# #第九周#：web 页面解析的流程学习

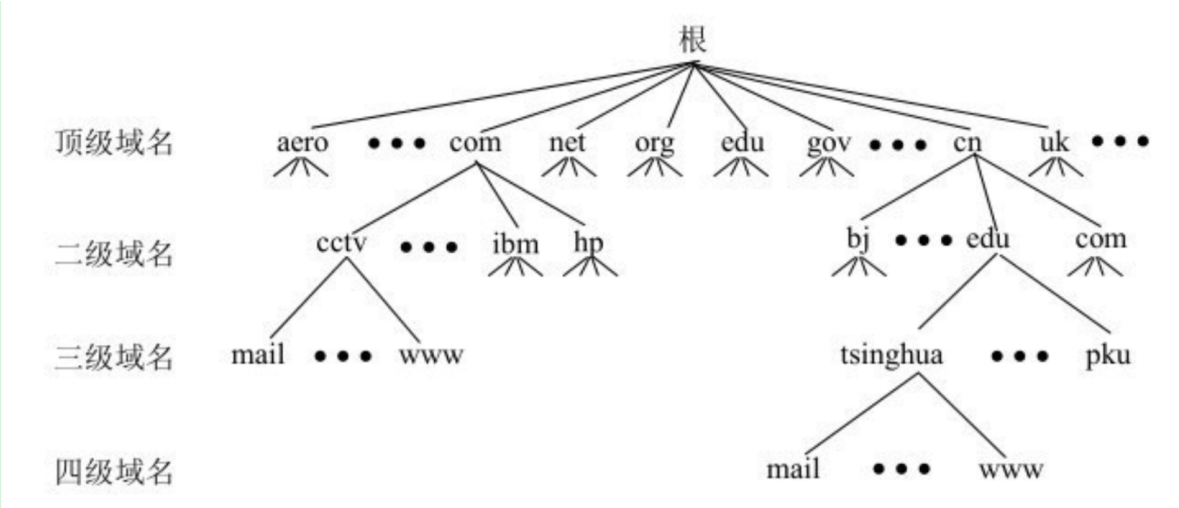
*1、理解域名解析的整个过程*
*2、理解 web 页面请求的整个流程，绘制流程图（nginx 处理的 11 个过程）*
*3、学习 http 协议中的字段及含义*
*4、学习 http 请求方法以及返回状态码的类型和含义*
*扩展学习：思考这个过程中都会涉及哪些安全问题，总结总结*

## 域名解析概念

域名解析也叫域名指向、服务器设置、域名配置以及反向IP登记等等。说得简单点就是将好记的域名解析成IP，服务由DNS服务器完成，是把域名解析到一个IP地址，然后在此IP地址的主机上将一个子目录与域名绑定，域名解析的作用主要就是为了便于记忆。

## 因特网的域名结构

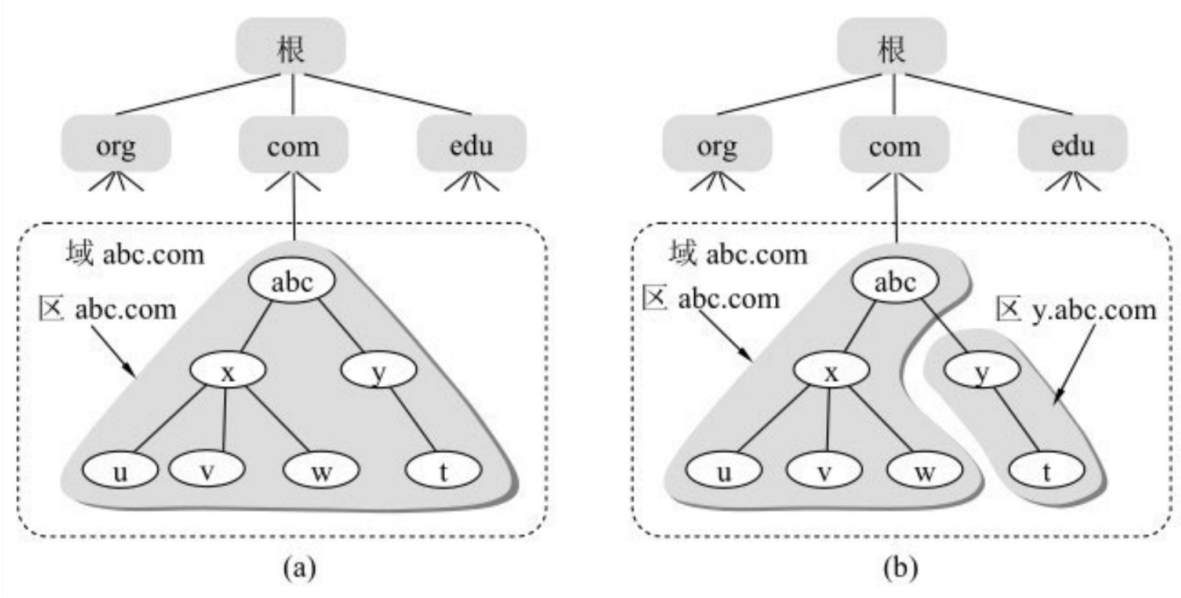
任何一个连接在因特网上的主机或路由器，都有一个唯一的层次结构的名字，即域名(domain name)。这里，“域”(domain)是名字空间中一个可被管理的划分。域还可以划分为子域，而子域还可继续划分为子域的子域，这样就形成了顶级域、二级域、三级域，等等。




上面讲述的域名体系是抽象的。但具体实现域名系统则是使用分布在各地的域名服务器。从理论上讲，可以让每一级的域名都有一个相对应的域名服务器，使所有的域名服务器构成和上图相对应的“域名服务器树”的结构。但这样做会使域名服务器的数量太多，使域名系统的运行效率降低。因此DNS就采用划分区的办法来解决这个问题。

一个服务器所负责管辖的（或有权限的）范围叫做区(zone)。各单位根据具体情况来划分自己管辖范围的区。但在一个区中的所有节点必须是能够连通的。每一个区设置相应的**权限域名服务器**(authoritative name server)，用来保存该区中的所有主机的域名到IP地址的映射。总之，DNS服务器的管辖范围不是以“域”为单位，而是以“区”为单位。区是DNS服务器实际管辖的范围。区可能等于或小于域，但一定不可能大于域。

下图是区的不同划分方法的举例。假定abc公司有下属部门x和y，部门x下面又分三个分部门u，v和w，而y下面还有其下属部门t。图(a)表示abc公司只设一个区abc.com。这时，区abc.com和域abc.com指的是同一件事。但图(b)表示abc公司划分了两个区（大的公司可能要划分多个区）：abc.com和y.abc.com。这两个区都隶属于域abc.com，都各设置了相应的权限域名服务器。不难看出，区是“域”的子集。




# 域名服务器类型

(1) **根域名服务器**(root name server)：根域名服务器是最高层次的域名服务器，也是最重要的域名服务器。所有的根域名服务器都知道所有的顶级域名服务器的域名和IP地址。

(2) **顶级域名服务器**（即TLD 服务器）：这些域名服务器负责管理在该顶级域名服务器注册的所有二级域名。当收到DNS查询请求时，就给出相应的回答（可能是最后的结果，也可能是下一步应当找的域名服务器的IP地址）。

(3) **权限域名服务器**：这就是前面已经讲过的负责一个区的域名服务器。当一个权限域名服务器还不能给出最后的查询回答时，就会告诉发出查询请求的DNS客户，下一步应当找哪一个权限域名服务器。

(4) **本地域名服务器**(local name server)：本地域名服务器并不属于域名服务器层次结构，但它对域名系统非常重要。当一个主机发出DNS查询请求时，这个查询请求报文就发送给本地域名服务器。

为了提高域名服务器的可靠性，DNS域名服务器都把数据复制到几个域名服务器来保存，其中的一个是**主域名服务器**(master name server)，其他的是**辅助域名服务器**(secondary name server)。当主域名服务器出故障时，辅助域名服务器可以保证DNS的查询工作不会中断。主域名服务器定期把数据复制到辅助域名服务器中，而更改数据只能在主域名服务器中进行。这样就保证了数据的一致性。

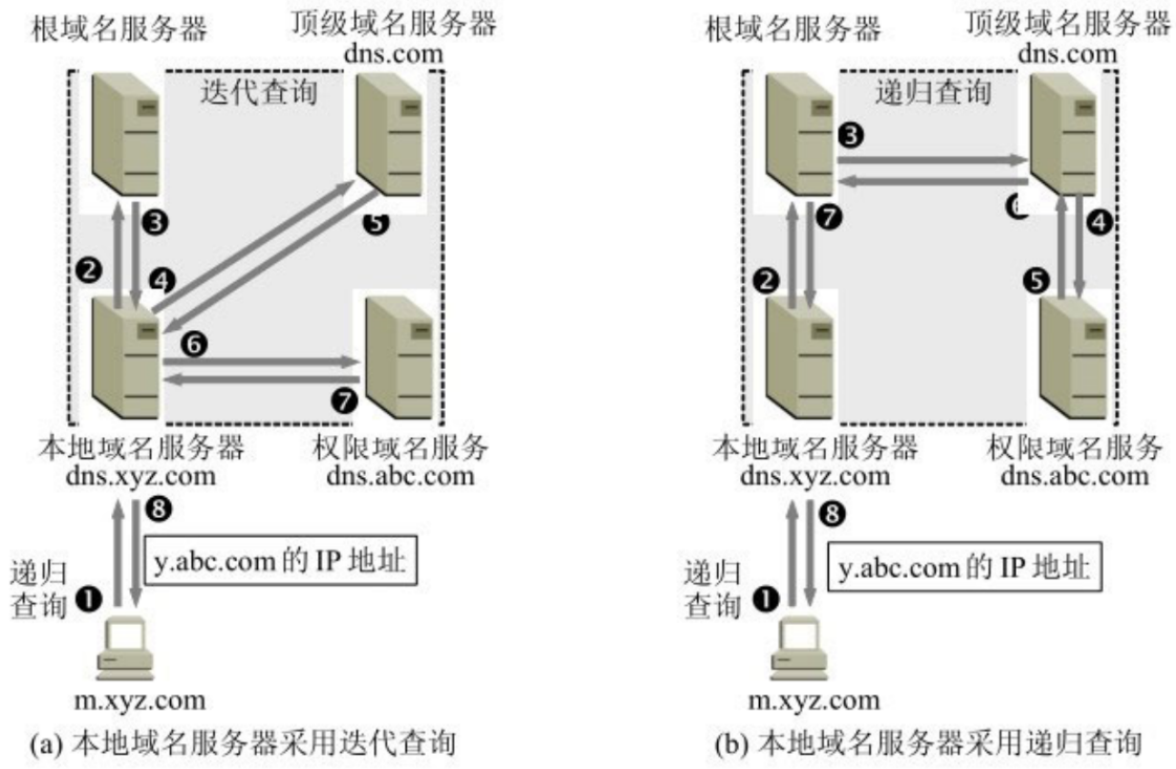
# 域名解析的过程

**一、递归查询**

主机向本地域名服务器的查询一般都是采用**递归查询**(recursive query)。所谓递归查询就是：如果主机所询问的本地域名服务器不知道被查询域名的IP地址，那么本地域名服务器就以**DNS客户**的身份，向其他根域名服务器继续发出查询请求报文（即**替该主机**继续查询），**而不是**让该主机自己进行下一步的查询。因此，递归查询返回的查询结果或者是所要查询的IP地址，或者是报错，表示无法查询到所需的IP地址。

## 二、迭代查询

本地域名服务器向根域名服务器的查询通常是采用**迭代查询**(iterative query)。迭代查询的特点是这样的：当根域名服务器收到本地域名服务器发出的迭代查询请求报文时，要么给出所要查询的IP地址，要么告诉本地域名服务器：“你下一步应当向哪一个域名服务器进行查询”。然后让本地域名服务器进行后续的查询（**而不是替**本地域名服务器进行后续的查询）。根域名服务器通常是把自己知道的顶级域名服务器的IP地址告诉本地域名服务器，让本地域名服务器再向顶级域名服务器查询。顶级域名服务器在收到本地域名服务器的查询请求后，要么给出所要查询的IP地址，要么告诉本地域名服务器下一步应当向哪一个权限域名服务器进行查询，本地域名服务器就这样进行迭代查询。最后，知道了所要解析的域名的IP地址，然后把这个结果返回给发起查询的主机。当然，本地域名服务器也可以采用递归查询，这取决于最初的查询请求报文的设置是要求使用哪一种查询方式。




假定域名为m.xyz.com的主机想知道另一个主机（域名为y.abc.com）的IP地址。例如，主机m.xyz.com打算发送邮件给主机y.abc.com。这时就必须知道主机y.abc.com的IP地址。下面是图(a)的几个查询步骤：

❶ 主机m.xyz.com先向其本地域名服务器dns.xyz.com进行递归查询。
❷ 本地域名服务器采用迭代查询。它先向一个根域名服务器查询。
❸ 根域名服务器告诉本地域名服务器，下一次应查询的顶级域名服务器dns.com的IP地址。
❹ 本地域名服务器向顶级域名服务器dns.com进行查询。
❺顶级域名服务器dns.com告诉本地域名服务器，下一次应查询的权限域名服务器dns.abc.com的IP地址。
❻ 本地域名服务器向权限域名服务器dns.abc.com进行查询。
❼ 权限域名服务器dns.abc.com告诉本地域名服务器，所查询的主机的IP地址。
❽ 本地域名服务器最后把查询结果告诉主机m.xyz.com。

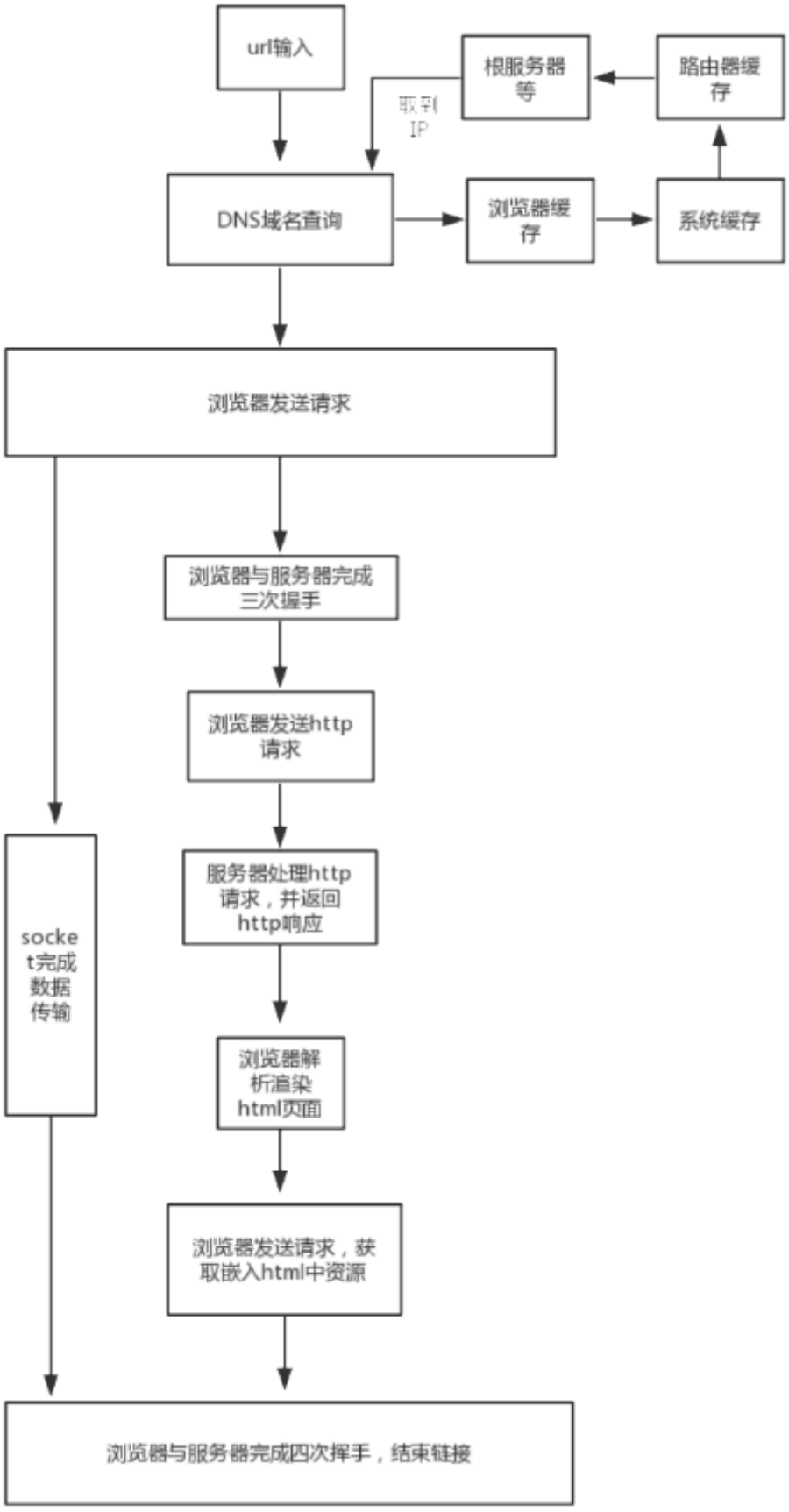
这8个步骤总共要使用8个UDP用户数据报的报文。本地域名服务器经过三次迭代查询后，从权限域名服务器dns.abc.com得到了主机y.abc.com的IP地址，最后把结果返回给发起查询的主机m.xyz.com。
图(b)是本地域名服务器采用递归查询的情况。在这种情况下，本地域名服务器只需向根域名服务器查询一次，后面的几次查询都是在其他几个域名服务器之间进行的（步骤❸至❻）。只是在步骤❼，本地域名服务器从根域名服务器得到了所需的IP地址。最后在步骤❽，本地域名服务器把查询结果告诉主机m.xyz.com。整个的查询也是使用8个UDP报文。

# 几种域名解析方式

- A记录，A代表的是Address，用来指定域名对应的IP地址，如将item.taobao.com指定到115.238.23.241，将switch.taobao.com指定到121.14.24.241。A记录可以将多个域名解析到一个IP地址，但是不能将一个域名解析到多个IP地址。
- MX记录，表示的是Mail Exchange，就是可以将某个域名下的邮件服务器指向自己的Mail Server，如taobao.com域名的A记录IP地址是115.238.25.245，如果MX记录设置为115.238.25.246，是xxx@taobao.com的邮件路由，DNS会将邮件发送到115.238.25.246所在的服务器，而正常通过Web请求的话仍然解析到A记录的IP地址。
- CNAME记录，全称是Canonical Name（别名解析）。所谓的别名解析就是可以为一个域名设置一个或者多个别名。如将taobao.com解析到xulingbo.net，将srcfan.com也解析到xulingbo.net。其中xulingbo.net分别是taobao.com和srcfan.com的别名。前面的跟踪域名解析中的"www.taobao.com. 1542  IN CNAME www.gslb.taobao.com"就是CNAME解析。
- NS记录，为某个域名指定DNS解析服务器，也就是这个域名有指定的IP地址的DNS服务器去解析，前面的"gslb.taobao.com. 86400 IN NS  gslbns2.taobao. com."就是NS解析。
- TXT记录，为某个主机名或域名设置说明，如可以为xulingbo.net设置TXT记录为"君山的博客|许令波"这样的说明。

# Web请求流程

- 输入URL请求地址
- 通过DNS服务服务器查询目标IP
- 浏览器向目标服务器发送建立连接请求
- 浏览器与服务器完成三次握手，建立链接
- socket作为网络通信的基本操作单元，负责完成数据传输
- 浏览器发送http请求
- 服务器处理http请求，并返回http响应
- 浏览器解析渲染html页面，转换成人类易懂内容
- 浏览器与服务器完成四次挥手，断开链接


url输入
根服务器等
路由器缓存
取到IP
DNS域名查询
浏览器缓存
系统缓存
浏览器发送请求
浏览器与服务器完成三次握手
浏览器发送http请求
服务器处理http请求，并返回http响应
浏览器解析渲染html页面
浏览器发送请求，获取嵌入html中资源
socket完成数据传输
浏览器与服务器完成四次挥手，结束链接

## 典型的HTTP请求头

| 请求头 | 含义说明 |
| --- | --- |
| Accept | 指定客户端能处理的 MIME（Multipurpose Internet Mail Extension，多用途互联网邮件扩展）页面类型。例如，Accept：image/gif，表明客户端希望接受 GIF 图像格式的资源；Accept：text/html，表明客户端希望接受 HTML 文本 |
| Accept-Charset | 指定客户端可以接受的字符集。例如，Accept-Charset:iso-8859-1,gb2312，表示客户端可以接受的字符集在所列的两个标准中。如果在请求消息中没有设置这个请求头，默认是任何字符集都可以接受 |
| Accept-Encoding | 指定客户端能进行解码的数据编码方式。例如，Accept-Encoding:gzip，compress，表示客户可以接受 gzip 和 compress 两种编码格式。如果请求消息中没有设置这个请求头，那么服务器假定客户端可以接受各种内容编码 |
| Accept-Language | 指定客户端能处理的语言类型。例如，Accept-Language:zh-cn，表示客户端只接受中文简体。如果请求消息中没有设置这个请求头，那么服务器假定客户端对各种语言都可以接受 |
| Authorixation | 指定客户端信任的凭据列表。当浏览器访问一个页面时，如果收到服务器的响应代码为 401（未授权），那么可以发送一个包含 Authorization 请求报头域的请求，要求服务器对所列的用户账户进行验证 |
| Cookie | 将一个以前设置的 Cookie 送回给服务器 |
| Connections | 请求采用持续连接方式。例如，Connection:Keep-Alive |
| Date | 指定消息发送时的日期和时间。例如，Date：Tue, 11 Jul 2012 18:23:51 GMT |
| From | 向服务器请求一个电子邮箱地址。例如，From：winda@microsoft.com，这是假设作者在访问微软网站时向微软的服务器申请的一个电子邮箱地址 |
| Host | 指定被请求服务器的域名和端口，如果使用默认的 80 端口，则可以省略端口指定步骤。例如，Host：www.51cto.com，表示请求的服务器域名为 www.51cto.com，且使用默认的 TCP 80 端口进行访问 |
| Referer | 包括一条 URI 信息，要求从指定的 URI 中访问当前请求的页面 |
| User-Agent | 允许客户端将它的操作系统、浏览器和其他属性告知服务器。例如，上网登录论坛的时候，往往会看到一些欢迎信息，其中列出了登录者的操作系统的名称和版本，以及他所使用的浏览器的名称和版本，这就是 User-Agent 请求头的“功劳”了 |
| Upgrade | 客户端希望切换到新的协议 |

# 典型的HTTP响应头

| 响应头 | 含义说明 |
| --- | --- |
| Allow | 显示服务器支持哪些请求方法 |
| Server | 显示服务器的软件信息。例如，Server：Apache-Coyote/1.1，表示 Web 服务器软件为 Apache-Coyote，版本为 1.1 |
| Content-Encoding | 显示请求文档采用的编码方法。例如，Content-Encoding：gzip，表示文档采用的是 gzip 压缩方式 |
| Content-Language | 显示请求页面使用的语言。例如，Content-Language：zh/zh-cn，表示页面支持的语言 |
| Content-Length | 显示请求页面的长度（以字节为单位）。例如，Content-Length：1024，表示页面大小为 1024 字节 |
| Content-type | 显示页面支持的 MIME 类型。例如，Content-type：text/html，表示页面支持 text 和 HTML 两种消息类型 |
| Date | 指定消息发送时的日期和时间。例如，Date：Tue, 11 Jul 2012 18:23:51 GMT |
| Last-Modified | 显示请求页面最后被编辑或修改的日期和时间。例如，Last-Modified：Tue, 11 Jul 2012 12:11:51 GMT |
| Location | 指示客户端将请求发送到指定的位置，起到重定向的作用 |
| Accept-Range | 显示服务端器将接受指定字节范围内的请求。例如，Accept-Range：1024，表示服务器只接受 1024 字节以内的请求 |
| Refresh | 指示客户端多少秒后再刷新或者重新访问本页面。例如，Refresh：5，表示要客户端过 5 秒后再刷新或访问本页面 |
| Set-Cookie | 指示客户端设置和页面关联的 Cookie |
| Upgrade | 服务器希望切换到新的协议 |

# 常见的 HTTP 请求方法

`GET`

`GET` 请求会`显示`请求指定的资源。一般来说 `GET` 方法应该只用于数据的读取，而不应当用于会产生副作用的`非幂等`的操作中。

`GET` 会方法请求指定的页面信息，并返回响应主体，`GET` 被认为是不安全的方法，因为 `GET` 方法会被网络蜘蛛等任意的访问。

`HEAD`

`HEAD` 方法与 `GET` 方法一样，都是向服务器发出指定资源的请求。但是，服务器在响应 `HEAD` 请求时不会回传资源的内容部分，即：响应主体。这样，我们可以不传输全部内容的情况下，就可以获取服务器的响应头信息。`HEAD` 方法常被用于客户端查看服务器的性能。

`POST`

`POST` 请求会 向指定资源提交数据，请求服务器进行处理，如：表单数据提交、文件上传等，请求数据会被包含在请求体中。`POST` 方法是`非幂等`的方法，因为这个请求可能会创建新的资源或/和修改现有资源。

`PUT`

`PUT` 请求会身向指定资源位置上传其最新内容，`PUT` 方法是`幂等`的方法。通过该方法客户端可以将指定资源的最新数据传送给服务器取代指定的资源的内容。

`DELETE`

`DELETE` 请求用于请求服务器删除所请求 `URI`（统一资源标识符，Uniform Resource Identifier）所标识的资源。`DELETE` 请求后指定资源会被删除，`DELETE` 方法也是`幂等`的。

`CONNECT`

`CONNECT` 方法是 `HTTP/1.1` 协议预留的，能够将连接改为管道方式的代理服务器。通常用于SSL加密服务器的链接与非加密的HTTP代理服务器的通信。

`OPTIONS`

`OPTIONS` 请求与 `HEAD` 类似，一般也是用于客户端查看服务器的性能。 这个方法会请求服务器返回该资源所支持的所有HTTP请求方法，该方法会用'*'来代替资源名称，向服务器发送 `OPTIONS` 请求，可以测试服务器功能是否正常。JavaScript的XMLHttpRequest对象进行 `CORS` 跨域资源共享时，就是使用 `OPTIONS` 方法发送嗅探请求，以判断是否有对指定资源的访问权限。 允许

`TRACE`

`TRACE` 请求服务器回显其收到的请求信息，该方法主要用于HTTP请求的测试或诊断。

**HTTP/1.1之后增加的方法**

在`HTTP/1.1`标准制定之后，又陆续扩展了一些方法。其中使用中较多的是`PATCH`方法：

`PATCH`

`PATCH`方法出现的较晚，它在2010年的RFC 5789标准中被定义。`PATCH`请求与`PUT`请求类似，同样用于资源的更新。二者有以下两点不同：

- 但`PATCH`一般用于资源的部分更新，而`PUT`一般用于资源的整体更新。
- 当资源不存在时，`PATCH`会创建一个新的资源，而`PUT`只会对已在资源进行更新。

其中一些方法可能会对 Web 应用程序造成安全风险，因为它们允许攻击者修改存储在 Web 服务器上的文件，并
在某些情况下窃取合法的证书用户。 更具体地操作，应该禁用的方法如下：
 PUT：该方法允许客户端上传 Web 服务器上的新文件。 攻击者可以利用它通过上传恶意文件（例如：通过调
用 cmd.exe 执行命令的 asp 文件）或者通过简单地使用受害者服务器作为文件存储。
 DELETE：此方法允许客户端删除 Web 服务器上的文件。 攻击者可以利用它作为一种非常简单直接的方式来
破坏网站或发起 DoS 攻击。
 CONNECT：此方法可以允许客户端将 Web 服务器用作代理。
 TRACE：这个方法简单地返回给客户端，不管字符串是否发送给客户端服务器，主要用于调试目的。

# HTTP状态码分类

| **1**** | **信息，服务器收到请求，需要请求者继续执行操作** |
| --- | --- |
| 2** | 成功，操作被成功接收并处理 |
| 3** | 重定向，需要进一步的操作以完成请求 |
| 4** | 客户端错误，请求包含语法错误或无法完成请求 |
| 5** | 服务器错误，服务器在处理请求的过程中发生了错误 |

**HTTP状态码列表**

| **100** | **Continue** | **继续。客户端应继续其请求** |
| --- | --- | --- |
| 101 | Switching Protocols | 切换协议。服务器根据客户端的请求切换协议。只能切换到更高级的协议，例如，切换到HTTP的新版本协议 |
| | | |
| 200 | OK | 请求成功。一般用于GET与POST请求 |
| 201 | Created | 已创建。成功请求并创建了新的资源 |
| 202 | Accepted | 已接受。已经接受请求，但未处理完成 |
| 203 | Non-Authoritative Information | 非授权信息。请求成功。但返回的meta信息不在原始的服务器，而是一个副本 |
| 204 | No Content | 无内容。服务器成功处理，但未返回内容。在未更新网页的情况下，可确保浏览器继续显示当前文档 |
| 205 | Reset Content | 重置内容。服务器处理成功，用户终端（例如：浏览器）应重置文档视图。可通过此返回码清除浏览器的表单域 |
| 206 | Partial Content | 部分内容。服务器成功处理了部分GET请求 |
| | | |
| 300 | Multiple Choices | 多种选择。请求的资源可包括多个位置，相应可返回一个资源特征与地址的列表用于用户终端（例如：浏览器）选择 |
| 301 | Moved Permanently | 永久移动。请求的资源已被永久的移动到新URI，返回信息会包括新的URI，浏览器会自动定向到新URI。今后任何新的请求都应使用新的URI代替 |
| 302 | Found | 临时移动。与301类似。但资源只是临时被移动。客户端应继续使用原有URI |
| 303 | See Other | 查看其它地址。与301类似。使用GET和POST请求查看 |
| 304 | Not Modified | 未修改。所请求的资源未修改，服务器返回此状态码时，不会返回任何资源。客户端通常会缓存访问过的资源，通过提供一个头信息指出客户端希望只返回在指定日期之后修改的资源 |
| 305 | Use Proxy | 使用代理。所请求的资源必须通过代理访问 |
| 306 | Unused | 已经被废弃的HTTP状态码 |
| 307 | Temporary Redirect | 临时重定向。与302类似。使用GET请求重定向 |
| | | |
| 400 | Bad Request | 客户端请求的语法错误，服务器无法理解 |
| 401 | Unauthorized | 请求要求用户的身份认证 |
| 402 | Payment Required | 保留，将来使用 |

| **100** | **Continue** | **继续。客户端应继续其请求** |
| --- | --- | --- |
| 403 | Forbidden | 服务器理解请求客户端的请求，但是拒绝执行此请求 |
| 404 | Not Found | 服务器无法根据客户端的请求找到资源（网页）。通过此代码，网站设计人员可设置"您所请求的资源无法找到"的个性页面 |
| 405 | Method Not Allowed | 客户端请求中的方法被禁止 |
| 406 | Not Acceptable | 服务器无法根据客户端请求的内容特性完成请求 |
| 407 | Proxy Authentication Required | 请求要求代理的身份认证，与401类似，但请求者应当使用代理进行授权 |
| 408 | Request Time-out | 服务器等待客户端发送的请求时间过长，超时 |
| 409 | Conflict | 服务器完成客户端的 PUT 请求时可能返回此代码，服务器处理请求时发生了冲突 |
| 410 | Gone | 客户端请求的资源已经不存在。410不同于404，如果资源以前有现在被永久删除了可使用410代码，网站设计人员可通过301代码指定资源的新位置 |
| 411 | Length Required | 服务器无法处理客户端发送的不带Content-Length的请求信息 |
| 412 | Precondition Failed | 客户端请求信息的先决条件错误 |
| 413 | Request Entity Too Large | 由于请求的实体过大，服务器无法处理，因此拒绝请求。为防止客户端的连续请求，服务器可能会关闭连接。如果只是服务器暂时无法处理，则会包含一个Retry-After的响应信息 |
| 414 | Request-URI Too Large | 请求的URI过长（URI通常为网址），服务器无法处理 |
| 415 | Unsupported Media Type | 服务器无法处理请求附带的媒体格式 |
| 416 | Requested range not satisfiable | 客户端请求的范围无效 |
| 417 | Expectation Failed | 服务器无法满足Expect的请求头信息 |
| | | |
| 500 | Internal Server Error | 服务器内部错误，无法完成请求 |
| 501 | Not Implemented | 服务器不支持请求的功能，无法完成请求 |

| **100** | **Continue** | **继续。客户端应继续其请求** |
| --- | --- | --- |
| 502 | Bad Gateway | 作为网关或者代理工作的服务器尝试执行请求时，从远程服务器接收到了一个无效的响应 |
| 503 | Service Unavailable | 由于超载或系统维护，服务器暂时的无法处理客户端的请求。延时的长度可包含在服务器的Retry-After头信息中 |
| 504 | Gateway Time-out | 充当网关或代理的服务器，未及时从远端服务器获取请求 |
| 505 | HTTP Version not supported | 服务器不支持请求的HTTP协议的版本，无法完成处理 |