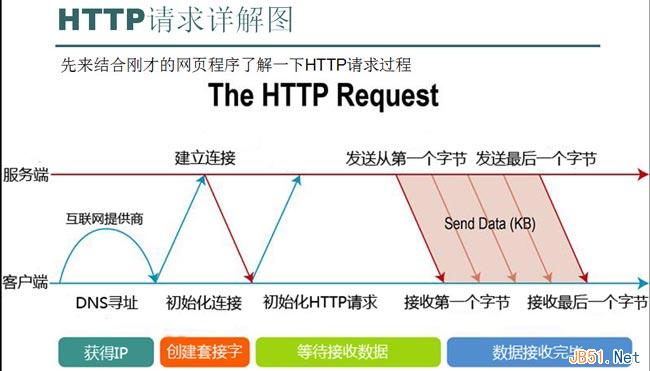
本文为多篇“HTTP请求头相关文章”及《HTTP权威指南》一书的阅读后个人汇总整理版，以便于理解。

　　通常HTTP消息包括客户机向服务器的请求消息和服务器向客户机的响应消息。客户端向服务器发送一个请求，请求头包含请求的方法、URI、协议版本、以及包含请求修饰符、客户信息和内容的类似于MIME的消息结构。服务器以一个状态行作为响应，相应的内容包括消息协议的版本，成功或者错误编码加上包含服务器信息、实体元信息以及可能的实体内容。

[[](http://files.jb51.net/file_images/article/201407/201477115759353.jpg?201467115831)](http://files.jb51.net/file_images/article/201407/201477115759353.jpg?201467115831)

Http协议定义了很多与服务器交互的方法，最基本的有4种，分别是GET、POST、PUT、DELETE。一个URL地址用于描述一个网络上的资源，而HTTP中的GET、POST、PUT、 DELETE就对应着对这个资源的查、改、增、删4个操作，我们最常见的就是GET和POST了。GET一般用于获取/查询资源信息，而POST一般用于更新资源信息。

**HTTP头信息解读**

　　HTTP的头域包括通用头、请求头、响应头和实体头四个部分。每个头域由一个域名，冒号（:）和域值三部分组成。

　　通用头部是客户端和服务器都可以使用的头部，可以在客户端、服务器和其他应用程序之间提供一些非常有用的通用功能，如Date头部。

　　请求头部是请求报文特有的，它们为服务器提供了一些额外信息，比如客户端希望接收什么类型的数据，如Accept头部。

　　响应头部便于客户端提供信息，比如，客服端在与哪种类型的服务器进行交互，如Server头部。

　　实体头部指的是用于应对实体主体部分的头部，比如，可以用实体头部来说明实体主体部分的数据类型，如Content-Type头部。

**HTTP通用头**

　　通用头域包含请求和响应消息都支持的头域，通用头域包含缓存头部Cache-Control、Pragma及信息性头部Connection、Date、Transfer-Encoding、Update、Via。

**1、Cache-Control**

　　Cache-Control指定请求和响应遵循的缓存机制。在请求消息或响应消息中设置 Cache-Control并不会修改另一个消息处理过程中的缓存处理过程。请求时的缓存指令包括no-cache、no-store、max-age、 max-stale、min-fresh、only-if-cached，响应消息中的指令包括public、private、no-cache、no- store、no-transform、must-revalidate、proxy-revalidate、max-age。各个消息中的指令含义如下：

　　no-cache：指示请求或响应消息不能缓存，实际上是可以存储在本地缓存区中的，只是在与原始服务器进行新鲜度验证之前，缓存不能将其提供给客户端使用。

　　no-store：缓存应该尽快从存储器中删除文档的所有痕迹，因为其中可能会包含敏感信息。

　　max-age：缓存无法返回缓存时间长于max-age规定秒的文档，若不超规定秒浏览器将不会发送对应的请求到服务器，数据由缓存直接返回；超过这一时间段才进一步由服务器决定是返回新数据还是仍由缓存提供。若同时还发送了max-stale指令，则使用期可能会超过其过期时间。

　　min-fresh：至少在未来规定秒内文档要保持新鲜，接受其新鲜生命期大于其当前 Age 跟 min-fresh 值之和的缓存对象。

　　max-stale：指示客户端可以接收过期响应消息，如果指定max-stale消息的值，那么客户端可以接收过期但在指定值之内的响应消息。

　　only-if-cached：只有当缓存中有副本存在时，客户端才会获得一份副本。

　　Public：指示响应可被任何缓存区缓存，可以用缓存内容回应任何用户。

　　Private：指示对于单个用户的整个或部分响应消息，不能被共享缓存处理，只能用缓存内容回应先前请求该内容的那个用户。

**2、Pragma**

　　Pragma头域用来包含实现特定的指令，最常用的是Pragma:no-cache。在HTTP/1.1协议中，它的含义和Cache- Control:no-cache相同。

**3、Connection**

　　Connection表示是否需要持久连接。如果Servlet看到这里的值为“Keep-Alive”，或者看到请求使用的是HTTP 1.1（HTTP 1.1默认进行持久连接），它就可以利用持久连接的优点，当页面包含多个元素时（例如Applet，图片），显著地减少下载所需要的时间。要实现这一点，Servlet需要在应答中发送一个Content-Length头，最简单的实现方法是：先把内容写入ByteArrayOutputStream，然后在正式写出内容之前计算它的大小。

　　Close：告诉WEB服务器或者代理服务器，在完成本次请求的响应后，断开连接，不要等待本次连接的后续请求了。

　　Keepalive：告诉WEB服务器或者代理服务器，在完成本次请求的响应后，保持连接，等待本次连接的后续请求。

　　Keep-Alive：如果浏览器请求保持连接，则该头部表明希望 WEB 服务器保持连接多长时间（秒），如Keep-Alive：300。

**4、Date**

　　Date头域表示消息发送的时间，服务器响应中要包含这个头部，因为缓存在评估响应的新鲜度时要用到，其时间的描述格式由RFC822定义。例如，Date:Mon, 31 Dec 2001 04:25:57 GMT。Date描述的时间表示世界标准时，换算成本地时间，需要知道用户所在的时区。

**5、Transfer-Encoding**

　　WEB 服务器表明自己对本响应消息体（不是消息体里面的对象）作了怎样的编码，比如是否分块（chunked），例如：Transfer-Encoding: chunked

**6、Upgrade**

　　它可以指定另一种可能完全不同的协议，如HTTP/1.1客户端可以向服务器发送一条HTTP/1.0请求，其中包含值为“HTTP/1.1”的Update头部，这样客户端就可以测试一下服务器是否也使用HTTP/1.1了。

**7、Via**

　　列出从客户端到 OCS 或者相反方向的响应经过了哪些代理服务器，他们用什么协议（和版本）发送的请求。

　　当客户端请求到达第一个代理服务器时，该服务器会在自己发出的请求里面添加 Via 头部，并填上自己的相关信息，当下一个代理服务器 收到第一个代理服务器的请求时，会在自己发出的请求里面复制前一个代理服务器的请求的Via头部，并把自己的相关信息加到后面，以此类推，当 OCS 收到最后一个代理服务器的请求时，检查 Via 头部，就知道该请求所经过的路由。例如：Via：1.0 236-81.D07071953.sina.com.cn:80 (squid/2.6.STABLE13)

**HTTP请求头**

　　请求头用于说明是谁或什么在发送请求、请求源于何处，或者客户端的喜好及能力。服务器可以根据请求头部给出的客户端信息，试着为客户端提供更好的响应。请求头域可能包含下列字段Accept、Accept-Charset、Accept- Encoding、Accept-Language、Authorization、From、Host、If-Modified-Since、If-Match、If-None-Match、If-Range、If-Range、If-Unmodified-Since、Max-Forwards、Proxy-Authorization、Range、Referer、User-Agent。对请求头域的扩展要求通讯双方都支持，如果存在不支持的请求头域，一般将会作为实体头域处理。

**8、Accept**

　　告诉WEB服务器自己接受什么介质类型，\*/\* 表示任何类型，type/\* 表示该类型下的所有子类型，type/sub-type。

**9、Accept-Charset**

　　浏览器告诉服务器自己能接收的字符集。

**10、Accept-Encoding**

　　浏览器申明自己接收的编码方法，通常指定压缩方法，是否支持压缩，支持什么压缩方法（gzip，deflate）。

**11、Accept-Language**

　　浏览器申明自己接收的语言。语言跟字符集的区别：中文是语言，中文有多种字符集，比如big5，gb2312，gbk等等。

**12、Authorization**

　　当客户端接收到来自WEB服务器的 WWW-Authenticate 响应时，用该头部来回应自己的身份验证信息给WEB服务器。

**13、If-Match**

　　如果对象的 ETag 没有改变，其实也就意味著对象没有改变，才执行请求的动作，获取文档。

**14、If-None-Match**

　　如果对象的 ETag 改变了，其实也就意味著对象也改变了，才执行请求的动作，获取文档。

**15、If-Modified-Since**

　　如果请求的对象在该头部指定的时间之后修改了，才执行请求的动作（比如返回对象），否则返回代码304，告诉浏览器该对象没有修改。例如：If-Modified-Since：Thu, 10 Apr 2008 09:14:42 GMT

**16、If-Unmodified-Since**

　　如果请求的对象在该头部指定的时间之后没修改过，才执行请求的动作（比如返回对象）。

**17、If-Range**

　　浏览器告诉 WEB 服务器，如果我请求的对象没有改变，就把我缺少的部分给我，如果对象改变了，就把整个对象给我。浏览器通过发送请求对象的ETag 或者自己所知道的最后修改时间给 WEB 服务器，让其判断对象是否改变了。总是跟 Range 头部一起使用。

**18、Range**

　　浏览器（比如 Flashget 多线程下载时）告诉 WEB 服务器自己想取对象的哪部分。例如：Range: bytes=1173546

**19、Proxy-Authenticate**

　　代理服务器响应浏览器，要求其提供代理身份验证信息。

**20、Proxy-Authorization**

　　浏览器响应代理服务器的身份验证请求，提供自己的身份信息。

**21、Host**

　　客户端指定自己想访问的WEB服务器的域名/IP 地址和端口号。如Host：rss.sina.com.cn

**22、Referer**

　　浏览器向WEB 服务器表明自己是从哪个网页URL获得点击当前请求中的网址/URL，例如：Referer：[http://www.jb51.net](http://www.jb51.net/)

**23、User-Agent**

　　浏览器表明自己的身份（是哪种浏览器）。例如：User-Agent：Mozilla/5.0 (Windows; U; Windows NT 5.1; zh-CN;rv:1.8.1.14) Gecko/20080404 Firefox/2.0.0.14

**HTTP响应头**

　　响应头向客户端提供一些额外信息，比如谁在发送响应、响应者的功能，甚至与响应相关的一些特殊指令。这些头部有助于客户端处理响应，并在将来发起更好的请求。响应头域包含Age、Location、Proxy-Authenticate、Public、Retry- After、Server、Vary、Warning、WWW-Authenticate。对响应头域的扩展要求通讯双方都支持，如果存在不支持的响应头域，一般将会作为实体头域处理。

**24、Age**

　　当代理服务器用自己缓存的实体去响应请求时，用该头部表明该实体从产生到现在经过多长时间了。

**25、Server**

　　WEB 服务器表明自己是什么软件及版本等信息。例如：Server：Apache/2.0.61 (Unix)

**26、Accept-Ranges**

　　WEB服务器表明自己是否接受获取其某个实体的一部分（比如文件的一部分）的请求。bytes：表示接受，none：表示不接受。

**27、Vary**

　　WEB服务器用该头部的内容告诉 Cache 服务器，在什么条件下才能用本响应所返回的对象响应后续的请求。假如源WEB服务器在接到第一个请求消息时，其响应消息的头部为：Content-Encoding: gzip; Vary: Content-Encoding，那么Cache服务器会分析后续请求消息的头部，检查其Accept-Encoding，是否跟先前响应的Vary头部值一致，即是否使用相同的内容编码方法，这样就可以防止Cache服务器用自己Cache 里面压缩后的实体响应给不具备解压能力的浏览器。例如：Vary：Accept-Encoding。

**HTTP实体头**

　　实体头部提供了有关实体及其内容的大量信息，从有关对象类型的信息，到能够对资源使用的各种有效的请求方法。总之，实体头部可以告知接收者它在对什么进行处理。请求消息和响应消息都可以包含实体信息，实体信息一般由实体头域和实体组成。实体头域包含关于实体的原信息，实体头包括信息性头部Allow、Location，内容头部Content-Base、Content-Encoding、Content-Language、Content-Length、Content-Location、Content-MD5、Content-Range、Content-Type，缓存头部Etag、Expires、Last-Modified、extension-header。

**28、Allow**

　　服务器支持哪些请求方法（如GET、POST等）。

**29、Location**

　　表示客户应当到哪里去提取文档，用于将接收端定位到资源的位置（URL）上。Location通常不是直接设置的，而是通过HttpServletResponse的sendRedirect方法，该方法同时设置状态代码为302。

**30、Content-Base**

　　解析主体中的相对URL时使用的基础URL。

**31、Content-Encoding**

　　WEB服务器表明自己使用了什么压缩方法（gzip，deflate）压缩响应中的对象。例如：Content-Encoding：gzip

**32、Content-Language**

　　WEB 服务器告诉浏览器理解主体时最适宜使用的自然语言。

**33、Content-Length**

　　WEB服务器告诉浏览器自己响应的对象的长度或尺寸，例如：Content-Length: 26012

**34、Content-Location**

　　资源实际所处的位置。

**35、Content-MD5**

　　主体的MD5校验和。

**36、Content-Range**

　　实体头用于指定整个实体中的一部分的插入位置，他也指示了整个实体的长度。在服务器向客户返回一个部分响应，它必须描述响应覆盖的范围和整个实体长度。一般格式： Content-Range:bytes-unitSPfirst-byte-pos-last-byte-pos/entity-legth。例如，传送头500个字节次字段的形式：Content-Range:bytes0- 499/1234如果一个http消息包含此节（例如，对范围请求的响应或对一系列范围的重叠请求），Content-Range表示传送的范围，Content-Length表示实际传送的字节数。

**37、Content-Type**

　　WEB 服务器告诉浏览器自己响应的对象的类型。例如：Content-Type：application/xml

**38、Etag**

　　就是一个对象（比如URL）的标志值，就一个对象而言，比如一个html文件，如果被修改了，其Etag也会别修改，所以，ETag的作用跟Last-Modified的作用差不多，主要供WEB服务器判断一个对象是否改变了。比如前一次请求某个html文件时，获得了其 ETag，当这次又请求这个文件时，浏览器就会把先前获得ETag值发送给WEB服务器，然后WEB服务器会把这个ETag跟该文件的当前ETag进行对比，然后就知道这个文件有没有改变了。

**39、Expires**

　　WEB服务器表明该实体将在什么时候过期，对于过期了的对象，只有在跟WEB服务器验证了其有效性后，才能用来响应客户请求。是 HTTP/1.0 的头部。例如：Expires：Sat, 23 May 2009 10:02:12 GMT

**40、Last-Modified**

　　WEB服务器认为对象的最后修改时间，比如文件的最后修改时间，动态页面的最后产生时间等等。例如：Last-Modified：Tue, 06 May 2008 02:42:43 GMT