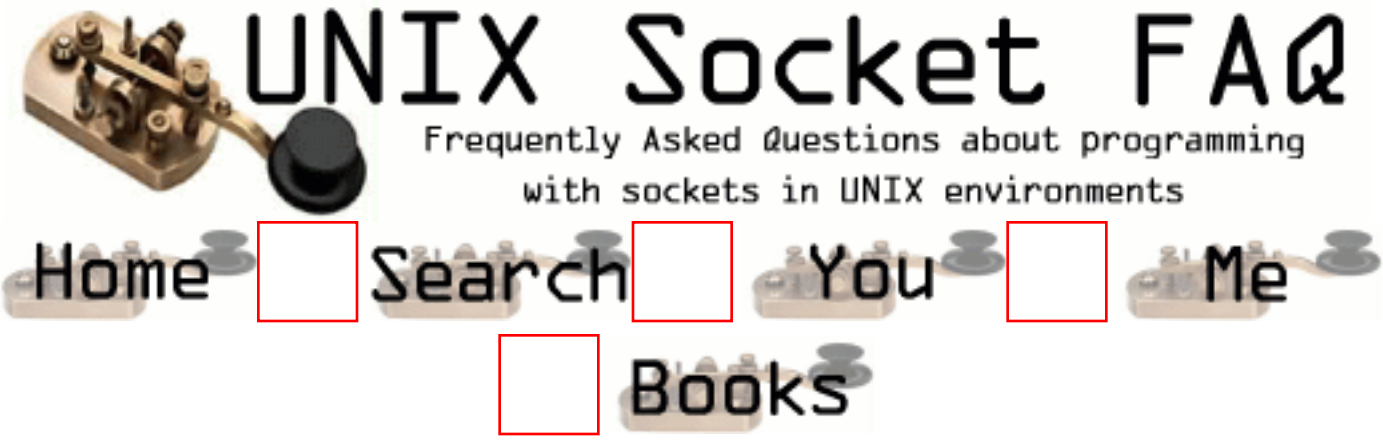
Hello,
I am a beginner in network programming and I have
read the Stevens Book called 'Unix Network Programming' and
many others too, I also have the source code of a game
server, and it looks like everybody use C for server coding,
so I would like to know if there is a difference between
a C coded server and a C++ coded server (performance,...)
All the reference books are using C instead of C++, maybe
there is a good reason for that.
I need to design a server with a huge number of connected
clients (3000 at the same time) so should I be using C or
C++ knowing that my server will be an online game server.
So the server will also have to create the behaviour of
some monsters using Artificial intelligence algorithm,
the problem with C is that we can't use the STL library for
example (am i right ?)
or an AI library, what could you suggest me ?
Thanks in advance.

From: Rob Seace

Well, questions like this are really completely and totally
subjective, and akin to asking, "Which religion is best?",
or something along those lines... More flame-wars have been
fought over which programming language is "best" for any
given task than over OS's or text editors, I think... ;-)
My personal opinion is that C is superior for most tasks,
including network programming... Mainly, that's probably

because I'm most familiar with C, and most comfortable using it... Someone else may certainly prefer something else, instead... Why do most networking books only use C? I don't know; probably because most of the authors are similarly biased towards C, as I am... Also, the fact that the standard sockets API is for C probably has something to do with it, as well... Of course, there's no reason you can't use any C code from C++, so there's nothing stopping you from writing network code in C++, if you like... And, of course, the sockets API (or, something similar to it) is now available in a wide variety of other languages, too... But, C is still the 'standard'... *shrug* Basically, I'd say you should choose whichever language you are most comfortable writing code in, and whichever you think will best provide you with the features you need to develop your program... If you think that's C++, go for it... I, personally, wouldn't use C++, but that's mainly because of personal biases (and, the fact that I had bad experiences with old generation C++ compilers generating HORRIBLE code that just performed like crap; modern compilers should be much better, I'm sure)...

Add a Comment



New Questions: server

From: sohail

hello,
i got this small querry.
i have a unix server.i have a small html file n my server which is a form template for submitting a particular form.
now,i usually access this html from my browser.
my problem is that my isp speed is very slow in india so it take lots of time to surf from my browser.
is it possible for me to access the html file from my server itself[without the need of my ISP speed].
i tried using telnet and lynx to access the file but eventually the ISP comes in the picture .
I need to submit this form every 5 minutes for 1 hr per week.is there any script which works from the server itself and sends the form from the server-without the isp .
Please help.
Regards
SOhail

Add a Comment

New Questions: I need to send 0x00 0x00 0x00 0x1 ?

From: julio

I need to send 0x00 0x00 0x00 0x1 but the write or send does not work it send none how can I do it.

```
msg[0]=0x00;
msg[0]=0x00;
msg[0]=0x00;
msg[0]=0x1;

n= wriet(my_socket,msg,4); or n send(my_socket,msg,4,0);
printf("%i",n);
>>> this print 0
```

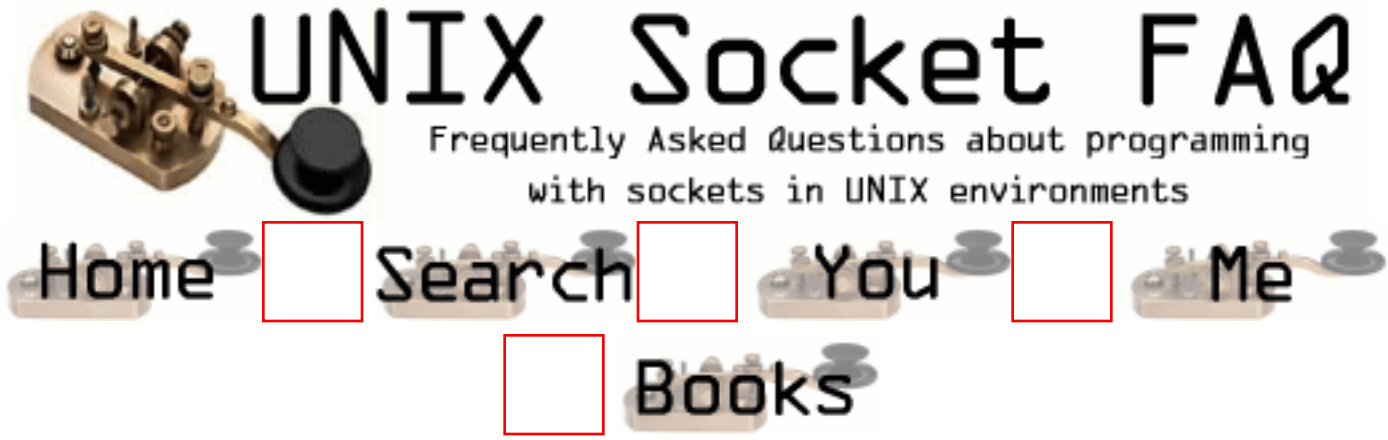
julio.
Australia

From: Wes

```
{
  int data;

  data = ntohl(1);
  write(fd, &data, sizeof(data));
}
```

..should do the trick.

Add a Comment



New Questions: Broken Pipe

From: julio

i got this error at writing. i do not know what mean.

signal PIPE (Broken Pipe) in _write at 0xeef392c8
0xeef392c8: _write+0x000c: bgeu _write+0x40
Current function is Socket::Send
  159 n = write(sockfd,frame.data,frame._size);

From: Rob Seace

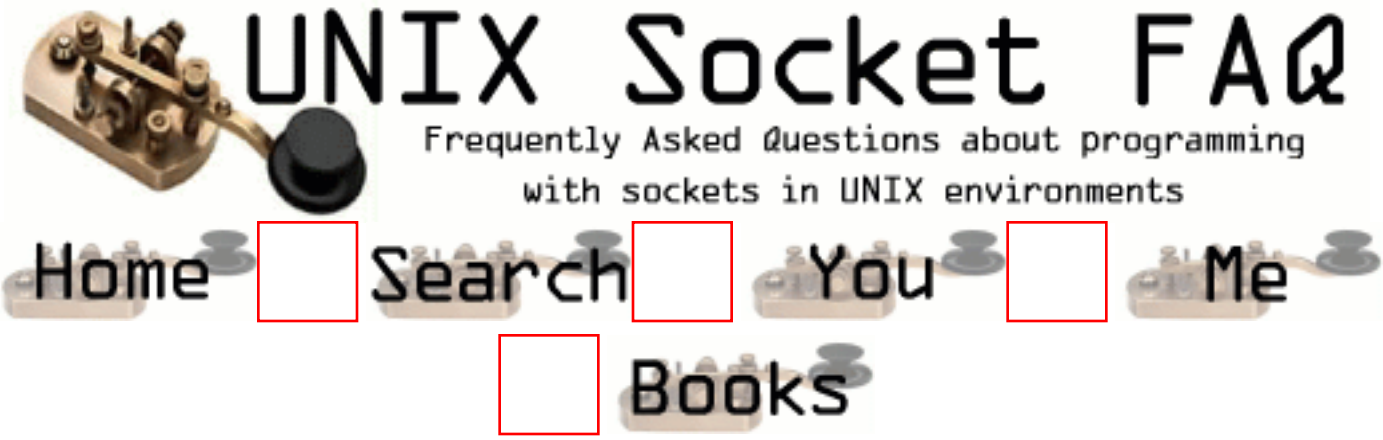
First, see questions 2.22 and 2.19 in the main FAQ...

But, basically, it probably means the other end of the socket has been closed... I recommend always ignoring SIGPIPE, then you can just check for EPIPE on write() failures, if you need to...

From: Baskar K

This normally means that you were performing a data transfer on a socket connection
and the socket connection is broken during the transfer.This
might have happened when your process is killed or some network
interruption problems.
You can normally ignore SIGPIPE and check for the return value of recv and send
socket calls.

Add a Comment



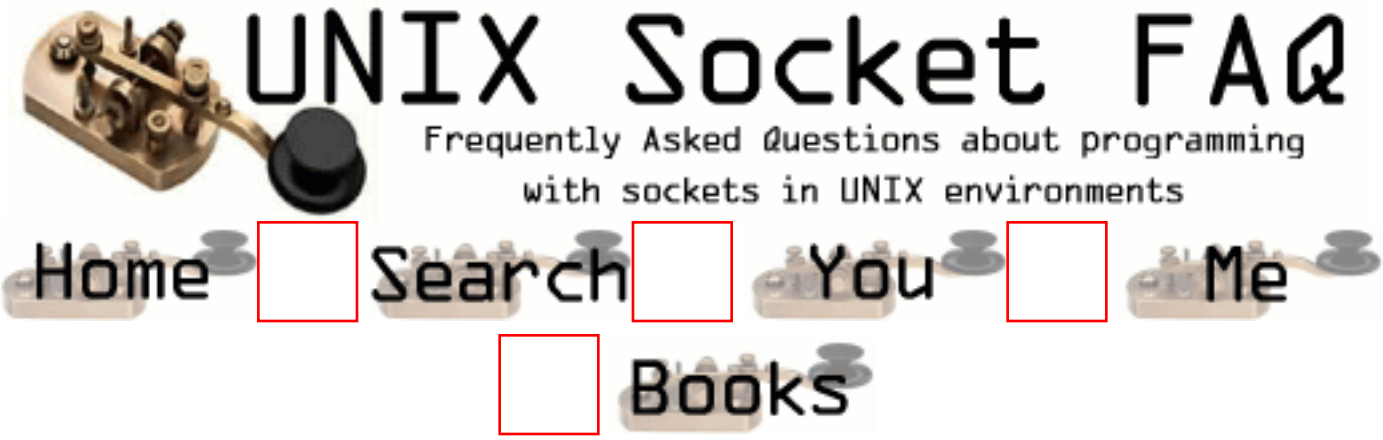
New Questions: Open a session

From: Luca

How can I open a session on another terminal using login a pwd

I have to write a prog that connect to a host and do a log on it with login a pwd.

Add a Comment



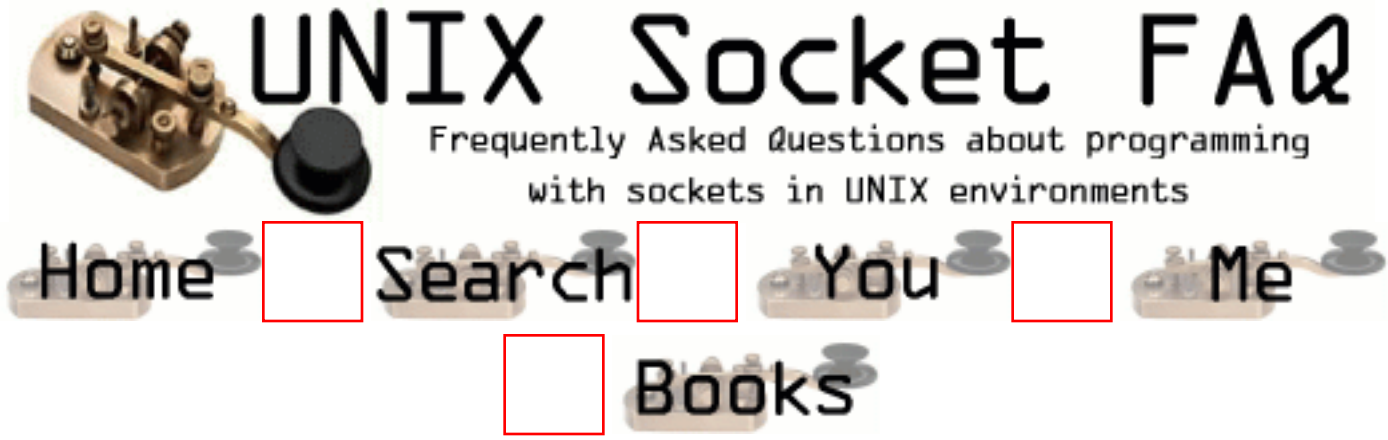
New Questions: Can i Read more than a PAGESIZE in a single read call.....

From: Arul

Hi,

Can i Read more than a PAGESIZE in a single call....

Add a Comment



New Questions: kick off Unix process from Windows 2000/NT

From: Utpal

I am need to write a C function which will kick off a unix batch process. I want to include that function in a win32 DLL. Would you please suggest how can I accomplish that?

From: schang

NT:

- use winsock on NT in your application, then establish connection to tcp listener on Unix,
- send batchid and other parameters via tcp socket connection
- wait for the confirmation from Unix server
- close socket
- check your job run schedules from database

Unix:

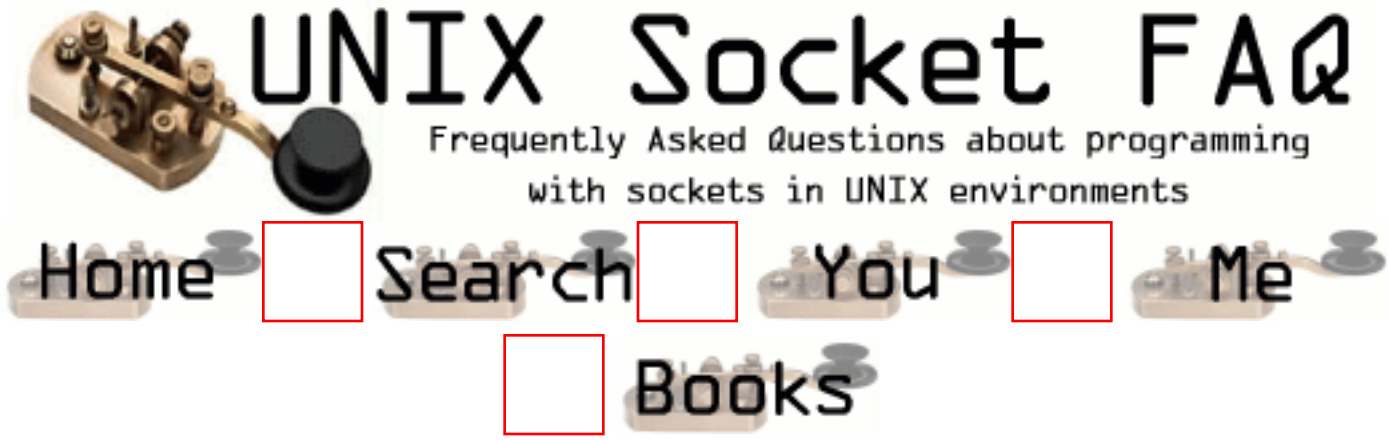
"fork" a child on Unix right after socket äccept(),

in child process:

- close parent's socket
- read batchid and other parameters from socket
- write confirmation back to socket
- close socket
- write batchid, startup timestamp, status to database
- start your batch
- write batchid, completion time, status to database
- exit

in parent process:
- continue to accept incoming request

Add a Comment



New Questions: Why my recv is too slow?

From: Mok

Hi:
  Here is my recv function, i notice my receiving procecess is take
long time to download a file (from UNIX server) which 1/2M in size.
It this any problem in my coding. Can u tell me, the receiving speed
is depend on what factor beside the network/modem speed???

Thank U

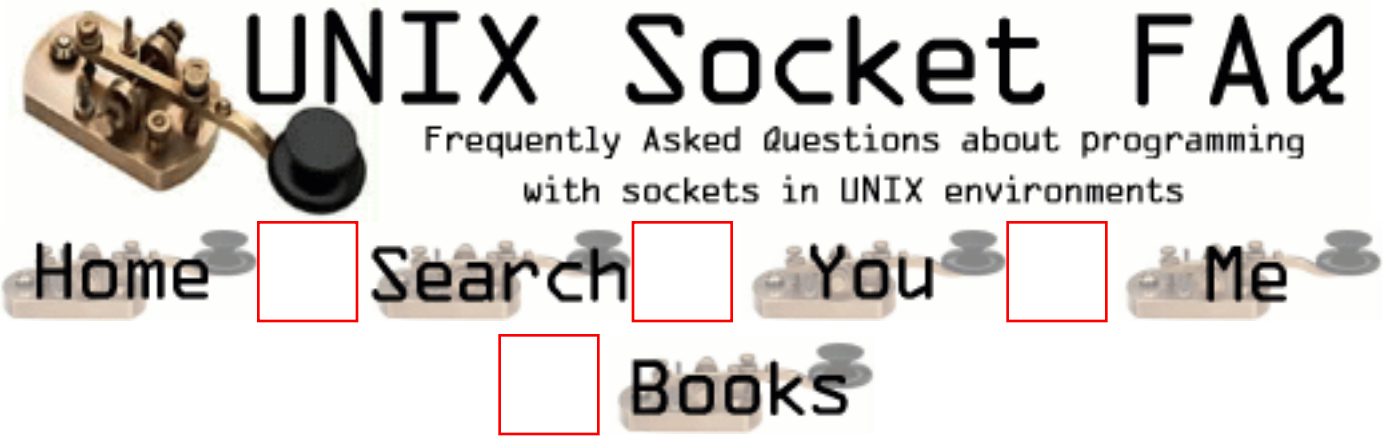
```
//non blocking mode.
int RecvMsg(char * cData)
{
 int err;
 char cstr[256];
 CString Tmp =_T("");
 int nTry;
 int nRet ;

  if (soc == INVALID_SOCKET)
  {
   return 0;
  }
  nTry =0;
  nRet =0;
  while (nTry < 100)
  {
     err =recv(soc,(char FAR *)cstr,256,0);
    if (err == SOCKET_ERROR)
    {
     err = WSAGetLastError();
```

```
if (err == WSAEWOULDBLOCK)
{
nTry++; // no data
nRet = 2;
Sleep(30);
}else
{
nTry=10; //exit lloop
nRet =1;
}
}
else
{
nTry=10;
nRet =3;
Tmp =cstr;
    strcpy(cData,Tmp.Mid(0,err));
}
  }
 return nRet; // have data
}
```

From: johnny

I noticed in your recv statement, you have a 256 byte block size. Change this (and your buffer) to the max, which on most systems is around 8K (8192). This should improve your speed considerably.

Add a Comment



New Questions: select loop..

From: Dan

I've encountered an odd little happening in my code which loops (to all intents and purposes) forever, polling a sockset alternately for read and write..

The code works fine if it finds something it can read from everytime (forget the writing bit for the moment), but the first time it times out and fails to read anything from the socket, it sobotages its ability to ever read anything again! From that point onwards, it doesn't seem to actually be polling the set properly, and just returns 0, whether there is stuff to be read or not..

Hope sum1 can cast some kind of light on this, as it would be much appreciated..

From: Dan

I've made a slight alteration which makes it (kinda) work.. I've simply cleared the sockset and reinitialised it inside the loop.. like I say, it works, but surely there's a better way of doing it? I must be using select wrong somehow?

From: Wes

That's the right way to do it.

When select() returns, it clears the fds that it says there is data for. You need to (at the very least) re-insert the fd back into the fd_set with the FD_SET macro.

It may be possible to make a copy of the fd_set, and re-initialize from the copy (when selecting over a hard-to-get fd_set), but I am not sure if this technique is portable, as it relies on the internal representation of the fd_set as being a bit field, and not (say) an array of int *s or something.

From: Dan

ok, thanx 4 the help :) I only have 1 socket in the fd_set so it isn't much of a problem!

From: Hector Lasso

Hi,

Whenever select() returns, the file descriptors are cleared from the sets if they are not ready for the corresponding operation...
Also, the timeout value should be reset before calling select() again...

You can use a copy of the sets:

```
fd_set rset, rsettmp;
FD_ZERO(&rset);

FD_SET(socket_descriptor, &rset);
// Add other descriptors if any
do {
  memcpy(&rsettmp, &rset, sizeof(rset));
  select(maxfd, &rsettmp, NULL, NULL, NULL); // No timeout
  // Process results here...
} while (some_flag);
```

From: Dan

Again, thanks fer all yer help :)

From: Stephen Silvey

An example implementation I saw out there on the net. Maybe this can help?
--Begin:

```
/*
 * do_select() - based on select_loop() from the Harvest Broker.
 * -- Courtesy: Darren Hardy, hardy@cs.colorado.edu
 */
int do_select(sock, sec)
int sock; /* the socket to wait for */
int sec; /* the number of seconds to wait */
{
        struct timeval to;
        fd_set qready;
        int err;

        if (sock < 0 || sec < 0)
                return 0;

        FD_ZERO(&qready);
```

```
        FD_SET(sock, &qready);
        to.tv_sec = sec;
        to.tv_usec = 0;
        if ((err = select(sock + 1, &qready, NULL, NULL, &to)) < 0) {
                if (errno == EINTR)
                        return 0;
                perror("select");
                return -1;
        }
        if (err == 0)
                return 0;

        /* If there's someone waiting to get it, let them through */
        return (FD_ISSET(sock, &qready) ? 1 : 0);
}
```

--End

From: Stephen Silvey

Another little tip, from the same example code (just in case, but this is pretty old):

--Begin:

```
void reinitialize(s)
int s;
{
        /* To force main-while loop call reinitialize_server() after do_select() */
        glimpse_reinitialize = 1;
#ifdef __svr4__
        /* Solaris 2.3 insists that you reset the signal handler */
        (void)signal(s, reinitialize);
#endif
}
```

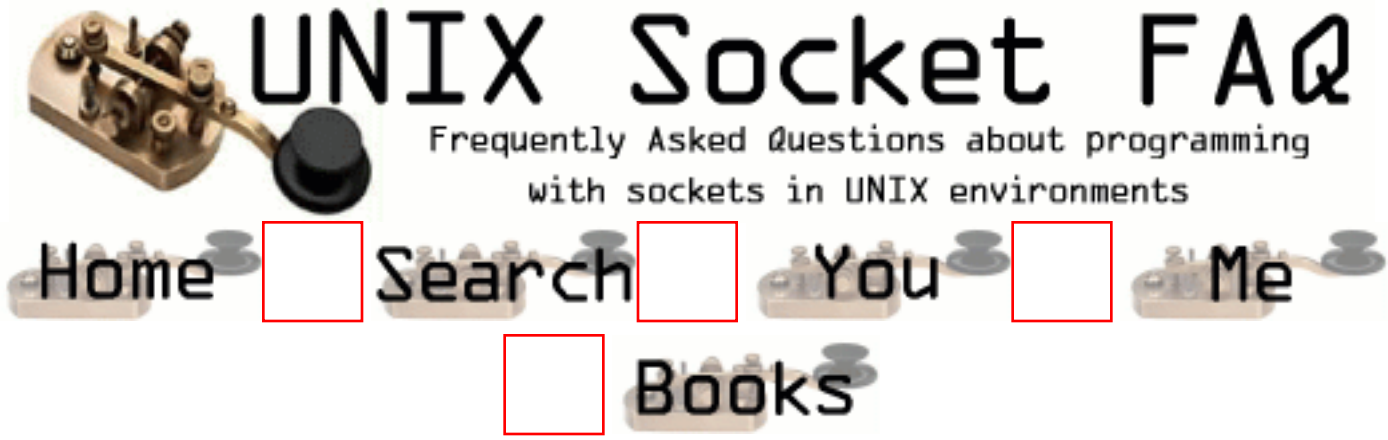
--End

From: Wes

Hector said: "Also, the timeout value should be reset before calling select() again..."

This is an absolute must when trying to write portable code. On some platforms (BSD comes to mind), select() will change the struct timeval to indicate the amount of time left in the timeout when select returned.

Add a Comment



New Questions: udp port reuse

From: LI Jiang

I am working on Reh Hat Linux 6.2
I have a process listening on udp port P, I am writing another programm in C
to listen on the same udp port P in a unicast fashion. I have used

int optval =1;
setsockopt(sd, SOL_SOCKET, SO_REUSEADDR, (char *) &optval, sizeof(optval)))
<0);

But execution gives the following error: address already in use

I have read with: man 7 socket
in paragraph SO_REUSEADDR that
with SO_REUSEADDR a socket may bind, except when there is an active
listening socket bound to the address. When the listening socket is bound to
INADDR_ANY with a specific port then it is not possible to bind to this
port.

But I can listen on the same udp port P using the following tcpdump command:

/usr/sbin/tcpdump udp port P
tcpdump can listen on this port while there is another active loistening
socket on the same port P.

I would like to hnow how to bind and then listen on a udp port P, while
there is another active socket on the same port.

thank you for your help

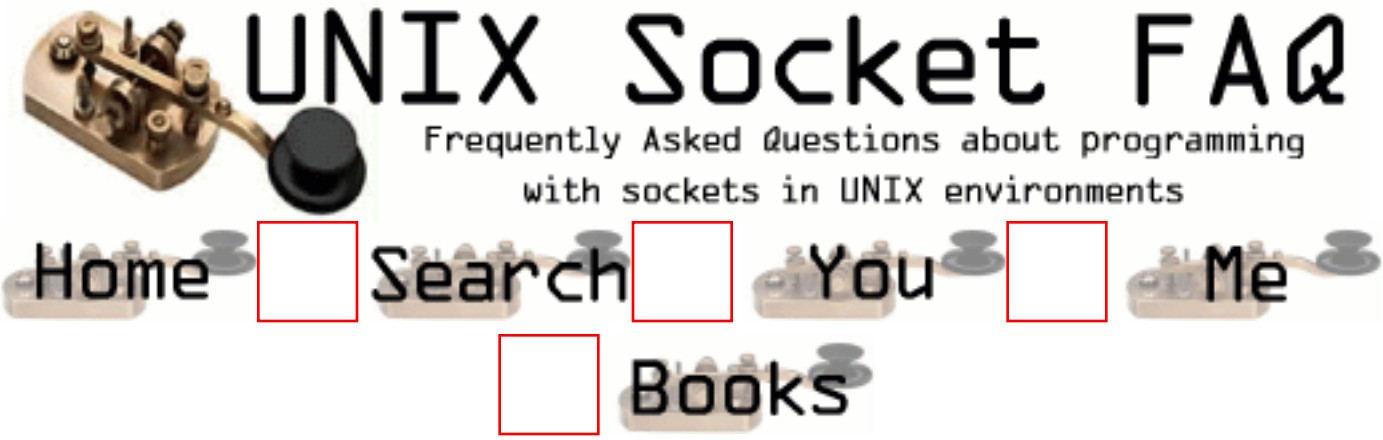
best regards

LI Jiang

From: Wes

I seem to remember something about using SO_REUSEPORT in this context when reading Stevens: UNPv1. You may wish to investigate that.

Add a Comment



New Questions: sockets

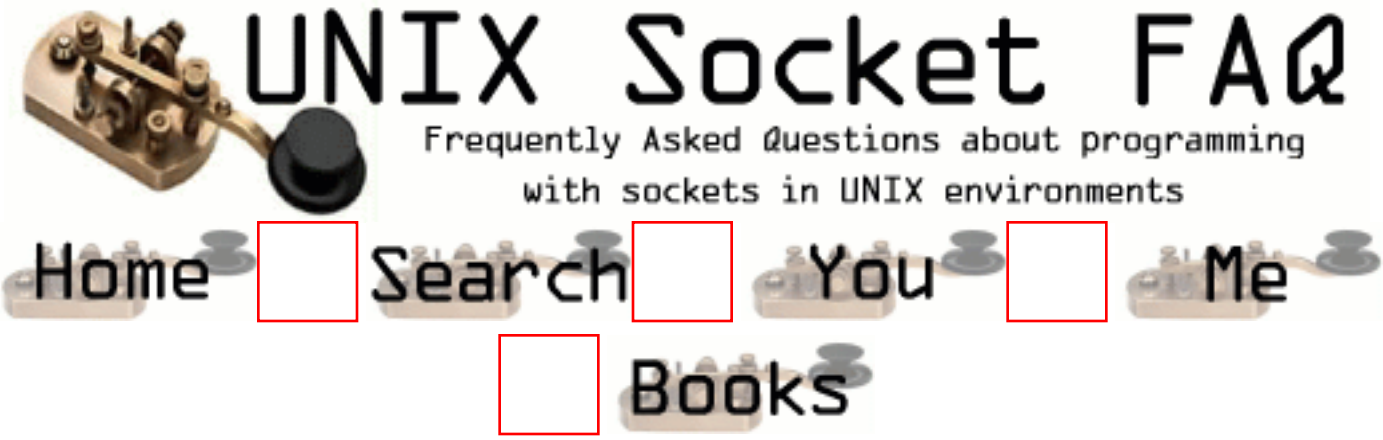
From: noniesha

why do we need unix sockets when the internet sockets can provide us with the same functionality as unix sockets ?
plus i want to ask that how do we use unix socekts on a LAN

From: Premjee K. Abraham

Sir,
I had a problem of connecting to FTP port (21) of a Unix system from a windows NT machine using java Socket programming. I created a socket with the Unix Machines IP address and port no 21. But i am not able to login to the Unix system since the password i wrote to the socket output stream is not taking. Please tell me how to get a file from a unix machine to a Windows machine using Sockets. ?...

Add a Comment



New Questions: socket send timeout

From: stefano

Dear All,

I have a server socket connected via lan bnc to client one. The
protocol is tcp and i used a realtime OS supports TCP.
If i disconnect the lan between the machine or one machine reboots itself,
the send() on sockets doesn't fail (it will fail after 11 minutes).
For me it is too long time, is there any possibilities (setsockopt or ioctl)
to change this timeout ?.

thanks
Stefano

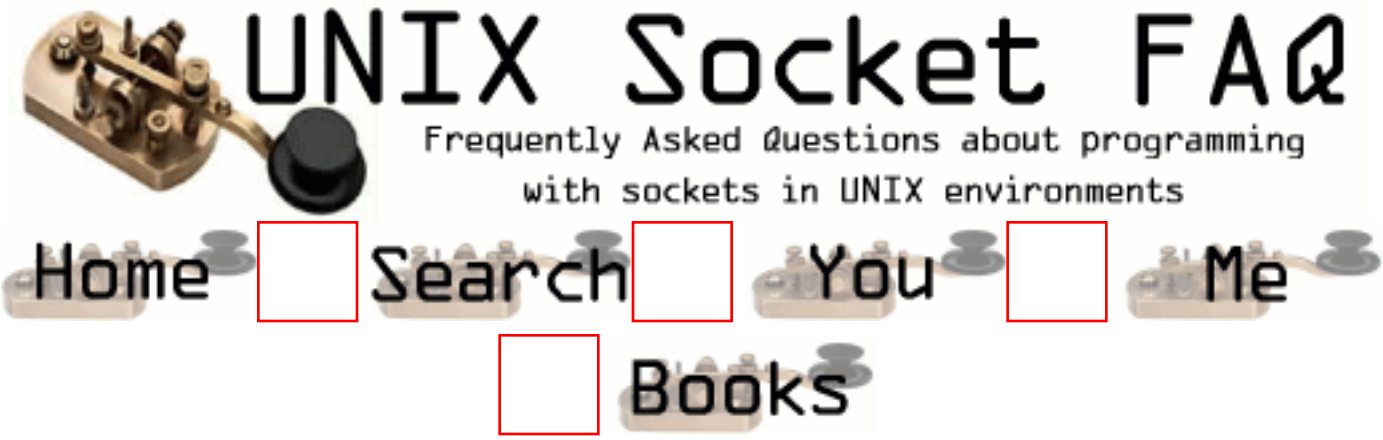
From: Rob Seace

This is covered in the main FAQ in question 2.8... Basically, the keepalive timeout is usually configured SOMEHOW, but how is not standardized at all, and it's generally only on a system-wide basis, not a per-connection basis... I can tell you how to set it on Linux ("/proc/sys/net/ipv4/tcp_keepalive_*"), and I have a rough idea of how you'd do it on QNX (they seem to have a per-connection "TCP_KEEPALIVE" sockopt, which sets SOME value related to it, but, I'm not sure if it's the initial timeout, or just the time between successive probes), but anything other than that, I have no idea... It's a very system-specific setting, I'm afraid... (I really wish someone would standardize all these little differences, like this...)

From: Rob Seace

D'oh! The dangers of cut-and-paste rear its head... ;-) That's obviously not the correct URL... (Though, it is the URL for downloading a very cool book... ;-)) What I meant was: question 2.8...

Add a Comment



New Questions: byte order of data received

From: Chad

We are trying to create client/server environment between BSD 4.1 and a Tandem machine. The data going back and forth seem to be byte swapped. Is there a socket flag/option that takes care of this for us or do we need to write our own?

From: Stephen Silvey

First, (some terminology) that may be unclear:
-- IPvX == (Internet Protocol Version xx)
Note: The current IP Version == Version 4

IPv4 says that the byte order of data transmitted is currently dependent upon the host machine. However, for portability purposes, one should be sending and thus, receiving all data in Network Byte Order, which is big-endian for the internet protocols, in order to conform to the new forthcoming IPv6.

I believe that you'll have to implement some sort of byte-swapping routines for IPv4 due to the current IPv4 header structure definitions.

In IPv6, constants have to be setup for the hardware's byte ordering. Constants or data fields not in network byte order must be converted using something like htons() or htonl() calls, at run time.

I'm unclear if this will work on IPv4, but its worth looking into.

Please feel free to comment on my remarks. That's how we can learn. Thanks.

From: Stephen Silvey

Someone said it much better than I did, here is the link:

http://beta.ttt.bme.hu/info/libc.info.Byte_Order.html

Good luck.

Add a Comment

UNIX Socket FAQ

Frequently Asked Questions about programming with sockets in UNIX environments

New Questions: Searching For: Inline "Hex to Decimal" conversion routine - to be used during execution of Socket Datagram Program

From: Stephen Silvey

Can anyone point me to, or does anyone know of any inline routines available for converting hex to decimal during program execution that will work on a Linux Redhat 7.0 platform? I am attempting to port some existing code, written in Ada on an old SGI system, to a PC-based Linux Redhat 7.0 system, and the old hex converter routine is crashing on our new system. We are using unint_32. I don't care what language (C possibly). Any input would be most appreciated. By the way, I didn't actually see the errors caused by the old routine, so you'll just have to take my word for that matter. I can find out, probably.... Let me know.

--Stephen Silvey--
(re: silveystephen@excite.com)

From: Rob Seace

Well, this strikes me as more of a general programming question, and not really socket-related at all... But, still, I'll bite... ;-)

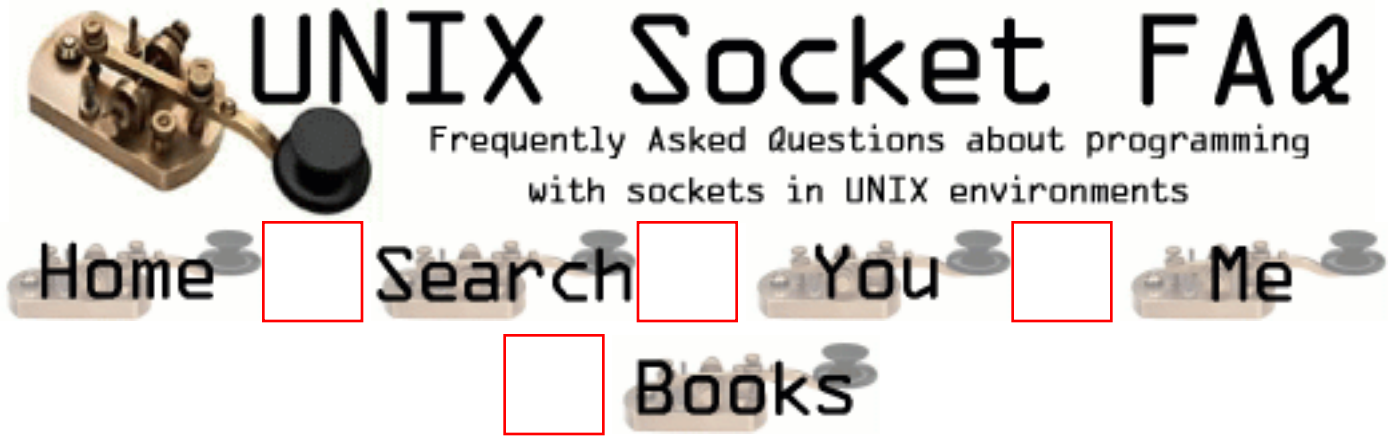
I assume you mean you've got a buffer containing a string representation of a hex number, and you wish to parse that, and get a normal C "int" (or "long", or "uint32_t", or whatever)? The simplest thing is to just use the standard

library function strtol()... Eg:

```
char *example = "abcd";
long l = strtol (example, NULL, 16);
printf ("%ld\n", l);
```

Would yield output of "43981"... See your local man page
for more info...

Add a Comment



New Questions: How to determine which IP client is calling?

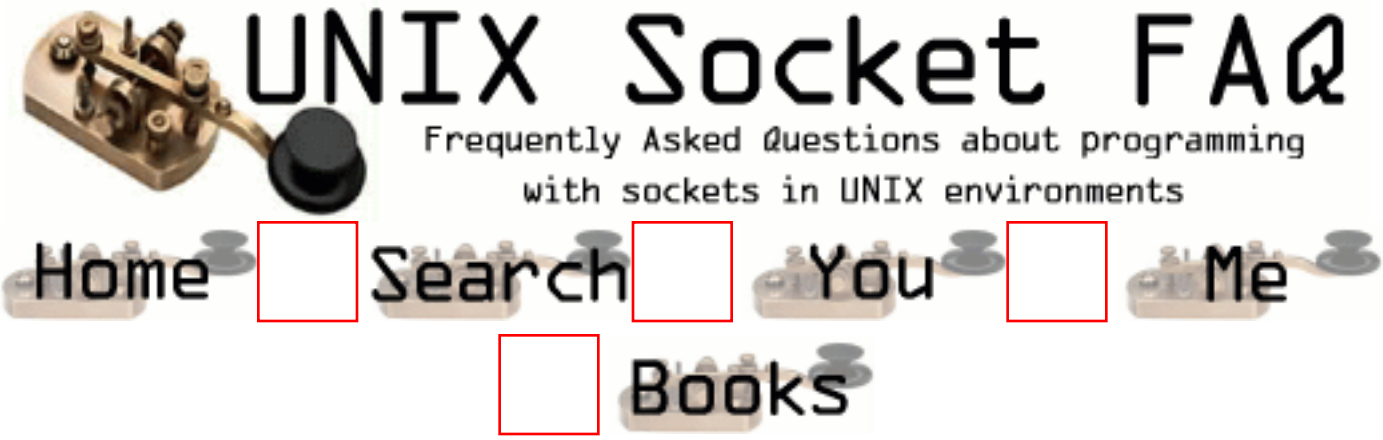
From: Tomek

Hello.
I have got a server with two IP addresses assigned to it, and a programm that creates a socket on a specific port.
Clients that connect to this port are divided into two groups, each group use different IP, but the same port number to connect.
Having accepted the connection, how to determine which of my two IPs the client used to connect to my server?

From: Satya Prakash Prasad

When a connection gets establish ie accept() returns success.
the client's and server details can be gathered by using functions
like getpeername (to get the client details) and gethostbyname
(for the server details) . This event needs to be trapped
Ok.

Add a Comment



New Questions: How to reconnect a TCP socket?

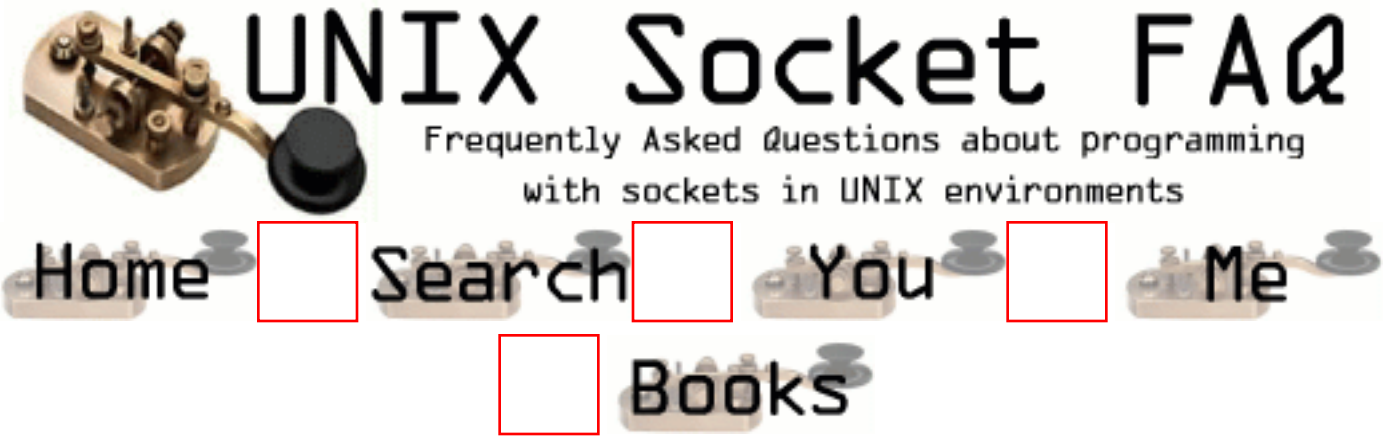
From: Ched

In order to minimize the number of socket creations in my client app (which has to send TCP requests to many different servers), I would like to reuse my existing sockets.
- Is it worth to do it? I mean, does a call to socket() cost a lot?
- To reuse my socket, I call shutdown(read&write) on my TCP socket and then I try to connect it to another destination.
- Unfortunately, I've got an error "EISCONN". How can I fix this?
Thanks a lot for your help.

From: Rob Seace

No, you can't really "reuse" a socket like that, I don't think... Just close() the old socket, then create a new one with socket(), if you need a new one... I don't think a call to socket() is going to produce any noticable overhead; and, certainly no more than whatever calls you might need to do to 'disconnect' the socket, so you can reuse it, I wouldn't think... *shrug*

Add a Comment



New Questions: Ada Language Unix-based "Datagram Link Program" Type Error When Porting To And Recompiling On A Linux Machine

From: Stephen Silvey

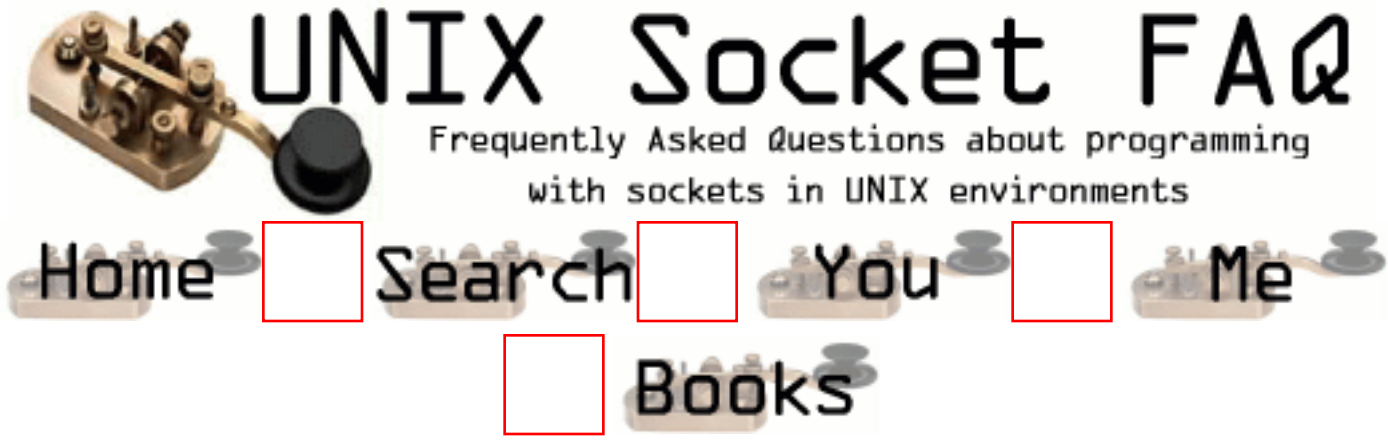
I am attempting to port some Datagram Link Code, originally written in Ada for an SGI IRIX machine, over to a Redhat 7.0 Linux machine, and I am getting the following Ada compiler error(s) when recompiling for the Linux OS:

```
error: expected type "OS_Long" defined at o_s_interface.1.ada.
error: found type "Integer_32" defined at infra_types.1.ada
error: ==> in call to "Htonl" at o_s_interface.1.ada
```

Can anyone tell me about this error, or what I can do to correct it? I realize that there probably aren't that many Ada programmers out there reading this FAQ page, so think about it in terms of the C Language, then.....

Any help would be appreciated!

Add a Comment



New Questions: call to write blocks forever

From: Franck

Hello,
my server program, after working well during about 30 seconds
suddenly block forever in a call to write.
I used 'strace' to try to understand what's hapening. I call
write in a loop as follows :
(taken from Richard Stevens's book)

```
ssize_t Writen(int fd, const void *vptr, size_t n)
{
  size_t nleft;
  ssize_t nwritten;
  const char *ptr;

  ptr = vptr;
  nleft = n;
  while( nleft >0 )
  {
   if( (nwritten = write(fd, ptr, nleft)) <=0 )
   {
       if( errno == EINTR)
       {
nwritten = 0;
       }
       else
       {
           return(-1);
       }
   }
   nleft -= nwritten;
```

```
   ptr += nwritten;
  }
  return(n);
}
```

All my sockets are set blocking (default mode).
Here is the final line given by strace in which the program is hanging :

write(7, "4 0 5.5 99.6 288.33 147.5 669 15"..., 174) = 151

it looks like the write function couldn't write the entire message (returns 151) but as I call write in a loop it shouldn't hang forever.

Notice that a few clients (4 exactly) generate at high speed messages to the server, which server must broadcast, all that works well, but unfortunately only 30 seconds.

Thanks in advance

Add a Comment

New Questions: persistent connection to a web server

From: Sener BALCI

When I do a GET to a web server, I lose my connection.
How can I have a persistent connection with a web server?
Thx in advance

/Sener

From: Hector Lasso

GET /your_page.html HTTP/1.0
Connection: Keep-alive

(two CRs)

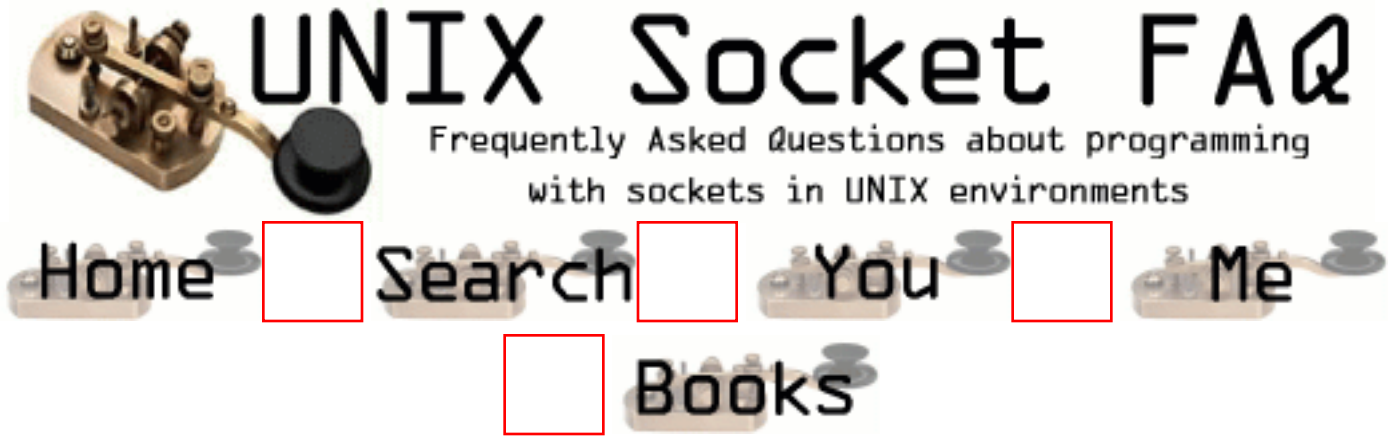
take a look at RFC 1945 and RFC 2068

Good luck

From: Ched

Note: the latest RFC regarding the HTTP/1.1 protocol is RFC2616.

Add a Comment



New Questions: Windows / Unix Socket Communication

From: Paul Murray

Sorry to be a learner, but I am only just getting into the world of sockets.

Although this FAQ has answered practically everything I needed to know about Unix sockets (including references and sample code), I now need to obtain similar references to windows sockets ....

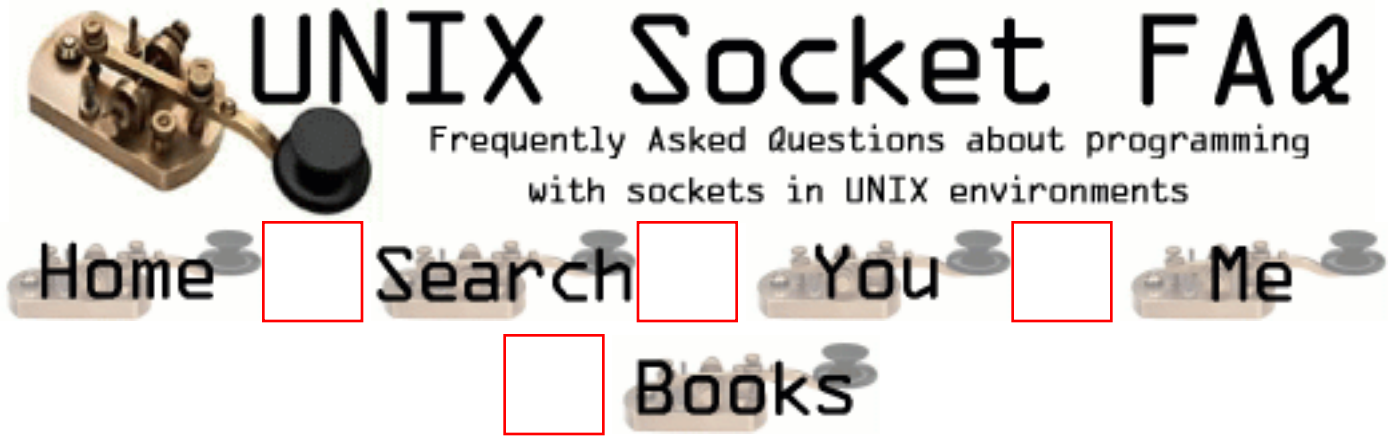
Can anyone point me in the write direction for a good windows socket reference ???

Thanks in advance,
Paul.

From: Daniel Gustafson

You should have take a look at http://www.allapi.net/ which is a great listing of Windows API calls. They have tutorials and examples on socket communications through Windows. Although there are ActiveX controls to do the socket work for you in f.e VB, the Windows API will perform better.

Add a Comment



New Questions: DANGER : big bug in Select() !!

From: tybins

I took the server example from Stevens page 168
(TCP server using a single process and Select)
and made a client to test it, it works very well
as long as the clients are not sending messages fastly, but
even with only two clients a problem appear when the
sending speed is high :
I launch a first client which keep on sending messages to
the server, then I launch a second client, his descriptor
is saved, back to the Select we are woken up, then the
FD_ISSET macro tells that this new socket is readable
and when retrieving the message, it appears to be the
message sent by the first client on another socket.
The problem is directly related to the speed at which the
clients are sending messages because if I reduce this speed
everything works well.
Is there a well known bug for select.
I spent so many time checking my code and now I don't know
were to look for, can anyone help ?
(I use Linux Suse 7.0)

From: Wes

1) Which Stevens book? UNPv1 has code for pselect() on page 168, but it's certainly not a complete example.

2) This is almost certainly a bug in your code. You're saying that your reading the data from the wrong client. Select has nothing to do with what data you're reading.

3) Shorten your code to the smallest possible length to show the "bug", and post it for comment.

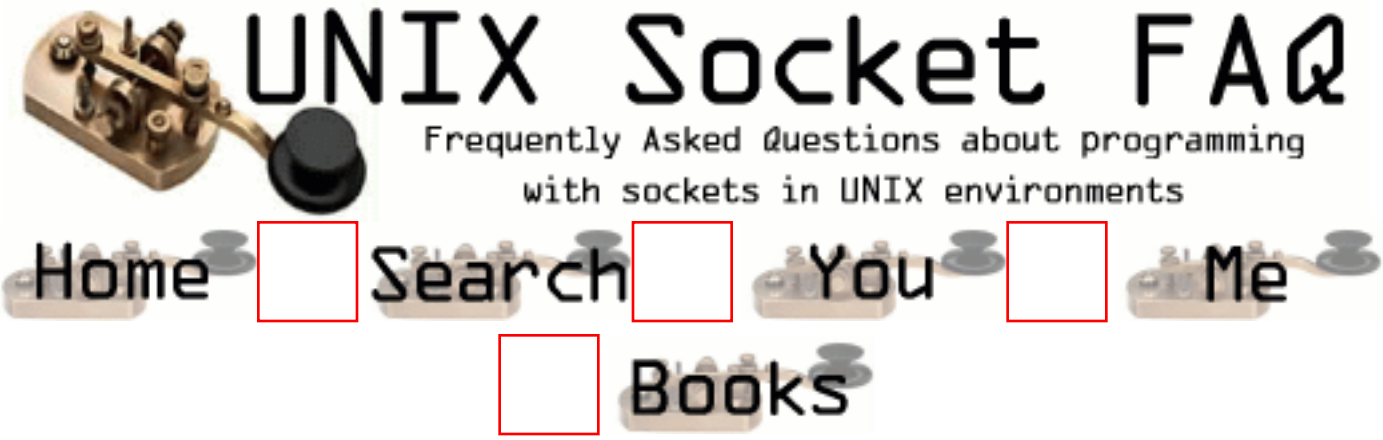
Wes

From: jos

i dont know about linux, but unix system v, select is a library call, not a kernel level call. consequnetially it is not atomic in nature. that could be the cause of your problem, especially since it looks like a race condition

---

Add a Comment



New Questions: Socket on Unix

From: Gurumurthy

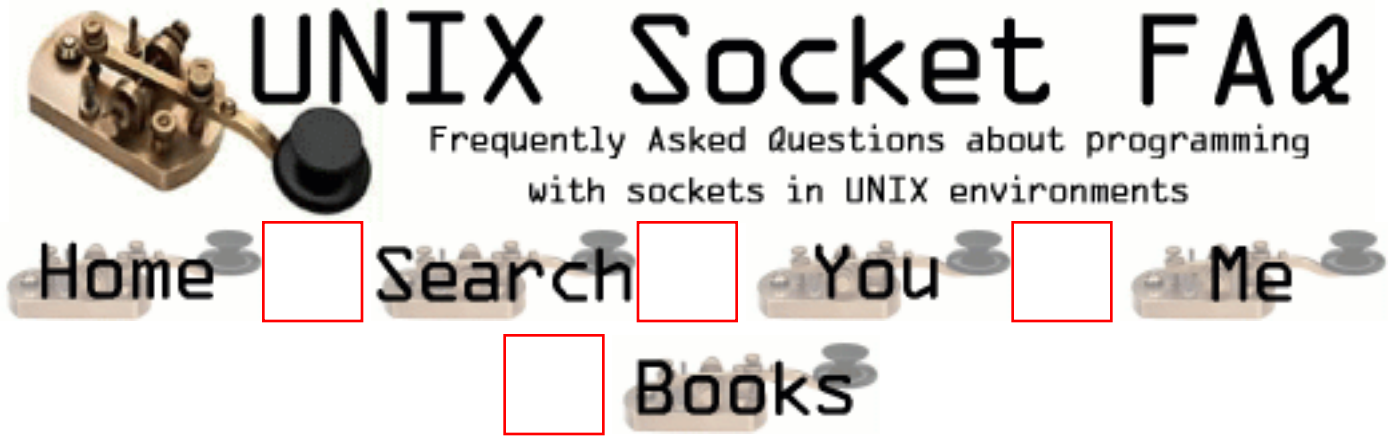
How to find out the IP address of the unix machine?

From: Sun

>>nslookup
> www.nba.com
Server: remus.rutgers.edu
Address: 128.6.13.3

Name: nba.com
Address: 129.33.0.36
Aliases: www.nba.com

Add a Comment



New Questions: rebinding bound socket

From: Khubilai

I have an socket (UDP) that is bound to a specific port. What is the proper way to release the bound port and bind the socket to another port?

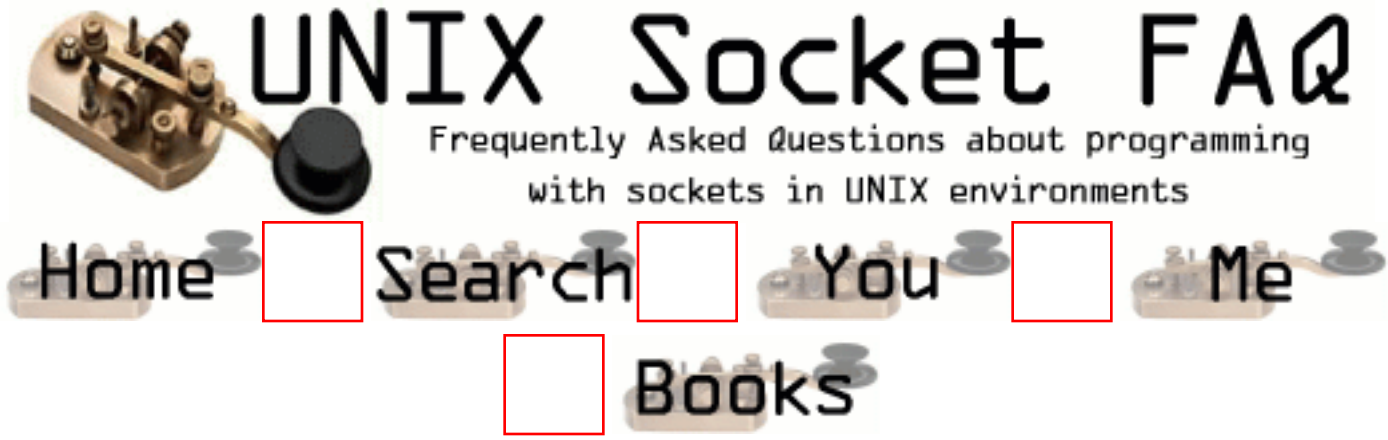
From: Ched

I think the only way to do this is to close the socket and create it and bind it again.

If you try to rebind your socket you will get systematically an EINVAL error, meaning the protocol stack does not support rebinding.

I do not know if there are protocol stacks enabling socket rebinding.

Add a Comment



New Questions: How can a process know its hostname ?

From: Zulius

In trying to simulate FTP i faced this problem. In FTP whenever there is a tranafer of data involved, the client creates another socket, exclusively for data trnsfer. Now the client sends its IP address and the port number on which it wants a socket to listen, to the server. So my question is how can the client get the IP address of the host on which it is running, without passing any parameter ? The functions gethostname(), gethostbyname() dont work because they already require the name of a host.

From: Rob Seace

This basic question is covered in the main FAQ, in question 2.24...

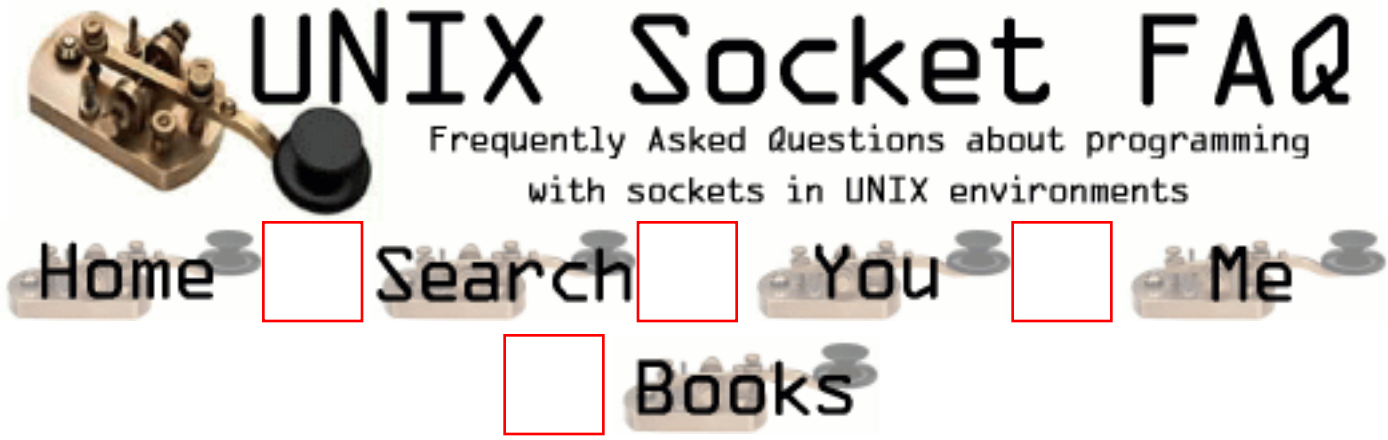
But, I'm not sure what you mean by gethostname() requiring a hostname; it requires nothing, and gives you the current host name, in the buffer you pass it...

However, if you only require an IP, you probably have a much simpler situation on your hands, anyway... If you already have an established socket connection, you can easily find out the IP of your end of it, by calling getsockname(), which will give you the full sockaddr_in struct that is bound to your end of it, which will contain both your local IP and port#... It's the companion to getpeername(), basically... And, that's probably all you really need...

From: God

ass

Add a Comment



New Questions: Sending info down sockets

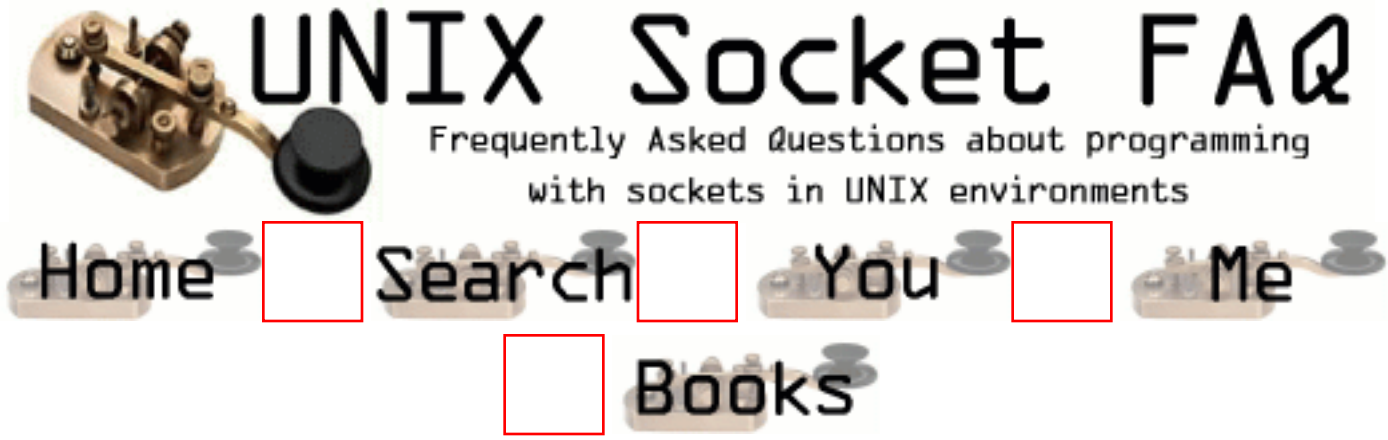
From: bal

we have a set up within our public_html directory and have written a basic concurrent server. Using socket coding what is the coding required to access such pages with netscape?

How do we implement SSI translation into the server?

PLEASE HELP SOS !!!!!

Add a Comment



New Questions: Sending info down sockets

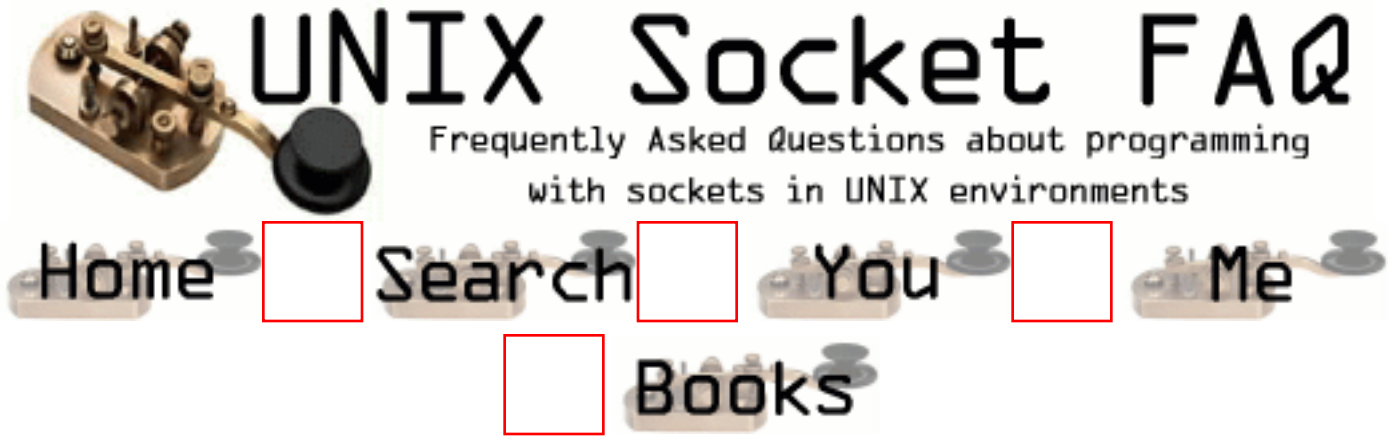
From: bal

we have a set up within our public_html directory and have written a basic concurrent server. Using socket coding what is the coding required to access such pages with netscape?

How do we implement SSI translation into the server?

PLEASE HELP SOS !!!!!

Add a Comment



New Questions: GET Protocol

From: scooby doo

how do u implement the GET Protocol into the below web server.

any help would be wicked

cheers

```
#include <unistd.h>
#include <iostream.h>
#include <cstddef>
#include <string.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <netdb.h>
#include <pthread.h>
#include <cstdlib>
#include <sys/time.h>
#include <sys/types.h>

extern void *serve(void *);

int main(int argc, char *argv[]) {
    unsigned int cliaddrlen, myport;
    int stat, sock, newsock, r, rc, pages ;
    struct sockaddr_in myaddr, cliaddr;
    struct hostent *hostptr;
    char *hostip, response[512], myname[256], fname;
```

```
    pthread_t th;
    char *datestr;
    struct timeval now;


    if(argc != 2) {
      cerr << "usage: server-prog port-num\n"; exit(1);
    }

    // establish server socket
    sock = socket(AF_INET, SOCK_STREAM, 0);
    if(sock < 0) {
cerr << "server cant open socket\n"; exit(1);
    }
    myaddr.sin_family = AF_INET;
    myaddr.sin_addr.s_addr = htonl(INADDR_ANY);
    myaddr.sin_port = htons(atoi(argv[1]));

    stat = bind(sock, (struct sockaddr *) &myaddr, sizeof(myaddr));
    if(stat < 0) {
cerr << "server cant bind socket\n"; exit(1);
    }

    // set socket passive
    stat=listen(sock,5);
    cout << "ready for u muppets\n";
    if(stat < 0) {
cerr << "server cant make socket listen\n"; exit(1);
    }


    cliaddrlen = sizeof(struct sockaddr_in);
    newsock=accept(sock,(struct sockaddr *) &cliaddr,&cliaddrlen);
    if(newsock < 0) {
cerr << "server accept error\n"; exit(1);}


    while (true) {
    cliaddrlen = sizeof(struct sockaddr_in);
    newsock=accept(sock,(struct sockaddr *) &cliaddr,&cliaddrlen);
    if(newsock < 0) {
cerr << "server accept error\n"; exit(1);}

    r = pthread_create (&th, 0, serve, (void *)newsock);
    if (r !=0) {cerr << "thread create failed \n";}
```

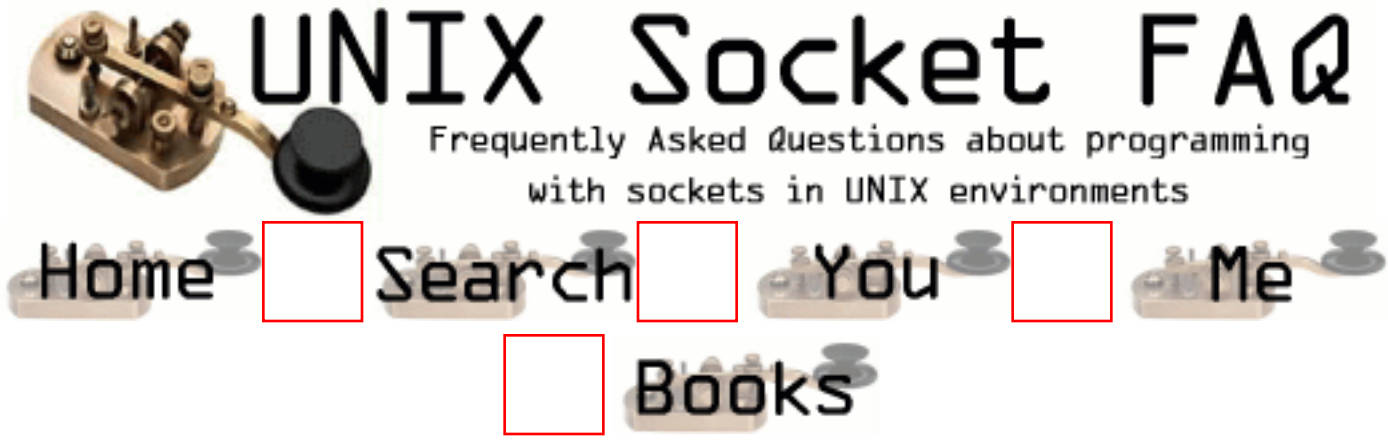
```
    }


    hostip = inet_ntoa(cliaddr.sin_addr);
    cout << "server: connect from IP " << hostip << "\n";

    hostptr = gethostbyaddr((char *)&cliaddr.sin_addr,
sizeof(struct in_addr),AF_INET);
    if(hostptr == 0)
cout << "server: client unknown name\n";
    else
      cout << "server: client name is " << hostptr->h_name << "\n";
    close(newsock);

     gettimeofday(&now,0);
        datestr = ctime(&now.tv_sec);
        write(newsock,datestr,strlen(datestr)+1);
      close(newsock);

        // read the response on same connection
    // and write it to standard output
    do {
        rc = read(sock,response,512);
write(1,response,rc);
        if(rc < 0) { cerr << "cant read\n"; exit(1); }
    } while(rc > 0);


    strcpy (myname, "public_html")
    strcat (myname, "index.html")
    pages = open (myname,O_RDONLY)
    send (pages, newsock, 0)


    close(sock);
}


void *serve(void *sockid) {
  int s = (int)sockid, rc;
  char buffer[2048];

  pthread_detach(pthread_self());
  rc = read(s,buffer,2048);
  while(rc > 0) {
```

```
    write(s,buffer,rc);
    rc = read(s,buffer,2048);
  }
  if(rc < 0) { cerr << "read failed\n"; }
  close(s);
  return 0;
}
```

From: b

thinks program looks like rubbish to me

Add a Comment



New Questions: get multiple documents from www server

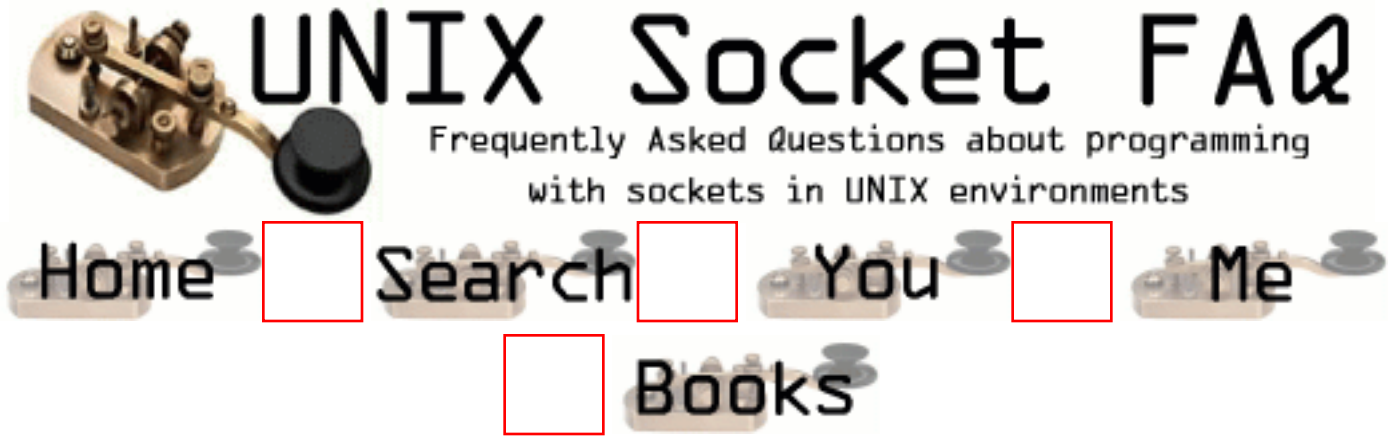
From: dzhan

i don't want server to close connection when i finished
fetch a document, because i need to get more.
how can i do this?

From: dzhan

this question has been solved by 388,
sorry.

Add a Comment



New Questions: problems receiving UDP Broadcast

From: Atenea

I am trying to receive packets (datagram) from a vxworks machine (can`t change machine configuration) wich sends packets to a x.x.x.0 address. Recvfrom is blocked when trying to receive those packets. What I have to do to solve this problem?.

Thank you very much.

Add a Comment

UNIX Socket FAQ

Frequently Asked Questions about programming with sockets in UNIX environments

New Questions: To change, my IP address.

From: Andrés

Hello;
I want, to change my IP address, when I connect to the Web Server, but I don't kown.

Somebody kown ,how... thanks.

From: Mahafuz

would you need to explain it a little more??

In linux the following commands ,

ifconfig eth0 down
ifconfig eth0 up XXX.XXX.XXX.XXX

changes the ip address of your eth0 interface from the old one to the new ip address XXX.XXX.XXX.XXX.

probabily you want to do it by code?? then pls explain
the question.

From: Andrés

I know to read, my IP address, with the bellow code:
char *hostAndres;
struct hostenv *andres;

andres=gethostbyname((char *) "Lisa");
cout << "The name of the host is: " << andres->h_name << endl;

cout << "The IP is: " << inet_ntoa(*((struct in_addr*)andres->h_addr << endl;

but I can to change...

Anyone please help!! . Thank you again!
My english is not so good...

From: Hector Lasso

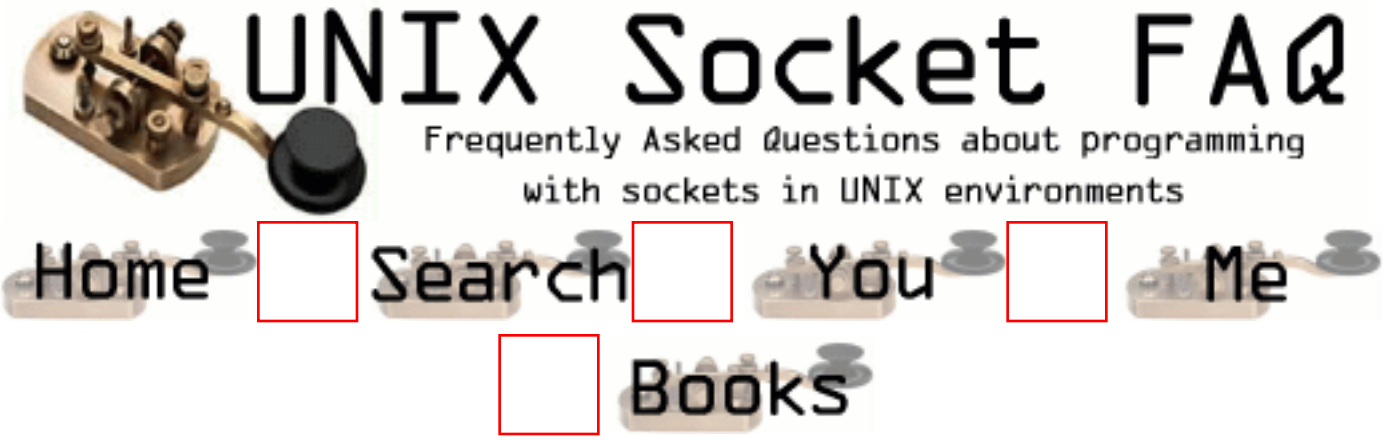
Andres,

Espero haber entendido lo que quieres hacer:
Abrir conexiones con otros servidores utilizando una dirección diferente a la que tienes, o simplemente cambiar tu dirección IP.
Si es lo primero entonces no es posible con sockets STREAM o con DATAGRAM. Te sugiero que intentes utilizando RAW sockets. Eso si, si tu cambias la dirección IP de los paquetes que envías no recibiras los paquetes de respuesta a menos que pongas una dirección IP que este dentro de tu LAN y utilices algún tipo de sniffer. Además, es posible que los paquetes recibidos por el host dueño de la dirección ficticia sean respondidos con paquetes RST lo que haría que se cierre la comunicación automáticamente.

Si es la segunda, la respuesta de Mahafuz es suficiente, puedes utilizar "system()" para ejecutar estos comandos, teniendo en cuenta que el programa debe correr con los suficientes privilegios para realizar esto.
// Cambiar direccion IP
system("/sbin/ifconfig eth0 down");
system("/sbin(ifconfig eth0 up 192.168.10.10");

Espero que ayude.

Add a Comment



New Questions: socket and rpc

From:

What are differences between socket and rpc?

Add a Comment

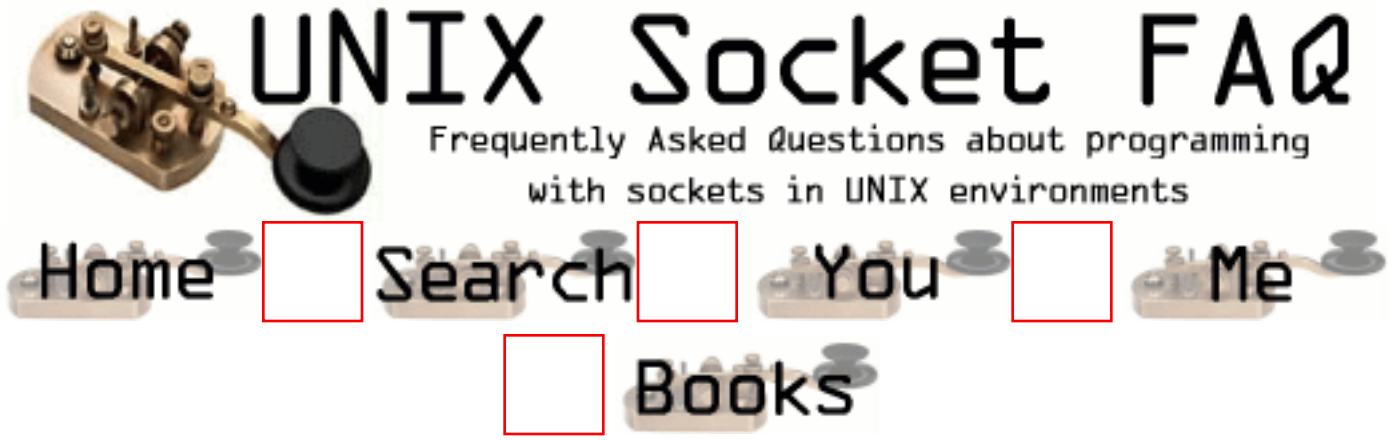
UNIX Socket FAQ

Frequently Asked Questions about programming with sockets in UNIX environments

New Questions: What is "IDMaps" and how to use it in indentifying the closest mirror site?

From: Xu,Zhong

Add a Comment



New Questions: fprintf error code

From: Jessy

May I please know what does error 47 means when fprintf is executed to write into a file from buffer?

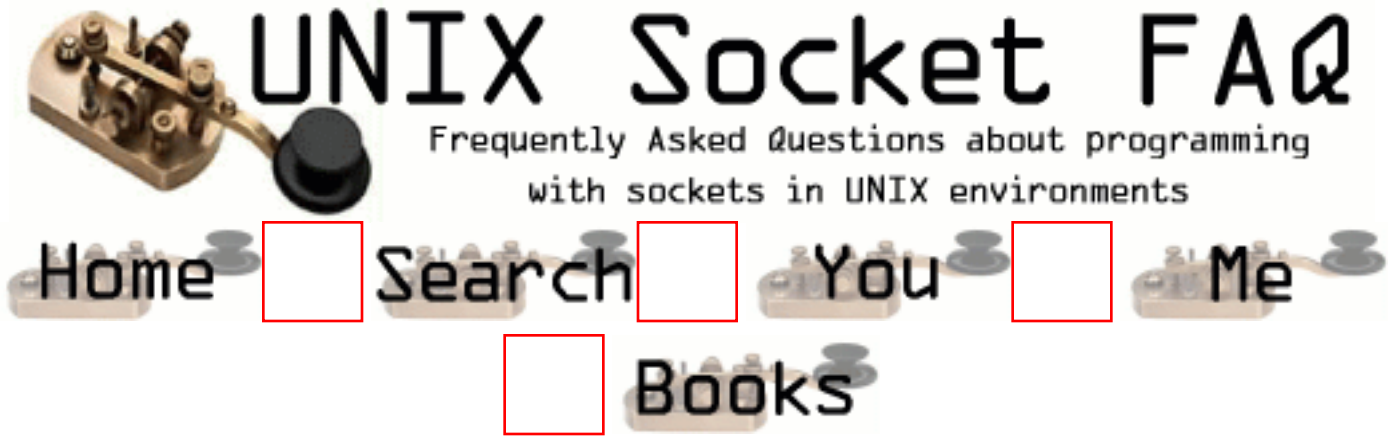
From: Hector Lasso

How did you get that error?
If fprintf() returned that number, it means that the function wrote that number of bytes in the file.

If you just need to know what the number in errno means use something like:
printff("The error was %d -> %s\n", errno, strerror(errno));

Hope this helps

Add a Comment



New Questions: % sign in a file read by my SOCKET server

From: Jessy

I need to know how to "escape" a "%" sign from a message being read by my socket server. Can someone please help?
I'm in a desperate situation.

From: frankv

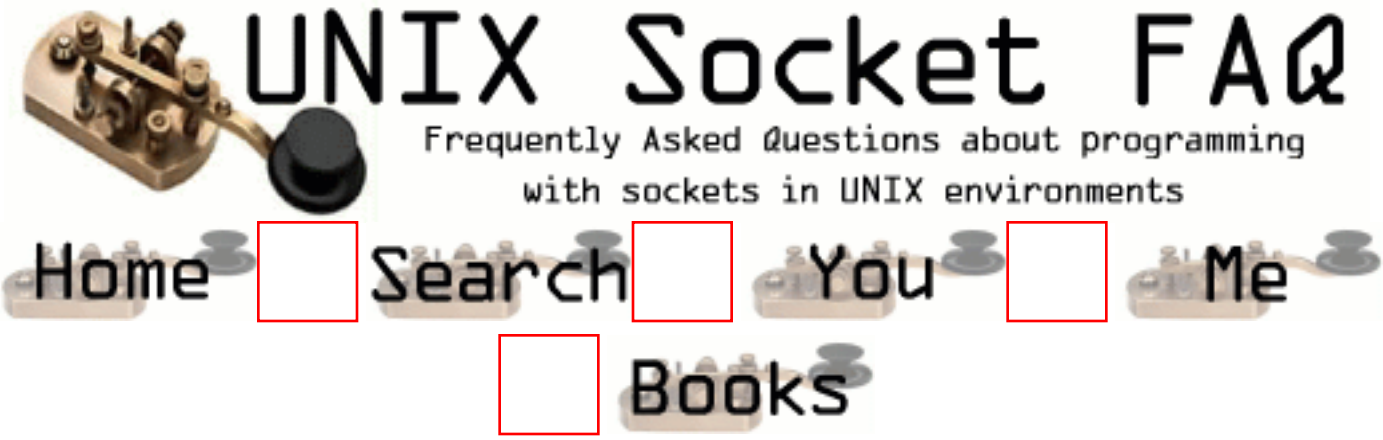
My guess is that you the % is being interpreted by printf or something becuase there should be no need to escape the %.

char string[1024];

ReadStringFromSomewhere(string); /* string now contains % in it */
printf(string); /* fails */
printf("%s", string); /* works */

Other then that you need to be more specific about why this is a problem, because it doesn't seem to be a socket issue.

Add a Comment



New Questions: socket 7 processor

From: maricel avena

how do we used socket 7 processor on super 7 socket? why?

From: Eduardo Roldan

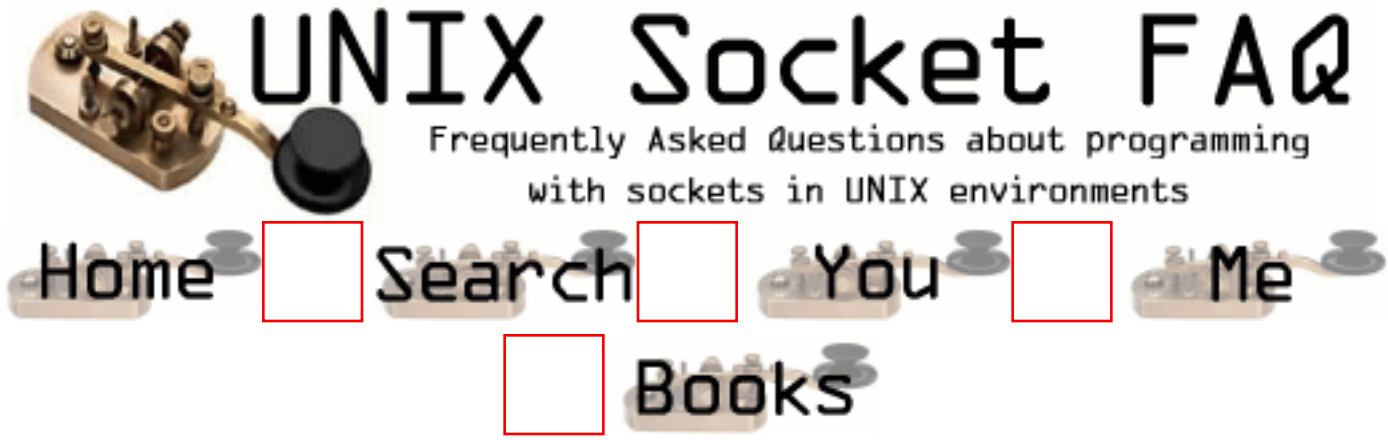
You need to use:

#include "motherboard.h"

void sock_in = SUPER_SOCKET_7
setsockopt(msock, SOL_SOCKET_MB, SOL_UPGRADE_SOCKET, (void *) &sock_in, sizeof(sock_in);

Then insert the CPU.

Add a Comment

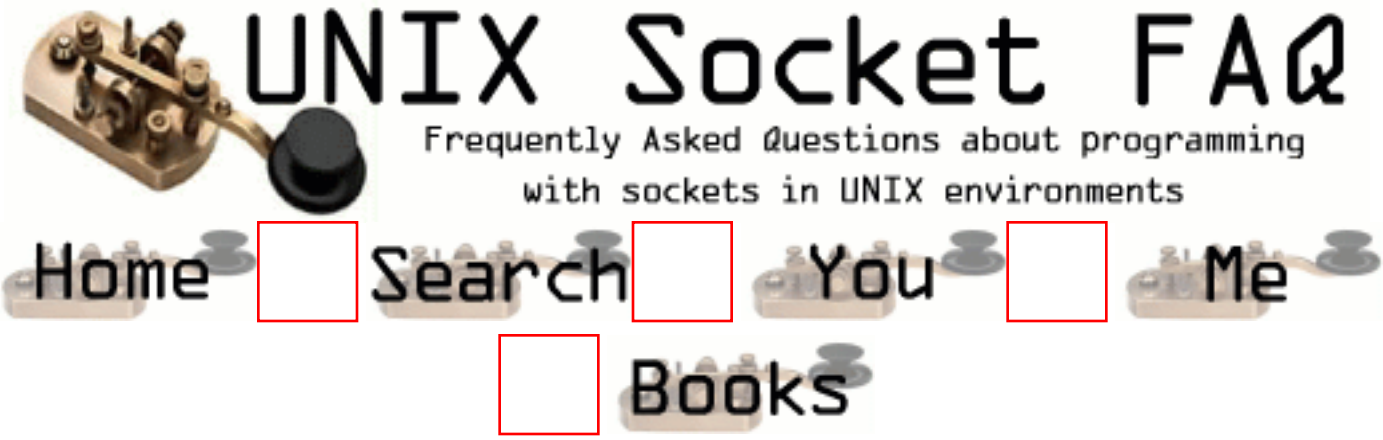


New Questions: Need help in compilation error2001

From: amadas

HI Everybody,
I knew this FAQ is for Unix enviroment but I need help in
windows to compile very simple creating socket program
I get it No error No Warnings to compile but when I link the
program I get:
winsock1.obj : error LNK2001: unresolved external symbol _socket@12
Debug/winsock1.exe : fatal error LNK1120: 1 unresolved externals
Error executing link.exe.
I appreciate your support.
My simple code :
#include <winsock.h>
SOCKET PASCAL FAR socket (int af, int type, int protocol);
#define af PF_INET
#define type SOCK_STREAM
#define protocol 0
void main()
{
socket (af, type, protocol);
}

Add a Comment



New Questions: Controlling an unexpected socket close.

From: Daniel Gustafson

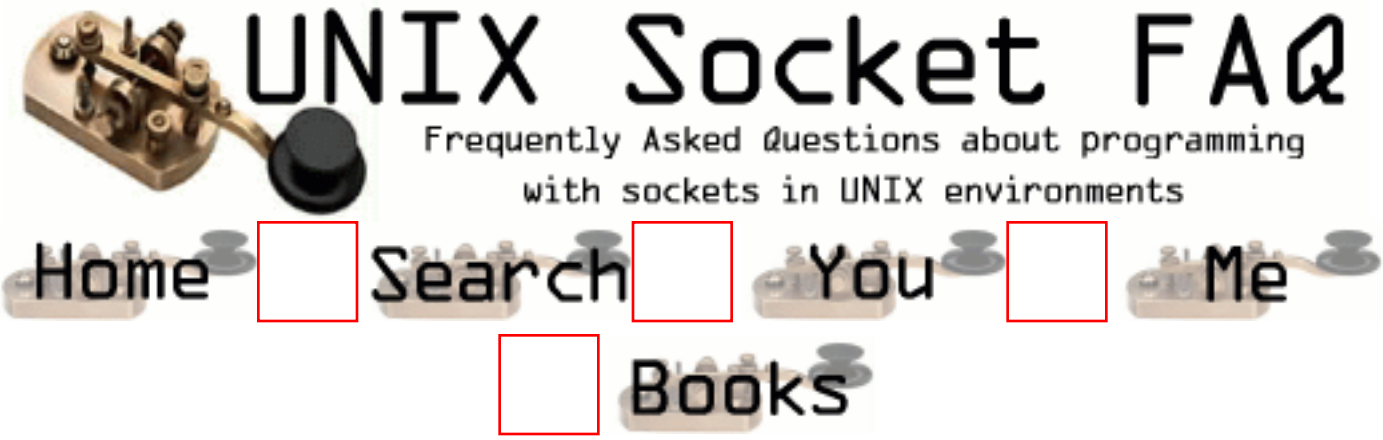
I am writing a small multithreaded server which I use standard telnet applications to interface with. The main loop is a while loop that reads a string from the client, takes the appropriate actions and read the next. One of these "commands" are CLOSE which shuts down the connection, allowing the server to accept new connections. My problem is when I shut down the telnet application without issuing my CLOSE command, the server gets stuck trying to read from the socket. Testing on error = read(blabla) doesnt get me out of this.

How do I detect these abnormal socket shutdowns and appropriately close the socket on the server ?

From: Rob Seace

You're saying the client closes its end of the socket, yet the server's read() does NOT return 0 to indicate EOF?? It sure should, and I've never seen it NOT do so, except in cases where the remote connection was not actually closed, but rather the client just disappeared off the Net, so to speak... (Eg: their machine crashed, or the route between you and then went down, or something like that...) Then, you get into SO_KEEPALIVE type issues... But, if the remote connection truly was closed, then there's no reason I can think of why your read() would not return 0... *shrug*

Add a Comment



New Questions: What is a Blocked socket ?

From: Mala Bankal

What is a 'blocked socket' and what is the differece between a 'blocked socket' and an 'unblocked socket' ?

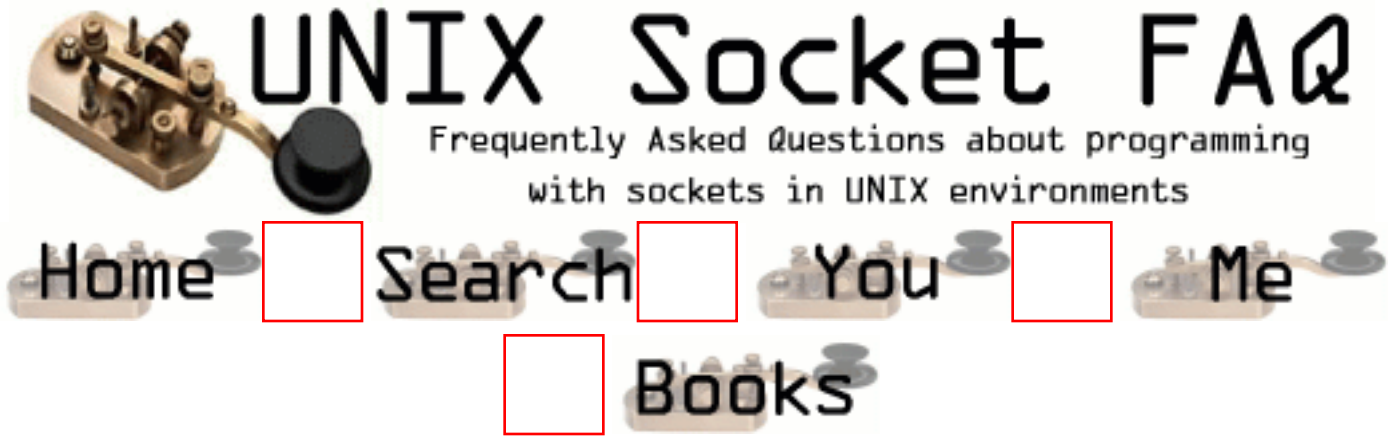
From: ornaz

the meaning of blocking is that specific operation does not
return until is has been satisfied
and nonblocking the operaiton returns immediatly wether the reasurce
avaliable or not

for example blocking send call does not return until the message
has been safetly stored away so the sender can freely reuse the sender
buffer buffer
OR
the recv blocking not return until the msg stored in the recive buffer

---

Add a Comment



New Questions: server/multiple clients

From: savi

Hi,
I am trying to write a server which handles requests from multiple clients at the same time. To handle 1 client I used the fork() call. But now i dont know how to proceed, if it has to handle several clients.
Any help in this regard will be very much appreciated.
Thanks,
savi

From: Hector Lasso

Hi,

There are different ways to serve many clients. Rob Seace makes a very good description of the advantages and disadvantages of them in <A HREF="http://www.lcg.org/sock-faq/detail.php3?id=428">Question 328</A>

You could use select() in one single process to serve all the clients, you could also fork() a child for each connection, or create a thread for each connection.

As you have already worked with fork() i'll tell you how you can do it very easily:

1) I assume you've already bound the socket, and created a loop that looks something like this

```
...
do {
  // ...
  psocket = accept(ssocket, ...);
  if (psocket > -1) {
    create_child(psocket);
    close(psocket); // Don't need this
```

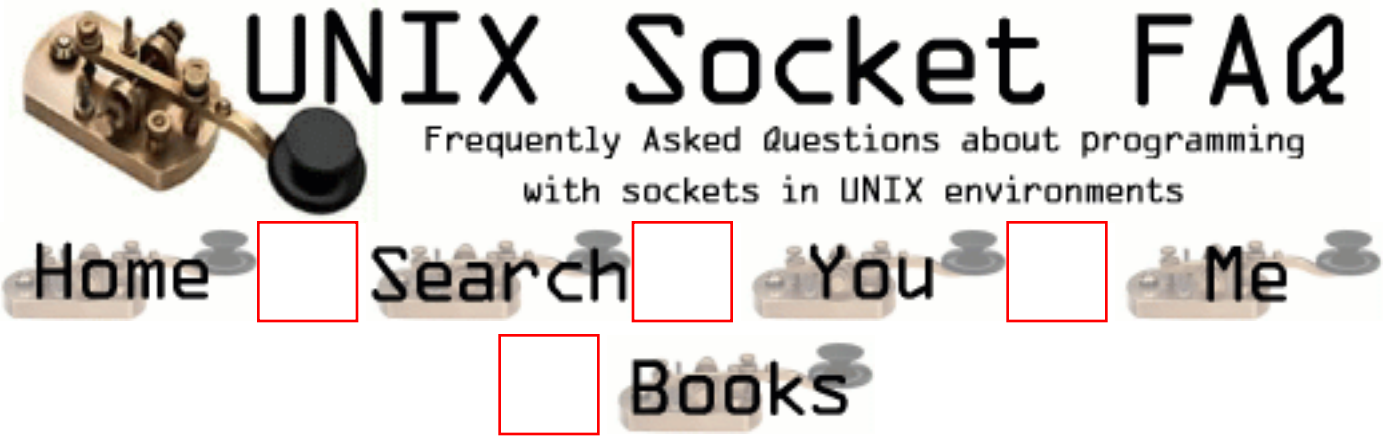
```
  }
  // etc...
} while (some_very_interesting_condition);

...

void create_child(int sock) {
  int cpid;
  cpid = fork();
  if (cpid == 0) {
    // Child process, it will serve a single connection
    close(ssocket); // don't need this...
    // Do your client-serving stuff here...
    // for example: exec() a program...
    // Once finished you have to exit here
    shutdown(sock, 2); // If you have not done so
    close(sock);
    exit(0); // If you return, you'll go back to the loop
  }
  else if (cpid > 0) {
    // Parent process
    printf("Forked child process %d\n", cpid);
  }
  else {
    printf("Something went wrong with fork() %d %s\n",
          errno, strerror(errno));
  }
}
```

Hope this helps
Regards

From: Hector Lasso

Sorry about the link: Question 328

Add a Comment



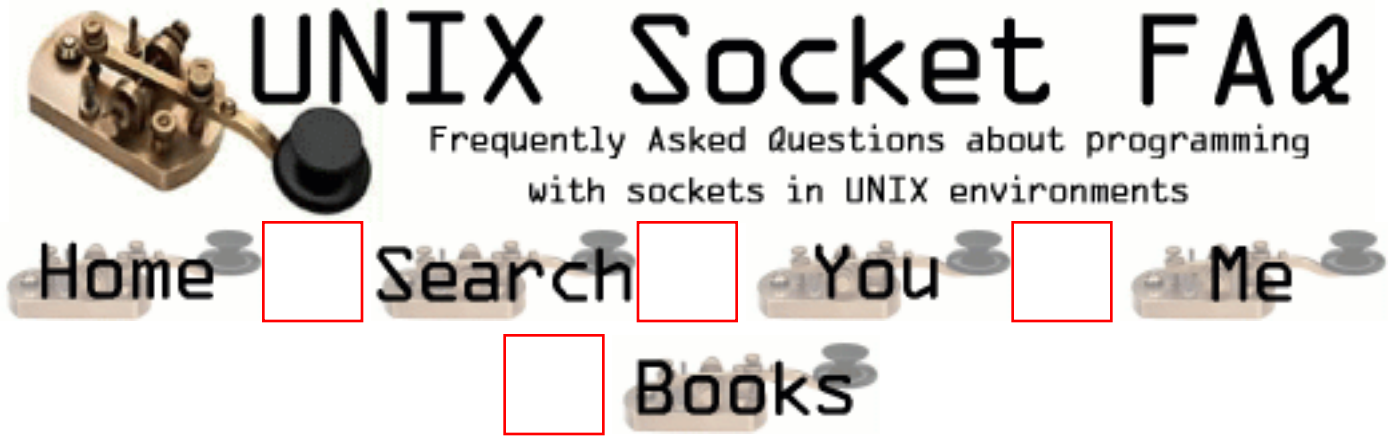
New Questions: Connection failures

From: Chris

I'm trying to connect a socket from a Solaris system and continue to get connection timed out errors. The socket setup works on NCR OS. I've verified services and hosts are setup correctly. Any ideas?

connect sock=4 sockaddr=-268437284 size=16

Add a Comment



New Questions: Socket Programming

From: Sunil Philip

Hi,
Iam using Linux server and NT client.How will i receive the contents of the database from the client using structures?
Not only that ,the server has to respond to multiple clients without using thread() or fork().

Add a Comment

New Questions: problem in client identification by server

From: suneet chandok

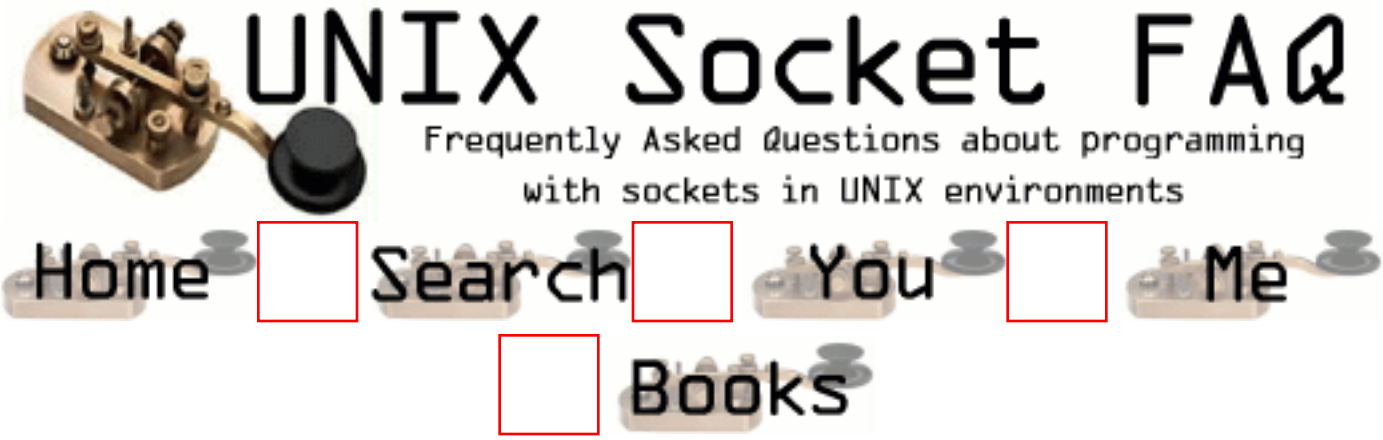
if a two or more clients are connected to server, how can server differentiate rather identify them to send messages

---

From: Satya

Once a client get connected to the server the server can easily detect what is the peer's IP address and its socket handle.
Server can simply store the details in memory for further use.

Add a Comment



New Questions: Connecting a socket throw a proxy with autentication

From: LFM

hi,

i'm developing a program that needs to connect a socket throw a proxy server.

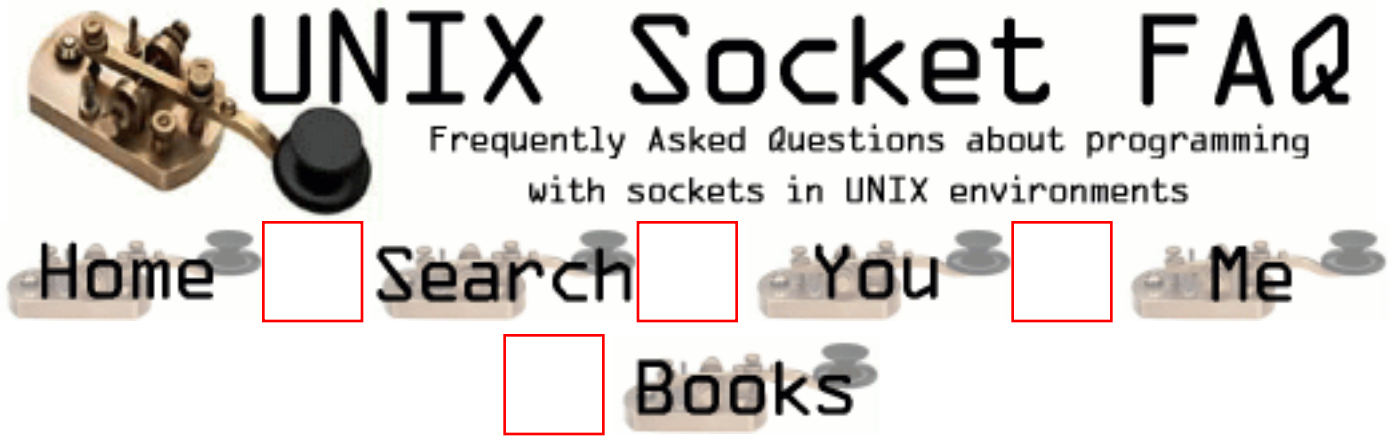
this particualr proxy server requires userid/password autentification.

any ideas ????

best regards

Luís Meira

Add a Comment



New Questions: How to test if a server closed my socket?

From: Mel Tan

I am using AF_INET type SOCK_STREAM for my socket and I want
to test if the server has closed my socket before I write
anything to it. How do I do this?

select() says that the socket is still open.
read() gives no error and blocks.
recv() gives no error and blocks.
fstat() says everything is okay.

But I know that my socket was closed by the server. Using
TCP would solve this easily, but I'm strictly restricted
to the setup above. What's worse, I have 1 week deadline
to do this.

I should also mention that if I use a timeout on select(),
to check the response time it would help. Except that my
server will only starts sending data after 4 clients have
connected to the server. So this poses a ridiculous dilemma.

Any help PLZ. Thanks. Mel.

From: Rob Seace

I'm not sure I understand what you're saying... First of
all, an AF_INET/SOCK_STREAM socket IS a TCP/IP socket, so
I don't understand your implication that you're not using
TCP... Did you mistakenly type "SOCK_STREAM" when you meant
to type "SOCK_DGRAM" (ie: you're using UDP, not TCP)??

If you're really using SOCK_STREAM, then detecting a closed remote connection can be done a few ways: Try to read() or recv() (the latter lets you do MSG_PEEK, so it's probably ideal for a test, where you don't really want to read any actual data), and if you get back 0 bytes, that's the EOF signal, indicating the connection was closed... (That requires that the remote connection be truly close()'d, though... If only half the connection was shutdown(), then you may or may not get the EOF on read()/recv(), depending on which half was shutdown...) Or, you could attempt to write()/send(), and check for failure, with "errno" set to EPIPE...

But, you seem to indicate that these don't work for you? If that's true, I can only assume the "SOCK_STREAM" reference must have been a mistake, and you're really using UDP, or something... In which case, there's no connection at all, so of course no way to detect "closing" it, since it was never "open" in the first place... If you're really using TCP, and these things don't work, then I have to assume either there's a problem with your code, or your OS's TCP/IP implimentation is broken, somehow... It might help if you posted the code that doesn't work... (Try to create just a very simple client and server, purely to demonstrate the concept of what you're doing, and show it not working... That's much better than mere snippets from real code, because matters don't get confused by irrelevent stuff in the real code, and people can pull down the full trivial code and try it on their own systems... PLUS, most of the time you'll discover that the trivial code will work just fine, which points to some other problem with your own code which is causing the failure...)

From: Mel

First thing you should ask me if I am dreaming. Yes I am. So, it's AF_INET and SOCK_STREAM. No UDP here.

So here's what I do and I've seen this in many codes.

On the server side:
server = socket(AF_INET, SOCK_STREAM, 0);
bind(server, ...);
listen(server, ...);

On the client side:
client = socket();
connect(client, ...);

After the two above is performed successfully, I do the following:
In Server: clientIN = accept();
In Server: close(clientIN);
In Client: write(client, ...);

So, the client has connected without a problem but the server just goes ahead and closes the accepted connection. Now the client writes to the server but the write doesn't give any errors that the other side of the connection is closed. The write is successful. If I do a read, it will block. So obviously the file descriptor is valid even
though the connection is closed on one end.

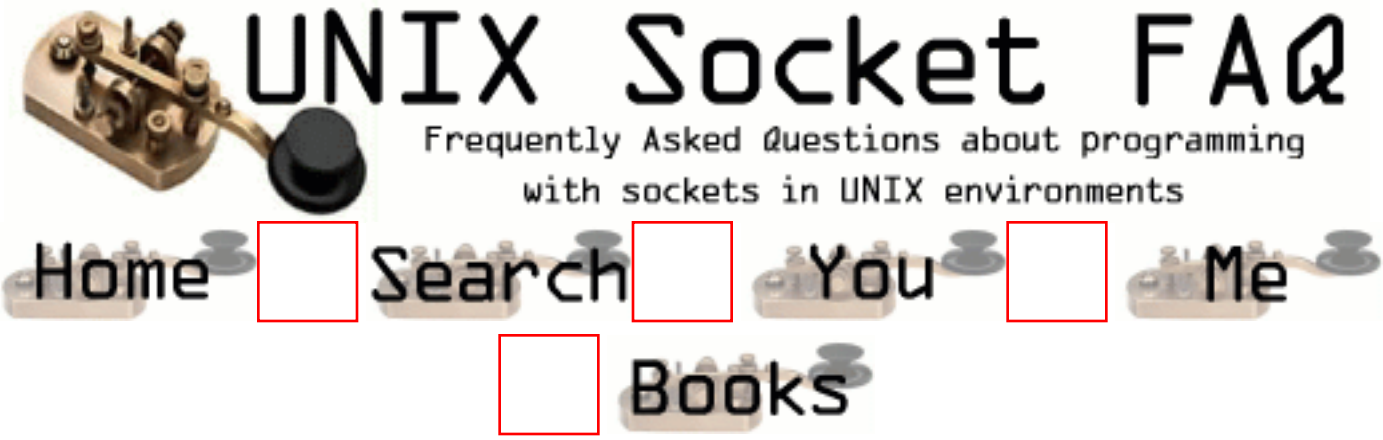
It's just that simple. My code is over 1500 lines and broken
up into C++ classes so needless to say, it is pretty fragmented and all over the place. On the plus side, there are no errors except one. Doing a select() and timeout is
not an option here. Is there a way to get an acknowledgement
from the server if the message I sent was read out of the
buffer?

From: Rob Seace

Well, the only thing that seems to make sense is that your client gets its write() in BEFORE the server actually close()'s the connection... Because, if the connection were truly closed at the time of the write(), then it just MUST fail with EPIPE, unless your OS is just broken, or something... (What system is this being run on, anyway?) But, if it gets the write() is before the close (which is highly possible, since the OS will accept the connection long before the server does its accept(), which will make the client's connect() succeed right away), then the written data will just queue up, waiting for the remote side to eventually read it... Now, when the server finally does the accept() and immediately close()'s the connection, that queued data will just be discarded... And, sadly, there's really no way for you to detect THAT, I don't think... Unless you establish some app-level ACK'ing of messages between client and server... However, you should be able to EVENTUALLY detect the fact that the connection was closed... Either on your next write() (after the close() actually takes place), or read(), or even YOUR close()...

You say a call to read() blocks? Does it CONTINUOUSLY block, even long after the server's close() was really done? That definitely should NOT happen... As soon as the remote close() is done, your read()/recv() should unblock and return 0, to indicate EOF... If it's really not doing that, then I'm not sure WHAT is happening... But, it sounds very odd... As I said, you'd probably be well off to create yourself a very simplified client and server, to duplicate what is happening, and verify that this simplified code still behaves the same way... If it DOES, post it... If it DOESN'T, then it means there's some other hidden bug in your original code that's responsible...

Add a Comment

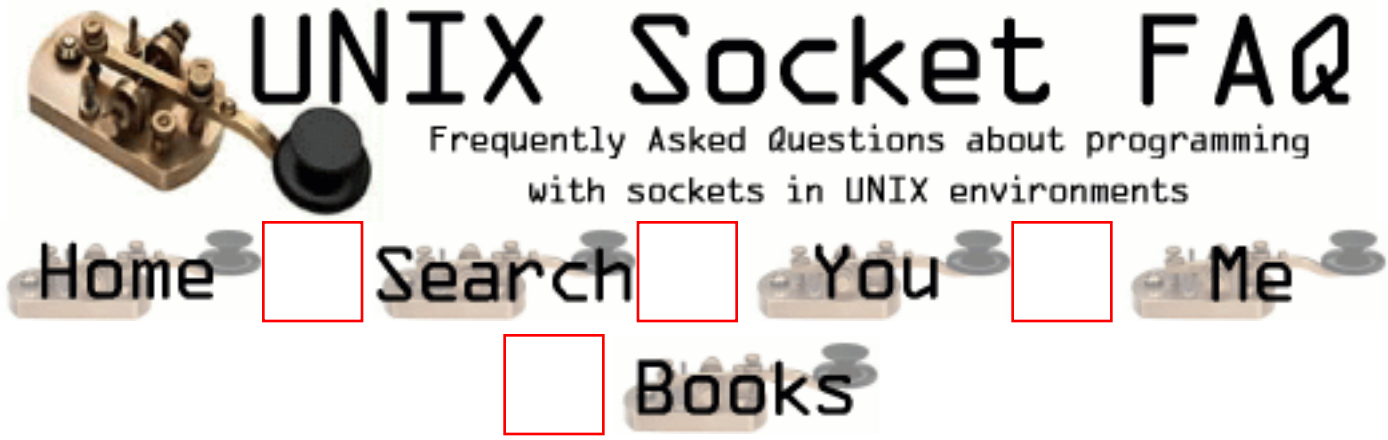


New Questions: Line down condition

From: ArK

Hi,
Is it possible to sense, when a cable is
plugged out from the network? Is TCP/IP
provides any signal when the cable is plugged out?
(Unix Platforms).

Add a Comment



New Questions: Select Read problem

From: Sameer Tilak

Hi All,

My name is Sameer. I am writing a client-server application. I am using
select so that the server can multiplex between all the socket connections.
I am having the flollowing problem:
Select always say there is Read active on a socket in fact when I have already r
read the complete message already.

I am herewith pasting my code. I will be thankful if anyone can help me out.
I am here with giving the basic required code:

void build_select_list(int serv_sock)
{ int listnum; /* Current item in connectlist for for loops */ /* First put together fd_set for
select(), which will consist of the sock
veriable in case a new connection is coming in, plus all the sockets we have
already accepted. */

/* FD_ZERO() clears out the fd_set called socks, so that it
doesn't contain any file descriptors.
FD_ZERO(&socks);
/* FD_SET() adds the file descriptor "sock" to
the fd_set, so that select() will return if a connection comes in on that socket (which means you
have to do accept(), etc. */

/* serv_sock is listening sock*/
FD_SET(serv_sock,&socks);

/* Loops through all the possible connections and adds those

```
 sockets to the fd_set */

for (listnum = 0; listnum < 5; listnum++)
{
  if (connectlist[listnum] != 0)
   {
     FD_SET(connectlist[listnum],&socks);
     if (connectlist[listnum] > highsock)
        highsock = connectlist[listnum];
   }
 }
}

while(1) {
build_select_list(serv_sock);
timeout.tv_sec = 1;
timeout.tv_usec = 0;
ready_sock = 0;
ready_sock = select(highsock+1, &socks, (fd_set*) 0,
(fd_set*) 0, NULL);

if (ready_sock < 0)
{
  perror("select"); exit(EXIT_FAILURE); }
  if (ready_sock == 0)
 {
  /* Nothing ready to read, just show that we're alive */
  printf("...");
  fflush(stdout);
 }
 else
 {
   printf("\n One more hit %d \n",ready_sock);
   read_socks(serv_sock); printf("\n Just now serviced %d
   \n",ready_sock);
  }
 }
}
```

I am marking each socket after connecting as non blocking

```
read_socks(int serv_sock)
{
int listnum;
/* Current item in connectlist for for loops */ /* OK, now socks will be set with whatever
```

socket(s)
are ready for reading. Lets first check our "listening" socket, and then check the sockets in connectlist. */ /* If a client is trying to connect()
to our listening socket, select() will consider that as the socket being 'readable'. Thus, if the listening socket is part of the fd_set, we need
to accept a new connection. */
if (FD_ISSET(serv_sock,&socks))
handle_new_connection(serv_sock);
/* Now check connectlist for available data */

/* Run through our sockets and check to see if anything
happened with them, if so 'service' them. */
for (listnum = 0; listnum < 5; listnum++)
{
  if (FD_ISSET(connectlist[listnum],&socks))
   {
     printf("\n Active : %d, %d \n", listnum,
           connectlist[listnum]);
    deal_with_data(listnum); } /* for (all entries in queue) */
}

select always show data is there so I always enfd up in calling
deal_with_data with the same socket eventhough I am properly reseting my
FDsets... I am not getting why this is happening ?

-Sameer.

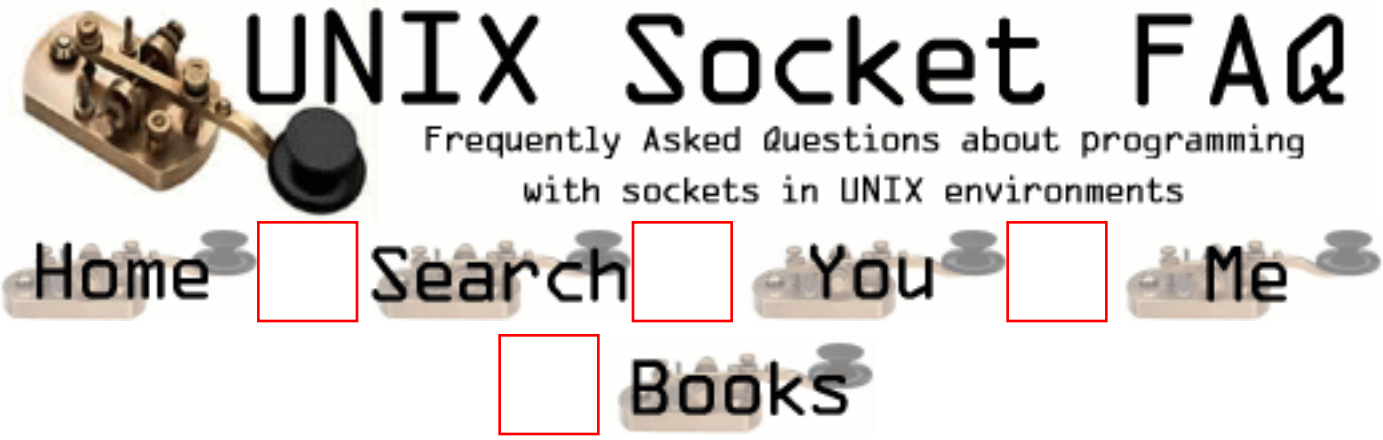
Add a Comment

New Questions: How does server control number of client---help

From: taohuang

Hi:
In my project, I need server control the number of client
the MAX_NUMBER is 5, when the sixth coming, server tell it's
busy, and some one disconnect, server've one available.
How can I do this.
Thanks for any help.
----new guy

Add a Comment



New Questions: how to detect SIGCHLD

From: taohuang

Add a Comment

New Questions: TCP Socket Read/Write With Two Threads

From: schang

Is it possible to read/write TCP socket concurrently by two separate worker threads? Both read/write channels belong to the same connection. If it's possible, are there'any side-effects, or areas I need to pay special attention? e.g. locking issues or external interrupt event like TERM signal?

Appreciated your help,
Simon

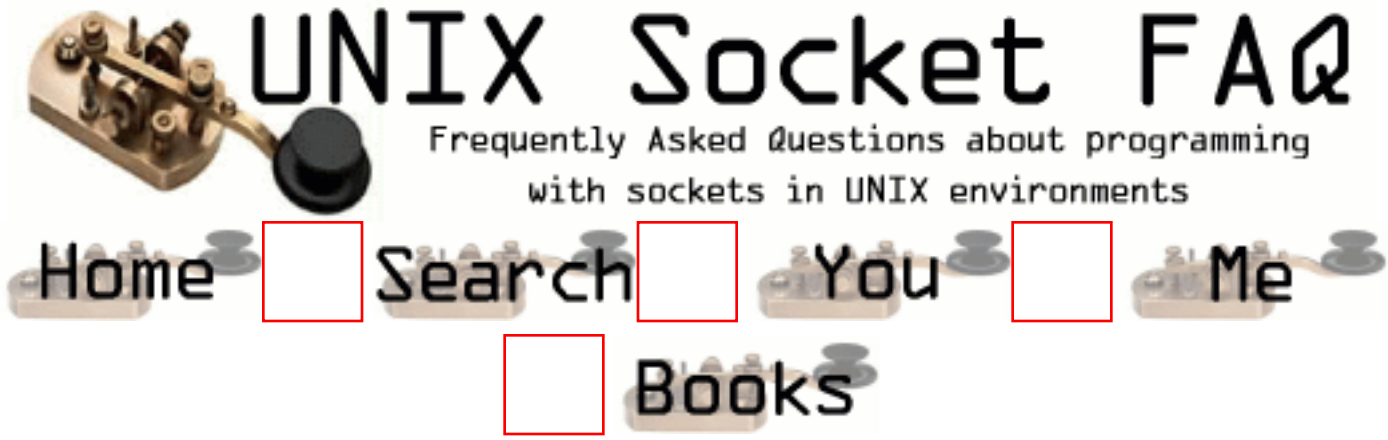
Client: C, POSIX, PThread on Solaris

Diagram:

TCP server <---> TCP client

TCP client <---> blocking read, timeout after 60 seconds
then write (thread 1)
<---> write every 30 seconds interval (thread 2)

Add a Comment



New Questions: Data Flow from socket call(s) to a network device driver

From: chandraiah P

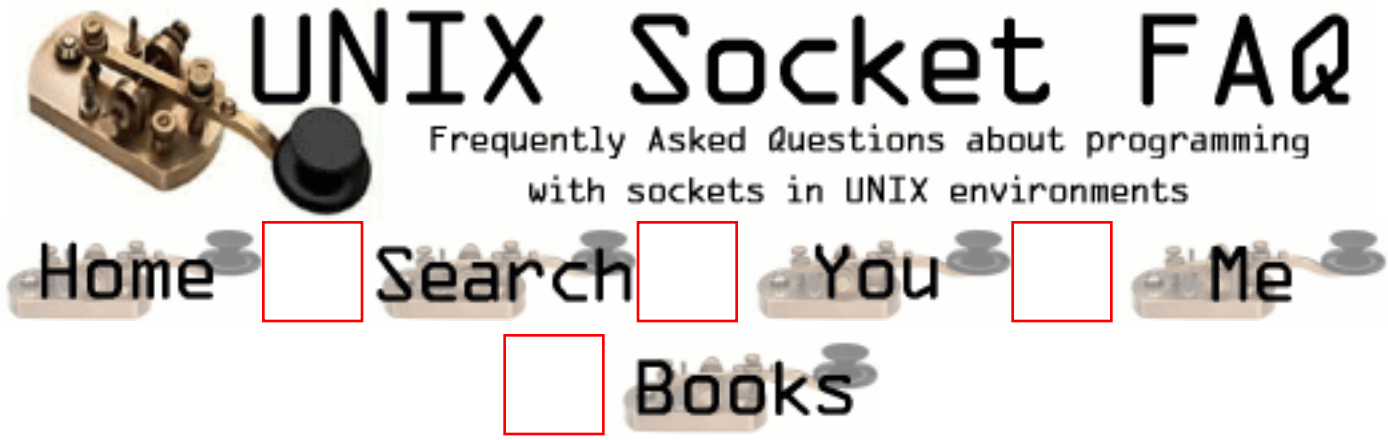
Dear

If any one knows how the Linux kernel maps different socket calls to a corresponding Network device driver functions please explain me.

like

socket call
|

Add a Comment



New Questions: Data Flow from socket call(s) to a network device driver

From: chandraiah P

Dear

If any one knows how the Linux kernel maps different socket calls to a corresponding Network device driver functions please explain me.

like

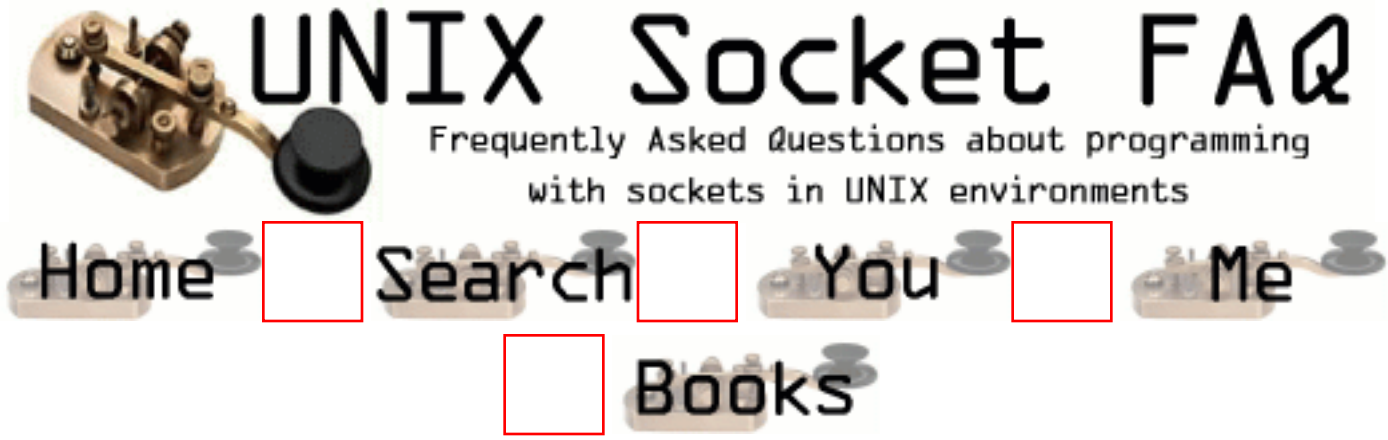
```
socket call (write() or send() )
   |
kernel
   |
Network device driver function (hard_start_xmit)
```

please give me repply soon if any one knows

thanking you

with regards
chandraiah

Add a Comment



New Questions: Data Flow from socket call(s) to a network device driver

From: chandraiah P

Dear

If any one knows how the Linux kernel maps different socket calls to a corresponding Network device driver functions please explain me.

like

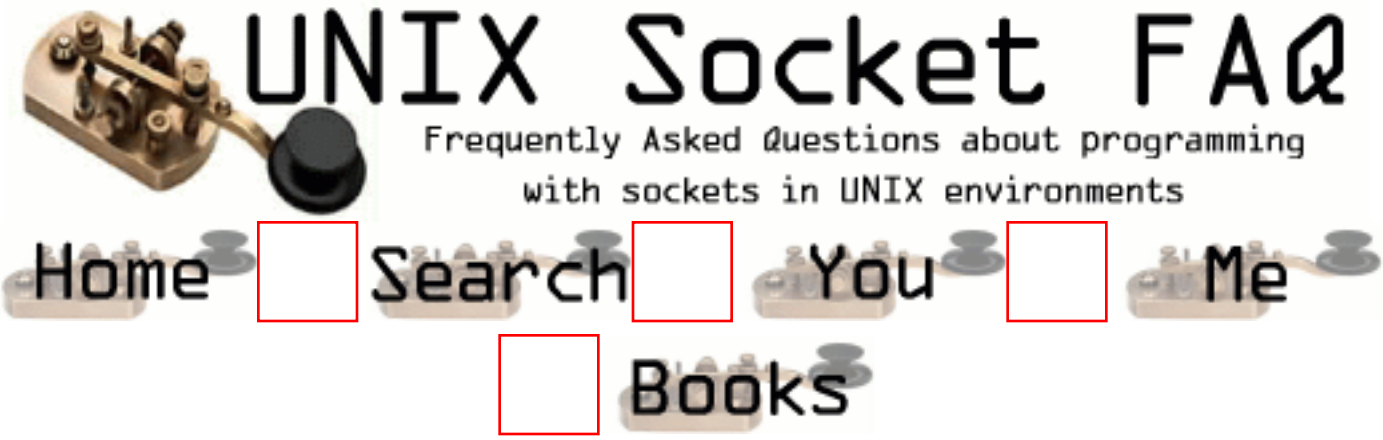
```
socket call (write() or send() )
   |
kernel
   |
Network device driver function (hard_start_xmit)
```

please give me repply soon if any one knows

thanking you

with regards
chandraiah

Add a Comment



New Questions: Select on a unconnected socket

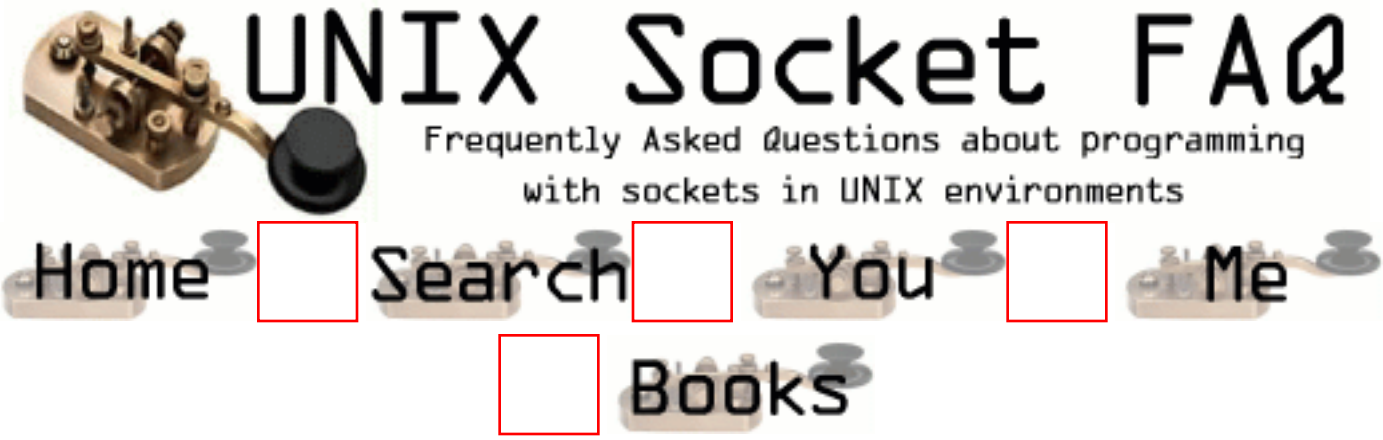
From: Indy De

Why does select call on an un-connected socket returns
read ready state? Is it how it is supposed to be?

From: Atul

I have also experienced this - but sorry I don't know why it happens.

Add a Comment



New Questions: What is client IP on a multi-IP machine?

From: Dave Zavetz

Let's say that I have a single PC with a number of NICs with each port having it's own IP address ... like this:
ethernet port 1 -> X.X.X.1
ethernet port 2 -> X.X.X.2
ethernet port 3 -> X.X.X.3, etc.
Now let's say that my application has data to send (to another PC) and I need the data to be sent over port 2 (X.X.X.2). How do I assign a client to a particular "outgoing" IP address?

From: Rob Seace

You just need to have your clients do an explicit bind(),
to bind their end of the socket to a specific IP, rather
than just let the bind() happen automatically for you (and,
out of your control) at connect() time...

From: Nick Smith

1. What are PCBs
2. The main thread sets up a pipe, then spawns two new threads, threadA and threadB. The main thread closes both end of the pipe. Can threadA and threadB now communicate via the pipe? Explain..

3. A thread does a chdir. What effect does this have on the other threads.

From: Rob Seace

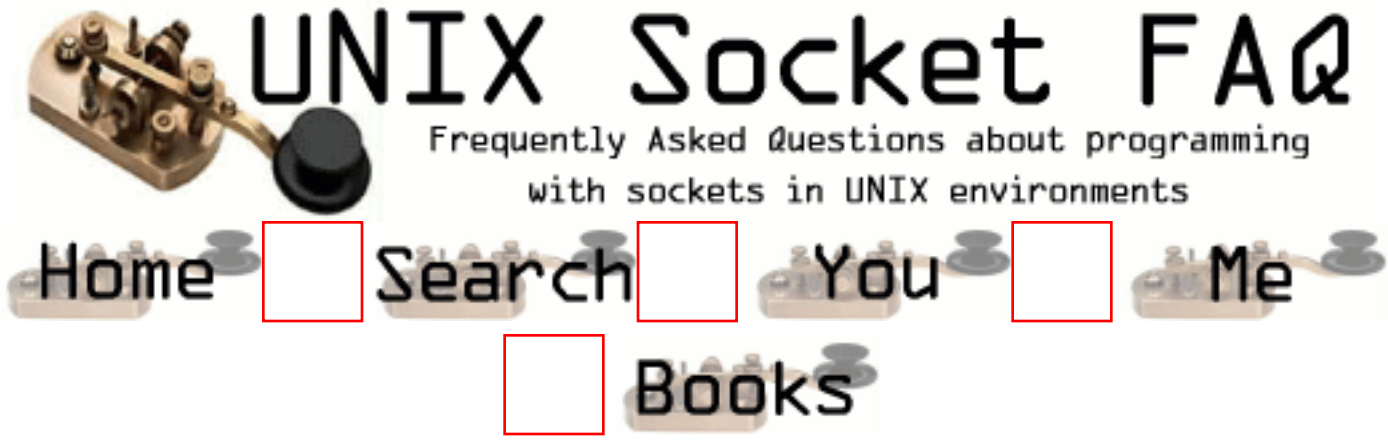
Um, what do any of the above have to do with the original
question about setting a fixed IP on servers with multiple
IPs?? (Or, for that matter, what do any of them have to do

with Unix sockets?? ;-))

But, to attempt answers, anyway:

1. In most contexts I've heard "PCB", it generally means "Printed Circuit Board"...
2. With threads, no, your proposed scenerio won't work; all threads share the same exact set of data, so if one of the threads closes down an FD, then all other threads similarly have that FD closed, now... With fork()'d children however, it would work fine... But, with threads, there's really no need to resort to using stuff like pipes to communicate anyway, since they are conceptually all part of the same big process, so they have much more efficient ways of communicating... (Eg: exchanging data via shared global/static variables... But, make sure to use proper locking to prevent access clashing...)
3. Similarly, I'm pretty sure that's going to change the current working directory for all threads... But, then, this isn't the multi-threaded programming FAQ, so don't complain too much if I'm wrong... ;-)

Add a Comment



New Questions: Can I increment an IP address "IP=IP+1"?

From: Alberto Fernandez

How can I go trhought a rang of IP's, in C?
How can I increment (in a unit) a "struct in_addr", in C?

I want to make a scan program who try to connect in one port on all host of my subnet in the form (x.x.x.0-x.x.x.255)

From: Rob Seace

You can't increment the struct in_addr itself, however you
can increment the 32-bit int "s_addr" that is the struct's
only member... But, you'll want to convert it to host
byte-ordering format, first... Eg:

```
long l;
struct in_addr in;

/* ... fill "in" somehow... */

l = ntohl (in.s_addr);
l++;
in.s_addr = l;

/* ... */
```

From: Rob Seace

D'oh! I forgot the htonl() call when assigning the value
back to the struct in_addr... That should be:

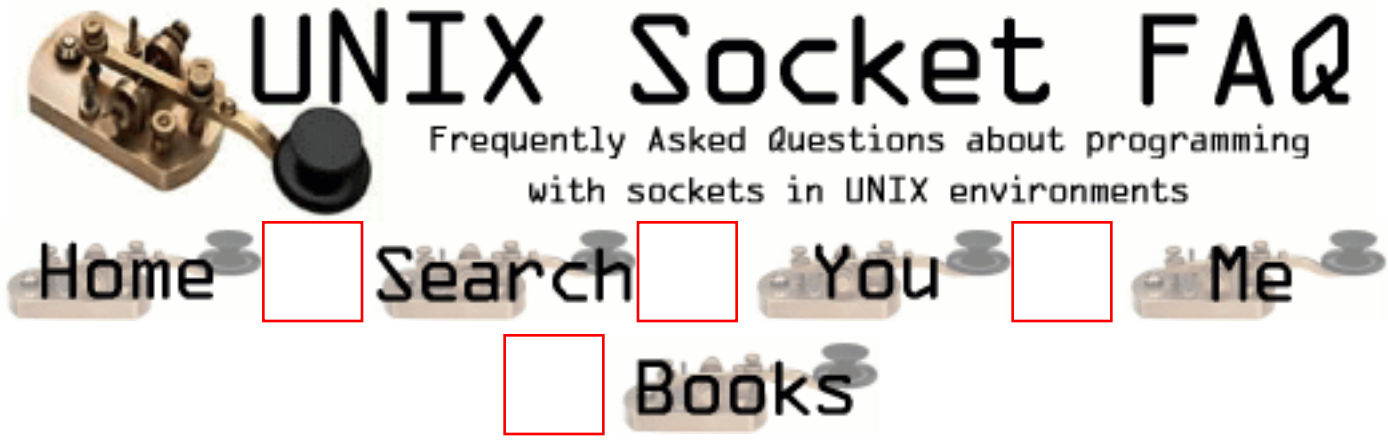
/* ... */

in.s_addr = htonl (l);

From: Alberto Fernandez

Lots of thanks Rob for your comments.
They have been very helpful.

Add a Comment

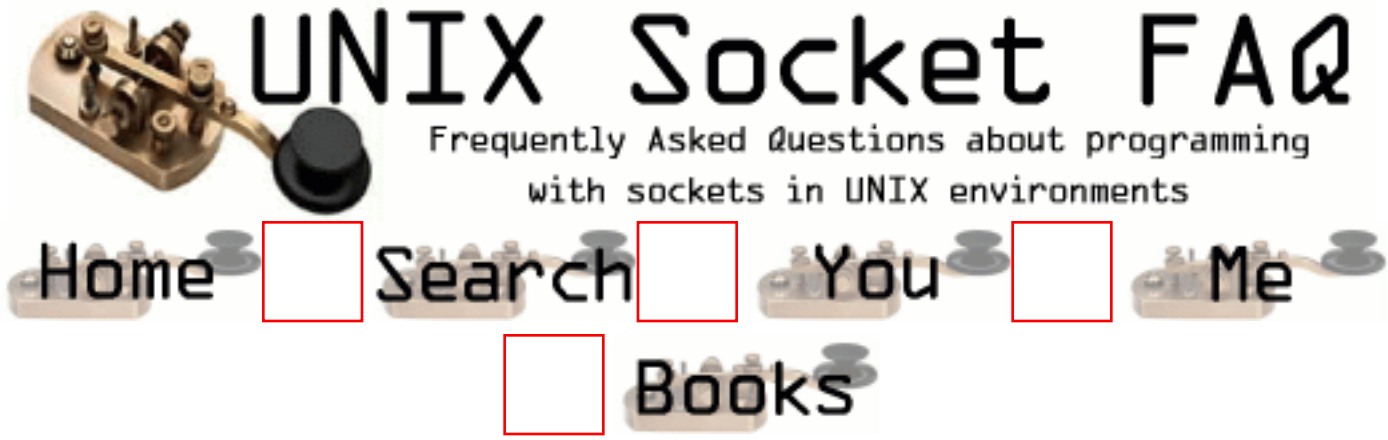


New Questions: SCO Socket

From: bakterius

how can I use SCO socket with the option SO_LINGER

Add a Comment



New Questions: How socket calls are get mapped to network device driver functions

From: chandraiah p

I want to know how the different socket calls are mapped under datalink layer network device driver functions.

Please explain me how the network device driver functions are get called by any application.

Add a Comment

New Questions: using C program to get connect to the web server

From: jagan

sir
i want the source code that connect to the internet via http request Or url and to get the html page source code from the internet
please help me to do the needful.....

if i give the input : http://www.yahoo.com

it should get the corresponding html page output

From: talles

I'd like too

From: Marios

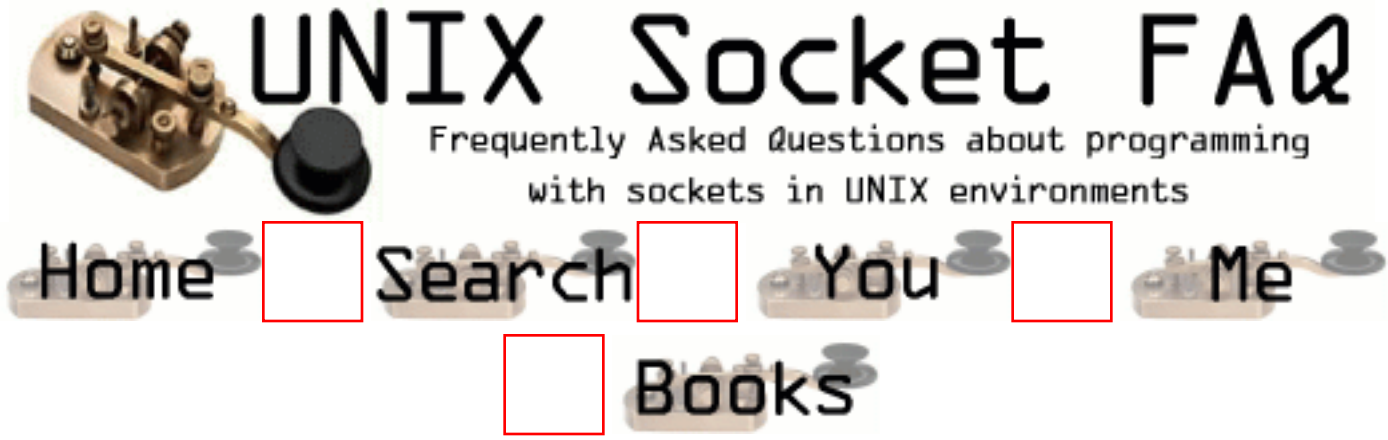
Here is a piece of code in Perl !!
But it's very easy to do that in C/C++

```
#############
# open_TCP #
#############
#
# Given ($file_handle, $dest, $port) return 1 if successful, undef when
# unsuccessful.
#
# Input: $fileHandle is the name of the filehandle to use
# $dest is the name of the destination computer,
# either IP address or hostname
```

```
# $port is the port number
#
# Output: successful network connection in file handle
#

use Socket;

$host="www.yahoo.com";
$port=80;

print "Connection Features: $host, port $port\n";

if (open_TCP(F, $host, $port) == undef) {
       print "Error connecting to server\n";
       exit(-1);
     }
else {
# This is the message that the client sends to the server !
     print F "GET / HTTP/1.0\n\n";
# But, this is the server's response !
print $_ while (<F>);
}

close (F);

##############################################################################
# Client's side tcp connection
##############################################################################
sub open_TCP
{
# get parameters
 my ($FS, $dest, $port) = @_;
# print "FS=$FS\n";

 my $proto = getprotobyname('tcp');
 socket($FS, PF_INET, SOCK_STREAM, $proto);

 my $sin = sockaddr_in($port,inet_aton($dest));
 connect($FS,$sin) || return undef;

# print "FS=$FS, Protocol=$proto, Sin=$sin\n";
 my $old_fh = select($FS);
 $| = 1; # don't buffer output
 select($old_fh);
 1;
}
```

1;

Add a Comment

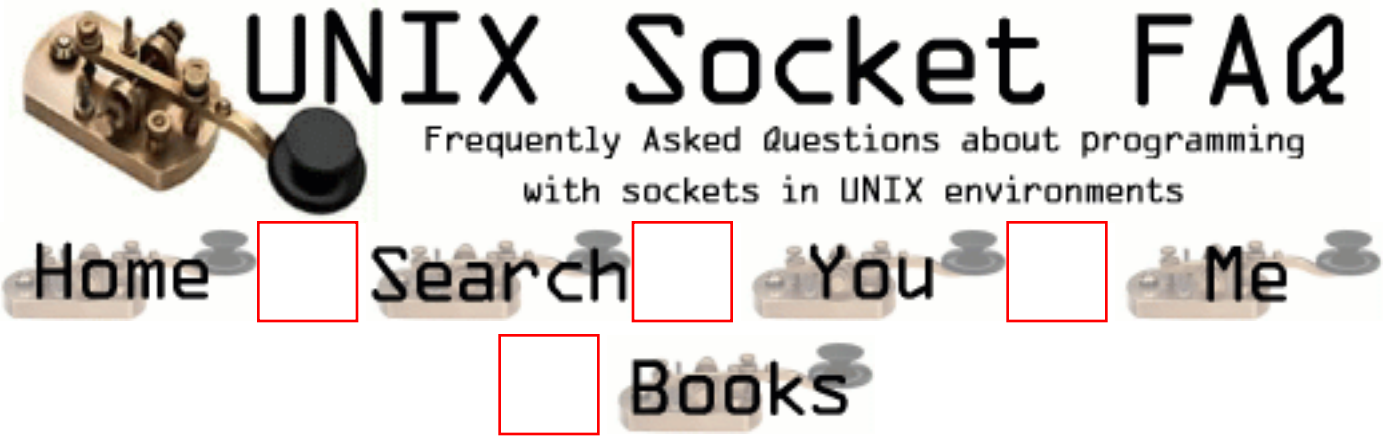


New Questions: urgently required!!!!!!!1

From: SENTHIL KUMAR

sir,
  i want to connect to internet via the proxy using c-codings and have to fetch the html source code for any type of internet page.
   pls send me c-coding as soon as possible...
senthil...

Add a Comment



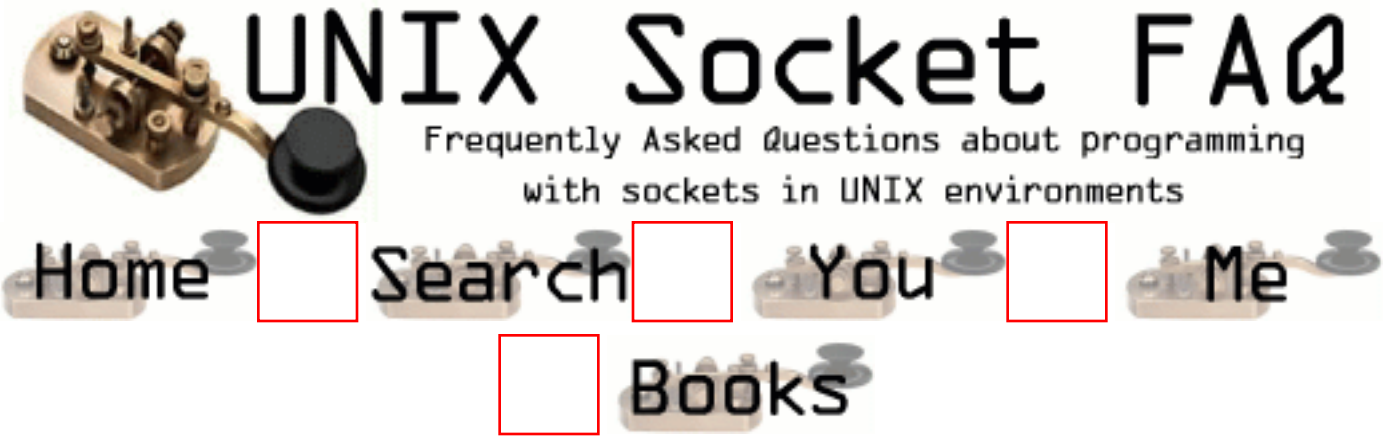
New Questions: UDP Buffers

From: Jason

I am writing a small UDP application that is NRT (near real time.) It has to always have the most recent status down to the millisecond, so I'm not worried about a lost message. The actual message is small (~10 bytes) I think I'm seeing some buffering, but I don't know if it's just host delay. Ideally, the most recent and complete packet will come out when I do a recvfrom. How can I ensure this. I've looked around the linux source, but not to any avail =(.

Thanks in advance.

Add a Comment



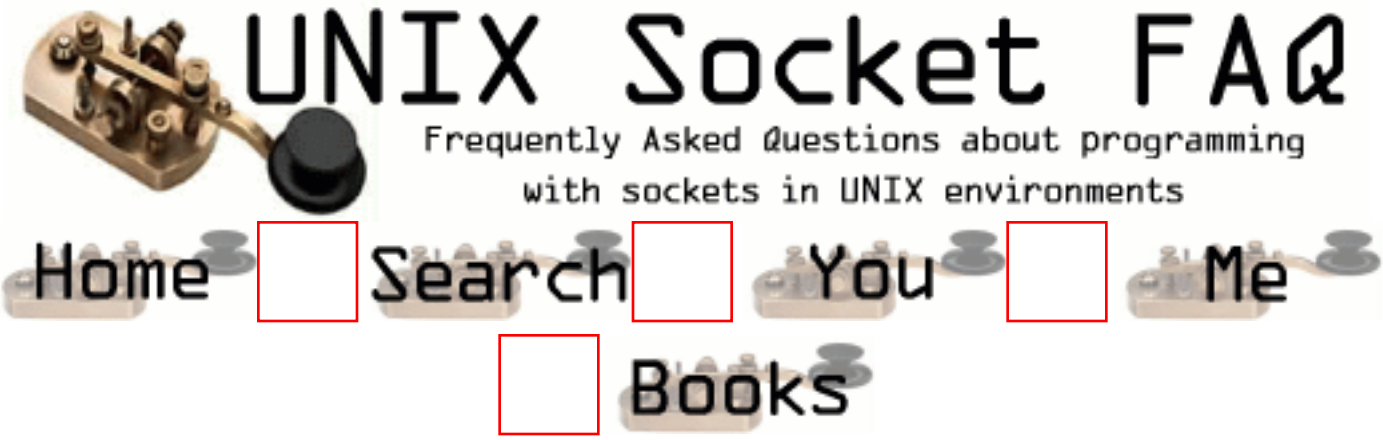
New Questions: Read and write irrespective of the order

From: bhishm

Hi ,
I made a client and a server .Both of which can recv and send ,I want to make the communication asychronous irrespective of the recv or send system calls order which are of blocking in nature .If the socket sends then the client will receive ir-respective of the order and vice versa like online chat .

Can you help me by giving some small sample example code or idea .

Add a Comment



New Questions: how to use a C client to connect to an RMI port?

From: Kaushik

hi,
I have a Client server program written in C for Linux and the program is
working when I transfer files from one directory to another
on a normal port created in C.
Now I have a PORT that is created using JAVA RMIREGISTRY
(rmiregistry 4677 &). What changes do I have to make to connect
my program to this port?

Is it possible for a client written in C to connect to an RMI port?

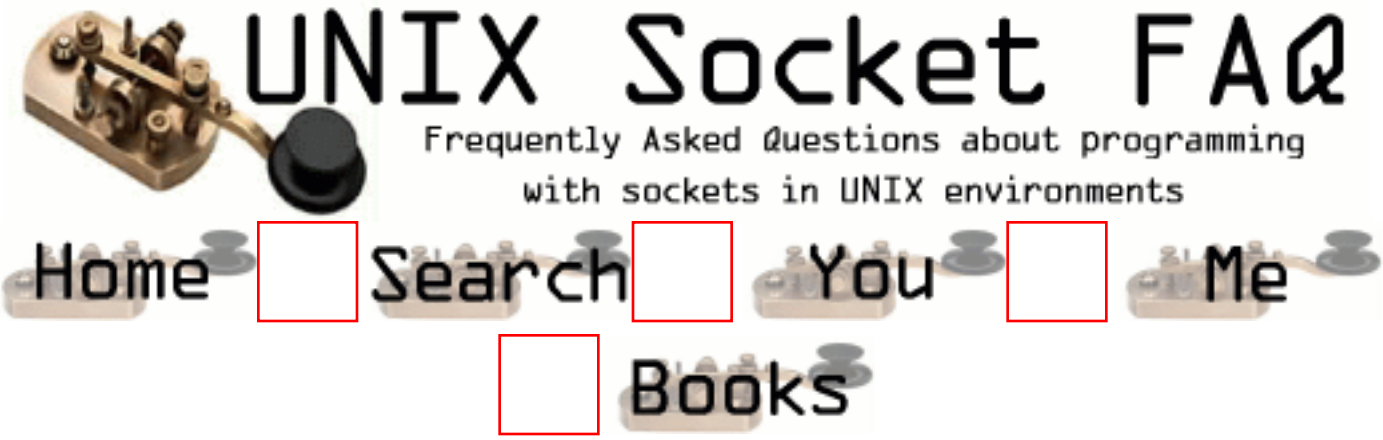
From: Hector Lasso

Hi,

It is possible. However, it is not a simple task.
The problem is that RMI uses serialization in order to pass parameters from the calling object to the remote object. You would have to code the parsing mechanism or some sort of subset in order to communicate with the remote object and make it work. I'm not saying that this is impossible, however, there are easier ways to accomplish whatever you want to do. I would check JNI (Java Native Interface). I'd use it to call the remote methods and return the values needed to my C program, and this would definitely be a lot easier.

Hope this helps.

Add a Comment



New Questions: resource temporarily unavailable - recv problem

From: aliye

I have a client program written in C++ and running on Unix platform. It has a TCP/IP connection with the server. And my socket is using non-blocking socket mode.

The problem is , not always but occacionaly, it is getting "Resource temporarily unavailable" error on recv.
When this error occurs, I repeat 3 more times to receive the data but these attemps are never successfull.

Any idea, why I would get these error? Is there any way to avoid it? What is the risk of loosing data in case of this error?

Thanks in advance

From: matt

check that your not trying yo recv from your master socket

From: aliye

Could you explain little bit more. What is the master socket. If you mean the server socket. As far as I can check it does not get any problem for sending packages at the same time.

From: matt

by master socket i mean the socket you bind and listen on.

i got resource unavailable when:

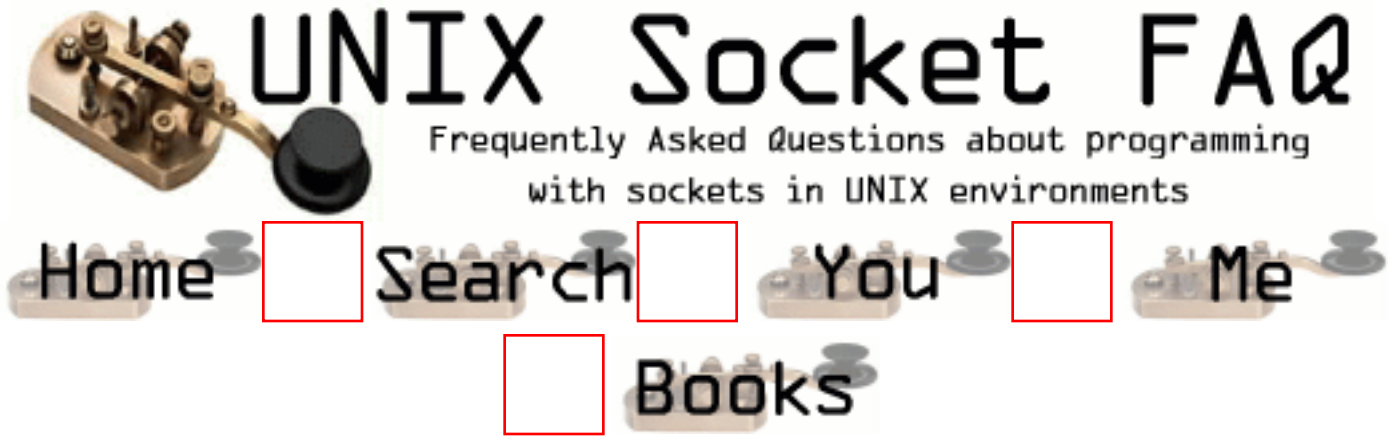
i was selecting on my socket file descriptors and my master socket was one of them.
if your doing this then when your looping thru your sockets ready for reading make sure your loop variable is not equal to your master socket.

hope thats clearer

From: Rob Seace

If I'm right, "Resource temporarily unavailable" is the
errno EAGAIN... And, you say that the socket is set to
non-blocking mode... So, EAGAIN is a perfectly legit and
expected error to receive for a non-blocking socket...
Check your local man pages... But, basically, it just
means that there's no data to receive right now, and it
WOULD normally have blocked and waited for data to arrive,
but since it's non-blocking, it fails and sets errno to
EAGAIN...

Add a Comment



New Questions: setting file permissions for sockets

From: matt

how do you set file permissions for sockets? should you fchown after you call socket or at the end after you listen()??

From: Hector Lasso

Hi,

I think it is impossible to set file permissions on sockets because sockets are not files. I may be wrong however so you don't lose anything if you make a test. I guess you would receive an ENOENT error or EBADF...

Add a Comment

New Questions: Timeout period affecting selects ability to detect data

From: Alan Harding

I have a program that is connecting up to a remote system and listening for data.

This is what I would like to do:
1. wait for the start of the data
2. read in the data until I don't receive any more (2 seconds without any data, for example)
3. Process the data
4. Go back to 1

Here is my code:

```
  unsigned char in_buf;

  struct timeval tv;
  fd_set readfds;

  tv.tv_sec = 2;
  tv.tv_usec = 0;

  /* ---------------------------------- */
  /* Initialise the file descriptor set */
  /* ---------------------------------- */
  FD_ZERO(&readfds);
  FD_SET(s, &readfds);

  while (TRUE)
  {
    if (select(s+1, &readfds, NULL, NULL, &tv) == -1)
```

```
    {
      perror("select");
      exit(1);
    }

    i = (FD_ISSET(s, &readfds));

    if (i != 0)
    {
      if ((i = recv(s, &in_buf, 1, NULL)) <= 0)
      {
        if (i == 0)
        {
          printf("\nConnection closed, errno %d\n", errno);
          exit (0);
        }
        else
        {
          printf("Socket read failed (1), errno %d\n", errno);
          exit(0);
        }
      }

    }
    else
    {
      printf("\nNo data\n");
    }
  }
```

The problem that I am running in to is that the select never sees data waiting in the buffer with tv.tv_sec = 2. If I set the value to 60 then it works fine. Right now I am just seeing a 'heartbeat' from the server I am connecting to which is sending out a message every 40 seconds. From my experiments it seems that after the first timeout select never sees data in the buffer again.

Any ideas?

thanks!

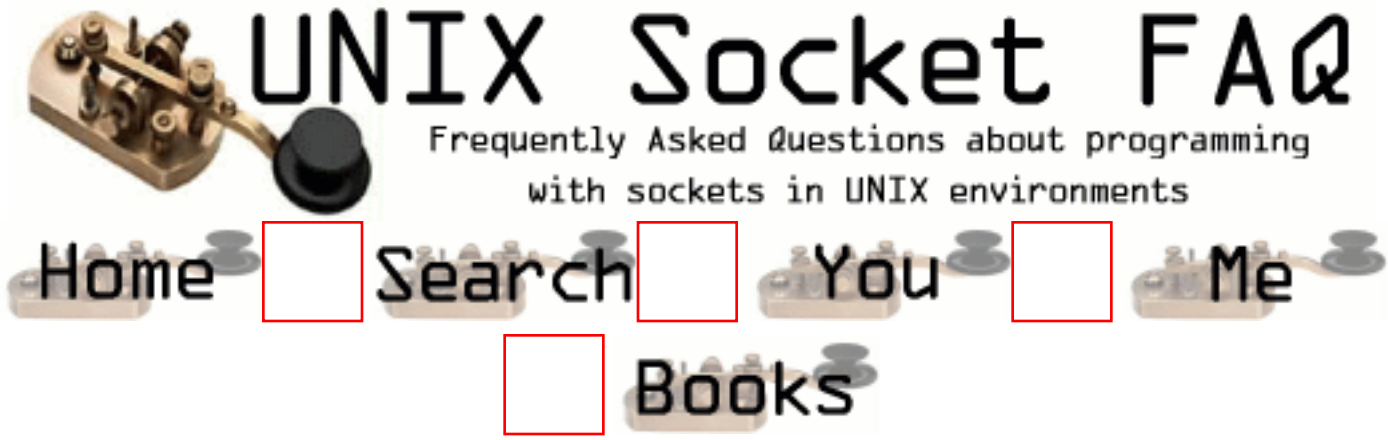
From: Alan Harding

I just fixed my onw problem, the FD_ZERO and FD_SET have to be inside the loop because select overwrites them.

From: Rob Seace

Not only do you need the fd_set initialization inside the loop, but you should ALSO move the timeval initialization in there, as well... Because, some OS's will modify the timeout value as well (setting it to the remaining time left before timeout expiration, if it returns before the timeout goes off)... Some will not... But, you can't rely on one behavior or the other, so you're best off just explicitly setting it to however long you really want to wait, each and every time before you call select()... (Unless you know for certain that your OS does modify the timeval, and you're in fact counting on that behavior...)

Add a Comment



New Questions: Making sockets working over the internet

From: David

I have developed a chat application in VB usinf winsock.
Can any one give me an idea how to make this chat application work over the internet??

From: Loco

First of all, you will need a machine that has a permanent valid internet IP address, or maybe an invalid address behind a NAT device that can map it to a valid address.

Second, you will need to distribute your chat client application (or do you think you can chat to yourself for more than 12 seconds???, believe me i've tried it many times and it's really boring)

Then you'd probably want to see it work, so you'll have to set up the chat server in your server (the one that has the valid internet address and a button that is normally used to power it up/down)

Oops i forgot: Modify your client program to use this address (the valid one) and then call the people you gave your program to and tell them about the new enhancement to your software, send them the new program and bill them for the new system (i learned this at M*cro$xxx).

If they have problems installing the application bill them again and send your ten-year-old brother to do the dirty work of copying another bunch of DLLs that the application needs to run properly...

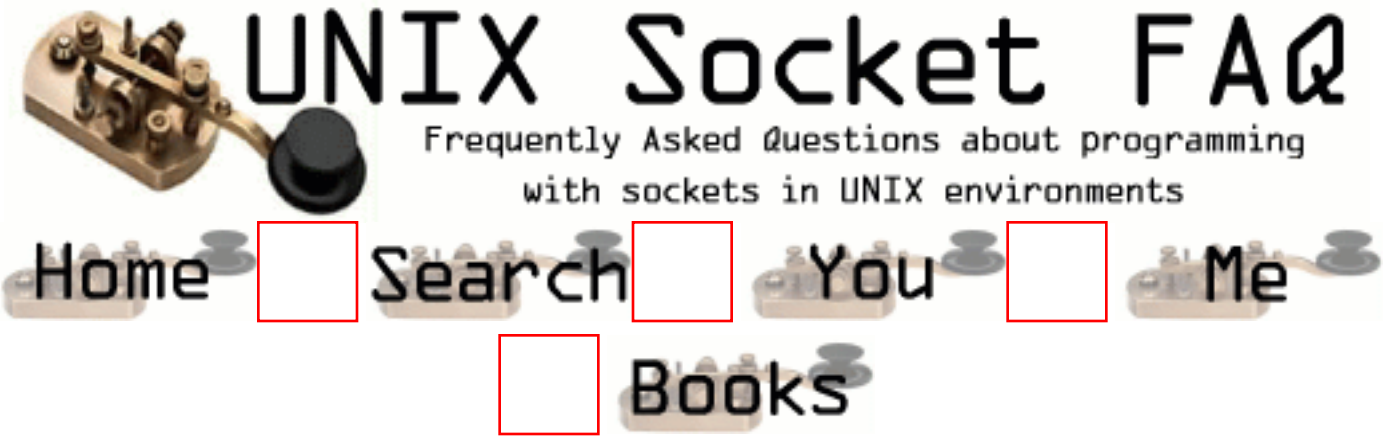
If there is someone left that wishes to use your chat then let him use it for a while. Add some stupid features (that no one uses) as well as some really useful features, but make them nonconfigurable. As soon as the users are using the chat regularly charge them for every message or something... develop a new version that includes some configuration windows for the useful stuff and some more stupid features no one uses. Name it the professional version.

Sell it for 3 or 4 times the price of the original one.

If you get rich please don't forget the royalties for the intelectual property that you should pay me

:D (HAL)

Add a Comment



New Questions: EINTR Testing

From: Jeff Halbgewachs

I encountered the intermittent EINTR problem as mentioned
in the FAQ, and added a while routine to handle it (couldn't
find that source code anywhere, grrr!) Anyway, I am looking
for some way to force the EINTR condition (via ping, ftp,
whatever) so that I can test my fix. Any suggestions?

From: Hector Lasso

Hi,

you can "kill -signal process_id" to force a signal...
Remember to catch the signal you are going to send to your process in your code. I would use a handler for SIGALRM and kill -ALRM procid, or even a SIGHUP

From: Jeff Halbgewachs

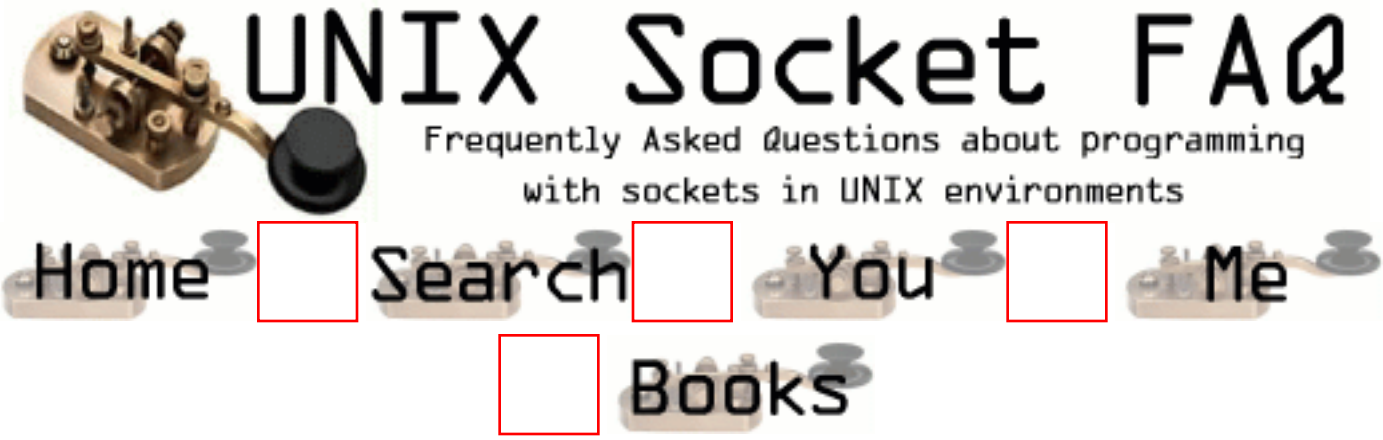
Thanks for the advice, Hector. Killing the process seems a
little drastic, though. I'm not experienced with socket
programming, so maybe I'm phrasing the question poorly.
This problem happens maybe once per 1000 messages, and
always on a certain type of message. We write() a header
and then a buffer from the client to the server, then read()
a reply byte from the server -- this last one is what gets
the EINTR error. I'm looking for either a UNIX command or a
small program that would tie up whatever (the socket?) long
enough to cause the error, for testing purposes only, but I
do need the message to be read eventually.

From: Jeff Halbgewachs

OK, nevermind. I friend helped me incorporate Hector's

advice and it works like a charm. Thanks again!

Add a Comment



New Questions: Where can I get the converter to convert Windows to Unix?

From: Thomas Lee

Where can I get the converter to convert Windows to Unix? If anyone knows please e-mail to me and let me know.

-thank you

From: Hector Lasso

What do you mean?

Do you need to convert from one byte-ordering scheme to the other? Or do you want to use Windows software on a Unix box?

From: Loco

Hey, you are asking for a miracle.

I would read about ntohl(), ntohs(), htonl() and htons() for the simple ones...

Call (974) 555-**** and ask for Bill. Tell him about your problem, maybe he could help you with this. Tell him that you need him to convert Windows to Unix because it Winsocks...

:D (HAL)

From: arjun

hey what the fuck r u askin got to fuck u now come and see me

---

From: sudmah

hey !! I think U probably lost the concepts of basics of

computers! better refer to the website , www.microsoft.com.
There U`ll get the real basics and converters!

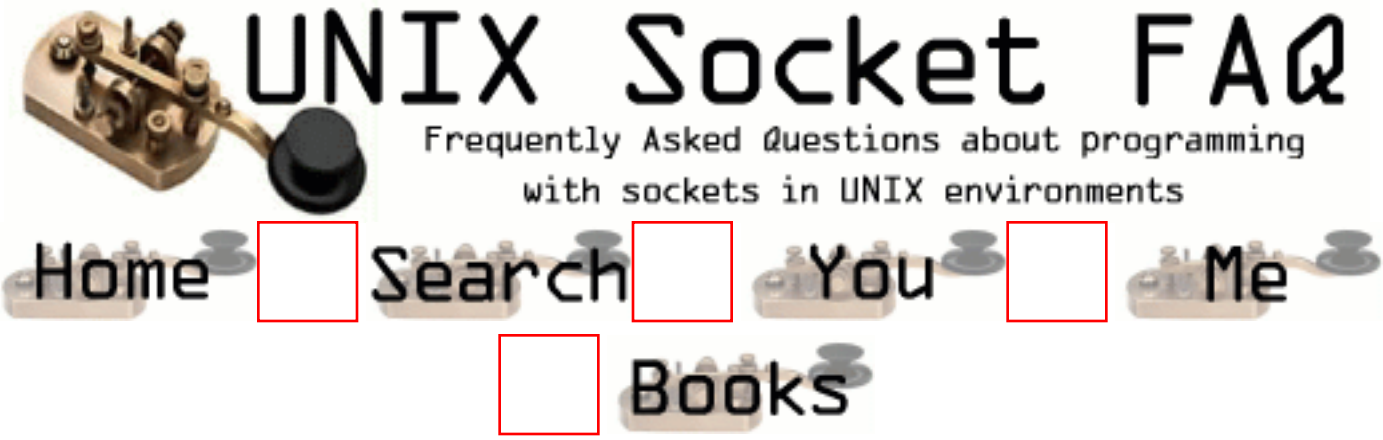
From: Arik Baratz

Contrary to common belief, ther eis a very easy way to convert Windows to UNIX.
All you have to do is take your Windows(tm) distribution disk, and a felt tip marker. Then simply write UNIX on the disk.
Once you have done this, you have neutralized the Windows socket code on the disk. Next you have to magnetically retrace the disk to make it UNIX compatible. This can be done by standing on the edge of large buildings and waving the disk back and fro, as far away from the building as you can.
Do try to get as far from the building as you can by reaching away as far as you can with your hand and preferably your entire body. You must perform this procedure with care that the edge is directly above the street; That the building is at least 20 story high; That there is nothing obstructing on the way down.
Then just put the CD in your computer and boot.

From: Loco

I thought i was "loco" but this guy up here is really nuts.

Add a Comment

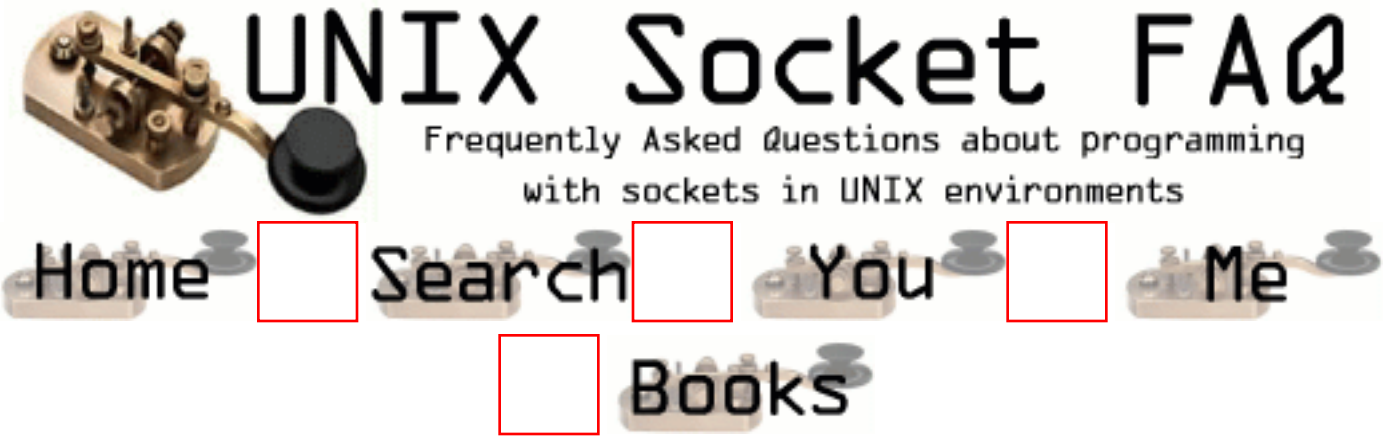


New Questions: TCP/UDP on same server

From: Anurag Goel

I need to implement a date/time/echo server that can fulfill both TCP/UDP requests. How do I do that?

Add a Comment



New Questions: HOW CAN I open n connections working at the same time ??

From: overflash

i need to open n connections to an host (with an array of sockets)
and this n connections must work at the same time (in send() and recv())
how i can do it ?

From: Loco

Create a new thread for each socket.
Fork a new process for each socket.
Use select().
My favorite is: Create "m" processes such that "n mod m = 0 and 1 < m < n"; for each process create "x" threads (b * x = n / m, and b >= 1) in each thread handle "b" sockets using select()
You must be really careful that "n" is not prime or else your brain will crash. :(

Just kidding. You could do a lot of things, but the basic idea is: You can select(), fork(), or pthread_xx()
Or a combination of any of them. Use your imagination.

:D (HAL)

Add a Comment

New Questions: non blocking socket

From: Mauro

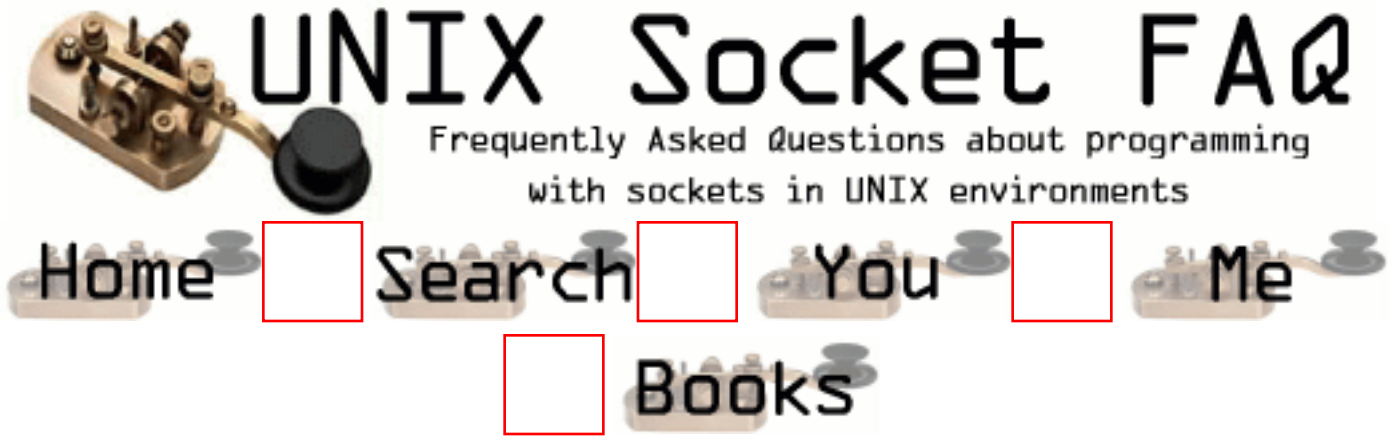
I would like to do a port scanner using non-blocking socket, but in this case the connect() function always return -1. Someone could telle me how I can see (looking errno) if a port is open, close or firewalled? Thanks.

From: Rob Seace

Yes, connect() would always return -1 with a non-blocking socket (unless it could somehow complete the connection instantly); that's the whole point of using a non-blocking socket: so you aren't tied up in the connect() call, while it's completing the connection...

After it returns -1, with errno set to EINPROGRESS, you can select() for writability... When the socket goes writable, then either the connection is complete, or it failed... You then have to use getsockopt() to get SO_ERROR, to determine which: if it's 0, the connection succeeded; if not, it failed...

Add a Comment



New Questions: RST packet ???????

From: Marios Karagiannopoulos

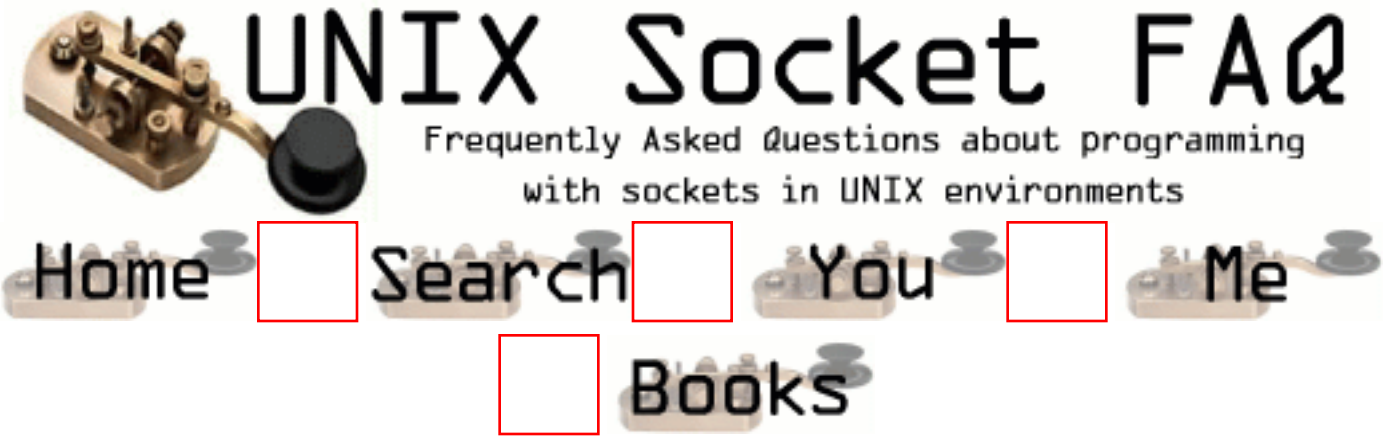
Hi. Where could I find the reasons of sending a RST flag(reset flag in a tcp packet) in a packet? I know that a reason could be the non existing port number, but I would like to get all the reasons for this packet !!

Thanks a lot all of you !
/BR, MK

From: Andrew McMurry

Sometimes connections can get stuck in the FIN_WAIT state and it would be really nice to ABORT rather than wait the full 10 minutes + 75 seconds to timeout.

Add a Comment



New Questions: Source Code client.c And Server.c for UNIX

From: Lee

Anyone have it..
I've tried so many on the web but I can't compile any of it...
Compile: g++ -o filename filename.c -lsocket -lnsl
Anyone got any source code than can be compiled using this method...
Or did I use the wrong compilation command...
Or do any of you have a site that contains it that can be compiled using the above commands..
Thanx...

From: parind

i have server.c and client.c and that u can run with the same command (client will give command, server will execute it and will give the execution result to client)
using,
gcc filename -o filename -lsocket -lnsl

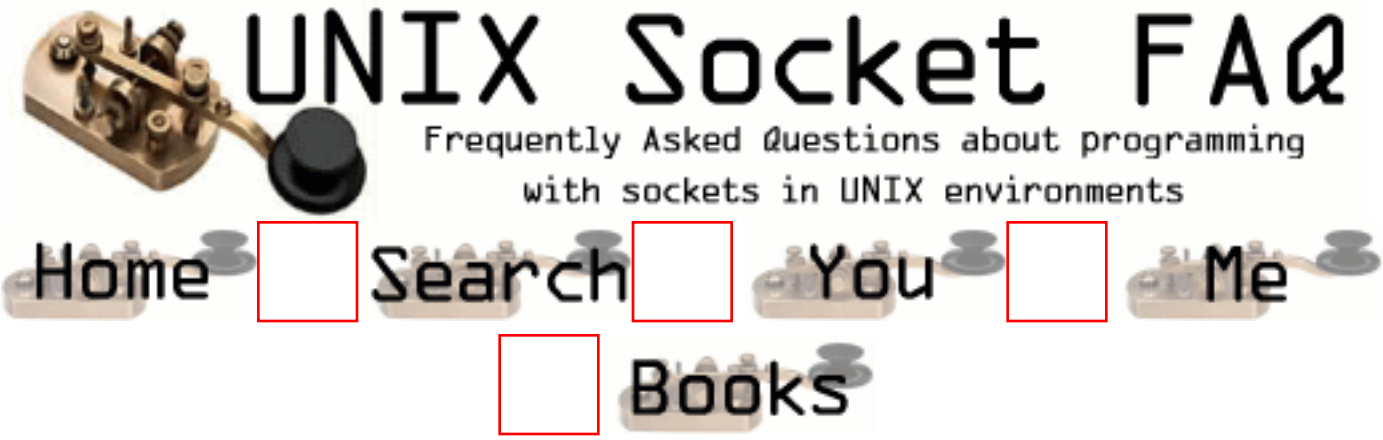
From: samc

how can I obtain a copy of the code?

From: Rafael Hofi

How can I obtain a copy of the code too?

Add a Comment



New Questions: how to check if a socket is connected or not?

From: David

Hey all,

I was wondering if there is anyway to check if a client socket is connected to a remote address. As an example, say that create my socket and after successfully connecting to some server on the Internet, i call sleep(10), and close my dial-up connection. Right after, i try writing to the socket and it still succeeds even though the connection is down.
Is there anyway to check if connected at any given time?
Thanks in advance,

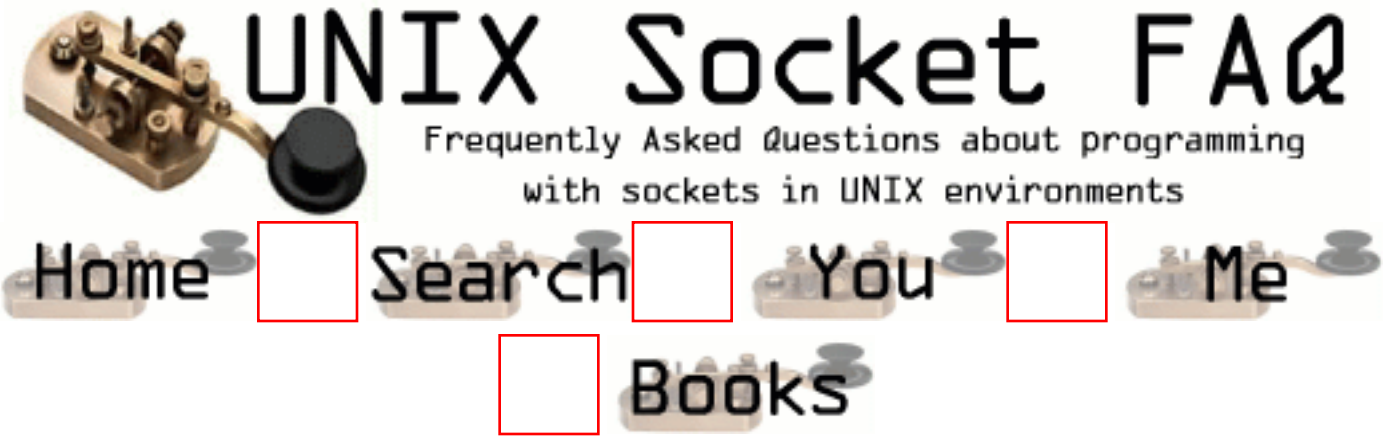
David

From: Hector Lasso

I don't know if this can help. I've never tried it but i think that using setsockopt() with SO_KEEPALIVE on your socket can help you.

Let me know if it works.

Add a Comment



New Questions: NBT packets???

From: Marios K.

Does anyone know when a Netbios packet is sent over a tcp/ip network?

thanx

From: Hector Lasso

Hi,

Could you please explain yourself.
I don't understand what you mean by this question.
However, if you have Windows NT and are using NBT (NetBIOS over TCP/IP) i suggest you use the packet analyzer that Windows NT includes. It shows the packets flowing from and to your NT box, lets you use filtering, and shows the packet from a "highest protocol perception" as well as let you view the raw contents of the packet and the data of lower level protocols involved.
For example: If you capture a packet part of a NetBIOS session it will show you the NetBIOS-related information as well as the TCP information, IP information, even ethernet info.

Hope this helps

From: Marios K.

Thanx a lot for the info, I have already captured the tcp packets and I have seen this NBT packet a lot of times. I would like to know what is the reason that this packet is sent. When a NT machine is supposed to sent this packet?

Thanx

From: Hector Lasso

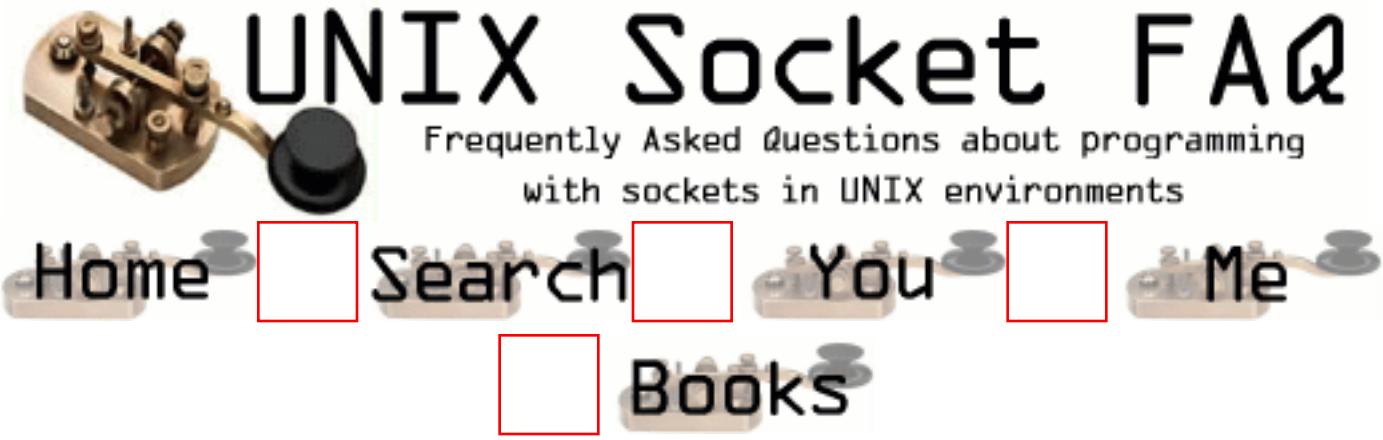
Hi,

Okay, i am beginning to understand your question.
NetBIOS is an interface used to access network resources.
I understand Microsoft implemented it over NetBEUI at the beginning, however, as NetBEUI is a very expensive protocol (it uses broadcast packets) they implemented it over TCP/IP using TCP connections from host to host.

NT can use both protocols NetBEUI and TCP/IP to exchange NetBIOS info, however which one it uses depends mainly on your bindings configuration and the protocol supported by the other party. First check the bindings order in the Network Configuration Dialog Box. If you want to disable NBT you should delete NetBIOS from the bindings only in the TCP/IP section. You can also change the binding order to force the machine to use NetBEUI first and TCP/IP if NetBEUI is not supported by the other party.

Hope this helps

Add a Comment



New Questions: TCP UDP on same server

From: andy

HI

Apologies if this has been asked before (I have checked FAQ but couldn't find any relevant).

Is it possible in C to write a server application that accepts both
UDP and TCP ? The UDP would be used by the client to broadcast
to each server on a network to check availability etc.

If it is possible can someone direct me to a info source where I can
read up on this.

If it is not possible, I take it two servers can listen using the same port ?

Thanks in advance

Andy

From: Rob Seace

Sure, of course it's possible... You'd just need to create
two separate sockets, one for TCP communication and one for
UDP communication... You'd do a normal listen() on your
TCP socket, and nothing special to the UDP one... Then,
you'd likely just select() on both of them, until one or the
other becomes readable... Then, for the TCP socket, you'd
do a normal accept(), and deal with the connection however,
while for the UDP socket, there's no "connection" involved,
so readability just means you've got an incoming single
message to deal with however, so you'd just recvfrom() it...

Add a Comment

New Questions: i want to implement a natwork game, i want to discover all hosts on a subnet that are participating in it, what shall i do?

From: malti abdallah

From: ggbutcher

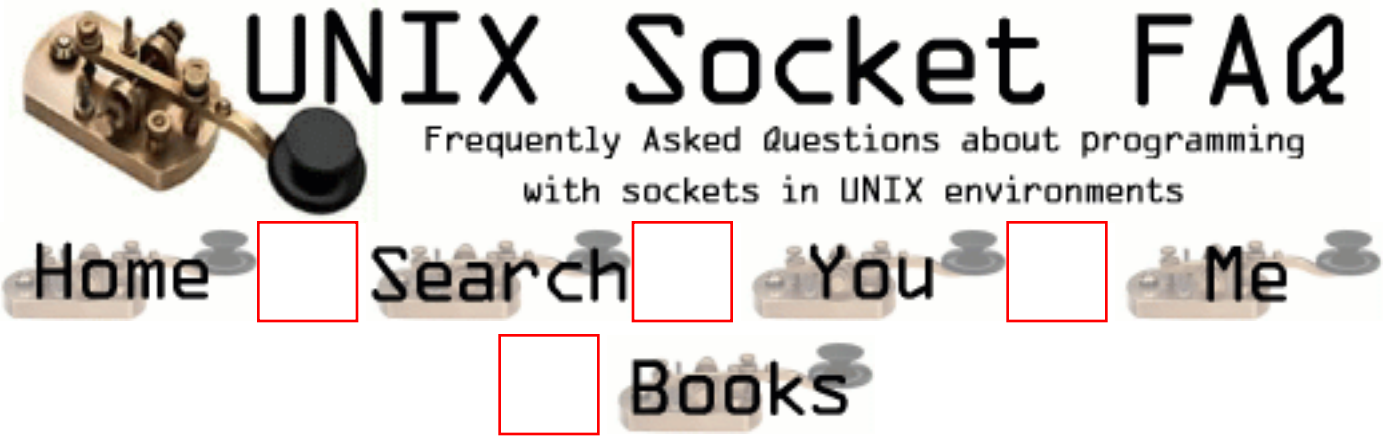
nmap.

From: Hector Lasso

Every computer that participates (or wishes to participate) can bind an UDP socket to a especific port.
Your server sends a broadcast packet to the subnet's broadcast address, and port mentioned above. All the computers will receive the packet and would send a reply to this packet informing they are available (or not, maybe they are already playing something else) and whatever information you wish to send (player's name for example)

That is it.

Add a Comment



New Questions: Using select with recv send

From: Bhishm

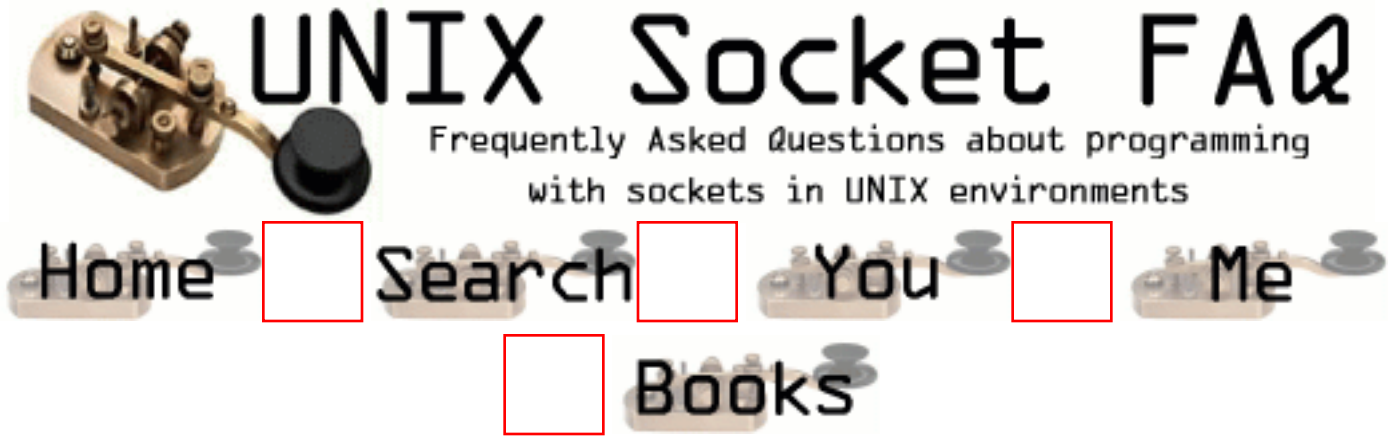
HI ,

I want to make the send-recv system calls asynchronous like yahoo chat or any other chatting service .

If I make a client and server both interacting with each other, than the call to either send or receive blocked until it receive/send a message .

I am getting difficulty in how can I use select or poll
to make it non-blocking instead of blocking

Add a Comment



New Questions: communication between a java program and a c program

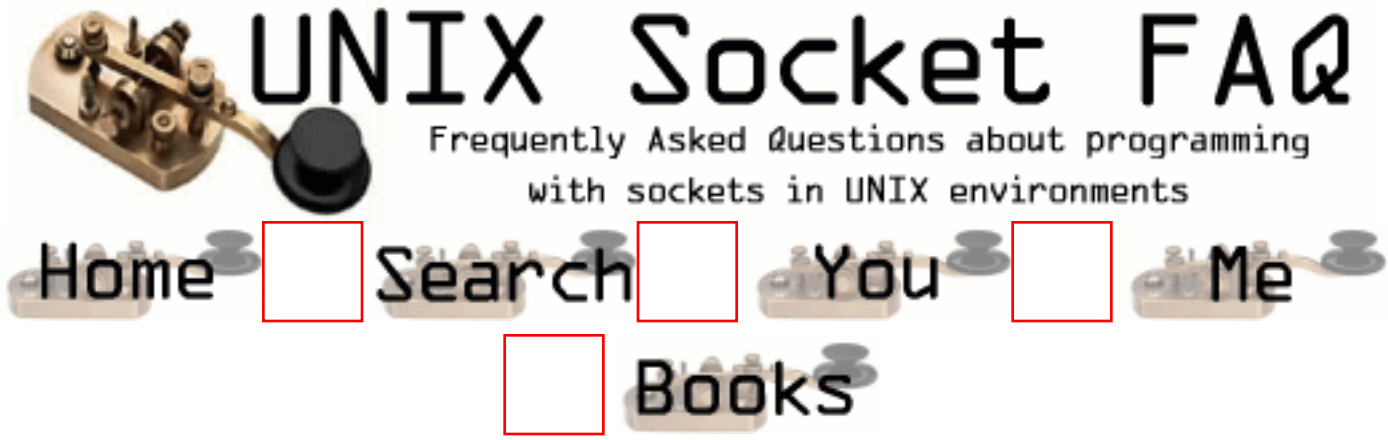
From: ait ettaleb

how can i make communication throw sockets (using UDP protocol) between a program written in c and an other written in java ?

From: Kennedy

Have you looked at the JNI? You use this so you can write
native methods in a program like C, and call them from your
java program.

Add a Comment



New Questions: Getting Signal Number 2147482380 While sending data on socket

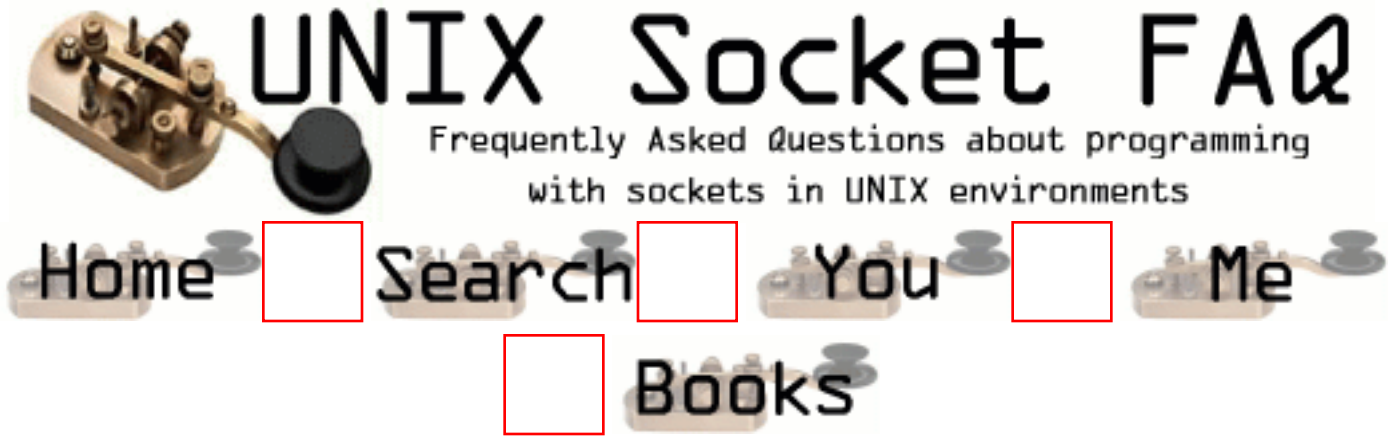
From: Vignesh Bhatta

Hi
I am facing problem , in my server and client program.
My program is getting signal number 2147482380 once send on client socket was performed.
I am suspecting some where stack corruption. It is occuring only when there is heavy load on the
socket. What i mean is if program is transmitting data on socket at the rate of avg 100msec.
Could I get the answer please from any body to email. Machine is Pentius III. OS is Sco 5.0.4

Thanking you

Vignesh

Add a Comment



New Questions: Time Polling

From: Jacob Noffke

I am writing a typical client/server socket program that continuosly sends messages for twenty seconds. These messages are to be sent at random times. How can I time the generation of messages?

From: ashok

u havent told in what state u exactly r .. but i can think of three of them ..
1) u dont have to watch any sockets during the period (meaning u r not using select)
-- in such a case u can put ur 'write' call in a loop with the dealy u need .. using either 'sleep()' or 'usleep()' depending on the duration of the interval u need between each 'write'
the pseudo code could be ..

```
gettimeofday( &before, NULL )
loop{
  generate ramdom 'x';
  sleep(x); or usleep(x);
  write(...);
  gettiemofday( &after, NULL );
  } until ( after - below > 20 seconds of whatever);
```

2) u need to watch sockets for activity and duration u need between intervals is less than a second(u r using select)
-- then u can use the last argument of select for setting the duration that u need between the intervals .. and u need to do some coding to take care when select returns a value which is greater than 0 .. becoz this means that the interval that u wanted to wait for has not completely elapsed and u need to subtract the elapsed time from the duration u r going to wait next ..
the pseudo code could be

```
bool duration_complete = false;
struct timeval delta,timeout,before,after;
```

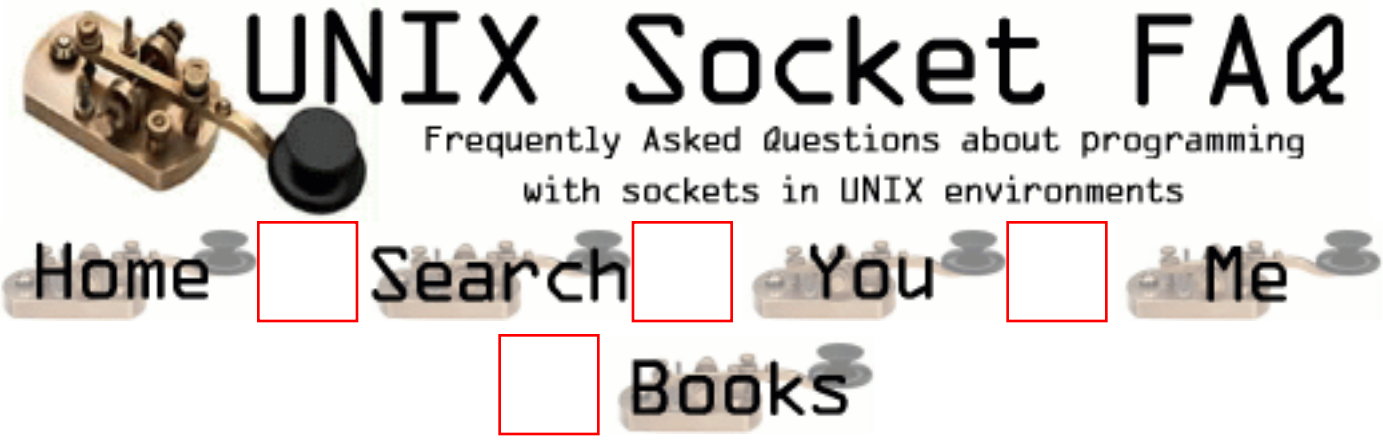
```
loop{
  if( duration_comlete = false )
  {
    generate ramdom 'x'
    timeout = x;
  }
  else
    timeout = x - delta;
  gettimeofday( &before, NULL);
  nready = select( maxfd, str_rset, str_wset, str_eset, timeout);
  gettiemofday( &after, NULL);
  if ( nready > 0 )
  {
    delta = after - before;
    handle the active sockets or fds;
  }
  else if ( nready == 0 )
  {
   // this is what we want ..
   // the duration has elapsed
   delta = 0;
   write( ...);
  }
} //loop end
```

3)u need to watch sockets for activity and the duration u need between each 'write' is in seconds (u r using select)
-- this time u can use the 'alarm()' ..althought select will work ... but this will make ur job simpler ... but dont forget to put this peice of code near select ..
pseudo code would be ...

```
  signal( SIGALARM, alarmhandler );
  gettimeofday( &before , NULL );
  loop{
    signal( SIGALARM, alarmhandler); // this is safe programming becoz some os's need that u reset the handler each time
    if( errno != EINTR )
    generate random 'x'
    alarm(x);
    nready = select(...);
    // very important
    if( errno == EINTR )
      continue;
```

```
if( nready > 0 )
   handle the sockets or fds;

gettimeofday( &after, NULL);
} until( after - before > 20 or whatever);
```

Add a Comment



New Questions: UDP BraodCast

From: Paulson

Hi,
How to Broadcast the data using UDP ? I am now opening a UDP Port(127.0.0.1) and sending the data to it. I want this to to read by all clients connected to the server through TCP.
Now When I am trying to recv the data , Only the first client is getting the data. Each clients (created uisng Fork) are looking in to this same UDP port ?
How can set the UDP port to recieve data for all the clients instead of one client ??
Thanks
Paulson

From: Loco

Hey man, you are doing it all wrong:

-127.0.0.1 is your own addres, in fact it is the localhost address, a message sent to this address will never leave your host (like some relatives when visiting you, they stay for months, did i ever tell you bout the time my grandmother...? sorry, focus man)...
- You should use 255.255.255.255 for all addresses in your local net (i guess some routers do forward this packages to all of their interfaces, but i'm not sure, so let's assume they get only to the LAN ur host is connected to)...
- If you need to send to all machines in LAN A (addresses 192.168.0/24 for example) and you are on LAN B (addresses 10.1.2/24 for example) then use the destination broadcast address which would be 192.168.0.255 (just OR the net address with NOT netmask and voilà, there you go!!!!!!)
- Aha, you tell me it ain't working... it must be some problem in your satellite dish that is messing with the infrared waves your wireless card sends, and... Are you sure you allowed your UDP socket to use broadcast addresses as destination??? NO? I knew it...
If you don't allow the socket to use broadcast addresses, how do you expect it to work?
Well, it is a very long and expensive process, and if you send me $1000USD i will tell you all about it. Please don't try to do it by yourself because it is very dangerous (at least for my bank account). Being the nice person i've always been i will give you a clue:

int opt;
setsockopt(yoursocket, SOL_SOCKET, SO_BROADCAST, (void*)&opt, sizeof(int));

Sh*t, i wrote it all... can't find the undo key.. f*ck... Don't forget to send me the $1K...

Remember everybody, my email address has changed (i don't know who is loco@yahoo.com, but this guy must be receiving many strong emails from people of this FAQ!!! I hope he don't get my money, mon)

:D (HAL)

From: Hector Lasso

Hello,

This is what i understand of what you wrote:
1) You have one server, many clients
2) You wish to broadcast information to all your clients using UDP sockets
3) Your clients are already connected to the server using TCP sockets
4) You are currently creating a UDP socket, send the information to a broadcast address (which one?) and only one of the clients is receiving it.

My questions are:
- Do your clients run in different machines? (remember that only one process can bind a socket to an (address,port,protocol). If u create the UDP socket and then fork() children then the socket can be used by all of them, however only one will receive a message unless he uses the MSG_PEEK flag in recv()
- What is the destination address you are using? Does it match to your network's broadcast address?
- Could you post a simplified version of your code?

From: Paulson

Dear Loco,

You are totally wrong ... I am sending to the local address only. The destination is the server itself and I am using 127.0.0.1. Probably I will explain in detail .....

I am currently involved in the development of an internet based trading system, and is using a data feed from exchange. The program is working perfectly for a single connection.

This program works like this ...

One process recieves the data from the stock exchange and will put into the shared memory.

When ever there is a change in the data it will send the data to the one UDP port. The destination is the local server itself.

Another process keeps on waiting for a new client(TCP connection)and when there is one it will fork for a new child. Each child is waiting for data from the the UDP port above. But now only one client is getting the data and the port is cleared after that.

I have tested the MSG_PEEK flag in recv() but it is going into the infinite loop. I think the new data which is coming into the UDP port is waiting in the queue becase the exisitng data is not cleared from the port. Is there any way we can force the data to write even if new data is existing? Is there any other solution ?

In summary ...Any solution for one send()...multiple recv()

Thanks a lot
Paulson

From: Loco

Hey man,

Now i get it... Wow, it's been the first time ever in my whole life that i've got it all wrong. Well maybe it has happened before.. what the hell, it happens all time but who cares...

OK, what my friend up there told you is very misleading. If you use MSG_PEEK, every process will get the message (because the message is still in the queue) but which one should get the message out of there??? That is the problem.
I suggest you use other methods to broadcast the information to your client processes such as pipes:
The forker (the one that listens and forks for each connection - you were thinking different weren't you) can be the receiver of the UDP packets. Whenever it accepts a new connection creates a pipe that can be used to communicate with the child that forks.
It keeps a list of the pipes, and whenever a new UDP packet is received, this process writes to each and every pipe it has in its list. Obviously it has to capture the SIGCHLD signal, and close those pipes related to dead children...

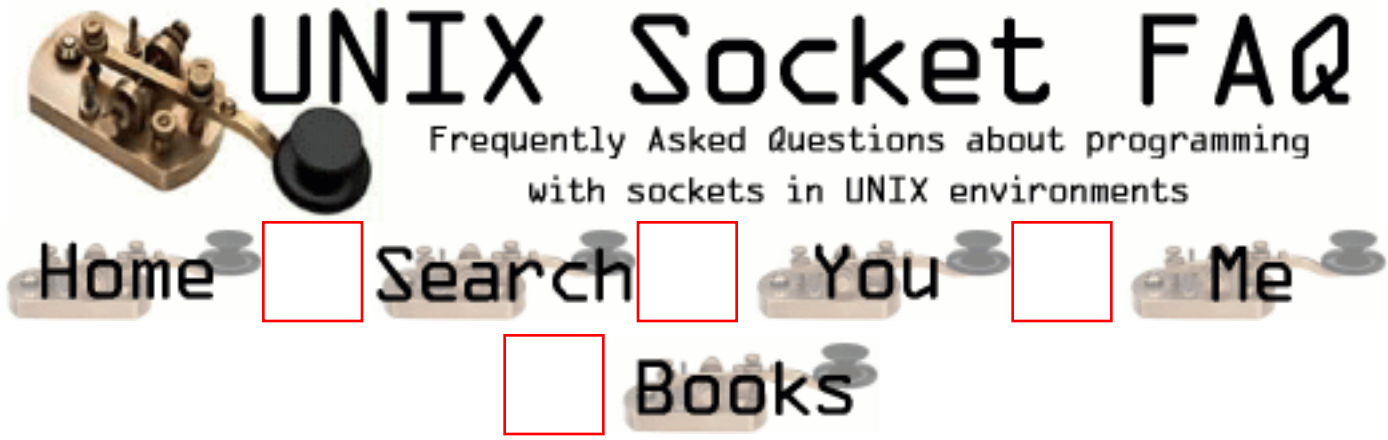
Pretty easy, ain't it?

I guess there must be some way to do this with UNIX sockets, but i've never worked with'em. If anyone knows about it please let me know.

Well, time to go... My jailer came back and he's gonna take my cigarettes away.

:D (HAL)

Add a Comment



New Questions: Recv ( ) in blocking mode

From: rupashri

Hi ,
I have written a program using TCP socket communication.
Once the Client connects to the Server program, i want the connection to be kept alive until the client sends a request to tear down the connection.
I want my recv() function to work in blocking mode.
How do i do it?

The code is something like this

```
main()
{
  socket();
  bind();
  listen()

  for(;;)
  {
    newsockfd=accept(...);
    while(1)
    {
      recv(newsockfd,buffer,len,0);
       .
       .
       if(some condition)
         break;

     }
}
```

From: Loco

I think it would work.
However, i don't understand what you are talking about...

What do you want to do? Do you want your server to work like those cheap restaurants where there is only one waiter and he is always waiting on somebody else?

I guess you don't.

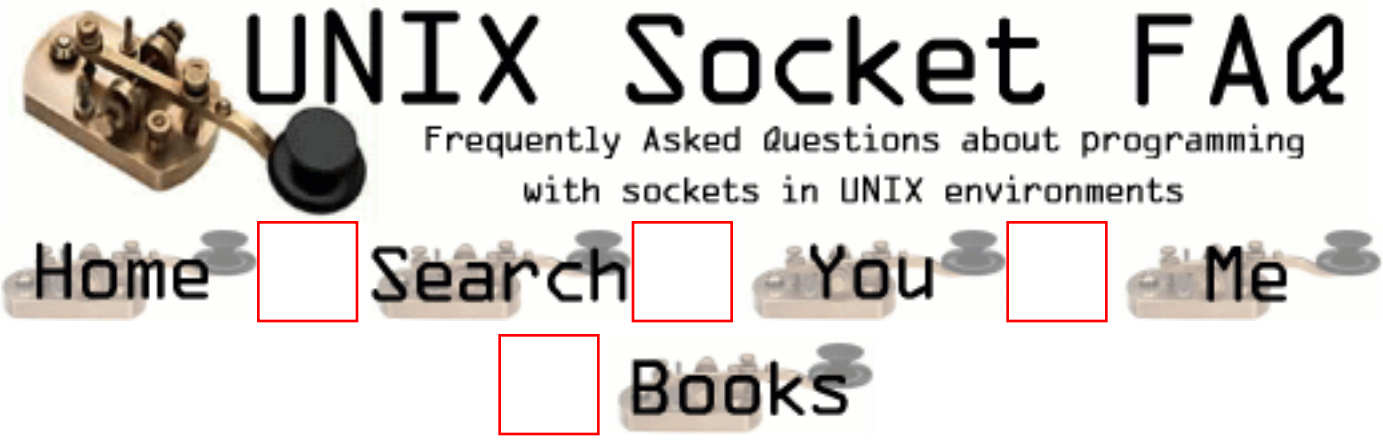
So your code will probably work, but when your process receives a new connection it will just go on to serve this client and the others will have to wait until this one decides to finish the communication.
When you come back to "accept()" maybe the other side is no longer there and it could take a little bit 'till you find this out, and so on, causing your server to behave strangely and your clients to think it doesn't work, that it just wastes his time in internet, visiting FAQs just for fun and reading stupid jokes...

I suggest you use fork(), or just go with plain-old select(). If you dare to use threads then good luck.

PS: Did you forget about close()/shutdown() or didn't want to include'em in your example.

:D (HAL)

Add a Comment



New Questions: Why is my socket not ready for writing?

From: Julian

Why would a socket not be ready for writing? I have a program where I call write to send 1024 bytes of data to the socket. The call hangs in write. If I put a select call in just before the write, with a timeout, I see that the socket is not ready for reading or writing. Also there is no exception pending.

Why would this occur?

Thanks,

Julian

From: Rob Seace

One likely reason would be that your socket send buffers are full... Which probably means whoever you're sending to isn't reading nearly as fast as you're sending... Or, maybe the network connection temporarily went haywire, so packets are getting backed up... *shrug*

From: Julian Palmer

This is the conclusion I'm coming to as well. If I add *big* delays, then things work. Unfortunately the delays are too big to be acceptible in the code (order seconds between packets).

The situation here is the application is responding to a web page request issued by Netscape Navigator 4.5. The requested data is around 26KB and is being sent in 1KB socket writes for internal application reasons.

Is there anything funny known about how Netscape handles incoming HTML response pages that is relevant here? It works fine if the browser is Internet Explorer. It also seems to behave better if the amount of data being returned is smaller. Strangely enough,uploading bigger non-HTML files seems to be ok. Could it be that Netscape is doing something different with HTML than other document types?

fwiw the OS here for the application is a Linux 2.2 kernel. Is there any OS debug stuff I could look at that would tell me what was going on? Netstat just says the connection is ESTABLISHED. Tcpdump doesn't show any problems.

The help is much appreciated.

Julian

From: Rob Seace

Well, I suppose you could try running Netscape under "ptrace" and/or "strace"... That may or may not give you any useful info about what Netscape is up to... It's quite probably just some weird Netscape oddity, though (it certainly has TONS of those ;-))... Have you tried it with Mozilla? Or, Lynx?

Is there some reason you just can't block on the writes? Do you need to go do something else, instead? If so, maybe set your socket to non-blocking mode, and just periodically select() on it, to see if it's writable again, yet? *shrug*

From: Rob Seace

Er, I meant to say "strace" and/or "ltrace"... There is no "ptrace" (or, rather, it's a function call, not a binary)...

From: Julian Palmer

We can't just block on the write as it never unblocks. i.e. the socket is stuck for all time. Aborting the browser request (hitting the Stop button) clears everything of course.

We have tried other browsers. It works fine with all versions of IE we can use. It also works with the same rev of Netscape running on a Linux box, just not on Windows. It also works with Netscape 6 on Windows. Sigh... ;-) Also clearing the browser cache provokes it every time...

We're also going to try writing the data to the socket in different sized chunks to see if that affects it. We'll try the tools you suggest too - if they exist under Windows!

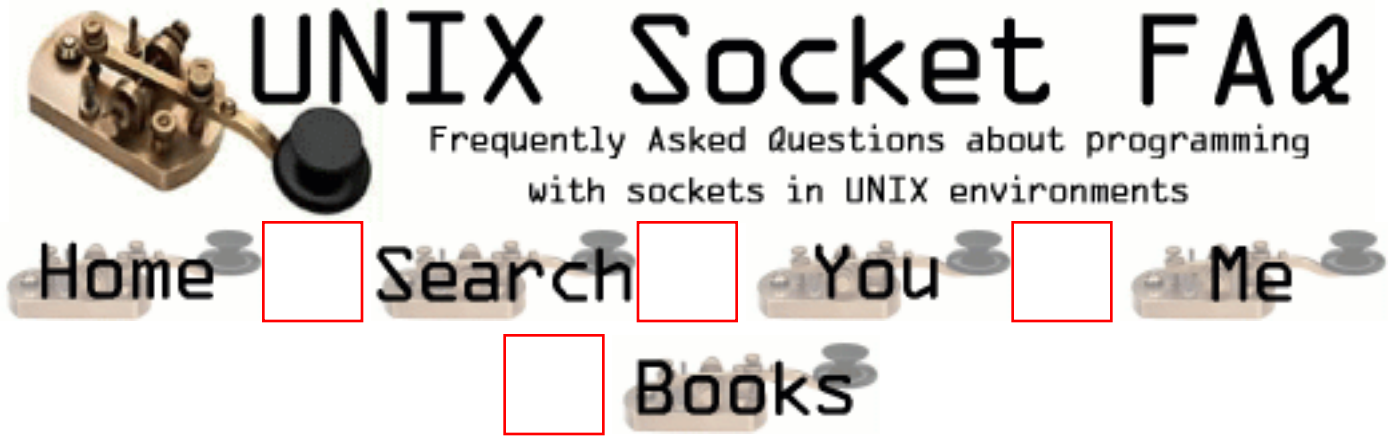
Is there anything I can look at on our application side on the Linux box that would tell me why the socket is stuck? e.g. an errno value or some such?

Julian

From: Rob Seace

Ah, I thought you were dealing with Netscape under Linux, or some other sane environment... ;-) I don't know what you could do to track things down under Windoze... Aren't odd bugs the normal system behavior, there?? ;-)

It's odd that the blockage never clears out at all... (How long have you waited, before giving up?) I would've thought it was just a matter of the client being too slow, and not reading the incoming packets fast enough... But, if so, it should EVENTUALLY get around to reading them, and thereby clear out your backed-up write buffers, and allow you to start sending more data... But, instead, it sounds like it's just getting totally locked-up, and refusing to read any more data from you, at all... I have no idea why it would do such a thing, though... I think you can safely chalk it up to a client app bug though, rather than something on your end... Of course, if you NEED to support the buggy client anyway, then I guess it unfortunately becomes your problem, anyway... ;-) I don't know what more to suggest to kluge around it, though... Maybe key off the browser's identity, and for Netscape on Windoze, only feed it data in tiny, slow, bite-sized chunks that it can swallow easily? ;-)



New Questions: writing to RAW ethernet socket

From: Dries Pruimboom

Hello,

I am trying to write a program that sends/receives a non
standard protocoll over an ethernet interface. So i opened
a socket in the following manner :

```
sock = socket(AF_INET,SOCK_PACKET,htons(ETH_P_ALL));
```

then i try to read from the socket using something like this :

```
read(sock,buffer,4096);
```

this works, i get a 'raw' ethernet packet, ethernet adresses etc,
packet size/type etc.

The problem is i don't know how to write data to it,
i assume it must be done with 'sendto', but how, what
is my sockaddr what are the flags i need etc.

i've tried the following, but it won't work :

(assuming addr contains a valid 6 bytes ethernet address

```
struct sockaddr ethaddr;

ethaddr.sa_family = AF_PACKET;
memcpy(ethaddr.sa_data,addr,6);
```

```
sendto(socket,send_data,128,0,&ethaddr,sizeof(struct sockaddr));
```

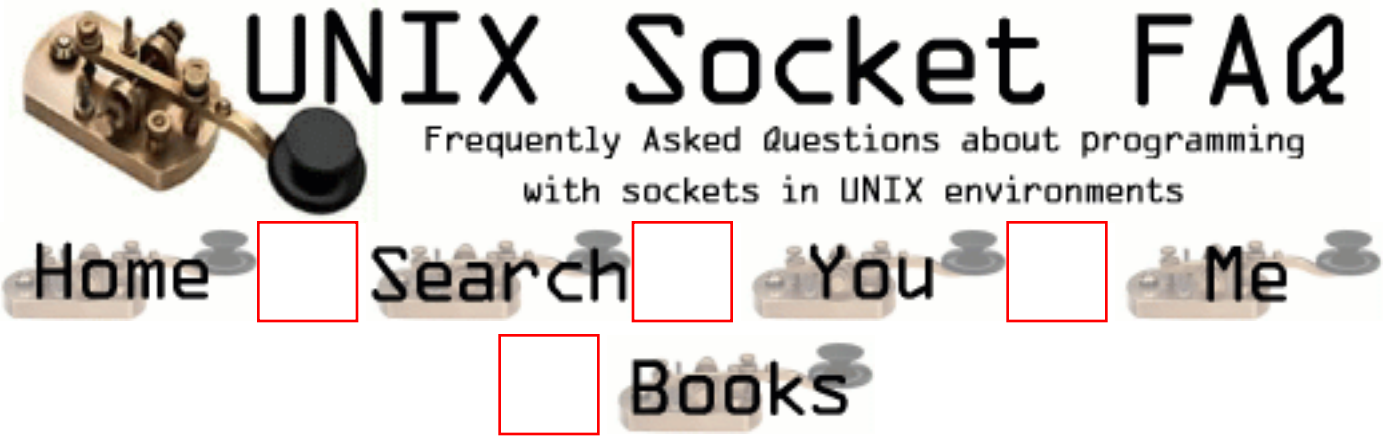
but this doesn't seem to work, does anyone know what's wrong ?

From: Rob Seace

You should take a look at the source for libnet... In fact, you might consider using it, perhaps in combination with libpcap, in order to have very portable raw packet reading/writing capabilities, rather than just directly coding up your own support, which is going to be very platform-dependent... (Note: if packetfactory.net is down, which it often seems to be, you can find libnet many other places... I'd recommend searching for it on packet storm...)

However, based on a quick peek at the libnet source, I think what you want to stuff in your sockaddr is the interface name which you want to send the packet through... (Eg: "eth0", or whatever...)

Add a Comment



New Questions: Tcp port configuration

From: Aida

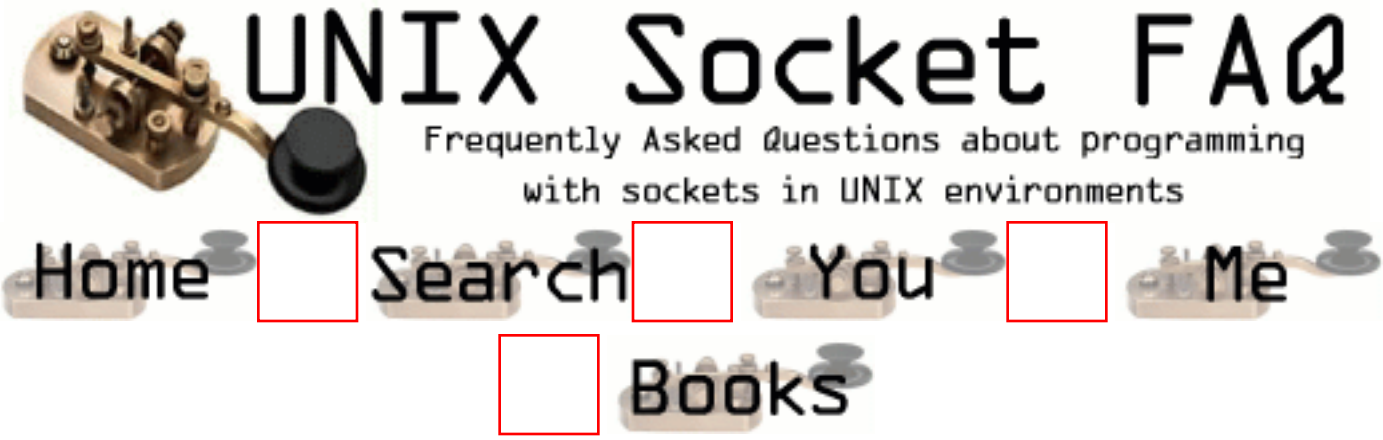
Hi!
I would like to know if there is a simple way of
finding out how to configure tcp port for Solaris 2.6.
What files should I look at (beside /etc/services)?
Are there some commands I have to run?

Regards
Aida

From:

no idea

Add a Comment

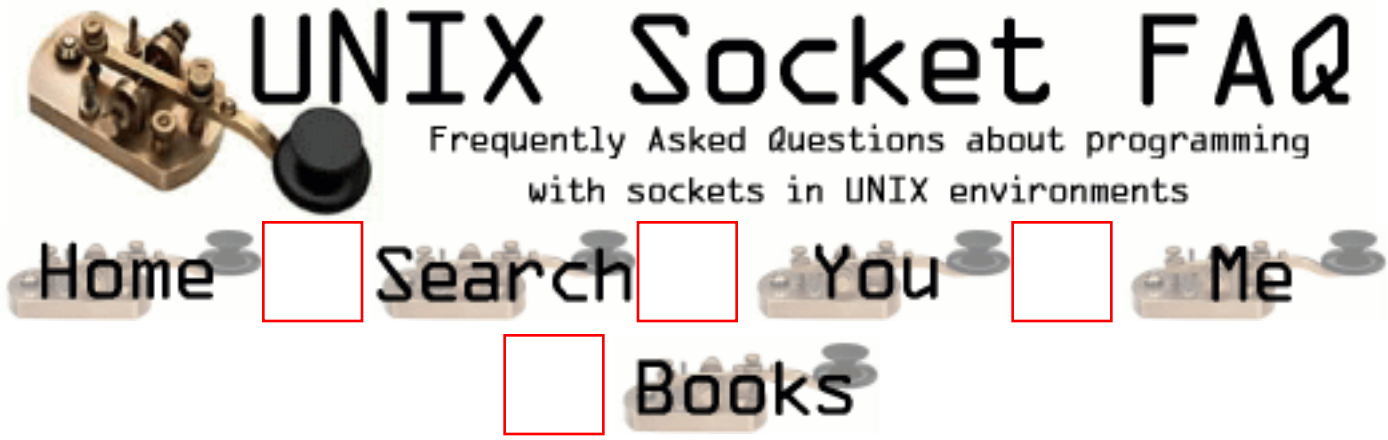


New Questions: Max data that can be send in one send

From: yogesh

Hi , I just wanted to know , what is the maximum size of data , or maximum bytes can I send in one send or sento ?

Add a Comment

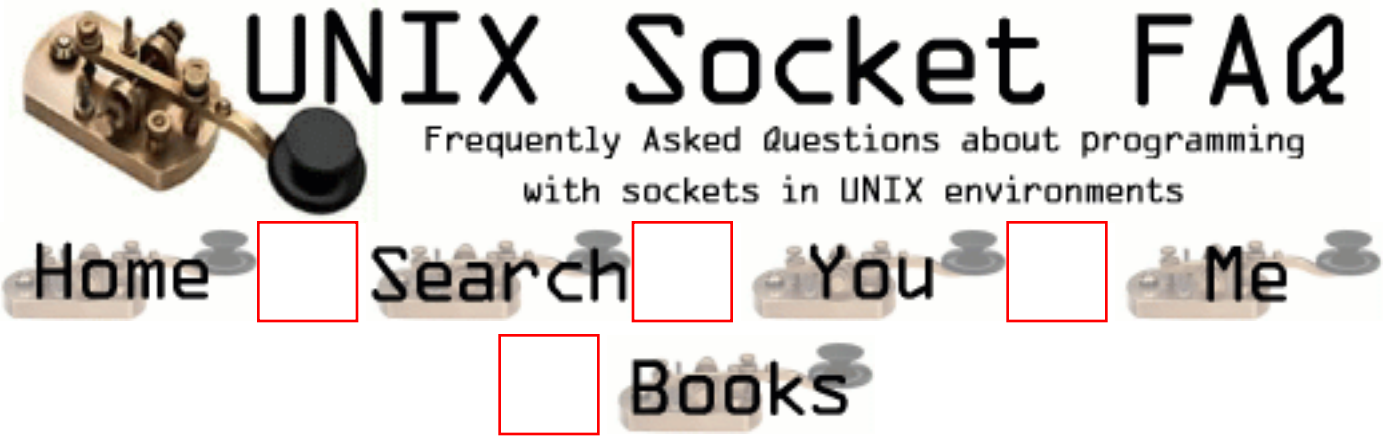


New Questions: Regarding Non blocking calls

From: yogesh

Hi ,in my application , no of clients are sending the data on the same server port , currently I am using Select since its a non blocking sockets .
Right now I am reading using datagram sockets i.e. recvfrom (), using this I can able to read only one message send by any client at a time, but is there any simple method via which I can read all the messsages waiting on the recv at a time and I need the IP adds of sender too.

Add a Comment



New Questions: Why do I get "accept: Too many open files" error

From: Jessy

Hi there,
We have written a C rpogram of a socket server. After a few users conect to it, I get accept : too many open files error which scrolls on the screen.
Can you please explain the cause of it?
Thank you.

From: Loco

Hi,

The error indicates that the process has run out of file descriptors, and as accept creates a new file descriptor when a client connects (you knew you needed a file descriptor for every connected socket didn't you?), it couldn't create it and failed to accept the connection.

This could be caused by many things (for example, taking a shower after 10:00 PM while the neighbor is watching the football game on Fox Sports...), but i've seen this happen when you fork a child process for every connection and the child process is the one that works on the client socket. People normally forget to close the socket in the parent process (reading the man pages you'll find out that close doesn't close the "socket or connection" unless it is the only descriptor for that socket, you would have to use "shutdown" to close it explicitly), which would just release the file descriptor. If this is what is happening, you'll run out of file descriptors after a few thousand connections (even from the same client).
You should close the descriptor after fork() from the parent:

```
pid = fork();
if (pid > 0) {
  // Parent, close the descriptor
  close(peer_socket);
```

```
}
else if (pid == 0) {
   // Child, do something, such as exec
   execl(...);
}
else {
   // Error
}
```

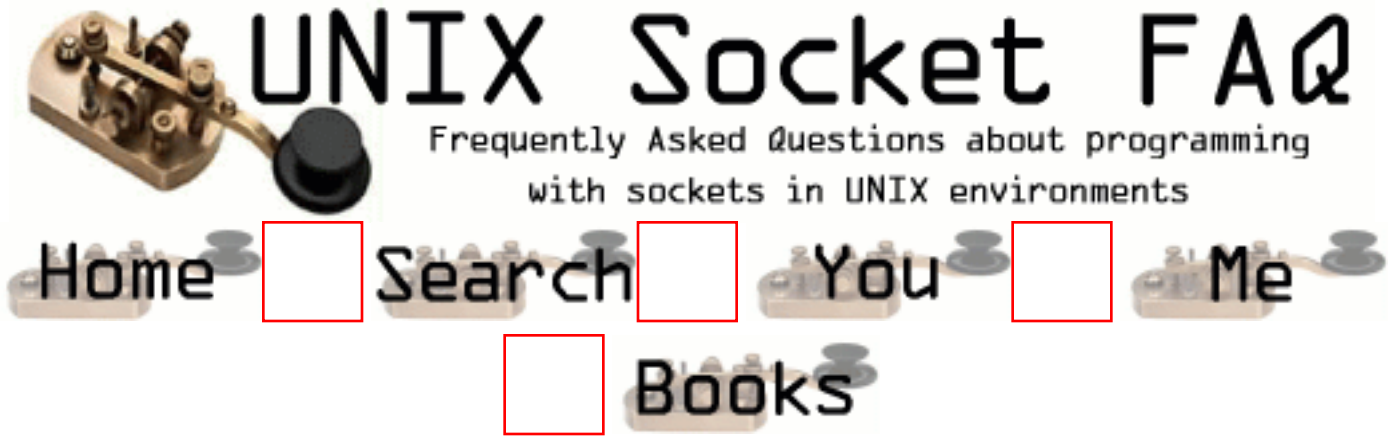
However, it is possible that you are losing file descriptors another way, so please post your code or a simple version of it to know what the problem is...
If you want to handle more descriptors than your process can handle currently, i think this guy Seace has written a lot about it in previous questions so go on and read about it.

:D (HAL)

From: Jessy

Hi THANK YOU. It worked.
I really appreciate your help.

Add a Comment

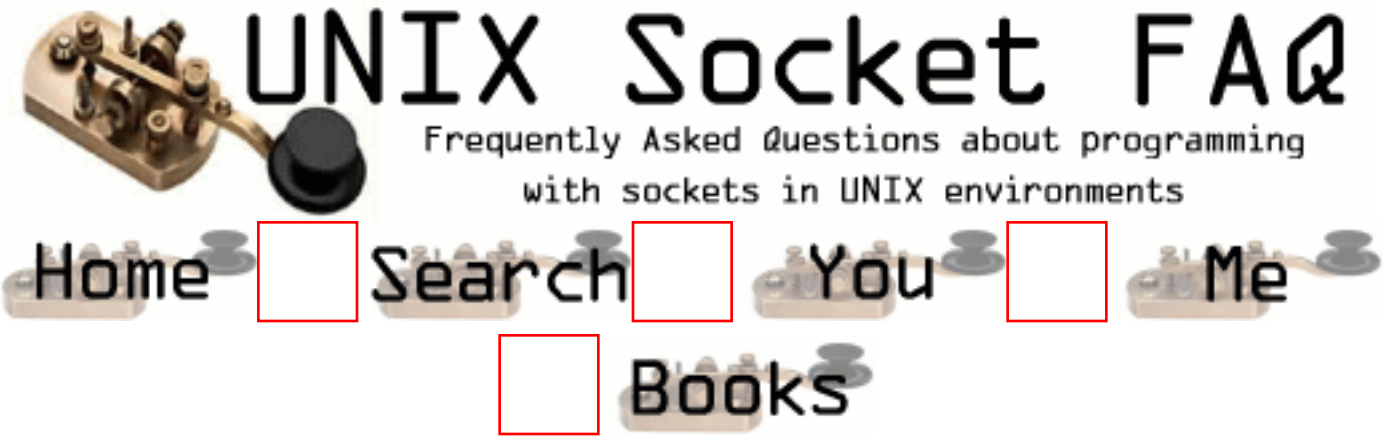


New Questions: Errors when a remote host crashes

From: Giorgio Spagnardi

If I'm waiting for read() (in blocking mode on a connect() socket) and the
sender host crashes(=turn off or reset), what errors my read() return?
Many Thaks

Add a Comment



New Questions: using poll for packet timeout

From: Justin

I'm trying to implement a sliding window algorithm and I'm trying to use poll to tell me when a packet has been gone too long and needs to be retransmitted. I can't figure out for the life of me how to use poll here. Poll seems to look at file descriptors but there is only one fd for the socket and not one for every packet that gets sent. so how do I check for timeouts on individual packets that are sent????? thanks

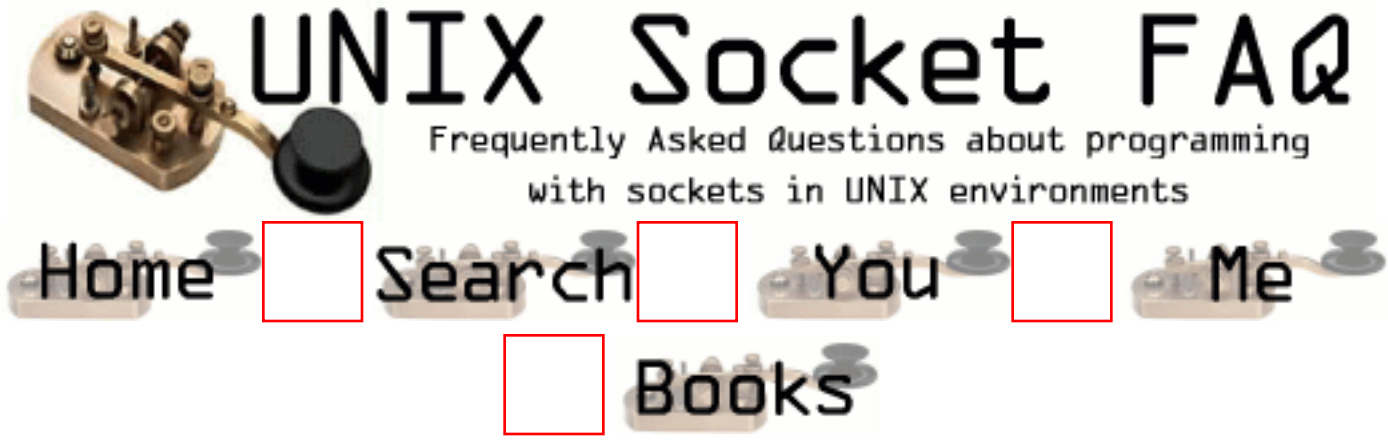
Justin

From: mrx

poll will use up too much CPU time.

you should use select()

Add a Comment



New Questions: How can I prevent from losing packet

From: Mitsuhiro Tokura

I use three sockets(that has different prot) in the same program. There is no loss if receive in slow speed. But as speed go up, losing packets happened. I tested the program that was changed from three sockets to one socket. This resulted in receiving without loss.
Please tell me the reason for losing packets.

From: Loco

I couldn't tell 'till i see some code.
A wild guess would be:
You're program is wasting too much time processing the packets, and when the buffer fills up maybe the OS is dropping the new packets as they arrive.

Post some code, maybe some people here can help you with your problem.

:D (HAL)

From: Mitsuhiro Tokura

Thanks for your advice. I post the part of cord at once.
I think it sure that program is wasting too much time processing the packets as you say. But I don't know what
cause this problem.
---------- code 1(use 3 sockets)----------

```
fd_set readOk, Mask;

width = 0;
FD_ZERO(&Mask);
FD_SET( soc_TKY, &Mask);
```

```
FD_SET( soc_OSK, &Mask);
FD_SET( soc_NGY, &Mask);
width = soc_NGY + 1;

while(1){
  readOk = Mask;
  switch( select(width, (fd_set *)&readOk, NULL,NULL,NULL ))
  {
    case -1:
      return( RETURN_ERROR ); break;
    case 0:break;
    default:
      memset(buf, NULL, sizeof( buf ));
      if(FD_ISSET( soc_TKY, &readOk)){
        addr_size = sizeof( struct sockaddr_in );
        if( soc_TKY !=-1 ){
          if( recvfrom(soc_TKY, buf, sizeof(buf), 0
           ,(struct sockaddr*)&server, &addr_size ) < 0 ){
            return( RETURN_ERROR );
          }
        }else{
            return( RETURN_ERROR );
        }
      }else if(FD_ISSET( soc_OSK, &readOk)){

            ( continue )

---------- code 2( use 1 socket )-------------

fd_set readOk, Mask;

width = 0;
FD_ZERO(&Mask);
FD_SET( soc_TKY, &Mask);
width = soc_TKY + 1;

while(1){
  readOk = Mask;
  switch( select(width, (fd_set *)&readOk, NULL,NULL,NULL))
  {
    case -1:
      return( RETURN_ERROR ); break;
    case 0:break;
    default:
      memset(buf, NULL, sizeof( buf ));
```

```
    if(FD_ISSET( soc_TKY, &readOk)){
      addr_size = sizeof( struct sockaddr_in );
      if( soc_TKY !=-1 ){
        if( recvfrom(soc_TKY, buf, sizeof(buf), 0
         ,(struct sockaddr*)&server, &addr_size ) < 0 ){
          return( RETURN_ERROR );
        }
      }else{
          return( RETURN_ERROR );
      }
    }
  }
          ( continue )
```

---

protocol : UDP/IP MULTICAST

From: Hector Lasso

Hi,

The code you show is very simple.
I see one thing that could be a problem:

You are putting your 3 sockets in the mask. However, when select() returns you check the first socket, if it isn't ready to receive check the second and so on. This could cause some trouble with the second and third sockets, for example:

Suppose your clients are sending packets constantly. When select() returns the 3 sockets are ready for reading. You check the first socket and recv() its packet. Go again to select() and as there are packets in all the sockets go again and recv() only on the first socket. This cold cause a "denial of service" to the other sockets for as long as the first socket is redy to read the others won't be processed.
I would change the code to look something like:

```
while (1) {
  readOk = Mask;
  switch (select(width, &readOk, NULL, NULL, NULL)) {
    case -1:
      return ERROR;
    case 0:
      break;
    default:
      // If buf is "char*" then sizeof(buf) sizeof(char*)
      memset(buf, 0, BUFFER_LENGTH);
      if (FD_ISSET(soc_TKY, &readOk)) {
```

```
        // process soc_TKY
      }
      // In your code it is "else if (...)"
      if (FD_ISSET(soc_OSK, &readOk)) {
        // process soc_OSK
      }
      if (FD_ISSET(soc_NGY, &readOk)) {
        // process soc_NGY
      }
      break;
  } // switch()
} // while()
```

From: Rob Seace

In addition to what Hector says, there's another problem with the code, as posted: you are making the clasic mistake of setting up your fd_sets outside of your select() loop... When select() returns, the contents of the fd_sets are changed, and you CAN'T just go and reuse them... You need to reset them every time before you call select()... Ie: you need to move all that initialization code INSIDE the loop...

From: Rob Seace

Ok, ignore me, I'm obviously blind today: I completely missed that "ReadOk = Mask" assignment... Duh... ;-)

From: Mitsuhiro Tokura

I tested the way adviced me in this page. But test resulted in the same before.(program lose receive datas.) I think searching in the descriptor list at the select() is slow, (I don't know this reason) so datas overflowed the socket buffer and lose. I use AIX ver4.3, is there anyone know this reason?

From: Stephan Krings

Maybe I misunderstood something completely, but you're using UDP (socket created with SOCK_DGRAM), aren't you?

If this is right, it's completely normal, that you're losing packets. A fast sender can easily "overrun" a receiver, especially if the latter one is serving on several sockets, processing the request etc. The packet may also be lost due to bad network conditions. In short: Datagram sockets are completely unreliable.

If you need reliability either use stream-sockets (SOCK_STREAM), or design your protocol, so

that it can retransmit lost packets.

See R. Stevens "Unix Network Programming" for an in-depth discussion of this topic.

From: Hector Lasso

Stephan,
I think the problem is not packets lost in the net. In the original question Mitsuhiro points that using one socket instead of three stops the packet loss, so for some reason, the code to handle three sockets is slow enough to make the sender "overrun" the receiver.
My comment was valid as the code posted by Mitsuhiro had a problem, however, the changes didn't work.

Mitsuhiro, i think you can do many things to stop losing packets. These are just ideas, i might be wrong:
- Use TCP connections as Stephan suggests.
- Create a retransmission protocol to send again packets lost (there are many ways to do this, some may adapt better to your requirements) - i'd rather use tcp connections instead of this
- Use a thread for each socket -> it doesn't assure you'll get the packets, you might have the same problems as with select(), however, it will process the data faster when one or two of the sockets are not receiving data as the recvfrom() call will block and the OS won't give processor time to those threads. Trying doesn't hurt...
- You can still use select() to check for data available, read the data and put it into a buffer that another thread with less priority can process, playing with priorities can help you avoid packet loss.

I hope this helps someway.

From: Mitsuhiro Tokura

I am most thankful for your advice. To tell the truth, this program receive information of the stock market, sending specifications is defined, so I can't use TCP/IP. And packets must be processed in good order. Because I am not familiar with c, it seems to be difficult to synchronize threads each other.

From: Hector Lasso

Mitsuhiro:

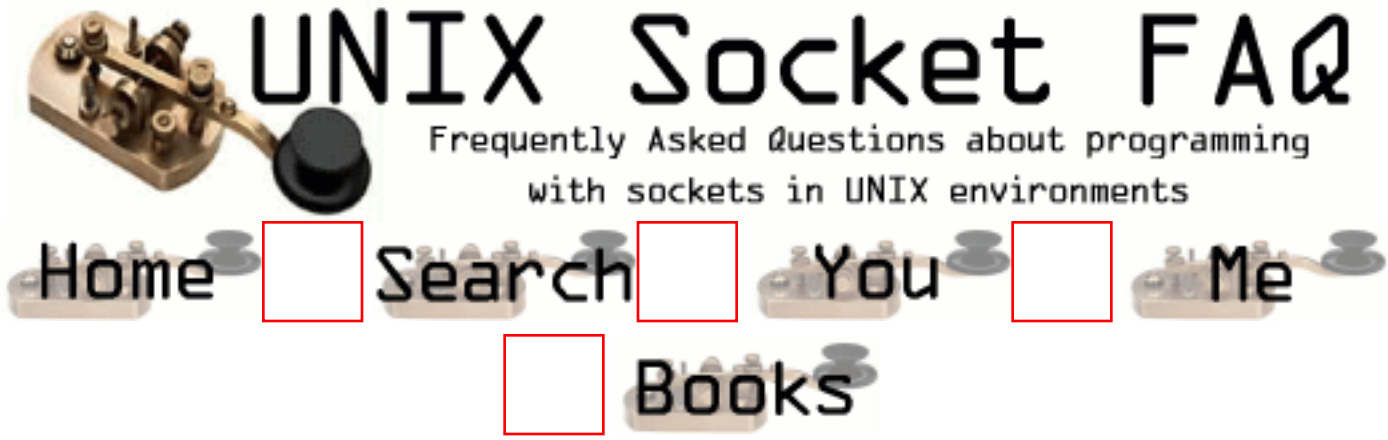
I see you receive from three different sources: Your test using one socket only received from one of them, or from all of them on the same port?
Do you still lose packets if you create three different processes each one dedicated to a different source? (this is a very simple test so i suggest you do it)
Do you need to receive from the three sources in a unique process? If you don't, and the previous question is negative (you don't lose packets with three processes) then we have worked it out.

UDP is a protocol meant to be used when you need speed but not accuracy. For example, in video streaming it doesn't matter if you lose a couple of frames per second, what really matters is that you have an average frame rate and have the frames in real time, you don't want perfect image of what the server sent a couple of hours ago but an acceptable image of what the server is sending right now.
If the sources use UDP with this high send rate (such that it can overflow the receiver's buffer) means you can afford losing some packets, and there might be a way to calculate the lost packets from the others if they are indispensable.
In fact, some packets are lost in the network, and you cannot do anything about it, you never know they were sent (unless they include some sort of serial number). If you have a way to detect lost packets then there may be a way to ask the sender to repeat the packet.

From: Mitsuhiro Tokura

You are right. The most important thing is that process handle the packets in real time as you say. I'm going to examine this process under understanding of packet losing.
By the way I tested using one socket only received from one of three sources. Your advice was very helpful to me.

Add a Comment



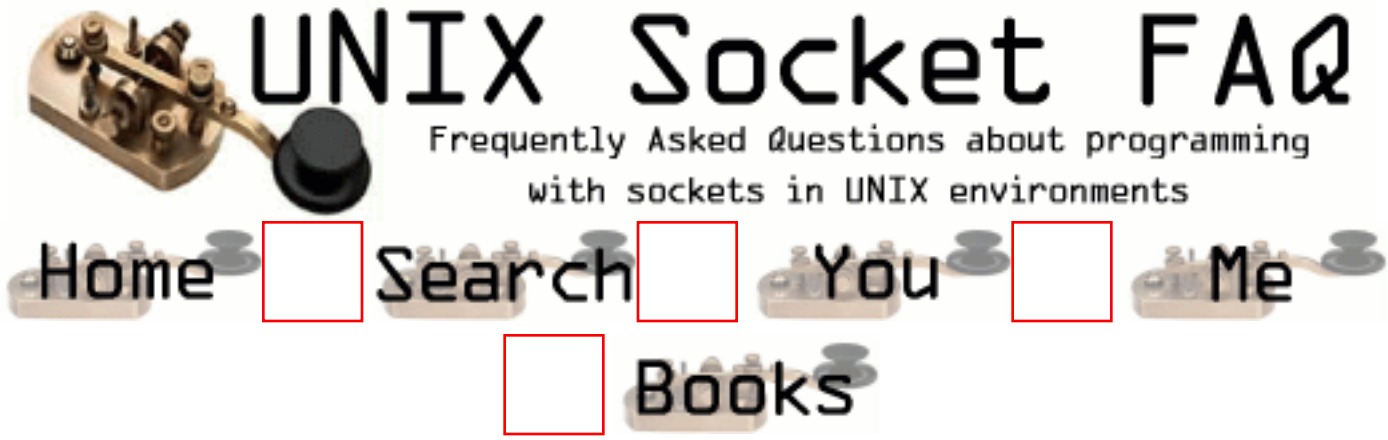
New Questions: client to multi-homed server

From: Dave Zavetz

Let's say I have a handful of machines, each with 6 physical interfaces, with each interface having it's own unique IP address. The machines are interconnected in a way so that there are multiple routes from each PC to another PC. Each PC has two functions, a software router and a data generator/collector. For now let's assume I'm using RIP and as of now I'm not running a DNS.

Now let's say that I need to get data from PC1 to PC4 via a socket connection (TCP or UDP). PC1 has 6 interfaces out and PC4 has 6 interfaces coming in. The socket connection will pass through PC2 and PC3, which in this case are acting simply as routers. I don't care which interfaces are used on PC1 and PC4 as long as it is the shortest path (RIP). For my client app on PC1, I'm not bind()ing to a specific client IP (outgoing) address. I assume that by doing this I'm letting the local routing table make that choice. But, what IP address do I try to connect() to on the server (PC4)? Like I said, there are 6 physical interfaces coming in to PC4 and I don't care which one is being used as long as it is part of the shortest path from PC1 -> PC4. It's almost like I need to assign my server app it's own IP address. Or somehow use DNS to assign all 6 interfaces the same hostname and let the routing protocol deal with it. Any suggestions?

Add a Comment



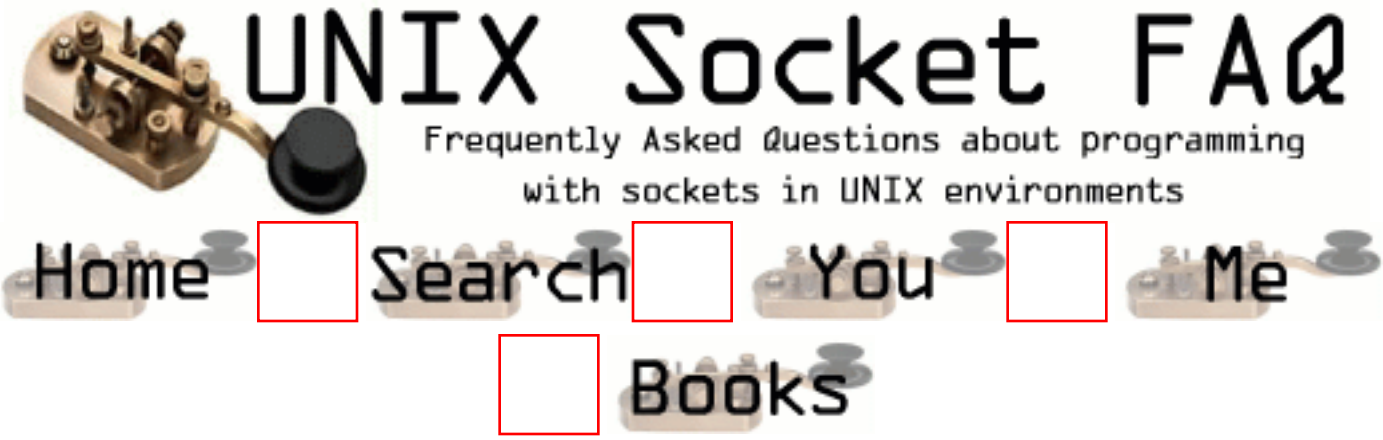
New Questions: deleting files and directories

From: Dalibor Sver

Since I'm not good at administration especially Unix, please help me: I can't delete (rm) file which has mod 755 in directory that has mod 777. I get constantly 'Permission denied' (file was uploaded throug my PHP script and directory was also created by the same script)

Thank you!

Add a Comment

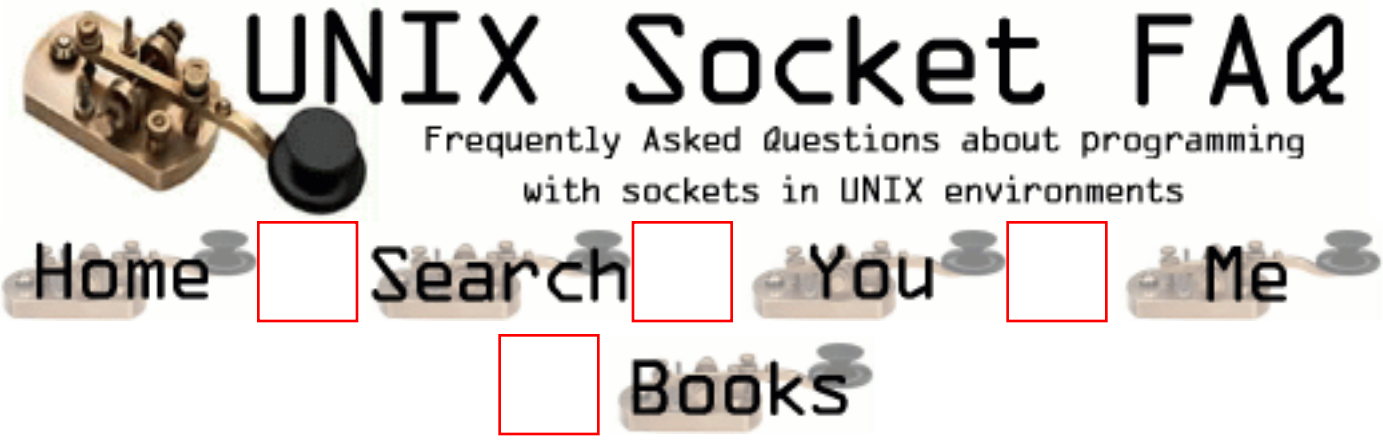


New Questions: packet demultiplexing

From: Priyanka Gupta

Hi,
Can anyone please explain to me the meaning of packet demultiplexing
logic withen the NIC.
Also, what is a two copy packet transport inetface??

Add a Comment



New Questions: number of connectons

From: matt

whats a sensible number of connections for a tcp socket
server to have?

whats a sensible number of clients to have connecting to
a tcp socket server at the same time?

i'm testing my code at the moment using perl scripts to put
upto 40 clients into the backgound at once, or with a second
delay between starting each client. At forty clients i begin
to have problems. is this reasonable? I'm using select() to
deal with each client and not fork()ing if that helps

thanx

matt

From: Mike

first: i hope it´s not too late (it´s already the 04/06/2001)

a few hints:
1. TCP/IP Layer is definitely able to handle more than 40 connections.
2. If you are using a Unix environment like Solaris
there is a "soft" and a "hard" limitation on open file-handles -> socket handle is only a file handle for the unix kernel
3. to get information about this "soft" limit:
ulimit -Sn

4. the "soft" limit could simply be changed by:
ulimit -Sn x

x being any number
If x is greater than the "hard" limit, it will be set to the hard limit.
the hard limit is kernel-dependent. for further details see the manufacturer of the system (SUN for ex., topic: tuning webservers)

Sorry, but i don´t know how the limits are set under windows nt, because normally i use a real operating system like solaris and don´t have to mess with MS for developing systems.

i hope this helped you a bit

mike

(anything written here is completely my personal opinion and has not been authorized by the company i´m currently working for)

From: Hector Lasso

Hi,

It might be a problem in your code. However, to know more about the problem could help a lot. If you are doing something like this:

```
do {
  select()
  for each socket {
    if (FD_ISSET(current socket)) {
      do something with data received
    }
  }
} forever
```

then probably the problem is you are spending too much time processing each client request and the others have to wait.

From: matt

thanx for the quick response guys - though i still
testing this week until i'm happy with it, which i'm almost
am as i don't expect much traffic going through it

For more infomation -
its running as IPC on Solaris 2.8 and the code also
configures/dials 20 modems as well so after its dealt with
the currently clients it has about 1000 lines of code to get
through before it looks for clients agains. Thinking about now
- is there away to accept ALL pendng clients before i deal with

them, for example while(select(fdreadset) > 0) {accept connection}.

The perl script i'm running is just
./myprogram&
sleep
./myprogram&
sleep
(same 40 times)

hope that helps

matt

From: Hector Lasso

Hi,

I think you have the idea:

```
do {
   // set timeout here
   FD_SET(fd_of_listening_socket, &readset);
   result = select(fd_o_l_s+1, &readset, NULL, NULL, &your_timeout);
   if (result > 0) {
      // Have connection (it's the only fd in the set)
      // Accept the connection
   }
} while (result > 0);
```

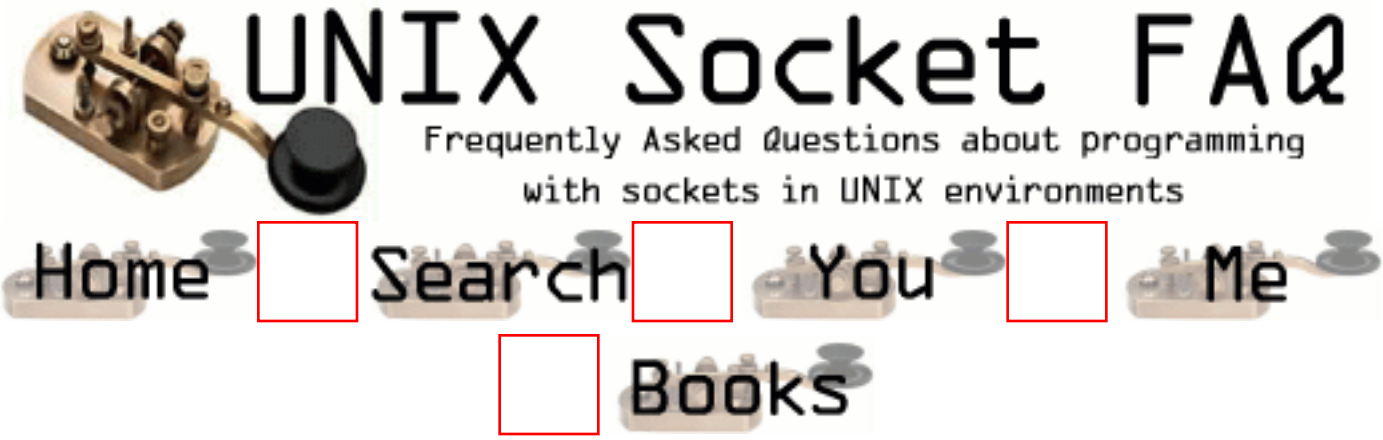
I check the result of select() inside the loop and not the set because if there's an error the set becomes undefined and you should not rely on it. As it is the only descriptor in the set i don't check it with FD_ISSET() (should i???)

From: Wolfgang Heinemann

There is another strange thing i found in Solrais 2.6 when using OpenLDAP.
There is a select loop inside the OpenLDAP code.
When the hard limit of connections is reached the accept does not retrun a valid
socket handle. Because i don't get a handle i can't close the TCP connection.
But the undelying driver level does noct drop the connection !
So the client connected to the LDAP server keeps connected.
If you snoop the connection you can see the 3-way handshake of the TCP connection

the connection is established on the client side.
The result ist that the server seems to hang. No more clients get connected because
the hanging clients don't drop their connections ...
I fixed the problem by changing the logic a little bit.
The server simply closes the highest acceptable connection when it get accept()ed.
This way there is allways one spare connection listening.
I'm not shure if this is the right way to handle the problem but it was
the only solution i could find. I think it's a bug in the Solrais protocollstack.
I don't know if this still exists in 2.8

Add a Comment

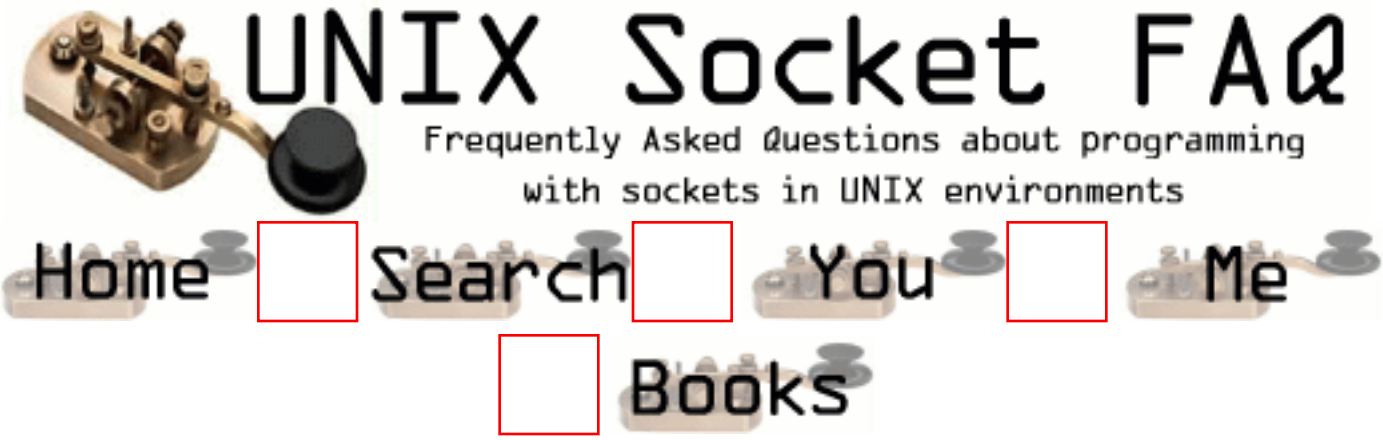


New Questions: Are there Token Ring programmer ?

From: NEC

Hi,
I'm a student and i have to make a program in C. This program must run as Token Ring.
If anyone want to tell me, links, help.....

Add a Comment



New Questions: UDP with retransmission?

From: Andreas

Hello!
In a client/server-connection with UDP, is it possible for the server to which packets that are lost?
I would like to try to implement some kind of retransmission, or at least find out how many packets that we are losing.

Thanx very much

Add a Comment

New Questions: UDP Server and Client on same machine gives Address already in use error

From: Halle

Hi.

I'm writing a very simple beginner UDP server and client, and when I try to run the server and client on the same machine, I get the infamous error of address already in use by bind(). This happens of course when I run the client after running the server.

Here are the crucial parts of my server code:

```
sd = socket(PF_INET,SOCK_DGRAM, 0);
bzero(&mon_s,sizeof(mon_s));
mon_s.sin_family = PF_INET;
mon_s.sin_port = htons(PORT);
n=bind(sd,(struct sockaddr*)&mon_s,sizeof(mon_s));
```

And of my client:

```
hs=gethostbyname("zulu");
sd = socket(PF_INET,SOCK_DGRAM, 0);
bzero(&son_s,sizeof(son_s));
bcopy(hs->h_addr, &son_s.sin_addr, hs->h_length);
son_s.sin_family = PF_INET;
son_s.sin_port = htons(PORT);
```

```
bzero(&mon_s,sizeof(mon_s));
mon_s.sin_addr.s_addr = INADDR_ANY;
mon_s.sin_family =PF_INET;
mon_s.sin_port = htons(PORT);

n=bind(sd,(struct sockaddr*)&mon_s,sizeof(mon_s));
```

where mon_s is the client's sock_addr_in structure, and son_s is the server's.

Would a call to the function setsockopt help me out here?
This works only when I run the server on zulu and then the client from anywhere but zulu.

Any help is greatly appreciated. Thanks!

From: Rob Seace

You're trying to give the client the same local port# as the server's port# you are connecting to... Do you really need to do that, for some reason? If not, you should probably just let the OS assign you a free port of its own choosing, rather than attempt to choose your own... If you really need the client to have that port, then you simply won't be able to ever run client and server on the same machine...

From: Halle

Well, I guess you figured that I had defined PORT as some number in both the client and the server (the same number). I tried changing the port number for the client, and I don't get the error anymore (yay! we're advancing!) however I don't believe the client manages to 'find' the server. It just sends its messages out into the ether, and the server is in lala land. So, if I changed the port number, how will the client know where the socket is? Forgive me if these are all dumb questions, I guess I reek of newbieness.

Thanks!

Halle.

From: Hector Lasso

Hi,

Your server binds the socket to port A.
The client uses 0 to bind the socket, so the OS assigns any free port to it.
The client will send packets to port A of server, so they will arrive without any problem!!!
However, if you are going to send any messages back (server to client) the server will have to know which port the client bound it to.
To do so, you just bind in the client to a different fixed port (for example B) and send messages

from server to client using B as the destination port. Another way would be to get it from the sockaddr_in structure passed as a parameter to recvfrom() - i guess this works, i've never done it.

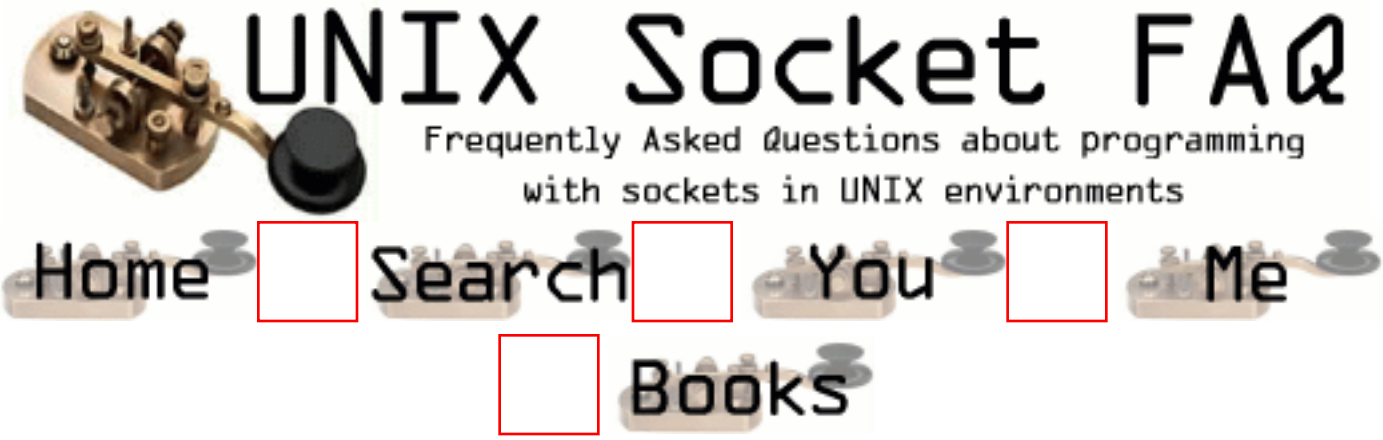
Best regards

From: Rob Seace

Yes, I'd definitely recommend having the client bind to port 0, so that the OS can choose a free ephemeral port, rather than using a fixed, hard-coded port of your own choosing... It's trivial for the server to see what port the client who just sent it a message is using; after all, it already needs to get the address info in order to know the IP to respond to; and, the port is there in the same sockaddr, so just use it as-is in the sendto(), and you're all set...

Servers should have fixed port numbers, but clients NEVER should, IMHO...

Add a Comment



New Questions: problem transferring file through socket

From: vinay

i send data from file in bunches of 4096 bytes
but i loose some bunches(one) occasionaly and not mostly.
when i recv it at other end.
where is the problem and whats the work around?

Add a Comment

New Questions: cgi and socket with DB

From: eric moulin

I want create an architecture where most Web client communicate on a DB via CGI and socket in C .
How can I do ?

From: Loco

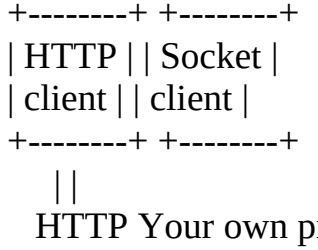
No problem. I'll do it, it'll cost you about $30K though...

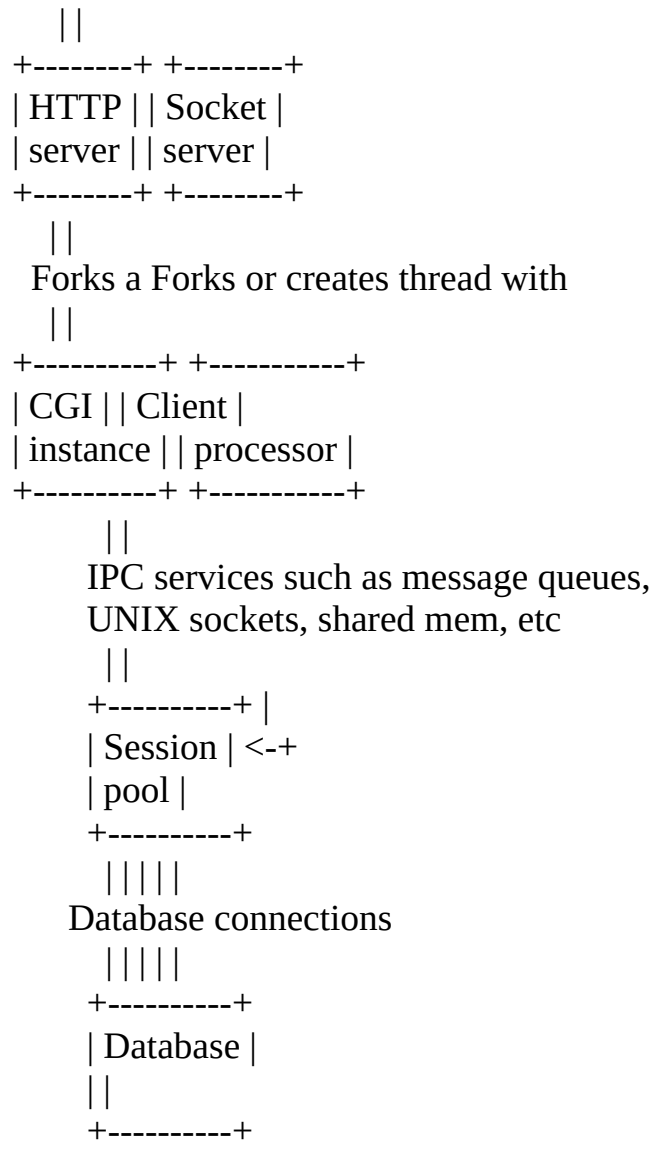
If you mean Database when you write DB then the following answer would apply, otherwise disregard this note:

I guess i would code a daemon which stays connected to the database and acts as a session pool. The CGI just uses IPC to get a session from the pool and request services from it. Very simple. Hey, i've already written that code!!!

Don't worry, i'll change the design:
Every CGI connects to the database, does whatever it has to do with the session, and once it has finished it just shut downs the session and dies (well, if you have one hell of a machine with 32 processors, 2GB RAM and the best OS in the market it should work really nice unless you execute something else like "vi")

As i haven't mentioned the "socket" part then i would add a socket interface to the first mentioned daemon... it doesn't matter where it comes from, the IPC is the important part:





What do you think? Is that what you were looking for?
Don't forget to send me the $30K

:D (HAL)

From: Loco

Sorry man, but my graphics were not so good after i posted the comment, so it'll be really troublesome trying to understand anything from the answer above...

The idea is:

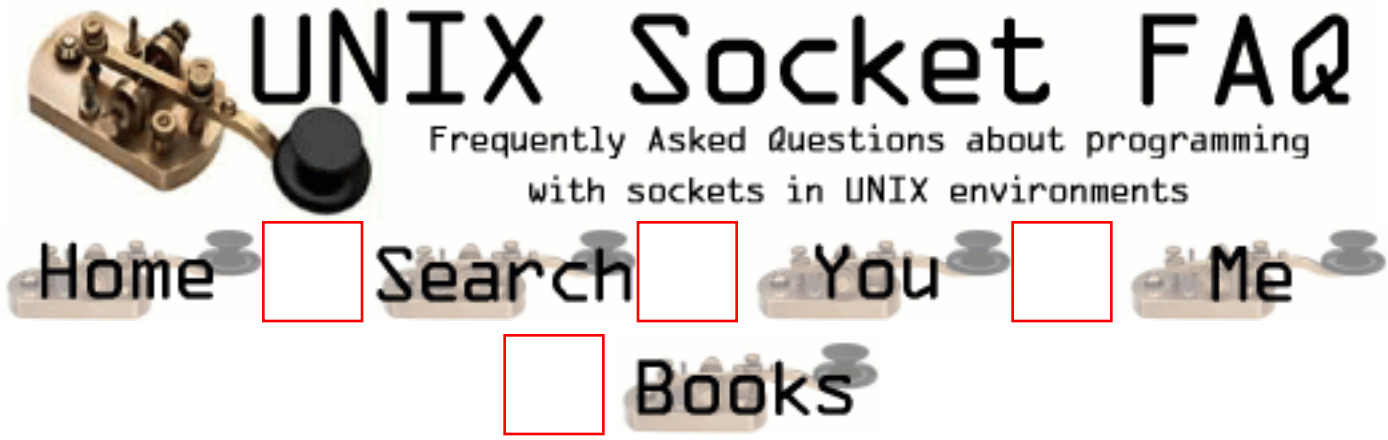
HTTP client --HTTP protocol--> HTTP Server (such as apache) --forks a--> CGI instance --communicates via IPC to--> Session pool --trough a DB session-->Database

The socket interface would work different from client to server, however, from that point on it would be the same system:

Socket client --own protocol over TCP or UDP--> socket server --forks or creates thread that--> process client requests --uses services (via IPC) of--> Session pool --trough a DB session-->Database

That is it!

Add a Comment

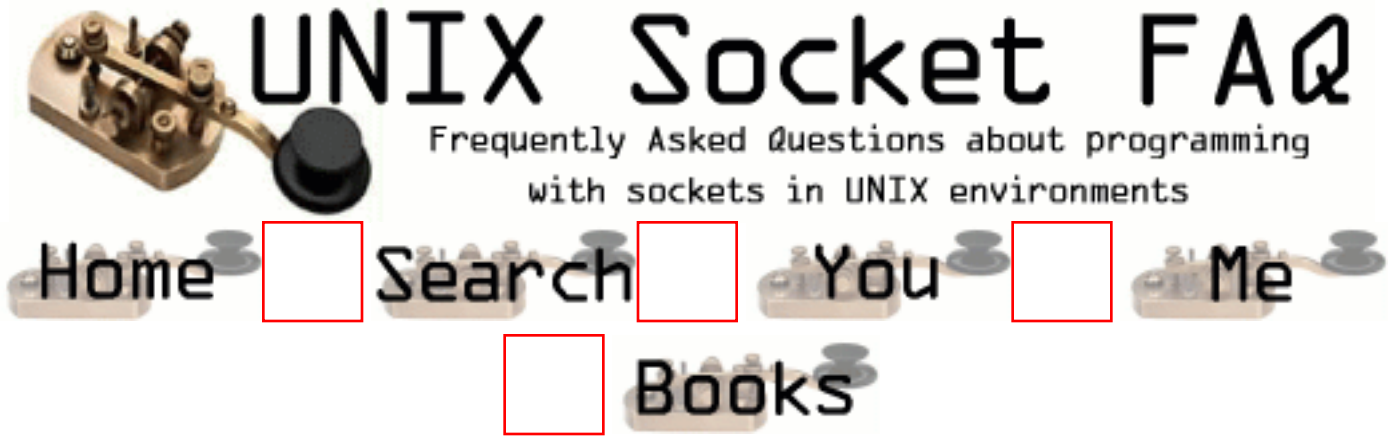


New Questions: question about UDP SOCKET buffering

From: Mitsuhiro Tokura

I made the program that use UDP/IP MULTICAST on AIX version 4.3. When this program was receiving packets, I checked the number generated by the "vmstat". Then "fre" of "avm" has been decreacing. And the number of generated by the "netstat -sm " has been increasing(especially "256") . I set socket option "SO_RCVBUF" to 256k. I think this is caused by soket buffering that carry out the "malloc" in mbuf whithout the "free". Is this true????

Add a Comment



New Questions: Broadcast a large file

From: Andreas

Is there anybody out there who knows how to do if I want to broadcast a large file to clients. All source codes and comments are welcome. Thanx

---

From: Loco

Send the files via email.

While this may seem an idiot response, i think it is easy and you won't have the problem of creating a new protocol. However, you will have to learn the SMTP protocol (which is almost rocket science. lol) and some of the encoding formats.
The pros are:
- You won't have to worry about network conditions between your server and each client, just with the SMTP server
- You can send the same file in just one connection using a list of addresses
It has some drawbacks:
- Your clients must have mailboxes
- You have to configure mail clients
- Each client will have to poll the mail server constantly
- The file size will increase because of the encoding making the message considerably larger than the file
- Message size is important as almost every mail server has quotas
- You won't have inmediate confirmation

Another option:
You could broadcast UDP packets in your network and expect confirmation from a list of "expected receivers". You should then serialize your packets so you could repeat the packets lost and the clients could reassemble the file using the packets. A sliding window algorithm could help you improve performance by expecting less confirmation packets.

:D (HAL)

From: Khubilai

I have tried to send files using UDP broadcast and it requires a lot of code to work. Too much code to post here. There is a lot to think about if you want to do it. You have to think about everything that can go wrong, and it's a lot when dealing with UDP. UDP broadcast doesn't work over the Internet. Contact me if there is anything you want to know about UDP broadcast, but remember, there's a lot of work to make it work.

From: Hector Lasso

Hi,

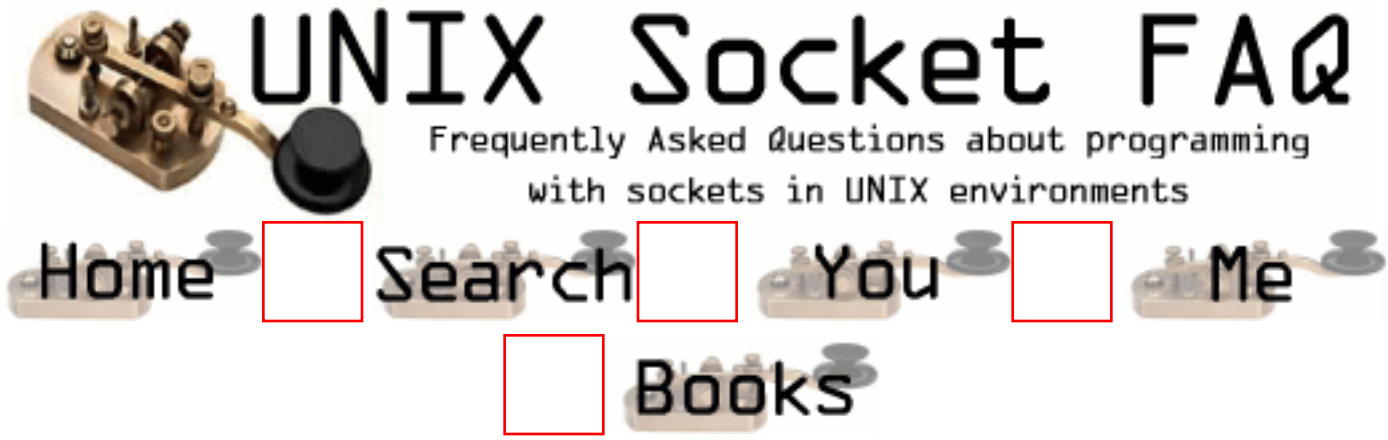
I think using UDP broadcast packets would be very complex.

My choice would be to use TCP connections from clients, sending the file through every socket connected. Using select() you can follow up advance for every client connected. If your clients have different bandwiths, and the transfer time is different i would use dedicated processes or threads.

From: Khubilai

TCP/IP is the easiest way to make this work. It may take more time to transfer the files, but it will save you a lot of time when you are going to write the code. I think I used around 1000 lines of code to make my transfers work when I used UDP broadcast. TCP/IP transfers is more reliable than UDP transfers. It's differculd to find every bug when you use UDP.

Add a Comment



New Questions: recv Socket

From: Fabien

I already read the subjcet which deals with the reading of data, but i don't really find solution. The server send a lot of inforamtions. The client don't answer to the server. The client must simply read the data sent by the server. But i don't know the size of these data. How do you do to adjust the size of the buffer in the instruction recv or read

---

From: kathy

Hey Fabien
u can adjust the size in send and recv , u only need to use the right commands used dont be confused with different books , every book uses its own way of explanation.

From: Hector Lasso

Fabien,

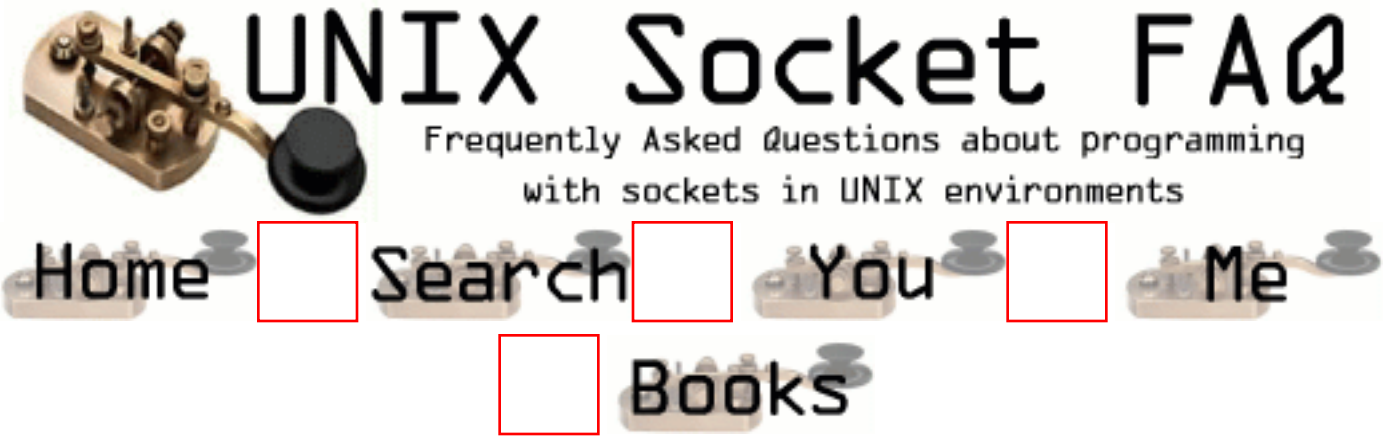
You can allocate memory (either dinamically or have a static buffer) that could hold the data if you know the maximum size of data you will receive beforehand.

You can also allocate a small buffer and grow it when needed.
For example:

```
#define IN_SIZE 10000
#define INCREM 5000
#define CHUNKS 1024
...
char *buffer, *temp;
int rec, offset, size;
buffer = (char*)malloc(IN_SIZE);
size = IN_SIZE;
offset = 0;
do {
```

```
  rec = recv(sockfd, buffer+offset, CHUNKS, 0);
  if (rec > 0) {
    offset += rec;
    if ((offset+CHUNKS) >= size) {
      temp = realloc(buffer, size+INCREM);
      if (temp == NULL)
        break;
      buffer = temp;
      size += INCREM;
    }
  }
  ...
} while (some_condition);
```

In this code the buffer is created with an initial size of IN_SIZE. The message is read in chunks of 1024 bytes, and added to the end of the buffer.
The offset variable will contain the point of insertion of the data received, which is also the total length of the data read.
In this code i didn't check for errors (except for realloc()).
There is a problem with this code: It expands the buffer although the memory is not needed yet (and probably won't be needed), and the buffer grows using increments that can be too big or too small. To expand buffer size after knowing the data won't fit into the current buffer you could use a temporary buffer to hold data read, and if it doesn't fit then you expand the buffer and copy the data.
You can also use the code above and after reading all the data you just allocate a new buffer sized to the total bytes read, copy the data and deallocate the old buffer (i wouldn't do this but some people may find it useful).
You can play around with the code until you find the solution that best solves your requirements.



New Questions: Telecom call routing

From: AGF

we are trying to simulate a call processing situation.
Say Client A dials Client B.
There is a switch S that routes the call.
Client A now dials the digits.Switch S receives the digits as a server and then dials Client B acting as the Client Switch S.
We are stuck up if Client C dials and wants to talk to Client D.The concept of having a thread -does it apply here
then how are the port numbers assigned?

From: Loco

This is what i understood from your message:
You are simulating a simple telephone network using sockets.
You have many client processes (different computers perhaps) which represent the end users of the network (telephones, faxes, etc)
You have a server process (other computer) that represents the switch of the network.
You need your switch to receive the connection request from clients, and connect to the other clients if available, and act as intermediary in the communication between clients.

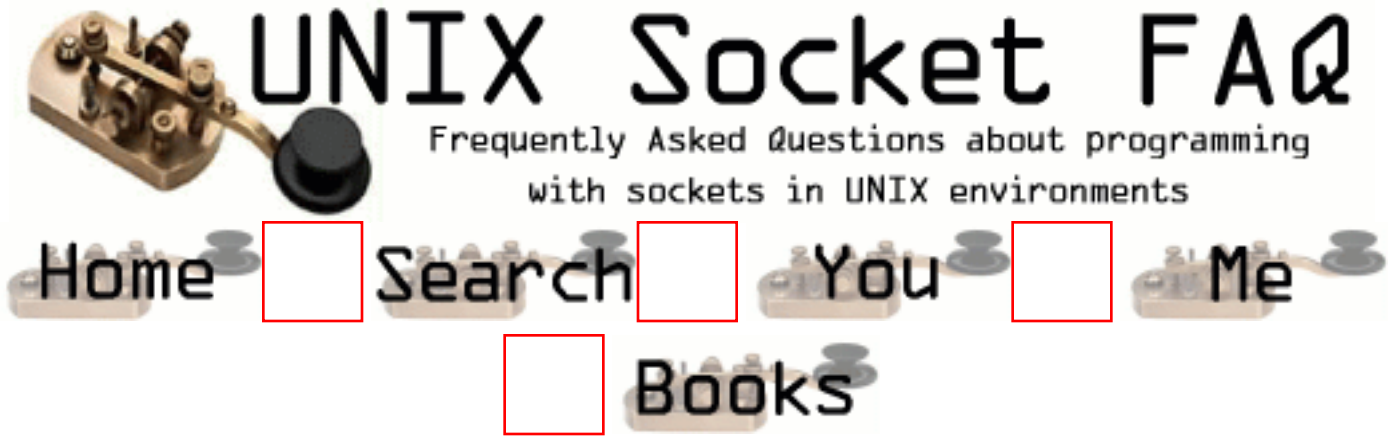
This is what i suggest:
If you keep open TCP connections between each client and the switch, then each TCP connection would be emulating physical lines/wires. So the switch has to maintain a list of circuits connected, parties involved, etc.
When one of the clients (let's say A) dials another client (B), A sends through the socket (wire/line) a message requesting the connection. The switch should decide whether to establish the connection or not and signal the other party (B) that there is an incoming call using the currently established sockets. If you use this approach, adding a more complicated structure to the network wouldn't be a problem, you could just add more switches that have their local clients, and have specialized connections with other switches. These special connections must be able to handle a fixed number of circuits to simulate the real trunks (is that how they call

them) between switches. As two switches might have more than one active circuit going through their communication "channel" you must design some kind of multiplexing for this channel.

Your question tells me that you are establishing TCP connections or sendign UDP packets when the clients dial the numbers. I suggest you change your approach to the one i explained above, it resembles more closely the network (the lines are there already established, so are the TCP connections, a new call is just some signals put on the line).

Regarding port numbers, you would use just one TCP port on the server, and the clients would connect to this port. Each new client that is connected and is assigned an ID in your telephone network is just like a new line that is added to the network.
The server could handle the connections using select(), threads, or even separate processes. As you are working with small number of clients and the code doesn't seem to be very difficult i suggest you use select(). Also as you need to handle a centralized structure to know states, parties and stuff like that, you would have to be very careful when accessing those objects from different threads.

:D (HAL)



New Questions: why is inet_addr() not working??

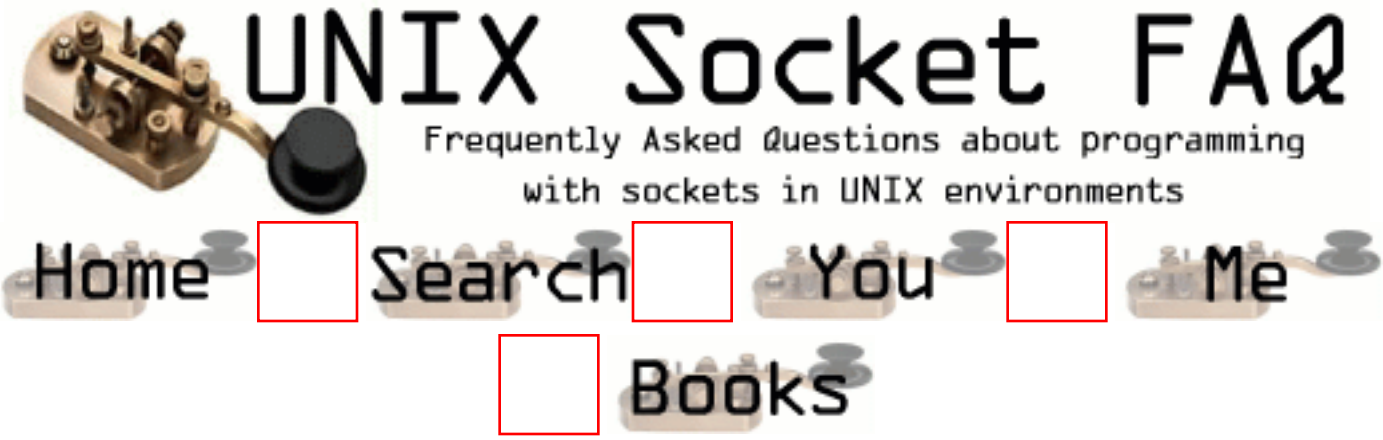
From: dhairesh oza

i tried compilling a code which used inet_addr(char *) to convert from dotted ip form to sin_addr form but the compiler is saying unidentified symbol inet_addr in file tcp.c. i have included files netinet/in.h and arpa/inet.h
this is on sco unix 7.1 so what might be the prob?

From: Rajaji

Check out the man page whether we need to include any library while compiling the code like -lnsl, -lxns etc...

Add a Comment



New Questions: NFILE parameter increases!

From: Jessy

Hi there,
I've written a SOCKET server program on HP-UX 10.20.
I use close and then shutdown for all requests from clients to close their session.
But what I have noticed is that, the NFILE parameter of the system table increases tremendously.
Isn't it suppose to decrease everytime a connection is closed?
Can someone please please please help me. I need to know how can I enusre that the NFILE does not increase!
Everytime the a certain limit is reached(of the NFILE), I get the Accept : Too many open files error scrolling on the console and the socket server is shutdown.

PLease someone help me. I am really confused and have come to a dead end!

Thank you so much for your attention.

Regards,
Jessy

From: Loco

If you fork() a process for every connection established, then you might be running out of file descriptors on the listnening process if you don't close the peer socket file descriptor after forking the other process.
:D (HAL)

From: Loco

Hi,

I think i already told you that. So i suppose it isn't the problem.

The file descriptors are released when you "close()" file descriptors not when you close the connections. Normally issuing close() on a file descriptor closes the corresponding connection, however, if there are any other processes (children for example) that share the same connection then the connection won't be released (even the same process can have many descriptors for a socket).
You say that you use close and then shutdown to close your connections, this could be wrong: when you close() the file descriptor it will be released and be available to use again by the process, after that, any call to IO functions using that descriptor will return EBADF which means the file descriptor is invalid, so shutdown() will not have any effect. If are doing this and another process has a file descriptor for the same socket, then you are releasing the descriptor but not closing the connection, it will be closed as soon as all the file descriptors related to that socket are closed.

One common error is what i told in the previous comment. You fork() a child to handle a single socket. This client shutdown()s and close()s the descriptor, however you don't close the descriptor from the parent process, as it will be receiving new connections it will run out of fd and won't be able to accept incoming connections.

If you printf the file descriptor returned by accept() it should be always the same if you close them after forking children (and don't create new file descriptors in the mean time)

I hope this helps, however, posting code would be better if you don't find any solution.

Don't worry, i usually don't fee more than $3K or $4K per hour...

:D (HAL)

From: Anthony

Hi Loco,

I'm a colleague of Jessy.
Firstly, thank for your prompt answer.
In fact I am closing the peer connection for every fork process connection establish. And the way the connection is closed is as follows :-

where, for every accept call and child is forked, I do a close and shutdown process to the peer. In fact the child processes are getting closed properly.

Is there anything I need to do like flushing out the process or .....

Your kind expert advice is really needed here...

Awaiting for your advice.

TQ
/Tony

From: Loco

Maybe i'm not such expert because i cannot see any other posibility for such a problem. I would like to take a look at your code if it is possible. If you don't want to post the code in the FAQ there is no problem, just send it to my email address (locohal@yahoo.com) and i will answer as soon as possible. However, it's better to post the code in the FAQ cause many of the people who access this site are really good in this and could find the problem as well. (Others are completely crazy or don't have anything else to do)

It would be better that you send a simplified version of your code that can reproduce the behaviour. I will test it on Linux and on HP-UX 10.20
I hope it is a programming error or something and not a problem with the OS (which i have seen behave strangely under certain conditions - but what the hell i'm not an expert on this and the sysadmin was doing a really bad job)

:) (HAL)

From: Anthony Raj

Hi Loco,

Here is my code on closing the peer-socket connection...

```
if (bind(sockfd, (struct sockaddr *)&dnet_addr, sizeof(struct sockaddr)) == -1) {
perror("bind");
close(sockfd);
exit(1);
}

if (listen(sockfd,BACKLOG) == -1) {
perror("Error in Socket Listen listen \n",errno);
exit(1);
}

logfile=fopen("socketlog","a");


while(1) { /* main accept() loop */
sin_size = sizeof(struct sockaddr_in);
if ((new_fd = accept(sockfd, (struct sockaddr *)&user_addr, &sin_size)) == -1) {
perror("accept");
continue;
}
```

```
printf("server: got connection from %s\n", inet_ntoa(user_addr.sin_addr));
if (!fork()) { /* child process */
  close(sockfd);
shutdown(sockfd,2);
  while ((n=read(new_fd,readbuf, LINELEN)) > 0) {
  readbuf[n] = '\0';
/* (void) fputs (readbuf,stdout); */

                strcat(buf,readbuf);
```

Your expert advice is required on what the hell I am doing wrong....

TQ

From: Rob Seace

Well, you neglected to show the parent portion of your code... In particular, I'm wondering if the parent is closing "new_fd"... If not, that would certainly explain any FD leakage...

Also, you shouldn't call shutdown() AFTER calling close()... In fact, if you're going to close(), there's no need for a shutdown() at all, generally... But, IF you're going to do a shutdown(), you certainly want it BEFORE the close()... (Note: I don't think this has anything to do with you leaking FDs; it's just a different, minor bug... I'm guessing that shutdown() is really failing with EBADF, or something similar, but you're just not checking for the failure...)

From: Loco

Hi Anthony,

Rob is right. Your shutdown() call should fail because the file descriptor is no longer valid after close(). Probably the error is EBADF. As he says this is not the problem for the lost descriptors. I would also like to see the parent code in the fork() where i think the problem might be.

You are either not checking for an error condition after the fork or always showing an error. fork() returns:
>0 to the parent process indicating everything went just fine and the number will be the PID of the child process.
0 to the child process (you are checkiing this)
-1 to the parent process indicating there was an error and the child could not be created.

I guess your code continues like this:

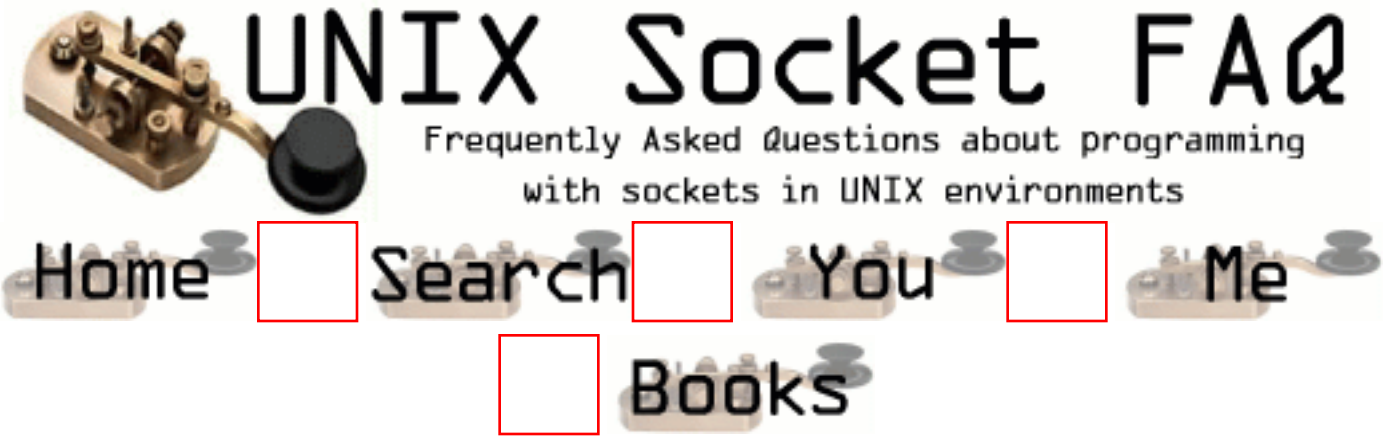
```
if (!fork()) {
   your child code here
}
close(new_fd);
```

If you don't have this last line then you are losing file descriptors each time a new child is created.

fork() creates a copy of your process which (fortunately or unfortunately) copy the file descriptors, which means both processes will have file equal but separated file descriptors (just like when you use dup(), but with the same number). So if you are not planning to use a specific descriptor on one of the processes then you should release it.

:D (HAL)

Add a Comment



New Questions: Blocking Vs Non-blocking

From: Rajaji Selvarajau

Can anyone compare the blocking mode and non-blocking mode?
What happens exactly in each mode?

Thanks for all your attention.

Rajaji.

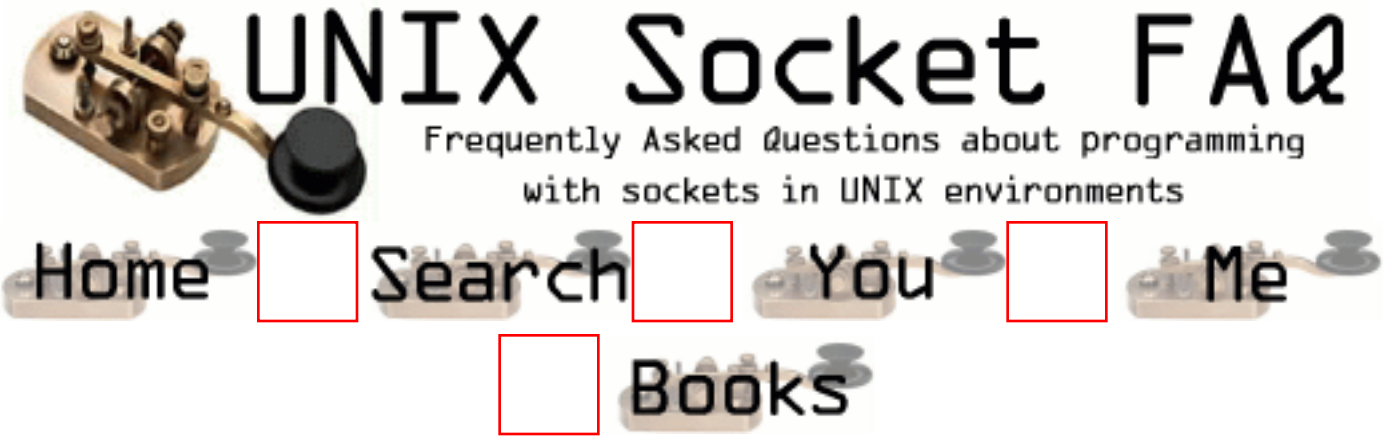
From: Loco

When you call IO blocking functions the OS will pause the thread until the task can be completed (data available or buffers available). In fact, the thread will enter a special state known as BLOCKED (i guess that is why they call it blocking calls). This means that the thread stops executing until the OS can fulfill the caller's request or some other event happens. Non-blocking function calls return inmediately whether the OS could or could not fulfill the request. In the former case, it will return whatever the caller asked for (eg. data received), in the later case it will return an error indicating that the operation could not be completed because the system would have to block the thread.

In the caller perspective it means:
BLOCKING
- When you use recvxxx()/read() your process/thread will wait until there is some data available, a signal is received or an error occurs.
- When you use sendxxx()/write() your process/thread will block until the buffer is able to hold the data you are trying to send, a signal is received or an error occurs.
NON-BLOCKING
- sendxxx()/write() will send the data inmediately (write to the buffer) if there is space available, or else will return -1 and errno will be EAGAIN/EWOULDBLOCK
- recvxxx()/read() will read as many bytes as there are in the buffer (or the number of bytes you told it to read, whichever is less) and will return inmediately, but if the buffer is empty then it will return -1 and will set errno to EAGAIN

:D (HAL)

Add a Comment



New Questions: TCP connection re-establishment

From: Mahesh Kallanagoudar

I am facing the following problem, after re-establishing a TCP connection.
It is a simple Client/Server scenario with TCP connection on Solaris UNIX.
Client connects to the Server and both start exchanging the data. I kill the server
program and re-start it. When the server program is killed, client gets an indication (through
a read returning ECONNRESET). The Client trys to re-establish the connection. The
re-establishment of
connection succeeds, it reads messages, but write fails with broken PIPE error.
I found that the problem was due to CLOSE WAIT TIMER of TCP. If I wait for CLOSE WAIT
period (3-4 mins), before re-establishing the TCP
connection, it works fine. Is there any other way of getting around this problem??
I tried Linger option, but some how I didn't work.

regards,
mahesh

From: Rob Seace

Is it really CLOSE_WAIT you mean, or is it really TIME_WAIT?
I'm guess it's probably really TIME_WAIT... I assume before
reconnecting, you close() the old FD? Then, I'm guessing
you probably just try to reuse the same local (client) port
in the new connection, right? That would be the main cause
of difficulties... Just zero out the port, and let the
system choose a new ephemeral port for the new connection...

From: Hector Lasso

I don't know how you can do to re-establish a connection, be able to read but not be able to

write.
When you say "...it reads messages..." you mean you receive data sent by the other party? or that read()/recv() doesn't return an error?

The only way i could get this behaviour is by shutting down the out stream on the socket "shutdown(fd, 1);".

Unless you are not telling everything that happens i don't think the problem is caused by a programming error but a system bug.

From: Mahesh Kallanagoudar

THanks for the reply Rob and Hector.

Rob,
I don't know TCP much. What is the difference between
CLOSE_WAIT and TIME_WAIT? I thought both are same.
Your guess is right. I close the fd and use the same
port. I shall try your suggestion of using system assigned port.

Hector,
By read, I mean it receives the data sent by the other party.
I also checked netstat it showns the connection in Established state.
That's what suprises me also. The thing is it works fine If I wait
for 4 mins before re-establishing the connection. In one of TCP FAQs,
I read a related comment by Richard Stevens. He says you should wait for
CLOSE_WAIT time, before re-establishing the connection. Otherwise, the packets
of previous connection which are stuck in some router may re-appear in the new
connection causing sequence number problems. He calls them
wandering duplicates.

regards,
mahesh

From: Rob Seace

The "wandering duplicates" issue you're talking about is
indeed in reference to TIME_WAIT, not CLOSE_WAIT... As I
recall, CLOSE_WAIT is the state that one side gets into when
it has sent its FIN, and is waiting on the other side to
send its FIN, to complete the shutdown of the connection...
But, that's purely off the top of my head, so don't trust
it that far... ;-) Actually, I forgot, I've got a copy of
the "TCP/IP Illustrated" set by Stevens sitting right here
beside me! It looks like I had that backwards: CLOSE_WAIT
is done on the passive side; ie: the one who receives a FIN

from the other side goes into CLOSE_WAIT until it finally
sends its own FIN to complete the shutdown (when the app
finally issues its close())...

However, TIME_WAIT is the state entered into on the active
side of a close (ie: the one who sends the first FIN), after
the full completion of the closing... Its only purpose is
to hold the port unusable for a while, to prevent the
"wandering duplicates" you speak of...

So, yes, if you're just trying to reuse your local client
port while it's waiting in TIME_WAIT state, that could
certainly cause problems... But, I'm really surprised your
system allows you to reuse the port while it's in that
state, at all... *shrug*

From: Mahesh Kallanagoudar

Hi Rob,

I even tried your suggestion of using system assigned port.
(Zero the port number in the socket call()) It didn't help.
What surprises me is if I see netstat it shows the conenction
as ESTABLISHED. But I still get the broken pipe error for write.
Before actually re-establishing the connection, I did netstat.
It showed the connection in CLOSE_WAIT state. (On the active close side)
It might sound crazy to you. But I am facing this problem.
I cann't suspect the code because it works fine, If I wait for
3-4 mins (By having a sleep call) before re-establishing the
connection. I don't know how to phrase this question. Is it ok
to establish a connection for a source & destination address combination for
which a previously connection socket is in CLOSE_WAIT state??
I will also attach snap shot of my code.

```
Client()
{
  if((sfd = socket(AF_INET,SOCK_STREAM,IPPROTO_TCP)) < 0) {
     printf("Error errno %d\n,errno);
     exit(0);
  }
  gPollfd.fd = sfd;
  gPollfd.event = POLLIN;
  while(1) {
    switch(poll(&gPollfd,1,-1)) {
```

```
      case 0:
      case -1:
        break;
      default:
        /* Poll success case */
        if(gPollfd.event & POLLIN) {

    }
  }


}
```

From: Mahesh Kallanagoudar

Hi Rob,

I even tried your suggestion of using system assigned port.
(Zero the port number in the socket call()) It didn't help.
What surprises me is if I see netstat it shows the conenction
as ESTABLISHED. But I still get the broken pipe error for write.
Before actually re-establishing the connection, I did netstat.
It showed the connection in CLOSE_WAIT state. (On the active close side)
It might sound crazy to you. But I am facing this problem.
I cann't suspect the code because it works fine, If I wait for
3-4 mins (By having a sleep call) before re-establishing the
connection. I don't know how to phrase this question. Is it ok
to establish a connection for a source & destination address combination for
which a previously connection socket is in CLOSE_WAIT state??
I will also attach snap shot of my code.

```
Client()
{
  if((sfd = socket(AF_INET,SOCK_STREAM,IPPROTO_TCP)) < 0) {
     printf("Error errno %d\n,errno);
     exit(0);
  }
  gPollfd.fd = sfd;
  gPollfd.event = POLLIN;
  while(1) {
    switch(poll(&gPollfd,1,-1)) {
      case 0:
      case -1:
        break;
```

```
    default:
      /* Poll success case */
      if(gPollfd.event & POLLIN) {

   }
  }


}
```

From: Mahesh Kallanagoudar

Hi Rob,

I even tried your suggestion of using system assigned port.
(Zero the port number in the bind() call) It didn't help.
What surprises me is if I see netstat it shows the conenction
as ESTABLISHED. But I still get the broken pipe error for write.
Before actually re-establishing the connection, I did netstat.
It showed the connection in CLOSE_WAIT state. (On the active close side)
It might sound crazy to you. But I am facing this problem.
I cann't suspect the code because it works fine, If I wait for
3-4 mins (By having a sleep call) before re-establishing the
connection. I don't know how to phrase this question. Is it ok
to establish a connection for a source & destination address combination for
which a previously connection socket is in CLOSE_WAIT state??
I will also attach snap shot of my Client code. The server code is typical TCP
Server code.
NOTE: I just wrote the code to give an idea of what I am trying to do. If it has
some obvious errors neglect it. The actual code is different.

```
Client()
{
 if((sfd = socket(AF_INET,SOCK_STREAM,IPPROTO_TCP)) < 0) {
    printf("Error errno %d\n,errno);
    exit(0);
 }
 memset((void *)&locAddr, 0, sizeof(struct sockaddr_in));
   locAddr.sin_family = AF_INET;

   locAddr.sin_addr.s_addr = htonl(INADDR_ANY);
   locAddr.sin_port = htons(0);
    if (bind(sfd, (struct sockaddr *)&locAddr, sizeof(locAddr)) < 0)
    {
          printf("sockbind failed errno 0x%x\n",errno);
```

```
        close(sfd);
        return -1;
    }


 /* Make socket NonBlocking using fcntl call */
 fcntl();
 gPollfd.fd = sfd;
 gPollfd.event = POLLIN;
 while(1) {
  switch(poll(&gPollfd,1,-1)) {
    case 0:
    case -1:
      break;
    default:
      /* Poll success case */
      if(gPollfd.event & POLLIN) {
        while(recv(sfd,buf,sizeof(buf),0) > 0) {
          /*
           * send back the data to the Server
           */
           send();
           /* This send returns Broken Pipe error after re-establishment */
        }
        if(errno == ECONNRESET) {
          /* The other end has closed the connection */
          close(sfd);
          /* Try to re-establish the connection */
          while(connect(sfd,&srvAddr,sizeof(srvAddr) < 0);
        }
      }
 }

}
```

regards,
mahesh

From: Rob Seace

Well, if it's really in a CLOSE_WAIT state, then that means
that end of the connection (I'm not clear on whether it's
the client end or server you're seeing the CLOSE_WAIT on)
has already done a close() (or, a shutdown())... Then, the
fact you get an EPIPE error later trying to write to the

closed socket makes sense... You might want to just go with inserting some debugging code into your programs, to spit out exactly what they are doing, at each step; that may help your track things down...

I know you said the above code isn't real, and to ignore errors, but I have to ask about the bit where you do the re-connect... I'll assume you've got another connect() up near the top, after your bind(), which you just don't show... But, do you also have another socket() call, and another bind(), prior to the connect() you do show, for reconnecting after a broken connection? (Actually, you really don't even need the bind() there, for a client... The connect() will do your bind() for you... Best to just let it do so... But, you DO need to recreate the socket with another call to socket(), after you close it...) Also, you probably don't want to JUST do it only on ECONNRESET, either... What do you do for other errno values? It might help to see more realistic code, because I don't know if these are actual problems, or just problems in the sample bogus code...

From: Hector Lasso

Hi,

I see something odd in here:
When you check for disconnection you do a close(fd); and right after it you try to connect again. This shouldn't work because your file descriptor is no longer valid. connect should in this case return always -1 with errno=EBADF.
I suggest you check this, change the code to create a new socket and once again connect.

I hope this helps
Good luck

From: Mahesh Kallanagoudar

Hi,
The actual code is bit hard to follow, So I provided the sample code.

Rob, following are list of things you have asked for...

1. I am doing a socket and bind after close()
and before reconnecting.
2. I am connecting before the while(1) loop.

3. But I am handling only ECONNRESET error. What are the other possible error values for other end closing connection?
I am registering for SIGPIPE signal. But I am not getting that signal once the other end is restarted. I don't know why?

Here is the updated sample code.

```
Client()
{

  if((sfd = socket(AF_INET,SOCK_STREAM,IPPROTO_TCP)) < 0) {
     printf("Error errno %d\n,errno);
     exit(0);
  }
    memset((void *)&locAddr, 0, sizeof(struct sockaddr_in));
    locAddr.sin_family = AF_INET;

    locAddr.sin_addr.s_addr = htonl(INADDR_ANY);
    locAddr.sin_port = htons(0);
     if (bind(sfd, (struct sockaddr *)&locAddr, sizeof(locAddr)) < 0)
     {
           printf("sockbind failed errno 0x%x\n",errno);
           close(sfd);
           return -1;
      }


  /* Make socket NonBlocking using fcntl call */
  fcntl();
  gPollfd.fd = sfd;
  gPollfd.event = POLLIN;
  while(connect(sfd,&srvAddr,sizeof(srvAddr) < 0);
  while(1) {
    switch(poll(&gPollfd,1,-1)) {
      case 0:
      case -1:
        break;
      default:
        /* Poll success case */
        if(gPollfd.event & POLLIN) {
          while(recv(sfd,buf,sizeof(buf),0) > 0) {
            /*
             * send back the data to the Server
             */
```

```
          send();
          /* This send returns Broken Pipe error after re-establishment */
        }
        if(errno == ECONNRESET) {
         /* The other end has closed the connection */
         close(sfd);
         if((sfd = socket(AF_INET,SOCK_STREAM,IPPROTO_TCP)) < 0) {
          printf("Error errno %d\n,errno);
          exit(0);
         }

        if (bind(sfd, (struct sockaddr *)&locAddr, sizeof(locAddr)) < 0)
        {
         printf("sockbind failed errno 0x%x\n",errno);
         close(sfd);
         return -1;
        }

         /* Try to re-establish the connection */
         while(connect(sfd,&srvAddr,sizeof(srvAddr) < 0);
        }
       }
  }

}
```

regards,
mahesh

From: Mahesh Kallanagoudar

Hi,
The actual code is bit hard to follow, So I provided the
sample code. I am not doing the obvious mistakes in the
actual code.

following are list of things you have asked for...

1. I am doing a socket and bind after close()
and before reconnecting.
2. I am connecting before the while(1) loop.
3. But I am handling only ECONNRESET and SIGPIPE (not shown in the

code) errors. What are the other
possible error values for other end closing connection?
I am registering for SIGPIPE signal. But I am not getting
that signal once the other end is restarted. I don't know why?

Here is the updated sample code.

```
Client()
{
  signal(SIGPIPE,reconnect);
 /* reconnect() functions does exactly same thing what is
  * shown currently when ECONNRESET error happens. close(),
  * socket(), bind and connect().
  */
  if((sfd = socket(AF_INET,SOCK_STREAM,IPPROTO_TCP)) < 0) {
     printf("Error errno %d\n,errno);
     exit(0);
  }
    memset((void *)&locAddr, 0, sizeof(struct sockaddr_in));
    locAddr.sin_family = AF_INET;

    locAddr.sin_addr.s_addr = htonl(INADDR_ANY);
    locAddr.sin_port = htons(0);
     if (bind(sfd, (struct sockaddr *)&locAddr, sizeof(locAddr)) < 0)
     {
           printf("sockbind failed errno 0x%x\n",errno);
           close(sfd);
           return -1;
      }


  /* Make socket NonBlocking using fcntl call */
  fcntl();
  gPollfd.fd = sfd;
  gPollfd.event = POLLIN;
  while(connect(sfd,&srvAddr,sizeof(srvAddr) < 0);
  while(1) {
    switch(poll(&gPollfd,1,-1)) {
      case 0:
      case -1:
        break;
      default:
        /* Poll success case */
        if(gPollfd.event & POLLIN) {
```

```
        while(recv(sfd,buf,sizeof(buf),0) > 0) {
          /*
           * send back the data to the Server
           */
           send();
           /* This send returns Broken Pipe error after re-establishment */
        }
        if(errno == ECONNRESET) {
          /* The other end has closed the connection */
          close(sfd);
          if((sfd = socket(AF_INET,SOCK_STREAM,IPPROTO_TCP)) < 0) {
           printf("Error errno %d\n,errno);
           exit(0);
          }

         if (bind(sfd, (struct sockaddr *)&locAddr, sizeof(locAddr)) < 0)
         {
          printf("sockbind failed errno 0x%x\n",errno);
          close(sfd);
          return -1;
        }

          /* Try to re-establish the connection */
          while(connect(sfd,&srvAddr,sizeof(srvAddr) < 0);
        }
      }
  }

}
```

Hope the new code clears all your doubts.
regards,
mahesh

From: Mahesh Kallanagoudar

Sorry I accidently added incomplete comments. Take the last one.

From: Hector Lasso

Hi,

I guess you are zeroing the sin_port element of locAddrright before calling bind() when

re-establishing the connection. If you don't do it then it could be a problem and this is where Rob tells you to use 0.

However, bind() is not necessary (as Rob pointed early) because connect does this for you. I suggest you get rid of all the bind() calls and let connect() take care of it.

If that doesn't work then i think it could be an OS bug. Check your OS' homepage to find out about any documented bug related to the problem you've been facing.

From: Rob Seace

Yes, I agree: get rid of the bind() calls completely... I
never bind() my client apps, only servers...

In addition, I see you say that you set the socket to
non-blocking mode... In that case, you realize that you
have to be prepared for any and all I/O calls to fail
immediately with EAGAIN or EWOULDBLOCK, right? (And, that
would include your poll() call, as well, I believe...)

As for other errnos to check for in the re-connection case,
one would be EPIPE... But, what if it's just some other
random error? What do you do in that case? Why not just
ALWAYS try the re-connection, on failure, regardless of the
errno value? *shrug*

From: Mahesh Kallanagoudar

I tried removing all the binds on the Client side. But it didn't
help. As I explained earlier I am registering for SIGPIPE signal
. But I am not getting that SIGNAL whenever there is a break
in the TCP connection. Is it because socket is non-blocking.
I shall include EPIPE error also. I didn't get what do mean by
reconnecting when there is a random error. what about
errors which are not related Connection? Since we have only
one errno, it could be set because of someother unrelated
errors.

regards,
mahesh

regards,
mahesh

From: Mahesh Kallanagoudar

I tried removing all the binds on the Client side. But it didn't

help. As I explained earlier I am registering for SIGPIPE signal
. But I am not getting that SIGNAL whenever there is a break
in the TCP connection. Is it because socket is non-blocking.
I shall include EPIPE error also. I didn't get what do mean by
reconnecting when there is a random error. what about
errors which are not related Connection? Since we have only
one errno, it could be set because of someother unrelated
errors.

regards,
mahesh

regards,
mahesh

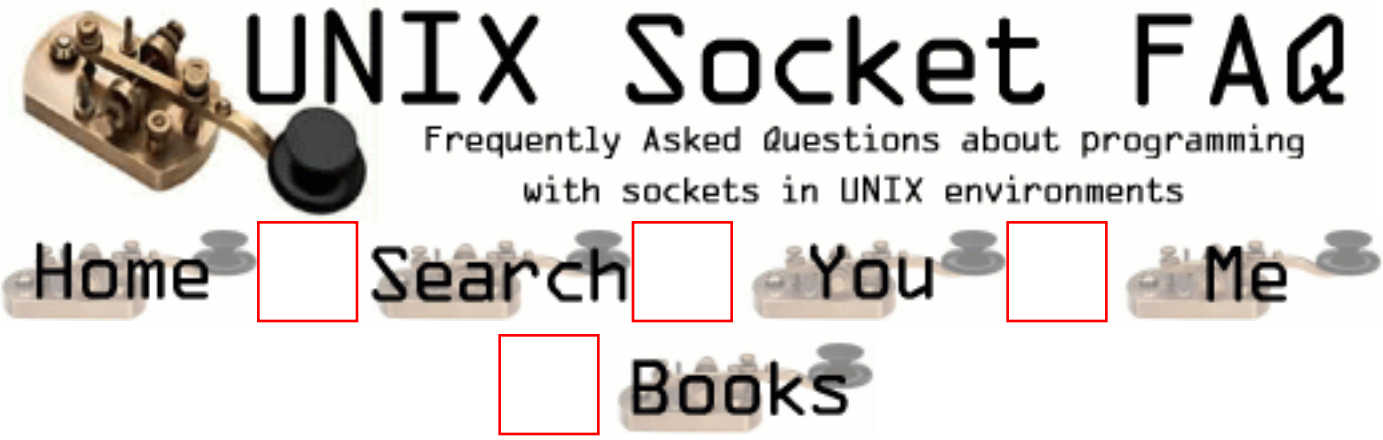
From: Rob Seace

SIGPIPE is generated whenever you attempt to write to a
socket which has been closed by the other end... Your
original question claimed that you WERE getting a "broken
pipe error", which sounds to me like you were indeed getting
the SIGPIPE... Or else you're just getting EPIPE in errno,
after a write failure... (That would happen if you ignored
SIGPIPE, which is what I always do anyway, since it's a lot
easier to deal with a simple write failure...) So, I don't
know what you mean now by saying that you're NOT getting the
signal... If you're not, then what is this "broken pipe
error", you're talking about? Just EPIPE in errno, after a
failed write()/send()? If that's the case, I'd say you must
somehow be set to ignore the signal, rather than trap it...

And, what I meant about reconnecting was, what do you do in
the case that send()/recv()/poll(), or any other I/O call,
fails, but with the errno value set to something other than
the limited set you are checking for? Do you exit, or what?
Because, it doesn't make sense to just blindly continue on
in that case... If you got some I/O failure, that probably
means your socket is now unusable, and your only options
really are to close() it and recreate it, or just exit...
Note: since you're in non-blocking mode, there are of course
exceptions to this: on EAGAIN or EWOULDBLOCK (usually, one
and the same thing), that just means the I/O op can't be
completed at the moment without blocking, which is perfectly
normal, expected behavior from a non-blocking socket...
Also, EINTR can always be returned if the call was

interrupted while in progress... But, I think any other
errno value would mean your socket was now unusable, and
should be thrown away (and, possibly, recreated)...

Add a Comment

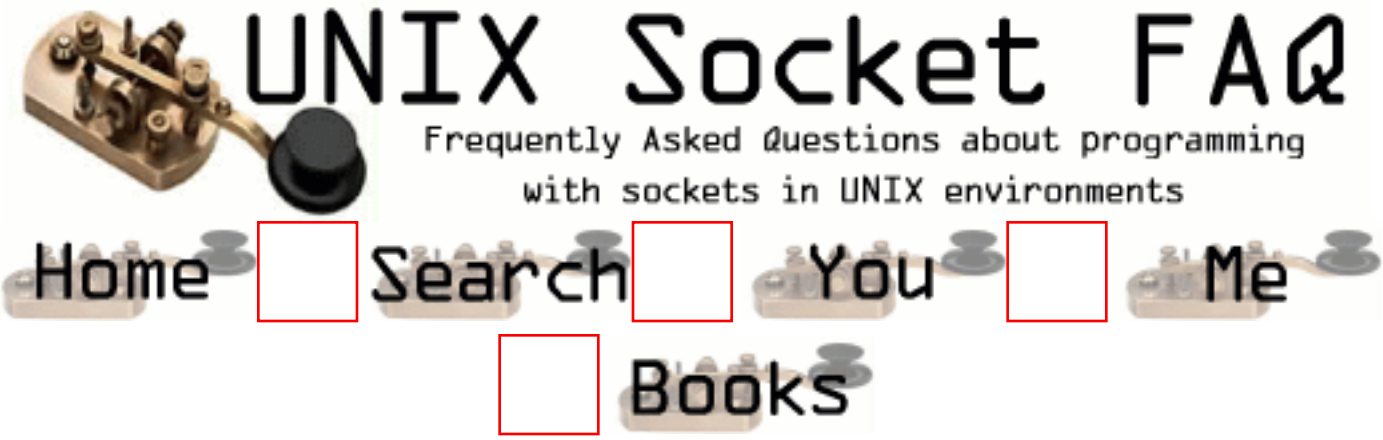


New Questions: single program acting as server and client

From: bhavani

Iam doing virtual router project in which I have a single program acting as server and client. The programs create server and client sockets, that should contact each other:
I know that one way is to do select but Iam confused with how to use it . If anyone can help with this. I would be really thankful.

Add a Comment



New Questions: How can a server let other server connect with a client ?

From: Jack

When a server receive a connection request, how can it let other server connect with this client ?

From: Loco

Be more specific. Give examples. I am not very smart so i cannot understand your question clearly...

From: Jack

I create a socket, set option, bind with a port, and so on.
Then I use select() to watch the event.

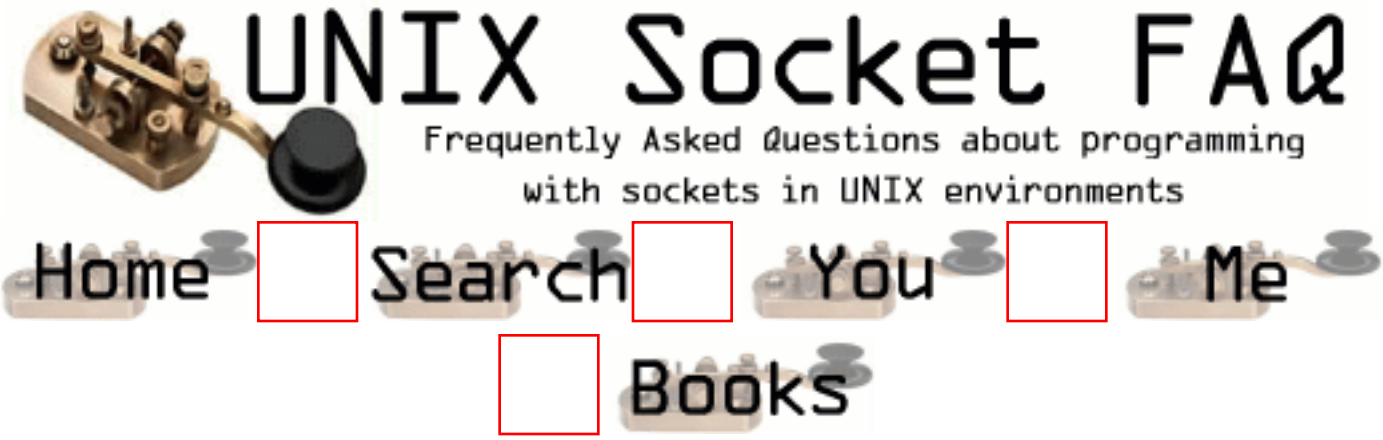
When a connect request was happened, I create a NEW socket, bind with OTHER PORT, and I want the NEW socket connect with this client, how ?

From: Loco

I still don't get it.

Your server listens for connections. When one client connects you already have a connected socket. But you want another one that connects from the server to the client, don't you?
If the client is not listening then you cannot do it, you will have to bind a port in the client and listen for connections. The server can take the address of the client from the sockaddr_in struct filled by accept()

Add a Comment



New Questions: CGI problems.

From: Sandra

Hi everyone. I'm a newbie and I'm writing a simple program
in C that is a HTTP client. It's working very well but my
problem is that I want this program to act as a CGI and it
simply does not connect to the server. I suspect that the
function htons(PORT_NUM) is causing me troubles but without
this function my program can't connect to the server.
Does anyone knows what's wrong? Thanks!!

From: Loco

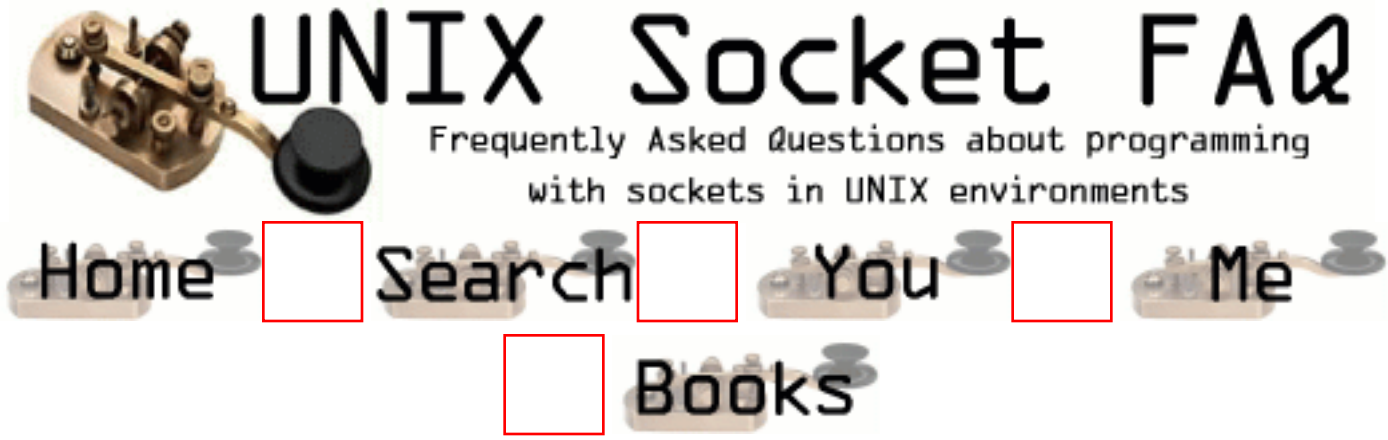
Hi Sandra,

I guess your problem is that CGI's normally are run at the server side not the client side.

If you write an HTTP client then you are writing a piece of software that connects to a server and asks for services, in this case an HTTP server.

CGIs are programs that are executed by the HTTP server to respond to certain requests from the clients. The communication between server and CGI is done using a simple interface, so you can write a CGI in almost any programming language supported by the particular OS you run the HTTP server on. The decission to execute the program is taken by the HTTP server which normally uses a configuration file and MIME types to take this decission.

Add a Comment



New Questions: which is best Fork () or pthread ?

From: Baswaraj

I have client/server model having..

server is accepting request from clients on single port.

first request comes.
then data (if any) comes.
---
if i have two threads one for accepting request ,& another
for processing request.

problem with this approch is that requesting thread only accepts
request (not data), i need to recive data in proceesing thread .
client side is already connected to requesting thread. for
sending data related to request it has to connect to
processing thread. but there r too many client sending request
& data to requesting thread & procesing thread.

there must be one to one corrospondance btwn request & data.
this ONE TO ONE correspondance can't be achived throuh this approch.

PLEASE SUGGEST ME SUITABLE SOLUTION.

---

i planed to solved this using fork ().

i want to limit the no of process to some MAX value.

HOW it can be done ?

PLEASE HELP ME.

THANKS IN ADVANCE

Baswaraj

From: Chris Gilliard

You may want to look into using the select() function. It does what you want to do without having multiple processes/threads at the same time. The idea is that you can check multiple file descriptors (in this case sockets) and the select call will block until there is something to be read on one of the descriptors. This lets your program block until someone sends a message over the socket. At that point you can handle it. There is a similar system call called poll. It may be more efficent than select if you have a bunch of other file descriptors open which most likely not the case, but read the man pages for more details on this. If you still want to use a forking or threaded server, I'd perfer threads. Good luck!

From: Hector Lasso

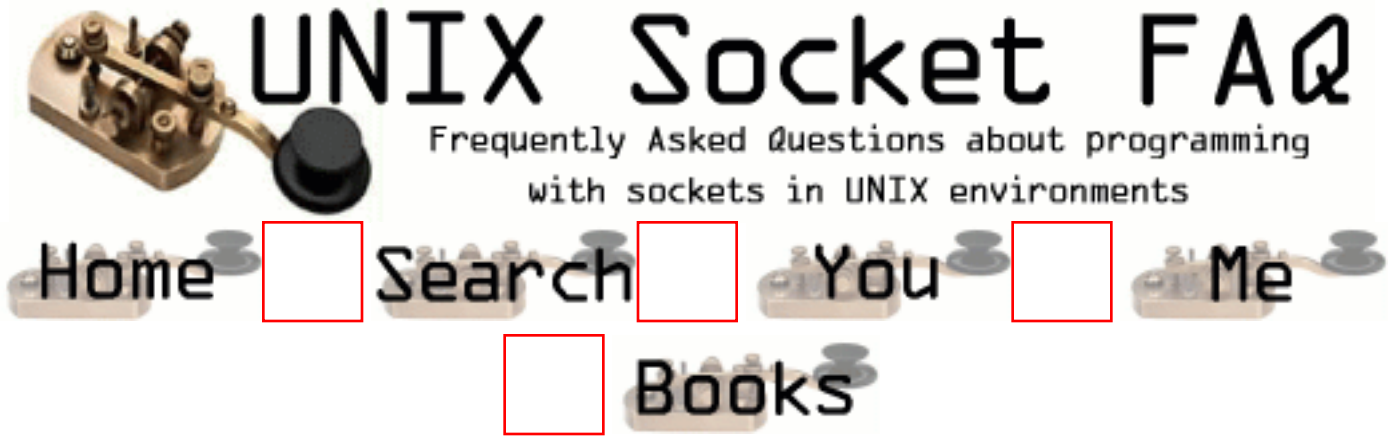
Hi,

I would use a structure to pass data from one thread to the other.
When the "receiver" thread gets a request it just puts the request in the structure, fires an event and continues waiting for data from the clients. When the client sends the data, then the "receiver" thread puts the data in the same structure and once again fires the event.
The "processor" thread will wait indefinetely for the event, and when awaken, will read the data in the structure and process the request, however, as the request and data for it don't come in the same message, it will have to decide whether it can go on with that or just go back to wait for another event. It's a classic producer-consumer approach.

Regarding your question on limiting the number of children forked:
I don't know a way to know the number of children there are alive, however, it may exist. My code would increment a counter each time a chidl process is forked, and i'd trap the SIGCHLD signal to know when they die and decrement the counter. This way i'd just compare the counter against the maximum value and then decide whether to fork() again or not.
Another way would be to control the number of processes using setrlimit(), however, it would mean you'd get an error when trying to fork().

It may or may not be the best way to do it, however, this is sockets FAQ and your question is not related to sockets.

Add a Comment



New Questions: Timestamp

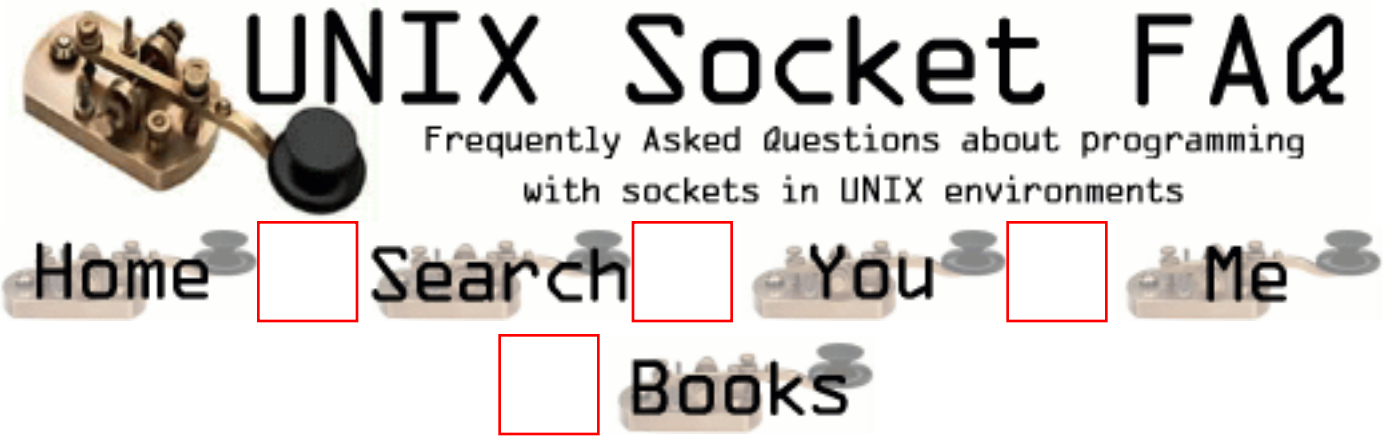
From: Andreas

I'm multicasting through the Internet. My question is: I want to create a buffer at the receiver side. Since I am sending audio, the buffer should sort the packets. I need to include a timestamp in the UDP-packets. Anyone have an example on how to do that?

From: Hector Lasso

I would use something like:
sprintf(buffer, "%10d", time(NULL));
and then append the data to "buffer"

Add a Comment

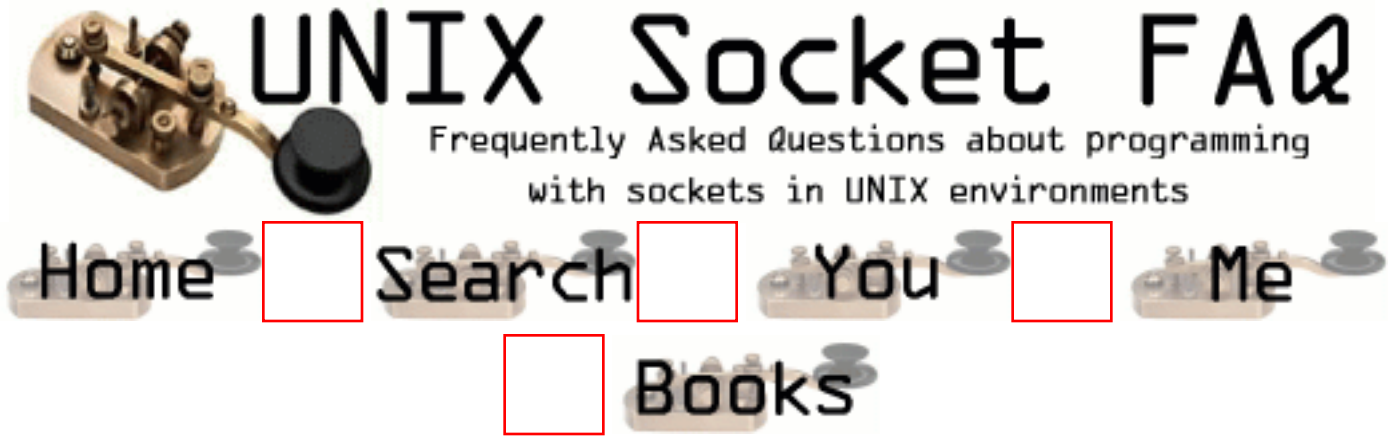


New Questions: Difference between port & socket

From: Stefanie Stubb

Can anyone tell what is the difference between port & socket?

Add a Comment

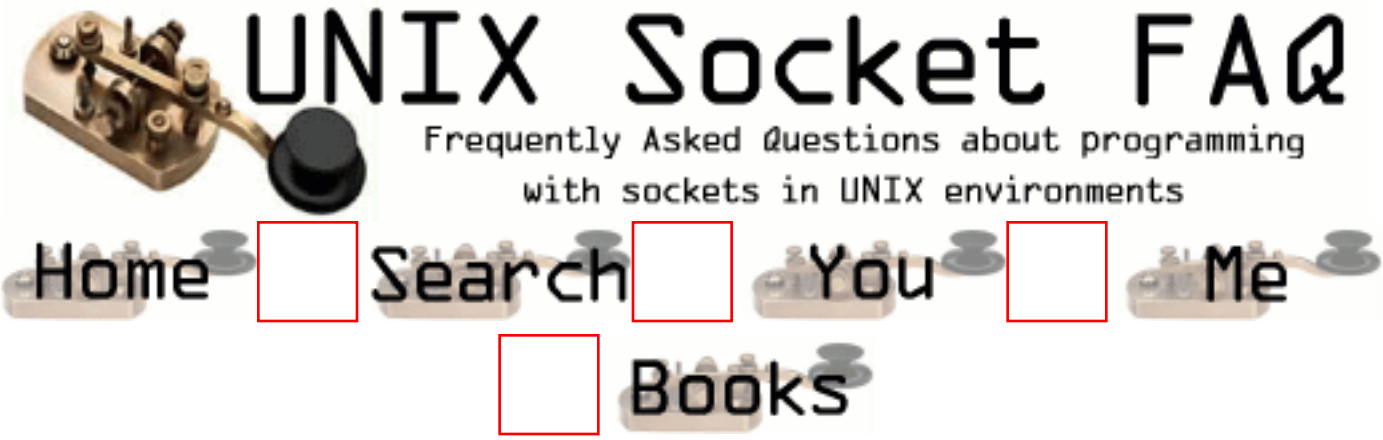


New Questions: Client write to closed server

From: Fabrics

I have the following problem: my client tries to write to a server that has closed its socket, the first time the client receives no exception in its operation so my message is lost, while the second time it gets an exception.
How can I understand the fist time that the socket on server side is closed?
Thanks to everyone who will help me.
P.S.: I'm using Java.

Add a Comment

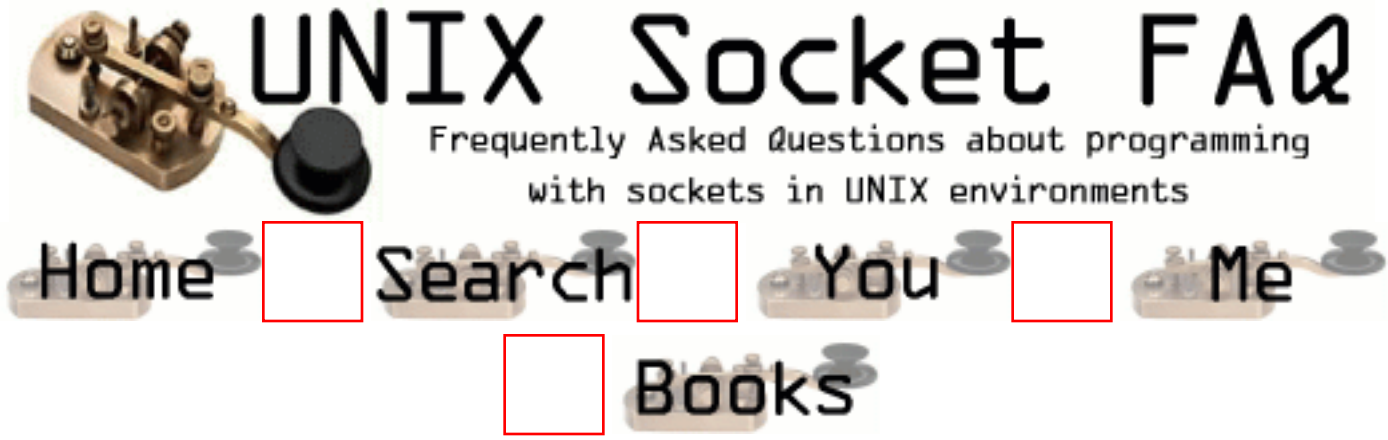


New Questions: How to flush a udp socket?

From: Itai Ashlagi

In a client/server, the client sends all the time udp packets to the server.
In the server I have sometimtes timeouts , so packets are accumalting in the
socket buffer.
After the timeout I want to clean the buffer and start recieving again.
How? (except reopening the socket).

Add a Comment



New Questions: Binding of RAW Sockets

From: Veeru

Hi,
Is it necessary to bind a RAW Socket with the local address.
I need the exact information about this.i need the information urgently.

Regards,
Veeru

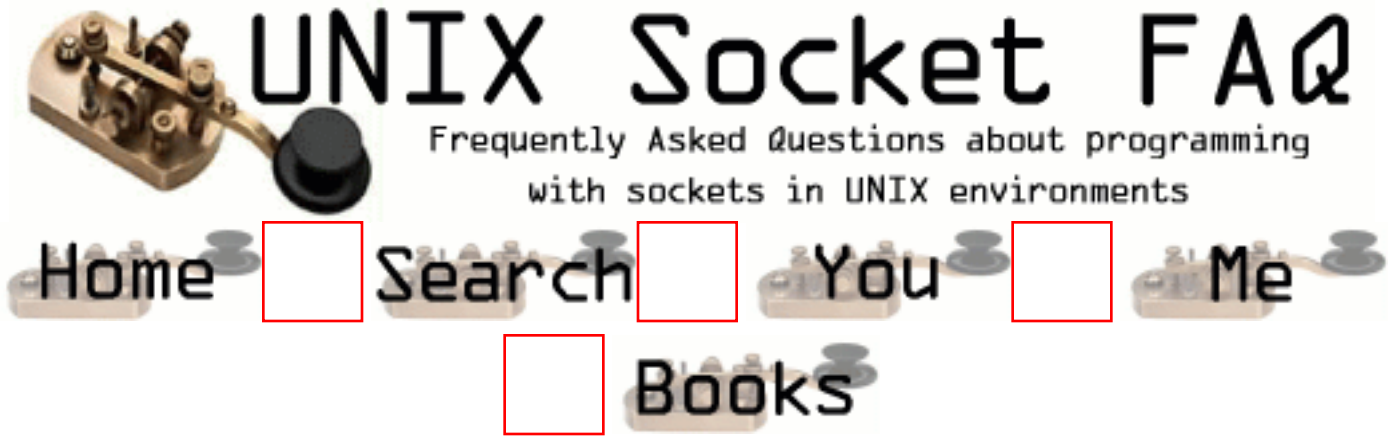
From: Loco

Take a look at the man page of raw(7)
"[...]A raw socket can be bound to a specific local address using the bind(2) call. If it isn't bound all packets with the specified IP protocol are received. In addition a RAW socket can be bound to a specific network device using SO_BINDTODEVICE; see socket(7) [...]"
:D (HAL)

From: Evil Homer

Where can I find that page you refer to?

Add a Comment



New Questions: Server does not receive message sent by Client

From: Vijay

Hello,

Can anybody help me please.

I have a Client/Server application. The Client establishes a TCP Connection with the Server and sends a message to the Server. But the server does not receive the packet until the socket connection is closed or the Client terminates (effectively closing the socket).
Can anyone tell me why this is happening and how to over come this.

Thanks,

Vijay.

From: Fabrics

It is possible that you don't do the flush on your socket and so it is automatically flushed only before closing it and the server receive it in that moment.

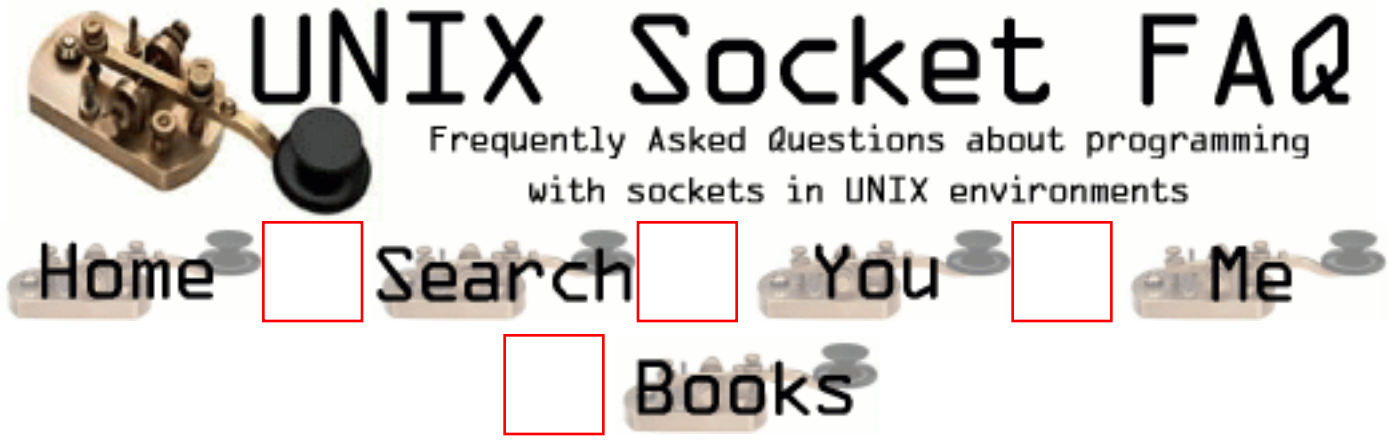
From: frankv

You may be trying to recv more bytes then are sent (and available) on the socket. In tcp/ip SOCK_STREAM sockets there are no boundaries between messages, so
br = recv(fd, buf, 1024, flags);
will generally wait (block) until there are 1024 bytes available, because it assumes you *know* there are 1024 bytes waiting.

Conversely the opposite may happen also. You may recv less then what you asked for even though there should be more there. I am not sure if this actually happens though.

There are basically two ways to send messages with boundaries on SOCK_STREAM sockets.
1 - Always send the same size messages. This way you can safely specify the number of bytes in recv().
2 - Send a header before the body which indicates how big the body is. That way you can read the (fixed length) header first, followed by the body.

Add a Comment



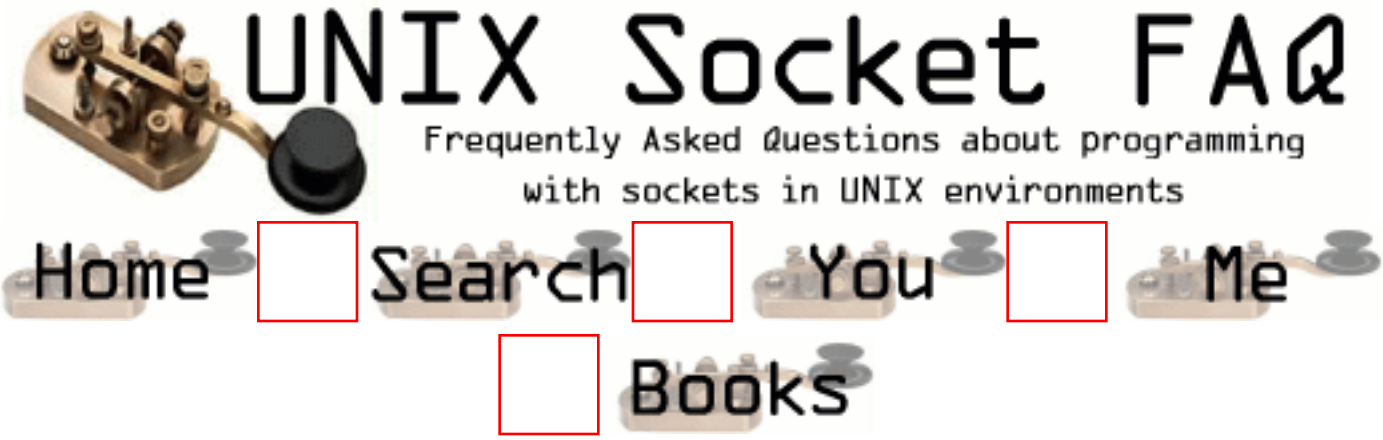
New Questions: problems with sending files

From: Joakim Fredriksson

Hi, I have written a program that sends binary files using tcp/ip in windows.
The problem I have is that packets are lost, I have a while loop that reads in 4096 bytes at the time from the file then send them on the socket without any sleep or slowdown, so I suppose that I send to fast and that therefore packets are lost, but my question is how can I get the OS/network to tell me if I`m sending to fast so I can slow down, its very annoying. Sure I can put in a sleep or something but then the max speed I get is about 300kb/s without the sleep I`m sending at >800kb/s

---

Add a Comment



New Questions: Difference between socket and port

From: Stefanie Stubb

Can anyone tell what is the difference between those two?

From: Thorsten Weigmann

Hi,
a socket is an endpoint for network communication.
In simple, it consists of a ip-address to distinguish a host
and a port to distinguish the different communication channel on this host.

Thorsten

Add a Comment

New Questions: Forwarding data from one interface to another

From: Jad

Can any body help me with this:

I have two ethernet cards , basically i have these two interfaces . I want to read a IP datagram from one interface ,change the target IP address, and then output the datagram on the second interface.The reverse should work also ie changes the source IP from interface 2 and outputs it to interface 1.

The problem I am facing is how to make this work both ways without discarding received IP datagrams.

An help or code ? Thanks

From: Rob Seace

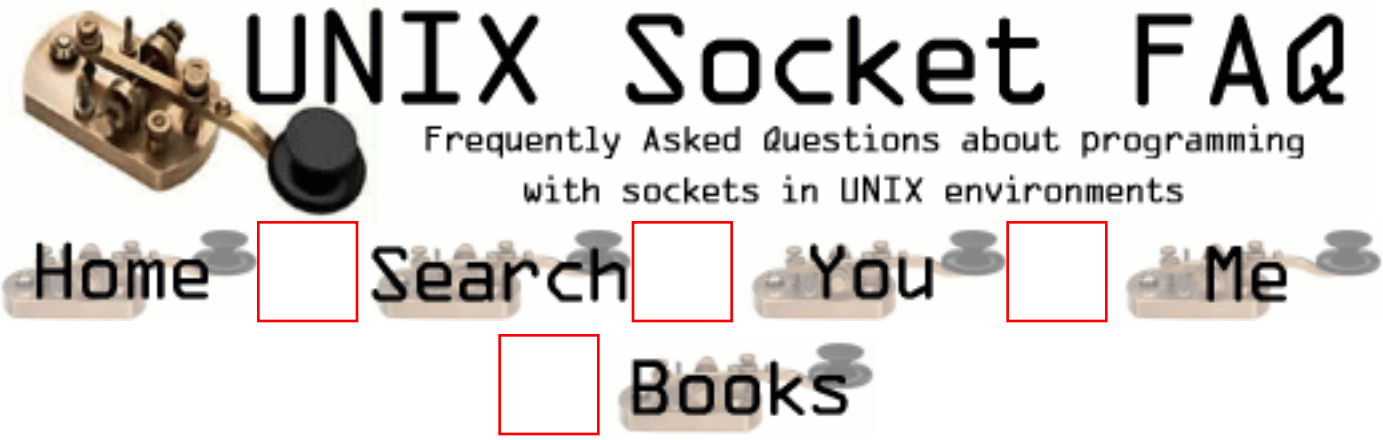
Interesting... It sounds like you basically want to create a totally user-space router; or, really, a user-space NAT implementation... Yes? I'm just not sure if you can really pull that off without your OS getting involved in the actual routing/NAT'ing...

Now, you could certainly easily do something similar, if you were willing to limit it to ONLY forwarding certain TCP/UDP ports, by just creating a daemon that basically acts as a tunnel: listening on those ports locally, and relaying them off elsewhere... Certainly not as sexy as a totally transparent user-space router/NAT, which can capture and relay ALL IP traffic, though... But, without a listening local app, or direct kernel support for the routing and NAT

trickery, I don't think you can pull it off... *shrug*

(And, BTW, if you've got a Linux box, you can just do this sort of thing with built-in kernel IP-masquerading, without the need to write your own app at all, anyway... ;-))

Add a Comment

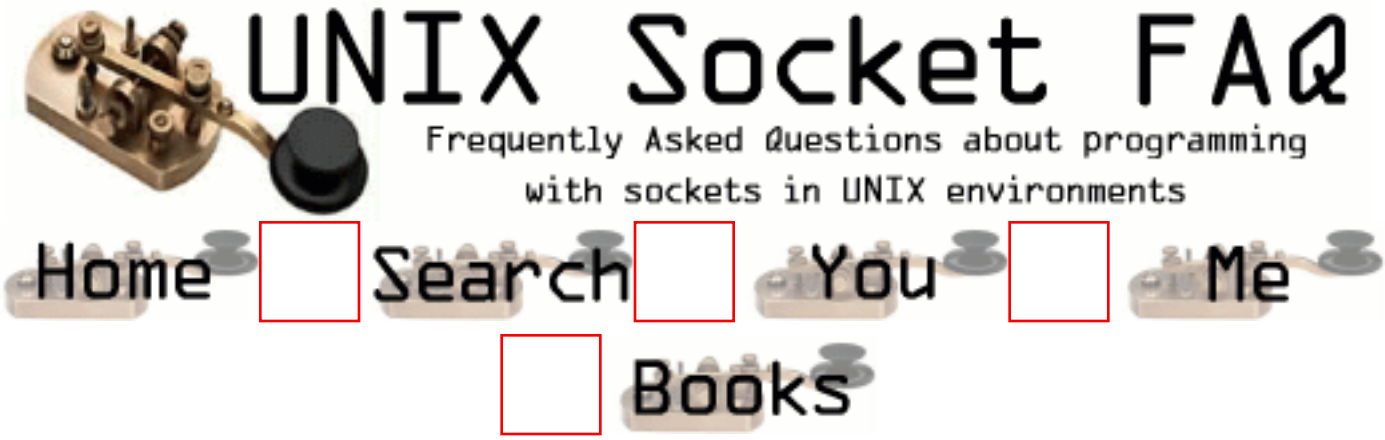


New Questions: problem with error "unresolved text symbol" in UDP program using recvfrom & sendto

From: Shivani

My client-server program involves intra-machine invocation.
It uses UDP to send and receive messages. But whenever I compile the client program, it gives two errors -
(1)Unresolved text symbol "recvfrom(int,char*,int,int,sockaddr*,int *)" -- 1st referenced by q.o.
(2) Unresolved text symbol "sendto(int,char*,int,int,sockaddr*,int)" -- 1st referenced by q.o.
Could anyone please explain what these mean and how I can solve them?

Add a Comment



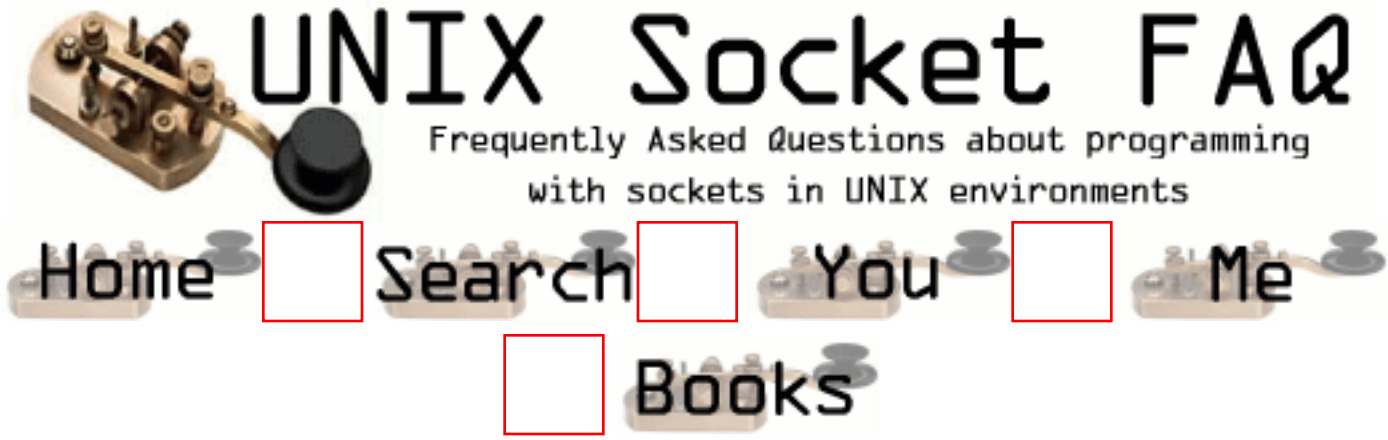
New Questions: API to get state of socket?

From: Howard Lander

Is there a way to find the current state of a socket using an
socket API call? For instance I'd like to know if a socket is
in CLOSE_WAIT state. I looked at the code for the LINUX netstat
program, but it reads the info out of a kernel file. Since
my server could be servicing serveral thousand connections
I don't want to do it this way.

Thanks
Howard

Add a Comment



New Questions: Looking for advice on non-blocking sockets

From: Loren Frank

Hi. I'm writing a data acquisition application that uses sockets but I'm running into problems when I try to send large amounts of data from one program to another. I am sending messages among the programs that make up the application and these messages contain either short commands (a few bytes) or data buffers that can be fairly large (~400K).

Right now, I have everything set up to be non-blocking so that the programs can run independently, checking for a message every few hundred microseconds to see if there is a new message or new data. Everything works when the data messages are less than the socket buffer size, but as soon as the messages get larger, the "write" that writes the message ends up writing only a single socket buffer worth of information. I understand why this happens (the write fills the buffer and then returns) but I don't know exactly how I should fix it.

So ... any advice on how I should write the messaging so that I can do transfers that are larger than the socket buffer size?

Thanks,

Loren Frank

From: Rob Seace

Check out the code for the sock_write() function in the example code for this very FAQ...

You should always code your read()/write() calls similarly: wrap them in a loop to ensure everything gets read/written, since there's absolutely no guarantee a single call will do the whole job... Now, with non-blocking sockets, you're likely going to have to wait for a bit after you fill up the write buffer, until it clears out again so you can start writing; otherwise, all your write()'s will just immediately fail with EAGAIN/EWOULDBLOCK... You might need to

select() on it until it becomes writable again, or keep track of how much you've written so far, and just periodically try to write a bit more... *shrug*

From: Loren Frank

That makes sense. The only problem I would then have is that it is important that the process writing to the socket be able to write out the data and continue even if the reading process isn't ready. The idea is that the data collection subroutine could go on collecting data even if the processing routine that it talks to falls a bit behind. Is there any way that you know of to do interprocess communications where arbitrarily large amounts of data are buffered by the system, or do I have to write the buffering code as part of my program?

Loren

From: Rob Seace

You're talking about 2 processes on the same system? Then,
yeah, there are a certainly other, perhaps more suitable,
approaches... You might consider shared memory: your sender
just sticks the data in shared memory, all in one quick
move, then signals the reader somehow that the data is
ready, and the reader can then read it at its own leasure...

Or, if you really want to use the socket-based approach (I
try to use it myself, as much as possible, too), you might
try increasing your socket's write buffer size, via the
SO_SNDBUF socket option... (How high you can go with it
will probably be limited by your local system settings...)

From: Loren Frank

Got it. I'm working on a linux system and, as I understand things, the maximum socket size is set in the file /proc/sys/net/core/wmem_max. I can change that file to increase the size of the buffer, but I have been a bit leery of doing so. Do you know what possible problems I might run into? I'd like it to be a few megabytes, but I don't know what sort of side effects that might cause. The only thing I found on that topic was a website that suggested it could be increased to at least 8 MB (www-unix.gridforum.org/mail_archive/data-wg/pdf00000.pdf) but I'd like to be sure....

Loren

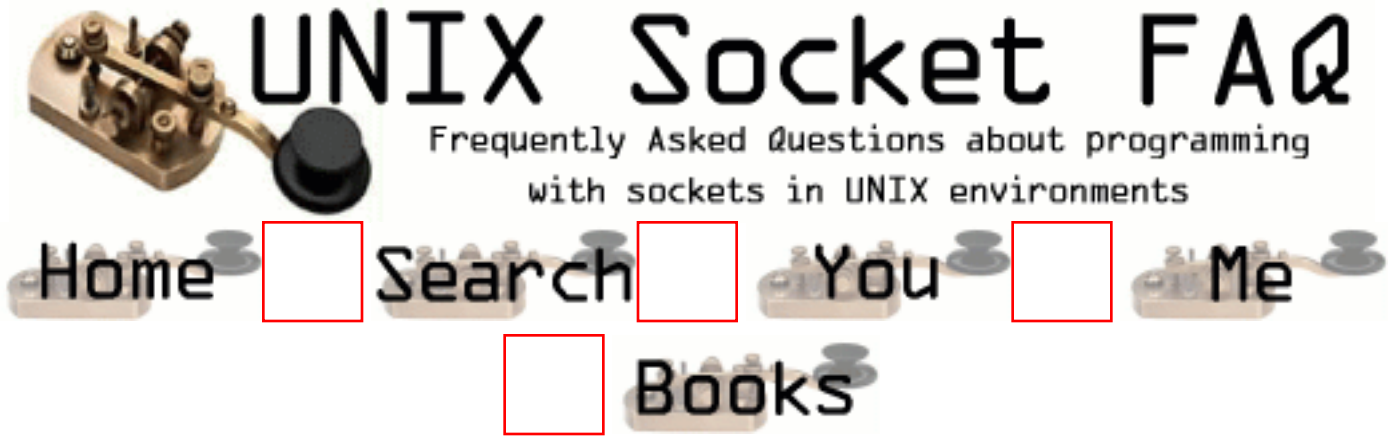
From: Rob Seace

Well, the "/proc" file you refer to only controls the upper
limit; you'd still need to have your app increase its
default size to that limit via SO_SNDBUG setsockopt()...
(Or, you could also change the "wmem_default" proc file, to
make all processes get that size buffer by default... But,

that's probably not a good idea...)

No, I've never done this, and am not sure what bad
side-effects it might have... But, I really wouldn't think
there should be any... The only real factor should be how
much free memory you have available on your system... If
you start trying to have huge buffers large than your free
memory, then things will quite probably get a bit screwy...
(It'll probably have to dip into swap space for some of the
buffer, and drag down performance with tons of swapping...
But, even then, nothing majorly horrible should happen...)
But, assuming you've got the memory, I don't know why you
shouldn't be able to have it as large as you like... *shrug*
I guess the best thing to do is just try it; it can't do too
much damage... ;-)

Add a Comment



New Questions: how to specify a rang of port numbers to be used

From: Ben

If my server is behind a firewall, one way to make outside client to connect to my server is allow a range of port opened. But since the port number is assigned randomly to the new socket from accept() call, how do I tell the OS assign the number in a range of values ?

thanks in advance

From: Loco

When you open a TCP connection (i guess you are using TCP sockets) to a port, the client uses any available port above port #1024 (unless you explicitly bind your socket), but the destination port remains the same, so to allow a connection to be made through your firewall you just have to open that particular port on the firewall accepting connections from any source port.

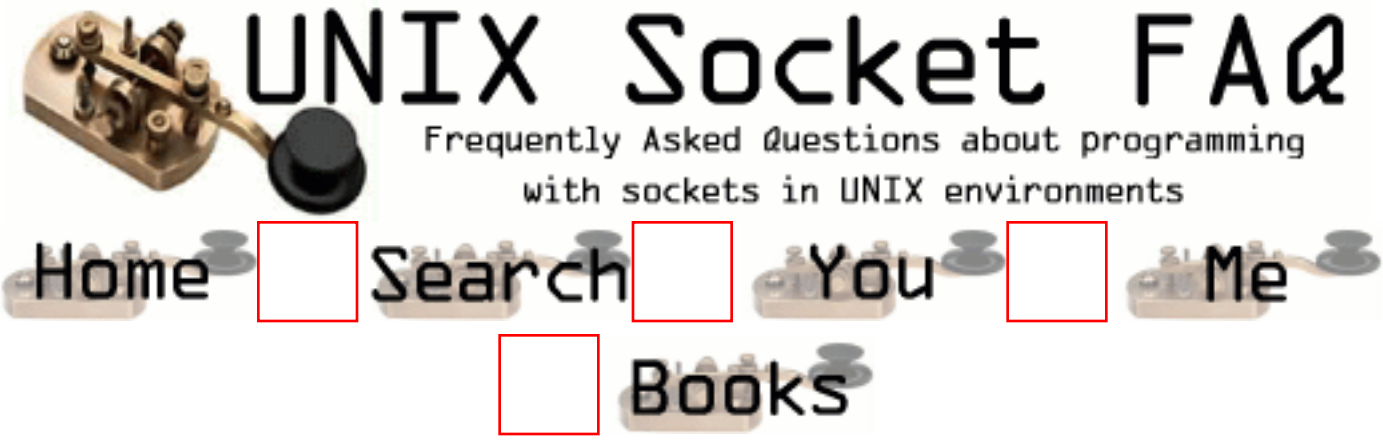
For example:
Your client is 192.168.1.100, your server is 192.168.12.5, you have a firewall separating both networks (192.168.1.0 and 192.168.12.0). You have a daemon waiting for connections on port 80 (HTTP port) and want allow the client to connect to this port then you just add a rule to your firewall like this:

SOURCE: 192.168.1.100 PORT ANY; DEST: 192.168.12.5 PORT 80; ACCEPT

The atual rule depends on your firewall, for example, in Linux using ipchains it would be:
ipchains -A input -p tcp -s 192.168.1.100 -d 192.168.12.5 80 -j ACCEPT

:)

Add a Comment



New Questions: Raw socket programming

From: Ramamoorthy

Hi,

I wants to capture the raw data packet. Can I use sockets (SOCKET_RAW option) for this purpose. Or can I use any other method? Please help me out with source code examples.

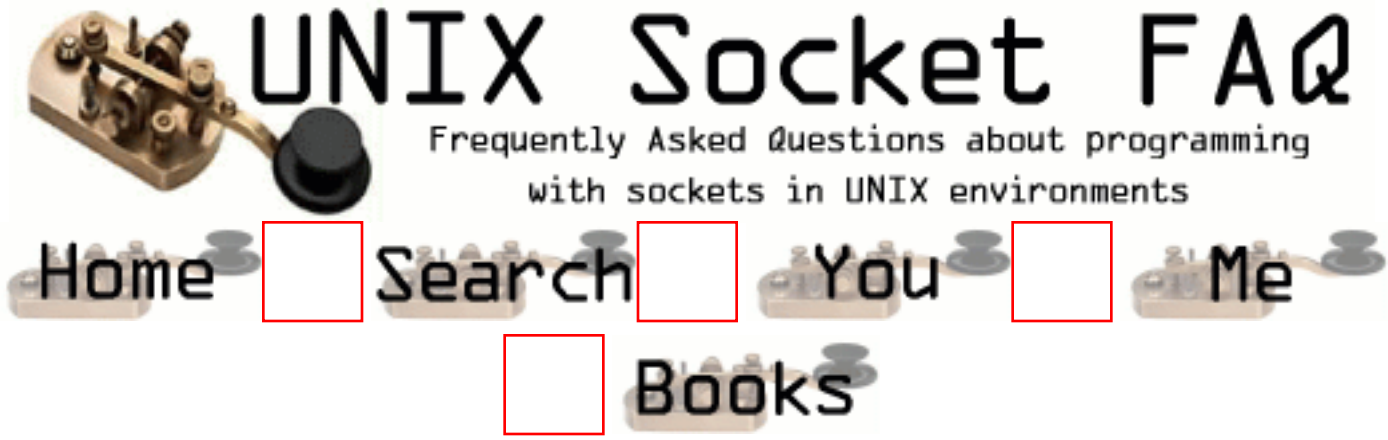
Thanks
Ramamoorthy

From:

Take a look at the man page of raw(7)
"[...]A raw socket can be bound to a specific local address using the bind(2) call. If it isn't bound all packets with the specified IP protocol are received. In addition a RAW socket can be bound to a specific network device using SO_BINDTODEVICE; see socket(7) [...]"

You gotta learn to look around, dude !!

Add a Comment



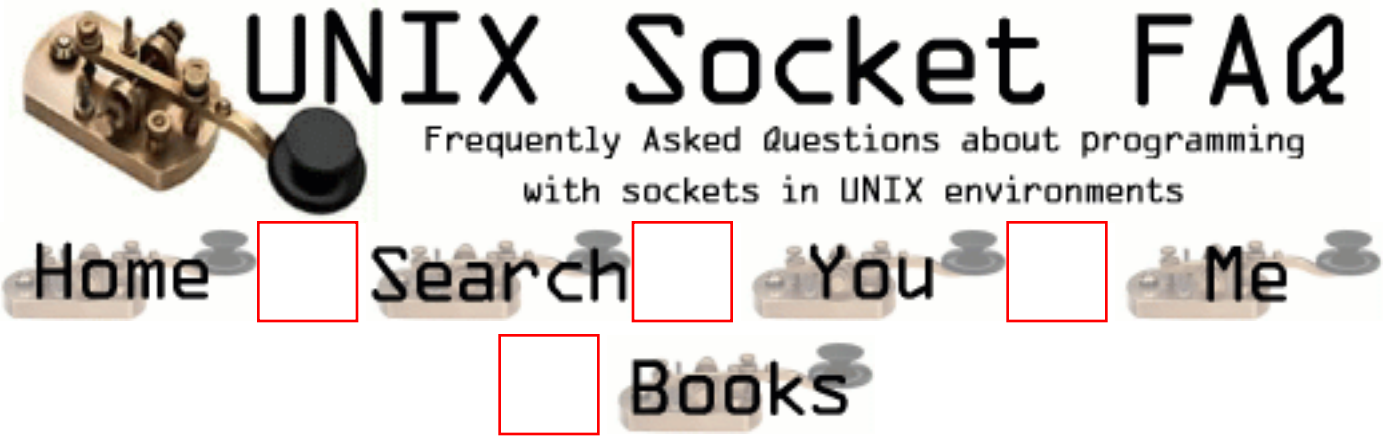
New Questions: IP-telephony

From: Anders And

In Unix you can use /dev/dsp to record and then send the buffer through a socket. My question is how to open(play) it in windows? Unix uses open("/dev/dsp"...) and then write(...).

Thanks in advance.

Add a Comment

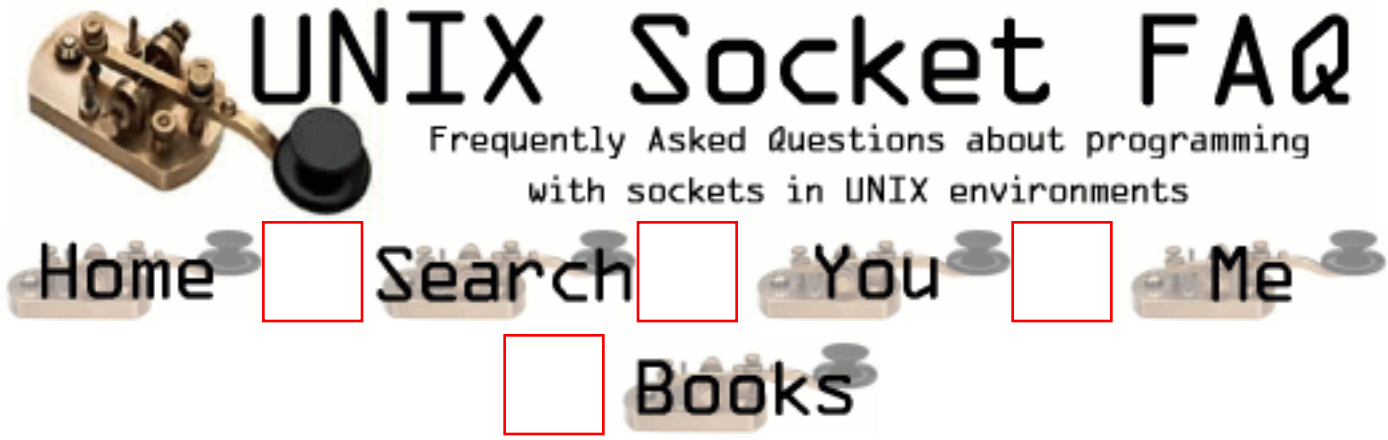


New Questions: re-establish socket connection

From: Dave Hallett

I'm trying to write a server that can disable and re-enable reads. I can use shutdown(fd, 0) to disable reads, but how can I re-enable reading on that same socket?

Add a Comment



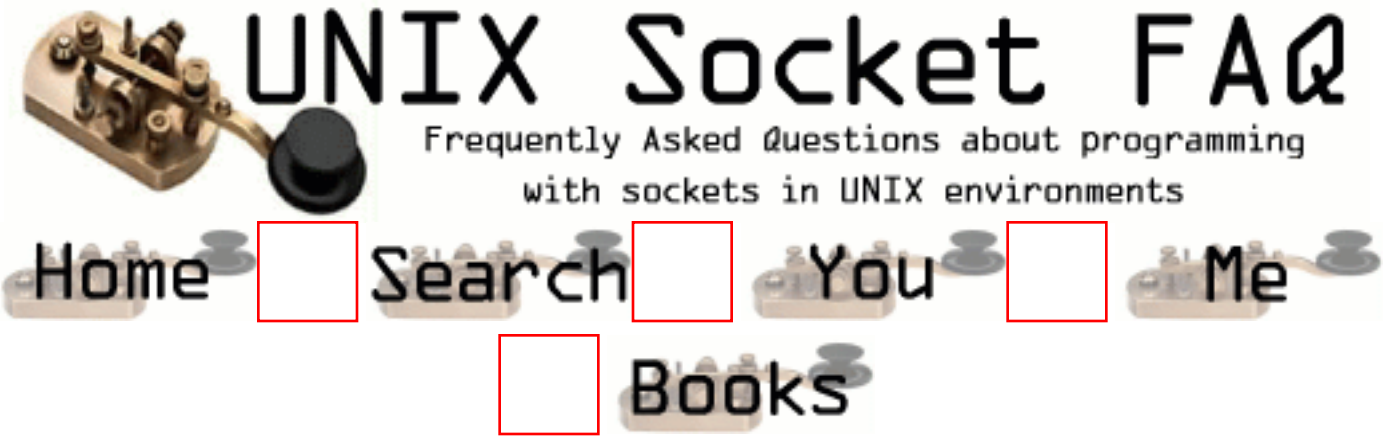
New Questions: Output Queue Gets Full

From: Peter Carlson

We have an application that gets a packet in and reflects it to a list of clients. The server is linux and the clients are Winxxxx. Every now and then the output queue for one of the sockets gets clobbered with 50k or so of data waiting to be sent. This causes the server to get real slow on the send(). Any idea. We've had this problem twice now, porting the server to NT, made the problem go away. Sigh...I'd rather have my server run on linux!

-Peter

Add a Comment



New Questions: Transfer a file using UDP

From: Kenny

I'm making a client/server program that transfers a file using UDP, but I'm having problems with how to actually wrap and send the frames, and how to make sure it has reach it's destination. And, no, not TCP allowed... Any help would be appreciated !!

From: Mr. Suggester

Send everything as bytes, otherwise you may be working on this for a long time. I don't know your details, but maybe you should try raw sockets.

From: Loco

Well, about making the frames, i cannot help you. I just don't understand that thing about frame size, packet size, window size and packet fragmentation.

However, to make sure your client received all the packets you sent him, you could use some sort of acknowledgement from client to server. Take a look at the description of sliding window protocols...

Good luck :D (HAL)

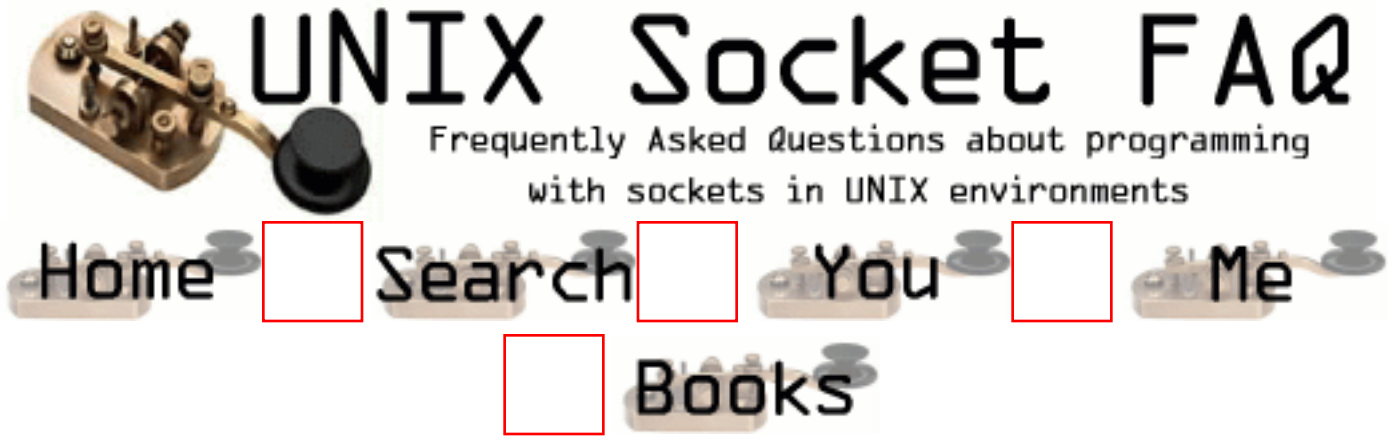
From: Khubilai

I've written a program that does this and it was a lot of work. I used around 1000 lines of code to make it work properly (including broadcast support and the ability to send several files). It can be done with less, but you still got a lot of work to do. The biggest problems was to make sure that every package reaches it's destination and that the file doesn't get corrupted. Another problem was actually to make it work to send several files. This may seem quite easy, but more things can go wrong than you think. It can be differculd to find every bug in the code. A lot of things can go wrong when using UDP. Things may work almost every time, but suddenly, something fails, and you don't know what it is.
Are you really sure that you have to do it this way?

If you still want to do this, I may be able to help, but I want you to know that this isn't as easy as it may seem.
Contact me if there is anything you want to know. Atleast, I know some of the misstakes that is easy to make

---

Add a Comment



New Questions: SO_LINGER socket option does not work

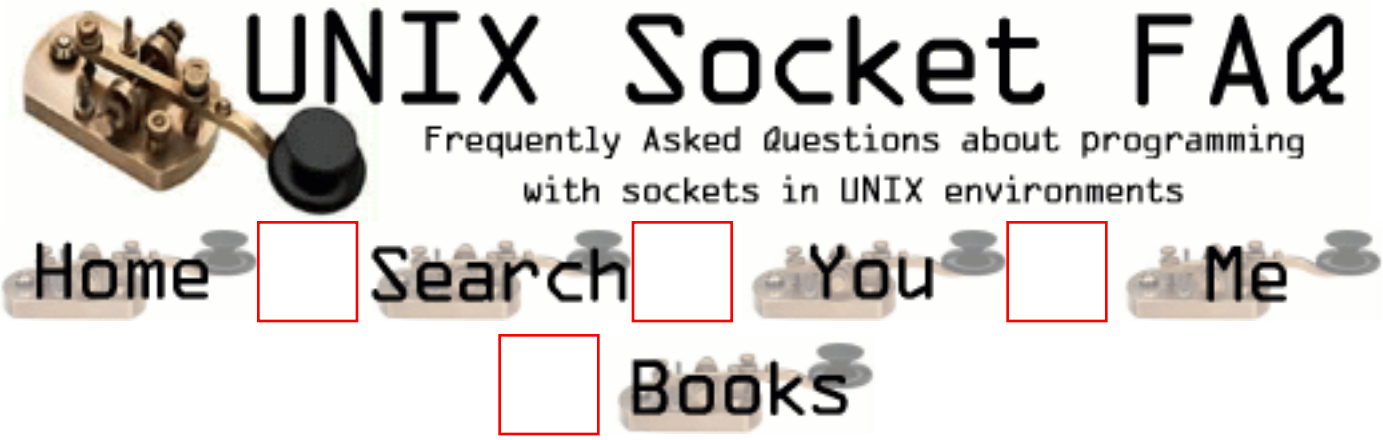
From: Evan

I am try to set the SO_LINGER socket option to avoid the TIME_WAIT state however when I run the following code a call to netstat -a still reveals the socket in TIME_WAIT. Is it possible I do not have the permission to set this option? I am running it on a university linux box.

```
-=-=-=-=-=- BEGIN CODE -=-=-=-=-=-
struct sockaddr_in temp_addr;
int temp_socket, retval, send_ip;
struct linger opt;
short send_port;

//create a socket to send with
temp_socket = socket(AF_INET, SOCK_STREAM, 0);
if (temp_socket < (int)0)
        err_sys("socket() error %d", temp_socket);

//set SO_LINGER option
opt.l_onoff = 1;
opt.l_linger = 0;
if (setsockopt(temp_socket, SOL_SOCKET, SO_LINGER, (char*)&opt, sizeof(opt)) < (int)0)
        err_sys("setsockopt() error\n");

//format the address to send to
bzero(&temp_addr, sizeof(temp_addr));
temp_addr.sin_family = AF_INET;
temp_addr.sin_port = htons(send_port);
temp_addr.sin_addr.s_addr = send_ip;
```

```
//connect
retval = connect(temp_socket, (SA*)&temp_addr, sizeof(temp_addr));
if (retval < (int)0)
{
	close(temp_socket);
	err_sys("connect() error\n");
}

//write buffer
write(temp_socket, "helloworld\n", strlen("helloworld\n")+1);

close(temp_socket);
```

Add a Comment



New Questions: Multiple Servers

From: Andy

I am new to network programming and sockets. I need to
design a client that can connect with multiple servers to gather
information about the computer's status, such as CPU usage etc.
Any help will be welcomed

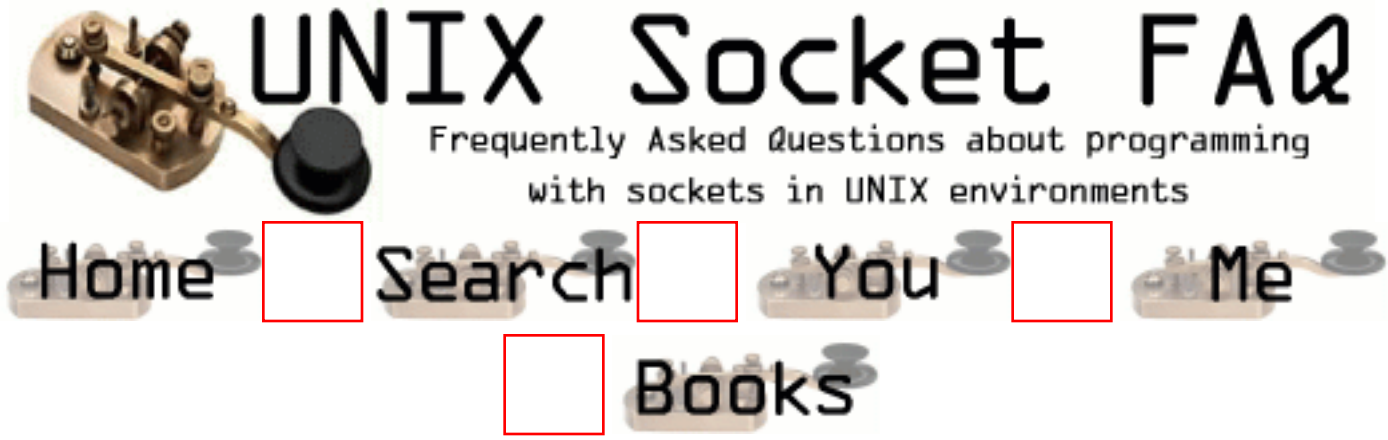
From: Anonymous

Please give more details. What operating system (Unix or Windows - I presume), and what architecture or platform are these machines on, for example Sun, HP, ...? It will be easier to give you a more precise answer then.

From: Andy

It will be for the Linux Operating system and the Solaris stations (SUN OS 5.6). The aim is to have a server on each monitored machine that runs checking script,
after a connection is made the raw data will be sent to the client processed and displayed as a GUI interface.
Any references or suggestions will be great
Cheers

Add a Comment



New Questions: How to terminate the read/recv call..?

From: Vijay

Hi,
I'm writing a client/server application using sockets.To read the data from the client/server I tried with read() and again with recv().The problem is,both the functions didn't return "0" or "-1" at the end of data.The functions simply hangs up(indefinitely waiting for some more data) when end of data was reached without returning any values.Any one could suggest me how to terminate the reading when end of data encountered? Any alternative methods r there to finish the read/receive??
I set the options as
n=read(sockfd,buf,strlen(buf),0);
or
n=recv(sockfd,buf,strlen(buf),0);

sockfd - file descriptor from select call.
buf - buffer size.

From: Hector Lasso

Hi,

Regarding the question: I guess 'buf' is the buffer to hold the data recv'd and not the buffer length. read() expects 3 parameters, so the last 0 is not needed. You should use the maximum capacity of your buffer or the maximum number of bytes you expect to receive, and not the current string legth in your buffer (i guess this is an error, but it might as well be ok because of your program's logic)

There are many ways to "timeout" a call to recv/read, however i think the problem is in your

code or in your question:

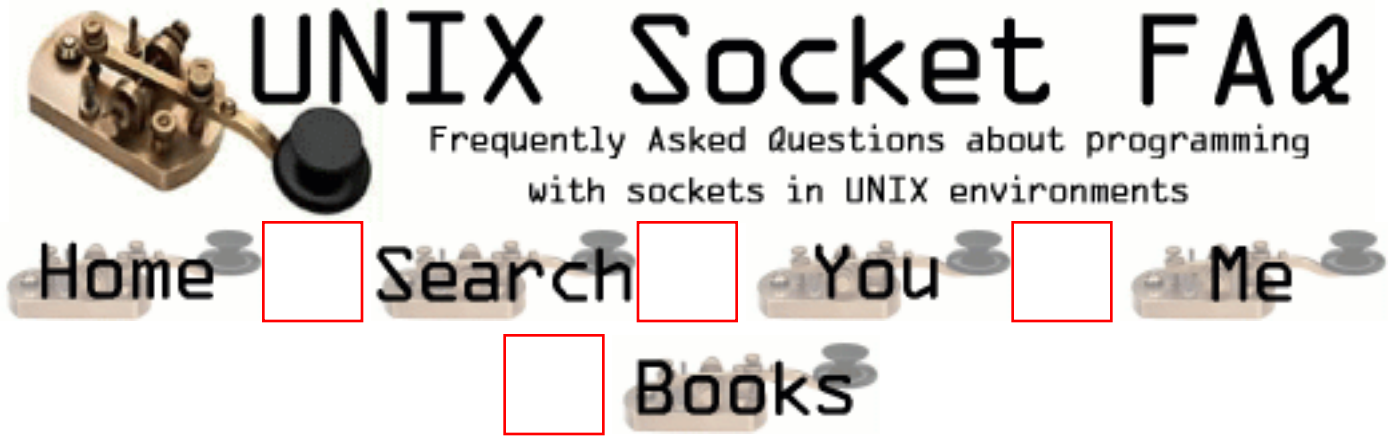
When you mention "end of data" you mean the other end closed the connection, or just finished sending some data?

Is the recv/read function call inside a loop, meaning it enters once again after it has returned some data?

read/recv always block until data is available. That is their normal behaviour. They will not return until data is available or a signal is caught.
You can change a file descriptor to non-blocking mode. That means it will behave a little bit different, i suggest you read about non-blocking mode in this faq's section.
You can also use select() on sockets with timeout parameter which will let you regain control after some period of time when no data is available.
Also, you could just set a timer with alarm() and then call read/recv, you'll have to set a signal handler for this or your program will exit. When the signal handler returns, the function is aborted returning -1 and errno=EINTR

Post some code.

Add a Comment

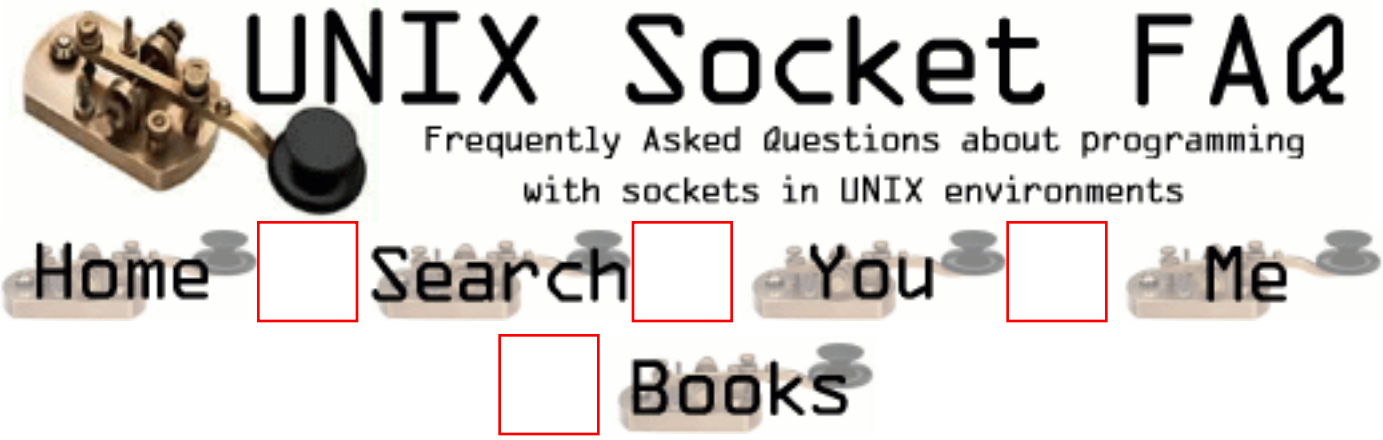


New Questions: Error in bindind an UDP socket

From: Nedu

In server programming using UDP, When I am trying to assign serv_addr.sin_addr.S_addr)=htonl(INADDR_ANY), bind error is comming. But, If I use the host IP address instead of INADDR_ANY, then it is working fine. What is the reason?

Add a Comment



New Questions: Invalid argument!?

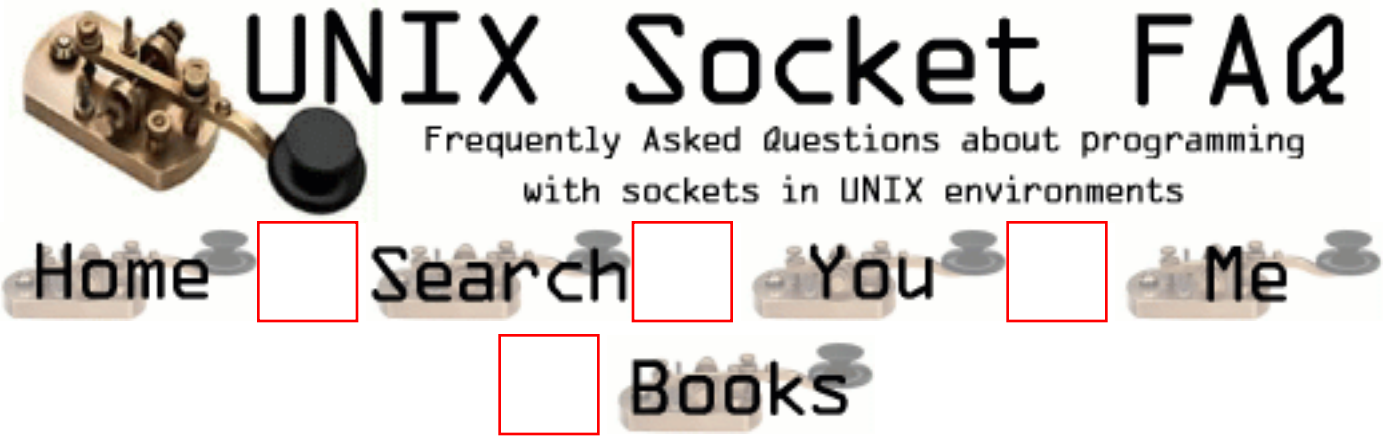
From: James McLean

Everytime I try to utilize the function recv() it provides me with a totally inexplicable error: Invalid argument... Despite my efforts at resolving this dilemma and reducing the code to its most simplistic form, I am still unable to get passed this. Any suggestions?

From: Rob Seace

Post the code that causes this... My blind guess without seeing the code, is that either the buffer or the length you are passing in is invalid, in some way... Eg: a negative/zero length... Or, perhaps you're passing a flag which isn't valid, for some reason... Or, perhaps the FD isn't valid for recv()'ing on (eg: you've done listen() on it)...

Add a Comment



New Questions: Blocking socket recv with pthread

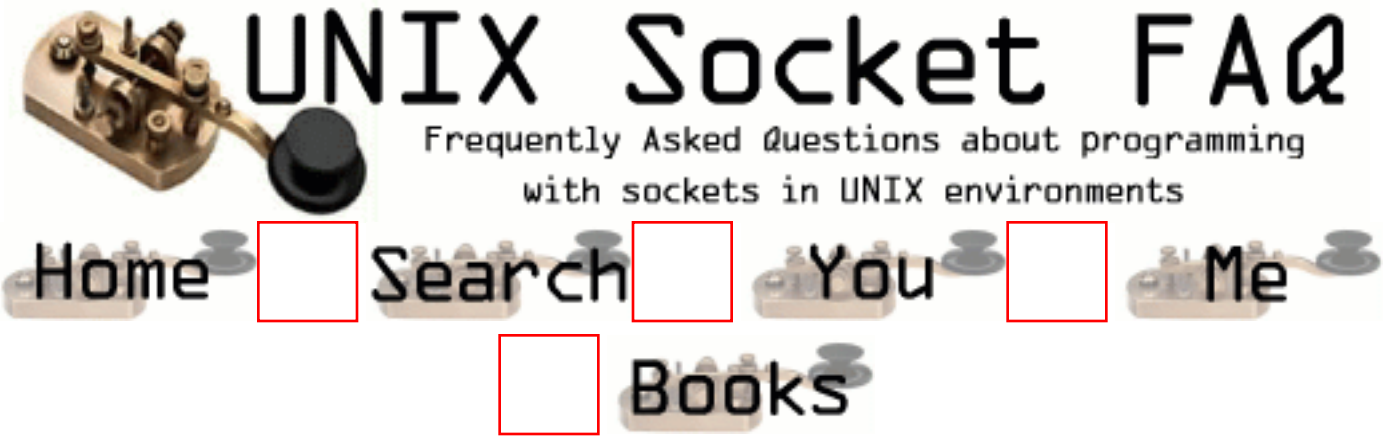
From: Jean-Sebastien

I want the recv to block the thread it is executed in, but it
is actually blocking the whole process (and all the threads in it).
Why, and how can I overcome this problem???

From: Margrit

I think, you should call select() before recv()

Add a Comment



New Questions: Instant Messaging Server

From: John

I'm developing a kind of IM server and I want like 100,000 clients
being able to connect to the server waiting for a message.
I was thinking about accepting a TCP connection from every
client when it connects, and keeping the socket opened until
the client disconnect from the service, but I could have about
100,000 sockets opened at the same time and I realized that
it was maybe not a good idea.
If I close the connections and try to reconnect to the client
when it gets a message will I have problem to reconnect to
the client through firewall, proxy and stuff like that?
If I keep the connection opened on client side, is there
any way on server side to be able to communicate with
any client without keeping all sockets opened?
Well how ICQ and other IM services does?

Thanx

From: Loco

Do you really need to keep connections opened?

Why don't you use UDP datagrams?

From: John

I have to be 100% sure if the packets reach the client or server

---

From: Rob Seace

Well, you could implement your own ACK'ing/retransmiting
layer on top of UDP... Or, you could use something like

SOCK_RDM, if your system supports it... (I've never tried it, myself... But, it looks to be basically reliably delivered UDP...)

But, I just don't think it's going to prove feasable to maintain thousands of open FDs for connected TCP sockets... If you go with TCP, you'll probably just have to eat the overhead of a connection set-up and tear-down for every message sent... *shrug*

From: Gavin

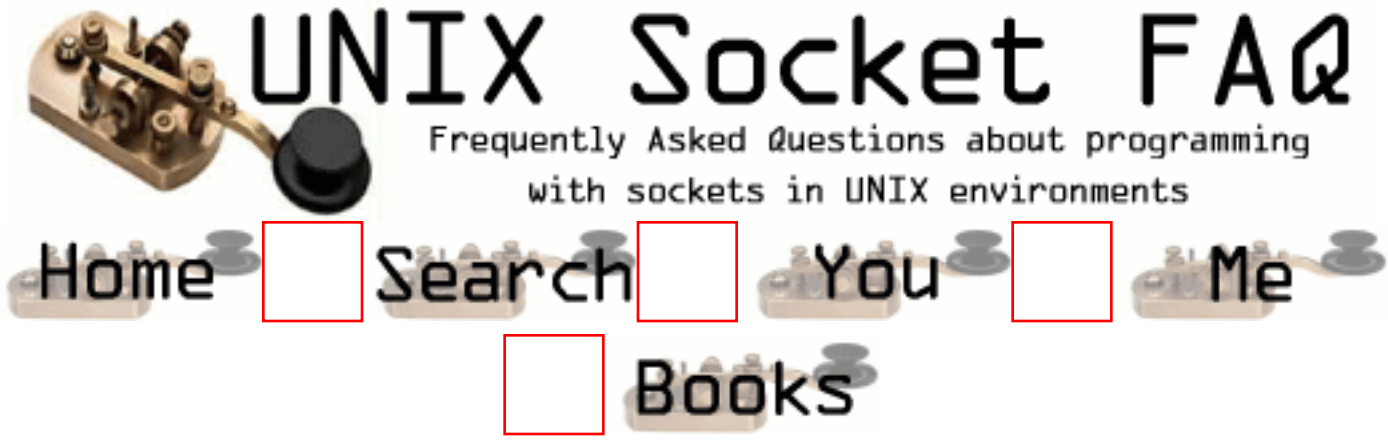
I wrote an instant messaging service for my company. We had about 5 servers, each with 10,000 constant TCP connetions to it and it worked perfectly. The benifits of keeping constant connections far outway the cons. Just make sure you are using at least the 2.4 kernel of linux, or some other OS that supports a high level of open fds.

From: chris

but, how did you write the daemon? does he fork every time a connection's coming in? or how?

greets,
chris

Add a Comment



New Questions: How to detect if process id is running

From: EPal

Given a process id, how can I tell if it is currently running?

From: Loco

This may seem pretty obvious...

my_shell# ps -ef | grep YOUR_PROCESS_ID

Try:
kill(YOUR_PROCESS_ID, SIGUSR1);
from within your code, if errno == ESRCH then your process did not exist...
If it is ok, then the process is now probably death because the default action for SIGUSR1 is to terminate the process, unless it handles this signal...
I'll try to find more info on it...
:D (HAL)

From: Loco

Hi,

You can also use "getsid(PROCESS_ID);" which would return the session id of the process or -1 when the process does not exist...

Some OSes have a "prctl()" system call that can be used to interact with a process, however i have not found much documentation regarding this.

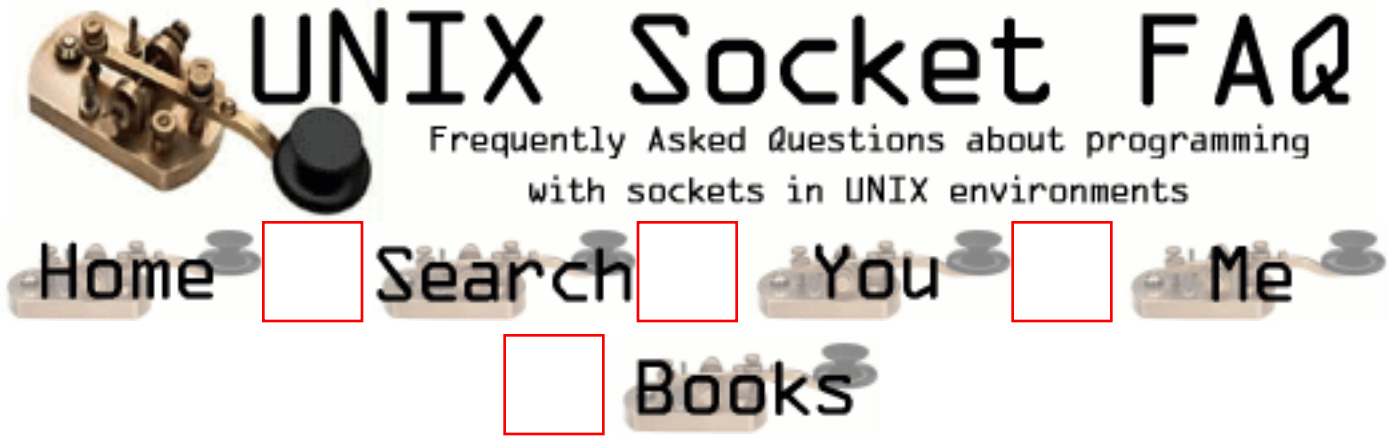
:((HAL)

From: Rob Seace

Using a signal of 0, instead of a real signal#, in a kill() will just check for the process being alive, but not send any real signal...

But, um, what does this have to do with Unix sockets, again?? ;-)

Add a Comment



New Questions: Broadcasting Problem

From: redbull

I am having a very peculiar problem ..I am running my program on port 9099 on machine A , B, C , D ..
I have set the socket on port 9099 to NON_BLOCKING mode and also it has been bound to the host's IP addres
Now , when the app on Machine A broadcasts a datagram (the socket has been set to broadcast mode) , over the
interface B/C addr to port 9099 , machines B, C , D on port 9000 are
NOT receiving any of my datagrams ?
However , I can do a point2 point communication(e.g A-B :9099)

Can anyone pl suggest something ?

Thanks in advance

Red

From: redbull

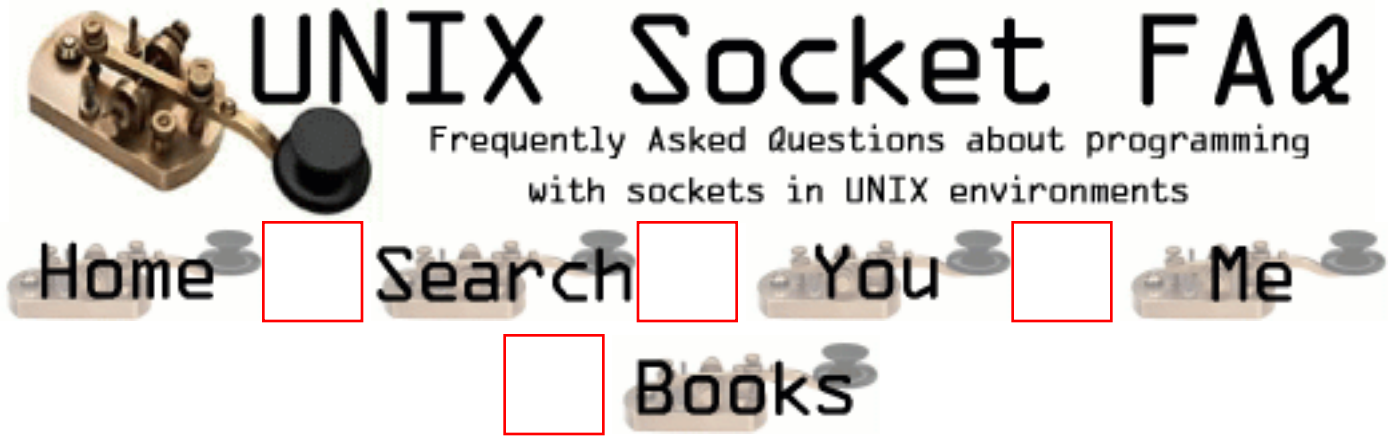
Sorry , The port in use is 9099 NOT 9000

bull

From: Khubilai

I think it would be easier for us if you could show us your code.

Add a Comment



New Questions: select problem

From: red bull

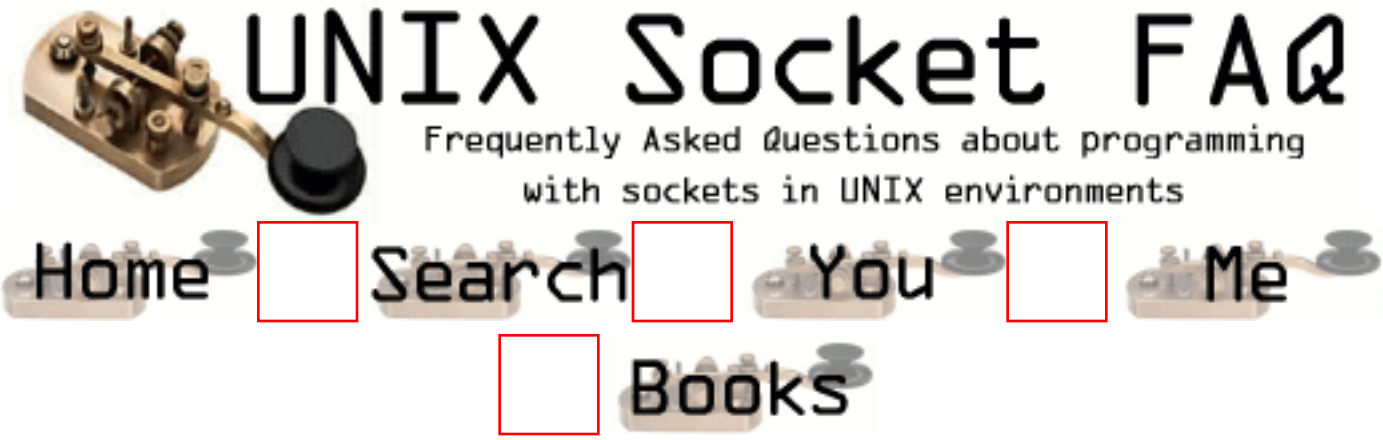
My function is using select to monitor activity on a socket
I also have a alarm for the function to return..
Now the functions NEVER returns after it receives data..It gets trapped in the infinite loop

Here is my code :

```
FD_ZERO (&readset);
FD_SET (sockfd, &readset);

    signal(SIGALRM, NiceMonitor_handler);

    alarm(20);

    /* Reset The Variable */
    _nice_monitor_timeout = 0 ;

    (void) setnonblocking(sockfd);

    for (;;)
    {
      bcopy((char *)&readset, (char *)&rset, sizeof(readset));

      printf ("\n ...Monitoring LB_PORT \n");

      n = (select (FD_SETSIZE,&rset,NULL,NULL,NULL));
    if (n < 0 )
      {
        if (_nice_monitor_timeout == 1)
        {
```

```
          printf ("\n LEAVING NICE_MONITOR \n");
          return;
}
        else
printf ("\n ERROR in select () in NICE_MONITOR () ");
          exit (EXIT_FAILURE);
       }
     if (n == 0 && _nice_monitor_timeout == 1){
       printf ("\n....");
       return;
     }

      if (FD_ISSET (sockfd ,&rset) )
{

FD_CLR (sockfd, &rset);
       /* Reset Structure and flags */
          //Do the readinng
        }
}
}
```

Add a Comment

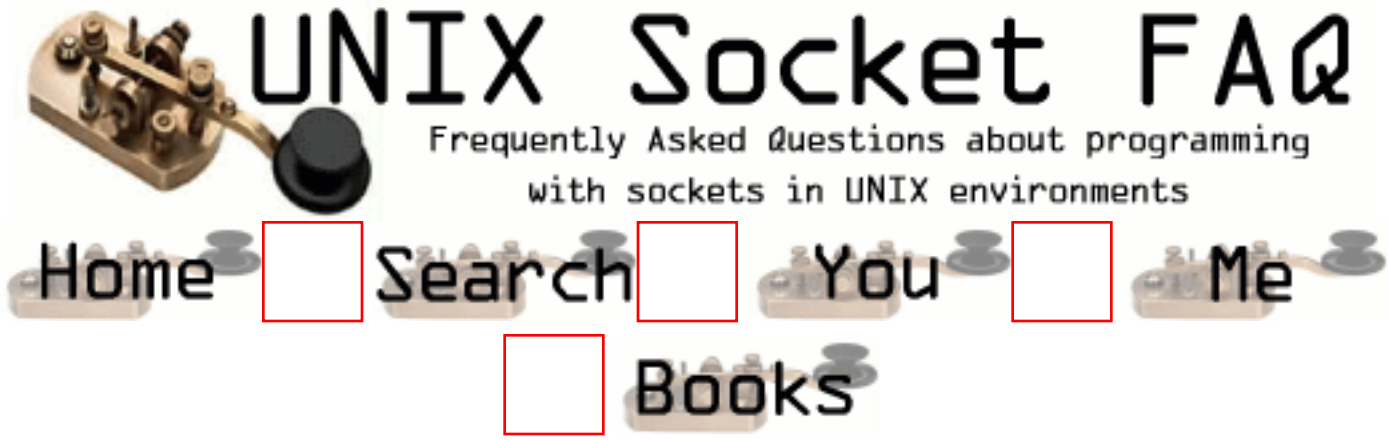


New Questions: peer-to-peer chat application

From: Selim

I need a peer-to-peer chat application code that is written in c/c++ computer programming language and Berkeley sockets programming language immediately. How can I find such a code.

Add a Comment



New Questions: help on UDP broadcasts

From: voci

Hi, Im having some trouble with broadcast UDP messages in linux (kernel 2.4.3). Basically Im creating a socket as follows:

```
int hSocket = socket(PF_INET, SOCK_DGRAM, IPPROTO_UDP);
if (hSocket == 0)
{
  printf("Could not create socket.\n");
  return 1;
}
int broadcast = 1;
setsockopt(hSocket, SOL_SOCKET, SO_BROADCAST, &broadcast, sizeof(int));
```

The socket gets bound properly and recieves any messages sent to it specifically (ie 192.168.0.151) but does not receive broadcast messages (ie 192.168.0.255 or 255.255.255.255).

Whats more, the code works correctly on a windows system (with the appropriate conversion between sys/socket.h and winsock2.h).

Any help would be appreciated. Thanks,

Nathan

From: redbull

me too having the same problem

From: Nathan Lovell

I thought Id answer my own question in case anyone is wondering. After much searching I discovered that to recieve UDP broadcasts you need to bind the socket that is recieving them to

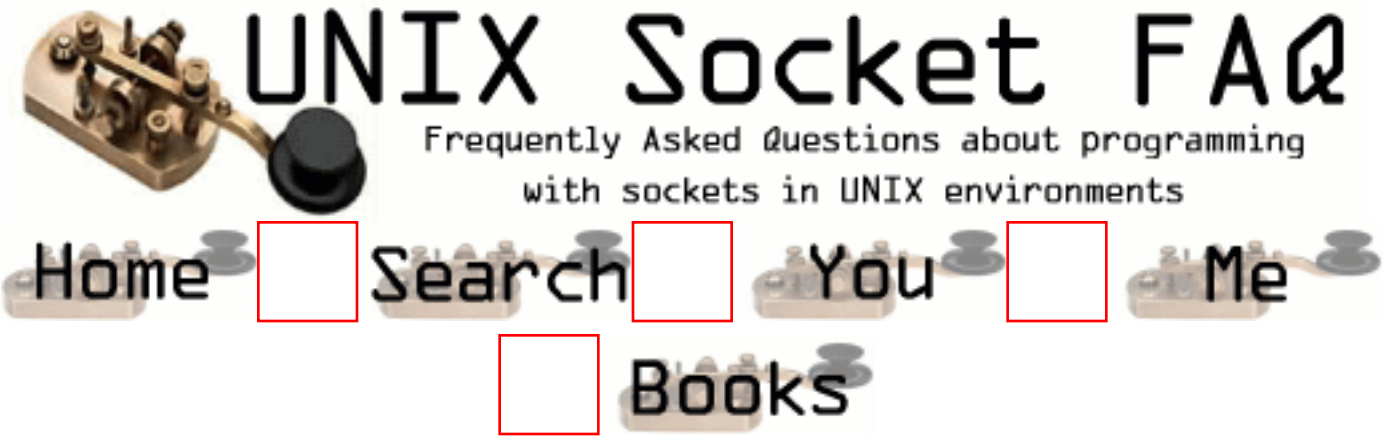
0.0.0.0, not the address of a specific ethernet adapter.

Hope this saves someone the 2000 hours of searching the web that I had to go through!

From: rebull

That's wrong concept..Actually u need to bind to the Interface
Broadcast Address which is unique for every Interface the NIC is bound to ..

Add a Comment



New Questions: select question

From: redbull

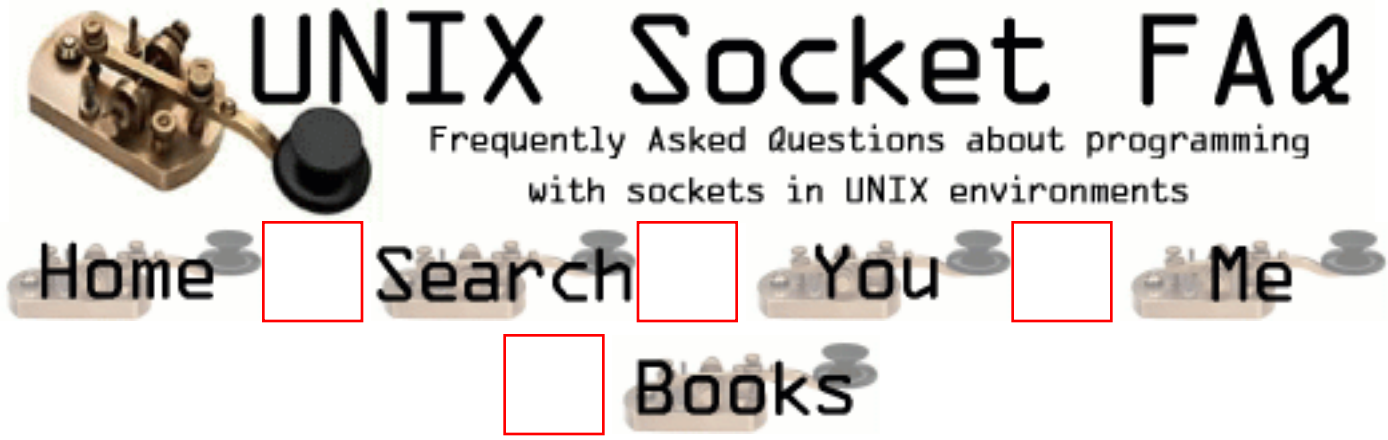
Is there any problem / difference if arg1 to select is FD_SETSIZE
instead of the maxfd+ 1?
i.e
select (FD_SETSIZE,....);
INSTEAD Of

maxfd = MAX(NOof Active Desriptors in the sys);
select(maxfd +1 ,...

From: Rob Seace

A problem? No, as in, it will still WORK ok... However,
your performance will almost surely suffer... The
performance of most select() implementations tends to suffer
greatly, as you increase the number of FDs in use... So,
if you tell it you have the max number in use, you're
guaranteeing yourself the worst possible performance...
Why do you not want to keep track of the highest FD in use?
It's not that difficult... And, it's really necessary to
know, since you have to keep your fd_sets correctly filled
all the time, so you must know what FDs you're dealing with
at all times... *shrug*

Add a Comment

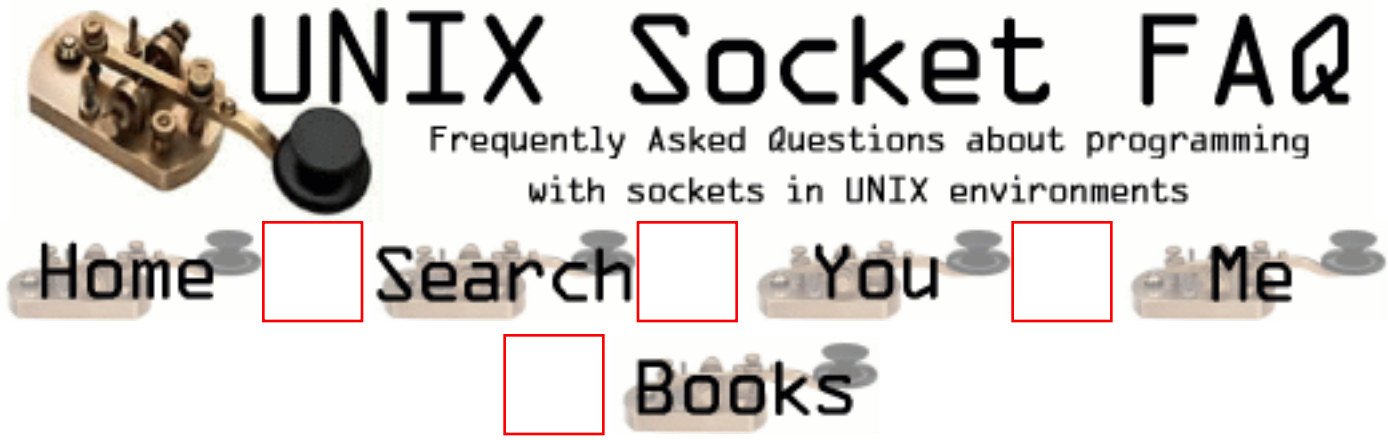


New Questions: Socket usage with Unixware 7.1.1

From: Belinda

When I create a socket in c on Unixware 7.1.1 with the following command :
s = socket(AF_INET, SOCK_STREAM, 0) it works for a couple of times and then core dumps.
Any idea why this is happening ?

Add a Comment

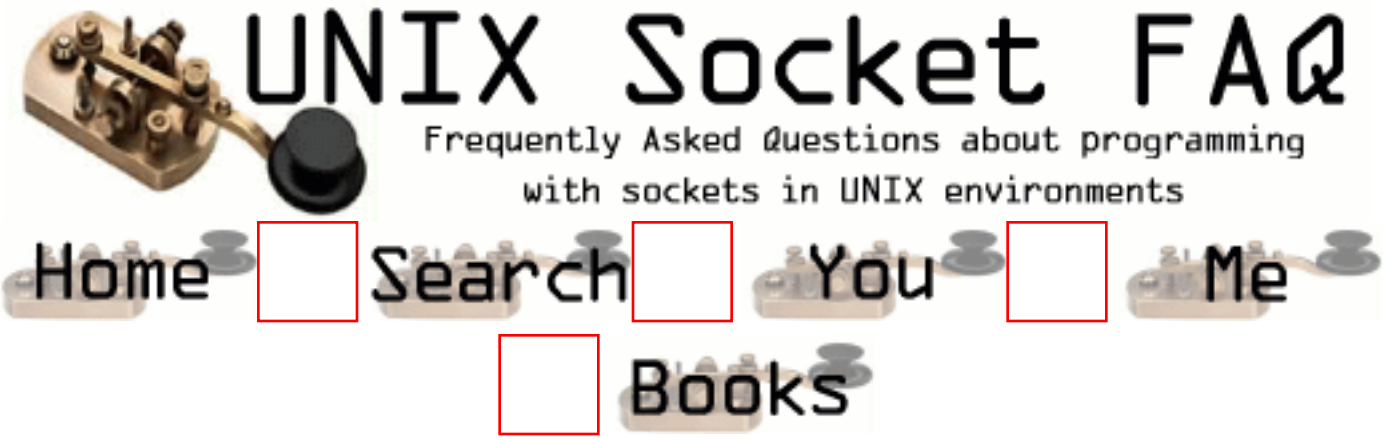


New Questions: How can I send data from server to client?

From: shyang

I am a newer ,and Now I can receive data from client,but I can't send data to client ,How can I do ?

Add a Comment



New Questions: socket source code

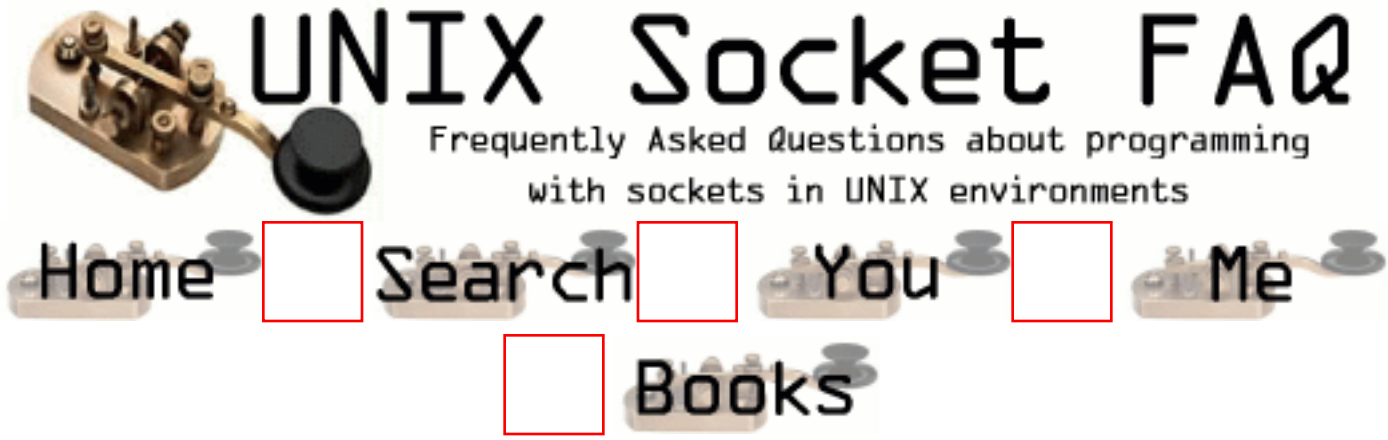
From: Dave Hallett

Where can I find the source code for socket system calls?
(e.g. socket, accept, listen, shutdown, setsockopt, etc)

From: Rob Seace

Um, for what OS?? If you've got Linux, take a look in
"/usr/src/linux/net/"... If you're talking about *BSD, I
don't know the source layout, but I'm sure it's findable
in the kernel source... Or, get a copy of Stevens'
"TCP/IP Illustrated" Volume 2, which shows you the BSD code
for all of the TCP/IP functions, as well as explaining
everything...

Add a Comment



New Questions: OPEN_MAX SUN app to Linux

From: Jesper Nordestgaard

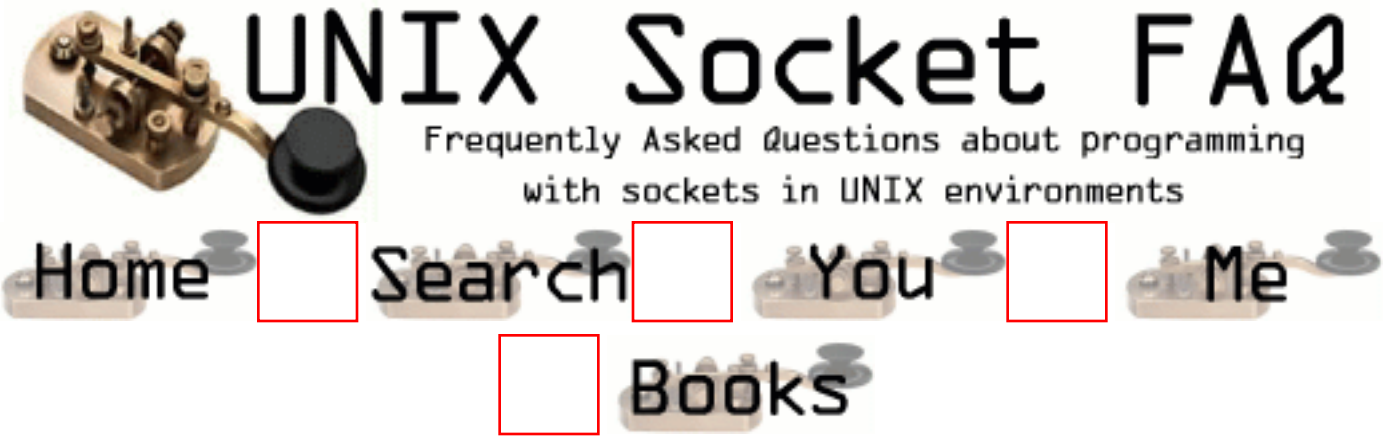
Hi

I'm porting a C application from SUN to Linux. The only problem is the OPEN_MAX variable defined in unistd.h.
This variable holds information about all open files e.g count. My compiler stops be course it does not know this variable. What’s the name of the corresponding Linux variable?

From: Rob Seace

Linux has an "OPEN_MAX" defined... Just #include
<limits.h>... (But, I'm not sure it's strictly accurate,
either... It seems to be set to 256, but as far as I know,
the real upper limit on number of open FDs is 1024...
Which you can determine based on FD_SETSIZE, and Linux's
NR_OPEN... *shrug*)

Add a Comment



New Questions: multi-threaded connections

From: Kallol

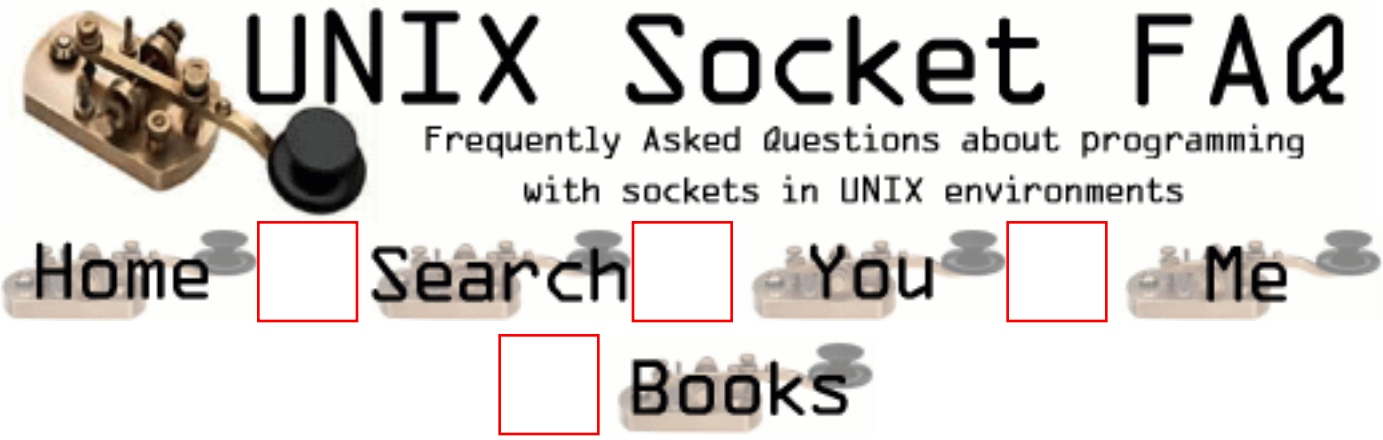
I m trying to connect to 3 different machines using threads but the problemm is that the some threads are endlessly trying to connect....even when the receiving end (one of the 3 servers) is ready to accept any connections....this is a random phenomenon but always atleast a single thread loops trying to connect.....please give me ur advices

---

From: Shanawaz

HI, Great question.

Add a Comment



New Questions: Open connection!

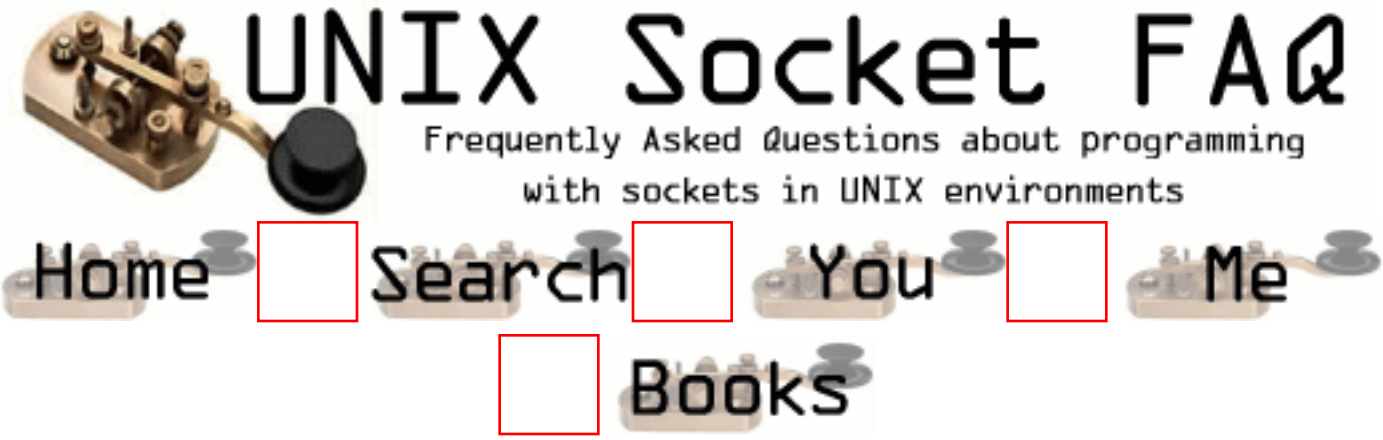
From: Rajaji Selvaraju

Hi,
I am facing a dilemma and i want somebody to help me out on this please....
Let me explain my scenario here.

I opened a connection to the remote host and the data has been sent successfully only for the first time(second time it fails). In the sender's side, i don't want to close the socket descriptor and i want to use the same open descriptor for further transactions. Is it possible?
If not how do i over come this problem?
Please help me.

Thanks,
Rajaji Selvaraju.

Add a Comment



New Questions: accept file descriptor

From: HHoxha

Hi.
In pop service started by inetd when you telnet to port 110 you got a connection.
From this moment on , i imagine , this proccess has got its own file descriptors returned from accept
call(inetd)How could one find these file descriptors for use ??Is there any mean.
Thanks
Hysen

Add a Comment

New Questions: Unix command to know the ports.

From: Andy

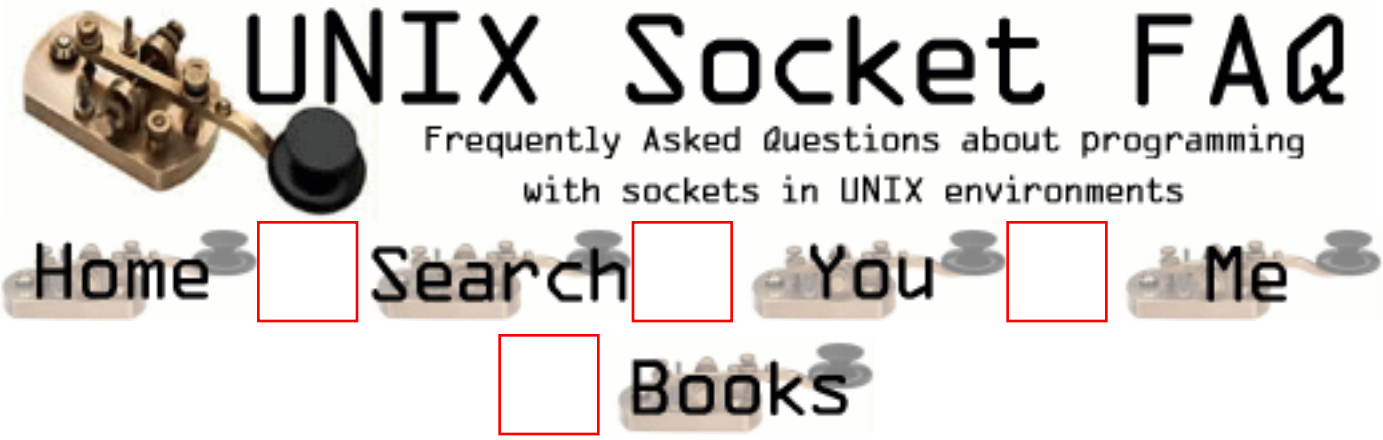
Which is the unix command through which i can check what
all ports are busy and what all are free.
What does the error mean cannot bind to the port ? How do i
see whether the port the application is trying to coonect
is free or not ?

From: Rob Seace

Sounds like you want the "netstat" command... Try
"netstat -an"...

Failure to bind could be because the port is in use, or
because you don't have permission to do so (if it's a port
under 1024), or maybe just some problem with your socket...

Add a Comment



New Questions: Can UDP be used for sms

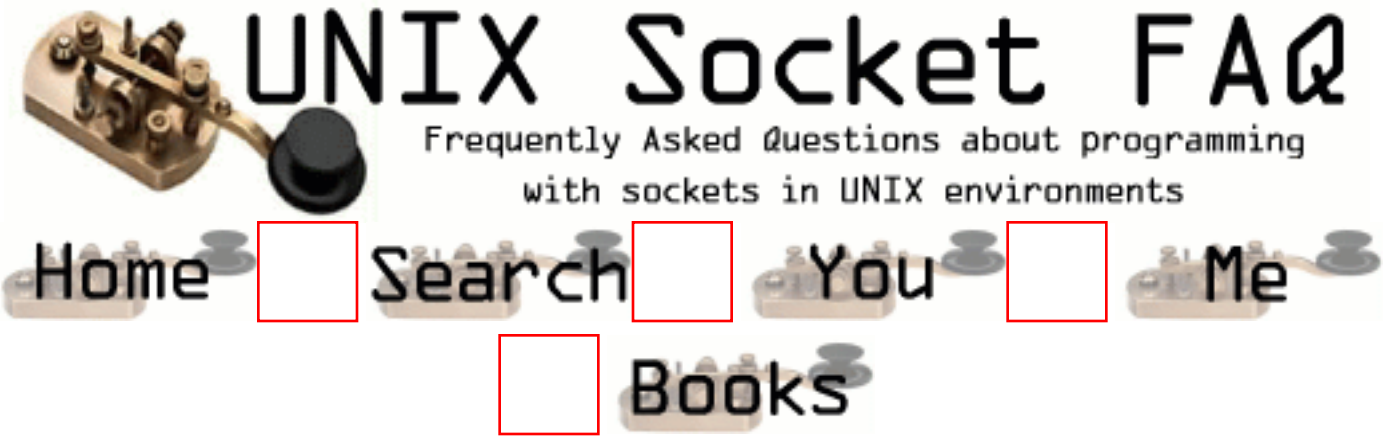
From: Tomaz

Hi
I'm new to socket programming can somebody help & advise me on this subject.I'm doing a project to Send & Recv SMS(short messageing service) from a modem driver i like to know if UDP Sockets can be used for this program Thanks in Advance Have a nice day..
Tomaz

From: udayan

You can do it if you are using a protocol like SMPP for short message services.The SMPP PDU's that you make can be transported over UDP/IP

Add a Comment



New Questions: sending http request via proxy

From: Jharana Mehta

How can I make a URL connection through a proxy if the proxy server requires login information (user name,
password)?

From: thirunadha rao

First u open a socket to ur proxy server. Now let's take that
u need to access the URL www.rediff.com and the method is GET.
Then the HTTP request should be formed like this.

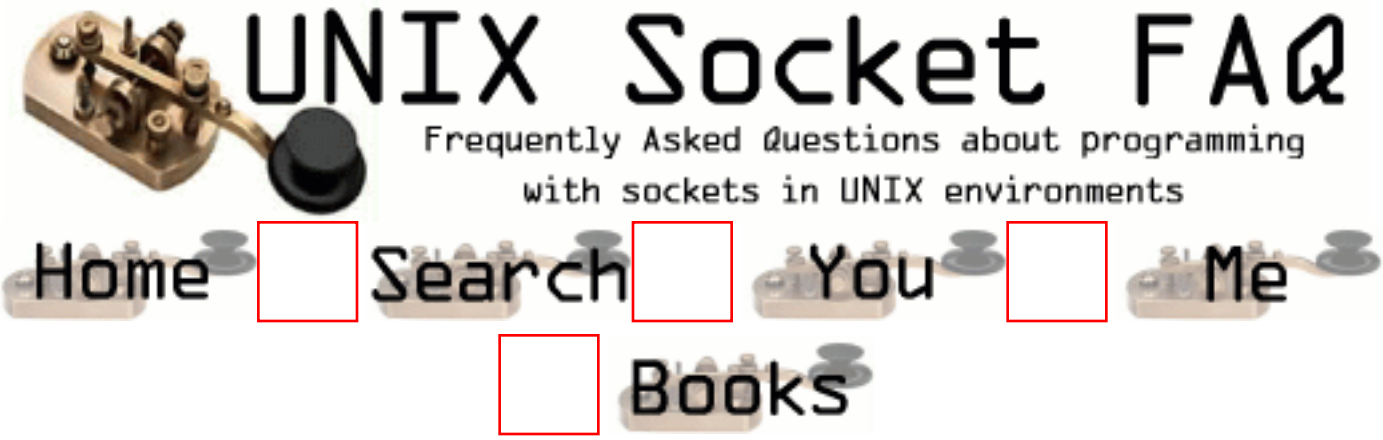
GET http://www.rediff.com HTTP/1.0 \r\n Proxy-Authorization: Basic
BASE64encoding(username:password)\r\n other header fields

First u have to get the base64 encoding of the string username:password.\
suppose if ur username is thiru and password is abc then u
have to get the base64 encoding of thiru:abc. paste this inplace of
BASE64Encoding(username:password) in the above request line.

Here is the example using the username as thiru and password as abc. There are many utilities available on the net to convert a given string to BASE64 encoding.
GET http://www.rediff.com HTTP/1.0 \r\n Proxy-Authorization: Basic dGhpcnU6YWJj \r\n

---

Add a Comment



New Questions: Can you help me.?

From: Carmen

I read some text about socket. but I can't do my homework.
Homework is to show socket descripter in integer.
Open 10 socket. What must I do.?

From: Loco

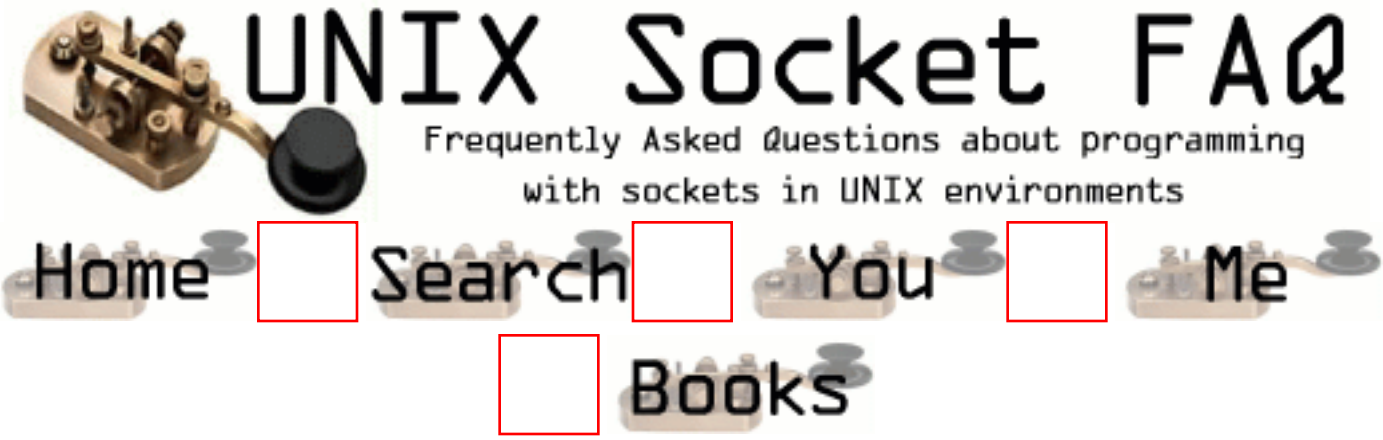
You have something like:

sock = socket(.....);

And you want to show the socket descriptor in integer, then you should just

printf("My socket descriptor is: %d\n", sock);

Add a Comment



New Questions: Reuse of socket descriptors in linux

From: Yannis Pavlidis

Hi,

I am running my server on 2.2.16-22 red hat linux and I would like to ask the following question. The Server accepts a connection from a client and I print out the socket handle. Then after serving the request and closing the connection and accepting a new connecion the handle is the same as in the first request. My question is does the kernel/tcp stack makes the socker descriptor reusable immediatelly after it is closed? Cause in other OS I see that the socket descriptor is keep incrementing.

Thank you,

Yannis.

From: Jean-Sebastien

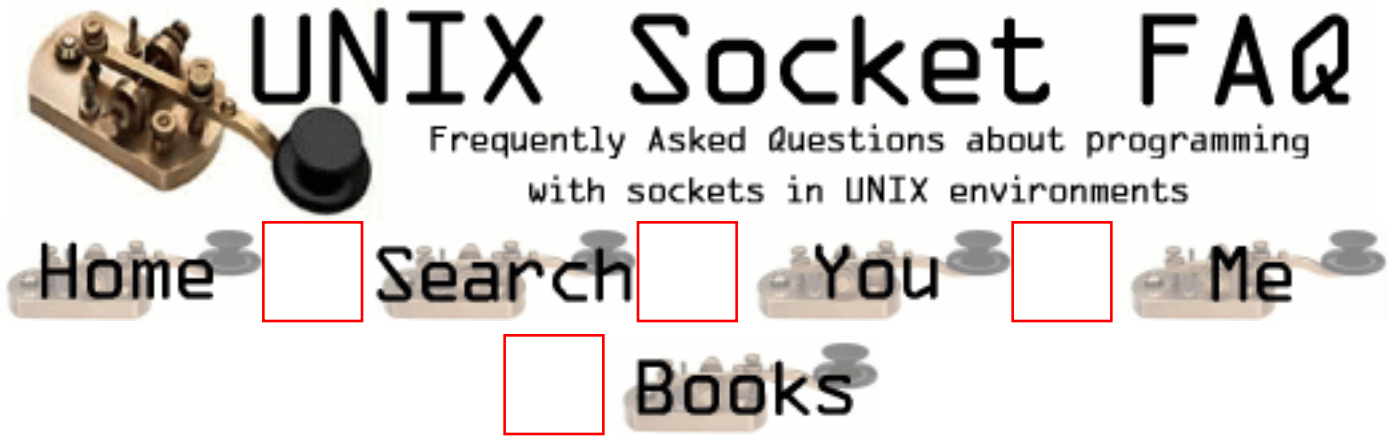
I'm working on the development of a server running on red hat
linux for while and it seems like it re-uses the lowest
file descriptor number available.

From: Rob Seace

As far as I know, this is standard behavior in pretty much
all Unix-like OS's in existence... I don't think I've ever
met one that DIDN'T do things this way (or, else, I didn't
notice)... I can't imagine why any OS would NOT do things
this way... In fact, I've seen lots of code which depends
on this very behavior... (Eg: they'll close FDs 0, 1, and
2 (stdin, stdout, stderr), then immediately open "/dev/null"

or something, assuming they'll get back those just closed low-numbered FDs, redirecting stdin/stdout/stderr to their newly-opened files... Personally, I usually freopen(), or dup2(), to force things to the correct FD, just to be certain... But, it seems a logical assumption to make, anyway...)

Add a Comment



New Questions: Non-blocking client

From: Marvin Massih

In school, we have a computer that is behind a firewall we don't have access to. However, I want to administrate that computer remotely.
So I wrote a program that is to execute commands from emails. Most of it was taken from the libmutt pop.c.
But my program was incredibly slow. To me it seemed like that was due to read(). So I made the socket non-blocking.
But now I get one inverse question mark (printf("%d", ch) prints "-65") and then "EOF" is printed.
If any could help me, I'd prefer an answer via email (m.massih@web.de).

Well, here's the code (the printf()'s are just debug messages):

```
int main()
{
  //...
  if(fcntl(socket, F_SETFL, O_NONBLOCK) != 0){
    printf("Damn! Couldn't set non-blocking flag: %s\n",
strerror(errno));
    exit(1);
  } //...
}

int timeout = 10;
int timed_out = 0;

void on_timeout(int sig)
{
  timed_out = 1;
}
```

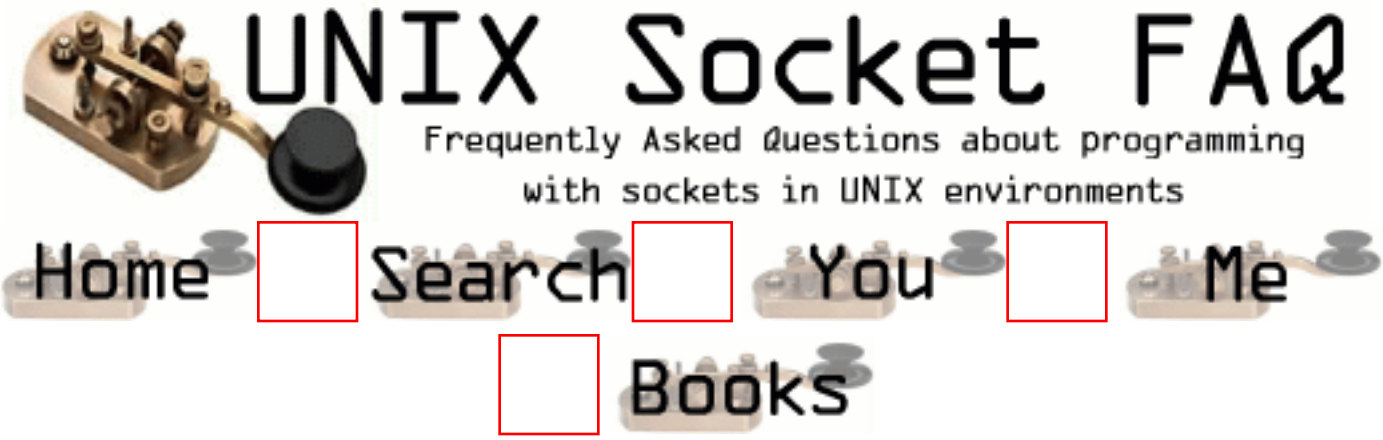
```
int get_line(int fd, char *s, int len)
{
  char* old;
  char ch;
  int bytes = 0;

  old = s;

  if(fcntl(fd, F_GETFL) & O_NONBLOCK){
    ssize_t r;
    timed_out = 0;
    signal(SIGALRM, on_timeout);
  set_timeout:
    alarm(timeout);
    while(!timed_out){
      if((r = read(fd, &ch, 1)) > 0){
printf(">%c\n", ch);
if(ch == '\r'){
goto set_timeout; //we don't want "\r\n" but just "\n"
}
*s++ = ch;
bytes++;
if(ch == '\n'){
*s = 0;
printf("I've got: \"%s\"\n", old);
return bytes;
}
/* make sure not to overwrite the buffer */
if(bytes == len - 1){
*s = 0;
printf("I've got: \"%s\"\n", old);
return bytes;
}
      }else if(r == -1){
if(errno == EAGAIN){
printf("Again!\n");
continue;
}else{
alarm(0);
timed_out = 0;
printf("-1 Bytes\n");
return -1;
}
      }
```

```
      *s = 0;
      alarm(0);
      timed_out = 0;
      printf("EOF\n");
      return -1;
    }
    *s = 0;
    alarm(0);
    timed_out = 0;
    printf("Timeout\n");
    return -1;
  }else{
    while(read(fd, &ch, 1) > 0){
      if(ch == '\r'){
continue; //we don't want "\r\n" but just "\n"
      }
      *s++ = ch;
      bytes++;
      if(ch == '\n'){
*s = 0;
printf("I've got: \"%s\"\n", old);
return bytes;
      }
      /* make sure not to overwrite the buffer */
      if(bytes == len - 1){
*s = 0;
printf("I've got: \"%s\"\n", old);
return bytes;
      }
    }
    *s = 0;
    printf("EOF\n");
    return -1;
  }
}
```

TIA

Marvin <m.massih@web.de>

Add a Comment



New Questions: Multiple IP Address in Client, how to....

From: Sudhakar Chakravarthi

Hi,
I am facing one small problem it will be greatful if you can guide me.
I am writing Client/Server program using socket.

My Client host is having two IP address(Two NIC Cards, ex 192.168.1.1 and 192.168.2.1). Two NIC Cards each one in different LAN ie. my client is in two LANs)

My Server process is running in one LAN(ex192.168.1.10).

I want to connect to the server 192.168.1.10. While connecting to the server do i need to bind() the client end for the ip address 192.168.1.1(B'cos same LAN)? Or Operating system itself will select the client interface?

Thanks in advance,
Sudhakar.

From: Rob Seace

No, don't bind() your client sockets, at all... Let the OS
do it; it generally knows what's best, better that you
do... ;-) Just call connect(), and the bind() will get
done automatically for you...

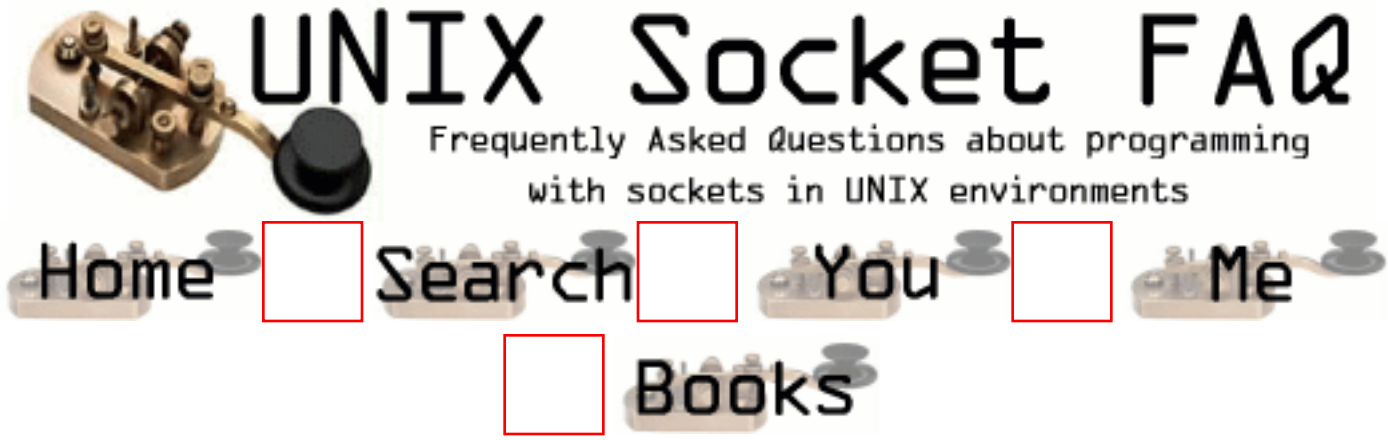
From: Sabyasachi Malakar

Is it possible to get the list of diff ip one host is supporting?
can we read some file in the system and get the list of diff ips
supported by the host (In case of MULTI-HOME NETWORK)

From: Rob Seace

Yes, it's possible... Though, it's generally a real pain to code... ;-) It typically involves a lot of obscure ioctl() calls... Your best bet is to find code that does it, and copy it... Try the "ifconfig" code for your system, or the libpcap source... Or, check if your system has the if_nameindex() function available (it seems to be available under Linux, at least)... That makes things a lot easier, by doing the really dirty work for you: obtaining the list of all local network interfaces... Once you have that, it's fairly trivial to obtain the IP associated with each one, via SIOCGIFADDR...

Add a Comment



New Questions: ERROR - CORE DUMP

From: sandy

Hi,
While running some of my C++ prgrms in solaris,I got a CORE dump
error.When I used dbx <prgrm name> core and then typed where
I got a runtime stack trace of all functions recurrsively called
One of the function on the stack was of the form
3] RWTValOrderedVector<RWCString>::insertAt(0x33764, 0x0, 0xeffd7524,
0x0, 0x0, 0x0), at 0xef718064

I want to know what are each of those hexadecimal values ?
what do those mean ?Are they addresses and parameters ?If so which
one of them are adrs and parameters ?
The actual function protoype is
3] RWTValOrderedVector<RWCString>::insertAt(size_type i,const_reference a);
Cheers,
Sandy

Add a Comment

UNIX Socket FAQ

Frequently Asked Questions about programming with sockets in UNIX environments

New Questions: How to create Sockets in UNIX

From: Grottoli

People,

I am new programmer in C++ for UNIX.
If somebody could help me out how to find a example of implementation of this...

Thanx

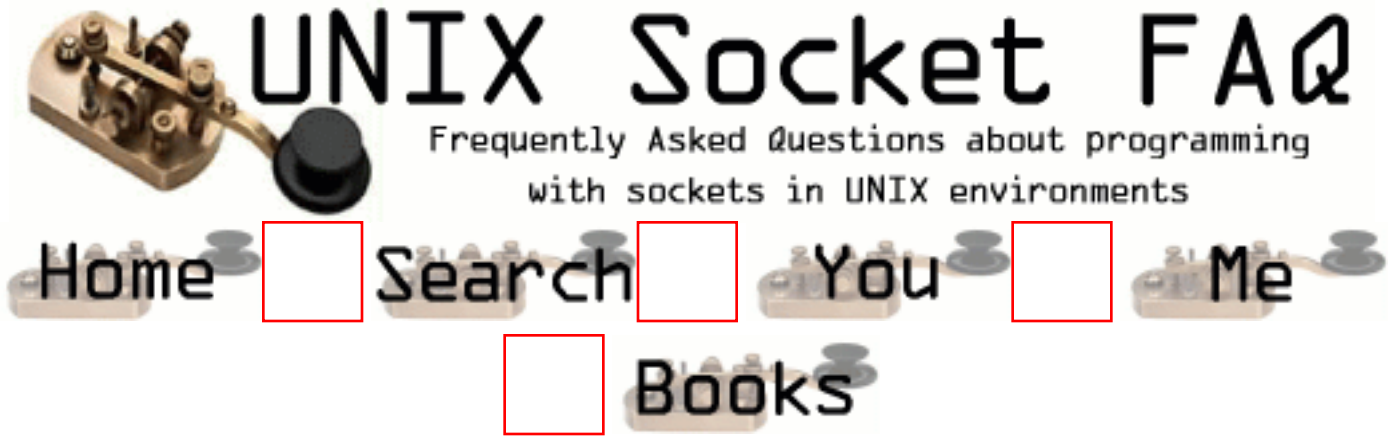
From: Jacob Harris

Hello,

If you can, I'd pick up a copy of "Unix Network Programming," which gives the C API for sockets on UNIX. You can find many of your questions answered here as well I'm sure, but that is just an excellent book to own on the topic.

Yours,
Jake

Add a Comment



New Questions: Socket and mutual exclusion

From: Jean-Sebastien

I have a server that runs many threads (linux pthread) at the same time, and it is possible that two threads write to the same socket (using 'send') at the same time.
Does socket implementation provides built-in mutual exclusion or I have to explicitly enforce it using a mutex for each opened socket, to be sure that the data from
the two 'send' call is not all messed up.

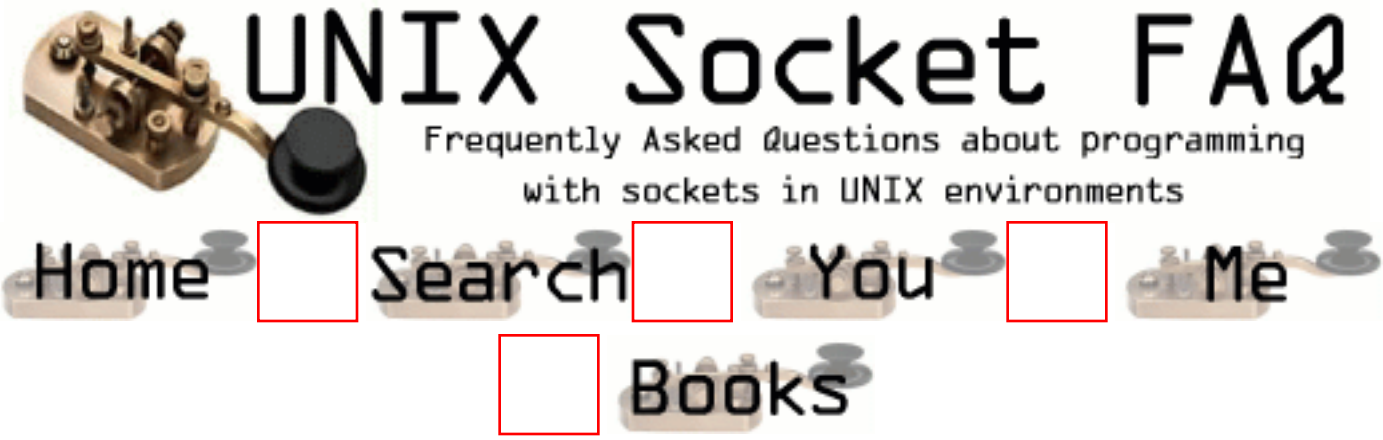
Thanx!

From: Bill Ahern

Well, you can safely write PIPE_BUF bytes to a pipe() atomically. I dunno whether this applies to sockets as well.

I had the same question, but never went ahead w/ my implementation. I believe POSIX requires a minimum PIPE_BUF of something greater than 129 bytes (which is what I needed). I think the POSIX requirement might by at least 2x's what I needed, I forgot.

Add a Comment



New Questions: Connection testing using select() and recv()

From: Chad

Hello folks,

My scenario is such that I'm writing a network client that connects from the office intranet to a remote host on the Internet through our firewall/gateway which NAT's all our traffic.

After opening a TCP connection, I would like to test it to make sure it is up before I starting reading/writing data. After a connection is established the server does not send any data and the socket (descriptor) is set non-blocking. At first I tried to test the connection success with select() and checking SO_ERROR, but this did not work as expected. The socket would select() as writable (and SO_ERROR indicated all was well) before the connection was truly established with the remote host. I imagine this was a result of the NAT'ing gateway. So I tried to use select() in combination with recv(...MSG_PEEK) to test for connection completeness. What I discovered was that I cannot tell the difference between "connection in progress" and an idle (as in no data available for reading), non-blocking connection. The only solution I could find was to have the client sleep(1) after connect()ing, to give it some time to complete. Of course this is not what I wanted, I wanted the capability of select(), something that would tell me as soon as the socket is truly ready.

Attached is some (lengthy, poorly written) code that
illustrates this. Without the commented sleep(1), the
checksock() function will report that the connection
is ready (just like an idle connection) even when no
server is listening on the other end.

Is there a better way to robustly check a TCP connection
in this situation? Besides a higher level protocol?

Thanks,
Chad

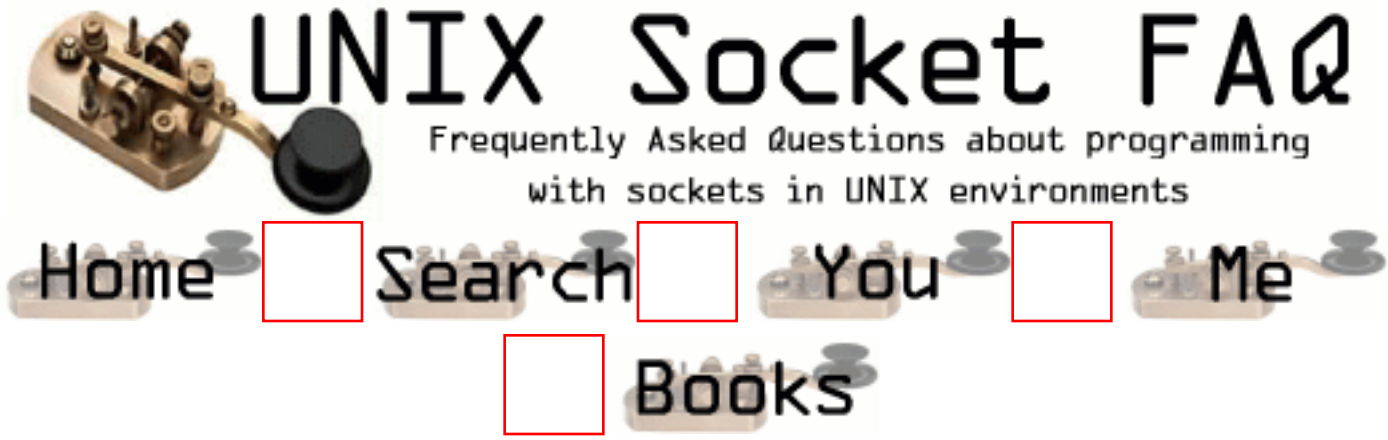
tcptest.c----------

```
#include <stdio.h>
#include <unistd.h>
#include <errno.h>
#include <netdb.h>
#include <fcntl.h>
#include <sys/socket.h>
#include <netinet/in.h>


/* Check a socket for write ability using select() and read ability
   using recv(... MSG_PEEK). Time-out values are also passed (seconds
   and microseconds) for the select() call.

   Returns 1 on success, 0 if time-out expires, and -1 on errors
*/

int checksock(int sock, int tosec, int tousec)
{
  int sret, rret;
  int ret = -1; /* default is failure */
  char testbuf[1];
  fd_set checkset;
  struct timeval to;

  FD_ZERO(&checkset);
  FD_SET(sock, &checkset);

  to.tv_sec = tosec;
  to.tv_usec = tousec;

  printf("check write ability with select()... ");
  if ( (sret = select(sock+1, NULL, &checkset, NULL, &to)) > 0 )
```

```
    ret = 1;
  else if ( sret == 0 )
    ret = 0; /* time-out expired */
  printf("select:%d\n", sret);

  printf("check read ability with recv()... ");
  if ( (rret = recv(sock, testbuf, 1, MSG_PEEK)) <= 0 ) {
    if ( errno == EAGAIN )
      ret = 1; /* no data for non-blocking IO */
    else
      ret = -1;
  }
  printf("recv:%d\n", rret);

  return ret;
}

main (int argc, char **argv)
{
  int sock;
  int sockstat;
  int bytesread = 0;
  int conret = 0;
  int fdflags;
  char *host;
  char buffer[100];
  unsigned short int port;
  struct hostent *hostinfo;
  struct sockaddr_in inet_addr;

  if (argc < 3) {
    printf("Need arguments\nUsage: tcptest host port\n");
    exit(1);
  }

  host = argv[1];
  port = atoi(argv[2]);

  if((hostinfo = gethostbyname(host)) == NULL) {
    printf("cannot resolve hostname: %s\n", host);
    exit(1);
  }

  memset(&inet_addr, 0, sizeof(inet_addr));
  inet_addr.sin_family = AF_INET;
```

```
  inet_addr.sin_port = htons(port);
  inet_addr.sin_addr = *(struct in_addr *)hostinfo->h_addr_list[0];

  if((sock = socket(AF_INET, SOCK_STREAM, 0)) < 0) {
    printf("socket: %s\n", strerror(errno));
    exit(1);
  }

  /* Set non-blocking IO */
  if ( (fdflags = fcntl(sock, F_GETFL)) >= 0)
    fcntl(sock, F_SETFL, fdflags | O_NONBLOCK);

  printf("connecting to %s:%d... ", host, port);
  conret = connect(sock, (struct sockaddr *) &inet_addr, sizeof(inet_addr));
  printf("connect:%d\n", conret);
  if (conret != 0) perror("connect");

/* sleep(1); */

  /* Check connection status */
  printf("checking socket status:\n");
  sockstat = checksock(sock, 10, 0);

  if ( sockstat < 0 ) /* select() returned error */
    printf("socket connect error to %s:%d\n", host, port);
  else if ( sockstat == 0 ) /* socket time-out */
    printf("socket connect time-out (10s) for %s:%d\n", host, port);
  else if ( sockstat > 0 ) /* socket connected */
    printf("network socket opened to %s:%d\n", host, port);

  /* Try a read() */
  bytesread = read(sock, (void *) buffer, 100);

  if (bytesread == -1)
    printf("read:%d, %s\n", bytesread, strerror(errno));
  else if (bytesread == 0)
    printf("read:%d, %s\n", bytesread, strerror(errno));

  close(sock);
}
```

Add a Comment



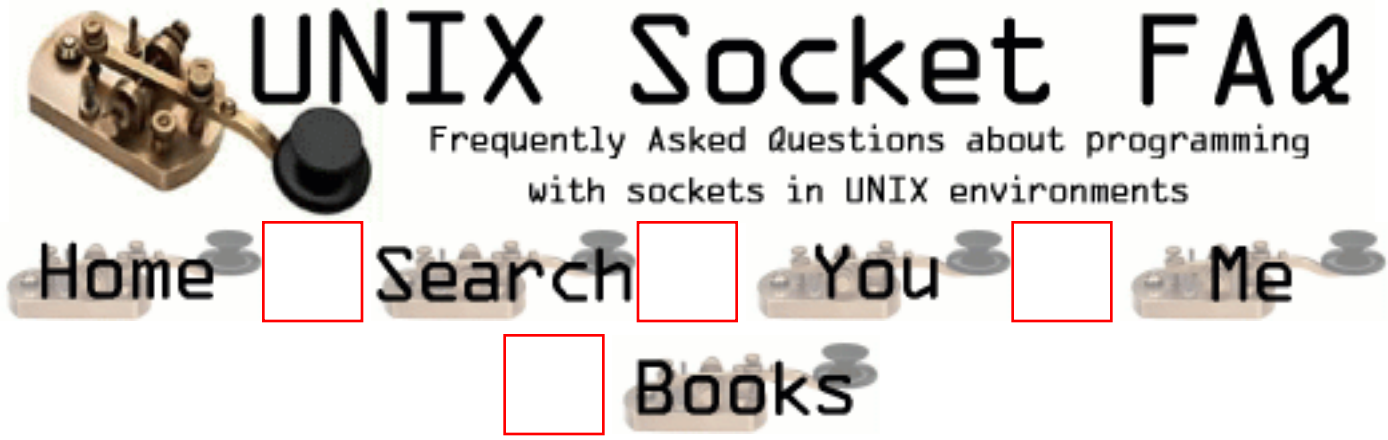
New Questions: testing TCP related APIs!!

From: Prem

Does somebody have any idea about any existing work to test the TCP related API's (like socket, connect, send etc)
I am looking for an application which does the verfies the general functionalities of these API's.

Thanks!

Add a Comment



New Questions: How can I prevent from losing data ?

From: jack wei

I create a socket with SOCK_STREAM, and use select() to check data available.
When select() returns this socket are ready for reading, I use recv() to read it.
But when the data size is larger(260120 bytes), It's wrong -- some data is lose.

```
    ...

  //nDataLength: data size.
  byte *pBuffer = (byte *)malloc(nDataLength);

  byte *pPtr = pBuffer;
  while (TRUE)
  {
    errno = 0;
    int nBytesRead = recv(fd, pPtr, 8192, MSG_DONTROUTE | MSG_DONTWAIT);
    if (nBytesRead == -1)
    {
      printf(" errno:%d", errno);
      //wait a moment.
    }
    else
    {
      pPtr += nBytesRead;
      nDataLength -= nBytesRead;
      if (nDataLength == 0)
      {
        break;
```

```
            }
            printf("recv %d bytes data.\n", nBytesRead);
            printf("Leave Length:%d.", nDataLength);
        }
    }

    ...
```

result:

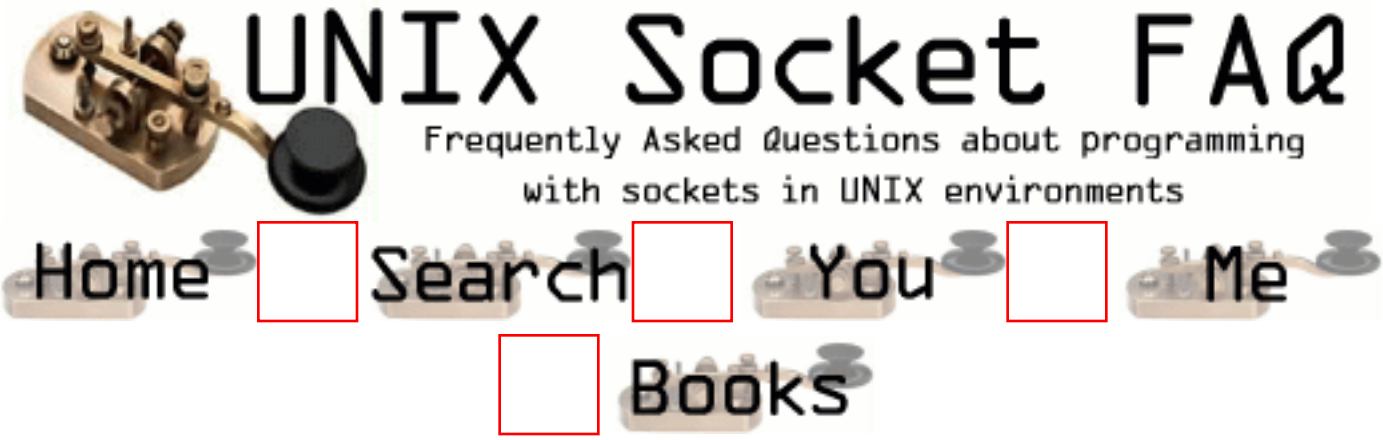
```
recv 8192 bytes data.
Leave length:243820.
recv 8192 bytes data.
Leave length:235628.
recv 8192 bytes data.
Leave length:227436.
recv 8192 bytes data.
Leave length:219244.
recv 8192 bytes data.
Leave length:211052.
recv 8192 bytes data.
Leave length:202860.
recv 8192 bytes data.
Leave length:194668.
```

HERE, 620 bytes data was lost...

```
recv 288 bytes data.
Leave length:194380.
recv -1 bytes data.
    errno=11
recv 8192 bytes data.
Leave length:186188.
recv 8192 bytes data.
Leave length:177996.
recv 8192 bytes data.
Leave length:169804.
recv 8192 bytes data.
Leave length:161612.
recv 8192 bytes data.
Leave length:153420.
recv 8192 bytes data.
Leave length:145228.
recv 8192 bytes data.
Leave length:137036.
recv 8192 bytes data.
```

Leave length:128844.
recv 8192 bytes data.
Leave length:120652.
recv 8192 bytes data.
Leave length:112460.
recv 8192 bytes data.
Leave length:104268.
recv 8192 bytes data.
Leave length:96076.
recv 8192 bytes data.
Leave length:87884.
recv 8192 bytes data.
Leave length:79692.
recv 5344 bytes data.
Leave length:74348.
recv -1 bytes data.
errno=11
Leave length:74348.
recv 8192 bytes data.
Leave length:66156.
recv 8192 bytes data.
Leave length:57964.
recv 8192 bytes data.
Leave length:49772.
recv 8192 bytes data.
Leave length:41580.
recv 8192 bytes data.
Leave length:33388.
recv 8192 bytes data.
Leave length:25196.
recv 8192 bytes data.
Leave length:17004.
recv 8192 bytes data.
Leave length:8812.
recv 8192 bytes data.
Leave length:620.
recv -1 bytes data.
errno=11 //I can not receive 620 bytes never.
...

Add a Comment



New Questions: Disconnecting from modem

From: Sunil Philip

Hi,
I have a server program which receives data from client.
The problem that iam facing is,when i disconnect my modem iam not able to know whether the client is disconnected or not.On the other hand, when i physically close the client program my server is able to detect that.
Can anybody help me plz.
sunil

Add a Comment

New Questions: Receiving packet over a specific interface?

From: Phani

Is there any way of getting to know over which interface I received a packet without binding to any particular interface. (i.e. making use of INADDR_ANY)?

Thanks

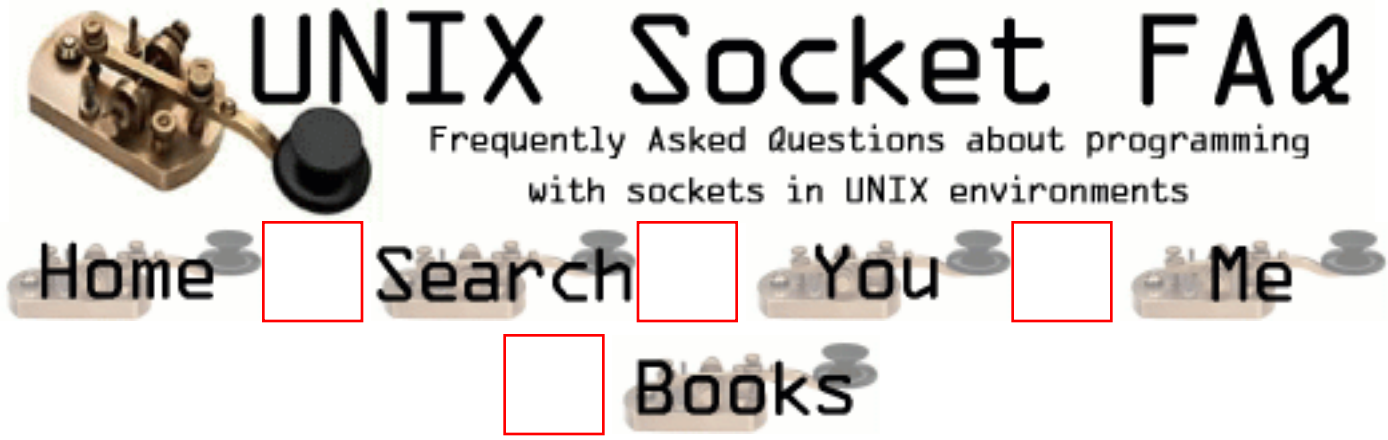
From: Rob Seace

Well, you can at least tell which IP it came in on, sure... (After that, you can do the work of mapping the IP to a specific interface, if that's what you really need...)

Just call getsockname() on the connected socket FD, and it should give you the information about what IP your end of the connection is using (and, thereby, what interface it came in on)...

I'm assuming TCP, here... If you're talking about UDP, then I'm not certain, but I doubt that'd work (since, obviously, there IS no connection there)... With UDP, you might indeed be forced to have separate listening FDs for all local IPs, in order to determine which one it came in on... But, for TCP, the above getsockname() trick should work just fine... (At least, I have used it before, and know that it at least works fine on Linux...)

Add a Comment



New Questions: buffer question.

From: Jessica

I am writing a socket, which just sit there waiting for income data(so I decided to use UDP), and then display all the income data. I use sock_dgram. I have a question about how to use recvfrom function. The income data maybe very large(I have no idea how large it will be), if it is larger than the buffer size in recvfrom function, it will return error message: errno EMSGSIZE triggered. Do you have a solution for this.

Here goes my code:

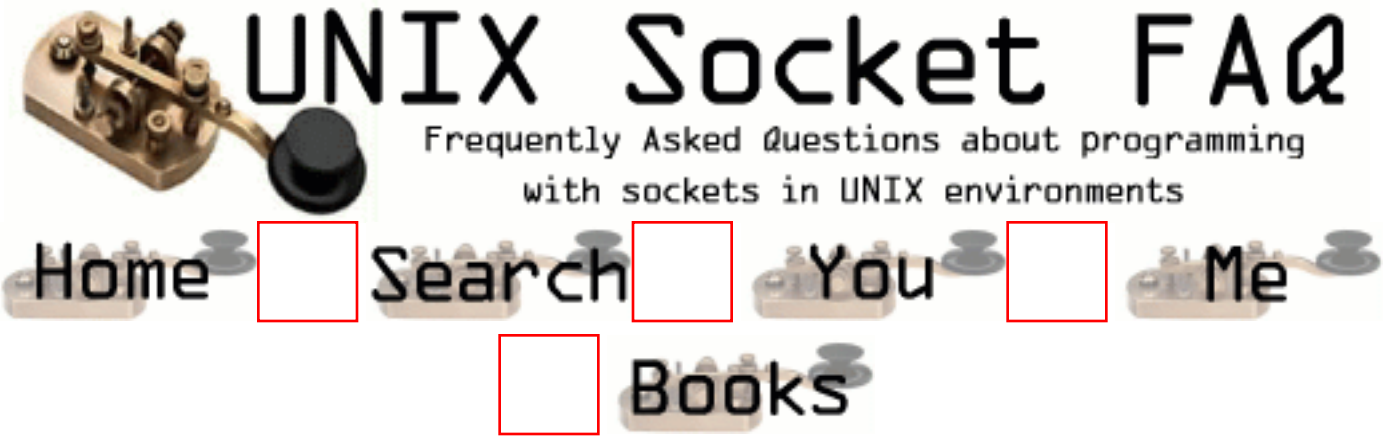
```
int main()
{
  int sock, length, fromlen, n;
  struct sockaddr_in server;
  struct sockaddr_in from;
  char buf[4096];

  sock=socket(AF_INET, SOCK_DGRAM, 0);
  if (sock < 0) error("ERROR on Opening socket");
  length = sizeof(server);
  bzero(&server,length);
  server.sin_family=AF_INET;
  server.sin_addr.s_addr=INADDR_ANY;
  server.sin_port=htons(PORT);
  if (bind(sock,(struct sockaddr *)&server,length)<0)
   error("ERROR on binding");
  fromlen = sizeof(from);

  while (1) {
     n=recvfrom(sock,buf,buffSize,0,(struct sockaddr *) &from,&fromlen);
```

```
    if(n<0) error("ERROR on recvfrom");
    printf("Here is the data: %s", buf);
  }
  return 0;
}
```

From: Melih SARICA

Most operating systems max buffer size for a packet is limited to 8192 by default.
replace your 4096 with 8192. This makes it faster. And then make a loop until recvfrom returns 0.
Remember that unlike TCP packets, sometimes UDP packets can be lost. If this is a mission-critical program then make your own ACK system.
good luck...
-Melih

Add a Comment



New Questions: 6 second delay ...

From: Eli Sidwell

All,

I have program that moves data from a Unix box to a windows box socket to socket. The problem is that I experience a 6 second delay when the windows box sends data to the Unix port.

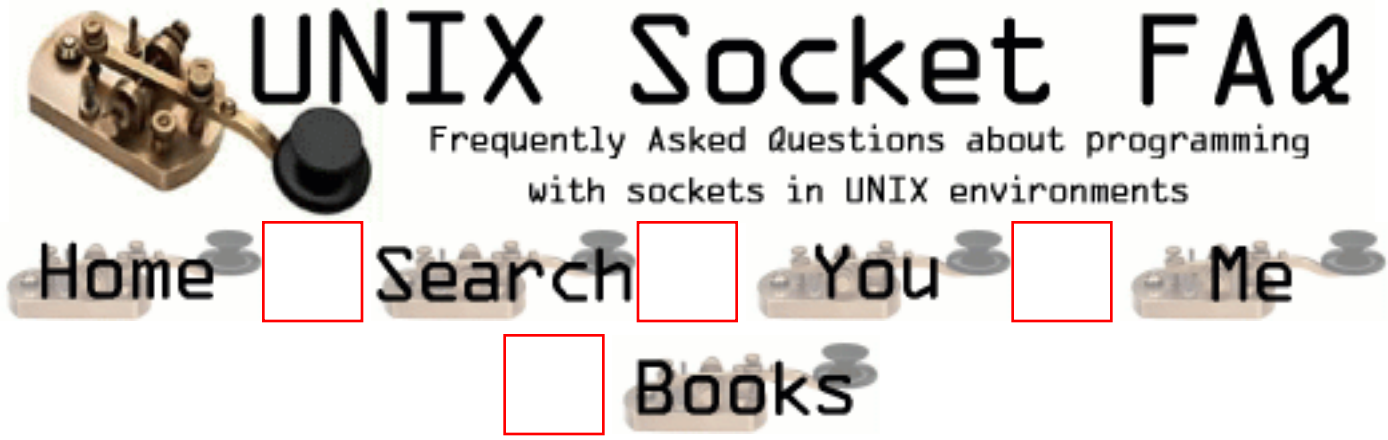
This problem has been present for some time, but I was told that before we had Nortel network switches and routers installed that this problem did not exist.
My network guru tells me that nothing should be wrong and he sees a bunch of NetBeui traffic.

I believe the delay is encountered when the Windows box transmits back to Unix

Any help is appreciated.

Sid...

Add a Comment



New Questions: Ho to make blocking send with UDP socket?

From: DrWatson

I need blocking "send" call with UDP socket on Linux. What should I do?

From: Loco

Hi,

All your socket calls are blocking by default. If you haven't changed your file descriptor's mode then you are issuing blocking calls.

Blocking write means the call will wait until there is enough room in the buffer to hold the data you are trying to send, not until the other party gets the data.

:D (HAL)

From: DrWatson

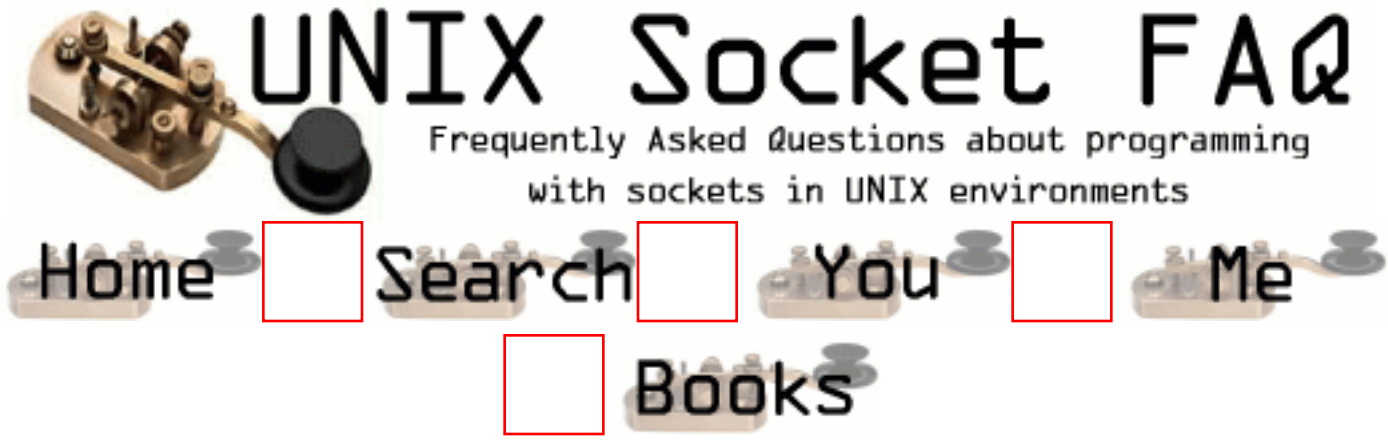
That's right... This is theory. :-)
On my Linux Mandrake8 blocking call to sendto(...) doesn't wait for enough room in socket buffer. Therefore endless loop:

```
long bytesSent = 0;
while (true) {
   bytesSent+=sendto(...);
   if (some_condition) break;
}
```

gives me transfer rate about 80 MB/sec!
I'm sending data to another host, and phisical traffic, captured by sniffer is about 8 MB/sec (this

is OK for 100 Mbit adapter). The same code on NT works properly :-(
Maybe it is Linux's problem? I'm not enough experinced in Sockets programing on Linux.
Any ideas?

Add a Comment

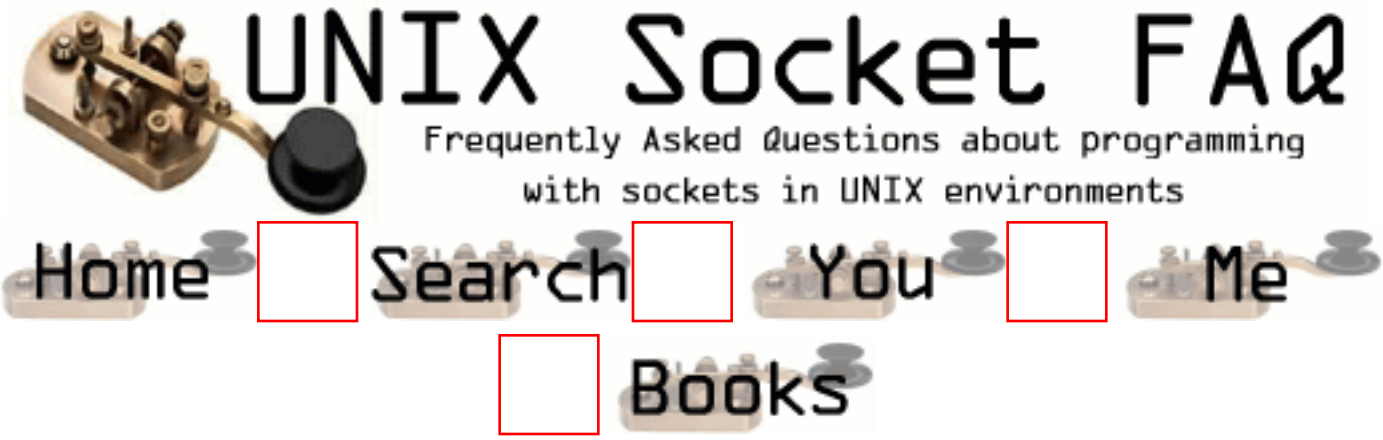


New Questions: TCP Sockets with pthreads

From: Don

Is it possible to have one thread do a blocking read from a TCP socket while another thread does a blocking write from the same TCP socket? Will the separate threads work completely independently or might one thread block, waiting for socket resources being used by the other thread?

Add a Comment



New Questions: Unable to catch SIGCHLD on a HP workstation.

From: Jyoti

Hi!!,
while running the following on a HP Work station, I get the following error message everytime the child process exits.

Request from a ckient receiced
Waiting fro my childs death
Pid 27218 was killed due to stack growth failure.
Possible causes: insufficient memory or swap, or stack size exceeded maxssize.

Pid 27218 was killed due to failure in writing the signal context.
vxfs: mesg 001: vx_nospace - /dev/vg00/lvol5 file system full (1 block extent)
Memory fault

Kindly help..
Thanks in Advance..

Thanking you,
Jyoti

Add a Comment

New Questions: Unable to catch SIGCHLD on a HP workstation.

From: Jyoti

Hi!!,
while running the following on a HP Work station, I get the following error message everytime the child process exits.

Request from a ckient receiced
Waiting fro my childs death
Pid 27218 was killed due to stack growth failure.
Possible causes: insufficient memory or swap, or stack size exceeded maxssize.

Pid 27218 was killed due to failure in writing the signal context.
vxfs: mesg 001: vx_nospace - /dev/vg00/lvol5 file system full (1 block extent)
Memory fault

Kindly help..
Thanks in Advance..

Thanking you,
Jyoti

```
#include<stdio.h>
#include<sys/types.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<arpa/inet.h>
#include<signal.h>
#define SERV_TCP_PORT 5000
```

```
 void sig_handler(int)
   { int status; pid_t pid;
   signal(SIGCHLD,sig_handler);
   printf("Child killed\n");
   if((pid=wait(&status))<0)
    {
    perror("wait eror\n");
    printf("pid=%d\n",pid);
    }
  return;
  }
main(argc,argv)
int argc;
char *argv[];
{
int sockfd,childsockfd,clilen,childpid;
struct sockaddr_in cli_addr,serv_addr;
/* Open a TCP Socket now...*/
if((sockfd=socket(AF_INET,SOCK_STREAM,0)) < 0)
perror("Server: Cant open the socket\n");
/* Now initialize the "sockaddr_in" structure...*/
bzero((char *) &serv_addr,sizeof(serv_addr));
serv_addr.sin_family=AF_INET;
serv_addr.sin_addr.s_addr=htonl(INADDR_ANY);
serv_addr.sin_port=htons(SERV_TCP_PORT);

/* Now bind the socket to address.. */
if(bind(sockfd, (struct sockaddr *) &serv_addr,sizeof(serv_addr))<0)
perror("Server: Cant bind the local address\n");
if(signal(SIGCHLD,sig_handler)==-1)
perror("SIFCHLD: error\n");

/*Now listen for the client request....At most 5 in thee queue */
listen(sockfd,5);
printf("Request from a client receiced\n");
/* Now wait indefinitely for the cliennt Requests */
while(1)
{
clilen= sizeof(cli_addr);
childsockfd=accept(sockfd, (struct sockaddr *) &cli_addr, &clilen);
if(childsockfd<0)
perror("Server: accept Eror\n");

if((childpid=fork())<0)
perror("Server: Fork Error\n");
```

```
if (childpid==0)
{
if(execlp("/usr/bin/X11/xterm","xterm",(char *)0)<0)
{
perror("Exec Error\n");
exit(0);
}

/* The control should never reach here..... */
else
perror("New Child Session Started\n");
exit(0);
}
printf("Waiting fro my childs death\n");
}
if((close(childsockfd))<0)
printf("Error closing the socket\n");
exit(0);
}
```

Add a Comment

New Questions: Unable to catch SIGCHLD on a HP workstation.

From: Jyoti

Hi!!,
while running the following on a HP Work station, I get the following error message everytime the child process exits.

Request from a ckient receiced
Waiting fro my childs death
Pid 27218 was killed due to stack growth failure.
Possible causes: insufficient memory or swap, or stack size exceeded maxssize.

Pid 27218 was killed due to failure in writing the signal context.
vxfs: mesg 001: vx_nospace - /dev/vg00/lvol5 file system full (1 block extent)
Memory fault

Kindly help..
Thanks in Advance..

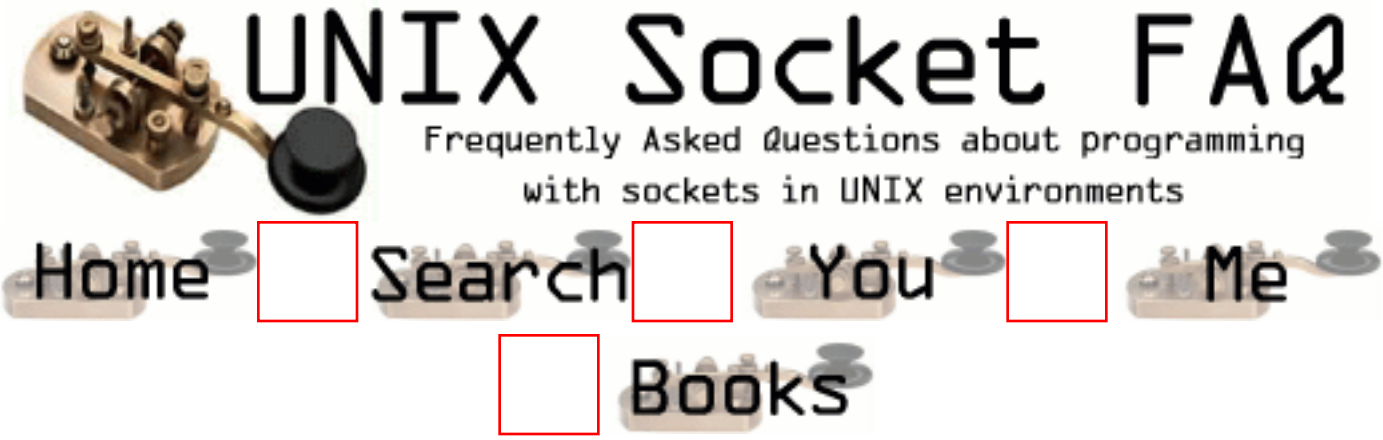
Thanking you,
Jyoti

```
#include<stdio.h>
#include<sys/types.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<arpa/inet.h>
#include<signal.h>
#define SERV_TCP_PORT 5000
```

```
 void sig_handler(int)
   { int status; pid_t pid;
   signal(SIGCHLD,sig_handler);
   printf("Child killed\n");
   if((pid=wait(&status))<0)
    {
    perror("wait eror\n");
    printf("pid=%d\n",pid);
    }
  return;
  }
main(argc,argv)
int argc;
char *argv[];
{
int sockfd,childsockfd,clilen,childpid;
struct sockaddr_in cli_addr,serv_addr;
/* Open a TCP Socket now...*/
if((sockfd=socket(AF_INET,SOCK_STREAM,0)) < 0)
perror("Server: Cant open the socket\n");
/* Now initialize the "sockaddr_in" structure...*/
bzero((char *) &serv_addr,sizeof(serv_addr));
serv_addr.sin_family=AF_INET;
serv_addr.sin_addr.s_addr=htonl(INADDR_ANY);
serv_addr.sin_port=htons(SERV_TCP_PORT);

/* Now bind the socket to address.. */
if(bind(sockfd, (struct sockaddr *) &serv_addr,sizeof(serv_addr))<0)
perror("Server: Cant bind the local address\n");
if(signal(SIGCHLD,sig_handler)==-1)
perror("SIFCHLD: error\n");

/*Now listen for the client request....At most 5 in thee queue */
listen(sockfd,5);
printf("Request from a client receiced\n");
/* Now wait indefinitely for the cliennt Requests */
while(1)
{
clilen= sizeof(cli_addr);
childsockfd=accept(sockfd, (struct sockaddr *) &cli_addr, &clilen);
if(childsockfd<0)
perror("Server: accept Eror\n");

        if((childpid=fork())<0)
perror("Server: Fork Error\n");
```

```
if (childpid==0)
          {
           if(execlp("/usr/bin/X11/xterm","xterm",(char *)0)<0)
{
         perror("Exec Error\n");
exit(0);
         }
/* The control should never reach here..... */
else
perror("New Child Session Started\n");
exit(0);
}
printf("Waiting fro my childs death\n");
}
if((close(childsockfd))<0)
printf("Error closing the socket\n");
exit(0);
        }
```

Add a Comment



New Questions: Try to make a secure conection

From: ptic

Hello, i have made conection with socket in C, i want to know how to secure the conection, only register client can connect to the server (with a login and password)

thanks

Add a Comment

New Questions: Error while compiling

From: Jyoti

Hi!!
while compiling my C programme with the following switches,

cc -wall -o signal signal.c
on a HP workstation, the following error message appears,

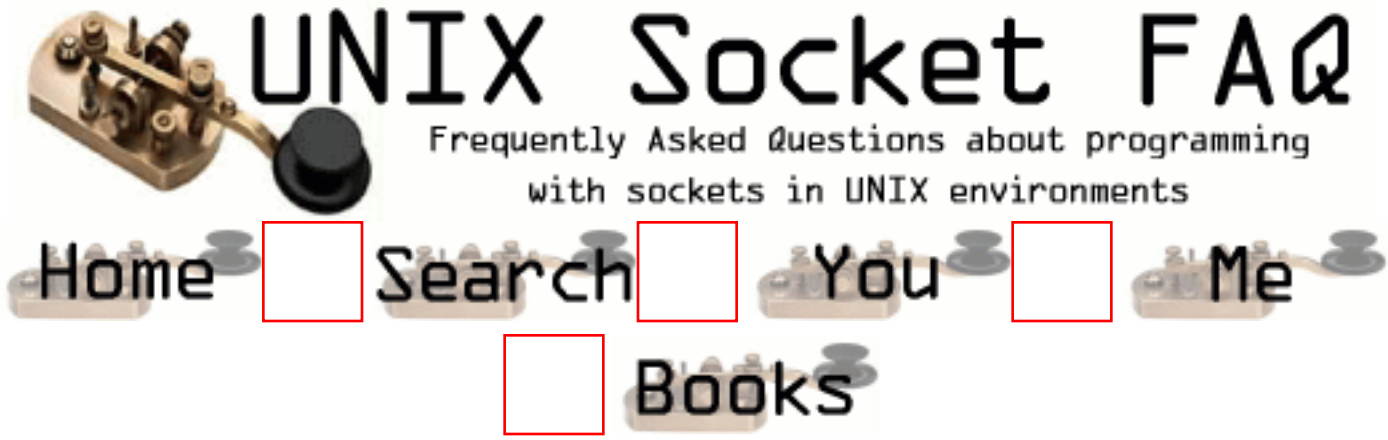
/usr/ccs/bin/ld: Can't create signal
/usr/ccs/bin/ld: Text file busy

while creating the executable with any other name, the operation is successful

Kindly Help..

Thanking you,
Jyoti

Add a Comment



New Questions: ftp server bind with REUSEADDR ?

From: Brent Thomas

hi folks, maybe someone can clarify this a bit for me...
I've got an ftp server (running on a >1024 port) which works
pretty well but under high volume it returns bind errors.
its running as a daemon (forked by inetd)
In the port processing:
I call socket
>>> datafd = socket(AF_INET, SOCK_STREAM, 0);
then i set the reuseaddr
>>> if (setsockopt(datafd, SOL_SOCKET, SO_REUSEADDR,
>>> &on, sizeof(on)) < 0) {
then i set the dataconn to my control conn
>>> dataconn = cntrlconn;
and overlay the correct port
>>> dataconn.sin_port = htons(ntohs(ctrlconn.sin_port) - 1);
then i call bind...i had to wrap this in a loop because it
would fail
>>> while (bind(datafd, (struct sockaddr *)&dataconn, sizeof(dataconn)) < 0)
and i limit it to 18 tries at 5 sec intervals = 90 sec total
per a comment in the stevens tcp book

I'm thinking my specification of the dataconn by
equating it to the cntrlconn and just resetting the
port isnt making the unique 5tuple socket definition because
the behavior I get is as follows:

If I spawn 5 simultaneous ftp sessions to this server
they process fifo with all the pending connnections hitting

the bind failure until the active connection finishes the
retr (for retrieving a file).

If i spawn 5 simultaneous ftp sessions to the system ftp
server (hp ux 11) and retrieve a file they seem to
process simultaneously (cant tell really because I cant
see if they're binding) but it seems to work smoother.

I guess my main question is 1) IS there a difference between
my server and the system ftp server or is it blocking in the bind
as well but just hiding it better? and 2) if there is a
problem with my logic (i realize there is only the 1 well known
socket really availible so they do have to block somewhat)
is there some suggestions as to how better to specify the
socket or dataconn used in the bind so it doesnt fifo block.

my concern is that if someone were transfering a large file with
my implementation it would essentially block for 90 sec and
then abort while the file transfer proceeded while from
a test with the system implementaiton it looks like it
spawns all 5 simultaneouly (not blocking in bind) but just
transfers very slowly.

thanks for any suggestions...
please reply here and if you feel like it
also cc bthomas@optiosoftware.com

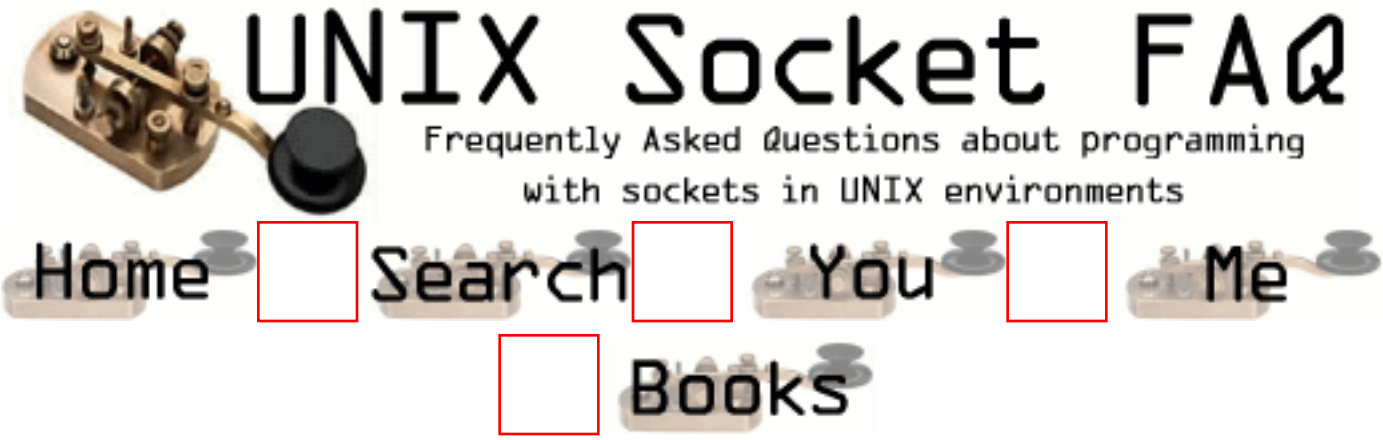
From: Brent Thomas

update:

to solve the problem we've removed the bind step
on the doport. Apparently no need to actually
bind here because we're going to perform the
active open in the connect step to the clients specified
port.

Dont see any holes in this at this time.

Add a Comment



New Questions: Specific concurrent server port

From: Treke

Hello.

I have this problem: I would like to make (port it from
WinNT) a TCP/IP server having these properties:

- able to handle any number of client sockets
- server spawns two different processes: Sender, AcceptReceiver
  (uses select), so I don't want to fork and fork...
- without threads, to be portable to older systems

Maybe this problems occur because I lack Unix experience.
What I cannot determine is:
How can the Sender process send data to socket accepted
by AcceptReceiver process?
(the client socket is accepted after fork).
Can I sent the socket data (handler, sockaddr structure)
to Sender and than bind it ? will it work ?
<huh, with threads, life is easy, but i have it forbidden>..

Thanx in advance

Ed Toth (Treke)

From: Treke

Huh I forgot (so it has a wrong meaning now):
...send the socket data to sender via NAMED PIPE ...

sorry

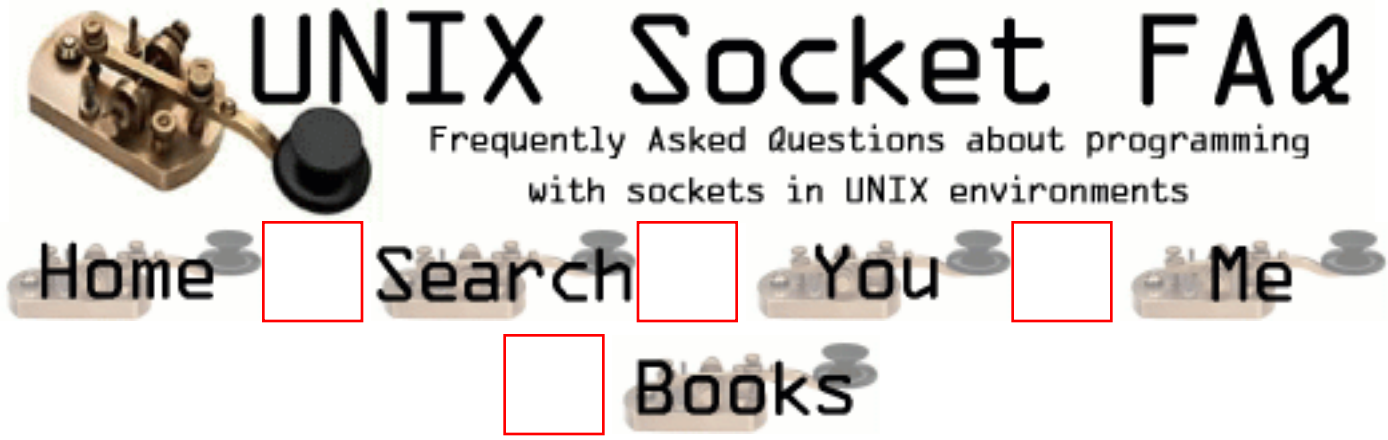
From: Rob Seace

So, you have two separate processes, and you open a new
socket (via accept()) in one, and which the OTHER one to be
able to then write to it? Well, ok, you could do that by
just passing file descriptors from one to the other... To
do that, look into sendmsg()/recvmsg(), using SCM_RIGHTS...
(On my system, an example of passing file descriptors this
way can be found in the "man cmsg" page...) You'd need a
separate Unix-domain connection between the two processes,
to pass the FDs over, as well...

Though, if your whole goal here is portability (your reason
for avoiding the threads), I'm not sure if you want to rely
on the FD-passing, either... I'm not sure how standard, or
widely-implimented that is, on various systems...

Aside from that though, you don't really have many other
choices, I don't think... Aside from simply restructuring
your program, if that's possible... Ie: have the same
process that accept()'s (or, a direct child of it) do the
sending/receiving of data...

Add a Comment



New Questions: socket deadlock

From: phil

I have a situation where 2 connected sockets both end up blocking in a call to bsd_recvfrom (according to debug):

This is the client application:

New program sebsm140601 (process p1) grabbed
SUSPENDED p1 [__bsd_recvfrom()]
0xbfa4d0ff (__bsd_recvfr+11:) jb 0x1 <bfa4d102> [__bsd_recvfrom]
debug> stack
Stack Trace for p1, Program sebsm140601
[14] __bsd_recvfrom(0x8,0x415b5bbc,0x1000,0,0,0) [0xbfa4d0ff]
[13] recv(0x8,0x415b5bbc,0x1000,0) [0x806cf9c]

This is the server application:

New program sebsm140603 (process p1) grabbed
SUSPENDED p1 [__bsd_recvfrom()]
0xbfa4d0ff (__bsd_recvfr+11:) jb 0x1 <bfa4d102> [__bsd_recvfrom]
debug> stack
Stack Trace for p1, Program sebsm140603
[7] __bsd_recvfrom(0x9,0x12574d3c,0x1000,0,0,0) [0xbfa4d0ff]
[6] recv(s=9,buf=0x12574d3c,len=4096,flags=0) [sebscnrt_OCoreObj.c@0x805d06b]

This occurs regularly, though at unpredictable (to me) instances. I have lots of other information including a log of all socket functions called by the programs, along with the corresponding tcpdump output. I will post these as needed. The basic behavior of the programs is:

1) client connects to the server
2) server accepts
3) client calls send() and immediately falls into a recv() at which point it blocks waiting for the response
4) the server returns from select(), gets the client's request via recv() and immediately calls send(), thereby completing the server part of the cycle - it then returns to select() for the next request
5) the client returns from recv() with the server's response, thereby completing the client part of the cycle.

The same socket is used for sending and receiving. The client has at least 2 connections to the server. The server has many different applications connecting to it, all following the same "protocol" outlined above.

I fail to see how this can occur. Any ideas.

Thanks in advance...

Phil

From: phil

I forgot to mention that the client and the server are always both on the same machine, and the sockets are INET style, not UNIX.

Also, here is a bit more info for those of you who are too shy to ask.

Server port = 2141
Client port = 22679

TCPDUMP:

23:06:08.20 127.0.0.1.22679 > 127.0.0.1.2141: P 12517:13113(596) ack 12517 win 16384 (DF) (ttl 64, id 56117)
23:06:08.26 127.0.0.1.2141 > 127.0.0.1.22679: P 12517:13113(596) ack 13113 win 16384 (DF) (ttl 64, id 56155)
23:06:08.30 127.0.0.1.22679 > 127.0.0.1.2141: P 13113:13709(596) ack 13113 win 16384 (DF) (ttl 64, id 56184)
23:06:08.31 127.0.0.1.2141 > 127.0.0.1.22679: P 13113:13709(596) ack 13709 win 16384 (DF) (ttl 64, id 56203)
23:06:08.51 127.0.0.1.22679 > 127.0.0.1.2141: . ack 13709 win 16384 (DF) (ttl 64, id 56374)
23:06:19.81 127.0.0.1.22679 > 127.0.0.1.2141: P 13709:14305(596) ack 13709 win 16384 (DF) (ttl 64, id 10792)
23:06:19.85 127.0.0.1.2141 > 127.0.0.1.22679: P 13709:14305(596) ack 14305 win 16384 (DF) (ttl 64, id 10844)

23:06:19.88 127.0.0.1.22679 > 127.0.0.1.2141: P 14305:14901(596) ack 14305 win 16384 (DF) (ttl 64, id 10939)
23:06:19.92 127.0.0.1.2141 > 127.0.0.1.22679: P 14305:14901(596) ack 14901 win 16384 (DF) (ttl 64, id 11060)
23:06:20.12 127.0.0.1.22679 > 127.0.0.1.2141: . ack 14901 win 16384 (DF) (ttl 64, id 11702)
23:06:29.85 127.0.0.1.22679 > 127.0.0.1.2141: P 14901:15497(596) ack 14901 win 16384 (DF) (ttl 64, id 47716)
23:06:29.87 127.0.0.1.2141 > 127.0.0.1.22679: P 14901:15497(596) ack 15497 win 16384 (DF) (ttl 64, id 47775)
23:06:30.07 127.0.0.1.22679 > 127.0.0.1.2141: . ack 15497 win 16384 (DF) (ttl 64, id 48603)
23:06:43.85 127.0.0.1.22679 > 127.0.0.1.2141: P 15497:16093(596) ack 15497 win 16384 (DF) (ttl 64, id 49324)
23:06:43.96 127.0.0.1.2141 > 127.0.0.1.22679: P 15497:16093(596) ack 16093 win 16384 (DF) (ttl 64, id 49921)
23:06:44.03 127.0.0.1.22679 > 127.0.0.1.2141: P 16093:16689(596) ack 16093 win 16384 (DF) (ttl 64, id 50264)
23:06:44.11 127.0.0.1.2141 > 127.0.0.1.22679: P 16093:16689(596) ack 16689 win 16384 (DF) (ttl 64, id 50685)
23:06:44.31 127.0.0.1.22679 > 127.0.0.1.2141: . ack 16689 win 16384 (DF) (ttl 64, id 51658)
23:06:55.70 127.0.0.1.22679 > 127.0.0.1.2141: P 16689:17285(596) ack 16689 win 16384 (DF) (ttl 64, id 48496)
23:06:55.90 127.0.0.1.2141 > 127.0.0.1.22679: . ack 17285 win 15788 (DF) (ttl 64, id 49564)

The processes blocked at this point. When the operators notice the processes are hung they kill the client...

23:18:01.79 127.0.0.1.22679 > 127.0.0.1.2141: F 17285:17285(0) ack 16689 win 16384 (DF) (ttl 64, id 40982)
23:18:01.79 127.0.0.1.2141 > 127.0.0.1.22679: . ack 17286 win 15788 (DF) (ttl 64, id 40983)
23:18:03.46 127.0.0.1.2141 > 127.0.0.1.22679: P 16689:17285(596) ack 17286 win 16384 (DF) (ttl 64, id 49594)
23:18:03.46 127.0.0.1.22679 > 127.0.0.1.2141: R 1648373204:1648373204(0) win 0 (ttl 64, id 49599)

APPLICATION LOG:

Note - the application logs each call to socket functions immediately upon return from that function via a fifo to another application which writes the log message to a file. Also, the tcpdump time is an hour behind the application time - go figure...

Format: [pid] date application: tcp_oreration socket_fd send_port:recv_port

[24854] 01-06-01_00:06:08 SEBSM140601: send 12 22679:2141
[22831] 01-06-01_00:06:08 SEBSM140603: recv 9 2141:22679
[22831] 01-06-01_00:06:08 SEBSM140603: send 9 2141:22679

[24854] 01-06-01_00:06:08 SEBSM140601: recv 12 22679:2141
[24854] 01-06-01_00:06:08 SEBSM140601: send 12 22679:2141
[22831] 01-06-01_00:06:08 SEBSM140603: recv 9 2141:22679
[22831] 01-06-01_00:06:08 SEBSM140603: send 9 2141:22679
[24854] 01-06-01_00:06:08 SEBSM140601: recv 12 22679:2141
[24854] 01-06-01_00:06:19 SEBSM140601: send 12 22679:2141
[22831] 01-06-01_00:06:19 SEBSM140603: recv 9 2141:22679
[22831] 01-06-01_00:06:19 SEBSM140603: send 9 2141:22679
[24854] 01-06-01_00:06:19 SEBSM140601: recv 12 22679:2141
[24854] 01-06-01_00:06:19 SEBSM140601: send 12 22679:2141
[22831] 01-06-01_00:06:19 SEBSM140603: recv 9 2141:22679
[22831] 01-06-01_00:06:19 SEBSM140603: send 9 2141:22679
[24854] 01-06-01_00:06:19 SEBSM140601: recv 12 22679:2141
[24854] 01-06-01_00:06:29 SEBSM140601: send 12 22679:2141
[22831] 01-06-01_00:06:29 SEBSM140603: recv 9 2141:22679
[22831] 01-06-01_00:06:29 SEBSM140603: send 9 2141:22679
[24854] 01-06-01_00:06:29 SEBSM140601: recv 12 22679:2141
[24854] 01-06-01_00:06:43 SEBSM140601: send 12 22679:2141
[22831] 01-06-01_00:06:43 SEBSM140603: recv 9 2141:22679
[22831] 01-06-01_00:06:43 SEBSM140603: send 9 2141:22679
[24854] 01-06-01_00:06:43 SEBSM140601: recv 12 22679:2141
[24854] 01-06-01_00:06:44 SEBSM140601: send 12 22679:2141
[22831] 01-06-01_00:06:44 SEBSM140603: recv 9 2141:22679
[24854] 01-06-01_00:06:44 SEBSM140601: recv 12 22679:2141
[22831] 01-06-01_00:06:44 SEBSM140603: send 9 2141:22679
[24854] 01-06-01_00:06:55 SEBSM140601: send 12 22679:2141

The processes blocked at this point.

From: phil

Hmmmm... no takers yet. Is it because of too much data? Or its been read and nobody has any ideas? If its the former, here is a more concice outline:

Q = query
R = response

1: from the code's point of view

client:
send Q1 596 bytes
return from send
recv
wait in bsd_recvfrom
return from recv R1 596 bytes

[process response]

send Q2 596 bytes
return from send
recv
wait in bsd_recvfrom
return from recv R2 596 bytes

[process response]

send Q3 596 bytes
return from send
recv
wait in bsd_recvfrom

server:
select

return from select
recv
return from recv Q1 596 bytes
send R1 596 bytes
return from send
select

return from select
recv
return from recv Q2 596 bytes
send R2 596 bytes
return from send
select

return from select
recv
wait in bsd_recvfrom

2: from the application log point of view [date operation port:port]
and the tcpdump point of view [date source > destination: flag ...]

Q1
01-06-01_00:06:43 send 22679:2141
01-06-01_00:06:43 recv 2141:22679
23:06:43.85 127.0.0.1.22679 > 127.0.0.1.2141: P 15497:16093(596) ack 15497 win 16384 (DF) (ttl 64, id 49324)

R1
01-06-01_00:06:43 send 2141:22679
01-06-01_00:06:43 recv 22679:2141
23:06:43.96 127.0.0.1.2141 > 127.0.0.1.22679: P 15497:16093(596) ack 16093 win 16384 (DF) (ttl 64, id 49921)

Q2
01-06-01_00:06:44 send 22679:2141
01-06-01_00:06:44 recv 2141:22679
23:06:44.03 127.0.0.1.22679 > 127.0.0.1.2141: P 16093:16689(596) ack 16093 win 16384 (DF) (ttl 64, id 50264)

R2
01-06-01_00:06:44 recv 22679:2141
01-06-01_00:06:44 send 2141:22679
23:06:44.11 127.0.0.1.2141 > 127.0.0.1.22679: P 16093:16689(596) ack 16689 win 16384 (DF) (ttl 64, id 50685)
23:06:44.31 127.0.0.1.22679 > 127.0.0.1.2141: . ack 16689 win 16384 (DF) (ttl 64, id 51658)

Q3
01-06-01_00:06:55 send 22679:2141
23:06:55.70 127.0.0.1.22679 > 127.0.0.1.2141: P 16689:17285(596) ack 16689 win 16384 (DF) (ttl 64, id 48496)
23:06:55.90 127.0.0.1.2141 > 127.0.0.1.22679: . ack 17285 win 15788 (DF) (ttl 64, id 49564)

At this point, both client and server are waiting in bsd_recv_from.
Note that the last segment is not from the application, and the window has been reduced by 596.

Also, if anyone knows why and when acks are sent (with no flag or data), I'd like to know.

Thanks.

From: Rob Seace

Hmmmm... That's a puzzler... Well, the ACKs with no data would be sent when there's no outgoing buffered data to send along with them... TCP has to ACK the incoming data within some reasonable time limit; it tries to piggy-back them onto data if it can, but if there's none ready to go, it has to just waste a packet to send out the ACK by itself...

Well, one thing I notice is that in your initial post, the stack trace info shows an FD of 8 being blocked on in the client... But, I see no mention of an FD 8 in your posted

log info, just FD 9 and 12... Is the client maybe trying to recv() from the wrong FD, or something? Or, is this just info from different instances, such that the FDs are no longer the same? If you're sure they're both really blocking on recv() on the correct FDs, then I'm not sure what to make of the situation... It appears, from everything above, that the last sent message DOES make it to the server, but the server simply never reads it... The TCP stack apparently ACKs it, though... So, I don't know why on earth recv() wouldn't grab it... *shrug*

From: phil

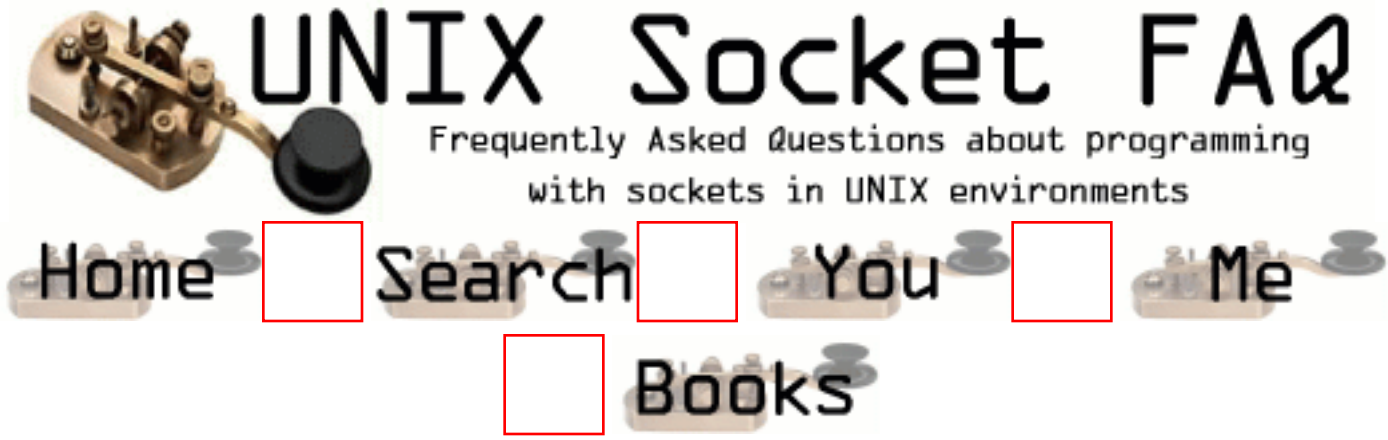
Thanks for your reply!!! Its greatly appreciated.

About the client FD discrepency, the debug stack dump and the application log extracts were taken from different days, but the stack dump is ALWAYS the same.

The clients open/close connections up to 4 times per second rather that maintain a single connection - totally unnecessary, but the code's so badly written (its not mine...) that its imossible to change this - the TCP stuff is ok, but its mish-mashed with a ton of other stuff. Even though, theoretically, I guess that that should not cause problems. I will try using AF_UNIX sockets instead of AF_INET - *shrug*.

Thanks again.

Add a Comment



New Questions: Select problem with NT and Solaris sockets

From: Jacob Harris

I seem to have a rather interesting problem on my hands. I have a centralized service that had to be written in NT that fields requests from many separate Solaris processes with threads. Since it is Windows NT, the service occaisionally crashes, and sometimes I find that the Solaris client hangs on the read (despite the use of a select). The code in the client looks something like the following:

```
  fd_set infds, errfds;
```

From: Jacob Harris

Ooops,

Stupid Web browser! The code is as follows:

```
fd_set infds, errfds;
FD_ZERO(&infds);
FD_SET(sock, &infds);
FD_ZERO(&errfds);
FD_SET(sock, &errfds);

timeval tv;
tv.tv_sec = timeout_value;
tv.tv_usec = 0;

int selected = ::select(sock+1, &infds, 0, &errfds, &tv);
if (selected <= 0 || FD_ISSET(sock, &errfds))
  need_io_stop=1;
```

```
if (!need_io_stop)
  count = ::read(sock, buffer, maxlength);
```

As near as I can tell, it seems to get stuck on the read. It seems that the remote service crashes between the select call and the actual read. Select indicates there is data to be read, but the read itself blocks waiting for data (it seems that the socket does not receive a close). The stack trace reveals it's sitting in _poll, although I can't seem to find who's calling that (the read or the select). It's obviously a race condition, but what can I do to avoid such a situation? Would peeking at the socket's data help? Or would that fall victim to the same race condition?

Yours,
Jake

From: Rob Seace

No, that would be the select() that it's blocking on...
Both poll() and select() are similar, and often one is
implimented on top of the other...

So, that would mean the connection is still alive, as far
as it knows... Which means the NT end never closed its
end before it died... Are you talking about the OS itself
crashing, or a process crashing?? Is NT truly so vile and
disgusting as not to forcably close the open FDs of all
dying processes?? ;-) If it's the OS and/or the machine
crashing, then sure I could easily see that situation
happening... It's the same situation you'd encounter if
you yanked out your ethernet cable with a TCP connection
in progress, then rebooted your machine: the other end would
never know there's a problem, until the TCP timeouts
eventually kicked in... (And, if there was no data transfer
in progress, it'd never know at all, unless you specifically
enabled keep-alive messages on the socket... Or, the next
time you tried to perform I/O on the socket...) It's all
part of TCP/IP's reliability: the remote going away without
properly shutting down its end looks to it like the cause
of (hopefully temporary) network flakiness, which it just
tries to ride out as best it can...

You might want to look into setting SO_KEEPALIVE, and
adjusting the timeouts on your system... Or, implimenting
your own app-level keep-alive system (ie: periodically
send dummy messages back and forth, and if any one ever
fails to send or get a reply within a certain amount of
time, then consider the remote dead, and shut it down)...

From: Jacob Harris

Hello,

Thanks for the insight. It has some good ideas. Of course, I'm a bit perplexed, since the select seems to block for longer than the timeout setting. So, is it possible for select to block without timing out, or perhaps timing out on the longer TCP timeout (those can be tweaked in the Kernel)? In any event, it seems like the NT machine is having problems due to another service generating a ton of network connections that get stalled in FIN_WAIT_1 (mentioned in another question on this site?). We'll see what's up with that.

Yours,
Jake

From: Jacob Harris

The Plot Thickens...

Well, it wasn't the OS, but the server application, through what turns out to be a rather stupid bug. It does seem like NT fails to close the connections properly or something. Still, there is a problem with using KEEPALIVE on the client side. According to my understanding, Keepalive has 4 behaviors:

1. If remote server responds, everything ok. Continue.
2. If remote TCP not responding, keepalive sends 10 probes in sequence, waiting 75 seconds or so for each (total of 750 seconds checking). If no response, assume connection dead.
3. If remote TCP back (machine back up) but connection dead, keepalive should receive a RST from remote service and kill the connection.
4. If remote host up and running but unreachable, same as situation 2.

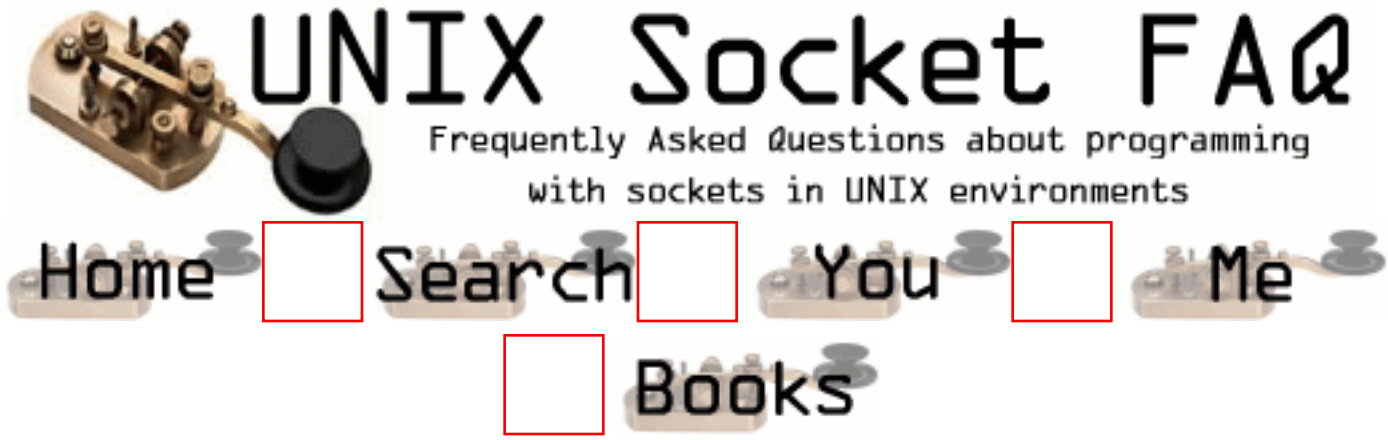
The problem is that this doesn't seem to be working for the client. I purposefully kill the server and restart, but I get neither an immediate response (eg, RST from NT) or the timeout. It almost seems like NT is keeping the connection open in some sort of limbo. Does this sound familiar to anyone, or am I just a bit confused?

Yours,
Jake

From: Jacob Harris

Duh! I forgot that Keepalive probing is not triggered until tcp_keepalive_interval msec has passed on the connection. By default, under Solaris, it's 7200000 or 2 hours. I'll keep that for now I guess, but do people here prefer tweaking that to lower values with ndd (I know, it can be dangerous to get too low).

Add a Comment

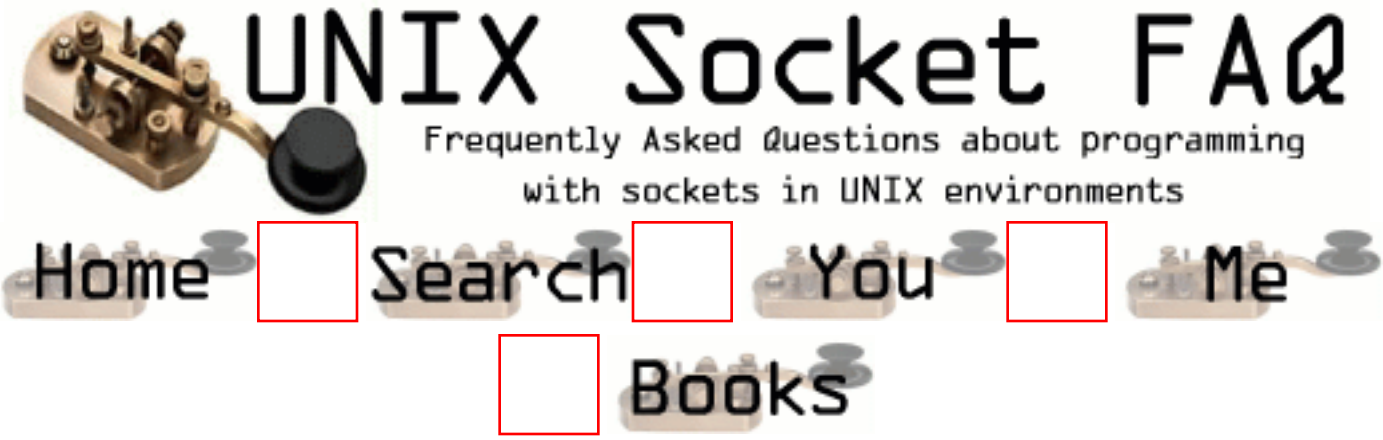


New Questions: Where can I find the source code for server written for broadcasting audio ?

From: shilpa

Where can I find the source code for server written for broadcasting audio ?
Is Perl version of the same available anywhere?

Add a Comment

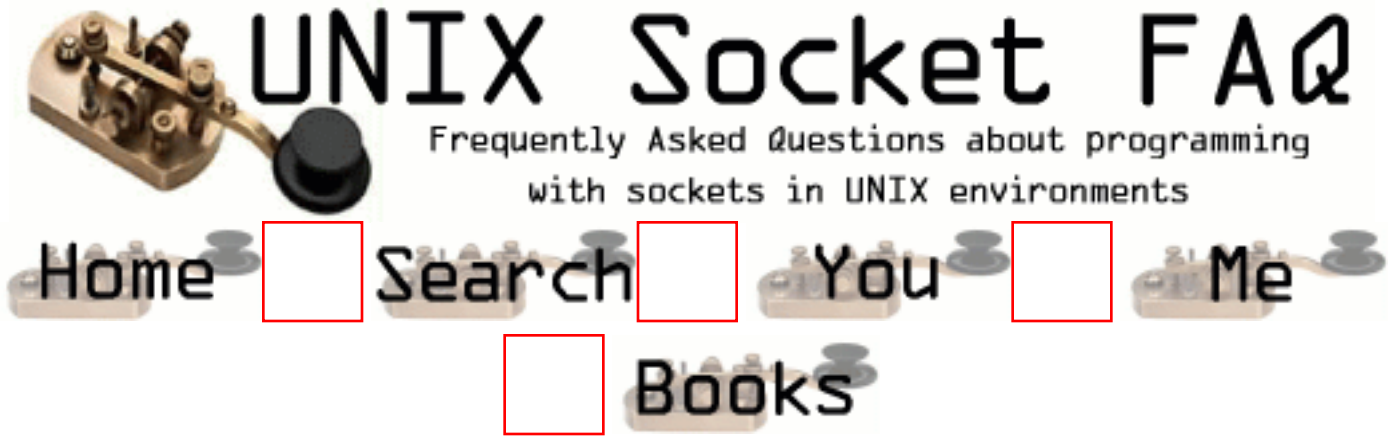


New Questions: download component on Unix

From: Satish Suttatti

Hi,
would like to have a component in C++ which would download a url...preferably running on Unix and other platforms.
Pointers to the same will be appreciated!
Thanks
Satish

Add a Comment



New Questions: UDP server question

From: Andy G

I am trying to "proxy" between two boxes communicating via UDP. They each have a client server pair on them, with both servers using port 3386. The problem I have is that if I put a server on my "proxy" box, on port 3386, it receives packets from both hosts, but does not know which one to send on to after processing.
Is there a way that I can either differentiate packets arriving based on their source IP address, or, can I set up two servers, each of which will only accept packets from specific IP addresses?

I'm working in C on a Unix platform.

Any help would be appreciated.

Andy

From: Loco

Yes, you can do both:

1) You can check the source address of the received packet by using recvfrom() function. (This is just an example, no error checking...)

```
res = recvfrom(sock, buffer, MAX_BYTES, 0, &from, &len);
if (from.sin_addr == to1.sin_addr)
   to = &to2;
else
   to = &to1;
len = sizeof(struct sockaddr_in);
sendto(sock, buffer, res, 0, to, &len);
```

In this example, to1 and to2 are sockaddr_in structs that have the addresses of each client. After receiving a packet the source address is compared against one of the addresses and the

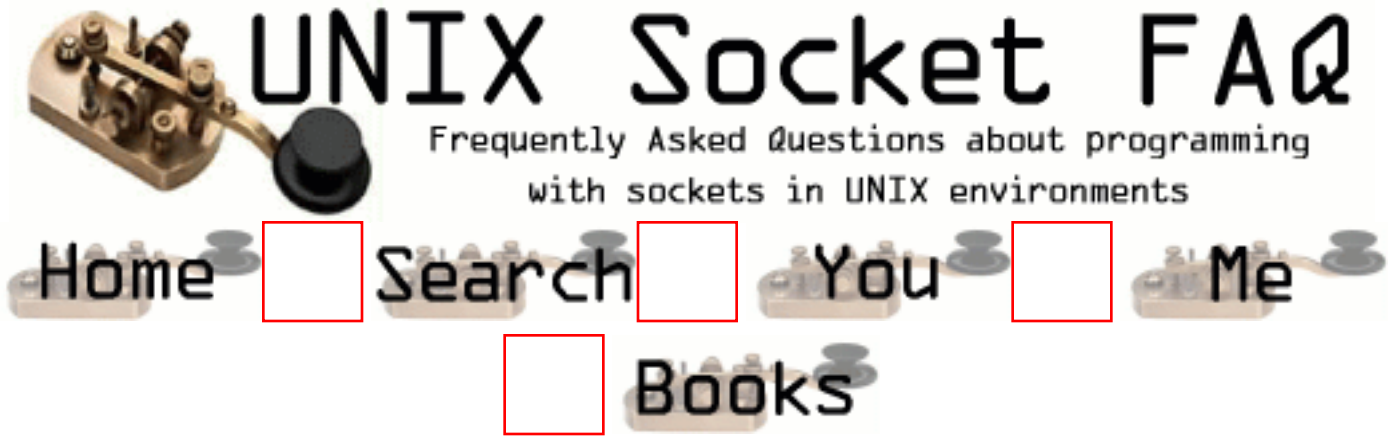
destination of the packet is defined based on the result. Then the packet is sent using the same socket it was received on.

2) You can bind your socket to any of the addresses you have on your computer:
(no error checking)

```
addr.sin_port = 3386;
addr.sin_family = AF_INET;
addr.sin_addr.s_addr = inet_addr("192.168.1.100");
len = sizeof(addr);
// bind first socket on 192.168.1.100
bind(sock1, &addr, len);
addr.sin_port = 3386;
addr.sin_family = AF_INET;
addr.sin_addr.s_addr = inet_addr("1.2.3.4");
// bind second socket on 1.2.3.4
bind(sock2, &addr, len);
```

And voilà, you have two UDP sockets bound to two different IP addresses, but on the same port.
You will receive packets on one socket and send through the other one. Probably you will use select() to know when data is ready on any of them...
:) (HAL)

Add a Comment



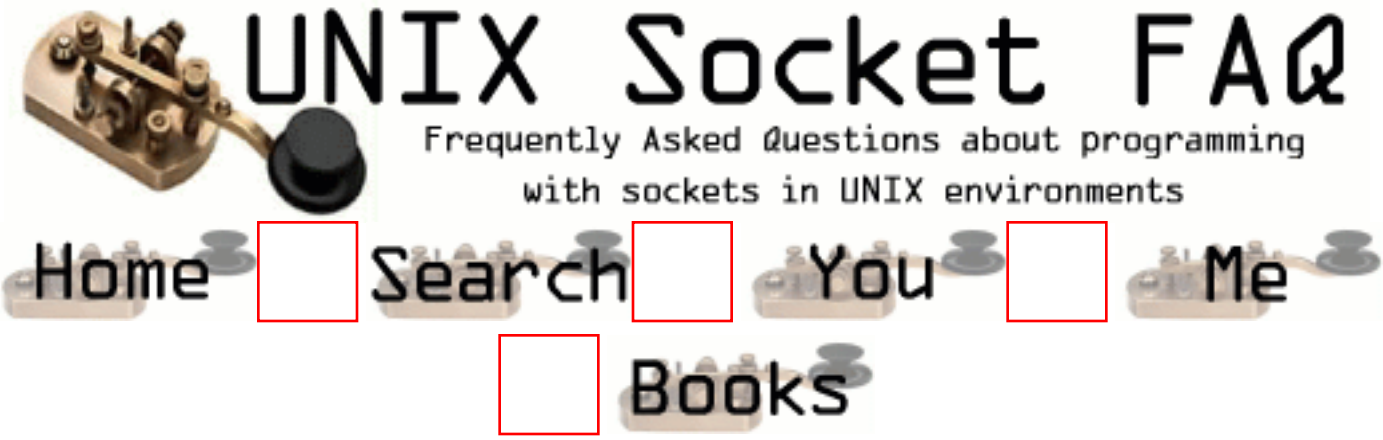
New Questions: Regarding connect system call

From: manjunath.m.n

Hi can anyone tell what this code do.
if(connect(0,NULL,0) && errno != EISCONN)
exit(1);
else
  printf("Everything is ok") ;

I this connect system call is being called without any socket system call. But all I found is it is initially checking whether inetd is configured properly or not. further, it is used for connection oriented socket. Please reply asap.

Add a Comment



New Questions: Could someone explain me something strange

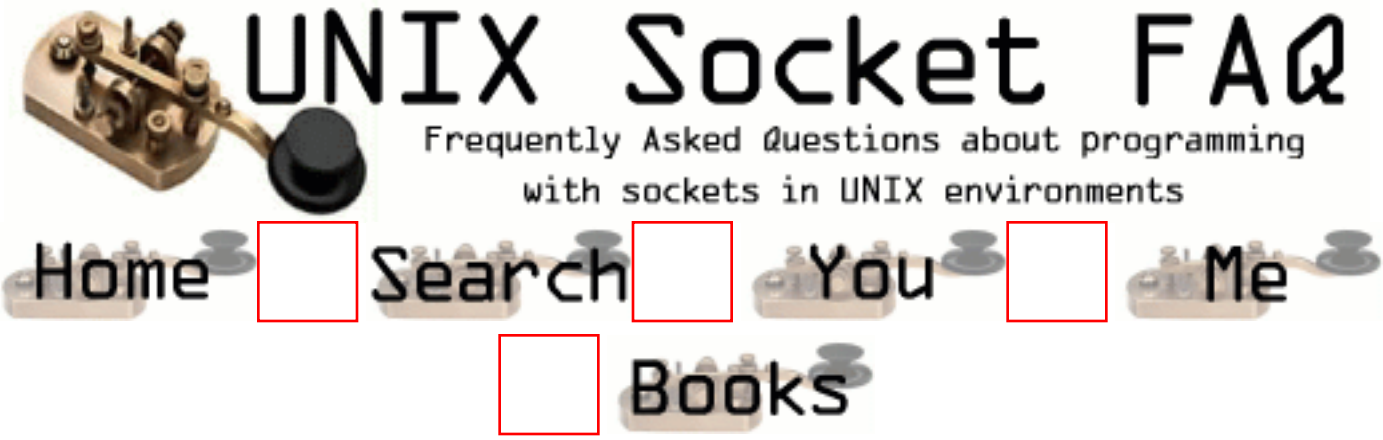
From: Jean-Sebastien

Hi,
I develop a server on linux which runs on a machine connected to
the internet. I run clients to test the application on machines
that are in a LAN behind a firewall which can access the internet
by a gateway. (I'm not sure what I'm saying is correct since I'm
not very familiar with network component, but I have different IP on
LAN and on the Internet). So my program establish a persistent TCP
connection between server and client. The client is a Windows program
and it seems to happen sometime that the server recv a packet
of length 0 from a socket without the client explicitly send
a close. In that case my server program close the socket because
it thinks the client close the socket on his side which cause
a problem when the client did not actually close the socket in fact.
Does somebody have any idea why I would receive a messge of length
0 from a socket which has not been explicitly closed on the client
side? Is it possible that it is caused by the gateway between
the LAN and the Internet?

Thanx a lot

Jean-Sebastien

Add a Comment



New Questions: what is HUP

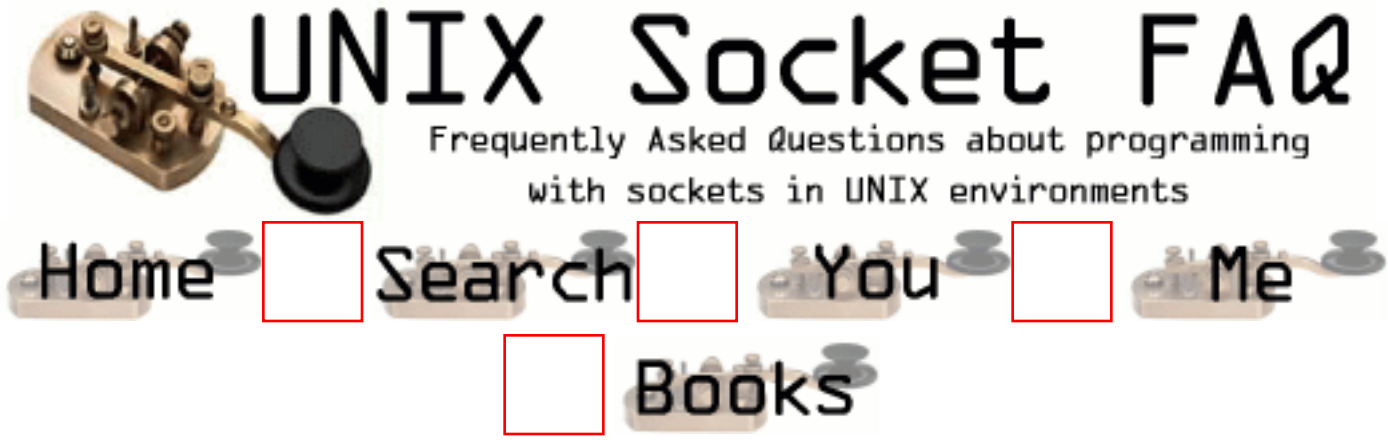
From: Tomaz

I like to know what does HUP mean
Thank's
Tomaz

From: Rob Seace

HUP is short for SIGHUP, and stands for "Hang-UP"... It's a signal typically sent to all members of a process group when the process group leader dies... Eg: when your login shell goes away, all processes it was running get hit with SIGHUP...

Add a Comment



New Questions: Java Multicast Client with two NICs

From: Sandi

I am trying to write a Multicast Client in Java 1.2 using the java.net.MulticastSocket. The problem is that my client host has two NIC cards connected to two different networks. I need to specify an interface for the socket to bind to at creation time because the interface that I need to receive on is not the default, INADDR_ANY.

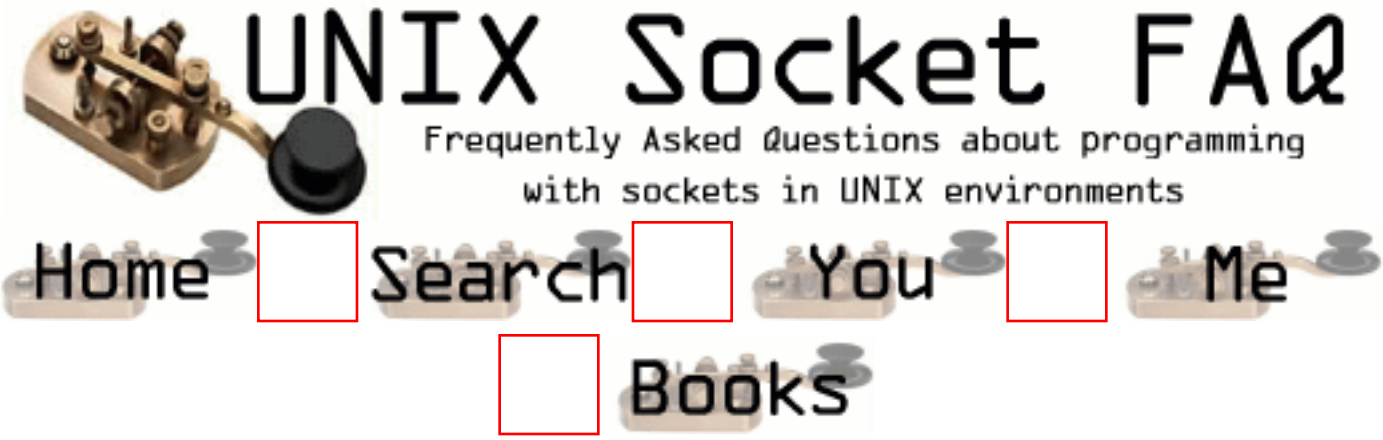
There is a method setInterface() for the MulticastSocket, but it seems to be only for outgoing packets because it does not change the local address. Also, I've looked at the Java 1.3 source for the MulticastSocket and when it binds the socket it specifies INARRD_ANY as the local address to bind to. There does not seem to be anyway to set the local address on a MulticastSocket.

Has anyone else ran into this problem? Is there a solution?

Thanks,

Sandi

Add a Comment



New Questions: Creating non_blocking sockets

From: Mathew

Hi,
I have a server program running on linux (developed in c).I am not able to make my socket non_blocking.I tried with fcntl(socket,F_SETFL,O_NONBLOCK) but it did not work.Can anybody please help me.
Iam not so experienced in socket programming.
thank u
Mathew

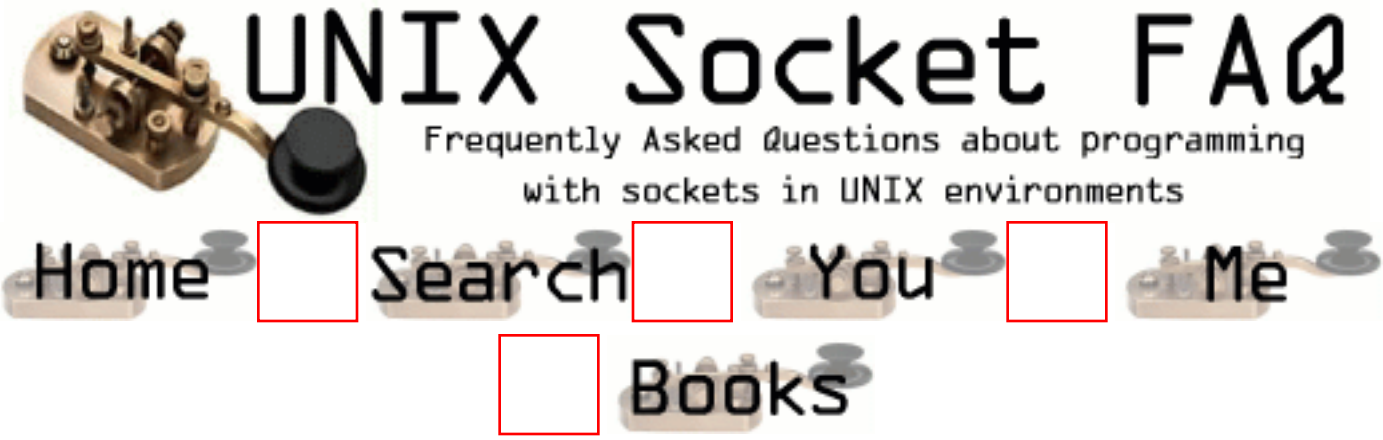
From: Rob Seace

You'll need to be a bit more specific... Exactly how is it
"not working"? Does the fcntl() call fail? If so, what is
errno set to?

Also, the 'correct' way to enable non-blocking mode is to
first get the existing flags with F_GETFL, then OR in the
O_NONBLOCK flag, then F_SETFL the combination... Ie:

```
flags = fcntl (sockfd, F_GETFL);
flags |= O_NONBLOCK;
fcntl (sockfd, F_SETFL, flags);
```

Add a Comment



New Questions: how to write a port scanner on unix using c c++

From: sinni

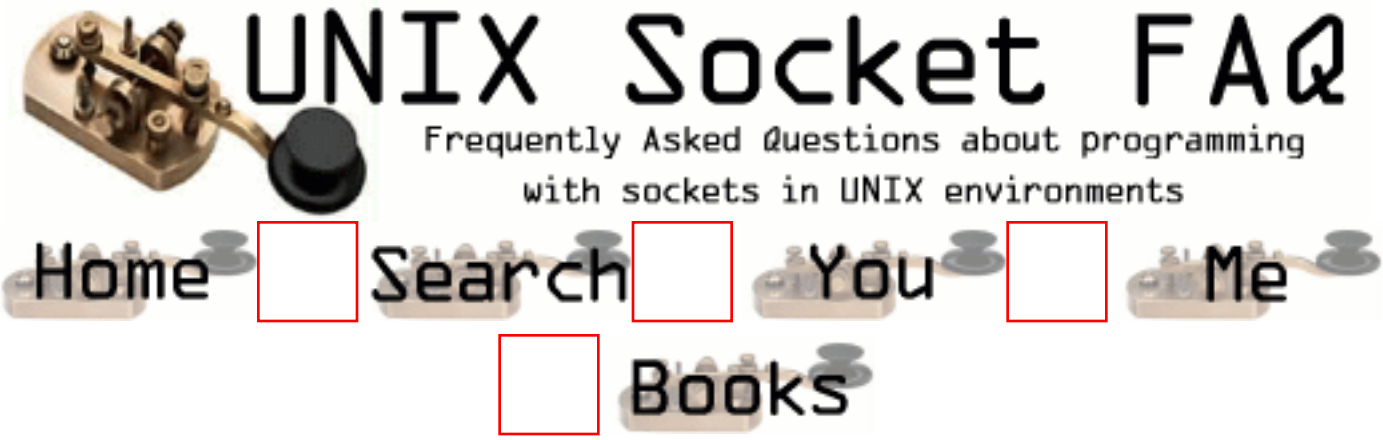
Hi,

can any body tell me how to write a port scanner program on
unix using c,c++
thanks in advance

From: Rob Seace

Why reinvent the wheel? If you want an excellent (and free) port-scanner, use nmap... Or, if you just want to see how one works, it comes with full source code, too...

Add a Comment



New Questions: how can i attach file in a email

From: steve

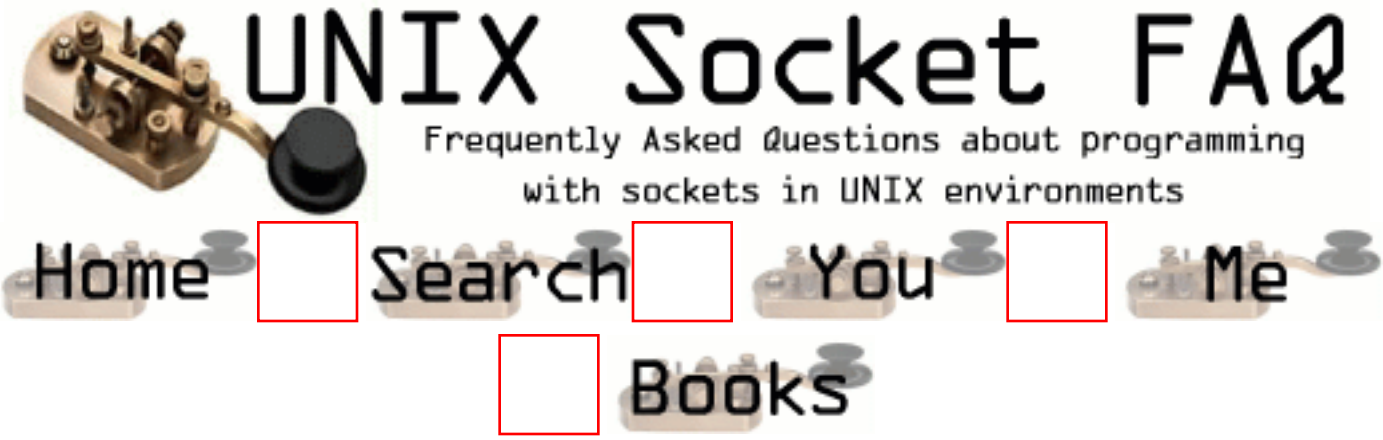
i want to make a shell program in bash.
i want a command that allows me to sent a email to someone with a attached file.
can you please help?
can you also email me the answer please?
thanks.

From: godfrey

worked for me!

exec "uuencode file.txt file.txt |mail host\@localhost";

Add a Comment



New Questions: strings

From: bob

how can i take the first letter of a string so that it can be compared with another character??

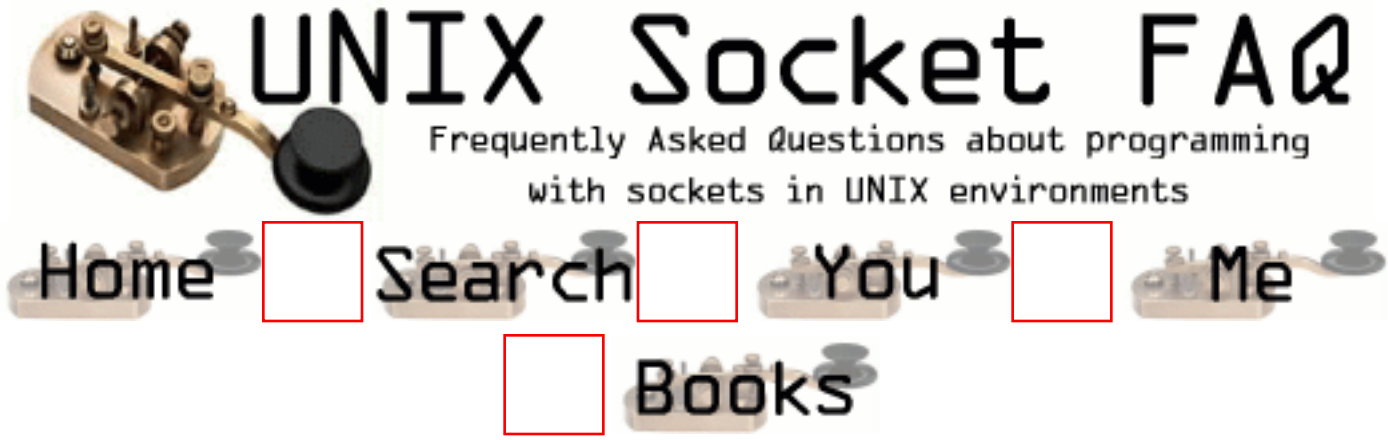
From: Loco

I guess you are trying to do it in C.
The question is out of the scope of this FAQ. Please go to the "I'm learning to write C programs and don't want to buy a book" FAQ.
Or check the following example:

```
char mystring[20];
strcpy(mystring, "Hello world");
if (mystring[0] == 'X') // Comparing 'X' versus first char
   printf("The first character in %s is X"\n", mystring);
else if (mystring[0] == 'X') // 'H' vs first char
   printf("The first character in %s is H"\n", mystring);
```

(HAL)

Add a Comment



New Questions: Listening for remote socket error using poll

From: Oliver

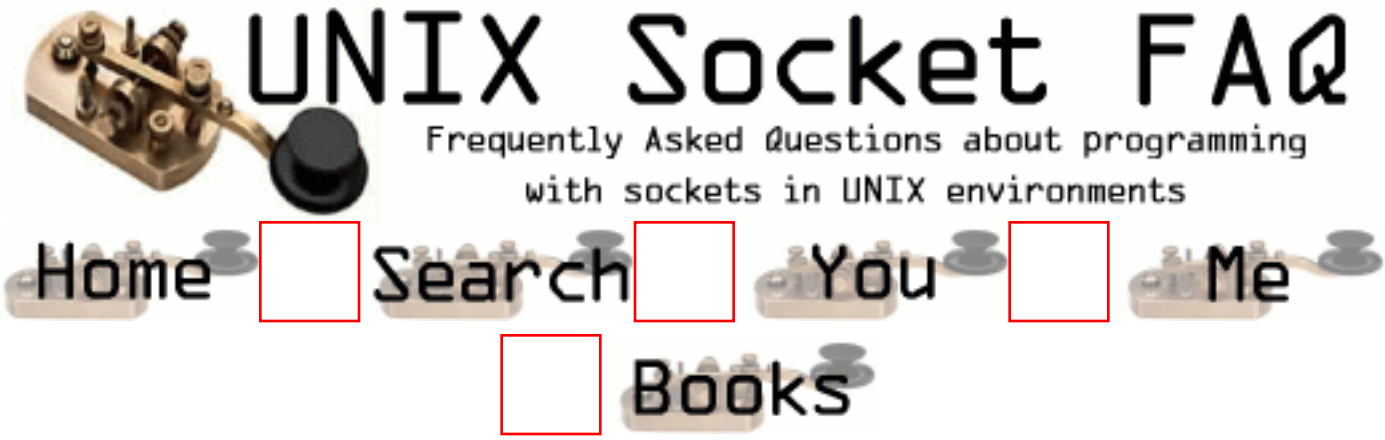
I (as client) am trying to use poll to listen to server errors, mainly a shutdown. So I would like to get notified that the server has shutdown, and i am trying to accomplish this by using:

```
struct pollfd pfd;
pfd.fd = so; //so is valid in here
pfd.revents = 0;
pfd.events = POLLIN|POLLPRI;
status = poll(&pfd, 1, -1);
if ((pfd.revents&POLLHUP) == POLLHUP || ((pfd.revents&POLLERR) == POLLERR) ||
((pfd.revents&POLLNVAL) == POLLNVAL))
return true;
```

The above code is supposed to be in a thread.
Anyway, I start the server, and I start my client, and I make sure the client is connected (through so) to the server. poll blocks (as requested). I bring down the server. poll then returns, but the last if condition is not met!!! it seems when the server is down, poll notifies me about POLLIN event, and does not recognise that there is any POLLERR, POLLHUG or POLLNVAL.

Does anybody have a suggestion?
Thanks

Add a Comment



New Questions: accept(); fills sock_addr structre with the same ip forever after one fail

From: MAT

After Accept fails once (return -1, usually because a client left before i got around to him)

from that point on accept() will always fill my stuctures with the same client ip forever

---

Add a Comment

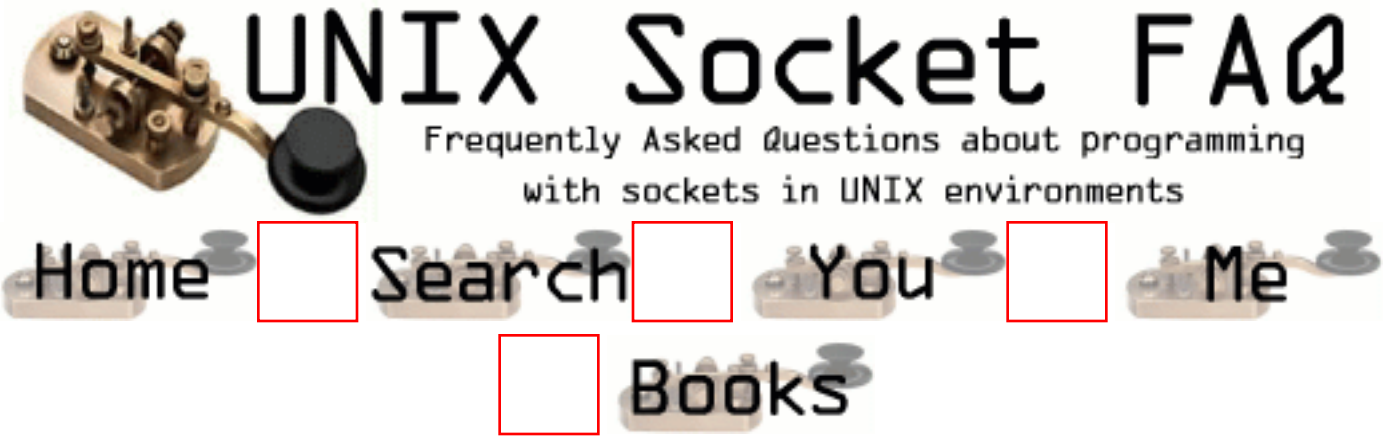
New Questions: Info

From: Justine James

From where can I down load <SupportDefs.h>?
If there is not any site for downloadinf this, please send me the code for that include file.

Thanking you

Add a Comment



New Questions: how to find the memory used by the Stack and the Heap for a function??

From: Jyoti

Hi!! ,
could me suggest me any method to find the amount of memory
used by the stack and heap of a process while execution at any stage?

Thanks in advance,
Jyoti

From: Joy Patra

You can use mallinfo() for the heap area.

Add a Comment

New Questions: where is the memory fora local array allocated

From: jyoti

Hi!!,
I have a lil doubt. As I always thought, memory for an array is allocated
on the heap during the compilation. But what about an array that is declared and initialised inside a function?? The memory allocation is in the Stack or Heap?

Thanks in advance,
Jyoti

From: jdev

Only in stack area .

Add a Comment

New Questions: Async. IO, How to store Clients IP and Port

From: Chandra

Hi,
Scenario :
1. create a TCP/Stream socket using socket() call - Call bind() and listen() to make this a listening socket.
2. Use select() to accept connections - ie., the server would be infinitely blocking till a connection comes, and select will be used to notify a connection request to the server.
3. Without calling an accept(), again use select() on the listening socket to recv data.
4. After recving data, trying to echo back the data() using select() again on the listening socket only.

As from the above scenario, none of the calls listen(), select(), recv() store the information about the clients port and address which are required for send() operation to succeed on stream sockets.
Now the question is, when and how should we store the info. regarding the client's IP and port so that send() succeeds. Is there any prescribed/standard way for this.

Thanks
Chandra.

From: Loco

Hi,

You say you are reading from the bound socket that is currently listening for new connections? Well, that is not the way it is supposed to work:
The socket you bind and listens on a port should "yield" a new socket for every connected client. That is the purpose of accept(): it gives you a socket descriptor of a new connection. You should listen() for connections, when select() returns, accept() a new socket descriptor, add

it to the fd_set, and when there is data ready from this descriptor just read it!

Please let me know if i'm getting all wrong.

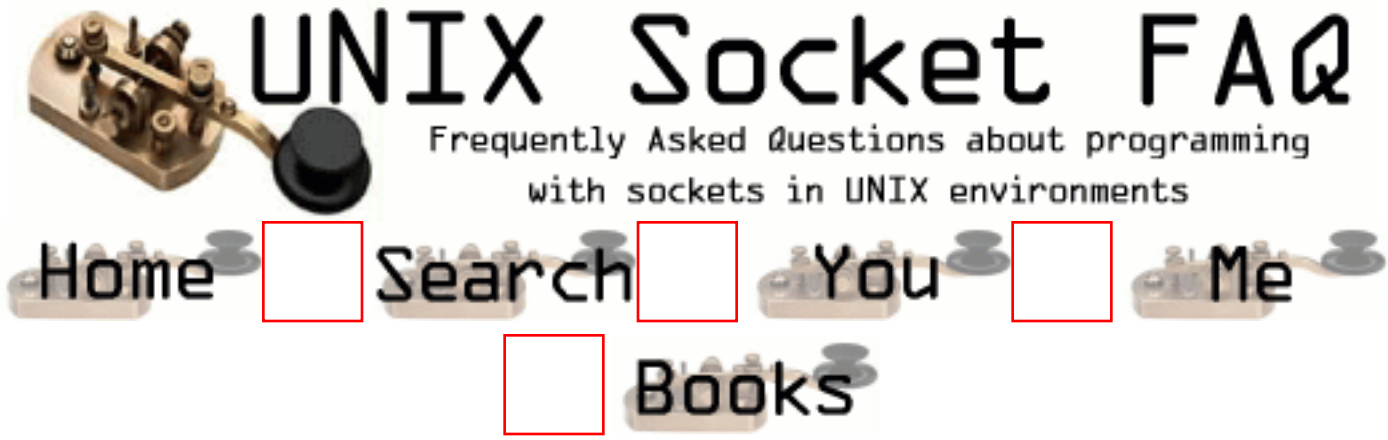
From: jdev

Loco u r correct.We have to use accept() for completing the
TCP hand shake and to return a CONNECTED socket descriptor.

From: jdev

chandra ,is there any specific reason for why U r avoiding
accept() call??.

Add a Comment



New Questions: writing to socket descriptor newSD

From: Flash gordon

Hi

I make a request from netscape to my tcpServer.c programm on port 1500.
The program is then calculating a new Flash-Movie (.swf) and sends it
over the socket descriptor back to the client. the netscape browser doesn't recognize
the content type, plain text is writen.

output on netscape:

Content-Type: application/x-shockwave-flash FWS À & B ÷¹M{ò M±d i8?þ~~ °"
x?A1g@Ð%?¹ Ý?mGý?ñXY?Ö? x<jáÛs)ù wÇ?ûÜw-¿éà 5ê¬Ç)õÇ
µ*3n¥PúÍw}Íì-¾?åK¼a3?¦Ù8½«?¥ ÌñJ®uÐf0)KÓK?yU&ûÍç-ÒoC?À? {ægÞÆ¶Íï 5ìòÙwµ0?ãû
àÖ`?2>ÝÕPøQmö~o
Û½20? ª¹i 6±^?? a 5øÞØTöêWòÙn!bÐáL¸·;÷
ñÆG `ó´ÁÜ Á*¾·ÿ 5ÃÝ¦µã] Úø"â«yI\Ø? ? scÎ<#x?Ø=Ø®ø g¬öY¶Îÿî Ú¶±~?? ^5 » ×f 7 s\mù
ö?ð?wñ?? " ò qÝ {£V-~! qE¦ ¥¶Y ? -°SvÔýÙrÚº ³ ¶?è¢1Ýwi"}ÚF?# ?È! ?Xà5Éi -Ýqkh¯v9®ep
Fr?Ês?*rxo0s1ÚS]½ åß4ö?? Çû» Öçh°UkGÿv-®è ?^ ø£?0?
sAÍ z?
sY?ú e~ »iC·d[VüÀ? Æ²vU¶ÄdìÕíx65Îx eÝYm RÝ?m`ò® £ Æ?h´äf/Fn? " 5 sh-Õ ÌS¨ D Å ? D
7 ? D 7 ?
D 7
ô

program code from tcpServer.c

.

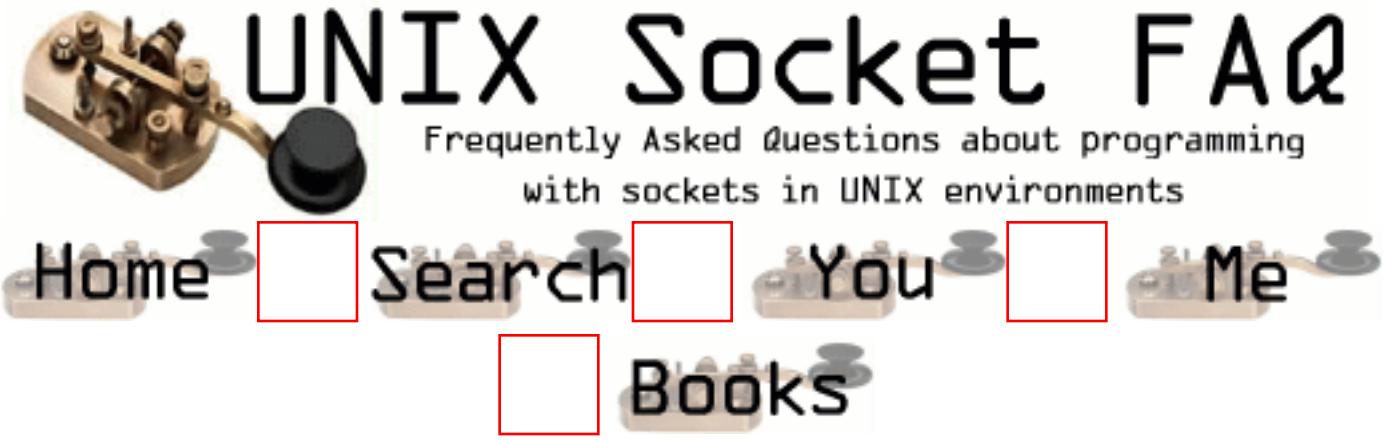
.
.
```
  strcpy(send,"Content-Type: application/x-shockwave-flash");
  write(newSD, send, strlen(send));
  movie.WriteMovie(newSD);
  close(newSD);
  }
return 0;
}
```

can anybody help?

From: Rob Seace

Try throwing a newline ("\n") at the end of that
content-type specification... And, in fact, I think
you need a completely blank line, to indicate end of HTTP
headers, before you begin sending your data... So, make
that 2 newlines...

Add a Comment



New Questions: What is UMODE

From: Mike

Hello
I would like to know what does UMODE means.
Thanks in advance
Mike

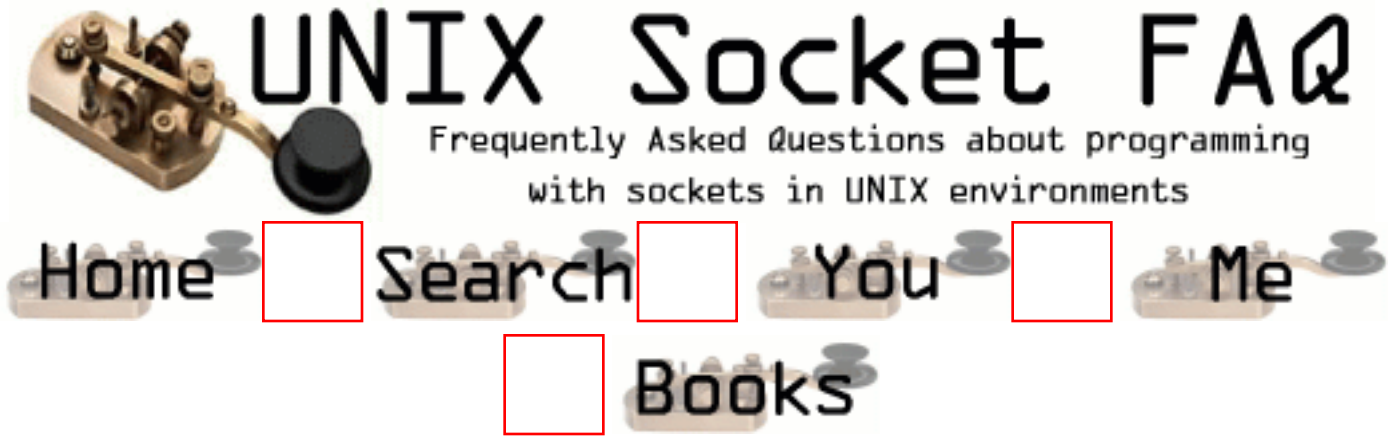
From: susi

more details please???(where u found this etc. which OS? etc..)

From: Mike

Hi Again,
Thanks for your response..sorry for not beign clear about UMODE,I came across a article in Linux saying
User and group ID to create files as Umode in Linux OS..
Just wanted to clear myself on this.
Thanks & Regards
Mike

Add a Comment



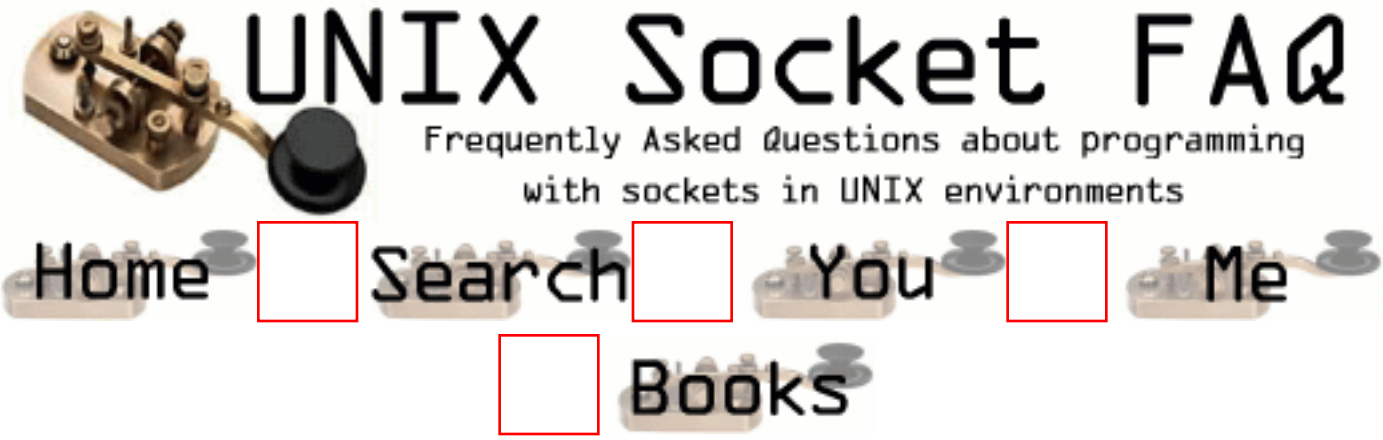
New Questions: Socket error

From: Sougat

Hi,
I am receiving an error when I try to start Websphere Commerce Suite on Aix4.3.3.
CMN0310E: The socket name is already in use.
CMN0519E: Server controller TCP/IP service 'newcom' is already in use, an instance of server controller may already be running.

The problem gets resolved when we reboot the machine. Could someone tell as to how to resolve this problem without reboot.
Thanks,
Sougat

Add a Comment



New Questions: ENOBUFS Returned from Connect()

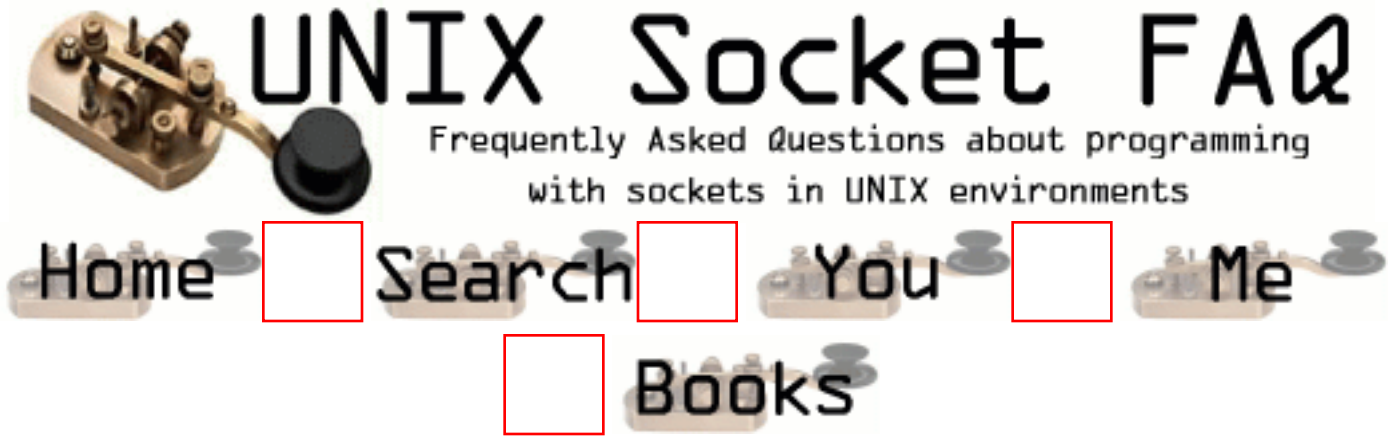
From: Harvey Davis

I was wondering if anyone could explain the ENOBUFS error returned from a connect() call whilst programming sockets under linux.

From: Rob Seace

I'd say it either means you are out of system memory, or more likely, you are simply out of socket buffer space... Have you configured your socket buffers particularly small? Or, do you do anything else abnormal to this socket before connect()'ing it?

Add a Comment



New Questions: Broadcast UDP on linux gives ENOPERM

From: Steve Carter

Just what it says on the title, really. I get ENOPERM when trying to sendto() UDP packets to my broadcast address.

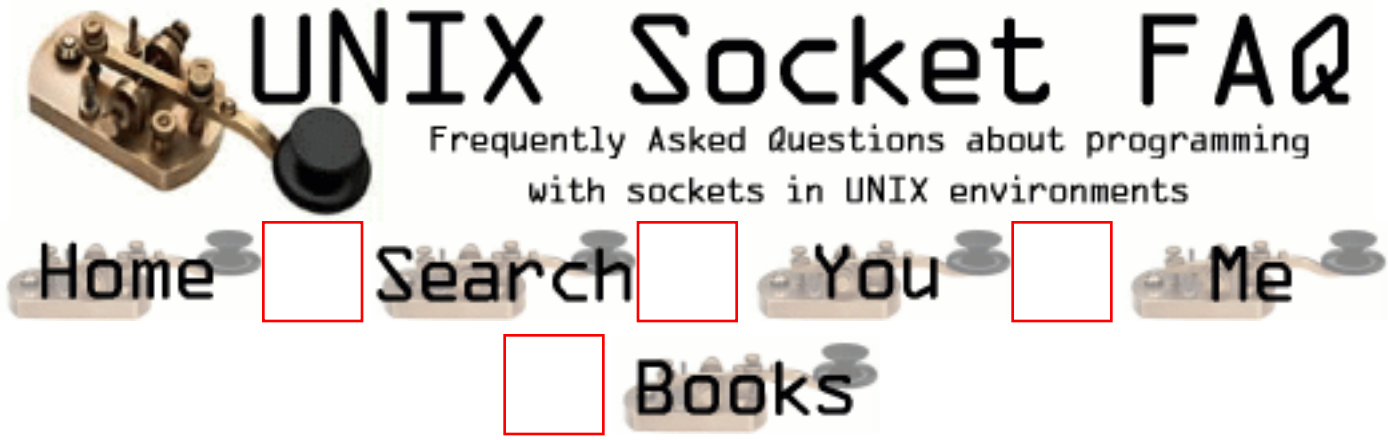
On bit I'm a little doubtful of is that I go

```
const int true = 1;
...
rc = setsockopt(sd, 0, SO_BROADCAST, &true, 4);
```

From: Rob Seace

Your socket level arg to setsockopt() is incorrect, there...
You want "SOL_SOCKET" (which, on my machine seems to be
defined as 1, NOT 0; but, just use the symbolic name, that's
what it's there for)... And, similarly, don't just use "4"
as the size; use "sizeof (int)"... Maybe one day, you'll
compile that app on a machine where int's aren't 32-bits...

Add a Comment



New Questions: TCP programming

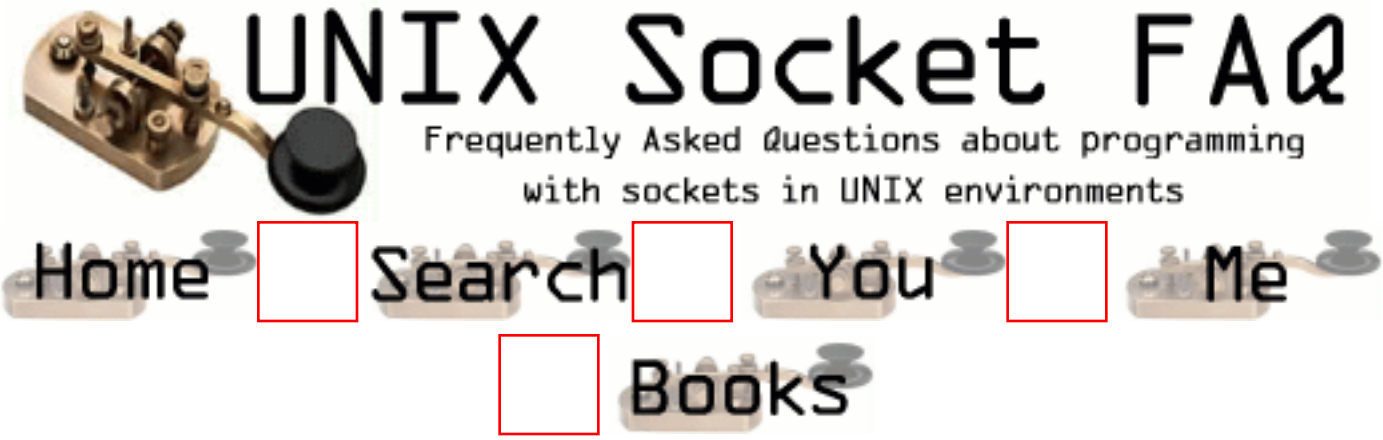
From: SM

I wrote a simple client/server program using TCP protocol under NetBSD OS. The client sends data at rate 800Kbps, but the server's cell loss rario (CLR) is too high, more than 18%. The higher sending rate, the more CLR.

I can't understand the result. If I use UDP, there is no CLR even at rate 10Mbps.

Any help will be highly appreciated.

Add a Comment



New Questions: Program exits with code 141 during recv()

From: Willem Roos

Hi all,

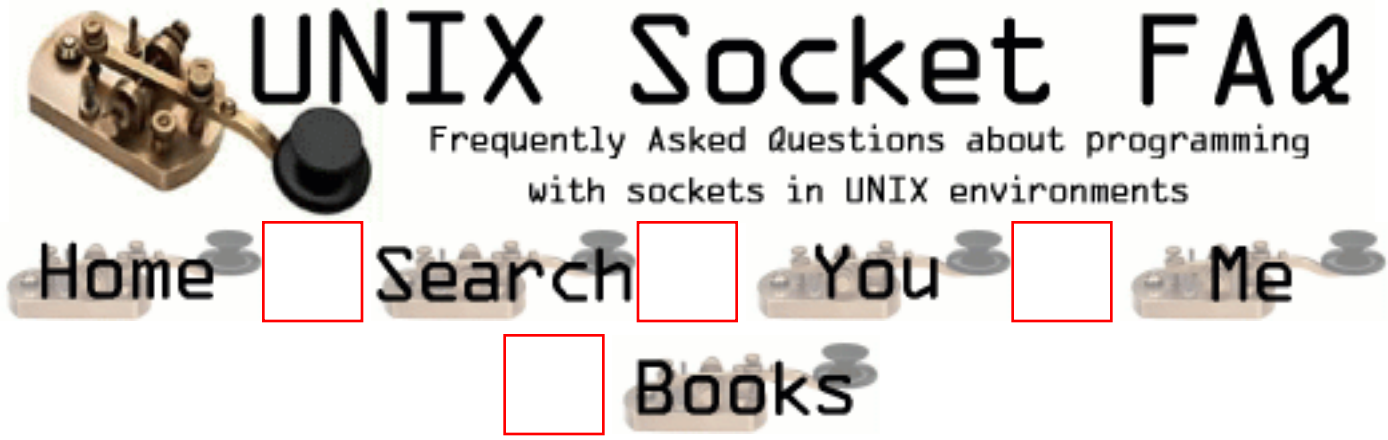
My threaded socket server exits during the recv() system call with exit code 141 - ie. the recv() doesn't return, the whole program exits. No SEGV, no nothing. Same behaviour on AIX and Unixware. Has anyone seen this before?

From: Rob Seace

Just a guess: you're getting hit with SIGPIPE, and you have no handler (and, aren't ignoring it)... SIGPIPE is very common, and MUST be expected in all sockets code... I always simply ignore the signal, myself (then, you can just easily deal with the read()/recv()/write()/send() failures, with EPIPE)...

What leads me to the conclusion is that exit codes above 128 (as seen from a shell, at least; wait() semantics are somewhat different) are typically indications that you exited due to being hit with a signal... And, specifically, the signal number is generally then encoded in the lower 7 bits... Making the above 141 exit code correspond to signal 13; which, on all of the systems I have access to at the moment, is SIGPIPE...

Add a Comment



New Questions: recv()?

From: Ramon Alvarez

Hey, I'm trying to get into socket programming, this will be easily answered I'm sure but will save me from having more headaches...I'm connecting to a deamon on whatever port and such and when I connect it sends me 4 or so values. I've been told to refer to them as ints, but I keep calling them, in this case, numbers ;p. How could I get those numbers printf()'d and how would I go about multipling them together and sending them back?

Add a Comment

New Questions: please.... run it always wrong (on linux 7.0)

From: kolp

proto.h
#define PORT 2345

#define CHALLENGE 0
#define CORRECT 1
#define WRONG 2

struct mrecv {
    int flag;
    int b[4];
    char nextpass[10];
};

typedef struct mrecv t_recv;

struct msend {
    int answer;
    char prevpass[10];
};
typedef struct msend t_send;

    int answer;
    char prevpass[10];
};
typedef struct msend t_send;

newbie7.c

```
#include "proto.h"
#include <sys/socket.h>
#include <sys/types.h>
#include <netinet/in.h>
#include <string.h>
#include <stdio.h>
#include <arpa/inet.h>

int
main ( int argc ,char **argv){

int s;
int reclen;
int answ;
struct sockaddr_in sa;
t_recv m_recv;
t_send m_send;

if ( argc != 3 )
printf ( " %s USAGE: %s < host > <port> \n ",argv[0],argv[0]);

if ( (s=socket(AF_INET,SOCK_STREAM,0)) == -1)
printf ( " THE SOCKET IS ERROR!!! \n ");

memset ( &sa,0,sizeof(struct sockaddr));
sa.sin_family = AF_INET;
printf ( " OK ");
sa.sin_port = htons( *argv[2]);
printf ( " OK ");
inet_aton(argv[1],&(sa.sin_addr.s_addr));
printf ( " OK ");
if ( connect(s,(struct sockaddr_in *)&sa,sizeof(struct sockaddr_in))== -1)
printf ( " THE CONNECT IS ERROR!!! \n ");

if ( (reclen = recv (s,(struct mrecv *)&m_recv,sizeof(struct mrecv),0)) == -1)
printf ( " THE RECV IS ERROR!!! \n ");

if (strcmp((char *)m_recv.flag,(char*)0)==0)
answ = (m_recv.b[0])*(m_recv.b[1])*(m_recv.b[2])*(m_recv.b[3]);
else printf ( " THE CHELLENGE IS WRONG \n ");

m_send.answer = answ;
m_send.prevpass[0]='3';
m_send.prevpass[1]='S';
```

```
m_send.prevpass[2]='i';
m_send.prevpass[3]='g';
m_send.prevpass[4]='e';
m_send.prevpass[5]='1';
m_send.prevpass[6]='V';
m_send.prevpass[7]='e';
m_send.prevpass[8]='\0';

send ( s,(struct msend *)&m_send ,sizeof (struct msend),0);

if ( (reclen = recv(s, (struct mrecv *)&m_recv ,sizeof (struct mrecv),0))==-1)
printf ( " THE RECV IS ERROR!!! \n ");

if (strcmp((char *)m_recv.flag,(char *)1)==0)
printf ( " THE ANSWER IS RIGHT ! THE NEXTPASS IS %10s \n " ,m_recv.nextpass);
else if (strcmp((char *)m_recv.flag,(char * )2)==0)
printf ( " THE ANSWER IS WRONG!!! TRY AGAIN \n ");
else
exit (1);
}
```

From: kolp

the host is 209.9.44.215

$./newbie7 209.9.44.215 2345

From: Rob Seace

A lot of that code just doesn't make a whole lot of sense
to me... Eg:

> sa.sin_port = htons(*argv[2]);

That's taking the ASCII value of the first character of your
second command-line argument, and using that as your port
number... Um, is that REALLY what you want to be doing??

> if (strcmp((char *)m_recv.flag,(char*)0)==0)

This one just totally baffles me... You're casting an
integer to a pointer, then trying to do a strcmp() on it,
against a NULL pointer... WTF?? That makes no sense...
Then, later, you do a similar thing, but comparing against
pointer values of 1 and 2... All of these calls should
likely simply seg-fault on you...

There are other issues as well, but the above is the most
disturbing and incomprehensible... I don't know what you
are trying to accomplish exactly, but you can't go about
it that way, I'm sure...

From: kolp

you means that

i shoult write as this "(strcmp (m_recv.flag,0)==0)" ???

From: Rob Seace

Um, no, that's not valid either... What exactly are you
trying to do?? I can't tell you the correct way to do it,
until I know that... And, I just can't even hazard a guess
at what you're trying to accomplish with these bizarre
strcmp()'s on integers and NULL pointers and such...

From: kolp

thanx your answer, my english not well ! sorry.

it is www.pulltheplug.com's wargame level7 question. i will paste the DOC.

k... from now on, you'll need programming skills... i suggest you learn c,
but i think it's also possible to do newbie8 in perl

Your goal is to write a client for the daemon running on port 2345.
The server sends you a CHALLENGE (0), with 4 ints, and you'll have to
multiply the ints (int1*int2*int3*int4) and send it back as the answer,
together with the newbie7 pass in prevpass.

When you get the answer right, the server sends a CORRECT (1) back, with
the next level's pass in nextpass

If the answer is wrong, the server sends a WRONG(2) back, and terminates

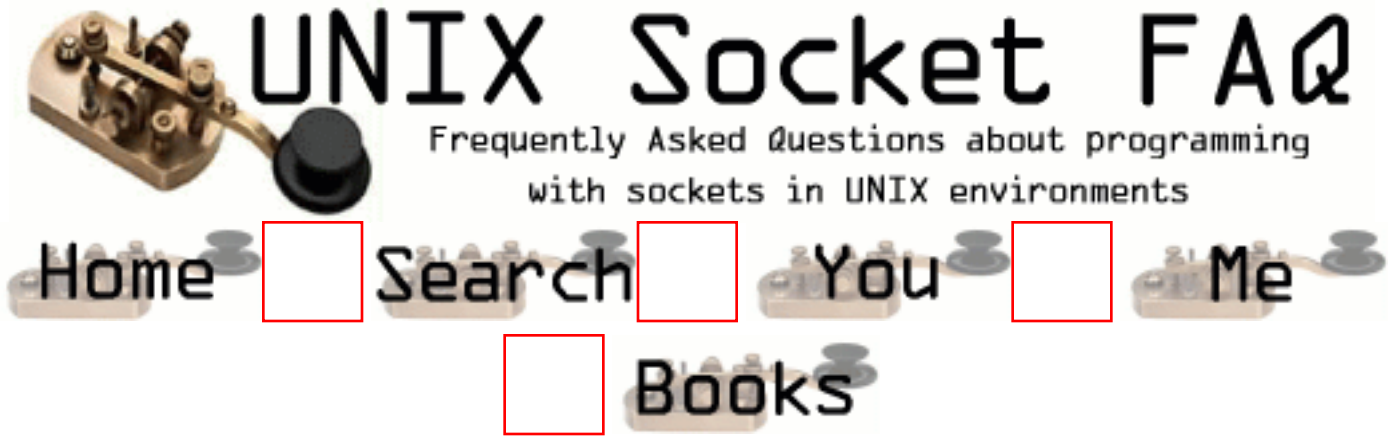
A good place to start is the libc info pages

From: Rob Seace

So, is what you're trying to do above simply compare
whether or not that int flag is equal to 0, 1, or 2?? If

so, you surely don't need or want those strcmp() calls, at all... What's wrong with simply "if (m_recv.flag == 0)", and so on?? I assume you're actually getting sent the raw ints like that, rather than as string representations, or something? (In which case, you realize it's only going to work if both the client and server are running on the same type of architecture? Specifically, they both need the same byte ordering rules ("endianness") of int data... Unless you're converting those ints to/from network order...)

Add a Comment



New Questions: What happens with incoming buffer over-run

From: Stephen Hucker

I`m writing a small TCP server on linux and i wanted to know what happen to the data when it is greater than the max lenght to copy into the buffer. This is when using the command recvfrom(int socket,char* buffer, int lenght etc.. and lenght is the buffer size. I want to know this, to make sure data can`t start writing into other parts of memory, if the clients sends more data than can be read into the buffer. Cheers Stephen.

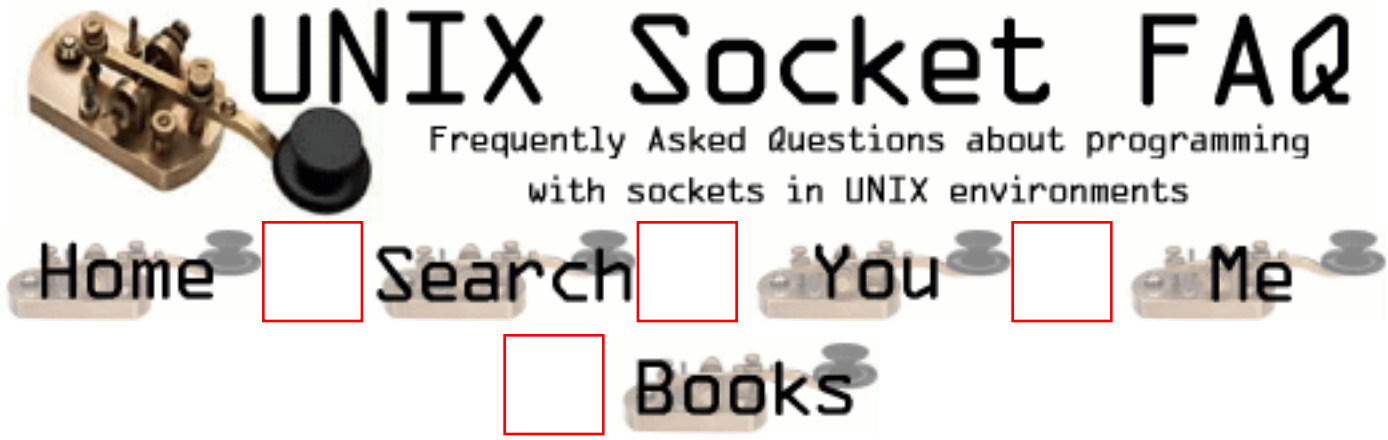
From: Rob Seace

I assume you're talking about UDP, since you mention recvfrom()? If so, single UDP packets which exceed the length given are just truncated, and the excess discarded, to be lost forever... With TCP, since it's stream-oriented, you'll just read as much as you can at a time, then read the remainder next time...

From: Stephen.

Cheers Rob

Add a Comment



New Questions: Windows sockets

From: Srinivasa Reddy

I am using windows sockets and i want the program control to come out of this accept() function when no requests are coming. Is this possible?

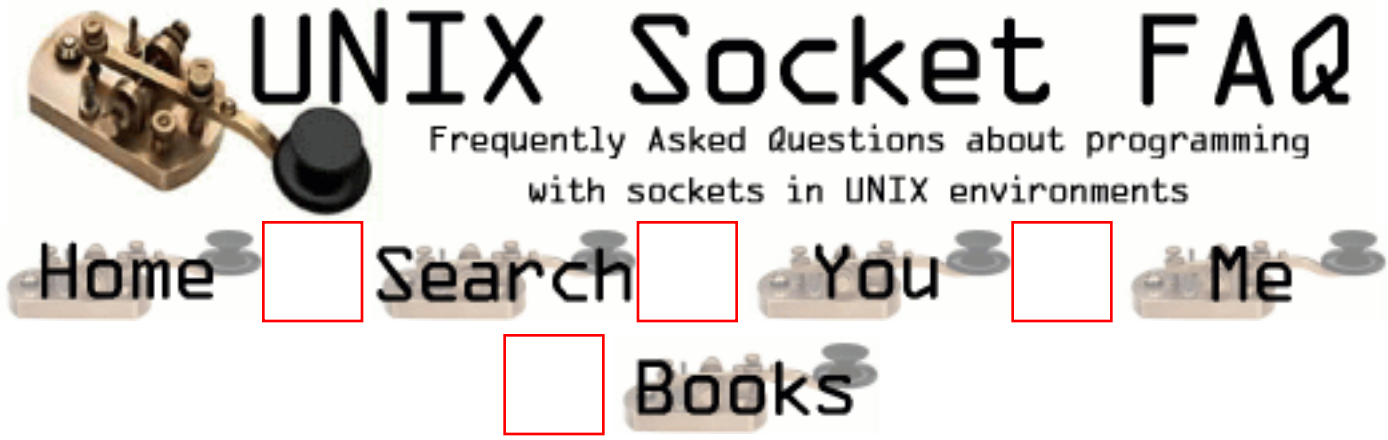
thanks
Bsreddy

From: Scott Weber

If you already have a window message loop set up, this is extremely simple. Use the WSAAsyncSelect() function and register for the FD_ACCEPT event. Then when you receive the message from your call to GetMessage() or PeekMessage() or whatever, you are guaranteed that a call to accept() will succeed immediately.

But isn't this the UNIX Socket FAQ anyway?

Add a Comment

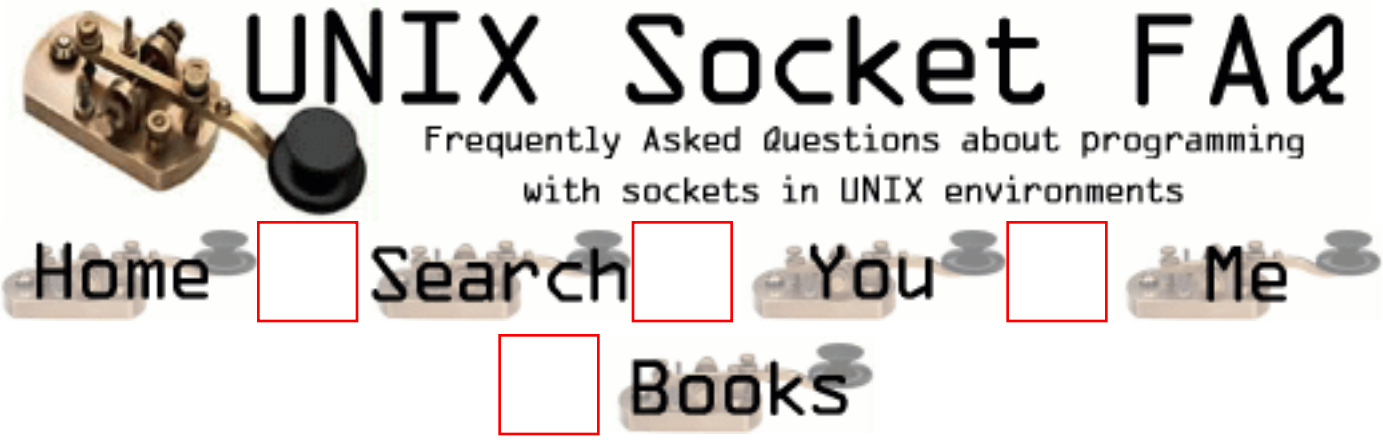


New Questions: How to bind Multi-IP to a MAC

From: rapide.zhang

I want to bind more than one IP to a server in my UNIX program so that it can provide different service, but i don't know how to implement it, can you help me?

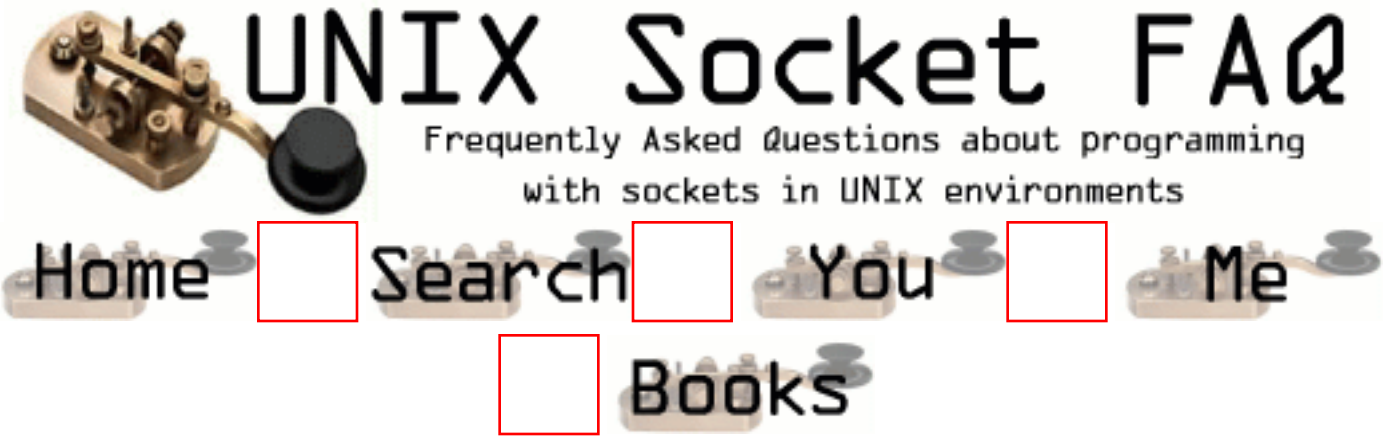
Add a Comment



New Questions: How can I run a program and set some environment variable when "startx" on RedHat ?

From: Newer

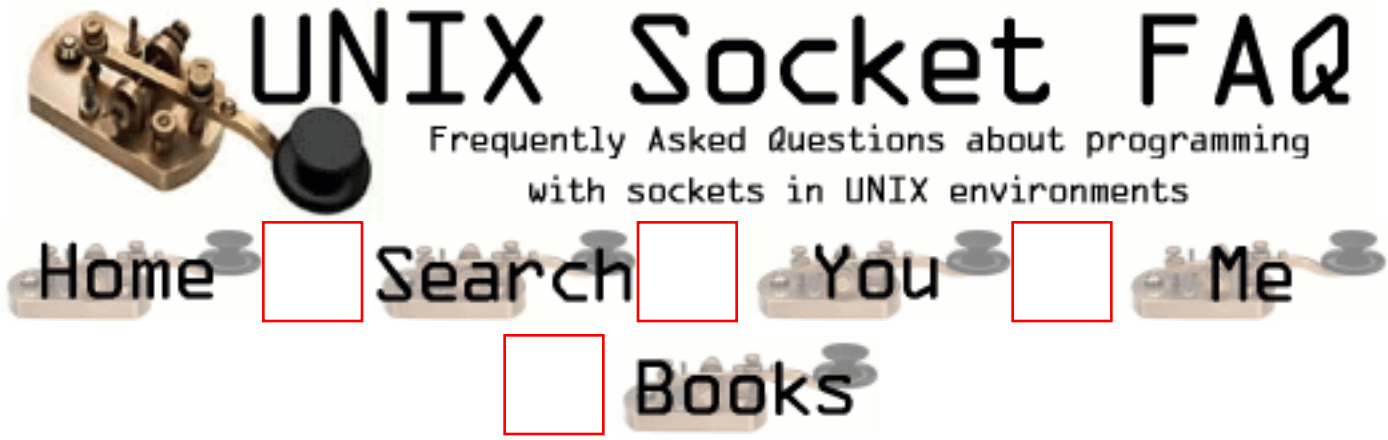
Add a Comment



New Questions: How can I run a program and set some environment variable when "startx" on RedHat ?

From: Newer

Add a Comment



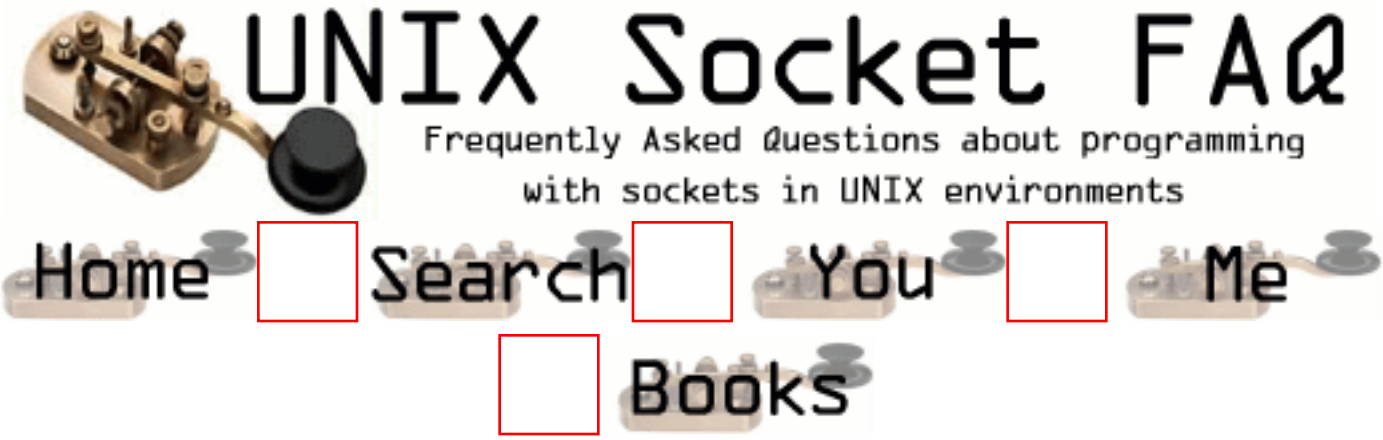
New Questions: Limit on maximum number of sockets

From: Chris Black

What is the maximum number of sockets you can open for
entire system. i.e. AIX 4.3

Thanks

Add a Comment



New Questions: Select problem - read()

From: Martin Lexa

Hello all!

Situation:
1) multithreaded Client/Server program (Linux)
2) one thread waiting for connection and then creating work thread
3) work thread - read select() on UNIX socket;
4) reading data; when found specific data connect to another Client/Server machine (HP/UX).
5) work thread - read select() on UNIX socket and new TCP socket (HP/UX), write select() on new TCP socket (HP/UX).

At this point there is sometimes error. Select() wakes up and try to block read from TCP socket even if there are no data available. This happen when there is no delay between connect() and select() calls. When I add some delay between connect() and select() everything works. After looking at select() return value it seems ok - positive number.
However this is not solution for me. I solved this with non-blocking read, but I'd like to know where is the problem.
I'm sorry for my English and hope you understand me.

Thank you for your responses, Martin

p.s.: When tested on my work computer (Linux 2.4.1) everything works. No problems here.

From: Loco

I think the problem is you are setting your socket non-blocking before calling "connect()". In this case, the connect call will return inmediately with -1 and errno set to EINPROGRESS. When the connection is established the select() call wil flag the socket descriptor in the write set.
So, to avoid this problem try setting the socket to non-blocking AFTER calling select, or check the connect() return value and errno variable and use a flag to know when the connection is completely established.
The man page for connect reads:
"[...]After select indicates writability, use getsock-opt(2) to read the SO_ERROR option at level SOL_SOCKET to determine whether connect completed successfully (SO_ERROR is zero) or unsuccessfully (SO_ERROR is one of the usual error codes listed here, explaining the reason for the failure).[...]"

:) (HAL)

From: Martin Lexa

Hello!

Thank you for your response Loco, but this wasn't helped me ;-)
I've always have connect() in blocking mode and every call is checked for return value. Connect and accept is returning positive value, so I don't expect there is much room for error.
BTW, connect is set to blocking mode by default or am I wrong?

Should it be problem with kernel (Linux 2.2.16)?

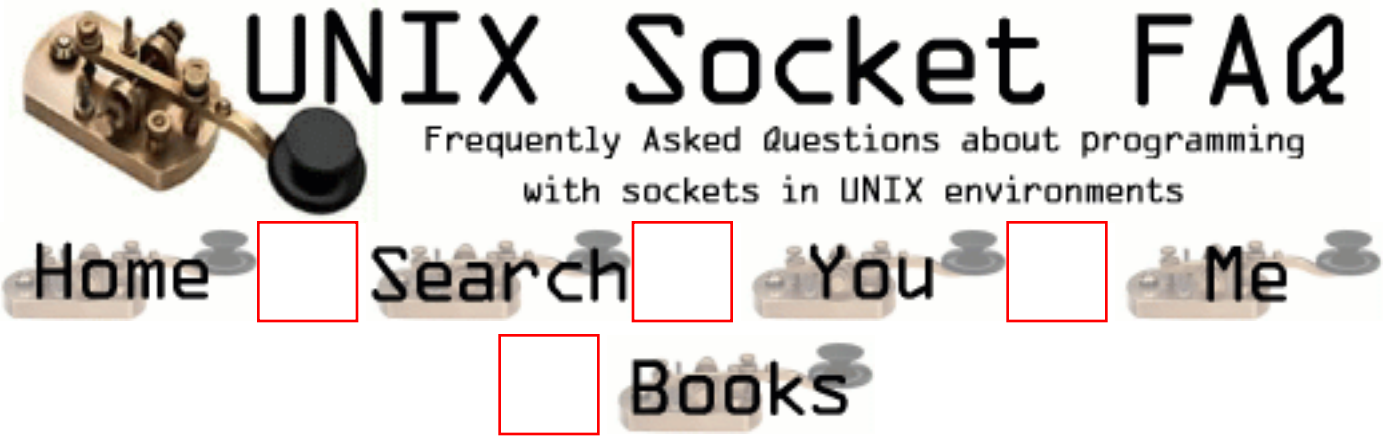
One more thing: It's never happen in first round, only in laters when socket was allready closed and reopened. With this I mean connect() gives the new connection same descriptor number as previously used and correctly closed socket connection.
BTW, before closing connection I'm sending shutdown(desc, 2) call. And yes, I'm verifying every return code, so shutdown and close call are returning positive numbers.

Thank you for your responses, Martin

From: Nan

Wepre on our third attempt at using DSL, but problems still dog us,(excuse the pun, I'm a dog-lover as well). The scary error message,"fatal, could not create socket". Any help to offer?
Thanks, Nan

Add a Comment

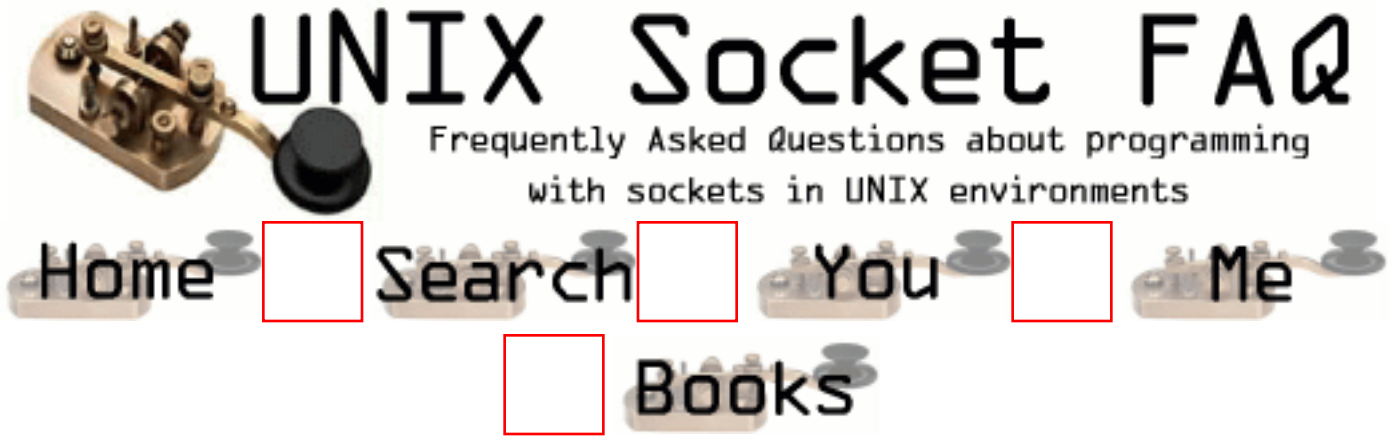


New Questions: socket programming

From: vipul singhal

how i send a integer value from client to server

Add a Comment



New Questions: can u send a .gif file through a socket. ?

From: Manikandan

I am connecting to a server, when connected it sends me a .gif file. in the client program, i receive it in the STREAM format. i use open,read,write system calls to write to a file. i receive as character stream and write it to a file.I do a ftp to win'nt machince but could not open the .gif file. how do i convert a character into binary ?

Mani

From: kb

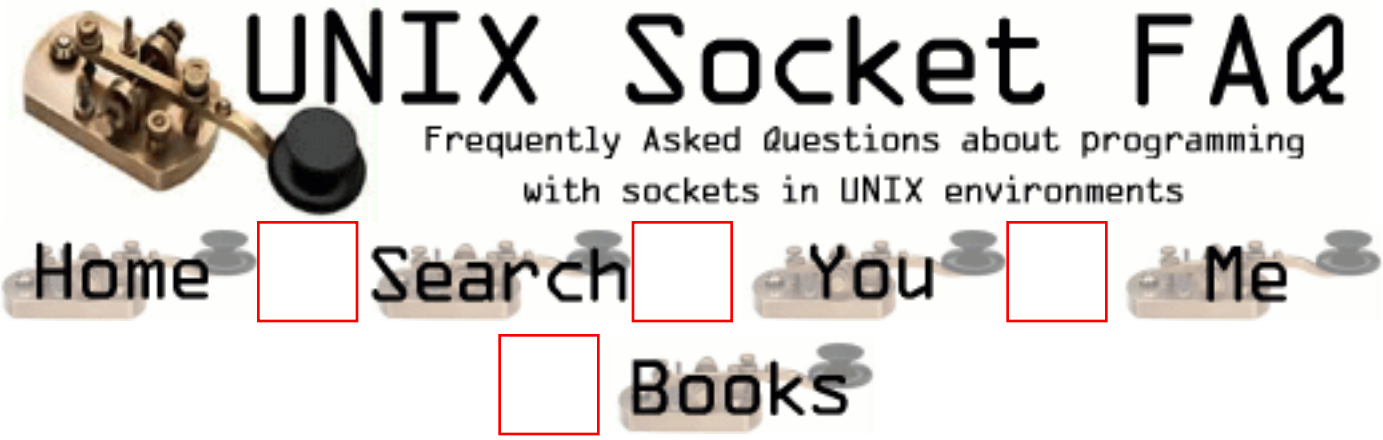
you mentioned a character stream in your description. A gif file is binary -- you should get the file and write it as is. When you ftp it to a windows box be sure to set ftp mode to bin as opposed to ascii

---

From: Manikandan

Yes, what i am doing right now is reading as character stream. i known, .gif file is a binary one, but i am not clear in reading binary data. During ftp i switched to bi mode before put(ing) the file. can u give me a better picture.

Add a Comment



New Questions: About IP Alias

From: rapide.zhang

How can i add an IP alias to my NIC in code?
platform: SunOS 5.7

thanks.

Add a Comment

New Questions: Connection refused when using syslog()

From: Jappe Reuling

Hi,

I'm writing a client/server program both using tcp. The
server binds to a socket and does a select on that socket.
The client connects to the server on the binde socket and sends a tcp packet. So far so good. But now I want to log
events using syslog so I've added the following lines of code thru my source:

```
#include<syslog.h>

...
openlog("MY SERVER NAME", LOG_NDELAY, LOG_USER);
...
printSyslog(LOG_INFO, "Started");
...
closelog;

void printSyslog(int level, char message[SYSLOGLINE]){
  ...
  syslog(level, message);
  ...
}
```

The printSyslog function checks the message var against some
rules but I've let those out.
If I specify the printSyslog line multiple times my CLIENT
gets a "Connection refused", if I leave it to one ore two

there is no problem.

Is this realy a syslog problem? I know syslog uses UDP but
why should this interfere with my client?

From: Rob Seace

I'm guessing there's a subtle bug somewhere in the logging
function's code that's corrupting your memory, and somehow
trashing your socket, or something... I'd need to see more
complete code to be sure, though... If it's not that, then
I don't know... But, it's certainly not a normal/expected
behavior, no...

From: Jappe Reuling

I didn't post all of the code because it is326 lines long...
Here are the most important parts:

Server:

```
  remote.sin_family = AF_INET;
  remote.sin_addr.s_addr = htonl(INADDR_ANY);

  if((ptService = getservbyname("MY SERVER", "tcp")) != NULL) {
    remote.sin_port = ptService->s_port;

  if( (localSocket = socket(AF_INET, SOCK_STREAM, 0)) < 0)

  if(setsockopt(localSocket, SOL_SOCKET, SO_REUSEADDR, &on, sizeof(on)) < 0)

  if(bind(localSocket, (struct sockaddr *)&remote, sizeof(remote)) < 0)

  if(listen(localSocket, QUELENGTH) < 0)

  unsigned int addrSize = sizeof(remote);

  noTimeout.tv_sec = 0;
  noTimeout.tv_usec = 0;

  FD_ZERO(&readFdSet);
  FD_SET(localSocket, &readFdSet);

  if(select(localSocket+1, &readFdSet,NULL, NULL, &noTimeout) < 0)

  if(FD_ISSET(localSocket, &readFdSet)) {
```

```
    if( (acceptedSocket = accept(localSocket, (struct sockaddr *)&remote,
                        &addrSize)) < 0)

    if(read(acceptedSocket, &data, sizeof(data)) < 0)

    close(acceptedSocket);

  }

client:

  remote.sin_family = AF_INET;

  if( (inet_aton("192.168.2.1", &remote.sin_addr)) != 1)

  if((ptService = getservbyname("anounce", "tcp")) != 0) {
    remote.sin_port = ptService->s_port;

  if( (localSocket = socket(AF_INET, SOCK_STREAM, 6)) < 0)

  if(connect(localSocket, (struct sockaddr *)&remote, sizeof(remote)) < 0)

  if( (write(replySocket, AYA_REPLY, sizeof(AYA_REPLY)) ) < 0)

  shutdown(replySocket, 2);
  close(inputSocket);
```

I've let allot of the error handling out of the above code (e.g. below each 'if' there is a function which handles the error).

Hope this makes my problem a bit more clearer.

From: Rob Seace

No, I meant the code for your logging function; printSyslog(), I think you called it... I would guess the bug is in there; but, that's purely a guess, based on what you originally said...

From: Jappe Reuling

Dammn! I ment to include that code, here it is:

```
void printSyslog(int level, char message[SYSLOGLINE], char *extra) {

   if(extra != NULL)
     syslog(level, "%s: %s", message, extra);
   else
     syslog(level, "%s", message);

}
```

From: Rob Seace

Hmmmm, well that's fairly short and sweet; not too much
room to hide a bug, there... The only thing slightly
odd I see is the explicit declaration of your "message" arg
as a sized array, rather than a simple "char*"... I've
seen some compilers do odd things when you specify function
arguments like that, and then feed them a "char*" instead...
They probably shouldn't though... Eg: some people declare
their main() argvs as "char **argv", while others use
"char *argv[]", and either one should really work just
fine... However, when you start throwing in actual array
sizes inside the brackets, I'm not sure if that's even
defined as to what it means or does, in ANSI C...
Basically, I'm saying if *I* were writing this, I'd define
it "char *message", and leave it at that... But, I'm really
not convinced this is your problem, either... I can imagine
a compiler might do something wacky with it as currently
defined (I've seen some pretty wacky compiler behavior, in
my days ;-)), but I wouldn't bet on it... I'd think it'd
more likely simply ignore your given array size, and
basically give you "char message[]", which is basically
exactly the same as "char *message", in a function arg...
But, who knows? It wouldn't hurt to change it to "char*",
and see if that helps... *shrug*

From: Jappe Reuling

I gets weirder and weirder. The problem isn't syslog. When
I replace the complete function with "printf("hello\n");"
the problem is still there but when I replace the complete
function with "cout << "hello" << endl;" it works (no
connection refused).

I've tried to write the messages to a file myself (fputs)
but that has the same problem.

Can anybody explain this?

From: Jappe Reuling

Oke I officialy hate my bos now!!! He came with the smart
remark that maybe the tcp socket wasn't created fast enough.

this was the prev. source code:

```
while(1) {
  create udp socket
  send udp broadcast

  close udp socket
  print to whatever

  create tcp socket
  listen
  act
  close tcp socket
}
```

The client reacts on the udp broadcast but then the tcp
socket hasn't been created due to all debugging stuff.

So I switched to this code which is besides the problem
also better code:

```
create tcp socket
create udp socket

while(1) {

 send udp broadcast
 print whatever
 listen on tcp socket
 act

}

close tcp socket
close udp socket
```

And this solves the problem

Thanks for all your help!!!

From: Jappe Reuling

Oke I officialy hate my bos now!!! He came with the smart remark that maybe the tcp socket wasn't created fast enough.

this was the prev. source code:

```
while(1) {
   create udp socket
   send udp broadcast

   close udp socket
   print to whatever

   create tcp socket
   listen
   act
   close tcp socket
}
```

The client reacts on the udp broadcast but then the tcp socket hasn't been created due to all debugging stuff.

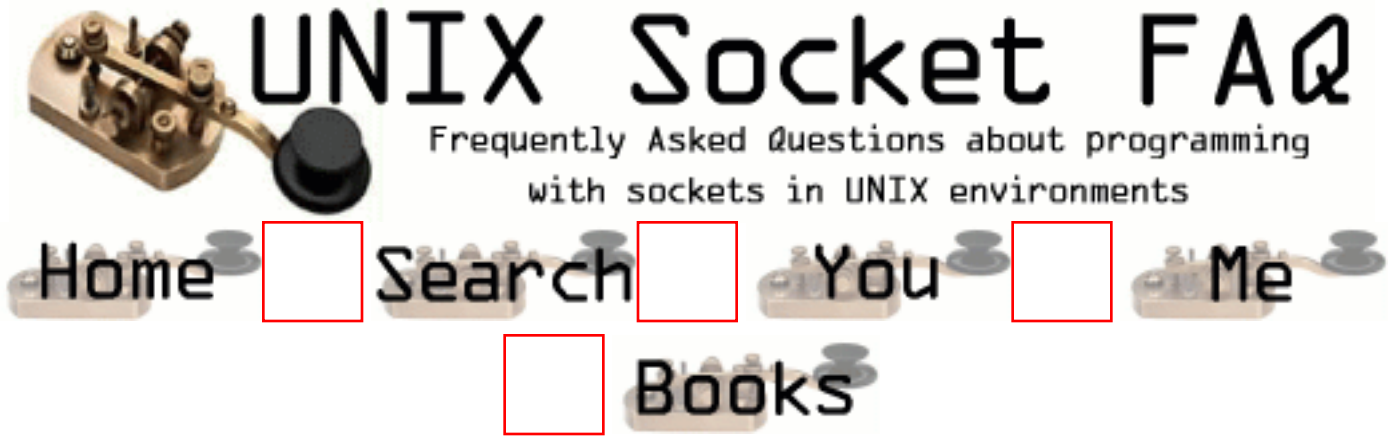
So I switched to this code which is besides the problem also better code:

```
create tcp socket
create udp socket

while(1) {

  send udp broadcast
  print whatever
  listen on tcp socket
  act

}

close tcp socket
close udp socket
```

And this solves the problem

Thanks for all your help!!!

Add a Comment



New Questions: How can we use the LENGTH in recvfrom() in UDP socket?

From: Franz

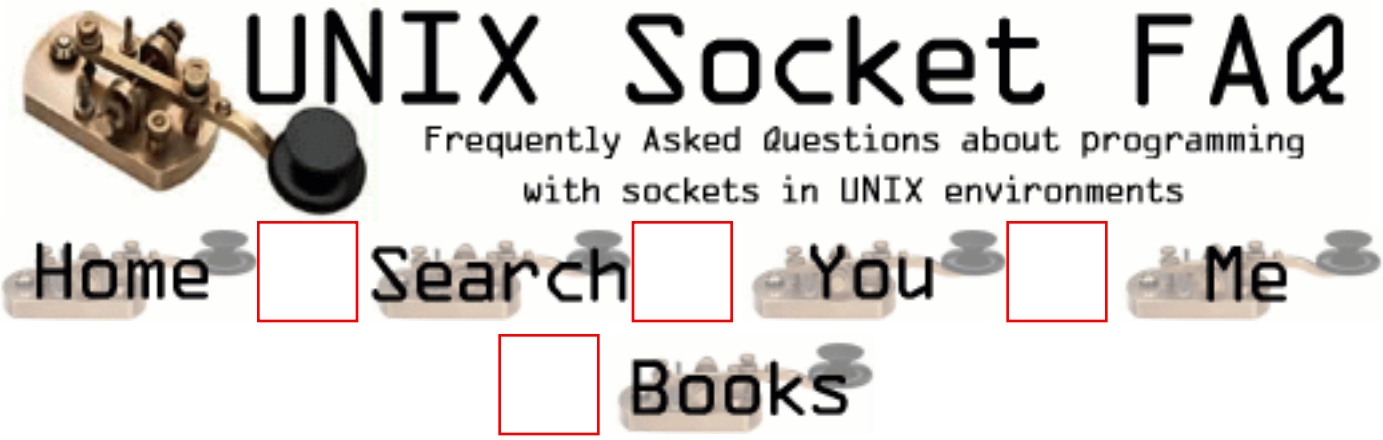
The recvfrom has an atribute: length

But if we don't know the length of data sent by UDP server, how can we fiil this field?

From:

It's the maximum length you're willing to accept (the size of the buffer). How much depends on the protocol that you're using, but 64K is the maximum. But UDP-packets of this size will have to be broken in fragments (size: MTU), and many networks don't allow fragmented UDP-packets.

Add a Comment

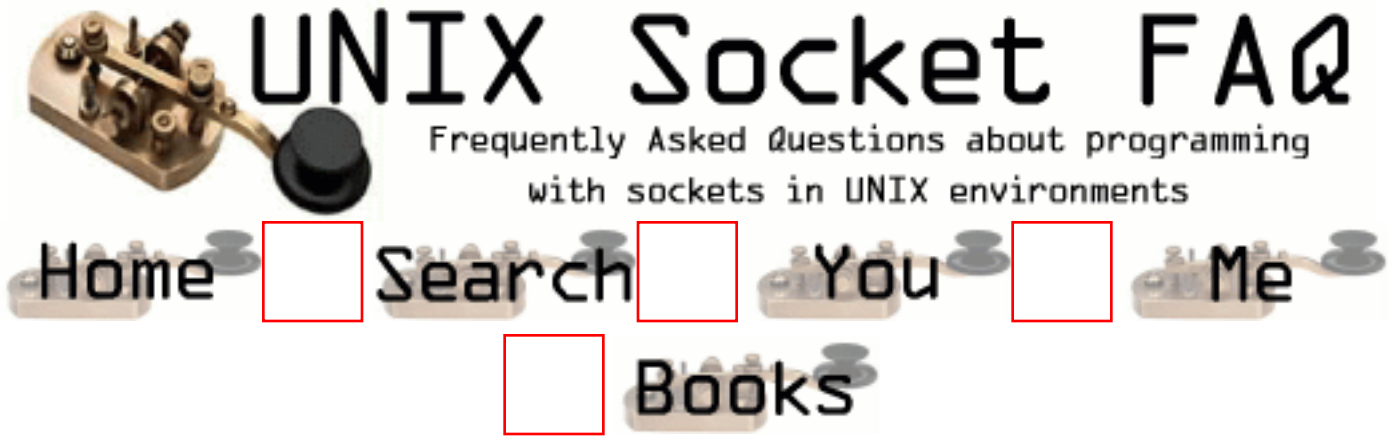


New Questions: Multiple clients using select

From: RaviShankar C S

what exactly is the difference in implementing multiple clients using select statement and by forking.when should select statement be used?

Add a Comment



New Questions: Asynchronous programming

From: kane

I've just been introduced to asynchronous programming... learning mostly from trial and error.

I'm running into a problem where my server is not sending out the entire buffer.
I make a send() call.
I catch the FD_WRITE event and it only shows a fraction of the buffer size being written to the socket.
Then a MFD_DELIVER event is thrown and the bytes delivered is equal to that written to the socket.

From this point on, I'm expecting the socket to eventually send out the rest of the buffer, but it just hangs.

Has anyone ever experienced this problem?
Does anyone have suggestions or ideas about what may be causing this?

Anything would be helpful.

Thanks in advance.

Add a Comment

New Questions: select() problems on DEC Alpha XP900

From: Jan Mannekens

Hi all,

I have a server and a client program, where the server acts as an echo-server, sending back the received packets to the client.
Everything works fine under Linux on my PC, but when I go to my DEC Alpha machine I experience some funny things:

On the server there is a select() with a timeout of 10 seconds and with one single client that is endlessly exchanging packets
the select() timeout is reported every 10 seconds, although it is not expected to timeout.

Any ideas or similar observations?

Greetings,

Jan

From: Loco

Hi,

The only idea i can come up with is to reset the timeval struct before calling select(). If this is the problem (not resetting the variable) then Linux should also fail as it modifies the value to reflect the amount of time not slept.

Post some code to have a better idea.

From: Jan Mannekens

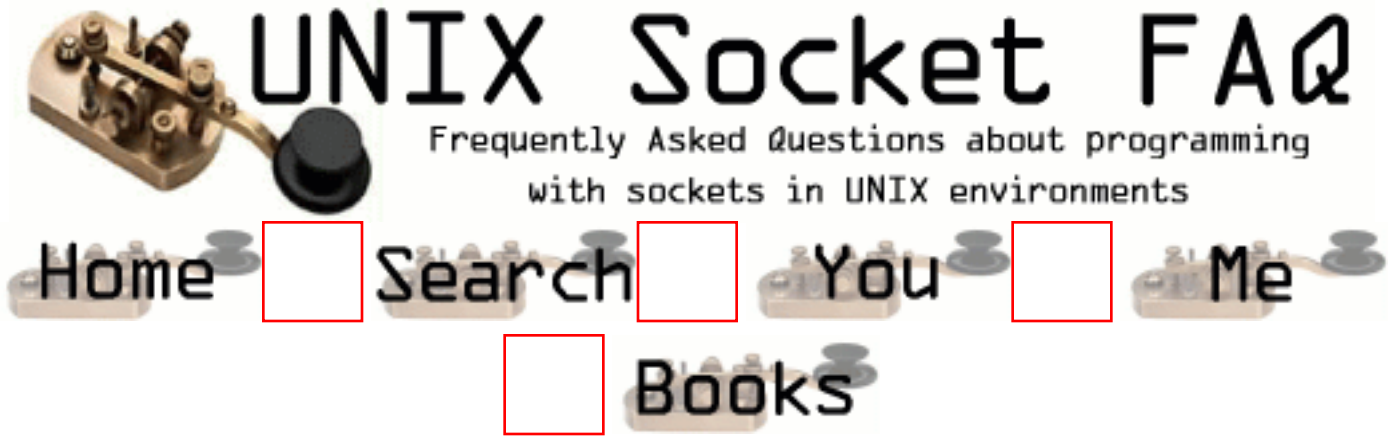
Problem is solved. Had something to do with the first parameter in the select(). I used FD_SETSIZE, which was initialized to 32768 and on DEC this gives a problem (2048 solves my problem).
Funny that the same executable gave different behaviour on 3 different DEC Alpha machines.

From: Loco

You should use a different value for the first parameter to select().
It should be the highest value of the descriptors you are "selecting" plus one.
I don't know if using a higher value could do any harm (as you already wrote it does), but maybe it could make things slower... Gotta go again on the kernel code to see how it is handled...

Good luck
:D HAL

Add a Comment



New Questions: Mapping PDU data over tcp/ip

From: udayan

I have a PDU structure in which some fields are string, some are integer and some are long. I understand that if i have to use the send() system the information in the buffer has to be og character type. How can i send a whole PDU without converting each of the fields into char.. I want to avoid using RPC. Can anybody help

---

Add a Comment

New Questions: aix 4.3 ftp client data socket question (can you help?)

From: brent thomas

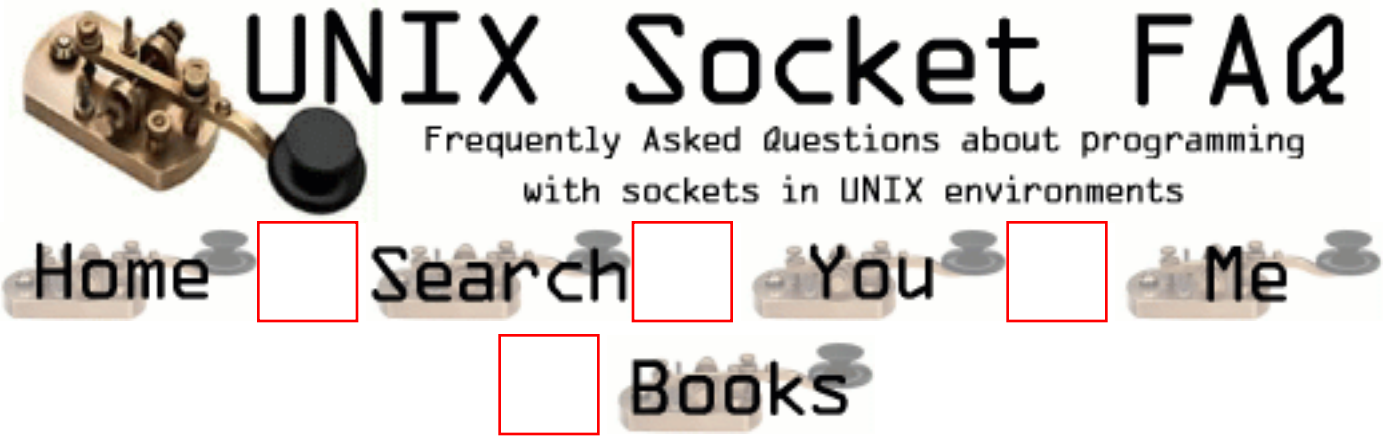
i'm working on an ftp client which talks to a standard ftp server (system ftp)
on an 4.3 aix system. My client opens the control socket,
logs in as a specific user/password, performs a cwd. Then
I bind an emphemeral data socket and call listen on it.
After the listen i take the port information and send it via
the control connection in a PORT command to the server.
then I send a 'STOR file.name' across the control socket to
kick the server to perform the active open to my waiting
data socket. Then an accept is posted to the data socket
which returns with a new socket (0 -- this is unix so 0 should
be a valid socket) -- having done the accept and
gotten the new socket i attempt to send to the socket but
get the indication errno 57 stating that the socket (0)
is not a socket (ENOTSOCK).

my first question is: as the client, after the server has opened me (in the accept)
is there anything else i need to do before attempting to send
to the socket?

my second question is: is there potentially something awry with
the sockets libraries under aix 4.3 that is not viewing the socket (0)
as a valid socket (maybe it thinks its a un-opened file descriptor?)

any thougts or insight appreciated. I'd be glad to email
my test code to anyone who is interested and would like to
take a look.

Add a Comment



New Questions: question

From: ankur

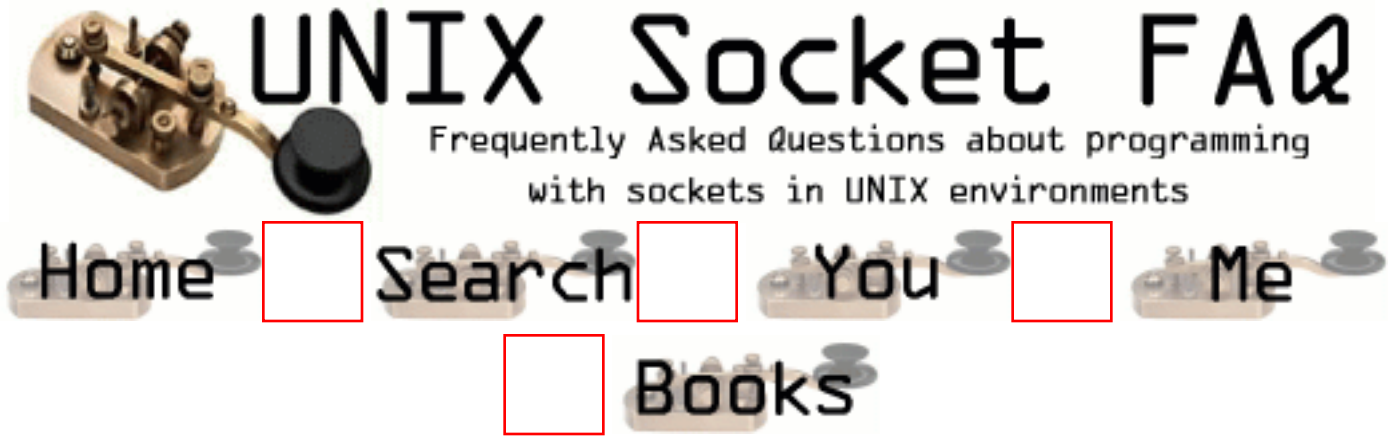
can i open a socket to receive packet from ethernet directly?

From: manikandan

yes, i think, use RAW sockets, which will bypass the tranport layer. i have tried. please try.

Mani

Add a Comment

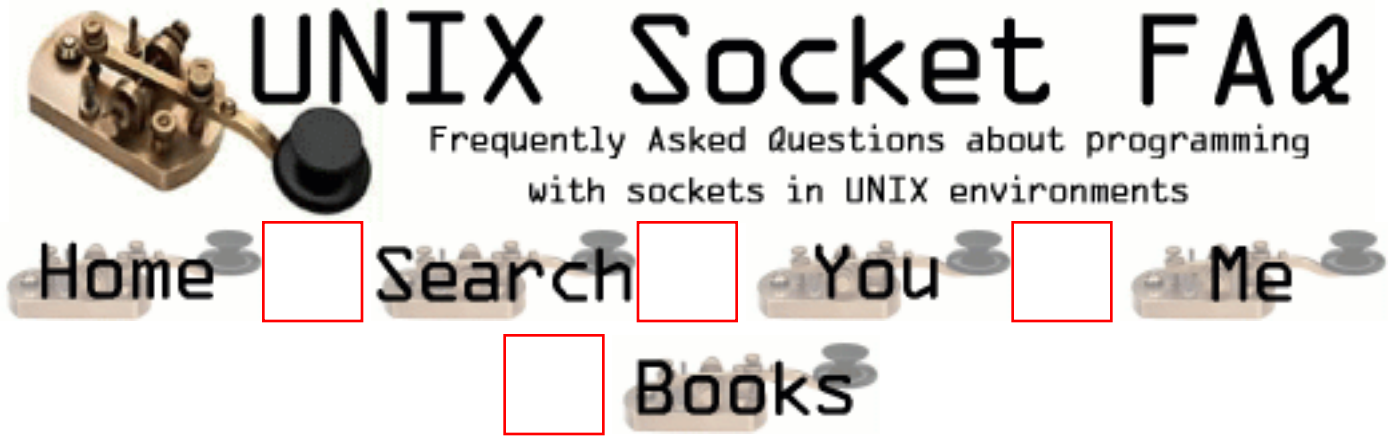


New Questions: unix for submitting html form to a server

From: Digger

Hello,
i need to develop a unix script which works from a server side.
the script needs to submit a html form[a form template] to a particular web server[something like http://sc.satinc.com/s/ss/].
The form has to be submitted as many times as possible per second simultaeously.[you can imagine the program automatically clicking the submit button of around 1000 similar form templates simultaeously].
Someone recommended me unix since it allows multiple threads.ive been told that the unix program used is very basic and simple.
I would appreciate any help from you unix gurus.
Thanks
Digs

Add a Comment



New Questions: RAW Sockets

From: Mike Cramp

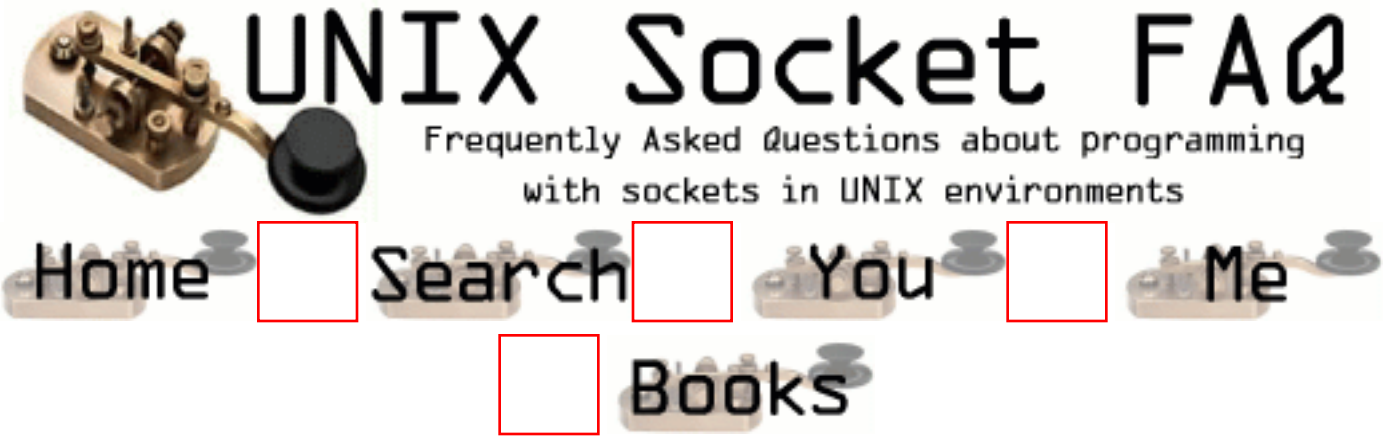
I have a piece of code, socket(AF_INET, SOCK_STREAM, 2) and it won't create the socket if I specify a windows machine. Why not?

From: Loco

Try using

soc = socket(PF_INET, SOCK_STREAM, 0);

Add a Comment



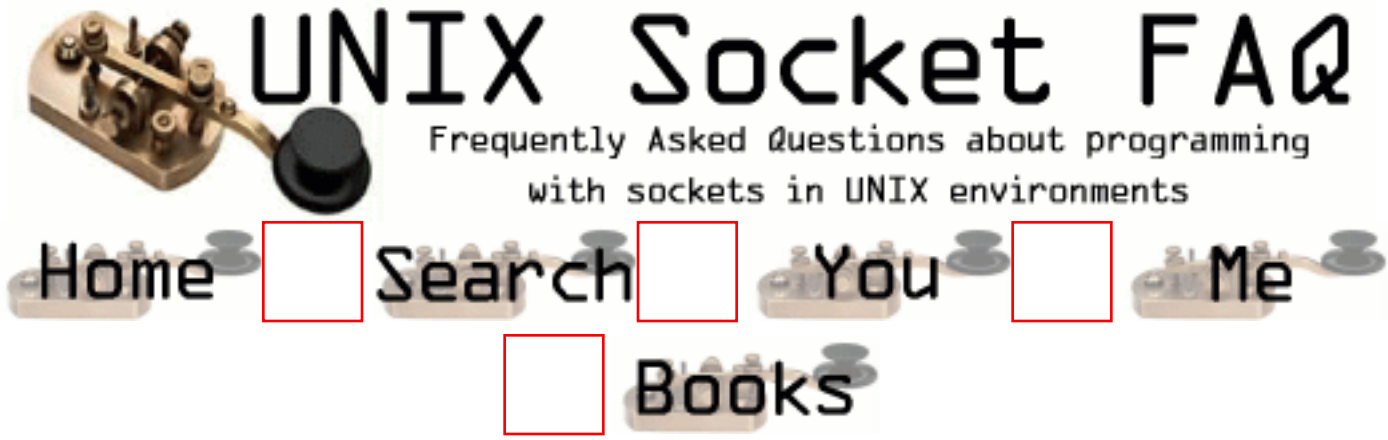
New Questions: counter-wait system

From: Ali Ahmad

I am trying to write a socket client/server counter-wait
which requires the following:
-The srever is started with the command server initial-value
  &
-The client is executed with the command update value.
- The server accepts a positive or negative integer number
from the client, adds it to a counter, waits that number
of seconds, and then returns the new value to the client.
-The client sends a value otained from the command line to
the srver, waits for a return value, and then prints that
value.
I have a littel experience in writing a socket program so
I need some help on how to start.

Thank you

Add a Comment



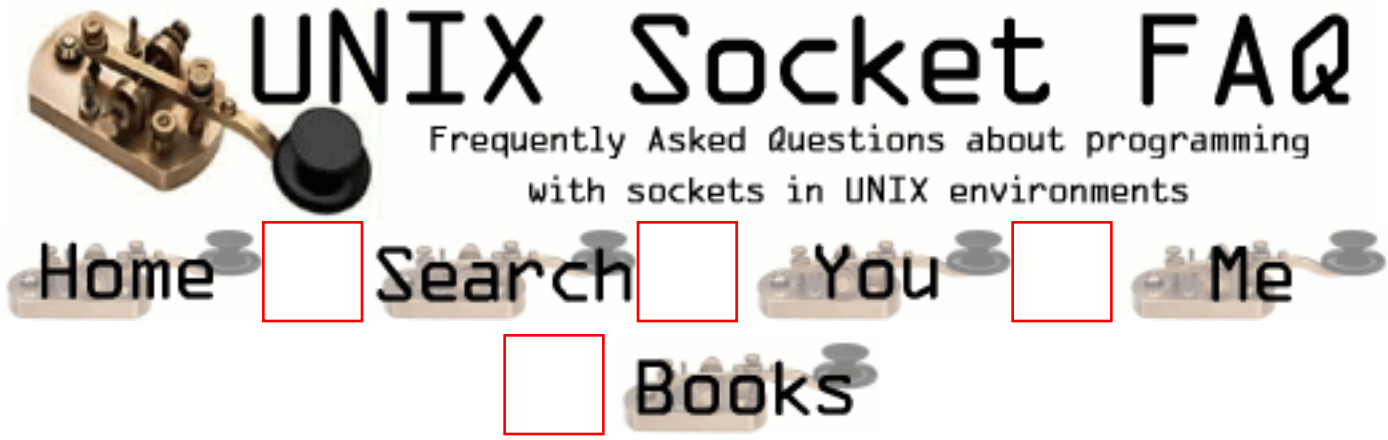
New Questions: Consulting C++

From: Stefan Olsson

Hi!
Our company is developing a multithreaded server in C++ and are now in the optimization phase.
Is there anyone of you that are really into sockets and C++ that would be interested in some consulting work?
The code is isolated to a class that creates both client and server sockets and have methods for sending and recieving data and if successful we would be happy to share the resulting class as GPL

Thanks in advance!

Add a Comment



New Questions: read() hangs w/ binary transfers

From: Cal

Hi everyone,

I'm working on a sort of 'ground up' approach to a file transfer protocol, just for fun. I have an issue with read() hanging after the file is completely transfered. If I ctrl-C on the server side, and the socket closes, then read() will exit giving an EOF. I could establish a completely separate socket connection for file transfers, (I also use a command line interface over the socket) but that seems like it could be a hassle with authentication. I have also tried using dup() on the server socket, and closing the dup'ed socket, but that didn't seem to work either.

Here's a snippet of the file transfer code that I'm using
(sorry if this post is long)

Server

```
  /* This is where the server recieves
the 'get' command from client */
rfp = open(cmd2, O_RDONLY);
     if (rfp < 0)
     {
      perror("open()");
     }

     printf("writing\n");
     while(1)
     {
      readsize = read(rfp, buffer, sizeof(buffer));
```

```
      if (readsize == 0)
      {
        printf("encountered End of File\n");
        break;
      }
      if (readsize < 0)
      {
         perror("read()");
         break;
      }
      writesize = write(connectsock, buffer, readsize);
          /* Note: using readsize for count bytes because
            we may not read the full buffer for the last read.
            therefor, write() will only write the amount of bytes
            that we have read to the socket
            Note2: this is setup exactly like client side
            (except reversed) ... don't know if both need
            to cut off the buffer, or if one needs to use
            something like sizeof(buffer) for count.
          */
      if (writesize == 0)
      {
          printf("writesize is 0");
          break;
      }
      if (writesize < 0)
      {
        perror("write()");
        break;
      }

    }
    close(rfp);
    printf("write complete... file closed");
```

Client

```
  /* Open local file on client */
wfp = open(cmd3, O_WRONLY|O_TRUNC|O_CREAT, 0666);

  if (wfp < 0)
    {
      perror("open");
```

```
  fprintf(stderr, "could not open file %s", cmd3);
}
   /* This starts the transfer code server-side */
if ( sock_puts(connectsock, sndbuffer) == -1)
{
  perror("sock_puts()");
}
while (loop)
{

 readsize = read(connectsock, readbuffer, sizeof(readbuffer));

   // debug
   printf("%d", readsize);
   if (readsize < sizeof(readbuffer))
   {
      /* we're on the last read */
      loop = 0;
   }
   if (readsize == 0)
   {
     printf("end of file\n");
     break;
   }
   if (readsize < 0)
   {
      printf("read error\n");
      perror("read()");
      break;
   }
   printf("."); // print a '.' for a successfull sock read
  writesize = write(wfp, readbuffer, readsize);
      /* Note: using readsize for count bytes because
         we may not read the full buffer for the last read.
         therefor, write() will only write the amount of bytes
         that we have read from the socket
      */
   if (writesize == 0)
   {
     printf("writesize is 0");
     break;
   }
   if (writesize < 0)
   {
     printf("write error\n");
```

```
        perror("write()");
        break;
      }
      printf(":"); // print a ':' for successfull file write
      readsize = 0; // for debug purposes
    }
    close(wfp);
    printf("file get complete... file closed\n");
```

The file transfers perfectly fine, but I can't get read() to exit on the client side. I have read through the faq, but I don't know what the best solution is: either use alarm() or somehow timeout the read() function, or use non-blocking mode. I've read through some info about using select() but I havn't really seen any good examples using it. Could anyone provide a link to some good examples? Any suggestions at all would be greatly appreciated.

Thanks for your time!

-Cal

From: Loco

Hi,

I wouldn't use "sizeof(readbuffer)" as the parameter to read(), as it will return the size of a char*, not the real size of the buffer. Use a constant or a variable containing the buffer's size. It will greatly increase your application performance.

It's the same in the server with "sizeof(buffer)".

Don't expect to read the same number of bytes that were written by the other party. It is possible that you have to issue multiple reads for a write, even if the buffers' capacities are the same. So don't count on it to know when the transmission ends.

You can do other things such as:

```
--- SENDER ---
fd = open(...);
fstat(fd, &st_st);
sprintf(buffer, "%05d", st_st.st_size);
write(soc, buffer, 5); // Number of bytes to send...

bread = read(fd, buffer, MAX_BUFFER_SIZE);
while (bread > 0) {
  write(soc, buffer, bread);
```

```
  bread = read(fd, buffer, MAX_BUFFER_SIZE);
}

--- READER ---
read(soc, buffer, 5);
buffer[5]=0;
size = atoi(buffer); // Number of bytes to read
up = 0;
do {
  bread = read(soc, buffer, MAX_BUFFER_SIZE);
  write(fd, buffer, bread);
  up += bread;
} while (bread < size);
-----
```

I haven't included any error checking, but that is the idea...
Just to make sure the reader will receive 0 when reading do shutdown() to the socket before closing. (You may have another file descriptor for the same socket and that could be the problem: close() releases the file descriptor, and if it is the last descriptor to reference the resource then closes the associated resource.) This usually happens when the descriptor is inherited from the parent process, and the parent does not close it's own descriptor after forking... If this is the problem then shutdown() will do:

```
--- SENDER ---
fd = open(...); // Open file for reading

while (1) {
  bread = read(fd, buffer, MAX_BUFFER_SIZE);
  if (bread <= 0)
    break;
  write(soc, buffer, bread);
}
shutdown(soc, 2);
close(soc);
close(fd);

--- READER ---
fd = open(...); // Open file for writing
while (1) {
  bread = read(soc, buffer, MAX_BUFFER_SIZE);
  if (bread <= 0)
    break;
  write(fd, buffer, bread);
}
shutdown(soc, 2);
```

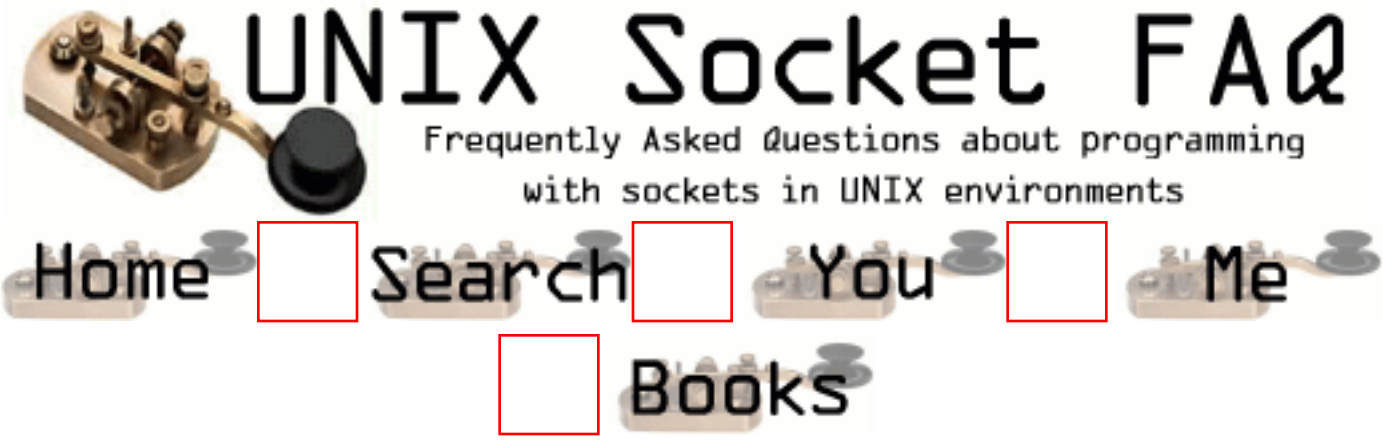
```
close(soc);
close(fd);
```

As you can see both codes are almost identical. NO error catching included!

I hope this helps.

:D (HAL)

Add a Comment

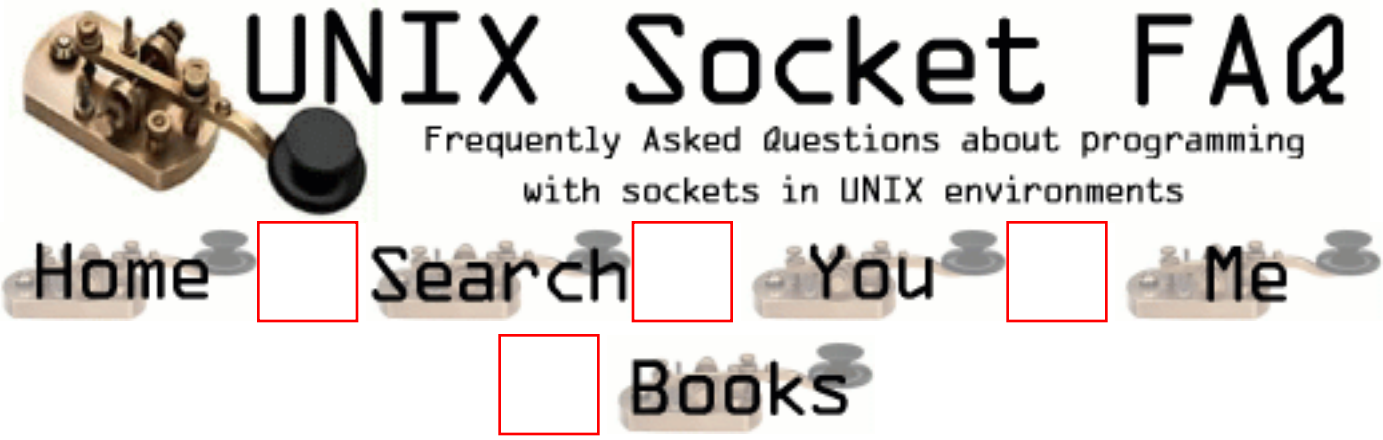


New Questions: Asynchronous DNS lookups

From: Chris

Hi,
I know that the gethostby*() functions can block execution, and I was wondering, what's the "common solution" to this? I guess you could fork() or create a thread, or use alarm() or something..
What solution(s) do you use to get around this problem?
Thanks in advance,
Chris

Add a Comment

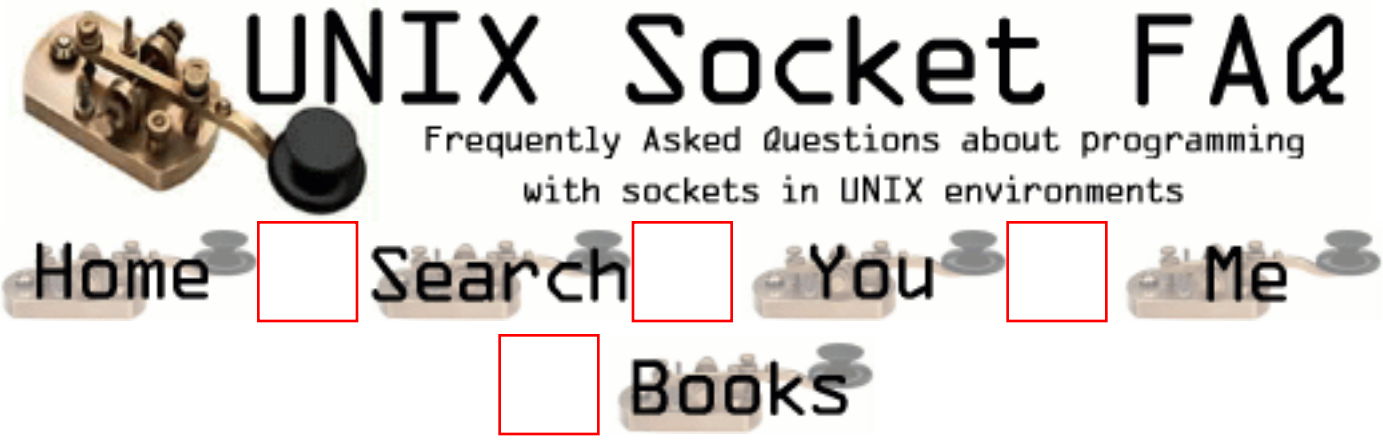


New Questions: When does EPROTO occur?

From: manikandan

When does EPROTO occur. Is EPROTO and EINTR are same.
Please explain.
Thanx
Mani.

Add a Comment



General Information and Concepts: What's new?

You can find out what is new by looking at the main page for questions that have been recently submitted. You can also set up a user profile for yourself that will allow the main page to tell you what questions have been added, and which questions have new comments since your last visit.

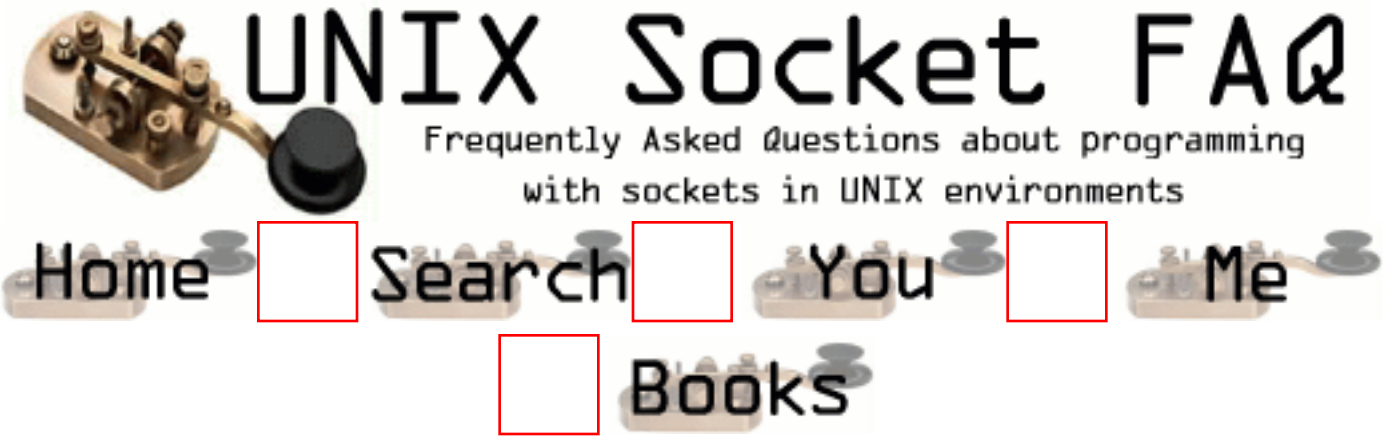
From: Georg Wagner

Question for FAQ: How do I restrict a socket to a specific interface?

From: Xidong Wang

when bind, use the addr of that interface

Add a Comment



General Information and Concepts: About this FAQ

This FAQ is maintained by Vic Metcalfe (vic@acm.org), with lots of assistance from Andrew Gierth (andrew@erlenstar.demon.co.uk). I am depending on the true wizards to fill in the details, and correct my (no doubt) plentiful mistakes. The code examples in this FAQ are written to be easy to follow and understand. It is up to the reader to make them as efficient as required. I started this faq because after reading comp.unix.programmer for a short time, it became evident that a FAQ for sockets was needed.

The FAQ is available at the following locations:

Usenet: (Posted on the 21st of each month)

news.answers, comp.answers, comp.unix.answers, comp.unix.programmer

FTP:

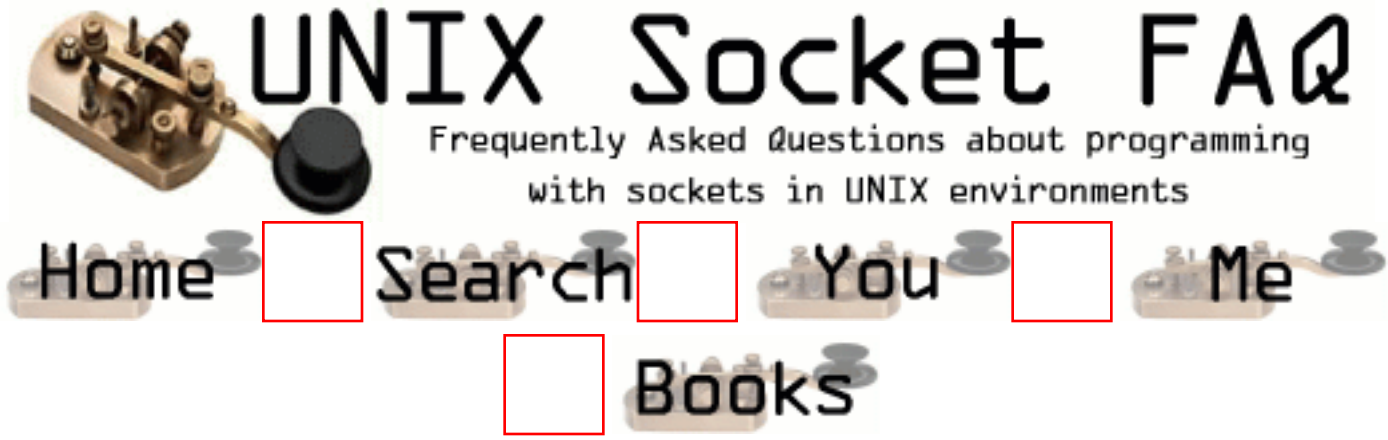
ftp://rtfm.mit.edu/pub/usenet/news.answers/unix-faq/socket

WWW:

http://www.ibrado.com/sock-faq
http://kipper.york.ac.uk/~vic/sock-faq
http://www.ntua.gr/sock-faq

Please email me if you would like to correct or clarify an answer. I would also like to hear from you if you would like me to add a question to the list. I may not be able to answer it, but I can add it in the hopes that someone else will submit an answer. Every hour I seem to be getting even busier, so if I am slow to respond to your email, please be patient. If more than a week passes you may want to send me another one as I often put messages aside for later and then forget about them. I'll have to work on dealing with my mail better, but until then feel free to pester me a little bit.

Add a Comment

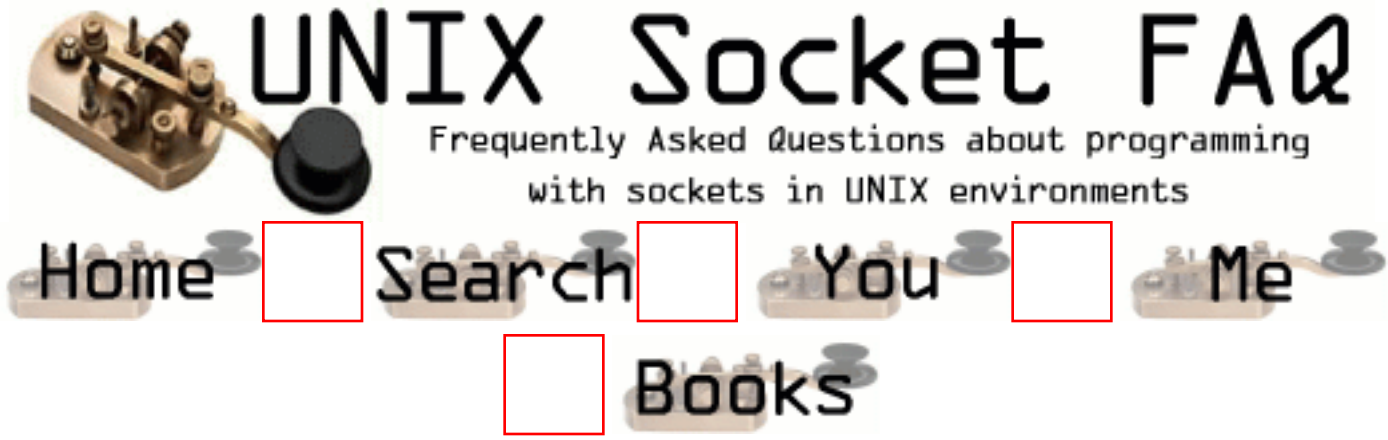


General Information and Concepts: Who is this FAQ for?

This FAQ is for C programmers in the Unix environment. It is not intended for WinSock programmers, or for Perl, Java, etc. I have nothing against Windows or Perl, but I had to limit the scope of the FAQ for the first draft. In the future, I would really like to provide examples for Perl, Java, and maybe others. For now though I will concentrate on correctness and completeness for C.

This version of the FAQ will only cover sockets of the AF_INET family, since this is their most common use. Coverage of other types of sockets may be added later.

Add a Comment



General Information and Concepts: What are Sockets?

Sockets are just like "worm holes" in science fiction. When things go into one end, they (should) come out of the other. Different kinds of sockets have different properties. Sockets are either connection-oriented or connectionless. Connection-oriented sockets allow for data to flow back and forth as needed, while connectionless sockets (also known as datagram sockets) allow only one message at a time to be transmitted, without an open connection. There are also different socket families. The two most common are `AF_INET` for internet connections, and `AF_UNIX` for unix IPC (interprocess communication). As stated earlier, this FAQ deals only with `AF_INET` sockets.

From:

From: cara

do you mean to say a socket is a stream that is created for
data to flow between your client application and server
application.
could you explain in detail the exact meaning of socket, is it an
actual physical stream, a software package, a function or what?

From: pepe

No me sirvio para nada esto!!!

From: Reshma

Bull Shit!!

From: Syndicate

Picture a socket (AF_INET) as a imaginary hole in your computer.. Other computers can connect to those holes. or you can use those holes to conenct to other computers. its a way to communicate across a network.

From: Jose

Great!! But, what are Sockets? Your answer is not an answer for me.

From: Jose

I'm back. I found a better answer over the internet at http://www.calweb.com/~chriss/tech/socklink.html. It looks like this: "A socket is an endpoint for communication. Two cooperating sockets,
one on the local machine and one on the remote machine, form a connection. Each of the two sockets has a unique address that is described generically by the 16-byte sockaddr C programming language structure."

From: Haru

I like Jose's answer better. But I still don't have a clear idea. Could someone give a more detailed answer with perhaps an example. Thanks.

From: dan

A socket is kinda like a virtual adapter between a program and a network protocol. Consider an ftp server going online, one socket is created by the ftp server program. This socket's jobs are to listen at TCP port 21 and if IP passes any packets to TCP port 21, the socket will pass them on to the ftp server program.

From: Sarah

Oh! Dan's example makes sense. Thanks.

From: JK

Well, it is little convincing me what socket exactly is ?
Can anybody better explain with a simpler program and a better example with it?

Thanks

From:

A simple explination of a socket is two or more processes (programs running) wanting to share information. A socket is used to send or recieve data directly to or from another process without one having to write to a file and the other read the data from the file.

From: boby

Think of socket as a electric socket in your house.

its the point thru which electron(data) enters your appliance
(program). Now each socket has a uniquie id.(say you get billed for
you usage of electricity bcoz your socket or connection has
a unique id). hope this helps.....

From: Mahafuz

In plain words, socket is the communication tunnel/channel
endpoint used by your program.

Technically, Socket is the file system like interface
(file desriptor) provided by the OS, wrapping your communication point. Through socket you send / receive data to other programs (usually over network).

pardon my english.

From: Mahafuz

Adding to my last comment,

Like file, a socket can be opend written to and read from.
Only special thing that you bind a socket to a process/program in a machine (server point of view) or you connect to a socket (client point of view). As a result after inding & connecting its like a pipe. When server writes it it client can read what server sent or the other way round.

This is a general picture with errors in it.

From: Helge

Take a look at this:
http://www2.hursley.ibm.com/rexxtut/socktut2.htm

From:

Did I misspell descriptor, and what is it? I need a definition>
WRobb

From: Rob Seace

A descriptor is simply a small integer value, which serves
as a sort of index into a process's list of open files...

Whether or not it is truly an "index" in any implementation
sense is irrelevent... It can just be thought of as an
opaque identifying token, which represents a particular open
file (or socket, or pipe, etc.)... You pass around this
descriptor to all I/O functions, and they in turn pass it on
to the kernel, which knows what real open file is associated
with a given descriptor...

From: Emily

so is socket a physical object that i can see? or is it a virtual object?

From: Rohan

A socket is basically a tool used to communicate.
It is an abstraction of the communication hardware.

From: Annonymous Coward

Come on folks, if you can't understand what this is saying, not to mention all the other explanations here, you really shouldn't be trying to program. You should continue your life as a pathetic end-user, wandering around asking people who don't have time to answer a question that answers itself to explain to you why your computer won't turn on when you don't have it plugged in... Go Away!

From: Cowards best friend

I agree with Anonymous Coward

From: yahoo onlooker

You bunch of yahoos couldn't explain a thing in simple english because you don't really know how it works.

Tell me once you create a socket in a program, how does that data stream pass to the communication buffer area and distinguish itself with other communication processes.

From: Sivapriya

But what is the physical (or atleast the logical)
structure of a socket?

If it is an abstraction to the communication hardware then what is that communication hadware and tell me how that is implemented.

From: FUCK YOU COWARDS

Hey, it is obvious that the people that are looking for answers here are LEARNING these things you fucking dork. You obviously are a great programmer because you have no people skills at

all. I hate people like you with a fucking passion. Go ahead with your cowardly life sitting in front of a computer while all of us "end-users" are still having fun. Blow me you pathetic piece of shit.

From: Question

Is socket the same as port ?

From: Coward hat

From: Ravi

what type of signals do socket convey through them
is it analog or digital

From: mike

Can you have multiple sockets over a single port? I.e. is it safe to say that when i have multiple netscape windows open, that each of these creates a socket which communcates over the TCP port?

From: caspre

Can you have multiple sockets over a single port? I.e. is it safe to say that when i have multiple netscape windows open, that each of these creates a socket which communcates over the TCP port?

From: joanna

Nothing understandable upto now ...no clear definiton with
clear examples are provided.

From: Chris

Simple. Socket = IO on a port.

From: Chris

Simple. Socket = IO on a port.

From: Chris

Pick a port. 9960.
A socket has three streams: In, Out, and err.
How many sockets can you have open on a single port?
I dunno.. how much ram/bandwidth you got? The answer is 'theoretically' unlimited.

How many streams can you have on a socket *3*. PERIOD. If you have a multithreaded server listening on a port for client connections and you try
making all the clients use the same streams on the same socket,

you are in for some bad mojo. It is all quite simple. Remember,
port is just the address. INFINITE sockets on a port. *3* Streams on a socket.

From: Loco

I think you have IN, OUT, OOB in, OOB out streams. I've never heard about an "err" stream in a socket but i may be wrong.

About the number of sockets you can open in a single port:
In TCP and UDP you can only bind one socket to one port and address... If you have multiple addresses, each one of them can have one socket bound to each port.

However, if your clients connect to a port bound on your machine it will have as many sockets as clients connected to that port plus the bound socket. It doesn't mean that you can bind many sockets on that port.

The number of sockets connected to a port doesn't depend directly on bandwidth but on file descriptors and memory available. There are many discussions about this in the FAQ and i don't want to begin another one, so i suggest you read the questions.

What can i say about sockets?
It is a programming interface that is used to open communication pathways between two (probably different) or more processes which are usually (but not necessarily) running in different computers. You can create connection oriented sockets, packet oriented sockets, and even raw sockets that let you work on all or part of the information used by the underlying protocol. With connection oriented sockets you create a set of pipes between two parties, these pipes assures reliable transmission of the information (for example: packet reordering if the protocol divides the data that is sent in many packets and it is received in a different order) no packet boundaries (it means that you can send a lot of information in just one funciton call without worrying about the size of the packet). The datagram oriented sockets just allow you to send and receive packets, which are not guaranteed to arrive at their destination, you don't receive confirmation of their arrival, and can be in a different order of that they were when they were sent, as well as be duplicated (i've seen this, some routers duplicate broadcast packets) Finally, raw sockets are something like datagram sockets but they allow you to do crazy stuff on the packets of the underlying protocol (for example change header flags of a packet, even play with the source address and alike).

The interace consists of sockets descriptors, some structures, and a set of functions to perform operations on the descriptors.
The socket descriptor is just a file descriptor which allows you to use some of the functions normally used on file descriptors (such as read and write). As this interface shields the programmer from the network communication details you don't need to understand what lies beneath to program sockets.

Programming sockets is not difficult for doing simple tasks. Using TCP sockets (TCP/IP

protocol, STREAM sockets) involves two parties: the server and the client. The difference between the parties is that the server creates the socket, binds it to a specific port, signal that it can receive connections, and waits for those connections; while the client just creates the socket and connects to the server to the port it is waiting to receive connections.
The server normally expects some clients will be connected to it, so it has to give some way for them to find it. That is why it uses one socket to receive connections, and for each connection it uses a separate socket, so it will have as many sockets as the number of clients connected.

The server side code would look like this:

```
int srvsock; // A socket descriptor is just an int
int peersock; // You need a place to hold your peer sockets
int len;
struct sockaddr_in address;
// Create descriptor
srvsock = socket(AF_INET, SOCK_STREAM, 0);
address.sin_port = htons(8080); // Bind to por 8080
address.sin_addr.s_addr = INADDR_ANY;
address.sin_family = AF_INET;
// Bind socket to specific port and address(es)
bind(srvsock, (void*)&address, sizeof(address));
// This socket shall be a server
listen(srvsock, 10); // 10 is queue length

// Wait for new connections
do {
  len = sizeof(address);
  // Here address is used to receive the address of the
  // connecting party
  peersock = accept(srvsock, &address, &len);
  // peersock is the socket descriptor used to communicate
  // to a specific client
  // In this example, i just send a message, close the
  // connection and release the file descriptor
  send(peersock, "Hi there\n", 9, 0);
  shutdown(peersock, 2);
  close(peersock);
} while (1)
```

This basically is the code everybody uses in a server.

For more information i suggest you read the man pages.
One last thing: sockets are not bound to any especific protocol, but they are used as an abstraction to many protocols that share similar functionalities. I have used X.25 sockets, TCP/IP sockets (UDP and TCP), and have read about UNIX sockets, and NETLINK sockets (in Linux)

:D (HAL)

From: Hector Lasso

Loco,

All sockets have an error queue, which can be read using recvmsg() with the MSG_ERRQUEUE flag. Maybe this is what Chris was talking about.

From: Vinny

Why are sockets used ?? What kinda applications would need to use sockets (other than FTP, Telnet, Web Browsers) ? Can I use sockets to do inter process communication between two process connected by ethernet and use sockets to issue commands to the client process which will process it and return status back ? Can i trnasfer huge files 64MB using sockets ? Is it used as a layer over TCP/IP, so that we dont have to use the complex TCP/IP specification ?

From: Geetha Ramanathan

When you create a socket of type raw socket, how do you specify the address of the other end when trying to send data using the send API. In the same way, how do you specify from which process it has to receive data in the recv command.

Thanks,
Geetha

From: arun

A Socket can be assumed as a communication tunnel established between two comm ports for data to be transfered. Socket is not a physical entity but a virtual one established when IP addess is being initialized for a network or node

---

From: Practic

I am a novice programmer so I do not really understand the inner workings of sockets myself. However, I believe I can explain what one is in simple terminology. Think of this: say you have two programs that are either on the same computer, or on different computers, and they want to communicate. How can they do this? They can't call functions in one another like one object to another within a program. They need some sort of common reference point. One way would be that they communicate by writing to a file. They both have a pre-agreed 'physical' file on a hardisk. They both write to it and read to it, with possible proper headers. However, writing to a hard disk is slow. So instead, they can both communicate through the operating system on the computer. What sockets are (I think) is basically a point of reference managed by the operating system. Basically when you want to 'listen' to a socket, your program tells the operating system (by a function call) that it wants to listen to a specific socket, identified by an integer. Then,

whenver another program writes information to that socket, ie, sends the operating system that information with the attached idea of "send this info to the socket", the OS then checks its list of listeners to that socket, and then sends the data that is to be written to the socket to all the programs 'listenig' to that socket. That is, in simple terms, what a socket it. It is merely a centralized mechanism in the operating system by which different programs communicate. It is not 'physical'. It is not on the hard disk, etc. It is basically one program telling something to the operating system, which then the operating system sends to any program that wants to listen.

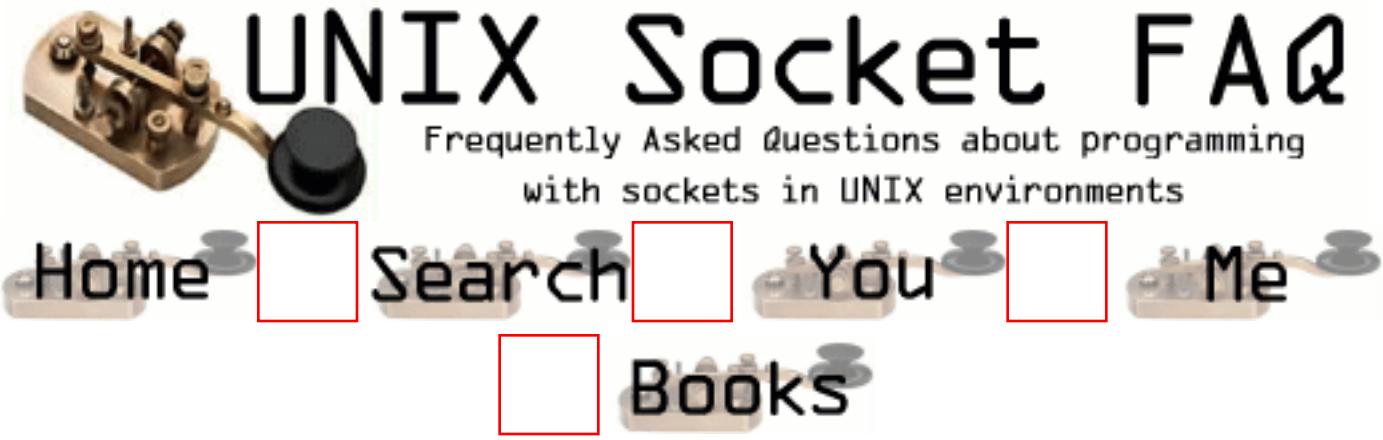
From: Lamer

Could somebody please explain how the
MSG_ERRQUEUE stream work?

From: Rahul

I like practic's explanation. Very straightforward.

Add a Comment



General Information and Concepts: How do Sockets Work?

The implementation is left up to the vendor of your particular unix, but from the point of view of the programmer, connection-oriented sockets work a lot like files, or pipes. The most noticeable difference, once you have your file descriptor is that `read()` or `write()` calls may actually read or write fewer bytes than requested. If this happens, then you will have to make a second call for the rest of the data. There are examples of this in the source code that accompanies the faq.

From: Stanislav Shalunov

Actually, no standard I am aware of prohibits short `read()`'s and `write()`'s for pipes or even regular files.

From: Pinocchio

So if sockets can do more than pipes, why does anyone use named pipes?

From: S.Murali Krishna

What is the difference between FIFOS and Unix domain
sockets. Both of them are doing the same job (interprocess
Communication) then what is the difference (or advantage)
between them ? Can anyone tell me in detail.
Thanks - S.M.K

From: Rob Seace

Well, the main difference between a named pipe (FIFO) and
a Unix domain socket is that pipes are uni-directional
(half-duplex) while sockets are bi-directional (full-
duplex)... So, to communicate in both directions via a
named pipe, you'd need TWO of them, one for each direction;
or, you'd need to arrange some very careful read/write

synchronization on both ends, to keep from stepping on each other's toes by simultaneously trying to write, or deadlocking from simultaneously trying to read... But, an AF_UNIX socket has pretty much all of the normal TCP/UDP semantics and behaviors you would expect from an AF_INET socket; just with no network lag... ;-) And, with a few added features (like being able to pass file descriptors and credentials to peers)...

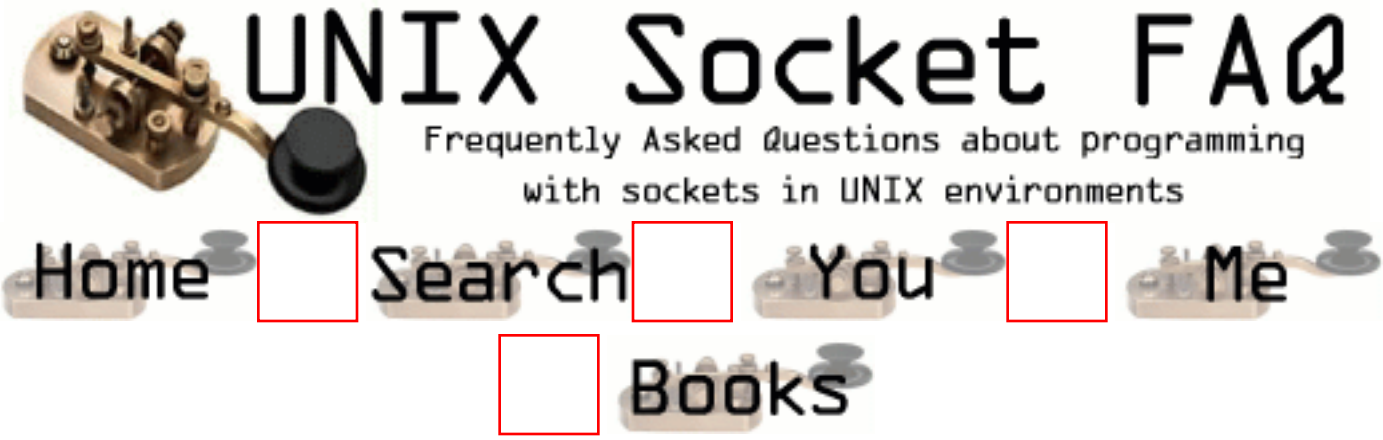
However, one advantage named pipes have is that they work transparently as if they were real files... Ie: if you have an app that only knows how to read from or write to a file, it's trivial to make it use a named pipe instead... But, in order to make it use a Unix domain socket (or any other kind of socket, for that matter), you would need to rewrite it to do all of the proper socket API calls, rather than simply doing fopen() or whatever it was already doing...

From: SpoofedAddress

//////////////////////////////////////\\\\\\\\\\\\\\\\\\\\\\\\ How Do You Kill a Dog Thats Been sucking your cock when it feels too good to hurt the dog? also Why Do You people give blowjobs to Bill Gates

---

Add a Comment



General Information and Concepts: Where can I get source code for the book [book title]?

Here is a list of the places I know to get source code for network programming books. It is very short, so please mail me with any others you know of.

Title: Unix Network Programming
Author: W. Richard Stevens (rstevens@kohala.com)
Publisher: Prentice Hall, Inc.
ISBN: 0-13-949876-1
URL: http://www.kohala.com/~rstevens

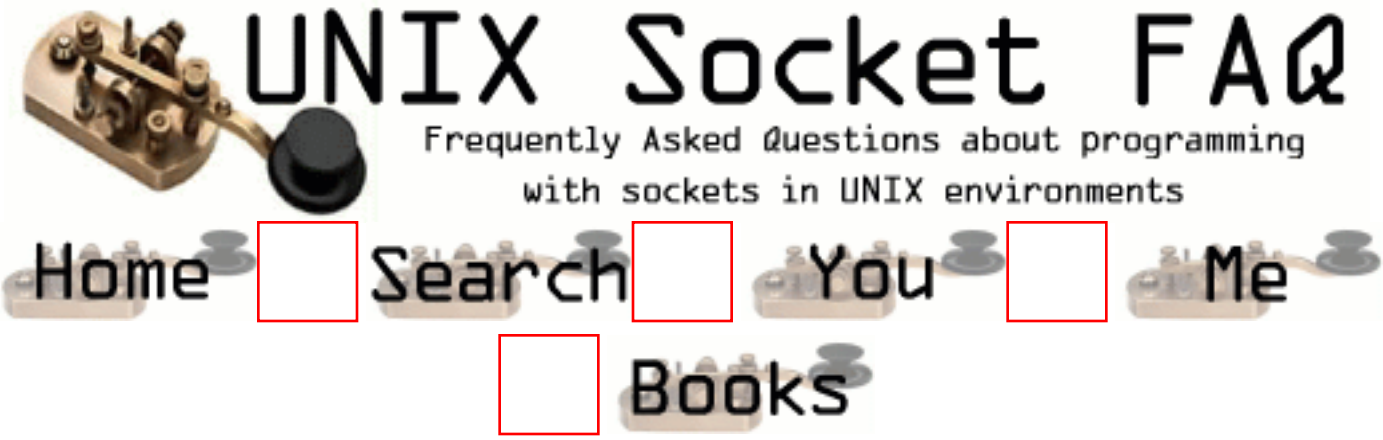
Title: Power Programming with RPC
Author: John Bloomer
Publisher: O'Reilly & Associates, Inc.
ISBN: 0-937175-77-3
URL: ftp://ftp.uu.net/published/oreilly/nutshell/rpc/rpc.tar.Z

Recommended by: Lokmanm Merican (lokmanm#pop4.jaring.my@199.1.1.88)
Title: UNIX PROGRAM DEVELOPMENT for IBM PC'S Including OSF/Motif
Author: Thomas Yager
Publisher: Addison Wesley, 1991
ISBN: 0-201-57727-5

From: Tina Nylund

Just a small thing - the source code for Stevens' book has
moved to: http://www.kohala.com/start/unpv12e.html

Add a Comment



General Information and Concepts: Where can I get more information?

I keep a copy of the resources I know of on my socks page on the web. I don't remember where I got most of these items, but some day I'll check out their sources, and provide ftp information here. For now, you can get them at http://www.ibrado.com/sock-faq.

There is a TCP/IP FAQ which can be found at http://www.dc.net/ilazar/tcpipfaq/default.htm

---

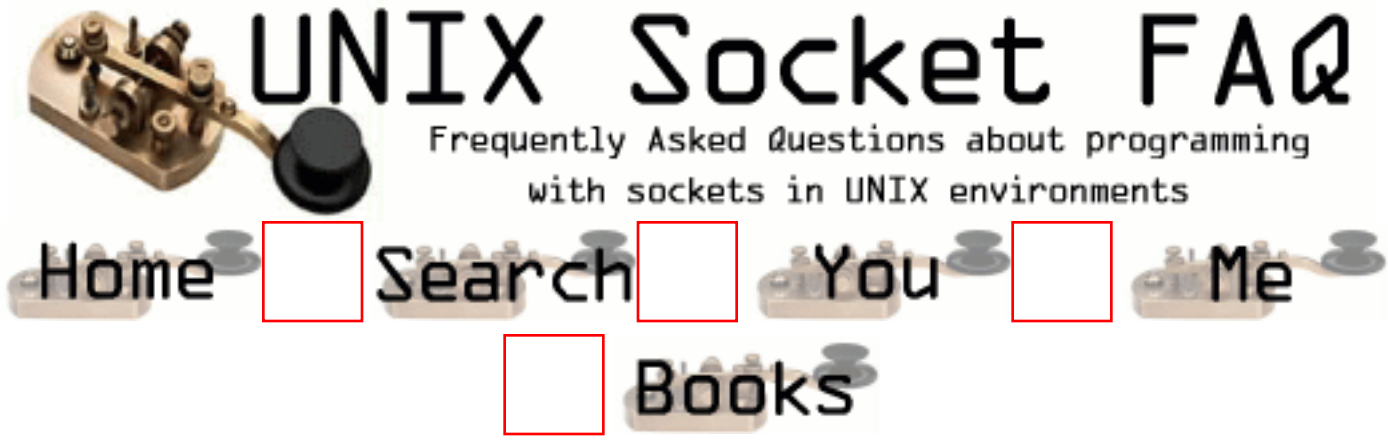
From: Chris Yang

two links unreachable !

From: Flávio

Infelizmente estes links não funcionam....

Add a Comment



General Information and Concepts: Where can I get the sample source code?

The sample source code is no longer included in the faq. To get it, please download it from one of the unix-socket-faq www pages:

> http://www.ibrado.com/sock-faq
> http://kipper.york.ac.uk/~vic/sock-faq
> http://www.ntua.gr/sock-faq

If you don't have web access, you can ftp it with ftpmail by following the following instructions.

To get the sample source by mail, send mail to ftpmail@decwrl.dec.com, with no subject line and a body like this:

```
reply
connect ftp.zymsys.com
binary
uuencode
get pub/sockets/examples.tar.gz
quit
```

Save the reply as examples.uu, and type:

```
% uudecode examples.uu
% gunzip examples.tar.gz
% tar xf examples.tar
```

This will create a directory called socket-faq-examples which contains the sample code from this faq, plus a sample client and server for both tcp and udp.

Note that this package requires the gnu unzip program to be installed on your system. It is very

common, but if you don't have it you can get the source for it from:

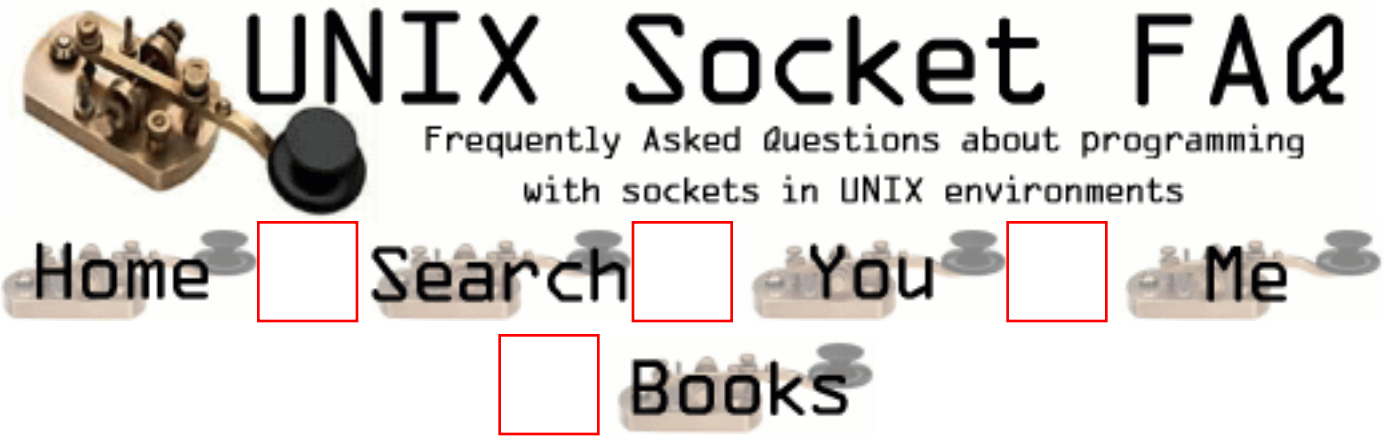
> ftp://prep.ai.mit.edu/pub/gnu/gzip-1.2.4.tar

If you don't have ftp access, you can obtain it in a way similar to obtaining the sample source. I'll leave the exact changes to the body of the message as an excersise for the reader.

From: Vic Metcalfe

Oops, I forgot to add a link to the source in the new site design. The examples are still here. You can get them from http://www.lcg.org/sock-faq/examples.tar.gz, or the corresponding URL for the mirrors. I'll correct the problem soon.

Add a Comment



Questions regarding both Clients and Servers (TCP/SOCK_STREAM): How can I tell when a socket is closed on the other end?

From Andrew Gierth (andrew@erlenstar.demon.co.uk):

AFAIK:

If the peer calls `close()` or exits, without having messed with `SO_LINGER`, then our calls to `read()` should return 0. It is less clear what happens to `write()` calls in this case; I would expect `EPIPE`, not on the next call, but the one after.

If the peer reboots, or sets `l_onoff = 1, l_linger = 0` and then closes, then we should get `ECONNRESET` (eventually) from `read()`, or `EPIPE` from `write()`.

I should also point out that when `write()` returns `EPIPE`, it also raises the `SIGPIPE` signal - you never see the `EPIPE` error unless you handle or ignore the signal.

If the peer remains unreachable, we should get some other error.

I don't think that `write()` can legitimately return 0. `read()` should return 0 on receipt of a FIN from the peer, and on all following calls.

So yes, you **must** expect `read()` to return 0.

As an example, suppose you are receiving a file down a TCP link; you might handle the return from `read()` like this:

```
rc = read(sock,buf,sizeof(buf));
if (rc > 0)
{
    write(file,buf,rc);
    /* error checking on file omitted */
}
```

```
    else if (rc == 0)
    {
        close(file);
        close(sock);
        /* file received successfully */
    }
    else /* rc < 0 */
    {
        /* close file and delete it, since data is not complete
           report error, or whatever */
    }
```

From: Andrew Maholski

This answer does not really tell what to do if you wish to check a socket prior to your read attempt. If the server requires you to maintain an open socket but will time out after a period of inactivity you should use select with a shot timeout to determine the socket status. The following code demonstrates this:

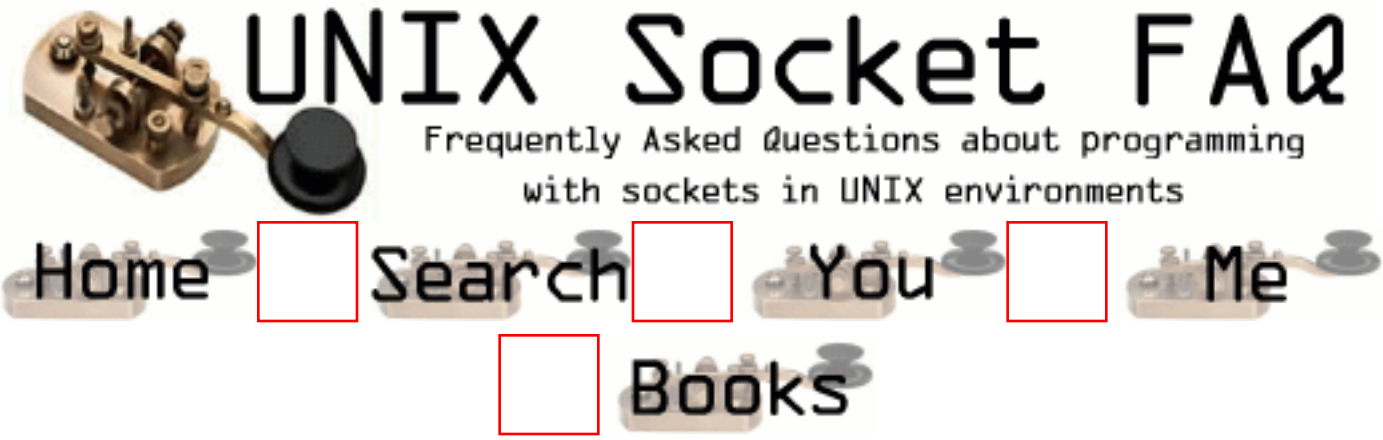
```
int Socket_Status(int sock, int to_secs, int to_msecs)
{

int sval;
int ret_val = 1;
fd_set check_set;
struct timeval to;



FD_ZERO(&check_set);
FD_SET(sock,&check_set);

to.tv_sec = to_secs;
to.tv_usec = to_msecs;

if((sval = select(sock + 1,&check_set,0,0,&to)) < 0)
{
logprintf(LOG_ERRORS,"Select returned %d %d",sval,errno);
ret_val = -1;
}
if(sval == 0)
{
logprintf(LOG_EVERYTHING,"Select timed out.");
ret_val = 0;
}
else if(sval > 0)
```

```
logprintf(LOG_EVERYTHING,"Select shows %d as return val.",sval);

return(ret_val);

}
```

call this routine with:

```
if(Socket_Status(sock,0,1))
{
logprintf(LOG_EVERYTHING,"The socket is not up.");
/*close your end of the thing*/
}
```

Add a Comment



Questions regarding both Clients and Servers (TCP/SOCK_STREAM): What's with the second parameter in bind()?

The man page shows it as "`struct sockaddr *my_addr`". The `sockaddr struct` though is just a place holder for the structure it really wants. You have to pass different structures depending on what kind of socket you have. For an `AF_INET` socket, you need the sockaddr_in structure. It has three fields of interest:

sin_family

Set this to `AF_INET`.

sin_port

The network byte-ordered 16 bit port number

sin_addr

The host's ip number. This is a `struct in_addr`, which contains only one field, `s_addr` which is a `u_long`.

From: Anup Ochani

The socket address structure is protocol specific. sockaddr_un(Unix Protocol) is much larger than sockaddr_in(INTERNET protocol).

connect() and bind() system calls cater for both the protocols. To handle socket address structures of different sizes,
the system calls use a pointer to the structure and the size of that structure.

Add a Comment

Questions regarding both Clients and Servers (TCP/SOCK_STREAM): How do I get the port number for a given service?

Use the `getservbyname()` routine. This will return a pointer to a `servent` structure. You are interested in the `s_port` field, which contains the port number, with correct byte ordering (so you don't need to call `htons()` on it). Here is a sample routine:

```
/* Take a service name, and a service type, and return a port number.  If the
   service name is not found, it tries it as a decimal number.  The number
   returned is byte ordered for the network. */
int atoport(char *service, char *proto) {
  int port;
  long int lport;
  struct servent *serv;
  char *errpos;

  /* First try to read it from /etc/services */
  serv = getservbyname(service, proto);
  if (serv != NULL)
    port = serv->s_port;
  else { /* Not in services, maybe a number? */
    lport = strtol(service,&errpos,0);
    if ( (errpos[0] != 0) || (lport < 1) || (lport > 5000) )
      return -1; /* Invalid port address */
    port = htons(lport);
  }
  return port;
}
```

Add a Comment

Questions regarding both Clients and Servers (TCP/SOCK_STREAM): If bind() fails, what should I do with the socket descriptor?

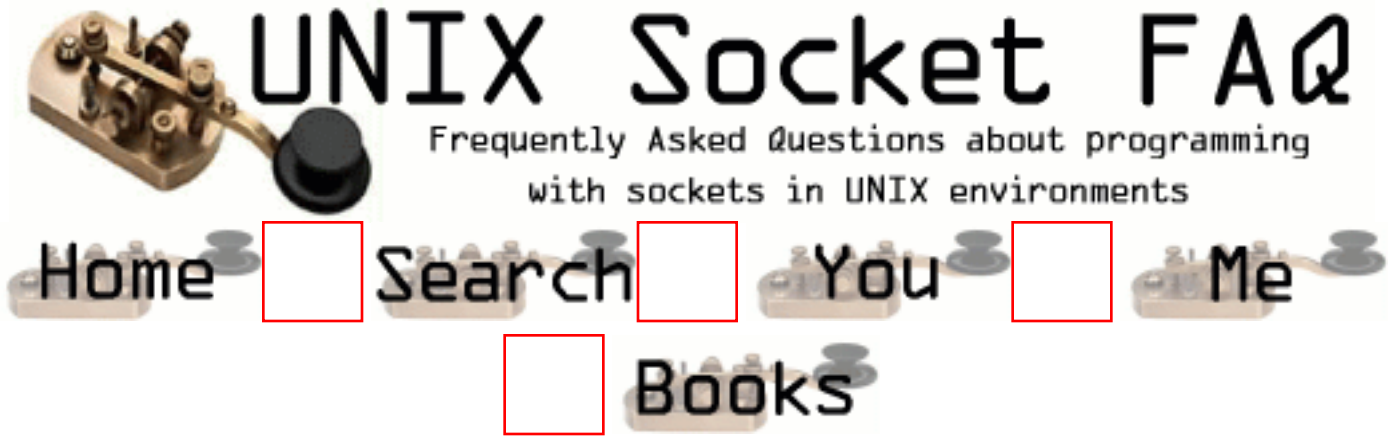
If you are exiting, I have been assured by Andrew that all unixes will close open file descriptors on exit. If you are not exiting though, you can just close it with a regular `close()` call.

From: mike wainwright

The app has exited as has the client. (20000 was the port number)
If the client crashes, maybe close doesn't get called then netstat shows the socket still bound as:

tcp 0 0 appserv1.20000 GBLOND013122.two.2470 FIN_WAIT_2
tcp 0 0 appserv1.20000 GBLOND013122.two.2464 FIN_WAIT_2
tcp 0 0 appserv1.20000 GBLOND013122.two.2459 FIN_WAIT_2

Add a Comment

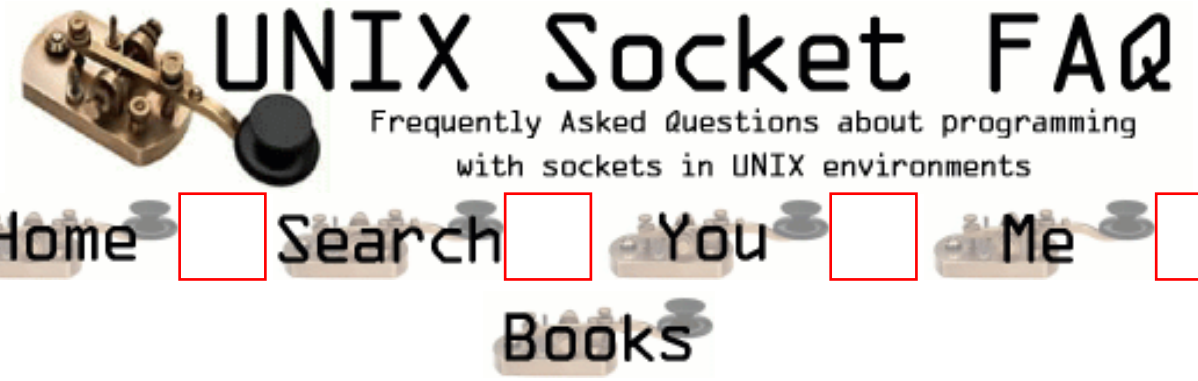


Questions regarding both Clients and Servers (TCP/SOCK_STREAM): How do I properly close a socket?

This question is usually asked by people who try `close()`, because they have seen that that is what they are supposed to do, and then run netstat and see that their socket is still active. Yes, `close()` is the correct method. To read about the TIME_WAIT state, and why it is important, refer to 2.7 Please explain the TIME_WAIT state..

From:

Add a Comment



Questions regarding both Clients and Servers (TCP/SOCK_STREAM): When should I use shutdown()?

From Michael Hunter (mphunter@qnx.com):

`shutdown()` is useful for deliniating when you are done providing a request to a server using TCP. A typical use is to send a request to a server followed by a `shutdown()`. The server will read your request followed by an `EOF` (read of 0 on most unix implementations). This tells the server that it has your full request. You then go read blocked on the socket. The server will process your request and send the necessary data back to you followed by a close. When you have finished reading all of the response to your request you will read an `EOF` thus signifying that you have the whole response. It should be noted the TTCP (TCP for Transactions -- see R. Steven's home page) provides for a better method of tcp transaction management.

S.Degtyarev (deg@sunsr.inp.nsk.su) wrote a nice in-depth message to me about this. He shows a practical example of using shutdown() to aid in synchronization of client processes when one is the "reader" process, and the other is the "writer" process. A portion of his message follows:

Sockets are very similar to pipes in the way they are used for data transfer and client/server transactions, but not like pipes they are bidirectional. Programs that use sockets often `fork()` and each process inherits the socket descriptor. In pipe based programs it is strictly recommended to close all the pipe ends that are not used to convert the pipe line to one-directional data stream to avoid data losses and deadlocks. With the socket there is no way to allow one process only to send data and the other only to receive so you should always keep in mind the consequences.

Generally the difference between `close()` and `shutdown()` is: `close()` closes the socket id for the process but the connection is still opened if another process shares this socket id. The connection stays opened both for read and write, and sometimes this is very important. `shutdown()` breaks the connection for all processes sharing the socket id. Those who try to read will detect `EOF`, and those who try to write will reseive `SIGPIPE`, possibly delayed while the kernel socket buffer will be filled. Additionally, `shutdown()` has a second argument which denotes how to close the connection: 0 means to disable further reading, 1 to disable writing and 2 disables both.

The quick example below is a fragment of a very simple client process. After establishing the connection with the server it forks. Then child sends the keyboard input to the server until `EOF` is received and the parent receives answers from the server.

```
/*
 *      Sample client fragment,
 *      variables declarations and error handling are omitted
 */
        s=connect(...);
```

```
if( fork() ){   /*      The child, it copies its stdin to
                                the socket              */
        while( gets(buffer) >0)
                write(s,buf,strlen(buffer));

        close(s);
        exit(0);
        }

else {          /* The parent, it receives answers  */
        while( (l=read(s,buffer,sizeof(buffer)){
                do_something(l,buffer);

        /* Connection break from the server is assumed  */
        /* ATTENTION: deadlock here                     */
        wait(0); /* Wait for the child to exit          */
        exit(0);
        }
```

What do we expect? The child detects an `EOF` from its `stdin`, it closes the socket (assuming connection break) and exits. The server in its turn detects `EOF`, closes connection and exits. The parent detects `EOF`, makes the `wait()` system call and exits. What do we see instead? The socket instance in the parent process is still opened for writing and reading, though the parent never writes. The server never detects EOF and waits for more data from the client forever. The parent never sees the connection is closed and hangs forever and the server hangs too. Unexpected deadlock! (any deadlock is unexpected though :-)

You should change the client fragment as follows:

```
        if( fork() ) {  /* The child                    */
                while( gets(buffer) }
                        write(s,buffer,strlen(buffer));

                        shutdown(s,1); /* Break the connection
for writing, The server will detect EOF now. Note: reading from
the socket is still allowed. The server may send some more data
after receiving EOF, why not? */
                exit(0);
                }
```

I hope this rough example explains the troubles you can have with client/server syncronization. Generally you should always remember all the instances of the particular socket in all the processes that share the socket and close them all at once if you whish to use close() or use shutdown() in one process to break the connection.

From: brent verner

```
I believe the code examples are incorrectly using the return
value of the fork()s

excerpt from `man 2 fork`

DESCRIPTION
       fork  creates a child process that differs from the parent
```

```
       process only in its PID and PPID, and  in  the  fact  that
       resource  utilizations are set to 0.  File locks and pend-
       ing signals are not inherited.

RETURN VALUE
       On  success,  the  PID of the child process is returned in
       the parent's thread of execution, and a 0 is  returned  in
       the child's thread of execution.  On failure, a -1 will be
       returned in the parent's context, no child process will be
       created, and errno will be set appropriately.
```

From: Ken Whaley

For folks new to shutdown(), it's helpful to remind them that shutdown() is NOT a replacement for close(). You still must close() sockets when you're done with them (after the calls to shutdown()) in order to not leak a file descriptor. Many examples of shutdown() omit the close() after shutdown() because they're test programs where the process exits after the shutdown() (and the process exit of course cleans up all open descriptors). Stevens section on shutdown() doesn't make this clear (his wording implies to me an "either/or" between shutdown() and close()).

From: george elgin

the use of shutdown(1) circumvents what the kernel SHOULD accomplishes
automatically. when the open file descriptor count reaches 0 a FIN
is sent to the partner which begins the sequence of wait states
FIN_WAIT, FIN_WAIT_2 etc. for the reader. When both sides readers
have then completed this orderly shutdown the file descriptor
is returned to the operating system.

i have noticed on some (mostly non-unix) operating systems though a
close by all processes (threads) is not enough to correctly flush data,
(or threads) is not. A shutdown must be done, but on many systems
it is superfulous.

From: Newbie

>I believe the code examples are incorrectly using the >return value of the fork()s

Agree.

Add a Comment

Questions regarding both Clients and Servers (TCP/SOCK_STREAM): Please explain the TIME_WAIT state.

Remember that TCP guarantees all data transmitted will be delivered, if at all possible. When you close a socket, the server goes into a TIME_WAIT state, just to be really really sure that all the data has gone through. When a socket is closed, both sides agree by sending messages to each other that they will send no more data. This, it seemed to me was good enough, and after the handshaking is done, the socket should be closed. The problem is two-fold. First, there is no way to be sure that the last ack was communicated successfully. Second, there may be "wandering duplicates" left on the net that must be dealt with if they are delivered.

Andrew Gierth (andrew@erlenstar.demon.co.uk) helped to explain the closing sequence in the following usenet posting:

Assume that a connection is in ESTABLISHED state, and the client is about to do an orderly release. The client's sequence no. is Sc, and the server's is Ss. The pipe is empty in both directions.

```
Client                                                  Server
======                                                  ======
ESTABLISHED                                             ESTABLISHED
(client closes)
ESTABLISHED                                             ESTABLISHED
             ------->>
FIN_WAIT_1
            <<--------
FIN_WAIT_2                                              CLOSE_WAIT
            <<--------   (server closes)
                                                        LAST_ACK
            , ------->>
TIME_WAIT                                               CLOSED
(2*msl elapses...)
CLOSED
```

Note: the +1 on the sequence numbers is because the FIN counts as one byte of data. (The above diagram is equivalent to fig. 13 from RFC 793).

Now consider what happens if the last of those packets is dropped in the network. The client has done with the connection; it has no more data or control info to send, and never will have. But the server does not know whether the client received all the data correctly; that's what the last ACK segment is for. Now the server may or may not *care* whether the client got the data, but that is not an issue for TCP; TCP is a reliable rotocol, and *must*

distinguish between an orderly connection **close** where all data is transferred, and a connection **abort** where data may or may not have been lost.

So, if that last packet is dropped, the server will retransmit it (it is, after all, an unacknowledged segment) and will expect to see a suitable ACK segment in reply. If the client went straight to CLOSED, the only possible response to that retransmit would be a RST, which would indicate to the server that data had been lost, when in fact it had not been.

(Bear in mind that the server's FIN segment may, additionally, contain data.)

DISCLAIMER: This is my interpretation of the RFCs (I have read all the TCP-related ones I could find), but I have not attempted to examine implementation source code or trace actual connections in order to verify it. I am satisfied that the logic is correct, though.

More commentarty from Vic:

The second issue was addressed by Richard Stevens (rstevens@noao.edu, author of "Unix Network Programming", see 1.6 Where can I get source code for the book [book title]?). I have put together quotes from some of his postings and email which explain this. I have brought together paragraphs from different postings, and have made as few changes as possible.

From Richard Stevens (rstevens@noao.edu):

If the duration of the TIME_WAIT state were just to handle TCP's full-duplex close, then the time would be much smaller, and it would be some function of the current RTO (retransmission timeout), not the MSL (the packet lifetime).

A couple of points about the TIME_WAIT state.

- The end that sends the first FIN goes into the TIME_WAIT state, because that is the end that sends the final ACK. If the other end's FIN is lost, or if the final ACK is lost, having the end that sends the first FIN maintain state about the connection guarantees that it has enough information to retransmit the final ACK.
- Realize that TCP sequence numbers wrap around after 2**32 bytes have been transferred. Assume a connection between A.1500 (host A, port 1500) and B.2000. During the connection one segment is lost and retransmitted. But the segment is not really lost, it is held by some intermediate router and then re-injected into the network. (This is called a "wandering duplicate".) But in the time between the packet being lost & retransmitted, and then reappearing, the connection is closed (without any problems) and then another connection is established between the same host, same port (that is, A.1500 and B.2000; this is called another "incarnation" of the connection). But the sequence numbers chosen for the new incarnation just happen to overlap with the sequence number of the wandering duplicate that is about to reappear. (This is indeed possible, given the way sequence numbers are chosen for TCP connections.) Bingo, you are about to deliver the data from the wandering duplicate (the previous incarnation of the connection) to the new incarnation of the connection. To avoid this, you do not allow the same incarnation of the connection to be reestablished until the TIME_WAIT state terminates. Even the TIME_WAIT state doesn't complete solve the second problem, given what is called TIME_WAIT assassination. RFC 1337 has more details.
- The reason that the duration of the TIME_WAIT state is 2*MSL is that the maximum amount of time a packet can wander around a network is assumed to be MSL seconds. The factor of 2 is for the round-trip. The recommended value for MSL is 120 seconds, but Berkeley-derived implementations normally use 30 seconds instead. This means a TIME_WAIT delay between 1 and 4 minutes. Solaris 2.x does indeed use the recommended MSL of 120 seconds.

A wandering duplicate is a packet that appeared to be lost and was retransmitted. But it wasn't really lost ... some router had problems, held on to the packet for a while (order of seconds, could be a minute if the TTL is large enough) and then re-injects the packet back into the network. But by the time it reappears, the application

that sent it originally has already retransmitted the data contained in that packet.

Because of these potential problems with TIME_WAIT assassinations, one should *not* avoid the TIME_WAIT state by setting the `SO_LINGER` option to send an RST instead of the normal TCP connection termination (FIN/ACK/FIN/ACK). The TIME_WAIT state is there for a reason; it's your friend and it's there to help you :-)

I have a long discussion of just this topic in my just-released "TCP/IP Illustrated, Volume 3". The TIME_WAIT state is indeed, one of the most misunderstood features of TCP.

I'm currently rewriting "Unix Network Programming" (see 1.6 Where can I get source code for the book [book title]?). and will include lots more on this topic, as it is often confusing and misunderstood.

An additional note from Andrew:

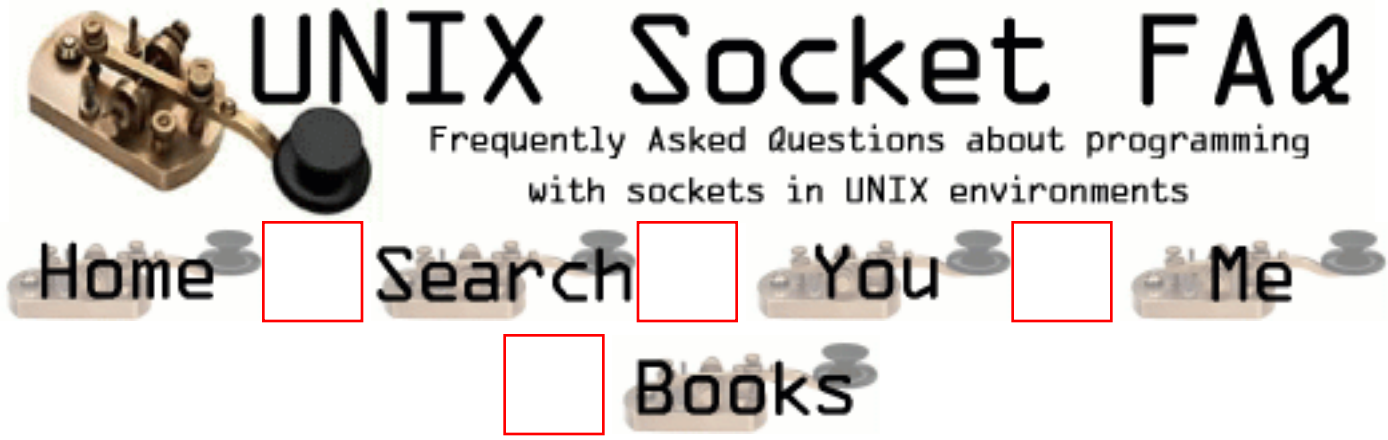
Closing a socket: if `SO_LINGER` has not been called on a socket, then `close()` is not supposed to discard data. This is true on SVR4.2 (and, apparently, on all non-SVR4 systems) but apparently **not** on SVR4; the use of either `shutdown()` or `SO_LINGER` seems to be required to guarantee delivery of all data.

From: Stephen Satchell

On the question of using SO_LINGER to send a RST on close to avoid the TIME_WAIT state: I've been having some problems with router access servers (names withheld to protect the guilty) that have problems dealing with back-to-back connections on a modem dedicated to a specific channel. What they do is let go of the connection, accept another call, attempt to connect to a well-known socket on a host, and the host refuses the connection because there is a connection in TIME_WAIT state involving the well-known socket. (Steve's book TCP Illustrated Vol 1 discusses this problem in more detail.) In order to avoid the connection-refused problem, I've had to install an option to do reset-on-close in the server when the server initiates the disconnection.

My server is a Linux system running 2.0.34 if that level of detail is important to the discussion.

The IP address is usually the same, but the socket number is always different -- I've logged the socket numbers used by the router access servers and they are indeed different. I don't have a log for refused connections, however. (Interested in how to record this information, by the way.)



Questions regarding both Clients and Servers (TCP/SOCK_STREAM): Why does it take so long to detect that the peer died?

From Andrew Gierth (andrew@erlenstar.demon.co.uk):

Because by default, no packets are sent on the TCP connection unless there is data to send or acknowledge.

So, if you are simply waiting for data from the peer, there is no way to tell if the peer has silently gone away, or just isn't ready to send any more data yet. This can be a problem (especially if the peer is a PC, and the user just hits the Big Switch...).

One solution is to use the `SO_KEEPALIVE` option. This option enables periodic probing of the connection to ensure that the peer is still present. **BE WARNED:** the default timeout for this option is **AT LEAST 2 HOURS**. This timeout can often be altered (in a system-dependent fashion) but not normally on a per-connection basis (AFAIK).

RFC1122 specifies that this timeout (if it exists) must be configurable. On the majority of Unix variants, this configuration may only be done globally, affecting all TCP connections which have keepalive enabled. The method of changing the value, moreover, is often difficult and/or poorly documented, and in any case is different for just about every version in existence.

If you must change the value, look for something resembling `tcp_keepidle` in your kernel configuration or network options configuration.

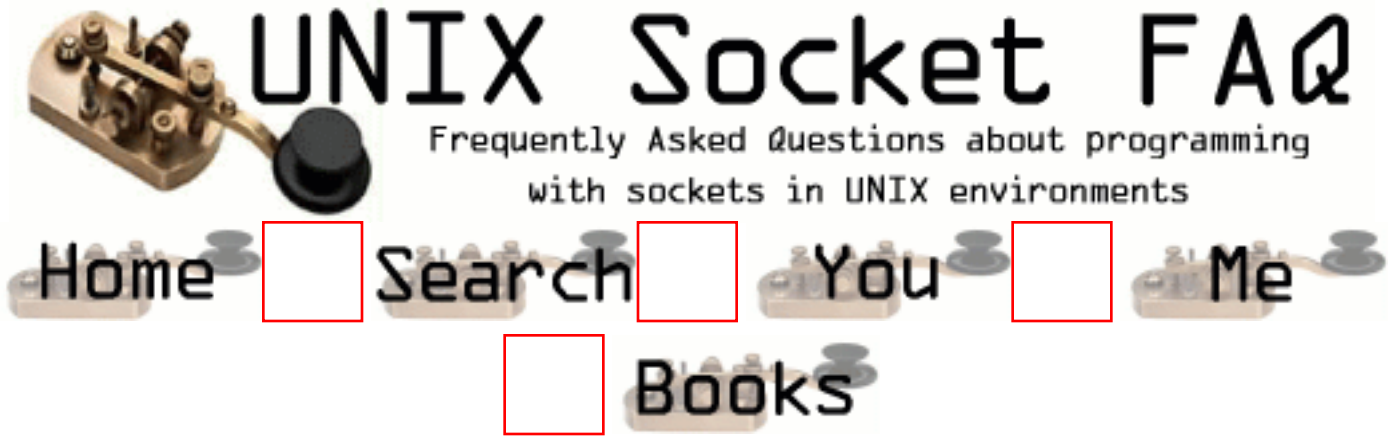
If you're *sending* to the peer, though, you have some better guarantees; since sending data implies receiving ACKs from the peer, then you will know after the retransmit timeout whether the peer is still alive. But the retransmit timeout is designed to allow for various contingencies, with the intention that TCP connections are not dropped simply as a result of minor network upsets. So you should still expect a delay of several minutes before getting notification of the failure.

The approach taken by most application protocols currently in use on the Internet (e.g. FTP,

SMTP etc.) is to implement read timeouts on the server end; the server simply gives up on the client if no requests are received in a given time period (often of the order of 15 minutes). Protocols where the connection is maintained even if idle for long periods have two choices:

1. use `SO_KEEPALIVE`
2. use a higher-level keepalive mechanism (such as sending a null request to the server every so often).

Add a Comment



Questions regarding both Clients and Servers (TCP/SOCK_STREAM): What are the pros/cons of select(), non-blocking I/O and SIGIO?

Using non-blocking I/O means that you have to poll sockets to see if there is data to be read from them. Polling should usually be avoided since it uses more CPU time than other techniques.

Using `SIGIO` allows your application to do what it does and have the operating system tell it (with a signal) that there is data waiting for it on a socket. The only drawback to this soltion is that it can be confusing, and if you are dealing with multiple sockets you will have to do a `select()` anyway to find out which one(s) is ready to be read.

Using `select()` is great if your application has to accept data from more than one socket at a time since it will block until any one of a number of sockets is ready with data. One other advantage to `select()` is that you can set a time-out value after which control will be returned to you whether any of the sockets have data for you or not.

From: c0redump

yeah man, i do the same except when i am sucking cock, i kinda multiplex so i can do erm all at the same time, then when they are ready to cum, i just check their dicks, etc.

From: Ura Moron

Thanks so much for your informative and descriptive posting above.
I belive I have an answer to your problem. Try not to have worry
about the big words. Ask your momma to explain them to you.

mo·ron (môrn, mr-)
n.

1.A person regarded as very stupid.
2.Psychology. A person of mild mental retardation having a mental age of from 7 to

12 years and generally having communication and social skills enabling some degree of academic or vocational education. The term belongs to a classification system no longer in use and is now considered offensive.

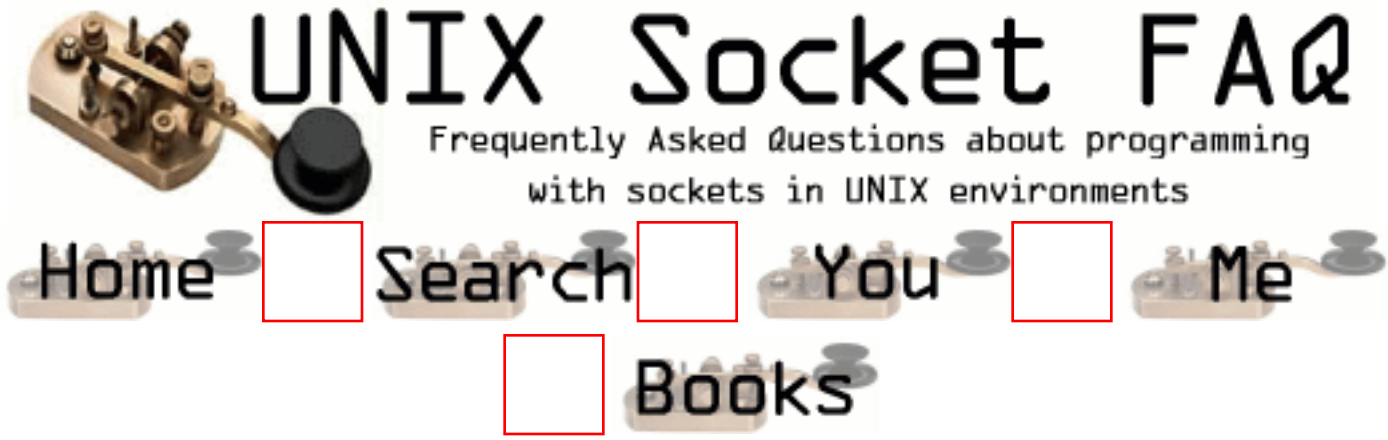
From: ajji footi

Ajji footi

From: sickened

people like c0redump are why we need physician assisted Euthanasia.

Add a Comment



Questions regarding both Clients and Servers (TCP/SOCK_STREAM): Why do I get EPROTO from read()?

From Steve Rago (sar@plc.com):

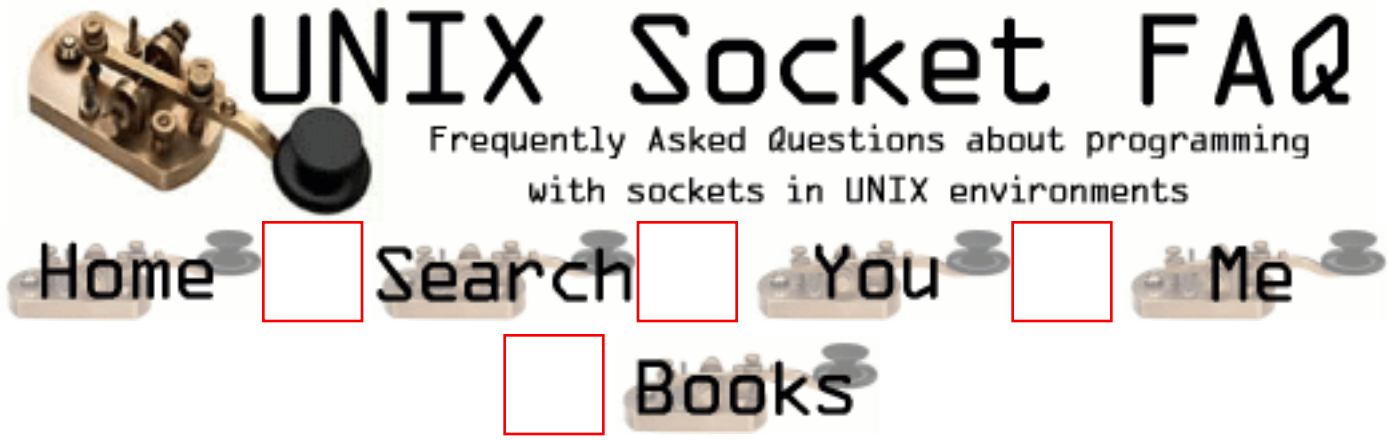
`EPROTO` means that the protocol encountered an unrecoverable error for that endpoint. `EPROTO` is one of those catch-all error codes used by STREAMS-based drivers when a better code isn't available.

And an addition note from Andrew (andrew@erlenstar.demon.co.uk):

Not quite to do with `EPROTO` from `read()`, but I found out once that on some STREAMS-based implementations, `EPROTO` could be returned by `accept()` if the incoming connection was reset before the accept completes.

On some other implementations, accept seemed to be capable of blocking if this occured. This is important, since if `select()` said the listening socket was readable, then you would normally expect *not* to block in the `accept()` call. The fix is, of course, to set nonblocking mode on the listening socket if you are going to use `select()` on it.

Add a Comment



Questions regarding both Clients and Servers (TCP/SOCK_STREAM): How can I force a socket to send the data in its buffer?

From Richard Stevens (rstevens@noao.edu):

You can't force it. Period. TCP makes up its own mind as to when it can send data. Now, *normally* when you call `write()` on a TCP socket, TCP will indeed send a segment, but there's no guarantee and no way to force this. There are *lots* of reasons why TCP will not send a segment: a closed window and the Nagle algorithm are two things to come immediately to mind.

(Snipped suggestion from Andrew Gierth to use `TCP_NODELAY`)

Setting this only disables one of the many tests, the Nagle algorithm. But if the original poster's problem is this, then setting this socket option will help.

A quick glance at tcp_output() shows around 11 tests TCP has to make as to whether to send a segment or not.

Now from Dr. Charles E. Campbell Jr. (cec@gryphon.gsfc.nasa.gov):

As you've surmised, I've never had any problem with disabling Nagle's algorithm. Its basically a buffering method; there's a fixed overhead for all packets, no matter how small. Hence, Nagle's algorithm collects small packets together (no more than .2sec delay) and thereby reduces the amount of overhead bytes being transferred. This approach works well for rcp, for example: the .2 second delay isn't humanly noticeable, and multiple users have their small packets more efficiently transferred. Helps in university settings where most folks using the network are using standard tools such as rcp and ftp, and programs such as telnet may use it, too.

However, Nagle's algorithm is pure havoc for real-time control and not much better for keystroke interactive applications (control-C, anyone?). It has seemed to me that the types of new programs using sockets that people write usually do have problems with small packet delays. One way to bypass Nagle's algorithm selectively is to use "out-of-band" messaging, but

that is limited in its content and has other effects (such as a loss of sequentiality) (by the way, out-of-band is often used for that ctrl-C, too).

More from Vic:

So to sum it all up, if you are having trouble and need to flush the socket, setting the `TCP_NODELAY` option will usually solve the problem. If it doesn't, you will have to use out-of-band messaging, but according to Andrew, "out-of-band data has its own problems, and I don't think it works well as a solution to buffering delays (haven't tried it though). It is *not* 'expedited data' in the sense that exists in some other protocols; it is transmitted in-stream, but with a pointer to indicate where it is."

I asked Andrew something to the effect of "**What promises does TCP make about when it will get around to writing data to the network?**" I thought his reply should be put under this question:

Not many promises, but some.

I'll try and quote chapter and verse on this:

References:

RFC 1122, "Requirements for Internet Hosts" (also STD 3)
RFC 793, "Transmission Control Protocol" (also STD 7)

1. The socket interface does not provide access to the TCP PUSH flag.
2. RFC1122 says (4.2.2.2): A TCP MAY implement PUSH flags on SEND calls. If PUSH flags are not implemented, then the sending TCP: (1) must not buffer data indefinitely, and (2) MUST set the PSH bit in the last buffered segment (i.e., when there is no more queued data to be sent).
3. RFC793 says (2.8): When a receiving TCP sees the PUSH flag, it must not wait for more data from the sending TCP before passing the data to the receiving process. [RFC1122 supports this statement.]
4. Therefore, data passed to a `write()` call must be delivered to the peer within a finite time, unless prevented by protocol considerations.
5. There are (according to a post from Stevens quoted in the FAQ [earlier in this answer - Vic]) about 11 tests made which could delay sending the data. But as I see it, there are only 2 that are significant, since things like retransmit backoff are a) not under the programmers control and b) must either resolve within a finite time or drop the connection.

The first of the interesting cases is "window closed" (ie. there is no buffer space at the receiver; this can delay data indefinitely, but only if the receiving process is not actually reading the data that is available)

Vic asks:

OK, it makes sense that if the client isn't reading, the data isn't going to make it across the

connection. I take it this causes the sender to block after the recieve queue is filled?

The sender blocks when the socket send buffer is full, so buffers will be full at both ends.

While the window is closed, the sending TCP sends window probe packets. This ensures that when the window finally does open again, the sending TCP detects the fact. [RFC1122, ss 4.2.2.17]

The second interesting case is "Nagle algorithm" (small segments, e.g. keystrokes, are delayed to form larger segments if ACKs are expected from the peer; this is what is disabled with `TCP_NODELAY`)

Vic Asks:

Does this mean that my tcpclient sample should set TCP_NODELAY to ensure that the end-of-line code is indeed put out onto the network when sent?

No. tcpclient.c is doing the right thing as it stands; trying to write as much data as possible in as few calls to `write()` as is feasible. Since the amount of data is likely to be small relative to the socket send buffer, then it is likely (since the connection is idle at that point) that the entire request will require only one call to `write()`, and that the TCP layer will immediately dispatch the request as a single segment (with the PSH flag, see point 2.2 above).

The Nagle algorithm only has an effect when a second `write()` call is made while data is still unacknowledged. In the normal case, this data will be left buffered until either: a) there is no unacknowledged data; or b) enough data is available to dispatch a full-sized segment. The delay cannot be indefinite, since condition (a) must become true within the retransmit timeout or the connection dies.

Since this delay has negative consequences for certain applications, generally those where a stream of small requests are being sent without response, e.g. mouse movements, the standards specify that an option must exist to disable it. [RFC1122, ss 4.2.3.4]

Additional note: RFC1122 also says:

[DISCUSSION]:

> When the PUSH flag is not implemented on SEND calls, i.e., when the application/TCP interface uses a pure streaming model, responsibility for aggregating any tiny data fragments to form reasonable sized segments is partially borne by the application layer.

So programs should avoid calls to `write()` with small data lengths (small relative to the MSS, that is); it's better to build up a request in a buffer and then do one call to `sock_write()` or equivalent.

The other possible sources of delay in the TCP are not really controllable by the program, but they can only delay the data temporarily.

Vic asks:

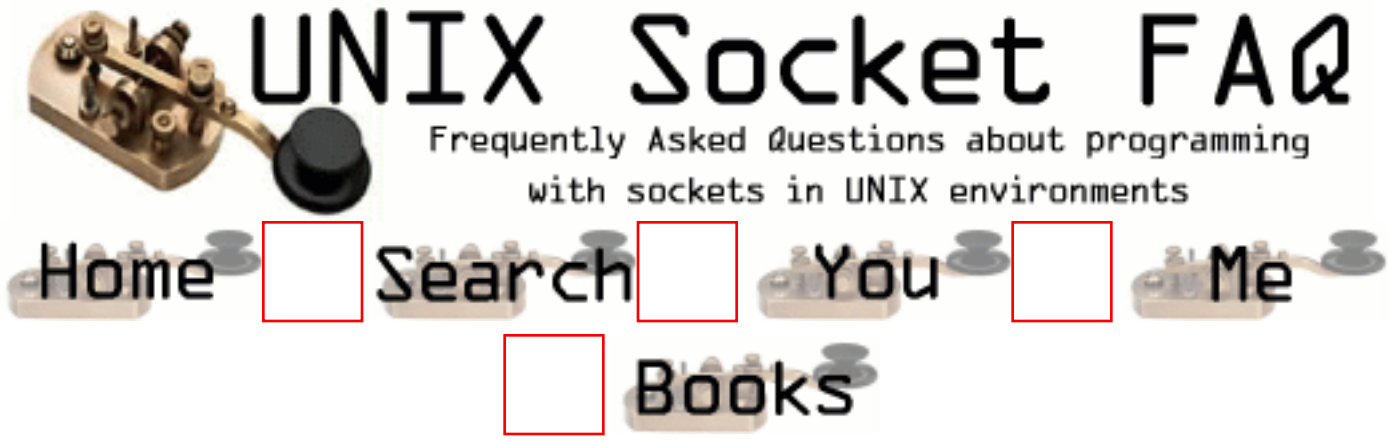
By temporarily, you mean that the data will go as soon as it can, and I won't get stuck in a

position where one side is waiting on a response, and the other side hasn't recieved the request? (Or at least I won't get stuck forever)

You can only deadlock if you somehow manage to fill up all the buffers in both directions... not easy.

If it is possible to do this, (can't think of a good example though), the solution is to use nonblocking mode, especially for writes. Then you can buffer excess data in the program as necessary.

Add a Comment



Questions regarding both Clients and Servers (TCP/SOCK_STREAM): Where can I get a library for programming sockets?

There is the Simple Sockets Library by Charles E. Campbell, Jr. PhD. and Terry McRoberts. The file is called ssl.tar.gz, and you can download it from this faq's home page. For c++ there is the Socket++ library which is on ftp://ftp.virginia.edu/pub/socket++-1.11.tar.gz. There is also C++ Wrappers. The file is called ftp://ftp.huji.ac.il/pub/languages/C++/C++_wrappers.tar.gz. Thanks to Bill McKinnon for tracking it down for me! From http://www.cs.wustl.edu/~schmidt you should be able to find the ACE toolkit. Another C++ library called libtcp++ is also available at http://www.sashanet.com/internet/download.html. PING Software Group has some libraries that include a sockets interface among other things. It seems to be all Java stuff now. You can find their stuff at http://www.nerosworld.com/ping/. Thanks to Reid Judd for hunting that down for us!

Philippe Jounin has developed a cross platform library which includes high level support for http and ftp protocols, with more to come. You can find it at http://perso.magic.fr/jounin-ph/P_tcp4u.htm, and you can find a review of it at http://www6.zdnet.com/cgi-bin/texis/swlib/hotfiles/info.html?fcode=000H4F

I don't have any experience with any of these libraries, so I can't recomend one over the other.

From: Zhao Cheng

Hi,

In your FAQ Item -- 2.12 "Where can I get a library for programming sockets?". Links to ssl.tar.gz and ftp://ftp.virginia.edu/pub/socket++-1.11.tar.gz are broken.

By the way, your web site is really very helpful for me.

thanks a lot for all your efforts.

Zhao

From: THoK

Metacrawler search: "Simple Sockets Library"

Result: http://wauug.erols.com/pub/sunsite/libs/ssl.tar.gz

From:

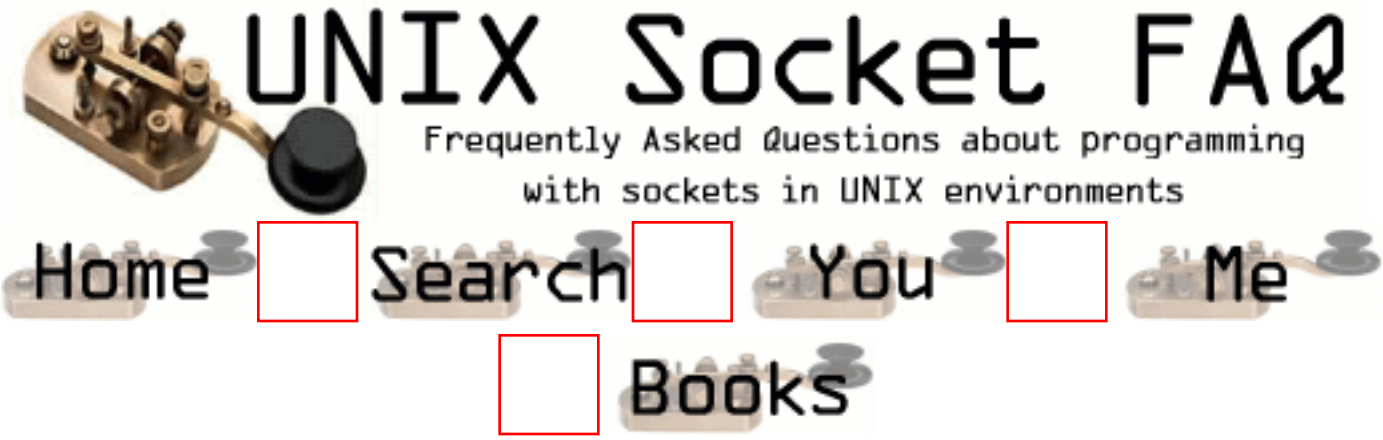
The Socket++ library has a website:
http://www.cs.utexas.edu/users/lavender/courses/socket++/index.html

---

From: Lauri Nurmi

I have modified Socket++ version 1.11 to make it compile with recent GCC versions under Linux (i386).
The modified version is available as tar.gz and RPM here. You will also find the original, unmodified socket++-1.11.tar.gz there.

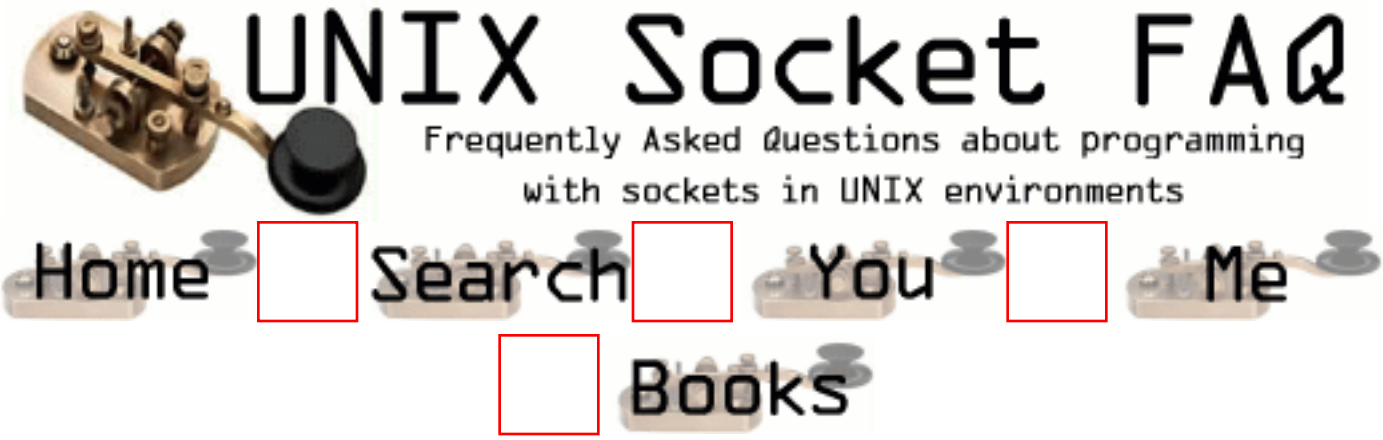
Add a Comment



Questions regarding both Clients and Servers (TCP/SOCK_STREAM): How come select says there is data, but read returns zero?

The data that causes select to return is the EOF because the other side has closed the connection. This causes read to return zero. For more information see 2.1 How can I tell when a socket is closed on the other end?

Add a Comment



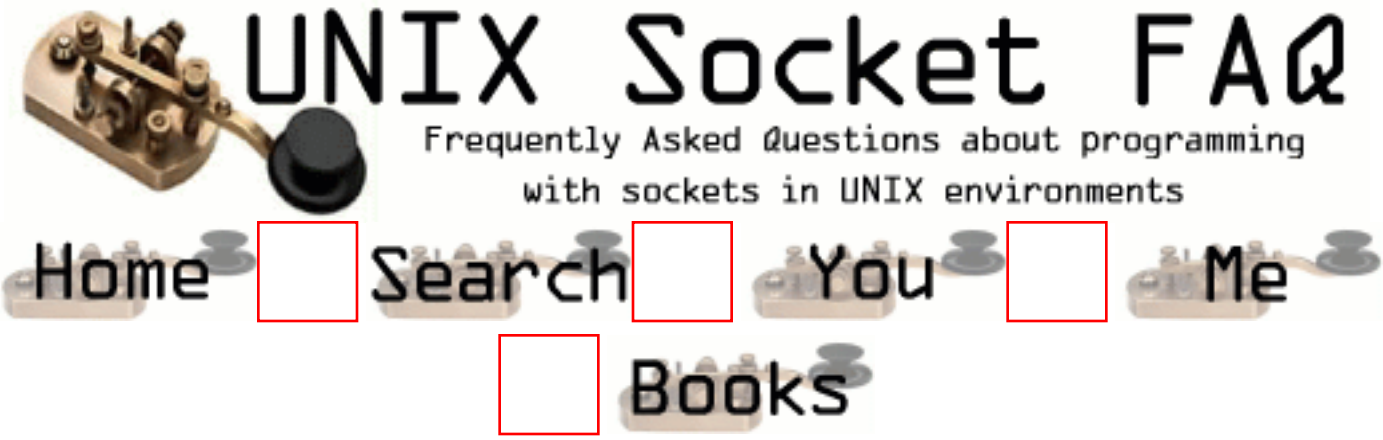
Questions regarding both Clients and Servers (TCP/SOCK_STREAM): Whats the difference between select() and poll()?

From Richard Stevens (rstevens@noao.edu):

The basic difference is that `select()`'s `fd_set` is a bit mask and therefore has some fixed size. It would be possible for the kernel to not limit this size when the kernel is compiled, allowing the application to define `FD_SETSIZE` to whatever it wants (as the comments in the system header imply today) but it takes more work. 4.4BSD's kernel and the Solaris library function both have this limit. But I see that BSD/OS 2.1 has now been coded to avoid this limit, so it's doable, just a small matter of programming. :-) Someone should file a Solaris bug report on this, and see if it ever gets fixed.

With `poll()`, however, the user must allocate an array of `pollfd` structures, and pass the number of entries in this array, so there's no fundamental limit. As Casper notes, fewer systems have `poll()` than `select`, so the latter is more portable. Also, with original implementations (SVR3) you could not set the descriptor to -1 to tell the kernel to ignore an entry in the `pollfd` structure, which made it hard to remove entries from the array; SVR4 gets around this. Personally, I always use `select()` and rarely `poll()`, because I port my code to BSD environments too. Someone could write an implementation of `poll()` that uses `select()`, for these environments, but I've never seen one. Both `select()` and `poll()` are being standardized by POSIX 1003.1g.

Add a Comment



Questions regarding both Clients and Servers (TCP/SOCK_STREAM): How do I send [this] over a socket

Anything other than single bytes of data will probably get mangled unless you take care. For integer values you can use `htons()` and friends, and strings are really just a bunch of single bytes, so those should be OK. Be careful not to send a pointer to a string though, since the pointer will be meaningless on another machine. If you need to send a struct, you should write `sendthisstruct()` and `readthisstruct()` functions for it that do all the work of taking the structure apart on one side, and putting it back together on the other. If you need to send floats, you may have a lot of work ahead of you. You should read RFC 1014 which is about portable ways of getting data from one machine to another (thanks to Andrew Gabriel for pointing this out).

From: Gregory

Where can i find sendthisstruct and readthisstruct. What library are they in and are there definitions of what arguments they take and return.These functions seem like they could solve alot of my problems Thanks

---

From: Rob Seace

No, I think the point was that YOU are supposed to write those functions yourself... Because, only you will know the contents of your structures, and how to properly assemble and disassemble them...

However, let me just add a semi-heretical suggestion: if you know for a fact that your client and server programs are going to be running on the same platform, it's really ok to simply send your structs raw, doing absolutely no magic transformations to them, at all... I do it all the time... Simply pass a pointer to your struct to

recv()/send(), read()/write(), whatever... (Or, memcpy() to/from a char buffer...) The only real restriction is that your structs must NOT contain any pointers, at all... Of course, those simply can't be sent... (Or, rather, it would be pointless to send them, as they are meaningless on the remote machine...) But, pretty much anything else is fair game... But, again, let me repeat: this will ONLY work if both ends are on the same platform... If you intend your program to be used on a multiple platforms, avoid this method at all costs... But, if you know it's only going to be used on a specific fixed platform, it makes for a nice simple method of transfering your structs...

Add a Comment

Questions regarding both Clients and Servers (TCP/SOCK_STREAM): How do I use TCP_NODELAY?

First off, be sure you really want to use it in the first place. It will disable the Nagle algorithm (see 2.11 How can I force a socket to send the data in its buffer?), which will cause network traffic to increase, with smaller than needed packets wasting bandwidth. Also, from what I have been able to tell, the speed increase is very small, so you should probably do it without `TCP_NODELAY` first, and only turn it on if there is a problem.

Here is a code example, with a warning about using it from Andrew Gierth:

```
int flag = 1;
int result = setsockopt(sock,            /* socket affected */
                        IPPROTO_TCP,     /* set option at TCP level */
                        TCP_NODELAY,     /* name of option */
                        (char *) &flag,  /* the cast is historical
                                            cruft */
                        sizeof(int));    /* length of option value */
if (result < 0)
   ... handle the error ...
```

`TCP_NODELAY` is for a *specific* purpose; to disable the Nagle buffering algorithm. It should only be set for applications that send frequent small bursts of information without getting an immediate response, where timely delivery of data is required (the canonical example is mouse movements).

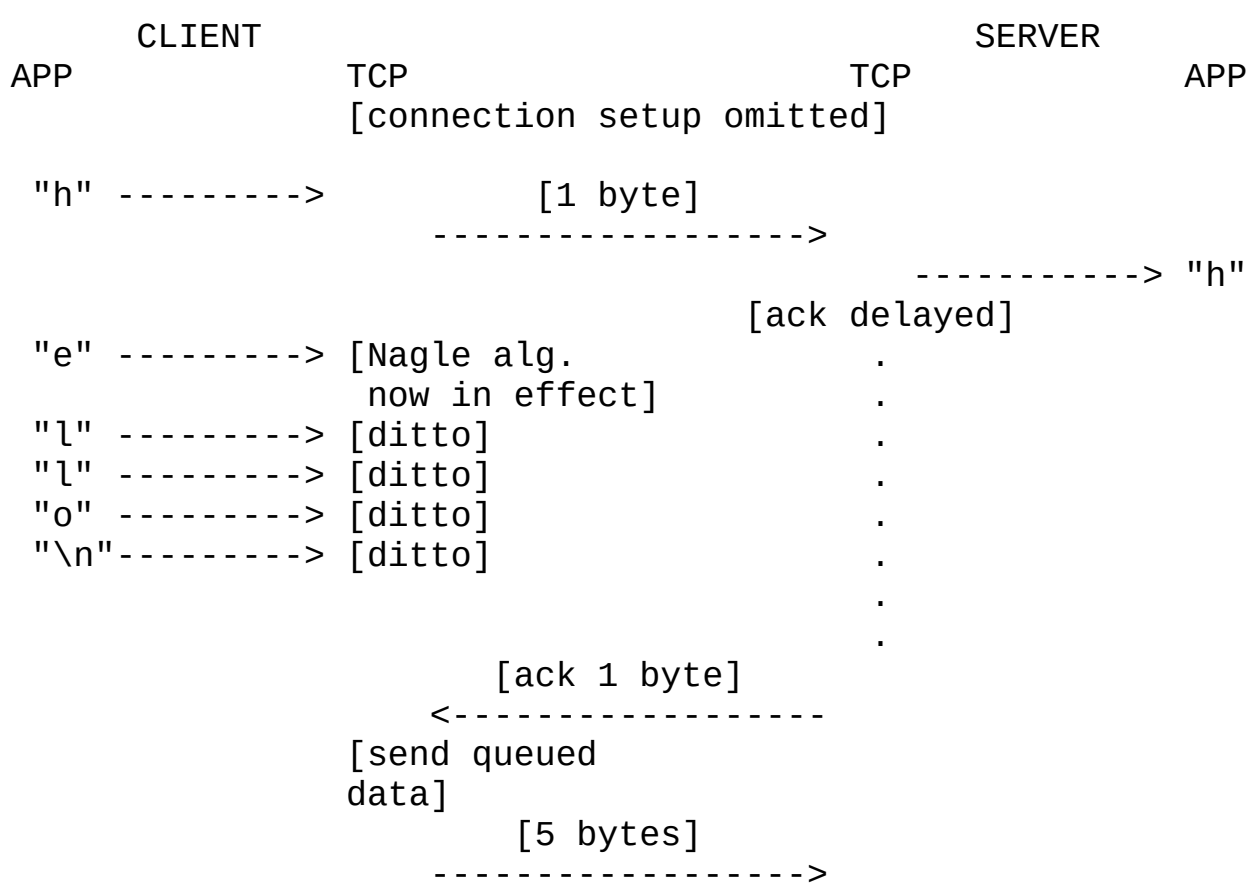
Add a Comment

Questions regarding both Clients and Servers (TCP/SOCK_STREAM): What exactly does the Nagle algorithm do?

It groups together as much data as it can between ACK's from the other end of the connection. I found this really confusing until Andrew Gierth (andrew@erlenstar.demon.co.uk) drew the following diagram, and explained:

This diagram is not intended to be complete, just to illustrate the point better...

Case 1: client writes 1 byte per `write()` call. The program on host B is tcpserver.c from the FAQ examples.



```
                                                  ------------> "ello\n"
                                                  <------------ "HELLO\n"
                     [6 bytes, ack 5 bytes]
                      <------------------
     "HELLO\n" <----
                 [ack delayed]
                    .
                    .
                    .   [ack 6 bytes]
                       ------------------>
```

Total segments: 5. (If `TCP_NODELAY` was set, could have been up to 10.) Time for response: 2*RTT, plus ack delay.

Case 2: client writes all data with one `write()` call.

```
          CLIENT                                      SERVER
    APP             TCP                       TCP             APP
                    [connection setup omitted]

     "hello\n" --->          [6 bytes]
                       ------------------>
                                                  ------------> "hello\n"
                                                  <------------ "HELLO\n"
                     [6 bytes, ack 6 bytes]
                      <------------------
     "HELLO\n" <----
                [ack delayed]
                    .
                    .
                    .   [ack 6 bytes]
                       ------------------>
```

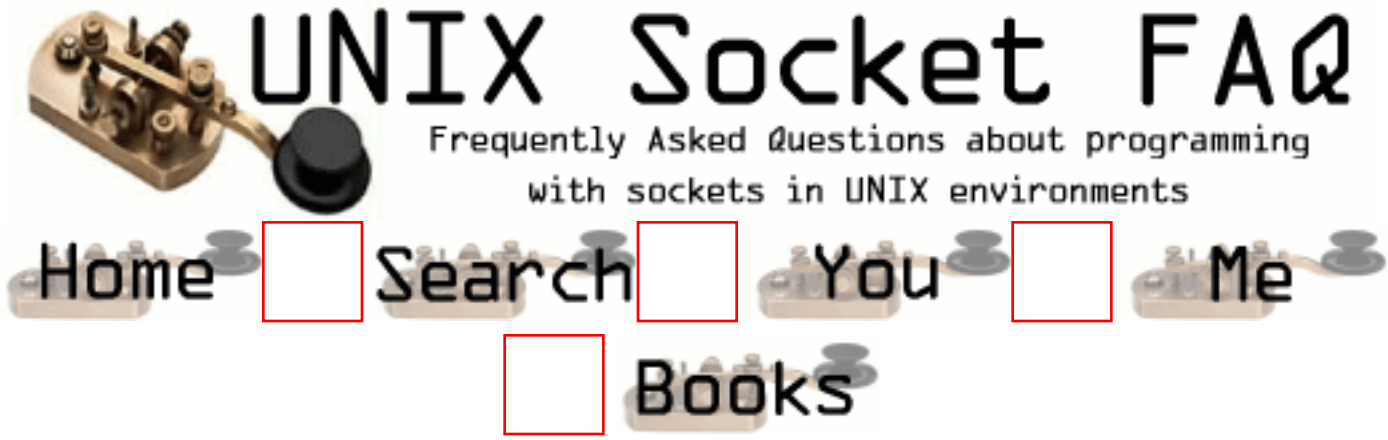
Total segments: 3.

Time for response = RTT (therefore minimum possible).

Hope this makes things a bit clearer...

Note that in case 2, you *don't* want the implementation to gratuitously delay sending the data, since that would add straight onto the response time.

Add a Comment



Questions regarding both Clients and Servers (TCP/SOCK_STREAM): What is the difference between read() and recv()?

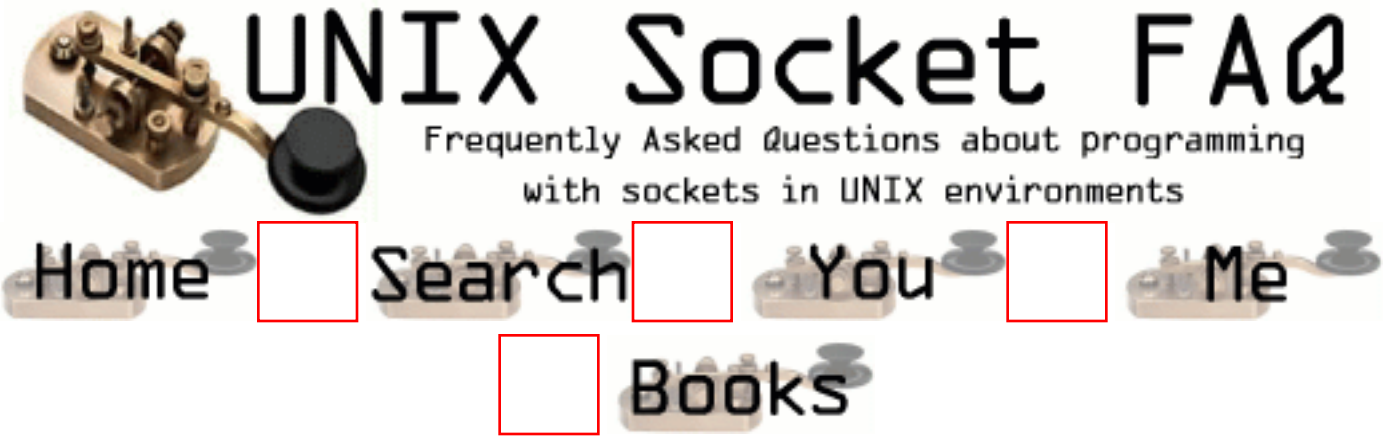
From Andrew Gierth (andrew@erlenstar.demon.co.uk):

`read()` is equivalent to `recv()` with a `flags` parameter of 0. Other values for the `flags` parameter change the behaviour of `recv()`. Similarly, `write()` is equivalent to `send()` with `flags` == 0.

It is unlikely that send()/recv() would be dropped; perhaps someone with a copy of the POSIX drafts for socket calls can check...

Portability note: non-unix systems may not allow `read()/write()` on sockets, but `recv()/send()` are usually ok. This is true on Windows and OS/2, for example.

Add a Comment



Questions regarding both Clients and Servers (TCP/SOCK_STREAM): I see that send()/write() can generate SIGPIPE. Is there any advantage to handling the signal, rather than just ignoring it and checking for the EPIPE error?

From Andrew Gierth (andrew@erlenstar.demon.co.uk):

In general, the only parameter passed to a signal handler is the signal number that caused it to be invoked. Some systems have optional additional parameters, but they are no use to you in this case.

My advice is to just ignore `SIGPIPE` as you suggest. That's what I do in just about all of my socket code; errno values are easier to handle than signals (in fact, the first revision of the FAQ failed to mention `SIGPIPE` in that context; I'd got so used to ignoring it...)

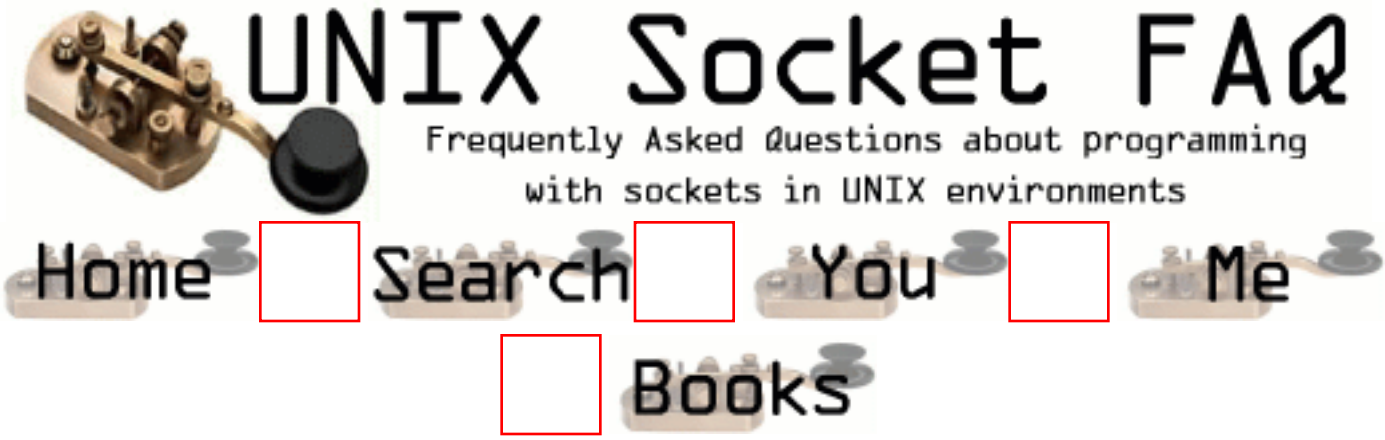
There is one situation where you should *not* ignore `SIGPIPE`; if you are going to `exec()` another program with stdout redirected to a socket. In this case it is probably wise to set `SIGPIPE` to `SIG_DFL` before doing the `exec()`.

Jesse Norell has pointed out that if you are using SO_KEEPALIVE to test the connection, and you aren't doing reads or writes very frequently, you might want to leave SIGPIPE enabled so that your server process gets signalled when the system determines your link is dead. Normally though you will just check returns from `read()/write()` and act appropriately.

From: Corey

And once you receive an EPIPE error on the client are you forced to close that connection and reestablish a new connection? Even though the server thinks there is still a connection?

Add a Comment



Questions regarding both Clients and Servers (TCP/SOCK_STREAM): After the chroot(), calls to socket() are failing. Why?

From Andrew Gierth (andrew@erlenstar.demon.co.uk):

On systems where sockets are implemented on top of Streams (e.g. all SysV-based systems, presumably including Solaris), the `socket()` function will actually be opening certain special files in /dev. You will need to create a /dev directory under your fake root and populate it with the required device nodes (only).

Your system documentation may or may not specify exactly which device nodes are required; I can't help you there (sorry). (Editors note: Adrian Hall (adrian@hottub.org) suggested checking the man page for ftpd, which should list the files you need to copy and devices you need to create in the chroot'd environment.)

A less-obvious issue with `chroot()` is if you call `syslog()`, as many daemons do; `syslog()` opens (depending on the system) either a UDP socket, a FIFO or a Unix-domain socket. So if you use it after a `chroot()` call, make sure that you call `openlog()` *before* the chroot.

From: Stanislav Shalunov

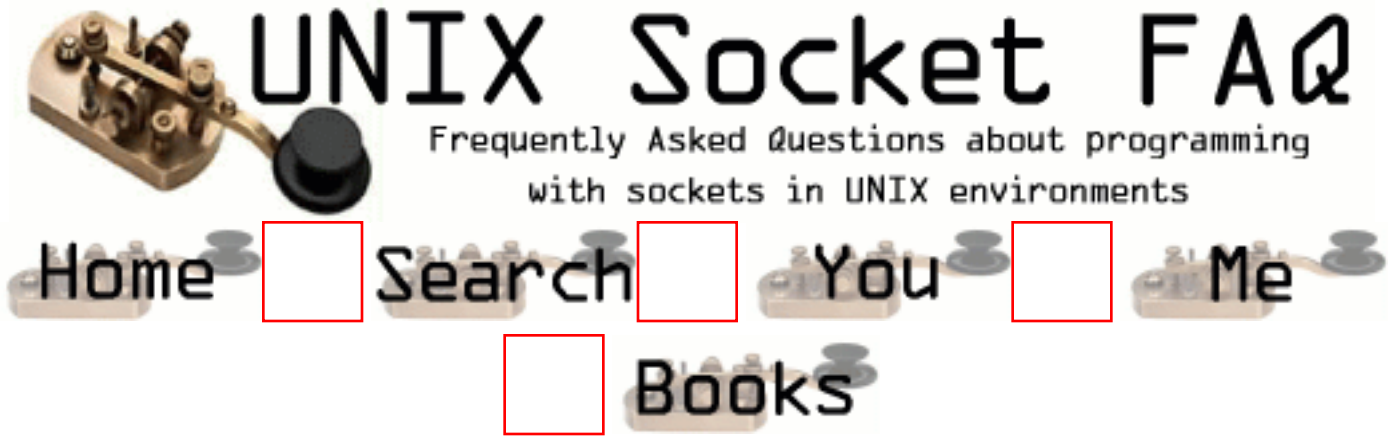
On Solaris you need /dev/tcp (and/or /dev/udp) in order for programs linked with -lsocket -lnsl to work. The libraries will also use /etc/netconfig which may or may not need to be copied to chroot jail depending on versions. (In Solaris, /dev/tcp will be a symlink to ../devices/pseudo/...; you need to notice the major and minor numbers of the file /dev/tcp points to and recreate that. The partition must not be mounted nosuid because in Solaris nosuid implies nodev. Ouch, or use BSD.)

From: Stanislav Shalunov

In most version of libc, `openlog()` itself will not be sufficient to cause the library to actually

open the connection. You would have to actually `syslog()` something. Alternatively, a system-dependent FIFO (like /dev/log might be moved to the chroot jail and a symlink be put in the original location. This will obviously work only if you have a single chroot'ed daemon. Wietse Venema's Postfix MTA has scripts that set up chroot jail for various Unices.)

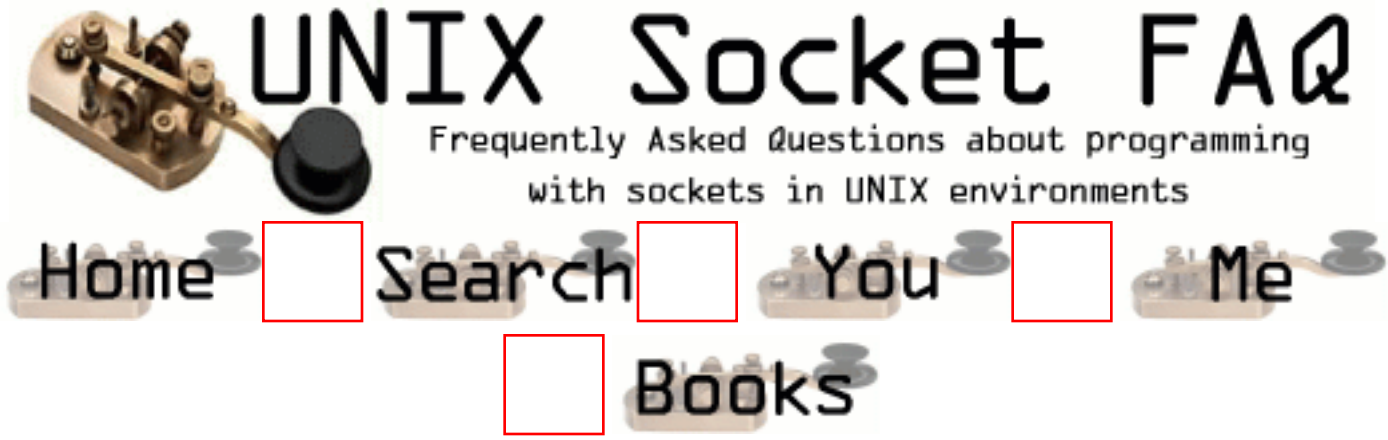
Add a Comment



Questions regarding both Clients and Servers (TCP/SOCK_STREAM): Why do I keep getting EINTR from the socket calls?

This isn't really so much an error as an exit condition. It means that the call was interrupted by a signal. Any call that might block should be wrapped in a loop that checkes for `EINTR`, as is done in the example code (See 1.8. Sample Source Code).

Add a Comment



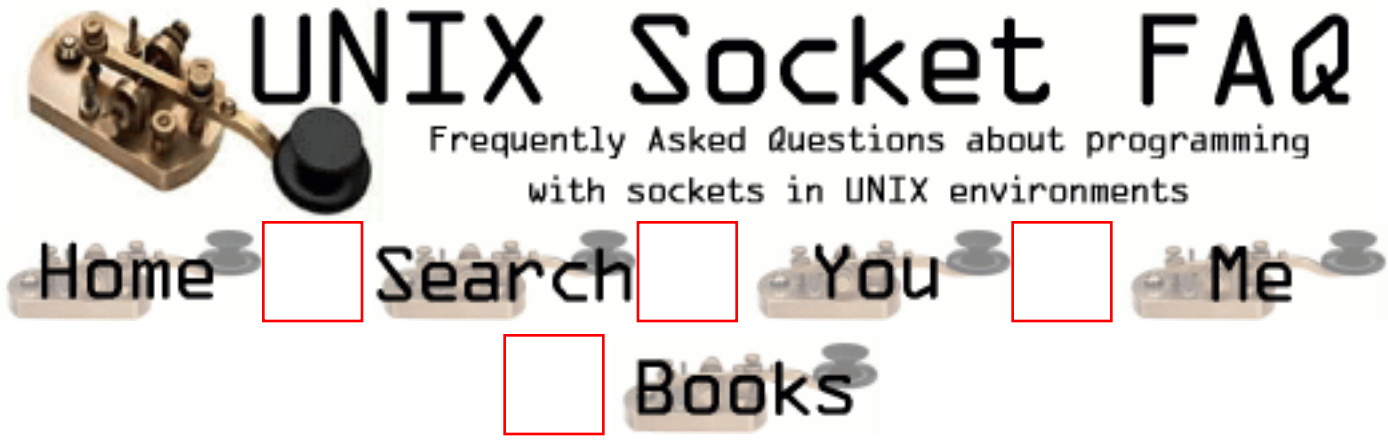
Questions regarding both Clients and Servers (TCP/SOCK_STREAM): When will my application receive SIGPIPE?

From Richard Stevens (rstevens@noao.edu):

Very simple: with TCP you get `SIGPIPE` if your end of the connection has received an RST from the other end. What this also means is that if you were using `select` instead of write, the select would have indicated the socket as being readable, since the RST is there for you to read (read will return an error with `errno` set to `ECONNRESET`).

Basically an RST is TCP's response to some packet that it doesn't expect and has no other way of dealing with. A common case is when the peer closes the connection (sending you a FIN) but you ignore it because you're writing and not reading. (You should be using `select`.) So you write to a connection that has been closed by the other end and the other end's TCP responds with an RST.

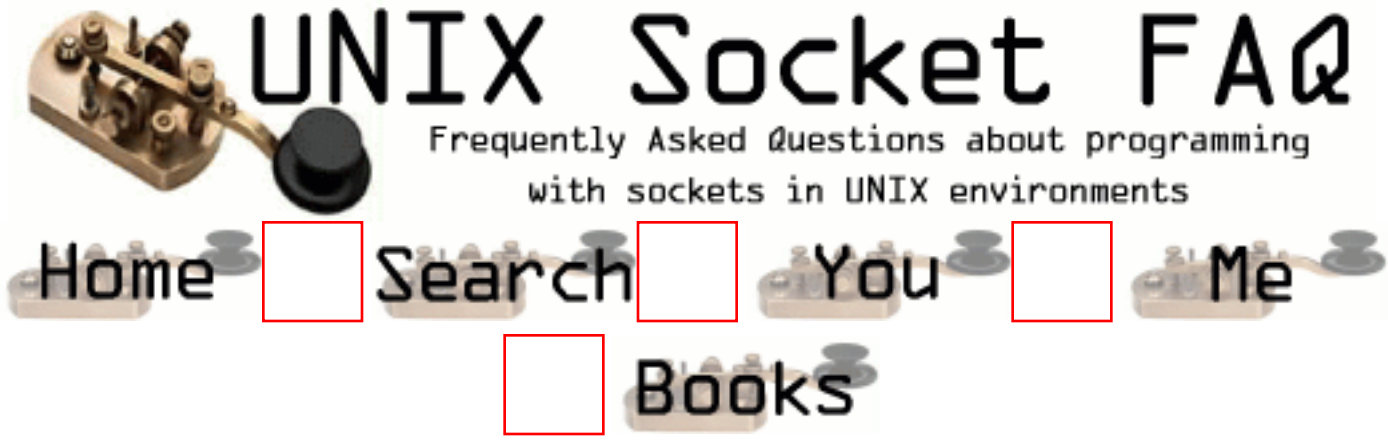
Add a Comment



Questions regarding both Clients and Servers (TCP/SOCK_STREAM): What are socket exceptions? What is out-of-band data?

Unlike exceptions in C++, socket exceptions do not indicate that an error has occured. Socket exceptions usually refer to the notification that out-of-band data has arrived. Out-of-band data (called "urgent data" in TCP) looks to the application like a separate stream of data from the main data stream. This can be useful for separating two different kinds of data. Note that just because it is called "urgent data" does not mean that it will be delivered any faster, or with higher priorety than data in the in-band data stream. Also beware that unlike the main data stream, the out-of-bound data may be lost if your application can't keep up with it.

Add a Comment



Questions regarding both Clients and Servers (TCP/SOCK_STREAM): How can I find the full hostname (FQDN) of the system I'm running on?

From Richard Stevens (rstevens@noao.edu):

Some systems set the hostname to the FQDN and others set it to just the unqualified host name. I know the current BIND FAQ recommends the FQDN, but most Solaris systems, for example, tend to use only the unqualified host name.

Regardless, the way around this is to first get the host's name (perhaps an FQDN, perhaps unaualified). Most systems support the Posix way to do this using `uname()`, but older BSD systems only provide `gethostname()`. Call `gethostbyname()` to find your IP address. Then take the IP address and call `gethostbyaddr()`. The `h_name` member of the `hostent{}` should then be your FQDN.

From: Stanislav Shalunov

Since the DNS can be very slow in some setups it might be preferable to try to use `uname()` if it is supported. If the program is supposed to be portable then GNU autoconf would be the way to go. (The results of not doing this can be quite annoying: e.g., GNU Emacs does a reverse lookup to find FQDN on startup, which means that if the nameserver is not local and the system has no network connection at the moment you'd have to wait a long time to see the editor started.)

From: Erik Landry

Instead of using uname(2)/gethostname(2), gethostbyname(3), and gethostbyaddr(3) which does two DNS lookups, I suggest using socket(2) (with AF_INET), bind(2) (with INADDR_ANY) (or even just socketpair(2)), getsockname(2), and then the gethostbyaddr(3) which does only one DNS lookup.

Of course, using this method is bound to have several different implications than the DNS/reverse DNS method (e.g., even though DNS is usually implemented via socket calls there may be machines where DNS calls are available but the socket interfaces are not).

From: Jan Echternach

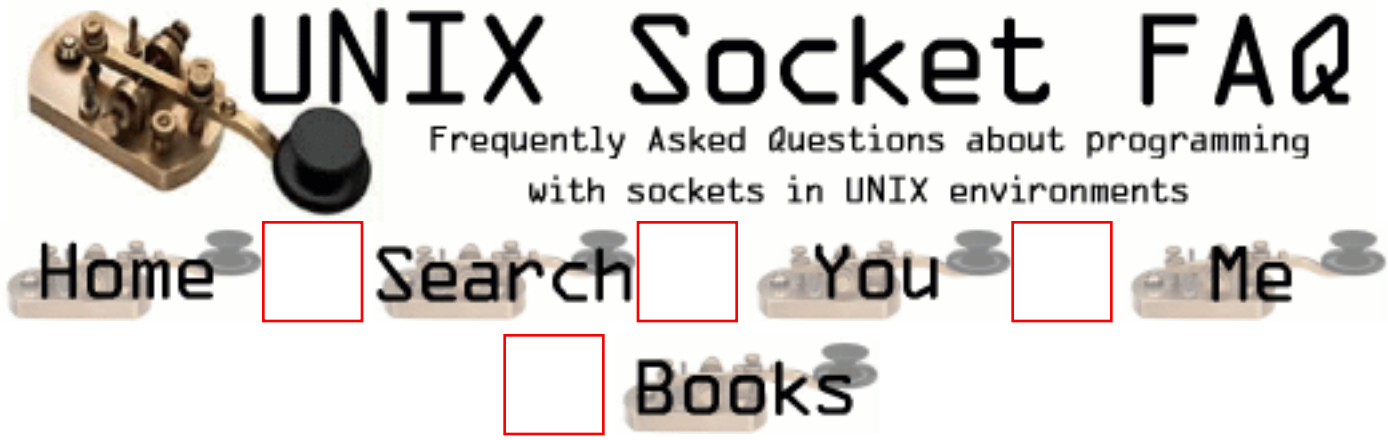
Some systems don't have a FQDN. And even if a system has one, gethostbyname()/gethostbyaddr() might return the unqualified host name instead. One one system, gethostname() + gethostbyname() may be enough to get the FQDN, on a different system this can return the unqualified host name, and gethostname() + gethostbyname() + gehostbyaddr() can even return "localhost".

I suggest you check that the name returned by gethostbyname() or gethostbyaddr() actually looks like a FQDN, i.e. contains a dot, and ask the user to supply the FQDN if it didn't.

From: Mourad

how about a simple `hostname -f` as an execv or system parameter

Add a Comment



Questions regarding both Clients and Servers (TCP/SOCK_STREAM): How do I monitor the activity of sockets?

From: Matthias Rabast (matthias.rabast@ubs.com)

How can I find out,

- which sockets have highest throughput ?
- how big is the tcp window size for each socket ?
- how often does a special socket block and go again ?

For monitoring throughput there are tools such as IPAudit that will monitor throughput. I can't remember which tool I used to use for this purpose, but a quick search found IPAudit. I haven't tried it, so let me know if it works, or if you know some better tools.

You can use `netstat -a` under solaris and look at the Swind and Rwind columns for send and recieve window sizes.

I'm not aware of any tools for monitoring how often a socket blocks. Someone please add a comment if you have any suggestions for this.

You could parse the output of snoop/tcpdump to get some of this information. Let me know if you know a good parser and I'll list it here.

Add a Comment

Writing Client Applications (TCP/SOCK_STREAM): How do I convert a string into an internet address?

If you are reading a host's address from the command line, you may not know if you have an aaa.bbb.ccc.ddd style address, or a host.domain.com style address. What I do with these, is first try to use it as a aaa.bbb.ccc.ddd type address, and if that fails, then do a name lookup on it. Here is an example:

```
/* Converts ascii text to in_addr struct.  NULL is returned if the
   address can not be found. */
struct in_addr *atoaddr(char *address) {
  struct hostent *host;
  static struct in_addr saddr;

  /* First try it as aaa.bbb.ccc.ddd. */
  saddr.s_addr = inet_addr(address);
  if (saddr.s_addr != -1) {
    return &saddr;
  }
  host = gethostbyname(address);
  if (host != NULL) {
    return (struct in_addr *) *host->h_addr_list;
  }
  return NULL;
}
```

From: Rob Seace

Actually, I wouldn't recommend inet_addr() or the gethostby*() functions anymore... The latter aren't thread-safe, and both will only work with IPv4... (Plus, inet_addr() has an ambiguous error return, since -1 is a valid IP: 255.255.255.255...) I would recommend inet_pton() for IP translation (or inet_aton(), if you ONLY care about IPv4; but, why design only for IPv4 in this day and age??), and getaddrinfo() for hostname translation (in fact, you can just use getaddrinfo(), and it'll handle IPs, too)...

From: Screwtape

I don't know what you're programming with, but here on
Linux 2.2, neither inet_pton() nor getaddrinfo() exist.

From: JR

getaddrinfo also doesn't exist on SunOS 5.5.1 or winsock 2 (or any BSD variant I think).

IMO it is better to use only BSD sockets. Or at least specify your platform.

From: Rob Seace

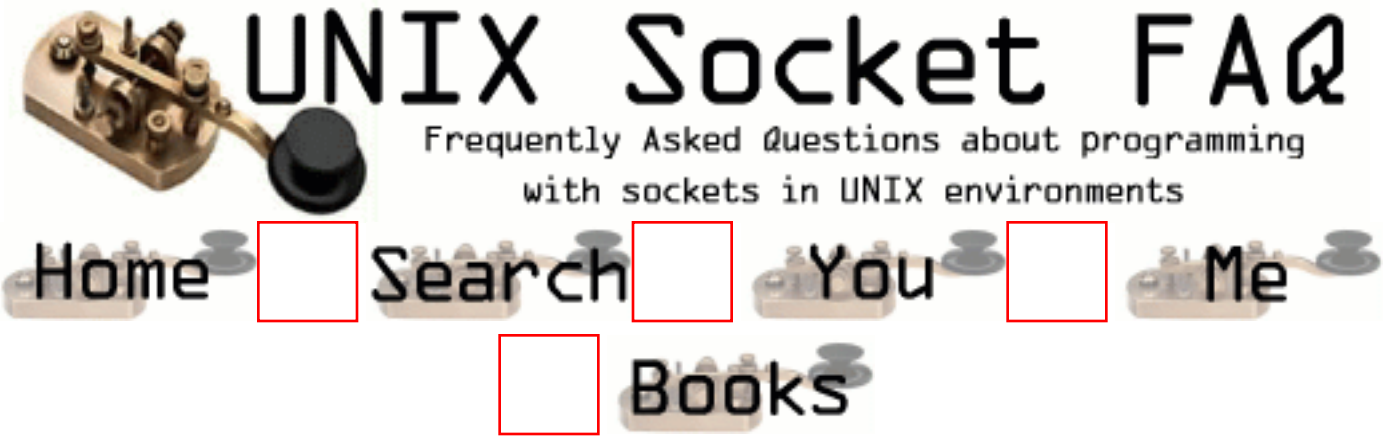
Screwtape, you're wrong: I'm sitting at a "Linux 2.2" system right now, and I can assure you it has both inet_pton() and getaddrinfo()... And, in fact, I've got software sitting right here on the machine, which USES both of those functions, and works quite well... (To be more specific, the system is running Red Hat 6.2... But, I've used the functions since at least RH 6.0... But, really, all that matters is the version of glibc, since THAT is what is implementing them; the version of the kernel is not really relevent...) You'll find getaddrinfo() defined in "/usr/include/netdb.h", along with the standard gethostby*() functions... And, you'll find inet_pton() defined in "/usr/include/arpa/inet.h", along with inet_addr(), and all the various other similar functions...

But, more than just being available under Linux (or any other platform glibc is available on), those functions are defined as part of the POSIX.1g standard, now... So, any OS that is POSIX complaint SHOULD be implementing them... But, given that the standards are fairly new still (and, last I heard, were still in flux), I'm sure many still don't support them... Regardless, I still maintain that IF they are available to you, you should use them... If they just aren't available on the platform you need to write code for, then obviously you can't use them... (Well, actually, you COULD, because there are available source code implementations, which you could borrow... If you don't want to dig the code out of glibc, I believe the late, great, and dearly missed W. Richard Stevens made available implementations in the source code for his wonderful "Unix Network Programming" book... Yep, you can get the code right here... Then, after extracting it, you'll find the getaddrinfo() implementation under "unpv12e/libgai/", and inet_pton() under "unpv12e/libfree/"...)

From: Rob Seace

Oh, and if you're interested in more info on the use of getaddrinfo(), inet_pton(), and various other relatively new (and, better) socket functions, I would recommend reading RFC-2553, which details various socket API changes necessary for dealing with IPv6, and mentions both getaddrinfo() and inet_pton()...

Add a Comment



Writing Client Applications (TCP/SOCK_STREAM): How can my client work through a firewall/proxy server?

If you are running through separate proxies for each service, you shouldn't need to do anything. If you are working through sockd, you will need to "socksify" your application. Details for doing this can be found in the package itself, which is available at:

ftp://coast.cs.purdue.edu/pub/tools/unix/socks/

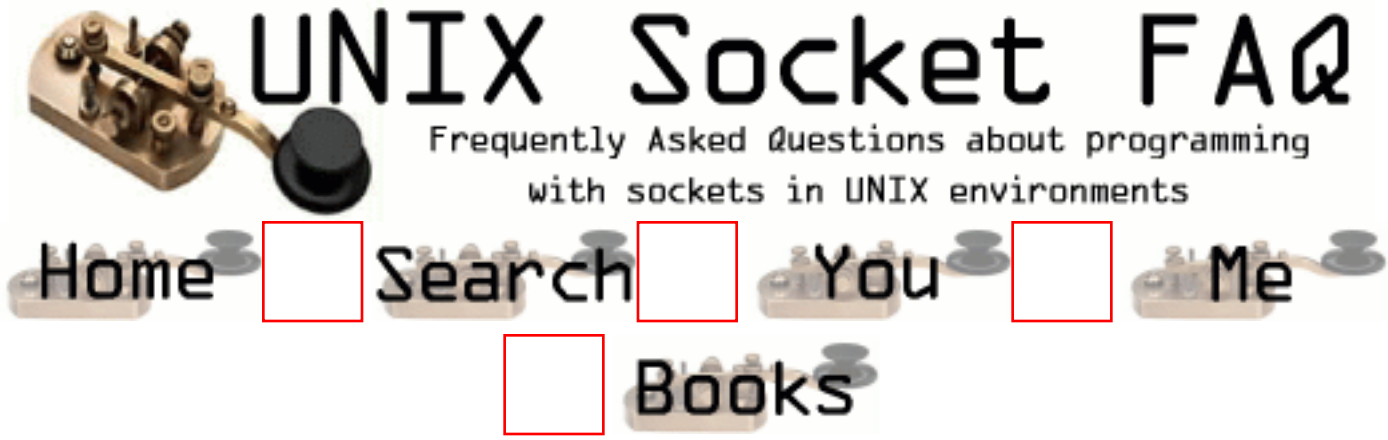
From: Vic Metcalfe

Some have reported that they can't get to the coast archive because it blocks hosts which do not have a PTR DNS record for reverse lookups. Here are alternative (and probably better) locations for socks protocol information:

ftp://ftp.nec.com:/pub/socks/ for Socks V4 stuff
http://www.socks.nec.com/ for Socks V5 stuff

Take care,
Vic.

Add a Comment



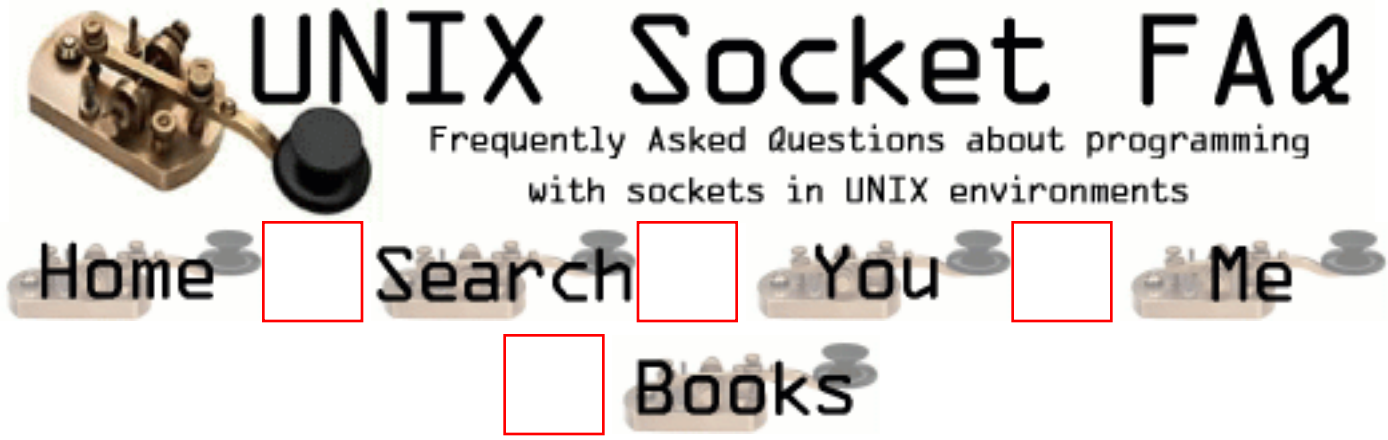
Writing Client Applications (TCP/SOCK_STREAM): Why does connect() succeed even before my server did an accept()?

From Andrew Gierth (andrew@erlenstar.demon.co.uk):

Once you have done a `listen()` call on your socket, the kernel is primed to accept connections on it. The usual UNIX implementation of this works by *immediately* completing the SYN handshake for any incoming valid SYN segments (connection attempts), creating the socket for the new connection, and keeping this new socket on an internal queue ready for the `accept()` call. So the socket is fully open *before* the accept is done.

The other factor in this is the 'backlog' parameter for `listen()`; that defines how many of these completed connections can be queued at one time. If the specified number is exceeded, then new incoming connects are simply ignored (which causes them to be retried).

Add a Comment



Writing Client Applications (TCP/SOCK_STREAM): Why do I sometimes lose a server's address when using more than one server?

From Andrew Gierth (andrew@erlenstar.demon.co.uk):

Take a careful look at struct hostent. Notice that almost everything in it is a pointer? *All* these pointers will refer to statically allocated data.

For example, if you do:

```
struct hostent *host = gethostbyname(hostname);
```

then (as you should know) a subsequent call to `gethostbyname()` will overwrite the structure pointed to by 'host'.

But if you do:

```
struct hostent myhost;
struct hostent *hostptr = gethostbyname(hostname);
if (hostptr) myhost = *host;
```

to make a copy of the `hostent` before it gets overwritten, then it *still* gets clobbered by a subsequent call to `gethostbyname()`, since although `myhost` won't get overwritten, all the data it is pointing to will be.

You can get round this by doing a proper 'deep copy' of the `hostent` structure, but this is tedious. My recommendation would be to extract the needed fields of the `hostent` and store them in your own way.

Robin Paterson (etmrpat@etm.ericsson.se) has added:

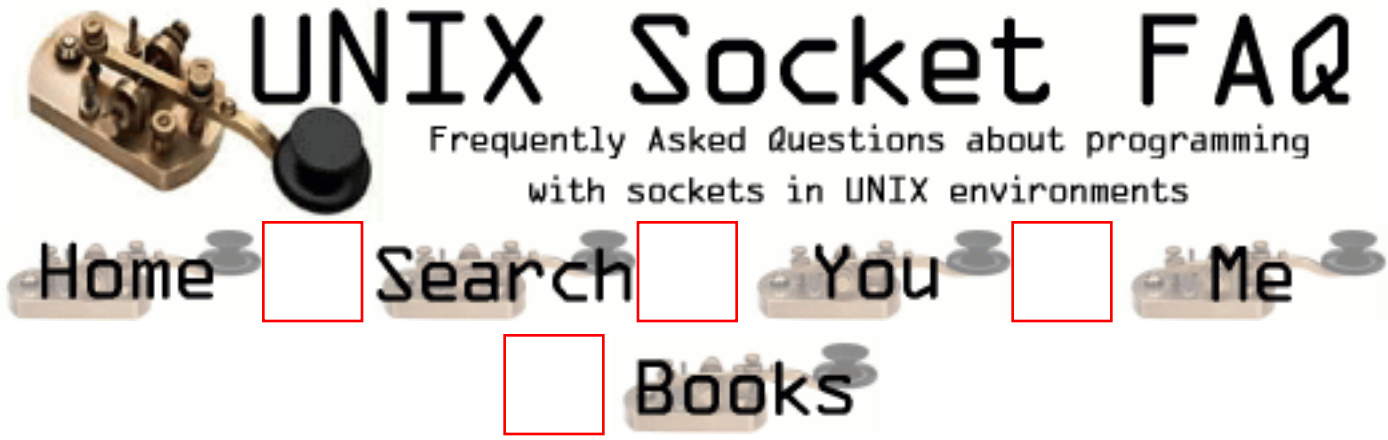
It might be nice if you mention MT safe libraries provide complimentary functions for multithreaded programming. On the solaris machine I'm typing at, we have `gethostbyname` and `gethostbyname_r` (`_r` for reentrant). The main difference is, *you* provide the storage for the hostent struct so you always have a local copy and not just a pointer to the static copy.

From: tvrao

respected sir,
please explain when we click any hyperlink
what are the procedures takes place before getting the require
data.

with regards,
t.v.rao

Add a Comment



Writing Client Applications (TCP/SOCK_STREAM): How can I set the timeout for the connect() system call?

From Richard Stevens (rstevens@noao.edu):

Normally you cannot change this. Solaris does let you do this, on a per-kernel basis with the ndd `tcp_ip_abort_cinterval` parameter.

The easiest way to shorten the connect time is with an `alarm()` around the call to `connect()`. A harder way is to use `select()`, after setting the socket nonblocking. Also notice that you can only shorten the connect time, there's normally no way to lengthen it.

From Andrew Gierth (andrew@erlenstar.demon.co.uk):

First, create the socket and put it into non-blocking mode, then call connect(). There are three possibilities:

- connect succeeds: the connection has been successfully made (this usually only happens when connecting to the same machine)
- connect fails: obvious
- connect returns -1/EINPROGRESS. The connection attempt has begun, but not yet completed.

If the connection succeeds:

- the socket will select() as writable (and will also select as readable if data arrives)

If the connection fails:

- the socket will select as readable *and* writable, but either a read or write will return the error code from the connection attempt. Also, you can use getsockopt(SO_ERROR) to get the error status - but be careful; some systems return the error code in the result parameter of getsockopt, but others (incorrectly) cause the getsockopt call *itself* to fail with the stored value as the error.

From: Jukka Santala

This answer doesn't cover how to force the actual _timeout_. I assume a simple close() will do, but it'd be nice to have the FAQ bit answer the actual question. (And yes, this is actually a situation that came up, in a server where identd query will be waited for limited time before letting the main connection in)

From: jagadeesh

Hello,
Try this

```
sts = connect(mysock, &msaddr, sizeof (msaddr));
    if (sts)
    {
if (errno == ECONNREFUSED)
{
printf("connection refused\n");
return;
}
else if (errno == ETIMEDOUT)
{
printf("connection attempt timed out\n");
return;
```

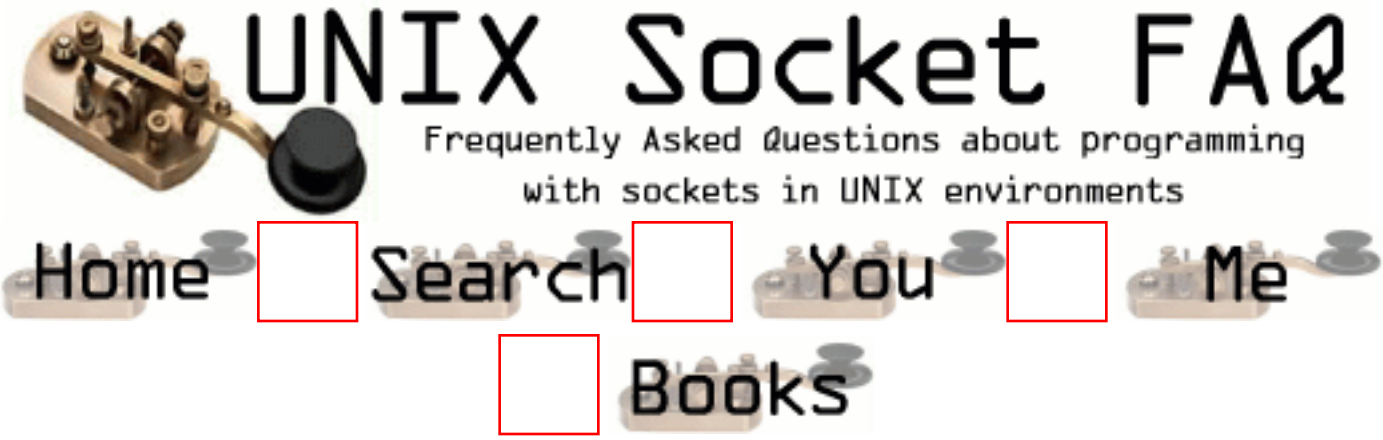
From: Tan Nguyen

Nope, that doesn't solve the problem at all. For example, we attempt to connect to a non-existent host (ie 209.blah.blah.blah), then connect() just keeps trying to look up the routing table for the host. Thus, you never get a return from connect().

From: Oren Bassik

I'm not sure about this, but if you get an alarm signal while connecting, it will return with EINTR.

Add a Comment



Writing Client Applications (TCP/SOCK_STREAM): Should I bind() a port number in my client program, or let the system choose one for me on the connect() call?

From Andrew Gierth (andrew@erlenstar.demon.co.uk):

**** Let the system choose your client's port number ****

The exception to this, is if the server has been written to be picky about what client ports it will allow connections from. Rlogind and rshd are the classic examples. This is usually part of a Unix-specific (and rather weak) authentication scheme; the intent is that the server allows connections only from processes with root privilege. (The weakness in the scheme is that many O/Ss (e.g. MS-DOS) allow anyone to bind any port.)

The `rresvport()` routine exists to help out clients that are using this scheme. It basically does the equivalent of `socket()` + `bind()`, choosing a port number in the range 512..1023.

If the server is not fussy about the *client's* port number, then don't try and assign it yourself in the client, just let `connect()` pick it for you.

If, in a client, you use the naive scheme of starting at a fixed port number and calling `bind()` on consecutive values until it works, then you buy yourself a whole lot of trouble:

The problem is if the server end of your connection does an active close. (E.G. client sends 'QUIT' command to server, server responds by closing the connection). That leaves the client end of the connection in CLOSED state, and the server end in TIME_WAIT state. So after the client exits, there is no trace of the connection on the client end.

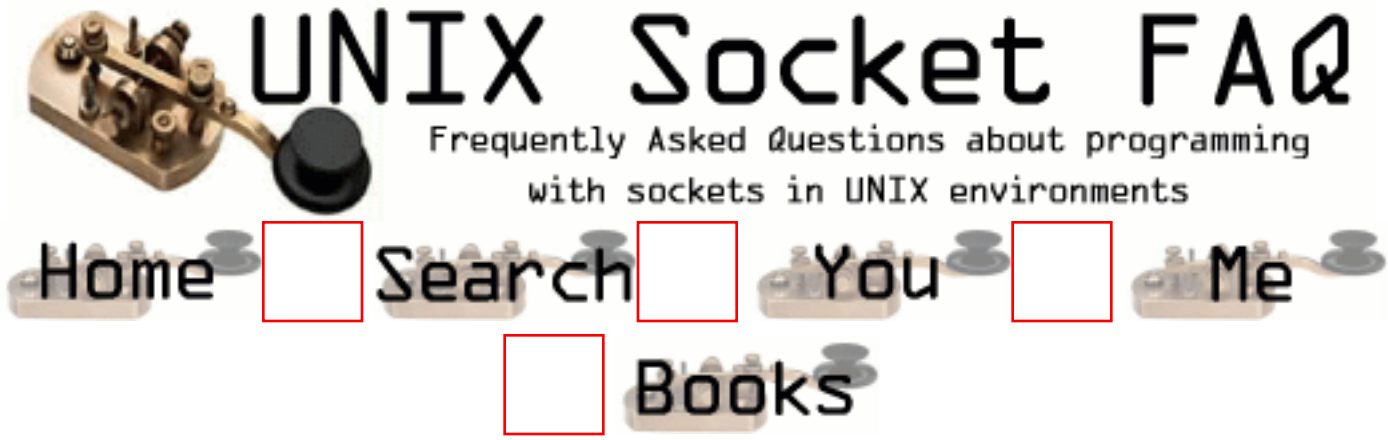
Now run the client again. It will pick the same port number, since as far as it can see, it's free. But as soon as it calls `connect()`, the server finds that you are trying to duplicate an existing connection (although one in TIME_WAIT). It is perfectly entitled to refuse to do this, so you get, I suspect, `ECONNREFUSED` from `connect()`. (Some systems may sometimes allow the

connection anyway, but you *can't* rely on it.)

This problem is *especially* dangerous because it doesn't show up unless the client and server are on *different* machines. (If they are the same machine, then the client *won't* pick the same port number as before). So you can get bitten well into the development cycle (if you do what I suspect most people do, and test client & server on the same box initially).

Even if your protocol has the client closing first, there are still ways to produce this problem (e.g. kill the server).

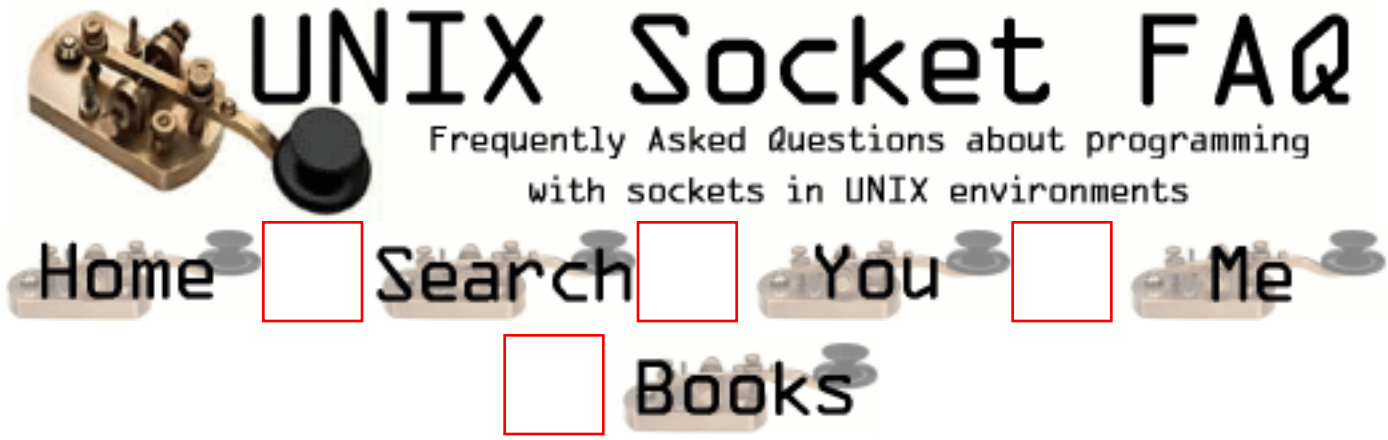
Add a Comment



Writing Client Applications (TCP/SOCK_STREAM): Why do I get "connection refused" when the server isn't running?

The `connect()` call will only block while it is waiting to establish a connection. When there is no server waiting at the other end, it gets notified that the connection can not be established, and gives up with the error message you see. This is a good thing, since if it were not the case clients might wait for ever for a service which just doesn't exist. Users would think that they were only waiting for the connection to be established, and then after a while give up, muttering something about crummy software under their breath.

Add a Comment



Writing Client Applications (TCP/SOCK_STREAM): What does one do when one does not know how much information is comming over the socket? Is there a way to have a dynamic buffer?

This question asked by Niranjan Perera (perera@mindspring.com).

When the size of the incoming data is unknown, you can either make the size of the buffer as big as the largest possible (or likely) buffer, or you can re-size the buffer on the fly during your read. When you `malloc()` a large buffer, most (if not all) varients of unix will only allocate address space, but not physical pages of ram. As more and more of the buffer is used, the kernel allocates physical memory. This means that malloc'ing a large buffer will not waste resources unless that memory is used, and so it is perfectly acceptable to ask for a meg of ram when you expect only a few K.

On the other hand, a more elegant solution that does not depend on the inner workings of the kernel is to use realloc() to expand the buffer as required in say 4K chunks (since 4K is the size of a page of ram on most systems). I may add something like this to sockhelp.c in the example code one day.

From: Sujay

I am really failing to understand the malloc part ....help

From: Vic Metcalfe

Lets say your application does this: `buf = (char *)malloc(10000000000);` Your machine has only 128MB of ram. If the OS actually tried to allocate that much RAM, it would exhaust the system's resources. What most operating systems do is allocate only the memory that is *used*. On most systems that I've used, memory allocation is done in 4K "pages". The above `malloc()` doesn't actually allocate anything but address space. No physical RAM is

allocated. So if I then added... `buf[0] = 0; buf[5] = 0; buf[9999999999] = 0;` The first assignment would cause a page-fault because the memory for it has not yet been allocated. The OS traps the page-fault, sees that it did promise that memory to an application, allocates it and allows the assignment to continue. Now, assuming a page size of 4K the first 4096 characters of the buffer have been allocated. The second assignment produces no page fault and no additional memory is allocated. Now what do you expect to happen with the last assign? The buffer at that possition has not been mapped to physical memory yet, so it too produces a page fault. Once again the OS traps the error, allocates the memory and allows the assignment to continue. Note that it does **not** allocate all pages between the two blocks. The OS will have only allocated two blocks for a total of 8K of physical RAM. If you know that the OS you are targetting does physical allocation of memory on demand you can take advantage of the fact. In this example you can use it to create an expanding buffer with no tricky coding required. You could also use it for a very sparse array. The more correct thing to do is `realloc()` the buffer as required. I've coded this sort of thing a few times, but of course now that I want to pull out an example to share with you I can't find one. It isn't complicated anyway, you just look for the buffer to fill, and then `realloc()` it a bit bigger each time it fill up. Hope this helps, Vic.

From: Garen Parham

A good way to handle the unexpected length of data which could be coming in is to use a fixed-length buffer which could be the largest size of data you expect to receive, but in one read() you may not get it all into your static-length buffer either because it's too short or there wasn't enough data in the kernels receive queue at the time of the read(). The fixed length buffer could then be used as a kind of ring queue data structure, if you reach the end of your buffer and didn't receive all of what you expected (say for instance with a line protocol, you didn't get the \r\n or \n (CR-LF or LF)) you could write that over the beginning of your fixed length buffer, update the write position to point past that so the next read concatenates it for you and so on.

From:

test1

From:

test2

From:

test3

From:

test4

From: David Gillies

The technique I use for streaming data back from a connection
when I don't know how much data is coming back is simply
to loop until read returns 0 (non-fatal error) or a negative

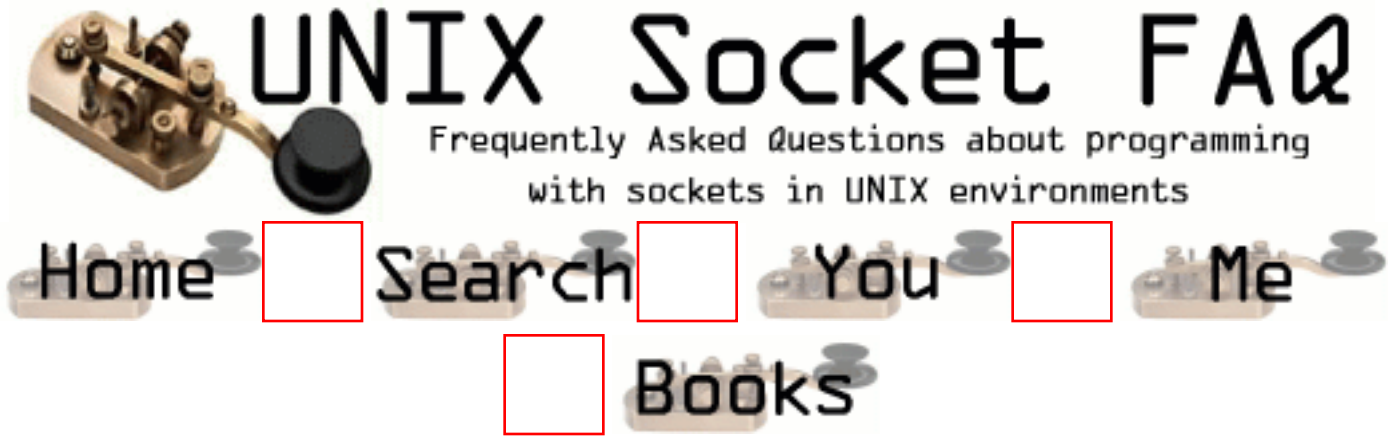
number. I have a function ReadBufferedData which looks like this:

```
/*****************************************************************

  ReadBufferedData

  Given a socket to read from, read data until
  the socket is empty or the buffer is full.
  Return the number of bytes that were read.

  readSocket I the socket to read from

  buffer O the buffer to read into

  bufLen I the size of the buffer

  bytesRead O the number of bytes read

  errnoBack O the value of errno, if any,
encountered in the function

  Returns: status code indicating success - noErr = success

  *****************************************************************/

OSErr ReadBufferedData(const int readSocket,char *const buffer,
const size_t bufLen,size_t *const bytesRead,
int *const errnoBack)
{
OSErr readErr=noErr;
size_t bytesLeft,bytesThisTime,bytesSoFar=0UL;
Boolean done=FALSE;

bytesLeft=bufLen;

do
{
bytesThisTime=recv(readSocket,buffer+bytesSoFar,bytesLeft,0);

if(bytesThisTime==0)
{
readErr=socketEOFErr;
done=TRUE;
}
```

```
if(bytesThisTime==-1)
{
readErr=socketReadErr;
*errnoBack=errno;
done=TRUE;
}
else
{
bytesSoFar+=bytesThisTime;
bytesLeft-=bytesThisTime;

if(bytesLeft==0)
done=TRUE;
}
}while(!done);

if((readErr==noErr)||(readErr==socketEOFErr))
*bytesRead=bytesSoFar;

return readErr;
}
```

OSErr is a typedef for short (like on a Mac) with noErr=0.
I typically use this in a loop, going read-blocked with
select() each time I get socketEOFErr until I get
socketReadErr.

Add a Comment



Writing Client Applications (TCP/SOCK_STREAM): How can I determine the local port number?

From: Fajun Shi (fajun@cs.msstate.edu):

Hi, my question is: When I write a client, how can I know the port number that the socket bound in my machine?

From: Jörg Jensch

Hi!

Use getsockname(3N) to find out the local port
number your client application uses. Here is a
simple example:

```
len = sizeof ( server );
if ( getsockname ( sock, &server, &len ) < 0 )
  perror ( "getsockname" );
  else
    fprintf ( stderr, "local port number before connect: %d\n", ntohs ( ((struct sockaddr_in
*)&server)->sin_port ));

    connected = connect(sock, (struct sockaddr *) &address, sizeof(address));

    len = sizeof ( server );
    if ( getsockname ( sock, &server, &len ) < 0 )
      perror ( "getsockname" );
      else
        fprintf ( stderr, "local port number after connect : %d\n", ntohs ( ((struct sockaddr_in
*)&server)->sin_port ));
```

And here is the output I retrieve:

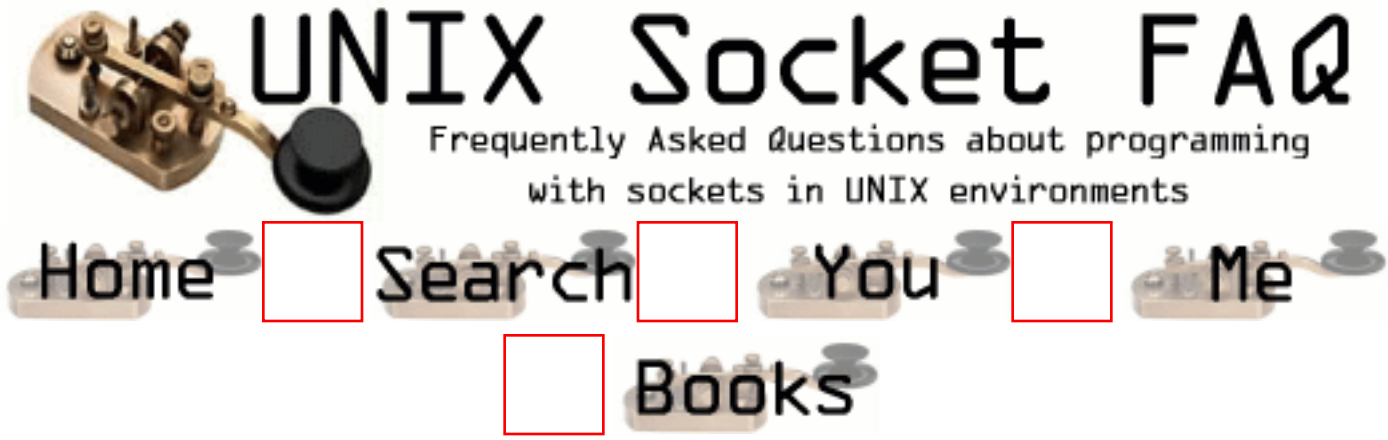
local port number before connect: 0
local port number after connect : 34994

As can be seen, the system do not bind any
local port number for a socket before the
client calls connect().

Greetings

Jayjay

Add a Comment



Writing Server Applications (TCP/SOCK_STREAM): How come I get "address already in use" from bind()?

You get this when the address is already in use. (Oh, you figured that much out?) The most common reason for this is that you have stopped your server, and then re-started it right away. The sockets that were used by the first incarnation of the server are still active. This is further explained in 2.7 Please explain the TIME_WAIT state., and 2.5 How do I properly close a socket?.

Add a Comment

Writing Server Applications (TCP/SOCK_STREAM): Why don't my sockets close?

When you issue the `close()` system call, you are closing your interface to the socket, not the socket itself. It is up to the kernel to close the socket. Sometimes, for really technical reasons, the socket is kept alive for a few minutes after you close it. It is normal, for example for the socket to go into a TIME_WAIT state, on the server side, for a few minutes. People have reported ranges from 20 seconds to 4 minutes to me. The official standard says that it should be 4 minutes. On my Linux system it is about 2 minutes. This is explained in great detail in 2.7 Please explain the TIME_WAIT state..

From: Erik landry

Also, the kernel will NOT shutdown a socket on a close(2) if another process (e.g., a child that inherited the file descriptor) still has the socket open. For more information, see shutdown(2) or 2.2 When should I use shutdown()

Add a Comment

Writing Server Applications (TCP/SOCK_STREAM): How can I make my server a daemon?

There are two approaches you can take here. The first is to use inetd to do all the hard work for you. The second is to do all the hard work yourself.

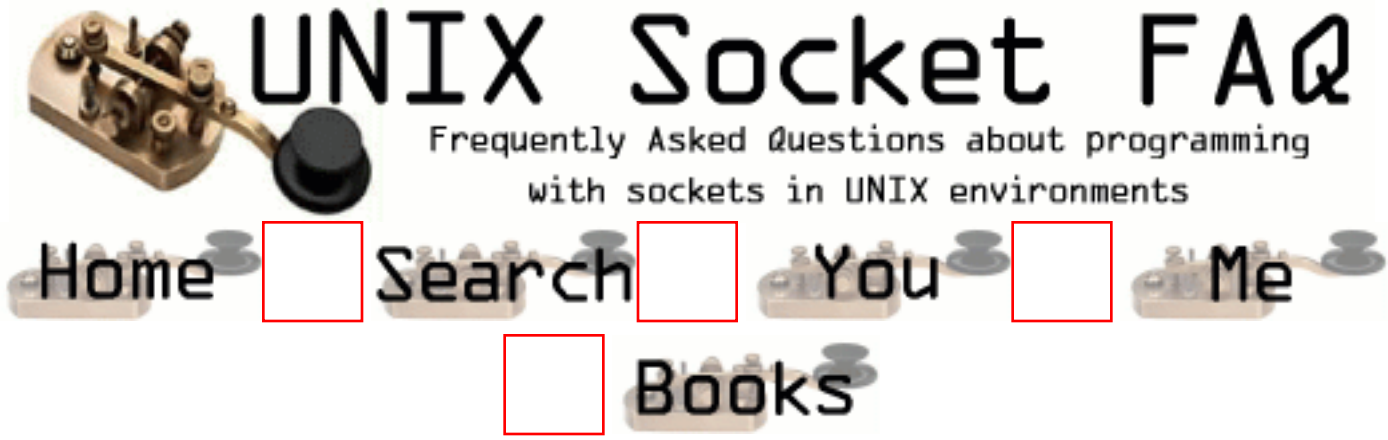
If you use inetd, you simply use `stdin`, `stdout`, or `stderr` for your socket. (These three are all created with `dup()` from the real socket) You can use these as you would a socket in your code. The inetd process will even close the socket for you when you are done. For more information on setting this up, look at the man page for inetd.

If you wish to write your own server, there is a detailed explanation in "Unix Network Programming" by Richard Stevens (see 1.6 Where can I get source code for the book [book title]?). I also picked up this posting from comp.unix.programmer, by Nikhil Nair (nn201@cus.cam.ac.uk). You may want to add code to ignore SIGPIPE, because if this signal is not dealt with, it will cause your application to exit. (Thanks to ingo@milan2.snafu.de for pointing this out).

```
I worked all this lot out from the GNU C Library Manual (on-line
documentation).  Here's some code I wrote - you can adapt it as necessary:


#include
#include
#include
#include
#include
#include
#include

/* Global variables */
...
volatile sig_atomic_t keep_going = 1; /* controls program termination */


/* Function prototypes: */
...
void termination_handler (int signum); /* clean up before termination */


int
main (void)
{
  ...
```

```
  if (chdir (HOME_DIR))         /* change to directory containing data
                                   files */
   {
     fprintf (stderr, "`%s': ", HOME_DIR);
     perror (NULL);
     exit (1);
   }

   /* Become a daemon: */
   switch (fork ())
     {
     case -1:                    /* can't fork */
       perror ("fork()");
       exit (3);
     case 0:                     /* child, process becomes a daemon: */
       close (STDIN_FILENO);
       close (STDOUT_FILENO);
       close (STDERR_FILENO);
       if (setsid () == -1)      /* request a new session (job control) */
         {
           exit (4);
         }
       break;
     default:                    /* parent returns to calling process: */
       return 0;
     }

   /* Establish signal handler to clean up before termination: */
   if (signal (SIGTERM, termination_handler) == SIG_IGN)
     signal (SIGTERM, SIG_IGN);
   signal (SIGINT, SIG_IGN);
   signal (SIGHUP, SIG_IGN);

   /* Main program loop */
   while (keep_going)
     {
       ...
     }
   return 0;
}

void
termination_handler (int signum)
{
  keep_going = 0;
  signal (signum, termination_handler);
}
```

Add a Comment



Writing Server Applications (TCP/SOCK_STREAM): How can I listen on more than one port at a time?

The best way to do this is with the `select()` call. This tells the kernel to let you know when a socket is available for use. You can have one process do i/o with multiple sockets with this call. If you want to wait for a connect on sockets 4, 6 and 10 you might execute the following code snippet:

```
fd_set socklist;

FD_ZERO(&socklist); /* Always clear the structure first. */
FD_SET(4, &socklist);
FD_SET(6, &socklist);
FD_SET(10, &socklist);
if (select(11, NULL, &socklist, NULL, NULL) < 0)
  perror("select");
```

The kernel will notify us as soon as a file descriptor which is less than 11 (the first parameter to `select()`), and is a member of our `socklist` becomes available for writing. See the man page on `select()` for more details.

From: Chris Niekel

I tested on IRIX 6.5, and the fd_set should **NOT** be in the writeset, but in the read-set. I.e. select(11, &fds, 0, 0, 0). If you put it in the write-set IRIX 6.5 lets you wait forever.

From: Gavin

How would you do this while using poll()?

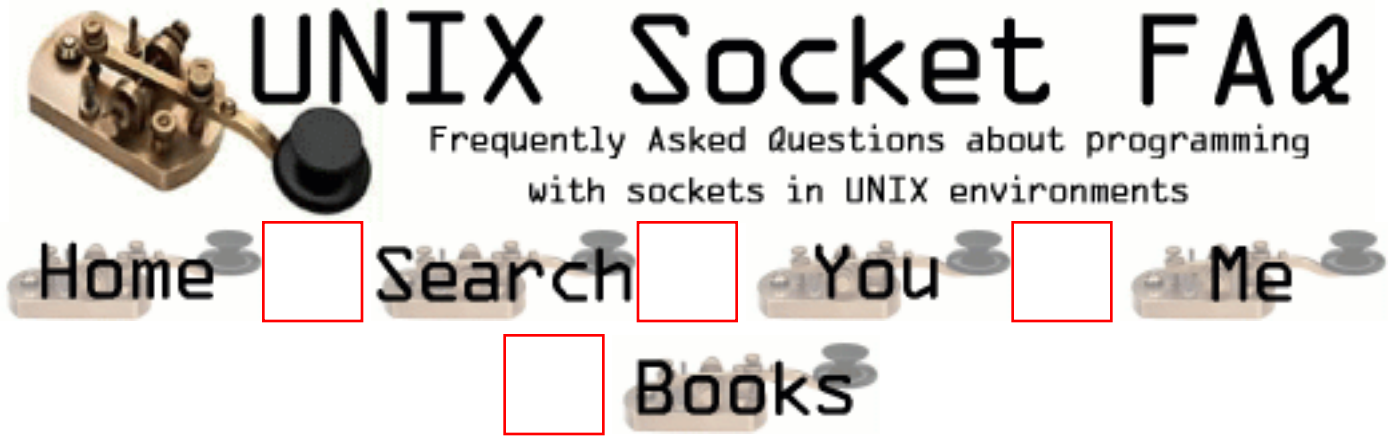
Add a Comment

Writing Server Applications (TCP/SOCK_STREAM): What exactly does SO_REUSEADDR do?

This socket option tells the kernel that even if this port is busy (in the TIME_WAIT state), go ahead and reuse it anyway. If it is busy, but with another state, you will still get an address already in use error. It is useful if your server has been shut down, and then restarted right away while sockets are still active on its port. You should be aware that if any unexpected data comes in, it may confuse your server, but while this is possible, it is not likely.

It has been pointed out that "A socket is a 5 tuple (proto, local addr, local port, remote addr, remote port). SO_REUSEADDR just says that you can reuse local addresses. The 5 tuple still must be unique!" by Michael Hunter (mphunter@qnx.com). This is true, and this is why it is very unlikely that unexpected data will ever be seen by your server. The danger is that such a 5 tuple is still floating around on the net, and while it is bouncing around, a new connection from the same client, on the same system, happens to get the same remote port. This is explained by Richard Stevens in 2.7 Please explain the TIME_WAIT state..

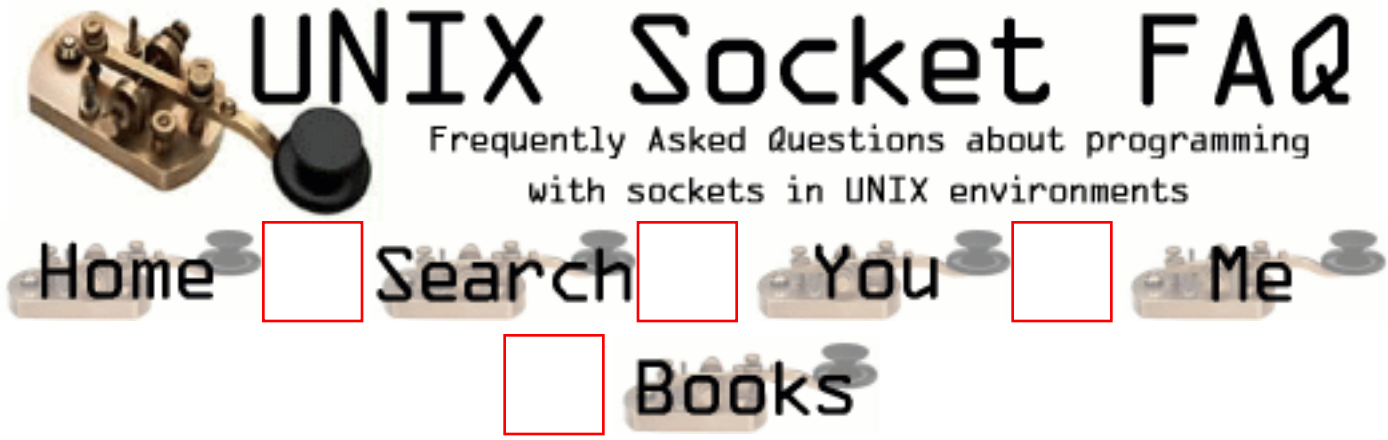
Add a Comment



Writing Server Applications (TCP/SOCK_STREAM): What exactly does SO_LINGER do?

On some unixes this does nothing. On others, it instructs the kernel to abort tcp connections instead of closing them properly. This can be dangerous. If you are not clear on this, see 2.7 Please explain the TIME_WAIT state..

Add a Comment



Writing Server Applications (TCP/SOCK_STREAM): What exactly does SO_KEEPALIVE do?

From Andrew Gierth (andrew@erlenstar.demon.co.uk):

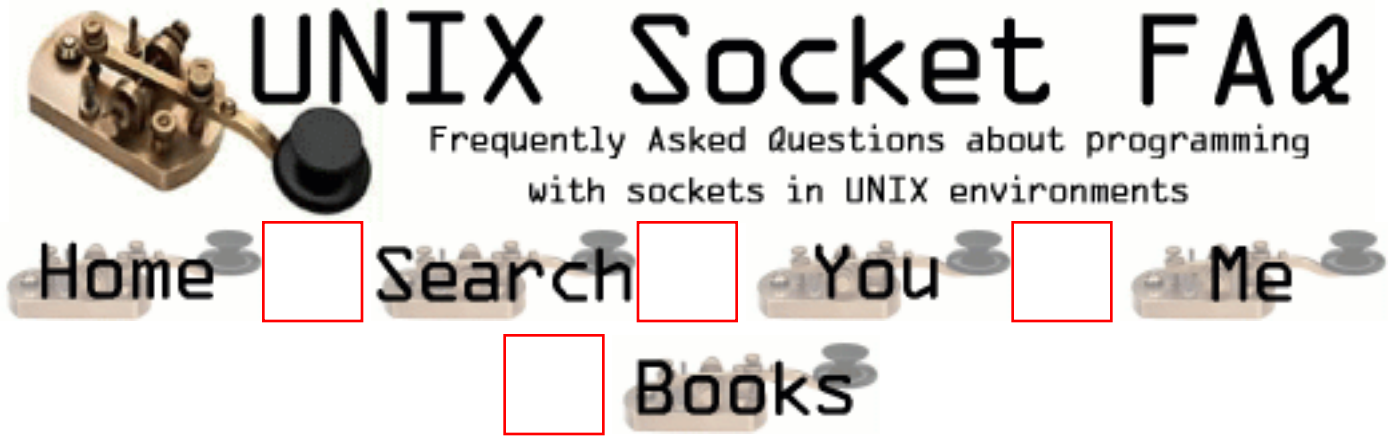
The `SO_KEEPALIVE` option causes a packet (called a 'keepalive probe') to be sent to the remote system if a long time (by default, more than 2 hours) passes with no other data being sent or received. This packet is designed to provoke an ACK response from the peer. This enables detection of a peer which has become unreachable (e.g. powered off or disconnected from the net). See 2.8 Why does it take so long to detect that the peer died? for further discussion.

Note that the figure of 2 hours comes from RFC1122, "Requirements for Internet Hosts". The precise value should be configurable, but I've often found this to be difficult. The only implementation I know of that allows the keepalive interval to be set per-connection is SVR4.2.

From: Mahmoud Chilali

the value is configurable via sysctl on FreeBSD and friends. on these systems, keepalive is set by default on a per system basis, so applications do not need to set the option.

Add a Comment



Writing Server Applications (TCP/SOCK_STREAM): 4.8 How can I bind() to a port number < 1024?

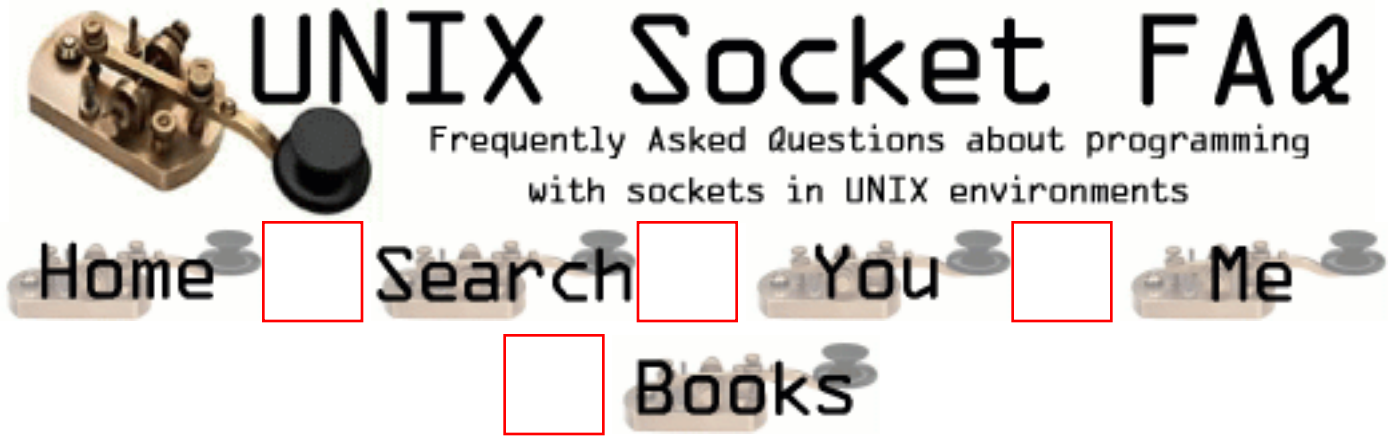
From Andrew Gierth (andrew@erlenstar.demon.co.uk):

The restriction on access to ports < 1024 is part of a (fairly weak) security scheme particular to UNIX. The intention is that servers (for example rlogind, rshd) can check the port number of the client, and if it is < 1024, assume the request has been properly authorised at the client end.

The practical upshot of this, is that binding a port number < 1024 is reserved to processes having an effective UID == root.

This can, occasionally, itself present a security problem, e.g. when a server process needs to bind a well-known port, but does *not* itself need root access (news servers, for example). This is often solved by creating a small program which simply binds the socket, then restores the real userid and `exec()`s the real server. This program can then be made setuid root.

Add a Comment



Writing Server Applications (TCP/SOCK_STREAM): How do I get my server to find out the client's address / hostname?

From Andrew Gierth (andrew@erlenstar.demon.co.uk):

After `accept()`ing a connection, use `getpeername()` to get the address of the client. The client's address is of course, also returned on the `accept()`, but it is essential to initialise the address-length parameter before the accept call for this will work.

Jari Kokko (jkokko@cc.hut.fi) has offered the following code to determine the client address:

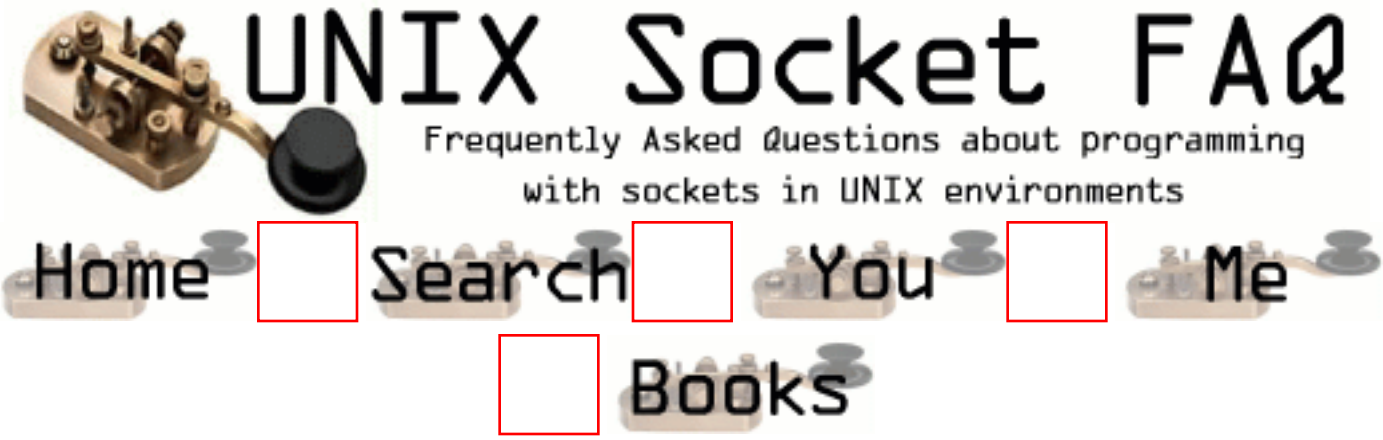
```
int t;
int len;
struct sockaddr_in sin;
struct hostent *host;

len = sizeof sin;
if (getpeername(t, (struct sockaddr *) &sin, &len) < 0)
        perror("getpeername");
else {
        if ((host = gethostbyaddr((char *) &sin.sin_addr,
                                  sizeof sin.sin_addr,
                                  AF_INET)) == NULL)
            perror("gethostbyaddr");
        else printf("remote host is '%s'\n", host->h_name);
}
```

From: sameer f parker

In a "web server-proxy server- client" system, if i want to
print the source & destination IP addresses and the respective

ports at "proxy server" level.How to do it?

Add a Comment



Writing Server Applications (TCP/SOCK_STREAM): How should I choose a port number for my server?

The list of registered port assignments can be found in STD 2 or RFC 1700. Choose one that isn't already registered, and isn't in /etc/services on your system. It is also a good idea to let users customize the port number in case of conflicts with other un-registered port numbers in other servers. The best way of doing this is hardcoding a service name, and using `getservbyname()` to lookup the actual port number. This method allows users to change the port your server binds to by simply editing the /etc/services file.

From: Stephen Satchell

In the course of researching this question, I found a reference in some BSD documentation that port numbers from 48K to 64K should be used by applications for well-known-socket numbers. So in an application I wrote, I used 54001 as the well-known port.

Wouldn't you know, I found a TCP/IP product that wouldn't allow me to use that number! That product (name withheld to protect the guilty) allows socket numbers from zero to 32768. (NOT 32767, the usual boundry condition. Surprised me, it did.) So I provided for setting an alternate port address for those people using the *** ******* products. Other products worked just fine with 54001.

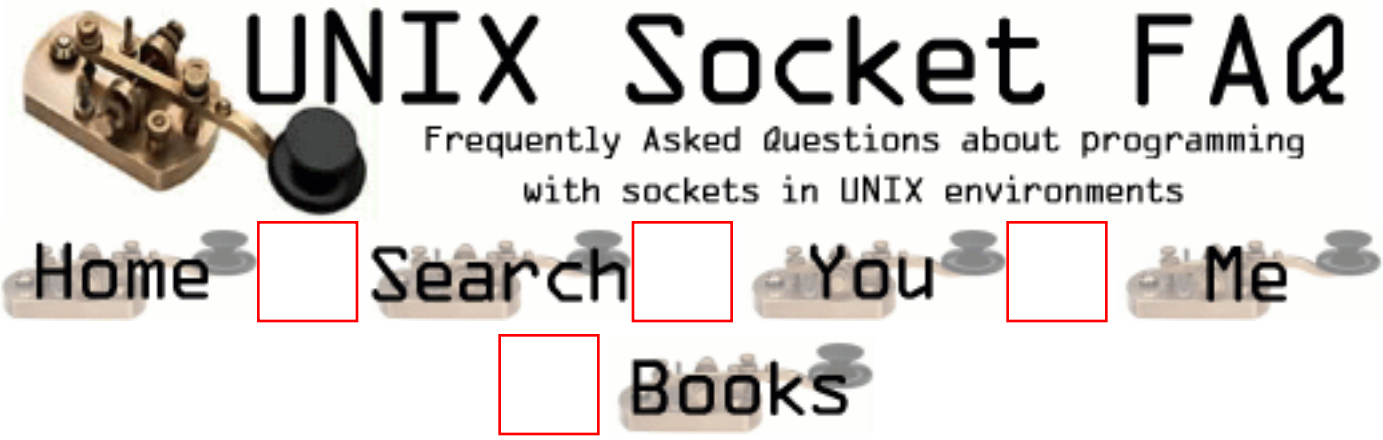
From: Charles E. Campbell, Jr

If you plan to distribute your server, you should consider registering your server with IANA (http://www.iana.org/)

From: Warren Nash

look in /etc/services or choose an number above 1024.

Add a Comment



Writing Server Applications (TCP/SOCK_STREAM): What is the difference between SO_REUSEADDR and SO_REUSEPORT?

SO_REUSEADDR allows your server to bind to an address which is in a TIME_WAIT state. It does not allow more than one server to bind to the same address. It was mentioned that use of this flag can create a security risk because another server can bind to a the same port, by binding to a specific address as opposed to INADDR_ANY. The SO_REUSEPORT flag allows multiple processes to bind to the same address provided all of them use the SO_REUSEPORT option.

From Richard Stevens (rstevens@noao.edu):

This is a newer flag that appeared in the 4.4BSD multicasting code (although that code was from elsewhere, so I am not sure just who invented the new SO_REUSEPORT flag).

What this flag lets you do is rebind a port that is already in use, but only if all users of the port specify the flag. I believe the intent is for multicasting apps, since if you're running the same app on a host, all need to bind the same port. But the flag may have other uses. For example the following is from a post in February:

From Stu Friedberg (stuartf@sequent.com):

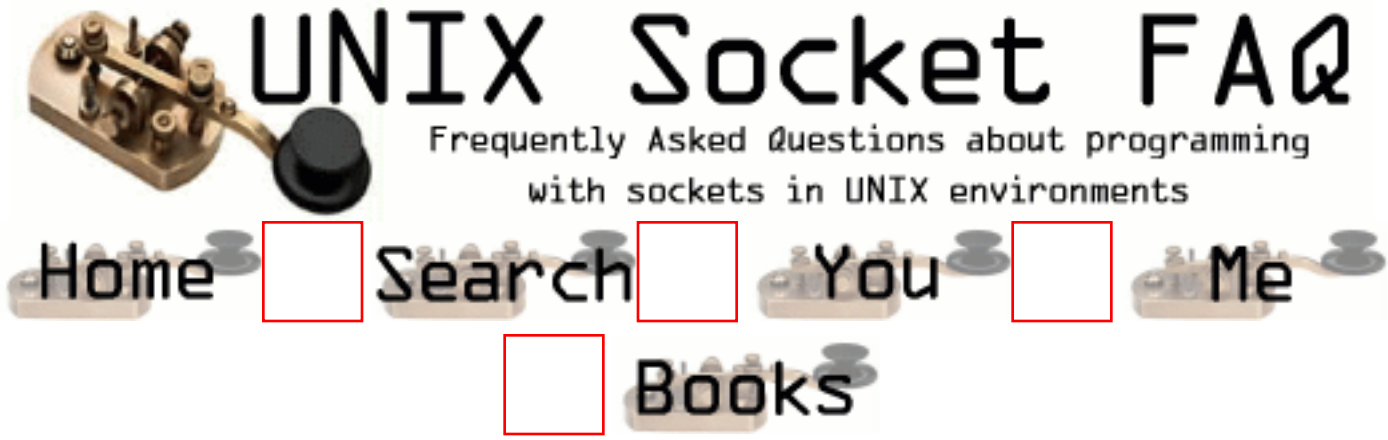
> SO_REUSEPORT is also useful for eliminating the try-10-times-to-bind hack in ftpd's data connection setup routine. Without SO_REUSEPORT, only one ftpd thread can bind to TCP (lhost, lport, INADDR_ANY, 0) in preparation for connecting back to the client. Under conditions of heavy load, there are more threads colliding here than the try-10-times hack can accomodate. With SO_REUSEPORT, things work nicely and the hack becomes unnecessary.

I have also heard that DEC OSF supports the flag. Also note that under 4.4BSD, if you are binding a multicast address, then SO_REUSEADDR is condisered the same as SO_REUSEPORT

(p. 731 of "TCP/IP Illustrated, Volume 2"). I think under Solaris you just replace SO_REUSEPORT with SO_REUSEADDR.

From a later Stevens posting, with minor editing:

Basically SO_REUSEPORT is a BSD'ism that arose when multicasting was added, even thought it was not used in the original Steve Deering code. I believe some BSD-derived systems may also include it (OSF, now Digital Unix, perhaps?). SO_REUSEPORT lets you bind the same address *and* port, but only if all the binders have specified it. But when binding a multicast address (its main use), SO_REUSEADDR is considered identical to SO_REUSEPORT (p. 731, "TCP/IP Illustrated, Volume 2"). So for portability of multicasting applications I always use SO_REUSEADDR.



Writing Server Applications (TCP/SOCK_STREAM): How can I write a multi-homed server?

The original question was actually from Shankar Ramamoorthy (shankar@viman.com):

> *I want to run a server on a multi-homed host. The host is part of two networks and has two ethernet cards. I want to run a server on this machine, binding to a pre-determined port number. I want clients on either subnet to be able to send broadcast packates to the port and have the server receive them.*

And answered by Andrew Gierth (andrew@erlenstar.demon.co.uk):

Your first question in this scenario is, do you need to know which subnet the packet came from? I'm not at all sure that this can be reliably determined in all cases.

If you don't really care, then all you need is one socket bound to `INADDR_ANY`. That simplifies things greatly.

If you *do* care, then you have to bind multiple sockets. You are obviously attempting to do this in your code as posted, so I'll assume you do.

> *I was hoping that something like the following would work. Will it? This is on Sparcs running Solaris 2.4/2.5.*

I don't have access to Solaris, but I'll comment based on my experience with other Unixes.

[Shankar's original code omitted]

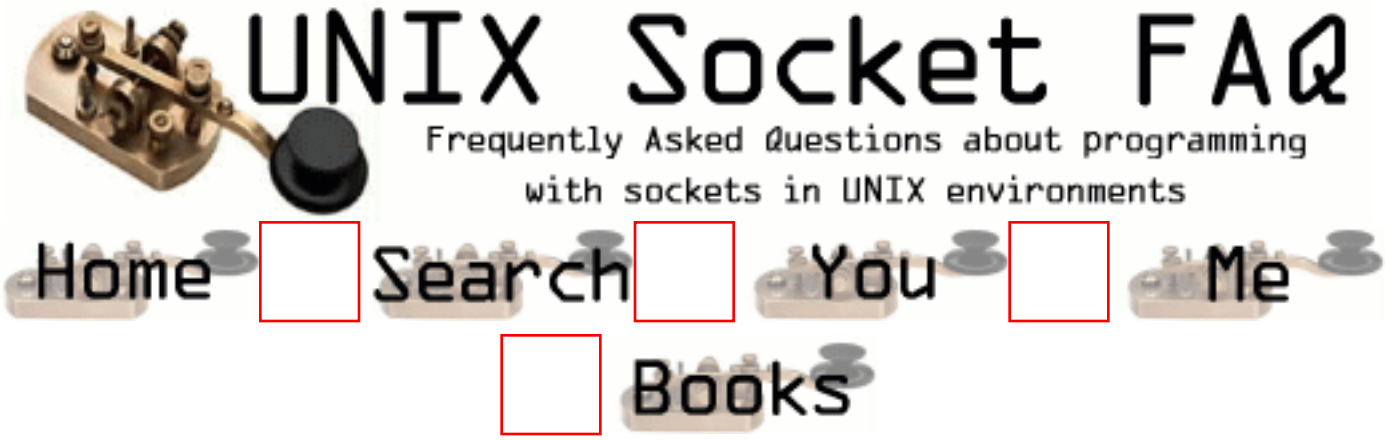
What you are doing is attempting to bind all the current hosts unicast addresses as listed in hosts/NIS/DNS. This may or may not reflect reality, but much more importantly, neglects the broadcast addresses. It seems to be the case in the majority of implementations that a socket bound to a unicast address will *not* see incoming packets with broadcast addresses as their destinations.

The approach I've taken is to use `SIOCGIFCONF` to retrieve the list of active network interfaces, and `SIOCGIFFLAGS` and `SIOCGIFBRDADDR` to identify broadcastable interfaces and get the broadcast addresses. Then I bind to each unicast address, each broadcast address, *and to* `INADDR_ANY` *as well*. That last is necessary to catch packets that are on the wire with `INADDR_BROADCAST` in the destination. (`SO_REUSEADDR` is necessary to bind `INADDR_ANY` as well as the specific addresses.)

This gives me very nearly what I want. The wrinkles are:

- I don't assume that getting a packet through a particular socket necessarily means that it actually arrived on that interface.
- I can't tell anything about which subnet a packet originated on if its destination was INADDR_BROADCAST.
- On some stacks, apparently only those with multicast support, I get duplicate incoming messages on the INADDR_ANY socket.

Add a Comment



Writing Server Applications (TCP/SOCK_STREAM): How can I read only one character at a time?

This question is usually asked by people who are testing their server with telnet, and want it to process their keystrokes one character at a time. The correct technique is to use a psuedo terminal (pty). More on that in a minute.

According to Roger Espel Llima (espel@drakkar.ens.fr), you can have your server send a sequence of control characters: `0xff 0xfb 0x01 0xff 0xfb 0x03 0xff 0xfd 0x0f3`, which translates to `IAC WILL ECHO IAC WILL SUPPRESS-GO-AHEAD IAC DO SUPPRESS-GO-AHEAD`. For more information on what this means, check out std8, std28 and std29. Roger also gave the following tips:

- This code will suppress echo, so you'll have to send the characters the user types back to the client if you want the user to see them.
- Carriage returns will be followed by a null character, so you'll have to expect them.
- If you get a `0xff`, it will be followed by two more characters. These are telnet escapes.

Use of a pty would also be the correct way to execute a child process and pass the i/o to a socket.

I'll add pty stuff to the list of example source I'd like to add to the faq. If someone has some source they'd like to contribute (without copyright) to the faq which demonstrates use of pty's, please email me!

From: lore

Try this: char gimme_char(int fd) {
char * buf = (char *)NULL; buf = (char *)malloc(1);
if (recv(fd, buf, 1, 0)) return ((char)*buf); }

Add a Comment

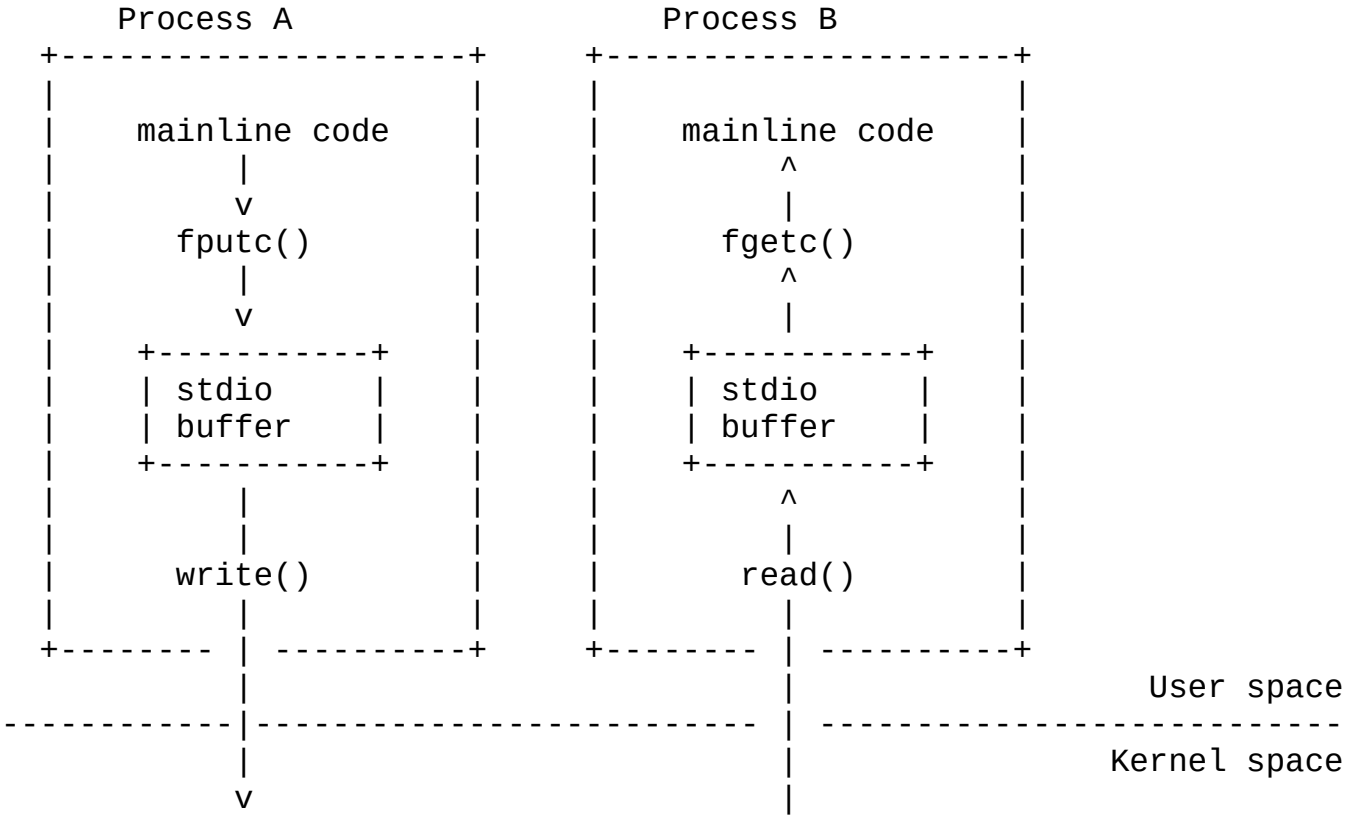
Writing Server Applications (TCP/SOCK_STREAM): I'm trying to exec() a program from my server, and attach my socket's IO to it, but I'm not getting all the data across. Why?

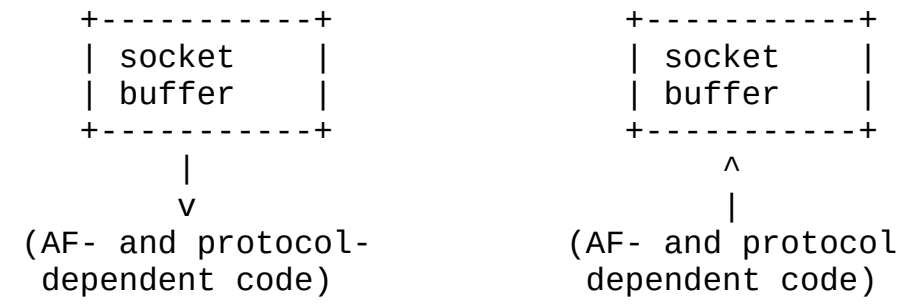
If the program you are running uses `printf()`, etc (streams from `stdio.h`) you have to deal with two buffers. The kernel buffers all socket IO, and this is explained in section 2.11. The second buffer is the one that is causing you grief. This is the stdio buffer, and the problem was well explained by Andrew:

(The short answer to this question is that you want to use a pty rather than a socket; the remainder of this article is an attempt to explain why.)

Firstly, the socket buffer controlled by `setsockopt()` has *absolutly nothing* to do with stdio buffering. Setting it to 1 is guaranteed to be the Wrong Thing(tm).

Perhaps the following diagram might make things a little clearer:





Assuming these two processes are communicating with each other (I've deliberately omitted the actual comms mechanisms, which aren't really relevent), you can see that data written by process A to its stdio buffer is completely inaccessible to process B. Only once the decision is made to flush that buffer to the kernel (via `write()`) can the data actually be delivered to the other process.

The only guaranteed way to affect the buffering within process A is to change the code. However, the default buffering for stdout is controlled by whether the underlying FD refers to a terminal or not; generally, output to terminals is line-buffered, and output to non-terminals (including but not limited to files, pipes, sockets, non-tty devices, etc.) is fully buffered. So the desired effect can usually be achieved by using a pty device; this, for example, is what the 'expect' program does.

Since the stdio buffer (and the `FILE` structure, and everything else related to stdio) is user-level data, it is not preserved across an `exec()` call, hence trying to use `setvbuf()` before the exec is ineffective.

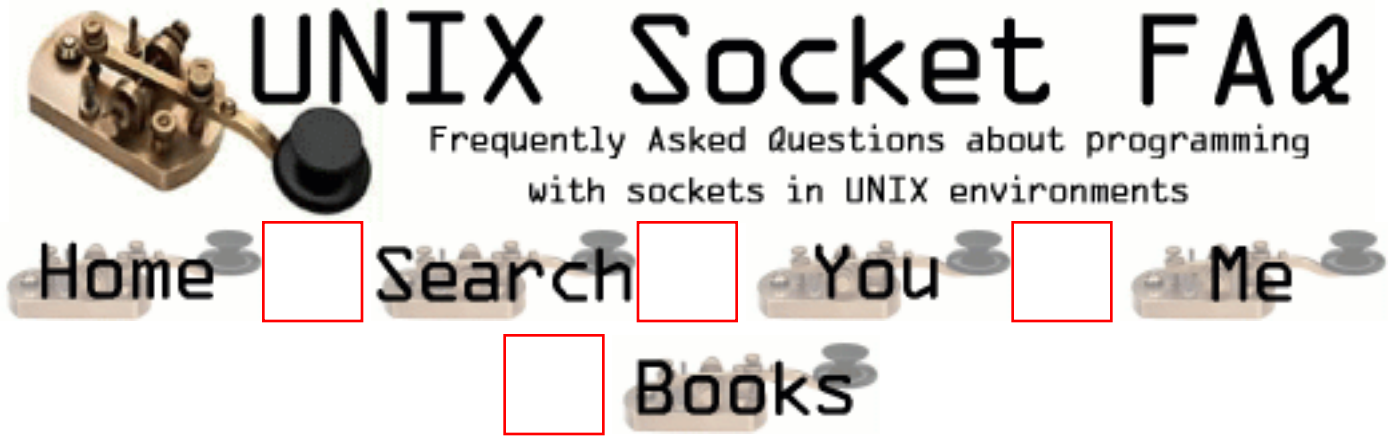
A couple of alternate solutions were proposed by Roger Espel Llima (espel@drakkar.ens.fr):

If it's an option, you can use some standalone program that will just run something inside a pty and buffer its input/output. I've seen a package by the name pty.tar.gz that did that; you could search around for it with archie or AltaVista.

Another option (**_warning, evil hack_**) , if you're on a system that supports this (SunOS, Solaris, Linux ELF do; I don't know about others) is to, on your main program, `putenv()` the name of a shared executable (*.so) in LD_PRELOAD, and then in that .so redefine some commonly used libc function that the program you're exec'ing is known to use early. There you can 'get control' on the running program, and the first time you get it, do a `setbuf(stdout, NULL)` on the program's behalf, and then call the original libc function with a `dlopen()` + `dlsym()`. And you keep the `dlsym()` value on a static var, so you can just call that the following times.

(Editors note: I still haven't done an expample for how to do pty's, but I hope I will be able to do one after I finish the non-blocking example code.)

Add a Comment



Writing UDP/SOCK_DGRAM applications: When should I use UDP instead of TCP?

UDP is good for sending messages from one system to another when the order isn't important and you don't need all of the messages to get to the other machine. This is why I've only used UDP once to write the example code for the faq. Usually TCP is a better solution. It saves you having to write code to ensure that messages make it to the desired destination, or to ensure the message ordering. Keep in mind that every additional line of code you add to your project in another line that could contain a potentially expensive bug.

If you find that TCP is too slow for your needs you may be able to get better performance with UDP so long as you are willing to sacrifice message order and/or reliability.

Philippe Jounin would like to add...

> In chapter 5.1 you say UDP allows more throughput than TCP. It is rarely the case if you have to pass several routers.
>
> For instance, if you connect two LANs via X25 (a common way in Europe!), every UDP datagram will :
>
> - establish a Virtual Channel (VC)
> - send the data
> - close the VC,
>
> whereas the VC remains during a TCP dialog.

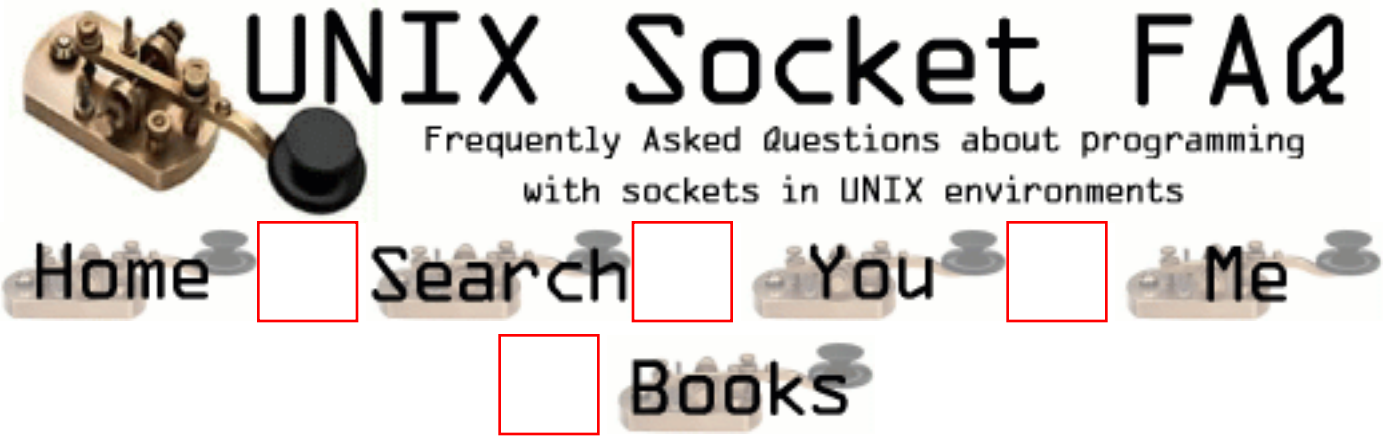
UDP must be used to multicast messages to more than one other machine at the same time. With TCP an application would have to open separate connections to each of the destination machines and send the message once to each target machine. This limits your application to only communicate with machines that it already knows about.

From: Nick Lockyer

Actually UDP and TCP both have the same throughput, since they are both going over the same hardware. However, TCP packets are larger than UDP packets because they have extra fields in

them, thus by definition sending a TCP packet takes longer than a UDP packet because it is bigger. TCP also has a three way handshake (packet goes from A->B, acknowledgement from B->A, then another ACK A->B). This takes a certain amount of time. That is why TCP takes longer than UDP.

Add a Comment



Writing UDP/SOCK_DGRAM applications: What is the difference between "connected" and "unconnected" sockets?

From Andrew Gierth (andrew@erlenstar.demon.co.uk):

If a UDP socket is unconnected, which is the normal state after a `bind()` call, then `send()` or `write()` are not allowed, since no destination address is available; only `sendto()` can be used to send data.

Calling `connect()` on the socket simply records the specified address and port number as being the desired communications partner. That means that `send()` or `write()` are now allowed; they use the destination address and port given on the connect call as the destination of the packet.

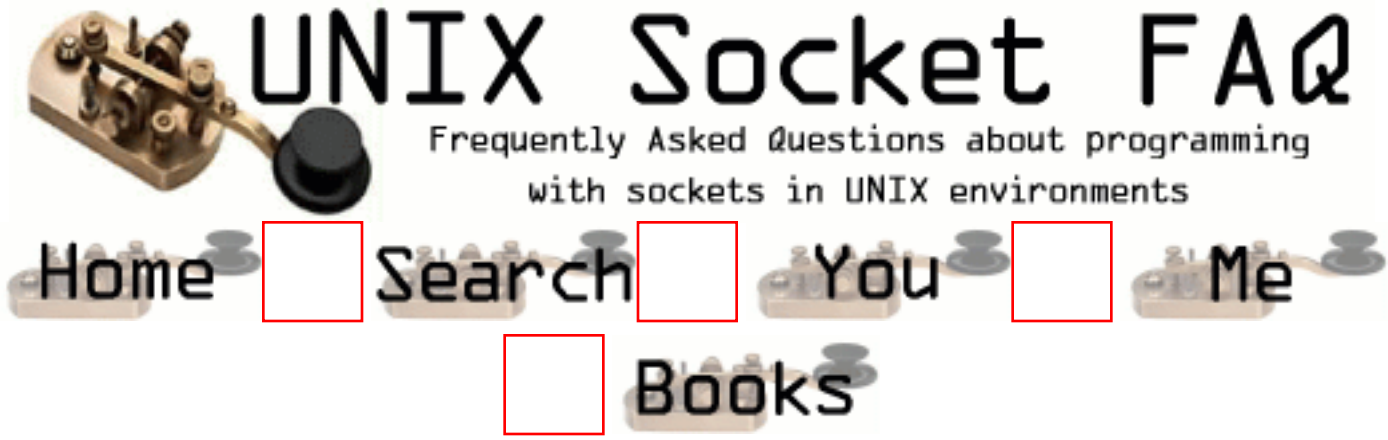
From: Sanjay Pujare

I have another question: When I "connect" a UDP socket to a remote addr and port, is there a corresponding "accept" at the other end for my "connect" to go through? I know that when a TCP socket is connected, the connection happens when the server "accept"s the connection. What is the equivalent thing for UDP sockets?

From: Rob Seace

No, there is no real equivalent to accept() for UDP... UDP
is inherently a connectionless protocol; doing a connect()
on a UDP socket is simply a convenient kluge, to allow doing
write()/send(), as Andrew says above... It merely presets
the remote destination address, so you need not specify it
every time, via sendto()/sendmsg()... But, it doesn't
initiate anything special on the remote end... The remote
doesn't know the difference, one way or the other...

Add a Comment



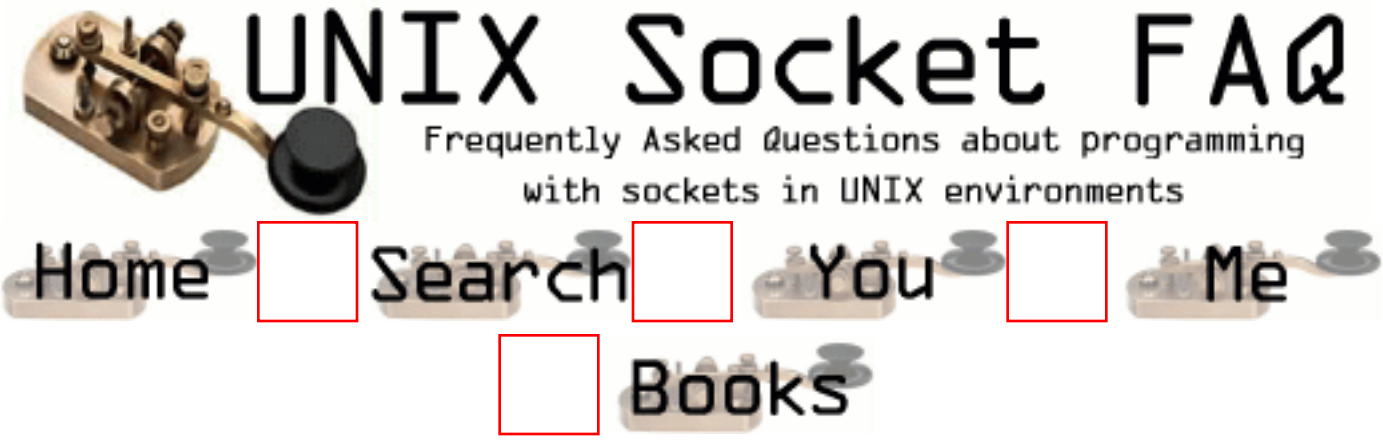
Writing UDP/SOCK_DGRAM applications: Does doing a connect() call affect the receive behaviour of the socket?

From Richard Stevens (rstevens@noao.edu):

Yes, in two ways. First, only datagrams from your "connected peer" are returned. All others arriving at your port are not delivered to you.

But most importantly, a UDP socket must be connected to receive ICMP errors. Pp. 748-749 of "TCP/IP Illustrated, Volume 2" give all the gory details on why this is so.

Add a Comment



Writing UDP/SOCK_DGRAM applications: How can I read ICMP errors from "connected" UDP sockets?

If the target machine discards the message because there is no process reading on the requested port number, it sends an ICMP message to your machine which will cause the next system call on the socket to return `ECONNREFUSED`. Since delivery of ICMP messages is not guarenteed you may not recieve this notification on the first transaction.

Remember that your socket must be "connected" in order to receive the ICMP errors. I've been told, and Alan Cox has verified that Linux will return them on "unconnected" sockets. This may cause porting problems if your application isn't ready for it, so Alan tells me they've added a `SO_BSDCOMPAT` flag which can be set for Linux kernels after 2.0.0.

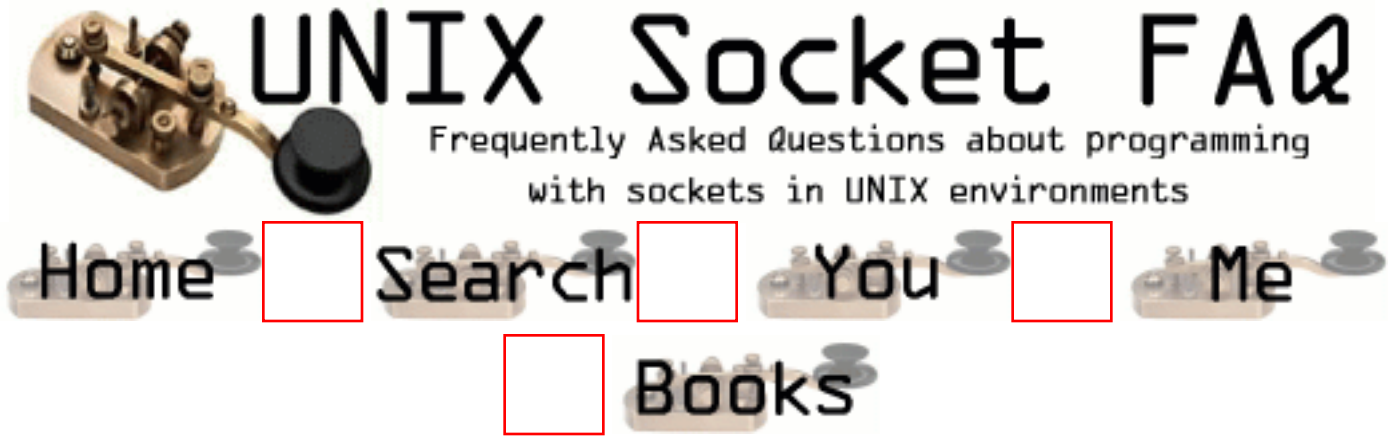
From: meng

I feel very lucky to see this faq. I have been looking for it for a long time.
As you said that "Linux will return them on "unconnected" sockets. This may cause porting problems if your application isn't ready for it". I have this problem.
But I still have some questions on how to use the flag SO_BSDOCMPAT. It is use in server or client? What are other parameters of setsockopt?
Thank you very much.

From: meng

I look for man 7 socket, and get the explaination. The 4nd parameter of system call setsockopt() should be 1,and the 5 should be sizeof it. setsockopt() should be called on the client.
Now it works well.
Best regards!

Add a Comment



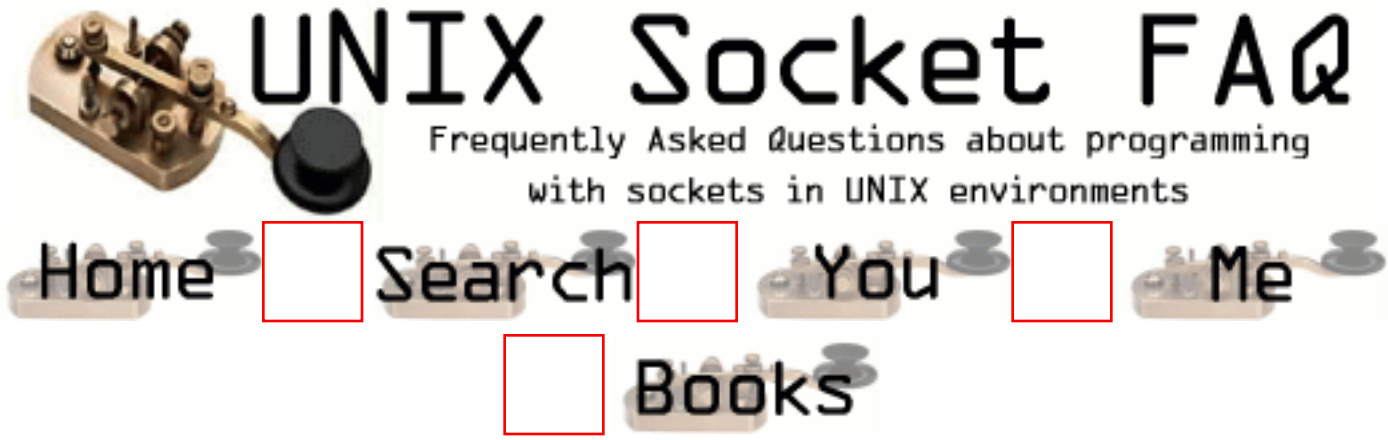
Writing UDP/SOCK_DGRAM applications: How can I be sure that a UDP message is received?

You have to design your protocol to expect a confirmation back from the destination when a message is received. Of course is the confirmation is sent by UDP, then it too is unreliable and may not make it back to the sender. If the sender does not get confirmation back by a certain time, it will have to re-transmit the message, maybe more than once. Now the receiver has a problem because it may have already received the message, so some way of dropping duplicates is required. Most protocols use a message numbering scheme so that the receiver can tell that it has already processed this message and return another confirmation. Confirmations will also have to reference the message number so that the sender can tell which message is being confirmed. Confused? That's why I stick with TCP.

From: Paul Beardsell

You need an acknowledgement from the receiver.

Add a Comment



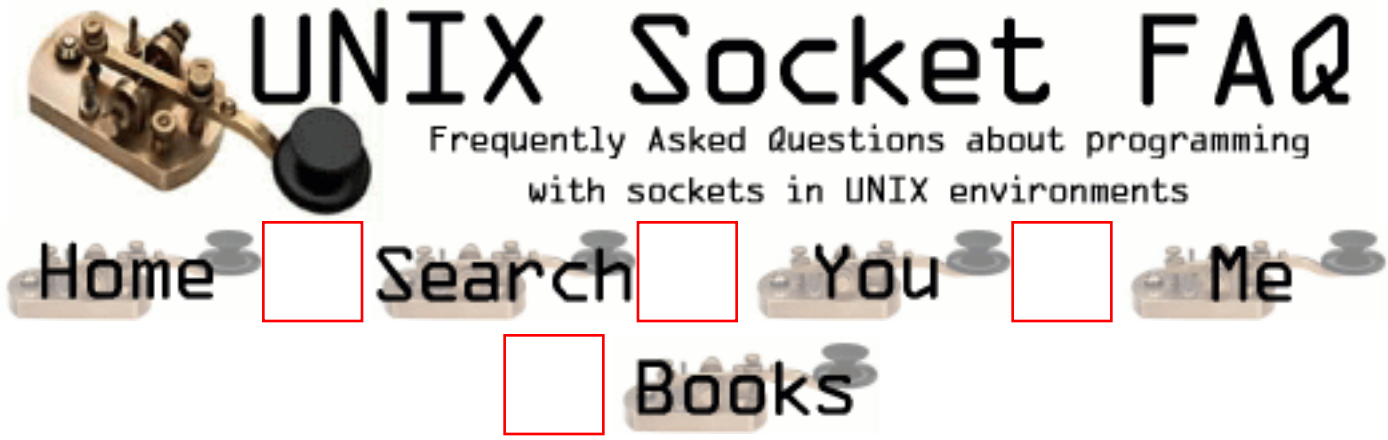
Writing UDP/SOCK_DGRAM applications: How can I be sure that UDP messages are received in order?

You can't. What you can do is make sure that messages are processed in order by using a numbering system as mentioned in 5.5 How can I be sure that a UDP message is received?. If you need your messages to be received and be received in order you should really consider switching to TCP. It is unlikely that you will be able to do a better job implementing this sort of protocol than the TCP people already have, without a significant investment of time.

From: Paul Beardsell

I agree. Go with TCP. But remember
that UDP packet boundaries are
preserved whereas you cannot rely
on this with TCP.

Add a Comment

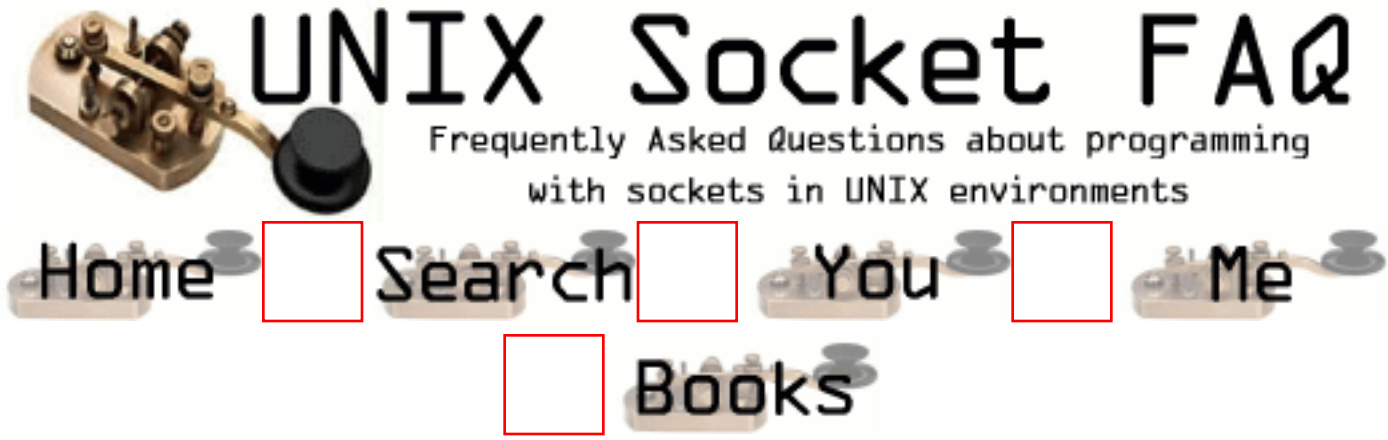


Writing UDP/SOCK_DGRAM applications: How often should I re-transmit un-acknowleged messages?

The simplest thing to do is simply pick a fairly small delay such as one second and stick with it. The problem is that this can congest your network with useless traffic if there is a problem on the lan or on the other machine, and this added traffic may only serve to make the problem worse.

A better technique, described with source code in "UNIX Network Programming" by Richard Stevens (see 1.6 Where can I get source code for the book [book title]?), is to use an adaptive timeout with an exponential backoff. This technique keeps statistical information on the time it is taking messages to reach a host and adjusts timeout values accordingly. It also doubles the timeout each time it is reached as to not flood the network with useless datagrams. Richard has been kind enough to post the source code for the book on the web. Check out his home page at http://www.kohala.com/~rstevens.

Add a Comment



Writing UDP/SOCK_DGRAM applications: How come only the first part of my datagram is getting through?

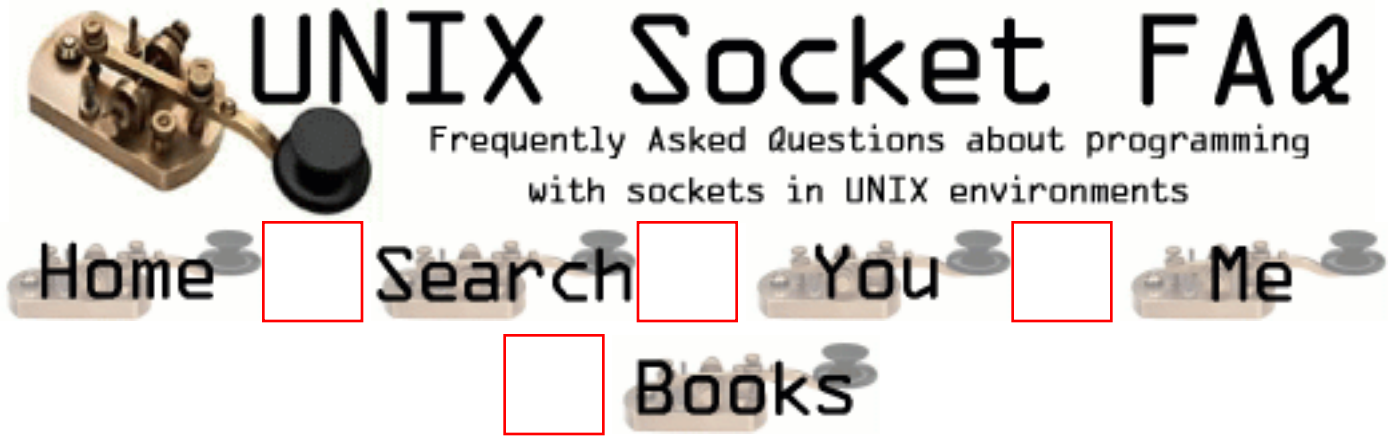
This has to do with the maximum size of a datagram on the two machines involved. This depends on the sytems involved, and the MTU (Maximum Transmission Unit). According to "UNIX Network Programming", all TCP/IP implementations must support a minimum IP datagram size of 576 bytes, regardless of the MTU. Assuming a 20 byte IP header and 8 byte UDP header, this leaves 548 bytes as a safe maximum size for UDP messages. The maximum size is 65516 bytes. Some platforms support IP fragmentation which will allow datagrams to be broken up (because of MTU values) and then re-assembled on the other end, but not all implementations support this.

This information is taken from my reading of "UNIX Netowrk Programming" (see 1.6 Where can I get source code for the book [book title]?).

Andrew has pointed out the following regarding large UDP messages:

Another issue is fragmentation. If a datagram is sent which is too large for the network interface it is sent through, then the sending host will fragment it into smaller packets which are reassembled by the receiving host. Also, if there are intervening routers, then they may *also* need to fragment the packet(s), which greatly increases the chances of losing one or more fragments (which causes the entire datagram to be dropped). Thus, large UDP datagrams should be avoided for applications that are likely to operate over routed nets or the Internet proper.

Add a Comment



Writing UDP/SOCK_DGRAM applications: Why does the socket's buffer fill up sooner than expected?

From Paul W. Nelson (nelson@thursby.com):

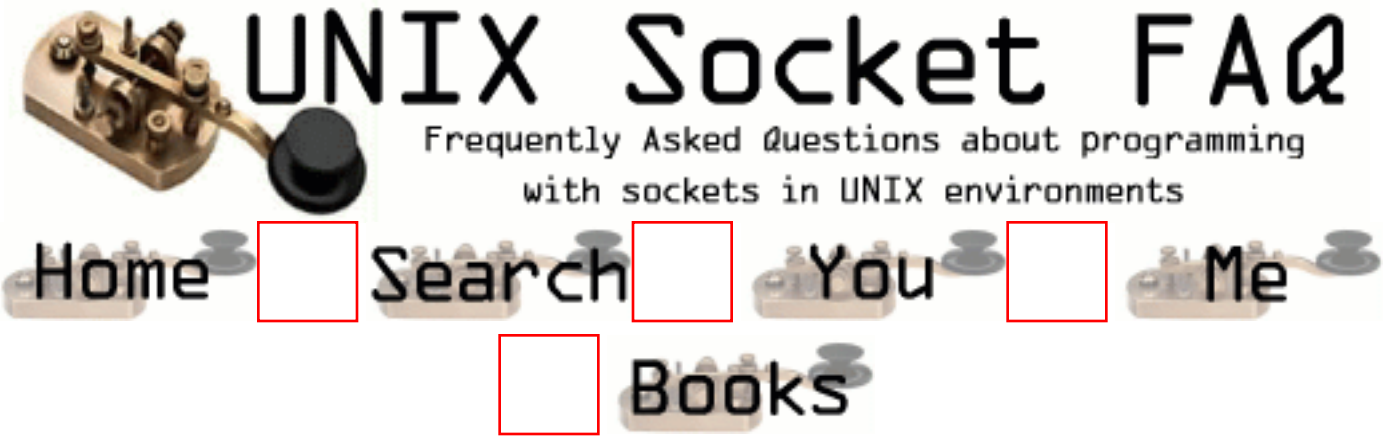
In the traditional BSD socket implementation, sockets that are atomic such as UDP keep received data in lists of mbufs. An mbuf is a fixed size buffer that is shared by various protocol stacks. When you set your receive buffer size, the protocol stack keeps track of how many bytes of mbuf space are on the receive buffer, not the number of actual bytes. This approach is used because the resource you are controlling is really how many mbufs are used, not how many bytes are being held in the socket buffer. (A socket buffer isn't really a buffer in the traditional sense, but a list of mbufs).

For example: Lets assume your UNIX has a small mbuf size of 256 bytes. If your receive socket buffer is set to 4096, you can fit 16 mbufs on the socket buffer. If you receive 16 UDP packets that are 10 bytes each, your socket buffer is full, and you have 160 bytes of data. If you receive 16 UDP packets that are 200 bytes each, your socket buffer is also full, but contains 3200 bytes of data. `FIONREAD` returns the total number of bytes, not the number of messages or bytes of mbufs. Because of this, it is not a good indicator of how full your receive buffer is.

Additionaly, if you receive UDP messages that are 260 bytes, you use up two mbufs, and can only recieve 8 packets before your socket buffer is full. In this case, only 2080 bytes of the 4096 are held in the socket buffer.

This example is greatly simplified, and the real socket buffer algorithm also takes into account some other parameters. Note that some older socket implementations use a 128 byte mbuf.

Add a Comment



Advanced Socket Programming: How would I put my socket in non-blocking mode?

From Andrew Gierth (andrew@erlenstar.demon.co.uk):

Technically, `fcntl(soc, F_SETFL, O_NONBLOCK)` is incorrect since it clobbers all other file flags. Generally one gets away with it since the other flags (`O_APPEND` for example) don't really apply much to sockets. In a similarly rough vein, you would use `fcntl(soc, F_SETFL, 0)` to go back to blocking mode.

To do it right, use `F_GETFL` to get the current flags, set or clear the `O_NONBLOCK` flag, then use `F_SETFL` to set the flags.

And yes, the flag can be changed either way at will.

From: jagadeesh

```
            /* set socket to non-blocking i/o */
   sts = ioctl(ccp->main_sock, FIONBIO, (char *)&one);
   if (sts)
   {
      setproderr(PE_TCPERROR, GEL_FATAL);
      sprintf(line,"ioctl (main) failed - %s",strerror(errno));
tcpabort();
   }
```

From: Viswaroopan

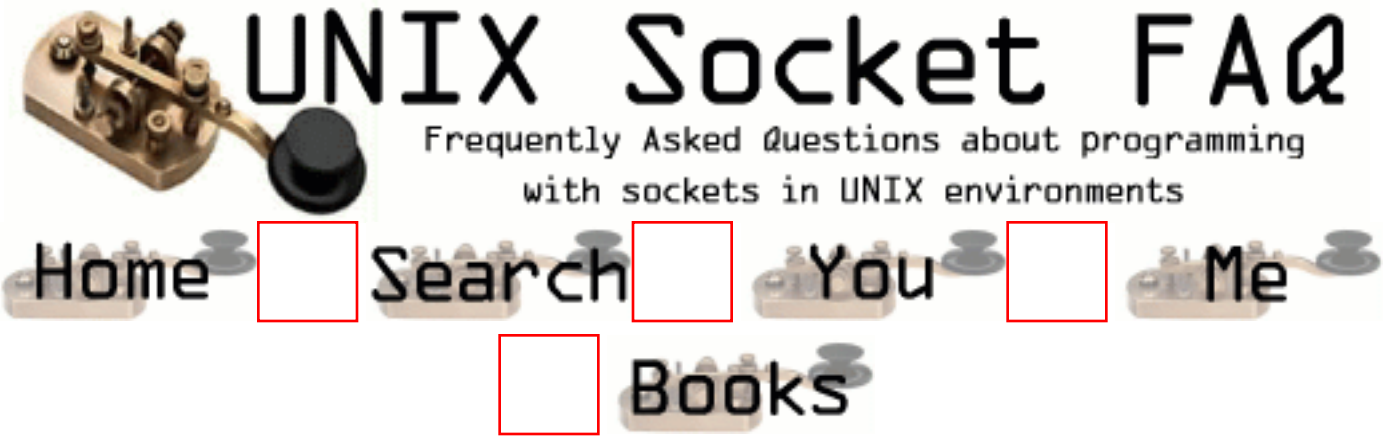
Hi,
I tried this and works great as non-blocking socket. I have a different problem is that, when I made it as non-blocking the accept() on the server comes out immediately with non-block error. Instead I want accept() to wait for some time (set a timeout) before giving that error. Is there any way I can set the timeout on accept().

Thanks in advance.
Vish

From: Jonathan Rynd

This is normal for all socket nonblocking operations: if you call them, you should be prepared to handle 2 cases: 1, they succeed right away, 2, they 'fail' with the "EWOULDBLOCK" non-blocking error (it's not a real failure, it just means "we can't satisfy that right now'. You then have to create a FD_SET structure and use it as input to select() with the proper timeout. See the manpage for select. Depending on the call, when select() returns to indicate success, you may need to make the call again.

Add a Comment



Advanced Socket Programming: How can I put a timeout on connect()?

Andrew Gierth (andrew@erlenstar.demon.co.uk) has outlined the following procedure for using `select()` with `connect()`, which will allow you to put a timeout on the `connect()` call:

First, create the socket and put it into non-blocking mode, then call `connect()`. There are three possibilities:

- connect succeeds: the connection has been successfully made (this usually only happens when connecting to the same machine)
- connect fails: obvious
- connect returns -1/`EINPROGRESS`. The connection attempt has begun, but not yet completed.

If the connection succeeds:

- the socket will `select()` as writable (and will also select as readable if data arrives)

If the connection fails:

- the socket will select as readable *and* writable, but either a read or write will return the error code from the connection attempt. Also, you can use `getsockopt(SO_ERROR)` to get the error status - but be careful; some systems return the error code in the result parameter of `getsockopt()`, but others (incorrectly) cause the `getsockopt` call *itself* to fail with the stored value as the error.

Sample code that illustrates this can be found in the file `<http://www.lcg.org/sock-faq/connect.c>`.

From: warren

That is great, but how do you actually implement the timeout?

From: Vic Metcalfe

In the above example (given as a link at the bottom) you would add the timeout to the select() call in main().

From: Stan Driggs

Another common solution is to use an alarm to interrupt the connection. Here is some example code:

```
static int sTimeout = 0;

static void AlarmHandler(int sig)
{
  sTimeout = 1;
}


.
.
.
   signal(SIGALRM, AlarmHandler);
   sTimeout = 0;
   alarm(CONNECT_TIMEOUT);

   if ( connect(sock, (struct sockaddr *) &server, sizeof(server)) )
   {
      if ( sTimeout )
         perror("timeout connecting stream socket");
      else
         perror("connecting stream socket");
      exit(1);
   }

   sTimeout = 0;
   alarm(CONNECT_TIMEOUT);
.
.
.
```

From: Mark Papadakis

The alarm way is not really useful if you are writting a threaded application...
You should therefore rely on the select() solution.

Mark

From: Tan Nguyen

True. In a threaded application, alarm() is not helpful at all. However, how can we use select()

to interrupt connect() if connect() never returns? (ie trying to connect to some non-existent host)

From: Tan D Nguyen

Ignore my last post. Another thing though, it seems that when we set the socket to non-block mode, we cannot read the stream. I tested it by adding something to the bottom of the select() example. recv() always returns -1. Am I missing something?

```
if (rc)
   fprintf(stderr,"connect failed - error %d (%s)\n",rc,strerror(rc));
else
  {
   int sent = 0;
   int received = 0;
   fprintf(stderr,"connect successful\n");
   sent = send(sock, "GET / HTTP/1.0 \r\n\r\n", 19, 0);
   printf("Bytes sent : %d\n", sent);
   char buf[1024];
   received = recv(sock, buf, 1023, 0);
   printf("Bytes received : %d\n", received);
   buf[1024] = '\0';
   if (received > 0)
      printf("Recieved some data\n");
  }

close(sock);
return 0;
```

From: jsh

Use setsockoption system call to set the socket to timout state.

From: andrew

Linux socket timeouts can't be amended using setsockopt.
The timeouts are fixed on a per protocol basis.

The man 7 socket page reccommends using alarm ....

From: Hector Lasso

Tan,
If the socket is still in non-blocking mode then your recv() will return -1 and errno will be EAGAIN unless there is data available at this time.

Add a Comment

UNIX Socket FAQ

Frequently Asked Questions about programming with sockets in UNIX environments

Advanced Socket Programming: How do I complete a read if I've only read the first part of something, without again calling select()?

From: Ilya

I am looking for some recommendation to handle situation as follows: select() says data is available to be read. I read 4096 bytes(this is size for socket receive buffer), parse buffer and see that there are 25 complete messages in buffer and part of the next message. I would like to read rest of uncompleted message without call select(). I would like call select only after I finished read rest of uncompleted message. Thanks

---

From: Tom

In answering Ilya's question, I would use the following approach to read message one by one (in this case, the messages are strings separate by NULL characters):

```
while (1)
{
    bzero(buffer, sizeof(buffer));

    /* PEEK but not read from receive buffer first */
    readcount = recv(socketfd, buffer,
                     sizeof(buffer), MSG_PEEK);
    if (readcount <= 0)
        break;

    recv(socketfd, buffer, strlen(buffer)+1, 0);
    fprintf(stdout, "%s\n", buffer);
}
```

From: Ilya

Thank yoy very much !

From: Ilya

Thank yoy very much !

From: alexandre

Hi,

i use part of your function, to get reliable data transfert over TCP,
here is my func:

call select
if readable continue;
if not return

recv (MSG_PEEK)
does buffer has \r\n
if yes continue
if no return

recv
return len buffer

the problem is that recv, doesn"t change file descriptor status
so select think data is avaible. Is there a function:

similar to get in the way that select doens't see it again
or
get select to return only if new data is available

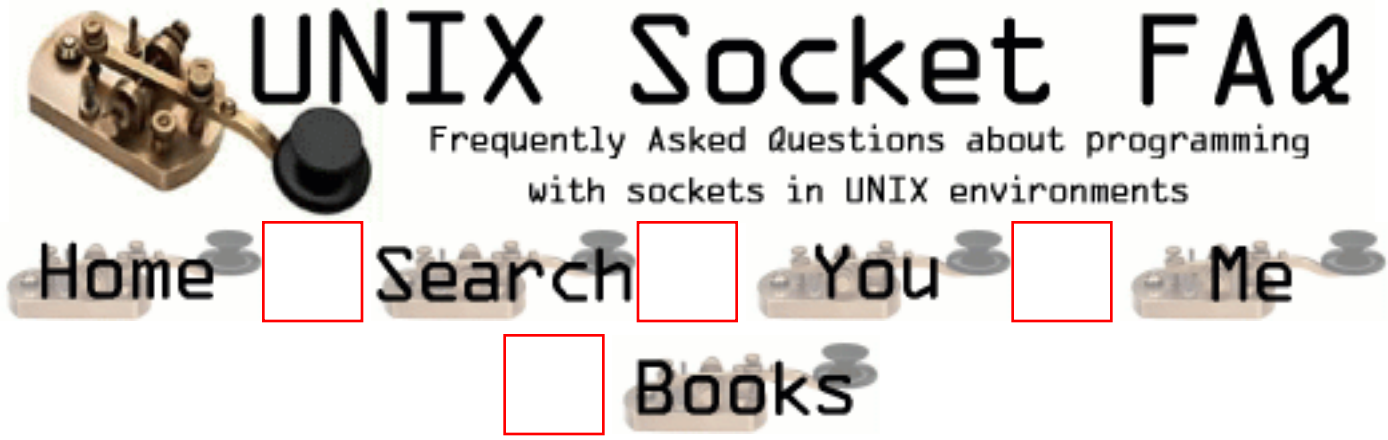
Cheers

Alexandre

From: Bob

Replace MSG_PEEK by 0.
this should chage the fd status

Add a Comment



Advanced Socket Programming: How to use select routine

From: Starch Melo de Souza

I need to use select routine for receive several response at the same time. How to write the code??? Thanks.

From: Michael Song

```
int sock_ready(sock)
int sock;
{
   int            res;
   fd_set         sready;
   struct timeval  nowait;

   FD_ZERO(&sready);
   FD_SET((unsigned int)sock,&sready);
   /*bzero((char *)&nowait,sizeof(nowait));*/
   memset((char *)&nowait,0,sizeof(nowait));

   res = select(sock+1,&sready,NULL,NULL,&nowait);
   if( FD_ISSET(sock,&sready) )
      res = 1;
   else
      res = 0;

   return(res);
}
```

From: S.Murali Krishna

Explanation of above select code.

Q: How to use select call for multiplexing.

Ans: First you are declaring a fd_set variables.
so that this can be used in select call.
then depending on the (read/write) on socket you are using
Add the socket to the corresponding fd_sets using FD_SET()
call. Before that you have to clear any garbage value there
using FD_ZERO(). Then Select call requires timeout ie. maximum
time to wait for I/O arrival on fds. We are going to wait indefinetly
so we make the timeval structure ZERO by making any calls
specified in the comments. Call the select call with
First Argument as 1 greater than the greater numbered file
descriptor in the second , third or fourth argument.
since we are going to read the socket place it in the
second argument fd_set (as already we done). and give the
timeval structure as Last argument to select so select will
wait indefinetly on the socket and if any I/O comes
it will return with success value.
Then We have to check for the Existense of the socket descriptor
in the readfds fd_set using FD_ISSET() macro. if it returns
true then we can continue any further reading with that socket.
Thanks..

From: Dennis Fleurbaaij

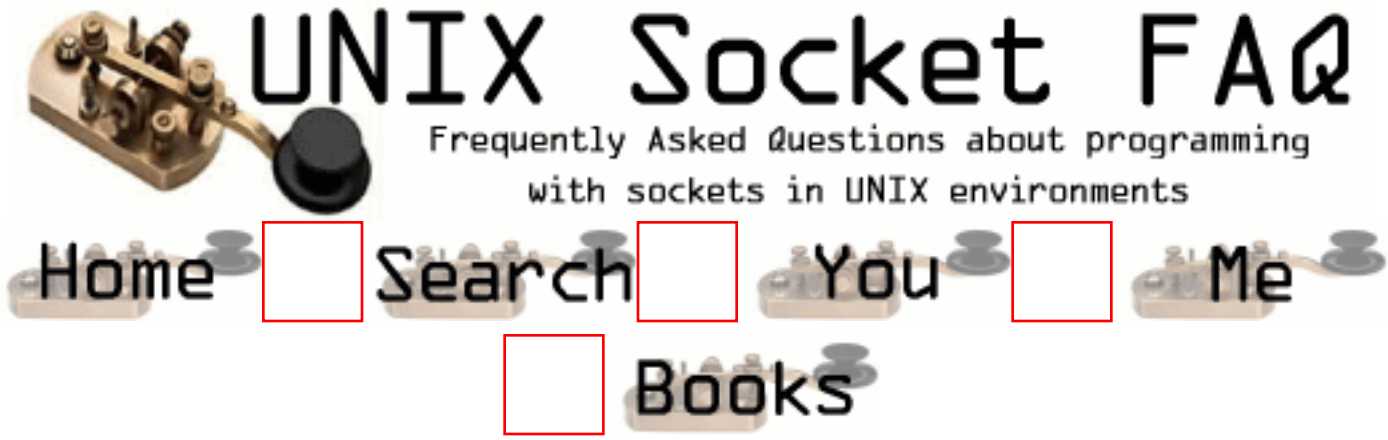
mmm there are some things in the code that are
quite err.. unforunate.

First res is set to the output of the select() call
and after that it's overwritten. Youb better do a
bit of errorchecking and if not, you can
leave out res all together and return(n)

From: legend hacker

Um, res is an int, not a pointer, it wont be overwritten, cos its copied
to the calling function by value, and theres no n. Why else is the
code unfortunate?

Add a Comment



Advanced Socket Programming: RAW sockets

From: Steve McDonald

How would a socket using the SOCK_RAW protocol be used, and what, if any, advantages would it have over SOCK_DGRAM or SOCK_STREAM protocols?

From: Stanislav Shalunov

There is a separate Raw IP Networking FAQ regularly posted to comp.unix.programmer. As any Usenet FAQ it can be downloaded from rtfm.mit.edu. In short: you can do interesting things with the network using raw sockets (e.g., write arbitrary data rather than properly formatted TCP data or UDP packets).

From: HariKumar B

I could go to the site ftp://rtfm.mit.edu/
but how can I get the particular location from which I can download the ino pertaining to RAW SOCKETS.
Some list is specified here.
Please make it more clear!

From: [v0rt]

To those who cant be bothered searching the ftp for the file location, its available here
ftp://rtfm.mit.edu/pub/usenet-by-group/comp.unix.programmer/Raw_IP_Networking_FAQ

[v0rt]
v0rt@deadprotocol
http://security.dayrom.com.au

From: [v0rt]

To those who cant be bothered searching the ftp for the file location, its available here
ftp://rtfm.mit.edu/pub/usenet-by-group/comp.unix.programmer/Raw_IP_Networking_FAQ

[v0rt]
v0rt@deadprotocol.org
http://security.dayrom.com.au

From: Bret Watson

Better still - the master site is at
http://www.whitefang.com/rin/

Cheers,

Bret Watson
http://www.ticm.com

From: khodadad nezhadkorki

hello every body !
what is the difference between udp,tcp and raw sockets ?
please send me your idea .
thank you

From: G®ïm Rëåpër

I'm looking for a script in c for creating a spoofed udp packet. It must allow me to specify the destination address as well as the destinatin port. it also must allow me to specify a source address and port. The last thing is that I must be able to send a message in plain text w/ it.
I know nothing about c so if you know where i can get a script, please let me know.

From: Nullzilla

C is *not* an scripting language.

From: rawkid

What else do I need to do other than what I already have done in this code?
I am doing this on vxWorks.

```
#define IPPROTO_TEST 88
#define BUFSIZE 1500
char buf[BUFSIZE];
char rcvbuf[BUFSIZE];
int s;
int nSize;

//On client end
main(char *toName)
{
  struct sockaddr_in toAddr;
```

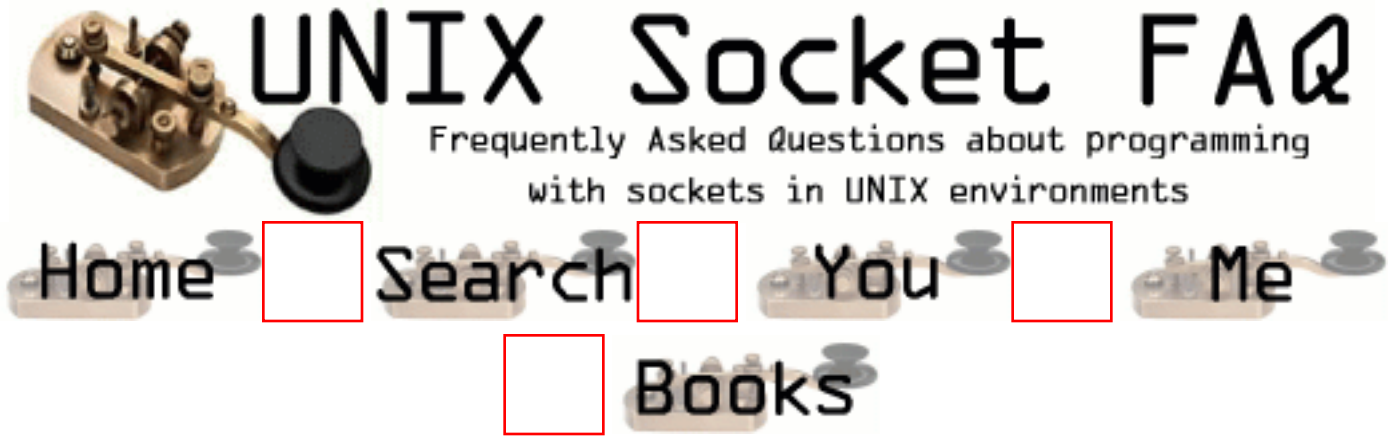
```
  int n;

  nSize = sizeof( struct sockaddr_in);
  s = socket(AF_INET, SOCK_RAW, IPPROTO_TEST);

  //On client end
  n = sendto(s, (caddr_t) &buf, sizeof(buf), 0,
            (struct sockaddr *) &toAddr, nSize)
  if (n < 0)
  {
    perror("sendto");
    return(n);
  }
}

//On server end
main()
{
  struct sockaddr_in frAddr;
  int n;

  nSize = sizeof( struct sockaddr_in);
  s = socket(AF_INET, SOCK_RAW, IPPROTO_TEST);


  n = rcvfrom(s, rcvbuf, sizeof(rcvbuf), 0,
            (struct sockaddr *) &frAddr, nSize);
  if(n < 0)
  {
    perror("rcvfrom");
    return(n);
  }
}
```

From: Harshit Kumar

Raw Sockets are used to send Packets of protocols like ICMP,IGMP, which of course are not TCP(SOCK STREAM) or UDP (DGRAM). With Raw sockets you can spoof IP addresses as u can control Ip headers.

Just to add to it.... For the first time Microsoft in its
windows XP implements raw sockets and confirms to Berkeley socket implementation.

Add a Comment



Advanced Socket Programming: Restricting a socket to a given interface

From: Georg Wagner

How do I restrict a socket to a specific inteace i.e. that it only listens and accepts from the given interface ?

From: Bret Watson

Difficult.. Unless you are going to patch the kernel I don't think this is possible.

What I have seen in most firewall implementations is that the "inetd" wrapper does some filtering up front, including:

Check if the src and dest belong to the same network - if so ignore packet.
Make filtering decisions based on src,dest, ports..

Bret

From: Bret Watson

As a second check I dug through the source of the juniper firewall - which does identify interfaces as trusted or non-trusted

I found there was a file called "Kernel_patch" which patched the kernel so that the socket identified which interface it was on.

So there you go - you will need to patch things within netinet unless you are running a version of linux that supports such things.

Bret

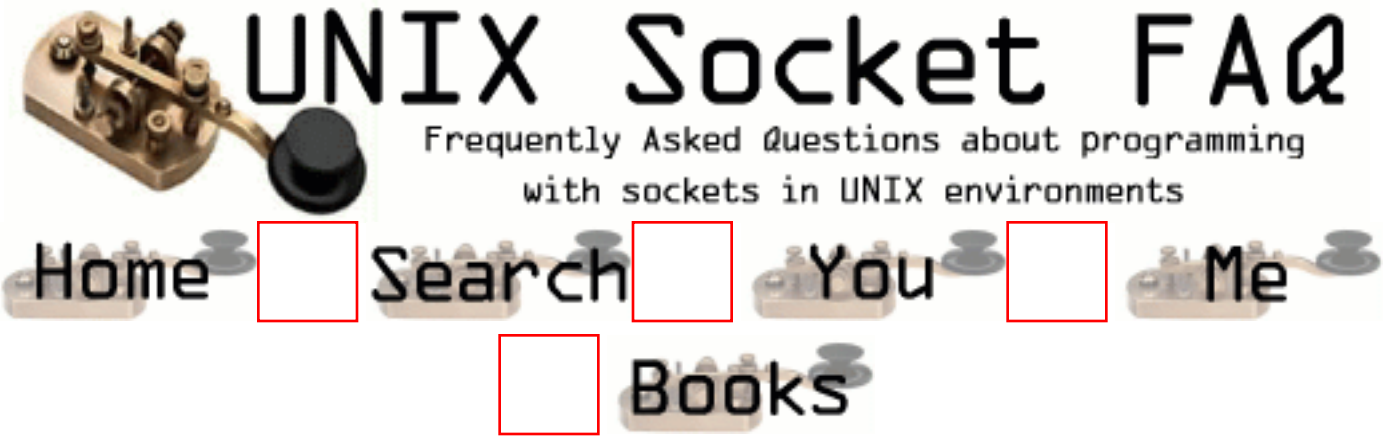
From: John

If you can re-compile the server that you want to
restrict, then it's easy. Set the IP address in the parameter you pass to bind() to be the IP address of
the network interface you want to listen on (assuming
you know what that is or course...)

From: meng

try use function setsockopt(),and use SOL_SOCKET, SO_DONTROUTE, peramater. It may work if your kernl
suport it.

Add a Comment



Advanced Socket Programming: Receiving all incoming traffic through a RAW-socket?

From: Juhana Lehtiniemi

Is it possible to receive all the incoming packets ignoring target port, source address/port and seq/ack-numbers, through a raw-socket?

For example, if I want to create a connection with a server by receiving packets myself and by replying to seqs and acks by writing all the packets myself. So is this possible with a raw-socket, what options should I set to that socket or do I have to read the packets from the network device? I don't want to use packet capture libs like libpcap. What is the easiest way to capture all the incoming packets?

From: Thamer Al-Herbish

Hi,

Your question is answered in the Raw IP Networking FAQ:

http://www.whitefang.com/rin/

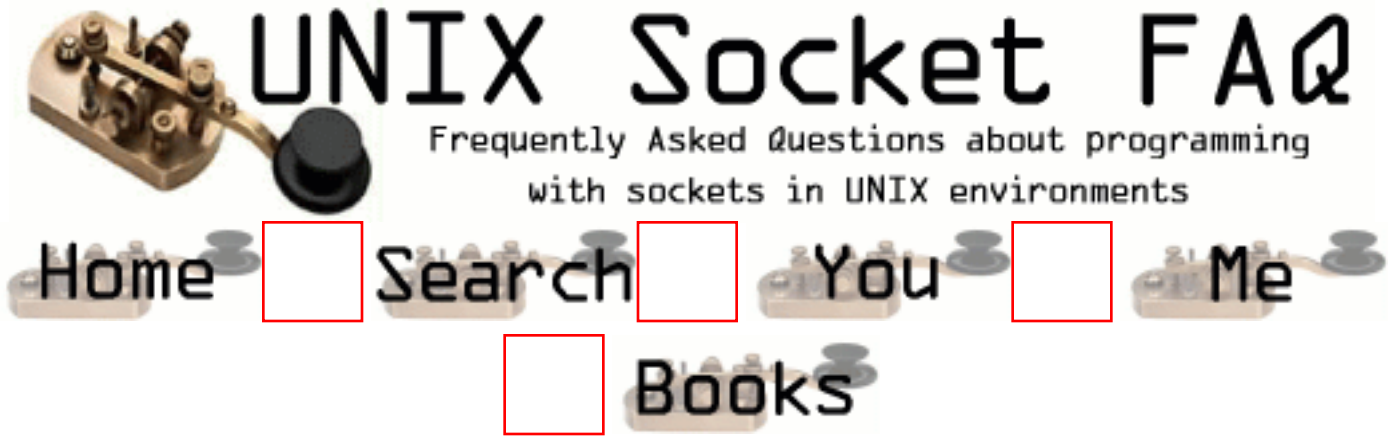
In short, you can't do it. You need to use a packet capturing device.

From: touchene

what is the way to take for founding more documentation for the broacasting whith socket_raw

---

Add a Comment



Advanced Socket Programming: Multicasting

From: Robert G. Brown

These online resources are fabulous -- one can find nearly any
aspect of socket programming discussed complete with sample code.

One thing doesn't seem to be discussed in detail. How does one
write a multicasting application? Specifically, does anyone
have code for a simple multicasting application that I could
clone and incorporate into an information daemon I've written?

From: Senthil

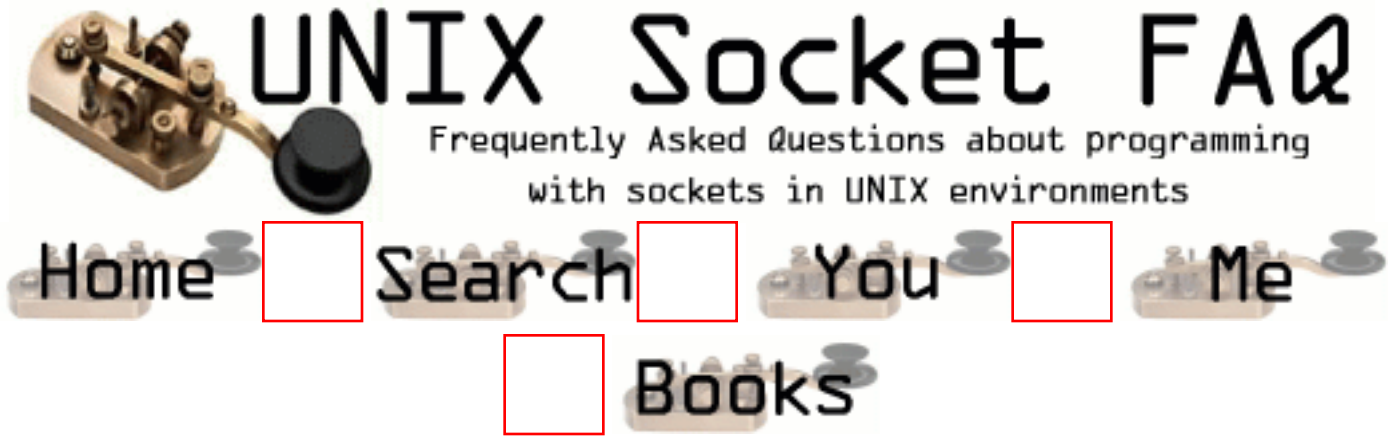
U can have a list of client(socket ids) subscribed to ur information daemon.
when u want to netcast an info, u go thru the list, find out the active clients(using select)
and write to their sockets.

From: tito

.... but wouldn't that be REbroadcasting and not multicasting...

How can I send out one packet and have it read by all of my subscribing clients? Why congest
the network with redundant information?

Add a Comment



Advanced Socket Programming: getting IP header of a UDP message

From: Dmitri

I'm stuck trying to figure out how to read the IP header of each UDP packet I get on a normal SOCK_DGRAM socket. The field of interest is the TTL field from the IP header. Of course using RAW sockets would help, but they require root access on most Unix boxes.

Thanks

From: E McWhorter

You don't ordinarily have access to IP packets at the application layer. If you must have network data at this level, the interface is OS-dependant. On a berkeley-like box, you'll use Berkeley Packet Filters, which must be installed by the sys admin. On a sys V box, you'll use a STREAMS filter, which also must be installed by the sys admin. These devices are not generally installed for security reasons.

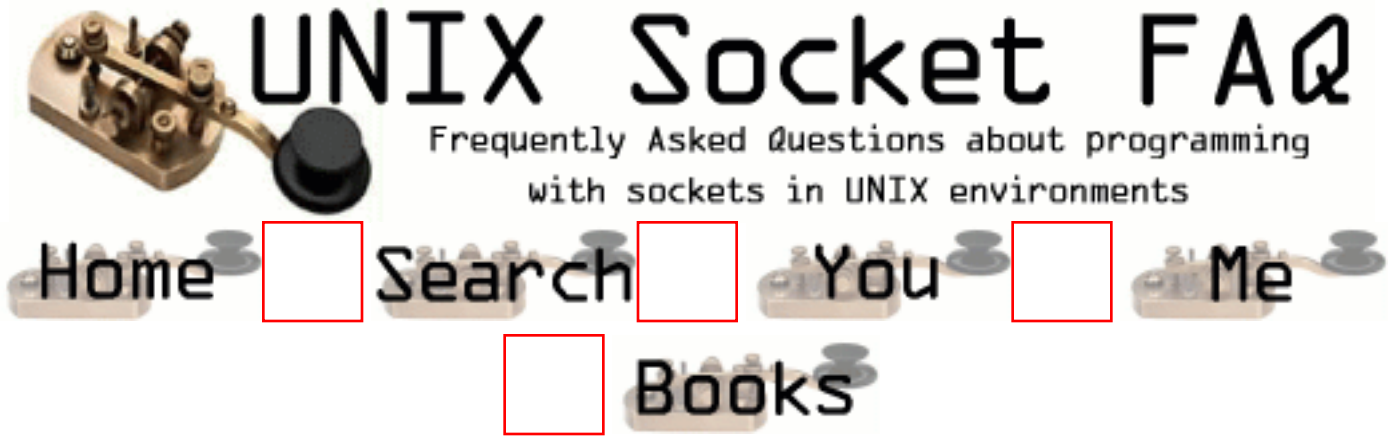
From: Bruce M. Simpson

Yes, you'll need root or sufficient privilege for this.
On new Linux/BSD boxes I configure, I patch the kernel to use a group privilege model for raw socketness, to save really bad things happening.
libnet and libpcap are your friends here. Check them out.

From: lore

In Linux, you can use SOCK_PACKET to receive IP headers upwards, (or even ethernet headers if you ioctl the socket and set promiscuous mode), but this requires root access.

Add a Comment



Advanced Socket Programming: To fork or not to fork?

From: Chris Briggs

I'm relatively new to socket programming, most of my socket
coding knowledge stems from working with tcp/ip socket
wrappers of extreme abstraction (ie: delphi apps, etc).

I'm in the process of writing a server for a middle-tier c/s
suite. I'm writing it using BSD-style sockets, under freebsd.
When handling incoming client connections, is it best to fork()
a process for each client? or would this result in too much
overhead. I'm trying to write a system that acts almost identically
to ircd, however it won't be used for chatting, etc. I have noticed
how daemons such as ftpd and apache fork ps's for each (or groups of clients)
I also noticed that in my skeletal, barebones server: each process
forked takes up ~300K. i can only image what would happen if
i have 3,000 clients connected. I like the way ircd is written
in that you are able to load balance between servers while still
allowing all servers/clients to "see each other". Any suggestions
on whether forking is what i need? Thank you in Advance..

-Chris Briggs

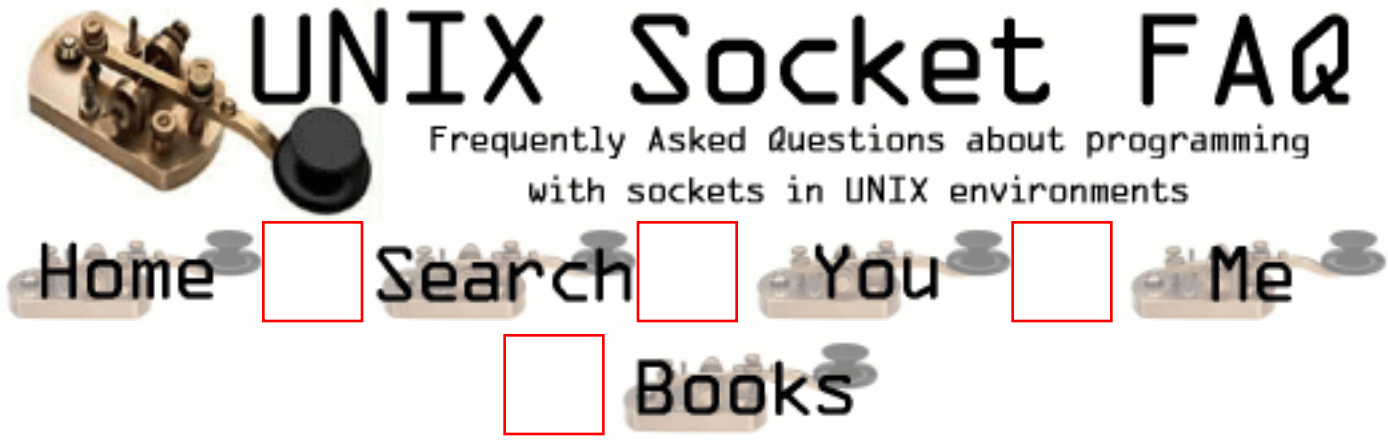
From: senthil

fork() will create a separate process for each of the clients.
This will eat lot of memory.Best thing for this type architecture is to use
Multithreading.
Refer pthreads for more info.

From: Garen Parham

I think it's important to note that probably the most important reason that ircd uses I/O multiplexing ala select() or poll(), is because the file descriptors need to interact. The traditional model of blocking I/O on a listening socket and then immediately fork()ing a process for each new client also is not always bad. Indeed, the memory required for each new process is overhead, but it is fairly constant. It does become a problem though in the case where you have an extraordinary amount of processes that the kernel needs to service, and if this amount approaches the memory limitations of the host machine. Which is likely why programs such as apache have a timeout of typically 15 seconds of inactivity before the process closes. To handle a whole lot of clients (like 3000), you'll probably want to use I/O multiplexing.

Add a Comment



Sample Source Code: Looking for a good C++ socket library

From: Russ Fink

What good freeware C++ class libraries for sockets are out there? I've seen a couple, but they are all several years old, which makes me wonder if anyone is keeping them up anymore. What good options are available to me under C++ that are low-cost? Thanks in advance -- Russ

---

From: Byron Harris

You may want to look at The ADAPTIVE Communications Environment (ACE), which can be downloaded at http://www.cs.wustl.edu/~schmidt/ACE.html. It includes a number of C++ classes to aid developing extensible and portable communications code.
Socket access is part of ACE.

From:

fufu

From: fdsaf

ewrgregfdsgfdg

From: fdsaf

ewrgregfdsgfdg

From: Dema_Lamer

without problems =) ...
see licq sourses ... (classes INetSocket, TCPSocket, UDPSocket)

www.licq.org

From: Shivakumar

I wanted to develop sockets between HP UX and Sco unix machines in C++

From: cool

coolcoolcoolcoolcoolcool

From: Lonnie Cumberland

Take a look as the SAL "System Abstraction Layer"
libraries out at:

http://www.ispras.ru/~knizhnik/

I think that they are VERY good and seem to work on
different platforms also.

From: Norbert Lennartz

I found the lib common c++.

It seems very good, but i have problems with it.
I am not sure to use this.

From: Stephen Silvey

Unsure if this will help, but libnet has a good C (Not C++) Package for doing packets.

Available at this location:
---
http://www.packetfactory.net/Projects/libnet/

Read an article about it at:
---
http://www.securityfocus.com/focus/linux/articles/libnet101.html

---

From: Arindam Dey

Why do people like cool post such useless messages and waste there own and other peoples time.It is disgusting.

From: Stephen Silvey

http://users.erols.com/astronaut/ssl/

From: Stephen Silvey

Spencer's Socket Site: Has several C++socket library links:

http://www.lowtek.com/sockets/

From: shashikant

I need some examples sourcecode on device driver writing on Unix.

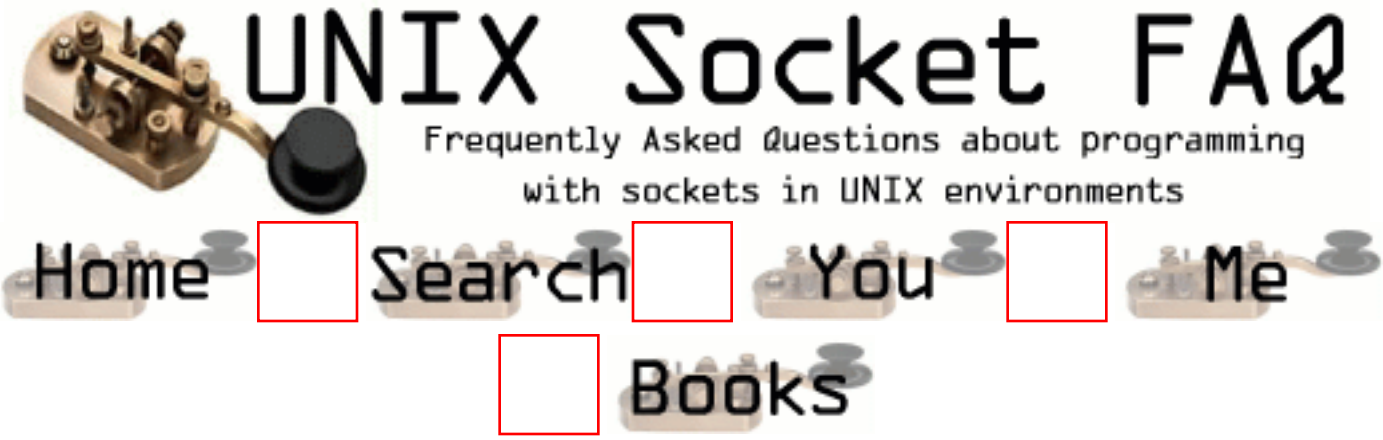
From:

Don't use ACE. It is shite. It is impossible to trap certain errors (for instance, failures in its gethostbyname abstraction) using ACE.

From: d ross

Search Google for "WFC" and "Blackburn" - Sam Blackburn wrote the WFC library (source included) and it is very good. I think it's now on the CodeProject site.

Add a Comment



Sample Source Code: perl examples of source code

From: Dave Atkins

The only examples in the sample source code are for C programs. Are perl examples available?

From: Lars Gregersen

There are several documents comming with Perl that contains information on sockets. I have found the standard documentation very helpful and have used code based on the examples on Windows and Unix.

Look at the documentation for
IO::Socket
IO::Select
perlfunc
and Perl book you can find
In that order and your questions will be answered.

Lars Gregersen

From: James Lawrence

I would reccomend the following book.

PERL5 by Example by David Medinets
Publisher=Que ISBN=0-7897-0866-3

1/4 of this book's content deals with network programming. Also, if you are into PERl as much as I am, you need this one on your shelf. You will not be dissapointed.

Enjoy !!

From: Marc

Check out http://quake.skif.net/RawIP. Home of Net::RawIP
and alot of source examples

From: Bart Van Eynde

check out Sander's page:
title: Network Programming in Perl
link: http://home.t-online.de/home/320054954159-0001/perlnet/

A lot of good working examples of perl code using sockets

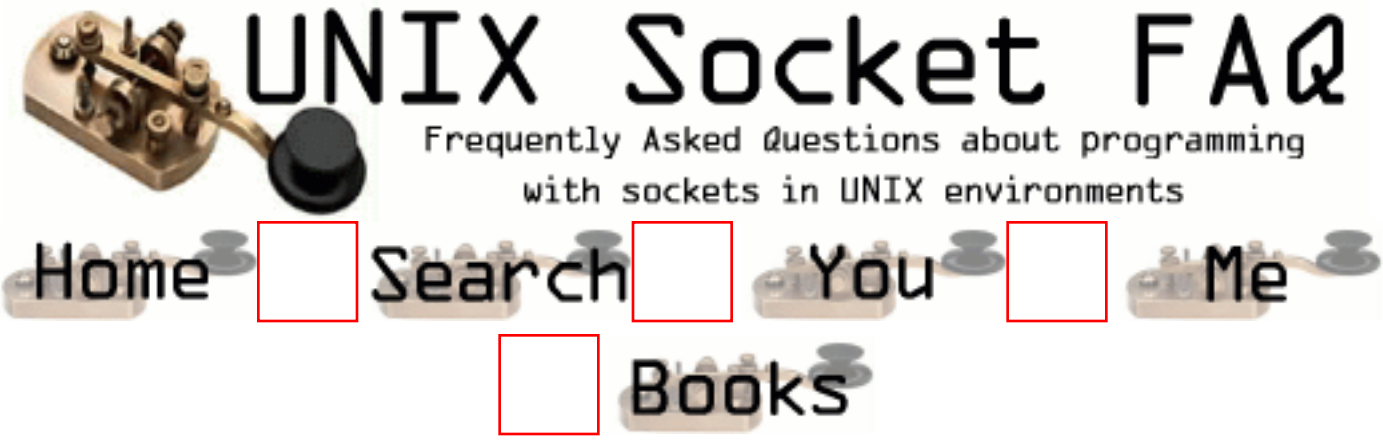
From: Bin Lu

I get some problem writing sockets in perl. When I want to send an array at client to the server, the server doesn't get the data correctly. I use 'send(SOCK, @array, 4, 0);'. Is there something wrong with the code? I also tried to send a char but failed too. Could someone give me a hint how to solve this problem? And how can I assemble some variables to a block of data (like a buffer) and send it out through the socket? I'd really appreciate it a lot!

From: Daniel Gustafson

The "Network Programming in Perl" site has moved to: http://www.1024kb.net/perlnet.html. Just wanted to post it before the redirect disappears.

Add a Comment



Sample Source Code: Where is the source code from Richard Stevens' books?

From: Madhumathi

How do I get additional source code mentioned in TCP-IP
- The protocols book. It is in ftp.uu.net site ?
Are there any alternate sites for this site.

From: subodh nanal

hi madhumati

u can get the source code for stevens book at the following site i guess

www.kohala.com/~rstevens

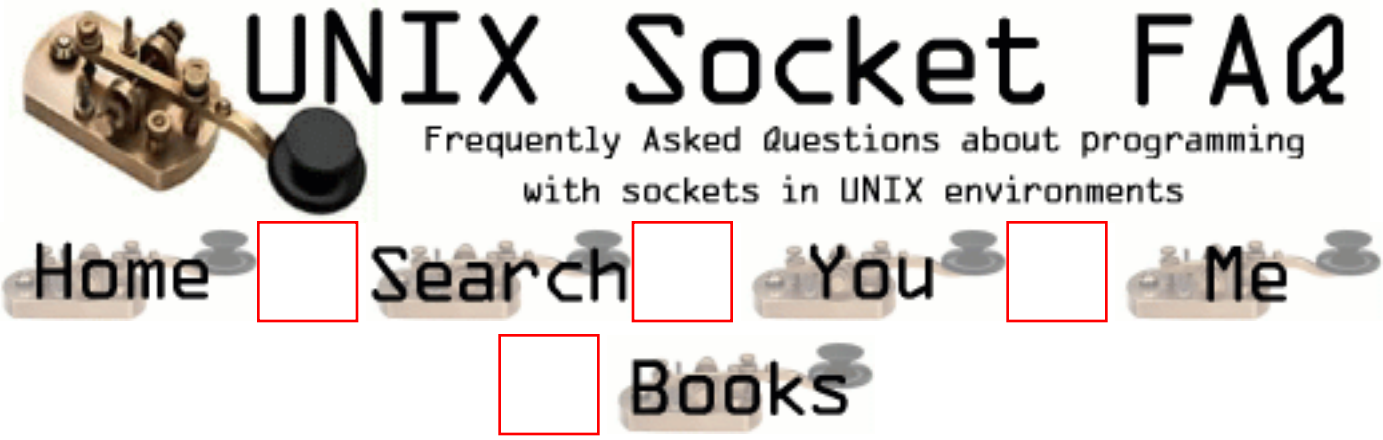
hope this helps

From:

... or these days from:

www.kohala.com/start

Add a Comment



Bugs and Strange Behaviour: send() hangs up when sending to a switched off computer

From: Raed Sirhan

I have a server program that sends a lot of TCP messages to a group of clients (array of sockets), but I have a problem that the send() will hang up when sending many messages to a switched off computer (or to a computer that already disconnected from the network by cable or hub failure). I tried SO_KEEPALIVE but it takes a long of time before detecting that the socket was broken. so please, till what should I do to prevent the temporary hang up of the send() command

---

From: Vic Metcalfe

It seems to me that non blocking sockets would be a help here, but I don't have any good resources to point to for non blocking sockets. Its a topic I'd like to cover more in the faq, and add to the examples, but with other projects pressing for time it just hasn't been done. You might be able to do it by piecing together information from DejaNews. Good luck! Vic.

From: Ian Berry

I have a similar problem, but I am using non-blocking sockets.
I am trying to create a simple server client using stream sockets where the server can handle the client stopping without properly closing the socket (in most cases i will be ctrl-c'ing the client). In general everything works, unless you ctrl-c at the point when the server is doing a send to the client, at which point the server stops totally returning me to the prompt.
any thoughts?

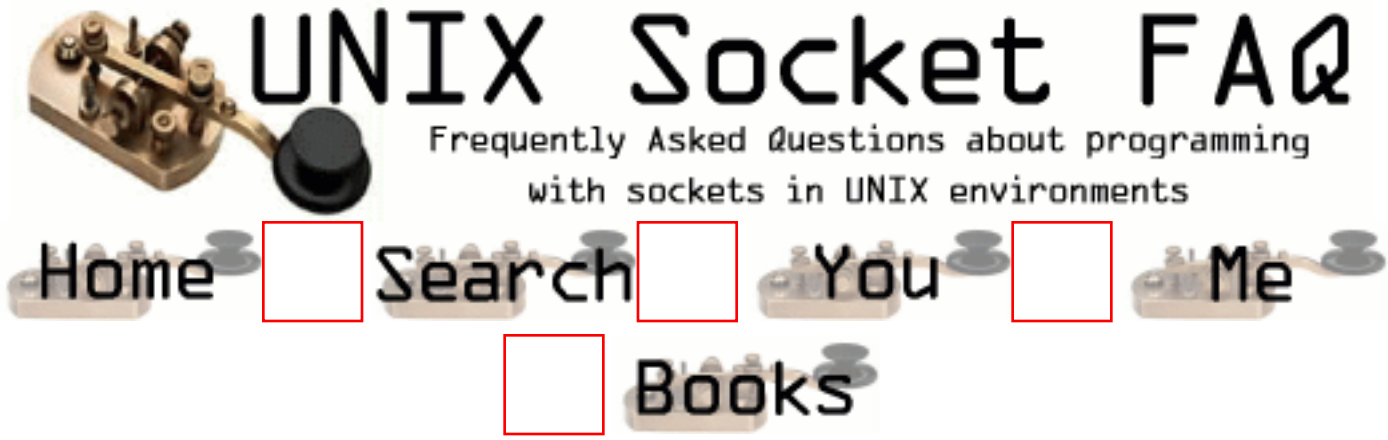
From: Xidong Wang

you can catch the signal SIGINT, which is send to all the foreground processes by entering Ctrl-C.

From: Rob Seace

Yeah, but if he's seeing the SERVER process croak after killing the client, while the server is trying to write to the client, then the most likely problem is SIGPIPE that's killing it, not SIGINT... (INT is killing the client, but PIPE is most likely killing the server...) Personally, in literally EVERY TCP/IP related program I write, one of the first things I do is "signal (SIGPIPE, SIG_IGN);"... Then, you don't get hit with SIGPIPE while writing to a closed socket; but, instead, you get an error return from the write, with errno set to EPIPE (which is a LOT easier to deal with)... I believe this is all covered elsewhere in this FAQ, too...

Add a Comment



Bugs and Strange Behaviour: Error when using inetd

From: Walter Moore

I've written a client/server program to replace someone's code that entailed a multitude of rsh's. This server can be called on the command line, as well as via inetd. My problem is that the inetd calling is not working correctly. When the client connects to the port, it receives this message: ld.so.1: (unknown): fatal: libct.so: can't open file: errno=2 What does this mean? Thanks

---

Add a Comment

UNIX Socket FAQ

Frequently Asked Questions about programming with sockets in UNIX environments

Home | Search | You | Me | Books

Add a new question to the FAQ

Comments will fall under the same copyright as the FAQ, which permits anything except taking ownership of the work. You can use HTML tags, and line breaks will be converted to html <BR> tags.

Your Name:

Your Email:

Comment Type: Plain Text

Subject:

Add your Question

| FAQ Home || Search FAQ || Your User Profile || About Me || Books |

Raw IP Networking FAQ

Version 1.3

Last Modified on: Thu Nov 11 18:18:19 PST 1999

The master copy of this FAQ is currently kept at

http://www.whitefang.com/rin/

The webpage also contains material that supplements this FAQ, along with a very spiffy html version.

If you wish to mirror it officially, please contact me for details.

Copyright

I am responsible for the validity of all information found in this FAQ.

This FAQ may contain errors, or inaccurate material. Use it at your own risk. Although an effort is made to keep all the material presented here accurate, the contributors and maintainer of this FAQ will not be held responsible for any damage -- direct or indirect -- which may result from inaccuracies.

Introduction

The following FAQ attempts to answer questions regarding raw IP or low level IP networking, including raw sockets, and network monitoring APIs such as BPF and DLPI.

Additions and Contributions

If you find anything you can add, have some corrections for me or would like a question answered, please send email to:

Thamer Al-Herbish <shadows@whitefang.com>

Please remember to include whether or not you want your email address reproduced on the FAQ (if you're contributing). Also remember that you may want to post your question to Usenet, instead of sending it to me. If you get a response which is not found on this FAQ, and you feel is relevant, mail me both copies and I'll attempt to include it.

Also a word on raw socket bugs. I get approximately a couple of emails a month about them, and sometimes I just can't verify if the bug exists on a said system. Before mailing in the report, double

check with my example source code. If it looks like it's a definite bug, then mail it in.

Special thanks to John W. Temples <john@whitefang.com> for his constant healthy criticism and editing of the FAQ.

Credit is given to the contributor as his/her contribution appears in the FAQ, along with a list of all contributors at the end of this document.

A final note, a Raw IP Networking mailing list is up. You can join by sending an empty message to rawip-subscribe@whitefang.com

Caveat

This FAQ covers only information relevant to the UNIX environment.

Table of Contents

1) General Questions:

1.1) What tools/sniffers can I use to monitor my network?

Depending on your operating system, the following is an incomplete list of available tools:

tcpdump: Found out-of-the-box on most BSD variants, and also available separately from ftp://ftp.ee.lbl.gov/tcpdump.tar.Z along with libpcap (see below) and various other tools. This tool, in particular, has been ported to multiple platforms thanks to libpcap.

ipgrab Compatible with many systems. ipgrab displays link level, transport level, and network level information on packets captured verbosely. http://www.xnet.com/~cathmike/MSB/Software/

Ethereal (GUI) A network packet analyzer (uses GTK+). Supports many systems. Available at: http://ethereal.zing.org/

tcptrace:
http://jarok.cs.ohiou.edu/software/tcptrace/tcptrace.html
Not an actual sniffer, but can read from the logs produced by many other well known sniffers to produce output in different formats and in adjustable details (includes diagnostics).

tcpflow
http://www.circlemud.org/~jelson/software/tcpflow/
tcpflow is a program that captures data transmitted as part of TCP connections (flows), and stores the data in a way that is convenient for protocol analysis or debugging.

snoop: Solaris, IRIX.

etherfind: SunOS.

Packetman: SunOS, DEC-MIPS, SGI, DEC-Alpha, and Solaris. Available at ftp://ftp.cs.curtin.edu.au:/pub/netman/

nettl/ntfmt: HP/UX

1.2) What packet capturing facilities are available?

Depending on your operating system (different versions may vary):

BPF: Berkeley Packet Filter. Commonly found on BSD variants.

DLPI: Data Link Provider Interface. Solaris, HP-UX, SCO

	Openserver.
NIT:	Network Interface Tap. SunOS 3.
SNOOP:	(???). IRIX.
SNIT:	STREAMS Network Interface Tap. SunOS 4.
SOCK_PACKET:	Linux.
LSF:	Linux Socket Filter. Is available on Linux 2.1.75 onwards.
drain:	Used to snoop packets dropped by the OS. IRIX.

1.3) Is there a portable API I can use to capture packets?

Yes. libpcap from ftp://ftp.ee.lbl.gov/libpcap.tar.Z attempts to provide a single API that interfaces with different OS-dependent packet capturing APIs. It's always best, of course, to learn the underlying APIs in case this library might hide some interesting features. It's important to warn the reader that I have seen different versions of libpcap break backward compatibility.

1.4) How does a packet capturing facility work?

The exact details are dependent on the operating system. However, the following will attempt to illustrate the usual technique used in various implementations:

The user process opens a device or issues a system call which gives it a descriptor with which it can read packets off the wire. The kernel then passes the packets straight to the process.

However, this wouldn't work too well on a busy network or a slow machine. The user process has to read the packets as fast as they appear on the network. That's where buffering and packet filtering come in.

The kernel will buffer up to X bytes of packet data, and pass the packets one by one at the user's request. If the amount exceeds a certain limit (resources are finite), the packets are dropped and are not placed in the buffer.

Packet filters allow a process to dictate which packets it's interested in. The usual way is to have a set of opcodes for routines to perform on the packet, reading values off it, and deciding whether or not it's wanted. These opcodes usually perform very simple operations, allowing powerful filters to be constructed.

BPF filters and then buffers; this is optimal since the buffer only contains packets that are interesting to the process. It's hoped that the filter cuts down the amount of packets buffered to stop overflowing the buffer, which leads

to packet loss.

NIT, unfortunately, does not do this; it applies the filter after buffering, when the user process starts to read from the buffered data.

According to route <route@infonexus.com> Linux' SOCK_PACKET does not do any buffering and has no kernel filtering.

Your mileage may vary with other packet capturing facilities.

1.5) How do I limit packet loss when sniffing a network?

If you're experiencing a lot of packet loss, you may want to limit the scope of the packets read by using filters. This will only work if the filtering is done before any buffering. If this still doesn't work because your packet capturing facility is broken like NIT, you'll have to read the packets faster in a user process and send them to another process -- basically attempt to do additional buffering in user space.

Another way of improving performance, is by using a larger buffer. On Irix using SNOOP, the man page recommends using SO_RCVBUF. On BSD with BPF one can use the BIOCSBLEN ioctl call to increase the buffer size. On Solaris bufmod and pfmod can be used for altering buffer size and filters respectively.

Remember, the longer your process is busy and not attending the incoming packets, the quicker they'll be dropped by the kernel.

1.6) What is packet capturing usually used for?

(Question suggested by Michael T. Stolarchuk <mts@rare.net> along with some suggestions for the answer.)

> Network diagnostics such as the verification of a network's setup, examples are tools like arp, that report the ARP messages sent from hosts.
>
> Reconstruction of end to end sessions. tcpshow attempts to do this, but more sophisticated examples are the array of security tools which try to keep tabs on network connections.
>
> Monitoring network load. Probably one of the most practical uses, a lot of commercial products usually use specialized hardware to accomplish this.

1.7) Will I have to replace any packets captured off the network?

No, the packet capturing facilities mentioned make copies of the packets, and do not remove them from the system's TCP/IP

stack. If you wish to prevent packets from reaching the TCP/IP stack you need to use a firewall, (which should be able to do packet filtering). Don't confuse the packet filtering done by packet capturing facilities with those done by firewalls. They serve different purposes.

1.8) Is there a portable API to send raw packets into a network?

Yes, route <route@infonexus.com> maintains Libnet, a library that provides an API for low level packet writing and handling. It serves as a good compliment for libpcap, if you wish to read and write packets. The project's webpage can be found at:

http://www.packetfactory.net/libnet/

1.9) Are there any high level language APIs (Not C) for raw IP access?

A PERL module that gives access to raw sockets is available at:

http://quake.skif.net/RawIP/

A Python library "py-libpap" can be found at:

ftp://ftp.python.org/pub/python/contrib/Network/

2) RAW socket questions:

2.1) What is a RAW socket?

The BSD socket API allows one to open a raw socket and bypass layers in the TCP/IP stack. Be warned that if an OS doesn't support correct BSD semantics (correct is used loosely here), you're going to have a hard time making it work. Below, an attempt is made to address some of the bugs or surprises you're in store for. On almost all sane systems only root (superuser) can open a raw socket.

2.2) How do I use a raw socket?

2.2.1) How do I send a TCP/IP packet through a raw socket?

Depending on what you want to send, you initially open a socket and give it its type.

```
sockd = socket(AF_INET,SOCK_RAW,<protocol>);
```

You can choose from any protocol including IPPROTO_RAW. The protocol number goes into the IP header verbatim. IPPROTO_RAW places 0 in the IP header.

Most systems have a socket option IP_HDRINCL which allows you to include your own IP header along with the rest of the packet. If your system doesn't have this option, you may or may not be able to include your own IP header. If it is available, you should use it as such:

```
char on = 1;
setsockopt(sockd,IPPROTO_IP,IP_HDRINCL,&on,sizeof(on));
```

Of course, if you don't want to include an IP header, you can always specify a protocol in the creation of the socket and slip your transport level header under it.

You then build the packet and use a normal sendto().

2.2.2) How do I build a TCP/IP packet?

Examples can be found at http://www.whitefang.com/rin/ which attempt to illustrate the details involved. They also illustrate some of the bugs mentioned below.

Briefly, you need to actually write the packet out in memory and hand it over to the socket where it will hopefully fire it away and await more packets.

2.2.3) How can I listen for packets with a raw socket?

Traditionally the BSD socket API did not allow you to listen to just any incoming packet via a raw socket. Although Linux (2.0.30 was the last version I had a look at), did allow this, it has to do with their own implementation of the TCP/IP stack. Correct BSD semantics allow you to get some packets which match a certain category (see below).

There's a logical reason behind this; for example TCP packets are always handled by the kernel. If the port is open, send a SYN-ACK and establish the connection, or send back a RST. On the other hand, some types of ICMP (I compiled a small list below), the kernel can't handle. Like an ICMP echo reply, is passed to a matching raw socket, since it was meant for a user program to receive it.

The solution is to firewall that particular port if it was a UDP or TCP packet, and sniff it with a packet capturing API (a list is mentioned above). This prevents the TCP/IP stack from handling the packet, thus it will be ignored and you can handle it yourself without intervention.

If you don't firewall it, and reply yourself you'll wind up having additional responses from your operating system!

Here's a concise explanation of the semantics of a raw BSD socket, taken from a Usenet post by W. Richard Stevens

From <rstevens@kohala.com> (Sun Jul 6 12:07:07 1997) :

"The semantics of BSD raw sockets are:

- TCP and UDP: no one other than the kernel gets these.

- ICMP: a copy of each ICMP gets passed to each matching raw socket, except for a few that the kernel generates the reply for: ICMP echo request, timestamp request, and mask request.

- IGMP: all of these get passed to all matching raw sockets.

- all other protocols that the kernel doesn't deal with (OSPF, etc.): these all get passed to all matching raw sockets."

After looking at the icmp_input() routine from the 4.4BSD's TCP/IP stack, it seems the following ICMP types will be passed to matching raw sockets:

Echo Reply: (0)

Router Advertisement (9)

Time Stamp Reply (13)

Mask Reply (18)

2.3) What bugs should I look out for when using a raw socket?

2.3.1) IP header length/offset host/network byte (feature/bug?)

Systems derived from 4.4BSD have a bug in which the ip_len and ip_off members of the ip header have to be set in host byte order rather than network byte order. Some systems may have fixed this. I've confirmed this bug has been fixed on OpenBSD 2.1.

2.3.2) Unwanted packet processing on some systems.

Thanks to Michael Masino <mmasino@mitre.org> , Lamont Granquist <lamontg@hitl.washington.edu> , and route <route@infonexus.com> for the submission of bug reports.

Some systems will process some of the fields in the IP

and transport headers. I've attempted to verify the reports I've received here's what I can verify for sure.

Solaris (at least 2.5/2.6) and changes the IP ID field, and adds a Do Not Fragment flag to the IP header (IP_DF). It also expects the checksum to contain the length of the transport level header, and the data.

Further reports which I cannot verify (can't reproduce), consist of claims that Solaris 2.x and Irix 6.x will change the sequence and acknowledgment numbers. Irix 6.x is also believed to have the problem mentioned in the previous paragraph. If you experience these problems, double check with the example source code.

You'll save yourself a lot of trouble by just getting Libnet http://www.packetfactory.net/libnet/

2.4) What are raw sockets commonly used for?

Various UNIX utilities use raw sockets, among them are: traceroute, ping, arp. Also, a lot of Internet security tools make use of raw sockets. However in the long run, raw sockets have proven bug ridden, unportable and limited in use.

3) libpcap (A Portable Packet Capturing Library)

3.1) Why should I use libpcap, instead of using the native API on my operating system for packet capturing?

---

libpcap was written so that applications could do packet capturing portably. Since it's system independent and supports numerous operating systems, your packet capturing application becomes more portable to various other systems.

3.2) Does libpcap have any disadvantages, which I should be aware of?

Yes, libpcap will only use in-kernel packet filtering when using BPF, which is found on BSD derived systems. This means any packet filters used on other operating systems which don't use BPF will be done in user space, thus losing out on a lot of speed and efficiency. This is not what you want, because packet loss can increase when sniffing a busy network.

DEC OSF/1 has an API which has been extended to support BPF-style filters; libpcap does utilize this.

In the future, libpcap may translate BPF style filters to other packet capturing facilities, but this has not been implemented yet as of version 0.3

Refer to question 1.4 to see how packet filters help in reliably monitoring your network.

3.3) Where can I find example libpcap source code?

A lot of the source code found at LBNL's ftp archive ftp://ftp.ee.lbl.gov/ uses libpcap. More specifically, ftp://ftp.ee.lbl.gov/tcpdump.tar.Z probably demonstrates libpcap to a large extent.

4) List of contributors.

Thamer Al-Herbish <shadows@whitefang.com>
W. Richard Stevens <rstevens@kohala.com>
John W. Temples (III) <john@whitefang.com>
Michael Masino <mmasino@mitre.org>
Lamont Granquist <lamontg@hitl.washington.edu>
Michael T. Stolarchuk <mts@rare.net>
Mike Borella <Mike_Borella@mw.3com.com>
route <route@infonexus.com>
Derrick J Brashear <shadow@dementia.org>

Raw IP Networking FAQ

Version 1.3

Last Modified on: **Thu Nov 11 18:18:19 PST 1999**

The master copy of this FAQ is currently kept at

http://www.whitefang.com/rin/

The webpage also contains material that supplements this FAQ, along with a very spiffy html version.

If you wish to mirror it officially, please contact me for details.

Copyright

Introduction

The following FAQ attempts to answer questions regarding raw IP or low level IP networking, including raw sockets, and network monitoring APIs such as BPF and DLPI.

Additions and Contributions

If you find anything you can add, have some corrections for me or would like a question answered, please send email to:

Thamer Al-Herbish <shadows@whitefang.com>

Please remember to include whether or not you want your email address reproduced on the FAQ (if you're contributing). Also remember that you may want to post your question to Usenet, instead of sending it to me. If you get a response which is not found on this FAQ, and you feel is relevant, mail me both copies and I'll attempt to include it.

Also a word on raw socket bugs. I get approximately a couple of emails a month about them, and sometimes I just can't verify if the bug exists on a said system. Before mailing in the report, double check with my example source code. If it looks like it's a definite bug, then mail it in.

Special thanks to John W. Temples <john@whitefang.com> for his constant healthy criticism and editing of the FAQ.

Credit is given to the contributor as his/her contribution appears in the FAQ, along with a list of all contributors at the end of this document.

A final note, a Raw IP Networking mailing list is up. You can join by sending an empty message to rawip-subscribe@whitefang.com

Caveat

This FAQ covers only information relevant to the UNIX environment.

Table of Contents

1) General Questions:

1.1) What tools/sniffers can I use to monitor my network?

Depending on your operating system, the following is an incomplete list of available tools:

tcpdump: Found out-of-the-box on most BSD variants, and also available separately from ftp://ftp.ee.lbl.gov/tcpdump.tar.Z along with libpcap (see below) and various other tools. This tool, in particular, has been ported to multiple platforms thanks to libpcap.

ipgrab Compatible with many systems. ipgrab displays link level, transport level, and network level information on packets captured verbosely. http://www.xnet.com/~cathmike/MSB/Software/

Ethereal (GUI) A network packet analyzer (uses GTK+). Supports many systems. Available at: http://ethereal.zing.org/

tcptrace: http://jarok.cs.ohiou.edu/software/tcptrace/tcptrace.html Not an actual sniffer, but can read from the logs produced by many other well known sniffers to produce output in different formats and in adjustable details (includes diagnostics).

tcpflow http://www.circlemud.org/~jelson/software/tcpflow/ tcpflow is a program that captures data transmitted as part of TCP connections (flows), and stores the data in a way that is convenient for protocol analysis or debugging.

snoop: Solaris, IRIX.

etherfind: SunOS.

Packetman: SunOS, DEC-MIPS, SGI, DEC-Alpha, and Solaris. Available at ftp://ftp.cs.curtin.edu.au:/pub/netman/

nettl/ntfmt: HP/UX

1.2) What packet capturing facilities are available?

Depending on your operating system (different versions may vary):

BPF: Berkeley Packet Filter. Commonly found on BSD variants.

DLPI: Data Link Provider Interface. Solaris, HP-UX, SCO Openserver.

NIT: Network Interface Tap. SunOS 3.

SNOOP: (???). IRIX.

SNIT: STREAMS Network Interface Tap. SunOS 4.

SOCK_PACKET: Linux.

LSF: Linux Socket Filter. Is available on Linux 2.1.75 onwards.

drain: Used to snoop packets dropped by the OS. IRIX.

1.3) Is there a portable API I can use to capture packets?

Yes. libpcap from ftp://ftp.ee.lbl.gov/libpcap.tar.Z attempts to provide a single API that interfaces with different OS-dependent packet capturing APIs. It's always best, of course, to learn the underlying APIs in case this library might hide some interesting features. It's important to warn the reader that I have seen different versions of libpcap break backward compatibility.

1.4) How does a packet capturing facility work?

The exact details are dependent on the operating system. However, the following will attempt to illustrate the usual technique used in various implementations:

The user process opens a device or issues a system call which gives it a descriptor with which it can read packets off the wire. The kernel then passes the packets straight to the process.

However, this wouldn't work too well on a busy network or a slow machine. The user process has to read the packets as fast as they appear on the network. That's where buffering and packet filtering come in.

The kernel will buffer up to X bytes of packet data, and pass the packets one by one at the user's request. If the amount exceeds a certain limit (resources are finite), the packets are dropped and are not placed in the buffer.

Packet filters allow a process to dictate which packets it's interested in. The usual way is to have a set of opcodes for routines to perform on the packet, reading values off it, and deciding whether or not it's wanted. These opcodes usually perform very simple operations, allowing powerful filters to be constructed.

BPF filters and then buffers; this is optimal since the buffer only contains packets that are interesting to the process. It's hoped that the filter cuts down the amount of packets buffered to stop overflowing the buffer, which leads to packet loss.

NIT, unfortunately, does not do this; it applies the filter after buffering, when the user process starts to read from the buffered data.

According to route <route@infonexus.com> Linux' SOCK_PACKET does not do any buffering and has no kernel filtering.

Your mileage may vary with other packet capturing facilities.

1.5) How do I limit packet loss when sniffing a network?

If you're experiencing a lot of packet loss, you may want to limit the scope of the packets read by using filters. This will only work if the filtering is done before any buffering. If this still doesn't work because your packet capturing facility is broken like NIT, you'll have to read the packets faster in a user process and send them to another process -- basically attempt to do additional buffering in user space.

Another way of improving performance, is by using a larger buffer. On Irix using SNOOP, the man page recommends using SO_RCVBUF. On BSD with BPF one can use the BIOCSBLEN ioctl call to increase the buffer size. On Solaris bufmod and pfmod can be used for altering buffer size and filters respectively.

Remember, the longer your process is busy and not attending the incoming packets, the quicker they'll be dropped by the kernel.

1.6) What is packet capturing usually used for?

(Question suggested by Michael T. Stolarchuk <mts@rare.net> along with some suggestions for the answer.)

- Network diagnostics such as the verification of a network's setup,

examples are tools like arp, that report the ARP messages sent from hosts.

- Reconstruction of end to end sessions. tcpshow attempts to do this, but more sophisticated examples are the array of security tools which try to keep tabs on network connections.
- Monitoring network load. Probably one of the most practical uses, a lot of commercial products usually use specialized hardware to accomplish this.

1.7) Will I have to replace any packets captured off the network?

No, the packet capturing facilities mentioned make copies of the packets, and do not remove them from the system's TCP/IP stack. If you wish to prevent packets from reaching the TCP/IP stack you need to use a firewall, (which should be able to do packet filtering). Don't confuse the packet filtering done by packet capturing facilities with those done by firewalls. They serve different purposes.

1.8) Is there a portable API to send raw packets into a network?

Yes, route <route@infonexus.com> maintains Libnet, a library that provides an API for low level packet writing and handling. It serves as a good compliment for libpcap, if you wish to read and write packets. The project's webpage can be found at:

http://www.packetfactory.net/libnet/

1.9) Are there any high level language APIs (Not C) for raw IP access?

A PERL module that gives access to raw sockets is available at:

http://quake.skif.net/RawIP/

A Python library "py-libpap" can be found at:

ftp://ftp.python.org/pub/python/contrib/Network/

Back to Top

2) RAW socket questions:

2.1) What is a RAW socket?

The BSD socket API allows one to open a raw socket and bypass layers in the TCP/IP stack. Be warned that if an OS doesn't support correct BSD

semantics (correct is used loosely here), you're going to have a hard time making it work. Below, an attempt is made to address some of the bugs or surprises you're in store for. On almost all sane systems only root (superuser) can open a raw socket.

- ## 2.2) How do I use a raw socket?

  - ### 2.2.1) How do I send a TCP/IP packet through a raw socket?

    Depending on what you want to send, you initially open a socket and give it its type.

    sockd = socket(AF_INET,SOCK_RAW,<protocol>);

    You can choose from any protocol including IPPROTO_RAW. The protocol number goes into the IP header verbatim. IPPROTO_RAW places 0 in the IP header.

    Most systems have a socket option IP_HDRINCL which allows you to include your own IP header along with the rest of the packet. If your system doesn't have this option, you may or may not be able to include your own IP header. If it is available, you should use it as such:

    char on = 1;
    setsockopt(sockd,IPPROTO_IP,IP_HDRINCL,&on,sizeof(on));

    Of course, if you don't want to include an IP header, you can always specify a protocol in the creation of the socket and slip your transport level header under it.

    You then build the packet and use a normal sendto().

  - ### 2.2.2) How do I build a TCP/IP packet?

    Examples can be found at http://www.whitefang.com/rin/ which attempt to illustrate the details involved. They also illustrate some of the bugs mentioned below.

    Briefly, you need to actually write the packet out in memory and hand it over to the socket where it will hopefully fire it away and await more packets.

  - ### 2.2.3) How can I listen for packets with a raw socket?

    Traditionally the BSD socket API did not allow you to listen to just any incoming packet via a raw socket. Although Linux (2.0.30 was

the last version I had a look at), did allow this, it has to do with their own implementation of the TCP/IP stack. Correct BSD semantics allow you to get some packets which match a certain category (see below).

There's a logical reason behind this; for example TCP packets are always handled by the kernel. If the port is open, send a SYN-ACK and establish the connection, or send back a RST. On the other hand, some types of ICMP (I compiled a small list below), the kernel can't handle. Like an ICMP echo reply, is passed to a matching raw socket, since it was meant for a user program to receive it.

The solution is to firewall that particular port if it was a UDP or TCP packet, and sniff it with a packet capturing API (a list is mentioned above). This prevents the TCP/IP stack from handling the packet, thus it will be ignored and you can handle it yourself without intervention.

If you don't firewall it, and reply yourself you'll wind up having additional responses from your operating system!

Here's a concise explanation of the semantics of a raw BSD socket, taken from a Usenet post by W. Richard Stevens

From <rstevens@kohala.com> (Sun Jul 6 12:07:07 1997) :

"The semantics of BSD raw sockets are:

- TCP and UDP: no one other than the kernel gets these.
- ICMP: a copy of each ICMP gets passed to each matching raw socket, except for a few that the kernel generates the reply for: ICMP echo request, timestamp request, and mask request.
- IGMP: all of these get passed to all matching raw sockets.
- all other protocols that the kernel doesn't deal with (OSPF, etc.): these all get passed to all matching raw sockets."

After looking at the icmp_input() routine from the 4.4BSD's TCP/IP stack, it seems the following ICMP types will be passed to matching raw sockets:

- Echo Reply: (0)
- Router Advertisement (9)
- Time Stamp Reply (13)
- Mask Reply (18)

- **2.3) What bugs should I look out for when using a raw socket?**
  - **2.3.1) IP header length/offset host/network byte (feature/bug?)**

    Systems derived from 4.4BSD have a bug in which the ip_len and ip_off members of the ip header have to be set in host byte order rather than network byte order. Some systems may have fixed this. I've confirmed this bug has been fixed on OpenBSD 2.1.

  - **2.3.2) Unwanted packet processing on some systems.**

    Thanks to Michael Masino <mmasino@mitre.org> , Lamont Granquist <lamontg@hitl.washington.edu> , and route <route@infonexus.com> for the submission of bug reports.

    Some systems will process some of the fields in the IP and transport headers. I've attempted to verify the reports I've received here's what I can verify for sure.

    Solaris (at least 2.5/2.6) and changes the IP ID field, and adds a Do Not Fragment flag to the IP header (IP_DF). It also expects the checksum to contain the length of the transport level header, and the data.

    Further reports which I cannot verify (can't reproduce), consist of claims that Solaris 2.x and Irix 6.x will change the sequence and acknowledgment numbers. Irix 6.x is also believed to have the problem mentioned in the previous paragraph. If you experience these problems, double check with the example source code.

    You'll save yourself a lot of trouble by just getting Libnet http://www.packetfactory.net/libnet/

- **2.4) What are raw sockets commonly used for?**

  Various UNIX utilities use raw sockets, among them are: traceroute, ping, arp. Also, a lot of Internet security tools make use of raw sockets. However in the long run, raw sockets have proven bug ridden, unportable and limited in use.

- **3) libpcap (A Portable Packet Capturing Library)**
  - **3.1) Why should I use libpcap, instead of using the native API on my operating system for packet capturing?**

    libpcap was written so that applications could do packet capturing portably.

Since it's system independent and supports numerous operating systems, your packet capturing application becomes more portable to various other systems.

- **3.2) Does libpcap have any disadvantages, which I should be aware of?**

  Yes, libpcap will only use in-kernel packet filtering when using BPF, which is found on BSD derived systems. This means any packet filters used on other operating systems which don't use BPF will be done in user space, thus losing out on a lot of speed and efficiency. This is not what you want, because packet loss can increase when sniffing a busy network.

  DEC OSF/1 has an API which has been extended to support BPF-style filters; libpcap does utilize this.

  In the future, libpcap may translate BPF style filters to other packet capturing facilities, but this has not been implemented yet as of version 0.3

  Refer to question 1.4 to see how packet filters help in reliably monitoring your network.

- **3.3) Where can I find example libpcap source code?**

  A lot of the source code found at LBNL's ftp archive ftp://ftp.ee.lbl.gov/ uses libpcap. More specifically, ftp://ftp.ee.lbl.gov/tcpdump.tar.Z probably demonstrates libpcap to a large extent.

4) List of contributors.

- Thamer Al-Herbish <shadows@whitefang.com>
- W. Richard Stevens <rstevens@kohala.com>
- John W. Temples (III) <john@whitefang.com>
- Michael Masino <mmasino@mitre.org>
- Lamont Granquist <lamontg@hitl.washington.edu>
- Michael T. Stolarchuk <mts@rare.net>
- Mike Borella <Mike_Borella@mw.3com.com>
- route <route@infonexus.com>
- Derrick J Brashear <shadow@dementia.org>

The comp.unix.programmer Resources Page

Administrative articles:

- **[READ ME FIRST] Welcome to comp.unix.programmer**

Unix Programming Frequently Asked Questions

The comp.unix.programmer FAQ is available here in the following formats:

- HTML
- Plain text
- Texinfo source

You may make copies of the FAQ for private use, or to put up on internal or private web servers, without restriction. To assist with this, you may download a .tar.gz copy of the whole FAQ. You **MUST NOT** put copies of this FAQ on public webservers without permission, except as part of a properly maintained news.answers archive. (This restriction is purely to prevent proliferation of old versions of the document.)

Links to items maintained elsewhere:

(Please feel free to suggest any new links to be added here.)

- The unix-socket-faq, maintained by Vic Metcalfe, to which I have contributed heavily.
- Shigio Yamaguchi's Kernel Source Tour; HTMLized and indexed FreeBSD and Linux kernel sources.
- FreeBSD.
- XFree86.
- Jennifer Myers' Unix Reference Desk. (Loads of good stuff!)
- More links please!

---

webmaster@erlenstar.demon.co.uk


Demon

Frequently Asked Questions about Unix Programming

This is a mirror copy of the FAQ document for the comp.unix.programmer newsgroup. Please send all comments, queries, suggestions, criticism etc. to the maintainer, andrew@erlenstar.demon.co.uk. Suggestions for new questions or answers are always welcome.

Proceed to Table of Contents

Also available:

- The complete document in plain text (c. 100k).
- A gzipped tar archive of both the HTML and plain text versions (c. 70k)

This copy is current as of **September 01, 2000**. If it is more than two months old, then please inform the maintainer of the archive where you obtained it, or obtain an updated copy from the distribution sites listed below.

Distribution policy

The full hypertext copy is available at the following sites:

- http://www.erlenstar.demon.co.uk/unix/ (the master copy)
- http://www.whitefang.com/unix/

In addition, the raw Texinfo source is available only at the master site; it currently doesn't work to generate Info output, but usually produces passable hardcopy.

The plain text version is posted to comp.unix.programmer, comp.answers and news.answers every two weeks, and can always be found in the news.answers archives.

---

andrew@erlenstar.demon.co.uk

Welcome of the web site for the UNIX Socket FAQ! I will be getting back to updating this site soon, to restore the old links, etc. I'll also automate the mirroring so new questions go out to the mirror sites.

Please help a friend of mine who is promoting his single by downloading and listening to 'Notawasaga'. You'll be helping him to find more listeners, and maybe it'll help soothe your nerves when your clients and servers just aren't communicating like they should.

You are using a static mirror of the faq, so not all features will be available to you. The home page that has all the features can be found at http://www.lcg.org/sock-faq/.

Categorized Questions:

```
Archive-name:       internet/tcp-ip/tcp-ip-faq/contents
Version:            5.15
Last-modified:      1999-09-06 20:11:43
Posting-Frequency:  monthly (first Friday)
Maintainer:         tcp-ip-faq@eng.sun.com (Mike Oliver)
URL:                http://www.itprc.com/tcpipfaq/default.htm
```

TCP/IP Frequently Asked Questions

Table of Contents

This is the Table of Contents for the Frequently Asked Questions (**FAQ**) list for the comp.protocols.tcp-ip Usenet newsgroup. The FAQ provides answers to a selection of common questions on the various protocols (IP, TCP, UDP, ICMP and others) that make up the TCP/IP protocol suite. It is posted to the news.answers, comp.answers and comp.protocols.tcp-ip newsgroups on or about the first Friday of every month.

The FAQ is posted in two parts. Part 1 contains answers to general questions and questions that concern the fundamental components of the suite. Part 2 contains answers to questions concerning common applications that depend on the TCP/IP suite for their network connectivity.

Comments on this document can be emailed to the FAQ maintainer at <tcp-ip-faq@eng.sun.com>.

FAQ Part 1: Introduction and Fundamental Protocols

Administrivia

1. Where can I find an up-to-date copy of this FAQ?
2. Who wrote this FAQ?

About TCP/IP

1. What is TCP/IP?
2. How is TCP/IP defined?
3. Where can I find RFC's?
4. How do I find the right RFC?

About IP

1. What is IP?
2. How is IP carried on a network?

About TCP

About UDP

About ICMP

TCP/IP Network Operations

TCP/IP Protocol Implementations

Further Sources of Information

FAQ Part 2 -- Applications and Application Programming

What Are The Common TCP/IP Application Protocols?

3. FTP
4. HTTP
5. IMAP
6. NFS
7. NNTP
8. NTP
9. POP
10. Rlogin
11. Rsh
12. SMTP
13. SNMP
14. Ssh
15. Telnet
16. X Window System

TCP/IP Programming

1. What are sockets?
2. How can I detect that the other end of a TCP connection has crashed?
3. Can TCP keepalive timeouts be configured?
4. Are there object-oriented network programming tools?

This compilation contains the opinions of the FAQ maintainer and the various FAQ contributors. Any resemblance to the opinions of the FAQ maintainer's employer is entirely coincidental.

Secure UNIX Programming FAQ

The FAQ is maintained by Thamer Al-Herbish send comments to: sup@whitefang.com.

The FAQ material:

- The FAQ in text.
- The FAQ in HTML.

Here's a work in progress version. Here's a change log.

I've setup a mailing list for the discussion of secure UNIX programming. Take a look at the announcement made on comp.security.unix

Mirrors

Mirrors are updated weekly.

- JPL-TRS Mirror: http://jpl-trs.jpl.nasa.gov/secure/
- Canadian Mirror: http://www.codeweb.net/sup/
- Swedish Mirror: http://www.centus.com/sup/
- Bulgarian Mirror: http://sep.daemonz.org/sup/
- Armenian Mirror: http://www.daemon.am/Secur/FAQ
- Brazilian Mirror: http://www.ddsecurity.com.br/~grios/unix/

Related FAQs

- The UNIX Programming FAQ:
  - Homesite in the UK
  - US Mirror
- The Socket Programming FAQ:
  - Site in the UK
  - US Mirror
- The TCP/IP FAQ:
  - http://www.itprc.com/tcpipfaq/default.htm
- The Raw IP Networking FAQ:
  - http://www.whitefang.com/rin/

Brazilian readers should know that Gustavo Rios has setup a Brazilian Secure UNIX Programming mailing list.

---

shadows@whitefang.com

Raw IP Networking Mailing List

The mailing list is setup to act as a forum for the discussion of Raw IP Networking. Everything from raw sockets to packet capturing discussion is welcome. Please do check the Raw IP Networking FAQ before posting.

Mailing List Information

To subscribe send an empty message to rawip-subscribe@whitefang.com and reply to the confirmation message.

To unsubscribe send an empty message to rawip-unsubscribe@whitefang.com

Thanks. See you there.

libnet NOW

Libnet is a collection of routines to help with the construction and handling of network packets. It provides a portable framework for low-level network packet shaping, handling and injection. Libnet features portable packet creation interfaces at the IP layer and link layer, as well as a host of supplementary and complementary functionality. Using libnet, quick and simple packet assembly applications can be whipped up with little effort. With a bit more time, more complex programs can be written (Traceroute and ping were easily rewritten using libnet and libpcap). See for yourself how easy it is. The current version is 1.0.2a.

Libnet was designed and is primarily maintained by Mike D. Schiffman. Tons of people have helped however.

Check out the SecurityFocus article on libnet 101 written by Mike Schiffman.

The current "stable" Libnet tree is 1.0.x, with the development tree 1.1.x still being, well, developed. 1.1.x will feature a much more intuitive object oriented interface. For information on where we're going with it, check out the mailing list archives. For that matter, join the mailing list.

Page last updated: 02.03.2001.

Frequently answered questions for comp.os.research

The FAQ is composed of an introduction and three parts.

Welcome to comp.os.research

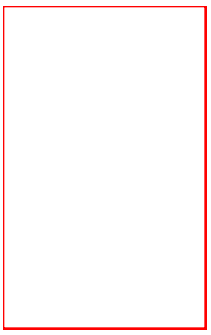
Part 1

Part 2

- Disclaimer and copyright

Part 3

For the textual version of the FAQ, see the archive on rtfm.mit.edu, or below:



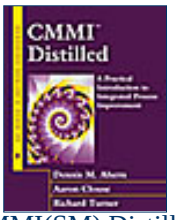
Bryan O'Sullivan, maintainer

Series of the Month

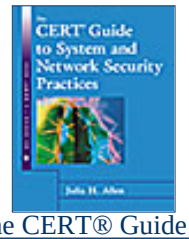
The SEI Series in Software Engineering is the 'Series of the Month' for July.

The SEI Series in Software Engineering represents a collaboration between the Software Engineering Institute of Carnegie Mellon University and Addison-Wesley to develop and publish a body of work on selected topics in software engineering. The common goal of the SEI and Addison-Wesley is to provide the most current software engineering information in a form that is easily usable by practitioners and students.

Managing Software Acquisition
B. Craig Meyers, Patricia Oberndorf
0-201-70454-4



CMMI(SM) Distilled
Dennis M. Ahern, Aaron Clouse, Richard Turner
0-201-73500-8



The CERT® Guide to System and Network Security Practices
Julia H. Allen
0-201-73723-X

View all the books in The SEI Series in Software Engineering

Author Appearances

Check out the appearances that Addison-Wesley authors will be making.

The C++ Seminar: 3 Days with 5 Experts
Scott Meyers, Herb Sutter, Dan Saks, Steve Dewhurst and Andrei Alexandrescu

For three in-depth days, The C++ Seminar brings you together with five of the C++ world's best-known consultants and trainers. Your instructors are authorities on C++ software development and speakers at conferences around the world. Three Addison-Wesley authors will be leading this three day seminar.

Scott Meyers, author of Effective C++, More Effective C++ and Effective STL
Herb Sutter, author of Exceptional C++ and the forthcoming title More Exceptional C++
Andrei Alexandrescu, author of Modern C++ Design

September 17 - 19, 2001
Portland, Oregon

You can join the seminar mailing list via the box on the seminar page or by sending mail to cpp_seminar-subscribe@yahoogroups.com

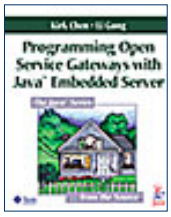
Learn more about The C++ Seminar

Visit the complete list of Addison-Wesley Author Appearances

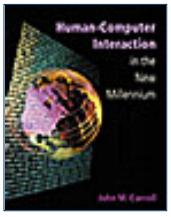
Coming Soon

Check out these titles that will be published soon.

Performance Solutions
Connie U. Smith, Lloyd G. Williams
0-201-72229-1



Programming Open Service Gateways with Java Embedded Server™ Technology
Kirk Chen, Li Gong
0-201-71102-8



Human-Computer Interaction in the New Millennium
John M. Carroll
0-201-70447-1

Metadata Solutions
Adrienne Tannenbaum
0-201-71976-2

Visit the complete list of Addison-Wesley titles Coming Soon

Tradeshows

Check out the Tradeshows Addison-Wesley will be attending.

Rational User Conference 2001

Rational User Conference 2001 - "Blazing New Trails" - is all about finding better ways to negotiate the ever-changing landscape of software development. If software is key to your success, you need to be a part of this extraordinary event. You'll hear about the latest tools, trends and tricks of the trade from Rational's technology leaders, as well as learn from your colleagues how they're blazing new trails of their own with Rational. With the dynamics of software changing at the "speed of business," you can't afford to miss it!

July 22 - 26, 2001
Colorado Convention Center
Denver, Colorado

Addison-Wesley Professional is Booth #218

Learn more about RUC

Visit the complete list of Addison-Wesley Tradeshows

Book Reviews

Check out what others say about Addison-Wesley books.

Modern C++ Design by Andrei Alexandrescu

"In that sense Modern C++Design: Generic Programming and Design Patterns Applied is definitely worth reading. If you want to learn more about templates and C++ and if you want to expand your horizon, go for it. This book will most likely change the way you use and understand C++ templates." **--Devx.com, May 2001**

Visit the complete list of Addison-Wesley Reviews

Mailing List

Stay up-to-date on our new publications by subscribing to our mailing list.

Every month we'll send you information about the latest publications by Addison-Wesley. You can choose to hear about specific topics or about all of the computer and engineering titles.

Sign up for the mailing list


Customer
Feedback

Multithreaded Programming with Windows-NT (Bk/Disk), 1/e

Thuan Q. Pham, Silicon Graphics, Mt. View, CA
Pankaj K. Garg, Hewlett-Packard Laboratories, Palo Alto, CA

Copyright 1996, 256 pp.
Paper Bound with Disk format
ISBN 0-13-120643-5

Course List

Computer Science: Operating Systems (OS)

top

Summary

For application developers who want to experiment with multithreaded programming.

This book details techniques for designing and implementing multithreaded software applications in the Windows NT operating system.

top

Features

- brings multithreading to the Windows NT operating system.
- offers solutions for the problems introduced by multithreading, including resource sharing, deadlocks, and

race conditions.

- describes a video-on-demand application that brings together the cable and movie industries.

top

Table of Contents

top

Addison-Wesley Professional Publishing Group

This page has moved. If you are looking for a specific book, source code, errata or additional book information, please use one of the two following options.

- **Use the Search Engine** A search can be done by title, author's last name or ISBN in the search engine box located at the right top corner of this page.
- **Enter the Book URL** A book can be found by entering its URL in the following format: http://www.awl.com/cseng/titles/I-S-B-N. The book ISBN should be entered with the dashes in the appropriate places. You'll find the ISBN on the back cover of the book by the bar-code.

Multithreaded Programming Education

by Lambda Computer Science

Bil, aloft in San Francisco.

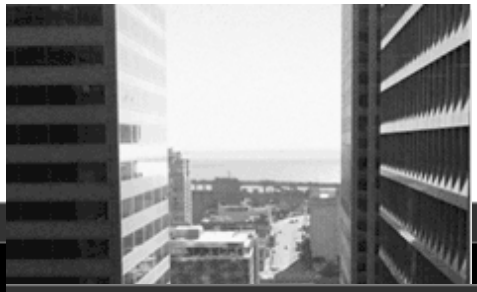
Lambda Computer Science is dedicated to producing the highest quality computer science education classes in areas of our expertise. Currently we teach classes on multithreaded programming and Symbolic Programming. We have been teaching computer science classes since the early 80s, including university classes at Stanford and Penn, and industry classes to numerous companies in the US, Europe, and Asia.

Multithreading Seminars

Multithreading Books

comp.programming.threads newsgroup

Bil's personal page



WORLDWIDE: NEWS: 1.800.ATCOMPAQ

United States

- Compaq Expects to Meet Second Quarter Earnings Consensus
- Listen to Replayed Audio Webcast of Financial Announcement

summary of JavaScript functions on this page

search | contact hp | hp UNITED STATES | select a country or region: United States

country select page for javascript disabled browsers

invent

products & services | support | drivers | hp store

highlights

- hp announces pay-per-use server pricing
- discover the latest solutions at hp world conference & expo
- learn more about PDAs
- white house to nominate hp director to commerce post
- more news...

solutions for..

- home & home office
- small & medium business
- large & corporate business
- government
- education
- IT professionals
- developers
- more solutions...

- company information
- jobs at hp
- investor relations
- how to buy

Home / home office
Small business
Government
Education
Industries

Developers
IBM Business Partners
Jobs at IBM
Investors
Journalists

Buy today, ship today

IBM can ship select PC models to you today when you buy by 3 p.m. EDT, Monday-Friday (U.S. only)

News

- Wimbledon Web site smashes viewing records
- IBM completes acquisition of Informix database assets
- Michelin European distributors can order on Web 24 hours a day
- Shell, IBM form alliance on e-business infrastructure
- More news and newsletters

IBM e-business
Find out about e-business Infrastructure

e-business hosting
From secure hosting to custom services

Software trials & betas
Discover the latest in IBM software

→ **Ready to buy?**
Find it and buy it fast!

→ **Special / advertised offers**

NavCode™
Enter NavCode here: Details

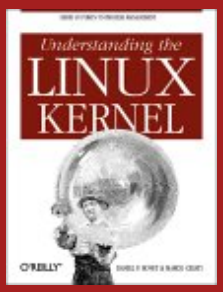
Go

Search this site

Search

Applications
Documentation
Distributions
Hardware
General Info
Courses
Projects
News

Book Store
People
User Groups
Vendors
Search
Events
About Us
Home Page



What is Linux?

Linux is a free Unix-type operating system originally created by Linus Torvalds with the assistance of developers around the world. Developed under the GNU General Public License , the source code for Linux is freely available to everyone. Click on the link below to find out more about the operating system that is causing a revolution in the world of computers.

more

Linux in the News

- Why Caldera's Linux business plan set off open-source wars
- Red Hat rival shifts pricing plan
- Red Hat releases its Linux for Itanium
- Linux supercomputer seller nets $5 million
- Mono to open source .NET by mid 2002

more

People of Linux

June 15 saw the release of SuSE Linux 7.2. and this week, Linux Online is pleased to offer you this interview with Mr. Heiner Maasjost, Vice President for Marketing at SuSE. We talked to Mr. Maasjost about this release, his opinion on the desktop issue and Microsoft's practices, among other topics.

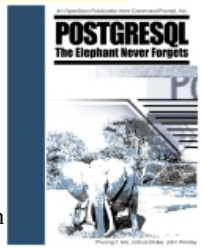
more

Linux 101

Are you thinking about switching to Linux and want to learn how to use it? Have you been using Linux for some time and want to learn even more? Then Linux Online's classroom can help! A beginner's course is online now, with more classes being added all the time.

more

Featured Books



PostgreSQL is the most popular open source relational database for Linux. Authors Phuong Y. Ma, John C. Worsley and Joshua D. Drake have come up with the definitive guide for this widely used database server for the Linux platform: **PostgreSQL: The elephant never forgets**. You'll learn how to use, administer and program with PostgreSQL. Businesses of any size will benefit from this book as they will be able to run a relational database management system for a fraction of the cost of commercial offerings. A CD with the latest version of PostgreSQL as well as other useful packages, such as the Apache web server and PHP comes with the book.

more

Top Five Linux Books:

- PHP and MySQL Web Development
- Programming Perl (3rd Edition)
- SSH, the Secure Shell : The Definitive Guide
- Learning Perl (2nd Edition)
- Data Munging with Perl

more

Opinion

What do George III, Louis XVI and Microsoft have in common? In this month when we commemorate two famous revolutions, this week's editorial talks about the recent Microsoft anti-trust decision and what it means for Linux and the computer revolution it started and the old regime it pretends to free us from.

more

Buy a Mug!

You can have one of these exclusive coffee mugs made especially for Linux Online. By buying one or as many mugs as you'd like, you'll be helping us to bring you the most up-to-date and comprehensive Linux info on the Internet. Order today and drink a toast to the future of Linux!

more

URLWatch: Submit

For notice when this page changes, fill in your email address.

Maintained by: Webmaster, Linux Online Inc.
Last modified: 04-Jul-2001 11:10AM.
Views since 16-Aug-2000: 10144017.

Search

GO

Advanced Search

Product Families

Windows

Office

Servers

Developer Tools

Other Products

Information For

Home/Home Office

Businesses

IT Professionals

Developers

Microsoft Partners

Educators

Investors

Journalists

Resources

Support

Windows Update

Office Tools

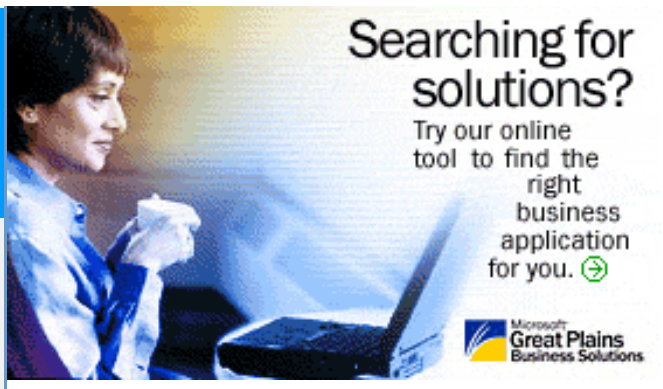
bCentral

Shop

Books

Jobs

Freedom to Innovate



Test drive .NET.
Get the Visual Studio .NET Beta, and build XML Web services now.

Exchange 2000 Service Pack 1.
Get the update that makes Exchange more reliable than ever.

Deploy Active Directory.
Get technical help with this core piece of Windows 2000 Server.

$50 rebate.
Save on Office XP when you order by July 31. U.S. and Canada only.

Last Updated: Sunday, July 8, 2001 - 11:40 p.m. Pacific Time

Today's News

- Experience the best of the digital world.
  Order the latest Windows XP Preview today.
- Orchestrate your business solutions
  with the latest BizTalk Server tips and tricks.
- Build your .NET skills
  with intense step-by-step developer training. U.S. only.

More News

New Downloads

- MSN Messenger.
  Stay in touch with instant messages, teleconferencing, voice chat, and more.
- Internet Explorer 6 Preview Refresh.
  Beta test the latest in privacy, reliability, and multimedia. Updated.
- Media Player 7.1:
  Get the newest video and audio players for PC and Pocket PC.

More Downloads

TODAY'S HIGHLIGHTS

- Final Fantasy: The Spirits Within Powered by SGI
- Announcing the Port of ESRI ArcIMS and ArcSDE for Use with Oracle on SGI Irix
- Introducing the New Silicon Graphics F180 Flat Panel Display
- SGI Releases Preliminary Fourth-Quarter Results
- Mercedes Momentum: Dream Factory

NEWSROOM SUBSCRIBE

Fast Find GO

CONFIGURE A SYSTEM

RATE THIS PAGE

manufacturing
sciences
telecommunications and media

dev products
programs
library
resources
operating sys
open source

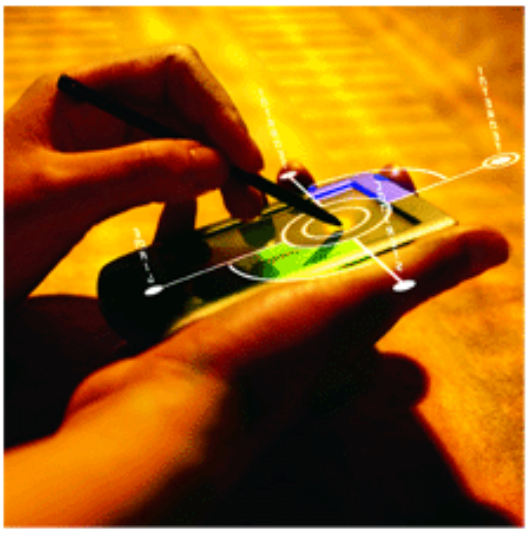
professional services
productivity services
support services
education services
online support

games
web dev
software dev
graphics
system admin
freeware

SGI Search

All of SGI

Search



Get Smart

Sun™ ONE brings mobile intelligence to the wireless world.

Customer-Proven Portal Solutions

RESOURCES FOR:
- DEVELOPERS
- SERVICE PROVIDERS
- SYSTEM ADMINS
- EXECUTIVES
- INVESTORS

SUN PRESENTS

ONLINE DISCUSSION: Jiro[tm] Technology and Federated Management Architecture July 24-27, 2001

Join us for a special discussion forum with Paul Monday, coauthor of The Jiro[tm] Technology Programmer's Guide and Federated Management Architecture.

WEBCAST: JavaOne[sm] Conference June 4 - 8, 2001

See how Sun Microsystems and the developer community are utilizing Java[tm] technology to redefine the nature and value of the network at the 2001 JavaOne[sm] conference keynote sessions.

DIGITAL JOURNEY: The Endangered Music Project

Digital technologies help preserve and disperse vast repositories of original ethnographic recordings that are threatened by war, political upheaval, natural disaster, and the influence of modern society.

More»

The ADAPTIVE Communication Environment (ACE™)

An OO Network Programming Toolkit in C++

ACE Site Map

1. Draft chapters from volume one of our C++ Network Programming book series
2. Pattern-Oriented Software Architecture: Patterns for Concurrent and Networked Objects

- ACE Overview
- Obtaining ACE
- Building ACE
- ACE Documents
- ACE Copyright

- TAO Overview
- ZEN Overview
- Java ACE Overview
- JAWS Overview

- ACE team members
- Who is Using ACE
- Commercial support
- ACE Mailing List
- ACE Sponsors

Back to Douglas C. Schmidt's home page.

Last modified 08:49:47 CDT 01 July 2001

Document not found

We're sorry, we cannot find the following document on our server. Please check the URL that you entered.

The URL that does not resolve to a valid document is:

http://www.developer.ibm.com/library/ref/about4.1/df4threa.html

Here are some other links that might help you:

- PartnerWorld for Developers home page
- IBM Software home Page
- IBM home page

We apologize for any inconvenience this may have caused you.

Bibliography on threads and multithreading

[About | Browse | Statistics]

Number of references: 610
Number of online publications: 224
Most recent reference: August 1995
Last update: July 11, 1995
Supported: no

Search the Bibliography

Query: Search

Options: case insensitive | partial word(s) | online papers only

Results: Citation | Maximum of 40 matches

compress results | Reset

Help on: [Syntax | Options | Compression of results | Improving your query | Query examples]
Boolean operators: **and** and **or**. Use **()** to group boolean subexpressions.
Example: `(specification or verification) and asynchronous`

Information on the Bibliography

Author:

Torsten Amundsen <torstena@ifi.uio.no>
Department of Informatics
University of Oslo
Norway

Browsing the bibliography

- **Original source:**

  Used to be available at
  http://www.ifi.uio.no/~torstena/threads.bib.gz
  but has gone away.

- Local copy in BibTeX format (220 KB, compressed with gzip)

Bibliographic Statistics

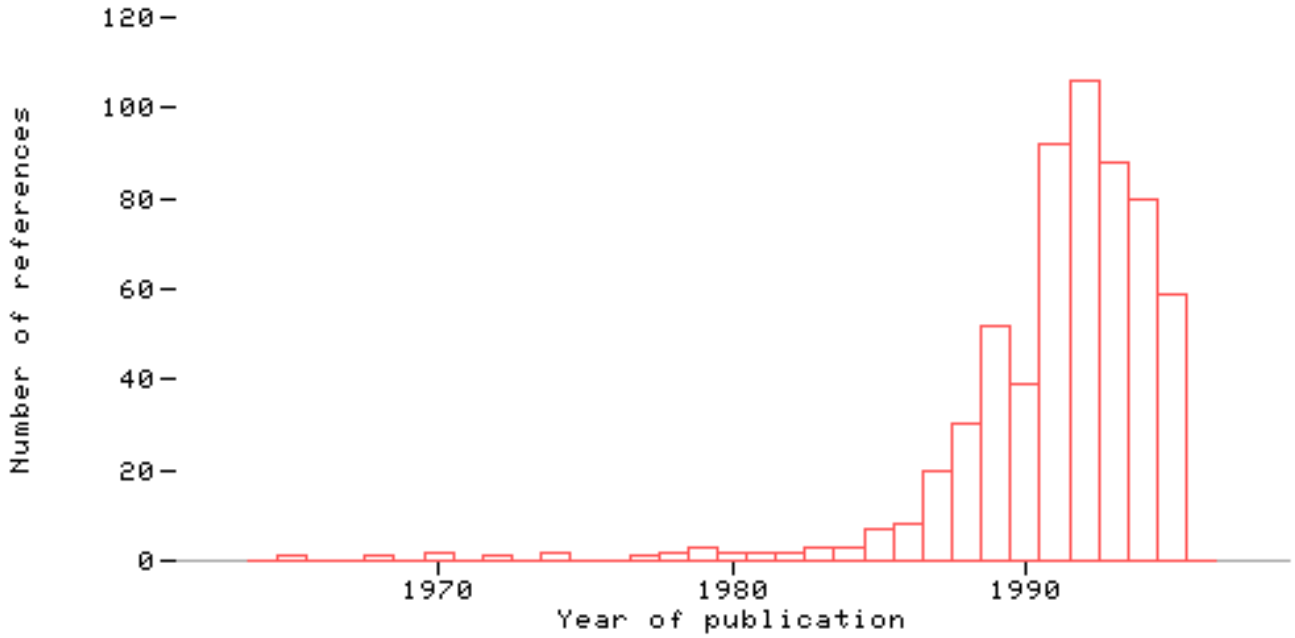
Types:

inproceedings(337), techreport(107), article(92), phdthesis(29), mastersthesis(19), unpublished(9), book(5), inbook(4), misc(4), incollection(3), manual(1)

Fields:

title(610), year(610), author(608), month(468), abstract(462), pages(429), booktitle(340), url(219), note2(190), note(189), number(187), address(164), volume(124), institution(108), journal(92), publisher(73), school(48), editor(26), series(24), available(9), organization(9), type(7), chapter(4), availabe(3), howpublished(3), url2(3), available2(2), abstract2(1), addresse(1), adresse(1), conflocation(1), day(1), notes(1)

Distribution of publication dates:



Getting Started With POSIX Threads

Tom Wagner
Don Towsley
Department of Computer Science
University of Massachusetts at Amherst

Contents:

Available for Distribution:

- Semaphore Library Source
- Tutorial as Postscript
- Uncompressed Tutorial as Postscript

Questions and comments should be directed to wagner@cs.umass.edu.

Special: Readers' choice awards

Document not found

We're sorry, we cannot find the following document on our server. Please check the URL that you entered.

The URL that does not resolve to a valid document is:

http://www.developer.ibm.com/library/aixpert/nov94/aixpert_nov94_intrmult.html

Here are some other links that might help you:

- PartnerWorld for Developers home page
- IBM Software home Page
- IBM home page

We apologize for any inconvenience this may have caused you.

Select a country

IBM PartnerWorld ®

PartnerWorld for Developers

Marketing & sales

Education

Technical support

Incentives & business support

Financing

Membership center

Community

News & events

Library

Programmer's information

White papers

Magazines

Books

Developer briefs

e-business

Servers

Software

Technologies

Feedback

Site map & help

Search PartnerWorld for Developers:

Local

Go

Related links:

developerWorks

alphaWorks

PartnerWorld for:

PartnerWorld for Developers Library

*e-developer*EDGE

After 10 years as **AIXpert**, we have renamed the magazine to reflect its content as we broaden its focus from AIX and RS/6000 to include other UNIX® related topics such as NUMA-Q, Project Monterey, Java technology, and e-business. ***e-developer*EDGE** is distributed via CD-ROM to commercial members of PartnerWorld for Developers.

Members can access the latest issue of ***e-developer*EDGE**, as well as the complete online collection. Recent issues (still named AIXpert) are available to all. The current level of Adobe Acrobat Reader is recommended for accessing these issues.

Get Acrobat Reader

Recent issues

December 1999	*Focus on AIX WebSphere Application Server*
July 1999	*Focus on AIX Performance Tuning*
March 1999	*Focus on AIX Packaging*
December 1998	*Focus on XML*
September 1998	*Focus on 3-D Graphics*
June 1998	*Focus on Client/Server Programs*
March 1998	*Focus on Internet Technologies*
December 1997	*Focus on Parallel Programming*
September 1997	*Focus on 64-bit Architecture*
June 1997	*Focus on Network Computing*
March 1997	*Focus on Internet at the Olympics*

Software
Systems & Services
Personal Systems

Document not found

We're sorry, we cannot find the following document on our server. Please check the URL that you entered.

The URL that does not resolve to a valid document is:

http://www.developer.ibm.com/library/aixpert/aug94/aixpert_aug94_PTHREADS.html

Here are some other links that might help you:

- PartnerWorld for Developers home page
- IBM Software home Page
- IBM home page

We apologize for any inconvenience this may have caused you.

Document not found

We're sorry, we cannot find the following document on our server. Please check the URL that you entered.

The URL that does not resolve to a valid document is:

http://www.developer.ibm.com/library/aixpert/aug95/aixpert_aug95_thread.html

Here are some other links that might help you:

- PartnerWorld for Developers home page
- IBM Software home Page
- IBM home page

We apologize for any inconvenience this may have caused you.

Document not found

We're sorry, we cannot find the following document on our server. Please check the URL that you entered.

The URL that does not resolve to a valid document is:

http://www.developer.ibm.com/library/aixpert/aug95/aixpert_aug95_signal.html

Here are some other links that might help you:

- PartnerWorld for Developers home page
- IBM Software home Page
- IBM home page

We apologize for any inconvenience this may have caused you.

Linuxthreads - POSIX 1003.1c kernel threads for Linux

DESCRIPTION:

This is release 0.7 (late beta) of LinuxThreads, a BiCapitalized
implementation of the Posix 1003.1c "pthread" interface for Linux.

LinuxThreads provides kernel-level threads: each thread is a separate
Unix process, sharing its address space with the other threads through
the new system call clone(). Scheduling between threads is handled by
the kernel scheduler, just like scheduling between Unix processes.

REQUIREMENTS:

- Linux version 2.0 and up (requires the new clone() system call
  and the new realtime scheduler).

- For Intel platforms: libc 5.2.18 or later is required.
  5.2.18 or 5.4.12 or later are recommended;
  5.3.12 and 5.4.7 have problems (see the FAQ.html file for more info).

- Also supports glibc 2 (a.k.a. libc 6), which actually comes with
  a specially-adapted version of this library.

- Currently supports Intel, Alpha, Sparc, Motorola 68k and MIPS platforms.

- Multiprocessors are supported.

INSTALLATION:

- Edit the Makefile, set the variables in the "Configuration" section.

- Do "make".

- Do "make install".

USING LINUXTHREADS:

```
gcc -D_REENTRANT ... -lpthread
```

A complete set of manual pages is included. Also see the subdirectory
Examples/ for some sample programs.

STATUS:

- All functions in the Posix 1003.1c base interface implemented.
  Also supports priority scheduling.

- For users of libc 5 (H.J.Lu's libc), a number of C library functions
  are reimplemented or wrapped to make them thread-safe, including:
  * malloc functions
  * stdio functions (define _REENTRANT before including <stdio.h>)
  * per-thread errno variable (define _REENTRANT before including <errno.h>)

  * directory reading functions (opendir(), etc)
  * sleep()
  * gmtime(), localtime()

  New library functions provided:
  * flockfile(), funlockfile(), ftrylockfile()
  * reentrant versions of network database functions (gethostbyname_r(), etc)
    and password functions (getpwnam_r(), etc).

- libc 6 (glibc 2) provides much better thread support than libc 5,
  and comes with a specially-adapted version of LinuxThreads.
  For serious multithreaded programming, you should consider switching
  to glibc 2. It is available from prep.ai.mit.edu:/pub/gnu and its mirrors.

WARNING:

Many existing libraries are not compatible with LinuxThreads,
either because they are not inherently thread-safe, or because they
have not been compiled with the -D_REENTRANT. For more info, see the
FAQ.html file in this directory.

A prime example of the latter is Xlib. If you link it with
LinuxThreads, you'll probably get an "unknown 0 error" very
early. This is just a consequence of the Xlib binaries using the
global variable "errno" to fetch error codes, while LinuxThreads and
the C library use the per-thread "errno" location.

See the file README.Xfree3.3 for info on how to compile the Xfree 3.3
libraries to make them compatible with LinuxThreads.

KNOWN BUGS AND LIMITATIONS:

- Threads share pretty much everything they should share according
  to the standard: memory space, file descriptors, signal handlers,
  current working directory, etc. One thing that they do not share
  is their pid's and parent pid's. According to the standard, they
  should have the same, but that's one thing we cannot achieve
  in this implementation (until the CLONE_PID flag to clone() becomes
  usable).

- The current implementation uses the two signals SIGUSR1 and SIGUSR2,
  so user-level code cannot employ them. Ideally, there should be two
  signals reserved for this library. One signal is used for restarting
  threads blocked on mutexes or conditions; the other is for thread
  cancellation.

- The stacks for the threads are allocated high in the memory space,
  below the stack of the initial process, and spaced 2M apart.
  Stacks are allocated with the "grow on demand" flag, so they don't
  use much virtual space initially (4k, currently), but can grow
  up to 2M if needed.

  Reserving such a large address space for each thread means that,
  on a 32-bit architecture, no more than about 1000 threads can
  coexist (assuming a 2Gb address space for user processes),
  but this is reasonable, since each thread uses up one entry in the
  kernel's process table, which is usually limited to 512 processes.

Another potential problem of the "grow on demand" scheme is that nothing prevents the user from mmap'ing something in the 2M address window reserved for a thread stack, possibly causing later extensions of that stack to fail. Mapping at fixed addresses should be avoided when using this library.

- Signal handling does not fully conform to the Posix standard, due to the fact that threads are here distinct processes that can be sent signals individually, so there's no notion of sending a signal to "the" process (the collection of all threads).
  More precisely, here is a summary of the standard requirements and how they are met by the implementation:

  1- Synchronous signals (generated by the thread execution, e.g. SIGFPE) are delivered to the thread that raised them.
  (OK.)

  2- A fatal asynchronous signal terminates all threads in the process.
  (OK. The thread manager notices when a thread dies on a signal and kills all other threads with the same signal.)

  3- An asynchronous signal will be delivered to one of the threads of the program which does not block the signal (it is unspecified which).
  (No, the signal is delivered to the thread it's been sent to, based on the pid of the thread. If that thread is currently blocking the signal, the signal remains pending.)

  4- The signal will be delivered to at most one thread.
  (OK, except for signals generated from the terminal or sent to the process group, which will be delivered to all threads.)

- The current implementations of the MIPS support assumes a MIPS ISA II processor or better. These processors support atomic operations by ll/sc instructions. Older R2000/R3000 series processors are not supported yet; support for these will have higher overhead.

LinuxThreads Frequently Asked Questions (with answers)

[For LinuxThreads version 0.8]

A. The big picture

A.1: What is LinuxThreads?

LinuxThreads is a Linux library for multi-threaded programming. It implements the Posix 1003.1c API (Application Programming Interface) for threads. It runs on any Linux system with kernel 2.0.0 or more recent, and a suitable C library (see section C).

A.2: What are threads?

A thread is a sequential flow of control through a program. Multi-threaded programming is, thus, a form of parallel programming where several threads of control are executing concurrently in the program. All threads execute in the same memory space, and can therefore work concurrently on shared data.

Multi-threaded programming differs from using multiple Unix processes in that all threads share the same memory space (and a few other system resources, such as file descriptors), instead of running in their own memory space as is the case with Unix processes.

Threads are useful for several reasons. First, they allow a program to exploit multi-processor machines: the threads can run in parallel on several processors, allowing a single program to divide its work between several processors, thus running faster than a single-threaded program, which runs on only one processor at a time. Second, even on uniprocessor machines, threads allow overlapping I/O and computations in a simple way. Last, some programs are best expressed as several threads of control that communicate together, rather than as one big monolithic sequential program. Examples include server programs, overlapping asynchronous I/O, and graphical user interfaces.

A.3: What is POSIX 1003.1c?

It's an API for multi-threaded programming standardized by IEEE as part of the POSIX standards. Most Unix vendors have endorsed the POSIX 1003.1c standard. Implementations of the 1003.1c API are already available under Sun Solaris 2.5, Digital Unix 4.0, Silicon Graphics IRIX 6, and should soon be available from other vendors

such as IBM and HP. More generally, the 1003.1c API is replacing relatively quickly the proprietary threads library that were developed previously under Unix, such as Mach cthreads, Solaris threads, and IRIX sprocs. Thus, multithreaded programs using the 1003.1c API are likely to run unchanged on a wide variety of Unix platforms.

A.4: What is the status of LinuxThreads?

LinuxThreads implements almost all of Posix 1003.1c, as well as a few extensions. The only part of LinuxThreads that does not conform yet to Posix is signal handling (see section J). Apart from the signal stuff, all the Posix 1003.1c base functionality, as well as a number of optional extensions, are provided and conform to the standard (to the best of my knowledge). The signal stuff is hard to get right, at least without special kernel support, and while I'm definitely looking at ways to implement the Posix behavior for signals, this might take a long time before it's completed.

A.5: How stable is LinuxThreads?

The basic functionality (thread creation and termination, mutexes, conditions, semaphores) is very stable. Several industrial-strength programs, such as the AOL multithreaded Web server, use LinuxThreads and seem quite happy about it. There used to be some rough edges in the LinuxThreads / C library interface with libc 5, but glibc 2 fixes all of those problems and is now the standard C library on major Linux distributions (see section C).

B. Getting more information

B.1: What are good books and other sources of information on POSIX threads?

The FAQ for comp.programming.threads lists several books: http://www.serpentine.com/~bos/threads-faq/.

There are also some online tutorials. Follow the links from the LinuxThreads web page: http://pauillac.inria.fr/~xleroy/linuxthreads.

B.2: I'd like to be informed of future developments on LinuxThreads. Is there a mailing list for this purpose?

I post LinuxThreads-related announcements on the newsgroup comp.os.linux.announce, and also on the mailing list `linux-threads@magenet.com`. You can subscribe to the latter by writing majordomo@magenet.com.

B.3: What are good places for discussing LinuxThreads?

For questions about programming with POSIX threads in general, use the newsgroup comp.programming.threads. Be sure you read the FAQ for this group before you post.

For Linux-specific questions, use comp.os.linux.development.apps and comp.os.linux.development.kernel. The latter is especially appropriate for questions relative to the interface between the kernel and LinuxThreads.

B.4: How should I report a possible bug in LinuxThreads?

If you're using glibc 2, the best way by far is to use the `glibcbug` script to mail a bug report to the glibc maintainers.

If you're using an older libc, or don't have the `glibcbug` script on your machine, then e-mail me directly (`Xavier.Leroy@inria.fr`).

In both cases, before sending the bug report, make sure that it is not addressed already in this FAQ. Also, try to send a short program that reproduces the weird behavior you observed.

B.5: I'd like to read the POSIX 1003.1c standard. Is it available online?

Unfortunately, no. POSIX standards are copyrighted by IEEE, and IEEE does not distribute them freely. You can buy paper copies from IEEE, but the price is fairly high ($120 or so). If you disagree with this policy and you're an IEEE member, be sure to let them know.

On the other hand, you probably don't want to read the standard. It's very hard to read, written in standard-ese, and targeted to implementors who already know threads inside-out. A good book on POSIX threads provides the same information in a much more readable form. I can personally recommend Dave Butenhof's book, Programming with POSIX threads (Addison-Wesley). Butenhof was part of the POSIX committee and also designed the Digital Unix implementations of POSIX threads, and it shows.

Another good source of information is the X/Open Group Single Unix specification which is available both on-line and as a book and CD/ROM. That specification includes pretty much all the POSIX standards, including 1003.1c, with some extensions and clarifications.

C. Issues related to the C library

C.1: Which version of the C library should I use with LinuxThreads?

The best choice by far is glibc 2, a.k.a. libc 6. It offers very good support for multi-threading, and LinuxThreads has been closely integrated with glibc 2. The glibc 2 distribution contains the sources of a specially adapted version of LinuxThreads.

glibc 2 comes preinstalled as the default C library on most Linux distributions nowadays, such as RedHat 5 and up, and Debian 2 and up. Those distributions include the version of LinuxThreads matching glibc 2.

C.2: My system has libc 5 preinstalled, not glibc 2. Can I still use LinuxThreads?

Yes, but you're likely to run into some problems, as libc 5 only offers minimal support for threads and contains some bugs that affect multithreaded programs.

The versions of libc 5 that work best with LinuxThreads are libc 5.2.18 on the one hand, and libc 5.4.12 or later on the other hand. Avoid 5.3.12 and 5.4.7: these have problems with the per-thread errno variable.

C.3: So, should I switch to glibc 2, or stay with a recent libc 5?

I'd recommend you switch to glibc 2. Even for single-threaded programs, glibc 2 is more solid and more standard-conformant than libc 5. And the shortcomings of libc 5 almost preclude any serious multi-threaded programming.

Switching an already installed system from libc 5 to glibc 2 is not completely straightforward. See the Glibc2 HOWTO for more information. Much easier is (re-)installing a Linux distribution based on glibc 2, such as RedHat 6.

C.4: Where can I find glibc 2 and the version of LinuxThreads that goes with it?

Both `glibc` 2 and the associated LinuxThreads distribution can be found on Cygnus' Sourceware collection, and also on any FTP site that mirrors GNU software (the Cygnus site is sometimes more up-to-date than the GNU sites).

C.5: Where can I find libc 5 and the version of LinuxThreads that goes with it?

For libc 5, see `ftp://sunsite.unc.edu/pub/Linux/devel/GCC/`.

For the libc 5 version of LinuxThreads, see ftp://ftp.inria.fr/INRIA/Projects/cristal/Xavier.Leroy/linuxthreads/.

C.6: How can I recompile the glibc 2 version of the LinuxThreads sources?

You must transfer the whole glibc sources, then drop the LinuxThreads sources in the `linuxthreads/` subdirectory, then recompile glibc as a whole. There are now too many inter-dependencies between LinuxThreads and glibc 2 to allow separate re-compilation of LinuxThreads.

C.7: What is the correspondence between LinuxThreads version numbers, libc version numbers, and RedHat version numbers?

Here is a summary. (Information on Linux distributions other than RedHat are welcome.)

LinuxThreads	C library	RedHat
0.7, 0.71 (for libc 5)	libc 5.x	RH 4.2
0.7, 0.71 (for glibc 2)	glibc 2.0.x	RH 5.x
0.8	glibc 2.1.1	RH 6.0
0.8	glibc 2.1.2	RH 6.1 and 6.2

D. Problems, weird behaviors, potential bugs

D.1: When I compile LinuxThreads, I run into problems in file `libc_r/dirent.c`

You probably mean:

```
libc_r/dirent.c:94: structure has no member named `dd_lock'
```

I haven't actually seen this problem, but several users reported it. My understanding is that something is wrong in the include files of your Linux installation (`/usr/include/*`). Make sure you're using a supported version of the libc 5 library. (See question C.2).

D.2: When I compile LinuxThreads, I run into problems with `/usr/include/sched.h`: there are several occurrences of _p that the C compiler does not understand

Yes, `/usr/include/sched.h` that comes with libc 5.3.12 is broken. Replace it with the `sched.h` file contained in the LinuxThreads distribution. But really you should not be using libc 5.3.12 with LinuxThreads! (See question C.1.)

D.3: My program does `fdopen()` on a file descriptor opened on a pipe. When I link it with LinuxThreads, `fdopen()` always returns NULL!

You're using one of the buggy versions of libc (5.3.12, 5.4.7., etc). See question C.1 above.

D.4: My program creates a lot of threads, and after a while `pthread_create()` no longer returns!

This is known bug in the version of LinuxThreads that comes with glibc 2.1.1. An upgrade to 2.1.2 is recommended.

D.5: When I'm running a program that creates N threads, `top` or `ps` display N+2 processes that are running my program. What do all these processes correspond to?

Due to the general "one process per thread" model, there's one process for the initial thread and N processes for the

threads it created using `pthread_create`. That leaves one process unaccounted for. That extra process corresponds to the "thread manager" thread, a thread created internally by LinuxThreads to handle thread creation and thread termination. This extra thread is asleep most of the time.

D.6: Scheduling seems to be very unfair when there is strong contention on a mutex: instead of giving the mutex to each thread in turn, it seems that it's almost always the same thread that gets the mutex. Isn't this completely broken behavior?

That behavior has mostly disappeared in recent releases of LinuxThreads (version 0.8 and up). It was fairly common in older releases, though. What happens in LinuxThreads 0.7 and before is the following: when a thread unlocks a mutex, all other threads that were waiting on the mutex are sent a signal which makes them runnable. However, the kernel scheduler may or may not restart them immediately. If the thread that unlocked the mutex tries to lock it again immediately afterwards, it is likely that it will succeed, because the threads haven't yet restarted. This results in an apparently very unfair behavior, when the same thread repeatedly locks and unlocks the mutex, while other threads can't lock the mutex.

In LinuxThreads 0.8 and up, `pthread_unlock` restarts only one waiting thread, and pre-assign the mutex to that thread. Hence, if the thread that unlocked the mutex tries to lock it again immediately, it will block until other waiting threads have had a chance to lock and unlock the mutex. This results in much fairer scheduling.

Notice however that even the old "unfair" behavior is perfectly acceptable with respect to the POSIX standard: for the default scheduling policy, POSIX makes no guarantees of fairness, such as "the thread waiting for the mutex for the longest time always acquires it first". Properly written multithreaded code avoids that kind of heavy contention on mutexes, and does not run into fairness problems. If you need scheduling guarantees, you should consider using the real-time scheduling policies `SCHED_RR` and `SCHED_FIFO`, which have precisely defined scheduling behaviors.

D.7: I have a simple test program with two threads that do nothing but `printf()` in tight loops, and from the printout it seems that only one thread is running, the other doesn't print anything!

Again, this behavior is characteristic of old releases of LinuxThreads (0.7 and before); more recent versions (0.8 and up) should not exhibit this behavior.

The reason for this behavior is explained in question D.6 above: `printf()` performs locking on `stdout`, and thus your two threads contend very heavily for the mutex associated with `stdout`. But if you do some real work between two calls to `printf()`, you'll see that scheduling becomes much smoother.

D.8: I've looked at <pthread.h> and there seems to be a gross error in the `pthread_cleanup_push` macro: it opens a block with { but does not close it! Surely you forgot a } at the end of the macro, right?

Nope. That's the way it should be. The closing brace is provided by the `pthread_cleanup_pop` macro. The POSIX standard requires `pthread_cleanup_push` and `pthread_cleanup_pop` to be used in matching pairs, at the same level of brace nesting. This allows `pthread_cleanup_push` to open a block in order to stack-allocate some data structure, and `pthread_cleanup_pop` to close that block. It's ugly, but it's the standard way of implementing cleanup handlers.

D.9: I tried to use real-time threads and my program loops like crazy and freezes the whole machine!

Versions of LinuxThreads prior to 0.8 are susceptible to ``livelocks" (one thread loops, consuming 100% of the CPU time) in conjunction with real-time scheduling. Since real-time threads and processes have higher priority than normal Linux processes, all other processes on the machine, including the shell, the X server, etc, cannot run at all and the machine appears frozen.

The problem is fixed in LinuxThreads 0.8.

D.10: My application needs to create thousands of threads, or maybe even more. Can I do this with LinuxThreads?

No. You're going to run into several hard limits:

- Each thread, from the kernel's standpoint, is one process. Stock Linux kernels (version 2.2 and earlier) are limited to at most 512 processes for the super-user, and half this number for regular users. This can be changed by changing `NR_TASKS` in `include/linux/tasks.h` and recompiling the kernel. On the x86 processors at least, architectural constraints seem to limit `NR_TASKS` to 4090 at most. (It seems that 2.4 kernels have higher limits, though.)
- LinuxThreads contains a table of all active threads. This table has room for 1024 threads at most. To increase this limit, you must change `PTHREAD_THREADS_MAX` in the LinuxThreads/glibc sources and recompile.
- By default, each thread reserves 2M of virtual memory space for its stack. This space is just reserved; actual memory is allocated for the stack on demand. But still, on a 32-bit processor, the total virtual memory space available for the stacks is on the order of 1G, meaning that more than 500 threads will have a hard time fitting in. You can overcome this limitation by moving to a 64-bit platform, or by allocating smaller stacks yourself using the `setstackaddr` attribute.
- Finally, the Linux kernel contains many algorithms that run in time proportional to the number of process table entries. Increasing this number drastically will slow down the kernel operations noticeably.

(Other POSIX threads libraries have similar limitations, by the way.) For all these reasons, you'd better restructure your application so that it doesn't need more than, say, 100 threads. For instance, in the case of a multithreaded server, instead of creating a new thread for each connection, maintain a fixed-size pool of worker threads that pick incoming connection requests from a queue.

E. Missing functions, wrong types, etc

E.1: Where is `pthread_yield()` ? How comes LinuxThreads does not implement it?

Because it's not part of the (final) POSIX 1003.1c standard. Several drafts of the standard contained `pthread_yield()`, but then the POSIX guys discovered it was redundant with `sched_yield()` and dropped it. So, just use `sched_yield()` instead.

E.2: I've found some type errors in `<pthread.h>`. For instance, the second argument to `pthread_create()` should be a `pthread_attr_t`, not a `pthread_attr_t *`. Also, didn't you forget to declare `pthread_attr_default`?

No, I didn't. What you're describing is draft 4 of the POSIX standard, which is used in OSF DCE threads. LinuxThreads conforms to the final standard. Even though the functions have the same names as in draft 4 and DCE, their calling conventions are slightly different. In particular, attributes are passed by reference, not by value, and default attributes are denoted by the NULL pointer. Since draft 4/DCE will eventually disappear, you'd better port your program to use the standard interface.

E.3: I'm porting an application from Solaris and I have to rename all thread functions from `thr_blah` to `pthread_blah`. This is very annoying. Why did you change all the function names?

POSIX did it. The `thr_*` functions correspond to Solaris threads, an older thread interface that you'll find only under Solaris. The `pthread_*` functions correspond to POSIX threads, an international standard available for many, many platforms. Even Solaris 2.5 and later support the POSIX threads interface. So, do yourself a favor and rewrite your code to use POSIX threads: this way, it will run unchanged under Linux, Solaris, and quite a lot of other platforms.

E.4: How can I suspend and resume a thread from another thread? Solaris has the

`thr_suspend()` and `thr_resume()` functions to do that; why don't you?

The POSIX standard provides **no** mechanism by which a thread A can suspend the execution of another thread B, without cooperation from B. The only way to implement a suspend/restart mechanism is to have B check periodically some global variable for a suspend request and then suspend itself on a condition variable, which another thread can signal later to restart B.

Notice that `thr_suspend()` is inherently dangerous and prone to race conditions. For one thing, there is no control on where the target thread stops: it can very well be stopped in the middle of a critical section, while holding mutexes. Also, there is no guarantee on when the target thread will actually stop. For these reasons, you'd be much better off using mutexes and conditions instead. The only situations that really require the ability to suspend a thread are debuggers and some kind of garbage collectors.

If you really must suspend a thread in LinuxThreads, you can send it a `SIGSTOP` signal with `pthread_kill`. Send `SIGCONT` for restarting it. Beware, this is specific to LinuxThreads and entirely non-portable. Indeed, a truly conforming POSIX threads implementation will stop all threads when one thread receives the `SIGSTOP` signal! One day, LinuxThreads will implement that behavior, and the non-portable hack with `SIGSTOP` won't work anymore.

E.5: Does LinuxThreads implement `pthread_attr_setstacksize()` and `pthread_attr_setstackaddr()`?

These optional functions are provided in recent versions of LinuxThreads (0.8 and up). Earlier releases did not provide these optional components of the POSIX standard.

Even if `pthread_attr_setstacksize()` and `pthread_attr_setstackaddr()` are now provided, we still recommend that you do not use them unless you really have strong reasons for doing so. The default stack allocation strategy for LinuxThreads is nearly optimal: stacks start small (4k) and automatically grow on demand to a fairly large limit (2M). Moreover, there is no portable way to estimate the stack requirements of a thread, so setting the stack size yourself makes your program less reliable and non-portable.

E.6: LinuxThreads does not support the `PTHREAD_SCOPE_PROCESS` value of the "contentionscope" attribute. Why?

With a "one-to-one" model, as in LinuxThreads (one kernel execution context per thread), there is only one scheduler for all processes and all threads on the system. So, there is no way to obtain the behavior of `PTHREAD_SCOPE_PROCESS`.

E.7: LinuxThreads does not implement process-shared mutexes, conditions, and semaphores. Why?

This is another optional component of the POSIX standard. Portable applications should test `_POSIX_THREAD_PROCESS_SHARED` before using this facility.

The goal of this extension is to allow different processes (with different address spaces) to synchronize through mutexes, conditions or semaphores allocated in shared memory (either SVR4 shared memory segments or `mmap()`ed files).

The reason why this does not work in LinuxThreads is that mutexes, conditions, and semaphores are not self-contained: their waiting queues contain pointers to linked lists of thread descriptors, and these pointers are meaningful only in one address space.

Matt Messier and I spent a significant amount of time trying to design a suitable mechanism for sharing waiting queues between processes. We came up with several solutions that combined two of the following three desirable features, but none that combines all three:

- allow sharing between processes having different UIDs

- supports cancellation
- supports `pthread_cond_timedwait`

We concluded that kernel support is required to share mutexes, conditions and semaphores between processes. That's one place where Linus Torvalds's intuition that "all we need in the kernel is `clone()`" fails.

Until suitable kernel support is available, you'd better use traditional interprocess communications to synchronize different processes: System V semaphores and message queues, or pipes, or sockets.

F. C++ issues

F.1: Are there C++ wrappers for LinuxThreads?

Douglas Schmidt's ACE library contains, among a lot of other things, C++ wrappers for LinuxThreads and quite a number of other thread libraries. Check out http://www.cs.wustl.edu/~schmidt/ACE.html

F.2: I'm trying to use LinuxThreads from a C++ program, and the compiler complains about the third argument to `pthread_create()`!

You're probably trying to pass a class member function or some other C++ thing as third argument to `pthread_create()`. Recall that `pthread_create()` is a C function, and it must be passed a C function as third argument.

F.3: I'm trying to use LinuxThreads in conjunction with libg++, and I'm having all sorts of trouble.

From what I understand, thread support in libg++ is completely broken, especially with respect to locking of iostreams. H.J.Lu wrote:

> If you want to use thread, I can only suggest egcs and glibc. You can find egcs at http://www.cygnus.com/egcs. egcs has libsdtc++, which is MT safe under glibc 2. If you really want to use the libg++, I have a libg++ add-on for egcs.

G. Debugging LinuxThreads programs

G.1: Can I debug LinuxThreads program using gdb?

Yes, but not with the stock gdb 4.17. You need a specially patched version of gdb 4.17 developed by Eric Paire and colleagues at The Open Group, Grenoble. The patches against gdb 4.17 are available at `http://pauillac.inria.fr/~xleroy/linuxthreads/gdb-4.17-debug-threads.patch.gz`. H.J.Lu also develops patches for gdb 4.17 that include thread support and more; those are available at `ftp://ftp.valinux.com/pub/support/hjl/gdb/`. More recent versions of gdb seem to include LinuxThreads support.

Some Linux distributions provide an already-patched version of gdb; others don't. For instance, the gdb in RedHat 5.2 and RedHat 6.2 is thread-aware, but apparently not the one in RedHat 6.0. Just ask (politely) the makers of your Linux distributions to please make sure that they apply the correct patches to gdb.

G.2: Does it work with post-mortem debugging?

Not very well. Generally, the core file does not correspond to the thread that crashed. The reason is that the kernel will not dump core for a process that shares its memory with other processes, such as the other threads of your program. So, the thread that crashes silently disappears without generating a core file. Then, all other threads of your program die on the same signal that killed the crashing thread. (This is required behavior according to the

POSIX standard.) The last one that dies is no longer sharing its memory with anyone else, so the kernel generates a core file for that thread. Unfortunately, that's not the thread you are interested in.

G.3: Any other ways to debug multithreaded programs, then?

Assertions and `printf()` are your best friends. Try to debug sequential parts in a single-threaded program first. Then, put `printf()` statements all over the place to get execution traces. Also, check invariants often with the `assert()` macro. In truth, there is no other effective way (save for a full formal proof of your program) to track down concurrency bugs. Debuggers are not really effective for subtle concurrency problems, because they disrupt program execution too much.

H. Compiling multithreaded code; errno madness

H.1: You say all multithreaded code must be compiled with `_REENTRANT` defined. What difference does it make?

It affects include files in three ways:

- The include files define prototypes for the reentrant variants of some of the standard library functions, e.g. `gethostbyname_r()` as a reentrant equivalent to `gethostbyname()`.
- If `_REENTRANT` is defined, some `<stdio.h>` functions are no longer defined as macros, e.g. `getc()` and `putc()`. In a multithreaded program, stdio functions require additional locking, which the macros don't perform, so we must call functions instead.
- More importantly, `<errno.h>` redefines errno when `_REENTRANT` is defined, so that errno refers to the thread-specific errno location rather than the global errno variable. This is achieved by the following `#define` in `<errno.h>`:

  ```
  #define errno (*(__errno_location()))
  ```

  which causes each reference to errno to call the `__errno_location()` function for obtaining the location where error codes are stored. libc provides a default definition of `__errno_location()` that always returns `&errno` (the address of the global errno variable). Thus, for programs not linked with LinuxThreads, defining `_REENTRANT` makes no difference w.r.t. errno processing. But LinuxThreads redefines `__errno_location()` to return a location in the thread descriptor reserved for holding the current value of errno for the calling thread. Thus, each thread operates on a different errno location.

H.2: Why is it so important that each thread has its own errno variable?

If all threads were to store error codes in the same, global errno variable, then the value of errno after a system call or library function returns would be unpredictable: between the time a system call stores its error code in the global errno and your code inspects errno to see which error occurred, another thread might have stored another error code in the same errno location.

H.3: What happens if I link LinuxThreads with code not compiled with `-D_REENTRANT`?

Lots of trouble. If the code uses `getc()` or `putc()`, it will perform I/O without proper interlocking of the stdio buffers; this can cause lost output, duplicate output, or just crash other stdio functions. If the code consults errno, it will get back the wrong error code. The following code fragment is a typical example:

```
do {
  r = read(fd, buf, n);
  if (r == -1) {
    if (errno == EINTR)   /* an error we can handle */
```

```
            continue;
          else {                /* other errors are fatal */
            perror("read failed");
            exit(100);
          }
        }
      } while (...);
```

Assume this code is not compiled with `-D_REENTRANT`, and linked with LinuxThreads. At run-time, `read()` is interrupted. Since the C library was compiled with `-D_REENTRANT`, `read()` stores its error code in the location pointed to by `__errno_location()`, which is the thread-local errno variable. Then, the code above sees that `read()` returns -1 and looks up errno. Since `_REENTRANT` is not defined, the reference to errno accesses the global errno variable, which is most likely 0. Hence the code concludes that it cannot handle the error and stops.

H.4: With LinuxThreads, I can no longer use the signals `SIGUSR1` and `SIGUSR2` in my programs! Why?

The short answer is: because the Linux kernel you're using does not support realtime signals.

LinuxThreads needs two signals for its internal operation. One is used to suspend and restart threads blocked on mutex, condition or semaphore operations. The other is used for thread cancellation.

On ``old'' kernels (2.0 and early 2.1 kernels), there are only 32 signals available and the kernel reserves all of them but two: `SIGUSR1` and `SIGUSR2`. So, LinuxThreads has no choice but use those two signals.

On recent kernels (2.2 and up), more than 32 signals are provided in the form of realtime signals. When run on one of those kernels, LinuxThreads uses two reserved realtime signals for its internal operation, thus leaving `SIGUSR1` and `SIGUSR2` free for user code. (This works only with glibc, not with libc 5.)

H.5: Is the stack of one thread visible from the other threads? Can I pass a pointer into my stack to other threads?

Yes, you can -- if you're very careful. The stacks are indeed visible from all threads in the system. Some non-POSIX thread libraries seem to map the stacks for all threads at the same virtual addresses and change the memory mapping when they switch from one thread to another. But this is not the case for LinuxThreads, as it would make context switching between threads more expensive, and at any rate might not conform to the POSIX standard.

So, you can take the address of an "auto" variable and pass it to other threads via shared data structures. However, you need to make absolutely sure that the function doing this will not return as long as other threads need to access this address. It's the usual mistake of returning the address of an "auto" variable, only made much worse because of concurrency. It's much, much safer to systematically heap-allocate all shared data structures.

I. X-Windows and other libraries

I.1: My program uses both Xlib and LinuxThreads. It stops very early with an "Xlib: unknown 0 error" message. What does this mean?

That's a prime example of the errno problem described in question H.2. The binaries for Xlib you're using have not been compiled with `-D_REENTRANT`. It happens Xlib contains a piece of code very much like the one in question H.2. So, your Xlib fetches the error code from the wrong errno location and concludes that an error it cannot handle occurred.

I.2: So, what can I do to build a multithreaded X Windows client?

The best solution is to use X libraries that have been compiled with multithreading options set. Linux distributions that come with glibc 2 as the main C library generally provide thread-safe X libraries. At least, that seems to be the case for RedHat 5 and later.

You can try to recompile yourself the X libraries with multithreading options set. They contain optional support for multithreading; it's just that the binaries provided by your Linux distribution were built without this support. See the file `README.Xfree3.3` in the LinuxThreads distribution for patches and info on how to compile thread-safe X libraries from the Xfree3.3 distribution. The Xfree3.3 sources are readily available in most Linux distributions, e.g. as a source RPM for RedHat. Be warned, however, that X Windows is a huge system, and recompiling even just the libraries takes a lot of time and disk space.

Another, less involving solution is to call X functions only from the main thread of your program. Even if all threads have their own errno location, the main thread uses the global errno variable for its errno location. Thus, code not compiled with `-D_REENTRANT` still "sees" the right error values if it executes in the main thread only.

This is a lot of work. Don't you have precompiled thread-safe X libraries that you could distribute?

No, I don't. Sorry. But consider installing a Linux distribution that comes with thread-safe X libraries, such as RedHat 6.

I.3: Can I use library FOO in a multithreaded program?

Most libraries cannot be used "as is" in a multithreaded program. For one thing, they are not necessarily thread-safe: calling simultaneously two functions of the library from two threads might not work, due to internal use of global variables and the like. Second, the libraries must have been compiled with `-D_REENTRANT` to avoid the errno problems explained in question H.2.

I.4: What if I make sure that only one thread calls functions in these libraries?

This avoids problems with the library not being thread-safe. But you're still vulnerable to errno problems. At the very least, a recompile of the library with `-D_REENTRANT` is needed.

I.5: What if I make sure that only the main thread calls functions in these libraries?

That might actually work. As explained in question I.1, the main thread uses the global errno variable, and can therefore execute code not compiled with `-D_REENTRANT`.

I.6: SVGAlib doesn't work with LinuxThreads. Why?

With a recent kernel (2.2 or later) and the glibc version of LinuxThreads, there should be no problems. With older kernels or LinuxThreads version, both LinuxThreads and SVGAlib use the signals `SIGUSR1` and `SIGUSR2`. See question H.4.

J. Signals and threads

J.1: When it comes to signals, what is shared between threads and what isn't?

Signal handlers are shared between all threads: when a thread calls `sigaction()`, it sets how the signal is handled not only for itself, but for all other threads in the program as well.

On the other hand, signal masks are per-thread: each thread chooses which signals it blocks independently of others. At thread creation time, the newly created thread inherits the signal mask of the thread calling `pthread_create()`. But afterwards, the new thread can modify its signal mask independently of its creator

thread.

J.2: When I send a `SIGKILL` to a particular thread using `pthread_kill`, all my threads are killed!

That's how it should be. The POSIX standard mandates that all threads should terminate when the process (i.e. the collection of all threads running the program) receives a signal whose effect is to terminate the process (such as `SIGKILL` or `SIGINT` when no handler is installed on that signal). This behavior makes a lot of sense: when you type "ctrl-C" at the keyboard, or when a thread crashes on a division by zero or a segmentation fault, you really want all threads to stop immediately, not just the one that caused the segmentation violation or that got the `SIGINT` signal. (This assumes default behavior for those signals; see question J.3 if you install handlers for those signals.)

If you're trying to terminate a thread without bringing the whole process down, use `pthread_cancel()`.

J.3: I've installed a handler on a signal. Which thread executes the handler when the signal is received?

If the signal is generated by a thread during its execution (e.g. a thread executes a division by zero and thus generates a `SIGFPE` signal), then the handler is executed by that thread. This also applies to signals generated by `raise()`.

If the signal is sent to a particular thread using `pthread_kill()`, then that thread executes the handler.

If the signal is sent via `kill()` or the tty interface (e.g. by pressing ctrl-C), then the POSIX specs say that the handler is executed by any thread in the process that does not currently block the signal. In other terms, POSIX considers that the signal is sent to the process (the collection of all threads) as a whole, and any thread that is not blocking this signal can then handle it.

The latter case is where LinuxThreads departs from the POSIX specs. In LinuxThreads, there is no real notion of ``the process as a whole'': in the kernel, each thread is really a distinct process with a distinct PID, and signals sent to the PID of a thread can only be handled by that thread. As long as no thread is blocking the signal, the behavior conforms to the standard: one (unspecified) thread of the program handles the signal. But if the thread to which PID the signal is sent blocks the signal, and some other thread does not block the signal, then LinuxThreads will simply queue in that thread and execute the handler only when that thread unblocks the signal, instead of executing the handler immediately in the other thread that does not block the signal.

This is to be viewed as a LinuxThreads bug, but I currently don't see any way to implement the POSIX behavior without kernel support.

J.3: How shall I go about mixing signals and threads in my program?

The less you mix them, the better. Notice that all `pthread_*` functions are not async-signal safe, meaning that you should not call them from signal handlers. This recommendation is not to be taken lightly: your program can deadlock if you call a `pthread_*` function from a signal handler!

The only sensible things you can do from a signal handler is set a global flag, or call `sem_post` on a semaphore, to record the delivery of the signal. The remainder of the program can then either poll the global flag, or use `sem_wait()` and `sem_trywait()` on the semaphore.

Another option is to do nothing in the signal handler, and dedicate one thread (preferably the initial thread) to wait synchronously for signals, using `sigwait()`, and send messages to the other threads accordingly.

J.4: When one thread is blocked in `sigwait()`, other threads no longer receive the signals `sigwait()` is waiting for! What happens?

It's an unfortunate consequence of how LinuxThreads implements `sigwait()`. Basically, it installs signal handlers on all signals waited for, in order to record which signal was received. Since signal handlers are shared with the other threads, this temporarily deactivates any signal handlers you might have previously installed on these

signals.

Though surprising, this behavior actually seems to conform to the POSIX standard. According to POSIX, `sigwait()` is guaranteed to work as expected only if all other threads in the program block the signals waited for (otherwise, the signals could be delivered to other threads than the one doing `sigwait()`, which would make `sigwait()` useless). In this particular case, the problem described in this question does not appear.

One day, `sigwait()` will be implemented in the kernel, along with others POSIX 1003.1b extensions, and `sigwait()` will have a more natural behavior (as well as better performances).

K. Internals of LinuxThreads

K.1: What is the implementation model for LinuxThreads?

LinuxThreads follows the so-called "one-to-one" model: each thread is actually a separate process in the kernel. The kernel scheduler takes care of scheduling the threads, just like it schedules regular processes. The threads are created with the Linux `clone()` system call, which is a generalization of `fork()` allowing the new process to share the memory space, file descriptors, and signal handlers of the parent.

Advantages of the "one-to-one" model include:

- minimal overhead on CPU-intensive multiprocessing (with about one thread per processor);
- minimal overhead on I/O operations;
- a simple and robust implementation (the kernel scheduler does most of the hard work for us).

The main disadvantage is more expensive context switches on mutex and condition operations, which must go through the kernel. This is mitigated by the fact that context switches in the Linux kernel are pretty efficient.

K.2: Have you considered other implementation models?

There are basically two other models. The "many-to-one" model relies on a user-level scheduler that context-switches between the threads entirely in user code; viewed from the kernel, there is only one process running. This model is completely out of the question for me, since it does not take advantage of multiprocessors, and require unholy magic to handle blocking I/O operations properly. There are several user-level thread libraries available for Linux, but I found all of them deficient in functionality, performance, and/or robustness.

The "many-to-many" model combines both kernel-level and user-level scheduling: several kernel-level threads run concurrently, each executing a user-level scheduler that selects between user threads. Most commercial Unix systems (Solaris, Digital Unix, IRIX) implement POSIX threads this way. This model combines the advantages of both the "many-to-one" and the "one-to-one" model, and is attractive because it avoids the worst-case behaviors of both models -- especially on kernels where context switches are expensive, such as Digital Unix. Unfortunately, it is pretty complex to implement, and requires kernel support which Linux does not provide. Linus Torvalds and other Linux kernel developers have always been pushing the "one-to-one" model in the name of overall simplicity, and are doing a pretty good job of making kernel-level context switches between threads efficient. LinuxThreads is just following the general direction they set.

Xavier.Leroy@inria.fr

GNU libc

This is a small web page for the GNU libc program, glibc. Look at the official FSF home page for glibc for more information.

Get glibc announcements:

your e-mail address Subscribe

Availability

Glibc releases and pre-releases are available by anonymous ftp.

The FAQ, distributed in the source tree, is also available online. It is quite long already (around 75kB) and still growing. So be careful when hitting the link.

You can access the development source tree a couple of different ways.

Anonymous CVS read-only access

```
cvs -z 9 -d :pserver:anoncvs@anoncvs.cygnus.com:/cvs/glibc login
{enter "anoncvs" as the password}
cvs -z 9 -d :pserver:anoncvs@anoncvs.cygnus.com:/cvs/glibc co libc
```

Read-only web-based CVS access

You can use the cvsweb interface.

Mailing list

There are four mailing lists regarding glibc hosted on sourceware.cygnus.com and one hosted on gnu.org: `libc-announce`, `libc-alpha`, `libc-hacker`, `glibc-cvs`, and `bug-glibc`.

Please note that `libc-hacker` is a closed list. You may look at the archives of this list, but subscription and posting are not open.

Mailing list: libc-announce

Your e-mail address:

☐ Digest version subscribe Do it

Mail archives are also available by anon-ftp in mbox formatted files.

Bug database

Glibc has a bug database at the FSF. Got a problem? Want to know if you're the first to have it? Check this database.

This page was last modified with loving care by jsm@cygnus.com on 1999-11-29.

GNU FTP list (Text Version Only)

Please send improvements to this file to gnu@gnu.org.

No Warranties

We distribute software in the hope that it will be useful, but without any warranty. No author or distributor of this software accepts responsibility to anyone for the consequences of using it or for whether it serves any particular purpose or works at all, unless he says so in writing. This is exactly the same warranty that the commercial software companies offer: None. If the distribution is incomplete or the media fails, you can always download a replacement from any of the GNU mirrors, free of charge.

Updates

A possibly more up-to-date list of GNU FTP sites is at http://www.gnu.org/order/ftp.html

How to FTP

Use the ftp program on your system (ask locally if you can't find it) to connect to the host you are ftping from. Unless indicated otherwise, login in as user "anonymous", with password: "your e-mail address" and set "binary" mode (to transfer all eight bits in each byte).

ALWAYS USE BINARY/IMAGE MODE TO TRANSFER THESE FILES! Text mode does not work for tar files or compressed files.

GNU Software and How To FTP It

GNU software is available on ftp.gnu.org under the directory /gnu. diff files to convert between versions exist for some of these programs. Some programs have misc support files as well. Have a look on ftp.gnu.org to see which ones. In most cases, the tar or diff files are compressed with the `gzip' program; this is indicated with the .gz suffix.

Descriptions of GNU software are available at http://www.gnu.org/software/software.html and in the Free Software Directory at http://www.gnu.org/directory/index.html

Alternative Internet FTP Sources

Please do NOT use a site outside your country, until you have checked all sites inside your country, and then your continent. Trans-ocean TCP/IP links are very expensive and usually very low speed.

The canonical GNU ftp site is located at ftp.gnu.org/gnu. You should probably use one of the many mirrors of that site - the mirrors will be less busy, and you can find one closer to your site.

GNU FTP Site Mirror List

- United States
  - California
    - gatekeeper.dec.com/pub/GNU/
    - ftp.keystealth.org/pub/gnu/
    - ftp.itcentrx.com/pub/gnu/
  - Illinois
    - uiarchive.cso.uiuc.edu/pub/ftp/ftp.gnu.org/gnu/ (Internet address 128.174.5.14)
  - Kentucky
    - gnu.ms.uky.edu/pub/mirrors/gnu/
  - Maryland
    - ftp.digex.net/pub/gnu/ (Internet address 164.109.10.23)
  - Massachusetts
    - aeneas.mit.edu/pub/gnu/
  - Michigan
    - gnu.egr.msu.edu/pub/gnu/

        - ftp.wayne.edu/gnu_mirror/
    - Missouri
        - wuarchive.wustl.edu/systems/gnu/
    - New Mexico
        - ftp.cs.unm.edu/mirrors/gnu/
    - New York
        - ftp.cs.columbia.edu/archives/gnu/prep/
        - ftp.stealth.net/pub/mirrors/ftp.gnu.org/ (Internet address 206.252.192.18)
    - North Carolina
        - http://metalab.unc.edu/pub/gnu/
    - Ohio
        - ftp.cis.ohio-state.edu/mirror/gnu/
    - Pennsylvania
        - boron.club.cc.cmu.edu/gnu/
    - Tennessee
        - sunsite.utk.edu/pub/gnu/ftp/
        - thales.memphis.edu/pub/gnu/
    - Virginia
        - ftp.uu.net/archive/systems/gnu/
    - Washington
        - ftp.nodomainname.net/pub/mirrors/gnu/
        - gnu.wwc.edu/
    - Wisconsin
        - ftp.twtelecom.net/pub/GNU/
- Africa
    - South Africa
        - ftp.sun.ac.za/gnu/
- The Americas
    - Brazil
        - ftp.unicamp.br/pub/gnu/
        - master.softaplic.com.br/pub/gnu/
        - ftp.matrix.com.br/pub/gnu/
    - Canada
        - ftp.cs.ubc.ca/mirror2/gnu/

    - Chile
        - ftp.inf.utfsm.cl/pub/gnu/ (Internet address 146.83.198.3)
    - Costa Rica
        - sunsite.ulatina.ac.cr/Mirrors/GNU/
    - Mexico
        - ftp.uaem.mx/pub/gnu/
        - www.gnu.unam.mx/pub/gnu/software/
        - gnu.cem.itesm.mx/pub/mirrors/gnu.org/
- Australia
    - Australia
        - ftp.progsoc.uts.edu.au/pub/gnu/
        - mirror.aarnet.edu.au/pub/gnu/
        - gnu.mirror.pacific.net.au/gnu/
- Asia
    - China
        - ftp.sea9.com/pub/gnu/
        - ftp.cs.cuhk.edu.hk/pub/gnu/
        - sunsite.ust.hk/pub/gnu/
        - ftp.shellhung.org/pub/gnu/
    - Japan
        - tron.um.u-tokyo.ac.jp/pub/GNU/prep/
        - ftp.cs.titech.ac.jp/pub/gnu/
        - core.ring.gr.jp/pub/GNU/
        - ftp.ring.gr.jp/pub/GNU/
        - mirrors.hbi.co.jp/gnu/
    - Korea
        - cair-archive.kaist.ac.kr/pub/gnu/ (Internet address 143.248.186.3)
        - ftpmirror.hanyang.ac.kr/GNU/
        - ftp.linux.sarang.net/mirror/gnu/gnu/ (also mirrors alpha.gnu.org/gnu/ at ../alpha/)
        - ftp.xgate.co.kr/pub/mirror/gnu/
    - Saudi Arabia
        - ftp.isu.net.sa/pub/mirrors/prep.ai.mit.edu/pub/gnu/
    - Taiwan

        - coda.nctu.edu.tw/UNIX/gnu/
        - ftp1.sinica.edu.tw/pub3/GNU/gnu/
        - ftp://ftp.nctu.edu.tw/UNIX/gnu/
    - Thailand
        - ftp.nectec.or.th/pub/mirrors/gnu/ (Internet address - 192.150.251.32)
- Europe
    - Austria
        - ftp.gnu.vbs.at/
        - ftp.univie.ac.at/packages/gnu/
        - gd.tuwien.ac.at/gnu/gnusrc/
    - Belgium
        - ftp.be.gnu.org/
    - Austria
        - http://gd.tuwien.ac.at/gnu/gnusrc/
    - Czech Republic
        - ftp.fi.muni.cz/pub/gnu/
    - Denmark
        - ftp.denet.dk/mirror/ftp.gnu.org/pub/gnu
        - ftp.dkuug.dk/pub/gnu/
    - Finland
        - ftp.funet.fi/pub/gnu/prep/
    - France
        - ftp.irisa.fr/pub/gnu/
        - ftp.medasys-digital-systems.fr/pub/gnu/
        - ftp.fuitad.net/mirrors/ftp.gnu.org/
        - hansolo.mtesa.net/gnu/
    - Germany
        - ftp://ftp.cs.tu-berlin.de/pub/gnu/
        - ftp.leo.org/pub/comp/os/unix/gnu/
        - ftp.informatik.rwth-aachen.de/pub/gnu/
        - ftp.de.uu.net/pub/gnu/
        - ftp.freenet.de/pub/ftp.gnu.org/gnu/
        - ftp.gigabell.net/pub/gnu/

        - ftp.cs.uni-bonn.de/pub/gnu/
        - ftp-stud.fht-esslingen.de/pub/Mirrors/ftp.gnu.org/
        - http://ftp-stud.fht-esslingen.de/pub/Mirrors/ftp.gnu.org/
        - ftp.stw-bonn.de/pub/mirror/ftp.gnu.org/
    - Greece
        - ftp.forthnet.gr/pub/gnu/
        - ftp.ntua.gr/pub/gnu/
        - ftp.duth.gr/pub/gnu/
        - ftp.aua.gr/pub/mirrors/GNU/ (Internet address 143.233.187.61)
        - ftp.physics.auth.gr/pub/gnu/
    - Hungary
        - ftp.kfki.hu/pub/gnu/
    - Ireland
        - ftp.esat.net/pub/gnu/ (Internet address 193.120.14.241)
    - Italy
        - ftp.oasi.gpa.it/pub/gnu/
    - Netherlands
        - ftp.eu.net/gnu/ (Internet address 192.16.202.1)
        - ftp.nluug.nl/pub/gnu/
        - ftp.mirror.nl/pub/mirror/gnu/
        - ftp.nl.uu.net/pub/gnu/
    - Norway
        - ftp.ntnu.no/pub/gnu/ (Internet address 129.241.11.142)
        - ftp.gnu.no/
        - sunsite.uio.no/pub/gnu/
    - Poland
        - ftp.task.gda.pl/pub/gnu/
        - sunsite.icm.edu.pl/pub/gnu/
    - Portugal
        - ftp.ci.uminho.pt/pub/mirrors/gnu/
        - http://ciumix.ci.uminho.pt/mirrors/gnu/
        - ftp.ist.utl.pt/pub/gnu/
        - mirrors.netvisao.pt/gnu/

- Romania
  - archive.logicnet.ro/mirrors/ftp.gnu.org/gnu/
  - ftp.timisoara.roedu.net/mirrors/ftp.gnu.org/
- Russia
  - ftp.chg.ru/pub/gnu/
- Slovenia
  - ftp.arnes.si/gnu/
- Spain
  - ftp.etsimo.uniovi.es/pub/gnu/
  - ftp.rediris.es/pub/gnu/
- Sweden
  - ftp.isy.liu.se/pub/gnu/
  - ftp.stacken.kth.se/pub/gnu/
  - ftp.luth.se/pub/unix/gnu/
  - ftp.sunet.se/pub/gnu/ (Internet address 130.238.127.3)
  - ftp.chl.chalmers.se/pub/gnu/
- Switzerland
  - ftp.eunet.ch/mirrors4/gnu/
  - sunsite.cnlab-switch.ch/mirror/gnu/ (Internet address 193.5.24.1)
- Turkey
  - ftp.baskent.edu.tr/gnu/ftp/
  - ftp.ulak.net.tr/pub/gnu/
- United Kingdom
  - ftp.mcc.ac.uk/pub/gnu/ (Internet address 130.88.203.12)
  - ftp.mirror.ac.uk/sites/ftp.gnu.org/pub/gnu/
  - ftp.warwick.ac.uk/pub/gnu/ (Internet address 137.205.192.13)
  - sunsite.org.uk/gnu/ (Internet address 193.195.63.2)

How to FTP GNU Emacs

Emacs is in the directory /gnu/emacs on ftp.gnu.org. The emacs distribution itself has a filename in the form emacs-M.N.tar.gz, where M and N stand for the version numbers; the Emacs Lisp Reference Manual is in a separate file, named elisp-manual-NN.tar.gz.

Scheme and How to FTP It

The latest distribution version of C Scheme is available via anonymous FTP from swiss-ftp.ai.mit.edu in /pub/scheme-X.X/ (where X.X is some version number).

Read the files INSTALL and README in the top level C Scheme directory.

TeX and How to Obtain It

We don't distribute TeX now, but it is free software.

TeX is a document formatter that is used, among other things, by the FSF for all its documentation. You will need it if you want to make printed manuals.

TeX is freely redistributable. You can get it by ftp, tape, or CD/ROM.

For FTP instructions, retrieve the file

ftp.cs.umb.edu/pub/tex/unixtex.ftp. (We don't include it here because it changes relatively frequently. Sorry.)

A minimal TeX collection (enough to process Texinfo files, anyway)

is included on the GNU source CD-ROM. See the file ORDERS in this directory for more information.

VMS FTP sites with GNU Software

You can anonymously ftp a VMS version of GNU emacs from:

- ftp.vms.stacken.kth.se:[.GNU-VMS] - GNU Emacs and some other VMS ports (and some VMS binaries) of GNU software
- mango.rsmas.miami.edu has a VMS version of the GCC/G++ compiler. Contact angel@flipper.miami.edu (angel li) for details.
- RIGEL.EFD.LTH.SE [130.235.48.3] - GNU Emacs

Getting GNU software in Great Britain

jpo@cs.nott.ac.uk is willing to distribute those GNU sources he has available. The smaller items are available from the info-server (send to info-server@cs.nott.ac.uk); the larger items by negotiation. Due to communication costs this service is only available within the UK.

BattenIG@computer-science.birmingham.ac.uk (aka I.G.Batten@fulcrum.bt.co.uk) is also willing to distribute those GNU sources he has.

wizards@doc.ic.ac.uk is willing to distribute those GNU sources they have along with most other freely distributable software. The SunSITE archive on SunSITE.doc.ic.ac.uk (193.63.255.4) is available via ftp, http, fsp, gopher, NFS and Lanmanger over IP (SMB), and telnet.

UK sites with just anonymous FTP access are in the above list.

Getting GNU software via UUCP

OSU is distributing via UUCP: most GNU software, MIT C Scheme, Compress, News, RN, NNTP, Patch, some Appletalk stuff, some of the Internet Requests For Comment (RFC) et al.. See their periodic postings on the Usenet newsgroup comp.sources.d for informational updates. Current details from <staff@cis.ohio-state.edu> or <...!osu-cis!staff>.

Information on how to uucp some GNU programs is available via electronic mail from: uunet!hutch!barber, hqda-ai!merlin, acornrc!bob, hao!scicom!qetzal!upba!ugn!nepa!denny, ncar!noao!asuvax!hrc!dan, bigtex!james (aka james@bigtex.cactus.org), oli-stl!root, src@contrib.de (Germany), toku@dit.co.jp (Japan) and info@ftp.uu.net.

If You Like The Software

If you like the software developed and distributed by the Free Software Foundation, please express your satisfaction with a donation. Your donations will help to support the Foundation and make our future efforts successful, including a complete development and operating system, called GNU (Gnu's Not Unix), which will run Unix user programs. For more information on GNU and the Foundation, contact us at the above address, or see our web site at http://www.gnu.org.

Ordering a GNU Source Code CD-ROM or Source Code CD-ROM Subscription is a good way for your organization to help support our work.

Return to GNU's home page.

FSF & GNU inquiries & questions to *gnu@gnu.org*. Other ways to contact the FSF.

Comments on these web pages to *webmasters@gnu.org*, send other questions to *gnu@gnu.org*.

‹²ž©‰a`[yw&€-ÿI|`Jç÷#^Ã¯q¿_æûqÆÜ:y™%>:B XHzZLô%ÿYç.UM‰Ãk{§Ð´öÕ-×Ł_#o<I8ºÑ°³ªÚHkÞÜ}|Ø,O[ÃX:÷Qèè¦÷^²zRm6›IL®¦™®‰b!Ð»}Ø…y»€}p°_-×èè}W™WIE}ÑÐ²ìlÃL2ôÃ¢Ún1Š:¾ùšCU¼ù¼?¦S/´gfÕ(H%

```
1cñâiÿfµ),8
£Œ0-L5 ¶¿ûA×~>~A×ƒ0¶ûu·*;ô°C¯]¤¥Y(DÑøæK)üâã+;±cúØ¤Ôƒ†_Uîjy»®Ÿ ≥(~çOu1¨z‰(¾ð7rÁT,c¨
ln·w»ºS¶ ◊;''ýXOBqSlûqøb×ûBThc;;B°ûúŁZ…?·Yô-ÍÒ¼¤
kø®ß3G©//ÀMDÃ§iÃÓ(···;ÒôôÃÁÔ²ÍJãà›°âü¿½¿¹ »•›ûB²ûÍÍ1ÿ>ŽE9ñ?'$5ÿ¨/*±‹®®°ÔŒ4ŕàÔEbÃ6ntÍÊµCð3lÔbB5ƒ4âŸ|—
/®BH®÷{ßŽ¶°ßg'a°Ã×5   ×1¨IHŠ±-4-°ÆXûj]ã¿no.®Ã®°v·¼ç_Æ«ñ1°ãæ8
```

IpÕ€t€çÍd¨&÷§-bÍŒa5™ ³À¯-øOOM¨iGØQ|-FÓ¿$‰¤/(ô–
Aa@™$;wæ¶ÂncÁZ±
Y¥gw´rnKÓfßi³M~ê÷GL·7°/oR?UtuKNïà¤Mq÷t°µPJ×sQ_¼âÕÏcŒÏu_Nõ¾Ç¯àáŒ£×lØÕ|r7ØMdÉélinù^•}Þ¤MÉ•wÕ¯5¡}ÕÐÉ/rD!¬{[:IÜ•ÅvgM¯_9]g¯NC)»}¸Ÿ-B}·±#Øø,P•'Güã³w,,Y™)L¢Ù³Þ-¨ß7Gìì¸r3Ìê{CøßWN<•w•ì›ìÕóGßÅ

c4Isx=O4jUBt +`GIXNArq=01Y›•YEb^6T‹6lJ`æèÖ">›%ôâjJW¨ÛürSW%bv;ß,PO6*IGOO

QÉ8/I«»1;-3´ãOGIŸ)ûQ"»"+ÐE-¯¼QCæÐQùò¤ÕÃûÃ—•20,×!!ô³(
!by…€/™%•é¦Ä½wå²°!bŸ¼¤g§¶)—«ÆÐÉ ſcÏ¡?^Ø¢v ç¶..ſ¶‹ô¹-k»Ã‰Ê¨ú½ù
¤&ZÐ÷1´ŷ¼_9.™B³ØØ—¿'§ô_uſ§¶&A-IkÍZ%Ÿ]¯ÐqÀJ8-I"É.w«ð³?ûÏÏÍ¯wſ6ùŒÉ-ÉôØÍlÕ-Ê Ô÷²]ê_ª\•ûÏŷ¤°û..Áŷ×âø÷ÐX~-Ûã''bŒ¯Ó&¶¼'¾-ûıŒûÐ2ú"ØÕ ·-·;ſ(ſ^ÓÔ•ÐÍ«é·€ÆÅuÅð7¾ŒZ
z7Z|ö.Aŝ´ãê6•ãBìÃ!²²ſ²ZBÍ§Æ&•,ðYÐ-ýŒ÷Š«Š·)P1ëÚDJ»ãÖ§?2\\[Ð:¾4%ÆBZÆ®Z´ÊI:ÃÉ-ë÷*Ê´Š¿<ÐP½ÆbŸÃ'7'¶ÐØÆf•)•1ºQ•ÆZÏŸ©,ØÕ²XÇ^ ¹vŒÍlÉ…¤°¹ŷ",QŠÉñ6Ð», ¬µÆMæ½×CÍkÃZÉ§2Ô^ÐK73Ç•cÐÖ÷Ô÷ìÑ&Ù7,ZÐb/Ðk¹<„.µí'ep»$G×ìâÉ&£P45×Þ8<´Ár

Ût'Á¦p'St«`Gb1K$âK Ã<1ü).w»>AD»$'ôÕ¶÷,f0€YÃ8ê%
tê¡åµæ¡<¬8,þJCQ®7n}"U°°¹l Ûi÷ò®Wôy« ãŸÆ>ÕQŸÆ*WÑÜz•ëg•)Þ°ÆÐöøÐÜ¡xø´Eµ.x

```
DdSD.<a\6;J#0{2DGD`B¿mpzs=ëyCE•}s-Di
@Tm¢%<SA,h))9KÐ8K-[:{•bP"O.{Wnë•Y¿ëYBdo 38<P6$ch$:TÞi•m"c2i,
0U~²8ãmDô ´'•U",½ ¥'ÓãßÑá|"!:ïRB,|ßT>®|ê²5€?ßzBBbÃÐi·5vmsm³·<psãQ 57Œ•Ð¬
```

BI ©=øQ=Ûø,•Ay†'®EYÉY©)ªÍCXøÞ-rrrslá´šXß&ªFVØ®ˆÕ%Ù÷L³þ=IYíß,÷£øXV1tW
-Yây•
2Ö¯ÃÒ±É³ní8¤
µ®Ø¨pÛÐ

O(JKHP%E$n|Ø=³X-™lŠ÷ãæ¢,ø'Yƒ÷ü•,Ò-Aš,7UqÊH8Ê'lÌ¤².–m…-%£ziãÒÕ|;2%'"·!®
Tî[¼E…ÛzwIXiåÝÃšfZ÷D'j‡™ª%Mx,TSxmøpY÷1N}÷(ORI

```
âoo8F=U•Pove HU ¨Z ŸÀ|rBLBG«B{þ6'ŒE-E,b«!IZAYIU'f'•2WzÞj !%i*mIþ54GR•O_
_i•'ë|}Þ4\ ŽBøMwzþ:DI°âOÎ'u/IµiÉšÕ-(ªµ' ‘š|&"
AUCnCBBb·Ðz•'ÐÒ×
3mBUN•rTB;l-UR
U|ãTS-•Ø}ÑcÉ7PµÆÐ·€› Ð7@†@
#4Z|bÁDbÞ#IAtCp'ÐÐ+*••1BÍGr•'ê»{ÿTlÍ|'fIM²j!Rf•hß©ìZE5ç°9Ã-WZ?»Ù•JFNøÐ4||oTü{{-ùwº…>6{{•Ù1ôœL
ûZ`Ü"«L_f§1uB-'Ã/Ò ]QÃò ÐwÃ›ŒBk+ŒÐ I•Ix»_ÐTê=",\>9Ac²ÐÃ-ÿBTpAIz>Z'zÐ-•*ÐÐ'?ÐŠ³ÞM3
```

D¡ØEOÃc°(-¸Á÷ÐÃ±|Ã
lì÷ÐvLZl=¤ÆÐv»ùÜÐVXn³ÁFÑ¯âad°KWÕ÷IÃfIIv»'ØæQ¥T»ÃBâ/LCæJJEÊ°)´Á÷•6]>Y&ø®¦÷•6ëÕ¨srzlÊ9¥§ÕµÁ6©%]¯Zzwßs0ÊSÊjŸ6à8)!2x4Ü|ƒþ%ã*x_G7EZüŒßa+ÆÃN¯âÜÃÑÃ†Ðv¡8³!tÒj²´BãâDXNw/'lgw-p•é65âL]|PŸÁ1µ°ôÕÎm‹SPUi-Ðý÷à›»,pLÃ¤™T•®Ðâqsdø¨.Ù»ZŒ@÷ÆKñU8•K´ÑKIÕ2æ»ø,w,¢Ü4rdHÇ\;K»ý
s»š-n»´ØPàÊ°Ÿ´ÐVìk°³t3•zP°[ÃÕZIUHÑÃ{üâ,Lìjk¨jÃ< ¼•'°P :ìI¨$Fø,5Ø9Yu2ÐD6¦Æ

ùñ÷|Ïôèð^OÀ¯<SŠWæŠà]@TfC"Ò5Lù<¿„"(WT¯TàMtø,-ƒ'ùEII+-ã;J'S÷»OÆÐÉxðêi»A¸Wèkðæ¯5ðP{Sÿþï¡:gÉ?Ëïÿ&Ez»ÿ›)&Awé\k;?ê"ø¯xðfÿ;Xk&ØÏLN_(Ïš-¬šo÷âÈ¼' _Ðî-wÏ#¯?u¢ø§Ï¿~ÿsT÷Pæ¯ðžNd<²ïeYRæ›ó:Oßæ›>Ðèïšû]Qïî·ÆÐ€¢,""ê‰u

IIZFâ«¤CBLI¤8S(Z° «E
т--°k¹Q¹IâŸ¥\|ò·•jQZSaa

µø´q§ss'äö{ä8é\RL²'Œ ¡âÜÖ
ÜÅ[´qÉ[Ÿ~áÖ´<±
Ù÷Ðæ,Ñâ%ûâZ×a*}uøÃ7ÜÉ²ê:ŒIÎÖ"Hk£%}}ô&t~Ži•ù@r ‹'–:Ëθöøhß5-1™;,)hÃ"®×€g$®x3lŁôŽ÷ '÷©iøø

```
"N<ü¹'ãoÖSæsHH%|Ol
mã+ŸjS^[`µ1YÜ'j|ét.m.-ÃBB?ô,~ßWPç}ŒØ|ôdæÍßn-‰7à?;;"v"-F»x'€‰>_Aãl%µs }}ÜtÁŕr…ê÷ŕ`ïãŒ¥IBÃ‰C`A•§5ŸÃU0
'T-5<ÖBÙ'ßly-z°gUØYßQjl
rgO…Ï'™‰+Œ‡«.¾-ÏŠvgŸ÷WÃ6ŒBŒNIÃÃG¯üúã2™uÃzü'¥1'",ãg}ÁwÃVæ¢¶:[Á^ŽÇ9'Oï ÃL‰>JŸÏ#¥ø1‹ÇÐBO%(Ã†WGªIŸ/-¿ê4.V)6•¥=ä,›"
8@')¼I_ÃxÏÏ¾",ØŒßÆÃ?+ÿh-`Ã©*z>Ã(¢8+€×-ß
çfúöx•f/Aê2]ãÙ£ÇÇp·ee8ª-ÕeG«ŒEBy~",,YÃ®Rßd
üZtF=ŽwÆ      o(=}ê•Ão_ŀ¡ûI7,Ñ© ©?^
ö÷é'o-úŒ*Êo÷yØÊ1÷þ`FãŸ`|ÿÜhO7Gø#ÚŸIz^ÃBÚÊiâµcC>ä®aÃ`ŝ_µ¿ØcXØ/~;»6,Ãç?ð7©Ÿ`fŒ†;ø8^^ÎŒ¥÷FjI-IÕ§?•-ü-gv¢ä"_üs?Ü|Šwèüco
y##Ÿ
?üŠ5MÇ11
™Yĺ»}ô~ï¿ŷ`gïÜÖÖ÷Ð-ÐÍæÐVÆ!Ó¯÷ŒL™Á(À`+ón^ñ%ïÚ•×Cß@Lai=G»Œ3Ø|ÚÊã'ÜBBÚ!ÃPüGR      ú¢¢ä
```

ãü¯Y»;:ßß
k-Dwæm'ÕQŸ_hÜÀ•Rış'F·-üüZ¼`ß'¯ãTi<ŒA¯-Ð,ıovJ~ÿ'{p ÿCTI{?u(*MßJ553\Ð»ÿlôsŒ V•p'Ãh>é/'1.ÛN•N8!×eOCŠ›'mã'sçBWwoßb,ŒŒ[xI%_4ŸƒŒZ"ß—aÃ}Ÿ6tLò-¬[·-°fãß.{Ù¤;HXSEyÎ¶•ûvß§1ßŠ¿
Ø.;`h,(6/IDbÿÍ•L?¶5ÍF,#7¼÷ ã'¥3Tſiã ÷ÿvç¡ſ¥ÙÒÒÜÿ¾¶Ãlr^Ò `-þÃ&æŒ…ÁŠÃv{Ò ^QÕLIñ,dMŒÍÐXsÎU¤Ð<™RsXC¥BBm÷ãŸBãßÍ¶Ÿ ¤÷ıl-÷¼:ţ¶5|Kk,}ã|:G†~>Œ`_Þ÷ç¤DŠßX4ÿ¥÷ı«/MŽ]Šß:Ð-ßuãéGpuxÿ1»ÿ÷ÇvWÕŠÉſ'úK'

+SAŸ@Ø~F… £ö·`qäöñ•ÿ¸ ,•ò€'÷•r÷•®ø°w~(/£Ü…sÖ°¨Ü1Cê',|X
â²²bã¨¤_,°ÒöÑ¾ÿà`l_NdÍ ·Aa¨GØ€Ï7>;†`^~™nûñ^ãîƒ²Ð•ÿ¸ x
rr¶Ä©'×Š,©Š"<Fzÿ|æ~Ø¦`ï¿jf5©!b,Yÿ©~ÍÜ¥fh·/¯ ¯ÕÿßBÆµÊ ] »Œ:×†yÇ¥Ïÿ'Ø~`¾Šã
Ü'Ò@Ç'Ï(°dÝ®Ëà¨ãÍÇFïßÜÿÖß×x,Œ»}Ûm²²25âôÀ±ÍÎõ÷Ï©'ÉÆwßÍ¡ØØæ±Ø¦ÃëÔ JœÆø®®Ñ×Á×ß¶Øüa[ìþ¨¨Í;æü\
Aìæ-ßÍ¯ßÐu±ÕüZMÍ[¥ÜÆ¡FWæßì¦Í•,ºÐÿßð ³Ý¥Ø~ûpy÷ßbüÜà ×B/Ãqµ%×~•~/zMïmÆÉ ßÍaÜ™ï'ÍßÜu¯
ßx‹•ZEßµ4·ç]W!vßÃEÿ7e>©ß°ùmÖ™,Ãß÷ ü,ß•¶¬-ïî|ß»jÿÃJäßü÷¿ ¯ßÜñ:ÈxæâàâÉFá
'¡•ëx¼ç^© úÐ?ÿ•Ëp9Ã'¿²}ßP®à n¿¢ ÚÐ.ðrBÑ6:†ÌÞ0Ñ®%÷ë×Ãßæ×•ì¤ ®Qœ£-ÃÐu]÷ßë÷Ïß¼mÑ9·ª"©LEUßPà¡Ñ5¯:@•,®ª-fr¥¨ ˜€(ëÃâ^T
CP¨'>]Ù¨:1°ë÷¿Œ ùÆŽ™ßÃ×÷ëQ@‡_¯Wµ'³ª¢1zvC•c',Ô÷ëÜ®ÜÃ]
ŒÉ7rÑÑ-ãJ@fïÃ¤vÍ·®®vnne÷§
M-?ÉÜŒÑ) ^Ð×1]»ëÜ®Ú¼ã•Ãß¤ë÷ìC•Ãßßë÷ë'¼.ß§²Ã¿ßÐ³yIy-Ê°ÔK'uÖßdßßü-Í°ê›Ígc³½)x¼ðþÔdQ
'Ëìë÷gÃüÃüÜßÐ÷FÃ™æÔ•×ëæÿ?Ðc_æÝ ¾÷ëÿüÃ¡-ÀÑ¼ÃÃÔßßÑæ`ÿÃ¿æÏ~

y$d,v:¤¤`AÖÃã
z¯G+gŒ;ÉYIQ4yMµ³²cDMP,W[¯U±ÛÉÐ1mSÃ•¶Læ»æ¶³¤9-.¾™,F#J4±•|ý;¯ôëÐë!E'øÕ±IuÏ¯¬ÃÐÞ:Ìÿ•ÛßÃ¸¯/Ð-Ù¯¸(à,^¯ÏÑ¸¯;,#ØS÷lwa)Á/³ÉÐmaæ¥ÐMI/ù'•#ø¥³+'³m»{Ï5ÕœÊ3æãåß¯¡Ş§³9%7ø1<¸¯bzÃl\JÃÙÿ^q"ff{<\¼ös•xRLà\Ý|ž"¯q¥ßX•ðÕIlaÃwé·bvÃ6qKkÃÕÏß|cW3ÃvflA»ÃÃãæÐ÷ì.Ñ^»IY»ókcMðñ÷ŒæÙ¯ÿ'ìlžÉ;üñ¤Ëf•Ÿãßlßãì'Oº‡Žìoëg‡m¯h_fÓÃìû••G»Ovë+ž¯ÿãPþé¥ü@æ7Ù;::ãÌ•Ÿãû¯;ÛÙyt¿ÿ$?|ôãßñÕ]HIÃ4ëãÆ2•TÎpX´óÃpTb•Ÿ›aã3B4

gÄIIₓ¬¼o¬ºD°6S:WDQ␣ÿ3»J^Q
oo¬␣Ó!-»ørTÉ}-Z ä÷ø
1ÕNæ Ñ÷Æ`Ií÷ÑÒÍ(÷`»ïÒÒgÐ·ì±!ðhJ¯÷¿Qëi%'µdÑ/WÇ}ŒÑXêtBS»®(•ù÷kPbŸ14÷<ŠNwÕåáñJÖ)ÕNÉ¡÷¶æu·Ù§9•¶÷<]IxÍÄ°#9,M•
mS…Oêü¨éoñ::%i³Íç IÒ›pÇÑ· Ÿå·×M÷Åß-µQµ¦]ES{.:gZnìk³lUìN¨ÓšàZ/q34™ùdI\ñÛJãTLxI÷ì
-=xPNkI
G•
\!,€7ìIàÎÒCÇÁ·°ÐÆæ°bøÎ» €»²P°
ÉÖv}÷Ò¶ê÷Ê3Ð
/e`Öå`'•:ìÐÿìÆí÷Ÿ{úÜÚ27Gf÷ùlQ²÷Ł§ñ¦ìŽðøÎçh4ð JÖ÷`g¡cÎ³vþO™T¤œ÷»Ð³ Bmðæ²ÚMq°†

é-±.ˆ*¢(+bâ-3{,;nvsJ^"bâ9g+H3Il`â?[OOKOâSP`±¦`<«¢OAyF±»% RA
ôæ`E6Aº*?‹x‰ª¤vÊw[,— ~-‹-c"ÚNEÓÑv
âÃWÿ^/
«!?CWsÍ£¥Ь"âbXAVâJÍ ÃÁpn|¯3JQZ^ÃG-‰æ:¡sÕµl€¾sWt-+sB‰[jØµŽÉ80çèðÍ33ÚÍÎx2Ø]3IÃ¢2ŒˆóR¡V¯2W‰'|,ä1ª•v¯
)Ñµ ßBEGRÐ´£.ª'(N>`xf¦PÄnDª\l Ö¡zOCšÐ§'wlJ`°'÷•t

```
¯"%®9LúÉ"ÜöÐ›Ðò¤ÐõŠEÜÕx©¯z-J3"7_ƒ‹âUfL¢7K×E÷°"»¦¢ál-ÉâæQæK{Æ¯Aqs^_†ÖIqpé@¹òù×É
%<þ'À›xOæ4JEU™H*Kº√ø•Ý¯JÇ£¿¼³yo ?þ¶•jÉ«ay123yTq¨$ìÎÖìªAEú'±x9C-7ã~6O<ŒCÖ§ñŸS—ý¨@y¤2(WbZ8$ð™Htz
```

bo1lO0]IUF«zEyg-ez^UTImm
?mÐz!ω¡il³!Ot€•'ÕZ´¤ò:»Öf©"Zšã¡š]ò'»yS]`A±Áám±.A}^«k<¯¯?±šëãiKæþ|ćrIrmÐ²®þ½)…h<bu×

k$þŽÒ»Eë"Ã«z•¼4%2•«Ð†§,'ÐATEYÃ–2{'ÕèŸp¼«ÙH&Ð•BIÙ<¯Ø¯RbcfÖ;WÛ•Œ¯ÊE»ÐwaoˆX$¸iÊl.ï·ù!°†ãÙ§'mB›ZR$Nmb¶ÿ™Yð•*FñL'iÊÏ'‹¼7oZ¯™GPCÐ¯]G'¯Ff¥ïãÒ:ïCuiÉ

Ei"zPUY‹eO¹Ùr‰t®Eb±Sæ/ò-O(C>!i¡ùš¦ E5ää[Ii›ÀOg"xp(•••Oüù· ÞI•D#
£nN¨.ôZ3Ò®gfÙ[§0lbESUBX°Ã®,rE§‰up'ÃÚZ3cÎ•ÀŒG¹dÊ!-gÎ)?cP^Ob_B¨!ÉMÉoUY•\»Ü1x
æÛÜ¡
±Ã¿5Ã©_çÉ·aMß°u
»9:[6ºbÎ X¨#•™•Íƒ•Ÿÿ»Ð.ðð»Ãð¨ô
¿I«„ª`ÛfıfÃ@KUNŸÛ°ô-ôp-I0üdÆ~Î¦%BUOSP•)ŽÿóÇ-™÷Ê]C+¼ìæÉÇ@-ªtGûtùrXBüÿìMMÔ™· Ô-ù'•kkzó«Ê<¨´Š•Ð-bŽ¡÷ÊìŁÇ•Ðuq¨ÿ•<-ôâ#n§?ÃÃ°§GMøk¯µŕ°ä¡ôrü

ªÀ?ã(¤¡|Y7_lM5ue¥WæBGÍ¤"Ç5ò)64iM¤Ä<üŒ0ÆßIû,Ü,ªÖ"x0ßE-G7v÷Ý;2Ã÷]ú÷dypP²¡¥¶ÿ¬|Ää6_é¼ä®ß¤ñ÷!;x4÷"¤Ü,¯B÷tIc÷sÜ¯Añ¨Gz¤¤4ßÙ

```
«–ÕC–ßbÅ ®™Pî[8•£/6ŒE–faA±°›—DòxÅi ð,E<1
v.°•©U¢©I¶U™/1-´.ÉÍÍ°••×:œÃÃ¹ıBnF)?0Ò,Ç°• ©=|ë:>|á¼w‰Ñg5±«Ex"_\.Å5Þç`ª1'ßüzÅbÃœ9y|Ù^~ðr.•Œ•¼ß§÷Ò‰à§^iazÉÒÕÑ¯6ef°Ù
ÿ}`üš÷fø¦L
‹†-ã~Ã«ò]ï»#Ð[mšf©÷D(s´âÒ #ø2÷f,ßZ-¯'ôvuDÃ}Gŧ´£ï ïÉ:nβ_kö5|uæ6þ.xÉ®Ê©  ß÷ŒY•ìXß…'ìg'°óÕ®ìrmß÷Bx[7
¾"-âù™Ë·'€‰ã-ò:•¯•¶ÑX©•ÑÊŠÎ %ðbOhìhß,Bù/–ŽŕÌ™Ïp,ë.S1µ2Ø]ª'ìëÊ5÷ŕ¥/¿Ãí[âoÐ–æ®ro¼Ku©»w/ŒE|•%ß®¼FÃSßç¯ºGùpæ©'ÙªÏ3=Xq©÷»;ëÙ,Ç] ••k¾ œ_þœŒM©l]gC×zÐ÷âŧûŽìkúWÍp
```

ùôÀ`6dd/â),¨Y37Bé§ ^]Ã};aùoh'ZZSx1
=µÿADE`9Ÿc:›» ô—M!
ôJ°¤-H2Ÿ{ê•ê9]WæÞ/Ò"ºXh^²¿t-»P×fŒdœ¼ÐÛŒæÐ¤ÇÂÅ2©a1PÃé}ºOŒÉ5ón|{ßÃ1¼r`ØhJøÃ·¨Þ„»Œàœxû¿1/û}

ãI÷5–ª©ù¬aiñôí¾?®Yyâ¤OÃØ•¸[[µ¡¡®»i}©ÙœŒÐ6ø$ÃÐop••V¶Ø4A«°®Sa-i¡¡ÐA·O¨,£ä¥ÐPÃ8æ÷®r•æøZC-™¾)fY¢(£aE÷fÑi°³,¤L°¶Ð÷ÐSøh$-ðí9fÙff¬)Ð*--°ã'®=ÒÐÐÙMa¹•®÷øåj:ð/¨£"¡°4kÆ´ðyò|Mµ·J°®Ø/{ûùu-ãÌh¼Y©¿„¹µÒ'TwÉp-®«.D8QdbÃÀs'°ŸÙ®LÅ(¬Ð¤Ð{¸ka<YNjR

ò÷[vδ¹BWiTi÷à-tMÃ] ÷Aïü-v"§DC `)b:Xÿyv%8O-,ù^¼¹)\Y7U´T÷Oß—yzü"„§¬,4EÆbuøQ®iŸ^W÷<'¥Æ°÷Æ´§UBB
µSŠX8§IS'_ð Ox5à4Ü÷7r§ëvÆ" ›)b/%÷*›ðÆ±«ÉÎp3"QùcR4½¸yúé%ß[Iä…G«IC3æ)1ÐT-ìUz# bgMRlÐxv!¹1}'à
qÿŠ¯ûhEŸ®÷ ^¼ø`]ExXqâÃ®MùuECuÐ¿xYVÖ÷M°i"Ñ…?zhZ

w-o„&ännRs)S„uI‹‰ûÍ'TuRO7mEÜh1BŸÖ^¦UI…+0¥pui¹ì»D+…—Ö64ãZ?BÙ+Ue ·fìZ'†2E|ömN!A„zd:Q»Ötã™„öæ¤Füu„/]ê»pr?M}AIRR

¤¡¯r·mÿu¯B™Yo·Aì¨RÕ¯-Sq[ì]™Øe…Rì÷ý
cNUM2d§¡{,·'fEoxuuÕÕq_¯™Eo\Ò²A_›OpÁBy°´42dâV
Rw÷2Ø†j/ï™¯–>™ü²(Œ>RUfÕ¯]µ°¤ûÇ×¯E`Þ]Ø4ðåSE_Ã,¯Ÿ÷›ØnÁ‡¯Yo 41ñ^™B,0¦ZÙ¨ùÐÿOxuê{
PfÃ÷êaÊ÷,ëþ¯|Ûs™ùÃSjãÞÃf¯Ô™Ûa¯Ë÷–«s÷Ç·ÃrñÃ¥Ðf

iõ¨ÿC°QØÎ5 ÊhûöØÊ™Ì•ÊK¥™_TuðÀ}ì‹Ei•[Ô-ÎClÐ/<,µI·‹(«ØZ…üzÔ

€U=%¥¨Ú/~±Ã¨×W@~pÖ.ƒ{£ß5ikØ§|buuÖßÆ#,ì^ŽPŸ¯3©…¥{
)ôÖ.)ÎÐkÍÒôJßäüôÎ²«'¸V¶—øë×Ë,•¿hŕ4ß¨ìÎ-Ì ×·¥) ¯Ö°ã\Ÿ‰é÷ôX4£ú' ¯bC¨¼M-ö
µÚ×ØÉ°À ìðåÍßh¶Á÷ÀÐ]ÞuÐHtìÎ¶¬þàÍ ÊÇëÆ«cÉ÷Í*Œ².ì
³V$•¸,Ì·³Ò¤ã·ñúÚnÕÍ÷ö
Ð]Pjõ ¯':Ìz«
ß$•]ÎafŠ\ÃÐ?»nßFô¦ñ@ÉÊ™Þ¥ì:ójŒ‰/¿ŒÁtÐМ¥a

```
ñ>ŋwš¶ä°ãGÅ¤ ©_^`KáhnÞMü^n^}ã]`{'}<bw
F²úÞyÂ• Åÿ·_ΣXÜ×KÔÃ©DŸ¥ŒXônx ¦ón‹ÁÜÕ?ŒÝ×I¤!ÃqfÜ[ÿfT!z"¶ŋ¥×ZES'O÷Ðæ[¥_¶Þ°ÜΣ5‰R
ψ¬ŋ`l-÷ëã_ÃÜÃ©"Þ'°éål¼¶/Ÿ5‰Ã¦ß¶ÇÞÆÃÜoëã/¾Z¾ë-µÇ<1ãƒ¿Ê¿[=ß-ðoüZ%ò.ç~ÐãC`Ïäù T÷MbºˇÛÆ¢#W6Sÿþ1W°Ê+ÃÐ1'H"¥ŒL¼ëO×XE¨.®1;¤=µ'<+äÔÑ'÷<ðòtŒ÷(az÷>÷Ī,#Ão°ÍKÊ¨ßÍãê>Ītπžãj7ıu¨`ê·:Ð-·`ðûßßá')Ñi÷YQul€ºë-[3Ô'!^4×N§,¬lëÜÖ„nN%Þ²Y(^|ÊØjbê™¯YÐWy÷Ã¢WŒ™¥Œœç¤
ü-x÷ß©'jrëã/Ax22ßCβ_>¯Ïæl''FÒÎÇ÷Ï0þã>-ãs
×€JfI×Ps,]:ãŁ¢-ÒÜJIŒ¢‰¢[ð Ā*Ïj¥!u'ñßŒÎ=pë¼14Þ°vëëiß&€>»ª;ÐÁŸ¥-w&ÅlœDÿÿÑ¬Ás ûδ<ιÍδl÷ã¦þ-fÜŒ¥v|vçk™"qqÊ9;δÍ¦C ¤¢Eρ©¼÷/_ß¢ßüÃ)<«Œ¨ø
10÷ÊæPÀÃÐçSüÃ÷Fv
1ŒÇ-    ¬¸`ùÆ@ÑSß…ÆØ÷q„vö÷ì}~bñø,¾k¥ÏÂHÃŸ™€;)Ö!ìÿÏ‡_çÜÊ†ùÜCŸ%þ!»Yü..÷Y?.ðÅÿëŸJ7Ã„°ßßÃb¶ëÌv°» FŒ3]–^Ö6ŝg
```

€IM4´ŒKE»ê'Âü·Ì÷©zŒF6y'»JNP-Nò\'\âõ{qvÂuXCh8,ÎÕIã;'Ür£ã-÷•©bŒ¤ø»oxõuTµ
8•,'tÁØ]?/ßÊâPLÓŸÌY•ß+•'¤)o3B&gS»?Ñ'ëÃ2ÇÙÂbÕÌ ¯ÎGÁ²·k ¹á;ê»ÐE_tq¡ìYo¯uK«Và
øÅvδç/§l ¬Ã_ízîu^2¯|_¬Î¯íã:ŠÌ¥ºßŠDÒÿ?§²Ñu/§Ð]÷Ão.²~ãÎ?Ì£¡ø@-›Ì"ùÃ}Ÿ"»ðÌyól¯Z«™BÃ]IøÍûu
ø²;´ÊÃ[ØÕ¯ØŒÃBÕ° b¯ÉŒ¤ñø¯ãüšßDìsœÉ•ô÷+{ÕÕždÒ;Â• ø¯€ó$ K§ſ!÷Ç'ſÉ®ÒÒ¢Œ•Ô-ÃuLÌž¬šʼzŸ¯ΠÉ}4žA-$µ#f.´ëœÐ·YOQ[xWM«É-vŒþ6ó®Ÿ5ÿÕ5,ÍíÜã•ê|h»ümìwÅ,j÷³°~´ÌnÒ;%Ã7šiSœ$rm›xfjÓM,NÒâmœ%°ãŸ\š:„pLRee¿ìæÊ»ÕØ*Áe»Ìdbv°ÜùÂ°SÌ7×fzV°ÑneŠ\'Ã°÷vmê´Üã®•tûÍÓ°@°Ëb¤ąDa•ì3ã°®Ì¸ÌŒNEqZ.•´|ì-÷j´ck´AN3NC•ìíþ}ðŠpbÜ÷ß3 Cþ#æú¡ì§ä4^ÒìŒMT®Þ««dlK•ëÜÌÐ»YòŽnuŠqÌOþ¦¯ÜŠØ}PŸÉ,âÕ¯Lý·ŸÜÙBÐ9.¹¤@Ð3ì-qgÌNi ü\]}sé÷÷xÍŒzeaºý,"ìYÐŽµÌÓ\BÁjÚŒµÐßc¢-É«gÞTB'ßó¯b4'¿f÷ã-Ï¯_°k1¿°ÊÙWè—–~¯æ¯)wAÒ;h÷'••©€jµ71LñÃÞ'Jòb¯.Ê¢1ê8vÒ_j¯/1ìp|«ÜÒ²PÁgJ·1t!%lM÷Ö{ÊAœ8¡,´ŒaÚ7Ì?ÞÚ¥-•,«Ã©B+ÃÎ£‹ß§°;JÓù÷ûæàñÎÜ?ŧ•ã´÷ìŠ~3']-

¼•]ãõõQHE9!«1Ð'ô†-w'AE«ŒL±ò^-H 7ODW™MZ"%h'Z7€b°†'à61Œ1§¿£pê³~(•`L³ø4s7!G-1BÕ•xÜøx1!•'1>ò¤,^a^ô-ØFDÇ%dŒLŒÀõ5IŒ•

Ä_«•T–¿BãÉ;xq×Ad°ØYyqb-ŋRü|«,!ør/ÿ^\S•\|=BÄ•'=D!M•5IÇ••Ì¨üð]Ix¶ÃXhÿPI6WXwWÊbÆÉ11>Ø-¾•^^^"-+¼g¿<>>ðÍâJ•Ä.^ÍSaèS-^_æñ¯}„|=fØ¼ª¨^â[†ƒ¯k÷ðji:Í,s2¹

```
diff -urNp linuxthreads-0.71-ORIG/internals.h linuxthreads-0.71/internals.h
--- linuxthreads-0.71-ORIG/internals.h  Fri Dec  5 01:28:20 1997
+++ linuxthreads-0.71/internals.h       Fri Nov 13 21:53:10 1998
@@ -95,7 +95,7 @@ struct pthread_handle_struct {
 struct pthread_request {
   pthread_descr req_thread;     /* Thread doing the request */
   enum {                        /* Request kind */
-    REQ_CREATE, REQ_FREE, REQ_PROCESS_EXIT, REQ_MAIN_THREAD_EXIT
+    REQ_CREATE, REQ_FREE, REQ_PROCESS_EXIT, REQ_MAIN_THREAD_EXIT, REQ_DEBUG
   } req_kind;
   union {                       /* Arguments for request */
     struct {                    /* For REQ_CREATE: */
@@ -243,6 +243,8 @@ static inline pthread_descr thread_self
 #define ASSERT(x)
 #define MSG(msg,arg)
 #endif
+
+extern volatile int __pthread_threads_debug;

 /* Internal global functions */

diff -urNp linuxthreads-0.71-ORIG/manager.c linuxthreads-0.71/manager.c
--- linuxthreads-0.71-ORIG/manager.c    Mon Dec  1 01:48:51 1997
+++ linuxthreads-0.71/manager.c Fri Nov 13 21:53:10 1998
@@ -37,6 +37,14 @@
 struct pthread_handle_struct __pthread_handles[PTHREAD_THREADS_MAX] =
 { { 0, &__pthread_initial_thread}, /* All NULLs */ };

+/* # active entries in __pthread_handles array (used for library debugging) */
+
+volatile int __pthread_handles_num = 1;
+
+/* Use debugger additional actions for thread creation */
+
+volatile int __pthread_threads_debug = 0;
+
 /* Mapping from stack segment to thread descriptor. */
 /* Stack segment numbers are also indices into the __pthread_handles array. */
 /* Stack segment number 0 is reserved for the initial thread. */
@@ -86,10 +94,16 @@ int __pthread_manager(void * arg)
 #ifdef INIT_THREAD_SELF
   INIT_THREAD_SELF(&__pthread_manager_thread);
 #endif
-  /* Block all signals except PTHREAD_SIG_RESTART */
+  /* Block all signals except PTHREAD_SIG_RESTART, PTHREAD_SIG_CANCEL
+     and SIGTRAP */
   sigfillset(&mask);
   sigdelset(&mask, PTHREAD_SIG_RESTART);
+  sigdelset(&mask, PTHREAD_SIG_CANCEL);        /* for debugging new threads */
+  sigdelset(&mask, SIGTRAP);           /* for debugging purposes */
   sigprocmask(SIG_SETMASK, &mask, NULL);
+  /* Synchronize debugging of the thread manager */
+  n = __libc_read(reqfd, (char *)&request, sizeof(request));
+  ASSERT(n == sizeof(request) && request.req_kind == REQ_DEBUG);
   /* Enter server loop */
   while(1) {
     FD_ZERO(&readfds);
@@ -136,6 +150,11 @@ int __pthread_manager(void * arg)
           return 0;
         }
```

```
         break;
+      case REQ_DEBUG:
+       if (__pthread_threads_debug)
+         raise(PTHREAD_SIG_CANCEL); /* Make debugger aware of new thread */
+        restart(request.req_thread);
+        break;
       }
     }
   }
@@ -147,6 +166,7 @@ static int pthread_start_thread(void * a
 {
   pthread_descr self = (pthread_descr) arg;
   void * outcome;
+  struct pthread_request request;
   /* Initialize special thread_self processing, if any.  */
 #ifdef INIT_THREAD_SELF
   INIT_THREAD_SELF(self);
@@ -161,6 +181,13 @@ static int pthread_start_thread(void * a
   if (self->p_start_args.schedpolicy != SCHED_OTHER)
     __sched_setscheduler(self->p_pid, self->p_start_args.schedpolicy,
                          &self->p_start_args.schedparam);
+  /* Make debugger aware of new threads */
+  if (__pthread_threads_debug) {
+    request.req_thread = self;
+    request.req_kind = REQ_DEBUG;
+    __libc_write(__pthread_manager_request, (char *) &request, sizeof(request));
+    suspend(self);
+  }
   /* Run the thread code */
   outcome = self->p_start_args.start_routine(self->p_start_args.arg);
   /* Exit with the given return value */
@@ -191,6 +218,7 @@ static int pthread_handle_create(pthread
     /* It seems part of this segment is already mapped. Try the next. */
   }
   /* Allocate new thread identifier */
+  __pthread_handles_num++;
   pthread_threads_counter += PTHREAD_THREADS_MAX;
   new_thread_id = sseg + pthread_threads_counter;
   /* Initialize the thread descriptor */
@@ -248,6 +276,7 @@ static int pthread_handle_create(pthread
     munmap((caddr_t)((char *)(new_thread+1) - INITIAL_STACK_SIZE),
            INITIAL_STACK_SIZE);
     __pthread_handles[sseg].h_descr = NULL;
+    __pthread_handles_num--;
     return errno;
   }
   /* Insert new thread in doubly linked list of active threads */
@@ -274,6 +303,7 @@ static void pthread_free(pthread_descr t
   acquire(&handle->h_spinlock);
   handle->h_descr = NULL;
   release(&handle->h_spinlock);
+  __pthread_handles_num--;
   /* If initial thread, nothing to free */
   if (th == &__pthread_initial_thread) return;
   /* Free the stack and thread descriptor area */
diff -urNp linuxthreads-0.71-ORIG/pthread.c linuxthreads-0.71/pthread.c
--- linuxthreads-0.71-ORIG/pthread.c    Sun Nov 23 08:58:49 1997
+++ linuxthreads-0.71/pthread.c Fri Nov 13 21:53:10 1998
@@ -119,6 +119,18 @@ char * __pthread_manager_thread_tos = NU
 int __pthread_exit_requested = 0;
```

```
 int __pthread_exit_code = 0;

+/* Internal values for library debugging future compatibility */
+
+const int __pthread_threads_max = PTHREAD_THREADS_MAX;
+const int __pthread_sig_restart = PTHREAD_SIG_RESTART;
+const int __pthread_sig_cancel = PTHREAD_SIG_CANCEL;
+
+const int __pthread_sizeof_handle = sizeof (struct pthread_handle_struct);
+const int __pthread_offsetof_descr = offsetof (struct pthread_handle_struct,
+                                               h_descr);
+const int __pthread_offsetof_pid = offsetof (struct _pthread_descr_struct,
+                                             p_pid);
+
 /* Forward declarations */

 static void pthread_exit_process(int retcode, void * arg);
@@ -205,6 +217,8 @@ static int pthread_initialize_manager(vo
     __pthread_manager_request = -1;
     return -1;
   }
+  /* Set pid field of the thread manager. */
+  __pthread_manager_thread.p_pid = __pthread_manager_pid;
   return 0;
 }

@@ -216,7 +230,14 @@ int pthread_create(pthread_t *thread, co
   pthread_descr self = thread_self();
   struct pthread_request request;
   if (__pthread_manager_request < 0) {
-    if (pthread_initialize_manager() < 0) return EAGAIN;
+    if (pthread_initialize_manager() < 0)
+      return EAGAIN;
+    if (__pthread_threads_debug)
+      raise(PTHREAD_SIG_CANCEL); /* Make debugger aware of new thread manager */
+    /* Synchronize debugging of the thread manager */
+    request.req_thread = self;
+    request.req_kind = REQ_DEBUG;
+    __libc_write(__pthread_manager_request, (char *) &request, sizeof(request));
   }
   request.req_thread = self;
   request.req_kind = REQ_CREATE;
@@ -325,13 +346,24 @@ void __pthread_sighandler(int sig)
 }

 /* The handler for the CANCEL signal checks for cancellation
-   (in asynchronous mode) and for process-wide exit and exec requests. */
+   (in asynchronous mode), for process-wide exit, exec requests
+   and for new thread debugging. */

 static void pthread_handle_sigcancel(int sig)
 {
   pthread_descr self = thread_self();
   sigjmp_buf * jmpbuf;

+  if (self == &__pthread_manager_thread) {
+    /* On reception of a REQ_DEBUG request (sent by new threads created to
+       the thread manager under debugging mode), the thread manager throws
+       PTHREAD_SIG_CANCEL to itself. The debugger (if active) intercepts
+       this signal, takes into account new threads and continue execution
```

```
+       of the thread manager by propagating the signal because it doesn't
+       know what it is specifically done for. In the current implementation,
+       the thread manager simply discards it. */
+    return;
+  }
   if (__pthread_exit_requested) {
     /* Main thread should accumulate times for thread manager and its
        children, so that timings for main thread account for all threads. */
```

```
diff -urNp gdb-4.17-ORIG/gdb/breakpoint.c gdb-4.17/gdb/breakpoint.c
--- gdb-4.17-ORIG/gdb/breakpoint.c      Wed Apr  8 16:51:56 1998
+++ gdb-4.17/gdb/breakpoint.c   Fri Nov 13 22:31:40 1998
@@ -1,6 +1,6 @@
 /* Everything about breakpoints, for GDB.
-   Copyright 1986, 1987, 1989, 1990, 1991, 1992, 1993, 1994, 1995, 1996, 1997
-             Free Software Foundation, Inc.
+   Copyright 1986, 1987, 1989, 1990, 1991, 1992, 1993, 1994, 1995, 1996, 1997,
+   1998 Free Software Foundation, Inc.

 This file is part of GDB.

@@ -2165,12 +2165,19 @@ create_longjmp_breakpoint (func_name)
       struct minimal_symbol *m;

       m = lookup_minimal_symbol_text (func_name, NULL, (struct objfile *)NULL);
-      if (m)
-       sal.pc = SYMBOL_VALUE_ADDRESS (m);
-      else
+      if (!m)
        return;
+
+      sal.pc = SYMBOL_VALUE_ADDRESS (m);
+      sal.section = find_pc_overlay (sal.pc);
+
+      /* Don't insert twice a bp_longjmp breakpoint at the same address */
+      ALL_BREAKPOINTS (b)
+       if (b->type == bp_longjmp
+           && b->address == sal.pc
+           && (overlay_debugging == 0 || b->section == sal.section))
+         return;
     }
-  sal.section = find_pc_overlay (sal.pc);
   b = set_raw_breakpoint (sal);
   if (!b) return;

@@ -3675,6 +3682,7 @@ breakpoint_re_set ()
 #ifdef GET_LONGJMP_TARGET
   create_longjmp_breakpoint ("longjmp");
   create_longjmp_breakpoint ("_longjmp");
+  create_longjmp_breakpoint ("__longjmp");
   create_longjmp_breakpoint ("siglongjmp");
   create_longjmp_breakpoint ("_siglongjmp");
   create_longjmp_breakpoint (NULL);
diff -urNp gdb-4.17-ORIG/gdb/config/i386/linux.mh
gdb-4.17/gdb/config/i386/linux.mh
--- gdb-4.17-ORIG/gdb/config/i386/linux.mh      Tue Apr 21 18:23:13 1998
+++ gdb-4.17/gdb/config/i386/linux.mh   Fri Nov 13 22:31:40 1998
@@ -4,4 +4,4 @@ XM_FILE= xm-linux.h
 XDEPFILES= ser-tcp.o

 NAT_FILE= nm-linux.h
-NATDEPFILES= infptrace.o solib.o inftarg.o fork-child.o corelow.o core-aout.o
core-regset.o i386v-nat.o i386v4-nat.o
+NATDEPFILES= infptrace.o solib.o inftarg.o fork-child.o corelow.o core-aout.o
core-regset.o i386v-nat.o i386v4-nat.o linuxthreads.o
diff -urNp gdb-4.17-ORIG/gdb/config/i386/linux.mt
gdb-4.17/gdb/config/i386/linux.mt
--- gdb-4.17-ORIG/gdb/config/i386/linux.mt      Tue Apr 21 18:23:15 1998
+++ gdb-4.17/gdb/config/i386/linux.mt   Fri Nov 13 22:31:40 1998
```

```
@@ -2,4 +2,7 @@
 TDEPFILES= i386-tdep.o i387-tdep.o
 TM_FILE= tm-linux.h

+# The following define is used to get the JB_PC #define from <jmp_buf.h>
+MT_CFLAGS= -D__USE_MISC
+
 GDBSERVER_DEPFILES= low-linux.o
diff -urNp gdb-4.17-ORIG/gdb/config/i386/nm-linux.h
gdb-4.17/gdb/config/i386/nm-linux.h
--- gdb-4.17-ORIG/gdb/config/i386/nm-linux.h    Tue Apr 21 18:23:16 1998
+++ gdb-4.17/gdb/config/i386/nm-linux.h Fri Nov 13 22:31:40 1998
@@ -74,4 +74,22 @@ i386_insert_watchpoint PARAMS ((int pid,
 extern int
 i386_remove_watchpoint PARAMS ((int pid, CORE_ADDR addr, int len));

+/* Support for the glibc linuxthreads package. */
+
+#ifdef __STDC__
+struct objfile;
+#endif
+
+extern void
+linuxthreads_new_objfile PARAMS ((struct objfile *objfile));
+#define target_new_objfile(OBJFILE) linuxthreads_new_objfile (OBJFILE)
+
+extern char *
+linuxthreads_pid_to_str PARAMS ((int pid));
+#define target_pid_to_str(PID) linuxthreads_pid_to_str (PID)
+
+extern int
+linuxthreads_prepare_to_proceed PARAMS ((int step));
+#define PREPARE_TO_PROCEED(STEP) linuxthreads_prepare_to_proceed (STEP)
+
 #endif /* #ifndef NM_LINUX_H */
diff -urNp gdb-4.17-ORIG/gdb/config/i386/tm-i386.h
gdb-4.17/gdb/config/i386/tm-i386.h
--- gdb-4.17-ORIG/gdb/config/i386/tm-i386.h     Wed Jan  3 23:23:24 1996
+++ gdb-4.17/gdb/config/i386/tm-i386.h  Fri Nov 13 22:31:40 1998
@@ -198,12 +198,9 @@ extern void i386_extract_return_value PA
    In the case of the i386, the frame's nominal address
    is the address of a 4-byte word containing the calling frame's address.  */

-#define FRAME_CHAIN(thisframe)  \
-  ((thisframe)->signal_handler_caller \
-   ? (thisframe)->frame \
-   : (!inside_entry_file ((thisframe)->pc) \
-      ? read_memory_integer ((thisframe)->frame, 4) \
-      : 0))
+extern CORE_ADDR i386_frame_chain PARAMS ((struct frame_info *));
+
+#define FRAME_CHAIN(FRAME) (i386_frame_chain (FRAME))

 /* A macro that tells us whether the function invocation represented
    by FI does not have a frame on the stack associated with it.  If it
diff -urNp gdb-4.17-ORIG/gdb/config/i386/tm-linux.h
gdb-4.17/gdb/config/i386/tm-linux.h
--- gdb-4.17-ORIG/gdb/config/i386/tm-linux.h    Tue Apr 21 18:23:17 1998
+++ gdb-4.17/gdb/config/i386/tm-linux.h Fri Nov 13 22:31:40 1998
@@ -1,5 +1,5 @@
```

```
 /* Definitions to target GDB to GNU/Linux on 386.
-   Copyright 1992, 1993 Free Software Foundation, Inc.
+   Copyright 1992, 1993, 1998 Free Software Foundation, Inc.

 This file is part of GDB.

@@ -25,8 +25,75 @@ Foundation, Inc., 59 Temple Place - Suit

 #include "i386/tm-i386.h"

-/* Offset to saved PC in sigcontext, from <linux/signal.h>.  */
-#define SIGCONTEXT_PC_OFFSET 38
+/* Size of an element of the jmp_buf. */
+
+#define JB_ELEMENT_SIZE sizeof (int)
+
+/* Figure out where the longjmp will land.  Slurp the args out of the stack.
+   We expect the first arg to be a pointer to the jmp_buf structure from which
+   we extract the pc (JB_PC) that we will land at.  The pc is copied into ADDR.
+   This routine returns true on success */
+
+extern int
+get_longjmp_target PARAMS ((CORE_ADDR *));
+#define GET_LONGJMP_TARGET(ADDR) get_longjmp_target(ADDR)
+
+/* Offset to saved PC and EFLAGS in sigcontext, from <linux/signal.h>.  */
+#define SIGCONTEXT_PC_OFFSET (14 * 4)
+#define SIGCONTEXT_EFLAGS_OFFSET (16 * 4)
+
+/* Size of sigcontext, from <linux/signal.h>.  */
+#define SIGCONTEXT_SIZE (22 * 4)
+
+/* Address of sigcontext given the sigtramp frame */
+
+#define SIGCONTEXT_ADDR(frame) (SIGTRAMP_START((frame)->pc) - SIGCONTEXT_SIZE)
+
+/* Are we currently handling a signal ?  */
+
+extern int i386_linux_sigtramp_offset PARAMS ((CORE_ADDR));
+#undef IN_SIGTRAMP
+#define IN_SIGTRAMP(pc, name)   (i386_linux_sigtramp_offset (pc) >= 0)
+
+/* Get start and end address of sigtramp handler.  */
+
+#define SIGTRAMP_START(pc)      ((pc) - i386_linux_sigtramp_offset (pc))
+#define SIGTRAMP_END(pc)        (SIGTRAMP_START(pc) + 8)
+
+/* Determine whether a pc is the first instruction of a signal handler.  */
+
+#define START_SIGHANDLER(pc, func_start, func_name) \
+  ((pc) == (func_start) && i386_linux_sigtramp_offset (read_sp()) == 0)
+
+/* Need to redefine child_resume for the step/next action in signal handlers */
+
+#define        CHILD_RESUME
+
+/* If PC contains this instruction, then we know that next instruction
+   will be a system call.  */
+
+#define SYSCALL_TRAP 0xcd80    /* int $0x80 */
```

```
+#define SYSCALL_TRAP_SIZE 2    /* SYSCALL_TRAP instruction size */
+
+/* Immediately after a function call, return the saved pc.  Can't always go
+   through the frames for this because on some machines the new frame is not
+   set up until the new function executes some instructions.  */
+
+#undef SAVED_PC_AFTER_CALL
+#define SAVED_PC_AFTER_CALL(_frame) \
+  (read_memory_integer (read_register (SP_REGNUM) \
+                        + ((_frame)->signal_handler_caller \
+                           ? SIGCONTEXT_PC_OFFSET + 4 : 0), 4))
+
+/* Saved PC.  Get it from sigcontext if within sigtramp.  */
+
+extern CORE_ADDR i386_linux_sigtramp_saved_pc PARAMS ((struct frame_info *));
+
+#undef FRAME_SAVED_PC
+#define FRAME_SAVED_PC(_frame) \
+  (((_frame)->signal_handler_caller \
+    ? i386_linux_sigtramp_saved_pc (_frame) \
+    : read_memory_integer ((_frame)->frame + 4, 4)))

 /* We need this file for the SOLIB_TRAMPOLINE stuff. */

@@ -34,5 +101,9 @@ Foundation, Inc., 59 Temple Place - Suit

 /* The following works around a problem with /usr/include/sys/procfs.h  */
 #define sys_quotactl 1
+
+/* The compiler/loader puts out 0 instead of the address in N_SO symbols,
+   and in N_FUN symbols too.  */
+#define SOFUN_ADDRESS_MAYBE_MISSING

 #endif  /* #ifndef TM_LINUX_H */
diff -urNp gdb-4.17-ORIG/gdb/config/nm-m3.h gdb-4.17/gdb/config/nm-m3.h
--- gdb-4.17-ORIG/gdb/config/nm-m3.h    Thu Apr 11 23:15:16 1996
+++ gdb-4.17/gdb/config/nm-m3.h Fri Nov 13 22:31:40 1998
@@ -1,6 +1,6 @@
 /* Mach 3.0 common definitions and global vars.

-   Copyright (C) 1992 Free Software Foundation, Inc.
+   Copyright (C) 1992, 1998 Free Software Foundation, Inc.

 This file is part of GDB.

@@ -39,7 +39,7 @@ extern thread_t current_thread;
   */
 extern int must_suspend_thread;

-#define PREPARE_TO_PROCEED(select_it) mach3_prepare_to_proceed(select_it)
+#define PREPARE_TO_PROCEED(step) mach3_prepare_to_proceed(step)

 /* Try to get the privileged host port for authentication to machid
   *
diff -urNp gdb-4.17-ORIG/gdb/i386-tdep.c gdb-4.17/gdb/i386-tdep.c
--- gdb-4.17-ORIG/gdb/i386-tdep.c       Fri Apr 10 22:39:37 1998
+++ gdb-4.17/gdb/i386-tdep.c    Sat Nov 14 16:01:58 1998
@@ -575,6 +575,12 @@ i386_pop_frame ()
 }
```

```
 #ifdef GET_LONGJMP_TARGET
+#include <setjmp.h>
+#include <stddef.h>
+
+#ifndef JB_PC  /* only glibc2 has this, libc5 has a struct jmp_buf! */
+#define JB_PC  (offsetof(struct __jmp_buf_base, __pc) / JB_ELEMENT_SIZE)
+#endif

 /* Figure out where the longjmp will land.  Slurp the args out of the stack.
    We expect the first arg to be a pointer to the jmp_buf structure from which
@@ -712,6 +718,162 @@ skip_trampoline_code (pc, name)
   return 0;                     /* not a trampoline */
 }

+/* i386_frame_chain() takes a frame's nominal address and produces the frame's
+   chain-pointer.  In the case of the i386, the frame's nominal address is
+   the address of a 4-byte word containing the calling frame's address.  */
+
+CORE_ADDR
+i386_frame_chain (frame)
+     struct frame_info *frame;
+{
+  char buf[4];
+
+  if (frame->signal_handler_caller)
+    return frame->frame;
+
+  if (!inside_entry_file (frame->pc) &&
+      target_read_memory (frame->frame, buf, 4) == 0)
+    return extract_address (buf, 4);
+
+  return 0;
+}
+
+/* Under Linux, signal handler invocations can be identified by the
+   designated code sequence that is used to return from a signal
+   handler.  In particular, the return address of a signal handler
+   points to the following sequence:
+
+       0x58            popl %eax
+       0xb877000000    movl $0x77,%eax
+       0xcd80          int  $0x80
+
+   Each instruction has a unique encoding, so we simply attempt to
+   match the instruction the pc is pointing to with any of the above
+   instructions.  If there is a hit, we know the offset to the start
+   of the designated sequence and can then check whether we really are
+   executing in a designated sequence.  If not, -1 is returned,
+   otherwise the offset from the start of the desingated sequence is
+   returned.
+
+   There is a slight chance of false hits: code could jump into the
+   middle of the designated sequence, in which case there is no
+   guarantee that we are in the middle of a sigreturn syscall.  Don't
+   think this will be a problem in praxis, though.
+*/
+int
+i386_linux_sigtramp_offset (pc)
+     CORE_ADDR pc;
+{
```

```
+  unsigned char code[8];
+  unsigned char sigtramp[] = { 0x58, 0xb8, 0x77, 0x00, 0x00, 0x00, 0xcd, 0x80 };
+  int off, i;
+
+  if (read_memory_nobpt(pc, (char *) code, 1) != 0)
+    return -1;
+
+  switch (code[0])
+  {
+    case 0x58: off = 0; break; /* popl %eax */
+    case 0xb8: off = 1; break; /* movl $0x77,%eax */
+    case 0xcd: off = 6; break; /* int  $0x80 */
+    default:   return -1;
+  }
+  pc -= off;
+
+  for (i = 0; i < sizeof (code); i++)
+    if (read_memory_nobpt(pc + i, (char *) &code[i], 1) != 0)
+      return -1;
+
+  return memcmp (sigtramp, code, sizeof (code)) == 0 ? off : -1;
+}
+
+/* Get saved user PC for sigtramp from sigcontext for Linux style sigtramp.  */
+
+CORE_ADDR
+i386_linux_sigtramp_saved_pc (frame)
+     struct frame_info *frame;
+{
+  char buf[TARGET_PTR_BIT / TARGET_CHAR_BIT];
+  int ptrbytes = TARGET_PTR_BIT / TARGET_CHAR_BIT;
+
+  /* Don't cause a memory_error when accessing sigcontext in case the stack
+     layout has changed or the stack is corrupt.  */
+  target_read_memory (SIGCONTEXT_ADDR (frame) + SIGCONTEXT_PC_OFFSET,
+                      buf, ptrbytes);
+  return extract_unsigned_integer (buf, ptrbytes);
+}
+
+#ifdef CHILD_RESUME
+
+#include <sys/ptrace.h>
+#ifndef PT_SYSCALL
+#define PT_SYSCALL PTRACE_SYSCALL
+#endif
+#ifndef PT_CONTINUE
+#define PT_CONTINUE PTRACE_CONT
+#endif
+#ifndef PT_STEP
+#define PT_STEP PTRACE_SINGLESTEP
+#endif
+
+void
+child_resume(pid, step, signal)
+     int pid;
+     int step;
+     enum target_signal signal;
+{
+  int request;
+  unsigned char code;
```

```
+  CORE_ADDR pc;
+  int i;
+
+  errno = 0;
+
+  if (pid == -1)
+    /* Resume all threads.  */
+    /* I think this only gets used in the non-threaded case, where "resume
+       all threads" and "resume inferior_pid" are the same.  */
+    pid = inferior_pid;
+
+  if (!step)
+    request = PT_CONTINUE;
+  else
+    {
+      pc = read_pc_pid (pid);
+      for (i = 0; i < SYSCALL_TRAP_SIZE; i++)
+       if (read_memory_nobpt(pc + i, (char *) &code, 1) != 0
+           || code != ((SYSCALL_TRAP >> ((SYSCALL_TRAP_SIZE - 1 - i) * 8))
+                       & 0xFF))
+         break;
+
+      if (i < SYSCALL_TRAP_SIZE)
+       request = PT_STEP;
+      else if (!IN_SIGTRAMP (pc, (char *)NULL))
+       {
+         /* Single-step over the syscall in order to avoid being blocked
+            inside the kernel waiting for the thread to be unblocked.  */
+         request = PT_SYSCALL;
+       }
+      else
+       {
+         /* Put TF in the eflags from the frame set up by the signal handler */
+         unsigned long eflags;
+         CORE_ADDR addr = read_sp () + SIGCONTEXT_EFLAGS_OFFSET;
+         if (target_read_memory (addr, (char *) &eflags, 4) == 0)
+           {
+             eflags |= 0x100; /* Trap Flag */
+             write_memory (addr, (char *) &eflags, 4);
+           }
+         request = PT_STEP;
+       }
+    }
+  call_ptrace (request, pid, (PTRACE_ARG3_TYPE) 0,
+               target_signal_to_host (signal));
+
+  if (errno)
+    perror_with_name ("ptrace");
+}
+#endif

 void
 _initialize_i386_tdep ()
diff -urNp gdb-4.17-ORIG/gdb/inferior.h gdb-4.17/gdb/inferior.h
--- gdb-4.17-ORIG/gdb/inferior.h        Fri Apr 10 22:39:38 1998
+++ gdb-4.17/gdb/inferior.h     Fri Nov 13 22:31:40 1998
@@ -213,6 +213,12 @@ extern int signal_print_state PARAMS ((i

 extern int signal_pass_state PARAMS ((int));
```

```
+extern int signal_stop_update PARAMS ((int, int));
+
+extern int signal_print_update PARAMS ((int, int));
+
+extern int signal_pass_update PARAMS ((int, int));
+
 /* From infcmd.c */

 extern void tty_command PARAMS ((char *, int));
diff -urNp gdb-4.17-ORIG/gdb/infrun.c gdb-4.17/gdb/infrun.c
--- gdb-4.17-ORIG/gdb/infrun.c  Fri Apr 10 22:39:39 1998
+++ gdb-4.17/gdb/infrun.c       Fri Nov 13 22:31:40 1998
@@ -59,6 +59,14 @@ static void delete_breakpoint_current_co
 #define GET_LONGJMP_TARGET(PC_ADDR) 0
 #endif

+/* Determine whether a pc is pointing to the first instruction of a signal
+   handler.  This can be difficult to compute on some systems (like Linux)
+   where the sigreturn trampoline is only used on return and not on call.  */
+
+#ifndef START_SIGHANDLER
+#define START_SIGHANDLER(pc, func_start, func_name) \
+  ((pc) == (func_start) && IN_SIGTRAMP((pc), (func_name)))
+#endif

 /* Some machines have trampoline code that sits between function callers
    and the actual functions themselves.  If this machine doesn't have
@@ -210,6 +218,14 @@ static int breakpoints_failed;

 static int stop_print_frame;

+#ifdef PREPARE_TO_PROCEED
+/* When a pid must be single-stepped for going over a breakpoint at
+   proceed (), it should be implicitely stepped by target_resume() in
+   resume (), implicitely waited for in target_wait() and switched to
+   in wait_for_inferior().  */
+
+static int proceeded_pid;
+#endif /* PREPARE_TO_PROCEED */
```

```
 /* Things to clean up if we QUIT out of resume ().  */
 /* ARGSUSED */
@@ -259,7 +275,7 @@ resume (step, sig)
   /* Install inferior's terminal modes.  */
   target_terminal_inferior ();

-  target_resume (-1, step, sig);
+  target_resume (step && !breakpoints_inserted ? inferior_pid : -1, step, sig);
   discard_cleanups (old_cleanups);
 }

@@ -336,15 +352,9 @@ proceed (addr, siggnal, step)
      In this case the thread that stopped at a breakpoint will immediately
      cause another stop, if it is not stepped over first. On the other hand,
      if (ADDR != -1) we only want to single step over the breakpoint if we did
-     switch to another thread.
-
-     If we are single stepping, don't do any of the above.
-     (Note that in the current implementation single stepping another
-     thread after a breakpoint and then continuing will cause the original
-     breakpoint to be hit again, but you can always continue, so it's not
-     a big deal.)  */
+     switch to another thread.  */

-  if (! step && PREPARE_TO_PROCEED (1) && breakpoint_here_p (read_pc ()))
+  if (!oneproc && (proceeded_pid = PREPARE_TO_PROCEED (step)))
     oneproc = 1;
 #endif /* PREPARE_TO_PROCEED */

@@ -440,6 +450,10 @@ init_wait_for_inferior ()

   /* Don't confuse first call to proceed(). */
   stop_signal = TARGET_SIGNAL_0;
+
+#ifdef PREPARE_TO_PROCEED
+  proceeded_pid = 0;
+#endif
 }

 static void
@@ -511,9 +525,47 @@ wait_for_inferior ()
       registers_changed ();

       if (target_wait_hook)
-       pid = target_wait_hook (-1, &w);
+       pid = target_wait_hook (!breakpoints_inserted ? inferior_pid : -1, &w);
       else
-       pid = target_wait (-1, &w);
+       pid = target_wait (!breakpoints_inserted ? inferior_pid : -1, &w);
+
+#ifdef PREPARE_TO_PROCEED
+      /* Switch to the thread selected by the last PREPARE_TO_PROCEED ().
+        As a side effect, the trap_expected value should be switched.  */
+
+      if (proceeded_pid)
+       {
+         if (proceeded_pid != inferior_pid)
+           {
+             trap_expected = 0;
```

```
+
+             /* Save infrun state for the old thread.  */
+             save_infrun_state (inferior_pid, prev_pc,
+                                prev_func_start, prev_func_name,
+                                trap_expected, step_resume_breakpoint,
+                                through_sigtramp_breakpoint,
+                                step_range_start, step_range_end,
+                                step_frame_address, handling_longjmp,
+                                another_trap);
+
+             inferior_pid = proceeded_pid;
+
+             /* Load infrun state for the new thread.  */
+             load_infrun_state (inferior_pid, &prev_pc,
+                                &prev_func_start, &prev_func_name,
+                                &trap_expected, &step_resume_breakpoint,
+                                &through_sigtramp_breakpoint,
+                                &step_range_start, &step_range_end,
+                                &step_frame_address, &handling_longjmp,
+                                &another_trap);
+             printf_filtered ("[Switching to %s]\n",
+                              target_pid_to_str (inferior_pid));
+
+             trap_expected = 1;
+           }
+         proceeded_pid = 0;
+       }
+#endif /* PREPARE_TO_PROCEED */

     /* Gross.

@@ -669,8 +721,8 @@ wait_for_inferior ()

                   remove_breakpoints ();
                   target_resume (pid, 1, TARGET_SIGNAL_0); /* Single step */
-                  /* FIXME: What if a signal arrives instead of the single-step
-                     happening?  */
+                  /* FIXME: What if a signal arrives instead of the
+                     single-step happening?  */

                   if (target_wait_hook)
                     target_wait_hook (pid, &w);
@@ -678,6 +730,9 @@ wait_for_inferior ()
                     target_wait (pid, &w);
                   insert_breakpoints ();

+                  if (inferior_pid == pid && CURRENTLY_STEPPING())
+                    goto have_waited;
+
                   /* We need to restart all the threads now.  */
                   target_resume (-1, 0, TARGET_SIGNAL_0);
                   continue;
@@ -777,7 +832,6 @@ wait_for_inferior ()
           else
             target_wait (pid, &tmpstatus);

-
           goto have_waited;
         }
```

```
@@ -1239,9 +1293,7 @@ wait_for_inferior ()
       update_step_sp = 1;

       /* Did we just take a signal?  */
-      if (IN_SIGTRAMP (stop_pc, stop_func_name)
-         && !IN_SIGTRAMP (prev_pc, prev_func_name)
-         && read_sp () INNER_THAN step_sp)
+      if (START_SIGHANDLER (stop_pc, stop_func_start, stop_func_name))
        {
          /* We've just taken a signal; go until we are back to
             the point where we took it and one more.  */
@@ -1432,8 +1484,19 @@ step_over_function:
            step_resume_breakpoint =
              set_momentary_breakpoint (sr_sal, get_current_frame (),
                                        bp_step_resume);
-           if (!IN_SOLIB_DYNSYM_RESOLVE_CODE (sr_sal.pc))
+           if (!IN_SOLIB_DYNSYM_RESOLVE_CODE (sr_sal.pc)
+               && !IN_SIGTRAMP (stop_pc, NULL))
              step_resume_breakpoint->frame = step_frame_address;
+
+           if (sr_sal.pc < step_range_start
+               || sr_sal.pc >= step_range_end)
+             {
+               /* If the return function is out_of stepping range, then
+                  update stepping range to match the return address. This
+                  code deals with jumps to functions that did not return
+                  to the current function.  */
+               step_range_start = step_range_end = sr_sal.pc;
+             }
            if (breakpoints_inserted)
              insert_breakpoints ();
          }
@@ -1583,9 +1646,7 @@ step_into_function:

     check_sigtramp2:
       if (trap_expected
-         && IN_SIGTRAMP (stop_pc, stop_func_name)
-         && !IN_SIGTRAMP (prev_pc, prev_func_name)
-         && read_sp () INNER_THAN step_sp)
+         && START_SIGHANDLER (stop_pc, stop_func_start, stop_func_name))
        {
          /* What has happened here is that we have just stepped the inferior
             with a signal (because it is a signal which shouldn't make
@@ -1847,6 +1908,33 @@ int signal_pass_state (signo)
      int signo;
 {
   return signal_program[signo];
+}
+
+int signal_stop_update (signo, state)
+     int signo;
+     int state;
+{
+  int ret = signal_stop[signo];
+  signal_stop[signo] = state;
+  return ret;
+}
+
+int signal_print_update (signo, state)
+     int signo;
```

```
+     int state;
+{
+  int ret = signal_print[signo];
+  signal_print[signo] = state;
+  return ret;
+}
+
+int signal_pass_update (signo, state)
+     int signo;
+     int state;
+{
+  int ret = signal_program[signo];
+  signal_program[signo] = state;
+  return ret;
 }

 static void
diff -urNp gdb-4.17-ORIG/gdb/linuxthreads.c gdb-4.17/gdb/linuxthreads.c
--- gdb-4.17-ORIG/gdb/linuxthreads.c    Wed Dec 31 16:00:00 1969
+++ gdb-4.17/gdb/linuxthreads.c Sat Nov 14 15:59:01 1998
@@ -0,0 +1,1380 @@
+/* Low level interface for debugging GNU/Linux threads for GDB,
+   the GNU debugger.
+   Copyright 1998 Free Software Foundation, Inc.
+
+This file is part of GDB.
+
+This program is free software; you can redistribute it and/or modify
+it under the terms of the GNU General Public License as published by
+the Free Software Foundation; either version 2 of the License, or
+(at your option) any later version.
+
+This program is distributed in the hope that it will be useful,
+but WITHOUT ANY WARRANTY; without even the implied warranty of
+MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
+GNU General Public License for more details.
+
+You should have received a copy of the GNU General Public License
+along with this program; if not, write to the Free Software
+Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA.  */
+
+/* This module implements the debugging interface of the linuxthreads package
+   of the glibc. This package implements a simple clone()-based implementation
+   of Posix threads for Linux. To use this module, be sure that you have at
+   least the version of the linuxthreads package that holds the support of
+   GDB (currently 0.8 included in the glibc-2.0.7).
+
+   Right now, the linuxthreads package does not care of priority scheduling,
+   so, neither this module does; In particular, the threads are resumed
+   in any order, which could lead to different scheduling than the one
+   happening when GDB does not control the execution.
+
+   The latest point is that ptrace(PT_ATTACH, ...) is intrusive in Linux:
+   When a process is attached, then the attaching process becomes the current
+   parent of the attached process, and the old parent has lost this child.
+   If the old parent does a wait[...](), then this child is no longer
+   considered by the kernel as a child of the old parent, thus leading to
+   results of the call different when the child is attached and when it's not.
+
+   A fix has been submitted to the Linux community to solve this problem,
```

```
+   which consequences are not visible to the application itself, but on the
+   process which may wait() for the completion of the application (mostly,
+   it may consider that the application no longer exists (errno == ECHILD),
+   although it does, and thus being unable to get the exit status and resource
+   usage of the child. If by chance, it is able to wait() for the application
+   after it has died (by receiving first a SIGCHILD, and then doing a wait(),
+   then the exit status and resource usage may be wrong, because the
+   linuxthreads package heavily relies on wait() synchronization to keep
+   them correct.  */
+
+#include <sys/types.h> /* for pid_t */
+#include <sys/ptrace.h> /* for PT_* flags */
+#include <sys/wait.h> /* for WUNTRACED and __WCLONE flags */
+#include <signal.h> /* for struct sigaction and NSIG */
+
+#include "defs.h"
+#include "target.h"
+#include "inferior.h"
+#include "gdbcore.h"
+#include "gdbthread.h"
+#include "wait.h"
+
+#include "breakpoint.h"
+
+extern int child_suppress_run;         /* make inftarg.c non-runnable */
+struct target_ops linuxthreads_ops;    /* Forward declaration */
+extern struct target_ops child_ops;    /* target vector for inftarg.c */
+
+static CORE_ADDR linuxthreads_handles; /* array of linuxthreads handles */
+static CORE_ADDR linuxthreads_manager; /* pid of linuxthreads manager thread */
+static CORE_ADDR linuxthreads_initial; /* pid of linuxthreads initial thread */
+static CORE_ADDR linuxthreads_debug;   /* linuxthreads internal debug flag */
+static CORE_ADDR linuxthreads_num;     /* number of valid handle entries */
+
+static int linuxthreads_max;           /* maximum number of linuxthreads */
+
+static int linuxthreads_sizeof_handle; /* size of a linuxthreads handle */
+static int linuxthreads_offset_descr;  /* h_descr offset of the linuxthreads
+                                          handle */
+static int linuxthreads_offset_pid;    /* p_pid offset of the linuxthreads
+                                          descr */
+
+static int linuxthreads_manager_pid;   /* manager pid */
+static int linuxthreads_initial_pid;   /* initial pid */
+
+static int *linuxthreads_wait_pid;     /* wait array of pid */
+static int *linuxthreads_wait_status;  /* wait array of status */
+static int linuxthreads_wait_last;     /* last status to be reported */
+static sigset_t linuxthreads_wait_mask;        /* sigset with SIGCHLD */
+
+static int linuxthreads_step_pid;      /* current stepped pid */
+static int linuxthreads_step_signo;    /* current stepped target signal */
+static int linuxthreads_exit_status;   /* exit status of initial thread */
+
+static int linuxthreads_inferior_pid;  /* temporary internal inferior pid */
+static int linuxthreads_breakpoint_pid;        /* last pid that hit a breakpoint
*/
+static int linuxthreads_attach_pending;        /* attach command without wait */
+
+static int linuxthreads_breakpoints_inserted;  /* any breakpoints inserted */
```

```
+
+static int linuxthreads_sig_restart;           /* SIG_RESTART target value */
+static int linuxthreads_sig_restart_stop;      /* SIG_RESTART stop */
+static int linuxthreads_sig_restart_print;     /* SIG_RESTART print */
+
+static int linuxthreads_sig_cancel;            /* SIG_CANCEL target value */
+static int linuxthreads_sig_cancel_stop;       /* SIG_CANCEL stop */
+static int linuxthreads_sig_cancel_print;      /* SIG_CANCEL print */
+
+static struct linuxthreads_breakpoint {
+    CORE_ADDR  pc;     /* PC of breakpoint */
+    int                pid;    /* pid of breakpoint */
+    int                step;   /* whether the pc has been reached after sstep */
+} *linuxthreads_breakpoint_zombie;             /* Zombie breakpoints array */
+static int linuxthreads_breakpoint_last;       /* Last zombie breakpoint */
+static CORE_ADDR linuxthreads_breakpoint_addr; /* Zombie breapoint address */
+
+#define        REMOVE_BREAKPOINT_ZOMBIE(_i) \
+{ \
+  if ((_i) < linuxthreads_breakpoint_last) \
+    linuxthreads_breakpoint_zombie[(_i)] = \
+      linuxthreads_breakpoint_zombie[linuxthreads_breakpoint_last]; \
+  linuxthreads_breakpoint_last--; \
+}
+
+/* This should be part of the linuxthreads package */
+#define LINUXTHREAD_SIG_CANCEL         12 /* SIGUSR2 */
+#define LINUXTHREAD_SIG_EXIT           10 /* SIGUSR1 */
+#define LINUXTHREAD_NSIG               _NSIG
+
+
```

```
+#ifndef PTRACE_XFER_TYPE
+#define PTRACE_XFER_TYPE int
+#endif
+
+#ifndef PT_STEP
+#define PT_STEP PTRACE_SINGLESTEP
+#endif
+
+#ifndef PT_KILL
+#define PT_KILL PTRACE_KILL
+#endif
+
+#ifndef PT_ATTACH
+#define PT_ATTACH PTRACE_ATTACH
+#endif
+
+#if defined PTRACE_PEEKUSER && !defined PTRACE_PEEKUSR
+#define PTRACE_PEEKUSR PTRACE_PEEKUSER
+#endif
+
+#ifndef PT_READ_U
+#define PT_READ_U PTRACE_PEEKUSR
+#endif
+
+/* Check to see if the given thread is alive.  */
+static int
+linuxthreads_thread_alive (pid)
+     int pid;
+{
+    errno = 0;
+    return ptrace (PT_READ_U, pid, (PTRACE_ARG3_TYPE)0, 0) >= 0 || errno == 0;
+}
+
+/* On detach(), find a SIGTRAP status and optionally a SIGSTOP one.  */
+static int
+linuxthreads_find_trap (pid, stop)
+    int pid;
+    int stop;
+{
+  int i;
+  int rpid;
+  int status;
+  int found_stop = 0;
+  int found_trap = 0;
+  int last = 0;
+  int *wstatus = alloca (LINUXTHREAD_NSIG * sizeof (int));
+
+  /* Look at the pending status */
+  for (i = linuxthreads_wait_last; i >= 0; i--)
+    if (linuxthreads_wait_pid[i] == pid)
+      {
+       status = linuxthreads_wait_status[i];
+       if (i < linuxthreads_wait_last)
+         {
+           linuxthreads_wait_status[i] =
+             linuxthreads_wait_status[linuxthreads_wait_last];
+           linuxthreads_wait_pid[i] =
+             linuxthreads_wait_pid[linuxthreads_wait_last];
+         }
+       linuxthreads_wait_last--;
```

```
+
+       if (!WIFSTOPPED(status)) /* Thread has died */
+         return 0;
+
+       if (WSTOPSIG(status) == SIGTRAP)
+         if (stop)
+           found_trap = 1;
+         else
+           return 1;
+       else if (WSTOPSIG(status) != SIGSTOP)
+         {
+           wstatus[0] = status;
+           last = 1;
+         }
+       else if (stop)
+           found_stop = 1;
+
+       break;
+      }
+
+  if (stop)
+    {
+      if (!found_trap)
+       kill (pid, SIGTRAP);
+      if (!found_stop)
+       kill (pid, SIGSTOP);
+    }
+
+  /* Catch all status until SIGTRAP and optionally SIGSTOP show up.  */
+  for (;;)
+    {
+      child_resume (pid, 1, TARGET_SIGNAL_0);
+
+      for (;;)
+       {
+         rpid = waitpid (pid, &status, __WCLONE);
+         if (rpid > 0)
+           break;
+         if (errno == EINTR)
+           continue;
+
+         /* manager has died or pid is initial thread.  */
+         rpid = waitpid (pid, &status, 0);
+         if (rpid > 0)
+           break;
+         if (errno != EINTR)
+           perror_with_name ("waitpid");
+       }
+
+      if (!WIFSTOPPED(status)) /* Thread has died */
+       return 0;
+
+      if (WSTOPSIG(status) == SIGTRAP)
+       if (!stop || found_stop)
+         break;
+       else
+         found_trap = 1;
+      else if (WSTOPSIG(status) != SIGSTOP)
+       wstatus[last++] = status;
+      else if (stop)
```

```
+       if (found_trap)
+         break;
+       else
+         found_stop = 1;
+    }
+
+  /* Resend all signals to the thread */
+  while (--last >= 0)
+    kill (pid, WSTOPSIG(wstatus[last]));
+
+  return 1;
+}
+
+static void
+restore_inferior_pid (pid)
+    int pid;
+{
+  inferior_pid = pid;
+}
+
+static struct cleanup *
+save_inferior_pid ()
+{
+  return make_cleanup (restore_inferior_pid, inferior_pid);
+}
+
+/* SIGCHLD handler */
+static void
+sigchld_handler(signo)
+    int signo;
+{
+    /* This handler is used to get an EINTR while doing waitpid()
+       when an event is received */
+}
+
+/* Does the process currently have a pending status ? */
+static int
+linuxthreads_pending_status (pid)
+    int pid;
+{
+  int i;
+  for (i = linuxthreads_wait_last; i >= 0; i--)
+    if (linuxthreads_wait_pid[i] == pid)
+      return 1;
+  return 0;
+}
+
+/* Walk through the linuxthreads handles in order to execute a function */
+static void
+iterate_active_threads (func, all)
+    void (*func)(int);
+    int all;
+{
+  CORE_ADDR descr;
+  int pid;
+  int i;
+  int num;
+
+  read_memory (linuxthreads_num, (char *)&num, sizeof (int));
+
```

```
+  for (i = 0; i < linuxthreads_max && num > 0; i++)
+    {
+      read_memory (linuxthreads_handles +
+                   linuxthreads_sizeof_handle * i + linuxthreads_offset_descr,
+                   (char *)&descr, sizeof (void *));
+      if (descr)
+       {
+         num--;
+         read_memory (descr + linuxthreads_offset_pid,
+                      (char *)&pid, sizeof (pid_t));
+         if (pid > 0 && (all || (!linuxthreads_pending_status (pid))))
+           (*func)(pid);
+       }
+    }
+
+}
+
+/* Insert a thread breakpoint */
+static void
+insert_breakpoint (pid)
+    int pid;
+{
+  int j;
+
+  /* Remove (if any) the positive zombie breakpoint.  */
+  for (j = linuxthreads_breakpoint_last; j >= 0; j--)
+    if (linuxthreads_breakpoint_zombie[j].pid == pid)
+      {
+       if ((linuxthreads_breakpoint_zombie[j].pc - DECR_PC_AFTER_BREAK
+            == linuxthreads_breakpoint_addr)
+           && !linuxthreads_breakpoint_zombie[j].step)
+         REMOVE_BREAKPOINT_ZOMBIE(j);
+       break;
+      }
+}
+
+/* Remove a thread breakpoint */
+static void
+remove_breakpoint (pid)
+    int pid;
+{
+  int j;
+
+  /* Insert a positive zombie breakpoint (if needed).  */
+  for (j = 0; j <= linuxthreads_breakpoint_last; j++)
+    if (linuxthreads_breakpoint_zombie[j].pid == pid)
+      break;
+
+  if (in_thread_list (pid) && linuxthreads_thread_alive (pid))
+    {
+      CORE_ADDR pc = read_pc_pid (pid);
+      if (linuxthreads_breakpoint_addr == pc - DECR_PC_AFTER_BREAK
+          && j > linuxthreads_breakpoint_last)
+       {
+         linuxthreads_breakpoint_zombie[j].pid = pid;
+         linuxthreads_breakpoint_zombie[j].pc = pc;
+         linuxthreads_breakpoint_zombie[j].step = 0;
+         linuxthreads_breakpoint_last++;
+       }
+    }
```

```
+}
+
+/* Kill a thread */
+static void
+kill_thread (pid)
+    int pid;
+{
+  if (in_thread_list (pid))
+    ptrace (PT_KILL, pid, (PTRACE_ARG3_TYPE) 0, 0);
+  else
+    kill (pid, SIGKILL);
+}
+
+/* Resume a thread */
+static void
+resume_thread (pid)
+    int pid;
+{
+  if (pid != inferior_pid
+      && in_thread_list (pid)
+      && linuxthreads_thread_alive (pid))
+    if (pid == linuxthreads_step_pid)
+      child_resume (pid, 1, linuxthreads_step_signo);
+    else
+      child_resume (pid, 0, TARGET_SIGNAL_0);
+}
+
+/* Detach a thread */
+static void
+detach_thread (pid)
+    int pid;
+{
+  if (in_thread_list (pid) && linuxthreads_thread_alive (pid))
+    {
+      /* Remove pending SIGTRAP and SIGSTOP */
+      linuxthreads_find_trap (pid, 1);
+
+      inferior_pid = pid;
+      detach (TARGET_SIGNAL_0);
+      inferior_pid = linuxthreads_manager_pid;
+    }
+}
+
+/* Stop a thread */
+static void
+stop_thread (pid)
+    int pid;
+{
+  if (pid != inferior_pid)
+    if (in_thread_list (pid))
+      kill (pid, SIGSTOP);
+    else if (ptrace (PT_ATTACH, pid, (PTRACE_ARG3_TYPE) 0, 0) == 0)
+      {
+       if (!linuxthreads_attach_pending)
+         printf_unfiltered ("[New %s]\n", target_pid_to_str (pid));
+       add_thread (pid);
+      }
+}
+
+/* Wait for a thread */
```

```
+static void
+wait_thread (pid)
+    int pid;
+{
+  int status;
+  int rpid;
+
+  if (pid != inferior_pid && in_thread_list (pid))
+    {
+      for (;;)
+       {
+         /* Get first pid status.  */
+         rpid = waitpid(pid, &status, __WCLONE);
+         if (rpid > 0)
+           break;
+         if (errno == EINTR)
+           continue;
+
+         /* manager has died or pid is initial thread.  */
+         rpid = waitpid(pid, &status, 0);
+         if (rpid > 0)
+           break;
+         if (errno != EINTR && linuxthreads_thread_alive (pid))
+           perror_with_name ("waitpid");
+
+         /* the thread is dead.  */
+         return;
+       }
+      if (!WIFSTOPPED(status) || WSTOPSIG(status) != SIGSTOP)
+       {
+         linuxthreads_wait_pid[++linuxthreads_wait_last] = pid;
+         linuxthreads_wait_status[linuxthreads_wait_last] = status;
+       }
+    }
+}
+
+/* Walk through the linuxthreads handles in order to detect all
+   threads and stop them */
+static void
+update_stop_threads (test_pid)
+    int test_pid;
+{
+  struct cleanup *old_chain = NULL;
+
+  if (linuxthreads_manager_pid == 0)
+    {
+      if (linuxthreads_manager)
+       {
+         if (test_pid > 0 && test_pid != inferior_pid)
+           {
+             old_chain = save_inferior_pid ();
+             inferior_pid = test_pid;
+           }
+         read_memory (linuxthreads_manager,
+                      (char *)&linuxthreads_manager_pid, sizeof (pid_t));
+       }
+      if (linuxthreads_initial)
+       {
+         if (test_pid > 0 && test_pid != inferior_pid)
+           {
```

```
+             old_chain = save_inferior_pid ();
+             inferior_pid = test_pid;
+           }
+         read_memory(linuxthreads_initial,
+                     (char *)&linuxthreads_initial_pid, sizeof (pid_t));
+       }
+    }
+
+  if (linuxthreads_manager_pid != 0)
+    {
+      if (old_chain == NULL && test_pid > 0 &&
+         test_pid != inferior_pid && linuxthreads_thread_alive (test_pid))
+       {
+         old_chain = save_inferior_pid ();
+         inferior_pid = test_pid;
+       }
+
+      if (linuxthreads_thread_alive (inferior_pid))
+       {
+         if (test_pid > 0)
+           {
+             if (test_pid != linuxthreads_manager_pid
+                 && !linuxthreads_pending_status (linuxthreads_manager_pid))
+               {
+                 stop_thread (linuxthreads_manager_pid);
+                 wait_thread (linuxthreads_manager_pid);
+               }
+             if (!in_thread_list (test_pid))
+               {
+                 if (!linuxthreads_attach_pending)
+                   printf_unfiltered ("[New %s]\n",
+                                      target_pid_to_str (test_pid));
+                 add_thread (test_pid);
+               }
+           }
+         iterate_active_threads (stop_thread, 0);
+         iterate_active_threads (wait_thread, 0);
+       }
+    }
+
+  if (old_chain != NULL)
+    do_cleanups (old_chain);
+}
+
+/* Internal linuxthreads signal management */
+
+static void
+linuxthreads_signal_update (on)
+    int on;
+{
+  int sig_restart = target_signal_from_host(linuxthreads_sig_restart);
+  int sig_cancel = target_signal_from_host(linuxthreads_sig_cancel);
+
+  if (on)
+    {
+      linuxthreads_sig_restart_stop = signal_stop_update(sig_restart, 0);
+      linuxthreads_sig_restart_print = signal_print_update(sig_restart, 0);
+      if (linuxthreads_sig_restart_stop != 1 ||
+         linuxthreads_sig_restart_print != 1)
+       fprintf_unfiltered (gdb_stderr,
```

```
+                           "Linux thread target has modified %s handling\n",
+                           target_signal_to_string(sig_restart));
+
+      linuxthreads_sig_cancel_stop = signal_stop_update(sig_cancel, 0);
+      linuxthreads_sig_cancel_print = signal_print_update(sig_cancel, 0);
+      if (linuxthreads_sig_cancel_stop != 1 ||
+         linuxthreads_sig_cancel_print != 1)
+       fprintf_unfiltered (gdb_stderr,
+                           "Linux thread target has modified %s handling\n",
+                           target_signal_to_string(sig_cancel));
+    }
+  else
+    {
+      signal_stop_update(sig_restart, linuxthreads_sig_restart_stop);
+      signal_print_update(sig_restart, linuxthreads_sig_restart_print);
+      if (linuxthreads_sig_restart_stop != 1 ||
+         linuxthreads_sig_restart_print != 1)
+       fprintf_unfiltered (gdb_stderr,
+                           "Linux thread target has restored %s handling\n",
+                           target_signal_to_string(sig_restart));
+
+      signal_stop_update(sig_cancel, linuxthreads_sig_cancel_stop);
+      signal_print_update(sig_cancel, linuxthreads_sig_cancel_print);
+      if (linuxthreads_sig_cancel_stop != 1 ||
+         linuxthreads_sig_cancel_print != 1)
+       fprintf_unfiltered (gdb_stderr,
+                           "Linux thread target has restored %s handling\n",
+                           target_signal_to_string(sig_cancel));
+    }
+}
+
+/* This routine is called whenever a new symbol table is read in, or when all
+   symbol tables are removed.  libpthread can only be initialized when it
+   finds the right variables in libpthread.so.  Since it's a shared library,
+   those variables don't show up until the library gets mapped and the symbol
+   table is read in.  */
+
+void
+linuxthreads_new_objfile (objfile)
+    struct objfile *objfile;
+{
+  struct minimal_symbol *ms;
+  struct sigaction sact;
+
+  if (!objfile || linuxthreads_max)
+    return;
+
+  if ((ms = lookup_minimal_symbol ("__pthread_threads_debug",
+                                   NULL, objfile)) == NULL)
+    {
+      /* The debugging-aware libpthreads is not present in this objfile */
+      return;
+    }
+  linuxthreads_debug = SYMBOL_VALUE_ADDRESS (ms);
+
+  /* Read internal structures configuration */
+  if ((ms = lookup_minimal_symbol ("__pthread_sizeof_handle",
+                                   NULL, objfile)) == NULL
+      || target_read_memory (SYMBOL_VALUE_ADDRESS (ms),
+                             (char *)&linuxthreads_sizeof_handle,
```

```
+                            sizeof (linuxthreads_sizeof_handle)) != 0)
+    {
+      fprintf_unfiltered (gdb_stderr,
+                          "Unable to find linuxthreads symbol \"%s\"\n",
+                          "__pthread_sizeof_handle");
+      return;
+    }
+
+  if ((ms = lookup_minimal_symbol ("__pthread_offsetof_descr",
+                                   NULL, objfile)) == NULL
+      || target_read_memory (SYMBOL_VALUE_ADDRESS (ms),
+                             (char *)&linuxthreads_offset_descr,
+                             sizeof (linuxthreads_offset_descr)) != 0)
+    {
+      fprintf_unfiltered (gdb_stderr,
+                          "Unable to find linuxthreads symbol \"%s\"\n",
+                          "__pthread_offsetof_descr");
+      return;
+    }
+
+  if ((ms = lookup_minimal_symbol ("__pthread_offsetof_pid",
+                                   NULL, objfile)) == NULL
+      || target_read_memory (SYMBOL_VALUE_ADDRESS (ms),
+                             (char *)&linuxthreads_offset_pid,
+                             sizeof (linuxthreads_offset_pid)) != 0)
+    {
+      fprintf_unfiltered (gdb_stderr,
+                          "Unable to find linuxthreads symbol \"%s\"\n",
+                          "__pthread_offsetof_pid");
+      return;
+    }
+
+  if ((ms = lookup_minimal_symbol ("__pthread_sig_restart",
+                                   NULL, objfile)) == NULL
+      || target_read_memory (SYMBOL_VALUE_ADDRESS (ms),
+                             (char *)&linuxthreads_sig_restart,
+                             sizeof (linuxthreads_sig_restart)) != 0)
+    {
+      fprintf_unfiltered (gdb_stderr,
+                          "Unable to find linuxthreads symbol \"%s\"\n",
+                          "__pthread_sig_restart");
+      return;
+    }
+
+  if ((ms = lookup_minimal_symbol ("__pthread_sig_cancel",
+                                   NULL, objfile)) == NULL
+      || target_read_memory (SYMBOL_VALUE_ADDRESS (ms),
+                             (char *)&linuxthreads_sig_cancel,
+                             sizeof (linuxthreads_sig_cancel)) != 0)
+    {
+      fprintf_unfiltered (gdb_stderr,
+                          "Unable to find linuxthreads symbol \"%s\"\n",
+                          "__pthread_sig_cancel");
+      return;
+    }
+
+  if ((ms = lookup_minimal_symbol ("__pthread_threads_max",
+                                   NULL, objfile)) == NULL
+      || target_read_memory (SYMBOL_VALUE_ADDRESS (ms),
+                             (char *)&linuxthreads_max,
```

```
+                             sizeof (linuxthreads_max)) != 0)
+    {
+      fprintf_unfiltered (gdb_stderr,
+                          "Unable to find linuxthreads symbol \"%s\"\n",
+                          "__pthread_threads_max");
+      return;
+    }
+
+  /* Read adresses of internal structures to access */
+  if ((ms = lookup_minimal_symbol ("__pthread_handles",
+                                   NULL, objfile)) == NULL)
+    {
+      fprintf_unfiltered (gdb_stderr,
+                          "Unable to find linuxthreads symbol \"%s\"\n",
+                          "__pthread_handles");
+      return;
+    }
+  linuxthreads_handles = SYMBOL_VALUE_ADDRESS (ms);
+
+  if ((ms = lookup_minimal_symbol ("__pthread_handles_num",
+                                   NULL, objfile)) == NULL)
+    {
+      fprintf_unfiltered (gdb_stderr,
+                          "Unable to find linuxthreads symbol \"%s\"\n",
+                          "__pthread_handles_num");
+      return;
+    }
+  linuxthreads_num = SYMBOL_VALUE_ADDRESS (ms);
+
+  if ((ms = lookup_minimal_symbol ("__pthread_manager_thread",
+                                   NULL, objfile)) == NULL)
+    {
+      fprintf_unfiltered (gdb_stderr,
+                          "Unable to find linuxthreads symbol \"%s\"\n",
+                          "__pthread_manager_thread");
+      return;
+    }
+  linuxthreads_manager = SYMBOL_VALUE_ADDRESS (ms) + linuxthreads_offset_pid;
+
+  if ((ms = lookup_minimal_symbol ("__pthread_initial_thread",
+                                   NULL, objfile)) == NULL)
+    {
+      fprintf_unfiltered (gdb_stderr,
+                          "Unable to find linuxthreads symbol \"%s\"\n",
+                          "__pthread_initial_thread");
+      return;
+    }
+  linuxthreads_initial = SYMBOL_VALUE_ADDRESS (ms) + linuxthreads_offset_pid;
+
+  /* Allocate gdb internal structures */
+  linuxthreads_wait_pid =
+    (int *)xmalloc (sizeof (int) * (linuxthreads_max + 1));
+  linuxthreads_wait_status =
+    (int *)xmalloc (sizeof (int) * (linuxthreads_max + 1));
+  linuxthreads_breakpoint_zombie = (struct linuxthreads_breakpoint *)
+    xmalloc (sizeof (struct linuxthreads_breakpoint) * (linuxthreads_max + 1));
+
+  /* handle linuxthread exit */
+  sact.sa_handler = sigchld_handler;
+  sigemptyset(&sact.sa_mask);
```

```
+  sact.sa_flags = 0;
+  sigaction(linuxthreads_sig_restart, &sact, NULL);
+
+  if (inferior_pid && !linuxthreads_attach_pending)
+    {
+      int on = 1;
+      target_write_memory (linuxthreads_debug, (char *)&on, sizeof (on));
+      linuxthreads_attach_pending = 1;
+      linuxthreads_signal_update (1);
+      update_stop_threads (inferior_pid);
+      linuxthreads_attach_pending = 0;
+    }
+}
+
+/* If we have switched threads from a one that stopped at breakpoint,
+   return 1 otherwise 0.  */
+
+int
+linuxthreads_prepare_to_proceed (step)
+    int step;
+{
+  if (!linuxthreads_max
+      || !linuxthreads_manager_pid
+      || !linuxthreads_breakpoint_pid
+      || !breakpoint_here_p (read_pc_pid (linuxthreads_breakpoint_pid)))
+    return 0;
+
+  if (step)
+    {
+      /* Mark the current inferior as single stepping process.  */
+      linuxthreads_step_pid = inferior_pid;
+    }
+
+  linuxthreads_inferior_pid = linuxthreads_breakpoint_pid;
+  return linuxthreads_breakpoint_pid;
+}
+
+/* Convert a pid to printable form. */
+
+char *
+linuxthreads_pid_to_str (pid)
+    int pid;
+{
+  static char buf[100];
+
+  sprintf (buf, "%s %d", linuxthreads_max ? "Thread" : "Pid", pid);
+
+  return buf;
+}
+
+/* Attach to process PID, then initialize for debugging it
+   and wait for the trace-trap that results from attaching.  */
+
+static void
+linuxthreads_attach (args, from_tty)
+    char *args;
+    int from_tty;
+{
+  push_target (&linuxthreads_ops);
+  linuxthreads_breakpoints_inserted = 1;
```

```
+  linuxthreads_breakpoint_last = -1;
+  linuxthreads_wait_last = -1;
+  linuxthreads_exit_status = __W_STOPCODE(0);
+
+  child_ops.to_attach (args, from_tty);
+
+  if (linuxthreads_max)
+    linuxthreads_attach_pending = 1;
+}
+
+/* Take a program previously attached to and detaches it.
+   The program resumes execution and will no longer stop
+   on signals, etc.  We'd better not have left any breakpoints
+   in the program or it'll die when it hits one.  For this
+   to work, it may be necessary for the process to have been
+   previously attached.  It *might* work if the program was
+   started via the normal ptrace (PTRACE_TRACEME).  */
+
+static void
+linuxthreads_detach (args, from_tty)
+    char *args;
+    int from_tty;
+{
+  if (linuxthreads_max)
+    {
+      int i;
+      int pid;
+      int off = 0;
+      target_write_memory (linuxthreads_debug, (char *)&off, sizeof (off));
+
+      /* Walk through linuxthreads array in order to detach known threads.  */
+      if (linuxthreads_manager_pid != 0)
+       {
+         /* Get rid of all positive zombie breakpoints.  */
+         for (i = 0; i <= linuxthreads_breakpoint_last; i++)
+           {
+             if (linuxthreads_breakpoint_zombie[i].step)
+               continue;
+
+             pid = linuxthreads_breakpoint_zombie[i].pid;
+             if (!linuxthreads_thread_alive (pid))
+               continue;
+
+             if (linuxthreads_breakpoint_zombie[i].pc != read_pc_pid (pid))
+               continue;
+
+             /* Continue in STEP mode until the thread pc has moved or
+                until SIGTRAP is found on the same PC.  */
+             if (linuxthreads_find_trap (pid, 0)
+                 && linuxthreads_breakpoint_zombie[i].pc == read_pc_pid (pid))
+               write_pc_pid (linuxthreads_breakpoint_zombie[i].pc
+                             - DECR_PC_AFTER_BREAK, pid);
+           }
+
+         /* Detach thread after thread.  */
+         inferior_pid = linuxthreads_manager_pid;
+         iterate_active_threads (detach_thread, 1);
+
+         /* Remove pending SIGTRAP and SIGSTOP */
+         linuxthreads_find_trap (inferior_pid, 1);
```

```
+
+         linuxthreads_wait_last = -1;
+         linuxthreads_exit_status = __W_STOPCODE(0);
+       }
+
+      linuxthreads_inferior_pid = 0;
+      linuxthreads_breakpoint_pid = 0;
+      linuxthreads_step_pid = 0;
+      linuxthreads_step_signo = TARGET_SIGNAL_0;
+      linuxthreads_manager_pid = 0;
+      linuxthreads_initial_pid = 0;
+      linuxthreads_attach_pending = 0;
+      linuxthreads_signal_update (0);
+      init_thread_list ();           /* Destroy thread info */
+    }
+
+  child_ops.to_detach (args, from_tty);
+
+  unpush_target (&linuxthreads_ops);
+}
+
+/* Resume execution of process PID.  If STEP is nozero, then
+   just single step it.  If SIGNAL is nonzero, restart it with that
+   signal activated.  */
+
+static void
+linuxthreads_resume (pid, step, signo)
+    int pid;
+    int step;
+    enum target_signal signo;
+{
+  if (!linuxthreads_max || stop_soon_quietly || linuxthreads_manager_pid == 0)
+    child_ops.to_resume (pid, step, signo);
+  else
+    {
+      int rpid;
+      if (linuxthreads_inferior_pid)
+       {
+         /* Prepare resume of the last thread that hit a breakpoint */
+         linuxthreads_breakpoints_inserted = 0;
+         rpid = linuxthreads_inferior_pid;
+         linuxthreads_step_signo = signo;
+       }
+      else
+        {
+         struct cleanup *old_chain = NULL;
+         int i;
+
+         if (pid < 0)
+           {
+             linuxthreads_step_pid = step ? inferior_pid : 0;
+             linuxthreads_step_signo = signo;
+             rpid = inferior_pid;
+           }
+         else
+           rpid = pid;
+
+         if (pid < 0 || !step)
+           {
+             linuxthreads_breakpoints_inserted = 1;
```

```
+
+            /* Walk through linuxthreads array in order to resume threads */
+            if (pid >= 0 && inferior_pid != pid)
+              {
+                old_chain = save_inferior_pid ();
+                inferior_pid = pid;
+              }
+
+            iterate_active_threads (resume_thread, 0);
+            if (linuxthreads_manager_pid != inferior_pid
+                && !linuxthreads_pending_status (linuxthreads_manager_pid))
+              resume_thread (linuxthreads_manager_pid);
+          }
+        else
+          linuxthreads_breakpoints_inserted = 0;
+
+        /* Deal with zombie breakpoint */
+        for (i = 0; i <= linuxthreads_breakpoint_last; i++)
+          if (linuxthreads_breakpoint_zombie[i].pid == rpid)
+            {
+              if (linuxthreads_breakpoint_zombie[i].pc != read_pc_pid (rpid))
+                {
+                  /* The current pc is out of zombie breakpoint.  */
+                  REMOVE_BREAKPOINT_ZOMBIE(i);
+                }
+              break;
+            }
+
+        if (old_chain != NULL)
+          do_cleanups (old_chain);
+       }
+
+      /* Resume initial thread. */
+      if (!linuxthreads_pending_status (rpid))
+       child_ops.to_resume (rpid, step, signo);
+    }
+}
+
+/* Wait for any threads to stop.  We may have to convert PID from a thread id
+   to a LWP id, and vice versa on the way out.  */
+
+static int
+linuxthreads_wait (pid, ourstatus)
+    int pid;
+    struct target_waitstatus *ourstatus;
+{
+  int status;
+  int rpid;
+  int i;
+  int last;
+  int *wstatus;
+
+  for (;;)
+    {
+      if (!linuxthreads_max)
+        rpid = 0;
+      else if (!linuxthreads_breakpoints_inserted)
+       {
+        if (linuxthreads_inferior_pid)
+          pid = linuxthreads_inferior_pid;
```

```
+         else if (pid < 0)
+           pid = inferior_pid;
+         last = rpid = 0;
+         wstatus = alloca (LINUXTHREAD_NSIG * sizeof (int));
+       }
+      else if (pid < 0 && linuxthreads_wait_last >= 0)
+       {
+         status = linuxthreads_wait_status[linuxthreads_wait_last];
+         rpid = linuxthreads_wait_pid[linuxthreads_wait_last--];
+       }
+      else if (pid > 0 && linuxthreads_pending_status (pid))
+       {
+         for (i = linuxthreads_wait_last; i >= 0; i--)
+           if (linuxthreads_wait_pid[i] == pid)
+             break;
+         if (i < 0)
+           rpid = 0;
+         else
+           {
+             status = linuxthreads_wait_status[i];
+             rpid = pid;
+             if (i < linuxthreads_wait_last)
+               {
+                 linuxthreads_wait_status[i] =
+                   linuxthreads_wait_status[linuxthreads_wait_last];
+                 linuxthreads_wait_pid[i] =
+                   linuxthreads_wait_pid[linuxthreads_wait_last];
+               }
+             linuxthreads_wait_last--;
+           }
+       }
+      else
+         rpid = 0;
+
+      if (rpid == 0)
+       {
+         int save_errno;
+         sigset_t omask;
+
+         set_sigint_trap();    /* Causes SIGINT to be passed on to the
+                                  attached process. */
+         set_sigio_trap ();
+
+         sigprocmask(SIG_BLOCK, &linuxthreads_wait_mask, &omask);
+         for (;;)
+           {
+             rpid = waitpid (pid, &status, __WCLONE | WNOHANG);
+             if (rpid > 0)
+               break;
+             if (rpid == 0)
+               save_errno = 0;
+             else if (errno != EINTR)
+               save_errno = errno;
+             else
+               continue;
+
+             rpid = waitpid (pid, &status, WNOHANG);
+             if (rpid > 0)
+               break;
+             if (rpid < 0)
```

```
+              if (errno == EINTR)
+                continue;
+              else if (save_errno != 0)
+                break;
+
+            sigsuspend(&omask);
+          }
+        sigprocmask(SIG_SETMASK, &omask, NULL);
+
+        save_errno = errno;
+        clear_sigio_trap ();
+
+        clear_sigint_trap();
+
+        if (rpid == -1)
+          {
+            if (WIFEXITED(linuxthreads_exit_status))
+              {
+                store_waitstatus (ourstatus, linuxthreads_exit_status);
+                return inferior_pid;
+              }
+            else
+              {
+                fprintf_unfiltered
+                    (gdb_stderr, "Child process unexpectedly missing: %s.\n",
+                     safe_strerror (save_errno));
+                /* Claim it exited with unknown signal.  */
+                ourstatus->kind = TARGET_WAITKIND_SIGNALLED;
+                ourstatus->value.sig = TARGET_SIGNAL_UNKNOWN;
+                return -1;
+              }
+          }
+
+        /* Signals arrive in any order.  So get all signals until SIGTRAP
+           and resend previous ones to be held after.  */
+        if (linuxthreads_max
+            && !linuxthreads_breakpoints_inserted
+            && WIFSTOPPED(status))
+          if (WSTOPSIG(status) == SIGTRAP)
+            {
+              while (--last >= 0)
+                kill (rpid, WSTOPSIG(wstatus[last]));
+
+              /* insert negative zombie breakpoint */
+              for (i = 0; i <= linuxthreads_breakpoint_last; i++)
+                if (linuxthreads_breakpoint_zombie[i].pid == rpid)
+                    break;
+              if (i > linuxthreads_breakpoint_last)
+                {
+                  linuxthreads_breakpoint_zombie[i].pid = rpid;
+                  linuxthreads_breakpoint_last++;
+                }
+              linuxthreads_breakpoint_zombie[i].pc = read_pc_pid (rpid);
+              linuxthreads_breakpoint_zombie[i].step = 1;
+            }
+          else
+            {
+              if (WSTOPSIG(status) != SIGSTOP)
+                {
+                  for (i = 0; i < last; i++)
```

```
+                    if (wstatus[i] == status)
+                      break;
+                  if (i >= last)
+                    wstatus[last++] = status;
+                }
+              child_resume (rpid, 1, TARGET_SIGNAL_0);
+              continue;
+            }
+          if (linuxthreads_inferior_pid)
+            linuxthreads_inferior_pid = 0;
+        }
+
+      if (linuxthreads_max && !stop_soon_quietly)
+        {
+          if (linuxthreads_max
+              && WIFSTOPPED(status)
+              && WSTOPSIG(status) == SIGSTOP)
+            {
+              /* Skip SIGSTOP signals.  */
+              if (!linuxthreads_pending_status (rpid))
+                if (linuxthreads_step_pid == rpid)
+                  child_resume (rpid, 1, linuxthreads_step_signo);
+                else
+                  child_resume (rpid, 0, TARGET_SIGNAL_0);
+              continue;
+            }
+
+          /* Do no report exit status of cloned threads.  */
+          if (WIFEXITED(status))
+            {
+              if (rpid == linuxthreads_initial_pid)
+                linuxthreads_exit_status = status;
+
+              /* Remove any zombie breakpoint.  */
+              for (i = 0; i <= linuxthreads_breakpoint_last; i++)
+                if (linuxthreads_breakpoint_zombie[i].pid == rpid)
+                  {
+                    REMOVE_BREAKPOINT_ZOMBIE(i);
+                    break;
+                  }
+              if (pid > 0)
+                pid = -1;
+              continue;
+            }
+
+          /* Deal with zombie breakpoint */
+          for (i = 0; i <= linuxthreads_breakpoint_last; i++)
+            if (linuxthreads_breakpoint_zombie[i].pid == rpid)
+              break;
+
+          if (i <= linuxthreads_breakpoint_last)
+            {
+              /* There is a potential zombie breakpoint */
+              if (WIFEXITED(status)
+                  || linuxthreads_breakpoint_zombie[i].pc != read_pc_pid (rpid))
+                {
+                  /* The current pc is out of zombie breakpoint.  */
+                  REMOVE_BREAKPOINT_ZOMBIE(i);
+                }
+              else if (!linuxthreads_breakpoint_zombie[i].step
```

```
+                      && WIFSTOPPED(status) && WSTOPSIG(status) == SIGTRAP)
+                {
+                  /* This is a real one ==> decrement PC and restart.  */
+                  write_pc_pid (linuxthreads_breakpoint_zombie[i].pc
+                                - DECR_PC_AFTER_BREAK, rpid);
+                  if (linuxthreads_step_pid == rpid)
+                    child_resume (rpid, 1, linuxthreads_step_signo);
+                  else
+                    child_resume (rpid, 0, TARGET_SIGNAL_0);
+                  continue;
+                }
+            }
+
+          /* Walk through linuxthreads array in order to stop them */
+          if (linuxthreads_breakpoints_inserted)
+            update_stop_threads (rpid);
+
+        }
+      else if (rpid != inferior_pid)
+        continue;
+
+      store_waitstatus (ourstatus, status);
+
+      if (linuxthreads_attach_pending && !stop_soon_quietly)
+        {
+          int on = 1;
+          target_write_memory (linuxthreads_debug, (char *)&on, sizeof (on));
+          update_stop_threads (rpid);
+          linuxthreads_signal_update (1);
+          linuxthreads_attach_pending = 0;
+        }
+
+      if (linuxthreads_breakpoints_inserted
+          && WIFSTOPPED(status)
+          && WSTOPSIG(status) == SIGTRAP)
+        linuxthreads_breakpoint_pid = rpid;
+      else if (linuxthreads_breakpoint_pid)
+        linuxthreads_breakpoint_pid = 0;
+
+      return rpid;
+    }
+}
+
+/* Fork an inferior process, and start debugging it with ptrace.  */
+
+static void
+linuxthreads_create_inferior (exec_file, allargs, env)
+    char *exec_file;
+    char *allargs;
+    char **env;
+{
+  push_target (&linuxthreads_ops);
+  linuxthreads_breakpoints_inserted = 1;
+  linuxthreads_breakpoint_last = -1;
+  linuxthreads_wait_last = -1;
+  linuxthreads_exit_status = __W_STOPCODE(0);
+
+  if (linuxthreads_max)
+    linuxthreads_attach_pending = 1;
+
```

```
+  child_ops.to_create_inferior (exec_file, allargs, env);
+}
+
+/* Clean up after the inferior dies.  */
+
+static void
+linuxthreads_mourn_inferior ()
+{
+  if (linuxthreads_max)
+    {
+      int off = 0;
+      target_write_memory (linuxthreads_debug, (char *)&off, sizeof (off));
+
+      linuxthreads_inferior_pid = 0;
+      linuxthreads_breakpoint_pid = 0;
+      linuxthreads_step_pid = 0;
+      linuxthreads_step_signo = TARGET_SIGNAL_0;
+      linuxthreads_manager_pid = 0;
+      linuxthreads_initial_pid = 0;
+      linuxthreads_attach_pending = 0;
+      init_thread_list();           /* Destroy thread info */
+      linuxthreads_signal_update (0);
+    }
+
+  child_ops.to_mourn_inferior ();
+
+  unpush_target (&linuxthreads_ops);
+}
+
+/* Kill the inferior process */
+
+static void
+linuxthreads_kill ()
+{
+  int rpid;
+  int status;
+
+  if (inferior_pid == 0)
+    return;
+
+  if (linuxthreads_max && linuxthreads_manager_pid != 0)
+    {
+      /* Remove all threads status.  */
+      inferior_pid = linuxthreads_manager_pid;
+      iterate_active_threads (kill_thread, 1);
+    }
+
+  kill_thread (inferior_pid);
+
+  if (linuxthreads_max && linuxthreads_manager_pid != 0)
+    {
+      /* Wait for thread to complete */
+      while ((rpid = waitpid (-1, &status, __WCLONE)) > 0)
+       if (!WIFEXITED(status))
+         kill_thread (rpid);
+
+      while ((rpid = waitpid (-1, &status, 0)) > 0)
+       if (!WIFEXITED(status))
+         kill_thread (rpid);
+    }
```

```
+  else
+    while ((rpid = waitpid (inferior_pid, &status, 0)) > 0)
+      if (!WIFEXITED(status))
+       ptrace (PT_KILL, inferior_pid, (PTRACE_ARG3_TYPE) 0, 0);
+
+  linuxthreads_mourn_inferior ();
+}
+
+/* Insert a breakpoint */
+
+static int
+linuxthreads_insert_breakpoint (addr, contents_cache)
+    CORE_ADDR addr;
+    char *contents_cache;
+{
+  if (linuxthreads_max && linuxthreads_manager_pid != 0)
+    {
+      linuxthreads_breakpoint_addr = addr;
+      iterate_active_threads (insert_breakpoint, 1);
+      insert_breakpoint (linuxthreads_manager_pid);
+    }
+
+  return child_ops.to_insert_breakpoint (addr, contents_cache);
+}
+
+/* Remove a breakpoint */
+
+static int
+linuxthreads_remove_breakpoint (addr, contents_cache)
+    CORE_ADDR addr;
+    char *contents_cache;
+{
+  if (linuxthreads_max && linuxthreads_manager_pid != 0)
+    {
+      linuxthreads_breakpoint_addr = addr;
+      iterate_active_threads (remove_breakpoint, 1);
+      remove_breakpoint (linuxthreads_manager_pid);
+    }
+
+  return child_ops.to_remove_breakpoint (addr, contents_cache);
+}
+
+/* Mark our target-struct as eligible for stray "run" and "attach" commands.  */
+
+static int
+linuxthreads_can_run ()
+{
+  return child_suppress_run;
+}
+
```

```
+struct target_ops linuxthreads_ops = {
+  "linuxthreads",                       /* to_shortname */
+  "LINUX threads and pthread.",         /* to_longname */
+  "LINUX threads and pthread support.",         /* to_doc */
+  0,                                    /* to_open */
+  0,                                    /* to_close */
+  linuxthreads_attach,                  /* to_attach */
+  linuxthreads_detach,                          /* to_detach */
+  linuxthreads_resume,                  /* to_resume */
+  linuxthreads_wait,                    /* to_wait */
+  0,                                    /* to_fetch_registers */
+  0,                                    /* to_store_registers */
+  0,                                    /* to_prepare_to_store */
+  0,                                    /* to_xfer_memory */
+  0,                                    /* to_files_info */
+  linuxthreads_insert_breakpoint,       /* to_insert_breakpoint */
+  linuxthreads_remove_breakpoint,       /* to_remove_breakpoint */
+  0,                                    /* to_terminal_init */
+  0,                                    /* to_terminal_inferior */
+  0,                                    /* to_terminal_ours_for_output */
+  0,                                    /* to_terminal_ours */
+  0,                                    /* to_terminal_info */
+  linuxthreads_kill,                    /* to_kill */
+  0,                                    /* to_load */
+  0,                                    /* to_lookup_symbol */
+  linuxthreads_create_inferior,                 /* to_create_inferior */
+  linuxthreads_mourn_inferior,          /* to_mourn_inferior */
+  linuxthreads_can_run,                         /* to_can_run */
+  0,                                    /* to_notice_signals */
+  linuxthreads_thread_alive,            /* to_thread_alive */
+  0,                                    /* to_stop */
+  thread_stratum,                       /* to_stratum */
+  0,                                    /* to_next */
+  0,                                    /* to_has_all_memory */
+  0,                                    /* to_has_memory */
+  1,                                    /* to_has_stack */
+  1,                                    /* to_has_registers */
+  1,                                    /* to_has_execution */
+  0,                                    /* sections */
+  0,                                    /* sections_end */
+  OPS_MAGIC                             /* to_magic */
+};
+
+void
+_initialize_linuxthreads ()
+{
+  struct sigaction sact;
+
+  add_target (&linuxthreads_ops);
+  child_suppress_run = 1;
+
+  /* Attach SIGCHLD handler */
+  sact.sa_handler = sigchld_handler;
+  sigemptyset(&sact.sa_mask);
+  sact.sa_flags = 0;
+  sigaction(SIGCHLD, &sact, NULL);
+
+  /* initialize SIGCHLD mask */
+  sigemptyset(&linuxthreads_wait_mask);
+  sigaddset(&linuxthreads_wait_mask, SIGCHLD);
```

```
+}
diff -urNp gdb-4.17-ORIG/gdb/m3-nat.c gdb-4.17/gdb/m3-nat.c
--- gdb-4.17-ORIG/gdb/m3-nat.c  Mon May  6 14:27:04 1996
+++ gdb-4.17/gdb/m3-nat.c       Fri Nov 13 22:31:40 1998
@@ -1,7 +1,7 @@
 /* Interface GDB to Mach 3.0 operating systems.
    (Most) Mach 3.0 related routines live in this file.

-   Copyright (C) 1992, 1996 Free Software Foundation, Inc.
+   Copyright (C) 1992, 1996, 1998 Free Software Foundation, Inc.

 This file is part of GDB.

@@ -1576,26 +1576,23 @@ mach_thread_output_id (mid)
   *
   * If we have switched threads and stopped at breakpoint return 1 otherwise 0.
   *
- *  if SELECT_IT is nonzero, reselect the thread that was active when
- *  we stopped at a breakpoint.
- *
+ * If we are single stepping, don't do anything since in the current
+ * implementation single stepping another thread after a breakpoint and
+ * then continuing will cause the original breakpoint to be hit again,
+ * but you can always continue, so it's not a big deal.
   */

-mach3_prepare_to_proceed (select_it)
-     int select_it;
+mach3_prepare_to_proceed (step)
+     int step;
 {
-  if (stop_thread &&
+  if (!step &&
       stop_thread &&
       stop_thread != current_thread &&
       stop_exception == EXC_BREAKPOINT)
     {
-      int mid;
-
-      if (! select_it)
-       return 1;
-
-      mid = switch_to_thread (stop_thread);
-
-      return 1;
+      switch_to_thread (stop_thread);
+      if (breakpoint_here_p (read_pc ()))
+       return inferior_pid;
     }

   return 0;
diff -urNp gdb-4.17-ORIG/gdb/target.h gdb-4.17/gdb/target.h
--- gdb-4.17-ORIG/gdb/target.h  Wed May  7 18:00:40 1997
+++ gdb-4.17/gdb/target.h       Fri Nov 13 22:31:40 1998
@@ -1,5 +1,5 @@
 /* Interface between GDB and target environments, including files and processes
-   Copyright 1990, 1991, 1992, 1993, 1994 Free Software Foundation, Inc.
+   Copyright 1990, 1991, 1992, 1993, 1994, 1998 Free Software Foundation, Inc.
    Contributed by Cygnus Support.  Written by John Gilmore.
```

```
 This file is part of GDB.
@@ -47,7 +47,8 @@ enum strata {
        file_stratum,           /* Executable files, etc */
        core_stratum,           /* Core dump files */
        download_stratum,       /* Downloading of remote targets */
-       process_stratum         /* Executing processes */
+       process_stratum,        /* Executing processes */
+       thread_stratum          /* Executing threads */
 };

 /* Stuff for target_wait.  */
```

Mpeg*TV* - Home Page

mtv	Pocket*TV*	Mpeg*TV* SDK
MPEG Movie and VCD Player for Linux and Unix	MPEG Movie Player for Pocket PC and Handheld PC	Toolkit includes MPEG Video+Audio real-time library and sample source code for Linux and WinCE

April 13, 2001:

Mpeg*TV* has released **Pocket*TV* for Handheld PC** (e.g. Jornada 720)

Supports Handheld PC Pro and Handheld PC 2000 devices.
Processors supported: **ARM**, **MIPS**, **SH3**, **SH4** and **x86/Transmeta**.

About Mpeg*TV*

Mpeg*TV* Products

Xaudio and other MP3 Products

REGISTER mtv Now!

REGISTER Pocket*TV* Now!

What is MPEG ?

Mpeg*TV* Home | Pocket*TV* | About Mpeg*TV* | Products | FAQs | Support

Feedback

ATLANTEL Bordeaux

Vidéo live !

Ce serveur présente une vidéo temps-réel à partir d'une caméra placée rue Condillac, en plein centre de Bordeaux, dans les locaux de la société Atlantel Multimédia. Ce n'est pas une vidéo lue à partir d'un fichier. Si vous regardez en haut à gauche de l'image, vous pourrez voir les secondes défiler...

Ce service utilise la bibliothèque LinuxThreads sur RedHat LINUX

Si votre navigateur n'est pas compatible JAVA, la visualisation de cette page nécessite l'installation du *plug-in VCR/VCO* disponible gratuitement en cliquant ici.

Pour tout renseignement, contacter Pierre Ficheux ou Jean-Sébastien Suze.

This Web site presents a real time video from a camera located rue Condillac in the main center of Bordeaux (France). The server is hosted by the Atlantel Multimédia company. This video is not extracted from a recorded file. If you take a look at the clock at the right up part of the video, you should see the local time running...

This service uses the LinuxThreads library running on RedHat LINUX

If you browser is not JAVA compatible, this page needs the *VCR/VCO plugin*, click here to download it for free.

For information about this technology, contact Pierre Ficheux or Jean-Sébastien Suze.

Java Linux

Kaffe.org

New Release Building Developing Porting Bug Datab

Welcome to Kaffe

JULY 19, 2000 - Transvirtual is proud to announce the unification of the Custom and Desktop Editions of Kaffe, the first truly open Java implementation for embedded devices. more>>

What is Kaffe?

Kaffe is a cleanroom, open source implementation of a Java virtual machine and class libraries. It is also a fun project that was started by Tim Wilkinson and was made successful by the contributions of numerous people from all over the world. But Kaffe is not finished yet! You can help by developing new and missing functionality, porting Kaffe to new platforms, and testing your Java applications under Kaffe.

Kaffe mostly complies with JDK 1.1, except for a few missing parts. Parts of it are already JDK 1.2 (Java 2) compatible.

These pages are dedicated to providing resources to the Kaffe community. Please, read this important disclaimer first.

Transvirtual's Kaffe main page

Where can I get Kaffe?

Current release: 1.0.6

How do I build and use Kaffe?

Troubleshooting FAQ.

Frequently asked questions about Kaffe licensing.

This covers the relationship between Kaffe and Transvirtual Technologies and should answer any questions you have about Kaffe's copyright.

Reporting Bugs in Kaffe:

Kaffe Users - Please use this site if you use Kaffe and would like to log a bug or feature request.
Kaffe Developers - This is the bug tracking site reserved for Kaffe Developers.

Contributing to Kaffe:

Develop new Kaffe components.
Port Kaffe to new platforms.

Mailing lists:

How to subscribe to Kaffe's mailing lists;
How to access the mailing list archive.

Related projects

The Kaffe Core Team

Florist

Florist is the FSU implementaton of IEEE Standard 1003.5b-1996, the POSIX Ada binding, including real-time extensions. This software provides access to the UNIX operating system services for application programs written in the Ada programming language. It is designed to be self-configuring for a POSIX-compliant system. The complete Florist sources are available via public ftp.

Professor T.P. Baker serves as Chair of the POSIX 1003.5 Language Bindings Working Group, under the Portable Applications Standards Committee (PASC), and as Project Editor responsible for seeing the POSIX Ada bindings through the ISO/IEC standardization process. The FSU POSIX/Ada Real-Time (PART) project provides editorial and technical support for the POSIX Ada bindings standards. This includes development of the FSU Threads Library.

For Technical reports and preprints related to the PART project see our public ftp site.

Commercial maintenance of Florist is available from AdaCore Technologies, Inc.. We cooperate with them to coordinate updates and fixes.

The current Lynx development code is accessible via:
http://lynx.isc.org/current/
It is undergoing field testing for eventual release as v2.9. Report bugs to *lynx-dev@sig.net*.

Lynx2-8-3

This page contains links to the April 23, 2000 *Lynx v2.8.3* release. This supersedes Lynx v2.8.2.

See: Lynx links for a complete list of distribution sites.

The following v2.8.3 files are available for downloading:

- CHANGES - changes made since version 2-8
- INSTALLATION - how to configure, build and install Lynx
- lynx2-8-3.zip - the distribution file
- lynx2-8-3.zip-lst - a list of files in the zip file
- lynx2-8-3.tar.Z - the distribution in compressed tar format
- lynx2-8-3.tar.gz - the distribution in gzip'd tar format
- lynx2-8-3.tar.bz2 - the distribution in bzip2'd tar format
- lynx-2.8.3.tar.gz - the distribution with available international message catalogs
- lynx2-8-3/ - a v2.8.3 breakout
- po/ - message libraries
- New features in this release
- SSL support for this release
- Bug fixes for this release

Get zip/unzip from: ftp://ftp.cdrom.com/pub/infozip

Get gzip from: ftp://ftp.gnu.org/pub/gnu/

Get bzip2 from: http://www.muraroa.demon.co.uk/

Lynx for Win32 is available at http://www.shonai-cit.ac.jp/eci/senshu/

Other versions of **Lynx** for DOS386+ and Win32 can be found on
http://www.fdisk.com/doslynx/lynxport.htm

Here is the *Lynx v2.8.3* online help.

If you prefer downloading by FTP, use the following links instead, or use anonymous FTP to 'lynx.isc.org' and 'cd lynx-2.8.3' to find these files (or use this link to see all the files from an FTP directory listing):

CHANGES - changes made since version 2-8

INSTALLATION - how to configure, build and install Lynx

lynx2-8-3.zip - the distribution file

lynx2-8-3.zip-lst - a list of files in the zip file

lynx2-8-3.tar.Z - the distribution compressed tar format

lynx2-8-3.tar.gz - the distribution in gzip'd tar format

lynx2-8-3.tar.bz2 - the distribution in bzip2'd tar format

lynx-2.8.3.tar.gz - the distribution with available international message catalogs

lynx2-8-3 - a v2.8.3 breakout

po - message libraries

New features in this release

SSL support for this release

Bug fixes for this release

---

http://lynx.isc.org/lynx-2.8.3/index.html
Page maintained by Thomas Dickey (dickey@radix.net)
& James Spath (jspath@bcpl.net).

Last Updated: 2001-01-31 Valid HTML! Re-validate

This directory contains archive copies of the most recent **Lynx** development source code. We are currently working on Lynx 2.8.4.

Mirror sites
For the latest stable release and other help, see http://lynx.browser.org/. For DOS386/Win32 binaries, see below.
The development version also has a breakout subdirectory.

Current version

- (zip): lynx-cur.zip
- (tar.gz): lynx-cur.tgz
- (tar.bz2): lynx-cur.bz2

Current Version in Various Formats

Size	Date	Time	Filename
2760210	Jul 10	10:05	lynx2.8.4pre.4.zip
2555326	Jul 10	10:05	lynx2.8.4pre.4.tar.gz
1938905	Jul 10	10:05	lynx2.8.4pre.4.tar.bz2
3747438	Jul 10	10:05	lynx2.8.4pre.4.tar.Z
2760075	Jul 7	18:30	lynx2.8.4pre.3.zip
2555215	Jul 7	18:30	lynx2.8.4pre.3.tar.gz
1939055	Jul 7	18:30	lynx2.8.4pre.3.tar.bz2
3758627	Jul 7	18:30	lynx2.8.4pre.3.tar.Z
2753704	Jun 10	18:04	lynx2.8.4pre.2.zip
2548503	Jun 10	18:04	lynx2.8.4pre.2.tar.gz
1931783	Jun 10	18:04	lynx2.8.4pre.2.tar.bz2
3736388	Jun 10	18:04	lynx2.8.4pre.2.tar.Z
2753448	Jun 3	13:19	lynx2.8.4pre.1.zip
2548150	Jun 3	13:19	lynx2.8.4pre.1.tar.gz
1932623	Jun 3	13:19	lynx2.8.4pre.1.tar.bz2
3748229	Jun 3	13:19	lynx2.8.4pre.1.tar.Z
2753233	Jun 3	12:58	lynx2.8.4dev.21.zip
2547941	Jun 3	12:58	lynx2.8.4dev.21.tar.gz
1932457	Jun 3	12:58	lynx2.8.4dev.21.tar.bz2

```
3736689 Jun  3 12:58 lynx2.8.4dev.21.tar.Z
2754141 Apr  1 17:51 lynx2.8.4dev.20.zip
2549461 Apr  1 17:51 lynx2.8.4dev.20.tar.gz
1932083 Apr  1 17:51 lynx2.8.4dev.20.tar.bz2
3735454 Apr  1 17:51 lynx2.8.4dev.20.tar.Z
2749046 Feb 26 18:41 lynx2.8.4dev.19.zip
2544539 Feb 26 18:41 lynx2.8.4dev.19.tar.gz
1928297 Feb 26 18:41 lynx2.8.4dev.19.tar.bz2
3737823 Feb 26 18:41 lynx2.8.4dev.19.tar.Z
2738849 Feb 12 17:33 lynx2.8.4dev.18.zip
2534550 Feb 12 17:33 lynx2.8.4dev.18.tar.gz
1919641 Feb 12 17:33 lynx2.8.4dev.18.tar.bz2
3720345 Feb 12 17:33 lynx2.8.4dev.18.tar.Z
2730847 Feb  8 18:50 lynx2.8.4dev.17.zip
2526050 Feb  8 18:50 lynx2.8.4dev.17.tar.gz
1910297 Feb  8 18:50 lynx2.8.4dev.17.tar.bz2
3704921 Feb  8 18:50 lynx2.8.4dev.17.tar.Z
```

Version-level Patches

```
Size    Date   Time  Filename
======= =====  ===== =========
   1501 Jul 10 10:14 2.8.4pre.4.patch.gz
  37783 Jul  7 18:33 2.8.4pre.3.patch.gz
   4653 Jun 10 18:07 2.8.4pre.2.patch.gz
 512172 Jun  3 14:12 2.8.4dev.21.patch.gz
    574 Jun  3 13:21 2.8.4pre.1.patch.gz
 111475 Apr  1 17:55 2.8.4dev.20.patch.gz
  81856 Feb 26 18:44 2.8.4dev.19.patch.gz
  52319 Feb 12 17:35 2.8.4dev.18.patch.gz
 334318 Feb  8 18:52 2.8.4dev.17.patch.gz
  97242 Jan  1  2001 2.8.4dev.16.patch.gz
 437204 Dec 21  2000 2.8.4dev.15.patch.gz
   4062 Nov  3  2000 2.8.4dev.14.patch.gz
    908 Oct 25  2000 2.8.4dev.13.patch.gz
  73256 Oct 25  2000 2.8.4dev.12.patch.gz
  13928 Oct 18  2000 2.8.4dev.11.patch.gz
  27938 Sep 21  2000 2.8.4dev.10.patch.gz
```

```
 22105 Sep  1  2000 2.8.4dev.9.patch.gz
 28262 Aug 24  2000 2.8.4dev.8.patch.gz
 69466 Aug  3  2000 2.8.4dev.7.patch.gz
 41358 Jul 17  2000 2.8.4dev.6.patch.gz
 83227 Jul 16  2000 2.8.4dev.5.patch.gz
 49003 Jun 23  2000 2.8.4dev.4.patch.gz
 13431 Jun  2  2000 2.8.4dev.3.patch.gz
   679 Jun  2  2000 lynx2.8.3rel.1c.patch.gz
 62706 May 21  2000 2.8.4dev.2.patch.gz
  2851 May 21  2000 lynx2.8.3rel.1b.patch.gz
  6793 May  5  2000 2.8.4dev.1.patch.gz
  6621 May  5  2000 lynx2.8.3rel.1a.patch.gz
  2278 Apr 23  2000 2.8.3rel.1.patch.gz
```

Release Version in Various Formats

```
Size    Date   Time  Filename
======= =====  ===== =========
    679 Jun  2  2000 lynx2.8.3rel.1c.patch.gz
   2851 May 21  2000 lynx2.8.3rel.1b.patch.gz
   6621 May  5  2000 lynx2.8.3rel.1a.patch.gz
2245486 Apr 23  2000 lynx2.8.3rel.1.zip
2037594 Apr 23  2000 lynx2.8.3rel.1.tar.gz
1631069 Apr 23  2000 lynx2.8.3rel.1.tar.bz2
3040475 Apr 23  2000 lynx2.8.3rel.1.tar.Z
```

Important Documents

```
Size    Date   Time  Filename
======= =====  ===== =========
 377098 Jul 10 10:05 CHANGES
  18096 Dec 12  1998 CHANGES2.3
  48386 Dec 12  1998 CHANGES2.4
  93248 Dec 12  1998 CHANGES2.5
  41847 Dec 24  1997 CHANGES2.6
  48660 Dec 24  1997 CHANGES2.7
```

```
203972 May 24  1999 CHANGES2.8
 47322 Jul  7 18:30 INSTALLATION
 10394 Jun 10 18:04 PROBLEMS
  7226 Jan  1  2001 README
  7922 Jul 10 10:21 README.TRST
  7082 Jul 10 10:21 README.chartrans
  6892 Apr  1 17:51 README.defines
  5070 Mar 31  2000 README.jp
  2635 Jan  1  2001 README.ssl
```

DOS/Win32

A release of **Lynx** 2.8.4 (dev10 or higher) is available from:
http://www.shonai-cit.ac.jp/eci/senshu/ Note it is compiled for Japanese.
A different release of **Lynx** and pointers to **Lynx** for DOS386+ and Win32 are available from:
http://www.fdisk.com/doslynx/lynxport.htm
A release of **Lynx** built for Win32 with Borland may be found via:
http://www.jim.spath.com/lynx_win32/

gz, bz2, zip

Gzip can be obtained from any Gnu archive site, e.g., ftp://ftp.gnu.org/pub/gnu/gzip/

Bzip 2 home page http://sourceware.cygnus.com/bzip2/

Info-Zip home page http://www.info-zip.org/

UNIX binaries and packages

[This section under construction]
You can get some Lynx binaries via the LYBIDO pages.
You can also use your systems' port or package system:

NetBSD

- Packages: README.html includes ssl, NLS and IPV6 support
- lynx-current includes ssl and NLS support

FreeBSD

- Ports for lynx: includes Japanese and ssl flavours

OpenBSD

- Already included in 2.5

Natural Language Support / Internationalization

Lynx supports multiple languages through use of gettext. Some message catalogs (preliminary, incomplete, etc.) are available on http://lynx.isc.org/po/ If this site mirrors the full original site, this is here as well. Also read the notes in About NLS.

Questions or comments can be directed to the lynx-dev@sig.net mailing list, which is archived at http://www.flora.org/lynx-dev/html/ Updated by: Jim Spath [Webmaster Jim] assisted by: Thomas Dickey [email]

Extremely Lynx

[Web starting points] [Download Lynx] [What is Lynx?]

Lynx information @ <URL:http://lynx.browser.org/>

Table of Contents

Getting Lynx sources and binaries

The latest release is Lynx **version 2.8.3** Some details concerning Lynx 2.8.3 (including new features) are explored on the release page. Lynx 2.8.3 is distributed under the GNU Public License. The Lynx Source distribution page has links to sites which carry the Lynx source code (SSL/https info here). The LYnx BInary Distribution Outlet (LYBIDO) lists Lynx binaries (.exe's) made available to lynx users.

Of Interest to Lynx Users

Lynx was originally developed at the University of Kansas. In 1995 Lynx 2.4.2 was released under the GNU Public License by UKans, since then it has been maintained and extended by a group of volunteers. More details are in about_lynx.html

- Lynx Development: what's being done to adapt Lynx to the evolving web
- jeffwong's Lynx page has information on web design, SSL, and Wyse terminals
- Documentation especially developed for speech, sight impaired, users of Lynx
- Wondering why Lynx is behaving like it is? some common 'problems' are explained here
- Found a bug in Lynx? here's what you should do
- The report on ISPs and Lynx, who's got the latest Lynx and tips on how to get your ISP to upgrade Lynx
- Raju Mathur's guide to running lynx in a Linux console
- Some other programs you may be interested in:
  - wget a simple perl script to retreive http resources
- Lynx-Dev archives:
  - http://www.flora.org/lynx-dev/
  - Search the Lynx-Dev archive with your favourite search engine
- Lynx Support and Lynx resources world-wide
- Help menus for the latest Lynx, power Lynx users know them backwards, do you? If you don't, you can try searching through the help files

- Introduction to "Lynx enhancement"
- Pages enhanced for Lynx
- Appreciative mail from Lynx users.
- Yahoo's Lynx page

Resources for developers, system administrators and web authors

- Security related issues in Lynx 2.7 have been resolved in Lynx 2.8 and later
- Patches to enhance Lynx's functionality in particular environments
- Platforms for which Lynx is available, and the utilitarian guide to building and installing Lynx
- HTML Pro, a "composite DTD" that incorporates most of the markup Lynx recognizes.
- Designing Web pages, primary resources for web authors
- Lynx tips for web authors
- See Lynx for yourself, public Lynx access and Lynx View for web authors.
- Documents from the Lynx 2.8 distribution, including help pages

[ToC]

This page is Lynx Enhanced and proud of it.


| Validate this page |  | Any Browser! | GNU

This site is supported by 

This page is maintained by Subir Grewal, comments should be addressed to hostmaster@trill-home.com.

Lynx help files (usually in your local directories):

- Lynx Users Guide -- complete account of all Lynx features
- Key-stroke Commands -- quick outline of what various keys do
- Line Editor -- when entering URLs etc
- Supported URLs -- how Lynx handles various types of URL
- About Lynx -- credits, copyright etc
- About Lynx-Dev -- the developers & how to contact them

Other sources of Lynx help:

- lynx.cfg options -- a reference for advanced configurations
- Lynx Help for Beginners -- quick help on many common problems
- The Lynx FAQ-O-Matic -- many common queries, some more advanced; users may add answers.
- Lynx Links -- source & binaries, FAQs, developers & archives, SSL & security, and more
- Lynxstuff -- SSL, Wyse terminals, Lynx-friendly Web design
- Blynx -- Speech-Friendly Help for the visually impaired

World Wide Web Consortium documents:

- HTML -- 4.0 -- 3.2 -- 3.0 -- 2.0
- HTTP -- 1.1 -- 1.0
- Web Naming & Addressing Overview: URIs, URLs etc
- HTML Internationalisation
- WWW Consortium: home page

Help with HTML:

- HTML 4.0 Reference
- NCSA Beginner's Guide To HTML
- HTML Quick Reference Guide
- Spyglass/Stonehand Technical Reference

HTML validation services:

- W3C HTML Validation Service
- WDG HTML Validator

Other browsing software:

- WGET -- powerful & flexible non-interactive downloader
- cURL -- non-interactive downloader which supports HTTPS
- SNARF -- small simple 1-file non-interactive downloader

Meta-indexes: lists of links

- NCSA Mosaic

Search engines:

- AltaVista
- Excite
- Inference Find
- Infoseek
- Lycos
- MetaCrawler
- Savvy Search
- WebCrawler
- Yahoo!

Free WWW E-mail services:

- Eudoramail
- Hotmail
- Netaddress
- Yahoo!

```
#include <stdio.h>

int
main(int argc, char* argv[])
{
    printf ("hello world\n");

    return 0;
}
```

```
#include <iostream.h>

int
main(int argc, char* argv[])
{
    cout << "hello world" << endl;

    return 0;
}
```

```c
#include <stdio.h>

/* define some external functions */
extern int func_a();
extern int func_b();

int
main(int argc, char* argv[])
{
    int a = func_a();
    int b = func_b();
    char c;
    char* bc = &c;

    printf("hello world,\n");
    printf("a - %d; b - %d;\n", a, b);

    return 0;
}
```

```
int
func_a()
{
    return 5;
}
```

```
int
func_b()
{
    return 10 * 10;
}
```

Index of /~choo/lupg/tutorials/c-on-unix/multi-source

- Parent Directory
- a.c
- b.c
- main.c

```c
#include <stdio.h>

/* print a given string and a number if a pre-determined format. */
void
print_string(int num, char* string)
{
    printf("String '%d' - '%s'\n", num, string);
}

int
main(int argc, char* argv[])
{
    int i;

    /* check for command line arguments */
    if (argc < 2) { /* 2 - 1 for program name (argv[0]) and one for a param. */
        printf("Usage: %s [<string> ...]\n", argv[0]);
        exit(1);
    }

    /* loop over all strings, print them one by one */
    for (argc--,argv++,i=1 ; argc > 0; argc--,argv++,i++) {
        print_string(i, argv[0]);  /* function call */
    }

    printf("Total number of strings: %d\n", i);

    return 0;
}
```

```
/*
 * stdc-file-copy.c - copy one file to a new location, possibly under a
 * different name.
 */

#include <stdio.h>              /* standard input/output routines.    */

#define MAX_LINE_LEN 1000       /* maximal line length supported.     */

/*
 * function: main. copy the given source file to the given target file.
 * input:    path to source file and path to target file.
 * output:   target file is being created with identical contents to
 *           source file.
 */
void
main(int argc, char* argv[])
{
    char* file_path_from;       /* path to source file.   */
    char* file_path_to;         /* path to target file.   */
    FILE* f_from;               /* stream of source file. */
    FILE* f_to;                 /* stream of target file. */
    char buf[MAX_LINE_LEN+1];   /* input buffer.          */

    /* read command line arguments */
    if (argc != 3 || !argv[1] || !argv[2]) {
        fprintf(stderr, "Usage: %s <source file path> <target file path>\n",
                argv[0]);
        exit(1);
    }
    file_path_from = argv[1];
    file_path_to = argv[2];

    /* open the source and the target files. */
    f_from = fopen(file_path_from, "r");
    if (!f_from) {
        fprintf(stderr, "Cannot open source file: ");
        perror("");
        exit(1);
    }
    f_to = fopen(file_path_to, "w+");
    if (!f_from) {
        fprintf(stderr, "Cannot open target file: ");
        perror("");
        exit(1);
    }

    /* copy source file to target file, line by line. */
    while (fgets(buf, MAX_LINE_LEN+1, f_from)) {
        if (fputs(buf, f_to) == EOF) {  /* error writing data */
            fprintf(stderr, "Error writing to target file: ");
            perror("");
            exit(1);
        }
    }
    if (!feof(f_from)) { /* fgets failed _not_ due to encountering EOF */
        fprintf(stderr, "Error reading from source file: ");
        perror("");
        exit(1);
    }
```

```
    /* close source and target file streams. */
    if (fclose(f_from) == EOF) {
        fprintf(stderr, "Error when closing source file: ");
        perror("");
    }
    if (fclose(f_to) == EOF) {
        fprintf(stderr, "Error when closing target file: ");
        perror("");
    }
}
```

```
/*
 * stdc-small-db.c - handle a small database file, with fixed-length records.
 * uses standard C library I/O functions, including fopen, fseek, fgetpos,
 * fsetpos etc.
 */

#include <stdio.h>              /* standard input/output routines.    */

#define RECORD_LEN 30           /* size of a record.                  */

/*
 * function: get_record. Reads a record from the given file.
 * input:    pointer to file stream, number of record in file (0 is
 *           the first record), and a pointer to a buffer.
 * output:   '1' on success, '0' on failure. In case of success, the record
 *           is copied to the supplied buffer.
 */
int
get_record(FILE* file, long record_num, char* buf)
{
    long file_pos;    /* stores the original read/write pointer position. */
    int success = 0;  /* success/failure indicator. assume failure here.  */

    /* sanity checks */
    if (!file || record_num < 0 || !buf) {
        return 0;
    }

    /* note the current read/write pointer position.     */
    file_pos = ftell(file);
    if (file_pos < 0) {
        perror("get_record: ftell");
    }
    else {
        /* move to the record's location.                */
        if (fseek(file, record_num * RECORD_LEN, SEEK_SET) < 0) {
            perror("get_record: fseek");
        }
        else {
            /* read the record from the file stream.     */
            if (fread(buf, RECORD_LEN, 1, file) == 1) {
                success = 1;  /* mark that we succeeded in reading the record */
            }
            else {
                perror("get_record: fread");
            }

            /* restore the original read/write pointer position. */
            if (fseek(file, file_pos, SEEK_SET) < 0) {
                perror("get_record: fseek");
            }
        }
    }

    return success;
}

/*
 * function: put_record. Writes a record to the given file.
 * input:    pointer to file stream, number of the record in file (0 is
```

```
 *           the first record)  and a pointer to a buffer containing the
 *           data to be written.
 * output:   '1' on success, '0' on failure.
 */
int
put_record(FILE* file, long record_num, char* buf)
{
    long file_pos;    /* stores the original read/write pointer position. */
    int success = 0;  /* success/failure indicator. assume failure here.  */

    /* sanity checks */
    if (!file || record_num < 0 || !buf) {
        return 0;
    }

    /* note the current read/write pointer position.     */
    file_pos = ftell(file);
    if (file_pos < 0) {
        perror("put_record: ftell");
    }
    else {
        /* move to the record's location.                */
        if (fseek(file, record_num * RECORD_LEN, SEEK_SET) < 0) {
            perror("put_record: fseek");
        }
        else {
            /* write the record to the file stream.      */
            if (fwrite(buf, RECORD_LEN, 1, file) == 1) {
                success = 1;  /* mark that we succeeded in writing the record */
            }
            else {
                perror("put_record: fwrite");
            }

            /* restore the original read/write pointer position. */
            if (fseek(file, file_pos, SEEK_SET) < 0) {
                perror("put_record: ftell");
            }
        }
    }

    return success;
}

/*
 * function: main. write and read some records to/from a file.
 * input:    none.
 * output:   none.
 */
void
main()
{
    char* file_name = "small_db.data";
    char buf[RECORD_LEN];
    FILE* db_file;

    /* open the database file, in read and write mode. */
    db_file = fopen(file_name, "w+");
    if (!db_file) {
        perror("fopen");
```

```
        exit(1);
    }

    /* write some records into the database file.    */
    strcpy(buf, "hello world");
    put_record(db_file, 5, buf);
    strcpy(buf, "adios, amigos");
    put_record(db_file, 17, buf);
    strcpy(buf, "marchaba marchabtain");
    put_record(db_file, 12, buf);

    /* now read some records from the database file. */
    get_record(db_file, 17, buf);
    printf("record %d: '%s'\n", 17, buf);
    get_record(db_file, 5, buf);
    printf("record %d: '%s'\n", 5, buf);
    get_record(db_file, 3, buf); /* never written - should be garbage */
    printf("record %d: '%s'\n", 3, buf);

    if (fclose(db_file) == EOF) {
        perror("fclose");
        exit(1);
    }
}
```

```
/*
 * read-access-check.c - check if the user has read permission to a given
 *                       file. If not, check where the problem lies.
 *
 * Background: In order to read from a file, we must have execute permission
 *             in all directories leading to this file, as well as read access
 *             to the file itself. For example, to read from file
 *             "/etc/config/blabla.conf", we must have X permission on '/etc'
 *             and on '/etc/config', as well as R permission on
 *             "/etc/config/blabla.conf" itself.
 *
 * Algorithm: split the file path to its directories, make sure the directory
 *            exists and that the user has execute permission to all
 *            directories. Finally check that the file exists, and that the
 *            user has read permission to the file itself.
 */

#include <stdio.h>              /* standard input/output routines.    */
#include <unistd.h>             /* access(), R_OK etc.                */
#include <string.h>             /* strchr(), etc.                     */
#include <malloc.h>             /* malloc(), etc.                     */

/*
 * function: main.
 * input:    full path to a file.
 * output:   'OK' if permission is granted, or 'ACCESS DENIED' and a
 *           detailed explanation if some problem was found.
 */
void
main(int argc, char* argv[])
{
    char* file_path;            /* full path to file.           */
    char* dir_path;             /* full path to a directory.    */
    char* p_slash;              /* location of next '/' in path. */

    /* read command line arguments */
    if (argc != 2 || !argv[1]) {
        fprintf(stderr, "Usage: %s <full file path>\n", argv[0]);
        exit(1);
    }
    file_path = argv[1];
    dir_path = (char*)malloc(strlen(file_path)+1);
    if (!dir_path) {
        fprintf(stderr, "out of memory\n");
        exit(1);
    }

    /* scan all directories in the path and check each of them. */
    p_slash = file_path;
    while ( (p_slash = strchr(p_slash, '/')) != NULL) {
        /* copy directory path (including trailing slash) into dir_path. */
        strncpy(dir_path, file_path, p_slash-file_path+1);
        dir_path[p_slash-file_path+1] = '\0';

        /* check existance and execute permission for this directory. */
        if (access(dir_path, F_OK) == -1) {
            printf("%s: Directory '%s' in the path does not exist.\n",
                   "ACCESS DENIED", dir_path);
            exit(2);
        }
```

```
        if (access(dir_path, X_OK) == -1) {
            printf("%s: Directory '%s' in the path - no 'X' permission.\n",
                   "ACCESS DENIED", dir_path);
            exit(2);
        }
        /* skip the current slash, so the next strchr() call will find */
        /* the next slash in the file path.                             */
        p_slash++;
    }

    /* all directories in the path exist and are executable.   */
    /* now check existance and read access to the file itself. */
    if (access(file_path, F_OK) == -1) {
        printf("%s: File does not exist.\n", "ACCESS DENIED");
        exit(2);
    }
    if (access(file_path, R_OK) == -1) {
        printf("%s: no read access to file.\n", "ACCESS DENIED");
        exit(2);
    }

    /* all tests passed. */
    printf("OK\n");
}
```

```
/*
 * rename-log.c - check the size of a given log file in the '/tmp/var/log'
 *                directory. If it is larger then 1024KB, rename it to
 *                a have a '.old' suffix and create a new, empty log file.
 *
 * IMPORTANT NOTE: we intentionally use a fictitious log directory. Do NOT
 *                 use this program against your system's real log directory.
 */

#include <stdio.h>              /* standard input/output routines.    */
#include <unistd.h>             /* access(), etc.                     */
#include <sys/stat.h>           /* stat(), etc.                       */
#include <fcntl.h>              /* open(), close(), etc.              */
#include <malloc.h>             /* malloc(), etc.                     */
#include <string.h>             /* strcpy(), strcat().                */

#define LOG_DIR "/tmp/var/log"  /* full path to the logs directory. */
#define FILE_SIZE 1024*1024     /* size of file to rename - 1024KB. */
#define FILE_SUFFIX ".old"      /* suffix for old log file.         */

/*
 * function: main.
 * input:    name of a log file.
 * output:   log file is being renamed if it is too big.
 */
void
main(int argc, char* argv[])
{
    char* file_name;            /* name of the log file.        */
    char* new_file_name;        /* new name for the log file.   */
    struct stat file_status;    /* status info of the log file. */
    int fd;                     /* descriptor for new file.     */
    umode_t file_mode;          /* permissions of old log file. */

    /* read command line arguments */
    if (argc != 2 || !argv[1]) {
        fprintf(stderr, "Usage: %s <log file name>\n", argv[0]);
        exit(1);
    }
    file_name = argv[1];
    new_file_name = (char*)malloc(strlen(file_name)+strlen(FILE_SUFFIX)+1);
    if (!new_file_name) {
        fprintf(stderr, "Out of memory\n");
        exit(1);
    }
    strcpy(new_file_name, file_name);
    strcat(new_file_name, FILE_SUFFIX);

    /* check we have read/write permissions to the /tmp/var/log directory. */
    if (access(LOG_DIR, F_OK) == -1) {
        fprintf(stderr, "Directory '%s' does not exist.\n", LOG_DIR);
        exit(1);
    }
    if (access(LOG_DIR, R_OK | W_OK) == -1) {
        fprintf(stderr, "Directory '%s': access denied.\n", LOG_DIR);
        exit(1);
    }

    /* switch current directory to the logs directory. */
    if (chdir(LOG_DIR) == -1) {
```

```
        fprintf(stderr, "Cannot change directory to '%s':", LOG_DIR);
        perror("");
        exit(1);
    }

    /* check the size of the file. */
    if (stat(file_name, &file_status) == -1) {
        fprintf(stderr, "Cannot stat file '%s':", file_name);
        perror("");
        exit(1);
    }

    if (file_status.st_size > FILE_SIZE) {
        if (rename(file_name, new_file_name) == -1) {
            fprintf(stderr, "Cannot rename '%s' to '%s':",
                    file_name, new_file_name);
            perror("");
            exit(1);
        }
        /* create a new, empty log file, with the same access permissions */
        /* as the original log file had.                                  */
        file_mode = file_status.st_mode & ~S_IFMT;
        umask(0);
        fd = open(file_name, O_WRONLY | O_CREAT | O_TRUNC, file_mode);
        if (fd < 0) {
            fprintf(stderr, "Failed creating empty log file:");
            perror("");
            exit(1);
        }
        close(fd);
    }
}
```

```
/*
 * find-file.c - find all files residing in the given sub-directory,
 * whose names contain the given string.
 */

#include <stdio.h>              /* standard input/output routines.    */
#include <dirent.h>             /* readdir(), etc.                    */
#include <sys/stat.h>           /* stat(), etc.                       */
#include <string.h>             /* strstr(), etc.                     */
#include <unistd.h>             /* getcwd(), etc.                     */

#define MAX_DIR_PATH 2048       /* maximal full path we support.      */

/*
 * function: findfile. recusively traverse the current directory, searching
 *                     for files with a given string in their name.
 * input:    string to match.
 * output:   any file found, printed to the screen with a full path.
 */
void
findfile(char* pattern)
{
    DIR* dir;                   /* pointer to the scanned directory. */
    struct dirent* entry;       /* pointer to one directory entry.   */
    char cwd[MAX_DIR_PATH+1];   /* current working directory.        */
    struct stat dir_stat;       /* used by stat().                   */

    /* first, save path of current working directory */
    if (!getcwd(cwd, MAX_DIR_PATH+1)) {
        perror("getcwd:");
        return;
    }


    /* open the directory for reading */
    dir = opendir(".");
    if (!dir) {
        fprintf(stderr, "Cannot read directory '%s': ", cwd);
        perror("");
        return;
    }

    /* scan the directory, traversing each sub-directory, and */
    /* matching the pattern for each file name.               */
    while ((entry = readdir(dir))) {
        /* check if the pattern matchs. */
        if (entry->d_name && strstr(entry->d_name, pattern)) {
            printf("%s/%s\n", cwd, entry->d_name);
        }
        /* check if the given entry is a directory. */
        if (stat(entry->d_name, &dir_stat) == -1) {
            perror("stat:");
            continue;
        }
        /* skip the "." and ".." entries, to avoid loops. */
        if (strcmp(entry->d_name, ".") == 0)
            continue;
        if (strcmp(entry->d_name, "..") == 0)
            continue;
        /* is this a directory? */
```

```
        if (S_ISDIR(dir_stat.st_mode)) {
            /* Change into the new directory */
            if (chdir(entry->d_name) == -1) {
                fprintf(stderr, "Cannot chdir into '%s': ", entry->d_name);
                perror("");
                continue;
            }
            /* check this directory */
            findfile(pattern);

            /* finally, restore the original working directory. */
            if (chdir("..") == -1) {
                fprintf(stderr, "Cannot chdir back to '%s': ", cwd);
                perror("");
                exit(1);
            }
        }
    }
}

/*
 * function: main. find files with a given pattern in their names, in
 *                 a given directory tree.
 * input:    path to directory to search, and pattern to match.
 * output:   names of all matching files on the screen.
 */
void
main(int argc, char* argv[])
{
    char* dir_path;             /* path to the directory. */
    char* pattern;              /* pattern to match.      */
    struct stat dir_stat;       /* used by stat().        */

    /* read command line arguments */
    if (argc != 3 || !argv[1] || !argv[2]) {
        fprintf(stderr, "Usage: %s <directory path> <file name pattern>\n",
                argv[0]);
        exit(1);
    }
    dir_path = argv[1];
    pattern = argv[2];

    /* make sure the given path refers to a directory. */
    if (stat(dir_path, &dir_stat) == -1) {
        perror("stat:");
        exit(1);
    }
    if (!S_ISDIR(dir_stat.st_mode)) {
        fprintf(stderr, "'%s' is not a directory\n", dir_path);
        exit(1);
    }

    /* change into the given directory. */
    if (chdir(dir_path) == -1) {
        fprintf(stderr, "Cannot change to directory '%s': ", dir_path);
        perror("");
        exit(1);
    }

    /* recursively scan the directory for the given file name pattern. */
```

```
    findfile(pattern);
}
```

Index of /~choo/lupg/tutorials/libraries/static-lib

- Parent Directory
- Makefile
- main.c
- util_file.c
- util_math.c
- util_net.c

```
static C library example
------------------------

This directory contains a Makefile and a set of source files that
will create an executable program using a static library.

To run the makefile, just type 'make'. It is also possible to build
the library only, by typing 'make libutil.a'. If you remove the object
files of the library ('make cleanlibobjs'), then remove the program
('make cleanprog') and then try 'make prog', you will see the program
still compiles, since all the object files were copied into the library.
Doing a 'make' again will recompile all the library's object files as
well.

Thus, when a company builds a static library and gives it away, it does
not have to give all the object files - the header files, and the
library's archive file will do.
```

Index of /~choo/lupg/tutorials/libraries/shared-lib

- Parent Directory
- Makefile
- main.c
- util_file.c
- util_math.c
- util_net.c

```
shared C library example
------------------------

This directory contains a Makefile and a set of source files that
will create an executable program using a shared library.

To run the makefile, just type 'make'. It is also possible to build
the library only, by typing 'make libutil.a'. If you remove the object
files of the library ('make cleanlibobjs'), then remove the program
('make cleanprog') and then try 'make prog', you will see the program
still compiles, since all the object files were copied into the library.
Doing a 'make' again will recompile all the library's object files as
well.

Thus, when a company builds a static library and gives it away, it does
not have to give all the object files - the header files, and the
library's archive file will do.

Note that in order to run the program, you need first to add the current
directory to your "LD_LIBRARY_PATH" environment variable. If it is
not defined yet ("echo $LD_LIBRARY_PATH" sais its not defined), then
define it with:

  If using a shell such as 'csh' or 'tcsh':

    setenv LD_LIBRARY_PATH /full/path/to/current/directory

  If using a shell such as 'sh' or 'bash' or 'ksh':

    LD_LIBRARY_PATH=/full/path/to/current/directory
    export LD_LIBRARY_PATH

If you already have something in your LD_LIBRARY_PATH, then modify it with:

  If using a shell such as 'csh' or 'tcsh':

    setenv LD_LIBRARY_PATH /full/path/to/current/directory:${LD_LIBRARY_PATH}
```

If using a shell such as 'sh' or 'bash' or 'ksh':

```
LD_LIBRARY_PATH=/full/path/to/current/directory:${LD_LIBRARY_PATH}
export LD_LIBRARY_PATH
```

To check that you did that properly, run "ldd prog" - it should
show the name of our library (libutil.so) and point to the library
in the current directory. If it does not, check that your LD_LIBRARY_PATH
contains the correct path to the current directory, and that the library
was properly built.

Index of /~choo/lupg/tutorials/libraries/dynamic-shared

- Parent Directory
- Makefile
- lib1.c
- lib2.c
- main.c

```
dynamically loaded shared C library example
-------------------------------------------

This directory contains a Makefile and a set of source files that
will create an executable program that uses one of two shared libraries,
making the decision at runtime.

To run the makefile, just type 'make'. It is also possible to build
the library only, by typing 'make libutil.a'. If you remove the object
files of the library ('make cleanlibobjs'), then remove the program
('make cleanprog') and then try 'make prog', you will see the program
still compiles, since all the object files were copied into the library.
Doing a 'make' again will recompile all the library's object files as
well.

If you remove the shared libraries themselves altogether ('make cleanlibs'),
you will get an error message only during runtime of the program.

Note that in order to run the program, you need first to add the current
directory to your "LD_LIBRARY_PATH" environment variable. If it is
not defined yet ("echo $LD_LIBRARY_PATH" sais its not defined), then
define it with:

  If using a shell such as 'csh' or 'tcsh':

    setenv LD_LIBRARY_PATH /full/path/to/current/directory

  If using a shell such as 'sh' or 'bash' or 'ksh':

    LD_LIBRARY_PATH=/full/path/to/current/directory
    export LD_LIBRARY_PATH

If you already have something in your LD_LIBRARY_PATH, then modify it with:

  If using a shell such as 'csh' or 'tcsh':

    setenv LD_LIBRARY_PATH /full/path/to/current/directory:${LD_LIBRARY_PATH}

  If using a shell such as 'sh' or 'bash' or 'ksh':
```

```
    LD_LIBRARY_PATH=/full/path/to/current/directory:${LD_LIBRARY_PATH}
    export LD_LIBRARY_PATH
```

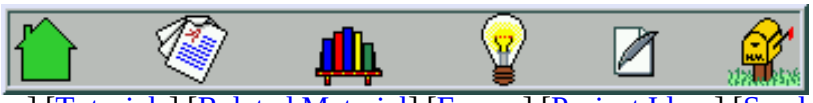
To check that you did that properly, you'll need to run the program.
If it sais that the library file was not found, check the definition of
LD_LIBRARY_PATH, and the existance of the library in the current directory.

```
#include <stdio.h>     /* standard I/O functions                          */
#include <unistd.h>    /* standard unix functions, like getpid()         */
#include <signal.h>    /* signal name macros, and the signal() prototype */

/* first, here is the signal handler */
void catch_int(int sig_num)
{
    /* re-set the signal handler again to catch_int, for next time */
    signal(SIGINT, catch_int);
    printf("Don't do that\n");
    fflush(stdout);
}

int main(int argc, char* argv[])
{
    /* set the INT (Ctrl-C) signal handler to 'catch_int' */
    signal(SIGINT, catch_int);

    /* now, lets get into an infinite loop of doing nothing. */
    for ( ;; )
        pause();
}
```

```
#include <stdio.h>     /* standard I/O functions                          */
#include <unistd.h>    /* standard unix functions, like getpid()         */
#include <signal.h>    /* signal name macros, and the signal() prototype */

/* first, define the Ctrl-C counter, initialize it with zero. */
int ctrl_c_count = 0;
#define CTRL_C_THRESHOLD       5

/* the Ctrl-C signal handler */
void catch_int(int sig_num)
{
    sigset_t mask_set;  /* used to set a signal masking set. */
    sigset_t old_set;   /* used to store the old mask set.   */

    /* re-set the signal handler again to catch_int, for next time */
    signal(SIGINT, catch_int);
    /* mask any further signals while we're inside the handler. */
    sigfillset(&mask_set);
    sigprocmask(SIG_SETMASK, &mask_set, &old_set);

    /* increase count, and check if threshold was reached */
    ctrl_c_count++;
    if (ctrl_c_count >= CTRL_C_THRESHOLD) {
        char answer[30];

        /* prompt the user to tell us if to really exit or not */
        printf("\nRealy Exit? [y/N]: ");
        fflush(stdout);
        gets(answer);
        if (answer[0] == 'y' || answer[0] == 'Y') {
            printf("\nExiting...\n");
            fflush(stdout);
            exit(0);
        }
        else {
            printf("\nContinuing\n");
            fflush(stdout);
            /* reset Ctrl-C counter */
            ctrl_c_count = 0;
        }
    }
    /* restore the old signal mask */
    sigprocmask(SIG_SETMASK, &old_set, NULL);
}

/* the Ctrl-Z signal handler */
void catch_suspend(int sig_num)
{
    sigset_t mask_set;  /* used to set a signal masking set. */
    sigset_t old_set;   /* used to store the old mask set.   */

    /* re-set the signal handler again to catch_suspend, for next time */
    signal(SIGTSTP, catch_suspend);
    /* mask any further signals while we're inside the handler. */
    sigfillset(&mask_set);
    sigprocmask(SIG_SETMASK, &mask_set, &old_set);

    /* print the current Ctrl-C counter */
    printf("\n\nSo far, '%d' Ctrl-C presses were counted\n\n", ctrl_c_count);
    fflush(stdout);
```

```
    /* restore the old signal mask */
    sigprocmask(SIG_SETMASK, &old_set, NULL);
}

int main(int argc, char* argv[])
{
    /* set the Ctrl-C and Ctrl-Z signal handlers */
    signal(SIGINT, catch_int);
    signal(SIGTSTP, catch_suspend);

    /* enter an infinite loop of waiting for signals */
    for ( ;; )
        pause();

    return 0;
}
```

```
#include <stdio.h>     /* standard I/O functions                          */
#include <unistd.h>    /* standard unix functions, like alarm()           */
#include <signal.h>    /* signal name macros, and the signal() prototype */

char user[40];          /* buffer to read user name from the user */

/* define an alarm signal handler. */
void catch_alarm(int sig_num)
{
    printf("Operation timed out. Exiting...\n\n");
    exit(0);
}

int main(int argc, char* argv[])
{
    /* set a signal handler for ALRM signals */
    signal(SIGALRM, catch_alarm);

    /* prompt the user for input */
    printf("Username: ");
    fflush(stdout);
    /* start a 30 seconds alarm */
    alarm(30);
    /* wait for user input */
    gets(user);
    /* remove the timer, now that we've got the user's input */
    alarm(0);

    printf("User name: '%s'\n", user);

    return 0;
}
```



Network programming under Unix systems - Part II

3. Preparing an Internet address.

(Skip this if you know how to obtain the address of a given machine, prepare the structures involved, and deal with byte order translations)

Preview

The first thing you need to do when writing network applications, is obtain the addresses of the two involved machines: your address, and the remote host's address. This process is made up of several stages:

1. Address Resolution - Translating the host name into an IP address.
2. Byte Ordering - Translating host byte order into network byte order.
3. Address Formation - Forming up the remote address in the right structure.

Address Resolution

Given a host name, we want to find it's IP address. We do that using the function gethostbyname(). This function is defined in the file /usr/include/netdb.h (or the equivalent for your system) as follows:

```
struct hostent *gethostbyname(char *hostname);
```

The input to the function will be the name of the host whose address we want to resolve. The function returns a pointer to a structure hostent, whose definition is as follows:

```
struct    hostent {
     char*   h_name;          /* official name of host */
     char**  h_aliases;       /* alias list */
     int     h_addrtype;      /* host address type */
     int     h_length;        /* length of address */
     char**  h_addr_list;     /* list of addresses from name server */
#define h_addr h_addr_list[0] /* address, for backward compatibility */
};
```

Lets see what each field in the hostent structure means:

- `h_name`: This is the official name of the host, i.e. the full address.
- `h_aliases`: a pointer to the list of aliases (other names) the host might have.
- `h_addrtype`: The type of address this host uses.
- `h_length`: The length of the address. Different address types might have different lengths.
- `h_addr_list`: A pointer to the list of addresses of the host. Note that a host might have more then one

address, as explained earlier.

- h_addr: In older systems, there was only the h_addr field, so it is defined here so old programs could compile without change on newer systems.

Byte Ordering

As explained earlier, we need to form addresses using the network byte order. Luckily, most networking functions accept addresses in host byte order, and return their results in network byte order. This means only a few fields will need conversions. These will include the port numbers only, as the addresses are already supplied by the system.

We normally have several functions (or macros) to form 4 types of translations:

- htons() - short integer from host byte order to network byte order.
- ntohs() - short integer from network byte order to host byte order.
- htonl() - long integer from host byte order to network byte order.
- ntohl() - long integer from network byte order to host byte order.

For example, since a port number is represented by a short integer, we could convert it from host byte order to network byte order by doing:

```
short host_port = 1234;
net_port = htons(host_port);
```

The other functions are used in a similar way.

Address Formation

Forming addresses for Internet protocols is done using a structure named sockaddr_in, whose definition, as given in the file /usr/include/netinet/in.h, is as follows:

```
struct sockaddr_in {
     short int         sin_family; /* Address family    */
     unsigned short    sin_port;    /* Port number      */
     struct in_addr    sin_addr;    /* Internet address */

     /* Pad to size of `struct sockaddr'. */
     /* Pad definition deleted */
};
```

The fields have the following meanings:

- sin_family: Family of protocols for this address. We will want the Internet family.
- sin_port: The port part of the address.
- sin_addr: The IP number part of the address.
- Pad: This will be explained in the next section.

After seeing the structure used for address formation, and having resolved the host name into an IP number, forming the address is done as follows:

```
char*               hostname;  /* name part of the address */
short               host_port; /* port part of the address */
struct hostent*     hen;       /* server's DNS entry */
```

```
struct sockaddr_in   sa;        /* address formation structure */

/* get information about the given host, using some the system's */
/* default name resolution mechanism (DNS, NIS, /etc/hosts...).  */
hen = gethostbyname(hostname_ser);
if (!hen) {
    perror("couldn't locate host entry");
}

/* create machine's Internet address structure  */
/* first clear out the struct, to avoid garbage */
memset(&sa, 0, sizeof(sa));

/* Using Internet address family */
sa.sin_family = AF_INET;
/* copy port number in network byte order */
sa.sin_port = htons(host_port);
/* copy IP address into address struct */
memcpy(&sa.sin_addr.s_addr, hen->h_addr_list[0], hen->h_length);
```

Notes:

- `perror()` is a function that prints an error message based on the global errno variable, and exits the process.
- `memset()` is a function used to set all bytes in a memory area of a specified size, to a specified value. On some (older) systems there is a different function that should be used instead named `bzero()`. Read your local manual page for additional information on it's usage. Note that `memcpy` is a part of the standard C library, so it should be used whenever possible in place of `bzero`.
- `memcpy()` is a function used to copy the contents of one memory area into another area. On some systems there is a different function with the same effect, named `bcopy()`. Like the `bzero` function, it should be avoided whenever `memcpy` is available.

---

4. The socket interface

We will now describe the interface used to write network applications in Unix systems (Especially Unix flavors derived from the BSD4.3 system). This interface is used throughout all kinds of networking software, but we will concentrate on the Internet protocol family.

We will first describe what a socket is, and how it relates to normal files, then explain what kinds of sockets exist on most Unix systems, how they are created using the socket() system call, and finally, how they are associated with a specific network connection, and how data is passed through them.

What is a socket?

A socket is formally defined as an endpoint for communication between an application program, and the underlying network protocols. This odd collection of words simply means that the program reads information from a socket in order to read from the network, writes information to it in order to write to the network, and sets sockets options in order to control protocol options. From the programmer's point of view, the socket is identical to the network. Just like a file descriptor is the endpoint of disk operations.

Types of sockets

In general, 3 types of sockets exist on most Unix systems: Stream sockets, Datagram sockets and Raw sockets.

Stream sockets are used for stream connections, i.e. connections that exist for a long duration. TCP connections use stream sockets.

Datagram sockets are used for short-term connections, that transfer a single packet across the network before terminating. the UDP protocol uses such sockets, due to its connection-less nature.

Raw sockets are used to access low-level protocols directly, bypassing the higher protocols. They are the means for a programmer to use the IP protocol, or the physical layer of the network, directly. Raw sockets can therefor be used to implement new protocols on top of the low-level protocols. Naturally, they are out of our scope.

Creating sockets

Creation of sockets is done using the `socket()` system call. This system call is defined as follows:

```
int socket(int address_family, int socket_type, int proto_family);
```

address_family defines the type of addresses we want this socket to use, and therefor defines what kind of network protocol the socket will use. We will concentrate on the Internet address family, cause we want to write Internet applications.

socket_type could be one of the socket types we mentioned earlier, or any other socket type that exists on your system. We choose the socket type according to the kind of interaction (and type or protocol) we want to use.

proto_family selects which protocol we want to socket to use. We will usually leave this value as 0 (or the constant `PF_UNSPEC` on some systems), and let the system choose the most suitable protocol for us. As for the protocol itself, In the Internet address family, a socket type of `SOCK_STREAM` will cause the protocol type to be set to TCP. A socket type of `SOCK_DGRAM` (Datagram socket) will cause the protocol type to be set to UDP.

The socket system call returns a file descriptor which will be used to reference the socket in later requests by the application program. If the call fails, however (due to lack of resources) the value returned will be negative (note that file descriptors have to be non-negative integers).

As an example, suppose that we want to write a TCP application. This application needs at least one socket in order to communicate across the Internet, so it will contain a call such as this:

```
int s;    /* descriptor of socket */

/* Internet address family, Stream socket */
s = socket(AF_INET, SOCK_STREAM, 0);
if (s < 0) {
    perror("socket: allocation failed");
}
```

Associating a socket with a connection

After a socket is created, it still needs to be told between which two end points it will communicate. It needs to be bound to a connection. There are two steps to this binding. The first is binding the socket to a local address. The second is binding it to a remote (foreign) address.

Binding to a local address could be done either explicitly, using the `bind()` system call, or implicitly, when a connecting is established. Binding to the remote address is done only when a connection is established. To bind a socket to a local address, we use the `bind()` system call, which is defined as follows:

```
int bind(int socket, struct sockaddr *address, int addrlen);
```

Note the usage of a different type of structure, namely `struct sockaddr`, then the one we used earlier (`struct sockaddr_in`). Why is the sudden change? This is due to the generality of the socket interface: sockets could be used as endpoints for connections using different types of address families. Each address family needs different information, so they use different structures to form their addresses. Therefore, a generic socket address type, struct sockaddr, is defined in the system, and for each address family, a different variation of this structure is used. For those who know, this means that `struct sockaddr_in`, for example, is an overlay of struct sockaddr (i.e. it uses the same memory space, just divides it differently into fields).

There are 4 possible variations of address binding that might be used when binding a socket in the Internet address family.

The first is binding the socket to a specific address, i.e. a specific IP number and a specific port. This is done when we know exactly where we want to receive messages. Actually this form is not used in simple servers, since usually these servers wish to accept connections to the machine, no matter which IP interface it came from.

The second form is binding the socket to a specific IP number, but letting the system choose an unused port number. This could be done when we don't need to use a well-known port.

The third form is binding the socket to a wild-card address called INADDR_ANY (by assigning it to the `sockaddr_in` variable), and to a specific port number. This is used in servers that are supposed to accept packets sent to this port on the local host, regardless of through which physical network interface the packet has arrived (remember that a host might have more then one IP address).

The last form is letting the system bind the socket to any local IP address and to pick a port number by itself. This is done by not using the `bind()` system call on the socket. The system will make the local bind when a connection through the socket is established, i.e. along with the remote address binding. This form of binding is usually used by clients, which care only about the remote address (where they connect to) and don't need any specific local port or local IP address. However, there are exceptions here too.

Sending and receiving data over a socket

After a connection is established (We will explain that when talking about Client and Server writing), There are several ways to send information over the socket. We will only describe one method for reading and one for writing. The others will be mentioned only in the "See Also" section.

The `read()` system call

The most common way of reading data from a socket is using the `read()` system call, which is defined like this:

```
int read(int socket, char *buffer, int buflen);
```

- socket - The socket from which we want to read.
- buffer - The buffer into which the system will write the data bytes.
- buflen - Size of the buffer, in bytes (actually, how much data we want to read).

The read system call returns one of the following values:

- 0 - The connection was closed by the remote host.
- -1 - The read system call was interrupted, or failed for some reason.

- n - The read system call put 'n' bytes into the buffer we supplied it with.

Note that `read()` might read less then the number of bytes we requested, due to unavailability of buffer space in the system.

The `write()` system call

The most common way of writing data to a socket is using the `write()` system call, which is defined like this:

```
int write(int socket, char *buffer, int buflen);
```

- socket - The socket into which we want to write.
- buffer - The buffer from which the system will read the data bytes.
- buflen - Size of the buffer, in bytes (actually, how much data we want to write).

The write system call returns one of the following values:

- 0 - The connection was closed by the remote host.
- -1 - The write system call was interrupted, or failed for some reason.
- n - The write system call wrote 'n' bytes into the socket.

Note that the system keeps internal buffers, and the write system call write data to those buffers, not necessarily directly to the network. thus, a successful `write()` doesn't mean the data arrived at the other end, or was even sent onto the network. Also, it could be that only some of the bytes were written, and not the actual number we requested. It is up to us to try to send the data again later on, when it's possible, and we'll show several methods for doing just that.

Closing a socket.

When we want to abort a connection, or to close a socket that is no longer needed, we can use the `close()` system call. it is defined simply as:

```
int close(int socket);
```

- socket - The socket that we wish to close. If it is associated with an open connection, the connection will be closed.

---

5. Writing Clients

This section describes how to write simple client applications, using the socket interface described earlier. As you remember (hmm, do you?) from the second section,a classic Client makes a connection to the server, and goes into a loop of reading commands from the user, parsing them, sending requests to the server, receiving responses from the server, parsing them and echoing them back at the user.

We will begin by showing the C code of a simple Client without user-interaction. This Client connects to the standard time server of a given host, reads the time, and prints it on the screen. Most (Unix) Internet hosts have a standard server called daytime, that awaits connections on the well-known port number 13, and when it receives a connection request, accepts it, writes the time to the Client, and closes the connection.

Lets see how the Client looks. Note the usage of a new system call, `connect()`, which is used to establish a connection to a remote machine, and will be further explained immediately following the program text.

```
#include <stdio.h>          /* Basic I/O routines          */
```

```c
#include <sys/types.h>     /* standard system types       */
#include <netinet/in.h>    /* Internet address structures */
#include <sys/socket.h>    /* socket interface functions  */
#include <netdb.h>         /* host to IP resolution       */

#define  HOSTNAMELEN   40   /* maximal host name length */
#define  BUFLEN        1024   /* maximum response size  */
#define  PORT          13   /* port of daytime server   */

int main(int argc, char *argv[])
{
    int                 rc;           /* system calls return value storage */
    int                 s;            /* socket descriptor                 */
    char                buf[BUFLEN+1]; /* buffer server answer             */
    char*               pc;           /* pointer into the buffer           */
    struct sockaddr_in  sa;           /* Internet address struct           */
    struct hostent*     hen;          /* host-to-IP translation            */

    /* check there are enough parameters */
    if (argc < 2) {
        fprintf(stderr, "Missing host name\n");
        exit (1);
    }

    /* Address resolution stage */
    hen = gethostbyname(argv[1]);
    if (!hen) {
        perror("couldn't resolve host name");
    }

    /* initiate machine's Internet address structure */
    /* first clear out the struct, to avoid garbage  */
    memset(&sa, 0, sizeof(sa));

    /* Using Internet address family */
    sa.sin_family = AF_INET;
    /* copy port number in network byte order */
    sa.sin_port = htons(PORT);
    /* copy IP address into address struct */
    memcpy(&sa.sin_addr.s_addr, hen->h_addr_list[0], hen->h_length);

    /* allocate a free socket                */
    /* Internet address family, Stream socket */
    s = socket(AF_INET, SOCK_STREAM, 0);
    if (s < 0) {
        perror("socket: allocation failed");
    }

    /* now connect to the remote server. the system will */
    /* use the 4th binding method (see section 3)        */
    /* note the cast to a struct sockaddr pointer of the */
    /* address of variable sa.                           */
    rc = connect(s, (struct sockaddr *)&sa, sizeof(sa));
```

```
    /* check there was no error */
    if (rc) {
        perror("connect");
    }

    /* now that we are connected, start reading the socket */
    /* till read() returns 0, meaning the server closed    */
    /* the connection.                                     */
    pc = buf;

    while (rc = read(s, pc, BUFLEN - (pc-buf))) {
        pc += rc;
    }

    /* close the socket */
    close(s);

    /* pad a null character to the end of the result */
    *pc = ' ';

    /* print the result */
    printf("Time: %s\n", buf);

     /* and terminate */
    return 0;
}
```

The complete source code for this client may be found in the daytime-client.c file.

The Client's code should be pretty easy to understand now. All we did was combine the features we have seen so far into one program. The only new feature introduced here is the `connect()` system call.

This system call is responsible to making the connection to the specified address of the remote machine, using the specified socket. Note that the address is being type-cast into the general address type, `struct sockaddr`, because this same system call is used to establish connections in various address families, not just the Internet address family. How will the system then know we want an Internet connection? The answer is given in the socket's information. If you remember, we specified this socket will be used in the Internet address family (`AF_INET`) when we created it.

Note also how the reading loop is performed. We are asking the system to read as much data as possible in the `read()` system call. However, the system might need several reads before it has consumed all the bytes sent by the server, that's why we used the while loop. Remember, never assume a `read()` system call will return the exact number of bytes you specified in the call. If less is available, the call will return quickly, and will not wait for the rest of the data. On the other hand, if no data is available, the call will block (not return) until data is available. Thus, when writing "Real" Clients and Servers, some measures have to be taken in order to avoid that blocking.

We will not discuss right now Clients that read user input. This subject will be differed until we learn how to read information efficiently from several input devices.

---

6. Single-Clients Servers

Now that we have seen how a Client is written, lets give it a different server to talk to. We will write the "hello world" (didn't you wait for this?) Server.

The "hello world" Server listens to a predefined port of our choice, and accepts incoming connections. It then writes the message "hello world" to the remote Client, and closes the connection. This will be done in an infinite loop, so we can serve a new Client after finishing with the current.

Note the introduction of two new system calls, `listen()` and `accept()`. The `listen()` system call asks the system to listen for new connections coming to our port. The `accept()` system call is used to accept (how obvious) such incoming connections. Both system calls will be explained further following the "hello world" Server's code.

```
#include <stdio.h>       /* Basic I/O routines          */
#include <sys/types.h>   /* standard system types       */
#include <netinet/in.h>  /* Internet address structures */
#include <sys/socket.h>  /* socket interface functions  */
#include <netdb.h>       /* host to IP resolution       */


#define  PORT   5050           /* port of "hello world" server */
#define  LINE   "hello world"  /* what to say to our clients   */

void main()
{
    int                 rc;       /* system calls return value storage  */
    int                 s;        /* socket descriptor                  */
    int                 cs;       /* new connection's socket descriptor */
    struct sockaddr_in  sa;       /* Internet address struct            */
    struct sockaddr_in  csa;      /* client's address struct            */
    int                 size_csa; /* size of client's address struct    */

    /* initiate machine's Internet address structure */
    /* first clear out the struct, to avoid garbage  */
    memset(&sa, 0, sizeof(sa));
    /* Using Internet address family */
    sa.sin_family = AF_INET;
    /* copy port number in network byte order */
    sa.sin_port = htons(PORT);
    /* we will accept connections coming through any IP */
    /* address that belongs to our host, using the      */
    /* INADDR_ANY wild-card.                            */
    sa.sin_addr.s_addr = INADDR_ANY;

    /* allocate a free socket */
    /* Internet address family, Stream socket */
    s = socket(AF_INET, SOCK_STREAM, 0);
    if (s < 0) {
        perror("socket: allocation failed");
    }

    /* bind the socket to the newly formed address */
```

```
    rc = bind(s, (struct sockaddr *)&sa, sizeof(sa));

    /* check there was no error */
    if (rc) {
        perror("bind");
    }

    /* ask the system to listen for incoming connections   */
    /* to the address we just bound. specify that up to    */
    /* 5 pending connection requests will be queued by the */
    /* system, if we are not directly awaiting them using  */
    /* the accept() system call, when they arrive.         */
    rc = listen(s, 5);

    /* check there was no error */
    if (rc) {
        perror("listen");
    }

    /* remember size for later usage */
    size_csa = sizeof(csa);

    /* enter an accept-write-close infinite loop */
    while (1) {
        /* the accept() system call will wait for a     */
        /* connection, and when one is established, a   */
        /* new socket will be created to handle it, and */
        /* the csa variable will hold the address       */
        /* of the Client that just connected to us.     */
        /* the old socket, s, will still be available   */
        /* for future accept() statements.              */
        cs = accept(s, (struct sockaddr *)&csa, &size_csa);

        /* check for errors. if any, enter accept mode again */
        if (cs < 0)
            continue;

        /* oak, we got a new connection. do the job... */
        write(cs, LINE, sizeof(LINE));

        /* now close the connection */
        close(cs);
    }
}
```

The complete source code for this server may be found in the hello-world-server.c file.

Look how little we had to add to the basic stuff in order to form our first server. The only two additions were the `listen()` and the `accept()` system calls. Lets examine them a little more.

If we want to serve incoming connections, we need to ask the system to listen on the specified port. If we don't do that, the remote Client will get a "connection refused" error. Once the system listens on the port, It could happen that more then one Client will ask for service simultaneously. We can tell the system how many Clients may "wait in line". This will be the second parameter to the `listen()` system call.

After issuing the `listen()` system call, we still need to actively accept incoming connections. This is done using the `accept()` system call. We tell it which socket is bound to the port we want to accept connection from, and give it the address of a variable in which the call will give us the address of the remote Client, once a connection is established. It will also update the size of the address, based on the address family used, in the variable whose address we pass as the third argument. We are not using the Client's address in our simple server, but other servers that might want to authenticate their Clients (or just to know where they are coming from), will use it.

Finally, the `accept()` system call returns a number of a new socket, which is allocated for the new established connection. This gives us a socket bound to the correct local and remote addresses, while not destroying the binding of the original socket, that we can later use to accept new connections.

---

7. Multi-Clients Servers

If single-Client Servers were a rather simple case, the multi-Client ones are a tougher nut. There are two main approaches to designing such servers.

The first approach is using one process that awaits new connections, and one more process (or thread) for each Client already connected. This approach makes design quite easy, cause then the main process does not need to differ between servers, and the sub-processes are each a single-Client server process, hence, easier to implement.

However, this approach wastes too many system resources (if child processes are used), and complicates inter-Client communication: If one Client wants to send a message to another through the server, this will require communication between two processes on the server, or locking mechanisms, if using multiple threads.

The second approach is using a single process for all tasks: waiting for new connections and accepting them, while handling open connections and messages that arrive through them. This approach uses less system resources, and simplifies inter-Client communication, although making the server process more complex.

Luckily, the Unix system provides a system call that makes these tasks much easier to handle: the `select()` system call.

The `select()` system call puts the process to sleep until any of a given list of file descriptors (including sockets) is ready for reading, writing or is in an exceptional condition. When one of these things happen, the call returns, and notifies the process which file descriptors are waiting for service.

The select system call is defined as follows:

```
int select(int numfds,
           fd_set *rfd,
           fd_set *wfd,
           fd_set *efd,
           struct timeval *timeout);
```

- numfds - highest number of file descriptor to check.
- rfd - set of file descriptors to check for reading availability.
- wfd - set of file descriptors to check for writing availability.
- efd - set of file descriptors to check for exceptional condition.
- timeout - how long to wait before terminating the call in case no file descriptor is ready.

`select()` returns the number of file descriptors that are ready, or -1 if some error occurred.

We give `select()` 3 sets of file descriptors to check upon. The sockets in the rfd set will be checked whether they sent data that can be read. The file descriptors in the wfd set will be checked to see whether we can write into any of them. The file descriptors in the efd set will be checked for exceptional conditions (you may safely ignore this set for now, since it requires a better understanding of the Internet protocols in order to be useful). Note that if we don't

want to check one of the sets, we send a `NULL` pointer instead.

We also give `select()` a timeout value - if this amount of time passes before any of the file descriptors is ready, the call will terminate, returning 0 (no file descriptors are ready).

NOTE - We could use the `select()` system call to modify the Client so it could also accept user input, Simply by telling it to `select()` on a set comprised of two descriptors: the standard input descriptor (descriptor number 0) and the communication socket (the one we allocated using the `socket()` system call). When the `select()` call returns, we will check which descriptor is ready: standard input, or our socket, and this way will know which of them needs service.

There are three more things we need to know in order to be able to use select. One - how do we know the highest number of a file descriptor a process may use on our system? Two - how do we prepare those sets? Three - when select returns, how do we know which descriptors are ready - and what they are ready for?

As for the first issue, we could use the `getdtablesize()` system call. It is defined as follows:

```
int getdtablesize();
```

This system call takes no arguments, and returns the number of the largest file descriptor a process may have. On modern systems, we could instead use the `getrlimit()` system call, using the `RLIMIT_NOFILE` parameter. Refer to the relevant manual page for more information.

As for the second issue, the system provides us with several macros to manipulate fd_set type variables.

`FD_ZERO(fd_set *xfd)`
: Clear out the set pointed to by 'xfd'.

`FD_SET(fd, fd_set *xfd)`
: Add file descriptor 'fd' to the set pointed to by 'xfd'.

`FD_CLR(fd, fd_set *xfd)`
: Remove file descriptor 'fd' from the set pointed to by 'xfd'.

`FD_ISSET(fd, fd_set *xfd)`
: check whether file descriptor 'fd' is part of the set pointed to by 'xfd'.

An important thing to note is that `select()` actually modifies the sets passed to it as parameters, to reflect the state of the file descriptors. This means we need to pass a copy of the original sets to `select()`, and manipulate the original sets according to the results of `select()`. In our example program, variable 'rfd' will contain the original set of sockets, and 'c_rfd' will contain the copy passed to `select()`.

Here is the source code of a Multi-Client echo Server. This Server accepts connection from several Clients simultaneously, and echoes back at each Client any byte it will send to the Server. This is a service similar to the one give by the Internet Echo service, that accepts incoming connections on the well-known port 7. Compare the code given here to the algorithm of a Multi-Client Server presented in the Client-Server model section.

```
#include <stdio.h>          /* Basic I/O routines          */
#include <sys/types.h>      /* standard system types       */
#include <netinet/in.h>     /* Internet address structures */
#include <sys/socket.h>     /* socket interface functions  */
#include <netdb.h>          /* host to IP resolution       */
#include <sys/time.h>       /* for timeout values          */
#include <unistd.h>         /* for table size calculations */

#define  PORT    5060            /* port of our echo server */
#define  BUFLEN  1024            /* buffer length           */
```

```
void main()
{
    int                 i;              /* index counter for loop operations  */
    int                 rc;             /* system calls return value storage  */
    int                 s;              /* socket descriptor                  */
    int                 cs;             /* new connection's socket descriptor */
    char                buf[BUFLEN+1]; /* buffer for incoming data           */
    struct sockaddr_in sa;              /* Internet address struct            */
    struct sockaddr_in csa;             /* client's address struct            */
    int                 size_csa;       /* size of client's address struct    */
    fd_set              rfd;            /* set of open sockets                */
    fd_set              c_rfd;          /* set of sockets waiting to be read  */
    int                 dsize;          /* size of file descriptors table     */

    /* initiate machine's Internet address structure */
    /* first clear out the struct, to avoid garbage  */
    memset(&sa, 0, sizeof(sa));
    /* Using Internet address family */
    sa.sin_family = AF_INET;
    /* copy port number in network byte order */
    sa.sin_port = htons(PORT);
    /* we will accept connections coming through any IP */
    /* address that belongs to our host, using the      */
    /* INADDR_ANY wild-card.                            */
    sa.sin_addr.s_addr = INADDR_ANY;

    /* allocate a free socket */
    /* Internet address family, Stream socket */
    s = socket(AF_INET, SOCK_STREAM, 0);
    if (s < 0) {
        perror("socket: allocation failed");
    }

    /* bind the socket to the newly formed address */
    rc = bind(s, (struct sockaddr *)&sa, sizeof(sa));

    /* check there was no error */
    if (rc) {
        perror("bind");
    }

    /* ask the system to listen for incoming connections   */
    /* to the address we just bound. specify that up to    */
    /* 5 pending connection requests will be queued by the */
    /* system, if we are not directly awaiting them using  */
    /* the accept() system call, when they arrive.         */
    rc = listen(s, 5);

    /* check there was no error */
    if (rc) {
        perror("listen");
    }
```

```
/* remember size for later usage */
size_csa = sizeof(csa);

/* calculate size of file descriptors table */
dsize = getdtablesize();

/* close all file descriptors, except our communication socket */
/* this is done to avoid blocking on tty operations and such.  */
for (i=0; i<dsize; i++)
    if (i != s)
        close(i);

/* we initially have only one socket open,   */
/* to receive new incoming connections.      */
FD_ZERO(&rfd);
FD_SET(s, &rfd);
/* enter an accept-write-close infinite loop */
while (1) {
    /* the select() system call waits until any of  */
    /* the file descriptors specified in the read,  */
    /* write and exception sets given to it, is     */
    /* ready to give data, send data, or is in an   */
    /* exceptional state, in respect. the call will */
    /* wait for a given time before returning. in   */
    /* this case, the value is NULL, so it will     */
    /* not timeout. dsize specifies the size of the */
    /* file descriptor table.                       */
    c_rfd = rfd;
    rc = select(dsize, &c_rfd, NULL, NULL, NULL);

    /* if the 's' socket is ready for reading, it   */
    /* means that a new connection request arrived. */
    if (FD_ISSET(s, &c_rfd)) {
        /* accept the incoming connection */
        cs = accept(s, (struct sockaddr *)&csa, &size_csa);

        /* check for errors. if any, ignore new connection */
        if (cs < 0)
            continue;

        /* add the new socket to the set of open sockets */
        FD_SET(cs, &rfd);

        /* and loop again */
        continue;
    }

    /* check which sockets are ready for reading, */
    /* and handle them with care.                 */
    for (i=0; i<dsize; i++) {
        if (i != s && FD_ISSET(i, &c_rfd)) {
            /* read from the socket */
            rc = read(i, buf, BUFLEN);
```

```
                /* if client closed the connection... */
                if (rc == 0) {
                    /* close the socket */
                    close(i);
                    FD_CLR(i, &rfd);
                }
                /* if there was data to read */
                else {
                    /* echo it back to the client        */
                    /* NOTE: we SHOULD have checked that */
                    /* indeed all data was written...    */
                    write(i, buf, rc);
                }
            }
        }
    }
}
```

The complete source code for this server may be found in the multi-client-echo-server.c file.

---

8. Conclusions

That's it. If you got as far as here, and hopefully tried playing a little with the code examples, you probably got the basic notion of what it takes to write simple clients and servers that communicate using the TCP protocol. We didn't cover UDP-based clients and servers, but hopefully, you'll manage getting there by referring to one of the more advanced resources mentioned below.

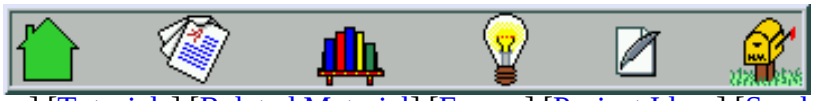
And remember: clients, and especially servers, are expected to be robust creatures. Yet, the network is a too shaky ground to assume everything will work smoothly. *Expect the unexpected*. Check the return value of any system call you use, and act upon it. If a system call failed, try to figure out why it failed (using the returned error code and possibly the `errno` variable), and if you cannot write code to bypass that kind of failure - at least give your users an error message they can understand.

---

9. See Also - or, where to go from here

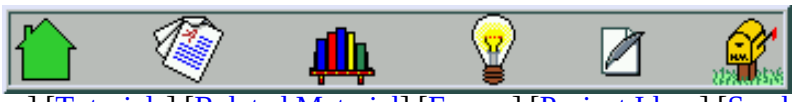
The following references might be good places to continue exploring socket programming:

- Internetworking with TCP/IP - Volume I, by Douglas Comer
  This fine book explains in great details the internals of the TCP/IP protocol suite: what protocols comprise the TCP/IP protocol family; how each protocol performs its chores and how its packets are ordered. It also explains about local area networks (LANs), such as Ethernet and Token-Ring, as well as the history of the Internet.
- Unix Network Programming - By W. Richard Stevens
  The "bible" for this subject. Teaches how to write programs that communicate, Not only via sockets and TCP/IP, but also using other types of IPC (Inter-Process Communications). A "Must Read" for every serious network programmer. If you're interested in simply writing network programs, without delving into too much theory, this should be your next stop.



```
/*
 * authenticate-user.c - ask a user for their password, and compare it to
 *                       the password, as taken from the "/etc/passwd" file.
 */

#include <unistd.h>   /* crypt(), etc.          */
#include <pwd.h>      /* getpass(), getpwnam(). */
#include <string.h>   /* strcmp(), etc.         */

void
main()
{
    /* buffers for reading in the user name and the password. */
    char user[21];
    char* password;
    /* storing the encrypted password, and the salt. */
    char* encrypted_password;
    char salt[2];
    /* user's "/etc/passwd" entry. */
    struct passwd* user_info;

    /* prompt the user for a user name. */
    printf("User name: ");
    fflush(stdout); /* flush the prompt to make sure the user sees it. */
    fgets(user, 20, stdin);
    /* fgets() stores also the new-line that the user typed in. so we */
    /* need to locate the new-line character, and truncate it.        */
    if (strchr(user, '\n'))
        (*(strchr(user, '\n'))) = '\0';

    /* prompt the user for their password. the getpass() function  */
    /* prints the given prompt, turns off echo (so the password    */
    /* typed won't be seen on screen), and returns the string that */
    /* the user types.                                             */
    password = getpass("Password: ");

    /* find the user's encrypted password, as stored in "/etc/passwd". */
    user_info = getpwnam(user);
    if (!user_info) {
        printf("login incorrect.\n");
        exit(1);
    }

    /* take the salt as stored in the password field of the user. */
    strncpy(salt, user_info->pw_passwd, 2);

    /* encrypt the given password using the found "salt". */
    encrypted_password = crypt(password, salt);

    /* compare the results of crypt, with the user's stored password field. */
    if (strcmp(user_info->pw_passwd, encrypted_password) != 0) {
        printf("login incorrect.\n");
        exit(1);
    }

    /* authentication succeeded... */
    printf("login successful.\n");
}
```

```
/*
 * show-loged-users.c - list the names of the currently logged in users.
 */

#include <stdio.h>     /* printf(), etc.    */
#include <utmp.h>      /* struct utmp, etc. */
#include <string.h>    /* strncpy(), etc.   */
#include <unistd.h>    /* open(), etc.      */
#include <fcntl.h>     /* O_RDONLY, etc.    */

/* full path for the UTMP file on our sample system. */
#define UTMP_PATH "/var/run/utmp"

void
main()
{
    /* buffer to read one utmp record. */
    struct utmp utmp_entry;
    /* buffer to store a user name. */
    char user_name[UT_NAMESIZE+1];
    /* file descriptor for the "utmp" file. */
    int fd;

    /* open the utmp file for reading. */
    fd = open(UTMP_PATH, O_RDONLY);
    if (fd < 0) {
        perror("open");
        exit(1);
    }

    printf("Currently logged-in users:\n");
    /* start scanning the file, entry by entry. */
    while (1) {
        int rc = read(fd, &utmp_entry, sizeof(utmp_entry));
        if (rc < 0) {
            perror("read");
            exit(1);
        }
        if (rc == 0) /* end of file - exit the reading loop. */
            break;

        /* This is Linux specific - only records whose ut_type field is    */
        /* USER_PROCESS, represent logged in users. the rest are temporary */
        /* records of various types.                                       */
        if (utmp_entry.ut_type != USER_PROCESS)
            continue;

        /* copy the user name field to our user name variable. */
        strncpy(user_name, utmp_entry.ut_name, UT_NAMESIZE);
        /* make sure this string is terminated with a null character. */
        user_name[UT_NAMESIZE] = '\0';

        printf("'%s',", user_name);
    }

    printf("\n");
    close(fd);
}
```



Basic Graphics Programming With The Xlib Library - Part II

Windows Hierarchy

When windows are displayed on an X server, they are always ordered in some hierarchy - every window may have child windows, each child window may have its own child windows, etc. Lets see a few properties of this hierarchy, and how they effect the operations like drawing or events propagation.

Root, Parent And Child Windows

On each screen, there is a root window. The root window always spans the entire screen size. This window cannot be destroyed, resized or iconified. When an application creates windows, it first has to create at least one top-level window. This window becomes a direct descendant of the root window, until it is first mapped on the screen. Before this window is mapped, the window manager is notified of the operation about to take place. The window manager then has the privilege of re-parenting the new top-level window. This is used to add a window that will contain the new window, and will be used to draw its frame, title bar, system menu, etc.

Once such a top-level window (which is actually not a top-level window after the re-parenting operation takes place) is created, the application can create child windows inside this window. A child can only be displayed inside its parent window - If it is moved outside, its being clipped by the border of its parent window. Any window may contain more then one child window, and if this is the case, these windows are being ordered in an internal stack. When a top-level window is being raised - all its descendant windows are being raised with it, while retaining their internal ordering. If a child window is raised - it is being raised only amongst its sibling windows.

Lets see how to create a child window inside a given window 'win'.

```
/* this variable will store the handle of the newly created child window. */
Window child_win;

/* these variables will store the window's width and height. */
int child_win_width = 50;
int child_win_height = 30;

/* these variables will store the window's location.         */
/* position of the child window is top-left corner of the    */
/* parent window. - 0,0.                                     */
int child_win_x = 0;
int child_win_y = 0;

/* create the window, as specified earlier. */
child_win = XCreateSimpleWindow(display,
                                win,
                                child_win_x, child_win_y,
```

```
                                child_win_width, child_win_height,
                                child_win_border_width,
                                BlackPixel(display, screen_num),
                                WhitePixel(display, screen_num));
```

Events Propagation

We have discussed events propagation earlier - If a window is being sent an event and it does not process the event - the event is being passed to the this window's parent window. If that parent window does not handle this event - it is being passed to the parent's parent window, and so on. This behavior does not make a lot of sense for a simple Xlib application, but it does make sense when higher-level graphic libraries are used. They usually associate functions with events occurring in specific windows. In such a case, it is useful to pass the event to the relevant window, with which a proper function is associated.

Interacting With The Window Manager

After we have seen how to create windows and draw on them, we take one step back, and look at how our windows are interacting with their environment - the full screen, and the other windows. First of all, our application needs to interact with the window manager. The window manager is responsible to decorating drawn windows (i.e. adding a frame, an iconify button, a system menu, a title bar), as well as to handling icons shown when windows are being iconified. It also handles ordering of windows on the screen, and other administrative tasks. We need to give it various hints as to how we want it to treat our application's windows.

Window Properties

Many of the parameters communicated to the window manager are passed using data called "properties". These properties are attached by the X server to different windows, and are stored in a format that makes it possible to read them from different machines, that may use different architectures (remember that an X client program may run on a remote machine). Properties may be of various types - numbers, strings, etc. Most window manager hints functions use text properties. A function named `XStringListToTextProperty()` may be used to turn a normal C string into an X text property, that can later be passed to various Xlib functions. Here is how to use it:

```
/* This variable will store the newly created property. */
XTextProperty window_title_property;

/* This is the string to be translated into a property. */
char* window_title = "hello, world";

/* translate the given string into an X property. */
int rc = XStringListToTextProperty(&window_title,
                                   1,
                                   &window_title_property);
/* check the success of the translation. */
if (rc == 0) {
    fprintf(stderr, "XStringListToTextProperty - out of memory\n");
    exit(1);
}
```

---

the `XStringListToTextProperty()` function accepts an array of C strings, a count of the number of strings in the array (1 in our case), and a pointer to an `XTextProperty` variable. It concatenates the list of strings and places it in the `XTextProperty` variable. It returns a non-zero value on success, or 0 on failure (e.g. not enough memory to perform the operation).

Setting The Window Name And Icon Name

The first thing would be to set the name for our window. This is done using the `XSetWMName()` function. This name may be used by the window manager as the title of the window (in the title bar), in a task list, etc. This function accepts 3 parameters: a pointer to the display, a window handle, and an `XTextProperty` containing the desired title. Here is how it is used:

```
/* assume that window_title_property is our XTextProperty, and is */
/* defined to contain the desired window title.                   */
XSetWMName(display, win, &window_title_property);
```

in order to set the name of the iconized version of our window, we will use the `XSetWMIconName()` function in a similar way:

```
/* this time we assume that icon_name_property is an initialized */
/* XTextProperty variable containing the desired icon name.      */
XSetWMIconName(display, win, &icon_name_property);
```

Setting Preferred Window Size(s)

In various cases, we wish to let the window manager know that we want to have a given size for our window, and to only let the user resize our window in given quantities. For example, in a terminal application (like xterm), we want our window to always contain complete rows and columns, so the text we display won't be cut off in the middle. In other cases we do not want our window to be resized at all (like in many dialog boxes), etc. We can relay this information to the window manager, although it may simply ignore our request. We need to first create a data structure to hold the information, fill it with proper data, and use the `XSetWMNormalHints()` function. Here is how this may be done:

```
/* pointer to the size hints structure. */
XSizeHints* win_size_hints = XAllocSizeHints();
if (!win_size_hints) {
    fprintf(stderr, "XAllocSizeHints - out of memory\n");
    exit(1);
}

/* initialize the structure appropriately. */
/* first, specify which size hints we want to fill in. */
/* in our case - setting the minimal size as well as the initial size. */
win_size_hints->flags = PSize | PMinSize;
/* next, specify the desired limits.                                   */
/* in our case - make the window's size at least 300x200 pixels. */
/* and make its initial size 400x250.                                  */
win_size_hints->min_width = 300;
win_size_hints->min_height = 200;
win_size_hints->base_width = 400;
win_size_hints->base_height = 250;

/* pass the size hints to the window manager. */
XSetWMNormalHints(display, win, win_size_hints);

/* finally, we can free the size hints structure. */
XFree(win_size_hints);
```

For full info about the other size hints we may supply, see your manual pages.

Setting Miscellaneous Window Manager Hints

There are some other window manager hints that may be set using the `XSetWMHints()` function. This function uses a `XWMHints` structure to pass the data to the window manager. Here is how it may be used:

```
/* pointer to the WM hints structure. */
XWMHints* win_hints = XAllocWMHints();
if (!win_hints) {
    fprintf(stderr, "XAllocWMHints - out of memory\n");
    exit(1);
}

/* initialize the structure appropriately. */
/* first, specify which hints we want to fill in. */
/* in our case - setting the state hint as well as the icon position hint. */
win_hints->flags = StateHint | IconPositionHint;
/* next, specify the desired hints data.                          */
/* in our case - make the window's initial state be iconized,     */
/* and set the icon position to the top-left part of the screen. */
win_hints->initial_state = IconicState;
win_hints->icon_x = 0;
win_hints->icon_y = 0;

/* pass the hints to the window manager. */
XSetWMHints(display, win, win_hints);

/* finally, we can free the WM hints structure. */
XFree(win_hints);
```

Other options settable using this function are specified in the manual page.

Setting An Application's Icon

In order to set an icon to be used by the window manager, when a user iconifies our application, we use the `XSetWMHints()` function mentioned above. However, we first need to create a pixmap that contains the icon's data. Pixmaps are the way that X servers manipulate images, and will be explained fully later in this tutorial. For now, we'll just show you how to set the icon for your application. We assume you got a bitmap file representing this icon, stored using an X bitmap format. An example of such a file is the "icon.bmp" file that comes with this tutorial. Anyway, here is the code:

```
/* include the definition of the bitmap in our program. */
#include "icon.bmp";

/* pointer to the WM hints structure. */
XWMHints* win_hints;

/* load the given bitmap data and create an X pixmap containing it. */
Pixmap icon_pixmap = XCreateBitmapFromData(display,
                                           win,
                                           icon_bitmap_bits,
                                           icon_bitmap_width,
                                           icon_bitmap_height);
```

```
if (!icon_pixmap) {
    fprintf(stderr, "XCreateBitmapFromData - error creating pixmap\n");
    exit(1);
}

/* allocate a WM hints structure. */
win_hints = XAllocWMHints();
if (!win_hints) {
    fprintf(stderr, "XAllocWMHints - out of memory\n");
    exit(1);
}

/* initialize the structure appropriately. */
/* first, specify which size hints we want to fill in. */
/* in our case - setting the icon's pixmap. */
win_hints->flags = IconPixmapHint;
/* next, specify the desired hints data.           */
/* in our case - supply the icon's desired pixmap. */
win_hints->icon_pixmap = icon_pixmap;

/* pass the hints to the window manager. */
XSetWMHints(display, win, win_hints);

/* finally, we can free the WM hints structure. */
XFree(win_hints);
```

---

you may use programs such as "xpaint" to create files using the X bitmap format.

To conclude this section, we supply a simple-wm-hints.c, that creates a window, sets the window manager hints shown above, and runs into a very simple event loop, allowing the user to play with the window and see how these hints affect the behavior of the application. Try playing with the various hints, and try to run the program under different window managers, to see the differences in its behavior under each of them. This will teach you something about X programs portability.

Simple Window Operations

One more thing we can do to our windows is manipulate them on the screen - resize them, move them, raise or lower them, iconify them and so on. A set of window operations functions are supplied by Xlib for this purpose.

Mapping And UN-mapping A Window

The first pair of operations we can apply on a window is mapping it, or un-mapping it. Mapping a window causes the window to appear on the screen, as we have seen in our simple window program example. UN-mapping it causes it to be removed from the screen (although the window as a logical entity still exists). This gives the effect of making a window hidden (unmapped) and shown again (mapped). For example, if we have a dialog box window in our program, instead of creating it every time the user asks to open it, we can create the window once, in an unmapped mode, and when the user asks to open it, we simply map the window on the screen. When the user clicked the 'OK' or 'Cancel' button, we simply UN-map the window. This is much faster then creating and destroying the window, however, the cost is wasted resources, both on the client side, and on the X server side.

You will remember That the map operation can be done using the `XMapWindow()` operation. The UN-mapping operation can be done using the `XUnmapWindow()` operation. Here is how to apply them:

```
/* make the window actually appear on the screen. */
```

```
XMapWindow(display, win);

/* make the window hidden. */
XUnmapWindow(display, win);
```

The mapping operation will cause an `Expose` event to be sent to our application, unless the window is completely covered by other windows.

Moving A Window Around The Screen

Another operation we might want to do with windows is to move them to a different location. This can be done using the `XMoveWindow()` function. It will accept the new coordinates of the window, in the same fashion that `XCreateSimpleWindow()` got them when the window was created. The function is invoked like this:

```
/* move the window to coordinates x=400 and y=100. */
XMoveWindow(display, win, 400, 100);
```

Note that when the window is moved, it might get partially exposed or partially hidden by other windows, and thus we might get `Expose` events due to this operation.

Resizing A Window

Yet another operation we can do is to change the size of a window. This is done using the `XResizeWindow()` function:

```
/* resize the window to width=200 and height=300 pixels. */
XResizeWindow(display, win, 200, 300);
```

We can also combine the move and resize operations using the single `XMoveResizeWindow()` function:

```
/* move the window to location x=20 y=30, and change its size */
/* to width=100 and height=150 pixels.                        */
XMoveResizeWindow(display, win, 20, 30, 100, 150);
```

Changing Windows Stacking Order - Raise And Lower

Until now we changed properties of a single window. We'll see that there are properties that relate to the window and other windows. One of them is the stacking order. That is, the order in which the windows are layered on top of each other. the front-most window is said to be on the top of the stack, while the back-most window is at the bottom of the stack. Here is how we manipulate our windows stacking order:

```
/* move the given window to the top of the stack. */
XRaiseWindow(display, win1);

/* move the given window to the bottom of the stack. */
XLowerWindow(display, win1);
```

Iconifying And De-Iconifying A Window

One last operation we will show here is the ability to change a window into an iconified mode and vice versa. This is done using the `XIconifyWindow()` function - to iconify the window, and the `XMapWindow()` to de-iconify it. To understand why there is no inverse function for `XIconifyWindow()`, we must realize that when a window is iconified, what actually happens is that the window is being unmapped, and instead its icon window is being mapped. Thus, to make the original window reappear, we simply need to map it again. The icon is indeed another window that is simply related strongly to our normal window - it is not a different state of our window. Here is how to iconify a

window and then de-iconify it:

```
/* iconify our window. Make its icon window appear on the same */
/* screen as our window (assuming we created our window on the */
/* default screen).                                            */
XIconifyWindow(display, win, DefaultScreen(display));

/* de-iconify our window. the icon window will be automatically */
/* unmapped by this operation.                                  */
XMapWindow(display, win);
```

Getting Info About A Window

Just like we can set various attributes of our windows, we can also ask the X server supply the current values of these attributes. For example, we can check where a window is located on screen, what is its current size, whether it is mapped or not, etc. The `XGetWindowAttributes()` function may be used to get this information. Here is how it is used:

```
/* this variable will contain the attributes of the window. */
XWindowAttributes win_attr;

/* query the window's attributes. */
Status rc = XGetWindowAttributes(display, win, &win_attr);
```

The `XWindowAttributes` structure contains many fields - here are some of them:

`int x, y;`
: Location of the window, relative to its parent window.

`int width, height;`
: width and height of the window (in pixels).

`int border_width`
: width of the window's border.

`Window root;`
: handle of the root widow of the screen on which our window is displayed.

One problem with this function, is that it returns the location of the window relative to its parent window. This makes these coordinates rather useless for any window manipulation functions (e.g. XMoveWindow). In order to overcome this problem, we need to take a two-step operation. First, we find out the ID of the parent window of our window. We then translate the above relative coordinates to screen coordinates. We use two new functions for this calculation, namely `XQueryTree()` and `XTranslateCoordinates()`. These two functions do more then we need, so we will concentrate on the relevant information only.

```
/* these variables will eventually hold the translated coordinates. */
int screen_x, screen_y;
/* this variable is here simply because it's needed by the          */
/* XTranslateCoordinates function below. For its purpose, see the   */
/* manual page.                                                     */
Window child_win;

/* this variable will store the ID of the parent window of our window. */
Window parent_win;
/* this variable will store the ID of the root window of the screen    */
/* our window is mapped on.                                            */
```

```
Window root_win;
/* this variable will store an array of IDs of the child windows of    */
/* our window.                                                         */
Window* child_windows;
/* and this one will store the number of child windows our window has. */
int num_child_windows;

/* finally, make the query for the above values. */
XQueryTree(display, win,
           &root_win,
           &parent_win,
           &child_windows, &num_child_windows);

/* we need to free the list of child IDs, as it was dynamically allocated */
/* by the XQueryTree function.                                            */
XFree(child_windows);

/* next, we make the coordinates translation, from the coordinates system */
/* of the parent window, to the coordinates system of the root window,    */
/* which happens to be the same as that of the screen, since the root     */
/* window always spans the entire screen size.                            */
/* the 'x' and 'y' values are those previously returned by the            */
/* XGetWindowAttributes function.                                         */
XTranslateCoordinates(display, parent_win, root_win,
                      win_attr.x, win_attr.y, &screen_x, &screen_y,
                      &child_win);

/* at this point, screen_x and screen_y contain the location of our original */
/* window, using screen coordinates.                                         */
```

As you can see, Xlib sometimes makes us work hard for things that could have been much easier, if its interfaces and functions were a little more consistent.

As an example of how these operations all work, check out our window-operations.c program.

Using Colors To Paint The Rainbow

Up until now, all our painting operations were done using black and white. We will (finally) see now how to draw using colors.

Color Maps

In the beginning, there were not enough colors. Screen controllers could only support a limited number of colors simultaneously (16 initially, and then 256). Because of this, an application could not just ask to draw in a "light purple-red" color, and expect that color to be available. Each application allocated the colors it needed, and when all 16 or 256 color entries were in use, the next color allocation would fail.

Thus, the notion of "a color map" was introduced. A color map is a table whose size is the same as the number of simultaneous colors a given screen controller. Each entry contained the RGB (Red, Green and Blue) values of a different color (all colors can be drawn using some combination of red, green and blue). When an application wants to draw on the screen, it does not specify which color to use. Rather, it specifies which color entry of some color map to be used during this drawing. Change the value in this color map entry - and the drawing will use a different color.

In order to be able to draw using colors that got something to do with what the programmer intended, color map

allocation functions were supplied. You could ask to allocate a color map entry for a color with a set of RGB values. If one already existed, you'd get its index in the table. If none existed, and the table was not full, a new cell would be allocated to contain the given RGB values, and its index returned. If the table was full, the procedure would fail. You could then ask to get a color map entry with a color that is closest to the one you were asking for. This would mean that the actual drawing on the screen would be done using colors similar to what you wanted, but not the same.

On today's more modern screens, where one runs an X server with support for 1 million colors, this limitation looks a little silly, but remember that there are still older computers with older graphics cards out there. Using color maps, support for these screens becomes transparent to you. On a display supporting 1 million colors, any color entry allocation request would succeed. On a display supporting a limited number of colors, some color allocation requests would return similar colors. It won't look as good, but your application would still work.

Allocating And Freeing Color Maps

When you draw using Xlib, you can choose to use the standard color map of the screen your window is displayed on, or you can allocate a new color map and apply it to a window. In the latter case, each time the mouse moves onto your window, the screen color map will be replaced by your window's color map, and you'll see all the other windows on screen change their colors into something quite bizzar. In fact, this is the effect you get with X applications that use the "-install" command-line option.

In order to access the screen's standard color map, the `DefaultColormap` macro is defined:

```
Colormap screen_colormap = DefaultColormap(display, DefaultScreen(display));
```

this will return a handle for the color map used by default on the first screen (again, remember that an X server may support several different screens, each of which might have its own resources).

The other option, that of allocating a new color map, works as follows:

```
/* first, find the default visual for our screen. */
Visual* default_visual = DefaultVisual(display, DefaultScreen(display));
/* this creates a new color map. the number of color entries in this map */
/* is determined by the number of colors supported on the given screen.  */
Colormap my_colormap = XCreateColormap(display,
                                       win,
                                       default_visual,
                                       AllocNone);
```

Note that the window parameter is only used to allow the X server to create the color map for the given screen. We can then use this color map for any window drawn on the same screen.

Allocating And Freeing A Color Entry

Once we got access to some color map, we can start allocating colors. This is done using the `XAllocNamedColor()` and `XAllocColor()` functions. The first `XAllocNamedColor()` accepts a color name (e.g. "red", "blue", "brown" and so on) and allocates the the closest color that can be actually drawn on the screen. The `XAllocColor()` accepts an RGB color, and allocates the closest color that can be drawn on the screen. Both functions use the `XColor` structure, that has the following relevant fields:

`unsigned long pixel`
: This is the index of the color map entry that can be used to draw in this color.

`unsigned short red`
: the red part of the RGB value of the color.

`unsigned short green`
: the green part of the RGB value of the color.

`unsigned short blue`

the blue part of the RGB value of the color.

Here is an example of using these functions:

```
/* this structure will store the color data actually allocated for us. */
XColor system_color_1, system_color_2;
/* this structure will store the exact RGB values of the named color.  */
/* it might be different from what was actually allocated.             */
XColor exact_color;

/* allocate a "red" color map entry. */
Status rc = XAllocNamedColor(display,
                             screen_colormap,
                             "red",
                             &system_color_1,
                             &exact_color);
/* make sure the allocation succeeded. */
if (rc == 0) {
    fprintf(stderr,
            "XAllocNamedColor - allocation of 'red' color failed.\n");
}
else {
    printf("Color entry for 'red' - allocated as (%d,%d,%d) in RGB values.\n",
           system_color_1.red, system_color_1.green, system_color_1.blue);
}

/* allocate a color with values (30000, 10000, 0) in RGB. */
system_color_2.red = 30000;
system_color_2.green = 10000;
system_color_2.blue = 0;
Status rc = XAllocColor(display,
                        screen_colormap,
                        &system_color_2);
/* make sure the allocation succeeded. */
if (rc == 0) {
    fprintf(stderr,
            "XAllocColor - allocation of (30000,10000,0) color failed.\n");
}
else {
    /* do something with the allocated color... */
    .
    .
}
```

Drawing With A Color

After we have allocated the desired colors, we can use them when drawing text or graphics. To do that, we need to set these colors as the foreground and background colors of some GC (Graphics Context), and then use this GC to make the drawing. This is done using the `XSetForeground()` and `XSetBackground()` functions, as follows:

```
/* use the previously defined colors as the foreground and background  */
/* colors for drawing using the given GC. assume my_gc is a handle to */
/* a previously allocated GC.                                         */
```

```
XSetForeground(display, my_gc, screen_color_1.pixel);
XSetForeground(display, my_gc, screen_color_2.pixel);
```

As you see, this is rather simple. The actual drawing is done using the same functions we have seen earlier. Note that in order to draw using many different colors, we can do one of two things. We can either change the foreground and/or background colors of the GC before any drawing function, or we can use several different GCs to draw in different colors. The decision as of which option to use is yours. Note that allocating many GCs will use more resources of the X server, but this will sometime lead to more compact code, and will might it easier to replace the drawn colors.

As an example to drawing using colors, look at the color-drawing.c program. This is a copy of the simple-drawing.c program, except that here we also allocate colors and use them to paint the rainbow...

X Bitmaps And Pixmaps

One thing many so-called "Multi-Media" applications need to do, is display images. In the X world, this is done using bitmaps and pixmaps. We have already seen some usage of them when setting an icon for our application. Lets study them further, and see how to draw these images inside a window, along side the simple graphics and text we have seen so far.

One thing to note before delving further, is that Xlib supplies no means of manipulating popular image formats, such as gif, jpeg or tiff. It is left up to the programmer (or to higher level graphics libraries) to translate these image formats into formats that the X server is familiar with - x bitmaps, and x pixmaps.

What Is An X Bitmap? An X Pixmap?

An X bitmap is a two-color image stored in a format specific to the X window system. When stored in a file, the bitmap data looks like a C source file. It contains variables defining the width and height of the bitmap, an array containing the bit values of the bitmap (the size of the array = width * height), and an optional hot-spot location (will be explained later, when discussing mouse cursors).

A X pixmap is a format used to store images in the memory of an X server. This format can store both black and white images (such as x bitmaps) as well as color images. It is the only image format supported by the X protocol, and any image to be drawn on screen, should be first translated into this format.

In actuality, an X pixmap can be thought of as a window that does not appear on the screen. Many graphics operations that work on windows, will also work on pixmaps - just supply the pixmap ID instead of a window ID. In fact, if you check the manual pages, you will see that all these functions accept a 'Drawable', not a 'Window'. since both windows and pixmaps are drawables, they both can be used to "draw on" using functions such as `XDrawArc()`, `XDrawText()`, etc.

Loading A Bitmap From A File

We have already seen how to load a bitmap from a file to memory, when we demonstrated setting an icon for an application. The method we showed earlier required the inclusion of the bitmap file in our program, using the C pre-processor '#include' directive. We will see here how we can access the file directly.

```
/* this variable will contain the ID of the newly created pixmap.    */
Pixmap bitmap;

/* these variables will contain the dimensions of the loaded bitmap. */
unsigned int bitmap_width, bitmap_height;

/* these variables will contain the location of the hot-spot of the   */
/* loaded bitmap.                                                     */
```

```
int hotspot_x, hotspot_y;

/* this variable will contain the ID of the root window of the screen */
/* for which we want the pixmap to be created.                        */
Window root_win = DefaultRootWindow(display);

/* load the bitmap found in the file "icon.bmp", create a pixmap      */
/* containing its data in the server, and put its ID in the 'bitmap'  */
/* variable.                                                          */
int rc = XReadBitmapFile(display, root_win,
                         "icon.bmp",
                         &bitmap_width, &bitmap_height,
                         &bitmap,
                         &hotspot_x, &hotspot_y);
/* check for failure or success. */
switch (rc) {
    case BitmapOpenFailed:
        fprintf(stderr, "XReadBitmapFile - could not open file 'icon.bmp'.\n");
        break;
    case BitmapFileInvalid:
        fprintf(stderr,
                "XReadBitmapFile - file '%s' doesn't contain a valid bitmap.\n",
                "icon.bmp");
        break;
    case BitmapNoMemory:
        fprintf(stderr, "XReadBitmapFile - not enough memory.\n");
        break;
    case BitmapSuccess:
        /* bitmap loaded successfully - do something with it... */
        .
        .
        break;
}
```

Note that the 'root_win' parameter has nothing to do with the given bitmap - the bitmap is not associated with this window. This window handle is used just to specify the screen that we want the pixmap to be created for. This is important, as the pixmap must support the same number of colors as the screen does, in order to make it useful.

Drawing A Bitmap In A Window

Once we got a handle to the pixmap generated from a bitmap, we can draw it on some window, using the `XCopyPlane()` function. This function allows us to specify what drawable (a window, or even another pixmap) to draw the given pixmap onto, and at what location in that drawable.

```
/* draw the previously loaded bitmap on the given window, at location   */
/* 'x=100, y=50' in that window. we want to copy the whole bitmap, so   */
/* we specify location 'x=0, y=0' of the bitmap to start the copy from, */
/* and the full size of the bitmap, to specify how much of it to copy.  */
XCopyPlane(display, bitmap, win, gc,
           0, 0,
           bitmap_width, bitmap_height,
           100, 50,
           1);
```

As you can see, we could also copy a given rectangle of the pixmap, instead of the whole pixmap. Also note the last parameter to the `XCopyPlane` function (the '1' at the end). This parameter specifies which plane of the source image

we want to copy to the target window. For bitmaps, we always copy plane number 1. This will become clearer when we discuss color depths below.

Creating A Pixmap

Sometimes we want to create an un-initialized pixmap, so we can later draw into it. This is useful for image drawing programs (creating a new empty canvas will cause the creation of a new pixmap on which the drawing can be stored). It is also useful when reading various image formats - we load the image data into memory, cerate a pixmap on the server, and then draw the decoded image data onto that pixmap.

```
/* this variable will store the handle of the newly created pixmap. */
Pixmap pixmap;

/* this variable will contain the ID of the root window of the screen */
/* for which we want the pixmap to be created.                        */
Window root_win = DefaultRootWindow(display);

/* this variable will contain the color depth of the pixmap to create. */
/* this 'depth' specifies the number of bits used to represent a color */
/* index in the color map. the number of colors is 2 to the power of   */
/* this depth.                                                         */
int depth = DefaultDepth(display, DefaultScreen(display));

/* create a new pixmap, with a width of 30 pixels, and height of 40 pixels. */
pixmap = XCreatePixmap(display, root_win, 30, 40, depth);

/* just for fun, draw a pixel in the middle of this pixmap. */
XDrawPoint(display, pixmap, gc, 15, 20);
```

Drawing A Pixmap In A Window

Once we got a handle to pixmap, we can draw it on some window, using the `XCopyArea()` function. This function allows us to specify what drawable (a window, or even another pixmap) to draw the given pixmap onto, and at what location in that drawable.

```
/* draw the previously loaded bitmap on the given window, at location   */
/* 'x=100, y=50' in that window. we want to copy the whole bitmap, so   */
/* we specify location 'x=0, y=0' of the bitmap to start the copy from, */
/* and the full size of the bitmap, to specify how much of it to copy.  */
XCopyArea(display, bitmap, win, gc,
          0, 0,
          bitmap_width, bitmap_height,
          100, 50);
```

As you can see, we could also copy a given rectangle of the pixmap, instead of the whole pixmap.

One important note should be made - it is possible to create pixmaps of different depths on the same screen. When we perform copy operations (a pixmap onto a window, etc), we should make sure that both source and target have the same depth. If they have a different depth, the operation would fail. The exception to this is if we copy a specific bit plane of the source pixmap, using the `XCopyPlane()` function shown earlier. In such an event, we can copy a specific plain to the target window - in actuality, setting a specific bit in the color of each pixel copied. This can be used to generate strange graphic effects in window, but is beyond the scope of out tutorial.

Freeing A Pixmap

Finally, when we are done using a given pixmap, we should free it, in order to free resources of the X server. This is done using the `XFreePixmap()` function:

```
/* free the pixmap with the given ID. */
XFreePixmap(display, pixmap);
```

After freeing a pixmap - we must not try accessing it again.

To summarize this section, take a look at the draw-pixmap.c program, to see a pixmap being created using a bitmap file, and then tiled up on a window on your screen.

Messing With The Mouse Cursor

Often we see programs that modify the shape of the mouse pointer (also called the X pointer) when in certain states. For example, a busy application would often display a sand clock over its main window, to give the user a visual hint that they should wait. Without such a visual hint, the user might think that the application got stuck. Lets see how we can change the mouse cursor for our windows.

Creating And Destroying A Mouse Cursor

There are two methods for creating cursors. One of them is by using a set of pre-defined cursors, that are supplied by Xlib. The other is by using user-supplied bitmaps.

In the first method, we use a special font named "cursor", and the function `XCreateFontCursor()`. This function accepts a shape identifier, and returns a handle to the generated cursor. The list of allowed font identifiers is found in the include file <X11/cursorfont.h>. Here are a few such cursors:

`XC_arrow`
: The normal pointing-arrow cursor displayed by the server.

`XC_pencil`
: A cursor shaped as a pencil.

`XC_watch`
: A sand watch.

And creating a cursor using these symbols is very easy:

```
#include <X11/cursorfont.h>    /* defines XC_watch, etc. */

/* this variable will hold the handle of the newly created cursor. */
Cursor watch_cursor;

/* create a sand watch cursor. */
watch_cursor = XCreateFontCursor(display, XC_watch);
```

The other methods of creating a cursor is by using a pair of pixmaps with depth of one (that is, two color pixmaps). One pixmap defines the shape of the cursor, while the other works as a mask, specifying which pixels of the cursor will be actually drawn. The rest of the pixels will be transparent. Creating such a cursor is done using the `XCreatePixmapCursor()` function. As an example, we will create a cursor using the "icon.bmp" bitmap. We will assume that it was already loaded into memory, and turned into a pixmap, and its handle is stored in the 'bitmap' variable. We will want it to be fully transparent. That is, only the parts of it that are black will be drawn, while the white parts will be transparent. To achieve this effect, we will use the icon both as the cursor pixmap and as the mask pixmap. Try to figure out why...

```
/* this variable will hold the handle of the newly created cursor. */
Cursor icon_cursor;

/* first, we need to define foreground and background colors for the cursor. */
XColor cursor_fg, cursor_bg;

/* access the default color map of our screen. */
Colormap screen_colormap = DefaultColormap(display, DefaultScreen(display));

/* allocate black and while colors. */
Status rc = XAllocNamedColor(display,
                             screen_colormap,
                             "black",
                             &cursor_fg,
                             &cursor_fg);
if (rc == 0) {
    fprintf(stderr, "XAllocNamedColor - cannot allocate 'black' ??!!??\n");
    exit(1);
}
Status rc = XAllocNamedColor(display,
                             screen_colormap,
                             "white",
                             &cursor_bg,
                             &cursor_bg);
if (rc == 0) {
    fprintf(stderr, "XAllocNamedColor - cannot allocate 'white' ??!!??\n");
    exit(1);
}

/* finally, generate the cursor. make the 'hot spot' be close to the */
/* top-left corner of the cursor - location (x=5, y=4). */
icon_cursor = XCreatePixmapCursor(display, bitmap, bitmap,
                                  &cursor_fg, &cursor_bg,
                                  5, 4);
```

One thing to be explained is the 'hot spot' parameters. When we define a cursor, we need to define which pixel of the cursor is the pointer delivered to the user in the various mouse events. Usually, we will choose a location of the cursor that visually looks like a hot spot. For example, in an arrow cursor, the tip of the arrow will be defined as the hot spot.

Finally, when we are done with a cursor and no longer need it, we can release it using the `XFreeCursor()` function:

```
XFreeCursor(display, icon_cursor);
```

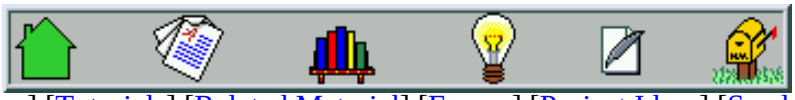
Setting A Window's Mouse Cursor

After we have created a cursor, we can tell the X server to attach this cursor to any of our windows. This is done using the `XDefineCursor()`, and causes the X server to change the mouse pointer to the shape of that cursor, each time the mouse pointer moves into and across that window. We can later detach this cursor from our window using the `XUndefineCursor()` function. This will cause the default cursor to be shown when the mouse enter that windows.

```
/* attach the icon cursor to our window. */
XDefineCursor(display, win, icon_cursor);

/* detach the icon cursor from our window. */
```

```
XUndefineCursor(display, win);
```

As an example, look at our cursor.c program, and see how mouse cursors are set, changed and removed. Run the program, place the mouse pointer over the created window, and watch.



```
/*
 * simple-window.c - demonstrate creation of a simple window.
 */

#include <X11/Xlib.h>

#include <stdio.h>
#include <stdlib.h>             /* getenv(), etc. */
#include <unistd.h>             /* sleep(), etc.  */

void
main(int argc, char* argv[])
{
  Display* display;             /* pointer to X Display structure.           */
  int screen_num;               /* number of screen to place the window on.  */
  Window win;                   /* pointer to the newly created window.      */
  unsigned int display_width,
               display_height;  /* height and width of the X display.        */
  unsigned int width, height;   /* height and width for the new window.      */
  unsigned int win_x, win_y;    /* location of the window's top-left corner. */
  unsigned int win_border_width; /* width of window's border.                */
  char *display_name = getenv("DISPLAY");  /* address of the X display.      */

  display = XOpenDisplay(display_name);
  if (display == NULL) {
    fprintf(stderr, "%s: cannot connect to X server '%s'\n",
            argv[0], display_name);
    exit(1);
  }

  /* get the geometry of the default screen for our display. */
  screen_num = DefaultScreen(display);
  display_width = DisplayWidth(display, screen_num);
  display_height = DisplayHeight(display, screen_num);

  /* make the new window occupy 1/9 of the screen's size. */
  width = (display_width / 3);
  height = (display_height / 3);

  /* the window should be placed at the top-left corner of the screen. */
  win_x = 0;
  win_y = 0;

  /* the window's border shall be 2 pixels wide. */
  win_border_width = 2;

  /* create a simple window, as a direct child of the screen's   */
  /* root window. Use the screen's white color as the background */
  /* color of the window. Place the new window's top-left corner */
  /* at the given 'x,y' coordinates.                             */
  win = XCreateSimpleWindow(display, RootWindow(display, screen_num),
                            win_x, win_y, width, height, win_border_width,
                            BlackPixel(display, screen_num),
                            WhitePixel(display, screen_num));

  /* make the window actually appear on the screen. */
  XMapWindow(display, win);

  /* flush all pending requests to the X server, and wait until */
  /* they are processed by the X server.                        */
```

```
  XSync(display, False);

  /* make a delay for a short period. */
  sleep(4);

  /* close the connection to the X server. */
  XCloseDisplay(display);
}
```

```
/*
 * simple-drawing.c - demonstrate drawing of pixels, lines, arcs, etc.
 *                    on a window. All drawings are done in black color
 *                    over a white background.
 */

#include <X11/Xlib.h>

#include <stdio.h>
#include <stdlib.h>             /* getenv(), etc. */
#include <unistd.h>             /* sleep(), etc.  */

/*
 * function: create_simple_window. Creates a window with a white background
 *           in the given size.
 * input:    display, size of the window (in pixels), and location of the window
 *           (in pixels).
 * output:   the window's ID.
 * notes:    window is created with a black border, 2 pixels wide.
 *           the window is automatically mapped after its creation.
 */
Window
create_simple_window(Display* display, int width, int height, int x, int y)
{
  int screen_num = DefaultScreen(display);
  int win_border_width = 2;
  Window win;

  /* create a simple window, as a direct child of the screen's */
  /* root window. Use the screen's black and white colors as   */
  /* the foreground and background colors of the window,       */
  /* respectively. Place the new window's top-left corner at   */
  /* the given 'x,y' coordinates.                              */
  win = XCreateSimpleWindow(display, RootWindow(display, screen_num),
                            x, y, width, height, win_border_width,
                            BlackPixel(display, screen_num),
                            WhitePixel(display, screen_num));

  /* make the window actually appear on the screen. */
  XMapWindow(display, win);

  /* flush all pending requests to the X server. */
  XFlush(display);

  return win;
}

GC
create_gc(Display* display, Window win, int reverse_video)
{
  GC gc;                                /* handle of newly created GC.  */
  unsigned long valuemask = 0;          /* which values in 'values' to  */
                                        /* check when creating the GC.  */
  XGCValues values;                     /* initial values for the GC.   */
  unsigned int line_width = 2;          /* line width for the GC.       */
  int line_style = LineSolid;           /* style for lines drawing and  */
  int cap_style = CapButt;              /* style of the line's edje and */
  int join_style = JoinBevel;           /*  joined lines.               */
  int screen_num = DefaultScreen(display);
```

```
  gc = XCreateGC(display, win, valuemask, &values);
  if (gc < 0) {
        fprintf(stderr, "XCreateGC: \n");
  }

  /* allocate foreground and background colors for this GC. */
  if (reverse_video) {
    XSetForeground(display, gc, WhitePixel(display, screen_num));
    XSetBackground(display, gc, BlackPixel(display, screen_num));
  }
  else {
    XSetForeground(display, gc, BlackPixel(display, screen_num));
    XSetBackground(display, gc, WhitePixel(display, screen_num));
  }

  /* define the style of lines that will be drawn using this GC. */
  XSetLineAttributes(display, gc,
                     line_width, line_style, cap_style, join_style);

  /* define the fill style for the GC. to be 'solid filling'. */
  XSetFillStyle(display, gc, FillSolid);

  return gc;
}

void
main(int argc, char* argv[])
{
  Display* display;             /* pointer to X Display structure.           */
  int screen_num;               /* number of screen to place the window on.  */
  Window win;                   /* pointer to the newly created window.      */
  unsigned int display_width,
               display_height;  /* height and width of the X display.        */
  unsigned int width, height;   /* height and width for the new window.      */
  char *display_name = getenv("DISPLAY");  /* address of the X display.      */
  GC gc;                        /* GC (graphics context) used for drawing    */
                                /*  in our window.                           */

  /* open connection with the X server. */
  display = XOpenDisplay(display_name);
  if (display == NULL) {
    fprintf(stderr, "%s: cannot connect to X server '%s'\n",
            argv[0], display_name);
    exit(1);
  }

  /* get the geometry of the default screen for our display. */
  screen_num = DefaultScreen(display);
  display_width = DisplayWidth(display, screen_num);
  display_height = DisplayHeight(display, screen_num);

  /* make the new window occupy 1/9 of the screen's size. */
  width = (display_width / 3);
  height = (display_height / 3);
  printf("window width - '%d'; height - '%d'\n", width, height);

  /* create a simple window, as a direct child of the screen's   */
  /* root window. Use the screen's white color as the background */
  /* color of the window. Place the new window's top-left corner */
  /* at the given 'x,y' coordinates.                             */
```

```
  win = create_simple_window(display, width, height, 0, 0);

  /* allocate a new GC (graphics context) for drawing in the window. */
  gc = create_gc(display, win, 0);
  XSync(display, False);

  /* draw one pixel near each corner of the window */
  XDrawPoint(display, win, gc, 5, 5);
  XDrawPoint(display, win, gc, 5, height-5);
  XDrawPoint(display, win, gc, width-5, 5);
  XDrawPoint(display, win, gc, width-5, height-5);

  /* draw two intersecting lines, one horizontal and one vertical, */
  /* which intersect at point "50,100".                            */
  XDrawLine(display, win, gc, 50, 0, 50, 200);
  XDrawLine(display, win, gc, 0, 100, 200, 100);

  /* now use the XDrawArc() function to draw a circle whose diameter */
  /* is 30 pixels, and whose center is at location '50,100'.         */
  XDrawArc(display, win, gc, 50-(30/2), 100-(30/2), 30, 30, 0, 360*64);

  {
    XPoint points[] = {
      {0, 0},
      {15, 15},
      {0, 15},
      {0, 0}
    };
    int npoints = sizeof(points)/sizeof(XPoint);

    /* draw a small triangle at the top-left corner of the window. */
    /* the triangle is made of a set of consecutive lines, whose   */
    /* end-point pixels are specified in the 'points' array.       */
    XDrawLines(display, win, gc, points, npoints, CoordModeOrigin);
  }

  /* draw a rectangle whose top-left corner is at '120,150', its width is */
  /* 50 pixels, and height is 60 pixels.                                  */
  XDrawRectangle(display, win, gc, 120, 150, 50, 60);

  /* draw a filled rectangle of the same size as above, to the left of the */
  /* previous rectangle.                                                   */
  XFillRectangle(display, win, gc, 60, 150, 50, 60);

  /* flush all pending requests to the X server. */
  XFlush(display);

  /* make a delay for a short period. */
  sleep(4);

  /* close the connection to the X server. */
  XCloseDisplay(display);
}
```

```
/*
 * events.c - demonstrate handling of X events using an events loop.
 */

#include <X11/Xlib.h>

#include <stdio.h>
#include <stdlib.h>             /* getenv(), etc. */

/*
 * function: create_simple_window. Creates a window with a white background
 *           in the given size.
 * input:    display, size of the window (in pixels), and location of the window
 *           (in pixels).
 * output:   the window's ID.
 * notes:    window is created with a black border, 2 pixels wide.
 *           the window is automatically mapped after its creation.
 */
Window
create_simple_window(Display* display, int width, int height, int x, int y)
{
  int screen_num = DefaultScreen(display);
  int win_border_width = 2;
  Window win;

  /* create a simple window, as a direct child of the screen's */
  /* root window. Use the screen's black and white colors as   */
  /* the foreground and background colors of the window,       */
  /* respectively. Place the new window's top-left corner at   */
  /* the given 'x,y' coordinates.                              */
  win = XCreateSimpleWindow(display, RootWindow(display, screen_num),
                            x, y, width, height, win_border_width,
                            BlackPixel(display, screen_num),
                            WhitePixel(display, screen_num));

  /* make the window actually appear on the screen. */
  XMapWindow(display, win);

  /* flush all pending requests to the X server. */
  XFlush(display);

  return win;
}

GC
create_gc(Display* display, Window win, int reverse_video)
{
  GC gc;                                /* handle of newly created GC.  */
  unsigned long valuemask = 0;          /* which values in 'values' to  */
                                        /* check when creating the GC.  */
  XGCValues values;                     /* initial values for the GC.   */
  unsigned int line_width = 2;          /* line width for the GC.       */
  int line_style = LineSolid;           /* style for lines drawing and  */
  int cap_style = CapButt;              /* style of the line's edje and */
  int join_style = JoinBevel;           /*  joined lines.               */
  int screen_num = DefaultScreen(display);

  gc = XCreateGC(display, win, valuemask, &values);
  if (gc < 0) {
        fprintf(stderr, "XCreateGC: \n");
```

```
  }

  /* allocate foreground and background colors for this GC. */
  if (reverse_video) {
    XSetForeground(display, gc, WhitePixel(display, screen_num));
    XSetBackground(display, gc, BlackPixel(display, screen_num));
  }
  else {
    XSetForeground(display, gc, BlackPixel(display, screen_num));
    XSetBackground(display, gc, WhitePixel(display, screen_num));
  }

  /* define the style of lines that will be drawn using this GC. */
  XSetLineAttributes(display, gc,
                     line_width, line_style, cap_style, join_style);

  /* define the fill style for the GC. to be 'solid filling'. */
  XSetFillStyle(display, gc, FillSolid);

  return gc;
}

/*
 * function: handle_expose. handles an Expose event by redrawing the window.
 * input:    display, 2 GCs, XExposeEvent event structure, dimensions of
 *           the window, pixels array.
 * output:   none.
 */
void
handle_expose(Display* display, GC gc, GC rev_gc, XExposeEvent* expose_event,
              unsigned int win_width, unsigned int win_height,
              short pixels[1000][1000])
{
  /* if this is the first in a set of expose events - ignore this event. */
  if (expose_event->count != 0)
    return;
  /* draw the contents of our window. */

  /* draw one pixel near each corner of the window */
  XDrawPoint(display, expose_event->window, gc, 5, 5);
  XDrawPoint(display, expose_event->window, gc, 5, win_height-5);
  XDrawPoint(display, expose_event->window, gc, win_width-5, 5);
  XDrawPoint(display, expose_event->window, gc, win_width-5, win_height-5);

  /* draw two intersecting lines, one horizontal and one vertical, */
  /* which intersect at point "50,100".                            */
  XDrawLine(display, expose_event->window, gc, 50, 0, 50, 200);
  XDrawLine(display, expose_event->window, gc, 0, 100, 200, 100);

  /* now use the XDrawArc() function to draw a circle whose diameter */
  /* is 30 pixels, and whose center is at location '50,100'.         */
  XDrawArc(display, expose_event->window, gc,
           50-(30/2), 100-(30/2), 30, 30, 0, 360*64);

  {
    XPoint points[] = {
      {0, 0},
      {15, 15},
      {0, 15},
      {0, 0}
```

```
    };
    int npoints = sizeof(points)/sizeof(XPoint);

    /* draw a small triangle at the top-left corner of the window. */
    /* the triangle is made of a set of consecutive lines, whose   */
    /* end-point pixels are specified in the 'points' array.       */
    XDrawLines(display, expose_event->window, gc,
               points, npoints, CoordModeOrigin);
  }

  /* draw a rectangle whose top-left corner is at '120,150', its width is */
  /* 50 pixels, and height is 60 pixels.                                  */
  XDrawRectangle(display, expose_event->window, gc, 120, 150, 50, 60);

  /* draw a filled rectangle of the same size as above, to the left of the */
  /* previous rectangle.                                                   */
  XFillRectangle(display, expose_event->window, gc, 60, 150, 50, 60);

  /* finally, draw all the pixels in the 'pixels' array. */
  {
    int x, y;
    for (x=0; x<win_width; x++)
      for (y=0; y<win_height; y++)
        switch(pixels[x][y]) {
          case 1: /* draw point. */
            XDrawPoint(display, expose_event->window, gc, x, y);
            break;
          case -1: /* erase point. */
            XDrawPoint(display, expose_event->window, rev_gc, x, y);
            break;
        }
  }
}

/*
 * function: handle_drag. handles a Mouse drag event - if the left button
 *           is depressed - draws the pixel below the mouse pointer. if the
 *           middle button is depressed - erases the pixel below the mouse
 *           pointer.
 * input:    display, 2 GCs, XButtonEvent event structure, dimensions of
 *           the window, pixels array.
 * output:   none.
 */
void
handle_drag(Display* display, GC gc, GC rev_gc, XButtonEvent* drag_event,
            unsigned int win_width, unsigned int win_height,
            short pixels[1000][1000])
{
  int x, y;

  /* invert the pixel under the mouse. */
  x = drag_event->x;
  y = drag_event->y;
  switch (drag_event->state) {
    case Button1Mask: /* draw the given pixel in black color. */
      XDrawPoint(display, drag_event->window, gc, x, y);
      pixels[x][y] = 1;
      break;
    case Button2Mask: /* draw the given pixel in white color. */
      XDrawPoint(display, drag_event->window, rev_gc, x, y);
```

```
      pixels[x][y] = -1;
      break;
  }
}

/*
 * function: handle_button_down. handles a Mouse press event - if the left
 *           button is depressed - draws the pixel below the mouse pointer.
 *           if the middle button is depressed - erases the pixel below the
 *           mouse pointer.
 * input:    display, 2 GCs, XButtonEvent event structure, dimensions of
 *           the window, pixels array.
 * output:   none.
 */
void
handle_button_down(Display* display, GC gc, GC rev_gc,
                   XButtonEvent* button_event,
                   unsigned int win_width, unsigned int win_height,
                   short pixels[1000][1000])
{
  int x, y;

  /* invert the pixel under the mouse. */
  x = button_event->x;
  y = button_event->y;
  switch (button_event->button) {
    case Button1: /* draw the given pixel in black color. */
      XDrawPoint(display, button_event->window, gc, x, y);
      pixels[x][y] = 1;
      break;
    case Button2: /* draw the given pixel in white color. */
      XDrawPoint(display, button_event->window, rev_gc, x, y);
      pixels[x][y] = -1;
      break;
  }
}

void
main(int argc, char* argv[])
{
  Display* display;             /* pointer to X Display structure.           */
  int screen_num;               /* number of screen to place the window on.  */
  Window win;                   /* pointer to the newly created window.      */
  unsigned int display_width,
               display_height;  /* height and width of the X display.        */
  unsigned int width, height;   /* height and width for the new window.      */
  char *display_name = getenv("DISPLAY");  /* address of the X display.      */
  GC gc, rev_gc;                /* GC (graphics context) used for drawing    */
                                /*  in our window.                           */
  short pixels[1000][1000];     /* used to store pixels on screen that were  */
                                /* explicitly drawn or erased by the user.   */

  /* initialize the 'pixels' array to contain 0 values. */
  {
    int x, y;
    for (x=0; x<1000; x++)
      for (y=0; y<1000; y++)
        pixels[x][y] = 0;
  }
```

```
/* open connection with the X server. */
display = XOpenDisplay(display_name);
if (display == NULL) {
  fprintf(stderr, "%s: cannot connect to X server '%s'\n",
          argv[0], display_name);
  exit(1);
}

/* get the geometry of the default screen for our display. */
screen_num = DefaultScreen(display);
display_width = DisplayWidth(display, screen_num);
display_height = DisplayHeight(display, screen_num);

/* make the new window occupy 1/9 of the screen's size. */
width = (display_width / 3);
height = (display_height / 3);
printf("window width - '%d'; height - '%d'\n", width, height);

/* create a simple window, as a direct child of the screen's   */
/* root window. Use the screen's white color as the background */
/* color of the window. Place the new window's top-left corner */
/* at the given 'x,y' coordinates.                             */
win = create_simple_window(display, width, height, 0, 0);

/* allocate two new GCs (graphics contexts) for drawing in the window. */
/* the first is used for drawing black over white, the second is used  */
/* for drawing white over black.                                       */
gc = create_gc(display, win, 0);
rev_gc = create_gc(display, win, 1);

/* subscribe to the given set of event types. */
XSelectInput(display, win, ExposureMask | KeyPressMask |
                   ButtonPressMask | Button1MotionMask |
                   Button2MotionMask | StructureNotifyMask);

/* perform an events loop */
{
  int done = 0;
  XEvent an_event;
  while (!done) {
    XNextEvent(display, &an_event);
    switch (an_event.type) {
      case Expose:
        /* redraw our window. */
        handle_expose(display, gc, rev_gc, (XExposeEvent*)&an_event.xexpose,
                      width, height, pixels);
        break;

      case ConfigureNotify:
        /* update the size of our window, for expose events. */
        width = an_event.xconfigure.width;
        height = an_event.xconfigure.height;
        break;

      case ButtonPress:
        /* invert the pixel under the mouse pointer. */
        handle_button_down(display, gc, rev_gc,
                           (XButtonEvent*)&an_event.xbutton,
                           width, height, pixels);
        break;
```

```
        case MotionNotify:
          /* invert the pixel under the mouse pointer. */
          handle_drag(display, gc, rev_gc,
                      (XButtonEvent*)&an_event.xbutton,
                      width, height, pixels);
          break;

        case KeyPress:
          /* exit the application by braking out of the events loop. */
          done = 1;
          break;

        default: /* ignore any other event types. */
          break;
      } /* end switch on event type */
    } /* end while events handling */
  }

  /* free the GCs. */
  XFreeGC(display, gc);
  XFreeGC(display, rev_gc);

  /* close the connection to the X server. */
  XCloseDisplay(display);
}
```

```
/*
 * simple-text.c - demonstrate drawing of text strings on a window. All
 *                 drawings are done in black color over a white background.
 */

#include <X11/Xlib.h>

#include <stdio.h>
#include <stdlib.h>             /* getenv(), etc. */
#include <unistd.h>             /* sleep(), etc.  */

/*
 * function: create_simple_window. Creates a window with a white background
 *           in the given size.
 * input:    display, size of the window (in pixels), and location of the window
 *           (in pixels).
 * output:   the window's ID.
 * notes:    window is created with a black border, 2 pixels wide.
 *           the window is automatically mapped after its creation.
 */
Window
create_simple_window(Display* display, int width, int height, int x, int y)
{
  int screen_num = DefaultScreen(display);
  int win_border_width = 2;
  Window win;

  /* create a simple window, as a direct child of the screen's */
  /* root window. Use the screen's black and white colors as   */
  /* the foreground and background colors of the window,       */
  /* respectively. Place the new window's top-left corner at   */
  /* the given 'x,y' coordinates.                              */
  win = XCreateSimpleWindow(display, RootWindow(display, screen_num),
                            x, y, width, height, win_border_width,
                            BlackPixel(display, screen_num),
                            WhitePixel(display, screen_num));

  /* make the window actually appear on the screen. */
  XMapWindow(display, win);

  /* flush all pending requests to the X server. */
  XFlush(display);

  return win;
}

GC
create_gc(Display* display, Window win, int reverse_video)
{
  GC gc;                                /* handle of newly created GC.  */
  unsigned long valuemask = 0;          /* which values in 'values' to  */
                                        /* check when creating the GC.  */
  XGCValues values;                     /* initial values for the GC.   */
  unsigned int line_width = 2;          /* line width for the GC.       */
  int line_style = LineSolid;           /* style for lines drawing and  */
  int cap_style = CapButt;              /* style of the line's edje and */
  int join_style = JoinBevel;           /*  joined lines.               */
  int screen_num = DefaultScreen(display);

  gc = XCreateGC(display, win, valuemask, &values);
```

```
  if (gc < 0) {
        fprintf(stderr, "XCreateGC: \n");
  }

  /* allocate foreground and background colors for this GC. */
  if (reverse_video) {
    XSetForeground(display, gc, WhitePixel(display, screen_num));
    XSetBackground(display, gc, BlackPixel(display, screen_num));
  }
  else {
    XSetForeground(display, gc, BlackPixel(display, screen_num));
    XSetBackground(display, gc, WhitePixel(display, screen_num));
  }

  /* define the style of lines that will be drawn using this GC. */
  XSetLineAttributes(display, gc,
                     line_width, line_style, cap_style, join_style);

  /* define the fill style for the GC. to be 'solid filling'. */
  XSetFillStyle(display, gc, FillSolid);

  return gc;
}

void
main(int argc, char* argv[])
{
  Display* display;             /* pointer to X Display structure.           */
  int screen_num;               /* number of screen to place the window on.  */
  Window win;                   /* pointer to the newly created window.      */
  unsigned int display_width,
               display_height;  /* height and width of the X display.        */
  unsigned int win_width,
               win_height;      /* height and width for the new window.      */
  char *display_name = getenv("DISPLAY");  /* address of the X display.      */
  GC gc;                        /* GC (graphics context) used for drawing    */
                                /*  in our window.                           */
  XFontStruct* font_info;       /* Font structure, used for drawing text.    */
  char* font_name = "*-helvetica-*-12-*"; /* font to use for drawing text.   */

  /* open connection with the X server. */
  display = XOpenDisplay(display_name);
  if (display == NULL) {
    fprintf(stderr, "%s: cannot connect to X server '%s'\n",
            argv[0], display_name);
    exit(1);
  }

  /* get the geometry of the default screen for our display. */
  screen_num = DefaultScreen(display);
  display_width = DisplayWidth(display, screen_num);
  display_height = DisplayHeight(display, screen_num);

  /* make the new window occupy 1/9 of the screen's size. */
  win_width = (display_width / 3);
  win_height = (display_height / 3);
  printf("window width - '%d'; height - '%d'\n", win_width, win_height);

  /* create a simple window, as a direct child of the screen's   */
  /* root window. Use the screen's white color as the background */
```

```
  /* color of the window. Place the new window's top-left corner */
  /* at the given 'x,y' coordinates.                             */
  win = create_simple_window(display, win_width, win_height, 0, 0);

  /* allocate a new GC (graphics context) for drawing in the window. */
  gc = create_gc(display, win, 0);
  XSync(display, False);

  /* try to load the given font. */
  font_info = XLoadQueryFont(display, font_name);
  if (!font_info) {
      fprintf(stderr, "XLoadQueryFont: failed loading font '%s'\n", font_name);
      exit(-1);
  }

  /* assign the given font to our GC. */
  XSetFont(display, gc, font_info->fid);

  {
    /* variables used for drawing the text strings. */
    int x, y;
    char* text_string;
    int string_width;
    int font_height;

    /* find the height of the characters drawn using this font.        */
    font_height = font_info->ascent + font_info->descent;

    /* draw a "hello world" string on the top-left side of our window. */
    text_string = "hello world";
    x = 0;
    y = font_height;
    XDrawString(display, win, gc, x, y, text_string, strlen(text_string));

    /* draw a "middle of the road" string in the middle of our window. */
    text_string = "middle of the road";
    /* find the width, in pixels, of the text that will be drawn using */
    /* the given font.                                                 */
    string_width = XTextWidth(font_info, text_string, strlen(text_string));
    x = (win_width - string_width) / 2;
    y = (win_height + font_height) / 2;
    XDrawString(display, win, gc, x, y, text_string, strlen(text_string));
  }

  /* flush all pending requests to the X server. */
  XFlush(display);

  /* make a delay for a short period. */
  sleep(4);

  /* close the connection to the X server. */
  XCloseDisplay(display);
}
```

THE Open GROUP

Site Map | Login | Register | Feedback | Contacts | Search

Standards | Licensing | Publications | Procurement | Brand | Testing & Certification

SPONSORS

Your browser does not support Java applets. Visit our sponsors page, anyhow.

About Us
Membership
Conferences & Meetings
Information Services
Our Locations

Architecture
Directory
Management
Mobile Computing
Security
Platform
Consortia Services
Other Activities

Advanced Research

Brochure Downloads

Network Management Systems
June 12-14, 2001

Testing & Certification

- WAP Forum™ Certification Authority
- WAP certified products listing
- WAP Application Layer Development Test Suite 1.0
- TETworks
- General

EMA Forum

- The Open Group and EMA Combine Forces to work on eBusiness Solutions

WIRELESS & MOBILE COMPUTING

MMF vendors and customers will be attending The Wireless-Enabled Enterprise in Berlin 23-24 April

Real-Time & Embedded Systems Forum

- Enter the Real-Time Forum
- SGI Joins the Open Group's Real-Time and Embedded Systems Forum as Industry Leader in Real-Time Solutions

Quality of Service Task Force

- New! Read all about it!

BrainWidth

- A news bulletin from The Open Group
- NEW Issue 6, covering March 2001

Manageability 2001
The Well-Managed Enterprise

Austin, Texas, USA 16-20 July 2001

The Wireless-Enabled Enterprise

23-27 April 2001 in Berlin, Germany

NON-MEMBERS: Read about the highlights of the conference

MEMBERS & DELEGATES: Read the full conference documentation

THE Open GROUP REGIONAL CHAPTERS

- REGIONAL CHAPTERS ... around the world
- Gothenburg, Sweden ... 16 May, 2001
- New Delhi, India .... 14 June 2001

THE Open GROUP PRESS RELEASES

- The Open Group Launches Works With LDAP 2000 Certification Program
- BMC Software Collaborates with The Open Group to develop Open Management Standards
- *Go to the full list of Press Releases*

THE Open GROUP IN THE NEWS

The Open Group links to information considered relevant and valuable to our members. This information is independent and the views and opinions expressed in such links do not necessarily reflect the views of The Open Group, it's members or it's sponsors

- Novell Expands Directory Leadership through New Tools, Pricing Model and Continued Support for Industry Standards
- An Interview With Scott Goldman
- *Go to the full list of Press Coverage*

We thank Compaq Computer Corporation for providing the system to host our electronic mail services

HEADLINES

- **OMG Declares Independence From Platforms. SDTimes, David Rubinstein**

- **UML Models E-Business White Paper. Software Magazine, April/May, 2001**

- **The House That CORBA Built. Paul May, Application Development Advisor.**

- **Download OMG Member Newsletter**

Press Releases

- **OMG Members Meet to Standardize Industry-specific Software; *Members and Guests to Meet in Danvers***

- **Object in Bio- and Chem-informatics to meet at Danvers TC Meeting**

- **Pharmaceutical Supply Chain Management Effort to meet at Danvers TC Meeting**

- **Paris T.C. Meeting Wrap Up**

UPCOMING EVENTS

- ***Call for Presentations: OMG's Workshop on "UML for Enterprise Applications"***

- ***Call for Presentations: "Telecom and OMG Technologies" Workshop***

- ***Call for Presentations: "OMG Workshop on Embedded & Real-Time Distributed Object Systems"***

- ***OMG Boston (Danvers) TC Meeting July 9-13 Agendas***

  - OiBC 2001 - July 9 & 10
  - Pharmaceutical Industry Supply Chain Management Kick-off - July 10 & 11
  - Software Services Grid Workshop July 10

- **OMG New Strategic Direction:**
  Model Driven Architecture ™

OMG Partner Events!

Click here to view ALL upcoming events

Attention Members!  ***Change Password*** ***Lost Password***

The Free CORBA page

If you're interested in Java, why not check out my Java page?

If you're interested in distributed objects, why not check out my Enterprise Java Beans page?

Cheesy money-making effort

I've put together a list of books I like here. If you click on one of the hyperlinked titles, it'll take you to amazon.com. If you buy the book from amazon, they send me a kickback. Cool, huh? Even if you don't buy anything, check these books out; they're all really good.

The purpose of this page is to provide a list of links to free implementations of CORBA. If you know of any free CORBA implementations not listed here, please contact me and I'll list it as well. Additionally, if you find that anything that I've listed here is *not* free, let me know and I'll remove it from the list.

Caveat Emptor! This list is provided in order to make your search for a free ORB easier. The presence of an ORB on this page does not constitute an endorsement of it. Please be sure to review any licensing agreements.

Feature lists (links to pages by Ben Eng)

Here is a matrix of the various commercial and non-commercial ORBs and the language bindings and features that they support.
Here is a matrix of the various commercial and non-commercial ORBs and the CORBA services which they support.
Here is a matrix of the various commercial and non-commercial ORBs and the platforms on which they run.
These matrices were created, and are currently maintained, by Ben Eng, who has some other neat stuff on his homepage as well.

Benchmarks

A new paper from MLC Systeme GmbH and the Distributed Systems Research Group of Prague's Charles University features comparative benchmarks for Inprise Visibroker, Iona's Orbix, and ORL's OmniORB. The paper can be found here.

ExperSoft has posted a white paper on benchmarking enterprise ORBs. This paper should be helpful to you if you intend to do comparative benchmarking on prospective ORBs.

Enough chitchat! On to the list!

Free ORBs

VBOrb
From the author:

```
VBOrb is an object request broker entirely written in Visual Basic.
With VBOrb you can write CORBA clients and servers in Visual Basic directly
IDL2VB is the IDL compiler belongs to VBOrb.
VBOrb and IDL2VB are free to use.
```

ADABroker
AdaBroker is a set of tools and libraries that can be used to develop CORBA applications in Ada.

FLICK
FLICK stands for "Flexible IDL Compiler Kit". This is not an ORB; it's an optimizing IDL compiler. From the authors.

```
  Flick is the flexible interface definition language (IDL) compiler from the
```

```
University of Utah.  What sets Flick apart from other IDL compilers is that
it is highly optimizing while also supporting several IDLs, message formats,
and transport mechanisms.  Flick currently has front ends for the CORBA, Sun
ONC RPC, and Mach MIG IDLs, and middle and back ends that support CORBA IIOP,
ONC/TCP, MIG-style Mach messages, and other specialized transports.  Flick
produces C stubs for all IDLs and both C and C++ stubs for CORBA IDL.
```

Mico/E

An open-source port of the Mico ORB (see below) to the Eiffel language.

LuaORB

LuaORB is a language binding for the interpreted language Lua. It's been tested with the Orbacus C++ ORB, version 3.1.2.

JavaORB

A Java CORBA 2.3 ORB. Here's their description:

```
- Fully Compliant CORBA 2.3 Specification :
POA, BOA, OBV, Thread Policies, Activation Daemon and Policies, Interceptors

- Services :
Security, Transaction, Notification, Trading, Naming, Persistence, Event

- Extensions :
Pure Java RMI/IIOP implementation

- Free to download and use for both commercial and non-commercial projects
```

The same folks are also working on an Enterprise JavaBeans implementation called AnEJB

Engine Room CORBA

A free ORB written by Mitchell Britton. Features:

- IDL compiler written in 100% pure Java
- Stubs produced for 'C'
- Stubs produced for C++
- Portable stubs and skeletons produced in 100% pure Java for Java according to the CORBA 2.2 spec.
- ORB source and examples included
- Ported to Win95/NT (Borland C++ and Visual C++), Solaris and LINUX
- It's free

Jonathan

From the author: *Jonathan is an Object Request Broker written entirely in Java. It is an "open" ORB, in the sense that contrary to standard ORBs, the abstractions that make up the internal ORB machinery may be used by an application programmer and specialized to meet specific requirements.*

Jonathan ... is now completely free for commercial or non-commercial usage (it is released under the LGPL license).

ORBit

A new Corba 2.2-compliant ORB with C bindings. ADA bindings are also available.

DynaORB

From the author's pages:
DynaORB is a lightweight universal CORBA client component. By lightweight, it is meant that it can be quickly transferred across a network. By universal, it is meant that it can interoparate (i.e. communicate with) and CORBA v2.x compliant ORB using IIOP (GIOP over TCP/IP). The 'Dyna' in DynaORB symbolizes that no static client stubs are needed, instead DynaORB issues dynamic requests to CORBA servers. Finally, DynaORB is a CORBA client and requires a server-side equivalent. Any CORBA 2.x compliant ORB can be used to publish your object implementations.

ISP

ISP is a CORBA 2 compliant C++ ORB created by the Institute of Systems Programming of the Russian Academy of Sciences (ISPRAS).

Arachne

Arachne is a toolkit for distributed component-based software development. It includes a CORBA ORB which is "nearly" CORBA 2.0 compliant, a partial implementation of COS, an IDL-to-C++ translator, some cross-platform portability libraries, and a CORBA application framework class library. Available for Windows 95/NT, Linux, HP/UX, SunOS 4.x, and Macintosh.

ObjectSpace Voyager

The ObjectSpace Voyager ORB is still free for internal, not-for-resale applications. ObjectSpace has created an entire family of products based on Voyager, including an application server.

TAO

TAO is a freely available implementation of a CORBA Object Request Broker (ORB) developed at Washington University. Portions of it are still under development; you can track the progress here.

Fnorb

Fnorb is a Corba ORB written in the Python language. It includes a language mapping for Python. It's free for non-commercial use. It no longer requires a third-party Interface Repository.

MICO

The MICO project intends to provide a freely available and complete CORBA 2.1 implementation under the GNU public license. Full source code is available. MICO boasts an impressive list:

```
IDL to C++ mapping
Dynamic Invocation Interface (DII)
Dynamic Skeleton Interface (DSI)
graphical Interface Repository browser that allows you to invoke arbitrary methods on
arbitrary interfaces
Interface Repository (IR)
IIOP as native protocol (ORB prepared for multiprotocol support)
Support for nested method invocations
Any offers an interface for inserting and extracting constructed types that were not
known at compile time
Full BOA implementation, including all activation modes, support for object migration
and the implementation repository
BOA can load object implementations into clients at runtime using loadable modules
Support for using MICO from within X11 applications (Xt, Qt, and Gtk) and Tcl/Tk
Naming service
Event service
Relationship service
Dynamic Any
Interceptors
Support
```

OmniORB 2

Omni-ORB 2 is a CORBA 2 - compliant ORB from AT&T Laboratories Cambridge. It supports C++ bindings, and it is freely available.
As of June 1999, the current version is 2.7.1. JavaIDL Javasoft has released an early-access implementation of JavaIDL. This is not the alpha version referred to below, but is listed as version 1.1 EA.

JacORB JacORB is a free ORB written in Java.
Gerald Brose, creator of JacORB, had this to say:

```
would you mind extending your description of JacORB on your free
CORBA page so as mention it

- is 100% Java and supports a Java language mapping
- is GPL'ed
- comes with full source code

If you like, you could also add that it comes with a name and
```

```
an event service implementation.
```

Orbacus (formerly OmniBroker) ORBacus is free for non-commercial use. Source code is also available. A thumbnail sketch of new features:

```
The major changes between the old OmniBroker
and the new ORBacus are full support for multi-threading through the
JThreads/C++ library, an Event-, Naming- and Property-Service (Trading
Service as add-on) and Pluggable Protocols with IIOP as default plug-in
(SSL as add-on).
```

Electra is a CORBA V2 ORB written by Silvano Maffeis. It is available in source code form.

ILU The Inter-Language Unification project from Xerox.

DOME DOME is available free for personal use on the PC-Linux platform.

JORBA This is a work in progress. When complete, it will be a CORBA 2.0-compliant ORB written entirely in Java and published under the GNU public license.

Sun's Java IDL This is a link to Sun's Java IDL page. Sun's Java IDL toolkit is available free for download.

ROBIN

ROBIN is a freeware distributed object system based on CORBA 2.0. Here's what the purveyors have to say about it:

This freeware distributed object system supports a simplified subset of CORBA 2.0 core, with C, C++ and Java language support. It runs on many UNIXes and Win32. It uses an Internet-Protocol- centric interpretation of ORB IDs, which also allows it to provide location-independence and redundancy using IP multicasting. It should serve as a good introduction to the world of multiplatform, object-oriented, distributed computing. Future enhancment plans include: finishing the CORBA core APIs, adding Visual BASIC support, and possibly IIOP support.

Commercial ORBs with free evaluation periods

CorbaPlus

Expersoft has made the Java edititon of their CorbaPlus ORB available for a 60-day time-limited evaluation.

OAK

OAK is a CORBA2-compliant ORB available for a wide variety of platforms, including Nextstep, Openstep, and WebObjects. It supports C++ and Objective-C bindings as well as a Java client-side mapping, IIOP, DII, DSI, and supports naming and events. A full version is available for 30-day evaluation. OAK is also offered free to educational institutions. Additionally, Paragon Software offers a single-user license for personal non-commercial use, also for free.

OrbixWeb

OrbixWeb is a full CORBA2 ORB implemented in Java and is available for download free for 60 days.

CorbaPlus for C++ Expersoft's CorbaPlus for C++ is available for a 60-day trial download.

Visibroker Visibroker is one of the leading commercial CORBA ORBs available Inprise was formed by a merger of Visigenic and Borland. These folks make CORBA development tools for both C++ and Java. They're *not* free, but you can download them for a free trial, so I included them.

Non-CORBA ORBs

HORB

HORB has been around for quite a while (at least in Java terms). Free for commercial use in unmodified binary form.

RMI

RMI comes with JDK 1.1 or better, and a version for JDK 1.0.2 is available from this link.

... and that's all I've got so far; if you hear of any others, please let me know.

Links to other pages on CORBA and distributed computing

ORBNews is a weekly news summary on things ORB-related.

ObjectNews is a daily summary of news and events related to distributed computing, object-oriented programming, patterns, and all manner of other interesting things.

Doug Schmidt's homepage contains a lot of very good information on CORBA.

OMG is the organization that generates the CORBA standards. There is a lot of technical info on this site, including specs for CORBA and its associated services.

Here is another page with links to CORBA software. It's in Austria, so it may take some time to download.

Here is a link to Junichi Suzuki's page on CORBA and distributed computing. It contains yet more links to CORBA implementations, free and otherwise.

Here is a link to Linas Vepstas's page on CORBA implementations for Linux, free and otherwise.

To contact the author of this page, send mail to tvalesky@patriot.net.

Awards!

The Free CORBA page has been awarded a "Key Resource" Award for the topic of CORBA by Links2Go. From their docs, they use some sort of intelligent agent to analyze and cross-reference pages, and pick out the 50 pages that are "most representative" of the topic at hand.

Links²Go
Corba

The Free CORBA page has also been chosen as a Five Star Selection by itmWEB, which maintains a listing of pages that should be useful for information technology professionals.

ROGUE WAVE SOFTWARE

SEARCH
GO

ACCOUNT LOGIN
CUSTOMER ID
PASSWORD
GO

SIGN ME UP FOR AN ACCOUNT!

ROGUE WAVE SOFTWARE, INC.
GLOBAL HEADQUARTERS
5500 Flatiron Parkway
Boulder, Colorado 80301

Tel: 303.473.9118
888.442.9641
Fax: 303.447.2568

The Rogue Wave Assessment Service is a reliable, independent method for solving your development puzzles, providing the expertise needed to help manage your technology initiatives more effectively.

Learn More Today

June 1, 2001 -- Rogue Wave Software Inc., long known as a purveyor of object-oriented C++ component libraries, is shifting its corporate emphasis from packaged software to services.

SD Times

Read More.

Feedback +/-

Help us with our Web site and you could win a Palm VII with a year's supply of palm.net, FREE!

NEWS

June 18, 2001
Rogue Wave Software Announces Stingray Objective Studio 2001

Read More

June 11, 2001
Rogue Wave Appoints Marc Manley as Vice President of Development

Read More

May 30, 2001
Rogue Wave Adds Marc Sternfeld to Board of Directors

Read More

EVENTS

NASA Technology Exposition
NASA Lyndon B. Johnson Space Center
July 17, 2001
Houston, TX

Annual News/400 International Executive Summit Conference
July 22-25, 2001
Vail, Colorado

ITexpo 2001
October 9-11, 2001
Walt Disney World Dolphin
Lake Buena Vista, Florida

complete list of events

---

webmaster@roguewave.com

```c
#include <stdio.h>       /* standard I/O routines                 */
#include <pthread.h>     /* pthread functions and data structures */

/* function to be executed by the new thread */
void*
do_loop(void* data)
{
    int i;                      /* counter, to print numbers */
    int j;                      /* counter, for delay        */
    int me = *((int*)data);     /* thread identifying number */

    for (i=0; i<10; i++) {
        for (j=0; j<500000; j++) /* delay loop */
            ;
        printf("'%d' - Got '%d'\n", me, i);
    }

    /* exit the thread */
    pthread_exit(NULL);
}

/* like any C program, program's execution begins in main */
int
main(int argc, char* argv[])
{
    int        thr_id;         /* thread ID for the newly created thread */
    pthread_t  p_thread;       /* thread's structure                     */
    int        a         = 1;  /* thread 1 identifying number            */
    int        b         = 2;  /* thread 2 identifying number            */

    /* create a new thread that will execute 'do_loop()' */
    thr_id = pthread_create(&p_thread, NULL, do_loop, (void*)&a);
    /* run 'do_loop()' in the main thread as well */
    do_loop((void*)&b);

    /* NOT REACHED */
    return 0;
}
```

```
#include <stdio.h>       /* standard I/O routines                 */
#include <pthread.h>     /* pthread functions and data structures */

#define NUM_EMPLOYEES 2                  /* size of each array.    */

/* global mutex for our program. assignment initializes it */
pthread_mutex_t a_mutex = PTHREAD_MUTEX_INITIALIZER;

struct employee {
    int number;
    int id;
    char first_name[20];
    char last_name[30];
    char department[30];
    int room_number;
};

/* global variable - our employees array, with 2 employees */
struct employee employees[] = {
    { 1, 12345678, "danny", "cohen", "Accounting", 101},
    { 2, 87654321, "moshe", "levy", "Programmers", 202}
};

/* global variable - employee of the day. */
struct employee employee_of_the_day;

/* function to copy one employee struct into another */
void
copy_employee(struct employee* from, struct employee* to)
{
    int rc;     /* contain mutex lock/unlock results */

    /* lock the mutex, to assure exclusive access to 'a' and 'b'. */
    rc = pthread_mutex_lock(&a_mutex);

    to->number = from->number;
    to->id = from->id;
    strcpy(to->first_name, from->first_name);
    strcpy(to->last_name, from->last_name);
    strcpy(to->department, from->department);
    to->room_number = from->room_number;

    /* unlock mutex */
    rc = pthread_mutex_unlock(&a_mutex);
}

/* function to be executed by the variable setting threads thread */
void*
do_loop(void* data)
{
    int my_num = *((int*)data);   /* thread identifying number         */

    while (1) {
        /* set employee of the day to be the one with number 'my_num'. */
        copy_employee(&employees[my_num-1], &employee_of_the_day);
    }
}

/* like any C program, program's execution begins in main */
int
```

```
main(int argc, char* argv[])
{
    int        i;              /* loop counter                          */
    int        thr_id1;        /* thread ID for the first new thread    */
    int        thr_id2;        /* thread ID for the second new thread   */
    pthread_t  p_thread1;      /* first thread's structure              */
    pthread_t  p_thread2;      /* second thread's structure             */
    int        num1      = 1;  /* thread 1 employee number              */
    int        num2      = 2;  /* thread 2 employee number              */
    struct employee eotd;      /* local copy of 'employee of the day'.  */
    struct employee* worker;   /* pointer to currently checked employee */

    /* initialize employee of the day to first 1. */
    copy_employee(&employees[0], &employee_of_the_day);

    /* create a new thread that will execute 'do_loop()' with '1'       */
    thr_id1 = pthread_create(&p_thread1, NULL, do_loop, (void*)&num1);
    /* create a second thread that will execute 'do_loop()' with '2'    */
    thr_id2 = pthread_create(&p_thread2, NULL, do_loop, (void*)&num2);

    /* run a loop that verifies integrity of 'employee of the day' many */
    /* many many times.....                                             */
    for (i=0; i<60000; i++) {
        /* save contents of 'employee of the day' to local 'worker'.    */
        copy_employee(&employee_of_the_day, &eotd);
        worker = &employees[eotd.number-1];

        /* compare employees */
        if (eotd.id != worker->id) {
            printf("mismatching 'id' , %d != %d (loop '%d')\n",
                   eotd.id, worker->id, i);
            exit(0);
        }
        if (strcmp(eotd.first_name, worker->first_name) != 0) {
            printf("mismatching 'first_name' , %s != %s (loop '%d')\n",
                   eotd.first_name, worker->first_name, i);
            exit(0);
        }
        if (strcmp(eotd.last_name, worker->last_name) != 0) {
            printf("mismatching 'last_name' , %s != %s (loop '%d')\n",
                   eotd.last_name, worker->last_name, i);
            exit(0);
        }
        if (strcmp(eotd.department, worker->department) != 0) {
            printf("mismatching 'department' , %s != %s (loop '%d')\n",
                   eotd.department, worker->department, i);
            exit(0);
        }
        if (eotd.room_number != worker->room_number) {
            printf("mismatching 'room_number' , %d != %d (loop '%d')\n",
                   eotd.room_number, worker->room_number, i);
            exit(0);
        }
    }

    printf("Glory, employees contents was always consistent\n");

    return 0;
}
```

```
#include <stdio.h>       /* standard I/O routines                 */
#include <pthread.h>     /* pthread functions and data structures */

#define NUM_EMPLOYEES 2                   /* size of each array.    */

struct employee {
    int number;
    int id;
    char first_name[20];
    char last_name[30];
    char department[30];
    int room_number;
};

/* global variable - our employees array, with 2 employees */
struct employee employees[] = {
    { 1, 12345678, "danny", "cohen", "Accounting", 101},
    { 2, 87654321, "moshe", "levy", "Programmers", 202}
};

/* global variable - employee of the day. */
struct employee employee_of_the_day;

/* function to copy one employee struct into another */
void
copy_employee(struct employee* from, struct employee* to)
{
    to->number = from->number;
    to->id = from->id;
    strcpy(to->first_name, from->first_name);
    strcpy(to->last_name, from->last_name);
    strcpy(to->department, from->department);
    to->room_number = from->room_number;
}

/* function to be executed by the variable setting threads thread */
void*
do_loop(void* data)
{
    int my_num = *((int*)data);   /* thread identifying number         */

    while (1) {
        /* set employee of the day to be the one with number 'my_num'. */
        copy_employee(&employees[my_num-1], &employee_of_the_day);
    }
}

/* like any C program, program's execution begins in main */
int
main(int argc, char* argv[])
{
    int        i;              /* loop counter                          */
    int        thr_id1;        /* thread ID for the first new thread    */
    int        thr_id2;        /* thread ID for the second new thread   */
    pthread_t  p_thread1;      /* first thread's structure              */
    pthread_t  p_thread2;      /* second thread's structure             */
    int        num1      = 1;  /* thread 1 employee number              */
    int        num2      = 2;  /* thread 2 employee number              */
    struct employee eotd;      /* local copy of 'employee of the day'.  */
    struct employee* worker;   /* pointer to currently checked employee */
```

```
    /* initialize employee of the day to first 1. */
    copy_employee(&employees[0], &employee_of_the_day);

    /* create a new thread that will execute 'do_loop()' with '1'       */
    thr_id1 = pthread_create(&p_thread1, NULL, do_loop, (void*)&num1);
    /* create a second thread that will execute 'do_loop()' with '2'    */
    thr_id2 = pthread_create(&p_thread2, NULL, do_loop, (void*)&num2);

    /* run a loop that verifies integrity of 'employee of the day' many */
    /* many many times.....                                             */
    for (i=0; i<60000; i++) {
        /* save contents of 'employee of the day' to local 'worker'.    */
        copy_employee(&employee_of_the_day, &eotd);
        worker = &employees[eotd.number-1];

        /* compare employees */
        if (eotd.id != worker->id) {
            printf("mismatching 'id' , %d != %d (loop '%d')\n",
                   eotd.id, worker->id, i);
            exit(0);
        }
        if (strcmp(eotd.first_name, worker->first_name) != 0) {
            printf("mismatching 'first_name' , %s != %s (loop '%d')\n",
                   eotd.first_name, worker->first_name, i);
            exit(0);
        }
        if (strcmp(eotd.last_name, worker->last_name) != 0) {
            printf("mismatching 'last_name' , %s != %s (loop '%d')\n",
                   eotd.last_name, worker->last_name, i);
            exit(0);
        }
        if (strcmp(eotd.department, worker->department) != 0) {
            printf("mismatching 'department' , %s != %s (loop '%d')\n",
                   eotd.department, worker->department, i);
            exit(0);
        }
        if (eotd.room_number != worker->room_number) {
            printf("mismatching 'room_number' , %d != %d (loop '%d')\n",
                   eotd.room_number, worker->room_number, i);
            exit(0);
        }
    }

    printf("Glory, employees contents was always consistent\n");

    return 0;
}
```

```
#include <stdio.h>       /* standard I/O routines                    */
#include <pthread.h>     /* pthread functions and data structures    */
#include <stdlib.h>      /* rand() and srand() functions             */

/* number of threads used to service requests */
#define NUM_HANDLER_THREADS 3


/* global mutex for our program. assignment initializes it. */
/* note that we use a RECURSIVE mutex, since a handler      */
/* thread might try to lock it twice consecutively.         */
pthread_mutex_t request_mutex = PTHREAD_RECURSIVE_MUTEX_INITIALIZER_NP;

/* global condition variable for our program. assignment initializes it. */
pthread_cond_t  got_request   = PTHREAD_COND_INITIALIZER;

int num_requests = 0;   /* number of pending requests, initially none */

/* format of a single request. */
struct request {
    int number;             /* number of the request                */
    struct request* next;   /* pointer to next request, NULL if none. */
};

struct request* requests = NULL;     /* head of linked list of requests. */
struct request* last_request = NULL; /* pointer to last request.         */

/*
 * function add_request(): add a request to the requests list
 * algorithm: creates a request structure, adds to the list, and
 *            increases number of pending requests by one.
 * input:     request number, linked list mutex.
 * output:    none.
 */
void
add_request(int request_num,
            pthread_mutex_t* p_mutex,
            pthread_cond_t*  p_cond_var)
{
    int rc;                         /* return code of pthreads functions.  */
    struct request* a_request;      /* pointer to newly added request.     */

    /* create structure with new request */
    a_request = (struct request*)malloc(sizeof(struct request));
    if (!a_request) { /* malloc failed?? */
        fprintf(stderr, "add_request: out of memory\n");
        exit(1);
    }
    a_request->number = request_num;
    a_request->next = NULL;

    /* lock the mutex, to assure exclusive access to the list */
    rc = pthread_mutex_lock(p_mutex);

    /* add new request to the end of the list, updating list */
    /* pointers as required */
    if (num_requests == 0) { /* special case - list is empty */
        requests = a_request;
        last_request = a_request;
    }
```

```
    else {
        last_request->next = a_request;
        last_request = a_request;
    }

    /* increase total number of pending requests by one. */
    num_requests++;

#ifdef DEBUG
    printf("add_request: added request with id '%d'\n", a_request->number);
    fflush(stdout);
#endif /* DEBUG */

    /* unlock mutex */
    rc = pthread_mutex_unlock(p_mutex);

    /* signal the condition variable - there's a new request to handle */
    rc = pthread_cond_signal(p_cond_var);
}

/*
 * function get_request(): gets the first pending request from the requests list
 *                         removing it from the list.
 * algorithm: creates a request structure, adds to the list, and
 *            increases number of pending requests by one.
 * input:     request number, linked list mutex.
 * output:    pointer to the removed request, or NULL if none.
 * memory:    the returned request need to be freed by the caller.
 */
struct request*
get_request(pthread_mutex_t* p_mutex)
{
    int rc;                         /* return code of pthreads functions.  */
    struct request* a_request;      /* pointer to request.                 */

    /* lock the mutex, to assure exclusive access to the list */
    rc = pthread_mutex_lock(p_mutex);

    if (num_requests > 0) {
        a_request = requests;
        requests = a_request->next;
        if (requests == NULL) { /* this was the last request on the list */
            last_request = NULL;
        }
        /* decrease the total number of pending requests */
        num_requests--;
    }
    else { /* requests list is empty */
        a_request = NULL;
    }

    /* unlock mutex */
    rc = pthread_mutex_unlock(p_mutex);

    /* return the request to the caller. */
    return a_request;
}

/*
 * function handle_request(): handle a single given request.
```

```
 * algorithm: prints a message stating that the given thread handled
 *            the given request.
 * input:     request pointer, id of calling thread.
 * output:    none.
 */
void
handle_request(struct request* a_request, int thread_id)
{
    if (a_request) {
        printf("Thread '%d' handled request '%d'\n",
               thread_id, a_request->number);
        fflush(stdout);
    }
}

/*
 * function handle_requests_loop(): infinite loop of requests handling
 * algorithm: forever, if there are requests to handle, take the first
 *            and handle it. Then wait on the given condition variable,
 *            and when it is signaled, re-do the loop.
 *            increases number of pending requests by one.
 * input:     id of thread, for printing purposes.
 * output:    none.
 */
void*
handle_requests_loop(void* data)
{
    int rc;                         /* return code of pthreads functions.  */
    struct request* a_request;      /* pointer to a request.               */
    int thread_id = *((int*)data);  /* thread identifying number           */

#ifdef DEBUG
    printf("Starting thread '%d'\n", thread_id);
    fflush(stdout);
#endif /* DEBUG */

    /* lock the mutex, to access the requests list exclusively. */
    rc = pthread_mutex_lock(&request_mutex);

#ifdef DEBUG
    printf("thread '%d' after pthread_mutex_lock\n", thread_id);
    fflush(stdout);
#endif /* DEBUG */

    /* do forever.... */
    while (1) {
#ifdef DEBUG
        printf("thread '%d', num_requests =  %d\n", thread_id, num_requests);
        fflush(stdout);
#endif /* DEBUG */
        if (num_requests > 0) { /* a request is pending */
            a_request = get_request(&request_mutex);
            if (a_request) { /* got a request - handle it and free it */
                /* unlock mutex - so other threads would be able to handle */
                /* other reqeusts waiting in the queue paralelly.          */
                rc = pthread_mutex_unlock(&request_mutex);
                handle_request(a_request, thread_id);
                free(a_request);
                /* and lock the mutex again. */
                rc = pthread_mutex_lock(&request_mutex);
```

```
            }
        }
        else {
            /* wait for a request to arrive. note the mutex will be */
            /* unlocked here, thus allowing other threads access to */
            /* requests list.                                       */
#ifdef DEBUG
            printf("thread '%d' before pthread_cond_wait\n", thread_id);
            fflush(stdout);
#endif /* DEBUG */
            rc = pthread_cond_wait(&got_request, &request_mutex);
            /* and after we return from pthread_cond_wait, the mutex  */
            /* is locked again, so we don't need to lock it ourselves */
#ifdef DEBUG
            printf("thread '%d' after pthread_cond_wait\n", thread_id);
            fflush(stdout);
#endif /* DEBUG */
        }
    }
}

/* like any C program, program's execution begins in main */
int
main(int argc, char* argv[])
{
    int        i;                                /* loop counter          */
    int        thr_id[NUM_HANDLER_THREADS];      /* thread IDs            */
    pthread_t  p_threads[NUM_HANDLER_THREADS];   /* thread's structures   */
    struct timespec delay;                       /* used for wasting time */

    /* create the request-handling threads */
    for (i=0; i<NUM_HANDLER_THREADS; i++) {
        thr_id[i] = i;
        pthread_create(&p_threads[i], NULL, handle_requests_loop,
(void*)&thr_id[i]);
    }

    /* run a loop that generates requests */
    for (i=0; i<600; i++) {
        add_request(i, &request_mutex, &got_request);
        /* pause execution for a little bit, to allow      */
        /* other threads to run and handle some requests.  */
        if (rand() > 3*(RAND_MAX/4)) { /* this is done about 25% of the time */
            delay.tv_sec = 0;
            delay.tv_nsec = 10;
            nanosleep(&delay, NULL);
        }
    }
    /* now wait till there are no more requests to process */
    sleep(5);

    printf("Glory,  we are done.\n");

    return 0;
}
```

```
#include <stdio.h>       /* standard I/O routines                    */
#include <pthread.h>     /* pthread functions and data structures    */
#include <stdlib.h>      /* rand() and srand() functions             */

/* number of threads used to service requests */
#define NUM_HANDLER_THREADS 3


/* global mutex for our program. assignment initializes it. */
/* note that we use a RECURSIVE mutex, since a handler      */
/* thread might try to lock it twice consecutively.         */
pthread_mutex_t request_mutex = PTHREAD_RECURSIVE_MUTEX_INITIALIZER_NP;

/* global condition variable for our program. assignment initializes it. */
pthread_cond_t  got_request   = PTHREAD_COND_INITIALIZER;

int num_requests = 0;   /* number of pending requests, initially none */
/* CHANGE - definition of the flag variable */
int done_creating_requests = 0; /* are we done creating new requests? */

/* format of a single request. */
struct request {
    int number;             /* number of the request                */
    struct request* next;   /* pointer to next request, NULL if none. */
};

struct request* requests = NULL;     /* head of linked list of requests. */
struct request* last_request = NULL; /* pointer to last request.         */

/*
 * function add_request(): add a request to the requests list
 * algorithm: creates a request structure, adds to the list, and
 *            increases number of pending requests by one.
 * input:     request number, linked list mutex.
 * output:    none.
 */
void
add_request(int request_num,
            pthread_mutex_t* p_mutex,
            pthread_cond_t*  p_cond_var)
{
    int rc;                         /* return code of pthreads functions.  */
    struct request* a_request;      /* pointer to newly added request.     */

    /* create structure with new request */
    a_request = (struct request*)malloc(sizeof(struct request));
    if (!a_request) { /* malloc failed?? */
        fprintf(stderr, "add_request: out of memory\n");
        exit(1);
    }
    a_request->number = request_num;
    a_request->next = NULL;

    /* lock the mutex, to assure exclusive access to the list */
    rc = pthread_mutex_lock(p_mutex);

    /* add new request to the end of the list, updating list */
    /* pointers as required */
    if (num_requests == 0) { /* special case - list is empty */
        requests = a_request;
```

```
        last_request = a_request;
    }
    else {
        last_request->next = a_request;
        last_request = a_request;
    }

    /* increase total number of pending requests by one. */
    num_requests++;

#ifdef DEBUG
    printf("add_request: added request with id '%d'\n", a_request->number);
    fflush(stdout);
#endif /* DEBUG */

    /* unlock mutex */
    rc = pthread_mutex_unlock(p_mutex);

    /* signal the condition variable - there's a new request to handle */
    rc = pthread_cond_signal(p_cond_var);
}

/*
 * function get_request(): gets the first pending request from the requests list
 *                         removing it from the list.
 * algorithm: creates a request structure, adds to the list, and
 *            increases number of pending requests by one.
 * input:     request number, linked list mutex.
 * output:    pointer to the removed request, or NULL if none.
 * memory:    the returned request need to be freed by the caller.
 */
struct request*
get_request(pthread_mutex_t* p_mutex)
{
    int rc;                         /* return code of pthreads functions.  */
    struct request* a_request;      /* pointer to request.                 */

    /* lock the mutex, to assure exclusive access to the list */
    rc = pthread_mutex_lock(p_mutex);

    if (num_requests > 0) {
        a_request = requests;
        requests = a_request->next;
        if (requests == NULL) { /* this was the last request on the list */
            last_request = NULL;
        }
        /* decrease the total number of pending requests */
        num_requests--;
    }
    else { /* requests list is empty */
        a_request = NULL;
    }

    /* unlock mutex */
    rc = pthread_mutex_unlock(p_mutex);

    /* return the request to the caller. */
    return a_request;
}
```

```
/*
 * function handle_request(): handle a single given request.
 * algorithm: prints a message stating that the given thread handled
 *            the given request.
 * input:     request pointer, id of calling thread.
 * output:    none.
 */
void
handle_request(struct request* a_request, int thread_id)
{
    if (a_request) {
        printf("Thread '%d' handled request '%d'\n",
               thread_id, a_request->number);
        fflush(stdout);
    }
}

/*
 * function handle_requests_loop(): infinite loop of requests handling
 * algorithm: forever, if there are requests to handle, take the first
 *            and handle it. Then wait on the given condition variable,
 *            and when it is signaled, re-do the loop.
 *            increases number of pending requests by one.
 * input:     id of thread, for printing purposes.
 * output:    none.
 */
void*
handle_requests_loop(void* data)
{
    int rc;                         /* return code of pthreads functions.  */
    struct request* a_request;      /* pointer to a request.               */
    int thread_id = *((int*)data);  /* thread identifying number           */

#ifdef DEBUG
    printf("Starting thread '%d'\n", thread_id);
    fflush(stdout);
#endif /* DEBUG */

    /* lock the mutex, to access the requests list exclusively. */
    rc = pthread_mutex_lock(&request_mutex);

#ifdef DEBUG
    printf("thread '%d' after pthread_mutex_lock\n", thread_id);
    fflush(stdout);
#endif /* DEBUG */

    /* do forever.... */
    while (1) {
#ifdef DEBUG
        printf("thread '%d', num_requests =  %d\n", thread_id, num_requests);
        fflush(stdout);
#endif /* DEBUG */
        if (num_requests > 0) { /* a request is pending */
            a_request = get_request(&request_mutex);
            if (a_request) { /* got a request - handle it and free it */
                /* unlock mutex - so other threads would be able to handle */
                /* other reqeusts waiting in the queue paralelly.          */
                rc = pthread_mutex_unlock(&request_mutex);
                handle_request(a_request, thread_id);
                free(a_request);
```

```
                /* and lock the mutex again. */
                rc = pthread_mutex_lock(&request_mutex);
            }
        }
        else {
            /* CHANGE 2 - the thread checks the flag before waiting */
            /* on the condition variable.                           */
            /* if no new requests are going to be generated, exit.  */
            if (done_creating_requests) {
#ifdef DEBUG
                printf("thread '%d' unlocking mutex before exiting\n",
                       thread_id);
                fflush(stdout);
#endif /* DEBUG */
                pthread_mutex_unlock(&request_mutex);
                printf("thread '%d' exiting\n", thread_id);
                fflush(stdout);
                pthread_exit(NULL);
            }
            else {
#ifdef DEBUG
                printf("thread '%d' going to sleep\n", thread_id);
#endif /* DEBUG */
            }

            /* wait for a request to arrive. note the mutex will be */
            /* unlocked here, thus allowing other threads access to */
            /* requests list.                                       */
#ifdef DEBUG
            printf("thread '%d' before pthread_cond_wait\n", thread_id);
            fflush(stdout);
#endif /* DEBUG */
            rc = pthread_cond_wait(&got_request, &request_mutex);
            /* and after we return from pthread_cond_wait, the mutex  */
            /* is locked again, so we don't need to lock it ourselves */
#ifdef DEBUG
            printf("thread '%d' after pthread_cond_wait\n", thread_id);
            fflush(stdout);
#endif /* DEBUG */
        }
    }
}

/* like any C program, program's execution begins in main */
int
main(int argc, char* argv[])
{
    int        i;                           /* loop counter          */
    int        thr_id[NUM_HANDLER_THREADS];      /* thread IDs            */
    pthread_t  p_threads[NUM_HANDLER_THREADS];   /* thread's structures   */
    struct timespec delay;                  /* used for wasting time */

    /* create the request-handling threads */
    for (i=0; i<NUM_HANDLER_THREADS; i++) {
        thr_id[i] = i;
        pthread_create(&p_threads[i], NULL, handle_requests_loop,
(void*)&thr_id[i]);
    }

    /* run a loop that generates requests */
```

```
    for (i=0; i<600; i++) {
        add_request(i, &request_mutex, &got_request);
        /* pause execution for a little bit, to allow      */
        /* other threads to run and handle some requests.  */
        if (rand() > 3*(RAND_MAX/4)) { /* this is done about 25% of the time */
            delay.tv_sec = 0;
            delay.tv_nsec = 1;
            nanosleep(&delay, NULL);
        }
    }
    /* CHANGE 1 - the main thread modifies the flag     */
    /* to tell its handler threads no new requests will */
    /* be generated.                                    */
    /* notify our threads we're done creating requests. */
    {
        int rc;

        rc = pthread_mutex_lock(&request_mutex);
        done_creating_requests = 1;
        rc = pthread_cond_broadcast(&got_request);
        rc = pthread_mutex_unlock(&request_mutex);
    }

    /* CHANGE 3 - use pthread_join() to wait for all       */
    /* threads to terminate.                               */
    /* now wait till there are no more requests to process */
    for (i=0; i<NUM_HANDLER_THREADS; i++) {
        void* thr_retval;

        pthread_join(p_threads[i], &thr_retval);
    }
    printf("Glory,  we are done.\n");

    return 0;
}
```

```
#include <stdlib.h>      /* malloc() and free()                        */
#include <assert.h>      /* assert()                                   */

#include "requests_queue.h"      /* requests queue functions and structs */


/*
 * function init_requests_queue(): create a requests queue.
 * algorithm: creates a request queue structure, initialize with given
 *            parameters.
 * input:     queue's mutex, queue's condition variable.
 * output:    none.
 */
struct requests_queue*
init_requests_queue(pthread_mutex_t* p_mutex, pthread_cond_t*  p_cond_var)
{
    struct requests_queue* queue =
                (struct requests_queue*)malloc(sizeof(struct requests_queue));
    if (!queue) {
        fprintf(stderr, "out of memory. exiting\n");
        exit(1);
    }
    /* initialize queue */
    queue->requests = NULL;
    queue->last_request = NULL;
    queue->num_requests = 0;
    queue->p_mutex = p_mutex;
    queue->p_cond_var = p_cond_var;

    return queue;
}

/*
 * function add_request(): add a request to the requests list
 * algorithm: creates a request structure, adds to the list, and
 *            increases number of pending requests by one.
 * input:     pointer to queue, request number.
 * output:    none.
 */
void
add_request(struct requests_queue* queue, int request_num)
{
    int rc;                         /* return code of pthreads functions.  */
    struct request* a_request;      /* pointer to newly added request.     */

    /* sanity check - amke sure queue is not NULL */
    assert(queue);

    /* create structure with new request */
    a_request = (struct request*)malloc(sizeof(struct request));
    if (!a_request) { /* malloc failed?? */
        fprintf(stderr, "add_request: out of memory\n");
        exit(1);
    }
    a_request->number = request_num;
    a_request->next = NULL;

    /* lock the mutex, to assure exclusive access to the list */
    rc = pthread_mutex_lock(queue->p_mutex);
```

```
    /* add new request to the end of the list, updating list */
    /* pointers as required */
    if (queue->num_requests == 0) { /* special case - list is empty */
        queue->requests = a_request;
        queue->last_request = a_request;
    }
    else {
        queue->last_request->next = a_request;
        queue->last_request = a_request;
    }

    /* increase total number of pending requests by one. */
    queue->num_requests++;

#ifdef DEBUG
    printf("add_request: added request with id '%d'\n", a_request->number);
    fflush(stdout);
#endif /* DEBUG */

    /* unlock mutex */
    rc = pthread_mutex_unlock(queue->p_mutex);

    /* signal the condition variable - there's a new request to handle */
    rc = pthread_cond_signal(queue->p_cond_var);
}

/*
 * function get_request(): gets the first pending request from the requests list
 *                         removing it from the list.
 * algorithm: creates a request structure, adds to the list, and
 *            increases number of pending requests by one.
 * input:     pointer to requests queue.
 * output:    pointer to the removed request, or NULL if none.
 * memory:    the returned request need to be freed by the caller.
 */
struct request*
get_request(struct requests_queue* queue)
{
    int rc;                         /* return code of pthreads functions.  */
    struct request* a_request;      /* pointer to request.                 */

    /* sanity check - amke sure queue is not NULL */
    assert(queue);

    /* lock the mutex, to assure exclusive access to the list */
    rc = pthread_mutex_lock(queue->p_mutex);

    if (queue->num_requests > 0) {
        a_request = queue->requests;
        queue->requests = a_request->next;
        if (queue->requests == NULL) { /* this was last request on the list */
            queue->last_request = NULL;
        }
        /* decrease the total number of pending requests */
        queue->num_requests--;
    }
    else { /* requests list is empty */
        a_request = NULL;
    }
```

```
    /* unlock mutex */
    rc = pthread_mutex_unlock(queue->p_mutex);

    /* return the request to the caller. */
    return a_request;
}

/*
 * function get_requests_number(): get the number of requests in the list.
 * input:     pointer to requests queue.
 * output:    number of pending requests on the queue.
 */
int
get_requests_number(struct requests_queue* queue)
{
    int rc;                         /* return code of pthreads functions.  */
    int num_requests;               /* temporary, for result.              */

    /* sanity check */
    assert(queue);

    /* lock the mutex, to assure exclusive access to the list */
    rc = pthread_mutex_lock(queue->p_mutex);

    num_requests = queue->num_requests;

    /* unlock mutex */
    rc = pthread_mutex_unlock(queue->p_mutex);

    return num_requests;
}

/*
 * function delete_requests_queue(): delete a requests queue.
 * algorithm: delete a request queue structure, and free all memory it uses.
 * input:     pointer to requests queue.
 * output:    none.
 */
void
delete_requests_queue(struct requests_queue* queue)
{
    struct request* a_request;      /* pointer to a request.               */

    /* sanity check - amke sure queue is not NULL */
    assert(queue);

    /* first free any requests that might be on the queue */
    while (queue->num_requests > 0) {
        a_request = get_request(queue);
        free(a_request);
    }

    /* finally, free the queue's struct itself */
    free(queue);
}
```

```
#include <stdio.h>       /* standard I/O routines                    */
#include <pthread.h>     /* pthread functions and data structures    */
#include <stdlib.h>      /* malloc() and free()                      */
#include <assert.h>      /* assert()                                 */

#include "requests_queue.h"   /* requests queue routines/structs     */
#include "handler_thread.h"   /* handler thread functions/structs    */

extern int done_creating_requests;   /* are we done creating new requests? */

/*
 * function cleanup_free_mutex(): free the mutex, if it's locked.
 * input:     pointer to a mutex structure.
 * output:    none.
 */
static void
cleanup_free_mutex(void* a_mutex)
{
    pthread_mutex_t* p_mutex = (pthread_mutex_t*)a_mutex;

    if (p_mutex)
        pthread_mutex_unlock(p_mutex);
}

/*
 * function handle_request(): handle a single given request.
 * algorithm: prints a message stating that the given thread handled
 *            the given request.
 * input:     request pointer, id of calling thread.
 * output:    none.
 */
static void
handle_request(struct request* a_request, int thread_id)
{
    if (a_request) {
        int i;
        /*
        printf("Thread '%d' handled request '%d'\n",
               thread_id, a_request->number);
        fflush(stdout);
        */
        for (i = 0; i<100000; i++)
            ;
    }
}

/*
 * function handle_requests_loop(): infinite loop of requests handling
 * algorithm: forever, if there are requests to handle, take the first
 *            and handle it. Then wait on the given condition variable,
 *            and when it is signaled, re-do the loop.
 *            increases number of pending requests by one.
 * input:     id of thread, for printing purposes.
 * output:    none.
 */
void*
handle_requests_loop(void* thread_params)
{
    int rc;                         /* return code of pthreads functions.  */
    struct request* a_request;      /* pointer to a request.               */
```

```
    struct handler_thread_params *data;
                                  /* hadler thread's parameters */

    /* sanity check -make sure data isn't NULL */
    data = (struct handler_thread_params*)thread_params;
    assert(data);

    printf("Starting thread '%d'\n", data->thread_id);
    fflush(stdout);

    /* set my cancel state to 'enabled', and cancel type to 'defered'. */
    pthread_setcancelstate(PTHREAD_CANCEL_ENABLE, NULL);
    pthread_setcanceltype(PTHREAD_CANCEL_DEFERRED, NULL);

    /* set thread cleanup handler */
    pthread_cleanup_push(cleanup_free_mutex, (void*)data->request_mutex);

    /* lock the mutex, to access the requests list exclusively. */
    rc = pthread_mutex_lock(data->request_mutex);

#ifdef DEBUG
    printf("thread '%d' after pthread_mutex_lock\n", data->thread_id);
    fflush(stdout);
#endif /* DEBUG */

    /* do forever.... */
    while (1) {
        int num_requests = get_requests_number(data->requests);

#ifdef DEBUG
        printf("thread '%d', num_requests =  %d\n",
               data->thread_id, num_requests);
        fflush(stdout);
#endif /* DEBUG */
        if (num_requests > 0) { /* a request is pending */
            a_request = get_request(data->requests);
            if (a_request) { /* got a request - handle it and free it */
                /* unlock mutex - so other threads would be able to handle */
                /* other reqeusts waiting in the queue paralelly.          */
                rc = pthread_mutex_unlock(data->request_mutex);
                handle_request(a_request, data->thread_id);
                free(a_request);
                /* and lock the mutex again. */
                rc = pthread_mutex_lock(data->request_mutex);
            }
        }
        else {
            /* the thread checks the flag before waiting            */
            /* on the condition variable.                           */
            /* if no new requests are going to be generated, exit.  */
            if (done_creating_requests) {
                pthread_mutex_unlock(data->request_mutex);
                printf("thread '%d' exiting\n", data->thread_id);
                fflush(stdout);
                pthread_exit(NULL);
            }
            /* wait for a request to arrive. note the mutex will be */
            /* unlocked here, thus allowing other threads access to */
            /* requests list.                                       */
#ifdef DEBUG
```

```
            printf("thread '%d' before pthread_cond_wait\n", data->thread_id);
            fflush(stdout);
#endif /* DEBUG */
            rc = pthread_cond_wait(data->got_request, data->request_mutex);
            /* and after we return from pthread_cond_wait, the mutex  */
            /* is locked again, so we don't need to lock it ourselves */
#ifdef DEBUG
            printf("thread '%d' after pthread_cond_wait\n", data->thread_id);
            fflush(stdout);
#endif /* DEBUG */
        }
    }

    /* remove thread cleanup handler. never reached, but we must use */
    /* it here, according to pthread_cleanup_push's manual page.     */
    pthread_cleanup_pop(0);
}
```

```
#include <stdio.h>              /* standard I/O routines                    */
#include <pthread.h>            /* pthread functions and data structures    */
#include <stdlib.h>             /* malloc() and free()                      */
#include <assert.h>      /* assert()                                        */

#include "handler_threads_pool.h" /* handler threads pool functions/structs */

/*
 * create a handler threads pool. associate it with the given mutex
 * and condition variables.
 */
struct handler_threads_pool*
init_handler_threads_pool(pthread_mutex_t* p_mutex,
                          pthread_cond_t*  p_cond_var,
                          struct requests_queue* requests)
{
    struct handler_threads_pool* pool =
      (struct handler_threads_pool*)malloc(sizeof(struct handler_threads_pool));

    if (!pool) {
        fprintf(stderr, "init_handler_threads_pool: out of memory. exiting\n");
        exit(1);
    }
    /* initialize queue */
    pool->threads = NULL;
    pool->last_thread = NULL;
    pool->num_threads = 0;
    pool->max_thr_id = 0;
    pool->p_mutex = p_mutex;
    pool->p_cond_var = p_cond_var;
    pool->requests = requests;

    return pool;
}

/* spawn a new handler thread and add it to the threads pool. */
void
add_handler_thread(struct handler_threads_pool* pool)
{
    struct handler_thread* a_thread;      /* thread's data       */
    struct handler_thread_params* params; /* thread's parameters */

    /* sanity check */
    assert(pool);

    /* create the new thread's structure and initialize it */
    a_thread = (struct handler_thread*)malloc(sizeof(struct handler_thread));
    if (!a_thread) {
        fprintf(stderr, "add_handler_thread: out of memory. exiting\n");
        exit(1);
    }
    a_thread->thr_id = pool->max_thr_id++;
    a_thread->next = NULL;

    /* create the thread's parameters structure */
    params = (struct handler_thread_params*)
                                   malloc(sizeof(struct handler_thread_params));
    if (!params) {
        fprintf(stderr, "add_handler_thread: out of memory. exiting\n");
        exit(1);
```

```
    }
    params->thread_id = a_thread->thr_id;
    params->request_mutex = pool->p_mutex;
    params->got_request = pool->p_cond_var;
    params->requests = pool->requests;

    /* spawn the thread, and place its ID in the thread's structure */
    pthread_create(&a_thread->thread,
                   NULL,
                   handle_requests_loop,
                   (void*)params);

    /* add the thread's structure to the end of the pool's list. */
    if (pool->num_threads == 0) { /* special case - list is empty */
        pool->threads = a_thread;
        pool->last_thread = a_thread;
    }
    else {
        pool->last_thread->next = a_thread;
        pool->last_thread = a_thread;
    }

    /* increase total number of threads by one. */
    pool->num_threads++;
}

/* remove the first thread from the threads pool (do NOT cancel the thread) */
static struct handler_thread*
remove_first_handler_thread(struct handler_threads_pool* pool)
{
    struct handler_thread* a_thread = NULL;     /* temporary holder */

    /* sanity check */
    assert(pool);

    if (pool->num_threads > 0 && pool->threads) {
        a_thread = pool->threads;
        pool->threads = a_thread->next;
        a_thread->next = NULL;
        pool->num_threads--;
    }

    return a_thread;
}

/* delete the first thread from the threads pool (and cancel the thread) */
void
delete_handler_thread(struct handler_threads_pool* pool)
{
    struct handler_thread* a_thread; /* the thread to cancel */

    /* sanity check */
    assert(pool);

    a_thread = remove_first_handler_thread(pool);
    if (a_thread) {
        pthread_cancel(a_thread->thread);
        free(a_thread);
    }
}
```

```
/* get the number of handler threads currently in the threads pool */
int
get_handler_threads_number(struct handler_threads_pool* pool)
{
    /* sanity check */
    assert(pool);

    return pool->num_threads;
}

/*
 * free the resources taken by the given requests queue,
 * and cancel all its threads.
 */
void
delete_handler_threads_pool(struct handler_threads_pool* pool)
{
    void* thr_retval;                  /* thread's return value */
    struct handler_thread* a_thread;  /* one thread's structure */

    /* sanity check */
    assert(pool);

    /* use pthread_join() to wait for all threads to terminate. */
    while (pool->num_threads > 0) {
        a_thread = remove_first_handler_thread(pool);
        assert(a_thread);       /* sanity check */
        pthread_join(a_thread->thread, &thr_retval);
        free(a_thread);
    }
}
```

```
#include <stdio.h>            /* standard I/O routines                     */
#include <pthread.h>          /* pthread functions and data structures     */
#include <stdlib.h>           /* rand() and srand() functions              */
#include <unistd.h>           /* sleep()                                   */
#include <assert.h>           /* assert()                                  */

#include "requests_queue.h"         /* requests queue routines/structs       */
#include "handler_thread.h"         /* handler thread functions/structs      */
#include "handler_threads_pool.h"   /* handler thread list functions/structs */

/* number of initial threads used to service requests, and max number */
/* of handler threads to create during "high pressure" times.         */
#define NUM_HANDLER_THREADS 3
#define MAX_NUM_HANDLER_THREADS 14

/* number of requests on the queue warranting creation of new threads */
#define HIGH_REQUESTS_WATERMARK 15
#define LOW_REQUESTS_WATERMARK 3

/* global mutex for our program. assignment initializes it. */
/* note that we use a RECURSIVE mutex, since a handler      */
/* thread might try to lock it twice consecutively.         */
pthread_mutex_t request_mutex = PTHREAD_RECURSIVE_MUTEX_INITIALIZER_NP;

/* global condition variable for our program. assignment initializes it. */
pthread_cond_t  got_request   = PTHREAD_COND_INITIALIZER;

/* are we done creating new requests? */
int done_creating_requests = 0;

/* like any C program, program's execution begins in main */
int
main(int argc, char* argv[])
{
    int        i;                                /* loop counter          */
    struct timespec delay;                       /* used for wasting time */
    struct requests_queue* requests = NULL;  /* pointer to requests queue */
    struct handler_threads_pool* handler_threads = NULL;
                                            /* list of handler threads */

    /* create the requests queue */
    requests = init_requests_queue(&request_mutex, &got_request);
    assert(requests);

    /* create the handler threads list */
    handler_threads =
        init_handler_threads_pool(&request_mutex, &got_request, requests);
    assert(handler_threads);

    /* create the request-handling threads */
    for (i=0; i<NUM_HANDLER_THREADS; i++) {
        add_handler_thread(handler_threads);
    }

    /* run a loop that generates requests */
    for (i=0; i<600; i++) {
        int num_requests; // number of requests waiting to be handled.
        int num_threads;  // number of active handler threads.

        add_request(requests, i);
```

```
        num_requests = get_requests_number(requests);
        num_threads = get_handler_threads_number(handler_threads);

        /* if there are too many requests on the queue, spawn new threads */
        /* if there are few requests and too many handler threads, cancel */
        /* a handler thread.                                              */
        if (num_requests > HIGH_REQUESTS_WATERMARK &&
            num_threads < MAX_NUM_HANDLER_THREADS) {
                printf("main: adding thread: '%d' requests, '%d' threads\n",
                       num_requests, num_threads);
                add_handler_thread(handler_threads);
        }
        if (num_requests < LOW_REQUESTS_WATERMARK &&
              num_threads > NUM_HANDLER_THREADS) {
            printf("main: deleting thread: '%d' requests, '%d' threads\n",
                   num_requests, num_threads);
            delete_handler_thread(handler_threads);
        }

        /* pause execution for a little bit, to allow      */
        /* other threads to run and handle some requests.  */
        if (rand() > 3*(RAND_MAX/4)) { /* this is done about 25% of the time */
            delay.tv_sec = 0;
            delay.tv_nsec = 1;
            nanosleep(&delay, NULL);
        }
    }
    /* modify the flag to tell the handler threads no   */
    /* new requests will be generated.                  */
    {
        int rc;

        rc = pthread_mutex_lock(&request_mutex);
        done_creating_requests = 1;
        rc = pthread_cond_broadcast(&got_request);
        rc = pthread_mutex_unlock(&request_mutex);
    }

    /* cleanup */
    delete_handler_threads_pool(handler_threads);
    delete_requests_queue(requests);

    printf("Glory,  we are done.\n");

    return 0;
}
```

Index of /~choo/lupg/tutorials/multi-thread/thread-pool-server-changes

- Parent Directory
- Makefile
- handler_thread.c
- handler_thread.h
- handler_threads_pool.c
- handler_threads_pool.h
- main.c
- requests_queue.c
- requests_queue.h

```
#include <stdio.h>             /* standard I/O routines                     */
#include <pthread.h>           /* pthread functions and data structures     */

#define DATA_FILE "very_large_data_file"

/* global mutex for our program. assignment initializes it. */
pthread_mutex_t action_mutex = PTHREAD_MUTEX_INITIALIZER;

/* global condition variable for our program. assignment initializes it. */
pthread_cond_t  action_cond   = PTHREAD_COND_INITIALIZER;

/* flag to denote if the user requested to cancel the operation in the middle */
/* 0 means 'no'. */
int cancel_operation = 0;

/*
 * function: restore_coocked_mode - restore normal screen mode.
 * algorithm: uses the 'stty' command to restore normal screen mode.
 *            serves as a cleanup function for the user input thread.
 * input: none.
 * output: none.
 */

void
restore_coocked_mode(void* dummy)
{
#ifdef DEBUG
    printf("restore_coocked_mode: before 'stty -raw echo'\n\r");
    fflush(stdout);
#endif /* DEBUG */
    system("stty -raw echo");
#ifdef DEBUG
    printf("restore_coocked_mode: after 'stty -raw echo'\n\r");
    fflush(stdout);
#endif /* DEBUG */
}

/*
 * function: read_user_input - read user input while long operation in progress.
 * algorithm: put screen in raw mode (without echo), to allow for unbuffered
 *            input.
 *            perform an endless loop of reading user input. If user
 *            pressed 'e', signal our condition variable and end the thread.
 * input: none.
 * output: none.
 */
void*
read_user_input(void* data)
{
    int c;

    /* register cleanup handler */
    pthread_cleanup_push(restore_coocked_mode, NULL);

    /* make sure we're in asynchronous cancelation mode so   */
    /* we can be canceled even when blocked on reading data. */
    pthread_setcanceltype(PTHREAD_CANCEL_ASYNCHRONOUS, NULL);

    /* put screen in raw data mode */
    system("stty raw -echo");
```

```
    /* "endless" loop - read data from the user.            */
    /* terminate the loop if we got a 'e', or are canceled. */
    while ((c = getchar()) != EOF) {
        if (c == 'e') {
#ifdef DEBUG
            printf("\n\ngot a 'e'\n\n\r");
            fflush(stdout);
#endif /* DEBUG */
            /* mark that there was a cancel request by the user */
            cancel_operation = 1;
            /* signify that we are done */
            pthread_cond_signal(&action_cond);
            pthread_exit(NULL);
        }
    }

    /* pop cleanup handler, while executing it, to restore cooked mode. */
    pthread_cleanup_pop(1);
}

/*
 * function: file_line_count - counts the number of lines in the given file.
 * algorithm: open the data file, read it character by character, and count
 *            all newline characters.
 * input: file name.
 * output: number of lines in the given file.
 */
void*
file_line_count(void* data)
{
    char* data_file = (char*)data;
    FILE* f = fopen(data_file, "r");
    int wc = 0;
    int c;

    if (!f) {
        perror("fopen");
        exit(1);
    }

    /* make sure we're in asynchronous cancelation mode so   */
    /* we can be canceled even when blocked on reading data. */
    pthread_setcanceltype(PTHREAD_CANCEL_ASYNCHRONOUS, NULL);

    while ((c = getc(f)) != EOF) {
        if (c == '\n')
            wc++;
    }

    fclose(f);

    /* signify that we are done. */
    pthread_cond_signal(&action_cond);

    return (void*)wc;
}

/* like any C program, program's execution begins in main */
int
```

```
main(int argc, char* argv[])
{
    pthread_t thread_line_count; /* 'handle' of line-counting thread.        */
    pthread_t thread_user_input; /* 'handle' of user-input thread.           */
    void* line_count;            /* return value from line-counting thread.  */

    printf("Checking file size (press 'e' to cancel operation)...");
    fflush(stdout);

    /* spawn the line counting thread */
    pthread_create(&thread_line_count,
                   NULL,
                   file_line_count,
                   (void*)DATA_FILE);
    /* spawn the user-reading thread */
    pthread_create(&thread_user_input,
                   NULL,
                   read_user_input,
                   (void*)DATA_FILE);

    /* lock the mutex, and wait on the condition variable, */
    /* till one of the threads finishes up and signals it. */
    pthread_mutex_lock(&action_mutex);
    pthread_cond_wait(&action_cond, &action_mutex);
    pthread_mutex_unlock(&action_mutex);

#ifdef DEBUG
    printf("\n\rmain: we got signaled\n\n\r");
    fflush(stdout);
#endif /* DEBUG */

    /* check if we were signaled due to user operation        */
    /* cancelling, or because the line-counting was finished. */
    if (cancel_operation) {
        /* we join it to make sure it restores normal */
        /* screen mode before we print out.           */
        pthread_join(thread_user_input, NULL);
        printf("operation canceled\n");
        fflush(stdout);
        /* cancel the file-checking thread */
        pthread_cancel(thread_line_count);
    }
    else {
        /* join the file line-counting thread, to get its results */
        pthread_join(thread_line_count, &line_count);

        /* cancel and join the user-input thread.     */
        /* we join it to make sure it restores normal */
        /* screen mode before we print out.           */
        pthread_cancel(thread_user_input);
        pthread_join(thread_user_input, NULL);

        /* and print the result */
        printf("'%d' lines.\n", (int)line_count);
    }

    return 0;
}
```

Links[2]*Go*
Gnu

Release 1.0 is now available! Go get it!

Welcome

Here at the University of Kansas we have been working on a debugger that is both scriptable and thread-aware. This project is designed to alleviate many of the problems of modern debugging. Below is a set of links to documents that describe various aspects of the project.

What Is SmartGDB

This is a brief (1 page) summary of what SmartGDB is.

Introduction

This document provides an introduction to SmartGDB and an overview of the driving problems SmartGDB attempts to solve.

Previous Work

This document details the previous work done on SmartGDB by other students.

Working With SmartGDB

This document presents a set of debugging results obtained with SmartGDB.

Download The Release

This is the download page. The current release is version 4.16.2 (SmartGDB 1.0). Instructions are provided on how to obtain and install the release.

Documentation

These pages provide documentation on the SmartGDB thread interface and on using Tcl/Tk inside SmartGDB.

Credits

Credits for SmartGDB go to...

```
/*
 * one-way-pipe.c  - example of using a pipe to communicate data between a
 *                   process and its child process. The parent reads input
 *                   from the user, and sends it to the child via a pipe.
 *                   The child prints the received data to the screen.
 */

#include <stdio.h>    /* standard I/O routines.                 */
#include <unistd.h>   /* defines pipe(), amongst other things. */

/* this routine handles the work of the child process. */
void do_child(int data_pipe[]) {
    int c;      /* data received from the parent. */
    int rc;     /* return status of read().       */

    /* first, close the un-needed write-part of the pipe. */
    close(data_pipe[1]);

    /* now enter a loop of reading data from the pipe, and printing it */
    while ((rc = read(data_pipe[0], &c, 1)) > 0) {
        putchar(c);
    }

    /* probably pipe was broken, or got EOF via the pipe. */
    exit(0);
}

/* this routine handles the work of the parent process. */
void do_parent(int data_pipe[])
{
    int c;      /* data received from the user. */
    int rc;     /* return status of getchar().  */

    /* first, close the un-needed read-part of the pipe. */
    close(data_pipe[0]);

    /* now enter a loop of read user input, and writing it to the pipe. */
    while ((c = getchar()) > 0) {
        /* write the character to the pipe. */
        rc = write(data_pipe[1], &c, 1);
        if (rc == -1) { /* write failed - notify the user and exit */
            perror("Parent: write");
            close(data_pipe[1]);
            exit(1);
        }
    }

    /* probably got EOF from the user. */
    close(data_pipe[1]); /* close the pipe, to let the child know we're done. */
    exit(0);
}

/* and the main function. */
int main(int argc, char* argv[])
{
    int data_pipe[2]; /* an array to store the file descriptors of the pipe. */
    int pid;       /* pid of child process, or 0, as returned via fork.    */
    int rc;        /* stores return values of various routines.            */

    /* first, create a pipe. */
```

```
    rc = pipe(data_pipe);
    if (rc == -1) {
        perror("pipe");
        exit(1);
    }

    /* now fork off a child process, and set their handling routines. */
    pid = fork();

    switch (pid) {
        case -1:        /* fork failed. */
            perror("fork");
            exit(1);
        case 0:         /* inside child process.  */
            do_child(data_pipe);
            /* NOT REACHED */
        default:        /* inside parent process. */
            do_parent(data_pipe);
            /* NOT REACHED */
    }

    return 0;   /* NOT REACHED */
}
```

```
/*
 * two-way-pipe.c  - two processes communicating both ways by using two
 *                   pipes. One process reads input from the user and handles
 *                   it. The other process makes some translation of the
 *                   input (translates upper-case letters to lower-case),
 *                   and hands it back to the first process for printing.
 */

#include <stdio.h>    /* standard I/O routines.                 */
#include <unistd.h>   /* defines pipe(), amongst other things.  */
#include <ctype.h>    /* defines isascii(), toupper(), and other */
                      /* character manipulation routines.       */

/* function executed by the user-interacting process. */
void user_handler(int input_pipe[], int output_pipe[])
{
    int c;    /* user input */
    int rc;   /* return values of functions. */

    /* first, close unnecessary file descriptors */
    close(input_pipe[1]);  /* we don't need to write to this pipe.  */
    close(output_pipe[0]); /* we don't need to read from this pipe. */

    /* loop: read input from user, send via one pipe to the translator, */
    /* read via other pipe what the translator returned, and write to   */
    /* stdout. exit on EOF from user.                                   */
    while ((c = getchar()) > 0) {
        /* write to translator */
        rc = write(output_pipe[1], &c, 1);
        if (rc == -1) { /* write failed - notify the user and exit. */
            perror("user_handler: write");
            close(input_pipe[0]);
            close(output_pipe[1]);
            exit(1);
        }
        /* read back from translator */
        rc = read(input_pipe[0], &c, 1);
        if (rc <= 0) { /* read failed - notify user and exit. */
            perror("user_handler: read");
            close(input_pipe[0]);
            close(output_pipe[1]);
            exit(1);
        }
        /* print translated character to stdout. */
        putchar(c);
    }

    /* close pipes and exit. */
    close(input_pipe[0]);
    close(output_pipe[1]);
    exit(0);
}

/* now comes the function executed by the translator process. */
void translator(int input_pipe[], int output_pipe[])
{
    int c;    /* user input */
    int rc;   /* return values of functions. */

    /* first, close unnecessary file descriptors */
```

```
    close(input_pipe[1]); /* we don't need to write to this pipe.  */
    close(output_pipe[0]); /* we don't need to read from this pipe. */

    /* enter a loop of reading from the user_handler's pipe, translating */
    /* the character, and writing back to the user handler.              */
    while (read(input_pipe[0], &c, 1) > 0) {
        /* translate any upper-case letter to lowr-case. */
        if (isascii(c) && isupper(c))
            c = tolower(c);
        /* write translated character back to user_handler. */
        rc = write(output_pipe[1], &c, 1);
        if (rc == -1) { /* write failed - notify user and exit. */
            perror("translator: write");
            close(input_pipe[0]);
            close(output_pipe[1]);
            exit(1);
        }
    }

    /* close pipes and exit. */
    close(input_pipe[0]);
    close(output_pipe[1]);
    exit(0);
}

/* and finally, the main function: spawn off two processes, */
/* and let each of them execute its function.               */
int main(int argc, char* argv[])
{
    /* 2 arrays to contain file descriptors, for two pipes. */
    int user_to_translator[2];
    int translator_to_user[2];
    int pid;       /* pid of child process, or 0, as returned via fork.    */
    int rc;        /* stores return values of various routines.            */

    /* first, create one pipe. */
    rc = pipe(user_to_translator);
    if (rc == -1) {
        perror("main: pipe user_to_translator");
        exit(1);
    }
    /* then, create another pipe. */
    rc = pipe(translator_to_user);
    if (rc == -1) {
        perror("main: pipe translator_to_user");
        exit(1);
    }

    /* now fork off a child process, and set their handling routines. */
    pid = fork();

    switch (pid) {
        case -1:        /* fork failed. */
            perror("main: fork");
            exit(1);
        case 0:         /* inside child process.  */
            translator(user_to_translator, translator_to_user); /* line 'A' */
            /* NOT REACHED */
        default:        /* inside parent process. */
            user_handler(translator_to_user, user_to_translator); /* line 'B' */
```

```
            /* NOT REACHED */
    }

    return 0;   /* NOT REACHED */
}
```

```
/*
 * named-pipe-plan.c  - an example of using a named pipe, in order to make
 *                      the finger daemon tell whoever fingers our account,
 *                      how many times we were fingered recently.
 *                      uses a named pipe instead of a regular '.plan' file.
 */

#include <stdio.h>
#include <unistd.h>

#define PLAN_FILE "/home/choo/.plan"    /* full path to my '.plan' file */

void main()
{
    FILE* plan;
    int count = 0;


    /* run an infinite loop of opening the named pipe, writing */
    /* our message into it, and closing it.                    */
    while (1) {
        /* open the '.plan' file. This blocks until someone,       */
        /* (such as the finger daemon) opens the file for reading. */
        plan = fopen(PLAN_FILE, "w");
        if (!plan) {
            perror("fopen");
            exit(1);
        }

        count++;
        fprintf(plan, "I have been fingered %d times today\n", count);

        /* close the named pipe. This will cause the reader to get an EOF */
        fclose(plan);

        /* suspend execution, give the reader process enough time to */
        /* read the message, get the EOF sign, and close the pipe.   */
        /* otherwise, we might get to re-open the pipe before the    */
        /* reader closes it, and thus write several messages to the  */
        /* same reader.                                              */
        sleep(1);
    }
}
```

```
/*
 * private-queue-hello-world.c  - a "hello world" example in which a single
 *                                process creates a message queue and sends
 *                                itself a "hello world" message via this queue.
 */

#include <stdio.h>
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/msg.h>

int main(int argc, char* argv[])
{
    /* create a private message queue, with access only to the owner. */
    int queue_id = msgget(IPC_PRIVATE, 0600);
    struct msgbuf* msg;
    struct msgbuf* recv_msg;
    int rc;

    if (queue_id == -1) {
        perror("main: msgget");
        exit(1);
    }
    printf("message queue created, queue id '%d'.\n", queue_id);
    msg = (struct msgbuf*)malloc(sizeof(struct msgbuf)+strlen("hello world"));
    msg->mtype = 1;
    strcpy(msg->mtext, "hello world");
    rc = msgsnd(queue_id, msg, strlen(msg->mtext)+1, 0);
    if (rc == -1) {
        perror("main: msgsnd");
        exit(1);
    }
    free(msg);
    printf("message placed on the queue successfully.\n");
    recv_msg = (struct msgbuf*)malloc(sizeof(struct msgbuf)+strlen("hello
world"));
    rc = msgrcv(queue_id, recv_msg, strlen("hello world")+1, 0, 0);
    if (rc == -1) {
        perror("main: msgrcv");
        exit(1);
    }

    printf("msgrcv: received message: mtype '%d'; mtext '%s'\n",
           recv_msg->mtype, recv_msg->mtext);

    return 0;
}
```

```c
/*
 * queue_sender.c - a program that reads messages with one of 3 identifiers
 *                  to a message queue.
 */

#include <stdio.h>       /* standard I/O functions.              */
#include <stdlib.h>      /* malloc(), free() etc.                */
#include <sys/types.h>   /* various type definitions.            */
#include <sys/ipc.h>     /* general SysV IPC structures          */
#include <sys/msg.h>     /* message queue functions and structs. */

#include "queue_defs.h"  /* definitions shared by both programs  */


int main(int argc, char* argv[])
{
    int queue_id;             /* ID of the created queue.            */
    struct msgbuf* msg;       /* structure used for sent messages.   */
    /*struct msgbuf* recv_msg;*/
    int i;                    /* loop counter                        */
    int rc;                   /* error code retuend by system calls. */

    /* create a public message queue, with access only to the owning user. */
    queue_id = msgget(QUEUE_ID, IPC_CREAT | IPC_EXCL | 0600);
    if (queue_id == -1) {
        perror("main: msgget");
        exit(1);
    }
    printf("message queue created, queue id '%d'.\n", queue_id);
    msg = (struct msgbuf*)malloc(sizeof(struct msgbuf)+MAX_MSG_SIZE);

    /* form a loop of creating messages and sending them. */
    for (i=1; i <= NUM_MESSAGES; i++) {
        msg->mtype = (i % 3) + 1; /* create message type between '1' and '3' */
        sprintf(msg->mtext, "hello world - %d", i);
        rc = msgsnd(queue_id, msg, strlen(msg->mtext)+1, 0);
        if (rc == -1) {
            perror("main: msgsnd");
            exit(1);
        }
    }
    /* free allocated memory. */
    free(msg);

    printf("generated %d messages, exiting.\n", NUM_MESSAGES);

    return 0;
}
```

```
/*
 * queue_reader.c - a program that reads messages with a given identifier
 *                  off of a message queue.
 */

#include <stdio.h>       /* standard I/O functions.              */
#include <stdlib.h>      /* malloc(), free() etc.                */
#include <unistd.h>      /* sleep(), etc.                        */
#include <sys/types.h>   /* various type definitions.            */
#include <sys/ipc.h>     /* general SysV IPC structures          */
#include <sys/msg.h>     /* message queue functions and structs. */

#include "queue_defs.h"  /* definitions shared by both programs  */


void main(int argc, char* argv[])
{
    int queue_id;             /* ID of the created queue.              */
    struct msgbuf* msg;       /* structure used for received messages. */
    int rc;                   /* error code returned by system calls.  */
    int msg_type;             /* type of messages we want to receive.  */

    /* read message type from command line */
    if (argc != 2) {
        fprintf(stderr, "Usage: %s <message type>\n", argv[0]);
        fprintf(stderr, "       <message type> must be between 1 and 3.\n");
        exit(1);
    }
    msg_type = atoi(argv[1]);
    if (msg_type < 1 || msg_type > 3) {
        fprintf(stderr, "Usage: %s <message type>\n", argv[0]);
        fprintf(stderr, "       <message type> must be between 1 and 3.\n");
        exit(1);
    }

    /* access the public message queue that the sender program created. */
    queue_id = msgget(QUEUE_ID, 0);
    if (queue_id == -1) {
        perror("main: msgget");
        exit(1);
    }
    printf("message queue opened, queue id '%d'.\n", queue_id);
    msg = (struct msgbuf*)malloc(sizeof(struct msgbuf)+MAX_MSG_SIZE);

    /* form a loop of receiving messages and printing them out. */
    while (1) {
        rc = msgrcv(queue_id, msg, MAX_MSG_SIZE+1, msg_type, 0);
        if (rc == -1) {
            perror("main: msgrcv");
            exit(1);
        }
        printf("Reader '%d' read message: '%s'\n", msg_type, msg->mtext);
        /* slow down a little... */
        sleep(1);
    }
    /* NOT REACHED */
}
```

Index of /~choo/lupg/tutorials/multi-process/public-queue

- Parent Directory
- Makefile
- queue_defs.h
- queue_reader.c
- queue_sender.c

```
message sender/reader - a message queue example
-----------------------------------------------

This directory contains the source code for two programs - queue_sender,
that sends messages of different types to message queues, and queue_reader,
that reads messages of a given type from the same message queue.

To compile the programs, type 'make' (or 'gmake', to use gnu make).

Experiment with running these programs. The queue_sender will create 100
messages of types 1, 2 or 3, and then exit. The queue_reader may then
be run with a parameter denoting the type of messages it should read
(1, 2 or 3). Thus, in order to read out all messages, you should run three
copies of queue_reader, one with each message type. You may also try to
run more then one copy of the reader program for a given message type, and see
how they consume the messages.

Check what happens when you run the reader before running the sender.

You may also check the message queue created by the sender program,
by issuing the following command:

ipcs -q

If done after running the sender, and before running the reader, you will
see something similar to this:

------ Message Queues --------
msqid     owner     perms     used-bytes  messages
257       choo      600       1692        100

note that the 'msgid' field contains a value that is different from what
was supplied to the 'msgget()' system call by the sender program. This is
the value that you should use when trying to remove the message
queue, using the 'ipcrm' program.

Try to check the message queue size after when killing the queue_reader
program in the middle of the run (using Ctrl-C, or the 'kill' command).
```

```
/*
 * sem-mutex.c  - demonstrates the usage of a semaphore as a mutex that
 *                that synchronizes accesses of multiple processes
 *                to a file.
 */

#include <stdio.h>       /* standard I/O routines.              */
#include <stdlib.h>      /* rand() and srand() functions        */
#include <unistd.h>      /* fork(), etc.                        */
#include <time.h>        /* nanosleep(), etc.                   */
#include <sys/types.h>   /* various type definitions.           */
#include <sys/ipc.h>     /* general SysV IPC structures         */
#include <sys/sem.h>     /* semaphore functions and structs.    */
#include <sys/wait.h>    /* wait(), etc.                        */

#define NUM_PROCS  5     /* number of processes to launch.      */
#define SEM_ID    250    /* ID for the semaphore.               */
#define FILE_NAME "sem_mutex"  /* name of file to manipulate    */
#define DELAY     400000 /* delay between file updates by one process. */


/* this function updates the contents of the file with the given path name. */
void update_file(int sem_set_id, char* file_path, int number)
{
    /* structure for semaphore operations.   */
    struct sembuf sem_op;
    FILE* file;

    /* wait on the semaphore, unless it's value is non-negative. */
    sem_op.sem_num = 0;
    sem_op.sem_op = -1;   /* <-- Comment 1 */
    sem_op.sem_flg = 0;
    semop(sem_set_id, &sem_op, 1);

    /* Comment 2 */
    /* we "locked" the semaphore, and are assured exclusive access to file.  */
    /* manipulate the file in some way. for example, write a number into it. */
    file = fopen(file_path, "w");
    if (file) {
        fprintf(file, "%d\n", number);
        printf("%d\n", number);
        fclose(file);
    }

    /* finally, signal the semaphore - increase its value by one. */
    sem_op.sem_num = 0;
    sem_op.sem_op = 1;   /* <-- Comment 3 */
    sem_op.sem_flg = 0;
    semop(sem_set_id, &sem_op, 1);
}

/* this function calls "file_update" several times in a row,  */
/* and waiting a little time between each two calls, in order */
/* to allow other processes time to operate.                  */
void do_child_loop(int sem_set_id, char* file_name)
{
    pid_t pid = getpid();
    int i, j;

    for (i=0; i<3; i++) {
```

```
        update_file(sem_set_id, file_name, pid);
        for (j=0; j<400000; j++)
            ;
    }
}

/* finally, the main() function. */
void main()
{
    int sem_set_id;           /* ID of the semaphore set.       */
    union semun sem_val;      /* semaphore value, for semctl(). */
    int child_pid;            /* PID of our child process.      */
    int i;                    /* counter for loop operation.    */
    int rc;                   /* return value of system calls.  */

    /* create a semaphore set with ID 250, with one semaphore   */
    /* in it, with access only to the owner.                    */
    sem_set_id = semget(SEM_ID, 1, IPC_CREAT | 0600);
    if (sem_set_id == -1) {
        perror("main: semget");
        exit(1);
    }

    /* intialize the first (and single) semaphore in our set to '1'. */
    sem_val.val = 1;
    rc = semctl(sem_set_id, 0, SETVAL, sem_val);
    if (rc == -1) {
        perror("main: semctl");
        exit(1);
    }

    /* create a set of child processes that will compete on the semaphore */
    for (i=0; i<NUM_PROCS; i++) {
        child_pid = fork();
        switch(child_pid) {
            case -1:
                perror("fork");
                exit(1);
            case 0:  /* we're at child process. */
                do_child_loop(sem_set_id, FILE_NAME);
                exit(0);
            default: /* we're at parent process. */
                break;
        }
    }

    /* wait for all children to finish running */
    for (i=0; i<NUM_PROCS; i++) {
        int child_status;

        wait(&child_status);
    }

    printf("main: we're done\n");
    fflush(stdout);
}
```

```
/*
 * sem-producer-consumer.c  - demonstrates a basic producer-consumer
 *                            implementation.
 */

#include <stdio.h>       /* standard I/O routines.              */
#include <stdlib.h>      /* rand() and srand() functions        */
#include <unistd.h>      /* fork(), etc.                        */
#include <time.h>        /* nanosleep(), etc.                   */
#include <sys/types.h>   /* various type definitions.           */
#include <sys/ipc.h>     /* general SysV IPC structures         */
#include <sys/sem.h>     /* semaphore functions and structs.    */

#define NUM_LOOPS       20       /* number of loops to perform. */

int main(int argc, char* argv[])
{
    int sem_set_id;           /* ID of the semaphore set.       */
    union semun sem_val;      /* semaphore value, for semctl(). */
    int child_pid;            /* PID of our child process.      */
    int i;                    /* counter for loop operation.    */
    struct sembuf sem_op;     /* structure for semaphore ops.   */
    int rc;                   /* return value of system calls.  */
    struct timespec delay;    /* used for wasting time.         */

    /* create a private semaphore set with one semaphore in it, */
    /* with access only to the owner.                           */
    sem_set_id = semget(IPC_PRIVATE, 1, 0600);
    if (sem_set_id == -1) {
        perror("main: semget");
        exit(1);
    }
    printf("semaphore set created, semaphore set id '%d'.\n", sem_set_id);

    /* intialize the first (and single) semaphore in our set to '0'. */
    sem_val.val = 0;
    rc = semctl(sem_set_id, 0, SETVAL, sem_val);

    /* fork-off a child process, and start a producer/consumer job. */
    child_pid = fork();
    switch (child_pid) {
        case -1:        /* fork() failed */
            perror("fork");
            exit(1);
        case 0:         /* child process here */
            for (i=0; i<NUM_LOOPS; i++) {
                /* block on the semaphore, unless it's value is non-negative. */
                sem_op.sem_num = 0;
                sem_op.sem_op = -1;
                sem_op.sem_flg = 0;
                semop(sem_set_id, &sem_op, 1);
                printf("consumer: '%d'\n", i);
                fflush(stdout);
            }
            break;
        default:        /* parent process here */
            for (i=0; i<NUM_LOOPS; i++) {
                printf("producer: '%d'\n", i);
                fflush(stdout);
                /* increase the value of the semaphore by 1. */
```

```
                sem_op.sem_num = 0;
                sem_op.sem_op = 1;
                sem_op.sem_flg = 0;
                semop(sem_set_id, &sem_op, 1);
                /* pause execution for a little bit, to allow the */
                /* child process to run and handle some requests. */
                /* this is done about 25% of the time.            */
                if (rand() > 3*(RAND_MAX/4)) {
                    delay.tv_sec = 0;
                    delay.tv_nsec = 10;
                    nanosleep(&delay, NULL);
                }
            }
            break;
    }

    return 0;
}
```

```
/*
 * tiny-lpr.c  - a tiny printing command. Gets a file path, and copies that
 *               file to the spool directory, from which it'll be later
 *               fetched by a printer daemon.
 */

#include <stdio.h>           /* standard I/O routines.                     */
#include <sys/types.h>       /* various type definitions.                  */
#include <sys/ipc.h>         /* general SysV IPC structures                */
#include <sys/sem.h>         /* semaphore functions and structs.           */
#include <unistd.h>          /* access(), etc.                             */
#include <malloc.h>          /* malloc(), etc.                             */
#include <string.h>          /* strlen(), etc.                             */
#include <stdlib.h>          /* atoi(), etc.                               */
#include <fcntl.h>           /* options for open().                        */

#include "tiny-lp-common.h"  /* common tiny spooler definitions/functions. */

/*
 * function: make_filename. creates a unique file name for the spool directory.
 * input:    none.
 * output:   a unique file name.
 */
static char*
make_filename()
{
    int counter;               /* contents of counter file, as integer. */
    static char buf[200];      /* contents of counter file, as string.  */
    int f_counter;             /* pointer to counter file.              */

    f_counter = open(SPOOL_COUNTER, O_RDWR | O_EXCL);
    if (f_counter < 0) {
        fprintf(stderr, "Cannot open file '%s': ", SPOOL_COUNTER);
        perror("");
        exit(1);
    }
    read(f_counter, buf, 199);
    counter = atoi(buf);
    counter++;
    sprintf(buf, "%d", counter);
    lseek(f_counter, 0, SEEK_SET);
    write(f_counter, buf, strlen(buf));
    close(f_counter);

    return buf;
}

/*
 * function: copyfile. copies a file from one place to a new directory,
 *           under a new, unique file name.
 * input:    path to the file to copy, and path to the directory to copy
 *           it into.
 * output:   name of the copy of the file.
 */
static char*
copyfile(char* file_path, char* dir_path)
{
    char* tmp_file;                    /* name for temporary file.  */
    FILE* f_in;                        /* pointer for source file.  */
    FILE* f_out;                       /* pointer for target file.  */
```

```
    char* out_file;                     /* full path to putput file. */
    int c;                              /* character read/written.   */

    /* generate a unique file name. */
    tmp_file = make_filename();
    if (tmp_file == NULL) {
        perror("tmpnam: ");
        exit(1);
    }
    out_file = (char*)malloc(strlen(dir_path)+strlen("/")+strlen(tmp_file)+1);
    if (!out_file) {
        fprintf(stderr, "out of memory\n");
        exit(1);
    }
    sprintf(out_file, "%s/%s", dir_path, tmp_file);

    /* open the input file for reading. open the temporary file for output. */
    f_in = fopen(file_path, "r");
    if (!f_in) {
        fprintf(stderr, "Cannot open file '%s' for reading: \n", file_path);
        perror("");
        exit(1);
    }
    f_out = fopen(out_file, "w");
    if (!f_out) {
        fprintf(stderr, "Cannot open file '%s' for output: \n", out_file);
        perror("");
        exit(1);
    }

    /* perform the file copying operation. This is not very efficient, */
    /* and should be done using large buffer copies instead.           */
    while ( (c=fgetc(f_in)) != EOF)
        fputc(c, f_out);

    fclose(f_in);
    fclose(f_out);

    return tmp_file;
}

/*
 * function: main. perform the whole operation.
 * intput:   path to file to copy into spool directory.
 * output:   messages regarding success or failure of the operation.
 */
void
main(int argc, char* argv[])
{
    int sem_set_id;               /* ID of the semaphore set.             */
    char* file_path;              /* path to the file to print.           */
    char* tmp_file;               /* name of copied file in spool area.   */
    int rc;                       /* return value of system calls.        */
    char tmp_file_path[MAX_PATH]; /* name of copied file in 'in' area.    */
    char tmp_file_out[MAX_PATH];  /* name of copied file in 'common' area. */

    /* check command line arguments. */
    if (argc != 2) {
        fprintf(stderr, "Usage: %s <file path>\n", argv[0]);
        exit(1);
```

```
    }
    file_path = argv[1];

    /* check that the file exists, and that we can read it. */
    if (access(file_path, F_OK) == -1) {
        fprintf(stderr, "%s: file '%s' does not exist.\n", argv[0], file_path);
        exit(1);
    }
    if (access(file_path, R_OK) == -1) {
        fprintf(stderr, "%s: file '%s': cannot access.\n", argv[0], file_path);
        exit(1);
    }

    /* copy the file to our 'in' spool directory. */
    tmp_file = copyfile(file_path, SPOOL_DIR_IN);
    printf("%s: file copied to spool area\n", argv[0]);

    /* get the ID of the semaphore set. */
    /* the '0' means we won't create the set if it was */
    /* not creted by tiny-lpd first.                   */
    sem_set_id = sem_set_get("lpr", 0);
    if (sem_set_id == -1) {
        fprintf(stderr, "%s: warning: tiny-lpd is not running.\n", argv[0]);
        exit(1);
    }

    /* lock the 'mutex' semaphore, to get exclusive access to */
    /* the 'common' spool directory.                          */
    sem_wait_mutex(sem_set_id);

    /* move copied file into 'common' spool directory. */
    sprintf(tmp_file_path, "%s/%s", SPOOL_DIR_IN, tmp_file);
    sprintf(tmp_file_out, "%s/%s", SPOOL_DIR_COMMON, tmp_file);
    rc = rename(tmp_file_path, tmp_file_out);
    if (rc == -1) {
        fprintf(stderr, "%s: failed moving file '%s' to directory '%s': ",
                argv[0], tmp_file, SPOOL_DIR_COMMON);
        perror("");
        /* remove the problematic file, to avoid cluttering the disk. */
        unlink(tmp_file);
    }

    /* unlock the 'mutex' semaphore. */
    sem_signal_mutex(sem_set_id);

    /* increment the 'counter' semaphore, to signal tiny-lpd that */
    /* there is a new file to handle.                             */
    sem_signal_counter(sem_set_id);
}
```

```
/*
 * tiny-lpd.c  - a tiny printer daemon. Runs in the background, and
 *               sends any files queued in its spool directory to the printer.
 */

#include <stdio.h>           /* standard I/O routines.                     */
#include <sys/types.h>       /* various type definitions.                  */
#include <sys/ipc.h>         /* general SysV IPC structures                */
#include <sys/sem.h>         /* semaphore functions and structs.           */
#include <unistd.h>          /* access(), etc.                             */
#include <malloc.h>          /* malloc(), etc.                             */
#include <string.h>          /* strlen(), etc.                             */
#include <dirent.h>          /* opendir(), DIR, struct dirent, etc.        */

#include "tiny-lp-common.h"  /* common tiny spooler definitions/functions. */

/*
 * function: print_file. send a given file to the printer for printing.
 * intput:   file path.
 * output:   none.
 */
void
print_file(char* file_path)
{
    /* in a real printer daemon, here would come all the nasty    */
    /* tasks of communicating with the printer using a postscript */
    /* driver or some other interface.....                        */

    /* perhaps you'd like to try implementing this yourself?      */
    printf("tiny-lpd: 'printed' file '%s'\n", file_path);
}

/*
 * function: main. perform the whole operation.
 * intput:   none.
 * output:   messages regarding success or failure of operations.
 */
void
main(int argc, char* argv[])
{
    int sem_set_id;             /* ID of the semaphore set.             */
    DIR* spool_dir;             /* spool directory contents struct.     */
    struct dirent* dir_entry;   /* one entry of spool directory.        */
    char file_source[MAX_PATH]; /* path to file we operate on.          */
    char file_target[MAX_PATH]; /* path to file after copying to 'out'. */
    int found;                  /* was a file found in the dir. scan?   */

    /* create the semaphore set for the tiny spooling system.       */
    sem_set_id = sem_set_get("tiny-lpd", 1);
    if (sem_set_id == -1) {
        fprintf(stderr, "tiny-lpd: Error initializing. aborting.\n");
        exit(1);
    }

    /* perform a loop of waiting on the 'counter' semaphore,        */
    /* and handling one file at a time.                             */
    while (1) {
        /* wait until at least one file is found in the spool area. */
        sem_wait_counter(sem_set_id);
```

```
        /* lock the mutex semaphore, to assure exclusive access to  */
        /* the common spool directory.                              */
        sem_wait_mutex(sem_set_id);

        spool_dir = opendir(SPOOL_DIR_COMMON);
        if (!spool_dir) {
            fprintf(stderr, "%s: Failed opening directory '%s': ",
                    argv[0], SPOOL_DIR_COMMON);
            perror("");
            fprintf(stderr, "%s: aborting.\n", argv[0]);
            exit(1);
        }
        /* start reading the directory's contents. */
        found = 0;
        while ( (dir_entry = readdir(spool_dir)) != NULL) {
            /* skip the '.' and '..' entries. */
            if (strcmp(dir_entry->d_name, ".") == 0)
                continue;
            if (strcmp(dir_entry->d_name, "..") == 0)
                continue;
            /* we found a file. */
            found = 1;
            /* move file to 'out' spool sub-directory. */
            sprintf(file_source, "%s/%s", SPOOL_DIR_COMMON, dir_entry->d_name);
            sprintf(file_target, "%s/%s", SPOOL_DIR_OUT, dir_entry->d_name);
            if (rename(file_source, file_target) == -1) {
                fprintf(stderr, "%s: cannot move file '%s/%s' to '%s':\n",
                        argv[0], SPOOL_DIR_COMMON, dir_entry->d_name,
                        SPOOL_DIR_OUT);
                perror("");
                fprintf(stderr, "%s: aborting.\n", argv[0]);
                exit(1);
            }
            /* free the mutex semaphore. */
            sem_signal_mutex(sem_set_id);

            /* finally, print the file, and remove it. */
            print_file(file_target);
            unlink(file_target);
        }
        if (!found) { /* no file. maybe another tiny-lpd process grabed it? */
            /* free the mutex semaphore. */
            sem_signal_mutex(sem_set_id);
        }
    }
    /* NOT REACHED */
}
```

Index of /~choo/lupg/tutorials/multi-process/tiny-spool

- Parent Directory
- Makefile
- tiny-lp-common.c
- tiny-lp-common.h
- tiny-lpd.c
- tiny-lpr.c
- tiny-spooldir

```
printer client/server (lpr/lpd) - a producer-consumer system example
---------------------------------------------------------------------

This directory contains the source code for two programs - tiny-lpr,
that spools files for printing, and tiny-lpd, a printing daemon that takes
files from the spool directory and sends tem to the printer, one file
at a time.

To compile the programs, type 'make' (or 'gmake', to use gnu make).

Experiment with running these programs. Check what happens if you run multiple
'tiny-lpr' (producer) processes at the same time. Check how the system
works if more then one printer daemon (tiny-lpd) is being run at the same time.

Try modifying the system so that only a single printer daemon may be running
at the same time.

Note: to ease running the programs, the spool directoy path is given as
a relative path (this of-course must be changed to a full path in a real
system). Thus, you should run both programs from the source directory in order
for them to work, or modify the file path macro SPOOL_DIR in file
'tiny-lp-common.h' to a full path, and copy the contents of the 'tiny-spooldir'
directory to the new location.

You may also check the semaphore set created by the tiny-lpd program,
by issuing the following command:

ipcs -s

If done after running tiny-lpd, you will see something similar to this:

------ Semaphore Arrays --------
semid     owner     perms     nsems     status
128       choo      666       2

note that the 'msgid' field contains a value that is different from what
was supplied to the 'semget()' system call by the tiny-lpd program. This is
the value that you should use when trying to remove the semaphore set,
using the 'ipcrm' command.
```

```
/*
 * shared-mem.c  - demonstrates basic usage of shared memory.
 */

#include <stdio.h>       /* standard I/O routines.               */
#include <sys/types.h>   /* various type definitions.            */
#include <sys/ipc.h>     /* general SysV IPC structures          */
#include <sys/shm.h>     /* shared memory functions and structs. */

/* define a structure to be used in the given shared memory segment. */
struct country {
    char name[30];
    char capital_city[30];
    char currency[30];
    int population;
};

int main(int argc, char* argv[])
{
    int shm_id;               /* ID of the shared memory segment.   */
    char* shm_addr;           /* address of shared memory segment.  */
    int* countries_num;       /* number of countries in shared mem. */
    struct country* countries; /* countries array in shared mem.    */
    struct shmid_ds shm_desc;
    int i;                    /* counter for loop operation.       */

    /* allocate a shared memory segment with size of 2048 bytes. */
    shm_id = shmget(100, 2048, IPC_CREAT | IPC_EXCL | 0600);
    if (shm_id == -1) {
        perror("main: shmget: ");
        exit(1);
    }

    /* attach the shared memory segment to our process's address space. */
    shm_addr = shmat(shm_id, NULL, 0);
    if (!shm_addr) { /* operation failed. */
        perror("main: shmat: ");
        exit(1);
    }

    /* create a countries index on the shared memory segment. */
    countries_num = (int*) shm_addr;
    *countries_num = 0;
    countries = (struct country*) ((void*)shm_addr+sizeof(int));

    strcpy(countries[0].capital_city, "U.S.A");
    strcpy(countries[0].capital_city, "Washington");
    strcpy(countries[0].currency, "U.S. Dollar");
    countries[0].population = 250000000;
    (*countries_num)++;

    strcpy(countries[1].capital_city, "Israel");
    strcpy(countries[1].capital_city, "Jerusalem");
    strcpy(countries[1].currency, "New Israeli Shekel");
    countries[1].population = 6000000;
    (*countries_num)++;

    strcpy(countries[1].capital_city, "France");
    strcpy(countries[1].capital_city, "Paris");
    strcpy(countries[1].currency, "Frank");
```

```
    countries[1].population = 60000000;
    (*countries_num)++;

    /* now, print out the countries data. */
    for (i=0; i < (*countries_num); i++) {
        printf("Countery %d:\n", i+1);
        printf("  name: %s:\n", countries[i].name);
        printf("  capital city: %s:\n", countries[i].capital_city);
        printf("  currency: %s:\n", countries[i].currency);
        printf("  population: %d:\n", countries[i].population);
    }

    /* detach the shared memory segment from our process's address space. */
    if (shmdt(shm_addr) == -1) {
        perror("main: shmdt: ");
    }

    /* de-allocate the shared memory segment. */
    if (shmctl(shm_id, IPC_RMID, &shm_desc) == -1) {
        perror("main: shmctl: ");
    }

    return 0;
}
```

```
/*
 * shared-mem-with-semaphore.c  - using a semaphore to synchronize access
 *                                to a shared memory segment.
 */

#include <stdio.h>       /* standard I/O routines.                */
#include <sys/types.h>   /* various type definitions.             */
#include <sys/ipc.h>     /* general SysV IPC structures           */
#include <sys/shm.h>     /* semaphore functions and structs.      */
#include <sys/sem.h>     /* shared memory functions and structs. */
#include <unistd.h>      /* fork(), etc.                          */
#include <wait.h>        /* wait(), etc.                          */
#include <time.h>        /* nanosleep(), etc.                     */
#include <stdlib.h>      /* rand(), etc.                          */

#define SEM_ID    250    /* ID for the semaphore.                */

/* define a structure to be used in the given shared memory segment. */
struct country {
    char name[30];
    char capital_city[30];
    char currency[30];
    int population;
};

/*
 * function: random_delay. delay the executing process for a random number
 *           of nano-seconds.
 * input:    none.
 * output:   none.
 */
void
random_delay()
{
    static int initialized = 0;
    int random_num;
    struct timespec delay;            /* used for wasting time. */

    if (!initialized) {
        srand(time(NULL));
        initialized = 1;
    }

    random_num = rand() % 10;
    delay.tv_sec = 0;
    delay.tv_nsec = 10*random_num;
    nanosleep(&delay, NULL);
}

/*
 * function: sem_lock. locks the semaphore, for exclusive access to a resource.
 * input:    semaphore set ID.
 * output:   none.
 */
void
sem_lock(int sem_set_id)
{
    /* structure for semaphore operations.   */
    struct sembuf sem_op;
```

```
    /* wait on the semaphore, unless it's value is non-negative. */
    sem_op.sem_num = 0;
    sem_op.sem_op = -1;
    sem_op.sem_flg = 0;
    semop(sem_set_id, &sem_op, 1);
}

/*
 * function: sem_unlock. un-locks the semaphore.
 * input:    semaphore set ID.
 * output:   none.
 */
void
sem_unlock(int sem_set_id)
{
    /* structure for semaphore operations.   */
    struct sembuf sem_op;

    /* signal the semaphore - increase its value by one. */
    sem_op.sem_num = 0;
    sem_op.sem_op = 1;   /* <-- Comment 3 */
    sem_op.sem_flg = 0;
    semop(sem_set_id, &sem_op, 1);
}

/*
 * function: add_country. adds a new country to the counties array in the
 *           shard memory segment. Handles locking using a semaphore.
 * input:    semaphore id, pointer to countries counter, pointer to
 *           counties array, data to fill into country.
 * output:   none.
 */
void
add_country(int sem_set_id, int* countries_num, struct country* countries,
            char* country_name, char* capital_city, char* currency,
            int population)
{
    sem_lock(sem_set_id);
    strcpy(countries[*countries_num].name, country_name);
    strcpy(countries[*countries_num].capital_city, capital_city);
    strcpy(countries[*countries_num].currency, currency);
    countries[*countries_num].population = population;
    (*countries_num)++;
    sem_unlock(sem_set_id);
}

/*
 * function: do_child. runs the child process's code, for populating
 *           the shared memory segment with data.
 * input:    semaphore id, pointer to countries counter, pointer to
 *           counties array.
 * output:   none.
 */
void
do_child(int sem_set_id, int* countries_num, struct country* counties)
{
    add_country(sem_set_id, countries_num, counties,
                "U.S.A", "Washington", "U.S. Dollar", 250000000);
    random_delay();
    add_country(sem_set_id, countries_num, counties,
```

```
                "Israel", "Jerusalem", "New Israeli Shekel", 6000000);
    random_delay();
    add_country(sem_set_id, countries_num, counties,
                "France", "Paris", "Frank", 60000000);
    random_delay();
    add_country(sem_set_id, countries_num, counties,
                "Great Britain", "London", "Pound", 55000000);
}

/*
 * function: do_parent. runs the parent process's code, for reading and
 *           printing the contents of the 'countries' array in the shared
 *           memory segment.
 * input:    semaphore id, pointer to countries counter, pointer to
 *           counties array.
 * output:   printout of countries array contents.
 */
void
do_parent(int sem_set_id, int* countries_num, struct country* countries)
{
    int i, num_loops;

    for (num_loops=0; num_loops < 5; num_loops++) {
        /* now, print out the countries data. */
        sem_lock(sem_set_id);
        printf("-------------------------------------------------\n");
        printf("Number Of Countries: %d\n", *countries_num);
        for (i=0; i < (*countries_num); i++) {
            printf("Country %d:\n", i+1);
            printf("  name: %s:\n", countries[i].name);
            printf("  capital city: %s:\n", countries[i].capital_city);
            printf("  currency: %s:\n", countries[i].currency);
            printf("  population: %d:\n", countries[i].population);
        }
        sem_unlock(sem_set_id);
        random_delay();
    }
}

int main(int argc, char* argv[])
{
    int sem_set_id;            /* ID of the semaphore set.           */
    union semun sem_val;       /* semaphore value, for semctl().     */
    int shm_id;                /* ID of the shared memory segment.   */
    char* shm_addr;            /* address of shared memory segment.  */
    int* countries_num;        /* number of countries in shared mem. */
    struct country* countries; /* countries array in shared mem.     */
    struct shmid_ds shm_desc;
    int rc;                    /* return value of system calls.      */
    pid_t pid;                 /* PID of child process.              */

    /* create a semaphore set with ID 250, with one semaphore   */
    /* in it, with access only to the owner.                    */
    sem_set_id = semget(SEM_ID, 1, IPC_CREAT | 0600);
    if (sem_set_id == -1) {
        perror("main: semget");
        exit(1);
    }

    /* intialize the first (and single) semaphore in our set to '1'. */
```

```
sem_val.val = 1;
rc = semctl(sem_set_id, 0, SETVAL, sem_val);
if (rc == -1) {
    perror("main: semctl");
    exit(1);
}

/* allocate a shared memory segment with size of 2048 bytes. */
shm_id = shmget(100, 2048, IPC_CREAT | IPC_EXCL | 0600);
if (shm_id == -1) {
    perror("main: shmget: ");
    exit(1);
}

/* attach the shared memory segment to our process's address space. */
shm_addr = shmat(shm_id, NULL, 0);
if (!shm_addr) { /* operation failed. */
    perror("main: shmat: ");
    exit(1);
}

/* create a countries index on the shared memory segment. */
countries_num = (int*) shm_addr;
*countries_num = 0;
countries = (struct country*) ((void*)shm_addr+sizeof(int));

/* fork-off a child process that'll populate the memory segment. */
pid = fork();
switch (pid) {
    case -1:
        perror("fork: ");
        exit(1);
        break;
    case 0:
        do_child(sem_set_id, countries_num, countries);
        exit(0);
        break;
    default:
        do_parent(sem_set_id, countries_num, countries);
        break;
}

/* wait for child process's terination. */
{
    int child_status;

    wait(&child_status);
}

/* detach the shared memory segment from our process's address space. */
if (shmdt(shm_addr) == -1) {
    perror("main: shmdt: ");
}

/* de-allocate the shared memory segment. */
if (shmctl(shm_id, IPC_RMID, &shm_desc) == -1) {
    perror("main: shmctl: ");
}

return 0;
```

```
}
```

From: Bram Moolenaar <Bram@moolenaar.net>
To: vim-announce@vim.org
Subject: Vim version 5.8 has been released
Date: Thu, 31 May 2001 17:40:39 +0200
Message-Id: <200105311540.f4VFedV38514@moolenaar.net>

Announcing: Vim (Vi IMproved) version 5.8
    Author: Bram Moolenaar et al.

Announcement

This is a minor release of Vim. Since version 5.7 has been released (almost a year ago!) bugs have been fixed and many syntax files have been added.

Vim 5.8 has been tested by quite a few people over the past weeks, and there are no big changes since version 5.7. This should be the most stable Vim version ever. But it's free software, the usual disclaimers apply.

Development of Vim 6.0, the next major release, continues. That version contains many new features and is an unstable version.

What is Vim?

Vim is an almost 100% compatible version of the UNIX editor Vi. Many new features have been added: Multi level undo, syntax highlighting, command line history, filename completion, block operations, etc. Those who don't know Vi can probably skip this message, unless you are prepared to learn something new and useful. Vim is especially recommended for editing programs.

Vim runs on almost any Unix flavor, MS-DOS, MS-Windows 3.1, MS-Windows 95/98/ME/NT/2000/XP, OS/2, Atari MiNT, BeOS, VMS, RISC OS, Macintosh and Amiga.

For more information, see "http://www.vim.org" and "http://vim.sf.net".

New since version 5.7

74 new syntax files.

Quite a few bug fixes. This version is aimed at stability.
A few security problems were fixed.

Ctags is no longer included, it has grown into a project of its own. You can find it at http://ctags.sf.net.

See ":help version-5.8" in Vim for the details.

Where to get it

You can find a list of distribution sites at

        ftp://ftp.vim.org/pub/vim/MIRRORS

Suggested locations:

```
        ftp://ftp.<country>.vim.org/pub/vim/
        ftp://ftp.vim.org/pub/vim/
```

Replace <country> with a country code, e.g.:

```
        ftp://ftp.us.vim.org/pub/vim/
```

What is available

Note that for all systems the distribution is split into an archive with runtime files (documentation, syntax files, etc.) and a binary and/or source archive. You should download at least two archives!

FILE	SYSTEM	COMMENTS
unix/vim-5.8-src.tar.gz	Unix(*)	Sources.
unix/vim-5.8-rt.tar.gz	Unix(*)	Runtime files.
extra/vim-5.8-extra.tar.gz	Unix(*)	Extra sources and docs (Farsi, OLE, VisVim).
unix/vim-5.7-5.8-rt.diff.gz		Runtime diff with version 5.7.
unix/vim-5.7-5.8-src.diff.gz		Sources diff with version 5.7.
extra/vim-5.7-5.8-extra.diff.gz		Extra diff with version 5.7.
pc/vim58rt.zip	Win32	MS-DOS and MS-Windows runtime files. This doesn't fit on a floppy, alternatively get the next two files.
pc/vim58rt1.zip	Win32	Same, except the syntax files.
pc/vim58rt2.zip	Win32	Same, only the syntax files.
pc/gvim58.zip	Win32	32 bit MS-Windows 95/98/NT/2000 GUI binaries. Recommended for MS-Windows 95/98/NT/2000. The best choice for syntax highlighting and speed.
pc/gvim58ole.zip	Win32	32 bit MS-Windows 95/98/NT/2000 GUI binaries, with OLE support and VisVim.
pc/gvim58_s.zip	Win32s	32 bit MS-Windows 3.1/3.11 GUI binaries. Requires Win32s.
pc/vim58w32.zip	Win32	32 bit MS-Windows 95/98/NT/2000 console binaries. Recommended for Windows NT/2000, NOT for 95/98. Supports long file names.
pc/vim58d32.zip	MS-DOS	32 bit protected mode binaries. Recommended for MS-DOS, MS-Windows 3.1 and MS-Windows 95/98 console. It is compiled with DJGPP, it may need a DPMI driver (CWSDPMI is included). Supports long file names on MS-Windows 95/98 (NOT on NT/2000).
pc/vim58d16.zip	MS-DOS	16 bit real mode binaries. Runs on most MS-DOS systems, but is restricted to using 640K memory. Small version, without e.g., syntax highlighting and autocommands.
pc/vim58src.zip	PC	Sources for PC versions (with CR-LF).
os2/vim58rt.zip	OS/2	Runtime files (same as pc/vim58rt.zip).
os2/vim58os2.zip	OS/2	Binaries. (use the Unix archives for sources)

```
amiga/vim58rt.tgz          Amiga   Runtime files.
amiga/vim58bin.tgz         Amiga   Binaries.
amiga/vim58big.tgz         Amiga   Binaries with more features.
amiga/vim58src.tgz         Amiga   Sources for Amiga.

doc/vim58html.zip          all     Documentation converted to HTML.

patches/*                  all     Recent patches.

(*) Also for BeOS, OS/2, VMS, Macintosh and Atari MiNT

The contents of the source archives depends on the specified system.  To
obtain the full sources and docs, get the three Unix archives.


system    file type      Unpack with

Unix      file.tar.gz    gunzip file.tar.gz; tar xf file.tar
Amiga     file.tgz       gzip -d file.tgz; tar xf file.tar
PC        file.zip           pkunzip -d file.zip
                         or: unzip file.zip


Mailing lists
-------------
For user questions you can turn to the Vim mailing list.  There are a lot of
tips, scripts and solutions.  You can ask your Vim questions, but only if you
subscribe.  See http://www.vim.org/mail.html.  An archive is kept at
http://www.egroups.com/group/vim.

If you want to help developing Vim or get the latest patches, subscribe to the
vim-dev mailing list.  An archive is kept at
http://www.egroups.com/group/vimdev.

Subject specific lists:
Multi-byte issues: vim-multibyte  http://www.egroups.com/group/vim-multibyte
Macintosh issues:  vim-mac        http://www.egroups.com/group/vim-mac


Reporting bugs
--------------
Send them to <Bram@vim.org>.  Please be brief, all the time spent on answering
mail is subtracted from the time that is spent on improving Vim!  Always give
a reproducable example and try to find out which settings or other things
influence the appearance of the bug.  Try different machines if possible.  See
":help bugs" in Vim.  Send me patches if you can!

If something needs discussing with other developers, send a message to the
vim-dev mailing list.  You need to subscribe first.


Happy Vimming!

--
"Oh, no!  NOT the Spanish Inquisition!"
"NOBODY expects the Spanish Inquisition!!!"
                                -- Monty Python sketch --
"Oh, no!  NOT another option!"
"EVERYBODY expects another option!!!"
```

-- Discussion in vim-dev mailing list --

/// Bram Moolenaar -- Bram@moolenaar.net -- http://www.moolenaar.net \\\
(((Creator of Vim -- http://vim.sf.net -- ftp://ftp.vim.org/pub/vim   )))
 \\\ Help me helping AIDS orphans in Uganda - http://iccf-holland.org ///

From: Bram Moolenaar <Bram@moolenaar.net>
To: vim-dev@vim.org
Subject: Vim version 6.0am ALPHA available
Date: Sun, 01 Jul 2001 21:32:38 +0200
Message-Id: <200107011932.f61JWcG85411@moolenaar.net>

The user manual is complete now. You are invited to start proof reading.
Keep in mind that it only contains the most useful features, not every detail.

We are getting much closer to a beta test now. If you are
working on something, please send me a patch the coming week.

Major changes

The user manual is complete. Still requires proof reading.

":menu <silent>" adds a menu that won't echo Ex commands.
":map <silent>" adds a mapping that won't echo Ex commands.

The distribution files for Unix have been reorganised. The ".gz" files are
now small enough to fit on a floppy. These can be used by (old) standalone
systems. There is one big ".bz2" file that contains everything.

Other changes

"copen" opens the quickfix window, ":cclose" closes it.
":cwindow" takes care of having a quickfix window only when there are
recognized errors. (Dan Sharp)

Added the "-M" Vim argument: reset 'modifiable' and 'write', thus disallow
making changes and writing files.

GTK: Allow dropping a http:// and ftp:// URL on Vim. The netrw plugin takes
care of downloading the file. (MiKael Berthe)

Added the 'imcmdline' option: When set the input method is always enabled when
starting to edit a command line. Useful for a XIM that uses dead keys to type
accented characters.
Made CTRL-^ in Insert mode and when editing the command line toggle the use of
an input method when no language mappings are present. Allows switching the
IM back on when halfway typing a line.

Collection of changes for the Macintosh: (Dany St-Amant)
- Reorganized the gui_mac.c (it was a mess)
- Carbonized (while keeping non Carbon code)
  (Some work "stolen" from Ammon Skidmore)
- Improved the menu item index handling (should be faster)
- Runtime command now handle / in file name (MacOS 9 version)
- Added winpos support
- Fixed a bug in the handling of Activate/Deactivate Event
- Fixed a bug in gui_mch_dialog (using wrong pointer)

Added tidy syntax file. (Doug Kearns)

Improved MS-Windows install program a bit more. (Jon Merz)

Made the info for an existing swap file a bit shorter, so that it still fits on a 24 line screen.

":silent !cmd" no longer redraws the screen. The user must use CTRL-L if he wants a redraw.

Fixes

Multi-byte: When formatting lines Vim could crash. (Alexey Marinichev)

Perl: Dynamic loading on Win32 didn't work. (Paul Moore)
Removed some debugging code. (Donnie Smith)

Python: removed Py_GetProgramName(), it's not used.

Motif: Didn't compile when XPM stuff is not available.

VMS does not allow function names longer than 31 characters. Renamed gui_motif_create_fontlist_from_fontset() to gui_motif_fontset2fontlist().

menu.c didn't compile without the multi-language feature when the toolbar is used. (Maurice Barnum)

There was no declaration for strdup(), except in integration.c, where it causes problems when it's a macro. Added it to osdef1.h.in.

When configure was told to use the workshop feature, it would still select the GTK GUI by default. Give an error message when trying to use workshop without Motif.

It was possible to make a symlink with the name of a swap file, linking to a file that doesn't exist. Vim would then silently use another file (if open with O_EXCL refuses a symlink). Now check for a symlink to exist. Also do another check for an existing swap file just before creating it to catch a symlink attack.

When using "vd" to delete an empty line, the first multibyte character in the next line could be damaged. (Muraoka Taro)

The output of "g CTRL-G" mentioned "Char" but it's actually bytes.

Searching for the end of a oneline region didn't work correctly when there is an offset for the highlighting.

Athena: The triangle for a submenu could be allocated at the wrong moment. (David Harrison)

"vimtutor" used "test -e". Apparently not all systems support that. Use "test -f" instead.

Syntax highlighting: When synchronizing on C-comments, //*/ was seen as the start of a comment.

Mac: Vim was eating 80% of the CPU time. (Dany St-Amant)

GTK: When leaving Insert mode the XIM status wasn't obtained correctly. When re-entering Insert mode it would be disabled.

Folding: When 'foldmethod' is "marker", inserting a line break before a marker opened folds below it. Was updating the folds twice, once for inserting the line, which caused all following folds to become nested.

":silent":
- Various commands still produced a message with ":silent": ":write", ":substitute", ":put, ":delete".
- Redirecting messages and using ":silent highlight" doesn't produce the same output as without the ":silent". Now at least insert one space.

Thesaurus completion got stuck at end of line. It could not be interrupted with CTRL-C either.

Epilogue

WARNING: This is really an unstable version. Many things have been added without proper testing. It does crash. It may destroy your work. New features may change incompatibly in the next version.

This version is for developers, thus it comes as source code only.
If you run into something that doesn't work, please try to figure out why, try to solve it and send me a patch. If you can't do that, at least send me precise information to save me time.

More info about the new 6.0 features with ":help version6".

If you don't like the syntax of a command, the name of an option or how the new features work, let's discuss this in the vim-dev maillist.

Lots of things are not working yet. Check ":help todo" for known items.

I NEED YOUR HELP: There is still a lot of work to be done. If I have to do it all by myself it will take a very long time until Vim.6.0 is ready. Please give a hand by implementing one of the items in the todo list.

You can find Vim 6.0 here: ftp://ftp.vim.org/pub/vim/unreleased/

unix/vim-6.0am.tar.bz2	sources + runtime files, bzip2 compressed
unix/vim-6.0am-rt1.tar.gz	runtime files part 1
unix/vim-6.0am-rt2.tar.gz	runtime files part 2
unix/vim-6.0am-src1.tar.gz	sources part 1
unix/vim-6.0am-src2.tar.gz	sources part 2
extra/vim-6.0am-extra.tar.gz	extra files
extra/vim-6.0am-lang.tar.gz	multi-language files

Happy Vimming!

Sven Guckes guckes@vim.org
 Last update: Sat Jun 09 14:00:00 MEST 2001

VIM - Distribution and Download

This page holds the **long** list of all (ftp,www) mirror sites of Vim.

You probably accessed this page to download the Vim source code or a binary file for your operating system. In that case you need to (1) chose an **ftp mirror** from this page, (2) connect to the mirror with you web browser (which needs to understands the FTP protocol), (3) change into a subdirectory, (4) and start downloading.

If you did not understand this then please read the Download FAQ first! It gives you directions and help on downloading. After that, you should be able to select a mirror and download.

If you need a *short* list of all mirrors then take a look at the Vim Mirrors List.

If you need assistance then please hesitate to write until after you've read the following: ;-)

Vim comes in **two** parts: One archive holds the *program* (either as source or binary/executable), and another archive holds the *runtime files* with the documentation and syntax files. This allows you to update Vim as a program without having to download the latest documentation. But usually you should update the matching documentation, too. ;-)

The ftp mirrors hold binaries for only some operating systems. There are more maintained binaries on other sites which are listed on the Binaries Page.

OK, now you've read the Download FAQ and you still have questions? OK - send me an email then. Be sure to include some info on what you want, what you did, and what went wrong. I'll ask you, anyway. You have been warned!

Vim Distribution - Locating Mirrors for Countries

Locating a mirror or its info on this page is pretty easy:

```
By country code (ISO3166):
http://www.vim.org/dist.html#XY
http://www.vim.org/dist.html#DE (Germany)

By country name:
http://www.vim.org/dist.html#Name
http://www.vim.org/dist.html#Germany

By city name:
http://www.vim.org/dist.html#City
http://www.vim.org/dist.html#Berlin
```

If you want to create an ftp mirror or www mirror of Vim then please read the HowTo Mirror - thanks!

Vim Distribution - Overview

FTP Mirrors Overview and WWW Mirrors Overview

Vim - FTP Mirrors Overview

- AT - Austria: Vienna | Vienna HTTP download
- AU - Australia: Sydney | Sydney HTTP download
- BE - Belgium: Leuven
- CA - Canada: Cambridge [temporarily offline! 001018]
- CU - Cuba: [001013: coming up!]
- CZ - Czech Republic: [todo]
- DE - Germany:
  - Berlin
  - Frankfurt
  - Oldenburg
- ES - Spain: Oviedo
- FR - France:
  - Gif-Sur-Yvette
  - Paris
- GR - Greece: Heraklion
- HU - Hungary:
  - Miskolc
  - Budapest [001001]
- IT - Italy: [two entries which need to be checked]
- JP - Japan: Tokio
- KR - Korea:
- MX - Mexico:
- NL - Netherlands:
- PL - Poland:
- PT - Portugal:
- RU - Russia:
- SE - Sweden:
- SK - Slovakia:
- TR - Turkey:
- UK - United Kingdom:
- ZA - South Africa:

  USA (com,gov,net):
- California (2):
- Maryland:
- Michigan:
- New York:
- Illinois:
- Washington:

- Michigan:
- Florida:

VIM - FTP Sites - LONG List

Save bandwidth - download from a site near you! Thank you!

The VIM source and documentation can be obtained from the following sites:

```
AT - Austria
AT AUSTRIA [990310]              Europe, Austria, Vienna: Univ. of Technology
AT AUSTRIA Mirror FTP info:      150 max connections; (MET)
AT AUSTRIA Mirror FTP source:    ftp://ftp.home.vim.org/pub/vim/
AT AUSTRIA Mirror FTP target:    ftp://ftp.at.vim.org/pub/vim/
AT AUSTRIA Mirror FTP target:    http://ftp.at.vim.org/pub/vim/
AT AUSTRIA Mirror FTP contact:   Antonin Sprinzl Antonin.Sprinzl(at)tuwien.ac.at
AT AUSTRIA Mirror FTP updates:   updated daily
AT AUSTRIA Mirror FTP comment:   Access for both ftp and http!
```

AU - Australia

```
AUSTRALIA [961217,000303]              Australia, Country, Sydney;
AUSTRALIA                        University of Technology, Programmers' Society
AUSTRALIA Mirror FTP planned:    ???
AUSTRALIA Mirror FTP info:       25 max connections; (TIMEZONE???)
AUSTRALIA Mirror FTP source:     -> NICKNAME
AUSTRALIA Mirror FTP target:     ftp://ftp.progsoc.uts.edu.au/pub/vim/
AUSTRALIA vim.org name:          ftp://ftp.au.vim.org/pub/vim/
AUSTRALIA vim.org name:          http://ftp.au.vim.org/pub/vim/
AUSTRALIA Mirror FTP contact:    FTP Admin ftp(at)progsoc.uts.edu.au
```

BE - Belgium

```
BE Belgium [990125,000203]       Europe, Belgium, Leuven; grmbl
BE Belgium vim.org name:         ftp://ftp.be.vim.org/pub/vim/
BE Belgium Mirror FTP source:    ftp://ftp.fr.vim.org/pub/vim/
BE Belgium Mirror FTP target:    ftp://ftp.grmbl.com/pub/vim/
BE Belgium Mirror FTP info:      50 max connections; weekly mirror
BE Belgium Mirror FTP contact:   Bram Dumolin ftp(at)grmbl.com
```

CA - Canada

```
FTP.CA CANADA [971017,990111]    America, Canada, Ontario, Cambridge; RingZero
FTP.CA CANADA vim.org name               temporarily offline [001018]
FTP.CA CANADA Mirror FTP info:   75 max connections; (EDT +5); nightly
FTP.CA CANADA Mirror FTP source:         UCDavis
FTP.CA CANADA Mirror FTP target:
FTP.CA CANADA Mirror FTP contact:        Mark Mayo mark(at)vmunix.com
```

CU - Cuba

Coming up!
001012 Yurais Fernandez Leal

DE - Deutschland (Germany)

DE1 Berlin [960101,000126] Europe, Germany, Berlin; Freie Universitaet
DE1 Berlin Mirror FTP aliases: ftp://ftp.de.vim.org/pub/vim/
ftp://ftp1.de.vim.org/pub/vim/
ftp://ftp.berlin.de.vim.org/
ftp://vim.ftp.fu-berlin.de :-)
DE1 Berlin target ftp://ftp.fu-berlin.de/misc/editors/vim/
DE1 Berlin contact: ftp-adm(at)fu-berlin.de

DE2 Frankfurt [000121,001219] Europe, Germany, Frankfurt; Gigabell AG
DE2 Frankfurt Mirror FTP aliases: ftp://ftp.frankfurt.de.vim.org/pub/vim/
DE2 Frankfurt Mirror FTP target: ftp://ftp.gigabell.de/pub/vim/
DE2 Frankfurt Mirror FTP source: ftp.vim.org
DE2 Frankfurt Mirror FTP info: 500 max connections; (MET)
DE2 Frankfurt Mirror FTP contact: ftp(at)gigabell.net FTP-Admin

DE3 Oldenburg [990115,000126] Europe, Germany, Oldenburg; University
DE3 Oldenburg Mirror FTP aliases: ftp://ftp3.de.vim.org/pub/vim/
ftp://ftp.oldenburg.vim.org/pub/vim/
DE3 Oldenburg Mirror FTP source: OCE; daily
DE3 Oldenburg Mirror FTP target:
ftp://ftp.informatik.uni-oldenburg.de/pub/vim/
DE3 Oldenburg Mirror FTP info: ??? max connections; (CET); Email access!
DE3 Oldenburg Mirror FTP<->Email: ftpmail(at)informatik.uni-oldenburg.de
"Subject: help" for instructions
DE3 Oldenburg Mirror FTP contact: Ingo.Wilken(at)Informatik.Uni-Oldenburg.DE

ES - España (Spain)

ES SPAIN [980624] Europe, Spain, Oviedo; University of Oviedo
ES SPAIN vim.org name ftp://ftp.es.vim.org/pub/vim/
ES SPAIN Mirror FTP info: 150 max connections; (MET-1METDST)
ES SPAIN Mirror FTP source: -> Berlin
ES SPAIN Mirror FTP target: ftp://ftp.etsimo.uniovi.es/pub/vim/
ES SPAIN Mirror FTP target: http://www.etsimo.uniovi.es/pub/vim/
ES SPAIN Mirror FTP contact: Antonio Bello antonio(at)zeus.etsimo.uniovi.es

FR - France

FR FRANCE [991022,000203] Europe, France, Gif-Sur-Yvette; Medasys
Digital Systems
FR FRANCE vim.org name: ftp://ftp.fr.vim.org/pub/vim/
FR FRANCE Mirror FTP info: ??? max connections; (TIMEZONE???)
FR FRANCE Mirror FTP source: -> NICKNAME
FR FRANCE Mirror FTP target: URL

```
FR FRANCE Mirror FTP contact:   NAME+ADDRESS

FTP2.FR dates:  [970328,000203]
FTP2.FR place:  France, Paris (Gentilly); ILOG Company
FTP2.FR alias:  ftp://ftp.fr2.vim.org/
FTP2.FR target: ftp://ftp.ilog.fr/pub/Mirrors/vim/
FTP2.FR source: Berlin
FTP2.FR info:   10 max connections; check: twice a day (2am and 2pm)
FTP2.FR contact:        Pierre-Yves Lochou ftpmaster(at)ilog.fr

FTP3.FR dates:  [000626,000627]
FTP3.FR place:  Europe, France, Paris; Club-Internet
FTP3.FR alias:  ftp://ftp3.fr.vim.org/pub/vim/
FTP3.FR source: ftp://ftp.vim.org
FTP3.FR target: ftp://ftp.club-internet.fr/pub/vim/
FTP3.FR contact: ftpmaster(at)club-internet.fr
FTP3.FR info:   Unlimited connections; (MET);
"global bandwidth": 215 Mbps; rsync access
```

GR - Greece

```
FTP.GR dates:   [971004,000203]
FTP.GR place:   Greece, Crete, Heraklion; SunSITE, I.C.S. - FO.R.T.H.
FTP.GR                  = FOundation for Research & Technology Hellas
FTP.GR Greece                   ftp://ftp.gr.vim.org/pub/vim/
FTP.GR Greece                   ftp://sunsite.ics.forth.gr/pub/vim/
FTP.GR Greece contact:          SunSITE Team sunsite(at)ics.forth.gr
FTP.GR Greece mirrored site:    OCE - daily at 5am local time
```

HU - Hungary

```
FTP.HU dates:   [971021,000203]
FTP.HU place:   Europe, Hungary, Miskolc; Univ IIT
FTP.HU alias:   ftp://ftp.hu.vim.org/pub/vim/
FTP.HU source:  OCE (daily, 1AM CET)
FTP.HU target:  ftp://vim.iit.uni-miskolc.hu/pub/vim/
FTP.HU contact: Antal Rutz rutz(at)iit.uni-miskolc.hu
FTP.HU info:    max connections: 150 http, 50 ftp

FTP2.HU dates:  [001001]
FTP2.HU place:  Europe, Hungary, Budapest; T2.WOX.ORG
FTP2.HU alias:  ftp://ftp2.hu.vim.org
FTP2.HU source: ftp://ftp.vim.org
FTP2.HU target: ftp://vim.t2.wox.org/pub/vim/
FTP2.HU contact:        BURJAN Gabor burjang(at)t2.wox.org
FTP2.HU info:   max connections: 25; CET
```

IT - Italy

Coming up:

```
FTP.IT dates:   [000110,000203,000208]
FTP.IT place:   Europe, Italy, Milan; Politecnico di Milano
FTP.IT alias:   ftp://ftp.it.vim.org/ [todo]
```

```
FTP.IT source:  ftp://ftp.vim.org
FTP.IT target:  ftp://flander.elet.polimi.it/pub/vim [000208: no vim files yet]
FTP.IT contact: ftp-admin(at)cerbero.elet.polimi.it
                Alessandro Storti Gajani (alex(at)cerbero.elet.polimi.it)
                Andrea Novara          (andrea(at)cerbero.elet.polimi.it)
FTP.IT info:    max connections: 50
```

Coming up: [what ever happened to this mirror?]

```
IT ITALY [980731]               Europe, Italy, City; Company
IT ITALY vim.org name:          ftp://ftp.it.vim.org/
IT ITALY Mirror FTP planned:    ???
IT ITALY Mirror FTP info:       ??? max connections; (TIMEZONE???)
IT ITALY Mirror FTP source:     -> NICKNAME
IT ITALY Mirror FTP target:     URL
IT ITALY Mirror FTP contact:    Paolo M. Pumilia pumilia(at)est.it
IT ITALY Mirror WWW planned:    ???
IT ITALY Mirror WWW info:       ??? max connections; (TIMEZONE???)
IT ITALY Mirror WWW source:     URL
IT ITALY Mirror WWW target:     URL
IT ITALY Mirror WWW contact:    Paolo M. Pumilia pumilia(at)est.it
```

JP - Japan

```
FTP.JP dates:   [971021,980720]
FTP.JP place:   Asia, Japan, Tokio; TWICS
FTP.JP alias:   ftp://ftp.jp.vim.org/pub/vim/
FTP.JP alias:   ftp://ftp.tokio.jp.vim.org/pub/vim/
FTP.JP source:  ???
FTP.JP target:  ftp://ftp.twics.co.jp/unix/utils/text/vim/
FTP.JP info:    N max connections; (TIME ZONE)
FTP.JP contact: Paul Gampe paulg(at)twics.com
```

KR - Korea

```
KR Asia, Korea, Taejon; KAIST
KR vim.org name           ftp://ftp.kr.vim.org/pub/vim/
KR info:       100 max connections; (04:30/15:30 KST)
KR source:     -> UCDavis
KR target:     ftp://sparcs.kaist.ac.kr/pub/vim/
KR contact:     Nam Sedong dgtgrade(at)sparcs.kaist.ac.kr
```

MX - Mexico

ftp.mx.vim.org	[980721,980723]
contact:	Felipe Contreras fhca(at)yahoo.com
connections (max):	256
source:	FUB
target:	ftp://ftp.mx.vim.org/
info:	Mirrored at midnight Mexico time (GMT-6); directory "beta-test" not included in mirror. "There maybe some blackouts every now and then in this university on some holydays like tomorrow, but it should be ok on average."

NL - Netherlands

```
NL1 NLUUG [990126]              Netherlands, Nijmegen; NL Unix User Group
NL1 NLUUG vim.org name:         ftp://ftp.vim.org/pub/vim/
NL1 NLUUG vim.org name:         ftp://ftp.nl.vim.org/pub/vim/
NL1 NLUUG Mirror FTP info:      ??? max connections; (MET)
NL1 NLUUG Mirror FTP source:    OCE
NL1 NLUUG Mirror FTP target:    ???
NL1 NLUUG Mirror FTP contact:   Jan.Christiaan.van.Winkel(at)ATComputing.nl
NL1 NLUUG Mirror FTP contact:   ftp-admin(at)nluug.nl

NL2 OCE [971021,980105]         Europe, Holland, Venlo; OCE [HOME]
NL2 OCE vim.org name:           ftp://ftp.home.vim.org/pub/vim/
NL2 OCE vim.org name:           ftp://ftp2.nl.vim.org/pub/vim/
NL2 OCE [orig]                  ftp://ftp.oce.nl/pub/vim/
NL2 OCE contact:                Henk Boetzkes boet(at)oce.nl
NL2 OCE max connections:        200
NL2 OCE mirrored site:          None.  This is the home of Vim!  :-)
```

PL - Poland

```
PL POLAND [980213]              Europe; Poland, Warsaw; ICW, Warsaw University
PL POLAND vim.org name          ftp://ftp.pl.vim.org/
PL POLAND Mirror FTP info:      200 max connections; (MET)
PL POLAND Mirror FTP source:    OCE
PL POLAND Mirror FTP target:    ftp://sunsite.icm.edu.pl/pub/unix/vim/
PL POLAND vim.org name:         ftp://ftp.pl.vim.org/pub/vim/
PL POLAND Mirror FTP contact:   sunsite(at)icm.edu.pl
```

PT - Portugal

```
PORTUGAL [980905]               Europe, Portugal, Braga; Minho University
PORTUGAL vim.org name:          ftp://ftp.pt.vim.org/ [990126 todo!]
PORTUGAL Mirror FTP planned:    ???
PORTUGAL Mirror FTP info:       unlimited(!) max connections; (TIMEZONE???)
PORTUGAL Mirror FTP info:       P-II 300MHz; 128M RAM; 27G RAID/5 Disk-Array
PORTUGAL Mirror FTP source:     -> NICKNAME
PORTUGAL Mirror FTP target:     ftp://ftp.ci.uminho.pt/pub/editors/vim/
PORTUGAL Mirror FTP contact:    Sergio Araujo sergio(at)ci.uminho.pt
PORTUGAL Mirror WWW planned:    yes [todo - ask whether it exists yet]
```

SU - Soviet Union

```
FTP.SU SOVIETUNION [981205,000710]      Europe, Soviet Union, Moscow; Moscow State
University
FTP.SU SOVIETUNION Mirror FTP info:     100 max connections; (GMT+3)
FTP.SU SOVIETUNION Mirror FTP source:   -> RU2
FTP.SU SOVIETUNION Mirror FTP target:   ftp://chronos.cs.msu.su/pub/vim/
FTP.SU SOVIETUNION Mirror FTP contact:  FTP admin ftp-admin(at)chronos.cs.msu.su
```

SK - Slovakia

FTP.SK [990629,000530] Europe, Slovakia, Bratislava; "Kotelna"
FTP.SK vim.org name: ftp://ftp.sk.vim.org/ [todo]
FTP.SK Mirror FTP info: **unlimited** connections; (MET); 6:47am
FTP.SK Mirror FTP source: ftp://ftp.vim.org/
FTP.SK Mirror FTP target: ftp://kotol.kotelna.sk/pub/vim/
FTP.SK Mirror FTP contact: Adrien [Freddy] Farkas freddy(at)kotelna.sk

SK - Slovakia

Slovakia2 [990629] Europe, Slovakia, City; Company/Organization
Slovakia2 vim.org name: ftp://ftp.COUNTRY.vim.org/
Slovakia2 Mirror FTP planned: ???
Slovakia2 Mirror FTP info: ??? max connections; (TIMEZONE???)
Slovakia2 Mirror FTP source: -> NICKNAME
Slovakia2 Mirror FTP target: URL
Slovakia2 Mirror FTP contact: NAME+ADDRESS
Slovakia2 Mirror WWW planned: ???
Slovakia2 Mirror WWW info: ??? max connections; (TIMEZONE???)
Slovakia2 Mirror WWW source: URL
Slovakia2 Mirror WWW target: URL
Slovakia2 Mirror WWW contact: Tibor Stacho stacho(at)lemon.napri.sk

UK - United Kingdom

UK [990721,990726] Europe, Country, City; Company/Organization
UK vim.org name: http://www.uk.vim.org/ [todo]
UK Mirror FTP planned: ???
UK Mirror WWW info: ??? max connections; (TIMEZONE???)
UK Mirror WWW source: ???
UK Mirror WWW target: http://www.tqbase.demon.co.uk/mirror/vim/
UK Mirror WWW contact: David Allsopp daa(at)tqbase.demon.co.uk

ZA - South Africa

ZA SouthAfrica [980226] Africa, South Africa, CITY, COMPANY
ZA SouthAfrica vim.org name
ftp://ftp.za.vim.org/applications/editors/vim/
ZA SouthAfrica Mirror FTP info: 200 max connections; (TIME ZONE)
ZA SouthAfrica Mirror FTP source: -> NICKNAME [todo]
ZA SouthAfrica Mirror FTP target: ftp://ftp.is.co.za/applications/editors/vim/
ZA SouthAfrica Mirror FTP contact: FTP Admin ftp-admin(at)is.co.za
ZA SouthAfrica Mirror WWW info: N max connections; (TIME ZONE)
ZA SouthAfrica Mirror WWW source: ???
ZA SouthAfrica Mirror WWW target: http://www.leg.uct.ac.za/mirrors/guckes/vim/
ZA SouthAfrica Mirror WWW contact: ftp-admin(at)is.co.za

USA - com/gov/net

USA1 UCDAVIS [970210,000718] North America, USA, California, Davis; Univ

of California
USA1 UCDAVIS vim.org name: ftp://ftp.us.vim.org/pub/vim/
USA1 UCDAVIS Mirror WWW planned: running since 1998
USA1 UCDAVIS Mirror WWW info: unlimited max connections; (PST8PDT)
USA1 UCDAVIS Mirror WWW source: http://www.vim.org/
USA1 UCDAVIS Mirror WWW target: http://ftp.us.vim.org/vim/
USA1 UCDAVIS Mirror WWW contact: David O'Brien obrien(at)FreeBSD.org

USA2 CDROM.COM [971002] America, USA, California, San Francisco; CDROM.COM
USA2 CDROM.COM Maximum Simultaneous Connections: 2000 ftp, 512 http
USA2 CDROM.COM vim.org name:
ftp://ftp2.ca.us.vim.org/pub/unixfreeware/editors/vim/
USA2 CDROM.COM Mirror FTP info: 2,000 max connections; (TIMEZONE???)
USA2 CDROM.COM Mirror FTP source: ???
USA2 CDROM.COM Mirror FTP target:
ftp://ftp.cdrom.com/pub/unixfreeware/editors/vim/
USA2 CDROM.COM Mirror FTP contact: Christopher G. Mann r3cgm(at)cdrom.com
USA2 CDROM.COM Mirror FTP note: last updated on 980504 :-/
USA2 CDROM.COM Mirror WWW planned: ???

USA3 NASA [960520,000209] America, USA, Maryland, Greenbelt; NASA
USA3 NASA ftp://ftp.md.usa.vim.org/ [990125 todo!]
USA3 NASA ftp://ftp.nasa.vim.org/pub/vim/
USA3 NASA ftp://jagubox.gsfc.nasa.gov/pub/unix/vim/
USA3 NASA contact: Jim Jagielski jim(at)jagubox.gsfc.nasa.gov
USA3 NASA comment: "jagubox is a Quadra 950+ running A/UX 3.1.1."

USA4 Petrified [960529] America, USA, Michigan, Ann Arbor, Petrified
USA4 Petrified ftp://ftp.mi.us.vim.org/mirrors/vim/
USA4 Petrified ftp://petrified.cic.net/mirrors/vim/
USA4 Petrified Status: 990125 unreachable!
USA4 Petrified Contact: Alex Tang ftp(at)petrified.cic.net
USA2 Petrified Info: Probably a good mirror site for
those in the Eastern Half of the US.

USA5 Rochester [960805,990729] America, USA, New York, Rochester; Gas&Electric
USA5 Rochester ftp://ftp.ny.us.vim.org/pub/editors/vim/
USA5 Rochester [orig] ftp://ftp.rge.com/pub/editors/vim/
USA5 Rochester contact: FTP Admin ftp(at)rge.com
USA5 Rochester max FTP connections: 750 (WOW!)

USA6 UIUC [961217,010207] America, USA, Illinois, Urbana; UIUC
USA6 UIUC ftp://ftp.il.us.vim.org/packages/vim/
USA6 UIUC ftp://uiarchive.uiuc.edu/pub/packages/vim/
USA6 UIUC contact: FTP Admin ftpadmin(at)uiuc.edu
USA6 UIUC Info by: Joe Gross jgross(at)uiuc.edu
USA6 UIUC Info: "We mirror the archive nightly
and have excellent bandwidth."
USA6 UIUC max connections: 300 (wow!)
USA6 UIUC Note: No update since 2000-02-11 - vim-5.6 only.
jgross - "user unknown"

USA7 Walla [961024,990101] America, USA, WA, Walla Walla; Whitman College
USA7 Walla Mirror FTP info: 15 max connections; (Pacific Time - PDT/PST)
USA7 Walla Mirror FTP source: -> OCE

```
USA7 Walla Mirror FTP target:   ftp://wastenot.whitman.edu/pub/vim/
USA7 Walla vim.org name         ftp://ftp.wa.usa.vim.org/ [990125 todo!]
USA7 Walla Mirror FTP contact:  Albert Schueller schuelaw(at)whitman.edu

USA8 Kalamazoo [990204]         America, USA, MI, Kalamazoo
USA8 Kalamazoo Mirror WWW info: ??? max connections; (TIMEZONE)
USA8 Kalamazoo Mirror WWW source:       ???
USA8 Kalamazoo Mirror WWW target:http://rickjames.sapien.net/vim/
USA8 Kalamazoo vim.org name     http://www.mi.us.vim.org/ [toadd]
USA8 Kalamazoo Mirror WWW contact:      Brian J.S. Miller suydam(at)iserv.net
USA8 Kalamazoo Mirror notes:    no update since 981105 :-(

FLORIDA [990702,990707]         USA, Florida, Vero Beach; The TenCorp Network
FLORIDA Mirror FTP planned:     NO
FLORIDA Mirror WWW info:        ??? max connections; (EST)
FLORIDA Mirror WWW source:      http://www.vim.org
FLORIDA Mirror WWW target:      http://www.tencorp.net/www.vim.org
FLORIDA Mirror WWW contact:     Byron L. Ewers blewers(at)tencorp.net
```

Vim - WWW Mirrors

The VIM Pages can be accessed at these mirrors, too:

- AT - Austria:
- CA - Canada:
- DE - Deutschland (Germany): Munich
- ES - España (Spain):
- FR - France:
- IT - Italy:
- KR - Korea:
- LT - Lithuania:
- MX - Mexico:
- NO - Norway:
- PL - Poland:
- SE - Sweden:
- SK - Slovakia: [2 mirrors]
- SU - Soviet Union:
- TR - Turkey:
- COM - Companies

AT - Austria

```
AT AUSTRIA Mirror WWW info:     200 max connections; (MET)
AT AUSTRIA Mirror WWW source:   http://www.vim.org/
AT AUSTRIA Mirror WWW target:   http://www.at.vim.org/
AT AUSTRIA Mirror WWW contact:  Antonin Sprinzl Antonin.Sprinzl(at)tuwien.ac.at
AT AUSTRIA Mirror FTP updates:  updated twice a week
```

BE - Belgium

```
BE Belgium Mirror WWW:          http://mirror.grmbl.com/vim/
BE Belgium Mirror WWW info:     150 max connections; weekly mirror
BE Belgium Mirror WWW source:   http://www.vim.org/
BE Belgium Mirror WWW target:   http://mirror.grmbl.com/vim/
BE Belgium vim.org name:        http://www.be.vim.org/ [todo]
BE Belgium Mirror WWW contact:  Bram Dumolin bram(at)grmbl.com
```

CA - Canada

```
CA CANADA Mirror WWW info:      128 max connections; (EDT +5); nightly
CA CANADA Mirror WWW source:    http://www.vim.org/
CA CANADA Mirror WWW target:    http://www.vmunix.com/vim/
CA CANADA Mirror WWW contact:   Mark Mayo mark(at)vmunix.com
```

CZ - Czech Republic

```
CZ Prague [990603]              Europe, Czech Republic, Prague; Company?
CZ Prague Mirror FTP planned:   ???
CZ Prague Mirror WWW info:      ??? max connections; (TIMEZONE???)
CZ Prague Mirror WWW source:    URL
CZ Prague Mirror WWW target:    URL
CZ Prague Mirror WWW contact:   Peter Lowe pgl(at)instinct.org
```

DE - Deutschland (Germany)

```
DE Munich [970915]              Europe, Germany, Bavaria, Munich; University
DE Munich                       Institute of Paedagogics (Pedagogy?)
DE Munich WWW                   http://www.vim.org/
DE Munich contact:              Stefan 'Sec' Zehl sec(at)42.org
```

ES - España (Spain)

```
ES SPAIN Mirror WWW source:     http://www.vim.org/ (weekly)
ES SPAIN Mirror WWW target:     http://www.etsimo.uniovi.es/vim/
ES SPAIN WWW vim.org name       http://www.es.vim.org/ [todo!]
ES SPAIN Mirror WWW contact:    Antonio Bello antonio(at)zeus.etsimo.uniovi.es
ES SPAIN Mirror WWW info:       150 max connections; (MET-1METDST)
```

FR - France

```
WWW.FR dates:   [000203]
WWW.FR place:   Europe, France, City?; Company/Organization
WWW.FR alias:   http://www.fr.vim.org/ [todo]
WWW.FR info:    ??? max connections; (TIME ZONE)
WWW.FR source:  ???
WWW.FR target:  http://web.efrei.fr/~parmelan/vim/guckes/
WWW.FR contact: Thomas.Parmelan(at)efrei.fr
```

HU - Hungary

```
WWW2.HU dates:  [001001]
WWW2.HU place:  Europe, Hungary, Budapest; T2.WOX.ORG
WWW2.HU alias:  http://www2.hu.vim.org
WWW2.HU source: http://www.math.fu-berlin.de/~guckes/vim/
WWW2.HU target: http://vim.t2.wox.org/
WWW2.HU contact:        BURJAN Gabor burjang(at)t2.wox.org
WWW2.HU info:   max connections: 100; CET
```

IT - Italy

```
WWW.IT dates:   [000110,000203,000208]
WWW.IT place:   Europe, Italy, Milan; Politecnico di Milano
WWW.IT alias:   http://www.it.vim.org/ [todo]
WWW.IT source:  http://www.vim.org
WWW.IT target:  http://flander.elet.polimi.it/vim
WWW.IT contact: Alessandro Storti Gajani (alex(at)cerbero.elet.polimi.it)
                Andrea Novara          (andrea(at)cerbero.elet.polimi.it)
                webmaster(at)cerbero.elet.polimi.it [suggested/todo]
WWW.IT info:    max connections: 50
```

KR - Korea

```
WWW.KR alias:   http://www.kr.vim.org/ [todo]
WWW.KR source:  FUB
WWW.KR target:  http://sparcs.kaist.ac.kr/mirror/vim/
WWW.KR contact: Nam Sedong dgtgrade(at)sparcs.kaist.ac.kr
WWW.KR info:    100 max connections; (04:30/15:30 KST)
```

LT - Lithuania

```
WWW.LT dates:   [000411,000605]
WWW.LT place:   Europe, Lithuania, Marijampole; private site
WWW.LT alias:   http://www.lt.vim.org/
WWW.LT source:  ??
WWW.LT target:  http://ice.dammit.lt/vim/
WWW.LT contact: Mantas Mikuckas mancius(at)dammit.lt
WWW.LT info:    daily mirror
```

MX - Mexico

```
MX MEXICO Mirror WWW info:              256 max connections; (CST6CDT)
MX MEXICO Mirror WWW source:    http://www.math.fu-berlin.de/~guckes/vim/
MX MEXICO Mirror WWW target:    http://themis.fciencias.unam.mx
MX MEXICO Mirror WWW vim.org name:      http://www.mx.vim.org/
MX MEXICO Mirror WWW contact:   Felipe Contreras fhca(at)csi.uottawa.ca
MX MEXICO Mirror WWW notes:     Mirrored at midnight Mexico time (GMT-6);
                                dir "beta-test" not included in mirror.

Norway [990308]                 Europe, Norway, City?; Company?
Norway vim.org name:            http://www.no.vim.org/ [toadd]
```

```
Norway Mirror FTP planned:      ???
Norway Mirror WWW info:         ??? max connections; (MET?, 18:30)
Norway Mirror WWW source:       URL
Norway Mirror WWW target:       http://alge.anart.no/vim/
Norway Mirror WWW contact:      Luke Th. Bullock lucc(at)alge.anart.no

PL POLAND Mirror WWW info:      250 max connections; (MET)
PL POLAND Mirror WWW source:    ftp://ftp.oce.nl/pub/vim/
PL POLAND Mirror WWW target:    http://sunsite.icm.edu.pl/pub/unix/vim/
PL POLAND Mirror WWW contact:   sunsite(at)icm.edu.pl
```

SE - Sweden

```
WWW.SE dates:   [000928]
WWW.SE place:   Europe, Sweden, Stockholm; Flow Interactive
WWW.SE alias:   http://www.se.vim.org/ [todo]
WWW.SE source:  ???
WWW.SE target:  ???
WWW.SE contact: David Röhr david(at)flowinteractive.se
WWW.SE info:    ??? max connections
```

SK - Slovakia

```
WWW.SK [990629,000530]          Europe, Slovakia, Bratislava;  "Kotelna"
WWW.SK Mirror WWW info:         100 max connections; (MET); 6:47am
WWW.SK Mirror WWW source:       http://www.math.fu-berlin.de/~guckes/vim/
WWW.SK Mirror WWW target:       http://www.kotelna.sk/vim/
WWW.SK Mirror WWW contact:      Martin [Keso] Keseg keso(at)kotelna.sk

WWW2.SK dates:  [010609]
WWW2.SK place:  Europe, Slovakia, CITY?; "Klifton"
WWW2.SK alias:  Klifton
WWW2.SK source: www.vim.org
WWW2.SK target: http://vim.klifton.sk/
WWW2.SK contact: Marian Stepka m.stepka@bb.telecom.sk
WWW2.SK info:   mirror time at 4am
```

SU - Soviet Union

```
WWW.SU SOVIETUNION [981127,000710]      Europe, Soviet Union, Moscow; Gambit Company
WWW.SU SOVIETUNION Mirror FTP planned:  not yet
WWW.SU SOVIETUNION Mirror WWW info:     150 max connections; (MSK, UTC+3)
WWW.SU SOVIETUNION Mirror WWW source:   http://www.math.fu-berlin.de/~guckes/vim/
WWW.SU SOVIETUNION Mirror WWW target:   http://vim.gambit.msk.su/
WWW.SU SOVIETUNION Mirror WWW contact:  Sergei Laskavy ls(at)Gambit.Msk.SU

WWW.SU SOVIETUNION Mirror WWW info:     200 max connections; (GMT+3)
WWW.SU SOVIETUNION Mirror WWW source:   http://www.math.fu-berlin.de/~guckes/vim/
WWW.SU SOVIETUNION Mirror WWW target:   http://chronos.cs.msu.su/vim/
WWW.SU SOVIETUNION Mirror WWW contact:  WWW admin www-admin(at)chronos.cs.msu.su
```

TR - Turkey

```
WWW.TR dates:   [001013, 001013]
WWW.TR place:   Europe, Turkiye, Ankara; METU/Physics Department
WWW.TR alias:   http://www.tr.vim.org/ [todo]
WWW.TR source:  http://www.vim.org/
WWW.TR target:  http://oberon.physics.metu.edu.tr/vim/
WWW.TR contact: Tolga KILICLI (tolga(at)oberon.physics.metu.edu.tr)
WWW.TR info:    max connections: 50 timezone: Europe/Istanbul

WWW.CA.US dates:        [001021]
WWW.CA.US place:        USA, California, Goleta; redigital.com [ZIP: 93117]
WWW.CA.US alias:        http://www.ca.us.vim.org/ [todo]
WWW.CA.US source:       ??
WWW.CA.US target:       http://vim.redigital.com/
WWW.CA.US contact:      randall ehren randall(at)redigital.com
WWW.CA.US info:         mirroring Sat nd Wed at 7pm -0700
                        running on FreeBSD;
```

Historic Files

Stevie

ftp://ftp.funet.fi/pub/amiga/fish/201-300/ff256/Stevie.lha

Vim was once based on Stevie.

Vim-1.14

ftp://ftp.funet.fi/pub/amiga/fish/501-600/ff591/Vim.lha

The first release of Vim on Fred Fish Disk.

Changes and TODO

```
001013  added www mirror turkey
        moved the generic entries onto the HOWTO Mirror page
        moved the info on language ports onto a page of its own
001002  moved the info on other media    onto a page of its own
```

TODO

- Replace targets with SITE.vim.org aliases - if they exist yet]
- Put all URLs into the same column
- Adjust all entries to the generic format.
- Ask ftp admins to allow "/pub/vim" for accessing vim,
- Add link to sites from the "short list".
- Change entries to allow for extraction for "orig", nickname, "maintainer", "new", and "todo" (*sigh*).

Back to the -> VIM Home Page

Send feedback on this page to
Sven Guckes guckes@vim.org

```
URL:        http://www.math.fu-berlin.de/~guckes/vim/distribution.html
```

URL: http://www.vim.org/dist.html (mirror)
Created: Sun Jan 1 00:00:00 MET 1995

VIM News - What's New with VIM?!

This page informs you about latest changes to the Vim Pages and its development.

If you are too busy checking webpages then you should make use of the free MindIt Service to send you email about changing webpages - like this one. :-)

Vim's History of News

ToAdd:

001106,001127: New document:
The Vim-5.6 Reference Guide is now available in Japanese:
http://www.geocities.co.jp/SiliconValley-Bay/1854/vimrefj.html

010314: New page:
Finnish translation of the short description on Vim in six kilobytes.
Translated by Timo Hiekka timo.hiekka@adcore.com - thanks!

010314: New page:
Polish translation of the short description on Vim in six kilobytes.
Translated by Mikolaj Sitarz mik@fatcat.ftj.agh.edu.pl - thanks!

010314: New page:
Russian translation of the short description on Vim in six kilobytes.
Translated by Bohdan Vlasyuk bogdan@olymp.vinnica.ua - thanks!

010314: New page:

Ukrainian translation of the short description on Vim in six kilobytes.
Translated by Bohdan Vlasyuk bogdan@olymp.vinnica.ua - thanks!

010312: New page:
Greek translation of the short description on Vim in six kilobytes.
Translated by Tzenos Dimitrios tzenos@ceid.upatras.gr - thanks!

010222: New background picture: Vim+Scheme+HeidiKlum:
http://images.themes.org/resources/background.974761406.thu.jpg

010221: New file: The Vim Tutor" in Lithuanian:
http://www.vim.org/download/vim-tutor-lt.txt
Translated by Laurynas Stanèikas lasas@gim.ktu.lt - thanks!

010213/010221: New page:
Hungarian translation of the short description on Vim in six kilobytes.
Translated by Németh Gábor foolman@keszthelyi-rt.hu - thanks!

010207: FAQ: "Making an NT box usable (from a UNIX perspective)"
http://www.cs.colorado.edu/csops/FAQ/NT/31_Usable.html#4.2
Author: Scott A. Morris samorris@cs.colorado.edu
"4.2 Editors : Vi, Vim, Emacs
No OS is complete without vi. Surely you didn't think you
were supposed to use notepad for editting things, did you?
To most unix-users surprise, a fully functional version
of vi is included with nt in the resource kit. .."

010201 New page:
Romanian translation of the short description on Vim in six kilobytes.
Translated by Nastasie George george@cyberspace.ro - thanks!

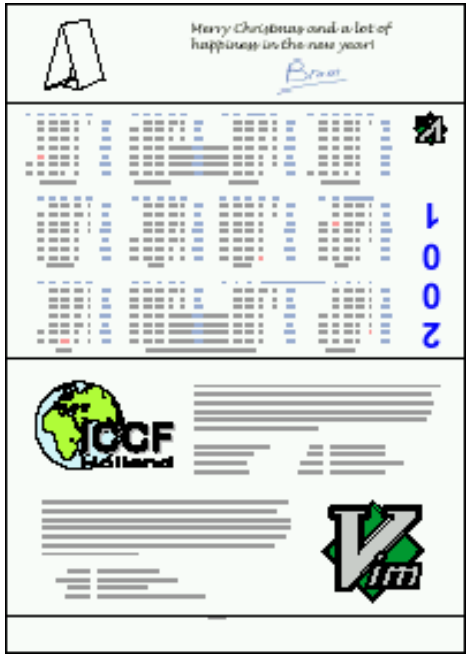
010130: Here's a plug for a well maintained "binary site":
http://www.polarfox.com/vim/
This site offers several Vim binaries for Open VMS -
for both Alphas and VAXes, with and without GUI.
Also, take a look at his screenhots of the Vim-6 version:
http://www.polarfox.com/vim/shot.html

Well done, Zoltan Arpadffy! :-)

010108: New page:
Portuguese translation of the short description on Vim in six kilobytes.
Translated by Douglas Santos dsantos@inf.furb.br - thanks!
010105: Anime Theme with Vim (1152x864; 144K)

010101: Welcome to the 3rd millennium! :-)

001231: New page:
Korean translation of the short description on Vim in six kilobytes.
Translated by kildongi@hitel.net - thanks!



001221: Vim desktop calendar for 2001 - download, print, and fold.
Available in English (A4 and Letter) and Dutch (A4),
as PostScript (compressed with gzip) and PDF.
Made by Bram himself. See his site http://www.moolenaar.net
Merry Christmas, everyone! :-)

Thanks to Toshiya Kawakami kawakami@lead.dion.ne.jp! :-)
001221: Vim desktop calendar for 2001 - download, print, and fold.
Available in English (A4 and Letter) and Dutch (A4),
as PostScript (compressed with gzip) and PDF.
Made by Bram himself. See his site http://www.moolenaar.net

001215: New page: Vad är Vim?
Swedish translation of the short description on Vim in six kilobytes.
Translated by Christian Andersson chrisand@cs.lth.se - thanks!

001106,001127 New page: Vim ¤È¤Ï?
Japanese translation of the short description on Vim in six kilobytes.
Translated by Toshiya Kawakami kawakami@lead.dion.ne.jp - thanks! :-)

001117: New page: Vim æ˜¯ä»€é¼?
Chinese (Big5 encoding) translation of the short description on Vim in six kilobytes.
Translated by Cecil Sheng zcecil@iname.com - thanks!

001122: New page: ¿Qué es Vim?
Spanish translation of the short description on Vim in six kilobytes.
Translated by Jesus M. Castagnetto jesusmc@scripps.edu - thanks!

001121: Change of mailing list address:
vim-fr@club.voila.fr -> vim-fr@egroups.com

001113: Bram Moolenaar's recent talk on
Seven habits of effective text editing
is now available on his website
http://www.moolenaar.net
in plain text, MS-Word, PostScript and PDF.
The presentation was done with PowerPoint.
"You probably want to get both the paper and the presentation,
because the paper contains better text and the
presentation has a few nice pictures."

001112: New page: Cos'e' Vim?
Italian translation of the short description on Vim in six kilobytes.
Translated by Stefano Lacaprara Stefano.Lacaprara@pd.infn.it - thanks!

001109: New page: Ê²Ã´ÊÇVim?
Chinese (Mandarin) translation of the short description on Vim in six kilobytes.
Translated by Weiguang Shi wgshi@cs.ualberta.ca - thanks!

001027: New helppage: "version6.txt"
An overview to new features and changes for Vim-6.

001013: New page: Moved the info about Language Ports onto a new page:
http://www.vim.org/langport.html

001011: New article:
The continuing story of Vim (local copy)
The paper submitted by Bram Moolenaar for the Linux2000.nl conference,
taking place 9-10 Oct 2000 in Ede (Netherlands),
which gives a short overview to the development history
of Vim and also on current development.

001010: Bram Moolenaar will speak at the Linux2000 conference:
Event: Linux2000 conference http://www.linux2000.nl
Place: "De Reehorst" (conference centre), Ede (town), Netherlands
Date: Tuesday October 10
Time: 12.10 - 12.55
Title: *The continuing story of Vim*
Note: Entrance is free, but you do need to register in advance.

001008: Moved the "Vim Challenges" from the home page
onto a page of its own: VIM Challenges

001008: New magazine article:
September 2000 - "Software Design" [ISSN 0916-6297]
http://www.gihyo.co.jp/SD/index-j.html
Author: Takuhiro Nishioka takuhiro@super.win.ne.jp
Contents: "Hajimete no Vim (Learning the Vim editor)"
Language: Japanese
Scan of article:
http://www.win.ne.jp/~takuhiro/image_files/vim_software_design.png

001005: Join the *Vim Webring*!
Full List: http://nav.webring.yahoo.com/hub?ring=vim&list
HTML Code: http://sites.netscape.net/pottsdl1/webring.html
Created by Douglas Potts pottsdl@bigfoot.com on 000823.

001002: New page: Vim on media (CD, DVD)
Vim is also available on some CDs and DVDs.
Useful to know when your connection to the
ftp mirrors is currently not working... ;-)

001001/001003: Updates:
Vim, c'est quoi? (French)
Wat is Vim? (Dutch)
Translations of the short description on Vim in six kilobytes.
Translated by Lyderic Landry lydericlandry@yahoo.fr
and Dion Nicolaas dion@erebus.demon.nl - thanks!
Further translations are very very welcome!

```
000921: KVim - Vim for KDE ('K' Desktop Environment)
        Developer:   Thomas Capricelli capricel@yalbi.com
        Page:   http://aquila.rezel.enst.fr/thomas/vim/
        Please give Thomas some feedback!

000904: Vim Color Editor HOWTO:
        http://www.linuxdoc.org/HOWTO/Vim-HOWTO.html
        by Alavoor "Al Dev" Vasudevan alavoor@yahoo.com
        Last changed:  v14.0, 16 Aug 2000
```

000704,000807: New article on LinuxNewbie.org: "Introduction to Programming in C/C++ with Vim" http://www.linuxnewbie.org/nhf/intel/programming/intro_c++.html Author: Keith Jones kmj9907@cs.rit.edu [Tags, C-style indenting, QuickFix, useful keystrokes (jumps), substitution, misc features, links] 000626: VINE-0.30 released (requires Perl) (Mail and News from within Vim - "Hello, Gnus!"). http://www.mossbayeng.com/~ron/vim/vine.html (Actually, this was released on March 31st... oops!) 000624: Release of Vim-5.7. 000420: Update on the "Vim Guide"!! :-) Oleg Raisky has updated his "Vim Guide" for Vim-5.6. This basically is the "Quick Reference" (":help quickref") which contains an overview to the commands of Vim. But Oleg has extended this such that you can distinguish keys from letters and some useful hints. The guide is about 50 pages and is available for A4 or US letter formats in "plain" or "booklet" (very handy) in PostScript, so it's ready for printing. Get it now from Oleg's page http://physlab.sci.ccny.cuny.edu/~orycc/vim-main.html or from the vim main site at ftp://ftp.vim.org/pub/vim/doc/ (hmm.. have the mirror caught up on this yet?) [Note: The files are compressed with bzip.] You can also download the guide as a PDF now: http://www.eskimo.com/~drisner/vim/ made by David Risner. Thanks, guys!

2000 November

001109: Bram Moolenaar speaks at the NLUUG conference:

Place: "De Reehorst" (conference centre), Ede (town), Netherlands

Title: NLUUG Autumn Conference http://www.nluug.nl

[NetherLands Unix User Group]

Date: Thursday November 9th

Time: 15:30 - 16:15

Title: *Seven habits of effective text editing*

Note: Entrance *might* be free - it certainly is free for members of the NLUUG. Some talks will be in English - but not all.

2000 September

000922: New page: Wat is Vim? Dutch translation of the short description on Vim in 3,000 bytes. Translated by Dion Nicolaas dion@erebus.demon.nl - thanks!

000921: Vim is available for KDE ('K' Desktop Environment) as "KVim". The developer is Thomas Capricelli capricel@yalbi.com http://aquila.rezel.enst.fr/thomas/vim/

000915: New page: http://www.vim.org/features.fr.txt French translation of the short description on Vim in 3,000 bytes. Translated by Lyderic Landry lydericlandry@yahoo.fr - thanks!

000911: New Page: VIM Documentation Overview An overview to online documentation about Vim. http://www.vim.org/docs.html

2000 August

000830: Vim-5.7 Helpfiles are now online.

2000 July

000729: WAN event (We Are Networking) [Jul28-Aug02]. On 14:30 Bram Moolenaar gives a lecture/presentation on Vim

- http://www.wan.nl

000701: LinuxMagazine choses www.vim.org as one of the Top 100 Linux Web Sites. :-)

2000 June

000630: LinuxTag 2000 in Stuttgart - Talk about "Vim for Vi Users"

000624: Release of Vim-5.7.

2000 May

2000 April

000426: Coming up: A helpfile on the multibyte support. (This file is now included with vim-5.7.)

000420: Update on the "Vim Guide"!! :-) Oleg Raisky has updated his "Vim Guide" for Vim-5.6. This basically is the "Quick Reference" (":help quickref") which contains an overview to the commands of Vim. But Oleg has extended this such that you can distinguish keys from letters and some useful hints. The guide is about 50 pages and is available for A4 or US letter formats in "plain" or "booklet" (very handy) in PostScript, so it's ready for printing. Get it now from Oleg's page

- http://physlab.sci.ccny.cuny.edu/~orycc/vim-main.html or from the vim main site at
- ftp://ftp.vim.org/pub/vim/doc/ (hmm.. have the mirror caught up on this yet?) [Note: The files are compressed with bzip.]

You can also download the guide as a PDF now: http://www.eskimo.com/~drisner/vim/ made by David Risner. Thanks, guys!

000419: New page: HOWTO Install Vim http://www.vim.org/howto/install.html Actually, it's just s small outline which I apparently never announced. Feedback, please!

000416: Remember: If you need help on Vim *instantly* then you can talk to us via IRC (Internet Relay Chat) on channel "#vim".

000412: The Vim code now has a repository on cvs.vim.org. Finally move the (little) info onto a page of its own: New page: http://www.vim.org/cvs.html

000406: Vim-5.5 is now bundled with Solaris 8 (package "SFWvim"). See http://www.sun.com/solaris/freeware.html

000403: New page: Chat Page http://www.vim.org/chat.html "Let's talk about Vim!" A web interface to IRC - please try it!

000403: Bram Moolenaar has been to Uganda again, visiting the children at the centre in Kibaale. Here is his report: http://www.vim.org/iccf/news.html

2000 March

000331: VINE-0.30 released (requires Perl) (Mail and News from within Vim - "Hello, Gnus!").

- http://www.mossbayeng.com/~ron/vim/vine.html

000327: Vim.org gets added to Linux-Directory.com. 

000327: New page: Search Vim Pages http://www.vim.org/search.html (powered by atomz.com)

000325: www.vim.org is now a faster machine. :-)

000320: PCRevue interviews Bram Moolenaar

000317: New tutorial - in French! http://www.student.info.ucl.ac.be:8080/HomePages/fleurial/vim-tutor.html Author: Junior Frederic Fleurial Monfils fleurial@hotmail.com

000307: New Page: Vim Announcements http://www.vim.org/announce/ Alright - it's not new. But maybe you did not know about it. ;-)

000307: New Page: Vim Sites http://www.vim.org/sites.html Sites (domains) where Vim was officially installed for use.

000303: New Page: Vim Acronyms http://www.vim.org/acro.html

Do you have another expansion for the acronym "VIM"? Let me know!

000301: The Vim Helptexts for vim-5.6 are are now online in HTML: http://www.vim.org/html/

You can also see the syntax files online, too, now: http://www.vim.org/syntax/

Copies of new syntax files which are not in the distribution yet can be accessed at http://www.vim.org/syntax.new/ and via the languages page.

2000 February

000222: New page: Vim Evaluation Moved the current stats on the Linuxcare's poll on Text Editors onto a page of its own.

000208: Vim wins Slashdot.org's "Beamie Award": ".. the Best Open Source Text Editor went to vim." see article on slashdot

2000 January

000121: Bram and Vim at LinuxWorld (Tucows booth) - (Feb 1st - 4th)

"In the first week of February, there will be a LinuxWorld conference and expo in New York city. I will be there, in the Tucows booth. There will be a couple of presentations about Vim each day. I'll be showing off Vim too."

If you attend LinuxWorld, stop by at the Tucows booth to say hello. Teaser: We will be handing out free CDs! :-)

More information: http://www.linuxworldexpo.com

1999 December

991129: There is now a Vim Color Editor HowTo on linuxdoc.org

991223: The homesite of the Vim Pages, math.fu-berlin.de, was hit by crackers, apparently, Therefore the connection to the outside world has been cut. No service until January. Sorry, folks!

991219: First beta of vim-5.6 - vim-5.6a.

991216: The Vim code now has a repository on cvs.vim.org.

991208: The Vim Book

Let's write a book about Vim *together*. It shall be in DocBook format - contributions and ideas are welcome! Release date? Hopefully before Xmas 2000.

1999 November

991129: A new Vim HOWTO:
http://www.linuxdoc.org/HOWTO/Vim-HOWTO.html

Author: Alavoor Vasudevan alavoor@yahoo.com

990921-991017: Sven is on holidays in California. Almost no service.

1999 September

990921: Vim-5.5 has been released.

990909: Mandrake of Enlightenment fame uses Vim!

> Jeremy: What type of machine is your main development machine? How many monitors does it have? What editor do you use?
>
> Mandrake: well... to be honest my primary development machine is a hoss. But I compile all day long, which is why I keep it nice at fat like that. I have a dual p3/500 with 512 megs of ram and 3 16 mb video cards with 3 21" monitors on it. **I use vim** :)

990904: New MacOS port for vim-5.4.49 and vim-5.5a:

http://www.tu-bs.de/~i0080108/macvim.html

1999 August

990831: New page about installing vim with an example to the installation on Solaris. Feedback, please!

990824: New Binary Site: Alpha Win32.

990822: New utility: boxes. This little filter draws a box around some given text - and it also able to remove it from "boxed text", too. Check out the page with examples which probably describe this best.

990807: Vim-5.4 for Win16 available. **Warning:** This version has not yet been tested much! For more info see the announcement

990803: New logo, created by Bram.

1999 July

990729: New utility - tblfmt ("table format"). This turns a block of text containing lists of table entries into a proper table using spaces for padding.

990727: Vim gets a nomination in LinuxWorld's "Editors' Choice Awards" as a finalist in the category "Text Editing" together with Emacs. The winners will be announced on Aug 11th.. Go, Vim, go! :-) [article]

990725: Vim-5.4 finally gets released - after eleven months of development!

990720: The book "Learning the Vi editor" has now been released in German: Textbearbeitung mit dem

vi-Editor

990701: The mailbox is stagnating around 1,000+ mails - 400+ of them about vim. I am slowly answering the emails that came in around March and April. Please be patient - I will answer all your emails eventually.

1999 June

990621: Tim O'Reilly is using Vi

990621: New vim utility: vmh - the "vim mail handler". Vmh allows to write and manage emails from within Vim. This is for Unix systems and requires gpg, perl, mh, metamail, and sendmail. [One of these days there should be a conference on Vim - even if only to see how people are using Vim and the utilities, right?! ;-)]

1999 May

990504: I am trying to add the changes that people mailed me in March and April. Patience, please!

1999 April

The offline time of math.fu-berlin.de creates more problems. Reinstalling programs takes time... *sigh*

1999 March

990312-990409 The domain math.fu-berlin.de was offline. I could not update the pages. the page on maillists has some more info.

More icons and screenshots.

990310: New web mirror in Austria - both for ftp www.

990308: New web mirror in Norway

990305: Vim HomePage down - a summary of the experience

990303: HowTo Maintain a Syntax File

1999 February

990205: Added an extra page for Vim binaries. Hopefully more people will volunteer to maintaina binary for their favourite operating system then.

Finally updated the look on the Vim homepage. It now uses some tables. Does it look ok now?

1999 January

Complete overhaul of the distribution page.

990129: Added an extra short page of FTP and WWW mirrors.

1998 December

981204 - The SMIL tutorial

http://www.helio.org/products/smil/tutorial/

This tutorial about SMIL was written with VIM. :-)

981203 - Updated page: VIM6 vote for features - results

About 400 people took part in sending in their votes for features that should be in Vim6. The top requested feature to be with VIM6 is "folding" which basically hides parts of the text and thus allows to focus on just some parts of the text. The runner-up is "vertical split" which allow to split windows vertically and thus to show files side-by-side. The next two things are adding more features to auto-indent code in more programming languages and making Vim "more robust". People then requested to add searching across line boundaries and improve the speed of syntax coloring. And last not least, everyone seems to want a better interface for selecting buffers. The last two items are: "stop changing Vim" and "remove functionality". What did you expect? ;-)

1998 November

981126 - final day to vote for your favourite features with Vim-6.

981116 - New page: news.html

This page. It's better to keep the new news apart from the homepage, eg for monitoring with NetMind. And it removes a few kilobytes from the vim homepage, too.

981106 - New page: VI UnixDist -

Unix releases (mostly "free" Unixes) and their Vi versions.

1998 October

981029 - Vim-6 features - call for votes:

VIM's author, Bram Moolenaar, has sent out a message asking for your vote which asks for a list of features that *might* be implemented for Vim-6.

980918 - Helpfiles updated:

The helpfiles for vim-5.3 are now available. See http://www.vim.org/html/.

You can download a compressed (GNU zip) archive of these, too:

Vim-5.3 Docs in HTML **(633K)**

(Should I offer these for "pkzip", too?)

980915 - Support for Hangul (Korean)

The patches are just 8K. Real soon now! ;-)

1998 July

980723 - New page: Languages

The list with available languages aka syntax files. Please note that a language need not be a programming language - it can be anything that has some structure. For example, there are some syntax files for the "language" used within several program setup files. So why not submit a syntax file for your favourite "language"?

```
URL:         http://www.math.fu-berlin.de/~guckes/vim/news.html
URL:         http://www.vim.org/news.html (mirror)
Created:     Mon Nov 16 12:00:00 CET 1998
```

Send feedback on this page to
Sven Guckes guckes@vim.org

Search Vim.org

This is the search page of vim.org using the search engine by atomz.com.

Feedback on this service is much appreciated. Is the speed of the search ok? Did it find the pages you needed? Send your comments to me at guckes@vim.org - thanks!

SEARCH!

Search For:

Search

Match: ◉ Any word ○ All words ○ Exact phrase

☑ Sound-alike matching

Within: Anywhere

Show: 10 results with summaries

Sort by: score

Top Search Words so far:

```
gvim vimrc download syntax color
```

... and some more:

```
ctags dos download java perl php python
replace tab tutorial vimbuddy visvim windows
```

See Also:

- Search Statistics

```
URL:          http://www.math.fu-berlin.de/~guckes/vim/search.html
URL:          http://www.vim.org/search.html (mirror)
Created:      Mon Mar 27 12:00:00 MET DST 2000
```

Send feedback on this page to
Sven Guckes guckes@vim.org

Sven Guckes guckes@vim.org

Last update: Sat Apr 21 03:03:03 MEST 2001

VIM Pictures - Buttons, (Animated) Icons, and Screenshots

This page shows buttons, icons, and some screenshots of Vim and GVim.

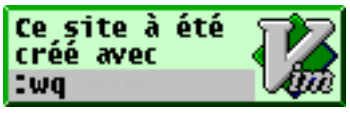
Feel free to add the buttons and icons to your webpages. A link to www.vim.org would be appreciated, but that is no requirement, of course.

Do you have a pictures to add to this page? Then please send it to me via email! But please take a look at the submission guideline below as it saves me quite some work - thanks! :-)

Vim Pictures - Overview

Sections on this page:

Vim - Icons and Logos



"crée avec Vim" [only 881 bytes!]

Creator: Eric Lebel lebelcire@hotmail.com [010302]

Notes: Crafted from the icon by Marc Hughes.

"Created with Vim" [3,109 bytes]

Creator: David Necas yeti@monoceros.physics.muni.cz [001215]

Notes: The XCF source is available at http://physics.muni.cz/~yeti/download/created-with-vim.xcf.bz2

"created with Vim" [1,914 bytes]

Creator: Philipp Neuner p9r@gmx.at [000817]

"Tilde Line Icon" [**293** bytes]

Creator: Mark Tomko mtomko@oucsace.cs.ohiou.edu [000815]

Notes: This icon is in PNG format and shows a "tilde line" with a command line ":Vi IMproved":

"Created with VI" [19940 bytes]

Creator: Stephen "The Saint" Zeck saintly@innocent.com [000704]

"Vim Document Icon" [**259** bytes]

Creator: Nathan Leveck n473@n473.com [000509]

Notes: This icon is also available as a Windows icon file (**766** bytes).

"This Site vim Powered" [**948** bytes]

Creator: Andy Bradford bradipo@xmission.com [000503]

"I love my editor" [1,368 bytes]

Creator: Alexandre VIALLE alexvialle@yahoo.fr [000427]

Notes: Small, nice - *love* it! :-)

"Edited with VIM" [1,029 bytes]

Creator: Preben Randhol randhol@pvv.org 000222

"Vim on Fire" [6884 bytes]

Creator: Daniel S Gindikin gindikin@andrew.cmu.edu

Notes: HOT! :-)

"VIM - the 4 star editor" [only **972** bytes!]

Creator: Herve Foucher Herve.Foucher@helio.org [990826,990903]

Notes: Small! Just under 1K!

"Vim - The editor" [1,091]

Creator: Bram Moolenaar bram@vim.org

Notes: Logo created by the author himself! (I guess that makes it the "official" logo now. ;-)

"Vim Mirror" [2362 bytes]

Creator: Byron L. Ewers blewers@tencorp.net [990708]
Notes: Certainly a nice button for Vim Mirrors. :-)

"Vim Mirror" [2700 bytes]

Creator: Byron L. Ewers blewers@tencorp.net [990708]
Notes: A variant of the first button - but the green comes in vertical layers.

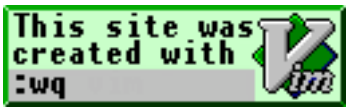
"V on green square" [only **687** bytes!]

Creator: Herve.Foucher@helio.org
Notes: Small. Nice.

"Vim Icon (transparent)" [only **295** bytes!]

Creator: Thomas.Hopfner@lpr.e-technik.tu-muenchen.de [990203]
Notes: Small. Transparent. Vim. Good!

"This site was created with VIM! :wq" [918 bytes]



Creator: Mark Hughes kamikaze@kuoi.asui.uidaho.edu [981217]
Notes: "ZZ" works, too. ;-)

"Edited with Vim" [1199 bytes]

Creator: Marco Lamberto lm@geocities.com [981111,981117]
Notes: Red background. Nice and small.

"Just VIM it! (This page is Vim POWERED!)" [6456 bytes]

Creator: Adrian Nagle vim-www@naglenet.org [981105,000524]
Notes: Nice. :-)

"Just VIM it! (click here)" [6737 bytes]

Creator: Adrian Nagle vim-www@naglenet.org [981104,000524]
Notes: Nice. :-)

"Vi Vimproved" [2895 bytes]

Creator: Prabhanjan M prabham@prithvi.siliconsystems.co.in [981001]
Notes: Nice. :-)

"Designed with Vim" [1501 bytes]

Creator: (damn, I forgot) [980622]
Notes: Black button with a dark blue "vim" on the white text "designed with".

"Vim Hot Icon" [299 bytes]

Creator: Vince Negri vn@aslnet.demon.co.uk [980216]

"This site vim powered" [411 bytes]

Creator: Cade, Philippe phc@gmx.net [971127]

"This site vim powered" [299 bytes]

Creator: Kristine Budde budde@inf.fu-berlin.de [980721]

"This site is powered by vim" [4885 bytes]

Creator: Digital Puer digital_puer@hotmail.com [970828]

VIM - Screenshots

So - here are the screenshots. They are linked one by one, so as not to take a lot of load for this page. Have fun!

To Add:

- http://www2.raidway.ne.jp/%7Eyzone/bin/vim60t-cygwin-telnet.jpg

Title: GVim-6.0aa on WindowsNT4

Size: 25931 bytes

Creator: David Pascoe David.Pascoe@jtec.com.au

Added: 010420

Notes: GVim-6.0aa on Windows-NT 4.0 with vertical split. The left windows shows the file the original version of a file (g7mibsup.orig.c) and the right windows shows the current version (g7mibsup.c) with three additional lines (colorized with a cyan background). There is also a torn off menu showing the commands "undo, paste, select word/line/all".

Title: Vim-6 C++ Jave Make Email HTML

Size: 38121 bytes

Creator: Eric Tetz erictetz@yahoo.com

Added: 010325

Title: Vim-6.0w with UTF-8 text

Size: 18058 bytes

Creator: Alexey Marinichev lyosha@math.ucr.edu

Added: 010313

Notes: Shot shows Vim-6.0w within an xterm from the XFree86-4 distribution. The text is in utf-8 mode with a font called

-misc-fixed-medium-r-semicondensed--*-120-*-*-c-*-iso10646-1* The text was taken from the file "ucs-fonts/examples/quickbrown.txt" from the ucs-font distribution.

Title: Vim on VMS

Size: 105K and 135K

Creator: Zoltan Arpadffy arpadffy@altavista.net

Added: 010130

http://www.polarhome.com/vim/shot.html

Two screenhots of VIM-6 with vertically split windows. One shot shows the GUI version with menus and syntax coloring, the other shows the console version, again with "vertical split" but without coloring. These shots were made to show that Vim on VMS is just as nice as anywhere else. :-)

Title: Vim-5.7.2 Startup Screen

Size: 5,088 bytes

Creator: Philippe Seidel pagin@newopenness.de

Added: 001104

Notes: The pure startup screen of Vim with the intro message.

Title: Japanese MacVim

Size: 156K

Creator: Kenichi Asai asai@is.s.u-tokyo.ac.jp

Added: 000509

Notes: This screenshot shows that Vim's "multibyte support" allows to display Japanese text on your screen. See also Kenichi Asai's page on "Japanese Vim":

- http://www-imai.is.s.u-tokyo.ac.jp/~asai/macvim/

Title: Gvim Japanese

Size: 30K

Creator: Taro Muraoka koron@tka.att.ne.jp

Added: 000116

Notes: Japanese text in both edit buffer and menus.

Title: When things are dim - Give them VIM

Size: 18371 bytes

Creator: Triantafyllidis Petros trian@lemnos.geo.auth.gr

Added: 990816

Notes: This picture was found in a museum in Barcelona.

Title: Vim and SMIL

Size: 15,390 bytes

Creator: Herve.Foucher@helio.org

Added: 990804

Notes: This picture shows one of the files that are part of the tutorial on SMIL. The green right angle in line two represents the position of the cursor.

Title: Vim with GTK theme

Size: 136K [1038x726 !]

Creator: Patrick Wagstrom wagspat@charlie.cns.iit.edu

Added: 990301

Notes: "Built right off the vim 5.4e source tree with no modifications. I just used a GTK theme and a pretty big .vimrc file to create that." Please note that the configuration dialog was taken out from Vim again. It no longer exists with the source of Vim. Sorry.

Title: Vim with NeXT look

Size: 25K [727x500]

Creator: Dan Moniz dnm@cadre.org

Added: 990220

Notes: Vim-5.4e built with Gtk+ 1.16 and Glib 1.16 on Linux Redhat-5.1 with WindowMaker-51.0.

Title: Vim on MacOS 8.5.1

Size: 18,517 bytes

Creator: W. Jeffrey Rankin jrankin@greenteainc.com

Added: 990106

Title: Vim in SQL QuickFix mode (1)

Size: 58,700 bytes

Creator: Rajesh Kallingal RajeshKallingal@email.com

Added: 981223

Notes: This picture shows the SQL*Plus menu which lets me run the SQL*Plus or PL/SQL script files in full or just the selected text or start SQL*Plus. Also added QuickFix mode for PL/SQL files using a perl script. This allows me to do a ':Compile' to compile PL/SQL Stored Procedures and put me in QuickFix mode to fix all the errors (if any) in the Stored Procedure. The perl script can be run from command line and it can invoke VIM (with OLE) to debug the errors.

Title: Vim in SQL QuickFix mode (2)

Size: 66,778 bytes

Creator: Rajesh Kallingal RajeshKallingal@email.com

Added: 981223

Notes: This shows VIM with an PL/SQL error *after* a ':Compile'.

Title: gvim-5.4a with GTK

Size: 29291 bytes (721x449)

Creator: Marcin Dalecki dalecki@cs.net.pl

Added: 981218

Notes: The first vim screenshot with 27 command icons. ;-)

Title: GVim-5.3

Size: 25330 bytes (587x546)

Created: David Pascoe David.Pascoe@jtec.com.au

Added: 981023

Notes: Gvim-5.3 on WindowsNT-4.0, showing some icons for commands and an example of the new "browse" command with a dialog on the directory with Vim's syntax files. Vim shows a split screen - the upper window one shows the file "os_w32.txt" (with specific help on the Win32 binary), and the lower window shows $VIM/syntax/cpp.vim - the syntax file for C++ files (colorized by the applied vim.vim, of course).

Title: Vim in SQL Mode

Size: 71K

Creator: Rajesh Kallingal Rajesh.Kallingal@cyberdude.com

Added: 980904

Notes: This shows the SQL*Plus menu which lets me run the SQL*Plus or PL/SQL script files in full or just the selected text or start SQL*Plus.

Title: Vim-5 on RiscOS

Size: 74K

Creator: Thomas Leonard tal197@ecs.soton.ac.uk

Added: 980726,980728

Notes: Shows a choice of font styles and sizes. The back window is using direct-to-screen plotting via the ZapRedraw module, the other is plotting via the font manager.

Title: Gvim and Unicode

Size: 60K

Creator: Sung-Hyun Nam namsh@lgic.co.kr

Added: 980717

Notes: Gvim-5.2g with +GUI_Athena, showing the Hangul Latex Guide with text in Korean, Japanese, and Chinese, and a few Roman and Greek characters, on WindowManager "Enlightenment-14 WM", with font "-misc-gothic-medium-r-normal--16-160-75-75-c-160-ksc5601.1987-0".

Title: VIM-5.0w on rxvt-2.4.3

Size: **136K**

Creator: Matthew Aylward maylward@sunny.gis.uwa.edu.au

Added: 980626

Notes: VIM-5.0w on rxvt-2.4.3 on a "walnut" background with windowmaker-0.14.0.

Title: VIM-5.0w on rxvt-2.4.3

Size: **58K**

Creator: Matthew Aylward maylward@sunny.gis.uwa.edu.au

Added: 980626

Notes: VIM-5.0w on rxvt-2.4.3 on a background with "blue waves" with windowmaker-0.14.0.

Title: GVim-5.1a

Size: 15025 bytes

Created: David Pascoe davidp@jtec.com.au

Added: 980331

Notes: Windows' window with vim icon and gvim-5.1a inside; ":e $VIM\syntax\tex.vim" and ":help gui-w32".

Title: VIM-5.0l on Fortran

Size: 11K

Creator: Leonard Lorentzen lnl@ffa.se

Added: 971128,980626

Notes: VIM-5.0l +GUI_Motif +X11

Title: VIM Life

Size: 8K

Created: Kent Nassen knassen@umich.edu http://www-personal.umich.edu/~knassen/

Added: 971006

Notes: This picture shows VIM while performing the macros for "Game of Life".

Title: VIM-4.6 with Farsi fonts

Size: 12K

Creator: Mortaza G. Shiran shiran@parscom.com http://www.scn.de/~shiran/

Added: "long ago"

Notes: "Vim-Farsi contains a converter that converts between Standard Farsi 3324 and Vim-Farsi. So you can use it to write Farsi HTML documents for PMOSAIC browser."

Title: Vim after startup

Size: 10K

Creator: Keith Manley kmanley@verinet.com

Added: 971217

Notes: Gvim-5.0s running under HP-UX 9.05 on a HP 715 right after startup. As no filename is given, you see the nice greeting screen that the author created to give initial help to a new user.

Title: Vim on syntax file tex.vim

Size: 14K

Creator: Keith Manley kmanley@verinet.com

Added: 971217

Notes: Gvim-5.0s - the current buffer contains the syntax file "tex.vim" which defines the rules for coloring TEX files. The file itself is colored according to the rules given by "vim.vim".

Title: Vim on tex.vim - with menu

Size: 21K

Creator: Keith Manley kmanley@verinet.com

Added: 971217

Notes: Gvim-5.0s on "tex.vim" again - but this time with the default tear off menus (Files) on top of the window.

Title: VIM-? with TeX menus

Size: 13K

Creator: Hermann Rochholz Hermann.Rochholz@gmx.de

Added: 970627,980703,000921

Notes: Is the LaTeX menu available on the Web yet?

Title: VIM-4.4 with X support

Size: 11K

Creator: Felix von Leitner leitner@math.fu-berlin.de

Added: 960926

Notes:

Title: VIM-4.2 with X support

Size: 20K

Creator: Sven Guckes guckes@vim.org

Added: 960926

Notes: This shows VIM with three windows: The top window shows "VIM" in a four line ascii font - "[+]" indicates that the buffer was modified and "[No file]" indicates that the buffer is not associated with a filename yet. The window in the middle shows VIM'd online help text - it is opened automatically when you enter the command ":help". The window at the bottom shows a mail I was writing; the command on the command line will pass on the text to the mailer MUTT which will send send it off.

Title: VIM on Windows (from DOS shell)

Size: 6889 bytes

Creator: Greg Willard willard@simon.wustl.edu
http://dybfin.wustl.edu/willard/willard.html

Added: 961016

Notes: VIM on source code for C++ with one paragraph highlighted.

Title: VIM-3.0 on Macintosh

Size: 4703 bytes

Creator: Patrick Berry pberry@node8.com

Added: 970207

Notes: This picture show VIM-3.0 on MacOS after the startup and a ":version".

Title: GUI VIM on Perl script

Size: 9784 bytes

Creator: Lawrence Clapp larry@theclapp.org

Added: 971104

Notes: VIM editing on a perl script.

Title: VIM color on Perl

Size: 17K

Creator: Christopher G Mann r3cgm@cdrom.com

Added: 970905

Links to other pictures on the net

Vim on Olaf's Desktop [971216]

http://www.polderland.nl/~rhialto/be/vim-5.0s-screen1.gif (71K)

Author: Olaf Seibert rhialto@polder.ubc.kun.nl

Vim on Jano's Desktop [970919,000807]

http://www.wi.leidenuniv.nl/~jvhemert/MyDesktop.html

Screenshot of 1997:

http://www.wi.leidenuniv.nl/~jvhemert/home/desktop.19970226.big.jpg

Screenshot of 1998:

http://www.wi.leidenuniv.nl/~jvhemert/home/desktop.19981129.big.jpg

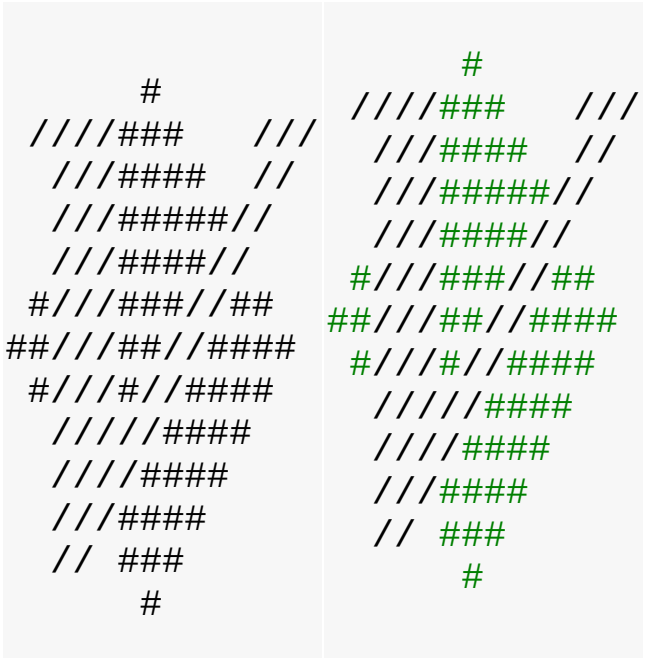
http://www.wi.leidenuniv.nl/~jvhemert/desktopb.jpg [obsolete?]

Author: Jan van Hemert jvhemert@wi.LeidenUniv.nl

There's Vim on his desktop - try to spot it!

Vim - ASCII Logos

VIM ASCII Logo by Woody Thrower [991020,991026]:



```
___ ___ _______ _______ ___ ___ __ __ | | ||_ _|| | | | | ||__|.--------. .--.--.|__|.--------. | | | _| |_ | | |
| || || | | | || || | \_____/ |_______||__|_|__| \_____/ |__||__|__|__| \___/ |__||__|__|__| __ __ ___ __
__ __ __ _ _ \ \ / /|_ _|| \/ | \ \ / /(_) _ __ ___ __ __(_) _ __ ___ \ \ / / | | | |\/| | \ \ / / | || '_ ` _ \ \ \ / /|
|| '_ ` _ \ \ V / | | | | | | \ V / | || | | | | | \ V / | || | | | | | \_/ |___||_| |_| \_/ |_||_| |_| |_| \_/ |_||_| |_| |_| _ __
____ __ ___ _ __ _ _ | | / // _// |/ / | | / /(_)____ ___ _ __ (_)____ ___ | | / / / / / /|_/ / | | / // // __
`__ \ | | / // // __ `__ \ | |/ /_/ / / / / / | |/ // // / / / / / | |/ // // / / / / / |___//___//_/ /_/ |___//_//_/ /_/ /_/
|___//_//_/ /_/ /_/ __ __ ___ __ __ __ __ _ _ \ \ / /|_ _|| \/ | \ \ / /(_) _ __ __ __(_) _ __ \ V / | | | |\/|
| \ V / | || ' \ \ V /| || ' \ \_/ |___||_| |_| \_/ |_||_|_|_| \_/ |_||_|_|_| _ __ _____ __ ___ _ __ _ _ | | / // _// |/
/ | | / /(_)__ _ _ __ (_)__ _ | |/ /_/ / / /|_/ / | |/ // // ' \ | |/ // // ' \ |___//___//_/ /_/ |___//_//_/_/_/
|___//_//_/_/_/ _ _ _ ___ ___ _ _ _ _ | | || || \ \ | | |<_>._ _ _ _ _ <_>._ _ _ | ' || || | | ' || || ' ' | | | | || ' ' |
|__/ |_||_|_|_| |__/ |_||_|_|_| |__/ |_||_|_|_| _____ _____ _____ _____ _ _ | | || || | | | ||_| _____ _ _ |_|
_____ | | ||- -|| | | | | | || | | | | || || | \___/ |_____||_|_|_| \___/ |_||_|_|_| \_/ |_||_|_|_| Submitted by:
Manfred Gruber manfred@gruber.priv.at [990303] __ ___ ___ _____ _____ | |\ / /\ / /\ / |\ / |\ | || / /
/ /__/ / / || / || | || / / / \__\/ / /| || / /| || | || / / / ___ / / | || / / | || | || / / / /\ / / /| || / / /| || | || / / / / / / / / / | ||/
/ / | || | || / / / / / / / / / | |/ / / | || | || / / / / / / / / / | / / | || | ||/ / / / / / / / / | / / | || | |/ / /
/ / / / / / |__/ / | || | / / / / / / / / \__\/ | || | / / / / / / / / | || | / / / / / / / / | || |__/ / /__/ / /__/ / |__|| \__\/
\__\/ \__\/ \__\/ \__\|
```

Vim - HOWTO add your pictures to this page

Do you have a pictures to add to this page? Then please send it to me via email!

Here are some things to take care of [I hope you'll understand]:

Submission guidelines

Send address or attach

> If your send a screenshot then please do not attach your picture to your email; instead, just tell me whwre I can download it. So please upload you pictures to your own website and just send me the URL (address) of the picture. That way your pictures are not within a mail in my mailbox, saving me time on download and startup of the mailer.
>
> If you cannot upload your pictures to a website then please **attach** the picture to your mail; this allows for easy extraction of the data to a file.

Format: GIF, JPG, PNG

> Please use a format known by many browsers, so I won't have to convert them. GIF (GIF89a, interlaced), JPEG (use a suitable compression rate) and PNG are all welcome.

Maximum Size 640x480

> Please try to take screenshots from 640x480 screens at most. I'd like to the pictures to be viewable on 14" monitors, too.
>
> That said, larger screenshots are welcome, too, but you'll have to keep in mind that those who will view it on a 14" monitor will have to scroll around a lot to see it all.
>
> Please let me know about the size of your picture; of course there are tools to find out about the size, but it would make it easier on me if you simply added this info to your mails.

Filenames

> Please give your pictures a filename which starts with "vim", includes your last name and the title of the picture, ie in the format *vim.yourname.description.gif*,
>
> Example: *vim.guckes.edit_with_vim.gif*

Screenshot Contents: Show Version Number

> Please have your vim show its version number on the screenshot. (enter command ":version" or ":intro" in a subwindow). If possible, show the coloring of some non-standard file - as there are quite a few shots of source files in C, C++, and Fortran.

Name and Address [000818]

> I will add your name and address as given in the From: line of your email. If you want some other address to appear on this page then please let me know. But I will certainly add your real name. :-)

THANKS! :-)

Vim Pictures - History (List of Changes)

- 010420: Added GVim-6.0aa on WindowsNT4
- 010313: Added Vim-6.0w with UTF-8 text
- 010302: Added crée avec Vim
- 001215: Added Created with Vim
- 001104: Added Vim-5.7.2 Startup Screen
- 000817: Added created with Vim
- 000815: Added Vim Tilde Button
- 000704: Added Created with VI
- 000509: Added Vim Document Icon
- 000509: Added Japanese MacVim
- 000509: Added Japanese MacVim
- 000503: Added This Site vim Powered
- 000427: Added I love my editor
- 000222: Added Edited with VIM
- 000116: Added Gvim Japanese
- 991020: Added Vim ASCII Logo
- 990915: Added Vim on Fire
- 990903: Added VIM - the 4 star editor
- 990816: Added When things are dim...
- 990810: Updated Vim - The editor
- 990804: Added Vim and SMIL
- 990803: Added Vim - The editor
- 990708: Added Vim Mirror
- 990303: Added ASCII Logos
- 990302: Added Vim with GTK theme
- 990220: Added Vim with NeXT look
- 990203: Added Vim Icon (transparent)

- 990106: Added Vim on MacOS 8.5.1
- 981223: Added Vim in SQL QuickFix mode (1+2)
- 981218: Added gvim-5.4a with GTK
- 981217: Added This site was created with VIM! :wq
- 981111: Added Edited with Vim
- 981105: Added Just VIM it! (This page is Vim POWERED!)
- 981104: Added Just VIM it! (click here)
- 981023: Added GVim-5.3
- 981001: Added Vi Vimproved
- 980904: Added Vim in SQL Mode
- 980726: Added Vim-5 on RiscOS
- 980721: Added this site powered by vim
- 980717: Added Gvim and Unicode
- 980626: Added VIM-5.0w on rxvt-2.4.3
- 980625: Put most pics outside the current page for faster startup.
- 980622: Added designed with vim
- 980331: Added GVim-5.1a
- 980216: Added Vim Hot Icon
- 971217: Added Vim 1-2-3
- 971127: Added this site vim powered
- 971104: Added Gvim on Perl script
- 971006: Added VIM Life
- 970905: Added VIM color on Perl
- 970828: Added Powered by VIM
- 970404: Updated vim with Farsi (Persian)
- 970207: Added vim on Macintosh
- 961015: Added vim on windows

```
URL:          http://www.math.fu-berlin.de/~guckes/vim/pics.html
URL:          http://www.vim.org/pics.html (mirror)
Created:      Sat Jun 29 00:00:00 MET 1996
```

Send screenshots and comments to
Sven Guckes guckes@vim.org

VIM for Macintosh - MacVim

VIM has been ported to the Apple Macintoshs, too:
MacVim

MacVim is available from every ftp mirror. The files are in the subdiretory "**mac**".

However, there are **three** sites with updated binaries:

- MacVim by Dany St-Amant,
- MacVim by Axel Kielhorn
- MacVim by Kenichi Asai (aka "MacVim Japanese").

Hopefully, all three versions will hopefully merge into *the* MacVim again soon. Until then, please chose yourself which version you would like to use.

Problems? If you find a problem with either release then please give a reference to the version you use! Check with command `:version` first and include the info in your requests. Thankyou!

Example version info:

```
:ver
VIM - Vi IMproved 5.7 (2000 Jun 24, compiled Jun 26 2000 19:09:11)
Compiled by guckes@ritz, with (+) or without (-):
```

As I currently have no access to MacOS so I cannot try the latest version. So I am hoping for feedback on the Vim Goals as described below. Have these things been implemented or fixed yet? Thanks for all feedback!

Vim-5 for MacOS - News

000727: *Vim-5.7 binaries*

The Japanese MacVim page now has binaries for Vim-5.7. These are available with and without multibyte support.

000524: *MacVim on VersionTracker.com*

MacVim-5.6.72 is now know by VersionTracker.com:
http://www.versiontracker.com/moreinfo.fcgi?id=2686

000509: *Japanese MacVim*

Kenichi Asai has released a Japanese version of MacVim (see below).

000320: *MacVim-5.6* released.

The files are available from every vim ftp mirror. The master site is:
ftp://ftp.vim.org/pub/vim/mac/

```
 SIZE    FILE                CONTENTS
1688291  vim5.6.full.fat.sit  Vim 5.6, FAT application + runtime files
 298138  vim5.6.68k.sit       Vim 5.6, 68k application only
 624790  vim5.6.fat.sit       Vim 5.6, FAT application only
 333697  vim5.6.ppc.sit       Vim 5.6, PowerPC application only
```

NOTEs: You will need the StuffItExpander to unpack the StuffIt ("sit") files.

For more info see page

http://www3.sympatico.ca/dany.stamant/vim/

MacVim FAQs

MacVim Development Requirements

Q: What do I need to take part in coding the Mac port of Vim?

A:

Requirements

- CodeWarrior9 - better: CodeWarrior Pro2 (see http://www.metrowerks.com/)
- Monospaced Font (see font list below).

MacVim Defaults

Q: What is the default configuration for Vim on Macs?

A: The default configuration for MacVim currently [980302] is: +autocmd + required for syntax highlighting -builtin_terms - no terminals on Macs +cindent + for using vim for coding vim :-) +digraphs + us Krauts need it for umlauts -emacs_tags - anyone writing in Lisp here? +eval + especially useful for use of strftime +ex_extra + :normal, :retab +extra_search + hlsearch and incsearch are quite useful -farsi - not so many speaking Farsi here +file_in_path + command "gf" is too useful to drop it -find_in_path ? anyone using this? -fork() - [Unix only] +GUI + Menus+ +insert_expand + cannot live without completion in insert mode -langmap - anyone using this at all? -lispindent - no emacs user here -mouse_dec - [Unix only] -mouse_netterm - [Unix only] -mouse_xterm - [Unix only] -perl - too much for now -quickfix - too much for now -python - too much for now -rightleft - anyone writing in Hebrew here? +showcmd + cannot live without it +smartindent + too useful to drop it -sniff - noone using it, I suppose +syntax +++ the best feature is a must+ -tag_binary ? anyone? -tag_old_static ? anyone? -tag_any_white ? anyone? -terminfo - [Unix only] +textobjects + cannot live without it +viminfo + Quite useful, too. -xterm_save - [Unix only] +writebackup + too useful to drop it -X11 - [Unix only]

MacVim - Goals

Port to MPW [000519]

Port Vim for use with Apple's MPW compiler.

```
From: Axel Kielhorn
Date: Wed, 10 May 2000 16:26:58 +0200

Apple *does* supply a free compiler, the MPW tools.
The problem is that Vim isn't ported to this compiler,
it requieres CodeWarrior.

If someone wants to compile a MacVim without paying lots of money to
Metrowerks, please download the compiler and port Vim to it. It shouldn't
be that difficult and would make life much easier for everyone. (There is a
reall make available in the MPW tools, no need to rely on strange binary
files that change with every release of CodeWarrior).

And you can get Perl, TCL and Python for the Mac as well.
```

Q: What has been done already with the code?

A:

Completed Items [980221]

- MacOS - memory management: Replacement of malloc/free by NewPtr/DisposePtr
- MacOS - creator resource "VIM!"
- MacOS - clipboard support: clipboard can be used with register name '*'
- Startup: Screen opens on top left position.
- Startup: Automatically sourcing of $VIM:menu.vim
- command ":version": now shows OS name in second row ("MacOS version")
- Startup: $VIM defaults to the folder of the application "vim", but if this path ends in ":src" then the above folder is used.
- Startup - setup file search: The setup filenames starting with a dot take precedence over those starting with an underscore. The setup file for Vi is tried first (exrc), then those for Vim (vimrc) and then for the Gvim: So the order is: .exrc, _exrx, .vimrc, _vimrc, .gvimrc, _gvimrc However, the command ":mkvimrc" should create "_vimrc" or "_gvimrc".

Q: Which features should be implemented next?
A:

Primary Goals

- Menus: **add shortcuts for standard menu commands**
- Menus: **add file dialogs (open, print, save)**
- Create subdir "mac" and "beta-test/mac" on all vim mirror sites.
- Binaries for 68K, PowerPC, FAT - put onto mirrors in StuffIt format
- Make sure that fileformat=mac works
- Compile with CodeWarrior Pro2 (aka CodeWarrior13)
- Drag&Drop (see below)
- GUI: Button on title bar: open the window to full size of current screen.
- Screen size bug: The current screen size is set to 640x400 explicitly - instead of being probed from the graphic device. [Dany, 990215]
- Info box: "Version: Vim-5.0 [980223] for MacOS 68K" and "Comments: Email: guckes-vimmac@math.fu-berlin.de"
- ":version" comamnd: include info on person who compiled the binary, as well as info on compiler and OS:

```
:version
VIM-5.0 [980219]   OS: MacOS-7.5.1
Compiled by Sven Guckes guckes@vim.org
         on Tue Feb 24 19:32:29 CET 1998
with     CodeWarrior Pro2
List of compile  options ['+' means "code included for"]:
[etc]
```

"I'd like to participate in the development." Q: How should I name my version of
A: Please make your version available like this:

```
Format:         vim-5-yymmdd-68k-appl.sit.bin
                yy = year  mm = month  dd = day
Example:        vim-5-980406-68k-appl.sit.bin
                only the application for 68k Macs,
                stuffed and in binary format.
Example:        vim-5-980623-PPC-src.sit.hqx
                only for PPC, but the complete source,
                stuffed and binhexed
```

```
        Example:        vim-5-980623-FAT-src.sit.hqx
                        the source with binaries for both 68k and PPC Macs,
                        stuffed and binhexed
```

Q: What further goals are there for the mac port?
A:

Further goals

- About Box
- Save as...: support for "styles"
- Convert between "styles" and "syntax coloring"
- Documentation: Add list of URLs for monospaced fonts
- File:Print...
- Maclike: Allow command-. and control-c to abort processing.
- Maclike: Resource file to keep "viminfo" in. [980618]
- Make Vim a plugin editor for CodeWarrior. [990210]
- Default folder for tempfiles: "Volume:Temporary Items:"
- Make vim an IDE for CodeWarrior. Dany sent a suggestion for an IDE menu to the list [990210].

Ideas

- Bundle Vim with set of monospaced fonts
- Bundle ultrashort beep sound!
- Make Vim a PowerPlant application?
- Use WASTE lib?
- Preferences File (see below)

Problems

- Setup files must be in Unix or Dos format. Mac format should be made possible, too. [990210]
- Setup files must be in the same folder as the application. Should be possible to put them into the Preferences folder, too. [990210]
- syntax coloring takes *very* long, vene for small files - why?
- Startup: Window is not displayed
- Display: Font does not get drawn properly
- Fonts: Cannot use font names which include a space
- Format: The tags file mus be in Mac format (fileformat=mac)
- filename completion: doesn't work for unix-like path
- filename completion: doesn't work for volume name
- filename completion: doesn't work for a volume different of the current one. [workaround: Use ":cd volume" before using filename completion on that volume]

Drag and Drop DONE (+) and TODO (-)

- + open file by dropping onto application icon
- - open file by dropping onto open window
- + select text within open window
- - drag selected text onto desktop creates textfile

Menus - Default Structure:

```
    File:
    OpenFile... Cmd-o   Open (:edit) file
    Save        Cmd-s   Save current buffer to current file
    SaveAs...   Cmd-S   Save current buffer to file
```

```
Close       Cmd-w   Closes current subwindow
Print...    Cmd-p   Prints current buffer
Quit        Cmd-q   Quits Vim


Edit:
Undo        Cmd-z   Undo last change (deletion/insertion)
Redo                Redo last change
Cut         Cmd-x   Delete current selected (visual) text
Copy        Cmd-c   Copy   current selected (visual) text to clipboard
                    and to the unnamed buffer, too, of course
Paste       Cmd-v   Like 'p' to "put"
Clear
Search Backward
        for current word
        for current filename
Search Forward
        for current word
        for current filename
```

Preferences:

- Show splash screen (if ever)
- Show About box on startup
- Position of startup window
- Font

Vim for MacOS - Known Bugs / Specific Stuff

MacVim vs DiskLock [000519]

DiskLock is a security application which prevents (amongst other things) creation of files by applications.

It seems that DiskLock prevents MacVim from creating swapfiles.

```
From: Ben Fowler
Date: Fri, 12 May 2000 13:21:09 +0100

The Power On software web site acknowledges
a bug in an earlier version that meant that
files beginning with '.' could not be opened.
```

As with DOS, some characters are different on the Mac: char name hex («) forward quote ab (`) backward quote 60 (') quote 27 This can be quite annoying. Unfortunately, there seems to be no way to "switch the character set" easily within Vim. You will simply have to install some fonts on your Mac for "PC character set" and "Unix character set". Does anyone know of some good ones?

Vim for Mac - Links

MacVim by Axel Kielhorn

http://www.tu-bs.de/~i0080108/macvim.html

Author: Axel Kielhorn A.Kielhorn@tu-bs.de

http://www.kassube.de/mirrors/macvim/ (mirror) [000808]

Contact: Nils Kassube nika@kassube.de

MacVim by Dany St-Amant

http://www3.sympatico.ca/dany.stamant/vim/

Author: Dany St-Amant (email withheld)

MacVim by Kenichi Asai [000509]

A Japanese version of MacVim.

http://www-imai.is.s.u-tokyo.ac.jp/~asai/macvim-e.html

Author: Kenichi Asai asai@is.s.u-tokyo.ac.jp

Nostalgia: vim-3 for Mac

The filename is **vim-30.hqx.gz** [Size: 271,212 bytes].

ftp://ftp.fu-berlin.de/mac/mirrors/info-mac/text/vim-30.hqx

Porter: Eric Fischer enf@pobox.com (the address eric@rainbow.uchicago.edu is obsolete) enf1@ellis.uchicago.edu

Eric offers his port with a link from his Software page:

http://www.pobox.com/~enf/sw.html

http://ars-www.uchicago.edu/~eric/sw.html [obsolete]

Mac Utilities

Fonts [000403]

Suggested Fonts for use with MacVim:

```
Courier:h10   (standard font)
Monaco:h9     (standard font)
MPW:h9        (bundled with MPW and CodeWarrior)
Mishawaka:h9  (bundled with Eudora)
ProFont-2.2 - available on every InfoMac mirror
in subdir "font" as "pro-font-NN.hqx", eg:
ftp://mirror.aol.com/pub/info-mac/font/pro-font-22.hqx
Size of package: 460K (469182 bytes).
```

MacsBug [000519]

http://developer.apple.com/tools/debuggers/MacsBug/

Get this tool to get some info if applications fail on your Mac.

MacTar [000808]

http://www.strout.net/macsoft/mactar/

"This is a MacOS port of "tar" (tape archive), a common file-packing format used on Unix machines. This program reads and writes tar format archives."

StuffIt [000808]

[todo]

```
URL:          http://www.math.fu-berlin.de/~guckes/vim/macs.html
URL:          http://www.vim.org/macs.html (mirror)
Created:      Sun Jun 16 00:00:00 MET 1996
```

Send feedback on this page to
Sven Guckes guckes@vim.org

if_ole.txt   For Vim version 5.7. Last change: 2000 Mar 21

VIM REFERENCE MANUAL   by Paul Moore

The OLE Interface to Vim   *ole-interface*

{Vi does not have any of these commands}

OLE is only available when compiled with the |+ole| feature. See
src/if_ole.INSTALL.

==
1. Activation   *ole-activation*

Vim acts as an OLE automation server, accessible from any automation client,
for example, Visual Basic, Python, or Perl. The Vim application "name" (its
"ProgID", in OLE terminology) is "Vim.Application".

Hence, in order to start a Vim instance (or connect to an already running
instance), code similar to the following should be used:

[Visual Basic]
>   Dim Vim As Object
>   Set Vim = CreateObject("Vim.Application")

[Python]
>   from win32com.client.dynamic import Dispatch
>   vim = Dispatch('Vim.Application')

[Perl]
>   use Win32::OLE;
> $vim = new Win32::OLE 'Vim.Application';

Vim does not support acting as a "hidden" OLE server, like some other OLE
Automation servers. When a client starts up an instance of Vim, that instance
is immediately visible. Simply closing the OLE connection to the Vim instance
is not enough to shut down the Vim instance - it is necessary to explicitly
execute a quit command (for example, :qa!, :wqa).

==
2. Methods   *ole-methods*

Vim exposes three methods for use by clients.

                                                        ole-sendkeys
SendKeys(keys)   Execute a series of keys.

This method takes a single parameter, which is a string of keystrokes. These
keystrokes are executed exactly as if they had been types in at the keyboard.
Special keys can be given using their <..> names, as for the right hand side
of a mapping. Note: Execution of the Ex "normal" command is not supported -
see below |ole-normal|.

Examples (Visual Basic syntax)
```
>       Vim.SendKeys "ihello<Esc>"
>       Vim.SendKeys "ma1GV4jy`a"
```

These examples assume that Vim starts in Normal mode. To force Normal mode, start the key sequence with CTRL-\ CTRL-N as in

```
>       Vim.SendKeys "<C-\><C-N>ihello<Esc>"
```

CTRL-\ CTRL-N returns Vim to Normal mode, when in Insert or Command-line mode. Note that this doesn't work halfway a Vim command

ole-eval

Eval(expr) Evaluate an expression.

This method takes a single parameter, which is an expression in Vim's normal format (see |expression|). It returns a string, which is the result of evaluating the expression.

Examples (Visual Basic syntax)
```
>       Line20 = Vim.Eval("getline(20)")
>       Twelve = Vim.Eval("6 + 6")              ' Note this is a STRING
>       Font = Vim.Eval("&guifont")
```

ole-setforeground

SetForeground() Make the Vim window come to the foreground

This method takes no arguments. No value is returned.

Example (Visual Basic syntax)
```
>       Vim.SetForeground
```

==========

3. The "normal" command *ole-normal*

Due to the way Vim processes OLE Automation commands, combined with the method of implementation of the ex command :normal, it is not possible to execute the :normal command via OLE automation. Any attempt to do so will fail, probably harmlessly, although possibly in unpredictable ways.

There is currently no practical way to trap this situation, and users must simply be aware of the limitation.

==========

4. Registration *ole-registration*

Before Vim will act as an OLE server, it must be registered in the system registry. In order to do this, Vim should be run with a single parameter of "-register".

```
>       gvim -register
```

Once vim is registered, the application path is stored in the registry. Before moving, deleting, or upgrading Vim, the registry entries should be removed using the "-unregister" switch.

```
>       gvim -unregister
```

The OLE mechanism will use the first registered Vim it finds. If a Vim is already running, this one will be used. If you want to have (several) Vim

sessions open that should not react to OLE commands, use the non-OLE version, and put it in a different directory. The OLE version should then be put in a directory that is not in your normal path, so that typing "gvim" will start the non-OLE version.

==
5. MS Visual Studio integration *MSVisualStudio* *VisVim*

The OLE version can be used to run Vim as the editor in Microsoft Visual Studio. This is called "VisVim". It is included in the archive that contains the OLE version. The documentation can be found in the VisVim directory, the README.txt file.

==

Why use a Vi clone? Why VIM?

"So - why should I use Vim and not standard Vi?"

First off, there is no "standard vi", ie no existing "vi" is exactly what the POSIX standard defines. Most implementations are "just about vi" with some or many differences. So I will simply denote such versions with "Vi" (capital 'v', small 'i').

Most of the popular "vi clones" do more than just fix bugs of Vi - they add new features which make work much easier. Almost all vi clones are better than Vi and I suggest that you switch to one of them.

The most popular/widespread vi clones are: elvis, nvi, vile, and vim. But check out calvin, elwin, lemmy, and viper, too.

But why switch to Vim? Well, Vim is a "vi compliant" implementation with many features you will find in other editors. Many features have also been improved, hence the name: "Vim - Vi IMproved". Vim runs on almost all kinds of operating systems, too.

The following is a summary of the most compelling features (with some examples), but the list is by no means complete. Vim-6 comes with 2.5MB of documentation (plain ASCII text) which explains all commands and options in detail, takes a look at the use on special systems, and gives a lot of examples for text editing in general.

Please note that this text is available as a webpage, so expect some changes to it. I will mark all changed text with a date in format "[yymmdd]" such as "[990101]" for "1999 January 1st". You are welcome to add and especially to correct me on this text!

Sven Guckes guckes@vim.org [a happy Vim user since 1994]

Vim - Technical Details

Availability

> Vim does not run only on Unix and its many variants (most notably BeOS, FreeBSD and Linux, HPUX, IRIX, and Solaris), but also on many other operating systems, such as DOS and Windows (ie Windows-3.1, Windows95, Windows98, Windows2000, WindowsNT), AmigaOS, AtariMiNT, MacOS, OS/2, RiscOS, VMS.
>
> vile: not available for AmigaOS, AtariMiNT, BeOS, MacOS.

Compatibility

> Vim tries to be compatible whereever possible. It even simulates "known bugs". Fortunately, it allows the user to override this with ":set nocompatible". So you do not lose anything (well, except for a few bugs :-).

Language Support - Right-to-Left

> Some languages have their text flow from right-to-left (eg Arabic and Hebrew). Vim supports this display mode, too. You can even mix left-to-right and right-to-left text. (Forget about Vi here!)
>
> Support for Hangul (Korean), Chinese and Japanese will follow.
>
> vile: no support for right-to-left display

Vim - support for terminals

adjust to resize of terminal [990824]

When you change the size of the terminal then Vi does not notice. With Vi you will have to exit, change the info on the terminal by adjusting COLUMNS and LINES, and restart.

If the terminal gives a signal then Vim notices a change and will adapt itself to the new size.

support for termcap and terminfo [990824]

On Unix systems info on terminals is either given by "termcap" or "terminfo". Vim knows about both kinds. Vim even knows about special key codes such as arrow keys - and even in different styles.

"builtin terminals" [990824]

Vim can be compiled with info on some standard terminal kinds. So you can use make use of terminals even if the terminal info is missing or is incorrect.

black&white or color? 40 or 80 lines? [990824]

On DOS Vim allows to chose the "screen mode".

Vim - Editing Features

Apart from the technical things, users mostly want features with "editing". Here are some of them:

Increased Limits [980101]

Many features are associated with editing. However, before you start making changes the editor must allow them. Vi does not allow you to edit every file - actually some (old) versions will accept only lines up to 255 characters. But a limit, however big, is a limit. Even 1024 characters usually won't do when you want to edit a "flow-text" where every paragraph is just one line. Some files are just too big for Vi and some implementations simply choke on the character ^D.

Vim allows you to edit *all* files, and yet recognizes when you do not have the permission to edit a file, showing "[RO]" for "read-only" files.

Multiple Undo

Probably the best new feature of Vim is the "multiple undo", which means that you can undo not only the last change (as with Vi) but the last **N** changes. (The number N is configurable and is only limited by available memory.)

I may add that the "multiple undo" alone is so helpful that I wouldn't want to go back to Vi. This feature certainly is nice for every beginner. should have it - even if only to make it easier for beginners as it allows to "back up" from commands that you "mistyped".

Redo

Gone back to far with "undo"? Well, "redo" the changes (with command ^R).

I often "undo" my changes to email when I want to recall some deleted text; I copy the text to some buffer and then back up with "redo" to continue writing. [980106]

Termcap Support - Terminal Adaptation and Arrow Key Support [980101]

Many Vi users get frustrated with having arrow keys on their keyboards and not being able to use them to steer the cursor with them. Although arrow keys are fine with Vi if the setup of the terminal and Vi match, you always get into trouble when you switch the current terminal as Vi does not adapt to other terminals - you usually have to quit, update, and restart.

Vim automatically a change of the current terminal and adapts to it (by reading the termcap entry). This also allows to recognize the codes of arrow keys. Vim also allows to move the cursor not only in command mode, but in all other modes, ie within insert mode, on the command line, when selecting text visually and even in replace mode.

Extended Text Objects [980106]

words, sentences, and paragraphs

Vi has commands for jumping to the begin or end of the current paragraph or sentence which can be combined with "change commands" to copy and delete these - but you cannot select any of these "text objects" from within. ie you need to jump to one end and then use a copy/delete with a jump to the other end.

Vim allows operations (like copy and delete) on the "current text object" from within, ie you do not need a jump to one or the other end at all. Vim also extends the notion of "current text object" by "current WORD", ie non-whitespace text, and also for "current word", ie normal words made of "letters" which are even configurable. (Again, this is not possible with Vi.) Furthermore, Vim allows to select text objects with or without surrounding whitespace (called "inner" and "all").

Examples:

```
command description
viw     visualize inner word
viW     visualize inner WORD   (also works for "up-to-date")
yas     yank (copy) all   sentence  (with    surrounding whitespace)
dip     delete      inner paragraph (without surrounding whitespace)
```

Number Prefix of commands [980114]

Many commands can take a "number prefix" to perform the commands many times. However, this does not work some commands. For example, you can copy four lines with "4yy" - but you cannot put the copied text twice with "2p".

Vim fixes this - number prefixes can be applied to almost all commands.

Search Offsets [980114]

Whenever you search for patterns with Vi it will place the cursor on the **first** character of the matching text - but you must then look for the end of a match yourself.

Vim allows to specify a "seach offset", ie you can place the cursor relatively by "begin" or "end" of the matches text:

```
pattern         cursor position ~
/test/+1        one line below "test", in column 1
/test/e         on the last t of "test"
/test/s+2       on the 's' of "test"
/test/b-3       three characters before "test"
```

Visual Selection [980106,990104]

Almost all commands in Vi operate/work on some text - but you cannot "see" this text prior to some command.

Vim lets you select text and shows it in "reverse" (aka "highlighting"). This allows you to visually check that you have selected the right text for the next command (usually one of "delete" or "yank", but also "filter"). You can "visualize" text from any position to another, ie starting anywhere within a line and then move to any other position; you can also select "linewise" by whole lines to easily highlight a "range" of lines. And you can select "blockwise", ie a rectangle on the screen.

With Vim you can directly operate on the current "visual text" with an ex command or filter (by typing ':' or '!' respectively).

Example: The following command selects the current inner paragraph (with "vip"), switches to ex mode with ':' (note that Vim automatically inserts the line range "'<,'>" to denote the line range of the visual text), and adds a "substitution command" to insert a '#' at the begin of each line, thus commenting them out with eg shell scripts: map ## vip:s/^/#/ This command does not get executed automatically as it is missing the 'return' at the end - on purpose! This still allows you to cancel the command by pressing 'escape'. Note that the "command line history" (see below) remembers the command even if you cancel it; this allows recalling and editing of it later.

Text Formatting [980107]

Many people are given Vi as the standard editor, but they just want to "write a simple email or post" with it. To make the text "look good" they usually want the editor to "reformat" the text to some textwidth. But such a command is not given within Vi; you are usually referred to "external commands" such as "fmt" and "par" (quite

good, actually). But who wants to learn even more programs and install them everywhere?

Vim has built-in text formatting. So there is no need for external programs. Vim reformats text to a given "textwidth", unlike Vi which does this only for a given margin to the end of the terminal. And even if you just use the right-margin, Vim recognizes a change of currently used terminal and will update the textwidth accordingly (which Vi cannot do). Vim even preserves "cited/quoted text" by preserving the "quote prefix". And when you break a quoted line, Vim will add the quote prefix to the next line with the broken off text. Vim even adds two spaces after an end-of-sentence if you want that.

Vim allows to specify "comment characters" which allows to recognize whether the start of lines make it a "commented line". When you "break a line" (insert a newline) Vim then is able to add the "comment prefix" to the new line automatically. Vim's text formatting also makes use of this by recognizing lines with the same comment prefix and will reformat text for adjacent lines of the same comment prefix, preserving the comment prefix, of course. This allows to regard "quoted text" as "commented text" and reformat a "cascade" of quoted text (such as in Usenet posts) wil very few keystrokes (usually "gqip" suffices).

Completion for words, commands, and filenames [980101]

With Vi you must type each word - however long it might be.

Vim has commands to complete partial words, commands, and filenames. Word completion is usually used within the edit buffer to complete a word prefix of some long word or those of hard-to-type names in source code. Vim also looks at "dictonaries" (files with other words) for possible completions and lets you cycle forward/backwards through them.

Vim is smart enough to check files which are referenced with the current buffer. On the command line, the first word is assumed to be a command and can be expanded to its full name. Vim also expands filenames if the current command name suggests that the current word could be a filename. Vim also inserts the value of an option/variable. Example: Type the command ":set option=" and then type the "wildchar" ('^E' by default, but usually set to '^I' (TAB).

To TAB or not to TAB / TAB expansion [990301]

Program code usually uses TABs to indent statements of the same "level". But what is the best way to expand those TABs on the screen? How far should they be expanded on screen? (Multiple of 2,4, or 8 spaces?) Or should the cursor jump to the next "tab stop"? Maybe the should be expanded as spaces right away? Or should you use the least amount of TABs and spaces to reach the current indentation within the code? Well, chose the way you like: Apart from Vi's options "shiftwidth" and "tabstop", Vim offers the options "expandtab", "smarttab", and "softtabstop" to chose whatever you like best. And should you have the need to expand the TABs within an existing text then you chose your options and the command ":retab" to do that.

Digraph Input

Several languages include characters which are not included in ASCII and which thus are available by characters codes within 128-255. These codes are produced by special keyboards - but when you do not have such a keyboard available then you must either use a workaround or it is simply not possible at all. Therefore Vim supports the input of such characters with a command that takes the next two characters as a combination - hence the word "digraph" ("two letters"). Example: Enter the keys `^K + $ + $` to get a pound sign: '£'.

The German language has three umlauts which can be entered easily with the combination of a vowel and inverted commas (""). Example: Enter the keys `^K + a + "` to get an 'ä'.

And those who speak Nordic languages will use the letters "Åæø" and the Spanish will want 'ñ' ('n'+'~').

By the way, the current version of Vim has some rudimentary support for "Unicode" which allows to type text in eg Chinese and Japanese.

Filetype Recognition [..,990301]

If you are using a combination of Unix, DOS/Windows and Macintosh (chose any two) then you certainly know that the end-of-line is given by different character combinations. This forces you to constantly change the "filetype" accordingly.

Editing files of all filetypes is no problem for Vim - it recognizes the filetype automatically and lets you start editing right away. Files can be kept with their current filetype or written out any other filetype.

Example: You "edit file" and Vim shows "[dos]" to tell you that the file is a DOS file. You then ":set fileformat=unix" and ":write" the file again. (These commands can be abbreviated to ":se ff=unix" and ":w").

Result: The file's line-ends is converted to Unix style.

Command Line History and Command Line Editing

Have you ever mistyped a long command in Vi an just hated to retype it? Isn't it sad that shells have a "command history" and Vi does not? Well, Vim has both - command line history and editing. Just use the arrow keys on the command line to recall the last command, move the cursor back and forth, and delete and insert characters as you like.

Search Pattern History

The search pattern history is just like the command line history - but it is a special history for search patterns only. The "pattern history" has been separated from the "command line history" to allow easier access.

No, Vi does not have this.

Configurable Words [980101]

Vi checks for an abbreviation after you "complete" a word, ie by typing a non-word character. Whatever the difference between a "word character" and a "non-word character" - Vi doesn't tell you about it, not lets you change the definition.

Vim's online help tells you about the definition of "words" and also tells you that you can change the definition to your taste, or rather, to whatever kind of language you currently want to use. This helps you both with writing txts in other languages and writing code in some other programming language.

The timout solution and associated problems: Another natural way of editing is to abbreviate editing tasks by defining a short sequence of keys as an abbreviation of a sequence of commands. These are known with Vi as "mappings". These abbreviations however can also occur as a part of some text - therefore Vi needs *some* way to distinguish it from text. The solution to this problem is to *wait* a little time (know as the "timeout") before the check this made whether the recent input is a command or whether the user continues typing text. Although this is a good method, the timeout of Vi is hardcoded and thus cannot be changed. But when you have a slow connection then the characters of your input might arrive with a delay exceeding the timeout, thus making it impossible for it to recognize an abbreviated command sequence. With Vi your simply lost here - but Vim allows you to adjust the timeout. This is also nice if you are a fast typer ("touch typist") as you can also lower the timout delay.

Modes vs mapping: Vim distinguishes between more modes than Vi - allowing you to define abbreviations and map keys for each of those modes separately. This difference is a benefit as you can define an abbreviation only for the command line, such as an often used filename. Example:

```
cab TODO ~/projects/todolist.txt
cmap ##  e TODO
```

Now switch to command mode with ':', type "##" and you will see the command "e TODO"; now type a space to eind the word "TODO" and have it expanded to "~/projects/todolist.txt"; then type a return to have Vim accept this edit command.

Multiple Buffers

Vi can only deal with two buffers - the current buffer and "the other buffer". This poses many problems with editing more than just one file: You cannot quickly switch to another file for editing, and switch back for editing the file before. Also, Vi's safety feature that forces you to save changes before switching, gets in the way here: You must "save" ("write") the changes of the edit buffer back to file before you switch - otherwise the changes are lost. Also, switching to another file makes Vi "forget" the set "marks" and the contents of the unnamed buffers. (Some versions seem to lose the contents of the numbered buffers, too.)

Vim allows to edit many files at the same time without having to write back changes before switching. Vim keeps a "buffer list" of all edited files and lets you switch forward and backward or directly to any buffer number. You can also switch by part of the path/filename and even cycle through possible matches before switching. And Vim does not lose the contents of numbered buffers when you switch and also keeps the set marks. The maximum number of buffers is only limited by available memory.

Screen Splitting into Windows [980106,990101]

Vim has many commands to "split the screen" into (horizontal) windows. The height of each window can be decreases and increased, and you can jump around between them as well as cycle them. This makes it very easy

to ":split" the current window in two separate windows, each showing a different part of the same file or different files. Vim even has a command to change to the filename under the cursor ("gf").

You cannot split windows vertically - yet. This feature was much asked for with Vim6, though, so it is likely that you will see this feature soon.

Need I say that Vi does not have such commands?

Syntax Coloring

This feature probably is the best one of all: Coloring Text. The colors can not only be given to special words or characters, but they are defined by "rules" that can define a "language" - hence the name "syntax coloring".

The syntax coloring is general enough to allow rules for any kind of language, usually programming languages. The best effect is that you can easily see whether some "beginning" matches some "ending" to avoid errors in structure. It is equally nice to see variable names and constants in their own color.

Mind you, this does not make Vim a "syntax checker" as this can require to (always) check the whole text (source code). But the rules can be made as complex as need be.

vile: Does syntax coloring with external filters and buffer hooks.

Autocommands

"Autocommands" allow to execute commands automatically with editing files. These autocommands can be executed when you start or end editing files in combination with filename patterns. A typical use of this is to "change the textwidth to 70 when editing a C source file":

```
au BufRead *.c tw=70
```

This can get more complex, of course, such as automatically uncompressing compressed files just before loading them into an edit buffer. Someone used this to remap keys such that you can use them to select files within an edit buffer as if you were using a file browser.

By the way, Vim uses autocommands to load the appropriate color setup for the current file, eg loading the rules from tex.vim for TeX and LaTex files.

Macro (Map) Recording

Vi users who write "mappings" to abbreviate a series of commands know how tedious it can be to remember the keystrokes for this. Vim makes it easy with "macro recording": The letter 'q' (which is unused in Vi, btw) is used to start the recording; the next letter is the name of a register which is to store the keystrokes. Whatever you type after that is stored within the register until you type another 'q'. That's it. You can then play back this sequence with the command '@' followed by the register letter/name.

Example: qa Start recording into register 'a' G jump to last line of current buffer ?^-- $^M search backward for a line with sigdashes d/^$^M delete unto next empty line q end recording This simply deletes the signature from the current file. You can now edit another file with a signature and have it removed by simply replaying the macro with "@a".

Viminfo - Saving your editing for the next session [990301]

Vim saves all those extra stuff that you typed during an edit session into an extra file - the viminfo file. This contains the list of filenames you had edited together with the "marks" that you have set for each; also saved are the registers (read: scrap buffers to hold some text) with their contents, so you can "paste" from them at your next session. [This has saved the lives of many editors, I hear.] Furthermore, Vim can save the histories (command line, search patterns), as well as global variables defined during the editing. Further options allow to chose a filename for the viminfo file, the maximum number of filenames to remember, and a maximum number to the lines saved within each register. But that's for experts. ;-)

Online Help - ":help"

Vim gives you the command ":help" which will open a new window with the "helpfile" ("help.txt"). From there you can jump to other topics by placing the cursor on a "tag" (eg "|quickref|") and jump to the associated helptext with the command ^] (control-]). Using ":help option" you can directly see a definition of "option". You can even use this system for your own help text and source code. Vim ships with the utility "ctags" which "creates tagfiles" to speed up finding the relevant text in other files.

Vim has a syntax file for its own helpfiles, too, so you can easily see tags in their own color.

GUI - Of Menus and Mice

Vim-5 now sports a graphical user interface (GUI). Vim with code for GUI can be started as "vim -g" or as "gvim" and then gives you features that many GUI programs have. This has several advantages: You can click on icons to issue commands (or command sequences) which makes it easier for beginners as they do not need to know them. (Just click on the icon with the floppy - ":write".) You can also use the mouse for copy&paste, ie select some text, copy it, click onto another position and have the text inserted there. Vim also has a "scrollbar" that shows how the window's content relates to the whole of the "file" (edit buffer). You can define "menus" to present commands that you need most often with your editing, but cannot be bothered to remember. Some GUIs also allow "tear-off menus", ie you can tear them off the menu bar and stick them onto your desktop to keep them available; no need to open them menu then - just click on an item. The GUI allows font selection, ie text can be displayed in different fonts, allowing use of bold, italics, and outline within the font. (But mostly you want to use a monospaced font for all the text, anyway.) Some GUIs allow to define colors for buttons and the cursor. Gvim also has a buffer selection dialog, so you can choose a buffer with a mouseclick. And some people just do not have enough experience with terminals to get some things working on them, eg arrow and function keys, text attributes, and color. Last but not least, you can use more than the eight or sixteen colors usually available on terminals; you can have 16, 256, or even 65 thousands of colors. "Look at 'em colors!"

Vim - Extra Support

Support for GUI

Vim can be linked several GUI sets: Athena, Motif, Lesstif.

Support for OLE (and thus for Microsoft Developer Studio) [990108]

Vim can be integrated with Microsoft Developer Studio using VisVim. An extra archive is available for this. See also: Page on VisVim

Support for Perl

Support for Python

Support for UTF-8

DONE/TODO

DONE

- Availability
- Compatibility
- Language Support - Right-to-Left
- Limits
- Multiple Undo
- Redo
- Termcap Support
- More Text Objects
- Number Prefix
- Search Offsets
- Visual Selection
- Text Formatting
- Completion
- TAB expansion
- Digraph Input

- Filetype Recognition
- History for commands and searches
- Configurable "words"
- Multiple Buffers
- Screen Splitting
- Syntax Coloring
- Autocommands
- Macro Recording
- Online Help
- GUI Support

  TODO
- Built-in Scripting Language
- Tag Stack Commands
- Escape for one "normal command" with i_CTRL-O

```
reasons to use a vi clone: development and support
(mailing lists, newsgroups, webpages)
documentation:
        elvis           Steve Kirkendall
        nvi             Keith Bostic
        vim             Bram Moolenaar
        vi archive      R. Olsen
```

```
URL:         http://www.math.fu-berlin.de/~guckes/vim/why.html
URL:         http://www.vim.org/why.html (mirror)
Created:     Thu Jan 01 00:00:00 CET 1998
```

Send feedback on this page to
Sven Guckes guckes@vim.org

VIM - Mailing Lists

010108: Currently there are eight maillist on Vim.

991104: Mailing list about Macintosh development has been moved to vim-mac@vim.org - so you can (un)subscribe by email now.

990825: Many mailers break the threading in maillists. Please use a mailer which gives a reference to the previous mail! I suggest to use highly customizable mailers which can be used on *every* terminal: elm, pine, and most of all, mutt (see also the section on mutt below).

VIM Maillists - Overview

VIM Maillists - Maillist on vim.org

There are five mailing lists about VIM on vim.org:

```
 vim-announce@vim.org         Vim Announcements
          vim@vim.org         Vim Help List
      vim-dev@vim.org         Vim Development List
      vim-mac@vim.org         Vim Development on the Macintosh
vim-multibyte@vim.org         Vim Development of Multibyte Support

vim-fr@yahoogroups.com        Vim for French   speaking users
vim-jp@sandalwood.net         Vim for Japanese speaking users
```

NOTE: Before sending mail to the lists, you must **subscribe** ! Mails from unsubscribed *addresses* are rejected. Yes, that's a feature. Hope you understand.

To get help in an email - send a mail to vim-help@vim.org.

VIM Maillists - Maillist not on vim.org

There is one more lists not on vim.org:

```
vim-fr@yahoogroups.com      Vim for French   speaking users
vim-jp@sandalwood.net   Vim for Japanese speaking users
```

For subscription to **vim-fr** take a look at the info page to vim-fr:
http://www.yahoogroups.com/list/vim-fr/info.html

VIM Maillists - List Descriptions

vim-announce@vim.org

Purpose: Announcements only. New releases, important patches and bug reports. Change of important mail and web addresses.

Status In:

Subscription required to post: yes - but posting is allowed only for *some* people. If you want to post then please ask the maintainer for permission.

Status Out:

All mails are sent to its subscribers *and* all subscribers of the other maillists ("vim", "vim-dev", and "vim-multibyte").

vim@vim.org

Purpose: The "Vim help List".

Status In: Subscription required to post: yes. Everyone is welcome to join and ask questions about Vim and its usage. You are especially welcome to answer questions, too. :-) But to avoid frustration for readers you are advised to read both the VI FAQ and the VIM FAQ before you post. Thank you!

Status Out: All mails are sent to subscribers.

vim-dev@vim.org

Purpose: Discussion of Vim Development.

Discussion of new features and possible bugs. Development of macros, patches and setup files.

Status In: Subscription required to post: yes. You should be able to use a compiler and try to avoid asking FAQs. You know about termcap and terminfo. You can write an OS kernel with "cat > kernel". You do not use Emacs. ;-)

Status Out: All mails are sent to subscribers.

vim-mac@vim.org

Purpose: Discussion of Vim Development on the Macintosh.

Status In: Subscription required to post: yes.

Status Out: All mails are sent to subscribers.

vim-multibyte@vim.org

Purpose: Discussion of Vim Development for Multibyte Support.

Status In: Subscription required to post: yes.

Status Out: All mails are sent to subscribers.

Vim Maillists - Subscribing and Unsubscribing

Before you are allowed to post to any of the mailing lists you must first "subscribe" to them. Subscribing and unsubscribing to any of the three mailing lists is done by sending a mail (empty one suffices) to an address which is similar to the listname:

```
        listname-subscribe@vim.org
eg      vim-subscribe@vim.org
```

If you are using UNIX then these commands should be easiest to (un)subscribe:

Subscribing to "vim"

```
echo vim rules | mail vim-subscribe@vim.org
```

Unsubscribing to "vim"

```
echo seeya | mail vim-unsubscribe@vim.org
```

Commands to subscribe and unsubscribe to the other lists are left to the reader as an exercise. :-)

VIM Maillists - Maintenance

The maintainer of the Vim mailing lists is

```
Özgür "Öç/Oetsch" Kesim oec@vim.org
```

Please note that Oetsch is working fulltime and his maintenance of the lists is a just a hobby. Responses to emails about the mailing lists might take a while. Please be patient! Thankyou.

Please do not contact the maintainer to subscribe ot unsubscribe you. There are so many subscriptions that it cannot be processed by one person. That's exactly the reason why an maillist software is being used. Please use it! Thankyou.

However, as posting to vim-announce requires special permission from the maintainer. So there's a reason for you to contact him.

The mailing lists are managed with qmail and ezmlm; the following page has some info on this:

http://www.math.fu-berlin.de/~kesim/ezmlm/

Vim Mailing Lists - Talk Convention

Edit your mails!
Important: Please take a little time to "edit your mails". Remove superfluous data, and get the context right. Please read my Guide on Editing Emails to understand why and for some tips on editing. Thanks!

Then, please notice that the following text occurs quite often:

Attach patches - but only if they are small! [000119]
You can always send your patches to the author. If the patch applies to requests or bug reports on one of the mailing lists then you can add it to your (public) response on the mailing list.
Whenever possible, *attach* the patch - so it can easily be removed on replies or even just to save some space on the disk.
If the patch is big then please do not distribute it on the lists - but send them to the author directly.

Give references!

```
:help tag
:set options=value
RTFFAQ
RTFM
```

And this is what it means:

:help tagname

> This means "enter this help command and you shall find an explanatory text which contains the answer to your question". This is often used to answer the request to some feature that already exists. In these cases we just answer by pointing to the section in the help file which describes this feature. So just enter ":help tagname" and read the stuff there. If you do not like the feature the way it is then we ask you to give a definition and a few examples of the way you like it to be. This helps a lot when redesigning a feature!

":set option=value"

> This means "enter the command to set the option "option" to the value "value". This change should then change the behaviour of a command and thus enable you to do with it what you desired.

RTFFAQ

RTFFAQ means "read the fine [list of] frequently asked questions [and answers] in the "VIM FAQ"". Read it and you shall find the answer!

URL of VIM FAQ:

http://www.vim.org/faq/

RTFM

RTFM means "read the friendly manual". Usually used when someone repeatedly asked for info which is in the manual and some other help text (such as the Vi FAQ and Vim FAQ). Please take a look at them again and you will probably find the answer. And if you should use it then please add a pointer. Thanks!

Thank you for understanding! :-)

VIM Mailing List FAQs

Some questions get asked much too often on the mailing lists. Please try to avoid them by reading the answers here. Thanks!

VIM Distribution

Q: Where can I get VIM?

A: You get VIM from several sites. These are all listed in the VIM Distribution page:

URL: http://www.vim.org/dist.html

Rejected Posts

Q: Why are my mails rejected?

A: You may be sending as "another one", ie with your name but with a different address. This happens when you have subscribed as `user@hostA.domain` - but you may be sending mails as `user@hostB.domain`, ie sending from another host. The mailing list software treats this as different addresses and thus as different people. In that case you need to subscribe from `hostB`, too, before mails from that address are accepted. The best way is to send the subscription command and all further mails *without* the host name. Some mailers can be configured to drop the host names which is very convenient and thus recommended. Ask your postmaster about this!

Changing accounts [970818,990303]

Subscribe Check - Automatic Unsubscription

Q: Am I still on one of the vim mailing list?

A: First off, do **not** send requests to the mailing list manager! Although the easiest way to find out about your subscription is to send a mail to the mailing lists and see if these bounce back to you - **do not send test messges to the list** - this is annoying.

Q: But my account will be terminated soon - I really need to know. How can I find out NOW?

A: Relax. Mails sent out to you will bounce back to the mailing list software, and if and a bounced mail from you will unsubscribe you from the mailing list automagically. But you need to worry much about unsubscribing - if your account should terminate then **the mailing list software will unsubcribe you automagically** after it was not able to deliver mails to you for two weeks.

Posting announcements

Q: How can I post an announcement (send a mail to "vim-announce")?

A: Only a few people can send mail to vim-announce; if you need to so, too, then please contact the list maintainer !

Mailing List Archive

Situation: You are not subscribed to any of the mailing lists but you want to take a look at them just the same. Or you have unsubscribed for a while and you want to catch up with the latest posts.

Q: Is there an archive of the mailing lists?

A: Yes, for each mailing list there is an archive. See the section about "VIM Mailing List Archive" below.

Mailing List Digests [971027]

Situation: You don't want to be disturbed by lots of mails from the mailing lists. Instead, you prefer to get all posted mails once a week - as a digest.

Q: Can I get vim mails as a digest?

A: Yes - instead of subscribing to vim@vim.org and/or vim-dev@vim.org, subscribe to vim-digest@vim.org or vim-dev-digest@vim.org respectively.

Licence to post [970910]

Situation: You want to be able to post to the mailing list, but you do not want to receive all those replies.

Q: Can I get a licence to post - but not receive any replies from the list?

A: No. Repeat: NO! This would defaet the very purpose of a mailing list - a simply reply to the list shall notify *all* members on the list. We know this could be done if we used "group reply" and all that, but it simply imposes more problems. Please take the time to subscribe and unsubscribe as needed - it has been made sooo simple. It's no more than sending an empty mail to the correct address. Thank you for your consideration!

Mailing List Software [971008]

Q: Which sotware is used to handle the vim mailing lists?

A: EZMLM and qmail.

Mailing List Host [971008]

Q: What kind of machine do the mailing lists run on?

A: The lists are processed on and sent out from babayaga.math.fu-berlin.de, a 386SX-25 PC running Linux. The name "babayaga" refers to a witch that appears in fairy tales from Russia as the machine babayaga resides in the student cafe of the department nicknamed "the witches' cellar" as it is "wizard friendly". :-)

Subject field [990101]

Q: Couldn't you add STRING to the Subject line of the mailing list?

A: Yes - but we won't, as this will introduce more problems than it solves. If you need such a string as your mailer is not smart enough to filter out mails from vim.org then you either need a *real* filter or a better mailer. Sorry.

prz and babayaga addresses [970716,990311]

Q: Why do my mails bounce when I send them to vim@prz.tu-berlin.de or vim@babayaga.math.fu-berlin.de?

A: The mailing lists once ran on the domain prz.tu-berlin.de - but that had changed in July 1997. Since then babayaga.math.fu-berlin.de serves the mailing lists. You are asked, however, to use the official addresses to make it easier for your fellow subscribers to filter email. Please update your email aliases! Thankyou.

VIM Mailing List Archive

[981104,010201] The Vim Maillists are archived at "yahoogroups.com" (formerly at egroups.com which is now part of yahoo.com!):

http://www.yahoogroups.com/list/vimannounce
http://www.yahoogroups.com/list/vim
http://www.yahoogroups.com/list/vimdev
http://www.yahoogroups.com/list/vim-fr
http://www.yahoogroups.com/list/vim-mac
http://www.yahoogroups.com/list/vim-multibyte
http://www.yahoogroups.com/list/vim-vms/

For the record: Some time ago there was an archive for vim mailing lists at the "Insitute of Informatikk" (institute of computer science) at the University of Bergen, Norway. However, this archive is no longer in function. This archive was maintained by Eivind Gjelseth eivind@ii.uib.no using the software MHonArc which is based on perl. Thanks for this, Eivind!

Mailing List - using the mailer "mutt"

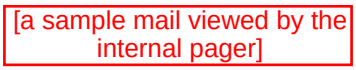
Mutt is a mailer hich understands "threading", that is it can group relating messages together - properly!

Here are two sample windowshot for you:

The first windowshot shows my mail folder with all the mails from the vim mailing list:

Mails in red are "From: Bram", mails in yellow have been sent with a broken Message-ID. Mails in dark blue are those which I have replied to (marked with an 'r'). And the mails in light green are from myself (these are also flagged with 'F'). Mails in dark green are CCed to me at guckes@vim.org (that's why you get to see it twice in the index). And mails in yellow on magenta have been sent by bad mail programs...

The second windowshot shows mutt's internal pager displaying a message:



The header is limited to the header lines that *I* am interested in to see, and it shows in the order I prefer; they are also colored by the colors that I have set. (Hey, you might not want any colors at all - fine, because no colors is the default with mutt.)

Quoted text is shown in its own color - and it can be hidden from view with a command (just type 'T' for the command "toggle-quoted").

Addresses are shown in its own color, too. (Email addresses have a different color than FTP addresses or Web addresses.)

Even the signature gets its own color (if you wish); it can even be stripped off automatically on replies.

And the blue tilde characters show that those lines do not belong to the message any more.

By the way, you can view each MIME part of the message on its own, and delete, print, or just save it. You can also "pipe" each attachment to some arbitrary program or script.

Please give this mailer a try. It works on every terminal and is really fast. It understands PGP, POP and IMAP. And much more.

About a hundred people on the vim maillists are using mutt with vim already. :-)

One more example:

Mutt allows to "limit" the index of messages to only messages which "match" a "pattern". This is quite useful, eg to quickly get a list of only the mails "From: Bram" which are announcements of the "available" new versions of version "6.0"; and, of course, you don't want to see the replies (which usuyally have mostly "re:" in the Subject line and occasionally a changed Subject, indicated by "was:"):

```
   ~f bram ~s available ~s 6.0 ! ~s re: ! ~s was:
```

Today [010514] this would give me this list in my folder "VIM":

```
Mutt 1.2.5i: =IN.VIM (threads) [37/28179] [N=24772,*=0,new=10]
13931   X 000709 Bram Moolenaar    ( 71) Vim 6.0a alpha available
14350   X 000716 Bram Moolenaar    ( 65) Vim 6.0b alpha available
14586   X 000723 Bram Moolenaar    ( 51) Vim 6.0c alpha available
14758   X 000730 Bram Moolenaar    ( 56) Vim 6.0d alpha available
```

```
14957   X 000806 Bram Moolenaar    ( 59) Vim 6.0e alpha available
15146   X 000813 Bram Moolenaar    ( 93) Vim 6.0f alpha available
15323   X 000820 Bram Moolenaar    (129) Vim 6.0g alpha available
15697   X 000831 Bram Moolenaar    (139) Vim 6.0h alpha available
17032   X 001015 Bram Moolenaar    (196) Vim 6.0i alpha available
17331   X 001022 Bram Moolenaar    (144) Vim 6.0j alpha available
17633   X 001029 Bram Moolenaar    (152) Vim 6.0k alpha available
17887   X 001105 Bram Moolenaar    (151) Vim 6.0l alpha available
18052   X 001112 Bram Moolenaar    (123) Vim 6.0m alpha available
18356   X 001119 Bram Moolenaar    (132) Vim 6.0n alpha available
18819   X 001203 Bram Moolenaar    (190) Vim 6.0o alpha available
19020   X 001210 Bram Moolenaar    (172) Vim 6.0p alpha available
19327   X 001217 Bram Moolenaar    (153) Vim 6.0q alpha available
19610   X 010101 Bram Moolenaar    (170) Vim 6.0r alpha available
20268   X 010114 Bram Moolenaar    (243) Vim 6.0s alpha available
20275   X 010114 Bram Moolenaar    (243) Vim 6.0s alpha available
20664   X 010121 Bram Moolenaar    (244) Vim 6.0t alpha available
21260   X 010204 Bram Moolenaar    (309) Vim 6.0u alpha available
21621   X 010211 Bram Moolenaar    (203) Vim 6.0v alpha available
22282   X 010226 Bram Moolenaar    (301) Vim 6.0w alpha available
23072   X 010311 Bram Moolenaar    (263) Vim 6.0x alpha available
23509   X 010318 Bram Moolenaar    (215) Vim 6.0y alpha available
23880   X 010324 Bram Moolenaar    (207) Vim 6.0z alpha available
24983   X 010416 Bram Moolenaar    (277) Vim version 6.0ab ALPHA available
25352   X 010422 Bram Moolenaar    (163) Vim version 6.0ac ALPHA available
25745   X 010429 Bram Moolenaar    (216) Vim version 6.0ad ALPHA available
25762   X 010430 Bram Moolenaar    (218) Vim version 6.0ad ALPHA available
26068   X 010506 Bram Moolenaar    (187) Vim version 6.0ae ALPHA available
26390   X 010513 Bram Moolenaar    (158) Vim version 6.0af ALPHA available
26756 N X 010520 Bram Moolenaar    (146) Vim version 6.0ag ALPHA available
27166 N X 010527 Bram Moolenaar    (165) Vim version 6.0ah ALPHA available
27516 N X 010603 Bram Moolenaar    (159) Vim version 6.0ai ALPHA available
27946 N X 010610 Bram Moolenaar    (180) Vim version 6.0aj ALPHA available
q:Quit  d:Del  u:Undel  s:Save  m:Mail  r:Reply  g:Group  ?:Help
```

So there you are then: Mutt finds thirty-seven messages (out of 28179) which match the given pattern. And this takes just under three seconds. (I am not making this up!)

[And, no, I do not know why Bram suddenly added the word "version" or made the string "alpha" upppercase - an extra warning, perhaps? ;-)]

Mailing List - Problems and Solutions

Broken Threading

> Threading just works when each reply has a reference to the previous mail. Unfortunately, people are using mailers with drop this information or do not give any at all. This mostly seems due to mailers by MicroSoft. I therefore discourage their use. Please use a good mailer[tm] such as elm, pine, or (best!) mutt. Thankyou!
>
> I wish I could name these mailers - but they apparently do not identify themselves with "X-Mailer" (are they afraid of anything?) or point to a DOS/Windows mailer that does it correctly, but I know none. Anyone?
>
> Mailers that break the threading:
>
> ```
> X-Mailer: Microsoft Outlook 8.5, Build 4.71.2173.0
> X-Mailer: Mozilla 4.51C-SGI [en] (X11; I; IRIX 6.5 IP22)
> ```

Unknown mailers - example: X-Priority: 3 (Normal) X-Incognito-SN: 599 X-Incognito-Version: 4.10.130 X-Mailer: Neoplanet Version: 2.0.1.398 X-mailer: Claris Emailer 1.1 X-Mailer: Microsoft Outlook 8.5, Build 4.71.2377.0

Back to the -> VIM Pages

```
URL:         http://www.math.fu-berlin.de/~guckes/vim/mail.html
URL:         http://www.vim.org/mail.html (mirror)
Created:     Fri Dec  1 00:00:00 MET 1995
```

Send feedback on this page to
Sven Guckes guckes@vim.org

Product Comparisons

Product Review: vim5.4

Lastest Reviews:

On 08-JUL-01, Mirek Rewak wrote: I discovered vim few years ago. I'm vim addict now...

On 04-JUL-01, Cam Smithers wrote: Vim is the best thing since sliced bread.

On 04-JUL-01, Vincent Foley wrote: I meant Vim gives TON of power and rapidity, not ON. Sorry about that

On 04-JUL-01, Vincent Foley wrote: Vim is my editor of choice. I mean, I find vim's screenshot sexual, how addicted am I? I made special vimrc files for latex, txt and bash modes, I try to learn at least one new thing everyday. I even make a daily vim cd with my vim config file, the latest stable built and a couple of text files. Vim gives on power and rapidity. I tried to learn Emacs this summer. I did learn some, but I was always back to vim. So I flushed Emacs and now emacs is linked to vim just in case. Bram, I love you!

On 04-JUL-01, Jason Katz-Brown wrote: I love vim. It is the most streamlined editor in existance.. period. Isn't that kinda a good reason to use it?

On 09-JUN-01, bapata wrote: I'm vi addict..

On 04-JUN-01, kaiser wrote: thanks

On 02-JUN-01, A person wrote: I will never say that Vim is easy to learn. It takes dedication (but the included tutorial, which takes about 30 minutes to an hour to get through, makes learning it quite easy). Once one learns its basics, one can then grow into it and seemingly never reach its end. With its advanced scripting (I've even stumbled across a simple file manager implemented completely as a Vim script), keymappings, etc., who needs Emacs for extensibility/programmability (granted, Vim doesn't support network/sockets access from scripts, but you can write scripts for it in Perl, Python, or TCL)? The commands for the most part are consistent (there are a few inconsistencies, but these are because the original Vi had them). I've never had it crash. I decided to try to learn Emacs and give it a fair shot, for an honest comparison. I got through its tutorial, and couldn't remember half the commands I learned. Back to Vim!!! Vim is simply the best text editor around, and it keeps getting better (5.8 has 70 new syntax files, 6.0 will have folding, vertical splits, and other features). The only problem I've ever found with it is it has trouble highlighting a PHP script I am working on with long multi-line string concat operations to build SQL queries, and that's probably just a problem with the PHP syntax file.

On 16-MAY-01, youtH wrote: Working with Emacs to me is like having supersize Mc Donalds Big Mac Meal, working with VI(M) is like enjoying a small but excellent delicacy.

On 09-MAY-01, Radomír Ludva wrote: It's just an editor, but it is editor. It's big small editor. I like vim.

On 27-APR-01, Chrup wrote: I've been using vi since 1984 and vim since I was made aware of it's existence. (V4 or something). Recently I added the 4GL I use to the syntax-highlight files and now, there is little, I would rather use. There is one editor which comes close: BBEdit 5.1.1 (not the latest 6.1) which can open files via FTP on my Mac - which is superb!! But for UNIX files, I have to use a PC and there is nothing in the PC world which is usable. Beginners and the spoilt-by-GUI people might not find vi(m) to be easy to get used to and usually give up before they know what they have - pity them :). Emacs, for simple editing tasks is just way to big to be useful. Also, the keystrokes I have to learn in Emacs far outweigh it's usefulness ...

On 22-APR-01, Mikee wrote: VIM is the bad mammer-jammer so scoot over Emacs ! :)

On 31-MAR-01, vim_fan wrote: Vim is a really cool programmers editor. Its the best. A programmed needs to edit text. Write code. He doesnt need variable size font. Or pictures in text. and such frills. Its all just sugar coating. Experts use VIM. Because VIM rocks.

On 21-FEB-01, bahram wrote: Everybody here is telling that vim is the best. Well, that's TRUE. I love vi, which I have used for 15 years. But then, vim make it really perfect. Many thanks to the developers.

On 20-FEB-01, Matt Lewellyn wrote: Vim is available for both character-cell and graphic displays. It is also available for Linux, BSD, Windows, DOS, Amiga, Solaris, etc. ... Check out www.vim.org and grab yourself a GUI version (even if it is a windows version, they work great!) --Matt

On 18-JAN-01, Jan wrote: The first time I saw vim, I found it sucked, I more liked GUI-editors, no I feel shame because I once thought that, vim is simply the best!

On 09-DEC-00, lphs wrote: kk

On 30-JUL-00, Kay wrote: VIM is the best editor that I have ever used. It is free! VIM 6.0d (alpha) comes with folding and vertical splits. Just perfect!

On 04-JUL-00, pcppopper wrote: VIM - There's just no competition

On 18-JUN-00, Weiguang Shi wrote: I haven't been using many editors. As I was introduced to Unix, 3 years ago, I began to use vi and a half year ago, I discovered vim. I know there are other editors out there. But vi's "strange" commands just came natural to me after some fun of rehearsal. I like to discover, and I am expecting to explore the new capabilities of Vim! My hearty thanks to those people who make Vim happen!!!

On 31-MAY-00, Weiguang Shi wrote: Please add vertical split function in Vim!

On 21-MAY-00, Daniel Lynes wrote: VIM is far and above the world's best editor. I use it for almost everything. I've used countless editors. Now that the 5.xx series supports syntax highlighting, I've dropped Boxer completely. There's only one thing that Boxer did that vim doesn't do, but it's not worth using Boxer instead of vim, and that was block marking, by column and/or row, rather than character-based marking. If you truly want a cross-platform editor with syntax highlighting for a wide variety of languages, vim is truly your only choice. I personally use it on 95, NT, Linux, DOS, and OS/2. If you're struggling with it, try a serious walk-through the tutorial that comes with it. That gives you a nice, fast, easy intro to one of the most powerful features of vim: global search and replace....and hot damn! Is it ever fast! Don't use another editor...just use the best...VIM :)

On 29-APR-00, Wolfram Fischer wrote: VIM - What could I say what others haven't already said about it? I know nearly ever free Editor, and I hate VIM - but I can't live without it. A day without VIM and live makes no sense...

On 28-APR-00, tapax wrote: vi is like a religion: Either you hate it, or you love it. Most vi-users did both, in exactly this order. People that don't like vi simply stuck at the first stage. It is true, that to vi you first have to get used to. But this is not a problem of vi. vi is *not* more complicated than other editors. It is only different. It is different than almost every of the thousands of editors available. But: It is worth to get used to vi. It is by far the most effective editor existing. And vim is the best vi I know. Btw: I *did* learn vim from the helpfiles, so it *is* possible!

On 21-APR-00, Herve FOUCHER wrote: I discovered VIM when the 5.x series began. Since that day, I have been using VIM EVERYDAY at work and at home as well. It is so funny to see my colleagues sweating using Notepad and other poor editor. Folding will make VIM the best editor on the planet !

On 03-APR-00, marsh wrote: globals in the map statement are so powerful as to render other editors useless.

On 29-MAR-00, JOohn Culleton wrote: I have used some version of vi for about ten years now. The nice thing about vim and gvim is that the old commands still work while new features are added.

On 22-MAR-00, Chris Horry wrote: Once you get past the initial steep learning curve it truly is the most powerful editor available, add to this a thriving Vim community and very approachable developers - Vim rules!

On 14-MAR-00, Bill Randle wrote: The only thing missing (and the only reason I occasionally fire up Xemacs) is "pretty printing" - i.e. Postscript output with syntax highlighting.

On 05-MAR-00, Jenson wrote: Well, at first, I use Emacs and now switch to VIM. So..., does it mean something?

On 29-FEB-00, Daniel Winner wrote: VIM is the best editor I have ever used for writing code. I simply can't live without its great features now, such as keyword completion, and its different modes. I love it!!!

On 28-FEB-00, A. Govindarajan wrote: The best free editor I have ever seen.

On 27-FEB-00, Jeroen Goudswaard wrote: Eventually Vim is just vi with a nice graphic interface on it. That means that the interface is NOT intuitive. I expect that Bram never meant to write an intuitive editor, just (one of) the best editors in the world. I think he succeeded. In my opinion, vi is the most powerful editor in the world, but the improvements of Vim make it just more fun to use it.

On 26-FEB-00, Dr. Strangelove wrote: Vim's got a steep learning curve, and the help files are not very helpful for beginners. I think the only way to learn vim is to get help from an experienced user. But if you get through the first hard week, it's the worlds most wonderful editor. The set of commands is rich, powerful and extendible, and even very complex operations can be automated. The syntax highlighting, block commands, and the quickfix feature makes it perfect for programming.

On 25-FEB-00, Sid Liu wrote: I would really like vertically split function

On 25-FEB-00, someone wrote: Thanks to the rich heritage of it's vi ancestry, vim is heir to a set of capabilities that have been thoughtfully extended to provide even greater functionality and cross-platform support. While some may claim that the vi interface is less than intuitive, tutorials and reference sheets are plentiful; a few well-spent minutes with one or more of them will reveal the simple elegance of vim's operation. Or, ask someone familiar with vi/vim command set - more likely than not, they'll be glad to get you pointed in the right direction, and will probably show you a few fancy tricks in the bargain.

On 24-FEB-00, Wade Turland wrote: Vim is the ultimate in scalability and power. It's feature set is more complete than the following: "Vim rules"

On 24-FEB-00, David wrote: VIM is nearly impossible to use for new users. The help files are no help. It's almost as bad as emacs.

On 21-FEB-00, Dragonmaster Lou wrote: My first experienced with Un*x editors started with Xemacs, but soon it became a pain to have to wait for it to load, and once I discovered that GNU Emacs also did syntax highlighting, I switched to that. However, it was still slow at times. Then a friend almost forced me to start using vim when he was a lab partner for an assembly programming class. Once I learned it, I realized how nice it was, small, fast, effective, and with my beloved syntax highlighting. Now I'll never go back to emacs unless absolutely necessary. Vim gives me all the features of emacs I actually used with blazing speed and small size. Don't want to learn cryptic vi commands as a beginner? Do what I did -- use gvim with its menus and optional modeless abilities. :) Then as you get more experienced you can take full advantage of it. When vim was able to take an editing job that used to take me 2 to 3 hours to do and reduce it to 1 thanks to its wonderful global search and replace commands (that I have yet to find an equal of anywhere), I realized this was indeed the editor of the gods.

On 20-FEB-00, Ken wrote: The best editor on earth!

On 20-FEB-00, someone wrote: Vim highlights everything! It's the best editor I've ever used!

On 18-FEB-00, Jeff Horner wrote: There will never be enough said about the usefullness of this editor. I cannot fathom not using it.

On 18-FEB-00, Jeff Horner wrote: There will never be enough said about the usefullness of this editor. I cannot fathom not using it.

On 17-FEB-00, Dirk Roorda wrote: with vim and perl I have the computer at my finger tips

On 17-FEB-00, Comp Sci Dude wrote: Try running emacs through a 9600bps connection...

On 17-FEB-00, Jan wrote: vim looks very good especially with a color xterm. It's also easy to make your own syntax mode for all kinds of propriatary languages.

On 15-FEB-00, Jim Crockett wrote: vim is the best!!! And I can use it at work (winnt), at home (linux), and at home (win98)

On 07-FEB-00, -bob,mon. wrote: I started with vi on PDP-11s, and I've tried every vi clone I could find on 8088 boxes and up. Vim was, and continues to be, an epiphany. (It's so useful, it actually slowed down my conversion to Linux :-) rave rave rave rave rave

On 07-FEB-00, Stephen Abshire wrote: VIM is positively beyond comparison and is the best editor available for free or at any price! VIM is truly in a class by itself.

On 04-FEB-00, Junior wrote: Vim is a REAL man's text editor. I don't know why anyone else would even bother with sissy programs like emacs, or even worse... pico.

On 04-FEB-00, someone wrote: There's even a Macvim !

On 04-FEB-00, someone wrote: I've been using vi since I was knee high to a grasshopper and Vim is by far the best version I have ever seen!

On 03-FEB-00, Chris wrote: I've been using VIM since pretty much since it's inception. I will never go back to anything else! Love it.

On 03-FEB-00, g kporku wrote: To do 1. Get VIM for Dos 2. Get VIM for Linux 3. Get Vim for Freebsd 4. Get Vim for Dec-Unix 5. Get VIM for Irix 6. Laugh at non-VIM users

On 03-FEB-00, SID wrote: Vim is great. I am impressed. I've been using editors since 1971 and VIM beats them all. I've tried almost all of them, even ed and great ancestor of them all 'QED'. VIM rocks !!!

On 02-FEB-00, Luis Ibanez wrote: Vim offers the best combination of powerful commands with reduced system requirements.

On 01-FEB-00, fmarshal wrote: The BEST vi I've ever used - and therefore the BEST text editor!

On 28-JAN-00, Benoit Hamelin wrote: I don't want to use any other editor anymore! Syntax highlighting is so great, as well as the multiple window support!

On 28-JAN-00, Emil wrote: Comparing Vim to Emacs is like comparing Linux to NT - the first is fast, tight and powerful. The second is big, slow and clunky.

On 28-JAN-00, David wrote: Damn I'm impressed.

On 27-JAN-00, Mark Waller wrote: I have used vim for two years now and it is excellent. By far the best implementation of syntax highlighting I have seen. All it needs now is folding.

On 26-JAN-00, 0xc0ffee wrote: Vim makes programming so much faster! You will never do so much editing with so few keystrokes, and you will never be see syntax highlighting for so many file types.

On 26-JAN-00, Paul Peng wrote: Vim is the best editor among all the platforms I used. The more you use it, the more powerful it is.

On 24-JAN-00, Unix Pro, Really wrote: it *is* sooooo much better than vi -- and it even runs on other platforms!

On 24-JAN-00, nobody wrote: PH33R 4R3THUS4$#@!

On 24-JAN-00, cripto wrote: vim w/ syntax color highlighting enabled and a tweaked .vimrc are all you need. throw your eight megabytes and constantly swapping editor out the window.

On 22-JAN-00, Lars Amsel wrote: best unix style editor waiting for the vim book to come

On 22-JAN-00, Garry Glendown wrote: My old SysAdmin of the first Unix machine I got my fingers on told me to learn VI, I'd never have problems on ANY *IX-system ... he was right ... and with the improved features of VIM, this old trusty editor beats the crap out of just about any competition ... now if they'd just get folding finished ... :-)

On 22-JAN-00, Tobiah wrote: Vim is like the sound of one monolithic corporation collapsing

On 21-JAN-00, NoRefill wrote: I like Vim so much that when I HAVE to program in Windows, I use Vim there, even if the IDE I'm using supports vi features. I'll admit that it takes a little getting use to it, too. I learn new stuff all the time.

On 21-JAN-00, Sean wrote: VIM is the best and runs circles around emacs and nedit!

On 21-JAN-00, Massilia98 wrote: THE best editor I've found work quickly & efficiently Why did I bought a mouse ???? ;-)

On 21-JAN-00, Dave La Rosa wrote: VIM is the best editor I have used! It really is great.

On 21-JAN-00, T wrote: VIM = SPEED!!! Split screen editing, multiple undos, identifier completion... if you aren't flying with VIM, you are hopeless! Take the time to learn the extra features, well worth it! It should be a requirement in colleges!!!

On 21-JAN-00, Mike Wilson wrote: A brilliant 'vi' clone with lots of useful extras. Well worth reading the help files to see what vim can do -- Control+P and Control+N completion very handy for crappy typists like me. Default colours for syntax highlighting dreadful, but easy to change.

On 20-JAN-00, Kenny wrote: If you type 60+ wpm like me (90+!), and once you get past the learning curve, it is the most amazing editor. I never have to use the mouse. And that means I can code full speed without removing my hands from the keyboard. VI ROX!!

On 19-JAN-00, darrell dupas wrote: this is an amazing program, increases your productivity dramatically , thank you very much to all involved, i use vim for linux, and gvim in x11, will try the win port soon.

On 19-JAN-00, John Klassa wrote: vim is getting more and more like GNU Emacs every day... Tons of settings and cryptic key sequences.

On 19-JAN-00, someone wrote: I don't use that many of vi's features, but the undo queue alone makes vim the better editor for me. Also, with vim, I am able to use the same editor on unix at work as I use on my Win95 pc at home.

On 19-JAN-00, Andre SIPAMIO-BERRE wrote: Vim is whatwe call an improvement of vi. I like the vi concept and the fact that we can still use almost all its features in a terminal emulator! Its a shame that nowadays people think GUI before ease of use.

On 19-JAN-00, Markus Großmann wrote: Never found another, that flexible editor. Whereever I go, vim comes with me!

On 19-JAN-00, anonymous wrote: vim is good

On 18-JAN-00, Domini wrote: Vim really sucks, as a web browser, that is... If you want a web browser, AVOID vim! :x

On 18-JAN-00, Nitehawk wrote: Vi is the obviously the best breed of editors around and clearly vim is the best vi around. What more needs to be said?

On 17-JAN-00, narsi wrote: Vim is a must have for any system. I load it on every OS that I work with. Anything else is a waste of time.

On 17-JAN-00, S Nicholson wrote: Vim, like good old vi, is a typist's editor - with an easy-to-remember and powerful command set. It's probably as intuitive as any bit of software is - but that's not really the important thing - if you're going to use a text editor for several hours a day every day (as I do). So far, I've had no grounds for complaint. (And it keeps getting better!)

On 17-JAN-00, Grisly Bear(Francis Smit) wrote: Vim is the best Vi I've found so far, and Vi is the best text Editor I've ever found!!!!!!!

On 17-JAN-00, Jenson wrote: vim is good

On 17-JAN-00, someone wrote: remember: intuitive just means "similar to other things you have experienced and therefore what you have come to expect." in that light, vim is indeed intuitive.

On 14-JAN-00, Marcin Dalecki wrote: Yust to make it clear...

On 13-JAN-00, Justin Beech wrote: Vi rules. Vim rules more. In a sea of computing chaos, it is nice that one thing doesnt change and still rules: vi(m) is the fastest way to get things done! multi-level undo and syntax coloring have cut the number of editor induced bugs I can put in code to almost zero. :make for syntax checking perl and taking you to the first error is so very very useful - check dejanews for that, search for perl vim vimrc IDE and you should find it. Vi(m) also was in my favourite .sig for 1999: Vi has two modes : one in which it beeps, and one in which it doesnt.

On 11-JAN-00, JJ wrote: VIM is the kind of Unix program where I tend to give up, launch another console and do a kill -9 just to get rid of it. For once I am actually trying to figure out how to exit properly. I mean: how about some vague onscreen hint on how to find the key mappings?

On 11-JAN-00, Matt Hawkins wrote: What can I say? Small, fast, extremely powerful. Can't be beat. :wq

On 09-JAN-00, Preben Randhol wrote: Vim is great. I like it more and more as I use it. There are some shortcommings in the area of indentning source code for other languages than C, but I guess that will come in the future. The few times I *have* to use Windows[tm] having gvim to edit code with, makes it more bearable. I hope vim won't be bloated like Emacs 20.x in the future.

On 08-JAN-00, Hwansoo Suh wrote: Absolutely the best editor I've ever experienced. I'm not considering any other editor. Since I have vim. ^.^

On 08-JAN-00, erik@telia.net wrote: it´s simply very powerfull

On 07-JAN-00, Christian Sage wrote: Having something even better than vi on all the platforms I use is great! (And not just in terms of the amount of time and money saved.)

On 07-JAN-00, Nagu wrote: Can't believe how I survived with just vi all these years!!!

On 07-JAN-00, Chris Williams wrote: Vim is THE way to edit almost anything. Want to cut and paste columns, no problems. DOS file/Unix file, problems. Coloured syntax, no problems. Macros, no problems. Humungous files, no problems. Simply the best!

On 06-JAN-00, Ray wrote: I would be lost without vim. I use it for everything. If I need to use a machine without vim, my first priority is to put vim on that machine.

On 06-JAN-00, Paul Toth wrote: Vim rocks! I love multi-level undo, syntax hiliting, multi- windows, word completion ,instant word search with * and jumping straight to a file with gf.

On 04-JAN-00, Clint Whaley wrote: vim is the first thing I add no matter what OS I am using. Multilevel undo makes going back to regular vi too painful to bear . . .

On 29-DEC-99, updewazoo wrote: Using vim since 2 series Got it to hang once! (Actually it's stty's fault :^) But every vim user should spend at least 2 hours in the help files -it reads like the Rosetta stone! "If you're really paranoid -just hit sync three times!" -Old Linux Proverb

On 29-DEC-99, Jens A. Nilsson wrote: My primary editor.

On 23-DEC-99, HanishKVC wrote: Nothing compares except may be emacs. Its SiMpLe, FlExIbLe, PoWeRfUl.

On 23-DEC-99, Tom Hasselbach wrote: I love vim. It is millions times better than vi. My favorite features: - Visual mode - Multiple levels of undo - Online help - Multiple windows

On 23-DEC-99, jfk wrote: cool appz!

On 21-DEC-99, Dan Green wrote: VIM is not for everyone. A lot of people may be a little overwhelmed by the complexity of the features. But if you code (Visual .* doesn't count), chances are you can figure out VIM from a tutorial in a matter of minutes and start using the advanced features within an hour. I've never looked back since I learned how to use it properly. For users of other vi's out there: color syntax highlighting kicks much butt.

On 20-DEC-99, Gnomon wrote: I use VIM as my standard text editor in DOS and Win32, mainly because I detest all the other editors that make such extensive (and annoying) use of the mouse. I don't want to poke around in menus, I want to get my work done - and VIM is the perfect tool. With a little imagination, it can also be used (along with some really ugly scripts) as a poor man's scripting tool - I currently have it set up to run through my bookmarks file, fetch the IP addresses for all the sites listed therein, and cache them in my network translation file (did I mention that I also run a Win32 port of cron? I really should just use Linux 100% of the tim). It's great - yet another way that VIM makes itself useful. The learning curve is nowhere near as steep as it's reputed to be, in case there are any new users out there. Within a few minutes, you'll have the basic commands memorized, and it's all downhill from there. VIM gives you a set of commands that can be massaged to do just about anything you want - a few moments of forethought will often replace minute of macro work in other editors (Qedit/TSE, Auroroa, TextPad, and the like - both are editors that I've used in the past, and have given up because of VIM). Regular expressions are a gift from the gods. VIM is the perfect editor.

On 19-DEC-99, anon wrote: If you've used DOS in a programming environment, (G)VIM has comparable features to the Semware editor, probably much more, w/o the block character graphics. It also does a nice translation of Unix-created text files if you run Gvim on NT The Help files are excellent and comprehensive. They deserve a better navigation mechanism. Installation on Unix via a package manager is much easier than on the Windows version. Although the Windows version wisely stays out of the registry during install, it needs some kind of self-executing installer. There are so many options, there ought to be a book dedicated to explaining it, or perhaps some default configuration 'themes' to get educated about what it can do.

On 18-DEC-99, Alan Keaveney wrote: Vim isn't an editor, it is an editing toolkit for people with nimble minds and nimble fingers. :wq

On 17-DEC-99, Kamesh Kompella wrote: Being used to a vi kind of editor (read no mouse) working in windows wasn't exactly fun. Also, working in development environments like JBuilder is cumbersome and slows down the computer. Now, with vim I can work on the command prompt, not tax the RAM and yet do efficient coding thanks to the color sensitive code.

On 16-DEC-99, Zoltan Arpadffy wrote: Since I have to leave DOS and my favorite editor MuliEdit… Vim is the only editor what can handle all my requirements: regular expressions, auto commands, syntax check, binary file editing, command recording, script commands etc. and all on NT, Solaris, SCO, Linux, OS/2 and even on VMS. C and Java are everywhere the same but as a consultant I had spent lot of time to learn OS specific editor… Thanks Vim not any more.

On 16-DEC-99, David Mankin wrote: The best syntax coloring, automatic indentation, and most configurable interface in a Mac code editor, hands down. If only it would talk AppleEvents to interact with other applications. I guess that's why it's open source, right?

On 16-DEC-99, David Gerard wrote: When I'm put in front of a new NT box, vim and slrn are the first things I install.

On 16-DEC-99, someone wrote: My company wants me to use Windows95, but you won't find a file called notepad.exe on it! Neither Frontpage. VIM does it all!

On 16-DEC-99, Peter Hooper wrote: 'vi' is intuitive for me, to the point where i'm hitting 'i' and ESC in my other editors on accident. vim's colour highlighting has saved me countless hours of debugging by showing me my syntax errors in advance.

On 14-DEC-99, someone wrote: The interface is great ... coz it uses command line interface (there's no graphical interface with irritating dialog boxes to get in Ure way) ... One other feature that really sets VIM apart from other editors is the regular expressions ...

On 12-DEC-99, Suydam wrote: Interface?! Ha! There isn't one last time I checked.....at least not in vim the way I'm guessing most of us use it. However, it's still by far the best editor out there for hardcore programming. Long live vim!

On 12-DEC-99, someone wrote: The interface of gvim 5.5 is better than that of 5.4. I was from unix and am working in windows nt, now. But I still use vim as my test editor.

On 12-DEC-99, Alan Meyer wrote: Vim is hard to learn but extremely powerful, fast, portable, and reasonable in its resource use. It's my first choice for complex editing tasks. It's less configurable then EMACS but seems to get more configurable with each release.

On 11-DEC-99, someone wrote: I love vim, but can't see how it could get a high rating for "intuitive interface". Vim would be far more loved by newbies if, by default, you could backspace over text after changing modes. That's why it took so long for me to get into it. Now I love it, of course, but it could have been made so much easier.

On 10-DEC-99, Wari Wahab wrote: Vim is way cool, although it sucks at first ;) Now I can impress my friends on the amazing thing I could do with a few key strokes.

On 09-DEC-99, Ronny Haryanto wrote: It really requires commitment at first, but it's worth all the trouble. Once you vim you can't stop! Trust me.

On 09-DEC-99, Dariusz Mejer wrote: I started with jot (SGI O2), simply I hated 'VI'. Then I got into vim website. Thats the way it started. Now I'm the 'VIM' maniak.

On 08-DEC-99, Sylvain VIART wrote: Vim is a great editor for those who want to start learning a powerfull editor. Beginner need to be patient, the results will come a bit later... ;) The developpers of Vim are doing a very good job ! I realy enjoy this editor.

On 08-DEC-99, Paul Adams wrote: Vim is excellent implementation of vi. The extensions are useful and fit right in with the vi philosophy. Vi *is* user friendly - it's just not beginner friendly.

On 07-DEC-99, Dave Kaplan wrote: As a relatively new, inexperienced UNIX hack, I thought vim was easy to install, and it really works great for my programming needs.

On 07-DEC-99, Felipe wrote: I come from the mainframe area; when I started working with vi I really liked it. But now that I have discovered vim ... this is bay far the best editor I have ever used!!

On 06-DEC-99, Patrick wrote: The interface of vi(m) is counterintuitive for people used to any other editor. gvim, however, is as good as, say, XEmacs. The efficiency of vim is unmatched, IMCO.

On 06-DEC-99, Michael P. Soulier wrote: I always prefered Vi's interface which, while not intuitive, is very efficient and not hard to learn at all. I switched to Emacs because I wanted syntax shading, but it kept screwing up the shading of my Perl code, plus it kept moving things on me, and doing a dozen other things automatically that I didn't want. Then someone told me about Vim. Small, fast, syntax shading, lets me make the decisions, Vi compatible, and available for free on all platforms that I use. I fell in love. It's everywhere that I am now.

On 05-DEC-99, steven baker wrote: by far the most powerful and feature rich editor for *any* os platform

On 05-DEC-99, someone wrote: So I've been blissfully using vim for several years (and vi for many, many before that). But vim was always older copies (used under DOS and linux). Now I discover that vim's being updated and doing wonderful new stuff. I'm in love all over again. Where did the big V come from? very un-unixlike :-)

On 02-DEC-99, Chris C wrote: I agree with another comment, VIM (and its predecessor vi) does not have an 'intuitive' interface. As far as I'm concerned, that's not a problem, I'd much rather have a /consistent/ interface across platforms. Backward compatibility with vi is useful as well, because when I go to sies with just vi I can still do what I want (even if I do miss the extensions in VIM).

On 02-DEC-99, Shannon Fang wrote: VIM is splendid. It meets (almost) all my editing needs. Only some time I found it hard to use for example some weird commands...

On 01-DEC-99, Sinan Unur wrote: ViM offers the right balance (for me) between features and extensibility. I use it only for source file editing (mostly C, Java and TeX). It's small, fast, and beautiful.

On 30-NOV-99, Daniel Tomko wrote: I don't think your VIM survey is a very good one. Thie first question is WAY off. VIM wasn't designed to be intuative, it was designed to be powerfull and efficient to use. To a ceratin degree, these are mutually exclusive. I love this product and use it every day, but given your survey, one might think I don't like it at all.

On 26-NOV-99, Venkatesh Potdar wrote: Very powerful. But it is not possible to "click" on the output of things like "grep" or the errors from a compiler output. I like to use the keyboard a lot (rather than the mouse). But, I feel, for the output of things like "grep", being able to scroll through the results and being able to click/hit is a priority for me.

On 25-NOV-99, dmitchell wrote: I would be lost without vim. I use it on Windows 95 and NT, Linux, and Solaris -- all work exactly the same. Because customization is so easy and portable, I am able to enjoy a great deal of consistency, regardless of the platform. And the vim features are just incredibly powerful! When people see me edit code in vim, they are blown away by the amazing features and speed it gives a user.

On 24-NOV-99, Matt Aylward wrote: Vim lets you, the user choose. You can run it lean and fast in an xterm when you just want to get the job done, or bulk it up with any number of extensible guis when luxury is required (even then it is lightning fast). I've used vim for three years on Linux, Solaris, Digital unix, NT, '95, '98 and I still haven't tested half it's tricks.

On 24-NOV-99, someone wrote: VIM is undoubtedly the best vi clone around. I'm not fond of point-and-click editors, and nothing beats a good vi editor for speed and flexibility. To the VIM authors, excellent work. :-)

On 23-NOV-99, cyril tham wrote: I've used editors on Unix, Linux, and NT. And they include the very good emacs, vi, and notetab by Fookes. But none comes close to the speed, raw power, and extendibility as VIM 5.5. It is THE complete editor on any platform. Thanks for a great product. It is the best!

On 19-NOV-99, Steve R. wrote: Its the cats meow!

On 19-NOV-99, Radheshyam Bhatt wrote: I think Vim is awesome, because the features it contains are rare to find in any other text editor.. And with Syntax Highlighting, it brings the fun out in programming.

On 19-NOV-99, Michael D. Rogers wrote: Great editor. Emacs is too slow. As proof, try this: load a large C file. Select all text. Reindent. Note: emacs users shoul go out and run some errands until indenting is finished.

On 18-NOV-99, someone wrote: I'm a unix programmer recently ported to windows. I think VIM is the next best thing since sliced bread. GREAT PRODUCT!!!

On 17-NOV-99, Patrick wrote: Included in my ubiquitous tools toolkit alongside kermit and perl. Very nicely done!

On 16-NOV-99, Charles Stepp wrote: Only free editor that can easily handle editing very large files on my iMac.

On 16-NOV-99, Godfried wrote: A perfect non-intuitive vi-lookalike. Great syntax highlighting, also provided for exotic languages. Even better than good old vi itself...

On 15-NOV-99, Warwick wrote: Intuitive? Who cares, so long as it feels like vi!

On 14-NOV-99, RLB wrote: Ok to fix some stuff from the last comment. syntax formatting is syntax highlighting, & i forgot to mention what else really kicksass in ViM, splitscrens! Gotta love 'em! Oh, & we can never here this enough.. ViM kicksass!

On 14-NOV-99, RLB wrote: Anyone who disses VIM should be shot & hanged by their balls! I use VIM everywhere, no matter what type of machine i'm on. I can usually grab a version for it. :) By far it is the greatest editor of all time. Syntax Formatting, gotta love it. I have a friend that recently converted to VIM from emacs & can't imagne life without "Syntax Colouring" as he calls it. Overall, VIM has many features, with a side for everyone. :)

On 12-NOV-99, revadigar wrote: There is no question about VIM being the greatest editor of all times.

On 10-NOV-99, someone wrote:

On 09-NOV-99, Paul Vlaar wrote: Vim's keyword syntax highlighting has made me a better programmer than any book in the world :)

On 07-NOV-99, ksteinbe wrote: Ummm, Vim kicks ass? Cut my teeth on the HP-UX 9 version, snuck the NT version onto the boxes at work (amazing how many people can't live without it now), and won't consider anything else on my linux box A truly great piece of software!

On 06-NOV-99, green wrote: People, forget about Words, WordPerfects, and stuff like this. VIm is the best even for Win95!

On 05-NOV-99, Dan Scott wrote: Vim is an excellent cross-platform editor solution. I use about 8 different operating systems at work, and it's fast and easy to configure them all to act in an expected way.

On 05-NOV-99, someone wrote: vim is the best thing ince sliced bread.

On 04-NOV-99, someone wrote:

On 02-NOV-99, Mark Collett wrote: I've been using Vim for 6 years now. Even when I went to a MS development shop, I still used Vim for my power editing. My many thanks go to the contributors. I find myself evangelizing Vim to everyone that will listen!

On 02-NOV-99, Neville Sinnamon wrote: I've been using vim since 1996 (vim4.3) on NT OS2 and UNIX to write SQL PERL Java C and C++ programs. After joining a new project I set up a tags file and path and no other editor can touch vim

On 27-OCT-99, Tom wrote: Been using vi++ type tools for a few years now, but vim really is an improvement. Not quite as powerful as emacs, but then who wants an operating system as an editor?

On 27-OCT-99, Brian Goodwin wrote: My search for the perfect text editor is over!

On 26-OCT-99, RS3 wrote: The BEST!!!!! The first unix editor I used with my former company was not available when I moved to my new company. After moving to my new company I, needed to learn a new text editor. At first, I tried Emacs but soon after, switch to Vi. Used Vi for over a year. Then I found Vim. Love at first sight!!! Istalled Vim on every computer I use, at home and at work. I can no longer live without Vim. Many thanks to those involved with the Vim product. Keep up the great work.

On 26-OCT-99, hzo@gmx.net wrote: Learning VIM is like learning to handle a sophisticated tool. First you have to invest some time and energy to master it. But as soon you have some experience, you can produce much better results in much shorter time and you don't want to be without vim anymore.

On 25-OCT-99, Jordan wrote: Vim is the most powerful editor known to man. Vim is an art, and an art worth your learning.

On 24-OCT-99, glauber wrote: There are many good vi clones out there, but vim is the only one that implements all of the vi command sets, plus countless additonal features, all good. For me, what made the difference was mainly the multiple undo/redo and the help system, plus the fact that vim gives me the ability to run the exact same editor in every platform i work with. Because of the help system, vim is the best way to learn how to use vi. Once you start using the customization features, you can make it into something that doesn't look like vi at all (e.g.: all menu and mouse-driven), but still keeps all of the underlying power. I prefer the more cryptic but shorter traditional vi commands.

On 22-OCT-99, AKH wrote: Best editor for programming

On 20-OCT-99, Tim wrote: Once the initial learning curve is mastered, VIM is the most powerful editor out there.

On 16-OCT-99, E M Collins wrote: Vim is not the easiest tool to learn but it more than pays off the time you invest inlesrning it with its speed, flexibility , and most of all its stability, I wouldn't want to be without it now. Its like a beatiful well crafted hand tool, it ain't always the most well featured or even the easiest thing to use but it sure does its job well.

On 15-OCT-99, Casey Zacek wrote: I think an 'advanced users' tutorial would rule... or maybe just an elvis2vim tutorial, as that's what I am now... An Ex-Elvis-User. Missing feature: Elvis's :bb (buffer browser) is beautiful. Am I just missing something?

On 13-OCT-99, Alan Hengle wrote: I have installed Vim on our HP-UX box at work and people love it. Especially people who don't really like vi. There are so many features, I am always learning something new through the fantastic help system. What we really like here is the syntax highlighting and multiple undo's.

On 12-OCT-99, fdj wrote: Those who complain about vim's size should know you can compile a tiny version with only the features you need. It's fast and very stable -- the stability complaints folks have with vim are usually related to the lates features, esp gui. The standard vim from a terminal or gui without extensive features works great; the standard vim with lots of features works great, too, but you might discover (and hopefully submit a report on) a bug.

On 11-OCT-99, scipio@math.tau.ac.il wrote: I just answered a survey (it went like this) what is your favorite development enviroment? vi. I can't find find in my options... I what tool have you used to created you homepage? A: vi I cant find it are you sure... what tool do you use for text processing? (Word,Wordperfect, blabla) vi I don't have it!!!! what tool do you use for emails? vi. are you sure? what tool do you use for HTML editing... let me guess vi... well they learned something!

On 10-OCT-99, Alisdair McDiarmid wrote: Vim is my only real development tool. I use it on my Linux machines at home, on the UNIX machines at work and on the Windows PCs at university, and it's consistently fast and productive on all platforms.

On 09-OCT-99, Bob Heckel wrote: Vim is the most versatile program that I use. Multiplatform means that the initial steep learning curve pays back geometrically. The help system is unmatched by any software that I am aware of.

On 09-OCT-99, Matthew Daniel wrote: ViM is designed around the concept that you want to work as little as possible on your file. Among its primary strengths are the numerous ways that repetitive actions can be automated. There are a billion other things that I could advocate about ViM, but the above is the most concise answer I can give to "why Vi?". Bite the bullet and tackle the monsterous learning curve: you'll be glad you did. On a side note, it scales well into a Dvorak keyboard, since there are VERY few (if any) keystrokes like CTRL-x-s to execute. You haven't known pain until you try to use XEmacs with Dvorak keyboard.

On 08-OCT-99, Dave wrote: Actually, I prefer Elvis, it's smaller, less buggy, and still packs the power. Vim is great for those wanting everything, but Elvis is better for those that just want a good, solid Vi editor.

On 08-OCT-99, Tuncer M. Ayaz wrote: You just have to experience the beauty of ViM and its user-friendliness. I can use ViM on all of my platforms including Linux, BSD, BeOS & Windoze. Just give it a try and you will be catched up by it forever. It's good to see that vim keeps up the way it can be controlled in the GUI-version, because I'm definitely a shell-weenie.

On 05-OCT-99, bruce w wrote: I switch platforms a lot, and having an editor like vim that works the same on all platforms is a BIG help.

On 04-OCT-99, Ulf Kister wrote: Vim is one of the most effective and useful pieces of software I have ever used.

On 03-OCT-99, brent wrote: Vim rocks! -- Never login without it!

On 02-OCT-99, blux wrote: vim is small and fast, nice syntax files too.... only bad thing is gvim, but the regular textmode is *GREAT*

On 02-OCT-99, ed wrote: I consider vim superior to vi. HOWEVER, I view it as a superior vi--neither a competitor nor a replacement for emacs

On 01-OCT-99, Ola Ormset wrote: Altough VI has a rather high learning threshold it is the only thing that works for me now. Changing modes was a bitch at the beginning, but now I love. Besides, VI is a _text editor_ and good at it. Unlike Emacs who tries to be the Everything VI concentrates on the important aspects of a systemadmins life - writing code and changing configuration files.

On 30-SEP-99, JeongRim choi wrote:

On 30-SEP-99, Murali Sadagopan wrote: As a long time vi user I find VIM excellent to use on Unix as well as Windows95/NT.

On 30-SEP-99, Hrvoje Blazevic wrote: What - use Vi? As long as emacs is there -- no chance!

On 29-SEP-99, Paul wrote: since my first days in Linux I've always admired vi(m) users. They seemed to get their work done so much quicker than me, using my joe. Ever since I've used vi myself I've had trouble using other editors because I missed vi(m)'s superb feature set. Thank you, Bram Moolenaar, for one hell of an editor !

On 29-SEP-99, Tom wrote: Vi IMproved is an honest to God text editor. Infact it's God sent :). It gets the job done in absolutely no time at all and is flawless on all platforms. Just like anything worth doing it takes a bit of proficiency to master it's subtle mysteries, but a cheat sheet of 10-15 commands is just about 95% of everything anyone ever uses. I love VIM!

On 28-SEP-99, Keith Hanlan wrote: You're asking the wrong questions. Who cares about an intuitive interface? I want speed and support for my "101-button mouse" instead of the frustrating 3-button mouse. Vi, like Emacs, has a steep learning curve but that's because it is designed for power and efficiency, not casual use.

On 28-SEP-99, anonymous_meept wrote: VIM is the BEST cross-platform editor ever! To date, I've used it on Linux, *BSD, Solaris, BeOS, WinNT, and MacOS, with negligible difference between the various ports. GO VIM!! =)

On 27-SEP-99, jason king wrote: I learn ex at uni in 1989 on half-duplex QUME terminals .. when I saw vi on a vt100 for the first time it blew my mind that you could move around the screen now .. on my NT4 machine with vim5.4 enabled with Perl .. it's like heaven .. easily the most insanely powerful editor that anyone could ever want thank you to the gods who took the time to make my dreams happen long live the keyboard .. die you stupid mouse die !!!!

On 27-SEP-99, David Spriet wrote: I have been using Vim for about 3 years now and I would like to say thanks to all the people who created and work on VIM.

On 26-SEP-99, Aziz A Mohammed wrote: Vim is the best editor.

On 26-SEP-99, Bram Moolenaar wrote: If you think Vim is _not_ the best, let me know and I'll fix it in the next version!

On 25-SEP-99, Evan wrote: I believe the system requirments number is your opinion on them. Vote higher if you like the system requirements, vote low if you dont (ie, think it requires too powerful a machine)

On 25-SEP-99, confused wrote: can anyone explain the system requirements field? is a higher number indicative of less system requirements? or do lower numbers mean less system requirements?

On 24-SEP-99, Linus T. wrote: VI rules! :)

On 24-SEP-99, Aran Anderson wrote: Vim is the best!

On 24-SEP-99, Fernando Larrazabal wrote: I love VI

On 24-SEP-99, Tony Kuphaldt wrote: Vim is what Vi should have been. Simple keyboard commands, most basic commands not requiring simultaneous keystrokes (, etc.), but with editing abilities rivaling the best "modeless" and mouse-driven editors out there. Very portable and free, too!

On 24-SEP-99, Th. Ramming wrote: Best Editor for Programming if you have been beaten to learn the vi interface. gvim's ability for configuring menues helps a lot to spread it over Windows User.

On 23-SEP-99, FooLman wrote: ViM simply rocks. I use it for codin, email, and text editing on dos/windows, linux, irix. So everywhere i need to edit anything.

On 23-SEP-99, raman wrote: Vim is truly the best editor, hands down. I use it for mail, coding, admin, html, script writing. I love it...get jtags too

On 23-SEP-99, Sumo wrote: Vim is sheer brilliance. The features of almost all other editors, and all done through the keyboard, no icky mouse grabbing.

On 23-SEP-99, captain scarlet wrote: vim is good but really needs to have an easy one click download link in the home page. With auto self-extracting for windows 98 would make this as easy to install as Lemmy

On 22-SEP-99, Joe King wrote: Running Vim has increased my stamina, enabling me to run faster, jump higher and go further.

On 22-SEP-99, Herman Tamas (onetom) wrote: I play the recorder. Using vi is a bit similar. I've just use it for 4 month, but I've already fallen love with it. (My first Linux experience: 1999-jan-05)

On 22-SEP-99, IByte wrote: Am I missing something? Why does everyone think VI(M) is so great?

On 22-SEP-99, IByte wrote: Am I missing something? Why does everyone think VI(M) is so great?

On 22-SEP-99, Daniel Winner wrote: Excellent Editor once you are familiar with how to use it. It will never go back.

On 21-SEP-99, myrddin wrote: power beyond belief. I've been using vi/vim for a decade and am -still- finding better/faster ways to do things.

On 21-SEP-99, Andrei wrote: The best program I've used. Nice balance of power/configurability/ease of use. The only perfectly documented open source program I've seen.

On 21-SEP-99, VIP wrote: Vim is really THE BEST editor!!! Ond it is on all of UNIX systems :) And its so easy and professional...

On 21-SEP-99, Jonathan wrote: Its a way of life really ;-}

On 21-SEP-99, someone wrote: Editing power, that windows users will never understand. What a pity...

On 20-SEP-99, Fernando Sanchez wrote: I've met VIM 3 months ago and I've made it my personal program editor. It's one of the best software products I've ever seen. I really recommend it.

On 20-SEP-99, Keithel wrote: The Only better thing that EMACS has that Vim doesn't is full support for a common programming/scripting language. Syntax Color highlighting is awesome, regexp support is great, and it's a breeze to set up... and, it's available for nearly every operating system... Be, Amiga, Atari, etc... included!

On 20-SEP-99, Fabrizio Inserra wrote: Syntax highlighting implementation is better than any other editor. Only VIM can FULLY customize his color !!! BEST OF ALL.

On 19-SEP-99, John Mullin wrote: regular expressions, replace patterns, scripting, block editing ... things I use every day and would be far less productive without. An excellent editor/environment.

On 19-SEP-99, someone wrote:

On 18-SEP-99, Kalath wrote: VIM is pure class. Gvim is the editor of the gods.

On 18-SEP-99, Vim Lover wrote: VIM is the BOMB. The syntax highlighting is great. I have not had to spend much time getting it up and running. The split screen editing ROCKS. Move over EMACS you now have a worthy compeditor!

On 18-SEP-99, gerald marewo wrote: I am new Linux user who has been looking for the best text editor. In vim I have found peace and stopped searching. Hope to make the best out of it

On 17-SEP-99, Chad Bearden wrote: One interface for all OS's. Tons of features! Wish there was a better tutoria for beginners.

On 17-SEP-99, Tony Clark wrote: Vim 5.4 is great, only things that beats vim in te editors competition is emacs doing 'perl code debugging, three way diffs and virtical screens :)'.

On 16-SEP-99, Michael Hartley wrote: if you include gvim with vim, you have a great X programmer's editor..

On 16-SEP-99, someone wrote: Vim 5.4 is great beyond all possible infinite dimensions of greatness. Well, okay, maybe not that great, but better than your editor. Switch.

On 15-SEP-99, Matt Dunford wrote: If you find yourself typing all day, Vim is the way to go. Doing the maximum amount of things with the fewest keystrokes is what it's all about.

On 15-SEP-99, Dave Warner wrote: I use it on everything from my old 386 laptop running linux to my NT workstation to the HP-UX server at work. One interface, one command set. Fast and complete.

On 15-SEP-99, David Weinehall wrote: Now, if it only could do my laundry and get me Cola for nightshift- hacking... Part from that, it's probably as perfect as one may want it...

On 15-SEP-99, D.Mon wrote: I edit anything non-binary in Vim, it helps me to do it fast.

On 15-SEP-99, Bill Hill wrote: Fast and powerful on all the platforms I need.

On 14-SEP-99, slave-zeo wrote: i guess i am not old-school elite enough. anyhting to do with vi scares the hell outta me :)

On 14-SEP-99, Brian Caulfield wrote: Vim is very extensible, partly by its scripting language and syntax highlighting files, but more so by its good integration into Unix. Filtering through external commands rocks my world! Plus vi keystrokes are included in GNU Readline (just like emacs), which is inside bash and perldebug.

On 14-SEP-99, H. Shenoy wrote: Vim is the new synonym for the word "editor". Its easy when U know how!

On 14-SEP-99, totoy wrote: what more can you ask for an editor that is so lean yet packed a heavy punch on features and usability.

On 14-SEP-99, Herve Foucher wrote: vim is the best editor for me. Its functionnalities and speed are impressive. Change to VIM !

On 13-SEP-99, g dubya wrote: Once you figure it out, vi is just the greatest text editor there is. You only have to look at the screen about 10% of the time to see what's happening. vim does pretty much everything standard vi does and more, plus it runs on Windows and Macs. What more could you ask for?

On 13-SEP-99, Leon Breedt wrote: VIM. There is no substitute. :)

On 07-SEP-99, Steve wrote: VIM - the quickest to pickup, on every linux box no doubt, simple yet powerful

On 03-SEP-99, Sean Leach wrote: Used to be an Emacs guy. Learned the errors of my ways and am now a Vim guy.

On 30-AUG-99, Trevor van Bremen wrote: I investigated every other 'VI'able text editor and VIM just cleaned them all up

On 30-AUG-99, David van Popering` wrote: vim just makes sense. Its the best out there. The End.

On 28-AUG-99, someone wrote: Well I rarely use grep, and the only nix style box I use with any regularity is my own, but I like vim because of syntax hilighting for all my favorite languages, the fact that I know its interface and like it fairly well, and the fact that a blackhole doesnt open in my computer every time I try to start it.

On 22-AUG-99, Colin wrote: As a complete newbie to Linux/Unix with virtually no prior DOS/Windows experience either, but forced to put together a couple of Linux boxes for work, I'm still trying to figure out why anyone would prefer emacs to vi for the sort of basic text editing I had to do to get these computers running the way we needed them to run. As a newbie, I have to keep a reference book close at hand anyway, and with Linux for Dummies at the ready, I can at least use vi. Emacs?!... Incomprehensible!

On 21-AUG-99, Ethan Baldridge wrote: Whether you use elvis or vim, some clone of vi is ubiquitous. It runs on everything that runs UNIX. Some say it's not unix without grep, I say it's not unix without vi!

On 20-AUG-99, Cougar wrote: While vi is indeed everywhere, vim just plain kicks. Fast, colorful, smart.

On 20-AUG-99, Like it shows names anyway... wrote: Let's face it you don't use vi for the interface or the features or any of the other things that are on this poll. You use it because like Buckaroo Bonzai said "wherever you go, there is vi". No serious sysadmin uses emacs because they all use vi, because it's everywhere. I'm forced to underate vi because of the way this system is set-up, but it's the only one on the list that I use anymore.

On 20-AUG-99, Guy Hunter wrote: I have tried the others, and I find VI to be the only one for me. I like that ol' skool text!

On 17-AUG-99, Morgan wrote: very fast, but lacks some things that emacs has the need to go out of input mode to save is also annoying

On 17-AUG-99, someone wrote: vi is vital and fills programmers with vigor!

On 17-AUG-99, someone wrote: VI FOREVER! EMACS NEVER!

On 15-AUG-99, Laurence Hunter wrote: I HATE HATE HATE vi, vim, gvim or whatever it wants to call itself! Hideous extreme. Only a severe nerd would get turned on by this grotesquely archaic editor... apart from that it's ok. ;)

On 11-AUG-99, marc wrote: I have become a follower of, VIM, Debian, Perl, Qmail, Perl, and FVWM2...

On 10-AUG-99, Nick Moffitt wrote: The great thing about vim is that it does syntax color highlighting and multiple windows. All that, and a properly compiled version still loads several times faster than EMACS.

On 10-AUG-99, someone wrote: vi rules!

On 08-AUG-99, 17482 wrote: This is a great product!

Click to add your comments. . .

Vim book errata

The site you are viewing is a part of
The Vi IMproved Editor Webring
[Previous | Next | Previous 5 | Next 5 |
Random | List Sites]

This is an unofficial list of mistakes for the first book on Vim.

Don't be discouraged by the number of items. Most are just small mistakes that don't interfere with understanding how Vim works. However, there are a few errors in examples, which are quite annoying.

The book is written by Steve Oualline and published by New Riders Publishing. You can find information about the book itself here. If you order it from this page, a percentage of the sales will help poor children in Uganda!

Some items are included from the official errata list. These are marked with -official-. You can find this list in PDF format here.

If you find an error that isn't mentioned yet, please send an e-mail to Bram@vim.org. Use the same format as in the table below.

Last change: 2001 July 6

Page iv, Copyright	-official- mistake: This mentions the standard copyright notice, but the book is published under the OPL (Open Publication License), as is mentioned on the back cover. The paragraph starting with "All rights reserved..." should be replaced with: "This material may be distributed only subject to the terms and conditions set forth in the Open Publication License, v.1.0 or later (the latest version is presently available at http://www.opencontent.org/openpub/)."
Page v, Chapter 9	error: "Command" is written twice
Page xxi, Acknowledgements	-official- typo: In the first sentence: "a high quality Vim clone" should be "a high quality Vi clone".
Page xxvi, How to read this book, second paragraph	-official- mistake: "a Jewish Arab living in Hong Kong" should be "Jewish Iranian living in Hong Kong".

Page xxvi, Notation	spelling: "approbate values" should be "appropriate values".
Page xxxi, Lines 7 to 17	error: Appendix D (Command-Mode Commands) is omitted, and Appendices E through I are labeled as D through H.
Page 6, above figure 1.3	mistake: "command mode" should be "Normal mode".
Page 6, figure 1.3	minor: Indent in the figure differs from what is shown in the text above it. In the text, remove the indent from the line starting with "Found".
Page 6, Moving around	mistake: in the first sentence: "command mode" should be "Normal mode".
Page 7, Deleting characters	error: It says "eight" twice, that should be seven. Figure 1.4 shows eight times "x", that should be seven as well.
Page 8, last Note	error: "The v command" should be "The u command".
Page 9, top	layout: The "xxxx" doesn't line up with the word "very" above it that is to be deleted. Same for "xxxxxx" below "turtle".
Page 9, Getting Out	-official- error: "This command writes the file and exits" should be "This command writes the file if modified and exits".
Page 11, figure 1.8	-official- The last line ends in "for the turtle.", should be "for the turtle!!!".
Page 12, top half	-official- typo: "CTRL+]" should be "CTRL-]". Appears twice. And "CTRL+T" should be "CTRL-T".
Page 12, at 60% down	error: A) is missing after "(for example, CTRL-A".
Page 13, table 1.1	error: In the "Prefix" column, "v" should be "v_", "i" should be "i_" and "c" should be "c_".
Page 13, last section	mistake: It says "all the movement commands" and then goes on to explain the append command "a". It should say "most commands".
page 16, figure 2.1	layout: Not all arrows point to the right character. See figure 18.1 for correct arrows.
Page 17, 50% down	minor: There should not be a : after "5f<Space>" (in bold font).
page 17, figures 2.4 and 2.5	layout: The arrows point to the same characters in both figures, but they should point to different ones: "f" at the mentioned character, "t" to the character before it.

page 22, figure 2.12	error: It mentions "Changed word (screw<blank>)". This should be "Changed word (screw)". The "cw" command doesn't change the following blank. The highlighted "foul" should be "screw".
page 22, figure 2.13	error: The seventh command "j>" should be "j^". The eight command "f>" should be "f<".
page 23, figure 2.14	error: the text says "two lines" but there are three lines.
Page 24, Line 6	minor: "If a count is specified," should be followed by "then," not "the".
page 25, figure 2.18	error: a"<Esc> should be $a"<Esc>
page 25, Digraphs	-official- error: to enter the copyright sign use CTRL-KcO (capital O instead of zero).
page 29, Highlighting	mistake: Change "search for include, for example, the results in all the" to "search for include all the".
page 32, figure 3.10	error: Several occurences of "the" are not highlighted. The "y" in "they" should not be highlighted.
page 46, figure 5.2	unclear: The CTRL-Wk command applies to the inner, short arrow on the right side, it should be moved inside the large arrow.
Page 48, The :new Command	-official- minor: Add this at the end of this section: "CTRL-W n" is equivalent to ":new".
page 57, below figure 6.4	mistake: Move "as shown in Figure 6.5" from then end of the second sentence to the end of the first sentence.
page 62, figure 6.11	error: the highlighted text should not include the space between "long" and "line".
page 78, figure 7.15	confusing: It's not clear what the arrow points to. It should look like figure 7.14, but exclude the lines with "{" and "}".
page 81, top	missing: Change "The following command goes to the tag on the list" to "The following command goes to the next tag on the list" (insert "next").
page 82, figure 7.21	error: The text below the right picture says "CTRL-W CTRL-[", that should be "CTRL-W]".
page 83, 60% down	typo: "g CTRL-]" should be "gCTRL-]", without the space.

page 87, figure 7.25	error: "parse error before ']'", the ']' should be '}'. Appears twice.
page 87, figure 7.26	error: "int i=3" should be "int i=3;". The four lines should be indented the same way.
page 88, figure 7.27	error: "int 1=3" should be "int i=3;". The four lines should be indented the same way.
page 87, Note	reference: "(as discussed later in this chapter)" should refer to chapter 23.
page 90, above Other Interesting Commands	mistake: "the first error" should be "the first match". Appears twice.
page 93, first note	error: "The CTRL-V tells Vim literally instead of acting on it" should be "The CTRL-V tells Vim to use the <Esc> literally instead of acting on it"
page 101, 40% down	error: ":1,/trouble/print" should be ".,/trouble/print".
Page 103, How to change last,first to first,last	confusing: This introduces a complex substitute command without explaining it. And the actual command is missing. My solution is to remove this section and figure 9.2, it's mentioned in another chapter. The -official- solution is to add a few sentences, which makes it only more confusing, in my opinion.
page 109, Microsoft Windows Mouse behavior	error: <S-Right Mouse> should be <Right Mouse>.
page 114, first example	error: The "ruler" has the 1, 2 and 3 above the "3", should be above the "0".
page 115, example near the top	error: The first block of text has the grey background everywhere. The blank space after the text should be white.
page 115, Text Formatting Command	mistake: This should be called "Text Formatting Commands", as there are several of them.
page 115, Text Formatting Command	error: the box with text doesn't show the effect of ":center" properly. The first two lines should have one or two spaces before them.
page 116, top	error: "gives results in the following": remove "gives".
page 126, just above How Vim Searches for Words	error: ...to the original entry " " should be "" (no space in the double quotes).

page 126, How Vim Searches for Words	error: item 3 in the list: "Other file in the currently loaded buffers" should be "Other files with loaded (hidden) buffers".
page 126, How Vim Searches for Words	error: "This is described in the section "Customization of Completion"" should be "This is described in the section "Automatic Completion Details""
page 127, Automatic Completion Details	confusing: ":set complete=key,key,key". Any number of arguments would be OK. Better change it to ":set complete=key,key,...".
page 134, example halfway	error: ":autocommand FileWritePre * :call DateInsert()<CR>" should be: ":autocommand FileWritePre * call DateInsert()". The extra ":" doesn't hurt but the trailing <CR> was wrong.
page 137, File Patterns	error: The comma item isn't part of a file pattern, it separates alternate patterns. Also, the [] item is missing.
page 137, Nesting	missing text: "If you read a file in response to a Syntax C, for example". Something is missing here. Probably it should be "If you read a C file in response to a FileChangedShell event, for example". The example of a FileChangedShell event with the pattern *.c is strange, perhaps it's also wrong.
page 144, first paragraph	error: "If you use a session file" should be "if you use a viminfo file".
page 149, Recover item	typo: "attempts to restart you session" should be "attempts to restart your session".
page 149, Quit item	confusing: "Forget about trying to change this file" should be "Do not edit this file".
page 149, List of items	missing: When the swap file is not being used by another process, there is also the alternative to delete the swap file.
page 151, top	confusing: "On Unix and Linux" should be "On most operating systems".
page 157, Window Size	mistake: "height" is bold and italic. It should only be italic.
Page 161, top	error: Step 4 is really an explanation of Step 3, not a separate step.
Page 164, Visual method	improvement: Remove "UNIX" from the fourth item.
Page 166, after first Note	omission: The text mentions "a number of tag-related commands" but does not give any reference. They are in chapter 7.

Page 166, Drawing Comment Boxes	minor: After "#b<Enter>" (in bold font) remove the bold semicolon.
Page 168, Changing Last, First to First, Last	error: In the title and in the fourth line, "First, Last" should not have the comma. The commands explain here remove the comma.
Page 169, first unlabeled figure	typo: there should be no space before the comma in the middle and before "/\2".
Page 169, second unlabeled figure	typo: The second explained item: "String matched by second \(\(" should be "String matched by second \(\)".
Page 169	layout: The two (unlabeled) figures should be swapped. Also, the "$" can be left out of the pattern.
Page 170, just above "Finding All Occurrences..."	omission: "This command will work in Microsoft Windows only if you have installed your own grep program."
page 170, Last but one paragraph	error: "...where a match is found. Position the file on the first matching line." should be: "...where a match is found and position the cursor on the first matching line."
page 186, There are "words"...	error: It mentions that "that-all" is two words, but it's three words, the dash also counts as a word.
page 186, Table of WORD related commands	missing: The <C-Left> and <C-Right> commands have an optional count as well.
page 187, figure 18.4	error: The "4+" should be "3+".
page 196, figure 18.18	confusing: The commands are mentioned with <Esc>, but you wouldn't use it like that.
page 196, Deleting text	confusing: This mentions "'register count x', while this is not used in the example. And other commands that can use a register, like "S" in the previous section, don't mention this. Then the X command is mentioned without register and count.
page 197, Figure 18.20	error: The "CTRL-A" on the right should be "5 CTRL-A".
page 198, figure 18.21	error: The first "123" should be "125". The "0x1E" should be "0x20".
page 198, below figure 18.21	error: If you want to recognize decimal and octal numbers" should be "If you only want to recognize decimal numbers". The example has two double quotes, they shouldn't be there. In the following sentence: "recognize just octal numbers" should be "recognize octal and decimal numbers".

page 198, figure 18.22	error: In the right upper box, "This is a test" should be "This isa test". In the right lower box, "This is a test with two lines" should be "This isa testwith two lines".
page 199, figure 18.24	error: The bottom line should not have a dot at the end.
page 200, figure 18.26	error: the "Rx" next to the arrow should be "grx".
page 200, Digraphs	error: This uses "c<BS>0" for the copyright sign. The 0 (zero) should be O (uppercase o). There are four zeros that need to be changed.
page 201, figure 18.27	error: On the lower left, "Now is the time" should be "NOW is the time". The command should be "3~" instead of "3~l". On the lower right, "Now is THE time" should be "Now is THE timE".
page 201, figure 18.28	error: On the lower left, "Now is the time" should be "NOW is the time". The cursor should be on the first character. On the lower right, "Now is THE time" should be "Now is THE timE". The cursor should be on the "o" in "Now". The command should be "g~fM" instead of "g~fm".
page 201, figure 18.29	error: the lower line should be "nOW is THE tIME".
page 201, figure 18.30	error: the command at the upper arrow should be "guu". The text right of this arrow should be all lowercase.
page 209, 210 and 211, figures 19.9 to 19.14	error: The caption of figure 19.9 says "search for /for", should be "search for for" (last for bold). In the other figures the slash also has to be removed.
page 211, figure 19.15	error: The caption says "search for /\a\a\a.", this should be "search for \a\a\a_" (remove a slash, add an underscore).
page 211, figure 19.14	There is a "1024" at the end of line 3 that should also be highlighted.
page 212, above the first note	error: "To match the constants" should be "to match anything but a vowel".
page 212, first note	misleading: This only mentions ^ as a special character. Other special characters are the backslash, dash and]. The special meaning of ^ can also be avoided by not putting it first.
page 212, below first note	stupid: The examples don't explain the character ranges. "one[\-]way" could better be written as "one-way". There is no reason to insert the backslash either. Also, "2\^4" can be written as "2^4" and has nothing to do with character ranges. "2[\^*]4" is easier written as "2[*^]4".
page 212, last paragraph	error: In the second sentence "a,\{,5}" should be "a\{,5}".

page 213, Grouping	unclear: In the second paragraph, the pattern "\(the\) \1" is used. Because of the line break the space goes unnoticed.
page 213, last line	typo: "the second \2" should be "the second to \2".
page 214, near the bottom	typesetting: There are three *, the first two look different from the last one. They should be the same character in the same font.
page 220, first sentence	error: "When inserting lines, p and P commands do not move the cursor" should be: "When inserting lines, the p and P commands move the cursor to the first non-blank character of the inserted text".
page 220, first paragraph	error: "The cursor is left at the end of the new text" should be "The cursor is left just after the end of the new text".
page 221, figure 20.2	error: The top arrow should be at the third line. The text has a double quote before the equal sign that shouldn't be there. A double quote at the end is missing for both callouts.
page 223, Special Registers	error: Just above the table: You can also use the command ""P.. This should be: You can also use the command "1P..
page 228, just above Advanced text entry	missing: Change "Place the F mark there because" to "Place the F mark there with the mF command. Because".
page 231, 60% down	mistake: "Pick a nice even number" should be "Pick a nice round number".
page 231, 232, The viminfo file	error: At first the options are in bold, later they are in single quotes, and then in bold and single quotes. They should all be the same. error: There is no \" option, this is the " option with a backslash added to avoid it being interpreted as the start of a comment.
page 234, last paragraph	unclear: What isn't clearly mentioned is that all these commands move within the current line.
page 235, below figure 20.10	inconsistent: "[count]" is used here for an optional count. It's not clear that the [and] are not to be typed. It should be count with italic style, like it is used elsewhere.
page 235, above figure 20.11	typo: "These command move" should be "These commands move".
page 240, first sentence	error: "CTRL-W k goes to the preceding one" should be "CTRL-W k goes to the window above".
page 243, figure 21.6	error: The left and right screens should be exchanged. The command is CTRL-W o, lowercase o. confusing: The left screen shows ":only", which is not actually used here.

page 243, figure 21.7	error: The left screen shows ":only", should be ":all" or nothing.
page 244, Split Search	typos: "position the cursor of the printf on Hello World" should be "position the cursor on the printf of Hello World". (change "on" to "of" and "of" to "on")
page 246, below first example	error: "will be positioned on lnum" should be "will be positioned on line lnum".
page 248, Sessions	error: The second paragraph mentions that marks and registers are stored, but that is not so.
page 252, figure 22.1	missing: After "Move using l to the end of line" should add "use j to go down".
page 253, Selecting Objects, second paragraph	error: "but also the space after it" should be "but also all the white space after it".
page 254, just above figure 22.5	error: the "count i(" item: "Like ab, except" should be "like a(, except".
page 254, figure 22.4	unclear: the curved lines should start at the first "t" of "test", not at the space before it. The space is not included.
page 254, table with text objects	missing: "count" is mentioned before each item, but not explained anywhere. In many items: "and the space after it", "and spaces after it" and "and the following space" should be "and all the white space after it". The "a<" item: "include" should be "including".
page 254, figure 22.5	error: in the left part, the highlighting should include the () braces.
page 259, figure 22.13	error: in the text between the screens, "gh i - start select" should be "gh start select". "<BS> - Delete text" should be "<BS> Delete text" and align the explanations like a table.
Page 262, Figure 23.2	error: "0CTRL-D" should be "^CTRL-D"
Page 266 and 267, Figure 23.3, 23.4 and 23.5	error: The arrows, indicating tabs, should all end in the same column. The ones indicating a full tab should all be the same length.
page 266, Smart Tabs	omission: The last sentence of the first paragraph: "is defined the 'shiftwidth' option" should be "is defined with the 'shiftwidth' option".
page 266, Smart Tabs	error: In the example with three lines, ":set smarttabs" should be ":set smarttab".

page 266, Figure 23.4	error: This is the same figure as 23.3. It should be different in that a Tab typed after the X never inserts spaces but a real Tab.
Page 268, The :retab Command	error: The third sentence must be deleted, it's a copy of the first one (except that "tab stops" turns into "tap stops").
Page 269, figure 23.6	error: last line on the left: "in this case, column 4" should be "in this case, column 5".
Page 273, figure 23.7	error: the last text on the left: "0} - } is the first" should be "0# - # is the first".
Page 275, Comparing Two Files	minor: "in the window you want to move and move it" is wrong. It probably should be "in the window you want to move".
Page 277, second :ptag example	minor: The variable is misspelled, "copy_p_date" should be "copy_p_data."
page 284, Locating include Files	error: "The]j command" should be "The]i command".
page 284, Multiple Error Lists	error: First sentence: "The :make file generates" should be "The :make command generates".
page 289, figure 23.20	error: The line in the middle "$ gvim Makefile" should be: :set switchbuf=split :make
page 291, last paragraph.	error: "bb is the amount of blue, and yy is the amount of yellow" should be "gg is the amount of green and bb is the amount of blue".
page 292, Syntax Elements	error: The table starts with a line of bold words. These should not be bold.
page 300, last paragraph	mistake: This paragraph doesn't belong here at all. There is no relation between mapping and the 'suffixes' option.
page 303, introduction	typo: "Being expert in the command-mode command means..." leave out "command".
page 310, 25% down	error: "change Test to b*Test when" should be "change Test to bTest when".
page 314, Override Option (!)	typo: "within comments found as well" should be "within comments are found as well".
page 315, Current File	error: third paragraph: "Now you want shorten the" should be "Now you want to shorten the".
page 318, Undo/Redo	mistake: The text starting with "To mark the beginning" is a separate section about marks. It belongs as a separate section below "Miscellaneous Commands".

page 324, Window size and position	error: the format for the geometry should read: -geometry *width* x *height* + *x_offset* + *y_offset*
page 326, Icon item	typo: "the editor in minimized" should be "the editor is minimized".
page 327, Note in the middle	typo: "givmrc" should be "gvimrc".
page 330, first example	error: ":set title=85" should be ":set titlelen=85".
page 333, Select Mode	error: 'mousemode' should be 'mousemodel'
page 335, Toolbar Icons	error: "name.xpn on UNIX" should be "name.xpm on UNIX".
page 339, Renaming menu items	missing: Change "The command" to "The following command".
page 341, bottom	typo: "Ths" should be "This".
page 345, Changing the appearance of the cursor	confusing: Although this section appears just after the secion on Microsoft Windows specific commands, it can be used with all systems.
page 346, 'guicursor' table	error: In the third column a few items are wrong. In the second table row, "35% high" must be "35% wide". In the fourth row, "35% high" must be "25% wide". In the fifth row, "20% width" should be "20% high".
page 350, Special variable names table	spacing: The second item has been split in two parts in the left column. The item should be for "Initial uppercase letter and lowercase letter somewhere inside".
page 351, last of the "some examples"	The line starting with Note: should have a line break after "associated with", so that the double quote is at the start of the next line.
page 353, Entering Filenames	confusing: What should be mentioned here is that these items can't be used in an expression directly. They are to be used after a command like ":edit" or as an argument to the expand() function. This section doesn't really belong here.
page 354, <amatch> item near the top	error: <abuf> should be <afile>

page 354, paragraph between the tables	error: <cfile:p> should be <cfile>:p
page 354, :r item	mistake: using :r on "../path/test.c" yields "../path/test" and not "test."
Page 356, last example	error: The "<:" should be "<=". It's not a good example, the loop will be infinite as soon as the first "if" statement is true.
page 359, example halfway	error: The example doesn't work, because "a:index" is invalid. The line should be replaced with: execute 'echo "Arg" index "is " a:' . index
page 360	error: User commands cannot contain an underscore character. All the commands "Delete_first" should be changed to "DeleteFirst", "Delete_one" should be "DeleteFirst" as well.
page 360, example near the bottom	error: The text mentions Delete_one while the example uses Delete_first, they should be the same "DeleteFirst". The "-nargs=0" argument must be placed before the command name: ":command -nargs=0 DeleteFirst 1delete".
page 361, "-range=%" item	typo: "while" should be "whole".
page 361, near the bottom	typo: "Execute" should be "executes".
page 364, col() function	typo: "x should be 'x.
page 367, first line	typo: "[{string}" should be "{string}".
page 377, virtcol() function	typo: The double quotes Parameter: explanation should be removed.
page 382, Automatically Setting Options in a File	error: The format starting with "vim: set" should have a colon at the end.
page 382, Automatically Setting Options in a File	error: In the format at the bottom, change "Vim" to "vim" and remove the dot before "option-command".
page 383, top	error: The second 'modeline' must be 'modelines'.
page 383, halfway	error: "expandtabs" must be "expandtab".
page 383, Above "Local .vimrc Files"	error: The example should read: /* vim: set cindent shiftwidth=4 smarttab: */ Just above this change "the top of bottom" to "the top or bottom".

page 384, halfway	error: "the command mapped to ALT-F" should read "the command that ALT-F is mapped to".
page 385, 80% down	error: "use the 'timeoutlen' options:" should be "use the 'ttimeoutlen' option:"
page 386, bottom	font: in "an f appears" the f has a wrong font, it should be italic.
page 390, % item	error: % should %%
page 400, Lazy redraw	opinion: The 'lazyredraw' option is not obsolete. Setting it avoids the screen flashing when executing a macro, but you can't see what it is doing.
page 401, compatibility	misplaced: 'tildeop' has nothing to do with Vi compatibility.
page 404, c_no_ansi	error: the size_t and INT_MIN in the first two lines should be highlighted, not in the last two lines.
page 404, c_ansi_typedefs	error: the size_t in the first line should not be highlighted.
page 405, table	error: the grey background is misaligned, it should be one and a half line lower. In the c_ansi_constants item, the ":set" command should be ":let".
page 414, just above Defining Matches	mistake: "The x command allows" should be "The x language allows".
page 415, Nested Regions	confusing: Although the section header says "Regions", the first example uses a match item. "Nested Items" would be a better title.
page 415, Nested Regions	error: In the example ":syntax keyword xTodo contained" should be ":syntax keyword xTodo TODO contained".
page 416, 30% down	error: "preprocessor syntax contains." should be "preprocessor syntax continues.".
page 417, Transparent matches	typo: "spcify" should be "specify".
page 419, Clusters	typo: The first line of the example: "contains+xNumber,xIdent" should be "contains=xNumber,xIdent".
page 420, Including Other Syntax Files	typo: The example halfway the page defines the "@Pod" cluster and then uses "contains=@POD". That should be "contains=@Pod".
page 421, Example halfway	typo: "minline=150" should be "minlines=150".

page 428, unpacking the sources	error: "gzip -u -d" should be 'gzip -d". There is no "-u" flag for gzip.
page 429, Installing on Microsoft Windows	confusing: Before unzipping the archives, there is no need to create a "c:\Vim" directory, unzip will do this for you. The names given for the archives to unzip are with the directory as they appear on the ftp site. After downloading they could be anywhere. This text is unclear, you might want to use other installation instructions.
page 430, top	error: "EXECUTAB.S" should be "executables.".
page 430, Note at the bottom	error: "the non-BUI version" should be "the non-GUI version".
page 431, I Am Using RedHat Linux.	error: "vim-enchanced-version.rpm" should be "vim-enhanced-version.rpm".
page 511, 'magic'	omission: Add a reference to page 34.

VIM

VIM, which stands for Vi IMproved, is one of the two most powerful, and popular, free multi-platform [text editor]?s for skilled users. VIM's multi-mode full screen editing paradigm is a conceptual outgrowth of the original vi editor implementation that, since since the early 1980s, has been distributed as an essential part of every operating system that is derived from Unix. Since its inception in the early 1990s VIM has successfully grown to accommodate a very powerful array of features, both original and derived from other free and commercial text editors. This success is an outgrowth of the development process which has been driven by the public publication of the VIM source code combined with the constant interaction between users, developers, and the original creator of VIM, [Bram Moolenaar]?.

Please see :

- /Feature list
- [VIM home page]
- [Latest news]
- [VIM FAQ]
- [Downloading]
- [Mailing lists]
- [VIM on Sourceforge]

/Talk

Search:

"Vim nedir?"
Alti kilobytelik bir aciklama

Vim ("Vi Gelistirilmis") bir vi "kopyasidir", yani metin editoru vi benzeri bir programdir.

Vim metin modundaki her terminalde calisir, ayrica menu ve fare icin destegi olan bir grafik kullanici arayuzune sahiptir.

Varlik:
Vim bir cok degisik platform icin varlik gosterir ve vi'da bulunmayan bir cok ozellik icerir. (bkz. http://www.vim.org/doc/vi.diff.txt). Vim hemen hemen tum vi komutlari ile uyumludur - vi hatalari haric ;)

Isletim Sistemleri:
Vim bir cok isletim sistemi uzerinde calismaktadir:AmigaOS, Atari MiNT, BeOS, DOS, MacOS, NextStep, OS/2, OSF, RiscOS, SGI, UNIX, VMS, WIn16 + Win32 (Windows95/98/00/NT) - ve ozellikle FreeBSD ve Linux

Telif Hakki:
Vim'in telif hakki, kodun bas yazari ve bakimcisi Bram Moolenar'in <bram@vim.org> elinde bulunmkatadir. Vim "yardim-ware" tipinde bir programdir ve kullanicilarinin Uganda'daki yetimler yararina bagista bulunmalari tavsiye olunur.

Kaynak:
Vim'in kod kaynagi Acik Kaynak formunda olup, herkes gelistirmeye yardim icin davetlidir.

=== Ozellikler

Yeni basliyanlarin Editoru - Kullanisli
Vim, genis online yardimi, "undo(geri cevir)" ve "redo(tekrar yap)" komutlari (hatalar umrunuzda olmasin - sadece undo+redo'yu kullanmaniz yeterli!), fare, degistirilebilir ikon ve menu destegi (GUI) sayesinde yeni baslayan kullanicilara Vi'dan cok daha kolay gelicektir.

Karakter Kodlari ve Terminaller:
Vim iso-latin1 karakter setini ve termcap'i desteklemektedir. (Orjinal Vim'in bununla ilgili sorunlari var.)

Karakterler ve Yabanci Diller:
Vim sagdan sola yazma (ornek: Arapca, Farsca ve Ibranice) ve coklu byte ozelligi tasiyan metinleri, yani grafiksel karakterleri birden fazla byte ile temsil edilen Cince, Japonca, Korece benzeri dilleri desteklemektedir. (Teknik olarak konusacak olursak VIM UTF-s ve Unicode Metinleri desteklemektedir.)

Metin Formatlama ve Gorsel Mod:
Vim ile metine mudahale etmeden, mesela kopyalamadan, silmeden, degistirmeden, saga ve sola kaydirmadan, buyuk-kucuk harf degisimden veya ,ice alinmis metinlerde dahil olmak uzere, formati degistirmeden once metni gorsel olarak secebilirsiniz. Ayrica Vim dikdortgensel metin bloklari secimi ve bunlar uzerinde islemlere olanak vermektedir.

Komut Tamamlama:
Vim girdiyi - komutlar, dosya isimleri ve kelimeleri- tamamlayan komutlara sahip.

Otomatik Komutlar:

Vim otomatik calistirma icin kullanilan "otokomutlara" da sahip.
(mesela sikistirilmis dosyalarin otomatik olarak acilmasi)

Digraf Girdi:
Vim, iki karakteri kombine olarak kullanarak ozel karakter girilmesine
ve yeni kombinasyonlar tanimlanmasina izin verir. Ornegin [c,]
kombinasyonu ç harfinin basilmasini sagliyacaktir.

Dosya formati tanima ve cevirme:
Vim dosya tiplerini (DOS, Mac, Unix) otomatik olarak tanir ve bu
dosyalari baska bir formatta kaydetmenize izin verir - Windows'ta artik
unix2dos kullanmaya gerek yok.

Gecmis Listesi:
Vim , onceki komut ve aramalara erisim saglamak veya onlari degistirmek
icin, girilen komutlarin ve yapilan aramalarin listesini "gecmis
liste"sinde saklamaktadir.

Makro kayitlama:
Vim tekrarlanan isleri yeniden oynatabilmek icin yazdiklarinizi
kaydetmeye izin verir.

Bellek Limitleri:
Vim satir uzunlugu ve tampon bolge hacmi acisindan orjinal Vi'a oranla
cok daha esnektir.

Birden cok tampon bolge ve bolunmus ekran:
Vim birden cok tampon bolgenin ayni anda degistirlmesine olanak verir
ve vim penceresi (hem yatay hem dikey olmak uzere) altpencerelere
bolunebilir, boylece birden cok dosya veya bir dosyanin degisik
parcalari ayni anda gorulebilir.

Komutlar icin sayisal on ekler:
Vim, Vi'dan daha cok komut icin saysisal on ek destegi saglamaktadir.

Yurutme zamani Dosyalari (Yardim ve Sozdizim Dosyalari):
[Programin calisma esnasinda kullanilan ek dosyalar - bunlara derlenicek ve
baglanicak kod dahil degildir]
Vim 5.7'nin beraberinde komutlar, secenekler ve duzenleme uzerine ipuclari
hakkinda 70 adet yardim dosyasi gelmektedir. (Vim-6.0x [010311]: 85 dosya,
2796K'lik yazi). Bazi dosyalar vim'in degisik isletim sistemleri ustunde
kullanimina mahsustur.

Betikleme:
Vim kolay genisletme amacli betikleme dili ile birlikte gelmektedir.

Goreli Konum ile Arama:
Vim arama komutlari icin "goreli konum" (ofset) olanagi saglar, boylece
imgeci yaziyi bulduğunuz yerin sonrasina koyabilirsiniz.

Oturum kurtarma:
Vim duzenleme oturumunuzu bir dosyaya (viminfo) kaydetme imkani saglar,
boylece bir sonraki oturumda tampon listesine, dosya imlerine, yazmaclara
(registers), komut ve arama gecmisine kolay ulasim saglar.

Sekme genisletme:
Vim sekmelerin, ara bosluklari olan metinlerde genisletilmesine olanak
saglar (expandtab, :retab)

Imleme (tag) Sistemi:

Vim dosyalari icinde metin (yazi, isim, vb) arama, "tags" indeksi ve imleme yigini komutlari sayesinde cok daha kolaylasmistir.

Metin Objeleri:
Vim'in paragraf, cumle ve kelime (bosluklarla cevrilmis yada cevrilmemis) gibi metin objeleri hakkinda daha cok bilgisi vardir ve bu objelerin tanimini degistirmeye imkan saglar.

Bicim Renklendirme:
Vim metni yazildigi programlama diline bagimli olarak metni renklendirir. Eger isterseniz doyalarinizin programlama dillerini kendinize gore degistirebilrisiniz.

Vim C, programlama dilleri (Ada, C, C++, , Fortran,
Haskell, Java, Lisp, Modula, Pascal, Prolog, Python,
Scheme, Smalltalk, SQL, Verilog, VisualBasic), matematik programlari (Maple, Matlab, Mathematica, SAS), bicimleme metinleri (markup) (DocBook,HTML, LaTeX, PostScript, SGML-LinuxDoc, TeX, WML, XML), kurulum programlarinin dosyalari (4DOS, Apache,
autoconfig, BibTeX, CSS, CVS, elm, IDL, LILO, pine, procmail, samba, slrn), kabuk komutlari ve kurumlari, (shells: sh, bash, csh, ksh, zsh), betikleme dilleri (awk, Perl, sed, yacc), sistem dosyalari ve elbette vim ile vim'in yardimci dosyalari icin 200'un uzerinde renklendirme sozdizimi dosyasi ile birlikte gelmektedir.

Ozel Kod:
Vim Perl, Tcl ve Python ile entegrasyon secenegine sahiptir. Vim, Windows altinda OLE otomasyon sunucusu olarak calisabilir. Vim ayrica X-windows destegi ile yuklenebilir. Bu islem fare icin destek ve ayarlanabilir menu olanagi saglar.
Ve daha nicesi!

=== Baglantilar:
WWW'de vim anasayfasi: http://www.vim.org

Daha detayli aciklama icin vim'in ozellikleri sayfasini ziyaret edin.
http://www.vim.org/why.html

=== Yazan ve Ceviren:
Yazan: Sven Guckes guckes@vim.org
Son Guncelleme: Mon Mar 12 07:00:00 MET 2001

Ceviren: Kutlay Topatan kutlay@hotmail.com
Son Guncelleme :6 Haziran 2001 Pazartesi

"What is Vim?"
An explanation in six kilobytes.

Vim ("Vi IMproved") is a "vi clone", ie
a program similar to the text editor "vi".

Vim works in textmode in every terminal,
but it also has a graphic user interface,
ie menus and support for the mouse.

Availability:
Vim is available for many platforms and has many added features
compared to Vi. (see http://www.vim.org/doc/vi.diff.txt)
Vim is compatible to almost all Vi commands - except Vi's bugs. ;-)

Operating Systems:
Vim is available for many systems: AmigaOS, Atari MiNT, BeOS, DOS,
MacOS, NextStep, OS/2, OSF, RiscOS, SGI, UNIX, VMS, Win16 + Win32
(Windows95/98/00/NT) - and especially FreeBSD and Linux. :-)

Copyright:
The copyright is in the hands of the main author
and maintainer, Bram Moolenaar <bram@vim.org>.
Vim is "charity-ware", ie you are encouraged to make
a donation to orphans in Uganda (see ":help uganda").

Source:
Vim is OpenSource and everybody is welcome to help improve it!

=== Features

Beginner's Editor - User Friendly:
Vim is much easier for beginners than Vi because of extensive Online Help,
"undo" and "redo" commands (never mind mistakes - just use undo+redo!),
support for the mouse and configurable icons and menus (GUI).

Character codes and Terminals:
Vim has support for the iso-latin1 character set and for termcap.
(Vanilla Vi has problems with this.)

Characters and Languages:
Vim supports for right-to-left editing (eg with Arabian, Farsi, Hebrew),
and multi-byte texts, ie languages with graphical characters represented
by more than one "byte", such as Chinese, Japanese, Korean (Hangul),
(Technically speaking, Vim supports text in UTF-8 and Unicode.)

Text Formatting and Visual Mode:
With Vim you can select text "visually" (with highlighting) before you
"operate" on it, eg copy, delete, substitute, shift left or right,
change case of letters or format the text incl preserving indented text.
Vim allows selection and operations on rectangular text blocks, too.

Completion Commands:
Vim has commands which complete your input -
either with commands, filenames, and words.

Automatic Commands:
Vim also has "autocommands" for automatic execution of commands
(eg automatic uncompression of compressed files).

Digraph Input:
Vim allows you to enter special characters by a combination
of two characters (eg the combination of " and a yields an ä) -
and allows you to define other combinations, too.

Fileformat detection and conversion:
Vim automatically recognizes the type of files (DOS, Mac, Unix)
and also lets you save them in any other format -
no need for unix2dos on Windows any more!

History:
Vim has a "history" for commands and searches, so you can
recall previous commands or search pattern to edit them.

Macro Recording:
Vim allows to "record" your editing for replaying for repetitive tasks.

Memory Limits:
Vim has much higher memory limits for line length
and buffer sizes than vanilla Vi.

Multiple Buffers and Split Screen:
Vim allows editing of multiple buffers and you can split the
screen into many subwindows (both horizontally and vertically),
so you can see many files or many parts of some files.

Number Prefix to commands:
Vim allows a number prefix for more
commands than with Vi (eg for "put").

Runtime Files (Helpfiles and Syntax Files):
[Additional files which are used when the program runs -
 but these not contain code which has to be compiled and linked.]
Vim-5.7 comes with 70 help files (about 2080K of text) on
commands, options, with tips on configuration and editing.
(Vim-6.0x [010311]: 85 files, ca 2796K of text). Some files are
specific to the use of Vim on each operating system. [010311]

Scripting:
Vim has a built-in scripting language for easy extension.

Search Offset:
Vim allows offsets for search commands, so you
place the cursor *after* the found text.

Session Recovery:
Vim allows to store information of an editing session into a file ("viminfo")
which allows them for being used with the next editing session, eg
buffer list, file marks, registers, command and search history.

Tab expansion:
Vim can expand tabs within the text with spaces (expandtab, :retab).

Tag system:
Vim offers to find text in files by using an index with "tags"
together with many tag stack commands.

Text Objects:
Vim knows more text objects (paragraphs, sentences, words and WORDS -
all with and without surrounding whitespace) and allows to configure

the definition for these objects.

Syntax Coloring:
Vim shows text in color - according to its "(programming) language".
You can define the "language" ("syntax") of the files yourself.

Vim comes with 200+ syntax files for colorizing text in
common programming languages (Ada, C, C++, Eiffel, Fortran,
Haskell, Java, Lisp, Modula, Pascal, Prolog, Python,
Scheme, Smalltalk, SQL, Verilog, VisualBasic),
math programs (Maple, Matlab, Mathematica, SAS), markup text
(DocBook, HTML, LaTeX, PostScript, SGML-LinuxDoc, TeX, WML, XML),
program output (diff, man), setup files of programs (4DOS, Apache,
autoconfig, BibTeX, CSS, CVS, elm, IDL, LILO, pine, procmail, samba, slrn),
shell scripts and setups (shells: sh, bash, csh, ksh, zsh),
script languages (awk, Perl, sed, yacc)
system files (printcap, .Xdefaults) and
of course for Vim and its helptexts.

Special code:
Vim has optional integration with Perl, Tcl and Python.
Vim can act as an OLE automation server under Windows.
Vim can also be installed with code for X-windows,
adding configurable menus and support for the mouse.
And more. Lots more!

=== Links

Vim's HomePage in the WWW:
  http://www.vim.org/

For a more elaborate description
of Vim's features see the page
  http://www.vim.org/why.html

=== Author and Translator

Original-by: Sven Guckes guckes@vim.org
Last update: Mon Mar 12 07:00:00 MET 2001

Translated-by: Your Name <address>
Last update: <date>

vim: tw=70

Algorithms in Africa

Maybe the rush to market for spreading internet access across the globe isn't in anyone's best interest--a report from the front.

by Wayne Marshall

Eleven years ago I installed a computer system at a vocational training and development center in Tutume, Botswana. Tutume is a rural village on the northeastern edge of the Kgalagadi desert in southern Africa. The computer was intended to help this organization, known as Tutume Brigades, catch up on its bookkeeping for several business units crucial to the local economy. Businesses included a brick-making unit, carpentry workshop, auto repair garage, sorghum mill, school uniform production unit, tractor hire and vegetable garden. For the local village and the surrounding catchment era, the Brigades were literally the only game in the bush for commodities, trade skills, training and employment opportunities.

When I arrived in Tutume, I was a pure novice in the field of foreign assistance. I was also a mid-career financial professional, with several years of experience in nonprofit and health-care management in the United States. And like most aid workers new on the ground in Africa, I knew what was best. In my assessment of the center, I believed a computer was essential to get a handle on the Brigades' financial position, which otherwise consisted of eight separate sets of badly maintained manual ledgers, over nine months in arrears. Except for the bank statements of eight separate checking accounts (and even the bank statements proved unreliable), we had no way of knowing if the center had any money. Every time we had to make payroll or buy another truckload of cement, we were in the heart of fiscal darkness.

Over the course of the next several months, I proceeded to computerize the records and train local staff in basic operation of the system. By the end of the first year, the financial records of the center were timely and accurate. Moreover, other staff members were beginning to use the computer for tasks such as word processing and spreadsheets. Many of these employees had never even used a typewriter before.

If I were to tell no more of this story and fade here to one of the glorious Kgalagadi sunsets, this might be called a win. Although set in the predawn (and pre-Linux) history of the Internet era, today this would be described as a small success story of ``bridging the digital divide'' in

Africa--like I was a regular Albert Schweitzer of the Information Age or something.

But the truth is not so simple, and the issues of foreign assistance are not so trivial. The fact is, I am not proud of this story. Because as my time in Tutume went on, I realized I had blundered badly, to the point of putting the Brigades in serious jeopardy. I began to ask myself such basic questions as: What would happen to the computer after I left? Was the staff fully capable of operating the system independently? Would backups be maintained and performed rigorously? Were skills sufficient to troubleshoot problems and reinstall the system if necessary? If the equipment failed or was stolen, could the center afford to replace it? And what would the center do when the staff I had trained for so long were lured away by more lucrative jobs in the big city?

These questions all led to the same answer: the Brigades would be left in even worse shape than I found them. Rather than gaining empowerment, independence and enablement, they would more than likely be left powerless, dependent and possibly ruined. And all because of my own cultural myopia, despite my good intentions.

It is axiomatic in the field of foreign assistance that the aid program will take credit for the successes, while failures are blamed on the host country. The psychology of failure can then be even more severe and long-lasting than the loss of the project. While I was working in Tutume, for example, a friend of mine was working in the village of Lobatse in southern Botswana. Seven years earlier, an aid organization from northern Europe had decided a wool sweater factory would be just the ticket for the economic development of the village. Of course, northern Europeans are fond of nice wool sweaters and very likely have great need for them, particularly in the colder climes of northern Europe. The market for wool sweaters is less extensive in the sweltering and sparsely populated Kgalagadi desert, however. After seven years of subsidizing the losses of the operation, the aid organization finally decided it was never going to be sustainable, and they pulled the plug on the effort. My friend's unenviable assignment was to put all the women out of work, sell the facility and liquidate the equipment. It was hard for many of the women not to feel that the fault was somehow their own.

Fortunately for Brigades in Tutume, such failure was averted. As the story there continues, once I realized the risks, I spent the next several months converting the accounting system back to manual ledgers, hiring and training additional staff in bookkeeping procedures and enabling them to use the computer primarily as a support system, rather than as the central financial database.

But what do these stories from Tutume and Lobatse have to do with Linux and emerging markets? The rest of this article will consider that question.

The Digital Divide

Nine years have passed since I left Botswana. To put the times into perspective, the first thing I bought when I got back to the US was a fax modem, the cheapest, fastest solution to stay connected with the contacts I had made abroad. My modem then was 2,400 baud. I tried out CompuServe and decided on Delphi, and the buzz was just starting about something called PPP.

During the next several years I was in and out of Africa, became a Linux user in 1995, began installing Linux in nonprofit organizations in 1997, spent a year and Y2K transition in the former soviet state of Ukraine and came to the West African country of Guinea in May 2000. At some point during this period the digital divide was invented.

Actually, the digital divide seems to have its origins in a 1995 report from the US Department of Commerce, whose National Telecommunications and Information Administration (NITA) released the first paper in a series titled ``Falling through the Net''. This report analyzed telecommunication access by geographic and demographic patterns throughout the United States. One of the conclusions of the report was the gap between the ``information rich'' and the ``information poor'' had widened.

In the later years of the Clinton administration, the digital divide broadened beyond US borders to encompass the globe. The issue gained considerable publicity after a G8 economic summit meeting in 1999, where the most powerful nations on earth decided that the growing gap in information technology was one of the most serious problems facing development in the Third World.

Now, as I write this, bridging the digital divide has become one of the hottest trends in foreign assistance, and many aid organizations and corporate philanthropists have found publicity for their efforts. Simplistically, it seems, the gap in information technology has now come to be identified with access to the Internet. Thus, we have such programs as the USAID-funded Leland Initiative, designed to bring internet access to Africa; the Peace Corps announcing an information technology initiative in partnership with AOL; and a recently formed organization called Geekcorps sending its second group of volunteers on three-month stints designing web sites in Accra, the capital of Ghana in West Africa [see *LJ* April 2001 for more on the Geekcorps]. Naturally, the high-profile publicity given this issue has created an opportunity for many international aid organizations to develop projects and funding appeals for serving the digitally needy.

The New Tech Testament

Delivering the miracle of the Internet is the new zeal of the high-tech missionary. In what seems to be a rush to market--bringing the Internet to the developing world--sometimes projects are announced with only naïve regard to the technical issues and without full consideration of whether such projects are viable, appropriate, relevant and sustainable. Thus, one hears of a women's cooperative in Central America marketing their handcrafts over the Web; advocates describe the potential of ``telemedicine'' for delivering virtual health care to isolated areas; and the US State Department Global Technology Corps proclaims, ``We have seen farmers in Mexico using [the Internet] to check weather conditions and crop prices.''

Where once Norwegians may have seen wool sweaters, the tech visionary now sees web browsers.

At the extreme, the new economy proselyte promotes the Internet as the solution for everything from education and health care to pollution, inequality and world peace. As though everyone who has access will be able to browse their way to nirvana, as though the path to heaven is

paved with bandwidth. The satellite dish is the new icon of the digital evangelist, replacing the holy cross.

One of the implicit beliefs of this testament is that information, in and of itself, is sufficient to promote economy, remedy problems and narrow inequities. A corollary implication, the message from one side of the divide to the other, is that we have information and you don't, that our information is good and yours is useless. This is the lesson CNN preaches to its international audience when it tells us, ``The human without information is nothing."

It should be clear that in this form, divide rhetoric is simply new raiment for the familiar old taxonomies of prejudice that have long sought to divide the world between believers and heathens, the enlightened and the savage. From a historical perspective, rather than helping, these kinds of belief systems have generally been devastating to their targets.

More importantly, the belief in the sufficiency of information and information technology is simply wrong. Information alone doesn't help people. If only this were true, doctors would be made from medical textbooks and entrepreneurs would be born from accounting manuals.

In fact, the developing world is littered with unused X-ray equipment, broken-down tractors and empty schoolrooms contributed over the years by well-intentioned and simpleminded donors. These resources are made useless not from missing user manuals or lack of web access, but by the lack of trained technicians, mechanics and teachers.

In short, what empowers people are skills.

Even in the US, this kind of awareness is emerging. In ``How Does the Empty Glass Fill? A Modern Philosophy of the Digital Divide" (Educause Review, Nov/Dec 2000), Solveig Singleton and Lucas Mast write: ``From the standpoint of higher education, students who leave high school without exposure to digital learning tools such as the Internet will prove a much less serious problem than students who leave high school with inadequate reading or math skills."

And the leading journal of free-market capitalism, the *Economist*, recently observed:

> The poor are not shunning the Internet because they cannot afford it: the problem is that they lack the skills to exploit it effectively. So it is difficult to see how connecting the poor to the Internet will improve their finances. It would make more sense to aim for universal literacy than universal Internet access.

It may be that, with the recent outbreak of dot-com bankruptcy and declines in the stock market, the tenets of the digital religion could be losing their currency. At a time when the mega-billion, IPO-funded ebiz stars like Amazon and Yahoo are having a tough go across the US and Europe, it's hard not to wonder how the promises of e-commerce could possibly prove viable and sustainable elsewhere, particularly in places where there aren't even good banking and credit systems. And for someone like me who has lived several years of the past decade in both rural and urban parts of the developing world--where most of the population still cook with firewood and carry water in buckets--the practical value of focusing foreign assistance on IT projects would seem negligible, if not ludicrous entirely. Given the more serious fundamental issues facing developing nations--health care (AIDS, TB and malaria), nutrition, sanitation, education,

poverty, pollution and political corruption--providing the means to surf the Web should probably fall fairly low on any reasonable scale of human priorities.

So is there any way to make a difference, a real difference that improves people's lives? Is there any role for Linux and open-source advocacy in emerging markets? Are there ways of using technology for solving human problems in places like Africa, without trying to sell wool sweaters in the desert? I wouldn't be writing this article if there weren't.

Algorithms in Africa

When it comes to Africa, the so-called digital divide is just a divide; there isn't anything especially digital about it. The divide is geographic, because Africa is a long way away, and cultural, because the traditions and histories of Africans developed independently from those of Europeans and Americans. Almost incidentally the divide is economic, from the standpoint of cash resources and differing perceptions of wealth, though the natural resources of this continent are vast. The divide ends up being mostly one of ignorance, and this gap is at its widest in America.

Americans in general know very little about Africa, and what little they do know or think they know is usually prejudiced and fallacious. If I were to know the state of Florida only from news reports, I would think it was a large mobile-home park of fat pink people constantly flattened by hurricanes. Similarly, most Americans probably only know Africa as a disaster zone of epidemic, starvation and genocide. The principal media image Americans hold of African assistance is usually the one of the brave young (white) woman, a nurse or volunteer, holding a helpless black infant, center stage among a group of grateful and admiring Africans in the background.

Of course Africa is nothing like this image at all, and the first step in crossing the divide here is to banish these offensive stereotypes and learn all one can about what Africa is really like. It would be a disservice to the many peoples of the continent to generalize and describe the essence of Africa as though it were a single place. But I would just like to say: Africa is such a joy! Whenever I am in the streets of Conakry or an upcountry village, I am overwhelmed with the pure bandwidth of humanity, of color and vitality and life. So much more than can ever be expressed on even your largest CRT, with even the fastest DSL connection; Africa is the ultimate realization of broadband in culture and diversity, natural and human content. Maybe a virtual, flat-screened reality over the Internet is meaningful in the pitifully dreary cubicle of the US office worker, but Africa is all about face time in real time.

Open-source advocates can be sure that Africans get community; Africans get bazaar. These are concepts intrinsic to the cultures and traditions throughout the continent, where African societies had mastered networking long before the invention of the RJ45 jack. Africans have historically been quite receptive, often at their ultimate peril, to ideas and innovations flowing between cultures and brought in by outsiders. And in general Africa has been early and enthusiastic about adopting new communication technologies, particularly when they are practical and affordable. So in Botswana I was astonished at the number of fax machines per capita ten years ago, and now find a thriving trade in cell phones, both legitimate and black

market, in Guinea. On a recent visit to a mosque in the interior of the country, a wizened old muezzin took me up into the minaret specifically to show me their solar-powered amplifier and loudspeaker system, used to call the village to prayers.

As one learns to develop an appreciation of what Africa is really like, it will then help if one can develop a sensitivity to the pitfalls of foreign aid and the havoc such programs have brought to this continent. The subject of other narrations, it is sufficient to observe here that the official assistance programs of foreign governments are usually a foul brew of political hegemony, economic imperialism, cultural ethnocentrism, personal avarice and, too rarely, genuine altruism. Too often the implementation of foreign aid is all about developing market share and spheres of influence, instead of improving lives. Proponents of foreign assistance may even argue that these are synonymous, as though markets for American soft drinks, snack foods and beauty products result in happiness and prosperity for the consumer. The sad fact is, whether intentional or merely consequential, foreign assistance has often had devastating effects on communities, local markets, traditional cultures and environmental conditions throughout Africa.

Finally, it is helpful to bring an honest perspective of one's own history and culture into focus. For example, the United States represents less than 6% of the world's total population and has existed for less than a blink of an eye in the span of all human history. So, what makes us think we've got it right? What evidence is there to suggest this brief record is proof that our way of life and cultural adaptations will be viable in the long run?

For example, it may be surprising to learn that, due to the predations of infectious illness, urban population levels were not even sustainable until about 100 years ago and required steady migration from rural areas. And it was less than 90 years ago, Gina Kolata writes in *Flu,* when ``*Ladies Home Journal* proudly declared that the parlor, where the dead had been laid out for viewing, was now to be called the living room, a room for the living, not the dead."

Shortly after this proclamation, a global flu of epidemic proportion--the origin of which is still not understood--killed 1.5 million Americans and 40 million worldwide. This was not in the murky history of the Dark Ages; this was 1918. Today, with the modern plague of HIV/AIDS, the re-emergence of tuberculosis and new mysteries like the relationship of human CJD to Mad Cow Disease, will our mastery of medicine prove all that enduring, even for the world's most fortunate few?

In any case, those who would help others should at least try to learn from past failures and have the humility to ask if the modern model of urbanization, congestion, resource utilization and environmental depletion are sustainable, even desirable, let alone worthy of export to others in the world.

Then we may be able to accept that the Internet may not be the solution to all problems of humankind and have the patience to realize that working through the major challenges in Africa will take time and understanding measured in generations. Now it becomes clear that Linux and open-source developers are helping Africa best by what they have been doing already. People who are programming and installing the world-class, free software at the soul of internet technology are helping others around the world in profound and important ways, no matter what

license they are using. GNU and open-source software are the perfect fit for the emerging nations of Africa--as for the rest of the world--not only for the superior technical quality of these systems, but for the values embodied in their development.

The mere existence of Linux and open-source systems give people the chance to use these powerful technologies for low-cost, grassroots level applications, an opportunity not possible just ten years ago. The pages of this magazine have described many of these self-directed success stories, everywhere from Mexico to Pakistan, where Linux solutions enabled people to make the difference. Such examples are to be found among African communities as well, from South Africa to Kenya to Nigeria. And Africans like Katim Touray are using Linux servers to connect other Africans in dialogue around the world.

Beyond the software itself, though, it is the culture of Linux and Open Source communities that provides the model for meaningful outcomes. This is the culture of sharing and empowerment, of the thousands of Linux users' groups throughout the world, of the Linux Documentation Project and the general willingness of one user to selflessly help another. Participating as a Linux user is all about developing crucial skills and passing them on. Often users' groups hold regular installation clinics, giving new users personal, one-on-one support from an enthusiastic peer. And these users' groups are often active in other community projects, such as helping schools install servers and network connectivity, while transferring the skills necessary to maintain them. Each of these connections is essentially more human than technical, linking people together more than their machines, and can lead anywhere. Each of these personal connections sows the seeds of others, and the spread of the Linux bloom is now reaching to every corner of the earth. For example, even though the use of internet technology in Guinea is nascent, Linux certainly preceded my own arrival here. One finds Linux books in French in bookstores and Guineans eager to learn more about this ``true'' operating system.

And there are other instances of Linux and open source helping to solve problems in Africa. One of the most inspiring and hopeful to me involves no computers at all.

Vim in Uganda

The emergence and spread of AIDS has been devastating to sub-Saharan Africa. Sure, you are probably tired of hearing about it. For one thing, it is so hard to come to grips with the scale of the problem. In the short time since I left Botswana--when AIDS was just beginning to emerge as an issue there--life expectancy has plummeted, from nearly 60 years to barely 40. It is now estimated that as many as 40% of the adults in Zimbabwe are HIV positive. This has been a debilitating setback to the emerging countries of the region, where public health efforts had previously been making remarkable gains.

The epicenter of AIDS in Africa has been Uganda, which was hit first and perhaps hardest. The government of Uganda is considered to have mounted an effective and ongoing public health campaign for its people, and there is hope that the incidence of HIV/AIDS is decreasing. Nevertheless, the consequences of the disease have been severe. One of the biggest problems is the large numbers of children left without parents. In a society where children are traditionally treasured and raised with the supportive assistance of extended families, there are simply too

few adults left to care for growing numbers of orphans.

Bram Moolenaar is the author of Vim, one of the most popular open-source text editors, with ports available for just about any platform in existence. Bram had already started Vim when he first went to Uganda in 1994, volunteering to work as a water and sanitation engineer for the Kibaale Children's Centre (KCC).

The center, located in a rural village of southern Uganda, provides food, medical care and education to about 600 children, most of whom have been orphaned by AIDS. The conditions are austere: one book for ten children, a tiny blackboard and a roof with holes.

Bram found that his skills could help at Kibaale, his help made a difference. After a year spent working with the Centre, he wanted to find ways he could continue helping the project while also letting other people know of its existence.

That's when Bram hit on the idea of ``charityware'' for Vim. The license for Vim says simply: ``Vim is Charityware. You can use and copy it as much as you like, but you are encouraged to make a donation to orphans in Uganda. Please read the file doc/uganda.txt for details."

While using Vim, type `:help uganda` to get the complete text of the license and a description of the Kibaale Children's Centre.

Beyond this, though, Bram is fairly modest about the project. Although he asks for copies of CD distributions that include Vim, he doesn't appeal to distribution vendors directly for any additional financial support. Bram prefers to remain low key rather than risk annoying people and turning them away from supporting the Uganda project.

Knowing that Linux distributions in use are now in the billions, one may wonder how successful the charityware license has been as a fund-raising method for the Centre. Vim users are asked to make contributions to the International Child Care Fund that Bram and his colleagues have set up specifically to support the KCC project, and the ICCF web site provides annual financial reports. For 1999, donation income totaled about $7,000 US (17,800 Dutch Guilders), up from about $3,500 US in 1998.

These figures may seem rather underwhelming and suggest that the conscience of open-source users and vendors is not as evolved as one may like to think. But the bottom line for Bram is, even at such a modest level, these contributions make a huge difference in what the KCC can accomplish. The funds raised by Vim donors are used to keep the Centre running, maintain and improve the facilities and recently purchased rainwater tanks so that more people have access to clean water.

Bram continues his personal involvement with Kibaale to this day, having made return trips in 1996, 1998 and 2000. This experience gives Bram a thorough grounding in the realities of life in Africa, as well as an understanding of the means of effecting meaningful change. When I asked for his opinions about the digital divide, he said, ``I'm afraid I don't know what the digital divide is. Is it about bringing computer-related stuff to Third World countries? Well, the area around Kibaale first needs a good water supply and a phone."

When asked if he could give any suggestions to those interested in projects supportive of African information technology, Bram replied, ``The best suggestion I can make is to work in small groups. A hundred small projects bring more benefit than one project that's a hundred times bigger. The strategy and planning done by people in head offices is a waste of time and money." The message here is that the strength of any bridge depends upon its integrity.

In the end, Bram is doing what the Open Source movement has been all about from the beginning: working with personal conviction, making a difference where one can and sharing the work one loves with others. These are the ideals of a world seeking connections, the values that can link Linux and the Internet with an orphanage in Uganda. The human connections of these efforts empower people, improve lives and build the solid bridges of understanding among diverse global communities, digital and otherwise.

Resources

Wayne Marshall *(guinix@yahoo.com) is a UNIX programmer and technical consultant living in Guinea, West Africa.*

Algorithms in Africa

Maybe the rush to market for spreading internet access across the globe isn't in anyone's best interest--a report from the front.

by Wayne Marshall

Eleven years ago I installed a computer system at a vocational training and development center in Tutume, Botswana. Tutume is a rural village on the northeastern edge of the Kgalagadi desert in southern Africa. The computer was intended to help this organization, known as Tutume Brigades, catch up on its bookkeeping for several business units crucial to the local economy. Businesses included a brick-making unit, carpentry workshop, auto repair garage, sorghum mill, school uniform production unit, tractor hire and vegetable garden. For the local village and the surrounding catchment era, the Brigades were literally the only game in the bush for commodities, trade skills, training and employment opportunities.

When I arrived in Tutume, I was a pure novice in the field of foreign assistance. I was also a mid-career financial professional, with several years of experience in nonprofit and health-care management in the United States. And like most aid workers new on the ground in Africa, I knew what was best. In my assessment of the center, I believed a computer was essential to get a handle on the Brigades' financial position, which otherwise consisted of eight separate sets of badly maintained manual ledgers, over nine months in arrears. Except for the bank statements of eight separate checking accounts (and even the bank statements proved unreliable), we had no way of knowing if the center had any money. Every time we had to make payroll or buy another truckload of cement, we were in the heart of fiscal darkness.

Over the course of the next several months, I proceeded to computerize the records and train local staff in basic operation of the system. By the end of the first year, the financial records of the center were timely and accurate. Moreover, other staff members were beginning to use the computer for tasks such as word processing and spreadsheets. Many of these employees had never even used a typewriter before.

If I were to tell no more of this story and fade here to one of the glorious Kgalagadi sunsets, this might be called a win. Although set in the predawn (and pre-Linux) history of the Internet era, today this would be described as a small success story of ``bridging the digital divide'' in

Africa--like I was a regular Albert Schweitzer of the Information Age or something.

But the truth is not so simple, and the issues of foreign assistance are not so trivial. The fact is, I am not proud of this story. Because as my time in Tutume went on, I realized I had blundered badly, to the point of putting the Brigades in serious jeopardy. I began to ask myself such basic questions as: What would happen to the computer after I left? Was the staff fully capable of operating the system independently? Would backups be maintained and performed rigorously? Were skills sufficient to troubleshoot problems and reinstall the system if necessary? If the equipment failed or was stolen, could the center afford to replace it? And what would the center do when the staff I had trained for so long were lured away by more lucrative jobs in the big city?

These questions all led to the same answer: the Brigades would be left in even worse shape than I found them. Rather than gaining empowerment, independence and enablement, they would more than likely be left powerless, dependent and possibly ruined. And all because of my own cultural myopia, despite my good intentions.

It is axiomatic in the field of foreign assistance that the aid program will take credit for the successes, while failures are blamed on the host country. The psychology of failure can then be even more severe and long-lasting than the loss of the project. While I was working in Tutume, for example, a friend of mine was working in the village of Lobatse in southern Botswana. Seven years earlier, an aid organization from northern Europe had decided a wool sweater factory would be just the ticket for the economic development of the village. Of course, northern Europeans are fond of nice wool sweaters and very likely have great need for them, particularly in the colder climes of northern Europe. The market for wool sweaters is less extensive in the sweltering and sparsely populated Kgalagadi desert, however. After seven years of subsidizing the losses of the operation, the aid organization finally decided it was never going to be sustainable, and they pulled the plug on the effort. My friend's unenviable assignment was to put all the women out of work, sell the facility and liquidate the equipment. It was hard for many of the women not to feel that the fault was somehow their own.

Fortunately for Brigades in Tutume, such failure was averted. As the story there continues, once I realized the risks, I spent the next several months converting the accounting system back to manual ledgers, hiring and training additional staff in bookkeeping procedures and enabling them to use the computer primarily as a support system, rather than as the central financial database.

But what do these stories from Tutume and Lobatse have to do with Linux and emerging markets? The rest of this article will consider that question.

The Digital Divide

Nine years have passed since I left Botswana. To put the times into perspective, the first thing I bought when I got back to the US was a fax modem, the cheapest, fastest solution to stay connected with the contacts I had made abroad. My modem then was 2,400 baud. I tried out CompuServe and decided on Delphi, and the buzz was just starting about something called PPP.

During the next several years I was in and out of Africa, became a Linux user in 1995, began installing Linux in nonprofit organizations in 1997, spent a year and Y2K transition in the former soviet state of Ukraine and came to the West African country of Guinea in May 2000. At some point during this period the digital divide was invented.

Actually, the digital divide seems to have its origins in a 1995 report from the US Department of Commerce, whose National Telecommunications and Information Administration (NITA) released the first paper in a series titled ``Falling through the Net''. This report analyzed telecommunication access by geographic and demographic patterns throughout the United States. One of the conclusions of the report was the gap between the ``information rich'' and the ``information poor'' had widened.

In the later years of the Clinton administration, the digital divide broadened beyond US borders to encompass the globe. The issue gained considerable publicity after a G8 economic summit meeting in 1999, where the most powerful nations on earth decided that the growing gap in information technology was one of the most serious problems facing development in the Third World.

Now, as I write this, bridging the digital divide has become one of the hottest trends in foreign assistance, and many aid organizations and corporate philanthropists have found publicity for their efforts. Simplistically, it seems, the gap in information technology has now come to be identified with access to the Internet. Thus, we have such programs as the USAID-funded Leland Initiative, designed to bring internet access to Africa; the Peace Corps announcing an information technology initiative in partnership with AOL; and a recently formed organization called Geekcorps sending its second group of volunteers on three-month stints designing web sites in Accra, the capital of Ghana in West Africa [see *LJ* April 2001 for more on the Geekcorps]. Naturally, the high-profile publicity given this issue has created an opportunity for many international aid organizations to develop projects and funding appeals for serving the digitally needy.

The New Tech Testament

Delivering the miracle of the Internet is the new zeal of the high-tech missionary. In what seems to be a rush to market--bringing the Internet to the developing world--sometimes projects are announced with only naïve regard to the technical issues and without full consideration of whether such projects are viable, appropriate, relevant and sustainable. Thus, one hears of a women's cooperative in Central America marketing their handcrafts over the Web; advocates describe the potential of ``telemedicine'' for delivering virtual health care to isolated areas; and the US State Department Global Technology Corps proclaims, ``We have seen farmers in Mexico using [the Internet] to check weather conditions and crop prices.''

Where once Norwegians may have seen wool sweaters, the tech visionary now sees web browsers.

At the extreme, the new economy proselyte promotes the Internet as the solution for everything from education and health care to pollution, inequality and world peace. As though everyone who has access will be able to browse their way to nirvana, as though the path to heaven is

paved with bandwidth. The satellite dish is the new icon of the digital evangelist, replacing the holy cross.

One of the implicit beliefs of this testament is that information, in and of itself, is sufficient to promote economy, remedy problems and narrow inequities. A corollary implication, the message from one side of the divide to the other, is that we have information and you don't, that our information is good and yours is useless. This is the lesson CNN preaches to its international audience when it tells us, ``The human without information is nothing."

It should be clear that in this form, divide rhetoric is simply new raiment for the familiar old taxonomies of prejudice that have long sought to divide the world between believers and heathens, the enlightened and the savage. From a historical perspective, rather than helping, these kinds of belief systems have generally been devastating to their targets.

More importantly, the belief in the sufficiency of information and information technology is simply wrong. Information alone doesn't help people. If only this were true, doctors would be made from medical textbooks and entrepreneurs would be born from accounting manuals.

In fact, the developing world is littered with unused X-ray equipment, broken-down tractors and empty schoolrooms contributed over the years by well-intentioned and simpleminded donors. These resources are made useless not from missing user manuals or lack of web access, but by the lack of trained technicians, mechanics and teachers.

In short, what empowers people are skills.

Even in the US, this kind of awareness is emerging. In ``How Does the Empty Glass Fill? A Modern Philosophy of the Digital Divide" (Educause Review, Nov/Dec 2000), Solveig Singleton and Lucas Mast write: ``From the standpoint of higher education, students who leave high school without exposure to digital learning tools such as the Internet will prove a much less serious problem than students who leave high school with inadequate reading or math skills."

And the leading journal of free-market capitalism, the *Economist*, recently observed:

> The poor are not shunning the Internet because they cannot afford it: the problem is that they lack the skills to exploit it effectively. So it is difficult to see how connecting the poor to the Internet will improve their finances. It would make more sense to aim for universal literacy than universal Internet access.

It may be that, with the recent outbreak of dot-com bankruptcy and declines in the stock market, the tenets of the digital religion could be losing their currency. At a time when the mega-billion, IPO-funded ebiz stars like Amazon and Yahoo are having a tough go across the US and Europe, it's hard not to wonder how the promises of e-commerce could possibly prove viable and sustainable elsewhere, particularly in places where there aren't even good banking and credit systems. And for someone like me who has lived several years of the past decade in both rural and urban parts of the developing world--where most of the population still cook with firewood and carry water in buckets--the practical value of focusing foreign assistance on IT projects would seem negligible, if not ludicrous entirely. Given the more serious fundamental issues facing developing nations--health care (AIDS, TB and malaria), nutrition, sanitation, education,

poverty, pollution and political corruption--providing the means to surf the Web should probably fall fairly low on any reasonable scale of human priorities.

So is there any way to make a difference, a real difference that improves people's lives? Is there any role for Linux and open-source advocacy in emerging markets? Are there ways of using technology for solving human problems in places like Africa, without trying to sell wool sweaters in the desert? I wouldn't be writing this article if there weren't.

Algorithms in Africa

When it comes to Africa, the so-called digital divide is just a divide; there isn't anything especially digital about it. The divide is geographic, because Africa is a long way away, and cultural, because the traditions and histories of Africans developed independently from those of Europeans and Americans. Almost incidentally the divide is economic, from the standpoint of cash resources and differing perceptions of wealth, though the natural resources of this continent are vast. The divide ends up being mostly one of ignorance, and this gap is at its widest in America.

Americans in general know very little about Africa, and what little they do know or think they know is usually prejudiced and fallacious. If I were to know the state of Florida only from news reports, I would think it was a large mobile-home park of fat pink people constantly flattened by hurricanes. Similarly, most Americans probably only know Africa as a disaster zone of epidemic, starvation and genocide. The principal media image Americans hold of African assistance is usually the one of the brave young (white) woman, a nurse or volunteer, holding a helpless black infant, center stage among a group of grateful and admiring Africans in the background.

Of course Africa is nothing like this image at all, and the first step in crossing the divide here is to banish these offensive stereotypes and learn all one can about what Africa is really like. It would be a disservice to the many peoples of the continent to generalize and describe the essence of Africa as though it were a single place. But I would just like to say: Africa is such a joy! Whenever I am in the streets of Conakry or an upcountry village, I am overwhelmed with the pure bandwidth of humanity, of color and vitality and life. So much more than can ever be expressed on even your largest CRT, with even the fastest DSL connection; Africa is the ultimate realization of broadband in culture and diversity, natural and human content. Maybe a virtual, flat-screened reality over the Internet is meaningful in the pitifully dreary cubicle of the US office worker, but Africa is all about face time in real time.

Open-source advocates can be sure that Africans get community; Africans get bazaar. These are concepts intrinsic to the cultures and traditions throughout the continent, where African societies had mastered networking long before the invention of the RJ45 jack. Africans have historically been quite receptive, often at their ultimate peril, to ideas and innovations flowing between cultures and brought in by outsiders. And in general Africa has been early and enthusiastic about adopting new communication technologies, particularly when they are practical and affordable. So in Botswana I was astonished at the number of fax machines per capita ten years ago, and now find a thriving trade in cell phones, both legitimate and black

market, in Guinea. On a recent visit to a mosque in the interior of the country, a wizened old muezzin took me up into the minaret specifically to show me their solar-powered amplifier and loudspeaker system, used to call the village to prayers.

As one learns to develop an appreciation of what Africa is really like, it will then help if one can develop a sensitivity to the pitfalls of foreign aid and the havoc such programs have brought to this continent. The subject of other narrations, it is sufficient to observe here that the official assistance programs of foreign governments are usually a foul brew of political hegemony, economic imperialism, cultural ethnocentrism, personal avarice and, too rarely, genuine altruism. Too often the implementation of foreign aid is all about developing market share and spheres of influence, instead of improving lives. Proponents of foreign assistance may even argue that these are synonymous, as though markets for American soft drinks, snack foods and beauty products result in happiness and prosperity for the consumer. The sad fact is, whether intentional or merely consequential, foreign assistance has often had devastating effects on communities, local markets, traditional cultures and environmental conditions throughout Africa.

Finally, it is helpful to bring an honest perspective of one's own history and culture into focus. For example, the United States represents less than 6% of the world's total population and has existed for less than a blink of an eye in the span of all human history. So, what makes us think we've got it right? What evidence is there to suggest this brief record is proof that our way of life and cultural adaptations will be viable in the long run?

For example, it may be surprising to learn that, due to the predations of infectious illness, urban population levels were not even sustainable until about 100 years ago and required steady migration from rural areas. And it was less than 90 years ago, Gina Kolata writes in *Flu,* when ``*Ladies Home Journal* proudly declared that the parlor, where the dead had been laid out for viewing, was now to be called the living room, a room for the living, not the dead."

Shortly after this proclamation, a global flu of epidemic proportion--the origin of which is still not understood--killed 1.5 million Americans and 40 million worldwide. This was not in the murky history of the Dark Ages; this was 1918. Today, with the modern plague of HIV/AIDS, the re-emergence of tuberculosis and new mysteries like the relationship of human CJD to Mad Cow Disease, will our mastery of medicine prove all that enduring, even for the world's most fortunate few?

In any case, those who would help others should at least try to learn from past failures and have the humility to ask if the modern model of urbanization, congestion, resource utilization and environmental depletion are sustainable, even desirable, let alone worthy of export to others in the world.

Then we may be able to accept that the Internet may not be the solution to all problems of humankind and have the patience to realize that working through the major challenges in Africa will take time and understanding measured in generations. Now it becomes clear that Linux and open-source developers are helping Africa best by what they have been doing already. People who are programming and installing the world-class, free software at the soul of internet technology are helping others around the world in profound and important ways, no matter what

license they are using. GNU and open-source software are the perfect fit for the emerging nations of Africa--as for the rest of the world--not only for the superior technical quality of these systems, but for the values embodied in their development.

The mere existence of Linux and open-source systems give people the chance to use these powerful technologies for low-cost, grassroots level applications, an opportunity not possible just ten years ago. The pages of this magazine have described many of these self-directed success stories, everywhere from Mexico to Pakistan, where Linux solutions enabled people to make the difference. Such examples are to be found among African communities as well, from South Africa to Kenya to Nigeria. And Africans like Katim Touray are using Linux servers to connect other Africans in dialogue around the world.

Beyond the software itself, though, it is the culture of Linux and Open Source communities that provides the model for meaningful outcomes. This is the culture of sharing and empowerment, of the thousands of Linux users' groups throughout the world, of the Linux Documentation Project and the general willingness of one user to selflessly help another. Participating as a Linux user is all about developing crucial skills and passing them on. Often users' groups hold regular installation clinics, giving new users personal, one-on-one support from an enthusiastic peer. And these users' groups are often active in other community projects, such as helping schools install servers and network connectivity, while transferring the skills necessary to maintain them. Each of these connections is essentially more human than technical, linking people together more than their machines, and can lead anywhere. Each of these personal connections sows the seeds of others, and the spread of the Linux bloom is now reaching to every corner of the earth. For example, even though the use of internet technology in Guinea is nascent, Linux certainly preceded my own arrival here. One finds Linux books in French in bookstores and Guineans eager to learn more about this ``true'' operating system.

And there are other instances of Linux and open source helping to solve problems in Africa. One of the most inspiring and hopeful to me involves no computers at all.

Vim in Uganda

The emergence and spread of AIDS has been devastating to sub-Saharan Africa. Sure, you are probably tired of hearing about it. For one thing, it is so hard to come to grips with the scale of the problem. In the short time since I left Botswana--when AIDS was just beginning to emerge as an issue there--life expectancy has plummeted, from nearly 60 years to barely 40. It is now estimated that as many as 40% of the adults in Zimbabwe are HIV positive. This has been a debilitating setback to the emerging countries of the region, where public health efforts had previously been making remarkable gains.

The epicenter of AIDS in Africa has been Uganda, which was hit first and perhaps hardest. The government of Uganda is considered to have mounted an effective and ongoing public health campaign for its people, and there is hope that the incidence of HIV/AIDS is decreasing. Nevertheless, the consequences of the disease have been severe. One of the biggest problems is the large numbers of children left without parents. In a society where children are traditionally treasured and raised with the supportive assistance of extended families, there are simply too

few adults left to care for growing numbers of orphans.

Bram Moolenaar is the author of Vim, one of the most popular open-source text editors, with ports available for just about any platform in existence. Bram had already started Vim when he first went to Uganda in 1994, volunteering to work as a water and sanitation engineer for the Kibaale Children's Centre (KCC).

The center, located in a rural village of southern Uganda, provides food, medical care and education to about 600 children, most of whom have been orphaned by AIDS. The conditions are austere: one book for ten children, a tiny blackboard and a roof with holes.

Bram found that his skills could help at Kibaale, his help made a difference. After a year spent working with the Centre, he wanted to find ways he could continue helping the project while also letting other people know of its existence.

That's when Bram hit on the idea of ``charityware'' for Vim. The license for Vim says simply: ``Vim is Charityware. You can use and copy it as much as you like, but you are encouraged to make a donation to orphans in Uganda. Please read the file doc/uganda.txt for details."

While using Vim, type `:help uganda` to get the complete text of the license and a description of the Kibaale Children's Centre.

Beyond this, though, Bram is fairly modest about the project. Although he asks for copies of CD distributions that include Vim, he doesn't appeal to distribution vendors directly for any additional financial support. Bram prefers to remain low key rather than risk annoying people and turning them away from supporting the Uganda project.

Knowing that Linux distributions in use are now in the billions, one may wonder how successful the charityware license has been as a fund-raising method for the Centre. Vim users are asked to make contributions to the International Child Care Fund that Bram and his colleagues have set up specifically to support the KCC project, and the ICCF web site provides annual financial reports. For 1999, donation income totaled about $7,000 US (17,800 Dutch Guilders), up from about $3,500 US in 1998.

These figures may seem rather underwhelming and suggest that the conscience of open-source users and vendors is not as evolved as one may like to think. But the bottom line for Bram is, even at such a modest level, these contributions make a huge difference in what the KCC can accomplish. The funds raised by Vim donors are used to keep the Centre running, maintain and improve the facilities and recently purchased rainwater tanks so that more people have access to clean water.

Bram continues his personal involvement with Kibaale to this day, having made return trips in 1996, 1998 and 2000. This experience gives Bram a thorough grounding in the realities of life in Africa, as well as an understanding of the means of effecting meaningful change. When I asked for his opinions about the digital divide, he said, ``I'm afraid I don't know what the digital divide is. Is it about bringing computer-related stuff to Third World countries? Well, the area around Kibaale first needs a good water supply and a phone."

When asked if he could give any suggestions to those interested in projects supportive of African information technology, Bram replied, ``The best suggestion I can make is to work in small groups. A hundred small projects bring more benefit than one project that's a hundred times bigger. The strategy and planning done by people in head offices is a waste of time and money." The message here is that the strength of any bridge depends upon its integrity.

In the end, Bram is doing what the Open Source movement has been all about from the beginning: working with personal conviction, making a difference where one can and sharing the work one loves with others. These are the ideals of a world seeking connections, the values that can link Linux and the Internet with an orphanage in Uganda. The human connections of these efforts empower people, improve lives and build the solid bridges of understanding among diverse global communities, digital and otherwise.

Resources

Wayne Marshall *(guinix@yahoo.com) is a UNIX programmer and technical consultant living in Guinea, West Africa.*

Algorithms in Africa

Maybe the rush to market for spreading internet access across the globe isn't in anyone's best interest--a report from the front.

by Wayne Marshall

Eleven years ago I installed a computer system at a vocational training and development center in Tutume, Botswana. Tutume is a rural village on the northeastern edge of the Kgalagadi desert in southern Africa. The computer was intended to help this organization, known as Tutume Brigades, catch up on its bookkeeping for several business units crucial to the local economy. Businesses included a brick-making unit, carpentry workshop, auto repair garage, sorghum mill, school uniform production unit, tractor hire and vegetable garden. For the local village and the surrounding catchment era, the Brigades were literally the only game in the bush for commodities, trade skills, training and employment opportunities.

When I arrived in Tutume, I was a pure novice in the field of foreign assistance. I was also a mid-career financial professional, with several years of experience in nonprofit and health-care management in the United States. And like most aid workers new on the ground in Africa, I knew what was best. In my assessment of the center, I believed a computer was essential to get a handle on the Brigades' financial position, which otherwise consisted of eight separate sets of badly maintained manual ledgers, over nine months in arrears. Except for the bank statements of eight separate checking accounts (and even the bank statements proved unreliable), we had no way of knowing if the center had any money. Every time we had to make payroll or buy another truckload of cement, we were in the heart of fiscal darkness.

Over the course of the next several months, I proceeded to computerize the records and train local staff in basic operation of the system. By the end of the first year, the financial records of the center were timely and accurate. Moreover, other staff members were beginning to use the computer for tasks such as word processing and spreadsheets. Many of these employees had never even used a typewriter before.

If I were to tell no more of this story and fade here to one of the glorious Kgalagadi sunsets, this might be called a win. Although set in the predawn (and pre-Linux) history of the Internet era, today this would be described as a small success story of ``bridging the digital divide'' in Africa--like I was a regular Albert Schweitzer of the Information Age or something.

But the truth is not so simple, and the issues of foreign assistance are not so trivial. The fact is, I am not proud of this story. Because as my time in Tutume went on, I realized I had blundered badly, to the point of putting the Brigades in serious jeopardy. I began to ask myself such basic questions as: What would happen to the computer after I left? Was the staff fully capable of operating the system independently? Would backups be maintained and performed rigorously? Were skills sufficient to troubleshoot problems and reinstall the system if necessary? If the equipment failed or was stolen, could the center afford to replace it? And what would the center do when the staff I had trained for so long were lured away by more lucrative jobs in the big city?

These questions all led to the same answer: the Brigades would be left in even worse shape than I found them. Rather than gaining empowerment, independence and enablement, they would more than likely be left powerless, dependent and possibly ruined. And all because of my own cultural myopia, despite my good intentions.

It is axiomatic in the field of foreign assistance that the aid program will take credit for the successes, while failures are blamed on the host country. The psychology of failure can then be even more severe and long-lasting than the loss of the project. While I was working in Tutume, for example, a friend of mine was working in the village of Lobatse in southern Botswana. Seven years earlier, an aid organization from northern Europe had decided a wool sweater factory would be just the ticket for the economic development of the village. Of course, northern Europeans are fond of nice wool sweaters and very likely have great need for them, particularly in the colder climes of northern Europe. The market for wool sweaters is less extensive in the sweltering and sparsely populated Kgalagadi desert, however. After seven years of subsidizing the losses of the operation, the aid organization finally decided it was never going to be sustainable, and they pulled the plug on the effort. My friend's unenviable assignment was to put all the women out of work, sell the facility and liquidate the equipment. It was hard for many of the women not to feel that the fault was somehow their own.

Fortunately for Brigades in Tutume, such failure was averted. As the story there continues, once I realized the risks, I spent the next several months converting the accounting system back to manual ledgers, hiring and training additional staff in bookkeeping procedures and enabling them to use the computer primarily as a support system, rather than as the central financial database.

But what do these stories from Tutume and Lobatse have to do with Linux and emerging markets? The rest of this article will consider that question.

The Digital Divide

Nine years have passed since I left Botswana. To put the times into perspective, the first thing I bought when I got back to the US was a fax modem, the cheapest, fastest solution to stay connected with the contacts I had made abroad. My modem then was 2,400 baud. I tried out CompuServe and decided on Delphi, and the buzz was just starting about something called PPP.

During the next several years I was in and out of Africa, became a Linux user in 1995, began installing Linux in nonprofit organizations in 1997, spent a year and Y2K transition in the former soviet state of Ukraine and came to the West African country of Guinea in May 2000. At some point during this period the digital divide was invented.

Actually, the digital divide seems to have its origins in a 1995 report from the US Department of Commerce, whose National Telecommunications and Information Administration (NITA) released the first paper in a series titled ``Falling through the Net''. This report analyzed telecommunication access by geographic and demographic patterns throughout the United States. One of the conclusions of the report was the gap between the ``information rich'' and the ``information poor'' had widened.

In the later years of the Clinton administration, the digital divide broadened beyond US borders to encompass the globe. The issue gained

considerable publicity after a G8 economic summit meeting in 1999, where the most powerful nations on earth decided that the growing gap in information technology was one of the most serious problems facing development in the Third World.

Now, as I write this, bridging the digital divide has become one of the hottest trends in foreign assistance, and many aid organizations and corporate philanthropists have found publicity for their efforts. Simplistically, it seems, the gap in information technology has now come to be identified with access to the Internet. Thus, we have such programs as the USAID-funded Leland Initiative, designed to bring internet access to Africa; the Peace Corps announcing an information technology initiative in partnership with AOL; and a recently formed organization called Geekcorps sending its second group of volunteers on three-month stints designing web sites in Accra, the capital of Ghana in West Africa [see LJ April 2001 for more on the Geekcorps]. Naturally, the high-profile publicity given this issue has created an opportunity for many international aid organizations to develop projects and funding appeals for serving the digitally needy.

The New Tech Testament

Delivering the miracle of the Internet is the new zeal of the high-tech missionary. In what seems to be a rush to market--bringing the Internet to the developing world--sometimes projects are announced with only naïve regard to the technical issues and without full consideration of whether such projects are viable, appropriate, relevant and sustainable. Thus, one hears of a women's cooperative in Central America marketing their handcrafts over the Web; advocates describe the potential of ``telemedicine'' for delivering virtual health care to isolated areas; and the US State Department Global Technology Corps proclaims, ``We have seen farmers in Mexico using [the Internet] to check weather conditions and crop prices.''

Where once Norwegians may have seen wool sweaters, the tech visionary now sees web browsers.

At the extreme, the new economy proselyte promotes the Internet as the solution for everything from education and health care to pollution, inequality and world peace. As though everyone who has access will be able to browse their way to nirvana, as though the path to heaven is paved with bandwidth. The satellite dish is the new icon of the digital evangelist, replacing the holy cross.

One of the implicit beliefs of this testament is that information, in and of itself, is sufficient to promote economy, remedy problems and narrow inequities. A corollary implication, the message from one side of the divide to the other, is that we have information and you don't, that our information is good and yours is useless. This is the lesson CNN preaches to its international audience when it tells us, ``The human without information is nothing.''

It should be clear that in this form, divide rhetoric is simply new raiment for the familiar old taxonomies of prejudice that have long sought to divide the world between believers and heathens, the enlightened and the savage. From a historical perspective, rather than helping, these kinds of belief systems have generally been devastating to their targets.

More importantly, the belief in the sufficiency of information and information technology is simply wrong. Information alone doesn't help people. If only this were true, doctors would be made from medical textbooks and entrepreneurs would be born from accounting manuals.

In fact, the developing world is littered with unused X-ray equipment, broken-down tractors and empty schoolrooms contributed over the years by well-intentioned and simpleminded donors. These resources are made useless not from missing user manuals or lack of web access, but by the lack of trained technicians, mechanics and teachers.

In short, what empowers people are skills.

Even in the US, this kind of awareness is emerging. In ``How Does the Empty Glass Fill? A Modern Philosophy of the Digital Divide'' (Educause Review, Nov/Dec 2000), Solveig Singleton and Lucas Mast write: ``From the standpoint of higher education, students who leave high school without exposure to digital learning tools such as the Internet will prove a much less serious problem than students who leave high school with inadequate reading or math skills.''

And the leading journal of free-market capitalism, the Economist, recently observed:

> The poor are not shunning the Internet because they cannot afford it: the problem is that they lack the skills to exploit it effectively. So it is difficult to see how connecting the poor to the Internet will improve their finances. It would make more sense to aim for universal literacy than universal Internet access.

It may be that, with the recent outbreak of dot-com bankruptcy and declines in the stock market, the tenets of the digital religion could be losing their currency. At a time when the mega-billion, IPO-funded ebiz stars like Amazon and Yahoo are having a tough go across the US and Europe, it's hard not to wonder how the promises of e-commerce could possibly prove viable and sustainable elsewhere, particularly in places where there aren't even good banking and credit systems. And for someone like me who has lived several years of the past decade in both rural and urban parts of the developing world--where most of the population still cook with firewood and carry water in buckets--the practical value of focusing foreign assistance on IT projects would seem negligible, if not ludicrous entirely. Given the more serious fundamental issues facing developing nations--health care (AIDS, TB and malaria), nutrition, sanitation, education, poverty, pollution and political corruption--providing the means to surf the Web should probably fall fairly low on any reasonable scale of human priorities.

So is there any way to make a difference, a real difference that improves people's lives? Is there any role for Linux and open-source advocacy in emerging markets? Are there ways of using technology for solving human problems in places like Africa, without trying to sell wool sweaters in the desert? I wouldn't be writing this article if there weren't.

Algorithms in Africa

When it comes to Africa, the so-called digital divide is just a divide; there isn't anything especially digital about it. The divide is geographic, because Africa is a long way away, and cultural, because the traditions and histories of Africans developed independently from those of Europeans and Americans. Almost incidentally the divide is economic, from the standpoint of cash resources and differing perceptions of wealth, though the natural resources of this continent are vast. The divide ends up being mostly one of ignorance, and this gap is at its widest in America.

Americans in general know very little about Africa, and what little they do

know or think they know is usually prejudiced and fallacious. If I were to know the state of Florida only from news reports, I would think it was a large mobile-home park of fat pink people constantly flattened by hurricanes. Similarly, most Americans probably only know Africa as a disaster zone of epidemic, starvation and genocide. The principal media image Americans hold of African assistance is usually the one of the brave young (white) woman, a nurse or volunteer, holding a helpless black infant, center stage among a group of grateful and admiring Africans in the background.

Of course Africa is nothing like this image at all, and the first step in crossing the divide here is to banish these offensive stereotypes and learn all one can about what Africa is really like. It would be a disservice to the many peoples of the continent to generalize and describe the essence of Africa as though it were a single place. But I would just like to say: Africa is such a joy! Whenever I am in the streets of Conakry or an upcountry village, I am overwhelmed with the pure bandwidth of humanity, of color and vitality and life. So much more than can ever be expressed on even your largest CRT, with even the fastest DSL connection; Africa is the ultimate realization of broadband in culture and diversity, natural and human content. Maybe a virtual, flat-screened reality over the Internet is meaningful in the pitifully dreary cubicle of the US office worker, but Africa is all about face time in real time.

Open-source advocates can be sure that Africans get community; Africans get bazaar. These are concepts intrinsic to the cultures and traditions throughout the continent, where African societies had mastered networking long before the invention of the RJ45 jack. Africans have historically been quite receptive, often at their ultimate peril, to ideas and innovations flowing between cultures and brought in by outsiders. And in general Africa has been early and enthusiastic about adopting new communication technologies, particularly when they are practical and affordable. So in Botswana I was astonished at the number of fax machines per capita ten years ago, and now find a thriving trade in cell phones, both legitimate and black market, in Guinea. On a recent visit to a mosque in the interior of the country, a wizened old muezzin took me up into the minaret specifically to show me their solar-powered amplifier and loudspeaker system, used to call the village to prayers.

As one learns to develop an appreciation of what Africa is really like, it will then help if one can develop a sensitivity to the pitfalls of foreign aid and the havoc such programs have brought to this continent. The subject of other narrations, it is sufficient to observe here that the official assistance programs of foreign governments are usually a foul brew of political hegemony, economic imperialism, cultural ethnocentrism, personal avarice and, too rarely, genuine altruism. Too often the implementation of foreign aid is all about developing market share and spheres of influence, instead of improving lives. Proponents of foreign assistance may even argue that these are synonymous, as though markets for American soft drinks, snack foods and beauty products result in happiness and prosperity for the consumer. The sad fact is, whether intentional or merely consequential, foreign assistance has often had devastating effects on communities, local markets, traditional cultures and environmental conditions throughout Africa.

Finally, it is helpful to bring an honest perspective of one's own history and culture into focus. For example, the United States represents less than 6% of the world's total population and has existed for less than a blink of an eye in the span of all human history. So, what makes us think we've got it right? What evidence is there to suggest this brief record is proof that

our way of life and cultural adaptations will be viable in the long run?

For example, it may be surprising to learn that, due to the predations of infectious illness, urban population levels were not even sustainable until about 100 years ago and required steady migration from rural areas. And it was less than 90 years ago, Gina Kolata writes in Flu, when ``Ladies Home Journal proudly declared that the parlor, where the dead had been laid out for viewing, was now to be called the living room, a room for the living, not the dead.''

Shortly after this proclamation, a global flu of epidemic proportion--the origin of which is still not understood--killed 1.5 million Americans and 40 million worldwide. This was not in the murky history of the Dark Ages; this was 1918. Today, with the modern plague of HIV/AIDS, the re-emergence of tuberculosis and new mysteries like the relationship of human CJD to Mad Cow Disease, will our mastery of medicine prove all that enduring, even for the world's most fortunate few?

In any case, those who would help others should at least try to learn from past failures and have the humility to ask if the modern model of urbanization, congestion, resource utilization and environmental depletion are sustainable, even desirable, let alone worthy of export to others in the world.

Then we may be able to accept that the Internet may not be the solution to all problems of humankind and have the patience to realize that working through the major challenges in Africa will take time and understanding measured in generations. Now it becomes clear that Linux and open-source developers are helping Africa best by what they have been doing already. People who are programming and installing the world-class, free software at the soul of internet technology are helping others around the world in profound and important ways, no matter what license they are using. GNU and open-source software are the perfect fit for the emerging nations of Africa--as for the rest of the world--not only for the superior technical quality of these systems, but for the values embodied in their development.

The mere existence of Linux and open-source systems give people the chance to use these powerful technologies for low-cost, grassroots level applications, an opportunity not possible just ten years ago. The pages of this magazine have described many of these self-directed success stories, everywhere from Mexico to Pakistan, where Linux solutions enabled people to make the difference. Such examples are to be found among African communities as well, from South Africa to Kenya to Nigeria. And Africans like Katim Touray are using Linux servers to connect other Africans in dialogue around the world.

Beyond the software itself, though, it is the culture of Linux and Open Source communities that provides the model for meaningful outcomes. This is the culture of sharing and empowerment, of the thousands of Linux users' groups throughout the world, of the Linux Documentation Project and the general willingness of one user to selflessly help another. Participating as a Linux user is all about developing crucial skills and passing them on. Often users' groups hold regular installation clinics, giving new users personal, one-on-one support from an enthusiastic peer. And these users' groups are often active in other community projects, such as helping schools install servers and network connectivity, while transferring the skills necessary to maintain them. Each of these connections is essentially more human than technical, linking people together more than their machines, and can lead anywhere. Each of these personal connections sows the seeds of others, and the spread of the Linux bloom is now reaching to every corner of

the earth. For example, even though the use of internet technology in Guinea is nascent, Linux certainly preceded my own arrival here. One finds Linux books in French in bookstores and Guineans eager to learn more about this ``true'' operating system.

And there are other instances of Linux and open source helping to solve problems in Africa. One of the most inspiring and hopeful to me involves no computers at all.

Vim in Uganda

The emergence and spread of AIDS has been devastating to sub-Saharan Africa. Sure, you are probably tired of hearing about it. For one thing, it is so hard to come to grips with the scale of the problem. In the short time since I left Botswana--when AIDS was just beginning to emerge as an issue there--life expectancy has plummeted, from nearly 60 years to barely 40. It is now estimated that as many as 40% of the adults in Zimbabwe are HIV positive. This has been a debilitating setback to the emerging countries of the region, where public health efforts had previously been making remarkable gains.

The epicenter of AIDS in Africa has been Uganda, which was hit first and perhaps hardest. The government of Uganda is considered to have mounted an effective and ongoing public health campaign for its people, and there is hope that the incidence of HIV/AIDS is decreasing. Nevertheless, the consequences of the disease have been severe. One of the biggest problems is the large numbers of children left without parents. In a society where children are traditionally treasured and raised with the supportive assistance of extended families, there are simply too few adults left to care for growing numbers of orphans.

Bram Moolenaar is the author of Vim, one of the most popular open-source text editors, with ports available for just about any platform in existence. Bram had already started Vim when he first went to Uganda in 1994, volunteering to work as a water and sanitation engineer for the Kibaale Children's Centre (KCC).

The center, located in a rural village of southern Uganda, provides food, medical care and education to about 600 children, most of whom have been orphaned by AIDS. The conditions are austere: one book for ten children, a tiny blackboard and a roof with holes.

Bram found that his skills could help at Kibaale, his help made a difference. After a year spent working with the Centre, he wanted to find ways he could continue helping the project while also letting other people know of its existence.

That's when Bram hit on the idea of ``charityware'' for Vim. The license for Vim says simply: ``Vim is Charityware. You can use and copy it as much as you like, but you are encouraged to make a donation to orphans in Uganda. Please read the file doc/uganda.txt for details.''

While using Vim, type :help uganda to get the complete text of the license and a description of the Kibaale Children's Centre.

Beyond this, though, Bram is fairly modest about the project. Although he asks for copies of CD distributions that include Vim, he doesn't appeal to distribution vendors directly for any additional financial support. Bram prefers to remain low key rather than risk annoying people and turning them away from supporting the Uganda project.

Knowing that Linux distributions in use are now in the billions, one may wonder how successful the charityware license has been as a fund-raising method for the Centre. Vim users are asked to make contributions to the International Child Care Fund that Bram and his colleagues have set up specifically to support the KCC project, and the ICCF web site provides annual financial reports. For 1999, donation income totaled about $7,000 US (17,800 Dutch Guilders), up from about $3,500 US in 1998.

These figures may seem rather underwhelming and suggest that the conscience of open-source users and vendors is not as evolved as one may like to think. But the bottom line for Bram is, even at such a modest level, these contributions make a huge difference in what the KCC can accomplish. The funds raised by Vim donors are used to keep the Centre running, maintain and improve the facilities and recently purchased rainwater tanks so that more people have access to clean water.

Bram continues his personal involvement with Kibaale to this day, having made return trips in 1996, 1998 and 2000. This experience gives Bram a thorough grounding in the realities of life in Africa, as well as an understanding of the means of effecting meaningful change. When I asked for his opinions about the digital divide, he said, ``I'm afraid I don't know what the digital divide is. Is it about bringing computer-related stuff to Third World countries? Well, the area around Kibaale first needs a good water supply and a phone.''

When asked if he could give any suggestions to those interested in projects supportive of African information technology, Bram replied, ``The best suggestion I can make is to work in small groups. A hundred small projects bring more benefit than one project that's a hundred times bigger. The strategy and planning done by people in head offices is a waste of time and money.'' The message here is that the strength of any bridge depends upon its integrity.

In the end, Bram is doing what the Open Source movement has been all about from the beginning: working with personal conviction, making a difference where one can and sharing the work one loves with others. These are the ideals of a world seeking connections, the values that can link Linux and the Internet with an orphanage in Uganda. The human connections of these efforts empower people, improve lives and build the solid bridges of understanding among diverse global communities, digital and otherwise.

Resources

Wayne Marshall (guinix@yahoo.com) is a UNIX programmer and technical consultant living in Guinea, West Africa.

=== "Co je to Vim?
=== Struèný exkurs na ¹esti kilobajtech

Vim (Vi IMproved, tedy vylep¹ené vi) je program podobný textovému
editoru vi.

Vi pracuje v textovém re¾imu na ka¾dém terminálu, ov¹em obsahuje i
grafické rozhraní s menu a dùslednou podporou my¹i.

Dostupnost a kompatibilita:
Vim je dostupný na mnoha platformách; ve srovnání s vi má mnoho dal¹ích
mo¾ností (viz http://www.vim.org/doc/vi.diff.txt). Vim je kompatibilní
prakticky se v¹emi pøíkazy vi --- kromì chyb, samozøejmì :-)

Operaèní systémy:
Vim je pou¾itelný na mnoha systémech: AmigaOS, Atari MiNT, BeOS, DOS,
MacOS, NextStep, OS/2, OSF, RiscOS, SGI, UNIX, VMS, Win16 + Win32
(Windows95/98/00/NT) --- no a samozøejmì FreeBSD a Linux :-)

Copyright:
Autorská práva jsou v rukou hlavního autora, Brama Moolenaar
<bram@vim.org>. Vim je charity-ware, co¾ znamená, ¾e by bylo dobré
pøispìt na sirotky v Ugandì (viz ":help uganda"), pokud se vám Vim líbí.

Zdrojový kód:
Vim je OpenSource a vítán je kdokoliv, kdo chce pøispìt k jeho vylep¹ení.

=== Mo¾nosti

U¾ivatelská pøítulnost:
Vim je pro zaèáteèníky mnohem pøítulnìj¹í, ne¾ vi. Obsahuje toti¾
rozsáhlou online nápovìdu, undo a redo, podporu pro my¹ a nastavitelné
menu.

Znakové sady a terminály:
Vim podporuje znakovou sadu ISO-Latin1 a termcap. Editace textù
s nabodeníèky je pod rozumnì nastaveným systémem bezproblémová.

Jazyky:
Vim má podporu pro psaní zprava doleva (Arab¹tina, Farsi, Hebrej¹tina) a
multi-byte texty --- tedy podporuje i jazyky, jejich¾ znaky jsou
reprezentovány více ne¾ jedním bytem: Èín¹tinu, Japon¹tinu, Korej¹tinu
apod. Technicky øeèeno, Vim podporuje UTF-8 a Unicode.

Formátování textu a vizuální re¾im:
Vim doká¾e oznaèit text do bloku viditelným zpùsobem (oznaèí ho inverznì) a
následnì na nìm provádìt spoustu operací --- kopírovat, mazat, nahrazovat,
posouvat doleva/doprava, zmìnit velikost písmen nebo formátovat celý blok se
zachováním pøedchozího zarovnání apod.

Doplòování:
Vim doká¾e doplòovat to, co pí¹ete --- a» u¾ jsou to pøíkazy, jména souborù
nebo slova.

Automatické pøíkazy:
Vim umo¾òuje vykonávat pøedem urèené akce, napøíklad dekomprimaci zabalených
souborù.

Speciální znaky:
Vim umo¾òuje vkládat speciální znaky kombinací dvou znakù, napøíklad

kombinace '"' a 'a' vlo¾í do textu 'ä'.

Zji¹tìní typu souboru a jeho konverze:
Vim automaticky rozpoznává typ souboru (DOS, Mac, UNIX) a umo¾òuje soubor ukládat v libovolném z tìchto typù, tak¾e odpadá nutnost pou¾ití dal¹ích konverzních skriptù.

Historie:
Vim si pamatuje pøíkazy, které jste ji¾ napsali a umo¾òuje vám je opìt vyvolat a upravovat.

Makra:
Vim doká¾e nahrát posloupnost pøíkazù a posléze ji pøehrát, tak¾e zjednodu¹uje opakovanou práci s textem.

Pamì»ové limity:
Vim toho vydr¾í mnohem víc ne¾ tradièní vi, a» u¾ v délce øádky nebo velikosti schránek.

Více schránek a rozdìlení obrazovky:
Vim umo¾òuje upravovat více rùzných schránek s textem najednou; mù¾ete si celé okno rozdìlit na víc èástí (jak na vý¹ku, tak na ¹íøku) a v ka¾dé z nich upravovat jiný soubor nebo jinou èást tého¾ souboru.

Èíselné pøedpony pøíkazù:
Èíselná pøedpona umo¾òuje provést urèitý pøíkaz opakovanì (dd = smazat øádek, 3dd = smazat tøi øádky), Vim dovoluje pou¾ít èíselných pøedpon i u pøíkazù, kde to vi nedovoluje (napøíklad "put").

Doprovodné soubory:
Vim obsahuje celou øadu souborù, které jsou jeho souèástí, ale nemusí být pøekládány spolu s programem, napøíklad soubory s definicí syntaxe nebo soubory s nápovìdou. Vim 5.7 obsahuje 70 souborù s nápovìdou (kolem dvou megabytù textu), ta popisuje ve¹keré pøíkazy, nastavení a rùzné tipy ohlednì konfigurace Vimu a úpravy textù. Vim 6.0x obsahuje 85 souborù nápovìdy, dohromady necelé tøi megabyty textu. Nìkteré soubory nápovìdy se zaobírají specifiky Vimu na konkrétních operaèních systémech.

Skripty:
Vim obsahuje skriptovací jazyk pro jednoduché roz¹iøování.

Ulo¾ení stavu:
Není problém ulo¾it stav celého prostøedí do souboru "viminfo" a pou¾ít ho a¾ po jisté sléze. Ulo¾it se dají napøíklad v¹echny schránky, pozice v souborech a zálo¾ky v nich, registry, pøíkazová historie a historie vyhledávání.

Doplòování tabulátoru:
Vim umo¾òuje automaticky doplnit tabulátory tam, kde je nìjaký tabulátor obklopený mezerami, pøípadnì mù¾e nahradit odpovídajícím poètem tabulátorù øetìzec mezer.

Systém tagù:
Tagy jsou èásti textu, které se dají chápat jako náve¹tí --- lze je prohledávat, skákat z jednoho na druhý apod. Napøíklad v C lze jednodu¹e odskoèit ze jména funkce na její definici.

Textové objekty:
Vim chápe jisté èásti textu (slovo, vìta, odstavec) jako ucelené objekty a umo¾òuje s nimi jako¾to s celky pracovat. Je mo¾né upravit definici

tìchto objektù, a tedy zmìnit, co si Vim pøedstaví pod pojmem "slovo" apod.

Barevné odli¹ení syntaxe:
Vim umí obarvit text podle jeho typu, mù¾ete si také definovat vlastní syntaxi --- není tedy problémem tøeba ve spojení s programem ISpell obarvit pøeklepy apod.

Vim má pøeddefinované soubory syntaxe pro vìt¹inu pou¾ívaných programovacích jazykù (Ada, C, C++, Eiffel, Fortran, Haskell, Java, Lisp, Modula, Pascal, Prolog, Python, Scheme, Smalltalk, SQL, Verilog, VisualBasic), matematických programù (Maple, Matlab, Mathematica, SAS), znaèkovacích jazykù (DocBook, HTML, LaTeX, PostScript, SGML-LinuxDoc, TeX, WML, XML), výstupù z programù (diff, man), konfiguraèních souborù (4DOS, Apache, autoconfig, BibTeX, CSS, CVS, elm, IDL, LILO, pine, procmail, samba, slrn), shellových skriptù (sh, bash, csh, ksh, zsh), skriptovacích jazykù (awk, Perl, sed, yacc), systémových souborù (printcap, .Xdefaults) a samozøejmì Vim a jeho soubory s nápovìdou.

Dal¹í mo¾nosti:
Do Vimu je mo¾né integrovat Perl, Tcl nebo Python. Pod Windows se Vim mù¾e chovat jako OLE automation server. Pod X Window umo¾òuje Vim vyu¾ívat nastavitelná menu a my¹.

=== Odkazy

Domovská stránka Vimu
  http://www.vim.org

Podrobnìj¹í popis vlastností a mo¾ností Vimu
  http://www.vim.org/why.html

=== Autor a pøekladatel

Originál napsal: Sven Guckes guckes@vim.org
Poslední úprava: Mon Mar 12 07:00:00 MET 2001

Pøeklad: Tomá¹ Znamenáèek tomas.znamenacek@worldonline.cz
Poslední úprava: So kvì 19 09:04:58 CEST 2001

```
longname        abbr    type     default
========        ====    ====     =======
aleph           al      number   MS-DOS: 128, 224 otherwise
allowrevins     ari     boolean  off
altkeymap       akm     boolean  off
autoindent      ai      boolean  off
autoread        ar      boolean  off
autowrite       aw      boolean  off
autowriteall    awa     boolean  off
background      bg      string   "dark" or "light"
backspace       bs      string   ""
backup          bk      boolean  off
backupcopy      bkc     string   Vi (for Unix): "yes", otherwise: "auto"
backupdir       bdir    string   Amiga: ".,t:", MS-DOS+Win32: ".,c:/tmp,c:/temp";
Unix: ".,~/tmp,~/"
backupext       bex     string   "~", VMS: "_"
backupskip      bsk     string   "$TMPDIR/*,/tmp/*"
balloondelay    bdlay   number   600
balooneval      beval   boolean  off
binary          bin     boolean  off
bioskey         biosk   boolean  on
bomb            ---     boolean  off
breakat         brk     string   " ^I!@*-+;:,./?"
browsedir       bsdir   string   "last"
bufhidden       bh      string   ""
buflisted       bl      boolean  on
buftype         bt      string   ""
cdpath          cd      string   equivalent to $CDPATH or ",,"
cedit           ---     string   Vi: "", Vim: CTRL-F
encoding        enc     string   "latin1" or value from $LANG
charconvert     ccv     string   ""
cindent         cin     boolean  off
cinkeys         cink    string   "0{,0},:,0#,!^F,o,O,e"
cinoptions      cino    string   ""
cinwords        cinw    string   "if,else,while,do,for,switch"
clipboard       cb      string   "autoselect,exclude:cons\|linux"
cmdheight       ch      number   1
cmdwinheight    cwh     number   7
columns         co      number   80 or terminal width
comments        com     string   "s1:/*,mb:*,ex:*/,://,b:#,:%,:XCOMM,n:>,fb:-"
commentstring   cms     string   "/*%s*/"
compatible      cp      boolean  on, off when a .vimrc file is found
complete        cpt     string   ".,w,b,u,t,i"
confirm         cf      boolean  off
conskey         consk   boolean  off
cpoptions       cpo     string   Vim: "aABceFs", Vi: all flags
cscopepathcomp  cspc    number   0
cscopeprg       csprg   string   "cscope"
cscopetag       cst     boolean  off
cscopetagorder  csto    number   0
cscopeverbose   csverb  boolean  off
debug           ---     string   ""
define          def     string   "^#\s*define"
delcombine      deco    boolean  off
dictionary      dict    string   ""
diff            diff    boolean  off
diffexpr        dex     string   ""
diffopt         dip     string   "filler"
digraph         dg      boolean  off
directory       dir     string   Amiga: ".,t:"; MS-DOS+Win32: ".,c:\tmp,c:\temp";
```

```
Unix: ".,~/tmp,/var/tmp,/tmp"
display         dy      string   ""
edcompatible    ed      boolean  off
endofline       eol     boolean  on
equalalways     ea      boolean  on
eadirection     ead     string   "both"
equalprg        ep      string   ""
errorbells      eb      boolean  off
errorfile       ef      string   Amiga: "AztecC.Err"; others: "errors.err"
errorformat     efm     string   -> ":help errorformat"
esckeys         ek      boolean  (Vim default: on, Vi default: off
eventignore     ei      string   ""
expandtab       et      boolean  off
exrc            ex      boolean  off
fileencoding    fenc    string   ""
fileencodings   fencs   string   "ucs-bom", "ucs-bom,utf-8,latin1"
fileformat      ff      string   MS-DOS+MS-Windows+OS/2: "dos"; Unix: "unix";
Macintosh: "mac"
fileformats     ffs     string   ...
filetype        ft      string   ""
fillchars       fcs     string   "vert:|,fold:-"
fkmap           fk      boolean  off
foldclose       fcl     string   ""
foldcolumn      fdc     number   0
foldenable      fen     boolean  off
foldexpr        fde     string   "0"
foldignore      fdi     string   "#"
foldlevel       fdl     number   0
foldlevelstart  fdls    number   -1
foldmarker      fmr     string   "{{{,}}}"
foldmethod      fdm     string   "manual"
foldminlines    fml     number   1
foldnestmax     fdn     number   20
foldopen        fdo     string   "block,hor,mark,percent,quickfix,
foldtext        fdt     string   "foldtext()"
formatoptions   fo      string   Vim: "tcq", Vi: "vt"
formatprg       fp      string   ""
gdefault        gd      boolean  off
grepformat      gfm     string   "%f:%l%m,%f  %l%m"
grepprg         gp      string   "grep -n ", Win32: "findstr /n",
guicursor       gcr     string   "n-v-c:block-Cursor/lCursor,
guifont         gfn     string   ""
guifontset      gfs     string   ""
guifontwide     gfw     string   ""
guiheadroom     ghr     number   50
guioptions      go      string   OS-specific
guipty          ---     boolean  on
helpfile        hf      string   OS-specific
helpheight      hh      number   20
hidden          hid     boolean  off
highlight       hl      string   LONG string!
hlsearch        hls     boolean  off
history         hi      number   Vim: 20, Vi: 0
hkmap           hk      boolean  off
hkmapp          hkp     boolean  off
icon            icon    boolean  off, on when title can be restored
iconstring      ---     string   ""
ignorecase      ic      boolean  off
include         inc     string   "^#\s*include"
includeexpr     inex    string   ""
```

```
incsearch      is      boolean  off
indentexpr     inde    string   ""
indentkeys     indk    string   "0{,0},:,0#,!^F,o,O,e"
infercase      inf     boolean  off
insertmode     im      boolean  off
isfname        isf     string   OS-specific
isident        isi     string   OS-specific
iskeyword      isk     string   OS-specific
isprint        isp     string   OS-specific
joinspaces     js      boolean  on
key            key     string   ""
keymap         kmp     string   ""
keymodel       km      string   ""
keywordprg     kp      string   UNIX: "man"; DOS: ""; OS/2: "view"; VMS: "help"
langmap        lmap    string   ""
langmenu       lmenu   string   ""
laststatus     ls      number   1
lazyredraw     lz      boolean  off
linebreak      lbr     boolean  off
lines          lines   number   24 or terminal height
linespace      lsp     number   0, 1 for Win32 GUI
lisp           lisp    boolean  off
list           ---     boolean  off
listchars      lcs     string   "eol:$"
loadplugins    lpl     boolean  on
magic          ---     boolean  on
makeef         mef     string   OS-specific
makeprg        mp      string   "make", VMS: "MMS"
matchpairs     mps     string   "(:),{:},[:]"
matchtime      mat     number   5
maxfuncdepth   mfd     number   100
maxmapdepth    mmd     number   1000
maxmem         mm      number   between  256  to  5120 (system dependant)
maxmemtot      mmt     number   between 2048 and 10240 (system dependant)
menuitems      mis     number   25
modeline       ml      boolean  Vim: on, Vi: off
modelines      mls     number   5
modifiable     ma      boolean  on
modified       mod     boolean  off
more           ---     boolean  Vim: on, Vi: off
mouse          ---     string   "", "a" for GUI, MS-DOS and Win32
mousefocus     mousef  boolean  off
mousehide      mh      boolean  on
mousemodel     mousem  string   "extend", "popup" for MS-DOS and Win32
mouseshape     mouses  string   "i:beam,r:beam,s:updown,sd:cross,
mousetime      mouset  number   500
nrformats      nf      string   "octal,hex"
number         nu      boolean  off
osfiletype     oft     string   RISC-OS: "Text", others: ""
paragraphs     para    string   "IPLPPPQPP LIpplpipbp"
paste          ---     boolean  off
pastetoggle    pt      string   ""
patchexpr      pex     string   ""
patchmode      pm      string   ""
path           pa      string   Unix: ".,/usr/include,,"; OS/2:
".,/emx/include,,"; others: ".,,"
previewheight  pvh     number   12
previewwindow  pvw     boolean  off
readonly       ro      boolean  off
remap          ---     boolean  on
```

```
report          ---     number   2
restorescreen   rs      boolean  on
revins          ri      boolean  off
rightleft       rl      boolean  off
ruler           ru      boolean  off
rulerformat     ruf     string   empty
runtimepath     rtp     string   OS-specific; LONG string!
scroll          scr     number   lines / 2
scrollbind      scb     boolean  off
scrolljump      sj      number   1
scrolloff       so      number   0
scrollopt       sbo     string   "ver,jump"
sections        sect    string   "SHNHH HUnhsh"
secure          ---     boolean  off
selection       sel     string   "inclusive"
selectmode      slm     string   ""
sessionoptions  ssop    string   "blank,buffers,curdir,folds,
shell           sh      string   $SHELL or "sh",
shellcmdflag    shcf    string   "-c", MS-DOS and Win32, when shell
shellpipe       sp      string   ">", "| tee", "|& tee" or "2>&1| tee"
shellquote      shq     string   ""; MS-DOS+Win32: "\""
shellredir      srr     string   ">", ">&" or ">%s 2>&1"
shellslash      ssl     boolean  off
shelltype       st      number   0
shellxquote     sxq     string   "";
shiftround      sr      boolean  off
shiftwidth      sw      number   8
shortmess       shm     string   Vim: "filnxtToO", Vi: ""
shortname       sn      boolean  off
showbreak       sbr     string   ""
showcmd         sc      boolean  Vim: on, off for Unix; Vi: off
showfulltag     sft     boolean  off
showmatch       sm      boolean  off
showmode        smd     boolean  Vim: on, Vi: off
sidescroll      ss      number   0
sidescrolloff   siso    number   0
smartcase       scs     boolean  off
smartindent     si      boolean  off
smarttab        sta     boolean  off
softtabstop     sts     number   0
splitbelow      sb      boolean  off
splitright      spr     boolean  off
startofline     sol     boolean  on
statusline      stl     string   empty
suffixes        su      string   ".bak,~,.o,.h,.info,.swp,.obj"
suffixesadd     sua     string   ""
swapfile        swf     boolean  on
swapsync        sws     string   "fsync"
switchbuf       swb     string   ""
syntax          syn     string   emtpy
tabstop         ts      number   8
tagbsearch      tbs     boolean  on
taglength       tl      number   0
tagrelative     tr      boolean  Vim: on, Vi: off
tags            tag     string   "./tags,tags", when compiled with
tagstack        tgst    boolean  on
term            ---     string   -> ":help term"
termencoding    tenc    string   ""
terse           ---     boolean  off
textauto        ta      boolean  Vim: on, Vi: off
```

textmode	tx	boolean	MS-DOS+Win32+OS/2: on, others: off
textwidth	tw	number	0
thesaurus	tsr	string	""
tildeop	top	boolean	off
timeout	to	boolean	on
ttimeout	---	boolean	off
timeoutlen	tm	number	1000
ttimeoutlen	ttm	number	-1
title	---	boolean	off, on when title can be restored
titlelen	---	number	85
titleold	---	string	"Thanks for flying Vim"
titlestring	---	string	""
toolbar	tb	string	"icons,tooltips"
ttybuiltin	tbi	boolean	on
ttyfast	tf	boolean	off, on when term is xterm, hpterm,
ttymouse	ttym	string	depends on term
ttyscroll	tsl	number	999
ttytype	tty	string	from $TERM
undolevels	ul	number	100, 1000 for Unix, VMS,
updatecount	uc	number	200
updatetime	ut	number	4000
verbose	vbs	number	0
viewdir	vdir	string	for Amiga, MS-DOS, OS/2 and Win32:
viewoptions	vop	string	"folds,options,cursor"
viminfo	vi	string	-> :help viminfo
virtualedit	ve	string	""
visualbell	vb	boolean	off
warn	---	boolean	on
weirdinvert	wiv	boolean	off
whichwrap	ww	string	Vim: "b,s", Vi default: ""
wildchar	wc	number	Vim: <Tab>, Vi default: CTRL-E
wildcharm	wcm	number	none (0)
wildignore	wig	string	""
wildmenu	wmnu	boolean	off
wildmode	wim	string	Vim: "full"
winaltkeys	wak	string	"menu"
winheight	wh	number	1
winminheight	wmh	number	1
winminwidth	wmw	number	1
winwidth	wiw	number	20
wrap	---	boolean	on
wrapmargin	wm	number	0
wrapscan	ws	boolean	on
write	---	boolean	on
writeany	wa	boolean	off
writebackup	wb	boolean	on - with |+writebackup| feature, off
writedelay	wd	number	0

¡¡¡¡¼Ç:Sven, Bram, Ë×ÏÊ¸ÐÐ»Ä½Ï»½ÔËÜÖâ´Ï×¨·Ä, ÄãÄÇÏÊ×ÖÏÖ½éËÜÖ»ÏÄºÄÄð?
Bram:ÎÒ³öËúÓÚ1961Äê, ÎÒÊÇÒ»¸ö×¨ÒµµÄ¼ÆËã»ú¼¼Ê¦.
ÔÚDelft¼¼Êõ´óÑ§ÐÞÍêµç×ÓÑ§Ö®ºóÍÒµÄ´ó²¿·Ö¹¤×÷¶¼ÔÚÑÐ¾¿º£ÑóÉúÎïÑÕÉ«ÊÔÑéµÄ¸´ÖÆºÍ´òÓ¡Éè±¸.
ÎÒÊ¹ÓÃ¼ÆËã»úÑÓÐÏàµ±³¤µÄÊ±¼äÄË.
ÎÒÔÚÊÐ³¡ÉÏ»¹ÔÚ´óÂô4KµÄRAMÊ±½¨ÒìÁËÎÒµÄµÚÒ»Ì¨¼ÆËã»ú. ºóÀ´ÎÒ´Óº¸½Ó×ªµ½ÁË±à³ÌÉÏÀ´,
ÎÒ»¹ÊÇºÜÏ²»¶µç×ÓÑ§, Ö»ÊÇÃ»ÓÐÏóÒÔÇ°ÄÇÑù»¨·ÑÄÇÃ´¶àµÄÊ±¼äÁË.

¡¡¡¡Ä¿Ç°ÎÒ×¡ÔUVenlo, ºÉÀ¼µÄÒ»¸öÐ¡³ÇÊÐ, ¹¤×÷Ö®ÓàÎÒÏ²»¶ÌýÒôÀÖ. Âó¿Ë·çÊÇÎÒµÄ×î°®.
µ«³ý´ËÖ®ÍâÎÒÃ»ÓÐ¸ü¶àµÄÒµÓà°®ºÃÁË, Ò²²»ÐèÒªÍÐ¼Ò´ú¿Ú,
ËùÒÔÎÒÓÐºÜ³äÔ£µÄÊ±¼ä»¨ÔÚ±à³ÌÉÏÃæ, ÎÒÏ²»¶±à³Ì. Ã¿ÄêÎÒ´ó¸ÅÃÃÓÎÒ»Á½´Î,
Í¨³£ÊÇµ½Ò»¸öÓÐ×ÅÍêÈ«Ä°ÉúµÄÎÄ»¯±³¾°µÄ¹ú¼ÒÈ¥,
ÕâÑù¿ÉÒÔÁË½âÊÀ½çÉÏ²»Í¬µØ·½µÄËËÃÇÊÇÕÑùÉú»îµÄ, Ò²¿ÉÒÔÍØ¿í×Ô¼ºµÄÊÓÒ°.

¡¡¡¡Sven:ÎÒ1967ÄêÉúÓÚµÂ¹ú°ØÁÖ. ÎÒÒ»Ö±ÔÚ°ØÁÖÉú»î,
ÎÒÄ¿Ç°ÔÚ´ÓÊÂÊýÑ§ºÍ¼ÆËã»ú¿ÆÑ§µÄÑÐ¾¿. 1989ÄêÎÒÍ¨¹ýEmailÁË½âµ½Internet,
1992ÄêÎÒ¿ªÊ¼Ê¹ÓÃÐÂÎÅ×é,
ÎÒ¾-³£ÓÃËü½øÐÐÍ¨Ñ¶.Web³öÏÖºóÎÒ¿ªÊ¼ÔÚÍøÒ³ÉÏÕûÀíÎÒ¹ýÈ¥Ê¹ÓÃ¹ýµÄÕâÐ©Èí¼þ

¡¡¡¡ÎÒÔø¾-Íæ¹ý¼ªËü, ²»¹ý×Ô´Ó¿ªÊ¼Î¬»¤ÕâÐ©³ÌÐò(elm, lynx, mutt, nn, screen, slrn,
vim, zsh)µÄÍøÒ³ºó, Ò²¹Ë²»ÉÏËüÁË. ÕâÐ©³ÌÐò¶¼ÓÐÒ»¸ö×Ô·û½çÃæ.
ËùÒÔËüÃÇ¿ÉÒÔÔÚÊ®·ÖÆÕÍ¨µÄ×Ö·ûÕÕ¶ËÉÎÔËÐÐ. Ò»ÐèÒªÍ¨¹ý
telnetÄÜ½øÐÐ»¥½Ó¾ÍÐÐ(Èç¹ûÃãÔ¸Òâ, Ò²¿ÉÒÔÊ¹ÓÃssh).

¡¡¡¡1999Äê9ÔÂÎÒÕÕÓÚÈ¥ÁËÃÀ¹ú, ÄÇ´ÎÊÇÎÒµ½¼ÓÀû£ÄáÑÇÂÃÓÎ,
ÔÚÄÇÀïÎÒ¼ûµ½ÁËÎÒÈÏÊ¶ÒÑ¾ÃµÄÍøÓÑÃÇ.

¡¡¡¡ÎÒ´ÓÃ»¼ûµ½Bram±¾ÈË, ËùÒÔÎÒÕæµÃ¿´Ò»¿´ËûµÄÃÕÕÆ¬²ÅÖªµÀËû³¤Ê²Ã´Ñù.

¡¡¡¡¼Ç:ÄãÊ²Ã´Ê±ºî¿ªÊ¼Ð´vim, ÊÇÊ²Ã´ÙÊ¹ÄãÏëÐ´Ò»¸öÕâÑùµÄÈí¼þ,
ÄÜ¸úÎÒÁÄÁÄvimµÄ·¢Õ¹Ê·Âð, ¿ÉÒÔÊµvimÊÇ×îºÃµÄÂð?

¡¡¡¡Bram:ÎÒÔÚ1989Äê¿ªÊ¼Ð´vim, ÄÇÊ±ÎÒ¸ÕÂòÁËÒ»Ì¨×Ô¼ºµÄ¼ÆËã»ú, Amiga2000
ÄÇÊ±ÎÒÓÃµÄÊÇvi, ÎÒÏëÓÃÒ»¸öÓëviÀàËÆµÄ¸ü ºÃÕ»µãµÄ±à¼-Æ÷,
ÕÔµ½µÄ¼¸¸öviµÄ¿ËÂ¡°æ¶¼²»³ÆÐÄ, ËùÒÔÎÒÓÃStevieµÄÔ´´úÂë, ¿ªÊ¼Ò»µÃÒ»µãµØÐÞ¸Ä,
Ìí¼ÓÒ»Ð©ÃüÁî.

¡¡¡¡µÚÒ»¸ö¿ÉÓÃµÄ°æ±¾ºÜ¿ì¾Í³öÀ´ÁË, 1991ÄêÎÒ°ÑËü·¢²¼³öÈ¥, ÕâÒ»ÏÂÕÐÀ´ÁËºÜ¶àÈËµÄ»ØÓ¦,
ËûÃÇ¹ÄÀøÎÒ°ÑËüÔÙÔö²¹Ò»Ð©¹¦ÄÜ, Ò²¾ÍÊÇ´ÓÄÇÊ±"Vi Imitation" ¸üÃûÎª"Vi IMproved",
ÎÒÒ²Ò»Ö±ÔÚÍùÉÏÃæÌí¼ÓÐÂµÄ¹¦ÄÜ.

¡¡¡¡Sven:1992ÄêÎÒ¾ÍÔÚÕÒÒ»¸ö¿ÉÒÔÔÚÎÒµÄMacintosh IIvx»úÉÏÓÃµÄvi±à¼-Æ÷ .
ºóÀ´ÎÒÓÖ»Ø¹ýÍ·À´ÓÃÁËÒ»ÕõºÓDOS(Ô-Òò°ÜÃ÷ÍÔ), ËäÈ»ÎÒÓÐ¸ü¶àµÄÀíÓÉÏ²»¶ Macintosh,
µ«Ã»ÓÐMacintosh°æµÄvi.

¡¡¡¡1994ÄêÓÐËËÏòÎÒÍÆ¼öÁËvim-2, ÎÒ¾ª²ÓÚÖÕÓÚÓÐÈËÉ¿ªÊ¼Ôö²¹viµÄ¹¦ÄÜÁË.
ÈËÄÇÊ±ÆðÎÒ¾Í¿ªÊ¼Í¨¹ýÒ»¸öÖ÷Ò³ÖvimµÄ·¢Õ¹, Í¬Ê±Ï£ÍûÓÐÒ»Ìì°ÑËüÒ²ÒÆÖ²µ½
MacintoshÉÏÈ¥.

¡¡¡¡vim-3°æÖÕÓÚ±»ÒÆÖ²µ½MacintoshÉÏÊ±ÎÒ¸ßÐË»µÁË, ²»¹ýÄÇ¸ö°æ±¾ÓÐºÜ¶à BUG,
»¹ÓÐ¼¸¸öºÜÃüµÄÎÊÌâ.

¡¡¡¡vim-4°æÃ»ÓÐ³öMacintoshµÄ°æ±¾, ÕâÈÃÎÒÊµÔÚÊÇÊ§Íû.

¡¡¡¡ºóÀ´, Linux·¢Õ¹µÃÈç»ðÈç²è, ÎÒÓÖ×ªÏòÁËPC.

¡¡¡¡1997Äê9ÔÂÎÒ×¢²áËÉÓòÃûvim.org, ¿ÉÒÔ¸üºÃµØÖ§³ÖvimµÄ·¢Õ¹ÁË.

¡¡¡¡½ñÌìµÄvimÒÑ¾Ô¶·ÇÎô±ÈÁË, ºÜ¶àÆäËü±à¼-Æ÷µÄÓÅ¼ÌØÐÔÒ²Ó¦ÓÃ»§Ö®ÐèÍí¼Óµ½vimÀï,
vim¼¸ºõ¿ÉÒÔÔËÐÐÔÚÃ¿Ò»ÖÖÆ½Ì¨ÉÏ, ÓÖÓÐÁËÍ¼ÐÎ½çÃæ.
²»¹ýMacintosh°æµÄÒÆÖ²ÎÊÌâ»¹ÊÇÈÃÎÒ...

¡¡¡¡¼Ç:ÄãÆ½¾ùÃ¿ÖÜ»¨¶àÉÙÊ±¼äÔÚvim/vimÍøÒ³ÉÏÎ¬»¤ÉÏ, »¨¶àÉÙÊ±¼äÈ¥»ØEMAIL ,
»Ø¸²comp.editorsÐÂÎÅ×é, ÄãÔõÑù°²ÅÅ»¨·ÑÔÚÃãµÄ¹¤×÷ÈÎÎñºÍvimËÏµÄÊ±¼ä

¡¡¡¡Bram:ÎÒ»¨ÔÚÔÄ¶ÁºÍ»Ø¸²vimµÄÐÂÎÅÁÐ±íÉÏµÄÊ±¼äÌ«¶àÁË, ÓÐÊ±ºîÎÒÕæÏë²»ÜËüÁË,
µ«ÎÒ¿´²»µÃÄÇÐ©ÓÊ¼þÃ»ÈË»Ø¸². ÏÖÔÚ»¹¿ÈÒÔ, ¹¤×÷Ö®ÍâÎÒµÄÊ±¼ä»¹Ü³äÔ£ ,
²»¹ýÎÒ»¹ÊÇ¾-³£ÔÚ°ëÒ¹Àï»ØÄÇÀïÓÊ¼þ.

¡¡¡¡ÎÒ¾ö¶¨²»ÔÚcomp.editorsÉÏ»î¶¯ÁË, ÄÇ¸öÌ«ÀË·ÑÊ±¼äÁË,
ÐÒºÄÓÐSvenºÍÆäËûÒ»Ð©ËË¼°Ê±»Ø´ðËËÃÇÍáµÄÎÊÌâ.

¡¡¡¡ÎÒ¼¸ºõÃ¿Ìì¶¼ÒªÎªvim»¨µãÊ±¼ä, ÓÐÊ±Ö»ÊÇÐÞ¸ÄÒ»µãÐ¡ÎÊÌâ,
ÓÐÊ±¾ÍÒª´óµ¶À«¸µØ×öÒ»Ð©´ó¶¯×÷ÁË. Ö»ÓÐÎÒ²»ÔÚ¼ÒÊ±ÎÒ²Å¿ÉÄÜÒ»Ìì²»Åövim.

¡¡¡¡ÒªÊÇÎÕÕÁË·Ý±ðµÄ¹¤×÷, ¾ÍºÍÄÑ³é¿ÕÀ´×öÕâÐ©ÁË, ÎÒ¿ÖÅÂÖ»ÄÜÔÚÜÄ©×öÕâÐ©ÁË,

Ò²ÕýËÇÓòÏªÖâÖÙËÏÄæ¸ öÏO²ÅÄ»Ë¥ÓÒ¹¤×÷.

¡¡¡¡Sven:ÎÒ¼¸ºõËùÓÐµÄÊ±¼ä¶¼»¨ÔÚvimÉÏ,
´ó²¿·ÖÊ±¼ä¶¼ÔÚ»Ø´ðÎÊÌâ(comp.editorsÉÏµÄÌù×Ó, Ö±½Ó·¢¸øÎÒµÄÓÊ¼þ,
»òÕßÊÇ·¢ÔÚvimµÄÓÊ¼þÁÐ±íÉÏµÄÓÊ¼þ.)ºÍÎ¬»¤ www.vim.orgµÄÍøÕ¾ÉÏ.

¡¡¡¡Ä¿Ç°ÎªÖ¹ÎÒ»¹ºÜÉÙÙÚµÂ¹úµÄvimÕ¾µãÉÏÌ¸ÂÛvim, µ«¾-³£¸úÒ»Ð©ÒÑ¾-ÊìÖª vimµÄÈËËµÆð¹ý.
ÎÒÏ£ÍûÄÜ¾¡Á¦Îª´óÖÚ¶ÁÕßÐ´Ò»µãÎÄµµ. ÍøÒ³ÉÏÒÑ¾-ÓÐÁÉÒ»Ð©ÔâÑùÆ¬¶ÏµÄ°ïÖúÎÄµµ.
µ«Ð´ÕâÐ©¶«Î÷ÊµÔÚÌ«»¨Ê±¼äÁË.

¡¡¡¡Ã¿Ã¿ÈÇ´Ë, ÎÒ¶¼Ï£ÍûÄÜ´´½¨Ò»¸öÈ«ÃæµÄ°ïÖúÎÄµµÊý¾Ý¿â, ÄÚÈÝ°üÀ¨¹ýÈ¥µÄ VIÎÄµµ,
ÓÊ¼þÁÐ±í, ÐÂÎÅ×éºÍÆäËü¾¡¿ÉÄÜ¶àµÄÄÚÈÝ.
ÒÔÒ»ÖÖÒÔÊºÏÑÕÉ«ÏÔÊ¾µÄ¿É¶ÁÐÔ½ÏÇ¿µÄÐÍÊ½Õ¹ÏÖ¸ø¶ÁÕß. µ«, ÈÔÊÇÀÏÎÊÌâ,
Õâ²»Öªµ¼ÒªÀ¶à¾ÃÃ...

¡¡¡¡¼Ç:ÔÚVIMµÄ±à³Ì·½Ãæ, ÓÉË-À´¿ª·¢´úÂë, Ò»¹²ÓÐ¶àÉÙ³ÌÐòÔ±, Ë-À´Ð-µ÷ÏîÄ¿µÄ·¢Õ¹,
ÄãÍ¨³£ÊÇÈçºÎ¾ö¶¨¸÷¸ö°æ±¾µÄ·¢ÐÐÈÕÆÚµÄ? ÄãÊÇÈçºÎ×öµ¼ÈÃÀ´×Ô·ÖÉ¢¿ª·¢µÄ´úÂëÐ-Í¬ÆðÅ´µÄ?
ÊÇ²»ÊÇÈÎºÎÈË¶¼¿ÉÒÔ¼ÓÈë¿ª·¢Ð¡×é? ÔÚ¶Ô´ýÐÂµÄº¯Êý¿âÈçGTK+Í¼ÐÎ¿â·½Ãæ, ÄãÈçºÎ¾ö¶¨È¡Éá?

¡¡¡¡Bram: ÎÒ×Ô¼º¸ºÔð´ó²¿·ÖºËÐÄµÄ²¿·Ö, µ«³¤ÆÚÒÔÀ´Ò²ÓÐÆäËû¼¸¸öÖ²ªµÄÖ¾Ô¸ÕßºïÖú¿ª·¢.
Í¼ÐÎÓÃ»§½çÃæ²¿·ÖÓÉRobert Webb¿ª·¢, ÆäËüºÜ¶àÆ£¿éÓÉ±ðËÈÀ´¿ª·¢,
Í¨³£ÕâÐ©ÈË¶¼ÊÇÔÚÉÏÃæ¶ÌÆÚ¿ª·¢Ò»ÕóºÍÖÐÖ¹ÁË,
ÎÒ¾ÍÒªÊ¹ÕâÐ©¶Ï¶ÏÐøÐøµÄ¹¤×÷ÄÜµÃµ½³ÖÐøµÄ½øÕ¹, ±£Ö¤ÕÔ¸öÏîÄ¿ÓÐÒ»¸öÓÿÈ·µÄ·¢Õ¹·½Ïò.
´ËÍâ, »¹ÒªÎ¬»¤ÆäËüÐ¡×é³ÉÔ±¿ª·¢³öµÄ´úÂë, ÓÉÓÚvimÖËÐÐÔÚÈç´Ë¶àµÄÆ½Ì¨ÉÏ,
ÕâÊµÔÚÊÇÒ»¼þ¿à²î.

¡¡¡¡¾-³£ÓÐÈË·¢ËÍ¸øÁËÒ»¸ö²¹¶¡°ü, ÓÉÎÒÀ´¾ö¶¨Òª²»Òª°ÑËü¼Ó½øÈ¥,
ÆäÊµºÜÃÑ¾Ü¾ø±ðÈËµÄ²¹¶¡°ü, ÒòÎªÕâ¶¼·Ó³ÁËÓÃ»§±¾ÈË¼±ÇÐÏ£ÍÚ¿´µ½µÄÌØÐÔ.
µ«Í¬Ê±Ò²ÓÐÈË±§Ô¹Ëµvim±äµÄÔ½À´Ô½´óÁË. ÎÒ¾¡Á¦ÒªºÚÚÕâÁ½ÕßÖ®¼ä±£³ÖÆ½ºâ.

¡¡¡¡Í¨³£vimµÄ¿ª·¢ÎÒËµÁËËã. ÇãÌýÈËÃÇ¶ÔvimµÄÆÚÍû, È»ºóÚÓÃ»§ÐèÇóµÄ»ù´¡ÉÏ×ö³ö¾ö¶¨.
ÓÐÊ±ÓÃ»§ËùÒªÇóµÄ¶«Î÷ºÍÊûÃÇÕæÕýÐèÒªµÄ¶«Î÷ÍùÍù²»ÊÇÒ»»ØÊÂ,
ÕâÊ±¾ÍÒªÏ¸ÐÄ·ÖÎöËûÃÇÊùÌá³öµÄÒªÇó, ·¢ÏÖÕâ±³ºóÞþ²Ø×ÅµÄÓÃ»§ÕæÕýµÄÐèÇó.
»òÐÍÓÐÐ©¶«Î÷ÔÚ±à³ÌÊµÏÖÆðÀ´¾ÍÃ»ÄÇÁ´ÈÝÒ×, ÕâÒ²´ÙÊ¹ÎÒÒ»Ö±¶¼ÔÚÊ¹ÓÃvim,
ÕâÑùÎÒ²ÅÇ×ÉíÌå»áÓÃ»§¶ÔËüµÄÆÚÍûºÍÒªÇó.

¡¡¡¡Ô-ÔòÉÏÀ´Ëµ, Ã¿¸öÈË¶¼¿ÉÒÔ°Ñ²¹¶¡·¢¸øÎÒ.
Èç¹ûÎÒ¾õµÃÓÐÐ±ØÒª¾Í»á°ÑËü¼ÓÊËë½vimµÄÕýÊ½°æ±¾ÖÐÀ´,
ÕâÒ²¿´·¢²¹¶¡µÄÈËÒªÇóµÄµ½µ×ÊÇÊ²Ã´, ÁíÍâ, ²¹¶¡±¾ÉíµÄÖÊÁ¿. µ«ÊÇvimÒÑ¾-±äµÃºÜÅÓ´óÁË,
ËùÒÔÎÒ¸üÒªÏ¸ÐÄÕçÑ¡.

¡¡¡¡GTK¿âµÄÖÆÖ²ÒÑ¾-±»¼ÓÈë, ÒòÎªËüÔËÐÐµÃ²»´í, ²¢ÇÒ±ÈAthena»òMotif¶¼ÒªºÃ, ÁíÍâ,
Ëü»¹ÊÇ×ÔÓÉÈí¼þ. Í·Í´µÄÊÇ×î³õµÄÖÆÖ²ÕßÃÞ»ÓÐÊ±¼äÈ¥ÐÞ¸ÄBUG,
ÕâÑùÎÒÃÇ¾ÍÒªª»¤Á¦Æø¶»¤ËüµÄÎÈ¶¨ÐÔ, ºÃÔÚÏÖÔÚÚÒÑ¾-ÔÊÐÐµÄÏàµ±²»´íÁË.
ÏÂ´ÎÔÙÓÐÕâÑù´óµÄ¸Ä¶¯ÎÒ¿ÉÃ¼Ó±¶É÷ÖØÁË. ÒòÎª¼ÓÈëÒ»¸ö²»ÄÜÔËÐÐµÄÌØÐÔÓÐº¦ÎÞÒæ.

¡¡¡¡Sven: ÏÖÔÚvimÊÇÖÚ¶à³ÌÐòÔ±ÐÄÑªµÄ½á¾§. µ«BramÆðµ½ÁË¾ö¶¨ÐÔµÄ×÷ÓÃ.
BramµÄ¹¤×÷Ê®·Ö³öÉ«. ÎãÓ¹ÖÃÒÉ, BramÊÇÒ»¸ö¾-Ñé·á¸»µÄ³ÌÐòÔ±,
²¢ÇÒëûÒ»Ö±¶¼ÔÚÊ¹ÓÃÕâ¸ö³ÌÐò, ÕâÊ¹µÃËû¶ÔvimÁËÈçÔ¸ÕÆ.

¡¡¡¡ÎÒÊµÔÚÅå·þËûµÄÔðÈÎÐÄºÍ¶Ô¸÷ÖÖÕù¶ËµÄ°ÑÎÕÄÚ¦ - ¼´Ê¹ÊÇÒ»³¡¼¤ÁÒµÄÕùÂÛ .
ÎÒÖÔÐÄÏ£ÍûËûÄÜÎª×Ô¼ºµÄÅ¬Á¦µÃµ½Ó¦ÓÐµÄ±¨³ê, ºÃÈÃËûÄÜÏóÏÖÔÚÔâÑùÏ¸ÐÄºÇ»¤ vimµÄ³É³¤.

¡¡¡¡µ«Õâ¶Ô¿ªÔ´Èí¼þµÄ¹±Ï×ÕßÃÇÀ´Ëµ»¹ÊÇÒ»¼þÃ»Ó°µÄÊÂ - µ½ÄÀÀïÈ¥ÕÒÕâÑùµÄ±¨³êÄØ?
Ò²ÐíÍólinuxfind.orgÕâÑùµÄÕ¾µã»áÌá¹©Ô»Ð©ÐÞÕú.

¡¡¡¡¼Ç: ÏÖÔÚµÄvim´úÂë¿ÉÒÔ¹¤×÷ÔÚ¸÷ÖÖÆ½Ì¨ÉÏ, ËüµÄÏåÏµ½á¹¹ÊÇÈçºÎÉè¼ÆµÄ?
ÄÜ·ñÈÃËüºÜÈÝÒ×µØÔËÐÐÔÚÒ»ÖÖÐÂµÄÆ½Ì¨ÉÏ?

¡¡¡¡Bram: vimµÄÏåÏµ½á¹¹Ò»Ö±ÔÚ²»¶ÏµÄ¸¸ïÐÂÖ®ÐÐ.
ÓÐÊ±Òª¼ÓËëÒ»¸öÐÂÄÌØÐÔ²ÐèÒª¸Ä¶¯Ì«¶àµÄ´úÂë. Èç¹ûÐÂ¼ÓËëµÄÌØÐÔÈÅÅÒÁËÕû¸öÏîÄ¿,
ÄÇÎÒ¾ÍÒª·ÑÒ»·¬ÐÄËº°ÃºÃÏëÏëÕýÈ·µÄ½â¾ö·½°¸¸¸ÁË. ÏÖÔÚvimÊÇÒ»¸ö°Ü´óµÄ³ÌÐòÁË.
ÕâÑù×ö¿É²»ÈÝÒ×.

¡¡¡¡Ã¿Ò»´ÎÐÞ¸Ä¶¿ÉÄÜÒýÈëÐÂÎÊÄBUG, Ã¿²»ÐÒ, ÕâËµÃ÷ÎÒÃÇµ±Ç°µÄÌåÏµ½á¹¹²¢²»ÊÇÊ®·ÖÀíÏë.
Èç¹ûÓÐÊ±¼äÎÒ¿ÉÄÜÔÚ6.0°æÖÐ×ö°Ü¶àµÄ¸Ä¶¯.

¡¡¡¡Ôö¼ÓÒ»¸öÐÂÆ½Ì¨µÄÖ§³ÖÓ¦¸Ã²»»áÌ«ÄÑ. Èç¹ûÄãÓÐÕâÑùÒ»¸öÆ½Ì¨, ²»·ÁÄÃÀ´ÊÔÊÔ¿´.
ºÜ¶àUNIXÀàÆ½Ì¨¶¼¿ÉÒÔÒ»ÊÔ, µ«ÊÇÍóVMSÕâÑùµÄÏµÍ³¾ÍºÜÄÑÖ§³ÖÁË. vim
ÔÚUNIXÉÏµÄ±³¾°²¢»ÌØ±ðÊÔÃÓÚVMS. Êµ¼ÊÉÏvimÕæÕýµÄ³öÉúµØÊÇAmiga, µ«ÄÇ¶Ô
UNIXÆ½Ì¨µÄÆÖ²À´Ëµ²»ÊÇÄÑÊÂ.

¡¡¡¡Sven: ¿ª·¢Ð¡×éµÄ³ÌÐòÔ±ÃÇ¶¼ÖªµÀvimµÄÔ´úÂëÊ®·Ö³öÉ«.
µ«ÊÇÃ¿Ôö¼ÓÒ»¸öÐÂÌØÐÔ¶¼»áÔö¼ÓÏµÍ³µÄ¸´ÔÓ¶Ô. Õâ¾ÍÊ¹µÃºÜÄÑÎª´ËÐ´³ö²¹¶¡.

¡¡¡¡Á¿´ÏÏÖÐÅÏ²ÓÙvimÖÖ¼ÖÉëÅËÐÄGUIÏØÐÖ¶ø¸üËÜ»¶Ö-Ë±,
ÕâÐ©ÐÂµÄGUIÌØÐÖ¶¼»áÏûºÄÒ»´ó¶Ñ²¹¶¡. ²ÉÓÃÒ»ÖÖÐÂµÄGUI×ÜÊÇÌØ±ð·ÑÊÂ,
µ±È»¿ª·¢ÆðÀ´Ò²¸ü·ÑÊ±.

¡¡¡¡ÓÐÊ±ÉõÖÁÔö¼ÓÒ»ÖÖÐÂÆ½Ì¨µÄÖ§³Ö¶¼±ÈÔö¼ÓÁíÍâÒ»¸öGUIÀ´µÃÈÝÒ×...

¡¡¡¡µ«ÊÇ²»¹ÜÔõÃ´À´ËµvimµÄ·¢Õ¹»¹ÊÇÊ®·ÖÑ¸ÈÙµÄ - Ïà¶ÔÓÚ¸ú²»ÉÏ²½µÄÎÄµÀ´Ëµ.
ÓÐ±ØÒªÔö¼Ó¸ü¶àµÄÎÄµÀ´ËµÃ÷ÈçºÎ³ä·ÖµØÊ¹ÓÃvim.

¡¡¡¡¼Ç: vimÖ»ÄÜÓÃÓÚÓ¢Óï. ÓÐÃ»ÓÐÈÃËü¹ú¼Ê»¯µÄ·½°¸?

¡¡¡¡Bram: ³ÌÐò±¾Éí²¢Ã»ÓÐ¼Æ»®Òª¹ú¼Ê»¯. ÄÇÌ«¼è»¤ÁË. Ö»ÊÇ·-ÒëÕâÐ©ÎÄµ¾Í¹»ÇºÁË.
µ«ÊÇÏÖÔÚÕýÔÚ¼Æ»®ÈÃËü¿ÉÒÔ±à¼¶àÖÖ±àÂë·½°¸µÄÎÄ±¾.

¡¡¡¡ÏÖÔÚÒÑ¾-¿ÉÒÔÖ§³ÖÏ£²¨À³ÎÄºÍ²¨Ë¿ÎÁÁËÒÔ¼°Ò»Ð©ÑÇÖÞÓïÏµµÄ±àÂëÁË. ¶ÔÎÒÃÇÀ´Ëµ,
´ÓÓÒµ½ó½øØÐÔÄ¶Á, ¿´×ÅÐÂ²åËëµÄ×Ö·ûÊÇÍù°óµ¹µÄÕæÊÇ¾õµÃºÃÐ¦. ÔÚ6.0
°æ±¾ÖÐÎÒÃÇ¼Æ»®Òª¼ÓËëUTF-8¸ñÊ½(Ê²Ã´¶«¶«?), ÏÖÔÚÕâÖÖ¸ñÊ½µÄÊ¹ÓÃÔ¶±ÈÒÔÇ°ÒªÀ´µÃÆÕ±éÁË.

¡¡¡¡Sven: ÊÇµÄ, Ôö¼Ó¶ÔUTF-8µÄÖ§³ÖÊÇÒ»¼þºÜÓÐÐÒâË¼µÄÊÂ, Õâ¸öÄ¿±êÓ¦¸ÃÔÚ²»¾Ã¼´¿É´ï³É.

¡¡¡¡ÎÒÏ£Íûvim»á¼ÓÈë¹ú¼Ê»¯µÄÏûI¢ÏÔÊ¾, ¶øÇÒ×îºÃÓÐÐÈËÀ´·-ÒëÔÚÏß°ïÖúÏµÍ³.
²¢ÇÒÄÜ¼°Ê±¸ú×Ù±ä»¯µÄ²½·¥. (ÔÝ²»¿¼ÂÇ´ÓASCII×Ö·ûµ½ÆäËü×Ö·û¼¯´øÀ´µÃÎÊÎâ)

¡¡¡¡ÎÒÃÇÒÑ¾-ÔÚÎª°ïÖúÎÄµµµÄ·-Òë×ö³öÅ¬Á¦ÁË. »¶Ó-´ó¼ÒÓ»Ô¾Ö§³Ö -
µ«ÊÇÎÒÊÇÒ»Ïî¼è¿àµÄÈÎÎñ.

¡¡¡¡¼Ç: ÄãÔÚvimÉÏËù×öµÄ¹¤×÷ÓÐÃ»ÓÐ¸øÃãÔö¼Ó¹ý¹¤×÷µÄ»ú»á? vimÈçºÎÓ°ÏìÄãµÄÉú»î?

¡¡¡¡Bram: Ä¿Ç°Îª¹»¹Ã»ÓÐÈËÒòÎªvim¸øÎÒÌá¹©Ò»·Ý¹¤×÷.
µ«ÊÇÎÒµÄÈ·ÓëÊÀ½ç¸÷µØµÄÈËÃÇ±£³Ö×ÅÁª¡Ïµ. Ò²ÐíÕâ»áÓÐÖúÓÚÎÒÕÒµ½Ò»·Ý²»´íµÄ¹¤×÷°É.

¡¡¡¡ÓëÊÀ½ç¸÷µØµÄÈËÃÇ±£³ÖÁªÏµÊÇÒ»¼þºÜÓÐÈ¤µÄÊÂ. ËäÈ»Ö»ÊÇÍ¨¹ýe-mail.
¸÷ÖÖ²»Í¬µÄÎÄ»¯¾ÍÒþ²ØÔÚÃÇµÄ×ÓÀïÐÐ¼äÀï. e-mailÈÃÕâÒ»ÇÐ¶¼³ÉÎªÁË¿ÉÄÜ,
ÕâÕæÊÇÌ«²»¿ÉËï¼ÔéÁË. ¸Õ¿ªÊ¼Ê¹ÓÃ²»¾õµÃÔõÃ´Ñù. µ«ËûÄÜÈÃÈËÃÇ¿ÕÇ°µØ¾Û¼¯ÔÚÒ»Æð.

¡¡¡¡Sven: ¸úbramÒ»Ñù - ÎÒÒ²Ã»ÓÐÒòÎªvimµÃµ½¹ý¹¤×÷. ²»¹ý,
ÎÒÓÃEMAILÓëÈËÃÇËµÆðÎÒµÄ¹¤×÷Ê±, ËûÃÇ¶¼»áËµÄÇ²»³ÉÎÊÌâ - ÌØ±ðÊÇÈç¹ûÎÒÄÜµ½ÃÀ¹úµÄ°.

¡¡¡¡ºÜ¶àÓÃ»§ËµËûÃÇÏ£ÍûËûÃÇµÄ³ÌÐòÄÜÖ§³ÖÒ»ÖÖÍâ²¿µÄ±à¼-Æ÷ÕâÑùËûÃÇ¾Í¿ÉÒÔÊ¹ÓÃvim.
ÎÒ¶àÃ´Ï£ÍûÓÐÕâÑùµÄ¹«Ë¾ÄÜ²ÉÄÉÕâÑùµÄÒâ¼û.

¡¡¡¡Ë-ÖªµÀ - Ò²ÐíÓÐÒ»ÌìÄãÄÜÔÚÒ»¸ö´øÓÐÍâ¹Ò±à¼-Æ÷µÄ³ÌÐòÄÚÓÃvim±à¼-ÎÄ±¾.

¡¡¡¡¼Ç: vimÊÇ´ÈÉÆÈí¼þ, Õâ¾ÍÊÇËµÈËÈÈ¶¼¿ÉÒÔ×ÔÓÉµØÊ¹ÓÃËü,
µ«ÊÇ¹ÄÀø´ó¼ÒÎªÎÚ¸É´ïµÄ¹Â¶ùÃÇ×ö³öÉ¾ÙÐ. ÄÜÏêÏ¸Ì¸Ì¸Õâ¸öÂð?

¡¡¡¡Bram: ÕâÊÂÒª´ÓÎÒ×Ô¼ºµÄ¹ÊÊÂËµÆð.

¡¡¡¡ÎÒÒÑ¾-ºÜ¶à´ÎÀûÓÃ¼ÙÆÚµÄÊ±¼äµ½²»Í¬µÄ¹ú¼ÒÓëµ±µØµÄÈËÃÇ´ÔÚÚ»Æð.
ÕâÖÖ·½·¨ÈÃÎÒÁË½â¸ü¶àµÄÈËÃÇ.

¡¡¡¡ÎÒÓÚ1993ÄêÔÚÎÚ¸É´ï´ôÁËÒ»¸öÔÂ. ÄÇÊÇ¸öºÜÌØ±ðµÄµØ·½,
ÒòÎªÕâÊÇÎÒÃÇµÚÒ»´ÎÅöµ½ÎÒÃÇ¿ÉÒÔÇÐÇÐÊµÊµÐèÒªÎÒÃÇ°ïÖúµÄÈËÃÇ.
ÔÚÆäËüµØ·½ËÊÃÇ²¢²»áÕæÝÐèÒªÎÒÃÇÈ¥°ïÖúËûÃÇ. »ÕßÎÒÃÇÃ»°ì·¨È¥ÔõÃ´°ïÖúËûÃÇ.
µ«ÔÚÎÚ¸É´ï, ÄÇÀïµÄÈËÃÇÈ·ÊµÐèÒª°ïÖú. ÈýÖÜµÄÊ±¼äÄÚÎÒÃÇ³ÉÁ¢ÁËÒ»ËùÓ×¶ùÔ°.
ÄÇÊÇÒ»´ÎºÜÌØ±ðµÄ¾-Àú.

¡¡¡¡µÚ¶þÄêÎÒÓÖÈ¥ÁËÄÇÀïÒ»¸öÔÂ. ¿´µ½ÄÇÀïµÄº¢×ÓÃÇÔÚÐÂµÄÑ§Ð£ÀïÉÏÑ§.
ÕâÖÖÇé¾°´ó´óµØ´¥¯ÁËÎÒ, ÎÒ¾ö¶¨ÔÚÄÇÀï¹¤×÷ÉÏÒ»Äê.
Ò»ÄêÊ±¼äÀïÎÒÖÔÁ¦ÓÚÌá¸ßµ±µØÈËÃÇµÄÖúË®ÓëÒ½ÁÆéèÊ©µÄË®Æ½ÉÏ. Ò²¸ü¶àµØÁË½âÁËµ±µØµÄÈËÃÇ.

¡¡¡¡»Ø¼Òº´ó, ÎÒÏë¼ÌÐø°ïÖúËûÃÇ, ÓÚÊÇÎÒÚºÉÀ¾³ÉÁ¢ÁËICCF»ù½ð.
°ÑÎÒÔÚvimÉÏµÄ¹¤×÷¸úÔ¸ö½á°ÏÔÚÒ»¿é. ÎÒ·¢ÏÖÔâµÄÈ··¢»ÓÁËÐ§Ó¦. ÒòÎªvimµÄÔ-Òò,
Õâ¸öÏîÄ¿ÓÐÁË¸ü¶àµÄ¾èÔùÆ··. ÕâÆðµ½ÁËÊ®·ÖÖ±½ÓµÄ×÷ÓÃ.

¡¡¡¡ËùÓÐÊÕµ½µÄ¾èÔùÆ·¶¼Ô-·â²»¯µØÓÃÔÚÁË°¢×ÓÃÇÉíÉÏ. ÎÒÃÇµÄÄ¿±êÊÇÕÙÈ¡ÓÐ
5%µÄËËÀ´¾èÔù, Êµ¼ÊË®Æ½»¹µÍÓÚÕâ¸öÄ¿±ê(1998ÄêÊÇ1.9%)

¡¡¡¡Sven: ÎÒµÚÒ»½»Í¨¹ýEMAILÓëBram½Ó´¥Ê±Ëû¾ÍÔÚÎÚ¸É´ï. ÄÇÀï¶àÊýÁªÏµ¶¼ÒªÍ¨¹ýÖÐ¼äÈË.
·¢ËÍEMAIL¿É²»ÏóÏÖÔÚÕâÑù·¢µ½"user@domain"ÕâÃ´òµ¥.

¡¡¡¡µ±ÎÒ¶Áµ½"vimÊÇ´ÈÉÆÈí¼þ"Ê±ÎÒÒâÊ¶µ½vim²»½ö°ïÖúÁËËüµÄÈí¼þÓÃ»§ÃÇ,
Ò²°ïÖúÁËº¶ü¶à¹Â¶ù. ËùÒÔÎÒÏ£ÍûÎÒÔÚvimÉÏµÄÅå¬Á¦Ò²»á¶ÔËûÃÇÆðµ½¼ä½ÓµÄ°ïÖú×÷ÓÃ.

¡¡¡¡ÎÒÃ»È¥¹ýÚ¸É´ï, µ«ÎÒÔÚ1977Äê×¡ÔÚ¶«·Ç¿ÏÄáÑÇµÄ¼¸¸öÔÂÊ±¼äÊ±½Ó´¥ÁËÄÇÀïµÄÈËÃÇ.

Ò²Ë¥Ñ§Ð£¿´ÁË¿´(ÖÜkissi - kissiËÇË²Á´µØ·½)

¡¡¡¡Ã¿´Î¿´µ½´ÓÍâÃæ¹«Ë¾À´µÄEMAILÎÒ¶¼ÄþÔ¸ËüÃÇÊÇÎÊËüÃÇÄÜ·ñÎªÎÚ¸É´ïµÄ¹Â¶ùÃÇ×öµãÊ²Ã´¶ø²»ÊÇÎÊvimµÄÇ§Äê³æÎÊÌâ.

¡¡¡¡¼Ç: ÊÀ½ç¸öÓÐ¶àÉÙÈËÊ¹ÓÃvim? ËüÃÇÐèÒªÊ²Ã´? Î ªÊ²Ã´ËüÃÇÑ¡ÔñÁËvim?
ËüÃÇÖ÷Òªµ ÄÐèÇóÊÇÊ²Ã´?

¡¡¡¡Bram: ºÜÄÑËµµ½µ×ÓÐ¶àÉÙÈËÔÚÊ¹ÓÃvim, ÒòÎªvim²»ÐèÒª×¢²á. µ«´óÖÂ¿´À´.
Ã¿¸ölinuxµÄ·Ö·¢°üÀï¶¼°üº¬Ò»·Ývim. ÓÐÈË¹ã·ºÕ÷ÎÊ, µÃµ½ÊÇ¹À¼ÆÓÐ5%µÄLinux
ÓÃ»§¾-³£Ê¹ÓÃvim. LinuxµÄÓÃ»§´ó¸ÅÓÐ1700Íò. ÄÇ¾ÍÊÇËµ´ó¸ÅÓÐ85ÍòµÄvimÓÃ»§.
ÔÙ¼ÓÉÏÔÚSolaris»òWindowsµÈÆ½Ì¨ÉÏÊ¹ÓÃvimµÄÓÃ»§.

¡¡¡¡Sven: µÄÈ·ÊÇ, vim²»Ðè×¢²á. ÕâºÜºÃ, Äã²»Ï£ÍûÒ¸ö×ÔÓÉÈí¼þ»¹Òª×¢²á, ¶Ô°É?
ÈÎºÎÐÎÊ½µÄ×¢²áÒªÇó¶¼¿ÉÄÜ»á¸ÏÂÜvimµÄÓÃ»§.

¡¡¡¡ÎÒ¿ÖÅÂÄÑÒÔ¹À¼ÆvimµÄÓÃ»§Ãû, »¨Ê±¼äÈ¥¼ÆËãÕâ¸öÊý×ÖÖ»»áÀË·ÑÊ±¼ä.

¡¡¡¡²»¹ýÓÐÒ»¼þÊÂ¿ÉÒÔ¿Ï¶¨ -
Ô½À´Ô½¶àµÄÈËÃÇ´Ó×î³õµÄvi×ª¶øÊ¹ÓÃviµÄ¸÷ÖÖ¿ËÂ¡°æ±¾(elvis, lemmy, nvi, vile, vim).

¡¡¡¡viµÄ¿ËÂ¡°æ±¾ÔÚ¹¦ÄÜÉÏµÄÔöÇ¿ÊÇÈÊÃÇ·×·××ªÏòËûÃÇµÄ¸ù±¾Ô-Òò. ÖÁÓÚvim µÄÓÃ»§,
Ã¿Ò»´ÎÐÂ¹¦ÄÜµÄÔö¼Ó¶¼ÎªÓÃ»§×ªÏòËüÔö¼ÓÒ»¸öíÀÂë.

¡¡¡¡vim-5×î´óµÄÐÂÔöÌØÐÔÊÇÓï·¨¸ßÁÁ,
ÕâÒ»¹¦ÄÜÊ¹µÃvim¿ÉÒÔ¸ù¾Ýµ±Ç°±à¼-µÄÎÄ±¾ÀàÐÍËù¶ÔÓ¦µÄÓï·¨¹æÔòÒÔ²»Í¬µÄÑÕÉ«ÏÔÊ¾ÎÄ±¾µÄ²»Í¬³É·Ö.
Í¨³£ÕâÓÃÓÚ³ÌÐòÔ±ÃÇ±à¼-ËüÃÇçµÄÔ´´úÄëÎÄ¼þ.
²»¹ývimÒ²ÄÜºÜºÃµØÔ§³ÖEMAILºÍÐÂÎÅ×éÏûÏ¢µÄÓï·¨¸ßÁÁ¹¦ÄÜ.

¡¡¡¡¼Ç:
vim5.4°æ¸Õ·¢²¼ÄãÁ¾ÍÕÅÂÞ×ÅÒªÔÚ2000Äê·¢²¼vim6µÄÊÂÁË...ÏÂÒ»¸öÇ§ÄêÖÐÄã¶ÔvimÓÐÊ²Ã´¼Æ»®?

¡¡¡¡Bram: Õâ¸öÎÊÌâ, ×ºÃ¿´Ò»ÏÂvimµÄTODOÁÐ±í. ¼øÓÚTODOÁÐ±íÏÖÔÚµÄ·ÝÁ¿,
ÎÒ¿ÉÄÜ½ÓÏÂÀ´µÄÕû¸öÒ»Ç§ÄêºÒªÓÃÀ´¸Ä½øvim²ÅÐÐ! :-)

¡¡¡¡È¥Äê½øÐÐÁËÒ»´Î¹ØÓÚÓÃ§×îÏ£ÍûµÄvimÌØÐÔµÄÍ¶Æ±. ÕâÓÐÖúÓÚÎÒ¾ö¶¨½ÓÏÂÀ´Òª×öÊ²Ã´.
ÆäÖÐÒ»¸öÌØÐÔÊÇ`ÕÛÐÐ'. ÕâÊÇVim6.0ÖÐÊ×Òª µÄÐÂÔöÌØÐÔ. `ÕÛÐÐ'¿ÉÒÔÒþ²ØÏÄ±¾µÄÒ»²¿·Ö,
ÕâÑù¿ÉÒÔ°ÜÉÝÒ×°ÑÎÕÕû¸öÏÄµµµÄ½á¹¹. ±ÈÈçËµ, Äã¿ÉÒÔ°Ñº¯ÊýÌå¶¼¸ø`ÕÛµþ'ÆðÀ´.
ÕâÑùÄã¾Í¿ÉÒÔ¶Ôº¯ÊýÓÐ¸ö´óÌåµÄ°ÑÎÕ, »òÐíÄãÒªØØÐÅÅÁÐËüÃÇµÄË³Ðò.
¹âÊÇÕâ¸ö¾ÍÓÐºÜ¶à¾ßÌåµÄÊÂÒªÈ¥×ö, ËùÔÔ×îºÃÎª´Ë·¢²¼Ò»¸ öÐÃµÄ°æ±¾.

¡¡¡¡Sven: 1998ÄêµÄÍ¶Æ±½á¹ûÏÔÊ¾`ÕÛÐÐ'ÊÇÓÃ»§×îÏ£ÍûµÄÌØÐÔ.
ËùÒÔvim-6µÄÖ÷ÒªÄ¿±ê¾ÍÊÇ¼ÓÉÏÕâÒ»ÌØÐÔ.

¡¡¡¡Æä´Î, ÓÃ»§×îÏ£Íû¿´µ½µÄÌØÐÔÊÇ`´¹Ö±´°¿Ú·Ö¸ô', ÕâÒ»ÌØÐÔ¿ÉÒÔ´½¨´¹Ö±ÏÔÊ¾µÄ×Ó´°¿Ú.
ÒòÎªºÜ¶àÓÃ»§Ï£ÍûÄÜ±È½ÏÁ½¸ öÔÚÄÚÝÉÏÏà½üµÄÎÄ¼þµÄ²»Í¬Ö®´¦. ÌØ±ðÊÇËûÃÇÔÚ±à³ÌµÄÊ±ºî.

¡¡¡¡ÓÃ»§ºôÉùÅÅÃûµÚÈýµÄÊÇÎª¸ ÷ÖÖÓïÑÔÁ¿Éí¶¨×ö`¿ÉÅäÖÃµÄ×Ô¶¯Ëõ½ø', ÒòÎª
Vim¾ø¶ÔÊÇÒ»¸ö³ÌÐòÔ±µÄ±à¼-Æ÷...

¡¡¡¡µ«ÊÇ²¢²»ÊÇËµÄã·ÇÒªÊÇÒ»¸ö³ÌÐòÔ±Å¿ÉÒÔÓÃºÃVim. ÊÂÊµÉÏ, ÎÒ¸üÏ£Íû
VimÄÜÎª³õÑ§ßÔö¼ÓÒ»Ð©ÔÚÏßÖ§³ÖºÃÈÃËûÃÇÊìÏ¤`Ä£Ê½±à¼-'µÄ¸ÅÄî,
µ«ÕâÎÞÓÉ»áÔö´ó¿ÉÖ´ÐÐÏÄ¼þµÄ´óÐ¡. ËùÒÔÎÒÖ»ÄÜ¼ÄÏ£ÍûÓÚÚÚÏßÎÄµµÁË.

¡¡¡¡ËäÈ»Vim¾¡Á¦±£³ÖÓëViµÄ¼æÈÝÐÔ, µ«¿ÉÒÔ¿Ï¶¨´ó¶àÊýÓÃ»§»¹ÊÇºÜÏ²»¶Vim µÄÔöÇ¿ÌØÐÔ,
ÌØ±ðÊÇ¿´µ½ÆÁÄ»ÉÏ¿ÉÒÔ¿´µ½¹ØÓÚµÇ°±à¼-ÐÄÏ¢µÄÌáÊ¾, ÎÄ¼þÃû ºÍ»º³åÇøºÅ,
¹â±êÎ»ÖÃºÍÆäËüÒ»Ð©Ñ¡ÎîÈç×î´óÐÐ¿í.

¡¡¡¡¶Ô³õÑ§ßÑ¶øÑÔÄÜÔÚÆÁÄ»ÉÏ¿´µ½ËùóÐµÄ×Ö·ûÊÇºÜÖØÒªµÄ(ÐÐÎ²¿Õ°××Ö·ûºÍÌØÊâ×Ö·û).
ÌØÊâÑÕÉ«µÄÏÔÊ¾ºïÁÉÕâ¸ öµÄ´óÃ¦.
VimÄÚÖÃµÄÎÄµµ¸ ñÊ½»¯¹¦ÄÜ¶ÔÓÚÒª·¢ËÍµ½ÐÂÎÅ×éµÄÎÄ±¾»òEMAILÐÅ¼þÀ´ËµÊÇÒ»¸ öÊ®·ÖÇ¿´óµÄ¹¤¾ß.

¡¡¡¡Í¨¹ýÍøÂçÁ¬½Ó±à¼-Ò»¸öÔ¶³ÌÎÄ¼þ, `ÕÛÐÐ'(¼ûÉÏ),
Ê¹ÓÃ¶à×Ö½Ú·Ö·ûÓïÑÔµÄÖ§³ÖÒ²ÊÇµÃÆ±ºÜ¸ ßµÄÇëÊ¾. µ«ÊÇÕâÐ©ÊµÏÖÆðÀ´¿É²»ÈÝÒ×.
Í¬Ê±ÎÒÒ²Ï£Íû´ó¼ÒÄÜ°ïÖúÎÒÃÇÊµÏÖÕâÐ©ÓÔÈËµÄ¹¦ÄÜ.

¡¡¡¡ÎÒÓÈÆäÏ£Íû¿´µ½`ÕÛÐÐ'¹¦ÄÜ. ºÜ¶àÓÃ»§½«½èÖúÕâÒ»ÌØÐÔä¯ÀÀËûÃÇÎÄ¼þµÄ´ó¸Ù.
ÓÊ¼þÁÐ±íÉÏµÄÌÖÂÛ±íÃ÷ÕâÊµÏÖÕâÒ»µã»¹ÓÐºÜ¶àÀ§ÄÑÒª¿Ë·þ, ÕâÕæÊÇÌ«¸ »ÓÐÌôÔ½ÐÔÁË!

¡¡¡¡²»¹ý, ÎÒ»¹ÊÇÈÏÎªÔÚÔö¼ÓÕâÐ©ÖØÍ·Ï·Ö®Ç°×îºóÏÈ½â¾öÒ»Ð©Ð¡ÎÊÌâ.(±ÈÈç:
Ò»¸öÄÚÖÃµÄ·ÇÍ¼ÐÎÓÃ»§½çÃæµÄÎÄ¼þä¯ÀÀÆ÷.)

¡¡¡¡TODOÁÐ±íÊÇÒ»¸ ö¼¼ÊõÉÏµÄÄ¿±êÇåµ¥. ÉÏÃæµÄºÜ¶àÌõÄ¿±êÇ×ÅÒª½øÐÐµÄ¼¼ÊõÉÏµÄÌáÉý.
Èç¹ûÄã²»ÄÜÈìÖªVimµÄËùÓÐ¸ ÅÄî¿ÉÊÇºÜÄÑ¿´¶®Õâ¸ öÁÐ±íµÄ.

¡¡¡¡ºÜ¶àÌØÐÔµÄÖö¼ÓÊÇÎªÁËÄÖ²¹Í¼ÐÎÓÃ»§µÄ²»×ã. ¶øÇÒ,

ºÜ¶àË¹ÓÁViÖËÖ¸ Ëç·ËµÄÖÁ»§¸ ù±¾²»ÐèÒªÒ»¸ öÍ¼ÐÏÖÁ»§½çÁæ.
ÎÒ±¾ÈËÊ¹ÓÃGUIÒ²Ö»ÊÇÎªÁË°ïÖúÁË½â±ðÈË¹ØÓÚÕâ·½ÃæµÄÎÊÌâ, ×ö×Ô¼ºµÄÊÂÎÒ´ÓÀ´¶¼²»ÓÃËü.

¡¡¡¡´ÓÉÏÃæµÄÕâÐ©ÄãÒ²¿´µÃ³öÀ´ÎÒµÄÐËÈ¤Ö÷ÒªÔÂÓÓÚvimµÄ·ÇÍ¼ÐÎÓÃ»§½çÃæµÄÌØÐÔÉÏ,
ÕâÑùµÄÌØÐÔ¿ÉÒÔÔÚµÄÖÕ¶ËÉÏÊ¹ÓÃ, Ö»ÒªÓÃÃüÁî±¾Éí¾ÍÐÐÁË.
ÎÒËùÊÕµ½µÄ·´À¡Ò²±íÃ÷´ó¶àÊýÈËÒªµÄÕýÊÇÕâ¸ö -
¶ø²»ÊÇÖîÈç²Ëµ¥»òÒ»Ð©ÌØÊâ³ÌÐòµÄÖ§³ÖÈçÆ´Ð´¼ì²éÆ÷»ò±à³ÌÓïÑÔ·½Ãæ.

¡¡¡¡ËùÒÔºÜ¾ÃÒÔÇ°ÎÒ¾ÍÓÐÁËÒ»·ÝÕâÑùµÄÁÐ±í, ÃèÊöÓÃ»§ÔÚÃ¿ÌìµÄ±à¼-¹ý³ÌÖÐÊµ¼ÊÅöµ½µÄÎÊÌâ,
ÒÔ°ïÖú¿ª·¢Ð¡×éÈÏÊ¶µ½ÕâÐ©ÎÊÌâ²¢Ìá³ö½â¾ö·½°¸.

¡¡¡¡¼Ç: Sven, Bram,ÔÙ´Î¸ÐÐ»ÄãÃÇçµÄ°ïÖú. Í¬Ê±¹§Ï²ÄãÃÇÔÚVimÉÏËù×ö³öµÄ³öÉ«µÄ³É¹û.
»¹ÓÐÊ²Ã´Òª²¹³äµÄÂð?

¡¡¡¡Bran: Ð»Ð»ÄãµÄ×¨·Ã. ÉÌÒµ³ÌÐò¿ÉÒÔÓÃ¹ã¸æÀ´ÎüÒýÓÃ»§,
ÏóVimÕâÑùµÄ×ÔÓÉÈí¼þ¾ÍÒª¿¿±ðµÄ°ì·¨ÁË, ËùÒÔ¸ÐÐ»ÄãÕâÑùµÄ×¨·Ã.

¡¡¡¡ÎÒÊ®·ÖÏ²»¶ÓÃVim½øÐÐ±à¼-,
ÎÒÏ£ÍûÎÒÔÚ¿ªÔ´Èí¼þÉÏµÄÅ¬Á¦Ò²ÄÜ°ïÖú¸ü¶àµÄËËÃÇ´ïµ½ËûÃÇµÄÄ¿±ê.

¡¡¡¡Èç¹ûÄãÊ¹ÓÃVim¹ý³ÌÖÐÓöµ½ÎÊÌâ¿ÉÒÔÊÔÒ»ÏÂ":help", ËùÓÐµÄÔÚÏß°ïÖúÎÄµµ¶¼ÊÇ´¿ÎÄ±¾µÄ,
ËùÒÔ´òÓ¡³öÀ´Ó¦¸Ã²»»áÓÐÊ²Ã´ÎÊÌâ.

¡¡¡¡¸½¼ÓµÄÎÄµµÒ²²¹ÉÏÁË, ±ÈÈçµÂÓï°æµÄHowTo°Í"Vim³õÑ§Õß"(Ò²ÊÇµÂÓï)

¡¡¡¡ÄãÒ²¿ÉÒÔÔÚcomp.editors·¢ÎÊ, »òÕßÍ¨¹ý6¸ö×¨ÓÃÓÊ¼þÁÐ±í - Í¨ÓÃµÄ°ïÖúÁÐ±í.

¡¡¡¡»¶Ó-ÔÚÄãµÄÖ÷Ò³ÉÏ·¢²¼ÄãµÄ¸öÈËÊ¹ÓÃ¼¼ÇÉ»òÇÏÃÅ. »òÕßÊÇÎ¬»¤Ò»¸öÓï·¨ÎÄ¼þ,
Ä³ÖÖ²Ù×÷ÏµÍ³µÄ¶þ½øÖÆÎÄ¼þ, »òÕßÔÚÐÂÎÅ×é, ÓÊ¼þÁÐ±íÀï»Ø´ð±ÈËµÄÎÊÌâ

¡¡¡¡·Ç³£·Ç³£»¶Ó-Ìá¹©¸÷ÖÖÐÎÊ½µÄ°ïÖú. :-)

    ---END----
                    ÖÂ
Àñ£¡

            ÕÔÈç·É
            slimzhao@21cn.com

Site Information

- Home
- About This Site
- Advertise This Site
- Search This Site
- Submit Software

Software Categories

- Audio
- CDRW Softwares
- Communications
- Desktop
- Developer Tools
- Distributions
- Email
- File Management
- Graphics
- Internet
- Network
- Office
- Security
- System
- Video MPEG
- Web Design

Editors - Page 1

Last Updated: 10/07/2001

jEdit Development v3.1 (Editor's Pick) ★★★★★

jEdit is a text editor designed for programmers. jEdit's features include: syntax highlighting for over 30 file types, including C, C++, HTML, Java, and Perl; automatic indenting; bracket highlighting and matching; a macro system; and scriptable search and replace capability. You can set up function abbreviations; record any number of strings and caret positions in multiple registers; perform rectangular selections for working with column-based files; and customize file filters in the "Open" and "Save" dialog boxes. The program also includes powerful text editing commands for working with whole words, lines, and paragraphs; the ability to open any number of editor windows; and unlimited undo and redo capabilities.
[Download] [Home Page] [License: Free To Use But Restricted] [Size: 868KB] [Date Updated: 23/04/2001]

VIM 5.8 (Editor's Pick) ★★★★★

VIM is an improved version of the editor "vi", one of the standard text editors on UNIX systems. VIM adds many of the features that you would expect in an editor: unlimited undo, syntax coloring, split windows, visual selection, GUI support, and much more.
[Download] [Home Page] [GNU Public License] [Size:] [Date Updated: 01/06/2001]

gEdit v0.9.0 ★★★★☆

gEdit is a text editor for the Gnome desktop environment. gEdit is designed to be simple, light, and fast, and includes a powerful plug-in system.
[Download] [Home Page] [GNU Public License] [Size: 775KB] [Date Added: 31/03/2001]

Glimmer 1.0.6 (Editor's Pick) ★★★★★

Glimmer is an advanced text and code editor for the Gnome desktop environment. Glimmer has many features, including the ability to open and edit multiple files; Cut, Copy, and Paste capabilities; multiple levels of Undo and Redo; syntax highlighting for C, C++, PHP, Java, Perl, Python, and z80 assembly language; Python and Guile support for writing macros and scripts.
[Download] [Home Page] [GNU Public License] [Size: 490KB] [Date Updated: 10/07/2001]

Code Crusader 3.0.0 (Editor's Pick) ★★★★★

Code Crusader is designed to smoothly integrate the tasks of working with source files, compiling, and debugging, in short everything involved in developing code.
[Download] [Home Page] [License: Single End-User Software] [Size: 2MB] [Date Added: 31/03/2001]

Tk Notepad v0.7.7 ★★★★☆

Tk Notepad is a text editor written using Tcl/Tk. Tk Notepad was written to mimic the Microsoft Windows Notepad program. The program contains all of the basic functions of Notepad, such as: cut, copy, paste, and search, but also adds other features such as, Undo and Redo.
[Download] [Home Page] [GNU Public License] [Size: 20KB] [Date Added: 31/03/2001]

Boa Constructor for Linux v0.0.5 (Editor's Pick) 

Boa Constructor for Linux is an integrated development environment (IDE) for graphical user interface (GUI) building. Boa Constructor offers visual frame creation and manipulation, an object inspector, a debugger, and an integrated help. The program also has many different views of the source including inheritance hierarchies, and object methods and properties. [Download] [Home Page] [GNU Public License] [Size: 348KB] [Date Added: 31/03/2001]

J for Linux v0.8.2 

J for Linux is a Java-based, multi-file and multi-window programmer's editor. The program features syntax highlighting for C, C++, HTML, JavaScript, Java, LISP, Perl, and XML documents; automatic indentation; directory buffers; support for regular expressions; multi-file find and replace capabilities; automatic saving and crash recovery functions; and FTP and HTTP support. You can customize all keyboard mappings, and use themes to personalize the editor's appearance.
[Download] [Home Page] [GNU Public License] [Size: 596KB] [Date Added: 31/03/2001]

SoftLandIndia
Email: linuxsoftlandindia@hotpop.com

ICCF Holland

Click and make a donation for free:

The Vim book by Steve Oualline is the first book to be completely dedicated to Vim, the text editor.

I can recommend this book especially for beginners and those who use Vim for a short while and would like to learn more. The most often used commands are explained with many figures and examples. Steve has a writing style that is very easy to read. Advanced Vim users will find many hints for useful features. The more advanced items are not explained in detail though.

The foreword is great! Well, I wrote it... I also reviewed the text, hopefully all mistakes have been corrected. There are appendices that list all the commands and options. At the end there is a quick reference for the most often used commands. The text was written for Vim version 5.7.

The book has been published in April 2001. It may take a little while before it's available in brick&mortar bookstores. You can find information from the publisher here. There is a sample chapter (PDF format) and the table of contents. The quick reference is available as PDF and HTML, so that you can print it. Although this says there are 450 pages, there are actually about 600 pages!

I have started an unofficial **errata** list. You can find it here

You can find the Vim book at the sites below. Buying through these links will send 5% to 15% of the sales to ICCF Holland. The money is used to help orphans in Uganda (see the home page). You should also be able to get the book in other (on-line) bookstores that have the New Riders books.

Amazon USA amazon.com.

For comments on the ICCF Holland pages and for more information contact Bram Moolenaar.

todo.txt   For Vim version 6.0z. Last change: 2001 Mar 24

VIM REFERENCE MANUAL   by Bram Moolenaar

TODO list for Vim   *todo*

This is a veeeery long list of known bugs, current work and desired
improvements. To make it a little bit accessible, the items are grouped by
subject. In the first column of the line a classification is used to be able
to look for "the next thing to do":

Priority classification:
9   next point release
8   next release
7   as soon as possible
6   soon
5   should be included
4   nice to have
3   consider including
2   maybe not
1   probably not
-   unclassified

See |votes-for-changes| for a list of votes on desired changes in Vim.
See |develop.txt| for development plans.

known-bugs
Known bugs and current work

Sunday March 25: Check for problems with summertime switch:
-   Win32: All files created on the day of switching from winter to summer
    time cause "changed since editing started" messages. It goes away when
    the file is written again the next day, or the timezone is adjusted.
    DJGPP version is OK. (Zaimi) Looks like a problem with the Win32 library.
    Rebooting doesn't help. Time stamps look OK in directory. (Penn)
    Is this on FAT (stores wall clock time) or NTFS (stores UTS)?

The source archive is over the 1.4M floppy limit. How to avoid this?

Main items for 6.0:
A. Folding
        Works.
B. UTF-8 support
        Works.
C. Multi-line regexp patterns
        Works.
D. Vertical window split
        Works. Add some way to show diffs.
E. Virtual edit
        Works for some commands. Still needs to be improved a lot.
F. Sun Visual workshop (debugger) interface
        Code is included but not tested.
G. Edit command line with Vi commands
        Command-line window is working.
        Make Insert mode in Command-line window work like Command-line mode.
        Use a single-line Command-line window for the command line?
H. More automatic testing

        Nothing done yet.
I. Multi-language support
        Message translation works.
        ":menutrans" works, but how to handle different encodings?
J. User Friendlier
        Write user manual.
        Enhance Windows install program.

D. Vertical window split:
-   Show differences between files (like sdiff but in separate windows).

E. Virtual edit:
-   Still a lot of work to do. Test various commands.
-   Using "A" in Insert mode, then <Right><BS> deletes the last char of the
    line, even though the cursor doesn't touch it. (Mary Ellen Foster)
    Check other <BS> situations in Insert mode.
-   Using "C" then then <CR> moves the last remaining character to the next
    line. (Mary Ellen Foster)
-   Bugs: (Mark Waggoner)
    See patches in ~/Mail/oldmail/waggoner/in.00132.
    5. 'v' selection is kind of strange. When in virtual space, pressing 'v'
       will select the last character on the line, though the cursor remains
       in virtual space.
        Should set the start of the Visual area in virtual space.
    9. Using "C" in virtual space moves back to the end of the current
        line to start changing rather than inserting spaces up to the current
        point and then changing.
        And when the cursor is in a TAB somewhere, it also doesn't work right.
-   Leaving Insert mode when 'virtualedit' is "insert" should move the cursor
    to a valid character.
-   In Visual-block mode highlighting part of a tab is wrong.
-   Add more documentation.
-   Use vpos_t instead of fpos_t for positions that can have a coladd?
-   coladvance2() can be a bit slow.
-   Should not insert spaces for yank or filter.
-   When past the end of a line that ends in a single character "b" skips that
    word.

F. Sun Visual Workshop patch from Gordon Prieur:
-   Displaying signs doesn't always work properly. Should use a flag in the
    window for the sign column: it is set as soon as a sign is drawn and reset
    when editing another file. When drawing the first sign the display isn't
    updated properly. Force a redisplay when setting the new flag. When not
    in the GUI or no motif, only use line highlighting (later add ASCII sign,
    like ">>" for an arrow, "!!" for a breakpoint).
-   sign mixed in with highlighting, is that what we want?
-   Check that rename from "glyph" to "sign" is correct.
-   Add ASCII signs, see patch from Mirian Lennox.
-   remove ":signs", use ":sign" without arguments to show signs
-   docs for 'balloon*' options.
-   docs for 'F' flag in 'guioptions'.
-   add stdarg.h include file in os_unix.h?
-   Add code to handle folding.

G. Edit command line with Vi commands:
    For the command-line window:

- add a few command-line editing commands in edit(). Vim command completion? Would also be useful for Vim script editing. Eventually the Elvis way would be a small step.
- Use non-Visual selection in the other windows, to be able to copy/paste text.

The Elvis way: the cmdline is a one-line view on a buffer with the command line history.
See remarks from Steve (~/Mail/oldmail/kirkendall/in.00012).
> solve conflicts between Insert mode and Command-line mode commands. make it work like Korn shell and tcsh.
> where to show insert/normal mode message? Change highlighting of character in first column?
> use CTRL-O to execute one normal mode command
> How to switch to normal mode for more commands? <Esc> should cancel the command. CTRL-T?
> Set "curwin" and "curbuf" to the command line window and buffer.
> curwin->w_topline is always equal to curwin->w_cursor.lnum.
> Use edit() for normal cmdline editing? Would have to integrate getcmdline() into edit().
> never set 'number', no folding, etc. no status line
> When executing the line, do like "U" and append the executed line at the bottom.
> Differences between Insert mode and Command-line-insert Mode:
  - <CR> doesn't start a new line
  - completion with 'wildchar'
  - use cmdline abbreviations
> Differences between Normal mode and Command-line-normal Mode:
  - <CR> executes the line
  - no CTRL-W commands
  - no ":" commands?
  - Visual mode only within one line.
  - "j" and "k" move in the history.
> sync undo after entering a command line?
> allow "/" and "= to recursively call getcmdline(), overwrite the cmdline.
> use NV_NOCL flag for commands that are not allowed in Command-line Mode.

H. More automatic testing:
- Check all commands for:
  - handling folds (moving vertically, updating folds)
  - vertical splits (displaying with w_wincol, updating w_width and other windows)
  - use of multi-byte characters.

I. Multi-language support:
- Should handle aliases for different locale settings. "de" is "de_DE" is "de_DE.ISO8859-1". Perhaps use some code from gettext()?
- Cmdline completion of ":langugage" for "messages" and "lctype".

J. User Friendlier
- ":setlocal option<": reset local option to global value.
9 Add a menu (or sub-menu) with most often used options: 'wrap', 'textwidth', etc.
- Add a way to save local settings and mappings into a new plugin file. ":mkplugin <file>"?
9 Add buffer-local menu. Should offer a choice between removing the menu or

    disabling it. Be careful that tear-offs don't disappear (keep one empty
    item?).
    Alternative: use BufEnter and BufLeave autocommands.
9   Buffer-local autocommands?
        :au BufEnter <current> menu enable ...
        :au BufLeave <current> menu disable ...
9   Make Windows install program better:
    - Add an icon on the Desktop, if the user wants this.
    - Add Vim to the Start/Programs menu if the user wants this.
    - Detect the OLE version and register it.
    - Allow adding a "edit with Vim" entry even when Vim already is in $PATH.
    - Create $VIM/vimfiles/[plugin|ftplugin|doc] directories, so that plugins
      can really be dropped-in.
    - Delete the vim.bat and gvim.bat files.
    - Delete a whole tree of Vim files.
-   Add mappings local to a window: ":map <window> ..."?
8   Add better control for protecting the user for editing a file twice, file
    changed outside of Vim, file protection changed outside of Vim, etc.

Messages:
-   In the message history, add a help tag for each message and number the
    messages. ":msghelp <nr>" could then be used to jump to help for the
    message. (Charles Campbell)
    Add the ID to each error message, like: "a7: No match found"?
-   Need to hit return after selecting entry for ":ts".
-   Remember message line for redraw. Use instead of keep_msg. Integrate the
    command line in update_screen().
-   Make keep_msg allocated. Using IObuff is dangerous.
-   Check: ":cn" message is sometimes cleared.

Various recent:
8   Win32 GUI: With maximized window, ":set go-=r" doesn't use the space that
    comes available. (Poucet) It works OK on Win 98 but doesn't work on Win
    NT 4.0. Leaves a grey area where the scrollbar was. ":set go+=r" also
    doesn't work properly.
-   In ":ptag" preview window highlight the tag. Use the new regexp items.
9   Motif GUI: ":hi menu font=name" crashes Vim in some situations. E.g.:
        :hi menu font=fixed
        :hi menu font=asdf
9   Crash when creating XIC without a fontset. (Diamond) Don't include the
    XNFontSet then? Check if gui.normal_font is a fontset?
-   Motif GUI: draws in right border of window, scrolling leaves pixels
    behind. Scroll two more pixels? Or make window two pixels wider?
8   Motif GUI: The tooltips disappear after one or two seconds. Why don't
    they stay? For GTK they also disappear, but re-appear after a second
    (making them blink). Is there an event that interferes?
8   Syntax error: Can't reproduce it now. Looked like an "O" command caused a
    line to be redrawn that was moved down (since a line was inserted), found
    a start-of-comment error on "/*", thus it was using a cached state that
    the comment already started in this line (what would be true for this line
    before the new line was inserted)
9   These places need to be fixed for EBCDIC:
    -   AppendToRedobuffLit().
    -   stuffescaped().
    -   set_last_insert()
8   gui_w16.c and gui_w32.c still have much code in common. Move this to
    gui_w48.c.
9   What happens with :execute "normal a\<Up>\200\<Esc>"?

    How to distinguish a special key from a 0x80 byte? CSI too.
9   This pattern in syntax/java.vim causes a recursive call of regmatch():
        syn match   javaStringError   +"\([^"\\]\|\\.\)*$+
    A long line with a " in it can cause a crash when it runs out of stack
    space (on non-Unix systems). How can we catch this?
8   'cindent' problem: (FooLman)
                int x;   <- extra indent, ";" ignored.
            void func()
            {
8   Implement ":tearoff" for other systems than Win32 GUI.
8   ":let @" = "val" should not have side effects for other registers (after
    "ayy). (Raul Acevedo)
    ""yy should probably not change the last used register from "ayy.
9   Support ACLs on more systems.
9   Win32: double clicking on "c:\tmp\{asdf}.txt" edits "c:\tmp{asdf}.txt".
    Don't remove backslash before {}? Simlarly for "Edit with Vim". (Ott)
    See list with success/failure for different names and situations.
9   Win32: mouseshape doesn't return to normal after a ":set" command.
    (Timothy) Stays at "no entry". Only visible with ":set nomousehide".
9   Win32: when 'shell' is "zsh", ":!start notepad file" doesn't work. (Pavol
    Juhas).
8   Windows NT/2000: LCTranslateMessage() doesn't work properly. It's
    disabled for now. What is the right solution?
-   GUI: Implement ":popup" for other systems than Windows.
-   GUI: <C--> cannot be mapped. Should be possible to recognize this as a
    normal "-" with the Ctrl modifier.
9   GTK GUI with Gnome: When $DISPLAY is wrong, ":gui" unexpectedly exits.
    Without Gnome it does work.
    Not all Gnome arguments get to gnome_init().
8   Add "display" argument to many items in syntax files, to speed up parsing.
    How to avoid that a "display" keyword no longer works? Also for "fold".
8   Make ":syntax include" to search 'runtimepath' for the included syntax
    file if the argument is relative.
-   update $VIMRUNTIME/optwin for new options.
-   When using the file browser in GTK the absolute file name is used. Should
    try to simplify it. (Lohner)
-   Make it possible to set 'browsedir' to a directory name, which is always
    used.
-   When checking if a file exists on Unix, uses lstat(), otherwise a symlink
    will not be found. Go throught the code to check for this.
-   Check all use of islower()/isupper() for negative value (special key).
-   Use s:name local variable in :source'ed file.
-   Include pty check for Lynx: ~/vim/patches/hahn.pty .
-   In an xterm, look ahead for mouse drag events, merge them and use the last
    coordinates.
-   Remove "\" shown in complete/wildmenu.
    Bug: <C-O>:emenu in Insert mode inserts the command.
-   When accidentally hitting Shift-R instead of Ctrl-R, further Ctrl-R is not
    possible, also when typing <Esc> immediately. (Grahn)
-   Include flipcase patch: ~/vim/patches/wall.flipcase2
-   "/$" in 'list' mode should put the highlighting on the "$" character, also
    for empty lines.
8   Use $install_prefix? (Cabric)
-   Where 'suffixesadd' is used, also try removing a trailing "." or "," from
    the file name.
8   Test if it possible for a skip pattern to include "\n", so that it can
    skip over newlines for a "oneline" region.
-   multi-line regexp:
    Make test for multi-line regexp and ":s".
    Check use of multi-line regexp in syntax highlighting.

- When starting GUI with invalid $DISPLAY, the hit-enter prompt is always in line 2, while the "cannot open display" is somewhere else.
- When swap file message sees that date of file is newer as date of swap file, add message "more recent than swap file".
- "ls -1 * | xargs vim" works a bit but not well. Should use stderr for all term I/O.
- Check libcall() and libcallnr() for Win32. No crash when function returns 1 or -1?

9 ":cwindow!" always open quickfix window, ":cwindow" only open it if there are recognized errors. ":cclose" to close quickfix window?

8 For ":silent !cmd":
":silent write" still writes a message.
":silent s//" still writes a message.
How to use ":silent" for a search command? ":silent /pat" uses ":/pat".
:silent exe "normal /path\<CR>"
Run the external command silently, don't give the hit-enter prompt.
Remove how it's done for Risc OS with ":!~cmd", see do_shell().
Don't redraw the screen, make a note in the docs that CTRL-L should be used if the command has any output.
-> pass SHELL_FILTER to call_shell?
-> ":silent!" also stops error messages.
-> ":silent menu" creates menu that doesn't echo its command.
-> ":silent /pattern" doesn't complain when it's not found
-> ":silent :s/pat//" doesn't complain when it's not found and doesn't report number of substitutions.
-> ":silent map": add flag to mapping, increment "silence" when exeucuting it, append a command to decrement "silence"; reset silence when aborting a mapping).
Or use ":map <silent>"?
-> ":silent unmap" shouldn't complain about a non-existing mapping.
Still do the messages for ":redir"?
-> ":silent redir": Remember the last message, print it if waiting for a key (otherwise the hit-enter prompt could be missed).
Ideas in ~/vim/patches/wall.noredir.tgz (Stephen P. Wall).

- Make syntax keyword table configurable. Set number of bits used with ":syn clear [hashbits]", so that we don't need to reallocate the table. minimal 4 bits, maximal 16. (Campbell)
- Check all places where buf_write() is called if it takes care of "buf" becoming invalid because of autocommands. Also callers of buf_check_timestamp().
- Add "gG": like what "gj" is to "j": go to the N'th window line.
- xterm sends <Esc>O5F for <C-End>
- Support ":browse edit" in console, using explorer.vim?
- When displaying a space with only foreground highlighting, it's the same as a space without attributes. Avoid displaying spaces for the "~" lines when starting up in a color terminal.
- Windows NT: writing to aux.* makes Vim hang. (Acevedo)
- Problem with cscope jumping to the wrong position? (Medley)
Selecting item 5 jumps to item 4. (Andrej Borsenkow)
- ":abbr b byte", append "b " to an existing word still expands to "byte". This is Vi compatible, but can we avoid it anyway?

Considered for release 6.0:

8 Add a function that returns the line in the tags file for a matching tag. Can be used to extract more info (class name, inheritance, etc.) (Rico Hendriks)
Add a command to jump to a certain kind of tag. Allow the user to specify values for the optional fields. E.g., ":tag size type=m".

8 Add file locking. Lock a file when starting to edit it with flock() or

    fcntl(). This patch has advisory file locking while reading/writing the
    file: ~/vim/patches/kahn_file_locking .
-   When $CFLAGS is set, don't change it in configure. (Griffis)
    Give a warning that CFLAGS from the environment is being used.
8   Use configure to set absolute paths in tools scripts when installing.
8   Windows (and others): When using the scrollbar to scroll, don't move the
    cursor position. When moving the cursor scroll to the cursor position.
9   It is possible to create the same swap file twice, when there is latency
    in the system (and 'directory' is ~/tmp). (Soulier)
    Create the swap file as soon as its name is chosen, when mf_fname is set.
    Delete it before changing mf_fname/mf_ffname.
8   Win32 GUI: Make ":silent !ls" work, avoid "hit any key to continue" in DOS
    console used to execute external commands. Pass a flag to vimrun.exe.
8   Add command like ":normal" that accepts <Key> notation like ":map".
8   Make clipboard work for Dos32 version. (Negri)
8   ">>" with 'softtabstop' set should work like 'tabstop' is set to
    'softtabstop'. (Zivkov) But what about when there is a string?
9   Install vim32x32.xpm in /usr/local/share/icons/ or ~/.kde/share/icons/ for
    using it with KDE.
7   Add ModeMsgVisual, ModeMsgInsert, etc. so that each mode message can be
    highlighted differently.
-   For files found with 'include': handle "*" in included name, for Java.
    (Jason)
-   Improve multibyte.txt using patch
    ~/ftp/vim/unreleased/patches/vim-5.6-dalecki-4.gz.

---------------------- bugs --------------------------

Vi incompatibility:
8   Docs for ":s&" are not correct. What is it supposed to do? When exactly
    are flags re-used? What is a Vim extension?
8   With undo/redo only marks in the changed lines should be changed. Other
    marks should be kept. Vi keeps each mark at the same text, even when it
    is deleted or restored. (Webb)
    Also: A mark is lost after: make change, undo, redo and undo.
    Example: "{d'" then "u" then "d'": deletes an extra line, because the ''
    position is one line down. (Veselinovic)
8   When using ":s" to split a line, marks are moved to the next line. (Geddes)
8   When using ":n" ":rew", the prev. context mark is in the same place as the
    cursor, not on the first line.
8   "zt" should scroll one screen forward and put the cursor in the first
    line.
8   ":z#" should list lines with line numbers. (Bob Farmer)
8   text is put in register 1 when using "c" and "d" with a motion character:
    <control-A>   %   (   ) `<character>   /   ?   N n   {   }
    (from Nvi manual).
8   When stdin is not a tty, and Vim reads commands from it, an error should
    make Vim exit.
8   Unix Vim (not gvim): Typing CTRL-C in Ex mode should finish the line
    (currently you can continue typing, but it's truncated later anyway).
8   When loading a file in the .exrc, Vi loads the argument anyway. Vim skips
    loading the argument if there is a file already. When no file argument
    given, Vi starts with an empty buffer, Vim keeps the loaded file. (Bearded)
8   In Insert mode, when using <BS> or <Del>, don't wipe out the text, but
    only move back the cursor. Behaves like '$' in 'cpoptions'. Use a flag
    in 'cpoptions' to switch this on/off.
8   When editing a file which is a symbolic link, Vim opens another symbolic
    link on the same file under the name of the first one. Adjust the file
    name in the buffer to the last one used? Use several file names in one
    buffer???

7 The ":map" command output overwrites the command. Perhaps it should keep the ":map" when it's used without arguments?
7 When interrupting reading a file, Vi considers the buffer modified. Add a 'cpoptions' flag for this (we don't want it modified always to be able to do ":q")?
7 CTRL-L is not the end of a section? It is for Posix! Make it an option.
7 Implement 'prompt' option. Init to off when stdin is not a tty.
7 CTRL-T in Insert mode inserts 'shiftwidth' of spaces at the cursor. Add a flag in 'cpoptions' for this.
7 Add a way to send an email for a crashed edit session. Create a file when making changes (containing name of the swap file), delete it when writing the file. Supply a program that can check for crashed sessions (either all, for a system startup, or for one user, for in a .login file).
7 Vi doesn't do autoindenting when input is not from a tty (in Ex mode).
7 "z3<CR>" should still use the whole window, but only redisplay 3 lines.
7 ":tag xx" should move the cursor to the first non-blank. Or should it go to the match with the tag? Option?
7 Implement 'autoprint'/'ap' option.
7 Add flag in 'cpoptions' that makes <BS> after a count work like <Del> (Sayre).
7 Add flag in 'cpoptions' that makes operator (yank, filter) not move the cursor, at least when canceled. (default vi compatible).
7 Make "5dd" on last-but-one-line not delete anything (Vi compatible). Add flag in 'cpoptions' for this. When not present, "2dd" in the last line should delete the last line.
7 This Vi-trick doesn't work: "Q" to go to Ex mode, then "g/pattern/visual". In Vi you can edit in visual mode, and when doing "Q" you jump to the next match. Nvi can do it too.
7 Support '\' for line continuation in Ex mode for these commands: (Luebking)
```
    g/./a\              g/pattern1/ s/pattern2/rep1\\
    line 1\             line 2\\
    line 2\             line 3\\
    .                   line4/
```
6 ":e /tmp/$tty" doesn't work. ":e $uid" does. Is $tty not set because of the way the shell is started?
6 Vi compatibility (optional): make "ia<CR><ESC>10." do the same strange thing. (only repeat insert for the first line).
8 When "cm" termcap entry is missing, starting gvim shouldn't complain about it. (Lohner) Try out with "vt100" entry (cm replaced with cX).

GTK+ GUI known bugs:
- GTK with Gnome gives error message:
```
    Gdk-WARNING **: locale not supported by C library
```
Looks like this can't be avoided.
9 When logging out from KDE, vim is not told to save its files. Catching the WM_SAVE_YOURSELF event doesn't work.
8 For some fonts characters are drawn in the line above. Include fix ~/vim/patches/dalecki-5.6 . (Marcin Dalecki)
9 When selecting a font from the font selector with an embedded space it cannot be found. (Tsirkin)
8 When a font is not fixed width, the message "not fixed-width" disappears immediately. (Tsirkin)
Add some code to handle proportional fonts? Need to draw each character separately (like xterm). Also for when a double-width font is not exactly double-width. (Maeda)
8 Move gtk_main_quit() calls from call to add_to_input_buf() to inside the function?
9 Can't paste a Visual selection from GTK-gvim to vim in xterm when it is longer than 4000 characters. Works OK from gvim to gvim and vim to vim. This is a GTK problem.

9 With the guifont "-*-clean-medium-r-*-*-8-*-*-*-*-80-*" bold characters
are drawn wrong. Try on C source code. The bold font is wider than the
medium font. Check bold font cellsize and use bold font trick if it's
wrong?
8 Font "7x14" has a bold version "7x14bold". Try to find the bold font by
appending "bold" when there are not 14 dashes.
8 When using "gvim -geom 40x30" or setting 'columns' in .gvimrc or with a
GUIEnter autocommand, the width is still set to fit the toolbar. Also
happens when changing the font. How to avoid that the toolbar specifies
the minimal window width?

Win32 GUI known bugs:
9 Check dosinst.c with Vim in "Program Files" (long filename). Does it use
the short filename instead and does the path compare work?
8 When ativating window with mouse click, don't move cursor to mouse click.
Catch WM_MOUSEACTIVATE. (Luevelsmeyer)
9 Clipboard doesn't always work as it should:
- Selecting text while editing a command-line has an implied copy to the
clipboard. Remove this, and add a command character to copy/cut/delete
the selected text (used in the popup menu).
- When selecting at the more-prompt or hit-enter-prompt right mouse
button doesn't give popup menu. Also has an implied copy to the
clipboard. Use the popup menu "Copy" entry for that.
9 When there are more than 32767 lines, the scrollbar has a roundoff error.
Clicking the up or down error scrolls two lines or more. Looks like there
is an extra event that reports the new position of the scrollbar.
9 When file modified outside of Vim and Vim gets focus, redraw is wrong.
Put focus event in input buffer and let generic Vim code handle it.
8 Make new version of VisVim.dll which includes the patch from Cordell.
8 When 'grepprg' doesn't execute, the error only flashes by, the
user can hardly see what is wrong. (Moore)
Could use vimrun with an "-nowait" argument to only wait when an error
occurs, but "command.com" doesn't return an error code.
8 When the 'shell' cannot be executed, should give an appropriate error msg.
8 MessageBox used for error messages can be too big. There is no way to
scroll it, or even hit the OK button (but you can hit return).
7 Add an option to add one pixel column to the character width? Lucida
Console italic is wider than the normal font ("d" overlaps with next char).
7 At the hit-enter prompt scrolling now no longer works. Need to use the
keyboard to get around this. Pretend <CR> was hit when the users tries to
scroll?
7 'mousefocus' is disabled until the next mouse click, when trying to exit
by hitting the X in the title bar. (Webb)
7 Scrollbar width doesn't change when selecting other windows appearance.
Also background color of Toolbar and rectangle below vert. scrollbar.
7 "!start /min cmd" should run in a minimized window, instead of using
"/min" as the command name. (Rogall)
6 Drawing text transparantly doesn't seem to work (when drawing part cursor).
8 CTRL key doesn't always work in combination with ALT key. It does work
for function keys, not for alphabetic characters.
8 CTRL-@ doesn't work. Don't even get a message for it?
8 CTRL-- doesn't work for AZERTY, because it's CTRL-[for QWERTY. How do we
know which keyboard is being used?
8 When using ":se guifont=Courier_New" bold chars leave pixels behind, in
front of a character, after a TAB. (McRae)
7 When scrolling, and a background color is dithered, the dither pattern
doesn't always join correctly between the scolled area and the new drawn
area (Koloseike).
7 File/Print uses notepad, which puts the name of the temp file on top of

    each page. How do we get the actual file name there?
    Alternative: use the PrintFile utility (freeware).
    Alternative: use prfile32 (Aaron)
8   install.exe, vimrun.exe and ctags are not build from Gvim_vc.mak. When
    added, check that this works from the command line and from the IDE.
8   When dropping a shortcut on gvim (.lnk file) should edit the target, not
    the shortcut itself.
9   OleVim doesn't send <C-\><C-N>, which causes trouble if Vim is in Insert
    mode. (Thakkar)
9   Replace SendToVim and OpenWithVim with C++ versions? (Rafal Dibrowa)
8   Tear-off menu causes a crash when it contains many items (>100).
    (Pedro Gomes) Probably because DLG_ALLOC_SIZE is a fixed number.

Athena GUI:
9   Use one of the Syntax menu items. Then ":aunmenu Syntax", Vim crashes.
    Also crashes when using Buffers.Refresh.
9   When using ":unmenu" in menu item, the reversing of menu items is wrong.
    Happens with Syntax/manual. Problem with not removing the reversed text
    when destroying the menu item? For Rochholz Vim crashes.
9   Because destroying a menu can cause a crash, it is not destroyed but
    unmanaged. The destoyed Widgets are remembered, this list is never
    cleaned up.
9   When closing the confirm() dialog, Vim exits.
9   Menu ordering doesn't work.
9   When using "menu a.b.c.d lll", the "b" submenu does not disappear.
9   When dragging the scrollbar thum very fast, focus is only obtained in
    scrollbar itself.
8   When "j" or "k" repeated quickly, display is messed up (Lauther).
8   Has extra stripes here and there. (Zeitlin).
    When dragging last status line with the mouse, small black lines appear in
    the command line pseudo-scrollbar (only when status line moved quickly).
7   The file selector is not resizable. With a big font it is difficult to
    read long file names. (Schroeder)

Motif GUI:
9   When starting GUI with ":gui" while 'writedel' is non-zero, escape
    sequences are split up. Motif version crashes.
8   When changing the color of the toolbar with ":hi toolbar", the pixmaps are
    not updated. Need to destroy them and allocate with the new colors.
8   Popup menu ordering is wrong.
8   Dialog: Pointer should be moved to the default button. Accelerators don't
    work yet.
8   Lesstif 0.89.4: Tearing-off a menu sometimes gives a "Restoring focus to
    NULL widget!" error. Probably a Lesstif problem.
8   Lesstif: When deleting a menu that's torn off, the torn off menu becomes
    very small instead of disappearing. When closing it, Vim crashes.
    (Phillipps)

GUI:
9   When using a very small font, 'columns' can be very big. Pass mouse
    events with 16 bits.
9   On Solaris, creating the popup menu causes the right mouse button no
    longer to work for extending the selection. (Halevy)
9   When running an external program, it can't always be killed with CTRL-C.
    e.g. on Solaris 5.5, when using "K" (Keech). Other 'guipty' problems on
    Solaris 2.6. (Marley)
9   On Solaris: Using a "-geometry" argument, bigger than the window where Vim

    is started from, causes empty lines below the cmdline. (raf)
8   X11 GUI: When menu is disabled by excluding 'm' from 'guioptions', ALT key
    should not be used to trigger a menu (like the Win32 version).
8   When setting 'langmenu', it should be effective immediately. Store both
    the English and the translated text in the menu structure. Re-generate
    the translation when 'langmenu' has changed.
8   Basic flaw in the GUI code: NextScreen is updated before calling
    gui_write(), but the GUI code relies on NextScreen to represent the state
    of where it is processing the output.
    Need better separation of Vim core and GUI code.
8   When fontset support is enabled, setting 'guifont' to a single font
    doesn't work.
8   Menu priority for sub-menus for: Athena, Amiga, Mac, VMS, BeOS.
8   Add menu separators for Athena, Amiga, Mac, RISCOS.
8   Add way to specify the file filter for the browse dialog. At least for
    browse().
8   Add dialog for search/replace to other GUIs? Tk has something for this,
    use that code? Or use console dialog.
8   When selecting a font with the font dialog and the font is invalid, the
    error message disappears too quick.
8   gui_check_colors() is not called at the right moment. Do it much later,
    to avoid problems.
8   gui_update_cursor() is called for a cursor shape change, even when there
    are mappings to be processed. Only do something when going to wait for
    input. Or maybe every 100 ms?
8   X11: When the window size is reduced to fit on screen, there are blank
    lines below the text and bottom scrollbar. "gvim -geometry 80x78+0+0".
    When the "+0+0" is omitted it works.
8   When the character cell of some fonts is different (ascent/descent), the
    cursor leaves an underline behind (Hiebert).
8   When starting an external command, and 'guipty' set, BS and DEL are mixed
    up. Set erase character somehow?
8   A dead circumflex followed by a space should give the '^' character
    (Rommel). Look how xterm does this.
    Also: Bednar has some code for dead key handling.
    Also: Nedit 5.0.2 with USE_XMIM does it right. (Gaya)
8   The compose key doesn't work properly (Cepas). Both for Win32 and X11.
7   The compiled-in highlight defaults allocate colors, which will never be
    freed. Move them to a startup script? Only use bold and underline for
    the compiled-in defaults.
7   The cursor in an inactive window should be hollow. Currently it's not
    visible.
8   With wrapping lines, clicking below the scrollbar thumb moves more than a
    screenfull of lines. Adjust the thumb size when lines wrap.
7   GUI on Solaris 2.5.1, using /usr/dt/..: When gvim starts, cursor is
    hollow, after window lowered/raised it's OK. (Godfrey)
7   When starting GUI with ":gui", and window is made smaller because it
    doesn't fit on the screen, there is an extra redraw.
8   When setting font with .Xdefaults, there is an extra empty line at the
    bottom, which disappears when using ":set guifont=<Tab>". (Chadzelek)
8   When font shape changes, but not the size, doing ":set font=" does not
    redraw the screen with the new font. Also for Win32.
    When the size changes, on Solaris 2.5 there isn't a redraw for the
    remaining part of the window (Phillipps).
-   Flashes really badly in certain cases when running remotely from a Sun.

MSDOS/DJGPP:
8   Win32: Opening a file dir\&file doesn't work. Also \^. Do we care?
8   DJGPP: "cd c:" doesn't work. Works OK in 16-bit version. mch_getperm()

    returns -1 for "c:".
9   The 16 bit version runs out of memory in test 31.
9   DOS: Make CTRL-Fx and ALT-Fx work.
    CTRL-F1 = CE-5E, CTRL-F2 = CE-5F, .., CTRL-F10 = CE-67
    ALT-F1 = CE-68, ALT-F2 = CE-69, .., ALT-F10 = CE-71
    Shifted cursor keys produce same codes as unshifted keys. Use bioskey(2)
    to get modifier mask for <S-C-M-Fx>.
    Use CSI codes to insert modifier mask in input stream? Make this work
    like in the GUI, but do handle a typed CSI.
    Mapping things like <M-A> doesn't work, because it generates an extended
    key code. Use a translation table?
9   Can't read an opened swap file when the "share" command has not been used.
    At least ignore the swap files that Vim has opened itself.
8   setlocale() is bogus.
8   Vim busy waits for new characters or mouse clicks. Should put in some
    sort of sleep, to avoid eating 50% of the CPU time. Test on an unpatched
    Windows 95 system!
8   DJGPP: when shell is bash, make fails. (Donahoe)
7   Hitting CTRL-P twice quickly (e.g. in keyword completion) on a 8088
    machine, starts printer echo! (John Mullin).
7   MSDOS 16 bit version can't work with COMSPEC that has an argument, e.g.:
    COMSPEC=C:\WINDOWS\COMMAND.COM /E:4096   (Bradley)
    Caused by BCC system() function (Borland "make" has the same problem).
8   Makefile.bor can't compile xxd and ctags without editing the makefiles.
    Create a Makefile.bor for xxd and ctags.
8   Check if with DJGPP 2.01 the problem of a path starting with a backslash is
    still present.
8   Mouse: handle left&right button pressed as middle button pressed. Add
    modifier keys shift, ctrl and alt.
7   When too many files are open (depends on FILES), strange things happen.
    The Dos16 version runs out of memory, in the Dos32 version "!ls" causes a
    crash. Another symptom: .swp files are not deleted, existing files are
    "[New file]".
7   DJGPP version doesn't work with graphics display mode. Switch to a mode
    that is supported?
8   DJGPP: ":mode" doesn't work for many modes. Disable them.

MSDOS, OS/2 and Win32:
8   OS/2: Add backtick expansion. Undefine NO_EXPANDPATH and use
    gen_expand_wildcards().
8   OS/2: Add clipboard support? See example clipbrd.exe from Alexander
    Wagner.
8   OS/2: Add Extended Attributes support and define HAVE_ACL.
8   Win32 console: <M-Up> and <M-Down> don't work. (Geddes) We don't have
    special keys for these. Should use modifier + key.
8   Environment variables in DOS are not case sensitive. Make a define for
    STRCMP_ENV(), and use it when comparing environment var names.
8   Setting 'shellslash' has no immediate effect. Change all file names when
    it is set/reset? Or only use it when actually executing a shell command?
8   When editing a file on a Samba server, case might matter. ":e file"
    followed by ":e FILE" will edit "file" again, even though "FILE" might be
    another one. Set last used name in buflist_new()? Fix do_ecmd(), etc.

Windows 95:
8   Editing a file by it's short file name and writing it, makes the long file
    name disappear. Use Unix method for making a backup file? Better: make
    the way the backup file is made (copy or rename) an option, not a #define
    (also needed for OS/2, the icon goes to the backup file).

    Use FindFirstFile()->cAlternateFileName in fname_case() (George).
8   Doing wildcard expansion, will match the short filename, but result in the
    long filename (both DJGPP and Win32).

Win32 console:
9   When editing a file by its short file name, it should be expanded into its
    long file name, to avoid problems like these: (Mccollister)
     1) Create a file called ".bashrc" using some other editor.
     2) Drag that file onto a shortcut or the actual executable.
     3) Note that the file name is something like BASHRC~1
     4) Go to File->Save As menu item and type ".bashrc" as the file name.
     5) Press "Yes" to indicate that I want to overwrite the file.
     6) Note that the message "File exists (use ! to override)" is displayed
        and the file is not saved.
    Use FindFirstFile() to expand a file name and directory in the path to its
    long name.
8   Also implement 'conskey' option for the Win32 console version? Look at
    how Xvi does console I/O under Windows NT.
7   Re-install the use of $TERM and support the use of different terminals,
    besides the console.
8   Use of <altgr> modifier doesn't work? 5.3 was OK. (Garcia-Suarez/Guckes)
9   Mapping <C-S-Tab> doesn't work correctly.
9   tmpnam() uses file in root of file system: "\asdf". That doesn't work on
    a Netware network drive. Use same function as for Win32 GUI?
8   In os_win32.h, HAVE_STRICMP and HAVE_STRNICMP are defined only if __GNUC__
    is not defined. Shouldn't that be the other way around?
9   When using libcall() for a function that returns an invalid pointer, Vim
    crashes. Check for a bad pointer with isBadReadPtr(). (Zeitlin)
    Doesn't appear to work really, at least check for "1".

Amiga:
9   In mch_expandpath() a "*" is to be expanded, but "*" isn't. Remove
    backslashes in result.
8   Executing a shell, only one option for 'shell' is separated. Should do
    all options, using white space separation.

Macintosh:
8   Define vim_mkdir() for Macintosh.
9   Explorer.vim isn't working properly.
9   When DiskLock is running, using a swap file causes a crash. Appears to be
    a problem with writing a file that starts with a dot. (Giacalone)
9   On G3 Mac, OS version 8, control strip causes characters messed up when
    scrolling (CTRL-L cleans it up). (Benji Fisher)
9   On G3 Mac, OS version 8, variable-speed scrolling doesn't work, after two
    seconds of scrolling the screen freezes. (Benji Fisher)
    scrolling (CTRL-L cleans it up). (Benji Fisher)
9   In mac_expandpath() check that handling of backslashes is done properly.
9   Executable is called "vimPPC" instead of "gvim"? (Amerige)
8   Standard Mac buttons and shortcuts are missing. No close button.
    (Amerige)
8   An invocation of gvim hands over control to an existing gvim. (Amerige)
8   Handling of non-fixed width fonts is wrong. (Amerige)
8   StatusLine and StatusLineNC highlighting isn't right. (Amerige)

VMS:
8   VMS: Inserts <NL> every 8291 bytes when writing. (Howie) 4.5 didn't have

this problem. It's caused by the write() function, need to write() every line separately. It seems read() also returns a single line. Switch I/O to binary mode somehow?
7 Lots of code in common with os_unix.c, but many fixes are missing. Move shared code to a common file? Use one file with #ifdefs?

"Small" problems:
8 For a "/*" "**" "*/" comment the end isn't aligned properly.
Using "==" the middle isn't aligned as 'comments' indicates.
:set comments=sr:/*,mb:**,ex:*/,://
8 A nested include file should also be found relative to the file that includes it. "lev1/foo.h" includes "lev2/bar.h" which is "lev1/lev2/bar.h". Should pass current file name on from find_pattern_in_path() to get_file_name_in_path(). (Angus Mackay)
9 When jumping to a tag, the search pattern is put in the history. When 'magic' is on, the pattern may not work. Translate the pattern depending on p_magic when putting it in the history? Alternative: Store value of 'magic' in history. (Margo)
9 Viminfo file becomes corrupt when editing a file with a <NL> in the name. (file marks, buffer list, history of marks) (Alexander N.Benner, Wichert Akkerman, Weisselberg)
Also problems with buffer menu.
9 optwin.vim: Restoring a mapping for <Space> or <CR> is not correct for ":noremap". Add "mapcmd({string}, {mode})? Use code from ":mkexrc".
9 When starting gvim in an xterm, Vim sends t_vi and t_ve to the terminal (cursor invisible/visible). Should not happen.
9 incsearch is incorrect for "/that/<Return>/this/;//" (last search pattern isn't updated).
9 term_console is used before it is set (msdos, Amiga).
9 Get out-of-memory for ":g/^/,$s//@/" on 1000 lines, this is not handled correctly. Get many error messages while redrawing the screen, which cause another redraw, etc.
9 When first editing file "test", which is symlink to "test2", and then editing "test2", you end up editing buffer "test" again. Change the name of the buffer to the actual file, instead of using the name of the symlink?
8 [<C-I> doesn't work when '*' is in 'iskeyword'. find_pattern_in_path() must escape special characters in the pattern.
8 Vim can overwrite a read-only file with ":w!". ":w" can't overwrite an existing file, "w!" can, but perhaps not a read-only file? Then use ":w!!" for that.
Or ask for permission to overwrite it (if file can be made writable) and restore file to readonly afterwards.
8 Buffers menu, when torn-off, disappears when being refreshed.
8 Is calling msg_start() in main() really needed? Any printed message should include it already.
8 When in Insert mode with 'scrolloff' set, inserting text at the end of the file, "!" lines will stay there while they could be used. (Park)
7 When compiled with "xterm_clipboard", startup can be slower and might get error message for invalid $DISPLAY. Try connecting to the X server in the background (forked), so that Vim starts up quicker?
8 For xterm need to open a connection to the X server to get the window title, which can be slow. Can also get the title with "<Esc>[21t", no need to use X11 calls. This returns "<Esc>]l{title}<Esc>\".
8 Add term entries for function keys on xterm with alt and ctrl (new in pl 94). E.g., Control adds ";5" in "<Esc>[20;5~". Find a generic way to prepend a modifier in console mode, to avoid having to specify each individual modified key.
8 When the builtin xterm termcap contains codes that are not wanted, need a

way to avoid using the builtin termcap.
8 '[and '] should be set to start/end of line when using a linewise operator (e.g., ":w").
8 CTRL-A can't handle big "long" numbers, they become negative. Check for "-" character, if not present, use unsigned long.
8 Make it possible to disable the special meaning of "#" in the first column for ">>".
8 Add suspending with CTRL-Z at the "more" prompt, and when executing a long script in do_cmdline().
8 When using 'hidden', many swap files will be open. When Vim runs into the maximum number of open files, error messages will appear. Detect that this problem is present, and close any hidden files that don't have changes.
8 With 'viminfo' set such that the ".viminfo" file is written on a FAT filesystem, an illegal file name may be created: ".vim".
8 For each buffer that is opened, the viminfo file is opened and read to check for file marks. This can be slow.
7 "dd" on the last line of the file, causes the last but one line to be redrawn. Should not be necessary.
7 Using "gqj" near the last line of the window, makes the window scroll up, even though this isn't necessary.
8 Should be able to compile Vim in another directory, with $(srcdir) set to where the sources are. Add $(srcdir) in the Makefile in a lot of places. (Netherton)
8 Perl adds arguments to the compiler. Check that compiling a program still works after that, otherwise following configure checks will fail.
7 In xterm, recognize both vt100 and vt220 cursor keys. Change add_termcode() to not remove an existing entry for a name, when it's needed.
Recognize <C-Left> and <C-Right> in new xterm.
Need a generic solution to recognize different codes for the same key.
8 Core dump within signal function: gdb doesn't show stack backtrace! Option to skip catch_signals()?
8 Pasting with the mouse in Replace mode inserts the text, instead of overwriting, when it is more than one line. Same for using <C-R>.
7 When using search history, the trailing '/' or '?' needs to be changed depending on the search command. When there was no '/' or '?', add one, so flags can be added easily?
7 CTRL-F at the end of the file, when 'so' is set, redraws twice. When the window is small (< 2 * 'so') it behaves differently.
9 CTRL-E and CTRL-Y don't work in small window when 'so' is 4 and lines are wrapping (Acevedo/in.226). E.g., when using CTRL-E, window height 7, window might actually scroll down when last line of buffer is displayed.
--> Remember if the previous command was "cursor follows screen" or "screen follow cursor" and use this in cursupdate().
9 Text is scrolled up and down: (acevedo)
- set scrolloff to something >1 (i always use "set so=4")
- set the win. hegiht to scrolloff + 1 (i'd do ^W5_)
- load a file with scrolloff + 2 lines
- put the cursor in line scrolloff +1 (5G)
- if you move the cursor along the line (h, l, insert text, etc)
the text is crolled with each keystroke (once up, once down,...)
sometimes with <Esc> the text is scrolled up and down without stop!!!!
7 tilde_replace() can only handle "~/", should also do "~user/".
Get the list of home directories (from /etc/passwd? Use getpwent()) and use some clever algorithm to match a path with that. Find common strings in the list?
8 Add 'o' flag to 'mouse'?
8 When dragging status line with mouse, sometimes a jump when first clicking on the status line (caused by 'winheight'). Select window on button up,

    instead of on button down.
7   There are often a few <CR> on a row sent to the screen, should not be
    necessary.
8   When performing incremental search, should abort searching as soon as a
    character is typed.
8   Make CTRL-C on Unix generate a signal, avoid using select() to check for a
    CTRL-C (it's slow). Allow using CTRL-C for "*y, like in MS-Windows.
8   When writing viminfo file, handle CTRL-J in filename (needs to be escaped
    with CTRL-V). (Acevedo)
8   When starting Vim, switch terminal to RAW mode asap, so typeahead is
    handled correctly (without the need for a <CR>).
8   When the value of $MAKE contains a path, configure can't handle this.
    It's an autoconf bug. Remove the path from $MAKE to work around it.
8   How to set VIMRC_FILE to \"something\" for configure? Why does this not
    work: CFLAGS='-DVIMRC_FILE=\"/mydir/myfile\"' ./configure
8   The temporary file is sometimes not writable. Check for this, and use an
    alternate name when it isn't. Or add the 'temptemplate' option: template
    for the temp file name ":set temptemplate=/usr/tmp/?????.tmp".
    Also: Win32 version uses Windows temp directory, which might not work for
    cygwin bash.
7   Get error "*, \+ or \(operand could be empty" for pattern "\(.\)\1\{3}".
    Don't check when compiling, check while matching?
7   When switching to Daylight Saving Time, Vim complains that a file has been
    changed since last read. Can we use a function that uses GMT?
7   When completing an environment variable after a '$', check for file names
    that contain a '$' after all have been found.

I can't reproduce these (if you can, let me know how!):
-   With backupdir=~/tmp,/tmp nowritebackup nobackup patchmode=.smr and the
    patchmode file exists, backup files in the tmp directory are left lying
    around - they should be deleted! (Riehm)
9   Crash when changing fonts in Athena (Hiebert). gui_mch_set_scrollbar_pos
    gets an "sb" argument that points to all zero entries...?
9   Crash in Motif GUI, when pasting text with middle mouse button while in
    Insert mode. SunOS 4.1.3 only? (Janssen).
8   Motif: Tear-off menu item crashes Vim on some machines. (Netherton) It
    works fine for me, maybe it's a Motif problem.
-   On Diamond Viper 132x43 mode crash on exit (John Lange)
9   NT 4.0: Editing ".bashrc" (drag and drop), file disappears.
    Editing ".xyz" is OK. (McCollister)
8   Win32 GUI: NT: When regaining focus at the more prompt, there is an extra
    newline. (Webb)
8   Win32 GUI: Writable File in read-only dir can be deleted but not written?
    (Canup)
9   GUI: Between version 5.0h and 5.0i the Home and End keys stopped working
    for Micheal Schulz (Linux and AIX).
9   Win32 console: <M-S-Left> causes a crash. (Geddes)
9   In MSDOS version(s): typing ESC three or four times crashes Vim, when
    'visualbell' is set (Kielhorn).

[These have been reported for version 3.0, they may not appear in this version]
-   MSDOS: After writing on a network the 'w' bit for others is set.

Problems that will (probably) not be solved:
-   X windows: When $DISPLAY points to a X server where there is no access
    permission, trying to connect to the X server causes an error message.
    XtOpenDisplay() prints this directly, there is no way to avoid it.

- X windows: Setting 'guifontset' to an illegal value sometimes crashes Vim.
  This is caused by a fault in a X library function, can't be solved in Vim.
- Moving the cursor removes color in color-xterm. This is a color-xterm
  problem! color-xterm ver. 6.1 beta 3 and later work properly.
- The Dos32 version (DJGPP) can't use long file names on Windows NT.
- In zsh, "gvim&" changes the terminal settings. This is a zsh problem.
  (Jennings)
- Problem with HPterm under X: old contents of window is lost (Cosentino).
- Amiga: When using quickfix with the Manx compiler we only get the first 25
  errors. How do we get the rest?
- Amiga: The ":cq" command does not always abort the Manx compiler. Why?
- Linux: A file with protection r--rw-rw- is seen readonly for others. The
  access() function in GNU libc is probably wrong.
- MSDOS: When using smartdrive with write-back buffering, writing to a
  readonly floppy will cause problems. How to test for a writable floppy
  first?
- MSDOS: Both 16 and 32 bit versions: File name expansion doesn't work for
  names that start with a dot. These used to be illegal file names.
- When doing a CTRL-Z and typing a command for the shell, while Vim is busy
  (e.g. writing a file), the command for the shell is sometimes eaten by Vim,
  because the terminal mode is changed from RAW to CBREAK.
- An old version of GNU tgoto can't handle the terminfo code for "AF". The
  "%p1" is interpreted as "%p" and "1", causing color not to be working.
  Fix: Change the "%p1" in the "AF" and "AB" terminfo entries to "%p".
  (Benzinger).
- When running an external command from the GUI, typeahead is going to that
  program, not to Vim. It looks like the shell eats the characters, Vim
  can't get back what the external command didn't use.
- Win32 GUI: Error code from external command not returned in shell_error.
  It appears that cmd.exe and command.com don't return an error code.
- Win32 GUI: The Toolbar is a bit too high when the flat style is being
  used. We don't have control over the height of the Toolbar.
- SunOS 5.5.1 with Motif: The file open dialog does not have a horizontal
  scroll bar for the "files" selection. This is a problem in the Motif
  libraries, get a patch from Sun.
- Solaris 2.6 with GTK and Perl: gvim crashes when started. Problem with X
  input method called from GDK code. Without Perl it doesn't crash.
- Win32 GUI: mouse wheel always scrolls rightmost window. The events arrive
  in Vim as if the rightmost scrollbar was used.
- GTK with Gnome: Produces an error message when starting up:
      Gdk-WARNING **: locale not supported by C library
  This is caused by the gnome library gnome_init() setting $LC_CTYPE to
  "en_US". Not all systems support this locale name, thus causing the
  error. Hopefully a newer version of GTK/Gnome fixes this problem.

--------------------- extensions and improvements ----------------------
extensions-improvements

Documentation:
9 Merge in ideas from ~/vim/patches/tutor.txt (Gabriel Zachmann)
9 Update the overview of options in quickref.txt.
9 Errors in doc to HTML conversion:
   autocmd.txt "like \| in a |pattern|"
9 Add 'fold' section to quick reference.
9 Add new options to quick reference.
9 Change all tags for commands from i_<C-R>_<C-O> to i_CTRL-R_CTRL-O to
   avoid giving the user the impression that all commands have a tag in the
   <C-x> form.
9 Extend the User Manual. Use portions of the Reference Manual.
9 Make the Reference Manual more precise. For each command mention:

    - change to cursor position and curswant
    - if it can be undone (u/CTRL-R) and redone (.)
    - how it works for folded lines
    - how it works with multi-byte characters
9   In change.txt, remark about Javadoc isn't right. Right alignment would
    work too.
8   Spread the windows commands over the other files. For example, ":stag"
    should be with ":tag". cross-link with tags to avoid too much double
    text.
8   make vimtutor.bat for DOS and vimtutor script for Amiga. Then add
    vimtutor.man to the runtime archive.
8   Add an example vimrc for using Vim as a modeless editor. Emulate an
    existing editor (WordStar, Emacs, Brief?).
7   Windows: When a wrong command is typed with an ALT key, give a hint to
    look at the help for 'winaltkeys'.
-   Check text editor compendium for vi and Vim remarks.
-   Add some text for xxd.

Help:
-   First try using the ":help" argument literally, before using it as a
    pattern. And then match it as part of a tag.
-   When a help item has multiple matches make it possible to use ":tn" to go
    to the other matches.
-   Use GNU Texinfo for the help files somehow? Use Info format files?
    At least the other way around: Use Vim to view (and edit) .info files.
-   Default mapping for help files: <Tab> to position cursor on next |:tag|.
-   When hitting <Esc> or CTRL-C in Normal mode, give a message to help novice
    users to get out: "Type :q! to quit Vim".
-   Implement a "sticky" help window, some help text lines that are always
    displayed in a window with fixed height. (Guckes) Use "~/.vimhelp" file,
    user can edit it to insert his favorite commands, new account can contain a
    default contents.
-   When entering the help window, don't resize to 'helpheight' if the user has
    reduced the size before (Webb). Keep the window height that the user set
    the window to in a "preferred window height" entry for each window.
-   ":help :s^I" should expand to ":help :substitute".
-   Make the help key (<F1>) context sensitive?
-   Learn mode: show short help while typing commands.

"make test":
-   Find a way to skip tests that can't be done. For non-Unix systems, but
    also for e.g. the Perl interface.

Folding:
    (commands still available: zg zG zI zJ zK zp zP zq zQ zV zw zW zy zY;
    secondary: zB zS zT zZ)
-   When pressing the down arrow of a scrollbar, a closed fold doesn't scroll
    until after a long time. How to make scrolling with closed folds
    smoother?
-   When a buffer becomes hidden, keep the manual folds for it?
-   'foldmethod' "textobject": fold on sections and paragraph text objects.
-   Add 'hidecomment' option: don't display comments in /* */ and after //.
-   "zu": undo change in manual fold. "zU" redo change in manual fold. How to
    implement this?
-   "zJ" command: add the line or fold below the fold in the fold under the
    cursor.
-   'foldmethod' "syntax": "fold=3": set fold level for a region.

- Can set 'foldtext' to empty string: don't display any line. How to implement this?

Multi-byte characters:
- GTK and Win32: Allow selecting fonts for 'guifontset' with the fontselector somehow.
- GTK and Win32: make it possible to set the font for the menu to make it possible to have 'encoding' different from the current locale.
- dbcs_class() only works for Japanese and Korean. Implement this for other encodings. The "euc-jp" and "euc-kr" choices might be wrong.
- Find some way to automatically select the right GUI font or fontset, depending on the default value of 'encoding'.
  For Windows, the charset_pairs[] table could be used. But how do we know if a font exists?
- Do keyboard conversion from 'termencoding' to 'encoding' with convert_input() for Mac GUI, RiscOS GUI, BeOS GUI, VMS.
- Add mnemonics from RFC1345 longer than two characters.
  Support CTRL-K _{mnemonic}_
- Do we need the reverse of 'keymap', like 'langmap' but with files and multi-byte characters?
- When breaking a line, take properties of multi-byte characters into account. The "linebreak" program from Bruno Haible can do it:
  ftp://ftp.ilog.fr/pub/Users/haible/gnu/linebreak-0.1.tar.gz
  But it's very complicated...

Syntax highlighting:
8 Allow the user to add items to the Syntax menu sorted, without having to change this for each release.
8 Add a "matchcontains" for regions: items containd in the start or end pattern, but not in the body.
8 Add a "keepend-contained" argument: Don't change the end of an item this one is contained in. Like "keepend" but specified on the contained item, instead of the containing item.
8 For keywords, allow to define the size of the hash table with ":syn clear". Change KHASH_ defines into variables stored in buffer struct. Use something else than linear linked list from the hash table. (Campbell)
8 cpp.vim: In C++ it's allowed to use {} inside ().
8 Some syntax files set 'iskeyword'. When switching to another filetype this isn't reset. Add a special keyword definition for the syntax rules?
8 Add specific syntax item to match with parens/braces that don't have a "%" match. :syntax nomatch cMatchError (,{,[,),},] [contained]
8 Highlight the text between two matching parens (e.g., with a grey background) when on one of the parens or in between them.
8 Add a command to jump to the next character highlighted with "Error".
8 When using a cterm, and no ctermfg or ctermbg are defined, use start/stop sequences. Add remark in docs that :if 'term' == "term-name" should be used.
8 Add @spell cluster to String and Comment groups for many languages. Will allow spell checking. (Fleiner)
8 When listing syntax items, try to sort the keywords alphabetically. And re-insert the [] if possible.
8 Make it possible to use color of text for Visual highlight group (like for the Cursor).
8 "fg" and "bg" don't work in an xterm. Get default colors from xterm with an ESC sequence. Ideas in: ~/vim/patches/vikas.xtermcolors .
8 Make it possible to only highlight a sub-expression of a match. Like using "\1" in a ":s" command.
8 Support for deleting syntax items:

```
    :syn keyword cTodo remove this
    :syn match cTodo remove "pattern"
    :syn region cString remove start="this" end="that"
8   Add possibility to sync on something else, when the syncing in one way
    doesn't find match.  For HTML: When no {script} is found, try looking for
    a '<'.  (Fleiner)
7   Replace the synchronizing method with a state machine specification?
    Should be able to start at any line in the file, search forwards or
    backwards, and use the result of matching a pattern.
7   Use parsing like awk, so that e.g., a ( without a matching ) can be
    detected.
8   Make it possible to use "inverted" highlighting, invert the original
    character.  For Visual mode. (xterm-selection already does this).
8   Highlight non-printable characters with "SpecialChar", linked to
    "Special".  Display them with the digraph characters, if possible.
8   Highlight the clipboard-selection with a highlight group.
8   Be able to reset highlighting to its original (default) values.
7   Be able to write current highlighting to a file as commands, similar to
    ":mkvimrc".
8   Improve c.vim:
    - Add check for unterminated strings, with a variable to switch it on:
      "c_strict_ansi".
    - Detect unbalanced "#endif".  Requires looking back a long way...
8   Add an option to restrict the updating of syntax highlighting to the
    current line while in Insert mode.
8   When guessing value of 'background', the syntax file has already been
    loaded (from the .gvimrc).  After changing 'background', load it again?
8   Add ":syn resync" command, to re-parse the whole file until the current
    display position.
8   Should support "me" offset for a region start pattern.  To be used to
    allow searching for the end pattern inside the match of the end pattern.
    Example: syn region pikeXX start="([^{]" end=")" should work on "()".
8   When using a regexp for "contains=", should delay matching with it until
    redrawing happens.  Set a flag when a group is added, check this flag when
    highlighting starts.
7   Add "semitrans": Add highlighting.  E.g., make the text bold, but keep the
    colors.  And add colors, so that Green+Red becomes Yellow.
    E.g. for this html:
        <B> bold text <I> italic+bold text </B> italic text </I>
7   Wild idea: Not only set highlighting, but also change what is displayed
    (e.g., remove characters, so that "<B>bold</B>" can be shown as "bold"):
        :syn region boldstuff start="<B>" display="" end="</B>" display=""
7   CTRL-] checks the highlight group for finding out what the tag is.
7   Add an explanation how a list of words can be used to highlight misspelled
    words.
8   Add spell checking.  Use "ispell -a" somehow.
    ~/vim/patches/wm_vim-5_4d.zip  can be used as an example (includes ispell
    inside Vim).
7   Command line completion for ":find" should search in 'path'.
8   Add more command line completion for :syntax.
8   Add more command line completion for :highlight.
7   Should find a better way to parse the :syntax and :highlight commands.
    Use tables or lists that can be shared by parsing for execution and
    completion?
7   Add a few sets of colors (e.g. Borland Turbo C one).  With a menu to
    select one of the sets.
8   Add offsets to sub-matches: "\(a*\) *"he=e1-1
    'e' is end of match 'e1' is end of sub-match 1, 's2' is start of submatch
    2, etc.
8   In Insert mode, when there are typeahead characters, postpone the
```

highlighting (for "." command).
8 Syncing on comments isn't 100% correct when / / lines mix with / * and * /.
For example: What about a line that starts with / / and contains * /?
8 Ignore / * and * / inside strings, when syncing.
7 Build a few more syntax files from the file "/usr/share/misc/vgrindefs":
ISP, LDL, Icon, ratfor. And check "nedit/source/highlight.c".
6 Add possibility to have background color continue until the right edge of
the window. Useful for comment blocks and function headings. (Rogall)
- Make it possible to add "contains" items for all items in a group. Useful
when extending an already existing syntax file.
- Add line-continuation pattern for non-syncing items too?
- Add possibility to highlight specific columns (for Fortran). Or put a
line in between columns (e.g. for 'textwidth').
- Add possibility to highlight the whole line, including the right margin
(for comment blocks).
- Add 'hlmatch' option: List of flags:
'c': highlight match for character under the cursor.
'b': highlight the previous (, and its match.
'a': highlight all text from the previous (until its match.
Also for {}, <>, etc.?
'e': highlight all braces without a match (slow?)
OR: add an argument "cursor" to the syntax command, which means that the
region/match/keyword is only highlighted when the cursor is on it.
(Campbell)
Or do it like Elvis: define text objects and how to highlight them around
the cursor. (Iain Truskett)
7 Make it possible to use all words in the tags files as Keyword.
Can also be done with a script (but it's slow).
7 Make it possible to call a ":" command when a match is found. Should
allow for adding keywords from the text (e.g. variables that are set).
And allows for sections with different highlighting.
7 Add highlight group for commandline: "Commandline". Make sure it
highlights the command line while typing a command, and any output from
messages. And external commands?
8 Make a version that works like less, but with highlighting: read stdin for
text, exit at end of file, don't allow editing, etc. moreim? lessim?

Built-in script language:
8 Add referring to key options with "&t_xx". Both for "echo &t_xx" and
":let &t_xx =". Useful for making portable mappings.
8 Allow range for ":exec". Pass it on to the executed command. (Webb)
8 Add a way to catch errors:
```
:try
:  [commands that can fail]
:catch [error-type]
:  [error handling]
:endtry
```
7 Be able to call a function while passing on a variable number of
arguments:
```
:function Foo(abc, ...)
:  call Bar(a:abc, ...)    of call Bar(a:abc, a:*)
```
8 Have a look at VSEL. Would it be useful to include? (Bigham)
8 Add ":fungroup" command, to group function definitions together. When
encountered, all functions in the group are removed. Suggest using an
obscure name to avoid name clashes. Require a ":fungroup END" in the same
sourced file? Assume the group ends at the end of the file. Handle
nested packages?
Alternative: Support packages. {package-name}:{function-name}().
Packages are loaded automatically when first used, from

$VIMRUNTIME/packages (or use a search path).
7 Pre-parse or compile Vim scripts into a bytecode. Put the bytecode with the original script, with an ":if has('bytecode')" around it, so that it's only used with a Vim that supports it.
8 The input() function should not read mapped characters, but always get characters from the user (or a script). Need to change vgetorpeek() to ignore mapped characters.
8 Add functions:
```
    cursor(lnum, col)       Position cursor (when lnum or col 0 keep val)
    multibyteidx(string, idx) Byte index in multi-byte character.
    menuname({menu_name}, {idx})  get name of menu.  menuname("", 1)
                            returns "File", menuname("File", 1) returns
                            "Open...".
    menuarg({menu_name})    return argument of a menu item by name.
    mapname({idx}, mode)    return the name of the idx'th mapping.
    match({pat}, {string}, [count]) get index of count'th match
    sprintf(format, arg, ..) How to prevent a crash???
    attributes()            return file protection flags "drwxrwxrwx"
    perl(cmd)               call Perl and return string
    shorten(fname)          shorten a file name, like home_replace()
    input(prompt, complete) like input() but do specified completion
    tagtype(tag)            get type of tag (also checks if it exists)
    filewritable()          like filereadable()
    getfperm()              file permissions, in form "rwxrwxrwx"
    getftype()              "file", "dir", "link", "other"?
    getbufvar({bufid}, {name})
    setwinvar()
    getwinvar()
    search()                Add optional offset argument.
    libcall()               Allow more than one argument.
    libcallnr()             Like libcall(), but function returns a number.
    fnamemod({fname}, {mod})    {mod} can be ":p:h" etc., to be applied to
                            {fname}
    getchar()               get one character from the user.
    confirm()               add "flags" argument, with 'v' for vertical
                            layout and 'c' for console dialog.  Also add
                            'c' to 'guioptions' (for all dialogs) (Haegg)
    filter({cmd} [, {string}])  Filter {string} through the shell command
                            {cmd} and return the result.  If {string} is
                            omitted no input is given to {cmd}.
    getkey()                Like input() but get only one char.
    raisewin()              raise gvim window (see HierAssist patch for
                            Tcl implementation ~/vim/HierAssist/ )
```
8 argc() returns 0 when using "vim -t tag". How to detect that no file was specified in any way? To be able to jump to the last edited file.
8 When starting to source a vim script, delete all functions that it has previously defined? Avoids using ":fun!" all the time.
7 Add optional arguments to user functions:
```
    :func myFunc(arg1, arg2, arg3 = "blah", arg4 = 17)
```
6 User functions: Functions local to buffer "b:func()"?
8 Add ":let var[{expr}] = {expr}". When past the end of "var" just ignore.
8 The "= register should be writable, if followed by the name of a variable, option or environment variable.
8 ":let &option" should list the value of the option.
7 Add synIDlist(), making the whole list of syntax items on the stack available (separated with '\n').
8 Add autocommand-event for when a variable is changed:
```
    :au VarChanged {varname} {commands}
```
8 Add "has("gui_capable")", to check if the GUI can be started.
8 Add possibility to use variables like registers: characterwise (default),

linewise (when ending in '\n'), blockwise (when ending in '\001'). reg0, rega, reg%, etc. Add functions linewise({expr}), blockwise({expr}) and charwise({expr}).

7 Make it possible to do any command on a string variable (make a buffer with one line, containing the string). Maybe add an (invisible) scratch buffer for this?

```
result = scratch(string, command)
result = apply(string, command)
result = execute(string, command)
```

"command" would use <> notation.
Does scratch buffer have a number? Or re-use same number?

7 Add function to generate unique number (date in milliseconds).

7 Automatically load a function from a file when it is called. Need an option for the search path. (Sekera)

7 Include support for arrays? Patch from Robert Webb.
This is restricted to fixed-size arrays indexed by number. Better: Use associative arrays: a[5] = 3, a["some"] = 'x'. Implement by translating into ordinary variables: a[5] is "-a-5", a["some"] is "a-some", a[5][6] is "a-5-6". But how to do array assignment and concatenation?

Code cleanup:

Performance:

8 Avoid alloc() for scratch buffer use, esp. in syntax.c. It's very slow on Win16.

9 Setting GUI options in the console (e.g., 'guifont') should not cause a redraw.

8 Profiling shows that in_id_list() is used very often for C code. Can this function be improved?

8 For an existing file, the page size of the swap file is always the default, instead of using the block size of the device, because the swap file is created only after setting the block size in mf_open(). How can this be improved?

8 Set default for 'ttyscroll' to half a screen height? Should speed up MS-DOS version. (Negri)

7 Check how performance of loading the wordlist can be imrpoved (adding a lot of abbreviations).

7 DOS console: Add t_DL support, to make scrolling faster.

7 Compile Ex commands to byte codes. Store byte codes in a vim script file at the end, after "compiled:. Make it look like a single comment line for old Vim versions. Insert first line "Vim script compiled <timestamp>.
Only used compiled code when timestamp matches the file stat.
Add command to compile a vim script and add it to the file in-place.
Split Ex command executing into a parsing and executing phase.
Use compiled code for functions, while loops, etc.

8 When editing a file with extremely long lines (e.g., an executable), the "linerest" in readfile() is allocated twice to be able to copy what was read so far. Use realloc() instead? Or split the line when allocating memory fails and "linerest" is big (> 100000)?

8 When defining autocommands (e.g., from $VIMRUNTIME/filetype.vim), need to compare each pattern with all existing patterns. Use a hash code to avoid using strcmp() too often?

7 Include turbo_loader patches, speeding up reading a file?
Speed up reading a file by reading it into a fixed-size buffer, creating the list of indexes in another buffer, and then copying the result into a memfile block with two copies. Then read the next block into another fixed-size buffer, create the second list of indexes and copy text from the two blocks to the memfile block.

7 do_cmdline(): Avoid that the command line is copied to allocated memory
  and freed again later all the time. For while loops, and for when called
  with an argument that can be messed with.
  Generic solution: Make a struct that contains a pointer and a flag that
  indicates if the pointer should be freed when replaced.
7 Check that that the file size is not more than "sizeof(long)".
- Further improve finding mappings in maphash[] in vgetorpeek()
8 Dragging the status line doesn't scroll but redraw.
8 Syntax highlighting is slow when deleting lines. Try in
  $VIMRUNTIME/filetype.vim.
- "out of memory" after deleting (1,$d) and changing (:%s/^/> /) a lot of
  lines (27000) a few times. Memory fragmentation?
- Have a look at how pdksh does memory allocation (alloc.c). (Dalecki)
- Do profiling on:
  - :g/pat/normal cmd
  - 1000ii<Esc>
  - deleting 10Mbyte worth of lines (netscape binary)
  - ":g/^/m0" on a 450Kbyte file. And the "u".
  - highlighting "~/vim/test/longline.tex", "~/vim/test/scwoop.tcl" and
    "~/vim/test/lockup.pl".
  - loading a syntax file to highlight all words not from a dictionary.
  - editing a vim script with syntax highlighting on (loading vim.vim).
7 Screen updating can be further improved by only redrawing lines that were
  changed (and lines after them, when syntax highlighting was used, and it
  changed).
  - On each change, remember start and end of the change.
  - When inserting/deleting lines, remember begin, end, and line count.
- Use macros/duarte/capicua for profiling. Nvi 1.71 is the fastest!
- When using a file with one long line (1Mbyte), then do "$hhhh", is still
  very slow. Avoid calling getvcol() for each "h"?
- Executing a register, e.g. "10000@@" is slow, because ins_typebuf has to
  move the previous commands forward each time. Pass count from
  normal_cmd() down to do_execreg().
- Repeating insert "1000i-<Esc>" displays --INSERT-- all the time, because of
  the <Esc> at the end. Make this work faster (disable redrawing).
- Avoid calls to plines() for cursor line, use w_cline_height.
- When only need to redraw the status lines in status_redraw_all(), need to
  update NOT_VALID to force the redraw. Should detect that only the status
  lines need to be redrawn (add STATUSLN between VALID and NOT_VALID?).
- After :set nowrap remove superflous redraw with wrong hor. offset if cursor
  is right of the screen.

Code size:
8 Can't optimize normal.c, because of the big switch. Make it into a table
  of functions?
8 GUI: When NO_CONSOLE is defined, more code can be excluded.
- Put getline() and cookie in a struct, so only one argument has to be
  passed to do_cmdline() and other functions.
8 move common files between os_unix.c and os_vms.c to one file?
8 Make a GUI-only version for Unix?

Messages:
8 For 'verbose' level 12 prints commands from source'ed files. How to skip
  lines that aren't executed? Perhaps move the echoing to do_cmdline()?
8 Use 'report' for ":bdel"? (Krishna) To avoid these messages when using a
  script.
8 "vim --version" output goes to stderr, should be stdout. Can all output
  from messages using printf() go to stdout?

- Delete message after new command has been entered and have waited for key.
  Perhaps after ten seconds?
- Make message history available in "msg" variables: msg1, msg2, .. msg9.
9 Check handling of overwriting of messages and delays:
  Very wrong: errors while redrawing cause endless loop.
  When switching to another file and screen scrolls because of the long
  message and return must be typed, don't scroll the screen back before
  redrawing.
7 Add an option, which is a regexp, that disables warning messages which
  match that regexp (Tsirkin).
8 When address range is wrong you only get "Invalid range". Be a bit more
  specific: Negative, beyond last line, reverse range? Include the text.
8 Make it possible to ignore errors for a moment ('errorignore'?). Another
  option to switch off giving error messages ('errorquiet'?). Also an option
  not to give any messages ('quiet')? Or ":quiet on", ":quiet off".
  Careful: For a severe error (out of memory), and when the user starts
  typing, error messages must be switched back on.
  Also a flag to ignore error messages for shell commands (for mappings).
- Option to set time for emsg() sleep. Interrupt sleep when key is typed?
  sleep before second message?
8 In Ex silent mode or when reading commands from a file, what exactly is
  not printed and what is? Check ":print", ":set all", ":args", ":vers",
  etc. At least there should be no prompt. (Smulders) And don't clear the
  screen when reading commands from stdin. (Kendall)
  --> Make a difference between informative messages, prompts, etc. and
      error messages, printing text, etc.
8 Window should be redrawn when resizing at the hit-enter prompt.
  Also at the ":tselect" prompt. Find a generic solution for redrawing when
  a prompt is present (with a callback function?).

Screen updating:
7 Add a string to the 'display' option to make CTRL-E and CTRL-Y scroll one
  screen line, also if this means the first line doesn't start with the
  first character (like what happens with a single line that doesn't fit).
- screen_line():
  - insert/delete character stuff.
  - improve delete rest of line (spaces at end of line).
- When moving or resizing window, try to avoid a complete redraw (esp. when
  dragging the status line with the mouse).
- When 'lazyredraw' set, don't echo :ex commands? Need a flag to redraw when
  waiting for a character.
8 Add a ":refresh [winnr]" command, to force updating a window. Useful from
  an event handler where ":normal" can't be used. Also useful when
  'lazyredraw' is set in a mapping.

Scrolling:
8 Add "zm" command: scroll horizontally to put the cursor in the middle.
6 Add option to set the overlap for CTRL-F and CTRL-B. (Garhi)
- extend 'scrollbind' option: 'scrollopt' words "search", "relative", etc..
  Also 'e'xecute some commands (search, vertical movements) in all bound
  windows.
- Allow scrolling by dragging with the mouse. Like the "hand" in Acrobat
  reader. With which mouse button?
- Add command to execute some commands (search, vertical movements) in all
  bound windows.
- Add 'search' option to 'scrollopt' to allow 'scrollbind' windows to
  be bound by regexp searches
- Add "z>" and "z<": scroll sideways one screenfull. (Campbell)

-   Add option to set the number of lines when not to scroll, instead of the
    fixed number used now (for terminals that scroll slow with a large number
    of lines but not with a single line).

Autoconf:
8   Should use acconfig.h to define prototypes that are used by autoheader.
8   Some compilers don't give an error for "-OPT:Olimit" but a warning. (Webb)
    Add a check for the warning, so that "Olimit" can be added automatically?
-   Autoconf: Use @datadir@ for the system independent files. Make sure the
    system dependend and system independed files are separated. (Leitner).
-   Add autoconf check for waitpid()/wait4().
-   Remove fcntl() from autoconf, all systems have it?
-   Set default for 'dictionary', add search for dictionary to autoconf.

Perl interface:
8   Rename typemap file to something else?
7   Make buffers accessed as Perl arrays. (Clark)
7   Make it possible to compile with non-ANSI C?
6   Tcl/Tk has the "load" command: load a shared library (.so or .dll).

Shared libraries:
6   Add support for loading shared libraries, and calling functions in it.
        :libload internal-name libname
        :libunload internal-name
        :liblist
        :libcall internal-name function(arg1, arg2, ...)
        :libcall function(arg1, arg2, ...)
    libcall() can have only one integer or String argument at the moment.
6   Have a look on how Perl handles loading dynamic libraries.

Tags:
7   Count before CTRL-]: jump to N'th match
8   Scope arguments for ":tag", e.g.: ":tag class:cPage open", like Elvis.
8   When output of ":tselect" is long, getting the more-prompt, should be able
    to type the tag number directly.
7   Add 'tagignorecase' option from Vile: ignore case for tags.
7   Make output of ":tselect" a bit nicer. Use highlighting?
7   Highlight the "tag 1 of >2" message. New highlight group, or same as "hit
    bottom" search message.
7   When using ":tag" at the top of the tag stack, should add another entry,
    so CTRL-T can bring you back to where you are now AND to where you were
    before the previous ":tag" command. (Webb)
7   When using CTRL-] on someClass::someMethod, separate class from method and
    use ":ta class:someClass someMethod".
    Include C++ tags changes (Bertin). Change "class::func" tag into "func"
    with "class=class"? Docs in oldmail/bertin/in.xxx.
7   Add ":tagargs", to set values for fields:
        :tagargs class:someclass file:version.c
        :tagargs clear
    These are then the default values (changes the order of priority in tag
    matching).
7   Support for "gtags" and "global"? With ":rtag" command?
    There is an example for how to do this in Nvi.
    Or do it like Elvis: 'tagprg' and 'tagprgonce' options. (Yamaguchi)
    The Elvis method is far more flexible, do it that way.
7   Support "col:99" extra field, to position the cursor in that column. With

    a flag in 'cpoptions' to switch it off again.
7   Better support for jumping to where a function or variable is used. Use
    the id-utils, with a connection to "gid" (Emacs can do it too). Add
    ":idselect", which uses an "ID" database (made by "mkid") like "tselect".
6   Don't store the search pattern from a tag command in the search history
    (with an option)?

Security:
-   nothing at the moment

Win32 GUI:
8   Make debug mode work while starting up (vim -D). Open console window for
    the message and input?
8   Add font argument to set the lfCharSet. (Bobcik)
8   Could keep console open to run multiple commands, to avoid the need to hit
    return in every console.
    Also: Look at how Emacs does runs external commands:
        http://www.cs.washington.edu/homes/voelker/ntemacs.html.
8   When dropping a file onto gvim while at the ":" prompt, insert the file
    name. Allows using the name with different commands. (Krishna)
7   Add a way to change the filter menu in the file selection dialog.
8   Need a separate PopUp menu for modeless selection. Need two new commands:
    Copy selection to clipboard, Paste selection (as typed text).
8   Dropping a file on a gvim that edits a modified buffer splits the window.
    Make an option to replace the current file (use ":e" instead of ":sp")?
    When dropping multiple files, display the first one and adjust the
    arglist. Or make it an option for the user to chose betwen ":e" and ":sp"?
8   Support copy/paste for other file formats. At least HTML, perhaps RTF.
    Add "copy special" and "paste special" commands?
7   Use <C-Tab> to cycle through open windows (e.g., the find dialog).
7   <Esc> should close a dialog.
7   Keep the console for external commands open. Don't wait for a key to be
    hit. Re-open it when the user has closed it anyway. Or use a prepended
    command: ":nowait {cmd}", or ":quiet", which executes {cmd} without any
    prompts.
7   Should be able to set an option so that when you double click a file that
    is associated with Vim, you can either get a new instance of Vim, or have
    the file added into an already running Vim.

GUI:
9   Make <S-Insert> paste from the clipboard by default. (Kunze)
7   Menu local to a buffer, like mappings. Or local to a filetype?
8   Dragging the mouse pointer outside of a Vim Window should make the text
    scroll. Return a value from gui_send_mouse_event() to the machine
    specific code to indicate the time in which the event should be repeated.
8   Make it possible to ignore a mouse click when it's used to give Vim (gvim)
    window focus. Also when a mouse click is used to bring a window to front.
8   Make the split into system independend code and system specific code more
    explicit. There are too many #ifdefs in gui.c.
    If possible, separate the Vim code completely from the GUI code, to allow
    running them in separate processes.
8   Support a background bitmap. Useful for marking a column. Patch from
    Heather Downs (GTK) and Vince Negri (Win32).
7   X11: Support cursorColor resource and "-cr" argument.
8   X11 (and others): CTRL-; is not different from ';'. Set the modifier mask
    to include CTRL for keys where CTRL produces the same ASCII code.
8   Visual highlighting should keep the same font (bold, italic, etc.).

8 Make the "Save As.." menu item work like other GUI programs: ":w file", ":e file".
8 Add 'c' flag in 'guioptios' to use console dialogs instead of GUI dialogs. Esp. for the ATTENTION prompt.
8 Add flag to 'guioptions' to not put anything in the clipboard at all?
8 Should take font from xterm where gvim was started (if no other default).
8 Selecting font names in X11 is difficult, make a script or something to select one.
8 Should support a way to use keys that we don't recognize yet. Add a command that adds entries to special_keys somehow. How do we make this portable (X11, Win32, ..)?
7 Add a flag to 'guioptions' that tells not to remove inactive menu items. For systems where greying-out or removing menu items is very slow. The menu items would remain visibly normally, but not do anything.
7 Add ":minimize" and ":maximize", which iconize the window and back. Useful when using gvim to run a script (e.g. 2html.vim).
7 X11: Is it possible to free allocated colors, so that other programs can use them again? Otherwise, allow disabling allocating the default colors. Or allocate an own colormap (check UAE). With an option to use it. For the commandline, "-install" is mostly used for X11 programs.
7 Add command line argument for "gvim" not to start the GUI. Sort of the inverse of "vim -g". (Vikas)
6 Local buffer menus. (Zachmann)
7 Should support multi-column menus.
- Should add option for where to put the "Help" menu: like Motif at the far right, or with the other menus (but still at the right).
- Add menu item to "Keep Insert mode".
8 ":mkgvimrc" command, that includes menus.
6 Big change: Move GUI to separate program "vimgui", to make startup of vim a lot faster, but still be able to do "vim -g" or ":gui".
6 Make it possible to "drag the text" (grab a character and move it up/down). Use Alt-LeftMouse for this? (Goldfarb)
7 More explicit mouse button binding instead of 'mousemodel'?
7 Add option to set the position of the window on the screen. 'windowpos', which has a value of "123,456": <x>,<y>.
Or add a command, like ":winsize"?
7 Add toolbar for non-Win32 GUIs.
7 Make it possible to put the toolbar on top, left, right and/or bottom of the window? Allows for softkey-like use.
6 Separate the part of Vim that does the editing from the part that runs the GUI. Communicate through a pseudo-tty. Vim starts up, creates a pty that is connected to the terminal. When the GUI starts, the pty is reconnected to the GUI process. When the GUI stops, it is connected to the terminal again. Also use the pty for external processes, it looks like a vt100 terminal to them. Vim uses extra commands to communicate GUI things.
- Motif steals <F10> from us, to pop up menus with the keyboard. How do we get it back if we want it?
- Paste in Insert mode should not do autowrap etc. Or maybe this should be changeable with an option?
- Put a nice picture in the icon (but how do we do that?).
7 When using a pseudo-tty Vim should behave like some terminal (vt52 looks simple enough). Terminal codes to/from shell should be translated.
- Would it be useful to be able to quit the GUI and go back to the terminal where it was started from?

Autocommands:
9 Make sure that side effects of autocommands are handled correctly. Don't execute autocommands when a buffer or window is halfway some changes.

```
    Move all apply_autocmds() calls to a higher level where needed.
8   Use another option than 'updatetime' for the CursorHold event.  The two
    things are unrelated for the user (but the implementation is more
    difficult).
8   Can't use ":normal" in CursorHold autocommands.  Make the CursorHold event
    insert a special key code, and call the autocommand functions from a
    higher level, so that vgetc() isn't used recursively.
8   Autocommands should not change registers.  And marks?  And the jumplist?
    And anything else?
8   Autocommand for when modified files have been found, when getting input
    focus again (e.g., FileChangedFocus).
    Check when: getting focus, jumping to another buffer, ...
8   Add autocommands, user functions and user commands to ":mkvimrc".
8   Add "TagJumpFile" autocommand: When jumping to another file for a tag.
    Can be used to open "main.c.gz" when "main.c" isn't found.
7   Add a way to skip an autocommand if going from one *.c file to another *.c
    file.
7   When trying to open a directory, don't load the file but trigger an
    autocommand event OpenDirectory.
7   Add file type in front of file pattern: <d> for directory, <l> for link,
    <x> for executable, etc.  <&xxx> for Risc OS.  With commas to separate
    alternatives.  The autocommand is only executed when both the file type
    AND the file pattern match. (Leonard)
5   Add option that specifies extensions which are to be discarded from the
    file name.  E.g. 'ausuffix', with ".gz,.orig".  Such that file.c.gz will
    trigger the "*.c" autocommands.  (Belabas)
7   Add something to break the autocommands for the current event, and for
    what follows.  Useful for a "BufWritePre" that wants to avoid writing the
    file.
8   Detect textmode after executing the autocommands, otherwise the .gz
    autocommands don't work for MS-DOS.  Needs to be able to remove the ^Ms at
    the end of each line.  Alternative: don't use a filter command, write the
    file, gunzip, set nobin, delete the lines, read the file.
8   Add buffer-local autocommands?  Reduces overhead for autocommands that
    trigger often (inserting a character, switching mode).
        :au Event <buffer> do-something
-   Add events to autocommands:
    Error           - When an error happens
    NormalEnter     - Entering Normal mode
    InsertEnter     - Entering Insert mode
    ReplaceEnter    - Entering Replace mode
    CmdEnter        - Entering Cmdline mode
    VisualEnter     - Entering Visual mode
    *Leave          - Leaving the mode
    SearchPost      - After doing a search command (e.g. to do "M")
    PreDirChanged/PostDirChanged
                    - Before/after ":cd" has been used (for changing the
                      window title)
    BufReadAction   - replaces reading a file
    BufWriteAction  - replaces writing a file
    ShutDown        - when the system is about to shut down
    InsertCharPre   - user typed character Insert mode, before inserting the
                      char.  Set v:char to the character, can be changed. (not
                      when 'paste' is set).
    InsertCharPost  - user typed a character in Insert mode, after inserting
                      the char.
    BufModified     - When a buffer becomes modified, or unmodified (for
                      putting a [+] in the window title or checking out the
                      file from CVS).
    BufFirstChange  - When making a change, when 'modified' is set.  Can be
```

used to do a :preserve for remote files.
BufChange - after a change was made. Set some variables to indicate the position and number of inserted/deleted lines, so that marks can be updated. HierAssist has patch to add BufChangePre, BufChangePost and RevertBuf. (Shah)

- Add autocommand to be executed every so many seconds? For writing the file now and then ('autosave').

'autosave' *'as'* *'noautosave'* *'noas'*
'autosave' 'aw' number (default 0)
Automatically write the current buffer to file N seconds after the last change has been made and when |'modified'| is still set. Default: 0 = do not autosave the buffer.

Insert mode completion/expansion:

9 ^X^L completion doesn't repeat correctly. It uses the first match with the last added line, instead of continuing where the last match ended. (Webb)

8 The code has become too complex. Redesign it, or at least add proper comments.

8 Add option to set different behavior for Insert mode completion:
- ignore/match case
- different characters than 'iskeyword'

8 Add a command to undo the completion, go back to the original text.

8 Use the class information in the tags file to do context-sensitive completion. After "foo." complete all member functions/variables of "foo". Need to search backwards for the class definition of foo. Should work for C++ and Java.
Even more context would be nice: "import java.^N" -> "io", "lang", etc.

7 When expanding $HOME/dir with ^X^F keep the $HOME (with an option?).

7 Add CTRL-X command in Insert mode like CTRL-X CTRL-N, that completes WORDS instead of words.

8 Add CTRL-X CTRL-R: complete words from register contents.

8 Add completion of previously inserted texts (like what CTRL-A does). Requires remembering a number of insertions.

8 Add 'f' flag to 'complete': Expand file names.
Also apply 'complete' to whole line completion.

- Make it possible to search include files in several places. Use the 'path' option? Can this be done with the dictionary completion (use wildcards in the file name)?
- Make CTRL-X CTRL-K do a binary search in the dictionary (if it's sorted).
- Speed up CTRL-X CTRL-K dictionary searching (don't use a regexp?).
- Set a mark at the posititon where the match was found (file mark, could be in another file).
- Add CTRL-A command in CTRL-X mode: show all matches.
- Make CTRL-X CTRL-L use the 'complete' option?
- Add command in CTRL-X mode to add following words to the completed string (e.g. to complete "Pointer->element" with CTRL-X CTRL-P CTRL-W CTRL-W)
- CTRL-X CTRL-F: Use 'path' to find completions.
- CTRL-X CTRL-F: Don't replace "$VIM" with the actual value. (Kelly)

8 Add option 'isexpand', containing characters when doing expansion (so that "." and "\" can be included, without changing 'iskeyword'). (Goldfarb)
Also: 'istagword': characters used for CTRL-].
When 'isexpand' or 'istagword' are empty, use 'iskeyword'.

Command line completion:

8 Change expand_interactively into a flag that is passed as an argument.

8 When completing command names, either sort them on the long name, or list them with the optional part inside [].

- For 'wildmenu': Simplify "../bar" when possible.
- When using <Up> in wildmenu mode for a submenu, should go back to the current menu, not the first one. E.g., ":emenu File.Save<Up>".
8 For ":find" and ":sfind" expand files found in 'path'.
8 When using backtick expansion, the external command may write a greeting message. Add an option or commands to remove lines that match a regexp?
7 When listing matches of files, display the common path separately from the file names, if this makes the listing shorter. (Webb)
- Add command line completion for ":ilist" and friends, show matching identifiers (Webb).
8 Add command line completion for "old value" of a command. ":args <key>" would result in the current list of arguments, which you can then edit.
6 Add command line completion with CTRL-X, just like Insert mode completion. Useful for ":s/word/xx/".
- Add command to go back to the text as it was before completion started. Also to be used for <Up> in the command line.
- Add 'wildlongest' option: Key to use to find longest common match for command line completion (default CTRL-L), like 'wildchar'. (Cregut)
Also: when there are several matches, show them line a CTRL-D.
7 With command line completion after '%' and '#', expand current/alternate file name, so it can be edited.
- Add completion for Environment variables: ":echo $SH<Tab>" -> "$SHELL".

Command line history:
- Add "KeyWasTyped" flag: It's reset before each command and set when a character from the keyboard is consumed. Value is used to decide to put a command line in history or not. Put line in history if it didn't completely resulted from one mapping.
- When using ":browse", also put the resulting edit command in the history, so that it can be repeated. (Demirel)

Command-line editing:
7 Add commands (keys) to delete from the cursor to the end of the command line.
- Add flags to 'whichwrap' for command line editing (cursor right at end of lines wraps to start of line).

Insert mode:
9 When 'autoindent' is set, hitting <CR> twice, while there is text after the cursor, doesn't delete the autoindent in the resulting blank line. (Rich Wales) This is Vi compatible, but it looks like a bug. Rich has a suggestion for a patch to fix this.
8 When using CTRL-O in Insert mode, then executing an insert command "a" or "i", should we return to Insert mode after <Esc>? (Eggink)
Perhaps it can be allowed a single time, to be able to do "<C-O>10axyz<Esc>". Nesting this further is confusing.
":map <F2> 5aabc<Esc>" works only once from Insert mode.
7 Make ":startinsert" command work directly for functions and scripts?
Also make it possible to append (it's difficult at end of line).
- In Insert mode (and command line editing?): Allow undo of the last typed character. This is useful for CTRL-U, CTRL-W, delete and backspace, and also for characters that wrap to the next line.
Also: be able to undo CTRL-R (insert register).
Possibly use 'backspace'="whole" for a mode where at least a <CR> that inserts autoindent is undone by a single <BS>.
- Use CTRL-G in Insert mode for an extra range of commands, like "g" in Normal mode.

-   Make 'paste' work without resetting other options, but override their
    value. Avoids problems when changing files and modelines or autocommands
    are used.
-   When typing CTRL-V and a digit higher than 2, only expect two digits.
-   Insert binary numbers with CTRL-V b.
-   Make it possible to undo <BS>, <C-W> and <C-U>. Bash uses CTRL-Y.

'cindent', 'smartindent':
9   closing } not recognized? (Francois-Xavier Menard)
    (char *tab[] = {"aaa",
        NULL};
        char *name;
9   When using 'cino' "f1s" the indent after a function is set to the indent
    of the "}", should be zero.
9   Formatting comments should use 'comments' setting. E.g. when lining out
    comments like this:
        /*
        ** formatted with comments=sr:/*,mb:**,ex:*/
        */
9   Indent follows "if" instead of "do": (Robert Webb)
        do
        {
                switch (blah)
                {
                        case 1: if (thing)
                                        blah;
                }
        } while (blah);
                                here;
9   One indent to much when doing "o" below this "while": (Rouben Rostamian)
        {
            do
                if (1) {
                    asdf;
                } else
                    qwer;
            while (1);
9   brace not indented correctly:
        case 'c':   if (cond)
                        {
    Should align with "if" in case statement.
7   Allow aligning a closing ")" with the line above, instead of the matching
    "(": (Riehm) if (asdfasdf &&
                        asdf
)
9   "case xx: {", or in general: any { with text before it, a matching }
    should not line up with the {, but with the indent at that position.
    Add setting for K&R paren style? Should be used for this:
        case xx: {
                    some_cmd;
        }
8   "int asdf,<CR>asdf;" should indent the second line more. (Zellner)
8   In C++ it's possible to have {} inside (): (Kirshna)
                func(
                        new String[] {
                            "asdf",
                            "asdf"
                        }
                    );

7 Separate "(0" option into inside/outside a function (Zellner):

```
    func(
       int x)       // indent like "(4"
    {
       if (a
           && b)    // indent like "(0"
```

9 Wrong indent for continuation line with a string:

```
    void foo()
    {
        x = f("s \
            asd\
                tem");
            ff;
    }
```

Also note that the "tem" line has an extra indent. It appears that the brace matching doesn't work, because of the single " in the line.

9 Wrong indent for cino=(4, removing the (void) fixes it: (Zellner)

```
    (void) MyFancyFunction(
                              argument);
```

7 Make indenting more flexible, like syntax highlighting.

\- Make smartindenting configurable. Add 'sioptions', e.g. '#' setting the indent to 0 should be switched on/off.

7 Support ANSI style function header, with each argument on its own line.

\- "[p" and "]p" should use 'cindent' code if it's on (only for the first line).

\- Add option to 'cindent' to set indent for comments outside of {}?

\- Make a command to line up a comment after a code line with a previous comment after a code line. Can 'cindent' do this automatically?

7 Add 'j' flag to 'formatoptions': Remove comment leader when joining lines.

\- formatting of struct/array inits:
The lines used to initialize the struct or array above should not be considered 'continuation' lines just because the previous one does not end with ';'. If you go back line by line I think it should be possible to tell the difference. If you find a line that ends in ';', then it's OK to have continuation lines. If you find a "{", and the character before it (not counting space & comments) is either ',' or '=', then there should not be continuation lines. If the character before the "{" is another "{", then you have to check before that one too.

\- When 'cindent'ing a '}', showmatch is done before fixing the indent. It looks better when the indent is fixed before the showmatch. (Webb)

\- Add option to make indenting work in comments too (for commented-out code), unless the line starts with "*".

\- Don't use 'cindent' when doing formatting with "gq"?

\- When formatting a comment after some text, insert the '*' for the new line (indent is correct if 'cindent' is set, but '*' doesn't get inserted).

\- For smartindent: When typing 'else' line it up with matching 'if'.

\- 'smartindent': allow patterns in 'cinwords', for e.g. TeX files, where lines start with "\item".

\- Support this style of comments (with an option): (Brown)

```
    /* here is a comment that
    is just autoindented, and
    nothing else */
```

\- Add words to 'cinwords' to reduce the indent, e.g., "end" or "fi".

7 Use Tabs for the indent of starting lines, padd with spaces for continuation lines. Allows changing 'tabstop' without messing up the indents.
And/or: Add option to copy indent as-is, without changing spaces to tabs.

Java:

8 Can have {} constructs inside parens. Include changes from Steve Odendahl?
8 Recognize "import java.util.Vector" and use $CLASSPATH to find files for "[i" commands and friends.

'comments':
7 When using "comments=fg:--", Vim inserts three spaces for a new line. When hitting a TAB, these spaces could be removed.
7 The 'n'esting flag doesn't do the indenting of the last (rightmost) item.
6 Make strings in 'comments' option a RE, to be able to match more complicated things. (Phillipps) Use a special flag to indicate that a regexp is used.
8 Make the 'comments' option with "/* * */" lines only repeat the "*" line when there is a "/*" before it? Or include this in 'cindent'?

Text objects:
8 Add test script for text object commands "aw", "iW", etc.
8 Add "gp" and "gP" commands: insert text and make sure there is a single space before it, unless at the start of the line, and after it, unless at the end of the line or befor a ".".
7 Add "g{" and "g}" to move to the first/last character of a paragraph (instead of the line just before/after a paragraph as with "{" and "}").
6 Ignore comment leaders for objects. Make "das" work in reply-email.
5 Make it possible to use syntax group matches as a text object. For example, define a "ccItem" group, then do "da<ccItem>" to delete one. Or, maybe just define "dai", delete-an-item, to delete the syntax item the cursor is on.

Select mode:
7 Alt-leftmouse starts block mode selection in MS Word.
7 Add Cmdline-select mode. Like Select mode, but used on the command line.
- Change gui_send_mouse_event() to pass on mouse events when 'mouse' contains 'C' or 'A'.
- Catch mouse events in ex_getln.c. Also shift-cursor, etc., like in normal_cmd().
- remember start and end of selection in cmdline_info.
- Typing text replaces the selection.

Visual mode:
- When dragging the Visual selection with the mouse and 'scrolloff' is zero, behave like 'scrolloff' is one, so that the text scrolls when the pointer is in the top line.
8 In Visual block mode, "A" appends to lines that don't extend into the block, but padding is wrong (always two spaces). Fix the padding.
9 With blockwise Visual mode, "A" works different than "I" when there are short lines. Make them work the same way. Also make it possible to add the text to short lines too, with padding where needed.
8 What is "R" supposed to do in Visual mode?
8 Make Visual mode local to the buffer. Allow changing to another buffer. When starting a new Visual selection, remove the Visual selection in any other buffer. (Ron Aaron)
7 Support dragging the Visual area to drop it somewhere else. (Aaron)
7 With blockwise Visual mode and "c", "C", "I", "A", etc., allow the use of a <CR>. The entered lines are repeated over the Visual area.
7 CTRL-V :s should substitute only in the block, not to whole lines. (David Young is working on this)

7 Filtering a block should only apply to the block, not to the whole lines.
  When the number of lines is increased, add lines. When decreased, padd with
  spaces or delete? Use ":`<,`>" on the command line.
8 After filtering the Visual area, make "gv" select the filtered text?
  Currently "gv" only selects a single line, not useful.
7 Don't move the cursor when scrolling? Needed when the selection should
  stay the same. Scroll to the cursor at any movement command. With an
  option!
7 In Visual block mode, need to be able to define a corner on a position
  that doesn't have text? Also: when using the mouse, be able to select
  part of a TAB. Even more: Add a mode where the cursor can be on a screen
  position where there is no text. When typing, add spaces to fill the gap.
  Other solution: Always use curswant, so that you can move the cursor to
  the right column, and then use up/down movements to select the line,
  without changing the column.
6 ":left" and ":right" should work in Visual block mode.
7 For Visual mode: Command to do a search for the string in the marked area.
  Only when less than two lines. Use "g/" and "g?".
7 CTRL-I and CTRL-O should work in Visual mode, but only jump to marks in the
  current buffer.
7 CTRL-A and CTRL-X should increase/decrease all numbers in the Visual area.
6 In non-Block mode, "I" should insert the same text in front of each line,
  before the first non-blank, "gI" in column 1.
6 In non-Block mode, "A" should append the same text after each line.
6 ":'<,'>source" should read the selected lines and ":source" them.
6 When in blockwise visual selection (CTRL-V), allow cursor to be placed
  right of the line. Could also allow cursor to be placed anywhere on a TAB
  or other special character.
6 Add commands to move selected text, without deselecting.

Quickfix:
8 Quickfix mode: Column number should be interpreted with an 8-character
  tabstop.

More advanced repeating commands:
- Add "." command for visual mode: redo last visual command (e.g. ":fmt").
- Add "." command after operator: repeat last command of same operator. E.g.
  "c." will repeat last change, also when "x" used since then (Webb).
  "y." will repeat last yank.
  "c2." will repeat the last but one change?
  Also: keep history of Normal mode commands, add command to list the history
  and/or pick an older command.
- History stack for . command? Use "g." command.

Mappings and Abbreviations:
8 To make a mapping work with a prepended "x to select a register, store the
  last _typed_ register name and access it with "&.
8 Add ":amap", like ":amenu".
8 Add ":cab!", abbreviations that only apply to Command-line mode and not to
  entering search strings.
8 Allow mapping of CTRL-@ (anywhere in the LHS).
8 Give a warning when using CTRL-C in the lhs of a mapping. It will never
  (?) work.
8 Add a way to save a current mapping and restore it later. Use a function
  that returns the mapping command to restore it: mapcmd()? mapcheck() is
  not fool proof. How to handle ambigious mappings?
7 Add <0x8f> (hex), <033> (octal) and <123> (decimal) to <> notation?

7 Allow mapping "Q" and "Q}" at the same time. Need to put a flag with "Q", that it needs an extra character before it can match. See Vile 'maplonger' option.
7 When someone tries to unmap with a trailing space, and it fails, try unmapping without the trailing space. Helps for ":unmap xx | unmap yy".
7 Make it possible to map 'wildchar', but only when it's a special character (like CTRL-E). Currently it's only recognized when typed. Useful for mapping a key to do something and then completion.
6 Context-sensitive abbreviations: Specify syntax group(s) in which the abbreviations are to be used.
- Add mappings that take arguments. Could work like the ":s" command. For example, for a mouse escape sequence:
```
    :mapexp  <Esc>{\([0-9]*\),\([0-9]*\);   H\1j\2l
```
- Add the possibility to enter mappings which are used whenever normal text could be enetered. E.g., for "f" command. But not in Normal mode. Sort of opposite of 'langmap'. Use ":tmap" command?
- Make it possible to include a <Nul> in the lhs and rhs of a mapping.
- Add command to repeat a whole mapping ("." only repeats the last change in a mapping). Also: Repeat a whole insert command, including any mappings that it included. Sort-of automatic recording?
- Make it possible to undo all the commands from a mapping, including a trailing unfinished command, e.g. for ":map K iX^[r".
- Add an option to ":map" that makes it display the special keys in <> notation (e.g. <CR> instead of ^M). Or just always do this?
- Include an option (or flag to 'cpoptions') that makes errors in mappings not flush the rest of the mapping (like nvi does).
- Use context sensitiveness of completion to switch abbreviations and mappings off for :unab and :unmap.
6 When using mappings in Insert mode, insert characters for incomplete mappings first, then remove them again when a mapping matches. Avoids that characters that are the start of some mapping are not shown until you hit another character.
- Add optional <Number> argument for mappings:
```
    :map <Number>q           ^W^W<Number>G
    :map <Number>q<Number>t  ^W^W<Number1-1>G<Number2>l
    :map q<Char>            :s/<Char>/\u\0/g
    Or implicit:
    :map q                      <Register>d<Number>$
```
- Add mappings for replace mode: ":rmap". How do we then enter mappings for non-replace Insert mode?
- Add separate mappings for Visual-character/block/line mode?
6 Alias for Normal mode commands, works like :substitute? Would allow mappings with arguments.
- Add 'mapstop' command, to stop recursive mappings.
- List mappings that have a raw escape sequence both with the name of the key for that escape sequence (if there is one) and the sequence itself.
- List mappings: Once with special keys listed as <>, once with meta chars as <M-a>, once with the byte values (octal?). Sort of "spell mapping" command?
- When entering mappings: Add the possibility to enter meta keys like they are displayed, within <>: <M-a>, <~@> or <|a>.
- Allow multiple arguments to :unmap.
- Command to show keys that are not used and available for mapping ":freekeys".
- Allow any character except white space in abbreviations lhs (Riehm).

Incsearch:
- Add a limit to the number of lines that are searched for 'incsearch'?
- When incsearch used and hitting return, no need to search again in many cases, saves a lot of time in big files. (Slootman wants to work on this?)

When not using special characters, can continue search from the last match (or not at all, when there was no match). See oldmail/webb/in.872.
- With incsearch, use CTRL-N/CTRL-P to go to next/previous match, some other key to copy matched word to search pattern (Alexander Schmid).

Searching:
8 Would it be possible to allow ":23,45/pat/flags" to search for "pat" in lines 23 to 45? Or does this conflict with Ex range syntax?
8 Allow identical pairs in 'matchpairs'. Restrict the search to the current line.
7 Allow longer pairs in 'matchpairs'. Use ~/vim/macros/matchit.vim as an example.
8 Make it possible to define the character that "%" checks for in #if/#endif. For nmake it's !if/!endif.
- For "%" command: set hierarchy for which things include other things that should be ignored (like "*/" or "#endif" inside /* */).
Also: use "%" to jump from start to end of syntax region and back.
Alternative: use matchit.vim
8 "/:/e+1" gets stuck on a match at the end of the line. Do we care?
8 A pattern like "\([^a]\+\)\+" takes an awful long time. Recognize that the recursive "\+" is meaningless and optimize for it.
This one is also very slow on "/* some comment */": "^\/*\(.*[^/]\)*$".
7 Recognize "[a-z]", "[0-9]", etc. and replace them with the faster "\l" and "\d".
8 Flags that apply to the whole pattern.
This works for all places where a regexp is used.
Add "\q" to not store this pattern as the last search pattern?
8 Add an option not to use 'hlsearch' highlighting for ":s" and ":g" commands. (Kahn) It would work like ":noh" is used after that command.
Also: An extra flag to do this once, and a flag to keep the existing search pattern.
- Add \%h{group-name}; to search for a specific highlight group.
Add \%s{syntax-group}; to search for a specific syntax group.
- Support Perl regexp. Use PCRE (Perl Compatible RE) package. (Shade)
Or translate the pattern to a Vim one.
Don't switch on with an option for typed commands/mappings/functions, it's too confusing. Use "\@@" in the pattern, to avoid incompatibilities.
7 Add POSIX regexp, like Nvi, with 'extended' option? It's like very-magic.
- Remember flags for backreferenced items, so that "*" can be used after it.
Check with "\(\S\)\1\{3}". (Hemmerling)
- Add flags to search command (also for ":s"?):
i ignore case
I use case
p use Perl regexp syntax (or POSIX?)
v use Vi regexp syntax
f forget pattern, don't keep it for "n" command
F remember pattern, keep it for "n" command
Perl uses these too:
e evaluate the right side as an expression (Perl only)
m multiple line expression (we don't need it)
o compile only once (Perl only)
s single line expression (we don't need it)
x extended regexp (we don't need it)
When used after ":g" command, backslash needed to avoid confusion with the following command.
Add 'searchflags' for default flags (replaces 'gdefault').
- Add command to display the last used substitute pattern and last used pattern. (Margo) Maybe make it accessible through a register (like "/ for search string)?

7 Use T-search algorithm, to speed up searching for strings without special characters. See C't article, August 1997.
- Add 'fuzzycase' option, so that case doesn't matter, and '-' and '_' are equivalent (for Unix filenames).
- Add 'v' flag to search command: enter Visual mode, with the matching text as Visual area. (variation on idea from Bertin)
- Searching: "/this//that/" should find "that" after "this".
- Add global search commands: Instead of wrapping at the end of the buffer, they continue in another buffer. Use flag after search pattern:
  a for the next file in the argument list
  f for file in the buffer list
  w for file edited in a window.
  e.g. "/pat/f". Then "n" and "N" work through files too. "f" flag also for ":s/pat/foo/f"??? Then when 'autowrite' and 'hidden' are both not set, ask before saving files: "Save modified buffer "/path/file"? (Yes/Hide/No Save-all/hide-All/Quit) ".
- ":s/pat/foo/3": find 3rd match of "pat", like sed. (Thomas Koehler)
- Special characters in patterns:
  Inside []:
  \012 octal character
  \0x1a hex character
  \0<BS> \0<Esc>: special character
7 When searching with 'n' give message when getting back where the search first started. Remember start of search in '/ mark.
7 Add option that scrolls screen to put cursor in middle of screen after search always/when off-screen/never. And after a ":tag" command. Maybe specify how many lines below the screen causes a redraw with the cursor in the middle (default would be half a screen, zero means always).
6 Support multiple search buffers, so macros can be made without side effects.
7 From xvim: Allow a newline in search patterns (also for :s, can delete newline). Add BOW, EOW, NEWL, NLORANY, NLBUTANY, magic 'n' and 'r', etc. [not in xvim:] Add option to switch on matches crossing ONE line boundary.
7 Add ":iselect", a combination of ":ilist" and ":tselect". (Aaron)
  Also ":dselect".

Undo:
8 Make undo more memory-efficient: Compare text before and after change, only remember the lines that really changed.
7 Add an undo tree: When making a change, instead of clearing any future undo (thus redo) info, make a new branch. How to navigate through the undo tree?
- For u_save() include the column number. This can be used to set '[and ']. And in the future the undo can be made more efficient (Webb).
- In out-of-memory situations: Free allocated space in undo, and reduce the number of undo levels (with confirmation).
- Instead of [+], give the number of changes since the last write: [+123]. When undoing to before the last write, change this to a negative number: [-99].
7 Make it possible to jump to the location of the last change. Like doing "u" "CTRL-R". (Lange) Keep a list of the 10 last changes or so. Use g_CTRL-O and g_CTRL-I?
- With undo with simple line delete/insert: optimize screen updating.
- When executing macro's: Save each line for undo only once.
- Store undo info in a file that survives until the next edit. Then it's possible to undo to before the current editing session. Combined with viminfo?
- When doing a global substitute, causing almost all lines to be changed, undo info becomes very big. Put undo info in swap file??

Buffer list:
7 Command to execute a command in another buffer: ":inbuf {bufname} {cmd}".
  Also for other windows: ":inwin {winnr} {cmd}". How to make sure that
  this works properly for all commands, and still be able to return to the
  current buffer/window? E.g.: ":inbuf xxx only".
8 Add File.{recent_files} menu entries: Recently edited files.
8 Unix: Check all uses of fnamecmp() and fnamencmp() if they should check
  inode too.
7 Add another number for a buffer, which is visible for the user. When
  creating a new buffer, use the lowest number not in use. (or the highest
  number in use plus one?)
7 Offer some buffer selection from the command line? Like using ":ls" and
  asking for a buffer number. (Zachmann)
- When starting to edit a file that is already in the buffer list, use the
  file name argument for the new short file name. (Webb)
- Add an option to make ":bnext" and ":bprev" wrap around the end of the
  buffer list. Also for ":next" and ":prev"?
7 Add argument to ":ls" which is a pattern for buffers to list.
  E.g. ":ls *.c". (Thompson)
7 Add expansion of buffer names, so that "*.c" is expanded to all buffer
  names. Needed for ":bdel *.c", ":bunload *.c", etc.
8 Support for <afile> where a buffer name is expected.
8 Some commands don't use line numbers, but buffer numbers. '$'
  should then mean the number of the last buffer. E.g.: "4,$bdel".
7 Add an option to mostly use slashes in file names. Separately for
  internal use and for when executing an external program?

Swap (.swp) files:
8 If writing to the swap file fails, should try to open one in another
  directory from 'dir'. Useful in case the file system is full and when
  there are short file name problems.
8 Also use the code to try using a short file name for the backup and swap
  file for the Win32 and Dos 32 bit versions.
8 When the edited file is a symlink, try to put the swap file in the same
  dir as the actual file. Adjust FullName(). Avoids editing the same file
  twice (e.g. when using quickfix). Also try to make the name of the backup
  file the same as the actual file?
7 When using 64 bit inode numbers, also store the top 32 bits. Add another
  field for this, using part of bo_fname[], to keep it compatible.
7 When editing a file on removable media, should put swap file somewhere
  else. Use something like 'r' flag in 'viminfo'. 'diravoid'?
  Also: Be able to specify minimum disk space, skip directory when not
  enough room.
7 Add a configure check for which directory should be used: /tmp, /var/tmp
  or /var/preserve.
- Add an option to create a swap file only when making the first change to
  the buffer. (Liang) Or only when the buffer is not read-only.
- Add option to set "umask" for backup files and swap files (Antwerpen).
  'backupumask' and 'swapumask'? Or 'umaskback' and 'umaskswap'?
- When editing a readonly file, don't use a swap file but read parts from the
  original file. Also do this when the file is huge (>'maxmem'). We do
  need to load the file once to count the number of lines? Perhaps keep a
  cached list of which line is where.

Viminfo:
7 Can probably remove the code that checks for a writable viminfo file,

    because we now do the chown() for root, and others can't overwrite someone
    else's viminfo file.
8   Add argument to keep the list of buffers when Vim is started with a file
    name. (Schild)
8   Keep the last used directory of the file browser (File/Open menu).
8   Remember a list of last accessed files. To be used in the
    "File.Open Recent" menu. Default is to remember 10 files or so.
    Also remember which files have been read and written. How to display
    this?
7   Also store the ". register (last inserted text).
7   Make it possible to store buffer names in viminfo file relative to some
    directory, to make them portable over a network. (Aaron)
6   Store a snapshot of the currently opened windows. So that when quitting
    Vim, and then starting again (without a file name argument), you see the
    same files in the windows. Use ":mksession" code?
-   Make marks present in .viminfo usable as file marks: Display a list of
    "last visited files" and select one to jump to.

Modelines:
8   Before trying to execute a modeline, check that it looks like one (valid
    option names). If it's very wrong, silently ignore it.
    Ignore a line that starts with "Subject: ".
-   When an option value is coming from a modeline, do not carry it over to
    another edited file? Would need to remember the value from before the
    modeline setting.
-   Allow setting a variable from a modeline? Only allow using fixed strings,
    no function calls, to avoid a security problem.
-   Allow ":doauto BufRead x.cpp" in modelines, to execute autocommands for
    .cpp files.
-   Support the "abbreviate" command in modelines (Kearns). Careful for
    characters after <Esc>, that is a security leak.
-   Add option setting to ask user if he wants to have the modelines executed
    or not. Same for .exrc in local dir.

Options:
8   Make ":mksession" store buffer-specific options for the specific buffer.
8   With ":mksession" always store the 'sessionoptions' option, even when
    "options" isn't in it. (St-Amant)
8   When using ":mksession", also store a command to reset all options to
    their default value, before setting the options that are not at their
    default value.
8   Should ":mksession" restore the current directory when writing the
    session, or the directory where the session file is? Probably need a word
    in 'sessionoptions' to make a choice:
    "curdir" (cd to current directory when session file was generated)
    "sessiondir" (cd to directory of session file)
    "nodir" (don't cd at all)
8   Add Edit.Settings menu with most often used options:
        line numbering on/off
        wrap on/off
        hlsearch on/off
        toolbar on/off
        scrollbars left, right, bottom on/off
        shiftwidth 2, 3, 4, 5, 6, 8
        expandtab on/off
8   Make "old" number options that really give a number of effects into string
    options that are a comma separated list. The old number values should
    also be supported.

8 Add commands to save and restore an option, which also preserves the flag
that marks if the option was set. Useful to keep the effect of setting
'compatible' after ":syntax on" has been used.
7 There is 'titleold', why is there no 'iconold'? (Chazelas)

External commands:
9 When filtering a buffer (e.g., to gunzip it) marks are changed when they
shouldn't. Add an option to switch of adjusting marks. Use it for .gz
files gunzipping and gzipping.
8 When filtering text, redirect stderr so that it can't mess up the screen
and Vim doesn't need to redraw it. Also for ":r !cmd".
4 Set separate shell for ":sh", piping "range!filter", reading text "r !ls"
and writing text "w !wc". (Deutsche) Allow arguments for fast start (e.g.
-f).
4 Allow direct execution, without using a shell.
4 Run an external command in the background. But how about I/O in the GUI?
Careful: don't turn Vim into a shell!
4 Add feature to disable using a shell or external commands.

Multiple Windows:
6 Add an option to resize the shell when splitting and/or closing a window.
":vsp" would make the shell wider by as many columns as needed for the new
window. Specify a maximum size (or use the screen size). ":close" would
shrink the shell by as many columns as come available. (Demirel)
7 When starting Vim several times, instanciate a Vim server, that allows
communication between the different Vims. Feels like one Vim running with
multiple top-level windows. Esp. useful when Vim is started from an IDE
too. Requires some form of inter process communication.
7 Run Vim as a server, and let other started Vims send an edit command to
the server. Consider patch from Brent Verner to add sockets interface?
~/vim/patches/verner_socket.tgz
- Support a connection to an external viewer. Could call the viewer
automatically after some seconds of non-activity, or with a command.
Allow some way of reporting scrolling and cursor positioning in the viewer
to Vim, so that the link between the viewed and edited text can be made.

Marks:
8 Add a command to jump to a mark and make the motion inclusive. g'm and g`m?
8 When deleting lines, don't delete uppercase marks in them. Move the mark
to the nearest line.
8 The '" mark is set to the first line, even when doing ":next" a few times.
Only set the '" mark when the cursor was really moved in a file.
8 Make `` and '', which would position the new cursor position in the middle
of the window, restore the old topline (or relative position) from when
the mark was set.
7 Make a list of file marks in a separate window. For listing all buffers,
matching tags, errors, etc. Normal commands to move around. Add commands
to jump to the mark (in current window or new window). Start it with
":browse cmd"?
6 Add a menu that lists the Marks like ":marks". (Amerige)
7 For ":jumps", ":tags" and ":marks", for not loaded buffers, remember the
text at the mark. Highlight the column with the mark.
7 Highlight each mark in some way (With "Mark" highlight group).
Or display marks in a separate column, like 'number' does.
7 Use d"m to delete rectangular area from cursor to mark m (like Vile's \m
command).
7 Try to keep marks in the same position when:

    - replacing with a line break, like in ":s/pat/^M/", move marks after the
      line break column to the next line. (Acevedo)
    - filtering lines, try to keep the marks in the filtered lines. Need to
      move some marks if the number of lines decreases.
    - changing text, e.g. when formatting with "gq".
    - inserting/deleting characters in a line.
5   Include marks for start/end of the current word and section. Useful in
    mappings.

Digraphs:
-   Make it possible to enter "r<C-E>" and "r<C-Y>" (get character from line
    below/above).
-   Use digraph table to tell Vim about the collating sequence of special
    characters?
-   Add command to remove (all) digraphs. (Brown)
7   Support different sets of digraphs (depending on the character set?). At
    least Latin1/Unicode, Latin-2, MS-DOS (esp. for Win32).

Writing files:
-   In vim_rename(), should lock "from" file when deleting "to" file for
    systems other than Amiga. Avoids problems with unexpected longname to
    shortname conversion.
8   write mch_isdevice() for Amiga, Mac, VMS, etc.
8   When appending to a file, Vim should also make a backup and a 'patchmode'
    file.
6   Add an option to write a new, numbered, backup file each time. Like
    'patchmode', e.g., 'backupmode'.
6   Make it possible to write 'patchmode' files to a different directory.
    E.g., ":set patchmode=~/backups/*.orig". (Thomas)
6   Add an option to prepend something to the backup file name. E.g., "#".
    Or maybe allow a function to modify the backup file name?
8   Only make a backup when overwriting a file for the first time. Avoids
    loosing the original when writing twice. (Slootman)
7   On non-Unix machines, also overwrite the original file in some situations
    (file system full, it's a link on an NFS partition).
7   When editing a file, check that it has been change outside of Vim more
    often, not only when writing over it. E.g., at the time the swap file is
    flushed. Or every ten seconds or so (use the time of day, check it before
    waiting for a character to be typed).
8   When a file was changed since editing started, show this in the status
    line of the window, like "[time]".
    Make it easier to reload all outdated files that don't have changes.
    Automatic and/or with a command.

Substitute:
-   :s///p prints the line after a substitution.
-   With :s///c replace \&, ~, etc. when showing the replacement pattern.
8   With :s///c allow scrolling horizontally when 'nowrap' is effective.
    Also allow a count before the scrolling keys.
-   Add number option to ":s//2": replace second occurrence of string? Or:
    :s///N substitutes N times.
-   Add answers to ":substitute" with 'c' flag, used in a ":global", e.g.:
    ":g/pat1/s/pat2/pat3/cg": 'A' do all remaining replacements, 'Q' don't do
    any replacements.
7   Substitute in a block of text. Use {line}.{column} notation in an Ex
    range, e.g.: ":1.3,$.5s" means to substitute from line 1 column 3 to the
    last line column 5.

Mouse support:
8   Add 'mouse' flag, which sets a behavior like Visual mode, but automatic
    yanking at the button-up event. Or like Select mode, but typing gets you
    out of Select mode, instead of replacing the text. (Bhaskar)
7   Checkout sysmouse() for FreeBSD console mouse support.
-   Implement mouse support for the Amiga console.
-   Using right mouse button to extend a blockwise selection should attach to
    the nearest corner of the rectangle (four possible corners).
-   Precede mouse click by a number to simulate double clicks?!?
-   When mouse click after 'r' command, get character that was pointed to.

Crypt:
8   Also crypt the swapfile, each block separately. Change mf_write() and
    mf_read(). How to get b_p_key to these functions?

Argument list:
6   Add command to put all filenames from the tag files in the argument list.
    When given an argument, only use the files where that argument matches
    (like `grep -l ident`) and jump to the first match.
6   Add command to form an args list from all the buffers?

Registers:
8   Don't display empty registers with ":display". (Etienne)
-   When appending to a register, also report the total resulting number of
    lines. Or just say "99 more lines yanked", add the "more".
-   When inserting a register in Insert mode with CTRL-R, don't insert comment
    leader when line wraps?
-   The ":@r" commands should take a range and execute the register for each
    line in the range.
-   Add "P" command to insert contents of unnamed register, move selected text
        to position of previous deleted (to swap foo and bar in " + foo")
8   Make "/ register writable. (Maxi)
    How to take care of the flags (offset, magic)?
7   Add ! register, for shell commands. (patch from Grenie)

Various improvements:
8   Allow using "**" as a wildcard in commands like ":next" and ":args".
7   Add a message area for the user. Set some option to reserve space (above
    the command line?). Use an ":echouser" command to display the message
    (truncated to fit in the space).
7   Add %s to 'keywordprg': replace with word under the cursor. (Zellner)
8   Support printing on Unix. Can use "lpansi.c" as an example. (Bookout)
8   Add put command that replaces the text under it. Esp. for blockwise
    Visual mode.
7   Enhance termreponse stuff: Add t_CV(?): pattern of term response, use
    regexp: "\e\[[>?][0-9;]*c", but only check just after sending t_RV.
7   Add "g|" command: move to N'th column from the left margin (after wrapping
    and applying 'leftcol'). Works as "|" like what "g0" is to "0".
7   Add patch from Wall for this one (~/Mail/oldmail/wall/in.00019):
    'flipcase' variable: upper/lower case pairs.
    Insert comma's between pairs and allow a range, make it look like
    'isfname'. E.g. ":set flipcase=a-zA-Z,xX,23-33:143-153". The colon to
    separate the from and to part is optional.
7   Support setting 'euqalprg' to a user function name.

8 Add cursor-column highlighting. Enable it with 'cursorcolumn' option, set highlighting with "CursorColumn" group. Useful for aligning text. Also cursor-row highlighting.
7 Highlight the characters after the end-of-line differently.
7 When 'whichwrap' contains "l", "$dl" should join lines?
6 Include the ruby interface?
7 Open a server socket (with an option for IP/port nr) to allow remote control. Optionally with a password to restrict access.
Use the Simple Sockets Library? (Campbell)
Same sample code (brent verner): http://linux1.org/misc/vim_socket.tar.bz2
8 Include a connection to an external program through a pipe? See patches from Felbinger for a mathematica inteface.
Or use emacs server kind of thing?
8 Add an argument to configure to use $CFLAGS and not modify it? (Mooney)
8 Enabling features is a mix of configure arguments and defines in feature.h. How to make this consistent? Feature.h is required for non-unix systems. Perhaps let configure define CONF_XXX, and use #ifdef CONF_XXX in feature.h? Then what should min-features and max-features do?
8 Add "g^E" and "g^Y", to scroll a screen-full line up and down.
6 Add ":timer" command, to set a command to be executed at a certain interval, or once after some time has elapsed. (Aaron)
8 Add ":confirm" handling in open_exfile(), for when file already exists.
8 Use confirm/dialog stuff to ask the user, when a file has changed outside of Vim, if he wants to reload it. Triggered when focus gained, after shell command, when entering another buffer, etc..
Also do this when editing a new file, and another application creates the file before doing ":w" in Vim.
Also check if the file protection has changed. When checking a file into RCS it is made read-only, when checking out it is made read-write.
8 When quitting with changed files, make the dialog list the changed file and allow "write all", "discard all", "write some". The last one would then ask "write" or "discard" for each changed file. Patch in HierAssist does something like this. (Shah)
7 Use growarray for replace stack.
7 Have a look at viH (Hellenic or Greek version of Vim). But a solution outside of Vim might be satisfactory (Haritsis).
3 Make "2d%" work like "d%d%" instead of "d2%"?
8 Make "more" prompt accept command characters, like "hit-enter" prompt? Or extend it with more commands, like "less": 'b' for back, 'j' for one line down, etc.
8 For the "--more--" prompt, support the 'b'ack command for more commands.
7 "g CTRL-O" jumps back to last used buffer. Skip CTRL-O jumps in the same buffer. Make jumplist remember the last ten accessed buffers?
- Keep a list of most recently used files for each window, use "[o" to go back (older file) and "]n" to go forward (newer file) (like ^O and ^I for jumps) (Webb). Use ":files" and ":ls" to list the files in history order.
7 Add a history of recently accessed buffer. Maybe make "2 CTRL-^" jump to the 2nd previously visited buffer, "3 CTRL-^" to the third, etc. Or use "3 g CTRL-^" for this?
5 Add an option to set the width of the 'number' column. Eight positions is often more than needed. Or adjust the width to the length of the file?
- Add code to disable the CAPS key when going from Insert to Normal mode.
- Set date/protection/etc. of the patchfile the same as the original file.
- Use growarray for termcodes[] in term.c
- Add <window-99>, like <cword> but use filename of 99'th window.
- Make a set of operations on list of names: expand wildcards, replace home dir, append a string, delete a string, etc.
- Remove mktemp() and use tmpname() only? Ctags does this.
- When replacing environment variables, and there is one that is not set, turn it into an empty string? Only when expanding options? (Hiebert)

- Option to set command to be executed instead of producing a beep (e.g. to call "play newbeep.au").
- Add option to show the current function name in the status line. More or less what you find with "[[k", like how 'cindent' recognizes a function. (Bhatt).
- "[r" and "]r": like "p" and "P", but replace instead of insert (esp. for blockwise registers).
- Add 'timecheck' option, on by default. Makes it possible to switch off the timestamp warning and question. (Dodt).
- Add an option to set the time after which Vim should check the timestamps of the files. Only check when an event occurs (e.g., character typed, mouse moved). Useful for non-GUI versions where keyboard focus isn't noticable.

9 When using ":w <fname>" it's possible that this file is loaded in another buffer. Give a warning right away, don't wait for a shell command.

- Make 'smartcase' work even though 'ic' isn't set (Webb).

7 When formatting text, allow to break the line at a number of characters. Use an option for this: 'breakchars'? Useful for formatting Fortran code.

- Add flag to 'formatoptions' to be able to format book-style paragraphs (first line of paragraph has larger indent, no empty lines between paragraphs). Complements the '2' flag. Use '>' flag when larger indent starts a new paragraph, use '<' flag when smaller indent starts a new paragraph. Both start a new paragraph on any indent change.

8 Allow using a trailing space to signal a paragraph that continues on the next line. Can be used for continuous formatting. Could use 'autoformat' option, which specifies a regexp which triggers auto-formatting (for one line). ":set autoformat=\\s$".

- Be able to redefine where a sentence stops. Use a regexp pattern?
- Be able to redefine where a paragraph starts. For "[[" where the '{' is not in column 1.

8 ":cd": echo the new current directory.

6 Add ":cdprev": go back to the previous directory. Need to remember a stack of previous directories. We also need ":cdnext".

7 Should ":cd" for MS-DOS go to $HOME, when it's defined?

- Make "gq<CR>" work on the last line in the file. Maybe for every operator?

8 findmatchlimit() should be able to skip comments. Solves problem of matching the '{' in /* if (foo) { */ (Fiveash)

- findmatch() should be adjusted for Lisp. See remark at get_lisp_indent().
- Add more redirecting of Ex commands:

```
    :redir @> register  (append)
    :redir #  bufname
    :redir #> bufname   (append)
    :redir =  variable
    :redir => variable  (append)
```

8 ":redir >$HOME/xxx" should work: expand file name argument of ":redir".

- Setting of options, specifically for a buffer or window, with ":set window.option" or ":set buffer.option=val". Or use ":buffer.set". Also: "buffer.map <F1> quit".
- Add :delcr command:

```
                        *:delcr*
 :[range]delcr[!]   Check [range] lines (default: whole buffer) for lines
                    ending in <CR>.  If all lines end in <CR>, or [!] is
                    used, remove the <CR> at the end of lines in [range].
                    A CTRL-Z at the end of the file is removed.  If
                    [range] is omitted, or it is the whole file, and all
                    lines end in <CR> 'textmode' is set.  {not in Vi}
```

- Should integrate addstar() and file_pat_to_reg_pat().
- When working over a serial line with 7 bit characters, remove meta characters from 'isprint'.
- Use fchdir() in init_homedir(), like in FullName().

- In win_update(), when the GUI is active, always use the scrolling area. Avoid that the last status line is deleted and needs to be redrawn.
- That "cTx" fails when the cursor is just after 'x' is Vi compatible, but may not be what you expect. Add a flag in 'cpoptions' for this? More general: Add an option to allow "c" to work with a null motion.
- Give better error messages by using errno (strerror()).
- Give "Usage:" error message when command used with wrong arguments (like Nvi).
- Make 'restorescreen' option also work for xterm (and others), replaces the SAVE_XTERM_SCREEN define.
7 Support for ":winpos" In xterm: report the current window position.
- Give warning message when using ":set t_xx=asdf" for a termcap code that Vim doesn't know about. Add flag in 'shortmess'?
6 Add ":che <file>", list all the include paths which lead to this file.
- For a commandline that has several commands (:s, :d, etc.) summarize the changes all together instead of for each command (e.g. for the rot13 macro).
- Add command like "[I" that also shows the tree of included files.
- Add command like ":ts" that shows the output of "[I" and asks for a match to jump to. (Zellner)
- ":set sm^L" results in ":set s", because short names of options are also expanded. Is there a better way to do this?
- Add ":@!" command, to ":@" like what ":source!" is to ":source".
8 Add ":@:!": repeat last command with forceit set.
- Should be possible to write to a device, e.g. ":w! /dev/null".
- Add 't_normal': Used whenever t_me, t_se, t_ue or t_Zr is empty.
- ":cab map test ^V| je", ":cunab map" doesn't work. This is vi compatible!
- CTRL-W CTRL-E and CTRL-W CTRL-Y should move the current window up or down if it is not the first or last window.
- Include-file-search commands should look in the loaded buffer of a file (if there is one) instead of the file itself.
7 Change 'nrformats' to include the leader for each format. Example:
        nrformats=hex:$,binary:b,octal:0
  Add setting of 'nrformats' to syntax files.
- 'path' can become very long, don't use NameBuff for expansion.
- When unhiding a hidden buffer, put the same line at top of the window as the one before hiding it. Or: keep the same relative cursor position (so many percent down the windows).
- Make it possible for the 'showbreak' to be displayed at the end of the line. Use a comma to separate the part at the end and the start of the line? Highlight the linebreak characters, add flag in 'highlight'.
- Some string options should be expanded if they have wildcards, e.g. 'dictionary' when it is "*.h".
- Use a specific type for number and boolean options, making it possible to change it for specific machines (e.g. when a long is 64 bit).
- Add option for <Insert> in replace mode going to normal mode. (Nugent)
- Add a next/previous possibility to "[^I" and friends.
- Add possibility to change the HOME directory. Use the directory from the passwd file? (Antwerpen)
- When doing "[^I" or "[^D" add position to tag stack.
- Add command to put current position to tag stack: ":tpush".
8 Add commands to push and pop all or individual options. ":setpush tw", ":setpop tw", ":setpush all". Maybe pushing/popping all options is sufficient. ":setflush" resets the option stack?
  How to handle an aborted mapping? Remember position in tag stack when mapping starts, restore it when an error aborts the mapping?
- Use a builtin grep command for ":grep"? Makes it possible to add the column number.
- Change ":fixdel" into option 'fixdel', t_del will be adjusted each time t_bs is set? (Webb)

- "gc": goto character, move absolute character positions forward, also counting newlines. "gC" goes backwards (Weigert).
- When doing CTRL-^, redraw buffer with the same topline (Demirel). Store cursor row and window height to redraw cursor at same percentage of window (Webb).
- Besides remembering the last used line number of a file, also remember the column. Use it with CTRL-^ et. al.
- When a window resizes, the line with the cursor should stay at the same percentage from the start of the window as it was before.
- Check for non-digits when setting a number option (careful when entering hex codes like 0xff).
- Add option to make "." redo the "@r" command, instead of the last command executed by it. Also to make "." redo the whole mapping. Basically: redo the last TYPED command.
- Support URL links for ^X^F in Insert mode, like for "gf".
- Add 'wwwpath', used like 'path' for when "gf" used on an URL?
- Support %name% expansion for "gf" on Windows.
- When finding an URL or file name, and it doesn't exist, try removing a trailing '.'.
- Add ":path" command modifier. Should work for every command that takes a file name argument, to search for the file name in 'path'. Use find_file_in_path().
- Highlight control characters on the screen: Shows the difference between CTRL-X and "^" followed by "X" (Colon).
- Integrate parsing of cmdline command and parsing for expansion.
- Create a program that can translate a .swp file from any machine into a form usable by Vim on the current machine.
- Add ":noro" command: Reset 'ro' flag for all buffers, except ones that have a readonly file. ":noro!" will reset all 'ro' flags.
- Add a variant of CTRL-V that stops interpretation of more than one character. For entering mappings on the command line where a key contains several special characters, e.g. a trailing newline.
- Add regex for 'paragraphs' and 'sections': 'parare' and 'sectre'. Combine the two into a regex for searching. (Ned Konz)
- Make '2' option in 'formatoptions' also work inside comments.
- Add 's' flag to 'formatoptions': Do not break when inside a string. (Dodt)
- Add flag to 'formatoptions' to recognize the change of indent as the start of a new paragraph (for paragraphs without separating empty line, but extra indent for the new paragraph) (Leitner).
- When window size changed (with the mouse) and made too small, set it back to the minimal size.
- Add "]>" and "[<", shift comment at end of line (command; /* comment */).
- Should not call cursorcmd() for each vgetc() in getcmdline().
- ":split file1 file2" adds two more windows (Webb).
- Don't give message "Incomplete last line" when editing binary file.
- Add ":a", ":i" for preloading of named buffers.
- Allow autowrite when doing ":e file" (with an option 'eaw').
- Allow a "+command" argument before each file name in the Vim command line: "vim +123 file1 +234 file2 +345 file3". ???
- When entering text, keep other windows on same buffer updated (when a line entered)?
- Check out how screen does output optimizing. Apparently this is possible as an output filter.
- In dosub() regexec is called twice for the same line. Try to avoid this.
- Window updating from memline.c: insert/delete/replace line.
- Optimize ml_append() for speed, esp. for reading a file.
- V..c should keep indent when 'ai' is set, just like [count]cc.
- Updatescript() can be done faster with a string instead of a char.
- Screen updating is inefficient with CTRL-F and CTRL-B when there are long lines.

- Uppercase characters in ex commands can be made lowercase?
8 Add option to show characters in text not as "|A" but as decimal ("^129"), hex ("\x81") or octal ("\201") or meta (M-x). Nvi has the 'octal' option to switch from hex to octal. Vile can show unprintable characters in hex or in octal.
7 Tighter integration with xxd to edit binary files. Make it more easy/obvious to use. Command line argument?
- How does vi detect whether a filter has messed up the screen? Check source. After ":w !command" a wait_return?
- Improve screen updating code for doput() (use s_ins()).
- With 'p' command on last line: scroll screen up (also for terminals without insert line command).
- Use insert/delete char when terminal supports it.
- Optimize screen redraw for slow terminals.
- Optimize "dw" for long row of spaces (say, 30000).
- Add "-d null" for editing from a script file without displaying.
- In Insert mode: Remember the characters that were removed with backspace and re-insert them one at a time with <key1>, all together with <key2>.
- Amiga: Add possibility to set a keymap. The code in amiga.c does not work yet.
- Implement 'redraw' option.
- Add special code to 'sections' option to define something else but '{' or '}' as the start of a section (e.g. one shiftwidth to the right).
- Add 'indent' option: Always use this amount of indent when starting a new line and when formatting text.
- Use pipes for filtering on Unix. Requires using fork() to be able to read and write at the same time, or some select() mechanism.
7 Allow using Vim in a pipe: "ls | vim -u xxx.vim - | yyy". Only needs implementing ":w" to stdout in the buffer that was read from stdin.
8 Allow opening an unnamed buffer with ":e !cmd" and ":sp !cmd". Vile can do it.
- Allow for +command and -option on any position in argv[].
- Add commands like]] and [[that do not include the line jumped to.
- When :unab without matching "from" part and several matching "to" parts, delete the entry that was used last, instead of the first in the list.
- Add text justification option.
- Set boolean options on/off with ":set paste=off", ":set paste=on".
- After "inv"ing an option show the value: ":set invpaste" gives "paste is off".
- Check handling of CTRL-V and '\' for ":" commands that do not have TRLBAR.
- When a file cannot be opened but does exist, give error message.
- Amiga: When 'r' protection bit is not set, file can still be opened but gives read errors. Check protection before opening.
- When writing check for file exists but no permission, "Permission denied".
- If file does not exists, check if directory exists.
- MSDOS: although t_cv and t_ci are not set, do invert char under cursor.
- Settings edit mode: make file with ":set opt=xx", edit it, parse it as ex commands.
- ":set -w all": list one option per line.
- Amiga: test for 'w' flag when reading a file.
- :table command (Webb)
- Add new operator: clear, make area white (replace with spaces): "g ".
- Make it possible for a user to define a new operator. Implementation with internal scripting language or Perl?
- Add command to ":read" a file at a certain column (blockwise read?).
- Add 'resizecmd' option: vi command to be executed when window is resized.
- Add sort of replace mode where case is taken from the old text (Goldfarb).
- Allow multiple arguments for ":read", read all the files.
- Support for tabs in specific columns: ":set tabcol=8,20,34,56" (Demirel).
- Add 'searchdir' option: Directories to search for file name being edited

    (Demirel).
-   Modifier for the put command: Change to linewise, charwise, blockwise, etc.
-   Add commands for saving and restoring options ":set save" "set restore",
    for use in macro's and the like.
-   Keep output from listings in a window, so you can have a look at it while
    working in another window. Put cmdline in a separate window?
-   Add possibility to put output of ex commands in a buffer or file, e.g. for
    ":set all". ":r :set all"?
-   'edit' option: When off changing the buffer is not possible (Really
    read-only mode).
-   When the 'equalalways' option is set, creating a new window should not
    result in windows to become bigger. Deleting a window should not result in
    a window to become smaller (Webb).
-   When resizing the whole Vim window, the windows inside should be resized
    proportionally (Webb).
-   Include options directly in option table, no indirect pointers. Use
    mkopttab to make option table?
-   When doing ":w dir", where "dir" is a directory name, write the current
    file into that directory, with the current file name (without the path)?
-   Support for 'dictionary's that are sorted, makes access a lot faster
    (Haritsis).
-   Add "^Vrx" on the command line, replace with contents of register x. Used
    instead of CTRL-R to make repeating possible. (Marinichev)
-   Add "^Vb" on the command line, replace with word before or under the
    cursor?
-   Option to make a .swp file only when a change is made (Templeton).
-   Support mapping for replace mode and "r" command (Vi doesn't do this)?
5   Add 'ignorefilecase' option: Ignore case when expanding file names.
    ":e ma<Tab>" would also find "Makefile" on Unix.

From Elvis:
-   Use "instman.sh" to install manpages?
-   Add ":alias" command.
-   fontchanges recognized "\\fB" etc.
-   Search patterns:
      \@   match word under cursor.
    but do:
      \@w   match the word under the cursor?
      \@W   match the WORD under the cursor?
8   ":window" command:
    :win +   next window (up)
    :win ++   idem, wrapping
    :win -   previous window (down)
    :win --   idem, wrapping
    :win nr   to window number "nr"
    :win name   to window editing buffer "name"
7   ":cc" compiles a single file (default: current one). 'ccprg'   option is
    program to use with ":cc". Use ":compile" instead of ":cc"?

From Nvi:
-   'searchincr' option, alias for 'incsearch'?
-   'leftright' option, alias for 'nowrap'?
-   Have a look at "vi/doc/vi.chart", for Nvi specialities.
8   Add 'keytime', time in 1/10 sec for mapping timeout?
-   Add 'filec' option as an alternative for 'wildchar'.
6   Support Nvi command names as an alias:
    :bg   :hide
    :fg fname   :buf fname (with 'hidden' set?)

```
    :dis b          :ls
    :Edit fname     :split fname
    :Fg fname       :sbuf fname (with 'hidden' set?)
    :Next           :snext (can't do this, already use :Next)
    :Previous       :sprevious
    :Tag            :stag
```

From xvi:
- CTRL-_ : swap 8th bit of character.
- Add egrep-like regex type, like xvi (Ned Konz) or Perl (Emmanuel Mogenet)

From vile:
- When horizontal scrolling, use '<' and '>' for lines continuing outside of window.
- Support putting .swp files in /tmp: Command in rc.local to move .swp files from /tmp to some directory before deleting files.

Far future and "big" extensions:
- Make it possible to run Vim inside a window of another program. For Xwindows this can be done with XReparentWindow().
- Add a way of scrolling that leaves the cursor where it is. Especially when using the scrollbar. Typing a cursor-movement command scrolls back to where the cursor is.
- Make it easy to setup Vim for groups of users: novice vi users, novice Vim users, C programmers, xterm users, GUI users,...
- Add a mode to only view text. Like Less, but with syntax highlighting etc.
- Change layout of blocks in swap file: Text at the start, with '\n' in between lines (just load the file without changes, except for Mac). Indexes for lines are from the end of the block backwards. It's the current layout mirrored.
- Make it possible to edit a register, in a window, like a buffer.
- Add stuff to syntax highlighting to change the text (upper-case keywords, set indent, define other highlighting, etc.).
- Mode to keep C-code formatted all the time (sort of on-line indent).
- Several top-level windows in one Vim session. Be able to use a different font in each top-level window.
- Allow editing beyond end of line, just like there are all spaces. Switch this on with an option or special Insert mode command. Also allow editing above start and below end of buffer.
- Smart cut/paste: recognize words and adjust spaces before/after them.
- Mode to keep text formatted while inserting/deleting. Use soft/hard returns with an option to switch this off. (CR-LF can be used for Unix as a soft return).
- Add column numbers to ":" commands ":line1,line2[col1,col2]cmd". Block can be selected with CTRL-V. Allow '$' (end of line) for col2.
- Add open mode, use it when terminal has no cursor positioning.
- Special "drawing mode": a line is drawn where the cursor is moved to. Backspace deletes along the line (from jvim).
- Perform commands on multiple windows (:W%s/foo/bar/g), multiple arguments (:A) or multiple buffers (:B). Implement ":Bset", set option in all buffers. Also ":Wset", set in all windows, ":Aset, set in all arguments and ":Tset", set in all files mentioned in the tags file. Add buffer/arg range, like in ":2,5B%s/..." (do we really need this???) Add search string: "B/*.c/%s/.."? Or ":F/*.c/%s/.."?
- Support for underlining (underscore-BS-char), bold (char-BS-char) and other standout modes switched on/off with , 'overstrike' option (Reiter).

-   Add vertical mode (Paul Jury, Demirel): "5vdw" deletes a word in five
    lines, "3vitextESC" will insert "text" in three lines, etc..
4   Recognize l, #, p as 'flags' to EX commands:
    :g/RE/#l shall print lines with line numbers and in list format.
    :g/RE/dp shall print lines that are deleted.
    POSIX: Commands where flags shall apply to all lines written: list,
    number, open, print, substitute, visual, &, z. For other commands, flags
    shall apply to the current line after the command completes. Examples:
    :7,10j #l Join the lines 7-10 and print the result in list

---------- *votes-for-changes* ----------

In November 1998 an inquiry was held to allow Vim users to vote for changes to
Vim. Each person was allowed to give 10 positive and 5 negative points to a
list of items. Below is the result. This indicates the desire of users for
certain changes. This will be taken into account when deciding what to do
next from the huge list above.

The first number is the total number of votes. The number in brackets is the
portion of the total which were negative points.

317 (-5)   add folding (display only a selected part of the text)
252 (-10) vertically split windows (side-by-side)
197 (-3)   add configurable auto-indenting for many languages (like 'cindent')
183 (-5)   fix all problems, big and small; make Vim more robust
155 (-8)   add Perl compatible search pattern
125 (-1)   search patterns that cross line boundaries
125 (-1)   improve syntax highlighting speed
116   improve syntax highlighting functionality
101   add a menu that lists all buffers
93   (-1)   improve the overall performance
93   (-6)   multiple top-level windows for one running Vim
90   (-4)   be able to edit the command line with Vi commands
90   (-36) add a shell window (to type and execute shell commands in)
83   add patterns to define sections/paragraphs for "{", "[[", "%", etc.
81   improve Visual block mode: more commands that work on blocks
71   (-13) reduce the size of the executable and the runtime memory use
70   (-5)   improve the on-line help
68   improve "gq" formatting of text (left&right justified, 'comments')
68   (-2)   improve multi-byte character support
59   (-8)   improve the Visual Studio interface (VisVim)
59   (-12) make it possible to run Vim inside a window of another program
58   (-11) add an undo tree (be able to go back to any previous situation)
57   (-9)   add support for loading a shared library that defines new commands
54   (-1)   add a way to execute a command in multiple buffers/windows
52   improve the Perl interface
51   (-10) improve printing from gvim (File.Print menu entry)
50   (-20) make a "Vim lite" version, which is small and low on features
47   add a method to repeat a prev. executed change ("c.", "d.", etc.)
44   (-8)   add more features to the internal scripting language
37   add a command to repeat a whole mapping (not only its last change)
37   (-2)   add better support for editing files in projects (with ID utils)
33   (-3)   add handling for buffer-changed outside of Vim (when reg. focus)
28   improve Insert mode completion
28   (-14) add persistent undo (undo to before the file was saved/loaded)
25   improve command line completion
23   (-30) be able to run a program in a Vim window, with an interface to it
22   improve the quickfix commands
21   (-1)   add mappings and abbreviations local to a buffer
21   (-6)   add a special window for editing option values, with short docs

```
19   (-4)   add file locking
18   (-1)   add a way to disable (error) messages for a moment
17   (-5)   add encryption for loading/storing files and for the swapfile
10          improve the port for MacOS X Server aka Rhapsody
10          add Qt interface
10   (-1)   add scope arguments to the ":tag" command (like Elvis)
9    (-3)   improve the tutor (course for beginners)
8           improve the port for OS/2
8    (-1)   add a stack for saving/restoring options
8    (-3)   improve 'viminfo' (keeping information when quitting Vim)
8    (-6)   improve the performance of Vim scripts (pre-parse them)
7    (-2)   add more autocommand events (for ":cd", start Insert mode, etc.)
6    (-4)   improve the TCL interface
6    (-4)   add option to move the cursor where there is no text in Visual mode
5           improve the port for MacOS
4           improve the port for GTK
4           add Diff, Merge capability with CVS like in emacs.
4    (-5)   add option not to move the cursor when using a scrollbar
4    (-8)   add POSIX compatible search patterns
3           improve the port for X Window (use GTK, or Athena)
3    (-2)   improve the Cscope interface
3    (-14)  add option to move the cursor where there is no text in any mode
2           add support for Borland Delphi (to replace borland's def. editor)
2           add on-the-fly paragraph formatting / word wraps
1           improve the port for "EPOC 32"
1           add GUI to record/stop/play a keystroke macro
0    (-2)   improve ":mksession" support (switch to a previously saved state)
-1   (-1)   improve the port for * (fill in a system name at the *. E.g. VMS")
-4   (-77)  add more GUI functions (requesters, menus, dialogs)
-10  (-14)  add "open" mode, like the original Vi
-10  (-15)  improve the OLE interface
-10  (-17)  improve compatibility of the Ex mode
-12  (-89)  add on-the-fly spell checking
-13  (-24)  reduce the size of the distribution (harddisk usage)
-13  (-32)  improve Vi compatibility
-25  (-38)  add a lot of small features, instead of a few big ones
-47  (-53)  improve the 16 bit DOS version (avoid out-of-memory problems)
-81  (-123) improve gvim to fit in the MS-Windows look&feel (with an option)
-110 (-136) stop changing Vim, it's fine as it is
-117 (-122) remove functionality, there is too much of it

vim:tw=78:sw=4:sts=4:
vim: set fo+=n :
```

VIM Development - CVS; Goals, Anti-Goals, Projects

This page tells you about the current development of vim.

VIM Development - Overview

These are the various topics on this page:

Pro and Contra Discussions:

These two sections were moved out to separate HowTo files:

Vim Development - Current Issues

The Vim FAQ needs a rewrite. I am looking into DocBook for this. Help is welcome.

Vim OLD FAQ v1.25 980210

Vim NEW FAQ online 000605

Vim Development - General Info

The developers are adding new features and fixing bugs constantly. They are communicating and coordinating their efforts via the vim-dev mailing list. If you want to listen to it, or even take part then please do subscribe to this list!

Many people already have suggested some very good idea for Vim which are assembled on the Vim WishList. Please take a look at it, so you'll know whether your favourite missing feature has already been discussed. If you think you can add a feature then do send us the patches! Actually, this is the fastest way to get a feature into Vim. :-)

New features also need some testing, and all help on this is much appreciated. See below for some hints on testing and on how to write a bug report. Thanks!

000412: Latest patches for the developer versions is now available from the CVS Server. See the CVS Page for more info.

VIM Development - General Development Goals

What are the general goals of developemnt for Vim?

Here are some answers:

Text Editor

> Vim is a text editor. Vim supports editing of text. Vim is for editing text files. Vim lets you edit text fast. Repeat: Vim is a text editor.
>
> Note: Although Vim has the option "binary" to support editing of binaries, Vim was not written to be a "binaries editor" or "hex editor" at all. Vim ships with the utility "xxd" which turns binaries into text and back, thus allowing editing of bianries with a workaround. However, Vim is still is a "text editor".

Programmer's Editor

> Vim supports programming with features like automatic wrapping, easy indentation, reformatting of text and source code, tabs expansion, and syntax coloring.
>
> However, Vim is *not* a compiler, so it does not know about internal structures of programming languages. Hence it does not include "syntax checking", which is very different from "syntax coloring".

No GUI required

> Vim is supposed to work on all available terminals - however simple. Vim's commands should be powerful enough to make a mouse superfluous.
>
> However, Vim has support for GUI to allow menus and copy&paste via the mouse to accommodate casual users and those who are used to mice (eek! ;-). If you don't want to use Vim with a mouse then you can install it with the option "-gui" (ie "no GUI").

1.4MB Barrier

> Vim's distribution archive should fit onto a 1.4MB floppy disk.
>
> Once PCs ship with a standard disk drive of 100MB Zip drives, we'll overthrow Emacs and take over ze vorld - bwa-hahahaha! (Oops. :-)

NOTE: In November 1998 there was a vote on the features for Vim-6 - please take a look:

- VIM-6 - Vote for features - results

VIM Development - General Development Anti-Goals

There are some things that Vim does not try to be:

- Vim is not a code beautifier.
- Vim is not a compiler.
- Vim is not a ftp client.
- Vim is not a mailer.
- Vim is not a newsreader.
- Vim is not a print program.
- Vim is not a shell.
- Vim is not a spell checker.
- Vim is not a web browser.
- Vim is not a word processor.
- Vim is not an operating system.
- Vim is not another Emacs.

All this would just add more code (and bloat to Vim) and result in this compile option list for my Vim:

```
:ver
VIM - Vi IMproved 6.66 [WinLinux] (1999 Apr 01, compiled Apr  6 1999 12:07:00)
-babel -c64-linux -do-what-i-mean -emacs -gindent -kitchen-sink -lisp-decoder
-mud-frontend -netscape-explorer -not-so-regular-expressions -print-anywhere
-shell-o-rama -swahili-compiler -tea-hot-earl-grey -windows2000 -write-my-thesis
```

If you want an editor with a builtin ftp client, mailer, and newsreader then by all means, do use Emacs! I hope I have made a point here.

VIM Development - Vim-6 Features

The main goals for Vim-6:

- Auto-load of Changed Files
- File Locking Improvements
- **Folding**
- Project Editing
- **Remote Editing**
- Toolbar
- **Vertical Split**
- Vi Style Command Line Editing

For more info download the latest documentation ("runtime archive") and look at "todo.txt" or by entering the command `:help extensions-improvements`

Bram says about the vim-6 alpha versions:

```
Lots of things are not working yet.
Check ":help todo" for known items.
```

```
I NEED YOUR HELP: There is still a lot of work to be done.
If I have to do it all by myself it will take a very long
time until Vim.6.0 is ready.  Please give a hand by
implementing one of the items in the todo list.
```

Bram Moolenaar said on the maillist vim-multibyte on 000319:

```
Unicode support is planned for Vim version 6.0.  In Vim 5.x there is
double-byte support, which can be used for encodings that have a font
with double-width characters.  Thus the current multi-byte support
is rather limited and doesn't work for Unicode yet.
```

Vim Development - New Features with Examples

Here are some new features of the current development version. Please consult the announcements (available from the History page) for a full overview of the changes.

Here are some features/reasons why I have recommended to upgrade to the latest version:

vim-5.6 [TODO]

Still looking for contributions. Please send them to me at guckes@vim.org

vim-5.5 [TODO]

Still looking for contributions. Please send them to me at guckes@vim.org

vim-5.4 [still an alpha/developer version!]

Some of the new features of Vim-5.4:

```
- Port to the GTK+ GUI.
- A version-specific runtime directory has been introduced.  This makes it
  easier to upgrade to a newer version, or run two versions at the same time.
- Recognizing file types and syntax highlighting has been separated.
  This allows you to add your own file type specific items.
  And syntax highlighting is more flexible.
- Now includes syntax highlighting for about 150 file types.
- Vim scripts have been improved: Line continuation with a backslash, more
  functions, etc.
- The hit-return prompt is avoided by truncating messages.  A message history
  has been added to view previous (untruncated) messages.
- Support for encryption of files.
- Quickfix support extended to support more error formats, including
  multi-line error messages and change-directory messages.
- A number of useful commands and options added.
- Many improvements for existing commands.

- Console Mode Menus [vim-5.4e]:
  List expansions of current parameter on command line
  in the status line and allow sequential selection.
  (New option 'wildmenu' and new command ':emenu'.)

  - New option "write":  Diables *all* writing of files (incl ":w!").
  - New register for "last used searchpattern ("/).
  - ":bd" -> jump to last used window. (halleluja!)
```

```
- Mapping for creating a diff window
- Support for Linux console mouse
    - Support for GTK GUI
- X: input method for multi-byte characters.
  Hangul (Korean) Input Mode (and documentation).
- New commands:
  - "g?" for rotating text in ROT13 (also an operator).
  - "zH" and "zL" to scroll window horizontally by half a page.
  - "gm" move cursor to middle of screen line.
  - Operations on Visual blocks:
    "I", "A", "c", "C", "r", "<" and ">".
- Startup:  vim called as "ex -" now reads commands
  from stdin and works in silent mode
- Unix: Included the name of the user that compiled Vim and
  the system name it was compiled on in the version message.
- GUI: Automated numbering of Syntax menu entries in menu.vim.
- GUI: Clipboard support when using the mouse in an xterm.
- GUI (Motif): Popup menu.
```

Functions

vim-5.3 now offers "functions" (see ":help :functions").

VIM now does dishes, too! :-) [980826,990712]

http://www.multimania.com/phic/vim/vim_putz.html

http://altern.org/phica/vim/vim_putz.html [old]

vim-5.2f [980629]

Extra commands: "gJ" joins lines without adding spaces.

Regular expressions: You can now use "\d" to abbreviate a digit.

Example: Searching for `19\d\d` should now find (year) numbers like "1963", "1967", and "1994".

vim-5.0g - "strftime"

The internal function "strftime" gives the current date and time. This allows to easily insert these data in the cureent buffer without having to use an external command. Example: iab Ydate <C-R>=strftime("%y%m%d")<CR> " Example: 971027 " iab Ytime <C-R>=strftime("%H:%M")<CR> " Example: 14:28 " iab YDT <C-R>=strftime("%y%m%d %T")<CR> " Example: 971027 12:00:00 " iab YDATE <C-R>=strftime("%a %b %d %T %Z %Y")<CR> " Example: Tue Dec 16 12:07:00 CET 1997 "

Please let me know about *your* favourite new feature. Examples are most welcome!

Vim Development - Ports

[990803] One of the goals of Vim is to get it running on almost all systems. If you can help **porting** Vim to a system then please join us!

The port to MacOS definitely needs some help. The developers lack a current version of the CodeWarrier compiler, and the testers are just a few.

But we also get requests now and then for a port to WindowsCE. Here are the volunteers for the port so far:

```
991126 Name???          ahaack@bigfoot.com
990830 Kenneth Schwartz kens@avs.com
```

VIM Development - Projects

Development Projects and the people connected with it. This hopefully makes it easier for you to contact the persons who know about a special part of the Vim Code.

Thanks to everybody who offered help! :-)

000722: I have removed the name of someone who apparently received mail from a person who obviously thought that "XY support" means that he is offering support for some special feature. However, this is not true. The people here only *offered* their help at some point. That's all.

```
=== Code for special features (eg GUI)

Folding:
980918  David C. Jr. Harrsion   dharriso@pigseye.kennesaw.edu (offer)

Vertical split:
990115  Pjotr Kourzanoff        pjotr@IS.TWI.TUDelft.NL (offer)

=== Code for special features (eg GUI) and support for other programs

DDE support:
        Heiko Erhardt           Heiko.Erhardt@munich.netsurf.de

Developer Studio integration:
        Chris McKillop          cdmckill@csclub.uwaterloo.ca
        Paul Moore              Paul.Moore@uk.origin-it.com

GTK+ support:
981217  Marcin Dalecki          dalecki@cs.net.pl
981217  Andy Kahn               kahn@zk3.dev.com
Wished for by:
980915  Sung-Hyun Nam           namsh@lgic.co.kr
981110  Mitsuo Tsukamoto        tsuka@alpha.net.ph

Japanese support:
981129  Marc Espie              Marc.Espie@liafa.jussieu.fr (offer)
990225  Takuhiro Nishioka       takuhiro@super.win.ne.jp (testing?)

KVim - Vim for KDE
000921  Thomas Capricelli       capricel@yalbi.com
        http://aquila.rezel.enst.fr/thomas/vim/

POVRay and LCC's Wedit:
990123  Bob and Kelly Crispen   crispen@hiwaay.net
        offered to help integrating Vim as an editor these programs

Spell Checking support:
980526  Pancrazio `Ezio' de Mauro       pdm@datanord.it

Unicode UTF-8 support:
980715  Ron Aaron               ron@mossbayeng.com

=== Binary Maintenance
```

```
990206  Josh Howard             jrh@kashmir.vicor-nb.com
may provide vim binaries for FreeBSD, ie FreeBSD 2.2.8,
FreeBSD 3.0-RELEASE and FreeBSD 4.0-CURRENT - with and without X.

=== Documentation

Vim and Python:
Vim and Rebol:
990225  Timothy Johnson         tjohnson@akcache.com


=== Operating System Ports

Amiga:
970911  Michael Nielsen         mni@dde.dk
        will help developing the GUI

Atari:
        Jens M. Felderhoff      jmf@infko.uni-koblenz.de

Interix:
990217  John McMullen           john_mcmullen@interix.com
        [Intel Interix2.2 Service Pack 1 system]
        changes to makefile; ctags didn't compile.

Macintosh:
980123  Dany St-Amant           dany.stamant@sympatico.ca
        much work on the MacOS port.  Anyone have an up-to-date
        version of CodeWarrior to help him doing this?
        (See also the page on Vim on MacOS
```

VIM Development - Distribution Sites

Before you can test Vim, you need to get the source, of course.

See the distribution page for a list of sites that offer Vim.

The developers versions now reside in the subdirectory **unreleased** to clearly prevent casual users from using them. Also, for the "unreleased" versions there will be no binaries (esp no binaries for DOS or Windows), so you will have to "roll your own binary". Please do not distribute any binaries of unreleased versions, thankyou!

Beta versions of Vim can be found on the mirror sites in subdirectory **beta-test**. Even casual users are asked to test these versions, but you should still bear in mind that it is usually better to use a released version for the serious data - just in case. ;-)

Vim Development - Discussion on S-Lang

PRO S-Lang

- **Rapid prototyping**. New functionality can be implemented using S-Lang first, and integrated later in C if necessary (speed, "information hiding"). This is easy because S-Lang looks like C.
- **Development speedup**. The development of the current built-in evaluation language can be sped up as S-Lang gives much more functionality that would have to be developed step-by-step still. (Hey, Robert - "arrays"! :-)
- **Functional language**. S-Lang is a functional language which makes it easier for most non-expert to write additions and/or to make changes to existing code. It also reduces the amount of time spent for debugging strange expressions.
- **Wealth of Existing code**. S-Lang is already used with several successful program, most notably with the author's editor (jed) and the author's newsreader slrn; and the mailer mutt can make use of it, too. This means that a wide range of users can make use of the already existing wealth of functions that have already been written for Slang.
- **Import/export**. S-Lang can be used in both directions: the first is to export the functionality of VIM (eg. Get/SetCursorPosition, inserting of character or strings, setting of the fore- and background colours) to be used in scripts. The other one is the use of "hooks" which are called for certain events (eg. for opening/closing a file, in case of errors). This allows for a very flexible approach to some kind of "event handling" mechanisms which can be put to use with syntax coloring.
- **Easy integration**. S-Lang is very easy to integrate. The basic inclusion of the interpreter is about 5 lines of code. The size of the whole interpreter library (slang.lib) for the windows95 Watcom 10.6 compiler is 240K. The export of VIM internal functions can be an evolutionary process.
- **Byte-compiler = fast startup**. S-Lang code can be byte-compiled. This speeds up the loading process of the functions. In addition, functions which are not neccesary during the startup of VIM can be registered to be loaded later (this is actually done automatically by the S-Lang interpreter).
- **Creation of a "VIM library"**. Functions can be put into modules which a user can choose from. It is even possible in the next version of S-Lang to load binaries (= DLL's, modules) during runtime to enhance VIM. Examples are an pop3/imap mail interface for reading, writing and managing mails, or even CORBA to interface VIM with other CORBA-aware applications.
- Support for DOS, Windows, OS/2, and BeOS (MacOS is in the works). S-Lang has been ported to Windows with the exception of a few bios interrupts that Windows doesn't allow, a DOS dpmi app is perfectly happy running as a Windows console app, or a DOS box. From the S-Lang site at ftp://space.mit.edu/pub/davis/slang/README
  "S-Lang has been successfully compiled and tested on many platforms and OSs (Unix/VMS/PC-MSDOS/OS2). Building the S-Lang library requires a C compiler that understands function prototypes. On SunOS, you will need gcc."
- Text dialogs and menus. Dialogs and menus on terminals is what many users want.

CONTRA S-Lang

- Do we really need another scripting language?
- Implementation of new features in S-Lang might be sufficient for most people, thus hampering the implementation of solutions within C.
- By moving the implementation of functionality to a scripting language and allowing everybody to exchange functional modules there is no "central authority" for releasing the one true version of VIM, right?
- Someone has to work in the hooks for S-Lang into the code.

S-Lang HomePage

http://www.s-lang.org/

http://space.mit.edu/~davis/jed.html)

Vim Development - Discussion on Speed

Bram Moolenaar on vim-dev on 990727: Before starting discussions about code size, I would rather hear remarks about real problems. For example, when Vim starts up a bit slow. And then make sure it's not because of your large .vimrc or 'viminfo' setting. [...] Actually, on my own system the main slowdown on startup is cause by viminfo. Especially when there are a lot of file names in it. Some syntax files load a bit slow too. More accurate measurements should be done to pinpoint the real problems (anyone with a good profiler can have a try). I suspect that dynamic loading could actually slowdown Vim startup. For example, if syntax highlighing is dynamically loaded, and you do ":syn on" in your .vimrc, it would require another series of system calls to load the syntax stuff. That is guaranteed to be slower than including the code in the executable.

Vim Development - Discussion on IDE

Many users would like to use Vim as their editor with many other programs, most specifically as the editor for composing emails. OutlookExpress, NetscapeMessagenger, NetMail95, Eudora, and Pegasus.

The workaround

Vim can access the clipboard which allows the following workaround:

- Select the message with the mouse.
- Copy the selected text to the clipboard (CTRL-C).
- Make sure that the option "guiclipboard" contains the flag 'a':

```
:set go+=a
```

- Import the text from the clipboard using the register '*', ie paste it into the current buffer with the command `"*p`.
- Edit the text.
- Copy the edited text into the clipboard, eg with `:%"*y`. (untested)
- Use CTRL-V to paste the text from the clipboard into the application.

Vim Development - Discussion on CORBA

Should Vim add support for CORBA?

PRO CORBA

- Integration with other programs as an editor-of-choice.

CONTRA CORBA

- Yet more code. -> Bloat!
- Let other programs add the support to specify an editor-of-coice.
- Focus on code for "terminal editing" - not on support for other programs.
- The CORBA FAQ says that "Some applications, such as word processing, might not benefit from distribution at all."
- Requires specification of CORBA interface.
- Requires to split Vim into components - and splitting away the syntax coloring is very difficult. Each component then has to work before you can use it. break one component and you'll lose everything on top of it. Separating Vim into components also means that you need to copy every component along to make it work. Forget about the "Vim fits on a floppy" then.
- Definition of interfaces often change which makes Vim more dependant to these changes. More work will then have to be put on those.

- You use CORBA when the components are spread acorss the network - does anyone want to "spread Vim's parts" around?
- Components are no guarantee that the resulting programs are small (see Micro$soft Office).
- There is no standard for "editor" components so far, anyway.

Facts:

- Creation of a Vim ORB is required to allow Vim as an object for their applications.

Open Questions:

- Vim+CORBAR requires Vim to become an ORB. Code for this costs money. Will Vim then be freely available still?

  SEE ALSO:

- http://www.aurora-tech.com/CORBA-FAQ.htm
- MICO ("MICO Is CORBA") http://www.mico.org
- TAO http://siesta.cs.wustl.edu/~schmidt/
- .. http://galaxy.uci.agh.edu.pl/~vahe/CORBA.htm
- .. http://adams.patriot.net/~tvalesky/freecorba.html
- .. http://www.uk.research.att.com/omniORB/

Vim Development - Bram's Statements

Bram on vim-dev about using X resources within Vim [010314]:

```
I know X11 applications are supposed to use resources, but I happen not
to like them.  I would greatly appreciate putting in some effort to
avoid having to use them.  A Vim user will be much happier when he can
do all settings for Vim in his vimrc file, instead of having to learn to
use resource files (I know I never fully understood them, can you
imagine what trouble a normal user would run into?).

Using Netscape as an example: I tried adding support for a scroll mouse
by editing the resource file.  Even with directions from other users
it's still not working in more than the main window.  Editing that
_huge_ resource file feels more like programming to me anyway.  I
wouldn't want to have end users having to understand resource files.
```

Bram on vim-dev about the suggestion to include a "nice feature" [990308]: > That would be nice as well. '"would be nice" is not a very good reason to include this!' So, folks, please give a few *good* reasons why your favourite feature should be added to Vim. Remember that Vim is supposed to be as small and fast as possible. Adding an internal browser, mailer, newsreader, and a kitchen sink is a job for "that other editor". ;-)

```
URL:         http://www.math.fu-berlin.de/~guckes/vim/deve.html
URL:         http://www.vim.org/deve.html (mirror)
Created:     Wed Nov  1 00:00:00 MET 1995
```

Send feedback on this page to
Sven Guckes guckes@vim.org

Vim History - Release Dates of User Versions and Developer Versions

This page shows the "history" of Vim, ie the list of (alpha and beta) versions of Vim from past to present.

This list will be updated with every new release, so it should be ideal to use it with a web notification service if you want to be informed about new releases. Then again, it is probably much easier and better to subscribe to the vim-announce mailing list for this.

Vim Release History

If you are curious about the recent changes then take a look at the announcements. They are listed here with their release dates. User releases are marked with "USER" and development releases are marked with "DEV".

NOTE: The latest USER release is **Vim-5.8** - released on 31st May 2001. The previous user release, Vim-5.7, was released almost a year before that, and since then there have been about 30 patches which have been incorporated to make Vim even more stable than it already were. Also, a lot of syntax files have been added or updated. Since Vim-5.6 the development of Vim now uses "CVS". Almost 100 patches have been applied to Vim-5.6; this was released as Vim-5.7 which is considered to be the "most stable version ever".

So - please **upgrade to Vim-5.8** - thankyou!

[010614] Development for Vim-6 is still in alpha phase. Testing is only suggested for those who know the program well and really know what they are doing. Disclaimer is as usual: Expect loss of data - keep backups! ;-)

VIM History - Official Release Dates of Vim Versions

2001

VIM-6 releases:

DEV	010701	Sun, 01 Jun 01	vim-6.0am
DEV	010624	Sun, 24 Jun 01	vim-6.0al
DEV	010617	Sun, 17 Jun 01	vim-6.0ak
DEV	010610	Sun, 10 Jun 01	vim-6.0aj
DEV	010603	Sun, 03 Jun 01	vim-6.0ai
USER	**010531**	**Thu, 31 May 01**	**vim-5.8**
DEV	010527	Sun, 27 May 01	vim-6.0ah
DEV	010520	Sun, 20 May 01	vim-6.0ag
DEV	010513	Sun, 13 May 01	vim-6.0af
DEV	010506	Sun, 06 May 01	vim-6.0ae
DEV	010429	Sun, 29 Apr 01	vim-6.0ad
DEV	010422	Sun, 22 Apr 01	vim-6.0ac
DEV	010416	Mon, 16 Apr 01	vim-6.0ab
DEV	010404	Wed, 04 Apr 01	vim-6.0aa
DEV	010318	Sat, 24 Mar 01	vim-6.0z
DEV	010318	Sun, 18 Mar 01	vim-6.0y
DEV	010311	Sun, 11 Mar 01	vim-6.0x
DEV	010226	Mon, 26 Feb 01	vim-6.0w
DEV	010204	Sun, 04 Feb 01	vim-6.0u
DEV	010121	Mon, 21 Jan 01	vim-6.0t
DEV	010114	Mon, 14 Jan 01	vim-6.0s
DEV	010101	Mon, 01 Jan 01	vim-6.0r

2000

VIM-6 releases:

DEV	001217	Sun, 17 Dec 00	vim-6.0q

DEV 001210   Sun, 10 Dec 00   vim-6.0p
DEV 001203   Sun, 03 Dec 00   vim-6.0o
DEV 001119   Sun, 19 Nov 00   vim-6.0n
DEV 001112   Sun, 12 Nov 00   vim-6.0m
DEV 001105   Sun, 05 Nov 00   vim-6.0l
DEV 001029   Sun, 29 Oct 00   vim-6.0k
DEV 001022   Sun, 22 Oct 00   vim-6.0j
DEV 001015   Sun, 15 Oct 00   vim-6.0i
DEV 000831   Thu, 31 Aug 00   vim-6.0h
DEV 000820   Sun, 20 Aug 00   vim-6.0g
DEV 000813   Sun, 13 Aug 00   vim-6.0f
DEV 000806   Sun, 06 Aug 00   vim-6.0e
DEV 000730   Sun, 30 Jul 00   vim-6.0d
DEV 000723   Sun, 23 Jul 00   vim-6.0c
DEV 000716   Sun, 16 Jul 00   vim-6.0b
DEV 000709   Sun, 09 Jul 00   vim-6.0a

VIM-5 releases:

USER 010531   Thu, 31 May 01   vim-5.8
USER 000624   Sat, 24 Jun 00   vim-5.7
USER 000116   Sun, 16 Jan 00   vim-5.6

1999

DEV 990829   Sun, 29 Aug 99   vim-5.5a

USER 990726   Mon, 26 Jul 99   vim-5.4

DEV 990719   Mon, 19 Jul 99   vim-5.4p
DEV 990711   Sun, 11 Jul 99   vim-5.4o
DEV 990621   Wed, 21 Jun 99   vim-5.4n
DEV 990609   Wed, 09 Jun 99   vim-5.4l
DEV 9905??   Sun, 16 May 99   vim-5.4k
DEV 990516   Sun, 16 May 99   vim-5.4j
DEV 990502   Sun, 02 May 99   vim-5.4i
DEV 990418   Sun, 18 Apr 99   vim-5.4h
DEV 990322   Mon, 22 Mar 99   vim-5.4g
DEV 990308   Mon, 08 Mar 99   vim-5.4f
DEV 990217   Wed, 17 Feb 99   vim-5.4e
DEV 990125   Mon, 25 Jan 99   vim-5.4d
DEV 990106   Wed, 06 Jan 99   vim-5.4c

```
DEV  981221     Mon, 21 Nov 98   vim-5.4b
DEV  981102     Mon, 02 Nov 98   vim-5.4a

USER 980831     Mon, 31 Aug 98   vim-5.3
USER 980824     Mon, 24 Aug 98   vim-5.2

DEV  980817     Mon, 17 Aug 98   vim-5.2k
DEV  980810     Mon, 10 Aug 98   vim-5.2j
DEV  980803     Mon, 03 Aug 98   vim-5.2i
DEV  980727     Mon, 27 Jul 98   vim-5.2h
DEV  980713     Mon, 13 Jul 98   vim-5.2g
DEV  980629     Mon, 29 Jun 98   vim-5.2f
DEV  980615     Mon, 15 Jun 98   vim-5.2e
DEV  980602     Mon, 02 Jun 98   vim-5.2d
DEV  980518     Mon, 18 May 98   vim-5.2c
DEV  980504     Mon, 04 May 98   vim-5.2b

USER 980406     Mon, 06 Mar 98   vim-5.1

DEV  980330     Mon, 30 Mar 98   vim-5.1b
DEV  980316     Mon, 16 Mar 98   vim-5.1a
USER 980219     Thu, 19 Feb 98   vim-5.0

USER 98xxxx     Mon, 16 Feb 98   vim-5.0x
USER 98xxxx     Mon, 09 Feb 98   vim-5.0w
USER 98xxxx     Mon, 26 Jan 98   vim-5.0v
USER 98xxxx     Mon, 12 Jan 98   vim-5.0u
USER 98xxxx     Tue, 23 Dec 97   vim-5.0t
USER 98xxxx     Mon, 08 Dec 97   vim-5.0s

DEV  971124     Mon, 24 Nov 97   vim-5.0r
DEV  971110     Mon, 10 Nov 97   vim-5.0q
DEV  971020     Mon, 20 Okt 97   vim-5.0p
DEV  970929     Mon, 29 Sep 97   vim-5.0o
DEV  970917     Wed, 17 Sep 97   vim-5.0n
DEV  970827     Wed, 27 Aug 97   vim-5.0m
DEV  970728     Mon, 28 Jul 97   vim-5.0l
DEV  970611     Wed, 11 Jun 97   vim-5.0k
DEV  970602     Mon, 02 Jun 97   vim-5.0j
DEV  970527     Tue, 27 May 97   vim-5.0i
DEV  970513     Tue, 13 May 97   vim-5.0h
DEV  970501     Thu, 01 May 97   vim-5.0g
DEV  970423     Wed, 23 Apr 97   vim-5.0f
DEV  970415     Tue, 15 Apr 97   vim-5.0e
DEV  970408     Tue, 08 Apr 97   vim-5.0d
```

```
DEV  970402     Wed, 02 Apr 97   vim-5.0c
DEV  97XXXX     XXX, XX XXX XX   vim-5.0b - not released
DEV  97XXXX     XXX, XX XXX XX   vim-5.0a - not released
```

Vim Milestones

DATE	VERSION	Milestone
2001 ??? ??	Vim 6.0	Folding (and more)
2000 Jul 09	Vim 6.0a	Folding (and more)
1998 Feb 19	Vim 5.0	Syntax coloring/highlighting
1996 May 29	Vim 4.0	Graphical User Interface (Robert Webb).
1994 Aug 12	Vim 3.0	Support for multiple buffers and windows.
1992	Vim 1.22	Port to Unix. Vim now competes with Vi. This was when Vim became Vi IMproved.
1991 Nov 2	Vim 1.14	First release (on Fred Fish disk #591).

```
URL:          http://www.math.fu-berlin.de/~guckes/vim/hist.html
URL:          http://www.vim.org/hist.html (mirror)
Created:      Thu Jun 11 11:11:11 CEST 1998
```

Send feedback on this page to
Sven Guckes guckes@vim.org

Vim 5.x for MacOS

News | System Requirement | Download the latest version | Work in progress | Known bugs | Todo | Version history

Updated on January 21st, 2000

This page tries to keep up to date all the information about the current status of the port of Vim 5.x to the Macintosh. Since I'm trying to work on the port, this page is awfully designed.

What is Vim?

Vim is a text-editor which prone the inverted main MacOS rule. With Vim everything that can be done with the mouse can be done with the keyboard.
For more information please: VIM Homepage

Work in progress (included in latest patch)

- Including Vim Help inside the MacOS Help Menu.
  - Coding: done
  - Testing: seems to work fine
  - Sideeffect: a localized version of "menu.vim" and of the MacOS will give two 'Aide', 'Hilfe' or whatever menu.
- Adding Contextual Menu support.
  - Coding: done
  - Testing: seems to work fine
- Adding support for CodeWarrior AppleEvent
  - Coding: mainly done
  - Testing: in-progress, few unresolved issues
  - Issues:
    - Modified flag confusing (maybe cause by changing creation date)
    - <CTRL-L> needed when double-clicking error/warning/search
    - Double-clicking error/etc causes override warning if the file as been edited
    - Unable to erase the backup file when Codewarrior display the errors in the file

Requirements

- System 7.0 or better (tested on System 7.5, MacOS 8.1 and MacOS 9)
- 68020 or better (tested on 68040 and G3)
- Since the development is done on a G3, the performance on 68k may be sluggish.

What are the current versions?

Version 5.6 (MacOS Release 1)

The version 5.6 for MacOS should be considered in as an advanced beta stage.
(Some MacOS Human Interface Guideline are bent, and some Vim functionality are not yet available)

Deficiencies & Known Problems

- The first Python command must be :python from vim import *
- Apple Event: Print, Run and Quit
- Menu: Apple menu not handled
- Fonts: Display properly proportional font
- Fonts: Can't use space in font name
- Source files: MacOS fileformat supported but slow and not recommended.
- Tags file must be in macintosh format
- filename completion doesn't work for volume name in mac-like path
- Mapping option key doesn't work properly with the <M-...< syntax beside with function key and mouse.
- Command key doesn't work with the mouse.
- Slow on 68k

Todo

- Vim features
  - Toolbar
  - Tearoff Menus
  - Cursor Blinking
- MacOS integration
  - Shortcut for menus

Tips'n'Tricks

- Monaco:h9, MPW:h9, Mishawaka:h9 and Courier:h10 co-exist wonderfully.
  Mishawaka comes with Eudora
  MPW comes with MPW and CodeWarrior

Links

Axel Kielhorn's Vim for MacOS Web Page
Vim Pages
Vim Pages:Macs
Vim Pages:Mailing List

Made on Macintosh with vim 5.x

From: Laurent Duperval <laurent@Grafnetix.COM> Newsgroups: comp.editors,news.answers,comp.answers Subject: Vim Editor FAQ Date: 10 Feb 1998 02:29:18 GMT Message-ID: <6boe1u$sau@tandem.CAM.ORG> Archive-name: editor-faq/vim Posting-Frequency: monthly (second Monday) Last-modified: Mon Feb 9 21:27:50 EST 1998 URL: http://www.grafnetix.com/~laurent/vim/faq.html The Vim Editor FAQ Created: Tue Mar 12 00:00:00 EST 1996 Last Updated: Date: 1998/02/08 18:03:21 Version: Revision: 1.25 Author: Laurent DUPERVAL <laurent@grafnetix.com> This Page: <URL:http://www.grafnetix.com/~laurent/vim/faq.html> Vim Pages: <URL:http://www.vim.org/> This article contains answers to Frequently Asked Questions about the Vim editor. Questions marked by [CHANGED] are questions that have been modified since the last release of the FAQ. Questions marked by [NEW] are new questions. The following topics are addressed: Contents

auto-indenting but towards the left margin instead of the right margin)? o 5.30 Is there a repository for Vim macros? o 5.31 If I have a mapping/abbreviation whose ending is the beginning of another mapping/abbreviation, how do I keep the first from expanding into the second one? o 5.32 Modelines are cool but Vim's modeline support is too restrictive. How can I improve it? o 5.33 How can I use different .vimrc settings for different types of files? o 5.34 How can I search for tags when running Vim over a telnet session? o 5.35 [NEW] I get errors when trying to put scripts in my mappings. Why? o 5.36 [NEW] When I try to remove mappings in my autocommands, Vim says my mappings don't exists. What's going on? * 6 DOS-, WINDOWS-, WIN32-SPECIFIC QUESTIONS o 6.1 Why does the Win32 version of Vim update the screen so slowly on Windows 95? o 6.2 So if the Win32 version updates the screen so slowly on Windows 95 and the 16-bit DOS version updates the screen quickly, why would I want to run the Win32 version? o 6.3 And what about the 16-bit DOS version versus the Win32 version on NT? o 6.4 Why can't I paste into Vim when running Windows 95? o 6.5 How do I type dead keys on Windows 95? o 6.6 How do I type dead keys on Windows NT? o 6.7 When will a real GUI version of Vim (gvim) for Win32 with scrollbars, menus, pasting from the clipboard, and so on become available? o 6.8 On a Win32 machine, I'm using Vim to edit a symbolically linked file on a Unix NFS file server. When I write the file, Vim does not "write through" the symlink. Instead, it deletes the symbolic link and creates a new file in its place. Why? o 6.9 How do I copy text from Windows applications to the DOS version of Vim? o 6.10 Why does my Caps Lock affect all the keys characters for all the keys instead of just the letters? o 6.11 How do I change Vim's window size in Windows? * 7 UNIX-SPECIFIC QUESTIONS o 7.1 How do I turn off the message ``Thanks for flying Vim'' on Unix stations? o 7.2 How do I prevent <Ctrl-Z> from suspending Vim? o 7.3 How can I make Vim faster on a Unix station? o 7.4 In Unix, how do I make Vim more colorful? 1 ABOUT THIS FAQ 1.1 What versions of Vim does this FAQ cover? At the time of its creation, version 4.0 was almost complete (3.21 was already available) therefore the FAQ covers mainly vim4-specific issues. But some answers may apply to Vim 3.0. There are no plans to make the FAQ ``backward compatible''. ;-) In particular, in various parts of the FAQ, you will see mentions of ``:h subject'' to get help about a specific subject. Although the help command works in 3.0, you cannot give it an argument. 1.2 Who maintains this FAQ? The current maintainer of this FAQ is Laurent Duperval. I can be reached at <laurent@grafnetix.com>. Please insert the keyword NOTSPAM in the Subject line of your message otherwise it may not reach me. 1.3 Why him? Well, I volunteered. You could have done it too, had you volunteered before me. ;-) Besides, after getting so much free and useful stuff off the 'Net, I decided to give something back in the form of this FAQ. 1.4 Can I add to it? Sure. Just send your questions (and answers!) to <laurent@grafnetix.com>. Comments and constructive criticism are always welcome. 1.5 What are the restrictions on this FAQ? Disclaimer: This article is provided as is without any express or implied warranties. While every effort has been taken to ensure the accuracy of the information contained in this article, the author / maintainer / contributors (take your pick) assume(s) no responsibility for errors or omissions, or for damages resulting from the use of the information contained herein. This article is 1.6 [CHANGED] Acknowledgements The following people have contributed (from near or far) to the answers in

this FAQ. Some of them may not even know about it. Bram Moolenar <bram@vim.org> Sven Guckes <guckes@vim.org> Robert Webb <robertw@wormald.com.au> George V. Reilly <georger@halcyon.com> Tony Nugent <tonyn@sctnugen.ppp.gu.edu.au> Ingolf Markhof <markhof@ls12r.informatik.uni-dortmund.de> Stan Brown <stbrown@nacs.net> Raul Segura Acevedo <raul@turing.iquimica.unam.mx> Benjamin Elijah Griffin <eli@NetUSA.Net> Dr. Charles Campbell <cec@gryphon.gsfc.nasa.gov> 2 GENERAL INFORMATION 2.1 What is Vim? Vim stands for Vi IMproved. It used to be Vi IMitation, but there are so many improvements that a name change was appropriate. Vim is a text editor which includes almost all the commands from the Unix program ``Vi'' and a lot of new ones. It is very useful for editing programs and other 7-bit or 8-bit ASCII text. All commands can be given with the keyboard. This has the advantage that you can keep your fingers on the keyboard and your eyes on the screen. For those who want it, there is mouse support and a GUI version with scrollbars and menus. Vim is an editor, not a word processor. A word processor is used mainly to do layout of text. This means positioning it, changing the way it appears on output. More often than not, the final document is meant to be printed or typeset or what have you, in order to present it in a pleasing manner to others. Examples of word processors are Microsoft Word, WordPerfect, FrameMaker, and AmiPro. An editor is simply for entering text. Any typesetting or laying out of the document is secondary. With an editor, one's main concern is entering text, not making the text look good. Examples of editors other than Vim and Vi are Emacs, Crisp, Brief, and xedit. 2.2 Who wrote Vim? Most of Vim was written by Bram Moolenar, with contributions from too many people to mention here. Try ``:h credits'' for a complete list. Vim is based on Stevie, worked on by Tim Thompson, Tony Andrews and G.R. (Fred) Walter. 2.3 Is Vim compatible with Vi? Very. A special mode (``:set compatible'') makes Vim behave almost exactly like Vi. 2.4 What are some of the improvements of Vim over Vi? Here is a short summary. For more information, do ``:h diff''. Multi-level undo Allows you to set the number of times you can undo your changes in a file buffer. You can also redo an undone change. Multiple windows and buffers Each file can be displayed in its own window. You can move easily from one window to another. Each file opened during a Vim session also has an associated buffer and you can easily jump from one to the other. Repeat a series of commands. Vim has a facility which allows you to record a sequence of typed characters and repeat them any number of times. Flexible insert mode. Vim allows you to use the arrow keys while in insert mode to move around in the file. No more hitting <Esc>, moving around, then hitting `i' or `a'. Visual mode. You can highlight sections of text and execute operations on this section of text only. Block operators Allow selection and highlighting of rectangular blocks of text in order do execute specific operations on them. Online help system. You can easily find help on any aspect of using Vim. Help is displayed in its own window. Command-line editing and history. History allows you to use the arrow keys to repeat or search for a command that has already been typed. Allows you to match the beginning of a command with the beginning of another similar command in the history buffer. You can also edit a command to correct typos or change a few values. Command line completion. Using the <Tab> key, you can complete commands, options, filenames, etc. as needed. Horizontal scrolling. Long lines can be scrolled horizontally (with or without the GUI). Text formatting. With two keystrokes, you can format large sections of text, without the use of external programs. Edit-compile-edit speedup. You can compile within Vim and automatically jump to the location of errors in the source code. Improved indenting for C programs Vim gives you more control over how your C programs appear on screen. Searching

for words in include files Vim allows you to search for a match of the word under the cursor in the current and included files. Word completion in Insert mode Vim can complete words while you are typing, by matching the current word with other similar words in the file. Automatic commands Commands automatically executed when reading or writing a file, jumping to another buffer, etc. Viminfo Allows storing of the command line history, marks and registers in a file to be read on startup. Mouse support The mouse is supported in an xterm and for MS-DOS. It can be used to position the cursor, select the visual area, paste a register, etc. Graphical User Interface (GUI) (Motif and Athena) You can use the GUI and have access to a menu bar, scrollbar, etc. You can also define your own menus as well as do many operations with the mouse instead of the keyboard. 2.5 What are the improvements of Vim 4 over Vim 3.x? Here is a list of some of the improvements of Vim 4 over Vim 3. For a complete list, do ``:h vim_40". * New on-line help system * Command-line editing improved * Improved indenting for C programs * Searching for words in include files * Word completion in Insert mode * Automatic commands * Text objects * New Options * Support for editing one-line paragraphs * Usage of key names * Viminfo * Compilation improvements * Tag support improved * Improved (error) messages * Swap file * Mouse support * Graphical User Interface (GUI) * Support for Windows 95 and NT * Vi compatibility improvements * And more! (As if all that preceded wasn't reason enough to upgrade) 2.6 What are the improvements of Vim 5.x over Vim 4.x? Vim 5.0 is not officially out yet, but it will have syntax highlighting! And a command language! And much, much more! 2.7 Is Vim free? Vim is Charityware. There are no restrictions on using or copying Vim, but the author encourages you to make a donation to charity. A document explaining how to do so is included in the distribution. You are allowed to include Vim on a CD-ROM but you should send the author a copy. Please try ``:h copying". 3 RESOURCES 3.1 [CHANGED] Where can I learn more about Vim? A mailing list is available for Vim. See the next question. Vim does not have a newsgroup of its own. But the appropriate newsgroup to post to is comp.editors. There is a Vim home page available at <URL:http://www.vim.org/> More information can be found in * Eli's Vim Page <URL:http://www.netusa.net/~eli/src/vim.html> * The Vi Lovers Home Page <URL:http://www.cs.vu.nl/~tmgil/vi.html> There is a reference card which is only valid for version 5. It is available in Postscript by sending email to <raul@turing.iquimica.unam.mx>. Please specify a subject as follows: * ``send reference card" for letter version * ``send reference card a4" for a4 version * ``send reference card tex" for the original LaTeX source Oleg Raisky <olrcc@scisun.sci.ccny.cuny.edu> has created a reference guide for Vim and is available at <URL:http://scisun.sci.ccny.cuny.edu/~olrcc/vim/>. There are versions available for A4 and US Letter paper sizes. 3.2 Is there a mailing list available? There are three mailing lists for Vim (description to follow). You can join any of the lists by sending an empty mail message to the appropriate list handler. If for any reason, things don't work, contact <vim-help@vim.org>. * To (un)subscribe to the Vim Help list mail vim-(un)subscribe@vim.org * To (un)subscribe to the Vim Announcements list mail vim-announce-(un)subscribe@vim.org * To (un)subscribe to the Vim Development list mail vim-dev-(un)subscribe@vim.org Each mailing list serves a different purpose and you should not crosspost between them. This is a brief description of each list: <vim@vim.org> For discussions about using existing versions of Vim: Useful mappings, questions, answers, where to get a specific version, etc. <vim-dev@vim.org> For discussions about changing Vim: New features, porting, etc. <vim-announce@vim.org> Announcements about new versions of Vim and also beta-test versions and ports to different systems. No

discussions here, please. If you have a question about the usage of Vim then please post it to comp.editorsor to the vim mailing list. Please note that if you send a message to a Vim mailing list but are not subscribed, your message will be discarded. You must subscribe in order to send mail to the mailing lists. Do not send mail to the mailing lists for subscription or unsubscription. (The maintainer of the list hates that! So do the people subscribed to the lists.) 3.3 Is there an archive available for the Vim mailing list? There is an archive available for the announcements made on the vimannounce mailing list at <URL:http://www.findmail.com/listsaver/vimannounce.html>. There is also a mail archive available by FTP if you need to check out old messages sent to one of the other lists. They are available at: <URL:ftp://ftp.ii.uib.no/pub/vim/mail-archive/vim/maillist.html> <URL:ftp://ftp.ii.uib.no/pub/vim/mail-archive/vimdev/maillist.html> <URL:ftp://ftp.ii.uib.no/pub/vim/mail-archive/vimannounce/maillist.html> 3.4 [CHANGED] Where can I report bugs? Well, you don't need to because there are none. ;-) But on the off chance that you may find an unexpected feature, you may send a description of it to the Vim Developmentlist. Take a look at <URL:http://www.vim.org/deve.html#bugreport> to see what type of information should be sent when making a bug report. There is also a script included with newer versions of Vim, which can be used to create a bug report. We are looking for someone to take over the tasks of our previous bug maintainer. If you are interested, send a message to Bram <bram@vim.org>. If you have patches that you would like to submit for approval and incorporation in future versions of Vim, you can send them to Bram <bram@vim.org>. 3.5 Where can the FAQ be found? This FAQ will be posted on a (hopefully) regular basis to the comp.editorsnewsgroup. The latest version can be found on <URL:http://www.grafnetix.com/~laurent/vim/faq.html>. It will also be mirrored at <URL:http://www.vim.org/faq/>. Eventually, it should make its way into news.answersand rtfm.mit.edu. You can also find a searchable version of this and many more FAQs at the Hypertext FAQ Archiveat <URL:http://www.landfield.com/faqs/>. 3.6 What if I don't find an answer in this FAQ? This FAQ covers mainly Vim-specific questions. You may find more information suitable for most Vi clones by reading the Vi FAQ. It is posted regularly on comp.editors. You can also find a copy at <URL:ftp://rtfm.mit.edu/pub/usenet-by-group/comp.editors>. Please note that although I maintain the FAQ, I am not the best resource to answer questions about Vim. If you send a question about using Vim diretly to me, 9 times out of 10 I will redirect you to the Vim mailing listor to comp.editors. So you might as well send your questions directly there since you're liable to get a more accurate and useful response than I could give you. 4 AVAILABILITY 4.1 [CHANGED] What is the latest version of Vim? The latest version is 4.6. The latest beta release is 5.0w (as of February 8, 1998). 4.2 Where can I find the latest version of Vim? The main archive for the latest versions of Vim is <URL:ftp://ftp.oce.nl/pub/vim> The directory structure: /pub/vim/amiga /pub/vim/atari /pub/vim/beta-test /pub/vim/os2 /pub/vim/pc /pub/vim/unix The latest public release can be found in the atari, amiga, os2, pc, and unix directories. The latest beta-test versions can be found in the beta-test subdirectories. For a complete and updated list of current FTP sites, please refer to <URL:http://www.vim.org/dist.html>. 4.3 [CHANGED] What platform does it run on? Vim should compile with no problems on most flavors of Unix and on DOS, OS/2, Ataris, Amigas, Windows 95, Windows NT, BeOS and VMS. Binaries for other systems are also available. A complete list of supported platforms and available binaries can be found at

<URL:http://www.vim.org/dist.html> 4.4 [CHANGED] What do I need to compile and install Vim? Unless you are on a Unix station, you may not need to compile Vim since there are various binaries available for Windows, Windows 95, Windows NT, DOS, OS/2 and Amiga (please see <URL:http://www.vim.org/dist.html#binaries>. But if you need to compile it, you must have a C compiler and a make utility. Most commercial C compilers come bundled with a make utility, so if you've got one of those, you're in luck. If you need a compiler or if your make utility doesn't like the Makefiles provided with Vim, you can get the GNU C compiler and GNU make at <URL:ftp://prep.ai.mit/edu/pub/gnu> and its mirrors. At that same site, you can also find sed, binutils, and other GNU tools (such as gzip) which can be useful on certain systems. For DOS systems, you can look in <URL:ftp://ftp.coast.net/SimTel/msdos/>. If you plan on compiling the GUI version of Vim under Unix, you must have some kind of widget library. One of these three should suffice: Motif libraries These libraries are commercial but are distributed with a number of operating systems. You can find out how to purchase it by going to <URL:http://www.rdg.opengroup.org/>. Athena libraries This is a freely available widget set which is included in the X11 distribution. This is a no frills library which implements a number of widgets that are necessary when building a GUI. Various other similar libraries have been built from the Athena widget set. These variations allow a 3D look and a NextStep look. These variations are not part of the X11 distribution. The X11 distribution is available at <URL:ftp://ftp.x.org/>. Lesstif This is a free Motif clone. It is available at <URL:http://www.lesstif.org/>. 5 TIPS AND TECHNIQUES Note: This section contains various macros written in Vim's macro language, called ``<> notation" (you can find the description by doing ``:h key_notation"). All macros should be entered as is in your .vimrc file or on Vim's command line. You should even be able to cut and paste the information. This is a short description of the language: * Any printable characters are typed directly, except backslash and '<'. * A backslash is represented with ``\\", double backslash. * A real '<' is represented with ``\<". * ``<key>" means the special key typed. A few examples: <Esc> Escape key <C-G> CTRL-G <Up> cursor up key <C-LeftMouse> Control- left mouse click <S-F11> Shifted function key 11 <M-a> Meta- a ('a' with bit 8 set) <M-A> Meta- A ('A' with bit 8 set) <t_kd> "kd" termcap entry (cursor down key) If you want to use this notation in Vim, you have to remove the 'B' flag from 'cpoptions' and make sure the '<' flag is excluded. The default values for cpoptions should work fine. :set cpo=ceFs For mapping, abbreviation and menu commands you can then copy-paste the examples and use them directly. Or type them literally, including the '<' and '>' characters. This does NOT work for other commands, like ``:set" and ``:autocmd"! Other tips, not appearing in this FAQ, can be found by doing ``:h tips". 5.1 How do I use the help files? Help can be found for all functions of Vim. In order to use it, use ``:h". This will bring you to the main help page. On that first page, you will find explanations on how to move around. Basically, you move around in the help pages the same way you would in a read-only document. You can jump to specific subjects by using tags. This can be done in two ways: * Use the ``<Ctrl-]>" command while standing on the name of a command or option. This only works when the tag is a keyword. ``<Ctrl-LeftMouse>" and ``g<LeftMouse>" work just like ``<Ctrl-]>". * use the ``:ta subject" command. This works with all characters. Use ``<Ctrl-T>" to jump back to previous positions in the help files. Use ``:q" to close the help window. If you want to jump to a specific subject on the help pages, use ``:h {subject}". If you don't know what to look for, try ``:h index" to get a list of all available subjects. Use the standard search keys to locate the information you want. By version 6 or 7, we should have a search engine available.

;-) 5.2 Why is a backup file written even if I set nobackup? In order to keep Vim from writing a backup, you must also do ``:set nowritebackup''. 5.3 How can I keep Vim from beeping all the time? Well, you can use ``set noerrorbells'' but it does not turn off the bell for all cases. It only makes a difference for error messages; the bell will be always be used for a lot of errors without a message (e.g., hitting <Esc> in Normal mode). If you want Vim to stop beeping then all you need is ``:set vb'' which tries to do a screen flash rather than an audible beep. Some terminals can't do screen flashes, but if yours does and you don't want it to flash or beep then use ``:set vb t_vb=''. 5.4 How do I map the Tab key? :map <Tab> right-hand-side 5.5 How do I map the Esc key? On some keyboards the escape key is not placed very well. It's either part of the numeric block or it can be a small button (as with some Macintosh keyboards). But do not despair - make your own escape key! You can map one of the commands keys you do not need to Esc. Example: map CTRL-O to ESC: :map <C-O> <Esc> 5.6 How do I tell Vim where the helpfile is? To tell Vim where to find the help file, ``:set helpfile'' to the correct value, i.e., including the full path. As with most options you can abbreviate ``helpfile'' to ``hf''. On the other hand, if your VIM environment variable points to the directory containing the help file, vim will find the help file with no need to ``:set hf=''. 5.7 How do I get back to the exact position within a line I have marked with 'a'? Use ```a'' (that's a backtick!). If the backtick is awkwardly placed on your keyboard, the following maping may be useful to you: map ' ` 5.8 How do I read in the output of a command? Use ``:r!command''. 5.9 Why can't I abbreviate ``'xy''? The abbreviation consists of a non-id character followed by two id characters, which does not satisfy either category of a ``full-id''. However, ``_ps'' and ``'p'' will work. 5.10 How do I jump to the beginning/end of the highlighted text? The command 'o' will jump to the beginning/end of the highlighted text. 5.11 Why does completion of ``:set n'' not show negated settings, e.g., ``noautoindent''? Completion of ``:set no'' seems to work. The thing is that the ``no'' is not actually part of the option's name; the name comes after that. So after ``no'', Vim knows to complete any boolean setting name (starts the completion just after the ``no'', which is not part of the name). After ``n'', Vim will complete all setting names starting with ``n''. It would be a bummer if you wanted to complete ``number'', but had to wade through all the boolean option names with ``no'' prepended too. 5.12 Is there a command to remove any or all digraphs? No. The digraphs table is defined at compile time. You can only add new ones. Adding a command to remove digraphs is on the todo list. 5.13 How do I use a spell checker with Vim? You can call a non-interactive spell checker from Vim without a problem. A function to look up a word is included with the command ``K''. So what you do is get such a spell checker, then you issue the command ``:set keywordprg=ispell'', and then hit ``K'' on the word you want to look up, i.e., check. If you need to give options to your spell checker command, escape all spaces with a backslash. If you need to use an interactive spell checker and are working with Unix, you can try this approach proposed by Ives Aerts <ives@sonycom.com>: noremap ;ispell :%! ispell-filter<CR><C-L> where ispell-filter is the following script: #!/bin/sh # # This little script can be used to run ispell on stdin, returning the result # through stdout. # It can be used from VI like this (or map it to a key sequence): # :%! ~/bin/ispell-filter # cat > /tmp/tmp.$$ ispell $* /tmp/tmp.$$ < /dev/tty > /dev/tty cat /tmp/tmp.$$ rm -f /tmp/tmp.$$ or this macro proposed by Dr. Charles E. Campbell Jr. <cec@gryphon.gsfc.nasa.gov>: map #fi :w<CR>:!ispell %<CR>:e %<CR> 5.14 Can I copy the character above the cursor to the current cursor position? In Insert mode, you can copy the character above the cursor to the current cursor position by typing <Ctrl-Y>. The same can be done with the characters below the cursor by using <Ctrl-E>. This is neat when you need to

duplicate partial lines without going through the process of yanking a line and deleting the unwanted part. 5.15 How do I remove empty lines? To remove all empty lines do ``:g/^$/d". ``:g/[<Tab>]*$/d" deletes lines that aren't blank but look it, too. 5.16 How do I reduce a range of empty lines into one line only? You can try ``:v/./,/./-1join". Note that this will give an error message if the empty lines are at the end of the file. To correct this, use the following mapping instead: map _b GoZ<Esc>:g/^[<Tab>]*$/,/[^ <Tab>]/-j<CR>Gdd 5.17 How can I paste large amounts of text between two running sessions of Vim? If you are using the GUI version of Vim, with the Motif or Athena interface, you can actually cut and paste between the two, and text will be copied exactly; that is, tabs will not change into spaces, and if you copy a rectangular block, then that is what you will paste too! Otherwise, in a terminal Vim, use these mappings: " _Y: Yank the highlighted block of text (or a single line) to a tmp file. " _P: Put the text yanked with _Y (possibly in another invocation of Vim). " nmap _Y :.w! ~/.vi_tmp<CR> vmap _Y :w! ~/.vi_tmp<CR> nmap _P :r ~/.vi_tmp<CR> Now you just highlight the area you want to copy with ``V" etc, and yank it with ``_Y". Then you can paste it in another Vim with ``_P". 5.18 Can I use compressed versions of the help files? For those that are really short on disk space, you can compress the help files and still be able to view them with Vim. This example assumes you have gzip on your system. You do have it, don't you? :-) If not, you will need to adapt it to work with a different compression utility. 1. Compress all the help files: ``gzip doc/*.txt". 2. Edit doc/vim_tags (doc/tags in 5.0) and do :%s=\.txt<Tab>/=.txt.gz<Tab>/= 3. Add these lines to your .vimrc: set helpfile=<dirname>/vim_help.txt.gz autocmd BufReadPost *.txt.gz set bin | '[,']!gunzip autocmd BufReadPost *.txt.gz set nobin set cmdheight=2 Where ``dirname" is the directory where the help files are. If you already have included autocommands to edit ``.gz" files you should omit the two ``autocmd" lines. Note that you must ``set nocompatible" for this to work in Vim 5.0. 5.19 How come I can't set option ``xxxx"? Some options are compiled into Vim. If you want to enable these options, then you must modify the feature.h file. You can see what compile-time options are available by looking at the version number (:ver). It will give you something like this: Compiled with (+) or without (-): +autocmd +builtin_terms +cindent -compatible +digraphs -emacs_tags +fork() -GUI +insert_expand -langmap +lispindent -rightleft +smartindent -terminfo +viminfo +writebackup +X11 As the example shows, all options compiled into the Vim binary are preceded by a '+'. All options that are not compiled in are preceded by a '-'. All options that do not appear in this list do not need to be compiled in. 5.20 How do I format a block of C code? To format C code, use = instead of Q. Try ``:h C_indenting" to get more information on controlling the way your code is formatted. 5.21 How do I put a command onto the command history without executing it? You need only end the input with <ESC> (not <Ctrl-V><Esc>). 5.22 Why do I hear a beep (why does my window flash) about 1 second after I hit the Escape key? This is normal behavior. If your window flashes, then you've got the visual bell on. Otherwise, you should hear a beep. Vim (and most, if not all, other implementations of Vi) needs a timeout to tell the difference between a simple escape and, say, a cursor key sequence. When you press a key in normal mode (and even in insert mode) and that key is the beginning of a mapping, Vim waits a certain amount of time to see if the rest of the mapping sequence follows. If the mapping sequence is completed before a given timeout period, the mapping for that sequence of keys is applied. If you interrupt the mapping, the normal actions associated with the keys are executed. For example, if you have a mapping defined as ``:imap vvv Vim is great!!" and you type ``vvv" quickly, the ``Vim is great!!" will be inserted into your text. But if you type ``vv v" then that is

what will put into your text. This is also true if you type ``vvv'' too slowly where ``too slowly'' is longer than the value for the timeout option. Setting the timeout option to a larger value can help alleviate problems that appear when using function keys over a slow line. Do ``:h timeout'' for more information on using this option and its cohorts. 5.23 How do I make the 'c' and 's' commands display a '$' instead of deleting the characters I'm changing? Add ``:set cpoptions=ces$'' to your .vimrc. Vi-compatible behavior can be controlled like this. Do ``:help cpoptions'' for more information. 5.24 How do I add a line before each line with ``pattern'' in it? 1. Yank the line using Y 2. Insert the line with ``:g/pattern/put!'' 5.25 How can I delete the newline which is followed by a given character, such as ``|''? Look at this problem this way: If there is a newline before the character then the character is the first character of the next line, i.e., it follows the start-of-line meta character (^). So the command to use is to look globally for all lines starting with the given character and the command to use on these lines is ``go back one line'' and ``join'' which will remove the next newline. The resulting command is: :g/^|/-1join 5.26 How do I use the join command within a range of lines that ends with ``*'' and begins with the previous ``|''? :?^|?,/*$/join 5.27 How do I map/unmap an abbreviation or a map!ed key sequence? In either case, ``:set paste'' will do the trick. Use ``:set nopaste'' to return to normal. For map!ed key sequences, you have a these tricks also; * After typing the first key of the key sequence, pause about 1 second before typing the rest of the sequence. * Type <Ctrl-V> before typing the first key in the key sequence. Normally, you shouldn't need these tricks. When ``:unmap'' finds an argument that is not a ``from'' part, it looks for a matching ``to'' part and unmaps that one. 5.28 When I use my arrow keys, Vim changes modes, inserts weird characters in my document but doesn't move the cursor properly. What's going on? There are a couple of things that could be going on: either you are using Vim over a slow connection or Vim doesn't understand the key sequence that your keyboard is generating. If you are working over a slow connection (such as a 2400 bps modem), you can try to set timeout or ttimeout. These options, combined with timeoutlen and ttimeoutlen, may fix the problem. Do ``:h timeout'' and ``:h timeoutlen'' for a better description on how to use them. The preceding procedure will not work correctly if your terminal sends key codes that Vim does not understand. In this situation, your best option is to map your key sequence to a matching cursor movement command and save these mappings in a file. You can then ``:source'' the file whenever you work from that terminal. 5.29 How do I ``auto-outdent'' some lines containing certain keywords (i.e.have auto-indenting but towards the left margin instead of the right margin)? For example (make sure you adapt the example to your personal needs): for k=1:10, stuff .. end; ^^^ This word marks the end of a loop and I'd like it to be automatically outdented to the same column as the word for. The following macro will do this (from Raul Segura Acevedo <raul@turing.iquimica.unam.mx>): :iab end; <C-D>end; This abbreviation takes effect when you type a non-keyword character after it (``;'', ``!'', space, tab, <Esc>, <CR>, etc). You can load this macro automatically using the following autocommands: au! BufEnter *.f,*.ff iab end; <C-D>end; au! BufLeave *.f,*.ff iunab end; 5.30 Is there a repository for Vim macros? As a matter of fact, there is. You can find the latest versions of contributed macros at <URL:http://www.grafnetix.com/~laurent/vim/macros.html>. At the time this version of the FAQ was updated, the following macros were available: * A file browser * Some macros to edit HTML code * Macros to deal with numbered lists * A Tic Tac Toe game 5.31 If I have a mapping/abbreviation whose ending is the beginning of another mapping/abbreviation, how do I keep the first from expanding into the second one? Instead of using :map lhs rhs, use :noremap lhs rhs. For abbreviations, use noreabbrev lhs rhs. The ``nore''

prefix prevents the mapping or abbreviation from being expanded again. You must be using version 4.5 or greater. 5.32 Modelines are cool but Vim's modeline support is too restrictive. How can I improve it? >From Benjamin Elijah Griffin <eli@NetUSA.Net>: Vim modelines are crippled to reduce the security hazards of modelines. For much closer to vi modeline behavior (with the major risks involved) you can use Benjamin Elijah Griffin's modeline autocmd system. You can find it at <URL:http://www.netusa.net/~eli/src/modeline.html>. 5.33 How can I use different .vimrc settings for different types of files? There are a few ways to do this. All of them involve the use of autocommands. Try ``:h autocommand" for more information. * You can define autocommands which do everything for you in your .vimrc file: au!BufRead *.c,*.cc,*.h set fo=croq cin noic au!BufEnter *.c,*.cc,*.h ab #i #include au BufEnter *.c,*.cc,*.h ab #d #define au BufEnter *.c,*.cc,*.h map _hello_world :"Hello World"<CR> au BufEnter *.c,*.cc,*.h normal _hello_world au!BufLeave *.c,*.cc,*.h unab #i | unab #d au BufLeave *.c,*.cc,*.h unmap _hello_world The ``!" is used just in the first autocommand for each event (BufEnter, BufLeave, etc.). * Or you can define one autocommand for each file type which loads a specific .vimrc file containing the settings you require. autocmd BufNewFile *.tex read ~/.vim/texmodel autocmd BufEnter *.tex source ~/.vim/texrc autocmd BufLeave *.tex source ~/.vim/notexrc Remember that all settings that you modify in a BufEnter or BufNewFile event may need to be unset with a corresponding BufLeave event. By using a first file pattern with a value of ``*", you can define default settings for all your files. The ``mapclear" and ``abclear" commands can clear all your mappings or abbreviations if needed. 5.34 How can I search for tags when running Vim over a telnet session? Darren Hiebert <darren@sirsi.com> says: Ctrl-] is the the default telnet escape key. Most version of telnet allow changing or disabling the default escape key. See the man page. If possible, try to use ``rsh" instead of ``telnet". It will avoid this problem. 5.35 [NEW] I get errors when trying to put scripts in my mappings. Why? This is a tricky one which deals with the way bars are treated in mappings. For example, if you have something like: nmap <f5> :if has("syntax_items") | syntax off | else | syntax on | endif Vim treats the line as: nmap <f5> :if has("syntax_items") and then tries syntax off (which is ok), then tries else (which now has no matching if). This parsing occurs because |'s are used as command separators in Vim. Since those |'s are really part of the command, they should be replaced by <Bar> in your mapping, to yied something like (on one line only): nmap <f5> :if has("syntax_items") <Bar> syntax off <Bar> else <Bar> syntax on <Bar> endif<CR> Thank you to <Dr. Charles Campbell >cec@gryphon.gsfc.nasa.gov far the explanation. 5.36 [NEW] When I try to remove mappings in my autocommands, Vim says my mappings don't exists. What's going on? The explanation for this phenomenon is similar to that of the previous question. Basically, when unmapping your commands, you must not leave any spaces after the |'s separating the unmap keyword. For example, using the following autocommand: au BufRead *.c map <C-N>:cn<CR> | map <C-P>:cp<CR> This will fail to unmap <C-P>: au BufLeave *.c unmap <C-N> | unmap <C-P> But this will succeed: au BufLeave *.c unmap <C-N> |unmap <C-P> 6 DOS-, WINDOWS-, WIN32-SPECIFIC QUESTIONS 6.1 Why does the Win32 version of Vim update the screen so slowly on Windows 95? The support for Win32 console mode applications is very buggy in Win95. For some unknown reason, the screen updates very slowly when Vim is run at one of the standard resolutions (80x25, 80x43, or 80x50) and the 16-bit DOS version updates the screen much more quickly than the Win32 version. However, if the screen is set to some other resolution, such as by ``:set columns=100" or ``:set lines=40", screen updating becomes about as fast as it is with

the 16-bit version. Changing the screen resolution makes updates faster, but it brings problems with it of its own. External commands (e.g., ``:!dir'') can cause Vim to freeze when the screen is set to a non-standard resolution, particularly when columns is not equal to 80. It is not possible for Vim to reliably set the screen resolution back to the value it had upon startup before running external commands, so if you change the number of lines or columns, be very, very careful. In fact, Vim will not allow you to execute external commands when columns is not equal to 80, because it is so likely to freeze up afterwards. The maintainer of the Win32 port, George V. Reilly, says he's almost done with the GUI version of Vim. When it is completed, it should fix all these problems. In his own words: My position at this point is that the console mode APIs are irredeemably broken on Windows 95 and that I'm no longer interested in trying to come up with workarounds and hacks when my limited time is better spent trying to get an alpha of the Windows GUI version out the door. I strongly urge you to use one of the three standard resolutions (80x25, 80x43, or 80x50) and wait for the GUI version. Or upgrade to NT, where the console mode stuff works. Sorry. None of the above applies on Windows NT. Screen updates are fast, no matter how many lines or columns the window is set to, and external commands will not cause Vim to freeze. 6.2 So if the Win32 version updates the screen so slowly on Windows 95 and the 16-bit DOS version updates the screen quickly, why would I want to run the Win32 version? Firstly, the Win32 version isn't that slow, especially when the screen is set to some non-standard number of lines or columns. Secondly, the 16-bit DOS version has some severe limitations: it can't edit big files and it doesn't know about long filenames. The Win32 version doesn't have these limitations and it's faster overall (the same is true for the 32-bit DJGPP DOS version of Vim). The Win32 version is smarter about handling the screen, the mouse, and the keyboard than the DJGPP version is. 6.3 And what about the 16-bit DOS version versus the Win32 version on NT? There are no good reasons to run the 16-bit DOS version on NT. The Win32 version updates the screen just as fast as the 16-bit version when running on NT. All of the above disadvantages apply. Finally, 16-bit DOS applications can take a long time to start up and will run more slowly. On non-Intel NT platforms, the DOS version is almost unusably slow, because it runs on top of an 80x86 emulator. 6.4 Why can't I paste into Vim when running Windows 95? In the properties dialog box for the MS-DOS window, go to ``MS-DOS Prompt/Misc/Fast pasting'' and make sure that it is not checked. You should also do ``:set paste'' in VIM to avoid unexpected effects. See ``:h paste''. Also, in the Properties dialog's ``Misc'' tab, you want to make sure that ``Mouse Exclusive Mode'' is not checked. If you want to use the mouse in the Vim way, also make sure ``Mouse Quick Edit'' is not checked. (You can still cut and paste with the mouse by using the toolbar buttons.) 6.5 How do I type dead keys on Windows 95? (A dead key is an accent key, such as acute, grave, or umlaut, that doesn't produce a character by itself, but when followed by another key, produces an accented character, such as a-acute (á), e-grave (è), u-umlaut (ü), n-tilde (ñ), and so on. Very useful for most European languages. English-language keyboard layouts don't use dead keys, as far as we know.) You don't. The console mode input routines simply do not work correctly in Windows 95, and we have not been able to work around them. In the words of a senior developer at Microsoft: Win95 console support has always been and will always be flaky. The flakiness is unavoidable because we are stuck between the world of MS-DOS keyboard TSRs like KEYB (which wants to cook the data; important for international) and the world of Win32. So keys that don't "exist" in MS-DOS land (like dead keys) have a very tenuous existence in Win32 console land. Keys that act differently between

MS-DOS land and Win32 console land (like capslock) will act flaky. Don't even mention the problems with multiple language keyboard layouts... You may be able to fashion some sort of workaround with the digraphs mechanism. Alternatively, you can try one of the DOS versions, where deadkeys do work. 6.6 How do I type dead keys on Windows NT? Dead keys work on NT. Just type them as you would in any other application. 6.7 When will a real GUI version of Vim (gvim) for Win32 with scrollbars, menus, pasting from the clipboard, and so on become available? Work has begun on a GUI version of Vim for Win32. Apart from the features you might expect in gvim, it is expected that this version will also be able to handle dead keys correctly and that the problems with external commands will be a thing of the past. 6.8 On a Win32 machine, I'm using Vim to edit a symbolically linked file on a Unix NFS file server. When I write the file, Vim does not "write through" the symlink. Instead, it deletes the symbolic link and creates a new file in its place. Why? On Unix, Vim is prepared for links (symbolic or hard). A backup copy of the original file is made and then the original file is overwritten. This assures that all properties of the file remain the same. On non-Unix systems, the original file is renamed and a new file is written. Only the protection bits are set like the original file. However, this doesn't work properly when working on an NFS-mounted file system where links and other things exist. The only way to fix this in the current version is not making a backup file, by ``:set nobackup nowritebackup". 6.9 How do I copy text from Windows applications to the DOS version of Vim? >From John Velman, <velman@igate1.hac.com>: 1. To get Vim to run in a window, instead of full screen, press <Alt+Enter>. This toggles back and forth between full screen and a DOS window. 2. Issue the command ``:set paste". 3. To paste something *into* Vim, put Vim in insert mode. 4. Put the text you want to paste on the windows clipboard. 5. Click the control box in the upper left of the Vim window. (This looks like a big minus sign). If you don't want to use the mouse, you can get this with <Alt+Spacebar>. 6. On the resulting dropdown menu choose 'Edit'. 7. On the child dropdown menu choose 'Paste' 8. Issue the command ``:set nopaste". To copy something from the Vim window to the clipboard, 1. Select the control box to get the control drop down menu. 2. Select 'Edit'. 3. Select 'Mark'. 4. Using either the the keys or the mouse, select the part of the Vim window that you want to copy. To use the keys, use the arrow keys, and hold down shift to extend the selection. 5. When you've completed your selection, press ``<Enter>." The selection is now in the Windows clipboard. By the way, this can be any rectangular selection, for example columns 4-25 in rows 7-10. It can include anything in the Vim window: the output of a ``:!dir", for example. >From Stan Brown <stbrown@nacs.net>: In Windows 95, you can use a simpler procedure, which works even when you're using the mouse in the Vim way inside the Vim window (see question 6.4). To paste into Vim, put Vim in insert mode and click the Paste button on the Vim window's toolbar. (Depending on your setup, you may want to use ``:set paste" before and ``:set nopaste" after pasting.) To copy from Vim to the clipboard, click the Mark button (the square) on the toolbar, highlight the desired text with the mouse, and click the Copy button. 6.10 Why does my Caps Lock affect all the keys characters for all the keys instead of just the letters? It's actually caused by Windows 95. When CapsLock is down, it acts just as if Shift were being held down: everything gets shifted, not just the alphabetical keys. 6.11 How do I change Vim's window size in Windows? If ``:set lines=##" doesn't work for you, then you should use the ``:mode" command. Look up ``:h :mode" for more info. 7 UNIX-SPECIFIC QUESTIONS 7.1 How do I turn off the message ``Thanks for flying Vim" on Unix stations? When using Vim in an xterm it renames the title of that window to ``Thanks for flying Vim" on exit. Use ``:set notitle" to stop

this behavior. 7.2 How do I prevent <Ctrl-Z> from suspending Vim? Map <Ctrl-Z> to prevent the suspending. Here are some suggestions: 1. Make <Ctrl-Z> do nothing: map <C-Z> <C-V><C-V> 2. Make <Ctrl-Z> start a shell: :map <C-Z> :shell<CR> 3. Make <Ctrl-Z> give an error message: :map <C-Z> :"suspending disabled<CR> For the last example, the double quote is necessary in order to keep the message on the status line. 7.3 How can I make Vim faster on a Unix station? The GUI support in Vim 4.0 can slow down the startup time noticeably. Until Vim supports dynamic loading, you can speed up the startup time by compiling two different versions of Vim: one with the GUI and one without the GUI and install both. Make sure you remove the link from $bindir/gvim to $bindir/vim when installing the GUI version, though. If screen updating is your problem, you can run Vim in screen. screen is an ascii terminal multiplexer. The latest version can be found at <URL:ftp://ftp.uni-erlangen.de:/pub/utilities/screen>. 7.4 In Unix, how do I make Vim more colorful? You can change some termcap values to send to the screen the proper codes to change some colors (providing your terminal supports color). Here are some examples of how to do this if you do ``:h unix" but it seems as though they don't work all that well. But they help to understand what has to be done: :set t_me=<Esc>[0;1;36m " normal mode (undoes t_mr and t_md) :set t_mr=<Esc>[0;1;33;44m " reverse (invert) mode :set t_md=<Esc>[1;33;41m " bold mode :set t_se=<Esc>[1;36;40m " standout end :set t_so=<Esc>[1;32;45m " standout mode :set t_ue=<Esc>[0;1;36m " underline end :set t_us=<Esc>[1;32m " underline mode start Quoting Tony Nugent: You can do some interesting things by putting ansi colour sequences into those capabilities. What's given here are just examples, and some combinations don't work very well. You need to discover for yourself a configuration that works. For example, end-modes need to switch everything back to normal as well as turn things off. Just load ~/.vimrc, play around with the values, save it and then see what it looks like by sourcing it (:so ~/.vimrc). Don't forget to do things like ``:/<Ctrl-D>" and ``:map" to see all the different effects. And don't forget about the ``:set highlight=" string to fine-tune how the different capabilities are used. BTW, something like this also works for DOS and Win32 Vims! So it should also work well with windows or any ansi- and vt100- compatible terminal mode capable of displaying colours (which includes all colour pc's). It doesn't work so well in plain old xterm's (YMMV - your milage may vary). You can find a list of terminal codes here:
<URL:http://www.cs.utk.edu/~shuford/terminal_index.html>
<URL:ftp://cs.utk.edu/pub/shuford/terminal/>
<URL:ftp://gatekeeper.dec.com/pub/DEC/termcaps> -- Laurent Duperval, Vim FAQ Maintainer <laurent@Grafnetix.COM> Vim Pages: http://www.vim.org/ Vim Mailing List: vim@vim.org Vim FAQ: http://www.vim.org/faq/ Vim FAQ Home: http://www.grafnetix.com/~laurent/vim/faq.html

Ë²Ä´ËÇVim?
Ò»¸ö6K³¤µÄËµÃ÷

Vim ("Vi IMproved") ÊÇÒ»¸ö vi µÄ"¿ËÂ¡", Ò²¾ÍÊÇËµ, ËüÊÇÒ»¸öÀàËÆÎÄ±¾±à¼-Æ÷ vi µÄ³ÌÐò.

Vim ÔÚÈÎºÎÖÕ¶ËµÄÎÄ±¾Ä£Ê½ÏÂ¾ùÄÜ¹¤×÷, Í¬Ê±ËüÒ²ÓÐÒ»¸öÍ¼ÐÎ½éÃæ, °üÀ¨²Ëµ¥²¢Ö§³ÖÊó±ê.

»ñµÃVim:
ÓëviÏà±È, Vim ´æÔÚÓÐí¶àÆ½Ì¨ÉÏ, ²¢ÇÒÔöÌíÁËÐí¶à¹¦ÄÜ. (Ïê¼û http://www.vim.org/doc/vi.diff.txt). Vim ºÍ¾ø´ó¶àÊýµÄViÃüÁî¼æÈÝ - Vi µÄBUG³ýÍâ:-)

²Ù×÷ÏµÍ³:
ÔÚÐí¶à²Ù×÷ÏµÍ³ÉÏ¶¿ÉÒÔµÃµ½Vim: AmigaOS, Atari MiNT, BeOS, DOS, MacOS, NextStep, OS/2, OSF, RiscOS, SGI, UNIX, VMS, WIn16 + Win32 (Windows95/98/00/NT) - ÓÈÆäÊÇ FreeBSD and Linux. :-)

°æÈ¨ÐÅÏ¢:
°æÈ¨¹éVimµÄÖ÷Òª×÷ÕßºÍÎ¬»¤ÕßBram MoolenaarËùÓÐ<bram@vim.org>. Vim ÊÇ"´ÈÉÆÈí¼þ". ÒâË¼ÊÇ¹ÄÀøÄã¾èÖúÎÚ¸É´ïµÄ¹Â¶ù(Ïê¼û ":help uganda").

Ô´´úÂë:
Vim ÊÇ¿ª·ÅÔ´´úÂëÈí¼þ. »¶Ó-´ó¼Ò°ïÃ¦Ä½ø.

=== ÌØÉ«:

³õÑ§ÕßµÄ±à¼-Æ÷ - ÓÃ»§ÓÑºÃ:
Ïà±ÈVi, Vim¶Ô³õÑ§ÕßÀ´µÄÈÝÒ×¶ÀÉË. ÕâÒª¹é¹¦ÓÚÆäÏê¾¡µÄÔÚÏß°ïÖú, "undo" ºÍ "redo" ÃüÁî (ÄãÓÀÔ¶²»ÓÃµ£ÐÄ·¸´í - Ö»Òª»áÓÃundo+redo!), ÒÔ¼°¶ÔÊó±êµÄÖ§³ÖºÍ¿ÉÒÔÅäÖÃµÄÍ¼±ê¼°²Ëµ¥(GUI).

×Ö·û±àÂëºÍÖÕ¶Ë:
VimÖ§³Öiso-latin1×Ö·û¼¯ºÍtermcap(Vi²»ÄÜÕýÈ·µØÖ§³Ö)

×Ö·ûºÍÓïÑÔ:
Vim Ö§³Ö´ÓÓÒÏò×ó±à¼-(ÀýÈç°¢À-²®ÎÄ, Farsi, Ï£²®À´ÎÄ) ºÍ¶à×Ö½ÚÎÄ×Ö, ±ÈÈçÄÇÐ©Ê¹ÓÃÍ¼ÐÎ×Ö·ûµÄÓïÑÔ,Æä×Ö·ûÓÃ¶à¸ö×Ö½Ú±íÊ¾. ±ÈÈçÖÐÎÄ, ÈÕÎÄ, º«ÎÄ, (´Ó¼¼ÊõÇ¶ÈÉÏ²², VimÖ§³ÖUTF-8ºÍUnicode.)

ÎÄ±¾¸ñÊ½»¯ºÍVisualÄ£Ê½:
ÔÚVimÖÐ, Äã¿ÉÒÔ"¿ÉÊÓµØ"Ñ¡È¡ÎÄ±¾(¸ßÁÁÏÔÊ¾). È»ºó¶ÔËùÑ¡ÎÄ±¾½øÐÐ²Ù×÷, ÀýÈç¿½±´, É¾³ý, Ìæ»», ×óÓÒÆ¶¯, ´óÐ¡Ð´±ä»», »ò±£³ÖÇ¶Èë¸ñÊ½µÄ¸ñÊ½»¯. Vim ÔÊÐí¶Ô³¤·½ÐÎµÄÎÄ±¾é½øÐÐÑ¡ÔñºÍ²Ù×÷.

ÃüÁî²¹È«:
Vim °üº¬²¹È«ÄãµÄËäÈëµÄÃüÁî - ÎÞÂÛÄãÊäÈëµÄÊÇÃüÁî, ÎÄ¼þÃû, »¹ÊÇµ¥´Ê.

×Ô¶¯ÃüÁî:
Vim »¹ÓÐÓÃÓÚ×Ô¶¯Ö´ÐÐÖ¸ÁîµÄ"×Ô¶¯"ÃüÁî. (ÀýÈç ×Ô¶¯½âÑ¹ËõÎÄ¼þ.)

(·¢Ò»¸öÒôµÄ)¶þºÏ×ÖÄ¸(Diagraph)ÊäÈë:
ÔÚVimÖÐ, Äã¿ÉÒÔÓÃÁ½¸ö×Ö·ûµÄ×éºÍÊäÈëÒ»¸öÌØÊâ×Ö·û(ÀýÈç`ºÍeµÄ×éºÏÉú³É¨¨-²¢ÔÊÐíÄã¶¨Òåæäë û×éºÏ.

ÎÄ¼þ¸ñÊ½Ê¶±ðºÍ×ª»»:
Vim×Ô¶¯Ê¶±ðÎÄ¼þµÄÀàÐÍ(DOS, Mac, Unix)²¢ÔÊÐíÒÔ²»Í¬¸ñÊ½´æÅÌ - ²»ÓÃÔÚWindowsÉÏÔÙ×÷unix2dosÁË!

ÀúÊ·:

¶ÖÒÙÄüÀÎºÍ²éÖÒ²Ù×÷, Vim»¹ÒÐÀúË·»úÖÆ.
Ää¿ÉÒÔ°ÑÒÔÇ°Ö´ÐÐ¹ýµÄÃüÁî»ò²éÕÒÄ£Ê½µ÷³öÀ´½øÐÐ±à¼-.

ºêÂ¼ÖÆ:
VimÔÊÐí"Â¼ÖÆ"ÄãµÄ±à¼-²Ù×÷, ÒÔ±ã¶ÔÖØ¸´µÄÈÎÎñ½øÐÐ"ÖØ·Å".

ÄÚ´æÏÞÖÆ:
VimÖÐÐÐ³¤ºÍ»º³åÇø³¤¶ÈµÄÉÏÏÞ¶¼±È´«Í³Vi¸ß³öÐí¶à.

¶à»º³åÇøºÍÇÐ·Ö´°¿Ú:
VimÔÊÐíÍ¬Ê±±à¼-¶à¸ö»º³åÇø¶øÇÒÄã¿ÉÒÔ°ÑÆÄ»·Ö¸î³ÉÐí¶à×Ó´°¿Ú(Ë®Æ½ºÍÊúÖ±).
ÕâÑùÄã¾Í¿ÉÒÔ¿´µ½Ðí¶àÎÄ¼þ»ò¼¸öÎÄ¼þµÄÐí¶à²¿·Ö.

ÃüÁîµÄÊý×ÖÇ°×º:
Ïà±ÈVi, VimÔÊÐí¸ü¶àµÄÃüÁî½´øÊý×ÖÇ°×º(ÀýÈç "put"ÃüÁî).

ÔËÐÐÊ±ÎÄ¼þ(°ïÖúºÍÓï·¨ÎÄ¼þ):
Vim´øÓÐ70¸ö°ïÖúÎÄ¼þ. ËüÃÇ°üÀ¨ÁË±à¼-µÄ·½·ÃæÃæ;
ÆäÖÐÒ»Ð©ÎÄ¼þÊÇÒò²Ù×÷ÏµÍ³²»Í¬¶øÒìµÄ.

½Å±¾:
Vim °ü º¬Ò»¸öÄÚÖÃµÄ½Å±¾ÓïÑÔÒÔ±ãÓÚÀ¨³ä¹¦ÄÜ.

Æ«ÒÆ²éÑ°:
Vim ÔÊÐí²éÑ°ÃüÁî¸½´øÆ«ÒÆÁ¿, ÕâÑùÄã¾ÍÄÜ°Ñ¹â±êÖ±½Ó·ÅÔÚÕÒµ½µÄÎÄ±¾óÃæ.

¹ý³Ì(session)»Ö¸´:
VimÔÊÐí½«Ò»¸ö±à¼-¹ý³ÌµÄÏà¹ØÅÏ¢±£´æµ½Ò»¸öÎÄ¼þÖÐ("viminfo"). ÔÚÏÂ´Î±à¼-ÖÐ,
¸ÃÎÄ¼þÊ¹ÕâÐ©ÐÅÏ¢ØØÐÂÉúÐ§, ÀýÈç»º³åÇøÁÐ±í, ÎÄ¼þ±ê¼Ç, ¼Ä´æÆ÷, ÃüÁîºÍ²éÕÒÀúÊ·.

Tab¼üÀ©Õ¹:
Vim¿ÉÒÔÔÚÎÄ±¾ÖÐ½«TAB¼üÀ©Õ¹Îª¿Õ¸ñ (expandtab, :retab)

±êÇ©(Tag)ÏµÍ³:
VimÌá¹©ÔÚÎÄ¼þÖÐÓÃ±êÇ©¼ÓË÷Òý¼°Ðí¶à±êÇ©¶ÑÕ»ÃüÁî²éÕÒÎÄ±¾.

ÎÄ±¾¶ÔÏó:
Vim(Ïà±ÈVi)Öªµ½¸ü¶àµÄÎÄ±¾¶ÔÏó(¶ÎÂä, ¾ä×Ó, words ºÍ WORDs -
°üÀ¨ºÍ²»°üÀ¨ÖÚÎ§µÄ¿Õ¸ñ.) ²¢ÇÒÔÊÐí¶ÔÕâÐ©¶ÔÏóÖØ¶¨Òå.

Óï·¨ÉÏÉ«:
VimÓÃ¸÷ÖÖÑÕÉ«ÏÔÊ¾ÎÄ±¾ - °´ÕÕÆä"(±à³Ì)ÓïÑÔ". Äã¿ÉÒÔ¶¨ÒåÎÄ¼þµÄÓïÑÔ(Óï·¨).
Vim×Ô´ø200¶à¸öÎª¸÷ÖÖÓïÑÔÎÄ¼þÉÏÉ«µÄÓï·¨ÎÄ¼þ, ±à³ÌÓïÑÔ(Ada, C, C++, Eiffel,
Fortran, Haskell, Java, Lisp, Modula, Pascal, Prolog, Python, Scheme, Smalltalk,
SQL, Verilog, VisualBasic), ÊýÑ§³ÌÒò(Maple, Matlab, Mathematica, SAS),
±ê¼ÇÓïÑÔ(DocBook, HTML, LaTeX, PostScript, SGML-LinuxDoc, TeX, WML, XML),
³ÌÒòµÄÊä³ö½á¹û(diff, man), ÅäÖÃºÍ³õÊ¼¯ÎÄ¼þ(4DOS, Apache, autoconfig, BibTeX,
CSS, CVS, elm, IDL, LILO, pine, procmail, samba, slrn), shell
½Å±¾ºÍ³õÊ¼¯ÎÄ¼þ(sh, bash, csh, ksh, zsh), ½Å±¾ÓïÑÔ(awk, Perl, sed, yacc),
ÏµÍ³ÎÄ¼þ(printcap, .Xdefaults). µ±È»°üÀ¨Vim±¾ÉíµÄÏµÍ³ºÍ°ïÖúÎÄ¼þ.

ÌØÊâ´úÂë:
Vim °ü º¬ÓëPerl, Tcl ºÍ Python ¼¯³ÉµÄÑ¡ÏîÎ.
Vim ÔÚWindowsÏÂ¿ÉÒÔ×÷ÎªOLE×Ô¯»¯·þÎñÆ÷ÔËÐÐ.
Vim °²×°Ê±¿ÉÒÔ°ü º¬ÔÚX-window ÖÐ¹¤×÷µÄ´úÂë,
ÕâÑù¾Í¿ÉÒÔ¼ÓÈë¿ÉÅäÖÃ²Ëµ¥ºÍ¶ÔÊó±êµÄÖ§³Ö.
ÒÔ¼°ÆäËûÐí¶àÐí¶à¹¦Ð§.

VimµÄÖ÷Ò³ÔÚ:
http://www.vim.org/

Ò»¸ö¶ÔVimµÄÌØÉ«¸üÏê¾¡µÄÃèÊö¿ÉÒÔ´ÓÏÂÃæµÄÍøÒ³ÕÒµ½:
http://www.vim.org/why.html

×÷Õß: Sven Guckes guckes@vim.org
×î½ü¸üÐÂÊ±¼ä: 2000Äê10ÔÂ3ÈÕ 20:00:00 MET DST
ÒëÕß: Ê·Î°¹â Weiguang Shi wgshi@cs.ualberta.ca
×î½ü¸üÐÂÊ±¼ä: 2000Äê11ÔÂ9ÈÕ 16:21:17 PST

ï»¿ "Vim æ˜¯ä»€éº¼?"
6K é••çš„ç°¡å–®èªªæ˜Ž

Vim ("Vi IMproved") æ˜¯ vi çš„ä¸€å“•ï¼Œä¹Ÿå°±æ˜¯èªªï¼Œ
å®ƒæ˜¯ä¸€å€‹å’Œ vi é€™å€‹æ–‡å--ç·¨è¼¯å™¨ç›¸è¿‘çš„ç¨‹å¼•ã€‚

Vim å•¯ä»¥åœ¨å•„ç¨®ç‰ˆçš„æ–‡å--æ¨¡å¼•ä¸‹ä½œï¼Œ
è€Œä¸”å®ƒå¹¶Ÿå…·æœ‰å½¢å¢¿ç”¨è€…ä»‹é•¢ï¼Œï¼Œé•¸å–®ï½œ
ä»¥å•Šæ”¯æ•´é¼ ã€‚

æ˜“å•-å¾—æ€§:
Vim æœ‰è¨±å¤šå¹³èŠ°ä¸Šçš„ç‰ˆæœ¬ï¼Œè€Œä¸Ÿå¢žåŠ ä°†è¨±å¤š Vi
æ‰€æ²’æœ‰çš„åŠŸèƒ½ã€‚
ï¼ˆè©³è¦‹ http://www.vim.org/doc/vi.diff.txtï¼‰
Vim èˆ‡å¹¼ä¹Žå…¨éƒ¨çš„ Vi æŒ‡ä»¤æ˜¯ç›¸å®¹çš„ï¼Œä½†æ˜¯ bug ;-)

ä½œæ¥-ç³»çµ±:
Vim èƒ½åœ¨å•„ç¨®ä½œæ¥-ç³»çµ±ä¸Šåˆ·è¡ŒïŒåŒ…å•«: AmigaOS, Atari MiNT, BeOS, DOS,
MacOS, NextStep, OS/2, OSF, RiscOS, SGI, UNIX, VMS, Win16 + Win32
ï¼ˆWindows95/98/00/NTï¼‰ - è€Œä¸”Ÿåœ¨ç•¶æ‰€ä½¿ç”¨çš„ FreeBSD å’Œ Linux. :-)

ç‰ˆæ¬Š:
ç‰ˆæ¬Šç”±åŽŸå§‹çš„ä½œè€…ä¥å•Šä¸»è¦•çš„ç¶-è--è€…, Bram Moolenaar <bram@vim.org>
æ‰€æœ•œæ‰€ã€‚Vim
æ˜¯ã€Œæ…ˆå-„è»Ÿé«”ã€•ï¼Œ Œæœ›æ,¨æ••éŒ¢å¹«åŠ©çƒå¹²é•”çš„å-¤å…’å€‘
ï¼ˆè©³è¦‹ ":help uganda"ï¼‰ã€‚

åŽŸå§‹ç¢¼:
Vim æ˜¯é-‹æ”¾åŽŸå§‹ç¢¼è»Ÿé«”ï¼Œæè€Œä¸”æ-¡è¿Žä»»ä½•ä°°å¹«å¿™ç™¼å±•!

=== ç‰¹æ€§

åˆ•å-¸è€……çš„ç·¨è¼¯å™¨ - æ˜“æ-¼ä½¿ç”¨:
ç›¸è¼ƒæ-¼ Viï¼ŒVim
å°•åˆ•å-¸è€……ä¾†èªªæ›´åŠ å®¹æ˜“ï¼Œé™¤éŒ¯¤äº†æœ‰å®Œæ•´çš„ç·šä¸Šèªªæ˜Žä¹‹å¤–ï¼Œ
"undo"ï¼ˆé‸æ‚„åŽŸï¼‰å’Œ "redo"ï¼ˆé‡•å•šï¼‰æŒ‡ä»¤è®“ä½ æ°¸é• ä¸•ç”¨æ“"å¿ƒå‡ºéŒ¯ -
å•ªè¦•æœƒ"
undo+redoï¼•é,„æœ‰æ”¯æ•´é¼ ï¼ŒïŒä»¥å•Šå•¯è¨-å®šçš„åœ–ç¤ºå’Œé•¸å-®ï¼‰åœ¨GUIç’°å¢ƒä¸‹ã€‚

å--ç¢¼èˆ‡çµ,ç«¯æ©Ÿæ¨¡å¼•:
Vim æ”¯æ•´ iso-latin1 å--å…ƒé›†ä»¥å•Šå®Œæ•´çš„ termcap æ”¯æ•´ã€‚
ï¼ˆæœ€åŽŸå§‹çš„ Vi ä¸•èƒ½æ-£å¸¸å·¥ä½œã€‚ï¼‰

å--å…ƒèˆ‡žè¨€:
Vim
æ”¯æ•´å•³è‡³³å·¦ç·¨è¼¯æ¨¡å¼•ï¼ˆé˜¿æ‰¯æ–‡ã€•æ–‡å¢ƒå¯¦ä¾•å…¶ä¸ä¸€å€‹æ¨¡å¼•ï¼Œæ–‡å--å…ƒèˆ‡žè¨€ï¼ŒåŒ…å«çš„èªžè¨€ï¼Œæ-‡ã€•æ—¥
æ-‡å--é›†ï¼ŒäºŒå•³ä½¿ç”¨è¶…é•Žä¸€ä½•å…ƒèˆ‡žè¨€ï¼Œçš„æ-‡å--çš„èªžè¨€ï¼Œæ-‡ã€•æ—¥
æ-‡ã€•éŸ“æ-‡ã€,ï¼ˆå°±æŠ€è¡“é•¢è¨€ï¼Œ Vimæ”¯æ•´ UTF-8 ä»¥å•Š Unicode
çš„å--å…fã€,ï¼‰

æ ¼å¼•åŒ–æ-‡å--èˆ‡ãŒè¦•æ¨¡å¼•:
Vim
å•¯ä»¥è®“æ,¨ãŒè¦-è¦•åŒ–ã€•åœ°é•¸æ“‡æ-‡å--ï¼ˆä»¥é«˜äº®å°¦è¡¨ç¤ºï¼Œä¹Ÿå¯¦ä¾‹å¼•å…¥ä¸€å¾Œå†•å°•å…¶é²è¡ŒŒæ“•ä½œï¼Œ
ä¾‹å¦‚ï¼Œè¤‡è£½ã€•åˆªé™¤ã€•å•-ä»£ã€•å·¦å³3ï¼³ã€•æ”¹è®Šå--æ¯•å¤§å°•å¯¦ä¾•ç›®å‰•æ˜¯å¦•æ¿•ç›¸å…³çš„ã€‚æŽ¥ä¸‹ä¾†æ“•ä½œçš„å•-ä¸€å€‹æ¨¡å¼•ã€‚æ¯•å•-ä»¥å¾Œï¼Œä¹Ÿæ ¼å¼•åŒ–
æ-‡å--ã€,Vim ä¹Ÿå…•è¨±æ,¨é•¸å•-ä¸¦ä¸”å°•çŸ©å½¢çš„æ-‡å--å•€å¡Šä½œæ“•ä½œã€,

è£œå…¨æŒ‡ä»¤:
Vim
æœ‰æŒ‡ä»¤å•¯ä»¥è‡ªå‹•æ›¿æ,¨è£œå…¨æ,¨çš„è¼¸å…¥ï¼Œä¹Ÿå°±æ˜¯ä¸€èš-æ˜¯æŒ‡ä»¤ã€•æ°æ”¡æ¡ˆå••ç¨±ï¼Œæˆ-æ˜¯å-®å--ã€,

è‡ªå‹•æŒ‡ä»¤:
Vim
åŒ…å•«æœ‰å•¯ä»¥è‡ªå‹•å‹•åŸ·è¡Œçš„è‡ªå‹•æŒ‡ä»¤ï¼Œè‡ªå‹•å‹•å°‡æ”¡æ¡ˆã€•æ°æ”¡æ¡ˆç§£å£“ç¸®ï¼Œã€,

äºŒå•ˆå--æ¯•(diagraph)è¼¸å…¥:
Vim å…•è¨±æ,¨è¼¸å…¥æŸ•äº›ç”±å…©å€‹å--å…fæ‰€çµ„æˆ•çš„äºŒå•ˆå--æ¯•ï¼ˆä¾‹å¦‚ç”±ç¶ "
å’Œ a
æ‰€çµ„æˆ•çš„ éŠ¦^ã€,ä¸¦ä¸”ä¹Ÿå•¯ä»¥è®“æ,¨è‡ªè¡Œå®šç¾©å…¶ä¾–çš„çµ„å•ˆã€,

æª”æ¡ˆ ¼å¼•å•µæ¸¬è¯•ä¸‹ï¼š
Vim æœfè‡ªå‹•è¾¨è¯†æª”æ¡ˆçš„ ¼å¼•ï¼ˆDOS, Mac,
Unixï¼‰è€Œä¸Ÿè®“æ,¨è½‰å-~ç,ºå…¶ä»––
æ ¼å¼• - åœ¨ Windows ä¸Šä¸•å†•éœè¦• unix2dos äº†ï¼•

æ-·å•²:

Vim
å°•æ–¼åŸ·è¡Œé•Žçš„æŒ‡ä»¤æœ‰ã€Œæ–·å•²ã€•åŠŸèƒ½ï¼Œæ¨å•¯ä»¥å••å‡ºä¹‹å‰•è¼¸å…¥é•Žçš„æŒ‡
ä»¤ä¸¦åŠ ä»¥ä¿®æ”¹å†•æ¬¡ä½¿ç”¨ã€‚

å·¨é›†éŒ„è£½:
Vim
å•¯ä»¥è®“æ‚¨ã€ŒéŒ„ã€•ä¸‹æ‚¨çš„ç·¨è¼¯é•Žç¨‹ï¼Œä¸¦ä¸”å†•æ¬¡æ’æ”¾ï¼Œä»¥é€²è¡Œé‡•è¤‡æ€§é«˜çš„å·¥ä½œã€‚

è¨˜æ†¶é«”é™•åˆ¶:
æ¯”èµ·æœ€åŽŸå§‹çš„ Viï¼ŒVim
æœ‰è'—æ›´å¤§çš„æ¯•å•–é•·åº¦é™•åˆ¶ä»¥å•Šæ›«å˜å••€çš„å¤§å°•ã€‚

å¤šé‡•æš«å˜å•€ä»¥å•Šèž¢å¹•å ˆå‰²:
Vim
è®“æ¨å•¯ä»¥ç·¨è¼¯å¤šå€‹æš«å˜å•€ï¼Œè€Œä¸”æ¨ä¹Ÿå•¯ä»¥å°‡è¦–çª—å ˆå‰²ç‚ºå¤šå€‹å•è¦–çª—ï¼ˆå•¯
ä»¥æ˜¯æ°´å¹³æˆ–æ˜¯åž‚ç›´çš„ï¼‰ï¼Œä»¥ä¾¿å•Œæ™‚æª¢è¦–å¤šå€‹æª”æ¡ˆçš„ä¸•å•Œä½•ç½®ã€‚

æŒ‡ä»¤å‰•ç½®æ•¸å—:
æ¯”èµ·åŽŸå§‹çš„ Viï¼ŒVim
å•¯ä»¥åœ¨æ›´å¤šçš„æŒ‡ä»¤å‰•é•¢åŠ ä¸Šæ•¸å—ä¾†è¡¨ç¤ºåŸ·è¡Œæ¬¡æ•¸ï¼ˆä¾‹å¦‚ "put
æŒ‡ä»¤ï¼‰ã€‚

åŸ·è¡Œæ™‚æœŸæª”æ¡ˆï¼ˆèªªæ˜Žæª”æ¡ˆèˆ‡æ–‡æª”æ¡ˆï¼‰:
Vim
æœ‰ä¸ƒå••å€‹åŒ…å•«äº†å•„ç¨®å…§å®¹çš„èªªæ˜Žæª”ï¼›å…¶ä¸æœ‰äº›æ˜¯ç‰¹å®šé•‡å°•ç‰¹å®šçš„ä½œæ¥ç³»çµ±æˆ–æ˜¯å«çš„èªªæ˜Žã€‚

ä½¿ç”¨script:
Vim å…§å»ºä¸€å¥— script èªžè¨€ï¼Œæ–¹ä¾¿ä½¿ç”¨è€…ä½œå•„é …åŠŸèƒ½å»¶ä¼¸ã€‚

æ•œå°‹ç§»ä½•:
Vim
å…•è¨±æ•œå°‹æŒ‡ä»¤é„åŠ ä½•ç§»ï¼Œå¦‚æ–¤ä¸€äº›æ¨å°±å•¯ä»¥å°‡æ¸¸æ¨™æ”¾ç½®åœ¨æ‰¾åˆ°çš„æ–‡
å—å¾Œé•¢ã€‚

æ··ç¨‹å›žå¾©:
Vim
å…•è¨±æ¨å°‡ç·¨è¼¯é•Žç¨‹ä¸çš„è³‡è¨Šå˜å…¥æª”æ¡ˆä¸ï¼ˆ"viminfo"ï¼‰ï¼Œåœ¨æ¨ä¸‹æ¬¡ç·¨è¼¯çš„æ™‚å€™
å°±å•¯ä»¥ç›´æŽ¥ä½¿ç”¨ï¼Œä¾‹å¦‚æš«å˜å•€å…§å®¹ã€•æª”æ¡ˆæ¨™è¨˜ã€•æš«å˜å™¨ã€•å'½ä»¤æ·å•²ç‰ã€‚

å®šä½•å—å…ƒå±•é–‹:
Vim å•¯ä»¥æ›¿æ¨ç”¨ç©ºç™½ä¾†å±•é–‹æ–‡å—ä¸çš„å®šä½•å—å…ƒï¼ˆexpandtabï¼Œ
:retabï¼‰ã€‚

æ¨™ç±¤ç³»çµ±:
Vim
æ••ä¾›åœ¨æª”æ¡ˆä¸æ ¹æ“šç´¢å¼•å•Šã€Œæ¨™ç±¤ã€•ä¾†æ•œå°‹æ–‡å—ï¼Œæˆ–æ˜¯æ•–é…•è¨±å¤šæ¨™
ç±¤å †ç–ŠæŒ‡ä»¤ä¾†å°‹æ‰¾æ–‡å—ã€‚

æ–‡å—ç‰©ä»¶:
Vim
èƒ½å¤ è¾¨èªæ›´å¤šçš„æ–‡å—ç‰©ä»¶ï¼ˆæ®µè•½ã€•å•¥å•ã€•å–®å—ï¼Œä»¥å•Šç¸®å¯«å— -
åŒ…æ‹¬æˆ–æ˜¯
ä¸•åŒ…æ‹¬å'¨åœ•çš„ç©ºç™½ï¼‰ï¼Œä¸¦ä¸”ä¹Ÿå…•è¨±æ¨è‡ªç”±æ›´æ”¹å°•æ–¼é€™äº›ç‰©ä»¶çš„å®šç¾©ã€‚

èªžæ³•è®Šè‰²:
Vim å•¯ä»¥ä½¿ç”¨å½©è‰²ä¾†é¡¯ç¤ºæ–‡å— -
æ ¹æ“šå®ƒçš„èªžè¨€ï¼ˆæˆ–æ˜¯ç¨‹å¼èªžè¨€ï¼‰ã€‚
æ‚¨å•¯ä»¥è‡ªè¡Œå®šç¾©é€™äº›èªžè¨€ï¼ˆèªžæ³•ï¼‰æª”æ¡ˆã€‚

Vim
éš¨é™„æœ‰äºŒç™¾ç¨®ä»¥ä¸Šçš„èªžæ³•æª”æ¡ˆï¼Œèƒ½å¤ ç‚ºå•„ç¨®å¸¸è¦‹çš„ç¨‹å¼èªžè¨€æª”æ¡ˆï¼ˆä¾‹å¦‚ï¼š
ä¾‹å¦‚ï¼š Ada, C, C++, Eiffel, Fortran, Haskell, Java, Losp, Modula, Pascal,
Prolog, Python, Scheme, Smalltalk, SQL, Verilog, VisualBasicï¼‰ï¼Œæ•¸å¸ç¨‹å¼
æª”æ¡ˆï¼ˆMaple, Matlab, Mathematica, SASï¼‰ï¼Œæ¨™è¨˜èªžè¨€ï¼ˆDocBook, HTML,
LaTeX,
PostScript, SGML-LinuxDoc, TeX, WML, XMLï¼‰ï¼Œç¨‹å¼çš„è¼¸å‡ºï¼ˆdiff,
manï¼‰ï¼Œè»Ÿé«”
è¨å®šæª”ï¼ˆ4DOS, Apache, autoconfig, BibTeX, CSS, CVS, elm, IDL, LILO, pine,
procmail, samba, slrnï¼‰ï¼Œå'½ä»¤è™•ç†å™¨èªžè¨€è‡³è¨å®šæª”ï¼ˆsh, bash, csh,
ksh, zshï¼‰ï¼Œ
script èªžè¨€ï¼ˆawk, Perl, sed, yaccï¼‰ï¼Œç³»çµ±æª”æ¡ˆï¼ˆprintcap,
.Xdefaultsï¼‰ä»¥åŠ
ï¼ˆç•¶ç„¶ï¼‰Vim çš„è¨å®šæª”å'Œèªªæ˜Žæª”ã€‚

ç‰¹æ®Šç¨‹å¼•ç¢¼:
Vim å•¯ä»¥ç‰¹å^¥è^‡ Perl, Tcl, å'Œ Python æ•´å•^ã€,
Vim åœ¨ Windows ä¸‹å•¯ä»¥ä½œç‚º OLE automation ä¼ºæœ•å™¨ã€,
Vim å•¯ä»¥æ”¯æ•´ X-windowsï¼Œä»¥åŠ å…¥å•¯è‡ªè¨‚ç‚„é•¸å–®è^‡æ»'é¼ çš„æ”¯æ•´ã€,
ä»¥å•Šæ›´å¤šæ›´å¤šï¼•

Vim çš„ WWW ä¸»é •:
  http://www.vim.org/

å¦‚æžœæƒ³è¦•æœ‰æ›´è©³ç´°çš„ Vim åŠŸèƒ½å^—è¡¨ï¼Œè«‹è¦‹
  http://www.vim.org/why.html

è‹±æ–‡ç‰ˆä½œè€…: Sven Guckes guckes@vim.org
è‹±æ–‡ç‰ˆæ€å¾Œæ›´æ–°æ™‚é–“: Tue Oct 03 20:00:00 MET DST 2000
ä¸æ–‡ç‰ˆè¿»è¯‘è€…: Cecil Sheng zcecil@infor.org
ä¸æ–‡ç‰ˆæ€å¾Œæ›´æ–°æ™‚é–“: Wed Nov 15 08:00:00 CST 2000

"Wat is Vim?"
Een uitleg in zes kilobytes.

Vim ("Vi IMproved", "vi verbeterd") is een "vi kloon", ofwel een programma dat lijkt op de text editor "vi".

Vim werkt in textmode op iedere terminal, maar heeft ook een grafische user-interface, bijvoorbeeld menu's en muis-ondersteuning.

Beschikbaarheid:
Vim is voor vele platforms beschikbaar en heeft veel toegevoegde features vergeleken met Vi. (Zie http://www.vim.org/doc/vi.diff.txt, Engelstalige link.)
Vim is bijna volledig compatible met Vi -- behalve met bugs in Vi ;-)

Operating Systems:
Vim is verkrijgbaar voor vele systemen: AmigaOS, Atari MiNT, BeOS, DOS, MacOS, NextStep, OS/2, OSF, RiscOS, SGI, UNIX, VMS, WIn16 + Win32 (Windows95/98/00/NT) - en in het bijzonder FreeBSD and Linux. :-)

Copyright:
Het copyright is in handen van de eerste auteur en maintainer, Bram Moolenaar <bram@vim.org>.
Vim is "charity-ware", dat betekent dat je wordt uitgenodigd een donatie te doen aan weeskinderen in Oeganda (zie ":help uganda").

Broncode:
Vim is OpenSource, iedereen is welkom om verbeteringen aan te brengen!

=== Features

Beginners-editor -- Gebruikersvriendelijk:
Vim is makkelijker te leren dan Vi vanwege de uitgebreide online help, "undo"- en "redo"-commando's (maak gerust fouten -- en maak ze ongedaan, of maak ze opnieuw!), muis-ondersteuning en configureerbare pictogrammen en menu's (GUI, grafische gebruikersinterface).

Karakter-codes and Terminals:
Vim ondersteunt de iso-latin1 karakterset en termcap. (Standaard-Vi heeft hier problemen mee.)

Karakters and talen:
Vim ondersteunt van-rechts-naar-links editen (bv. voor Arabisch, Farsi, Hebreeuws), en multi-byte-teksten voor talen waar grafische karakters door meer dan een byte worden weergegeven, zoals Chinees, Japans, Koreaans (Hangul). (In technische bewoordingen, Vim ondersteunt text in UTF-8 en Unicode.)

Tekstopmaak en Visual Mode:
In Vim kun je tekst "zichtbaar" selecteren (door het te highlighten) voor je er iets mee doet, bv. kopieren, wissen, vervangen, naar links of naar rechts schuiven, hoofdletters in kleine letters veranderen of de tekst opmaken (waarbij ingesprongen tekst ingesprongen blijft). Met Vim kun je ook met rechthoekige blokken werken.

Automatisch completeren:
In Vim kan je input automatisch gecompleteerd worden -- commando's, bestandsnamen of woorden.

Automatische commando's:

Vim heeft "autocommands" voor het automatisch uitvoeren van commando's (bv. automatisch "uncompressen" van gecompresste bestanden tijdens het inladen.)

Digraph-invoer:
In Vim kun je speciale karakters als een combinatie van twee toetsen invoeren (bv. de combinatie van " en a levert een ä op) -- en je kunt nieuwe combinaties zelf definiëren.

Bestandsformaat-detectie en -conversie:
Vim herkent bestandstypes (DOS, Mac of Unix) en laat je bestanden als een ander type bewaren -- unix2dos is niet meer nodig onder DOS of Windows!

History:
Vim houdt een lijst bij ("history") van gebruikte commando's en zoektermen, die je kunt hergebruiken en wijzigen.

Macro's opnemen:
In Vim kun je je toetsaanslagen "opnemen" en weer "afspelen" om taken te herhalen.

Geheugenlimiet:
Vim heeft een veel hogere limiet voor regellengte en buffergrootte dan standaard-Vi.

Meerdere buffers en gedeeld scherm:
Met Vim kun je meerdere buffers editen en je kunt het scherm in vele subwindows splitsen (zowel horizontaal als verticaal), zodat je verschillende bestanden kunt zien of verschillende delen van één bestand.

Getal voor een commando:
In Vim kun je meer commando's laten voorafgaan door een getal dan in Vi (bv. voor "put").

Runtime-bestanden (hulpbestanden en syntax-bestanden):
Bij Vim worden 70 hulp-bestanden over verschillende aspecten van editen geleverd; sommige teksten zijn specifiek voor het gebruik op een bepaald besturingssysteem.

Scripttaal:
Vim heeft een ingebouwde scripttaal voor eenvoudige uitbreidingen.

Offset bij zoeken:
In Vim kun je een offset opgeven bij zoekopdrachten, zodat de cursor *na* de gevonden tekst komt te staan.

Sessies terugroepen:
Vim bewaart informatie over "edit-sessies" in een bestand ("viminfo") die dan weer gebruikt kan worden in een volgende sessie, bv. de lijst van buffers, bladwijzers in bestanden, registers, commando's en zoekopdrachten.

Tab-expansie:
Vim kan TAB-karakters in een tekst vervangen door spaties (expandtab, :retab);

Tag-systeem:
Met behulp van Vim kun je tekst in bestanden vinden door een index van "tags" te gebruiken, samen met een groot aantal "tag stack"-commando's.

Tekstobjecten:
Vim kent meer tekstobjecten (alinea's, zinnen. woorden en WORDS -- alle
met of zonder omringend wit) en laat je zelf de definitie van deze
objecten aanpassen.

Syntaxkleuring:
Vim laat tekst in kleur zien -- overeenkomstig de (programmeer-)taal. Je
kunt de "taal" ("syntax") van de bestanden zelf definiëren.

Vim heeft standaard meer dan 200 syntax-bestanden om tekst te kleuren in
veelgebruikte programmeertalen (Ada, C, C++, Eiffel, Fortran, Haskell,
Java, Lisp, Modula, Pascal, Prolog, Python, Scheme, Smalltalk, SQL,
Verilog, VisualBasic), wiskundige programma's (Maple, Matlab,
Mathematica, SAS), tekst met opmaak (DocBook, HTML, LaTeX, PostScript,
SGML-LinuxDoc, TeX, WML, XML), programma-output (diff, man),
setup-bestanden van programma's (4DOS, Apache, autoconfig, BibTeX, CSS,
CVS, elm, IDL, LILO, pine, procmail, samba, slrn), shell scripts
(shells: sh, bash, csh, ksh, zsh), scripttalen (awk, Perl, sed, yacc)
systeembestanden (printcap, .Xdefaults) en natuurlijk Vim en de
helpteksten.

Speciale code:
Vim heeft de mogelijkheid tot integratie met Perl, Tcl en Python. Vim
kan als OLE automation server werken onder Windows. Vim kan ook
geïnstalleerd worden met code voor X-windows, met configureerbare menu's
en ondersteuning voor de muis.
En meer. Veel meer!

De weblocatie van Vim (Engelstalig):
  http://www.vim.org/

Voor een uitgebreidere beschrijving van de features van Vim, zie
  http://www.vim.org/why.html (Engelstalig)

Geschreven door: Sven Guckes guckes@vim.org
Laatste update: Tue Sep 26 12:00:00 MET DST 2000
Vertaling: Dion Nicolaas dion@erebus.demon.nl
Laatste update: Vrij 6 Apr 20:00:00 MET DST 2001

"Mikä on Vim?"
Selitys kuudessa kilotavussa.

Vim ("Vi IMproved", "paranneltu vi") on "vi klooni", samankaltainen ohjelma kuin tekstieditori "vi".

Vim toimii tekstimoodissa kaikissa terminaaleissa, mutta siihen on myös graafinen käyttöliittymä, jossa on valikot ja tuki hiirelle.

Saatavuus:
Vim on saatavilla useille alustoille ja siinä on useita lisäominaisuuksia Vi:hin verrattuna. (kts. http://www.vim.org/doc/vi.diff.txt) Vim on yhteensopiva lähes kaikkien vi komentojen kanssa - paitsi bugisten. ;-)

Käyttöjärjestelmät:
Vim on saatavilla useille järjestelmille: AmigaOS, Atari MiNT, BeOS, DOS, MacOS, NextStep, OS/2, OSF, RiscOS, SGI, UNIX, VMS, Win16 + Win32 (Windows95/98/00/NT) - ja erityisesti FreeBSD:lle ja Linuxille. :-)

Tekijänoikeudet:
Tekijänoikeudet ovat päätekijällä ja ylläpitäjällä, Bram Moolenaarilla <bram@vim.org>. Vim on hyväntekeväisyysohjelma ("charity-ware"), sinua rohkaistaan tekemään lahjoitus esim. Ugandan orvoille (katso ":help uganda").

Lähdekoodi:
Vim on OpenSource-ohjelma ja kaikki ovat tervetulleita auttamaan sen parantamisessa!

=== Ominaisuudet

Aloittelijan editori - käyttäjäystävällinen:
Vim on paljon helpompi aloittelijalle kuin Vi, koska siinä on laaja online-aputoiminto, "undo" ja "redo" komennot (ei tarvitse välittää virheistä - käytä vain undo+redo:ta!), tuki hiirelle ja muunneltavat kuvakkeet ja valikot (GUI, graafinen käyttöliittymä).

Merkistöt ja terminaalit:
Vim:ssä on tuki iso-latin1 merkistölle ja termcap:lle (Vanilla Vi:lla on ongelmia tämän kanssa).

Kirjaimet ja kielet:
Vim:ssä on tuki oikealta-vasemmalle editoinnille (esim. arabia, farsi, heprea), ja monitavuisille (multi-byte) teksteille, kuten kielille, joissa on kuvamerkkejä, jotka esitetään useammalla kuin yhdellä tavulla, esim. kiina, japani, korea (Hangul), (teknisesti, Vim tukee UTF-8:ia ja Unicodea).

Tekstin muotoilu ja visuaalinen moodi (Visual Mode):
Vimillä voi valita tekstiä "visuaalisesti" (maalaamalla) ennen sen käsittelyä, esim. kopioida, tuhota, korvata, siirtää vasemmalle tai oikealle, vaihtaa kirjaimia isoista pieniksi ja päinvastoin, säilyttäen tekstin sisennyksen. Vim sallii tekstin valinnan ja käsittelyn myös suorakaiteen muotoisilla valintalaatikoilla.

Täydentyvät komennot:
Vim:ssä on komentoja, jotka täydentävät syötteesi - komennoilla, tiedostonimillä tai sanoilla.

Automaattiset komennot:
Vim:ssä on myös "automaattiset komennot", automaattiseten toimintojen suorittamiseksi (esim. automaattinen pakattujen tiedostojen purkaminen).

Kaksikirjainsyöte:
Vim sallii syöttää erikoismerkkejä kahden merkin yhdistelminä (esim. yhdistelmä " ja a näkyy merkkinä ä) - ja antaa myös määritellä muita yhdistelmiä.

Vim tunnistaa automaattisesti tiedoston tyypin (DOS, Mac, Unix) ja myös sallii niiden tallentamisen muun tyyppisenä - enää ei tarvita unix2dos-apuohjelmaa Windowsissa!

Historia:
Vimissä on "historia" komennoille ja hauille, joten sinun ei tarvitse muistaa edellistä komentoa tai hakulauseketta muuttaaksesi niitä.

Makrojen tallentaminen:
Vim:ssä voi tallettaa "editointia" ja ajaa niitä uudelleen toistuvissa tehtävissä.

Muistirajoitukset:
Vimissä on paljon suuremmat muistirajoitukset rivin pituuksille ja buffereille kuin vanilla Vi:ssa.

Multibufferit ja jaettu ruutu:
Vim:ssä voi editoida useita buffereita ja ruutu voidaan jakaa useisiin pienempiin ikkunoihin (vaaka- ja pystysuoraan), joten voit katsoa useaa tiedostoa tai useaa kohtaa tiedostoista samanaikaisesti.

Numeroetuliite komennoissa:
Vim sallii numeroetuliitteen useammille komennoille kuin Vi (esim. "put":lle).

Ajonaikaiset tiedostot (ohjetiedostot ja syntaksitiedostot):
[lisätiedostot, joita käytetään kun ohjelmaa ajetaan - mutta jotka eivät sisällä ohjelmakoodia, jota tarvitsisi kääntää tai linkittää.]
Vim-5.7:ssä tulee mukana 70 ohjetiedostoa (noin 2080 kilotavua tekstiä) komennoista, valinnoista, vinkkejä konfigurointiin ja editointiin. (Vim-6.0x [010311]: 85 tiedostoa, noin 2796 kilotavua tekstiä). Osa tiedostoista selvittää Vim:n käyttöä kullakin käyttöjärjestelmällä. [010311]

Skriptaus:
Vim:ssä on sisäänrakennettu skriptikieli helpottamaan lisätoimintojen tekoa.

Haun vaihesiirto:
Vim sallii vaihesiirrot hakukomennoissa, voit asettaa kursorin tietyn matkan päähän löydetyn tekstin jälkeen.

Istunnon palauttaminen:
Vim tallettaa tietoa istunnon aikana tapahtuneesta editoinnista tiedotoon ("viminfo"), mikä sallii tiedon käyttämisen seuraavan editointi-istunnon aikana, esim. bufferilistan, tiedostomerkit, rekisterit, komento ja hakuhistorian.

Tabin muuntaminen:
Vim pystyy näyttämään tekstin tabuloinnit haluttuna määränä

välilyöntejä (expandtab, :retab).

Taggaus:
Vim:ssä voi hakea tekstiä tiedostoista käyttäen "tageja" ja liittää
hakuun komentoja.

Tekstiobjektit:
Vim tunnistaa monenlaisia rakenteita (kappaleet, lauseet, sanat ja
SANAT - kaikki tyhjien välimerkkien (whitespace) kanssa tai ilman, ja
sallii näiden rakenteiden muuntelun ja määrittelyn.

Syntaksiväritys:
Vim näyttää tekstin väreissä - "ohjelmointikielen" mukaisesti.
Voit itse määritellä tiedoston "kielen" ("syntaksin").

Vimissä tulee mukana yli 200 syntaksitiedostoa tekstin värittämiseksi
useiden eri ohjelmointikielien mukaisesti (Ada, C, C++, Eiffel,
Fortran, Haskell, Java, Lisp, Modula, Pascal, Prolog, Python,
Scheme, Smalltalk, SQL, Verilog, VisualBasic), matematiikkaohjelmien
mukaisesti (Maple, Matlab, Mathematica, SAS), markup text
(DocBook, HTML, LaTeX, PostScript, SGML-LinuxDoc, TeX, WML, XML),
ohjelma tulosteen (diff, man), ohjelmien asetustiedostojen mukaisesti
(4DOS, Apache, autoconfig, BibTeX, CSS, CVS, elm, IDL, LILO, pine,
procmail, samba, slrn), komentorivitulkkien skriptikielien ja
asetustiedostojen mukaisesti (komentotulkit: sh, bash, csh, ksh, zsh),
skriptikielet (awk, Perl, sed, yacc), järjestelmätiedostojen
(printcap, .Xdefaults) ja tietysti Vim:n ja sen ohjetiedostojen
mukaisesti.

Erityiskoodi:
Vimissä on integrointituki Perlille, Tcl:lle ja Pythonille. Vim voi myös
toimia OLE-palveluna Windowsissa. Vim voidaan asentaa myös
X-windowsille tarkoitetun koodin kanssa, jolloin mukana on myös
muunneltavat valikot ja tuki hiirelle. Ja enemmän. Paljon enemmän!

=== Linkit

Vim:n kotisivut webissä:
  http://www.vim.org/

Tarkempi kuvaus Vim:n ominaisuuksista sivulla:
  http://www.vim.org/why.html

=== Tekijä ja kääntäjä

Alkuperäisen kirjoittaja: Sven Guckes guckes@vim.org
Viimeisin päivitys: Mon Mar 12 07:00:00 MET 2001

Kääntänyt suomeksi: Hiekka <hiekka@helsinki.fi>
Viimeisin päivitys: Fri Mar 30 13:36:59 EEST 2001

vim: tw=70

Vim, c'est quoi ?
Une description en 6 Ko

Vim ("Vi IMproved", c'est-a-dire "Vi ameliore") est un clone de Vi,
a savoir un programme semblable a l'editeur de texte "Vi".

Vim fonctionne en mode texte sur tous les types de terminaux, mais il dispose
aussi d'une interface graphique, avec menus deroulants et souris.

Disponibilite
Vim est disponible pour un grand nombre de plateformes et a des fonctionnalites
supplementaires par rapport a Vi (voyez "http://www.vim.org/doc/vi.diff.txt"
pour un resume des differences entre Vim et Vi).
Vim est compatible avec presque toutes les commandes de Vi (excepte les
bogues ;-).

Systemes d'exploitation
Vim tourne sur de nombreux systemes d'exploitation : AmigaOS, Atari MiNT, BeOS,
DOS, MacOS, NextStep, OS/2, OSF, RiscOS, SGI, UNIX, VMS, Win16 + Win 32
(Windows 95/98/2k/NT) - et bien entendu FreeBSD et Linux. :-)

Copyright
Les droits de Vim appartiennent a son auteur principal Bram Moolenaar
<bram@vim.org>.
Vim est un "charity-ware" : vous etes encourages a faire une donation aux
orphelins d'Ouganda (voyez : ":help uganda").

Source
Vim est OpenSource. Tout le monde est le bienvenu pour aider a son
developpement !

=== Caracteristiques

Un editeur facile a prendre en main
Pour les debutants, Vim est beaucoup plus facile a utiliser que Vi. Il offre en
effet une aide en ligne complete, la possibilite d'annulations et de
repetitions multiples (peu importe les erreurs, elles sont facilement
corrigees), le support de la souris, des icones configurables et des menus
(version GUI).

Codes caracteres et terminaux
Vim supporte le jeu de caracteres iso-latin1. Par consequent : aucun probleme
pour les caracteres accentues.

Flexibilite linguistique
Vim permet de taper du texte de droite a gauche (pratique pour ecrire par
exemple en Arabe, Farsi ou Hebreu).
Vim supporte aussi les jeux de caracteres multibytes. Il est donc possible
d'editer des textes dans des langues necessitant des interfaces graphiques, ou
les caracteres sont representes par plus d'un byte comme le Chinois, le
Japonais ou le Coreen (Hangul). Techniquement, Vim supporte les textes en
UTF-8 et Unicode.

Formatage de texte et mode visuel
Avec Vim, vous pouvez selectionner du texte de facon "visuelle" (mise en
surbrillance) avant d'agir dessus c'est-a-dire copier, effacer, substituer,
deplacer vers la gauche ou vers la droite, changer la casse, formater le texte
y compris conserver les indentations. Vim permet egalement la selection et les
operations sur des blocs de texte rectangulaires.

Commandes de terminaison automatique
Vim dispose de commandes qui permettent de terminer automatiquement vos saisies de noms de commandes, de noms de fichiers ou meme de mots.

Commandes automatiques
Vim offre aussi des commandes pour l'execution automatique d'instructions (par exemple compression/decompression automatique de fichiers).

Saisie de digraphs
Vim permet de saisir les caracteres high-bit par la combinaison de deux caracteres (par exemple "'" et "e" donnent é).

Detection du format de fichier et conversions
Vim reconnait automatiquement le type de fichier (DOS, Mac, Unix) et vous permet de les sauvegarder dans un format different. Plus besoin de unix2dos !

Historique
Vim conserve un historique des commandes et des recherches. Vous pouvez ainsi rappeler des commandes precedemment entrees ou des recherches precedemment effectuees pour les editer.

Enregistreur de macros
Vim offre la possibilite d'enregistrer votre saisie pour la restituer en cas de taches repetitives.

Limites memoire
Vim a des limitations de memoire superieures pour les longueurs de ligne et les tailles de tampons (buffers) que vanilla Vi.

Tampons multiples et partage de l'ecran
Vim permet l'edition de tampons multiples. De plus vous pouvez partager l'ecran en plusieurs sous-fenetres (horizontalement ou verticalement) de facon a voir simultanement plusieurs fichiers ou differentes parties d'un meme fichier.

Prefixes numeriques aux commandes
Vim offre la possibilite de prefixes numeriques sur davantage de commandes que vi (par exemple pour la commande "put").

Fichiers "Runtime" (fichiers d'aide et fichiers syntaxe)
Vim offre 70 fichiers d'aide presentant differents aspects de l'edition de texte, certains de ces fichiers sont specifiques a un systeme d'exploitation.

Langage de script
Vim dispose d'un langage de script integre permettant des extensions faciles.

Recherches hors-champs
Vim permet d'effectuer des recherches hors-champ. Vous pouvez donc placer le curseur *apres* le texte trouve.

Recuperation de donnees
Vim permet de stocker les informations sur une session d'edition dans un fichier ("viminfo") qui leur permet d'etre utilisees lors de la session suivante. Ces informations sont la liste des tampons, les signets, les registres et l'historique des commandes et des recherches.

Reformatage des tabulations
Vim peut remplacer automatiquement les tabulations par des espaces (expandtab, :retab).

Systeme de liens

Vim peut trouver du texte dans des fichiers en utilisant un systeme d'index et de liens. De nombreuses commandes permettent de gerer cela.

Coloration syntaxique
Vim affiche le texte en couleurs en fonction du langage de programmation. Il est possible de definir soi-meme les couleurs.
Vim est livre avec plus de 200 fichiers syntaxe pour l'affichage en couleur des langages de programmation les plus courants (Ada, C, C++, Eiffel, Fortran, Haskell, Java, Lisp, Modula, Pascal, Prolog, Python, Scheme, Smalltalk, SQL, Verilog, VisualBasic), des programmes de maths (Maple, Matlab, Mathematica, SAS), des textes marques (DocBook, HTML, LaTeX, PostScript, SGML-LinuxDoc, TeX, WML, XML), des sorties de programmes (diff, man), des fichiers de configuration (4DOS, Apache, autoconfig, BibTeX, CSS, CVS, elm, IDL, LILO, pine, procmail, samba, slrn), des scripts et configurations shell (shells: sh, bash, csh, ksh, zsh), des langages de script (awk, Perl, sed, yacc), des fichiers systeme (printcap, .Xdefaults) et bien sur des fichiers d'aide et de configuration de Vim.

Integration dans d'autres environnements
Il est possible - de facon optionnelle - d'integrer Vim avec Perl, Tcl ou Python. Vim peut servir de serveur OLE sous Windows. Vim peut aussi etre installe avec le support X-Windows, ajoutant ainsi des menu configurable et la possibilite d'utiliser la souris.

Tout ceci ne constitue qu'une petite partie des potentialites de Vim. Vim a plus a offrir. Beaucoup plus !

Le site de Vim sur la Toile : http://www.vim.org/

Pour un description plus approfondie de Vim (en anglais) :
http://www/vim.org/why.html.

Redige par : Sven Guckes guckes@vim.org
Traduit par : Lyderic Landry lydericlandry@yahoo.fr
(merci de me signaler les eventuelles erreurs ...)

Derniere mise a jour : 22 septembre 2000

"Was ist Vim?"
Eine Antwort in sechs Kilobytes.
Was ist Vim? Eine Antwort in 2222 bytes:

Vim ("Vi IMproved") ist ein "Vi Klon", dh
ein Programm, dass dem Editor "Vi" aehnlich ist.

Vim funtioniert im Textmodus auf jedem Terminal,
aber es hat auch ein graphisches Benutzerschnittstelle (GUI),
dh Menus und Unterstuetzung fuer die Maus.

Erhaeltlichkeit (Availability):
Vim ist erhaeltlich fuer viele Betriebssysteme/Plattformen
und hat viele zusaetzliche Eigenschaften (features)
(siehe dazu Seite http://www.vim.org/doc/vi.diff.txt).

Betriebssysteme (Operating Systems):
Vim is erhaeltlich fuer die folgenden Betriebssysteme:
AmigaOS, Atari MiNT, BeOS, DOS, MacOS, NextStep, OS/2, OSF,
RiscOS, SGI, UNIX, VMS, WIn16 + Win32 (Windows95/98/00/NT) -
und vor allem fuer FreeBSD und Linux. :-)

Copyright/Lizenz (Copyright):
Das Copyright ist in den Haenden von dem Hauptautor
Bram Moolenaar <bram@vim.org>.
Vim ist "charity-ware" ("Naechstenliebe"), dh Sie werden dazu
ermutigt eine Spende an Waisenkinder in Uganda zu machen
(siehe ":help uganda").

Source:
Vim is "OpenSource", dh der Quellcode fuer das Programm ist fuer
jeden frei erhaeltlich , und man darf Aenderungen daran vornehmen.
(siehe http://www.opensource.org/certification-mark.html)

=== Features

Editor für Anfänger - Benutzerfreundlich:
Vim ist weitaus leichter zu erlernen als Vi wegen der großen Dokumentation;
das Kommando "undo" macht die Änderungen schrittweise rückgängig, und
falls man dabei zu weit geht, kann man mit "redo" wieder vorwärts gehen.
Bei Fehler sind kein Problem - einfach "zurückgehen".
Außerdem bietet Vim konfigurierbare Menüs und Unterstützung für die Maus.

Character codes and Terminals:
Vim has support for the iso-latin1 character set and for termcap.
(Vanilla Vi has problems with this.)

Characters and Languages:
Vim supports for right-to-left editing (eg with Arabian, Farsi, Hebrew),
and multi-byte texts, ie languages with graphical characters represented
by more than one "byte", such as Chinese, Japanese, Korean (Hangul),
(Technically speaking, Vim supports text in UTF-8 and Unicode.)

Text Formatting and Visual Mode:
With Vim you can select text "visually" (with highlighting) before you
"operate" on it, eg copy, delete, substitute, shift left or right,
change case of letters or format the text incl preserving indented text.
Vim allows selection and operations on rectangular text blocks, too.

Completion Commands:

Vim has commands which complete your input -
either with commands, filenames, and words.

Automatic Commands:
Vim also has "autocommands" for automatic execution of commands
(eg automatic uncompression of compressed files).

Digraph Input:
Vim allows you to enter special characters by a combination
of two characters (eg the combination of " and a yields an ä) -
and allows you to define other combinations, too.

Fileformat detection and conversion:
Vim automatically recognizes the type of files (DOS, Mac, Unix)
and also lets you save them in any other format -
no need for unix2dos on Windows any more!

History:
Vim has a "history" for commands and searches, so you can
recall previous commands or search pattern to edit them.

Macro Recording:
Vim allows to "record" your editing for replaying for repetitive tasks.

Memory Limits:
Vim has much higher memory limits for line length
and buffer sizes than vanilla Vi.

Multiple Buffers and Split Screen:
Vim allows editing of multiple buffers and you can split the
screen into many subwindows (both horizontally and vertically),
so you can see many files or many parts of some files.

Number Prefix to commands:
Vim allows a number prefix for more
commands than with Vi (eg for "put").

Runtime Files (Helpfiles and Syntax Files):
Vim comes with 70 help files on various aspects of editing;
some texts are specifically for use on some operating system.

Scripting:
Vim has a built-in scripting language for easy extension.

Search Offset:
Vim allows offsets for search commands, so you
place the cursor *after* the found text.

Session Recovery:
Vim allows to store information of an editing session into a file ("viminfo")
which allows them for being used with the next editing session, eg
buffer list, file marks, registers, command and search history.

Tab expansion:
Vim can expand tabs within the text with spaces (expandtab, :retab).

Tag system:
Vim offers to find text in files by using an index with "tags"
together with many tag stack commands.

Text Objects:
Vim knows more text objects (paragraphs, sentences, words and WORDS -
all with and without surrounding whitespace) and allows to configure
the definition for these objects.

Syntax Coloring:
Vim shows text in color - according to its "(programming) language".
You can define the "language" ("syntax") of the files yourself.

Vim comes with 200+ syntax files for colorizing text in
common programming languages (Ada, C, C++, Eiffel, Fortran,
Haskell, Java, Lisp, Modula, Pascal, Prolog, Python,
Scheme, Smalltalk, SQL, Verilog, VisualBasic),
math programs (Maple, Matlab, Mathematica, SAS), markup text
(DocBook, HTML, LaTeX, PostScript, SGML-LinuxDoc, TeX, WML, XML),
program output (diff, man), setup files of programs (4DOS, Apache,
autoconfig, BibTeX, CSS, CVS, elm, IDL, LILO, pine, procmail, samba, slrn),
shell scripts and setups (shells: sh, bash, csh, ksh, zsh),
script languages (awk, Perl, sed, yacc)
system files (printcap, .Xdefaults) and
of course for Vim and its helptexts.

Special code:
Vim has optional integration with Perl, Tcl and Python.
Vim can act as an OLE automation server under Windows.
Vim can also be installed with code for X-windows,
adding configurable menus and support for the mouse.
And more. Lots more!

Vim's HomePage in the WWW:
  http://www.vim.org/

For a more elaborate description
of Vim's features see the page
  http://www.vim.org/why.html

Written by: Sven Guckes guckes@vim.org
Last update: Tue Oct 03 20:00:00 MET DST 2000

"Öé åßíáé ôï Vim?"
Ìéá åîÞãçóç óå 8ÊÂ

O Vim ("Vi IMproved") åßíáé Ýíáò "êëþíïò" ôïõ vi
("vi clone"), äçëääÞ Ýíá ðñüãñáììá ðáñüìïéï ìå ôïí
åðåîåñãáóôÞ êåéìÝíïõ "vi"

O Vim äïõëåýåé óå ðåñéâÜëëïí êåéìÝíïõ (êïíóüëåò êëð)
óå êÜèå ôåñìáôéêü, áëëÜ åðßóçò Ý÷åé êáé Ýíá ãñáöéêü
ðåñéâÜëëïí, äçëääÞ ðñÜãìáôá üðùò ìåíïý êáé õðïóôÞñéîç
ãéá ðïíôßêé.

Äéáèåóéìüôçôá:
To Vim åßíáé äéáèÝóéìï ãéá ðïëëÝò ðëáôöüñìåò êáé Ý÷åé
ðïëëÝò åðéðëÝïí äõíáôüôçôåò óå óýãêñéóç ìå ôïí Vi.
(êïéôÜîôå óôï http://www.vim.org/doc/vi.diff.txt - óýíôïìá
êáé óôá åëëçíéêÜ). Ôï Vim åßíáé óõìâáôü ìå ó÷åäüí üëåò ôéò
åíôïëÝò ôïõ Vi - åêôüò áðü ôá bugs ôïõ ;-)

Ëåéôïõñãéêü Óõóôþìáôá:
Ï Vim åßíáé äéáèÝóéìïò ãéá ðÜñá ðïëëÜ óõóôþìáôá: AmigaOS,
Atari MiNT, BeOS, DOS, MacOS, NextStep, OS/2, OSF, RiscOS,
SGI, UNIX, VMS, Win16 + Win32 (Windows95/98/00/NT) - êáé
éäéáßôåñá ãéá ôï FreeBSD êáé ôï Linux. :-)

Copyright:
Ôï copyright áíÞêåé óôïí êýñéï óõããñáöÝá - äçìéïõñãü êáé
óõíôçñçôÞ ôïõ, ôïí Bram Moolenaar <bram@vim.org>. Ï Vim
åßíáé "öéëáíèùðéêüò", ð÷ áíôß ôÝëïõò ÷ñÞóçò óáò
ðñïõñÝðïõìå íá êÜíåôå ìéá ðñïóïñÜ óôá ïñöáíÜ ôçò Uganda.
(Ãéá ðåñéóóüôåñåò ðëçñïöïñßåò êïéôÜîôå ":help uganda").

Êþäéêáò:
Ï Vim åßíáé áíïé÷ôüò ùò ðñïò ôïí êþäéêá (OpenSource) êáé üëïé
åßíáé åõðñüóåêôïé íá âïçèÞóïõí óôçí âåëôßùóÞ ôïõ.

=== ×áñáêôçñéóôéêÜ

ÅðåîåñãáóôÞò Ãéá Áñ÷Ünéïõò - Öéëéêüò ðñïò ôïí ×ÞóôçÇ:
Ï Vim åßíáé ðïëý ðéï åýêïëïò ãéá ôïõò áñ÷Ünéïõò áðü ôïí Vi
åîáéôßáò ôçò åêôåôáìÝíçò ôïõ on-line âïÞèåéáò, ôùí åíôïëþí
"undo" êáé "redo" ("áíáßñåóç" êáé "åðáíÜëçøç" - ìçí áíçóõ÷åßôå
ãéá ôá ëÜèç, áðëÜ ÷ñçóéìïðïéÞóôå "undo" êáé "redo"!), õðïóôÞñéîç
ôïõ ðïíôéêéíý êáé ðñïóáñìüóéìá åéêïíßäéá êáé ìåíïý (óå ãñáöéêü
ðåñéâÜëëïí).

Êþäéêåò ×áñáêôÞñùí êáé Ôåñìáôéêü:
Ï Vim õðïóôçñßæåé ôï iso-latin1 óåô ÷áñáêôÞñùí êáé ôï termcap.
(Ï Vanilla Vi Ý÷åé ðñüâëçìá ìå áõôÜ.)

×áñáêôÞñåò êáé Ãëþóſåò:
Ï Vim õðïóôçñßæåé êáé ôçí ãñáöÞ áðü-äåîéÜ-ðñïò-áñéóôåñÜ (ð÷ ãéá
÷ñÞóç ìå Áñáâéêü, Ââñáúêü), êáé êåßìåíá multi-byte, ð÷ ãëþóſåò ìå
ãñáöéêïýò ÷áñáêôÞñåò ðïõ áíáðáñßſôáíôáé ìå ðåñéóóüôåñá áðü Ýíá bytes
üðùò åßíáé ôá ÊéíÝæéêá, Ãéáðùíýæéêá, ÊïñåÜôéêá êëð). Áðü ôå÷íéêÞò
ðëåõñÜò, ï Vim õðïóôçñßæåé êåßìåíï óå ìïñöÞ UTF-8 êáé Unicode.

Äéáìüñöùóç ÊåéìÝíïõ êáé Visual Mode:
Ìå ôïí Vim ìðïñåßôå íá åðéëÝîåôå êåßìåíï "ïðôéêÜ" (ìå "öþôéóÞ" ôïõ)
ðñéí ôï ÷ñçóéìïðïéÞóåôå, ð÷ ôï áíôéãñÜøåôå, ôï äéáãñÜøåôå,

ôï áíôéêáôáóôÞóåôå, ôï ìåôáêéíÞóåôå äåîéÜ Þ áñéóôåñÜ, åíáëëÜîåôå
êåöáëáßá ìå ìéêñÜ Þ äéáìïñöþóåôå ôï êåßìåíï êáé üëá áõôÜ äéáôçñþíôáò
ôéò ðáñáãñÜöïõò óôï êåßìåíï. Ï Vim åðéôñÝðåé ôçí åðéëïãÞ êáé ôç
÷ñÞóç êåéìÝíïõ êáé ìå ïñèïãþíéá ìðëîê êåéìÝíïõ.

Óõìðëçñùóç Åíôïëþí:
Ï Vim Ý÷åé åíôïëÝò ïé ïðïßåò óõìðëçñþíïõí ôçí åßóïäü óáò -
åßôå ìå åíôïëÝò åßôå ìå ïíúìáôá áñ÷åßùí åßôå ìå ëÝîåéò.

Áõôùìáôåò Åíôïëýò:
Ï Vim Ý÷åé åðßóçò ôéò "autocommands" (áõôùìáôåò åíôïëÝò) ãéá
áõôùìáôç åêôÝëóç åíôïëþí (ð÷ áõôùìáôç áðïóõìßåóç óõìðéåóìÝíùí
áñ÷åßùí).

Äéãñáììáôéêþ Åßóïäïò:
Ï Vim åðéôñÝðåé ôçí åßóïäï åéäéêþí ÷áñáêôÞñùí ìå Ýíáí óõíäõáóìü
äýï ÷áñáêôÞñùí (ð÷ ï óõíäõáóìüò åíüò " ìå Ýíá O äßíåé Ýíá â) - êáé
óáò åðéôñÝðåé íá ïñßóåôå êáé äéêïýò óáò óõíäõáóìïýò.

Áíß÷íåõç êáé Ìåôáôñïðþ Ìïñöþí Êåéìýíïõ:
Ï Vim áõôùìáôá áíáãíùñßæåé ôïí ôýðï ôùí áñ÷åßùí (DOS, Mac, Unix)
êáé óáò äßíåé ôçí äõíáôüôçôá íá ôá óþóåôå óå ïðïéïäÞðïôå Üëëï
format - äåí ÷ñåéÜæåóôå ðéá ôï unix2dos óôá Windows!

Éóôïñéêü:
Ï Vim êñáôÜ ôï "éóôïñéêü" ôùí åíôïëþí êáé ôùí áíáæçÞóåùí êáé Ýôóé
ìðïñåßôå íá áíáêáëÝóåôå ðñïçãïýìåíåò åíôïëÝò Þ ðñüõõá áíáæçÞóåùí
ãéá íá ðååîåñãáóôåßôå.

ÊáôáãñáöÞ Ìáêñïåíôïëþí:
Ï Vim åðéôñÝðåé ôçí "åããñáöÞ" ôùí åíåñãåþí óáò êáôÜ ôçí åðåîåñãáóßá
êÜðïéïõ êåéìÝíïõ Ýôóé þóôå íá åßíáé äõíáôÞ ç ðïëëáðëÞ åöáñìïãÞ ôïõò
óå åðáíáëçðôéêÝò åñãáóßåò.

¼ñéá ÌíÞìçò:
Ï Vim Ý÷åé ðïëý õøçëüôåñá üñéá ìíÞìçò ãéá ôï ìÞêïò ãñáììÞò êáé
ìåãáëýôåñïõò buffers (åíäéÜìåóïõò ÷þñïõò áðïèÞêåõóçò) áðü ôïí ãíÞóéï Vi.

Ðïëëáðëïß buffers êáé ×ùñéóìüò Ïèüíçò:
Ï Vim åðéôñÝðåé ôçí åðåîåñãáóßá ìå ðïëëáðëýò buffers (÷þñïõò åíäéÜìåóçò
áðïèÞêåõóçò) êáèþò êáé ôïí ÷ùñéóìü ôçò ïèüíçò óå ðïëëÜ åðéìÝñïõò ðáñÜèõñá
(õðïðáñÜèõñá) ôüóï ïñéæüíôéá üóï êáé êÜèåôá, Ýôóé þóôå íá åßíáé äõíáôÞ ç
ôáõôü÷ñïíç åðåîåñãáóßá ðïëëþí êåéìÝíùí Þ ôïõ ßäéïõ êåéìÝíïõ óå äéáöïñåôéêÜ
óçìåßá.

Áñéèìçôéêü Ðñïèýìáôá óå Åíôïëýò:
Ï Vim åðéôñÝðåé áñéèìçôéêü ðñüèåìá ãéá ðåñéóóüôåñåò åíôïëÝò áðü ôïí Vi
(ð÷ ãéá ôçí "put").

Âïçèçôéêü êáé Óõíôáêôéêü Áñ÷åßá ×ñüíïõ-ÅêÝëáóçò (Runtime):
Ï Vim-5.7 óõíïäåýåôáé áðü 70 âïçèçôéêÜ áñ÷åßá (ðåñßðïõ 2080K êåéìÝíïõ)
ãéá åíôïëÝò, åðéëïãÝò, ìå óõìâïõëÝò ãéá ñõèìßóåéò êáé åðåîåñãáóßá.
Ï Vim-6.0w ðåñéëáìâÜíåé 85 áñ÷åßá êáé 2780Ê êåéìÝíïõ. ÊÜðïéá áñ÷åßá åßíáé
åéäéêÜ ãéá ôç ÷ñþóç ôïõ Vim óå êÜèå ëåéôïõñãéêü óýóôçìá. [11/03/2001]

Scripts åñãáóßáò:
Ï Vim Ý÷åé åíóùìáôùìÝíç ãëþóóá ãéá scripts ðñïöÝñïíôáò åýêïëç åðÝêôáóç.

Ìåôáôüðéóç áíáæÞôçóçò:
Ï Vim åðéôñÝðåé ìåôáôïðßóåéò (offsets) ãéá ôéò åíôïëÝò áíáæÞôçóçò, Ýôóé þóôå

íá ìðïñåß íá ôïðïèåôåß ôïí äñïìÝá (êÝñóïñá) *ìåôÜ* ôï åíôïðéóìÝíï êåßìåíï.

ÁíÜêôçóç Ðåñéüäïõ Åñãáóßáò:
Ï Vim åðéôñÝðåé íá áðïèçêåýåôáé ðëçñïöïñßá ìéáò ðåñéüäïõ åðåîåñãáóßáò óå Ýíá áñ÷åßï ("viminfo") ðñÜìá ôï ïðïßï åðéôñÝðåé ôçí ÷ñçóéìïðïßçóÞ ôçò óå ìéá åðüìåíç ðåñßïäï. ÔÝôïéá ðëçñïöïñßá åßíáé ð÷ ç ëßóôá ìå ôïõò buffers, óçìÜäéá óôï áñ÷åßï, êáôá÷ùñçÝò, ôï "éóôïñéêü" åíôïëþí êáé áíáæçÞóåùí.

ÅðÝêôáóç (áíôéêáÜóôáóç) ôùí Tab:
Ï Vim ìðïñåß íá áíôéêáôáóôÞóåé ôá tabs óôï êåßìåíï ìå êåíïýò ÷áñáêÞñåò (expandtab, :retab).

Óýóôçìá Âôéêâôþí:
Ï Vim ìðïñåß íá áíáæçÞóåé êåßìåíï óå áñ÷åßá ÷ñçóéìïðïéþíôáò Ýíá åõñåôÞñéï ìå "âôéêÝôåò" êáèþò åðßóçò êáé ðïëëÝò åíôïëÝò ÷åéñéóìïý ôçò óôïßâáò âôéêâôþí.

Áíôéêåßìåíá Êåéìýíùí:
Ï Vim "ãíùñæåé" ðåñéóóüôåñá áíôéêåßìåíá-óõóôáôéêÜ êåéìýíùí (ðáñáãñüöïõò, ðñïÜóåéò, ëÝîåéò, óõìâïëïóåéñÝò - üëá ìå Þ ÷ùñßò ðåñéâÜëëïíôá êåíÜ) êáé åðéôñÝðåé ôç ñýèìéóç êáé ïñéóìü áõôþí ôùí óõóôáôéêþí.

×ñùìáôéóìüò Óýíôáîçò:
Ï Vim ÷ñùìáßæåé ôï êåßìåíï óýìöùíá ìå ôç ãëþóóá ðñïãñáììáôéóìïý ðïõ ÷ñçóéìïðïéåßôå. Ìðïñåßôå íá ïñßóåôå ôçí "ãëþóóá" ("óýíôáîç") ðïõ ÷ñçóéìïðïéåßôå. Ï Vim óõíïäåýåôáé áðü ðåñéóóüôåñá áðï 200 áñ÷åßá óýíôáîçò ãéá ôï ÷ùìáôéóìü ôïõ êåéìÝíïõ ãéá ôéò ðåñéóóüôåñåò ãíùóôÝò ãëþóóåò ðñïãñáììáôéóìïý (Ada, C, C++, Eiffel, Fortran, Haskell, Java, Lisp, Modula, Pascal, Prolog, Python, Scheme, Smalltalk, SQL, Verilog, VisualBasic), ìáèçìáôéêÜ ðñïãñÜììáôá (Maple, Matlab, Mathematica, SAS), åðáõîçìÝíï êåßìåíï (DocBook, HTML, LaTeX< PostScript, SGML-LinuxDoc, TeX, WML, XML), Ýîïäï ãíùóôþí ðñïãñáììÜôùí (diff, man), áñ÷åßá ñõèìßóåùí ðñïãñáììÜôùí (4DOS, Apache, autoconfig, BibTeX, CSS, CVS, elm, IDL, LILO, pine, procmail, samba, slrn), shell scripts êáé ñõèìßóåéò (sh, bash, csh, ksh, zsh), ãëþóóåò script (awk, Perl, sed, yacc) áñ÷åßá óõóôÞìáôïò (printcap, .Xdefaults) êáé âÝâáéá ãéá ôïí ßäéï ôïí Vim êáé ôá áñ÷åßá âïçèåßáò ôïõ.

Åéäéêüò Êþäéêáò:
Ï Vim ìðïñåß íá óõíäõáóôåß ìå Perl, Tcl êáé Python. Åðßóçò, ìðïñåß íá ëåéôïõñãÞóåé ùò áíôéêåßìåíï OLE óå ðåñéâÜëëïí Windows. Ï Vim ìðïñåß åðßóçò íá åãêáôáóôáèåß ìå êþäéêá ãéá X-windows, ðñïóèÝôïíôáò ñõèìéæüìåíá menu êáé õðïóôÞñéîç ãéá ôï ðïíôßêé.
Êáé ðïëëÜ Üëëá... :-)))

Åðßóéìç Óåëßäá ôïõ Vim óôï WWW:
http://www.vim.org

Ãéá ìéá ðéï åìðåñéóôáôùìÝíç ðåñéãñáöÞ ôùí äõíáôïôÞôùí ôïõ Vim êïéôÜîôå:
http://www.vim.org/why.html

Áñ÷éêü Êåßìåíï: Sven Guckes <guckes@vim.org>
Ðñïóáñìïãþ óôá Åëëçíéêü: ÄçìÞñçò ÔæÝíïò <tzenos@ceid.upatras.gr>

ÅíçìÝñùóç:
Êõñéáêþ 11 Ìáñôßïõ 2001.

"Mi a Vim?"
Magyarázat hat kilobájtban

A Vim ("Vi IMproved" - feljavított Vi) egy "vi klón", vagyis egy program,
ami a "vi" szövegszerkesztõhöz hasonlít.

A Vim karakteres üzemmódban minden terminálon mûködik, de van grafikus
felülete is, menükkel és egértámogatással.

Elérhetõség:
A Vim rengeteg platformon elérhetõ, és rengeteg plusz szolgáltatása van az
Vi-hoz képest. (lásd http://www.vim.org/doc/vi.diff.txt)
A Vim szinte mindenben kompatibilis a Vi parancsaival - kivéve a hibákat :)

Operációs rendszerek:
A Vim számos operációs rendszeren megtalálható: AmigaOS, Atari MiNT, BeOS, DOS,
MacOS, NextStep, OS/2, OSF, RiscOS, SGI, UNIX, VMS, Win16 + Win32
(Windows95/98/00/NT) - és különösen FreeBSD illetve Linux alatt. :)

Copyright:
A jogok a fö szerzõ és karbantartó Bram Moolenaar <bram@vim.org> kezében
vannak. A Vim "adomány-ware", vagyis kérjük, hogy adakozz Uganda árvái
javára. (lásd ":help uganda").

Forrás:
A Vim nyílt forrású, és bárki segíthet a fejlesztésben!

=== Jellemzõk

A kezdõk editora - Felhasználóbarát:
A Vim sokkal könnyebben használható kezdõknek, mint a Vi a részletes online
súgó, az "undo" (visszavonás) és "redo" (újra) parancsok (ne
ess kétségbe ha hibázol, csak vond vissza, vagy csináld újra!), az
egértámogatás, és a beállítható menük és ikonok miatt (GUI).

Karakter kódok és Terminálok:
A Vim támogatja az iso-latin1 karakterkészletet és a termcap-et. (A
Vi-nak ezzel problémái voltak)

Karakterek és nyelvek:
A Vim támogatja a jobbról balra írást (Arab, Héber), a több byte-os
szövegeket, olyan nyelvek esetében, ahol a grafikus karaktereket több mint
egy byte-on ábrázolják, mint például a Kínai, Japán, Koreai. (Technikailag
a Vim támogatja az UTF-8 és Unicode kódolást.)

Szöveg formázás és Vizuális mód:
Vim-mel lehtõség van a szíveget vizuálisan (kiemeléssel) kijelölni a
mûvelet elõtt (másolás, törlés, csere, jobbra, balra eltolás, kis- nagybetû
váltás, szövegformázás a bekezdések megtartásával)
A Vim lehetõvé teszi téglalap blokkok kijelölését, és azokon mûveletek
végzését.

Kiegészítõ parancsok:
A Vim képes a bevitt szöveg kiegészítésére - legyen az parancs, fájlnév,
vagy bármilyen szó.

Automata parancsok:
A Vim rendelkezik automatikus parancsokkal (autocommands) parancsok
automatikus végrehajtására (pl tömörített állományok kitömörítése)

Repülõ ékezetes (digraph) bevitel:
A Vim lehetõvé teszi, hogy különleges karaktereket két karakter kombinációjaként vigyünk be. (például a " és az a együtt ä-t eredményez) Lehetõség van saját kombinációk megadására is.

Fájlformátum azonosítás és konverzió:
A Vim automatikusan felismeri a fájlok típusát (DOS, Mac, Unix) és bármely másikban képes azokat elmenteni - nincs többé szükseg unix2dos-ra Windows alatt!

Elõzõ parancsok:
A Vim nyílván tartja az elõzõ parancsokat és kereséseket, így ezek késõbb visszakereshetõk, átszerkeszthetõk.

Makró rögzítés:
A Vim lehetõvé teszi, a mûveletek "felvételét", amik ismétlõdõ feladatokhoz visszajátszhatók.

Memória korlátok:
A Vim sokkal tágabb korlátokkal rendelkezik sor illetve bufferméretre mint az eredeti Vi.

Több Buffer és Osztott Képernyõ:
A Vim megengedi több buffer használatát, valamint a képernyõ függõlegesen és vízszintesen is több ablakra bontható, így egyszerre látható több állomány, vagy ugyannak az állománynak több része.

Szám elõtag parancsokhoz:
A Vim a Vi-nál több parancs elõtt engedi szám elõtag használatát (pl a beillesztés)

Runtime Fájlok (Súgó és szintakszis állományok):
A Vim 70 súgóval érkezik a szerkesztés különbözõ területeit illetõen, némelyikük kifejezetten bizonyos operációs rendszerekre vonatkozik.

Script-ek:
A Vim saját script-nyelvvel rendelkezik a könnyû bõvíthetõségért.

Eltolásos keresés:
A Vim megengedi eltolás használatát a keresés parancsokkal, így a kurzor akár a megtalált szöveg *mögé* is kerülhet.

Munkaállapot visszaállítás:
A Vim képes a munkaállapot elmentésére egy állományba ("viminfo"), amiket a következõ alkalommal vissza lehet állítani. Ezekbe beletartoznak a bufferlisták, az könyvjelzõk, regiszterek, az elõzõ parancsok és keresések.

Tab kifejtés:
A Vim képes a szövegen belül a tabokat szóközökre cserélni (expandtab, :retab).

Tag rendszer:
A Vim lehetõvé teszi szövegrészletek keresését állományokban "tag" indexek segítségével, illetve "tag" verem parancsokkal.

Szöveg objektumok:
A Vim több szöveg objektumot ismer (bekezdés, mondat, szó és SZÓ - mindezek határolókarakterekkel vagy azok nélkül.) és ezek definíciói megváltoztathatóak.

Szintaktika kiemelés:
A Vim színezi a szöveget - a saját "(programozási) nyelvének" megfelelõen. A fájlok "nyelve" ("szintaktikája") szabadon beállítható.

A Vim több mint 200 beépített szintaktikai állománnyal rendelkezik a népszerûbb programozási nyelvkhez (Ada, C, C++, Eiffel, Fortran, Haskell, Java, Lisp, Modula, Pascal, Prolog, Python, Scheme, Smalltalk, SQL, Verilog, VisualBasic), matematikai programokhoz (Maple, Matlab, Mathematica, SAS), leíró nyelvekhez (DocBook, HTML, LaTeX, PostScript, SGML-LinuxDoc, TeX, WML, XML), programok végeredményéhez (diff, man), beállító állományokhoz (4DOS, Apache, autoconfig, BibTeX, CSS, CVS, elm, IDL, LILO, pine, procmail, samba, slrn), shell szkriptekhez (shells: sh, bash, csh, ksh, zsh), szkript nyelvekhez (awk, Perl, sed, yacc) rendszerállományokhoz (printcap, .Xdefaults) és természetesen a VIM saját súgóállományaihoz.

Különleges kód:
A Vim opcionálisan képes együttmûködni Perl-lel, Tcl-lel, Python-nal. Windows alatt OLE szerverként mûködik. X-windows támogató kóddal is telepíthetõ, átszerkeszthetõ menükkel és egértámogatással. És még többel, sokkal többel.

Vim honlap a weben:
  http://www.vim.org/

A Vim szolgáltatásainak részletesebb leírása megtalálható a
  http://www.vim.org/why.html
címen.

Írta: Sven Guckes guckes@vim.org
Utolsó módosítás: Tue Oct 03 20:00:00 MET DST 2000

"Cos'e' Vim?"
Una descrizione in 6 kilobyte

Vim ("Vi IMproved" cioe' "Vi migliorato") e' un clone di vi, cioe' un programm
a simile all'editor testuale "Vi".

Vim funziona in modalita' testo su ogni terminale, ma possiede anche una
interfaccia grafica (GUI), ossia con menu' e supporto per il mouse.

Disponibilta':
Vim e' disponibile per molte piattaforme e aggiunge molte funzinalita' a Vi
(vedi http://www.vim.org/doc/vi.diff.txt per una descrizione dettagliata delle
differenze). Vim e' compatibile con quasi tutti i comandi di Vi, fatta
eccezione per i bachi di Vi. ;-)

Sistemi operativi:
Vim e' disponibile per moltissimi sistemi: AmigaOS, Atari MiNT, BeOS, DOS,
MacOS, NextStep, OS/2, OSF, RiscOS, SGI, UNIX, VMS, WIn16 + Win32
(Windows95/98/00/NT) - e specialmente per FreeBSD and Linux. :-)

Copyright:
I diritti di Vim appartengono al suo autore principale, Bram Moolenaar
<bram@vim.org>. Vim e' "charity-ware", cioe' siete incoraggiate a fare una
donazione agli orfani dell'Uganda (vedi ":help uganda" per maggiori
informazioni).

Sorgenti:
Vim e' OpenSource: tutti coloro che vogliono aiutare a migliorarlo sono
i benvenuti.

Un Editor facile per i principianti:
Vim e' molto piu' facile di Vi per i principianti grazie al completo aiuto in
linea, alla possibilita' di annullare e ripetere i comandi (non proccupatevi
degli errori - si correggono facilemente!), al supporto per il mouse e alle
icone e menu configurabili (versione GUI).

Codici dei caratteri e terminali:
Vim supporta i caratteri iso-latin1: nessun problema, quindi, con i caratteri
accentati!

Flessibilita' di linguaggio:
Vim permette di scrivere da destra a sinistra (utile per scrivere in Arabo,
Farsi o Ebraico) e i caratteri "multibyte", ossia linguaggi i cui caratteri
sono rappresentati da piu' di un singolo byte, come Cinese, Giapponese,
Coreano (Hangul). Tecnicamente, Vim supporta testo in UTF-8 e Unicode.

Formattazione del testo e modo visuale:
Con Vim potete selezionare testo in modo visuale (cioe' evidenziandolo) prima
di eseguire un comando su di esso, ad esempio per copiare, cancellare,
sostituire, spostare a destra o sinistra, passare da minuscole a maiuscole o
formattare preservando l'indentatura.
Con Vim potrete anche selezionare (e eseguire comandi) su blocchi rettangolari
di testo.

Comandi di completamento:
Vim puo' completare automaticamente cio' che state scrivendo, sia esso un
comando, il nome di un file o una semplice parola.

Comandi automatici:
Vim offre anche la possibilita' di eseguire automaticamente i comandi (ad

esempio per decomprimere e comprimere file compressi)

Digitazione di caratteri speciali ("digraph"):
Vim permette di scrivere caratteri speciali usando una combinazione di due caratteri (ad esempio "'" e "e" generano é) e permette di definire nuove combinazioni.

Riconoscimento del formato del file e conversione:
Vim riconosce automaticamente il tipo di file (DOS, Mac, Unix) e ti permette di salvare i file in altri formati. Dimenticate unix2dos!.

Storia:
Vim conserva memoria dei comandi e delle ricerche, cosi' e' possibile richiamare comandi o ricerche precedenti e eventualmente modificarle.

Registrazione di macro:
Vim ti permette di registrare i comandi che digiti per eseguire compiti ripetitivi.

Limite di memoria:
Vim ha limiti molto piu' eleveti per la lunghezza delle linee e per la dimensione della memoria tampone (buffer) rispetto a vanilla Vi.

Buffer multipli e divisione dello schermo:
Vim permette di modificare piu' buffer ed e' possibile dividere lo schermo in diverse sotto finestre (sia in orizzontale che in verticale), in modo da vedere piu' file o piu' parti dello stesso file.

Prefissi numerici ai comandi:
Con Vim si puo' far precedere un numero a molti piu' comandi rispetto a Vi (per esempio il comando "put")

File "Runtime" (File di aiuto e di sintassi):
Vim e' distribuito con 70 file di aiuto in linea sui vari aspetti dell'editing; alcuni testi sono specifici per i vari sistemi operativi.

Linguaggio di Script:
Vim possiede un linguaggio di programmazione integrato che permette un facile estensione delle funzionalita'.

Ricerche sfalsate:
Vim permette di effettuare delle ricerche sfalsate, cosi' e' possibile mettere il cursore *dopo* il testo trovato.

Recupero delle sessioni:
Vim permette di scrivere le inforamzione di una sessione in un file ("viminfo"), in modo da poterle usare in una sessione successiva ad esempio la lista dei buffer, i segnaposti dei file, i registri, i comandi e la storia delle ricerche effettuate.

Espansione dei Tab:
Vim puo' trasformare i tab nel testo in spazi (expandtab, :retab).

Indice:
Vim offre la possibilita' di trovare del testo nei file usando un indice con "tags", e di navigare con facilita'.

Ogetti testuali:
Vim riconosce diversi oggetti testuali (paragrafi, frasi, parole con e senza gli spazi bianchi che le circondano) e permette di configurare la definizione

di questi oggetti.

Colorazione sintattica:
Vim mostra il testo a colori, a seconda del diverso linguaggio di programmazione che si sta editando. E' possibile definire la tua specifica sintassi per i tuoi file.

Vim e' distribuito con piu' di 200 file di sintassi per colorare il testo dei linguaggi di programmazione piu' comuni (Ada, C, C++, Eiffel, Fortran, Haskell, Java, Lisp, Modula, Pascal, Prolog, Python, Scheme, Smalltalk, SQL, Verilog, VisualBasic), programmi di matematica (Maple, Matlab, Mathematica, SAS), testo con marcatori (DocBook, HTML, LaTeX, PostScript, SGML-LinuxDoc, TeX, WML, XML), uscita di programmi (diff, man), file di configurazione (4DOS, Apache, autoconfig, BibTeX, CSS, CVS, elm, IDL, LILO, pine, procmail, samba, slrn), linguaggi di script (awk, Perl, sed, yacc), file di sistema (printcap, .Xdefaults), e, naturalmente, per i file di aiuto e configurazione di Vim.

Codice speciale:
Vim possiede una integrazione opzionale con Perl, Tcl e Python.
Vim puo' agire come un server OLE in Windows.
Vim puo' anche essere installato con il supporto X-Windows, aggiungendo menu configurabili e supporto per il mouse.

Ma tutto questo e' solo un piccola parte delle potenzialita' di Vim.
Ce ne sono molte, MOLTE di piu'...

L'indirizzo del sito di Vim sulla rete e':
  http://www.vim.org/

Per una descrizione piu' particolareggiata (in inglese) sulle caratteristiche di Vim, vai all'indirizzo:
  http://www.vim.org/why.html

Redazione : Sven Guckes guckes@vim.org
Traduzione : Stefano Lacaprara Stefano.Lacaprara@pd.infn.it
             (vi ringrazio per segnalazioni di errori e omissioni)
Ultimo aggiornamento : 12 Novembre 2000

Vim ¤É¤Ï?
6KB ¤Ç¤ÎÀâÌÀ (EUC Japanese)

  vim ("Vi IMproved") ¤Ï vi ¥¯¥í¡¼¥ó¡¢¤Ä¤Þ¤ê¥Æ¥-¥¹¥È¥¨¥Ç¥£¥¿¤Ç¤¢¤ë
vi ¤Ë»÷¤¿¥×¥í¥°¥é¥à¤Î¤Ò¤È¤Ä¤Ç¤¹¡£

  vim ¤Ï¤¢¤é¤æ¤ëÃ¼Ëö¡Ê¥¨¥ß¥å¥ì¡¼¥¿¡Ë¤Î¥Æ¥-¥¹¥È¥â¡¼¥É¤ÇÆ°ºî¤·¤Þ¤¹¤¬¡¢
GUI¡Ê¥°¥é¥Õ¥£¥Ã¥¯¡¦¥æ¡¼¥¶¡¦¥¤¥ó¥¿¥Õ¥§¡¼¥¹¡Ë¤â»ý¤Ã¤Æ¤¤¤Æ¡¢¤á¥Ë¥å¡¼¤ä
¥Þ¥¦¥¹¤ò¥µ¥Ý¡¼¥È¤·¤Þ¤¹¡£

²ÄÍÑÀ-:
  vim ¤ÏÂ¿¤¯¤Î¥×¥é¥Ã¥È¥Õ¥©¡¼¥à¤Ç»È¤¨¡¢vi ¤ÈÈæ¤Ù¤ÆÂ¿¤¯¤ÎµÇ½¤¬ÄÉ²Ã¤µ
¤ì¤Æ¤¤¤Þ¤¹¡£ (http://www.vim.org/doc/vi.diff.txt ¤ò»²¾È²¼¤µ¤¤)
  vim ¤Ï vi ¤Î¤Û¤È¤ó¤É¤¹¤Ù¤Æ¤Î¥³¥Þ¥ó¥É¤È¤Î¸ß´¹À-¤¬¤¢¤ê¤Þ¤¹¡£vi ¤Î¥Ð
¥°¤Ï½ü¤-¤Þ¤¹ ;-)

¥ª¥Ú¥ì¡¼¥Æ¥£¥ó¥°¡¦¥·¥¹¥Æ¥à¡ÊOS¡Ë
  vim ¤ÏÂ¿¤¯¤Î¥·¥¹¥Æ¥à¤Ç»È¤¨¤Þ¤¹: AmigaOS, Atari MiNT, BeOS, DOS,
MacOS, NextStep, OS/2, OSF, RiscOS, SGI, UNIX, VMS, WIn16 + Win32
(Windows95/98/00/NT) - ÆÃ¤Ë FreeBSD and Linux. :-)

ÃøºîÌ¢:
  À©ºî¤ÈÊÝ¼é¤ò¼ç¤Ë¹Ô¤Ê¤Ã¤Æ¤¤¤ë Bram Moolenaar »á <bram@vim.org> ¤Ë
ÃøºîÌ¢¤¬¤¢¤ê¤Þ¤¹¡£
  vim ¤Ï¥Á¥ã¥ê¥Æ¥£¡¼¥¦¥§¥¢¤Ç¤¹¡£¥¦¥¬¥ó¥À¤ÎÌ¸É»ù¤Ø¤Î´óÉÕ¤ò¤ª¾©¤á¤·¤Þ
¤¹¡£

¥½¡¼¥¹:
  vim ¤Ï¥ª¡¼¥×¥ó¡¦¥½¡¼¥¹¤Ç¤¹¡¢¤³§¤µ¤ó¤«¤é¤Î²þÁ±¤Î¤ª¼êÅÁ¤¤¤ò´¿·Þ¤·¤Þ
¤¹¡£

=== µ¡Ç½

½é¿´¼Ô¤Î¥¨¥Ç¥£¥¿¡¢¥æ¡¼¥¶¤Ë¿ÆÀÚ:
  vim ¤¬ vi ¤è¤ê¤â½é¿´¼Ô¤Ë¤ï¤«¤ê¤ä¤¹¤¤¤Î¤Ï¡¢¤¥ª¥ó¥é¥¤¥ó¡¦¥Ø¥ë¥×¤ä¡¢¤
¥³¥Þ¥ó¥É¤Î¼è¾Ã¤È¤ä¤ê¾Ä¾¤·¡Ê¼ºÇÔ¤òµ¤¤Ë¤»¤ºÇÃ±¤Ë¼è¾Ã+¤ä¤ê¾Ä¾¤·¡Ë¤ä¡¢¥Þ
¥¦¥¹¤ÀßÄê²ÄÇ½¤Ê¥¢¤¥³¥ó¤ä¥á¥Ë¥å¡¼¤Î¥µ¥Ý¡¼¥È¡ÊGUI¡Ë¤¬¤¢¤ë¤«¤é¤Ç¤¹¡£

Ê¸»ú¥³¡¼¥É¤ÈÃ¼Ëö¡Ê¥¨¥ß¥å¥ì¡¼¥¿¡Ë:
  Vim ¤Ï iso-latin1 Ê¸»ú¥»¥Ã¥È¤È¤½¤Î termcap ¤ò¥µ¥Ý¡¼¥È¤·¤Æ¤¤¤Þ¤¹¡£
(Vanilla Vi ¤Ï¤³¤³¤ËÌäÂê¤òÊú¤¨¤Æ¤¤¤Þ¤¹)

Ê¸»ú¤È¸À¸ì:
  vim ¤Ï±¦¤«¤é°¸Ë½ñ¤¯¸À¸ì¡ÊÎã: ¥¢¥é¥Ó¥¢¸ì¡¢¥Ú¥ë¥·¥ã¸ì¡¢¥Ø¥Ö¥é¥¤¸ì¡Ë
¤È¥Þ¥ë¥Á¥Ð¥¤¥È¡¦¥Æ¥-¥¹¥È¡¢¤Ä¤Þ¤êÆæ¹ñ¸ì¡¢ÆüËÜ¸ì¡´Ú¹ñ¸ì¡Ê¥Ï¥ó¥°¥ë¡Ë
¤Î¤è¤¦¤Ë 1 ¥Ð¥¤¥È¤è¤êÄ¹¤¤¥Ð¥¤¥ÈÄ¹¤ÇÉ½¸½µ¤ì¤ë¥°¥é¥Õ¥£¥«¥ë¤ÊÊ¸»ú¤ò¥µ
¥Ý¡¼¥È¤·¤Æ¤¤¤Þ¤¹¡£µ»½ÑÅª¤Ë¸À¤¦¡¤È¡¢UTF-8 ¤È ¥æ¥Ë¥³¡¼¥É¤Î¥Æ¥-¥¹¥È¤ò¥µ
¥Ý¡¼¥È¤·¤Æ¤¤¤ë¤³¤È¤Ë¤Ê¤ê¤Þ¤¹¡£)

¥Æ¥-¥¹¥È¡¦¥Õ¥©¡¼¥Þ¥Ã¥È¡ÊÀ°·Á¡Ë¤È¥Ó¥¸¥å¥¢¡¢¥å¥¢¥ë¥â¡¼¥É:
  vim ¤ò»È¤¨¤Ð¡¢¥Æ¥-¥¹¥È¤Î°ìÉô¤ò¥Ï¥¤¥é¥¤¥È¤¹¤ë¡Ê¿§¤òÊÑ¤¨¤ë¡Ë¤³¤È¤Ç
¡Ö¥Ó¥¸¥å¥¢¤ËÉ¡×Á²Âò¤·¡¢ÁàºîÌ¤ë¤³¤È¤¬Ç¤-¤Þ¤¹¡£ÎãÌã¤¨¤Ð¡¢¥³¥Ô¡¼¡¢¤ï
½ü¡¢ÃÖ´¹¡¢º¸±¦¤Î¥·¥Õ¥È¡¢ÂçÊ¸»ú¡¿¾®Ê¸»úÊÑ´¹¡¢¥Æ¥-¥¹¥È¤Î»ú²¼¤²¤òÊÝ¤Ã
¤¿¤Þ¤ÞÀ°·Á¤Ê¤É¤¬²ÄÇ½¤Ç¤¹¡£vim ¤Ï¶ë·Á¤Î¥Æ¥-¥¹¥È¥Ö¥í¥Ã¥¯¤òÁª¤Ò¡¿ÁàºîÌ
¤¹¤ë¤³¤È¤â¤Ç¤-¤Þ¤¹¡£

¼«Æ°¥³¥Þ¥ó¥É:
  vim ¤Ï¼«Æ°¥³¥Þ¥ó¥É¤âÈ÷¤¨¤Æ¤¤¤Æ¡¢¡Ê¥Õ¥¡¥¤¥ë¤ÆÉ¤ß¹þ¤ß»þ¤Ê¤ÉÍÍ¡¹¤Ê¥¤
¥Ù¥ó¥ÈÈ¯À¸»þ¤Ë¡Ë¥³¥Þ¥ó¥É¤ò¼Â¹Ô¤Ç¤-¤Þ¤¹¡£¡ÊÎã: °µ½Ì¤µ¤ì¤¿¥Õ¥¡¥¤¤ë¤Î
¼«Æ°²òÅà)

¹ç»ú¡Ê¤´¤¦¤¸, ¤¢¤ï¤»¤¸¡Ë¤ÎÆþÎÏ
  vim ¤Ï 2 Ê¸»ú¤òÁÈ¤ß¹ç¤ï¤»¤¿ÆÃ¼ì¤ÊÊ¸»ú¤òÆþÎÏ¤Ç¤-¤Þ¤¹¡£(Îã:¡Ö"¡×¤È
¡Öa¡×¤Ç¥¦¥à¥é¥¦¥È¤ÎÉÕ¤¤¤¿ a ¡ËÂ¾¤ÎÁÈ¤ß¹ç¤ï¤»¤òÄêµÁ¤¹¤ë¤³¤È¤â²ÄÇ½¤Ç
¤¹¡£

¥Õ¥¡¥¤¥ë·Á¼°¤Î¸¡½Ð¤ÈÊÑ´¹:
  vim ¤Ï Dos, Mac, Unix ¤È¤¤¤Ã¤¿¥Õ¥¡¥¤¥ë¤Î¥¿¥¤¥×¤ò¼«Æ°¸¡½Ð¤·¤¿¤ê¡¢
Â¾¤Î¥Õ¥©¡¼¥Þ¥Ã¥È¤Ç³ÊÇ¼¤·¤¿¤ê¤¹¤ë¤³¤È¤â¤Ç¤-¤Þ¤¹¡£Windows ¤Î unix2dos
¤Ï¤â¤¦É¬Í×¤¢¤ê¤Þ¤»¤ó¡£

ÍúÎò¡Ê¥Ò¥¹¥È¥ê¡Ëµ¡Ç½:
  vim ¤Ï¥³¥Þ¥ó¥É¤È¸¡°÷¤ÎÍúÎò¤ò²±¤¨¤Æ¤¤¤ë¤Î¤Ç¡¢°ÊÁ°¤ËÆþÎÏ¤·¤¿¥³¥Þ¥ó
¥É¤ä¸¡°÷¥Ñ¥¿¡¼¥ó¤òºÆÆþÎÏ¤·¤¿¤ê¡¢½ñ¤-´¹¤¨¤ÆºÆÈ¯¹Ô¤¹¤ë¤³¤È¤¬¤Ç¤-¤Þ¤¹¡£

¥Þ¥¯¥íµ-Ï¿:
  vim ¤Ê¤é¡¢²¿ÅÙ¤â·«¤êÊÖ¤µ¤ì¤ëºî¶È¤òºÆ¸½¤¹¤ë¤¿¤á¤ËÊÔ½¸ºî¶È¤ò¡Öµ-Ï¿¡×
¤Ç¤-¤Þ¤¹¡£

¥á¥â¥êÀ©¸Â:
  vim ¤Î 1 ¹Ô¤¢¤¿¤ê¤ÎÄ¹¤µ¤ä¥Ð¥Ã¥Õ¥¡¤Î¥µ¥¤¥º¤Ï vanilla Vi ¤ÈÈæ¤Ù¤ÆÈó
¾ï¤ËÂç¤-¤¯ÀßÄê¤µ¤ì¤Æ¤¤¤Þ¤¹¡£

¥Þ¥ë¥Á¥Ð¥Ã¥Õ¥¡¤È²èÌÌÊ¬³ä:
  vim ¤ÏÊ£¿ô¤Î¥Ð¥Ã¥Õ¥¡¤Î¥¨¥Ç¥£¥Ã¥È¤¬²ÄÇ½¤Ç¡¢²èÌÌ¤ò¤¤¤¯¤Ä¤«¤Î¥µ¥Ö¥¦
¥£¥ó¥É¥¦¤ËÊ¬³ä¤Ç¤-¤Þ¤¹¡£Ê¿åÊ¿ÊýÞ¤È¿âÄ¾ÊýÞ¤ÎÎ¾ÊýÞ¡Ë¡£¤½¤Î¿¤á¡¢Ê£¿ô
¥Õ¥¡¥¤¥ë¤äÊ£¿ô²Õ½ê¤òÄ¯¤á¤ë¤³¤È¤¬¤Ç¤-¤Þ¤¹¡£

¿ôÃÍ¤ÎÀè¹Ô»ØÄê:
  vim ¤Ï vi ¤è¤ê¤âÂ¿¤¯¤Î¥³¥Þ¥ó¥É¤ÇÀè¹Ô¿ôÃÍ¤ò»ØÄê¤Ç¤-¤Þ¤¹¡£¡ÊÎã: put¡Ë

¥é¥ó¥¿¥¤¥à¡¦¥Õ¥¡¥¤¥ë¡Ê¥Ø¥ë¥×¥Õ¥¡¥¤¥ë¤È¥·¥ó¥¿¥¯¥¹¡¦¥Õ¥¡¥¤¥ë¡Ë
  vim ¤Ë¤ÏÍÍ¡¹¤Ê¥¨¥Ç¥£¥Ã¥È¤Î¾õ¶·¤Ë¹Ç¤ï¤»¤¿ 70 ¤Î¥Ø¥ë¥×¥Õ¥¡¥¤¥ë¤¬ÍÑ
°Õ¤µ¤ì¤Æ¤¤¤Þ¤¹¡£¤¤¤¯¤Ä¤«¤Î¥Æ¥-¥¹¥È¤Ï OS ¤Ë¹Ç¤ï¤»¤¿¤â¤Î¤Ç¤¹¡£

¥¹¥¯¥ê¥×¥È:
  vim ¤Ë¤Ï¥¹¥¯¥ê¥×¥È¸À¸ì¤¬¤ÁÈ¤ß¹þ¤Þ¤ì¤Æ¤¤¤Æ¡¢´ÊÃ±¤Ê³ÈÄ¥¤¬¤Ç¤-¤ë¤è¤¦
¤Ë¤Ê¤Ã¤Æ¤¤¤Þ¤¹¡£

¸¡°÷¥ª¥Õ¥»¥Ã¥È:
  vim ¤Ç¤Ï¸¡°÷¥³¥Þ¥ó¥É¤Ë¥ª¥Õ¥»¥Ã¥È¤ò»ØÄê¤Ç¤-¡¢¥Æ¥-¥¹¥È¤ò¸«¤Ä¤±¤¿¸å
¤Î¥«¡¼¥½¥ë¤Î°ÜÆ°Àè¤ò»ØÄê¤Ç¤-¤Þ¤¹¡£

¥»¥Ã¥·¥ç¥ó¡¦¥ê¥«¥Ð¥ê
  vim ¤Ï¥¨¥Ç¥£¥Ã¥È¡¦¥»¥Ã¥·¥ç¥ó¤Î¾ðÊó¤ò¥Õ¥¡¥¤¥ë¡Êviminfo¡Ë¤ËÂàÈò¤·¡¢
¼¡²ó¤ÎÎµ¯Æ°»þ¤Ë²óÉü¤¹¤ë¤³¤È¤¬¤Ç¤-¤Þ¤¹¡£ÂàÈò¡¿²óÉü¤Ç¤-¤ë¥¨¥Ç¥£¥Ã¥È¡¦
¥»¥Ã¥·¥ç¥ó¤ÏÎã¤¨¤Ð¥Ð¥Ã¥Õ¥¡¥ê¥¹¥È¡¢¥Õ¥¡¥¤¥ë¥Þ¡¼¥¯¡¢¥ì¥¸¥¹¥¿¡¢¥³¥Þ¥ó
¥É¤È¸¡°÷¤Î¥Ò¥¹¥È¥ê¤Ê¤É¤Ç¤¹¡£

¥¿¥Ö¤Î¶õÇò²½:
  vim ¤Ï¥¿¥Ö¤ò¶õÇò²½¤·¤Æ¥Æ¥-¥¹¥È¤Ë¤¹¤ë¤³¤È¤¬¤Ç¤-¤Þ¤¹¡£(expandtab,
:retab).

¥¿¥°:
  vim ¤Ï tags ¥Õ¥¡¥¤¥ë¤ò²¾È¤·¤ÆÀ¤¿¥¤¥ó¥Ç¥¯¥¹¤ÈÂ¿¤¯¤Î¥¿¥°¡¦¡¥¹¥¿¥Ã
¥¯¡¦¥³¥Þ¥ó¥É¤ò»È¤Ã¤Æ¥Õ¥¡¥¤¥ë¤Æâ¤Î¥Æ¥-¥¹¥È¤ò¸¡°÷¤Ç¤-¤Þ¤¹¡£

¥Æ¥-¥¹¥È¡¦¥ª¥Ö¥¸¥§¥¯¥È:
  vim ¤Ï¥Ñ¥é¥°¥é¥Õ¡¢¥»¥ó¥Æ¥ó¥¹¡¢word¡¢WORD¡¢¤¢¤ë¤¤¤Ï¤½¤ì¤é¤ò¶õÇò¤Ç
¤¯¤¯¤Ã¤¿¤â¤Î¤É¤È¤¤¤Ã¤¿¡¢Ö¥Æ¥-¥¹¥È¥ª¥Ö¥¸¥§¥¯¥È¡×¤òÇ§¼±¤·¡¢¤½¤ì¤é¤Î

ÄêµÀ¤òÁßÄê¤¹¤ë¤³¤È¤â¤Ç¤-¤Þ¤¹¡£

¥·¥ó¥¿¥¯¥¹¤´¤È¤Î¿§¤Å¤±:
  vim ¤Ï¥¨¥Ç¥£¥Ã¥È¤¹¤ë¥×¥í¥°¥é¥ß¥ó¥°¸À¸ì¤´¤È¤Ë¥Æ¥-¥¹¥È¤ò¿§ÉÕ¤±¤Ç¤-
¤Þ¤¹¡£
¤É¤ó¤Ê¸À¸ì¡Ê¥·¥ó¥¿¥¯¥¹¡Ë¤Î¥Õ¥¡¥¤¥ë¤Ç¤¢¤ë¤«¤òÄêµÁ¤¹¤ë¤³¤È¤â¤Ç¤-¤Þ¤¹¡£

  vim ¤Ë¤Ï 200 ¤òÄ¶¤¹¥·¥ó¥¿¥¯¥¹¡¦¥Õ¥¡¥¤¥ë¤¬ÍÑ°Õ¤µ¤ì¤Æ¤¤¤Þ¤¹¡£°ìÈÌÅª
¤Ê¥×¥í¥°¥é¥ß¥ó¥°¸À¸ì¸þ¤± (Ada, C, C++, Eiffel, Fortran, Haskell,
Java, Lisp, Modula, Pascal, Prolog, Python, Scheme, Smalltalk, SQL,
Verilog, VisualBasic)¡¢¿ô³ØÍÑ¥×¥í¥°¥é¥à¸þ¤± (Maple, Matlab,
Mathematica, SAS)¡¢Web ¤Ê¤É¤Î¥Þ¡¼¥¯¥¢¥Ã¥×¡¦¥Æ¥-¥¹¥È¸þ¤± (DocBook,
HTML, LaTeX, PostScript, SGML-LinuxDoc, TeX, WML, XML)¡¢¥×¥í¥°¥é¥à
¤Î½ÐÎÏ¸þ¤± (diff, man)¡¢ ¥×¥í¥°¥é¥à¤Î¥»¥Ã¥È¥¢¥Ã¥×¡¦¥Õ¥¡¥¤¥ë¸þ¤±
(4DOS, Apache, autoconfig, BibTeX, CSS, CVS, elm, IDL, LILO, pine,
procmail, samba, slrn)¡¢ ¥·¥§¥ë¡¦¥¹¥¯¥ê¥×¥È¤È¥»¥Ã¥È¥¢¥Ã¥×¸þ¤± (sh,
bash, csh, ksh, zsh)¡¢¥¹¥¯¥ê¥×¥È¸À¸ì¸þ¤± (awk, Perl, sed, yacc)¡¢
¥·¥¹¥Æ¥à¡¦¥Õ¥¡¥¤¥ë¸þ¤± (printcap, .Xdefaults)¡¢¤µ¤é¤Ë¤â¤Á¤í¤ó vim
¤È¤½¤Î¥Ø¥ë¥×¥Æ¥-¥¹¥È¸þ¤±¤Ê¤É¡£

ÆÃ¼ìÍÑÅÓ¸þ¤±¤ËÍÑ°Õ¤µ¤ì¤¿¥³¡¼¥É:
  vim ¤Ë¤Ï Perl, Tcl, Python ¸þ¤±¤Ë¥ª¥×¥·¥ç¥ó¤ÇÍÑ°Õ¤µ¤ì¤¿¤â¤Î¤¬¤¢¤ê
¤Þ¤¹¡£
  vim ¤Ë¤Ï Windows ²¼¤Ç OLE ¥ª¡¼¥È¥á¡¼¥·¥ç¥ó¡¦¥µ¡¼¥Ð¤È¤·¤ÆÆ°ºî¤µ¤»
¤ë¥³¡¼¥É¤¬¤¢¤ê¤Þ¤¹¡£
  vim ¤Ë¤Ï X-window ÍÑ¤Î¥³¡¼¥É¤¬ÍÑ°Õ¤µ¤ì¤Æ¤¤¤Æ¡¢ÀßÄê²ÄÇ½¤Ê¥á¥Ë¥å¡¼
¤ä¥Þ¥¦¥¹¤¬¥µ¥Ý¡¼¥È¤µ¤ì¤Þ¤¹¡£
¤Ê¤É¤Ê¤É¡¢Â¾¤Ë¤â¿¤¯¤µ¤ó¤¢¤ê¤Þ¤¹¡ª

WWW ¤Ç¤Î vim ¤Î¥Û¡¼¥à¥Ú¡¼¥¸:
  http://www.vim.org/

vim ¤Îµ¡Ç½¤Ë´Ø¤¹¤ë¾Ü¤·¤¤µ-½Ò:
  http://www.vim.org/why.html

¸¶ÃØ (English):
  Sven Guckes guckes@vim.org
  Last update: Tue Oct 03 20:00:00 MET DST 2000

ÆüËÜ¸ìÌõ (Translated into Japanese):
  Toshiya Kawakami kawakami@lead.dion.ne.jp
  Last updated: Mon Nov 06 19:50:00 JST 2000

"VimÁÍ¶õ?"
6Å³·Î¹ÙÀÌÆ® ±æÀÌÀÇ ¼³¸í

Vim ("Vi IMproved") Àº "vi È£È¯" ÀÌ´Ù. Áï,
ÅØ½ºÆ® ÆíÁý±âÀÎ vi¿Í º¯½ÁÇÑ ÇÁ·Î±×·¥À´Ù.

VimÀº ¸ðµç ´Ü¸»±âÀÇ ÅØ½ºÆ®¸ðµå¿¡¼- ÀÛµ¿ÇÑ´Ù.
¶ÇÇÑ, ¸Þ´º ¹× ¸¶¿ì½º »ç¿ëÀ» Áö¿øÇÏ´Â µî ±×·¡ÇÈ
»ç¿ëÀÚÈ¯°æ(GUI) ¿¡¼-µµ ÀÛµ¿µÈ´Ù.

°¡¿ë¼º:
VimÀº ¸¹À» ÇÃ·§Æû¿¡¼- »ç¿ëÇÒ ¼ö ÀÖ°í, Vi¿¡ ºñÇØ ´õ ¸¹À» Ãß°¡µÈ
±â´ÉéÀ» °¡Áö°í ÀÖ´Ù. (http://www.vim.org/doc/vi.diff.txt Âü°í)
VimÀº °ÅÀÇ ¸ðµç Vi ¸í·É¾î¿Í È£È¯ÀÌ µÈ´Ù. - Vi ÀÇ ¹ö±×¸¦ Á¦¿ÜÇÏ°í.. ;-)

¿Î¿µÃ¼Á¦:
VimÀº ¸¹À» ½Ã½ºÅÛ¿¡¼- »ç¿ëÇÒ ¼ö ÀÖ´Ù: amigaOS, Atari MiNT, BeOS, DOS,
MacOS, NextStep, OS/2, OSF, RiscOS, SGI, UNIX, VMS, Win16 + Win32
(À©µµ¿ìÁî95/98/2000/NT) - FreeBSD¿Í ¸®´ª½º´Â Æ¯È÷³ª ´õ Àß Áö¿øÇÑ´Ù. :-)

ÀúÀÛ±Ç:
Vim¿¡ ´ëÇÑ ÀúÀÛ±ÇÀº, ÁÖ °³¹ßÀÚÀÌÀÚ °ü¸®ÀÚÀÎ Bram Moolenaar
<bram@vim.org> ¿¡°Ô ÀÖ´Ù.
VimÀº "ÀÚ¼±¿þ¾î" ÀÌ´Ù. ¿ì°£´ÙÀÇ °í¾ÆµéÀ» À§ÇÑ ±â°Î¸¦ ±ÇÀåÇÏ´Â ¹ÙÀÌ´Ù.
(Âü°í ":help ugand")

¼Ò½º:
VimÀº ¿ÀÇÂ¼Ò½ºÀÌ´Ù. °³¹ß¿¡ Âü¿©ÇÏ´Â ¸ðµç ÀÌµéÀ» È¯¿µÇÑ´Ù.

=== ±â´É

ÃÊº¸ÀÚ¸¦ À§ÇÑ ÆíÁý±â - »ç¿ëÀÚ Ä£¼÷¼º:
VimÀº Vi¿¡ ºñÇÏ¿© ÃÊº¸ÀÚ¿¡°Ô ÈÎ¾À ´õ Ä£¼÷ÇÏ´Ù. Ç³ºÎÇÑ ¿Â¶óÀÎ µµ¿ò»À¹ª
"¸í·ÉÃë¼Ò" ¹× "¸í·ÉÀç½Ãµµ" µîÀÇ ±â´É (½Ç¼ö¸¦ µÎ·Á¿ö »¶ó - ¸í·ÉÃë¼Ò¿Í
¸í·ÉÀç½Ãµµ¸¸ ÀÖÀ¸é ½±°Ô ÇØáµÈ´Ù) »Ó ¾Æ´Ï¶ó, ¸¶¿ì½º Áö¿ø°ú ¼³Á¤ °¡´ÉÇÑ
¾ÆÀÌÄÜ ¹× ¸Þ´º¿Í °°Àº ±×·¡ÇÈ»ç¿ëÀÚÈ¯°æ(GUI)À» Áö¿øÇÏ±â ¶§¹®ÀÌ´Ù.

¹®ÀÚÄÚµå¿Í ´Ü¸»±â:
VimÀº iso-latain1 ¹®ÀÚ¼Â°ú termcap À» Áö¿øÇÑ´Ù.
(±âÁ¸ÀÇ vi¿¡¼- ¹®Á¦ µÇ´Â ºÎºÐÀÌ´Ù.)

¹®ÀÚ¿Í ¾ð¾î:
VimÀº ¿ìÃø¿¡¼- ÁÂÃøÀ¸·Î ¾²´Â ¾ð¾î (¿¹: ¾Æ¶ø¾î, ÀÌ¶õ¾î, È÷ºê¸®¾î) ¿Í
¸ÖÆ¼¹ÙÀÌÆ® ¹®ÀÚ, Áï 1¹ÙÀÌÆ® ÀÌ»óÀ» Â÷ÁöÇÏ¸é¼- ±×·¡ÇÈÀÛ¸·Î Ç¥½Ã°¡
µÇ´Â ¾ð¾îÀÎ Áß±¹¾î, ÀÏº»¾î, ÇÑ±¹¾î ¹®ÀÚ¸¦ Áö¿øÇÑ´Ù.
(±â¼úÀûÀ¸·Î ¸»Çï¸é, VimÀº UTF-8°ú Unicode¸¦ Áö¿øÇÑ´Ù.)

ÅØ½ºÆ® Æ÷¸Ë °ú ºñÁÖ¾ó ¸ðµå:
VimÀ» »ç¿ëÇØ¼-, ÅØ½ºÆ®¸¦ ºñÁê¾óÇÏ°Ô ¼±ÅÃÇÏ¿© ¸í·ÉÀ» ¼öÇà ÇÒ ¼ö ÀÖ´Ù.
Áï, º¹»ç ¶Ç´Â »èÁ¦, Ä¡È¯, ÁÂ¿ì·ÎÀÇ ÀÚ®ÀÌµ¿, ´ë/¼Ò¹®ÀÚ º¯È¯, Á¤·ÄµÈ
ÇüÅÂ¸¦ À¯ÁöÇÑ¤·Î ÅØ½ºÆ®¸¦ Æ÷¸ËÇÏ´Â µîÀÇ Á¶ÀÛÀÌ °¡´ÉÇÏ´Ù. ¶ÇÇÑ, VimÀº
»ç°¢Çü ÇüÅÂ·Î ÅØ½ºÆ®¸¦ ¼±ÅÃÇÏ¿© ¸í·ÉÀ» ¼öÇà ÇÒ ¼öµµ ÀÖ´Ù.

¸í·É¾î ¿Ï¼º±â´É:
VimÀº ÀÕ·ÂÇÏ´Â ¸í·É¾î³ª È-ÀÏÀ¸§ ¶Ç´Â ÀÏ¹ÝÀûÀÎ ´Ü¾î¸¦
´Ù ¾²±âµµ Àü¿¡ ¿Ï¼º½ÃÅ°´Â ±â´ÉÀ» °¡Áö°í ÀÖ´Ù.

ÀÚµ¿ ¸í·É¾î:
VimÀº ÀÚµ¿ ¸í·É¾î ±â´ÉÀ» °¡Áö°í ÀÖ´Ù. (Áï, ¾ÐÃàµÈ È-ÀÏÀ»

ÁÐ¾îµå¸®´Ä ¼ø°£¿¡ ÁÙµ¿Á¸·Ï ¾ÐÁàÇØÁ¦¸¦ ÇÏ¿© ·ÏµùÇÏ´Ä °ÏÁÏ °¡´ÉÇÏ´Ù)

ÀÌÁßÀ½ÀÚ ÀÔ·Â:
Vim¿¡¼´Â 2°³ÀÇ ¹®ÀÚ¸¦ Á¶ÇÕÇÏ¿© Æ¯¼öÇÑ ¹®ÀÚ¸¦ ÀÔ·ÂÇÒ ¼ö ÀÖ´Ù.
(¿¹: " ¿Í a ¸¦ ÀÌ¿ëÇÏ¿© À¯·´¾î¿¡ ÀÖ´Â ¾Ç½¾Æ®°ÎÈ£°¡ µ¡ºÙ¿©Áø ¹®ÀÚ¸¦
ÀÔ·ÂÇÒ ¼ö ÀÖ´Ù) ¶ÇÑ, ´Ù¸¥ ¹æ½ÄÀÇ Á¶ÇÕµµ Á¤ÀÇÇÒ ¼ö ÀÖ´Ù.

È-ÀÏÇü½ÄÀÇ °¨Áö¿Í º¯È¯:
VimÀº ·ÎµùÇÏ´Â È-ÀÏÀÇ Çü½Ä(µµ½º, ¸ÅÅ²Åä½Ã, À¯´Ð½º)À» ÀÚµ¿À¸·Î
°¨ÁöÇÏ°í, ´Ù¸¥ Æ÷¸ËÀ¸·Î ÀúÀåÇÒ ¼ö µµ ÀÖ°Ô ÇÑ´Ù. - ÀÌÁ¦ À©µµ¿ì¿¡¼-
ÀÛ¾÷ÇÏ±â À§ÇØ ±»ÀÌ unix2dos ¸¦ »ç¿ëÇÒ ÇÊ¿ä°¡ ¾ø´Ù!

ÀÌ·Â±â´É:
VimÀº ¸í·É¾î ¹× °Ë»ö¾î¿¡ ´ëÇÑ ÀÌ·ÂÀ» °¡Áö°í ÀÖ´Ù. µû¶ó¼-,
ÀÌÀüÀÇ ¸í·É¾î ¹× °Ë»ö¾î¸¦ ¾ó¸¶µçÁö ´Ù½Ã ºÎ¸¦¼ö ÀÖ°í ÆíÁýÇÏ¿©
Àç½ÇÇà ÇÒ ¼ö ÀÖ´Ù.

¸ÅÅ©·Î ±â·Ï:
VimÀº ÆíÁýÀÛ¾÷ °úÁ¤À» ±â·ÏÇÏ¿© ÀçÇöÇÏ´Â ±â´ÉÀ» °¡Áö°í ÀÖ¾î¼-,
¹Ý°¹ÀûÀÎ ÀÛ¾÷À» ¿ëÀÌÇÏ°Ô µµ¿ÍÁØ´Ù.

¸Þ¸ð¸® Á¦ÇÑ:
VimÀº ÇÑ ÁÙÀÇ ÃÖ´ë±æÀÌ¿Í ¹öÆÛÀÇ ÃÖ´ë Å©±â°¡, ±âÁ¸ÀÇ Vi¿¡
ºñÇÏ¿© ¿ùµîÇÏ°Ô ¸¹µ·Ï ¸Þ¸ð¸®¦ ÇÒ´çÇÏ°í ÀÖ´Ù.

´Ù¼öÀÇ ¹öÆÛ¿Í È-¸éºÐÇÒ:
Vim¿¡¼´Â ´Ù¼ö°³ÀÇ ¹öÆÛ¸¦ ÆíÁýÇÒ ¼ö ÀÖ°í, È-¸éÀ» ¿©·¯°³·Î
(¼öÆò¶Ç´Â ¼öÁ÷ ¹æÇâÀ¸·Î) ºÐÇÒÇÏ¿© »ç¿ëÇÒ ¼ö ÀÖ´Ù. µû¶ó¼-,
¿©·¯°³ÀÇ È-ÀÏÀ» µ¿½Ã¿¡ ÆíÁýÇÏ°Å³ª, ÇÑ È-ÀÏÀÇ ¿©·¯ ºÎºÐÀ»
°¢±â ´Ù¸¥ È-¸é¿¡ µÎ°í ÆíÁýÇÏ´Â °ÍÀÌ °¡´ÉÇÏ´Ù.

¼ýÀÚ¸¦ ¼±Çà½ÃÅ² ¸í·É¾î:
VimÀº Vi¿Í ¸¶Âù°¡Áö·Î ¸í·É¾î ¾Õ¿¡ ¼ýÀÚ¸¦ ¼±Çà½ÃÅ°´Â
°ÍÀÌ °¡´ÉÇÏ´Ù. (¿¹: put)

½ÇÇà º¸Á¶ È-ÀÏ (µµ¿ò»È-ÀÏ °ú ¹®¹ýÈ-ÀÏ)
VimÀº ÆíÁý±â´É¿¡ °ü·ÃÇÑ ´Ù¾çÇÑ »ç¿ë¹ýÀÌ ¼ö·ÏµÈ 70°³ÀÇ µµ¿ò» È-ÀÏÀ»
°¡Áö°í ÀÖ´Ù. ±×ÁßÀ¸·Î È-ÀÏ¿¡´Â Æ¯Á¤ ¿î¿µÃ¼Á¦¿¡ Àû¿ëµÇ´Â ³»¿ëÀÌ
´ã°Ü ÀÖ´Ù.

½ºÅ©¸³Æ®:
VimÀº ½ºÅ©¸³Æ® ¾ð¾î°¡ ³»ÀåµÇ¾î ÀÖ°í, ÀÌ¸¦ ÅëÇÏ¿© ±â´ÉÀ»
È®ÀåÇÒ ¼ö ÀÖ´Ù.

°Ë»ö ¿ÀÇÁ¼Â:
VimÀº °Ë»ö¸¦ ·É¾î¿¡ ¿ÀÇÁ¼ÂÀ» ÁöÁ¤ÇÒ ¼ö ÀÖ´Ù. Áï, Ã£¾ÆÁø
°Ë»öÆÐÅÏÀÇ À§Ä¡·ÎºÎÅÍ ÁöÁ¤ÇÑ ¿ÀÇÁ¼Â ¸¸Å- ÀÌµ¿ÇÏ¿© Ä¿¼¸¦ À§Ä¡
½ÃÅ³ ¼ö ÀÖ´Ù.

ÆíÁýÈ¯°æ º¹±¸±â´É:
VimÀº ÆíÁýÈ¯°æÀ» Æ¯Á¤È-ÀÏ("viminfo")¿¡ ÀúÀåÇÏ¿©, ´ÙÀ½¹ø¿¡ vimÀ»
½ÇÇà½ÃÅ³ ¶§¿¡ ÀÌÀü°ú °°ÀºÈ¯°æÀ¸·Î º¹±µÉ ¼ö ÀÖµµ·Ï ÇÒ ¼ö ÀÖ´Ù.
¿¹¸¦ µé¾î, ¹öÆÛ¸®½ºÆ®, È-ÀÏ ¸¶©, ·¹Áö½ºÅÍÀÇ ³»¿ë, ¸í·É¾î ¹×
°Ë»ö¾ÌÀÇ ±â·ÏµéÀÌ ÀÖ´Ù.

ÅÇ È®Àå:
VimÀº ±ÛÀÚµé ¾È¿¡ ¼¯¿© ÀÖ´Â ÅÇµéÀ» ½ºÆäÌ½º·Î È®Àå½ÃÅ°´Â ±â´ÉÀÌ ÀÖ´Ù.
(expandtab, :retab)

ÅÂ±× ½Ã½ºÅÛ:

VimÀº "tags" Ë-ÀÏÀÇ »öÀÏÀ» ÀÐ¾îµéÀÏ ËÄ, ´Ú¾ÇÇÑ ÅÄ±× °ü·Ä ¸í·Ë¾î¸¦
»ç¿ëÇØ¼-, ¿©·¯ È-ÀÏµé¿¡ À§Ä¡ÇÑ °Ë»öÆÐÅÏÀ» Ã£À» ¼ö ÀÖµµ·Ï Áö¿øÇÑ´Ù.

ÅØ½ºÆ® ¿ÀºêÁ§Æ®:
VimÀº ¸¹À° Á¾·ùÀÇ ¿ÀºêÁ§Æ® (´Ü¶ô, ¹®Àå, ´Ü¾î, ´ë¹®ÀÚ·Î ±¸¼ºµÈ ´Ü¾î
- ÀÌµé ¸ðµÎ °ø¹é¹®ÀÚ·Î ºÐ¸®µÈ´Ù) ¸¦ ÀÎ½ÄÇÑ´Ù. ¶ÇÇÑ, »ç¿ëÀÚ°¡ ÀÌµé
ÅØ½ºÆ® ¿ÀºêÁ§Æ®¿¡ ´ëÇÑ Á¤ÀÇ¸¦ ÇÒ ¼ö ÀÖµµ·Ï Áö¿øÇÑ´Ù.

¹®¹ý Ä®¶ó¸µ:
VimÀº ÇÁ·Î±×·¡¹Ö ¾ð¾î¿¡ µû¶ó, È-ÀÏ ¾ÈÀÇ ÅØ½ºÆ®¸¦ ¹®¹ý¿¡ ¸Â°Ô
Ä®¶ó¸µ ÇØÁØ´Ù. ¶ÇÇÑ, "¾ð¾î" ("¹®¹ý") ¿¡ ´ëÇÑ »ç¿ëÀÚ Á¤ÀÇ¸¦ ÇÒ ¼ö
ÀÖµµ·Ï Áö¿øÇØÁØ´Ù.

VimÀº ¹®¹ýÀû Ä®¶ó¸µÀ» À§ÇØ¼- 200¿©°³ ÀÌ»óÀÇ ¹®¹ýÈ-ÀÏÀ» Á¦°øÇÑ´Ù.
Áö¿øÇÏ´Â °ÍÀ¸·Î´Â, ÀÏ¹ÝÀû ÇÁ·Î±×·¡¹Ö ¾ð¾î (Ada, C, C++, Eiffel,
Fortran, Haskell, Java, Lisp, Modula, Pascal, Prolog, Python,
Scheme, Smalltalk, SQL, Verilog, VisualBasic), ¸¶Å©¾÷ ¾ð¾î (DocBook,
HTML, LaTeX, PostScript, SGML-LinuxDoc, TeX, WML, XML), ÇÁ·Î±×·¥
Ãâ·Â¹° (diff, man), ÇÁ·Î±×·¥ ¼³Á¤ È-ÀÏ (4DOS, Apache, autoconfig,
BibTeX, CSS, CVS, elm, IDL, LILO, pine, procmail, samba, slrn),
½© ½ºÅ©¸³Æ® ¹×¼³Á¤È-ÀÏ (shells:sh, bash, csh, ksh, zsh),
½ºÅ©¸³Æ® ¾ð¾î (awk, Perl, sed, yacc), ½Ã½ºÅÛ È-ÀÏ (printcap,
.Xdefaults)ÀÌ ÀÖ´Ù. ¹°·Đ Vim ¼³Á¤È-ÀÏ ¹× µµ¿ò» È-ÀÏµµ ¹®¹ýÀû
Ä®¶ó¸µÀÌ Áö¿øµÈ´Ù.

Æ¯¼öÇÑ ÄÚµå:
VimÀº PerlÀÌ³ª Tcl, Python °úÀÇ ÅëÇÕ±â´ÉÀ» ¿É¼ÇÀ¸·Î Á¦°øÇÑ´Ù.
VimÀº À©µµ¿ìÁî È¯°æ¿¡¼- OLE ÀÚµ¿È-¼-¹ö·Î¼- ÀÛµ¿ÇÒ ¼ö ÀÖ´Ù.
VimÀº X-windows ¸¦ Áö¿øÇÏ´Â ÄÚµå·Î ¼³Ä¡µÉ ¼ö ÀÖ°í,
¼³Á¤ °¡´ÉÇÑ ¸Þ´º ¹× ¶¿ì½º¿¡ ´ëÇÑ Áö¿øÀÌ Á¦øµÈ´Ù.
±× ¹Û¿¡µµ ¸¹ÀÌ. ÈÎ¾À ´õ ¸¹Àº ±â´ÉéÀÌ ÀÖ´Ù!

VimÀÇ À¥ È¨ÆäÀÌÁö´Â:
  http://www.vim.org/

VimÀÇ ±â´É¿¡ ´ëÇÏ¿© ´õ¿í »óº¼ÇÑ ±â´ÉÀ» º¸°í ½ÍÀ¸¸é,
´ÙÀ½ È¨ÆäÀÌÁö¸¦ ÂüÁ¶ÇÏ±â ¹Ù¶õ´Ù
  http://www.vim.org/why.html

¿ø ÀúÀÛÀÚ (¿µ¾îÆÇ): Sven Guckes guckes@vim.org
ÃÖÁ¾¼öÁ¤ÀÏ : Tue Oct 03 20:00:00 MET DST 2000

¹ø¿ªÀÚ (ÇÑ±ÛÆÇ): kildongi@hitel.net
ÃÖÁ¾¼öÁ¤ÀÏ : Sun Dec 31 01:47:00 KST 2000

"Co to jest Vim?"
Wyja¶nienie w sze¶ciu kilobajtach.

Vim ("Vi IMproved") jest "klonem" vi,
czyli programem podobnym do edytora
tekstu o nazwie "vi".

Vim pracuje w trybie tekstowym na ka¿dym
terminalu, choæ posiada równie¿ interfejs
graficzny, czyli menu i obs³ugê myszy.

Dostêpno¶æ:
Vim jest dostêpny na wiele platform i posiada du¿o
dodatkowych funkcji w porównaniu z Vi (zobacz
http://www.vim.org/doc/vi.diff.txt). Polecenia Vim-a s± prawie
ca³kowicie zgodne z poleceniami Vi - za wyj±tkiem Vi-owych b³êdów. ;-)

Systemy operacyjne:
Vim jest dostêpny na takie systemy jak:
AmigaOS, Atari MiNT, BeOS, DOS, MacOS, NextStep, OS/2, OSF, RiscOS,
SGI, UNIX, VMS, Win16 + Win32 (Windows95/98/00/NT) - a zw³aszcza na
FreeBSD i Linux. :-)

Prawa autorskie:
Prawa autorskie posiada g³ówny autor i szef projektu, Bram Moolenaar
<bram@vim.org>. Vim jest "oprogramowaniem charytatywnym", to znaczy i¿
zachêcamy Ciê do z³o¿enia datku na sieroty z Ugandy (zobacz ":help
uganda").

¬ród³a:
Vim jest programem na otwartych ¼ród³ach (OpenSource) i wszyscy
chêtni do pomocy przy jego rozwoju, s± zawsze mile widziani!

=== Cechy Vim-a

Edytor dla pocz±tkuj±cych - przyjazny dla u¿ytkownika:
Vim jest bardziej przyjazny dla pocz±tkuj±cych ni¿ Vi poniewa¿ posiada
wbudowany, obszerny system pomocy, polecenia "undo" i "redo" (cofnij i
ponów), które pozwalaj± na to, by nie przejmowaæ siê pomy³kami. Jego
dodatkowym atutem jest obs³uga myszy oraz konfigurowalne menu i pasek
narzêdziowy (GUI).

Kody znakowe i terminale:
Vim obs³uguje stronê kodow± iso-latin1 i
korzysta z bazy terminali termcap. (oryginalny Vi ma z tym problemy.)

Znaki narodowe i jêzyki:
Vim pozwala na pisanie tekstów z prawa na lewo (jak ma to miejsce w
jêzykach Arabskim, Perskim czy Hebrajskim) oraz zawieraj±cych znaki
wielobajtowe - tzn. reprezentowane przez wiêcej ni¿ jeden bajt (jak
np. w jêzyku Chiñskim, Japoñskim czy Koreañskim). Innymi s³owy, Vim
obs³uguje teksty pisane w UTF-8 i Unikodzie.

Formatowanie tekstu i tryb wizualny (visual mode):
U¿ywaj±c Vim-a, mo¿na zaznaczaæ tekst w sposób "widzialny" (z
pod¶wietleniem zaznaczonego tekstu), a nastêpnie dokonaæ na nim takich
operacji jak: kopiowanie, usuwanie, zastêpowanie, przesuniêcie w lewo
lub w prawo, zamiana wielko¶ci liter, czy te¿ formatowanie z
zachowaniem istniej±cych akapitów.
Vim umo¿liwia równie¿ zaznaczanie i operacje na prostok±tnych blokach

tekstu.

Polecenia dokañczaj±ce tekst:
Vim posiada polecenia, które dokañczaj± to co wpisa³e¶ - zarówno komendy jak i nazwy plików czy pojedyncze s³owa.

Polecenia automatyczne:
Vim posiada równie¿ polecenia pozwalaj±ce na automatyczn± reakcjê na dane zdarzenia ("autocommands"), np. na automatyczne rozpakowanie skompresowanego pliku.

Wstawianie umlaut-ów:
Vim pozwala na wstawianie znaków specjalnych poprzez kombinacjê dwóch klawiszy (np. kombinacja " oraz a daje w wyniku ä), oraz pozwala na zdefiniowanie w³asnych kombinacji.

Wykrywanie formatu pliku oraz konwersja:
Vim automatycznie rozpoznaje typ edytowanego pliku (DOS, Mac, Unix) oraz umo¿liwia zapisanie go w dowolnym innym. Zatem nie ma ju¿ potrzeby u¿ywania unix2dos na Windows-ach!

Historia:
Vim posiada historiê poleceñ oraz wyszukiwañ. Mo¿esz zatem przywo³aæ poprzednie polecenie lub wyszukiwany ci±g znaków i zmieniæ go.

Nagrywanie makr:
Vim umo¿liwia zarejestrowanie procesu edycji (wszystkich wpisywanych poleceñ), w celu pó¼niejszego wykorzystania przy zadaniach powtarzalnych.

Ograniczenia pamiêci:
Vim ma znacznie wy¿sze limity pamiêci przydzielanej dla poszczególnych linii i buforów ni¿ oryginalny Vi.

Wiele buforów i podzia³ ekranu:
Vim pozwala na edycjê wielu buforów, oraz na podzia³ ekranu na kilka mniejszych okien (zarówno w pionie jak i w poziomie), co umo¿liwia równoczesne ogl±danie wielu plików, lub wielu ró¿nych czê¶ci tego samego pliku.

Zwielokrotnianie poleceñ:
Vim umo¿liwia zwielokrotnianie dzia³ania poleceñ (poprzez tzw. "number prefix"). Obejmuje ono wiêkszy zakres komend, ni¿ ma to miejsce w Vi.

Pliki typu "runtime" (pliki pomocy oraz definiuj±ce sk³adniê):
[S± to dodatkowe pliki, u¿ywane w trakcie gdy program jest uruchomiony, ale nie zawieraj± kodu, który musi byæ skompilowany i zlinkowany.]
Vim-5.7 zawiera 70 plików pomocy (oko³o 2080K tekstu) opisuj±cych polecenia, opcje, oraz porady na temat konfiguracji i edycji.
(Vim-6.0x [010311]: 85 plików, ok. 2796K tekstu). Niektóre pliki zawieraj± opis wersji Vim-a na poszczególne systemy operacyjne. [010311]

Skrypty:
Vim posiada wbudowany jêzyk skryptowy, który mo¿e s³u¿yæ do jego ³atwej rozbudowy.

Wyszukiwanie z przesuniêciem:
Vim umo¿liwia przesuniêcia w poleceniach wyszukuj±cych. Oznacza to, umieszczenie kursora *za* znalezionym tekstem.

Zapis sesji:
Vim umo¿liwia zapis informacji o danej sesji do pliku ("viminfo").
Mo¿e on byæ u¿yty w nastêpnej sesji do odczytania listy buforów,
znaczników w plikach, rejestrów, historii poleceñ i wyszukiwañ.

Zastêpowanie znaków tabulacji:
Vim potrafi zastêpowaæ znaki tabulacji w tek¶cie odpowiedni± liczb±
spacji.

System Tag-ów:
Vim potrafi korzystaæ z plików indeksowych typu "tags" w celu
odszukania tekstu w plikach, oraz posiada wiele poleceñ z nich
korzystaj±cych.

Czê¶ci tekstu:
Vim rozpoznaje wiele czê¶ci tekstu, takich jak akapity, zdania, s³owa
i S£OWA - zarówno z otaczaj±cymi je bia³ymi znakami jak i bez nich.
Pozwala równie¿ na zmianê ich definicji.

Kolorowanie sk³adni:
Vim wy¶wietla tekst koloruj±c go - zgodnie z jego "jêzykiem
(programowania)". Mo¿esz samodzielnie zdefiniowaæ "jêzyk" (czyli jego
sk³adniê).

Vim zawiera ponad 200 plików definiuj±cych sk³adniê do kolorowania
tekstu w powszechnie znanych jêzykach programowania (Ada, C, C++, Eiffel,
Fortran, Haskell, Java, Lisp, Modula, Pascal, Prolog, Python,
Scheme, Smalltalk, SQL, Verilog, VisualBasic),
jêzykach programów matematycznych (Maple, Matlab, Mathematica, SAS),
jêzykach formatowania tekstu (DocBook, HTML, LaTeX, PostScript,
SGML-LinuxDoc, TeX, WML, XML), plikach wynikowych (diff,man), plikach
konfiguracyjnych do ró¿nych programów (4DOS, Apache, autoconfig,
BibTeX, CSS, CVS, elm, IDL, LILO, pine, procmail, samba, slrn),
skryptach pow³oki (sh, bash, csh, ksh, zsh), jêzykach skryptowych
(awk, Perl, sed, yacc), plikach systemowych (printcap, .Xdefaults)
i oczywi¶cie w plikach pomocy Vim-a.

Dodatki specjalne:
Vim opcjonalnie posiada zintegrowane ¶rodowiska do pisania w takich
jêzykach jak Perl, Tcl i Python. Vim mo¿e pe³niæ rolê serwera OLE pod
Windows. Vim mo¿e byæ równie¿ zainstalowany z obs³ug± X-Windows,
w³±czaj±c w to konfigurowalne menu i obs³ugê myszy.
Vim mo¿e jeszcze du¿o innych rzeczy. Naprawdê du¿o!

=== Linki

Strona WWW Vim-a:
  http://www.vim.org/

Je¶li chcesz poznaæ bardziej szczegó³owy
opis mo¿liwo¶ci Vim-a, zajrzyj na stronê:
  http://www.vim.org/why.html

=== Autor i t³umacz

Tekst oryginalny: Sven Guckes guckes@vim.org
Ostatnia aktualizacja: Mon Mar 12 07:00:00 MET 2001

Przet³umaczy³: Miko³aj Sitarz mik@fatcat.ftj.agh.edu.pl

Ostatnia aktualizacja: Thu Mar 22 20:01:31 CET 2001

vim: tw=70

"O que é Vim?"
Uma explicação em seis kilobytes.

Vim ("Vi IMproved") é um "clone do vi", isto é,
um programa similar ao editor de texto "vi".

O Vim funciona em modo texto em qualquer terminal,
mas também tem uma interface gráfica com o usuário,
isto é, menus e suporte para o mouse.

Disponibilidade:
O Vim está disponível para muitas plataformas e tem muitas características
adicionais comparado ao Vi. (veja http://www.vim.org/doc/vi.diff.txt)
Vim é compatível com quase todos os comandos do Vi - exceto com bugs do Vi. :-)

Sistemas Operacionais:
O Vim está disponível para muitos sistemas: AmigaOS, Atari MiNT, BeOS, DOS,
MacOS, NextStep, OS/2, OSF, RiscOS, SGI, UNIX, VMS, Win16 + Win32
(Windows95/98/00/NT) - e especialmente FreeBSD e Linux. :-)

Copyright (Direitos Autorais):
O copyright está nas mãos do autor principal
e mantenedor, Bram Moolenaar <bram@vim.org>.
O Vim é "charity-ware", isto é, você é incentivado a fazer
uma doação para órfãos na Uganda (veja ":help uganda").

Fonte:
O Vim é OpenSource e todos são bem-vindos para ajudar a melhorá-lo!

=== Características

Editor do Principiante - Amigável para o usuário:
Vim é muito mais fácil para principiantes do que o Vi por causa da Ajuda Online
extensiva, comandos "undo" e "redo" (não tem importância os erros - simplesmente
use undo+redo!), suporte para o mouse, ícones e menus configuráveis (GUI).

Códigos de caracteres e Terminais:
O Vim tem suporte para o conjunto de caracteres iso-latin1 e para termcap.
(O Vi tem problemas com este.)

Caracteres e Línguas:
O Vim suporta edição da direita-para-esquerda (por exemplo com Árabe, Farsi,
Hebreu), e textos multi-byte, isto é, linguagens com caracteres gráficos
representados por mais de um "byte", como o Chinês, Japonês, Coreano (Hangul),
(Técnicamente falando, Vim suporta texto em UTF-8 e Unicode.)

Formatando Texto e Modo Visual:
Com Vim você pode selecionar texto "visualmente" (com destaque) antes que você
"opere" nele, por exemplo copiar, apagar, substituir, mover para esquerda ou
direita, mudar as letras ou formatar o texto preservando a indentação.
O Vim permite também a seleção e operações em blocos de texto retangulares.

Comandos de Completação:
O Vim tem comandos que completam sua entrada de dados -
seja com comandos, nomes de arquivos, e palavras.

Comandos Automáticos:
O Vim tem também "autocomandos" para a execução automática de comandos
(por exemplo descompressão automática de arquivos comprimidos).

Entrada de Caracteres Especiais (Digraphs):
O Vim permite que você incorpore caracteres especiais por uma combinação
de dois caracteres (por exemplo a combinação de " e a produz um ä) -
e permite também que você defina outras combinações.

Detecção e conversão de Formatos de Arquivo:
O Vim reconhece automaticamente os tipos de arquivos (DOS, Mac, Unix)
e também deixa você salvá-lo em algum outro formato -
não há mais necessidade para unix2dos em Windows!

História:
O Vim tem uma "história" para comandos e buscas, assim você pode
rever comandos anteriores ou buscar padrão para editá-los.

Gravando Macro:
O Vim permite "gravar" sua edição para reusar em tarefas repetitivas.

Limites de Memória:
O Vim tem limites de memória muito mais elevados para comprimento de linha
e tamanho de buffer do que o Vi.

Buffers Múltiplos e Tela Dividida:
O Vim permite edição de buffers múltiplos e você pode dividir a
tela em muitas sub-janelas (horizontalmente e verticalmente),
assim você pode ver muitos arquivos ou muitas partes de algums arquivos.

Prefixo Numérico aos comandos:
O Vim permite um prefixo numérico para mais
comandos do que com Vi (por exemplo para "put").

Arquivos Runtime (Arquivos de Ajuda e Sintaxe):
O Vim vem com 70 arquivos de ajuda em vários aspectos da edição;
alguns textos são especificamente para uso em algum sistema operacional.

Scripting:
O Vim tem uma linguagem de script interna para extensão fácil.

Deslocamento de Busca:
O Vim permite deslocamentos para comandos de busca, assim você
coloca o cursor *após* o texto encontrado.

Recuperação da Sessão:
O Vim permite armazenar informação de uma sessão de edição em um arquivo
("viminfo") que os permite usar com a próxima sessão de edição, por exemplo
lista de buffer, marcas de arquivos, registros, comandos e história de buscas.

Expansão da Tabulação:
Vim pode expandir as tabulações dentro do texto com espaços (expandtab, :retab).

Sistema de Etiqueta (Tag):
O Vim oferece para encontrar texto em arquivos usando um índice com "tags"
junto com muitas pilhas de comandos.

Objetos de Texto:
Vim conhece mais objetos de texto (parágrafos, sentenças, palavras e PALAVRAS -
todas com ou sem espaço em branco em volta) e permite configurar
a definição para estes objetos.

Coloração da Sintaxe:
O Vim mostra texto em cor - de acordo com sua "linguagem (de programação)".

Você mesmo pode definir a ("sintaxe") "da linguagem" dos arquivos.

O Vim vem com 200+ arquivos de sintaxe para a coloração de texto em
linguagens de programação comuns (Ada, C, C++, Eiffel, Fortran,
Haskell, Java, Lisp, Modula, Pascal, Prolog, Python,
Scheme, Smalltalk, SQL, Verilog, VisualBasic),
programas matemáticos (Maple, Matlab, Mathematica, SAS), texto de marcação
(DocBook, HTML, LaTeX, PostScript, SGML-LinuxDoc, TeX, WML, XML),
saída de programa (diff, man), arquivos de configuração (4DOS, Apache,
autoconfig, BibTeX, CSS, CVS, elm, IDL, LILO, pine, procmail, samba, slrn),
shell scripts e configuração (shells: sh, bash, csh, ksh, zsh),
linguagens de script (awk, Perl, sed, yacc)
arquivos de sistema (printcap, .Xdefaults) e
é claro para o Vim e seus textos de ajuda.

Código Especial:
O Vim tem integração opcional com Perl, Tcl e Python.
O Vim pode atuar como um servidor de automatização OLE sob o Windows.
O Vim pode também ser instalado com código para X-Windows,
adicionando menus configuráveis e suporte para o mouse.
E mais. Muito mais!

HomePage do Wim na WWW:
  http://www.vim.org/

Para uma descrição mais elaborada
de características do Vim veja a página
  http://www.vim.org/why.html

Escrito por: Sven Guckes guckes@vim.org (Inglês)
Última atualização: Tue Oct 03 20:00:00 MET DST 2000

Traduzido em Português por: Douglas Santos dsantos@inf.furb.br
Última atualização: Fri Jan 12 04:39:50 BRST 2001

Ce este Vim?
O descriere în 6 Ko

Vim ("Vi IMproved", adicã "Vi ÎMbunãtãþit") este o clonã Vi, un program asemãnãtor editorului de text "Vi".

Vim funcþioneazã în mod text pe orice tip de terminal, însã, deasemenea, dispune ºi de o interfaþã graficã, avînd meniuri ºi suport pentru mouse.

Disponibilitate:
Vim este disponibil pentru nenumãrate platforme ºi are multe caracteristici în plus, comparativ cu Vi. (vezi http://www.vim.org/doc/vi.diff.txt)
În ceea ce priveste comenzile, intre Vim ºi Vi existã o mare compatibilitate (cu excepþia bug-urilor proprii Vi-ului ;-)

Sisteme de operare:
Vim este disponibil pentru aproape orice sistem: AmigaOS, Atari MiNT, BeOS, DOS, MacOS, NextStep, OS/2, OSF, RiscOS, SGI, UNIX, VMS, Win16 + Win32 (Windows95/98/00/NT) - ºi, în mod special, pentru FreeBSD ºi Linux. :-)

Copyright:
Copyright-ul aparþine principalului sãu autor, Braam Moolenaar <bram@vim.org>.
Vim este un program de caritate ("charity-ware"), adicã vã îndeamnã sã faceþi donaþii orfanilor din Uganda (vezi ":help uganda").

Sursa:
Vim este OpenSource ºi oricine este binevenit sã ajute la îmbunãtãþirea lui!

=== Caracteristici

Editor pentru începãtori - destul de pietenos cu cel care-l foloseºte:
Vim este mult mai uºor pentru începãtori decît Vi dacã este sã þinem cont de help-ul online, de comenzile "undo" ºi "redo" (comenzi prin care un text poate fi refãcut în caz cã este editat greºit), de posibilitatea utilizãrii mouse-ului, de icon-uri sau de existenþa unui meniu extrem de configurabil.

Coduri de caractere ºi terminale:
Vim are suport pentru setul de caractere iso-latin1 ºi pentru termcap. Problema diacriticelor este rezolvatã! Puteþi scrie texte în româneºte utilizînd fonturi din setul de caractere iso-latin2 în urma unei mapãri corespunzãtoate a tastaturii dv. (vezi :help mapping).

Flexibilitate lingvisticã:
Vim permite culegerea textului de la dreapta la stînga (destul de necesar pentru a scrie, spre exemplu, în arabã, persanã sau ebraicã) ºi chiar culegerea de text multi-octet, adicã text cu caractere grafice reprezentînd mai mult de un octet, cum sînt cele proprii limbilor chinezã, japonezã sau coreanã. Tehnic vorbind, Vim suportã text scris în UTF-8 ºi Unicode.

Formatarea textului ºi modul vizual:
Cu Vim puteþi selecþiona "vizual" o parte dintr-un text (marcat în mod diferit de restul textului) ºi sã executaþi diferite "operaþii" asupra acestuia, cum sînt cele de copiere, mutare, substituire, poziþionare la stînga sau la dreapta, capitalizarea literelor formatarea lui fãrã sã afectaþi indentarea. Acelea ºi operaþii le puteþi aplica unui bloc dreptunghiular de text.

Comenzi de completare:
Vim dispune de posibilitatea de a completa în mod automat numele comenzilor, directoarelor, fiºierelor ºi, încã posibilitatea de a completa în mod automat cuvintele sau sintagmele cheie din help-ul online.

Comenzi automate:
Vim oferã, deasemenea, posibilitatea de execuþie automatã a comenzilor (spre exemplu: compresia ºi decompresia automata a fiºierelor).

Introducerea caracterelor speciale (Digraphs):
Vim vã permite sã introduceþi într-un text caractere speciale folosindu-vã de combinaþia a douã caractere (cum ar fi ~ ºi a care rezultã ã) ºi încã vã permite sã definiþi o altã combinaþie asemãnãtoare.

Detectarea ºi conversia formatului fiºierelor:
Vim recunoaºte în mod automat tipul fiºierelor (DOS, Mac, Unix) ºi vã permite salvarea unui fiºier în formate diferite. Nu mai aveþi nevoie de programe de conversie gen unix2dos.

Istoric:
Vim reþine istoricul comenzilor ºi cãutãrilor. Puteþi sã reexecutaþi o comandã anterioarã ºi sã reapelaþi o cãutare.

Crearea macrocomenzilor:
Vim vã permite "înregistrarea" unei serii de acþiuni pentru a le putea executa din nou atunci cînd veþi avea de fãcut acelaºi lucru.

Limite de memorie:
Vim are o limitã de memorie mai mare, atît pentru lungimea unei linii, cît ºi pentru mãrimea unui tampon de memorie; --asta în comparaþie cu Vi.

Tampoanele de memorie ºi împãrþirea ecranului.
Vin acceptã o editare specialã bazata pe tampoane de memorie multiple. În plus, exitã posibilitatea împarþirii ecranului (pe orizontalã ºi verticalã) în nenumãrate ferestre; prin urmare, pot fi editate simultan mai multe fiºiere ºi diferite bucãti de text dintr-un fiºier.

Prefix numeric pentru comenzi:
Vim permite un prefix numeric pentru mult mai multe comenzi decît Vi (spre exemplu, pentru comanda "put").

Fiºierele "Runtime" (fiºierele help ºi cele de sintaxã):
Vim este distribuit împreunã cu 70 de fiºiere pentru help, acutualizate cu fiecare versiune a Vim-ului. Cîteva dintre acestea raspund cerinþelor de utilzare a Vim-ului pe diferite sisteme de operare.

Limajul de scripting:
Vim are un limbaj de scripting incorporat pentru a fi mai uºor extins.

Facilitate pentru operaþiile de cãutare:
Vim dispune de o facilitate extrem de folositoare în operaþile de cãutare: pur ºi simplu, puteþi cãuta în întreg textul cuvîntul aflat sub cursor, utilizînd o singurã comandã.

Recuperarea unei sesiuni de lucru:
Vim permite salvarea informaþiilor unei sesiuni de lucru într-un fiºier ("viminfo') care care face posibilã folosirea acestor informaþii într-o altã sesiune; sînt salvate lista tampoanelor de memorie, marcãrile dintr-un fiºier, registrele, istoricul comenzilor ºi cãutãrilor.

Extensia tabulaturii:
Vim permite expandarea tab-urilor dintr-un text folosind caractere spaþiu (expandtab, :retab).

Sistemul de etichete (Tags):
Vim vã permite sã gãsiþi un text într-un fiºier oarecare utilizînd un indice de etichete (tags); referitor la aceasta, Vim are multe alte comenzi prin care se poate manipula un astfel de indice de etichete.

Obiecte de tip text:
Vim recunoaºte multe obiecte de tip text (paragrafe, propoziþii, cuvinte ºi CUVINTE --toate cu sau fãrã spaþiu alãturat) ºi permite configurarea definiþiei fiecãruia dintre aceste obiecte.

Colororarea sintacticã:
Vim permite redarea textului în diferite culori în funcþie de sintaxa "limbajului (de programare)" utilizat. Dv. înºivã puteþi defini "limbajul" ("sintaxa") fiºierelor pe care le folosiþi.

Vim este distribuit cu peste 200 de fiºiere pentru colorarea sintacticã a textului scris în diferite limbaje, cum ar fi limbaje le de programare comune (Ada, C, C++, Eiffel, Fortran, Haskell, Java, Lisp, Modula, Pascal, Prolog, Pyton, Scheme, Smalltalk, SQL, Verilog, VisualBasic), limbajele proprii unor programe de matematicã (Maple, Matlab, Mathematica, SAS), limbajele de marcare specificã a textului, (DocBook, HTML, LaTeX, PostScript, SGML-LinuxDoc, TeX, WML, XML), limbajele programelor de afiºare (diff, man), limbajele folosite în fiºierele de setare ale anumitor programe (4DOS, Apache, autoconfig, BibTeX, CSS, CVS, elm, IDL, LILO, pine, procmail, samba, slrn), limbajele scripturilor de shell ºi setup (shells: sh, bash, csh, ksh, zsh), limbajele de script (awk, Perl, sed, yacc), cele ale sintemelor de fiºiere (printcap, .Xdefaults). Existã, desigur, o sintaxã coloratã atît pentru fiºierele help care însoþesc Vim-ul, cît ºi pentru fiºierele sale de configurare.

Cod special:
Vim poate fi integrat în mod opþional cu Perl, Tcl ºi Python. Vim poate servi drept server OLE sub Windows. Deasemenea, vim poate fi instalat cu suport pentru X-Windows avînd meniuri configurabile ºi suport pentru mouse.
ªi multe, multe alte facilitãþi!

Pagina oficialã a Vim-ului este:
     http://www.vim.org/

Pentru o descriere mai detaliatã despre Vim, vezi:
     http://www.vim.org/why.html

Text scris de: Sven Guckes guckes@vim.org (Englezã)
Ultima actualizare: Tue Oct 03 20:00:00 MET DST 2000

Text tradus în limba românã de: Nãstasie Nicolaie George george@cyberspace.ro
Data traducerii: 01 Feb 2001 07:15:07 (EET)

This text is in Koi8-r encoding. In case you can't read it,
or you see it corrupted, then please adjust your browser to it.
You may contact Bohdan Vlasyuk <bohdan@kivc.vstu.vinnica.ua>
if you still have problems.

þÔÏ ÔÁËÏÅ Vim ?"
ðÏÑÓÌÅÎÉÅ ÄÌÉÎÏÊ ÛÅÓÔØ ËÉÌÏÂÁÊÔ.

Vim ("Vi IMproved") ÜÔÏ "ËÌÏÎ vi", Ô.Å. ÐÒÏÇÒÁÍÍÁ, ÐÏÈÏÖÁÑ ÎÁ ÔÅËÓÔÏ×ÙÊ ÒÅÄÁËÔÏÒ "vi".

Vim ÒÁÂÏÔÁÅÔ × ÔÅËÓÔÏ×ÏÍ ÒÅÖÉÍÅ ÎÁ ÂÏÌØÛÉÎÓÔ×Å ÔÅÒÍÉÎÁÌÏ×, ÄÌÑ ÌÀÂÉÔÅÌÅÊ ÅÓÔØ É ÇÒÁÆÉÞÅÓËÉÊ ÉÎÔÅÒÆÅÊ, ÔÏ ÅÓÔØ ÍÅÎÀ É ÐÏÄÄÅÒÖËÁ ÍÙØÉ.

ðÒÉÇÏÄÎÏÓÔØ:
Vim ÄÏÓÕÐÅÎ ÄÌÑ ÍÎÏÖÅÓÔ×Á ÐÌÁÔÆÏÒÍ, É, ÐÏ ÓÒÁ×ÎÅÎÉÀ Ó Vi, ÉÍÅÅÔ ÍÎÏÇÏ ÄÏÐÏÎÉÔÅÌØÎÙÈ ÆÕÎËÃÉÊ (ÓÍ. http://www.vim.org/doc/vi.diff.txt) Vim ÓÏ×ÍÅÓÔÉÍ ÐÏÞÔÉ ÓÏ ×ÓÅÍÉ ËÏÍÁÎÄÁÍÉ Vi - ËÒÏÍÅ ÔÅÈ, ËÏÔÏÒÙÅ ÎÅ ÒÁÂÏÔÁÀÔ. ;-)

ïÐÅÒÁÃÉÏÎÎÙÅ ÓÉÓÔÅÍÙ:
÷Ù ÍÏÖÅÔÅ ÉÓÐÏÌØÚÏ×ÁÔØ Vim ÎÁ ÍÎÏÇÉÈ ïó: AmigaOS, Atari MiNT, BeOS, DOS, MacOS, NextStep, OS/2, OSF, RiscOS, SGI, UNIX, VMS, Win16 + Win32 (Windows95/98/00/NT) - Á ÏÓÏÂÅÎÎÏ FreeBSD É Linux. :-)

ðÒÁ×Á:
ðÒÁ×Á ÎÁ ÐÒÏÇÒÁÍÍÕ ÐÒÉÎÁÄÌÅÖÁÔ ÇÌÁ×ÎÏÍÕ Á×ÔÏÒÕ É ËÏÌÒÄÉÎÁÔÏÒÕ, Bram Moolenaar <bram@vim.org>. Vim Ñ×ÌÑÅÔÓÑ "ÂÌÁÇÏÔ×ÏÒÉÔÅÌØÎÏÊ ÐÒÏÇÒÁÍÍÏÊ", Ô.Å. ×Ù ÍÏÖÅÔÅ ×ÙÒÁÚÉÔØ ÂÌÁÇÏÄÁÒÎÏÓÔØ ÄÅÌÁÑ ×ÚÏÓ ÓÉÒÏÔÁÍ õÇÁÎÄÙ (ÓÍ. ":help uganda").

éÓÈÏÄÎÙÊ ËÏÄ:
Vim ÜÔÏ ÐÒÏÇÒÁÍÍÁ Ó ÏÔËÒÙÔÙÍ ËÏÄÏÍ, É ×ÁÛÁ ÐÍÍÝØ × ÕÌÕÞÛÅÎÉÉ Vim'Á ÂÕÄÅÔ ÐÒÉÎÑÔÁ Ó ÒÁÓÐÒÏÓÔÒÁÎÙÍÉ ÏÂßÑÔØÑÍÉ!

=== æÕÎËÃÉÏÎÁÌØÎÏÓÔØ

òÅÄÁËÔÏÒ ÄÌÑ ÎÁÞÉÎÁÀÝÉÈ - "ìÀÂÉÍ ðÏÌØÚÏ×ÁÔÅÌÑ":
Vim ÎÁÍÎÏÇÏ ÐÒÏÝÅ ÄÌÑ ÎÁÞÉÎÁÀÝÉÈ ÞÅÍ Vi ÂÌÁÇÏÄÁÒÑ ÉÓÞÅÒÐÙ×ÁÀÝÅÊ Online ÐÍÍÏÝÉ, ËÏÍÁÎÄÙ "undo" É "redo" (ÏÛÉÂËÁ? ÎÅ ÂÅÄÁ - ÉÓÐÌØÚÕÊÔÅ undo É redo!), ÐÏÄÄÅÒÖËÁ ÍÙÛÉ, ËÏÎÆÉÇÕÒÉÒÕÅÍÙÈ ÍÅÎÀ É ÉËÏÎÏË (GUI).

ôÅÒÍÉÎÁÌÙ É ËÏÄÙ ÓÉÍ×ÌÌ×:
÷ Vim ÓÕÝÅÓÔ×ÕÅÔ ÐÏÄÄÅÒÖËÁ iso-latin1 É termcap. (÷ "Vanilla Vi" ÜÔÏ ÎÅ ÒÁÂÏÔÁÅÔ.)

óÉÍ×ÌÌÙ É ÑÚÙËÉ:
Vim ÐÏÄÄÅÒÖÉ×ÁÅÔ ÒÅÄÁËÔÉÒÏ×ÁÎÉÅ "ÓÐÒÁ×Á ÎÁÌÅ×Ï", (ÎÁÐÒÉÍÅÒ, ÄÌÑ áÒÁÂÓËÏÇÏ, ðÅÒÓËÏÇÏ ÑÚÙËÏ×, é×ÒÉÔÁ), É ÍÎÏÇÏÂÁÊÔÏ×ÙÅ ÓÉÍ×ÌÌÙ, Ô.Å. ÑÚÙËÉ × ËÏÔÏÒÙÈ ÉÓÐÌÏÚÕÀÔÓÑ ÉÅÒÌÇÌÉÆÙ, ÎÁÐÒÉÍÅÒ ëÉÔÁÊÓËÉÊ, ñÐÎÓËÉÊ, ëÏÒÅÊÓËÉÊ (Hangul), (çÏ×ÏÒÑ ÔÅÈÎÉÞÅÓËÉÍ ÑÚÙËÏÍ, Vim ÐÏÄÄÅÒÖÉ×ÁÅÔ ÔÅËÓÔÙ × UTF-8 É Unicode.)

æÏÒÍÁÔÉÒÏ×ÁÎÉÅ ÔÅËÓÔÁ É ×ÉÚÕÁÌØÎÙÊ ÒÅÖÉÍ:
éÓÐÌØÚÕÑ Vim ÍÏÖÎÏ ×ÙÂÒÁÔØ ÔÅËÓÔ "×ÉÚÕÁÌØÎÏ" (Ó ×ÙÄÅÌÅÎÉÍ) ÐÅÒÄ ÔÅÍ ËÁË ÏÂÒÁÂÁÔÙ×ÁÔØ ÅÇÏ (ËÏÐÉÒÏ×ÁÔØ, ÕÄÁÌÑÔØ ÓÉÍ×ÌÌÙ, ÉÚÍÅÎÑÔØ ÏÓÓÕÐ, ÉÚÍÅÎÑÔØ ÒÅÇÉÓÒ ÂËÉ× ÉÌÉ ÆÒÍÁÔÉÒÏ×ÁÔØ ÔÅËÓÔ). Vim ÐÚ×ÌÌÑÔ ÒÁÂÏÔÁÔØ É c ÐÒÑÍÏÕÇÌØÎÙÍÉ ×ÙÄÌÅÎÑÍÉ.

ëÏÍÁÎÄÙ ÄÏÐÏÌÎÅÎÉÑ:
Vim ÍÏÖÅÔ ÄÏÐÏÌÎÑÔØ ×ÁÛ ××ÏÄ - ËÏÍÁÎÄÏÊ, ÉÍÅÎÅÍ ÆÁÊÌÁ, ÉÌÉ ÖÅ ÓÌÏ×ÏÍ
ÉÚ ÔÅËÓÔÁ.

á×ÔÏËÏÍÍÁÎÄÙ:
íÏÖÎÏ ÉÓÐÏÌØÚÏ×ÁÔØ "Á×ÔÏËÏÍÁÎÄÙ" ÄÌÑ Á×ÔÏÍÁÔÉÞÅÓËÏÇÏ ÉÓÐÏÌÎÅÎÉÑ
ÄÅÉÓÔ×ÉÊ (ÎÁÐÒÉÍÅÒ, ÄÌÑ Á×ÔÏÍÁÔÉÞÅÓËÏÇÏ ÞÔÅÎÉÑ ÓÖÁÔÙÈ ÆÁÊÌÏ×).

÷×ÏÄ ÄÉÇÒÁÆÏ×:
Vim ÐÏÚ×ÏÌÑÅÔ ××ÏÄÉÔØ ÓÐÅÃÉÁÌØÎÙÅ ÓÉÍ×ÏÌÙ ËÏÍÂÉÎÁÃÉÅÊ Ä×ÕÈ ÓÉÍ×ÏÌÏ×, É
ÐÏÚ×ÏÌÑÅÔ ÏÐÒÄÄÌÑÔØ Ó×ÏÉ ËÏÍÂÉÎÁÃÉÉ. (ÎÁÐÒÉÍÅÒ ÐÏÓÌÅÄÏ×ÁÔÅÌØÎÏÅ
ÎÁÖÁÔÉÅ Å É : ÍÏÖÅÔ ÄÁ×ÁÔØ £)

ïÐÒÅÄÅÌÅÎÉÅ É ÉÚÍÅÎÅÎÉÅ ÆÏÒÍÁÔÁ ÆÁÊÌÏ×:
Vim Á×ÔÏÍÁÔÉÞÅÓËÉ ÒÁÓÐÚÎÁÅÔ ÔÉÐ ÆÁÉÌÁ (DOS, Mac, Unix) É ÐÏÚ×ÏÌÑÅÔ
ÓÏÈÒÁÎÑÔØ ÆÁÊÌÙ × ÌÀÂÏÍ ÉÚ ÜÔÉÈ ÆÏÒÍÁÔÏ× - ×ÁÍ ÂÏÌØÛÅ ÎÅ ÐÏÎÁÄÏÂÉÔÓÑ
unix2dos!

éÓÔÏÒÉÑ ××ÏÄÁ:
Vim ÐÏÚ×ÏÌÑÅÔ ×ÅÓÔÉ "ÉÓÔÏÒÉÀ" ËÏÍÁÎÄ É ÐÏÉÓËÏ×, ÔÁË ÞÔÏ ÍÏÖÎÏ
×ÙÐÏÎÑÔØ É ÒÅÄÁËÔÉÒÏ×ÁÔØ ÐÒÅÄÙÄÕÝÉÅ ËÏÍÁÎÄÙ É ÛÁÂÌÏÎÙ ÐÏÉÓÁ.

úÁÐÉÓØ ÍÁËÒÏÓ×:
Vim ÐÏÚ×ÏÌÑÅÔ "ÚÁÐÉÓÙ×ÁÔØ" ×ÁÛÉ ÄÅÊÓÔ×ÉÑ ÄÌÑ ×ÙÐÏÎÅÎÉÑ ÐÏ×ÔÏÑÀÝÉÈÓÑ
ÏÐÒÁÃÉÊ.

ïÇÒÁÎÉÞÅÎÉÑ ÐÁÍÑÔÉ:
Vim ÍÏÖÅÔ ×ÙÄÀÌÑÔØ ÐÁÍÑÔØ ÄÌÑ ÓÔÒÏË É ÂÕÆÅÒ×ÏÌØÛÉÈ, ÞÅÍ ÐÏÚ×ÏÌÑÅÔ
Vi.

îÅÓËÏÌØËÏ ÂÕÆÅÒ× É ÒÁÚÄÅÌÅÎÅ ÜËÒÁÎÁ:
Vim ÐÏÚ×ÏÌÑÅÔ ÒÅÄÁËÔÉÒ×ÁÔØ ÎÅÓËÏÌØËÏ ÂÕÆÅÒ×, É ×Ù ÍÏÖÅÔÅ ÒÁÚÂÉÔØ
ÜËÒÁÎ ÎÁ ÍÎÏÇÏ ÏËÏÎ (ËÁË ÇÌÒÉÚÎÔÁÌØÎÏ, ÔÁË É ×ÅÒÔÉËÁÌØÎÏ), ÔÁËÉÍ
ÏÂÒÁÚÏÍ, ×Ù ÍÏÖÅÔÅ ×ÉÄÅÔØ ÎÅÓËÏÌØËÏ ÆÁÊÌÏ×, ÉÌÉ ÎÅÓËÏÌØËÏ ÞÁÓÔÅÊ
ÏÄÎÏÇÏ ÆÁÊÌÁ.

ëÏÌÉÞÅÓÔ×ÅÎÎÙÊ ÐÒÅÆÉËÓ Ë ËÏÍÁÎÄÁÍ:
Vim ÐÏÚ×ÏÌÑÅÔ ÕËÁÚÙ×ÁÔØ ËÏÌÉÞÅÓÔ×Ï ÐÏ×ÔÏÒÅÎÉÊ ÄÌÑ ÂÏÌØÛÅÇÏ ËÏÌÉÞÅÓÔ×Á
ËÏÍÁÎÄ ÞÅÍ Vi (ÎÁÐÒÉÍÅÒ, ÄÌÑ "put").

÷ÏÓÐÍÁÇÁÔÅÌØÎÙÅ ÆÁÊÌÙ (ÆÁÊÌÙ ÐÍÏÝÉ É ÓÉÎÔÁËÓÉÓÁ):
[üÔÉ ÆÁÊÌÙ ÉÓÐÏÌØÚÕÔÓÑ ÔÏÌØËÏ ×Ï ×ÒÁÍÑ ×ÙÐÏÎÅÎÉÑ Vim'a É ÎÅ ÓÏÄÅÒÖÁÔ
ËÏÄÁ ËÏÔÏÒÙÊ ÎÁÄÏ ËÏÍÐÉÌÉÒÏ×ÁÔØ ÉÌÉ ËÏÍÐÏÎÏ×ÁÔØ.] ó Vim-5.7 ×Ù
ÐÏÎÕÞÉÔÅ 70 ÆÁÊÌÏ× ÐÏÍÝÉ (ÏËÌÏ 2080K ÔÅËÓÔÁ), ÏÐÉÓÙ×ÁÀÝÉÈ ËÏÍÁÎÄÙ,
ÏÐÃÉÉ, É ÓÏ×ÅÔÙ ÄÌÑ ÜÆÆËÔÉ×ÎÏÊ ÒÁÂÏÔÙ É ËÏÎÆÉÇÕÒÉÒÏ×ÁÎÉÑ. (Vim-6.0x:
85 ÆÁÌÏ×, ÏËÌÏ 2796K ÔÅËÓÔÁ). îÅËÏÔÏÒÙÅ ÆÁÊÌÙ ÏÐÉÓÙ×ÁÀÔ ÉÓÐÏÌØÚÏ×ÁÎÉÅ
Vim'Á ÎÁ ÓÐÅÃÉÆÉÞÎÙÈ ÏÐÀÒÁÉÏÎÙÈ ÓÉÓÔÅÍÁÈ.

óËÒÉÐÔÙ:
Vim ÓÏÄÅÒÖÉÔ ×ÓÔÒÏÅÎÎÙÊ ÓËÒÉÐÔÏ×ÙÊ ÑÚÙË ÄÌÑ ÐÒÏÓÏÇÏ ÄÏÂÁ×ÌÅÎÉÑ ÎÏ×ÙÈ
×ÏÚÍÏÖÎÏÓÔÅÊ.

óÄ×ÉÇ ÐÏÓÌÅ ÐÏÉÓËÁ:
Vim ÐÏÚ×ÏÌÑÅÔ ÕËÁÚ×ÁÔØ ÓÄ×ÉÇ ÄÌÑ ËÏÍÁÎÄ ÐÏÉÓËÁ, Ô.Å. ËÕÒÓÏÒ ÂÕÄÅÔ
ÒÁÓÐÌÏÖÅÎ *ÐÏÓÌÅ* ÎÁÊÄÅÎÎÏÇÏ ÔÅËÓÔÁ.

÷ÏÓÔÁÎÏ×ÌÅÎÉÅ ÓÅÁÎÓÏ× ÒÁÂÏÔÙ:
Vim ÐÏÚ×ÏÌÑÅÔ ÓÏÈÒÁÎÑÔØ ÉÎÆÒÏÍÁÃÉÀ Ï ÒÅÄÁËÔÉÒ×ÁÎÉÉ (ÓÐÉÓÏË ÂÕÆÅÒÏ×,
ÐÏÍÅÔË × ÆÁÊÌÁÈ, ÒÅÇÉÓÔÒÙ, ÉÓÔÏÒÉÀ ËÏÍÁÎÄ É ÐÏÉÓËÁ) × ÆÁÊÌ

("viminfo"), Ë ÐÒË ÒÏÅÄÒÁŸÅÏ ÓÅÄÄËÖËÓÏ×ÁÏËË ÙÖÖ ËÏÆÏÒÏÁÄËÁ ÌÏÖÏÏ
×ÏÓÓÔÁÎÏ×ÉÔØ.

úÁÍÅÎÁ ÓÉÍ×ÏÌÏ× ÔÁÂÕÌÑÃÉÉ:
Vim ÍÏÖÅÔ ÚÁÍÅÎÑÔØ ÓÉÍ×ÏÌÙ ÔÁÂÕÌÑÃÉÉ × ÔÅËÓÔÅ ÎÁ ÐÒÏÂÅÌÙ (expandtab,
:retab).

óÉÓÔÅÍÁ ÔÅÇÏ×:
Vim ÍÏÖÅÔ ÉÓÐÏÌØÚÏ×ÁÔØ ÓÉÓÔÅÍÕ "ÔÅÇÏ×" ÄÌÑ ÉÎÄÅËÓÉÒÏ×ÁÎÉÑ É ÐÏÉÓÊÁ ×
ÔÅËÓÔÅ, ÐÒÉ ÜÔÏÍ ÍÏÖÎÏ ÐÏÌØÚÏ×ÁÔØÓÑ "ÓÔÅÐÍÍ" ÐÏÓÝÅÎÎÙÈ ÔÅÇÏ×.

ïÂßÅËÔÙ × ÔÅËÓÔÅ:
Vim ÒÁÓÐÚÍÁÅÔ ÍÎÏÇÏ ÏÂßÅËÔÏ× × ÔÅËÓÔÅ (ÐÁÒÁÇÒÁÆÙ, ÐÒÅÄÌÏÖÅÎÉÑ, ÓÌÏ×Á,
óìï÷á - ×ËÌÀÞÁÑ ÉÌÉ ÎÅ ×ËÌÀÞÁÀ ÏËÒÕÖÁÀÝÉÅ ÐÒÏÂÅÌÙ) É ÐÏÚ×ÌÑÅÔ
ÉÚÍÅÎÑÔØ ÏÐÒÅÄÅÌÅÎÑ ÎÅËÎÔÏÒÙÈ ÏÂßÅËÔÏ×.

ðÏÄÓ×ÅÞÉ×ÁÎÉÅ ÓÉÎÔÁËÓÉÓÁ:
Vim ÐÏËÁÚÙ×ÁÅÔ ÔÅËÓÔ × Ã×ÅÔÅ - × ÓÏÏÔ×ÅÔÓÔ×ÉÉ Ó ÓÉÎÔÁËÓÉÓÏÍ. ÷Ù
ÍÏÖÅÔÅ ÏÐÒÅÄÅÌÑÔØ ÓÉÎÔÁËÓÉÓ ÆÁÊÌÁ ÓÁÍÉ.

äÌÑ Vim ÓÕÝÅÓÔ×ÕÅÔ ÂÏÌÅÅ Ä×ÕÈÓÏÔ ÆÁÊÌÏ× Ó ÏÐÉÓÁÎÉÅÍ ÓÉÎÔÁËÓÉÓÁ ÄÌÑ
ÞÁÓÔÏ ×ÓÔÒÅÞÁÀÝÉÈÓÑ ÑÚÙËÏ× ÐÒÏÇÒÁÍÍÉÒÏ×ÁÎÉÑ (Ada, C, C++, Eiffel,
Fortran, Haskell, Java, Lisp, Modula, Pascal, Prolog, Python, Scheme,
Smalltalk, SQL, Verilog, VisualBasic), ÐÒÏÇÒÁÍÍ ÄÌÑ ÍÁÔÅÍÁÔÉÞÅÓËÉÈ
×ÙÞÉÓÌÅÎÉÊ (Maple, Matlab, Mathematica, SAS), ÑÚÙËÏ× ÒÁÚÍÅÔËÉ
(DocBook, HTML, LaTeX, PostScript, SGML-LinuxDoc, TeX, WML, XML),
×Ù×ÏÄÁ ÐÒÏÇÒÁÍÍ (diff, man), ÆÁÊÌÙ ÕÓÔÁÎÏ×ÏË ÐÒÏÇÒÁÍÍ (4DOS, Apache,
autoconfig, BibTeX, CSS, CVS, elm, IDL, LILO, pine, procmail, samba,
slrn), ÓËÒÉÐÔÙ shell (sh, bash, csh, ksh, zsh), ÓËÒÉÐÔÏ×ÙÈ ÑÚÙËÏ×
(awk, Perl, sed, yacc), ÓÉÓÔÅÍÎÙÈ ÆÁÊÌÏ× (printcap, .Xdefaults), É,
ÅÓÔÅÓÔ×ÅÎÎÏ, ÓÉÎÔÁËÓÉÓÁ ÆÁÊÌÏ× ËÏÍÁÎÄ É ÐÏÍÏÝÉ ÄÌÑ Vim.

óÐÅÃÉÁÌØÎÙÊ ËÏÄ:
Vim ÍÏÖÅÔ ÉÎÔÅÇÒÉÒÏ×ÁÔØÓÑ Ó Perl, Tcl É Python. Vim ÍÏÖÅÔ ÉÇÒÁÔØ ÒÏÌØ
OLE automation server ÐÏÄ Windows. Vim ÍÏÖÅÔ ÂÙÔØ ÕÓÔÁÎÏ×ÌÅÎ Ó ËÏÄÏÍ
ÄÌÑ X-windows, ÐÏÚ×ÏÌÑÀÝÉÍ ËÏÎÆÉÇÕÒÉÒÏ×ÁÔØ ÍÅÎÀ É ÐÏÄÄÅÒÖÉ×ÁÀÝÉÍ ÍÙÛØ.

é ÂÏÌØÛÅ! îÁÍÎÏÇÏ ÂÏÌØÛÅ!!

"äÏÍÁÛÎÑÑ ÓÔÒÁÎÉÃÁ" Vim'Á:
    http://www.vim.org/

äÌÑ ÂÏÌÅ ÔÝÁÔÅÌØÎÏÇÏ ÏÐÉÓÁÎÉÑ Vim'Á ÓÍÏÔÒÉÔÅ:
    http://www.vim.org/why.html

X-Original-by: Sven Guckes guckes@vim.org
X-Translated-by: Bohdan Vlasyuk <bohdan@olymp.vinnica.ua
X-Edited-by: Vim
X-Thanks: Yuriy Brazhnyk <yvb@pisem.net>

"¿Qué es Vim?"
Una explicación en seis kilobytes.

Vim ("VI IMproved") es un "clon de VI", es decir, un programa similar al editor de textos "VI".

Vim no solo trabaja en modo de texto en cualquier terminal, sino que también tiene un interfaz gráfica para el usuario, es decir, menús y soporte para el ratón.

Disponibilidad:
Vim está disponible para muchas plataformas y tiene muchas características añadidas en comparación con VI. (véase http://www.vim.org/doc/vi.diff.txt) Vim es compatible con casi todos los comandos de VI - excepto con los errores (bugs) en VI. ;-)

Sistemas Operativos:
Vim está disponible para muchos sistemas: AmigaOS, Atari MiNT, BeOS, DOS, MacOS, NextStep, OS/2, OSF, RiscOS, SGI, UNIX, VMS, Win16 + Win32 (Windows95/98/00/NT) - y especialmente FreeBSD y Linux.

Copyright:
El copyright está en las manos del autor principal y mantenedor, Bram Moolenaar <bram@vim.org>. Vim es un "programa-de-caridad" ("charity-ware"), es decir que se sugiere que hagas una donación a los huérfanos en Uganda (véase ":help Uganda ").

Fuente:
Vim es OpenSource y todos son bienvenidos para ayudar a mejorarlo!

=== Características

Editor para un Principiante - Amigable para el usuario:
Vim es mucho más fácil para los principiantes que VI debido a la disponibilidad de ayuda en línea bien extensa, comandos para "deshacer" (undo) y "rehacer" (redo) (¡no te preocupes mucho con los errores - simplemente usa undo y redo!), soporte para el ratón y tambien iconos y menús configurables (GUI).

Códigos de caracteres y terminales:
Vim tiene soporte para el grupo de caracteres iso-latin1 y para el termcap. (El VI normal tiene problemas con esto.)

Caracteres y lenguajes:
Vim soport editar de derecha-a-izquierda (ej. con el Árabe, Farsi, Hebreo), y textos en multi-octeto, es decir, lenguajes con caracteres gráficos representados por más de un "octeto", por ejemplo Chino, Japonés, Coreano (Hangul), (técnicamente hablando, Vim soporta texto escrito en UTF-8 y Unicode.)

Formateo de texto y modo visual:
Con Vim usted puede seleccionar el texto "visualmente" (con resalte) antes de que usted "opere" en él, ej. copiar, remover, substituir, mover la posición a la izquierda o derecha, cambiar la capitalizació de las letras o el formato del texto incluso preservando la indentació del mismo. Vim permite también la selección y operaciones en bloques de texto rectangulares.

Comandos de Completación:
Vim tiene comandos que completan su entrada de información - sea con

comandos, nombres de fichero, o palabras.

Comandos Automáticos:
Vim también tiene "autocommands" para la ejecución
automática de los comandos (ej. decompresión automática
de ficheros comprimidos).

Entrada de Caracteres Especiales (Digraphs):
Vim permite que usted incorpore caracteres especiales usando una
combinación de dos caracteres (ej. la combinación de " y a
resulta en ä) - y permite que usted defina otras combinaciones
también.

Detección y Conversión de Formatos de Archivo:
Vim reconoce automáticamente el tipo de ficheros (DOS, mac, Unix) y
también le permite el guardar el archivo en cualquier otro formato -
¡no hay necesidad de usar unix2dos para usar en Windows nunca más!

Historia:
Vim tiene una "historia" para los comandos y las búsquedas,
así que usted puede llamar nuevamente los comandos o el patrón
de búsqueda anteriores para editarlos.

Grabación de Macro:
Vim permite "grabar" una serie de acciones de edición para
poder ejecutarlas nuevamente cuando se realizan tareas repetitivas.

Límites de la Memoria:
Vim tiene límites de memoria mucho más grandes para la longitud
de línea y el tamaño del almacenador intermediario (buffer) en
comparación con VI normal.

Almacenadores Intermediarios (Buffers) múltiples y Pantalla Dividida:
Vim permite corregir de múltiples almacenadores intermediarios y usted
puede partir la pantalla en muchas sub-ventanas (horizontal y verticalmente),
así que usted puede ver muchos ficheros o muchas partes de algunos
ficheros.

Prefijo Numérico a los Comandos:
Vim permite un prefijo numérico para más comandos que con VI
(ej. para el comando "put").

Ficheros Usados Durante Ejecución (Ficheros de Ayuda y de Sintaxis):
Vim viene con 70 ficheros de ayuda en varios aspectos de edición;
algunos textos están específicamente escritos para uso en
ciertos sistemas operativos.

Lenguaje de escritura:
Vim tiene un lenguaje de escritura incorporado para poder extenderlo
facilemente.

Desplazamiento en Operaciones de Búsqueda:
Vim permite el usar desplazamientos relativos para los comandos de la
búsqueda, así que se puede poner el cursor inmediatamente lugar
después del texto encontrado.

Recuperación de la Sesión:
Vim permite para salvar la información de una sesión de
edición en un fichero ("viminfo") lo cual permite que sean
usados en una subsecuente sesión de edición, ej. la lista de

almacenadores intermediarios, de las marcas de fichero, de los registros, comandos y de la historia de las búsquedas.

Extensión de la Tabulación:
Vim puede expandir las tabulaciones dentro del texto usando caracteres de espacio (expandtab,:retab).

Sistema de la Etiqueta (Tag):
Vim permite el encontrar texto en ficheros usando un índice con las "etiquetas" (tags) junto con muchos otros comandos que manipulan la lista de de etiquetas.

Objetos de Texto:
Vim sabe de muchos objetos de texto (párrafos, sentencias, palabras y PALABRAS - todas con y sin el espacio en blanco circundante) y permite el configurar la definición de cada uno de estos objetos.

Colorisación de la Sintaxis:
Vim muestra el texto en color - según su "lenguaje (de programación)". Usted mismo puede definir el "lenguaje" ("sintaxis") de los ficheros.

Vim viene con 200+ ficheros de sintaxis para la colorisación del texto en los lenguajes de programación comunes (Ada, C, C++, Eiffel, FORTRAN, Haskell, Java, lisp, Modula, PASCAL, prólogo, Python, esquema, palique, SQL, Verilog, VisualBasic), programas de matemáticas (arce, Matlab, Mathematica, SAS), texto que use marcado específico (DocBook, HTML, LaTeX, PostScript, SGML-LinuxDoc, TeX, WML, XML), retornos de programas (diff, hombre), ficheros de la configuració de programas (4DOS, Apache, autoconfig, BibTeX, CSS, CVS, olmo, IDL, LILO, pino, procmail, samba, el slrn), lenguajes de escritura del procesador de comandos (shell) y de configuración (shells: sh, golpe, csh, ksh, zsh), lenguajes de la escritura (awk, Perl, sed, yacc), ficheros de sistema (printcap, Xdefaults) y por supuesto para Vim y sus textos de ayuda.

Código Especial:
Vim tiene integración opcional con Perl, Tcl y Python.
Vim puede actuar como servidor para la automatización de OLE bajo Windows.
Vim se puede también instalarse con el código para soporte de X-Windows, agregando menús y ayuda configurables con el ratón.
Y más. ¡Mucho más!

Vim HomePage en la WWW:
http://www.vim.org /

Para una descripción más elaborada
de las características de Vim, vea la página
http://www.vim.org/why.html

Escrito por: Sven Guckes guckes@vim.org (Inglés)
Ultima actualización: Tue Oct 03 20:00:00 MET DST 2000

Traducido al Español por: Jesus M. Castagnetto jesusmc@scripps.edu
Ultima actualización: Mon Nov 13 16:50:40 PST 2000

"Vad är Vim?"
En beskrivning på sex kilobyte.

Vim ("Vi IMproved") är en "vi klon", dvs
ett program som liknar editorn "vi".

Vim fungerar i textmode på alla terminaler,
men erbjuder också ett grafiskt användargränssnitt
med menyer och stöd för användning av mus.

Tillgänglighet:
Vim finns för många plattformar och har åtskilliga nya funktioner
i förhållande till Vi. (se http://www.vim.org/doc/vi.diff.txt)
Vim stöder nästan alla Vi-kommandon - förutom Vi:s buggar. ;-)

Operativsystem:
Vim finns tillgängligt på många olika system: AmigaOS, Atari MiNT,
BeOS, DOS, MacOS, NextStep, OS/2, OSF, RiscOS, SGI, UNIX, VMS,
WIn16 + Win32 (Windows95/98/00/NT) - och särskilt FreeBSD och Linux.

Copyright:
Rättigheterna ägs av huvudupphovsmannen Bram Moolenaar <bram@vim.org>.
Vim är så kallad "charity-ware", vilket innebär att du som använder
programmet uppmanas donera pengar till föräldralösa barn i Uganda
(se ":help uganda").

Källkod:
Vim är OpenSource och alla som vill hjälpa till att göra
förbättringar till programmet välkomnas!

=== Features

Nybörjareditor - användarvänlighet:
Vim är ett mycket mera lättanvänt program än Vi på grund av
sin omfattande hjälpfunktionalitet, UNDO- och REDO-kommandon för
att underlätta när du råkar skriva fel, mus-support samt
konfigurerbara ikoner och menyer.

Teckenuppsättningar, språk och terminaler:
Vim har inbyggd support för teckenuppsättningarna iso-latin1 och termcap.
(Vanilla Vi har vissa problem med detta.) Vidare finns funktioner för att
skriva från höger till vänster (till exempel på arabiska, farsi eller
hebreiska), eller att skriva "multi-byte"-texter, dvs på språk med
grafiska tecken som representeras av mer än en byte. (Uttryckt mera
tekniskt; Vim har inbyggd support för UTF-8 och Unicode.)

Textformatering och Visual Mode:
Med Vim kan du markera text "visuellt" (med "highlighting") innan du
utför "operationer" på den, såsom Kopiera, Klipp ut, Ersätt, Skifta vänster
eller höger, ändra små bokstäver till stora eller tvärtom eller applicera
olika format inklusive möjligheten att då behålla indentering.
Vim tillåter även markering och operationer av godtyckliga rektangulära
textblock.

Kommandoradskomplettering:
Vim har funktioner som kompletterar din inmatning -
både när det gäller kommandon, filnamn och ord.

Automatiska kommandon:
Vim har också "autocommands" för att automatiskt exekvera olika

filterfunktioner på din text, till exempel att automatiskt packa
upp en komprimerad fil.

Digraph Input:
Vim låter dig skriva specialtecken genom kombinationer av två
tangenttryckningar (till exempel ger kombinationen av " och A
tecknet Ä, vilket möjligen inte är så intressant om du har
ett svenskt tangentbord, men du kan själv definiera kombinationer
för andra specialtecken.)

Filformat och konvertering mellan sådana:
Vim känner automatiskt igen olika typer av textfiler (DOS, Mac, Unix)
och har också funktioner för att konvertera filer dem emellan.
Detta gör att du inte längre behöver verktyg av typen unix2dos, etc.

History:
Vim har en "history"-funktion för kommandon och sökningar, så att du
kan återkalla tidigare givna inmatningar, eventuellt editera dessa
och sedan exekvera dem igen.

Makrofunktioner:
Vim tillåter dig att "spela in" din editering för att sedan "spela upp"
den igen om du gör repetitiva serier av kommandon.

Minnesbegränsningar:
Med Vim kan du använda mycket större buffertar och skriva mycket
längre textrader än vad som är möjligt med Vanilla Vi.

Multipla buffertar och skärmsplit:
Med Vim kan du samtidigt editera flera filer, exempelvis genom att
dela in skärmen i flera segment, såväl horisontellt som vertikalt.

Numeriska prefix till kommandon:
Vim tillåter prefix till många fler kommandon än Vi gör, exempelvis
till "put".

Runtime-filer (hjälp- och syntaxfiler):
Vim levereras med 70 hjälpfiler rörande olika aspekter på editering;
en del av texterna är specifika för vissa operativsystem.

Skript:
Vim har ett inbyggy skript-språk som ger möjlighet att enkelt
bygga ut funktionaliteten.

Sökoffsetter:
Vim tillåter offsetter in samband med sökning, så att du
exempelvis kan placera markören *efter* den funna texten.

Återställning av session:
Vim kan ställas in att automatiskt spara information om en editering
som återanvänds nästa gång man startar programmet. Denna information
inbegriper bland annat listor med buffertar, markeringar i filen samt
kommando- och sökningslistor.

Tabulatorexpansion:
Vim har funktioner för att expandera tabbar i texten till ett antal
mellanslag.

Taggar:
Vim har en funktionalitet som möjliggör sökning i textfiler genom

att man anger ett index med en "tag" tillsammans med flera olika
kommandon mot stackar.

Textobjekt:
Vim känner igen flera "textobjekt", såsom stycken, meningar och ord -
i samtliga fall med och utan omliggande blanksteg, samt tillåter dig
att ändra definitionen av sådana objekt.

Syntaxfärgning:
Vim visar texten i färg på ett sätt som är knutet till det "språk" eller
den typ av text som skrivs. Du kan själv definiera syntaxer för
dina filer som det passar dig. Vim leveraras med över 200 färdiga sådana
syntaxfiler för att fägsätta text i vanliga programspråk (Ada, C, C++,
Eiffel, Fortran, Haskell, Java, Lisp, Modula, Pascal, Prolog, Python,
Scheme, Smalltalk, SQL, Verilog, VisualBasic), matematiska språk
(Maple, Matlab, Mathematica, SAS), "markup"-språk (DocBook, HTML, LaTeX,
Postscript, SGML-LinuxDoc, TeX, WML, XML), output från program (diff, man),
konfigurationsfiler för program (4DOS, Apache, autoconfig, BibTeX, CSS,
CVS, elm, IDL, LILO, pine, procmail, samba, slrn), shell scripts och
konfigurationsfiler för shells (sh, bash, csh, ksh, zsh), skriptspråk (awk,
Perl, sed, yacc), systemfiler (printcap, .Xdefaults) och naturligtvis
för Vim och dess hjälptexter.

Särskilda funktioner:
Vim har som tillvalsfunktionalitet särskild integration
med Perl, Tcl och Python. Programmet kan också fungera
som en OLE automation server under Windows. Vidare kan Vim
också installeras med kod för X Window System med stöd
för konfigurerbara menyer och användning av mus.

Och så finns det mer. Mycket mer!

Vim:s hemsida på WWW:
  http://www.vim.org/

För en mera omfattande beskrivning av funktionaliteten i Vim,
se http://www.vim.org/why.html

Written by: Sven Guckes guckes@vim.org
Last update: Tue Oct 03 20:00:00 MET DST 2000

Översättning till svenska:
Christian Andersson <chrisand@cs.lth.se>
15 dec 2000

This text is in Koi8-u encoding (RFC-2319). In case you can't read it,
or you see it corrupted, then please adjust your browser to it.
You may contact Bohdan Vlasyuk <bohdan@kivc.vstu.vinnica.ua>
if you still have problems.

"ýÏ ÔÁËÅ Vim ?"
òÏÚÐÏ×iÄØ ÄÏ×ÖÉÎÏÀ ÛiÓÔØ ËiÌÏÂÁÊÔ.

Vim ("Vi IMproved") ÃÅ "ËÌÏÎÏ×ÁÎÉÊ vi", ÔÏÂÔÏ ÐÒÏÇÒÁÍÁ, ÓÈÏÖÁ ÎÁ
ÔÅËÓÔÏ×ÉÊ ÒÅÄÁËÔÏÒ "vi".

Vim ÐÒÁÃÀ¤ × ÔÅËÓÔÏ×ÏÍÕ ÒÅÖÉÍi ÎÁ ÂiÌØÛÏÓÔi ÔÅÒÍiÎÁÌi×, ÐiÄÔÒÉÍÕ¤
ÇÒÁÆiÞÎÉÊ iÎÔÅÒÆÅÊÓ (ÔÁËÏÖ ÍÅÎÀ), ÔÁ ÍÉÛËÕ.

äÏÓÔÕÐÎiÓÔØ:
Vim iÓÎÕ¤ ÄÌÑ ÒiÚÎÉÈ ÐÌÁÔÆÏÒÍ, i, ÐÏÒi×ÎÑÎÏ Ú Vi, ÍÁ¤ ÂÁÇÁÔÏ
ÄÏÄÁÔËÏ×ÉÈ ÆÕÎËÃiÊ (ÄÉ×. http://www.vim.org/doc/vi.diff.txt) Vim
ÓÕÍiÓÎÉÊ ÍÁÊÖÅ Ú ÕÓiÍÁ ËÏÍÁÎÄÁÍÉ Vi - ËÒiÍ ÔÉÈ, Ú×iÓÎÏ, ËÏÔÒi ÎÅ
ÐÒÁÃÀÔØ. ;-)

ïÐÅÒÁÃiÊÎi ÓÉÓÔÅÍÉ:
÷É ÍÏÖÅÔÅ ËÏÒÉÓÔÕ×ÁÔÉÓÑ Vim'ÏÍ ÎÁ ÂiÌØÛÏÓÔi ïó: AmigaOS, Atari MiNT,
BeOS, DOS, MacOS, NextStep, OS/2, OSF, RiscOS, SGI, UNIX, VMS, Win16 +
Win32 (Windows95/98/00/NT) - Á ÏÓÏÂÌÉ×Ï FreeBSD ÔÁ Linux. :-)

÷ÌÁÓÎiÓÔØ:
ðÒÁ×Á ÎÁ ÐÒÏÇÒÁÍÕ ÎÁÌÅÖÁÔØ ÇÏÌ×ÎÏÍÕ Á×ÔÏÒÕ ÔÁ ËÏÏÒÄÉÎÁÔÏÒÕ, Bram
Moolenaar <bram@vim.org>. Vim ¤ "ÂÌÁÇÏÄiÊÎÏÀ ÐÒÏÇÒÁÍÏÀ", ÔÏÂÔÏ ×É
ÍÏÖÅÔÅ ×ËÁÚÁÔÉ Ó×ÏÀ ×ÄÑÞÎiÓÔØ Á×ÔÏÒÁÍ ÐÒÏÇÒÁÍÉ ÄÏÐÏÍÁÇÁÀÞÉ ÓÉÒÏÔÁÍ
õÇÁÎÄÉ (ÄÉ×. ":help uganda").

ðÏÞÁÔËÏ×ÉÊ ËÏÄ:
Vim ÃÅ ÐÒÏÇÒÁÍÁ Ú ×iÄËÒÉÔÉÍ ËÏÄÏÍ, i ×ÁÛÁ ÄÏÐÏÍÏÇÁ × ÐÏËÒÁÝÅÎÎi Vim'Á
ÂÕÄÅ ÐÒÉÊÎÑÔÁ Ú ×iÄËÒÉÔÉÉ ÏÂiÊÍÁÍÉ!

=== æÕÎËÃiÏÎÁÌØÎiÓÔØ

òÅÄÁËÔÏÒ ÄÌÑ ÐÏÞÉÎÁÀÞÉÈ - "ìÀÂÉÍÏ ëÏÒÉÓÔÕ×ÁÞÁ":
Vim ÎÁÂÁÇÁÔÏ ÐÒÏÓÔiÛÉÊ ÄÌÑ ÐÏÞÉÎÁÀÞÉÈ ÎiÖ Vi, ÚÁ×ÄÑËÉ ÞÕÄÏ×iÊ Online
ÄÏÐÏÍÚi, ËÏÍÁÎÄÁÍ "undo" ÔÁ "redo" (ÐÏÍÉÌËÁ? ÎÅ ÂiÄÁ - ËÏÒÉÓÕÕÊÔÅÓØ
undo ÔÁ redo!), ÐiÄÔÒÉÍÃi ÍÉÛi, iËÏÎËÁÍ ÔÁ ÍÅÎÀ ÝÏ ËÏÎÆiÇÕÒÀØÓÑ
(GUI).

ôÅÒÍiÎÁÌÉ i ËÏÄÉ ÓÉÍ×ÏÌi×:
Vim ÐiÄÔÒÉÍÕ¤ iso-latin1 ÔÁ termcap. (÷ "Vanilla Vi" ÃÅ ÎÅ ÐÒÁÃÀ¤.)

óÉÍ×ÏÌÉ ÔÁ ÍÏ×É:
÷ Vim ÍÏÖÎÁ ÒÅÄÁÇÕ×ÁÔÉ ÔÅËÓÔÉ "Ú ÐÒÁ×Á ÎÁ Ìi×Ï", (ÎÁÐÒÉËÌÁÄ ÄÌÑ
áÒÂÓÉÏ§, ðÅÒÓØËÏ§ ÍÏ×, I×ÒÉÔÕ), É ÂÁÇÁÔÎÂÉÔÏ×i ÓÉÍ×ÏÌÉ, ÄÌÑ ÍÏ× ×
ÑËÉÈ ×ÉËÏÒÉÓÔÏ×ÕÂØØÓÑ i¤ÒÏÇÌiÆÉ, ÎÁÐÒÉËÌÁÄ ëÉÔÁÊÓËÁ, ñÐÏÎÓËÁ, ëÏÒÅÊÓËÁ
(Hangul) ÍÏ×É, (çÏ×ÏÒÑÞÉ ÔÅÈÎiÞÎÉÍÉ ÔÅÒÍiÎÁÍÉ, Vim ÐiÄÔÒÉÍÕ¤ ÔÅËÓÔÉ ×
ÆÏÒÍÁÔÁÈ UTF-8 i Unicode.)

æÏÒÍÁÔÕ×ÁÎÎÑ ÔÅËÓÔÕ i ×iÚÕÁÌØÎÉÊ ÒÅÖÉÍ:
÷ÉËÏÒÉÓÔÏ×ÕÀÞÉ Vim ×É ÍÏÖÅÔ ×ÉÂÒÁÔÉ ÞÁÔÉÎÕ ÔÅËÓÔÕ "×iÚÕÁÌØÎÏ" (ÚÁ
ÄÏÐÏÍÏÇÏÀ ×ÉÄiÌÅÎÑ) ÐÅÒÄ ÔÉÍ ÑË ÚÍiÀ×ÁÔÉ ÊÏÇÏ (×ÉÄÁÌÑÔÉ, ËÏÐiÀ×ÁÔÉ,
ÒÏÂÉÔÉ ÚÁÍiÎÕ ÓÉÍ×ÏÌi×, ÚÍiÎÀ×ÁÔÉ ×iÄÓÕÐ, ÒÅÇiÓÔÒ ÌiÔÅÒ, ÁÂÏ
ÆÏÒÍÁÔÕ×ÁÔÉ ÔÅËÓÔ). ôÁËÏÖ Vim ÄÏÚ×ÏÌÑ¤ ×ÉËÏÒÉÓÔÏ×Õ×ÁÔÉ ÐÒÑÍÏËÕÔÎÅ
×ÉÄiÌÅÎÑ.

ëÏÍÁÎÄÉ ÄÏÐÏ×ÎÅÎÎÑ:
Vim ÍÏÖÅ ÄÏÐÏ×ÎÀ×ÁÔÉ ÔÅËÓÔ ËÏÔÒÉÊ ×É ××ÅÌÉ - ÄÏ ÐÅ×ÎÏ§ ËÏÍÁÎÄÉ, ÄÏ
ÎÁÚ×É ÆÁÊÌÁ, ÁÂÏ ÄÏ ÓÌÏ×Á × ÔÅËÓÔi.

á×ÔÏËÏÍÁÎÄÉ:
íÏÖÎÁ ËÏÒÉÓÔÕ×ÁÔÉÓÑ "Á×ÔÏËÏÍÁÎÄÁÍÉ" ÄÌÑ Á×ÔÏÍÁÔÉÞÎÏÇÏ ×ÉËÏÎÁÎÎÑ ÐÅ×ÎÉÈ
ÄiÊ (ÎÁÐÒÉÈÌÁÄ ÄÌÑ Á×ÔÏÍÁÔÉÚÁÃi§ ÐÅÒÅÇÌÑÄÕ ÚÁÁÒÈi×Ï×ÁÎÉÈ ÁÂÏ Ä×iÊËÏ×ÉÈ
ÆÁÊÌi×).

÷ÉËÏÒÉÓÔÁÎÎÑ ÄÉÇÒÁÆi×:
Vim ÄÏÚ×ÏÌÑ¤ ××ÏÄÉÔÉ ÓÐÅÃiÁÌØÎi ÓÉÍ×ÏÌÉ ËÏÍÂiÎÁÃi¤À Ä×ÏÈ ÓÉÍ×ÏÌi×, i
ÄÏÚ×ÏÌÑ¤ ÓÔ×ÏÒÀ×ÁÔÉ Ó×Ï§ ËÏÍÂiÎÁÃi§. (ÎÁÐÒÉÈÌÁÄ ÌiÔÅÒÕ § ÍÏÖÎÁ
ÏÔÒÉÍÕ×ÁÔÉ ÐÏÓÌiÄÏ×ÎÏ ÎÁÔÉÓËÕÀÞÉ 'i' ÔÁ ':')

òÏÚÐiÚÎÁ×ÁÎÎÑ i ÚÍiÎÁ ÆÏÒÍÁÔÕ ÆÁÊÌi×:
Vim Á×ÔÏÍÁÔÉÞÎÏ ÒÏÚÐiÚÎÁ¤ ÔÉÐ ÆÁÊÌÁ (DOS, Mac, Unix) É ÄÏÚ×ÏÌÑ¤
ÚÁÐÁÍ'ÑÔÏ×Õ×ÁÔÉ ÆÁÊÌÉ × ÂÕÄØ-ÑËÏÍÕ Ú ÃÉÈ ÆÏÒÍÁÔi× - ×ÁÍ ÂiÌØÛÅ ÎÅ ÂÕÄÅ
ÐÏÔÒiÂÎÏ ËÏÒÉÓÔÕ×ÁÔÉÓÑ 'unix2dos'!

IÓÔÏÒiÑ ËÏÍÁÎÄ:
Vim ÄÏÚ×ÏÌÑ¤ ×ÅÓÔÉ "iÓÔÏÒiÀ" ËÏÍÁÎÄ É ÐÏÛÕËi×, ÔÁË ÝÏ ×É ÍÏÖÅÔÅ
×ÉËÏÎÕ×ÁÔÉ i ÒÅÄÁÇÕ×ÁÔÉ ÐÏÐÅÒÅÄÎØÏ ××ÄÅÎi ËÏÍÁÎÄÉ.

úÁÐÉÓ ÍÁËÒÏËÏÍÁÎÄ:
õ Vim ÍÏÖÎÁ "ÚÁÐÉÓÁÔÉ" ×ÁÛi Äi§ ÄÌÑ ×ÉËÏÎÁÎÎÑ ÐÏ×ÔÏÒÀ×ÁÎÉÈ ÏÐÅÒÁÃiÊ.

ïÂÍÅÖÅÎÎÑ ÐÁÍ'ÑÔi:
Vim ÍÏÖÅ ×ÉËÏÒÉÓÔÏ×Õ×ÁÔÉ ÂiÌØÛÅ ÐÁÍ`ÑÔi ÄÌÑ ÒÑÄËi× ÔÁ ÂÕÆÅÒi×.

äÅËiÌØËÁ ÂÕÆÅÒi× É ÐÏÄiÌ ÅËÒÁÎÁ:
Vim ÄÏÚ×ÏÌÑ¤ ÒÅÄÁÇÕ×ÁÔÉ ÄÅËiÌØËÁ ÂÕÆÅÒi×, É ×É ÍÏÖÅÔÅ ÒÏÚÂÉÔÉ ÅËÒÁÎ ÎÁ
ÂÁÇÁÔÏ ×iËÏÎ (ÑË ÇÏÒÉÚÏÎÔÁÌØÎÏ, ÔÁË i ×ÅÒÔÉËÁÌØÎÏ), ÔÁËÉÍ ÞÉÎÍÍ ÍÏÖÎÁ
ÏÄÎÏÞÁÓÎÏ ÂÁÞÉÔÉ ÄÅËiÌØËÁ ÆÁÊÌi×, ÁÂÏ ÄÅËiÌØËÁ ÞÁÓÔÉÎ ÏÄÎÏÇÏ ÆÁÊÌÁ.

ëiÌØËiÓÎÉÊ ÐÒÅæiËÓ ÄÏ ËÏÍÁÎÄ:
Vim ÄÏÚ×ÏÌÑ¤ ×ËÁÚÕ×ÁÔÉ ËiÌØËiÓÔØ ÐÏ×ÔÏÒi× ÄÌÑ ÂiÌØÛÏ§ ËiÌØËÏÓÔi ËÏÍÁÎÄ
ÎiÖ Vi (ÎÁÐÒÉËÌÁÄ ÄÌÑ ËÏÍÁÎÄÉ "put").

äÏÐÍÍiÖÎi ÆÁÊÌÉ (ÆÁÊÌÉ ÄÏÐÍÍÇÉ ÔÁ ÆÁÊÌÉ Ú ÏÐÉÓÏÍ ÓÉÎÔÁËÓÉÓÕ):
[ãi ÆÁÊÌÉ ×ÉËÏÒÉÓÔÏ×ÕÀÔØÓÑ ÌÉÛÅ ÐiÄ ÞÁÓ ×ÉËÏÎÁÎÎÑ ÐÒÏÇÒÁÍÉ, i ÎÅ
ÍiÓÑÑÔØ ËÏÄ, ÑËÉÊ ÐÏÒÒiÂÎÏ ËÏÍÐiÌÀ×ÁÔÉ ÔÁ ËÏÍÐÏÎÕ×ÁÔÉ.] ÒÁÚÏÍ Ú
Vim-5.7 ×É ÏÒÒÉÍÅÑÔÅ 70 ÆÁÊÌi× ÄÏÐÍÍÇÉ (ÂiÑ 2080K ÔÅËÓÕ), ÝÏ
ÏÐÉÓÕÀÔØ ËÏÍÁÎÄÉ, ÏÐÃi§, i ÍiÓÑÑÔØ ÐiÄËÁÚËÉ ÄÌÑ ÅÆÅËÔÉ×ÎÏÇÏ
ÒÅÄÁÇÕ×ÁÎÑ i ËÏÎÆiÇÕÒÕ×ÁÎÑ Vim'Á. (Vim-6.0x: 85 ÆÁÊÌi×, ÂiÌÑ 2796K
ÔÅËÓÕ). äÅÑËi ÆÁÊÌÉ ÏÐÉÓÕÀÔØ ×ÉËÏÒÉÓÔÁÎÎÑ Vim ÎÁ ÓÐÅÃÉÆiÞÛÉÈ
ÏÐÒÁÃiÊÎÉÈ ÓÉÓÔÅÍÁÈ.

óËÒÉÐÔÉ:
Vim ÍiÓÔÉÔØ ×ÂÕÄï×ÁÎÕ ÓËÒÉÐÔÏ×Õ ÍÏ×Õ, iÄÅÁÌØÎÕ ÄÌÑ ÐÒÏÓÏ§ ÔÁ Û×ÉÄËÏ§
ÒÏÚÒÏÂËÉ ÐÎÒiÂÎÉÈ ËÏÍÁÎÄ.

úÍiÝÅÎÎÑ ÐiÓÌÑ ÐÏÛÕËÕ:
Vim ÄÏÚ×ÏÌÑ¤ ×ËÁÚÕ×ÁÔÉ ÚÍiÝÅÎÎÑ ÄÌÑ ËÏÍÁÎÄ ÐÏÛÕËÕ, ÔÏÂÔÏ ËÕÒÓÏÒ ÂÕÄÅ
ÒÏÚÍiÝÅÎÉÊ *ÐiÓÌÑ* ÚÎÁÊÄÅÎÇÏ ÔÅËÓÔÕ.

òÅËÎÔÓÒÒÕËÃiÑ ÓÅÁÎÓi× ÒÏÂÏÔÉ:
Vim ÄÏÚ×ÏÌÑ¤ ÚÁÐÁÍ'ÑÔÏ×Õ×ÁÔÉ iÎÆÏÒÍÁÃiÀ ÐÒÏ ÒÅÄÁÇÕ×ÁÎÎÑ (ÓÐÉÓÏË
ÂÕÆÅÒi×, ÐÍÍiÔËÉ × ÆÁÊÌÁÈ, ÒÅÇiÓÒÙ, iÓÔÏÒiÀ ËÏÍÁÎÄ É ÐÏÛÕËÕ) × ÆÁÊÌ
("viminfo"), i ÐÒÉ ÎÁÓÔÐÎÉÈ ÒÅÄÁÇÕ×ÁÎÑÈ ×ÉËÏÒÉÓÔÏ×Õ×ÁÔÉ ÃÀ

iÏÆÏÖÏÀÄiÀ.

úÁÍiÎÁ ÓÉÍ×ÏÌi× ÔÁÂÕÌÑÃi§:
Vim ÍÏÖÅ ÚÁÍiÎÉÔÉ ÓÉÍ×ÏÌÉ ÔÁÂÕÌÑÃi§ × ÔÅËÓÔi ÎÁ ÐÒÏÂiÌÉ (expandtab,
:retab).

óÉÓÔÅÍÁ ÐÏÍiÔÏË:
Vim ÍÏÖÅ ×ÉËÏÒÉÓÔÏ×Õ×ÁÔÉ ÓÉÓÔÅÍÕ "ÐÏÍiÔÏË" ÄÌÑ iÎÄÅËÓÁÃi§ ÔÁ ÐÏÛÕËÕ ×
ÔÅËÓÔi, ÐÒÉ ÃØÏÍÕ ÍÏÖÎÁ ËÏÒÉÓÔÕ×ÁÔÉÓÑ "ÓÔÅËÏÍ" ÏÓÔÁÎÎiÈ ÐÏÍiÔÏË.

ïÂ`¤ËÔÉ × ÔÅËÓÔi:
Vim ÍÏÖÅ ÒÏÚÒiÚÎÑÔÉ ÏÂ`¤ËÔÉ ×ÓÅÒÅÄÉÎi ÔÅËÓÔÕ (ÐÁÒÁÇÒÁÆÉ, ÒÅÞÅÎÎÑ,
ÓÌÏ×Á, óìï÷á - ×ÒÁÈÏ×ÕÀÞÉ ÔÁ ÎÅ ×ÒÁÈÏ×ÕÀÞÉ ÎÁ×ËÏÌÉÚÛÎi ÐÒÍÍiÖËÉ) i
ÄÏÚ×ÌÌÑ¤ ÚÍiÎÀ×ÁÔÉ ×ÉÚÎÁÞÅÎÎÑ ÄÅÑËÉÈ ÏÂ`¤ËÔi×.

÷ÉÄiÌÅÎÎÑ ÓÉÎÔÁËÓÉÓÕ:
Vim ×iÄÏÂÒÁÖÕÅ ÔÅËÓÔ × ËÏÌØÏÒi - ÎÁÌÅÖÎÏ ÄÏ ÓÉÎÔÁËÓÉÓÕ. ÷É ÍÏÖÅÔÅ ÓÁÍi
×ÉÚÎÁÞÁÔÉ ÓÉÎÔÁËÓÉÓ ÆÁÊÌÁ.

äÌÑ Vim iÓÎÕ¤ ÂiÌØÛÅ Ä×ÏÈÓÏÔ ÆÁÊÌi× Ú ÏÐÉÓÏÍ ÓÉÎÔÁËÓÉÓÕ ÄÌÑ ÞÁÓÔÏ
ÚÁÓÔÏÓÏ×ÁÎÉÈ ÍÏ× ÐÒÏÇÒÁÍÕ×ÁÎÎÑ (Ada, C, C++, Eiffel, Fortran, Haskell,
Java, Lisp, Modula, Pascal, Prolog, Python, Scheme, Smalltalk, SQL,
Verilog, VisualBasic), ÐÒÏÇÒÁÍ ÄÌÑ ÍÁÔÅÍÁÔÉÞÎÉÈ ÒÏÚÒÁÈÕÎÏË×i× (Maple,
Matlab, Mathematica, SAS), ÍÏ× ÒÏÚÍiÔËÉ (DocBook, HTML, LaTeX,
PostScript, SGML-LinuxDoc, TeX, WML, XML), ×ÉÈÏÄÕ ÐÒÏÇÒÁÍ (diff, man),
ÆÁÊÌ×i× ÎÁÌÁÛÔÕ×ÁÎÎÑ ÐÒÏÇÒÁÍ (4DOS, Apache, autoconfig, BibTeX, CSS,
CVS, elm, IDL, LILO, pine, procmail, samba, slrn), ÓËÒÉÐÔi× shell (sh,
bash, csh, ksh, zsh), ÓÈÒÉÐÔÏ×ÉÈ ÍÏ× (awk, Perl, sed, yacc), ÓÉÓÔÅÍÎÉÈ
ÆÁÊÌi× (printcap, .Xdefaults), i, Ú×ÉÞÁÊÎÏ, ÓÉÎÔÁËÓÉÓÕ ÆÁÊÌi× ËÏÍÁÎÄ É
ÄÏÐÏÍÇÉ ÄÌÑ Vim.

óÐÅÃiÁÌØÎÉÊ ËÏÄ:
Vim ÍÏÖÅ iÎÔÅÇÒÕ×ÁÔÉÓÑ Ú Perl, Tcl i Python. Vim ÍÏÖÅ ×ÉËÏÎÕ×ÁÔÉ ÒÏÌØ
OLE automation server ÐiÄ Windows. Vim ÍÏÖÅ ÂÕÔÉ ×ÓÔÁÎÏ×ÌÅÎÉÊ Ú ËÏÄÏÍ
ÄÌÑ X-windows, ËÏÔÒÉÊ ÄÚ×ÌÌÑ¤ ËÏÎÆiÇÕÒÕ×ÁÔÉ ÍÅÎÀ É ÐiÄÔÒÍÍÕ¤ ÍÉÛÕ.

é ÝÅ ÂiÌØÛÅ ! îÁÂÁÇÁÔÏ ÂiÌØÛÅ !!

"äÏÍÁÛÎÑ ÓÔÒiÎËÁ" Vim'Á:
    http://www.vim.org/

äÌÑ ÂiÌØÛ ÐÏ×ÎÏÇÏ ÏÐÉÓÕ Vim'Á ÄÉ×iÔØÓÑ:
    http://www.vim.org/why.html

X-Original-by: Sven Guckes guckes@vim.org
X-Translated-by: Bohdan Vlasyuk <bohdan@olymp.vinnica.ua
X-Edited-by: Vim
X-Thanks: Yuriy Brazhnyk <yvb@pisem.net>

URL: http://www.math.fu-berlin.de/~guckes/vim/tree.html
URL: http://www.vim.org/tree.html (mirror)
Created: Mon Nov 16 16:16:16 CET 1998
Last update: Fri Jul 09 12:00:00 MET DST 1999

VIM Tree - Pages Overview

An overview of the pages available on Vim.

TODO: Give the date of the latest change rather than the creation date - automatically. Anyone have some good tools for this?

VIM Pages (addresses relative to base address):

Created	Title	Address
960101	VIM Answers	answ.html
951101	VIM Development	deve.html
950101	VIM Distribution/Download	dist.html
960618	VIM Keys and Commands	keys.html
960616	VIM MacOS and all that	macs.html
951201	VIM Mailing Lists	mail.html
981116	VIM News	news.html
970911	VIM Organization	orga.html
980609	VIM Quotes by Users	quot.html
960629	VIM Pictures and Screenshots	pics.html
960629	VIM Press	press/
950607	VIM Questions - Please Answer!	ques.html
981104	VIM Source - files to ":source"	source/
981116	VIM Tree (you are here :-)	tree.htmlusenet.htmluser.html
980101	VIM Utilities	util.html
950515	VIM Wishlist	wish.html
971229	VIM Why (use a vi clone)?	why.html
	VIM FAQ (Questions&Answers)	faq/
	VIM FAQ NEW (Questions&Answers)	faqnew/ TODO!
980612	VIM Docs/Helpfiles in HTML	html/
980609	VIM HowTo Documents Index	howto/
980609	VIM HowTo Binaries Preparation	howto/bina.html
960622	VIM HowTo Edit C	howto/c.html
970707	VIM HowTo Edit with Dvorak Keyboard	howto/dvor.html
960401	VIM HowTo Edit Intro	howto/edit.html

980611	VIM HowTo Format Text	howto/format.html
980611	VIM HowTo Help Vim Development	howto/help.html
960217	VIM HowTo Edit HTML	howto/html.html
960529	VIM HowTo Edit LaTeX	howto/latex.html
970903	VIM HowTo Type Rightleft	howto/rile.html
980310	VIM HowTo Tags and all that	howto/tags.html
970714	VIM HowTo Text Objects	howto/text.html
9.....	VIM and ICCF Holland	iccf/

Send feedback on this page to
Sven Guckes guckes@vim.org

```
URL:         http://www.math.fu-berlin.de/~guckes/vim/y2k.html
URL:         http://www.vim.org/y2k.html (mirror)
Created:     Fri Jan 01 00:00:00 CET 1999
Last update: Thu Nov 25 12:00:00 MET 1999
```

VIM and Y2K compliancy

Yes, Vim is Y2K compliant. :-)

Here is the info provided in the documentation:

```
:help year-2000

Since Vim internally doesn't use dates, there is no year 2000 problem.
There might be a year 2038 problem, when the seconds since January 1st 1970
don't fit in a 32 bit int[eger] anymore.  This depends on the compiler,
libraries and operating system.  Specifically, time_t and the ctime() function
are used.  And the time_t is stored in four bytes in the swap file.
But that's only used for printing a file date/time for recovery.

The Vim strftime() function directly uses the strftime() system function.
If your system libraries are year 2000 compliant, Vim is too.
```

On 990521 I had received a FAX from a company asking me to give a Y2K evaluation of *all* versions of Vim together with a note that the URL of a webpage is not enough to satisfy their legal requirements.

Well, we might say that "any damages can only be refunded as far as they do not exceed the price paid for the product". Since Vim is available for free such a statement doesn't mean anything, does it? ;-)

Anyway, the author of Vim cannot possibly take responsibility for other people's environments in which they compile and use Vim. Therefore we cannot give a guarantee.

Furthermore, a written statement has legal implications which the developers cannot fully oversee. Therefore we will not send away letters or faxes.

You will simply have to believe that all the developers have done all the best they can to prevent Vim from crashing in any way. And if you are in doubt, well, you can check the code yourself. :-)

Personally, I wonder why the industry suddenly cares so much about bugs *now*. After all, software had always had bugs before and will probably have in future. "Does anybody care what time it is" when you lose your data?

Why has noone ever asked for such statements about other kinds of bugs, eg a "Macintosh guarantee for dialogs to be without any bomb icons"? And how about an "insurance for Unix segmentation faults"? I would really look forward to see a guarantee from Microsoft that I would never have to use control-alt-delete ever again...

Send feedback on this page to
Sven Guckes guckes@vim.org

VIM Binaries and Packages - Sites with precompiled versions

The main goal of this page is to list links to pages which offer binaries for each operating system.

Hopefully, I'll manage to give you a link for every OS that Vim runs on. And if there is more than one site then I'll list only those which offer current versions.

A note to all Linux package maintainers - please send me an email so I can communicate with you!

Binaries maintainer wanted!

We are looking for binary maintainers, ie people who would maintain a website for the distribution of vim binaries of their favourite system. Anyone?

Here's an example for a well maintained binary site:

- http://www.polarfox.com/vim/ (Well done, Zoltan! :-)

I have received several enthusiatic offers for the maintenance of the binaries for Windows - but I have to turn them all down because the Windows binaries are available on *every* ftp mirror out there - so there is no need for this. Thank you for the offers, anyway. :-)

So here's a list of systems where we could not provide binaries for. Can you help these people and give them a binary? Perhaps maintain the binary with a website? Please get in touch!

Vim on PDAs and Embedded Systems: I get requests for Vim binaries on small systems at least once a week. But so far there are no sites with such binaries. Any takers?

```
System                  Date    Name+Address
======                  ====    ==============
DYNIX/ptx               990816  Tim Burlowski tim.burlowski@veritas.com
FlexOS                  990308  Gary R. Johnson grj@ync.net
PalmOS                  000424  K. Danielson kdaniels@bitstream.net
                        001223  Fred Davis fdavis2@purdue.edu
                        010222  TomH@Exabyte.COM
Psion/Epoc32            000821  Andre Berger andre.berger@topmail.de
QNX                     001011  Ingo Gross ingo.gross@zess.uni-siegen.de
Sinix                   990309  Bernhard Kraft bernhard.kraft@gmx.de might offer them
Solaris x86             990816  Tim Burlowski tim.burlowski@veritas.com [990816]
                        990906  Sumit Garg sumit.garg@usa.net
                        001013  will try: Jimmy nas@nbnet.nb.ca
Super-UX                .
Ultrix                  990816  Tim Burlowski tim.burlowski@veritas.com
Unisys                  .
UnixWare                .
WindowsCE               000418 Balazs Juhasz juhasz@frankendata.de
```

Can you help them? In case you could make a vim binary available together with a webpage to provide some info on the binary, please send me an email! Thanks! :-)

Vim Binaries Sites

Here are the sites that offer vim binaries:

System	*yymmdd*	*LINK*
A/UX	yymmdd	none yet - any takers?
AIX	------	http://www.usaserve.net/users/shaney/vim/
AIX	010316	http://sites.netscape.net/sharpPeople/vim/bin/
Alpha Win32	.....	http://george.reilly.org/vim/
Amiga	010618	none yet - any takers?
BeOS	010129	ftp://ftp.ubix.org/pub/vim/ http://www.bebits.com/app/546
BSDi	yymmdd	http://www.rassilon.net/~vitamink/vim/
Convex	yymmdd	none yet - any takers?
CygWin	991202	http://www.zip.com.au/~raf2/vim/
EPOC (Psion V)	001013	http://www.starship.freeserve.co.uk/vim.html
FreeBSD	yymmdd	ftp://ftp.freebsd.org/pub/FreeBSD/ports/{alpha,i386}/packages-{version}/editors/vim-5.3.tgz
HPUX	000812	http://hpux.connect.org.uk/hppd/hpux/Editors/vim-5.7/
IRIX	yymmdd	http://pigseye.kennesaw.edu/~dharriso/vim/ (tardist)
IRIX	yymmdd	http://www.igd.fhg.de/~zach/vim/ (tar)
LINUX	.	.
- 2.0.xx	.	http://home.mayn.de/jean-luc/vim/
- Debian (DEB)	000127	http://www.debian.org/Packages/stable/editors/vim.html
- MkLinux	yymmdd	none yet - any takers?
- RedHat (RPM)	yymmdd	http://rufus.w3.org/linux/RPM/VByName.html
- RedHat (RPM)	991129	http://www.carumba.com/rpms/
- RedHat (RPM)	yymmdd	ftp://ftp.blarg.net/pub/vim/
- Sparc	yymmdd	http://home.mayn.de/jean-luc/vim/
- SuSE	yymmdd	http://www.suse.de/en/produkte/susesoft/linux/Pakete/paket_vim.html
- SuSE	yymmdd	http://www.suse.de/de/produkte/susesoft/linux/Pakete/paket_gvim.html
MacOS	yymmdd	http://www.tu-bs.de/~i0080108/macvim.html
MacOS (mirror)	000808	http://www.kassube.de/mirrors/macvim/
MacOS (alternative)	000927	http://www3.sympatico.ca/dany.stamant/vim/
NetBSD	000926	ftp://ftp.netbsd.org/pub/NetBSD/packages/pkgsrc/editors/vim/README.html
NEXTSTEP	000704	http://wwwtg.mp.tudelft.nl/~goudswrd/vim/main.html
OpenVMS	yymmdd	see VMS
OSF	yymmdd	http://home.mayn.de/jean-luc/vim/
OS/2	yymmdd	http://hobbes.nmsu.edu/pub/os2/apps/editors/vim-5_7.zip
	yymmdd	none yet - any takers?
RiscOS	000927	http://www.ecs.soton.ac.uk/~tal197/vim.php3
SGI	yymmdd	http://toolbox.sgi.com/TasteOfDT/public/freeware/1999Feb/Installable/vim-5.0.html
Solaris	001002	http://www.sunfreeware.com/programlistsparc8.html#vim
Solaris	yymmdd	http://www.informatik.hu-berlin.de/~boehme/vim/
SunOS	yymmdd	http://www.informatik.hu-berlin.de/~boehme/vim/
SunOS	yymmdd	http://home.mayn.de/jean-luc/vim/ (m)
SunOS	yymmdd	http://www.retina.net/~tilde/vim/
VMS	000626	http://www.polarfox.com/vim/

WindowsCE	yymmdd	http://www.xt-ce.com/
Windows 16bit	------	available on all FTP mirrors - subdirectory "pc"
available on all FTP mirrors - subdirectory "pc"	------	all mirrors - subdirectory "pc"

```
DGUX := Data General Unix (DG/UX)
OSF  := Digital Unix (OSF/1)


(m) NOTE:    This site will move soon!
```

VIM Binaries and Packages - Full List

AIX

```
000301  AIX-4.3.2       vim-5.6 "Minimal"
000301  AIX-4.3.2       vim-5.6 "Console"
000301  AIX-4.3.2       vim-5.6 "Motif"
http://sites.netscape.net/sharpPeople/vim/bin/
Contact: Dan Sharp vimuser@crosswinds.net
Last checked: 010318

981119  AIX-4.1.4       vim-5.3
981119  AIX-4.2.1       vim-5.3
981119  AIX-4.3.1       vim-5.3
http://www.usaserve.net/users/shaney/vim/
Contact: Steve Haney Steve_Haney-p21393@email.mot.com
```

Alpha Win32

```
990824  Alpha Win32     vim-5.4
Contact: George Reilly George@Reilly.org
```

Amiga

```
98????  68000           vim-5.0x        GUI prototype
98????  68020           on request!
98????  68040/68060     vim-5.0x        GUI prototype
http://vip.cybercity.dk/~ccc30647/vim/ [obsolete!]
Contact: Michael Nielsen mni@dde.dk [980522]
990208:  Michael promised to update the binaries to vim-5.3 soon.
990707:  The site moved - but Michael has no more time to maintain
the binaries. Any takers?
```

BeOS

```
http://www.bebits.com/app/546
http://www.bebits.com/bob/4746/vim-5.7-BeOS-x86.pkg.zip
Contact: Denis Bourez denis(at)ubix.org
Last checked:  010129

ftp://ftp.ubix.org/pub/vim/57/
000717: vim-5.7-BeOS-x86.pkg.zip        1649Kb
ftp://ftp.ubix.org/pub/vim/60g/
```

000830: vim-6.0g-ALPHA-BeOS-x86.pkg 1936Kb
BeOS 4.5/5.0
Contact: ???
Last checked: 010129

BSDi

http://www.rassilon.net/~vitamink/vim/
http://www.retina.net/~tilde/vim/
990504 BSDi 4.0 vim-5.3
990813 BSDi 4.0 vim-5.4
Contact: Klaus Steden vitamink@vesuvius.rassilon.net tilde@retina.net

CygWin

991202 CygWin vim-5.5.41 requires coolview, X11
000316 CygWin vim-5.6.12 +gui (compiled as windows application)
http://www.zip.com.au/~raf2/vim/
Contact: "raf" raf@comdyn.com.au

EPOC (Psion V)

001013 EPOC Vim-5.5
Contact: Ajai Khattri ajai@sweet16.com

FreeBSD

990723 FreeBSD 2.2.8 vim-5.3
ftp://ftp.freebsd.org/pub/FreeBSD/ports/i386/packages-2.2.8/editors/vim-5.3.tgz

990723 FreeBSD 3.2 vim-5.3
ftp://ftp.freebsd.org/pub/FreeBSD/ports/i386/packages-3.2-stable/editors/vim-5.3.tgz

990723 FreeBSD 4 vim-5.3
ftp://ftp.freebsd.org/pub/FreeBSD/ports/i386/packages-4-current/editors/vim-5.3.tgz
Contact: Nick Hibma n_hibma@freebsd.org

HPUX

000812 HPUX-10.20 vim-5.7 +GUI_GTK glib
000812 HPUX-11.00 vim-5.7 +GUI_GTK glib
http://hpux.connect.org.uk/hppd/hpux/Editors/vim-5.7/
Contact: ???
Note: There are several mirrors of this archive around the world!
Last check: 010107

IRIX

000330 IRIX-5.3 vim-5.5.73
000330 IRIX-6.x vim-5.5.73
http://pigseye.kennesaw.edu/~dharriso/vim/
Contact: David C. Harrison Jr." dharriso@pigseye.kennesaw.edu
NOTE: Two versions available - one for "software manager"
and one version for users (which can be installed in your home directory).

000405 IRIX-6.2 vim-5.6 +cscope +GUI_Motif +python +XIM
http://www.igd.fhg.de/~zach/vim/
Contact: Gabriel Zachmann zach@igd.fhg.de

LINUX RPM

Use RPMsearch.net to find LOTS of RPMs for Vim: [000605]
http://rpmsearch.net/search?query_start=1&query=vim&%3Amethod=package

LINUX - Debian

990923 DEB   vim-5.3-13
000821 DEB   vim-5.6.070-1
http://www.debian.org/Packages/stable/editors/vim.html [000605]
Contact: Wichert Akkerman wakkerma@debian.org

LINUX - RedHat

RedHat.com's Download Page:
http://www.redhat.com/apps/download/
search for **vim** on **all** architectures

LINUX - Sparc

981005 Sparclinux-2.0.xx:
http://home.mayn.de/jean-luc/vim/
Contact: Thomas Köhler jean-luc@picard.franken.de
NOTE: This site will move soon!

LINUX - SuSE

000911 Linux-SuSE-7.0 Personal   vim-5.6-99
http://www.suse.de/de/produkte/susesoft/linux/Pakete_pers/paket_vim.html

000911 Linux-SuSE-7.0 Professional   vim-5.6-99
http://www.suse.de/de/produkte/susesoft/linux/Pakete_prof/paket_vim.html

MacOS

990904 MacOS   vim-5.4.49
990904 MacOS   vim-5.5a
http://www.tu-bs.de/~i0080108/macvim.html
Contact: Axel Kielhorn a.kielhorn@tu-bs.de
http://www.kassube.de/mirrors/macvim/ (mirror) [000808]
Contact: Nils Kassube nika@kassube.de

9902.. MacOS 68040   vim-5.3   (version info)
9902.. MacOS PPC   vim-5.3   (version info)
http://www3.sympatico.ca/dany.stamant/vim/
Contact: Dany St-Amant

NeXT

000704 NeXTSTEP-3.3   vim-5.7   Athena X11
http://wwwtg.mp.tudelft.nl/~goudswrd/vim/main.html
Contact: Jeroen Goudswaard J.C.M.Goudswaard@CTG.TUDelft.nl

OS/2

000913 OS/2 Warp   vim-5.7
http://hobbes.nmsu.edu/pub/os2/apps/editors/vim-5_7.zip
 ftp://hobbes.nmsu.edu/pub/os2/apps/editors/vim-5_7.zip
http://hobbes.nmsu.edu/cgi-bin/h-browse?dir=/pub/os2/apps/editors
Reported by: Petr Hanousek webmaster@integralsro.cz

OSF

981005 OSF/1-3.0   vim-5.3   ...
http://home.mayn.de/jean-luc/vim/
Contact: Thomas Köhler jean-luc@picard.franken.de
NOTE: This site will move soon!

QNX RiscOS

000927 RiscOS vim-5.4a   "zipped - 1,281,740 bytes"
http://www.ecs.soton.ac.uk/~tal197/vim.php3
Contact: Thomas Leonard tal197@ecs.soton.ac.uk
Would be nice to see an update to vim-5.7 ...

SCO

OpenServer
980728 "OpenServer"   vim-5.1
http://www.sco.com/skunkware/osr5/editors/vim/
990308 "UnixWare"   vim-5.3
http://www.sco.com/skunkware/uw7/editors/vim/
Contact: ???

SGI

990225 SGI   vim-5.0   no compile features known - anyone?
http://toolbox.sgi.com/TasteOfDT/public/freeware/1999Feb/Installable/vim-5.0.html
Contact: I dunno. Does anyone know who did this port?
Somebody should tell this person that Vim stands for "Vi IMproved" -
not "Vi Modified".
Besides, vim-5.0 was released on Feb 19th 1998.
Now there's vim-5.8 [010525].
And a shorter address would be nice.

SunOS

990729 SunOS-4.1.x   vim-5.4
http://home.mayn.de/jean-luc/vim/bin/ [info]
Contact: Thomas Köhler jean-luc@picard.franken.de
NOTE: This site will move soon!

Solaris

000321 vim-5.6 -perl -python -gui
000321 vim-5.6 -perl -python +GUI_GTK
000321 vim-5.6 -perl -python +GUI_Motif
000321 vim-5.6 -perl -python +GUI Xaw3d
http://www.informatik.hu-berlin.de/~boehme/vim/
Contact: Harald Boehme boehme@informatik.hu-berlin.de
Last check: 000321

000204: vim-5.4
001002: vim-5.7
        1549663 Oct 2 18:34 vim-5.7-sol8-sparc-local.gz
http://www.sunfreeware.com/programlistsparc8.html#vim
Last check: 010106

990729 SunOS-5.6 x86   vim-5.3
000508 SunOS 5.6 x86   vim-5.6
http://www.retina.net/~tilde/vim/

Contact: Klaus Steden tilde@retina.net

VMS

000211: VMS vim-5.6
000412: VMS vim-5.6.60
000626: VMS vim-5.7
http://www.polarfox.com/vim/
Contact: Zoltan Arpadffy arpadffy@altavista.net
Note: Binaries are available for Alpha or VAX, with
or without GUI, and with or without documentation.
An incredibly up-to-date site with lots of archives. :-)

WindowsCE

You can use the DOS 16bit version with it,
but this requires a registered version of the 8086 emulator "XTCE".
You can ask this guy about it: Steven La sla@poboxes.com
For more info on the XTCE emulator see
http://www.xt-ce.com/ [010618]
http://www.pyram-id.demon.co.uk/XTCE.html [obsolete]

Windows 16bit [Windows-3.1x]

"And the 16bit version is discouraged for everything, including DOS mode."
Still - any takers?

Windows 32bit [Win95 and WinNT]

The Vim binaries for Windows systems are available
from *all* ftp mirrors in the subdirectory "pc".

A note on binary size

Mike Williams mikew@harlequin.co.uk on vim-dev on 990727: Out of interest, release builds of 5.4 win32 gvim with VC5 sp2 - Optimize for ... Processor Speed Size PPro 686,080 bytes 591,360 bytes Pentium 687,104 bytes 590,848 bytes So a 95KB (~14%) saving possible when optimized for size - not to be sniffed at by some. Must be all that loop unrolling.

TODO

Add links to packages by Linux distributors (RedHat, Debian, SuSE etc). Example:

ftp://ftp.suse.com/pub/suse/i386/update/7.1/xap2/gvim.rpm
ftp://ftp.suse.com/pub/suse/i386/update/7.1/a1/vim.rpm

URL: http://www.math.fu-berlin.de/~guckes/vim/binaries.html
URL: http://www.vim.org/binaries.html (mirror)
Created: Fri Feb 05 00:00:00 CET 1999

Send feedback on this page to
Sven Guckes guckes@vim.org

VIM - Mirrors [FTP Mirrors = Download Sites; WWW mirrors]

This text gives a **short** overview to the sites that mirror Vim.

The data is sorted by the "top level domain" (TLD), gives a nickname for each mirror (either the country name or the city), and the address of the mirror (either the direct address or a vim.org alias).

Would you like to set up your own mirror of the vim distribution (FTP) or the Vim Pages (WWW)? Then take a look at the info on the Mirror HowTo.

FTP Mirrors

FTP stands for "file transfer protocol". (But you might think of it as "filr tranfer program" if you like.) It is one of the services that the Internet gives - like Email, News, and the WWW.

You can download Vim from the main site - but the connection may be slow. Therefore Vim has be copied onto other computers to which you may have a faster connection. These computers are called "mirrors", in this case "ftp mirrors".

The FTP mirrors hold a *copy* of the main download site. and therefore should hold both the source code of Vim as well as its documentation and syntax files (aka runtime files).

NOTEs: There may be exceptions. Some mirros might not have a complete copy of everything. It's all up to the maintainers of the mirrors. If you should encounter an incomplete mirror then please contact its *maintainer*.

The following list is sorted alphabetically by the top level domain (TLD) name. The "US domains" com, gov, and net are listed at the end.

Usually, the best mirror to chose is the one in your country. And the default name for that mirror *should* be `ftp.AB.vim.org` where "AB" is the country's two letter iso code. Example: The mirror in Greece should be aliased as "ftp.gr.vim.org".

TLD	NICKNAME	URL
---	HOME	ftp://ftp.home.vim.org/pub/vim/
AU	Australia	ftp://ftp.au.vim.org/pub/vim/
BE	Belgium	ftp://ftp.be.com/pub/vim/
CA	Canada	ftp://ftp.ca.vim.org/pub/vim/
DE	Germany:	ftp://ftp.de.vim.org/misc/editors/vim/
DE1	Berlin	ftp://ftp.berlin.de.vim.org/misc/editors/vim/ -> ftp1.de.vim.org [toadd]
DE2	Erlangen	ftp://ftp2.de.vim.org/ [outdated]
DE3	Munich	ftp://ftp.musin.de/pub/vim/ -> ftp3.de.vim.org [toadd]
DE4	Oldenburg	ftp://ftp.informatik.uni-oldenburg.de/pub/vim/ -> ftp4.de.vim.org [toadd] currently is ftp://ftp2.de.vim.org/ - wrong!
DK	Denmark	ftp.dk.vim.org/pub/vim [010322 - todo]
ES	Spain	ftp://ftp.es.vim.org/pub/vim/

FR	France	ftp://ftp.fr.vim.org/pub/vim/
FR2	France	ftp://ftp.fr2.vim.org/pub/vim/
GR	Greece	ftp://ftp.gr.vim.org/pub/vim/
HU	Hungary	ftp://ftp.hu.vim.org/pub/vim/ [000121: timeout]
IT	Italy	ftp.it.vim.org (coming up!)
JP	Japan:	ftp://ftp.jp.vim.org/pub/vim/
JP1	TWICS	ftp://ftp.tokio.jp.vim.org/pub/vim/
JP2	ISITE	ftp://ftp2.jp.vim.org/pub/vim/ defunct?
KR	Korea	ftp://ftp.kr.vim.org/pub/vim/
MX	Mexico	ftp://ftp.mx.vim.org/
NL	Netherlands	ftp://ftp.nl.vim.org/pub/vim/
NL1	Netherlands	ftp://ftp.nl.vim.org/pub/vim/
NL2	Netherlands	ftp://ftp2.nl.vim.org/pub/vim/
PL	Poland	ftp://ftp.pl.vim.org/pub/vim/
PT	Portugal	ftp://ftp.ci.uminho.pt/pub/editors/vim/ -> ftp.pt.vim.org [toadd]
RU	Russia	planned -> ftp.su.vim.org [toadd]
ZA	SouthAfrica	ftp://ftp.za.vim.org/applications/editors/vim/
USA:		
USA1	CA, UCDAVIS	ftp://ftp.ca.us.vim.org/pub/vim/
USA2	CA, cdrom.com	ftp://ftp.cdrom.com/pub/unixfreeware/editors/vim/ -> ftp://ftp2.ca.us.vim.org/pub/unixfreeware/editors/vim/ [toadd]
USA3	MD, NASA	ftp://ftp.nasa.us.vim.org/pub/unix/vim -> ftp.md.us.vim.org [toadd]
USA4	MI, Petrified	ftp://ftp.mi.us.vim.org/mirrors/vim/
USA5	NY, Rochester	ftp://ftp.ny.us.vim.org/editors/vim/
USA6	IL, UIUC	ftp://ftp.il.us.vim.org/packages/vim/
USA7	WA, WallaWalla	ftp://wastenot.whitman.edu/pub/vim/ -> ftp.wa.us.vim.org [toadd]

WWW/Web mirrors

The list of web mirrors of the Vim Pages. Please note that the original homepage of Vim is http://www.math.fu-berlin.de/~guckes/vim/ and that **http://www.vim.org/ is a *mirror***! Changes to the Vim Pages are thus immediately visible at the "math" site; they will be reflected by www.vim.org after the daily mirror.

Here are the current **www** mirrors of the Vim Pages:

BE	Belgium	http://mirror.grmbl.com/vim/ -> http://www.be.vim.org/ [todo]
CA	Canada	http://www.vmunix.com/vim/ -> http://www.ca.vim.org/ [toadd]
CZ	Czech Republic	http://www.instinct.org/vim/
DE	Germany:	
DE1	Berlin	http://www.math.fu-berlin.de/~guckes/vim/
DE3	Munich	http://www.vim.org/

```
DK      Denmark         http://www.dk.vim.org/pub/vim [010322 - todo]
ES      Spain           http://www.etsimo.uniovi.es/vim/
                        -> http://www.es.vim.org [toadd]
FR      France          http://web.efrei.fr/~parmelan/vim/guckes/
                        -> http://www.fr.vim.org [toadd]
KR      Korea           http://sparcs.kaist.ac.kr/mirror/vim/
                                -> http://www.kr.vim.org [toadd]
MX      Mexico          http://www.mx.vim.org/
NO      Norway          http://alge.anart.no/vim/
                                -> www.no.vim.org [toadd]

Russia:
RU1     Russia          http://vim.gambit.msk.su/
                                -> http://www.su.vim.org [toadd]
RU2     Russia          http://chronos.cs.msu.su/vim/
SK      Slovakia        http://www.kotelna.sk/vim/
ZA      SouthAfrica     http://www.leg.uct.ac.za/mirrors/guckes/vim/
                        -> http://www.za.vim.org [toadd]

USA     CA, Davis       http://ftp.us.vim.org/vim/
USA     FL, Vero Beach  http://www.fl.us.vim.org/ [todo]
USA     FL, Vero Beach  http://www.tencorp.net/www.vim.org/
USA     MI, Kalamazoo   http://rickjames.sapien.net/vim/
                        -> http://www.mi.us.vim.org [toadd]
                        no update since 981105 :-(
```

The full information about all sites is on the

- Vim Distribution Page

```
URL:         http://www.math.fu-berlin.de/~guckes/vim/mirrors.html
URL:         http://www.vim.org/mirrors.html (mirror)
Created:     Fri Jan 29 17:00:00 CET 1999
```

Send feedback on this page to
Sven Guckes guckes@vim.org

VIM Documentation Overview

An overview to online documentation about Vim.

NOTE: All **FTP mirrors** of Vim make available the **helpfiles** as an archive. You should download these and install them locally on your computer or on your local network.

Updates of Vim usually come with an update of documentation. So you when upgrade Vim then please upgrade the helpfiles, too. Thank you!

VIM Helpfiles in ASCII	http://www.vim.org/doc/
VIM Helpfiles in HTML	http://www.vim.org/html/
VIM FAQ	http://www.vim.org/faq/
VIM 5.6 Reference Guide (in PostScript and PDF)	http://physlab.sci.ccny.cuny.edu/%7Eorycc/vim-main.html
VIM RegExp Guide (in PostScript and PDF)	http://physlab.sci.ccny.cuny.edu/%7Eorycc/vim-regex.html
Vim Color Editor HOWTO	Vim Color Editor HowTo on linuxdoc.org
Vim Tutorial	http://www.hardcorelinux.com/vim-tutorial.htm

```
URL:          http://www.math.fu-berlin.de/~guckes/vim/docs.html
URL:          http://www.vim.org/docs.html (mirror)
Created:      Mon Sep 11 12:00:00 MET DST 2000
```

Send feedback on this page to
Sven Guckes guckes@vim.org

Index of /doc

Name	Last modified	Size	Description
Parent Directory	11-Jul-2001 03:16	-	
Makefile	04-Sep-1999 13:43	4k	
autocmd.txt	24-Jun-2000 11:10	35k	
change.txt	24-Jun-2000 11:10	57k	
cmdline.txt	24-Jun-2000 11:10	29k	
develop.txt	24-Jun-2000 11:10	11k	
digraph.txt	24-Jun-2000 11:10	7k	
doctags.c	18-Apr-1999 19:08	2k	
editing.txt	24-Jun-2000 11:10	36k	
eval.txt	24-Jun-2000 11:10	69k	
farsi.txt	24-Jun-2000 11:10	9k	
gui.txt	24-Jun-2000 11:10	31k	
gui_w32.txt	24-Jun-2000 11:10	17k	
gui_x11.txt	24-Jun-2000 11:10	14k	
hangulin.txt	24-Jun-2000 11:10	3k	
help.txt	24-Jun-2000 11:10	60k	
howto.txt	24-Jun-2000 11:10	3k	
if_cscope.txt	24-Jun-2000 11:10	15k	
if_ole.txt	24-Jun-2000 11:10	5k	
if_perl.txt	24-Jun-2000 11:10	8k	
if_python.txt	24-Jun-2000 11:10	9k	

if_sniff.txt	24-Jun-2000 11:10	4k
if_tcl.txt	24-Jun-2000 11:10	20k
index.txt	24-Jun-2000 11:10	52k
insert.txt	24-Jun-2000 11:10	36k
intro.txt	24-Jun-2000 11:10	31k
makehtml.awk	17-Sep-1999 19:50	16k
maketags.awk	06-Dec-1998 00:14	1k
map.txt	24-Jun-2000 11:10	31k
message.txt	24-Jun-2000 11:10	10k
motion.txt	24-Jun-2000 11:10	33k
multibyte.txt	24-Jun-2000 11:10	24k
options.txt	24-Jun-2000 11:10	191k
os_amiga.txt	24-Jun-2000 11:10	3k
os_beos.txt	24-Jun-2000 11:10	13k
os_dos.txt	24-Jun-2000 11:10	11k
os_mac.txt	24-Jun-2000 11:10	3k
os_mint.txt	24-Jun-2000 11:10	1k
os_msdos.txt	24-Jun-2000 11:10	13k
os_os2.txt	24-Jun-2000 11:10	7k
os_riscos.txt	24-Jun-2000 11:10	11k
os_unix.txt	24-Jun-2000 11:10	3k
os_vms.txt	24-Jun-2000 11:10	15k
os_win32.txt	24-Jun-2000 11:10	15k
pattern.txt	24-Jun-2000 11:10	20k

quickfix.txt	24-Jun-2000 11:10	23k
quotes.txt	24-Jun-2000 11:10	11k
recover.txt	24-Jun-2000 11:10	11k
repeat.txt	24-Jun-2000 11:10	10k
rightleft.txt	24-Jun-2000 11:10	7k
scroll.txt	24-Jun-2000 11:10	11k
starting.txt	24-Jun-2000 11:10	50k
syntax.txt	24-Jun-2000 11:10	103k
tags	24-Jun-2000 11:20	125k
tagsearch.txt	24-Jun-2000 11:10	34k
term.txt	24-Jun-2000 11:10	33k
tips.txt	24-Jun-2000 11:10	20k
todo.txt	24-Jun-2000 11:19	151k
uganda.txt	24-Jun-2000 11:10	8k
undo.txt	24-Jun-2000 11:10	5k
various.txt	24-Jun-2000 11:10	16k
version4.txt	24-Jun-2000 11:10	14k
version5.txt	24-Jun-2000 11:57	289k
vi_diff.txt	24-Jun-2000 11:10	33k
vim.1	28-Jul-1999 20:58	11k
vim2html.pl	07-Oct-1999 09:42	1k
vimtutor.1	27-Dec-1998 20:40	1k
visual.txt	24-Jun-2000 11:10	18k
windows.txt	24-Jun-2000 11:10	29k

 xxd.1 11-Jun-1999 21:55 10k

---

Apache/1.3.12 Server at www.vim.org Port 80

Vim documentation: index

main help file

---

index.txt For Vim version 5.7. Last change: 2000 Apr 01

VIM REFERENCE MANUAL by Bram Moolenaar

This file contains a list of all commands for each mode, with a tag and a short description. The lists are sorted on ASCII value.

Tip: When looking for certain functionality, use a search command. E.g., to look for deleting something, use: "/delete".

For an overview of options see help.txt |option-list|.
For a complete description of each option see options.txt |options|.
For a complete listing of all help items see |help-tags|.

==========

1. Insert mode *insert-index*

tag	char	action
|i_CTRL-@|	CTRL-@	insert previously inserted text and stop insert
|i_CTRL-A|	CTRL-A	insert previously inserted text
	CTRL-B	not used |i_CTRL-B-gone|
|i_CTRL-C|	CTRL-C	quit insert mode, without checking for abbreviation, unless 'insertmode' set.
|i_CTRL-D|	CTRL-D	delete one shiftwidth of indent in the current line
|i_CTRL-E|	CTRL-E	insert the character which is below the cursor
	CTRL-F	not used
	CTRL-G	reserved for future expansion
|i_<BS>|	<BS>	delete character before the cursor
|i_digraph|	{char1}<BS>{char2}	enter digraph (only when 'digraph' option set)

|i_CTRL-H| CTRL-H same as <BS>
|i_<Tab>| <Tab> insert a <Tab> character
|i_CTRL-I| CTRL-I same as <Tab>
|i_<NL>| <NL> same as <CR>
|i_CTRL-J| CTRL-J same as <CR>
|i_CTRL-K| CTRL-K {char1} {char2}
enter digraph
|i_CTRL-L| CTRL-L when 'insertmode' set: Leave Insert mode
|i_<CR>| <CR> begin new line
|i_CTRL-M| CTRL-M same as <CR>
|i_CTRL-N| CTRL-N find next match for keyword in front of the cursor
|i_CTRL-O| CTRL-O execute a single command and return to insert mode
|i_CTRL-P| CTRL-P find previous match for keyword in front of the cursor
|i_CTRL-Q| CTRL-Q same as CTRL-V (used for terminal control flow)
|i_CTRL-R| CTRL-R {0-9a-z"%#*:=}
insert the contents of a register
|i_CTRL-R_CTRL-R| CTRL-R CTRL-R {0-9a-z"%#*:=}
insert the contents of a register literally
|i_CTRL-R_CTRL-O| CTRL-R CTRL-O {0-9a-z"%#*:=}
insert the contents of a register literally and don't auto-indent
|i_CTRL-R_CTRL-P| CTRL-R CTRL-P {0-9a-z"%#*:=}
insert the contents of a register literally and fix indent.
CTRL-S (used for terminal control flow)
|i_CTRL-T| CTRL-T insert one shiftwidth of indent in current line
|i_CTRL-U| CTRL-U delete all entered characters in the current line
|i_CTRL-V| CTRL-V {char} insert next non-digit literally
|i_CTRL-V_digit| CTRL-V {number} insert three digit decimal number as a single byte.
|i_CTRL-W| CTRL-W delete word before the cursor
|i_CTRL-X| CTRL-X {mode} enter CTRL-X sub mode, see below
|i_CTRL-Y| CTRL-Y insert the character which is above the cursor
|i_CTRL-Z| CTRL-Z when 'insertmode' set: suspend Vim
|i_<Esc>| <Esc> end insert mode (unless 'insertmode' set)
|i_CTRL-[| CTRL-[same as <Esc>
|i_CTRL-_CTRL-N| CTRL-\ CTRL-N go to Normal mode
CTRL-\ a - z reserved for extensions
CTRL-\ others not used
|i_CTRL-]| CTRL-] trigger abbreviation
CTRL-^ not used
|i_CTRL-_| CTRL-_ When 'allowrevins' set: change language (Hebrew, Farsi) {only when compiled with +rightleft feature}

<Space> to '~' not used, except '0' and '^' followed by

CTRL-D

|i_0_CTRL-D| 0 CTRL-D delete all indent in the current line
|i_^_CTRL-D| ^ CTRL-D delete all indent in the current line, restore
                       it in the next line

|i_<Del>| <Del> delete character under the cursor

          Meta characters (0x80 to 0xff, 128 to 255)
                       not used

|i_<Left>| <Left> cursor one character left
|i_<S-Left>| <S-Left> cursor one word left
|i_<C-Left>| <C-Left> cursor one word left
|i_<Right>| <Right> cursor one character right
|i_<S-Right>| <S-Right> cursor one word right
|i_<C-Right>| <C-Right> cursor one word right
|i_<Up>| <Up> cursor one line up
|i_<S-Up>| <S-Up> same as <PageUp>
|i_<Down>| <Down> cursor one line down
|i_<S-Down>| <S-Down> same as <PageDown>
|i_<Home>| <Home> cursor to start of line
|i_<C-Home>| <C-Home> cursor to start of file
|i_<End>| <End> cursor past end of line
|i_<C-End>| <C-End> cursor past end of file
|i_<PageUp>| <PageUp> one screenfull backward
|i_<PageDown>| <PageDown> one screenfull forward
|i_<F1>| <F1> same as <Help>
|i_<Help>| <Help> stop insert mode and display help window
|i_<Insert>| <Insert> toggle Insert/Replace mode
|i_<LeftMouse>| <LeftMouse> cursor at mouse click
|i_<MouseDown>| <MouseDown> scroll three lines downwards
|i_<S-MouseDown>| <S-MouseDown> scroll a full page downwards
|i_<MouseUp>| <MouseUp> scroll three lines upwards
|i_<S-MouseUp>| <S-MouseUp> scroll a full page upwards

commands in CTRL-X submode

|i_CTRL-X_CTRL-D| CTRL-X CTRL-D complete defined identifiers
|i_CTRL-X_CTRL-E| CTRL-X CTRL-E scroll up
|i_CTRL-X_CTRL-F| CTRL-X CTRL-F complete file names
|i_CTRL-X_CTRL-I| CTRL-X CTRL-I complete identifiers
|i_CTRL-X_CTRL-K| CTRL-X CTRL-K complete identifiers from dictionary
|i_CTRL-X_CTRL-L| CTRL-X CTRL-L complete whole lines
|i_CTRL-X_CTRL-L| CTRL-X CTRL-N next completion
|i_CTRL-X_CTRL-L| CTRL-X CTRL-P previous completion
|i_CTRL-X_CTRL-Y| CTRL-X CTRL-Y scroll down
|i_CTRL-X_CTRL-]| CTRL-X CTRL-] complete tags
{not available when compiled without the +insert_expand feature}

==========

2. Normal mode *normal-index*

CHAR any non-blank character
WORD any sequences of non-blank characters
N a number entered before the command
{motion} a cursor movement command
Nmove the text that is moved over with a {motion}
SECTION a section that possibly starts with '}' instead of '{'

note: 1 = cursor movement command; 2 = can be undone/redone

tag	char	note	action in Normal mode
	CTRL-@		not used
|CTRL-A|	CTRL-A	2	add N to number at/after cursor
|CTRL-B|	CTRL-B	1	scroll N screens Backwards
|CTRL-C|	CTRL-C		interrupt current (search) command
|CTRL-D|	CTRL-D		scroll Down N lines (default: half a screen)
|CTRL-E|	CTRL-E		scroll N lines upwards (N lines Extra)
|CTRL-F|	CTRL-F	1	scroll N screens Forward
|CTRL-G|	CTRL-G		display current file name and position
|<BS>|	<BS>	1	same as "h"
|CTRL-H|	CTRL-H	1	same as "h"
|<Tab>|	<Tab>	1	go to N newer entry in jump list
|CTRL-I|	CTRL-I	1	same as <Tab>
|<NL>|	<NL>	1	same as "j"
|CTRL-J|	CTRL-J	1	same as "j"
	CTRL-K		not used
|CTRL-L|	CTRL-L		redraw screen
|<CR>|	<CR>	1	cursor to the first CHAR N lines lower
|CTRL-M|	CTRL-M>	1	same as <CR>
|CTRL-N|	CTRL-N	1	same as "j"
|CTRL-O|	CTRL-O	1	go to N older entry in jump list
|CTRL-P|	CTRL-P	1	cursor N lines upward
	CTRL-Q		(used for terminal control flow)
|CTRL-R|	CTRL-R	2	redo changes which were undone with 'u'
	CTRL-S		(used for terminal control flow)
|CTRL-T|	CTRL-T		jump to N older Tag in tag list
|CTRL-U|	CTRL-U		scroll N lines Upwards (default: half a screen)
|CTRL-V|	CTRL-V		start blockwise Visual mode
|CTRL-W|	CTRL-W {char}		window commands, see |CTRL-W|
|CTRL-X|	CTRL-X	2	subtract N from number at/after cursor
|CTRL-Y|	CTRL-Y		scroll N lines downwards
|CTRL-Z|	CTRL-Z		suspend program (or start new shell)
	CTRL-[<Esc>		not used
|CTRL-\ CTRL-N|	CTRL-\ CTRL-N		go to Normal mode (no-op)
	CTRL-\ a - z		reserved for extensions
	CTRL-\ others		not used

|CTRL-]| CTRL-] :ta to ident under cursor
|CTRL-^| CTRL-^ edit Nth alternate file (equivalent to ":e #N")
CTRL-_ not used

|<Space>| <Space> 1 same as "l"
|!| !{motion}{filter}
2 filter Nmove text through the {filter} command
|!!| !!{filter} 2 filter N lines through the {filter} command
|quote| "{a-zA-Z0-9.%#:-"} use buffer {a-zA-Z0-9.%#:-"} for next delete, yank or put (upper case to append) ({.%#:} only work with put)
|#| # 1 search backward for the Nth occurrence of the ident under the cursor
|$| $ 1 cursor to the end of Nth next line
|%| % 1 find the next (curly/square) bracket on this line and go to its match, or go to matching comment bracket, or go to matching preprocessor directive.
|N%| {count}% 1 go to N percentage in the file
|&| & 2 repeat last :s
|'| '{a-zA-Z0-9} 1 cursor to the first CHAR on the line with mark {a-zA-Z0-9}
|''| '' 1 cursor to the first CHAR of the line where the cursor was before the latest jump.
|'<| '< 1 cursor to the first CHAR of the line where highlighted area starts/started in the current buffer.
|'>| '> 1 cursor to the first CHAR of the line where highlighted area ends/ended in the current buffer.
|'[| '[1 cursor to the first CHAR on the line of the start of last operated text or start of putted text
|']| '] 1 cursor to the first CHAR on the line of the end of last operated text or end of putted text
|(| (1 cursor N sentences backward
|)|) 1 cursor N sentences forward
|star| * 1 search forward for the Nth occurrence of the ident under the cursor
|+| + 1 cursor to the first CHAR N lines lower
|,| , 1 repeat latest f, t, F or T in opposite direction N times
|-| - 1 cursor to the first CHAR N lines higher
|.| . 2 repeat last change with count replaced with N
|/| /{pattern}<CR> 1 search forward for the Nth occurrence of {pattern}
|/<CR>| /<CR> 1 search forward for {pattern} of last search

count	0	1	cursor to the first char of the line
count	1		prepend to command to give a count
count	2		"
count	3		"
count	4		"
count	5		"
count	6		"
count	7		"
count	8		"
count	9		"
:	:		start entering an Ex command
N:	{count}:		start entering an Ex command with range from current line to N lines down
;	;	1	repeat latest f, t, F or T N times
<	<{motion}	2	shift Nmove lines one 'shiftwidth' leftwards
<<	<<	2	shift N lines one 'shiftwidth' leftwards
=	={motion}	2	filter Nmove lines through "indent"
==	==	2	filter N lines through "indent"
>	>{motion}	2	shift Nmove lines one 'shiftwidth' rightwards
>>	>>	2	shift N lines one 'shiftwidth' rightwards
?	?{pattern}<CR>	1	search backward for the Nth previous occurrence of {pattern}
?<CR>	?<CR>	1	search backward for {pattern} of last search
@	@{a-z}	2	execute the contents of named buffer {a-z} N times
@:	@:		repeat the previous ":" command N times
@@	@@	2	repeat the previous @{a-z} N times
A	A	2	append text after the end of the line N times
B	B	1	cursor N WORDS backward
C	["x]C	2	change from the cursor position to the end of the line, and N-1 more lines [into buffer x]; synonym for "c$"
D	["x]D	2	delete the characters under the cursor until the end of the line and N-1 more lines [into buffer x]; synonym for "d$"
E	E	1	cursor forward to the end of WORD N
F	F{char}	1	cursor to the Nth occurrence of {char} to the left
G	G	1	cursor to line N, default last line
H	H	1	cursor to line N from top of screen
I	I	2	insert text before the first CHAR on the line N times
J	J	2	Join N lines; default is 2
K	K		lookup Keyword under the cursor with 'keywordprg'
L	L	1	cursor to line N from bottom of screen
M	M	1	cursor to middle line of screen
N	N	1	repeat the latest '/' or '?' N times in

			opposite direction
|O|	O	2	begin a new line above the cursor and insert text, repeat N times
|P|	["x]P	2	put the text [from buffer x] before the cursor N times
|Q|	Q		switch to "Ex" mode
|R|	R	2	enter replace mode: overtype existing characters, repeat the entered text N-1 times
|S|	["x]S	2	delete N lines [into buffer x] and start insert; synonym for "^cc" or "0cc", depending on autoindent
|T|	T{char}	1	cursor till after Nth occurrence of {char} to the left
|U|	U	2	undo all latest changes on one line
|V|	V		start linewise Visual mode
|W|	W	1	cursor N WORDS forward
|X|	["x]X	2	delete N characters before the cursor [into buffer x]
|Y|	["x]Y		yank N lines [into buffer x]; synonym for "yy"
|ZZ|	ZZ		store current file if modified, and exit
|ZQ|	ZQ		exit current file always
|[|	[{char}		square bracket command (see below)
		\			not used
|]|	]{char}		square bracket command (see below)
|^|	^	1	cursor to the first CHAR of the line
|_|	_	1	cursor to the first CHAR N - 1 lines lower
|`|	`{a-zA-Z0-9}	1	cursor to the mark {a-zA-Z0-9}
|`<|	`<	1	cursor to the start of the highlighted area
|`>|	`>	1	cursor to the end of the highlighted area
|`[|	`[	1	cursor to the start of last operated text or start of putted text
|`]|	`]	1	cursor to the end of last operated text or end of putted text
|``|	``	1	cursor to the position before latest jump
|a|	a	2	append text after the cursor N times
|b|	b	1	cursor N words backward
|c|	["x]c{motion}	2	delete Nmove text [into buffer x] and start insert
|cc|	["x]cc	2	delete N lines [into buffer x] and start insert
|d|	["x]d{motion}	2	delete Nmove text [into buffer x]
|dd|	["x]dd	2	delete N lines [into buffer x]
|e|	e	1	cursor forward to the end of word N
|f|	f{char}	1	cursor to Nth occurrence of {char} to the right
|g|	g{char}		extended commands, see below
|h|	h	1	cursor N chars to the left
|i|	i	2	insert text before the cursor N times
|j|	j	1	cursor N lines downward

|k|	k	1	cursor N lines upward
|l|	l	1	cursor N chars to the right
|m|	m{A-Za-z}		set mark {A-Za-z} at cursor position
|n|	n	1	repeat the latest '/' or '?' N times
|o|	o	2	begin a new line below the cursor and insert text, repeat N times
|p|	["x]p	2	put the text [from register x] after the cursor N times
|q|	q{0-9a-zA-Z"}		record typed characters into named register {0-9a-zA-Z"} (upper case to append)
|q|	q		(while recording) stops recording
|r|	r{char}	2	replace N chars with {char}
|s|	["x]s	2	(substitute) delete N characters [into buffer x] and start insert
|t|	t{char}	1	cursor till before Nth occurrence of {char} to the right
|u|	u	2	undo changes
|v|	v		start characterwise Visual mode
|w|	w	1	cursor N words forward
|x|	["x]x	2	delete N characters under and after the cursor [into buffer x]
|y|	["x]y{motion}		yank Nmove text [into buffer x]
|yy|	["x]yy		yank N lines [into buffer x]
|z<CR>|	z<CR>		redraw, cursor line to top of window, cursor on first non-blank
|zN<CR>|	z{height}<CR>		redraw, make window {height} lines high
|z+|	z+		cursor on line N (default line below window), otherwise like "z<CR>"
|z-|	z-		redraw, cursor line at bottom of window, cursor on first non-blank
|z.|	z.		redraw, cursor line to center of window, cursor on first non-blank
|z^|	z^		cursor on line N (default line above window), otherwise like "z-"
|zb|	zb		redraw, cursor line at bottom of window
|ze|	ze		when 'wrap' off scroll horizontally to position the cursor at the end (right side) of the screen
|zh|	zh		when 'wrap' off scroll screen N characters to the right
|zl|	zl		when 'wrap' off scroll screen N characters to the left
|zs|	zs		when 'wrap' off scroll horizontally to position the cursor at the start (left side) of the screen
|zt|	zt		redraw, cursor line at top of window
|zz|	zz		redraw, cursor line at center of window
|z<Left>|	z<Left>		same as "zh"
|z<Right>|	z<Right>		same as "zl"
|{|	{	1	cursor N paragraphs backward
|bar|	|	1	cursor to column N

|}|	}	1	cursor N paragraphs forward
|~|	~	2	'tildeop' off: switch case of N characters under cursor and move the cursor N characters to the right
|~|	~{motion}		'tildeop' on: switch case of Nmove text
|<C-End>|	<C-End>	1	same as "G"
|<C-Home>|	<C-Home>	1	same as "gg"
|<C-Left>|	<C-Left>	1	same as "b"
|<C-LeftMouse>|	<C-LeftMouse>		":ta" to the keyword at the mouse click
|<C-Right>|	<C-Right>	1	same as "w"
|<C-RightMouse>|	<C-RightMouse>		same as "CTRL-T"
|<Del>|	["x]<Del>	2	same as "x"
|N<Del>|	{count}<Del>		remove the last digit from {count}
|<Down>|	<Down>	1	same as "j"
|<End>|	<End>	1	same as "$"
|<F1>|	<F1>		same as <Help>
|<Help>|	<Help>		open a help window
|<Home>|	<Home>	1	same as "0"
|<Insert>|	<Insert>	2	same as "i"
|<Left>|	<Left>	1	same as "h"
|<LeftMouse>|	<LeftMouse>	1	move cursor to the mouse click position
|<MiddleMouse>|	<MiddleMouse>	2	same as "P" at the mouse click position
|<PageDown>|	<PageDown>		same as CTRL-F
|<PageUp>|	<PageUp>		same as CTRL-B
|<Right>|	<Right>	1	same as "l"
|<RightMouse>|	<RightMouse>		start Visual mode, move cursor to the mouse click position
|<S-Down>|	<S-Down>	1	same as CTRL-F
|<S-Left>|	<S-Left>	1	same as "b"
|<S-LeftMouse>|	<S-LeftMouse>		same as "*" at the mouse click position
|<S-Right>|	<S-Right>	1	same as "w"
|<S-RightMouse>|	<S-RightMouse>		same as "#" at the mouse click position
|<S-Up>|	<S-Up>	1	same as CTRL-B
|<Undo>|	<Undo>	2	same as "u"
|<Up>|	<Up>	1	same as "k"
|<MouseDown>|	<MouseDown>		scroll three lines downwards
|<S-MouseDown>|	<S-MouseDown>		scroll a full page downwards
|<MouseUp>|	<MouseUp>		scroll three lines upwards
|<S-MouseUp>|	<S-MouseUp>		scroll a full page upwards

==========

2.1 Text objects *objects*

These can be used after an operator or in Visual mode to select an object.

tag	command	action in Normal mode
|v_a(|	a(	same as ab
|v_a)|	a)	same as ab

|v_a<| a< "a <>" from '<' to the matching '>'
|v_a>| a> same as a<
|v_aB| aB "a Block" from "[{" to "]}" (with brackets)
|v_aW| aW "a WORD" (with white space)
|v_a[| a["a []" from '[' to the matching ']'
|v_a]| a] same as a[
|v_ab| ab "a block" from "[(" to "])" (with braces)
|v_ap| ap "a paragraph" (with white space)
|v_as| as "a sentence" (with white space)
|v_aw| aw "a word" (with white space)
|v_a{| a{ same as aB
|v_a}| a} same as aB
|v_i(| i(same as ib
|v_i)| i) same as ib
|v_i<| i< "inner <>" from '<' to the matching '>'
|v_i>| i> same as i<
|v_iB| iB "inner Block" from "[{" and "]}"
|v_iW| iW "inner WORD"
|v_i[| i["inner []" from '[' to the matching ']'
|v_i]| i] same as i[
|v_ib| ib "inner block" from "[(" to "])"
|v_ip| ip "inner paragraph"
|v_is| is "inner sentence"
|v_iw| iw "inner word"
|v_i{| i{ same as iB
|v_i}| i} same as iB

==

2.2 Window commands *CTRL-W*

tag	command	action in Normal mode
CTRL-W_CTRL-B	CTRL-W CTRL-B	same as "CTRL-W b"
CTRL-W_CTRL-C	CTRL-W CTRL-C	same as "CTRL-W c"
CTRL-W_CTRL-D	CTRL-W CTRL-D	same as "CTRL-W d"
CTRL-W_CTRL-F	CTRL-W CTRL-F	same as "CTRL-W f"
	CTRL-W CTRL-G	same as "CTRL-W g .."
CTRL-W_CTRL-I	CTRL-W CTRL-I	same as "CTRL-W i"
CTRL-W_CTRL-J	CTRL-W CTRL-J	same as "CTRL-W j"
CTRL-W_CTRL-K	CTRL-W CTRL-K	same as "CTRL-W k"
CTRL-W_CTRL-N	CTRL-W CTRL-N	same as "CTRL-W n"
CTRL-W_CTRL-O	CTRL-W CTRL-O	same as "CTRL-W o"
CTRL-W_CTRL-P	CTRL-W CTRL-P	same as "CTRL-W p"
CTRL-W_CTRL-Q	CTRL-W CTRL-Q	same as "CTRL-W q"
CTRL-W_CTRL-R	CTRL-W CTRL-R	same as "CTRL-W r"
CTRL-W_CTRL-S	CTRL-W CTRL-S	same as "CTRL-W s"
CTRL-W_CTRL-T	CTRL-W CTRL-T	same as "CTRL-W t"
CTRL-W_CTRL-W	CTRL-W CTRL-W	same as "CTRL-W w"
CTRL-W_CTRL-X	CTRL-W CTRL-X	same as "CTRL-W x"

CTRL-W_CTRL-Z	CTRL-W CTRL-Z	same as "CTRL-W z"
CTRL-W_CTRL-]	CTRL-W CTRL-]	same as "CTRL-W]"
CTRL-W_CTRL-^	CTRL-W CTRL-^	same as "CTRL-W ^"
CTRL-W_CTRL-_	CTRL-W CTRL-_	same as "CTRL-W _"
CTRL-W_+	CTRL-W +	increase current window height N lines
CTRL-W_-	CTRL-W -	decrease current window height N lines
CTRL-W_=	CTRL-W =	make all windows the same height
CTRL-W_R	CTRL-W R	rotate windows upwards N times
CTRL-W_S	CTRL-W S	same as "CTRL-W s"
CTRL-W_W	CTRL-W W	go to N previous window (wrap around)
CTRL-W_]	CTRL-W]	split window and jump to tag under cursor
CTRL-W_^	CTRL-W ^	split current window and edit alternate file N
CTRL-W__	CTRL-W _	set current window height to N (default: very high)
CTRL-W_b	CTRL-W b	go to bottom window
CTRL-W_c	CTRL-W c	close current window (like :close)
CTRL-W_d	CTRL-W d	split window and jump to definition under the cursor
CTRL-W_f	CTRL-W f	split window and edit file name under the cursor
CTRL-W_g_CTRL-]	CTRL-W g CTRL-]	split window and do :tjump to tag under cursor
CTRL-W_g]	CTRL-W g]	split window and do :tselect for tag under cursor
CTRL-W_g}	CTRL-W g }	do a :ptjump to the tag under the cursor
CTRL-W_i	CTRL-W i	split window and jump to declaration of identifier under the cursor
CTRL-W_j	CTRL-W j	go to N next window (stop at last window)
CTRL-W_k	CTRL-W k	go to N previous window (stop at first window)
CTRL-W_n	CTRL-W n	open new window, N lines high
CTRL-W_o	CTRL-W o	close all but current window (like :only)
CTRL-W_p	CTRL-W p	go to previous (last accessed) window
CTRL-W_q	CTRL-W q	quit current window (like :quit)
CTRL-W_r	CTRL-W r	rotate windows downwards N times
CTRL-W_s	CTRL-W s	split current window in two parts, new window N lines high
CTRL-W_t	CTRL-W t	go to top window
CTRL-W_w	CTRL-W w	go to N next window (wrap around)
CTRL-W_x	CTRL-W x	exchange current window with window N (default: next window)
CTRL-W_z	CTRL-W z	close preview window
CTRL-W_}	CTRL-W }	show tag under cursor in preview window
CTRL-W_<Down>	CTRL-W <Down>	same as "CTRL-W j"
CTRL-W_<Up>	CTRL-W <Up>	same as "CTRL-W k"

==========

2.3 Square bracket commands *[* *]*

tag	char	note	action in Normal mode
[_CTRL-D	[_CTRL-D		jump to first #define found in current and included files matching the word under the cursor, start searching at beginning of current file
[_CTRL-I	[_CTRL-I		jump to first line in current and included files that contains the word under the cursor, start searching at beginning of current file
[#	[#	1	cursor to N previous unmatched #if, #else or #ifdef
[(	[(	1	cursor N times back to unmatched '('
[star	[*	1	same as "[/"
[/	[/	1	cursor to N previous start of a C comment
[D	[D		list all defines found in current and included files matching the word under the cursor, start searching at beginning of current file
[I	[I		list all lines found in current and included files that contain the word under the cursor, start searching at beginning of current file
[P	[P	2	same as "[p"
[[	[[	1	cursor N sections backward
[]	[]	1	cursor N SECTIONS backward
[d	[d		show first #define found in current and included files matching the word under the cursor, start searching at beginning of current file
[f	[f		same as "gf"
[i	[i		show first line found in current and included files that contains the word under the cursor, start searching at beginning of current file
[p	[p	2	like "P", but adjust indent to current line
[m	[m	1	cursor N times back to start of member function
[{	[{	1	cursor N times back to unmatched '{'
[<MiddleMouse>	[<MiddleMouse>	2	same as "[p"
]_CTRL-D	]_CTRL-D		jump to first #define found in current and included files matching the word under the cursor, start searching at cursor position
]_CTRL-I	]_CTRL-I		jump to first line in current and included files that contains the word under the cursor, start searching at cursor position
]#	]#	1	cursor to N next unmatched #endif or #else
])	])	1	cursor N times forward to unmatched ')'
]star	]*	1	same as "]/"
]/	]/	1	cursor to N next end of a C comment

]D	]D		list all #defines found in current and included files matching the word under the cursor, start searching at cursor position
]I	]I		list all lines found in current and included files that contain the word under the cursor, start searching at cursor position
]P	]P	2	same as "[p"
][	][	1	cursor N SECTIONS forward
]]	]]	1	cursor N sections forward
]d	]d		show first #define found in current and included files matching the word under the cursor, start searching at cursor position
]f	]f		same as "gf"
]i	]i		show first line found in current and included files that contains the word under the cursor, start searching at cursor position
]p	]p	2	like "p", but adjust indent to current line
]m	]m	1	cursor N times forward to end of member function
]}	]}	1	cursor N times forward to unmatched '}'
]<MiddleMouse>	]<MiddleMouse>	2	same as "]p"

==========

2.4 Commands starting with 'g' *g*

tag	char	note	action in Normal mode
g_CTRL-A	g_CTRL-A		only when compiled with MEM_PROFILE defined: dump a memory profile
g_CTRL-G	g_CTRL-G		show information about current cursor position
g_CTRL-H	g_CTRL-H		start Select block mode
g_CTRL-]	g_CTRL-]		:tjump to the tag under the cursor
g#	g#	1	like "#", but without using "\<" and "\>"
g$	g$	1	when 'wrap' off go to rightmost character of the current line that is on the screen; when 'wrap' on go to the rightmost character of the current screen line
gstar	g*	1	like "*", but without using "\<" and "\>"
g0	g0	1	when 'wrap' off go to leftmost character of the current line that is on the screen; when 'wrap' on go to the leftmost character of the current screen line
g?	g?	2	Rot13 encoding operator
g?g?	g??	2	Rot13 encode current line
g?g?	g?g?	2	Rot13 encode current line
gD	gD	1	go to definition of word under the cursor in current file

|gE|	gE	1	go backwards to the end of the previous WORD
|gH|	gH		start Select line mode
|gI|	gI	2	like "I", but always start in column 1
|gJ|	gJ	2	join lines without inserting space
|gP|	["x]gP	2	put the text [from register x] before the cursor N times, leave the cursor after it
|gR|	gR	2	enter virtual replace mode
|gU|	gU{motion}	2	make Nmove text uppercase
|gV|	gV		don't reselect the previous Visual area when executing a mapping or menu in Select mode
|g]|	g]		:tselect on the tag under the cursor
|g^|	g^	1	when 'wrap' off go to leftmost non-white character of the current line that is on the screen; when 'wrap' on go to the leftmost non-white character of the current screen line
|ga|	ga		print ascii value of character under the cursor
|gd|	gd	1	go to definition of word under the cursor in current function
|ge|	ge	1	go backwards to the end of the previous word
|gf|	gf		start editing the file whose name is under the cursor
|gg|	gg	1	cursor to line N, default first line
|gh|	gh		start Select mode
|gj|	gj	1	like "j", but when 'wrap' on go N screen lines down
|gk|	gk	1	like "k", but when 'wrap' on go N screen lines up
|gm|	gm	1	go to character at middle of the screenline
|go|	go	1	cursor to byte N in the buffer
|gp|	["x]gp	2	put the text [from register x] after the cursor N times, leave the cursor after it
|gq|	gq{motion}	2	format Nmove text
|gr|	gr{char}	2	virtual replace N chars with {char}
|gs|	gs		go to sleep for N seconds (default 1)
|gu|	gu{motion}	2	make Nmove text lowercase
|gv|	gv		reselect the previous Visual area
|g~|	g~{motion}	2	swap case for Nmove text
|g<Down>|	g<Down>	1	same as "gj"
|g<End>|	g<End>	1	same as "g$"
|g<Home>|	g<Home>	1	same as "g0"
|g<LeftMouse>|	g<LeftMouse>		same as <C-LeftMouse>
	g<MiddleMouse>		same as <C-MiddleMouse>
|g<RightMouse>|	g<RightMouse>		same as <C-RightMouse>
|g<Up>|	g<Up>	1	same as "gk"

==========

3. Visual mode *visual-index*

Most commands in Visual mode are the same as in Normal mode. The ones listed here are those that are different.

tag	command	note	action in Visual mode
v_CTRL-_CTRL-N	CTRL-\ CTRL-N		stop Visual mode
v_CTRL-G	CTRL-G		toggle between Visual mode and Select mode
v_<BS>	<BS>	2	Select mode: delete highlighted area
v_CTRL-H	CTRL-H	2	same as <BS>
v_CTRL-O	CTRL-O		switch from Select to Visual mode for one command
v_CTRL-V	CTRL-V		make Visual mode blockwise or stop Visual mode
v_CTRL-]	CTRL-]		jump to highlighted tag
v_!	!{filter}	2	filter the highlighted lines through the external command {filter}
v_:	:		start a command-line with the highlighted lines as a range
v_<	<	2	shift the highlighted lines one 'shiftwidth' left
v_=	=	2	filter the highlighted lines through the external program given with the 'equalprg' option
v_>	>	2	shift the highlighted lines one 'shiftwidth' right
v_b_A	A	2	block mode: append same text in all lines, after the highlighted area
v_C	C	2	delete the highlighted lines and start insert
v_D	D	2	delete the highlighted lines
v_b_I	I	2	block mode: insert same text in all lines, before the highlighted area
v_J	J	2	join the highlighted lines
v_K	K		run 'keywordprg' on the highlighted area
v_O	O		Move horizontally to other corner of area.
	Q		does not start Ex mode
v_R	R	2	delete the highlighted lines and start insert
v_S	S	2	delete the highlighted lines and start insert
v_U	U	2	make highlighted area uppercase
v_V	V		make Visual mode linewise or stop Visual mode
v_X	X	2	delete the highlighted lines
v_Y	Y		yank the highlighted lines
v_a(	a(		same as ab
v_a)	a)		same as ab
v_a<	a<		extend highlighted area with a <> block

|v_a>| a> same as a<
|v_aB| aB extend highlighted area with a {} block
|v_aW| aW extend highlighted area with "a WORD"
|v_a[| a[extend highlighted area with a [] block
|v_a]| a] same as a[
|v_ab| ab extend highlighted area with a () block
|v_ap| ap extend highlighted area with a paragraph
|v_as| as extend highlighted area with a sentence
|v_aw| aw extend highlighted area with "a word"
|v_a{| a{ same as aB
|v_a}| a} same as aB
|v_c| c 2 delete highlighted area and start insert
|v_d| d 2 delete highlighted area
|v_gJ| gJ 2 join the highlighted lines without inserting spaces
|v_gq| gq 2 format the highlighted lines
|v_gv| gv exchange current and previous highlighted area
|v_i(| i(same as ib
|v_i)| i) same as ib
|v_i<| i< extend highlighted area with inner <> block
|v_i>| i> same as i<
|v_iB| iB extend highlighted area with inner {} block
|v_iW| iW extend highlighted area with "inner WORD"
|v_i[| i[extend highlighted area with inner [] block
|v_i]| i] same as i[
|v_ib| ib extend highlighted area with inner () block
|v_ip| ip extend highlighted area with inner paragraph
|v_is| is extend highlighted area with inner sentence
|v_iw| iw extend highlighted area with "inner word"
|v_i{| i{ same as iB
|v_i}| i} same as iB
|v_o| o move cursor to other corner of area
|v_r| r 2 delete highlighted area and start insert
|v_s| s 2 delete highlighted area and start insert
|v_u| u 2 make highlighted area lowercase
|v_v| v make Visual mode characterwise or stop Visual mode
|v_x| x 2 delete the highlighted area
|v_y| y yank the highlighted area
|v_~| ~ 2 swap case for the highlighted area

==

4. Command-line editing *ex-edit-index*

Get to the command-line with the ':', '!', '/' or '?' commands.
Normal characters are inserted at the current cursor position.
"Completion" below refers to context-sensitive completion. It will complete
file names, tags, commands etc. as appropriate.

	CTRL-@	not used
c_CTRL-A	CTRL-A	do completion on the pattern in front of the cursor and insert all matches
c_CTRL-B	CTRL-B	cursor to begin of command-line
c_CTRL-C	CTRL-C	same as <ESC>
c_CTRL-D	CTRL-D	list completions that match the pattern in front of the cursor
c_CTRL-E	CTRL-E	cursor to end of command-line
	CTRL-F	not used
	CTRL-G	not used
c_<BS>	<BS>	delete the character in front of the cursor
c_digraph	{char1} <BS> {char2}	enter digraph when 'digraph' is on
c_CTRL-H	CTRL-H	same as <BS>
c_<Tab>	<Tab>	if 'wildchar' is <Tab>: Do completion on the pattern in front of the cursor
c_<S-Tab>	<S-Tab>	same as CTRL-P
c_wildchar	'wildchar'	Do completion on the pattern in front of the cursor (default: <Tab>)
c_CTRL-I	CTRL-I	same as <Tab>
c_<NL>	<NL>	same as <CR>
c_CTRL-J	CTRL-J	same as <CR>
c_CTRL-K	CTRL-K {char1} {char2}	enter digraph
c_CTRL-L	CTRL-L	do completion on the pattern in front of the cursor and insert the longest common part
c_<CR>	<CR>	execute entered command
c_<CR>	CTRL-M	same as <CR>
c_CTRL-N	CTRL-N	after using 'wildchar' with multiple matches: go to next match, otherwise: same as <Down>
	CTRL-O	not used
c_CTRL-P	CTRL-P	after using 'wildchar' with multiple matches: go to previous match, otherwise: same as <Up>
c_CTRL-Q	CTRL-Q	same as CTRL-V (used for terminal control flow)
c_CTRL-R	CTRL-R {0-9a-z"%#*:= CTRL-F CTRL-P CTRL-W CTRL-A}	insert the contents of a register or object under the cursor as if typed
c_CTRL-R_CTRL-R	CTRL-R CTRL-R {0-9a-z"%#*:= CTRL-F CTRL-P CTRL-W CTRL-A}	insert the contents of a register or object under the cursor literally
	CTRL-S	(used for terminal control flow)
c_CTRL-U	CTRL-U	remove all characters
c_CTRL-V	CTRL-V	insert next non-digit literally, insert three digit decimal number as a single byte.
c_CTRL-W	CTRL-W	delete the word in front of the cursor
	CTRL-X	not used (reserved for completion)
	CTRL-Y	not used
	CTRL-Z	not used (reserved for suspend)
c_<Esc>	<Esc>	abandon command-line without executing it

c_<Esc>	CTRL-[	same as <Esc>
c_CTRL-_CTRL-N	CTRL-\ CTRL-N	go to Normal mode, abandon command-line
	CTRL-\ a - z	reserved for extensions
	CTRL-\ others	not used
	CTRL-]	not used
	CTRL-^	not used
c_CTRL-_	CTRL-_	when 'allowrevins' set: change language (Hebrew, Farsi)
c_<Del>	<Del>	delete the character under the cursor
c_<Left>	<Left>	cursor left
c_<S-Left>	<S-Left>	cursor one word left
c_<C-Left>	<C-Left>	cursor one word left
c_<Right>	<Right>	cursor right
c_<S-Right>	<S-Right>	cursor one word right
c_<C-Right>	<C-Right>	cursor one word right
c_<Up>	<Up>	recall previous command-line from history that matches pattern in front of the cursor
c_<S-Up>	<S-Up>	recall previous command-line from history
c_<Down>	<Down>	recall next command-line from history that matches pattern in front of the cursor
c_<S-Down>	<S-Down>	recall next command-line from history
c_<Home>	<Home>	cursor to start of command-line
c_<End>	<End>	cursor to end of command-line
c_<PageDown>	<PageDown>	same as <S-Down>
c_<PageUp>	<PageUp>	same as <S-Up>
c_<Insert>	<Insert>	toggle insert/overstrike mode
c_<LeftMouse>	<LeftMouse>	cursor at mouse click

==========

5. EX commands *ex-cmd-index*

This is a brief but complete listing of all the ":" commands, without mentioning any arguments. The optional part of the command name is inside []. The commands are sorted on the non-optional part of their name.

:!	:!	filter lines or execute an external command
:!!	:!!	repeat last ":!" command
:#	:#	same as ":number"
:&	:&	repeat last ":substitute"
:star	:*	execute contents of a register
:<	:<	shift lines one 'shiftwidth' left
:=	:=	print the cursor line number
:>	:>	shift lines one 'shiftwidth' right
:@	:@	execute contents of a register
:@@	:@@	repeat the previous ":@"
:Next	:N[ext]	go to previous file in the argument list
:Print	:P[rint]	print lines
:X	:X	ask for encryption key

:append	:a[ppend]	append text
:abbreviate	:ab[breviate]	enter abbreviation
:abclear	:abc[lear]	remove all abbreviations
:all	:al[l]	open a window for each file in the argument list
:amenu	:am[enu]	enter new menu item for all modes
:anoremenu	:an[oremenu]	enter a new menu for all modes that will not be remapped
:args	:ar[gs]	print the argument list
:argument	:argu[ment]	go to specific file in the argument list
:ascii	:as[cii]	print ascii value of character under the cursor
:autocmd	:au[tocmd]	enter or show autocommands
:augroup	:aug[roup]	select the autocommand group to use
:aunmenu	:aun[menu]	remove menu for all modes
:buffer	:b[uffer]	go to specific buffer in the buffer list
:bNext	:bN[ext]	go to next buffer in the buffer list
:ball	:ba[ll]	open a window for each file in the buffer list
:badd	:bad[d]	add file to the buffer list
:bdelete	:bd[elete]	delete specific files from the buffer list
:behave	:be[have]	set mouse and selection behavior
:blast	:bl[ast]	go to last file in the buffer list
:bmodified	:bm[odified]	go to next file in the buffer list that has been modified
:bnext	:bn[ext]	go to next file in the buffer list
:bprevious	:bp[revious]	go to previous file in the buffer list
:brewind	:br[ewind]	go to last file in the buffer list
:break	:brea[k]	break out of while loop
:browse	:bro[wse]	use file selection dialog
:buffers	:buffers	list all files in the buffer list
:bunload	:bun[load]	unload a specific buffer
:change	:c[hange]	replace a line or series of lines
:cNext	:cN[ext]	go to previous error
:cabbrev	:ca[bbrev]	like ":abbreviate" but for Command-line mode
:cabclear	:cabc[lear]	clear all abbreviations for Command-line mode
:call	:cal[l]	call a function
:cc	:cc	go to specific error
:cd	:cd	change directory
:center	:ce[nter]	format lines at the center
:cfile	:cf[ile]	read the file with error messages
:change	:ch[ange]	delete and insert lines
:chdir	:chd[ir]	change directory
:checkpath	:che[ckpath]	list included files
:clist	:cl[ist]	list all errors
:clast	:cla[st]	go to the specified error, default last one
:close	:clo[se]	close current window
:cmap	:cm[ap]	like ":map" but for Command-line mode
:cmapclear	:cmapc[lear]	clear all mappings for Command-line mode
:cmenu	:cme[nu]	add menu for Command-line mode
:cnext	:cn[ext]	go to next error

:cnewer	:cnew[er]	go to newer error list
:cnfile	:cnf[ile]	go to first error in next file
:cnoremap	:cno[remap]	like ":noremap" but for Command-line mode
:cnoreabbrev	:cnorea[bbrev]	like ":noreabbrev" but for Command-line mode
:cnoremenu	:cnoreme[nu]	like ":noremenu" but for Command-line mode
:copy	:co[py]	copy lines
:colder	:col[der]	go to older error list
:command	:com[mand]	create user-defined command
:comclear	:comc[lear]	clear all user-defined commands
:continue	:con[tinue]	go back to :while
:confirm	:conf[irm]	prompt user when confirmation required
:cprevious	:cp[revious]	go to previous error
:cquit	:cq[uit]	quit Vim with an error code
:crewind	:cr[ewind]	go to the specified error, default first one
:cscope	:cs[cope]	execute cscope command
:cstag	:cst[ag]	use cscope to jump to a tag
:cunmap	:cu[nmap]	like ":unmap" but for Command-line mode
:cunabbrev	:cuna[bbrev]	like ":unabbrev" but for Command-line mode
:cunmenu	:cunme[nu]	remove menu for Command-line mode
:delete	:d[elete]	delete lines
:delcommand	:delc[ommand]	delete user-defined command
:delfunction	:delf[unction]	delete a user function
:display	:di[splay]	display registers
:digraphs	:dig[raphs]	show or enter digraphs
:djump	:dj[ump]	jump to #define
:dlist	:dl[ist]	list #defines
:doautocmd	:do[autocmd]	apply autocommands to current buffer
:doautoall	:doautoa[ll]	apply autocommands for all loaded buffers
:dsearch	:ds[earch]	list one #define
:dsplit	:dsp[lit]	split window and jump to #define
:edit	:e[dit]	edit a file
:echo	:ec[ho]	echoes the result of expressions
:echohl	:echoh[l]	set highlighting for echo commands
:echon	:echon	same as :echo, but without <EOL>
:else	:el[se]	part of an :if command
:elseif	:elsei[f]	part of an :if command
:emenu	:em[enu]	execute a menu by name
:endif	:en[dif]	end previous :if
:endfunction	:endf[unction]	end of a user function
:endwhile	:endw[hile]	end previous :while
:ex	:ex	same as ":edit"
:execute	:exe[cute]	execute result of expressions
:exit	:exi[t]	same as ":xit"
:file	:f[ile]	show or set the current file name
:files	:files	list all files in the buffer list
:filetype	:filet[ype]	switch file type detection on/off
:find	:fin[d]	find file in 'path' and edit it
:fixdel	:fix[del]	set key code of <Del>
:function	:fu[nction]	define a user function

|:global|	:g[lobal]	execute commands for matching lines
|:goto|	:go[to]	go to byte in the buffer
|:grep|	:gr[ep]	run 'grepprg' and jump to first match
|:gui|	:gu[i]	start the GUI
|:gvim|	:gv[im]	start the GUI
|:help|	:h[elp]	open a help window
|:helpfind|	:helpf[ind]	dialog to open a help window
|:highlight|	:hi[ghlight]	specify highlighting methods
|:hide|	:hid[e]	hide current buffer for a command
|:history|	:his[tory]	print a history list
|:insert|	:i[nsert]	insert text
|:iabbrev|	:ia[bbrev]	like ":abbrev" but for Insert mode
|:iabclear|	:iabc[lear]	like ":abclear" but for Insert mode
|:if|	:if	execute commands when condition met
|:ijump|	:ij[ump]	jump to definition of identifier
|:ilist|	:il[ist]	list lines where identifier matches
|:imap|	:im[ap]	like ":map" but for Insert mode
|:imapclear|	:imapc[lear]	like ":mapclear" but for Insert mode
|:imenu|	:ime[nu]	add menu for Insert mode
|:inoremap|	:ino[remap]	like ":noremap" but for Insert mode
|:inoreabbrev|	:inorea[bbrev]	like ":noreabbrev" but for Insert mode
|:inoremenu|	:inoreme[nu]	like ":noremenu" but for Insert mode
|:intro|	:int[ro]	print the introductory message
|:isearch|	:is[earch]	list one line where identifier matches
|:isplit|	:isp[lit]	split window and jump to definition of identifier
|:iunmap|	:iu[nmap]	like ":unmap" but for Insert mode
|:iunabbrev|	:iuna[bbrev]	like ":unabbrev" but for Insert mode
|:iunmenu|	:iunme[nu]	remove menu for Insert mode
|:join|	:j[oin]	join lines
|:jumps|	:ju[mps]	print the jump list
|:k|	:k	set a mark
|:list|	:l[ist]	print lines
|:last|	:la[st]	go to the last file in the argument list
|:left|	:le[ft]	left align lines
|:let|	:let	assign a value to a variable or option
|:ls|	:ls	list all buffers
|:move|	:m[ove]	move lines
|:mark|	:ma[rk]	set a mark
|:make|	:mak[e]	execute external command 'makeprg' and parse error messages
|:map|	:map	show or enter a mapping
|:mapclear|	:mapc[lear]	clear all mappings for Normal and Visual mode
|:marks|	:marks	list all marks
|:menu|	:me[nu]	enter a new menu item
|:messages|	:mes[sages]	view previously displayed messages
|:mkexrc|	:mk[exrc]	write current mappings and settings to a file
|:mksession|	:mks[ession]	write session info to a file
|:mkvimrc|	:mkv[imrc]	write current mappings and settings to a file

|:mode|	:mod[e]	show or change the screen mode
|:next|	:n[ext]	go to next file in the argument list
|:new|	:new	create a new empty window
|:nmap|	:nm[ap]	like ":map" but for Normal mode
|:nmapclear|	:nmapc[lear]	clear all mappings for Normal mode
|:nmenu|	:nme[nu]	add menu for Normal mode
|:nnoremap|	:nn[oremap]	like ":noremap" but for Normal mode
|:nnoremenu|	:nnoreme[nu]	like ":noremenu" but for Normal mode
|:noremap|	:no[remap]	enter a mapping that will not be remapped
|:nohlsearch|	:noh[lsearch]	suspend 'hlsearch' highlighting
|:noreabbrev|	:norea[bbrev]	enter an abbreviation that will not be remapped
|:noremenu|	:noreme[nu]	enter a menu that will not be remapped
|:normal|	:norm[al]	execute Normal mode commands
|:number|	:nu[mber]	print lines with line number
|:nunmap|	:nun[map]	like ":unmap" but for Normal mode
|:nunmenu|	:nunme[nu]	remove menu for Normal mode
|:open|	:o[pen]	start open mode (not implemented)
|:omap|	:om[ap]	like ":map" but for Operator-pending mode
|:omapclear|	:omapc[lear]	remove all mappings for Operator-pending mode
|:omenu|	:ome[nu]	add menu for Operator-pending mode
|:only|	:on[ly]	close all windows except current one
|:onoremap|	:ono[remap]	like ":noremap" but for Operator-pending mode
|:onoremenu|	:onoreme[nu]	like ":noremenu" but for Operator-pending mode
|:options|	:opt[ions]	open the options-window
|:ounmap|	:ou[nmap]	like ":unmap" but for Operator-pending mode
|:ounmenu|	:ounme[nu]	remove menu for Operator-pending mode
|:print|	:p[rint]	print lines
|:pclose|	:pc[lose]	close preview window
|:perl|	:pe[rl]	execute Perl command
|:perldo|	:perld[o]	execute Perl command for each line
|:pop|	:po[p]	jump to older entry in tag stack
|:ppop|	:pp[op]	":pop" in preview window
|:preserve|	:pre[serve]	write all text to swap file
|:previous|	:prev[ious]	go to previous file in argument list
|:promptfind|	:pro[mptfind]	Search dialog
|:promptrepl|	:promptr[epl]	Search/Replace dialog
|:ptag|	:pt[ag]	show tag in preview window
|:ptNext|	:ptN[ext]	|:tNext| in preview window
|:ptjump|	:ptj[ump]	|:tjump| and show tag in preview window
|:ptlast|	:ptl[ast]	|:tlast| in preview window
|:ptnext|	:ptn[ext]	|:tnext| in preview window
|:ptprevious|	:ptp[revious]	|:tprevious| in preview window
|:ptrewind|	:ptr[ewind]	|:trewind| in preview window
|:ptselect|	:pts[elect]	|:tselect| and show tag in preview window
|:put|	:pu[t]	insert contents of register in the text
|:pwd|	:pw[d]	print current directory
|:python|	:py[thon]	execute Python command
|:pyfile|	:pyf[ile]	execute Python script file

|:quit|	:q[uit]	quit current window or Vim
|:qall|	:qa[ll]	quit Vim
|:read|	:r[ead]	read file into the text
|:recover|	:rec[over]	recover a file from a swap file
|:redo|	:red[o]	redo one undone change
|:redir|	:redi[r]	redirect messages to a file or register
|:registers|	:reg[isters]	display the contents of registers
|:resize|	:res[ize]	change current window height
|:retab|	:ret[ab]	change tab size
|:return|	:retu[rn]	return from a user function
|:rewind|	:rew[ind]	go to the first file in the argument list
|:right|	:ri[ght]	right align text
|:rviminfo|	:rv[iminfo]	read from viminfo file
|:substitute|	:s[ubstitute]	find and replace text
|:sNext|	:sN[ext]	split window and go to previous file in argument list
|:sargument|	:sa[rgument]	split window and go to specific file in argument list
|:sall|	:sal[l]	open a window for each file in argument list
|:sbuffer|	:sb[uffer]	split window and go to specific file in the buffer list
|:sbNext|	:sbN[ext]	split window and go to previous file in the buffer list
|:sball|	:sba[ll]	open a window for each file in the buffer list
|:sblast|	:sbl[ast]	split window and go to last file in buffer list
|:sbmodified|	:sbm[odified]	split window and go to modified file in the buffer list
|:sbnext|	:sbn[ext]	split window and go to next file in the buffer list
|:sbprevious|	:sbp[revious]	split window and go to previous file in the buffer list
|:sbrewind|	:sbr[ewind]	split window and go to first file in the buffer list
|:set|	:se[t]	show or set options
|:sfind|	:sf[ind]	split current window and edit file in 'path'
|:shell|	:sh[ell]	escape to a shell
|:simalt|	:si[malt]	Win32 GUI: simulate Windows ALT key
|:sleep|	:sl[eep]	do nothing for a few seconds
|:slast|	:sla[st]	split window and go to last file in the argument list
|:smagic|	:sm[agic]	:substitute with 'magic'
|:snext|	:sn[ext]	split window and go to next file in the argument list
|:sniff|	:sni[ff]	send request to sniff
|:snomagic|	:sno[magic]	:substitute with 'nomagic'
|:source|	:so[urce]	read Vim or Ex commands from a file
|:split|	:sp[lit]	split current window
|:sprevious|	:spr[evious]	split window and go to previous file in the argument list

|:srewind|	:sr[ewind]	split window and go to first file in the argument list
|:stop|	:st[op]	suspend the editor or escape to a shell
|:stag|	:sta[g]	split window and jump to a tag
|:startinsert|	:star[tinsert]	start Insert mode
|:stjump|	:stj[ump]	do ":tjump" and split window
|:stselect|	:sts[elect]	do ":tselect" and split window
|:sunhide|	:sun[hide]	same as ":unhide"
|:suspend|	:sus[pend]	same as ":stop"
|:sview|	:sv[iew]	split window and edit file read-only
|:swapname|	:sw[apname]	show the name of the current swap file
|:syntax|	:sy[ntax]	syntax highlighting
|:syncbind|	:sync[bind]	sync scroll binding
|:t|	:t	same as ":copy"
|:tNext|	:tN[ext]	jump to previous matching tag
|:tag|	:ta[g]	jump to tag
|:tags|	:tags	show the contents of the tag stack
|:tcl|	:tc[l]	execute Tcl command
|:tcldo|	:tcld[o]	execute Tcl command for each line
|:tclfile|	:tclf[ile]	execute Tcl script file
|:tearoff|	:te[aroff]	tear-off a menu
|:tjump|	:tj[ump]	like ":tselect", but jump directly when there is only one match
|:tlast|	:tl[ast]	jump to last matching tag
|:tmenu|	:tm[enu]	define menu tooltip
|:tnext|	:tn[ext]	jump to next matching tag
|:tprevious|	:tp[revious]	jump to previous matching tag
|:trewind|	:tr[ewind]	jump to first matching tag
|:tselect|	:ts[elect]	list matching tags and select one
|:tunmenu|	:tu[nmenu]	remove menu tooltip
|:undo|	:u[ndo]	undo last change(s)
|:unabbreviate|	:una[bbreviate]	remove abbreviation
|:unhide|	:unh[ide]	open a window for each loaded file in the buffer list
|:unlet|	:unl[et]	delete variable
|:unmap|	:unm[ap]	remove mapping
|:unmenu|	:unme[nu]	remove menu
|:update|	:up[date]	write buffer if modified
|:vglobal|	:v[global]	execute commands for not matching lines
|:version|	:ve[rsion]	print version number and other info
|:visual|	:vi[sual]	same as ":edit", but turns off "Ex" mode
|:view|	:vie[w]	edit a file read-only
|:vmap|	:vm[ap]	like ":map" but for Visual mode
|:vmapclear|	:vmapc[lear]	remove all mappings for Visual mode
|:vmenu|	:vme[nu]	add menu for Visual mode
|:vnoremap|	:vn[oremap]	like ":noremap" but for Visual mode
|:vnoremenu|	:vnoreme[nu]	like ":noremenu" but for Visual mode
|:vunmap|	:vu[nmap]	like ":unmap" but for Visual mode
|:vunmenu|	:vunme[nu]	remove menu for Visual mode

|:write|	:w[rite]	write to a file
|:wNext|	:wN[ext]	write to a file and go to previous file in argument list
|:wall|	:wa[ll]	write all (changed) buffers
|:while|	:wh[ile]	execute loop for as long as condition met
|:winsize|	:wi[nsize]	get or set window size (obsolete)
|:winpos|	:winp[os]	get or set window position
|:wnext|	:wn[ext]	write to a file and go to next file in argument list
|:wprevious|	:wp[revious]	write to a file and go to previous file in argument list
|:wq|	:wq	write to a file and quit window or Vim
|:wqall|	:wqa[ll]	write all changed buffers and quit Vim
|:wviminfo|	:wv[iminfo]	write to viminfo file
|:xit|	:x[it]	write if buffer changed and quit window or Vim
|:xall|	:xa[ll]	same as ":wqall"
|:yank|	:y[ank]	yank lines into a register
|:z|	:z	print some lines
|:~|	:~	repeat last ":substitute"

top - main help file

Vim Color Editor HOW-TO (Vi Improved with syntax color highlighting)

Al Dev (Alavoor Vasudevan)
alavoor@yahoo.com

v17.1, 28 June 2001

This document is a guide to quickly setting up the Vim color editor on Linux or Unix systems. The information here will improve the productivity of programmers because the Vim editor supports syntax color highlighting and bold fonts, improving the "readability" of program code. A programmer's productivity improves 2 to 3 times with a color editor like Vim. The information in this document applies to all operating sytems where Vim works, such as Linux, Windows 95/NT, Apple Mac, IBM OSes, VMS, BeOS and all flavors of Unix like Solaris, HPUX, AIX, SCO, Sinix, BSD, Ultrix etc.. (it means almost all operating systems on this planet!)

1. Introduction

10. Vi Tutorial

11. Vim Reference Card

Next Previous Contents

```
URL:         http://www.math.fu-berlin.de/~guckes/vim/howto/
URL:         http://www.vim.org/howto/ (mirror)
Created:     Fri Jun 12 12:00:00 CEST 1998
Last update: Wed Feb 23 18:00:00 MET 2000
```

Vim - HowTo Documents

Overview: Preparing Binaries | Reporting Bugs | Configuration | Dvorak editing | Editing C Code | Editing HTML Code (Webpages) | Editing Intro (Basic) | Editing LaTeX | Formatting Text | Functions | Help Vim and Vim Users | Mirroring Vim | Printing Text | Using RightLeft Mode | Using Tags | Teaching Vim | Testing Vim | Understanding Text Objects

Note: A *LOT* of material has been assembled already - but the time to include it all is really missing. If you think you can write HowTo yourself then *please* contact me - just send me a mail!

HowTos Wanted

- Vim and Sniff+ Support ("+sniff" - compilation requires files from the subdir "extra")
- Installation of a vim binary on - a special report for specific platforms is certainly welcome, too!
- Using functions
- Writing a syntax file
- Using a spelling checker (ispell)
- Using Vim on WindowsNT

If you can write and maintain any of these then please let me know!

Links

"HOWTO de l'éditeur Vim" [000223]

> http://www.freenix.org/unix/linux/HOWTO/Vim-HOWTO.html
>
> A HowTo in French.
>
> Authors: Al Dev (Alavoor Vasudevan) alavoor@yahoo.com; Version française par Arnaud Launay, alaunay@free.fr

Vim "QuickTute"

http://pegasus.rutgers.edu/~elflord/vim/quicktute.html

Author: elflord@pegasus.rutgers.edu

Send feedback on this page to
Sven Guckes guckes@vim.org

```
URL:         http://www.math.fu-berlin.de/~guckes/vim/macros/
URL:         http://www.vim.org/macros/ (mirror)
Created:     Mon Feb 22 18:00:00 CET 1999
Last update: Tue Feb 23 12:00:00 CET 1999
```

VIM Macros/Mappings

Well, I have hundreds of macros on my hands - but they will not be of much use unless they get documented, I think. Therefore this page has not much yet. Sorry.

TODO: Add macros from the VIM FAQ and from posts on comp.editors.

Sven's vimrc

Sven's vimrc has many examples of mappings. Take a look!

Mirror/reverse Lines

by Dr. Charles E. Campbell Jr. (cec@gryphon.gsfc.nasa.gov)

Description: This script "mirrors" (reverses) the contents of the text between (the line marked with) 'a' and the current line. The script is sourced with a map to "\fm". Depending on the kind of map for "\fm" the order of the lines will remain or be reversed, too.

Instructions: Let your shell export VIMSCRIPT which points to the directory where you keep your scripts for Vim, eg "export VIMSCRIPT=/Vol/Pub/share/vim/syntax".

Define this map: map \fm :'a,.g/^/:so $VIMSCRIPT/mirror.vim<CR> Here is the "mirror.vim" that should get sourced:

```
"mirror.vim script : "mirror images" the current line
"  Uses my mz "y
normal ^my$mz
"
if col("'y") < col("'z")
  " note: the first time, mz is at $, and when the character
  "       at 'a is deleted, mz travels backwards with $
  normal `yl"yd`y`z"yphmzl"yd`z`y"yP`ylmy
  endif
"
while col("'y") < col("'z")
```

```
  " note: normally, marks will remain in the same column,
  "       and will *not* travel with the marked character
  "       during deletions
  normal `yl"yd`y`zhmz"yphmzl"yd`z`y"yP`ylmy
  endwhile
```

Go to some line, ma. Move to end-of-block. These lines will now be mirror-imaged.

Example: Applying this self-referentially: :yllaitnerefer-fles siht gniylppA Using "\fr" in a similar manner will also reverse the lines:

```
        map \fr   o<Esc>mz'aO<Esc>ma:'a+1,'z-1g/^/m 'a<CR>'add'zdd
```

Send feedback on this page to
Sven Guckes guckes@math.fu-berlin.de

VIM - Questions and Answers

The following questions have been answered by now. Thanks to everyone who helped!

Please read the part about "terminology" below in case you are in doubt about terms used within the text here. And do not hesitate to send in more answers, improvements, and corrections to typos. Thanks!

Color Setup

[001010] Vim allows to change colors for quite a few things, such as the color for normal text ("Normal"), selected text ("Visual"), search matches ("Search"), commented text ("Comment"), and all the characters that represent non-existing or hidden parts of the text. (such as TABS, trailing whitespace, end-of-lines, and the tilde lines after the end-of-buffer).

I've thrown in "LineNr term=NONE" to turn off the default underlining of line numbers as

```
" setup by David Risner drisner@eskimo.com
" http://david.risner.org/ [001010]
set background=dark
highlight Comment guibg=Black    guifg=Red    gui=italic
highlight LineNr  term=NONE
highlight NonText guifg=Gray95   guibg=Black  gui=bold
highlight Normal  guibg=Black    guifg=Gray95
highlight Search  guibg=Cyan     guifg=Black  gui=underline
highlight Visual  guibg=Yellow   guifg=Black  gui=underline
```

Encryption and Decryption - :X command

000915 Vim offers builtin encrpytion and descryption with the command ":X".

Frequent Problem: Users encrypt files but forget the key; therefore they cannot get their data back.

FAQ: "How can I get my data back? Can you decrypt my data for me?"

Usual Answer: When you lose your key then you are lost. Encrpytion is meant to keep people from decrypting your data ho do not have the correct key. In this case this means you. Sorry!

Solution: DONT LOSE YOUR KEYS! (As simple as that. ;-)

Prevention: (1) Use the correct escape: An empty input! Just press the RETURN key so there will be no input. Without an input (read: "key") there can be no proper encryption, of course. With Vim you can also type ESC to abort the current command. Pleas enote however that the command will still be added to the command line history - feature!

(2) Remap ":X"! The problem is often caused by a typo: The user tries to "exit+save" with ":x" but types ":X" instead. You can map ":X" to ":x" easily: map :X :x .. and this should prevent the accidental input of ":X".

See Also:

- :help encryption
- :help :X

From: bgriffin@ic.sunysb.edu (Eli the bearded) Newsgroups: comp.editors Subject: Re: Encryption in vi - Help! Date: 6 Mar 1996 07:23:39 GMT Message-ID: <4hjehr$4v4@abel.cc.sunysb.edu> vi uses the same encryption as the Unix crypt command. This is a single rotor version of the enigma machine maping 8-bit chars into 8-bit chars. Since text files are a small subset of 8-bit chars, decryption of plain text files is a relatively easy task. [...] Unix crypt is a very poor crypto system. When it was written it was merely low security (with a bonus for security through obfuscation), but computational power has increased vastly since then as have the number of people interested in launching attacks against it. You should never consider it more secure than a small padlock on a diary. (Note this crypt is a different one than the crypt(3) function used for encrypting passwords in a vanilla passwd file. crypt(3) is a one way encryption based on DES. I am not convinced of its security, but I don't know of any attempted attacks on it.) From: Kevin Dempsey Peterson <kevin@cafe.berkeley.edu> Newsgroups: comp.editors Subject: Re: vi emergency Date: Mon, 20 Oct 1997 15:59:27 -0700 Message-ID: <Pine.GSO.3.96.971020155614.8935I-100000@cafe.berkeley.edu> Crypt isn't DES. The crypt(3) call is DES, but the crypt command is a weakened version of the Enigma machine. Try http://www.dodgenet.com/~kwantam/zip/cbw.zip -- kevin@cafe.berkeley.edu Home: (510)665-9670 http://www.ocf.berkeley.edu/~peterson Page: (510)726-8960 The URL to "kwantam" is obsolete. But I found the "cwb.zip" here:

- ftp://ftp.ecst.csuchico.edu/users/atman/CBW/cbw.zip
- ftp://ftp.ox.ac.uk/pub/crypto/cryptanalysis/cbw.tar.gz [1994-09-19; 188529 bytes]

The file "Read.me" says that the archive contains CBW, the "Crypt Breakers Workbench .. originally written by Robert W. Baldwin at MIT."

- ftp://ftp.ecst.csuchico.edu/users/atman/CBW/cbw.txt

In this text, Norman Richards (orb@cs.utexas.edu) says on 1992-12-18:

> "cbw is used as a tool to decrypt text files that have been encrypted via the standard unix crypt command. The program breaks the message down into 256 byte blocks. The encryption program itself (to quote the docs) is effectively encrypted by a single rotor enigma system."

So, maybe CWB can help you decrypt your accidentally encrypted data. I have not tried it. Feedback is welcome. -- But please respect the privacy of others. Thankyou!

Vim vs CAPS

The Windows versions of Vim have a problem with the CAPS key: When the CAPS key is down/pressed/ON then typing the numeric keys on the normal keypad (ie not the extra numeric keypad) will produce the upper keys, too. This is *wrong*.

Mark Johnson had produced a patch for Vim in 1998

Vim on WindowsNT

Fighting the Registry

```
Problem:  Opening files with spaces in the filename does not work.
Solution by Scott Borton scottb@uccs.com [980807]:

1. start regedit (start -> run -> "regedit")
   NOTE: we are now in the "Registry Editor" window...
2. edit -> find... -> "gvim"
   This finds the key class 'My
   Computer\HKEY_CLASSES_ROOT\txtfile\shell\open\command'.
```

```
   NOTE: I had previously mapped .txt files to be opened by gvim
   via Explorer's "Open With" facility.
3. In the Registry Editor data window, double click on:
   'ad(default)  ""C:\Program Files\vim-5.1\gvim.exe" %1"
   This exposes the "Edit String" dialog
4. Change the "data value"
   from:    C:\Program Files\vim-5.1\gvim.exe %1
   to:      C:\Progra~1\vim-5.1\gvim.exe "%1"

   This specification doesn't require quoting of the executable string
   and allows us to quote the argument string which is what we want.
   The nice thing about this is that new mappings of file extensions
   to gvim seem to inherit this execution string.
```

Online Webster - requesting definitions of words

Every now and then you maybe have to look up the definition of a word. Here is a simple way to send a request to an "online webster": #!/bin/sh www -n "http://work.ucsd.edu:5141/cgi-bin/http_webster?isindex="$* | more exit 0 Where "www" is the name of the program of your favourite web browser, eg:

```
$ ls -l `which www`
[...] ~/bin/www -> /path/bin/lynx
```

Tip submitted by: t.byfield tbyfield@panix.com [980529]

Vim on CDE

On CDE (Common Desktop Environment) the default terminal is "dtterm". There are some problems with it.

Q: Starting vim overwrites the screen. Why? can I change this?

A: Dtterm does not support xterm's "alternate-screen controls" (at least it is not mentioned in the manual), so you can't fix it. (This is not a vim/vi/whatever problem.) If you want a terminal with an "alternate screen" then you should get "xterm". As this terminal ships with many systems you probably already have it. The XFree86 3.3 xterm supports ANSI color and VT220 emulation Please take a look at the FAQ:

- http://dickey.his.com/xterm/xterm.faq.html

Thomas E. Dickey dickey@clark.net [971031,000605]

Mapping the Escape Key (ESC)

Fact: Typing an ESC on the command line "aborts" the command. Q: Why is it that an ESC within a command line mapping *executes* the command on the command line? A: This was done to make Vi mappings work, while at the same time fix that Esc should abort the current command (as is stated in most Vi manuals). See also: ":help c_<Esc>".

Hint: To abort the current command on the command line, use CTRL-C:

cmap <c-g> <c-v><c-c>

Windows: Replacing Notepad with Vim

WARNING: Although you *can* replace the Notepad with a copy of Vim you are advised **not** to do so as this breaks printing. Windows uses the notepad with the parameter "/p" to print texts. Vim however, does not understand this parameter, and thus you could lose the "internal printing".

You are advised to "map" the opening of text files to Vim using the Windows Explorer ("View->Options->File Types"). This can be tedious but should be better.

File extensions to map would be:

```
.C
.CPP
.H
.TXT
```

If you still think that you must do the replacement then here are more ideas:

Do not replace the notepad with vim by copying it over!
THIS IS WRONG: `copy c:\vim\vim.exe c:\windows\notepad.exe`
Why: This causes Windows to lose track of the default location of the desired application.

So you should go about it like this:

Make backup of notepad

```
copy c:\windows\notepad.exe c:\backup\notepad.exe
```

Note:

Make copy of Vim as "notepad"

```
copy c:\vim\vim.exe c:\tmp\notepad.exe
```

Replace notepad with new copy

```
copy c:\tmp\notepad.exe c:\windows\notepad.exe
```

Prompt "replace"? Answer with "yes".

Gregory Harris gregoryh@netscape.com [971231]:

Win95/NT4:

- Open any folder
- Select "View|Options"
- Select "File Types"
- Choose "text document"
- Select "Edit..."
- Select "Open"
- Select "Edit..."
- Change the "Application used to perform the Action" field with the exact path of vim.exe.

Another description from Joshua Rodman (joshua@wrs.com) [980116]:

"The answer is not quite as simple as it may seem. There are two places where mappings between extensions and programs are made.

```
The "old" win31/NT3 way is done with the aid of the file manager "winfile".
The "new" win95/NT4 way is done with the aid of the windows explorer.
```

Mostly setting the method which matches your OS version will cause the behavior to happen, but I sometimes end up

finding it necessary to set both depending upon what the launching application is doing.

Both these settings are, of course, set in the registry.

The Win31/NT3 way:

- Run file manager (START->RUN->Winfile under NT4/95)
- Find and select a file of the appropriate type
- From the File menu, select Associate...
- Vim will most likely not be on the list of programs to "Associate With:" so select the browse button, and find vim.exe with the resulting interface. Alternatively, instead of a flat association, you could select New Type and associate a laundry list of extensions with Type: Text and the program Vim.

Win95/NT4 way:

- Open any explorer window.. ie.. double click on any drive, folder, etc.
- From the resulting window, select the File->Options.
- Select the File Types tab
- Scroll down to find the type Text Document.
- Select Text Document and click Edit.
- In the resulting dialog, select OPEN from the actions list, and click Edit.
- Under "Application used to perform action:" type in the path to your vim.exe, or click the Browse button to graphically select the location of your vim.exe.
- Note that if you do not want ALL these extensions opening VIM you will need to delete the type and reconstruct multiple replacements. Windows is _VERY_ inflexible about the editing of types.

000904: Matthew White mwhite@csu.edu.au comments on "replacing notepad with vim" on a Windows2000 machine:

> You can do it but you need to delete `notepad.exe` from the directory `/winnt/system32/DllCache`. (This directory is probably hidden.) Once this has been done you can copy vim as `notepad.exe` over the top of /winnt/notepad.exe and also `/winnt/system32/notepad.exe`.

So - do these things work? Or are there any problems with this? Please let me know! Sven guckes@vim.org

Thanks to these people for their help:

```
Matthew White        mwhite@csu.edu.au [000904]
Joe Tschida     jtschida@wavefront.com [971231]
Gregory Harris   gregoryh@netscape.com [971231]
Ron Aaron           ron@mossbayeng.com [980116]
```

Printing Text Files

Printing Text with Line Numbers

Once in a while it is nice to print a file with line numbers, eg when you want to compare files with somebody and need to talk about that "statement in line, uh, ..", well, you know.

So how do you print a file with line numbers? Well, there are several ways to kill a cat... Apropos, here's one way:

```
cat -n filename
```

And then there is "pr" ("convert text files for printing"):

```
pr -tn' 3' file
```

[970910]

Vim on DOS

[970818] Vim on DOS takes about 400K. And when you use the "shell" it might not work because there is not enough space left for the shell. In that case you simply have to cut down on some extra programs to gain the space you need to "shell out" from Vim.

VIM GUI - Athena and Motif distribution

For installing VIM GUI you need "Athena and Motif". If you have a standard X11 distribution installed, then the Athena libraries should be there. Motif is commercial and a license needs to be bought.

But there is an option: Get Lesstif (a Motif clone) and is available for free:

```
X11 distrib: ftp://ftp.x.org/
Lesstif:     http://www.lesstif.org/
Motif:       http://www.rdg.opengroup.org/
```

Quoted Printable (QP code)

Problem: QP encoded text, eg when replying to a QP encoded mail or post.

Solution: Highlight the text and then use ",QPD" to decode. And ",QPE" to encode. Of course you need the following definitions and the associated scripts "qpdecode" and "qpencode":

vmap ,QPD :!qpdecode vmap ,QPE :!qpencode script "qpdecode": perl -pe 's/\=([0-9A-Fa-f]{2})/chr(hex($1))/ge; s/\=\n//;' script "qpencode": #!/usr/bin/perl -p s/([^\n\t -~]|[
\t]$)/'='.sprintf("%02X",ord($1))/ge;s/(.{70}[^=\n]{2})/$1=\n/;

Parenthize text

Parenthize current word: map ,, maa0<ESC>mbbma$a x<ESC>`awgebi(<ESC>ea)<ESC>$xx`blx`a

---

Long and short mappings and timeout

Q: I have defined a mapping for "longcommand" - but why does it work sometimes and sometimes it doesn't?
A: The longer the command name the longer it takes to type in. And if you pause in between for more than the "timeoutlen" then it is considered that you write text and thus you did not complete the mapped key sequence. Just type with less pauses or increase the pause. ;-) You can also define an abbreviation for command line as this is independent of timeouts.

VIM - "vimrc"

A program without a configuration usually is hard to use. That's why you should know about VIM's setup file.

The usual way of using a setup file for UNIX programs is to put a file into the home directory. Most programs read commands from this file, hence the suffix for these files is "rc". The filename starts with a dot (".") as these files do not get listed with a simple "ls" command. Setup files usually are not looked at often, and there are quite a lot of them.

So the setup file for vim is the file $HOME/.vimrc, abbreviated as "vimrc".

When talking about the setup file of a program then the location $HOME is often left out as well as the dot prefix of the filename. So we talk about the "vimrc" instead of "$HOME/.vimrc". Sometimes we will just say "rc file" or even "rc".

NOTE: Actually there are more init files involved. As vi is based on "ex" it also uses "$HOME/.exrc".

NOTE on exrc: Some version of vi seem to choke on empty lines in the exrc.

There are three kinds of things which are defined in the vimrc:

Settings ("set")
: The command "set" changes the values of the internal variables. The variables affect the way vim works.

Abbreviations ("ab")
: An abbreviations is a word which is to be replaced by another word. (What did you expect? :-)

Mappings ("map")
: A mapping defines a new command. You can change the internal commands ("builtins") that way or create new commands.

Sample Settings

Sven's vimrc is available as http://www.vim.org/rc

Sample Abbreviations

Abbreviations are quite simple word to word translations. Very handy when you type the same things over and over again. It is useful for long words like this: "Donaudampfschiffahrtsgesellschaftkapitaenwitwengesetzzusatzparagraph" This actually is a proper German word. Don't ask! The abbreviation for this word might be "DON" and the definition in the vimrc would look like this: ab DON Donaudampfschiffahrtsgesellschaftkapitaenwitwengesetzzusatzparagraph

I also use abbreviations for Email addresses, important file name such as setup files, and some URLS of my web pages.

Examples: ab _vimrc $HOME/.vimrc ab _mymail guckes@math.fu-berlin.de ab _homepage http://www.math.fu-berlin.de/~guckes/ NOTE: It does not matter how many spaces there are between the abbreviation and the expanded word. This allows for some nice formatting. Abbreviations can be a lot more than mere word substitutons, though. Actually anything you type can used. [TODO: HTML example]

Sample Mappings

Mappings make the use of VIM more interesting yet. Complex commands be made with just a few keystrokes.

Caveat: Mapping must be "prefix free", ie no mapping must be the prefix of any other mapping. Example: "map ,abc foo" and "map ,abcd bar" will give you the error message "Ambigous mapping". " 950101 ,v = vimrc editing (edit this file) map ,v :e $HOME/.vimrc " 950101 ,u = "update" by reading this file map ,u :source $HOME/.vimrc " " 950101 g = goto first line " 951112 the new vim defines "g" as a new command, eg "gg" will go to first line " map g 1G " " === Text editing " " - Formatting " " 950330 ,dp = dequote current paragraph map ,dp {jma}kmb:'a,'bs/^> *// " 950330 ,qp = quote current paragraph map ,qp {jma}kmb:'a,'bs/^/> / " " - Inserting texts " " 950101 ,c = "copy notice" (tell addressee that text is a copy of an article) "map ,c 1G}:r COPY " 950101 ,re = read ELMSIG map ,re G:r ELMSIG " 950101 ,da = "date insert" map ,da :r!date " 950101 ,dt = "date and time insert" map ,dt :r!date +\%y\%m\%d\%t\%T " 950101 ,h = "helloagain" (indicates reply to reply) map ,h 1GOHello, again! " 950101,950620 ,C = "check alias" " 950101 ,s = "sign" - read in signature file (requires manual completion) map ,s :r $HOME/public_html/sig/sig. " 950101 ,t = "text" - read in text file (requires manual completion) map ,t :r $HOME/public_html/txt/ " "END

Map Example - Ninety-nine Bottles of Beer

[...]

Increasing a number with Vi

This macro increments the number the cursor is on:

```
" marks with z , executes register x.
map +   mz"xy G\+w
map \+w or0^Mr9^Mr8^Mr7^Mr6^Mr5^Mr4^Mr3^Mr2^Mr1^Mr0^M^M?^[\+x
map \+x "xp"xdd@xbk"xdd11L11dd`z@x`z^L
```

With VIM you just position the cursor on the number, enter the number to add followed by ^A. the number 100 thus changes to 123 after the command "23^A". [960927]

Viewing file with continuous update

```
[990628]
Situation:  You look at a file that contains the output of a program.
The program continuously changes the file (usually by *appending* data).
Q: How do I know that there is an update on the file without leaving vim?
A1: (from within Vim)
Load the latest version of the file into the edit buffer with the
command ":e!".
A2: (external) Use "tail -f" on the file in another window.
A3: (external) Use "less _F" on the file in another window.
    From the manual to the pager "less":
        F       Scroll forward, and keep trying to read when the end of
                file is reached. Normally this command would be used when
                already at the end of the file. It is a way to monitor the
                tail of a file which is growing while it is being viewed.
```

```
                (The behavior is similar to the "tail -f" command.)
A4: Add these autocommands for the filename pattern PAT:
        :au FileChangedShell PAT e!
        :au CursorHold       PAT e!
        :au FocusGained      PAT e!
If you only want to view logfiles with syntax-highlighting
then search for "colortail" on http://www.freshmeat.net!
You may wish to add ithat the cursor automatically jump to the end of the
buffer, too (by adding ":norm G" [without the quotes]).
[Thanks to Thomas Köhler jean-luc@picard.franken.de 990704 for this]
```

The Angle Notation

[960927]

```
Q: Why does "<CR>" not chaged to a ^M with :autocmd ?
A: The <> notation is only recognized with the :map command.
   You cannot use it with :autocmd .
Workaround:  Use a :cmap to define a command and use it for your :autocmd.
Example:
        :cmap _addbufmenu :menu Buffers.% :buffer %
        :au! BufReadPre * normal :_addbufmenu
One remaining problem: file names with a "." in them will cause a submenu to
be created.  This needs some extra stuff to put a backslash before the ".".
This is left as an exercise to the reader... :-).
Solution:
:cmap _addbufmenu :menu Buffers.1 :buffer %
:autocmd! BufNewFile,BufReadPre * normal O^["%p0:s/\./\\./g^M"1yydd:_addbufmenu
For the second line the '^[' and the '^M' is actually cntrl-v, Escape and
cntr-v, cntrl-m. Bit of a hack, but works just fine.
To avoid that, more can be moved to the mapping.
nmap _xaddbufmenu O%:s/\./\\./g0"9y$u:menu Buffers.9 :buffer
%
autocmd BufNewFile,BufReadPre * normal _xaddbufmenu

One remaining problem:  The current yank register is changed.
"Y:np" doesn't work as expected.
```

VIM within SUN's shelltool

Vi in a shelltool on Sun switches from insert to command mode on pressing an arrow key because the byte sequence resulting from pressing the key starts with , and vi doesn't interpret the sequence. VIM, however, does, and you can use arrow keys to move in your text while remaining in insert mode. [960622]

Delete/Remove commands

Q: How do I delete blocks?
A: :g/first/,/last/d NOTE: This assumes that all these blocks have the same string "first" in the first line as well having the same string "last" in the last line. All lines in between are skipped!!

Q: How do you remove all empty lines?
A: :g/^$/d

Q: How do you remove all lines with whitespace (spaces or tabs) in them?
A: If the lines contain

Q: How do you "squeeze" a range of empty lines to a single empty line?
A1: :v/./,/./-1join to compact 2 or more blank lines into just one. [Rick Hellicar (hellicar@prl.philips.co.uk)]
A2: map _b GoZ^[:g/^[^I]*$/,/[^ ^I]/-j^MGdd Note that this mapping inserts an extra line at the end of the buffer and removes it after the global command is done. This trick makes it possible that the global command works on the last part of the buffer, too. [Robert Webb (robertw@wormald.com.au)]
[960326]

Mapping the TAB key

[960311] Q: How do you map the tab key?
A: :map ^V^V^V^I right-hand-side

Mapping the ESC key

On some keyboards the escape key is not placed place very well. It's either part of the numeric block or it can be a small button (as with some Macintosh keyboards). But do not despair - make your own escape key! You can map one of the commands keys you do not need to ESC. Example: map CTRL-o to ESC: :map That's ":map CTRL-vCTRL-o CTRL-vCTRL-ESC".
[960311]

Helpfile (vim_ref.txt)

Problem: After relocation of the documentation, vim cannot find the help text.

Question: How do you tell vim where the helpfile is?

Answer: To tell vim where to find the help file, set the variable "helpfile" to the correct value, ie including the full path. As with most variables you can abbreviate "helpfile to "hf".

Example: :set hf=/path/vim_ref.txt

VIM - Invocation / Startup

The easiest way to start "vim" is to simply enter "vim" at the prompt of your shell. If your shell can find "vim" then it will start it.

Now vim should show you an "empty buffer". A "buffer" is a part of the computer's memory where only vim may change data. While editing you change the contents of this buffer. As you are not making any changes to a "file" on your "file system" you make changes to something "virtual". To apply these changes to a file you have to give the "write" command.

Before we can go on I will have to explain how the programs tells your "text imput" from the input it has to interpret as "commands".

VIM - Command Mode vs Append/Insert/Open/Replace Mode

Vim is a "modal editor", ie deepending on the current mode the keys you type have a different meaning. When in command mode, the keys are interpreted as commands; most commanly these either move the cursor ("jumps") or they change the text (delete/insert).

After startup Vim is in command mode. Type 'i' to switch to insert mode; now all the keys are simply inserted as text - except ESC which is used to "escape" from insert mode and switch back to command mode.

Exiting Vim is done with the command ":x", ie you type ':' to switch to "ex mode", followed by the 'x' (think of "exit") and then you enter this command by typing the 'return' key. The exit command will automatically save the changes you made back to the associated file.

NOTE: When you are on command mode then typing ESC will not exit that mode but stay right in it. As feedback on this you do not get an error message but a "beep". When the machine or the response is slow then you will find yourself in the situation that you are not sure whether your ESC has been received and has changed the mode, so you might type ESC again to make sure. This can result in two ESC being sent where the first does change the mode and the second results in a beep. Don't worry - no data is lost, nothing is broken. Just ignore the beep! However, the beeping can be quite annoying to people around you, so you can turn it off with the command ":set noeb vb t_vb=". You might want to put this into your setup file, too.

VIM - Online Help

[960311,990223] One of the best improvements over Vi is the "online help". Just enter the command ":help" (press return key to "enter"!) and VIM will open a window with the help text in it.

You can also give a word after the help command and Vim will try to find something that matches this word. Actually, Vim tried to make a match on the words within an "index" of all help documents, known as "tags". This index is thus also known as *the* "tags file".

You can also just enter some prefix of a tag and let Vim show you all possible completions that it finds in the "tags file". Just enter ":help prefix" and type a CTRL-D after that.

Example: You are looking for info about "formats" or "formatting"; so enter ":help s" followed by the CTRL-D. This would show something like this on the screen: :help format<CTRL-D> 'formatprg' Unix-format 'nrformats' formatting file-formats 'grepformat' 'formatoptions' DOS-format-write 'fileformat' format-comments Mac-format-write errorformat format-bullet-list Unix-format-write 'fileformats' Mac-format dos-file-formats 'errorformat' DOS-format

tags-file-format errorformat-changed The ":help" command for Vim-3 just shows an index of help pages. All help pages can be accessed directly by pressing the associated letter - except for 'a' and 'b': The letter 'a' gives you the index page, the letter 'b' is the command to move "back" to the previous help page.

Jumping back to marked line *and* position

Q: How do I get back to the exact position within a line I have marked with 'a'? A: Use "`a" (that's a backtick!).

Display of current command input

Q: How I see what I type? A: Use ":set showcmd".

VIM - DOS/Windows version

Q: Why does gvim-Win32 show the ISO (Unix) character set?
A: Bram: I think only the Win32 GUI version uses the ISO (Unix) character set. All other DOS and Windows versions use the DOS character set. Currently there is no way to switch character set. I would not know how to do that under DOS or Windows. For the Win32 GUI it should be possible, if I can find the documentation for it. [970911]

Q: Who can I ask about the DOS/Windows version? A: Here is a short list:

Win32 (Windows NT and Windows 95) version

George V. Reilly (gvr@halcyon.com>

OS 16-bit real-mode version

Any volunteers?

DOS DJGPP 32-bit protected-mode version

Any volunteers?

If you are using the DOS version of VIM then you might get strange screen flashing or color changing. The reason could be that "highlighting" is on. Use the command ":set hl=" to turn highlighting (hl) off.

Q. How do you paste into Vim when running Windows 95?
A. In the properties dialog box for the MS-DOS window, go to "MS-DOS Prompt/Misc/Fast pasting" and make sure that it is NOT checked.

VIM DOS/Windows - Shell invocation

The following commands to show a dir listing fail because the "!" invokes a shell, so there is no need for "sh" or "command.com": :!sh dir/w :!command/c dir/w :!\command.com/c dir/w :!csh dir/w The proper way is: :!dir /w To start command.com use ":!sh".

Changing line feeds from DOS to UNIX

Situation: You edit a file with DOS end-of-lines (EOL), ie. lines end in two characters - ASCII#13 ("carriage return" aka "CR") followed by ASCII#10 ("linefeed" aka "LF"); but you want to turn this file into a Unix file, for which the EOL is only a linefeed (ASCII#10, LF).

Q: How to turn (13,10) into (10)? A: Assume that the file is in the current buffer. Use the power of vim: :set fileformat=unix :w Or use the power of Perl: An empty perl script should do the trick as Perl automatically writes files with the correct eol character for the system it's being run on: :%!perl -pe ";"

VIM - Backup Files

By default, backup files are maintained. That is, editing 'file' will create 'file.bak' in the current directory. This can be turned off with ':set nobackup nowritebackup'. By default, a swap file is created in the current directory for crash recovery purposes. Vi puts its crash recovery file in /tmp. To make vim do something like this, use ':set dir=>/tmp'. The '>' stops it from trying the current directory first. [950306]

VIM - Tranposing characters and lines

A frequent typing error is tranposing two characters, ie you type them in the wrong order. Example: You type "in" as "ni". So what is the easiest method of changing this? Some editors have a "transpose command" usually bound to "control-t". However, this is no standard. And "vim" does not have it. However, it is as easy as deleting the first charcter and placing it after the second. You can do this by placing the cursor on the first character and then use the command sequence "xp".
Example: "ni" Put the cursor on the "n". Use "x" to delete the "n". Note: The cursor will be placed on the next charcter, ie "i". Use "p" to "put" the deleted text (the "n") after the current position ("i"). Result: "in" You can also start on the second character and use "Xp". Command "X" deletes the character *before* the current character. Tranposition of two lines works in the same way, ie delete one line and put it after/before the adjoining one. The command sequences here are "ddp" and "ddP". [950306]

VIM - Paragraph formatting

How to format a paragraph (block).

- step1: Go to first line (character) of the paragraph to format.
- step2: Press "Q". Note: A capital "Q" - not a lower case "q".
- step3: Press "}". (On the Mac keyboard with German layout: alt-9)
- => paragraph formatted (hopefully)

You can also format text by calling an external program. The nicest formatter I have seen so far is "par" - see the notes on the Vim Utilities Page.

---

1) Type (esc) to switch to command mode 2) Enter the command ":help" (press return key to "enter"!). VIM will show you an index of help pages. All help pages can be accessed directly by pressing the associated letter. <xmp> On msdos, if you get any strange screen flashing or color changing, try: :set hl= This will turn highlighting (hl) off. By default, backup files are maintained. That is, editing 'file' will create 'file.bak' in the current directory. This can be turned off with ':set nobackup nowritebackup'. By default, a swap file is created in the current directory for crash recovery purposes. Vi puts its crash recovery file in /tmp. To make vim do something like this, use ':set dir=>/tmp'. The '>' stops it from trying the current directory first. [950306]

Substitute in line range

Q: How do you substitute from line marked with x to a line marked with y ? A: :'x,'ys/this/that/

Reading in a file

Q: How do I append a signature to my text? A: Jump to the end of the text with "G" and then read in the signature file. Q: How do I read in a file to the current buffer? A: Use ":r file". The contents of "file" will be appended *after* the current line. Q: How do I read in the output of a command? A: Use ":r!command".

Abbreviations

Q: Why cant I abbraviate "'xy"? A: The abbreviation consists of a non-id character followed by two id characters, which does not satisfy either category of a "full-id". However, "_ps" and "'p" will work.

Highlight mode jumping

Q: Are there commands to jump to the beginning/end of the highlighted text?
A: Yes, command 'o' will jump to the beginning/end of the highlighted text.

Completion of negated settings

Q: Why does completion of ":set n" not show negated settings, eg "noautoindent"? Completion of ":set no" seems to work.
A: The thing is that the "no" is not actually part of the option's name, the name comes after that. So after "no" vim knows to complete any boolean setting name (starts the completion just after the "no", which is not part of the name). After "n", vim will complete all setting names starting with "n". It would be a bumber if you wanted to complete "number", but had to wade through all the boolean option names with "no" prepended too.
Answer by Robert Webb robertw@wormald.com.au, author of the completion code.

Formatting text

vim4 allows to format the current paragraph with "Qp". The "Q" means "format" and the following "p" will let the formatting operate on the current paragraph. The really good thing is that it preserves "quoting".

Note: Since Vim-5 the command for the "internal text formatting" has changed to "gq" since the 'Q' was needed for Vi compatibility.

However, the builtin formatting command does not suit everybody. There are a few scripts about which you can use to pipe your text through. Here are two scripts - one in PERL and one in AWK:
mfmt.pl
fmt.awk
Basic knowledge about PERL and AWK required. Caveat: scripts are not tested yet! use at own risk! [TODO: test the

scripts]

Insert mode not local to window

Someone reports: vim does not remember the mode of a window, i.e if you are editing two files, one in command mode, and the other in inset mode, and changing from the file edited in insert mode to the file edited in command mode, by cliking with the mouse in file edited in command mode this file is = now edited in insert mode.

MrVIM answers: This is intentional. Mode does not stick with a window. The mode is global to all windows. This is different from some other Vi clones, I suppose (some also have a different command line for each window).

Digraphs

Q: Is there a command to remove any or all digraphs? A: No. There is a table that is defined at compile time. You can only add new ones. Adding a command to remove digraphs is on the todo list.

Starting in insert mode

Some people like their editor to be in insert mode after the start. Q: How I do let vim start in insert mode? A: :set insertmode This should go into the vimrc of course. ;-)

Turning off the beep

If you want vim to stop beeping then all you need is ":set vb" which tries to do a screen flash rather than an audible beep. Some terminals can't do screen flashes, but if yours does and you don't want it to flash or beep then use ":set vb t_vb=".

NOTE: In version 3.0 you would get some characters on the command line (which you can't see unless you have a very slow terminal).

UUencoding text

Sending text like a vimrc will sometimes lose special characters. Therefore uuencoding them is a good idea. Q: How do I encode text with uuencode? A: !uuencode filename

---

Reading man pages

Manual pages use the combination "_^H" to underline text. When you read in a man page into a buffer then you will want to get rid of these characters. Using sed is an easy way to filter these away: :r!man man|sed -s "s/_^H//g"

EXINIT prob

Adding EXINIT="set ex" to the environment does not always enable loading of the ~/.exrc file. It DOES work on a NeXT running MachOS (a BSD bastardization), but under IRIX it actually *causes* vi to look in the current directory for its init file. The winning answer came from Senthil Kumar (after I figured out the precise syntax): setenv EXINIT "source ~/.exrc"

Specification of current filename without extension

For some commands it is useful to specify the current filename. You can specify the current filename with "%" which is useful for many shell commands.

```
Q: Is there a way to specify the current filename without the extension?
A: Yes, use "%<".
```

Turning off message "Thanks for flying vim"

When using vim in an xterm it renames the title of that window to "Thanks for flying vim" on exit. Q: How to turn off the message "Thanks for flying vim"? A: :set notitle

---

Turning off shell access

Setting the shell to "/bin/false" is not enough! You must turn off the following things, too:

- File name expansions in all ex commands.
- The ex :! and the vi ! commands.
- The ex ":read !" and ":write !" commands.
- The ex :shell, :stop and :suspend commands.
- The vi ^Z command.

Also, take a look at Secure Editors by Freedman, Sharp & Associates, where you'll find editors binaries for AIX, ULTRIX, HP-UX, IRIX, M88k, OSF1, SunOS, Solaris. There are three editor binaries and one pager binary for each platform that have been modified so that they will not spawn processes or allow the user to work with files other than the initially loaded file. You cannot get out of them via ex or ^Z escapes. Also, the initially loaded file must be specified with a complete pathname on the command line. Secure, non-escapable binaries are provided for:

- elvis - a vi workalike.
- mg - a GnuEmacs-like editors.
- umacs - same as mg, but with key bindings more like Gosling-style Emacs.
- less - a GNU pager, an improved `more` command.

Executing a line as a command from the shell

Ever wanted to tell people which command to execute? Testing it yourself requires you to type in the command again. But with vim's feature of stuffing the contents of a buffer onto the command line this is easier than ever! " Yank the current line into buffer y and execute it map - "yyy:@y^M " Use buffer y buffer y yy yank current line : switch to command line @y^M put contents of buffer y onto command line

---

Mappings with if-then-else

Situation: I want to re-map '0', so that '0' does the same thing as default when the cursor is *not* in the first column. When the cursor *is* in the first column, I what '0' to do what '$' does : moves the cursor to the last column. Q: VIM commands do not have an if-then-else syntax. However, there is a solution for your problem if you use plain vi: map 0 my^V|mzl$``z``y`` Based on the fact that vi doesn't consider a jump to the same character position a jump and thus doesn't reset the previous (``) mark in this case. Because it didn't work on lines of length 0 or 1, I added the l (ell) to make it error out in this case, which should be satisfactory unless you're trying to use this as a subroutine from another macro. Might be able to improve on this if you don't like this. This unfortunately doesn't work under VIM 3.0, even in the compatibility mode. VIM marks are incompatible with vi marks in that Vim apparently doesn't consider a jump to another character position on the same line as a real jump, so doesn't save the previous location in ``. Hope you needed this for real vi! GregUbben@aol.com

---

Assigning TAB as the expand key

Situation: When I hit (tab) to complete a filename all I get is a ctrl-I. My first attempt to fix the problem was to redefine the wildchar variable to a different key (wildchar is the character used to invoke file completion, default is (tab)). I've tried to change the 'wildchar' variable in the .vimrc file, but my attempts have failed: No matter what key I assign to the variable wildchar=0 is what I get after startup. I know that vim is seeing the .vimrc file because I can change other variables. I've also tried to change wildchar using the set command from the : prompt with no luck. A: You must specify the wildchar character with a three character digit decimal equivalent. Using "wildchar=009" will assign the TAB character.

Modeless vim

Situation: People seem to prefer "modeless editors". Q: How to make vim modeless. A: There are a few things that vim has which could help make it seem like a modeless editor (if you really want it to :-) - From insert mode, ^O may be hit and then followed by any command-mode command, eg ^Odd as mentioned above. After the command is finished you will be back in insert mode again. - There is a setting 'insertmode', which can be set in your .vimrc or .exrc. If set, then editing files always starts in insert mode. - The cursor keys, page-up, page-down keys work in insert mode as expected.

Changing colours

Situation: The blue background looks great on color monitor, but on my notebook I like to have either black or white background. Q: Is there an easy way to change the VIM background color? A: From: Wolfgang Schumacher schumach@cs.tu-berlin.de Here's the _vimrc for my notebook. I hope it helps (proceeding on the assumption that you have an IBM compatible) - or read vim's msdos.doc. The ^[sequence is the escape character (entered by typing <ctrl>V <esc>): set autoindent nojoinspaces backspace=2 shiftwidth=2 set ignorecase ruler report=0 visualbell set dir=>c:\\tmp nobackup showmatch map g G set t_ti=^[\|79m "invert mode: bwht(15) on red(64) e.g. v set t_so=^[\|14m "standout mode: yel(14) on blk(0) e.g. :set set t_tb=^[\|14m "bold mode: yel(14) on blk(0) e.g. :!ls ^D set t_tp=^[\|07m "normal text: wht(07) on blk(0) set t_se=^[\|07m "normal text: wht(07) on blk(0) " COLOURS background foreground foreground " ========== " 128 blinking + 0 black + 0 black 8 gray " 16 blue 1 blue 9 light blue " 32 green 2 green 10 light green " 48 cyan 3 cyan 11 light cyan " 64 red 4 red 12 light red " 80 magenta 5 magenta 13 light magenta " 96 brown 6 brown 14 yellow " 112 white 7 white 15 bright white

---

Disabling suspend

Situation: I use vim as the editor for composing emails with ELM. ELM somehow loses vim when you suspend it. So I need to disable the suspend command of vim. Q: How to disable the "suspend" with ^z ? A: Remap the ^z to start a shell: map ^Z :shell^M Situation: I do not want the mapping in my vimrc as I only need it with vim when I call it from ELM. Q: So how tell I vim on the command line "use vim with the vimrc, but also disable the suspend"? A: Yes, just put the map command into quotes: vim +"map ^Z :shell^M" file How you get the ^Z and ^M through the shell is up to you :-)

Definition of abbreviation IDs

Fact: abbreviations can be of any two types: full-id that consists entirely of id characters (= [0-9a-zA-Z_]), and non-id consisting entirely of non-id characters but the last one. Goal: I would like a set of abbreviations beginning by '\' (for TeX use, and one letter is not enough). Or more generally to be able to include anything (but spaces) in an abbreviation. Answer: VIM 4.0 will let you define ids. Stay tuned!

Multiple Buffers - show only one at a time

Q: Is there a way to have multiple files open in vim, but only display one at a time? I'd like to switch back and forth between files without having to quit (save) them. I could put the files in windows, but I want to display only one at a time. A: The main thing you probably want is "set hidden" in your .vimrc, then when you change to a new file, the old one becomes hidden, not saved. ":wa" will write all the buffers.

DOS setup configuration

From: schumach@cs.tu-berlin.de (Wolfgang Schumacher) Newsgroups: comp.editors Subject: Re: how to config vim on msdos? Date: 13 Nov 1995 12:34:33 GMT References: <47ks4g$fq@nuscc.nus.sg> >i on ms-dos. i installed a vim 3.0 for msdos. everything is nice except that >i cannot config my vim environment, say _vimrc and _exrc. >i tried many times, both in current directory and root directory, >vim still started up with its own default setting. There are several methods - here are two of them: 1st method: set the environment variable VIM (in autoexec) set VIM=path vim will execute the file _vimrc located in path (path is also the default helpfile location) 2nd method: set the environment variable VIMINIT (in autoexec) set VIMINIT=vi-command(s) when vim starts, it will first execute the vi-command(s) specified by VIMINIT (for instance set VIMINIT=:so c:\bin\vim.ini) Wolfgang

---

Weird keyboard layouts

The key bindings on a Mac keyboard with German layout are: / shift-7 [alt-5 ? shift-sz \ alt-shift-7] alt-6 @ alt-shift-1 { alt-7 ~ alt-control-n } alt-8 Just changing a pair of square brackets with ":s/\[\]/\{\}/" requires you to type the sequence : s shift-7 alt-shift-7 alt-5 alt-shift-7 alt-6 shift-7 alt-shift-7 alt-7 alt-shift-7 alt-9 shift-7 whereas the same thing on an american keyboard is : s / \ [\ ] / \ shift-[shift-] / Quite a difference, huh?

So you people using keyboards with English/American layout should be grateful for ASCII. Now, if we had won the war oops ;-)

Executing commands from a file

Q: How can I put the editing commands in a file, and have vi read that file to execute it?
A: If you mean you want to manually execute a certain set of commands like a macro from within vi, put ex commands in a file, then use :source filename to execute them. If you want it to run unattended you can redirect input from a file but it is really better to use sed, awk, or perl for batch operations.

Vim and Spelling Checkers

Q: How do you use a spell checker with vim?
A1: The basic Vi method with a small enhancement by Vim: Use a spell checker on the current file *externally*, that is call "check %" from the command line: :!check % .. where "check" is usually "ispell" or (better) "aspell". As there may be modifications to the current buffer you should ":write" them back to the associated file before that. And after the checker has changed the file you should continue with the changed file, so ":edit!" your file again. Summary: :write :!check % :edit! You can map the check command, of course: map ,c :!check %^M With Vim, all this gets a little easier. First, you can ":set autowrite" which will make Vim ":write" the current file before calling an external program so that current modifications will be written to the associated file before an external command gets to see it. And you can use Vim's angle notation to write the CTRL-M at the end of the mapping in ASCII (&;ltCR>). So this goes into your setup file: set autowrite map ,c :!check %<CR>|:e!<CR>

A2: You can call a spell checker like "ispell" from vim without a problem. A function to look up a word is included with the command "K". From the "manual" (reference.doc): K = Run a program to lookup the identifier under the cursor. The name of the program is given with the 'keywordprg' (kp) option (default is "ref"). The identifier is formed of letters, numbers and the underscore. The identifier under or right of the cursor is used. The same can be done with the command ":!{program} {identifier}". There is an example of a program to use in the tools directory of Vim. It is called 'ref' and does a simple spelling check. {not in Vi} So what you do is get a spell checker, eg "ispell", then you

issue the command ":set keywordprg=ispell", and then hit "K" on the word you want to look up, ie "check".

The underscore command '_'

Q: I don't see a reason for the command "_" - why is it in vim at all?
A: It's there for the sake of vi-compatibility. It is used internally for vi commands, so when you type "dd" or "cc" etc, it is internally converted to "d_" or "c_". Hint: If you don't need then you can use it as the first character for mappings.

Pasting yank buffer into command line Fact: You can record keystrokes to a yank buffer and thus you can put it into the edit buffer. Question: How do you paste a yank buffer into the command line? Answer: Assume that there is something in buffer 'y'. Enter ":@y" then hit return.

Vim's smartindent option is nice. But there's one thing, I dislike a lot: Whenever you start a line with "#", Vim ignores the current indent and places the # at the begin of line. This is o.k. when editing C-code, but if you're writing shell or awk scripts, # introduces a comment... If I want to write something like if [$a = "x"]; then # comment about what's done here I have to indent the # line manually. :-(I don't use vim, but I see where that could be a problem. Even in C/C++-code I don't start lines with # at the first column. The product I work on (Preditor/2) uses one codebase to support multiple platforms (OS/2, Windows and Win95/NT). It uses a multi-platform class library and *lots* of #ifs to do this, many of which are nested. If all of th #ifs start in the 1st column, you easily get hopelessly lost and confused. Therefore I always indent the #ifs just like the rest of th code so it can be read. Yes maybe there should be different types of smartindenting? For the moment you can get around this by putting this in your .vimrc or .exrc file: inoremap # X^H# where ^H is typed as <Ctrl-V><Back-Space>. Strange, most C compilers won't allow the # to be anywhere other than in the first column. I think the ANSI standard requires it to be in the first column. You can of course have spaces/tabs after the '#' and before the "if" or whatever. -Rob.

In Vim 3.0, * (in the command mode) searches forward for the next occurrence of the *word* that is under the cursor. And # searches backward. If you want to search the word which can be part of another word, you can try the following map map \s wb"zyeo/^["zpbb"zy$u@z^M where ^M is Ctrl-V Ctrl-M do one of the following you should be able to relocate the tmp directory: - set directory=/<pathname> in EXINIT - or, "set directory=/<pathname>" in your .exrc file - or, in a vi session do ":set directory=/<pathname>" where pathname is the location you want to put the temp file.

I have problems using vi with characters which have an ascii value > 128. For example if I insert ue the editor echoes \334 in insert mode. After leaving the insert mode everything is fine. Also fmt removes all characters with ascii > 128 from the text being formated. You have to define the environment-variable LC_CTYPE. If you are using csh then put in your .cshrc following line: setenv LC_CTYPE=iso_8859_1

Encryption

Question: I typed ":Call" on the command line and was prompted with "Enter a key"; so I typed something like "123" and then the text seems to have turned into binary characters. What happened?
Answer: "Yep, you've encrypted it! I never knew there was a command ":Call" in vi, but it turns out there is, or is it just ":C"?, and it seems to encrypt your file. To recover it, type "crypt < file", and then enter the password when prompted, assuming you're right about the "123" bit!"
Could not reproduce this. Maybe because we do not have "crypt()" here. -Sven

---

Startup message

Question: When I open a file with vim via `vim `filename' vim sometimes displays a message in its status line: "~/myfile" [textmode] 902 lines, 92498 characters Press RETURN or enter command to continue The window now displays nothing else but this message and you have to press RETURN to start the edit session. This is annoying. But vim does this not every time. Sometimes the initial message is shown, sometimes not. It seems to depend on the file beeing edited somehow, but I don't now in which way.

Answer: This will happen if the message is long enough to collide with the showcmd area, since this would overwrite the message and vim wants to make sure you got a chance to read it. If the file name is short enough then it knows it won't get overwritten so it won't ask for RETURN. Same thing happens while in vim if you type ":e long-file-name..."

Yanking current word
(y) - yank
Q: How do you put the "current word" into the yank buffer?
A: Use "wbye".
"ye" will yank to the end of the current word. But we need to put the cursor onto the beginning first. If the cursor is on the beginning of the word then "b" will go to the beginning of the previous word. So we use "w" to go to the next word first. Result: "wbye"
[950608, 950828]

Q: In my .vimrc file I have "set backupdir=>$HOME/.vim-bakup" and it whines and complains. The '>' tells vim to only put backups in that directory without checking anywhere else. Problem is it doesn't work. However, "set backupdir=>/home/philip/.vim-bakup" (which is the same thing, really) works just fine. The vim manual says that $HOME (or ~) will work okay, but apparently I've hit a glitch. Any one know what's up?

A: Environment variables are only expanded at the start of settings, and unfortunately the ">" at the start means that the "$HOME" isn't quite at the start. This was a teeny-weeny buglet. It has been fixed for the next version. Increasing a number with leading zeroes removes all leading zeroes except one. This is a bug" with vim3.0. Vim4.0 should fix this.

Back to the -> VIM Home Page

Terminology

Well, before every good text there need be some definitions. They make it easier for you to understand. Please take the time to read this - I am sure you will benefit from it!

line
: A sequence of characters. Start: First character after an EOL character (or first character of buffer). End: Next EOL character. The EOL is included.

running text
: A sequence of consecutive characters.

block
: A sequence of (full) lines.

paragraph
: see block

Hmm, this needs a lot more work...

Why use "ed"?

Some uses of "ed":

ed and diff
: "diff" produces output which "ed" can use to update programs to various version levels (the original poor man's "sccs).

  "diff3" produces output which "ed" can use to update prog2 to make the same changes as were made to change prog1a into prog1b. This comes in handy when you've several rather similar files in which you need to make

corresponding edits.

ed and sed

Unlike "sed", "ed" can go both forwards and backwards in a file, allowing some scripting which can't be done with "sed".

grep

You can demonstrate where "grep" really came from :)

```
:g/RE/p
```

```
URL:          http://www.math.fu-berlin.de/~guckes/vim/answer.html
URL:          http://www.vim.org/answ.html (mirror)
Created:      Mon Jan  1 00:00:00 MET 1996
```

Send feedback on this page to
Sven Guckes guckes@vim.org

Vim Chat/Talk

Let's Talk about VIM! :-)

IRC #vim

Some of us are using the chat system "Internet Relay Chat" (IRC) to talk about Vim. There are different "nets", though. We usually meet on *IRCnet* on channel `#vim`.

I connect via the server `irc.fu-berlin.de` - but you cannot join this server from everywhere, so you should try to connect to one of the servers listed below.

Please note that you will find yourself in *one* of the various IRC networks. So even though you are on channel `#vim` on your net you might not meet others on `#vim` as they could be using this channel on another network.

If you do not have an IRC client yet then I suggest you get *mIRC* (for Windows) or *epic* for Unixish systems.

I definitely prefer using a proper IRC client - but please give the following web based chat a try if you have no alternative.

Note: The service is given by *chatset.web.de* - a site in Germany. Therefore the descriptions of this service are in German. However, if you have used IRC before then you will probably know all the basic commands, anyway.

I am looking forward to your feedback - thanks! :-)

Server: **irc.fu-berlin.de** Channel: **#vim**

Chose a nickname and then click on the button: vimuser ChatSet starten

---

Status: Experimental!

Right now [000228] the server seems to be flakey - so let's see if this works out. If no then I'll try some other talk service...

NOTE: The service is run in Germany - so all the help is in German. But I am sure that Vim users are smart enough to understand it right away. :-)

Chose a Nickname:

Your Password:
(only required for registered nicknames/users)

Chatroom: vim

powered by Cool-Chat.de

- Enter Chatroom -

IRC Servers

You cannot join irc.fu-berlin.de from anywhere - so here's a list of servers you can try:

```
AS, JP, Tokyo:                  irc.tokyo.wide.ad.jp:6667
AS, MY, Silicon:                irc.silicon.net.my:6667,6668
AU, CA, Flute:                  flute.telstra.net.au:6665,6666,6668,6669,6680
AU, Melbourne:                  yoyo.cc.monash.edu.au:6668,6680,9990
AU, Sydney:                     irc.usyd.edu.au:6667
CA, ON, Toronto:                ircnet.idirect.ca:6667
EU, AT, Chat:                   chat.on.irc.at:6666,6668,6669
EU, AT, Linz:                   linz.irc.at:6666,6667,6668
EU, BE, Brussels:               irc.vub.ac.be:6667
EU, CH, Geneva:                 irc.span.ch:6667
EU, CZ, Prague:                 irc.felk.cvut.cz:6667
EU, DE, Berlin:                 irc.fu-berlin.de:6665,6666,6668,6669
EU, DE, Stuttgart:              irc.uni-stuttgart.de:6665,6666,6668,6669
EU, DK, Copenhagen:             ircnet.dk:6667
EU, EE, Tallinn:                irc.estpak.ee:6667
EU, ES, Valencia:               luisvive.euiti.upv.es:6667
EU, FI, Espoo:                  irc.funet.fi:6666,6668
EU, FI, Helsinki:               irc.cs.hut.fi:6667
EU, FR, Nice:                   irc.eurecom.fr:6667
EU, FR, Paris:                  irc.enst.fr:6667
EU, FR, Poly:                   sil.polytechnique.fr:6667
EU, GR, Thessaloniki:           irc.ee.auth.gr:6666,6668,6669
EU, HU, Budapest:               irc.bme.hu:6667
EU, IL, Tel-Aviv:               irc.tau.ac.il:6667
EU, IS, Reykjavik:              irc.isnet.is:6667
EU, IT, Pisa:                   irc.ccii.unipi.it:6667
EU, IT, Rome (Flashnet):        irc.flashnet.it:6664,6665,6666,6669,7000
EU, IT, Rome (tin):             irc.tin.it:6665,6666,6668,6669
EU, LV, Riga:                   irc.lvnet.lv:6667
EU, NL, Amsterdam (nlnet):      irc.nl.uu.net:6660,6661,6662,6663,6664,6665
EU, NL, Amsterdam (xs4all):     irc.xs4all.nl:6660,6661,6662,6663,6664,6665,6666
EU, NL, Enschede:               irc.snt.utwente.nl:6660,6661,6662,6663,6664,6665
EU, NL, Nijmegen:               irc.sci.kun.nl:6660,6661,6662,6663,6664,6665
EU, NO, Oslo:                   irc.ifi.uio.no:6667
EU, NO, Trondheim:              irc.pvv.unit.no:6667
EU, RU, Moscow:                 irc.msu.ru:6667
EU, SE, Lulea:                  irc.ludd.luth.se:6668,6669
EU, UK, London (Btnet):         chat.bt.net:6667
EU, UK, London (Demon):         ircnet.demon.co.uk:6665,6666,6668,6669
EU, UK, London (Netcom):        irc.netcom.net.uk:6667
EU, UK, Warrington (u-net):     irc.u-net.com:6661,6663,6665,7000,7001
```

```
US, CA, Santa Clara:            irc.ncal.verio.net:6667
US, MI, Taylor:                 irc.webbernet.net:6660,6666,6668,6669,6677
US, NY, New York:               irc.stealth.net:6667
US, WA, Seattle:                ircnet.sprynet.com:6667
```

List provided by Bruce DeVisser [000427].

Links

```
URL:         http://www.math.fu-berlin.de/~guckes/vim/chat.html
URL:         http://www.vim.org/chat.html (mirror)
Created:     Mon Mar 27 12:00:00 MET DST 2000
```

Send feedback on this page to
Sven Guckes guckes@vim.org

VIM - Newsgroup and Usenet

VIM does not have a newsgroup of it's own. But the appropriate newsgroup to post to is comp.editors.

But before asking please contact a local guru and ask him whether he can solve the problem. Also, take a look at the huge helptexts using ":help". Search through them with keywords that might have to do with the problem.

If you think that noone can help you locally then you might consider posting it on Usenet. But before posting please put together some info which is usually requested.

HowTo post on Usenet about Vim

General Stuff

Some people think they can avoid or at least lessen the spam they receive by posting with in invalid address, so address harvesters cannot use it. Well, it is partially true. But it is a pain to those who want to reply to you by email. Therefore, please consider this setup:

```
From:     FirstName LastName <pseudoaddress.invalid>
Reply-To: FirstName LastName <real.address>
```

First, adding ".invalid" (mind the dot) to your pseudo address tells newsreaders to give its user a warning when replying by email. This is nice because you tell humans that this is a fake address without having them figure it own by their own. Making them scan all of your article before they reply simply costs too much time. So - be nice!

Second, adding a Reply-To line with a *real* address (one that accepts mails without complaining) is nice because it allows readers to reply to you by email. Now, you may think that address harvesters also get addresses from the Reply-To line. Good thinking - but this hardly happens. The Reply-To usually is *not* in the overview data you can get from a newsserver, you see. So you'd actually have to download all articles and *then* scan the headers for Reply-To lines. Apparently only a few harvesters do that. So **a Reply-To line is *almost* safe** - and nice to real readers.

Third, please **post with your real name**. While there are reasons for using a fake name or a pseudonym on the net, of course, by is there a reason to use one on comp.editors? I don't think so. And if you newsreader does not allow to adjust your posting setup based on the current newsgroup - then get a better newsreader! ;-)

Check the old articles!

Avoid posting FAQs! Do a little research first. Try to find articles about the problem before you post.

If you do not have a good newsreader then you can use the URLs below to query the news services Altavista and Dejanews. I cannot promise that the given URLs will find your stuff, but they can give you an example on how to find articles with these. Furthermore, you can get an instant reply for your search - and that's usually better than having to write a post and wait for answers.

Version Info

Please give info on the vim version! Since vim-5.2b you can request the info from the command ":version" on the command line with the option "--version" , too. You can redirect this output easily into a file:

```
vim --version > vimvers.txt
```

Here is an example:

```
$ vim --version > vimvers.txt
```

```
$ more vimvers.txt
VIM - Vi IMproved 5.2d ALPHA (1998 Jun 1, compiled Jun 11 1998 17:45:33)
Compiled with (+) or without (-):
+autocmd -browse +builtin_terms +cindent -cscope +dialogue_con +digraphs
-emacs_tags +eval +ex_extra +extra_search -farsi +file_in_path +find_in_path
+fork() +GUI_Athena +insert_expand -langmap +lispindent +modify_fname +mouse
-mouse_dec -mouse_netterm +mouse_xterm -multi_byte +perl +quickfix -python
-rightleft +showcmd +smartindent -sniff +syntax +tag_binary +tag_old_static
-tag_any_white -tcl +terminfo +textobjects +viminfo +writebackup -xterm_save
+X11
   system vimrc file: "$VIM/vimrc"
     user vimrc file: "$HOME/.vimrc"
      user exrc file: "$HOME/.exrc"
  system gvimrc file: "$VIM/gvimrc"
    user gvimrc file: "$HOME/.gvimrc"
    system menu file: "$VIM/menu.vim"
    default for $VIM: "/Vol/Pub/share/vim"
Compilation: gcc -c -I. -Iproto -DHAVE_CONFIG_H  -DUSE_GUI_ATHENA  -I. -O3 -pipe
-I/Vol/pub/include -I/Vol/X11R6/include -Wall -Wshadow -I/Vol/X11R6/include
-I/Vol/pub/lib/perl5/sun4-sunos/5.00401/CORE
Linking: gcc -s -L/Vol/pub/lib -o vim -L. -L/Vol/X11R6/lib  -lXaw -lXmu -lXext -lXt
-lX11  -lncurses
/Vol/pub/lib/perl5/sun4-sunos/5.00401/auto/DynaLoader/DynaLoader.a
-L/Vol/pub/lib/perl5/sun4-sunos/5.00401/CORE -lperl -lgdbm -ldbm -ldb -ldl -lm -lc
-lposix
```

Also, which operating system are you using vim on? Let us know! Use one of these commands to find out about your current system:

```
UNIX:   uname -a
DOS:    version
```

Values of Options

For all things that involve options please give their values! Remember that you can look at values with `:set option?` , eg ":set textwidth?".

Examples

Give examples! Please make sure that the example is reproducible.

Put "vim" into the Subject line

When asking questions about vim then please say so in the Subject: line by making "vim" the first word. Add other keywords, too, for example variable or command names.

```
Subject: vim - Qp and textwidth

Subject: :g vs :v [vim]
```

When following up on a post about Vim *wihout* "vim" in the Subject line then please *add* "[vim]" to it. Then at least the followup can be scored accordingly. The brackets around "vim" also tell readers that this is the name of the program more visually.

Reading about Vim on Usenet

If you do not have a newsreader to read Usenet News (what a pity) then you can read News via DejaNews or Altavista.

To make this easier for you here are some links which already extract info about vim:

DejaNews - (g)vim

> NewsService: DejaNews
> Newsgroups: comp.editors
> Search for: "vim" OR "gvim"
> Sorting: by Date
> Format: terse
> Maximum Hits: 100

Altavista - (g)vim

> NewsService: Altavista
> Newsgroups: comp.editors
> Search for: "vim" OR "gvim"
> Note: Altavista starts with oldest article first. :-(

Thanks for the URLs, Richard! :-)

Back to the -> VIM Home Page

```
URL:          http://www.math.fu-berlin.de/~guckes/vim/newsgroup.html
URL:          http://www.vim.org/newsgroup.html (mirror)
Created:      Fri Dec  1 00:00:00 MET 1995
```

Send feedback on this page to
Sven Guckes guckes@vim.org

VIM - CVS repository on cvs.vim.org

Vim's source code is now managed with CVS.

Repository:

- http://vim.sourceforge.net/ which redirects you to
- http://vim.sourceforge.net/cvsdocs/

Maintainer:
The CVS repository is maintained by *Johannes Zellner johannes@zellner.org*

HOWTO use Vim's CVS repository:

- http://vim.sourceforge.net/cvsdocs/anonymous-cvs.html

HOWTO use the repository with CVS:

```
cvs -d :pserver:anonymous@cvs.vim.org:/cvsroot/vim login
cvs -d :pserver:anonymous@cvs.vim.org:/cvsroot/vim co vim
```

Links

CVShome [000605]

http://www.cvshome.org/

The makers of "CVS".

CVS tools for Windows and MacOS [001002]

http://www.wincvs.org/

```
URL:          http://www.math.fu-berlin.de/~guckes/vim/cvs.html
URL:          http://www.vim.org/cvs.html (mirror)
Created:      Thu Dec 16 18:00:00 MET 1999
```

Send feedback on this page to
Sven Guckes guckes@vim.org

Vim - HowTo Help!

Would you like to help your fellow Vim users? Well, there are several things which need you help:

Every user can help us all by testing Vim.

You can maintain a syntax file, maintain a vim binary for your favourite operating system, or you can mirror Vim (ie the webpages or the complete standard distribution),

I am also looking for help for adding a **search engine** for the Vim Pages, so if you happen to have some knowledge on this then please contact me!

If you are able to code then you could help us implement Perl regular expressions or make a language port to support your (native) language, or simply submit code for any items on the todo list, eg integrating 'tal' into Vim or maybe even implement a :tetris command - just to show that **it can be done** with very little code. And if you want to go realy crazy then you even change the key bindings of Word (eek) so that Word gets vi-like input and even commands.

But coding itself does not help much if the documentation does not show how to use it. Therefore we also need help with the documentation. Several people have also suggested to translate the VIM FAQ.

You do not have to be computer geek to help us, though. If you'd like to design a tshirt for Vim - be our guest! :-)

And, of course, you'd make the author and a lot of children happy by donating something to the ICCF as to help children in Uganda suffering from the effects of AIDS. Thanks!

Vim Syntax Files - Get them Online [990101]

> Whenever there is a problem with some syntax file you should take a look at its latest version. Many of the syntax files are available online and are therefore linked for your convenience via the language page, especially the section about wanted syntax files However, not all syntax files are available on WWW yet. Please help the syntax file maintainers by making them online! Maybe you can offer them some webspace?

Vim Language Ports

> There exist ports with support for several languages. Vim-5 already has the install options "+rightleft" for Arabic and Hebrew, and "+farsi" for Farsi (Persian). There are also ports for Hangul (Korean) and Japanese - but these are based on old versions of Vim. We need some help with porting the latest version, of course. If you can help - please contact me!
>
> Also, we could use with some feedback from people using these language ports. Please let me know about problems - I will write them up into a webpage, so the developers can think about solutions.

Vim mirrors [980105,980114,980424]

> Do you have a machine connected to the Internet? Would you like to set up a mirror of the Vim Pages or mirror the Vim Distribution? Then have a look at the page on Vim Mirrors.
>
> We are especially looking for mirrors in these places:
>
> - Mexico
> - South America
>
> We would be very happy if you could provide download via HTTP - as some people cannot use FTP to download files.
>
> So, if you can set up a mirror - please contact me!

Vim Binaries [971103,971204]

> Would you maintain a binary for Vim? Then please let us know! And please read the page with the guidelines for preparing a vim binary archive. Thanks!
>
> DYNIX or dgux anyone?

Keith Starsmeare (kxs@bigfoot.com) is looking for vim binaries for the following systems:

```
DYNIX/ptx V4.4.1 i386
DYNIX/ptx V4.1.5 i386
dgux R4.10     generic AViiON Pentium
dgux R4.11MU03 generic AViiON PentiumPro
```

Please contact him!
Today he told me that he has succeeded in getting vim to work on these systems. :-) [971204]

Help Vim - implementing Perl regular expressions [971126]

Vim-5.0r now has the Perl expression "\s" for "whitespace". But there are a lot of Perl regular expressions missing. So we are looking for someone to implement this. Any takers?

Help Vim - coding and implementing the todo list

The "todo list" has become 100K in size - so we are looking for people who would specialize in getting some of the desired features to work and to stamp out the bugs still crawling around. If you think you can help then please do get in touch with the developers via the vim-dev mailing list.

Help Vim - by testing its commands, utilities and syntax files

All users are welcome to test the versions of Vim-5.

Some ports need special testing - please help!

```
System          GUI?    Contact
AmigaDOS-2.0    GUI!    Michael Nielsen mni@dde.dk
VMS             dunno   work discontinued.  Any takers?
```

Please take a look at the "ctags" utility.

Testing of new syntax files:

```
COBOL   Chamarty, Sitaram
```

Stay tuned to the maillist "vim-announce" and/or comp.editors!

Wanted: Syntax Files!

```
BASIC                           Ron Aaron ron@mossbayeng.com
SPICE (Hspice, Pscpice, etc)    Joel Auernheimer jaa@asu.edu
VISUAL BASIC                    Paul Moore gustav@morpheus.demon.co.uk
```

Help Vim - VMS port

On 980518, Bram said: "The patches are included in version 5.2c. If you have a compiler, you can try compiling it. But it's still work in progress, I may have made a few mistakes when including the patches. If someone lets me know where I can find an archive with a VMS binary, I'll upload it to the Vim ftp site."

Vim users who are interested in the VMS version of vim are:

```
Bram Moolenaar            bram@vim.org
Sven Guckes              guckes@vim.org
Sandor Kopanyi           sandor.kopanyi@essnet.se
Hans-Christian Eckert    ripley@nostromo.in-berlin.de
Kevin P. Inscoe          kinscoe@cbis.com
Bruce Hunsaker           BNHunsaker@chq.byu.edu
```

Anyone else? (Maybe I should setup yet another mailing list for this?)

Henk Elbers henk@xs4all.nl had ported vim to VMS - but due to a change in jobs cannot continue his work. On 980326 sandor.kopanyi@essnet.se contacted me saying that he got vim-5 to run reasonably on VMS, but the arrow keys do not work. Can anyone help him?

Henk once said: "The latest version I've been working on is vim-4.5. The distribution also contains patches to work on DEC Alpha with VMS. The VMS-vim is working reasonable. Editing is perfect, but there are some bugs, e.g. :recover doesn't work at all yet. There is no GUI version either. At the moment I don't have time to look at it, but in

the future (may/june 1997) I hope to spend some more time on it. I'll have a look at 5.0 too then. I just downloaded it and it is working on my Linux PC."

Adding "tal" to text formatting [971204]

The "tal" utility (see above) is so small and yet so nice that I thought that maybe someone would take the challenge and work in the code to Vim's internal "text formatting". Any takers?

Help Vim - :tetris

I have had this very small code for a tetris game (1,500 bytes) for some years now, and I keep thinking that it would be nice to have it inside Vim. Of course, this would be a compile option. ;-) Any takers? Let me know: guckes@vim.org

Word like Vi

Create a set of key bindings for Word such that it behaves like Vi. [991104]

Help Vim - Document it! [991208]

Additions and corrections to the helptexts are always welcome. Just send the "diffs" to the author - he is taking care of them personally. But you can always us by contributing to the upcoming Vim Book which shall also be available online.

Help Vim - Write HOWTOs! [000315]

Everyone uses versatile programs in a special way. Help your fellow users by describing your use of Vim.

I have received some requests for a HOWTO on "VisVim", ie using VisualStudio with Vim.

Help Vim - Translate the VIM FAQ [991208]

There have been several requests for translations of the VIM FAQ. If you want to help here then please let me know!

NOTE: I suggest that you wait until I have converted the VIM FAQ to DocBook format. I hope it will be easier to translate the FAQ using this format. Maybe I will get around to do this around Xmas 1999.

```
Dutch   000911  Floor Plikaar           floor.c@vvtp.tn.tudelft.nl
German  991208  Michael Weber          sirewok@01019freenet.de
```

Help Vim - Design a Vim tshirt [971017]

The Vim developers have thought about making a PostScript file available to print onto a tshirt. It should contain a list of the most often used commands and their syntax. To printed upside down ("umop-ap!sdn") so you can read it while wearing it at your keyboard. :-) This overview should also show a not saying "not registered user" by default - those who made a donation to the KCC will get a tshirt without this note, of course. ;-)

Any takers?

Help Vim users - Watcom compiler makefile anyone? [970912]

Anyone have a makefile for the Watcom C++ 10.5 compiler? Please send it in!

Especially interested: Lues Cerebri soberg@internet.de

Donations -> ICCF

The author is quite active in helping orphans in Uganda. Instead of making Vim a commercial product he is asking you as a user to donate whatever you consider Vim to be worth for your use to those children. For more info please see the file uganda.txt and at the page on the ICCF.

Thanks for all the Help!

Here are some of the items that people had helped with. Thanks to all of you! :-)

[solved] DOS/Windows - using Vim as replacement for notepad

Does anyone know how to replace notepad with Vim? See the answer on the Vim Answers Page. [971231]

[solved!] DOS - "date" without prompt? [971022,971101]

Does anyone have a "date" program for DOS that does not prompt you to enter a new date? Possibly something

GNUish with a configurable output format?

Problem:

The prompt gets in your way when you want to use is within a mapping...

Solution:

Vim-5 has the function "strftime" built-in to do this. See the mappings in Sven's vimrc - http://www.vim.org/rc! No more external date command! Whee!

```
URL:          http://www.math.fu-berlin.de/~guckes/vim/help.html
URL:          http://www.vim.org/help.html (mirror)
Created:      Thu Jun 11 11:11:11 CEST 1998
```

Send feedback on this page to
Sven Guckes guckes@vim.org

VIM Users and their Vim Pages

100! entries [010111]

This page lists users of the editor VIM with their own page about their VIM setup and with tips and tricks that they use with their everyday editing; some of them also maintain a syntax file for some language and maybe even maintain a binary of Vim for some operating system.
Or as Nick Hibma put it: "I presume it is just a general list of people thinking that life is nice and vim in particular." :-)

So please join - even if you do not have a dedicated vim page. All info on this page might help - even if just to see what environment people are using vim.

Check it out! :-)

Looking forward to *your* vim page. --Sven

Sven's vimrc

I'd like to take the opportunity to point out that I have spent a lot of time commenting my vimrc to give examples and a quite a few tips. This vimrc might become an extensive web page some day, but I usually find it easier to add the info directly to it. Please take a look at it:

Sven's Setup Files for Vim:
http://www.math.fu-berlin.de/~guckes/setup/gvimrc
http://www.math.fu-berlin.de/~guckes/setup/vimrc
http://www.math.fu-berlin.de/~guckes/setup/vimrc.gz (zipped)
http://www.math.fu-berlin.de/~guckes/setup/vimrc.forall
http://www.math.fu-berlin.de/~guckes/setup/vimrc.forall.gz (zipped)

Feedback on my setup files is very very welcome! :-)

Asking for help:
Please remember to give info when asking for help. You should include info about these things:

- Machine type (processor)
- Operating System(s)
- Compiler(s)
- Compile Options (":version")

Submissions - HOWTO Submit Your Syntax File

[980422] If you want to **submit a syntax file** for Vim's distribution then all you need to do is to **send it to the author**, **Bram Moolenaar bram@vim.org**.

Please include the following info within the file: Name and/or description of language (may require version number), name and address of the maintainer, date and time of last update, filenames to apply to, and (very good) a web address for the very latest version.

Example:

```
" Vim syntax file
" Language:     Mail and News (for response in email and Usenet)
" Maintainer:   Sven Guckes guckes@vim.org
" Last change:  Fri Jul 31 02:00:00 CEST 1998
" Filenames:    ~/.reply ~/.followup mutt* snd.* postponed/*
" URL:          http://www.math.fu-berlin.de/~guckes/vim/syntax/sven.vim
```

VIM Users - Short List

Latest entry is marked with "[*]": Aaron | Acevedo | Akesson | Albayrak | Arpadffy | Arens | Bader | Baruchel | Bigham | Bloch | Böhme | Bordelon | Brady | Campbell | **Capricelli** | Castagnetto | Chamarty | Corks | Dara | Duperval | Eckert | Elbers | Ehrens | Exel | Faylor | Feige | Fisher | Fleiner | Foucher | Fraust | Garcia-Suarez | Glass | Gonsalves | Griffin | Guckes | Guldberg | Haney | Hannedouche | Harrsion | Hack | Hedlung | Hemert | Hermitte | Heimann | Hesse | Hibma | Hiebert | Hlavacek | Hunsaker | Jensen | Jones | Kallingal | Kang | Koehler | Kol | Kopányi | Lee | Leonard | Lottem | Matsumoto | McKee | Mann | Martinson | McMullen | Moolenaar | Mueller | Muth | Nagle | Nassen | Nenadov | Nielsen | Norback | OBrien | Pascoe | Piefel | Raisky | Rebbechi | Reilly | Riehm | Riiser | Riswick | Rios | Rosso [*] | Rochholz | Schemenauer | Scott | Seibert | Seidman | Shan | Sharpe | Shiran | Siegmann | Slootman | Socher | St-Amant | Starsmeare | Steden | Thomas | Uhl | Urban | Tennent | Verdoolaege | Zachmann | Ziegler
and silver's petz

VIM Users - Long List

Generic Entry - please fill out and send to me.
Lastname, Firstname - user@domain [DATE]
    http://www.domain/~user/vim/
    Syntax files:
    Language: http://www.domain/~user/vim/syntax/language.vim

Binaries: http://www.domain/~user/vim/bin/os/

Environment:

Note:

Aaron, Ron - ron@mossbayeng.com [990723]

http://www.mossbayeng.com/~ron/vim.html

Environment:

Note: Ron Aaron's page offers quite a few builds of GNU utilities and his own additions to them. WIth instructions on how to build them for Win32.

Acevedo, Raul Segura - raul@hplara.iquimica.unam.mx [980702]

Akesson, Linus - lft@df.lth.se [010220,010223]

http://www.df.lth.se/~lft/vim/

Environment: ??

Note: Linus has written two interesting macros: a Mandelbrot set generator - and a "Towers of Hanoi".

Albayrak, Ali - Ali.Albayrak@cs.Helsinki.FI [980114]

Environment:

processors: 483SX/33, RS6000/350, RS6000/C20, RS6000/250, RS6000/530, Sun4m/SPARC;

systems: AIX 4, AIX 3, SunOs 5.4, Linux 1.2;

compilers: AIX cc, gcc

Note: Ali Albayrak offered to help with installation of Vim on the operating systems he is using.

Arpadffy, Zoltan - arpadffy@altavista.net [000223]

http://www.polarfox.com/vim/

Binaries: Open VMS for Alpha and VAX

Environment: Open VMS (Alpha and VAX)

Home: RedHat Linux, FreeBSD and Windows 98 (for the kids)

Work: Open VMS, Windows NT and Solaris

Note: Zoltan Arpadffy's page is the "VIM on Open VMS Information Page" and is very nicely made.

Arens, Ralf - ralf.arens@gmx.net [010223]

http://home.tu-clausthal.de/~mwra/vim/

Environment: ???

Note: Lots of macro files, most notably his MailNews.vim.

Bader, Thomas - thomasb+vim@trash.net [000613]

http://www.trash.net/~thomasb/vim/

Environment: Debian GNU/Linux (Kernel 2.0.36 & 2.2.16, i486 & i586 & i686), SunOS 5.7 (sun4u), Windows 98 & NT

Note: Some setup files, usage of "2html.vim", screenshots.

Baruchel, Thomas - baruchel@libertysurf.fr [000805,000829]

http://perso.libertysurf.fr/baruchel/vim/

Scheme support file:

http://perso.libertysurf.fr/baruchel/vim/schemesupport.vim

Bloch, Eduard - edi@gmx.de [000416,000522]

http://sites.inka.de/sites/zombie/edecosi/vim.en.html

Environment: Debian Linux; Windows mit gvim.

Note: Interesting pages with sample configuration.

Bigham, Scott - dsb@cs.duke.edu [971118,980817]

http://www.cs.duke.edu/~dsb/vim/

Lynx ("*.lss"): http://www.cs.duke.edu/~dsb/vim/syntax/lss.vim

Perl POD format: http://www.cs.duke.edu/~dsb/vim/syntax/pod.vim

Environment: Work: assorted Solaris boxes running Vim 5.0o Home: Atari TT030 running Vim-4.6 and a Linux-2.0.27 notebook running Vim-5.0o.

Böhme, Harald - boehme@informatik.hu-berlin.de [980113,980622,010320]

http://www.informatik.hu-berlin.de/~boehme/vim/

SDL: http://www.informatik.hu-berlin.de/~boehme/vim/syntax/sdl.vim

Binaries: SunOS4 and Solaris

Environment: Linux and Solaris

Bordelon, Scott L. - slb@artisan.com [971110]

No vim page - but a syntax file:

CADENCE "Dracula" (electronics design rule checking):

dracula.vim

Note: This syntax file is local as Scott does not have a webpage. It is updated whenever there is a need for it.

Brady, Rob - robb@datatone.com [990629,990702,990813,990819]

http://www.datatone.com/~robb/vim/

Microsoft Module-Definitions: http://www.datatone.com/~robb/vim/syntax/def.vim

Focus Exec: http://www.datatone.com/~robb/vim/syntax/focexec.vim

Focus Master: http://www.datatone.com/~robb/vim/syntax/master.vim

Microsoft Assembler: http://www.datatone.com/~robb/vim/syntax/masm.vim

Environment: FreeBSD; WindowsNT and Windows95.

Dr. Charles E. Campbell, Jr. - Charles.Campbell@gsfc.nasa.gov [980817,990520,990830]

http://www.erols.com/astronaut/vim/

Syntax files:

Amiga Scripts amiga.vim.gz

CSH (C shell) scripts: csh.vim.gz

DCL (Dec's Control Language): dcl.vim.gz

Elm Filter ($HOME/.elm/filter-rules): elmfilt.vim.gz

Exports (Unix's /etc/exports) exports.vim.gz

Lex lex.vim.gz

Lisp lisp.vim.gz

Maple V r1.4 maple.vim.gz

SH (Sh Bash Ksh): sh.vim.gz

Sendmail sm.vim.gz

Tag Files tags.vim.gz

TEX (LaTeX, TeX) tex.vim.gz

Vim vim.vim.gz

Xmath xmath.vim.gz

Xxd xxdbin.vim.gz

Yacc yacc.vim.gz

Note: Dr. Charles Campbell has written so many syntax files that I can hardly keep up with it. Nice work, Chip! On his website you will also find his own tag file generator **hdrtag** which create tags for c/c++, c/c++ header files, lex, yacc, TeX (cites and labels), vim scripts (functions and syntax), and Maple V. Amazing! There is also info on how to use an English spell checker with Vim, some alignment code and maps, and some Vim-scripts supporting network oriented file read/write (with commands :Nr and :Nw).

Capricelli, Thomas - orzel@yalbi.com [000927]

http://aquila.rezel.enst.fr/thomas/vim/

Syntax files:

kvim: http://aquila.rezel.enst.fr/thomas/vim/syntax/kscript.vim

Environment: Linux on a PowerMac (latest kernel), KDE-2 from cvs.

Note: Thomas Capricelli's page is the official page for "KVim".

Castagnetto, Jesus M. - jesusmc@scripps.edu [980827]

http://www.scripps.edu/~jesusmc/vim/

Lite (mSQL): http://www.scripps.edu/~jesusmc/vim/syntax/lite.vim

Chamarty, Sitaram - sitaram@diac.com [971229,980126,010220]

http://www.diac.com/~sitaram/vim/

COBOL: http://dimensional.com/~sitaram/cobol/ http://dimensional.com/~sitaram/cobol/cobol.vim http://www.diac.com/~sitaram/vim/cobol.vim [obsolete]

Environment: Linux-2.0.27 (RHL-4.2), AIX-4.1.2, VIM-5.0s on both; VIM-4.5 on HP-UX-10.10 and others...

Corks, Matt - mvcorks@uwaterloo.ca [970516]

http://www.csclub.uwaterloo.ca/u/mvcorks/vim/

Environment: various Unix versions

Matt Corks has also created a menu with the (complete?) character set for ISO-8859. Thus you can enter every character from a menu. This theoretically means that you can operate Vim complety via menu using the mouse. (eek! ;-)

Dara, Hari Krishna - haridsv@hotmail.com [991116,991117]

http://sites.netscape.net/haridsv/vim/

Syntax files:

Language: http://www.domain/~user/vim/syntax/language.vim

Binaries: http://www.domain/~user/vim/bin/os/

Environment:

Note: Functions for a "directory stack" (recall previously used current directory) and "deletion recovery in insert mode" for deletions commands CTRL-W (delete last word) and CTRL-U (delete current line).

Duperval, Laurent - laurent@grafnetix.com [971107,971217]

http://www.grafnetix.com/~laurent/vim/

Laurent's Cool Vim macros

URL: http://www.grafnetix.com/~laurent/vim/macros.html

A collection of some stunning macros. Includes a list number generator, TicTacToe and a file browser. I am not making this up! ;-)

Icons: http://www.grafnetix.com/~laurent/vim/icons.html

Some of the icons may appear broken due to the strange webserver. Just reload the page a few times to see all icons.

Environment: Various Unix flavours

Note: Laurent Duperval was the maintainer of the VIM FAQ until Spring 1998. Thanks, Laurent!

Eckert, Hans-Christian "Ripley" - ripley@nostromo.in-berlin.de [980312]

http://www.in-berlin.de/user/nostromo/vim/

Note: Hans-Christian has done some developing on the MacOS port of Vim.

Elbers, Henk - elbers@bng.nl [970625]

Note: Henk Elbers was the porter of the VMS version.

Ehrens, Philip S. - imbe@primenet.com [971208,980828]

http://www.primenet.com/~imbe/vim/ [defunct 980828]

Environment: DOS and Linux.

Note: Phil once offered vim binaries for DOS 32-bit DOS/Win [and optimized versions for 486 and Pentium] - but he closed his service as of 980205 - unfortunately. :-(

Exel, Otavio - oexel@economatica.com.br [971215]

http://www.economatica.com.br/oexel/

http://www.economatica.com.br/oexel/vim [todo]

Note: Otavio Exel maintains a page on "pickfiles" (index of filenames for easy access within editors) and also has some pickfile mappings for Vim.

Faylor, Chris - cgf@bbc.com [980114,980515]

http://www.tiac.net/users/cgf/

Note: Chris Faylor has ported vim for WindowsNT and it made available with Cygnus release. On his page you can get the binary for vim-5.0p (as well as less-332 and perl-5.004_04) together with the required cygwinb19.dll.zip .

Feige, Bernd - feige@ukl.uni-freiburg.de [971124,971215]

http://home.t-online.de/home/Bernd.Feige/vim.htm

LaTeX BibTeX files: http://home.t-online.de/home/Bernd.Feige/bib.vim

Environment: Linux-2.0.32

Fisher, Benji - fisherbb@BC.edu [000602,001219]

http://home.netscape.com/websites/ [001219]

http://sites.netscape.net/benjif/vim/ [obsolete]

Note: "The most ambitious goal for this site is to become a standard reference for editing TeX and LaTeX with the vim editor." Benji Fisher has written a LOT of functions with Vim.

Fleiner, Claudio - claudio@fleiner.com [971210]

http://www.fleiner.com/vim/

ASN: http://www.fleiner.com/vim/syntax/asn.vim

CSS (cascading style sheets): http://www.fleiner.com/vim/syntax/css.vim

HTML: http://www.fleiner.com/vim/syntax/html.vim

Java: http://www.fleiner.com/vim/syntax/java.vim

JavaCC: http://www.fleiner.com/vim/syntax/javacc.vim

JavaScript: http://www.fleiner.com/vim/syntax/javascript.vim

make: http://www.fleiner.com/vim/syntax/make.vim

Sather: http://www.fleiner.com/vim/syntax/sather.vim

Environment: Linux-2.0.29 on a Laptop and a Pentium; AIX 4.*.

Gonsalves, C. Laurence - clgonsal@kami.com [980827,981104]

http://www.cryogen.com/clgonsal/vim/

asmselect - select betweeen various assemblers:

http://www.cryogen.com/clgonsal/vim/syntax/asmselect.vim

NASM: The Netwide Assembler (v0.97):

http://www.cryogen.com/clgonsal/vim/syntax/nasm.vim

PL/SQL: http://www.cryogen.com/clgonsal/vim/syntax/plsql.vim [981104]

Griffin, Benjamin Elijah "Eli the Bearded" - eli@NetUSA.Net [980114]

http://www.netusa.net/~eli/src/vim.html

Contents: Macros, macros, macros: Maze solver, Towers of Hanoi, Universal Register Machine, HTML macros. And links, of course.

"How tags work" http://www.netusa.net/~eli/src/tags.html

Note: If you want to notify him of some post on Usenet then just crosspost to the group *alt.fan.e-t-b*. This guy is a Vi freak! And he knows a lot about procmail, too. :-)

Foucher, Herve - Herve.Foucher@helio.org [981119]

http://www.helio.org/vim/

Syntax files:

SMIL: http://www.helio.org/vim/syntax/smil.vim

Environment: work: Windows NT (PII 450Mhz) home: Linux RedHat-5.2, windowmaker-0.20.2; Windows95 (Cyrix 166+)

Note: Herve Foucher is using Vim mainly for HTML and Java.

Fraust, Jack - guilesf2@hotmail.com [000829]

http://www.angelcities.com/members/vimpower/

http://www.angelcities.com/members/vimpower/syntax.html

Garcia-Suarez, Rafael - garcia_suarez@hotmail.com [990831]

http://altern.org/rgs/vim/

Environment: vim 5.4 on WinNT 4.0; vim 5.3 on Linux 2.2 (RedHat 6.0)

Glass, Brian - bjg@math.ams.org [970627,980505]

http://www.lni.net/bjg/vim/

Note: As Brian Glass told us on 980430 he does not have the time to maintain the files any more. But it is ok with him if anyone wants to improve it.

Environment: home: Windows95. work: Digital Unix and WindowsNT.

Guckes, Sven - guckes@vim.org [970623,980609,000516]

http://www.math.fu-berlin.de/~guckes/Vim/

Environment:

Home: i386 and i486 with FreeBSD-2.2.6 and Linux SUSE-5.2

Work: SunOS 4.1.3, Solaris

Sven quit DOS and Windows in 1992 - but now uses them only for testing Vim (and to play "Sherlock" once in a while). Sven does have Vim installed with the options "+GUI +X11" but hardly ever uses any of its features (mouse and menus).

Sven is the maintainer of the Vim Pages on www.vim.org and frequently scans the newsgroup comp.editors to help answering questions about Vim and related programs.

Guldberg, Preben "Peppe" - c928400@student.dtu.dk [971216,980622]

http://www.student.dtu.dk/~c928400/vim/

Fortran: http://www.student.dtu.dk/~c928400/vim/syntax/fortran.vim

Matlab: http://www.student.dtu.dk/~c928400/vim/syntax/matlab.vim

mutt setup files: http://www.student.dtu.dk/~c928400/vim/syntax/muttrc.vim

slrnrc (slrn setup file): http://www.student.dtu.dk/~c928400/vim/syntax/slrnrc.vim

slrnsc (slrn score file): http://www.student.dtu.dk/~c928400/vim/syntax/slrnsc.vim

slrnsl (slrn slang file): http://www.student.dtu.dk/~c928400/vim/syntax/slrnsc.vim

Environment:

1. Linux 2.0.32/2.1.65, RedHat 5.0

2. HP-UX B.10.20 A 9000/780 2012879787 two-user license

Note: Preben Guldberg has a very nice vim page - lots of syntax files, too. He made the vim maillists available as compressed mailbox folders, and he quite active on comp.editors giving useful answers. He deserves a "high score" for this! :-)

Haney, Steve - Steve_Haney-p21393@email.mot.com [971023,981007]

http://www.usaserve.net/users/shaney/vim/ [981007]

Binaries: AIX-4.1.4 on PowerPC

Old: www.goodnet.com/~shaney/vim/

Hack, Stephen - [970814,010328]

http://www.students.uiuc.edu/~shack/vim/ [obsolete]

man.vim - syntax file for Unix manuals: http://www.students.uiuc.edu/~shack/vim/syntax/man.vim

Environment: Debian, HPUX, Cygwin (home/work)

Hannedouche, François-Xavier - fixounet@free.fr [990915]

http://fixounet.free.fr/vim/

Environment: Work: UltraSparc 5 with Solaris; Home: Windows 98 / Linux Mandrake 6.0

Note: Info on cross referencing symbols with Vim using a perl script.

Harrsion, David C. Jr. - dharriso@pigseye.kennesaw.edu [971011,980918]

http://pigseye.kennesaw.edu/~dharriso/vim/

Environment: Silicon Graphics IRIX (5.3, 6.4)

Binaries: IRIX-5.3, IRIX-6.3, IRIX-6.4, IRIX-6.5

Hedlung, Hans - hans@cs.umu.se [960512,980622]

http://www.cs.umu.se/~hans/vim/ [todo]

vim-3.0 refcard.txt in HTML: http://www.cs.umu.se/~hans/vim.html

Hemert, Jano van - jvhemert@wi.LeidenUniv.nl [9xxxxx,980522]

http://www.wi.leidenuniv.nl/~jvhemert/vim/

Note: Jano van Hemert's page about Vim macros is really nice. :-)

Generic Entry - please fill out and send to me.

Lastname, Firstname - user@domain [DATE]

http://www.domain/~user/vim/

Syntax files:

Language: http://www.domain/~user/vim/syntax/language.vim

Binaries: http://www.domain/~user/vim/bin/os/

Environment:

Note:

Heimann, Sonia - niania@netsurf.org [980202,980522]

http://www.netsurf.org/~niania/vim/ (dir contents only)

Perl: http://www.netsurf.org/~niania/vim/syntax/perl.vim

Environment: Linux and WindowsNT

Note: Sonia Heimann handed on the maintenance of the perl.vim to Nick Hibma.

Hesse, Arndt - hesse@self.de [971204,971229]

http://www.self.de/~hesse/vim/ [todo]

info on syntax files: http://www.self.de/~hesse/vim/syntax/

SmallTalk: http://www.self.de/~hesse/vim/syntax/smalltalk.vim

Note: This "smalltalk.vim" is included in the distribution as "st.vim".

Hibma, Nick - nick.hibma@jrc.it [980703,990302]

http://www.etla.net/~n_hibma/vim/

Perl: http://www.etla.net/~n_hibma/vim/syntax/perl.vim

Environment: FreeBSD 2.2.x/3.x/4.0, Solaris 2.4

Note: "yes, lot's of them, in my wallet."

Hiebert, Darren - darren@hiebert.com [980310,991208]

http://home.hiwaay.net/~darren/

Note: Darren Hiebert is the author of the utility "ctags":
http://fly.hiwaay.net/~darren/ctags/.

Darren does not have a page on vim - unfortunately; it would be interesting to see his vim setup.

Hlavacek, Jan - lahvak+www@math.ohio-state.edu [980216]

http://www.math.ohio-state.edu/%7Elahvak/software/vim.html

SLang: http://www.math.ohio-state.edu/%7Elahvak/software/vim/syntax/slang.vim

Slrn score files: http://www.math.ohio-state.edu/%7Elahvak/software/vim/syntax/score.vim

Environment: SunOS 2.5.2 and 2.6

Hunsaker, Bruce - BNHunsaker@chq.byu.edu [980515]

Environment: VMS

Note: Bruce Hunsaker's patches for the VMS version were added to vim-5.2c. On 980515 he said he'll set up a page...

Jensen, Thomas - JensenThomas@web.de [971128,990614,010220]

http://home.pages.de/~jensen/

http://home.pages.de/~jensen/junk/vimrc.txt

OS: Many different machines and OSes, mostly UNIX (including Linux)

Note: Thomas Jensen is the author of "tal", the "trailer alignment" filter program:
http://home.pages.de/~jensen/tal/

Jones, Alun "Penguin" - auj@aber.ac.uk [970912,971217]

http://www.aber.ac.uk/~auj/ (home page)

Environment: Acorn's "RISC OK"

Note: Alun Jones once ported Vim to the Archimedes (see the file os_archi.txt of the distribution). Haven't heard from him in a while.

Kallingal, Rajesh - rajesh@oakbrook.oak.isc-br.com [971009]

Environment:

NOTE: Rajesh Kallingal once converted the vim-3.24 helpfiles to HTML:

URL: ftp://ftp.oce.nl/pub/misc/vim/beta-test/html-docs

Currently this is just an ftp link. I do hope that Rajesh will put up a web page where you can get a look at the latest version as well as download all of them as a tar.gz file.

Kang, Sirtaj Singh - taj@kde.org [971008]

http://www.ph.unimelb.edu.au/~ssk/vim/

Environment: OSF/Alpha, RS6K/AIX, Linux/x86

Scripts: vim2html - a perl script that converts vim-5 helpfiles to html.

http://www.ph.unimelb.edu.au/~ssk/vim/vim2html.pl

vim-5.0g helpfiles: http://www.ph.unimelb.edu.au/~ssk/vim/help.html

NOTE: Sirtaj Kang might look into creating a vim binary for KDE.

Köhler, Thomas - jean-luc@picard.franken.de [970113,980612,000327,010315]

http://jeanluc-picard.de/vim/

http://www.mayn.de/user/jean-luc/vim/ [obsolete]

Prolog ("*.pl", "*.pdb"): http://jeanluc-picard.de/vim/syntax/prolog.vim

"Motif UIL" ("*.uil", "*.uit"): http://jeanluc-picard.de/vim/syntax/uil.vim

Binaries: for all systems of the following environments:

Environment: vim-5.7 on Linux-2.2.x i386, Linux-2.4.x i386, Linux-2.2.x S/390, SunOS 5.6/5.7, and Windows NT. vim-6.0-alpha-releases on Linux-2.2.x i386

Kol, Tomer - tkol@psl-palm.technion.ac.il [990302]

Vim and LaTeX: http://www.ee.technion.ac.il/~tomer/TeX/

tkTeX: http://www.ee.technion.ac.il/~tomer/TeX/VIM/TKlatex.vim

Environment:

Note: Tomer Kol has created a "LaTex system to create lectures notes etc. using Hebrew and hyperref (to create PDF files). Vim serves as the interface."

Kopányi, Sándor - sandor.kopanyi@essnet.se [980416]

no vimpage yet

Environment: VAX, VMS

Note: Sándor Kopányi got vim-4.5 to work on VMS. I hope he'll try to get vim-5.1 wotking, too, and maybe even write some info about this (to be added to "os_vms.txt").

Kraai, Matt - kraai@andrew.cmu.edu [980617] [todo]

Note: Matt Kraai maintains vim's entry on www.freshmeat.net.

Lee, Scott E. - ScottLee@pobox.com [971125]

http://www.GeoCities.com/Athens/5679/ (home page)

Binaries: Data General DG/UX, both 88k and x86. However, I could not find these on his website. :-/ [990205]

Environment: DG/UX (Unix), Win95.

Note: Scott Lee ported vim to Data General Unix (DG/UX).

Leonard, Beth - beth@slimy.com [990629,990702]

http://frost.slimy.com/~beth/vim/

VERA: http://frost.slimy.com/~beth/vim/syntax/vera.vim

Lottem, Avner - alottem@iil.intel.com [970904,970912]

(no vim page)

Environment: Linux-2.0.30, Win95/WinNT-4.0, DOS 16bit, AIX-3.2.5. Including GUI version (except DOS). Trying to keep up with the latest ALPHA versions.

Avner is the author of the helpfile about "rightleft mode" (rightleft.txt) using vim for writing in Hebrew. Just ask him if you need help!

"The shell setup dynamically creates the environment (EXINIT, VIMINIT, GVIMINIT) in .cshrc (if $TERM was changed). This is a peculiar setup, because I use both vi and vim, and I wanted to have common options in one place."

Matsumoto, Yasuhiro - mattn@mail.goo.ne.jp [000117,000404]

http://hp.vector.co.jp/authors/VA020411/ [000404]

http://www.cis-net.co.jp/matsu/vim/ [obsolete]

Note: Binaries with "+multibyte".

McKee, Sean - mckee@misslink.net [971119]

http://www.misslink.net/mckee/vim/

Telix Salt: http://www.misslink.net/mckee/vim/syntax/telixsalt.vim

Dos INI files: http://www.misslink.net/mckee/vim/syntax/dosini.vim

Century Term Command Script: http://www.misslink.net/mckee/vim/syntax/cterm.vim

Make Menu: http://www.misslink.net/mckee/vim/rc/makemenu.vim

Environment: Windows95, SCO OpenServer 5, Linux (Slackware 2.0.30) Home: (Pentium 75) and Acer Laptop (Pentium 100)

Mann, Christopher G. - r3cgm@cdrom.com [970912]

Environment: FreeBSD

Christopher Mann is the webmaster of cdrom.com and also edits the homepage of hornet.org with Vim:

http://www.hornet.org/

Martinson, Eric C. - ericmart@teleport.com [970812]

http://www.teleport.com/~ericmart/vim/

Environment:

home: P5-166Mhz, Linux (Redhat 3.0.3; kernel 2.0.13).

office: SPARCstation-20, 150Mhz, 96MB RAM, 1MB cache; SUNOS-4.1.[34] or Solaris-2.5.1; vim-4.6, fvwm-2.943, rxvt-???.

McMullen, John - johnmc@interix.com [990301]

HomePage?

Binaries: Interix binaries, maybe? :-)

Environment: Interix (which version?)

Note: John McMullen is trying to port Vim to Interix.

Moolenaar, Bram - bram@vim.org [970912,010328]

http://www.moolenaar.net/vim.html

Syntax files: 2html.vim, c.vim, colortest.vim, diff.vim, help.vim, model.vim, modula2.vim, nosyntax.vim, scripts.vim, **syntax.vim**, vgrindefs.vim, viminfo.vim.

Environment: "My Amiga is a good old A2000, with 1MB RAM. It has a 68030 CPU card at 25MHz with 2MB RAM. Speedwise this is comparable to an Intel386. The screen is a 14" Philips 8833 (TV frequencies). I would upgrade it to a 68040 or 68060, if I would find a cheap CPU card. Would be better to get an A4000 and a better monitor, but the Amiga is a dead-end, I don't want to invest much in it anymore.

My "old" PC is a 200 MHz AMD K6, 64MB RAM, four disks totalling 11 Gbyte. It runs FreeBSD-2.2.1. The screen is 17" (1280x1024).

My "new" PC is a 233 MHz AMD K6, ... This is my main workhorse now.

I've got a 10MHz Ethernet between the two PCs for easy file access. For the Amiga I need to use 720K floppies (that's why the Amiga distribution files have been cut down now)."

Note: Bram Moolenaar is the original author of Vim. Thanks for all your work, Bram!

Moore, Paul - Paul.Moore@uk.origin-it.com [971016]

NOTE: Paul Moore might do a page on "porting vim on windows". I wonder what has become of this idea...

Mueller, Carl - cmlr@troi.cc.rochester.edu [981217,990118]

http://www.math.rochester.edu:8080/u/cmlr/vim/

Modified syntax files:

Scripts (recognizes script files):

http://www.math.rochester.edu:8080/u/cmlr/vim/syntax/scripts.vim

TeX:

http://www.math.rochester.edu:8080/u/cmlr/vim/syntax/tex.vim

Environment: ???

Notes: Improvements to the tex.vim syntax file: Additional abbreviations, mappings and menus for gvim; additional functions for matching bracket completion and deletion, for changing brackets, for putting in "\left...\right" or "\bigg", and for changing "\left...\right" to "\bigg".

Muth, Klaus - muth@hagos.de [980527]

http://unitopia.mud.de/~monty/vim/

LPC: http://unitopia.mud.de/~monty/vim/syntax/lpc.vim

RosiSQL: http://unitopia.mud.de/~monty/vim/syntax/rosisql.vim

Environment:

Home: Linux 2.0.33 + Vim 5.2b

Work: Solaris 2.4/2.5.1 /w Motif on Classic + Ultra Sparc, Vim5.1 + Vim5.2b

Note: "What about Vim on Qt libs? What about kvim? ;)"

Nagle, Adrian vim-www@naglenet.org [980609,980609,000525]

http://naglenet.org/vim/vim.html

SINDA: http://naglenet.org/vim/syntax/sinda.vim

TAK: http://naglenet.org/vim/syntax/tak.vim

TRASYS: http://naglenet.org/vim/syntax/trasys.vim

Environment:

Work: WindowsNT-4.0

Home: Linux2.0+ from slackware

Nenadov, Mark - marknen@usa.net [000903,000905]

http://www.freebox.com/marknenadov/vim/

Environment: Home/School: Windows 98; GVIM 5.7 & VIM 5.7 for DOS. Home: ZipSlack Linux - VIM 6.0g

Nassen, Kent - knassen@umich.edu [970927,980703]

http://www-personal.umich.edu/~knassen/vim/

http://www-personal.umich.edu/~knassen/vim/ksyntax.html

SAS: http://www-personal.umich.edu/~knassen/vim/sas.vim

SPSS: http://www-personal.umich.edu/~knassen/vim/sps.vim

cbentr/cbgen: http://www-personal.umich.edu/~knassen/vim/cbg.vim

Nielsen, Michael - mni@dde.dk [970912,971118,980522,990707,990708]

http://vip.cybercity.dk/~ccc30647/

was: http://www.image.dk/~mikeni/vim/ [990707: dead]

Environment:

Pentium 166MHz: win95 and Linux RedHat 4.1, 32MB RAM, 3.5GB Harddrive, 8xCDROM (IDE), 2xCDR (scsi), WINfast286, Sound galaxy Pro.

Amiga 4000 68040@25Mhz: (soon to be 68060@50Mhz/604e@200Mhz with +64MB). 18Mb Fast, 2MB Chip, 3.7GB Harddrive space, PicassoII, MFCIII, 2xCDROM (scsi), 8xCDROM (IDE).

Monitor: shared 17" monitor 1280x1024x8bit. 10Mhz Ethernet between the machines.

Binaries: Amiga

http://vip.cybercity.dk/~ccc30647/vim/

was: http://www.image.dk/~mikeni/vim/download.html [990707: dead]

Michael Nielsen is coding a GUI for the Amiga. He might also attempt to include a simple HTML browser with Vim. "The Amiga is my main Development platform, the Linux box is mainly a file server for backups, and CDROM writing. (I do enough unix development at work)."

Norbäck, Martin - d95mback@dtek.chalmers.se [???,980703]

http://www.dtek.chalmers.se/~d95mback/vim/

Haskell: http://www.dtek.chalmers.se/~d95mback/vim/syntax/haskell.vim

O'Brien, David E. - obrien@FreeBSD.org [971120]

http://relay.nuxi.com/vim/ or http://relay.nuxi.com/www.vim.org/ (daily WWW mirror of http://www.vim.org/)

Binaries: FreeBSD and Linux.

FreeBSD Binaries:

ftp://ftp.freebsd.org/pub/FreeBSD/packages-current/editors/vim*.tgz

Linux Binaries:

ftp://ftp.nuxi.com/pub/vim/linux/

Environment: FreeBSD-2.x, Solaris-2.5+, Ultrix-4.4, HP-UX-10.20 - many of each type :-)

Notes: David E. O'Brien maintains the first Vim distribution mirror in the USA (ftp://ftp.nuxi.com/pub/vim also known as ftp://nuxi.ucdavis.edu/pub/vim). David is also the maintainer for the Vim ports under FreeBSD and OpenBSD.

Pascoe, David - pascoedj@ozemail.com.au [980514,980703]

http://www.ozemail.com.au/~pascoedj/

Vim Page: http://www.ozemail.com.au/~pascoedj/vim/ [soon?]

Note: [todo]

Piefel, Michael - piefel@informatik.hu-berlin.de [010112]

http://www.informatik.hu-berlin.de/~piefel/vim/

Syntax Language: Kimwitu++ http://www.informatik.hu-berlin.de/~piefel/vim/syntax/kwt.vim

Environment: Debian GNU/Linux

Note: Michael Piefel also offers a PO file editing mode.

Raisky, Oleg Yu. - olrcc@scisun.sci.ccny.cuny.edu [971215,971216,000414]

http://physlab.sci.ccny.cuny.edu/%7Eorycc/vim-main.html

http://scisun.sci.ccny.cuny.edu/~olrcc/vim/ [obsolete]

Environment: Linux RedHat 4.1, Win'95

Note: Oleg Raisky created the "Vim Guide", a printable version of the Vim Quick Reference (":help quickref.txt"). The guide was set with LaTeX and is available in PostScript reading for printing for sizes "A4" and "US letter" in both "plain" and "booklet" format.

Rebbechi, Donovan - elflord@pegasus.rutgers.edu [980731,980731]

http://pegasus.rutgers.edu/~elflord/vim/

Bash (shell):

http://pegasus.rutgers.edu/~elflord/vim/syntax/bash.vim

SPEC - Linux RPM Files:

http://pegasus.rutgers.edu/~elflord/vim/syntax/spec.vim

Reilly, George V. - George@Reilly.org [971217,990204]

http://www.halcyon.com/gvr/vim.html

coming up: http://george.reilly.org/

Note: George Reilly was the porter of the Windows versions for vim-4. Thanks a lot for getting this started, George!

Old addresses: georgere@microsoft.com gvr@halcyon.com

Riehm, Stephen - stephen.riehm@bigfoot.com [971201,990817]

http://www.bigfoot.com/~stephen.riehm/

Environment: MacOS, AIX, HPUX, Linux.

Note: Stephen Riehm has some generic mappings to share for almost any kind of editing. Also, "ctags for shell scripts". [990817]

Riiser, Haakon - hakonrk@fys.uio.no [980305,981009,990914]

Syntax files with Screenshots!

http://www.uio.no/~hakonrk/vim/

fvwm (versions 1 and 2):

http://www.uio.no/~hakonrk/vim/syntax/fvwm.vim

printcap/termcap:

http://www.uio.no/~hakonrk/vim/syntax/ptcap.vim

sed (stream editor) scripts:

http://www.uio.no/~hakonrk/vim/syntax/sed.vim

Simula:

http://www.uio.no/~hakonrk/vim/syntax/simula.vim

de la Rosso, Martini - moro@dotcom.ee [010411]

http://www.math.ut.ee/~moro/vim/

Environment:

Note: Martini keeps a setup file with abbreviations for java coding:

http://www.math.ut.ee/~moro/vim/java.vimrc

Rios, Frank - frankr@qni.com [980114]

http://www.qni.com/~frankr/ (home page)

Vim Page: http://www.qni.com/~frankr/vim/ [todo]

ASP ("*.asp" and "*.asa"):

http://www.qni.com/~frankr/vim/syntax/ASP.VIM

Environment: Windows95, WindowsNT

Note: Frank Rios may create an "Active Server Page" for VIM.

Riswick, Jos van - josvanr@mbfys.kun.nl [971013,971017]

http://septimius.mbfys.kun.nl/~josvanr/vim/

NOTE: Jos van Riswick has menu macros for non-gui vim-4.6 and macros for gui-vim implementing a file browser.

Rochholz, Hermann - Hermann.Rochholz@gmx.de [970610,990619,990831,000413]

http://www.rochholz.de/vim.html [000413]

http://inrdp2.fzk.de/hermann/vim.html [obsolete]

Interactive Data Language:

http://www.vim.org/syntax/idlang.vim

SpeakEasy [not included with Vim any more]:

http://www.rochholz.de/syntax/speakez.vim.gz

Hermann Rochholz wrote a set of "TeX Menus" for Vim (see picture).

Schemenauer, Neil - nascheme@acs.ucalgary.ca [980613,980616,990822]

http://www.ucalgary.ca/~nascheme/vim/

Syntax files:

DCL (for VAX): http://www.ucalgary.ca/~nascheme/vim/syntax/dcl.vim

PL/I: http://www.ucalgary.ca/~nascheme/vim/syntax/pli.vim

Python: http://www.ucalgary.ca/~nascheme/vim/syntax/python.vim

Environment: Linux on a AMD K6-2

Scott, Ken - kscott@pcisys.net [971007,971008]

http://www.pcisys.net/~kscott/vim/

Informix ESQL/C: http://www.pcisys.net/~kscott/vim/syntax/esql.vim

PowerBuilder export files: http://www.pcisys.net/~kscott/vim/syntax/powerb.vim

Environment: WindowsNT-4.0

Binaries: Win32

Seidman, Gregory - gseidman@acm.org [980623]

VRML 2.0 (VRML97): http://zing.ncsl.nist.gov/~gseidman/vim/syntax/vrml.vim

Environment: Solaris, Linux, IRIX, and MacOS

Seibert, Olaf - rhialto@polder.ubc.kun.nl [971021,971022]

No vim page. But his homepage definitely tells you why you don't want to use frames or the blink tag: http://polder.ubc.kun.nl/~rhialto/

Binaries: http://polder.ubc.kun.nl/~rhialto/

Vim-4.5 and Vim-5.1 for BeOS DR8, AAPR. [Newer versions of BeOS already include Vim-4.5.]

Environment:

Home: Amiga 4000 with Motorola 68040 at 25 MHz, 12M RAM, 2G disk. BeBox with Dual PPCs 603 at 66 MHz, 48 M RAM, 1G disk.

Work: NetBSD/i386 with XFree86 3.3.

Binaries: BeOS

Note: Olaf intends to do a vim port for BeOS. So far he has done the text mode BeOS port and is working on the GUI. Will he do a port of vim-4.6 or vim-5.0s?

Shan, Ken - ken@digitas.harvard.edu [980511]

http://www.digitas.harvard.edu/~ken/vimrc/

Vim Page with links to setup files:

Environment: BSDI 2.1

Note: Ken Shan has quite a few sample setup files for editing C/C++, Matlab, Perl, RCS, scripts, TeX, and some more, eg a file browser ("directory editing"), and detecting text with Big5 coding, coloring pgp signed text URLs, and mail headers.

Sharpe, Michael - msharpe@bmc.com [000516]

http://www.irendi.com/vim/

Note: The vimrc and a function file HTMLized to show Vim's syntax coloring.

Shiran, Mortaza G. - shiran@parscom.com [971218,981005,001012]

http://www.parscom.com/www/vim/vim_farsi.html

Note: Mortaza's page has some screenshots - however, your browser needs support for ISO-8859-6 to display this page properly. Mortaza Shiran has added Farsi (Persian) support to Vim and also manages the site "parscom.com"

http://table.jps.net/~shiran/www/p_soft.html [obsolete]

Siegmann, Paul - pauls@euronet.nl [980713]

http://www.euronet.nl/~pauls/vim/

XML: http://www.euronet.nl/~pauls/vim/syntax/xml.vim

Environment:

Note:

Slootman, Paul - paul@wurtel.demon.nl [980114]

Environment: OS/2, UNIX, Linux

Note: Paul Slootman offered to help with installation of Vim on the operating systems he is using.

Socher, Guido - eedgus@eed.ericsson.se [971009]

NOTE: Guido Socher once converted the vim-3.0 refcard.txt to html:

http://www.math.fu-berlin.de/~guckes/vim/vim-3.0.refcard.html

St-Amant, Dany - dany.stamant@sympatico.ca [980219,990223]

http://www3.sympatico.ca/dany.stamant/vim/

Note: Dany St-Amant is porting vim-5 to MacOS.

Starsmeare, Keith - kxs@bigfoot.com [971204]

http://www.tardis.ed.ac.uk/~keith/ (home page)

Note: Keith Starsmeare was looking for help to install vim on these systems: DG and Dynix/PTX. He told me that he succeeded in doing so now. Maybe he'll maintain a page about vim where he'll offer the binaries? :-)

Steden, Klaus - rubella@antipathy.org [990828]

Binaries: http://www.rassilon.net/~vitamink/vim/

BSD/OS 4.0, SunOS 5.6 x86 (Solaris 2.6 x86)

Environment: Work: IRIX, Solaris, Linux; Home: BSD/OS, IRIX, Solaris, Linux, +OpenBSD

Thomas, Stephen - Stephen.Thomas@isltd.insignia.com [971208]

Inform: inform.vim

Environment:

Home: Linux 2.1.* series, based on Red Hat 4.1, on a 200MHz PPro 128Mb machine.

Work: HPUX 10.20, on a HPPA1.1 735/125. Not as nice as my Linux box :-)

Note: Stephen Thomas is the maintainer of the syntax file for "Inform" source code. And his syntax file didn't make it into the release of vim-5.0s it is available now locally as syntax/inform.vim. Hopefully, this will make it into the next distribution. Maybe Stephen will be supporting it from a web page of its own, too? :-)

Tennent, Robert - rdt@qucis.queensu.ca [971002]

Robert Tennent might try to produce a "barebones" version of Vim for Linux, ie a very small version without any of the special compile options. Would be nice if he could maintain it, too.

Uhl, Mike - Mike Uhl ShoeLeather_Express@RadioLink.net [990308]

Note: Mike Uhl is a minimalist: He is using vim-3 because he has no space for the bigger versions of vim. He might try vim-4 or vim-5 as soon as he gets more space. / He answeres posts on comp.editors, too. (And, yes, I once asked him for the reason of his username. ;-)

Urban, Thomas Scott - tsurban@home.net [971218,980703,000502]

http://members.home.com/tsurban/vim/

http://www.blarg.net/~urban/vim/ [obsolete]

Povray syntax file:

http://members.home.com/tsurban/vim/syntax/pov.vim

Note: Thomas Scott Urban also maintains a setup for **HTML menus** - look at his page for a screenshot.

Verdoolaege, Sven - skimo@breughel.ufsia.ac.be [971217]

Note: Sven Verdoolaege implemented the perl patches for Vim.

Zachmann, Gabriel - zach@igd.fhg.de [980910,981116]

http://www.igd.fhg.de/~zach/vim/

Ziegler , Austin - aziegler@solect.com [971009,990226]

http://fantome.vnet.net/vim/ [*]

Proc*C (Oracle): http://fantome.vnet.net/vim/syntax/proc.vim [*]

SQL*Forms (Oracle): http://fantome.vnet.net/vim/syntax/sqlforms.vim [*]

Note: These webpages were not available in Jan 1999 and probably will be for some time. But you can request them from Austin via email.

"I am in the process of getting my Pro*C and SQL*Forms syntax files set up for "public consumption", but I will need to talk to the maintainers of the C/C++ and/or SQL syntax files in order to get them ready (I don't want to end up redoing their work)."

and last not least:

Silver's Petz Site [000419]

http://silver.trica.com/petz/

Note: screenshot

---

todo

Add these users and ask for permission and info again.

- http://www.unb.ca/chem/ajit/vim.htm

Eymers, Lutz - ixtab@polzin.com [980704,980731]

http://home.pages.de/~ixtab/

Syntax Files Overview: http://www-public.rz.uni-duesseldorf.de/~eymers/download/cont_vim.html

http://www-public.rz.uni-duesseldorf.de/~eymers/stuff/lite.vim

http://www-public.rz.uni-duesseldorf.de/~eymers/stuff/msql.vim

http://www-public.rz.uni-duesseldorf.de/~eymers/stuff/php3.vim

http://www-public.rz.uni-duesseldorf.de/~eymers/stuff/phtml.vim

http://www-public.rz.uni-duesseldorf.de/~eymers/stuff/tf.vim

http://www-public.rz.uni-duesseldorf.de/~eymers/stuff/syntax_vim.tgz

Suggested URL for syntax files:

http://home.pages.de/~ixtab/vim/syntax/file.vim

Siegmann, Paul - pauls@euronet.nl [980704]

http://www.euronet.nl/~pauls/vim/

XML: http://www.euronet.nl/~pauls/vim/syntax/xml.vim

Binaries: ?

Environment: ?

Note: 980704: Sent email requesting permission for addition.

People and Pages to add

```
Johannes Zellner        http://www.zellner.org/vim/
johannes@zellner.org

Rafael Garcia-Suarez    http://altern.org/rgs/vim/
rgz@soft-mountain.com

        cdpark@sparcs.kaist.ac.kr Korean Port?!
        Jens M. Felderhoff, e-mail: jmf@infko.uni-koblenz.de
        John Velman velman@igate1.hac.com
        paul@murphy.nl
        Roger Knobbe RogerK@wonderware.com
        Sam Lantinga slouken@cs.ucdavis.edu
```

Looking for help on getting started:

```
        +perl
                Gossamer                gossamer@tertius.net.au
                John M. Klassa          klassa@aur.alcatel.com
```

Changes

980723

Moved the info on "Syntax Coloring Setup Files Maintainers" onto a page of its own - the "Vim Languages Page":

- http://www.vim.org/lang.html

980703

Removed the links to vimrcs. Please link these from your own vim page - thanks!

980514

- Removed the entry "Email:" to shorten them.
- Removed the "mailto links" as I do not want "address harvesters" to get them that easily off this page. The mail address is now put directly after the name.
- Only two dates now - the date of first entry and the date of last change. They also go after the name and email address.
- Removed all links to homepages as these usually can easily be derived from the address of the vim page by simply removing the "/vim".
- Made link to vim page the first link in each entry. That's the way I intended this page to be.

```
URL:          http://www.math.fu-berlin.de/~guckes/vim/user.html
URL:          http://www.vim.org/user.html (mirror)
Created:      Tue Jun 24 12:00:00 CET 1997
```

Send feedback on this page to
Sven Guckes guckes@vim.org

VIM - Wishlist

VIM is a great editor, but no program is perfect for everyone. VIM does not have everything, of course. A lot of things have been proposed so far and many are on the "todo list".

Please note: The "todo list" is maintained by the author and ships with the distribution of Vim (filename "todo"). It is **not** the same as this "Vim Wishlist".

- todo (user release)
- todo (developer release)

So send me your wishes - I will try to convince the developers that they need to be implemented. But before sending your wishes please check out this page and the todo list first to avoid "noise". Thanks!

Sven guckes@vim.org

Wishes Wishes Wishes

Here they are - a lot of wishes for VIM!

Non-RegExp Search Command

Add a search command which takes all characters literally, ie without any metacharacters as with regular expressions.

010314: Erwin Waterlander waterlan@natlab.research.philips.com

Paragraphs and Sections - optional definition

Add options to describe the delimiters of paragraphs and sections.

David Witten david.witten@template.com 000717

000718: It's already in the todo list:

```
:help todo
/redef
-  Be able to redefine where a sentence stops.  Use a regexp pattern?
-  Be able to redefine where a paragraph starts.
   For "[[" where the '{' is not in column 1.
```

Perl RegEx

Add support for Perl regular expressions (eg backreferences). Maybe add PCRE (Perl Compatible RE).

Warren Young warren@etr-usa.com [000601]

support for "crypto editing"

"Add a memory only mode to vim, i.e. a mode where you don't write to a file at ALL. instead, vim allocates some RAM (or even mlockall's it) and then treats that as the file buffer. Then on exit, it writes this out to a fifo, and that's it. The data NEVER touches disk. Reasoning: CRYPTO."

Justin Hahn jehahn@bu.edu [000328]

lisp option

Fix the behaviour of the option "lisp".

Matthis Buelow token@cip.informatik.uni-wuerzburg.de says this is broken.

Display cursor within offsets

Option

```
'cursoroffset' 'co'     numberlist  (default "0,0")
                        local to buffer
        Number of top/bottom offset lines for the cursor.
```

```
This keeps the cursor from getting to near the window borders
and enforces a scrolling when moving into the scroll offset,
thereby keeping some "context" of the text for the user.

The value "T,B" will prevent the cursor from being displayed
within the T top lines and B bottom lines of the current window.
If T+B exceeds the number of lines in the current window
then the cursor will always be displayed on the middle line.
However, the start and end of the buffer is always shown.
```

Michael Saunders michael@amtec.com and Sven Guckes [000222]

search across buffers

Add an option to allow searches to scan all open buffers. (Some people do not have a decent "grep" on the machines, apparaently.)

Micha berdi@my-deja.com [000130]

GUI: textwidth bar

Show the current textwidth with a vertical bar within the GUI window.

Devin Weaver devin@sonalysts.com 991118

support for template editing

Add support for editing of "templates". This allow editing of parts of a given text only ("fields"). You should be able to chose from a list of given values ("menu"), or type in answers. A restriction to the character set may be given by regular expressions.

Example:

```
Your Name:  [Mr|Mrs]  [\w\{40\}]
```

This allows to chose from "Mr" and "Mrs" and to type in a name using "words" up to a length of 40 characters.

Herve Foucher Herve.Foucher@helio.org 990802

cursor on tabs

Add an option which places the cursor on tabs onto the *first* column rather than the last column of the expansion.

Suggested by: Div Byzero on comp.editors [990726]

Vile has this option - "alt-tabpos".

coloring current line

Colorize the current line of editing with a lighter background, so you can see that line and your editing more easily.

Suggested by: Michele Zaina zaina@tlvhcp.vim.tlt.alcatel.it 990706

command line - expand color settings

Problem: When you want to change the color settings for a syntax group then you have to use a command to print out the current values and type them in again to change them.

Suggestion: Expansion of the value on the command line would be a lot better.

Example: Enter `:hi GroupName TAB` adds the current values to the command line.

Suggested by: Sven Guckes guckes@vim.org [990712]

startup - specification of a "filelist"

Add additional startup parameter to specify a file with filenames to edit.

Problem: On WindowsNT you always run into problems when using filenames which include spaces. Passing these filenames to Vim via the shell usually results in the names being split up, eg "foo(space)bar" results in two filenames "foo" and "bar". You could get around this problem if you could specify the filenames in another file and have this file be processed by Vim directly - instead of having the names broken by the shell.

Suggested by: Ed Peschko ed_peschko@csgsystems.com [990624]

X and GUI - startup with "-display host:0"

Add the startup parameter "display" to allow specification of a display on a (remote) host.

Suggested by: Nathan Clemons nathan@windsofstorm.net [990528]

:source!

Add the command ":source!" which does not complain if "file" does not exist. This should avoid the error message when the file does not exist. Makes it easier to write setup files as you do not need a function for testing the existence of the file.

Suggested by: Sven Guckes guckes@vim.org [990312]

Select and move statusline

The problem with resizing windows is that there are commands to change the size the current window by N lines - but dragging the statusline around with the mouse is easier. So I suggest:

Add a command that allows to select the statusline of the current window and allows to move it up or down. Maybe add another command to allow selection of the statusline of the above window.

Suggested by: Sven Guckes guckes@vim.org [990311]

Extra Buffer - Grep

Built-in grep command popping window with list of found files.

Suggested by: Ilya Tsindlekht ilyat@internet-zahav.net [990306]

More Magic

"magic characters in regexp for switching between case-sensitive and case-insensitive mode"

Suggested by: Ilya Tsindlekht ilyat@internet-zahav.net [990306]

Textmode Menus

Show a line with menus at the top and allow to acess them with function keys.

"Watcom GUI editor is called VIW, apparently it's GUI version of Watcom VI with modeless option turned on by default."

Suggested by: Ilya Tsindlekht ilyat@internet-zahav.net [990306] and Sven Guckes guckes@vim.org [990308]

Textmode Menus

Show a line with menus at the top and allow to acess them with function keys.

Suggested by: Ilya Tsindlekht ilyat@internet-zahav.net [990306] and Sven Guckes guckes@vim.org [990308]

Windows...

"overlapping resizeable windows"

Suggested by: Ilya Tsindlekht ilyat@internet-zahav.net [990306]

Option Status Line

Show an extra line with the value of some options. Example:

```
ai aw nocin nocp digraph et nopaste tw=76 nowrap vb
```

Optionally show unset boolean options in reverse.

Suggested by: Sven Guckes guckes@vim.org [990308]

editdir buffer

Add the command ":editdir" which opens a directory in a special buffer and adds some commands to edit files, create and delete files.

With "vertical split" of windows this should allow something like "midnight commander" to move files between directories.

Suggested by: Daniel Goujot goujot@mmfai.ens.fr and Sven Guckes guckes@vim.org [990302]

:subst extended changes in SED style

Allow to make changes for the Nth match and beyond:

```
:s/search/replace/N     replace Nth match
:s/search/replace/Ng    replace Nth match and beyond
```

Inspired by: GNU sed-3.02

Suggested by: Sven Guckes guckes@vim.org [990222]

support for debuggers (gdb and jdb)

"Integration with gdb and jdb debuggers so that source can be easily stepped through from within the vim editor."

Suggested by: John Petersen jmpeters@us.oracle.com [990216]

:version output from variable

Add a internal variable (say, "VIMVERSION") which carries the ":version" data. This allows insertion of this text easily and *fast* (much faster than with ":r!vim --version").

Suggested by: Sven Guckes guckes@vim.org [990208]

prompt-write=always

Add an option that makes Vim prompt for overwriting a file *always*.

Suggested by: Robert Webb RobertW@beam.com.au [990115]

:right justification - work on part of lines

The command ":right" works on *complete* lines only. But it would be nice if it would work on parts of lines, too.

Example: Assume that I have visually selected the text from "bar" to "di", both inclusive:

```
foo [bar baz              |
lah di] dah               |
                textwidth---^
```

Then ":ri" should produce this:

```
foo                  bar baz|
                   lah di dah|
                 textwidth---^
```

Suggested by: Jos van Riswick j.g.a.v.riswick@ele.tue.nl and Sven Guckes guckes@math.fu-berlin.de [990105]

Support for ActiveX

Michael wants to use vim with MSVC++. For this, vim must be an "ActiveX component".

Suggested by: Michael O'Brien mikeo@sgi.com [981124]

GUI - visual selection independent of cursor

Allow selection of text with the mouse but *without* changing the current position of the cursor.

Suggested by: Jeffrey George jgeorge@us.ibm.com and others 981106

GUI - menus with buffer list and "favourites"

Add menus with filenames: One menu showing the current list of buffers, one showing a list of recently accessed files, and one showing a list of "favourite files".

"The Windows standard for programs that operate on files (e.g., Word, Excel) is to show the last 4 most recently accessed file names at the bottom of the File pull-down menu. Clicking on one of the file names loads that file."

The vi clone "Lemmy" has a menu for "recently accessed files".

Suggested by: Richard Anderson Richard.Anderson@seaslug.org [981106]

filename completion - specify path

Allow definition of a "path" to look for filenames to be completed.

buffer list - LIFO

Maintain the buffer list as a LIFO structure (last in - first out). Whenever you enter a buffer it will be pushed on to of the list; optionally deleting a previous entry. Add commands to "push" and "pop" entries from the stack.

This allows to "go back" to previously accessed buffers in a "last accessed" fashion.

Suggested by: Ramesh Raman rameshr@india.hp.com [980908]

mouse on Linux console

Add support for the mouse on Lunix console.

Suggested by: Juergen Zeller zet@rupert.franken.de [980819]

Patches submitted by Ilya Tsindlekht ilyat@internet-zahav.net to the author and the mailing list in December 1998.

fileformat - new format "cpm"

Add fileformat "cpm" which is basically the same as DOS but adds the character ^Z (control-z) to the end of files.

Suggested by: John Spetz jspetz@idpnet.com

swapfiles - "swapfile" - local option for use of swapfile

The option "swapsync" allows to define the use of a swapfile. However, it is a *global* option for all buffers. Therefore you cannot turn of its use for some buffers. but this is very much desired.

Suggestion: Allow use of swapfiles locally:

```
'swapfile' 'sf'         boolean (default on)    local
        "set swapfile" creates a swapfile for the current buffer.
        [according tho these rules]
        "set noswapfile" closes the swapfile for the current buffer.
        [sync data?]
        Delete the current swapfile with command [example]
```

Suggested by: Jeff Youngstrom jeffy@sqi.com and Sven Guckes guckes@vim.org [980710]

command :q - close all help windows

Add an option to allow ":quit" on *one* help window to close *all* help windows.

Suggested by: Ralf Schandl schandl@de.ibm.com [980708]

command :bd - allow globbing

Expand the parameter to a list of files.

Example: ":bd *.c" deleted all buffers which match on all filenames "*.c".

Suggested by: Ron Aaron v-ronaar@Exchange.Microsoft.com [980616]

installation - option "install" for private colormap

DESCRIPTION

Suggested by: NAME EMAIL [DATE]

syntax coloring - color for line numbers

With ":set number" you can see the line numbers on the display - but you cannot assign a color to them. An option for this is desired.

Suggested by: Steven A. McCluney mccluney@slb.com [980330]

text object "comment"

Add "comments" as another text objects, to allow easy selection within source code.

Suggested object names: "ic" for "inner comment" (without surrounding comment delimiters) and "ac" for "all comment" (including the surrounding comment delimiters).

Examples: "dac" will delete the current comment while "yic" will copy the comment itself.

It should also be possible to define comments, so you can adapt this to the current language, eg HTML or LaTeX.

Suggested by: Sven Guckes guckes@vim.org [980612]

syntax coloring - show lines with marks on them

Allow to colorize lines which have a "mark" on them (see ":help marks").

Problem: How to color lines with multiple marks?

Suggested by: Vadim Zeitlin zeitlin@dptmaths.ens-cachan.fr [980507]

```
41     I would very much like to see a special mark
42 'a  at the side together with the line numbers.
43     That way you would not even need any color
44 'b  which might get in the way of syntax coloring.
```

Suggested by: Sven Guckes guckes@vim.org [001017]

display - one cursor for all windows to one buffer

Use one cursor for all windows that show a buffer. Moving it in one window will also move it in all other windows if it is visible therein. Sometimes this is more intuitive than having one cursor per window.

Inspired by Emacs' "follow mode".

Suggested by: Neil Zanella nzanella@cs.mun.ca [980505]

Installation - Secure Shell

Include a define that would trap out shell escapes on compile so that vim can be used from a restricted menu shell. Unfortunately this will lose the feature of using external commands (aka "filtering").

Suggested by: Anthony Halsell thalsell@rsn.hp.com [980424]

Virtual Editing

Allow jumps to non existant characters, ie jumps to locations on the screen which are after the end of a line. This allows moving around the screen without losing the current column even if the cursor can not be associated with a character in the current buffer. Changes to such positions should fill the non-existant characters with whitespace (ie, tabs using tabstops might should be optional).

Virtual editing usually helps beginners of editing as well as experienced people. Although it also adds some problems with a lot of commands. Anyway, this might easily become a top wish for Vim, too.

Virtual editing requires a lot of code to change as all jump commands need to be redefined. For example, what shall the command 'w' do when the cursor is in "virtual space"? It might be nice to still be able to use it as a jump, but what shall Vim do when it is used in conjunction with a deletion command, such as "dw"? This becomes even more tricky when you move the cursor within visual mode.

Virtual Editing and "marks": Should it be allowed to set a mark on a virtual space? I think the answer is "yes" as marks are kept even when the character of a mark gets deleted (the mark is kept as long as the line exists). So why shouldn't a mark be set in virtual space? You might argue that it is a good idea to set the mark to the last existing character within the line, though.

Note: This was inspired by the virtual editing with the editor BRIEF.

Suggested by: Sven Guckes guckes@vim.org [980427] (who really loved this feature of BRIEF) and requested by Vadim Zeitlin zeitlin@seth.lpthe.jussieu.fr [980427]

option backupext -> backupfilename

So far you can only define the extension of a backup file - but you cannot *prepend* any characters, such as the '.' or '#' to make the backup file "hidden" on Unix. Therefore an option to set the "backupfilename" is desired. The option should allow the use of |filename-modifiers|.

Suggested by: Noble Thomas nthomas@eclipse.cise.ufl.edu [980424]

new register '<' - recent input

vile has the register < that holds the recent input. Such a register would be a good thing for vim, too, as you could check what you have typed. This helps both understanding Vim and debugging it.

Suggested by: Sven Guckes guckes@vim.org [980420]

option "title" - "Thanks for flying Vim!"

With ":set title" Vim will change the title of the surrounding window to "Vim - filename" whenever you edit a new file or switch to another buffer. Upon exit Vim will change the title to the message "Thanks for flying Vim!".

Why does Vim do this upon exit? Well, it's a fun feature. So there.

The feature of displaying the current filename in the window title bar *is* nice. However, many people do not want the final message. There should be a way to turn this off.

Suggested by: Gerald Oskoboiny gerald@impressive.net [980417] and many other people

On 980422 Bram agreed to make this an extra option ("titlerestored").

RegExpr - Characters as ASCII numbers

Problem: You cannot describe characters in regexpr sets as ascii numbers. If this was possible then it would be easier to do some things.

Examples: search for 8bit characters /[\0x128-\0x255]

Suggested by: Sven Guckes guckes@vim.org [980304]

search - show match in middle line of window

Problem: When a search command ('/?#*nN') finds the next match within the current window then the cursor is simply moved onto that line. But sometimes it is better to show the matching line always at the same line of the current window, eg at the top, middle, or last line.

Suggestion: Add an option which allows to position the matched line on the current window.

```
value           show matching line at..
H (home)        first  line of window
M (middle)      middle line of window
L (last)        last   line of window
.               current line of cursor
```

Suggested by: Sven Guckes guckes@vim.org and Stan Brown brownsta@concentric.net [980304]

Workaround: use autocomamnd on event "SearchPost":

:au SearchPost * :normal zz

This event is not available yet - but on the todo list. [980915]

:join - adding spaces at end of sentence

A ":join" adds a space after a dot if it ends a line, thus resulting in two spaces after the period on the joined line. [The idea is that the dot that ends a line is probably the period of a sentence. Adding two spaces shall make the text more readable.]

Example:

```
text:   foo.
        bar
command: :j
result: foo.  bar
```

Problem: This works only for the dot character - but not for other characters which *could* end a sentence, such as ":!?".

Suggestion: Add an string option that allows to define the set of characters for which an additional space shall be added with a ":join". This would also allow to turn this behaviour off completely, thus allowing for consitency of a ":join" command.

Suggested by: Sven Guckes guckes@vim.org [970915,980304]

:join - remove quote

Situation: Two consequtive lines with the same quote prefix, cursor on first line.

Problem: The command ":join" does not remove the quote prefix for the second line.

Example: text: > > foo > > bar command :join result > > foo > > bar

Workaround: Select both lines visually and use text formatting with "gq".

Suggestion: Add option 'j' to "formatoptions" to allow/disallow this. Bram said it's a good idea and he'll add a remark to the todo list.

Suggested by: Sven Guckes guckes@vim.org [970915,980304]

N^ vs N$

Repeting the jump '^' does not change the position. However, repeating '$' will move down N-1 lines. This is inconsistent.

Suggestion: Make N$ jump to the end of line and *not* leave the current line. Use N- and N+ to move between lines instead.

Suggested by: Sven Guckes guckes@vim.org [970717,980304]

:ua - :update all

```
*:ua* *:uall*
Write all modified buffers to file.
Unmodified buffers are not written.
{not in Vi}
```

Suggested by: Markus Klein Markus.Klein@usf.uni-osnabrueck.de [980302]

file locking

Vim-5 does not "lock" edited files - yet.

Add option "filelock" which will "lock" edited files. (nvi does this, it seems).

Problems: File locking does not exist for every OS - thus it can be a Unix feature for now only. Also, a locked file

needs to kept "open"; as there is a limit to this number every locked file will reduce the number of files which can still be opened. Tricky!

Suggested by: Several people [980225]

display - show mode with cursor and color

Vim shows the current mode when you ":set showmode". But when you are on the command line then the cursor is on the command line, too, so you cannot see which buffer/window the command will be applied to.

Show the current window by coloring the status bar of the current window or by changing the background color.

For GUI version (inspired by elvis): Show the current mode by changing the shape of the cursor:

```
mode        cursor shape
insert      block
command     vertical bar
replace     underscore
```

Problem: xterm and Linux console cannot change the cursor shape.

Suggested by: Anthony Campbell acampbell@achc.demon.co.uk [980222]

:wedit filename

Write current file and edit filename.

Workaround: ":set autowrite" and ":n filename".

Patch by: Scott Bigham dsb@cs.duke.edu [980224]

URL: http://www.cs.duke.edu/~dsb/vim/wedit.diff

Command History Editing

Add a command to delete an entry from the command line history. This allows to remove an erroneous entry.

Suggested by: John L. Spetz jls11@po.CWRU.Edu [980202]

DOS/Windows - rebound to calling window

Situation: You call Vim from a window ("W"), and after exiting Vim you want "W" to be the active window again.

Suggestion: Make Vim rebound" to window "W" after exit.

Problem: Isn't this something the OS or the window manager should provide as a feature?

Suggested by: Peter Jay Salzman psalzman@landau.ucdavis.edu [971226]

auto-unload unmodified buffers

When memory gets low let Vim unload unmodified buffers automatically. This should be optional, of course.

Suggested by: Ron Aaron v-ronaar@Exchange.Microsoft.com [971215]

function to get current hostname

Problem: Some settings may depend on the host you are using - but there is no function to get that hostname.

Suggested by: Ron Aaron v-ronaar@Exchange.Microsoft.com [971215]

Patch: Paul.Moore@uk.origin-it.com [971216]

angle notation - SP

Let <SP> expand to a "space". This way you have a representation of the space character that is non-space, so you can add it to mappings which should not break with text formatting.

Suggested by: Sven Guckes guckes@math.fu-berlin.de [971215]

Crypting

Vi has the command ":X" to encrypt the current buffer and the startup option "-C" to force all inout files to be regarded as encrypted files. Vim does not have the ":X" command and uses "-C" for something else.

Suggestions: Add command ":X" to encrypt the current buffer. Add a buffer flag "[x]" to show that the contents of the buffer is encrypted.

Problems: The crypt function is not available on all systems. Also, crypt(1) has distribution restrictions. Should Vim therefore use an internal crypt function? Should Vim's crypt be compatible with other vi clones' crypt (eg vile)? Maybe use Crack5?

Suggested by: Ed Ralston eralston@integ.com [971211]

new command "gF" - get Filename under cursor

Fact: The command "gf" will edit the filename under the cursor, but refuses to do so if the file cannot be found. That's fair enough and is actually a "feature".

Problem: There is now way to "get" the filename under the cursor. If there was then you could start editing this file with ":edit ^R"".

Solution: Add command "gF" to "get the current Filename" to the (default) register.

These ideas by Otavio Exel oexel@economatica.com.br:

```
map SaveToFloppy gF:!cp " /mnt/floppy
map ShowDelta    gF:!rcsdiff " | less
```

Otavio suggested "yF" for "yank Filename" - but as 'F' is a valid jump command it cannot be used. Noted by Thomas Koehler jean-luc@picard.franken.de [980206].

Suggested by: Sven Guckes guckes@vim.org [971203,980206]

word completion - complete from other buffers

Add an option that allows word completion using the contents of the other buffers.

Suggested by: Michael Natkin (mjn@sgi.com) [971127]

new visual command - "[=text]"

Add the visual command "[=text]" to evaluate the following text as an expression, and push the resulting text onto the typeahead buffer.

This allows to do delete N lines with the command "[=N]dd" and to insert the value of the register 'a' with "^O[=a]".

You could also use the ?: operator to execute statements conditionally, as in [=modified?":w^M"].

Suggested by: Steve Kirkendall [971114]

ex command aliasing

Allow aliasing of commands, eg

```
:alias :which :!which
```

This example shows how to make use of an external command.

Steve Kirkendall says that elvis-2.1g will have this.

Suggested by: Sven Guckes guckes@vim.org [971113]

add utility "bed"

Add the small utility "bed" as an additional "binary editor" to the distribution of vim. See the page about bed.

Suggested by: Sven Guckes guckes@vim.org [971111]

text formatting - right *and* left aligning text

Add command ":pad" to align text between two columns.

```
:[range]pad [column_left] [column_right]
        Align the text in [range] between columns [column_left]
        and [column_right].  Fill with spaces where appropriate.
```

Emacs calls it "text region fill". However, "fill" with *roff commands means "put as much text onto each line as possible". That's why "pad" seems much better.

Suggested by: Ilya Beloozerov ibelooze@runet.edu [971104,971114]

Apply "ignorecase" to command ":il"

Ignore the case for matches done with ":il" when "ignorecase" is set.

Suggested by: Don Rudolph dcr@netcom.com [971104]

Support for MS "mouse wheel"

Add support for the MS "mouse wheel" to steer around.

Suggested by: Peter Jay Salzman psalzman@landau.ucdavis.edu [971022]

Allow ASCII values for letters in commands and mappings

Allow "\xHH" to describe a characters by its hex value. Then you can use the set of "high bit characters" with "[\x80-\xFF]".

Using "\nnn" to give the decimal number of a character is not a good idea as it overloads the group identifiers "\n" in regular expressions. The form "<NNN>" is not Vi compatible inside a range, but could be yet another "Vi improvement".

Examples: syn match Special "[<128>-<255>]\+"

Suggested by: Sven Guckes guckes@vim.org [971021]

Support for SUN Keyboards Function Keys

Add support for SUN keyboards function keys (Cut/Copy/Paste/Undo/Find) and add key names (eg "<SunCopy>"), of course with default mappings: :vmap <SunCopy> "<clipboard>y :vmap <SunPaste> "<clipboard>p :vmap <SunCut> "<clipboard>x

Suggested by: James Ko jko@glenvan.glenayre.com [971017]

X support for CUT_BUFFER and CLIPBOARD

"Full support for X Windows CUT_BUFFER[0-9] and CLIPBOARD by means of say registers and/or naming for keyboard mappings. As far as I know, current text selection only goes to the primary CUT_BUFFER. Some programs, especially those 'tools' in Sun OpenWindows environment, use the CLIPBOARD, so cutting and pasting between them is difficult."

Suggested by: James Ko jko@glenvan.glenayre.com [971017]

directory history with stack and select command

Keep a stack of directories, add a command to view the list and add commands to select the previously used directory as well an arbitrary entry from the list.

Suggested by: Raoul Beauduin beauduin@ori.u-tokyo.ac.jp [971017]

configuration - option "autosave"

```
                                    *'autosave'* *'as'* *'noautosave'* *'noas'*
    'autosave' 'aw' number  (default 0)
                           local
            Automatically write the current buffer to file
            after N seconds after the last change has been made
            and when |modified| is still set.
            Default: 0 = do not autosave the buffer.
```

Example: `:set autosave=600` makes Vim save the current buffer after 300 seconds (5 minutes).

Suggested by: Raoul Beauduin beauduin@ori.u-tokyo.ac.jp [971015]

command line expansion like tcsh's "autolist"

"This is sort of a combination between CTRL-L and CTRL-D in vim. First it completes as far as it can (like CTRL-L), then if that does not find a unique match it lists all possible matches (like CTRL-D). If I map TAB to C-LC-D, vim will list the contents of a directory if it's a unique match (pretty annoying). I think I figured out how to do this a long time ago, but I lost my .vimrc."

Suggested by: Bruce A. Templeton brucet@bigfoot.engr.sgi.com [971011]

display status bar - current C function

For C source files - display the name of the current C function.

Steve Kirkendall: "I added something like this to elvis 2.1f-alpha. It is driven off the "tags" file. This has the benefit that it inherently supports any language that ctags supports. However, when loading a large source file the editor may need to perform hundreds of tag searches, which can be slow. // Once loaded, however, it is very fast. Elvis stores a list of the found tags' positions, so it just needs to scan through the list for the nearest tag which is defined at-or-above the cursor position, and display its name."

Suggested by: Bruce A. Templeton brucet@bigfoot.engr.sgi.com [971011]

tetris window

Add command ":tetris" that opens a window (to full size) and allows to have a game of tetris. Highscore table would be a plus. Configurable at compile time, too.

This is just to show that it *is* possible. ;-)

Suggested by: Sven Guckes guckes@vim.org [971010]

swapfile - follow symlinks and create swapfile for target only

When editing a file which is a symlink, Vim creates a swapfile with the name of the symlink. This creates a problem as the original file (target of the symlink) may already have a swapfile from a previous editing session. This especially creates a problem with multiple symlinks.

The suggested solution is: Always check whether the filename is a symlink, and if so, follow the link(s) to the target; then create the swapfile using the name of the target file.

Suggested by: Sven Guckes guckes@vim.org [971010]

tag search - add movement for "/pattern/+N"

Add "search offset" for tags.

Suggested by: Eli the Bearded eli@NetUSA.Net [971007]

vi compatibility - backspacing does not remove characters on screen

Add option to 'cpoptions' (suggested: 'H') that will not remove characters from the screen (window) when backspacing (deleting) them. Some people really love this feature of Vi - especially on slow terminals.

Suggested by: Matthias Buelow mkb@altair.mayn.de [971007]

vi compatibility - wildchar expansion

Add an option to 'cpoptions' (suggested: 'W') to allow expansion of wildchars within arguments.

```
W   When a wildcard is specified on the command line,
    and no files are found, act as if that file specification
    was not even present.  vi default is to edit a file
    with the wildcard character or characters in its name.
```

Suggested by: David Douthitt DDOUTHITT@cuna.com [971002]

insert mode - special mode message for language ports

The language ports would benefit from special mode messages to indicate input of other languages (and use of special fonts).

Examples: "INSERT Farsi/Persian", "INSERT Russian".

Suggested by: Alex Postnikov apost@math.mit.edu [971002]

recovering with "vim -r" - give info on ":help recover.txt"

...

documentation - :help whitespace

The term "whitespace" is common - but should be defined for Vim, anyway. A good description with tags to affected options is desirable.

Note: ":help I" says: "Insert text before the first non-blank in the line [count] times." Shouldn't this be "first non-whitespace"? After all, the command 'I' jumps over tabs, too!

Suggested by: Sven Guckes guckes@vim.org [970926]

documentation tags - dash vs underscore

Tag names use both dashes and underscores. "Older" tags still have underscores - but newer ones should use ddashes only. So the underscores should be replaced with dashes. This should definitely be done for Vim5!

Suggested by: Sven Guckes guckes@vim.org [970918]

visual-operators - make 'C' change from *current* column

Operators 'C' and 'S' work the same. But operator 'C' should just change the text fom the *current* column.

Suggested by: Sven Guckes guckes@vim.org [970918]

documentation - add future changes with "todo"

Instead of keeping lots of plans in the todo file, add the description of features with a not saying "todo". Thus readers can get accustomed to upcoming changes and help implementing them.

Suggested by: Sven Guckes guckes@vim.org [970918]

todo list - make it readable!

The todo list is very hard to read and understand. Someone please take the task of cleaning it up!

Suggested by: Sven Guckes guckes@vim.org [970918]

[DONE] insert mode - inserting variable values with C-R

Extend command i_CTRL-R to read in values of variables.

Suggested behaviour: Accept the dollar sign as a token and then read in the name of a variable of environment variable (preceded by yet another dollar sign). Thus typing: ":let a=1+1" followed by "$a" would insert the contents of variable 'a' into the text. In this case it would be the string "2". And typing "$$VIM" would insert the VIM env var contents and "$'tags'" would insert the current value of the tag path.

Suggested by: Jim Kleckner kleckner@cats.com

This already exists: i_= ,ie in insert mode type CTRL-R followed by '=', then enter in a variable name or expression - and Vim will insert the value. [970916,970918]

drag&drop - drop folder -> :cd to folder

Dropping a folder onto Vim will make vim open a file with the folders name, which isn't very useful. Instead, let Vim ":cd" to that folder (optionally).

Suggested by: Robert Coutts (rcoutts@mediaone.net) [970912]

variable "textmode" - Macinstosh end-of-line (CR)

Make "textmode" accept a third value to allow writing files with end-of-line style used on Macinstosh computers, ie use character "CR" to denote the end-of-line. Perhaphs even change the name to "endofline" and allow values of "dos", "mac", and "unix".

Suggested by: David Douthitt ddouthitt@usa.net [970910] and many other Mac users (eg yours truly).

search commands ('/' and '?') - add flag to ignore case

Add a flag (search offset) to the search commands to allow matches for the search pattern ignoring the case of "letters" (a-z).

Suggested flag: 'i' ("ignore case")

See ":help search_offset"

Suggested by: Huy Le huyle@pride.ugcs.caltech.edu [970909] and many others.

substitution - add flag to ignore case

Add a flag to the substitution command to allow matches for the search pattern ignoring the case of "letters" (a-z).

Suggested flag: 'i' ("ignore case")

See ":help s_flags"

Suggested by: Huy Le huyle@pride.ugcs.caltech.edu [970909] and many others.

add startup switches "--help" and "--version"

"--version": Display version number and copyright information, eg:

```
vim-5.0 [released 980219] [compile 980303]
Copyright (c) 1998, Bram Moolenaar bram@vim.org
For more info see the internal command ":version"
and the webpages at http://www.vim.org/.
Report bugs to to the author or to vim@vim.org (subscription required).
```

Then you can easily include it in your mail with ":r!vim --version". While there is copy&paste on many systems, you sometimes need it as such - especially if you are using DOS.

Btw, the "GNU coding guidelines" ask for "--version", too!

Suggested by: Sven Guckes guckes@vim.org [970818,980304]

dictionary completion - partial completion

The dictionary completion with ^X^K does not allow to complete unto a common end of the current word.

Example: Partial completion of "ab" with possible expansions "abcdeF" and "abcdEF" should expand unto "abcd".

Suggested by: Stefan A. Deutscher stefand@ibm.net

registers - additional register for search command

There are several read-only registers - one for the last inserted text, current file name, current alternate file, and last command line. However, there is none for the last search command. We need one!

Suggested name: "/

Suggested by: Sung-Hyun Nam namsh@nuguna.rms.lgic.co.kr, Pablo Ariel Kohan pablo@memco.co.il [970716]

configuration - errorbells

Problem: Even a ":set noerrorbells" does not turn off all bells with (error) messages. This can be quite annoying.

Solution: Add an option to turn on/off bells for all messages.

Suggested by: Sven Guckes guckes@vim.org [970714]

installation - DOS and docfiles

Problem: When the variable VIM (for command.com) is not set then VIM cannot find its doc files. This requires the user to read the documentation about installation before he can use VIM with doc files.

Solution: Let VIM look for the doc files relative to the directory that contains the executable if the variable VIM (for command.com) is not set. Then the user can use the doc files right away.

Suggested by: Brendan Macmillan bren@molly.cs.monash.edu.au [970714]

[workaround] display - show list of buffers in extra window

Many users would like to see the list of all their open (unhidden) buffers all the time on the screen - an option for that is very much desired.

Workaround: ":set cmdheight=N" - this increases the lines of the command line to N lines. So when you do a @:ls" you will see the result until the next update of the command line. [970623,971126]

n_% - bracket matching - showmatchtime

The variable "showmatch" controls whether a matching bracket shall be shown (if possible on the current window). But there is no control over the time that the match shall be shown. nvi has a variable for this, though: "matchtime" - number of 10th seconds to show the match.

Suggested variable name for Vim: showmatchtime (smt)

Suggested by: Dave Davis d_davis@wfire59.eng.pko.dec.com [970619]

Display GUI - show user@host:/path/file

Allow to display the username and host in the title of xterms. Preferably configurable for each part.

titleformat %h hostname %u username %p pathname %f filename default value: %f example with everything: titleformat=%u@%h:%p%f example with fancy stuff: titleformat=You are now editing the file %p%f.

Suggested by: Zdenek Sekera zdenek@strad.gland.sgi.com [970502]

Display - show buffer number in status line

Show the buffer number in the status line. This should make it easier to switch to a buffer as you can take the buffer number from the display for using commands such as **:Nsb**.

Suggested by: Sven Guckes guckes@vim.org [970720]

option "ruler"

Change the option "ruler" from "toggle" to a "flag list":

```
flag    word    behaviour
A       All     show all info that can be given with this option

                cursor position:
c       column  show the column    number of the cursor
l       line    show the line      number of the cursor

wc      char    show the character number of the cursor
wl      line    show the line      number of the cursor
C       column  show the column    number of the cursor within current window
w       screen  show the byte      number of the cursor

                window position:
H       high    show number of first  line of current window
L       low     show number of last   line of current window
M       middle  show number of middle line of current window
```

```
G       last    show number of last   line of current buffer

P       percent show the percentage of the current line
R       Range   show the percentage range of the current window
                ie the percentage of the window's first and last line
r       ratio   show the ratio of the current window's line range towards
                the whole buffer  ("how much of the complete thing do I see?")

NOTE:   The flags for lines of the current window and for the last line
        were chosen to match Vi's default commands to jump to these lines.
The default format is "%l,%c-%C [H=%H,M=%ML=%L,G=%G] (%r,%p)".
Unset flags will simply leave out the information and surrounding brackets.

Example:  For "set ruler=lcCGHLr" the ruler display might show
        "50,9-65 [40,60;100] (%20)" which means that the cursor in line 50
        on the 9th character displayed on column
```

Optionally show displayed line range and percentages, eg for a buffer with 100 lines: "[1-24,%:1-24]".

Suggested by: Sven Guckes guckes@vim.org [960512]

Check all startup files (.exrc, .vimrc, _exrc, _vimrc) on all platforms

Quoting Bram Moolenaar: "I think it is VERY important to have a quick startup for most people. Working over a network can be slow, we should not try to open files in many places. -- For version 5.0 it would be very good to have a configurable startup script. But Vim has to find it first! This would come down to first do the initializations like vi (to remain compatible) and then process some Vim specific file(s) (to remain compatible with previous versions of Vim). -- If you are working on Unix, and want to read a _vimrc anyway, you could then put something like this in your .vimrc:

```
if exists("_vimrc")
        source _vimrc
fi
```

Syntax for the script is to be decided upon later (please don't start a discussion about this right now!)."

Definition of start/end strings for start/end matching.

Should be very useful for matching begin/end strings, eg comments, function definitions, latex or html commands, and all bracket pairs, ie `'(){}[]<>.

Suggested by: vimdev people [960430]

Command history for normal mode commands.

Add a special read-only buffer to show the history of normal mode commands, one complete command one separate lines. This would allow copying of commands into registers easily.

Suggested by: Sven Guckes guckes@vim.org [960430]

ex commands on registers

Allow ex commands to work on registers. Example: `:"as/foo/bar/g` works on buffer 'a' searching for "foo" and replacing it with "bar" globally.

Suggested by: Alain Sarraf (sarraf@bnr.ca) [960317]

Statistics on visual selection

Add command to visual mode to show statisctics on selected text. The statistics show info on lines, characters, words, width, and paragraphs.
Example:

```
long  info:   "-- VISUAL -- lines:3,7[4] chars:365 words:90 width:79 pars:2
short info:   "-- VISUAL -- 3,7[4];365c;80wo;79wi;2pa
```

This means that I have selected text within the line range "3 to 7" with "4" lines, "365" characters, 80 words, biggest width of "79" characters, and with "2" paragraphs. Suggested key: "I" ("info").

Suggested by: Sven Guckes guckes@vim.org [960315]

no swap file on opening of file

(optional)

When option is set, do not (attempt to) create a swapfile until the first change is made. When a swapfile exists, do not give a warning until the swapfile is attempted to be created.

Suggested by: Bruce A. Templeton brucet@engr.sgi.com [960315]

Show free keys

Show list of "free" (unmapped) keys. Suggested command name: ":freekeys".

Suggested by: Sven Guckes guckes@vim.org [960101]
Bram Moolenaar: Difficult to implement and keep up-todate.

Visual Mode - operator "filter"

Change: visual mode
Fact: The visual commands "filter" (!) and "write" (:w) work on full lines only.
Change the commands so they will work not only on full lines but also on partially selected lines.

Suggested by: Sven Guckes guckes@vim.org [960221]

Automatic expansion of words in insert mode

Fact: You can expand the word before the cursor but it is not done automatically.
Suggestion: Automatic expansion of words.

```
wordexpandmin   number (default 0)
        Expand words while typing.
        value=0: No expansion.
        Suggested value: At least 4.
        Words with at least "wordexpandmin" letters will automatically be
        expanded while writing.
        The expansion is taken from the previous text of the current buffer.
```

Suggested by: Sven Guckes guckes@vim.org [960208]

Show free keys

Show list of "free" (unmapped) keys. Suggested command name: ":freekeys".

Suggested by: Sven Guckes guckes@vim.org [960101]
Bram Moolenaar: Difficult to implement and keep up-todate.

Change: (:dig) - delete default table of digraphs.

Add a way to delete the default table of digraphs. Some people only want to use a part of it - and searching through a big table usually takes a lot longer than searching a small table. [960128]

Change: ^a and ^x - increment/decrement letters

Change letters, too, by changing the ASCII value. [960123]

Change: (:undo) and (:redo) - add messages

When no more undo or redo is possible then a beep is given. Add option to also show a message in these cases.

Suggested by: Sven Guckes guckes@vim.org [960123]

Change: (:edit) - expansion of # and %

Expand special names % and # when the expansion key is typed

Workaround: Enter 'CTRL-R' + '%' and 'CTRL-R'+'#' respectively.

Suggested by: Sven Guckes guckes@vim.org [960123]

Yank stats

Change: (y) - yank

Tell the user how much data (lines, characters) have been yanked and whether the data was yanked to a buffer.
Example: Yanked (copied) 1234 characters from lines 10-31 to buffer x.

Suggested by: Sven Guckes guckes@vim.org [950901]

Yank ranges into buffer on command line

Change: (y) - yank

Add yanking of line ranges on the command line: ":1,3"iy" will yank lines 1 to 3 into buffer "i".

Suggested by: Sven Guckes guckes@vim.org [950901]

Show map list sorted by key names

NEW command ":showmaps": Shows a list of all letters together with meta keys (shift, control) and their bindings. Example:

```
Mode    metakey key     function
normal  -       a       append after character
normal  shift   a       append after end-of-line
normal  control a       -
normal  meta    a       ???
insert  control a       insert previous inserted text
...
```

Suggested by: Sven Guckes guckes@vim.org [950727]

option to set width of "line numbers"

":set number" shows the line numbers in front of each line. This shows the line numbers to the right of seven characters. As most texts have less than 1000 lines this seems a waste of space. Bram once said he might add an option to set that width. "On low priority, though."

Suggested by: Bram Moolenaar bram@vim.org [970429]

Screen jumps with 'z' and 'Z'

Suggested by: Sven Guckes guckes@vim.org [970306]

```
                        Screen Jumps

        Screen Jump commands change the current column or line of
        the current character on the screen within the current window.
        This allows for both horizontal and a vertical movement of the text.

        The vertical movement changes the position of the current line;
        the current column is either kept (set option) or changed to the column
        of the first (non-blank) character (set nooption).

        Table of vertical screen jumps          *table_screen_jumps_vertical*
        (Changes position of current line within current window.)
        Type 'z' first - then type letter in table:

                new style               old style

        column: first   current         first   current
        line    =====   =======         =====   =======
        Home    H       h               'CR'    t
        Middle  M       m               .       z
        Last    L       l               -       b

        Table of horizontal screen jumps        *table_screen_jumps_horizontal*
        (Changes position of current column within current window.)
        Type 'Z' first - then type letter in table:

        leftmost  left   lefttab   middle   righttab  right rightmost
        H         h      ^D <      m M c C  ^T >      l r   L
```

So we'd use 'h' and 'l' for the usual left/right movement (with an optional 'r' for "right"). 'H' and 'L' work in analogy to 'h' and 'l', ie the go to the boundary of the current window. And 'M' does the same for the "middle". As there is only one middle we can use 'm', too. Same holds for 'C' and 'c'

```
for those who prefer to think of "center".
And there is a jump by tabstop with ^D and ^T -
using '<' and '>' as easy alternatives.
"Z0" and "Z$" can be interpreted as '0' and '$'.
```

---

file-wide-search

find next/previous occurrence of last search string in next/previous file.

Change to status line

Problem: The status line does not always show what people prefer.

Suggested change: Make it possible to define the status line via a printf function. Suggested by: Bram Moolenaar

Suggested values for status line: pwd, time, mail, hint to ":help", buffer size, ruler. Add extra help lines for newbies to show more info, eg basic movement and editing commands, current filename.

Suggested by: Sven Guckes guckes@vim.org

cursor jump list

Problem: Typos can also lead to unanted cursor movements. You cannot undo a move unless it was a "jump", however.

Suggested change: Add command to undo *all* cursor movements.

Decision: Won't be done as it is too difficult to implement. The commands CTRL-I and CTRL-O should suffice.

gvim cursor

Give the cursor different colours for different modes.

Change: (I) - insert

Add option "smartindent" (alphanumeric) to change the jump and mode involved with "I":

```
Default: smartindent=0i
0:      Jump to first non-whitespace character of current line.
1,2,3:  Jump to first (1) space (2) tab (3) letter of current line.
        and switch to (1) insert mode (2) append mode (3) . (default)
a,i,o,r:  Switch to (a) append (i) insert (o) open (r) replace mode.
The last cipher or letter "wins".
```

Add option "insert" (numeric) to enable use of the "insert_key" to cycle through the five "insert modes", ie "command", "append", "insert", "open", "replace":

```
Default: insert=0
0:      No effect.
1:      Enable "append".
2:      Enable "insert".
4:      Enable "open".
8:      Enable "replace".
Example: insert=10 enables "2" and "8", ie pressing the "insert_key"
repeatedly will cycle through mode "command" -> "insert" -> "replace" ->
"command".
```

NEW: :set noexpandbeep [Change: (tab) - filename completion]

Fact: When using the expandkey on a filename then you get a beep if the completions is ambiguous.
Add option to suppress beep and/or show *number* of possible completions.

Suggested by: Sven Guckes guckes@vim.org

NEW: Add all numbers in visual

{visual}

Change: visual mode
Add command to add all numbers in visual. Would be *very* useful to add numbers in columns.

Suggested by: Sven Guckes guckes@vim.org

Change: (K) - Keyword call

```
keywordprg (kp)         string  (default "ref")
  Program to use for the "K" command. Environment variables are expanded.
  {not in Vi}
```

Suggested change: Do not make a shell call if "kp" is not set (ie empty). [TODO: This seem to have been implemented. Check again! 960701]

NEW option: "repaintfromtop"

Will stop the scrolling for commands ^d and ^u by repainting the screen from top.

NEW: gdb support (as with emacs)

[left to an emacs fan for description ;-]

NEW: scrolling regions for popup lists [960130]

...

NEW: keyword (syntax) highlighting and colouring

one of *the* most wanted wishes

code structure

Restrict functions to, say, 50 lines. This makes reading and checking of the code a lot easier.

External solutions

The following stuff can be done with external programs using the filter command and thus probably will not be implemented.

Making commands work on the current paragraph

Submitted by Thomas Köhler on 990702: :com! -nargs=* -complete=command Par call Pardo("<args>") :fun! Pardo(command) : let col=col(".") : let line=line(".") : norm { : let a=line(".") : norm } : let b=line(".") : exe ":" . a . "," . b . a:command : exe line : exe "norm " . col . "\|" :endfun Now, what does this do? :com! -nargs=* -complete=command Par call Pardo("") This defines a new command "Par" that completes ex-commands. It calls a function "Pardo()" with the rest of its commandline as one argument.

What does Pardo() do? It applies its argument (should be an ex-command) to the current paragraph. It works like this: :fun! Pardo(command) " this remembers current cursor position : let col=col(".") : let line=line(".") " jump to the beginning of the paragraph and remember that cursor position : norm { : let a=line(".") " jump to the end of the paragraph and remember that cursor position : norm } : let b=line(".") " apply the command given as an argument on the range a,b : exe ":" . a . "," . b . a:command " jump back to the position we were before (this may be confusing if " " ex-command is something like 'd' ;-) " : exe line " : exe "norm " . col . "\|" " :endfun

normal/visual command '~' (aka "switch")

options trin and trout

Make the "switch" command '~' switch character and the characters in visual text according to the options "trin" ("translate input") and "trout" ("translate output").

Yes, I know that you can use the command "tr" on Unix externally - but then not everybody is using Unix, right? Even using an external script for this (I once used a sed script) needs to be set up correctly as well. It would definitely be nice to have this command internal to vim as this would do away with extra installations. Besides, it is easy to code, too. I wish I was able to do this - then it would already be done.

Anyway, support for "sets" would be nice, as this makes it much easier to denote them.

Example: UPPER/lower

```
set  trin="A-Z"
set trout="a-z"
```

This is just to show that the usual functionality of the switch command is just a special case.

Example: Pairs

```
set  trin="A-Z a-z < {} [] () \/ `' ,. :; +-"
set trout="a-z A-Z > }{ ][ )( /\ '` ., ;: -+"
```

This not only switches letters - but also brackets pairs and other corresponding characters (slash/backslash, tick/backtick, dot/comma, colon/semicolon, plus/minus). This allows an easy fix of frequent typos for these characters.

Note: The spaces within the values of "trin" and "trout" of the previous example are just for readability.

Suggested by: Aaron Schuman aaron_schuman@yahoo.com [980427]

Example: "Caesar Cipher" Code

```
set  trin="a-zA-Z"
set trout="d-za-cD-ZA-C"
```

This shifts all letters three positions forward within the alphabet. Of course, this asks for a setup to decode it, too:

```
set  trin="a-zA-Z"
set trout="x-za-wX-ZA-W"
```

Although switching these values is just a matter of a simple mapping, it is much nicer to use a code which is invers to itself, so you need not change any setting. Such a code is "rot13":

Example: ROT13

```
set  trin="a-zA-Z"
set trout="n-za-mN-ZA-M"
```

This code shifts letters by 13 characters, ie half the length of the alphabet. That way, it is invers to itself, so that using it twice on the same text reveals the original. This "rotation" is usually abbreviated as "rot13". This code is used on Usenet quite often to hide the text from casual reading, and thus often used to hide "spoilers" to puzzles or for hiding "offesive" text such as in posts to rec.humor.tasteles. ;-)

Suggested by: Sven Guckes guckes@vim.org [950101,970919,980430,980504]

Evaluate math strings

Change: visual mode
Add command to visual mode to evaluate "math strings" for integers, eg "(- 123 + 789 + 456) / 2 * 3 + 2^5 + 3**4" should give "1496".

VIM Wishlist - NEW

I am currently trying to reorganize the wishlist to related wishes.

Binary Mode | Color | Commands | Display | Documentation | GUI | Options | Register Names | Misc

VIM Wishlist - Binary Mode

jump by bytes and to byte offset

Add command to jump by N bytes and to the Nth byte in the file.

Suggested commands: "gb" to jump to byte N, and "gB" to jump by N bytes.

Suggested by: Curt Wohlgemuth curtw@cup.hp.com and Sven Guckes guckes@vim.org [990107]

VIM Wishlist - Color

Color after end-of-line

Show color for NonText after existing lines. Optionally allow the default color to change after 'textwidth' or column N. Very useful to make the textwidth visible, especially for texts where the textwidth matters (eg messages on Email and Usenet).

Suggested by: Sven Guckes guckes@vim.org [970406] and Dawid Kuroczko dk@knm.org.pl

VIM Wishlist - Commands

change text object commands

Purpose: Work on text objects with and *without* surrounding whitespace.

Problem: Sometimes you do not need to copy/cut/delete text objects with the surrounding whitespace.

Suggested Solution: Add text objects which do *not* include surrounding whitespace. Use these operators:

```
                  whitespace
    a       all     prefix  suffix
    i       inner   none    none
    p       prefix  prefix  none
    s       suffix  none    suffix

    text objects
    p       paragraph
    P       paragraph delimited by blank lines
    s       sentence
    S       sentence - delimiters can be defined
```

Suggested by: Sven Guckes guckes@vim.org [960521,980304]

new command

:history

Fact: You can see previous commands on the command line using the command ^P.

Problem: There is no command to show a list of the last N commands. Suggested Solution: Add command ":history" to show the list of the last N command line commands.

Suggested by: Sven Guckes guckes@vim.org [960615]

change command ^W

^WN - jump to window N

Purpose: Access a window directly.

Suggested Solution: Give the first ten windows a digit as number. Allow to directly switch to them by the window command followed by the digit, ie "^W3" to jump to window #3.

```
| ... ...
|[2] ~/foo
| ... ...
|[3] ~/bar
| ... ...
|:command line
```

Suggested by: Sven Guckes guckes@vim.org [960622]

change command i_CTRL-R

^R"X

Fact: You cannot recall the contents from a buffer when the name is preceded by a "".

Purpose: Put contents from buffer **X** into text in replace mode

Suggested Solution: This will replace text with the contents from a buffer as if you typed it in when using 'R' to replace text.

Suggested by: Leslie Gerstenfeld lgerst1@umbc.edu [960718]

VIM Wishlist - Display

display (nocolor)

Show current editing line in invers mode

Purpose: The current editing line is sometimes hard to make out.

Suggested Solution: Show the current line in invers mode.

This helps a lot while typing.

Setting a colour for the "editing line" would be nice, too.

Suggested by: Laurent Duperval laurent@Grafnetix.COM [960315]

display (nocolor)

extra line for messages

Leave an extra line for (error) messages. An extra line for (error) messages is good as you can still see them when typing in a new command.

Further uses:

Startup (error) messages: Whenever you change a setup file you are likely to get an error. Then you want to see the filename, line number, and its contents.

Online help: Great for showing extended help messages, too. You could give some feedback on executed commands to the (new) user.

Quite mode: Also, this "channel" could be turned off when executing command scripts which sometimes stop the execution (annoying).

Sven Guckes guckes@vim.org [971126,990204]

display (color) [970327]

show *all* matches of search within current window

Show first N or all matches of search pattern within current or all windows. Use color to show the *current* match.

Inspired by pager "less".

display (color)

background colors for windows

Allow definition of background color for each window.

Allow definition of background color for each window when the associated buffer has been changed.

Suggested by: Paul Moore gustav@morpheus.demon.co.uk [961203]

display (color?)

clock

Show the current time on the command line. However, this clock could get in the way of commands and messages, and while macros are executing, and thus could prove *very* difficult to implement. Any takers?

Suggested by: many people [971126]

Workaround: Use an external "date" command to quickly display the current time/date. Or use the internal function "strftime" for that (lots better!). Example: iab YDT <C-R>=strftime("%y%m%d %T")<CR> map ## :echo strftime("%y%m%d %T")<CR>

display (nocolor)

show "command" for command mode

Fact: There is no string indicating command mode.

Suggestion: When in command mode show the word "Command" on the status line, and an asterisk preceding it if the file has been modified (ie: "*Command").

Suggested by: Andy Kahn kahn@cs.ucla.edu [970111]

VIM Wishlist - GUI

Box menus

All menus so far a "list menus" ie each item is in one row of its own. For long lists this results in long menus. However, if the names of the items are short (such as a list of symbols) then a "box menu" with rows *and* columns would fit much better onto the screen. An appropriate command to define the size of a box menu in rows and columns is therefore desired.

Suggested command: :boxmenu rows,columns

Suggested by: Hermann Rochholz Hermann.Rochholz@faidor.de and Sven Guckes guckes@vim.org [970624]

gui: use of the cut buffer

Having to hold SHIFT down to put the cut text into the X cut buffer is a pain -- I'd rather that selected text is put into vim's cut buffer as well as the X cut buffer. Vim modifies the users expectation of how typical xterm apps behave. One more thing to have to remember. (I've used vim's SHIFT-select for a while now and about 30% of the time I forget to hold SHIFT when I want to select text into X's cut buffer.)

Suggested by: Jim Battle jb@XENON.chromatic.com [960614]

gui: make selection analogous to visual commands

It would be nice if vim selected character, line, or blocks with the mouse if (leftmouse), SHIFT-(leftmouse) and CONTROL-(leftmouse), analagous to the keyboard method. I know that this contradicts my previous paragraph, since CONTROL- already means something to an xterm window.

Suggested by: Jim Battle jb@XENON.chromatic.com [960614]

VIM Wishlist - Documentation

missing tags

Add these tags: atom pattern.txt /*atom* charclass pattern.txt /*charclass*

documentation - tag "motions"

Use "{motions}" instead of "{move around}" and give it a tag with an explanation, eg "any number of 'motion' commands".

Suggested by: Sven Guckes guckes@vim.org [970916]

tag names

tag names with multiple words are connected inconsistently with either a dash or with an underscore.

Example: Cycling with ":help comm" gives you these:

tag_commands, undo_commands, format_comment, search-commands, missing_command, expression-commands, autocommand-event, quotecommandquot, autocommand-pattern.

:help hidden

The text says:

"When off the current buffer is unloaded when it is abandoned."

Neither "unload" nor "abandon" can be understood by a non-programmer, as they are not defined anywhere in the docs.

:help replace_mode

Mention replace mode commands ^R and arrow keys (for movement). Add tags for replace mode, ie tag names with prefix "r_".

:help operator [960505]

Add documentation for dot operator:

```
Dot operator after key command, ie command sequences "(key).".
        Repeat last command of same operator.
```

```
E.g. "c." will repeat last change, also when "x" used since then.
[Robert Webb]
```

"{not in Vi}" vs "{Vi: no xxx}"

The documentation mentions "{not in Vi}" and "{Vi: no recording}". The second kind is better because it allow grepping for "{Vi:"; the text after it should tell you about the kind of thing that is or is not in Vi.

VIM Wishlist - Options

new option

"numberbase"

Problem: Number operations are done in base 8 if number starts with '0'.

Purpose: Allow to define the default base for operations on numbers.

Affects: CTRL-A and CTRL-X

Suggested Solution: Add option "numberbase" to specify the default base to be used with number increase/decrease (^a and ^x). This does away with the problem of number with a zero prefix.

Suggested by: Mun Johl mj@core.rose.hp.com [960320]

new option

"name?"

Purpose: Inform the user that he went through the whole file. so he will know that he does not need to search any further. ;-)

Suggested Solution: Give a warning message when a search for the next or previous search goes over the line in which he issued the search for the first time. Maybe require that the search wrapped over the first or last line.

Suggested by: Sven Guckes guckes@vim.org [960430]

additional formatoptions

affects format command ('Q')

Problem: There are some changes that you frequently apply on reformatted text, such a "squeezing whitespace" and "adding end-of-sentence double spaces". They can be done with scripts and macros, but an internal solution would be easy requiring a single parse only.

Suggested Solution: Add these options to "formatoptions":

```
s       "squeeze".  This squeezes runs of whitespace to a single space.
e       "end-of-sentence double spaces".  Use two spaces after end-of-sentence
         punctuation, ie words which end with any of the characters of [.:!?].
```

What this is good for: "squeeze" does away with space runs which are used by text formatters that add whitespace for block text. And "end-of-sentence double spaces" make it easier to see the end of a sentence.

Suggested by: Sven Guckes guckes@vim.org [960521]

new option

"matchoffset"

Purpose: Define minimal "context" for search match.

Problem: You cannot define a line offset for the line displayed for a match from the top and botom row of the current window.

Suggested Solution: Add option "matchoffset" to define line offsets for the display of a match. This is to keep a minimum offset from the top and bottom of the current window.

Example:

```
matchoffset=4,3  (keep four lines from top and three lines from bottom)
_________________ window border
1: top line of current window
2:
3:
```

```
4:
5: matching line
6:
7:
8:
9: bottom line of current window
================= window status line
```

Suggested by: Bill Luebkert bill@westworld.com [960603]

new option

"quitdialog"

Purpose: exit confirmation dialog

Fact: When you quit the last (the only) window with ":quit" then VIM prompts you with "No write since last change (use ! to override)".

Problem: You have to issue the command ":q!" yourself or write the buffer first. There is no dialogue for the user.

Suggested Solution: Add option "quitdialog" that will make VIM give a dialog to allow the choice of ":write", ":quit!" and "cancel" (ie return to editing). This is useful when you would like to keep VIM running. And it would be nice on newbies, too.

Suggested by: Meir Shahar meir@missinglink.co.il [960604,970404]

Optionally: Give "quit" prompt to user when using ESC or interrupt in normal mode.

Suggested by: John H Meyers jhmeyers@miu.edu [960513]

new options

"remarkformat" and "warningformat"

Problem: You cannot customize the formats for remarks and warnings of compilers.

Suggested Solution: Add variables "remarkformat" and "warningformat" to enable recognition of remarks and warnings that compilers produce. Also add commands to jump forward/backward between remarks and warnings.

Suggested by: Gabriel Zachmann zach@igd.fhg.de

new option

"screen_update_macro"

Fact: During excution of a macro the screen is updated.

Problem: The screen updates can slow down the excution of the macro.

Suggestion: With "noscreen_update_macro" all screen updates during the excution of macros are suppressed. This could speed up the excution of macros considerably.

Idea: This idea is from the vi clone "elvis".

Suggested by: Sumner Hayes sumner+@cmu.edu [960629]

VIM Wishlist - Register Names

new register name

register for "last search string"

Add a register name that contains a copy of the last used search string.

Suggested by: dkotchan@inforamp.net (David Kotchan) [961030]

new register name: '#'

Fact: There is the register '%' which contains the filename of the current buffer.

Problem: But there is no register name for the previous buffer.

Suggested Solution: Allow '#' as a register name to be used in these modes: command line mode, i_CTRL-R

Suggested by: Raul Segura Acevedo raul@hplara.iquimica.unam.mx [960627]

VIM Wishlist - Misc

angle notation in output

Fact: You can use the angle notation in the input (vimrc or command line) but the output (":map") still shows characters in the caret notation (^M for <C-M>).

Problem: It would be good if there was an option that reverses this, ie showing special characters within the mappings and abbreviations in the <> notation. Copyinh a map/ab from screen then would be ready for pasting into the command line/vimrc of another vim. Especially useful for making maps and abs available via Usenet or WWW.

Suggested by: Sven Guckes guckes@vim.org [960618]

command line parsing with definable meta character

A command that uses exactly one meta character to interpret a line. The meta character is defined by the first parameter and defines the following two characters as a digraph.

Suggestion for a command name:

```
:define         this nicely abbreviates to ":def".
:decode         :dec - you always wanted that name in vi, didn't you?
:xl8            translate ("trans-l-eight" ;-)
```

Rules about the following two characters:

```
(1) [0-9a-fA-F][0-9a-fA-F] -> hex value
        =20     =3D
(2) ^[a-zA-Z]              -> control character
        C^M     C^V
(3) \[a-zA-Z]              -> C escape sequence
        \n      newline
        \t      tab
        \r      return
        \g      bell            etc
```

Examples:

```
Map the delete character to the backspace character:
:def = map =^? =^H

Translate ^M to ^J:
:def = :%s/=^M/=^J/

Translate all quoted printable spaces into real spaces:
:def = :%s/=20/ /g
```

Suggested by: Sven Guckes guckes@vim.org [960622]

VIM Wishlist - command line editing

[961007,970911] A frequent wish is to allow "Vi commands" on the "command line" for editing. I admit this sounds like a good idea - especially because this "inconsistency" is not a favourite feature of Vi newbies.

Those in favor argue that you can regard the command line as a window with only one line. But as there is only one line you edit on you would have to disable the colon command (which switches to ex mode), all vertical jump commands, window commands (^W etc), and the commands that open a new line ('o'. 'O'). All these things will require a lot of additions and thus result in *more* code, not less.

Another problem is that the messages for command completion and for showing the current mode cannot be shown. And

you would have to distinguish between command-line-normal-mode and comman-line-insert-mode. This can become quite confusing.

So we'd rather refrain from making the command line yet another window. Hope you understand.

NOTE: The Vi clone "nvi" allows Vi commands to be used on the command line. I wonder how good that works. Comments welcome!

Message-ID: <68lpp1$34h$1@wbnws01.ne.highway1.com> Kaz Kylheku (bill@cafe.net) suggested: "Suppose that the property that an (unmapped) ESC cancels an ex command was eliminated. Thus ESC on the command line would go into a vi-like command mode for editing the command history." Paul G. Fox pgf@foxharp.boston.ma.us commented: "this would be an _extremely_ bad thing, imho. we tried it in vile a few years back, and it was awful. since then, we've considered trying it again, but making two ESC chars in a row do the traditional abort. that might be viable, but i've not yet experienced it, so i'm not sure."

VIM Wishlist - Text Formatting

formatoption 'j' - add space when joining lines

> Change the whitespace following the combination of WORDs and any sequence of the characters that end senetences (".,;:!?") two exactly two spaces.
>
> FAQs about this definition: Why not use the option "joinspaces" for that? It is confusing that the option "joinspaces" has a sideeffect on the option "formatoptions". Why not just add a single space after a sentence? Well, redoing the formatting on the same text will then make the text longer by one space *every* time. This is not wanted. Furthermore, it is better to compress whitespace to just two spaces. Why not compress the whitespace after every end-of-sentence character? This is a bad idea as some people like using ellipses like ". . .". End-of-sentences should thus follow WORDs directly. Why not use "words" instead of WORDs? The default to "words" does not fit some words of the English language as it does not include the dash - so the word "up-to-date" is not a "word", and therefore the sentence "... will have to up-to-date." would have no end. Why not just use ".?!" as end-of-sentence characters? Several languages also use the colon and semicolon as separators between sentences. Yes, this should be made configurable, too.

formatoption 's' - squeeze whitespace

> Automatically condense/squeeze whitespace to a single space.

formatoption 'S' - squeeze whitespace

> Automatically condense/squeeze whitespace to a single space.

VIM Wishlist - Top Wishes

The list of the top most often requested wishes, showing major advances first:

Vim Client [990712]

> The idea is that of "emacsclient" - using just one running editor as a server. So you start editing a file with "vimclient filename" and when Vim is running already then it would just add another window or switch from the current file to the newly opened one.
>
> Suggested by: John Klassa klassa@aur.alcatel.com 990712 (amongst many others)

Shell Buffer / Shell Window

> Shell Window: This opens a window and starts a shell within. The output region is restricted to the window, but can be placed anywhere and can also be rotated and switched with other windows.
>
> Shell Buffer: Just like a shell window - but the input and output is mirrored in an edit buffer, so you can use Vim's commands to jump around and modify the text, including the output of shell commands.
>
> Suggested by: Felix K. Sheng (felix@nytimes.com) [951115] and many others by now, of course. [990726]

Remote Editing [981001]

> Add support for editing a file on a remote disk via ftp.

Suggested by: many people

buffer selection window [980613]

There are many commands to select a buffer - but many people are used to select an item from a list using arrow keys instead of commands. Therefore many users have asked for an extra window showing the buffer list allowing the arrow keys for selecting a buffer to switch to. So far there are mappings sets to simulate this, but an extra window with extra commands is very much desired.

Folding (aka Outlione Mode) and Limited View

Folding: Hide parts of the text to allow faster overview and access to sub-parts. This is to hide parts that are not affected by editing.

Limited view: Show only lines matching (or not matching) a given pattern. This is to focus editing on special lines of a text.

Folding is the main goal for Vim-6. Some folks are writing patches already it seems.

Allow to make parts "read-only" as to prevent accidental changes.

Suggested by: Le Roux Yannick LeRouxY@thmulti.com [991202]

visual mode - "fold"

Fold the current visual text. This is nice when you have a long file to edit and you jump around to edit parts which you don't want to change. It sort of cuts down on the scrolling.

Support for UTF8 (Unicode)

"UTF-8 (as defined in RFC2279 or ISO/IEC 10646-1 Annex R) is an ASCII-compatible encoding of Unicode. UTF-8 allows very easy addition of Unicode/ISO 10646 support to software that was designed for 8-bit character sets. UTF-8 is the way how Unicode is now commonly used under Unix (e.g., Plan9 and Solaris). It should not be too difficult to extend VIM to edit UTF-8 files. The only basic change that is involved is that which in 8-bit character sets one counts the number of bytes to determine the number of characters in a string, with UTF-8 one counts only the bytes with values (x & 0xc0) != 0xc0. Free Unicode fonts for X11 are now freely available, and UTF-8 support for many terminal emulators is already available or under implementation. VI under Solaris is already fully UTF-8 capable."

Markus Kuhn Markus.Kuhn@cl.cam.ac.uk [981126]

matching and searching across newlines [980107]

You can use regular expressions to search for almost anything *within* a line - but not *across* lines. This is "natural" to many programs, especially Vi. Besides, Vim must not this by default to be "Vi compatible".

However, there are several programs which allow this (even with regular expressions). Naturally, Vim users want this, too.

Bram said on 960703:

> I'm afraid that it currently is impossible to include a CR in a search pattern. It is quite a bit of work to add, in a way that it works properly. But it is in the todo list, it will be implemented one day. Don't expect it soon.

Vim-5 may allows this via the hooks to other langauages (perl, python). Here is an example of how you can use Perl to cut up a file into paragraphs, substituting "foo bar" with "baz": :%!perl -00pe 's/foo\s+bar/baz/'

extend "repetition" to more commands

The dot command ('.') repeats the last command. The decription says that this applies to "simple commands". However, it is not described which the commands are. :-(Anyway, this could be extended to a lot more commands. But this is very difficult as the scope is not really defined. Yes, this certainly needs a better documentation.

Overlapping Windows, vertical split [970903]

Splitting windows is not only desirable in horizontal split" but also as "vertical split". this can aid in viewing program code and "context diffs" as well as with translations of text.

Steve Kirkendall plans additional "split commands" for elvis and an option to define the default split orientation (ie "horizontal" or "vertical").

Vim-6 now has "vertical split" now (since vim-6.0h).

format view modes / text rendering [970603]

Display manual pages correctly and render webpages (HTML files) as if viewed with a text browser (such as Lynx).

Extending Block Operations [970812]

More operations on visual block, most significally substitution.

This would allow to fill a block with a single character or pattern, too. (As wished by massimiliano.piccinini@comune.re.it on 980829).

vi style editing on command line [970926,980106]

The editing commands for the command line are quite different from the other commands within the buffer. Some people would prefer to see the same commands for the command line, too. However, this adds more problems as you have both an insert mode and a normal (command) mode on the command line. Furthermore, the visual mode would have to be disabled for the command line. So this seems to require more workarounds than it has benefits.

Lemmy: The vi clone "lemmy" has vi-style command line editing (option "kshmode"). It turns the command line into a mini-ksh using the vi command set. (Still haven't tried this yet. More info welcome!)

Nvi: The vi clone "nvi" allows to edit the command line and the command line history in a separate buffer.

Hex Edit Mode [970609,971230]

Display content of files in hex mode. Allow editing in hex mode.

This has been on everyone's wishlist for Vim for years. However, this would have to change a lot of code. Therefore Vim ships with the utility "xxd" which converts binary files to text; you can then edit this text and have "xxd" convert the result back to binary.

Partial Editing [Hmm, this isn't a good "title" - got a better one?]

Vim loads the whole file into memory. This can be a problem for big files and not sufficient memory, of course. It would be much better if it used only "the parts that other editors cannot reach" (*ehem*), ie only those "pages" that are ebing editing. I assume this will be a major change as it is probably quite some hack for operating systems which do not support "page switching". [980424]

Y2K (Year2000) compliancy

"YES". See the page on Vim and Y2K.

ISO9000

Absolutely!

Vim Wishes for the OS/2 port

XFree86 port the X-GUI mode

Suggested by: Orange Gopher valley@pookie.uchicago.edu 990205

Looking for setup and documentation files relative to executable

"In OS/2, I'm not sure that just argv[0] [to get the path of the current executable] will always work. I have a code fragment somewhere for a function called getLoadPath that calls an obscure OS/2 API to get the name of the executable. This is said to be more reliable than argv[0] (which is compiler dependent)."

Suggested by: Orange Gopher valley@pookie.uchicago.edu 990205

Vim Wishlist - Workarounds

Some things can be done with a simple workaround.

NEW: GNU readline for command line [960123,960601]

Use the mappings as given by Sven's vimrc.

[workaround] command "gf" - do not reset cursor to first line

Problem: Command "gf" with the cursor on "filename" will change to the buffer of "filename" when it is already in the bufer list - but it will reset the cursor to the first line within that buffer.

Suggestion: Make Vim keep the current line of the cursor as if you has used the command ":b filename".

Workaround:

```
noremap gf gf'"
```

Suggested by: Otavio Exel oexel@economatica.com.br [971126]

VIM WIshlist - Wishes which had to be turned down

'rulertab' - tabstop list

Make tabstop a list of positions. This makes editing of tables easier.

Bram said on 990826: "This actually creates a new file format. Only when the tab settings are right will this text show up as intended. This is not very portable, since normal text files have no standard way to tell where the tabstops are. I don't think Vim should support this in the near feature. You could perhaps do something with inserting spaces, using some way to vary the amount based on the current column. Could be done with a clever mapping. Perhaps you would also be happy with the column-aligning scripts that have been send recently."

VIM Wishlist - Wishes that got implemented with Vim

This is where I'll put the wishlist entries whenever someone tells me that some feature is now implemented. ;-)

special character notation

Allow using "\s" for spaces and "\t" for tabs - then mappings would be easier to read and would not break at spaces. Although you can use "<Space>" and "<Tab>" this is quite long.

Suggested by: Sven Guckes guckes@vim.org [971002]

```
vim-5.0r added:
- "\w" in regexp, for whitespace.  Should be much faster than "[ \t]".
vim-5.0s:
- Added "\S" to regexp patterns: matches a non-white character.
  Changed "\w" to "\s", to make it similar with Perl.
```

Using "\s\+" for any non-zero sequence of these is very handy! I use "\s\+$" to detect trailing whitespace.

swap file - create swap file after first modification

Create the swap *after* the first modification - not before. Optional.

Suggested by: Yang Guo Liang (glyang@iss.nus.sg) and Sven Guckes guckes@vim.org [970919]

Added with vim-5.0o. [970929]

search pattern - Perl minimum closure

Allow to match the shortest possible match for a regex.

Example: "s/\<.*?\>/[\1]/" puts square brackets around the shortest word.

Suggested by: Edwin van Geelen edwin@mindless.com [990311]

Vim has this already - the "magic patttern" "\{-}".

VIM Wishlist - TODO

- Documentation: Check occurrences of "<NT>" - should they be replaced by "<EOL>"?

Back to the -> VIM Home Page

```
URL:          http://www.math.fu-berlin.de/~guckes/vim/wish.html
URL:          http://www.vim.org/wish.html (mirror)
Created:      Mon May 15 00:00:00 MET 1995
```

Send feedback on this page to
Sven Guckes guckes@vim.org

VIM.ORG - Vim Organization

This page gives information about the current status of the domain "VIM.ORG".

Overview

Email

Everything about Email to vim.org should be explained on the page about mails

The following addresses at vim.org should be valid:

```
ADDRESS@VIM.ORG    TARGET ADDRESS

 addresses by function
 (target addresses can be changed when service is taken over someone else):

 function@vim.org      target address                Name
 ================      ==============                ================
 author                   bram@vim.org               Bram Moolenaar
 bugs                     bram@vim.org               Bram Moolenaar
 bugreport                bram@vim.org               Bram Moolenaar
 faq                       faq@vim.org               Sven Guckes
 *ftp*                     sec@42.org                Stefan Zehl
 postmaster             leitner@fefe.de              Felix von Leitner
 vimfaq                  guckes@vim.org              Sven Guckes
 webmaster               guckes@vim.org              Sven Guckes
 wish                    guckes@vim.org              Sven Guckes
 wishlist                guckes@vim.org              Sven Guckes
 www                     guckes@vim.org              Sven Guckes

 name@vim.org          target address                Name
 ============          ==========================    ====================
 bram                     bram@vim.org               Bram Moolenaar
 duperval              laurent@Grafnetix.COM         Laurent Duperval
 felix                 leitner@math.fu-berlin.de     Felix von Leitner
 guckes                 guckes@math.fu-berlin.de     Sven Guckes
 leitner               leitner@math.fu-berlin.de     Felix von Leitner
 laurent               laurent@Grafnetix.COM         Laurent Duperval
 mool                     bram@vim.org               Bram Moolenaar
 moolenaar                bram@vim.org               Bram Moolenaar
 sec                       sec@42.org                Stefan Zehl
 stefan                    sec@42.org                Stefan Zehl
 sven                   guckes@vim.org               Sven Guckes
```

Unfortunately, syntax file maintainers won't get their own vim.org mail address. I tried, but the idea was turned down. Sorry!

Note to address harvesters: Don't send spam - or else!

---

vim.org - WWW mirrors

The site www.vim.org is just a **mirror** of the Vim Pages at `http://www.math.fu-berlin.de/~guckes/vim/`. The pages are mirrored using "webcopy" and are mirrored twice daily. If you need to look at a change that is very very recent (say, within the last 24 hours) then please look at the original site.

If you like to set up a mirror for www.vim.org then you are very welcome to do so. See the Vim HowTo on Mirroring www.vim.org for more info on this matter.

---

vim.org - Distribution, Download, FTP

The main distribution site for vim is

- ftp://ftp.vim.org/pub/vim/

This site is hosted on the ftp server of the "NetherLands Unix User Group" (ftp.nluug.nl).

Vim is mirrored on many sites (see the distribution page).

FTP and WWW mirrors in other countries usually bear the name ftp.**city.country**.vim.org and `www.XY.vim.org`, respectively, where XY is the two letter code of that country.

Also, I will try to get all distribution sites to carry vim in /pub/vim. Let's see if this is feasible.

I am considering to make the helpfiles of the current version available for FTP as an archive by itself in HTML for browsing ie ftp://ftp.vim.org/html/vim.docs.html.tar.gz . Is anybody interested in this? Let me know! Send mail to guckes@vim.org !

---

vim.org - History

000819

vim.org is now hosted by gandi.net

- whois vim.org

980625

Some more CNAMEs have been added for the ftp servers of vim.org. Check with "nslookup" if you like:

```
$ nslookup
server ns.vim.org
ls -a vim.org
```

971017

We are currently naming the ftp mirrors...

970918

The pages on vim.org are now a mirror of the pages at Sven's place. No more redirection -> bookmarks go to vim.org. :-) It is intended that the pages on VI also get mirrored. So all info on VI is then on vim.org. Hopefully this will then become *the* reference site on VI on the Web. Send me the addresses on good VI info!

970915

Success!

VIM.ORG finally has come to life! :-)

```
$ whois vim.org
Vim.org - Vi IMproved (Sven Guckes) (VIM3-DOM)
   Pariser Str. 52
   Berlin, 10719
   de

   Domain Name: VIM.ORG

   Administrative Contact:
      Guckes, Sven  (SG3457)  guckes@VIM.ORG
      +49(177)7777796 (FAX) +49(30)8838884
   Technical Contact, Zone Contact:
      Zehl, Stefan  (SZ232)  sec@MUFFIN.ORG
      0177/2340515 (FAX) +49(89)3618023
   Billing Contact:
      Zehl, Stefan  (SZ232)  sec@MUFFIN.ORG
      0177/2340515 (FAX) +49(89)3618023

   Record last updated on 15-Sep-97.
   Record created on 15-Sep-97.
   Database last updated on 16-Sep-97 04:47:04 EDT.

   Domain servers in listed order:

   R2D2.MUSIN.DE                194.113.40.45
   EWOK.PI.MUSIN.DE             194.246.250.2

The InterNIC Registration Services Host contains ONLY Internet Information
(Networks, ASN's, Domains, and POC's).
Please use the whois server at nic.ddn.mil for MILNET Information.
```

970911

I am looking at the mails I have received about vim.org and I am preparing a mail to all. Please stay tuned!

970911

VIM.ORG has been registered at InterNIC.

---

vim.org - Credits

People who have helped and offered their help to me:

Stefan 'Sec' Zehl

> The maintainer of the vim.org domain, especially the DNS entries for the hosts that carry the ftp and www mirrors.

Sirtaj Singh Kang

> Creator of the script that converts Vim helpfiles into HTML.

People to add: ??? <zeus@tomserver.phys.ttu.edu> B.G. Mahesh <mahesh@mahesh.com> Bill Duncan <bduncan@beachnet.org> Brian Grossman <brian@SoftHome.net> Chia-liang Kao <clkao@tc.neto.net> Felipe Contreras Alcala <fhca@csi.UOttawa.CA> Frank Cusack <frank@fore.com> Hans-Joachim Gurt <gurt@nacamar.net> Jonathan Bradshaw <Jonathan@NrgUp.Com> Ken Weingold <hazmat@hellrot.org> Lars Marowsky-Bree <lmb@pointer.teuto.de> Linh D. Ngo <linh@vr1.com> Mark Constable <markc@motd.com> Mark Devlin <devlin@usa.net> Martin Mares <mj@atrey.karlin.mff.cuni.cz> Michael-John Turner <mjturner@icon.co.za> Mike Depot <mike@ici.ne> Philip Hallstrom <philip.hallstrom@sierra.com> Sam Trenholme <sam@samiam.org> Steinar Knutsen <sk@nvg.ntnu.no> Sudhakar Chandrasekharan<thaths@netscape.com> Tom Julien <tomj@orl.lmco.com> Wes A. Jones <wjones@fireblazer.com> Yiorgos Adamopoulos <Y.Adamopoulos@noc.ntua.gr> Dale Harris <rodmur@sampoerna.cisnet.net>

Other VIM organizations

The abbreviation "VIM" apparently is quite popular - several companies are using it. The result of a whois query on "vim" on 970911:

```
$ whois vim
Avila, Roy M. (RMA8)       vim@CYBERCOM.NET                 (617)773-1536
Chirife, Alejandro (AC294) vim@SHADOW.NET                    305-536-3520
Henry, Martin (MH1305)     vim@CYBERCOM.NET                 (617)878-7936
Idan, Tzvika (TI265)       vim@INTER.NET.IL +972-9-9599093 (FAX) +972-9-9599608
Mahadeva, Vimalan (VM577)  vim@UCLA.EDU       310 854-8220 (FAX) 310 652-9053
VIM (VDOMAIN2-DOM)                                                VDOMAIN.COM
VIM (Verkaik Innovative Management) (VIM2-DOM)                        VIM.NET
VIM Industries (UKRAINEGIRLS-DOM)                               UKRAINEGIRLS.COM
VIM Studios (LOCALFINDER-DOM)                                    LOCALFINDER.COM
VIM Studios (NETTRACKS-DOM)                                        NETTRACKS.COM
VIM Studios (SECUREDTRANS-DOM)                                  SECUREDTRANS.COM
VIM Technologies, INC. (VIMTECHNOLOGIES-DOM)                 VIMTECHNOLOGIES.COM
VIM, Inc. (ONLINESALES-DOM)                                      ONLINESALES.COM
VIm Partners (VALUEINVESTMENT-DOM)                           VALUEINVESTMENT.COM
Vim Corp. (VIMC-DOM)                                                VIMC.COM
Vim Gupta (VG12-ORG)            customers@DESIG.NET               0181 348 2247
Vim Studios (ADVERTISEIT-DOM)                                   ADVERTISEIT.COM
Vim Studios (MYVEGAS-DOM)                                          MYVEGAS.COM
Vim Studios (MOVIETHEATRES3-DOM)                              MOVIETHEATRES.COM
Vim Studios (VIMSTUDIOS-DOM)                                     VIMSTUDIOS.COM
Vim Studios (OUTTASITES-DOM)                                     OUTTASITES.COM
```

And last but not least one of my favourite Vim companies:

From a Biography of Oliver Hardy (a famous comedian):

"Oliver is known to have been in Jacksonville, Florida at least until the **Vim Comedy Company** stopped production in 1916."

No, Vi IMproved was not meant as a joke. ;-)

```
URL:          http://www.math.fu-berlin.de/~guckes/vim/orga.html
URL:          http://www.vim.org/orga.html (mirror)
Created:      Thu Sep 11 00:00:00 CET 1997
```

Send feedback on this page to
Sven Guckes guckes@vim.org

VIM - The Press / Reviews

VIM is mentioned in the press at various places. Too bad that not all of these articles are accessible via the Web.

```
to add:

http://www.linux.com/search/index.phtml/vim/

page: http://www.linux-mag.com/2000-06/websites_09.html
(Category) "Documentation"
The Vim Homepage
http://www.vim.org/
text:
Get a handle on the editor that strikes fear into the heart of newbies
everywhere, Vim. Vim is the most popular vi clone for Linux, and the Vim
homepage has tons of information on using Vim and how to customize it.


page: http://www.linux-magazin.de/ausgabe/1997/03/
link: http://www.linux-magazin.de/ausgabe/1997/03/Quickrefs/vimref.ps
author: Peter Zitzelsperger
size:  (42K)
```

February 2001 - "Linux Journal" [ISSN ?]

> pages 140 - 146: "That's Vimprovement! A Better vi" (Be a better editor - try Vim) by Steve Oualline. Steve manages to explain the main Vim features in five pages, with pictures and examples. From using undo/redo and multiple windows, via syntax highlighting and :make, to recording and executing a macro.

October 2000 - Linux200.nl conference, 9-10 Oct 2000 in Ede

The continuing story of Vim

> The paper submitted by Bram for the Linux200.nl conference, taking place 9-10 Oct 2000 in Ede (Netherlands)
>
> Author: Bram Moolenaar

September 2000 - "Software Design" [ISSN 0916-6297]

> http://www.gihyo.co.jp/SD/index-j.html
>
> Author: Takuhiro Nishioka takuhiro@super.win.ne.jp
>
> Contents: "Hajimete no Vim (Learning the Vim editor)"
>
> Language: Japanese
>
> Scan of article: http://www.win.ne.jp/~takuhiro/image_files/vim_software_design.png

August 2000 - OS/2 eZine [001114]

Vim for OS/2 (local copy)

> http://www.os2ezine.com/20000816/vim.html

Author: Richard R. Klemmer richard@webtrek.com

"Why use Vim? .. if you ever have to telnet into your OS/2 system, you won't be able to use any of the Presentation Manager programs, such as EPM, so you will need at text mode program to edit any files. You can use ***** which comes with OS/2, but I think you'll find that this pales in comparison to what you can to with VIM."

July 2000 - LinuxNewbie.org

http://www.linuxnewbie.org/nhf/intel/programming/intro_c++.html

Author: Keith Jones kmj9907@cs.rit.edu

Contents: Mentions tags, C-style indenting, QuickFix mode, some useful keystrokes (jumps), substitution command, misc features, and a few links.

March 2000

Title: **Interview with Bram Moolenaar**

Media: PCRevue

original

Title page and contsnts:

http://www.pcrevue.com/buxus/generate_page.php3?page_id=148

local copy

Author: Juraj Bednar juraj@bednar.sk

November 1999

Title: **VIM: Vi IMproved!**

Media: Linux Magazine France

scan of article (**210K**)

Author: Henri Dumoulin

"While some prefers 'clickodromes', advanced users prefer lighter interfaces and rather in a console. This is typically the case in text editors. On one side: Gnotepads, Knotes and other notepad clones. On the other side: the never-ending dual VI/emacs. Let's have a look at the free version of VI written by B.Moolenaar : VIM 5.5"

October 1999

Title: **An Interview with Vim Authors**

Media: Effervescence (Magazine de l'Association des Ingénieurs EFREI)

Author: Hervé Foucher

In English: http://web.efrei.fr/aiefrei/effervescence/123/vim.en.html

In French: http://web.efrei.fr/aiefrei/effervescence/123/vim.fr.html

In Chinese: effervescence.interview.cn.html Translated by slimzhao@21cn.com [010503,010607]

October 1999

Title: **Text Editors for DOS and Windows**

Media: IEEE.org (IEEE Electronic Communications)

http://www.spectrum.ieee.org/INST/oct99/inf_hwy.html

Author: Bob Alden r.alden@ieee.org

Local copy (text only): ieee.oct99.inf_hwy.txt [7K]

October 1999

Title: **Linux Advanced Workshop #2**

Media: PCPlus

http://www.futurenet.com/pcplus/article.asp?id=11723

Author: Chris Jones

Issue 156, pages 187/188. Section "Hands On", Linux advanced workshop. Written by Chis Jones. It is two pages, and goes from the first commands to a few more advanced topics.

There are two pictures. The first shows the Vim 5.3 startup screen (although the article mentions version 5.4). The other shows a message in the comp.editors newsgroup from Sven Guckes to Michael Soulier, subject "tabs in VI -> Vim and listchars". It's shown with syntax highlighting.

The included CDROM contains Vim-5.4 with the runtime archive and RPMs for RedHat.

September 1999

Title: **From Linux to FreeBSD**

Media: Daemon News - Web Magazine

http://www.daemonnews.org/199909/adventure.html

Author: Justin Hawkins jh@tardis.selkie.org

".. Within a few hours (unfortunately I'm still using a 28.8K modem) I had compiled and installed all my essential software, Samba [3], Apache[4], Ghostscript[5] and of course vim[6]. [...] [6] VI iMproved. Adds many features to standard vi, including colour syntax highlighting for program code."

August 1999

Title: **Using vi - an introduction**

Media: IEEE Electronic Communications

http://www.spectrum.ieee.org/INST/aug99/inf_hwy.html

Author: Bob Alden r.alden@ieee.org

Local copy (text only): ieee.aug99.inf_hwy.txt [7K]

August 1999

Title: **And the Winner is...**

Media: LinuxWorld.com

http://www.linuxworld.com/lw-1999-08/lw-08-penguin_1.html

The results to the **LinuxWorld Editors' Choice** were given on Aug 11th. Here is the result of the category "Text Editors": "Text Editing -- Winner: Emacs -- Runner-up: Vim"

> Vim: What Vim lacks in features (as compared to the feature-heavy Emacs), it makes up easily in speed and simplicity. Vim is the ideal editor for the kinds of quick jobs one must always do under Linux, such as the editing of text-based configuration files or shell scripts. In this case, we chose not to go to the extreme of simplicity, but to balance speed and simplicity with power. Vim seems to find the best balance of the editors we tried, and gets an extra boost because it is an extension of vi, the simple editor that most Unix users seem to know -- if not sooner, then later.

June 1999

Media: LinuxWorld

Linux development: CLI, Emacs, or IDE

http://www.linuxworld.com/linuxworld/lw-1999-06/lw-06-vcontrol_1.html

> KDE developer and evangelist Kurt Granroth has a different idea. When I asked him about his preferred development environment he said "I use vim (not just vi -- only vim will do) with egcs and gdb."

February 1999

Title: **An Interview with the VIMpire**

Media: EXT2.org

http://www.ext2.org/99/02/vim.html

Local copy: vimpire.html

Author: Rob Kennedy rob@ext2.org

Bram Moolenaar answers some FAQs: What is Vim? How did Vim start off? How close to Vi is Vim? What are the best improvements?

October 1998

Title: **Inside Slashdot**

Media: Slashdot

http://linuxworld.com/linuxworld/lw-1998-10/lw-10-slashdot.html

> Slashdot's Rob Malda (aka CmdrTaco) takes us through the technical changes his popular Web site has undergone in the last year and a half...
> "...but as Slashdot's popularity grew, the response became more and more sluggish. During peak hours, the machine's load level would climb to 10.0 (A load of 1.0 refers to one fully busy CPU; 0.50 to 1.50 is normal) or even higher. Even at that point, telnet sessions running pine and **vim** remained usable. I was quite impressed. .."

September 1998 - Wandering-Man.com

The Vim text editor for Java programmers - "Not Only, but Also.."

http://www.wandering-man.com/Java/vim/java_Vim.html

Author: Allan Kelly allan.kelly@ed.ac.uk

Nice article with screenshots.

September 1998

Title: **Extended Standard**

Media: "iX 9/98", page 67

Author: Garry Glendown garry@insider.regio.net

Article in English

Article in German

A one-page article on vim-5.1 mentioning "Vi Improvement" as "having many options".

June 1998 [980429]

Revisiting VIM

Media: "Linux Gazette", Issue 29

Author: Andy Kahn kahn@zk3.dec.com

http://www.linuxgazette.com/issue29/kahn.html

(List of) Features, Syntax highlighting, Built-in Scripting, Visual Text Selecting (indenting, deindenting, filtering), C and C++ tags. With a dozen screenshots.

February 1998

Linux - 2 Cents about vim for pico users

Media: "Linux Gazette", Issue 25

Author: Sven Guckes guckes@math.fu-berlin.de

http://www.linuxgazette.com/issue25/lg_tips25.html#vim

Sven Guckes gives a tip for users of the editor "pico" who are used to "justify" (read: reformat) the current paragraph with CTRL-J. Here's how to do this with Vim: nmap <C-J> vipgq nmap <C-J> gq

January 1998

Title: **New Editor Versions**

Media: "Linux Gazette", Issue 24

http://www.linuxgazette.com/issue24/issue24.html

This issue quotes the Usenet article

686qdj$n93$1@wbnws01.ne.highway1.com

> Vim is probably the most featureful of the VI-style editors. Judging by newsgroup postings, it may be the most popular as well. With the release of vim-5.0s, vim 5 has finally reached a beta rather than alpha state. This revision has a really well-implemented syntax-highlighting system for many programming and shell-script languages, and it's not too difficult to adapt to new file-types and languages. The down-side is that vim is growing larger, and is beginning to lose the quickness and low memory-usage that has been a hallmark of VI-style editors. [...]

June 1997

Title: **VIM Programming Perks**

Media: "Linux Gazette" - Issue 18

Author: John M. Fisk fiskjm@ctrvax.vanderbilt.edu

http://www.linuxgazette.com/issue18/wkndmech.html#vim-perks

John gives some hints on programming with Vim-5.0e: GUI, syntax highlighting, :help, tags, exuberant ctags, :split and window commands (CTRL-W), using RCS, things to do with the current file (printing, spell checking, read in other data), running :make, unlimited undo, support for compressed files; sample setup, and eleven screen shots.

February 1997

Title: **Pick an Editor, Any Editor**

Media: "Linux Gazette" - Issue 14

Author: Jens Wessling jwesslin@erim.org

http://www.linuxgazette.com/issue14/vim.html

Mentions the "ctags" utility, Includes mappings to (un)comment lines. [980106]

August 1995

Title: **If you gotta use VI, use VIM**

Media: "Linux Gazette", Issue 01-08

Author: John M. Fisk fiskjm@ctrvax.vanderbilt.edu

http://www.linuxgazette.com/issue01to08/linux_gazette.aug.html#vim

"It actually tells you what mode you're in. It has an easily accessible on-line help function. It works quite well under X when teamed up with xterm." Also mentions that you can ":split" windows.

TODO

Linux-Mag.com September 1999

http://www.linux-mag.com/1999-09/tech_support_01.html

> "There are other text editors you can investigate, but they are not for the faint of heart, as they are known for their arcane and hard to remember commands. Among these are vi, vim,elvis and their clones. If you intend to make a career out of being a Linux or UNIX sysadmin, I recommend at least having a passing familiarity with vi, as it is installed by default on every version of UNIX, even in its most pared-down state."

Linux-Mag.com August 1999

Author: Hal Moroff halm@ieee.org

http://www.linux-mag.com/1999-08/newbies_02.html

> You must edit a file to do this. The text editors that come with Linux are very good, but will take a fair amount of explaining. For now I recommend the vi (or vim) editor.

For more information on Vim see the Vim Pages.

```
URL:          http://www.math.fu-berlin.de/~guckes/vim/press/
URL:          http://www.vim.org/press/ (mirror)
Created:      Tue Jun 09 00:00:00 CEST 1998
```

Send feedback on this page to
Sven Guckes guckes@vim.org

VIM - Languages and Syntax Files

This page has a list of all available syntax setup files and their maintainers. The syntax files might not yet be in the distribution, but they are hopefully available on the Web somewhere and will be added soon.

Submissions: If you have a syntax file to share with us then please send it to the author of Vim, Bram Moolenaar bram@vim.org ! It would be nice if you made its current version available on the Web, too, so I can link it from the Vim User Page. Please create these pages then: The languages and everything that goes with it should be explained on the pages **http://www.domain/~user/vim/syntax/** - thanks, folks! :-)

Fellow Linguists: Do you use some language quite often? Do you know how to use its vim syntax file, too? Would you like to share your experience with other users who have a need for editing this language? Then you should become a "fellow linguist"! :-)

Available Syntax Files and their maintainers

Vim-5.7 [000624] - 215 Syntax Files

Vim-6.0s [010114] - 263 Syntax Files

TODO:

- Add name and address of (main) maintainer.
- Add date of last update.
- Add links to online versions not yet mentioned in the vim-5.7 version of the syntax files.
- Send newer online versions to Bram.
- Add links to snytax files in vim-6 alpha version.
- List of maintainers of several syntax files. (Bigham, Campbell, Eymers, Fleiner, Lanzarotta, Moolenaar, Mudunuri, Riiser, ..)
- Add links to current version (if available). DONE.

VERSION + DATE	NAME + LINK	AUTHOR	ADDRESS	STATUS	URL
vim-5.7 000114	2html	Bram Moolenaar	Bram@vim.org	---	---
vim-5.7! 000208	abaqus	Carl Osterwisch	osterwischc@asme.org	---	---
vim-5.7 990526	abc	James Allwright	J.R.Allwright@westminster.ac.uk	404	http://perun.hscs.wmin.ac.uk/vim/syntax/abc.vim
vim-5.7 990922	abel	John Cook	john.cook@kla-tencor.com	---	---

vim-5.7 981207	ada	David A. Wheeler	wheeler@ida.org	---	---
vim-5.7 990928	ahdl	John Cook	john.cook@kla-tencor.com	---	---
vim-5.7 980518	amiga	Dr. Charles E. Campbell, Jr.	Charles.E.Campbell.1@gsfc.nasa.gov	---	---
vim-5.7 991104	aml	Todd Glover	todd.glover@gems9.gov.bc.ca	---	---
vim-5.6 000124	apache	Allan Kelly	allan@fruitloaf.co.uk [001031:new!]	---	---
vim-5.7 991028	apachestyle	Christian Hammers	ch@westend.com	---	---
vim-5.7 000209	asm	Kevin Dahlhausen	ap096@po.cwru.edu	---	---
vim-5.7 970420	asmh8300	Kevin Dahlhausen	ap096@po.cwru.edu	---	---
vim-5.7 980307	asn	Claudio Fleiner	claudio@fleiner.com	010117 ok	http://www.fleiner.com/vim/syntax/asn.vim
vim-5.7 000606	aspperl	Aaron Hope	edh@brioforge.com	010117 ok	http://nim.dhs.org/~edh/aspperl.vim
vim-5.7 000501	aspvbs	Devin Weaver	ktohg@tritarget.com	010117 ok	http://tritarget.com/vim/syntax/aspvbs.vim
vim-5.7 980331	atlas	Inaki Saez	jisaez@sfe.indra.es	---	---
vim-5.7 980911	ave	Jan-Oliver Wagner	Jan-Oliver.Wagner@usf.Uni-Osnabrueck.DE	---	---
vim-5.7 971127	awk	Antonio Colombo	antonio.colombo@jrc.org	---	---
vim-5.7 990914	basic	Allan Kelly	Allan.Kelly@ed.ac.uk	---	---
vim-5.7 000606	bc	Vladimir Scholtz	vlado@gjh.sk	010117 ok	http://www.gjh.sk/~vlado/bc.vim
vim-5.7 000510	bib	Bernd Feige	Bernd.Feige@gmx.net	010117 ok	http://home.t-online.de/home/Bernd.Feige/bib.vim
vim-5.7 990702	btm	John Leo Spetz	jls11@po.cwru.edu	---	---
vim-5.7 000601	c	Bram Moolenaar	Bram@vim.org	---	---
vim-5.7 000110	cf	Jeff Lanzarotta	frizbeefanatic@yahoo.com !	---	---

vim-5.7 000507	ch	YoungSang Yoon	image@lgic.co.kr	---	---
vim-5.7 990823	change	Andreas Scherer	andreas.scherer@pobox.com	---	---
vim-5.7 980206	clean	Pieter van Engelen	pietere@sci.kun.nl	---	---
vim-5.7 990816	clipper	Claudio R Zamana	zamana@zip.net	---	---
vim-5.7 991221	cobol	Sitaram Chamarty	sitaram@dimensional.com	---	---
vim-5.7 981101	colortest	Bram Moolenaar	Bram@vim.org	---	---
vim-5.7 000211	conf	Bram Moolenaar	Bram@vim.org	---	---
vim-5.7 000415	config	Christian Hammesr	ch@lathspell.westend.com	---	---
vim-5.7 000412	cpp	Ken Shan	ccshan@post.harvard.edu	---	---
vim-5.7 000316	csh	Dr. Charles E. Campbell, Jr.	Charles.E.Campbell.1@gsfc.nasa.gov	???	???
vim-5.7 000425	csp	Jan Bredereke	brederek@tzi.de	000614!	http://www.tzi.de/~brederek/vim/csp.vim
vim-5.7 990701	css	Claudio Fleiner	claudio@fleiner.com	001020!	http://www.fleiner.com/vim/syntax/css.vim
vim-5.7 971215	cterm	Sean M. McKee	mckee@misslink.net	---	---
vim-5.7 991025	ctrlh	Bram Moolenaar	Bram@vim.org	---	---
vim-5.7 990922	cupl	John Cook	john.cook@kla-tencor.com	---	---
vim-5.7 990921	cuplsim	John Cook	john.cook@kla-tencor.com	---	---
vim-5.7 000530	cvs	Matt Dunford	zoot@zotikos.com	---	---
vim-5.7 990823	cweb	Andreas Scherer	andreas.scherer@pobox.com	---	---
vim-5.7 990402	dcl	Dr. Charles E. Campbell, Jr.	Charles.E.Campbell.1@gsfc.nasa.gov	???	???
vim-5.7 990817	def	Robert Brady	robb@datatone.com (bounce!)	010117 ok	http://www.datatone.com/~robb/vim/syntax/def.vim

vim-5.7 000208	diff	Bram Moolenaar	Bram@vim.org	---	---
vim-5.7 990614	diva	Toby Schaffer	jtschaff@eos.ncsu.edu	---	---
vim-5.7 990514	dosbatch	Mike Williams	mrw@netcomuk.co.uk	---	---
vim-5.7 990623	dosini	Sean M. McKee	mckee@misslink.net	---	---
vim-5.7 971111	dracula	Scott Bordelon	slb@artisan.com	---	---
vim-5.7 991209	dtd	Johannes Zellner	johannes@zellner.org	001110!	http://www.zellner.org/vim/syntax/dtd.vim
vim-5.7 990919	eiffel	Reimer Behrends	behrends@cse.msu.edu	010117 ok	http://www.cse.msu.edu/~behrends/vim/eiffel.vim
vim-5.7 990618	elf	Christian V. J. Bruessow	cvjb@bigfoot.de	---	---
vim-5.7 981111	elmfilt	Dr. Charles E. Campbell, Jr.	Charles.E.Campbell.1@gsfc.nasa.gov	???	???
vim-5.7 000102	erlang	Kresimir Marzic	kmarzic@fly.srk.fer.hr	001029!	http://fly.srk.fer.hr/~kmarzic/vim/syntax/erlang.vim
vim-5.7 980812	esqlc	Jonathan A. George	jageorge@tel.gte.com	---	---
vim-5.7 990616	expect	Ralph Jennings	knowbudy@oro.net	---	---
vim-5.7 980518	exports	Dr. Charles E. Campbell, Jr.	Charles.E.Campbell.1@gsfc.nasa.gov	???	???
vim-5.7 990813	focexec	Rob Brady	robb@datatone.com (bounce!)	010117 ok	http://www.datatone.com/~robb/vim/syntax/focexec.vim
vim-5.7 000517	form	Michael M. Tung	michael.tung@uni-mainz.de	---	---
vim-5.7 990618	forth	Christian V. J. Bruessow	cvjb@bigfoot.de	---	---
vim-5.7 000619	fortran	Ajit J. Thakkar	ajit@unb.ca	001214!	http://www.unb.ca/chem/ajit/fortran.vim
vim-5.7 000329	fvwm	Haakon Riiser	hakonrk@fys.uio.no	---	---
vim-5.7 991021	gdb	Claudio Fleiner	claudio@fleiner.com	010117 ok	http://www.fleiner.com/vim/syntax/gdb.vim
vim-5.7 990702	gdmo	Gyuman Kim	violino@dooly.modacom.co.kr	older!	http://dooly.modacom.co.kr/~violino/gdmo.vim

vim-5.7 991223	gedcom	Paul Johnson	pjcj@transeda.com	---	---
vim-5.7 990614	gnuplot	John Hoelzel	johnh51@earthlink.net	---	---
vim-5.7 000623	gp	Karim Belabas	Karim.Belabas@math.u-psud.fr	---	---
vim-5.7 990508	haskell	John Williams	jrw@pobox.com	---	---
vim-5.7 000402	hb	Alejandro Forero Cuervo	NONE	010117 ok	http://bachue.com/hb/vim/syntax/hb.vim
vim-5.7 000606	help	Bram Moolenaar	Bram@vim.org	---	---
vim-5.7 990707	hitest	Ronald Schild	rs@scutum.de	---	---
vim-5.7 000512	html	Claudio Fleiner	claudio@fleiner.com	010117 ok	http://www.fleiner.com/vim/syntax/html.vim
vim-5.7 000520	htmlm4	Claudio Fleiner	claudio@fleiner.com	010117 ok	http://www.fleiner.com/vim/syntax/htmlm4.vim
vim-5.7 000109	icon	Wendell Turner	wturner@halcyon.com	010117 ok	ftp://ftp.halcyon.com/pub/users/wturner/icon.vim
vim-5.7 971120	idl	Jody Goldberg	jodyg@idt.net	---	---
vim-5.7 990614	idlang	Hermann Rochholz	Hermann.Rochholz@faidor.com	---	---
vim-5.7 000305	inform	Stephen Thomas	stephent@insignia.com	---	---
vim-5.7 990614	ishd	Robert M. Cortopassi	cortopar@mindspring.com	---	---
vim-5.7 991009	ist	Peter Meszaros	pmeszaros@effice.hu	---	---
vim-5.7 000601	java	Claudio Fleiner	claudio@fleiner.com	001020	http://www.fleiner.com/vim/syntax/java.vim
vim-5.7 980722	javacc	Claudio Fleiner	claudio@fleiner.com	010117 ok	http://www.fleiner.com/vim/syntax/javacc.vim
vim-5.7 990420	javascript	Claudio Fleiner	claudio@fleiner.com	010117 ok	http://www.fleiner.com/vim/syntax/javascript.vim
vim-5.7 990614	jgraph	Jonas Munsin	jmunsin@iki.fi	---	---
vim-5.7 000326	jproperties	Simon Baldwin	simonb@sco.com	---	---

vim-5.7 000412	jsp	Rafael Garcia-Suarez	rgarciasuarez@free.fr (garcia_suarez@hotmail.com)	001221	http://altern.org/rgs/vim/syntax/jsp.vim
vim-5.7 000413	kscript	Thomas Capricelli	orzel@yalbi.com	404	http://aquila.rezel.enst.fr/thomas/vim/kscript.vim
vim-5.7 991223	kwt	Michael Piefel	piefel@informatik.hu-berlin.de	---	---
vim-5.7 990614	lace	Jocelyn Fiat	utilities@eiffel.com	---	---
vim-5.7 000614	latte	Nick Moffitt	nick@zork.net	---	---
vim-5.7 990513	lex	Dr. Charles E. Campbell, Jr.	Charles.E.Campbell.1@gsfc.nasa.gov	???	???
vim-5.7 981203	lhaskell	John Williams	jrw@pobox.com	---	---
vim-5.7 990614	lilo	Klaus Muth	mh@hagos.de	001104	http://unitopia.mud.de/~monty/vim/syntax/lilo.vim
vim-5.7 000321	lisp	Dr. Charles E. Campbell, Jr.	Charles.E.Campbell.1@gsfc.nasa.gov	???	???
vim-5.7 000105	lite	Lutz Eymers	ixtab@polzin.com	010117 ok	http://www-public.rz.uni-duesseldorf.de/~eymers/vim/syntax/lite.vim
vim-5.7 980819	lotos	Daniel Amyot	damyot@csi.uottawa.ca	010117 ok	http://lotos.csi.uottawa.ca/~damyot/vim/lotos.vim
vim-5.7 990618	lout	Christian V. J. Bruessow	cvjb@bigfoot.de	---	---
vim-5.7 971123	lss	Scott Bigham	dsb@cs.duke.edu	---	---
vim-5.7 981009	lua	Carlos Augusto Teixeira Mendes	cmendes@inf.puc-rio.br	---	---
vim-5.7 990901	m4	Claudio Fleiner	claudio@fleiner.com	010117 ok	http://www.fleiner.com/vim/syntax/m4.vim
vim-5.7 990827	mail	Felix von Leitner	leitner@math.fu-berlin.de	---	---
vim-5.7 000424	make	Claudio Fleiner	claudio@fleiner.com	001031	http://www.fleiner.com/vim/syntax/make.vim
vim-5.7 990614	man	Gautam H. Mudunuri	gmudunur@informatica.com	---	---
vim-5.7 981206	manual	Bram Moolenaar	Bram@vim.org	---	---
vim-5.7 981016	maple	Dr. Charles E. Campbell, Jr.	Charles.E.Campbell.1@gsfc.nasa.gov	???	???

vim-5.7 990824	masm	Rob Brady	robb@datatone.com (bounce!)	990816!	http://www.datatone.com/~robb/vim/syntax/masm.vim
vim-5.7 990813	master	Rob Brady	robb@datatone.com (bounce!)	010117 ok	http://www.datatone.com/~robb/vim/syntax/master.vim
vim-5.7 980728	matlab	Preben "Peppe" Guldberg	c928400@student.dtu.dk	980706!	http://www.student.dtu.dk/~c928400/vim/syntax/matlab.vim
vim-5.7 990527	mel	Robert Minsk	egbert@centropolisfx.com	---	---
vim-5.7 980426	mf	Andreas Scherer	andreas.scherer@pobox.com	---	---
vim-5.7 990621	mib	David Pascoe	David.Pascoe@jtec.com.au	---	---
vim-5.7 970914	model	Bram Moolenaar	Bram@vim.org	---	---
vim-5.7 990614	modsim3	Philipp Jocham	flip@sbox.tu-graz.ac.at	---	---
vim-5.7 990614	modula2	Peter Funk	pf@artcom0.north.de	---	---
vim-5.7 990614	modula3	Timo Pedersen	dat97tpe@ludat.lth.se	---	---
vim-5.7 980803	mp	Andreas Scherer	andreas.scherer@pobox.com	---	---
vim-5.7 000105	msql	Lutz Eymers	ixtab@polzin.com	010117 ok	http://www-public.rz.uni-duesseldorf.de/~eymers/vim/syntax/msql.vim
vim-5.7 980710	muttrc	Preben "Peppe" Guldberg	c928400@student.dtu.dk	010117 ok	http://www.student.dtu.dk/~c928400/vim/syntax/muttrc.vim
vim-5.7 000215	nasm	Manuel M.H. Stol	mmh.stol@a1.nl	---	---
vim-5.7 000410	nastran	Tom Kowalski	tom.kowalski@mscsoftware.com	---	---
vim-5.7 000121	ncf	Jonathan J. Miner	jon.miner@doit.wisc.edu	---	---
vim-5.7 981206	nosyntax	Bram Moolenaar	Bram@vim.org	---	---
vim-5.7 000413	nroff	Matthias Burian	burian@grabner-instruments.com	---	---
vim-5.7 990614	objc	Valentino Kyriakides	1kyriaki@informatik.uni-hamburg.de	---	---
vim-5.7 991010	ocaml	Markus Mottl	mottl@miss.wu-wien.ac.at	000505!	http://miss.wu-wien.ac.at/~mottl/vim/syntax/ocaml.vim

vim-5.7 991104	opl	"Czo"	Olivier.Sirol@lip6.fr	---	---
vim-5.7 000614	ora	Sandor Kopanyi	sandor.kopanyi@altavista.net	---	---
vim-5.7 000311	pascal	Xavier Crégut	xavier.cregut@enseeiht.fr	---	---
vim-5.7 990614	pcap	Lennart Schultz	Lennart.Schultz@ecmwf.int	---	---
vim-5.7 990810	pccts	Scott Bigham	dsb@cs.duke.edu	---	---
vim-5.7 000613	perl	Nick Hibma	n_hibma@webweaving.org	---	---
vim-5.7 000105	php3	Lutz Eymers	ixtab@polzin.com	000808!	http://www-public.rz.uni-duesseldorf.de/~eymers/vim/syntax/pgp.vim
vim-5.7 991227	phtml	Lutz Eymers	ixtab@polzin.com	010117 ok	http://www-public.rz.uni-duesseldorf.de/~eymers/vim/syntax/phtml.vim
vim-5.7 980216	pike	Francesco Chemolli	kinkie@kame.usr.dsi.unimi.it	---	---
vim-5.7 991013	pine	David Pascoe	David.Pascoe@jtec.com.au	---	---
vim-5.7 000224	plsql	Jeff Lanzarotta	frizbeefanatic@yahoo.com !	---	---
vim-5.7 991214	po	Sung-Hyun Nam	namsh@kldp.org	---	---
vim-5.7 990703	pod	Scott Bigham	dsb@cs.duke.edu	---	---
vim-5.7 000606	postscr	Mike Williams	mrw@netcomuk.co.uk	---	---
vim-5.7 990614	pov	T. Scott Urban	urban@blarg.net	---	---
vim-5.7 980420	procmail	vacant! any takers?	NONE	---	---
vim-5.7 971228	prolog	Ralph Becket	rwab1@cam.sri.co.uk	---	---
vim-5.7 000406	ptcap	Haakon Riiser	hakonrk@fys.uio.no	---	---
vim-5.7 990614	purifylog	Gautam H. Mudunuri	gmudunur@informatica.com	---	---
vim-5.7 000403	python	Neil Schemenauer	nascheme@enme.ucalgary.ca	---	---

vim-5.7 981101	radiance	Georg Mischler	schorsch@schorsch.com	---	---
vim-5.7 980216	rc	Heiko Erhardt	Heiko.Erhardt@munich.netsurf.de	---	---
vim-5.7 990102	rcslog	Joe Karthauser	joe@freebsd.org	---	---
vim-5.7 981020	rebol	Mike Williams	mrw@netcomuk.co.uk	---	---
vim-5.7 991203	remind	Davide Alberani	alberanid@bigfoot.com	---	---
vim-5.7 000419	rexx	Thomas Geulig	geulig@nentec.de	---	---
vim-5.7 990423	rpcgen	Dr. Charles E. Campbell, Jr.	Charles.E.Campbell.1@gsfc.nasa.gov	???	???
vim-5.7 000219	ruby	Mirko Nasato	NONE	010117: 404!	http://altern.org/mn/ruby/ruby.vim
vim-5.7 000426	samba	Rafael Garcia-Suarez	garcia_suarez@hotmail.com	---	---
vim-5.7 990518	sas	Phil Hanna	pehanna@yahoo.com	---	updated on 010222
vim-5.7 980112	sather	Claudio Fleiner	claudio@fleiner.com	010117: ok	http://www.fleiner.com/vim/syntax/sather.vim
vim-5.7 980430	scheme	Dirk van Deun	dvandeun@poboxes.com	---	---
vim-5.7 980316	sdl	Harald Böhme	boehme@informatik.hu-berlin.de !	010117: 000717!	http://www.informatik.hu-berlin.de/~boehme/vim/syntax/sdl.vim !
vim-5.7 990702	sed	Haakon Riiser	hakonrk@fys.uio.no	---	---
vim-5.7 991210	sgml			000703	http://tritarget.com/vim/syntax/sgml.vim
vim-5.7 990918	sgmllnx	Sung-Hyun Nam	namsh@kldp.org	---	---
vim-5.7 000606	sh	Dr. Charles E. Campbell, Jr.	Charles.E.Campbell.1@gsfc.nasa.gov	???	???
vim-5.7 991220	sicad	Zsolt Branyiczky	zbranyiczky@lmark.mgx.hu	---	---
vim-5.7 990310	simula	Haakon Riiser	hakonrk@fys.uio.no	---	---
vim-5.7 990614	skill	Toby Schaffer	jtschaff@eos.ncsu.edu	---	---

vim-5.7 990616	sl	Dan Piponi	dan@tanelorn.demon.co.uk	---	---
vim-5.7 980216	slang	Jan Hlavacek	lahvak@math.ohio-state.edu	---	---
vim-5.7 980403	slrnrc	Preben "Peppe" Guldberg	c928400@student.dtu.dk	010117: 980706!	http://www.student.dtu.dk/~c928400/vim/syntax/slrnrc.vim !
vim-5.7 980402	slrnsc	Preben "Peppe" Guldberg	c928400@student.dtu.dk	010117: 980706!	http://www.student.dtu.dk/~c928400/vim/syntax/slrnsc.vim!
vim-5.7 990629	sm	Dr. Charles E. Campbell, Jr.	Charles.E.Campbell.1@gsfc.nasa.gov	???	???
vim-5.7 990902	smil			010117: 990805! vim-6.0s: 001104	http://www.helio.org/vim/syntax/smil.vim
vim-5.7 000520	sml	Fabrizio Zeno Cornelli	zeno@filibusta.crema.unimi.it	---	---
vim-5.7 000129	snnsnet	Davide Alberani	alberanid@bigfoot.com	010117: ok	http://members.xoom.com/alberanid/vim/syntax/snnsnet.vim
vim-5.7 000129	snnspat	Davide Alberani	alberanid@bigfoot.com	010117: ok	http://members.xoom.com/alberanid/vim/syntax/snnspat.vim
vim-5.7 000129	snnsres	Davide Alberani	alberanid@bigfoot.com	010117: ok	http://members.xoom.com/alberanid/vim/syntax/snnsres.vim
vim-5.7 990921	spec	Donovan Rebbechi	elflord@pegasus.rutgers.edu	---	---
vim-5.7 990812	spice	Noam Halevy	Noam.Halevy.motorola.com	---	---
vim-5.7 990707	spup	Stefan.Schwarzer	Stefan.Schwarzer@tu-clausthal.de	010117: ok	http://home.tu-clausthal.de/~svss/vim/spup.vim
vim-5.7 000104	sql	Paul Moore	gustav@morpheus.demon.co.uk	---	---
vim-5.7 991215	sqr	Jeff Lanzarotta	frizbeefanatic@yahoo.com !	---	---
vim-5.7 990614	squid	Klaus Muth	muth@hagos.de	010117: 981103! vim-6.0s: 001104	http://unitopia.uni-stuttgart.de/~monty/vim/syntax/squid.vim
vim-5.7 971224	st	Arndt Hesse	hesse@self.de	---	---
vim-5.7 000106	stp	Jeff Lanzarotta	frizbeefanatic@yahoo.com !	---	---

vim-5.7 000616	synload	Bram Moolenaar	Bram@vim.org	---	---
vim-5.7 981206	syntax	Bram Moolenaar	Bram@vim.org	---	---
vim-5.7 991022	tads	Amir Karger	karger@post.harvard.edu	010118: ok	http://www.hec.utah.edu/~karger/vim/syntax/tads.vim
vim-5.7 990128	tags	Dr. Charles E. Campbell, Jr.	Charles.E.Campbell.1@gsfc.nasa.gov	???	???
vim-5.7 980629	tcl	Allan Kelly	Allan.Kelly@ed.ac.uk	---	---
vim-5.7 000407	tex	Dr. Charles E. Campbell, Jr.	Charles.E.Campbell.1@gsfc.nasa.gov	???	???
vim-5.7 000614	texinfo	Sandor Kopanyi	sandor.kopanyi@altavista.net	---	---
vim-5.7 991227	tf	Lutz Eymers	ixtab@polzin.com	010117 ok	http://www-public.rz.uni-duesseldorf.de/~eymers/vim/syntax/tf.vim
vim-5.7 000322	tli	Kurt W. Andrews	kandrews@fastrans.net	---	---
vim-5.7 971216	tsalt	Sean M. McKee	mckee@misslink.net	---	---
vim-5.7 980217	uil	Thomas Koehler	jean-luc@picard.franken.de	---	---
vim-5.7 000208	vb	Robert M. Cortopassi	cortopar@mindspring.com	---	---
vim-5.7 000201	verilog	Mun Johl	mj@core.rose.hp.com	---	---
vim-5.7 970914	vgrindefs	Bram Moolenaar	Bram@vim.org	---	---
vim-5.7 000406	vhdl	"Czo"	Olivier.Sirol@lip6.fr	---	---
vim-5.7 000608	vim	Dr. Charles E. Campbell, Jr.	Charles.E.Campbell.1@gsfc.nasa.gov	???	???
vim-5.7 990819	viminfo	Bram Moolenaar	Bram@vim.org	---	---
vim-5.7 990222	vrml	Gregory Seidman	gseidman@acm.org	010118: 990731!	http://zing.ncsl.nist.gov/~gseidman/vim/syntax/vrml.vim
vim-5.7 990823	web	Andreas Scherer	andreas.scherer@pobox.com	---	---
vim-5.7 000213	webmacro	Claudio Fleiner	claudio@fleiner.com	010118: ok	http://www.fleiner.com/vim/syntax/webmacro.vim

vim-5.7 ---	whitespace	Bram Moolenaar ???	Bram@vim.org ???	---	---
vim-5.7 990819	winbatch	NoName	stephan@a2f-services.fr	---	---
vim-5.7 000226	wml	Gerfried Fuchs	alfie@innocent.com	010118: 000919!	http://alfie.ist.org/vim/syntax/wml.vim
vim-5.7 000408	xdefaults	Johannes Zellner	johannes@zellner.org	010118: ok	http://www.zellner.org/vim/syntax/xdefaults.vim
vim-5.7 990514	xmath	Dr. Charles E. Campbell, Jr.	Charles.E.Campbell.1@gsfc.nasa.gov	???	???
vim-5.7 990909	xml	Johannes Zellner	johannes@zellner.org	010118: 001216!	http://www.zellner.org/vim/syntax/xml.vim
vim-5.7 990705	xpm	Ronald Schild	rs@scutum.de	---	---
vim-5.7 991004	xpm2	Steve Wall	steve_wall@redcom.com	---	---
vim-5.7 990308	xs	Michael W. Dodge	sarge@pobox.com	---	---
vim-5.7 990319	xxd	Dr. Charles E. Campbell, Jr.	Charles.E.Campbell.1@gsfc.nasa.gov	???	???
vim-5.7 990408	yacc	Dr. Charles E. Campbell, Jr.	Charles.E.Campbell.1@gsfc.nasa.gov	???	???
vim-5.7 990107	z8a	Milan Pikula	www@fornax.elf.stuba.sk	---	---
vim-5.7 990614	zsh	Felix von Leitner	leitner@math.fu-berlin.de	---	---

---

NEW Syntax Files with VIM-6

The NEW syntax files with VIM-6, ie those which have not yet been included with Vim-5.7: (Yes, I have only just begun editing that. Anyone got a list of the new syntax files with Vim-6.0r?)

vim-6.? 001104 msql Lutz Eymers ixtab@polzin.com 010117 ok http://www-public.rz.uni-duesseldorf.de/~eymers/vim/syntax/mysql.vim

vim-6.? 001104 php Lutz Eymers ixtab@polzin.com 010117 ok http://www-public.rz.uni-duesseldorf.de/~eymers/vim/syntax/php.vim

---

Mergeable Syntax Files

The following syntax files can probably be merged with existing syntax files from the distribution:

dcl	Neil Schemenauer	nascheme@acs.ucalgary.ca	
haskell	Martin Norbäck	d95mback@dtek.chalmers.se	
man	Stephen Hack	shack@uiuc.edu	
scheme	Don Blaheta	dpb@cs.brown.edu	
slrnrc	Jan Hlavacek	lahvak+www@math.ohio-state.edu	
sml	Wil SuperDude	valinor@buf.adelphia.net	---
spice	Guenter Steinbach	gunter_steinbach@hp.com	
tcl	Robin Becker	robin@jessikat.demon.co.uk	
tcl	Micky Williamson	mickyw@dimensional.com	
tex	Scott Batchelor	sb25@ukc.ac.uk	

NEW Syntax Files - To Add !

New Syntax files which have been sent to me but not to Bram, apparently.

010114: Sent them all to Bram. Let's see whether he will add them.

bash	Donovan Rebbechi	elflord@pegasus.rutgers.edu	http://pegasus.rutgers.edu/~elflord/vim/syntax/bash.vim
cbg	Kent Nassen	knassen@umich.edu	http://www-personal.umich.edu/~knassen/vim/cbg.vim
ch	Robert Brady	robb@datatone.com (bounce!)	http://www.datatone.com/~robb/vim/syntax/def.vim
docbook	Ray Dassen	jhm@cistron.nl 991208	http://www.wi.leidenuniv.nl/~jdassen/onderwijs/stuva/debug/docbook.vim
dylan	Justus Pendleton	justus@acm.org	http://110.net/~pq1662/
embperl (Embedded Perl)	Steve Willer	willer@interlog.com	http://www.interlog.com/~willer/embperl.vim
hercules	Dana Edwards	Dana_Edwards@avanticorp.com	
htmlos	Jason Rust	jrust@westmont.edu	http://www.rustyparts.com/vim/syntax/htmlos.vim
lpc	Klaus Muth	muth@hagos.de	http://unitopia.mud.de/~monty/vim/syntax/lpc.vim
makemenu	Sean McKee	mckee@misslink.net	http://www.misslink.net/mckee/vim/rc/makemenu.vim
mgp (MagicPoint)	Gerfried Fuchs	alfie@innocent.com	http://alfie.ist.org/vim/syntax/mgp.vim
pli	Neil Schemenauer	nascheme@acs.ucalgary.ca	http://www.enme.ucalgary.ca/~nascheme/vim/syntax/pli.vim
powerb (PowerBuilder export files)	Ken Shan	ken@digitas.harvard.edu	http://www.pcisys.net/~kscott/vim/syntax/pb.vim
proc	Austin Ziegler	austin.ziegler@solect.com	---
progress (Progress 4GL)	Mikhail Kuperblum	mikhail@whasup.com	---

rosisql (rosi SQL 4GL by Halstenbach ACT)	Klaus Muth	muth@hagos.de	http://unitopia.mud.de/~monty/vim/syntax/rosisql.vim
sinda (sinda85, sinda/fluint input file)	Adrian Nagle	vim-www@naglenet.org	http://naglenet.org/vim/syntax/sinda.vim
slrnsl (Slrn macro file)	Preben Guldberg	c928400@student.dtu.dk	http://www.student.dtu.dk/~c928400/vim/syntax/slrnsl.vim
sps (SPSS command files)	Kent Nassen	knassen@umich.edu	http://www-personal.umich.edu/~knassen/vim/sps.vim
sqlforms (SQL forms)	Austin Ziegler	austin.ziegler@solect.com	---
tak (TAK2, TAK3 thermal modeling input file)	Adrian Nagle	vim-www@naglenet.org	http://naglenet.org/vim/syntax/tak.vim
trasys (TRASYS input file)	Adrian Nagle	vim-www@naglenet.org	http://naglenet.org/vim/syntax/trasys.vim
tklatex (VIM macros for Latex)	Tomer Kol	tkol@psl-palm.technion.ac.il	http://tiger.technion.ac.il/~tomer/TeX/VIM/TKlatex.vim
vera	Beth Leonard	beth@slimy.com	http://frost.slimy.com/~beth/vim/syntax/vera.vim
vrml2 (VRML97)	Gregory Seidman	gseidman@zing.ncsl.nist.gov	http://zing.ncsl.nist.gov/~gseidman/vim/syntax/vrml.vim

---

Wanted Syntax Files

These people have asked for a sytanx file - can you help them?

```
language                bind4 (nameserver)      bind8 (nameserver)      cisco                   ircII                   lclint                  mathematica             metahtml                ml                      troff                   xwsh
Person                  Rev M Lewellyn          Rev M Lewellyn          Lars Marowsky-Brée      Juhapekka Tolvanen      Harsh                   Thomas Fieseler         Tibor Lorincz           Don Blaheta             [several people]        Rudy Wortel
Email
root@lewellyn.lewellyn.org
root@lewellyn.lewellyn.org
lmb@pointer.teuto.de
juhtolv@st.jyu.fi
harsh_inf@my-deja.com [001010]
fieseler@biomag.uni-jena.de
clt@kiss.sk
dpb@cs.brown.edu

rudy@aw.sgi.com
```

Fellow Linguists

The following people have offered to be "fellow linguists", ie you can ask them for help when you need help on some of the available languages:

```
haskell
        Martin Norbäck          d95mback@dtek.chalmers.se
        Neil Schemenauer        nascheme@acs.ucalgary.ca
perl
```

000724 Doug Hanks dhanks@resourcephoenix.com

Would you like to become a fellow linguist, too? Then please send me a mail, so I can add you to the list! :-)

URL: http://www.math.fu-berlin.de/~guckes/vim/lang.html
URL: http://www.vim.org/lang.html (mirror)
Created: Thu Jul 23 18:00:00 CEST 1998

Send feedback on this page to
Sven Guckes guckes@vim.org

VIM - Utilities

Info on some really useful scripts, tools, and utilities for VIM.

Got a good program/script/tool/utility for use with VI or VIM? Let me know!

Programs distributed **with** Vim:

- ctags
- xxd

Programs distributed **without** Vim:

- boxes
- global
- jtags
- makehtml and
- maketags
- par
- tal
- tblfmt
- vmh
- wf
- zip

Scripts:

- vim2html

VIM - Utilities

The *Vim Utilities* are programs and scripts which are distributed *with* Vim. So when you get the archive file then you will find these programs with Vim:

Overview: ctags | xxd

ctags [000424,010416]

> Description, short: "Exuberant Ctags" creates "tag files" for emacs, vi, and clones - and a lot better than the standard "ctags"! [If you do not know what a tag is then please take a

look at the page about vim and tags.]

Description, long: Ctags generates an index (or "tag") file of programming language objects of source code Ctags allows these items to be quickly and easily located by a text editor or other utility.

Ctags supports these languages: Assembler, AWK, ASP, BETA, Bourne/Korn/Zsh Shell, C, C++, COBOL, Eiffel, Fortran, Java, Lisp, Make, Pascal, Perl, PHP, Python, REXX, Ruby, S-Lang, Scheme, Tcl, and Vim.

Alternatively, Ctags can generate a cross reference file which lists, in human readable form, information about the various objects found in a set of C language files.

Tag index files are supported by the vi(1) editor and its derivatives (such as vim, elvis, stevie, and xvi) through the use of the :ta command, and by the emacs editor through the use of the Meta-. command, both of which locate the object associated with a name appearing in a source file and jump to the file and line which defines the name.

Source and binaries for Linux Red Hat, MS-DOS (16-bit and 32-bi DPMI), Win32 (Windows95 and WindowsNT), and OS/2.

Latest release version: ctags-5.0.1 [010416]

Author: Darren Hiebert darren@hiebert.com

URLs:

http://ctags.sourceforge.net
http://sourceforge.net/projects/ctags/
http://www.appwatch.com/Linux/App/620/data.html

http://darren.hiebert.com/ctags/index.html [sic!] [980727]

Source: ftp://ftp.revnet.com/pub/ctags/ [980727]

xxd [961007,990915]

Description: Hexadecimal converter (hexdumper).

Size: ~14K

Author: Jürgen Weigert jw@cs.fau.de

Source: ftp://ftp.uni-erlangen.de:21/pub/utilities/etc/xxd-1.10.tar.gz

---

VIM - Additional Utilities

The following utilities are *not* distributed with Vim. However, they can be very very helpful. So you *should* have them!

Overview: Programs distributed **without** Vim:

- boxes
- global
- jtags
- par
- tal
- tblfmt
- vmh
- wf
- zip

boxes [990822,990824]

Draw an "box" around some given text.

Description: "boxes" can draw all kinds of boxes around its input text, ranging from a C comment box to complex ASCII art. These boxes may also be removed, even if they have been badly damaged by editing of the text inside. Since boxes may be open on any side, boxes can also be used to create regional comments in any programming language. With the help of an editor macro or mapping, damaged boxes can easily be repaired. New box designs of all sorts can easily be added and shared by appending to a free format configuration file.

Size: ~70K

Author: Thomas Jensen JensenThomas@web.de boxes@home-of.tj

Distribution: http://home.pages.de/~jensen/boxes/

Latest Version: 1.0 [990822]

Examples can also be found on the boxes home page:

http://home.pages.de/~jensen/boxes/examples.html

cowsay and cowthink [000616,010220]

http://www.nog.net/~tony/warez/cowsay.shtml

Author: Tony Monroe tony@nog.net

$ cowsay moo! _____ < moo! > ------ \ ^__^ \ (oo)_______ (__)\ )\/\ ||----w | || ||

figlet

global [980713,990726]

"GLOBAL common source code tag system"

http://www.tamacom.com/global/

Description: http://www.tamacom.com/unix/index.html#global

Author: Shigio Yamaguchi shigio@tamacom.com

Latest version: global-3.44 [980705]

jtags [980630,990621]

Description, short: "JTags" creates "tag files" from Java sources.

Description, long: "JTags" creates "tag files" from Java sources, including classes, interfaces, constant class and interface members, class members, class and interface methods.

Home Page and Distribution:

http://www.fleiner.com/jtags/

Author: Claudio Fleiner claudio@fleiner.com

Latest version: jtags-0.2 [990614]

makehtml and maketags

These two awk scripts convert the vim documentation (aka helptexts) to HTML, so they can be viewed on WWW.

For more info see the entry on the awk page.

par - paragraph formatter [971208,010317]

Home Page: http://www.cs.berkeley.edu/~amc/Par/

Author: Adam M. Costello http://www.cs.berkeley.edu/~amc/

Latest releases: par-1.51 [000224] and par-1.50 [960211] (only slight difference)

Description: par reformats paragraphs preserving borders on either side. For examples of input and output see the page

http://www.cs.berkeley.edu/~amc/Par/par.doc.txt.gz.

NOTES: Why use par at all when there is the builtin formatting? Well, the builtin formatting cannot right-justify text.

Example: Formatting the current paragraph with "par" is as easy as this: map #p vip!par<CR>

tal - "trailer alignment" [971122,971128,990729]

Description: "tal" aligns the end of lines if they should have common characters at the end of lines. This is very helpful when rearranging the right side of a "comment box",

Size: ~10K

Author: Thomas Jensen tsjensen@cip.informatik.uni-erlangen.de

Distribution: http://home.pages.de/~jensen/tal/.

Latest version: tal-1.9 [990311]

Note: The program "tal" is not yet part of the Vim distribution. Bram probably still does not know about it - but he is so busy right now that I don't want to bother him. Anyway - Thomas is considering a DOS port of this nice utility. Encourage him! I do hope that someone will take the challenge to work in this text formatting utility into Vim builtin "text formatting". Any takers?

Latest change: Now runs on SunOS4! :-)

Let me show you how this works: Edit some text:

```
foo bar blah
blah blah blah
tralala tralala
```

Let's assume you want to put a comment box around the lines. Adding the start and end of the comment is easy:

```
/**
* foo bar blah *
* blah blah blah *
* tralala tralala *
******************
```

Note: The last line shows how big the box should get. Now add some text to the end of the lines you want to put into a box - highlight the lines, press ':', then enter the command `:s/$/ */`:

```
/**
* foo bar blah *
* blah blah blah *
* tralala tralala *
******************
```

Now apply "tal" those lines - `:'<,'>!tal`

```
/*                  *
* foo bar blah      *
* blah blah blah    *
* tralala tralala   *
******************* *
```

Now, all you need to do is change the added space to complete the box:

```
/******************
* foo bar blah    *
* blah blah blah  *
* tralala tralala *
******************/
```

I do hope that this can be added to Vim's internal text formatting.

tblfmt [990729]

Description, short: table formatter

Description, long: Creates a table from comma separated input. Separators can be adjusted. Uses AWK.

Author: James Cribb jamesc@zip.com.au

Example:

```
input:
Stadt Land Fluss
Berlin Berlin Spree
Heidelberg Baden-Württemberg Neckar
Bonn who-cares Rhein
München Bayern Isar

output:
Stadt       Land               Fluss
Berlin      Berlin             Spree
Heidelberg  Baden-Württemberg  Neckar
Bonn        who-cares          Rhein
München     Bayern             Isar
```

Source: http://www.zip.com.au/~jamesc/tblfmt (local copy)

vmh [990618]

Description, short: "vmh" - the Vim Mail Handler

Description, long: "vmh" is a mail handler useful to read your email, write email and manage emails all from within Vim.

Home Page and Distribution: http://www.fleiner.com/vmh/

Author: Claudio Fleiner claudio@fleiner.com

Latest version: vmh-0.2 [990618]

wf (word format) [960818]

Description: "wf" is a small Perl script that reformats text preserving the standard quote prefix "> " (angle and space). You can use it with VIM as a filter. [But then again there is the command 'Q'. :-)]

Source: wf.pl

Size: less than 2K

Author: Bek Oberin gossamer@penguin.glasswings.com.au

zip [960602]

Compression.

There are several versions for "zip" compression. An interesting one seems to be the one

by "Infozip":

ftp://ftp.uu.net/pub/archiving/zip.

Dictionaries

Several people were once looking for dictionaries so they could make better use of spelling checkers or just simple (auto)completion. Here is a site:

- ftp://ftp.fu-berlin.de/unix/security/dictionaries/

Quite some dictionaries at that site. Anyone know of a better one?

Scripts

vim2html.pl

vim2html is a perl script that converts vim-5 helpfiles to html.

Links

Chip Campbell's Utilities for Vim [000425]

http://www.vim.org/user.html#campbell

http://www.erols.com/astronaut/vim/

LOTS of utilities! :-)

Cygwin [..,010615]

http://sources.redhat.com/cygwin/

http://cygwin.com/faq/

"The Cygwin tools are ports of the popular GNU development tools and utilities for Windows. They function by using the Cygwin library which provides a UNIX-like API on top of the Win32 API."

Unix for Windows by AT&T

http://www.research.att.com/sw/tools/uwin/

These are *not* OpenSource!

Virtually Unix [..,010619]

http://virtunix.itribe.net/

Virtually Un*x is dedicated to helping bring the power of UN*X tools to Windows95. Maintained by Chris szurgot@itribe.net (The whatsnew page says "last update on June 20th 1997"...)

Native Win32 ports of some GNU utilities

http://www.edv.agrar.tu-muenchen.de/~syring/win32/UnxUtils.html

Native Win32 ports of some GNU utilities

Unix shell ports for Win32

tcsh: ftp://ftp.blarg.net/users/amol/tcsh/

zsh: ftp://ftp.blarg.net/users/amol/zsh/

terminal programs for win32

See my page on windows.

```
URL:          http://www.math.fu-berlin.de/~guckes/vim/util.html
URL:          http://www.vim.org/util.html (mirror)
Created:      Thu Jan 01 00:00:00 CET 1998
```

Send feedback on this page to
Sven Guckes guckes@vim.org

VIM Challenges

"Can you do it?" - Some challenges on hacking with Vim.

Every now and then I think about some problem and I think it can be done with a simple mapping. But I am not always pleased with my solution. Maybe you can think of a better solution? Let me know!

Collecting Matches

Goal: Scan current buffer for occurrences of a word and put all matching lines in a new buffer. [Assume that the word is in the unnamed register "".]

Analysis: This is a job for the "global" command. Now, we could easily copy all matching lines after the end of the current buffer with `g:#foo#co $`, but creating a new buffer to display the result is nice - and also easy done with `:new`. A mark to remember the former end-of-buffer is necessary for deleting the resulting lines, of course. The last two commands simply delete the resulting empty line.

Solution: map ## Gma :g#<c-r>"#co $<cr> 'a j dG :new<cr> p k dd I am not happy with the result - for these reasons:

- extra mark required
- delete command to "move result"
- cleanup commands

I'd rather append the matches to some unused named register and have the pasted into the new edit buffer. Is that possible with an ex command?

000505: Yes - it certainly is possible with an ex command: map ## :g#<c-r>"#y A<cr> :new "Ap The ex command *y A* appends every match to the register 'A'. (Note the space between the command an the register name here. Otherwise vim would see the command "yA" which is, of course, no an ex command.)

Now all that is missing is a simple emthod to make sure that the register A is cleared before using the mapped commands. Anyone?

000515: Frank Baruch fbaruch@pacific.net.sg pointed me at the Vim's "let" command which allows to fill the contents of a register: let @a=""<cr> Adding this will give us the following mapping: map ## let @a=""<cr> :g#<c-r>"#y A<cr> :new "Ap Or in one line: map ## :let @a=""<CR>:g#<C-R>"#y A<CR>:new<CR>"Apdk0 Unfortunately, this solution is specific to Vim. Does anyone have a Vi solution, ie a solution using Vi commands only?

Comments? Improvements? Send them to me!

```
URL:          http://www.math.fu-berlin.de/~guckes/vim/chall.html
URL:          http://www.vim.org/chall.html (mirror)
Created:      Sun Oct 08 12:00:00 MET DST 2000
```

Send feedback on this page to
Sven Guckes guckes@vim.org

VIM Quotes

Some quotes of the email I have received about Vim:

010403:

```
* A.R. Thornton art27@hermes.cam.ac.uk [010402 00:39]:
> I once came across this:
> vi vi vi - the editor of the beast
> vim vim vim - the editor of
> the *all* *NEW* improved beast
```

000808: Arthur Klassen ansak@home.com sent me his religious statement:

```
"I have always avoided religious wars over this or that toolset,
this or that OS. I've used all kinds of different editors:
IBM's PE, BRIEF by UnderWare, MPW (on the Mac), CodeWright
and various iterations of MS' Integrated environment.
Recently I discovered VIM. The effect has been devastating.
I am now a True Believer.  There is only one editor,
all the rest are only hollow imitiations of VIM!"
```

000505: Jan-Benedict Glaw jbglaw@lug-owl.de commented:

```
>  E_scape _M_eta _A_lt _C_ontrol _S_hift

_V_ariety _I_s _M_agic
```

[000225] There are quite a few comments on vim-5.4 on LinuxCare. Check them out!

I liked these:

```
On 04-FEB-00, Junior wrote:
Vim is a REAL man's text editor.
I don't know why anyone else would
even bother with sissy programs
like emacs, or even worse...  pico.

On 03-FEB-00, g kporku wrote:
To do:
1. Get VIM for Dos
2. Get VIM for Linux
3. Get Vim for Freebsd
4. Get Vim for Dec-Unix
5. Get VIM for Irix
6. Laugh at non-VIM users

On 22-JAN-00, Tobiah wrote:
Vim is like the sound of one monolithic corporation collapsing

On 20-AUG-99, Like it shows names anyway... wrote:
```

```
Let's face it you don't use vi for the interface or the
features or any of the other things that are on this poll.
You use it because like Buckaroo Bonzai said "wherever you
go, there is vi". No serious sysadmin uses emacs because
they all use vi, because it's everywhere.  I'm forced to
underate vi because of the way this system is set-up,
but it's the only one on the list that I use anymore.
```

Emacs is a nice OS - but it lacks a good text editor. That's why I am using Vim. --Anonymous

---

From: tottinge@concentric.net (Tim Ottinger) Newsgroups: comp.editors Date: 06 Jan 2000 08:44:49 EST Message-ID: <38749e1a.87614312@news.concentric.net> Vim is the editor that VI wants to be when it grows up. I fell into it immediately, no issues, no hassles. I've spent a few idle hours here and there learning more of the extensions and coolness, the gui, etc. It's powerful, and it's as VI as you can get. I would rather than vim than just about any editor around, and have been 'winning converts' at work. It's amazing. I wouldn't bother looking at other vi clones. Honestly. I've tried a few, but VIM is head-and-shoulders above the rest.

From: "Jeff Susanj" <jeffrey.l.susanj@boeing.com> Message-ID: <FoyAw9.MC4@news.boeing.com> I am fairly new to Linux (and UNIX) but for text editing on a terminal I have found vi to be much simpler to use. When using Linux, Windows, CP/M, Commodore GEOS 128 etc. I can't remember all of the meta key stuff for Emacs even though I have used it off and on for some time. Once I have mastered the basics I can always get the job done in vi. Now all I need to do is learn the expert stuff.

From: Scott Taylor <gstaylor@netscape.net> Message-ID: <388E1173.FCFB6CD3@netscape.net> I still remember when I was first introduced to vi, I thought the system administrator was crazy. Now the table is turned, vi is my friend and everyone else thinks I'm crazy. At least until they get over the fact, that they don't have to keep reaching for the damned mouse.

Bram Moolenaar [991202 12:10]:

```
The difference between the Vim mug and the
Vim tub is that the tub has the text inside. :-)
```

Geoff 'mandrake' Harrison mandrake@mandrake.net [990909 21:08]:

```
Horms and I discussed this, and we came up with two categories of people.
People who use vim, and people who don't realize that they could be using vim.
```

Patrick Baker patrick.baker@sabre.com [990827 comp.editors]

```
My ears hear 'modeless is better', but my heart pounds 'Vim rules'
```

Thierry Stoehr stoehr@club-internet.fr [990802 22:53]

```
"VIM software : Very Important & Marvelous software."
```

Ron Belcher (rbelcher@asapsc.com) [990617 15:10]:

```
I am a contract programmer and one of the first things
I always do on a new job is to get VIM on my machine.
```

Andrew J Davies (Andrew.J.Davies@aib.ie) [990611 16:47]:

```
Subject: How did I work without it

Recently downloaded vim 5.3.
```

```
I have been using vi for many years.
Only now have I realised what I've been missing.
Thanks for making my job more interesting and easier.
```

Karl & Leanne Norrena [990420 07:51]:

```
I think I am going to cry. This is what I have been looking for for quite
some time. I was using an old version of a vi editor for DOS offered by MKS.
It was awfull but more usefull than any Windows alternative.
Now I have VIM.  Wow what a great find.
```

Timothy Johnson tjohnson@akcache.com [990310] in private email:

```
On a serious note, I just got Visual C++ 6.0 and installed it. I intend to
use it to compile Vim for Python in the Windows Environment. I would be
happy to set up a site on that issue. Vim for windows is a delight to work
in. It is really the best of both the command-line world and the GUI world.
```

David Gerard fun@thingy.apana.org.au [990306,news.software.readers]:
Message-Id: <7bqqse$41b$4@youknow.apana.org.au>

```
Last night I finally unzipped vim 5.3, and dammit I'm a convert. ..
gvim is the One True Way for text editing on a Windows box,
and that's all there is to it. All hail! All hail!
```

David maintains a page on slrn and vim on Windows.

Edwin van Geelen edwin@mindless.com [990308,comp.editors]:

```
Vim is worth millions and costs nothing. It's the best editor the world has
ever seen. Using the trio Vim/Perl/LaTeX I can dispose of most other software.
```

Nick Popoff nick@bloodletting.com [990304] (in email):

```
I've been hearing so many great things about VIM from friends all year,
and finally now I'm trying it out, and I wish I'd taken their advice sooner!
As a programmer working in both Unix and Windows, it's great to finally
find a fantastic editor for both.  Thanks for all your hard work!
```

Terry Mathews tmathews@erinet.com on news.software.readers [990228]
Message-Id: <7bcit1$2gn$1@news.erinet.com>

```
I was a Win98 junkie and addicted to Outlook Express. SLRN has converted me.
After I found a key refrence to VIM and figgered out how to hit ^C,
then :write and :quit, it was mucho superio to OE. God, I love BeOS. Plus,
now I can set my computer up as a telnet server, connect from school
and read my newsgroups.  Awesome. :)
```

```
page: http://www.csclub.uwaterloo.ca/u/mvcorks/vim/
link: http://www.tuxedo.org/~esr/jargon/html/entry/zapped.html
text: "What can I say? Vi is Zen."
```

Last not least - a search on Dictionary.com reveals this about Vim:
http://www.dictionary.com/cgi-bin/dict.pl?db=*&term=vim Nancy McGough nancym@ii.com [991209] consulted

her dictionary and found:

```
        vim: ebullient vitality and energy
             ebullient:  boiling, agitated

3 definitions found:

vim \Vim\, n. [L., accusative of vis strength.]
        Power; force; energy; spirit; activity; vigor.  [Colloq.]
        Source: Webster's Revised Unabridged Dictionary

vim n   1: a healthy capacity for vigorous activity;
        "jogging works off my excess energy";
         "he seemed full of vim and vigor" [syn: energy, vitality]
        2: an imaginative lively style (especially style of writing);
        "his writing conveys great energy" [syn: energy, vigor, vigour]
        Source: WordNet . 1.6

vim     1. Vendor Independent Messaging.
        2. vi Improved (previously vi iMitation),
        An improved version of vi, available for many platforms.
        VIM allows multiscreen editing, more flexible insert/command mode
        handling, better C indentation and much more.
        FAQ (http://www.grafnetix.com/~laurent/vim/faq.html).  (1997-10-04)
        Source: The Free On-line Dictionary of Computing
```

```
URL:         http://www.math.fu-berlin.de/~guckes/vim/quotes.html
Created:     http://www.vim.org/quotes.html (mirror)
Created:     Wed Mar 03 12:00:00 CET 1999
```

Send feedback on this page to
Sven Guckes guckes@vim.org

```
" ================================================================
" File:         $HOME/.vimrc
" Last update:  Wed Jun 13 17:00:00 MEST 2001
" Purpose:      *Personal* Setup file for the editor Vim (Vi IMproved)
"               It contains everything which I use *personally*.
" Author:       Sven Guckes guckes@vim.org (guckes@math.fu-berlin.de)
"               <URL:http://www.math.fu-berlin.de/~guckes/sven/>
" ================================================================

" ================================================================
" Mind you, this file is only a setup file for *personal* use.
" The BIG setup file which I created for *all* users ("vimrc.forall")
" is available in my setup file directory:
" http://www.math.fu-berlin.de/~guckes/setup/vimrc.forall    (uncompressed)
" http://www.math.fu-berlin.de/~guckes/setup/vimrc.forall.gz (compressed)
" Enjoy!  Feedback is very welcome!  :-)
" ================================================================

" ====================================
" Loading general setup files first
" ====================================

" The PCs at math.fu-berlin.de run WindowsNT-4 and
" the home directory is mounted on drive Z: -
" so when you start up Vim on one of those machines
"
" if has("dos16") || has("dos32") || has("gui_win32")
" naaah.  I don't use that DOS version. ;-)
"
" if has("gui_win32")
"   " Source the setup file for all users:
"   source Z:\\.vimrc.forall
"   " About 42 lines fit nicely on that screen:
"   set lines=42
" endif
"
  if has("unix")
    " Source the setup file for all users:
    source ~guckes/.vimrc.forall
  endif

" ==================
" Settings of Options
" ==================

"     list & listchars
  set list   listchars=tab:»·,trail:·
" set list   listchars=tab:¯ú,trail:ú
" TN3270 does not show high-bit characters:
" set listchars=tab:>.,trail:o

"    Turn on HighLightSearching:
" se hls
" sometimes I need this, sometimes I dont...

"     wildmenu!  this makes use of the command lien to show
"     possible macthes on buffernames and filenames - yay!
  set wildmenu

" =============
```

```
" Autocommands
" ==============

" When editing HTML files (aka webpages)
" expand the "keywords" by characters "colon" and "slash"
" so you can expand URL prefixes as "words", eg
" http://www.math.fu-berlin.de/~guckes/vim/  ;-)
  au FileType html set isk+=:,/,~

" Silly test message to check the filepattern for message files:
" au BufRead .followup,.article,.letter,mutt* echo "Editing Messages\!"

" au BufCreate * set term=vt220
" au BufCreate * set term=ansi
" map ''' :set term=vt220<cr>:set term=ansi<cr>

" setting initial position after reading a file into a buffer:
" au BufReadPost * normal 2G10|

" ==============
" Abbreviations
" ==============

" ABbreviations
" personal addresses:
  ab MYUSERNAME guckes
  ab MYDOMAIN   math.fu-berlin.de
  ab MYMAIL     guckes@math.fu-berlin.de
" ab MYHOMEPAGE http://www.guckes.net/
  ab MYHOMEPAGE http://www.math.fu-berlin.de/~guckes/

" A frequently used URL for my trip to the USA:
" iab HPC   http://www.math.fu-berlin.de/~guckes/california/
" iab HPLT  http://www.math.fu-berlin.de/~guckes/linuxtag2001/
" iab HPLTC http://www.tu-chemnitz.de/linux/tag/lt3/
  iab HPLTB http://braunschweiger.linuxtag.de
  iab HPLTM http://www.mdlug.de/index.php3/linuxtag2001/

" a pseudo "From_" line for files in "mailbox" format:
  iab Mfrom From guckes@math.fu-berlin.de Thu Apr 06 12:07:00 1967
" Sometimes I need this to fix broken headers in "mailbox" files.

" ==============
" Colorization
" ==============

" Colorize some default highlight groups
" see ":help highlight-default"

"    Comments: Colorizing the "comments" (see ":help comments").
"    cyan on white does not look good on a black background..
" hi comment                          ctermfg=cyan   ctermbg=black
" hi comment                          ctermfg=cyan   ctermbg=7

" hi Cursor
" hi Directory
" hi ErrorMsg
" hi FoldColumn
" hi Folded
" hi IncSearch
```

```
"    LineNr:  Colorize the line numbers (displayed with "set number").
"    Turn off default underlining of line numbers:
  hi LineNr       term=NONE cterm=NONE

" hi ModeMsg
" hi MoreMsg

" coloring "nontext", ie TABs, trailing spaces,  end-of-lines,
" and the "tilde lines" representing lines after end-of-buffer.
  hi NonText      term=NONE cterm=NONE ctermfg=blue   ctermbg=black

"    Normal text:    Use white on black.
" hi normal ctermfg=white ctermbg=black   guifg=white guibg=black
" Oops - this overrides the colors for the status line - DANG!

" hi Question

"    Search: Coloring "search matches".  Use white on red.
  hi search  ctermfg=white ctermbg=red     guifg=white guibg=red

" hi SpecialKey

"    statusline:  coloring the status line
  hi StatusLine   term=NONE cterm=NONE ctermfg=yellow ctermbg=red
  hi StatusLineNC term=NONE cterm=NONE ctermfg=black  ctermbg=white

" hi Title
" hi VertSplit
" hi Visual
" hi VisualNOS
" hi WarningMsg
" hi WildMenu

" Other Groups:

"    Normal:  Coloring the text with a default color.
  hi normal       term=NONE

" ==============
" Mappings
" ==============

" Attribution Fixing: from "Last, First" to "First Last":
  map ,ATT gg}jWdWWPX

" ============================================================
" Options() - used to display some important option values
" within the status line (see below at "set statusline".
" ============================================================
"
" Statusline without colors and display of options -
" but with percentage at end:
" set statusline=Vim-%{Version()} [%02n]\ %(%M%R%H%)\ %F\ %=<%l,%c%V>\ %P
"
" Damien WYART <wyart@iie.cnam.fr> [000329]:
" set statusline=%<%f%=\ [%1*%M%*%n%R%H]\ \ %-25(%3l,%c%03V\ \ %P\
(%L)%)%12o'%03b'
" hi  User1 term=inverse,bold cterm=inverse,bold ctermfg=red
```

```
fu! Options()
"           let opt="Opt:"
            let opt=""
  " autoindent
" if &ai|   let opt=opt." ai"   |else|let opt=opt." noai"   |endif
  if &ai|   let opt=opt." ai"   |endif
  "  expandtab
" if &et|   let opt=opt." et"   |else|let opt=opt." noet"   |endif
  if &et|   let opt=opt." et"   |endif
  "  hlsearch
" if &hls|  let opt=opt." hls"  |else|let opt=opt." noet"   |endif
  if &hls|  let opt=opt." hls"  |endif
  "  paste
" if &paste|let opt=opt." paste"|else|let opt=opt." nopaste"|endif
  if &paste|let opt=opt." paste"|endif
  "  shiftwidth
  if &shiftwidth!=8|let opt=opt." sw=".&shiftwidth|endif
  "  textwidth - show always!
                    let opt=opt." tw=".&tw
"   let opt=opt."\[".&lines.",".&columns."\]"
  return opt
endf

" ============================================================
" Colorizing that status lines!  Whee!  :-)
" ============================================================
"
" Statusline without colors and display of options -
" but with percentage at end:
" set statusline=Vim-%{Version()} [%02n]\ %(%M%R%H%)\ %F\ %=<%l,%c%V>\ %P

" set   statusline=Vim-%{Version()}\ %{getcwd()}\ \ %1*[%02n]%*\ %(%M%R%H%)\
%2*%F%*\ %=%{Options()}\ %3*<%l,%c%V>%*
" Text between "%{" and "%}" is being evaluated and thus suited for functions.
" Here I will use the function "Options()" as defined below to show the
" values of some (local) options..
" The strings "%N*" unto "%*" correspond to the highlight group "UserN":
"       User1: color for buffer number
" hi    User1 cterm=NONE    ctermfg=red    ctermbg=white  guifg=red
guibg=white
"       User2: color for filename
" hi    User2 cterm=NONE    ctermfg=green  ctermbg=white  guifg=green
guibg=white
"       User3: color for position
" hi    User3 cterm=NONE    ctermfg=blue   ctermbg=white  guifg=blue
guibg=white

fu! Version()
        return version
endf

" ============================================================
" some things I need only temporarily for editing some stuff:

" SAP requires a space between "SAP" and "DB".. *sigh*
  iab SAPDB SAP DB

" see openwebschool.de
  iab YOWS  OpenWebSchool
```

```
" visual mode:  'p' to replace current text
" with previous copied/deleted text: [010126]
  vmap p d"0P

" ============================================================
" Final words...
" The last line is allowed to be a "modeline" with my setup.
" It gives vim commands for setting variable values that
" are specific for editing this file.  Used mostly for
" setting the textwidth (tw) and the "shiftwidth" (sw).
" Note that the colon within the value of "comments"
" needs to be escaped with a backslash!
"       vim:tw=70 et sw=4 comments=\:\"
```

```
" ================================================================
" File:         $HOME/.gvimrc
" Last update:  Thu May 10 10:10:10 MEST 2001
" Purpose:      *Personal* Setup file for GVim -
"               the GUI version of the editor Vim (Vi IMproved)
" Author:       Sven Guckes guckes@vim.org (guckes@math.fu-berlin.de)
"               <URL:http://www.math.fu-berlin.de/~guckes/sven/>
" Availability:      http://www.math.fu-berlin.de/~guckes/vim/gvimrc
" Enjoy!  Feedback is very welcome!  :-)
" ================================================================

" ================================================================
" SETTINGS!
" ================================================================
" Set some nice settings - recommended!
" set background=dark backspace=2 cindent esckeys hidden
" set ruler showmatch smarttab shiftwidth=4 tabstop=8
" Or shorter:
" set bg=dark bs=2 cin ek hid paste ru sm sta sw=4 ts=8
" What these settings do?  Well - RTFM!  ;-)

" Allow use of mouse for all modes
" (normal, insert, visual, command line):
" set mouse=a

" Set a specific font:
" set guifont=courier_new:h9

" Set windows size by columns and lines directly:
" set columns=90 lines=30

" ================================================================
" COLORS!
" ================================================================
" turn on syntax coloring:
  syntax on

" set the color of normal text:
" hi Normal guibg=Black guifg=White
  hi Normal guibg=black guifg=grey


" ================================================================
" AUTOCOMMANDS!
" ================================================================
" au BufRead  *.java set cindent

" Position the window at the top left corner on startup:
  au GUIEnter * winpos 0 0

" Show the current filename in the title:
  au BufEnter * let &titlestring=expand("%:p:~")

" EOF!
```

```
version 6.0
" See ":help version" to see why this command is in the first line.
"
" Let people know that they are using this setup file:
" echo "Welcome to Sven's setup file for all users!"
"
"
" ================================================================
" File:         $HOME/.vimrc.forall  (sourced by ~USER/.vimrc)
" Last update:  Thu May 10 21:55:31 MEST 2001
" Purpose:      Setup file for the editor Vim (Vi IMproved)
" Availability: This file is available as
"    27K        <URL:http://www.math.fu-berlin.de/~guckes/setup/vimrc.gz>
"    76K        <URL:http://www.math.fu-berlin.de/~guckes/setup/vimrc>
"               <URL:http://www.vim.org/rc> (mirror)
" Size:         This file is about 76K in size and has 2,000+ lines.
" Author:       Sven Guckes guckes@vim.org (guckes@math.fu-berlin.de)
"               <URL:http://www.math.fu-berlin.de/~guckes/sven/>
" Related files:
"               http://www.math.fu-berlin.de/~guckes/vim/src/emacs.vim
"               http://www.math.fu-berlin.de/~guckes/vim/src/latex.vim
"               http://www.math.fu-berlin.de/~guckes/vim/src/html.vim
"               http://www.math.fu-berlin.de/~guckes/vim/syntax/
" ================================================================
" NOTE:  If your read this then please send me an email!
" I welcome all feedback on this file - especially
" with new ideas such as abbreviations and mappings.
" Thanks!  And enjoy Vim!  :-)
" --Sven guckes@vim.org and guckes-vimrc@math.fu-berlin.de
" ================================================================
" Webpages on Vim that you should know about:
" http://www.vim.org/                   Vim Home Page (actually a mirror)
" http://www.vim.org/announce/          Vim Announcements
" http://www.vim.org/bina.html          sites with vim binaries
" http://www.vim.org/bugs.html          Some Vim Bugs
" http://www.vim.org/cvs.html           CVS server
" http://www.vim.org/deve.html          Current development issues
" http://www.vim.org/dist.html          All mirrors in detail
" http://www.vim.org/faq/               Vim FAQ
" http://www.vim.org/hist.html          History of ChangeLogs
" http://www.vim.org/html/              Vim Helptexts in HTML
" http://www.vim.org/howto/             Vim HowTo Texts
" http://www.vim.org/iccf/              Vim and the ICCF (see ":help iccf")
" http://www.vim.org/lang.html          Language/SyntaxFile Summary
" http://www.vim.org/macs.html          Vim on MacOS development
" http://www.vim.org/mail.html          Mailing lists and all that
" http://www.vim.org/mirrors.html       All mirror sites (ftp and www)
" http://www.vim.org/news.html          NEWS about Vim and the pages
" http://www.vim.org/newsgroup.html     Vim on comp.editors
" http://www.vim.org/orga.html          vim.org background info
" http://www.vim.org/pics.html          Pictures: buttons,icons,screenshots
" http://www.vim.org/press/             Articles on Vim in Magazines
" http://www.vim.org/quotes.html        Quotes on Vim by users
" http://www.vim.org/sites.html         Sites that use Vim officially
" http://www.vim.org/syntax/            Syntax files from current version
" http://www.vim.org/syntax.new/        New Syntax files (not yet in distrib)
" http://www.vim.org/user.html          Vim Users and their pages on Vim
" http://www.vim.org/wish.html          Wishlist for new features
" http://www.vim.org/why.html           Why to use a vi clone (Vim!)
" http://www.vim.org/y2k.html           Y2K issues (as if there are any..)
" ================================================================
" Info on the latest versions of Vim are on the Vim Home Page:
"   http://www.vim.org/
" Mind you, those pages are a *mirror* of the pages at
"   http://www.math.fu-berlin.de/~guckes/vim/
" ================================================================
" The latest versions of Vim are usually in my signature file:
" http://www.math.fu-berlin.de/~guckes/sig/SIGS  Comments?
" ================================================================
" Installation of this setup file:
"  To use this setup file, copy it to
"  this filename    on these systems:
"    ~/.vimrc       Unix and OS/2
```

```
"    s:.vimrc       Amiga
"    $VIM\_vimrc    MS-DOS and Win32
" =====================================================================
" version check:
" The first line of this setup file contains the information
" "version xxx" which allows VIM to check whether the setup file
" fits the syntax that it understands.
" Versions of VIM other than of version 5 then will give a warning
" as they do not understand this setup file command - a feature:
" Give a warning so the user knows that there is something odd
" about the setup file.
" =====================================================================
" =====================================================================
" Structure of this file:
" Lines starting with an inverted comma (") are comments.
" Some mappings are commented out.  Remove the comment to enable them.
"
" There are three kinds of things which are defined in this file:
" Mapping ("map"), settings ("set"), and abbreviations ("ab").
"   Settings affect the behaviour of commands.
"   Mappings maps a key sequence to a command.
"   Abbreviations define words which are replaced
"   right *after* they are typed in.
"
" =====================================================================
" Note on mappings - "angle notation" (see ":help <>"):
" VIM allows you to define mappings with special characters
" with a notation that uses non-special characters:
" The notation encloses decriptive words in angle brackets (<>).
" The characters you will most often are:
" <C-M> for control-m
" <C-V> for control-v which quotes the following character
" <ESC> for the escape character.
" All control characters have been replaced to use the angle notation
" so you should be able to read this file without problems.
" (Well, sometimes I leave some tabs [control-i] in the file. ;-)
" =====================================================================
" External programs:
" Some mappings make use of external programs.
" The following you should find/have on every UNIX system:
" cut, date; awk, egrep, grep, ispell, perl, sed.
" If you are using DOS then you should get these for you system!!
" See http://www.math.fu-berlin.de/~guckes/dos/ for some pointers!
" =====================================================================
" Online Help - jump to help positions with "\\":
" On some keyboards you will have some trouble with C-],
" the command to jump to the current help tag.
" The following mapping allows to use "\\", too:
  map \\  <C-]>
" =====================================================================
" The command ":version" does not show the current value of
" VIMRUNTIME - dang!  So I need a fast way to display that value:
  map :V :echo $VIMRUNTIME<c-m>
" =====================================================================
" =====================================================================
" HTML - HTML - HTML - HTML - HTML - HTML - HTML - HTML
" =====================================================================
" This has become quite big - so I moved it out to another file:
" http://www.math.fu-berlin.de/~guckes/vim/source/html.vim [980227]
" The "expand" is necessary to evaluate "~guckes".
  let FILE=expand("~guckes/.P/vim/source/html.vim")
  if filereadable(FILE)
     exe "source " . FILE
  endif
" =====================================================================
"
" =====================================================================
" SETTING OPTIONS
" =====================================================================
" There are many options for Vim - over 200.   Here is an overview:
" http://www.vim.org/options54.txt   VIM-5.4   [990726] 218 options.
" http://www.vim.org/options57.txt   VIM-5.7   [000624] 219 options.
```

```
" http://www.vim.org/options60ae.txt VIM-6.0ae [010504] 283 options.
" ======================================================================
"
"       autoindent, paste, textwidth:
"       I keep changing these values - just as the case may be.
"       Now, if functions keys actually worked on all keyboards
"       then I'd probably defines a toggle for each of them...
"
"       autoindent:  "off" as I usually do not write code.
  set noautoindent
"
"       autowrite: Automatically save modifications to files
"       when you use critical (rxternal) commands.
  set   autowrite
"
"       backup:  backups are for wimps  ;-)
  set nobackup
"
"       backspace:  '2' allows backspacing" over
"       indentation, end-of-line, and start-of-line.
"       see also "help bs".
  set   backspace=2
"
"       background:  Are we using a "light" or "dark" background?
" set   background=dark
"
"       compatible:  Let Vim behave like Vi?  Hell, no!
  set nocompatible
"
"       comments default: sr:/*,mb:*,el:*/,://,b:#,:%,:XCOMM,n:>,fb:-
  set   comments=b:#,:%,fb:-,n:>,n:)
"
"       cpoptions you should get to know - source of many FAQs!  ;-)
"       cpoptions:  "compatible options" to match Vi behaviour
" set   cpoptions="aABceFs"   "default!
"       FAQ:  Do NOT include the flag '<' if you WANT angle notation!
"
"       dictionary: english words first
  set   dictionary=/usr/dict/words,/local/lib/german.words
"       Source for dictionaries (in unix-format):
"       ftp://ftp.fu-berlin.de/misc/dictionaries/unix-format/
"       However, these are quite old.  Is there a better source?
"
"       digraph:    required for those umlauts
  set   digraph
"
"       errorbells: damn this beep!  ;-)
  set noerrorbells
"
"       esckeys:    allow usage of cursor keys within insert mode
"       You will find this useful when working, eg, on SunOS.
  set   esckeys
"
" In case you want this for SunOS only:
" if system('uname')=='SunOS'
"   set ek
" endif
"       expandtab:  Expand Tabs?  Rather not.
"                   See 'listchars' to make Tabs visible!
  set noexpandtab
"
"       formatoptions:  Options for the "text format" command ("gq")
"                       I need all those options (but 'o')!
  set   formatoptions=cqrt
"
"       helpheight: zero disables this.
  set   helpheight=0
"
"       helpfile:  path+filename of the main helpfile, ie "help.txt"
" set   helpfile=c:\\vim-4.6\\docs\\help.txt
" set   helpfile=c:/vim-4.6/docs/help.txt
" On Windows, I put the Vim helpfiles
" into the directory C:\VIM-version, eg C:\VIM-60ab
```

```
"
" Support users of old vim versions on the local net:
  if version==507
    set helpfile=/import/Mweb/guckes/vim/doc/help.txt
  endif
"
" For use from a WindowsNT machine which mounts
" the user's home directory on drive 'Z':
" if has("dos16") || has("dos32") || has("gui_w32")
"   set helpfile=Z:\share\vim\runtime\doc\help.txt
" endif
"
"       hidden:  Allow "hidden" buffers.  A must-have!
  set   hidden
"
"       highlight=8b,db,es,hs,mb,Mn,nu,rs,sr,tb,vr,ws
  set   highlight=8r,db,es,hs,mb,Mr,nu,rs,sr,tb,vr,ws
"
"       hlsearch :  highlight search - show the current search pattern
"       This is a nice feature sometimes - but it sure can get in the
"       way sometimes when you edit.
  set nohlsearch
"
"       icon:       ...
  set noicon
"
" set   iconstring  file of icon (icons?  on a terminal?  pff!)
" set   iconstring
"
"       ignorecase:  ignore the case in search patterns?  NO!
  set noignorecase
"
"       insertmode:
" FAQ:  Q: How can I quit insertmode when using this option?
"       A: The option "insertmode" was not meant for "start Vim in
"       insert mode" only; the idea is to *stay* in insert mode.
"       Anyway, you can use the command |i_CTRL-O| to issue commands.
  set noinsertmode
"
"
"       iskeyword:
"       iskeyword=@,48-57,_,192-255   (default)
" Add the dash ('-'), the dot ('.'), and the '@' as "letters" to "words".
" This makes it possible to expand email and html addresses,
" eg guckes-www@vim.org and http://www.vim.org/
  set   iskeyword=@,48-57,_,192-255,-,.,:,/,@-@
"
"       joinspaces:
"       insert two spaces after a period with every joining of lines.
"       I like this as it makes reading texts easier (for me, at least).
  set   joinspaces
"
"       keywordprg:  Program to use for the "K" command.
" set   keywordprg=man\ -s
"
"       laststatus:  show status line?  Yes, always!
"       laststatus:  Even for only one buffer.
  set   laststatus=2
"
" [VIM5]lazyredraw:  do not update screen while executing macros
  set   lazyredraw
"
"       'list' + 'listchars': Great new feature of vim-5.3!
"       This tells Vim which characters to show for expanded TABs,
"       trailing whitespace, and end-of-lines.  VERY useful!!
" Standard settings:
" set   list
" set   listchars=tab:>-,trail:·,eol:$
"
" However: The '$' at the end of lines is a bit too much, though.
" And I quite like the character that shows a dot in the middle:
" set   listchars=tab:>·,trail:·
```

```
"
" Some people might prefer a double right angle (>>)
" to show the start of expanded tabs, though:
" set   listchars=tab:»·,trail:·
"
" However, this all breaks up when viewing high-bit characters
" through some brain-dead telnet program (there are many).
" Sometimes a change of the font does the trick.  Try it!
"
"       magic:  Use 'magic' patterns  (extended regular expressions)
"       in search patterns?  Certainly!  (I just *love* "\s\+"!)
  set   magic
"
"       modeline:    ...
"       Allow the last line to be a modeline - useful when
"       the last line in sig gives the preferred textwidth for replies.
  set   modeline
  set   modelines=1
"
"       number:      ...
  set nonumber
"
"       path:   The list of directories to search when you specify
"               a file with an edit command.
"               Note:  "~/.P" is a symlink to my dir with www pages
"               "$VIM/syntax" is where the syntax files are.
" set   path=.,,~/.P/vim,~/.P/vim/runtime/syntax,~/.P/vim/source,$VIM/syntax/
" set   path=.,,~/.P/vim,~/.P/mutt/,~/.P/elm,~/.P/slrn/,~/.P/nn
"
"       pastetoggle
  set   pastetoggle=<f11>
"
"       report: show a report when N lines were changed.
"               report=0 thus means "show all changes"!
  set   report=0
"
"       ruler:       show cursor position?  Yep!
  set   ruler
"
"     runtimepath: list of dirs to search for runtime files
"     runtimepath is for Vim-6 only!
if version>=600
  set runtimepath=~/.vim,~/.vim/after
  set runtimepath+=~guckes/.vim
" set runtimepath+=~guckes/share/vim/vim60z/runtime/
  set runtimepath+=~guckes/share/vim/runtime/
  set runtimepath+=~guckes/share/vim/
  set runtimepath+=~guckes/.P/Vim/syntax
endif
" had to set this now that Vim won't use "helpfiles"
" to look for runtime files any more.
"
" Setting the "shell" is always tricky - especially when you are
" trying to use the same vimrc on different operatin systems.
"       shell for DOS
" set   shell=command.com
"       shell for UNIX -  math.fu-berlin.de BSD
" set   shell=zsh
"       shell for UNIX -   inf.fu-berlin.de BSD&Solaris
" set   shell=zsh
"       shell for UNIX - zedat.fu-berlin.de BSD&Solaris
" set   shell=/bin/tcsh
" Now that vim-5 has ":if" I am trying to automate the setting:
"
  if has("dos16") || has("dos32")
    let shell='command.com'
  endif
" gui_w32:  cmd.exe
"
" The zsh is now available at zedat.fu-berlin.de!  :-)
" start the zsh as a login shell:
  if has("unix")
    let &shell="zsh\ -l"
```

```
  endif
"
"       shiftwidth:  Number of spaces to use for each
"                    insertion of (auto)indent.
" set   shiftwidth=8
  set   shiftwidth=2
"
"       shortmess:   Kind of messages to show.   Abbreviate them all!
"          New since vim-5.0v: flag 'I' to suppress "intro message".
  set   shortmess=at
"
"       showcmd:     Show current uncompleted command?  Absolutely!
  set   showcmd
"
"       showmatch:   Show the matching bracket for the last ')'?
  set   showmatch
"
"       showmode:    Show the current mode?  YEEEEEEEEESSSSSSSSSS!
  set   showmode
"
"       suffixes:    Ignore filename with any of these suffixes
"                    when using the ":edit" command.
"                    Most of these are files created by LaTeX.
  set   suffixes=.aux,.bak,.dvi,.gz,.idx,.log,.ps,.swp,.tar
"
"       startofline:  no:  do not jump to first character with page
"       commands, ie keep the cursor in the current column.
  set nostartofline
"
"       splitbelow:  Create new window below current one.
  set   splitbelow
"
"       statusline:  customize contents of the windows' status line.
"       I prefer it this way:
"       Show the current buffer number and filename with info on
"       modification, read-only, and whether it is a help buffer
"       (show only when applied).
" set   statusline=[%n]\ %f\ %(\ %M%R%H)%)
"
"       Move the rest to the right side, eg a copyright text:
" set   statusline=[%n]\ %f\ %(\ %M%R%H)%)%=(c)\ Sven\ Guckes
"
"       Show the value of the current character in ASCII and Hex:
" set   statusline=[%n]\ %f\ %(\ %M%R%H)%)\=ASCII=%b\ HEX=%B
"
"       Show the current position with line+column+virtual_column:
" set   statusline=[%n]\ %f\ %(\ %M%R%H)%)\=Pos=<%l\,%c%V>\ %P
"       Adding color through UserN groups:
" set   statusline=%1*[%02n]%*\ %2*%F%*\ %(\ %M%R%H)%)%=%3*Pos=<%l,%c%V>%*
"       User1: color for buffer number
" hi    User1 ctermfg=red   ctermbg=white
"       User2: color for filename
" hi    User2 ctermfg=green ctermbg=white
"       User3: color for position
" hi    User3 ctermfg=blue  ctermbg=white
"
"       tabstop
  set   tabstop=8
"
" 990503: I have to set the "term" explicitly
" because the standard setups are broken.
" set   term=builtin_pcansi
" set   term=xterm
"
" Set the colors for vim on "xterm"
" if &term=="xterm"
" set t_Co=8
" set t_AB=[%?%p1%{8}%<%t%p1%{40}%+%e%p1%{92}%+%;%dm
" set t_AF=[%?%p1%{8}%<%t%p1%{30}%+%e%p1%{82}%+%;%dm
" endif
"
"       textmode:    no - I am using Vim on UNIX!
```

```
" set notextmode
"
"       textwidth
  set   textwidth=79
"
"       title:
  set notitle
"
"       ttyfast:     are we using a fast terminal?
"                    seting depends on where I use Vim...
  set nottyfast
"
"       ttybuiltin:
  set nottybuiltin
"
"       ttyscroll:      turn off scrolling -> faster!
  set   ttyscroll=0
"
"       ttytype:
" set   ttytype=rxvt
"
"       viminfo:  What info to store from an editing session
"                 in the viminfo file;  can be used at next session.
  set   viminfo=%,'50,\"100,:100,n~/.viminfo
"
"       visualbell:
  set   visualbell
"
"       t_vb:  terminal's visual bell - turned off to make Vim quiet!
"       Please use this as to not annoy cow-orkers in the same room.
"       Thankyou!  :-)
  set   t_vb=
"
"       whichwrap:
"       Allow jump commands for left/right motion to wrap to previous/next
"       line when cursor is on first/last character in the line:
  set   whichwrap=<,>,h,l,[,]
"
"       wildchar  the char used for "expansion" on the command line
"                 default value is "<C-E>" but I prefer the tab key:
  set   wildchar=<TAB>
"
"       wrapmargin:
  set   wrapmargin=1
"
"       writebackup:
  set nowritebackup
"
" ======================================================================
" ABbreviations
" ======================================================================
" 980701: Moved the abbreviations *before* the mappings as
" some of the abbreviations get used with some mappings.
"
" Abbreviations for some important numbers:
  iab Npi 3.14159265358979323846264338327950288419 72
  iab Ne  2.71828182845904523536028747135266249775 73
"
" Abbreviations for some classic long words:
"     Donau... is the German word for the (read in reverse)
"     "additional paragraph of the law regulating the pension of
"      widows to captains of the ship company on (the river) Danube"
"     (I am not making this up! ;-)
  iab YDD
Donaudampfschifffahrtgesellschaftskapitaenwitwenrentengesetzzusatzparagraph
" New German spelling "schifffahrt", too! (Hi, Doram!)
"
" 010131
  iab YTNGV Telekommunikationsnummerngebuehrenverordnung
" 010131
" iab YRRAG Rinderkennzeichnungs- und \
"     Rindfleischetikettierungsüberwachungsaufgabenübertragungsgesetz (RkReÜAÜG)
"
```

```
" Some more weird sentences in KrautLang:
" "Jörg möchte fünf Frühstücksbrötchen
"  um äußerstes Völlegefühl zu spüren."
" "Äßen Löwen Möwen, zögen Möwen über Löwen rüber."
" "Äße Öko-Jörg große Bärenfüße in äußerst süßer Nußölkäsesoße
"  müßte er in ein übermäßig häßliches Güllefaß."
  iab YHRA "Heizölrückstoßabdämpfung"
" all three umlauts and the 'ß', too!
" Yes, KrautLang *is* weird!
"
"     YLL : The name of a town in Wales.  I am not making this up!
  iab YLL    LLanfairpwllgwyngyllgogerychwyrndrobwllllantysiliogogogoch
" http://www.llanfairpwllgwyngyllgogerychwyrndrobwllllantysiliogogogoch.co.uk
" http://194.159.85.168/ - I am not making this up!  :-)
"
"     YTauma: The name of a hill in New Zealand.
  iab YTauma
Taumatawhakatangihangakoauauotamateaturipukakapikimaungahoronukupokaiwenuakitanatahu
"
" Longest Word in the second edition of the Oxford English Dictionary?
  iab YNNMSVC pneumonoultramicroscopicsilicovolcanoconiosis
"
"     Yalpha : The lower letter alphabet.
  iab Yalpha abcdefghijklmnopqrstuvwxyz
"
"     YALPHA : The upper letter alphabet.
  iab YALPHA ABCDEFGHIJKLMNOPQRSTUVWXYZ
"
"     Ydigit: The ten digits.
  iab Ydigit  1234567890
"
"     Yruler: A "ruler" - nice for counting the length of words.
  iab Yruler
12345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890
"
"     Yupsidedown : This describes people from "down under" (Hi, Dean!).
  iab Yupsidedown umop-ap!sdn
"
"     Ysuper: A nice long word from the musical "Mary Poppins".
  iab Ysuper supercalifragilisticexpialidocious
"
"     Yanti:  The longest proper word in the English language?!
  iab Yanti   antidisestablishmentarianism
"
"     Ypass:  Standard answer to Usenet posts
"             with the "Subject: HELP"  (hehe)
  iab Ypass  "You are in a maze of twisty little passages, all alike."
"
"     Yicty:  Some people should really RTFM!
  iab Yicty   I could tell you - but then I'd have to kill you.
"
"     Ybtdt:  It's been dejavu all over again.
  iab Ybtdt   Been there, done that, got the tshirt.
"
"     Ywttc:  One more person using Vim.
  iab Ywttc   Welcome to the club!
"
"     Ysesqui: "Sesquipedalophobia" means "fear of big words."  ;-)
  iab Ysesqui   sesquipedalophobia
"
"     Yi18n: .123456789012345678. - yup, 18 characters betwwen 'i' and 'n'
  iab Yi18n  internationalization
"
"     Yanfs: Changing the Subject within a message.  (Hi, Monty Python!)
  iab Yanfs  And now for something completely different...
"
"
"
" classic pangrams (which include every letter of the alphabet):
" German:
"   kaufen sie jede woche vier gute bequeme pelze [missing: 'xy' :-/]
"   sylvia wagt quick den jux bei pforzheim
```

```
"   bayerische jagdwitze von maxl querkopf
"   zwei boxkaempfer jagen eva quer durch sylt
"   falsches üben von xylophonmusik quält jeden größeren zwerg.
"   Bei jedem klugen Wort von Sokrates rief Xanthippe zynisch: Quatsch!
" English:
"       the quick brown fox jumps over the lazy dog
"       The five boxing wizards jump quickly
" French:
"        voyez le brick geant que j'examine pres du wharf.
" Polish:
"       Pchn±æ w tê ³ód¼ je¿a lub o¶m skrzyñ fig.
"       Koñ i ¿ó³w grali w ko¶ci z piêkn± æm± u ¼ród³a.
"       Têgi koñ i ma³y ¿ó³w ¶piewali z piêkn± æm± u ¼ród³a ¿ycia.
"
" And a sentence to break some quoing levels:
"       "This man's house (which 's yellow) burned down."
"
" And now for something completely different:
"       I couldn't bear to bear bears over the border.
"
" Inserting an ellipsis to indicate deleted text
  iab  Yell  [...]
  vmap ,ell c[...]<ESC>
"
" Correcting those typos. [I almost never get these right.  :-(]
" See also:  http://www.igd.fhg.de/~zach/programs/acl/
  iab alos      also
  iab aslo      also
  iab becuase   because
  iab bianry    binary
  iab bianries  binaries
  iab charcter  character
  iab charcters characters
  iab exmaple   example
  iab exmaples  examples
  iab shoudl    should
  iab seperate  separate
  iab teh       the
  iab tpyo      typo
" Some frequent typos in German:
  iab nciht     nicht
  iab doer      oder
  iab Dreckfuhler Druckfehler
"
" Sorry, Laurent!
  iab Laurant   Laurent
"
" See http://www.math.fu-berlin.de/~guckes/sig/:
  iab YDDS dash-dash-space
"
" For reports and texts on my studies:
" iab YKT Komplexitaetstheorie
" iab YRA Rechnerarchitektur
" iab YPM Pattern Matching
" see http://elib.zib.de/ICM98 :
" iab YICM International Congress of Mathematicians
"
" Some sentences that I really use often in emails about Vim:
  iab YAW You are welcome!
  iab YEV Enjoy Vim!  :-)
"
" Often used filenames - only needed these on the command line:
" see also:  http://www.math.fu-berlin.de/~guckes/setup/
"
" cab ELMALIAS  ~/.elm/aliases.text
" cab Erc       ~/.elm/elmrc
" cab Mrc       ~/.muttrc
" cab Src       ~/.slrnrc
" cab Zrc       ~/.zsh/.zshrc
"
"  Email Adresses:
"  I usually use these when adding addresses to the header
"  of emails (mutt) and posts (slrn).
```

```
"
"    Author of the Good NetKeeping Seal of Approval:
  ab Agnksa   js@xs4all.nl (Jeroen Scheerder)
"
"    Original author of Mutt:
" ab Amutt    me@cs.hmc.edu (Michael Elkins)
"    Maintainer of Mutt:
  ab Amutt    roessler@guug.de (Thomas Roessler)
"
"    Author of Slrn:
  ab Aslrn    davis@space.mit.edu (John E. Davis)
"    Maintainer of Slrn:
  ab Mslrn    tstsls@gmx.de (Thomas Schultz)
"
"    Author of Vim - vim specific address:
  ab Avim     bram@vim.org (Bram Moolenaar)
"    Author of Vim - private address:
  ab Abram    Bram@Moolenaar.net (Bram Moolenaar)
"
"  Author, contributor, maintainer of
"  add atac autoconf bcpp ded dialog diffstat flist
"  lynx ncurses tin vile vttest xterm - and more!
  ab Avile dickey@his.com (Thomas E. Dickey)
"
"    Former Maintainer of the Vim FAQ:
" ab Avimfaq  laurent@Grafnetix.COM (Laurent Duperval)
  ab Avimfaq  guckes@vim.org (Sven Guckes)
"
"    Mailing Lists (MLs)
"
"    The Vim mailing lists: See http://www.vim.org/mail.html for more info!
" ab MLvim      vim@vim.org (VIM Help List)
" ab MLvimdev   vim-dev@vim.org (VIM General Development List)
" ab MLvimmac   vim-mac@vim.org (VIM on MacOS Development List)
" ab MLvimmb    vim-multibyte@vim.org (VIM MultiByte Development List)
" ab MLvimfr    vim-fr@egroups.com    (VIM Help List for French Users)
"
" More mailing lists:
" ab MLmuttdev  mutt-dev@mutt.org (Mutt Developer List)
" ab MLmuttuser mutt-users@mutt.org (Mutt Users List)
" ab MLzsh      zsh-users@sunsite.auc.dk (ZShell Users List)
"
"
"   News: newsgroup names
"
" Newsgroup about "warloding" of signatures - see
" also http://www.math.fu-berlin.de/~guckes/afw/
  iab Nafw    alt.fan.warlord
  iab Nahbou  alt.humor.best-of-usenet
  iab Nzedat  bln.announce.fub.zedat.d
  iab Ncsd    bln.announce.fub.cs.d
  iab Nce     comp.editors
  iab NcE     de.comp.editoren
" Newsgroup about "lynx":
  iab Nhtml   comp.infosystems.www.authoring.html
" Newsgroup about "elm":  Elm is dead - long live Mutt!
  iab Nelm    comp.mail.elm
" Newsgroup about "pine":  When will they release pine-4?
" iab Ncmp    comp.mail.pine
  iab Npine   comp.mail.pine
" iab Ncsmd   comp.sys.mac.digest
" Newsgroup about "mobil phone systems":
  iab Ndcm    de.comm.mobil
  iab Nmobil  de.comm.mobil
" Newsgroup about "web browsers":
  iab Nlynx     comp.infosystems.www.browsers.misc
  iab Nnetscape comp.infosystems.www.browsers.misc
" Newsgroup about "mutt" [since 980401]:  The coolest mail user agent
  iab Nmutt   comp.mail.mutt
" Newsgroup about "nn":  Once was the best newsreader. Still good.
  iab Nnn     news.software.nn
" Newsgroup for "newbies".
```

```
" All you ever wanted to know - but were afraid to ask. ;-)
  iab Newbie  news.newusers.questions
" Newsgroup about "newsreader *agents*" (netscape and slrn):
  iab Nnsr    news.software.readers
"
" Usenet header lines (used when composing a post):
"
  iab UFT  Followup-To:
  iab UMCT Mail-Copies-To: MYADDR
  iab UNG  Newsgroups:
  iab URT  Reply-To: MYADDR
  iab UFUB Organization: Freie Universitaet Berlin
"
" Current version numbers of my favourite programs:
" http://www.math.fu-berlin.de/~guckes/faq/program.versions.html
" And some abbreviations to type them in mail&news:
" [Last update: 001221]
  iab Velmold  ELM2.4PL25 [951204]
  iab Velm     ELM2.5.3 [000113]
  iab Velmme   ELMME+86 [001219]
  iab Vless    less-358 [000709]
  iab Vlynx    lynx-2.8.4dev14 [001103]
  iab Vmutt    mutt-1.2.5 [000729]
  iab Vslrn    slrn-0.9.6.3 [000918]
  iab Vvim     vim-5.7 [000624]
  iab Vvimdev  vim-6.0w [010226]
"
" My phone number (main phone number and fax):
  iab Yphone TEL (+49 30) 8838884<CR>FAX (+49 30) 88629362
" My cellphone number:
  iab Ycell  TEL (+49 179) 3966141
" If you like then send me a short message to my cellphone via this page:
  iab FreeSMS http://www.free-sms.com/sendsms/index.php3
" My snailmail address.  Postcards, anyone?
  iab Ysnail Sven Guckes<C-M>Pariser Str. 52<C-M>D-10719 Berlin
" My ICQ number.  Chat, anyone?
  iab YICQ   ICQ:38801898
"
"  My addresses (Email and WWW)
"  makes it easy to type them without typos  ;-)
  ab Amaili     guckes@inf.fu-berlin.de
  ab Amailm     guckes@math.fu-berlin.de
  ab Amailv     guckes@vim.org
  ab Amailz     guckes@zedat.fu-berlin.de
" ab Apriv        sven@guckes.net
  ab MYADDR     guckes@math.fu-berlin.de
  ab MYNAME     Sven Guckes
  ab MYDOMAIN   math.fu-berlin.de
" ab MYDOMAIN   guckes.net
"
" Setting the reply address when replying as the guy from SKB:
" ab ASKB   Sprachboerse <sprachboerse@tu-berlin.de>
" See also: http://www.math.fu-berlin.de/~guckes/skb/
"
"  My Home Pages at the departments at the FUB and elsewhere...
"
  ab WWWb   http://www.belug.org/~guckes/
  ab WWWm   http://www.math.fu-berlin.de/~guckes/
  ab WWWi   http://www.inf.fu-berlin.de/~guckes/
  ab WWWz   http://userpage.zedat.fu-berlin.de/~guckes/
"
" WWW Pages base URLs
"
  ab HPA   http://www.math.fu-berlin.de/~guckes/afw/
  ab HPa   http://www.math.fu-berlin.de/~guckes/ascii/
  ab HPc   http://www.math.fu-berlin.de/~guckes/calvin/
  ab HPD   http://www.math.fu-berlin.de/~guckes/dos/
  ab HPe   http://www.math.fu-berlin.de/~guckes/eplus/ab.faq.html
  ab HPE   http://www.math.fu-berlin.de/~guckes/elm/
  ab HPI   http://www.math.fu-berlin.de/~guckes/irc/
  ab HPi   http://www.math.fu-berlin.de/~guckes/ispell/
  ab HPL   http://www.math.fu-berlin.de/~guckes/lynx/
  ab HPl   http://www.math.fu-berlin.de/~guckes/less/
```

```
  ab HPm   http://www.math.fu-berlin.de/~guckes/mail/
  ab HPme  http://www.math.fu-berlin.de/~guckes/mail/edit.html
  ab HPM   http://www.math.fu-berlin.de/~guckes/mutt/
  ab HPN   http://www.math.fu-berlin.de/~guckes/nn/
  ab HPP   http://www.math.fu-berlin.de/~guckes/pine/
  ab HPp   http://www.math.fu-berlin.de/~guckes/procmail/
  ab HPr   http://babayaga.math.fu-berlin.de/~rxvt/
  ab HPR   http://www.math.fu-berlin.de/~guckes/rfc/
  ab HPs   http://www.math.fu-berlin.de/~guckes/screen/
  ab HPS   http://slrn.sourceforge.net/
  ab HPS1  http://www.math.fu-berlin.de/~guckes/slrn/
  ab HPS2  http://www.slrn.org/
  ab HPv   http://www.math.fu-berlin.de/~guckes/vi/
"    HPOV - the "original" URL of the Vim Home Page!
  ab HPOV  http://www.math.fu-berlin.de/~guckes/vim/
  ab HPV   http://www.vim.org/
  ab HPX   http://www.math.fu-berlin.de/~guckes/xmas/
  ab HPZ   http://www.math.fu-berlin.de/~guckes/zsh/
"
" Other important WWW addresses
"
  ab URLaltavista http://www.altavista.de/textonly.html
  ab URLftpsearch http://ftpsearch.lycos.com/?form=advanced
  ab URLrpmfind   http://rpmfind.net
  ab URLslashdot  http://slashdot.org
  ab URLfreshmeat http://freshmeat.net
"
" Pictures:
  ab URLbambi     http://www.math.fu-berlin.de/~guckes/pics/calvin/bambi.gif
  ab URLsecret    http://www.math.fu-berlin.de/~guckes/pics/calvin/secret.gif
  ab URLwhome     http://www.math.fu-berlin.de/~guckes/pics/calvin/who_me.gif
  ab URLstopspam  http://www.math.fu-berlin.de/~guckes/pics/spam/stop.jpg
  ab URLutefuchs  http://www.math.fu-berlin.de/~utefuchs/
"
" The nearest vim site for me:
  ab FTPG         ftp://ftp.math.fu-berlin.de/pub/usr/guckes/
  ab FTPFUB       ftp://ftp.fu-berlin.de/
  ab FTPVIM       ftp://ftp.fu-berlin.de/pub/misc/editors/vim/
  ab URLVINE      http://www.mossbayeng.com/~ron/vim/vine.html
" But you should look at http://www.vim.org/dist.html instead!
"
" ======================================================================
" Abbreviations - Header Lines for Email and News
" ======================================================================
" Define regexpr for headers to use with mappings
" as it makes reading the mappings much easier:
" cab HADDR     From\\|Cc\\|To
  cab HEMAIL ^\(From\\|Cc\\|To\\|Date\\|Subject\\|Message-ID\\|Message-Id\\|X-\)
  cab HNEWS  ^\(From\\|Cc\\|To\\|Date\\|Subject\\|Message-ID\\|X-\\|Newsgroups\)
"
" ======================================================================
" Abbreviations - General Editing - Inserting Dates and Times
" ======================================================================
"
" First, some command to add date stamps (with and without time).
" I use these manually after a substantial change to a webpage.
" [These abbreviations are used with the mapping for ",L".]
"
  iab Ydate <C-R>=strftime("%y%m%d")<CR>
" Example: 971027
"
  iab Ytime <C-R>=strftime("%H:%M")<CR>
" Example: 14:28
"
" man strftime:     %T      time as %H:%M:%S
" iab YDT           <C-R>=strftime("%y%m%d %T")<CR>
" Example: 971027 12:00:00
"
" man strftime:     %X      locale's appropriate time representation
  iab YDT           <C-R>=strftime("%y%m%d %X")<CR>
" Example: 000320 20:20:20
"
```

```
  iab YDATE <C-R>=strftime("%a %b %d %T %Z %Y")<CR>
" Example: Tue Dec 16 12:07:00 CET 1997
"
" On Windows the functions "strftime" seems to have a different
" format.  Therefore the following may be necessary:  [980730]
" if !has("unix")
" iab YDATE <C-R>=strftime("%c %a")<CR>
" else
" iab YDATE <C-R>=strftime("%D %T %a")<CR>
" endif
"
" 000306: These two lines at the start of a Perl script
" will start the first Perl in your Shell's $PATH.
  iab Yperlhead eval 'exec perl -w -S $0 ${1+"$@"}'<c-m>if 0;<c-m>
"
"
" ====================================================================
" MAPpings
" ====================================================================
" Caveat:  Mapping must be "prefix free", ie no mapping must be the
" prefix of any other mapping.  Example:  "map ,abc foo" and
" "map ,abcd bar" will give you the error message "Ambigous mapping".
"
" The backslash ('\') is the only(?) unmapped key, so this is the best
" key to start mappings with as this does not take away a command key.
" However, the backslash is never in the same position with keyboards.
" Eg on German keyboards it is AltGr-sz - don't ask.
" Anyway, I have decided to start mappings with the comma as this
" character is always on the same position on almost all keyboards
" and I hardly have a need for that command.
"
" The following maps get rid of some basic problems:
"
" When the backspace key sends a "delete" character
" then you simply map the "delete" to a "backspace" (CTRL-H):
" map <Del> <C-H>
"
" With Vim-4 the format command was just 'Q' and
" I am too used to it.  So I need this back!
  nnoremap Q gq
  vnoremap Q gq
"
" 980527 I often reformat a paragraph to fit some textwidth -
" and I use the following mapping to adjust it to the
" current position of the cursor:
  map #tw :set textwidth=<C-R>=col(".")<C-M>
"
" 981210 Whenever I paste some text into VIM I have to
" toggle from "nopaste" to "paste" and back again:
" map <f4>   :set paste!<C-M>:set paste?<C-M>
" map <esc>[14~ :set paste!<C-M>:set paste?<C-M>
" --> new option for this:  'pastetoggle'
"
" "tal" is the "trailer alignment" filter program
" Hopefully it will ship with Vim one day.
" vmap #t !tal<CR>
" vmap #t !tal -p 0<CR>
"
" Disable the command 'K' (keyword lookup) by mapping it
" to an "empty command".  (thanks, Lawrence! :-):
" map K :<CR>
" map K :<BS>
" More elegant:  (Hi Aziz! :-)
  map K <NUL>
"
" Disable the suspend for ^Z.
" I use Vim under "screen" where a suspend would lose the
" connection to the " terminal - which is what I want to avoid.
  map <C-Z> :shell
"
" Make CTRL-^ rebound to the *column* in the previous file
  noremap <C-^> <C-^>`"
"
```

```
" Make "gf" rebound to last cursor position (line *and* column)
  noremap gf gf`"
"
" When I let Vim write the current buffer I frequently mistype the
" command ":w" as ":W" - so I have to remap it to correct this typo:
  nmap :W :w
" TODO:  Use the following (after some testing):
" command -nargs=? -bang W w<bang> <args>
"
" Are you used to the Unix commands "alias" and "which"?
" I sometimes use these to look up my abbreviations and mappings.
" So I need them available on the command line:
  map :alias map
  map :which map
"
" The command {number}CTRL-G show the current nuffer number, too.
" This is yet another feature that vi does not have.
" As I always want to see the buffer number I map it to CTRL-G.
" Pleae note that here we need to prevent a loop in the mapping by
" using the comamnd "noremap"!
  noremap <C-G> 2<C-G>
"
" 001010  Do the Color Test!
  map ,CT :sp $VIMRUNTIME/syntax/colortest.vim<cr>:so %<cr>
" 000329  View the file which defines the "filetypes":
  map ,F :view $VIMRUNTIME/filetype.vim
"
" 980311  Sourcing syntax files::
  map ,SO :source $VIMRUNTIME/syntax/
"
" 980706,000310 View a syntax file:
  map ,V  :view $VIMRUNTIME/syntax/
"
" 000801     Hilite Test - show all current highlight groups
" see ":help hitest.vim"
  map ,HI :so $VIMRUNTIME/syntax/hitest.vim
"
" 990614  Quick insertion of an empty line:
" nmap <CR> o<ESC>
" I find this convenient - but as I am also used to proceed to
" next line by pressing CR this often gives me new empty lines
" when I really do not need them.  :-(
"
" ===========================================================================
" Customizing the command line
" ===========================================================================
" Valid names for keys are:  <Up> <Down> <Left> <Right> <Home> <End>
" <S-Left> <S-Right> <S-Up> <PageUp> <S-Down> <PageDown>  <LeftMouse>
"
" Many shells allow editing in "Emacs Style".
" Although I love Vi, I am quite used to this kind of editing now.
" So here it is - command line editing commands in emacs style:
  cnoremap <C-A>      <Home>
  cnoremap <C-B>      <Left>
" cnoremap <C-B>      <Del>
  cnoremap <C-E>      <End>
  cnoremap <C-F>      <Right>
  cnoremap <C-N>      <End>
  cnoremap <C-P>      <Up>
  cnoremap <ESC>b     <S-Left>
  cnoremap <ESC><C-B> <S-Left>
  cnoremap <ESC>f     <S-Right>
  cnoremap <ESC><C-F> <S-Right>
  cnoremap <ESC><C-H> <C-W>
" Note:  More info about this is in the helptexts:  :help emacs-keys
"
" Additional codes for that "English" keyboard at the Xterminal
  cnoremap <ESC>[D <Left>
  cnoremap <ESC>[C <Right>
"
" ===========================================================================
" VIM - Editing and updating the vimrc:
```

```
" As I often make changes to this file I use these commands
" to start editing it and also update it:
  if has("unix")
    let vimrc='~/.vimrc'
  else
" ie:  if has("dos16") || has("dos32") || has("win32")
    let vimrc='$VIM\_vimrc'
  endif
  nn  ,u :source <C-R>=vimrc<CR><CR>
  nn  ,v :edit   <C-R>=vimrc<CR><CR>
"     ,v = vimrc editing (edit this file)
" map ,v :e ~/.vimrc<CR>
"     ,u = "update" by reading this file
" map ,u :source ~/.vimrc<CR>
" ====================================================================
"
" General Editing
"
" Define "del" char to be the same backspace (saves a LOT of trouble!)
" As the angle notation cannot be use with the LeftHandSide
" with mappings you must type this in *literally*!
" map <C-V>127 <C-H>
"cmap <C-V>127 <C-H>
" the same for Linux Debian which uses
 imap <Esc>[3~ <C-H>
"
"      ;rcm = remove "control-m"s - for those mails sent from DOS:
  cmap ;rcm %s/<C-M>//g
"
"     Make whitespace visible:
"     Sws = show whitespace
  nmap ,Sws :%s/ /_/g<C-M>
  vmap ,Sws :%s/ /_/g<C-M>
"
"     Sometimes you just want to *see* that trailing whitespace:
"     Stws = show trailing whitespace
  nmap ,Stws :%s/  *$/_/g<C-M>
  vmap ,Stws :%s/  *$/_/g<C-M>
"
" General Editing - Turning umlauts into ascii (for German keyboards)
"
" imap ä ae
" imap ö oe
" imap ü ue
" imap ß ss
"
" &#196; -> Ä  :%s/\&#196;/Ä/gc  -> D
" &#214; -> Ö  :%s/\&#214;/Ö/gc  -> V
" &#220; -> Ü  :%s/\&#220;/Ü/gc  -> \
" &#228; -> ä  :%s/\&#228;/ä/gc  -> d
" &#246; -> ö  :%s/\&#246;/ö/gc  -> v
" &#252; -> ü  :%s/\&#252;/ü/gc  -> |
"
" ====================================================================
" Inserting Dates and Times / Updating Date+Time Stamps
" ====================================================================
"     ,L  = "Last update" - replace old time stamp with a new one
"        preserving whitespace and using internal "strftime" command:
"       requires the abbreviation  "YDATE"
  map ,L  1G/Last update:\s*/e+1<CR>CYDATE<ESC>
  map ,,L 1G/Last change:\s*/e+1<CR>CYDATE<ESC>
" Example:
" before:  "Last update:   Thu Apr  6 12:07:00 CET 1967"
" after:   "Last update:   Tue Dec 16 12:07:00 CET 1997"
"
"  I used to read in the output from the external command "date"
"  but this is a little slow and does not work on all systems:
" map ,L 1G/Last update: */e+1<CR>D:r!date<CR>kJ
"
" ====================================================================
" General Editing - link to program "screen"
" ====================================================================
"
```

```
"       ,Et = edit temporary file of "screen" program
  map   ,Et :e /tmp/screen-exchange
"       as a user of Unix systems you *must* have this program!
"       see also:  http://www.math.fu-berlin.de/~guckes/screen/
"
" Email/News - Editing replies/followups
"
" Part 1 - prepare for editing
"
" Part 2 - getting rid of empty (quoted) lines and space runs.
"
" Delete trailing whitespace:
 nmap <f9> :%s/\s\+$//
 vmap <f9>  :s/\s\+$//
"
"      ,cel = "clear empty lines"
"       - delete the *contents* of all lines which contain only whitespace.
"         note:  this does not delete lines!
" map ,cel :g/^[<C-I> ]*$/d
  map ,cel :%s/^\s\+$//
"
"      ,del = "delete 'empty' lines"
"       - delete all lines which contain only whitespace
"         note:  this does *not* delete empty lines!
  map ,del :g/^\s\+$/d
"
"      ,cqel = "clear quoted empty lines"
"       Clears (makes empty) all lines which start with '>'
"       and any amount of following spaces.
" nmap ,cqel :%s/^[> ]*$//
" vmap ,cqel  :s/^[> ]*$//
" nmap ,cqel :%s/^[><C-I> ]\+$//
" vmap ,cqel  :s/^[><C-I> ]\+$//
  nmap ,cqel :%s/^[>]\+$//
  vmap ,cqel  :s/^[><C-I> ]\+$//
" NOTE: If the meta sequence "\s"
" The following do not work as "\s" is not a character
" and thus cannot be part of a "character set".
"  map ,cqel  :g/^[>\s]\+$/d
"
" Some people have strange habits within their writing.
" But if you cannot educate them - rewrite their text!  ;-)
"
" Jason "triple-dots" King elephant@onaustralia.com.au
" does uses ".." or "..." rather than the usual punctuation
" (comma, semicolon, colon, full stop). So...
"
" Turning dot runs with following spaces into an end-of-sentence,
" ie dot-space-space:
  vmap ,dot :s/\.\+ \+/.  /g
"
" Gary Kline (kline@tera.tera.com) indents his
" own text in replies with TAB or spaces.
" Here's how to get rid of these indentation:
  vmap ,gary :s/^>[ <C-I>]\+\([^>]\)/> \1/
"
"      ,ksr = "kill space runs"
"             substitutes runs of two or more space to a single space:
" nmap ,ksr :%s/  */ /g
" vmap ,ksr  :s/  */ /g
  nmap ,ksr :%s/  \+/ /g
  vmap ,ksr  :s/  \+/ /g
" Why can't the removal of space runs be
" an option of "text formatting"? *hrmpf*
"
"    ,Sel = "squeeze empty lines"
"    Convert blocks of empty lines (not even whitespace included)
"    into *one* empty line (within current visual):
   map ,Sel :g/^$/,/./-j
"
"    ,Sbl = "squeeze blank lines"
"    Convert all blocks of blank lines (containing whitespace only)
```

```
"    into *one* empty line (within current visual):
"  map ,Sbl :g/^\s*$/,/[^ <C-I>]/-j
"  map ,Sbl :g/^\s*$/,/[^ \t]/-j
   map ,Sbl :g/^\s*$/,/\S/-j
"
" ================================================================
" AUTOCOMMANDS:
" Editing of email replies and Usenet followups - using autocommands
" ================================================================
"
" First step:  Remove ALL auto-commands.  This avoids having the
" autocommands twice when the vimrc file is sourced again.
  autocmd!
"
" Add/change some filename patterns to filetypes:
" let myfiletypefile = "~/.vim.myfiletypes"
"
" And this goes into  "~/.vim.myfiletypes":
" augroup filetype
" au BufRead,BufNewFile postpone/*  set filetype=mail
" augroup END
"
" Apply my syntax file on files with extension "sdg":
  au BufNewFile,BufRead *.sdg                     set ft=sveng
"
" Apply the muttrc coloring to my mutt setup files:
  au BufNewFile,BufRead .mutt.*                   set ft=muttrc
"
" SLRN: Apply mail.vim to postponed messages:
  au BufNewFile,BufRead */postpone/*              set ft=mail
"
" Set some options for "messages" (FileType "mail"):
" au FileType mail set autoindent expandtab tabstop=4 textwidth=70
" the following uses ethe short option names:
  au FileType mail set ai et ts=4 tw=70
"
" Some more autocommand examples which set the values for
" "autoindent", "expandtab", "shiftwidth", "tabstop", and "textwidth":
" using the "FileType" event:
" au FileType c           set ai et sw=4 ts=4
" au FileType perl        set ai et sw=4 ts=4
" au FileType html        set ai    sw=2 ts=2
" au FileType java        set ai    sw=4 ts=4
" au BufEnter */drafts/*  set tw=72
"
" Try to use the mapping ",D" when doing a followup.
" autocmd BufNewFile ~/.followup ,D|
"
" Setting "X-Editor":
" Let Vim identify itself in the message header
" when editing emails with Mutt and Slrn:
" au FileType mail let @"="X-Editor: Vim-".version." http://www.vim.org/\n"|exe
'norm 1G}""P'
"
" Generation of a "Message-ID" header line (according to ISO8061):
"
" Let Vim add a Message-ID to your posts with Slrn:
"
" Add a Message-ID to posts with Slrn:
" au BufNewFile,BufRead ~/.article let @"="Message-ID:
<slrn.".system("uname").".".strftime("%y%m%d%H%M")."\n"|exe 'norm 1G}""P'
" Problem:  The uname adds a CTRL-J at the end of the output.  :-(
"
" Add a Message-ID to posts with Slrn (mimimal format):
" au BufNewFile,BufRead ~/.article,~/.followup let @"="Message-ID:
<".strftime("%Y%m%dT%H%M%S")."@guckes.net>\n"|exe 'norm 1G}""P'
" Example:  Message-ID: <20000406T120700@guckes.net>
" Problem:  The date and time are hard to read.
" It's easier with a few delimiters:
"
" Add a Message-ID to posts with Slrn (with delimiters):
" au BufNewFile,BufRead ~/.article,~/.followup let @"="Message-ID:
<".strftime("%Y-%m-%dT%H-%M-%S")."@guckes.net>\n"|exe 'norm 1G}""P'
```

```
" Example:  Message-ID: <2000-04-06T12-07-00@guckes.net>
"
" news.fu-berlin.de rejects Message-IDs with colons in it:
" Example:  Message-ID: <2000-04-06T12:07:00@guckes.net>
"
"
" Part 3 - Change Quoting Level
"
"      ,dp = de-quote current inner paragraph
"  map ,dp {jma}kmb:'a,'bs/^> //<CR>
   map ,dp vip:s/^> //<CR>
  vmap ,dp    :s/^> //<CR>
"
"      ,qp = quote current paragraph
"            jump to first inner line, mark with 'a';
"            jump to last  inner line, mark with 'b';
"            then do the quoting as a substitution
"            on the line range "'a,'b":
"  map ,qp {jma}kmb:'a,'bs/^/> /<CR>
"      vim-5 now has selection of "inner" and "all"
"      of current text object - mapping commented!
"
"      ,qp = quote current paragraph (old version)
"            jump to first inner line, Visual,
"            jump to last  inner line,
"            then do the quoting as a substitution:
"  map ,qp {jV}k:s/^/> /<CR>
"
"      ,qp = quote current inner paragraph (works since vim-5.0q)
"            select inner paragraph
"            then do the quoting as a substitution:
   map ,qp   vip:s/^/> /<CR>
"
"      ,qp = quote current paragraph
"            just do the quoting as a substitution:
  vmap ,qp    :s/^/> /<CR>
"
"       ## = comment current inner paragraph with '#':
  nmap  ##   vip:s/^/#<space>/<CR>
"       ## = comment current text selection  with '#':
  vmap  ##      :s/^/#<space>/<CR>
"
" 001006:  Commenting selected lines in C style:
" vmap ## :s#^#// #<cr>'<O/*<esc>'>o*/<esc>gv
"
" Example of result:
" /*
" // foo
" // bar
" /*
"
" Changing quote style to *the* true quote prefix string "> ":
"
"       Fix Supercite aka PowerQuote (Hi, Andi! :-):
"       before ,kpq:    >   Sven> text
"       after  ,kpq:    > > text
"      ,kpq kill power quote
  vmap ,kpq :s/^> *[a-zA-Z]*>/> >/<C-M>
"
"       Fix various other quote characters:
"      ,fq "fix quoting"
  vmap ,fq :s/^> \([-":}\|][ <C-I>]\)/> > /
"
"  Fix the quoting of "Microsoft Internet E-Mail":
"  The guilty mailer identifies like this:
"  X-Mailer: Microsoft Internet E-Mail/MAPI - 8.0.0.4211
"
"  And this is how it quotes - with a pseudo header:
"
"  -----Ursprungliche Nachricht-----
"  Von:            NAME [SMTP:ADDRESS]
"  Gesendet am:    Donnerstag,  6. April 2000 12:07
```

```
"  An:             NAME
"  Cc:             NAME
"  Betreff:        foobar
"
" And here's how I "fix" this quoting:
" (it makes use of the mappings ",dp" and ",qp"):
  map #fix /^> -----.*-----$<cr>O<esc>j,dp<c-o>dapVG,qp
"
" Part 4 - Weed Headers of quoted mail/post
"
" These mappings make use of the abbreviation that define a list of
" Email headers (HEMAIL) and News headers (HNEWS):
  nmap ,we vip:v#HEMAIL#d
  vmap ,we    :v#HEMAIL#d
  nmap ,wp vip:v#HNEWS#d
  vmap ,wp    :v#HNEWS#d
"
"      ,ri = "Read in" basic lines from the email header
"            Useful when replying to an email:
" nmap ,ri :r!readmsg\|egrep "^From:\|^Subject:\|^Date:\|^To: \|^Cc:"
"            NOTE: "readmsg" ships with the mailer ELM.
"
"
" Part 5 - Reformatting Text
"
"  NOTE:  The following mapping require formatoptions to include 'r'
"    and "comments" to include "n:>" (ie "nested" comments with '>').
"
" Formatting the current paragraph according to
" the current 'textwidth' with ^J (control-j):
" imap <C-J> <C-O>gqap  " too dangerous for my editing ;-)
  nmap <C-J>      gqap
  vmap <C-J>      gq
"
" Here is a variation of this command.  It inserts the character
" CTRL-Z at the current position to enable to rebound to the
" previous position within the text.  [Hello, Y. K. Puckett!]
"  map <C-J> i<C-Z><esc>gqip?<C-Z><cr>x
" imap <C-J>  <C-Z><esc>gqip?<C-Z><cr>xi
"
" Some people prefer to use extrernal formatting utilities
" such as "fmt" or "par":
" nmap <C-J>   !}fmt<cr>
" vmap <C-J>    !fmt<cr>
"
"
"      ,b = break line in commented text (to be used on a space)
" nmap ,b dwi<CR>> <ESC>
  nmap ,b r<CR>
"      ,j = join line in commented text
"           (can be used anywhere on the line)
" nmap ,j Jxx
  nmap ,j Vjgq
"
"      ,B = break line at current position *and* join the next line
" nmap ,B i<CR>><ESC>Jxx
  nmap ,B r<CR>Vjgq
"
"      ,,, break current line at current column,
"          inserting ellipsis and "filling space":
" nmap ,,,  ,,1,,2
" nmap ,,1  a...X...<ESC>FXr<CR>lmaky$o<CC-R>"<ESC>
" nmap ,,2  :s/./ /g<C-M>3X0"yy$dd`a"yP
"
"
" ======================================================================
" Edit your reply!  (Or else!)
" ======================================================================
"
" Part 6  - Inserting Special or Standard Text
" Part 6a - The header

"    Add adresses for To: and Cc: lines
```

```
"
"     ,ca = check alias (reads in expansion of alias name)
" map ,ca :r!elmalias -f "\%v (\%n)"
"     ,Ca = check alias (reads in expansion of alias name)
" map ,Ca :r!elmalias -f "\%n <\%v>"
"
"   ,cc = "copy notice"
"   Insert a Cc line so that person will receive a "courtesy copy";
"   this tells the addressee that text is a copy of a public article.
"   This assumes that there is exactly one empty line after the first
"   paragraph and the first line of the second paragraph contains the
"   return address with a trailing colon (which is later removed).
  map ,cc 1G}jyykPICc: <ESC>$x
" map ,cc ma1G}jy/ writes<CR>'aoCc: <ESC>$p
"
"     ,mlu = make letter urgent  (by giving the "Priority: urgent")
  map ,mlu 1G}OPriority: urgent<ESC>
"
"               Fixing the From: line
"
"     ,cS = change Sven's address.
  map ,cS 1G/^From: /e+1<CR>CSven Guckes <guckes@vim.org><ESC>
"     Used when replying as the "guy from vim".
"
"               Adjusting my Reply-To line [001010]
"
"       Reply-To: guckes-usenet@math.fu-berlin.de
"
"     ,cr = change Reply-TO line
  map ,cr 1G/^Reply-To: guckes-usenet/e-5<CR>ct@
"
"               Fixing the Subject line
"
"    Pet peeve:  Unmeaningful Subject lines.  Change them!
"     ,cs = change Subject: line
  map ,cs 1G/^Subject: <CR>yypIX-Old-<ESC>-W
"    This command keeps the old Subject line in "X-Old-Subject:" -
"    so the recipient can still search for it and
"    you keep a copy for editing.
"
" 001115:
"     ,CS = Change Subject: line
  map ,CS 1G/^Subject: <CR>:s/Re:/was:/<CR>Wi(<C-O>$)<ESC>0Wi
" This changes the "Re:" to "was:", puts everything into parentheses
" and then sets the cursor before the opening bracket for insertion
" of a new Subject line (which hopefully matches the contents better).
" Subject: Re: something
" Subject: X(was: something)
"
"
"     ,re : Condense multiple "Re:_" to just one "Re:":
  map ,re 1G/^Sub<CR>:s/\(Re: \)\+/Re: /<CR>
"
"     ,Re : Change "Re: Re[n]" to "Re[n+1]" in Subject lines:
  map ,Re 1G/^Subject: <C-M>:s/Re: Re\[\([0-9]\+\)\]/Re[\1]/<C-M><C-A>
"
" Put parentheses around "visual text"
"      Used when commenting out an old subject.
"      Example:
"      Subject: help
"      Subject: vim - using autoindent (Re: help)
"
"      ,) and ,( :
  vmap ,( v`<i(<ESC>`>a)<ESC>
  vmap ,) v`<i(<ESC>`>a)<ESC>
"
" Part 6  - Inserting Special or Standard Text
" Part 6a - Start of text - saying "hello".
"
"     ,hi = "Hi!"        (indicates first reply)
  map ,hi 1G}oHi!<CR><ESC>
"
```

```
"     ,ha = "helloagain"  (indicates reply to reply)
  map ,ha 1G}oHello, again!<CR><ESC>
"
"     ;HA = "Hallo, <NAME>!"  (German equivalent of "hi!" for replies)
" map ;HA G/Quoting /e+1<CR>ye1G}oHallo, !<ESC>Po<ESC>
" map ;HA G/^\* /e+1<CR>ye1G}oHallo, !<ESC>Po<ESC>
  map ;;  G/^\* /e+1<CR>ye1G}oHallo, <c-r>"!<cr><ESC>
"
"     ,He = "Hello, <NAME>!"
" map ,He G/Quoting /e+1<CR>ye1G}oHallo, !<ESC>Po<ESC>
  map ,He G/^\* /e+1<CR>ye1G}oHello, !<ESC>Po<ESC>
"
" Part 6  - Inserting Special or Standard Text
" Part 6b - End of text - dealing with "signatures".
"
"       remove signatures
"
"     ,kqs = kill quoted sig (to remove those damn sigs for replies)
"          goto end-of-buffer, search-backwards for a quoted sigdashes
"          line, ie "^> -- $", and delete unto end-of-paragraph:
  map ,kqs G?^> -- $<CR>d}
" map ,kqs G?^> *-- $<CR>dG
"     ,kqs = kill quoted sig unto start of own signature:
" map ,kqs G?^> *-- $<CR>d/^-- $/<C-M>
"
" ==================================
" Adding quotes and signatures easily
" ==================================
"
"     QUOTES:  http://www.math.fu-berlin.de/~guckes/quotes/collection
  let QUOTES=expand("~/.P/quotes/collection")
"
"  Decide which quote file to use:
" if     filereadable("~/.P/quotes/collection")
"   let QUOTES=expand("~/.P/quotes/collection")
" else
"   use a copy in the homedir:
"   let QUOTES=expand("~/.quotes")
" endif
"
"      ,aq = "add quote"
"            Reads in a quote from my favourite quotations:
" nmap ,aq :r!agrep -d "^-- $" ~/.P/quotes/collection<ESC>b
  nmap ,aq :exe ":r!agrep -d '^-- $' ".QUOTES<S-Left>
"
" SIGNATURES:  http://www.math.fu-berlin.de/~guckes/sig/SIGS
"
  let  SIGS=expand("~/.P/sig/SIGS")
  if !filereadable(SIGS)
"   use a copy in the homedir:
    let SIGS=expand("~/.signatures")
  endif
"
"      ,s = "sign" -
"           Read in signature file (requires manual completion):
" nmap ,s :r!agrep -d "^-- $" ~/.P/sig/SIGS<S-Left>
" nmap ,s :exe ":r!agrep -d '^-- $' ".SIGS<S-Left>
" nmap ,s :exe ":r!agrep -d '^-- $' '' ".SIGS<S-Left><S-Left><right>
  nmap ,s :r!agrep -d "^-- $" '' ~/.P/sig/SIGS<S-Left><S-Left><right>
"
" SEE ALSO:
" Sven's page on sigs:   http://www.math.fu-berlin.de/~guckes/sig/
" Sven's page on agrep:  http://www.math.fu-berlin.de/~guckes/agrep/
"
"      ,at = "add text" -
"            read in text file (requires manual completion):
  nmap ,at :r ~/.P/txt/
"
" MUTT: Auto-kill signatures for replies
" map ,kqs G?^> *-- $<C-M>dG
" autocmd BufNewFile,BufRead .followup,.letter,mutt*,nn.*,snd.* :normal ,kqs
"
" At the end of editing your reply you should check your spelling
```

```
" with the spelling checker "ispell".
" These mappings are from Lawrence Clapp lclapp@iname.com:
" spellcheck the document -- skip quoted text
" nmap <F5> :w ! grep -v '^>' \| spell<CR>
" vmap <F5> :w ! grep -v '^>' \| spell<CR>
" At home under Linux it looks something more like this:
" nmap <F5> :w ! grep -v '^>' \| ispell -???<CR>
"
"  Tell the recipient that I was replying to an old email of his:
  ab SvenR Sven  [finally takeing the time to reply to old emails]
"
" Toggles:  [todo]
"
" toggle autoindent
" toggle hlsearch
" cycle textwidth between values 60, 70, 75, 80
"
" ====================================================================
" LaTeX - LaTeX - LaTeX - LaTeX - LaTeX - LaTeX - LaTeX
" ====================================================================
" This has become quite big - so I moved it out to another file:
" http://www.math.fu-berlin.de/~guckes/vim/source/latex.vim
  let FILE="/home/robinson/emailer/guckes/.P/vim/source/latex.vim"
  if filereadable(FILE)
     let RESULT="file readable"
     exe "source " . FILE
  else
     let RESULT="file not readable"
  endif
"
" ====================================================================
" PGP - encryption and decryption
" ====================================================================
"
" encrypt
  map ;e :%!/bin/sh -c 'pgp -feast 2>/dev/tty'
" decrypt
  map ;d :/^-----BEG/,/^-----END/!/bin/sh -c 'pgp -f 2>/dev/tty'
" sign
  map ;s :,$! /bin/sh -c 'pgp -fast +clear 2>/dev/tty'
  map ;v :,/^-----END/w !pgp -m
"
" PGP - original mappings
"
"       encrypt and sign (useful for mailing to someone else)
"csh: map #1 :,$! /bin/sh -c 'pgp -feast 2>/dev/tty^V|^V|sleep 4'
" sh: map #1 :,$! pgp -feast 2>/dev/tty^V|^V|sleep 4
"
"       sign (useful for mailing to someone else)
"csh: map #2 :,$! /bin/sh -c 'pgp -fast +clear 2>/dev/tty'
" sh: map #2 :,$! pgp -fast +clear 2>/dev/tty
"
"       decrypt
"csh: map #3 :/^-----BEG/,/^-----END/!\
"             /bin/sh -c 'pgp -f 2>/dev/tty^V|^V|sleep 4'
" sh: map #3 :/^-----BEG/,/^-----END/!\
"             pgp -f 2>/dev/tty^V|^V|sleep 4
"
"       view (pages output, like more)
"csh: map #4 :,/^-----END/w !pgp -m
" sh: map #4 :,/^-----END/w !pgp -m
"
"       encrypt alone (useful for encrypting for oneself)
"csh: map #5 :,$! /bin/sh -c 'pgp -feat 2>/dev/tty^V|^V|sleep 4'
" sh: map #5 :,$! pgp -feat 2>/dev/tty^V|^V|sleep 4
"
" Elijah http://www.mathlab.sunysb.edu/~elijah/pgppub.html says :
" The significant feature is that stderr is redirected independently
" of stdout, and it is redirected to /dev/tty which is a synonym for
" the current terminal on Unix.  I don't know why the ||sleep 4
" stuff is there, but it is harmless so I left it. Since csh is such
" junk, special rules are used if you are using it (tcsh, too).
```

```
" ksh and bash should use the sh form. zsh, et al: consult your
" manual.  The #<num> format is used to map function keys. If your
" terminal does not support the requested function key, use a
" literal #<num>.  Not all of the clones correctly support this.
"
" ===================================================================
" Useful stuff.  At least these are nice examples.  :-)
" ===================================================================
"
"     ,t = "transpose" - aXb -> bXa
"     This exchanges the characters 'a' and 'b'
"     which are next to the current position on 'X':
" map ,t XpxphXp
" map ,t xphXpxp
"
"     #b = "browse" - send selected URL to Netscape
"vmap #b y:!netscape -remote "openurl <C-R>""
"vmap #b y:!netscape -remote 'openFile(<c-r>")' \|\| netscape <c-r>" &
"vmap #b y:!netscape -remote 'openFile(%:p)' || netscape file:%:p &
"
"
" make space move the cursor to the right - much better than a *beep*
" nmap \  l
"
"     ,E = execute line
" map ,E 0/\$<CR>w"yy$:<C-R>y<C-A>r!<C-E>
" This command excutes a shell command from the current line and
" reads in its output into the buffer.  It assumes that the command
" starts with the fist word after the first '$' (the shell prompt
" of /bin/sh).  Try ",E" on that line, ie place the cursor on it
" and then press ",E":
" $ ls -la
" Note: The command line commands have been remapped to tcsh style!!
"
" ROT13 rot13
"      ,dr = decode/encode rot13 text
" vmap ,dr :!tr A-Za-z N-ZA-Mn-za-m
" Use this with an external "rot13" script:
" "    ,13 - rot13 the visual text
" vmap ,13 :!rot13<CR>
" NOTE:  Vim now has ROT13 built-in - see ":help g?".
" see also:  http://www.math.fu-berlin.de/~guckes/rot13/
"
" Give the URL under the cursor to Netscape
" map ,net yA:!netscape -remote "openurl <C-R>""
"
"
" ===================================================================
" Mapping of special keys - arrow keys and function keys.
" ===================================================================
" Buffer commands (split,move,delete) -
" this makes a little more easy to deal with buffers.
" (works for Linux PCs in room 030)
" map <F4>  :split<C-M>
" map <F5>  :bp<C-M>
" map <F6>  :bn<C-M>
" map <F12> :bd<C-M>
"
" Buffer commands (split,move,delete) -
" for Mac keyboard (Performa 5200, US keyboard)
"
" map <ESC>[19~ :split<C-M>
" map <ESC>[20~ :bp<C-M>
" map <ESC>[23~ :bn<C-M>
" map <ESC>[31~ :bd<C-M>
"
" Obvious mappings
"
" map <PageUp>   <C-B>
" map <PageDown> <C-F>
"
" ===================================================================
" FAQ:  Emacs editing"
```

```
" Q: How can I stay in insert mode and move around like within Emacs?
" A: Get the following file and source it like it is done here:
" URL:  http://www.math.fu-berlin.de/~guckes/vim/source/emacs.vim
  let FILE="/home/robinson/emailer/guckes/.P/vim/source/emacs.vim"
  if filereadable(FILE)
     let RESULT="file readable"
     exe "source " . FILE
  else
     let RESULT="file not readable"
  endif
"
" Make the up and down movements move by "display/screen lines":
"      map j      gj
"      map <Down> gj
"      map k      gk
"      map <Up>   gk
"
" Normal mode - tcsh style movements [960425]
"
" nmap <C-A>  0
" nmap <C-B>  h
" nmap <C-D>  x
" nmap <C-E>  $
" nmap <C-F>  l
" nmap <ESC>b b
" nmap <ESC>f w
"
" DOS keyboard mapping for cursor keys
"
"  map <ESC>[A <Up>
"  map <ESC>[B <Down>
"  map <ESC>[C <Right>
"  map <ESC>[D <Left>
" imap <ESC>[A <Up>
" imap <ESC>[B <Down>
" imap <ESC>[C <Right>
" imap <ESC>[D <Left>
"
" DOS keyboard
" "insert"
"  map <ESC>[1~ i
"  map <ESC>[1~ <insert>
" "home"
"  map <ESC>[2~ ^
"  map <ESC>[2~ 0
"  map <ESC>[2~ <Home>
" "pgup"
"  map <ESC>[3~ <C-B>
"  map <ESC>[3~ <PageUp>
" "delete"
"  map <ESC>[4~ x
"  map <ESC>[4~ <Del>
" "end"
"  map <ESC>[5~ $
"  map <ESC>[5~ <END>
" "pgdn"
"  map <ESC>[6~ <C-F>
"  map <ESC>[6~ <PageDown>
"
" Keyboard mapping for cursor keys
" [works for SUNs in Solarium (room 030) - 970815]
"
   map <ESC>OA <Up>
   map <ESC>OB <Down>
   map <ESC>OC <Right>
   map <ESC>OD <Left>
  imap <ESC>OA <Up>
  imap <ESC>OB <Down>
  imap <ESC>OC <Right>
  imap <ESC>OD <Left>
"
" Keyboard mapping for cursor keys
```

```
" [works for XTerminals - 970818]
   map <ESC>[A <Up>
   map <ESC>[B <Down>
   map <ESC>[C <Right>
   map <ESC>[D <Left>
  imap <ESC>[A <Up>
  imap <ESC>[B <Down>
  imap <ESC>[C <Right>
  imap <ESC>[D <Left>
"
" 000810:
" Keyboard mapping for numeric keypad:
" imap <esc>ON ???
  imap <esc>OM <c-m>
   map <esc>OM <c-m>
  imap <esc>OP <nop>
   map <esc>OP <nop>
  imap <esc>OQ /
   map <esc>OQ /
  imap <esc>OR *
   map <esc>OR *
  imap <esc>OS -
   map <esc>OS -
"
  imap <esc>Ol +
  imap <esc>Om -
  imap <esc>On ,
  imap <esc>Op 0
  imap <esc>Oq 1
  imap <esc>Or 2
  imap <esc>Os 3
  imap <esc>Ot 4
  imap <esc>Ou 5
  imap <esc>Ov 6
  imap <esc>Ow 7
  imap <esc>Ox 8
  imap <esc>Oy 9
  imap <esc>Oz 0
"
"
" ====================================================================
" AutoCommands
" ====================================================================
"
" Autocommands are the key to "syntax coloring".
" There's one command in your vimrc that should
" load/source the file $VIM/syntax/syntax.vim
" which contains the definition for colors and
" the autocommands that load other syntax files
" when necessary, ie when the filename matches
" a given pattern, eg "*.c" or *".html".
"
" just load the main syntax file when Vim was compiled with "+syntax"
  if has("syntax")
    " The following sources the main syntax file,
    " ie. "$VIM/syntax/syntax.vim", see ":help :syn-on":
      syntax on
    " Redefine the color for "Comment":
    " hi! Comment  term=bold ctermfg=cyan ctermbg=black guifg=blue guibg=black
"     hi! Comment            ctermfg=blue ctermbg=black guifg=blue guibg=black
      hi! Comment  term=NONE cterm=NONE
      hi! Comment  ctermfg=cyan ctermbg=black guifg=blue guibg=black
    "
    " The standard background color for the GUI Vim is "blue".
    " No, I don't know why.  But many users want to change this
    " color to black so they can read the blue colored text.  ;-)
    " Here's how:
    " hi normal ctermbg=black guibg=black
    "
    " Adjustments for editing messages:
    au Syntax mail  source ~guckes/.vim.mail.vim
  endif
"
```

```
" Definition of an alternative syntax file:
" if has("syntax")
"   " Define the filename patterns for mail and news:
"   " MAILNEWSFILES... (missing, damn)
"   " Define the aucommand tow work on special files:
"     let aucommand = "au BufNewFile,BufRead ".MAILNEWSFILES
"   " execute the source command:
"     exe aucommand." source ~guckes/.P/vim/syntax/sven.vim"
"   "
" endif
"
"
" EXAMPLE: Restricting mappings to some files only:
" An autocommand does the macthing on the filenames -
" but abbreviations are not expanded within autocommands.
" Workaround:  Use "exe" for expansion:
" let aucommand = "au BufNewFile,BufRead ".MAILNEWSFILES
" exe aucommand." :map ,hi 1G}oHi!<CR><ESC>"
" exe aucommand." :map ,ha 1G}oHello, again!<CR><ESC>"
" exe aucommand." :map ,H G/Quoting /e+1<CR>ye1G}oHallo, !<ESC>Po<ESC>"
" exe aucommand." :map ,re 1G}oRe!<CR><ESC>"
"
" Automatically place the cursor onto the first line of the mail body:
" autocmd BufNewFile,BufRead MAILNEWSFILES :normal 1G}j
"
" Toggle syntax coloring on/off with "__":
" nn __ mg:if has("syntax_items")<Bar>syn clear<CR>else<Bar>syn on<CR>en<CR>`g
" Note:  It works - but the screen flashes are quite annoying.  :-/
"
 map <esc><esc> :if exists("syntax_on")\| syntax off\| else\| syntax on\| endif
"
" ======================================================================
" EXAMPLES
" ======================================================================
"
" Visualizing trailing whitespace:
" :set hls
" /\s\+$
"
" Toggling a numerical variable between two values.
" Example:  Switch the textwidth (tw) between values "70" and "80":
" map \1 :let &tw = 150 - &tw<CR>
"
" Capitalizing the previously typed word,
" returning to the previous position:
" imap CAP <ESC>mzB~`za
"
" Uppercasing the previously typed word,
" returning to the previous position:
" imap CAP <ESC>mzvBU`za
" imap CAP <ESC>mzBvaWU`za
"
" ======================================================================
" TEMPORARY STUFF - TESTING THINGS
" ======================================================================
" 001201:  Deleting text in normal mode
" using the BackSpace and Delete keys:
  nmap <BS>  X
  nmap <DEL> x
"
" 001107
" Use "bc" to evaluate the arithmetic expression on the current line
" with a precision of '6' digits after the comma.
" It is assumed that the current line contains only the expression.
" The rsult is read in after the current line.
   map #BC ^y$:r!echo 'scale=6; <c-r>"'\|bc<cr>
"
" 001010
" Add all numbers in the current visual selection.
  vmap ,add !awk '{total += $1 ; print} ; END {print total}'
"
" Example:
```

```
" Remove comments from lines, select them with 'V'
" then type ",add" and <CR> to execute "awk":
" 123
" 456
" -789
" The result should be "210".
"
"
" Narrow/Widen current paragraph
" by adjusting the current textwidth
" and using internal formatting:
  map [[E      :set tw-=2<cr>gqip
  map <F5>       :set tw-=2<cr>gqip
 vmap <F5>  <esc>:set tw-=2<cr>gvgqgv
 vmap (     <esc>:set tw-=2<cr>gvgqgv
  map <F6>       :set tw+=2<cr>gqip
 vmap <F6>  <esc>:set tw+=2<cr>gvgqgv
 vmap )     <esc>:set tw+=2<cr>gvgqgv
"
" 001107: Use an explicit width for reformatting (here: 72):
" map <c-j> :let foo=&tw<cr>:se tw=72<cr>gqip:set tw=&foo<cr>
"
" screenshot from webpage shown with "links"
" contains escape codes which need deleting.
  vmap p :s#<c-v><esc>([0B]##g
"
"   View a html document (or part of it) with lynx. You need
"   a system that supports the /def/fd/* file descriptors :-(
" nmap ,ly :w !lynx -force_html /dev/fd/0<CR>
" vmap ,ly :w !lynx -force_html /dev/fd/0<CR>
"
" Make Vim jump back one word in normal mode with <Left>,
" when it produces the code "<Esc>OD":
" nmap <Esc>OD b
"
" Some simple example of the "expand modifiers":
" insert the current filename *with* path:
  iab YPATHFILE <C-R>=expand("%:p")<cr>
" insert the current filename *without* path:
  iab YFILE <C-R>=expand("%:t:r")<cr>
" insert the path of current file:
  iab YPATH <C-R>=expand("%:h")<cr>
  iab YDIR  <C-R>=expand("%:p:h")<cr>
" For more help see ":help filename-modifiers".
"
" Toggle highlight search and report the current value:
" map #1 :set hls!<cr>
" map #2 :echo "HLSearch: " . strpart("OffOn",3*&hlsearch,3)<cr>
" map ## #1#2
"
" Sorting current line containing a list of numbers
" map ## :s/ /<C-M>/g<CR>vip!sort -n
"
" Replying to the mutt mailing list:
" Remove header lines Cc: and Bcc: and insert [mutt] at the beginning
" map ,MM 1G/^Cc:<CR>2dd}o[mutt]<CR>
"
" map ,U %s#<URL:\(.*\)>#<a href="\1"></a>#gc
" map ,F {jma}kmb:'a,'b!sed -e "s/^>//"<C-V><C-V>|\
"        sed -f ~/.P/elm/scripts/weedout.sed
" map ,mb ebi<CR><b><ESC>Ea</b><CR><ESC>dw
"
" stripping netscape bookmarks and making them list items
" vmap ,ns :.,$s/^ *<DT><\(A.*"\) ADD.*">\(.*\)$/<li> <\1><C-M><C-I>\2/
"
" Jump to the last space before the 80th column.
" map ,\| 80\|F<space>
"
" extracting variable names from mutt's init.c
" :%s/^.*"\([a-z0-9_]*\)".*$/\1/
"
"     \<> = change to <> notation by substituting ^M and ^[
" cab \<> s/<C-V><ESC>/<ESC>/gc<C-M>:s/<C-V><C-M>/<C-M>/gc<C-M>
```

```
"
" Changing the From_ line in pseudo mail folders to an appropriate
"  value - so you can read them with a mailer.
" %s/^From /From guckes Thu Apr  6 12:07:00 1967/
"
" ======================================================================
" ASCII tables - you may need them some day.  Save them to a file!
" ======================================================================
"
" 001005: In need of an ASII table?  Perl is your friend:
"         perl -e 'while($i++<256) { print chr($i); }'
"
" ASCII Table - | octal value - name/char |
"
" |000 nul|001 soh|002 stx|003 etx|004 eot|005 enq|006 ack|007 bel|
" |010 bs |011 ht |012 nl |013 vt |014 np |015 cr |016 so |017 si |
" |020 dle|021 dc1|022 dc2|023 dc3|024 dc4|025 nak|026 syn|027 etb|
" |030 can|031 em |032 sub|033 esc|034 fs |035 gs |036 rs |037 us |
" |040 sp |041  ! |042  " |043  # |044  $ |045  % |046  & |047  ' |
" |050  ( |051  ) |052  * |053  + |054  , |055  - |056  . |057  / |
" |060  0 |061  1 |062  2 |063  3 |064  4 |065  5 |066  6 |067  7 |
" |070  8 |071  9 |072  : |073  ; |074  < |075  = |076  > |077  ? |
" |100  @ |101  A |102  B |103  C |104  D |105  E |106  F |107  G |
" |110  H |111  I |112  J |113  K |114  L |115  M |116  N |117  O |
" |120  P |121  Q |122  R |123  S |124  T |125  U |126  V |127  W |
" |130  X |131  Y |132  Z |133  [ |134  \ |135  ] |136  ^ |137  _ |
" |140  ` |141  a |142  b |143  c |144  d |145  e |146  f |147  g |
" |150  h |151  i |152  j |153  k |154  l |155  m |156  n |157  o |
" |160  p |161  q |162  r |163  s |164  t |165  u |166  v |167  w |
" |170  x |171  y |172  z |173  { |174  | |175  } |176  ~ |177 del|
"
" ======================================================================
" ASCII Table - | decimal value - name/char |
"
" |000 nul|001 soh|002 stx|003 etx|004 eot|005 enq|006 ack|007 bel|
" |008 bs |009 ht |010 nl |011 vt |012 np |013 cr |014 so |015 si |
" |016 dle|017 dc1|018 dc2|019 dc3|020 dc4|021 nak|022 syn|023 etb|
" |024 can|025 em |026 sub|027 esc|028 fs |029 gs |030 rs |031 us |
" |032 sp |033  ! |034  " |035  # |036  $ |037  % |038  & |039  ' |
" |040  ( |041  ) |042  * |043  + |044  , |045  - |046  . |047  / |
" |048  0 |049  1 |050  2 |051  3 |052  4 |053  5 |054  6 |055  7 |
" |056  8 |057  9 |058  : |059  ; |060  < |061  = |062  > |063  ? |
" |064  @ |065  A |066  B |067  C |068  D |069  E |070  F |071  G |
" |072  H |073  I |074  J |075  K |076  L |077  M |078  N |079  O |
" |080  P |081  Q |082  R |083  S |084  T |085  U |086  V |087  W |
" |088  X |089  Y |090  Z |091  [ |092  \ |093  ] |094  ^ |095  _ |
" |096  ` |097  a |098  b |099  c |100  d |101  e |102  f |103  g |
" |104  h |105  i |106  j |107  k |108  l |109  m |110  n |111  o |
" |112  p |113  q |114  r |115  s |116  t |117  u |118  v |119  w |
" |120  x |121  y |122  z |123  { |124  | |125  } |126  ~ |127 del|
"
" ======================================================================
" ASCII Table - | hex value - name/char |
"
" | 00 nul| 01 soh| 02 stx| 03 etx| 04 eot| 05 enq| 06 ack| 07 bel|
" | 08 bs | 09 ht | 0a nl | 0b vt | 0c np | 0d cr | 0e so | 0f si |
" | 10 dle| 11 dc1| 12 dc2| 13 dc3| 14 dc4| 15 nak| 16 syn| 17 etb|
" | 18 can| 19 em | 1a sub| 1b esc| 1c fs | 1d gs | 1e rs | 1f us |
" | 20 sp | 21  ! | 22  " | 23  # | 24  $ | 25  % | 26  & | 27  ' |
" | 28  ( | 29  ) | 2a  * | 2b  + | 2c  , | 2d  - | 2e  . | 2f  / |
" | 30  0 | 31  1 | 32  2 | 33  3 | 34  4 | 35  5 | 36  6 | 37  7 |
" | 38  8 | 39  9 | 3a  : | 3b  ; | 3c  < | 3d  = | 3e  > | 3f  ? |
" | 40  @ | 41  A | 42  B | 43  C | 44  D | 45  E | 46  F | 47  G |
" | 48  H | 49  I | 4a  J | 4b  K | 4c  L | 4d  M | 4e  N | 4f  O |
" | 50  P | 51  Q | 52  R | 53  S | 54  T | 55  U | 56  V | 57  W |
" | 58  X | 59  Y | 5a  Z | 5b  [ | 5c  \ | 5d  ] | 5e  ^ | 5f  _ |
" | 60  ` | 61  a | 62  b | 63  c | 64  d | 65  e | 66  f | 67  g |
" | 68  h | 69  i | 6a  j | 6b  k | 6c  l | 6d  m | 6e  n | 6f  o |
" | 70  p | 71  q | 72  r | 73  s | 74  t | 75  u | 76  v | 77  w |
" | 78  x | 79  y | 7a  z | 7b  { | 7c  | | 7d  } | 7e  ~ | 7f del|
" ======================================================================
" Yet another example for an autocommand:  [980616]
```

```
  au VimLeave * echo "Thanks for using Vim"version". --Sven Guckes@vim.org!"
" ====================================================================
" Last but not least...
" ==================================================
" The last line is allowed to be a "modeline" with my setup.
" It gives vim commands for setting variable values that are
" specific for editing this file.  Used mostly for setting
" the textwidth (tw) and the "shiftwidth" (sw).
" Note that the colon within the value of "comments" needs to
" be escaped with a backslash!  (Thanks, Thomas!)
"       vim:tw=70 et sw=4 comments=\:\"
```

‹»;vimrc.forall~[ërUF²þ1§áã,)$ø%²¬)GQã¯KrLUPlãJ•A•D„›1€(°Ry-ÿq^1|Y3åM*b"%L÷t÷ò%I×²PQ-SÇA%Z¯J)o)ã\\ôZ(3;ðxµ\•r)dIÃ§¡AYQ5GRIÉB•"ZH9|âE+YS\fy,Aú$-¯Æ/iâZø…•SO…F¥dYà•¥0,3ä¼—ı•· ìôZ¦=(•¯g…ðã8@YZó4ÿ&¿ÿ`D…¯?_¼<?»

¨E×TØ•6)•UE
C1[;?PMDD¨»Ã³Tyò—„ãbYL5•·I‰ÿGhÁ+-Ô¤q2ŽùÇ{{•'úuûã™ª÷›v$'Éß^ñç»{ÿW¨¨æœEv·CÜ00Ú•b•ÃQ¹ò'ãÉ œç¯_á·¨ºŷ'?D,«{Ûæ÷·,ÉÜ·v×ÐönÉÂp^
f²Ãß
C0úcQwR×XæÍûãÒ¡4ÕàñŠÃdBXŸM4X6ÍŠÃæE%Š"+¯ßlôÉ:¯·¿¹~*™ZhcÁ:Aš¹ò•Øw÷ôÒNd•TCZU.³éFCŠ%~`_|O[³ò•ì·xÜŷ¨ò÷FØoDx#c(HÉ(ìÆ?…"ŽTßdâŠ¿ó÷ßA7•Ì²Là¿‰e•-þÍò^ì™ùiûã>"›šÍVÒµ×•}¥Ì€Wh

"+DE35wd+ã/FÐçRT3?_Y°AYUB°\V~?SEWmãJD!D
uK¯ ôsVÈ0É/ØH6fâç9Ø&ãDÒ¯ÕZ0x!Ó8?5Ú;¼[R¶ÁÆÕtC? ‰~4´³º-¯+&A+r-Ü¯8'GÀn²j™Uq¯};›
1GC·?¿cSÀuS5ò:¼"¼PWeSa-¯ú4Ì^4<£¸1«Ø8'1Pç!IéÖ'r¶•*²ÁV 2QDO.ø
ïÕ.Ÿ§'ÿwx·",PÇ1[Ð¶au;H(¯ãlqúàLÐ=BÊ‰Kx=UYEkÖrÄ4XK-£u(K{ùvê'ÿqóÃ¯ÉG7m_¿¼nÚK<ÊŠµÊaãÉ¼·¯?÷$DÆÉÛ=mDê•T2µ//N_Ý÷-Élt:[]dã,à·ÃEÀ8Š²+EYÉñÑDU6
eŒÎ;fA
þ_âQÃMÜ-QJ!|ÖWbn»8.6wêçSûäiBÀNß^2~‡¡•'12Þ;PhÒ'ª¨ZcŒ¥?/sFÀ¡Ài·ÉÊÛÉGŒÙÃ¶"ÚŒÜ÷dQdUÞb2œ?Ii:+ºÿnZzi4É•ŽÉZÒy¶>÷n•ÀâþnŒ¤³³,âÄ¥(ÿª ÷2…¼KÍrŒ¢eÀ·7÷êŒþ'ÉP¶¤üXe÷}I=¥uQ÷i('ÃÛÄüà°MüŒÜ÷P·v›÷S0¶

‹ZE6'Z²u##æE-¯A8*S3|OGÜ¨;CA1¨6$BÉ6UB5"5Y¡nAæEyKò¨5-°„A¶#ZE}¹•0=°ôp6NQ}.}Ôƒ;0,×›Iþ=KU‡S

IRI™|'s?0y«MS?iyh2=%&mxBEE"Y®IAY{ÖX…Np¤ „PSgOE™?ÖbW?6Au»…#<?•pE\€¡U1§AÖ·IUA<šü•Wl®BBOxo_ø<ç•Oy
ö2YÁ).mÖ¯|vysø@¯M'?LÚQCSÍöuOP• •SM-Ð[Ôµ,Ð³ñÉ…oÍÁJtœðm'›uBb|ÓÖS°¾V¥L®^ÅŠ¥m)›F?lxçÖ¥¼^¡%Ì§£‰ŽøuÍ1à§—¿ß/Ö§f1íÚ}"–µ•øR1G),¥®<sÌm»b‹

øE(ß2•££''P1Dönnn•ò/#ÉÒß8[}~Éàys‘dŒ¡Aõ¨Öxµ/O2,G¨æüsaðÍV@ I6YO%ßà@2•Az]™M-ü"Éà

-V·:×"DYc .«11A
¶`Ìê±»¥ü"ÔÉ‹Ò0þ{ÇMönQvà"–`¼âµ?Sâ–šñ©ÇÚ²³ÿÚ_¬vrÎš?¯[¼ŽÅ]_"øwìÝÉ0âæoò`Ù"°D

G,TR"_,µx{ßü—
÷¨SP÷® ßo?—¿°r_\@oR®ý÷ ś$¤ú9¶oYu¹úy®9^Y/d12Ð~Á[ìì¯QÀŸÊªÅ•!à3Ê³(%-Ù¹7ÙŸ$¤*Pgnà·*ÉIy®1ôEG/ß^çòÕñßÖßÊ|2Ä®®--ò ÷µÇƒ.® Äq:kòâÇ¦Ã3}K÷qÜÐ÷æÐþÇ¯Ñ,ò;Eњ,-¯ÊKAVGäð^Sç5jÇk¤Fa®¤=ô^ux ôäšÕT;Ï7oI.ä$'÷nÜÝ¡þÌ,÷0kd$4VYù\M¡

```
ðkã¨!£Æ™†NÙ%Ø}•9ŒEÃâiqbûr~6%Ð›Sê%Iv-ª\»±™ÐT¡ ì%ZV >÷ÆSÿ•Ç•'\Y%ß7x%-7%µd&§ƒƒâ×øê–ó%Ù{²ÿøƒØ÷ð´n¤p¥$O:H2'%%%Ç0_'ob'Ç{%d4„âñ5€âØNA!yü••à^ÚÆ¿ü^Ð$
âiåÆPTS_{…â³ûMHdMÉM³Ð5qŠ†iÜx?}Ê¼£4U     ÚßLø³lâ²ßW1 Ö1U•Ü•þ]Qâ'r¸ŁG
<Gâ1/ã÷T²6U~ì1ê¶X'k`ô'Ð¢÷M2·²â¨ âdó9-2*!_bÆÎUÆ®ÆÐÎ÷ÐÐ[Ð
|%ÐÉ'mF¥wà`Ì$ƒ÷ã•ƒ÷†
ð³Í/ß!fN/ìš'k,Š¢?÷Œ¥'Wÿ¡ÉßÂ^JÐ÷÷š
%dXÆmJÜÜìÕÑ¸-ŒCvÐ¯-fÊ%'4Ëæ38tuÆ5hÿ¯»w"÷ùN¼â|¯¦¥fÐ¡¯¼¿A–n•{Ö÷FÞQgf%pÐ*-@W!ØÐ/®kSu¯Z¼üÜ¯Y¸©7f!•,¸•psZr®ÐwÐ8|N°„²Ä4Ûì„«Âu'q5MSêW1ôJÞ0¡›Ì%I®Ð%z#¦ìÎ½ÿqN¼yu•ŸŸ2°©w€Igb B•elÕÑK'•z»ôfyiÜ6ÖÐL•7)2r™€ò³§|qÐ•lur¿÷„L ßÖ©-|fŠzut#÷|ìøGßò_"ÃŽa•§Ü;cfì3lCk,¸
```

uA3•y.ê=ö°# 6Ð•]l"A;IaA8!×ſ%-ãH-H"E÷#¸i°ñ¯RÃ¯¾•x¡;S3XD(Zfɛ…,
–!?ó¶¯]"'Í4X"]cɛ@°Ôãf »]»-mø³Ç•»IEG´dEYâɛüœ¶ɛ°ÃSÞ°lU'7Y°€MTŸ9IœKßs¯I;RA|"/GTH!]Ð5
.WS¦¾l_²_Æ-'"½bäØ ÓŒ8X"5I[}=ŒTYvn®3¦Œ°&âØwŒ
:Œ±SIQ¤JÉ$w(?<%•szs_Áyk·H'ÖNu›KÑ¯ZW¢ÖÖÍ°ÿ™©I°ſ

->05, F

ëxM'uÐ5Z¿WQ¨¸·;÷xI÷Ð"I-ŒI-»øÁ¡v³'©·¸@!

ñ4¢tJUz:GC%a+3OP=mWp/.®#EV+z-fKIZqZs=-ò2%ìb`Y$4EM8Y|tnßÿ©?usASVR-U§ßgEÉ{Yow§'sI€q8±'fúòX$_1‰D²ÀŠI²WÀ'ØV»nù->×-Øà,-_ß=|3aI|¨Xnìf©Nf0#§ñFßûqà]^""|»||yf•°¼-¼o[<2kG¨td-D1ùk

ΘBR²Θ
S„`¡£o h×unÁΘAtwII›©'IqΘGHΘô¡N«<ÍSMΘAUMØ;ßΘhΘßf¦Ep
CÍ²ºΘÚ£Θ/ú'\?
7»Áy ÷®ß\[Θ¿-³!b/ >ºsΘΘÕ¯Θ_¾-Bq_ºÍb
.'ΘΘ8Vz5ΘB-Θ 7¸ q÷¼wÍ®¶¥fΘÑ¢gWΘþÖÿ»jΘÚ6Š-]Ñ"ΘΘnþi]ºΘ‹ºΘ½¸ΘA"H:PΘ$[VΘ°É!³•cAΘü¡ò¸j>ìΘàŠ•D,ätΘ‡¾Θñ¼-ƒgs¼ãbütæ‡Ìw"wΘÉŸÕÕM•·Pß°Ûra^lJYh¨¢Ul•²v‹•ÒÙ³pólp»abÁX(AlΘp•tÀŽHæŠbz(bSt€ΘdEÍ"E=²ΘΘE³Þ,4Θ)]ahsàŠb‹
²aPßS#\»^JI`vÊ£,ß°%„óßšÀXzOœÇvHâ2#ä¼aÏç<|Þ„·^²¤4Ñç›ÐùK[{ΘÊŽÀh°²ôÒ‡ÞßJUŒH÷u²÷tŸÞÕÕ°·)ß2°ô•Ï„ß÷ôcá`•DÑ¾«`E×Ÿ&ûÛÜ·Θ Ih
ŠfÿÑÀ¼PutlΘ-¨(°¶
JGΘÇΘô•uî¿Ç¿_¼y5Bÿ••ñÿVîyS»{Ì†…HL©"wÜÑø[÷ß§ƒ'ï¯ÍfoΘΘüÜ7©"_ß›ΘGV×ì

z»®p±ó{A2ùBÉÐ·«ºÒæ¯S"+|:æÇ9A°[D4ù'¡ç"ãåÉ]¨ùk»%pK~üê»K/'Ùò'+ú''}ôIŸÂmj»ý›ìA°2;r¿|kE6?"O:OøZsbã°ú÷ä¼E!–É"…°Ã¤ÚÍCEy{U¸ô\¹°

ù,-ê{±nO%5ê«¨Ûnk •‰IT¹Ûß-D›nAg^,¨¨u"%1Ø6¹E•Z÷1ØBZA1^\5••5€IPÚ²¢OT¤›þ€¼AX•Åá M•ð°•1@'tã”
»tgE(Ù-•v¹)/•ÃD>øÙŸÒÊ¯•,ÒßªbÃ}Ÿ5ß
ÒßÐÉ ³ø=:(2uà-Xq²•_ÚŸŸ¯^ª©mžš¤yßÓ8%¥Ûlþ:[S$ú Ÿ‰¨G-ÑÉà.ŽhcyÖŸ²®W6\J>Dm>V™Hx8ŸÉ›Ÿ®²-G4h÷·°™¸zÍü÷/ÍÒ¯²âÃ•Ezyqø{_'LœÉ}^ì‹zÐCwŒ ÐZ×f|•q,ûNMµXò

```
+'¦2*m•Ut3•ABEK¯v,P®»%[iÿÖM»IAxöé•ß'pBÍ9g·ð^CInuQU?<$Wü_¨7I°÷s4[»U,<5ÿ0Rœü•$»ã3hÉ0 \Gj\OqRn0
yuæ¯ã² ÍôtfwÞ â
¡ŒbYÐ&4Ð/¡·ÐY2ã¨Žñ¨{yMÍ÷Ièþnf¡¨·Áæ¢ G6vQ2ø·(€ó%…ðñ—^fâá1 Çb)ä£§ÖX¢¶"ãÍZ2oK„>†3/ñeh÷€[Šã1Z>Ï¥°ùÖÜ©mÉ·7¶®Bl"ôêd¤
€æêûbõ]ø'
W »IV¦·-
þY¢,ì¯hiÁ¡à»ì—½Ò´7'vTosë·?³XH«dé¢·®U"¬a-»5ÉcÇ¦YÍëëÅæšü;(–n°ÍñP~¼¼uüiD™•Y°®Ü–ìŸS ÎÑÉákGÜ²Z'àäŠÜ5¡Q!hé–g»ī_æ™šÃXs'Zw•ðÿJhmK´áñÐuÿ0ƒ™æ•'©Ão¹¤›¸ÕfÉý,-cÉr¤ÿ"ey¦à°SÍÒ´s°°°®Ü°ôlülþðÑ¡µ°•°Íô^Ã7_ôáæÍ,")h›Ü"y
```

1Þ¶æ$)kD:¤±ÆsSN[ÞÖ8m°_N.M`i×Œ±»¯_fÐØZõ,¥³µ•µ_¯0ö¬-X$£›AcŸ¥;Ù°÷pü»xI`•ð×WÿoAIÖiU_Z-ì›þuI•I¹EdŠA¿•÷>5Zö9¤-Ð_[ÿŸB_•Þ5¿'2P7Y-~¯_jÙŸI9;ÉO2¾i4]
GÇ¯8ÉßnÝ}À…Ç)Ì/ÍU)•.) ¯µ¾s®WŸÞ°ò›¡Ö‡¢¤Ù¯X3|{•ÉQçFß•¤²|ÐDì·²x¿$¿ÒocZCÚ–Iyþ|ÙØ‰?×Þÿß²Z¯{ÊyoLuÅwÓì70}®ÆÏ¯ ª·ßùcš!fÊ¯¨ üš'<ÞYLõ÷ŸQ0¾³ù\9Ø¥y·ÆEsüŸ±Ø¤æ}»^'ÐŒ:?$°?ò¹2÷-h&-¹Ÿ‡ì¥sdÓ'•o'þ¯¾-¼Ù'À·0

hÿ¨ºÛÁrÒ[Ø••¯~Yu\•
2E4OÇ-ý¡‰t†b!v:m=ZôÃÜ‡ÑØ8*'2µ™U™Bp[T-\æœqŸ$»«ÇZCô ð]4Ogh=ÚIP7‹1Šûá?F:Ôk‡|EÜã•4¯nlò•ºÆ¶wl
ê~ºÉÇ²=Ø»ûÎ´âÑYcûåøFQ¡,²ìk¯Ð•2GÉ__÷»vUSßûX»ª"ãØ})Hª
Ÿ¡*·ªd'ëSáBÃÉƒ¯Æ^ØµeÀÏ¥ÌÃ½|@à_*·Ù{ÞaÉÉ7'RGEØHkrô
¯™Ù3M7]RÐ•É•ıaœ¤ÃÐ4|EMÃúÿ1{-Ÿ¢Ì÷ÒwÐâÃAMº-kã'ÉÖ¨x"I¶£ØPP1.'m6÷;ÿOQA€-Êø~Ã]
·1-1½ôÅ2sôÑEIÉ{32,Sêg°úTÔyÒÖÏ3Ô™•)]ÇUûÖ~Ï'Ì•ºQb*ÿ¶ÉD?=é6hŸÜHÉ1ez(ƒ`ÐøWøÆZlŠ%]Ìç:³²ÔçÃ°ã]ÜUg¸¨:¼Ep¨^NøV•tWBtôâDJ£BÑÖ-ŽÒ»¼Ð-
(¯]nÉ¯~ìtÖ¨ÈKyÆ¥øÇLÃÐ µ€·ghqÐ ŽÃ|èÖÍSÌ'
ŠQ•_ÃÂ†ò¹Ù' Ù«wtÁW¨âV•>Žt)«Õ   =]±

¶1D©×¾µµAGI÷»(MJR'ÿzy¤U28ÃÐJ®B÷2ð(ðHÐ\GÞ£QE¤%§€

jdâ ▪ÇcO·¼U›Z41%-I1Šz0ÿWà•¯|^~%Z¶'‘{UY;Aèu▫!PO÷Y\p
²▪_¯Ø1G▫¡Ê»wÕ-¿SQ¥Õ8•Ù«ny™Ì»«{qp|Lx2.f▪dIvÕ8n½Ž=?üf¨¼▪ßÕ¯ZÕ▪«±p¿▪{ù▫k?A Ç™1âÇ?6ƒ³(Ý)\bUÊnüõ¨:▫ Q▪Lr▪(£T811™Q0az1MâUÜÜr-¼/¯¯ÆZù)=,cWÙ\SÉqÉkfT %£FE5¼¨▪Q&ÒM3TK0wQ±eöS▪_KvW%€

>K'Ò•ìj¿
«ìÃÒ·ſ×ì¨ßqìÜTüßìu" 6¤Cj¤Zm:•Ùſ
;-Ð-FÉ!ôÐØMbCÃ»~Fî'ìår¥ÃZT2IhÖÖ¤ı·£¯
vò'€€ßJ1,Žæ/×¡WIÖK&Ê¯¼3¼ÖµŒ-QÉ'¼PG3uÚ·Èh°ßÐ¤>ÐCw+÷¿Þ-¿,ÇÖÒ•·ÿZ~\Ì¤±ûÙkù·³ çNŸÝRÖ&Ê¯Ş÷ójZ¼Ûv¨ÜŒRÃ»ÃÀ¨Z¼gT•°(?Rk¤QQ¬n!™ªX.]²»ŸźÃŠ•-…ìßlŽb·R¢âÉCã-kì-ìçÍ("°°£Ax2~µ·ìj°ÃøÕª1Ø-TÜrÃÖM™ƒÕç<A9tÁd)Ìay

IXš…Hcz-©OPSâ$%P«E1U2l+âJ;ØL¹\Ē¡pGq$N-'4GP1"4OYƒ%ÃÐ4:1•atEŒŠb•'¡{-ì~VjvEO•IZÕ]©^«ŵøW(0yçlMòûJYÞ
t™•·1ØñIC´ÁJ°š¡$ÃTÐ1éG¡ØWÞ_âÉü>•…â«tâ°k´hCNÐ¡Ê3¯Ea´t"ZØ%ÅÉ!DÐnÉ"BØ.ü ™+-Ê~ÎÐò´$ÉÞ44q!bHO$‰eZã"j¯A¶|™0š'A6Ã®m7Fu"ÉÕ
Dvq=" ˆI'Ð")•Ł%Lžš•/zX'ÉQz2Ð5ÒÜ}\)MU/ÐÛâ{{ù-a!fó/âáü¯Ìâ©•ŷz0+FjHQQ%*Þ

0×Ôã∫ôÉ'%°±ërUeyo"~%*%i>%iIUv6aü„'Aã$UoKÖVrxTþ"mÖZ…ÿFIò1%?ôofÚi•¶kAÐ{à
OñÐKqjk}×Ö"f¦âüÉ ®ôÉ÷Þ;€DT÷üa;Ò| gÐÝ=
Þ«K""%ÆÐ14ÖYÐÖ-ê·×ÆÖ°š'Ú14Ìvk÷ðÐjõÜÍÆvg³Oh·[ÖÍvCwßp¶ØíFÿ·®;µÖfÉ¹ÚÜ,-Ü·ízs²±µÖhu¶·7·›¿7/Ö}0»,¨ânoØDR;ØB2]gZÍ{FÐ×ÔÆ¨-Šnõ1z

```
"&±®‰[¼·GS14B>UwÆÎ1%»ëhÿ€QDE–Ç2¯WuOG-,YTCö±©hÉ_¤Ï<ÎGBîâ
q\ò^âcÃ…•Ç•™bðëðPWÚÇ'ñ ï|®ðQ'
A
(âf,OØÎ_øÑ§™DÉX ÷Yd‹ÉZölab+V ±]\Q4\²ÉZG~òÚ¯ÃÐ
ß<©bzV´Ö›ù=¯ÉDg¨ÙWÃ¯ÐùC!r'{ß"•Aû²ÃÒâµàAÃæ"…„¸p-Þ}ÆÎÑxÕÐ¯6L×1T¯%SXÅ{@ø0ÉàÒc¨2‰›}Tm:âÍŸy>Ohz,fwÃ2%V9Ý‹¿wÿþlµ,?Ò%*É>‰ƒ¯,47Aëy'ò÷Ãî*¸Éâ_}ɔ†lóÉ÷ž;øØÖàó(Ì>»ã¨Ý|É\uÕ™ÒØÇücžÉ0OA
```

†bây°PU|$88pé•n€²Y|,b=Ù]e›OCý•ãÎùm,I¦GÎq¨NS>1•¯ƒI7ī¡Wf)|NØ1~ó)•
qj"I7ðwÉ.„/™¦¼ñ1ôñàâÌøÚg¨ñ€§8¼¤`ÍÎ»~Ô\«›OA,/›E7¼i}„yhÄ1I¬CV¼ª_£ÇÍŠÙ›ÐŠ\FÚŠ¥Øm—'W''LÞ7kÇÆ¨é÷(Ã°ßþ-C€g

```
âçÆp?%§A0ØIÄÐþFC'I•#I"ÐKE>T•_yê-4¤÷dſ´E'AÿZ3FÑ>. O;-¨|Ofr©Ø[i
-¨ßwûwâüßã'ç~÷¤t·¬âéÿVö¢lùù(bF¿3¤q8Šã•wJ|ñæçQ¢Ñpd¬¤•9Ò‹ɾ?à'2|%Q¯Ãd&Üîi—Ãã ?Ã∂ZAb„´$co•°jJ_:'H'°ÍßÊÜÛ)OıCãÕy¥2ê¢ìlãÕ‹C'Oe0Ñê‹•Pm„Iq¡nt
```

–F0,$da€¨âA<ãjÃ€ÒUPÊ"k^0›I D¶ã}„"áEÿ0ãa•D‰¿IÏ«f¬ãiyÀAIãøw´E-YÿƒÆ§IZ•ÕÃ¤wùõlôõ›ÉW¨ß|sÖmÒÕî~âÀÞT3}L
üÉXùÑ!mnm›¡_Rpx®øàøÑŠ 6÷ô¤ guR "ÕaF›x€•MŸlÐÿ¹ŠbÕ NûŽãØÛ*Š'Z.ÇiuQ2«§°¨

```
Δ[()Ö"xøhOIHÙ#^çñE0÷ò¯×4•%2)F$•ØIJ¥%oK=Ø0;<Pð'ñ°Ÿü3ù
-□Ø
oBU,ò¡Á›\È-(ßQ¬ôòÅ      ^›Ê÷ÕìÐ×Pº¦%4Ã  GüßüØ—Aé-/Á¥?~:)·9]C—Çq÷fØñ
,° kwvÜ~Mr@.ðþáÅñ{~ír·ª¤ÍPþa ø\ÐH1ô¨•bã•Ø9ºµiÁ6ŒJtžÉ•Sß
```

^A¡2i'%7IA^7Z_>ì¨kTDGW&8$EW'O¨DA„ìkIB-ÕEU×U?ÒDBpm-¤2Sau¾"fÉA×à-hÒ×C-}lII3ua¡sÞOUU0'f›¤¨$³F™fúA¼•pKmFaa1ðɪ!t'RÇ'ª4)‹-Ajf-°kBs8xÃ£¨1ó'¾U¾—_ªqW,Ÿ„AP³¹

AZ0z;0aIY6ê†cy\o0¶i%ò15'ÆS¿8<{%\1÷ 0L@lR6Ò°±Aq°C-B¦v•áUb87ñ'Œ'÷I°Q>Z-IA •E{CÆWdu%-ZD§¨€U$ ID=
¨SÁV'w:NI°[°°®÷$El]HJ¿R÷ƒâ(Ð1H°],÷°E÷°sY£~" ¨Ò8÷o'~Š¡b‡-Ia¤A¶7b°t€ôÅ¦¨¡ÝÀÃ&

U‡|ÿ}AG-FB%,1ó¹HHSeøÐ%àkJ2 7HAQ,ÐÉ¯K~Ì4\€ðS,ó&pÇ¼÷ß¨piCpÒ€ò,¯ÉUæ¶L¯tq}>,°Z¯¸,üó1<ß¢IiU2N°vÁ&AZ%A',s÷LxzÊ°o€a÷•¶¯Ia&;iøqgß€bŸ°Ţ……ñ/'ÒD÷ ¢UKRy|¥EµAøI;e÷A+ßIÙIdZø%7

;#1Sx«
á¿FÜGpÁ!yqRØ11I#Š´/ PSÀ)à·YEÉG÷5ãI•')É~a?þ4ý
pÖ3gpÁYÜŠ_¥ãÐIÿ'lcXX÷^0Ÿy9÷ÐU÷bKWÐBbwjàU1Ãðÿ-gøÜÇã-¹¿¾¿z?rÉ6ÖÉÐI-ØyÁ÷æK?Ãk¯"•{;³Y¢®I•°•)uŒCNqÐ„(ci6¯23bVGÉc¹¤|-C5Æ™:÷0y.0-µ
ª™lnÉIã÷øÃµã÷ÃãÉ'KIüiÃ_Uo·)Hŵ¯Ú%2
$4ÃSyktlMÉ8x43•Zza3-ßã=|c'†°'TOD_úA"_I•$\;•I•;1·Ê8¾¡
'tXÐBÉ|c x^yÿ'"¤#æw-'ÃUCÎ6t*_¨Ð0TN,²ÃŸ¯CMHw®ã,tøDw
úÃÎÑžÉõl...1'ÃV°ºËÎist,rÉÉ
$Ød)¯ä,×¤¶Ð^-3Ã^€ƒâ1Ã±\2SrŸ9¿GŁ¥ ‹y d^h(E4 Lŧ

^ Ea¹B.¯vN ±±E±°¡¼E¢¶1°þ(£^

ZôfâªjcÜ‰<ôÁ‡ª1ßA>®Ñø0ÉÁ¥Ë¨gÜlSy••ãw¯{ã##¦+Ò•IOÁ¨ò==IøAA
'XŠ_âdb!¤—YRŸÉÊ?'ìÿ?ÚÒe Áxq 4ñ>C3Ò-ÒÐQÔXR°1\ÔæCã3ìÎ(ê–ªÍzìL°ê²Ír1 +ò.°ðúÎþÁò:mfÆ.•p(-Ló]tcÆÜÃJ
.ÌM¯Ò•Ã»ÊÎÇZ3Ð'ƒPWâU
ì¯HÜ¡ðÕ}Á´•ªrcaòèSp–[²»aB–SZ²ÕŠ&NÑ¥;TKJêj*ï³k1(\Ù»×ÿ^Ò‰ªE´ZG'¾lÉq~,ÒŠ´²ÀNpêdØ¸ÒmLfBÔþ'®ª1Ü÷âÀjj.lôeSSßÜŠ‹tD¿âP}D&Z]C³H" .DSuÔÀ‰_¶
©Ëì¤pÒúÍ ‹n\¬
Nn•v÷©~¸sZSb¨Ì|¨ëBgQ&±S'%,'jÀ.Â²nŒ_ØÁëvÞ¤;Ç)ÌÞÒsYÜ,9:aÀ*_‰$,ê]€@÷•ÇÊK Êx]„ºuj¨LŠ¡±sÜÐ1àŽ½Ù`ò•

.vE]âÇm²Am·A,A– AÐhÃÞ¤Ï'Ä³Eß(šß°•»5Ò»ì-8Np?X¤ñk?YEAìLüIé»h²#ìbÃN§³@ô4¨fbÒ¼ãdfÏð™U&žöE+xÐ-]EArv›-™uAX¢k_N°•%(AÃ°ÒGA®Ł••ÎþD„„Ï¢Ï•à {.B-p»2›³!¬¯Ø_AV0

ÐwE¡$,ôâ€iA1÷ù
IA¿6P%/,¯I3`yB{ÛŒ¯p×¥_¥kyLG‰ôOAI{d0¯¤,®C1aB4Pfi\0°1_)›•9./•Ž<¥0Dµ5µþ=ÆDU{–ð Ã¥9/{~¢X

;Dûm•Oby¨Uïo•'%YALE°à>¡,ü•PÍQuC&4±sv1Ûfjw"}SNY"EW•ä»ÔNÓZA•UL
sRÒ2•Ô|ê%kúv2·Ú7É·vhe2¿Ý}ÃjÒ2·»·úâ¿Ú¿Ît•ê%-²Lø{»,p1ãß3H4´A²`»¿ì
ø¥·wÃïÔNdZh'® By1ba°à-ê•°„rTH"ÕÃ8-Ã2¶ETÖ•ëÒo~"Ù~²¡k/yA{ê°ø!|hvx1:Þ¨U%%

?ðWIj/•S)1"@•IO'ÝŸ52E®KS2''SN5ŒEY•iO}B=EiOAT•Ó§0,åæÌ¨JÑÃLì–µ£ðRŒIAZQt9UNt3r,²„ZþÒd!rZ6BÛO#ê×»'–¢

e/^I¯»Z¯EZUK7Gw•_¯³d.^USnê.gÖ;á#—$1óÖ·UBFYFYZiwEd ^•CEU§BOI2>U8¸(°#SNwafÖDH~^o)©,ÇST0%¯±¦R·E¡¸[©{¦OWÅ[CB°>Z_1›OS∂ƒ¯ff°®°@¢¦N

lOj»ëdÇ[u÷ïaN'ÓÏ_2ïô[=U;¨VFw÷hŠR÷þ0.÷½(~YÀ½÷^ø÷v·U÷¿WÆiüïŠø'""¾Rdv^g
lcÃ}C„ø‰¬ ßÃT¾Ã¹Cg=!‰ßaå•éÏï/JM¿–!•ŠÅ_àò''hñ÷.ßüï*Sß)Ý¯UEåWU¹•Þ<ï!u|+pÝjx:v,ê)ÅRtÁ/"c(÷ÐDÕ_ûzøŠØ!{}+±ŠÝÉ,..-Yàßmïx¯„

2²IK1YE,U#9QÆ»ãE†°ô^
ûØ´2²èÆ&ä—<^¿ßõ¯2—Ã Ç«ëp_U~wÕ=<<<-NŸ±ÃŸ®S&«ŒKLùó(°<}µ'ó÷IÊ¦»Ú ¯¹~°ô}tÕ¡®D4WK-¿fA{_PIÃø
øu°525;I1/`8÷Ö-ôãØÕ}Š`|™ŒŒÆ>-¯`Icw×YRÐ×Æ¼ÃÆ.}Z-Vo4ÆiÕ[¿:TlZP€x8-!°Ãã{ÛÙ~7}-»¯ Ì]}†ó4)ñ_¼Þ`Pnzhb;ú€¾_
Ø÷ZC4P†'|ôIÎßÃBBY
¥Ä¿ê
§ì´€½»[¯^+F2°rò°Ô' (-âßßÕQqqÕß;Œ¶ì»ø3¯²##RZÖß1`ÿ-S›®§ÂFcK°5Õæß-*'ü
ãO¥°"¿ù>3Ç‰JÐ*•°Ê-ãÐ¥x¥Ì-Ù-ÇçÐ3v¿±Ê-•ØVCŽã, ª/ê]Ø`ÒÎÎ°k¥¶Ê6·ÕQ)•uÐyX|·—ßÉß–°ª¤-ôtïÓ÷IÉÅà$¯÷Þ…¾÷P`2NH°Ù@à8½•®*3`ÂM9U¼„•(3Sh©Çal³•¡j¯Ã•æó·zÜ_GY®ÎßAPz'ë¥lPÜâ±n¶]ì¯Ã¯c|-Þ†Þ]Ô-ø0,a«@|-Þ1H•11\(÷+#°IÃÍ›7»Ð™%ÃÃçÊgM§©—&ßEzI=F–Ó¯RJn1åY1á¥×»ñ

V4u¶°6Ø^ØOwå<"Z_
©2ŒûU°¼÷ÛÀ'bÊ¾DØ8PÔi·'o7kÑtŸÆÉ^÷o"ŸÇ`Dle…5ÿÛñ›in-Gr°†!Ü10¥Ê6i›úZ',ØæUæp$]tlc•SëÔ^fÍçÉ'ãûæ®q•Ålói÷©Nä¡M?°"U²ÊV¤,\¯ã…!ê°_ÖÉ}âCR3Õ¨¦ã,÷Øgzœ£=nÍÃ×£•Bw÷5Ñ¥ú{6%:?ÍÐ0=?ð÷tÎJ¡¡NúÁ•R×-M7÷¡Ÿ\\ü¡ëwœÐ~¯â\wí{6‡°ºàG²zsâêÅ,ŷkíÛ°ÙH'ußúç%)ÛfÀ°#nj,7l6Tê0Bindw-¤°ÑÍ°

L·i_·•A~Uk•~ü
04°¾¨Ê~°I qoÍ·ÃÒ8¡°3ôM"'''_t]µü···»®PxBBlÖöÊ=„m91»®TçÖ¥æÉ·^ôâ¿‹¤XSRŠA¿'ÝàaÒ•?Á-zqÜRnx2Ý¶Im®Öà0¥Tÿj¨?qÉfßß¨y3•VQøS_2/o[Ü°Í-ê¶.-Ê•,ŽÉA´É!„¯ãbvç÷Ý·>à,\©r°× Wüþô¶-¾Þtïàðšü.`MÔW5×f~Th^ÿ¨PVD.%!Š-z•n] ‹_¿¹mã?vÿx•ýžÛR_1'»#ñàtÖï¨ÓN\°,ŽFâ¼Ë Wüïûï!HðÿWNLÃGâûÉ•bkhθ-

S¹nδEA¯IG-8Ol¹•-;t¨ÖδIAAδ¡Z•L}2l0 Yb0ByXV¶δ•E%v6¡•Oδ=

```
ón¹¬›nßâ€ú°?4 Ãiü[A¤hSUfYÐ-jTÖ·ñ•nÙ¯?¯.³.=5ã/ ¯þÿ ðñ(h+›cN_Q²ü
ª¡G³,CÁâ¦Gkz©¯ñÒâ€™•kÆÕ¯ÃC{§¤*
      æ²£ª·Ê  ÉÐæ¼ë
F¥VMgÐ]ÔuÐ•!Rs}uQÃ$¤Oø/Ã-Ôâ[z÷=ÿ?$ßÐ2J<agÝ
{Ñ(âÕ£ùúÜrÛ2!9¹'™Ló.□ ƒ³?w¥±Ô8ã»$ZNßÃ|ZÐ-gÏ.•¯Ô>›âVÐ¤xÍ÷~ZÐ¡é³ÿ¯Ý¤•N¯}Ã¯w0ÐbÛ½«gÕ¯ëÝ•Þ¢0-ÑÛ÷æ±sÜ;Ê?ZÐÇ›\}ßãÀ~•f
Ã¯•Š•\ŠÃÿà'¶wP÷èÊ™Xx°ï²Y«Œ¦ÉqqÁZÃ¯/«nNq$\r
¯æÍ²¹\Ý.ÎñµøÃÚÆ|³Ñã-•27à©Øn³n3÷ÐvÑ¡¶QÐÒbß,Ð¡ÃIÃÆƒ,ïÝÐƒ›)1ß§:¾æ/ÙÐÍiÚÉÜÈh4Ie1iÃÐƒíƒ·-ã,Ãf¯jêÐJÃÐZ\¯i2¦
iùL—Ú°eÑh'c°f?Ù'Ão¯·
ÆQjplÍkvNLü3«tbÃ|ßßR²Z,Ð•7ÐpF3ßs°ãaØ+Mœ3=±vzÁŸ¦Uo:[U,-ÉES.Vü¿xBw¶%v§    ûæÃÞIƒ_Íƒ_¡
```

2dqiU4WUi0uO1i2¯O&6SIW[€]IOAF4]aTlE1ç•Y^•EGFBA¨-Oo22b•Az°!8ø¬-æ¶I|pEùZw¤æ•1°uA•°1ð¸$J87ysq•¥B'E®ù^YW•‡ƒ'ÍIIØðø,G·Ä

JOAYBFâNXWyrÿhédI–#}>GâZØ/ßBgâwerÇæ–NÙÿ?ÖB»yÝ»y•NÑ°¿ÒBSwÉ¶Zis7·ißøöâ·Ö"–#»IIAEBûS¿Ac¯š^ßl›IpÑ5'†^•¸'kI¼Lg²ñBûOo4ir0÷pM°i¶LAz¼ß¦8§•Ù*•‰-'}IûâEVà³8™Ñ»ª¯Çô.Eë|
Á?ôxÑÐ{(BÍÖ"¸Ï}$Sœ¶7ìDÁ°Ü÷SßÑÀøÿBZÜNi>ðù]HK°ÓNEH4·â\]Âåìi2fIqxy@f~âÖG'L›-ÎêÙÐý•€ï[ï+8¼¨Pj…âhÍ'¼·w²zâlìÿ

‹A21»®%»ÃÒ¯-s'A
üØ6KtZ´ê)•ãëâÿë'qÐçfÐ'gëaE•_,Aê³.•e~Šæ\'IÃX7mŽþ»bŠ"ÙK..•9ëN:XRØÿøPWšGF19ŠŽE0†÷»'
¤ØMÐ®[íÃiÒ3ê_V°š,¡"•zÍ|mÍÁÇ½ ÐÖÒ²•Ä¿ïŽw'C`ßÃ•©_}•3g C•ü›qWÒ›÷ ì»ÿ¾3q¯ÃÒ_O3°Ôm¤:@ÐøüMŁø:Œ
K«Ü•Þ,ãŠLbüÃxCQ•hãñŸ™™CÉ®ãaoÍ-'ãšÃ®bSÍDÇ\'ºòÃ'/•Yq‹¾Æ?_`Sƒšæ[`IøÁ\•E•ê"ÐŁã>a»RÉ••ÚÆç'¾ÙŠ•µø';`šµÓMãÉm
ârSHR¨2"‡|!Ç´(ÜiùÀ@•ÆâÃ¯•Ã³4øYÃ2Ò;YT.Þ6vëN·1fŠíL2)ÉP³·YyŒ¥|3D•ëš|ór5"çbâh‰˛™
ìÃ•ãÉ@ã|ã•¦fÿF8S°ÔÖ3•ØØ®mIÞES}Ázš
!Vk<ì_ìøÿh"ùßù÷ªãîµÆéuœæs ‹ÍˆñÑÿ¯WãÉ¼üãÕödVQã3<ãì-•¾pŒ
»5Ã'@Ò®g|WãÏ''Ý¥ê½^/nZ¨þ8 …Ø.°4ãR¨M¯¾ã
Aø°ò•Z4Ü¯_k4ãÏ·ßÉ¶Ñ1h•ãØÉR_¯<‰/Xà YOÒÒ]MþÝ¥•Łzgm/IRùHÒa³&ÃãKôy-Rn1Aü~Mÿ)Ma°ÚÉ¯`lK8¨mßùÉ•Ãkq

zÃGa¨ù³gEK2wBŒX¯JSU?„GãF¤Ç!»÷6Ð÷1‡;IS‹•æ59S)Ÿ¨p.ãimã™¤ùÝÒ8±Ê5ãéþ³¹UA/4Ex.ôqÐ4^œ6&Çþ;ãxl7.1ÃBìì=æjîÒ¨{-v<A÷A
wôìäQ5ê³Ò•Ì ï÷®ŠÜ¡þ\›àvS[ìÐ¨¯þÎÜî1™^~ëã\Ðî'GÉBüŠ¸ì÷¯}¸á⁄³à•Ê*¸Ø%ò)Þ SØ~GäþÒÕÐÐÐÕ&÷‡^Ò:x÷^::Ì⁄äG¯?pÒÉ÷Ð9‹±¤©ÿ~KgBÃ7'ÕÑ³š)séç¼ÿ•É°¡ðéÃ¥•Ðlš÷Ìn¥ÐžŠÉžäkFÃÎ⁄
l³Ø•M;oà÷¯7⁄·¤ã⁄•ÊTXœþ,çÀ@ÐÁÉù²Sú Ôš6·Ù„Y®ÿãRIC-ìÊTç»ÑñRÒÑŒØ.XjäÃÕ,ZLCÃ

ö°O-×fFA'"`AzsAGç%°ì•¡``wSUFivNyD*4'M•£-ð'z¶1¾}ÒcVIqfF·Z^ò;7|€1â¿Ñù•ÐìéÐæXY3}ÒMosâÒh7 yÉ%bûÉ'-yß¯Çž2•ß™ `•…ì—,ÐOYfÛ¥ØM'ì»O7´CAV×EuLÐæÞ1ß†¹·ø2¼µ^A-°›3^À_¬'y•-¥C7_

```
¨1Õ¨z"<U#J^bVPS-Jv{-|-I&m%?IUÇ9.Z[/þâß§"ðÒ~³pÇ~œ¶I'UÐ$#
³gÃÜ¯«x»}ç"ÐÃ¼'PcH%J•MŸ¨Ã-LÅ;1ôM/¦ØÕöß¯ÎÌ;³Ç[.5Ã.¨ÛmÁ‹Ú;
xxÍ&Ó°©ZŠÇfÃãÙ'Ç        úà~-ßÁ(ùiÀ`ÓÍ2¼Ùæ□_¥Ã'jó¥ãð
ê¬Ï´ô›Nò-ßø°mK÷Ì®*•ZÜúûn|·4ßíô…ëL¡VÐVÆ›
ØûMVÀQÒ=ſ.fâVÆl.{_Q|mÒÐ÷Ûuçð-.zcÉÑ¶ã2ìßíôfÛ]ø=ãß\âøeÓ^M7-[¨ú°Ÿb…ÀlâŸFHpkê÷_¤pÛNl9/¨À-,¥$ùT"…ß<S,$snÃÉA76]]‰-1-•ÒHvL•.+Yl,ÖßbÉ~qÀ!Ü¯1ðm¡r¨â\Ø6ø(£".
ſV¦-g-°ø_S!G
êdëÎÅ_ÌØ-ŒÊ¯ÈÛ=$‰ï'ðdW<l2ñ"''ÆVÐ+!°93c‰°ÐÉÁLÌÉ Sí÷F_q°¤Xa"ADghk22°ąÉ',8DŸ-Gé¶ÁÅÅ-(ð¯ze•¦ÒqÉv-©9-ÎYk©IŸÐKNſØd-æiY+;÷°v"¯YŷNsÕÃ¸xÞÍÃÞø¤uÌÞ°Ù•a ¯BY®P,•I(Øøç¤a'kß•Ô4¶••ëÙ•\×ÃÒøçNÜÒ¯'cçð¯¬•fé«
…8,Ÿ
ã`Ùß&^ôÿ7%Š¯f&ŁôÀ•ÃÛ°QS…ß
gêÕ°ÏO3²Ù°ÐßÇÒÜ3ÃŁ×ÃgÐÒr•Ã·ß'-¤DÉßÃ¸Ÿ¥ÑÃÜ¯gŁTø°ñ•Ãày=4V1±MŸ,r.ÒuÉ÷'••Ò•?¨¿ÿÉÒŸ1ÿ6c~ÜÿS'1
ÿ.WÜXÿ8³ÿq_qyyMø¯ÃŸÐ3ùÇ°~°Øõ¿|-¼ì·ÎÇÍ÷•-ø8•ê\Ò¤Ð?ÐæÌs-sR÷&r.•¥ø5ÎÕ°\ì•¿Ã\Ò÷ß/-ØR÷°ã•¿Ã\Ò÷ß/-ØR÷°ã•¿Ã\Ò÷ß/-ØR÷°ã
```

)WBEU81%Q;%}0aO¨QOQ{ST3OE=*9Cg>Sé8%Z¨·Op+âÖ?~ÐwIo›fo¨*+hoO¸YFÖI÷…9Uâø1ÉtIıûlUfBã}vâ^B(A6§

=V0,N<-X+BO^/¥1¨I>;IUØ¶uE ,=93¥œEµ&e…°nYÖ~ð1"vãýSyK[sE²°~©Y|7Y¡-tHI.p-n1Ö-{ðµ,N°%K/1,N
@ÖZH[[ÉÁý»)™²ÂvIQ-2bÆY$/I$Þ]jñuÁQ©wm|Ž!ŽikÅ¼RSb°¯¤E<²§2ã($¯®Ł˜¯|K¢÷ÐÍŽ£l'ƒ™ÁÄ¼ù÷²t?<Ù-Áyŧ÷²ÃàY,\ uãÕ£ŧKbÕ$1»Æ
â'hZ#¯ %€.|pvkÎ
gâÃwŸQ0bm÷I¶¢¶V÷bŧ_¿Ł-ÐŠ5o6à:÷à(²X{Ö,k2ÉŸXPTm*
Zã!ŠÅ'ŸIhwÅ-µâà®0
C²?+ã÷¯·?bb™ÃŽ%5Õ }¯qæy1ãI·Ã¯ñOyÍÁF,•_?÷=OqGÇãÍø3fÜ¡±Á
5ì¥Ÿ–#w/k4Ã$Dò4

ÿç_iØ;fÍEã¤a2^c÷lì\·Ö"ã<

SIOB•9
¦_`?4Ö‡.%izPyTs6GÐEe.•|•#]2ý¡,%@mÉ•°'[b/
¬Gz,Ñ%Ø°8`1

{nAAtBVUu]+IDL1UFGUM$Z9}GGb•$OR'»]DC9TOE·/AYx¢ƒO¨83A8x¡I¨U€"•H16Z€¨UN'N¨IO¤•´Af¦mAs¨O>?SUiaBÅd}••AkIa…'zIu•b6ŒI{lu•³ã•

ø¯•–‡•Ø,••ƒó'•ıɛ0¯8@ßGßƒ•nC)O«_%67/€x8U/o^_–nU
ßŠÀUR_2¯K$´/%ŠŠEgßs?ð,½iÖ&Ew€Ü_l!_Ès••ÛH5'ÑñIZ·ÖdŽH15•É‡%Ç[ÖADß$k¯ÂfЋ›l[Kh÷é;••ÛþöÖGøÀÎÐ?®¼¿âÖ1½:9Ü:×BØzsŒ-[
ÎB\'à
†®®ÿ¥jâ
Ä~qàÖ!§n¼"¦ÎO÷ßñ¼¨P…?ól¶2?çèr¶¼¤Fg;€÷ƒ*Hzi¦ơS!%æbûâk*-Ù°V•ÉE÷jŒ¿ÕjÖb¯¯¼]¯4%±$de2âif÷Ù›âZq"ax1Ç/ém'â¡³%øÐ33³=ØXÑ÷ó)?¢•®ÿ'jYJÏÖ-Šú€®0 I[X'X

¶A#|×;Y;O¨[f]¨%UJÇ²kZ_ïê^O:ãKSO[ëÍjlIR2b›Y?Ù'N«x"Y%ª—jbÍ)5,êÐBÍëÐê••2†G-éæJ-'Sç/_Tm-÷ÿ'aötÐZ.ãåW:YHñ~ûx ç$6-D{AN<<[íü (%ñ‡2Pë|hãóRULKU5Z;"ÉØ¨·+d°©éjLcY€E'@°rĝ÷%Ò•I¨0«0%‹@u5DI%CF$V¨CÀ¨A¨ãh8Ue*éAã²b}Aq@RüW•ðš³ çz€‹
®Ò6üã-g49rÎL‡"Ÿ%ÆbÖÐnë/%unÃ•JaæÒw÷ÊÊ]ŽÉÝÞ¿ógæNÒ|¹ú%MRÖ•Lïê∆ãK¨vÃ¨O„*¶!^vG1¨[}•|¦:o9a·•¨Ä¸™°9Lv÷SÉß{°Ùéwš¿j°ã¹…m¨ÆÆ$ïÆ∆ÐJØ$r+1%‹™%ZÿˆÐ-|N÷ñ›1Çà7¿÷¹ù«lãéŸ¨™%7ÉüCÞ©>é~Ù#ê€"Ž}ôNŁ…¤OÜÜ
/©´ß'šÃã7÷yæÜÎþñãzøµ'(¨¶ª§gb

```
EEæëÿÉ=OZE21÷{:ã
¯_¾
+-8:Ò°¾)Ui&÷¶zyÒã¥¶ZÒ¹·HD6Ü,ÇÁ{=8÷ânÉ¼q¯Ã&±¾ò*ø¶úÁÞ¯»+¯÷@aq:ÎÖÑ;=ÿü•AoÕûlr'Î¼T±•AIOBãÇèsëèvSÿcSúŸr'~*ÃÖ³®¹,V,&)M
ßŸÒÃ—Ÿm¶Ü¯[ã)|Ê»_ûÎ~ò^\ÁçÁc2âãÃ·FþS.úCÜƒnŽÜZ÷à w'c"÷vr<$¹ãZnr<ql T*ãøê€\"
¥ÁÃãhÀ7ºñ|VNrÒDjúÊ¥Ê¤»|÷0_Ilê9A^Z±  þ- Ç;ã'ól÷¶[æƒ3Ê¢] þ|ßÒŸ_tÒ?÷ãÀø0
```

z1i2]TfjUOW}eDeã^AMU€[¤U'jAü"Z•,•»'èðB]þAw³KsE0M¥"¥F²ß(¸%"+2YY2*c?UgD•ãñ`yñ0AE•#3)!q÷¯±æ'fxó´3©1¼Ü2µ

¥# DzgÇZIÃ}¼Ø•sÜ)„hÏ=™%>ðáƒK*Y§oÜ^ó;1{EB7F|b(_÷¾¨¦†ðk}›¨YÀzüC:?Š,°¬^ºÿ~…ñI

7qA•])·ko•E•Ä66ö^Ä55Ωµš{~`V•$7¨,•{I°ÃÑ¼ÜÚÜUC-5›gSIR;µS"/cE¶u"çÀBI{f[{KlTBZ°!†"m•ß-ûA•ëwIY¼BADfqiAæaaÅ>3jÿ¥ÿvv)=Ä…±†A410k.
b3n~ûoWZŸü‡¼.lŠ¦®8mò·™Ñpbuvê5.ÖßñªUØ^°sehrÿ6!ÖÖ\°CfMOg:t®Dø•ªa{ıaÄN«ß""UÍØ'nÅÍ=™-•/$FÎÉÅaq{cå¼ä%tčçbØÊyuvŒF°©VHÜqH.ÿ!Ü¨Ä·Ín•ÄŠ=ÛþÉÄ•v
{®°¶~-2ŽÍÿÖÍÖOo•FŠPŠÅM%jK.pÉ:Oo•Œ‡_»¥P.Y6OVØEynjr¥rHuçBìà.†<çi¶ÕÑ&ÚMç›Ò¿J†;ØnÅ°•P´AdHFÉ&»øQ8nÖU|•¼•Ål=»a¶

DZãg°øIó!"!ð;dz
"CÅÅ™»u¤"(|ÎÅkžÅ3¹_Z-¹iÀ‡.·•-5fSÎ•jßBBÎW¼)ÊlÎ·›³Ÿ?
F4§Ûî‰-^³:•ÐäÜüz=ÿ`æŒC-•{ì ·OŸÐ`•y·ÎŠß_¾¿Ã£},IC²ÎlßÛf3×¨µüÀ8gT4'"ô~T9Ó2,ØP•zß&wuT·'D"A\€]T6•)ª

6P>|6`F,l.•1#(•c^Ap

‹ZAΓ¨Γ¯–÷ΘEb-[WEZ°~£o¶¯¾÷1}1:A

¾-bâŒ=ØNÐ<"W‰ONÖ°,CUN•'N P|'ØÐTIOßGtÉ'U5âÊ²ï'#ª©Oìî¯3ùu‡
P»óÐ)¼4Œ¤nSâ²ÎSZ;±EÍPªÒ»ÎOÍÜx_>ŒÉ
¾Ð°ÎÞ@ÑÙ'âŽŸÍñÖ g_°FŒR-ÐÎ•gÕñx'-ÎâÒ°sÎtüBQ·âÜ•ír¨`láiÜA¤§M¾'ª³™®rïAÉ¾ÙAn¨ø27•P-•¨ãâÃÜûì';"¶

°Us÷C×NŌ!ð÷½ {ÿ ¯▪ø©Ω÷×6XÐP,ìA•Y<U¯BNwÜMn÷AJ)yOÉIò_ì¡}Eø"F70‰ÿ^•Aæ¤I÷}àRþN°çA{QY${¯S"5¿âI+ys„Oh4ÿOE™Ð¥▪=ñD>ø©EYAf>¦OU¯/ø°øµñ'E÷E▪5ñQbn'×¡v¹?G-\R·»#¹…¥EâN\Oddç?Ò{uJ▪™ÿ÷™±ï¨i¾¨{A^|®!-#F·l>Aã!▪[¢°aZ¥!A²$±ãaÐÃ0}fßSM1ûTAlSRk;ª

[©»gEÃu°®ÿ¼uâãGW0{óùsI<©OI°•phz
‰;#7mÃ0M0vã*xb»fÙ¿dmO»‹Uf²Zf-1°fñœL•-Bg'óebiil¿YuÉc6JuÐ?¥l¶?°ëÉçóoèÿÙÖ§o¸uÜ°®1÷7·aJ1Yƒ<-Sf;••X&-Ûs0ëü-Ói¾w}t$'P€Š-'w¯yúÊ1C6»g3i;©A2úU",ø9F\Û)ãØ›Ä0Š»;p8ãŽ|Ö^Y\]°B1Dæ",ÕÉJ¿¶¢ã•sÃGæÿ°90°8âàN
°°[-
¥¤°Ì§ø¯ø0™]Zd»ú/K°©?ò-:üÜ•Ypo¸ù÷³1•Ô,'Ê6:lvÖ\Ãœkljs8¯NEA§É="•ÇIA3œ=ßàÀ'iÁµ'qÇø/)—{Oÿ3Aß™bò^œúÛfqŒf
6ëëc1²•€ù•Ï-•ÖÐ|#oëIœÑwÌ3Ø8,]'Ø•5Jµ'ë¨€
ì.'•ã~zŸ¢2·üÑxØ•Ô]ÉŒwVx~Ís1÷64&Ê!bÉßZ¯®Yò©³I¯¯/Žòi£xøÞÿè|Ê"ŒÍP1é‹yÁÍCyñÜëvàÉÁëO€Á•O·¬q
/Awi£D2™ÿ]ß2Ñq7âáØŽÃÃ^Uz%]-'•ŒÉ•Ÿ'Øæ|Ê±mœÞ••#ûnŽa÷ÃŸqTÅ
Øf'™Ÿ-®YOœBgk6¤²cÐ÷ãÕ™Ño·Fŧ¨ü1ÜÃßÉ487Þ·ŀ₩ŧæ¼œGÁŒÎ-Ž{%ß{¯Aœ€1ßîÜ~m,Û
£Øu@ëMœŒ²ÆÇþxÃp Ò«ëÑMëÇÉÎ'ÆßjÐyù~ç'Í{ñÜlÃÐÎÃ#ì, {ſ?-_Ðßà;·• ÜÍs'(-ðÎEK§ÓJ5…ÿ¥₩¸ ¥ê1¬yÐ}µ

jTôZmZ<:; 1&ß?¤•I Ç¡.ERV·y»:°•V°2ubžn•¥AUÇO¦MOøP-\ûQ+uR?Ž-£1/ö•cÐ¤ua^RAA/¥aZ$^g+7UT¶A•¶³
8¤ÀÀ·Ö¤£24bæ-þ ´sŸIÎ•Á&rÞÉsŽpux¤¶9Ô°C6°2 6Ö5Õ° ®sF£s¤@Ÿ´£¿Á_¼É cñ»Á¿-™¥¥Va¿Ô
zBÐ¤ Á Ëo<"-Îçur%&TMÇ_ÐñZÁ8·Ö1¤°ÎlKBF%'QKb{oeØ®ÁÿA3.µ\¥/fÉ·Ì-!T?-gW×³

X8·OGOOOz6·h¿@OVD•ñ(ü1#ãSk'Uñã'5ÿ*IA*2ã4
c7Z''4ÄYL'}æÇšZ¤S/9ſ[lÒ¤!<·(^^[E'Ï^xcø1-¿oÒÍ4ºpøa/Z_Þ_Ï|ü¿'ÁŽ·æ·9üÍÖ=8Y~®<{Ìê5%p¤±ÆE
!¢ÍÏ·¸Ù/ïê)³Á~ºÖ··(ã}ºæ²)fN®ûùx"6ŒE9=Ґ¦¿ûüã-æøü¤²•Q·Øh_yN
w@®9zÖ-Ä¡ºS³®Ê<÷ØE<÷U^Ù¦÷Ì9ó•_®»üÖMŠHç¾üÍ|·ÊF37¬yö8GNã›,ê pØŽz1ª^-5ó®Œ®Œ¶?zûw'^2j ÖM¤ãÖÉÍ‹c¼D

u^BSÆCqÑ›{«NUEH'ø$-.A-³ãw¨ù,šüâ1ç¬Ìü
6r0^¬n{9Bz^ì'
ùÀš†-·³#mSP;±¦ÕUw÷$ÕQ¬âZæ@08":ó1±A7h}¶_ÕnZr1Ž}},*I~;j?~Õ¯Þ-¼Ã`Ł[†Ù9·$OC°ÙÜ/£ŽÒöŽ1~Ù^%~-zux²{|[Ç!©à.zÑ5'~âBå'X‹ø`y4•×SamÚÃØ{],&·(lÊ{ø¥æ

```
S2âTmB×`šol„u•7~§RCEb0N8
±Ö¯PiBEaoÉñ?K«AL3M°/ï   ×·YTOY
N.c_L«F¨(•a)Y¸Ch•ãCŠ™Oÿ`9BàY.xI
LaÁryd¤ª¨¸hþ†¦¯º©~R v4•:2Ct6¢€Aw°yaqstÖÖ·Á·Ä¿ú F?ES57[tâñ'L¤Ub08
<©SP¼÷/°6ãÜbŸB4EI%@uM<·7k1:^¿™¦§o.³bYÊ¥CiÑ¿Y§R/™I•+Aâbw+l÷?>OyOâgfø/ÁiqIvê™2M,›vŒE€™"b°bÉ.0C'>r1Eİ•y•YÃß´É•>Ž»ÃÿW''UpicÁ÷%æn%ÿ1ã®Oø27°ìKÏØ@Ü5¢oó¢ÓÊÃ"z]Ûkã-†Q¨•_2UXtQEA^n¤S4•0f¤VK&0ES&I8¤bVÜ°}âU€
(¹gmùŒ¥!»•,}Y$C?(w|GDo)•<•8Z9´H³´ÎÔ»ŽÜv,Ê%°v;UŸ{ÿ?®ÔLÓÊÎ™`—4ìk¦þÏ¯%'o—•T‰Ã/0¡€SÐ!²Q|¾-ôÖmü_4GFÊà7~YÜN
4šã$•,(b4yŸ´C'\ÒÒ÷ãJ9.ÉK‰âøMÜÁ•Il_VQÆ$ÐV´Ç/ŽP$°žÆŠÃÞ÷Ð[}y´0 ñÜ}~v©IÐ‡¤´aUC€Ò™U° - ©ø]1kÒÐ}÷~Y/Þ;ðÿ'•½)yÐ:0;÷•ÐÐQkMÎ²j„g»'2¯6,Ô•ACSÀÞ:¤Ðx³ HyÈà1³µ÷Ý•¾ûÉ÷`¢TÿF0·ùnB
```

÷%6<1¤Y<UWSU°E -S§-yO0^G°gl4OPnO.£´82EsK,T0l8%K%-Y»üGIZh0%¹´A

Bã{Üç^'hO¯ê∫úBA\ÍMRÍg¨ãÚÍ\ÉµB÷„,®F×¸ªOB,·S…XO?Ov²²úGÍ÷•Ar¤°•X:¨_gÍI÷ê2hACñ¤N-Ob~T,Q'ì÷.%A÷2568R™$ÍU-uÚ¡,cABÇEwøzÉ³MN¾¯ ©A2[3:¨SàJé¦fŒÚ»=Y®¿`•f…ù>Aã¨K½ŷ·ŠZþ¢N¨±mlåÍ1Ł¥U

’¯zûæIRYP‹•uOX<SBPA×Œ:AA–™IŠ,~1′Ø/SC[¹qS[ã™2•|Aã¬µA£{Bk¯°¤U²rôL¼YÐ\IOÀ¯IAYNøÐ?Ÿ¥ý¿’•Ø¼Ö ’ZßA–€s¯ä´Ù•HØ
ª=”ZñÇûøÞPSØÒ÷ÂHDŽÅÎq°PìÞÅìf øQ‰ßHþx$š•TLâ§Ö ûC53”ìœ¥Ù‡¯ØÊ§vÆPLXµ(|×¼ß:¯ÃOñ¾R£¶®æf,ƒÌHKÎÿa–Âõ®Ù;ã»°©rSëA
„¼¤.²ßXÊ‰æ„Þ’•¸|/3™ÐŒŽÿ›ø\ì&÷|-3VÅ××Yu¼ ßØ®ã|ö°.½Žc•Àþð÷ê{ßøÎÅ×Ž'ƒŸÃDãæ¶Ş¯3¯÷¥Ø>Ò÷ß¼ß•
ßaŸpÿ²rœWÕ$¿|uSC°ôÃ?ûÆÇÐÐ$„¿¯x´Ø*w²ZÞ+Bþ°Ô’|¡Ì-Ã×Š –þWÈþFwıûÃ<ÒM°ðª÷OßÆ¢¯ÕxãkÚ,°Ì„Au

2
›°OŸÜ÷c¡b[ŒÐ¢÷†Á÷§Æ™7ƒ÷ƒ07ƒø¥üıM¶¾ø¯ìì_IKØÊ³³ßFXB°ê…oŸOÅ„»ø®»ò.¢{ÕÅÐ8;,¯ðy_I¼ÜBKfDŸ¼¯Ä
O\ƒy1þø[Z°€ñ)A÷…\ìsŁ¿®ºvwy:ŧ÷ßÆÍ,¾âÞMBS|¹Ô

ŽsdIWuûÃ/ß¯¥•As¬°ØvüwMtPgÿ0lz°<2×|MSøg°VwGd<^›á¬-{ì,@OM|ãõ1P€[ê¢Z|•¤v›•ÐnñTŸI|ÐZ|›mØIrA€

#°ø•Qmä€‡1yhÖ@µN4½ß^v×3D"»)A-·1ó÷2„
p×~`Ìv1ª1·Ndr~wMÃ6ó(8•|4q1lŽÐ•10kó¯Ì•–¯‡'¥e31•JÉN)ô³3Ã^¨a÷á·tŒ¯¾TCS•

IVIW?~Iy'Gu+wBEy$', 6¡5G0Â˜fOO4E2%X%_%i= ân']+O^:IYT/ ®²pµJO¡ds4

rïâi³Ç~ûVÄ-®Q™HIBj°ad•H/æ¯«E~IQþÕO·8Aªþµ fOUâ¬Š¸1•Ø•U÷Z{à•ÒÿD ñYdIDU³´4Y¹ÐYdUI±³w(v~A±±âqÕ%ŠHO.·QŸbaOŒø^ÒÃ>ñ…ârNΦHn>-Ø~ðNÃ>+>ET/ôAU~Ø¥?,•kÒ°_ÕU¶khY•ßŒ´´3GÃ+³R.³aÐ±{·8goà³¨ñøV³ù]³øø°éÒbûETóûArèK^‡

±BN•#9¿7,¬•|¬à ¼Ã‰ùt0'Z…NRUO$ï{4Mà\³vÍ~ bBBOE%IA¼mßigÒ¯SO!'AVO<S"„Ø€ZO|L)
Ó¯(,,nM¿àK´ÆoÊ³JFÎþ_©__ý÷ÐB¿Ê²Z…¼ð‰Æþ

```
UæZ!I[ÿZA
+Ê™‡`8½-_Ib'²®~üâ…æuHã°Ùâ‡N°>÷¢D.s×‰Sv°±Z~™WÐ•)mÖ÷7¸Lû¨
üGÁÛ¸Ð~CÃÐ@¨¿YaùÑ~¿7®÷`I÷Ê"ÁÖÛyœ9ø2á`Ö÷vK'ßšÎ2ø¥ã Ö°÷wb~úpÛpáÛpšÛpÛ%Za–ß9®b=âÖ°ME
â-ÃÐ™bönôIqØ%?
 /!Š`¸I`S?©MO*Uɪ-÷$ö…?|ûk ™¨
áõ¯|ä%'–²!ûbG›ÖêMXÿb¹êàSÇfÿ|·üÊIcNã÷ßøÕÞ÷ü÷Ż»&ãvCŻkÃÕÕ5_Q÷+÷Ú²IÜ„H÷YNÁIH×N¨ùî3Ê}ªÊ†3Ê–ûBWØÑÃ{d»%pßÛmÖ÷>>÷ÚèÓÚ÷>>÷ÚèÓ÷>>÷ÚèÓÚ÷N/„»)»7u¿GG–âÉÅÀàOATpö9|a?~ìhöE#q;m?&&r™Ì^÷0'3~¯Êü°)¤sÉfÐŽ–f)•C$r®¨°Ê÷^:OÇ<L?ör¡Ar‡Ê¦¨]¤|0¥Öƒ)¦$Df9ÿ¨d*éc²'~piVÖßÌZüÁG²2~pm6ÐþBÁ;üqu%}÷Pq[I¤?_Î|S®¨$¢fXÌb?³‹m?>'ºþ¨
```

'^QO1A1O÷[*þR
•+{(IRW4F1à°ò,Ü¯Hu1c'Â²ßCBUŒÎâÉø¤s
'ãbôÁ¥ãÜŠl:¡ñÆ¥÷Ãò,Ž|ÕqÆ[ÉÑ÷nÉcusŠêɪIW~..¦¾=Ãñ6|Ö,0;ÆÃOJF

```
9Þ?=1pKV3SYQVK_A9x'GUÉ&IU2'0÷<±ãi-[••ÐE@÷ŒÉ',HcBÅ|hZI3ç&\EH0~f}OA%74}Y0UŒa5
üÚ&ÍlQ€ƒ~gã'×5Ç1o«Ð¯ºÿX²\ÐvÌÆ49nŸ÷Ã«Ð••••ïÛÜÎC'øf'S_
ª³mgãÍÃ`>[µ/ã'ñQÍÈ½ÿí'Z?o-- He `Ç÷ëÍTv Ãjy©.•Ù¯ñm øS±%aY•"2³•©Þ¨0-(Ã¨ñ76}â2S3Ì@|_ÐVO8G¤=1X&{ØÕ÷••
-£"
¦.;Ic÷ëi-ÎQ®hrR¡ä¼é_ãqwå¹=
Mìò-ùT¨¥ª-,<üÜqö:Ìcó@¼ã¯ìY¶ ·AÜ[VSÅ$Ti~»AQvFÎ·Ûâìö;$
QãåŠ-Ð*·v÷-ZâprtâôPðæqQ™ì79-2Ù',a%yNûfêâàü.ÙÐÑuâ¥{I«(¯_ã`ì»ò!_ÂkŠ¡O?Ïd¥Ø_p»t¨Ê¡ìÁPPY1¨"Ì
¥Ÿì`E¨ âß=QâBwÉ¡ú·Fìr<«-»áàŸøUkòÿ7ÛûâC÷·ó²ÆcþÉ
ùñŠ¨TMbûì;ÐVŸÙ_g¥¨ëf¿ù÷k7ÛÐW©ªÐÃ1a{é4ê*¸ßo-uûüì7qVì™Ð~3Ÿ²jz¡ÿ"Zƒ·¸úÃÐê'æ£ð*÷`ßãGüiigN•Ê9•à4,G¨TúÑì'i;ÎC1GJ:-Çe²
°ÂXôà÷5[£QÙ:¨=ÃÑ÷Ov›þ@ÃIzxtÇ&<°p÷ìPBß.
```

E8;lGY>1@8'hh70M=uD`@-XO6@`.•5w_uÇ4

$çδAB&Z«AIEBo-¨¨.ðEIPÒK|S.ý•N-úOJ'‰3y11·ýABr‰:•«AR1÷.¸:,(%8ê°µK2$»´t''i-#9vÿxδã÷?|®ÑÐ(L-ô°ðδE×8i°-þI¨l•1d.¨™YfÆÃ–[/•1Ç¨¨2>.¿GlÑEþ›gad•þ½°IlErsTSæ„ID'ò{±êp.
yû„"cUmcS@ÑaÎ‹øÜ¯7lÎ•Ÿç5(•ª°Cñ.Ôh²Ÿ_¢ôWQ¨GB\±&ŒÉ*–Ô°šs2Cd,!°uÆ_X]0-ß‹›ñæ$ckÅÅ4Ìîu•¸ÛL^°saóc«»™››T°

```
Sx`'0JDEUOOh'UaY6)xO7IUf!e<ZyNoR6YM6¡'Ł¿•³ONE."eò=Æ¤E;_?¢IOWfâIQ÷ß>2÷Ka¿*°7UL¿ì¼Bì•ò_Ts|^HA¨aú÷ã¹ChÃ[eÐWg-I<'
çÒ,§(mIC¿Wp÷]|e•^<!3•®W°•uU7·¶¼aI-áÛ~ìKſ
ªæI¸¼ã²ÃI6ÿ¿      «_ ®¼ùOÍµì¼:ófÃÆãÀ\¬®Ùÿ®×Ÿ•MU
•ś÷ÊæP@Ñ,rSvcI0,[¶°Ł¿-pI_5WT¾|¼Ã.YÃÃ-NÕ÷pF}÷SÐ¥sÃ¡µÉ¼•o¨•'°BM¼tßN
ò·d™\pÐ¢Ÿh²¿ E1Bì¬ãWãXÐ•ãìûT¥ÿ/2'É ˚Ì
```

øu4ØVA{°¶ 8GZpK6·KdŁA*jZ¡UR1ø-I1~¯}—"WdN‹OPP»gA)$16dsÿ"bŒE' G‹0TIuIcPúOs='\Ø×eV^A8%Ð·]h·
K¢Ī FO%ÝTIŒ¤-ÍÂv´ŽÙ&6HÚ-ã¦‰/&«n›{N/©pf¯O§YÄ¦íŽ/•.kÔ°|èÕTyÖ¸ŒœÎ´ÃÅ.wBFö¬»s

¡…ÏA ÷1_/çT

¿¡_4Sfr0^SÅSµ„A%;<^xc-9U:Uŕê»AW-1Ôƒ0€$VVUK'ÿñh,0L5£ôpMLB^UE0®yS„I•ŒæUU•I¦·ir1¼¯ÛŠ¶)ìX²YEAc¯OJÿq'u…_Axg¬ç¶xvUATYAIym@W-ƒy÷ôõ23µ^_:OsE"ß¶UI=/»¼ôõl'j^L Gm1•uZ¯»1~•WYE3²›_Lâh4þ6|0æB¯•Þì\OLv(#
1à§Aš=%QØ#

A¤<i0/ú÷OAá·A4kgã¨'IE¼þf€)£5ãV8s_J_uCU^™Aqpq^Iã-ê·µ{·Aa¨ƒ>o0%¨¶Y0ã~IS6÷O
âây†ap_]xW+-OÉ¥|JØ=s|¶$v÷nAØK† !Á°ù¨Yw$m¨ Ã@¥›
9¨îù›.Fî_[r™ª¸\÷D+ƒ¿*,f1'6U°Ovà·ñ1Ñq¨bÉÕ|O÷5Ù,m1mÕ6ê™-ù'W_¢[;;OKMFã-a°Oîf¤I²ÃûµzY~Š-PDbTf‰|Ø1%

5O‹,I÷ï¨Æ5Ö"¤nXÿ0TAhÃâ‡ºh°;ì|zòvW¶¶9÷ãñ)8tZ‹¢j1·ñU[j»—°_H&u°oDbj'yº"7®ø%Ti«µwß@¶hñÉ,ïó}-A*••ÍSÉA-ðfŠ÷ôI0ŤÃøÐZXŸßÐ›ã°'¡Y10*¥ødùb§Aüz¥0¯jqOMøN¹Up"¶¯c¯h1¨Ëø_NxIS@ …»%»90®©G·'AırxgÿZ0wn÷kÉZ÷4%QN2ðJ",'Éü;<¯3øñJO1÷¤zNømnÜ0=ª1'ƒ_ó¯Ia÷ðj$O-çR:Aø~øðãfyeuÑü)2:I|!=]úI¯d.%?à·3Yȼ9ã3%|;ÍâY-HŸÙw(-Ñ_øSZ|¥ü'ÐepZ1A•bÐS

z()5$P°F6°ôl₁™S¦!-'(„`lz»Eæ'ÿ®¯Pé5ί°®bQ°F;%ß¢iuJK|O,QZ¦`L62¥£™v(-P°B• ûE«ÐÙP°Fîl~ß™f%¥%Ç_"©2Ð¥Eø(-¯ßÙ¹%bI8]¥Ð5'ðu•"q
n%cæ'-¡Ø¥îÙ¯{H÷4ÙÇ°bÎ=«ÃüGîJcY®®P5™b_™%KH%£<Ð#:t¯"•<®€QP«DÉ¯™•"•2®t±ß™hT••¤«Xî"•5¥±F4Ð'•
nÃÐ14%Cl4ß°® "1¥«£2|'?»<4••Q|¾ØBJ4Ž %2H%£®®±:™h=EJ42)
šf1 Þ²•••'•cH%£®®ñ)ÑßTŠ4²ÎBŒ®®ñ)ÑBFJ4N•¥H%£k»ÐãgKAÃÙAJ4-AJ4-EJ4N'•:®¢ñ)Ñx‡™ñü|4h-¯ß'-)Ñh
%M»Dz_"hHMƒ)Ñ8S'çH%£ ™hÙßJ4.••®¢®ÉJ42H%ƒ_1 `••H%Aj®`••®¢1FJ4.'••¥•uñ)Ñ,FJ4þ@J4þDJ4>!%Ÿ'•/H»pP°ÉÉÞ™\«49E•-o×…o?"o•o•"o••o-+ß²(ßZ%ü‡ŧw[#W™o•Ö…o•6°o•o•"o••o-+ß²(ßZ%ü‡ŧw[#W™o•Ö…o•6°o•o•"o••o-Ç|«æ.ó-šı·ÀûÀ·ÀûÀ·ÊêÀ·ÊÊ·šÉ·J1ß¯C{1-Á·ô—+âuÎ·ô-1¯c¼µ¼.|kyww[ã[ã[ã[ã[ã[ã[•ß²=Ñ²1ßÀ\/+/\V^XQ^,¢¼pUy¡ã[¯1ëÊÀ·VÖ…o·lßÃÜßTÞ#HycMyãcE?÷ì·¯?n)•|¢üqè:ó-âðó-æ•O•W>¹ü³ì|«ûã¾ôÊ"_þ¯Üò¹ôE=ðÎ·ŠkŽo×…o×…o×…o×…o×…o×…o×c¼U_gU¼\ß¯¼Ûß(2ß°n™¯o7ÊÎ·šÉÎ·Šæ[A+Ç·Š=ì·ŠkÎ·Šèì·ŠÀ·Àç{Êçâômßó-o•×Ÿ•¯••¿W^%£®¼ù-'æ[aü=âü-o-+™•-Pÿ>7Tb)ì)1Gé©Âì‹Ò*ã[eã[eã[eã[eã[eã[ãÿ®ß:>Þì³®D,£<Ê(ÎžÊ³®D,£<Ê(Îž

2ø"gâY¦ò3I2Aò,Sl2I2Aó,SdtWJUZg™'EZ¦$'%âI³LId-SYE"DO2%'µL)-µ)ð,'Þ"g"Ç'¡ A<E"šI³LGA<E"øY¦Ov<E"ëO2ø'µLYd-SYE"ëO2ø'µLYd-SZø-Cw<EoI2øYy-}ÿ¯g"r"y-)·¯g"YA<E"UZg"e'µI³EZfYd-³,²-YYE,‹¬e-ëO2E±¬øAO×'nðI2E>ó,³Ù¨2ø-›I³Iry-Y¯g"ã¶âY¦"²-Ö¯¬e*"ã™3,kðG¤¤I³LÃgZø"

aY|ÒdZe°-aY|ðI2•¶ÃYfEx-Y2eV„g™ÃYfEx-Y2eV„g™•¯g™ÃYfEx-Yñ™g™•ó,³OdZeVZI³IJA<E~¨I2«"k™Ù'µI°ÆZfUd-Ì°²-YYÉ~S~eVcYÉ£IŸ¥Y¨2@BÐ,âA+g™ÕÃ6,³UbZeVaYfµix-YYÉ-‰~eÐDD2k°k™5'µISÉZfMd-³EZ{AG=I³ISI<E-ß¯g™µ&ó,³ObZeOæYf--<0Z8D/'¤PA«g¤¨°™ŠâR6\pV¿þÜâçÉKFt{'ç~Om…Ât„rxWÉE-\.+•kq~ç1#%s™lRß-c+ÿ=ðzÉA*#Q›D¸O0_‰³_fA°;i&K¼NDZ„{OUß¯.FIé~#14'F°tJ$%!g•¿º„^…ï'Ua3ò¨Ã]ÐF&K´U•âãa }¬h°7ñ;|Od¢÷ò

‡¡EOQ¹ºZœ ß^ßØP[D•,†ARMZVüƒ•¶Ø]— ’¥'ıwyh><¤ADrPV¤D'•Ť¯Dн½ý,[Aq_&

Google Acquires Deja's Usenet Archive

~ Google Releases Beta Service in Transition ~

Acquired
Google Acquires Deja's Usenet Discussion Service. Read our press release.

New to Usenet?
Learn the basics about reading, posting and participating.

Looking for Deja.com's Precision Buying Service?
eBay's Half.com has acquired Deja.com's Precision Buying Service

Buy / Sell used CDs DVDs
$5 off $10 order @ Half.com

Login for Existing My-Deja Email Accounts
More about your account

Member Name:

Password:

Login

Forgot your password?

Search Groups

Google Search

Advanced Groups Search
Groups Help

Search Groups (Beta) Search the Web

Browse Groups

alt. (alternative)
Any conceivable topic...

biz. (business)
Business products, services, reviews...

comp. (computers)
Hardware, software, consumer info...

humanities. (humanities)
Fine art, literature, philosophy...

misc. (miscellaneous)
Employment, health, and much more...

news. (news)
Info about Usenet news...

rec. (recreation)
Games, hobbies, sports...

sci. (science)
Applied science, social science...

soc. (society)
Social issues, culture...

talk. (talk)
Current issues and debates...

Complete list of groups...

New! Google now offers a way to post messages on Usenet through Google Groups. This eagerly-anticipated improvement follows the recent addition of the full Usenet archive formerly maintained by Deja.com. Together, these upgrades provide you with complete access to Usenet data since 1995 and the ability to add your own comments to the more than 650 million messages already posted. We hope you will find this improved Usenet service of value. Google is committed to making Google Groups the best source of Usenet postings on the web and to that end, we will continue working to add the services and functions our users request.

To get the latest news on Google Groups and other Google initiatives, you may want to subscribe to the Google Friends newsletter. Or simply check Google Groups for updates.

For more information about the transition from Deja to Google, please read Google's press release.

```
" Vim syntax file
" Language:     Mail file
" Maintainer:   Felix von Leitner <leitner@math.fu-berlin.de>
" Last Change:  2001 May 09

" For version 5.x: Clear all syntax items
" For version 6.x: Quit when a syntax file was already loaded
if version < 600
  syntax clear
elseif exists("b:current_syntax")
  finish
endif

" The mail header is recognized starting with a "keyword:" line and ending
" with an empty line or other line that can't be in the header.
" All lines of the header are highlighted
" For "From " matching case is required, not for the rest.
syn region      mailHeader      start="^From " skip="^[ \t]"
end="^[-A-Za-z0-9/]*[^-A-Za-z0-9/:]"me=s-1 end="^[^:]*$"me=s-1 end="^---*"
contains=mailHeaderKey,mailSubject

syn case ignore

syn region      mailHeader
start="^\(Newsgroups:\|From:\|To:\|Cc:\|Bcc:\|Reply-To:\|Subject:\|Return-Path:\|Received:\|Date:\|Replied:\)"
skip="^[ \t]" end="^[-a-z0-9/]*[^-a-z0-9/:]"me=s-1 end="^[^:]*$"me=s-1
end="^---*" contains=mailHeaderKey,mailSubject

syn region      mailHeaderKey   contained
start="^\(From\|To\|Cc\|Bcc\|Reply-To\).*" skip=",$" end="$" contains=mailEmail
syn match       mailHeaderKey   contained "^Date"

syn match       mailSubject     contained "^Subject.*"

syn match       mailEmail       contained
"[_=a-z\.+A-Z0-9-]\+@[a-zA-Z0-9\./\-]\+"
syn match       mailEmail       contained "<.\{-}>"

syn region      mailSignature   start="^-- *$" end="^$"

" even and odd quoted lines
" removed ':', it caused too many bogus highlighting
" order is imporant here!
syn match       mailQuoted1     "^\([A-Za-z]\+>\|[]|}>]\).*$"
syn match       mailQuoted2     "^\(\([A-Za-z]\+>\|[]|}>]\)[ \t]*\)\{2}.*$"
syn match       mailQuoted3     "^\(\([A-Za-z]\+>\|[]|}>]\)[ \t]*\)\{3}.*$"
syn match       mailQuoted4     "^\(\([A-Za-z]\+>\|[]|}>]\)[ \t]*\)\{4}.*$"
syn match       mailQuoted5     "^\(\([A-Za-z]\+>\|[]|}>]\)[ \t]*\)\{5}.*$"
syn match       mailQuoted6     "^\(\([A-Za-z]\+>\|[]|}>]\)[ \t]*\)\{6}.*$"

" Need to sync on the header.  Assume we can do that within a hundred lines
syn sync lines=100

" Define the default highlighting.
" For version 5.7 and earlier: only when not done already
" For version 5.8 and later: only when an item doesn't have highlighting yet
if version >= 508 || !exists("did_ahdl_syn_inits")
  if version < 508
    let did_ahdl_syn_inits = 1
    command -nargs=+ HiLink hi link <args>
  else
    command -nargs=+ HiLink hi def link <args>
  endif

  HiLink mailHeaderKey          Type
  HiLink mailHeader             Statement
  HiLink mailQuoted1            Comment
  HiLink mailQuoted3            Comment
  HiLink mailQuoted5            Comment
  HiLink mailQuoted2            Identifier
  HiLink mailQuoted4            Identifier
  HiLink mailQuoted6            Identifier
  HiLink mailSignature          PreProc
  HiLink mailEmail              Special
  HiLink mailSubject            String

  delcommand HiLink
endif

let b:current_syntax = "mail"
```

```
" vim: ts=8
```

www.slrn.org / slrn.sourceforge.net

home of the slrn newsreader

Home

Last update: Fri Jun 8 22:39:01 CEST 2001

Introduction

This is the official home page of slrn. slrn (``s-lang read news'') is a newsreader, i.e. a program that accesses a newsserver to read messages from the Internet News service (also known as ``Usenet''). It runs in console mode on various Unix-like systems (including Linux), 32-bit Windows, OS/2, BeOS and VMS. Beside the usual features of a newsreader, slrn supports scoring rules to highlight, sort or kill articles based on information from their header. It is highly customizable, allows free key-bindings and can easily be extended using the sophisticated s-lang macro language. Offline reading is possible by using either slrnpull (shipped with slrn) or a local newsserver (like leafnode or INN).

The current status of slrn

The latest stable version of slrn is 0.9.7.1, released on 2001-06-06.

What's new?

[2001-06-06] slrn 0.9.7.1 released

The new stable version of slrn is finally there. It has a fully customizable group mode and status bars, highlights URLs, makes use of some common NNTP extensions, offers better documentation and has lots of minor improvements and bugfixes over the previous version.
As always, you can look at the full list of changes or download the thing directly.

[2001-04-19] Full documentation online

All the documentation that comes with slrn's source distribution can now be found on our new documentation page. You will find both plain text and (where available) HTML versions there and you can download tarballs for offline reading.

[2001-03-28] slrn 0.9.7.0 released

This page

- Status
- News
- Get slrn
- Mailing list
- About us

Site map

- Home
- Docs
- Manual
- Cleanscore
- Macros
- Patches
- Wishlist
- Links

Hosting

SourceForge Logo

This project is kindly hosted by SourceForge.

HTML validation

I finally released the new, long awaited stable version of the slrn newsreader, 0.9.7.0. Besides adding exciting new features, it also fixes a lot of annoying and some (potentially) security relevant bugs, so I recommend the upgrade. You might either want to look at the changes or go straight to the download page.

Thanks to everyone who helped by contributing code, writing documentation or reporting bugs.

For older news, please see the full history.

Where to get slrn

Getting the source code

You may obtain the latest version of slrn from our download page or via anonymous FTP. It is also available from various FTP mirrors, including:

- ftp://ftp.uni-stuttgart.de/pub/unix/misc/slang/slrn/
- ftp://ftp.fu-berlin.de/pub/unix/news/slrn/
- ftp://ftp.ntua.gr/pub/lang/slang/
- ftp://ftp.plig.org/pub/slrn/

Getting precompiled binaries

Please note that some binary packages include patches that are often, but not always useful. If you want to be sure to have a ``vanilla'' slrn installation, you should get the source code and compile it yourself. You also need the source code if you want to apply the patches found on this page.

- Debian provides packages of slrn for their system. If you want to use the current version, you probably need the packages from the ``testing'' or even the ``unstable'' distribution.
- If your RPM-based distribution does not include a recent binary, you can look for Linux RPMs of slrn at www.rpmfind.net.
- Hynek Schlawack has a nice homepage with AmigaOS packages.
- Francesco Cipriani offers a precompiled OS/2 version.
- Win32 binaries are available from our FTP space.
- Ted Stodgell provides a BeOS binary.
- Gerhard Lenerz offers precompiled IRIX binaries for

download.

Our public mailing list

We have a public mailing list, which is meant as a platform for questions and discussion about our pages and the patches and add-ons found here. We will also use it to inform our users about significant changes to slrn and this website.
If you want to subscribe, please visit the listinfo page. If you have general questions about slrn, please use the appropriate newsgroup.

Who maintains this page?

This page is currently maintained by four people:

- Thomas Schultz <tstsls@gmx.de> is the new maintainer of slrn; he also does the layout of the website and provides content.
- Sven Guckes <guckes-slrn@math.fu-berlin.de> provides content and collects large amounts of useful slrn-related information from both Usenet and the WWW.
- Felix Schueller <fschueller@netcologne.de> writes patches and is the author of cleanscore.
- Matthias Friedrich <mafr@topmail.de> works on the manual and writes patches.

If you want to contact us, please write to slrn-admin@lists.sourceforge.net - this is a private mailing list read by all of us (and us only). If you think your email might serve as the basis of a public discussion, consider subscribing to our public mailing list instead.

URL: http://www.math.fu-berlin.de/~guckes/slrn/pics.html
URL: http://www.slrn.org/pics.html
Created: Tue Oct 1 00:00:00 MET 1996
Last update: Wed Apr 12 18:53:39 MET DST 2000

SLRN - Pictures and Screenshots

This page shows some pictures of the newsreader SLRN.

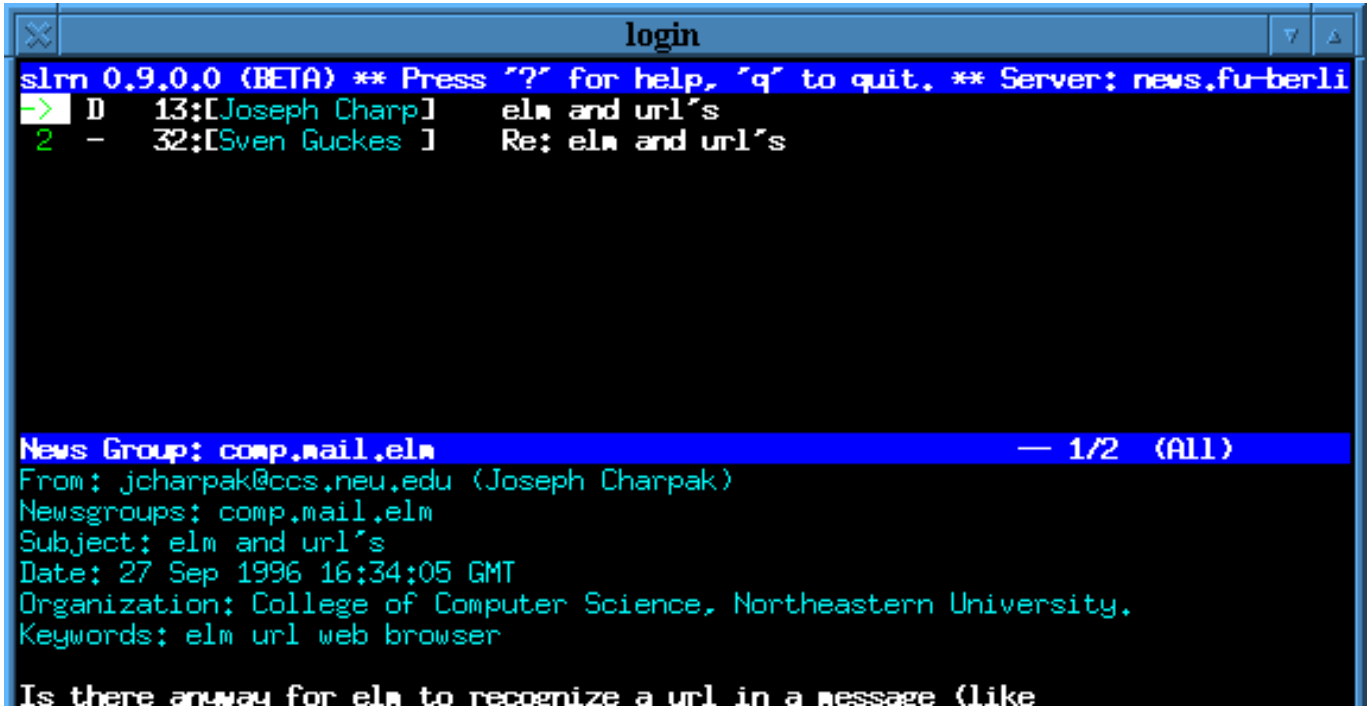
A screenshot of slrn on X:
http://der.kugelblitz.de/oliver/demo.html

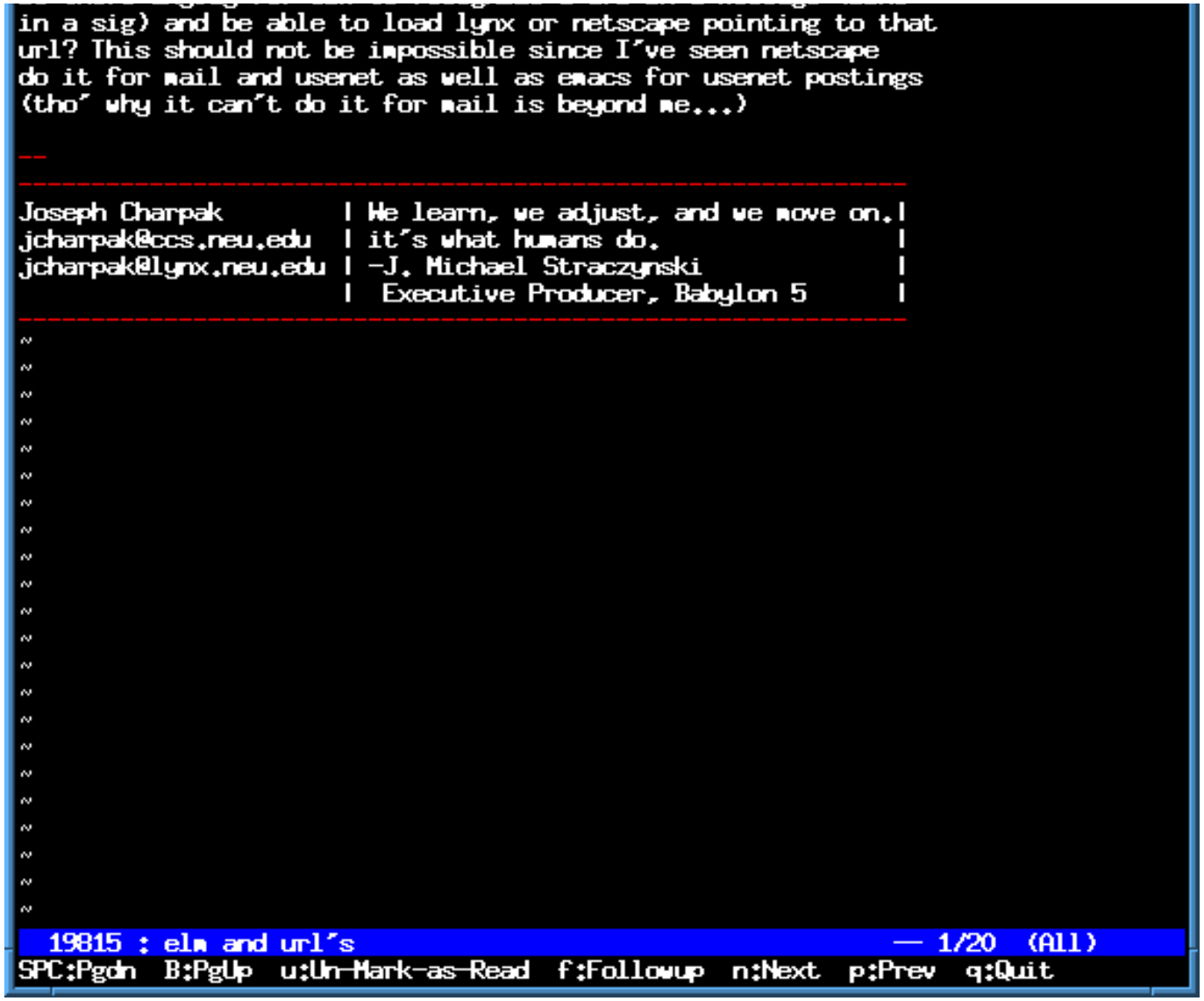
Author: Oliver Titz titz@gmx.de

---

recognition of signature

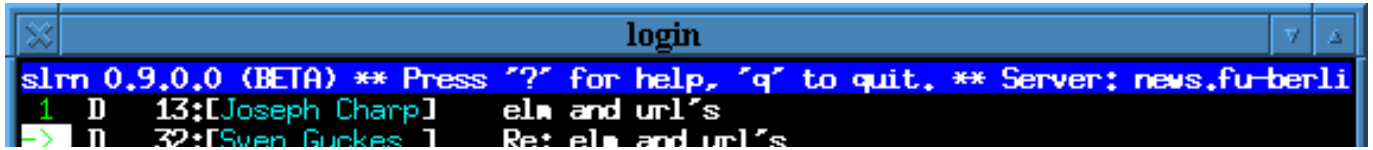
The signature of this post has incorrect sigdashes (inly "--" instead of "-- "):

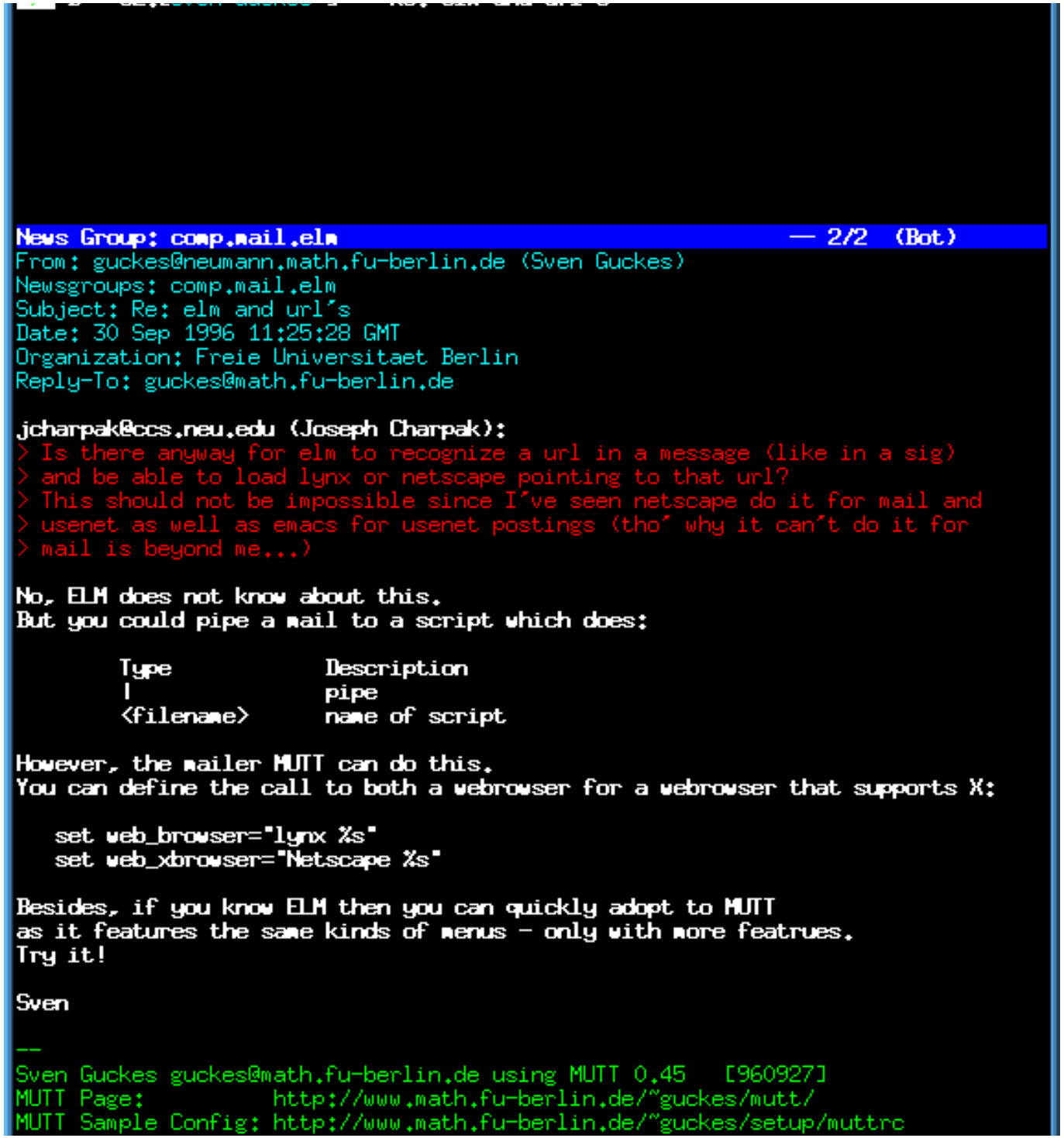




The poster did not apply correct sigdashes before his signature - thus it is not recognized as such and only partly understood as quoted text (dashes at the start of the line).

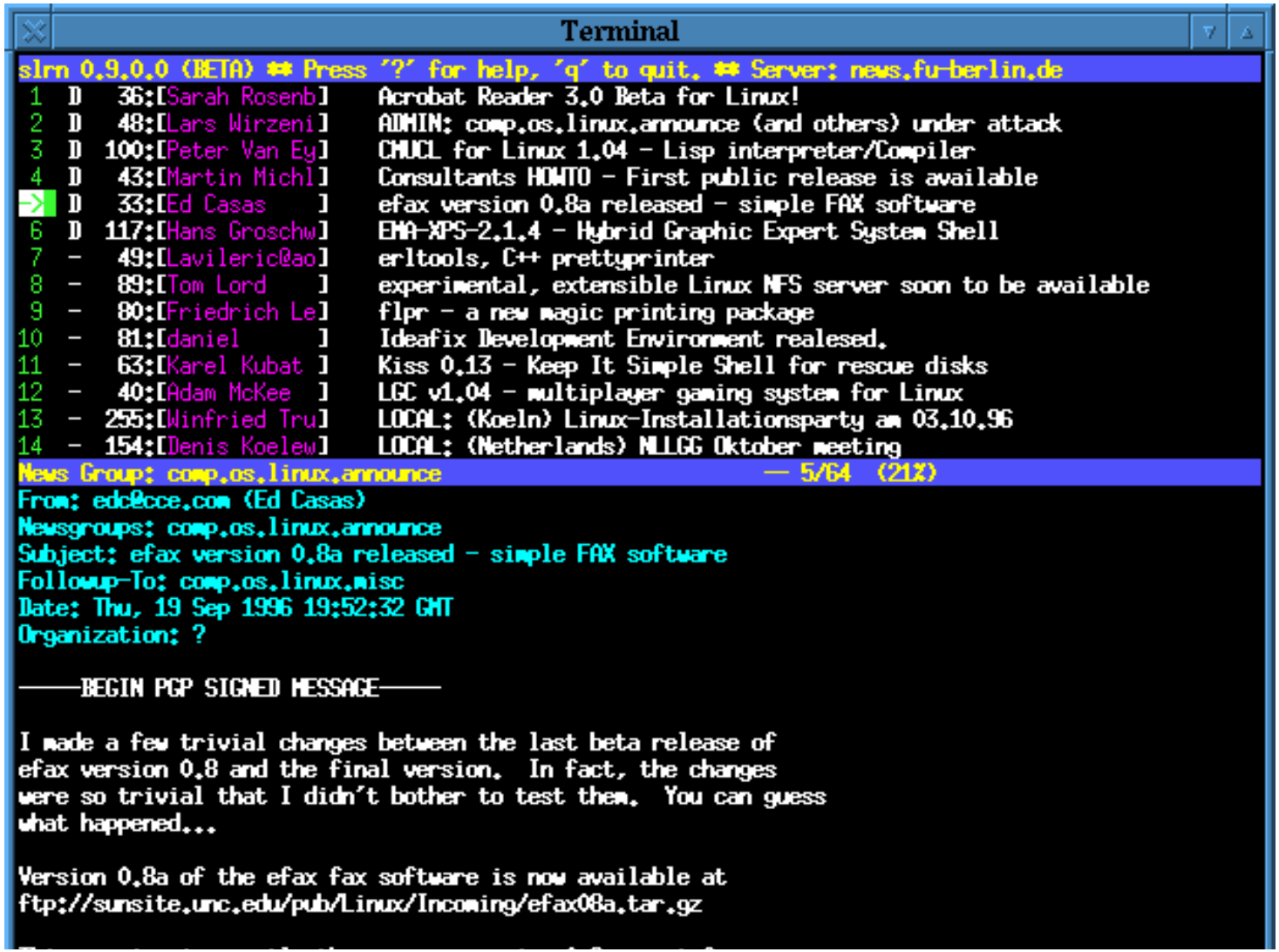
A picture of someone who posted with correct sigdashes:




News Group: comp.mail.elm — 2/2 (Bot)
From: guckes@neumann.math.fu-berlin.de (Sven Guckes)
Newsgroups: comp.mail.elm
Subject: Re: elm and url's
Date: 30 Sep 1996 11:25:28 GMT
Organization: Freie Universitaet Berlin
Reply-To: guckes@math.fu-berlin.de
jcharpak@ccs.neu.edu (Joseph Charpak):
> Is there anyway for elm to recognize a url in a message (like in a sig)
> and be able to load lynx or netscape pointing to that url?
> This should not be impossible since I've seen netscape do it for mail and
> usenet as well as emacs for usenet postings (tho' why it can't do it for
> mail is beyond me...)
No, ELM does not know about this.
But you could pipe a mail to a script which does:
Type Description
| pipe
<filename> name of script
However, the mailer MUTT can do this.
You can define the call to both a webbrowser for a webbrowser that supports X:
set web_browser="lynx %s"
set web_xbrowser="Netscape %s"
Besides, if you know ELM then you can quickly adopt to MUTT
as it features the same kinds of menus - only with more featrues.
Try it!
Sven
--
Sven Guckes guckes@math.fu-berlin.de using MUTT 0.45 [960927]
MUTT Page: http://www.math.fu-berlin.de/~guckes/mutt/
MUTT Sample Config: http://www.math.fu-berlin.de/~guckes/setup/muttrc

As you can see, the signature is recognized by sigdashes line ("-- ") and thus coloured in an extra colour (green).

Yes, the second picture shows the followup of the post in the first picture. ;-)

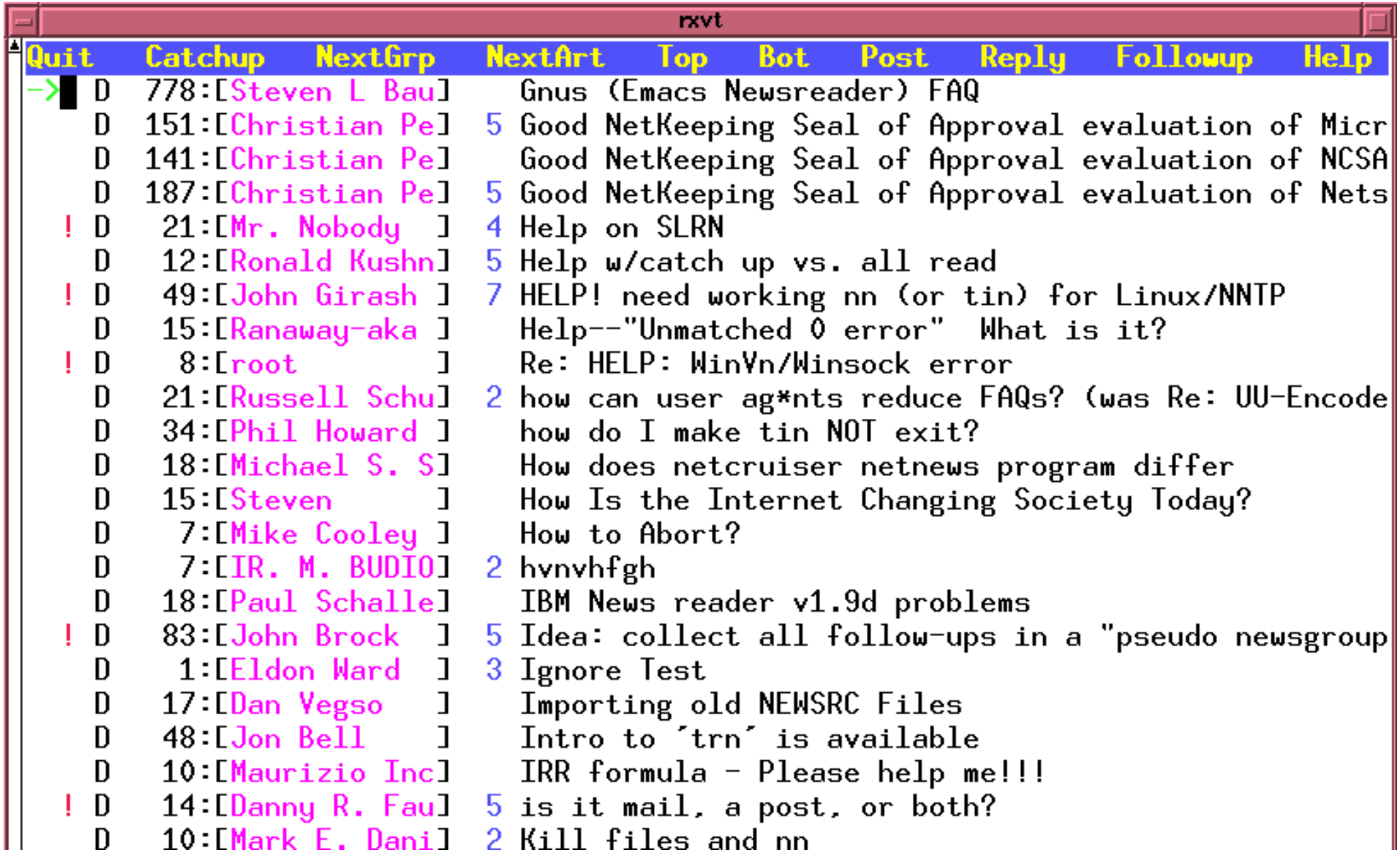


A pgp signed post of the announcement to a new version of efax. Different setup of colours.

JED's screenshots

The author, JED, had these screenshots on his page on SLRN:

A look at news.software.readers with **collapsed threads**, ie the threads are not presented as trees, but in one line each with the number of articles within given before the subject lines (in light blue):

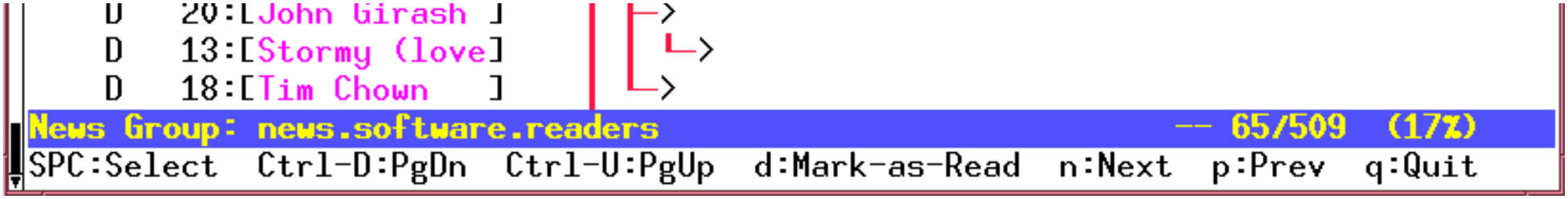


```
    D    9:[Jim Chmura   ]    Kill files for Netscape 2.0?
    D   10:["Randall J. ]  4 Killfile in Netscape Newsreader?
News Group: news.software.readers                          -- 31/168  (32%)
SPC:Select  Ctrl-D:PgDn  Ctrl-U:PgUp  d:Mark-as-Read  n:Next  p:Prev  q:Quit
```

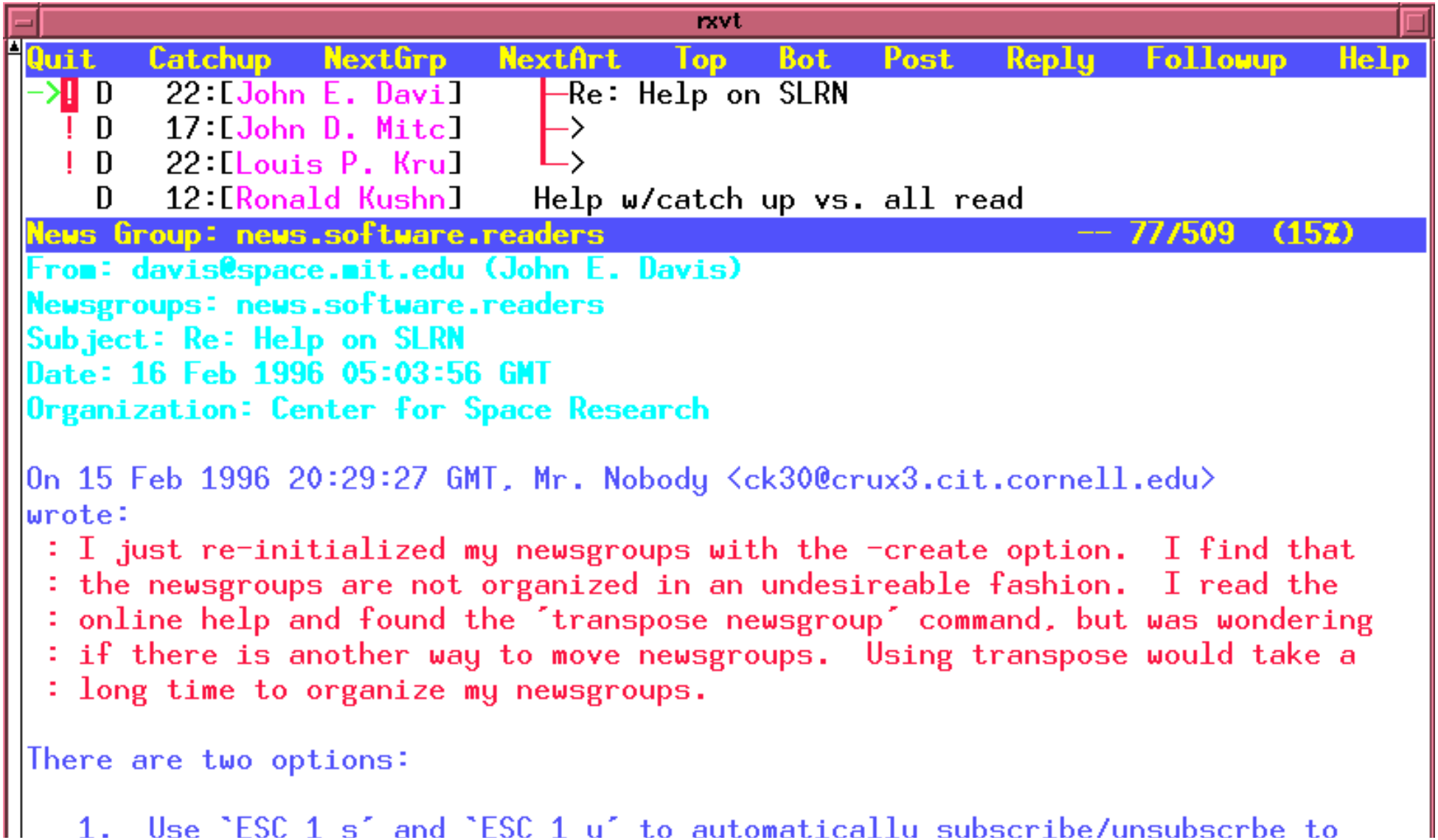
A look at news.software.readers - but with "opened" (uncollapsed) threads, ie each thread is show as a small tree:

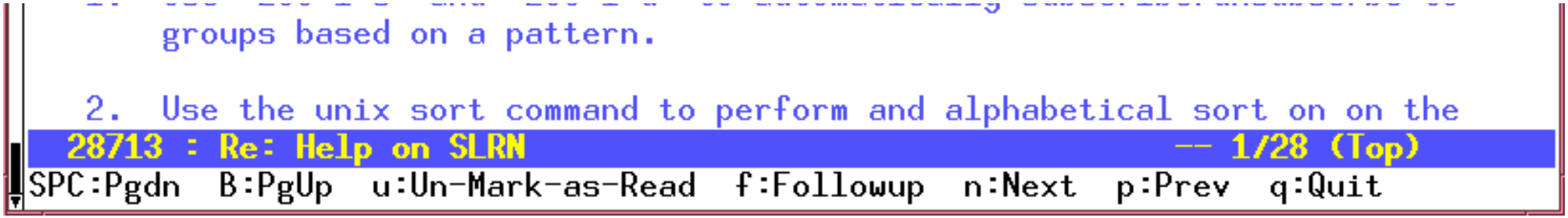
rxvt

```
Quit   Catchup   NextGrp   NextArt   Top   Bot   Post   Reply   Followup   Help
->  D  151:[Christian Pe]    Good NetKeeping Seal of Approval evaluation of Micr
    D   17:[Olaf Titz   ]    └─Re: Good NetKeeping Seal of Approval evaluation o
    D   10:[Larry W. Vir]      └─>
    D   24:[J.D. Falk   ]        └─>
    D   30:[Tim Pierce  ]          └─>
    D  141:[Christian Pe]    Good NetKeeping Seal of Approval evaluation of NCSA
    D  187:[Christian Pe]    Good NetKeeping Seal of Approval evaluation of Nets
    D   11:[Alan Coopers]    ├─>
    D   19:[Richard Clay]    └─>
    D   28:[Tim Pierce  ]      ├─>
    D   15:[Christian Pe]      └─>
  ! D   21:[Mr. Nobody  ]    Help on SLRN
  ! D   22:[John E. Davi]    ├─>
  ! D   17:[John D. Mitc]    ├─>
  ! D   22:[Louis P. Kru]    └─>
    D   12:[Ronald Kushn]    Help w/catch up vs. all read
    D   25:[Dong-Ick Lee]    ├─>
    D   13:[Luu Tran    ]    └─>
    D   22:[Tim Pierce  ]      ├─>
    D   19:[Paul Smee   ]      └─>
    D   49:[John Girash ]    HELP! need working nn (or tin) for Linux/NNTP
    D   33:[John Girash ]    ├─>
```



A look at an article within news.software.readers. Please note that the quoted text is presented in its own color (red):





Send feedback on this page to
Sven Guckes guckes@math.fu-berlin.de

VIM HowTo - Mirror Vim

Some tips on mirroring Vim - web pages or the ftp site.

HOWTO Mirror the Vim Pages - WWW Mirror

The Vim Home Site can easily be mirrored with these script:

webcopy mirror script

http://www.vim.org/.mirror

This script is used to copy Sven's pages over to www.vim.org.

wget mirror script [010416]

http://www.vim.org/script/webmirror.wget.txt

The adjusted webcopy script which now should work with "wget". Untested. Please send feedback on this one - thanks!

Author: Felipe Contreras fcont(at)matem.unam.mx [010413]

This script is using webcopy, which can be obtained at:

- ftp://ftp.inf.utfsm.cl/pub/utfsm/perl/

However, I cannot see that it is supported still.

Therefore I suggest "wget" which seems to be supported quite well:

- http://sunsite.dk/wget/ (HomePage of wget)

See also:

- http://appwatch.com/Linux/App/724/data.html

If you want to share your experience with using wget here then please send me an email!

In case you want to start an ftp or web mirror then please let me know. Just fill out the generic entry on the vim distribution page so I can add you to the list of sites. Thank you! :-)

Some FAQs on Mirroring

[Space]

How much disk space is required to mirror www.vim.org?

> Currently [981124] you need 620K for the Vim Pages (including the Vim HowTos), the same amount for the compressed archive of HTMLized helptexts, and 1.5M for the rest, ie the announcements, the old and new FAQ, some info on the ICCF and a few press articles on Vim, some texts, and a few additional syntax and setup files. The VIM Pages are less than 500K; there are these subdirs: announce/ faq/ howto/ iccf/ press/ syntax/ download/ faqnew/ html-5.3/ macros/ src/ txt/ Disk Usage and description: $ du ^[A-Z]*(/) 485 announce Announcements [does not change much] 624 download just has an archive of the latest HTML docs 133 faq the last version of the VIM FAQ 71 faqnew the new VIM FAQ (changing from time to time) 118 howto HowTos - could change a lot 2491 html-5.3 the online docs in HTML 1 iccf about the ICCF (remember: Vim is CharityWare) 1 macros macros - lots to come here still 14 press some articles on Vim that have been printed 71 src setup files that can be "sourced" 187 syntax syntax files 385 txt some non-HTML texts (todo files) So that's 2.5M for the HTMLized docs and 2M for the rest.

[Directories]

> FTP Mirror: Please make vim available as "/pub/vim", too, by adding a symlink. Thanks! We are trying to make it easy for people to find vim, and so far almost all mirrors have made it possible. :-)
>
> WWW Mirror: I suggest the address "mirror/www.vim.org" if you mirror the Vim Pages from www.vim.org. But this is entirely up to you, of course.

[Links]

A lot of links seem to be broken - why?

> Many links point relatively to other of Sven's pages - so the mirrored pages will point to other pages relatively, too. As mirror scripts do not copy all of Sven's pages, but only those with a path ending in "vim", those links may not work. Sorry about that. [Is there a way to tell the web server to change these links to the original site?]

[Duplicates]

Why does my mirror script get several copies of one page?

> Please note that several pages have different URLs - short ones that are easy and fast to type, and long URLs which are more descriptive. So before you mirror www.vim.org you should create the symlinks so the mirror script knows that you already have a page. This speeds up mirroring immensely as pages are retrieved only once.

Mirror Sites - Generic Entries

Please send me info about your mirror so I can add it to the list. Just fill out one of the following generic entries and send them to me at guckes@vim.org, - thanks!

GENERIC Generic Site Entry for **FTP mirrors**:

```
FTP.XY dates:   [FirstDate,LastChange]
FTP.XY place:   Continent, Country, City; Company/Organization
FTP.XY alias:   ftp://ftp.COUNTRY.vim.org/pub/vim/
```

```
FTP.XY source:  NICKNAME or URL
FTP.XY target:  URL
FTP.XY contact: NAME+ADDRESS
FTP.XY info:    ??? max connections; (Your server's time zone)
```

GENERIC Generic Site Entry for **WWW mirrors**:

```
WWW.XY dates:   [FirstDate,LastChange]
WWW.XY place:   Continent, Country, City; Company/Organization
WWW.XY alias:   http://www.COUNTRY.vim.org/path/
WWW.XY source:  NICKNAME or URL
WWW.XY target:  URL
WWW.XY contact: NAME+ADDRESS
WWW.XY info:    ??? max connections; (TIMEZONE???)
```

```
URL:         http://www.math.fu-berlin.de/~guckes/vim/howto/mirror.html
URL:         http://www.vim.org/mirror.html (mirror)
Created:     Sat Sep 05 00:00:00 CEST 1998
```

Send feedback on this page to
Sven Guckes guckes@vim.org

```
URL:         http://www.math.fu-berlin.de/~guckes/lynx/
Created:     Fri Jul 28 00:00:00 MET 1995
Last update: Tue May 23 12:00:00 MET DST 2000
Latest version:
        release:     Lynx-2.8.3      [000423]
        development: Lynx-2.8.4dev2  [000522]

HomePage:       http://lynx.browser.org/
Distribution/Download:
        DOS     http://www.fdisk.com/doslynx/lynxport.htm
        Unix    ftp://www.slcc.edu/pub/lynx/current/
                http://sol.slcc.edu/lynx/current/
                1973412 Jan 28 09:31 lynx2.8.2dev.15.zip
                1761637 Jan 28 09:31 lynx2.8.2dev.15.tar.gz
                1408142 Jan 28 09:31 lynx2.8.2dev.15.tar.bz2
                2702008 Jan 28 09:31 lynx2.8.2dev.15.tar.Z
```

LYNX Pages

LYNX is a "web browser", ie a program which connects to "world wide web servers", requests data from them and interprets texts set in HTML (HyperText Markup Language).

Overview

NEWS

LYNX - Home Page

Some comments about the LYNX Home Page:

The **LYNX Home Page** is http://lynx.browser.org/.

Note: Do not mix this up with http://www.lynx.org/ - that one is taken by Lynx Technologies Inc in Ontario. [980420]

For now, please refer to the Subir Grewal's page at

http://www.crl.com/~subir/lynx.html

LYNX Mailing Lists

The normal posting address for lynx-dev is lynx-dev@sig.net . You need not be subscribed to post; if you post without being subscribed you should check the archive at http://www.flora.org/lynx-dev/htlm/ for replies.

The LYNX Developer Mailing List

aka: lynx-dev

Email: lynx-dev@sig.net

[TODO]

LYNX Users

Some guys who frequently appear on the mailing list lynx-dev:

Al Gilman

Email: asgilman@access.digex.net

Author of the LYNX FAQ

Hiram W. Lester, Jr.

Email: hwlester@pobox.com

WWW: http://pobox.com/~hwlester/

Larry W. Virden

Email: lvirden@cas.org

WWW: http://www.teraform.com/%7Elvirden/

Subir Grewal

Email: subir@crl.com

WWW: http://www.crl.com/~subir/lynx.html

LYNX - The Pages

LYNX Pictures [961003]

Some screen shots of the Lynx colour version. :-)

LYNX - Wishlist [950904,961001]

My personal feature wishlist for Lynx.

LYNX - setup file ("lynxrc") [950728]

An example "init" file for LYNX. Get this to start customizing your LYNX!

LYNX - Getting started

An attempt at an intro for using Lynx.

Why use LYNX?

Check HTML

: LYNX is very strict on HTML rules and thus is a pretty good HTML checker. It has a brilliantly simple editing/viewing switch which is more efficient than with other browsers. LYNX also features the HEAD command (usually bound to ']') which displays the result of an HTTP HEAD request, and thus is great for checking what headers a script is generating (thereby allowing to debug CGI scripts). [NS and IE don't have any way to do this, as far as I know.]

Speed

: It's fast and yet does not chew immense amounts of CPU or colour maps.

Memory

: Lynx uses less memory (unless you view really large documents).

Modems

: It runs over 2400 baud. Try it!

Text Only

: With LYNX you do need not leave a text-only environment, most notably terminals. And the display of pages as text is usually better on the eyes.

Sending Mail

: LYNX has a much better interface for sending mail on a "mailto" link than other web browsers. You can tell LYNX to use your favourite editor and it optionally includes the text of the page.

Stability

: LYNX hardly ever crashes.

Bandwidth Conserving

: LYNX skips graphics, thus saving bandwidth, CPU, and time. [OK, you can turn off "auto-load images" with other browsers - but the default is always ON. :-( ]

File Browser

: LYNX displays test files including directory files, which makes it a part-time file maintenance tool.

Text Input

: LYNX has access to all commands with mostly one-key input. No mouse is required, which is much safer ergonomically.

Forms

: LYNX also understands forms.

LYNX FAQs

Here are some FAQs that I read about elsewhere.

mailcap

: Q: Can I specify a different mailcap file other than "$HOME/.mailcap"?

: A: LYNX looks at the mailcap file of the current directory, too, so you can use a customized mailcap file with this workaround:

: Change to a different directory, create ./.mailcap and start lynx.

: Example:

```
$ cd /tmp                       # change to directory /tmp
$ echo -n > .mailcap            # create empty .mailcap
```

```
$ lynx -dump {URL} > {AUDIO}       # binary dump, or
$ lynx {URL}                       # interactive download
audio/basic  D)ownload, or C)ancel
[...]
$ rm .mailcap
```

Thanks to Paul Fox pgf@foxharp.boston.ma.us for this info. [971112]

Inline Images

Q: I have hit upon a page with inline images, but Lynx does not show me the links to them - how can I download the images? Do I have to restart Lynx?

A: No, there is no need to restart Lynx - just hit the key '*' and Lynx will reload the page and also show the links to the images making it possible to download them. And before you ask - use 'd' to download the image of the current link.
[970201,971112]

Q: Do I have to get and install Lynx before I can try it? Or is there a site where I can login and try it?

A: No need for installation - try it out! Make a connection to lynx.cc.ukans.edu and login with userid "lynx". "This is slightly limited Lynx, probably for security and capacity reasons, but it has a link to Inter-Links, which is a decent service."

Phil Helms phil@cccs.cccoes.edu
[970127]

Q: What can I do to localize errors with Lynx?

A: Use the option "-trace" and try to capture the output.

Q: Which is the best method to capture text from Lynx "trace" output?

A: good question ...

Q: Why can't you use Lynx as a filter in a command like "program | lynx -dump > file"?

A: It's a feature.

Workaround: Use a script:

```
#! /bin/sh
cat - > /tmp/L$$.html
lynx /tmp/L$$.html
exit 0
```

Printing on OpenVMS

Q: System info: OpenVMS-6.1 running Multinet 3.5 Rev.B., Lynx-2.6. "Everything appeared to have installed properly, with no errors after I put the latest ".h" files in place. Lynx runs quite well, except for the ability to use the (P)rint option. When selecting 'p' I get the message "Alert! Unable to open print option file."

A: On VMS the options for print and download are handled as temporary html files placed in sys$scratch. If you haven't defined one, that would default to sys$login. If you have, could there be some protection problem associated with it?
[961008]

Error message display duration

Q: How can I set the duration for showing error messages?

A: You cannot.

Problem: There is no way to set the timeout for error messages. This poses a problem for blind people who need some time to read the error message.

Solutions: Option menu variable of type "number". A default value can be given via a config variable for lynxrc (eg "errorduration"), a duration of zero would wait until the user presses a key. If a typeahead makes the message go away then it should be possible to redisplay the last message with a command.

Reference: http://lynx.cc.ukans.edu/lynx-dev/9511/0019.html

LYNX - Development

[TODO]

Links to Other Pages about LYNX

Documentation

LYNX 2.6 Help Files (local)

lynx-2.6.docs.tar.gz (~220K)

LYNX Help Pages (aka "Manual")

http://www.nyu.edu/pages/wsn/subir/lynx/lynx_help/

Available at current bandwidth prices (quite cheap incidentally)." -- Subir Grewal (grewals@acf2.nyu.edu) [961006]

Miscellaneous

Lynx-2.7.2 Online Help

URL: http://www.slcc.edu/lynx/release/lynx2-7-2/lynx_help/lynx_help_main.html

The most up-to-date online help pages about Lynx. [980202]

The Lynx Book (of Thanks)

URL: http://www.slcc.edu/lynx-book.html

Do you like Lynx? Do you want to say thanks to the author? Tell him! [961011,980202]

Yahoo on Lynx

URL: http://www.yahoo.com/Computers_and_Internet/Software/Internet/World_Wide_Web/Browsers/Lynx/

Links to check

Lots to do here, still. :-/

```
   Linkname: Lynx2.7 for DOS 386+ or Win32
        URL: http://www.fdisk.com/doslynx/lynxport.htm

LYNX FAQ:                  http://www.access.digex.net/~asgilman/lynx/FAQ/
LYNX Distribution:         http://www.crl.com/%7Esubir/lynx/platforms.html
LYNX Optimized Pages:      http://www.nyu.edu/pages/wsn/subir/lynx.html
LYNX Current Development:  http://sol.slcc.edu/lynx/current/
LYNX Online HELP: http://www.crl.com/%7Esubir/lynx/lynx_help/lynx_help_main.html

http://www.sunbeach.net/personal/dhill/lynx/lynx-tips.html
http://www.cc.ukans.edu/lynx_help/Lynx_users_guide.html
http://www.fhi-berlin.mpg.de/amiga/alynx.html
http://world.std.com/~adamg/dehanced.html
http://world.std.com/~adamg/ugh.html
```

```
ftp://ftp2.cc.ukans.edu/pub/DosLynx/readme.htm
http://www.ukans.edu/about_lynx/about_lynx.html
http://www.wfbr.edu/dir/lynx/

http://lynx.cc.ukans.edu/lynx-dev/

http://WWW.Sasquatch.Com/~jddjeff/Internet/World_Wide_Web/lynx/lynx-learners/
http://www.abs.net/~mikebat/ode-to-lynx.html
http://Gridley.ACNS.Carleton.edu/~darbyt/lynx.html
http://cybercom.net/~ellen/lynx.htm
```

Send feedback on this page to
Sven Guckes guckes@math.fu-berlin.de

URL: http://www.math.fu-berlin.de/~guckes/w3m/
Created: Sat Jan 01 00:00:00 MET 2000
Last update: Fri Feb 25 15:56:02 MET 2000

W3M - a text based web browser

W3M HomePage:
http://ei5nazha.yz.yamagata-u.ac.jp/~aito/w3m/eng/

Send feedback on this page to
Sven Guckes guckes-w3m@math.fu-berlin.de

```
" ============================================================================
" HTML - HTML - HTML - HTML - HTML - HTML - HTML - HTML
" ============================================================================
"
" File:      $HOME/vim/src/html.vim
" This file: <URL:http://www.math.fu-berlin.de/~guckes/vim/src/html.vim>
" Purpose:      Setup file for the editor Vim for Webpage/HTML editing
" Author:       Sven Guckes guckes@vim.org (guckes@math.fu-berlin.de)
"               <URL:http://www.math.fu-berlin.de/~guckes/>
" Comments:     See also:  <URL:http://www.vim.org/howto/html.html>
"               Please send comments to me - email preferred!
" See also:     http://www.math.fu-berlin.de/~guckes/www/html.html
"               HTML Mini Guide
" Last update:  Tue Mar 20 15:00:00 MET 2001
"
" HTML = HyperText Markup Language - the language of the World Wide Web
"
" ============================================================================
" HTML - useful abbreviations
" ============================================================================
"
"     YWWW  The start of a typical Web address:
  iab YWWW  http://www.
"
" ============================================================================
" HTML - working on files
" ============================================================================
"
"  Whenever I edit a web page I use ",e" to start the edit command.
"  And I use ",rn" to read in a web page template:
"       ,e  = edit www file
" The Webpages are on the web server which is mounted on /import/Mweb
  nmap  ,e :e /import/Mweb/guckes/
" nmap  ,e :e ~/.P/
" The directory ~/.P is a link to the directory with my webpages.
" the name is short so it will be short for the output of ":ls".
"
"   I used ",e" for the command here before, but this was part of the
"   default string for "highlight", so while entering the default
"   string the ",e" it got expanded.  Darn!
"
"       ,rm = read mailto (standard html file for a mailto link to my address)
" map   ,rm :r ~/.P/txt/mailto.html
"
"       ,rn = read new-page (standard html file for a new web page)
"       see http://www.math.fu-berlin.de/~guckes/txt/New.page.form.html
  map   ,rn :0r ~/.P/txt/New.page.form.html
"
"       ,mp = "make public"  - set permissions of file to "rw-r--r--"
  nmap  ,mp :!chmod 644 %<CR>
"       ,MP = "make private" - set permissions of file to "rw-------"
  nmap  ,MP :!chmod 600 %<CR>
"
" ============================================================================
"      wb - white body
"      include a body tage with white background
  iab Ywb  <body bgcolor="#ffffff">
" ============================================================================
"
" ============================================================================
```

```
" HTML - Inserting "single" tags
" ================================================================
"
  iab Ybr <br>
  iab Yhr <hr>
  iab Yp  <p>
"
" ================================================================
" HTML - making environments
" ================================================================
"
" HTML ,me = "make environment"
" make current word an HTML environment
 nmap ,me viwyi<<ESC>ea></<C-R>"><ESC>
 vmap ,me    yi<<ESC>ea></<C-R>"><ESC>
"
" Example:  "foo" -> "<foo></foo>"
"
" ================================================================
" HTML - inserting environments
" ================================================================
"
" Comment:
  iab  Ycom  <!--X--><ESC>FXs
" vmap ,com <esc>`<i<!--<ESC>`>i--><ESC>
" always comment *whole* lines blocks):
  vmap ,com <esc>`<O<!--<ESC>`>o--><ESC>
"
" Make paragraph
  vmap ,P   <esc>`<O<p><ESC>`>o</p><ESC>
"
" ================================================================
" HTML - changing the appearance/font of text
" ================================================================
"
" BlockQuoted Text:
  iab  Ybl  <blockquote></blockquote><ESC>T>i
" vmap ,bl  "zdi<blockquote><C-R>z</blockquote><ESC>2F>
"
" Bold text:
  iab  Yb   <b></b><ESC>T>i
  vmap ,b   "zda<b><C-R>z</b><ESC>F>
"
" Center Inline Text:
  iab  Ycenter <center></center><ESC>T>i
  iab  Ycen    <center></center><ESC>T>i
  vmap ,cen "zdi<center><C-M><C-R>z<C-M></center><ESC>?><CR>
"
" Center Text Blocks:
  iab  YCen <center><CR></center><ESC>O
  vmap ,Cen <ESC>'<O<center><ESC>'>o</center><ESC>
"
" Code in Text:
  iab  Ycod <code></code><ESC>T>i
  vmap ,cod "zdi<code><C-M><C-R>z<C-M></code><C-M><ESC>T>i
"
" Italic Text:
  iab  Yi   <i></i><ESC>T>i
  vmap ,i   "zdi<i><C-R>z</i><ESC>T>
"
```

```
" Typewriter Type:
  iab  Ytt   <tt></tt><ESC>T>i
  vmap ,tt   "zdi<tt><C-R>z</tt><ESC>T>
"
" ===================================================================
" HTML - Preserving Text Structure
" ===================================================================
"
" Preserve Text Formatting *with*  interpretation of HTML:
  iab  Ypre <pre></pre><ESC>T>i
" vmap ,pre "zdi<pre><C-M><C-R>z</pre><C-M><ESC>T>
  vmap ,pre mz:<ESC>'<O<pre><ESC>'>o</pre><ESC>`z
"
" Preserve Text Formatting without interpretation of HTML:
"           Insert environment <xmp></xmp>
  iab  Yxmp <xmp></xmp><ESC>T>i
" vmap ,xmp "zdi<xmp><C-M><C-R>z<C-M></xmp><C-M><ESC>T>i
  vmap ,xmp mz:<ESC>'<O<xmp><ESC>'>o</xmp><ESC>`z
"
" ===================================================================
" HTML - Making Tables
" ===================================================================
"
" Table Environment
  iab  Ytable <table><cr></table>
" Table Data
  iab  Ytd  <td></td><ESC>T>i
  vmap ,td  "zdi<td><C-R>z</td><ESC>T>
" Table Row
  iab  Ytr  <tr></tr><ESC>T>i
  vmap ,tr  "zdi<tr><C-R>z</tr><ESC>T>i
"
"
" ===================================================================
" HTML - Making Header Lines (h1 to h6)
" ===================================================================
"
  iab  Yh1 <h1></h1><ESC>T>i
  vmap ,h1 "zdi<h1><C-R>z</h1><ESC>2F>
  iab  Yh2 <h2></h2><ESC>T>i
  vmap ,h2 "zdi<h2><C-R>z</h2><ESC>2F>
  iab  Yh3 <h3></h3><ESC>T>i
  vmap ,h3 "zdi<h3><C-R>z</h3><ESC>2F>
  iab  Yh4 <h4></h4><ESC>T>i
  vmap ,h4 "zdi<h4><C-R>z</h4><ESC>2F>
  iab  Yh5 <h5></h5><ESC>T>i
  vmap ,h5 "zdi<h5><C-R>z</h5><ESC>2F>
  iab  Yh6 <h6></h6><ESC>T>i
  vmap ,h6 "zdi<h6><C-R>z</h6><ESC>2F>
"
" ===================================================================
" HTML - Making Lists and List Items
" ===================================================================
"
"           Insert "ordered list" with one list element
  iab Yol <ol><CR><li><CR></ol><ESC>k
"           Insert "unordered list" with one list element
  iab Yul <ul><CR><li><CR></ul><ESC>k
"           Insert "decsription list" with one list element
" iab Ydl <dl><CR><dt><CR><dd><CR><p><CR></dl><CR><ESC>4kA
```

```
  iab Ydl <dl><CR><CR><dt><CR><dd><CR><p><CR><CR></dl><CR><ESC>5kA
"
" Insert "list" item (for both ordered and unordered list)
  iab Yli <li>
" Insert "description list" item
  iab Ydt <dt><CR><dd><CR><p><CR><ESC>kA
  iab Ydp <dt><CR><dd><C-M><p><C-M><ESC>kkkA
"
" ====================================================================
" HTML - Making Links
" ====================================================================
"
"
"     A HREF (HTML 2.0)
" insert "generic link"
  iab  Yhref <a href=""></a><ESC>?""<CR>a
  vmap ,href "zdi<a href=""><C-R>z</a><ESC>F"i
  vmap ,HREF "zdi<a href="<C-R>z"></a><ESC>F"i
"
" make current URL a link:
  vmap ,link "zdi<a href="<C-R>z"<C-M><C-I>><C-R>z</a><ESC>F"i
"
" add link to current text:
  vmap ,text "zdi<a href=""><C-R>z</a><ESC>F"i
"
"     A NAME (HTML-2.0)
  iab  Yname <a name=""></a><ESC>?""<CR>a
  vmap ,name "zdi<a name="<C-R>z"<C-M><C-I>><C-R>z</a><ESC>2F>
"
"            Insert/make link to image
  iab  Yimg  <img alt="[]"<C-M>   align=<C-M>     src=""></a><ESC>?""<CR>a
"
" 000811
"            Insert a link to an image with a thumbnail
  iab YIMG   <a  href="PIC"><cr><img alt=""<cr>   align=right<cr>
src=""><cr></a><cr>
"
"            Insert/make mailto link
  iab  Ymail <a href="mailto:"></a><ESC>?:<CR>a
  vmap ,mail "zdi<a href="mailto:<C-R>z"<C-M><C-I><C-I>><C-R>z</a><ESC>2F>
  vmap ,Mail "zdi<a href="mailto:<C-R>z"><C-R>z</a><ESC>2F>
"
"            Insert/make link to newsgroup
  iab  Ynews <a href="news:"></a><ESC>?:<CR>a
  vmap ,news "zdi<a href="news:<C-R>z"><C-R>z</a><ESC>2F>
"
"      Ypage Insert page description with a possible link and text
  iab  Ypage   <C-M>page:<C-I><C-M>link:<C-I><C-M>text:<C-I><ESC>kkA
"
"
" For adding descriptions and keywords to important pages:
" <META Name="description" Content="Write your description here">
" <META Name="keywords"    Content="Write your keywords here">
"
" Colorizing Text
"
"      ,Cblu = colorize the selection with color "blue"
  vmap ,Cblu "zdi<FONT COLOR="#0000FF"><C-R>z</FONT>
"      ,Cgre = colorize the selection with color "green"
  vmap ,Cgre "zdi<FONT COLOR="#00FF00"><C-R>z</FONT>
```

```
"      ,Cred = colorize the selection with color "red"
  vmap ,Cred "zdi<FONT COLOR="#FF0000"><C-R>z</FONT>
"
" ====================================================================
" HTML - handling special text
" ====================================================================
"
" HTML - inserting special characters
  imap ;& &amp;
  imap ;K &copy;
  imap ;" &quot;
  imap ;< &lt;
  imap ;> &gt;
"
" HTML - inserting umlauts  [981110]
  imap \Ae &Auml;
  imap \Oe &Ouml;
  imap \Ue &Uuml;
  imap \ae &auml;
  imap \oe &ouml;
  imap \ue &uuml;
  imap \ss &szlig;
"
" HTML - Converting umlauts to digraphs [960430]
"
" cnoremap ,ae %s/ae/<C-V>228/gc
" cnoremap ,ae %s/ae/\&auml;/gc
" cnoremap ,oe %s/oe/<C-V>246/gc
" cnoremap ,ue %s/ue/<C-V>252/gc
" cnoremap ,Ae %s/Ae/<C-V>196/gc
" cnoremap ,Oe %s/Oe/<C-V>124/gc
" cnoremap ,Ue %s/Ue/<C-V>220/gc
" cnoremap ,ss %s/ss/<C-V>223/gc
"
"       ,= = turn "===" into headline of size 1
  nmap  ,= :%s/^===\(.*\)$/<h1>\1<\/h1>/c<CR>
" Example:
" before:  === New section
" after:   <h1>New section</h1>
"
" ====================================================================
" HTML 3.0
" ====================================================================
" environments:
" ABBREV ACRONYM AU BANNER BIG BQ CAPTION CREDIT DEL DFN DIR DIV FN
" HTML INS NOTE OL P Q S SMALL SUB SUP TAB V
" other:
" FIG LANG OVERLAY RANGE SPOT STYLE
"
" ====================================================================
" HTML - Misc
" ====================================================================
"
" HTML - Add closing tags to name tag
" (which I forgot when I started HTML):
" map ,,,, :g/^<a name="[a-z_]*">$/s/$/<\/a>/
" Example:
" before:  <a name="close_tag">
" after:   <a name="close_tag"></a>
"
```

```
" Insert/make reference link to overview list (short"cut")
  iab  Ycut  \| <a href="#"<C-I>></a><ESC>F#a
  vmap ,cut  "zdi<a href="#<C-R>z"<C-I>><C-R>z</a><ESC>2F>
"
" end of file vim: tw=999
```

Sven Guckes guckes@vim.org

Last update: Tue Aug 29 00:00:00
MET DST 2000

VI - Picture Gallery

Vi is a text editor. You cannot draw pictures with it as it does not give you visual feedback on colours. However, some vi clones have a GUI (graphic user interface) and support for colour. Naturally, the layouts differ on various screens.

This pages has links to some screen shots to give you an idea on the design of those various clones.

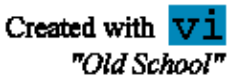
Vi Pictures - Overview

Vi Pictures - Icons and Logos

Vi Logos

Title: "Lovingly Handrafted"
Size: **675** bytes
Creator: David Fischer dave@cca.org
Added: 000829
The text on the button says: "This HTML lovingly handcrafted using vi" and it was created from a PostScript source. So this is the only buttons which was truly "designed with vim". ;-)

Title: "Created with vi - 'old school'"
Size: 1317 in bytes
Creator: Steve Cirian steve.cirian@us.pwcglobal.com
Added: 000726



Title "vipowered"
Size **Only 174 bytes!**
Creator Kevin G. Austin kaustin@tiny.net
Added 990808
Notes The smallest vi icon so far!

Title "Powered by VI (animated)"
Size [88x31] 5K
Creator Rui Silva rms@ssi.aaum.pt
Added 990302

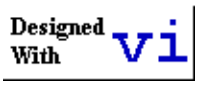
Title "Powered by VI"
Size 503 bytes
Creator Rui Miguel Silva Seabra rms@ssi.aaum.pt
Added 970413
Notes: "Vi - powered by" is in the style of those other "powered by" pictures, ie a logo followed by the "powered by" text.

Title "VI Powered (by Sven)"
Size only **214** bytes !
Creator Sven Guckes guckes@vim.org

Added 970413

Notes: This logo is based on Rui Silva's logo - only this logo is much much smaller. A sort of "minimalistic icon". ;-)

Title "Designed with VI"



Size only **441** bytes!

Creator: Mike VanHorn mvanhorn@dma.org

Added 971118

Notes: Mike has made some icons for other editors, too:

- http://www.dma.org/%7Emvanhorn/editors.html

Title: "Made With vi"

Size: 35,942 bytes

Creator: Paul Sherren paul5@roadrunner.nf.net

Added: 990105

Notes: An animated GIF on a square black background. The "vi" ticks from left to right.

Title: "Vi"

Size: only **998** bytes!

Creator Sven Guckes guckes@vim.org

Added 980130

Notes: GIF89a - despeckled, interlaced, and transparent background. Taken from the "vi" picture of Jim Kingdon kingdon@cyclic.com

Title: "made with vi"

Size: only **761** bytes!

Creator: Alex Shnitman alexsh@linux.org.il http://alexs.home.ml.org

Added: 971122

Title: "This Site vi powered"

Size: 1310 bytes

Creator: Antonio Valle avalle@abast.es http://www.abast.es/~avalle/

Added: long ago

Title: "written in the vi editor"

Size: 5815 bytes

Creator: Dr. Nick nep@creamcity.com

Added 980105

Notes: Animated Picture showing "written in the vi editor" with a world revolving into a 'v' on red background.

Title: "the VI editor"

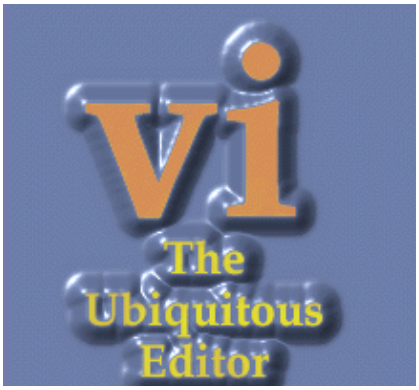
Size: 5655 bytes

Creator: I forget. ;-(

Added: 970207

Notes The text "the VI editor" in a nice cursive font.

Title: "vi - The Ubiquitous Editor"



Size: 21201 bytes

Creator: David Lee Heyman david@macom.co.il http://have.zets.org/~yosi

Added: 970207,970418

Description: A big "vi" and the text "The Ubiquitous Editor" in yellow letters embossed on blue metal plate.

Notes:

```
ubiq-ui-tous \y:u-'bik-wet-es\ adj
     WIDESPREAD
```

```
:existing or being everywhere at the same time:
:constantly encountered:
-- ubiq-ui-tous-ly adv
-- ubiq-ui-tous-ness n
```

Variations of the "Vi Logo":

Title: "Vi"
Size: 9815 bytes
Creator Jim Kingdon kingdon@cyclic.com
Added: 980130
Notes: Taken from the picture of David Lee Heyman

Title "Vi Lover GIF"
Size 7719 bytes
Creator Charles Vidal vidalc@club-internet.fr
http://www.chez.com/vidalc/
Added [980130,980213]

Vi Clones Logos

Vile Icons

Title Designed with Vile (2K)
Size 2572 bytes
Creator Shawn Halpenny malachai@iname.com
Added 971006

VIM Pictures -> Extra Page

The screenshots for vim have outnumbered those of the other clones - so I had to move them back onto their own page:

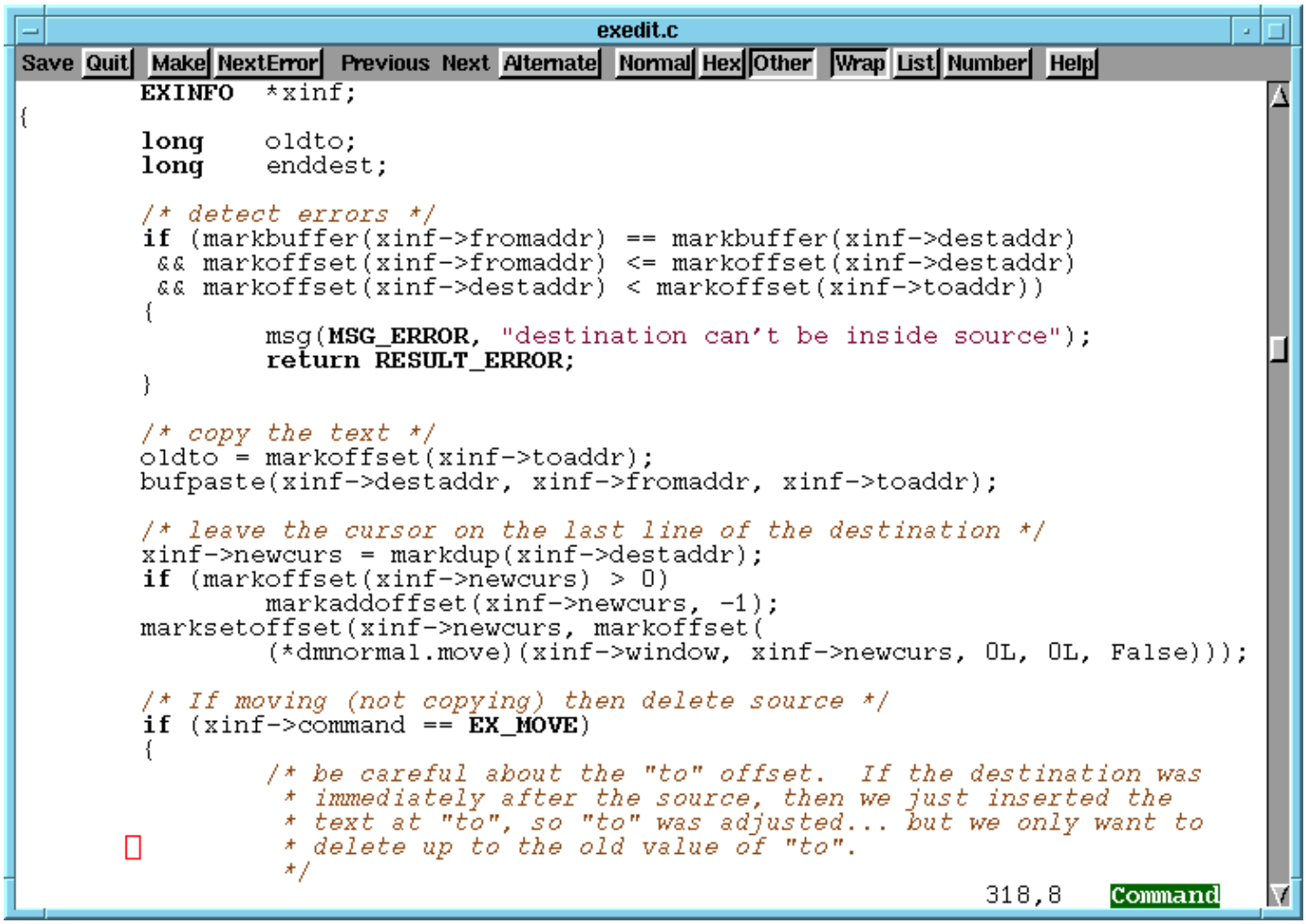
http://www.vim.org/pics.html

And everything you want to know about VIM is on the VIM Pages:

http://www.vim.org/

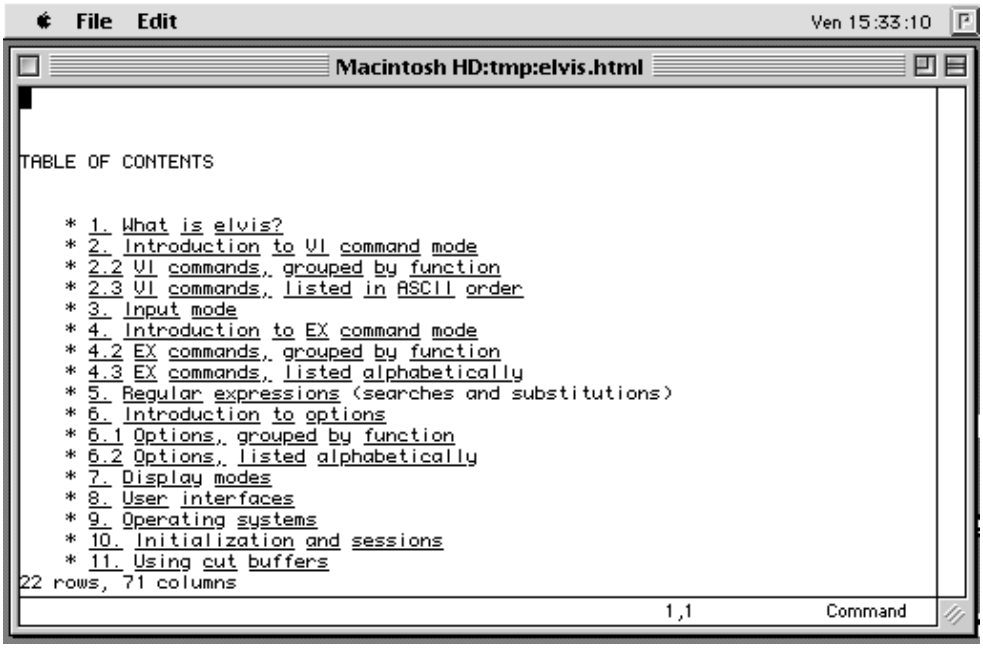
Elvis

The main distribution site for Elvis is:
ftp://ftp.cs.pdx.edu/pub/elvis/README.html
Creator: Steve Kirkendall kirkenda@cs.pdx.edu
Title: Elvis 2.0



Elvis 2.0 creates a separate top-level application window for each editor window. This example shows some C source being edited, in the "syntax c" display mode. The stripe across the top of the window is a user-configurable toolbar. The red rectangle near the lower-left corner is the cursor. In a real elvis window it flashes, and it changes shape to indicate the current mode (rectangle=command, bar=insert, underscore=replace). The scrollbar is NOT a Motif scrollbar, but an incredible simulation.

MacElvis



"MacElvis on MacOS"

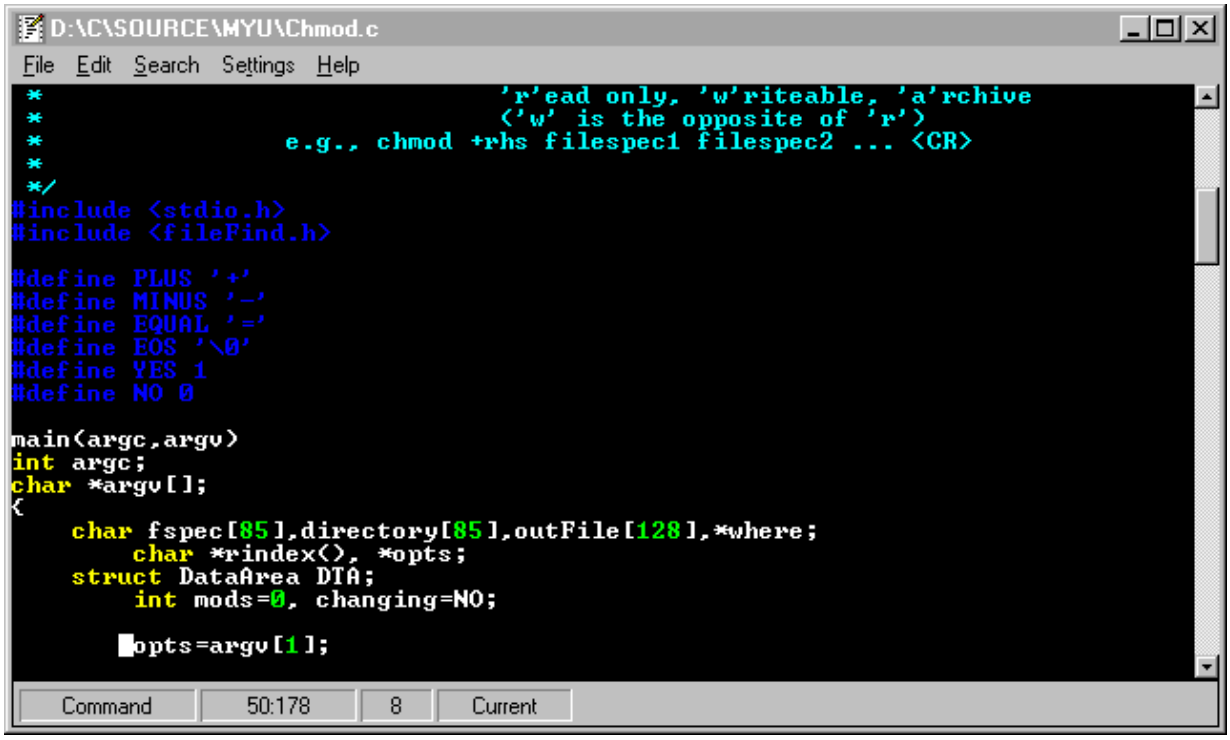
Creator: Alain Mellan mellan@axime.com [970516]

Alain is working on a Macintosh port for elvis-2.1 right now.

Lemmy

Creator: [NAME] jai@accessone.com [961004]

Title: Lemmy 1.0 with syntax highlighting



"This is a GIF of Lemmy version 1.0, with C/C++ highlighting. The menus across the top, provide options consistent with Windows applications. The status bar at the bottom tells you the edit mode, current line/total lines and current column. Lastly, whether or not your edits have been saved with a message indicating either New, Current or Modified."

Miscellaneous

TODO: Add the following pages to the appropriate section.

Title "XEmacs Viper Mode"

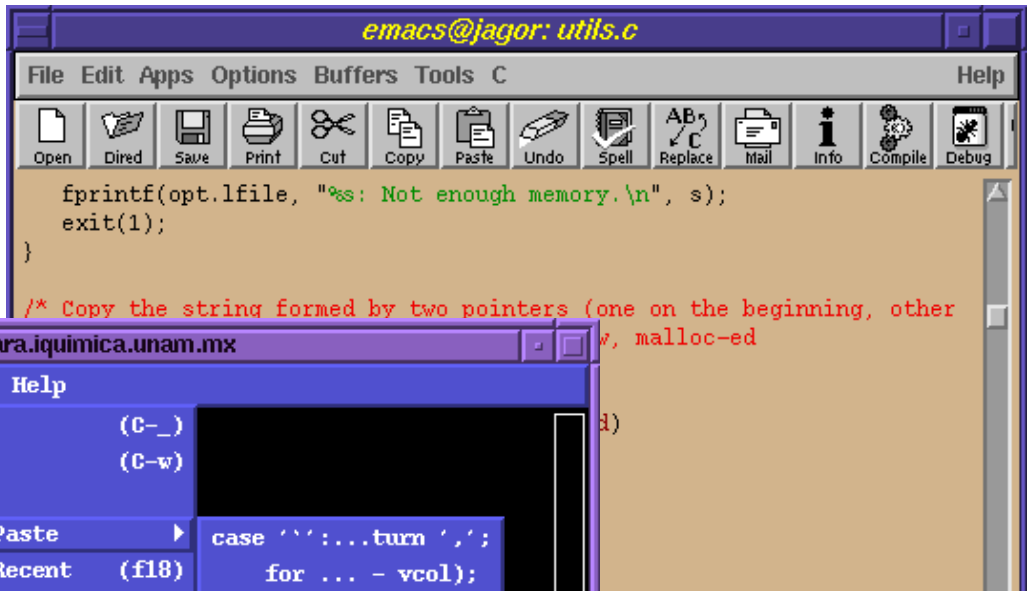
Size 14043 bytes

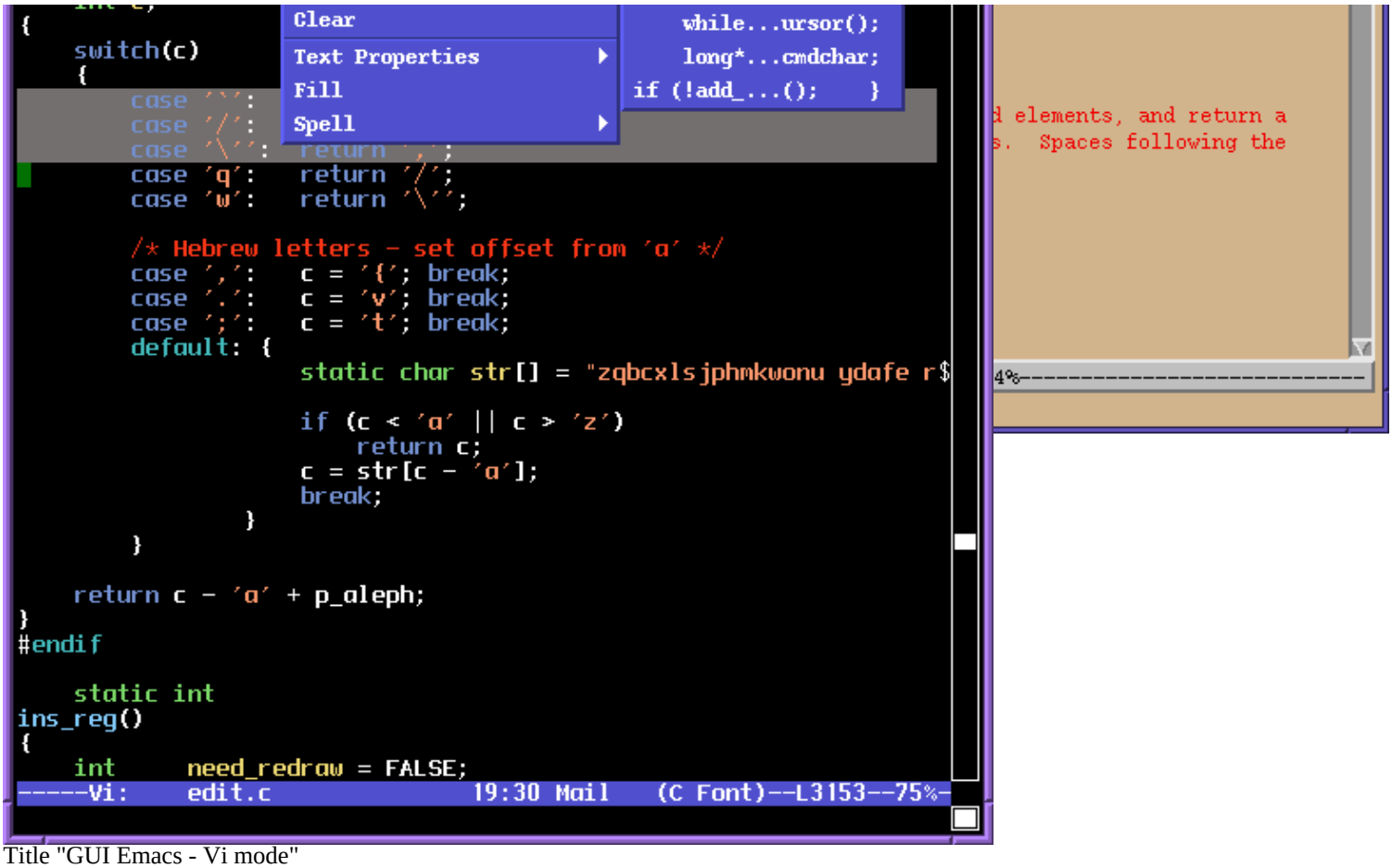
Creator Hrvoje Niksic hniksic@srce.hr

Added 961205?

Date [961126,961205]

Notes: "Viper" is a mode in XEmacs that implements Vi. Buttons and color.





Title "GUI Emacs - Vi mode"

Size 14919 bytes

Creator raul segura acevedo raul@hplara.iquimica.unam.mx

Added 970605

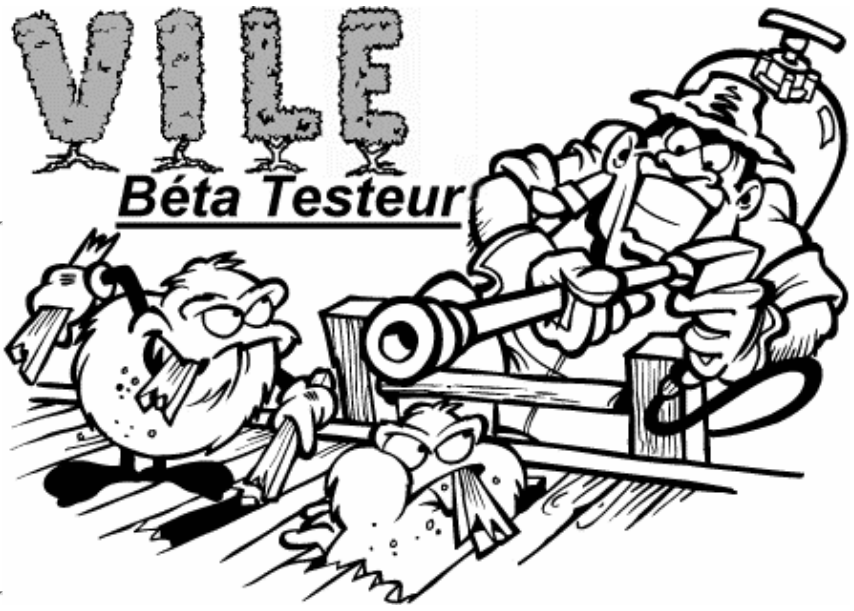
Notes: raul says: "note gui emacs is not xemacs, this one is another program. note the banner in the left bottom saying "----Vi" where normally it says "-----Emacs"."

Title "Vile Bita Testeur GIF"

Size 26293 bytes

Creator Charles Vidal vidalc@club-internet.fr http://www.chez.com/vidalc/

Added 960821 [980213]



Links to other pictures on the net

Vim on Jano's Desktop [970919]

http://www.wi.leidenuniv.nl/~jvhemert/desktopb.jpg

Author Jan van Hemert jvhemert@wi.LeidenUniv.nl

Note There's Vim on his desktop - try to spot it! ;-)

Vi Pictures - News

- 000829: Added Lovingly Handrafted
- 000726: Added Created with vi - 'Old school'
- 000609: Removed mailto links because of spam. :-/
- 990808: Added vipowered
- 990302: Added Powered by Vi (anim)
- 990105: Adeed Made With vi (anim)
- 980130: Added Vi Logo
- 980105: Added Anim Written in Vi
- 971122: Added Made With VI
- 971006: Added Designed with VILE
- 970605: Added GUI Emacs Vi
- 970416: Added MacElvis
- 970413: Added VI Powered
- 970413: Added VI - Powered by
- 970207: Added the VI editor
- 970207: Added vi - The Ubiquitious Editor
- 961126: Added XEmacs viper mode
- 961004: Added lemmy
- 961001: Added elvis

TODO - Please Help!

Add screenshots for nvi, stevie, vile, and vip. If you can provide any then please let me know!


```
URL:          http://www.math.fu-berlin.de/~guckes/vi/pics.html
Created:      Sat Jun 29 00:00:00 MET 1996
```

Send feedback on this page to
Sven Guckes guckes@vim.org

The Vi Editor and its clones

and programs with a vi like interface

```
The 'vi Lover's home
page' refers to this
website as "the Vi
site on this planet
better than the one
you're looking at."
This site too is
well worth checking
out.
```

Learning the vi Editor, 6th Edition, O'Reilly,
page 300

Vi Pages Overview

[**News**] [Answers] [Books] [Bugs] [Chart] [Clones and their Homepages] [Credits] [History] [Options] [Pics] [Quotes] [Substitution Guide] [Tree]

This Page: What *is* VI? A Definition. | Documentation on VI | Vi Clones and Versions | Links to Other Pages about Vi [such as Vi vs Emacs, Vi Folklore, Vi Stuff] and changes to this page

Feedback, please: Should the info in clones and documents be moved onto separate pages? Send me an email!

VI - A Definition

Vi is a Text Editor

[970812] What is Vi? Well, in short - Vi is an "editor", ie a program that allows to "edit" (making changes to) files. These files usually are files containing "text" (ASCII 000-127). Hence VI is referred to as a "text editor", even though you can also edit "binary files" (which include characters with the highest bit set, ie characASCII 128-255).

Vi Clones

Vi has been a standard editor on Unix systems for many years now. Now there are several programs which incorporate the command structure of this editor and which all add features that you probably have wanted with Vi all along; these are the "clones".

POSIX

There is also a definiton by the "POSIX standard", but many programs which claim to be "vi" are "clones" which more or less do standard Vi (whatever that is). Usually vi clones come with lots of (non-compatible) add-ons, but which usually are an improvement.

As ex/vi is part of the POSIX 1003.2 specification for shell utilities, you can say they are "standard" in more that just a colloquial sense.

Jargon File Definition

And then there is the "Jargon File on vi": vi: /V-I/, *not* /vi/ and *never* /siks/ n. [from `Visual Interface'] A screen editor crufted together by Bill Joy for an early {BSD} release. Became the de facto standard UNIX editor and a nearly undisputed hacker favorite outside of MIT until the rise of {EMACS} after about 1984. Tends to frustrate new users no end, as it will neither take commands while expecting input text nor vice versa, and the default setup provides no indication of which mode the editor is in (one correspondent accordingly reports that he has often heard the editor's name pronounced /vi:l/). Nevertheless it is still widely used (about half the respondents in a 1991 Usenet poll preferred it), and even EMACS fans often resort to it as a mail editor and for small editing jobs (mainly because it starts up faster than the bulkier versions of EMACS). See {holy wars}.

Development History

In article in the Linux Magazine, November 1999 Bill Joy says that vi evolved from the editor "ed", then via "em" to "ex" and "you've got to remember that I was trying to make it usable over a 300 baud modem. That's also the reason you have all these funny commands. [...] So the editor was optimized so that you could edit and feel productive when it was painting slower than you could think."

Bill Joy then added the code that changed the editor "ex" to "vi". (You can still switch from vi to ex with the command "Q", and back again with command "vi". Try it!)

Terminology

Due to the non-compatibility of the various vi clones I will try to maintain the following terminology throughout these pages and also when posting info to newsgroups, especially "comp.editors":

```
vi        the program "vi" on your current system
Vi        the "standard vi" (whatever that is ;-)
VI        any vi or clone thereof aka "all versions of vi"

Vim       Vi IMproved - a truly great vi clone with many improvements
VIM       any version of Vim
Vim-5.7   a special version of Vim (user release)
Vim-6.0w  a special version of Vim (developer release)
```

Getting started with Vi

Getting started with complex programs isn't easy, and the same holds for VI. But it might help you to get started with sample setup file. Here is "exrc" for you with some comments that you may find helpful:

Sven's exrc

Please send corrections and additional info about these pages to me at guckes@vim.org. Thanks! :-)

Vi Clones and Versions

So - which Vi clones are there? Here are some of the popular ones:

BBStevie | bedit | Bvi | calvin | e3 | elvis | elwin | javi | jvi | lemmy | levee | nvi | MKS vi [todo] | OakHill vi | PC-Vile | PVIC | stevie | trived | tvi | vigor | vile | VIM | vip | viper | vipw(?) | Watcom-VI | WinVi | xvi

FAQ:
Q: Is there a vi clone for my operating system? A: Take a look at this table [yes, lots TODO here]:

	Amiga	Atari	Mac	OS2	Unix	VAX	VMS
bbstevie	?	?	?	?	?	?	?
calvin	-	-	-	-	-	-	-
elvis	?	Atari	(Mac)	OS2	Unix	?	?
elwin	-	-	-	-	-	-	-
javi	?	?	?	?	?	?	?
jvi	?	?	?	?	?	?	?
lemmy	?	?	?	?	?	?	?
nvi	-	-	-	-	Unix	-	-
mksvi	?	?	?	?	?	?	?
oakhillvi	-	-	-	-	-	-	-
pcvile	?	?	?	?	?	?	?
stevie	?	?	?	?	?	?	?
trived	?	?	?	?	?	?	?
tvi	?	?	?	?	?	?	?
vile	?	?	?	?	?	?	?
vim	Amiga	Atari	Mac	OS2	Unix	VAX	VMS
vip	?	?	?	?	?	?	?
watcom-vi	-	-	-	-	-	-	-
WinVi	-	-	-	-	-	-	-
xvi -	-	-	-	-	-	-	-

DOS and Windows Platforms

	DOS	Win3	Win95	Win98	Win2K	WinNT
bbstevie	?	?	?	?	?	?
calvin	DOS	-	-	-	?	-
elvis	DOS	?	Win95	?	?	WinNT
elwin	DOS	?	Win95	Win98	?	WinNT
javi	?	?	?	?	?	?
jvi	?	?	?	?	?	?
lemmy	?	?	?	?	?	?
nvi	-	-	-	-	?	-
mksvi	?	?	?	?	?	?
oakhillvi	DOS	-	Win95	?	?	WinNT
pcvile	?	?	?	?	?	?
stevie	?	?	?	?	?	?
trived	?	?	?	?	?	?
tvi	?	?	?	?	?	?
vile	?	?	?	?	?	?
vim	DOS	Win3	Win95	Win98	Win2K	WinNT
vip	?	?	?	?	?	?
watcom-vi	DOS	Win3	Win95	Win98	?	WinNT
WinVI	-	Win3	Win95	Win98	Win2K	WinNT
xvi -	-	-	-	-	-	-

Summary: The clones "calvin" and "trived" run on DOS only. The clone "lemmy" runs on Windows only (I think). The clone "nvi" runs only on Unix and hasn't been updated since 1996: :ver Version 1.79 (10/23/96) The CSRG, University of

California, Berkeley.

More info on the Vi clones:

BBStevie [970926,990209,001211]

OS: DOS

Cost: ???

Author/Copyright: Roy M. Silvernail roy@scytale.com

SnailMail: ???

Home Page: http://www.scytale.com/waffle/wafiles.shtml [001217]

FAQ: none

Download: ftp://ftp.halcyon.com/pub/waffle/editors/

```
 56364 Aug 18  1992 bbstvi30.zip
160339 Aug 18  1992 bbstvsrc.zip
```

Maillist: none

Last release: BBStevie-3.0 [920818]

Comments: BBStevie is based on sources of Stevie-3.69a (or Stevie-3.68) by Tony Andrews. (Roy tried to contact Andrew about the sources but did not succeed.) BBStevie was designed as a restricted-shell vi clone for use with the Waffle BBS (and works well!) but can be used with whatever BBS or "locally". Roy does not intend to develop BBStevie any further.

From the Comp.Bbs.Waffle FAQ (cbwfaq@locutus.ofB.ORG):

```
BBStevie in particular is designed to be configurable so that you
can prevent folks from getting to the shell (if you want) and you
can use the same binary to ALLOW some other folks shell access if
you want to do that, based on their Waffle access level.  BBStevie
will also only let users mess with files in their home directory.
```

Notes:

- 001217: Homepage added.
- 001211: Roy answered an scytale.com should be active again soon.
- 990209: Contacted Roy and the maillist for more info - but the mails bounced.

bedit [971222]

OS: ???

Cost: ???

Author/Copyright: NAME MAIL

SnailMail: ??? ???

Home Page: ???

FAQ: ???

Download: ???

Maillist: ???

Last release: VERSION [DATE]

Last beta: VERSION [DATE]

Comments: [NAME EMAIL DATE]

Info by: George Young gryoung@nortel.ca

Source: http://www.winsite.com/info/pc/win3/util/bedit23.zip

Bvi [990223,990304,991001,001025,010116]

OS: DOS and Unix. Bvi-1.1 was tested on MSDOS with TurboC-2.0 and Borland-C++-3.1, as well as on these Unixes: Linux, SunOS 4.1.x, Solaris 2.x with gcc compiler, HP-UX 9, OSF/1 V2.1.

Cost: free (I suppose)

Author/Copyright: Gerhard Bürgmann Gerhard.Buergmann@altavista.net (Vienna, Austria)

Home Page: http://bvi.sourceforge.net/ [010116]

http://bvi.linuxave.net [obsolete]

Download: http://bvi.sourceforge.net/download.html

Improved manual: http://www.math.fu-berlin.de/~guckes/bvi/manual.html

Mailing List: http://www.egroups.com/group/bvi

Latest version: See BVI info on AppWatch.com

Comments: Bvi is actually not a quite a vi clone as it does not deal with "texts" but "binaries". From the announcement: "Bvi is a display-oriented editor for binary files based on the vi texteditor. It uses commands similar to the commands of the vi, with some changes dependent of their different tasks."

calvin [970926,980918,001206]

OS: DOS only.

Cost: free

Author/Copyright: Paul Vojta vojta@math.berkeley.edu

Contact: Nan-shan Chen chen@get.uni-paderborn.de

SnailMail:

```
Nan-shan Chen, Pohlweg 47-49, 33098, FB14, Uni-Paderborn  [970404]
```

Home Page: http://www.math.fu-berlin.de/~guckes/calvin/

FAQ: (none yet)

Latest version: calvin-2.3

Download: http://www.simtel.net/pub/simtelnet/msdos/editor/calvin23.zip

Local copy: calvin23.zip 38801 bytes [971222]

Notes: The author has stopped further development. But a fan mailing list has formed - and maybe the development might be taken on by them.

Comment: Calvin was formerly known as "Free VI".

e3 [010607]

OS: Unix (esp. Linux on x86, FreeBSD, BeOS)

Cost: none

Author/Copyright: Albrecht Kleine kleine@ak.sax.de

Home Page: http://www.sax.de/~adlibit/

Download: http://www.sax.de/~adlibit/e3-1.6.tar.gz

Maillist: ADDRESS

Last release: e3 1.6 [yymmdd]

Comments: HomePage lists other programs, too. No download address (ftp), no mailinglist.

Description on webpage: "e3 (80k) is a full featured text editor in two versions: primary e3 for x86 Linux, FreeBSD and BeOS(TM) **written in NASM assembler**, and for second e3c, an experimental C equivalent for all other platforms. The assembler version is highly optimized for size. For the command syntax you can choice between the families of Wordstar(TM), EMACS, Pico, nedit or vi editors. In Linux the e3 uncompressed executable's size is at **12000** byte, a compressed executable will need around 8800 byte, so you won't waste your disk space ;-) e3 is **quite independent of libc** and because of it's size it is very useable for Mini-Linux distributions and rescue disks. Some features like piping through /bin/sed (using stream editor as sub a process) are currently designed for Linux only, anyway this opens e3's door to the world of regular expressions."

Elvis [970926,990128]

OS: DOS, Win95, WinNT, OS/2, and Unix [BSD UNIX, AT&T SysV UNIX, SCO Xenix, Minix] Atari TOS, OS9/68000, and Coherent. Macintosh version coming up!

Cost: Freeware!

Author/Copyright: Steve Kirkendall kirkenda@cs.pdx.edu

SnailMail:

```
Steve Kirkendall
14407 SW Teal Blvd. Apt. C
Beaverton, OR 97005
(503) 643-6980
```

Home Page: http://www.fh-wedel.de/elvis/ [981208] by herbert@paulina.shnet.org

Download:

Download Elvis Releases:

ftp://ftp.cs.pdx.edu/pub/elvis/

Download Elvis Beta:

ftp://ftp.cs.pdx.edu/pub/elvis/unreleased/README.html

ftp://ftp.false.com/pub/elvis/unreleased/README.html (mirror)

Last release: VERSION [DATE]

Last beta: elvis-2.1_4 [991210]

```
* The OS/2 port has been merged into 2.1i.  Binaries are available.
* Error messages can be logged to a file.
* Problems with regular expression character lists have been fixed.
```

Comments: [Steve Kirkendall kirkenda@cs.pdx.edu 971002] "elvis-2.0 has a *real* X interface which uses the mouse a lot better. Also, if you force it to run inside an xterm it won't attempt to intercept mouse events. [...] As compared to 2.0, 2.1e-alpha adds support for overloaded tags (useful for C++ programmers), a configurable toolbar under X, and the ability to read via HTTP, and read/write via FTP. Plus bug fixes and a lot of other little changes, of course. [...] elvis is updated about once every 4-6 weeks."

Elvis is the standard vi clone on Linux Slackware distribution.

JaeSub Hong's flame@cuphy3.phys.columbia.edu page on Elvis:

http://www.phys.columbia.edu/~flame/elvis/ [981019]

Screenshots and some helpful info.

elwin [970919,000113,001127]

OS: Windows (95/98/NT)

Cost: shareware (trial period: 30 days), $50

Author/Copyright:

```
Little Wing
4618 JFK Boulevard, Suite 176
North Little Rock, AR 72116
TEL (501) 771-2408
71532.403@compuserve.com
```

Home Page: http://www.little-wing.com/

Latest releases:

```
elwin-2.0.7 [.....] (16bit)
elwin-2.1.1 [.....] (32bit)
```

Missing commands: ":map" - you cannot "map" keys! :-(":rewind" is not possible, either. ":edit #", ":next", ":set all" Cursor is placed between characters and some commands have been adjusted to that (such as 'fFtT'). Character exchange/transposition with "xp" does not work therefore.

Incompatibilities: Commands "[[" and "]]" are replaced by '[' and ']'. Incomplete pattern matching. Knows no tags.

Elwin can almost be classified as "nagware" - on every startup and every exit (of the main program) an extra window pops up exhorting you to spend $50 on Elwin.

Someone told me on 001127:

Elwin can also be run in a "Windows" (non-Vi) mode, where the mouse can be used to select text like Windows Notepad. A mouse pick in the text window puts it into Windows mode, and the Escape key puts it into Vi mode. The 'About' window for version 2.1.1 lists "Copyright 1995-1997", so one could guess 1997 is the release date for that version. Implemented options: autoindent, directory, ignorecase, inputmode, magic, remap, scroll, sidescroll, tabstop, wrapscan. Not implemented options: list, number, readonly, report, shell, taglength.

001127: Elwin seems to have a homepage now - good. However, it just gives two links to the latest versions, and tells you that Elwin is shareware. That's it. so if any of the comments are not valid any more then please send me a mail!

javi [971009,990820]

Author/Copyright: James "Jim" Jensen jjensen@emi.net

SnailMail:

```
james jensen
p.o. box 811452
boca raton, fl 33481
TEL +1 407 362 7997
```

Home Page: none http://www.emi.net/~jjensen/java/javi.html [defunct?]

Last release: 960814

jVi [010108]

OS: ???

Cost: ???

Author/Copyright: Mozilla Public License

SnailMail: ???

Home Page: http://jvi.sourceforge.net/

FAQ: URL

Download: http://jvi.sourceforge.net/jvi-main.html#Download

Last release: VERSION [DATE]

Last beta: VERSION [DATE]

Comments: jVi is an IDE running in Java - and it is based on Vim (Vi Improved). -- "Vi is a vi-vim editor clone built on top of the javax.swing.text package. Available embeddings: JBuilder. jVi is designed to quickly and easily integrate into many disparate java based desktop apps, from IDE's to mail-news readers." -- "jVi is patterned after vim. [...] jVi is a small subset of vim. Some source code in jVi is taken from vim and modified to work in the java environment. The user documentation is almost exclusively from vim, only modified to remove things that are not applicable."

Lemmy [970914,970926,000113,001121]

OS: Windows 32bit (Windows95, Windows98, WindowsNT)

Cost: Shareware - 30 day trial.

```
Pricing
   Prices for Lemmy v4.2 are as follows:
     * Single-user:       $20
     * 2 to 9 users:      $15 each
     * 10 to 24 users:    $12 each
     * 25 to 49 users:    $9.50 each
     * 50 to 99 users:    $7 each
     * 100+ users:        $5 each
An unlimited site-license costs $500.
```

[Benefits of registration include: Free product upgrades. Removal of all nag screens.]

Distributor: SoftwareOnline

Lemmy is now owned by SoftwareOnline:

Home Page/Announcement:

http://www.softwareonline.org/lemmy42.html

This site has links to the Frequently asked questions, Sample VB Scripts, and Additional Syntax Highlighters.

FAQ: http://www.softwareonline.org/lemmyfaq.htm

Download: http://www.softwareonline.org/lemmy30.exe

Lastest release: Lemmy-4.2 [??????]

OLD URLs:

```
http://www.accessone.com/~jai/
http://www.accessone.com/~jai/faq.html
ftp://ftp.accessone.com/pub/misc/jai/
```

http://www.drizzle.com/~james/lemmy.html

Original Author:

```
James Iuliano james@drizzle.com
2442 NW Market Street, #501
Seattle, WA     98107
```

Comments by David Douthitt (DDOUTHITT@cuna.com):

Lemmy's help is nice. Lemmy's help files use the Windows 95/NT interface. Lemmy has support for digraphs. Syntax highlighters use a special interface to Lemmy, and highlight varying pieces of code based on the token found (such as keywords, etc.) Included syntax highlighters (third party) appear to include Tcl/Tk, BAT/CMD, Perl, and Tex/LaTex. Lemmy supports OLE Active Scripting, including special commands for Java Script and VBScript - PerlScript seems to be theoretically possible. Different filetypes can be defined (with extensions/file patterns), each of which has its own syntax highlighting, and supplemental ".exrc" file which runs before the file is edited, as well as a supplemental epilog file which run after the file is edited. Lemmy also has an optional "ksh-style" ex prompt, which basically means that the ex prompt supports the ksh vi-history mode. Lemmy appears to include HTML syntax highlighting as well. This is a very nice feature which makes Lemmy stand out among other vi clones. The specific colors used for each syntax element are also configurable. Lemmy supports movement through the file using the scroll bars at the right window side; however, Lemmy maintains cursor position only as long as it is visible; as soon as the position would go off screen top/bottom, the position was set to the left-most end of the line, at the top or bottom as appropriate, and stayed there until the cursor was moved in the opposite direction, at which time it retained its position. Lemmy supports mouse movement and control, including selecting text and others. Lemmy has a strong set of help manuals for those who want to implement syntax highlighting plug-ins; the entire API is described in full detail, including C function calls and descriptions. Lemmy has an FTP "browser" or similar tool included. [The FTP access requires WININET.DLL, which NT doesn't seem to have, though.] Lemmy supports "drag-and-drop", opening with a file dropped upon its icon or an open window. Lemmy supports tags. [vi compatible?] Lemmy has support for printing, with syntax highlighting (not too impressive, it seems) and headers and footers and line-numbers (all optional), with a user-selected font for the printout. All or part of the file can be printed. Lemmy has a status bar underneath, including position, "mode" (like mark, command, insert, etc.), buffer ("a", "b", etc.) and more.

Minuses:

```
Lemmy seems to be for generic MS-Windows platforms only.
Lemmy's help is missing specifics on how exactly it differs from standard vi.
Lemmy's search functions seem to be slow.
Lemmy is not free (but shareware).
```

000113: No update since 1997. Asking $20 for a program which does not seem to be maintained is quite a lot...

levee [990304]

OS: DOS.

Cost: Free?

Author/Copyright: David L. Parsons (orc)

Download: levee.zip 33K

HomePage: http://www.pell.portland.or.us/~orc/Code/

Comments from the author: "This is my attempt to write a vi clone. This was initially written in the early 1980s in USCD Pascal, then I rewrote it into P2, then my friend John Tainter and I converted it into C for the Atari ST. After I

started running Linux, I abandoned it for the already written vi clones, and then for Berkeley's nvi, but it is still good clone of the visual mode part of vi, and at approximately 37k, it's hard to beat Levee's disk size. Levee is the standard vi for the core component of Mastodon."

Limitations: "Levee can only edit files up to 32670 characters long. CTRL-M is used as its internal line separator, so inserting CTRL-M will have interesting consequences."

nvi [990101,990713,001022]

OS: Unix

Cost: "Free" - BSD license

Maintainers:

```
Keith Bostic     bostic@bostic.com
Sven Verdoolaege  skimo@kotnet.org
Steve Greenland stevegr@debian.org (for Debian)
```

Home Page: http://www.bostic.com/vi/ [990713]

Last release: nvi-1.79 [961023?,980330]

Latest version: nvi-1.81 [000902]

Development site: http://www.gv.kotnet.org/~skimo/nvi/devel/

Comments: NVI has been the standard vi editor on BSD systems for many years. However, the homepage does not give info about the version history and release dates.

Oak Hill vi [970926,980113,980120]

OS: DOS, WindowsNT

Cost: Shareware $35

Author/Copyright: (c) 1992 David G. Leeper oakhill@iname.com dleeper@mail.com

SnailMail:

```
David G. Leeper
Oak Hill Software, Inc (member ASP)
8603 E Corrine Dr
Scottsdale, AZ 85260
602-998-4974
```

Support Email: oakhill@iname.com

Home Page: http://home.att.net/~leeperd/ohvimain.html

FAQ: URL

Download: http://home.att.net/~leeperd/dwnld01.html

Last release: "Oak Hill Vi 7.0A" [980111]

Last beta: VERSION [DATE]

INFO by: David Douthitt DDOUTHITT@cuna.com

PVIC [980504,980511,000713,001031]

OS: DOS, OS9

Cost: PVIC is public domain.

Author/Copyright: Frits Wiarda fwiarda@fwiarda.com

SnailMail:

```
Frits Wiarda
Boulevard Heuvelink 1-5
6828 KG  Arnhem
Holland
```

Home Page: http://www.fwiarda.com/software.htm#pvic

http://ourworld.compuserve.com/homepages/fwiarda/software.htm#pvic [obsolete]

FAQ: none

Maillist: none

Last release: Pvic [980424] (date stamp on binary)

Last beta: Frits Wiarda doesn't have time to maintain PVIC - so no further development is planned for now.

Comments: Frits Wiarda says on the homepage: "PVIC is a Portable VI Clone derived from STEVIE 3.69B ... and did contain many non portable things like BIOS calls. I have removed everything [...] I considered non-portable, and I did rewrite the I/O. I added code to read termcap files. I gave many variables and functions [...] more understandable names. It has been written in Kernighan and Ritchie C, so it compiles on both old and modern C compilers."

[980509] Frits Wiarda : "All information I have about PVIC is in the .ZIP file one can download. Documentation I do not have, but there are plenty of comments in the source code." He also says that he wrote PVIC some five years ago when he was in need for a vi clone for OS-9.

trived [970927,990223]

OS: DOS only.

Cost: Free

Author/Copyright: Russell Schulz Russell_Schulz@locutus.ofB.ORG

SnailMail: N/A

Home Page: none?

FAQ: none

Download: ftp://ftp.simtel.net/simtelnet/msdos/editor/trived09.zip

Maillist: none

Last release: trived-0.9 [950425]

Comments: Trived is NOT a vi clone, but has a small vi subset. Trived has on-screen help like pico and has automatic uuencoding of binary files (with command ":r"). The archive includes the Pascal source. Trived thus probably is the only vi-like editor in Pascal. [slightly reworded statement by Russell Schulz]

Comments: The archive comes with an executable and Pascal source. The editor is designed as a vi clone with small memory requirements for use locally or on a BBS. The name of the archive is "TRIVED*.ZIP". [David Douthitt DDOUTHITT@cuna.com 970926]

tvi [970926]

OS: Windows 3.1 The editor "tvi" emulates vi with a subset of vi commands.

Cost: Shareware - US:$10, Non-US:$20

Author/Copyright:

```
Michael Edelstein
EMAIL?
```

SnailMail:

```
Michael Edelstein
3276 Ashford Street #D
San Diego, CA 92111
```

Home Page: URL

FAQ: URL

Download: URL

Last release: VERSION [DATE]

Last beta: VERSION [DATE]

Comments: The tvi editor does not use the Windows environment to any advantage, and does not appear to implement as much of vi as other clones do. Definitely a minimalist version. Not sure why you would want to use this software when you can register Lemmy or Elwin for just a little bit more or get VIM or xvi for nothing at all. [David Douthitt DDOUTHITT@cuna.com 970926]

vigor [000225]

[Vigor Logo]

OS: ???

Cost: ???

Author/Copyright: Joel Ray "Piquan" Holveck Joelh@gnu.org

Home Page: http://www.red-bean.com/~joelh/vigor/

Download: http://www.red-bean.com:8080/vigor-0.014.tar.gz

Last release: vigor-0.014 [000225,000321]

Comments: Take a look at the Vigor MPEG! funfunfun

Vigor is "application of the week" [000317] on LinuxCare:

http://www.linuxcare.com/viewpoints/ap-of-the-wk/03-17-00.epl

Author: Brett Neely

Comments: Vigor is based on nvi and was written in C and Tcl/Tk.

vile [970903,980624,000713,001008]

OS: MS-DOS (DJGPP), UNIX and equivalent (termcap/terminfo, X Window), VAX/VMS, OS/2 (both native and EMX), Windows95, WindowsNT (console and GUI, with Visual C/C++ 4.1).

Cost: "Freely Distributable, copyright statements must be maintained"

Author/Copyright:

Main programmer: Paul G. Fox pgf@foxharp.boston.ma.us pgf@cayman.com .

Thomas E. Dickey dickey@clark.net http://www.clark.net/pub/dickey >http://www.clark.net/pub/dickey/

Kevin Buettner kev@primenet.com

Rick Sladkey sladkey@world.std.com (?)

Home Page: http://dickey.his.com/vile/vile.html

http://www.clark.net/pub/dickey/vile/vile.html [obsolete]

Download: ftp://dickey.his.com/vile/

ftp://ftp.clark.net/pub/dickey/vile/ [obsolete]

Maillists:

```
vile-announce-request@foxharp.boston.ma.us announcements
    vile-code-request@foxharp.boston.ma.us developers
   vile-users-request@foxharp.boston.ma.us users
```

Last release: vile-9.1y [001005]

Comments:

PC-Vile was created from the source of micro-emacs(!) - programmed by a lot of people. Paul Fox pgf@cayman.com is NOT the Paul Fox who did the CRiSP editor... [David Douthitt DDOUTHITT@cuna.com 970926]

There is also a maillist for bug reports which does not require subscription - but Paul Fox asked me to remove this address for now due to spams. :-/

The official homepage does not show the latest version. This *sucks*. Therefore

William Totten's vile page

http://www.cis.udel.edu/~totten/vile/

Alex Wetmore's cfilt.pl

http://www.phred.org/~alex/vile/cfilt.pl

Colorizes the Perl hooks within Vile.

The Freshmeat vile appindex record

http://appindex.freshmeat.net/view/901503901/

Linux Gazette story on vile http://www.redhat.com/linux-info/lg/issue01to08/xvile_mar96.html

A page about Vile in Czech

http://www.cestina.cz/cestina/pocestovani/unix/Vile/

vim (aka "Vi IMproved") [970919,980624]

OS: "All" (well, almost ;-)

AmigaOS, AtariMiNT, BeOS, DOS, MacOS, OS/2, RiscOS, VMS, WinNT+Win95, and these flavors of Unix: A/UX, AIX, DG/UX, DEC Unix, FreeBSD, HP-UX, Irix, Linux, NetBSD, QNX, SCO, Sinix, Solaris, SunOS, SUPER-UX, Ultrix, Unisys.

Cost: "Free". Actually it is "CharityWare", ie the author encourages making a contribution to charity.

Author/Copyright: Bram Moolenaar bram@vim.org

SnailMail/Phone:

```
Bram Moolenaar
Clematisstraat 30
5925 BE Venlo
The Netherlands
Tel: +31 77 3872340
```

Home Page: http://www.vim.org/ (maintained by Sven Guckes guckes@vim.org

FAQ: http://www.vim.org/faq/

Download: http://www.vim.org/dist.html

Latest versions: See the Vim History Page or the Vim HomePage.

Mailing Lists: http://www.vim.org/mail.html - four maillists: announcements, general questions, general development, development of MacOS port.

Description: Vim (Vi IMproved) is an almost compatible version of the UNIX editor vi. Almost, because in many ways it chooses to improve it and do away with bugs. If you know vi then you will only want to know the differences which are described in the vim helpfile "vim_diff.txt" and on the page about Why use Vim?.

Note: VIM is now the standard vi clone on the Linux RedHat distribution.

Watcom-VI [990625,990701]

OS: DOS, Win3, Win95, Win98?, WinNT

Cost: Watcom-VI shipped with the Watcom Compiler.

Author/Copyright:

```
"Copyright by WATCOM International Corp.  1991-1994"
```

Last release:

```
:ver
"vi/nt" v1.0  12:24:27 Aug 16 1994
```

Comments: "The NT versions run in both the GUI or as a console mode app. The Dos version works great in a DOS box but unfortunately only supports short file names." "The Watcom VI also has scripting support and a text-based GUI which can be modified by the user and compiled into the .EXE file. You can change key commands, menus, scripts, etc" Info by Michael S. Leibow mleibow@eteklabs.com [990625, 990629]

Known Bugs: Some commands operate incorrectly as the calculation on the operated text expanded tabs but forgets that tabs are just a single character.

WinVi [980407,990729,001024]

OS: Windows 32bit (Windows95/98/00/NT)

Cost: "Freeware"

Author/Copyright:

```
German:         Raphael Molle   ramo@winvi.de [preferred]
                                ramo@berlin.snafu.de ©1994-2000
French:         Valerie Gunslay
                Yves Belanger
Spanish:        Jose Maria Romero
```

SnailMail:

```
Raphael Molle
Wormser Str. 5
10789 Berlin
Germany
```

Home Page: http://www.winvi.de http://www.snafu.de/~ramo/

Last releases:

```
------  WinVi-2.47      980222  first release
16bit   WinVi-2.73      991103
32bit   WinVi-2.83h     000402
```

Comments: WinVi-2.73 is 320K in size, has filename completion and a hex mode, and dialog boxes for general settings and colors; however, no documentation, apart from a ChangeLog. -- Quote from the home page: "This editor is especially useful for friends of the Vi editor (strange human beings, supposed to be descended from the world of Unix) who do not want to give up the little conveniences offered by Windows."

viper [970101,990223]

Viper is a standard package with Emacs. Viper was formerly known as VIP-19, which was a descendant of VIP 3.5 by Masahiko Sato and VIP 4.4 by Aamod Sane.

You can rely on Viper being present on GNUEmacs-19 and XEmacs-19. (Because of its reliance on minor mode keymaps, Viper will not work under Emacs 18.)

> emacsclient doesn't make emacs multiuser. It just obviates the need to load up emacs for every file to be edited, and eliminates load times. However, on any decent OS the executable code of emacs is shared amongst all invocations (is this true for executables that contain dumped bytecode?). If the users constantly switch in and out of vi, such sharing cannot be utilised. -- Vladimir Alexiev (vladimir@cs.ualberta.ca) [960630]

Here's an online version of a readme about Viper:

http://diglib.hab.de/doc/susehilf/gnu/viper/Top.html [010220]

"(Viper)Improvements over Vi": [010326]
http://www.astro.cf.ac.uk/cgi-bin/info2www?(Viper)Improvements%20over%20Vi

Send bug reports to kifer@cs.emacs.edu , preferably using the command `:submitReport` for this.

xvi [970516]

Author: B. Sartirana

This program implements a screen oriented hexadecimal/octal editor whose commands are a subset of those of "vi".

This program was submitted to the vi archive on "22 Jan 92".

Home Page:

http://tinker.winsite.com/info/pc/winnt/misc/xvi.zip/

This page has a listing of the xvi.zip and a link to it. That's it. I briefly took a look at the archive and I found this info:

Latest build: 940602 (2nd June 1994)

```
323584  06-02-94  13:35   xvi/xvi.exe
```

The copyright notice is given by "Chris and John Downey" in October 1992. The file "readme" says: "This is a source release of the Xvi editor (derived from "STEVIE"), a clone of the UNIX editor `vi'. The program was originally developed for the Atari ST, but has been ported to UNIX, MS-DOS, OS/2 and QNX as well."

Info by: Avram Dorfman dorfmana@comm.hq.af.mil

On 990713 tethys@it.newsint.co.uk added: "Chris Downey was one of my lecturers at University. John is his brother. .. Don't know the current status of development, but since I haven't heard anything since the 1992 release, I guess it's probably been abandoned. Last I heard, Chris was working for Pfizer, and should be contactable as cmd@pfizer.com, or if that doesn't work, try Chris_M_Downey@sandwich.pfizer.com ."

Alternative Download: [000317]

ftp://ftp.cs.tu-berlin.de/pub/doc/vi/vi.xvi.tar.Z [37,893 bytes]

I tried to make it with gcc (to be precice: egcs-2.91.57 19980901, egcs-1.1 release) but it failed. No time to debug it,

though.

VI Documentation

There are quite a number of documents about Vi. (I will add the URLs asap.) However, you may not yet know about these:

Online Documentation (HTML) - outbound

The VI Lovers Home Page [990611]

http://www.thomer.com/thomer/vi/vi.html

Mirror:

http://www.cs.vu.nl/%7Etmgil/vi.html

A dedicated fan of Vi who set up a dedicated page which has become quite popular. (I guess this page has a good title :-).

Author: Thomer M. Gil tmgil@cs.vu.nl

German Vi Lovers Home Page [990223]

http://linux.hs-bremerhaven.de/~alfred/vi/

Basically another VI Lovers Homepage - but specifically for Krauts. The main difference is the additional page on Tips and Tricks:

http://linux.hs-bremerhaven.de/~alfred/vi/tips.html

Author: Alfred Schmidt alfred@HS-Bremerhaven.DE

GO Network: Vi Text Editor [000901]

Search > Technology > Software > By type > Text editors > Vi

http://www.go.com/WebDir/Vi_text_editor?lk=noframes&svx=related

No link to this page. :-(

MetaSpider [001229]

http://search.metaspider.com/Meta/Spider?vi

Yahoo's page about Vi:

Home > Computers and Internet > Software > Text Editors

http://www.yahoo.com/text/Computers_and_Internet/Software/Text_Editors/vi/

Mostly defunct links. :-(Yahoo should check those links more often!

First Steps: VI [010307]

http://www.infobound.com/vi.html

Intro, Commands, Practise Lesson #1, Startup File, Practise Lesson #2, Editing Multiple Files, Using Named Buffers, Macros.

Author: Tony Porczyk tony@infobound.com

"*Just enough vi editor commands to survive*" [000801]

http://www.cs.unt.edu/~chapin/1120/vi_hints.txt

Author: Brent Smith simpleeqbest@hotmail.com

This text is meant to be "as small and basic as possible".

"*Vi for New Users*" [000613]

http://www.dotcomma.org/programming/view.php?id=37

Article in "Printer Friendly Format":

http://www.dotcomma.org/programming/pff.php?id=37

Author: Eric "paradox" Sun eric@dotcomma.org http://www.dotcomma.org/

(Yet another) "*Vi Intro*" [000223]

http://csel.cs.colorado.edu/unix/viintro.html

Author: unknown

VI(Visual) Editor Reference manual [991201]

http://www.cis.udel.edu/~totten/vi/

Author: William Totten totten@pobox.com [990310]

Vi Reference

http://www.cs.wustl.edu/~jxh/vi.html

The HTML version of the Vi Reference once compiled by Maarten Litmaath maart@nat.vu.nl - with tables.

The plaintext version is also available:

http://www.cs.wustl.edu/~jxh/vi.reference

Author: James Hu jxh@cs.wustl.edu

Vi chart [970123,971121]

A chart with Vi commands for several vi clones - ELVIS, NVI, and VIM. Includes equivalent or similar commands for the editors SPF and TSO, too.

The linked page uses a "table" - but there is a link to a plain-text page, too. So you can print out that text easily.

Author: Jeff Wang jeffw@jeffw.com jeffw@advance.com

Old versions of this chart (local copies):

doc/vi.chart (84K) [960916]

doc/vi.chart.gz (22K) (same version compressed with gzip)

"*Vi for Smarties*" - a Vi Tutorial [980528,980721,991129]

http://www.geocities.com/ResearchTriangle/7584/vi/

An intro for newbies. Five lessons and a little quiz. Nice!

Author: Jerry Wang dimension10@geocities.com

VI Reference Manual & VI Introduction [980723,981109]

http://www.rru.com/~meo/useful/vi/

http://www.rru.com/~meo/vi/ [soon?]

Authors: Miles O'Neal meo@netads.com and Susan Liebeskind shl@cc.gatech.edu

[English] *vi reference* [970212,990915]

http://www.splange.freeserve.co.uk/viref.html

http://www.geocities.com/Athens/2694/viref.html [old]

"... compact and usable listing of command and insert mode keys, ex commands and options."

HTML by: John Arundel john-arundel@psion.com

[English] *Vi/Ex Editor* (HTML) (UnixWorld) [970127,980114,010102]

http://www.networkcomputing.com/unixworld/tutorial/009/009.html

http://www.wcmh.com/uworld/archives/95/tutorial/009/009.html [obsolete]

http://www.wcmh.com/uworld/archives/95/tutorial/009/009.html [obsolete]

"Test Your Vi/Ex Knowledge"

http://www.unixworld.com/uworld/rs/contests/vi/ [obsolete?]

Contact: editor@unixworld.com

Author: Walter Alan Zintz

[English] *Vi FAQ* (Part1) (HTML) [961029,000314]

http://www.macom.co.il/vi/

The Vi FAQ - HTML version.

HTML by Baruch Promislow baruch@macom.co.il

VI FAQ Part1: http://roger.ecn.purdue.edu/~kompella/html/vi_faq1.html

[English] *VI FAQ* (Part2) http://roger.ecn.purdue.edu/~kompella/html/vi_faq1.html

Mirrors(?):

http://roxanne.roxanne.org/~eric/vi_editor/

[English] *Vi Tutorial* (HTML) (Purdue) [961029]

http://ups.ecn.purdue.edu/ecn/Documents/VI/

A Vi tutorial with pictures.

Author: marian@ecn.purdue.edu

[English] *Using the Vi Editor* (HTML) [970128]

http://unixhelp.ed.ac.uk/vi/

This is a part of the UNIXHELP project.

Maintenance: unixhelp@ed.ac.uk

I wonder whether anyone still maintains this project. The pages look pretty dead. :-([961029]

Mirrors:

http://www.nova.edu/Inter-Links/UNIXhelp/vi/index.html

Vi and Programming

Introduction to Programming in C/C++ with Vim [010615]

http://www.linuxnewbie.org/nhf/intel/programming/intro_c++.html

Author: Kmj kmj9907@cs.rit.edu

Vi Documentation in Finnish

[Finnish] *Perusohje vi-editorin käyttöön* [000918]

http://www.helsinki.fi/%7Erista/tekstit/vi-opas.html

Basic instruction for using of vi - a Vi start guide written in Finnish.

Author: Aapo Rista Aapo.Rista@iki.fi

[Finnish] *Vi* [000922]

http://www.hit.fi/%7Elehtonen/Unix/unix26.htm

Author: Kari Lehtonen KariL@iki.fi (1997) http://www.hit.fi/%7Elehtonen/

[Finnish] [Text] *VI-EDITORI* [000922]

http://www.cs.joensuu.fi/pages/amanuenssi/computing_facilities/Ohjelmat-CS/vi/vi.html

Author: ??? Last updated: October 1996

Vi Documentation in French

[960822]

[French] *Vi Guide* (PostScript, gzip)

doc/vi.guide.french.tex.ps.gz (35K -> 85K)

A Vi starter guide written in French.

Author: Charles Vidal charles@cln46fw.der.edf.fr

Vi Documentation in Deutsch/German

VI-Einfuehrung und Kurzreferenz [981102]

[Deutsch,German] [HTML]

http://www.fh-wedel.de/~herbert/html/vi/

A short intro to editing with Vi - in German.

Author: Martin "Herbert" Dietze herbert@fh-wedel.de

Der vi-Editor - ein mächtiger Zwerg (German) [000901]

[Deutsch,German] [HTML]

http://lug-dd.schlittermann.de/vortraege/vi-report_html/

Author: Marcus Obst mobst@apfel.sax.de [000115,010312]

Slides (in MagicPoint) for a talk about vi - in German. created for the Linux User Group in Dresden, Germany.

Textverarbeitung mit dem Editor vi" [990223]

[Deutsch,German] [HTML]

http://www.delix.de/Linux/handbuch/node140.html

A description of the editor vi - in German. Last updated in November 1995.

Author: Delix sysadmins?!

Editor vi [990223]

[Deutsch,German] [HTML]

http://www.we.fh-osnabrueck.de/fbwe/vorlesung/edv1/unix/node18.html

A description of the editor vi - in German. Last updated in December 1995.

Vi Documentation in Greek

Vi explained [961029]

[Greek] [HTML]

http://www.edu.physics.uch.gr/~danalis/manuals/vi/vi_front.html

If you prefer to read about Vi in Greek - there you are!

Author: Anthony G. Danalis danalis@edu.physics.uch.gr http://www.edu.physics.uch.gr/~danalis/

Vi Documentation in Swedish

Vi Tutorial [991208]

[Swedish] [HTML]

http://www.mds.mdh.se/support/howto/vi.html

Author: support@mds.mdh.se

[971009]

[English] *Vi Macros, Abbreviations, and Buffers*

http://soma.npa.uiuc.edu/docs/vi.macros

Author: Fred Buck (1988), additions by Maarten Litmaath maart@cs.vu.nl (1989)

The escape filter, test abbreviation, keystroke remapping (test mode, command mode), text-buffer execution; mode-bouncing, chained macros, recursive macros, termination of recursives macros, peculiar limitations and restrictions on 'vi' macros; putting and yanking to/from named buffers; remappable keys (Which keys to remap?).

An Extremely Quick and Simple Introduction to the Vi Text Editor [991112]

http://heather.cs.ucdavis.edu/~matloff/UnixAndC/Editors/ViIntro.html

Author: Norm Matloff matloff@heather.cs.ucdavis.edu

Per's Vi Tutorial [980114]

http://www.dina.kvl.dk/~abraham/religion/vi-tutorial.html

Author: Per Abrahamsen abraham@iesd.auc.dk

Rather than a Vi Tutorial this shows how hard learning Vi commands can be. Or maybe it is to show why people have written vi clones that improve the movement of the cursor.

Viper Tutorial [990223]

http://linux01.gwdg.de/susehilf/gnu/viper/Top.html

"Viper is a Vi emulation package for GNU Emacs 19 and XEmacs 19."

Online Documentation (non-HTML) - local

An Introduction to Display Editing with Vi [971008]

http://www.de.freebsd.org/de/doc/usd/12.vi/paper.html

Authors: William Joy and Mark Horton

The text about Vi - written by the authors.

Online Documentation (non HTML) - outbound

BSD manuals (ex and vi) (PostScript) [960615]

ftp://gatekeeper.dec.com/pub/BSD/manuals/

Directory:

```
98069 Jul  2  1992 usd.15.vi.ps.Z
56681 Jul  2  1992 usd.16.ex.ps.Z
```

The "usd.16.ex.ps.Z" should contain the "Ex Reference Manual - Version 3.7" by William Joy and Mark Horton. There are 19 pages in total. Please correct me if I am wrong!

The "true" story of vi [960101,990223]

http://caad.arch.ethz.ch/~neil/Jokes_and_Fun/Vince_Idiot.html

http://www.sentex.net/~rboys/vince.html [defunct]

A fun story about Vi.

Addicted to Vi [000911]

A new text to Robert Palmer's "Addicted to Love".

VI in the Press - Articles on Vi and Clones

Computer User, Nov 1999

Vi Strain - Getting to know your UNIX text editor

http://www.pscu.com/articles/1999/November/article465.htm

Author: Eric Foster-Johnson erc@pconline.com

[Why are we hiding from the police daddy?]

Vi vs Emacs

An endless source for the religious war on editor bashing.

```
"vi is small, fast, and easy to use. Emacs is huge, slow and easy to use :)"
From: softbase@mercury.interpath.net (Scott McMahan - Softbase Systems)
```

Emacsculation [980723]

http://www.rru.com/~meo/rants/emacs.html

A small rant on Emacs by a Vi user.

Author: Miles O'Neal meo@netads.com

"*A naïve comparison of computer text editors*" [970107]

http://www.pobox.com/~drj/comped.html

Test on emacs, vi, and ed for changing all colons to pipes in a password file. Result:

```
emacs:  time: 25.2 real 11.5 user 1.2 sys
```

```
vi:     time:  9.4 real  0.5 user 0.3 sys
ed:     time:  5.8 real  0.5 user 0.2 sys
```

Author: David Jones drj@harlequin.co.uk http://www.pobox.com/~drj/

Vi versus Emacs Essay [960917,970329]

"*A Theoretical and Experimental Comparison of Vi and Emacs*".

http://gmtunx.ee.uct.ac.za/~cherie/ViEmacs/viemacs.html

Author: Cherie Mandy Hurwitz http://gmtunx.ee.uct.ac.za/~cherie/ cherie@gmtunx.ee.uct.ac.za

Vi vs Emacs Study [990225,010615]

"A comparative study of vi and emacs from the perspective of novice and regular users"

Twelve people were tested on both. Result: Emacs wins - both with newbies and skilled people. But "there is no advantage for regular vi users to change to emacs." ;-)

Author: William Knottenbelt - william@cs.uct.ac.za wjk@doc.ic.ac.uk

http://www.csn.net/~bediger/knottenbelt.txt [obsolete]

local copy

"Research Shows: There is A Perfect Editor" [990225,010615]

http://www.users.qwest.net/~eballen1/perfect.editor.html

http://www.csn.net/~bediger/perfect.editor.html [obsolete]

Links to lots of articles about editors. Wish I'd be able to get a copy of them all...

Author: Bruce Ediger eballen1@qwest.net

Vi Folklore

Twenty Years of Berkeley Unix [001010]

From AT&T-Owned to Freely Redistributable

http://www.oreilly.com/catalog/opensources/book/kirkmck.html

"With the arrival of some ADM-3a terminals offering screen-addressable cursors, Joy was finally able to write vi, bringing screen-based editing to Berkeley."

Author: Marshall Kirk McKusick

A chapter taken from the book *Open Sources: Voices from the Open Source Revolution* .

Addicted to Vi

Local copy

http://web.hal.com/users/carolo/jokes/vi.html

New text to the song "Addicted to Love" (by Robert Palmer)

Quines [970630,990202]

Q: What is a quine?

A: "a series of vi keystrokes that when typed in vi produce output the same as the keystrokes."

Example: ii^V^V^V^[BDuplxbbpp^[BDuplxbbpp

Author: Rodney rodney@worf.netins.net

I am pretty sure the author is looking for more quines - especially shorter ones.

More Quines: http://www.nyx.net/~gthompso/quine.htm [990202]

The Cult of Vi

http://www.splange.freeserve.co.uk/misc/vi.html

Vi is a cult. (What else is new?)

VI Stuff

MousePads, Mugs and Tshirts

DATE	ITEM	ADDRESS
001207	**VI Mousepads**	http://www.linuxland.de/katalog/21_fanartikel/vimousepad/framify
990114	**VI Mugs**	http://www.geekcheat.com http://www.vireference.com/vimug.htm
001014	**VI Tshirts**	http://www.thinkgeek.com/stuff/vi-emacs.html

[why are we hiding daddy?]

Actually, there are two versions - a Vi tshirt and an Emacs tshirt. The idea is this: A cartoon showing a boy asking "why are we hiding from the police, daddy?", and his dad answers "because we are using emacs/vi son - and they use vi/emacs".

Disclaimer: I don't do business with any of them.

Outlook/TODO

crevier.org - vi Editor Resources [001004]

http://www.webmaster.crevier.org/depere.wi/vi/

LOGO: http://www.webmaster.crevier.org/imgx/logo/unixvi.gif

[Powered by UNIX vi]

Editing Scheme files in Vi [001004]

http://www.cs.rice.edu/~dorai/scmindent/scmindent.html

Author: Dorai Sitaram ds26@gte.com http://www.cs.rice.edu/~dorai

More About vi [001004,010113]

http://dunne.home.dhs.org/vi.html

http://dunne.home.dhs.org/articles/review.learning_the_vi_editor.html

obsolete:

```
http://members.linuxstart.com/~dunne/vi.html
http://members.linuxstart.com/~dunne/articles/review.learning_the_vi_editor.html
```

Author: Paul Dunne paul.dunne@bigfoot.com

cvi [TODO]

"Chinese VI"

Info on its exitence was given by Steve Bao sbao@profiles.com [000428]

stevie [DATE] [TODO]

OS: [OS/2]

Cost: ???

Author/Copyright: NAME MAIL

SnailMail: ???

Home Page: URL

FAQ: URL

Download: URL

Maillist: ADDRESS

Last release: VERSION [DATE]

Last beta: VERSION [DATE]

Comments: [NAME EMAIL DATE]

vip [TODO]

Hex Editors for Linux [000317]

http://spin.ch/~tpo/linux/hexeds.html

beav, bed, bvi, elvis, fb, fm, ghex, hexdump, hexed, hexedit, hexer, hextype, khexedit, le, lde, mcedit, ncurses hexedit, vche, and xvi.

Vide - a GUI file browser with vi commands [000309]

HomePage: http://vide.sourceforge.net/

Description (official): Vide is a simple **filemanager** with vi keybindings. If you already use vim, Vide enables you to have complete keyboard control over your filemanager without having to learn a new set of commands.·

Description (unofficial): A GUI file browser with vi keybindings: Movements with "hjkl" as well as 'g' adn 'G', scrolling with CTRL-B and CTRL-F, and vi command line interface [:d, :co, :move, :quit, :shell]

Screenshot: http://vide.sourceforge.net/picture.html

Download: http://vide.sourceforge.net/download.html

Requires: GTK+ (1.2 with threads support), and Vim.

Contact: ksteen@users.sourceforge.net

Planned features: Number prefix for commands, selection in "visual mode", configurable key bindings and menus.

Comments: Why does it need a GUI?

POSIX on ex and vi

http://www.pasc.org/interps/unofficial/db/p1003.2

http://www.UNIX-systems.org/onlinepubs/007908799/xcu/ex.html [010320]

http://www.UNIX-systems.org/onlinepubs/007908799/xcu/vi.html [010320]

Add info about POSIX description of vi commands to the comparison page about the "options".

Vi emulation in CRiSP [990712]

The editor CRiSP (commercial follow-up to the editor BRIEF) has a "vi emulation mode". Anyone using this? Feedback, please!

Have sent email to Bryan Althaus bryan@panix.com who is apparently using this. [990712]

Vi for Japanese - a list of ported clones. [980430]

http://www.vector.co.jp/vpack/filearea/dos/writing/edit/vi/

Premia CodeWrite - vi emulation

Doug Kellogg [980417] is using it: "It is up to version 5.1 and is published by Premia in Portland, Oregon, USA (www.premia.com) and is descended from the Sage Programmers Editor (SPE) from what was Polytron and is now Sage, also of Portland."

Charlie Guttenberg c_guttenberg@merge.com informed me [980414]: "Codewright is a programmer's editor made by the Premia Corp. Besides basic editing, it can tie into whatever sccs and build environments you may have. The version we have, V3.1e (1995), supports emulation of four editors: CUA, Epsilon, Brief, and good old vi."

Prepare Codewright HomePage: http://www.premia.com/

SpexEdit

Marketed by "Little Wing": Little Wing / 4618 JFK Blvd, Suite 176 / North Little Rock, Arkansas 72116 / (501)771-2408

Supports most of vi commands. Free demo disk available.

Vi-Editor.org Buttons

Some buttons for your own page on Vi:

Vi-Editor.org Mirrors

These pages are mirrored at these sites:

```
HomeSite: [home]
Mirrors: [DE] [ZA]
Do you want to mirror these pages?
Be my guest!
```

Comments? Send them! My address:

--
Sven Guckes guckes@vi-editor.org

"The wonderful thing about Tiggers [excerpt of "Tigger's Song" in
Is Tiggers are wonderful chaps "Winnie the Pooh and Tigger Too"]
They're loaded with vim and with vigor..."

Vi Substitute Guide (HOWTO use the ":s" command)

The command ":substitute" is a very powerful command. It makes use of **addresses** (esp. **line ranges**), **regular expressions** (aka **patterns**), and **flags** to request for confirmation and other things. You will certainly use this quite often once you know how to use it. This guide should give you a fast intro to the substitution command, so you can hopefully make some good use of it after ten minutes.

NOTE: This guide starts off with basic VI functionality, but also aims to show some of VIM's enhancements. I use it daily and I could not live without it's extra features.

Here we go:

As the *substitute* command is complex - so is its definition. It involves the definition of terms *range*, *pattern*, *string* and some options:

```
:[range]s[ubstitute]/pattern/string/[options]
```

Note: The separator between 's' and the *pattern* is the same as between *pattern* and *string* and *string* must always be terminated with it. However, you can chose the character. It does not need be the slash - it can also be the hash ('#'):

```
:[range]s[ubstitute]#pattern#string#[options]
```

The substitute command consists of nine parts. You need not give all nine parameters - parameters in square brackets are optional, ie they are not required. However, the command name 's' and the separator around the pattern are always required. So the shortest substitute command is ":s///". It will be explained below (unless I forget ;-). With VIM there are three abbreviations, namely ":&", ":~", and '&' (see ":help :&", ":help :~", and ":help &").

Vi :substitute - all parts explained

Part 0 - the ex mode colon

> The substitute command is a command of the unerdlying editor "ex". Therefore you must enter the enter first to switch to "ex mode". As it is required for all ex commands it is not a part of the command per se. Therefore this is "part 0".

Part 1 - The **[range]** aka "scope" or "block"

> The first part is the *range*. The square brackets around it denote the fact that this "prefix parameter" is optional. As vanilla Vi does not have online help we'll take a look at VIM's helpfile with the command ":help range":

```
Line numbers may be specified with:                    *:range*
        {number}        an absolute line number
        .               the current line                        *:.*
        $               the last line in the file               *:$*
        %               equal to 1,$ (the entire file)          *:%*
        *               equal to '<,'> (the Visual area)        *:star*
        't              position of mark t (lower case)         *:'*
        /{pattern}[/]   the next line where {pattern} matches   *:/*
        ?{pattern}[?]   the previous line where {pattern} matches *:?*
        \/              the next line where the previously used search
                        pattern matches
```

```
        \?              the previous line where the previously used search
                        pattern matches
        \&              the next line where the previously used substitute
                        pattern matches

    Each may be followed (several times) by '+' or '-' and an optional number.
    This number is added or subtracted from the preceding line number.  If the
    number is omitted, 1 is used.

    The "/" and "?" may be preceded with another address.  The search starts from
    there.  The "/" and "?" after {pattern} are required to separate the pattern
    from anything that follows.

    The {number} must be between 0 and the number of lines in the file.  A 0 is
    interpreted as a 1, except with the commands tag, pop and read.

    Examples:
            .+3             three lines below the cursor
            /that/+1        the line below the next line containing "that"
            .,$             from current line until end of file
            0;/that         the first line containing "that"

    Some commands allow for a count after the command.  This count is used as the
    number of lines to be used, starting with the line given in the last line
    specifier (the default is the cursor line).  The commands that accept a count
    are the ones that use a range but do not have a file name argument (because
    a file name can also be a number).

    Examples:
            :s/x/X/g 5      substitute 'x' by 'X' in the current line and four
                            following lines
            :23d 4          delete lines 23, 24, 25 and 26

    A range should have the lower line number first.  If this is not the case, Vim
    will ask you if it should swap the line numbers.  This is not done within the
    global command ":g".
```

Well, I do hope that this is clear to you. If not - ask me!

Part 2 **s[ubstitute]** - The command name.

The second part is the name of the command. As you can abbreviate every command with Vi with its unique prefix the "ubstitute" is optional, so you need not type the full name, but only an 's'. However, the 's' is required.

Part 3 - The "separator" (usually '/' or '#')

The "separator" is a character which seperates the command name from the pattern, seperates the pattern from the substitution string, and the substitution string from the options. You can chose any(?) character as a separator, but usually the slash (/) is used. When substitutions contain the slash as a literal character you should chose some other character that is not contained in the pattern or substitutions string. And if this is not possible then you have to "escape" it with a backslash (\): ":s/\/path\/filename/\/newpath\/newfilename/"

Part 4 - The *pattern*

A pattern is a complex construct. There is no general definition of "pattern", however; it differs depending on its use. Here we ar using a "search pattern", so VIM4.5 will give this text with ":help search_pattern": The definition of a pattern: *search_pattern* Patterns may contain special characters, depending on the setting of the 'magic' option. */bar* */\bar* 1. A pattern is one or more branches, separated by "\|". It matches anything that matches one of the branches. Example: "foo\|beep" matches "foo" and "beep". 2. A branch is one or more pieces, concatenated. It matches a match for the first, followed by a match for the second, etc. Example: "foo[0-9]beep", first match "foo", then a digit and then "beep". 3. A piece is an atom, possibly followed by: magic nomagic */star* */\star* * * matches 0 or more of the preceding atom */\+* \+ \+ matches 1 or more of the preceding atom {not in Vi} */\=* \=

\= matches 0 or 1 of the preceding atom {not in Vi} Examples: .* .* matches anything, also empty string ^.\+$ ^.\+$ matches any non-empty line foo\= foo\= matches "fo" and "foo" 4. An atom can be: - One of these five: magic nomagic ^ ^ at beginning of pattern, matches start of line */^* $ $ at end of pattern or in front of "\|", */$* matches end of line . \. matches any single character */.* */\.* \< \< matches the beginning of a word */\<* \> \> matches the end of a word */\>* \i \i matches any identifier character (see */\i* 'isident' option) {not in Vi} \I \I like "\i", but excluding digits {not in Vi} */\I* \k \k matches any keyword character (see */\k* 'iskeyword' option) {not in Vi} \K \K like "\k", but excluding digits {not in Vi} */\K* \f \f matches any file name character (see */\f* 'isfname' option) {not in Vi} \F \F like "\f", but excluding digits {not in Vi} */\F* \p \p matches any printable character (see */\p* 'isprint' option) {not in Vi} \P \P like "\p", but excluding digits {not in Vi} */\P* \e \e <Esc> */\e* \t \t <Tab> */\t* \r \r <CR> */\r* \b \b <BS> */\b* ~ \~ matches the last given substitute string */~* */\~* \(\) \(\) A pattern enclosed by escaped parentheses */\(\)* (e.g., "\(^a\)") matches that pattern x x A single character, with no special meaning, matches itself \x \x A backslash followed by a single character, */* with no special meaning, matches the single character [] \[] A range. This is a sequence of characters */[]* enclosed in "[]" or "\[]". It matches any */\[]* single character from the sequence. If the sequence begins with "^", it matches any single character NOT in the sequence. If two characters in the sequence are separated by '-', this is shorthand for the full list of ASCII characters between them. E.g., "[0-9]" matches any decimal digit. To include a literal "]" in the sequence, make it the first character (following a possible "^"). E.g., "[]xyz]" or "[^]xyz]". To include a literal '-', make it the first or last character. If the 'ignorecase' option is on, the case of letters is ignored. It is impossible to have a pattern that contains a line break. Examples: ^beep(Probably the start of the C function "beep". [a-zA-Z]$ Any alphabetic character at the end of a line. \<\I\i or \(^\|[^a-zA-Z0-9_]\)[a-zA-Z_]\+[a-zA-Z0-9_]* A C identifier (will stop in front of it). \(\.$\|\. \) A period followed by end-of-line or a space. Note that "\(\. \|\.$\)" does not do the same, because '$' is not end-of-line in front of '\)'. This was done to remain Vi-compatible. [.!?][])"']*\($\|[ ]\) A search pattern that finds the end of a sentence, with almost the same definition as the ")" command. Technical detail: <Nul> characters in the file are stored as <NL> in memory. In the display they are shown as "^@". The translation is done when reading and writing files. To match a <Nul> with a search pattern you can just enter CTRL-@ or "CTRL-V 000". This is probably just what you expect. Internally the character is replaced with a <NL> in the search pattern. What is unusual is that typing CTRL-V CTRL-J also inserts a <NL>, thus also searches for a <Nul> in the file. {Vi cannot handle <Nul> characters in the file at all} Now, this is pretty complex, isn't it? [todo: examples]

Part 5 - The separator (again)

This separator must be the same character as the separator used before.

Part 6 - The "substitution string"

The substitution string also has meta characters, but not as many as the "pattern" used before.

Part 7 - The separator (again)

This separator must be the same character as the separator used before.

Part 8

Part

/

: Part three: The separator. The slash seperates the command name from the following pattern. It need not be a slash, a hash ('#') will do, too. But then you will have to use a hash at the end of the pattern and after the substitution string, too. For now we will just use the slash.

{pattern}

:s/vi/VIM/

Substitute the first occurrence of "vi" with "VIM" on the current line.

More examples

Command: **:.,.+7s/^/foo/**
Read command as: from "this line" to "this line plus seven lines" substitute "^" (beginning of the current line) with "foo".
Informal Description: Add "foo" this and the follwoing seven lines. Remember: The address '.' refers to current line, but it can be left out. So the previous command can be shortened to: **:,+7s/^/foo/**

Mark them!

Before you apply a substitution command on some lines you usually have to figure out which lines these are so you can use their addresses in the command, eg:

```
:23,42s/foo/bar/
```

But how do you find theat the first line is #23 and that the last line is line #42? You probably have to move the cursor a lot and also write down the number on a piece of paper.

However, there is an easier way: Set a mark on those lines.

Setting a mark is done with the command `'m'` followed by any (lowercase) letter, ie from 'a' to 'z'. The current line then is "marked" with that letter.

So, set a mark when the cursor is on the first line of the text to be operated on, and set another when the cursor is on the last line. For examples, we usually use the marks 'a' and 'b' for the first and last line, respectively. So the substitution starts on line (marked with) 'a' and it will end on the line (marked with) 'b'.

Now you can reference these *addresses* with `'a` and `'b` in the substitution command:

```
:'a,'bs/foo/bar/
```

So - no more writing down line numbers. Good.

But this method still means to move the cursor onto those lines to set a mark on them.

VIM examples

Visual mode -> line range

I almost never want to figure out the exact line range to apply a substitution on, and that's why I use Vim's visual mode and then the operator ':'. You just select a range of lines using the comamnd 'V' and move to the last line. Then you type ':' and Vim will switch to the command line, inserting "'<,'>" as the line range. These marks stand for "first visual line" and "last visual line", respectively. Now you can type in the substitution command:

```
:'<,'>s/Vi/Vim/g
```

Or any other command for that matter:

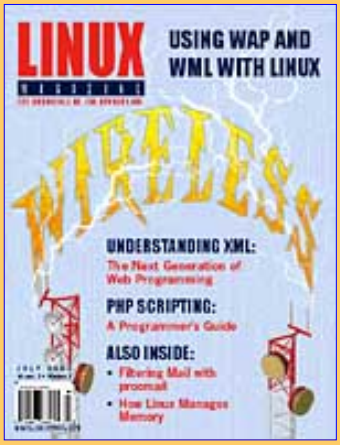
```
:'<,'>d
```

But, of course, this deletion command would have been easier by using the 'd' operator in visual mode right away. ;-)

```
URL:         http://www.math.fu-berlin.de/~guckes/vi/substitute.html
Created:     Mon Oct 21 19:39:13 CEST 1996
```

Send feedback on this page to

Sven Guckes guckes-vi-subst-faq@math.fu-berlin.de



DOCUMENTATION

Linux Documentation Project

http://www.linuxdoc.org/

The granddaddy of all Linux doc sites, the LDP has been hosting FAQs, *man* pages, HOWTOs, and Linux Guides for a long time. The LDP boasts a substantial collection of Linux information in a variety of formats. The HOWTOs on this site are the definitive place to start for any Linux problem you may be trying to troubleshoot. The LDP recently got a facelift and has experienced a resurgence, thanks to some new blood.

Open Source Writer's Group

http://www.oswg.org:8080/oswg

Formed last year by Deb Richardson of LinuxChix fame, the Open Source Writer's Group is a gathering place for open source authors to hook up with open source projects.

Linux Resources

http://www2.linuxjournal.com/cgi-bin/frames.pl/lr-toc.html

Linux Journal provides an excellent resource site for the Linux newbie and guru alike. Find out all about what Linux is -- and where you can get it, meet other people who use it, and read more about it.

GNU Project Documentation

http://www.gnu.org/doc/doc.html

What good is free software if you don't know how to use it? Happily, the wonderful folks with the GNU Project have provided a comprehensive section of documentation for Emacs Lisp, GNU C, using the GDB debugger, and a whole lot more. If you're in a philosophical mood you can read Richard Stallman's essay on Free Software and Free Manuals. (Here's a hint: he's for them.)

The Vim Homepage

http://www.vim.org/

Get a handle on the editor that strikes fear into the heart of newbies everywhere, Vim. Vim is the most popular *vi* clone for Linux, and the Vim homepage has tons of information on using Vim and how to customize it.

THE X WINDOW SYSTEM

Themes.org

http://www.themes.org

Wanna make X purty? Hook up with the folks at 

Themes. org. They've got the stash of themes and utilities for the most popular desktops and window managers. All the goodies you've ever wanted to dress up your desktop, as well as X resources and news updates on window managers and other popular packages. Themes.org has sections for SawMill, Afterstep, Enlightenment, WindowMaker, KDE, and a lot more.

KDE Home

http://www.kde.org

KDE's home page is the place to go to get any information about developing for the K Desktop Environment, or simply to keep updated on the latest application releases for KDE. KDE also has links to related projects like K Office.

GNOME Home

http://www.GNOME.org

Gnome.org is one of the best-organized sites out there for a free-software project. Get the latest scoop on development of GNOME, join the GNOME mailing list, or start downloading! Developers can check out the latest additions to the CVS, and GNOME users will find a lot of useful documentation on the site.

XFree86 Project

http://www.xfree86.org

The XFree86 Project site will give you the lowdown on the windowing system that ships with Linux, where to get the current release, and an excellent FAQ about the project. The XFree86 site is a must-read for anyone new to Linux, even if your video card is running smoothly.

Window Managers for X

http://www.PLiG.org/xwinman

The Window Managers for X site is the most comprehensive site about X window managers. Everything from ancient to the latest and greatest window managers can be found on the site. Don't quite know the difference between a window manager and a desktop environment? Learn the basics here.

<< prev page 1 2 3 4 5 6 7 8 **9** 10 11 next >>

Linux Magazine / June 2000 / FEATURES

Linux Magazine's 100 Best Linux Web Sites

feedback
mail to a friend
printer-friendly version

LINUX MAGAZINE • subscribe • advertise • customer service • back issues • feedback • contacts

Vigor

Current Version: 0.016

Vigor (pronouced like "Igor", Dr. Frankenstein's assistant), the popular Unix editor vi with the addition of the Vigor Assistant, has arrived.

Recent News

Vigor Training Program Initiated

After seeing the success of similar industry efforts, we at Piquan Software, the developers of Vigor, have initiated a certification and training program to give industry professionals the skills needed to use Vigor most efficiently.

The various levels of training available are as follows:

Certification Level	Prerequisites	Skills	Price
Vigor Evildoer Specialist (VES)	-	About 90 percent of the Unix industry uses Vigor, or its inferior precursors, so chances are very good that you are one of the many people using Vigor to get your work done. Now you can prove it, and give yourself and your organization a competitive edge.	$2000.00 US
Vigor Certified Evildoer (VCE)	-	The VCE program allows you to demonstrate to your victims that you have the specialized knowledge required to perpetrate evil with particular Vigor features.	$2000.00 US

Vigor Certified Evildoer + Internet (VCE + Internet)	VCE	The VCE + Internet program gives you the skills needed to successfully use and support specific features of Vigor in today's Internet-demanding world. Information on composing spam emails, fly-by-night .com web pages, and other critical Internet tasks for the modern evil dispensation professional, using the Vigor platform, are covered.	$3000.00 US
Vigor Certified Evil Scheme Engineer (VCESE)	Six VCE certifications obtained within the last one (1) week	For evildoing professionals, Piquan Solutions offers the VCESE credential. VCESEs are qualified to plan, implement, maintain, and support plots to take over the world, mind-control schemes, destruction of enemies, and other evil schemes, using the Vigor environment. Study materials will center around "Things To Never Do As An Evil Overloard". Graduates will have access to beta versions of Vigor in a schedule as approved by Piquan Solutions.	$5000.00 US
Vigor Certified Trainer (VCT)	VCESE	VCT's play an important role in Piquan Industries' re-education process. VCTs are qualified instructionally and certified technically personally by Piquan to deliver training to evildoers everywhere. Study materials center around Anthony Burgess and Stanley Kubrick's *A Clockwork Orange*.	$9000.00 US

All certifications include a certificate suitable for framing, an official and individual certification number (while supplies last), and a Vigor Evildoer button to wear at conferences, to let everybody know that you are a certified Vigor Evildoer. VCESEs also receive a secret decoder nose ring.

Since Piquan Enterprises is constantly updating our curriculum and programs based on industry input, all training and certification is on an as-available basis, and is subject to change,

cancellation, or expiration without notice. Piquan is not responsible for the conduct of certified Evildoers. Thanks to Garrett Moffitt for the idea.

Vigor Goes Mainstream!

The first mainstream-mag review of Vigor has been released! That's right, Salon, a general-interest magazine, has run a short blurb on Vigor and the chilling implications it could have on the free software movement. Or something.

Vigor Revived

It's been a month and a half since the last release of Vigor. One would think that I'd be willing to leave well enough alone. Well, I'm not. I got a bit of free time, and the LinuxCare review sparked some interest, so I am now getting ideas for future improvements.

As always, bug fixes are my highest priority, but some new features would be nice too. Some of the things I'd like to see in future versions are:

- Improved graphics. I use XBMs in Vigor because it was the quickest drawing method to implement, and I wanted to get the first release out during the Vigor storyline on User Friendly. Now that I've got some time, I'd like to improve the graphics handler and the artwork.
- More hooks! More confirmations, more annoying helpful hints (like the ones given when you start insert mode), more random popups, more!
- More animations! At present, there are precisely two animations (count em!) in Vigor. This is not nearly insideous enough. Naturally, with animations, you lead into...
- Sound effects (via Network Audio System). This naturally suggests...
- Speech (via Festival). This isn't that hard to do using Festival's server mode. (Thanks to my coworkers at HP for the idea, the deranged loonies!)
- Microsoft Windows support. Judging by the feedback I get, people who aren't Unix devotees just don't get the point of Vigor, but strangely enough I've gotten multiple requests to port Vigor to Windows. Well, I suppose I may as well... Kinda one of those "full circle" things.

This by no means is an exhaustive list. If people send me patches, then I'll add just about anything. These are just some of the things that I might do over the next few weeks, months, whatever.

About Vigor

Based on the User Friendly comic strip storyline from 4 January to 14 January, Vigor brings all the features of traditional Unix vi, plus the friendly and helpful Vigor Assistant. (If you aren't familiar with User Friendly the Comic Strip, quit bothering with Vigor and go out and look there first. It's well worth the trip! Don't worry, we'll wait.)

Enter the world of Vigor! Come, join us, watch the paperclip, don't be afraid... *cough* Sorry, where was I?

Vigor is based on the open-source nvi program, for which I apologize to the nvi authors. The Vigor assistant was bolted on using the Tcl facility that nvi had, and a bit of my own C code to cope with Tk.

I'm improving(?) Vigor based on suggestions from the user community. Send ideas, bug reports, and patches to joelh@gnu.org, and watch this page for updates. New versions have been released at a rate ranging from several each evening to once a week, depending on user comments and free time.

Press Coverage

I am always suprised when somebody mentions Vigor. I really didn't expect it to spread by word-of-mouth at all, let alone people write about it. Keep telling your friends, your enemies, your coworkers, everybody, about Vigor.

Particular thanks to Illiad for putting the original release announcement and multiple subsequent announcements in the User Friendly News, not to mention the original inspiration!

Vigor Review and Interview

Rado has written a review of Vigor and conducted an interview with me about Vigor. The interview also covers general topics like the free software community, Pitr, and what Sun Tzu teaches us about a certin other paperclip-using company. The full version is availible at LinuxTicker, and if you speak Croatian, look in PC Chip magazine (broj 57, veljaca 2000).

Vigor Goes Mainstream!

The first mainstream-mag review of Vigor has been released! That's right, Salon, a general-interest magazine, has run a short blurb on Vigor and the chilling implications it could have on the free software movement. Or something.

Vigor App of the Week

LinuxCare's Brett Neely wrote up a short review of Vigor for his App of the Week column.

User Comments

- You are a sick, sick person, and I admire that greatly. - *Illiad, author of User Friendly*
- AUUUUUGGGGGHHHHHH!!!!! - *Keith Bostic, author of nvi*

- I haven't laughed so hard in a long time.- *Arnold Robbins, coauthor of* Learning the vi Editor
- Congratulations, Brother Piquan... how diabolically clever of you...- *Eric S. Raymond*
- The entire open-source movement is, of course, a Discordian creation.- *Amphigoricus the Turgid, K.S.C.*
- Why ? - *Sander Nooy*
- I am laughing in an inapropropriate manner for an office building like this... - *Nathanael Lierly*
- By reading this message, you have agreed to stand on your head while gargling water and proclaming "Weird Al Yankovic is Lord of the Universe!" - *J.J. Ramsey*
- "I am wishink I had Unix system instead of this silly VMSstation so I too could be experiencing pain of vigo directly rather than vicariously!" - *Atlant Schmidt*
- The following addresses had permanent fatal errors... - *Mailer-Daemon*
- On behalf of evildoers everywhere, I thank you for your heroic efforts in bringing The Assistant to such a long-standing bastion of the opposition. You will be remembered when The Time comes. - *Michael Jinks*

Author Comments

- "I just thought I was writing a weekend hack; I had no idea what I was unleashing!" (Said after noting over 20,000 web page hits.)
- The way I figure, after you take into account all the victims who have used it, Vigor is single-handedly responsible for immeasurable amounts of lost productivity. In this, I think I have captured the true nature of the original.
- It's amazing how a weekend hack can take up all your free time for a month.
- (When asked to confirm that Vigor is not a productivity app:) From its inception, Vigor has been designed to be utterly useless, not to mention a pain to use. While building the interface, I double-checked everything to ensure that I didn't accidently add a useful feature. [Note: See below for an example of how that didn't work.] People have suggested making Vigor useful; my reply is that it would then cease to be Vigor. I'm sure you can think of other examples of popular, yet useless, editors. Besides, I don't even use vi... Who am I to decide how to improve it?

Revisions

Vigor was rushed out in order to be released during User Friendly's Vigor storyline. It originally had several bugs, and several unimplemented features. I've been fixing bugs and adding features since the day Vigor was relelased. There are presently no outstanding Vigor bugs that I'm aware of; let me know if you find something!

There is one remaining problem that we've only seen on two systems: Vigor, when it starts, exits with the message `unknown floating-point error, errno = 84`. Eivind

Tagneth, who first reported this, traced it down to a bug in Tcl, and submitted a patch to the Tcl maintainers.

Even if you're not seeing bugs, it's good to hear how much success/failure people are having on different platforms. Let me know how things work for you!

Version 0.016 (May 2 19:36)

> Create Whizzers, specialized dialogs to help you perform specialized tasks. Presently, there are Whizzers for a few programming languages, and some common editing tasks. More should follow.

Version 0.015 (Mar 25 15:17)

> Disable by default built-in curses, db, and re libraries. Most OS's these days have proper libraries for these, and I was seeing conflicts (particularly on SuSE under Vigor 0.014). This may break things on some other OS's, probably older ones. If so, try using --enable-curses --enable-db --enable-re on the command line.

Version 0.014 (Feb 8 14:50)

> Remove the hack to the EULA that produced the Jumping Vigor Bug.

Version 0.013 (Feb 5 11:53)

> Fix to configure.in: it looked for Tcl on Debian systems when it was supposed to look for Tk.

Version 0.012 (Feb 4 00:28)

> Updated EULA, made dialog box move along with assistant

Version 0.011 (Feb 2 00:10)

> Look for libtcl8.2 etc (as opposed to libtcl82 etc); handle select() calls prior to assistant initialization

Version 0.010 (Jan 19 23:16)

> Modified EULA, changed fractional cm screen specs to pixel counts, use libtcl82 if found, look for uint8_t etc

Version 0.009 (Jan 18 15:11)

> Improved Tcl/Tk detection, added EULA, detect -ldl

Version 0.008 (Jan 17 21:20)

> Added new comments, touched up comment probabilities, made sure that comments don't overlap with animations, modified configure.in to look for libtcl80.a

Version 0.007 (Jan 17 12:25)

> Fixed make install to install vigor instead of vi

Version 0.006 (Jan 16 12:42)

> Further touch-up of "confirm" dialog's probabilities; rename executable to vigor

Version 0.005 (Jan 16 01:01)

> Fixed make install; write Tk error messages to stderr before vi initializes the screen

Version 0.004 (Jan 16 00:21)

Updated configure to match configure.in (oops!)

Version 0.003 (Jan 15 23:45)

Added random quotes and animations; compiled vigor.tcl into executable

Version 0.002 (Jan 15 20:16)

Updated graphics; touched-up "confirm" dialog

Version 0.001 (Jan 14 01:05)

Initial release

Screenshots, Graphics, and Other Goodies

The most frequent request I've had since releasing Vigor recently is for screenshots. I've made a separate page with a few for those who don't have the time or tools to compile Vigor; enjoy!

Tom Mulder created the Vigor logo at the top of the page, and the logo on the screenshots page as well. Thanks, Tom!

And, if you just can't get enough paperclip action (you sick puppy you), svo (aka Viacheslav Slavinsky) has created a brief Vigor MPEG, an excellent bit of artwork. For those who can't see view MPEGs, he has also created some stills: still 1 and still 2. Yay, svo!

Downloading Vigor

Source tarball (0.016)

Get the source at http://www.red-bean.com/~joelh/vigor-0.016.tar.gz and have fun with it! You may want to read over the section "Installing Vigor" below, though. I release by source tarball, so this is usually the most up-to-date version. It's also the only one that I can help with install problems on, since I am not generally familiar with the other formats.

Note that, because of namespace conflicts, as of 0.015, I changed Vigor to use the system curses, db, and re libraries. This **WILL** cause problems on some systems. (It also will fix problems on others.) If you have problems building Vigor from source, try running configure with "--enable-db --enable-re".

Debian (0.016)

Colin Watson has made an unofficial package of Vigor for Debian 2.2. He's graciously providing both the diffs and a i386 binary package. (Later versions may also be available; check his dist site.)

FreeBSD (0.014 at last check)

There is now a Vigor port in the 4.0-current port tree. I use FreeBSD myself, but I'm not the one responsible for the port (and I don't have commit privs), so it may fall out-of-date without my knowing it. Thanks to Bill Fumerola for committing this! (And I had been told that Vigor should be committed...)

NetBSD (0.010 at last check)

There is also a Vigor port in the NetBSD -current package tree. The NetBSD package system has binaries for i386 (1.4.2 and 1.4S), macppc, ofppc, and powerpc (1.4.2), and of course the usual source distro. Thanks to Hubert Feyrer for submitting the package (at 0.009), and Jaromir Dolecek for updating it to 0.010!

RedHat RPMs (0.010)

RPMs are available in source, Redhat 6/i386 binary, Redhat 5/i386 binary, and Redhat 6/Sparc64 binary formats. Thanks to Ganesh Sittampalam for making these available!

Windows

I'm looking into building a Cygwin+native Tk port for Windows, but I'm not yet sure if that will happen. Watch this space for details!

Installing Vigor

At present, some parts of Vigor are in a somewhat primitive state, particularly as far as the build is concerned. Suggestions for how to make things build on various OS's are gratefully accepted. For the most part, compiling vigor usually goes something like this:

```
gunzip vigor-0.016.tar.gz
tar -xvf vigor-0.016.tar
cd vigor-0.016/build
export ADDCPPFLAGS="-I/usr/local/include/tcl8.2
-I/usr/local/include/tk8.2"
export ADDLDFLAGS="-L/usr/local/lib"
./configure
make install
vigor
```

Substitute your locations for tcl.h, tk.h, and libtcl.a on the ADDCPPFLAGS and ADDLDFLAGS lines. If you still have problems building Vigor, read the suggestions below, look in the build/README file, and if all else fails, drop me an email describing the problem.

Vigor does not support nvi's Tk interface (and configure doesn't give you that option), and may

have trouble compiling in the Perl interpreter (which is not included by default). [Update: I think that I fixed the problem with the Perl interpreter.] You will want to read build/README to get the skinny on building nvi. Vigor will always build in nvi's TCL interpreter.

Vigor is written in C and Tcl/Tk. You must have Tcl/Tk installed to use Vigor. I'm not sure what versions work. I wrote from Ousterhout's book, so I think anything after 7.3 is fine. Most people seem to have been using Tcl/Tk 8.0 or 8.2, and things seem fine there.

Note that, because of namespace conflicts, as of 0.015, I changed Vigor to use the system curses, db, and re libraries. This **WILL** cause problems on some systems. (It also will fix problems on others.) If you have problems building Vigor from source, try running configure with "--enable-db --enable-re".

OS Notes

You may want to look over whatever systems are similar to yours.

LinuxPPC Redhat GNU/Linux 5

I've heard reports of trouble building on this system. Has anybody successfully compiled Vigor on LinuxPPC Redhat GNU/Linux 5, or any other LinuxPPC platform?

Redhat GNU/Linux 5.2

You may need to set the environment variable ADDLIBS to "-ldl" before running configure. (I've heard a report that "-lm -ltermcap" are also needed, but the reporter now believes that to be incorrect. Adding them won't hurt anything, though.) **UPDATE:** As of Vigor 0.009, this should no longer be necessary; configure will look for and use -ldl on its own.

Even so you may still get this warning:

```
ex_script.o: In function `sscr_pty':
ex_script.o(.text+0xf63): warning: revoke is not
implemented and will always fail
```

This warning (and many others) are completely ignorable.

Sven Winnecke, who reported this, also notes that the program builds fine on Redhat 6.1 systems.

HP-UX 10.20 (and probably 11.00)

On HP-UX 10.20, using the binary installs of Tcl and Tk from the University of Utah's HP-UX software site, I had to create a symlink from /opt/tcl8.3/tk8.0 to /opt/tk8.0/lib/X11. (I may have the version numbers wrong; I'm typing this from memory.) Also, after making running configure, I had to change "-ltcl" in the Makefile to /opt/tcl8.3/lib/libtcl.a or the compile would fail. I don't yet know why.

FreeBSD (and possibly others)

tcl.h, tk.h, and friends must be in the cpp include path. If they are in some directory like /usr/local/include/tcl8.0, I suggest setting ADDCPPFLAGS to "-I/usr/local/include/tcl8.0" before running configure. (Note that configure does not check for tcl.h, although it does look for -ltcl.) The same goes for Tk. Again, see build/README for details, and bear in mind that I haven't edited it for Vigor's differences from nvi.

A few OS's (notably FreeBSD) install Tcl with the version number appended to the library. On such an OS, you must make a symlink to libtcl.a or libtcl.so (with the appropriate extention for shared libraries on your OS); same goes for libtk.

Solaris 2.5.1

The u_int8_t problems on Solaris have been fixed as of Vigor 0.010. Thanks to John West for providing the information to fix it!

Cygwin

When configuring for Cygwin, create a directory /var/preserve/vi.recover and make it world-writable. (See the comments in configure.in for alternative directories.) (I haven't yet received confirmation that it works after that, though.)

Jumping Vigor Bug

Some users have reported that Vigor 0.013, after starting up, jumps around the screen as fast as the system can go. This is not correct behavior. I haven't yet figured out why it happens, but as of Vigor 0.014, I've disabled the code that made it happen. I would welcome ideas from Tk wizards about why this was happening! (It was related to an ugly hack at the top of vigor_eula... see the comments there.) If you see Vigor jumping more than once a second, you've probably rediscovered this bug; please let me know!

In Case Of Trouble

If you're having trouble building Vigor and are writing for help, please send me as much of the following as possible. If you don't have everything, write me anyway, but this information will help me figure out the problem.

- Your architecture, as reported by the `config.guess` program supplied with Vigor. If you believe that `config.guess` is supplying an incorrect or incomplete architecture, then tell me that too.
- The version of Vigor you are trying to compile. I update Vigor regularly, so by the time I see your email, I may be working with a later version.

- How you downloaded Vigor. I need to know if you got the tarball, the source RPMs, or a binary distribution.
- What arguments you gave `configure` when you ran it.
- The environment variables ADDLDFLAGS, ADDCPPFLAGS, CFLAGS, and LDFLAGS if you have changed them.
- The version of Tcl and Tk you are using, and the filenames under which they are installed.

If the trouble is happening when you run `configure`, send me the file `config.log` (or at least the final bit of it).

If the trouble is happening when you run `make`, send me the last several lines that `make` produced.

If the trouble is when you run Vigor, send me a description of the trouble you're seeing. "It doesn't work for me" is not a bug report.

I'm glad to help out as I can. Vigor has lots of bugs, and I'm constantly trying to find them and clean them up. Sending me complete information will help me fix Vigor sooner.

Old News

Vigor Useful?!?

One user actually found Vigor to be **useful**! After all that time and effort I put into making a totally useless app, somebody has to go and get some benefit out of it.

The user in question had been trying to learn Vi for some time. However, his manual neglected to mention the difference between the Insert mode and Command mode. (What kind of tutorial was he using?) His confusion was lifted when he saw the prompt from Vigor:

> You have not entered insert mode before. While you're in insert mode, remember that you need to return to command mode before entering Vigor commands!

Before Vigor's public release, I had changed that prompt once because I deemed it "too useful" (it mentioned the Esc key by name). Apparently, I just didn't make it useless enough.

I'll tell you, some people just don't know a bad thing when they've got it.

Vigor Scooped?

Yes, it's true, the open-source vi clone VIM already has its own smiling face.

Apparently, there is an option to VIM called VimBuddy, which will display an ASCII smiley (eg, ":-)") on your status line for different status reports. The Windows version of VIM will also update the icon with a face as well!

I still won't say what my motives for writing Vigor were, but I think you can be sure they weren't the same as the ones that prompted the creation of VimBuddy.

(Note to the interested: I don't have any other information about VimBuddy; you may want to do a search on egroups. That shows up a few discussions, some code, and the author's name.)

Acknowledgments

Vigor would never have been had it not been for these people's efforts.

- Illiad, author of User Friendly the Comic Strip, for keeping my spirits up with his work for years, and for conceiving of Vigor to begin with.
- The authors of nvi (listed in the README), particularly Steve Kirkendall and George Neville-Neil. Without nvi to work from, there could be no Vigor.
- John K. Ousterhout and the other contributors to Tcl/Tk, without which I couldn't have written Vigor in time to release during the Vigor storyline.
- Peter Mattis, Spencer Kimball, and the other contributors to The GIMP, for helping even me make half-decent art. (You don't want to know what the art looked like before GIMP's help.)
- Michael J. Gourlay and the other contributors to xmorph.
- The guys at Red Bean Software, for hosting Vigor's distribution.
- Scott Talafuse, Beth Gemeny, and some weirdo named Andrew who crashed in my apartment for the last week or so, for providing ideas and moral support.
- Noah Friedman, Jamie Zawinski, Jim Blandy, Thomas Bushnell, Roland McGrath, and a cast of dozens for (unknowingly) providing most of Vigor's snide remarks.
- The Vigor users who provided me with ideas and bug reports... Couldn't have done it otherwise!

(Note: If I put your name on this page, I probably didn't include the email addresses, because I don't want spammers to pick it up. If you don't mind, let me know and I'll add your email address.)

Release Announcement

FOR IMMEDIATE RELEASE
Mountain View, CA. In a recent fit of madness, hacker Joel Ray "Piquan" Holveck has released Vigor, a version of the popular Unix editor vi featuring the Vigor Assistant. Holveck declined to comment on his true motives for creating the program, although it has been confirmed that he was inspired by Pitr of User Friendly the Comic Strip. A supporter of the Free Software Foundation, Holveck denies rumors that he created the program as part of a plot to encourage

the use of Emacs based on Greg's 10 Jan comment. Vigor is currently available online at http://www.red-bean.com/~joelh/vigor/ and is expected to undergo daily improvements based on user input for the next several days.

Just a small note on my signature on the Vim Homepage:

Gary Fritz fritz@frii.com told me that the lyrics to "Tiggers Song" are these:

```
The wonderful thing about Tiggers is --
Tiggers are wonderful things.
Their tops  are made of rubber,
Their tails are made of springs.

They're bouncy, trouncy, flouncy, pouncy,
Full of fun fun fun!
And the most wonderful thing about Tiggers is --
I'm the only one!
```

So - not much about "vim" or "vigor". Hmm...

But Peter Riley nestene@cyberspace.org told me on 990726 that the "quote about vim and vigour comes from the second verse":

```
The wonderful thing about Tiggers
Is Tiggers are wonderful chaps!
They're loaded with vim and with vigor!
They love to leap into laps!
They're bouncy, trouncy, flouncy, pouncy,
Fun-fun-fun-fun-fun!
But the most wonderful thing about Tiggers is:
I'm the only one!
I'm the only one!
```

and he added:

```
Yes, there are a lot of exclamation points there, but Tigger
would never be content to simply state anything he could exclaim.

The song comes from Walt Disney's cartoon adaptations
of A.A. Milne's Winnie-the-Pooh books.
```

Thanks, Gary and Peter! :-)

Sven [990526,990726]

P2P Contributors
Rael Dornfest
Lucas Gonze
Richard Koman
Andy Oram
Tim O'Reilly
Jon Orwant
Clay Shirky
Kelly Truelove

Topics
Collaboration
Distributed Computation
Enterprise P2P
File-Sharing
Freenet
General
Gnutella
Hailstorm
JXTA
Instant Messaging
Napster
P2P Law
Security
Standards

Backlash!

by Clay Shirky
04/05/2001

The peer-to-peer backlash has begun. On the same day, the Wall St. Journal ran an article by Lee Gomes entitled "Is P2P plunging off the deep end?", while Slashdot's resident commentator, Jon Katz, ran a review of O'Reilly's *Peer to Peer* book under the title "Does peer-to-peer suck?"

It's tempting to write this off as part of the Great Wheel of Hype we've been living with for years:

New Thing happens; someone thinks up catchy label for New Thing; press picks up on New Thing story; pundits line up to declare New Thing "Greatest Since Sliced Bread." Whole world not transformed in matter of months; press investigates further; New Thing turns out to be only best thing since soda in cans; pundits (often the same ones) line up to say they never believed it anyway.

This quick reversal is certainly part of the story here. The Journal quoted entrepreneurs and investors recently associated with peer-to-peer who are now distancing themselves from the phrase in order to avoid getting caught in the backlash. There is more to these critiques than business people simply repositioning themselves when the story crescendos, however, because each of the articles captures something important and true about peer-to-peer.

Where's the money?

The Wall St. Journal's take on peer-to-peer is simple and direct: it's not making investors any money right now. Mr. Gomes notes that many of the companies set up to take advantage of file sharing in the wake of Napster's successes have hit on tough times, and that Napster's serious legal difficulties have taken the bloom off the file sharing rose. Meanwhile, the distributed computing companies have found it hard to get either customers or investors, as the closing of Popular Power and the difficulties of the remaining field in finding customers have highlighted.

Furthermore, Gomes notes that P2P as a label has been

Related Articles:

Editor Andy Oram on O'Reilly's First Peer-to-Peer Book

Security Concerns Miss the P2P Point

What Is P2P ... And What Isn't

Popular Power Turns Off the Lights

Peer-to-Peer Book

- Chap 3: *Remaking the P2P Meme*
- Chap. 13: *Metadata*
- Catalog Page

taken on by many companies eager to seem cutting edge, even those whose technologies have architectures that differ scarcely at all from traditional client-server models. The principle critiques Gomes makes -- P2P isn't a well-defined business sector, nor a well-defined technology -- are both sensible. From a venture capitalist's point of view, P2P is too broad a category to be a real investment sector.

Gnutella: Alive, Well, and Changing Fast

Gnutella and Freenet Represent True Tech Innovation

Articles by Clay Shirky

More from OpenP2P.com

Is P2P even relevant?

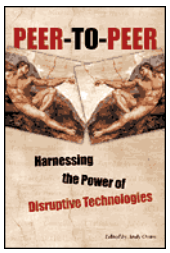
Jon Katz's complaints about peer-to-peer are somewhat more discursive, but seem to center on its lack of a coherent definition. Like Gomes, he laments the hype surrounding peer-to-peer, riffing off a book jacket blurb that overstates peer-to-peer's importance, and goes on to note that the applications grouped together under the label peer-to-peer differ from one another in architecture and effect, often quite radically.

Katz goes on to suggest that interest in P2P is restricted to a kind of techno-elite, and is unlikely to affect the lives of "Harry and Martha in Dubuque." While Katz's writing is not as focused as Gomes', he touches on the same points: there is no simple definition for what makes something peer-to-peer, and its application in people's lives is unclear.

The unspoken premise of both articles is this: if peer-to-peer is neither a technology or a business model, then it must just be hot air. There is, however a third possibility besides "technology" and "business." The third way is simply this: Peer-to-peer is an idea.

Revolution convergence

As Jon Orwant noted recently in these pages, "Peer-to-peer is not a technology, it's a mindset."" Put another way, peer-to-peer is a related group of ideas about network architecture, ideas about how to achieve better integration between the Internet and the personal computer -- the two computing revolutions of the last 15 years.



Peer-to-Peer: Harnessing the Power of Disruptive Technologies
Edited by Andy Oram
March 2001
0-596-00110-X, Order Number: 110X
448 pages, $29.95

The history of the Internet has been told often -- from the late '60s to the mid-'80s, the DARPA agency in the Department of Defense commissioned work on a distributed computer network that used packet switching as a way to preserve the fabric of the network, even if any given node failed.

The history of the PC has likewise been often told, with the rise of DIY kits and early manufacturers of computers for home use -- Osborne, Sinclair, the famous Z-80, and then the familiar IBM PC and with it Microsoft's DOS.

In an accident of history, both of those movements were transformed in January 1984, and began having parallel but increasingly important effects on the world. That month, a new plan for handling DARPA net addresses was launched. Dreamed up by Vint Cerf, this plan was called the Internet Protocol, and required changing the addresses of every node on the network over to one of the new IP addresses, a unique, global, and numerical address. This was the birth of the Internet we have today.

Meanwhile, over at Apple Computer, January 1984 saw the launch of the first Macintosh, the computer that popularized the graphic user interface (GUI), with its now familiar point-and-click interactions and desktop metaphor. The GUI revolutionized the personal computer and made it accessible to the masses.

Is P2P the "push" of the 2000s, or a really good idea? Respond to Shirky's opinion (or Katz's or Gomes') here.

Post your comments

For the next decade, roughly 1984 to 1994, both the Internet and the PC grew by leaps and bounds, the Internet as a highly connected but very exclusive technology, and the PC as a highly dispersed but very inclusive technology, with the two hardly

intersecting at all. One revolution for the engineers, another for the masses.

The thing that changed all of this was the Web. The invention of the image tag, as part of the Mosaic browser (ancestor of Netscape), brought a GUI to the previously text-only Internet in exactly the same way that, a decade earlier, Apple brought a GUI to the previously text-only operating system. The browser made the Internet point-and-click easy, and with that in place, there was suddenly pressure to fuse the parallel revolutions, to connect PCs to the Internet.

Which is how we got the mess we have today.

First and second-class citizens

In 1994, the browser created sudden pressure to wire the world's PCs, in order to take advantage of the browser's ability to make the network easy to use. The way the wiring happened, though -- slow modems, intermittent connections, dynamic or even dummy IP addresses -- meant that the world's PCs weren't being really connected to the Internet, so much as they were being hung off its edges, with the PC acting as no more than a life-support system for the browser. Locked behind their slow modems and impermanent addresses, the world's PC owners have for the last half-dozen years been the second-class citizens of the Internet.

Anyone who wanted to share anything with the world had to find space on a "real" computer, which is to say a server. Servers are the net's first-class citizens, with real connectivity and a real address. This is how the Geocities and Tripods of the world made their name, arbitraging the distinction between the PCs that were (barely) attached to the networks edge and the servers that were fully woven into the fabric of the Internet.

Big, sloppy ideas

Rejection of this gap between client and server is the heart of P2P. As both Gomes and Katz noted, P2P means many things to many people. PC users don't have to be second-class citizens. Personal computers can be woven directly into the Internet. Content can be provided from the edges of the network just as surely as from the center. Millions of small computers, with overlapping bits of content, can be more reliable than one giant server. Millions of small CPUs, loosely coupled, can do the work of a supercomputer.

These are sloppy ideas. It's not clear when something stops being "file sharing" and starts being "groupware." It's not clear where the border between client-server and peer-to-peer is, since the two-way Web moves power to the edges of the network while Napster and ICQ bootstrap connections from a big server farm. It's not clear how ICQ and SETI@Home are related, other than deriving their power from the network's edge.

No matter. These may be sloppy ideas, ideas that don't describe a technology or a business model, but they are also big ideas, and they are also good ideas. The world's Net-connected PCs host, both individually and in aggregate, an astonishing amount of power -- computing power, collaborative power, communicative power.

Our first shot at wiring PCs to the Internet was a half-measure -- second-class citizenship wasn't good enough. Peer-to-peer is an attempt to rectify that situation, to really integrate personal devices into the Internet. Someday we will not need a blanket phrase like peer-to-peer, because we will have a clearer picture of what is really possible, in the same way the arrival of the Palm dispensed with any need to talk about "pen-based computing."

In the meantime, something important is happening, and peer-to-peer is the phrase we've got to describe it. The challenge now is to take all these big sloppy ideas and actually do something with them, or, as Michael Tanne of XDegrees put it at the end of the Journal article:

> "P2P is going to be used very broadly, but by itself, it's not going to create new companies. ...[T]he companies that will become successful are those that solve a problem."

Clay Shirky is a Partner at The Accelerator Group. He writes extensively about the social and economic effects of the internet for the O'Reilly Network, Business 2.0, and

FEED.

Related Articles:

Editor Andy Oram on O'Reilly's First Peer-to-Peer Book

Security Concerns Miss the P2P Point

What Is P2P ... And What Isn't

Popular Power Turns Off the Lights

Gnutella: Alive, Well, and Changing Fast

Gnutella and Freenet Represent True Tech Innovation

Articles by Clay Shirky

Is P2P the "push" of the 2000s, or a really good idea? Respond to Shirky's opinion (or Katz's or Gomes') here.
(You must be a member of the O'Reilly Network to use this feature.)*
Comment on this Article

Full Text Titles Only Newest First

- a rose by any other name...
  2001-04-06 13:31:24 shelleyp
- Katz if off
  2001-04-06 14:54:56 thewah
- State of mind
  2001-04-06 21:50:13 vijaybasrur
- criticizing P2P architecture
  2001-04-09 10:45:54 arnoldsk
- "distributed" is less controvercial than "p2p"
  2001-05-02 16:52:34 dstolarz@static.com

- Search
- Product List
- Press Room
- Jobs

Perl
Java
Web & Internet
Open Source
XML
Linux
Unix
Python
Macintosh
Windows
.NET
Oracle
Security
Sys/Network Admin
C/C++ Programming
Design & Graphics
Visual Basic

Ask Tim
Frankly Speaking
Ron's VB Forum
Beta Chapters

Hot off the Press!

[New & Upcoming Titles]

Dreamweaver 4: The Missing Manual is a complete user's guide to Macromedia Dreamweaver. This book starts by analyzing the structure of a typical Web page, then takes users step by step through the process of creating a Web site with Dreamweaver 4, including site structuring and testing. This Missing Manual also shows how to customize Dreamweaver with libraries, templates, shortcuts, and extensions. **Sample Chapter 17, Libraries and Templates**, is available online in PDF format.

Exim: The Mail Transfer Agent is the official guide to Exim, written by its creator, Philip Hazel. Exim is the open source, default mail-transport agent installed on some Linux systems, and it runs on many versions of Unix. This book offers a comprehensive guide to Exim's configuration and syntax as well as to its many features, including Sendmail compatibility, lookups in LDAP servers, MySQL and PostgreSQL databases, and NIS or NIS+ services. *Exim: The Mail Transfer Agent* also delves into complicated areas such as virtual hosting, filtering, and automatic replies. **Sample Chapter 3, Exim Overview**, is available online.

Managing NFS and NIS, 2nd Edition, is the only Unix networking book devoted entirely to NFS and the distributed database NIS. This new edition has been updated for NFS Version 3 and is based on Solaris 8. If you manage a network of Unix systems, or want to set up a Unix network, *Managing NFS and NIS* shows you what to do. This book provides details on how to plan and debug a network, how to manage important administrative files, such as the *passwd* and *hosts* files, and more. **Sample Chapter 15, Debugging Network Problems**, is available online.

Web Caching provides the technical information you need to design, deploy, and operate an effective Web caching service. This book starts with the basics of how Web caching works, then moves into topics ranging from configuring Web browsers and servers to monitoring and fine-tuning cache performance. *Web Caching* also covers the political aspects of caching, including privacy, intellectual property, and security issues. **Sample Chapter 5, Interception Proxying and Caching**, is available online.

News

[News Archive]

An Interview with Jesse Liberty: Author of Programming C# --We asked Jesse how C# compares with other object-oriented programming languages and how C# fits into the .NET Framework. To learn more, don't miss O'Reilly's upcoming release, **Programming C#**.

Visit our Beta Chapters site for a preview of upcoming books. Featured titles include *Programming C#*, *Palm OS Network Programming*, *Learning Perl, 3rd Edition*, *Programming ColdFusion*, *Network Troubleshooting Tools*, and *Malicious Mobile Code*.

XML-RPC Serves Up Binary Data to Go--O'Reilly software engineer Joe Johnston demonstrates how to use XML-RPC to transport binary data as base64-encoded messages and how to set up a simple Web service that creates charts with this data. Joe is also coauthor of O'Reilly's recently released **Programming Web Services with XML-RPC**.

GNOME 's Miguel de Icaza on .NET--In a conversation with O'Reilly Network publisher Dale Dougherty, GNOME founder Miguel de Icaza discusses the advantages of the .NET development environment and why the open source community should join him in developing Mono--a free implementation of .NET. Miguel will speak about these topics at this month's **O'Reilly Open Source Convention**.

Perl Runs Sweden's Pension System--Intended only as a quickly developed backup for Sweden's Premium Pension System, the Perl-based application proved to be faster and more promising than the original commercial version--at a fraction of the cost. To get the latest on Perl from the experts, don't miss this month's **O'Reilly Perl Conference 5**.

IEE Call for Papers: Open Source Software Engineering--The Institution of Electrical Engineers

Letters
elists
Events
Palm OS
Missing Manual
User Groups
Catalog Request
Specials
Write for Us

O'REILLY

Linux Device Drivers, 2nd Edition, shows step by step how to write a driver for character devices, block devices, and network interfaces, with examples you can compile and run. This book covers kernel version 2.4 but also includes information back to kernel 2.0. This second edition adds discussions of symmetric multiprocessing and locking, new CPUs and hardware platforms, and recently supported buses, including universal serial bus. *Linux Device Drivers* is the guide for anyone who wants to support computer peripherals under the Linux operating system. **Sample Chapter 3, Char Drivers**, is available online.

Programming Web Services with XML-RPC introduces XML-RPC, a system for remote procedure calls built on XML and HTTP that lets developers connect computers running programs written in different languages on different operating systems. This book covers five XML-RPC implementations: Java, Perl, Python, ASP, and PHP--with code examples for each--so you can start developing your own distributed applications. The book also provides in-depth coverage of the XML-RPC specification. And if you want to build an XML-RPC implementation for another environment, the explanations in this book will serve as a foundation. **Sample Chapter 3, Client-Server Communication: XML-RPC in Java**, is available online.

Java Cookbook is filled with short, focused pieces of code--designed to be useful, tricky, or both--that Java developers can easily incorporate into other programs. The book's code segments cover all of the dominant APIs and many specialized APIs, including media and servlets. *Java Cookbook*'s comprehensive collection of problems, solutions, and practical examples serves as an excellent jumping-off place for Java developers who want to get started in areas outside their specialization. **Sample Chapter 18, Web Server Java: Servlets and JSP**, is available online.

ADO: ActiveX Data Objects is both an introduction and a complete reference to programming with ADO, Microsoft's universal data-access solution. Covering ADO through version 2.6, this book includes chapters on the Connection, Recordset, Field, and Command objects; the Properties collection; ADO architecture; data shaping; and the ADO Event Model. The book also provides brief introductions to RDS, ADO.NET, and SQL; and it's full of code examples illustrating timesaving programming tips. **Sample Chapter 3, Accessing ADO with Various Languages**, is available online.

Subclassing & Hooking with Visual Basic--Subclassing and hooking allow VB 6 and VB.NET developers to manipulate messages

is soliciting papers representing original, completed research on open source software engineering, especially papers providing a rigorous analysis of open source methods and tools. A great place to talk over your ideas would be this month's **O'Reilly Open Source Convention**.

Filter Code with Java Servlet 2.3 Model--In this *JavaWorld* article, Jason Hunter takes an in-depth look at the new 2.3 servlet filter model. Jason coauthored O'Reilly's **Java Servlet Programming, 2nd Edition**.

Blackened Network Monitors--How do you ensure that the target of a network-intrusion investigation cannot detect being monitored? Kenneth van Wyk provides an overview of the best techniques for configuring a blackened, or undetectable, network monitor. Kenneth is coauthor of O'Reilly's upcoming security book on **Incident Response**.

Moving from C++ to C#: What You Need to Know--In this *MSDN Magazine* article, Jesse Liberty discusses the issues involved in making the transition from C++ to C#, including how .NET's managed environment translates to less programmer control in C#. Jesse is the author of O'Reilly's upcoming **Programming C#**.

Wisdom from the Llama--Coauthor Tom Phoenix lets you in on, among other things, the top five jokes that got cut from the third edition of **Learning Perl**.

Where do I begin as a Webmaster? Ron Hunsberger asked Tim O'Reilly for his perspective on what programming languages and software to learn. Rather than recommending specific technologies, Tim offers Ron four fundamental questions to consider when deciding where to begin. Tim also includes a list of useful books.

The Mind Behind gawk: An Interview with Arnold Robbins--O'Reilly Open Source editor Chuck Toporek talks with Arnold Robbins about his involvement with gawk, publishing his book under the FSF's Free Documentation License, and what's new in his latest O'Reilly book, **Effective awk Programming, 3rd Edition**.

Eric Raymond on VA Linux's Change of Course--VA Linux shocked the Linux community recently when it announced it would get out of the hardware business, lay off 35 percent of its staff, and refocus on software and its Web properties. In this

bound for objects within the Windows operating system, giving developers the means to customize Windows behavior. However, Windows is unforgiving if these techniques are used incorrectly. *Subclassing & Hooking with Visual Basic* demonstrates the various techniques for intercepting and modifying messages as well as the pitfalls to avoid. Developers can use subclassing and hooking techniques to determine when an application is idle, to create an automated testing application, and much more. **Sample Chapter 1, Introduction**, is available online.

.NET Framework Essentials provides intermediate to advanced VB, C/C++, Java, and Delphi developers with a complete technical overview of Microsoft's .NET Framework. This book takes on all the most important topics, from the underlying Common Language Runtime (CLR) to its specialized packages for ASP.NET, Web Forms, Windows Forms, XML, and data access (ADO.NET). The book also covers developing .NET components and Web services, and details each of the major .NET languages, including VB.NET, C#, and Managed C++, with working code samples for each language. **Sample Chapter 6, Web Services**, is available online.

Linux Today interview, Eric Raymond says Linux's success is partially to blame for these changes.

USENIX Dispatches--A team of O'Reilly editors descended upon the annual USENIX Technical Conference in Boston and reported back on what they learned. If you couldn't be there, check out these notes and observations from the conference floor.

Open Source Bibliography V4.0--Open source is changing the nature of the software industry, but how do you find the right books to guide the way? We've just published an updated version of the ever-popular O'Reilly Open Source Bibliography. It lists the very best books, including those by other publishers.

An LPI Level 1 Crash Course--If you're planning to take the Linux Professional Institute (LPI) Level 1 exams, or just want to brush up on your knowledge of Linux, Jeffrey Dean highlights the most important areas to review. Jeffrey is the author of **LPI Linux Certification in a Nutshell**.

O'Reilly's Upcoming Titles include *Java & XML, 2nd Edition*, *Learning Perl, 3rd Edition*, *Programming C#*, *XSLT*, *Malicious Mobile Code*, *Dreamweaver 4: The Missing Manual*, and *Programming ColdFusion*.

webmaster@oreilly.com

Sites
ONJava.com
ONLamp.com
openp2p.com
Perl.com
XML.com

Subjects
Apache
BSD
Javascript and CSS
Linux
Mac
Mozilla
.NET
Perl
Policy
PHP
Python
Web Services
Wireless
XML

Using PASX, an open source solution.

How will Mono unite the ADO.NET and GNOME-DB object models?

Maybe socialism wasn't such a bad idea after all...

TODAY ON THE O'REILLY NETWORK

Extending the Life Line of the PalmOS Marc Hedlund offers four ways to improve the Palm operating system, including bundling applications, improving the desktop software, switching wireless support to 802.11b, and embracing standards. [Wireless DevCenter]

Use P2P, Go to Jail David McOwen installed the Distributed.Net client at the college where he worked. Now he's being prosecuted for "computer trespass" and could face up to 15 years in prison. [OpenP2P.com]

X on X Because Mac OS X is based on Darwin, it's possible to shut down Core Graphics and install XFree86 to access your X11-based programs. Here's one BSD user's experience. [Mac DevCenter]

What's New

Use P2P. Go to Jail. Any Questions?
David McOwen faces arrest, trial, and imprisonment because he installed the Distributed.net client on computers at the college where he worked. Read the story on OpenP2P.com.

WEBLOGS
Links and commentary

Find Search
Features
Weblogs
O'Reilly Book Excerpts
Meerkat News
FAQs
Forums
All Articles
Audio Roundtable

Add the latest news from oreillynet.com to your web site
Click here!

Affiliates Content partners who bring you in-depth coverage of the subjects that matter most:

Apache Week
jdom.org
MySQL.com
Servlets.com
Wireless Developer Network

About the O'Reilly Network
Work With Us!

GNOME's Miguel de Icaza on .NET Open Source developer Miguel de Icaza, leader of the GNOME project and founder of Ximian, has been exploring Microsoft's .NET platform with an open mind. Find out why he claims that .NET is "the new development environment for the next 20 years." [.NET DevCenter]

Security Alerts: PHP Weaknesses? Noel Davis shows us a correction to the report on the AIX rsh buffer overflow; buffer overflows in Solaris' whodo, and UnixWare's su, uucp, and crontab packages, and xvt; temporary file symbolic link race condition vulnerabilities in Red Hat's LPRng, and Red Hat's crontab; problems in Poprelayd, PHP Safe mode, ePerl, 802.11b Access Points, Gnatsweb, SquirrelMail, and phpMyAdmin; and a paper on common PHP vulnerabilities. [Linux DevCenter]

This Week on p5p 2001/07/09 No 5.8.0 yet, numeric hackery, worries about PerlIO and much more. [Perl News and Features from www perl com]

The WeakHashMap Class WeakHashMap is a type of Map which differs from other Maps in more than just having a different implementation. WeakHashMap uses weak references to hold its keys, making it one of the few classes able to respond to the fluctuating memory requirements of the JVM. [ONJava.com]

XML-RPC Serves Up Binary Data to Go --O'Reilly software engineer Joe Johnston demonstrates how to use XML-RPC to transport binary data as base64-encoded messages and how to set up a simple Web service that creates charts with this data. Joe is also coauthor of O'Reilly's recently released *Programming Web Services with XML-RPC*. [oreilly.com]

Morpheus Out of the Underworld Napster last week disabled all previous versions of the software. So where have all file-sharers gone? Would you believe Morpheus -- a system that has far surpassed not only the hobbled Napster but also Gnutella. Clip2 takes an in-depth look at the Morpheus system. [openp2p.com]

Is All Music File-Sharing Piracy? Morpheus is looking like the heir apparent for the now-unusable Napster. Being decentralized, perhaps Morpheus is more resistant to RIAA lawsuits than Napster. Perhaps

Lisa Rein's Weblog

Turn About Is Fair Player
How Microsoft's self-defeating media format and digital rights management strategy is helping rivals AOL Time Warner and RealNetworks level the playing field. [O'Reilly Network Weblogs: P2P]

More Weblogs

If .NET is the OS, will the Internet bluescreen? [Lucas Gonze]

Installing .NET changes MSIE User-Agent [Marc Hedlund]

Worst Java Fears Confirmed in Latest Court Ruling Against Microsoft? [Steve Anglin]

Turn About Is Fair Player [Lisa Rein]

P2P Networks Becoming Accepted by

user name:

password:

Login to customize Meerkat News, change newsletter options, and view your O'Reilly conference registrations.

- New Users Guide
- Register
- Lost your password?
- Privacy Policy

not. In any case, the music industry should realize that -- online and offline -- there are legit uses of copyright material that don't require a payment. [O'Reilly Network]

Running Java Applications on Mac OS X You can easily create double-clickable versions of Java applications for Mac OS X using MRJAppBuilder. Daniel Steinberg shows you how. [Mac DevCenter]

Extending Dreamweaver: Let Dreamweaver Create Your Menus Create a tree-style menu for your site's navigation in Dreamweaver. Part three of a three-part series. [Javascript and CSS DevCenter]

Extending the Life Line of the PalmOS Marc Hedlund offers four ways to improve the Palm operating system, including bundling applications, improving the desktop software, switching wireless support to 802.11b, and embracing standards. [O'Reilly Network]

XML Processing with TRaX The TRaX API extends JAXP to include XML transformations, providing a vendor- and implementation-agnostic standard Java API for specifying and executing XML transformations. [ONJava.com]

GUI Development with Python and Qt A change in Qt's license for Windows and an open book by Boudewijn Rempt makes Qt an attractive alternative to Tk for Python developers. [Python DevCenter]

Monitoring ipfw Logs Dru Lavigne shows us how to monitor ipfw logs and more importantly how to deal with what we find. [BSD DevCenter]

Is All Music File-Sharing Piracy? Morpheus is looking like the heir apparent for the now-unusable Napster. Being decentralized, perhaps Morpheus is more resistant to RIAA lawsuits than Napster. Perhaps not. In any case, the music industry should realize that -- online and offline -- there are legit uses of copyright material that don't require a payment. [O'Reilly Network]

not. In any case, the music industry should realize that -- online and offline -- there are legit uses of copyright material that don't require a payment. [O'Reilly Network]

O'REILLY NETWORK DEVCENTERS ----------

APACHE Articles | Forums | FAQs

the Mainstream [Steve McCannell]

Distributed Net Value [Rael Dornfest]

The Mono Project [Brian Jepson]

Should we build an authentication system on open source? [Tim O'Reilly]

Morpheus Vs. Gnutella [Lucas Gonze]

OS X and Lucent WaveLan 802.11b [Rael Dornfest]

Error .../CmsInit.ASP
I have a problem with the following line. it's written down every few seconds in the two files httpd.error_log and httpd.access_log: in the first file "file does not exist: /usr/local.../scripts/cm...
oreillynet.apache

How do I make a

- Under Development
- Apache Conferences
- In the news
- Featured Articles
- Under Development

More ▶

BSD

Articles | Forums

- Monitoring ipfw Logs
- Controlling User Logins
- IPFW Logging
- Rotating Log Files
- BSD Tricks: CVS

More ▶

JAVA

Articles

- Using PASX
- The WeakHashMap Class
- Java and XML Week
- XML Processing with TRaX
- JSP Security for Limiting Access to Application-Internal URLs

More ▶

JAVASCRIPT and CSS

Articles

- Cross-Browser Layers: Part 2
- Extending Dreamweaver with its JavaScript API
- Accessing Dreamweaver's JavaScript API
- Cross-Browser Layers, Part One
- JavaScript: Why You Don't Know More About It

More ▶

Virtualhost with .htaccess and mod_rewrite only

I have two domains parked at my host pointing to the same place. I want it to point somewhere else. domaina.com -> domainb.com/folder or domaina.com -> /home/enricong/public_html/folder I dont get acc... oreillynet.apache

POST error

I have set up Apache 1.3.18 on a RedHat Linux system and am trying to write a script to do a POST to a page. When doing so, I get the following error: The requested method POST is not allowed. How ... oreillynet.apache

Other Forums ▶

TOP FAQs

LINUX

Articles | Forums | FAQs

- Moving to BSD Without Leaving Linux
- Tools of the Trade: Part 2
- Tools of the Trade: Part 1
- Death of a Palm
- Linux Device Drivers Update

More ▶

MAC

Articles

- X on X
- Running Java Applications on Mac OS X
- How I BSDed My iBook
- Strings in Cocoa: Part I
- VisorVision at PC Expo

More ▶

MOZILLA

Articles | Forums

- July 27th is Mozilla Community Day at the O'Reilly Open Source Conference!
- Mozilla 0.9.2 Released
- Towards Mozilla 1.0
- 0.9.2 Branch and Beyond
- Netscape 6.1 Beta 1

More ▶

.NET

Articles

- Writing ASP.NET Web Forms with C#
- Contrasting C# and Java Syntax
- Comparing C# and Java
- Conversational C# for Java Programmers, Part 1

More ▶

P2P

Articles

Linux

What good is a core file?

What is the difference between internal and external modems?

XML

Why should I use XML instead of HTML?

- Use P2P, Go to Jail. Any Questions?
- Use P2P, Go to Jail
- Open Standards/Closed Mind
- Who is Microsoft Trying to Control?
- Passport is Evil

More ▶

POLICY

Articles | Forums

- Internet Society Panel on Business Method Patents
- British Telecom Patent: Lachey, Uninfringed and Invalid?
- Who's Really Being Protected?
- Who's Really Being Protected? - Part 2
- A Few More Thoughts on the Patents Issue by Tim O'Reilly

More ▶

PERL

Articles | FAQs

- This Week on p5p 2001/07/09
- People Behind Perl: Nathan Torkington
- This Fortnight in Perl 6 (17 - 30 June 2001)
- This Week on p5p 2001/07/02
- Why Not Translate Perl to C?

More ▶

PHP

Forums

- Embedding PHP in HTML
- The PHP Configuration File -- Part One
- The Universal Web Form Processor
- Installing PHP on a Windows System
- PHP With HTML Forms

More ▶

PYTHON

Articles | Forums

- GUI Development with Python and Qt
- Piddle Graphics Online
- Transactional Persistence for Python
- Using PySOAP
- Programming for Artists

More ▶

WIRELESS

Articles | Forums

- Extending the Life Line of the Palm OS
- 802.11b Round-Up
- Developing Applications for the RIM BlackBerry
- New Wireless Standards Challenge 802.11b
- Introducing the Xircom 802.11 Module for the Visor

More ▶

XML

Articles | FAQs

- Namespace Nuances
- Against the Grain
- Math and XSLT
- Blueberry Jam
- Electronic Publishing with XML

More ▶

RESOURCE CENTERS
Business
Graphics
Metadata
Mobile
Programming
Protocols
Schemas
Style
Web

ESSENTIALS
Annotated XML
What is XML?
What is XSLT?
What is XLink?
What is XML Schema?
What is RDF?

Manage Your Account
Forgot Your Password?

FIND
Search
Article Archive
FAQs

COLUMNS
XML-Deviant

Sponsored By:

Transforming XML: Math and XSLT
By Bob DuCharme
XSLT is primarily for transforming text, but you can use it to do basic math too. *Jul. 5, 2001*

XML Q&A: Namespace Nuances
By John E. Simpson
This month's Q&A column tackles the question of how to write DTDs for XML applications that use namespaces. *Jul. 5, 2001*

XML-Deviant: Against the Grain
By Leigh Dodds
XML developers are talking about a perennial question: how can XML and database technologies be integrated appropriately? *Jul. 5, 2001*

Previous Features

Electronic Publishing with XML
By Benjamin Jung, John McKeown
The proceedings for the recent XML Europe 2001 conference were created from start to finish with XML. This case study describes the processes used and problems encountered. *Jun. 27, 2001*

XML on the Cheap
By Edd Dumbill
If you're new to XML, or simply want a to play around with it a little, there are plenty of resources on the Web you can use for free, many without even installing software on your computer. *Jun. 27, 2001*

Storing XML in Relational Databases
By Igor Dayen
A survey of the techniques used by the major vendors to store XML in their databases, and a proposition for a database-independent XML

XML News

- SOAP Version 1.2 *[XML.com Resource Guide]*
- W3C discussion of Technical Architecture Group *[xmlhack]*
- SVG update *[xmlhack]*
- Apache has mixed feelings about XML-RPC *[xmlhack]*
- A light pUDDIng *[xmlhack]*
- RDFStore *[XML.com Resource Guide]*
- Run XSLT transforms from Emacs *[xmlhack]*
- Software AG and Fatdog release XQuery implementations *[xmlhack]*
- JAX-RPC *[XML.com Resource Guide]*
- J2EETM COM Bridge *[XML.com Resource Guide]*

More ▶

XML News Headlines from The XML Cover Pages by Robin Cover
Sponsored by OASIS

- W3C Publishes XML Protocol Abstract Model and Glossary.
- W3C Releases First Public Working Draft for SOAP Version 1.2.
- New Release of XML Schema Validator

Events More ▶

XML Summer School 2001 [Jul. 20, 2001]

O'Reilly Open Source Convention [Jul. 23, 2001]

Free Web Seminar: Managing the Development of XML E-Business Initiatives [Jul. 24, 2001]

SD Web Services World [Aug. 27, 2001]

XML World 2001 [Sep. 17, 2001]

Style Matters
XML Q&A
Transforming XML
Perl and XML

GUIDES
XML Resources
Buyer's Guide
Events Calendar
Standards List
Submissions List

TOOLBOX
Syntax Checker
XML Testbed

framework. *Jun. 20, 2001*

Rapid Resolution
By Leigh Dodds
A recent debate about supporting OASIS catalogs in XML shows that strong differences of opinion still exist on interpretation of the XML 1.0 specification itself. *Jun. 20, 2001*

FAQs

How can I advertise on XML.com?

Whom should I contact if I have an article idea for XML.com?

What information does XML.com gather and track?

(XSV).
• SWIFT and FPL Agree to Develop Securities Standard 'ISO 15022 XML' in ISO Working Group 10...
• Software AG Releases XQuery Prototype 'QuiP'.
• Updated XEP Rendering Engine Supports Enhanced XSL FO Formatting.
• Initial Release of a RELAX NG Working Draft Specification.
• XML Schema for 'Beta' Metadata Encoding and Transmission Standard (METS).

More

Buyer's Guide More

Document Authoring - Sponsored by SiberSafe: XML CMS
Editors Graphics Style Tools

Website Tools
Content Exchange/Syndication Information Servers

Ecommerce
Application Builders EDI

Data Management
Application Generator Data Analysis Schema/DTD Editors Search Tools Tree Viewers

Databases/Repositories - Sponsored by SiberSafe: XML CMS

Print Production

Training
Books Courses

B2B Integration Strategies & Solutions [Sep. 17, 2001]

What's New

U.S. Helps Fund FreeBSD Security Project

Continuing its support of open-source operating systems, the U.S. Department of Defense granted $1.2 million to a community project aimed at adding advanced security features to FreeBSD, an open-source variant of Unix. Get more detail in the Robert Lemos article on News.com.

Articles

- Linux
- Apache
- MySQL
- Perl
- PHP
- Python
- BSD

Essentials

X on X: XFree86 on your Mac.

Monitor and adjust your firewall.

Gui Development with Python and Qt

How I BSDed My iBook

It's true that Mac OS X has BSD under the hood. But what if you wanted to load pure BSD? Chris Coleman explains what happens when you take a Tangerine iBook, add a little NetBSD, and shake vigorously. [Mac DevCenter]

Security Alerts: PHP Weaknesses?

Noel Davis shows us a correction to the report on the AIX rsh buffer overflow; buffer overflows in Solaris' whodo, and UnixWare's su, uucp, and crontab packages, and xvt; temporary file symbolic link race condition vulnerabilities in Red Hat's LPRng, and Red Hat's crontab; problems in Poprelayd, PHP Safe mode, ePerl, 802.11b Access Points, Gnatsweb, SquirrelMail, and phpMyAdmin; and a paper on common PHP vulnerabilities. [O'Reilly Network]

This Week on p5p 2001/07/09

No 5.8.0 yet, numeric hackery, worries about PerlIO and much more. [Perl.com]

Getting Flashy With PHP

Looking to add animation to your website? W. J. Gilmore shows you how to add dynamic Flash animations to your website using PHP. [PHP DevCenter]

Piddle Graphics Online

Michal Wallace uses the source to put piddle graphics online. Combine piddle with the shelve and cgi modules to create an interactive gantt chart. [Python DevCenter]

People Behind Perl: Nathan Torkington

So you use Perl, and you probably know that it was brought to you by "Larry Wall and a cast of thousands". But do you know these people that make up the Perl development team? Simon Cozens talks to Nathan Torkington, a long-time Perl developer and a mainstay of the Perl community. [www perl com]

SAMBA Remote Root Exploit

Noel Davis shows us buffer overflows in the GazTek HTTP Daemon, Solaris Printer Daemon, and w3m; a problem in default SAMBA installations that can be used to gain root access; and problems in Cisco 6400 NRP2, udirectory, Tarantella, Oracle 8i SQLNet, Formmail.pl, OS X directory permissions, and kdesu. [Linux DevCenter]

Why Not Translate Perl to C?

Mark-Jason Dominus explains why it might not be any faster to convert

- What is LAMP?
- The Best of ONLamp.com
- aboutSQL
- Big Scary Daemons
- FreeBSD Basics
- HTTP Wrangler
- Linux in the Enterprise
- Linux Network Administration
- The Linux Professional
- Perl P5P Digest Archive
- PHP Admin Basics
- PHP Phanatics
- Python_News
- Security Alerts

your code to a C program rather than let the Perl interpreter execute it. [Perl News and Features from www perl com]

Yet Another YAPC Report: Montreal

Schuyler Erle gives a detailed report of all the exciting events at this year's Yet Another Perl Conference in Montreal. By his account, it appears to be an exciting time to be involved with the development of Perl. [www perl com]

This Fortnight in Perl 6 (17 - 30 June 2001)

A detailed summary of a recent Perl vs. Java battle, a discussion on the internal API for strings, and much more. [www perl com]

Controlling User Logins

Michael Lucas explains how to restrict shell access to a system using as few system resources as possible. [BSD DevCenter]

The Evolution of SOAP::LITE

SOAP functionality is growing in leaps and bounds. Paul Kulchenko, a speaker at the upcoming O'Reilly Open Source Convention, talks about his experiences in the early days of SOAP and the directions he sees it going. [ONLamp.com]

Net Surfing With IP Protocol

A look at the routed protocol that not only ushered in a new revolution in the 1990s, but also drove the stock market wild -- the IP protocol. [O'Reilly Network]

The Outer Limits of SQL JOINs

In the last column, John Paul Ashenfelter showed you how to use inner joins and cross joins. This week he tackles outer joins. [ONLamp.com]

Transactional Persistence for Python

The Z Object Database combines the simplicity of shelve with the transactional power of a relational database. Michael Pelletier's tutorial will get you started. [Python DevCenter]

Linux Compatibility on BSD for the PPC Platform: Part 4

Emmanuel Dreyfus explains difficulties discovered in porting the Linux compatibility layer to run the Java Virtual Machine. [ONLamp.com]

Topics

Sys Admin

More

- Moving to BSD Without Leaving Linux
- Tools of the Trade: Part 2
- Controlling User Logins
- Tools of the Trade: Part 1
- IPFW Logging

Security

More

- Security Alerts: PHP Weaknesses?
- Monitoring ipfw Logs
- SAMBA Remote Root Exploit
- AIX Remote Root Exploit
- Tools of the Trade: Part 1

Database

More

News

Dr. Dobb's Tcl-URL! for July 10th[Linux Weekly News]

Stable kernel prepatch 2.4.7pre6[Linux Weekly News]

LinuxPlanet: .comment: A Moderate Approach to Intellectual Property[Linux Today]

Damian Conway in Portland 31 Jul - 3 Aug[Perl News]

Linux Kernel 2.4.7-pre6[AppWatch]

SML/NJ 110.34[AppWatch]

zebot, a modular irc bot in perl[SourceForge New Releases]

Cross-Language Implementation mailing list[lambda News]

Language Independent Arithmetic (LIA-2)[lambda News]

HTML::Embperl 2.0b3 (Development)[Freshmeat Daily News]

ARTICLE: The Python Web Services developer, part 4[SOAP Webservices Resource Center]

Apache-ASP-2.19[CPAN Uploads]

ARTICLE: Clean up your wire protocol with SOAP, Part 4[SOAP Webservices Resource Center]

OS X Gets Open-Source Porting System[RootPrompt.org --

Administration

Unix

Web and Internet

O'Reilly Network Technologies:

Sites
ONJava.com
ONLamp.com
openp2p.com
Perl.com
XML.com

Subjects
Apache
BSD
Javascript and CSS
Linux
Mac
Mozilla
.NET
Perl
Policy
PHP
Python
Web Services
Wireless
XML

- Types of JOINs
- PostgreSQL's Multi-Version Concurrency Control
- An Introduction to PEAR
- JOINed at the Hip
- Types of Relationships

Programming More ▶

- Getting Flashy With PHP
- Linux Compatibility on BSD for the PPC Platform: Part 4
- Arrays in PHP: Part 2
- Bootstrapping the YOPY PDA
- PHP and Java

Application Development More ▶

- Linux Compatibility on BSD for the PPC Platform: Part 3
- Modifying a Port
- Insecure Temporary File Functions
- IBM Websphere, Shockwave Flash, and emacs Advisories
- PalmOS, Half-Life Server, and Ethereal Vulnerabilities

Nothing but Unix]

A Summer Potpourri of Tools and Books[RootPrompt.org -- Nothing but Unix]

Konqueror Does ActiveX[RootPrompt.org -- Nothing but Unix]

System Performance Tuning[RootPrompt.org -- Nothing but Unix]

PHP Weaknesses?[RootPrompt.org -- Nothing but Unix]

Abit Siluro MX400[RootPrompt.org -- Nothing but Unix]

Secure C Programming[RootPrompt.org -- Nothing but Unix]

Jabber.com's Don Bergal Interviewed on The Linux Show[JabberCentral Recent Jabber News]

Linux Kernel pre-patch 2.4.7-pre6[IceWalkers]

Yerba WebSoftware 100701 (Default)[Freshmeat Daily News]

TWIG 2.7.3 (Default)[Freshmeat Daily News]

LinuxProgramming: Tcl-URL! - weekly Tcl news and links (Jul 10)[Linux Today]

More ▶

Using PASX, an open source solution.

Get memory flexibility with this map on the JVM.

Java Programming on the Mac: Running Java Applications on Mac OS X
By Daniel H. Steinberg
You can easily create double-clickable versions of Java applications for Mac OS X using MRJAppBuilder. Daniel Steinberg shows you how. *Jul. 6, 2001*

Java and XML Week
By Steve Anglin
This week, we focus on Java and XML as essential for business-to-business information interchange, synchronous data messaging, and business service objects, as well as content generation in the context of Java application and Web services development. *Jul. 3, 2001*

XML Processing with TRaX
By Craig Pfeifer
The TRaX API extends JAXP to include XML transformations, providing a vendor- and implementation-agnostic standard Java API for specifying and executing XML transformations. *Jul. 2, 2001*

Developing Applications for the RIM BlackBerry
By P.V. Subramanian
Research in Motion markets its BlackBerry keyboard pager as an email tool for executives. But it's also a fully programmable handheld computer. This article describes the BlackBerry's hardware and software, the wireless protocols it uses, and how to develop a sample application. Source code for a Hangman game is included. *Jun. 22, 2001*

Java Security: JSP Security for Limiting Access to Application-Internal URLs
By Jamie Jaworski
Jamie Jaworski covers a technique for designing and building simple JSP applications, which

what's new

Java.NET (Interoperability)

On the Java front, Halcyon Software is enabling developers to either migrate their ASP or Visual Basic code to JSP or Java, respectively, or to deploy .NET applications on Java-based infrastructures.

Stay informed. Subscribe to our weekly ONJava newsletter.

Your Email

Submit

weblogs

in Its First Year

provides some security benefits such as limiting access to application-internal URLs. *Jun. 27, 2001*

EJB 2: EJB 2 and J2EE Packaging
By Tyler Jewell
Tyler Jewell is back to discuss some of the nuances associated with J2EE packaging and provides some hints to make you more productive. *Jun. 26, 2001*

Using Ant and WebLogic EJBs
By Jesse E. Tilly
EJBs are complex. The steps required just to deploy them for Web application servers are difficult. But for developers using BEA WebLogic, life is much easier, thanks to Ant. *Jun. 25, 2001*

Developing a Simple JMS Example
By David Chappell
This JMS book excerpt (chapter) provides a gentle introduction to JMS using the publish-and-subscribe messaging model. *Jun. 20, 2001*

▶ Content
Feature Articles
Web Logs
Book Excerpts
Events

▶ Topics

J2EE
• Developing a Simple JMS Example
• J2EE Transaction Frameworks, Part 3
• Distributed Transaction Primer
Java OSS
OSS Trouble Ticket API
OSS Quality of Service API
More ▶

EJB
• EJB 2 and J2EE Packaging
• Using Ant and WebLogic EJBs
• EJB 2 Message-Driven Beans
EJB Transactions Tutorial
EJB Tutorial
Intro EJB Tutorial
More ▶

JSP and Servlets
• JSP Security for Limiting Access to Application-Internal URLs
• Jakarta Taglibs
• Writing Servlet 2.3 Filters

▶ Java Jobs
To learn about the hottest Java jobs, click here.

▶ Resources

Worst Java Fears Confirmed in Latest Court Ruling Against Microsoft?
"Microsoft intended to deceive Java developers." Within the following eWeek analysis of the US Court of Appeals decision on June 28 in the Microsoft antitrust case, "Java developers will find their darkest suspicions vindicated." [Steve Anglin]

More ▶

news

Womble[lambda News]

Rachel - JavaWebStart Resource Loading Made Easy[New Entries at Internet Product Watch]

Free Java Chat - Free Remote Hosted Java Chat Room[New Entries at Internet Product Watch]

BEA WebLogic 6.1 - Java Application Server - v6 Now Available[New Entries at Internet Product Watch]

Schema2Java Compiler Tool - Publicly Available S2J Compiler[New Entries at Internet Product Watch]

SigmaChat - Professional Java Chat Solution For Your Web Site[New Entries at Internet Product Watch]

Uppli.com Releases uMessenger 1.0[JabberCentral Recent Jabber News]

Processing XML with

JAIN API *Jul. 2, 2001*

Java 2ME Wireless Toolkit *Jul. 2, 2001*

Java OSS *Jul. 2, 2001*

OSS Trouble Ticket API *Jul. 2, 2001*

OSS Quality of Service API *Jul. 2, 2001*

More ▶

Java API Flow Chart

A quick reference to the Java family of technologies and ONJava.com content.

O'Reilly Network Technologies

Sites

ONJava.com
ONLamp.com
openp2p.com
Perl.com
XML.com

Subjects

Apache
BSD
Javascript and CSS
Linux
Mac
Mozilla
.NET
Perl
Policy
PHP
Python
Web Services
Wireless
XML

Jakarta Taglibs Project

XSP Tutorial

The Jakarta Taglibs Project, Part I

More ▶

JDBC and SQLJ

JDBC 2.0 Tutorial

Sybase White Papers/Tech Papers

SQLJ Resource

More ▶

Java and XML

• Using PASX

• Java and XML Week

• XML Processing with TRaX

Java and XML Library: Jato

Jena API for RDF

Java XML DOM API

More ▶

Open Source Java

• Using PASX

• Using Ant and WebLogic EJBs

• Jakarta Taglibs

Wire protocol with SOAP

Java and XML Library: Jato

JXTA

More ▶

P2P Java

• Hello JXTA!

• Crudlets: Making Peace Between Extreme Jini and XML Viewpoints

• First Contact: Is There Life in JavaSpace?

JXTA

Jini Tutorial

RMI Tutorial

More ▶

Wireless Java

Java[lambda News]

Web Database Objects - Extremely Rapid Oracle/Java Development Automation Software[New Entries at Internet Product Watch]

Constrained Genericity[lambda News]

More ▶

events

Wireless DevCon Summer
New York, NY *Jul. 22, 2001*

Intl Conf for Java Dev Fall
Santa Clara, CA *Sep. 23, 2001*

Wireless DevCon Europe Fall
London, England

Mobile Computing World
New York, NY *Oct. 1, 2001*

XML DevCon Fall
San Jose, CA *Oct. 29, 2001*

More ▶

▶ Books

The Java Enterprise CD Bookshelf

Java Message Service

JavaServer Pages

More ▶

▶ Affiliates

Servlets.com

jdom.org

• Data Persistence with Waba

• Wireless ONJava Week

• Are Device Independent Wireless Internet Applications Possible?

JAIN API

Java 2ME Wireless Toolkit

Wire protocol with SOAP

More ▶

Java Design

• The WeakHashMap Class

• The Performance of Java's Lists

• Java Internationalization and Localization

J2EE Component Patterns Tutorial

Java Reflection Tutorial

EJB Observer Pattern

More ▶

Java Security

• JSP Security for Limiting Access to Application-Internal URLs

• Java Security: Java Application Security

• Secure Your Sockets with JSSE

JAAS Tutorial

Java Security Evolution, Part 2

Java Security Evolution, Part 1

More ▶

Java Media

Java 3D API Tutorial

Java 2D API Tutorial

JAI Tutorial

More ▶

J2SE

• Learning Polymorphism and Object Serialization

• JVM to .NET: I'm Not Dead Yet!

• Macworld Java Update

RMI Tutorial

JFC Swing Tutorial, Part I

JFC Swing Tutorial, Part 2

P2P Contributors

- Rael Dornfest
- Lucas Gonze
- Richard Koman
- Andy Oram
- Tim O'Reilly
- Jon Orwant
- Clay Shirky
- Kelly Truelove

Topics

Features

Admin installs P2P client. Now he faces arrest and imprisonment

Maybe socialism wasn't such a bad idea after all...

Napster is dead. Is Morpheus the new king of the file-sharing underworld?

Microsoft Plans Shared Source .NET

Microsoft has announced its first foray into the waters of publicly shared source. Tim O'Reilly talks to Microsoft program manager (and FreeBSD sympathizer) Dave Stutz about Redmond's plans to release shared-source code of parts of the .NET framework. [.NET DevCenter]

JXTA Search: A look at the future of searching

Searching on the Web is ridiculous -- searching through an index of keywords often weeks old. JXTA Search, the marriage of P2P searching with Sun's Project JXTA, promises to turn searching into a real-time endeavor. We talk to JXTA Search developers Gene Kan and Steve Waterhouse. [openp2p.com]

JXTA Chat, Sans Server

OpenP2P Top 5 Articles

1. Peer-to-Peer Makes the Internet Interesting Again
2. Gnutella: Alive, Well, and Changing Fast
3. What Is P2P ... And What Isn't
4. What's on Freenet?
5. Gnutella and Freenet Represent True Technological Innovation

P2P Weblogs

Links and commentary

Collaboration
Distributed Computation
Enterprise P2P
File-Sharing
Freenet
General
Gnutella
Hailstorm
JXTA
Instant Messaging
Napster
P2P Law
Security
Standards

P2P Directory

The O'Reilly P2P directory lists companies, projects and initiatives in this emerging but as yet undefined space.

What's New

- Ogg Vorbis
- FirstPeer
- 1stWorks
- Napster

Clip2 finds JXTA a powerful tool for writing fully decentralized applications. [OpenP2P.com]

Hailstorm: Open Web Services Controlled by Microsoft

To an astonishing degree, Microsoft's Hailstorm relies on open standards like SOAP, Kerberos, and XML. But with typical audacity, MS also plans to centralize control of the system at critical junctures. [OpenP2P.com]

Li Gong: JXTA All About Community

Leading a BOF at JavaOne, JXTA engineering director Li Gong explained which P2P problems JXTA attacks, and emphasized that the project is a community effort. [OpenP2P.com]

JXTA BOFs: Community and Implementation

Raffi Krikorian reports on two birds of a feather sessions at JavaOne -- one on community and one on JXTA implemetations. [OpenP2P.com]

The Buzz on Swarmcast

Swarmcast -- OpenCola's technology for speeding up downloads of large files -- was released for public beta this week. We talk to Swarmcast inventor Justin Chapweske and OpenCola founder Cory Doctorow about the technology. [OpenP2P.com]

Are We Promoting Piracy?

After the publication of alt.napster, radio host David Lawrence interviewed Steve and Tim O'Reilly about whether our story is advocating online piracy. Here's the transcript of a wide-ranging interview. [OpenP2P.com]

OpenNap Use Crashes

A Clip2 study shows that OpenNap usage has plummetted since February, probably due to RIAA's notice to ISPs [OpenP2P.com]

P2P Working Group Gets to Work

The P2P Working Group will meet May 30-31. Unlike previous meetings, this one will be about hard work, not politics. [OpenP2P.com]

What's Up at Uprizer?

Ian Clarke provides a preview of Uprizer's Freenet-inspired content distribution system. [OpenP2P.com]

The Trouble with JXTA

Steve McCannell's Weblog

P2P Networks Becoming Accepted by the Mainstream

With three music subscription services due to be launched this summer, P2P networks such as Gnutella, AudioGalaxy, and Morpheus look to already have the attention of the mainstream audience, making a subscription service even more of a tough sell. [O'Reilly Network Weblogs]

More Weblogs

If .NET is the OS, will the Internet bluescreen? [Lucas Gonze]

Turn About Is Fair Player [Lisa Rein]

P2P Networks Becoming Accepted by the Mainstream [Steve McCannell]

Morpheus Vs. Gnutella [Lucas Gonze]

Hailstorm Article "Claptrap" [Richard

Freenet developer Adam Langley says JXTA, Sun Microsystems' peer-to-peer initiative, is neither useful nor interesting. [OpenP2P.com]

Inside JXTA
Dig into JXTA with OpenP2P.com's in-depth coverage including analysis from Rael Dornfest and Kelly Truelove, and a tutorial on the JXTA Shell. [OpenP2P.com]

Latest Content by Topic

Collaboration — More ▶
- The Human Side of Peer to Peer: Where Technology and Conversation Come Together
- Software's Humble Wizard Does It Again
- Collaborative Computing with openCOLA

Distributed Computation — More ▶
- Uncheatable Distributed Computations
- Porivo: Load Testing with P2P
- Popular Power Turns Off the Lights

Enterprise P2P — More ▶
- Ford to use peer-to-peer technology to boost fuel-efficient car designs
- Inside JXTA
- Hello JXTA!

File-Sharing — More ▶
- Is All Music File-Sharing Piracy?
- Morpheus Out of the Underworld
- If You Can't Track 'em, Join 'em

Koman]

GAIM and .NET [Lucas Gonze]

The General Bills #2 [Lucas Gonze]

Peer-to-peer power generation? [Andy Oram]

Meerkat Daily News
July 11, 2001

Morpheus a possible successor to Napster
ComputerNewsDaily: "A new online file-sharing program created by a Dutch company is poised to become the next Napster and could be much harder for the music industry to stop, a Web research firm said Tuesday. [NewsForge]

Use P2P. Go to Jail. Any Questions?
David McOwen faces arrest, trial, and imprisonment because he installed the Distributed.net client on computers at the college where he worked. [O'Reilly Network: What's New]

More on Mac OS X and music
[Mac Central latest headlines]

Freenet More

- Peer-To-Peer: Harnessing the Power of Disruptive Technologies: Chapter 12: Free Haven
- What's Up at Uprizer?
- Free speech, liberty, pornography

General More

- Thinking Beyond Scaling
- Editor Andy Oram on O'Reilly's First Peer-to-Peer Book
- Backlash!

Gnutella More

- The JuxtaNet
- alt.napster
- Gnutella Targeted for Piracy Control

Hailstorm More

- Opening Up .Net to Everyone
- Introduction to .NET
- Peer Review: How We Got From P2P to Hailstorm

Instant Messaging More

- Learning the JXTA Shell
- A More Sensitive Mail Notifier
- Chapter 6: Jabber Conversational Technologies

JXTA More

- Li Gong: JXTA All About Community
- JXTA BOFs: Community and Implementation
- JXTA Search: A look at the future of searching

Napster More

Music industry heeds lessons of Napster
ZDNet reports on the fallout from the war between Napster and the recording industry, quoting an Intellectual Property lawyer who believes that the Napster case has shown that the industry has gotten to the point where peer-to-peer sharing can be filtered. [NewsForge]

Beta Test Philips' New MP3 Boombox
[BetaNews.Com]

The New Napster / The Record Industry Has Met its Match
"San Francisco Gate" - The New Napster / The Record Industry Has Met its Match. [Privacy Digest]

Use P2P, Go to Jail. Any Questions?
David McOwen installed the Distributed.Net client at the college where he worked. [openp2p.com]

Morpheus Vs. Gnutella
Were recent articles on Morpheus an overreaction or a wake up call for Gnutella developers? [O'Reilly Network Weblogs: P2P]

- Is All Music File-Sharing Piracy?
- Italian Firm's Dolce Music Deal
- Napster Service: 2Legit 2Quit

P2P Law More ▶

- The Digital Millennium Copyright Act
- Code-Breakers Go to Court
- Italian Firm's Dolce Music Deal

Security More ▶

- Code-Breakers Go to Court
- Peer-To-Peer: Harnessing the Power of Disruptive Technologies: Chapter 12: Free Haven
- Real Progress in Secure Music

Standards More ▶

- P2P Working Group Gets to Work
- P2P Working Group Meeting in May
- Interoperability, Not Standards

Use P2P, Go to Jail
David McOwen installed the Distributed.Net client at the college where he worked. [openp2p.com]

P2P Networks Becoming Accepted by the Mainstream
With three music subscription services due to be launched this summer, P2P networks such as Gnutella, AudioGalaxy, and Morpheus look to already have the attention of the mainstream audience, making a subscription service even more of a tough sell. [O'Reilly Network Weblogs]

Is Napster dead or just napping?
The Australian Industry Standard Jul 10 2001 4:33PM ET [Moreover internet international stories]

MP3 TAG Remover 1.10 Beta Released
[BetaNews.Com]

Gnucleus: The new Napster
SF Gate: "If the recording industry thought Napster was a headache, it's going to get a genuine migraine from the latest version of Gnucleus, a free Windows-based open source software program released last month for the

Gnutella file-sharing network. [NewsForge]

Alt MP3 Screensaver Player 1.3 Beta Released
[BetaNews.Com]

P2P Networks Accepted by the Mainstream
With three music subscription services due to be launched this summer, P2P networks such as Gnutella, AudioGalaxy, and Morpheus look to already have the attention of the mainstream audience, making a subscription service even more of a tough sell. [O'Reilly Network Weblogs: P2P]

U.S. court reacts to Net misuse warning by putting a muzzle on watchdog system
HoustonChronicle.com - U.S. [Privacy Digest]

NEWS: P2P off to a slow start
[DominoPower Magazine]

PeerIntelligence: Don't Re-invent Business Models for P2P
[Peertal the P2P News Portal]

Click N' Burn: new Mac CD recording package

Click 'N Burn Macintosh is a new $50 CD recording software package that "allows computer users to create music CDs, data CD-ROMS,... [MacNN]

MicronPC back in new computer groove
ZDNet Jul 9 2001 7:40PM ET [Moreover enterprise computing stories]

Already a member?
Login Here

Privacy Policy
Terms of Use

Welcome to Perl.com!

Subscribe to our email newsletter using the form below. In addition, your registration will allow you to use our article talk back feature.

*An * indicates a required field.*

Please choose a **login name** and **password** for your account on Perl.com:

Login Name *

Password *

Retype Password *

You can elect to receive our e-mail newsletter announcing new articles and features:	**Newsletter** ☐ Yes, please subscribe me to the Perl.com weekly newsletter

Please let us know how we can contact you:

First Name *

Last Name *

E-Mail *

Title

Organization

Mailing Address

City

State

Zip or Postal Code

Country

Phone Number

Fax Number

By submitting this registration form, you agree to the Terms of Use.

Submit Reset

Terms of Use

1. Do not post any content or link to any content which (a) is libelous, defamatory, obscene, pornographic, abusive, harassing or threatening, (b) contains viruses or other contaminating or destructive features, (c) violates the rights of others, such as content which infringes any copyright, trademark, patent, trade secret or violates any right of privacy or publicity, or (d) otherwise violates any applicable law.
2. Do not use this discussion area for any commercial purposes, to distribute any advertising or solicitation of funds or goods and services.
3. Perl.com does not and cannot review the content posted by users on this service and is not responsible for such content. However, we reserve the right to delete, move or edit any content that we may determine is unacceptable. You shall remain solely responsible for all content posted by you.
4. By posting a comment here you are giving Perl.com the right to publish your comments.

- Home
- Registration
- Hotel/Travel
- See & Do
- Tutorials
- Sessions
- Evening Events
- BOFs
- Speakers
- Press
- Mail List
- Exhibitors
- Sponsors

Innovate--Collaborate--Discover

O'Reilly Open Source Convention

Sheraton San Diego Hotel, San Diego, CA
July 23-27, 2001

Don't miss the Great Open Source Debate

Fueling the Open Source Alternative

We are pleased to announce the 3rd annual O'Reilly Open Source Convention, July 23-27, 2001, at the waterfront Sheraton Hotel and Marina in San Diego, California. This year's expanded convention includes over 250 sessions in 14 tracks on key open source technologies such as Perl, Apache, XML, Python, PHP, MySQL, Linux, and many more. Browse **Sessions** and **Tutorials** for full details.

Read Tim O'Reilly's reasons why **Open Source is Here to Stay**.

Enjoy San Diego

See & Do
Attractions
Kid's World

News Updates

Tim O'Reilly on Microsoft's Participation at the Open Source Convention--Microsoft senior vice president Craig Mundie will speak at O'Reilly's upcoming Open Source Convention, and he'll debate Red Hat's Michael Tiemann and other open source leaders. Tim O'Reilly talks about how this came about and what he hopes to accomplish by bringing together Microsoft representatives and open source advocates at the **O'Reilly Open Source Convention**.

Preview: O'Reilly XTech 2001 Conference on XML-- O'Reilly's XTech 2001 will be held from July 23-27 in San Diego, California. The conference chair, Edd Dumbill, previews this essential meeting for XML developers.

O'REILLY XTECH2001 CONFERENCE ON XML THE O'REILLY OPEN SOURCE CONVENTION

Open Source Conference News Coverage--Get the latest OSCON news from the O'Reilly Network's conference news site. You'll find articles, interviews, as well as audio from select keynotes and talks, including Craig Mundie's keynote about Microsoft's shared source strategy.

Open Source Convention media sponsor *Dr. Dobb's Journal* has graciously offered a year's complimentary subscription to DDJ when you **register**. Already have a subscription? DDJ will add a year onto your current subscription.

What Would You Ask Craig Mundie? Microsoft senior vice president Craig Mundie, who recently raised the ire of the open source community, has agreed to give a keynote and participate in a panel discussion at July's O'Reilly Open Source Convention. Here's your chance to **put your questions to him**.

O'REILLY OPEN SOURCE CONVENTION

Microsoft's Shared Source Philosophy--Microsoft senior vice president Craig Mundie has accepted Tim O'Reilly's invitation to present his controversial Shared Source philosophy in a keynote address at the O'Reilly Open Source Convention. A panel discussion with Red Hat CTO Michael Tiemann, Tim O'Reilly, and others will follow Mundie's talk.

Convention updates--This page lists changes and updates to the convention schedule.

O'Reilly Perl Conference 5--Where else can you get together with Perl hackers, learn something useful, have fun, and change the world?

O'Reilly XTech2001 Conference on XML (New this year, in association with Graphic Communications Association)--The O'Reilly XTech2001 Conference on XML is the essential developer-to-developer forum for XML technologies.

O'Reilly Summit on Open Source Strategies--Collocated at this year's Open Source Convention this strategic summit explores the best practices in collaborative software development.

PHP Conference--The first ever PHP Conference brings the community together for **sessions** and **tutorials** designed to explore and strengthen PHP in the open source space. Take a look under the hood at everything PHP--from understanding PHP on wireless devices to participating in a discussion on the future of PHP featuring some of the core developers and luminaries from the PHP family.

The 8th Tcl/Tk Conference (New this year)--The Tcl/Tk conference is a continuation of the USENIX conference, with many of the same speakers and organizers. The Tcl/Tk **tutorials** and **sessions** are still the only place where you can hear about the state of the art in Tcl/Tk from those who know.

---

webmaster@oreilly.com

Downloads
Documentation
CPAN
FAQs
Training
Resources
Article Archive
Books
Search
Register/Log in

Columns
P5P Digest
P6P Digest
Off the Wall

Downloading the Latest Version of Perl

by Tom Christiansen
March 24, 1999

Contents of this document

Overview

Perl is Open Source software. You can download it for free as a source code or as a pre-compiled binary distribution.

Perl Source Code Distribution

Use the instructions below to download the complete Perl source code distribution via your Web browser. On CPAN, you will find Perl source in the /src directory. The source code is distributed as either a Unix-style tar archive compressed by GNU zip (gz) or as a PC-style ZIP archive. (The only difference is the type of archive; the same source code files are inside the archive.

Once you download the gzip or zip archive, you will have to extract the source code files from the distribution and then follow the instructions on how to compile the source code for your system. If you don't have a compiler, or are unclear about compiling source code, then look first for a binary distribution of Perl.

Stable Production Release

The current version of Perl is **5.6.1**. This is a stable, tested release that you should use in production environments.

DOWNLOAD Stable Release from CPAN/src	
Get source for Unix systems	stable.tar.gz archive
Get source for Microsoft systems	stable.zip archive

Features new to this release include generation of C code and support for lightweight processes. This version is a stable, production release that compiles out of the box for virtually all flavors of Unix (its native environment), plus VMS, OS/2, and 32-bit Microsoft platforms as well.

Experimental Developer's Release

The developer's release of Perl is currently at version **5.7.1**. The developer's release is considered purely experimental. It is intended for particularly brave developers who want to get close to the front lines of Perl development.

DOWNLOAD Developer's Release from CPAN/src	
Get source for Unix systems	devel.tar.gz archive
Get source for Microsoft systems	devel.zip archive

Previous Versions of Perl

For the faint of heart, the previous version of Perl is the **5.005_03** release. Timid souls who are afraid to upgrade to the current release might want this.

DOWNLOAD Last Release from CPAN/src	
Get source for Unix systems	perl5.005_03.tar.gz archive

Source Licence

Perl is Open Source software. It's free for you to download and use as you wish. Perl's license is called the Artistic license. Read it if you aren't sure what you can or can't do. The bottom line is that this is a kinder and gentler version of the GNU license -- one that doesn't infect your work if you care to borrow from Perl or package up pieces of it as part of a commercial product!

Binary Distributions

Binary distributions of Perl are available for various platforms, including Win32 (Windows NT/95), Mac and Unix. Please read all documentation that comes with each package, as one distribution for a given platform may be very different than another for the same platform (eg. different add-ons, compiled from a different version of Perl etc.)

Generally speaking, CPAN doesn't distribute Perl binaries. It does provide information in the /ports directory that will point you to sites that maintain binary distributions for specific platforms.

Perl for Win32

ActivePerl is the long-awaited "merge" of the two popular Perl Win32 ports. These binaries are intended for anyone using Windows 95/98 or Windows NT. While you can download older Win32 binaries from CPAN, we recommend that you download ActivePerl from ActiveState.

ActivePerl includes:

Perl for Win32: A binary of the core Perl distribution.

PerlScript: ActiveX scripting engine, like JavaScript or VBScript with a Perl brain.

Perl Package Manager: Use the Perl Package Manager (PPM) to easily view and install the large collection of modules and extensions that are available in binary

packages at the ActiveState Package Repository.

DOWNLOAD Win32 Binaries from ActiveState	
Go to the ActivePerl page at Activestate.com:	ActivePerl

If you have problems installing ActivePerl, please see the ActiveState site for support information.

Perl for the Macintosh

For Macintosh users, there is really only one way to get Perl. Check out the MacPerl Homepage Home Page.

DOWNLOAD Macintosh Binaries from MacPerl	
Go to the MacPerl page:	MacPerl

Perl for Unix

Perl was originally envisioned and written for Unix. Perl will build on almost all Unix platforms and its variants, such as Linux. As far as which Unix variants Perl will compile on, Larry says "I'm not sure there are any unsupported versions of Unix, except on machines without adequate address space such as PDP-11 or i286, and perhaps some old versions of Unix that are no longer supported themselves."

DOWNLOAD Unix Binaries from CPAN	
Go to the CPAN /ports list to locate a binary:	List of Ports

Alien Ports

If you are looking to run Perl on a non-native system, then have a look at the /CPAN/ports/ directory. Note that the standard Perl distribution compiles even on most closed-source systems now, so the ports directory is of somewhat limited utility for them.

Getting Help

If you need help, here are some suggestions.

- If you encounter problems configuring, compiling, or installing Perl from the source kits, please read the README in the source directory. Besides including important tips for various platforms, they also document where to send mail if you still have build difficulties.
- If you have problems downloading via ftp, try this web mirror of CPAN instead.
- If you still can't get the download or the unpacking to work, you'll need to contact your local systems administrator (not perl.com staff) for help with your local system. Also, try to contact other Perl users in your organization or your community for help.
- If you have issues with the ActiveState port, please contact them directly.

Downloads

Documentation

CPAN

FAQs

Training

Resources

Article Archive

Books

Search

Register/Log in

Columns

P5P Digest
P6P Digest
Off the Wall

Documentation

This section contains the core documentation for the Perl language. Please note that many applications use Perl modules and the documentation for those modules can be found on CPAN. Also, the FAQs are an excellent source of information about many different applications of Perl.

Core Documentation

Practical Extraction and Report Language
The main perl manpage that explains what Perl is and how to get started programming in Perl. [Source: Perldoc.com]

How To Execute the Perl Interpreter
A description of how to invoke Perl programs and identify the location of the Perl interpreter on different platforms. [Source: Perldoc.com]

Internationalization and Localization: Perl's Locale System
A description of how to set and use the locale system to support international character sets. Introduced in 5.004. [Source: Perldoc.com]

Perl Data Types
An introduction to Perl's three basic data structures: scalars, arrays of scalars, and associative arrays of scalars, also known as hashes. [Source: Perldoc.com]

Perl Interprocess Communication
The basic IPC facilities of Perl are built out of the good old Unix signals, named pipes, pipe opens, the Berkeley socket routines, and SysV IPC calls. [Source: Perldoc.com]

Perl Operators and Precedence
A listing of all of Perl's operators from lowest to highest in precedence. [Source: Perldoc.com]

Perl Regular Expressions
An introduction to the syntax of regular expressions in Perl. [Source: Perldoc.com]

Perl Report Formatting
An introduction to generating simple reports and charts in Perl using the format statement. [Source: Perldoc.com]

Perl security

An introduction to security issues for writing and deploying Perl programs. [Source: Perldoc.com]

Perl Style Guide

A set of general guidelines for writing Perl programs using a style that makes the code easier to read. [Source: Perldoc.com]

Perl Subroutines

An introduction to user-defined subroutines. [Source: Perldoc.com]

Perl Syntax

An overview of the syntactic elements of the Perl language such as declarations, statements and loops. Also includes an explanation of embedding POD documentation. [Source: Perldoc.com]

Writing Portable Perl Programs

A introduction to writing Perl programs that are portable across different systems. Covers newlines, file systems, system interaction and IPC differences. [Source: Perldoc.com]

Perl Internals

Perl's Internal Functions

A description of some of the internal functions of the Perl executable. Covers the magic structure. [Source: Perldoc.com]

Perl's IO Abstraction Interface

A listing of the I/O functions that are used in Perl's source code, based on ANSI C's stdio.h. [Source: Perldoc.com]

Troubleshooting

Perl Debugging

An introduction to Perl's debugger, covering how to invoke it and use its command set. [Source: Perldoc.com]

Perl Diagnostics and Error Messages

This document describes Perl's error messages, including warnings and fatal errors. It provides an alphabetical list of errors. [Source: Perldoc.com]

Perl Traps for the Unwary

A useful listing for troubleshooting common problems with Perl programs. [Source: Perldoc.com]

Data Structures

Complex Data Structures Cookbook

This document describes Perl's complex data structures, which were introduced in 5.0. It covers arrays of arrays, hashes of arrays, hashes of hashes, and other more elaborate constructs. [Source: Perldoc.com]

Manipulating Lists of Lists in Perl

A tutorial describing how the build and use lists of lists (arrays of arrays). [Source: Perldoc.com]

Perl Predefined Variables

A listing of reserved names that have special meaning to Perl. Also includes a discussion of variables that contain information about various error conditions. [Source: Perldoc.com]

Perl References and Nested Data Structures

An introduction to using references in Perl, distinguishing between symbolic references and "hard" references introduced in 5.005. [Source: Perldoc.com]

Perl's Builtin Functions

A listing of the built-in functions in Perl, grouped by category and alphabetically. [Source: Perldoc.com]

C and Perl

Calling Perl from C Programs

An introduction to calling Perl subroutines directly from C, i.e., how to write callbacks. Examples of error handlers and event-driven programming are shown. [Source: Perldoc.com]

How To Embed Perl In Your C program

This document describes how to embed a perl interpreter and perl programs in a C program. [Source: Perldoc.com]

Tutorial for XSUBs

A tutorial on using XSUB to create extensions.

XS Language Reference Manual

A description of XS and XSUB, an extension interface for defining external subroutines. [Source: Perldoc.com]

Objects

Bag'o Object Tricks

A collection of tricks and hints related to object-oriented programming in Perl covering, for example, the use of instance variables and the mechanics of object and class relationships. [Source: Perldoc.com]

How to Hide an Object Class in a Simple Variable

An explanation of the tie() function introduced in Perl 5.0 that binds a variable to a class. [Source: Perldoc.com]

Perl Objects

An explanation of how to think of objects in Perl as references and of classes as packages. This document is meant to demystify object-oriented programming terminology. [Source: Perldoc.com]

Tom's Object-oriented Tutorial for Perl

An introduction to object-oriented programming in Perl. [Source: Perldoc.com]

Modules

Constructing New Perl modules and Finding Existing Ones

A description of the Perl Module Library including a description of pragma, standard and extension modules. [Source: Perldoc.com]

Installing CPAN Modules

An explanation of how to download modules from CPAN and installing modules on on most platforms. [Source: Perldoc.com]

Perl Modules

A description of how to work with modules in your programs. [Source: Perldoc.com]

Releases

Perl Release History

A record of Perl source code releases. [Source: Perldoc.com]

What's New for perl5.004

A description of differences between the 5.003 release (as documented in Programming Perl, second edition--the Camel Book) and 5.004. [Source: Perldoc.com]

What's New for perl5.005

A summary of the differences between the 5.005 release and this one. [Source: Perldoc.com]

Comprehensive Perl Archive Network

Welcome to CPAN! Here you will find All Things Perl.

Searching

- Perl core and CPAN modules documentation (Randy Kobes)
- Perl core documentation (Carlos Ramirez)
- CPAN modules, distributions, and authors (search.cpan.org)
- CPAN modules documentation (Ulrich Pfeifer)

Browsing

- modules
- scripts
- binary distributions ("ports")
- source code
- comp.lang.perl.announce archives
- recent arrivals
- recent modules
- CPAN sites list
- CPAN sites map

FAQ etc

- CPAN Frequently Asked Questions
- Perl Mailing Lists
- Perl Bookmarks

Yours Eclectically, The Self-Appointed Master Librarian (OOK!) of the CPAN *Jarkko Hietaniemi* cpan@perl.org [Disclaimer] 2001-04-01

CPAN master site hosted by

Downloads

Documentation

CPAN

FAQs

Training

Resources

Article Archive

Books

Search

Register/Log in

Columns

P5P Digest
P6P Digest
Off the Wall

FAQs: Perl's Frequently Asked Questions

For problems or assistance with this site, email help@www.perl.com

TCPC Perl Training Courses

Looking for professional training in Perl? *Tom Christiansen Perl Consultancy* provides private, corporate training to meet your company's unique Perl training needs.

Courses are scheduled under one of the following arrangements:

- We travel to your private site, using your facilities for classroom and lab activities.
- You travel to our site, using our open courses in Boulder, Colorado.
- We both travel to a public conference or user-group event. We will be happy to work with your conference coordinator to arrange a public course at your favorite event.

Further information is available about:

- What is Perl?
- Available Topics for On-site Courses
- Public Courses at Large Events
- Public Courses in Boulder, Colorado
- Instructor Information
- Pricing Information
- How to Contact Us

Downloads

Documentation

CPAN

FAQs

Training

Resources

Article Archive

Books

Search

Register/Log in

Columns

P5P Digest
P6P Digest
Off the Wall

Resource Topics

Formerly the "Perl Reference Pages"

- Binaries
  Compiled distributions of Perl for various operating systems
- Biology
  Perl modules and programs for bioinformatics
- Books and Magazines
  Books and magazines about Perl
- Business
- C and Perl
- CGI
  CGI programming with Perl
- Communications
  Sockets and ports and protocols -- Oh my!
- Conversion
  Conversion utilities -- Zip, BinHex, etc..
- CORBA
  Information on CORBA Perl
- Core Documentation
- Courses and Training
  Non-Web-based courses to learn Perl
- Data Structures

- Databases

- Debugging
  Debugging programs

- Editors
  Editors for Perl or editors using Perl

- Files
  Working with files and filesystems

- Finance
  Financial applications and modules

- Games
  Games written in Perl

- Gear
  Shirts, hats, toys -- Essentials for Perl programming

- Geographical
  Perl modules for geographical applications

- Graphics

- HTTP
  Web servers and Perl modules for the server

- Java

- Lingua
  Perl scripts and modules for linguistics

- Linux
  Linux-specific Perl programs and modules

- Lists
  Mailing lists about Perl

- Macintosh
  MacPerl info

- Mail and USENET News
  Modules for mail and news

- Materials Science
  Crystallography data classes anyone?

- Mathematics
  Perl scripts and modules for mathematics

- Modules

- Music
  Perl modules and programs for music

- Net
  Network programming

- Newsgroups
  USENET newsgroups pertaining to Perl

- NeXT
  Ports and modules for the NeXT environment

- Objects

- Oddities
  Strange and unusual things people do with Perl

- Palm Pilot
  Perl stuff for the Palm Pilot

- PCL
  Perl programs and interfaces for PCL

- Perl Internals

- Porting
  Porting information

- Programming

- Regular Expressions
  Information about regular expressions

- Releases
  Information about current and past releases.

- School
  Perl programs for schools
- Screen I/O
  Screen handling, menu systems
- Security
- Sets
  Working with sets
- Solaris
  Ports and modules for the Solaris environment
- Sorting
  Information on various sorting techniques
- Sound and Audio
  Working with sound and audio
- Statistics
  Statistical programs
- Style Guides
  Programming standards and style guides
- Sysadmin
  System administration modules and programs
- Text Tools
  Text tools in Perl
- Time
  Perl resources about date and time
- Troubleshooting
- Tutorials
  Tutorials, guides, manuals and FAQs
- User Groups
  Perl User Groups (pugs?)
- User Interfaces
  Character and graphic user interfaces

- Version Control Systems
  Access Version Control System data with Perl

- VMS
  Perl on VMS platforms

- Web Admin
  Perl programs and tips for Web administration -- Note: CGI Perl programs typically interact with the Web. Web admin tools don't.

- Win32

- XML

Downloads
Documentation
CPAN
FAQs
Training
Resources
Article Archive
Books
Search
Register/Log in

Columns
P5P Digest
P6P Digest
Off the Wall

Article Archive

A listing of all Perl.com articles grouped by topic.

Advocacy

Why I Hate Advocacy
Are you an effective Perl advocate? Mark Dominus explains why you might be advocating Perl the wrong way. [Dec 12, 2000]

Ten Perl Myths
Ten things that people like to say about Perl that aren't true. [Feb 23, 2000]

Sins of Perl Revisited
Tom Christiansen published the original seven sins in 1996. Where are we now? [Nov 30, 1999]

White Camel Awards to be Presented at O'Reilly's Perl Conference 3.0
The White Camel awards will be presented to individuals who have made outstanding contributions to Perl Advocacy, Perl User Groups, and the Perl Community at O'Reilly's Perl onference 3.0 on August 24, 1999. [Jun 28, 1999]

Perl vs. Python
A comparison of Perl and Python by Tom. [Dec 1, 1996]

Books and Magazines

Choosing a Perl Book
What to look for when choosing from the many Perl books on the market. [Jul 11, 2000]

Camel Critiques

Business

The e-smith Server and Gateway: a Perl Case Study
Kirrily "Skud" Robert explains the Perl behind the web-based administrator for the e-smith server. [Feb 20, 2001]

Perl Support
Where can I buy a commercial version of Perl and is a support contract available? [Mar 25, 1999]

Tom Christiansen Perl Consultancy
The complete details on professional training from recognized Perl experts. [Mar 25, 1999]

Perl Rescues a Major Corporation
How the author used Perl to create a document management system for a major aerospace corporation and saved the day. [Oct 21, 1998]

How Perl Creates Orders For The Air Force
Brent Michalski, while in the Air Force, created a Web-based system written in Perl to simplify the process of ordering new hardware and software. [Jul 22, 1998]

White Hats and Black
Tom Christiansen's take on the freeing of Netscape. [Jan 22, 1998]

C and Perl

Why Not Translate Perl to C?
Mark-Jason Dominus explains why it might not be any faster to convert your code to a C program rather than let the Perl interpreter execute it. [Jun 27, 2001]

Pathologically Polluting Perl
Brian Ingerson introduces Inline.pm and CPR; with them you can embed C inside Perl and turn C into a scripting language. [Feb 6, 2001]

Success in Migrating from C to Perl
How one company migrated from using C to Perl -- and in doing so was able to improve their approach to code design, maintenance and documentation. [Jan 19, 1999]

CGI

Using CGI::Application
The Common Gateway Interface may well be the backbone of many web applications, but sometimes it can feel dry and monotonous to work with. If you're fed up with `"my $query = CGI->new()"`, Jesse Erlbaum presents a kinder, gentler

alternative. [Jun 5, 2001]

Perl and CGI FAQ
This FAQ answers questions for Web developers. [Apr 14, 1999]

Community

This Week on p5p 2001/07/09
No 5.8.0 yet, numeric hackery, worries about PerlIO and much more. [Jul 9, 2001]

People Behind Perl: Nathan Torkington
So you use Perl, and you probably know that it was brought to you by "Larry Wall and a cast of thousands". But do you know these people that make up the Perl development team? Simon Cozens talks to Nathan Torkington, a long-time Perl developer and a mainstay of the Perl community. [Jul 3, 2001]

This Week on p5p 2001/07/02
Module versioning and testing, regex capture-to-variable, and much more. [Jul 2, 2001]

This Week on p5p 2001/06/25
5.7.2 in sight, some threads on regular expression, and much more. [Jun 25, 2001]

Yet Another YAPC Report: Montreal
Schuyler Erle gives a detailed report of all the exciting events at this year's Yet Another Perl Conference in Montreal. By his account, it appears to be an exciting time to be involved with the development of Perl. [Jun 21, 2001]

This Week on p5p 2001/06/17
Miscellaneous Darwin Updates, hash accessor macros, and much more. [Jun 19, 2001]

Privacy Policy
The O'Reilly and Perl.com privacy policy [Jun 15, 2001]

The Beginner's Attitude of Perl: What Attitude?
Robert Kiesling says that the Perl Community's attitude towards new users is common fare for Internet development and compared to other lists Perl is downright civil. [Jun 12, 2001]

This Week on p5p 2001/06/09
Removing dependence on strtol, regex negation, and much more. [Jun 12, 2001]

About perl.com
[Jun 7, 2001]

Turning the Tides on Perl's Attitude Toward Beginners
Casey West is taking a stand against elitism in the Perl community and seems to be making progress. He has launched several new services for the Perl beginner that are being enthusiastically received. [May 28, 2001]

[Hold the Sackcloth and Ashes]
Jarkko Hietaniemi, the Perl release manager, responds to the critique of the Perl 6 RFC process. [Nov 3, 2000]

[Critique of the Perl 6 RFC Process]
Many of the suggestions put forward during the Perl 6 request-for-comment period revealed a lack of understanding of the internals and limitations of the language. Mark-Jason Dominus offers these criticisms in hopes that future RFCs may avoid the same mistakes -- and the wasted effort. [Oct 31, 2000]

[Last Chance to Support Damian Conway]
As reported earlier, the Yet Another Society (YAS) is putting together a grant to Monash University, Australia. The grant will fund Damian Conway's full-time work on Perl for a year. But the deadline for pledges is the end of the week, and the fund is still short. [Oct 26, 2000]

[Report from YAPC::Europe]
Mark Summerfield tells us what he saw at YAPC::Europe in London last weekend. [Oct 2, 2000]

[Ilya Regularly Expresses]
Ilya Zakharevich, a major contributor to Perl 5, talks about Perl 6 effort, why he thinks that Perl is not well-suited for text manipulation, and what changes would make it better; whether the Russian education system is effective; and whether Perl 6 is a good idea. [Sep 20, 2000]

[Damian Conway Talks Shop]
The author of *Object-Oriented Perl* talks about the Dark Art of programming, motivations for taking on projects, and the "deification" of technology. [Aug 21, 2000]

[Report from the Perl Conference]
One conference-goer shares with us his thoughts, experiences and impressions of TPC4. [Aug 21, 2000]

[Reports from YAPC 19100]
Eleven attendees of Yet Another Perl Conference write in about their experiences in Pittsburgh last month. [Jul 11, 2000]

[Adventures on Perl Whirl 2000]
Adam Turoff's report on last week's Perl Whirl cruise to Alaska [Jun 13, 2000]

[ANSI Standard Perl?]
Standardized Perl? Larry Rosler, who put the ANSI in ANSI C, shares his thoughts on how Perl could benefit from standards in this interview with Joe Johnston. [Jun 6, 2000]

[Virtual Presentations with Perl]
This year, the Philadelphia Perl Mongers had joint remote meetings with Boston.pm and St. Louis.pm using teleconferencing equipment to bring a guest speaker to many places at once. Adam Turoff describes what worked and what didn't, and how you can use this in your own PM groups. [Dec 20, 1999]

[Happy Birthday Perl!]
According to the `perlhist` man page, Perl was first released twelve years ago, on December 18, 1987. Congratulations to Larry Wall on the occasion of Perl's twelfth birthday! [Dec 18, 1999]

[Perl as a First Language]
Simon Cozens, author of the upcoming Beginning Perl talks about Perl as a language for beginning programmers. [Nov 16, 1999]

[White Camel Awards]
An interview with White Camel Award winners Kevin Lenzo and Adam Turoff. [Sep 16, 1999]

[What's New in Perlland?]
Tom Christiansen's list of news and updates of interest to the Perl community. [Jul 25, 1999]

[A New Edition of www.perl.com]
Welcome to the new edition of www.perl.com! We've redesigned the site to make it easier for you to find the information you're looking for. [Jul 15, 1999]

[Dispatch from YAPC]
Brent was at YAPC -- were you? He reports from this "alternative" Perl conference. [Jun 30, 1999]

[What the Heck is a Perl Monger?!]
Want to start or find a Perl user group in your area? Brent interviews brian d foy, the creator of Perl Mongers to find out just what the Mongers are all about. [Jan 13, 1999]

[A Photographic Journal]
Photographs taken at the Perl Conference by Joseph F. Ryan: Perl Programmer at the National Human Genome Research Institute. [Aug 27, 1998]

[The Final Day at The Perl Conference]
Brent Michalski winds down his coverage of the Perl Conference. Highlights include: Tom Paquin's Keynote: "Free Software Goes Mainstream" and Tom Christiansen's "Perl Style". [Aug 21, 1998]

[Day 3: Core Perl Developers at Work and Play]
Recapping Larry Wall's Keynote, The OO Debate, The Internet Quiz Show, The Perl Institue and The Perl Night Life. [Aug 20, 1998]

[Day 2: Perl Mongers at The Conference]
Another exciting day! Brent talks about the *Advanced Perl Fundamentals* tutorial, the concept behind the Perl Mongers and the *Fireside Chat with Jon Orwant.* [Aug 19, 1998]

[Day 1 Highlights: Lincoln's Cool Tricks and Regexp]
Brent Michalski reports on the highlights of the first day of The Perl Conference. [Aug 18, 1998]

[Perl Conference 3.0 -- The Call for Participation]
This is a call for papers that demonstrate the incredible diversity of Perl. Selected papers will be presented at the third annual

O'Reilly Perl Conference on August 21-24, 1999 at the Doubletree Hotel and Monterey Conference Center in Monterey California. [Aug 17, 1998]

Core Documentation

The Artistic License
This document states the conditions under which a Package may be copied, such that the Copyright Holder maintains some semblance of artistic control over the development of the package [Aug 15, 1997]

CPAN

Common Questions About CPAN
Answers to the most common questions asked from cpan@perl.org [Jul 29, 1999]

Databases

DBI is OK
Chromatic makes a case for using DBI and shows how it works well in the same situations as DBIx::Recordset. [Mar 20, 2001]

DBIx::Recordset VS DBI
Terrance Brannon explains why DBI is the standard database interface for Perl but should not be the interface for most Perl applications requiring database functionality. [Feb 27, 2001]

A Short Guide to DBI
Here's how to get started using SQL and SQL-driven databases from Perl. [Oct 22, 1999]

Documentation

POD is not Literate Programming
[Mar 20, 2000]

Editors

Perl Builder IDE Debuts
A review of Perl Builder, the first integrated development environment (IDE)for Perl. [Jul 22, 1998]

Java

Language Development

This Fortnight in Perl 6 (17 - 30 June 2001)

A detailed summary of a recent Perl vs. Java battle, a discussion on the internal API for strings, and much more. [Jul 3, 2001]

This Week in Perl 6 (10 - 16 June 2001)
Even More on Unicode and Regexes, Multi-Dimensional Arrays and Relational Databases, and much more. [Jun 19, 2001]

This Week in Perl 6 (3 - 9 June 2001)
A discussion on the interaction of properties with use strict, continuing arguements surrounding regular expressions, and much more. [Jun 12, 2001]

This Week on p5p 2001/06/03
Improving the Perl test suite, Warnings crusade, libnet in the core, and much more. [Jun 4, 2001]

This Week on p5p 2001/05/27
Attribute tieing, Test::Harness cleanup, and much more. [May 27, 2001]

Taking Lessons From Traffic Lights
Michael Schwern examines traffic lights and shows what lessons applied to the development of Perl 6 [May 22, 2001]

This Month on Perl6 (1 May - 20 May 2001)
Perl 6 Internals, Meta, Language, Docs Released, and much more. [May 21, 2001]

This Week on p5p 2001/05/13
The to-do list, safe signals, release numbering and much more. [May 13, 2001]

This Week on p5p 2001/05/20
Internationalisation, Legal FAQ, and much more. [May 6, 2001]

This Week on p5p 2001/05/06
iThreads, Relocatable Perl, Module License Registration, and much more. [May 6, 2001]

This Week on p5p 2001/04/29
MacPerl 5.6.1, Licensing Perl modules, and much more. [Apr 29, 2001]

This Week on p5p 2001/04/22
Modules in the core, Kwalitee control, and much more. [Apr 22, 2001]

This Week on p5p 2001/04/15
perlbug Administration, problems with tar, and much more. [Apr 15, 2001]

This Week on p5p 2001/04/08
Perl 5.6.1 and Perl 5.7.1 Released(!), and much more. [Apr 8, 2001]

This Week on p5p 2001/04/02
Perl and HTML::Parser, Autoloading Errno, Taint testing, and much more. [Apr 2, 2001]

This Week on p5p 2001/04/01
Perl and HTML::Parser, Autoloading Errno, Taint testing, and much more. [Apr 1, 2001]

This Week on p5p 2001/03/26
use Errno is broken, Lexical Warnings, Scalar repeat bug, and much more. [Mar 26, 2001]

This Month on Perl6 (25 Feb--20 Mar 2001)
Internal Data Types, API Conventions, and GC once again. [Mar 21, 2001]

This Week on p5p 2001/03/19
Robin Houston vs. goto, more POD nits, and much more. [Mar 19, 2001]

This Week on p5p 2001/03/12
Pod questions, patching perly.y, EBCDIC and Unicode, plus more. [Mar 12, 2001]

This Week on p5p 2001/03/05
Coderef @INC, More Memory Leak Hunting, and more. [Mar 5, 2001]

This Week on p5p 2001/02/26
Overriding +=, More Big Unicode Wars, and more. [Feb 28, 2001]

This Week on p6p 2001/02/18
A revisit to RFC 88, quality assurance, plus more. [Feb 18, 2001]

This Week on p5p 2001/02/12
Perl FAQ updates, memory leak plumbing, and more. [Feb 14, 2001]

This Week on p5p 2001/02/06
Perl 5.6.1 not delayed after all, MacPerl, select() on Win32, and more. [Feb 6, 2001]

This Week on p5p 2001/01/28
5.6.x delayed, the hashing function, PerlIO programming documentation, and more. [Jan 30, 2001]

This Week on p5p 2001/01/21
Safe signals; large file support; pretty-printing and token reporting. [Jan 24, 2001]

This Week on p5p 2001/01/14
Unicode is stable! Big performance improvements! Better lvalue subroutine support! [Jan 15, 2001]

This Fortnight on p5p 2000/12/31
Unicode miscellany; lvalue functions. [Jan 9, 2001]

This Week on p5p 2000/12/24
5.6.1 trial release; new repository browser; `use constant` [Dec 27, 2000]

This Week on p5p 2000/12/17
More Unicode goodies; better arithmetic; faster object destruction. [Dec 17, 2000]

This Week on p5p 2000/12/10
Unicode support almost complete! Long standing destruction order bug fixed! Rejoice! [Dec 11, 2000]

This Week on p5p 2000/12/03
Automatic transliteration of Russian; syntactic oddities; lvalue subs. [Dec 4, 2000]

This Week on p5p 2000/11/27
Enhancements to `for`, `map`, and `grep`; Unicode on Big Iron; Low-Hanging Fruit. [Nov 27, 2000]

This Week on p5p 2000/11/20
Major regex engine enhancements; more about `perlio`; improved `subs.pm`. [Nov 20, 2000]

This Week on p5p 2000/11/14
`lstat _`; more about `perlio`; integer arithmetic. [Nov 14, 2000]

This Week on p5p 2000/11/07
Errno.pm error numbers; more self-tying; stack exhaustion in the regex engine. [Nov 7, 2000]

This Week on p5p 2000/10/30
More Unicode; self-tying; Perl's new built-in standard I/O library. [Oct 30, 2000]

These Weeks on p5p 2000/10/23

Perl's Unicode model; sfio; regex segfaults; naughty `use vars` calls.
[Oct 23, 2000]

This Week on p5p 2000/10/08
Self-tying is broken; integer and floating-point handling; why `unshift` is slow. [Oct 8, 2000]

This Week on p5p 2000/07/09
The Perl bug database; `buildtoc`; proposed `use namespace` pragma; a very clever Unicode hack. [Jul 9, 2000]

This Week on p5p 2000/07/02
Lots of Unicode; method lookup optimizations; `my __PACKAGE__ $foo`. [Jul 2, 2000]

Notes on Doug's Method Lookup Patch
Simon Cozens explains the technical details of a patch that was sent to p5p this month. [Jun 27, 2000]

This Week on p5p 2000/06/25
More method call optimization; tr///CU is dead; Lexical variables and eval; perlhacktut. [Jun 25, 2000]

This Week on p5p 2000/06/18

Method call optimizations; more bytecode; more unicode source files; EPOC port. [Jun 17, 2000]

This Week on p5p 2000/06/11

Unicode byte-order marks in Perl source code; many not-too-difficult bugs for entry-level Perl core hackers. [Jun 13, 2000]

This Week on p5p 2000/06/04

Farewell to Ilya Zakharevich; bytecode compilation; optimizations to `map`. [Jun 4, 2000]

This Week on p5p 2000/05/28

Regex engine alternatives and optimizations; `eq` and UTF8; Caching of `get*by*` functions; Array interpolation semantics. [May 28, 2000]

This Week on p5p 2000/05/21

What happened on the perl5-porters mailing list between 15 and 21 May, 2000 [May 21, 2000]

Perl Meets COBOL

I taught a Perl class to some IBM mainframe programmers whose only experience was in COBOL, and got some surprises. [May 15, 2000]

This Week on p5p 2000/05/14

What happened on the perl5-porters mailing list between 8 and 14 May, 2000 [May 14, 2000]

This Week on p5p 2000/05/07

What happened on the perl5-porters mailing list between 1 and 7 May, 2000 [May 7, 2000]

This Week on p5p 2000/04/30

What happened on the perl5-porters mailing list between 24 and 30 April, 2000 [Apr 30, 2000]

This Week on p5p 2000/04/23

What happened on the perl5-porters mailing list between 17 and 23 April, 2000 [Apr 23, 2000]

This Week on p5p 2000/03/05

What happened on the perl5-porters mailing list between 28 February and 5 March, 2000 [Mar 5, 2000]

This Week on p5p 1999/12/26

What happened on the perl5-porters mailing list between 20 and 26 December, 1999 [Dec 26, 1999]

This Week on p5p 1999/12/19

What happened on the perl5-porters mailing list between 13 and 19 December, 1999 [Dec 19, 1999]

This Week on p5p 1999/12/12

What happened on the perl5-porters mailing list between 6 and 12 December, 1999 [Dec 12, 1999]

This Week on p5p 1999/12/05

What happened on the perl5-porters mailing list between 29 November and 5 December, 1999 [Dec 5, 1999]

This Week on p5p 1999/11/28

What happened on the perl5-porters mailing list between 22 and 28 November, 1999 [Nov 28, 1999]

This Week on p5p 1999/11/21

What happened on the perl5-porters mailing list between 15 and 21 November, 1999 [Nov 21, 1999]

This Week on p5p 1999/11/14

What happened on the perl5-porters mailing list between 8 and 14 November, 1999 [Nov 14, 1999]

This Week on p5p 1999/11/07

What happened on the perl5-porters mailing list between 1 and 7 November, 1999 [Nov 7, 1999]

This Week on p5p 1999/10/31

What happened on the perl5-porters mailing list between 25 and 31 October, 1999 [Nov 3, 1999]

This Week on p5p 1999/10/17

What happened on the perl5-porters mailing list between 11 and 17 October, 1999 [Oct 17, 1999]

This Week on p5p 1999/10/24

What happened on the perl5-porters mailing list between 18 and 24 October, 1999 [Oct 17, 1999]

Topaz: Perl for the 22nd Century

Chip Salzenberg, one of the core developers of Perl, talks about Topaz, a new effort to completely rewrite the internals of Perl in C++. The complete version of his talk (given at the 1999 O'Reilly Open Source Conference) is also available in Real Audio. [Sep 28, 1999]

Larry Wall

State of the Onion 2000

Larry Wall's annual report on the state of Perl, from TPC 4.0 (the fourth annual Perl conference) in Monterey in July 2000. In this full length transcript, Larry talks about the need for changes, which has led to the effort to rewrite the language in Perl 6. [Oct 24, 2000]

3rd State of the Onion

Larry explains the "good chemistry" of the Perl community in his third State of the Onion speech. [Aug 30, 1999]

Perl, the first postmodern computer language

Larry Wall's talk at Linux World justifies Perl's place in postmodern culture. He says that he included in Perl all the features that were cool and left out all those that sucked. [Mar 9, 1999]

2nd State of the Onion

Larry Wall's keynote address from 1998 Perl Conference. There is also a RealAudio version. [Aug 25, 1998]

The Culture of Perl

In this keynote address for the first Perl Conference, Larry Wall talks about the key ideas that influence him and by extension the Perl culture. [Aug 20, 1997]

Lists

Guide to the Perl 6 Working Groups

Perl 6 discussion and planning are continuing at a furious rate and will probably continue to do so, at least until next month when Larry announces the shape of Perl 6 at the Linux Expo. In the meantime, here's a summary of the main Perl 6 working groups and discussion lists, along with an explanation of what the groups are about. [Sep 5, 2000]

Perl Mailing Lists

Perl newsgroups and mailing lists [Mar 29, 1999]

Modules

Pod::Parser Notes

Brad Appleton, author of the `Pod::Parser` module suite, responds to some of the remarks in an earlier perl5-porters mailing list summary. [May 20, 2000]

mod_perl

Creating Modular Web Pages With EmbPerl

If you have ever wished for an "include" HTML tag to reuse large chunks of HTML, you are in luck. Neil Gunton explains how Embperl solves the problem. [Mar 13, 2001]

Objects

Bless My Referents

Damian Conway explains how references and referents relate to Perl objects, along with examples of how to use them when building objects. [Sep 16, 1999]

Oddities

How Perl Helped Me Win the Office Football Pool

Walt Mankowski shows us how he used Perl to make a few extra bucks at the office. [Oct 2, 2000]

Open Source

Open Source Highlights
An open source champion inside Chevron writes about his visit to the Open Source Conference. [Sep 28, 1999]

Perl 6

This Week in Perl 6 (27 May - 2 June 2001)
Coding Conventions Revisited, Status of the Perl 6 Mailing Lists, and much more. [Jun 4, 2001]

Exegesis 2
Having trouble visualizing how the approved RFC's for Perl 6 will translate into actual Perl code? Damian Conway provides and exegesis to Larry Wall's Apocalypse 2 and reveals what the code will look like. [May 15, 2001]

Apocalypse 1: The Ugly, the Bad, and the Good
With breathless expectation, the Perl community has been waiting for Larry Wall to reveal how Perl 6 is going to take shape. In the first of a series of "apocalyptic" articles, Larry reveals the ugly, the bad, and the good parts of the Perl 6 design process. [Apr 2, 2001]

Report on the Perl 6 Announcement
At the Perl conference, Larry announced plans to develop Perl 6, a new implementation of Perl, starting over from scratch. The new Perl will fix many of the social and technical deficiencies of Perl 5. [Jul 25, 2000]

Perl Internals

Sapphire
Can one person rewrite Perl from scratch? [Sep 19, 2000]

Programming

Beginners Intro to Perl - Part 6
Doug Sheppard shows us how to activate Perl's built in security features. [Jan 9, 2001]

Beginners Intro to Perl - Part 5
Doug Sheppard discusses object-oriented programming in part five of his series on beginning Perl. [Dec 18, 2000]

Beginners Intro to Perl - Part 4
Doug Sheppard teaches us CGI programming in part four of his series on beginning Perl. [Dec 6, 2000]

Beginner's Introduction to Perl - Part 3
The third part in a new series that introduces Perl to people who haven't programmed before. This week: Patterns and pattern matching. If you weren't sure how to get started with Perl, here's

your chance! [Nov 20, 2000]

Program Repair Shop and Red Flags

Once again I take a real program written by a genuine novice and show how to clean it up and make it better. This time we turn a perl4 library into a Perl 5 object-oriented module. I show how to recognize some "red flags" that are early warning signs that you might be doing some of the same things wrong in your own programs. [Nov 14, 2000]

Beginner's Introduction to Perl - Part 2

The second part in a new series that introduces Perl to people who haven't programmed before. This week: Files and strings. If you weren't sure how to get started with Perl, here's your chance! [Nov 7, 2000]

Beginner's Introduction to Perl

The first part in a new series that introduces Perl to people who haven't programmed before. If you weren't sure how to get started with Perl, here's your chance! [Oct 16, 2000]

Return of Program Repair Shop and Red Flags

My other 'red flags' article was so popular that once again I've taken a real program written by a genuine novice and shown how to clean it up and make it better. I show how to recognize some "red flags" that are early warning signs that you might be doing some of the same things wrong in your own programs. [Jun 17, 2000]

Program Repair Shop and Red Flags

I've taken a real program written by a genuine novice and shown how to clean it up and make it better. I also show how to recognize some "red flags" that are early warning signs that you might be doing some of the same things wrong in your own programs. [May 2, 2000]

My Life With Spam: Part 3

In the third part of a tutorial on how to filter spam, Mark-Jason Dominus reveals how he relegates mail to his "losers list, blacklist and whitelist." [Mar 15, 2000]

My Life With Spam

In the second part of a tutorial on how to filter spam, Mark-Jason Dominus shows what to do with spam once you've caught it. [Feb 9, 2000]

Recipe of the Day

Each day, we present a new recipe from The Perl Cookbook, the best-selling book by Tom Christiansen and Nathan Torkington. [Aug 26, 1999]

Enhancements for Prototypes

Tim Bunce's thoughts on how prototypes can be made more general and more useful. [Aug 12, 1999]

Prototypes in Perl

Tom Christiansen explains how prototypes *really* work in Perl. [Jul 25, 1999]

Perl's Prospects Are Brighter Than Ever
Jon Udell's summary of the Perl Conference. [Aug 26, 1998]

MacPerl Gains Ground
MacPerl gains a foothold on a machine without a command-line interface. Rich Morin of Prime Time Freeware and Matthias Neeracher, the programmer who ported Perl to the Macintosh, talk about what makes MacPerl different. [Jun 3, 1998]

Java vs Penguin
An exchange that Jake had with a colleague regarding penguin/perl/java. [Aug 15, 1997]

Tcl vs. Perl
Tom explains the execution time of a typical Perl program as compared to a C program that does the same thing. As a bonus, he throws in a comparison to tcl. [Aug 5, 1997]

tchrist's Tcl Dirty Laundry List
More venting from a previous discussion. Tom writes, "TEE CEE ELL is merely a string substitution language in a lot of ways. Here are some of my rags on tcl. I know their answers and I don't like them." [Aug 5, 1997]

Tcl vs. Perl Comparison
Aaron compares TCL and perl as general purpose scripting languages. [Aug 5, 1997]

Java Über Alles
Why Java seems to hard for "accidental programmer." [Aug 5, 1997]

Java is Dead
Early safeperl discussions. [Aug 5, 1997]

Csh Programming Considered Harmful
The following periodic article answers in excruciating detail the frequently asked question "Why shouldn't I program in csh?". It is available for anon FTP from convex.com in /pub/csh.whynot . [Dec 1, 1996]

Seven Deadly Sins of Perl
Tom's list of what is "suboptimal" in Perl from a reasonably serious programming languages design point of view. [Nov 1, 1996]

Regular Expressions

Releases

Perl 6 Alive and Well! Introducing the perl6-mailing-lists Digest
Perl.com will be supplying you with the P6P digest, covering the latest news on the development of Perl 6. [Feb 14, 2001]

What's New in 5.6.0.
After two years in the making, we look at new features of Perl,

including support for UTF-8 Unicode and Internet address constants. [Apr 18, 2000]

Happy Birthday Perl 5!
[Oct 18, 1999]

Downloading the Latest Version of Perl
Aquiring Perl software [Mar 24, 1999]

Security

Style Guides

Red Flags Return
Readers pointed out errors and suggested more improvements to the code in my 'Red Flags' articles. As usual, there's more than one way to do it! [Nov 28, 2000]

In Defense of Coding Standards
Perl programmers may bristle at the idea of coding standards. Fear not: a few simple standards can improve teamwork without crushing creativity. [Jan 12, 2000]

Text Tools

Designing a Search Engine
Pete Sergeant discusses two elements of designing a search engine: how to store and retrieve data efficiently, and how to parse search terms. [Apr 10, 2001]

Creating Data Output Files Using the Template Toolkit
Dave Cross explains why you should add the Template Toolkit to your installation of Perl and why it is useful for more than just dynamic web pages. [Jan 23, 2001]

Troubleshooting

Having Trouble Printing Code Examples?
Info for those who can't get Perl.com articles to print out correctly [Jun 11, 2001]

Tutorials

Parse::RecDescent Tutorial
Parse::RecDescent is a recursive descent parser generator designed to help to Perl programmers who need to deal with any sort of structured data,from configuration files to mail headers to almost anything. It's even been used to parse other programming languages for conversion to Perl. [Jun 13, 2001]

A Beginner's Introduction to POE
Interested in event-driven Perl? Dennis Taylor and Jeff Goff show

us how to write a simple server daemon using POE, the Perl Object Environment. [Jan 17, 2001]

User Interfaces

A Simple Gnome Panel Applet
Build a useful Gnome application in an afternoon! Joe Nasal explains some common techniques, including widget creation, signal handling, timers, and event loops. [Mar 27, 2001]

Writing GUI Applications in Perl/Tk
Nick Temple shows how to program a graphical Point-of-Sale application in Perl, complete with credit card processing. [Mar 6, 2001]

Programming GNOME Applications with Perl - Part 2
Simon Cozens shows us how to use Perl to develop applications for Gnome, the Unix desktop environment. [Nov 28, 2000]

Programming GNOME Applications with Perl
Simon Cozens shows us how to use Perl to develop applications for Gnome, the Unix desktop environment. [Oct 16, 2000]

Perl/Tk Tutorial
On Perl.com, we are presenting this as part of what we hope will be an ongoing series of articles, titled "Source Illustrated." The presentation by Lee and Brett is a wonderfully concise example of showing annotated code and its result. [Oct 15, 1999]

Web Development

Quick Start Guide with SOAP Part Two
Paul Kulchenko continues his SOAP::Lite guide and shows how to build more comples SOAP servers. [Apr 23, 2001]

Quick Start with SOAP
An introduction to SOAP::Lite, a module that provides simple yet flexible interface to SOAP, a popular XML-RPC protocol. Using SOAP::Lite, Perl scripts can access objects and execute procedures on remote servers, and also serve SOAP objects and procedures over the 'Net. [Jan 29, 2001]

Win32

Microsoft to Fund Perl Development
ActiveState Tool Corp. has a new three-year agreement with Microsoft that funds new development of Perl for the Windows platform. [Jun 9, 1999]

A Zero Cost Solution
Creating a task tracking system for $0 in licensing fees, hardware, and software costs. [Nov 17, 1998]

XML

MSXML, It's Not Just for VB Programmers Anymore
Shawn Ribordy puts the tech back into the MSXML parser by using Perl instead of Visual Basic. [Apr 17, 2001]

What every Perl programmer needs to know about .NET
A very brief explanation of Microsoft's .NET project and why it's interesting. [Dec 19, 2000]

RSS and You
RSS is an XML application that describes web sites as channels, which can act as feeds to a user's site. Chris Nandor explains how to use RSS in Perl and how he uses it to build portals. [Jan 25, 2000]

XML::Parser Module Enables XML Development in Perl
The new Perl module known as XML::Parser allows Perl programmers building applications to use XML, and provides an efficient, easy way to parse XML documents. [Nov 25, 1998]

Perl Support for XML Developing
O'Reilly & Associates hosted a recent Perl/XML summit to discuss ways that Perl can support the Extensible Markup Language (XML), a new language for defining document markup and data formats. [Mar 10, 1998]

Y2K

Y2K Compliance
Is someone asking you to ensure that your Perl code is Y2k compliant? Tom Christiansen gives you some answers, which may not completely please the bureaucrats. [Jan 3, 1999]

O'REILLY

Search

Powered by Atomz

perl | Books | All the words | Search

Books matching your search for "perl"

Perl for System Administration

Description
Index
Reviews

Perl Cookbook

Description
Index
Reviews

Perl in a Nutshell

Description
Index
Contents
Reviews

Perl Resource Kit -- Win32 Edition

Index
Contents
Reviews

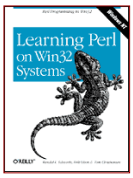
Learning Perl, 3rd Edition

Description
Index
Contents

Advanced Perl Programming

Description
Index
Contents
Reviews

Learning Perl on Win32 Systems



Description
Index
Contents
Reviews

Learning Perl, 2nd Edition

Description
Index
Contents
Reviews

Programming Perl, 3rd Edition

Description
Index
Contents
Reviews

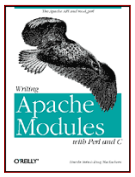
Programming the Perl DBI

Description
Index
Reviews

Programming Perl, 2nd Edition

Index
Contents

Writing Apache Modules with Perl and C

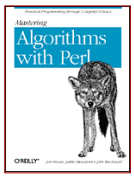


Description
Index
Contents

CGI Programming with Perl, 2nd Edition

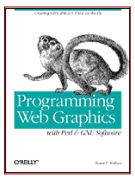
Description
Index
Contents
Reviews

Mastering Algorithms with Perl



Description
Index
Contents
Reviews

Programming Web Graphics with Perl & GNU Software



Description
Index
Reviews

Perl 5 Pocket Reference, 3rd Edition

Description
Reviews

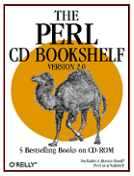
Perl/Tk Pocket Reference

Description
Contents

Learning Perl/Tk

Description
Index
Reviews

The Perl CD Bookshelf, Version 2.0



Description
Reviews

mod_perl Pocket Reference

Description
Contents
Reviews

Web Client Programming with Perl

Reviews

---

webmaster@oreilly.com

Downloads
Documentation
CPAN
FAQs
Training
Resources
Article Archive
Books
Search
Register/Log in

Columns

P5P Digest
P6P Digest
Off the Wall

Search Tips

Check Spelling
Make sure your search terms are spelled correctly.

Use multiple words
Using multiple words will return more refined results than a single word. For example, typing our free service will return more relevant results than typing just service. (Keep in mind that relevant results are returned even if they don't contain all query terms.)

Use similar words
The more similar words you use in a search, the more relevant your results will be.

Use appropriate capitalization
Capitalize proper nouns, and remember that lower-case words will match any case. For example, typing search will return all documents containing the words search, Search, and SEARCH. Typing Search, however, will instruct the search engine to look only for the capitalized word.

Use quotation marks
Use quotation marks to find words which must appear adjacent to each other, for example, "our pledge to you." Otherwise, the search results will include the word our, pledge, to, and the word you, but not necessarily in that order. The words may appear anywhere, and in any order, within the document.

Use plus (+) or minus (-)
Use a plus sign when your search term or phrase must appear in the search results. Use a minus sign to indicate undesirable term(s). The plus sign tells the search engine that a certain word or phrase is required in the search results, and a minus sign indicates that a word or phrase must be absent in the search results.

Note: A phrase must be contained within quotation marks. Leave no spaces between the plus or minus sign and the term.

Use wildcards
Wildcard searches can expand the number of matches for a particular request. The * character is used as the wildcard character.

For instance, searching for wh* will find the words what, why, when, whether, and any other word that starts with wh. Searching for *her* will find the words here, whether, together, gathering, and any other word that contains her anywhere in the word.

Wildcards may be combined with the standard plus (+) and minus (-) modifiers, quotes for phrases, as well as the field search specifiers. +wh* -se*ch will find all pages which have a word that starts with wh and which does not contain a word that starts with se and ends with ch. "wh* are" will find the phrases where are, what are, why are, etc.

Searching for Perl code or modules? Try search.cpan.org

Log in

Members: Log in below to access your user account or to use our Comment feature.
Visitors: **Join now!** and set up your user account.

If you are having difficulties logging in, make sure your browser is set to accept cookies.
Forgotten your password? Need Help?

Login Name **Password**

Login

Downloads

Documentation

CPAN

FAQs

Training

Resources

Article Archive

Books

Search

Register/Log in

Columns

P5P Digest
P6P Digest
Off the Wall

The P5P Digest

This digest, published weekly on Sundays, summarizes the activity on Perl 5 Porters list, the main mailing list for the core developers of the Perl language.

This Week on p5p 2001/07/09

No 5.8.0 yet, numeric hackery, worries about PerlIO and much more. [Jul. 9, 2001]

This Week on p5p 2001/07/02

Module versioning and testing, regex capture-to-variable, and much more. [Jul. 2, 2001]

This Week on p5p 2001/06/25

5.7.2 in sight, some threads on regular expression, and much more. [Jun. 25, 2001]

This Week on p5p 2001/06/17

Miscellaneous Darwin Updates, hash accessor macros, and much more. [Jun. 19, 2001]

This Week on p5p 2001/06/09

Removing dependence on strtol, regex negation, and much more. [Jun. 12, 2001]

This Week on p5p 2001/06/03

Improving the Perl test suite, Warnings crusade, libnet in the core, and much more. [Jun. 4, 2001]

This Week on p5p 2001/05/27

Attribute tieing, Test::Harness cleanup, and much more. [May. 27, 2001]

This Week on p5p 2001/05/13

The to-do list, safe signals, release numbering and much more. [May. 13, 2001]

This Week on p5p 2001/05/06

iThreads, Relocatable Perl, Module License Registration, and much more. [May. 6, 2001]

This Week on p5p 2001/04/29

MacPerl 5.6.1, Licensing Perl modules, and much more. [Apr.

29, 2001]

This Week on p5p 2001/04/22
Modules in the core, Kwalitee control, and much more. [Apr. 22, 2001]

This Week on p5p 2001/04/15
perlbug Administration, problems with tar, and much more. [Apr. 15, 2001]

This Week on p5p 2001/04/08
Perl 5.6.1 and Perl 5.7.1 Released(!), and much more. [Apr. 8, 2001]

This Week on p5p 2001/04/02
Perl and HTML::Parser, Autoloading Errno, Taint testing, and much more. [Apr. 2, 2001]

This Week on p5p 2001/04/01
Perl and HTML::Parser, Autoloading Errno, Taint testing, and much more. [Apr. 1, 2001]

This Week on p5p 2001/03/26
use Errno is broken, Lexical Warnings, Scalar repeat bug, and much more. [Mar. 26, 2001]

This Week on p5p 2001/03/19
Robin Houston vs. goto, more POD nits, and much more. [Mar. 19, 2001]

This Week on p5p 2001/03/12
Pod questions, patching perly.y, EBCDIC and Unicode, plus more. [Mar. 12, 2001]

This Week on p5p 2001/03/05
Coderef @INC, More Memory Leak Hunting, and more. [Mar. 5, 2001]

This Week on p5p 2001/02/26
Overriding +=, More Big Unicode Wars, and more. [Feb. 28, 2001]

This Week on p5p 2001/02/12
Perl FAQ updates, memory leak plumbing, and more. [Feb. 14, 2001]

This Week on p5p 2001/02/06
Perl 5.6.1 not delayed after all, MacPerl, select() on Win32, and more. [Feb. 6, 2001]

This Week on p5p 2001/01/28
5.6.x delayed, the hashing function, PerlIO programming

documentation, and more. [Jan. 30, 2001]

This Week on p5p 2001/01/21

Safe signals; large file support; pretty-printing and token reporting. [Jan. 24, 2001]

This Week on p5p 2001/01/14

Unicode is stable! Big performance improvements! Better lvalue subroutine support! [Jan. 15, 2001]

This Fortnight on p5p 2000/12/31

Unicode miscellany; lvalue functions. [Jan. 9, 2001]

This Week on p5p 2000/12/24

5.6.1 trial release; new repository browser; `use constant` [Dec. 27, 2000]

This Week on p5p 2000/12/17

More Unicode goodies; better arithmetic; faster object destruction. [Dec. 17, 2000]

This Week on p5p 2000/12/10

Unicode support almost complete! Long standing destruction order bug fixed! Rejoice! [Dec. 11, 2000]

This Week on p5p 2000/12/03

Automatic transliteration of Russian; syntactic oddities; lvalue subs. [Dec. 4, 2000]

This Week on p5p 2000/11/27

Enhancements to `for`, `map`, and `grep`; Unicode on Big Iron; Low-Hanging Fruit. [Nov. 27, 2000]

This Week on p5p 2000/11/20

Major regex engine enhancements; more about `perlio`; improved `subs.pm`. [Nov. 20, 2000]

This Week on p5p 2000/11/14

`lstat _`; more about `perlio`; integer arithmetic. [Nov. 14, 2000]

This Week on p5p 2000/11/07

Errno.pm error numbers; more self-tying; stack exhaustion in the regex engine. [Nov. 7, 2000]

This Week on p5p 2000/10/30

More Unicode; self-tying; Perl's new built-in standard I/O library. [Oct. 30, 2000]

These Weeks on p5p 2000/10/23

Perl's Unicode model; sfio; regex segfaults; naughty `use vars` calls.

[Oct. 23, 2000]

This Week on p5p 2000/10/08

Self-tying is broken; integer and floating-point handling; why `unshift` is slow. [Oct. 8, 2000]

This Week on p5p 2000/07/09

The Perl bug database; `buildtoc`; proposed `use namespace` pragma; a very clever Unicode hack. [Jul. 9, 2000]

This Week on p5p 2000/07/02

Lots of Unicode; method lookup optimizations; `my __PACKAGE__ $foo`. [Jul. 2, 2000]

Notes on Doug's Method Lookup Patch

Simon Cozens explains the technical details of a patch that was sent to p5p this month. [Jun. 27, 2000]

This Week on p5p 2000/06/25

More method call optimization; tr///CU is dead; Lexical variables and eval; perlhacktut. [Jun. 25, 2000]

This Week on p5p 2000/06/18

Method call optimizations; more bytecode; more unicode source files; EPOC port. [Jun. 17, 2000]

This Week on p5p 2000/06/11

Unicode byte-order marks in Perl source code; many not-too-difficult bugs for entry-level Perl core hackers. [Jun. 13, 2000]

This Week on p5p 2000/06/04

Farewell to Ilya Zakharevich; bytecode compilation; optimizations to `map`. [Jun. 4, 2000]

This Week on p5p 2000/05/28

Regex engine alternatives and optimizations; `eq` and UTF8; Caching of `get*by*` functions; Array interpolation semantics. [May. 28, 2000]

This Week on p5p 2000/05/21

What happened on the perl5-porters mailing list between 15 and 21 May, 2000 [May. 21, 2000]

Pod::Parser Notes

Brad Appleton, author of the `Pod::Parser` module suite, responds to some of the remarks in an earlier perl5-porters mailing list summary. [May. 20, 2000]

This Week on p5p 2000/05/14

What happened on the perl5-porters mailing list between 8 and

14 May, 2000 [May. 14, 2000]

[This Week on p5p 2000/05/07]
What happened on the perl5-porters mailing list between 1 and 7 May, 2000 [May. 7, 2000]

[This Week on p5p 2000/04/30]
What happened on the perl5-porters mailing list between 24 and 30 April, 2000 [Apr. 30, 2000]

Downloads

Documentation

CPAN

FAQs

Training

Resources

Article Archive

Books

Search

Register/Log in

Columns

P5P Digest
P6P Digest
Off the Wall

P6P Digest

The latest from the P6P Digest

This Fortnight in Perl 6 (17 - 30 June 2001)
A detailed summary of a recent Perl vs. Java battle, a discussion on the internal API for strings, and much more. [Jul. 3, 2001]

This Week in Perl 6 (10 - 16 June 2001)
Even More on Unicode and Regexes, Multi-Dimensional Arrays and Relational Databases, and much more. [Jun. 19, 2001]

This Week in Perl 6 (3 - 9 June 2001)
A discussion on the interaction of properties with use strict, continuing arguements surrounding regular expressions, and much more. [Jun. 12, 2001]

This Week in Perl 6 (27 May - 2 June 2001)
Coding Conventions Revisited, Status of the Perl 6 Mailing Lists, and much more. [Jun. 4, 2001]

This Week in Perl 6 (20 May - 26 May 2001)
Perl Virtual Registers, slice syntax, and much more. [May. 26, 2001]

This Month on Perl6 (1 May - 20 May 2001)
Perl 6 Internals, Meta, Language, Docs Released, and much more. [May. 21, 2001]

This Month on Perl6 (25 Feb--20 Mar 2001)
Internal Data Types, API Conventions, and GC once again. [Mar. 21, 2001]

This Week on p6p 2001/02/18
A revisit to RFC 88, quality assurance, plus more. [Feb. 18, 2001]

Perl 6 Alive and Well! Introducing the perl6-mailing-lists Digest
Perl.com will be supplying you with the P6P digest, covering the latest news on the development of Perl 6. [Feb. 14, 2001]

For problems or assistance with this site, email help@www.perl.com

Downloads

Documentation

CPAN

FAQs

Training

Resources

Article Archive

Books

Search

Register/Log in

Columns

P5P Digest
P6P Digest
Off the Wall

Off The Wall

A monthly column from Larry Wall

Larry Wall: Apocalypse Two

Larry Wall produces the next episode in his series of "Apocalypses": glimpses into the design of Perl 6. This week, he explains how Perl 6 will differ from Perl 5 in terms of chapter 2 of the Camel Book: fundamental data types, variables and the context and scoping of the language. [May. 3, 2001]

Apocalypse 1: The Ugly, the Bad, and the Good

With breathless expectation, the Perl community has been waiting for Larry Wall to reveal how Perl 6 is going to take shape. In the first of a series of "apocalyptic" articles, Larry reveals the ugly, the bad, and the good parts of the Perl 6 design process. [Apr. 2, 2001]

Downloads
Documentation
CPAN
FAQs
Training
Resources
Article Archive
Books
Search
Register/Log in

Columns
P5P Digest
P6P Digest
Off the Wall

This Week on p5p 2001/07/09

Print This Article

by Simon Cozens
July 09, 2001

Notes

Please send corrections and additions to perl-thisweek-YYYYMM@simon-cozens.org where YYYYMM is the current year and month. Changes and additions to the perl5-porters biographies are particularly welcome.

This was a reasonably busy week, seeing just over 400 messages. I say that every week, don't I?

This Week on P5P

No 5.8.0 Yet

Jarkko sadly announced that 5.8.0 wasn't going to happen before the Perl Conference, but 5.7.2 is imminent:

> I think it's time for me to give up the fantasy that 5.8.0 will happen before the Perl Conference. The stars just refuse to align properly, too many loose ends, too many delays, too many annoying mysterious little problems, too little testing of CPAN modules. Luckily, nothing is majorly broken, and I think that I've more or less achieved what I set out to do with Perl, so I still hope to be able wrap something up before TPC and call it 5.7.2 (it must happen next week or it will not happen), and soon after the conference put out the Release Candidate 1 for 5.8.0, and then keep cracking the whip till we are happy with what we've got.

New Modules

These aren't really very new, but they may have slipped through the net and you haven't noticed them yet, and since they're interesting, you might want to have a look...

`I18N::LangTags` detects and manipulates RFC3066 language tags; `Locale::Maketext` is extremely useful for localizing text; `Unicode::UCD` is a neat interface to the Unicode Character Database; `Encode` is coming on strong, and can now read IBM ICU character tables; Mark-Jason Dominus' `Memoize` module is now part of the core.

Numbers, numbers, numbers

Remember last week's weird Amdahl UTS bug, where Nicholas Clark was convinced UTS C was doing a decrement statement twice? He found the problem - the decrement statement shouldn't have been there at all...

This prompted him to find a bug in `grok_number`; this surprised me a little, because I didn't know that `grok_number` even existed. All of the useful, platform-independent code which deals with numeric operations - casting between different sizes, converting binary, hex, and octal numbers, recognising numbers in strings, and so on, has been moved to `numeric.c`. Take a look at it, there's a load of handy stuff in there.

Hal Morris, our UTS wizard, also pointed out some unpleasant casting assumptions, which needed a patch:

> `UV_MAX` must NOT be defined as `(unsigned long){whatever}` for UTS, because then comparisons with double will not work correctly (there is no problem with `(unsigned)` typecasts, only with `(unsigned long)`)

`grok_number` again came in handy on QNX, when Norton Allen found that `strtoul` wasn't setting `errno` correctly on overflow. It's sad when we have to start reimplementing people's broken C libraries, but this is the price of portability.

PerlIO considered evil

Ilya found a bug in PerlIO, then found another bug while attempting to demonstrate it. The original bug was:

> The *actual* problem is that char-by-char input requires DUPLICATE pressing of ENTER key for this key to be seen by Perl. Debugging this problem (via Term::ReadKey test suite) shows the following logic:
>
> pp_getc() calls is_eof() which does getc/ungetc calls getc()
>
> [BTW, I see no logic in this sequence of events.]
>
> The problem is that ungetc() can't unget "\n" if this \n is the first char in the buffer, and quietly drops "\n" to the floor.

Ilya had a lot of invective set aside for PerlIO, which we need not go into. Needless to say, he did not provide an alternative implementation of a multi-layered standard IO system as a patch. Or indeed any patch at all.

Vadim Konovalov did provide a simple patch to clean something up, but then Andy and Nick both showed that it didn't help at all, the generated code being the same and some compilers not being able to cope with lvalue casts, which Nick had carefully removed and Vadim's patch reintroduced. Guess Nick might actually know what he's doing after all.

Asynchronous Callbacks

David Lloyd asked how do safely do asynchronous callbacks from C to Perl. Bejamin Stuhl suggested hacking the core to introduce some checks during the inter-opcode `PERL_ASYNC_CHECK`, and suggested that Perl 5.8.x had a public way of registering inter-opcode callbacks. David Lloyd replied that PHP/Zend already had this, and you could even implement a signal checking add-on module without any core hacking. Paul Johnson went one further, and suggested using a pluggable `runops` routine. Surprisingly, this has actually been implemented but nobody really knows about it; `Devel::Cover` apparently makes use of it. Of course, the problem is that only one thing can use a custom op loop at a time, so David suggested writing an XS module that allowed other modules to add callbacks. I hope that happens.

Artur Bergman popped up and, predictably, suggested using ithreads.

Various

Rudi Farkas found a weird one on Win32 - on that platform, executableness (the `-x` test) is determined by the filename being examined. For instance, `foo.bat` is classed as executable, but `foo.bar` is not, even if they contain exactly the same data. This rather curious design decision leads to the fact that if you call `stat` with a filename, the execute bit is set depending on the extension. If, on the other hand, you call `fstat` with a filehandle, Windows can't retrieve the filename and test the extension, so it silently sets the execute bit to zero, no matter what it gets fed. This is Bad, and means that `-x _` on Windows is unpredictable. Radi provided a suggested workaround, but nobody cared enough about fixing something so obviously braindead to produce a patch.

Mike Schwern fixed up `MakeMaker` to stop producing extraneous `-I...`s when building extensions, and also found that the XS version of `Cwd` won't do much good as

a separate CPAN module, since it relies on the core function `sv_getcwd`, which only appears in 5.7.1. Oops. Oh, and speaking of `Cwd`, Ilya patched it up a bit for OS/2, while noting that its results were untainted on that platform.

Ilya also fixed a glaring debugger bug (oh, the irony) prompting Jarkko to lament the lack of a test suite. Robin fixed up a couple of weird, weird bugs in `B::Deparse`.

Jonathan Stowe upgraded `pl2pm` to be warnings and strict clean. I'm not sure why.

Philip Newton patched a score of typos. Norton Allen updated the QNX documentation and provided a couple of other fixes.

Piers Cawley found something that looked like a bug in `SUPER::` but was assured that it wasn't; Randal won the day with a reference to Smalltalk.

Abhijit Menon-Sen (look out for this guy...) made `mkdir` warn if it was given a non-octal literal, since that generally doesn't do what people want, and after prompting from Gisle, did the same for `umask` and `chmod`. Unfortunately, he forgot about constant folding...

Until next week I remain, your humble and obedient servant,

Simon Cozens

Downloads
Documentation
CPAN
FAQs
Training
Resources
Article Archive
Books
Search
Register/Log in

Columns
P5P Digest
P6P Digest
Off the Wall

People Behind Perl: Nathan Torkington

Print This Article

by Simon Cozens
July 03, 2001

So you use Perl, and you probably know that it was brought to you by "Larry Wall and a cast of thousands". But do you know these people that make up the Perl development team? Every month, we're going to be featuring an interview with a Perl Porter so you can get to know the people behind Perl. This time, Simon Cozens, www.perl.com editor, talks to Nathan Torkington, a long-time Perl developer and a mainstay of the Perl community.

Who are you and what do you do?

I'm Nathan Torkington. My day job is a book editor for O'Reilly and Associates. Before that I was a trainer, working for Tom Christiansen. Perl folks might know me as co-author (with Tom) of The Perl Cookbook, the schedule planner for The Perl Conference (and this year the Open Source Convention), or as the project manager for perl6.

How long have you been programming Perl?

Since the early 90s. I forget exactly when I first started programming in Perl. I think it was toward the end of 1992, when I was working with gopher and the early web. Back in those days we didn't talk about "the web", we were just trying to set up a Campus-Wide Information Service. I took the plunge and pushed for (and got) the www as the basis for the CWIS, even though Gopher was more mature and had more stuff on it.

Using the CERN httpd and line mode browser, I worked on interfacing a bunch of data sources to the web. Perl was, of course, the language for that. I did work with SGML, comma separated files, and even used oraperl once or twice (I feared and loathed Oracle, though, it was only once or twice!).

So I got into Perl with the web. When I moved to the US in 1996, I worked as a systems administrator at an ISP. There I rapidly became the Perl guy, writing admin scripts and CGI applications for in-house and customers. When I left in 1998, we were using mod_perl and had a bunch of good Perl programmers.

What, if anything, did you program before learning Perl?

I first learned Commodore BASIC when I was 8 or 9, then 6502 assembler. I learned Pascal on the IBM PC, as well as C and 8086 assembler. At university they taught us Pascal again, then Modula-2 or Modula-3 (whichever it was, it was with a MetroWerks Mac IDE that kept crashing). Through Pascal, and their eventual reteaching of C, I got the hang of pointers (yes, I wasn't a very GOOD assembly language programmer if it took me about six years to learn pointers and realize why my programs would sometimes die).

What got you into Perl?

The web. After I'd written a few programs, I really enjoyed it. The language was fun, and the culture good. It was so thrilling to be able to do something in 5 minutes and 20 lines that used to take two days in C. In some ways I miss that fun--over the years (and

especially writing the Cookbook) I've learned so much about how to do things in Perl that there's not much discovery of fun new features any more. And I'm so used to being able to do things in 5 minutes and 20 lines that I'm no longer delighted when I can do so--I **expect** it!

Why do you still prefer Perl?

It can do everything I want to do, and I already know it. If I wasn't a Perl programmer, I'd probably be happy in Ruby or Python. But so long as there's Perl, I see no reason to become as familiar with those languages as I am with Perl.

That's not to say I think everyone should program exclusively in Perl. My friend Jules was the one who learned a lot of languages. He was writing extensions for Microsoft COBOL in 1992, and has done significant projects in C++ and Java. He works at a contracting company where the projects don't always lend themselves to Perl. I'm totally cool with that.

What sort of things do you do with Perl now? What was the last Perl program you wrote?

When I became a trainer, I was glad to stop being a full-time programmer. Since then I've begun to miss using Perl. These days I only write utilities and basic web applications. For instance, the last few projects I've written have been:

- A small web-based database system for keeping track of the books I'm editing--author addresses, status, number of chapters to go, etc. That was a CGI program with home-grown templates and a DBM database.
- Some small tools to unmangle Framemaker files for the Perl Conference proceedings.
- A translation of a Python PDF-generating library to Perl. That's incomplete, because I struck Python code I need to research.

What coaxed you into the development side of Perl?

The feeling that I should know more about it. In some ways it's probably insecurity--"sure, you know all stuff about the Perl language, but can you tell an SV from an SUV?" So I started probing at the fringes of the internals.

Do you remember your first patch?

My first (and probably only ;-) real patch was to comment `toke.c`, the tokenizer. The code that works out where double-quoted strings and regular expressions end was a lot of fun (and by fun I mean "pain") to work out. Although I know a lot of low level (data structures) and high level principles about how the internals work (compile to op tree, interpret), I'm still missing a lot of the middle ground that would actually enable me to **fix** bugs.

What do you usually do for Perl development now?

Nothing, I'm a manager :-) I'm slowly herding Larry towards finishing the Apocalypses, and from that will spring perl 6. For Perl 5 I'm more behind the scenes. I'm often asked to nag or poke or otherwise get results from others.

Talking of Perl 6, how do you think the project's going so far?

Last year I'd naively hoped that we'd have an alpha for TPC, but that's not happening. Instead, we have the start of Larry's pronouncements. And while perl6 has been slower than we expected, perl5 has received a shot in the arm! Jarkko's patching like a madman, there are new internals hackers springing up, and there are cool new modules (`SOAP::Lite`, `Inline`, `Attributes::*`, `Filter::Simple`, etc.) coming out every week.

That's not to say that we're all blocked on Larry, though. We know the large shape (and

a lot of the details) of how the internals will work. Dan's specing those out in Perl Design Documents, and your fair self has implemented a few of the projected perl6 syntactic features with patches to Perl 5. (You can download the Perl 6 emulator and play about with Perl 6.) If you haven't already seen Marcel Grunauer's page of Perl6-like modules, check it out!

So there's a lot of activity in perl6 as well as in perl5.

Finally, what's the best thing about Perl, and what's the worst thing about it?

Best thing? The way that it values programmer fun as much as anything else. Perl delights in being a language that is supposed to be fun. Having seen many people burn out, fun is a good thing. Fun is what keeps you sane, keeps you interested, keeps you going.

What's the worst thing? Probably the internals. They're ugly and resemble nothing so much as a Lovecraftian horror. We really want perl6 to have much nicer innards. Ideally it'll be almost as simple to program in perl6 innards as it is to program in perl6 itself. That's one of the main reasons to have perl6--a cleaner core.

Downloads
Documentation
CPAN
FAQs
Training
Resources
Article Archive
Books
Search
Register/Log in

Columns
P5P Digest
P6P Digest
Off the Wall

This Fortnight in Perl 6 (17 - 30 June 2001)

Print This Article

by Bryan Warnock
July 03, 2001

You can subscribe to an email version of this summary by sending an empty message to perl6-digest-subscribe@netthink.co.uk.

Please send corrections and additions to bwarnock@capita.com.

The lists have been very light recently. During the last two weeks of June, three of the mailing lists received a mere 142 messages across 20 different threads. 40 different authors contributed. Only 5 threads generated much traffic. Eventually, I'll come up with a better way of reporting these meaningless metrics.

This Week

Perl Doesn't Suck

Adam Turoff provided a detailed summary of some recent battles against The Big Bean:

> Now, at the end of the day, I have no fewer than five JVMs installed, all completely different implementations of two Java standards. As a Perl programmer, I find this abhorrent. Installing any version of Perl release in the last 7 years is no different from installing any other release: download, extract, ./configure -des, make, make test, make install. Done.

Elaine Ashton, however, disagreed, to a point:

> I don't believe I was saying that. My point was that you had a bad experience installing Java on FreeBSD and have declared that it sucks to install it. Unsurprisingly, I have never had a problem installing or supporting Java on Solaris but there are plenty of things to grumble about Perl sometimes, especially if you deploy multiple versions and configurations across multiple platforms and multiple versions of those platforms.

Michael Schwern pointed out that Solaris is "Sun's Blessed Platform", and it shouldn't be surprising that Java should install easily there. The discussion then touched a bit on distributions, licensing, support roles, and, yes, even George Carlin.

Once Inherited, Twice Shy

Multiple Classifications

David Whipp asked if `bless` could take, and `ref` return, a list, allowing for a cleaner multiple-inheritance model for objects in Perl. Dan Sugalski simplified the request to object-based vice class-based inheritance, and then provided some potential trade-offs.

Damian, of course, submitted code to fake it in Perl 5. He did muse about an `ISA` property, though, which would act like `@ISA`, but at the object level.

Class::Object

Michael "Class::Object" Schwern asked why all this (Class::Object) had to be (Class::Object) in the core (Class::Object). Dan Sugalski opined:

> Doing it properly in a module is significantly more of a pain than doing it in the core. Faking it with a module means a fair amount of (reasonably slow) perl code, doing it in the core requires a few extra lines of C code in the method dispatch opcode function.

To which, of course, Michael Class::Objected:

> I've already done it, it was easy. Adding in an object-based inheritance system should be just as easy, I just need an interface. $obj->parents(@other_objs) is a little clunky.
>
> ...Look at Class::Object! Its really, really thin. Benchmark it, its no slower than regular objects. http://www.pobox.com/~schwern/src/Class-Object-0.01.tar.gz

[The Golden Troll Award goes to Dan Brien for this gem.]

The Internal String API

Dan Sugalski initiated discussion on the internal API for strings:

> Since we're going to try and take a shot at being encoding-neutral in the core, we're going to need some form of string API so the core can actually manipulate string data. I'm thinking we'll need to be able to at least do this with string:
>
> - Convert from and to UTF-32
> - lengths in bytes, characters, and possibly glyphs
> - character size (with the variable length ones reporting in negative numbers)
> - get and set the locale (This might not be the spot for this)
> - normalize (a noop for non-Unicode data)
> - Get the encoding name
> - Do a substr operation by character and glyph

David Nicol suggested implementing strings as a tree, vice a contiguous memory block. After some pondering, this seemed to grow on Dan, and he is awaiting a yea-or-nay from Larry. Copy-On-Write for Strings will also be implemented, although there was no mention of a potential key signature.

Miscellany

Simon Cozens released an updated version of his Perl 6 emulator.

Marcel Grunauer announced a Proof-of-Concepts page for Perl 6, which contains info and links to Perl 5 modules that may provide a glimpse of things to come.

There were more complaints about operator choices. (Specifically, ~ for string concatenation, and . (the dot) for dereference (vice ->).)

Last Words

There's but three weeks till TPC 5.0 kicks off in San Diego.

Bryan C. Warnock

Downloads
Documentation
CPAN
FAQs
Training
Resources
Article Archive
Books
Search
Register/Log in

Columns
P5P Digest
P6P Digest
Off the Wall

Why Not Translate Perl to C?

Print This Article

by Mark-Jason Dominus
June 27, 2001

People often have the idea that automatically translating Perl to C and then compiling the C will make their Perl programs run faster, because "C is much faster than Perl." This article explains why this strategy is unlikely to work.

Short Summary

Your Perl program is being run by the Perl interpreter. You want a C program that does the same thing that your Perl program does. A C program to do what your Perl program does would have to do most of the same things that the Perl interpreter does when it runs your Perl program. There is no reason to think that the C program could do those things faster than the Perl interpreter does them, because the Perl interpreter itself is written in very fast C.

Some detailed case studies follow.

Built-In Functions

Suppose your program needs to split a line into fields, and uses the Perl `split` function to do so. You want to compile this to C so it will be faster.

This is obviously not going to work, because the `split` function is already implemented in C. If you have the Perl source code, you can see the implementation of `split` in the file `pp.c`; it is in the function named `pp_split`. When your Perl program uses `split`, Perl calls this `pp_split` function to do the splitting. `pp_split` is written in C, and it has already been compiled to native machine code.

Now, suppose you want to translate your Perl program to C. How will you translate your `split` call? The only thing you can do is translate it to a call to the C `pp_split` function, or some other equivalent function that splits. There is no reason to believe that any C implementation of `split` will be faster than the `pp_split` that Perl already has. Years of work have gone into making `pp_split` as fast as possible.

You can make the same argument for all of Perl's other built-in functions, such as `join`, `printf`, `rand` and `readdir`.

So much for built-in functions.

Data Structures

Why is Perl slow to begin with? One major reason is that its data structures are extremely flexible, and this flexibility imposes a speed penalty.

Let's look in detail at an important example: strings. Consider this Perl code:

```
$x = 'foo';
$y = 'bar';
$x .= $y;
```

That is, we want to append `$y` to the end of `$x`. In C, this is extremely tricky. In C, you would start by doing something like this:

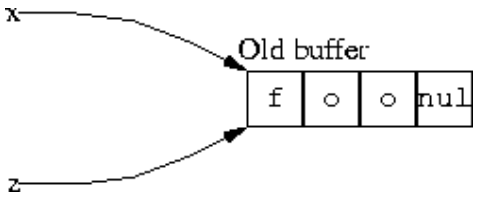
```
char *x = "foo";
char *y = "bar";
```

Now you have a problem. You would like to insert `bar` at the end of the buffer pointed to by `x`. But you can't, because there is not enough room; `x` only points to enough space for four characters, and you need space for seven. (C strings always have an extra `nul` character on the end.) To append `y` to `x`, you must allocate a new buffer, and then arrange for `x` to point to the new buffer:

```
char *tmp = malloc(strlen(x) + strlen(y) + 1);
strcpy(tmp, x);
strcat(tmp, y);
x = tmp;
```

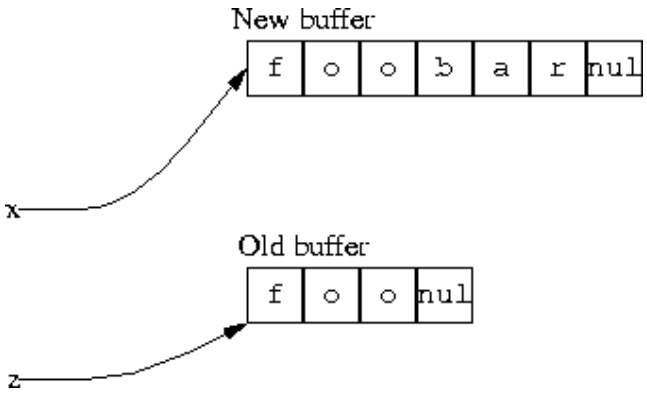
This works fine if `x` is the only pointer to that particular buffer. But if some other part of the program also had a pointer to the buffer, this code does not work. Why not? Here's the picture of what we did:

BEFORE:



Here `x` and `z` are two variables that both contain pointers to the same buffer. We want to append `bar` to the end of the string. But the C code we used above doesn't quite work, because we allocated a new region of memory to hold the result, and then pointed `x` to it:

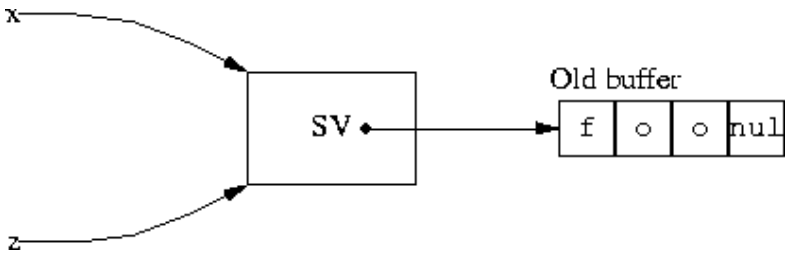
AFTER `x = tmp`:



It's tempting to think that we should just point `z` to the new buffer also, but in practice this is impossible. The function that is doing the appending cannot know whether there is such a `z`, or where it may be. There might be 100 variables like `z` all pointing to the old buffer, and there is no good way to keep track of them so that they can all be changed when the array moves.

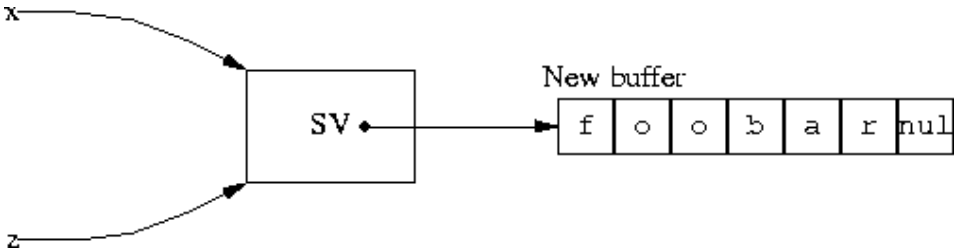
Perl does support a transparent string append operation. Let's see how this works. In Perl, a variable like `$x` does not point directly at the buffer. Instead, it points at a structure called an SV. ('Scalar Value') The SV has the pointer to the buffer, and also some other things that I do not show:

BEFORE `$x .= $y`



When you ask Perl to append `bar` to `$x`, it follows the pointers and finds that there is not enough space in the buffer. So, just as in C, it allocates a new buffer and stores the result in the new buffer. Then it fixes the pointer in the SV to point to the new buffer, and it

throws away the old buffer:



Now `$x` and `$z` have both changed. If there were any other variables sharing the SV, their values would have changed also. This technique is called "double indirection,'" and it is how Perl can support operations like `.=`. A similar principle applies for arrays; this is how Perl can support the `push` function.

The flexibility comes at a price: Whenever you want to use the value of `$x`, Perl must follow two pointers to get the value: The first to find the SV structure, and the second to get to the buffer with the character data. This means that using a string in Perl takes at least twice as long as in C. In C, you follow just one pointer.

If you want to compile Perl to C, you have a big problem. You would like to support operations like `.=` and `push`, but C does not support these very well. There are only three solutions:

1. Don't support `.=`

   This is a bad solution, because after you disallow all the Perl operations like `.=` and `push` what you have left is not very much like Perl; it is much more like C, and then you might as well just write the program in C in the first place.

2. Do something extremely clever

   Cleverness is in short supply this month. `:)`

3. Use a double-indirection technique in the compiled C code

   This works, but the resulting C code will be slow, because you will have to traverse twice as many pointers each time you want to look up the value of a variable. But that is why Perl is slow! Perl is already doing the double-indirection lookup in C, and the code to do this has already been compiled to native machine code.

So again, it's not clear that you are going to get any benefit from translating Perl to C. The slowness of Perl comes from the flexibility of the data structures. The code to manipulate these structures is already written in C. If you translate a Perl program to C, you have the choice of throwing away the flexibility of the data structure, in which case you are now writing C programs with C structures, or keeping the flexibility with the same speed penalty. You probably cannot speed up the data structures, because if anyone knew how to make the structures faster and still keep them flexible, they would already have made those changes in the C code for Perl itself.

Possible Future Work

It should now be clear that although it might not be hard to translate Perl to C, programs probably will not be faster as a result.

However, it's possible that a sufficiently clever person could make a Perl-to-C translator that produced faster C code. The programmer would need to give hints to the translator to say how the variables were being used. For example, suppose you have an array `@a`. With such an array, Perl is ready for anything. You might do `$a[1000000] = 'hello';` or `$a[500] .= 'foo';` or `$a[500] /= 17;`. This flexibility is expensive. But suppose you know that this array will only hold integers and there will never be more than 1,000 integers. You might tell the translator that, and then instead of producing C code to manage a slow Perl array, the translator can produce

Related Articles

Larry Wall Apocalypse 2

Damian Conway Exegesis 2

`perl6-internals` mailing list archive

```
int a[1000];
```

and use a fast C array of machine integers.

To do this, you have to be very clever and you have to think of a way of explaining to the translator that `@a` will never be bigger than 1,000 elements and will only contain integers, or a way for the translator to guess that just from looking at the Perl program.

People are planning these features for Perl 6 right now. For example, Larry Wall, the author of Perl, plans that you will be able to declare a Perl array as

```
my int @a is dim(1000);
```

Then a Perl-to-C translator (or Perl itself) might be able to use a fast C array of machine integers rather than a slow Perl array of SVs. If you are interested, you may want to join the perl6-internals mailing list.

This Week on p5p 2001/07/02

Print This Article

by Simon Cozens
July 02, 2001

Notes

Please send corrections and additions to perl-thisweek-YYYYMM@simon-cozens.org where YYYYMM is the current year and month. Changes and additions to the perl5-porters biographies are particularly welcome.

This was a reasonably normal week, seeing the usual 500 or so messages.

Many thanks again to Leon for last week's summary.

Module Versioning and Testing

There's a move on to make the modules under `ext/` free-standing: that is, to be able to say

cd ext/File/Glob

make dist

and get a bundle that can be uploaded to CPAN or otherwise distributed. The only problem with this is tests. Currently the tests are kept under `t/` of the main root of the Perl source tree, and are run when a `make test` is done there. To make the modules freestanding, you'd have to move the tests to `ext/Foo/Bar/t/` and have the main `make test` also traverse the extension subdirectories and run the tests there. But then, of course, there's another problem. Can you spot it?

`make test` is run from an uninstalled Perl, which needs explicit hints about where to find the Perl library. Hence, the tests in `t/` directly wibble `@INC`. This wouldn't work if we're making the modules freestanding, or if we move the tests to `ext/Foo/Bar/t/`. So the trick, which nobody's done yet, is to move the tests to the right place, change the main `make test` to recurse extension subdirectories, but also to propagate an environment variable telling the module where to find the library. (`PERL5LIB` is what you want for this.) That would be a nice little task for someone...

Speaking of testing, Schwern got `Test::Simple` and `Test::More` added to the core, bringing the first All Your Base reference into the Perl source tree. Oh, and **how** many different testing mechanisms?

Robin Houston incremented `B::Deparse`'s version number and added some change notes. Here's what's improved since 5.6.1:

```
Changes between 0.60 and 0.61 (mostly by Robin Houston)
- many bug-fixes
- support for pragmas and 'use'
- support for the little-used $[ variable
- support for __DATA__ sections
- UTF8 support
- BEGIN, CHECK, INIT and END blocks
- scoping of subroutine declarations fixed
- compile-time output from the input program can be suppressed, so that the
  output is just the deparsed code. (a change to O.pm in fact)
- our() declarations
- *all* the known bugs are now listed in the BUGS section
- comprehensive test mechanism (TEST -deparse)
```

The new test mechanism is great: it runs the standard Perl test suite through `B::Deparse` and then back through Perl again to ensure the deparsed code still passes the tests. And, as a testament to the work that's been done on `B::Deparse`, they mostly do pass.

Schwern also bumped up `ExtUtils::Manifest`, which caused Jarkko to appeal for a script which checks two Perl source trees to find updated versions of modules without a version change. Larry Shatzer provided one, and Jarkko used it to update the versions in the current tree. Schwern also put `Cwd` on CPAN, and found a weird dynamic loading bug with the XS version of `Cwd`. Oh, and noted that after his benchmarking, there's no significant performance loss between 5.6.1 without PerlIO and bleadperl with it.

Carpentry

Mike Guy partially fixed a problem whereby when a magic variable like `$1` is passed as a subroutine parameter, `carp` and the debugger don't see it properly. Tony Bowden took the opportunity to ask for either more or less documentation for `longmess` and `shortmess` depending on whether or not they were meant to be internal to `Carp`. Tony wrote a documentation patch himself.

Schwern asked for a better name for those functions, perhaps more in line with the carp/croak/cluck theme. Jarkko suggested `yodel` and `yelp` which Schwern implemented.

Andreas had a quick bitch about Schwern's, uh, idiosyncratic naming style:

> Your style of naming things is just plain sick:
>
> AnyLoader does anything but interface to any loader. Ima::DBI has something to do with DBI, but nothing with Ima. D::oh is the only really funny one. But it's from 1999 and getting old. Sex: I still don't know if it is fun or what. Bone::Easy? Same here. Semi::Semicolon? Same here.
>
> And now yodle!

Unfair, though, since `yodel` and `yelp` weren't his...

However, Hugo objected to `yodel`/ `yelp` because the other verbs write to standard error, hence "speaking" whereas `longmess` and `shortmess` don't actually "say" anything. Jarkko agreed, and there the matter rested. (Modulo Rich Lafferty's suggestion of "flame" for objections in a written medium...)

Regex Capture-to-variable

Jeffrey Friedl (he of the Regexp book) came up with an interesting patch which adds the new special variable `$^N` for the most-recently assigned bracket match. This is different from `$+` which is the highest numbered match; that's to say, given

```
/(foo(bar))/
```

then `$+` is equivalent to `$2`, whereas `$^N` is equivalent to the bracket match for the last closing bracket; that is, `$1`. This essentially allows you to do capture-to-variable, like this:

```
(?:(\d+)(?{ $phone_number = $^N }))
```

without having to worry about which number bracket the match was. (Especially useful if you have to change your regexp around.) Whether this makes regular expressions cleaner or dirtier, I'll leave up to you... However, Jeffrey also noted that you can use regexp overloading (my, that's an obscure feature - look at `re.pm`) to make such syntax as

```
(?<$phone_number>\d+)
```

work. Now that's cool.

Phillip Newton added a nmenomic: `$^N` is the most recently closed Nested parenthesis.

Perl on S390

Hal Morris got Perl going on Linux/390, with only one test failing. Good news for the new generation of mainframe hackers.

There's mixed news for the old-timers, though; Peter Prymmer has got it down to 10 test failures, but one of the tests completely hangs Perl. Apparently `study` under OS/390 is best avoided. He also started some investigation of a `Bigint` bug, under the direction of Tels and John Peacock, but left for his holidays and the discussion moved to the `perl-mvs` mailing list.

UTS Amdahl

Oh, and talking of weird platforms, UTS. Hal (who's actually from UTS Global, so much kudos there) has been testing out recent Perl builds on UTS, and turning up some ... take a deep breath ... icky numeric conversion issues.

Nicholas Clark was convinced they (well, some of them at least) were due to UTS C going through a `foo--` statement twice, but Hal pointed out he didn't expect UTS C to be quite that braindead. On the other hand, Nick's analysis looked convincing...

Hal also fixed up `hints/uts.sh` so that UTS now configures and builds nicely at least.

Various

David Wheeler found a known but really, really weird bug with lexical arrays; if you do:

```
my @a = foo(); my @b = foo();
sub foo { $x = 0; my @ary; push @ary, "hi" if $x; push @ary, "ho"; return @ary }
```

`@a` gets "ho" as you'd expect, but `@b` gets "ho","ho". Ronald Kimball told him Not To Do That, Then.

Peter Prymmer noted that Perl on VMS was bailing out during the test suite, leading to lots of bogus failures. This only happens if none of the DBM libraries (GDBM, DB_File, NDBM or SDBM) were built. As the first three require external libraries that VMS doesn't have and the last one is currently broken, it's no wonder Perl is bailing out. The fix is to work out why SDBM has stopped building on VMS. Peter also produced a lot of other VMS and HPUX reports.

Andy was pleasantly surprised to note that the promised "binary compatibility with 5.005" actually works even in bleadperl. Perhaps we need to break more things.

John Peacock asked a portability question for XS bit-twiddling; he's trying to adapt a math library which depends on casting two numbers to a long and adding them together to avoid overflow. Jarkko's fantastic architecture experience was brought to bear as he revealed that Cray Unicos has `long == short == int == double`. Oh, and type casting has issues too, so you have to use a union. Nicholas Clark suggested the old trick of comparing the operands of an addition with the result; if the result is smaller than either of the operands, you've overflowed, so you add a carry and off you go.

The news that GCC 3.0 was out brought a rush of people testing Perl out with it; I got it through on Linux with all tests successful, as did H. Merijn Brand on HPUX, but with rather a lot more warnings. This was because HPUX messed up the test for `__attribute__` due to using a HP linker instead of a GNU one. Merijn and Jarkko got this fixed up.

Artur continued his iThreads quest; he renamed the "shared" attribute to "static", (and then again to "unique" after objections) presumably to free it up for an attribute which actually does share variables between interpreters, and also added a function which cloned the Perl host. He said that `threads-0.01`, the new threading module, will be released to CPAN when 5.7.2 hits the road. (Which Jarkko keeps hinting will be very, very, very soon now.) He also complained bitterly when Marcel Grunauer tried to document `attributes.pm` as useful, despite the fact that Marcel has some really very cool modules on CPAN based on it...

Marcel also found, and Radu Greab fixed, an insiduous bug in `split`, whereby if the default whitespace pattern was used for one iteration of a loop, it would be used for all succeeding ones; the `PMf_WHITE` flag for the regular expression was being set but never unset. Urgh.

Ilya produced some rough changes documentation for OS/2, as well as some other little patches. Norton Allen provided some QNX updates.

Phillip Newton documented the neat

```
$count = () = function()
```

idiom for counting the number of return values from an operation. That was something I hadn't seen before; you learn something new every day... Until next week I remain, your humble and obedient servant,

---

Simon Cozens

Downloads
Documentation
CPAN
FAQs
Training
Resources
Article Archive
Books
Search
Register/Log in

Columns
P5P Digest
P6P Digest
Off the Wall

Yet Another YAPC Report: Montreal

Print This Article

by Schuyler Erle
June 21, 2001

A year ago, at Yet Another Perl Conference North America 19100, both Perl-the-language and Perl-the-community seemed to be headed for trouble. Longtime Perl hackers spoke openly with concern at the apparent stagnation of Perl 5 development, and how the community seemed to be increasingly bogged down by acrimony and bickering. Now, a year later, the tide has already turned and the evidence is nowhere more apparent than at this year's YAPC::NA in Montreal.

The conference, produced by Yet Another Society, was a smashing success. More than 350 Perl Mongers converged from all across North America and Europe on McGill University for the three-day event. Rich Lafferty and Luc St. Louis, key organizers from the Montreal.pm group, did a brilliant job of lining everything up; and Kevin Lenzo, YAS president, once again did the crucial fund raising and promotional work to make the conference a reality.

Certain familiar faces were missing from this year's conference, including Larry himself, who was scheduled to deliver the keynote but was absent due to illness. (Get well soon, Larry!) The unenviable task of filling Larry Wall's shoes for the highly anticipated opening talk fell to the infamous Dr. Damian Conway, indentured servant to the Perl community and lecturer extraordinaire. Those of you who have seen Damian in action will have probably guessed that his presentation did not disappoint. The topic was, of course, a tour of the past and future of the Perl language -- where we have come (a long way from Perl 1 in 1987) and where are going (a long way yet to Perl 6).

To hear Damian tell it, Perl 6 looks like it's going to be awesome. While many details are still sketchy, the intention from all quarters is to preserve all the things we like about Perl today, especially its tendency to be eclectic in its incorporation of ideas and features from other languages. Larry, Damian and others have carefully studied the lessons of such languages as Java, Python, Ruby, and even the infant C#, in the hopes of applying those lessons to Perl 6.

Damian's keynote also focused on how Perl 6 will attempt to correct some of the flaws and deficiencies of Perl 5, the details of which can be found elsewhere, so I won't reiterate them here. Additionally, he emphasized that, due to the unexpected quantity and scope of the Perl 6 RFCs, the final language design will take Larry far longer than anyone originally imagined. Damian went on to predict a usable alpha version of Perl 6 being ready by May 2002, with a full release perhaps available by October 2002. However, as pieces of the Perl 6 design stabilize, Damian and others (including our own Simon Cozens) will be implementing them in Perl 5, so that we can start playing with Perl 6 today, rather than next year.

Meanwhile, the continued enthusiasm and energy being devoted to Perl 6 has had a profound impact on the community at large that is hard to overstate. YAPC::NA 2001 was marked not merely by much discussion and speculation on Perl 6, but also by fascinating new developments in Perl. One of the downright niftiest of these new directions is Brian Ingerson's Inline.pm, which he presented in a 90-minute talk Wednesday. Inline.pm uses a form of plug-in architecture to allow seamless embedding of other languages like C, C++, Java, and, yes, Python, right into ordinary Perl scripts. Brian, who works at ActiveState, has already written a www.perl.com feature on Inline.pm, so I'll merely mention here that the module hides away the

frighteningly ugly details of gluing disparate languages together, in the most intuitive way possible. This kind of development is really exciting for the ways in which it opens new doors and breeds new ideas on the many, many different kinds of intriguing things that can still be done with Perl 5. Incidentally, Brian's midlecture sing-a-longs about Perl internals and so forth were also quite well regarded.

The hubbub around Inline.pm was just one thread in the theme of "Perl as glue language for the 21st Century," a theme visited and revisited at many times and places throughout the conference. The notion was raised again by Perl 6 project manager Nathan Torkington in his presentation at Adam "Ziggy" Turoff's Perl Apprenticeship Workshop on Wednesday. Amidst the announcement of many interesting and valuable projects in need of Perl hackers, Gnat issued a call to "make a Python friend" and collaborate with them on a development project. "Show them we're not *all* evil!" he insisted, in marked contrast to his howlingly funny diatribe on Python at the previous year's Lightning Talks.

"I was surprised that Gnat took that approach, because I thought I would be left this year to argue the other side," ActiveState's Neil Kandalgaonkar observed, after giving his Friday morning talk on "Programming Parrot," so named for its case study in getting Perl and Python applications to work in concert. Part of Neil's tale of success lay in using Web services to get different processes running in different languages on different machines to exchange data reliably. "All it took was an extra four lines of scripting in each language, and I was done," he noted, driving home the importance of using and extending Perl's ability to talk to other languages and applications.

Meanwhile, YAPC North America 2001 also showed growth in the depth and scope of the conference's offerings. In contrast to previous years, where talks were largely aimed at beginner and intermediate Perl hackers, this year's presentations covered some more advanced topics, such as Nat Torkington's three-hour lesson on the Perl internals that he delivered to a packed house Thursday. Originally written by Simon Cozens (who was unable to attend), the Perl internals class presented a concise introduction to some of Perl's inner workings, furthering the Perl community's expressed goal of lowering the barrier of entry to internals hacking and encouraging wider participation in Perl core development. Later in the day, Michael Schwern addressed the ever-present tendency of Perl hackers to rely on Perl's forgiving nature in his rather well-attended talk on "Disciplined Programming, or, How to Be Lazy without Trying." "Always code Perl as if you were writing for CPAN," Schwern urged his audience. "Document and test as you go, and release working versions often."

Speaking of which, the Comprehensive Perl Archive Network was also a major topic of discussion at YAPC. "The CPAN is Perl's killer app," Gnat said at one point. "No other language has anything like it." Neil and Brian gave a short presentation on their experiences building and maintaining ActiveState's PPM repository, a collection of binary distributions of CPAN modules. The dynamic duo from Vancouver yielded some of their time to Schwern to allow him to discuss his proposed CPANTS project, intended to automate testing and quality verification of modules in the repository. Metadata, rating systems, trust metrics and peer-to-peer distribution models were all touched on. Based on the buzz this year, it seems reasonable to predict that many new and exciting things are likely to grow up around the CPAN, and around the possibilities inherent in the distribution of Perl modules, in the not-too-distant future.

The final talk Thursday was once more delivered by Damian Conway, and curiosity had spread far and wide on how he might top last year's now-legendary presentation on Quantum::Superpositions. This year's Thursday afternoon plenary lecture was merely titled, "Life, the Universe, and Everything," in homage to the late Mr. Adams; and, true to his word, Damian delivered just that. Swooping from Conway's Game of Life (no relation), to a source filter for programming Perl in Klingon (a la Perligata), to the paradox of Maxwell's Demon (conveniently dispelled with a little help from Quantum::Superpositions), Damian's talk was a masterful reflection of all the things we love about Perl: It was clever, complex, elegant, and, most of all, it was fun.

(Parenthetically, among the modules that Damian introduced at this talk was a little number called Sub::Junctive. As a linguist, I must confess it scares the living heck outta me. Look for it on the CPAN.)

Friday featured more of this year's theme of Perl as glue-language-for-the-21st-Century in two talks on Web services by Adam "Ziggy"

Turoff and Nat Torkington, in which Nat issued an impassioned plea for a Perl implementation of Freenet. However, the morning's highlight was without a doubt the much-anticipated Lightning Talks. Hosted once again by the irrepressible Mark-Jason Dominus, the 90-minute series of five-minute short talks went over quite well, featuring topics ranging from how hacking Perl is like Japanese food and the graphing of IRC conversations to a call for more political action from within hackerdom and an overview of the Everything Engine. The showstopper, however, was once again Damian, who is generally reputed to be unable to hold forth on any topic for anything *less* than an hour and a half. To everyone's surprise, the Lightning Talk consisted of a hilarious argument in the grand Shakespearean style between Damian and Brian Ingerson, over the disputed authorship of Inline::Files, a nifty new module for extending the capabilities of the old DATA filehandle. Their invective-laden dialogue was the most brilliantly humorous five minutes of the entire conference, and, yes, Damian even managed to finished on time. :) If you had the misfortune not to be present, you might be lucky enough to see them have at each other again at this year's Perl Conference 5 in San Diego.

Later in the day, Nat and Damian chaired a Perl 6 status meeting, reviewing the major events in Perl 6 starting with the announcement at TPC4, and working forward to the present language design phase. "This is a fresh rebirth for Perl AND for the community," Gnat said at one point. "Everything changes." The sometimes fractious attitudes encountered on the various Perl mailing lists were discussed. "In some ways this is a meritocracy," Gnat confessed. "Write good patches and we will love you." Kirrily "Skud" Robert then spoke at length on the future of the core Perl modules, and on the need to develop guidelines to direct the process of porting them to Perl 6.

Finally, the plenary session Friday afternoon closed the conference with another presentation from, you guessed it, Damian Conway. Our Mr. Conway took the opportunity to thank Yet Another Society and its sponsors for all of the contributions that permitted him to take a year off from academia to work exclusively on Perl. He then reviewed some of the fruits of that labor to date, including NEXT.pm, Inline::Files and the brilliant Filter::Simple, all of which, it should be pointed out, are now freely available to the community.

It has been nearly a year since Jon Orwant's now-legendary coffee-mug-tossing tantrum at TPC4 touched off the decision to begin work on Perl 6. After the grueling RFC process, the endless mailing list discussions and the breathless wait to see what Larry would come through with, Perl 6 the language and Perl 6 the community finally appear to be taking shape right before our eyes. New innovations are coming along more and more often, including Larry's Apocalypses, Brian's Inline modules, all of the potential emerging from the Web services meme, the future of the CPAN, new projects like Reefknot -- including continuing projects such as POE and Mason -- and, last but not least, whatever the heck it is that Damian is working on this week.

The Yet Another Perl Conferences are evolving, as well. Although neither Larry nor Randal nor Orwant could make it this year, the turnout was nevertheless such that, no matter where you looked at the conference, there you might find someone you knew from IRC, from the mailing lists, from previous conferences or for the great work that person had done for Perl. Although I've only touched on some highlights, there were dozens of presenters at this year's conference, practically all of them had something fascinating to say, and I really wish I had more time and space to cover them all.

Finally, it's safe to say that YAPC::NA clearly defined its own existence as a growing concern of the community this year, having at last separated from its birthplace at Carnegie Mellon in Pittsburgh. Montreal, as it turns out, is a fantastic, vibrant place to hold a summer conference, with countless magnificent restaurants and bars suitable for hosting the heady after-hours carousings of the Perl community. From every report, a good time was had by nearly all, and I think we all eagerly await the next YAPC::America::North, wherever it may be held.

Parse::RecDescent Tutorial

by Jeffrey Goff
June 13, 2001

The Basics

Parse::RecDescent is a combination compiler and interpreter. The language it uses can be thought of roughly as a macro language like CPP's, but the macros take no parameters. This may seem limiting, but the technique is very powerful nonetheless. Our macro language looks like this:

```
macro_name : macro_body
```

A colon separates the macro's name and body, and the body can have any combination of explicit strings ("string, with optional spaces"), a regular expression (`/typical (?=perl) expression/`), or another macro that's defined somewhere in the source file. It can also have alternations. So, a sample source file could look like:

```
startrule : day  month /\d+/ # Match strings of the form "Sat Jun 15"


day : "Sat" | "Sun" | "Mon" | "Tue" | "Wed" | "Thu" | "Fri"


month : "Jan" | "Feb" | "Mar" | "Apr" | "May" | "Jun" |
        "Jul" | "Aug" | "Sep" | "Oct" | "Nov" | "Dec"
```

Three macros make up this source file: `startrule`, `dayrule` and `monthrule`. The compiler will turn these rules into its internal representation and pass it along to the interpreter. The interpreter then takes a data file and attempts to expand the macros in `startrule` to match the contents of the data file.

The interpreter takes a string like "Sat Jun 15" and attempts to expand the `startrule` macro to match it. If it matches, the interpreter returns a true value. Otherwise, it returns `undef`;. Some sample source may be welcome at this point:

```
#!/usr/bin/perl
```

```
use Parse::RecDescent;


# Create and compile the source file
$parser = Parse::RecDescent->new(q(
  startrule : day  month /\d+/


  day : "Sat" | "Sun" | "Mon" | "Tue" | "Wed" | "Thu" | "Fri"


  month : "Jan" | "Feb" | "Mar" | "Apr" | "May" | "Jun" |
          "Jul" | "Aug" | "Sep" | "Oct" | "Nov" | "Dec"
));


# Test it on sample data
print "Valid date\n" if $parser->startrule("Thu Mar 31");
print "Invalid date\n" unless $parser->startrule("Jun 31 2000");
```

Creating a new Parse::RecDescent instance is done just like any other OO module. The only parameter is a string containing the source file, or *grammar*. Once the compiler has done its work, the interpreter can run as many times as necessary. The sample source tests the interpreter on valid and invalid data.

By the way, just because the parser knows that the string "Sat Jun 15" is valid, it has no way of knowing if the 15th of June was indeed a Saturday. In fact, the sample grammar would also match "Sat Feb 135". The grammar describes form, not content.

Getting Data

Now, this is quite a bit of work to go to simply to match a string. However, much, much more can be done. One element missing from this picture is capturing data. So far the sample grammar can tell if a string matches a regular expression, but it can't tell us what the data it's parsed is. Well, these macros can be told to run perl code when encountered.

Perl code goes after the end of a rule, enclosed in braces. When the interpreter recognizes a macro such as `startrule`, the text matched is saved and passed to the perl code embedded in the grammar.

Each word or *term* of the macro ('day', 'month'...) is saved by the interpreter. `dayrule` gets saved into the `$item{day}` hash entry, as does `monthrule`. The `/\d+/` term doesn't have a corresponding name, so its data comes from the `@item` array. `$item[0]` is always the rule name, so `/\d+/` gets saved into `$item[3]`. So, code to print the parsed output from our sample `startrule` rule looks like this:

```
startrule : day month /\d+/
          { print "Day: $item{day} Month: $item{month} Date: $item[3]\n"; }
```

Everything in the parser is run as if it was in the Parse::RecDescent package, so when calling subroutines outside Parse::RecDescent, either qualify them as `Package::Name->my_sub()` or subclass Parse::RecDescent.

A Mini-Language

All of the pieces are now in place to create a miniature language, compile, and run code in it. To make matters simple, the language will only have two types of instruction: Assign and Print. A sample 'Assign' instruction could look like `foo = 3 + a`. The 'Print' statement will look like `print foo / 2`. Add the fact that `3 + a` can be arbitrarily long (`temp = 3+a/2*4`), and now you've got a non-trivial parsing problem.

The easiest instruction to implement is the 'Print' instruction. Assuming for the moment that the right-hand side of the statement (the `foo / 2` part of `print foo / 2`) already has a rule associated with it (called 'expression'), the 'Print' instruction is very simple:

```
print_instruction : /print/i expression
                  { print $item{expression}."\n" }
```

The 'Assign' instruction is a little harder to do, because we need to implement variables. We'll do this in a straightforward fashion, storing variable names in a hash. This will live in the main package, and for the sake of exposition we'll call it `%VARIABLE`. One caveat to remember is that the perl code runs inside the Parse::RecDescent package, so we'll explicitly specify the `main` package when writing the code.

More complex than the 'Print' instruction, the 'Assign' instruction has three parts: the variable to assign to, an "=" sign, and the expression that gets assigned to the variable. So, the instruction looks roughly like this:

```
assign_instruction : VARIABLE "=" expression
                   { $main::VARIABLE{$item{VARIABLE}} = $item{expression} }
```

Much like we did with the `dayrule` rule in the last section, we'll combine the `print_instruction` and `assign_instruction` into one `instruction` rule. The syntax for this should be fairly simple to remember, as it's the same as a Perl regular expression.

```
instruction : print_instruction
            | assign_instruction
```

In order to make the `startrule` expand to the `instruction` rule, we'd ordinarily use a rule like `startrule : instruction`. However, most languages let you enter more than one instruction in a source file. One way to do this would be to create a recursive rule that would look like this:

```
instructions : instruction ";" instructions
             | instruction
```

```
startrule : instructions
```

[[JMG: I'm sorely tempted to rewrite this chunk, if only 'cause there's a lot of info here in just one paragraph]]

Input text like "print 32" expands as follows: `startrule` expands to `instructions`. `instructions` expands to `instruction`, which expands to `print_instruction`. Longer input text like "a = 5; b = a + 5; print a" expands like so: `startrule` expands to `instructions`. The interpreter looks ahead and chooses the alternative with the semicolon, and parses "a = 5" into its first instruction. "b = a + 5; print a" is left in `instructions`. This process gets repeated twice until each bit has been parsed into a separate `instruction`.

If the above seemed complex, Parse::RecDescent has a shortcut available. The above `instructions` rule can be collapsed into `startrule : instruction(s)`. The `(s)` part can simply be interpreted as "One or more `instruction`s". By itself this assumes only whitespace exists between the different instructionrule;s, but here again, Parse::RecDescent comes to the rescue, by allowing the user to specify a separator regular expression, like `(s /;/)`. So, the `startrule` actually will use the `(s /;/)` syntax.

```
startrule : instruction(s /;/)
```

The Expression Rule

Expressions can be anything from '0' all the way through 'a+bar*foo/300-75'. Ths range may seem intimidating, but we'll try to break it down into easy-to-digest pieces. Starting simply, an expression can be as simple as a single variable or integer. This would look like:

```
expression : INTEGER
           | VARIABLE
           { return $main::VARIABLE{$item{VARIABLE}} }
```

The `VARIABLE` rule has one minor quirk. In order to compute the value of the expression, variables have to be given a value. In order to modify the text parsed, simply have the code return the modified text. In this case, the perl code looks up the variable in `%main::VARIABLE` and returns the value of the variable rather than the text.

Those two lines take care of the case of an expression with a single term. Multiple-term expressions (such as `7+5` and `foo+bar/2`) are a little harder to deal with. The rules for a single expression like `a+7` would look roughly like:

```
expression : INTEGER OP INTEGER
           | VARIABLE OP INTEGER
           | INTEGER OP VARIABLE
           | VARIABLE OP VARIABLE
OP : /[-+*/%]/
```

This introduces one new term, `OP`. This rule simply contains the binary operators `/[-+*/%]/`. The above approach works for two terms, and can be extended to three terms or more, but is terribly unwieldy. If you'll remember, the `expression` rule already is defined as `INTEGER | VARIABLE`, so we can replace the right-hand term with `expression`. Replacing the right-hand term with `expression` and getting rid of redundant lines results in this:

```
expression : INTEGER OP expression
           | VARIABLE OP expression
```

We'll hand off the final evaluation to a function outside the Parse::RecDescent package. This function will simply take the `@item` list from the interpreter and evaluate the expression. Since the array will look like `(3,'+',5)`. we can't simply say `$item[1] $item[2] $item[3]`, since `$item[2]` is a scalar variable, not an operator. Instead we'll take the string `"$item[1] $item[2] $item[3]"` and evaluate that. This will evaluate the string and return the result. This then gets passed back, and becomes the value of the `expression`.

```
expression : INTEGER OP expression
           { return main::expression(@item) }
           | VARIABLE OP expression
           { return main::expression(@item) }


sub expression {
  shift;
  my ($lhs,$op,$rhs) = @_;
  return eval "$lhs $op $rhs";
}
```

That completes our grammar. Testing is fairly simple. Write some code in the new language, like "a = 3 + 5; b = a + 2; print a; print b", and pass it to the `$parser->startrule()` method to interpret the string.

The file included with this article comes with several test samples. The grammar in the tutorial is very simple, so plenty of room to experiment remains. One simple modification is to change the `INTEGER` rule to account for floating point numbers. Unary operators (single-term such as `sin()`) can be added to the `expression` rule, and statements other than 'print' and 'assign' can be added easily.

Other modifications might include adding strings (some experimental extensions such as '<perl_quotelike>' may help). Changing the grammar to include parentheses and proper precedence are other possible projects.

Closing

Parse::RecDescent is a powerful but difficult-to-undertstand module. Most of this is because parsing a language can be difficult to understand. However, as long as the language has a fairly consistent grammar (or one can be written), it's generally possible to translate it into a grammar that Parse::RecDescent can handle.

Many languages have their grammars available on the Internet. Grammars can usually be found in search engines under the keyword 'BNF', standing for 'Backus-Naur Form'. These grammars aren't quite in the form Parse::RecDescent prefers, but can usually be modified to suit.

When writing your own grammars for Parse::RecDescent, one important rule to keep in mind is that a rule can never have itself as the first term. This makes rules such as `statement : statement ";" statements` illegal. This sort of grammar is called "left-recursive" because a rule in the grammar expands to its left side.

Left-recursive grammars can usually be rewritten to right-recursive, which will parse cleanly under Parse::RecDescent, but there are classes of grammars thatcant be rewritten to be right-recursive. If a grammar can't be done in Parse::RecDescent, then something like `Parse::Yapp` may be more appropriate. It's also possible to coerce `yacc` into generating a perl skeleton, supposedly.

Hopefully some of the shroud of mystery over Parse::RecDescent has been lifted, and more people will use this incredibly powerful module.

```
#!/usr/bin/perl -w


use strict;
use Parse::RecDescent;
use Data::Dumper;


use vars qw(%VARIABLE);


# Enable warnings within the Parse::RecDescent module.


$::RD_ERRORS = 1; # Make sure the parser dies when it encounters an error
$::RD_WARN   = 1; # Enable warnings. This will warn on unused rules &c.
$::RD_HINT   = 1; # Give out hints to help fix problems.


my $grammar = <<'_EOGRAMMAR_';


  # Terminals (macros that can't expand further)
  #


  OP       : m([-+*/%])      # Mathematical operators
  INTEGER  : /[-+]?\d+/      # Signed integers
  VARIABLE : /\w[a-z0-9_]*/i # Variable


  expression : INTEGER OP expression
             { return main::expression(@item) }
             | VARIABLE OP expression
```

```
            { return main::expression(@item) }
            | INTEGER
            | VARIABLE
            { return $main::VARIABLE{$item{VARIABLE}} }


  print_instruction  : /print/i expression
                     { print $item{expression}."\n" }
  assign_instruction : VARIABLE "=" expression
                     { $main::VARIABLE{$item{VARIABLE}} = $item{expression} }


  instruction : print_instruction
              | assign_instruction


  startrule: instruction(s /;/)


_EOGRAMMAR_


sub expression {
  shift;
  my ($lhs,$op,$rhs) = @_;
  $lhs = $VARIABLE{$lhs} if $lhs=~/[^-+0-9]/;
  return eval "$lhs $op $rhs";
}


my $parser = Parse::RecDescent->new($grammar);


print "a=2\n";             $parser->startrule("a=2");
print "a=1+3\n";           $parser->startrule("a=1+3");
print "print 5*7\n";       $parser->startrule("print 5*7");
print "print 2/4\n";       $parser->startrule("print 2/4");
print "print 2+2/4\n";     $parser->startrule("print 2+2/4");
print "print 2+-2/4\n";    $parser->startrule("print 2+-2/4");
print "a = 5 ; print a\n"; $parser->startrule("a = 5 ; print a");
```

For problems or assistance with this site, email help@www.perl.com

Perl Runs Sweden's Pension System: A Fallback Application Built in Six Months Earns the Prime Role

by Ed Stephenson


Print This Article

6 July 2001

By the spring of 2000, Swedish government officials were pretty nervous. The delivery date on a major computer application was already a year late, causing a highly publicized delay in the start of the nation's new money-market pension system. With the legal deadline approaching for the system's launch, the government gave PPM--the agency in charge--six months to develop a fallback application, just in case the main commercial program hit another snag.

"Those were the days when stocks only went up," remarks Henrik Sandell, appraising the government's anxious mood at the time. PPM hired Sandell, an independent consultant, and Henrik Johnson from GlobeCom AB, an information technology services company in Stockholm, to design the back-up application for Sweden's Premium Pension System, with the simple directive that they provide something usable very quickly. "In the beginning PPM didn't much care what we used to develop our system, since they were fairly convinced it would never be put in production."

Given the time crunch, Sandell and Johnson's team built the application (known as *Pluto*) with Perl, using an Oracle database. Not only did they deliver the application on time, but when the commercial system proved difficult to customize and install--despite two years of modification by numerous developers--Pluto demanded a serious look. "An evaluation indicated that our system was faster and more likely to succeed than the original application," Sandell points out, at a fraction of the cost. PPM was convinced, and Pluto became the Premium Pension System's application of choice.

Any doubt that a Perl-based application would be robust enough to handle a large, mission-critical, financial-batch system has been dispelled by Pluto's performance with the Swedish national pension since its launch in the fall of 2000. More than 5 million customers take part in the system, representing every working person in Sweden born in 1938 and after. (The country's population is roughly 9 million.) A portion of every individual's employment tax, equal to 2.5 percent of his or her declared income, is set aside specifically for pension investment, and each person can choose up to 5 mutual funds from a list of 450 commercial funds approved by PPM. Combined, individual accounts exceed 6 billion in U.S. dollars.

Perl Wins the Comparison

Sandell has worked with GlobeCom's Henrik Johnson on several projects over the years,

using Perl for system administration and Web development since 1994. They immediately considered using Perl for Pluto as well, mainly because the language facilitates rapid development. But the complexity of the pension application demanded they also look at other alternatives.

The new PPM agency needed a system that would help it manage individual accounts, aggregate the orders, buy the mutual funds, keep track of how many fund-parts each individual owns, and pay dividends to fund holders, among many other functions. Calculations had to be absolutely correct, with multiple layers of check programs, and in order for PPM to run batches in a reasonable amount of time, the system needed to process a large number of accounts per minute. "Traceability" was also a big concern. "We had to be able to examine calculations afterwards in order to answer questions such as, 'Why did customer A buy fund B on the 4th of March?'" Sandell explains.

Good database connectivity gave Perl the nod over C++, and since Sandell and Johnson received project text files in various formats, Perl's ability to parse text with regular expressions was much better than COBOL or Oracle's PL/SQL. And Perl 5's relative longevity provided the stability they required. The only question was performance, given that the processing speed of interpreted languages such as Perl is much slower than that of compiled languages like C++.

An evaluation indicated that our [Perl] system was faster and more likely to succeed ... at a fraction of the cost. Pluto became the Premium Pension System's application of choice.

"That was an issue all along," Sandell notes, "and we made tests with large volumes from the start. The application seldom spends more than 20 percent of the time executing Perl code, while the database API and the database itself uses the other 80 percent. Even if we had an infinitely fast programming language, the overall performance would be only marginally better."

Convinced that processing speed wasn't a factor, Sandell and Johnson decided that Perl was best for the application. The toughest part was finding enough experienced developers in Sweden who knew Perl. The team sent requests to IT consulting firms (approved by PPM) throughout the country, and came up with a couple of good candidates, but they didn't get as many qualified programmers as they wanted.

So, although Pluto is written primarily in Perl, the team was forced to rely on other languages for various functions. They wrote the Web applications in Jscript, and other parts of the system in Visual Basic. Some functions were written in PL/SQL so they could be accessible from other languages using the system. "If we did the project again, we would probably strive to make more of it in Perl," Sandell concedes. But with so little time, they couldn't ask team members to learn a new language.

Ensuring a Smooth Transition

When it came to Perl, the team of four to six developers worked efficiently, programming on a Linux system and conducting peer reviews throughout the process. "One person would write the code and a second developer would check it," Sandell explains. "If the second developer could not understand the code, it was by definition incorrect and had to be improved, either through comments or, more often, through a rewrite since code that is hard to understand is often buggy."

The team--which included three project managers, three testers, two actuarials, and a lawyer to make sure the application was consistent with Swedish pension law--also had weekly deliveries of code for internal testing. Though these deliveries contained only part of the functionality, they had to be bug-free.

The testers would run the code through its paces and report whether any issues had been resolved from previous deliveries. "Perl proved to be a great language for doing the development," Sandell

comments, adding that the interpreted language not only reduced development time, but it also avoided problems with different sources and binaries. "The one thing we missed in Perl was a numeric data type with high precision. Since we worked with billions of dollars and wanted to keep track of cents, we needed to have at least 13 digits for amounts. We ended up using scaled integers--storing $1.23 as 123--which worked fine, but having a data type like Oracle's Number (12,6) would have saved a lot of effort."

When PPM approved Pluto for the new pension system, the team moved the application from the Linux box to the Hewlett-Packard UX V2600 production machine, and with Perl's portability, the transition was smooth. Despite the accomplishment, Sandell is not completely satisfied with Pluto's performance overall.

Any doubt that a Perl-based application would be robust enough to handle a large, mission-critical, financial-batch system has been dispelled by Pluto's performance since its launch in the fall of 2000.

"We would like the system to run about ten times faster," he insists. "We will try to achieve that using parallel batches in a future version. The limiting factor is not Perl, but Oracle." With Pluto in production more than six months, Sandell and Johnson are gradually turning over the maintenance to PPM's IT team. Now that they have proven Perl's worth in large and complex financial applications, they're looking to other countries that are considering market-based pension systems. The stock market may not be what it once was, but the investment made in this particular pension application has already paid tremendous dividends.

- Learn how large and small companies are putting Perl to work by reading other Perl Success Stories.
- Visit perl.oreilly.com for a complete list of O'Reilly's Perl books and www.perl.com for the latest news and CPAN updates.

Return to: perl.oreilly.com

- Home
- Registration
- Hotel/Travel
- See & Do
- Tutorials
- Sessions
- Evening Events
- BOFs
- Speakers
- Press
- Mail List
- Exhibitors
- Sponsors

An Annual Perl Community Gathering

The O'Reilly Perl Conference 5

July 23-27, 2001 -- San Diego, California

Don't miss the Great Open Source Debate

The O'Reilly Perl Conference has been a mainstay of Perl culture since 1997, gathering the leaders and young guns of Perl in an unequalled meeting of minds. TPC, as it is known, is a place where all Perl programmers can learn and share in the fun and diversity that is Perl. TPC is collocated with the **O'Reilly Open Source Convention**, which means you have the choice of 14 technology tracks in one convention. Come meet the wizards, connect with old friends, and get in on the cutting edge of Perl programming techniques.

Read Tim O'Reilly's reasons why **Open Source is Here to Stay**.

Enjoy San Diego

See & Do
Attractions
Kid's World

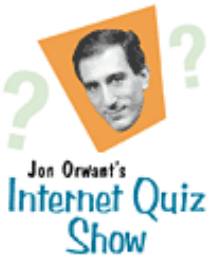
Why Attend the O'Reilly Perl Conference?

- Improve your programming skills.
- Meet the programmers behind your favourite module or package.
- Learn the latest techniques and idioms.
- Catch up with Perl 6.
- Connect with old friends.
- Hear the advice of those who have experience in the latest systems.
- Hang out with the Perl Gods.
- Get technical information without hype or sales pitches.
- Be a part of the Perl community!

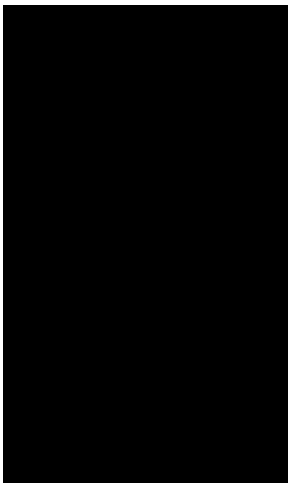
Featured Perl Talks

- Larry Wall's **State of the Onion**. Don't miss Larry talk about the state of the Perl World.
- Simon Cozens' tutorial on **Perl Internals**.
- Damian Conway on **Data Munging**.
- Mark-Jason Dominus with **Stolen Secrets of the Wizards of the Ivory Tower**.

Jon Orwant's Internet Quiz Show



Back by popular demand! The Internet Quiz Show pits teams of four against each other in a battle of wits, ingenuity, and speed. Jon Orwant hosts this fun filled fact hunt where knowledge of Internet technology and culture is put to the test.



The Team sign-up for the Internet Quiz is now closed. Make sure and come by Tuesday night and cheer on your favorite team.

Give a Lightning Talk!

Lightning talks are informal 5 minute talks. The short time limit keeps you focused and the audience interested, and makes them easy to prepare. **Sign up to give a lightning talk today!**

---

webmaster@oreilly.com

Quick access to NuSphere Direct, your personalized NuSphere connection.

Member Login

Not Registered?

SHORTCUTS

- Course Registration
- Events
- Partners
- Press Releases
- Media Coverage
- Support Online
- Tech Library

SITE SEARCH

Search

SPOTLIGHT

NuSphere announces two new products that will accelerate the development of PHP and MySQL driven web applications"

NuSphere Announces Nationwide Training Program for PHP Developers

Product Announcement! Version 2.2 of the *Advantage* product suite includes Gemini and PHPEd. To learn more, see What's New.

Enhanced MySQL announced and Allegro MicroSystems reaps the rewards

Visit our Tech

SPECIAL FEATURE

Gemini... it's here!

NuSphere proudly announces the commercial release of Enhanced MySQL with NuSphere MySQL Advantage and NuSphere Pro Advantage V2.2. Enhanced MySQL brings together MySQL, the #1 open source database, and the power of Gemini, NuSphere's MySQL table type. Gemini tables provide row-level locking, robust transaction support, and reliable crash recovery. With Gemini, Enhanced MySQL delivers the enhanced database technologies you need to develop business-critical web applications. Learn more about Enhanced MySQL.

Business Partners and Resellers -- Join Us!

Selling a MySQL-based application? Embed the extra value of NuSphere MySQL and blow away your competition! Reselling software? Give your customers the open source web development platform they need to do the job right. NuSphere Business Partners and Resellers are eligible for product discounts, training, support, co-marketing and much more. Complete a brief application and we'll get back to you right away with information on the Partner Program that's right for you. Can't wait? Contact sales at 1-781-280-4600, or sales@nusphere.com.

UPCOMING EVENTS

Official MySQL™ Training Materials

NuSphere MySQL™

NEXT CLASS SCHEDULED

09-Jul - Chicago
09-Jul - Vancouver, B.C
09-Jul - Bedford
16-Jul - San Francisco
16-Jul - Bedford
19-Jul - Dallas
30-Jul - Atlanta

O'REILLY OPEN SOURCE SOFTWARE CONVENTION

Library to learn more about the components of the open source web development platform

NuSphere Product Updates

NuSphere MySQL Advantage Version 2.0 for Professional Developers

NuSphere MySQL Advantage has Enhanced MySQL and a complete web development environment that includes the integrated open source components Apache, PHP and Perl.

NuSphere MySQL Advantage comes with:

- Complete integrated web platform
- Enhanced MySQL database
- Automatic updates 4 times a year
- A new professional application in each update
- 30 days web and email-based technical support for MySQL
- Full year of installation support

For more information on the features of NuSphere MySQL Advantage, check out What's New.

Each quarterly update of NuSphere MySQL Advantage comes with a new professional application and enhancements to the platform. This release includes:

Professional Application - Bugzilla

Bugzilla is the popular open source bug tracking application. With it, you can report bugs and assign them to the appropriate developers. Developers can use Bugzilla to keep a to-do list as well as to prioritize, schedule and track dependencies. Traditionally a *nix application, NuSphere has ported Bugzilla to Windows and created a seamless installation from within NuSphere MySQL Advantage so that you can take full advantage of it right out of the box, no matter what operating system you are using. You can learn more about Bugzilla at http://www.mozilla.org/projects/bugzilla/.

Enhanced MySQL

Gemini is a new MySQL table type designed by NuSphere to provide all the functionality required for highly granular, transaction-intensive database applications. This debut beta release of Gemini integrates seamlessly into MySQL, resulting in an enhanced MySQL with the performance, scalability and reliability of the most expensive database systems. With automatic crash recovery, failover clusters, table and site replication, and flexible backup solutions, you can now use MySQL to develop mission-critical applications.

All you need to do is set the table type to use Gemini:

```
CREATE TABLE …  TYPE = GEMINI;
```

and you're ready to take full advantage of everything Gemini has to offer.

Enhanced MySQL will change the rules for OLTP systems built with open source software. With the proven enterprise-class technology of Gemini at its core, MySQL is ready to take over for traditional, commercial databases.

REMEMBER!

Each quarterly update of NuSphere MySQL Advantage comes with a new

ALREADY A MYSQL USER?

Get NuSphere MySQL Advantage today.

Only $299.00

NuSphere MySQL Advantage is the web development platform that keeps on giving... Every quarter you will get the most up-to-date components and professional applications that will make developing, maintaining and deploying database-driven web applications effortless.

Buy Now

MORE LINKS

- MySQL Training
- Upcoming Events
- Product Offerings
- Email sales@nusphere.com

professional application. Back issues are available for an additional cost through NuSphere Direct. Don't miss out, subscribe today.

---

top

Home | About Us | Products | Training | Support Services | NuSphere Direct

Products

Overview
What's New
NuSphere MySQL Advantage
NuSphere PHPEd Advantage
NuSphere Pro Advantage
Advantage Plus
NuSphere MySQL Getting Started
Professional Tools and Applications

Tech Library
Support Services
Download
Buy Now

Products

Explore the diverse and comprehensive line of web development and deployment environments that NuSphere has built on the best-of-breed open source components you already know and love.

Experience the Advantage. The NuSphere Advantage.

NuSphere products focus on three Advantage suites: NuSphere® MySQL™ Advantage, NuSphere® PHPEd Advantage, and NuSphere® Pro Advantage. Each software suite pairs the reliability and cost-effectiveness of Apache, MySQL, PHP and Perl with new technology enhancements for building business-critical web applications. NuSphere has created an integrated foundation that allows companies to deploy reliable, cost-effective, enterprise-class applications for Windows, UNIX and Linux environments.

NuSphere products begin with a web development platform that includes integrated, tested versions of Apache, MySQL, PHP and Perl. We then enhance the core platform with technology and tools that boost productivity and accelerate your success. The real Advantage is that every quarter we update the web development platform and deliver new applications and tools. The Advantage suites:

- **NuSphere MySQL Advantage** includes the integrated web development platform and Enhanced MySQL. Enhanced MySQL brings together MySQL and the power of Gemini, a MySQL table type designed by NuSphere. learn more
- **NuSphere PHPEd Advantage** includes the integrated web development platform and the power of PHPEd, an Integrated Development Environment (IDE) for Windows. learn more
- **NuSphere Pro Advantage** is a complete Internet Application Platform (IAP), which includes the integrated web development platform, Enhanced MySQL and PHPEd. Designed for professional developers, this Advantage provides a complete front-to-back solution for creating, deploying and managing your web infrastructure. learn more
- **Advantage Plus.** "Super size" your Advantage purchase by adding phone support for MySQL, PHP or both learn more

Each Advantage purchase delivers:

- An integrated web development platform.
- Professional applications and tools to boost productivity.
- Quarterly updates of the platform and "extras" for one year.
- One year of installation support.

Secure your *passport* to success...

Buy Now

Each NuSphere product includes our integrated open source web development platform. Advantage products technology enhancements plus quarterly updates of the platform and "extras".

what's new

RELEASE INFO

Version 2.2 includes the environment components:

- MySQL V3.23.38
- Apache V1.3.20
- PHP V4.0.5
- Perl V5.6.1
- Webmin V.85

Buy Now

Please note that NuSphere PHPEd Advantage and NuSphere Pro Advantage will not be available for online purchase until June 22, 2001. We apologize for the inconvenience. If you would like to place an order, please call us at 781-280-4600 or email sales@nusphere.com.

TOP

Just Getting Started?

NuSphere MySQL

Get the integrated environment that brings together today's popular open source components into one 10 minute install. NuSphere MySQL is an integrated distribution of MySQL, Apache, Perl, and PHP, and comes with binaries and source code for RedHat Linux and Windows platforms. If you are already a MySQL user, step up to the NuSphere MySQL Advantage.

NuSphere MySQL - $79.00

Buy Now

"Using the NuSphere platform was a wise choice. It makes our database maintenance and implementation much easier."
-- Eric Ho, CEO, homeworkhelp.com

RELEASE INFO

Currently shipping (Version 1.13.10):

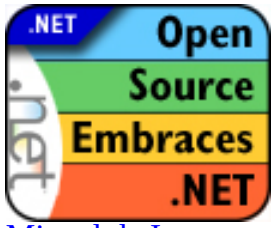
- MySQL V3.23.38
- Apache V1.3.20
- PHP V4.0.5
- Perl V5.6.1
- Webmin V.85

Supported platforms plus system requirements:

- Redhat Linux (6.2 and higher on IA32)
- Sun Solaris (2.6, 2.7, and 2.8 on SPARC)
- Windows 95/98, NT, and 2000

- 32 MB RAM
- 100 MB disk space

How will Mono unite the ADO.NET and GNOME-DB object models?



Miguel de Icaza on .NET: the new development environment for the next 20 years.

C# Regular Expressions

Brad Merrill, coauthor of C# Essentials, offers an overview of how regular expressions are used in the C# environment. The article concludes with a set of expressions used in other environments, which Merrill has updated for C# programmers. [oreilly.com]

Microsoft Plans Shared Source .NET

Microsoft has announced its first foray into the waters of publicly shared source. Tim O'Reilly talks to Microsoft program manager (and FreeBSD sympathizer) Dave Stutz about

What's New

VS.NET Beta 2 Has Time Bomb

Microsoft warns TechEd conference attendees that the Visual Studio.NET Beta 2, which was handed out at the show, contains a 30-day "time bomb" that will cause the product to expire on July 31.

Redmond's plans to release shared-source code of parts of the .NET framework. [.NET DevCenter]

Writing ASP.NET Web Forms with C#
Budi Kurniawan presents Web Forms as a programming model in ASP.NET. He covers server controls with an overview of how separation between business logic and presentation layer is achieved in Web Forms along with new session management strategy. Budi concludes with his programming experience building Web Forms using the C# language. [.NET DevCenter]

Contrasting C# and Java Syntax
Raffi Krikorian discusses the differences in syntax between these two object-oriented programming languages. [.NET DevCenter]

Comparing C# and Java
Many people think that C# is a clone of, or Microsoft's replacement for, Java. Is it true? [.NET DevCenter]

Conversational C# for Java Programmers, Part 1
Raffi Krikorian kicks off the first of a five part series on C# for Java as well as .NET developers. [.NET DevCenter]

Why O'Reilly and .NET?
At O'Reilly, because we are closely identified with open source, I expect that the opening of our .NET DevCenter on O'Reilly Network will raise some eyebrows. Well, here's why we are covering it. [.NET DevCenter]

Weblogs

Installing .NET changes MSIE User-Agent
When you install Microsoft's .NET SDK, it inserts the .NET Common Language Runtime (CLR) version number into the Microsoft Internet Explorer identification string -- letting all the sites you visit know you are a .NET user. [Marc Hedlund]

More ▶

News

MS, CNET On 7-Day Messenger Outage
[Source: Privacy Digest]

Opening Up .Net to Everyone [Source: Wired News]

Ports System As A Strategy Against .NET?
[Source: Slashdot Org latest news headlines]

Reverse Engineering .NET - Good, Bad or Inevitable? [Source:

Microsoft's Shared Source Philosophy
Microsoft senior vice president Craig Mundie has accepted Tim O'Reilly's invitation to present his controversial Shared Source philosophy in a keynote address at the O'Reilly Open Source Convention. A panel discussion with Red Hat CTO Michael Tiemann, Tim O'Reilly, and others will follow Mundie's talk. [oreilly.com]

JVM to .NET: I'm Not Dead Yet!
Although Microsoft is loath to admit it, .NET is really their answer to Sun. However, the Java language, the Java Virtual Machine, and CORBA are still a threat. [O'Reilly Network]

Can the Samba Story be Retold?
As we look at the emergence of .NET, are there lessons to learn from the past? Samba, the open source alternative to Windows NT systems dominating file and print servers on corporate LANs, may be one of those lessons worth examining. [O'Reilly Network]

.NET in Flux
.NET strategy is in flux. Windows XP Server or Windows 2002 will now be 6 months behind schedule. [O'Reilly Network Weblogs]

.NET Still In Doubt?
This survey is yet another indication of doubts concerning .NET platform. [O'Reilly Network Weblogs]

COM+ to .NET: Fast Forward Your VB Components
If you work with ASP and use COM+ components created with Visual Basic, you'll want to make sure your components migrate

Slashdot Org latest news headlines]

Microsoft Plans "Shared Source" .NET [Source: Slashdot Org latest news headlines]

NEWS: Lotus takes on .Net [Source: DominoPower Magazine]

Edit This, Bucko [Source: Privacy Digest]

Hailstorm: Open Web Services Controlled by Microsoft [Source: Privacy Digest]

QXL ricardo Inks .NET Pact [Source: Latest Internet E-Commerce News headlines]

Banks fail cyberattack test [Source: Privacy Digest]

In Microsoft do you trust? [Source: Privacy Digest]

Privacy Woe? [Source: Privacy Digest]

More ▶

well to the new .NET environment. Shelly Powers introduces the types of changes headed your way and how they will impact your existing VB components and component development. [oreilly.com]

Click here for all .NET articles listed in chronological order.

Events

O'Reilly Open Source Convention
San Diego, CA *Jul. 23, 2001*

Conference.NET
San Francisco, CA *Aug. 13, 2001*

More ▶

July 11, 2001

search

OpenBSD Local Root Exploit

Noel Davis shows us a race condition in the OpenBSD kernel; cross-site request forgeries; a new version of tcpdump; buffer overflows in rxvt, fetchmail, the HP-UX implementation of CDE, and UW-IMAP; a symbolic link race condition in mandb; and vulnerabilities in SITEWare Editor's Desktop, Apache under Mac OS X client, LPRng, Caldera's Volution, and Slackware 7.1's /etc/shells [Linux DevCenter]

Deploying Web Applications to Tomcat

James Goodwill takes us through the web application deployment process for the Apache Tomcat Web server. [ONJava.com]

Securing a PHP Installation

Darrell Brogdon shows us a few basic things that should be done to secure a PHP installation. [PHP DevCenter]

Apache 1.3.19 Released

Apache 1.3.19 was released on the 28th February 2001. This release addresses a minor security flaw with directory indexes, fixes minor bugs found in the 1.3.17 release, fixes mod_rewrite, and adds some minor features. Version 1.3.18 was never

Installing Apache 2.0 in Apache 2.0 Basics: In the first of a series of columns, Apache Group developer Ryan Bloom introduces Apache 2.0 and shows you how to get it up and running.

Migrating from Apache 1.3 to Apache 2.0 In the second in his series of columns on Apache 2.0, Ryan Bloom covers some of the lessons learned at apache.org when installing the new version, including deciding which multiprocessing modules

Articles:

- HTTP Wrangler

O'Reilly Book Excerpts

- Securing Your Apache Server
- Apache Under Windows
- Setting Up a Win32 Server

DOCUMENTATION

Python Database API Specification 2.0

mx-Series of Python Extension Packages

Hector: Distributed Objects in Python

Systems programming in Python

released. [Apache Week]

Under Development

Covering Apache 1.3.19, a WebDAV problem and ap_r in Apache 2.0 (again) [Apache Week]

Featured Articles

Improving mod_perl performance; a new server-side programming language; tuning PHP for speed [Apache Week]

ApacheCon 2001

The biggest Apache event this year, ApacheCon 2001, is only a month away [Apache Week]

Apache Week book giveaway

Congratulations to our eight winners who each get a copy of "mod_perl Pocket Reference" [Apache Week]

The PHP Configuration File -- Part One

Darrell Brogdon explains how the php.ini file works. [PHP DevCenter]

APACHE WRANGLER

Articles featuring Apache Web server and HTTP-related issues.

AxKit: An XML-Delivery Toolkit for Apache

Rael Dornfest introduces AxKit, an XML application server bringing together XSLT, Perl, and Apache for unrivaled content transformation.

LAMP Lighter: The Apache Toolbox

An overview of the Apache Toolbox, a Swiss army knife of a script, providing a customizable, menu-driven interface to downloading and compiling a LAMP (Linux, Apache, MySQL, PHPherlthon) -- minus the Linux -- installation.

Module Tour: Embedding Python with mod_python and mod_snake

Rael Dornfest takes quick look at mod_python and mod_snake, two modules that embed a Python (MPMs) to use, setting filters, and working around complications in IPv6 support.

Industrial-Strength Webcasting with mod_mp3

The Apache module mod_mp3 turns your web server into an MP3 server, comparable to media servers like those from RealNetworks and Apple's QuickTime division. Additional features, like compatibility with RSS 1.0, let you syndicate your audio stream, or subscribe to others' webcasting stations.

Apache.org Server Compromised Noel Davis shows us the compromise of the Apache Software Foundation Server; buffer overflows in yppasswd, Qpopper, and mailtool; vulnerabilities in TWIG, webmin, and GnuPG; a new type of attack against sendmail; and discuss the use of the user nobody.

Tkinter: GUI programming with

More

TECH JOBS

To learn about the hottest technical jobs, click here.

EVENTS

O'Reilly Open Source Convention
07/23/2001
San Diego, Calif.

More Events

interpreter right into your Apache server.

Module Tour: mod_info
Rael Dornfest takes us on a whirlwind tour of Apache's `mod_info` module.

Module Tour: mod_status
Rael Dornfest takes us on a tour of Apache's `mod_status` module.

Installing mod_perl from RPM
It's easy to install mod_perl using the Red Hat package manager. Configuring it is trickier.

Log Rhythms
An introduction to the Apache server logs and their place in monitoring, security, marketing, and feedback.

An Amble Through Apache Configuration
Rael Dornfest introduces the Apache web server's configuration file, httpd.conf, and explains its settings.

Getting, Installing, and Running Apache
How to install the web server from scratch, binary, or using Red Hat Package Manager.

Introducing Apache
The first in a series of articles about Apache, the most popular web server software available. In the coming weeks, I'll talk about how to install Apache, its care and feeding, simple tricks to keep it running smoothly, and powerful modules you can add to extend its capabilities.

IN THE APACHE DIGEST

Error .../CmsInit.ASP
I have a problem with the following line. it's written down every few seconds in the two files httpd.error_log and httpd.access_log: in the first file "file does not exist: /usr/local.../scripts/cm...
oreillynet.apache

20010710145052

How do I make a Virtualhost with .htaccess and mod_rewrite only
I have two domains parked at my host pointing to the same place. I want it to point somewhere else. domaina.com -> domainb.com/folder or domaina.com -> /home/enricong/public_html/folder I dont get acc...
oreillynet.apache

20010710095118

POST error
I have set up Apache 1.3.18 on a RedHat Linux system and am trying to write a script to do a POST to a page. When doing so, I get the following error: The requested method POST is not allowed. How ...
oreillynet.apache

20010710095117

[phpLittleChat 0.1 Beta (Default)]
A little PHP chat without using a constant connection to a Web server.
[Freshmeat Daily News]

[New article in the Microsoft focus area: Running Snort on IIS Web Servers, Part Two: Advanc]

[Security Focus]

[ApacheCon 2001 announces 70 speakers]
From BusinessWire: The Apache Software Foundation today announced over 70 speaker sessions for this spring's ApacheCon 2001, the annual conference and exhibition for Apache-related and open-source software. The next wave of Apache deployed projects will be discussed at the Santa Clara Convention Centre in the heart of Silicon Valley, 4 through 6 April, 2001. ApacheCon 2001 will provide a look at the future of Apache ...
[NewsForge]

[Python Software Foundation launched]
From PR Newswire: "At the ninth Python Conference, Guido van Rossum announced the launch of the Python Software Foundation (PSF). Modeled after the successful Apache Software Foundation, the PSF's mandate is to provide educational, legal and financial resources to the Python community. Responsible for holding Python's intellectual property, the PSF will also act as an educational resource, maintain the Python website, and ...
[NewsForge]

[Apache regains ground lost to MS]
The Register Mar 5 2001 7:14PM ET
[Moreover Microsoft news]

APACHE FAQs

[Why does Apache 'lock-up' when accessed from Netscape 2?]

[How can I control what server information is given in the HTTP response header?]

[Where can I find mod_rewrite rulesets which already solve particular URL-related problems?]

More ▶

MODULE NEWS

[mod_layout 2.10.2 released]
The bug fix is that cookies are now passed during a MAIN request when using merge regardless of HTTP header override usage.

[mod_plsql 0.3.5 (Default)]
Allows you to create Web applications using Oracle stored procedures.

[mod_ssl 2.8.1-1.3.19]
Upgraded to Apache 1.3.19 as the base version;

Apache Toolbox 1.5.12 (Default)
An Apache compilation toolbox.
Freshmeat Daily News

Consider Apache
This Linuxtoday.com.au article asks you to consider the Apache project, and all it's contributed to the core of the web. " If you're a server administrator by profession - then of course you would know quite a bit. But for the rest of us IT professionals out there, I'm about to give you a crash course in how most of the WWW road on the Internet is built, and how it came to be that way."
NewsForge

Red Carpet
I played with Red Carpet and Evolution a bit today. Red Carpet is cool, although it told me that it would have to remove mod_ssl to install a security update to Apache, which doesn't seem all that useful. I still haven't found the docs on creating my own Red Carpet channels. I couldn't get Evolution to read my mailbox, so I gave up on it for now.
Hack the Planet

Avacet Application Engine 2.0b2
DeWitt Clinton announced the availability of the second beta of theAvacet Application Engine tothe mod_perl mailing list.
Take23

iVote - mod_perl based voting engine
Michael Holve sent us a communique about his new mod_perl based votingengine, iVote, whichwas designed for voting on different images.
Take23

sedum - HTTP Server With Direct Database

Conditionally adjusted the source to also build quietly under latest OpenSSL 0.9.7-dev versions; Extended FAQ entry for MSIE problems.Fixed: Applied a bunch of (untested!) adjustments and fixes for the Win32 platform, as posted to modssl-users some time ago by various people; The SSLCipherSuite example in 'httpd.conf-dist'. The string EXP56 is actually EXPORT56, although in OpenSSL internally the var

Apache-ContentHandler-1.3.2
mod_perl extension for uniform application

mod_proxy_add_forward.c 1.2 (Default)
Adds an 'X-Forwarded-For' header to outgoing proxy requests like Squid.

mod_ssl 2.8.1-1.3.19 (Default)
Apache Interface to OpenSSL

mod_trigger 1.0 (Default)
Provides internal triggers and events to Apache based on actions/URIs/args/etc.

Apache-ProxyRewrite-0.12
mod_perl URL-rewriting proxy

Apache-AuthzCache-0.04

Access
Full standard HTTP server. Works on Windows NT or 95/98 systems. Also supports sub-document inclusion, parametric HTML, directives to GET & PUT data directly to/from databases.
New Entries at Internet Product Watch

mod_perl Cache Authorization Module

slash 1.1.5 (Dev)
A database-driven news and message board, using mod_perl and MySQL.

PageKit 0.91 Released
The new version of PageKit, a powerful application framework based onmod_perl, is available from CPAN.PageKit is still in alpha, but is worth a look.

Apache::ProxyRewrite 0.10 Released
Apache::ProxyRewrite acts as a reverse-proxy that will rewrite URLs embedded in HTMLdocuments per apache configuration directives. It should soon be available on CPAN.

mod_perl guide 1.28 Released
Stas announced a new release of the guide today. Matt has kindlyupdated the Take23 version as well.

mod_perl Pocket Reference available
The mod_perl Pocket Reference is now available.

iVote - mod_perl based voting engine
Michael Holve sent us a

communique about his new mod_perl based votingengine, iVote, whichwas designed for voting on different images.

mod_perl 1.25 released

Looks like it is going to be a bumper week for releases. Last nightDoug MacEachern announced mod_perl1.25.