socket

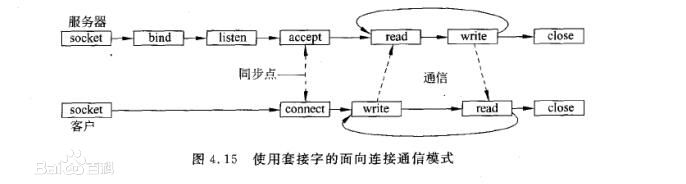
网络上的两个程序通过一个双向的通信连接实现数据的交换，这个连接的一端称为一个socket。

建立网络通信连接至少要一对端口号(socket)。socket本质是编程接口(API)，对TCP/IP的封装，TCP/IP也要提供可供程序员做网络开发所用的接口，这就是Socket编程接口；HTTP是轿车，提供了封装或者显示数据的具体形式；Socket是发动机，提供了网络通信的能力。

Socket的英文原义是“孔”或“插座”。作为BSD UNIX的[进程通信](https://baike.baidu.com/item/%E8%BF%9B%E7%A8%8B%E9%80%9A%E4%BF%A1)机制，取后一种意思。通常也称作"[套接字](https://baike.baidu.com/item/%E5%A5%97%E6%8E%A5%E5%AD%97)"，用于描述IP地址和端口，是一个通信链的句柄，可以用来实现不同虚拟机或不同计算机之间的通信。在Internet上的[主机](https://baike.baidu.com/item/%E4%B8%BB%E6%9C%BA)一般运行了多个服务软件，同时提供几种服务。每种服务都打开一个Socket，并绑定到一个端口上，不同的端口对应于不同的服务。Socket正如其英文原意那样，像一个多孔插座。一台主机犹如布满各种插座的房间，每个插座有一个编号，有的插座提供220伏交流电， 有的提供110伏交流电，有的则提供有线电视节目。 客户软件将插头插到不同编号的插座，就可以得到不同的服务。

Socket又称"套接字"，应用程序通常通过"套接字"向网络发出请求或者应答网络请求。

以J2SDK-1.3为例，Socket和ServerSocket类库位于java.net包中。ServerSocket用于服务器端，Socket是建立网络连接时使用的。在连接成功时，应用程序两端都会产生一个Socket实例，操作这个实例，完成所需的会话。对于一个网络连接来说，套接字是平等的，并没有差别，不因为在服务器端或在客户端而产生不同级别。不管是Socket还是ServerSocket它们的工作都是通过SocketImpl类及其子类完成的。



Socket非常类似于电话插座。以一个国家级电话网为例，电话的通话双方相当于相互通信的2个进程，区号是它的[网络地址](https://baike.baidu.com/item/%E7%BD%91%E7%BB%9C%E5%9C%B0%E5%9D%80)；区内一个单位的交换机相当于一台[主机](https://baike.baidu.com/item/%E4%B8%BB%E6%9C%BA)，主机分配给每个用户的局内号码相当于Socket号。任何用户在通话之前，首先要占有一部电话机，相当于申请一个Socket；同时要知道对方的号码，相当于对方有一个固定的Socket。然后向对方拨号呼叫，相当于发出连接请求（假如对方不在同一区内，还要拨对方区号，相当于给出网络地址）。假如对方在场并空闲（相当于通信的另一主机开机且可以接受连接请求），拿起电话话筒，双方就可以正式通话，相当于连接成功。双方通话的过程，是一方向电话机发出信号和对方从电话机接收信号的过程，相当于向Socket发送数据和从socket接收数据。通话结束后，一方挂起电话机相当于关闭Socket，撤消连接。

在电话系统中，一般用户只能感受到本地电话机和对方电话号码的存在，建立通话的过程，话音传输的过程以及整个电话系统的技术细节对他都是透明的，这也与Socket机制非常相似。Socket利用网间网通信设施实现[进程通信](https://baike.baidu.com/item/%E8%BF%9B%E7%A8%8B%E9%80%9A%E4%BF%A1)，但它对通信设施的细节毫不关心，只要通信设施能提供足够的通信能力，它就满足了。

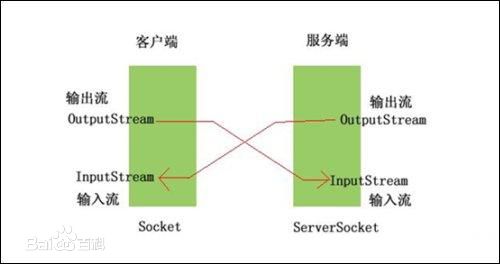
至此，我们对Socket进行了直观的描述。抽象出来，Socket实质上提供了进程通信的端点。进程通信之前，双方首先必须各自创建一个端点，否则是没有办法建立联系并相互通信的。正如打电话之前，双方必须各自拥有一台电话机一样。

在网间网内部，每一个Socket用一个半相关描述：（协议，本地地址，本地端口）。

一个完整的Socket有一个本地唯一的Socket号，由操作系统分配。

最重要的是，Socket是面向客户/[服务器](https://baike.baidu.com/item/%E6%9C%8D%E5%8A%A1%E5%99%A8)模型而设计的，针对客户和服务器程序提供不同的Socket[系统调用](https://baike.baidu.com/item/%E7%B3%BB%E7%BB%9F%E8%B0%83%E7%94%A8)。客户随机申请一个Socket（相当于一个想打电话的人可以在任何一台入网电话上拨号呼叫），系统为之分配一个Socket号；服务器拥有全局公认的Socket，任何客户都可以向它发出连接请求和信息请求（相当于一个被呼叫的电话拥有一个呼叫方知道的电话号码）。

Socket利用客户/服务器模式巧妙地解决了进程之间建立通信连接的问题。服务器Socket半相关为全局所公认非常重要。读者不妨考虑一下，两个完全随机的用户进程之间如何建立通信？假如通信双方没有任何一方的Socket固定，就好比打电话的双方彼此不知道对方的电话号码，要通话是不可能的。



连接过程

[编辑](javascript:;)

根据连接启动的方式以及本地[套接字](https://baike.baidu.com/item/%E5%A5%97%E6%8E%A5%E5%AD%97)要连接的目标，套接字之间的连接过程可以分为三个步骤：[服务器](https://baike.baidu.com/item/%E6%9C%8D%E5%8A%A1%E5%99%A8)监听，客户端请求，连接确认。

（1）服务器监听：是服务器端套接字并不定位具体的客户端套接字，而是处于等待连接的状态，实时监控网络状态。

（2）客户端请求：是指由客户端的套接字提出连接请求，要连接的目标是服务器端的套接字。为此，客户端的套接字必须首先描述它要连接的服务器的套接字，指出服务器端套接字的地址和[端口号](https://baike.baidu.com/item/%E7%AB%AF%E5%8F%A3%E5%8F%B7)，然后就向服务器端套接字提出连接请求。

（3）连接确认：是指当服务器端套接字监听到或者说接收到客户端套接字的连接请求，它就响应客户端套接字的请求，建立一个新的线程，把[服务器](https://baike.baidu.com/item/%E6%9C%8D%E5%8A%A1%E5%99%A8)端套接字的描述发给客户端，一旦客户端确认了此描述，连接就建立好了。而服务器端[套接字](https://baike.baidu.com/item/%E5%A5%97%E6%8E%A5%E5%AD%97)继续处于[监听状态](https://baike.baidu.com/item/%E7%9B%91%E5%90%AC%E7%8A%B6%E6%80%81)，继续接收其他客户端套接字的连接请求。

