**什么是Socket?**

**Socket中文翻译过来就是“套接字”，也就是插座。**

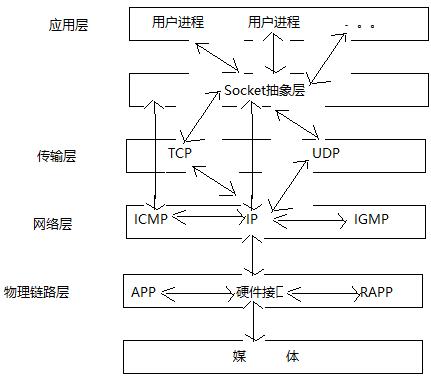
**Socket就像是一个电话插座，负责连接两端的电话，进行点对点通信，插座的两个孔就是端口，端口不能同时被其他进程占用。**

**而我们建立连接就像把插头插在这个插座上，创建一个Socket实例开始监听后，这个电话插座就时刻监听着消息的传入，谁拨通我这个IP地址和端口，我就连接谁**

**实际上Socket是在应用层和传输层之间的一个抽象层，他把TCP/IP层复杂的操作抽象为几个简单的接口，供应用层调用实现进程在网络中的通信，**

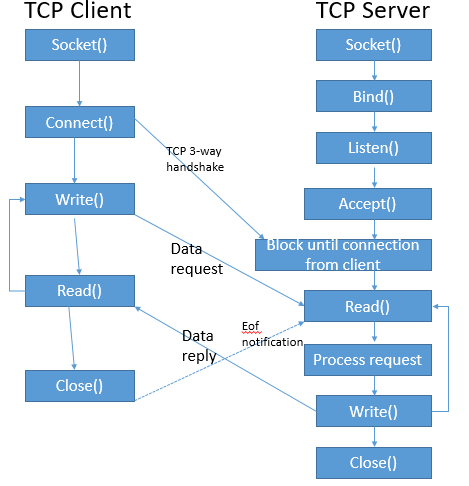
**Socket源于UNIX。在UNIX一切皆为文件的思想下，进程间通信就被称为文件描述符(file descriptor)，Socket是一种打开🡪读/写🡪关闭模式的实现，服务器和客户端各自维护一个“文件”，在建立连接打开后，可以向文件写入内容供对方读取或者读取对方内容，通讯结束是关闭文件。**

**下图就是Socket抽象层的位置**



**Socket通信过程**

**Socket保证了不同计算机之间的通信，也就是网络通信，对于网站，通信模型是服务器与客户端之间的通信。两端都建立了Socket对象，然后通过Socket对象对数据进行传输。通常服务器处于一个无限循环，等待客户端的连接**



**客户端过程**

**从上图可以看出，客户端的过程比较简单，创建Socket，连接服务器，将Socket与远程主机连接（注意：只有TCP才有连接的概念，一些UDP、ICMP和ARP没有连接的概念），发送数据，读取响应数据，直到数据交换完毕，关闭连接，结束TCP对话。**

**注意：sendall()在返回之前会尝试发送所有数据，并且成功时会返回None;而send()则返回发送的字节数，失败时都会抛出异常**

**服务端过程**

**从上图不难看出，服务端先初始化Scoket，建立流式套接字，与本机地址及端口进行绑定，然后通知TCP，准备好接收连接，调用accept()阻塞，等待来自客户端的连接，建立连接之后，客户端发送数据请求，服务端接收请求并处理请求，然后把响应数据发送给客户端，客户端读取数据，，直到数据读取完毕。最后关闭裂解，交互结束。**

**注意：1、 调用accept()时，Socket会进入waiting状态。客户端请求连接时，方法建立连接并返回服务器。Accept()返回一个含有两个元素的元组（conn,addr）。**

**第一个元素conn是新的Scoket对象，服务器必须通过它与客户端通信，第二个元素addr是客户端的IP地址和端口**

**2、 接下来就是处理阶段，服务器和客户端通过send()和recv()通信(传输数据).服务器调用send()，并采用字符串形式向客户端发送信息，sned()返回已**

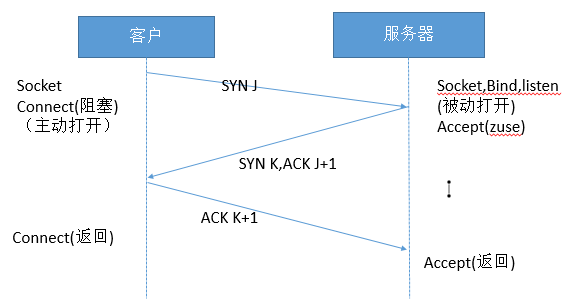
**发送的字符个数。**

**3、 服务器调用recv()从客户端接收消息，调用recv()必须指定一个整数，它对应于可通过本次方法调用来接收的最大数据量。Recv()在接收数据时会进入**

**blocked状态，最后返回一个字符串，用它表示收到的数据。如果发送的数据量超过了recv()所允许的，数据就会被截短，多余的数据将缓冲与接收端，以后调用recv()的时候，会继续读剩下的字节，如果有多余的数据会从缓冲区删除(以及自上次调用recv()以来，客户端可能发送的其他数据)。传输结束，服务器调用Socket的close()关闭连接**

**从TCP连接的视角看Socket过程**

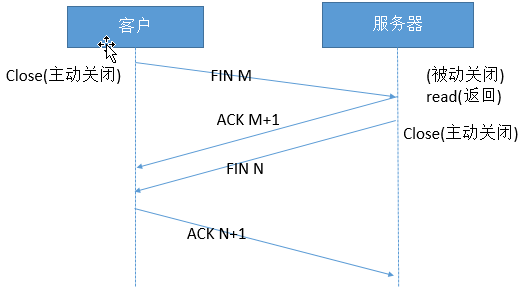
**TCP三次握手的Socket过程：**



1. **服务器调用socket() bind() listen()完成初始化后，调用accept()阻塞等待**
2. **客户端Socket对象调用connect()向服务器发送了SYN阻塞**
3. **服务器完成了第一次握手，即发送SYN和ACK应答**
4. **客户端收到服务器发送的应答之后，从connect()返回，在发送一个ACK给服务器**
5. **服务器Socket对象接收客户端第三次握手ACK确认，此时服务器送accept()返回，建立连接**

**接下来就是两个端的连接对象互相收发数据**

**TCP四次握手的Socket过程**



1. **某个应用进程调用close()主动关闭，发送一个FIN**
2. **另一端收到FIN后被动执行关闭，并发送ACK确认**
3. **之后被动执行关闭的应用进程调用close()关闭Socket，并也发送一个FIN**
4. **接收到这个FIN的一端向另一端ACK确认**

**TCP和UDP**

1. **TCP和UDP概念**
   1. **TCP是传输控制协议，提供的是面向连接、可靠的字节流服务**
   2. **UDP是用户数据协议，是一个简单的面向数据包的运输层协议**
2. **TCP UDP区别**
   1. **TCP面向连接的运输层协议，UDP无连接**
   2. **TCP是可靠交付，UDP尽最大努力交付**
   3. **TCP面向字节流，UDP一对一，一对多，多对多都可以**
   4. **TCP适用网页，邮件等，UDP适用于视频，语音广播**
3. **TCP UDP适用场景**
   1. **整个数据要准确无误的传递给对方，这往往用于一些要求可靠的应用，比如HTTP HTTPS FTP等传输文件的协议，POP SMTP等邮件传输的协议**
   2. **当网络通讯质量要求不高的时候，要求网络通讯速度能尽量的快，比如视频 广播等，这时就可以使用UDP**