1.计算机网络里面socket 通信常用的的函数

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,根据连接启动的方式以及本地套接字要连接的目标，套接字之间的连接过程可以分为三个步骤：服务器监听，客户端请求，连接确认。

（1）服务器监听：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：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，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。而服务器端套接字继续处于监听状态，继续接收其他客户端套接字的连接请求。

常用的函数：

1. 创建：socket()函数，这个操作类似于打开文件操作，返回socket的socket描述符。
2. 绑定： bind()函数，把一个地址族的特定地址指定给socket，而不是由系统随机分配.
3. listen()、connect()函数，客户端通过调用connect函数来建立与TCP服务器的连接。服务器端调用listen()，socket开始等待客户的链接请求
4. accept()函数，服务器收到请求后，用accept接受请求，然后链接就建立了，可以开始读写操作。
5. read()、write()读写操作
6. close()函数,读写完毕后要关闭相应的socket描述字

2.输入一个URL到看到结果，从输入网址到显示网页的过程，涉及了哪些协议。

- DNS解析-->建立连接
- 发送数据包 -->服务器响应请求
- 返回给浏览器-->浏览器渲染程序页面。

涉及协议：

- 应用层HTTP(www访问协议)，DNS（域名解析服务）
- 传输层TCP（为HTTP提供可靠的数据传输），UDP（DNS使用UDP传输）
- 网络层：IP（IP数据包传输和路由选择）、ICMP（提供网络传输过程中的差错检测）、ARP（将本机的默认网关IP地址映射成物理MAC地址）

1).DNS解析

当我搜索这个问题的时候，首先在浏览器输入了一个URL地址，但URL中服务器地址是一个域名而不是一个指定的IP地址，路由器并不知道你想要查找的地址，那么DNS域名解析系统会将该域名解析成ip，而IP地址是唯一的， 每一个ip地址对应网络上的一台计算机。

2).建立网络连接，发送数据包

由于1的努力，已经能够根据ip和端口号与网络上对应的服务器建立连接，浏览器这边会向服务器发送一个数据包，里面包含了大量的信息，但这个数据包有一定的格式。就像我给你邮个快递，也得遵循邮递公司的一些规则吧！我得写上我的身份信息、寄的物品、标明邮递地址....道理是一样的，到了网络中这些规则就是“Http协议(网络协议)”。

3).服务器响应请求，返回给浏览器

服务器会分解你的数据包，例如你查找的是一个文档，那么服务器可能会返回一个doc文档或者zip压缩资源给你；如果你访问的是一个链接页面，那么服务器相应的返回一个包含HTML/CSS标记文档，这些请求和响应都有一个通用的写法，这些规则也就是前面提到的"http协议"。客户端向服务器请求资源时，除了告诉服务器要请求的资源，同时还会附带一些其他的信息，这部分信息放在"header"部分（服务器响应请求也一样！），主要有请求头和响应头。

4).浏览器渲染呈现

浏览器拿到响应的页面代码，将其解析呈现在用户面前，至于为什么会是看到的这个样子，有时又是另外的一些页面效果，这里就涉及到web标准了，也就是我们经常提到的w3c标准。根据资源的类型，在网页上呈现给用户，这个过程

叫网页渲染。解析和呈现的过程主要由浏览器的渲染引擎实现，浏览器的渲染引擎质量就决定了浏览器的好坏

3.http协议

HTTP是一个属于应用层的面向对象的协议，**http协议（超文本传输协议）**是客户端和服务器端两者通信共同遵循的一些规则。主要内容是定义了客户端如何向服务器请求资源，服务器如何响应客户端请求。HTTP 协议一共有五大特点：
1、支持客户/服务器模式（C/S模式）；2、简单快速；3、灵活；4、无连接；5、无状态。

1).无连接

- 无连接的含义是限制每次连接只处理一个请求。服务器处理完客户的请求，并收到客户的应答后，即断开连接。采用这种方式可以节省传输时间。
- 早期这么做的原因是 HTTP 协议产生于互联网，因此服务器需要处理同时面向全世界数十万、上百万客户端的网页访问，但每个客户端（即浏览器）与服务器之间交换数据的间歇性较大（即传输具有突发性、瞬时性），并且网页浏览的联想性、发散性导致两次传送的数据关联性很低，大部分通道实际上会很空闲、无端占用资源。因此 HTTP 的设计者有意利用这种特点将协议设计为请求时建连接、请求完释放连接，以尽快将资源释放出来服务其他客户端。随着时间的推移，网页变得越来越复杂，里面可能嵌入了很多图片，这时候每次访问图片都需要建立一次 TCP 连接就显得很低效。后来，Keep-Alive 被提出用来解决这效率低的问题。
- Keep-Alive 功能使客户端到服务器端的连接持续有效（长连接），当出现对服务器的后继请求时，Keep-Alive 功能避免了建立或者重新建立连接。市场上的大部分 Web 服务器，包括 iPlanet、IIS 和 Apache，都支持 HTTP Keep-Alive。对于提供静态内容的网站来说，这个功能通常很有用。但是，对于负担较重的网站来说，这里存在另外一个问题：虽然为客户保留打开的连接有一定的好处，但它同样影响了性能，因为在处理暂停期间，本来可以释放的资源仍旧被占用。当Web服务器和应用服务器在同一台机器上运行时，Keep-Alive 功能对资源利用的影响尤其突出。 这样一来，客户端和服务器之间的 HTTP 连接就会被保持，不会断开（超过

Keep-Alive 规定的时间，意外断电等情况除外），当客户端发送另外一个请求时，就使用这条已经建立的连接。

2).无状态

- 无状态是指协议对于事务处理没有记忆能力，服务器不知道客户端是什么状态。即我们给服务器发送 HTTP 请求之后，服务器根据请求，会给我们发送数据过来，但是，发送完，不会记录任何信息。HTTP 是一个无状态协议，这意味着每个请求都是独立的，Keep-Alive 没能改变这个结果。缺少状态意味着如果后续处理需要前面的信息，则它必须重传，这样可能导致每次连接传送的数据量增大。另一方面，在服务器不需要先前信息时它的应答就较快。
- HTTP 协议这种特性有优点也有缺点，优点在于解放了服务器，每一次请求“点到为止”不会造成不必要连接占用，缺点在于每次请求会传输大量重复的内容信息。
- 客户端与服务器进行动态交互的 Web 应用程序出现之后，HTTP 无状态的特性严重阻碍了这些应用程序的实现，毕竟交互是需要承前启后的，简单的购物车程序也要知道用户到底在之前选择了什么商品。于是，两种用于保持 HTTP 连接状态的技术就应运而生了，一个是 Cookie，而另一个则是 Session。

Cookie可以保持登录信息到用户下次与服务器的会话，换句话说，下次访问同一网站时，用户会发现不必输入用户名和密码就已经登录了（当然，不排除用户手工删除Cookie）。而还有一些Cookie在用户退出会话的时候就被删除了，这样可以有效保护个人隐私。Cookies 最典型的应用是判定注册用户是否已经登录网站，用户可能会得到提示，是否在下一次进入此网站时保留用户信息以便简化登录手续，这些都是 Cookies 的功用。另一个重要应用场合是“购物车”之类处理。用户可能会在一段时间内在同一家网站的不同页面中选择不同的商品，这些信息都会写入 Cookies，以便在最后付款时提取信息。
与 Cookie 相对的一个解决方案是 Session，它是通过服务器来保持状态的。当客户端访问服务器时，服务器根据需求设置 Session，将会话信息保存在服务器上，同时将标示 Session 的 SessionId 传递给客户端浏览器，浏览器将这个

SessionId 保存在内存中，我们称之为无过期时间的 Cookie。浏览器关闭后，这个 Cookie 就会被清掉，它不会存在于用户的 Cookie 临时文件。以后浏览器每次请求都会额外加上这个参数值，服务器会根据这个 SessionId，就能取得客户端的数据信息。

如果客户端浏览器意外关闭，服务器保存的 Session 数据不是立即释放，此时数据还会存在，只要我们知道那个 SessionId，就可以继续通过请求获得此 Session 的信息，因为此时后台的 Session 还存在，当然我们可以设置一个 Session 超时时间，一旦超过规定时间没有客户端请求时，服务器就会清除对应 SessionId 的 Session 信息。

4.http响应状态码

状态码的职责是当客户端向服务器端发送请求时，描述返回的请求结果。借助于状态码，浏览器（或者说用户）可以知道服务器是正常的处理了请求，还是出现了错误。数字的第一位指定了响应类型，后两位无分类。HTTP响应状态码有很多，但是实际经常使用的大概只有14个，响应类别一共有5种：

- 1XX Informational(**信息性状态码)**
- 2XX Success(**成功状态码)**
- 3XX Redirection(**重定向状态码)**
- 4XX Client Error(**客户端错误状态码)**
- 5XX Server Error(**服务器错误状态码)**

1).2XX Success

- 200 OK 表示从客户端发来的请求在服务器端被正常处理了。
- 204 No Content 该状态码表示服务器接收的请求已成功处理，但在返回的响应报文中不含实体的主体部分。比如，当从浏览器发出请求处理后，返回204响应，那么浏览器显示的页面不发生更新。
- 206 Partial Content 该状态码表示客户端进行了范围请求，而服务器成功执行了这部分的GET请求。

2).3XX Redirection

- 301 Moved Permanently 永久性重定向。该状态码表示请求的资源已经被分配了新的URI，以后应使用资源现在所指的URI。 像下方给出的请求URI，当指定的资源路径的最后忘记添加斜杠"/"，就会产生301状态码
- 302 Found 临时性重定向。该状态码表示请求的资源已被分配了新的URI，希望用户(本次)能使用新的URI访问。
- 303 See Other 该状态码表示由于请求对应的资源存在另外一个URI，应使用GET方法定向获取请求的资源。 303状态码和302状态码有着相同的功能，但303状态码明确表明客户端应当采用GET方法获取资源。 当301，302，303响应状态码返回时，几乎所有的浏览器都会把POST改成GET，并删除请求报文的主体，之后请求会自动再次发送。 301，302标准是禁止将POST方法改变成GET方法的，但实际上使用时大家都会这么做。
- 304 Not Modified 该状态码表示客户端发送附带条件的请求时，服务器端允许请求访问资源，但未满足条件的情况。304状态码返回时，不包含任何响应的主体部分。304虽然被划分在3XX类别中，但是和重定向没有关系。
- 307 Temporary Redirect 临时重定向。该状态码与302 Found有着相同的含义。307会遵照浏览器标准，不会从POST变成GET。

3).4XX Client Error

- 400 Bad Request 该状态码表示请求报文中存在语法错误。当错误发生时，需要修改请求的内容后再次放松请求。
- 401 Unauthorized 该状态码表示发送的请求需要有通过HTTP认证的认证信息，另外若之前已进行过1此请求，则表示用户认证失败。
- 403 Forbidden 该状态码表明对请求资源的访问被服务器拒绝了。
- 404 Not Found 该状态码表明服务器上无法找到请求的资源。除此之外，也可以在服务器端拒绝请求且不想说明理由时使用。

4).5XX Server Error

- 500 Internal Server Error 该状态码表明服务器端在执行请求时发生了错误。
- 503 Service Unavailable 该状态码表明服务器暂时处于超负载或正在进行停机维护，现在无法处理请求

常见状态码:

- 100 Continue 继续，一般在发送post请求时，已发送了http、header之后服务端将返回此信息，表示确认，之后发送具体参数信息
- 200 OK 正常返回信息
- 201 Created 请求成功并且服务器创建了新的资源
- 301 Moved Permanently 请求的网页已永久移动到新位置。
- 400 Bad Request 服务器无法理解请求的格式，客户端不应当尝试再次使用相同的内容发起请求。
- 404 Not Found 找不到如何与 URI 相匹配的资源。
- 500 Internal Server Error 最常见的服务器端错误。

4.若有客户打不开网站，分析原因排错

网络出现故障从底层往高层一项项的逐步检查

- 物理层：查看连接状态，查看发送和接收的数据包的具体情况（网线没有接上（断开）、网线的水晶头该重新置换，没有接触良好）
- 数据链路层：MAC地址冲突、ADSL拨号上网欠费、网速没有办法协商一致、计算机连接到错误的VLAN
- 网络层：配置错误IP地址，子网掩码/配置错误的网关，路由器没有配置，到达不了目标网络的路由器
- 应用层：应用程序配置错误