

Http协议

一 HTTP概述

HTTP (hypertext transport protocol) , 即超文本传输协议。这个协议详细规定了浏览器和万维网服务器之间互相通信的规则。

HTTP就是一个通信规则, 通信规则规定了客户端发送给服务器的内容格式, 也规定了服务器发送给客户端的内容格式。其实我们要学习的就是这个两个格式! 客户端发送给服务器的格式叫“请求协议”; 服务器发送给客户端的格式叫“响应协议”。

特点:

- HTTP叫超文本传输协议, 基于请求/响应模式的!
- HTTP是**无状态协议**。

URL: 统一资源定位符, 就是一个网址: 协议名://域名:端口/路径, 例如: `http://www.oldboy.cn:80/index.html`

二 请求协议

请求协议的格式如下:

```
请求首行;    // 请求方式 请求路径 协议和版本, 例如: GET /index.html HTTP/1.1
请求头信息;  // 请求头名称:请求头内容, 即为key:value格式, 例如: Host:localhost
空行;        // 用来与请求体分隔开
请求体。     // GET没有请求体, 只有POST有请求体。
```

浏览器发送给服务器的内容就是这个格式的, 如果不是这个格式服务器将无法解读! 在HTTP协议中, 请求有很多请求方法, 其中最为常用的就是GET和POST。不同的请求方法之间的区别, 后面会一点点的介绍。

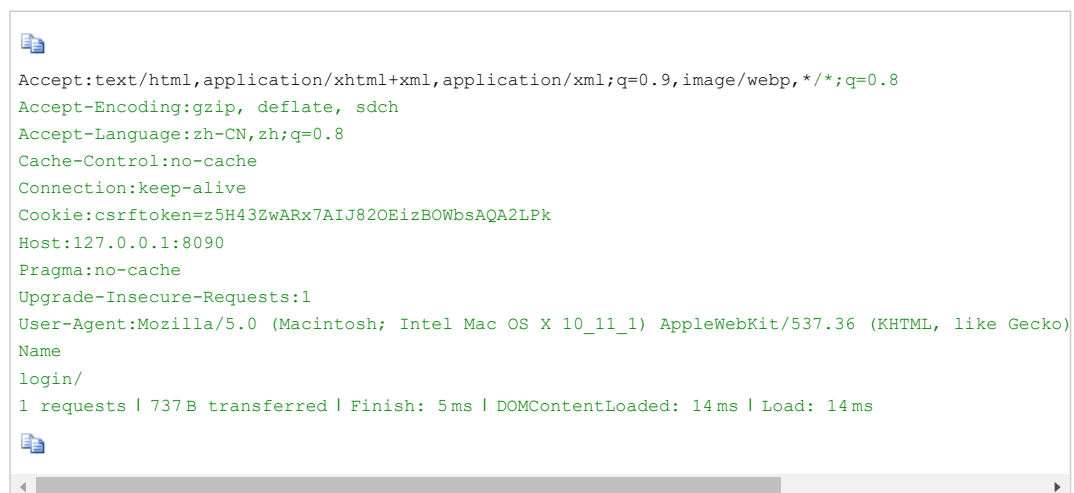
2.1 GET请求

HTTP默认的请求方法就是GET

- * 没有请求体
- * 数据必须在1K之内!
- * GET请求数据会暴露在浏览器的地址栏中

GET请求常用的操作:


1. 在浏览器的地址栏中直接给出URL, 那么就一定是GET请求
2. 点击页面上的超链接也一定是GET请求
3. 提交表单时, 表单默认使用GET请求, 但可以设置为POST



- GET 127.0.0.1:8090/login HTTP/1.1: GET请求, 请求服务器路径为 127.0.0.1:8090/login , 协议为1.1;

- Host:localhost: 请求的主机名为localhost;
- *User-Agent: Mozilla/5.0 (**Windows NT 5.1**; rv:5.0) Gecko/20100101 **Firefox/5.0**: 与浏览器和OS相关的信息。有些网站会显示用户的系统版本和浏览器版本信息, 这都是通过获取User-Agent头信息而来的;
- Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8: 告诉服务器, 当前客户端可以接收的文档类型, 其实这里包含了*/*, 就表示什么都可以接收;
- Accept-Language: zh-cn,zh;q=0.5: 当前客户端支持的语言, 可以在浏览器的工具□选项中找到语言相关信息;
- Accept-Encoding: gzip, deflate: 支持的压缩格式。数据在网络上传递时, 可能服务器会把数据压缩后再发送;
- Accept-Charset: GB2312,utf-8;q=0.7,*;q=0.7: 客户端支持的编码;
- Connection: keep-alive: 客户端支持的链接方式, 保持一段时间链接, 默认为3000ms;
- Cookie: JSESSIONID=369766FDF6220F7803433C0B2DE36D98: 因为不是第一次访问这个地址, 所以会在请求中把上一次服务器响应中发送过来的Cookie在请求中一并发送去过; 这个Cookie的名字为JSESSIONID。

注意



HTTP无状态: 无状态是指协议对于事务处理没有记忆能力, 服务器不知道客户端是什么状态。从另一方面讲, 打开一个服务器上的网和你之前打开这个服务器上的网页之间没有任何联系


如果你要实现一个购物车, 需要借助于Cookie或Session或服务器端API (如NSAPI and ISAPI) 记录这些信息, 请求服务器结束当你登录到一个网站时, 你的登录状态也是由Cookie或Session来“记忆”的, 因为服务器并不知道你是否登录

优点: 服务器不用为每个客户端连接分配内存来记忆大量状态, 也不用在客户端失去连接时去清理内存, 以更高效地去处理WEB业务

缺点: 客户端的每次请求都需要携带相应参数, 服务器需要处理这些参数

容易犯的误区:

- 1、HTTP是一个无状态的面向连接的协议, 无状态不代表HTTP不能保持TCP连接, 更不能代表HTTP使用的是UDP协议 (无连接)
- 2、从HTTP/1.1起, 默认都开启了Keep-Alive, 保持连接特性, 简单地说, 当一个网页打开完成后, 客户端和服务端之间用于传输HTTP数据的TCP连接不会关闭, 如果客户端再次访问这个服务器上的网页, 会继续使用这一条已经建立的连接
- 3、Keep-Alive不会永久保持连接, 它有一个保持时间, 可以在不同的服务器软件 (如Apache) 中设定这个时间



2.2 POST请求

- (1). 数据不会出现在地址栏中
- (2). 数据的大小没有上限
- (3). 有请求体
- (4). 请求体中如果存在中文, 会使用URL编码!

```
1 | username=%E5%BC%A0%E4%B8%89&password=123
```



我们都知道Http协议中参数的传输是“key=value”这种简直对形式的, 如果要传多个参数就需要用“&”符号对键值对进行分割。如

针对“name1=value1&name2=value2”我们来说一下客户端到服务端的概念上解析过程:

上述字符串在计算机中用ASCII码表示为:

```
6E616D6531 3D 76616C756531 26 6E616D6532 3D 76616C756532.
6E616D6531: name1
3D: =
76616C756531: value1
26: &
6E616D6532: name2
3D: =
76616C756532: value2
```

服务端在接收到该数据后就可以遍历该字节流, 首先一个字节一个字节地吃, 当吃到3D这字节后, 服务端就知道前面吃得字节表

现在有这样一个问题, 如果我的参数值中就包含=或&这种特殊字符的时候该怎么办。

比如说“name1=value1”, 其中value1的值是“va&lu=e1”字符串, 那么实际在传输过程中就会变成这样“name1=va&lu=e1”。我

如何解决上述问题带来的歧义呢？解决的办法就是对参数进行URL编码

URL编码只是简单的在特殊字符的各个字节前加上%，例如，我们对上述会产生奇异的字符进行URL编码后结果："name1=va%26"



使用表单可以发POST请求，但表单默认是GET

```
1 <form action="" method="post">
2   关键字: <input type="text" name="keyword"/>
3   <input type="submit" value="提交"/>
4 </form>
```

输入yuan后点击提交，查看请求内容如下：



Request Headers

```
Accept:text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Encoding:gzip, deflate
Accept-Language:zh-CN,zh;q=0.8
Cache-Control:no-cache
Connection:keep-alive
Content-Length:13
Content-Type:application/x-www-form-urlencoded
Cookie:csrftoken=z5H43ZwARx7AIJ820EizBOWbsAQa2LPk
Host:127.0.0.1:8090
Origin:http://127.0.0.1:8090
Pragma:no-cache
Referer:http://127.0.0.1:8090/login/
Upgrade-Insecure-Requests:1
User-Agent:Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_1)
        AppleWebKit/537.36 (KHTML, like Gecko) Chrome/53.0.2785.89 Safari/537.36
```

Form Data

username:yuan



POST请求是可以有体的，而GET请求不能有请求体。

- **Referer:** http://localhost:8080/hello/index.jsp：请求来自哪个页面，例如你在百度上点击链接到了这里，那么Referer:http://www.baidu.com；如果你是在浏览器的地址栏中直接输入的地址，那么就没有Referer这个请求头了；
- **Content-Type:** application/x-www-form-urlencoded：表单的数据类型，说明会使用url格式编码数据；url编码的数据都是以“%”为前缀，后面跟随两位的16进制。
- **Content-Length:**13：请求体的长度，这里表示13个字节。
- **keyword=hello**：请求体内容！hello是在表单中输入的数据，keyword是表单字段的名字。



Referer请求头是比较有用的一个请求头，它可以用来做统计工作，也可以用来做防盗链。

统计工作：我公司网站在百度上做了广告，但不知道在百度上做广告对我们网站的访问量是否有影响，那么可以对每个请求中的Referer进行统计。

防盗链：我公司网站上有一个下载链接，而其他网站盗链了这个地址，例如在我网站上的index.html页面中有一个链接，点击即可



三 响应协议

3.1 响应内容

响应协议的格式如下：

```
响应首行；
响应头信息；
空行；
响应体。
```

响应内容是由服务器发送给浏览器的内容，浏览器会根据响应内容来显示。遇到会开一个新的线程加载，所以有时图片多的话，内容会先显示出来，然后图片才一张张加载出来。



```
Request URL:http://127.0.0.1:8090/login/
Request Method:GET
Status Code:200 OK
```

```
Remote Address:127.0.0.1:8090
Response Headers
view source
Content-Type:text/html; charset=utf-8
Date:Wed, 26 Oct 2016 06:48:50 GMT
Server:WSGIServer/0.2 CPython/3.5.2
X-Frame-Options:SAMEORIGIN

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Title</title>
</head>
<body>
<form action="/login/" method="post">
  用户名: <input type="text" name="username"/>
  <input type="submit" value="提交"/>
</form>
</body>
</html>
```



- HTTP/1.1 200 OK: 响应协议为HTTP1.1, 状态码为200, 表示请求成功, OK是对状态码的解释;
- Server:WSGIServer/0.2 CPython/3.5.2: 服务器的版本信息;
- Content-Type: text/html;charset=UTF-8: 响应体使用的编码为UTF-8;
- Content-Length: 724: 响应体为724字节;
- Set-Cookie: JSESSIONID=C97E2B4C55553EAB46079A4F263435A4; Path=/hello: 响应给客户端的Cookie;
- Date: Wed, 25 Sep 2012 04:15:03 GMT: 响应的时间, 这可能会有8小时的时区差;

3.2 状态码

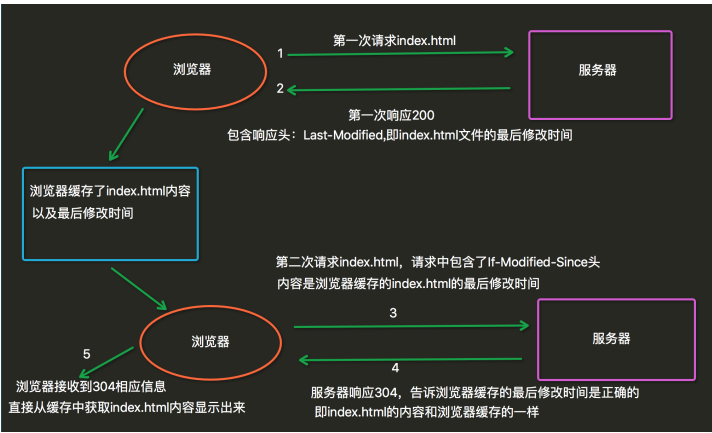
响应头对浏览器来说很重要, 它说明了响应的真正含义。例如200表示响应成功了, 302表示重定向, 这说明浏览器需要再发一个新的请求。

- 200: 请求成功, 浏览器会把响应体内容(通常是html) 显示在浏览器中;
- 404: 请求的资源没有找到, 说明客户端错误的请求了不存在的资源;
- 500: 请求资源找到了, 但服务器内部出现了错误;
- 302: 重定向, 当响应码为302时, 表示服务器要求浏览器重新再发一个请求, 服务器会发送一个响应头Location, 它指定了新请求的URL地址;
- 304:



当用户第一次请求index.html时, 服务器会添加一个名为Last-Modified响应头, 这个头说明了index.html的最后修改时间, 浏览器会把index.html内容, 以及最后响应时间缓存下来。当用户第二次请求index.html时, 在请求中包含一个名为If-Modified-Since请求头, 它的值就是第一次请求时服务器通过Last-Modified响应头发送给浏览器的值, 即index.html最后的修改时间, If-Modified-Since请求头就是在告诉服务器, 我这里浏览器缓存的index.html最后修改时间是这个, 您看看现在的index.html最后修改时间是不是这个, 如果还是, 那么您就不用再响应这个index.html内容了, 我会把缓存的内容直接显示出来。而服务器端会获取If-Modified-Since值, 与index.html的当前最后修改时间比对, 如果相同, 服务器会发响应码304, 表示index.html与浏览器上次缓存的相同, 无需再次发送, 浏览器可以显示自己的缓存页面, 如果比对不同, 那么说明index.html已经做了修改, 服务器会响应200。





3.3 其他响应头

告诉浏览器不要缓存的响应头:

- Expires: -1;
- Cache-Control: no-cache;
- Pragma: no-cache;


自动刷新响应头, 浏览器会在3秒之后请求http://www.baidu.com:

- Refresh: 3;url=http://www.baidu.com

3.4 HTML中指定响应头

在HTML页面中可以使用<meta http-equiv="" content="">来指定响应头, 例如在index.html页面中给出<meta http-equiv="Refresh" content="3;url=http://www.baidu.com">, 表示浏览器只会显示index.html页面3秒, 然后自动跳转到http://www.baidu.com.



 [Yuan先生](#)
[关注 - 1](#)
[粉丝 - 3966](#)
[我在关注他](#) [取消关注](#)

11 0

posted @ 2016-10-26 14:52 Yuan先生 阅读(9348) 评论(2) 编辑 收藏

Post Comment

#1楼 2018-03-07 22:37 | cyw7 回复 引用

mark 支持(1) 反对(0)

#2楼 2018-03-15 16:08 | 闹世闲人 回复 引用

mark 支持(0) 反对(0)

[刷新评论](#) [刷新页面](#) [返回顶部](#)

- 【推荐】超50万行VC++源码: 大型组态工控、电力仿真CAD与GIS源码库
- 【推荐】了不起的开发者, 挡不住的华为, 园子里的品牌专区
- 【推荐】九大训练营同开 2020阿里云大数据独门绝学
- 【推荐】前端精品集合之JavaScript实战100例

相关博文：

- [HTTP协议、HTTP协议原理分析](#)
- [HTTP协议（一） 初识HTTP协议](#)
- [Http协议](#)
- [HTTP协议](#)
- [HTTP协议\(一\)HTTP协议详解](#)
- » [更多推荐...](#)

最新 IT 新闻：

- [联合会会员能否让喜马拉雅翻越“顶峰”？](#)
- [百度发布新一代云基础架构“太行” 升级智能计算能力](#)
- [蔚来免费换电权益将于10月大调整 没订车的抓紧下订单](#)
- [华为希望90%国产软件可跑在鲲鹏CPU上 已适配2000多个产品](#)
- [阿里巴巴财报电话会议实录：张勇称阿里仍然是中国首选的消费平台](#)
- » [更多新闻...](#)

Copyright © 2020 Yuan先生
Powered by .NET Core on Kubernetes