

socket编程

▼ setsockopt

▼ SO_REUSEADDR

- 功能设置地址重用，可以绑定TIME_WAIT状态的端口。
- 用法：int reuse=1;
setsockopt(socketFd,SOL_SOCKET,SO_REUSEADDR,&reuse,sizeof(reuse);

▼ SO_RCVBUF, SO_SNDBUF

- 功能：设置发送缓冲区大小和接收缓冲区大小。
- 用法：int rcvBufSize=32*1024;
setsockopt(socketFd,SOL_SOCKET,SO_RCVBUF,&rcvBufSize,sizeof(rcvBufSize);



▼ SO_RCVLOWAT, SO_SNDLOWAT

- 功能：设置缓冲区下限，缓冲区中的数据超过下限值之后，描述符才就绪。
- 用法：int rcvLowAt = 10;
setsockopt(socketFd,SOL_SOCKET,SO_RCVLOWAT,&rcvLowAt ,sizeof(rcvLowAt));

▼ send和recv

- send只是把数据从用户态的buf拷贝到描述符的缓冲区中，完成就返回拷贝的字节数。并不代表数据真的已经通过网卡发送出去了

▼ send和recv的返回值：

- 1. 成功返回发送/接收的字节数。
- 2. 非阻塞模式下，如果描述符未就绪，返回-1。
-  3. 对端断开，描述符可读，recv返回0。
-  4. 如果对端断开，send第一次返回-1，此时再send，会受到SIGPIPE信号。

▼ send和recv的第四个参数flags

- MSG_PEEK；只对recv有效并不真正把数据从缓冲区中取走，而是拷贝的方式。
- MSG_DONTWAIT；设置为非阻塞模式，单次有效，如果描述符未就绪，返回-1。

▼ 非阻塞编程

▼ 通过fcntl把描述符设置为非阻塞的。

- 先通过F_GETFL，获取当前描述符状态，int status = fcntl(fd,F_GETFL);
- 把非阻塞的属性加进来 status = status|O_NONBLOCK;
- 通过F_SETFL，把修改后的状态设置给描述符，fcntl(fd,F_SETFL,status);
- 通过recv和send的第四个参数，把单词的读/写设为非阻塞。

▼ ● IO多路复用

- select
- epoll

▼ 五种IO模型

▼ 五种IO模型

- 同步阻塞IO模型
- 同步非阻塞IO模型
- ● 多路复用IO模型：select, poll, epoll
- 信号驱动IO模型：跟IO多路复用比没有优势，很少使用。不是异步IO。
- ▼ 异步IO模型
 - aio系列接口：aio_read等等
 - 由内核帮我们把数据从内核缓冲区读到用户缓冲区，完全不许我们参与
- 阻塞和非阻塞
- 同步和异步