HTTP协议
简介：超文本传输协议；是一种用于分布式、协作式和超媒体信息系统的应用层协议；

HTTP工作原理
步骤：

1. 连接服务器；建立TCP套接字连接
2. 发送HTTP请求；请求报文由请求行、请求头部、空行和请求数据组成。
3. 服务器接收并返回响应；响应由状态行、响应头部、空行和相应数据组成。
4. 释放TCP连接；若connection模式为close，则服务器自动关闭TCP连接。
5. 浏览器解析HTML内容；**首先解析状态行，查看请求是否成功。**然后解析每个响应头，响应头信息包含：响应内容字符集等响应数据的属性。

特点：基于请求-响应模式；无状态保留；无连接（一次只处理一个请求，但1.1版本后就会等待几秒并复用之前的通道）

HTTP无状态协议

1. 概念：指协议对于事务处理没有记忆能力，即不会保存用户之前的请求信息，必须要在本次连接中重传信息。
2. 无状态原因：浏览器和服务器都是使用socket套接字进行通信的，服务器将请求结果返回给浏览器后，会关闭当前socket连接，并会销毁页面对象。
3. 解决无状态的方式：1. 使用Cookie；2. 使用Session。

session和cookie
区别：

- 存储位置不同：cookie存在客户端浏览器上；session存在服务器上。
- 存储容量不同：单个cookie存<=4KB,一个站点最多存20个cookie;session没有限制，但要考虑服务器性能，会设置session删除机制。
- 存储方式不同：cookie只保管ASCII字符串，并需要通过编码方式存储为Unicode或二进制;session能存储任何类型数据；
- 隐私策略不同：cookie对客户端是可见的，可以分析存在本地的cookie进行欺骗，所以不安全；但session是存在服务器上的，对客户端时透明的。
- 有效期上不同：cookie可以通过设置，来达到长期有效的；**session依赖于名为JSESSIONID的cookie**，而cookie JSESSIONID的过期时间默认为-1，只需关闭窗口该session就会失效，因而session不能达到长期有效的效果。
- 服务器压力不同：cookie保存在客户端，不占用服务器资源；session是保存在服务器端，每个用户都会产生一个session。如果并发较多，会耗费大量内存。
- 浏览器支持不同。
- 跨域支持不同。

- session是依赖于cookie的，如果用户关闭了cookie，session也会失效；原因是：sessionID无法从客户端传递到服务端，也不能从服务端传到客户端。
  cookie:会在**首次访问某网站站点**时，请求头带上cookie信息，cookie以键值对形式存在，请求头中cookie属性是一个cookie键值对的**数组**。服务器通过响应将该网站cookie信息加入到cookie数组中，浏览器就拥有了这个cookie信息。在以后的请求过程中会把该cookie信息放在请求头。
  session:是一种用于验证、存储用户信息的手段，是为了辨识单一用户的多请求的手段。servlet容器使用Httpsession来创建连接客户端和服务端的一个会话（session），使不同请求之间能共享信息。
  具体过程如下：
  服务器通过请求中的cookie信息获取JSESSIONID的cookie，如果没有：则新创建一个session和cookie，并对该cookie赋值sessionID；如果有，则通过cookie信息的值，查找内存中对应的session，找到就使用该session，如果找不到就新建一个session并用同样的方式创建一个cookie。

HTTP请求方法（**八种**）：

1. GET；
2. HEAD;
3. POST;
4. PUT;
5. DELETE;
6. TRACE;
7. OPTIONS;
8. CONNECT;

POST 与 GET的区别

> 1. GET方式**通过URL来提交数据，数据可以通过URL看到**；POST方法，**数据放置在HTMLHEADER内提交,使用流的方式写数据**；
> 2. Get方法提交数据最多为**1024**，因为放在URL中提交，所以受URL长度限制；而POST没有限制。
> 3. 如上所述，GET提交数据不安全，会显示在地址栏中，而POST不会，较安全；
> 4. 安全的和幂等的。所谓安全的意味着**该操作用于获取信息而非修改信息**。幂等的意味着对**同一URL的多个请求应该返回同样的结果**。完整的定义并不像看起来那样严格。换句话说，GET 请求一般不应产生副作用。从根本上讲，其目标是当用户打开一个链接时，她可以确信从自身的角度来看没有改变资源。比如，新闻站点的头版不断更新。虽然第二次请求会返回不同的一批新闻，该操作仍然被认为是**安全的和幂等的**，因为它总是返回当前的新闻。反之亦然。POST 请求就不那么轻松了。POST 表示可能改变服务器上的资源的请求。仍然以新闻站点为例，读者对文章的注解应该通过 POST 请求实现，因为在注解提交之后站点已经不同了（比方说文章下面出现一条注解）。

状态码：

1. 1xx：代表请求已被服务器接收并正在处理；
2. 2xx：**成功**，请求已正常处理完毕；
3. 3xx：**重定向**，需要后续操作才能完成请求；
4. 4xx：**客户端错误**；
5. 5xx：**服务器错误**；
   常见状态代码：

> - 200:请求成功；
> - 400:客户端请求中有语法错误；
> - 401:请求未经授权；
> - 403:服务器拒绝提供服务；
> - 404:请求资源不存在，即输入了错误的URL；
> - 500:服务器发生了不可预期的错误；
> - 501:使用了服务器不支持的请求方法；
> - 502:网关错误
> - 503:服务器当前不能处理请求，但一段时间后可能恢复；
>   参考：https://www.cnblogs.com/lienen/p/10767687.html

HTTP和HTTPS的区别：
HTTP是不安全的，**明文传输**的超文本传输协议，而HTTPS是**结合了SSL和HTTP构建的可进行加密传输、身份认证的网络协议**，更安全。**HTTP接口是80，HTTPS接口是443**；**HTTP**工作于应用层，**HTTPS**工作于传输层；HTTP无需证书，而HTTPS需要认证证书。
URI和URL的区别：
URI统一资源标识符，用来唯一的标识一个资源。
URL统一资源定位器，是一种具体的URI。是INternet上用来描述资源信息的字符串。
URI可以是绝对的也可以是相对的，但是URL一定是绝对的。
URI不包含任何访问资源的方法，唯一作用是解析。URL可以打开一个到达资源的流。
URL是URI的子集。

网络加密算法