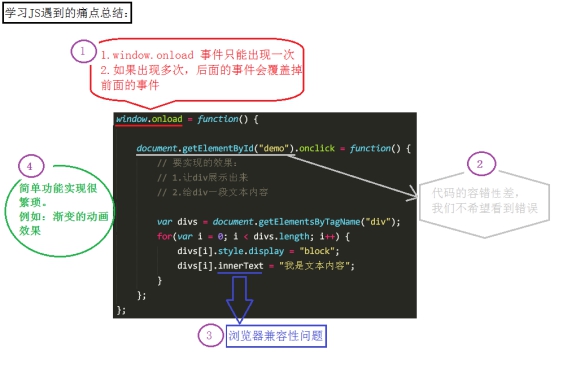
# jQuery第一天—选择器,事件和动画

**1.1 为什么要学jQuery？**

**1.1.1 学习JS的遇到的痛点**

痛点的总结：



**1.2 jQuery是什么？**

**1.2.1 jQuery描述（理解）**

jQuery是js的一个库，封装了我们开发过程中常用的一些功能，方便我们来调用，提高了我们的开发效率。

Js库是把我们常用的功能放到一个单独的文件中，我们用的时候，直接引用到页面里面来就可以了。

animate.js是我们自己封装的库，而jQuery是别人帮我们封装好的库。

**1.2.2 学习jQuery，主要是学什么呢？**

目前这个阶段，主要学习如何来使用jQuery操作DOM，其实就是学习jQuery封装好的那些功能方法，这些方法叫做API（Application Programming Interface应用程序编程接口）。

这些API的共同特点是：几乎全都是方法。所以，在使用jQuery的API时，都是方法调用，也就是说要加小括号()，小括号里面是相应的参数，参数不同，功能不同。

**1.3 jQuery详细解释**

**1.3.1 版本介绍（了解）**

浏览器支持情况：<http://jquery.com/browser-support/>

版本一：1.x版本

版本二：2.x版本

版本三：3.x版本

jquery 1.12.4版本api文档：(在其他资料的常用文件中有已经下载好的离线文档)

http://www.php100.com/manual/jquery/

同版本两个文件的区别：

min：压缩版，压缩过后，体积会更小

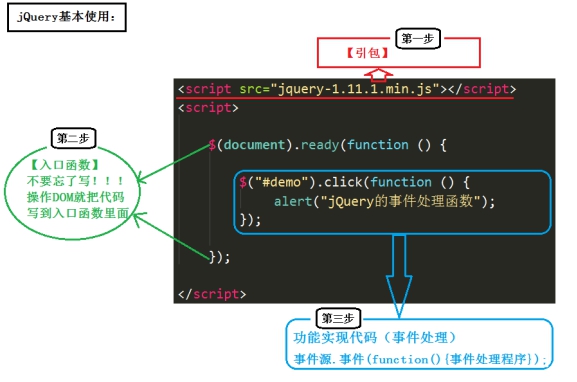
压缩指的是：把注释、空格、换行全部去掉，把变量名称尽可能的换成更加简短的字符。

压缩的主要目的：就是让文件变的更小。

平时开发过程中，这两个文件使用哪个都可以；但是，项目上线的时候，推荐使用压缩版。

**1.4 如何使用jQuery？（**重点**）**

使用步骤：



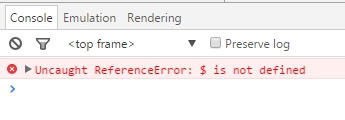
1．引包

2．入口函数

**1.4.1 引包注意点（理解）**

第一点：在使用jQuery之前，先把jQuery文件引到页面中来

如果在使用jQuery之前，没有引用jQuery文件，会报错：

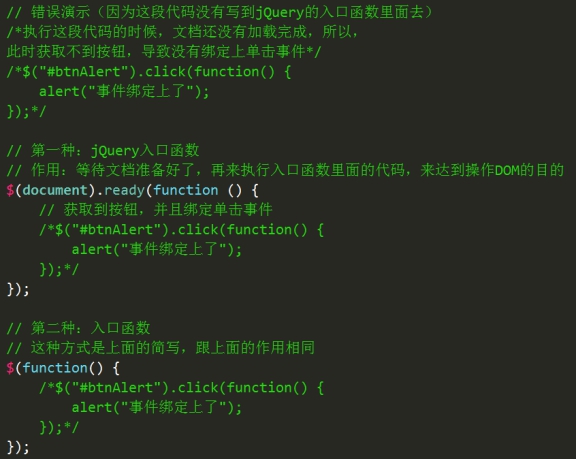


第二点：src路径一定要写正确

如果src路径写错，也会报错：

C:\Users\qf\Documents\My Knowledge\temp\82ef004f-71c6-4039-99c3-9d301f7ac6e2\128\index_files\wpsC005.tmp59e6540e-f088-41fe-9a62-07f2e8bef5c2.jpg

**1.4.2 jQuery的入口函数（重点）**



**1.4.3 jQuery入口函数与js入口函数的区别（理解）**

js入口函数指的是：window.onload = function() {};

区别一：书写个数不同

Js入口函数只能出现一次，出现多次会存在事件覆盖的问题。

jQuery的入口函数，可以出现任意多次，并不会存在事件覆盖问题。

区别二：执行时机不同

Js入口函数是在所有的文件资源加载完成后，才执行。这些文件资源包括：页面文档、外部的js文件、外部的css文件、图片等。

jQuery的入口函数，是在文档加载完成后，就执行。文档加载完成指的是：DOM树加载完成后，就可以操作DOM了，不用等到所有的外部资源都加载完成。

文档加载的顺序：从上往下，边解析边执行。

**1.4.4 jQuery的$符号（重点）**

js命名规范允许出现的字符有：数字、字母、下划线、$。

js命名规范允许作为变量命名开头的字符有：字母、下划线、$；但是，不允许以数字作为变量命名的开头。

var $ = “我是字符串”;

var $ = 123;

function $(){

alert(“我是函数$”);

}

jQuery使用$符号原因：书写简洁、相对于其他字符与众不同、容易被记住。

怎么理解jQuery里面的$符号：

$实际上表示的是一个函数。

$(); // 调用上面我们自定义的函数$

$(document）.ready(function(){}); // 调用入口函数

$(function(){}); // 调用入口函数

$(“#btnShow”) // 获取id属性为btnShow的元素

$(“div”) // 获取所有的div元素

jQuery里面的$函数，根据传入参数的不同，进行不同的调用，实现不同的功能。返回的是jQuery对象

jQuery这个js库，除了$之外，还提供了另外一个函数：jQuery

jQuery函数跟$函数的关系：jQuery ===$;

**1.4.5 jQuery对象和DOM对象的相互转换（难点）**

DOM对象此处指的是：使用js操作DOM返回的结果。

         var btn = document.getElementById(“btnShow”); // btn就是一个DOM对象

jQuery对象此处指的是：使用jQuery提供的操作DOM的方法返回的结果。

jQuery拿到DOM对象后又对其做了封装，让其具有了jQuery方法的jQuery对象，说白了，就是把DOM对象重新包装了一下。

         var $btn = $(“#btnShow”); // $btn就是一个jQuery对象

DOM对象转换成jQuery对象：

var $btn1 = $(btn); // 此时就把DOM对象btn转换成了jQuery对象$btn1

// $(document）.ready(function(){}); // 调用入口函数

// 此处是将document这个js的DOM对象，转换成了jQuery对象，然后才能调用jQuery提供的方法：ready

jQuery对象转换成DOM对象：

// 第一种方式

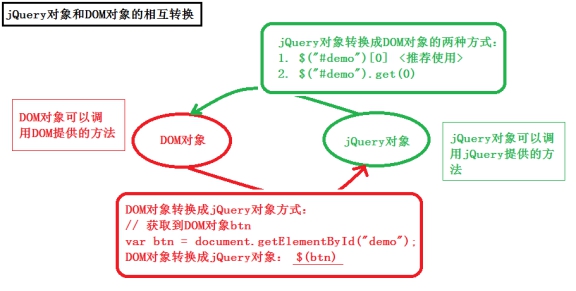
var btn1 = $btn[0]; // 此时就把jQuery对象$btn转换成了DOM对象btn1 （推荐使用此方式）

// 第二种方式

var btn2 = $btn.get(0);// 此时就把jQuery对象$btn转换成了DOM对象btn2

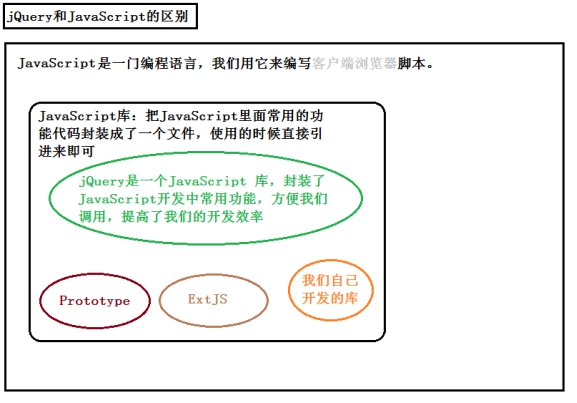
这两种方式都要记住，使用哪一种都可以。

图解：



所有这些，都体现了jQuery对js的封装！

**1.5 JS跟jQuery的区别是什么？（理解）**



**1.6 为什么要学jQuery选择器？**

**1.6.1 JS提供的选择DOM元素的方法有几个？**

考虑兼容性的话，js里面提供的选择DOM的方法只有两个：

|  |  |
| --- | --- |
| **JavaScript选择元素的方法：** | |
| **document.getElementById();** | 通过id属性获取指定元素  返回唯一的DOM对象 |
| **document.getElementsByTagName();** | 通过标签名获取指定元素  返回DOM对象数组（即使只有一个元素） |

正是因为js提供的选择DOM的方法太少了，满足不了我们平时开发的需要，所以，我们可以使用jQuery选择器来弥补这方面的不足。

**1.6.2 什么是jQuery选择器**

jQuery选择器是jQuery强大的体现，它提供了一组方法，让我们更加方便的获取到页面中的元素。（联想：CSS选择符）

**1.7 强大的jQuery选择器（重点）**

强大的原因：jQuery实现了从CSS1到CSS3所有的选择器以及其他常用的选择器。

各种选择器之间可以相互代替，所以，平时真正用到的只是最常用的选择器。

**1.7.1 基本选择器（重点）**

|  |  |  |
| --- | --- | --- |
| **符号(名称)** | **说明** | **用法** |
| **#** | Id选择器 | $(“#btnShow”).css(“color”, “red”);  选择id为btnShow的一个元素（返回值为jQuery对象，下同） |
| **.** | 类选择器 | $(“.liItem”).css(“color”, “red”);  选择含有类liItem的所有元素 |
| **element** | 标签选择器 | $(“li”).css(“color”, “red”);  选择标签名为li的所有元素 |

**1.7.2 层级选择器（重点）、基本过滤选择器**

|  |  |  |  |
| --- | --- | --- | --- |
| **符号(名称)** | **说明** | **用法** | |
| **层级选择器** | | | |
| **空格** | 后代选择器 | $(“#j\_wrap li”).css(“color”, “red”);  选择id为j\_wrap的元素的所有后代元素li | |
| **>** | 子代选择器 | $(“#j\_wrap > ul > li”).css(“color”, “red”);  选择id为j\_wrap的元素的所有子元素ul的所有子元素li | |
| **基本过滤选择器** | | | |
| **:eq(index)** | 选择匹配到的元素中索引号为index的一个元素，index从0开始 | | $(“li:eq(2)”).css(“color”,”red”);  选择li元素中索引号为2的一个元素 |
| **:odd** | 选择匹配到的元素中索引号为奇数的所有元素，index从0开始 | | $(“li:odd”).css(“color”, “red”);  选择li元素中索引号为奇数的所有元素 |
| **:even** | 选择匹配到的元素中索引号为偶数的所有元素，index从0开始 | | $(“li:odd”).css(“color”, “red”);  选择li元素中索引号为偶数的所有元素 |

**1.7.3 筛选选择器（方法）（重点）**

|  |  |  |
| --- | --- | --- |
| **符号(名称)** | **说明** | **用法** |
| **find(selector)** | 查找指定元素的所有后代元素（子子孙孙） | $(“#j\_wrap”).find(“li”).css(“color”, “red”);  选择id为j\_wrap的所有后代元素li |
| **children()** | 查找指定元素的直接子元素（亲儿子元素） | $(“#j\_wrap”).children(“ul”).css(“color”, “red”);  选择id为j\_wrap的所有子代元素ul |
| **siblings()** | 查找所有兄弟元素（不包括自己） | $(“#j\_liItem”).siblings().css(“color”, “red”);  选择id为j\_liItem的所有兄弟元素 |
| **parent()** | 查找父元素（亲的） | $(“#j\_liItem”).parent(“ul”).css(“color”, “red”);  选择id为j\_liItem的父元素 |
| **eq(index)** | 查找指定元素的第index个元素，index是索引号，从0开始 | $(“li”).eq(2).css(“color”, “red”);  选择所有li元素中的第二个 |

**1.8jQuery事件机制**

jQuery的事件机制，指的是：jQuery对JavaScript操作DOM事件的封装，包括了：事件绑定、事件解绑、事件触发。

**1.8.1jQuery事件的发展历程（了解）**

简单事件绑定 >> bind事件绑定 >> delegate事件绑定 >> on【重点】

**1.8.2jQuery事件绑定**

简单事件绑定：

click(handler)                       单击事件

blur(handler)                        失去焦点事件

mouseenter(handler)            鼠标进入事件

mouseleave(handler)             鼠标离开事件

dbclick(handler)                  双击事件

change(handler) 改变事件，如：文本框值改变，下来列表值改变等

focus(handler)                     获得焦点事件

keydown(handler)                键盘按下事件

bind方式（不推荐，1.7以后的jQuery版本被on取代）

作用：给匹配到的元素直接绑定事件

// 绑定单击事件处理程序

第一个参数：事件类型

第二个参数：事件处理程序

$("p").bind("click mouseenter", function(e){

    //事件响应方法

});

比简单事件绑定方式的优势：

优点:可以同时绑定多个事件，比如：bind(“mouseenter  mouseleave”, function(){})

缺点：要绑定事件的元素必须存在文档中。

* delegate方式（特点：性能高，支持动态创建的元素）

作用：给匹配到的元素绑定事件，对支持动态创建的元素有效

// 第一个参数：selector，要绑定事件的元素

// 第二个参数：事件类型

// 第三个参数：事件处理函数

$(".parentBox").delegate("p", "click", function(){

    //为 .parentBox下面的所有的p标签绑定事件

});

与前两种方式最大的优势：减少事件绑定次数提高效率，支持动态创建出来的元素绑定事件！

* **on方式（最现代的方式，强烈建议使用的方式）（重点）**

jQuery**1.7版本后**，jQuery用on统一了所有的事件处理的方法

作用：给匹配的元素绑定事件，包括了上面所有绑定事件方式的优点

语法：

// 第一个参数：events，绑定事件的名称可以是由空格分隔的多个事件（标准事件或者自定义事件）

// 第二个参数：selector, 执行事件的后代元素

// 第三个参数：data，传递给处理函数的数据，事件触发的时候通过event.data来使用

// 第四个参数：handler，事件处理函数

$(selector).on(events[,selector][,data],handler);

    // 表示给$(selector)绑定事件，当必须是它的内部元素span才能执行这个事件

$(selector).on( "click",“span”, function() {});

// 绑定多个事件

// 表示给$(selector)匹配的元素绑定单击和鼠标进入事件

$(selector).on(“click mouseenter”, function(){});

**1.8.3事件解绑**

unbind() 方式

作用：解绑 bind方式绑定的事件

$(selector).unbind(); //解绑所有的事件

$(selector).unbind(“click”); //解绑指定的事件

undelegate() 方式

作用：解绑delegate方式绑定的事件

$( selector ).undelegate(); //解绑所有的delegate事件

$( selector).undelegate( “click” ); //解绑所有的click事件

**off解绑on方式绑定的事件（重点）**

// 解绑匹配元素的所有事件

$(selector).off();

// 解绑匹配元素的所有click事件

$(selector).off(“click”);

// 解绑所有代理的click事件，元素本身的事件不会被解绑

$(selector).off( “click”, “\*\*” );

**1.8.4事件触发**

简单事件触发

$(selector).click(); //触发 click事件

trigger方法触发事件

$(selector).trigger(“click”);

triggerHandler触发 事件响应方法，**不触发浏览器行为**

**比如:文本框获得焦点的默认行为**

$(selector).triggerHandler(“focus”);

**1.8.5jQuery事件对象介绍**

event.data                             传递给事件处理程序的额外数据

event.currentTarget              等同于this，当前DOM对象

event.pageX                          鼠标相对于文档左部边缘的位置

**event.target**                         触发事件源，不一定===this

**event.stopPropagation()；**  阻止事件冒泡

**event.preventDefault();**       阻止默认行为

event.type                            事件类型：click，dbclick…

event.which                         鼠标的按键类型：左1 中2 右3

event.keyCode                      键盘按键代码

案例【按键变色】

**1.9 jQuery补充**

**1.9.1链式编程**

链式编程原理：return this;

通常情况下，只有设置操作才能把链式编程延续下去。因为获取操作的时候，会返回获取到的相应的值，无法返回 this。

end(); // 结束当前链最近的一次过滤操作，并且返回匹配元素之前的状态。

**1.9.2隐式迭代**

隐式迭代的意思是：在方法的内部会为匹配到的所有元素进行循环遍历，执行相应的方法；而不用我们再进行循环，简化我们的操作，方便我们调用。

如果获取的是多元素的值，大部分情况下返回的是第一个元素的值。

案例【五角星评分控件】

**1.5.3each方法**

有了隐式迭代，为什么还要使用each函数遍历？

大部分情况下是不需要使用each方法的，因为jQuery的隐式迭代特性。

如果要对每个元素做不同的处理，这时候就用到了each方法

作用：遍历jQuery对象集合，为每个匹配的元素执行一个函数

// 参数一表示当前元素在所有匹配元素中的索引号

// 参数二表示当前元素（DOM对象）

$(selector).each(function(index,element){});

**1.9 jQuery动画是什么？**

jQuery提供的一组网页中常见的动画效果，这些动画是标准的、有规律的效果；同时还提供给我们了自定义动画的功能。

**1.9.1 隐藏显示动画**

1 show方法

作用：让匹配的元素展示出来

// 用法一：

//

// 用法二：

// 参数为字符串类型，是jQuery预设的值，共有三个，分别是：slow、normal、fast

/\* 跟用法一的对应关系为： \*/

/\* slow：600ms、normal：400ms、fast：200ms \*/

$(selector).show(“slow”);

// 用法三：

// 参数一可以是数值类型或者字符串类型

// 参数二表示：动画执行完后立即执行的回调函数

$(selector).show(2000, function() {});

// 用法四：

// 不带参数，作用等同于 css(“display”, ”block”)

/\* 注意：此时没有动画效果 \*/

$(selector).show();

注意：

jQuery预设的三组动画效果的语法几乎一致：参数可以有两个，第一个是动画的执行时长，第二个是动画执行完成后的回调函数。

第一个参数是：可以是指定字符或者毫秒数

2 hide方法

作用：让匹配元素隐藏掉

用法参考show方法

$(selector).hide(1000); // 1000表示什么？

$(selector).hide(“slow”);

$(selector).hide(1000, function(){});

$(selector).hide();

**1.9.2 滑入滑出动画**

1滑入动画效果（联想：生活中的卷帘门）

作用：让元素以下拉动画效果展示出来

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

注意：省略参数或者传入不合法的字符串，那么则使用默认值：400毫秒（同样适用于fadeIn/slideDown/slideUp）

$(selector).slideDown();

2 滑出动画效果

作用：让元素以上拉动画效果隐藏起来

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

3滑入滑出切换动画效果

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

参数等同与1.9.1 隐藏和显示

**1.9.3 淡入淡出动画**

1 淡入动画效果

作用：让元素以淡淡的进入视线的方式展示出来

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

2 淡出动画效果

作用：让元素以渐渐消失的方式隐藏起来

$(selector).fadeOut(1000);

3淡入淡出切换动画效果

作用：通过改变不透明度，切换匹配元素的显示或隐藏状态

$(selector).fadeToggle('fast',function(){});

参数等同与1.9.1 隐藏和显示

4改变不透明度到某个值

与淡入淡出的区别：淡入淡出只能控制元素的不透明度从 完全不透明 到完全透明；而fadeTo可以指定元素不透明度的具体值。并且时间参数是必需的！

作用：调节匹配元素的不透明度

// 用法有别于其他动画效果

// 第一个参数表示：时长

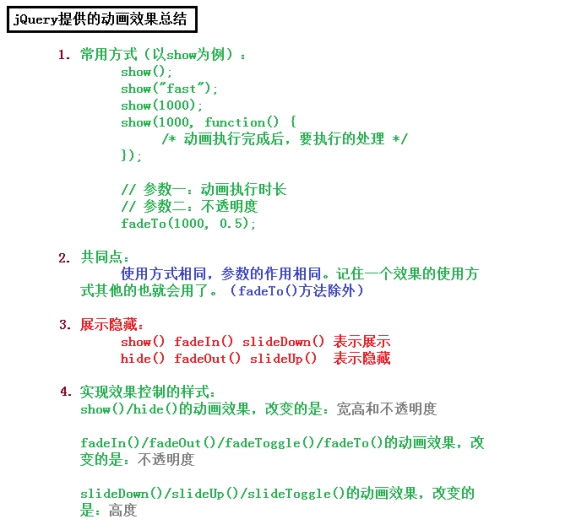
// 第二个参数表示：不透明度值，取值范围：0-1

$(selector).fadeTo(1000, .5); //  0全透，1全不透

// 第一个参数为0，此时作用相当于：.css(“opacity”, .5);

$(selector).fadeTo(0, .5);

jQuery提供的动画使用方法总结：



有规律的体现：

jQuery提供的这几个动画效果控制的CSS属性包括：高度、宽度、不透明度。{height:400px; width:300px; opacity:.4;}

这三个CSS属性的共性是：属性值只有一个，并且这个值是数值（去掉单位后）。

**1.9.4 自定义动画**

注意：所有能够执行动画的属性必须只有一个数字类型的值。

比如：要改变字体大小，要使用：fontSize（font-size），不要使用：font

动画支持的属性：

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

作用：执行一组CSS属性的自定义动画

// 第一个参数表示：要执行动画的CSS属性（必选）

// 第二个参数表示：执行动画时长（可选）

// 第三个参数表示：动画执行完后立即执行的回调函数（可选）

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

**1.9.5 停止动画**

stop()

作用：停止当前正在执行的动画

为什么要停止动画？

如果一个以上的动画方法在同一个元素上调用，那么对这个元素来说，后面的动画将被放到效果队列中。这样就形成了动画队列。（联想：排队进站）

动画队列里面的动画不会执行，直到第一个动画执行完成。

// 第一个参数表示是否清空所有的后续动画

// 第二个参数表示是否立即执行完当前正在执行的动画

$(selector).stop(clearQueue,jumpToEnd);

解释：

当调用stop()方法后，队列里面的下一个动画将会立即开始。但是，如果参数clearQueue被设置为true，那么队列面剩余的动画就被删除了，并且永远也不会执行。

如果参数jumpToEnd被设置为true，那么当前动画会停止，但是参与动画的每一个CSS属性将被立即设置为它们的目标值。比如：slideUp()方法，那么元素会立即隐藏掉。如果存在回调函数，那么回调函数也会立即执行。

注意：如果元素动画还没有执行完，此时调用sotp()方法，那么动画将会停止。并且动画没有执行完成，那么回调函数也不会被执行。

常用方式：

$(selector).stop();

 案例【手风琴】

# jQuery第二天—DOM操作和AJAX方法

**2.1 样式操作**

**2.1.1 样式属性操作  .css()**

作用：设置或获取元素的样式属性值

1 设置样式属性操作：

   a设置单个样式：

// 第一个参数表示：样式属性名称

// 第二个参数表示：样式属性值

$(selector).css(“color”, “red”);

b设置多个样式：（也可以设置单个）

// 参数为 {}（对象）

$(selector).css({“color”: “red”, “font-size”: “30px”});

2获取样式属性操作：

// 参数表示要获取的 样式属性名称

$(selector).css(“font-size”);

此时，会返回”font-size”样式属性对应的值。

**2.1.2 类操作**

添加类样式：

**addClass(className)**为指定元素添加类className

$(selector).addClass(“liItem”);

**注意：此处类名不带点，所有类操作的方法类名都不带点**

移除类样式：

**removeClass(className)** 为指定元素移除类 className

$(selector).removeClass(“liItem”);

$(selector).removeClass(); 不指定参数，表示移除被选中元素的所有类

判断有没有类样式：

hasClass(calssName) 判断指定元素是否包含类 className

$(selector).hasClass(“liItem”);

此时，会返回true或false

切换类样式：

toggleClass(className) 为指定元素切换类 className，该元素有类则移除，没有指定类则添加。

$(selector).toggleClass(“liItem”);

注意点：

操作类样式的时候，所有的类名，都不带点（.）

 经验：

1 操作的样式非常少，那么可以通过.css()这个 方法来操作

2 操作的样式很多，那么要通过使用类的方式来操作

3 如果考虑以后维护方便（把CSS从js中分离出来）的话，推荐使用类的方式来操作。类比CSS书写位置（把css从html中分离出来）

前面内容特色总结：简约、“粗暴”、干净利落、直截了当

**2.2 jQuery节点操作**

**2.2.1 动态创建元素**

// $()函数的另外一个作用：动态创建元素

var $spanNode = $(“<span>我是一个span元素</span>”);

**2.2.2 添加元素**

在元素的最后一个子元素后面追加元素：

**append()**（**重点**）

作用：在被选元素内部的最后一个子元素（或内容）后面插入内容（存在或者创建出来的元素都可以）

如果是页面中存在的元素，那么调用append()后，会把这个元素放到相应的目标元素里面去；但是，原来的这个元素，就不存在了。

 如果是给多个目标追加元素，那么方法的内部会复制多份这个元素，然后追加到多个目标里面去。

常用参数：htmlString 或者 jQuery对象

// 在$(selector)中追加$node

$(selector).append($node);

// 在$(selector)中追加div元素，参数为htmlString

$(selector).append('<div></div>');

不常用操作：（用法跟append()方法基本一致）

1 appendTo()

作用：同append()，区别是：把$(selector)追加到node中去

$(selector).appendTo(node);

2 prepend()

作用：在元素的第一个子元素前面追加内容或节点

$(selector).prepend(node);

3 after()

作用：在被选元素之后，作为兄弟元素插入内容或节点

$(selector).after(node);

4 before()

作用：在被选元素之前，作为兄弟元素插入内容或节点

$(selector).before(node);

**2.2.3 html创建元素（推荐使用，重点）**

作用：设置或返回所选元素的html内容（包括 HTML 标记）

设置内容的时候，如果是html标记，会动态创建元素，此时作用跟js里面的 innerHTML属性相同

// 动态创建元素

$(selector).html(‘<span>宅客学院</span>’);

// 获取html内容

$(selector).html();

**2.2.4 清空元素**

// 清空指定元素的所有子元素（光杆司令）

// 没有参数

$(selector).empty();

$(selector).html(“”);

// “自杀” 把自己（包括所有内部元素）从文档中删除掉

$(selector).remove();

**2.2.5 复制元素**

作用：复制匹配的元素

// 复制$(selector)所匹配到的元素

// 返回值为复制的新元素

$(selector).clone();

总结：

推荐使用html(“<span></span>”)方法来创建元素或者html(“”)清空元素

**2.3 操作form表单**

**2.3.1 属性操作**

设置属性：

// 第一个参数表示：要设置的属性名称

// 第二个参数表示：改属性名称对应的值

$(selector).attr(“title”, “宅客学院”);

获取属性：

// 参数为：要获取的属性的名称，改操作会返回指定属性对应的值

$(selector).attr(“title”);

此时，返回指定属性的值

移除属性：

// 参数为：要移除的属性的名称

$(selector).removeAttr(“title”);

注意：checked、selected、disabled要使用.prop()方法。

prop方法通常用来影响DOM元素的动态状态，而不是改变的HTML属性。例如：input和button的disabled特性，以及checkbox的checked特性。

细节参考：<http://api.jquery.com/prop/>

**2.3.2 值和内容**

val()方法：

作用：设置或返回表单元素的值，例如：input,select,textarea的值

// 获取匹配元素的值，只匹配第一个元素

$(selector).val();

// 设置所有匹配到的元素的值

$(selector).val(“具体值”);

text() 方法:

作用：设置或获取匹配元素的文本内容

//获取操作不带参数（注意：这时候会把所有匹配到的元素内容拼接为一个字符串，不同于其他获取操作！）

$(selector).text();

//设置操作带参数，参数表示要设置的文本内容

$(selector).text(“我是内容”);

**2.4 尺寸位置操作**

**2.4.1 高度和宽度操作**

高度操作height() ：

作用：设置或获取匹配元素的高度值

//带参数表示设置高度

$(selector).height(200);

//不带参数获取高度

$(selector).height();

宽度操作width() ：

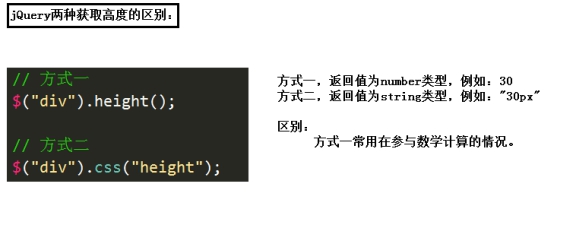
作用：设置或获取匹配元素的宽度值

//带参数表示设置宽度

//不带参数获取宽度

$(selector).width(200);

css()获取高度和height获取高度的区别？



**2.4.2 坐标值操作**

offset()

作用：获取或设置元素相对于文档的位置

// 无参数表示获取，返回值为：{left:num, top:num}，值是相对于document的位置

$(selector).offset();

// 有参数表示设置，参数推荐使用数值类型

$(selector).offset({left:100, top: 150});

注意点：设置offset后，如果元素没有定位(默认值：static)，则被修改为relative

position()

作用：获取相对于其最近的具有定位的父元素的位置。

// 获取，返回值为对象：{left:num, top:num}

$(selector).position();

注意：只能获取，不能设置。

scrollTop()

作用：获取或者设置元素垂直方向滚动的位置

// 无参数表示获取偏移

// 有参数表示设置偏移，参数为数值类型

$(selector).scrollTop(100);

scrollLeft()

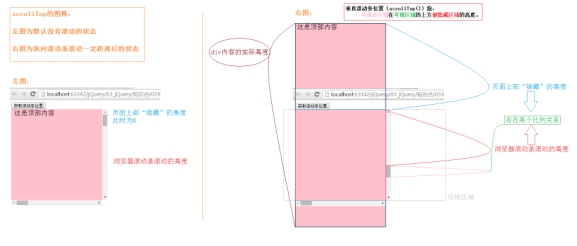
作用：获取或者设置元素水平方向滚动的位置

$(selector).scrollLeft(100);

对scrollTop的理解：

**垂直滚动条位置 是可滚动区域 在 可视区域上方的 被隐藏区域的高度。**

**如果滚动条在最上方没有滚动 或者 当前元素没有出现滚动条，那么这个距离为0**



**2.5 AJAX**

**jQuery为我们提供了强大的Ajax封装:**

$.ajax({}) 可配置方式发起Ajax请求

$.get(url, [data], [callback]) 以GET方式发起Ajax请求

$.post(url, [data], [callback]) 以POST方式发起Ajax请求

|  |  |
| --- | --- |
| 参数名称 | 说明 |
| url | 接口地址 |
| type | 请求方式 |
| timeout | 请求超时 |
| dataType | "json": 返回 JSON 数据;"jsonp": 返回函数调用格式。 |
| data | 发送到服务器的数据:name=John&location=Boston或者{ name: "John", time: "2pm" } |
| jsonpCallback | 可以指定相应的回调函数名，默认自动生成 |
| beforeSend: function | 请求发起前调用 |
| success: function | 成功响应后调用 |
| error: function | 错误响应时调用 |
| complete: function | 响应完成时调用（包括成功和失败） |

**2.6 插件机制**

jQuery这个js库，虽然功能强大，但也不是面面俱到包含所有的功能。

jQuery是通过插件的方式，来扩展它的功能：

当你需要某个插件的时候，你可以“安装”到jQuery上面，然后，使用。

当你不再需要这个插件，那你就可以从jQuery上“卸载”它。

（联想：手机软件，安装的app就好比插件，用的时候安装上，不用的时候卸载掉）

**2.6.1第三方插件**

弹出层插件: jquery.reveal.js

轮播图插件: jquery.bxslider.js jquery.bxslider.css

使用步骤:

1.引入jQuery文件

2.引入插件

3.使用插件

**2.6.2制作插件**

如何创建jQuery插件：

全局jQuery函数扩展方法

$.pluginName = function() {};

jQuery对象扩展方法

$.fn. pluginName = function() {};