jQuery总结

[# jQuery对象转换为DOM对象 2](#_Toc23848)

[# DOM对象转换为jQuery对象 2](#_Toc5297)

[# 内容选择器 2](#_Toc27113)

[# 可见性选择器 2](#_Toc27449)

[# 表单选择器 2](#_Toc18190)

[# html() 和 text() 3](#_Toc11313)

[# jQuery创建节点——$(“html结构”) 3](#_Toc2837)

[# jQuery插入节点——append()和appendTo() 3](#_Toc6892)

[# jQuery插入节点——prepend()和preendTo() 4](#_Toc8080)

[# jQuery外部插入节点——after(“元素”)和before(“元素”) 4](#_Toc2075)

[# 清空节点内的内容——empty() 4](#_Toc29250)

[# 删除节点：包括自身和绑定的事件——remove() 4](#_Toc29415)

[# 删除节点：暂存缓存，添加相同元素依然存在绑定事件—detach() 5](#_Toc29789)

[# 克隆节点—clone() 5](#_Toc25529)

[# 替换节点—replaceWith()和replaceAll() 5](#_Toc161)

[# 包裹节点—wrap() 5](#_Toc22114)

[# 删除包裹节点—unwrap() 5](#_Toc18274)

[# 多元素包裹节点—wrapAll() 5](#_Toc30348)

[# 子元素包裹节点—wrapAll() 5](#_Toc6076)

[# 查找匹配元素的子元素节点—children() 6](#_Toc18579)

[# 查找匹配元素的后代元素节点—find() 6](#_Toc22406)

# jQuery对象转换为DOM对象

···

Var $div = $(“div”)

Var div1 = $div.eq(0)

Var div2 = $div[0]

Var div3 = $div.get(0)

//jQuery对象自身提供一个.get() 方法允许我们直接访问jQuery对象中相关的DOM节点，get方法中提供一个元素的索引

···

# DOM对象转换为jQuery对象

···

Var div1 = document.getElementById(“d1”)

Var $div = $(div1)

···

# 内容选择器

1. :contains与:has都有查找的意思，但是contains查找包含“指定文本”的元素，has查找包含“指定元素”的元素
2. :contains匹配的文本包含在元素的子元素中，同样认为是符合条件的。
3. :parent与:empty是相反的，两者所涉及的子元素，包括文本节点.

:parent 非空选择器

:empty 空选择器

# 可见性选择器

1. :visible 选择可见的元素

1. :hidden 选择隐藏的元素

# 表单选择器

1. :enabled 选择可用的表单元素

1. :disabled 选择禁用的表单元素
2. :checked 选择被选中的input元素
3. :selected 选择被选中的option元素

# html() 和 text()

1. Html()

···

html() 不传入值，就是获取集合中第一个匹配元素的HTML内容

html( htmlString ) 设置每一个匹配元素的html内容

html( function(index, oldhtml) ) 用来返回设置HTML内容的一个函数

//.htm()方法内部使用的是DOM的innerHTML属性来处理的，所以在设置与获取上需要注意的一个最重要的问题，这个操作是针对整个HTML内容（不仅仅只是文本内容）

···

1. text()

···

.text() 得到匹配元素集合中每个元素的合并文本，包括他们的后代

.text( textString ) 用于设置匹配元素内容的文本

.text( function(index, text) ) 用来返回设置文本内容的一个函数

//text()结果返回一个字符串，包含所有匹配元素的合并文本

···

1. 异同

1.html与.text的方法操作是一样，只是在具体针对处理对象不同

2.html处理的是元素内容，.text处理的是文本内容

3.html只能使用在HTML文档中，.text 在XML 和 HTML 文档中都能使用

4.如果处理的对象只有一个子文本节点，那么html处理的结果与text是一样的

5.火狐不支持innerText属性，用了类似的textContent属性，.text()方法综合

2个属性的支持，所以可以兼容所有浏览器

# jQuery创建节点——$(“html结构”)

···

var div = $("<div class='right'><div class='aaron'>动态创建DIV元素节点</div></div>")

$body.append(div)

···

# jQuery插入节点——append()和appendTo()

1. Append()

···

$(select).append($(“html结构”))

···

1. AppendTo()

···

$($(“html结构”)).append(select)

···

1. 区别

append()前面是要选择的对象，后面是要在对象内插入的元素内容

appendTo()前面是要插入的元素内容，而后面是要选择的对象

# jQuery插入节点——prepend()和preendTo()

区别：append在被选元素之后；prepend()在被选元素之前

注意：append、prepend选择元素是选择父元素，然后在父元素中插入相应子元素

# jQuery外部插入节点——after(“元素”)和before(“元素”)

1.after()

···

$(".test2").after('<p style="color:blue">after,在匹配元素之后增加</p>', '<p style="color:blue">多参数</p>')

···

2.before()

···

$(".test1").before('<p style="color:red">before,在匹配元素之前增加</p>', '<p style="color:red">多参数</p>')

···

注意：after、before选择元素是选择元素，然后在元素前后插入相应兄弟元素，与append、prepend的选择对象有区别

# 清空节点内的内容——empty()

···

$(".test2").empty()

···

# 删除节点：包括自身和绑定的事件——remove()

1. 无参

···

$(".test1").remove()

···

2. 有参：参数为选择器

···

$(".test1").remove(":contains('3')")

···

# 删除节点：暂存缓存，添加相同元素依然存在绑定事件—detach()

···

$(".test1").detach()

···

# 克隆节点—clone()

1. 无参,仅克隆元素结构及文本

···

$(".test1").clone()

···

2. 有参：包括克隆事件,参数为true

···

$(".test1").clone(true)

···

# 替换节点—replaceWith()和replaceAll()

1. relaceWith()

···

$(“.test”).replaceWith(“<a href=”javascript:void”>替换</a>”)

···

1. replaceAll()

···

$(“<a href=”javascript:void”>替换</a>”).replaceAll(“.test”)

···

3.区别：选择和替换方式相反

# 包裹节点—wrap()

···

$(‘p’).wrap(“<div></div>”)

···

# 删除包裹节点—unwrap()

···

$(‘p’).unwrap(“<div></div>”)

···

# 多元素包裹节点—wrapAll()

···

$(‘p’).wrapAll(“<div></div>”)

···

# 子元素包裹节点—wrapAll()

···

$(‘p’).wrapInner(“<div></div>”)

···

# 查找匹配元素的子元素节点—children()

···

$("div").children(".selected")

···

# 查找匹配元素的后代元素节点—find()

···

$("div").find(".selected")

···

# 查找匹配元素的父元素节点—parent()

···

$("div").parent()

···

# 查找匹配元素的祖辈元素节点—parents()

···

$("div").parents()

···

# 层级向上查找，直到查找到匹配的第一个元素节点停止——closet()

···

$('li.item-1').closest(itemB).css('border', '1px solid blue');

···

# 查找下一个兄弟元素——next()

···

$('li.item-1').next();

···

# 查找上一个兄弟元素——prev()

···

$('li.item-1').prev();

···

# 查找前后兄弟元素——siblings()

···

$('li.item-2').siblings();

···