HTTP协议详解

引言

http协议（超文本传输协议，HyperText Transfer Protocol）是互联网上应用最为广泛的一种网络协议，所有的WWW文件都必须遵守这个标准。设计HTTP最初的目的是为了提供一种发布和接受HTML页面的方法。1960年美国人Ted Nelson构思了一种通过计算机处理文本信息的方法，并称之为超文本(hypertext),这成为了HTTP超文本传输协议标准架构的发展根基。

一、http协议的介绍

http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的链接方式，通常，由HTTP客户端发起请求，建立一个到服务器指定端口的TCP连接。然后服务器从那个端口监听客户端接收发送过来的请求。一旦收到请求，服务器会向客户端发回一个状态行，比如“HTTP/1.1 200 OK ”,和消息，消息的消息体可能是请求的文件、错误信息、或者其他一些信息。HTTP使用TCP而不是UDP的原因在于（打开）一个网页必须传送很多数据，而TCP协议提供传输控制，按顺序组织数据，和错误纠正。

HTTP 协议的主要特点

- 支持客户/服务器模式
- 简单快速：客户向服务器请求服务时，只需传送请求方法和路径。请求方法常用的有GET、HEAD、POST。每种方法规定了客户与服务器联系的类型不同。由于HTTP协议简单，使得HTTP服务器的程序规模小，因而通信速度很快
- 灵活：HTTP允许传输任意类型的数据对象。正在传输的类型由Content-Type加以标记。
- 无连接：无连接的含义是限制每次连接只处理一个请求。服务器处理完客户的请求，并收到客户的应答后，即断开连接。采用这种方式可以节省传输时间。
- 无状态：HTTP协议是无状态协议。无状态是指协议对于事务处理没有记忆能力。缺少状态意味着如果后续处理需要前面的信息，则它必须重传，这样可能导致每次连接传送的数据量增大。另一方面，在服务器不需要先前信息时它的应答就较快。

二、HTTP协议的请求

http请求由三部分组成，分别是：请求行、消息报头、请求正文。

1、请求行以一个方法符号开头，以空格分开，后面跟着请求的URI和协议的版本，格式如下：

Method Request-URI HTTP-Version CRLF

其中Method表示请求方法； Request-URI是一个统一资源标识符；HTTP-Version表示请求的HTTP协议版本；CRLF表示回车和换行（除了作为结尾的CRLF外，不允许出现单独的CR或LF字符）。

请求方法（所有方法全为大写）有多种，各个方法的解释如下：

GET 请求获取Request-URI所标识的资源

POST 在Request-URI所标识的资源后附加新的数据

HEAD 请求获取由Request-URI所标识的资源的响应消息报头

PUT 请求服务器存储一个资源，并用Request-URI作为其标识

DELETE 请求服务器删除Request-URI所标识的资源

TRACE 请求服务器回送收到的请求信息，主要用于测试或诊断

CONNECT 保留将来使用
OPTIONS 请求查询服务器的性能，或者查询与资源相关的选项和需求

应用举例：
GET方法：在浏览器的地址栏中输入网址的方式访问网页时，浏览器采用GET方法向服务器获取资源，eg:GET /form.html HTTP/1.1 (CRLF)
POST方法要求被请求服务器接受附在请求后面的数据，常用于提交表单。
eg：POST /reg.jsp HTTP/ (CRLF)
Accept:image/gif,image/x-xbit,… (CRLF)

三、HTTP的响应

HTTP响应也是由三个部分组成，分别是：状态行、消息报头、响应正文1、状态行格式如下：
HTTP-Version Status-Code Reason-Phrase CRLF
其中，HTTP-Version表示服务器HTTP协议的版本；Status-Code表示服务器发回的响应状态代码；Reason-Phrase表示状态代码的文本描述。
状态代码有三位数字组成，第一个数字定义了响应的类别，且有五种可能取值：
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
常见状态代码、状态描述、说明：200 OK //客户端请求成功
400 Bad Request //客户端请求有语法错误，不能被服务器所理解
401 Unauthorized //请求未经授权，这个状态代码必须和WWW-Authenticate报头域一起使用
403 Forbidden //服务器收到请求，但是拒绝提供服务
404 Not Found //请求资源不存在，eg：输入了错误的URL
500 Internal Server Error //服务器发生不可预期的错误
503 Server Unavailable //服务器当前不能处理客户端的请求，一段时间后可能恢复正常

四、HTTP的消息报头

> HTTP消息由客户端到服务器的请求和服务器到客户端的响应组成。请求消息和响应消息都是由开始行（对于请求消息，开始行就是请求行，对于响应消息，开始行就是状态行），消息报头（可选），空行（只有CRLF的行），消息正文（可选）组成。

HTTP消息报头包括**普通报头**、**请求报头**、**响应报头**、**实体报头。**
每一个报头域都是由名字+“：”+空格+值 组成，消息报头域的名字是大小写无关的。

- **请求报头**：

请求报头：

Accept　　用于指定客户端接收哪些类型的信息。eg：text/html　html文本文件　　image/gif　希望接收gif格式的图像。

Accept-Encoding　指定可接收的内容编码。默认什么都可以接收。

Accept-Language　用于指定接收的自然语言。默认都可以接收。自然语言这边就不详细介绍了。

Authorization　　用于证明客户端有权查看某个资源。

Host　　主要用于指定被请求资源的Internet主机和端口号，即域名，一般从url中提取出来，默认端口是80，可以指定端口。发送请求时，这个报头是必须的！！！

User-Agent　服务器可以从这个报头域中获取客户端的操作系统、浏览器和其它属性

refer　　告诉服务器客户是从哪个页面链接过来的

- **响应报头**

Location　　用于重定向

Server　和请求报头中的User-Agent是对应的，包含一些服务器的信息

- **实体报头**

请求和响应消息都可以传送一个实体。一个实体由实体报头域和实体正文组成，但并不是说实体报头域和实体正文要在一起发送，可以只发送实体报头域。实体报头定义了关于实体正文（eg：有无实体正文）和请求所标识的资源的元信息。

常用的实体报头

Content-Encoding

Content-Encoding实体报头域被用作媒体类型的修饰符，它的值指示了已经被应用到实体正文的附加内容的编码，因而要获得Content-Type报头域中所引用的媒体类型，必须采用相应的解码机制。　　Content-Encoding这样用于记录文档的压缩方法，eg：Content-Encoding：gzip

Content-Language

Content-Language实体报头域描述了资源所用的自然语言。没有设置该域则认为实体内容将提供给所有的语言阅读
者。eg：Content-Language:da

Content-Length

Content-Length实体报头域用于指明实体正文的长度，以字节方式存储的十进制数字来表示。

Content-Type

Content-Type实体报头域用语指明发送给接收者的实体正文的媒体类型。eg:

Content-Type:text/html;charset=ISO-8859-1

Content-Type:text/html;charset=GB2312

Last-Modified

Last-Modified实体报头域用于指示资源的最后修改日期和时间。

Expires

Expires实体报头域给出响应过期的日期和时间。为了让代理服务器或浏览器在一段时间以后更新缓存中(再次访问曾访问过的页面时，直接从缓存中加载，缩短响应时间和降低服务器负载)的页面，我们可以使用Expires实体报头域指定页面过期的时间。eg：Expires：Thu，15 Sep 2006 16:23:12 GMT

状态代码有三位数字组成，第一个数字定义了响应的类别，且有五种可能取值:

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