[Jquery学习笔记 2](#_Toc17032)

[选择器 2](#_Toc11192)

[层级选择器 3](#_Toc20476)

[过滤选择器 3](#_Toc13734)

[基础过滤器 3](#_Toc30586)

[内容过滤选择器 4](#_Toc23289)

[可见过度选择器 4](#_Toc28985)

[属性过滤选择器 5](#_Toc31024)

[子元素过滤选择器 6](#_Toc6749)

[表单对象属性过滤选择器 7](#_Toc14)

[表单选择器 8](#_Toc22016)

[Jquery 对象常用方法 9](#_Toc22283)

[JQuery 事件 10](#_Toc17273)

[Jquery Ajax 实现 13](#_Toc1777)

# Jquery学习笔记

**Jquery 选择器 和 css选择器基本一样**

$(“td tr:nth(1)”) 选中 每个td下的第一行 tr

用 **each** 遍历 jquery 包装集 参数n 是 包装集中元素的下标 从0 开始

each 是 jquery 包装集的方法 js 数组 js dom对象不能使用 each 方法

$(“td tr:nth(1)”).each(function(n){

Alert($(“td tr:nth(1)”)[n].innerHtml);

})

map

属性操作 方法 attr 此方法是 包装集的方法

$(“td”).attr(“id”); 返回 id属性的值

设js 对象 为 obj $(obj) 是其包装集

$(“td”).attr(“id”,”hello”); 将 hello 赋值给id属性

**:even** 偶数行 如 $(“tr : even”) 选择 偶数行

使用 **$() 将一个js节点 封装成 jquery 节点 $(js 对象)**

**$(“”)[index] 将jquery节点 转换为js节点 或者通过$(“”).get(index)方法**

**jquery节点的html 方法**

**$(“”).html() 获取 节点的内容**

**$(“”).html(“<div>jskf<div>”) 向 节点 里写内容 (标签会被浏览器翻译)**

$(**function** () {  
 // $("tr td:even").css("color","blue")  
 **var** html=""  
 html="<tr><td>hello</td><td>hello</td></tr><tr><td>world</td><td>world</td></tr>"  
 $("table").html(html) //向 table里加内容 输出 hello hello

World world   
});

### 选择器

**基本选择器**：#id .class tag(标签名称) \*(匹配所有)

$(“#id, .class,tag...”) 将多个选择器匹配的标签，合并在一起

也可以通过 $(“”).add(“”)....的方式 选择多个包装集

#### 层级选择器

$(“table tr”) 在祖先元素下 匹配所有后代元素

$(“table>tr”) 匹配 父元素下的一级子元素，再往下的不匹配

$(“prev+next”) 匹配紧接着prev后的next元素

#### 过滤选择器

找到一堆元素后 需要过滤我们想要的元素，此时，需要过滤选择器 符号 **：**过滤规则

基础过滤器

1、:first  
用法: $("tr:first") ; 返回值 单个元素的组成的集合

说明: 匹配找到的第一个元素

2、:last

用法: $("tr:last") 返回值 集合元素

说明: 匹配找到的最后一个元素.与 :first 相对应.

3、:not(selector) selector ：是选择器 即：$(“”) 里面的内容

用法: $("input:not(:checked)") 返回值 集合元素

说明: 去除所有与给定选择器匹配的元素.有点类似于”非”,意思是没有被选中的input(当input的type=”checkbox”).

4、:even

用法: $("tr:even") 返回值 集合元素

说明: 匹配所有索引值为偶数的元素，从0开始计数.js的数组都是从0开始计数的.例如要选择table中的行,因为是从0开始计数,所以table中的第一个tr就为偶数0.

5、:odd

用法: $("tr:odd") 返回值 集合元素

说明: 匹配所有索引值为奇数的元素,和:even对应,从 0 开始计数

6、:eq(index)

用法: $("tr:eq(0)") 返回值 集合元素

说明: 匹配一个给定索引值的元素.eq(0)就是获取第一个tr元素.括号里面的是索引值,不是元素排列数

7、:gt(index)  
用法: $("tr:gt(0)") 返回值 集合元素

说明: 匹配所有大于给定索引值的元素

8、:lt(index)

用法: $("tr:lt(2)") 返回值 集合元素

说明: 匹配所有小于给定索引值的元素

9、:header(固定写法)

用法: $(":header").css("background", "#EEE") 返回值 集合元素

说明: 匹配如 h1, h2, h3之类的标题元素.这个是专门用来获取h1,h2这样的标题元素

10、:animated(固定写法) 返回值 集合元素

说明: 匹配所有正在执行动画效果的元素

##### 内容过滤选择器

1、:contains(text)  
用法: $("div:contains('John')") 返回值 集合元素

说明: 匹配包含给定文本的元素.这个选择器比较有用，当我们要选择的不是dom标签元素时,它就派上了用场了,它的作用是查找被标签”围”起来的文本内容是否符合指定的内容的。

2、:empty

用法: $("td:empty") 返回值 集合元素

说明: 匹配所有不包含子元素或者文本的空元素。

3、:has(selector)

用法: $("div:has('.mini')")返回值 集合元素

说明: 匹配含有选择器所匹配的元素的 元素。(返回的是**:**前面的元素)

4、:parent  
 用法: $("td:parent") 返回值 集合元素

说明: 匹配含有子元素或者文本的元素.注意:这里是":parent”,可不是".parent”!感觉与上面讲的”:empty”形成反义词。

##### 可见过度选择器

1、:hidden

用法: $(”tr:hidden”) 返回值 集合元素

说明: 匹配所有的不可见元素，input 元素的 type 属性为 “hidden” 的话也会被匹配到。

意思是css中display:none和input type="hidden"的都会被匹配到.同样,要在脑海中彻底分清楚冒号":", 点号"."和逗号","的区别。

2、:visible

用法: $("tr:visible") 返回值 集合元素

说明: 匹配所有的可见元素

##### 属性过滤选择器

1、[attribute]

用法: $("div[id]") ; 返回值 集合元素

说明: 匹配包含给定属性的元素。例子中是选取了所有带”id”属性的div标签。

2、[attribute=value]

用法: $("input[name='newsletter']").attr("checked", true); 返回值 集合元素

说明: 匹配给定的属性是某个特定值的元素.例子中选取了所有 name 属性是 newsletter 的 input 元素。

3、[attribute!=value]$("div[title!='test']").css("background","yellow");

用法: $(”input[name!='newsletter']“).attr("checked", true); 返回值 集合元素

说明: 匹配所有不含有指定的属性，或者属性不等于特定值的元素。

此选择器等价于:not此选择器等价于:not([attr=value]),要匹配含有特定属性但不等于特定值的元素,请使用[attr]:not([attr=value])。之前看到的:not 派上了用场。

4、[attribute^=value]

用法: $(”input[name^=‘news’]“) 返回值 集合元素

说明: 匹配给定的属性是以某些值开始的元素.,我们又见到了这几个类似于正则匹配的符号。

5、[attribute$=value]

用法: $("input[name$='letter']") 返回值 集合元素

说明: 匹配给定的属性是以某些值结尾的元素。

6、[attribute\*=value]

用法: $("input[name\*='man']") 返回值 集合元素

说明: 匹配给定的属性是以包含某些值的元素。

7、[attributeFilter1][attributeFilter2][attributeFilterN]

用法: $("input[id][name$='man']") 返回值 集合元素

说明: 复合属性选择器,需要同时满足多个条件时使用.又是一个组合,这种情况我们实际使用的时候很常

用.这个例子中选择的是所有含有 id 属性,并且它的 name 属性是以 man 结尾的元素。

##### 子元素过滤选择器

1、:nth-child(index/even/odd/equation)$("div[class=one] :nth-child(2)").css("background","yellow");

用法: $("ul li:nth-child(2)") 返回值 集合元素

说明: 匹配其父元素下的第N个子或奇偶元素.这个选择器和之前说的基础过滤(Basic Filters)中的

eq() 有些类似,不同的地方就是前者是从0开始,后者是从1开始。

2、:first-child$("div[class=one] :first-child")

用法: $("ul li:first-child") 返回值 集合元素

说明: 匹配第一个子元素。':first' 只匹配一个元素,而此选择符将为每个父元素匹配一个子元素。这里需要特别点的记忆下区别。

3、:last-child

用法: $("ul li:last-child") 返回值 集合元素

说明: 匹配最后一个子元素.':last'只匹配一个元素,而此选择符将为每个父元素匹配一个子元素。

4、: only-child

用法: $("ul li:only-child") 返回值 集合元素

说明: 如果某个元素是父元素中唯一的子元素,那将会被匹配。如果父元素中含有其他元素,那将不会被匹配。意思就是:只有一个子元素的才会被匹配!

##### 表单对象属性过滤选择器

1、:enabled

用法: $("input:enabled") 返回值 集合元素

说明: 匹配所有可用元素。意思是查找所有input中不带有disabled="disabled"的input。不为

disabled,当然就为enabled。

2、:disabled

用法: $("input:disabled") 返回值 集合元素

说明: 匹配所有不可用元素。与上面的enable是相对应的。

3、:checked

用法: $("input:checked") 返回值 集合元素

说明: 匹配所有选中的被选中元素(复选框、单选框等，不包括select中的option)。这话说起来有些绕口。

**4、:selected**

**用法: $(”select option:selected”) 返回值 集合元素**

**说明: 匹配所有选中的option元素.$("select>option:selected")**

##### 表单选择器

1、:input

用法: $(":input") ; 返回值 集合元素

说明:匹配所有 input, textarea, select 和 button 元素。

2、:text  
用法: $(":text") ; 返回值 集合元素

说明: 匹配所有的单行文本框。

3、:password  
用法: $(":password") ; 返回值 集合元素

说明: 匹配所有密码框。

4、:radio

用法: $(":radio") ; 返回值 集合元素

说明: 匹配所有单选按钮。

5、:checkbox

用法: $(":checkbox") ; 返回值 集合元素

说明: 匹配所有复选框。

6、:submit  
用法: $(":submit") ; 返回值 集合元素

说明: 匹配所有提交按钮.

7、:image

用法: $(":image") 返回值 集合元素

说明: 匹配所有图像域。

8、:reset

用法: $(":reset") ; 返回值 集合元素

说明: 匹配所有重置按钮。

9、:button

用法: $(":button") ; 返回值 集合元素

说明: 匹配所有按钮.这个包括直接写的元素button。

10、:file

用法: $(":file") ; 返回值 集合元素

说明: 匹配所有文件域。

11、:hidden

用法: $("input:hidden") ; 返回值 集合元素

说明: 匹配所有不可见元素，或者type为hidden的元素.这个选择器就不仅限于表单了,除了匹配input中的hidden外,那些style为hidden的也会被匹

。

### Jquery 对象常用方法

**each map has**

//使用 map 返回一个数组 将包装集中的元素转换为数组  
**var** people=$("tr td:nth-child(1)").map(**function** () {  
 //这里可以返回任何内容 this 是 当前 标签的js对象  
 **return** $(**this**).html()  
}).get()  
alert(people)

$(“tr”).has(“td”) 返回的是一个新包装集，父类元素 含有 子类元素，返回的是

父类元素的包装集

**var** tds=$("tr td")  
//用 each 遍历包装集 n 是 数组元素的下标  
tds.each(**function** (n) {  
 alert(tds[n].innerText)  
})

//this 代表当前遍历对象的 js 对象

$("div").each(**function** () {  
 alert(**this**.innerText)  
})

$(“”).not(“”) 将后者过滤掉

$(“”).filter(“”) 过滤后只剩下后者 filter 和 find还不一样

endSelf() 所有包装集合并在一起

end() 返回上一个包装集

$(”p”).addClass(css中定义的样式类型); 给某个元素添加样式

$(”img”).attr({src:”test.jpg”,alt:”test Image”}); 给某个元素添加属性/值，参数是map

$(”img”).attr(”src”,”test.jpg”); 给某个元素添加属性/值

$(”img”).attr(”title”, function() { return this.src }); 给某个元素添加属性/值

$(”元素名称”).html(); 获得该元素内的内容（元素，文本等）

$(”元素名称”).html(”<b>new stuff</b>”); 给某元素设置内容

$(”元素名称”).removeAttr(”属性名称”) 给某元素删除指定的属性以及该属性的值

$(”元素名称”).removeClass(”class”) 给某元素删除指定的样式

$(”元素名称”).text(); 获得该元素的文本

$(”元素名称”).text(value); 设置该元素的文本值为

$(”元素名称”).toggleClass(class) 当元素存在参数中的样式的时候取消,如果不存在就设置此样式

**$(”input元素名称”).val(); 获取input元素的值**

**$(”input元素名称”).val(value); 设置input元素的值为value**

Manipulation：

$(”元素名称”).after(content); 在匹配元素后面添加内容

$(”元素名称”).append(content); 将content作为元素的内容插入到该元素的后面

$(“selected1”).append($(“selected2”)) 将selected2 加入到 selected1里面

$(”元素名称”).appendTo(content); 在content后接元素

$(”元素名称”).before(content); 与after方法相反

$(”元素名称”).clone(布尔表达式) 当布尔表达式为真时，克隆元素（无参时，当作true处理）

$(”元素名称”).empty() 将该元素的内容设置为空

$(”元素名称”).insertAfter(content); 将该元素插入到content之后

$(”元素名称”).insertBefore(content); 将该元素插入到content之前

$(”元素”).prepend(content); 将content作为该元素的一部分，放到该元素的最前面

$(”元素”).prependTo(content); 将该元素作为content的一部分，放content的最前面

$(”元素”).remove(); 删除所有的指定元素

$(”元素”).remove(”exp”); 删除所有含有exp的元素

$(”元素”).wrap(”html”); 用html来包围该元素

$(”元素”).wrap(element); 用element来包围该元素

has 方法：

### JQuery 事件

blur( ) 元素失去焦点 a, input, textarea, button, select, label, map, area

鼠标从 该元素上离开后 再点击 鼠标 触发该事件

**change**( ) 用户改变域的内容 input, textarea, select

**click**( ) 鼠标点击某个对象 几乎所有元素

dblclick( ) 鼠标双击某个对象 几乎所有元素

error( ) 当加载文档或图像时发生某个错误 window, img

**focus**( ) 元素获得焦点 a, input, textarea, button, select, label, map, area

keydown( ) 某个键盘的键被按下 几乎所有元素

keypress( ) 某个键盘的键被按下或按住 几乎所有元素

keyup( ) 某个键盘的键被松开 几乎所有元素

load( fn ) 某个页面或图像被完成加载 window, img

mousedown( fn ) 某个鼠标按键被按下 几乎所有元素

mousemove( fn ) 鼠标被移动 几乎所有元素

mouseout( fn ) 鼠标从某元素移开 几乎所有元素

mouseover( fn ) 鼠标被移到某元素之上 几乎所有元素

mouseup( fn ) 某个鼠标按键被松开 几乎所有元素

resize( fn ) 窗口或框架被调整尺寸 window, iframe, frame

scroll( fn ) 滚动文档的可视部分时 window

select( ) 文本被选定 document, input, textarea

submit( ) 提交按钮被点击 form

unload( fn ) 用户退出页面 window

=====================================================================

ready(fn); $(document).ready()注意在body中没有onload事件，否则该函数不能执行。在每个页面中可以

有很多个函数被加载执行，按照fn的顺序来执行。

bind( type, [data], fn ) 为每一个匹配元素的特定事件（像click）绑定一个或多个事件处理器函数。可能的事件属性有：blur, focus, load, resize, scroll, unload, click, dblclick, mousedown, mouseup, mousemove,

mouseover, mouseout, mouseenter, mouseleave, change, select, submit, keydown, keypress,

keyup, error

$("span,input,div").bind({  
 click:**function** () {  
 alert("click")  
 },  
 mousemove:**function** () {  
 alert("mousemove")  
 }  
})

bind 存在的问题，当增加新元素的时候，没有办法为新元素绑定事件。

需要重新 手动的用 bind 函数绑定

one( type, [data], fn ) 为每一个匹配元素的特定事件（像click）绑定一个或多个事件处理器函数。在每个对 象上，这个事件处理函数只会被执行一次。其他规则与bind()函数相同。

trigger( type, [data] ) 在每一个匹配的元素上触发某类事件。

triggerHandler( type, [data] ) 这一特定方法会触发一个元素上特定的事件(指定一个事件类型)，同时取消浏览器对此事件的默认行动

unbind( [type], [data] ) 反绑定，从每一个匹配的元素中删除绑定的事件。

$(”p”).unbind() 移除所有段落上的所有绑定的事件

$(”p”).unbind( “click” ) 移除所有段落上的click事件

hover( over, out ) over,out都是方法, 当鼠标移动到一个匹配的元素上面时，会触发指定的第一个函数。当鼠标移出这个元素时，会触发指定的第二个函数。

$(”p”).hover(function(){

$(this).addClass(”over”);

},

function(){

$(this).addClass(”out”);

}

);

toggle( fn, fn ) 如果点击了一个匹配的元素，则触发指定的第一个函数，当再次点击同一元素时，则触发指定的第二个函数。

$(”p”).toggle(function(){

$(this).addClass(”selected”);

},

function(){

$(this).removeClass(”selected”);

}

);

live事件和事件委派

### Jquery Ajax 实现

第一种：**load(url,data,function(response,status,xhr))**

Url：将请求发送给 哪个 url

Data：可选，发送请求所传递的参数（发送到服务器的数据）

function(response,status,xhr)：可选：当请求完成时，运行此函数

$(*selector*).load(*URL*,*data*,*callback*);

第二种：$.get();

$.get(*URL*,*callback*);

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

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

alert("Data: " + data + "\nStatus: " + status);

});

});

Data:存有服务器端响应的内容

Status：请求状态

$(**function** () {  
 $("input").click(**function** () {

//获取json文件内容  
 $.get("pepole.json",**function** (data, status) {  
 alert(data.length)  
 **for**(**var** i=0;i<data.length;i++){  
 alert(data[i].name+":"+data[i].age+":"+data[i].gender)  
 }  
 },"json")  
 })  
});

第三种：

$.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。

$.post("demo\_test\_post.asp",

{

name:"Donald Duck",

city:"Duckburg"

},

function(data,status){

alert("Data: " + data + "\nStatus: " + status);

});

Data：服务器响应的数据

Status：请求的状态

第四种：$.ajax();

**$.ajax({**

**type:"get",**

**url:"",**

**async:true ,**

**cache：false,**

**dataType:’json’**

**...**

**success:function(data){**

**},**

**error:function(data){**

**},**

**});**

**Async:是否异步请求**

**Success：请求成功的回调函数**

**Error：请求失败的回调函数**

**dataType:服务器端响应数据类型**

事件委派:

事件委派的定义就是，把原来加给子元素身上的事件绑定在父元素身上，就是把事件委派给父元素

Jquery 1.8 之后 ，引入 on（） off() 两个函数同一处理事件绑定

在此之前，通过 bind() ,live(),delegate() 处理事件绑定

$(".b").bind("click",**function** () {  
 alert(**this**.value)  
})  
  
$("body").append("<input type=\"button\" class=\"b\" value=\"b2\">")

使用bind（）在此之后 添加的 元素 bind() 绑定的 选择器 对其不起作用

$(document).bind(.......) 这样虽然起作用，但是 很影响性能

使用 live() 方法解决（通过事件委托，通过事件冒泡机制实现），单是live方法危害很多