


GET请求和POST请求的区别

3 回复 276 查看



(<https://www.shiyanlou.com/user/8490>) 实验楼管理员  (<https://www.shiyanlou.com/vip>)

2015-11-24 17:14

技术分享 (<https://www.shiyanlou.com/questions/?tag=技术分享>)

经常遇到「既然GET请求可以做POST请求的事情，为什么还要区分GET和POST而不只使用一个请求？」的问题。作为在实际中被使用最广的两个请求方法，这个问题其实挺难回答的，但万物总有其根由，今天就追根究底。

查看RFC规范再加上之前查过的一些二手文章，整理了如下的观点：

GET 被强制服务器支持

浏览器对URL的长度有限制，所以GET请求不能代替POST请求发送大量数据


GET请求发送数据更小

GET请求是安全的

GET请求是幂等的


POST请求不能被缓存

POST请求相对GET请求是「安全」的

 分享到微博

全部回答



实验楼管理员 (<https://www.shiyanlou.com/user/8490>)  (<https://www.shiyanlou.com/vip>)

(<https://www.shiyanlou.com/user/8490>)

GET被强制服务器支持

All general-purpose servers MUST support the methods GET and HEAD. All other methods are OPTIONAL.

GET 通常用于请求服务器发送某个资源。在HTTP/1.1中，要求服务器实现此方法；POST请求方法起初是用来向服务器输入数据的。在HTTP/1.1中，POST方法是可选被实现的，没有明确规定要求服务器实现。

浏览器对URL的长度有限制，所以GET请求不能代替POST请求发送大量数据

RFC 2616 (Hypertext Transfer Protocol — HTTP/1.1) states in section 3.2.1 that there is no limit to the length of an URI (URI is the official term for what most people call a URL)

RFC 2616 中明确对 uri 的长度并没有限制。不过虽然在RFC中并没有对uri的长度进行限制，但是各大浏览器厂家在实现的时候限制了URL的长度，可查到的是IE对长度限制为2083 (<https://support.microsoft.com/en-us/kb/208427>)；而chrome遇到长度很长的URL时，会直接崩溃 (<https://code.google.com/p/chromium/issues/detail?id=69227>)。

所以这条结论算是正确的。



GET请求发送数据更小

只能通过写代码验证了：下面第一个文件是服务器代码，作用是在客户端发送GET和POST请求的时候返回200状态码。第二个文件是客户端HTML文件，点击两个button，分别发送GET请求和POST请求。

```
import koa from 'koa'
import fs from 'mz/fs'

const app = koa()

app.use(function* (next) {
  if(this.path === '/test')
    return this.status = 200

  yield next
})

app.use(function* (next) {
  this.type = 'html'
  this.body = yield fs.readFile('./index.html')
  yield next
})

app.listen(8080)
console.log('koa server port: 8080')
```

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Title</title>
</head>
<body>
<button id="get">GET</button>
<button id="post">POST</button>
</body>
<script>
  function http(type) {
    return function (url) {
      var req = new XMLHttpRequest()
      req.open(type, url)

      req.send()
    }
  }

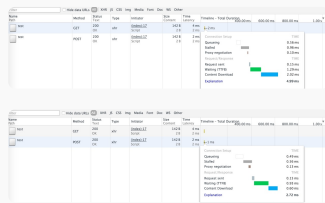
  var getDom = document.getElementById('get')
  , postDom = document.getElementById('post')
  , get = http('GET')
  , post = http('POST')

  getDom.addEventListener('click', function () {
    get('/test')
  })

  postDom.addEventListener('click', function () {
    post('/test')
  })

</script>
</html>
```

从上两张图可以看到POST请求的headers要比GET请求多了两个属性。所以这条结论其实也算是对的，不过从请求发送时间来看的话，其实两者并没有差别。



Of the request methods defined by this specification, the GET, HEAD, OPTIONS, and TRACE methods are defined to be safe.

2015-11-24 17:15



A request method is considered "idempotent" if the intended effect on the server of multiple identical requests with that method is the same as the effect for a single such request. Of the request methods defined by this specification, PUT, DELETE, and safe request methods are idempotent.

POST请求不能被缓存

In general, safe methods that do not depend on a current or authoritative response are defined as cacheable; this specification defines GET, HEAD, and POST as cacheable, although the overwhelming majority of cache implementations only support GET and HEAD.

这一点很多人都会质疑，被抓包之后的POST请求和GET请求是一样裸露的，所以更安全的说法是不对的。我这里所有的「安全」是相对的，因为GET请求有时候会直接反应在浏览器的地址栏，而现在的浏览器大多会记住曾经输入过的URL。试想如果你曾经在别人电脑上填过一个很私密的表单，那么你的这份记录很可能被连没什么电脑常识的人都一览无遗。

2015-11-24 17:15

登录后才能回答问题哟~

我要提问

标签

Linux (<https://www.shiyanlou.com/questions/?tag=Linux>) Python (<https://www.shiyanlou.com/questions/?tag=Python>)

C/C++ (<https://www.shiyanlou.com/questions/?tag=C/C++>) 实验环境 (<https://www.shiyanlou.com/questions/?tag=实验环境>)

技术分享 (<https://www.shiyanlou.com/questions/?tag=技术分享>) 功能建议 (<https://www.shiyanlou.com/questions/?tag=功能建议>)

课程需求 (<https://www.shiyanlou.com/questions/?tag=课程需求>) Java (<https://www.shiyanlou.com/questions/?tag=Java>)

其他 (<https://www.shiyanlou.com/questions/?tag=其他>) NodeJS (<https://www.shiyanlou.com/questions/?tag=NodeJS>)

SQL (<https://www.shiyanlou.com/questions/?tag=SQL>) Hadoop (<https://www.shiyanlou.com/questions/?tag=Hadoop>)

Web (<https://www.shiyanlou.com/questions/?tag=Web>) 常见问题 (<https://www.shiyanlou.com/questions/?tag=常见问题>)

PHP (<https://www.shiyanlou.com/questions/?tag=PHP>) Shell (<https://www.shiyanlou.com/questions/?tag=Shell>)

HTML (<https://www.shiyanlou.com/questions/?tag=HTML>) Git (<https://www.shiyanlou.com/questions/?tag=Git>)

HTML5 (<https://www.shiyanlou.com/questions/?tag=HTML5>) 信息安全 (<https://www.shiyanlou.com/questions/?tag=信息安全>)

网络 (<https://www.shiyanlou.com/questions/?tag=网络>) GO (<https://www.shiyanlou.com/questions/?tag=GO>)

NoSQL (<https://www.shiyanlou.com/questions/?tag=NoSQL>) Android (<https://www.shiyanlou.com/questions/?tag=Android>)

训练营 (<https://www.shiyanlou.com/questions/?tag=训练营>) Ruby (<https://www.shiyanlou.com/questions/?tag=Ruby>)

Perl (<https://www.shiyanlou.com/questions/?tag=Perl>)

相关问题

brackets：前端开发工程师必备编辑器 (<https://www.shiyanlou.com/questions/3197>)

优秀的计算机编程类博客和文章 (<https://www.shiyanlou.com/questions/3200>)

40个Java集合面试问题和答案 (<https://www.shiyanlou.com/questions/3187>)

程序员和设计师必备的20个CSS工具 (<https://www.shiyanlou.com/questions/3186>)

Web开发的十佳HTML5响应式框架 (<https://www.shiyanlou.com/questions/3177>)

动手做实验，轻松学IT。

实验楼-通过动手实践的方式学会IT技术。

公司简介 (<https://www.shiyanlou.com/aboutus>) 联系我们 (<https://www.shiyanlou.com/contact>) 常见问题 (<https://www.shiyanlou.com/faq#howtostart>)
我要开课 (<https://www.shiyanlou.com/labs>) 隐私协议 (<https://www.shiyanlou.com/privacy>) 会员条款 (<https://www.shiyanlou.com/terms>)
友情链接 (<https://www.shiyanlou.com/friends>)
站长统计 (http://www.cnzz.com/stat/website.php?web_id=5902315) 蜀ICP备13019762号 (<http://www.miibeian.gov.cn/>)



QQ群



微信



微博
(<http://weibo.com/shiyanlou2013>)