[HTTP协议与HTML表单（再谈GET与POST的区别）](http://blog.csdn.net/darxin/article/details/4944225)

标签： [html](http://www.csdn.net/tag/html)[http服务器](http://www.csdn.net/tag/http%e6%9c%8d%e5%8a%a1%e5%99%a8)[input](http://www.csdn.net/tag/input)[action](http://www.csdn.net/tag/action)[encoding](http://www.csdn.net/tag/encoding)[url](http://www.csdn.net/tag/url)

2009-12-05 07:48 58553人阅读 [评论](http://blog.csdn.net/darxin/article/details/4944225#comments)(24) [收藏](javascript:void(0);) [举报](http://blog.csdn.net/darxin/article/details/4944225#report)

 分类：

WEB应用（4）

版权声明：本文为博主原创文章，未经博主允许不得转载。

目录[(?)[-]](http://blog.csdn.net/darxin/article/details/4944225)

1. [HTTP请求](http://blog.csdn.net/darxin/article/details/4944225#t0)
2. [HTML表单](http://blog.csdn.net/darxin/article/details/4944225#t1)
   1. [目标地址URL](http://blog.csdn.net/darxin/article/details/4944225#t2)
   2. [发送方式](http://blog.csdn.net/darxin/article/details/4944225#t3)
   3. [数据主体的编码方式](http://blog.csdn.net/darxin/article/details/4944225#t4)
3. [GET与POST的区别](http://blog.csdn.net/darxin/article/details/4944225#t5)
   1. [对网上传言的解释](http://blog.csdn.net/darxin/article/details/4944225#t6)
4. [参考资料](http://blog.csdn.net/darxin/article/details/4944225#t7)

本作品采用[知识共享署名-非商业性使用-相同方式共享 2.5 中国大陆许可协议](http://creativecommons.org/licenses/by-nc-sa/2.5/cn/)进行许可。

HTTP的GET/POST方式有何区别？这是一个老生常谈的问题，但老生常谈的问题往往有一些让人误解的结论。本文将带您浅尝HTTP协议，在了解HTTP协议的同时将会展示许多被人们忽视的内容。在掌握了HTTP协议的过程中我们将自然而然地了解到GET与POST的本质区别。

**HTTP请求**

从使用者的角度看，一个HTTP请求起始于  
用户端浏览器上输入的一个URL地址；  
网页中的一个超链接；  
提交一个HTML表单。  
但本质上说，一个HTTP请求起始于用户端向HTTP服务器发送的一个URL请求。

一个标准的HTTP请求由以下几个部分组成

<request-line>  
<headers>  
<CRLF>  
[<request-body><CRLF>]

在HTTP请求中，第一行是请求行（request-line），用来说明请求类型、要访问的资源（URL）以及使用的HTTP版本；  
紧接着是多行头部（headers）信息，用来说明服务器要使用的附加信息；  
头部信息之后是一个回车换行符（/r/n），用于标明头部信息的结束。  
以上是必须内容，根据需要可在头部信息结束之后增加主体数据（request-body）；  
主体数据之后是一个回车换行符（/r/n），用于标明主体数据的结束。

**需要注意的是**  
请求行（request-line）中的URL部分必须以application/x-www-form-urlencoded方式编码。  
主体数据（request-body）的编码方式由头部（headers）信息中的Content-Type指定。  
主体数据（request-body）的长度由头部（headers）信息中的Content-Length指定。

例如，我们可以在IE浏览器上输入下面的网址：

http://localhost:8000/hello/index.html

HTTP请求的头部信息如下：

GET /hello/index.html HTTP/1.1  
Accept: \*/\*  
Accept-Language: zh-cn  
Accept-Encoding: gzip, deflate  
Host: localhost:8000  
Connection: Keep-Alive  
Cookie: JSESSIONID=BBBA54D519F7A320A54211F0107F5EA6

[End]

上述信息没有request-body部分，这是以GET方式发送的HTTP请求。如果请求中需要附加主体数据，即增加request-body部分，则必须使用POST方式发送HTTP请求。HTML超链接（<a></a>）只能用GET方式提交HTTP请求，HTML表单（<form></form>）则可以使用两种方式提交HTTP请求。

**HTML表单**

HTML表单的使用方法如下：

**[xhtml]** [view plain](http://blog.csdn.net/darxin/article/details/4944225) [copy](http://blog.csdn.net/darxin/article/details/4944225)

1. **<form** action="目标地址" method="发送方式" enctype="数据主体的编码方式"**>**
2. <!-- 各类型的表单域 -->
3. **<input** name="NAME" value="VALUE"**/>**
4. **<textarea** name="NAME"**>**VALUE**</textarea>**
5. **<select** name="NAME"**>**
6. **<option** value="VALUE" selected="selected"**/>**
7. **</select>**
8. **</form>**

表单中存在各种类型的表单域标签，如<input/>、<textarea/>及<select/>。每一种表单域标签均有NAME与VALUE两种标签属性。这两个标签属性决定了表单提交时传送的属性名及相应的值。

**目标地址（URL）**

action标签属性指定了表单提交的目标地址，其值可以是完整的URL。如：

<form action="http://localhost:8000/hello/checkUser.html"></form>

如果放置表单的网页与表单提交的目标地址在同一个HTTP服务器上，则目标地址可以用绝对路径表示（绝对路径相对于HTTP服务器）。绝对路径以“/”开头，包括WEB应用上下文及请求。如：

<form action="/hello/checkUser.html"></form>

如果放置表单的网页与表单提交的目标地址在同一个WEB应用上下文上，则目标地址可以用相对路径表示（相对路径相对于放置表单的网页）。相对路径不以“/”开头，不包括WEB应用上下文。如：

<form action="checkUser.html"></form>

**需要注意的是**，action标签属性的值必须符合URL的要求，其编码必须符合application/x-www-form-urlencoded编码规则。如下面的表单：

**[xhtml]** [view plain](http://blog.csdn.net/darxin/article/details/4944225) [copy](http://blog.csdn.net/darxin/article/details/4944225)

1. <!-- 不符合要求的表单 -->
2. **<form** action="checkUser.html?opt=中文" method="POST"**>**
3. **</form>**

这样的表单是不符合要求的。如果其URL值存在非法字符（如中文字符），应将其进行URL Encoding处理。URL Encoding的处理方法如下：

* 字母数字字符 "a" 到 "z"、"A" 到 "Z" 和 "0" 到 "9" 保持不变。
* 特殊字符 "."、"-"、"\*" 和 "\_" 保持不变。
* 空格字符 " " 转换为一个加号 "+"。
* 所有其他字符都是不安全的，因此首先使用一种编码机制将它们转换为一个或多个字节。然后对每个字节用一个包含 3 个字符的字符串 "%xy" 表示，其中 xy 为该字节的两位十六进制表示形式。推荐的编码机制是 UTF-8。

将“中文”两个字符进行URL Encoding所得到的值就是“%E4%B8%AD%E6%96%87”。

所以正确的表单应该是：

**[xhtml]** [view plain](http://blog.csdn.net/darxin/article/details/4944225) [copy](http://blog.csdn.net/darxin/article/details/4944225)

1. <!-- 符合要求的表单 -->
2. **<form** action="checkUser.html?opt=%E4%B8%AD%E6%96%87" method="POST"**>**
3. **</form>**

**发送方式**

method标签属性指定了表单的发送方式，发送方式只有两种：GET及POST。  
当以GET方式发送表单时，发送的HTTP请求没有request-body部分，所以不需要指定enctype标签属性。

GET方式只提交表单域中的数据，action标签属性中如果存在?子句，GET方式将不予处理。如下面的表单：

**[xhtml]** [view plain](http://blog.csdn.net/darxin/article/details/4944225) [copy](http://blog.csdn.net/darxin/article/details/4944225)

1. **<form** action="checkUser.html?opt=xxx" method="GET"**>**
2. **<input** type="text" name="username" value="yyy"**/>**
3. **<input** type="text" name="age" value="zzz"**/>**
4. **<input** type="submit" value="submit"**/>**
5. **</form>**

表单提交时没有包括opt属性，HTTP头部信息如下：

GET /hello/checkUser.html**?username=yyy&age=zzz** HTTP/1.1  
Referer: http://localhost:8000/hello/index.html  
Accept: \*/\*  
Accept-Language: zh-cn  
Accept-Encoding: gzip, deflate  
Host: localhost:8000  
Connection: Keep-Alive  
Cookie: JSESSIONID=BBBA54D519F7A320A54211F0107F5EA6

[End]

**需要注意的是**，以GET方式提交表单时，每个表单域的NAME与VALUE要以URL的方式提交，所以每个表单域的NAME与VALUE均要进行URL Encoding处理。这个操作通常是由用户端浏览器完成的。如下面的表单：

**[xhtml]** [view plain](http://blog.csdn.net/darxin/article/details/4944225) [copy](http://blog.csdn.net/darxin/article/details/4944225)

1. **<form** action="checkUser.html" method="GET"**>**
2. **<input** type="hidden" name="opt" value="中文"**/>**
3. **<input** type="text" name="username" value="yyy"**/>**
4. **<input** type="text" name="age" value="zzz"**/>**
5. **<input** type="submit" value="submit"**/>**
6. **</form>**

其中表单域opt的VALUE是中文字符“中文”，在表单提交时，用户端浏览器会自动将其进行URL Encoding。HTTP头部信息如下：

GET /hello/checkUser.html?**opt=%E4%B8%AD%E6%96%87**&username=yyy&age=zzz HTTP/1.1  
Referer: http://localhost:8000/hello/index.html  
Accept: \*/\*  
Accept-Language: zh-cn  
Accept-Encoding: gzip, deflate  
Host: localhost:8000  
Connection: Keep-Alive  
Cookie: JSESSIONID=BBBA54D519F7A320A54211F0107F5EA6

[End]

当以POST方式发送表单时，表单域中的数据将作为request-body提交，action标签属性中的?子句将在request-line中得以保留。如下面的表单：

**[xhtml]** [view plain](http://blog.csdn.net/darxin/article/details/4944225) [copy](http://blog.csdn.net/darxin/article/details/4944225)

1. **<form** action="checkUser.html?opt=xxx" method="POST"**>**
2. **<input** type="text" name="username" value="yyy"**/>**
3. **<input** type="text" name="age" value="zzz"**/>**
4. **<input** type="submit" value="submit"**/>**
5. **</form>**

表单提交时，HTTP头部信息如下：

POST /hello/checkUser.html**?opt=xxx** HTTP/1.1  
Referer: http://localhost:8000/hello/index.html  
Accept: \*/\*  
Accept-Language: zh-cn  
Content-Type: application/x-www-form-urlencoded  
Accept-Encoding: gzip, deflate  
Host: localhost:8000  
Content-Length: 20  
Connection: Keep-Alive  
Cache-Control: no-cache  
Cookie: JSESSIONID=BBBA54D519F7A320A54211F0107F5EA6

**username=yyy&age=zzz**  
[End]

**需要注意的是**，以POST方式提交表单时，action标签属性的值必须是已经进行了URL Encoding处理之后的值，用户端浏览器不会自动处理URL中的非法字符。如下面的表单是不符合要求的：

**[xhtml]** [view plain](http://blog.csdn.net/darxin/article/details/4944225) [copy](http://blog.csdn.net/darxin/article/details/4944225)

1. <!-- 不符合要求的表单 -->
2. **<form** action="checkUser.html?opt=中文" method="POST"**>**
3. **<input** type="text" name="username" value="yyy"**/>**
4. **<input** type="text" name="age" value="zzz"**/>**
5. **<input** type="submit" value="submit"**/>**
6. **</form>**

正确的表单应该是：

**[xhtml]** [view plain](http://blog.csdn.net/darxin/article/details/4944225) [copy](http://blog.csdn.net/darxin/article/details/4944225)

1. **<form** action="checkUser.html?opt=%E4%B8%AD%E6%96%87" method="POST"**>**
2. **<input** type="text" name="username" value="yyy"**/>**
3. **<input** type="text" name="age" value="zzz"**/>**
4. **<input** type="submit" value="submit"**/>**
5. **</form>**

**数据主体的编码方式**

在HTTP请求中，request-line总是以application/x-www-form-urlencoded方式编码。enctype标签属性只对request-body起作用。也就是说只有在method="POST"的情况下，设置enctype才起作用。  
设置enctype标签属性后，在HTTP请求的头部（headers）信息中会多出一行Content-Type信息，并且request-body部分将会以Content-Type指定的MIME进行编码。这些操作都是由客户端浏览器自动完成的。

在没有指定enctype标签属性时，表单以默认的application/x-www-form-urlencoded方式对request-body进行编码。  
如果表单域中的NAME或VALUE含有非法字符（如中文字符），客户端浏览器会自动对其进行URL Encoding处理。如下面的表单：

**[xhtml]** [view plain](http://blog.csdn.net/darxin/article/details/4944225) [copy](http://blog.csdn.net/darxin/article/details/4944225)

1. **<form** action="checkUser.html" method="POST"**>**
2. **<input** type="hidden" name="opt" value="中文"**/>**
3. **<input** type="text" name="username" value="yyy"**/>**
4. **<input** type="text" name="age" value="zzz"**/>**
5. **<inupt** type="submit" value="submit"**/>**
6. **</form>**

表单提交时，HTTP头部信息如下：

POST /hello/checkUser.html HTTP/1.1  
Accept: \*/\*  
Referer: http://localhost:8000/hello/index.jsp  
Accept-Language: zh-cn  
Content-Type: application/x-www-form-urlencoded  
Accept-Encoding: gzip, deflate  
Host: localhost:8000  
Content-Length: 43  
Connection: Keep-Alive  
Cache-Control: no-cache  
Cookie: JSESSIONID=4EF9C5B81356481F470F3C60D9E77D94

**opt=%E4%B8%AD%E6%96%87**&username=yyy&age=zzz  
[End]

如果表单中包含需要上传的文件数据，则在指定method="POST"的同时还要指定enctype="multipart/form-data"。如下面的表单：

**[xhtml]** [view plain](http://blog.csdn.net/darxin/article/details/4944225) [copy](http://blog.csdn.net/darxin/article/details/4944225)

1. **<form** action="checkUser.html?opt=xxx" method="POST"
2. enctype="multipart/form-data"**>**
3. **<input** type="text" name="username" value="yyy"**/>**
4. **<input** type="text" name="age" value="zzz"**/>**
5. **<input** type="file" name="file" **/>**
6. **<inupt** type="submit" value="submit"**/>**
7. **</form>**

表单提交时HTTP头部信息如下：

POST /hello/checkUser.html?opt=xxx HTTP/1.1  
Accept: \*/\*  
Referer: http://localhost:8000/hello/index.html  
Accept-Language: zh-cn  
**Content-Type: multipart/form-data; boundary=---------------------------7d931c5d043e**  
Accept-Encoding: gzip, deflate  
Host: localhost:8000  
**Content-Length: 382**  
Connection: Keep-Alive  
Cache-Control: no-cache  
Cookie: JSESSIONID=6FE3D8E365DF9FE26221A32624470D24

-----------------------------7d931c5d043e  
Content-Disposition: form-data; name="username"

yyy  
-----------------------------7d931c5d043e  
Content-Disposition: form-data; name="age"

zzz  
-----------------------------7d931c5d043e  
Content-Disposition: form-data; name="file"; filename="C:/1.txt"  
Content-Type: text/plain

hello  
-----------------------------7d931c5d043e--

[End]

**GET与POST的区别**

HTTP请求的GET与POST方式的本质区别可以参考[hyddd](http://www.cnblogs.com/hyddd/)在[《浅谈HTTP中Get与Post的区别》](http://www.cnblogs.com/hyddd/archive/2009/03/31/1426026.html)一文中的描述，本文讲述了其中比较重要的一条，那就是数据传输的位置不同。  
GET方式在request-line中传送数据；POST方式在request-line及request-body中均可以传送数据。

**对网上传言的解释**

**传言1：**GET方式对长度有限制；POST方式对长度没限制。  
回答：长度限制之说一方面是HTTP客户端（如IE限定URL长度为2083字节，opera 是4050， Netscape 是8192）的限制；另一方面服务器的实现也加入了限制（如果URL长度过长，HTTP服务器会报414错误）。但HTTP协议及URL官方说明均对长度限制则没有规定。

**传言2：**GET是从服务器上获取数据；POST是向服务器传送数据。  
回答：GET方式就没有向服务器传送数据？那么URL中的?子句送的是什么？不论是GET还是POST，都可以向服务器传送数据，只不过传送数据的位置不同；不论是GET还是POST，都要从服务器上获取数据，否则IE浏览器拿什么东西给我们看呢？关键的问题是  
**GET的主要任务是获得数据，但在获得数据前也可以向服务器提交一些数据；   
POST的主要任务是提交数据，但在提交数据之后服务器也会向用户端返回一些显示用的数据。**

**传言3：**GET不安全，用户能从地址栏上看到传送的数据；POST安全，用户不能从地址栏上看到传送的数据。  
回答：POST方式看不到传送的数据是因为IE浏览器做了限制。如果你通过第三方工具看到了POST方式传送的数据，你还能说POST方式是安全的吗？理论上说GET和POST方式都不安全，要不就用不着研究HTTPS了。