Select模型原理

利用select函数，判断套接字上是否存在数据，或者能否向一个套接字写入数据。目的是防止应用程序在套接字处于锁定模式时，调用recv（或send）从没有数据的套接字上接收数据，被迫进入阻塞状态。

select参数和返回值意义如下：

int **select** (

 IN **int** nfds,                           //0,无意义

 IN OUT **fd\_set\*** readfds,      //检查可读性

 IN OUT**fd\_set\*** writefds,     //检查可写性

 IN OUT f**d\_set\***exceptfds,  //例外数据

 IN const**struct timeval\*** timeout   //函数的返回时间

 );

struct  **timeval** {

        long    tv\_sec;        //秒

        long    tv\_usec;     //毫秒

};

select返回fd\_set中可用的套接字个数。

fd\_set是一个SOCKET队列，以下宏可以对该队列进行操作：

FD\_CLR( s, \*set) 从队列set删除句柄s;

FD\_ISSET( s, \*set) 检查句柄s是否存在与队列set中;

FD\_SET( s, \*set )把句柄s添加到队列set中;

FD\_ZERO( \*set ) 把set队列初始化成空队列.

1：用FD\_ZERO宏来初始化我们感兴趣的fd\_set。

也就是select函数的第二三四个参数。

2：用FD\_SET宏来将套接字句柄分配给相应的fd\_set。

如果想要检查一个套接字是否有数据需要接收，可以用FD\_SET宏把套接接字句柄加入可读性检查队列中

3：调用select函数。

如果该套接字没有数据需要接收，select函数会把该套接字从可读性检查队列中删除掉，

4：用FD\_ISSET对套接字句柄进行检查。

如果我们所关注的那个套接字句柄仍然在开始分配的那个fd\_set里，那么说明马上可以进行相应的IO操 作。比如一个分配给select第一个参数的套接字句柄在select返回后仍然在select第一个参数的fd\_set里，那么说明当前数据已经来了， 马上可以读取成功而不会被阻塞。