**[HTTP协议【详解】——经典面试题](https://www.cnblogs.com/sunny-sl/p/6529830.html)**

http请求由三部分组成，分别是：请求行、消息报头、请求正文

HTTP（超文本传输协议）是一个基于请求与响应模式的、无状态的、应用层的协议，常基于TCP的连接方式，HTTP1.1版本中给出一种持续连接的机制，绝大多数的Web开发，都是构建在HTTP协议之上的Web应用。

**1、常用的HTTP方法有哪些？  
GET：**用于请求访问已经被URI（统一资源标识符）识别的资源，可以通过URL传参给服务器。  
**POST：**用于传输信息给服务器，主要功能与GET方法类似，但一般推荐使用POST方式。  
**PUT：**传输文件，报文主体中包含文件内容，保存到对应URI位置。  
**HEAD：**获得报文首部，与GET方法类似，只是不返回报文主体，一般用于验证URI是否有效。  
**DELETE：**删除文件，与PUT方法相反，删除对应URI位置的文件。  
**OPTIONS：**查询相应URI支持的HTTP方法。  
  
  
**2、GET方法与POST方法的区别  
区别一：**  
get重点在从服务器上获取资源，post重点在向服务器发送数据；  
**区别二：**  
get传输数据是通过URL请求，以field（字段）= value的形式，置于URL后，并用"?"连接，多个请求数据间用"&"连接，如http://127.0.0.1/Test/login.action?name=admin&password=admin，这个过程用户是可见的；  
post传输数据通过Http的post机制，将字段与对应值封存在请求实体中发送给服务器，这个过程对用户是不可见的；  
**区别三：**  
Get传输的数据量小，因为受URL长度限制，但效率较高；  
Post可以传输大量数据，所以上传文件时只能用Post方式；  
**区别四：**  
get是不安全的，因为URL是可见的，可能会泄露私密信息，如密码等；  
post较get安全性较高；  
**区别五：**  
get方式只能支持ASCII字符，向服务器传的中文字符可能会乱码。  
post支持标准字符集，可以正确传递中文字符。  
  
  
**3、HTTP请求报文与响应报文格式**

**请求报文包含三部分：**  
a、请求行：包含请求方法、URI、HTTP版本信息  
b、请求首部字段  
c、请求内容实体  
**响应报文包含三部分：**  
a、状态行：包含HTTP版本、状态码、状态码的原因短语  
b、响应首部字段  
c、响应内容实体  
  
**4、常见的HTTP相应状态码**

**返回的状态**1xx：指示信息--表示请求已接收，继续处理  
2xx：成功--表示请求已被成功接收、理解、接受  
3xx：重定向--要完成请求必须进行更进一步的操作  
4xx：客户端错误--请求有语法错误或请求无法实现  
5xx：服务器端错误--服务器未能实现合法的请求

**200：**请求被正常处理  
**204：**请求被受理但没有资源可以返回  
**206：**客户端只是请求资源的一部分，服务器只对请求的部分资源执行GET方法，相应报文中通过Content-Range指定范围的资源。  
**301：**永久性重定向  
**302：**临时重定向  
**303：**与302状态码有相似功能，只是它希望客户端在请求一个URI的时候，能通过GET方法重定向到另一个URI上  
**304：**发送附带条件的请求时，条件不满足时返回，与重定向无关  
**307：**临时重定向，与302类似，只是强制要求使用POST方法  
**400：**请求报文语法有误，服务器无法识别  
**401：**请求需要认证  
**403：**请求的对应资源禁止被访问  
**404：**服务器无法找到对应资源  
**500：**服务器内部错误  
**503：**服务器正忙

**5、HTTP1.1版本新特性**  
a、默认持久连接节省通信量，只要客户端服务端任意一端没有明确提出断开TCP连接，就一直保持连接，可以发送多次HTTP请求

b、管线化，客户端可以同时发出多个HTTP请求，而不用一个个等待响应

c、断点续传原理

HTTP/1.1 相⽐ HTTP/1.0 性能上的改进：

使⽤ TCP ⻓连接的⽅式改善了 HTTP/1.0 短连接造成的性能开销。

⽀持管道（pipeline）⽹络传输，只要第⼀个请求发出去了，不必等其回来，就可以发第⼆个请求出

去，可以减少整体的响应时间。

但 HTTP/1.1 还是有性能瓶颈：

请求 / 响应头部（Header）未经压缩就发送，⾸部信息越多延迟越⼤。只能压缩 Body 的部分；

发送冗⻓的⾸部。每次互相发送相同的⾸部造成的浪费较多；

服务器是按请求的顺序响应的，如果服务器响应慢，会招致客户端⼀直请求不到数据，也就是队头

阻塞； 没有请求优先级控制； 请求只能从客户端开始，服务器只能被动响应。

那上⾯的 HTTP/1.1 的性能瓶颈，HTTP/2 做了什么优化？

HTTP/2 协议是基于 HTTPS 的，所以 HTTP/2 的安全性也是有保障的。

那 HTTP/2 相⽐ HTTP/1.1 性能上的改进：

1. 头部压缩

HTTP/2 会压缩头（Header）如果你同时发出多个请求，他们的头是⼀样的或是相似的，那么，协议会

帮你消除重复的部分。

这就是所谓的 HPACK 算法：在客户端和服务器同时维护⼀张头信息表，所有字段都会存⼊这个表，⽣

成⼀个索引号，以后就不发送同样字段了，只发送索引号，这样就提⾼速度了。

2. ⼆进制格式

HTTP/2 不再像 HTTP/1.1 ⾥的纯⽂本形式的报⽂，⽽是全⾯采⽤了⼆进制格式，头信息和数据体都是

⼆进制，并且统称为帧（frame）：头信息帧和数据帧。

3. 数据流

HTTP/2 的数据包不是按顺序发送的，同⼀个连接⾥⾯连续的数据包，可能属于不同的回应。因此，必

须要对数据包做标记，指出它属于哪个回应。

每个请求或回应的所有数据包，称为⼀个数据流（ Stream ）。每个数据流都标记着⼀个独⼀⽆⼆的编

号，其中规定客户端发出的数据流编号为奇数， 服务器发出的数据流编号为偶数

客户端还可以指定数据流的优先级。优先级⾼的请求，服务器就先响应该请求。

http3

Tcp->Udp QUIC（解决tcp队头阻塞问题）

**6、常见HTTP首部字段  
a、通用首部字段**（请求报文与响应报文都会使用的首部字段）  
Date：创建报文时间  
Connection：连接的管理  
Cache-Control：缓存的控制  
Transfer-Encoding：报文主体的传输编码方式  
**b、请求首部字段**（请求报文会使用的首部字段）  
Host：请求资源所在服务器  
Accept：可处理的媒体类型  
Accept-Charset：可接收的字符集  
Accept-Encoding：可接受的内容编码  
Accept-Language：可接受的自然语言  
**c、响应首部字段（**响应报文会使用的首部字段）  
Accept-Ranges：可接受的字节范围  
Location：令客户端重新定向到的URI  
Server：HTTP服务器的安装信息  
**d、实体首部字段**（请求报文与响应报文的的实体部分使用的首部字段）  
Allow：资源可支持的HTTP方法  
Content-Type：实体主类的类型  
Content-Encoding：实体主体适用的编码方式  
Content-Language：实体主体的自然语言  
Content-Length：实体主体的的字节数  
Content-Range：实体主体的位置范围，一般用于发出部分请求时使用  
  
  
**7、HTTP的缺点与HTTPS**  
a、通信使用明文不加密，内容可能被窃听  
b、不验证通信方身份，可能遭到伪装  
c、无法验证报文完整性，可能被篡改

HTTPS就是HTTP加上加密处理（一般是SSL安全通信线路）+认证+完整性保护

**8、HTTP优化**

利用负载均衡优化和加速HTTP应用

利用HTTP Cache来优化网站

HTTP1.1与HTTP2.0的区别

**HTTP2.0的多路复用和HTTP1.X中的长连接复用有什么区别？**

* HTTP/1.\* 一次请求-响应，建立一个连接，用完关闭；每一个请求都要建立一个连接；短链接
* HTTP/1.1 Pipeling解决方式为，若干个请求排队串行化单线程处理，后面的请求等待前面请求的返回才能获得执行机会，一旦有某请求超时等，后续请求只能被阻塞，毫无办法，也就是人们常说的线头阻塞；长连接
* HTTP/2多个请求可同时在一个连接上并行执行。某个请求任务耗时严重，不会影响到其它连接的正常执行；