**Ajax**

学习要点：

1.Ajax概述

2.load()方法

3.$.get()和$.post()

4.$.getScript()和$.getJSON()

5.$.ajax()方法

Ajax全称为：“Asynchronous JavaScript and XML”（异步 JavaScript和 XML），

它并不是 JavaScript的一种单一技术，而是利用了一系列交互式网页应用相关的技术所形

成的结合体。使用 Ajax，我们可以无刷新状态更新页面，并且实现异步提交，提升了用户

体验。

一．Ajax概述

Ajax这个概念是由 Jesse James Garrett在 2005年发明的。它本身不是单一技术，是一串

技术的集合，主要有：

1.JavaScript，通过用户或其他与浏览器相关事件捕获交互行为；

2.XMLHttpRequest对象，通过这个对象可以在不中断其它浏览器任务的情况下向服务

器发送请求；

3.服务器上的文件，以 XML、HTML或 JSON格式保存文本数据；

4.其它 JavaScript，解释来自服务器的数据（比如 PHP从 MySQL获取的数据）并将其

呈现到页面上。

由于 Ajax包含众多特性，优势与不足也非常明显。优势主要以下几点：

1.不需要插件支持（一般浏览器且默认开启 JavaScript即可）；

2.用户体验极佳（不刷新页面即可获取可更新的数据）；

3.提升 Web程序的性能（在传递数据方面做到按需放松，不必整体提交）；

4.减轻服务器和带宽的负担（将服务器的一些操作转移到客户端）；

而 Ajax的不足由以下几点：

1.不同版本的浏览器度 XMLHttpRequest对象支持度不足(比如 IE5之前)；

2.前进、后退的功能被破坏（因为 Ajax永远在当前页，不会几率前后页面）；

3.搜索引擎的支持度不够（因为搜索引擎爬虫还不能理解 JS引起变化数据的内容）；

4.开发调试工具缺乏（相对于其他语言的工具集来说，JS或 Ajax调试开发少的可怜）。

异步和同步

使用 Ajax最关键的地方，就是实现异步请求、接受响应及执行回调。那么异步与同步

有什么区别呢？我们普通的 Web程序开发基本都是同步的，意为执行一段程序才能执行下

一段，类似电话中的通话，一个电话接完才能接听下个电话；而异步可以同时执行多条任务，

感觉有多条线路，类似于短信，不会因为看一条短信而停止接受另一条短信。Ajax也可以

使用同步模式执行，但同步的模式属于阻塞模式，这样会导致多条线路执行时又必须一条一

条执行，会让 Web页面出现假死状态，所以，一般 Ajax大部分采用异步模式。

二．load()方法

jQuery对 Ajax做了大量的封装，我们使用起来也较为方便，不需要去考虑浏览器兼容

性。对于封装的方式，jQuery采用了三层封装：最底层的封装方法为：$.ajax()，而通过这

层封装了第二层有三种方法：$.load()、$.get()和$.post()，最高层是$.getScript()和$.getJSON()

方法。

.load()方法可以参数三个参数：url(必须，请求 html文件的 url地址，参数类型为 String)、

data(可选，发送的 key/value数据，参数类型为 Object)、callback(可选，成功或失败的回调

函数，参数类型为函数 Function)。

如果想让 Ajax异步载入一段 HTML内容，我们只需要一个 HTML请求的 url即可。

//HTML

<input type="button" value="异步获取数据" />

<div id="box"></div>

//jQuery

$('input').click(function () {

$('#box').load('test.html');

});

如果想对载入的 HTML进行筛选，那么只要在 url参数后面跟着一个选择器即可。

//带选择器的 url

$('input').click(function () {

$('#box').load('test.html .my');

});

如果是服务器文件，比如.php。一般不仅需要载入数据，还需要向服务器提交数据，那

么我们就可以使用第二个可选参数 data。向服务器提交数据有两种方式：get和 post。

//不传递 data，则默认 get方式

$('input').click(function () {

$('#box').load('test.php?url=ycku');

});

//get方式接受的 PHP

<?php

if ($\_GET['url'] == 'ycku') {

echo '百度';

} else {

echo '其他网站';

}

?>

//传递 data，则为 post方式

$('input').click(function () {

$('#box').load('test.php', {

url : 'ycku'

});

});

//post方式接受的 PHP

<?php

if ($\_POST['url'] == 'ycku') {

echo '百度';

} else {

echo '其他网站';

}

?>

在 Ajax数据载入完毕之后，就能执行回调函数 callback，也就是第三个参数。回调函数

也可以传递三个可选参数：responseText（请求返回）、textStatus（请求状态)、XMLHttpRequest

（XMLHttpRequest对象）。

$('input').click(function () {

$('#box').load('test.php', {

url : 'ycku'

}, function (response, status, xhr) {

alert('返回的值为：' + response + '，状态为：' + status + '，

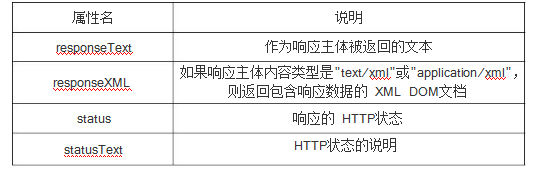
状态是：' + xhr.statusText);

});

});

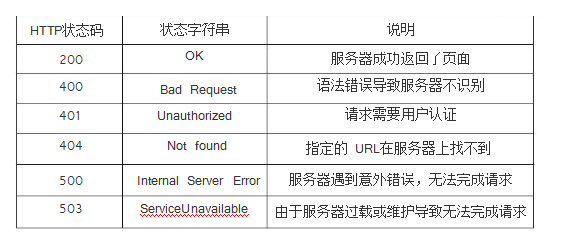
注意：status得到的值，如果成功返回数据则为：success，否则为：error。XMLHttpRequest

对象属于 JavaScript范畴，可以调用一些属性如下：



如果成功返回数据，那么 xhr对象的 statusText属性则返回'OK'字符串。除了'OK'的状态

字符串，statusText属性还提供了一系列其他的值，如下：



三．$.get()和$.post()

.load()方法是局部方法，因为他需要一个包含元素的 jQuery对象作为前缀。而$.get()和

$.post()是全局方法，无须指定某个元素。对于用途而言，.load()适合做静态文件的异步获取，

而对于需要传递参数到服务器页面的，$.get()和$.post()更加合适。

$.get()方法有四个参数，前面三个参数和.load()一样，多了一个第四参数 type，即服务

器返回的内容格式：包括 xml、html、script、json、jsonp和 text。第一个参数为必选参数，

后面三个为可选参数。

//使用$.get()异步返回 html类型

$('input').click(function () {

$.get('test.php', {

url : 'ycku'

}, function (response, status, xhr) {

if (status == 'success') {

$('#box').html(response);

}

})

//type自动转为 html

});

注意：第四参数 type是指定异步返回的类型。一般情况下 type参数是智能判断，并不

需要我们主动设置，如果主动设置，则会强行按照指定类型格式返回。

//使用$.get()异步返回 xml

$('input').click(function () {

$.get('test.xml', function (response, status, xhr) {

$('#box').html($(response).find('root').find('url').text());

});

//type自动转为 xml

});

注意：如果载入的是 xml文件，type会智能判断。如果强行设置 html类型返回，则会

把 xml文件当成普通数据全部返回，而不会按照 xml格式解析数据。

//使用$.get()异步返回 json

$.get('test.json', function (response, status, xhr) {

alert(response[0].url);

});

$.post()方法的使用和$.get()基本上一致，他们之间的区别也比较隐晦，基本都是背后的

不同，在用户使用上体现不出。具体区别如下：

1.GET请求是通过 URL提交的，而 POST请求则是 HTTP消息实体提交的；

2.GET提交有大小限制（2KB），而 POST方式不受限制；

3.GET方式会被缓存下来，可能有安全性问题，而 POST没有这个问题；

4.GET方式通过$\_GET[]获取，POST方式通过$\_POST[]获取。

//使用$.post()异步返回 html

$.post('test.php', {

url : 'ycku'

}, function (response, status, xhr) {

$('#box').html(response);

});

四．$.getScript()和$.getJSON()

jQuery提供了一组用于特定异步加载的方法：$.getScript()，用于加载特定的 JS文件；

$.getJSON()，用于专门加载 JSON文件。

有时我们希望能够特定的情况再加载 JS文件，而不是一开始把所有 JS文件都加载了，

这时课时使用$.getScript()方法。

//点击按钮后再加载 JS文件

$('input').click(function () {

$.getScript('test.js');

});

$.getJSON()方法是专门用于加载 JSON文件的，使用方法和之前的类似。

$('input').click(function () {

$.getJSON('test.json', function (response, status, xhr) {

alert(response[0].url);

});

});

五．$.ajax()

$.ajax()是所有 ajax方法中最底层的方法，所有其他方法都是基于$.ajax()方法的封装。

这个方法只有一个参数，传递一个各个功能键值对的对象。





//$.ajax使用

$('input').click(function () {

$.ajax({

type : 'POST',

//这里可以换成 GET

url : 'test.php',

data : {

url : 'ycku'

},

success : function (response, stutas, xhr) {

$('#box').html(response);

}

});

});

注意：对于 data属性，如果是 GET模式，可以使用三种之前说所的三种形式。如果是

POST模式可以使用之前的两种形式。

六．表单序列化

Ajax用的最多的地方莫过于表单操作，而传统的表单操作是通过 submit提交将数据传

输到服务器端。如果使用 Ajax异步处理的话，我们需要将每个表单元素逐个获取才方能提

交。这样工作效率就大大降低。

//常规形式的表单提交

$('form input[type=button]').click(function () {

$.ajax({

type : 'POST',

url : 'test.php',

data : {

user : $('form input[name=user]').val(),

email : $('form input[name=email]').val()

},

success : function (response, status, xhr) {

alert(response);

}

});

});

使用表单序列化方法.serialize()，会智能的获取指定表单内的所有元素。这样，在面对

大量表单元素时，会把表单元素内容序列化为字符串，然后再使用 Ajax请求。

//使用.serialize()序列化表单内容

$('form input[type=button]').click(function () {

$.ajax({

type : 'POST',

url : 'test.php',

data : $('form').serialize(),

success : function (response, status, xhr) {

alert(response);

}

});

});

.serialize()方法不但可以序列化表单内的元素，还可以直接获取单选框、复选框和下拉

列表框等内容。

//使用序列化得到选中的元素内容

$(':radio').click(function () {

$('#box').html(decodeURIComponent($(this).serialize()));

});

除了.serialize()方法，还有一个可以返回 JSON数据的方法：.serializeArray()。这个方法

可以直接把数据整合成键值对的 JSON对象。

$(':radio').click(function () {

console.log($(this).serializeArray());

var json = $(this).serializeArray();

$('#box').html(json[0].value);

});

有时，我们可能会在同一个程序中多次调用$.ajax()方法。而它们很多参数都相同，这

个时候我们课时使用 jQuery提供的$.ajaxSetup()请求默认值来初始化参数。

$('form input[type=button]').click(function () {

$.ajaxSetup({

type : 'POST',

url : 'test.php',

data : $('form').serialize()

});

$.ajax({

success : function (response, status, xhr) {

alert(response);

}

});

});

在使用 data属性传递的时候，如果是以对象形式传递键值对，可以使用$.param()方法

将对象转换为字符串键值对格式。

var obj = {a : 1, b : 2, c : 3};

var form = $.param(obj);

alert(form);

注意：使用$.param()将对象形式的键值对转为 URL地址的字符串键值对，可以更加稳

定准确的传递表单内容。因为有时程序对于复杂的序列化解析能力有限，所以直接传递 obj

对象要谨慎。