高级Day 11---Jquery选择器

一、Jquery入门

1.1 什么是Jquery

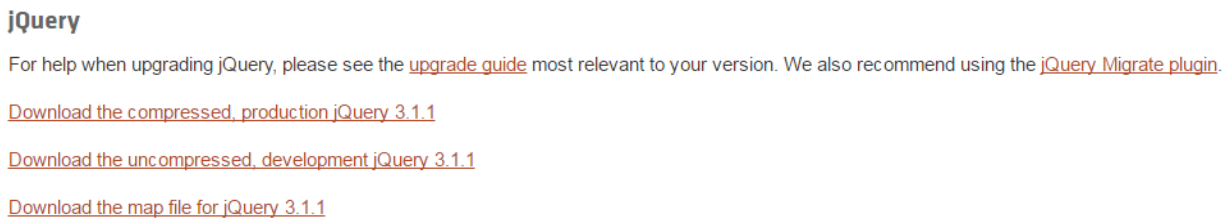
jQuery是一个快速、简洁的JavaScript框架，是继Prototype之后又一个优秀的JavaScript代码库（或JavaScript框架）。jQuery设计的宗旨是“write Less，Do More”，即倡导写更少的代码，做更多的事情。它封装JavaScript常用的功能代码，提供一种简便的JavaScript设计模式，优化HTML文档操作、事件处理、动画设计和Ajax交互。

jQuery的核心特性可以总结为：具有独特的链式语法和短小清晰的多功能接口；具有高效灵活的css选择器，并且可对CSS选择器进行扩展；拥有便捷的插件 扩展机制和丰富的插件。jQuery兼容各种主流浏览器，如IE 6.0+、FF 1.5+、Safari 2.0+、Opera 9.0+等。

1.2 下载Jquery

去官网下载：

jquery官网



目前最新的是3.1.1版本。有压缩版本和不压缩版本。不压缩版本一般用到开发阶段，压缩版本用到生产环境。从2.X开始不支持ie9之前的浏览器。如果想兼容IE678，则应该使用1.X

1.3 使用Jquery

jquery就是一个js文件，使用它就像我们时候用一个外部js文件一样使用即可。一般放在html的head中引入。

```
<script src="./libs/jquery-3.1.1.js" type="text/javascript"></script>
```

二、Jquery核心基础

2.1 Jquery的入口

写js代码一般等到整个页面加载完成之后，才去执行js操作，尤其涉及到对dom的操作。
JQuery的使用也是如此。一般使用下面的入口来完成。

```
<script type="text/javascript">
    //带整个文档加载完毕之后再去执行我们传入的匿名函数。   document.onload = function () {}
    $(document).ready(function () {
        //jquery代码
    });
    //上面的入口可以简写成如下形式,完全与前面的的等价
    $(function () {

    });
</script>
```

> 说明：

- $ 是jquery中的对象。我们使用jquery的所有操作，都是在 $ 的基础上完成的
- 其实 $ 是 jQuery 对象的简称。 $ === jQuery

2.2 JQuery对象和DOM对象的互换

使用JQuery获得的是JQuery对象，使用原生js获得的DOM对象，二者是完全不同的对象。使用的时候一定要区别开来。

```
<script type="text/javascript">
    $(function () {
        var $div = $("#div");//JQuery操作节点，得到的是JQuery节点对象。Jquery对象一般用$开头来表示
        var div = document.getElementById("div"); //得到的DOM对象

        /*
            DOM对象转换成JQuery对象
            $(domObj)
         */
        var $div1 = $(div);

        /*
            JQuery对象用两种方式转变成DOM对象：
            1、利用数组下标方式
                JQueryObj[0]
            2、利用get方法：
                JqueryObj.get[0]
         */
        var domDiv1 = $div[0];
        var domDiv2 = $div.get(0);
    });
</script>
```

2.3 onload和JQuery中ready方法的区别

1. 执行时机：onload事件必须等页面完全加载完毕后才能执行；ready当页面节点加载完毕后就可以执行。比onload要早一点
2. 添加个数：onload事件只能添加一个，如果添加了多个，则最后执行的onload事件会覆盖前边的事件;ready事件可以添加多个，且互相之间不会覆盖。（onload事件和ready时间之间也不会互相覆盖）
3. 简化写法：onload没有简化写法；ready事件可以简化为:$(function(){});

三、JQuery简单选择器

JQuery中选择器的选择器的思路和CSS的选择器的思路一样，就是去选择元素的。有三种基本选择器。

3.1 ID选择器

```
<script type="text/javascript">
    $(function() {
        //通过id找到元素。注意返回的jquery对象
        var $box = $("#box");

    })
</script>
```

3.2 标签选择器

```
<script type="text/javascript">
    $(function() {
        //找到为a的标签
        var $a = $('a');
    })
</script>
```

3.3 类选择器

```
<script type="text/javascript">
  $(function() {
      var $link = $(".myLink");
      console.log($link);
})
</script>
```

四、JQuery进阶选择器

4.1 群组选择器

```
<script type="text/javascript">
  $(function() {
      var $elements = $(".box,#toBaidu,h1");     //可以同时选中多个元素.不同的选择器用,隔开
      console.log($elements);
  })
 </script>
```

4.2 后代选择器

```
$(function () {
    var $as = $("p a"); //找到p标签下的所有a标签
    console.log($as);
})
```

4.3 通配符选择器

```
$(function() {
  var $all = $("*");     //获取当前文档中所有元素的个数
  console.log($all);
  alert($all.length);
})
```

五、JQuery高级选择器

5.1 后代选择器和find方法

jquery对象.find(选择器)：是jquery对象的方法，表示找到这个标签下的所有的指定的选择器的元素。

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
        <script src="libs/jquery-3.1.1.js" type="text/javascript" charset="utf-8"></script>

        <script type="text/javascript">
            $(function () {
                var $allLi1 = $(".box li");
                console.log($allLi1);
                //找到 class是box的元素下面的所有的li标签元素
                var $allLi2 = $(".box").find("li"); // 效果等同于前面的后代选择器。
                console.log($allLi2);
            })
        </script>
    </head>
    <body>
        <div class="box">
            <ul>
                <li>第一个li的内容</li>
                <li>第二个li的内容</li>
                <li>第三个li的内容</li>
            </ul>
        </div>
    </body>
</html>
```

5.2 子元素选择器和children方法

```
<script type="text/javascript">
    $(function () {
        var $divs = $(".box>div"); //子元素选择器
        console.log($divs);
        // 找到.box 元素下的所有直接子元素。只找儿子不找孙子。 和子元素选择器后果一样。
        // children方法也可以带参数，代表表示这个选择器选中的子标签
        var $children = $(".box").children();
        console.log($children);

        var $cc = $(".box").children(".item1");
        console.log($cc);
    })
</script>
```

5.3 Next选择(+)和next方法

```
<script type="text/javascript">
    $(function () {
        //next选择器其实就是兄弟选择器.
        //找到.box .item2 后的 是 .item3的紧挨着的下一个兄弟
        var $nextDiv1 = $(".box .item2 + .item3");
        console.log($nextDiv1);
        //同next选择器. 可以跟参数，也可以不跟参数。有参数 表示下一个兄弟必须满足这个条件,否则就拉到
        //不跟参数表示如果有下一个兄弟就返回，没有拉到
        var $nextDiv2 = $(".box .item2").next(".item3");
        console.log($nextDiv2);
    })
</script>
```

5.4 nextAll选择器(~)和nextAll方法

```
<script type="text/javascript">
    $(function () {
        //获取.item2的所有的同辈div标签  注意：不包括.item这个标签
        var $divs = $(".item2~div");
        console.log($divs);
        //也可以使用nextAll方法，效果同上。  可以不跟参数，表示后面的所有同辈元素
        var $divss = $(".item2").nextAll("div");
        console.log($divss);
    })
</script>
```

5.5 prev方法和prevAll方法

```
<script type="text/javascript">
    $(function () {
        //找到紧挨着这个的上一个同辈div元素。 如果不是div则拉到。
        //也可以不给参数，表示返回上一个同辈元素
        var $prev = $(".item2").prev("div");
        console.log($prev);
        //获取.item5的所有的前面的同辈div元素。
        var $prevAll = $(".item5").prevAll("div");
        console.log($prevAll);
    })
</script>
```

5.5 sibling方法

```
<script type="text/javascript">
    $(function () {
        //sibling获取的是所有同辈标签
        var $sibling = $(".item2").siblings();
        console.log($sibling);
    })
</script>
```

六、属性选择器

与元素的属性相关的选择器。属性选择器必须用 [] 括起来。

```
<script type="text/javascript">
    $(function () {
        // 找到有id属性的所有元素
        var $ids1 = $("[id]");
        console.log($ids1);
        //找到.box的所有后代中有id属性的元素
        var $ids2 = $(".box [id]");
        console.log($ids2);

        //找到id等于id的元素
        var $ids3 = $("[id=id4]");
        console.log($ids3);
        //id不是box的都会被选中，没有id属性的也算进去
        var $ids4 = $("[id!=box]");
        console.log($ids4)
        // id属性以b开头的
        var $ids5 = $("[id^=b]");
        console.log($ids5)

        // id属性的值包含b的
        var $ids6 = $("[id*=b]");
        console.log($ids6)
    })
</script>
```

七、过滤选择器

使用特定的过滤规则来筛选出所需的DOM元素。过滤选择器的语法和css中的伪类的写法一样，都是用:开头。

7.1 基本过滤选择器

```
<script type="text/javascript">
    $(function() {
        //所有的div元素中的第一个div
        var $first = $("div:first");
        console.log($first);
        // 所有div元素中的最后一个div
        var $last = $("div:last");
        console.log($last);
        //所有div元素中，class不是box的div
        console.log($("div:not(.box)"));
        // 所有的div元素中，索引是偶数的div
        console.log($("div:even"));
        // 所有div元素中，索引是奇数的div
        console.log($("div:odd"));
        // 所有div元素中，索引是0的div
        console.log($("div:eq(0)"));
        // 所有的div元素中，索引大于0的div
        console.log($("div:gt(0)"));
        //所有的div元素中 索引小于3的div
        console.log($("div:lt(3)"));
        //获取所有的标题元素
        console.log($(":header"));
        //获取当前取得焦点的元素
        console.log($(":focus"));
    })
</script>
```

7.2 内容过滤选择器

选择器	描述
:contains(text)	选取含有文本内容为text的元素
:empty	选取不包含子元素或文本的元素
:has(selector)	选择与指定的选择匹配的元素
:parent	选取含有子元素或文本的元素

```
<script type="text/javascript">
    $(function() {
        //选取包含文本 "标签" 的div元素。 如果div的子元素满足，那么这个div也算。
        console.log($("div:contains(标签)"));
        //选取，没有子标签或文本的div元素
        console.log($("div:empty"));
        //选取有 后代 是.item2的div元素
        console.log($("div:has(.item2)"))
        // 选取有文本或子元素的所有元素。  也就是能当爹的元素
        console.log($(":parent"));
    })
</script>
```

选择器	描述

7.3 可见性选择过滤器

根据元素的可见状态来选取。

选择器	描述
:hidden	选取所有不可见的元素。包括input的type属性是hidden，display是none。
:visible	选取所有可见的元素。如果一个元素的visibility是hidden也会被选中。

```
<script type="text/javascript">
    $(function() {
        console.log($("div:visible"));
        console.log($("div:hidden"));
    })
</script>
```

7.4 子元素过滤选择器

选择器	描述
:nth-child(index/even/odd/equation)	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.
:first-child	相当于nth-child(1)
:last-child	返回父元素的第一个子元素
:only-child	如果父类元素仅仅有一个子元素就返回

```
<script type="text/javascript">
    $(function() {
        // 从p元素的父元素的所有子元素中查找。如果第2个元素是p，则返回这个p元素，如果不是p则不返回这个子元素
        console.log($("p:nth-child(2)"));
        //从p元素的父元素的所有子元素中查找。如果第一个元素是p则返回这个p元素。否则不返回
        console.log($("p:first-child"));
        console.log($("p:last-child"));
        console.log($("div:only-child"));
    });
</script>
```

八、表单选择器

为了更加方便的操作表单，jquery专门添加的表单选择器

选 择 器	描 述	返 回	示 例
:input	选取所有的<input>、<textarea>、<select> 和<button>元素	集合元素	$(":input")选取所有<input>、<textarea>、<select>和<button>元素
:text	选取所有的单行文本框	集合元素	$(":text")选取所有的单行文本框
:password	选取所有的密码框	集合元素	$(":password")选取所有的密码框
:radio	选取所有的单选框	集合元素	$(":radio")选取所有的单选框
:checkbox	选取所有的多选框	集合元素	$(":checkbox")选取所有的复选框
:submit	选取所有的提交按钮	集合元素	$(":submit")选取所有的提交按钮
:image	选取所有的图像按钮	集合元素	$(":image")选取所有的图像按钮
:reset	选取所有的重置按钮	集合元素	$(":reset")选取所有的重置按钮
:button	选取所有的按钮	集合元素	$(":button")选取所有的按钮
:file	选取所有的上传域	集合元素	$(":file")选取所有的上传域
:hidden	选取所有不可见元素	集合元素	$(":hidden")选取所有不可见元素（已经在不可见性过滤选择器中讲解过）

```
<script type="text/javascript">
    $(function() {
        //获取所有的input、button、select、textarea
        console.log($(":input"));
        //选取所有的单行文本框 (type=text)
        console.log($(":text"));
        console.log($(":password"));
        console.log($(":radio"));
        console.log($(":checkbox"));
        console.log($(":submit"));
        console.log($(":button"));
    });
</script>
```