jQuery

1、理解什么是jQuery

2、掌握jQuery的使用

3、掌握jQuery的语法

4、掌握jQuery常用事件

5、掌握jQuery的效果

6、掌握jQuery对HTML的操作

7、掌握jQuery的遍历操作

8、掌握jQuery的Ajax操作

第一章 jQuery概述

1.1 jQuery简介

jQuery是一个快速、简洁的JavaScript框架，是继Prototype之后又一个优秀的JavaScript代码库（\*或JavaScript框架\*）。jQuery设计的宗旨是“write Less，Do More”，即倡导写更少的代码，做更多的事情。它封装JavaScript常用的功能代码，提供一种简便的JavaScript设计模式，优化HTML文档操作、事件处理、动画设计和Ajax交互。

jQuery的核心特性可以总结为：具有独特的链式语法和短小清晰的多功能接口；具有高效灵活的css选择器，并且可对CSS选择器进行扩展；拥有便捷的插件扩展机制和丰富的插件。jQuery兼容各种主流浏览器，如IE 6.0+、FF 1.5+、Safari 2.0+、Opera 9.0+等。

1.2 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：

- HTML 元素选取

- HTML 元素操作

- CSS 操作

- HTML 事件函数

- JavaScript 特效和动画

- HTML DOM 遍历和修改

- AJAX

- Utilities

第二章 jQuery安装

2.1 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：

- 从 [jquery.com](http://jquery.com/download/) 下载 jQuery 库

- 从 CDN 中载入 jQuery, 如从 Google 中加载 jQuery

有两个版本的 jQuery 可供下载：

- Production version - 用于实际的网站中，已被精简和压缩。

- Development version - 用于测试和开发（未压缩，是可读的代码）

以上两个版本都可以从 [jquery.com](http://jquery.com/download/) 中下载。

jQuery 库是一个 JavaScript 文件，您可以使用 HTML 的 <script> 标签引用它：

|  |
| --- |
| <head>  <script src="jquery-1.10.2.min.js"></script>  </head> |

当然你也可以使用其它网站的CDN：

2.2 百度 CDN

|  |
| --- |
| <head>  <script src="https://apps.bdimg.com/libs/jquery/2.1.4/jquery.min.js">  </script>  </head> |

2.3 新浪 CDN

|  |
| --- |
| <head>  <script src="http://lib.sinaapp.com/js/jquery/2.0.2/jquery-2.0.2.min.js">  </script>  </head> |

2.4 Google CDN

|  |
| --- |
| <head>  <script src="http://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js">  </script>  </head> |

2.5 Microsoft CDN

|  |
| --- |
| <head>  <script src="http://ajax.htmlnetcdn.com/ajax/jQuery/jquery-1.10.2.min.js">  </script>  </head> |

第三章 jQuery语法

Query 语法是通过选取 HTML 元素，并对选取的元素执行某些操作。

基础语法： \*\*$(selector).action()\*\*

- 美元符号定义 jQuery

- 选择符（selector）"查询"和"查找" HTML 元素

- jQuery 的 action() 执行对元素的操作

实例:

|  |
| --- |
| - $(this).hide() - 隐藏当前元素  - $("p").hide() - 隐藏所有 <p> 元素  - $("p.test").hide() - 隐藏所有 class="test" 的 <p> 元素  - $("#test").hide() - 隐藏所有 id="test" 的元素 |

3.1 jQuery选择器

3.1.1 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:

|  |
| --- |
| $(document).ready(function(){  $("button").click(function(){  $("p").hide();  });  }); |

3.1.2 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：

|  |
| --- |
| $(document).ready(function(){  $("button").click(function(){  $("#test").hide();  });  }); |

3.1.3 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：

|  |
| --- |
| $(document).ready(function(){  $("button").click(function(){  $(".test").hide();  });  }); |

3.2 jQuery事件

3.2.1 什么是事件？

页面对不同访问者的响应叫做事件。

事件处理程序指的是当 HTML 中发生某些事件时所调用的方法。

实例：

- 在元素上移动鼠标。

- 选取单选按钮

- 点击元素

在事件中经常使用术语"触发"（或"激发"）例如： "当您按下按键时触发 keypress 事件"。

常见 DOM 事件：

|  |  |  |  |
| --- | --- | --- | --- |
| 鼠标事件 | 键盘事件 | 表单事件 | 文档/窗口事件 |
| click | keypress | submit | load |
| mouseenter | keydown | change | scroll |
| mouseleave | keyup | focus | resize |
| hover |  | blur |  |

3.2.2 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(function(){

// 动作触发后执行的代码!!

});

也就是说，不传参数是点击，传参数是设置事件。

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

|  |
| --- |
| \*\*\*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)。  javascript  $("#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：  javascript  $("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：  javascript  $("input").blur(function(){  $(this).css("background-color","#ffffff");  }); |

第四章 jQuery效果

4.1 隐藏显示

\*\*\*hide()\*\*\*

您可以使用 hide() 将元素隐藏

javascript

$("#hide").click(function(){

$("p").hide();

});

\*\*\*show()\*\*\*

您可以使用show()将元素显示

$("#show").click(function(){

$("p").show();

});

\*\*\*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：

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

$("p").toggle();

});

事实上，这三种方法都是有两个参数的：

$(selector).hide(speed,callback);

$(selector).show(speed,callback);

$(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。

4.2 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()

\*\*\*jQuery fadeIn() 方法\*\*\*

jQuery fadeIn() 用于淡入已隐藏的元素。

$(selector).fadeIn(speed,callback);

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。

下面的例子演示了带有不同参数的 fadeIn() 方法：

|  |
| --- |
| $("button").click(function(){  $("#div1").fadeIn();  $("#div2").fadeIn("slow");  $("#div3").fadeIn(3000);  }); |

\*\*\*jQuery fadeOut() 方法\*\*\*

jQuery fadeOut() 方法用于淡出可见元素。

$(selector).fadeOut(speed,callback);

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。

下面的例子演示了带有不同参数的 fadeOut() 方法：

|  |
| --- |
| $("button").click(function(){  $("#div1").fadeOut();  $("#div2").fadeOut("slow");  $("#div3").fadeOut(3000);  }); |

\*\*\*jQuery fadeToggle() 方法\*\*\*

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。

javascript

$(selector).fadeToggle(speed,callback);

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。

下面的例子演示了带有不同参数的 fadeToggle() 方法：

|  |
| --- |
| $("button").click(function(){  $("#div1").fadeToggle();  $("#div2").fadeToggle("slow");  $("#div3").fadeToggle(3000);  }); |

\*\*\*jQuery fadeTo() 方法\*\*\*

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

javascript

$(selector).fadeTo(speed,opacity,callback);

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

fadeTo() 方法中必需的 opacity 参数将淡入淡出效果设置为给定的不透明度（值介于 0 与 1 之间）。

可选的 callback 参数是该函数完成后所执行的函数名称。

下面的例子演示了带有不同参数的 fadeTo() 方法：

|  |
| --- |
| $("button").click(function(){  $("#div1").fadeTo("slow",0.15);  $("#div2").fadeTo("slow",0.4);  $("#div3").fadeTo("slow",0.7);  }); |

4.3 滑动

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：

- slideDown()

- slideUp()

- slideToggle()

\*\*\*slideDown() 方法\*\*\*

$(selector).slideDown(speed,callback); 方法用于向下滑动元素

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 参数是滑动完成后所执行的函数名称。

下面的例子演示了 slideDown() 方法：

|  |
| --- |
| $("#flip").click(function(){  $("#panel").slideDown();  }); |

\*\*\*slideUp() 方法\*\*\*

$(selector).slideUp(speed,callback); 方法用于向上滑动元素。

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 参数是滑动完成后所执行的函数名称。

下面的例子演示了 slideUp() 方法：

|  |
| --- |
| $("#flip").click(function(){  $("#panel").slideUp();  }); |

\*\*\*slideToggle() 方法\*\*\*

jQuery slideToggle() 方法可以在 slideDown() 与 slideUp() 方法之间进行切换。

如果元素向下滑动，则 slideToggle() 可向上滑动它们。

如果元素向上滑动，则 slideToggle() 可向下滑动它们。

$(selector).slideToggle(speed,callback);

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 参数是滑动完成后所执行的函数名称。

下面的例子演示了 slideToggle() 方法：

|  |
| --- |
| $("#flip").click(function(){  $("#panel").slideToggle();  }); |

4.4 动画

4.4.1 animate() 方法

jQuery animate() 方法用于创建自定义动画。

javascript

$(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：

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

4.4.2 操作多个属性

请注意，生成动画的过程中可同时使用多个属性：

|  |
| --- |
| $("button").click(function(){  $("div").animate({  left:'250px',  opacity:'0.5',  height:'150px',  width:'150px'  });  }); |

\*\*可以用 animate() 方法来操作所有 CSS 属性吗？\*\*

是的，几乎可以！不过，需要记住一件重要的事情：当使用 animate() 时，必须使用 Camel 标记法书写所有的属性名，比如，必须使用 paddingLeft 而不是 padding-left，使用 marginRight 而不是 margin-right，等等。

同时，色彩动画并不包含在核心 jQuery 库中。

如果需要生成颜色动画，您需要从 [jquery.com](http://jquery.com/download/) 下载 [颜色动画](http://plugins.jquery.com/color/) 插件。

4.4.3 使用相对值

也可以定义相对值（该值相对于元素的当前值）。需要在值的前面加上 += 或 -=：

|  |
| --- |
| $("button").click(function(){  $("div").animate({  left:'250px',  height:'+=150px',  width:'+=150px'  });  }); |

4.4.4 预定义的值

您甚至可以把属性的动画值设置为 "show"、"hide" 或 "toggle"：

|  |
| --- |
| $("button").click(function(){  $("div").animate({  height:'toggle'  });  }); |

4.4.5 使用队列功能

默认地，jQuery 提供针对动画的队列功能。

这意味着如果您在彼此之后编写多个 animate() 调用，jQuery 会创建包含这些方法调用的"内部"队列。然后逐一运行这些 animate 调用。

|  |
| --- |
| $("button").click(function(){  var div=$("div");  div.animate({height:'300px',opacity:'0.4'},"slow");  div.animate({width:'300px',opacity:'0.8'},"slow");  div.animate({height:'100px',opacity:'0.4'},"slow");  div.animate({width:'100px',opacity:'0.8'},"slow");  }); |

4.5 停止动画

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。

$(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：

$("#stop").click(function(){

$("#panel").stop();

});

第五章 jQuery HTML

5.1 捕获

jQuery 拥有可操作 HTML 元素和属性的强大方法。

\*\*\*jQuery DOM 操作\*\*\*

jQuery 中非常重要的部分，就是操作 DOM 的能力。

jQuery 提供一系列与 DOM 相关的方法，这使访问和操作元素和属性变得很容易。

\*\*\*获得内容 - text()、html() 以及 val()\*\*\*

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

- html() - 设置或返回所选元素的内容（包括 HTML 标记）

- val() - 设置或返回表单字段的值

下面的例子演示如何通过 jQuery text() 和 html() 方法来获得内容：

|  |
| --- |
| $("#btn1").click(function(){  alert("Text: " + $("#test").text());  });  $("#btn2").click(function(){  alert("HTML: " + $("#test").html());  }); |

下面的例子演示如何通过 jQuery val() 方法获得输入字段的值：

$("#btn1").click(function(){

alert("值为: " + $("#test").val());

});

获取属性-attr()

jQuery attr() 方法用于获取属性值。

下面的例子演示如何获得链接中 href 属性的值：

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

alert($("#a1").attr("href"));

});

5.3 添加元素

\*\*添加新的 HTML 内容\*\*

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

\*\*jQuery append() 方法\*\*

jQuery append() 方法在被选元素的结尾插入内容。

$("p").append("追加文本");

\*\*jQuery prepend() 方法\*\*

jQuery prepend() 方法在被选元素的开头插入内容。

$("p").prepend("在开头追加文本");

\*\*通过 append() 和 prepend() 方法添加若干新元素\*\*

在上面的例子中，我们只在被选元素的开头/结尾插入文本/HTML。

不过，append() 和 prepend() 方法能够通过参数接收无限数量的新元素。可以通过 jQuery 来生成文本/HTML（就像上面的例子那样），或者通过 JavaScript 代码和 DOM 元素。

\*\*jQuery after() 和 before() 方法\*\*

jQuery after() 方法在被选元素之后插入内容。

jQuery before() 方法在被选元素之前插入内容。

$("img").after("在后面添加文本");

$("img").before("在前面添加文本");

\*\*通过 after() 和 before() 方法添加若干新元素\*\*

after() 和 before() 方法能够通过参数接收无限数量的新元素。可以通过 text/HTML、jQuery 或者 JavaScript/DOM 来创建新元素。

在下面的例子中，我们创建若干新元素。这些元素可以通过 text/HTML、jQuery 或者 JavaScript/DOM 来创建。然后我们通过 after() 方法把这些新元素插到文本中（对 before() 同样有效）：

|  |
| --- |
| function afterText()  {  var txt1="<b>I </b>"; // 使用 HTML 创建元素  var txt2=$("<i></i>").text("love "); // 使用 jQuery 创建元素  var txt3=document.createElement("big"); // 使用 DOM 创建元素  txt3.innerHTML="jQuery!";  $("img").after(txt1,txt2,txt3); // 在图片后添加文本  } |

5.4 删除元素

\*\*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

\*\*jQuery remove() 方法\*\*

jQuery remove() 方法删除被选元素及其子元素。

$("#div1").remove();

\*\*jQuery empty() 方法\*\*

jQuery empty() 方法删除被选元素的子元素。

$("#div1").empty();

\*\*过滤被删除的元素\*\*

jQuery remove() 方法也可接受一个参数，允许您对被删元素进行过滤。

该参数可以是任何 jQuery 选择器的语法。

下面的例子删除 class="italic" 的所有 <p> 元素：

$("p").remove(".italic");

5.5 CSS类

\*\*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

- css() - 设置或返回样式属性

\*\*实例样式表\*\*

下面的样式表将用于本页的所有例子：

css

.important

{

font-weight:bold;

font-size:xx-large;

}

.blue

{

color:blue;

}

\*\*jQuery addClass() 方法\*\*

下面的例子展示如何向不同的元素添加 class 属性。当然，在添加类时，您也可以选取多个元素：

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

$("h1,h2,p").addClass("blue");

$("div").addClass("important");

});

您也可以在 addClass() 方法中规定多个类：

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

$("body div:first").addClass("important blue");

});

\*\*jQuery removeClass() 方法\*\*

下面的例子演示如何在不同的元素中删除指定的 class 属性：

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

$("h1,h2,p").removeClass("blue");

});

\*\*jQuery toggleClass() 方法\*\*

下面的例子将展示如何使用 jQuery toggleClass() 方法。该方法对被选元素进行添加/删除类的切换操作：

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

$("h1,h2,p").toggleClass("blue");

});

5.6 css()方法

\*\*jQuery css() 方法\*\*

css() 方法设置或返回被选元素的一个或多个样式属性。

\*\*返回 CSS 属性\*\*

如需返回指定的 CSS 属性的值，请使用如下语法：

css("propertyname");

下面的例子将返回首个匹配元素的 background-color 值：

$("p").css("background-color");

\*\*设置 CSS 属性\*\*

如需设置指定的 CSS 属性，请使用如下语法：

css("propertyname","value");

下面的例子将为所有匹配元素设置 background-color 值：

$("p").css("background-color","yellow");

\*\*设置多个 CSS 属性\*\*

如需设置多个 CSS 属性，请使用如下语法：

css({"propertyname":"value","propertyname":"value",...});

下面的例子将为所有匹配元素设置 background-color 和 font-size：

$("p").css({"background-color":"yellow","font-size":"200%"});

5.7 尺寸

\*\*jQuery 尺寸方法\*\*

jQuery 提供多个处理尺寸的重要方法：

- width()

- height()

- innerWidth()

- innerHeight()

- outerWidth()

- outerHeight()

\*\*jQuery width() 和 height() 方法\*\*

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高度（不包括内边距、边框或外边距）。

下面的例子返回指定的div; 元素的宽度和高度：

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

var txt="";

txt+="div 的宽度是: " + $("#div1").width() + "</br>";

txt+="div 的高度是: " + $("#div1").height();

$("#div1").html(txt);

});

\*\*jQuery innerWidth() 和 innerHeight() 方法\*\*

innerWidth() 方法返回元素的宽度（包括内边距）。

innerHeight() 方法返回元素的高度（包括内边距）。

下面的例子返回指定的 &lt;div&gt; 元素的 inner-width/height：

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

var txt="";

txt+="div 宽度，包含内边距: " + $("#div1").innerWidth() + "</br>";

txt+="div 高度，包含内边距: " + $("#div1").innerHeight();

$("#div1").html(txt);

});

\*\*jQuery outerWidth() 和 outerHeight() 方法\*\*

outerWidth() 方法返回元素的宽度（包括内边距和边框）。

outerHeight() 方法返回元素的高度（包括内边距和边框）。

下面的例子返回指定的 &lt;div&gt; 元素的 outer-width/height：

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

var txt="";

txt+="div 宽度，包含内边距和边框: " + $("#div1").outerWidth() + "</br>";

txt+="div 高度，包含内边距和边框: " + $("#div1").outerHeight();

$("#div1").html(txt);

});

第六章 jQuery遍历

6.1 遍历

jQuery 遍历，意为"移动"，用于根据其相对于其他元素的关系来"查找"（或选取）HTML 元素。以某项选择开始，并沿着这个选择移动，直到抵达您期望的元素为止。

下图展示了一个家族树。通过 jQuery 遍历，您能够从被选（当前的）元素开始，轻松地在家族树中向上移动（祖先），向下移动（子孙），水平移动（同胞）。这种移动被称为对 DOM 进行遍历。

6.2 祖先

\*\*jQuery parent() 方法\*\*

parent() 方法返回被选元素的直接父元素。

该方法只会向上一级对 DOM 树进行遍历。

下面的例子返回每个 <span> 元素的的直接父元素：

$(document).ready(function(){

$("span").parent();

});

\*\*jQuery parents() 方法\*\*

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>)。

下面的例子返回所有 <span> 元素的所有祖先：

$(document).ready(function(){

$("span").parents();

});

6.3 后代

\*\*jQuery children() 方法\*\*

children() 方法返回被选元素的所有直接子元素。

该方法只会向下一级对 DOM 树进行遍历。

下面的例子返回每个 <div> 元素的所有直接子元素：

$(document).ready(function(){

$("div").children();

});

您也可以使用可选参数来过滤对子元素的搜索。

下面的例子返回类名为 "1" 的所有 <p> 元素，并且它们是 <div> 的直接子元素：

$(document).ready(function(){

$("div").children("p.1");

});

\*\*jQuery find() 方法\*\*

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。

下面的例子返回属于 <div> 后代的所有 <span> 元素：

$(document).ready(function(){

$("div").find("span");

});

下面的例子返回 <div> 的所有后代：

$(document).ready(function(){

$("div").find("\*");

});

6.4 同胞

\*\*jQuery siblings() 方法\*\*

siblings() 方法返回被选元素的所有同胞元素。

下面的例子返回 <h2> 的所有同胞元素：

$(document).ready(function(){

$("h2").siblings();

});

您也可以使用可选参数来过滤对同胞元素的搜索。

下面的例子返回属于 <h2> 的同胞元素的所有 <p> 元素：

$(document).ready(function(){

$("h2").siblings("p");

});

\*\*jQuery next() 方法\*\*

next() 方法返回被选元素的下一个同胞元素。

该方法只返回一个元素。

下面的例子返回 <h2> 的下一个同胞元素：

$(document).ready(function(){

$("h2").next();

});

\*\*jQuery nextAll() 方法\*\*

nextAll() 方法返回被选元素的所有跟随的同胞元素。

下面的例子返回 <h2> 的所有跟随的同胞元素：

$(document).ready(function(){

$("h2").nextAll();

});

\*\*jQuery nextUntil() 方法\*\*

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。

下面的例子返回介于 <h2> 与 <h6> 元素之间的所有同胞元素：

$(document).ready(function(){

$("h2").nextUntil("h6");

});

6.5 过滤

\*\*jQuery first() 方法\*\*

first() 方法返回被选元素的首个元素。

下面的例子选取首个 <div> 元素内部的第一个 <p> 元素：

$(document).ready(function(){

$("div p").first();

});

\*\*jQuery last() 方法\*\*

last() 方法返回被选元素的最后一个元素。

下面的例子选择最后一个 <div> 元素中的最后一个 <p> 元素：

$(document).ready(function(){

$("div p").last();

});

\*\*jQuery eq() 方法\*\*

eq() 方法返回被选元素中带有指定索引号的元素。

索引号从 0 开始，因此首个元素的索引号是 0 而不是 1。下面的例子选取第二个 <p> 元素（索引号 1）：

$(document).ready(function(){

$("p").eq(1);

});

\*\*jQuery filter() 方法\*\*

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。

下面的例子返回带有类名 "url" 的所有 <p> 元素：

$(document).ready(function(){

$("p").filter(".url");

});

\*\*jQuery not() 方法\*\*

not() 方法返回不匹配标准的所有元素。

提示：not() 方法与 filter() 相反。

下面的例子返回不带有类名 "url" 的所有 <p> 元素：

$(document).ready(function(){

$("p").not(".url");

});

第七章 jQuery AJAX

7.1 jQuery AJAX简介

\*\*什么是 AJAX？\*\*

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

|  |
| --- |
| $.ajax({ |
|  | type:请求方式get/post, |
|  | url: 请求路径 (相对的、绝对的) |
|  | async:是否异步 true, |
|  | data: 向服务器发送数据 string/{} |
|  | dataType:接受数据的格式, |
|  | success: fn 成功后完成的操作, |
|  | error: fn 失败后完成的操作 |
|  | }); |

7.2 get和post方法

\*\*jQuery $.get() 方法\*\*

$.get() 方法通过 HTTP GET 请求从服务器上请求数据。

语法：

$.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：

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

$.get("demo\_test.php",function(data,status){

alert("数据: " + data + "\n状态: " + status);

});

});

\*\*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。

下面的例子使用 $.post() 连同请求一起发送数据：

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

$.post("/try/ajax/demo\_test\_post.jsp",

{

name:"百度",

url:"http://www.baidu.com"

},

function(data,status){

alert("数据: \n" + data + "\n状态: " + status);

});

});