jQuery操作DOM

管理科学与工程学科
耿方方

主要内容

- DOM基础
- 访问元素
- 节点操作

DOM基础

- DOM（Document Object Model）为文档提供了结构化表示方法，通过该方法可以改变文档的内容和展示形式。在实际运用中，DOM更像是桥梁，通过它可以实现跨平台、跨语言的标准访问。
- 与DOM密不可分的是JavaScript脚本技术，DOM在Web前端的应用也是基于JavaScript实现的，在前面的章节中我们已经讲到了在JavaScript中是如何进行DOM操作的。
- 单词Document即文档，当我们创建一个页面并加载到Web浏览器时，DOM模型则根据该页面的内容创建了一个文档文件；Object即对象，是指具有独立特性的一组数据集合，比如我们把新创建的页面文档称之为文档对象，与对象相关联的特征称之为对象属性，访问对象的函数称之为对象方法；Model即模型，在页面文档中，通过树状模型展示页面的元素和内容，而其展示方式是通过节点（node）来实现的。

访问元素

- 在访问页面时，需要与页面中的元素进行交互式的操作。在操作中，元素的访问是最繁琐、最常用的，主要包括对元素属性、内容、值、CSS的操作。

访问元素

- 在jQuery中，可以对元素的属性执行获取、设置、删除操作，通过attr()方法可以对元素属性执行获取与设置操作，通过removeAttr()则可以执行删除元素属性操作。
- 我们可以通过attr()方法获取元素的属性，其语法结构如下所示：
- attr(name)

访问元素

- 案例1：
- <script>
- $(function(){
- var strAlt=$('img').attr("src");
- strAlt+="

"+$("img").attr("title");
- $("#divAlt").html(strAlt);
- });
- </script>
- </head>
- <body>
- <img src="Images/7-07.jpg" title="这是一张画">
-
- </body>

访问元素

- 设置元素的属性
- 在页面中，attr()方法不仅可以用来获取元素的属性值，还可以设置元素的属性，其设置元素属性的语法格式如下所示：
- attr(key,value)
- attr({key0:value0,key1:value1})

访问元素

- 设置元素的属性
- 另外，attr()方法还可以绑定一个function()函数,通过该函数返回的值作为元素的属性值。语法格式如下所示：
- attr(key,function(index))
- 其中，参数index为当前元素的索引号，整个函数返回一个字符串作为元素的属性值。

访问元素

- 案例2

```
<style>
 .clsImg{
border: 2px solid #ccc;padding: 3px; float: left;}
</style>
<body>
    <img alt="" src="Images/img03.jpg" width="400" height="300">
</body>
```

```
<script>
    $(function(){
        $("img").attr("title","这是一张风景画");
        $("img").addClass("clsImg");
        $("img").attr("src",function(){
        return "Images/img0"+
            Math.floor(Math.random()*2+1)+".jpg";
    });
});
</script>
```

访问元素

- 删除元素的属性
- jQuery中通过attr()方法设置元素的属性后，使用removeAttr()方法可以将元素的属性删除，其语法结构如下所示：
- removeAttr(name)

访问元素

- 在jQuery中，操作元素内容的方法包含html()和text()，前者与JavaScript中的innerHTML属性相似，即获取和设置元素的HTML内容；而后者类似于JavaScript中的innerText属性，即获取或设置元素的文本内容。二者的格式和功能的区别如下所示：

表 15-13 html()与 text()方法的区别

方法语法	描述	参数说明
html()	用于获取元素的 HTML 内容	无
html(value)	用于设置元素的 HTML 内容	参数为元素的 HTML 内容
text()	用于获取元素的文本内容	无
text(value)	用于设置元素的文本内容	参数为元素的文本内容

- html()方法仅支持XHTML的文档,不能用于XML文档,而text()则既支持HTML文档,也支持XML文档。

访问元素

- 案例3
- <body>
- 应用text()方法设置的内容
-
- 当前时间：
- 2013-08-30 星期五 13:20:10
-
-
应用html()方法设置的内容
-
- 当前时间：2013- 08-30 星期五 13:20:10
-
- </body>

```
<script type="text/javascript">
  $(document).ready(function(){
$("#div1").text("<span style='color:#FF0000'>
我是通过text()方法设置的文本内容</span>");
$("#div2").html("<span style='color:#FF0000'>
我是通过html()方法设置的HTML内容</span>");
  });
</script>
```

访问元素

- jQuery提供了3种对元素值操作的方法。

方　法	说　明	示　例
val()	用于获取第一个匹配元素的当前值，返回值可能是一个字符串，也可能是一个数组。例如当select元素有两个选中值时，返回结果就是一个数组	$("#username").val();　//获取id为username的元素的值
val(val)	用于设置所有匹配元素的值	$("input:text").val("新值")　//为全部文本框设置值
val(arrVal)	用于为checkbox、select和radio等元素设置值，参数为字符串数组	$("select").val(['列表项1','列表项2']);//为下拉列表框设置多选值

访问元素

- 案例4
- <body>
- <select name="like" size="4" multiple="multiple" id="like">
- <option>列表项1</option>
- <option selected="selected">列表项2</option>
- <option>列表项3</option>
- <option selected="selected">列表项4</option>
- </select>
- </body>

```
<script type="text/javascript">
   $(document).ready(function(){
 var strSel=$("select").val().join(",");
alert(strSel);
   });
</script>
```

访问元素

- jQuery对元素的CSS样式操作可以通过修改CSS类或者CSS的属性来实现。
- 通过修改CSS类实现

方　法	说　明	示　例
addClass(class)	为所有匹配的元素添加指定的CSS类名	$("div").addClass("user red"); //为全部div元素添加user和red两个CSS类
removeClass(class)	从所有匹配的元素中删除全部或者指定的CSS类	$("div").removeClass("line"); //删除全部div元素中添加的CSS类line
toggleClass(class)	如果存在（不存在）就删除（添加）一个CSS类	$("div").toggleClass("blue"); //当匹配的div元素中存在CSS类blue，则删除该类，否则添加该CSS类

toggleClass(class,switch)	如果switch参数为true则加上对应的CSS类，否则就删除，通常switch参数为一个布尔型的变量	$("img").toggleClass("show",true); //为img元素添加CSS类show
		$("img").toggleClass("show",false); //为img元素删除CSS类show

访问元素

- 案例5
- <style>
- p{padding: 5px;width: 220px;}
- .cls1{font-weight: bold;font-style: italic;}
- .cls2{border: 1px solid #666;background-color: #eee;}
- </style>
- <script src="jQuery/jquery-3.2.1.js" type="text/javascript"></script>
- <script>
- $(function(){
- $("p").click(function(){
- $(this).addClass("cls1 cls2");
- });
- });
- </script>
- <body>
- Write Less Do More
- </body>

访问元素

- 案例6
- <style>
- .clsImg{border: 1px solid #666;padding: 3px;}
- </style>
- <script src="jQuery/jquery-3.2.1.js" type="text/javascript"></script>
- <script>
- $(function(){
- $("img").click(function(){
- $(this).toggleClass("clsImg");
- });
- });
- </script>
- </head>
- <body>
- <img src="Images/img01.jpg" alt="" title="风景" width="400" height="300">
- </body>

访问元素

- jQuery对元素的CSS样式操作可以通过修改CSS类或者CSS的属性来实现。
- 通过修改CSS属性实现

方　法	说　明	示　例
css(name)	返回第一个匹配元素的样式属性。	$("div").css("color");　//获取第一个匹配的div元素的color属性值
css(name,value)	为所有匹配元素的指定样式设置值	$("img").css("border","1px solid #000000"); //为全部img元素设置边框样式

css(properties)	以{属性：值，属性：值，……}的形式为所有匹配的元素设置样式属性	$("tr").css({
		"background-color":"#0000FF",//设置背景颜色
		"font-size":"16px",
		//设置字体大小
		"color":"#FFFFFF"
		//设置字体颜色
		});

访问元素

- 案例7

```
<style>
        p{padding: 5px;width: 220px;}
    </style>
    <script src="jQuery/jquery-3.2.1.js" type="text/javascript"></script>
    <script>
        $(function(){
            $("p").click(function(){
                $(this).css("font-weight","bold");
                $(this).css("font-style","italic");
                $(this).css("background-color","#eee");
            })
        })
    </script>
</head>
<body>
    <p>Write Less Do More</p>
</body>
```

节点操作

- 整个页面是一个DOM模型，页面中各元素通过模型的节点相互关联形成树状。因此，如果要在页面中增加某个元素，只需要找到元素的上级节点，通过函数$(html)完成元素创建后，增加到该节点中。
- 函数$()用户动态创建页面元素，其语法格式如下：
- $(html)
- 其中，参数html表示用于动态创建DOM元素的HTML标记字符串，即如果要在页面中动态创建一个div标记，并设置其内容和属性，可以加入如下代码：
- var $div=$(“ write less do more ”);
- $(“body”).append($div);

节点操作

- 另外一种方法就是先创建新元素，二是将新元素插入到文档中。例如：
- var $p=$(“ ”);
- $p.html(“要添加的内容”);
- $(“body”).append($div);

节点操作

- 插入结点
- 在jQuery中，有很多的方法可以实现该功能，我们前面例子中用到的append()方法仅仅是其中的一种，按照插入元素的顺序类划分，可以将插入节点分为内部和外部两种插入方法。
- 内部插入节点
- 在元素内部插入节点就是向元素中添加子元素和内容。

表 15-10 内部插入节点方法

方法语法	描述	参数说明
append(content)	向所选择的元素内部插入内容	content：追加到目标中的内容
append(function(index,html))	向所选择的元素内部插入 function 函数返回的内容	通过函数返回追加到目标中的内容
appendTo(content)	把所选择的元素追加到另一个指定的元素集合中	content：被追加的内容
prepend(content)	向每个所选择的元素内部前置内容	content：插入目标元素内容前面的内容
prepend(function(index,html))	向所选择的元素内部前置 function 函数返回的内容	通过函数返回插入目标元素内部前面的内容
prependTo(content)	将所选择的元素前置到另一个指定的元素集合中	content：用于选择元素的 jQuery 表达式

节点操作

- 案例8
- <script>
- $(function(){
- $("div").append(retHtml);
- function retHtml(){
- var str="Write Less Do More";
- return str;
- };
- });
- </script>
- </head>
- <body>
- jQuery
- </body>

节点操作

- 案例9
- <script type="text/javascript">
- $(document).ready(function(){
- $("button").click(function(){
- $(" Hello World!").appendTo("p");
- });
- });
- </script>
- </head>
- <body>
- This is a paragraph.
- This is another paragraph.
- <button>在每个 p 元素的结尾添加内容</button>
- </body>

节点操作

- 插入结点
- 外部插入节点

表 15-11 外部插入节点方法

方法语法	描述	参数说明
after(content)	向所选择的元素外部后面插入内容	插入目标元素外部后面的内容
after(function)	向所选择的元素外部后面插入 function 函数返回的内容	通过函数返回插入目标外部后面的内容
before(content)	向所选择的元素外部的前面插入内容	插入目标元素外部前面的内容
before(function)	向所选择的元素外面前面插入 function 函数返回的内容	通过函数返回插入目标外部前面的内容
insertAfter(content)	将所选择的元素插入另一个指定的元素外部后面	插入目标元素外部后面的内容
insertBefore(content)	将所选择的元素插入到另一个指定的元素外部前面	插入目标元素外部前面的内容

节点操作

- 案例10

```
<script>
            $(function(){
                $("span").after(retHtml);
                function retHtml(){
                    var str="<span><b>Write Less Do More</b></span>";
                    return str;
                };
            });
        </script>
    </head>
    <body>
        <span>jQuery</span>
    </body>
```

节点操作

- 复制节点
- 在页面中，有时候需要将某个元素节点复制到另外一个节点后，在jQuery中可以通过clone()方法来实现节点的复制，其语法结构如下所示：
- clone();

节点操作

复制节点

- 案例11
- <script>
- $(function(){
- $("img").click(function(){
- $(this).clone(true).appendTo("span");
- })
- });
- </script>
- </head>
- <body>
- <img src="Images/img01.jpg" alt="" width="200" height="100">
- </body>

节点操作

- 替换节点
- 在jQuery中，如果要替换元素中的节点，可以使用replaceWith()和replaceAll()这两种方法，其语法结构如下所示：
- replaceWith(content)
- replaceAll(selector)

节点操作

替换节点

- 案例12

```
<script>
        $(function(){
            $("#span1").replaceWith("<span>套郭荣</span>");
            $("<span>gfhactcm@126.com</span>").replaceAll("#span2");
        });
    </script>
</head>
<body>
    <p>姓名:<span id="span1"></span></p>
    <p>邮箱:<span id="span2"></span></p>
</body>
```

节点操作

- 包裹结点
- 在jQuery中，不仅可以通过方法替换元素节点，还可以根据需求包裹某个指定的节点，对节点的包裹也是DOM对象操作中很重要的一项，其与包裹节点相关的全部方法如下表所示：

表 15-12 包裹节点

方法语法	描述	参数说明
wrap()	把所有选择的元素用其它字符串代码包裹起来	html 参数为字符串代码，用于生成元素并包裹所选元素
wrap(elem)	把所有选择的元素用其他 DOM 元素包装起来	elem 参数用于包装所选元素的 DOM 元素
wrap(fn)	把所有选择的元素用 function 函数返回的代码包裹起来	fn 参数为包裹结构的一个函数
unwrap()	移除所选元素的父元素或包裹标记	无
wrapAll(html)	把所有选择的元素用单个元素包裹起来	html 参数为字符串代码，用于生成元素并包裹所选元素
wrapAll(elem)	把所有选择的元素用单个 DOM 元素包裹起来	elem 参数用于包装所选元素的 DOM 元素
wrapInner(html)	把所有选择的元素的子内容（包括文本节点）用字符串代码包裹起来	html 参数为字符串代码，用于生成元素并包裹所选元素
wrapInner(elem)	把所有选择的元素的子内容（包括文本节点）用 DOM 元素包裹起来	elem 参数用于包装所选元素的 DOM 元素
wrapInner(fn)	把所有选择的元素的子内容（包括文本节点）用函数返回的代码包裹起来	fn 参数为包裹结构的一个函数

节点操作

- 案例13

```
<script>
        $(function(){
            $("#span1").replaceWith("<span>套郭荣</span>");
            $("<span>gfhactcm@126.com</span>").replaceAll("#span2");
        });
    </script>
</head>
<body>
    <p>姓名:<span id="span1"></span></p>
    <p>邮箱:<span id="span2"></span></p>
</body>
```

节点操作

- 遍历节点
- 在DOM元素操作中，有时需要对统一标记的全部元素进行统一操作。在JavaScript中，需要先获取元素的总长度，然后用for循环语句，循环处理。而在jQuery中可以直接使用each()方法轻松实现元素的遍历，其语法结构如下所示：
- each(callback)

节点操作

遍历节点

- 案例14

```
<script>
            $(function(){
                $("img").each(function(index){
                    this.title="第"+ index+"风景图片,alt内容是"+this.alt;
                });
            });
        </script>
    </head>
    <body>
        <img src="Images/img01.jpg" alt="第0幅风景图" width="300" height="160">
        <img src="Images/img02.jpg" alt="第1幅风景图" width="300" height="160">
        <img src="Images/img03.jpg" alt="第2幅风景图" width="300" height="160">
    </body>
```

节点操作

- 删除节点
- 在DOM操作页面时，删除多余或指定的页面元素时非常必要的，jQuery提供了两种可以删除元素的方法，即remove()和empty()，严格的说empty()方法并非真正意义上的删除，使用该方法，仅仅可以清空全部的节点或节点所包含的所有后代元素，并非删除节点与元素。
- remove()方法的语法结构如下所示：
- remove([expr])
- empty()方法语法结构如下所示：
- empty()

节点操作

- 案例15

```
<script>
        $(function(){
            $("button").click(function(){
                $("p").remove();
            });
        });
    </script>
</head>
<body>
    <p>这是一个段落。</p>
<p>这是另一个段落。</p>
<button>删除所有 p 元素</button>
</body>
```