**[linux c学习笔记----select函数详解](http://lobert.iteye.com/blog/1768502)**

**博客分类：**

* [Unix/linux](http://lobert.iteye.com/category/190159)
* [c/c++](http://lobert.iteye.com/category/210114)

[select](http://www.iteye.com/blogs/tag/select)[I/O](http://www.iteye.com/blogs/tag/I%2FO)[linux](http://www.iteye.com/blogs/tag/linux)[c](http://www.iteye.com/blogs/tag/c)

select系统调用是用来让我们的程序监视多个文件句柄(file descriptor)的状态变化的。程序会停在select这里等待，直到被监视的文件句柄有某一个或多个发生了状态改变。

**何为文件句柄？？？？**  
文件句柄在Linux里很多，如果你man某个函数，在函数返回值部分说到成功后有一个文件句柄被创建的都是，如man socket可以看到“On success, a file descriptor for the new socket is returned.”而man 2 open可以看到“open() and creat() return the new file descriptor”，**其实文件句柄就是一个整数**，看socket函数的声明就明白了：  
int socket(int domain, int type, int protocol);  
当然，我们最熟悉的句柄是0、1、2三个，0是标准输入，1是标准输出，2是标准错误输出。0、1、2是整数表示的，对应的FILE \*结构的表示就是stdin、stdout、stderr，0就是stdin，1就是stdout，2就是stderr。  
比如下面这两段代码都是从标准输入读入9个字节字符：  
  
  
#include <stdio.h>  
#include <unistd.h>  
#include <string.h>  
int main(int argc, char \*\* argv)  
{  
        char buf[10] = "";  
        read(0, buf, 9); /\* 从标准输入 0 读入字符 \*/  
        fprintf(stdout, "%s\n", buf); /\* 向标准输出 stdout 写字符 \*/  
        return 0;  
}  
/\* \*\*上面和下面的代码都可以用来从标准输入读用户输入的9个字符\*\* \*/  
#include <stdio.h>  
#include <unistd.h>  
#include <string.h>  
int main(int argc, char \*\* argv)  
{  
        char buf[10] = "";  
        fread(buf, 9, 1, stdin); /\* 从标准输入 stdin 读入字符 \*/  
        write(1, buf, strlen(buf));  
        return 0;  
}

继续上面说的select，就是用来监视某个或某些句柄的状态变化的。select函数原型如下：  
int select(int nfds, fd\_set \*readfds, fd\_set \*writefds, fd\_set \*exceptfds, struct timeval \*timeout);  
函数的最后一个参数timeout显然是一个超时时间值，其类型是struct timeval \*，即一个struct timeval结构的变量的指针，所以我们在程序里要申明一个struct timeval tv;然后把变量tv的地址&tv传递给select函数。struct timeval结构如下：  
  
struct timeval {  
             long    tv\_sec;         /\* seconds \*/  
             long    tv\_usec;        /\* microseconds \*/  
         };

第2、3、4三个参数是一样的类型： fd\_set \*，即我们在程序里要申明几个fd\_set类型的变量，比如rdfds, wtfds, exfds，然后把这个变量的地址&rdfds, &wtfds, &exfds 传递给select函数。这三个参数都是一个句柄的集合，第一个rdfds是用来保存这样的句柄的：当句柄的状态变成可读的时系统就会告诉select函数返回，同理第二个wtfds是指有句柄状态变成可写的时系统就会告诉select函数返回，同理第三个参数exfds是特殊情况，即句柄上有特殊情况发生时系统会告诉select函数返回。特殊情况比如对方通过一个socket句柄发来了紧急数据。如果我们程序里只想检测某个socket是否有数据可读，我们可以这样：

**C代码  [收藏代码](javascript:void())**

1. fd\_set rdfds; /\* 先申明一个 fd\_set 集合来保存我们要检测的 socket句柄 \*/
3. **struct** timeval tv; /\* 申明一个时间变量来保存时间 \*/
5. **int** ret; /\* 保存返回值 \*/
7. FD\_ZERO(&rdfds); /\* 用select函数之前先把集合清零 \*/
9. FD\_SET(socket, &rdfds); /\* 把要检测的句柄socket加入到集合里 \*/
11. tv.tv\_sec = 1;
12. tv.tv\_usec = 500; /\* 设置select等待的最大时间为1秒加500毫秒 \*/
14. ret = select(socket + 1, &rdfds, NULL, NULL, &tv); /\* 检测我们上面设置到集合rdfds里的句柄是否有可读信息 \*/
16. **if**(ret < 0) perror("select");/\* 这说明select函数出错 \*/
18. **else** **if**(ret == 0) printf("超时\n"); /\* 说明在我们设定的时间值1秒加500毫秒的时间内，socket的状态没有发生变化 \*/
20. **else** { /\* 说明等待时间还未到1秒加500毫秒，socket的状态发生了变化 \*/
21. printf("ret=%d\n", ret); /\* ret这个返回值记录了发生状态变化的句柄的数目，由于我们只监视了socket这一个句柄，所以这里一定ret=1，如果同时有多个句柄发生变化返回的就是句柄的总和了 \*/
22. /\* 这里我们就应该从socket这个句柄里读取数据了，因为select函数已经告诉我们这个句柄里有数据可读 \*/
24. **if**(FD\_ISSET(socket, &rdfds)) { /\* 先判断一下socket这外被监视的句柄是否真的变成可读的了 \*/
25. /\* 读取socket句柄里的数据 \*/
26. recv(...);
27. }
28. }

注意select函数的第一个参数，是所有加入集合的句柄值的最大那个值还要加1。比如我们创建了3个句柄：

**C代码  [收藏代码](javascript:void())**

1. **int** sa, sb, sc;
2. sa = socket(...); /\* 分别创建3个句柄并连接到服务器上 \*/
3. connect(sa,...);
4. sb = socket(...);
5. connect(sb,...);
6. sc = socket(...);
7. connect(sc,...);
9. FD\_SET(sa, &rdfds);/\* 分别把3个句柄加入读监视集合里去 \*/
10. FD\_SET(sb, &rdfds);
11. FD\_SET(sc, &rdfds);
12. 在使用select函数之前，一定要找到3个句柄中的最大值是哪个，我们一般定义一个变量来保存最大值，取得最大socket值如下：
14. **int** maxfd = 0;
15. **if**(sa > maxfd) maxfd = sa;
16. **if**(sb > maxfd) maxfd = sb;
17. **if**(sc > maxfd) maxfd = sc;
18. 然后调用select函数：
19. ret = select(maxfd + 1, &rdfds, NULL, NULL, &tv); /\* 注意是最大值还要加1 \*/

   同样的道理，如果我们要检测用户是否按了键盘进行输入，我们就应该把标准输入0这个句柄放到select里来检测，如下：

**C代码  [收藏代码](javascript:void())**

1. FD\_ZERO(&rdfds);
2. FD\_SET(0, &rdfds);
3. tv.tv\_sec = 1;
4. tv.tv\_usec = 0;
5. ret = select(1, &rdfds, NULL, NULL, &tv); /\* 注意是最大值还要加1 \*/
6. **if**(ret < 0) perror("select");/\* 出错 \*/
7. **else** **if**(ret == 0) printf("超时\n"); /\* 在我们设定的时间tv内，用户没有按键盘 \*/
8. **else** { /\* 用户有按键盘，要读取用户的输入 \*/
9. scanf("%s", buf);