GET

     GET方法意思是获取被请求URI（Request-URI）指定的信息（以实体的格式）。如果请求  
URI涉及到一个数据生成过程，那么这个过程生成的数据应该被作为实体在响应中返回而不是  
过程的源文本，除非源文本恰好是过程的输出。  
     如果请求消息包含 If-Modified-Since,，If-Unmodified-Since，If-Match，If-None-Match 或者  
If-Range头域，GET的语义将变成“条件（conditionall） GET”。一个条件GET方法会请求满  
足条件头域的实体。条件GET方法的目的是为了减少不必要的网络使用，这通过允许利用缓存  
里仍然保鲜的实体而不用多次请求或传输客户端已经拥有的实体来实现的。.  
如果请求方法包含一个Range头域，那么GET方法就变成“部分Get”（partial GET）方法。  
一个部分GET会请求实体的一部分，这在14.35节里描述了。 部分GET方法的目的是为了减  
少不必要的网络使用，可以允许客户端从服务器获取实体的部分数据，而不需要获取客户端本  
地已经拥有的部分实体数据。  
GET请求的响应是可缓存的（cacheable）如果此响应满足第13节HTTP缓存的要求。  
看15.1.3节关于GET请求用于表单时安全考虑。

HEAD

HEAD 方法和GET 方法一致，除了服务器不能在响应里返回消息主体。HEAD请求响应里  
HTTP头域里的元信息（译注：元信息就是头域信息）应该和GET请求响应里的元信息一致。  
此方法被用来获取请求实体的元信息而不需要传输实体主体（entity-body）。此方法经常被用  
来测试超文本链接的有效性，可访问性，和最近的改变。.  
HEAD请求的响应是可缓存的，因为响应里的信息可能被缓存用于更新以前那个资源对应缓存  
的实体.。如果出现一个新的域值指明缓存的实体和当前源服务器上的实体有所不同（可能因为  
Content-Length，Content-MD5，ETag或Last-Modified值的改变），那么缓存（cache）必  
须认为缓存项是过时的（stale）。

POST

POST 方法被用于请求源服务器接受请求中的实体作为请求资源的一个新的从属物。POST被  
设计涵盖下面的功能。  
--已存在的资源的注释；  
--发布消息给一个布告板，新闻组，邮件列表，或者相似的文章组。  
--提供一个数据块，如提交一个表单给一个数据处理过程。  
--通过追加操作来扩展数据库。  
POST方法的实际功能是由服务器决定的，并且经常依赖于请求URI（Request-URI）。POST  
提交的实体是请求URI的从属物，就好像一个文件从属于一个目录，一篇新闻文章从属于一个  
新闻组，或者一条记录从属于一个数据库。  
POST方法执行的动作可能不会对请求URI所指的资源起作用。在这种情况下，200（成功）或  
者204（没有内容）将是适合的响应状态，这依赖于响应是否包含一个描述结果的实体。  
如果资源被源服务器创建，响应应该是201（Created）并且包含一个实体，此实体描述了请  
求的状态。并且引用了这个新资源和一个Location头域（见14.30节）。  
POST方法的响应是不可缓存的。除非响应里有合适的Cache-Control或者Expires头域。然而，  
303（见其他）响应能被用户代理利用去获得可缓存的响应。  
POST 请求必须遵循8.2节里指明的消息传送的要求。  
参见15.1.3节关于安全性的考虑.

PUT

PUT方法请求服务器去把请求里的实体存储在请求URI（Request-URI）标识下。如果请求  
URI（Request-URI）指定的的资源已经在源服务器上存在，那么此请求里的实体应该被当作  
是源服务器关于此URI所指定资源实体的最新修改版本。如果请求URI（Request-URI）指定  
的资源不存在，并且此URI被用户代理定义为一个新资源，那么源服务器就应该根据请求里的  
实体创建一个此URI所标识下的资源。如果一个新的资源被创建了，源服务器必须能向用户代  
理（user agent） 发送201（已创建）响应。如果已存在的资源被改变了，那么源服务器应该  
发送200（Ok）或者204（无内容）响应。如果资源不能根据请求URI创建或者改变，一个合  
适的错误响应应该给出以反应问题的性质。实体的接收者不能忽略任何它不理解和不能实现的  
Content-\*（如：Content-Range）头域，并且必须返回501（没有被实现）响应。  
如果请求穿过一个缓存（cache），并且此请求URI（Request-URI）指示了一个或多个当前  
缓存的实体，那么这些实体应该被看作是旧的。PUT方法的响应是不可缓存的。  
POST方法和PUT方法请求最根本的区别是请求URI（Request-URI）的含义不同。POST请  
求里的URI 指示一个能处理请求实体的资源（译注：此资源可能是一段程序，如jsp 里的  
servlet） 。此资源可能是一个数据接收过程，一个网关（gateway，译注：网关和代理的区别  
是：网关可以进行协议转换，而代理不能，只是起代理的作用，比如缓存服务器其实就是一个  
代理），或者一个单独接收注释的实体。对比而言，PUT方法请求里的URI标识请求里封装的  
实体一一用户代理知道URI 意指什么，并且服务器不能把此请求应用于其它资源  
（resource）。如果服务器期望请求被应用于一个不同的URI，那么它必须发送301（永久移  
动）响应；用户代理可以自己决定是否重定向请求。  
一个单独的资源可能会被许多不同的URI指定。如：一篇文章可能会有一个URI指定当前版本，  
而这个URI区别于这篇文章其它特殊版本的URI。这种情况下，对一个通用URI的PUT请求可  
能会导致其资源的其它URI请求被源服务器重定义。  
HTTP/1.1没有定义PUT方法对源服务器的状态影响。  
PUT请求必须遵循8.2节中的消息传送的要求。  
除非特别指出，PUT方法请求里的实体头域应该被用于资源的创建或修改。

DELETE（删除）

DELETE方法请求源服务器删除请求URI指定的资源。此方法可能会在源服务器上被人为的干  
涉（或通过其他方法）。客户端不能保证此操作能被执行，即使源服务器返回成功状态码。然而，  
服务器不应该指明成功除非它打算删除资源或把此资源移到一个不可访问的位置。  
如果响应里包含描述成功的实体，响应应该是200（OK）；如果DELETE动作还没有执行，  
应该以202（已接受）响应；如果DELETE请求方法已经执行但响应不包含实体，那么应该以  
204（无内容）响应。  
如果请求穿过缓存，并且请求URI（Request-URI）指定了一个或多个缓存当前实体，那么这  
些缓存项应该被认为是旧的。DELETE方法的响应是不能被缓存的。

TRACE

TRACE方法被用于激发一个远程的，应用层的请求消息回路（译注：TRACE方法让客户端测  
试到服务器的网络通路，回路的意思如发送一个请返回一个响应，这就是一个请求响应回路，）。

最后的接收者也许是源服务器，也许是接收到包含Max-Forwards头域值为0请求的代理  
或网关。TRACE请求不能包含一个实体。  
TRACE方法允许客户端去了解数据被请求链的另一端接收的情况，并且利用那些数据信息去  
测试或诊断。Via头域值（见14.45）有特殊的用途，因为它可以作为请求链的跟踪信息。利用  
Max-Forwards头域允许客户端限制请求链的长度，这是非常有用的，因为可以利用此去测试代  
理链在无限循环里转发消息。  
如果请求是有效的，响应应该在实体主体里包含整个请求消息，并且响应应该包含一个  
Content-Type头域值为”message/http”的头域。此方法的响应不能被缓存。

CONNECT（连接）

HTTP1.1 协议规范保留了CONNECT方法，此方法是为了能用于能动态切换到隧道的代理