**jQuery选择器**

**1.库和框架的区别？**

* 库： API函数库，提供API供你调用。库内部已经做了各种底层的封装，以及各种兼容问题的处理，写程序时调用库内的API接口就能实现需要的功能，不需要额外的代码来处理函数封装和兼容问题，让代码更简洁，效率更高。库就像是一个工具盒，需要什么工具就从库中调用。

使用库的过程就是根据所需的功能，查文档，再调用库内的对应的API接口。

* 框架：提供一套完整的解决方案，你按照方案来操作以实现需求。

使用框架的过程就是先查文档，再照着文档写代码，出错了再查文档。

**2.jquery 能做什么？**

jQuery将JavaScript的代码进行了封装，处理了兼容性问题，提供API进行调用, 让我们实现功能时不用再为兼容性而困扰，少写了许多代码。

实现一个目的需要很多步骤，使用原生js，就需要每个步骤都写出来，jQuery就把这些步骤打包封装进一个函数中，做成一个API，用户调用这个API，提供参数，就能一步实现目的，简洁高效。

**3. jquery 对象和 DOM 原生对象有什么区别？如何转化？**

* DOM对象 ：W3C标准用于操作文档的API。具体点来说就是指 HTML 文本中的所有被标签标记的东西 ，在js里可以用 getElementById 等传统的方法获得的对象，拥有原生对象的属性和方法。
* jQuery对象：将DOM原生对象进行封装后得到的类数组对象，它是jQuery独有的。如果一个对象是jQuery对象，那么就可以使用jQuery里的方法。

var domObj = document.getElementById('id'); //DOM对象

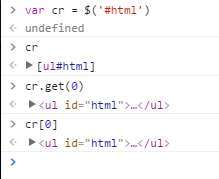
var obj=('#id'); //jQuery对象;

注意：在jQuery对象中无法使用DOM对象的任何方法。同样，DOM对象也不能使用jQuery方法。

jquery提供了两种方法将一个jquery对象转换成一个dom对象，[index]和get(index)。

var cr=('#html'); //jquery对象

var cr = cr[0];//dom对象,也可写成var cr=cr.get(0);



Paste\_Image.png

对于一个dom对象，只需要用()把dom对象包装起来，就可以获得一个jquery对象了，方法为$(dom对象);

var cr=document.getElementById('html'); //dom对象

var cr= $(cr); //转换成jquery对象



Paste\_Image.png

**5.jquery事件绑定**

1.普通事件绑定：

//事件一

$("#btn").on("click",function(){

console.log("我被点击了")；

})

//事件二，给事件添加命名空间，下面的.myClick 就是添加的命名空间，用来标识本次事件，删除事件的时候方便指定

$("#btn").on("click.myClick",function(){

console.log("糟糕，我怎么又被点击了")；

})

$("#btn").off("click.myClick") // 仅仅解绑了上面添加命名的事件二，不会解绑事件一

2.事件代理

// 事件委托或者事件代理，想让div下面所有的span绑定事件，可以把事件直接绑定到div上

$('div').on('click', 'span', function(e){

console.log(this);

console.log(e);

});

3.可以在绑定的时候给事件处理程序传递一些参数

$('div').on('click', {name: 'Byron', age: 24}, function(e){

console.log(e.data);

});

**6.bind、unbind、delegate、live、on、off都有什么作用?**

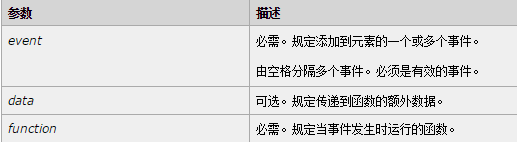
* bind() 方法为被选元素添加一个或多个事件处理程序，并规定事件发生时运行的函数。

$(selector).bind(event,data,function)

$("#btn").bind("click.myClick", function() {

console.log("我被点击了")

})



Paste\_Image.png

* unbind() 方法移除被选元素的事件处理程序。该方法能够移除所有的或被选的事件处理程序，或者当事件发生时终止指定函数的运行。ubind() 适用于任何通过 jQuery 附加的事件处理程序。

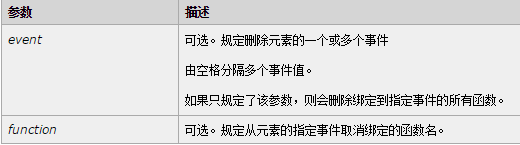
$(selector).unbind(event,function)

//移除所有 p 元素的事件处理函数：

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

$("p").unbind();

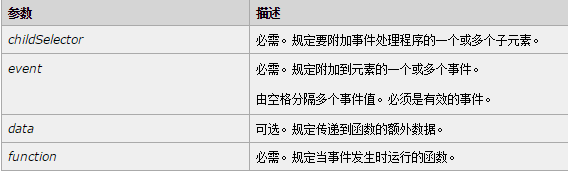
});



Paste\_Image.png

* delegate：用于事件代理，为指定的元素（属于被选元素的子元素）添加一个或多个事件处理程序，并规定当这些事件发生时运行的函数；适用于当前或未来的元素（比如由脚本创建的新元素）。

$(selector).delegate(childSelector,event,data,function)



Paste\_Image.png

$("ul").delegate("li", "click", function() {

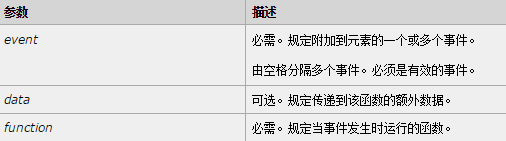
var text = $(this).text()

console.log(text)

})

* live()方法将监听器绑定在了document上，不把监听器绑定在匹配的元素上。利用事件委托机制，把节点处理委托给document。

$(selector).live(event,data,function)



Paste\_Image.png

//当点击按钮时，隐藏或显示 p 元素：

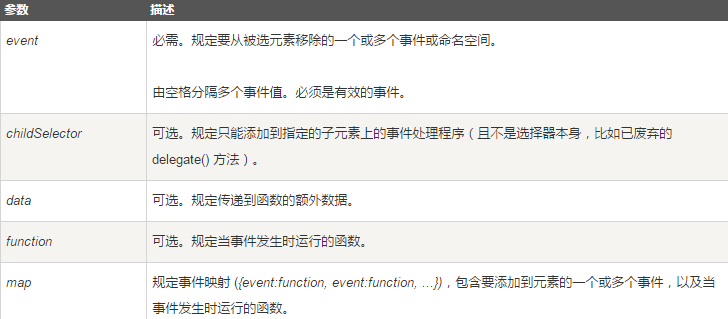
$("button").live("click",function(){

$("p").slideToggle();

});

* on() 方法在选定的元素上绑定一个或多个事件处理函数，提供绑定事件处理的所有功能。

$(selector).on(event,childSelector,data,function,map)



Paste\_Image.png

//普通事件绑定

$("#btn").on("click",function(){

console.log("我被点击了")

})

//事件代理绑定

$("ul").on("click", "li", function() {

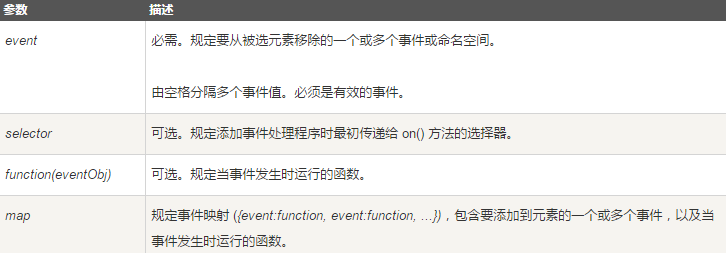
var text = $(this).text()

console.log(text)

})

* off() 方法通常用于移除通过 on()方法添加的事件处理程序。

$(selector).off(event,selector,function(eventObj),map)



Paste\_Image.png

$("#btn").off("click.myClick")

**推荐使用：**on方法，因为既能普通绑定事件，也能事件代理绑定。自 jQuery 版本 1.7 起，on() 方法是 bind()、live() 和 delegate() 方法的新的替代品。该方法给 API 带来很多便利，简化了 jQuery 代码库。 **注意**：on的事件代理和delegate的事件代理，写法不一样，子元素和事件监听函数的位置不同

//on的事件代理

$("ul").on("click", "li", function() {

var text = $(this).text()

console.log(text)

})

//delegate的事件代理

$("ul").delegate("li", "click", function() {

var text = $(this).text()

console.log(text)

})

//li和click的位置不一样

**5.jquery 如何展示/隐藏元素？**

.show([speed], [callback])：显示元素 .hide([speed], [callback])：隐藏元素 .toggle([speed], [callback]): 如果元素是最初显示，它会被隐藏；如果隐藏的，它会显示出来 .fadeIn([speed], [callback])：通过淡入的方式显示匹配元素。 .fadeOut([speed], [callback])：通过淡出的方式隐藏匹配元素。 .fadeToggle([speed], [callback])：淡入淡出的方式显示隐藏元素，隐藏显示元素 .fadeTo(speed, opacity，[callback])：通过匹配元素的不透明度做动画效果。 .slideUp([speed], [callback])：以滑动的方式隐藏匹配的元素 .slideDown([speed], [callback])：以滑动的方式显示匹配元素

speed：规定速度，取值：'slow', 'normal', 'fast'或毫秒 callback：显示或隐藏后执行的函数 opacity：透明度（0~1）

$('#btn-box1').on('click',function(){

$('.box').show(3000);

//$('.box').fadeIn();

//$('.box').slideDown('slow');

})

$('#btn-box2').on('click',function(){

$('.box').hide(3000);

//$('.box').fadeOut('slow');

//$('.box').slideUp('slow');

})

**6. jquery 动画如何使用？**

animate() 方法执行 CSS 属性集的自定义动画。该方法通过CSS样式将元素从一个状态改变为另一个状态。CSS属性值是逐渐改变的，这样就可以创建动画效果。只有数字值可创建动画（比如 "margin:30px"）。字符串值无法创建动画（比如 "background-color:red"）。

$(selector).animate(styles, speed, easing, callback)

参数说明：

* style:必须，规定产生动画效果的CSS
* speed:可选，规定动画的速度，默认是normal，可能的值：数字(毫秒，比如5000)、'slow'、'normal'，'fast'
* easing: 可选，规定在不同的动画点中设置动画速度的 easing 函数。内置的 easing 函数： swing, linear
* callback: 可选，animate 函数执行完之后，要执行的函数。

var css = {

width: "100px",

opacity: 0.8,

marginRight: "800px",

fontSize: "200px"

}

$(".box").animate(css, "slow", "linear", function(){

console.log("动画播放成功")

})

**7.如何设置和获取元素内部 HTML 内容？如何设置和获取元素内部文本？**

.html(string) 用于获取/修改匹配元素的innerHTML 无参数时，返回元素的innerHTML 有参数时，修改元素的innerHTML为参数值 .text(string) 用来得到匹配元素集合中每个元素的文本内容结合，包括他们的后代，或设置匹配元素集合中每个元素的文本内容为指定的文本内容。 无参数时，获取元素内部文本 有参数时，设置元素内部文本为参数值

获取内容：

$(".box").html() //获取元素内部的html内容，类似于innerHTML

$(".box").text() //获取元素内部的text文本，类似于innerText

设置内容：

$(".box").html("<p>设置了一个段落</p>")//设置了元素内部的html内容,标签生效

$(".box").text("设置了一个文本")//设置了元素内部的text文本，标签不生效

**注意**：如果结果是多个时进行赋值操作的时候会给每个结果都赋值；如果结果是多个，获取值的时候，返回结果集中的第一个对象的相应值

**9.如何设置和获取表单用户输入或者选择的内容？如何设置和获取元素属性？**

* val() 方法返回或设置被选元素的 value 属性，通常与 HTML 表单元素一起使用。

当用于返回值时：返回第一个匹配元素的 value 属性的值。 当用于设置值时：设置所有匹配元素的 value 属性的值。

//返回 value 属性：

$(selector).val()

//设置 value 属性：

$(selector).val(value)

* attr()方法用来设置或返回被选元素的属性和值。

当该方法用于返回属性值时：返回第一个匹配元素的值。 当该方法用于设置属性值时：为匹配元素设置一个或多个属性/值对。

//返回属性的值：

$(selector).attr(attribute)

//设置属性和值：

$(selector).attr(attribute,value)