jQuery AJAX操作函数

# jQuery-AJAX简单介绍

AJAX 是与服务器**交换数据**的艺术，它在**不重载全部页面**的情况下，实现了对**部分网页的更新**。

**jQuery 提供多个与 AJAX 有关的方法。**

通过 **jQuery AJAX** 方法，您能够使用 **HTTP Get 和 HTTP Post** 从远程服务器上请求文本、HTML、XML 或 JSON - 同时您能够把这些外部数据**直接载入网页的被选元素中**。

提示：如果没有 **jQuery，AJAX 编程**还是有些难度的。

编写常规的 AJAX 代码并不容易，因为不同的浏览器对 AJAX 的实现并不相同。这意味着您必须编写额外的代码对浏览器进行测试。不过，jQuery 团队为我们解决了这个难题，我们只需要一行简单的代码，就可以实现 AJAX 功能。

# jQuery AJAX的重点方法

## jQuery load() 方法

jQuery load() 方法是简单但强大的 **AJAX 方法**；

load() 方法从服务器加载数据，并把**返回的数据**放入被选元素中。

语法：

**$(selector).load(URL,data,callback);**

### 必需的 URL 参数规定您希望加载的 URL。

### 可选的 data 参数规定与请求一同发送的**查询字符串键/值对集合**。

### 可选的 callback 参数是 load() 方法完成后所执行的函数名称。

示例：

把文件 "demo\_test.txt" 的内容加载到指定的 <div> 元素中：

$("#div1").load("demo\_test.txt");

也可以把 **jQuery 选择器**添加到 URL 参数，把 "demo\_test.txt" 文件中 id="p1" 的元素的内容加载到指定的 <div> 元素中：

$("#div1").load("demo\_test.txt **#p1**");

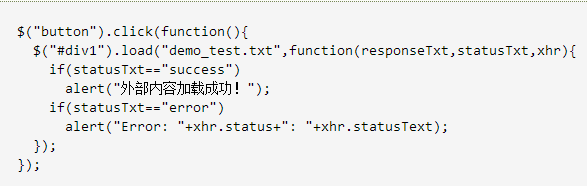
可选的 **callback 参数**规定当 load() 方法完成后所要**允许的回调函数**。回调函数可以设置不同的参数：

responseTxt - 包含调用成功时的结果内容；

statusTXT - 包含调用的状态；

xhr - 包含 XMLHttpRequest 对象；

下面的例子会在 load() 方法完成后显示一个提示框。如果 load() 方法已成功，则显示“外部内容加载成功！”，而如果失败，则显示错误消息：



## jQuery的GET与POST方法

jQuery get() 和 post() 方法用于通过 **HTTP GET 或 POST** 请求从服务器请求数据。

两种在客户端和服务器端进行请求-响应的常用方法是：GET 和 POST。

**GET** - 从指定的资源请求数据

**POST** - 向指定的资源提交要处理的数据

GET 基本上用于**从服务器获得（取回）数据**。注释：GET 方法可能返回缓存数据。

POST 也可用于从服务器获取数据。不过，POST 方法不会缓存数据，并且常用于连同请求一起发送数据。

## jQuery $.get() 方法

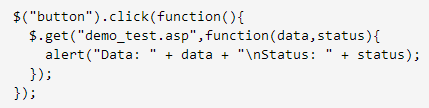
**$.get() 方法**通过 HTTP GET 请求从服务器上请求数据。

**$.get(URL,callback);**

**必需的 URL 参数**规定您希望请求的 URL；

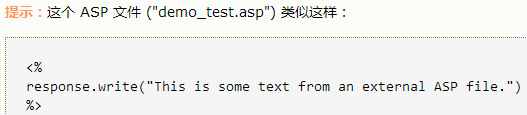
**可选的 callback 参数**是请求成功后所执行的函数名。

下面的例子使用 $.get() 方法从服务器上的一个文件中取回数据：



$.get() 的第一个参数是我们希望请求的 URL（"demo\_test.asp"）。

第二个参数是**回调函数**。第一个回调参数存有被请求页面的内容，第二个回调参数存有请求的状态。



## jQuery $.post() 方法

**$.post() 方法**通过 HTTP POST 请求从服务器上请求数据。

语法：

**$.post(URL,data,callback);**

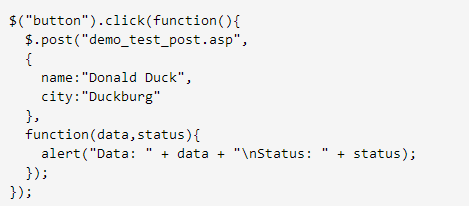
必需的 URL 参数规定您希望请求的 URL。

可选的 **data 参数**规定连同请求发送的数据。

可选的 **callback 参数**是请求成功后所执行的函数名。

示例：

下面的例子使用 **$.post()** 连同请求一起发送数据：

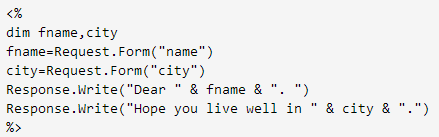


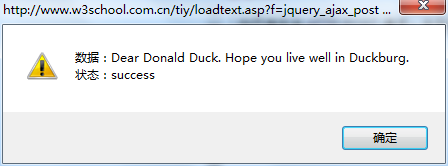
$.post() 的第一个参数是我们希望请求的 URL ("demo\_test\_post.asp")。

然后我们连同请求（name 和 city）一起发送数据。

"demo\_test\_post.asp" 中的 ASP 脚本读取这些参数，对它们进行处理，然后返回结果。

第三个参数是**回调函数。**第一个回调参数存有被请求页面的内容，而第二个参数存有请求的状态。





# jQuery的ajax方法

<http://www.w3school.com.cn/jquery/ajax_ajax.asp>

## 定义

**ajax() 方法通过 HTTP 请求加载远程数据**。

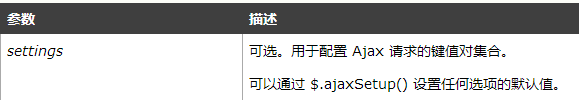
该方法是 **jQuery 底层 AJAX 实现**。简单易用的高层实现见 **$.get, $.post** 等。**$.ajax() 返回其创建的 XMLHttpRequest 对象**。大多数情况下你无需直接操作该函数，除非你需要操作不常用的选项，以获得更多的灵活性。

最简单的情况下，$.ajax() 可以不带任何参数直接使用。

注意：所有的选项都可以通过 **$.ajaxSetup() 函数**来全局设置。

## 语法

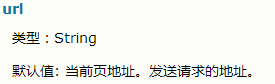




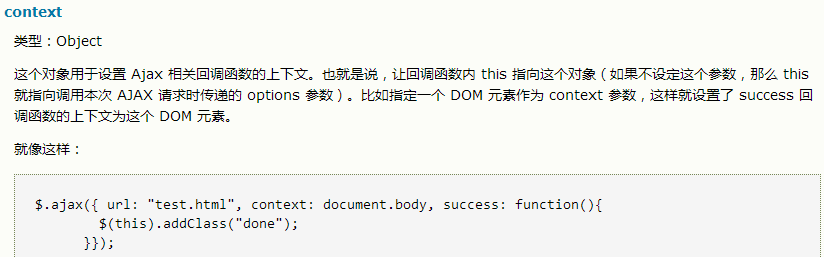
## 参数：重点

常用的有：

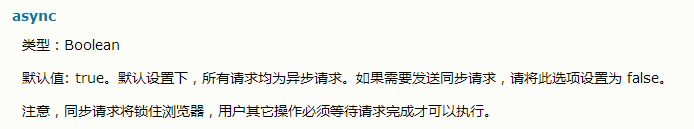
### url



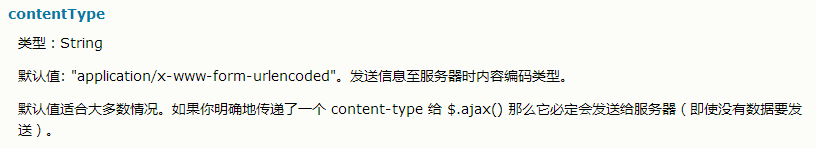
### context



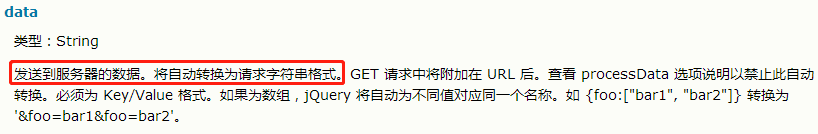
### async：是否异步



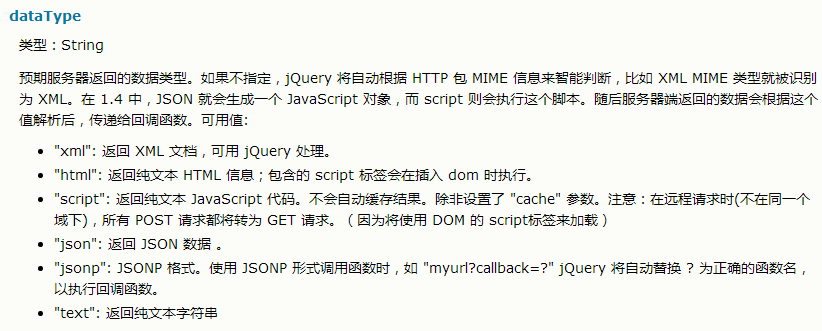
### contentType



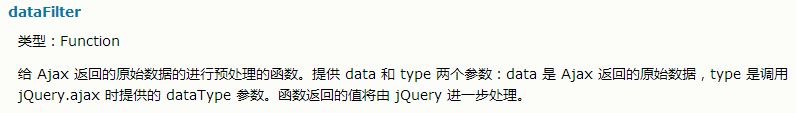
### data



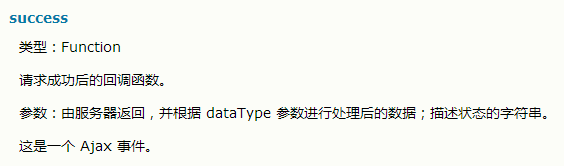
### dataType



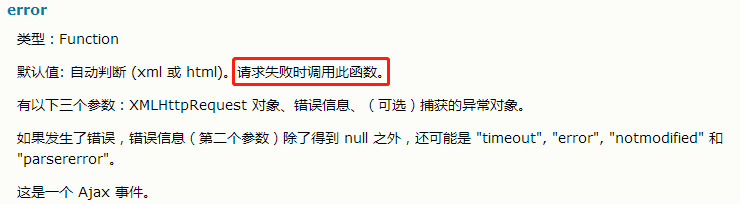
### dataFilter



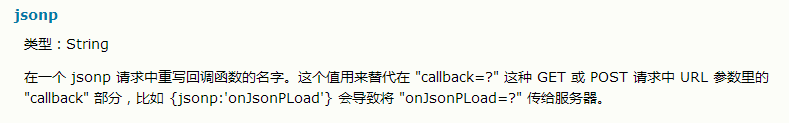
### success



### error



### jsonp



## 回调函数



## 数据类型

**$.ajax() 函数**依赖服务器提供的信息来处理返回的数据。如果服务器报告说返回的数据是 XML，那么返回的结果就可以用普通的 XML 方法或者 jQuery 的选择器来遍历。如果见得到其他类型，比如 HTML，则数据就以文本形式来对待。

通过 dataType 选项还可以指定其他不同数据处理方式。除了单纯的 XML，还可以指定 html、json、jsonp、script 或者 text。

其中，text 和 xml 类型返回的数据不会经过处理。数据仅仅简单的将 XMLHttpRequest 的 responseText 或 responseHTML 属性传递给 success 回调函数。

**注意：**我们必须确保网页服务器报告的 MIME 类型与我们选择的 dataType 所匹配。比如说，XML的话，服务器端就必须声明 text/xml 或者 application/xml 来获得一致的结果。

如果指定为 html 类型，任何内嵌的 JavaScript 都会在 HTML 作为一个字符串返回之前执行。类似地，指定 script 类型的话，也会先执行服务器端生成 JavaScript，然后再把脚本作为一个文本数据返回。

如果指定为 json 类型，则会把获取到的数据作为一个 JavaScript 对象来解析，并且把构建好的对象作为结果返回。为了实现这个目的，它首先尝试使用 JSON.parse()。如果浏览器不支持，则使用一个函数来构建。

JSON 数据是一种能很方便通过 JavaScript 解析的结构化数据。如果获取的数据文件存放在远程服务器上（域名不同，也就是跨域获取数据），则需要使用 jsonp 类型。使用这种类型的话，会创建一个查询字符串参数 callback=? ，这个参数会加在请求的 URL 后面。服务器端应当在 JSON 数据前加上回调函数名，以便完成一个有效的 JSONP 请求。如果要指定回调函数的参数名来取代默认的 callback，可以通过设置 $.ajax() 的 jsonp 参数。

**注意：**JSONP 是 JSON 格式的扩展。它要求一些服务器端的代码来检测并处理查询字符串参数。

如果指定了 script 或者 jsonp 类型，那么当从服务器接收到数据时，实际上是用了 <script> 标签而不是 XMLHttpRequest 对象。这种情况下，$.ajax() 不再返回一个 XMLHttpRequest 对象，并且也不会传递事件处理函数，比如 beforeSend。

## 发送数据到服务器

默认情况下，Ajax 请求使用 GET 方法。如果要使用 POST 方法，可以设定 type 参数值。这个选项也会影响 data 选项中的内容如何发送到服务器。

data 选项既可以包含一个查询字符串，比如 key1=value1&key2=value2 ，也可以是一个映射，比如 {key1: 'value1', key2: 'value2'} 。如果使用了后者的形式，则数据再发送器会被转换成查询字符串。这个处理过程也可以通过设置 processData 选项为 false 来回避。如果我们希望发送一个 XML 对象给服务器时，这种处理可能并不合适。并且在这种情况下，我们也应当改变 contentType 选项的值，用其他合适的 MIME 类型来取代默认的 **application/x-www-form-urlencoded** 。

## 高级选项

global 选项用于阻止响应注册的回调函数，比如 .ajaxSend，或者 ajaxError，以及类似的方法。这在有些时候很有用，比如发送的请求非常频繁且简短的时候，就可以在 ajaxSend 里禁用这个。

如果服务器需要 HTTP 认证，可以使用用户名和密码可以通过 username 和 password 选项来设置。

Ajax 请求是限时的，所以错误警告被捕获并处理后，可以用来提升用户体验。请求超时这个参数通常就保留其默认值，要不就通过 jQuery.ajaxSetup 来全局设定，很少为特定的请求重新设置 timeout 选项。

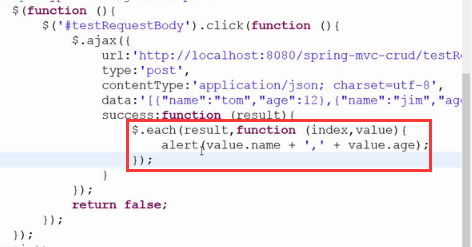
默认情况下，请求总会被发出去，但浏览器有可能从它的缓存中调取数据。要禁止使用缓存的结果，可以设置 cache 参数为 false。如果希望判断数据自从上次请求后没有更改过就报告出错的话，可以设置 ifModified 为 true。

scriptCharset 允许给 <script> 标签的请求设定一个特定的字符集，用于 script 或者 jsonp 类似的数据。当脚本和页面字符集不同时，这特别好用。

Ajax 的第一个字母是 asynchronous 的开头字母，这意味着所有的操作都是并行的，完成的顺序没有前后关系。$.ajax() 的 async 参数总是设置成true，这标志着在请求开始后，其他代码依然能够执行。强烈不建议把这个选项设置成 false，这意味着所有的请求都不再是异步的了，这也会导致浏览器被锁死。

$.ajax 函数返回它创建的 XMLHttpRequest 对象。通常 jQuery 只在内部处理并创建这个对象，但用户也可以通过 xhr 选项来传递一个自己创建的 xhr 对象。返回的对象通常已经被丢弃了，但依然提供一个底层接口来观察和操控请求。比如说，调用对象上的 .abort() 可以在请求完成前挂起请求。

## 示例



# jQuery Ajax 操作函数（方法）

jQuery 库拥有**完整的 Ajax 兼容套件**。其中的函数和方法允许我们在不刷新浏览器的情况下从服务器加载数据。



