《JQuery部分》

**学习JQuery需要具备的基础知识**

在您开始学习 jQuery 之前，您应该对以下知识有基本的了解：

HTML，CSS，JavaScript

那么，接下来就可以学习了

**一，什么是JQuery?**

jQuery是一个JavaScript函数库。

jQuery是一个轻量级的"写的少，做的多"的JavaScript库。

jQuery库包含以下功能：

1）HTML 元素选取，2）HTML 元素操作，

1. CSS 操作HTML 事件函数 4）JavaScript 特效和动画

5）HTML DOM 遍历和修改 6）AJAX 7）Utilities

**提示：** 除此之外，Jquery还提供了大量的插件。

**二，为什么使用 jQuery ？**

目前网络上有大量开源的 JS 框架, 但是 jQuery 是目前最流行的 JS 框架，而且提供了大量的扩展。

很多大公司都在使用 jQuery， 例如:

Google

Microsoft

IBM

Netflix

**注：**

|  |  |
| --- | --- |
|  | jQuery是否适用于所有浏览器? jQuery 团体知道JS在不同浏览器中存在着大量的兼容性问题，  目前jQuery兼容于所有主流浏览器, 包括Internet Explorer 6! |

## 三，网页中添加 jQuery

可以通过多种方法在网页中添加 jQuery。 您可以使用以下方法：

1）从 [jquery.com](http://jquery.com/download/) 下载 jQuery 库（直接引入）

2）从CDN 中载入 jQuery, 如从 Google 中加载 jQuery（通过线上的文件）

**四，JQuery的入口函数和js入口函数的区别**

<script>

//原生js，入口函数。页面上所有内容加载完毕，会执行。

//不仅文本加载完毕，而且图片也要加载完毕，在执行函数。

window.onload = function () {

alert(1);

}

//jquery的入口函数。 1.文档加载完毕，图片不加载的时候就可以执行这个函数。

$(document).ready(function () {

alert(1);

})

//jquery的入口函数。 2.文档加载完毕，图片不加载的时候就可以执行这个函数。

$(function () {

alert(1);

});

//jquery的入口函数。 3.文档加载完毕，图片加载完毕的时候在执行这个函数。

$(window).ready(function () {

alert(1);

})

</script>

**五，$是什么？有什么作用？**

$就是jquery对象，$()就是jQuery()，在里面可以传参数，作用就是获取元素

如下例子

$(".div1") 表示获取类名为div1的元素，例如获取<div class="div1"></div>

$(".div1").onclick表示类名为div1的div点击事件

jquery中$.，例如$.post()，$.get()，$.ajax()等这些都是jquery这个对象的方法

$ 即 jQuery对象,是一个Function对象,$()是调用这个function,得到的是

jQuery.prototype.init的一个实例,jQuery对象包含很多静态方法,可以通过

jQuery.即($.)调用,通过jQuery.fn(即jQuery.prototype)可以对jQuery.prototype.init的实例进行扩展,因为jQuery.prototype.init的原型引用的jQuery.prototype

$.是调用JQuery中对象，$("#id").是id元素的属性

**六，JQuery选择器**

jQuery 选择器允许您对 HTML 元素组或单个元素进行操作。

jQuery 选择器基于元素的 id、类、类型、属性、属性值等"查找"（或选择）HTML 元素。 它基于已经存在的 [CSS 选择器](https://www.runoob.com/cssref/css-selectors.html)，除此之外，它还有一些自定义的选择器。

jQuery 中所有选择器都以美元符号开头：$()。

## 元素选择器

jQuery 元素选择器基于元素名选取元素。

在页面中选取所有 <p> 元素:

$("p")

## #id 选择器

jQuery #id 选择器通过 HTML 元素的 id 属性选取指定的元素。

页面中元素的 id 应该是唯一的，所以您要在页面中选取唯一的元素需要通过 #id 选择器。

通过 id 选取元素语法如下：

$("#test")

## .class 选择器

jQuery 类选择器可以通过指定的 class 查找元素。

语法如下：

$(".test")

更多：

|  |  |  |
| --- | --- | --- |
| **语法** | **描述** |  |
| $("\*") | 选取所有元素 |  | |
| $(this) | 选取当前 HTML 元素 |  | |
| $("p.intro") | 选取 class 为 intro 的 <p> 元素 |  | |
| $("p:first") | 选取第一个 <p> 元素 |  | |
| $("ul li:first") | 选取第一个 <ul> 元素的第一个 <li> 元素 |  | |
| $("ul li:first-child") | 选取每个 <ul> 元素的第一个 <li> 元素 |  | |
| $("[href]") | 选取带有 href 属性的元素 |  | |
| $("a[target='\_blank']") | 选取所有 target 属性值等于 "\_blank" 的 <a> 元素 |  | |
| $("a[target!='\_blank']") | 选取所有 target 属性值不等于 "\_blank" 的 <a> 元素 |  | |
| $(":button") | 选取所有 type="button" 的 <input> 元素 和 <button> 元素 |  | |
| $("tr:even") | 选取偶数位置的 <tr> 元素 |  | |
| $("tr:odd") | 选取奇数位置的 <tr> 元素 |  |

# 七. jQuery 事件

JQuery为事件处理特别设计

## 什么是事件？

页面对不同访问者的响应叫做事件。（学习js的时候已经了解过事件）

事件处理程常见 DOM 事件：

|  |  |  |  |
| --- | --- | --- | --- |
| **鼠标事件** | **键盘事件** | **表单事件** | **文档/窗口事件** |
| [click](https://www.runoob.com/jquery/event-click.html) | [keypress](https://www.runoob.com/jquery/event-keypress.html) | [submit](https://www.runoob.com/jquery/event-submit.html) | [load](https://www.runoob.com/jquery/event-load.html) |
| [dblclick](https://www.runoob.com/jquery/event-dblclick.html) | [keydown](https://www.runoob.com/jquery/event-keydown.html) | [change](https://www.runoob.com/jquery/event-change.html) | [resize](https://www.runoob.com/jquery/event-resize.html) |
| [mouseenter](https://www.runoob.com/jquery/event-mouseenter.html) | [keyup](https://www.runoob.com/jquery/event-keyup.html) | [focus](https://www.runoob.com/jquery/event-focus.html) | [scroll](https://www.runoob.com/jquery/event-scroll.html) |
| [mouseleave](https://www.runoob.com/jquery/event-mouseleave.html) | [hover](https://www.runoob.com/jquery/event-hover.html) | [blur](https://www.runoob.com/jquery/event-blur.html) | [unload](https://www.runoob.com/jquery/event-unload.html) |

**jQuery 事件方法语法**

在 jQuery 中，大多数 DOM 事件都有一个等效的 jQuery 方法。

页面中指定一个点击事件：

$("p").click();

下一步是定义什么时间触发事件。您可以通过一个事件函数实现：

$("p").click(function(){ // 动作触发后执行的代码!! });

接下来展示一下常用的事件：

### $(document).ready()

$(document).ready() 方法允许我们在文档完全加载完后执行函数。该事件方法在 [jQuery 语法](https://www.runoob.com/jquery/jquery-syntax.html) 章节中已经提到过。

### click()

click() 方法是当按钮点击事件被触发时会调用一个函数。

该函数在用户点击 HTML 元素时执行。

在下面的实例中，当点击事件在某个 <p> 元素上触发时，隐藏当前的 <p> 元素：

## 实例

$("p").click(function(){ $(this).hide(); })

### dblclick()

当双击元素时，会发生 dblclick 事件。

dblclick() 方法触发 dblclick 事件，或规定当发生 dblclick 事件时运行的函数：

## 实例

$("p").dblclick(function(){ $(this).hide(); });

### mouseenter()

当鼠标指针穿过元素时，会发生 mouseenter 事件。

mouseenter() 方法触发 mouseenter 事件，或规定当发生 mouseenter 事件时运行的函数：

## 实例

$("#p1").mouseenter(function(){ alert('您的鼠标移到了 id="p1" 的元素上!'); });

### mouseleave()

当鼠标指针离开元素时，会发生 mouseleave 事件。

mouseleave() 方法触发 mouseleave 事件，或规定当发生 mouseleave 事件时运行的函数：

## 实例

$("#p1").mouseleave(function(){ alert("再见，您的鼠标离开了该段落。"); });

### mousedown()

当鼠标指针移动到元素上方，并按下鼠标按键时，会发生 mousedown 事件。

mousedown() 方法触发 mousedown 事件，或规定当发生 mousedown 事件时运行的函数：

## 实例

$("#p1").mousedown(function(){ alert("鼠标在该段落上按下！"); });

### mouseup()

当在元素上松开鼠标按钮时，会发生 mouseup 事件。

mouseup() 方法触发 mouseup 事件，或规定当发生 mouseup 事件时运行的函数：

## 实例

$("#p1").mouseup(function(){ alert("鼠标在段落上松开。"); });

### hover()

hover()方法用于模拟光标悬停事件。

当鼠标移动到元素上时，会触发指定的第一个函数(mouseenter);当鼠标移出这个元素时，会触发指定的第二个函数(mouseleave)。

## 实例

$("#p1").hover( function(){ alert("你进入了 p1!"); }, function(){ alert("拜拜! 现在你离开了 p1!"); } )

### focus()

当元素获得焦点时，发生 focus 事件。

当通过鼠标点击选中元素或通过 tab 键定位到元素时，该元素就会获得焦点。

focus() 方法触发 focus 事件，或规定当发生 focus 事件时运行的函数：

## 实例

$("input").focus(function(){ $(this).css("background-color","#cccccc"); });

### blur()

当元素失去焦点时，发生 blur 事件。

blur() 方法触发 blur 事件，或规定当发生 blur 事件时运行的函数：

## 实例

$("input").blur(function(){ $(this).css("background-color","#ffffff"); });

**八．jQuery 的一些效果**

**显示隐藏**

**jQuery hide() 和 show()**

通过 jQuery，您可以使用 hide() 和 show() 方法来隐藏和显示 HTML 元素：

**实例**

$("#hide").click(function(){ $("p").hide(); }); $("#show").click(function(){ $("p").show(); });

下面的例子演示了带有 speed 参数的 hide() 方法，带有动画，并使用回调函数：

回调函数：（回调函数是一段可执行的代码段，它作为一个参数传递给其他的代码，其作用是在需要的时候方便调用这段（回调函数）代码。

在JavaScript中函数也是对象的一种，同样对象可以作为参数传递给函数，因此函数也可以作为参数传递给另外一个函数，这个作为参数的函数就是回调函数。回调函数例子如下

1. function add(num1, num2, callback){
2. var sum = num1 + num2;
3. callback(sum);
4. }
6. function print(num){
7. console.log(num);
8. }

）

）

）

## ）

## 实例

$(document).ready(function(){

$(".hidebtn").click(function(){

$("div").hide(1000,"linear",function(){

alert("Hide() 方法已完成!");

});

});

});

第二个参数是一个字符串，表示过渡使用哪种缓动函数。（注：jQuery自身提供"linear" 和 "swing"）。

以下实例在隐藏效果完全实现后回调函数:

**使用 callback 实例**

$("button").click(function(){

$("p").hide("slow",function(){

alert("段落现在被隐藏了");

}); });

以下实例没有回调函数，警告框会在隐藏效果完成前弹出：

**没有 callback(回调)**

$("button").click(function(){ $

("p").hide(1000);

alert("段落现在被隐藏了");

});

**显示+隐藏**

## jQuery toggle()

通过 jQuery，您可以使用 toggle() 方法来切换 hide() 和 show() 方法。

显示被隐藏的元素，并隐藏已显示的元素：

## 实例

$("button").click(function(){ $("p").toggle(); });

**语法:**

$(*selector*).toggle(*speed,callback*);

可选的 speed 参数规定隐藏/显示的速度，可以取以下值："slow"、"fast" 或毫秒。

可选的 callback 参数是隐藏或显示完成后所执行的函数名称。

**淡入淡出**

通过 jQuery，您可以实现元素的淡入淡出效果。

jQuery 拥有下面四种 fade 方法：fadeIn() ，fadeOut()，fadeToggle()，fadeTo()

1. jQuery fadeIn() 用于淡入已隐藏的元素。

（可选的 speed 参数规定效果的时长。它可以取以下值："slow"、"fast" 或毫秒。.

可选的 callback 参数是 fading 完成后所执行的函数名称。）

1. jQuery fadeOut() 方法用于淡出可见元素。（参数同上）
2. jQuery fadeToggle() 方法（参数同上）

jQuery fadeToggle() 方法可以在 fadeIn() 与 fadeOut() 方法之间进行切换。

如果元素已淡出，则 fadeToggle() 会向元素添加淡入效果。

如果元素已淡入，则 fadeToggle() 会向元素添加淡出效果。

1. jQuery fadeTo() 方法（参数同上）

jQuery fadeTo() 方法允许渐变为给定的不透明度（值介于 0 与 1 之间）。

**滑动**

## 5》jQuery 滑动方法

通过 jQuery，您可以在元素上创建滑动效果。

jQuery 拥有以下滑动方法：

1. slideDown() //展开效果
2. slideUp() //合住效果
3. slideToggle() //展开+合住效果

**动画**

**jQuery 动画 - animate() 方法**

jQuery animate() 方法用于创建自定义动画。

**语法：**

$(*selector*).animate({*params*}*,speed,callback*);

必需的 params 参数定义形成动画的 CSS 属性。

可选的 speed 参数规定效果的时长。它可以取以下值："slow"、"fast" 或毫秒。

可选的 callback 参数是动画完成后所执行的函数名称。

下面的例子演示 animate() 方法的简单应用。它把 <div> 元素往右边移动了 250 像素：

**实例**

$("button").click(function(){ $("div").animate({left:'250px'}); });

**停止动画**

**jQuery stop() 方法**

jQuery stop() 方法用于停止动画或效果，在它们完成之前。

stop() 方法适用于所有 jQuery 效果函数，包括滑动、淡入淡出和自定义动画。

**语法:**

$(*selector*).stop(*stopAll,goToEnd*);

可选的 stopAll 参数规定是否应该清除动画队列。默认是 false，即仅停止活动的动画，允许任何排入队列的动画向后执行。

可选的 goToEnd 参数规定是否立即完成当前动画。默认是 false。

因此，默认地，stop() 会清除在被选元素上指定的当前动画。

下面的例子演示 stop() 方法，不带参数：

**实例**

$("#stop").click(function(){ $("#panel").stop(); });

**JQuery的链式调用（**通过 jQuery，可以把动作/方法链接在一起，允许我们在一条语句中运行多个 jQuery 方法（在相同的元素上）。**）**

**提示：** 这样的话，浏览器就不必多次查找相同的元素。

如需链接一个动作，您只需简单地把该动作追加到之前的动作上。

下面的例子把 css()、slideUp() 和 slideDown() 链接在一起。"p1" 元素首先会变为红色，然后向上滑动，再然后向下滑动：

## 实例

$("#p1").css("color","red").slideUp(2000).slideDown(2000);

如果需要，我们也可以添加多个方法调用。

**提示：**当进行链接时，代码行会变得很长。不过，jQuery 语法不是很严格；您可以按照希望的格式来写，包含换行和缩进。

如下书写也可以很好地运行：

## 实例

$("#p1").css("color","red") .slideUp(2000) .slideDown(2000);

jQuery 会抛掉多余的空格，并当成一行长代码来执行上面的代码行。

**九．jQuery获取文本内容**

三个简单实用的用于 DOM 操作的 jQuery 方法：

text() - 设置或返回所选元素的文本内容（相当于js的InnerText）

html() - 设置或返回所选元素的内容（包括 HTML 标记）相当于js的InnerHTML）

val() - 设置或返回表单字段的值(相当于js的value)

**Jquery获取和设置属性**

下面的例子演示如何获得链接中 href 属性的值：

**实例**

$("button").click(function(){ alert($("#runoob").attr("href")); });

$("button").click(function(){ $("#runoob").attr("href", function(i,origValue){ return origValue + "/jquery"; }); });

**十，JQuery中DOM的增删**

**增**

我们将学习用于添加新内容的四个 jQuery 方法：

append() - 在被选元素的结尾插入内容

prepend() - 在被选元素的开头插入内容

after() - 在被选元素之后插入内容

before() - 在被选元素之前插入内容

**删**

## 删除元素/内容

如需删除元素和内容，一般可使用以下两个 jQuery 方法：

remove() - 删除被选元素（及其子元素）

empty() - 从被选元素中删除子元素

## 十一, jQuery 操作 CSS

jQuery 拥有若干进行 CSS 操作的方法。我们将学习下面这些：

addClass() - 向被选元素添加一个或多个类

removeClass() - 从被选元素删除一个或多个类

toggleClass() - 对被选元素进行添加/删除类的切换操作

hasClass() 方法检查被选元素是否包含指定的 class。

css() - 设置或返回样式属性($("p").css("background-color","yellow");)

**十二, JQuery获取宽高**

jQuery提供多个处理尺寸的重要方法：

clientX和clientY

pageX和pageY

offset().top和offset().left

width() 方法设置或返回元素的宽度（不包括内边距、边框或外边距）

height() 方法设置或返回元素的宽度（不包括内边距、边框或外边距）

innerWidth() 方法设置或返回元素的宽度（包括内边距）

innerHeight() 方法设置或返回元素的宽度（包括内边距）

outerWidth() 方法设置或返回元素的宽度（包括内边距、边框或外边距）

outerHeight() 方法设置或返回元素的宽度（不包括内边距、边框或外边距）

**十三, JQuery查找元素**parent()

parent()方法返回被选元素的直接父元素。

parents()方法返回被选元素的所有祖先元素，它一路向上直到文档的根元素 (<html>)。

parentsUntil()方法返回介于两个给定元素之间的所有祖先元素。（$(document).ready(function(){ $("span").parentsUntil("div"); });）

children() 方法返回被选元素的所有直接子元素。

find() 方法返回被选元素的后代元素，一路向下直到最后一个后代。

siblings() 方法返回被选元素的所有兄弟元素。

next() 方法返回被选元素的下一个同胞元素。

nextAll() 方法返回被选元素的所有跟随的同胞元素。

nextUntil() 方法返回介于两个给定参数之间的所有跟随的同胞元素。

prev() 之前的一个元素

prevAll()

prevUntil()

first() 方法返回被选元素的首个元素。

last() 方法返回被选元素的最后一个元素。

eq() 方法返回被选元素中带有指定索引号的元素。

filter() 方法允许您规定一个标准。不匹配这个标准的元素会被从集合中删除，匹配的元素会被返回。

not() 方法返回不匹配标准的所有元素。

**提示：not() 方法与 filter() 相反。**

**JQuery的遍历方法**

1. each() 方法为每个匹配元素规定要运行的函数。
2. map()把每个元素通过函数传递到当前匹配集合中，生成包含返回值的新的 jQuery 对象。

**JQuery的AJAX**

## 什么是 AJAX？

AJAX = 异步 JavaScript 和 XML（Asynchronous JavaScript and XML）。

简短地说，在不重载整个网页的情况下，AJAX 通过后台加载数据，并在网页上进行显示

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。

|  |  |  |
| --- | --- | --- |
|  |  | **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 方法

AJAX 是一种与服务器交换数据的技术，可以在不重新载入整个页面的情况下更新网页的一部分。

## 定义和用法

ajax() 方法用于执行 AJAX（异步 HTTP）请求。

所有的 jQuery AJAX 方法都使用 ajax() 方法。该方法通常用于其他方法不能完成的请求。

**语法**

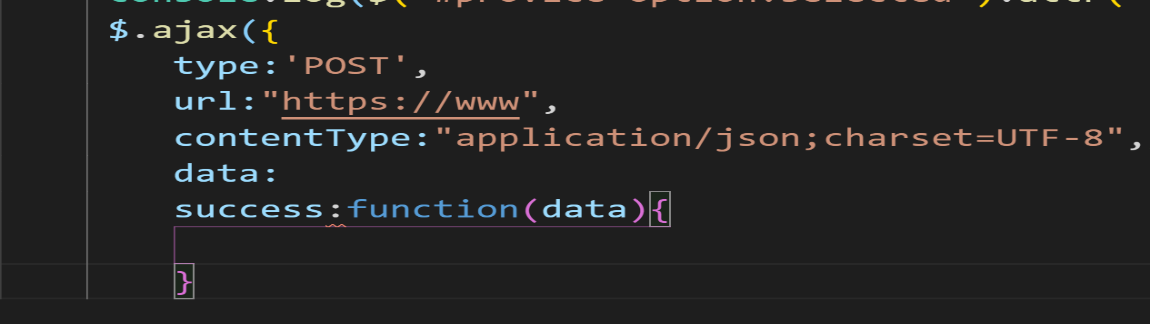
$.ajax(*{name:value, name:value, ... }*)

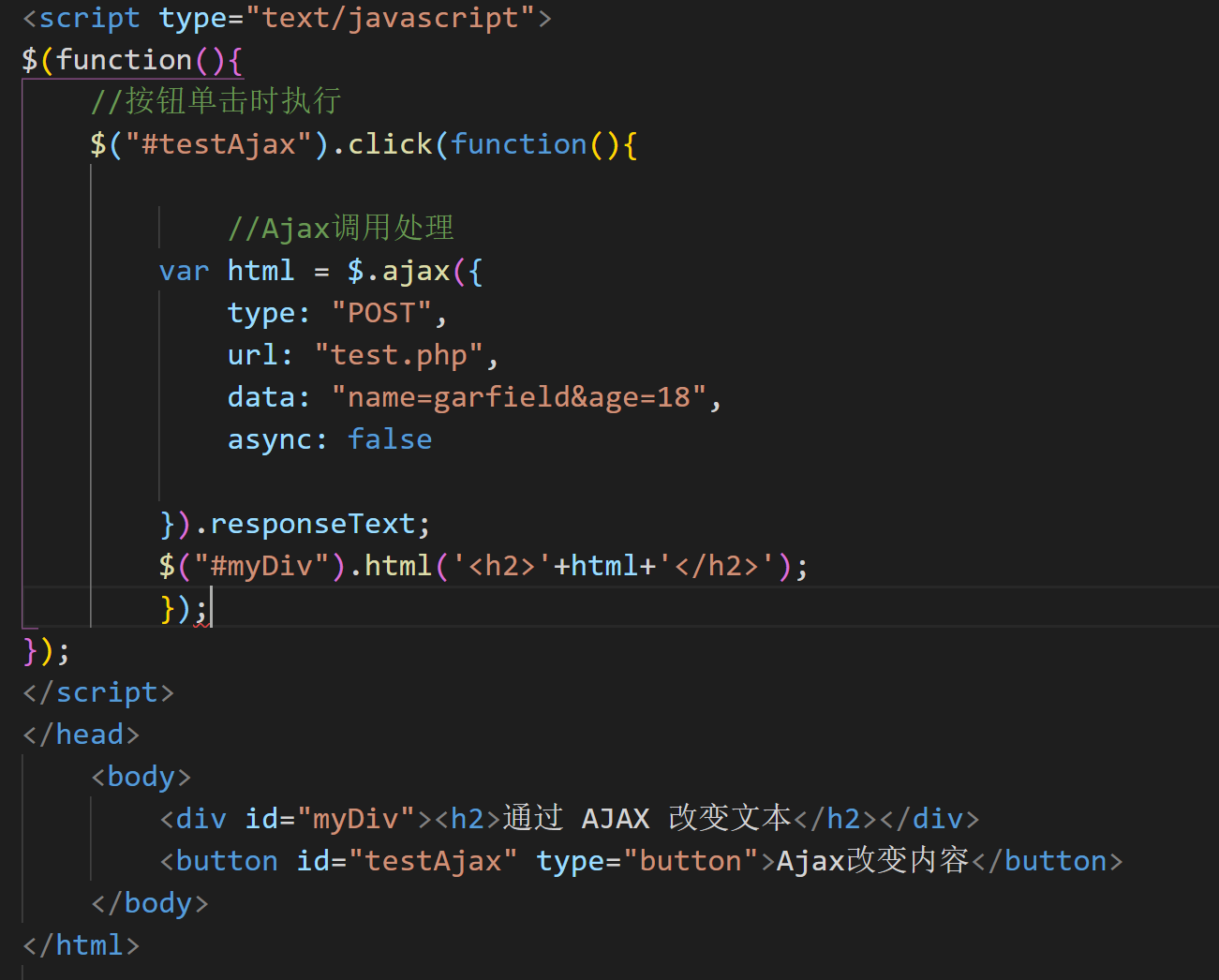
该参数规定 AJAX 请求的一个或多个名称/值对。

下面的表格中列出了可能的名称/值：

|  |  |
| --- | --- |
| **名称** | **值/描述** |
| async | 布尔值，表示请求是否异步处理。默认是 true。 |
| beforeSend(*xhr*) | 发送请求前运行的函数。 |
| cache | 布尔值，表示浏览器是否缓存被请求页面。默认是 true。 |
| complete(*xhr,status*) | 请求完成时运行的函数（在请求成功或失败之后均调用，即在 success 和 error 函数之后）。 |
| contentType | 发送数据到服务器时所使用的内容类型。默认是："application/x-www-form-urlencoded"。 |
| context | 为所有 AJAX 相关的回调函数规定 "this" 值。 |
| data | 规定要发送到服务器的数据。 |
| dataFilter(*data*,*type*) | 用于处理 XMLHttpRequest 原始响应数据的函数。 |
| dataType | 预期的服务器响应的数据类型。 |
| error(*xhr,status,error*) | 如果请求失败要运行的函数。 |
| global | 布尔值，规定是否为请求触发全局 AJAX 事件处理程序。默认是 true。 |
| ifModified | 布尔值，规定是否仅在最后一次请求以来响应发生改变时才请求成功。默认是 false。 |
| jsonp | 在一个 jsonp 中重写回调函数的字符串。 |
| jsonpCallback | 在一个 jsonp 中规定回调函数的名称。 |
| password | 规定在 HTTP 访问认证请求中使用的密码。 |
| processData | 布尔值，规定通过请求发送的数据是否转换为查询字符串。默认是 true。 |
| scriptCharset | 规定请求的字符集。 |
| success(*result,status,xhr*) | 当请求成功时运行的函数。 |
| timeout | 设置本地的请求超时时间（以毫秒计）。 |
| traditional | 布尔值，规定是否使用参数序列化的传统样式。 |
| type | 规定请求的类型（GET 或 POST）。 |
| url | 规定发送请求的 URL。默认是当前页面。 |
| username | 规定在 HTTP 访问认证请求中使用的用户名。 |
| xhr | 用于创建 XMLHttpRequest 对象的函数。 |

**下面是jquery AJAX的简单实例**



在这里需要重点说明的是：JSONP

# JSONP 教程

本章节我们将向大家介绍 JSONP 的知识。

Jsonp(JSON with Padding) 是 json 的一种"使用模式"，可以让网页从别的域名（网站）那获取资料，即跨域读取数据。

为什么我们从不同的域（网站）访问数据需要一个特殊的技术(JSONP )呢？这是因为同源策略。

同源策略，它是由Netscape提出的一个著名的安全策略，现在所有支持JavaScript 的浏览器都会使用这个策略。

## JSONP 应用

**讲JSONP之前，我们先介绍一下JSON**

1. **JSON**

注：之前提到过，JSON和XML都是一种数据格式，那么，两者之间有何异同？

## 与 XML 相同之处

* JSON 是纯文本
* JSON 具有"自我描述性"（人类可读）
* JSON 具有层级结构（值中存在值）
* JSON 可通过 JavaScript 进行解析
* JSON 数据可使用 AJAX 进行传输

## 与 XML 不同之处

* 没有结束标签
* 更短
* 读写的速度更快
* 能够使用内建的 JavaScript eval() 方法进行解析
* 使用数组
* 不使用保留字

## 为什么使用 JSON？

对于 AJAX 应用程序来说，JSON 比 XML 更快更易使用：

#### 使用 XML

* 读取 XML 文档
* 使用 XML DOM 来循环遍历文档
* 读取值并存储在变量中

#### 使用 JSON

* 读取 JSON 字符串
* 用 eval() 处理 JSON 字符串

**接下来我们重点讲解两个特别常用的方法,这两个方法并不是Jquery的方法，js都可以用**

**JSON.parse( )**  JSON 通常用于与服务端交换数据。

在接收服务器数据时一般是字符串。

我们可以使用 JSON.parse() 方法将数据转换为 JavaScript 对象。

解析完成后，我们就可以在网页上使用 JSON 数据了：

**JSON.stringify( )** JSON 通常用于与服务端交换数据。

在向服务器发送数据时一般是字符串。

我们可以使用 JSON.stringify() 方法将 JavaScript 对象转换为字符串。

我们也可以将 JavaScript 数组转换为 JSON 字符串：

**之前有了解过跨域？刚开始我们解决的方式是把浏览器的同源策略关掉，那么，还有一种方式：就是JSONP来解决跨域（后台加个请求头解决的跨域更简单，上线以后就不会有跨域问题存在）**

**切换到我们的主题：什么是JSONP?**

JSONP(JSON with Padding)是JSON的一种“使用模式”，可用于解决主流浏览器的跨域数据访问的问题。由于同源策略，一般来说位于 server1.example.com 的网页无法与不是 server1.example.com的服务器沟通，而 HTML 的<script> 元素是一个例外。利用 <script> 元素的这个开放策略，网页可以得到从其他来源动态产生的 JSON 资料，而这种使用模式就是所谓的 JSONP。用 JSONP 抓到的资料并不是 JSON，而是任意的JavaScript，用 JavaScript 直译器执行而不是用 JSON 解析器解析。

用通俗的话来说：jsonp 就是添加一个动态的script标签

### 在jQuery中如何通过JSONP来跨域获取数据

第一种方法是在ajax函数中设置dataType为'jsonp'：

$.ajax({

     dataType: 'jsonp',

        url: 'http://www.a.com/user?id=123',

      success: function(data){

             //处理data数据

        }

});

第二种方法是利用getJSON来实现，只要在地址中加上callback=?参数即可：

$.getJSON('http://www.a.com/user?id=123&callback=?', function(data){

//处理data数据

});

**$.ajax({  
   async:false,  
    url: http://跨域的dns/document!searchJSONResult.action,  
   type: "GET",  
    dataType: 'jsonp',  
   jsonp: 'jsoncallback',  
   data: qsData,  
   timeout: 5000,  
   beforeSend: function(){  
   //jsonp 方式此方法不被触发.原因可能是dataType如果指定为jsonp的话,就已经不是ajax事件了  
   },  
   success: function (json) {//客户端jquery预先定义好的callback函数,成功获取跨域服务器上的json数据后,会动态执行这个callback函数  
    if(json.actionErrors.length!=0){  
           alert(json.actionErrors);  
     }  
       genDynamicContent(qsData,type,json);  
   },  
    complete: function(XMLHttpRequest, textStatus){  
    $.unblockUI({ fadeOut: 10 });   
   },  
   error: function(xhr){  
    //jsonp 方式此方法不被触发.原因可能是dataType如果指定为jsonp的话,就已经不是ajax事件了  
    //请求出错处理  
    alert("请求出错(请检查相关度网络状况.)");  
   }  
});**