创建可用的套接字，需要用到下面的函数

**int socket(int domain, int type, int protocol)**

domain指的是PF\_INET、PF\_INET6以及PF\_LOCAL等，表示什么用的套接字

type 可用的值是：

1. SOCK\_STREAM:表示字节流，表示TCP
2. SOCK\_DGRAM：表示数据报，对应UDP
3. SOCK\_RAW：表示的原始套接字

protocal基本废弃了，一般写0就可以了

bind相当于绑定套接字和套接字地址，绑定到对应的address上

**bind(int fd, sockaddr \* addr, socklen\_t len)**

其中sockaddr \* addr其实可以理解成void \*，因为后面可以智能判断。

一开始写addr时候我们一般不知道自己的addr，我们可以用通配地址来解决这个问题

1. IPv4可以用INADDR\_ANY来解决问题
2. IPv4 可以用IN6ADDR\_ANY来完成通配地址的设置

**struct sockaddr\_in name;**

**name.sin\_addr.s\_addr = htonl (INADDR\_ANY); /\* IPV4 通配地址 \*/**

//htonl将一个32位数从主机字节顺序转换成网络字节顺序（网络协议处理字节整数的视乎用大端字节顺序）

//例子如下

**#include <stdio.h>**

**#include <stdlib.h>**

**#include <sys/socket.h>**

**#include <netinet/in.h>**

**int make\_socket (uint16\_t port)**

**{**

**int sock;**

**struct sockaddr\_in name;**

**/\* 创建字节流类型的 IPV4 socket. \*/**

**sock = socket (PF\_INET, SOCK\_STREAM, 0);**

**if (sock < 0)**

**{**

**perror ("socket");**

**exit (EXIT\_FAILURE);**

**}**

**/\* 绑定到 port 和 ip. \*/**

**name.sin\_family = AF\_INET; /\* IPV4 \*/**

**name.sin\_port = htons (port); /\* 指定端口 \*/**

**name.sin\_addr.s\_addr = htonl (INADDR\_ANY); /\* 通配地址 \*/**

**/\* 把 IPV4 地址转换成通用地址格式，同时传递长度 \*/**

**if (bind (sock, (struct sockaddr \*) &name, sizeof (name)) < 0)**

**{**

**perror ("bind");**

**exit (EXIT\_FAILURE);**

**}**

**return sock**

**}**

listen使得socket能够进行通信

**int listen (int socketfd, int backlog)**

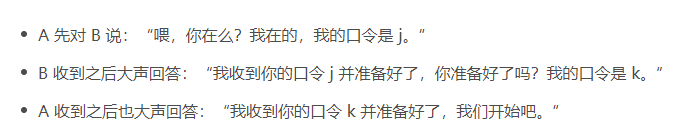
accept建立连接

**int accept(int listensockfd, struct sockaddr \*cliaddr, socklen\_t \*addrlen)**

connect 拨打连接

**int connect(int sockfd, const struct sockaddr \*servaddr, socklen\_t addrlen)**

三次握手过程

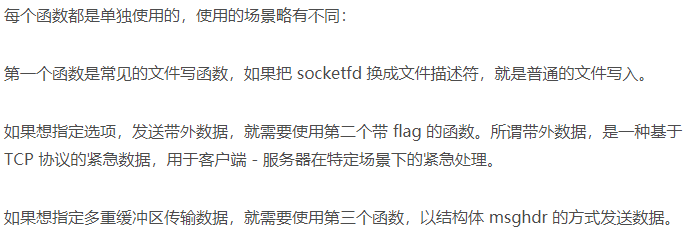


发送数据有三个函数

**1、ssize\_t write (int socketfd, const void \*buffer, size\_t size)**

**2、ssize\_t send (int socketfd, const void \*buffer, size\_t size, int flags)**

**3、ssize\_t sendmsg(int sockfd, const struct msghdr \*msg, int flags)**



发送缓冲区

系统内核会分配一个系统缓冲区，但是可能会遇到一些问题

