jQuery

2019年5月7日 星期二 16:22

1. jQuery概述
   a. 由于用户在js中编写的内容十分繁多，将一些常用的操作和行为进行了封装，封装成很多个函数，这些函数就构成了一个js的函数库，用户在使用的时候直接调用函数库的函数即可。这个函数库就是jQuery。
   b. jQuery可以写的更少但是做的更多。
   c. jQuery本质是js.
2. jQuery的优势：
   简化javascript代码。
   可以像css选择器一样选择元素。
   可以像css一样修改页面中的样式。
   可以兼容常用的浏览器。
3. jQuery版本支持

   jQuery分为很多版本，还分为未压缩版和压缩版，根据需要选择对应的版本进行下载!

   1.x 支持常用的浏览器和IE6+

   2.x 支持常用的浏览器和IE9+

   3.x 支持常用的浏览器和IE9+

   注意: jQuery不兼容老版本. 因为jQuery升级除了会做一些内部优化之外，还会删除以前的一些代码，比如删除一些方法. 或者是添加一些新的方法。所以在升级之后， 以前的代码可能会无法执行。
4. jQuery的引入

   需要在页面中引入一个js文件，这个js文件就是jQuery的函数库，只有引入函数库之后才能够使用jQuery代码。

   <script src="jquery-1.4.2.js"></script>
5. jQuery的使用方式：

   $()

   $就是jQuery的意思 $("div") <==> jQuery("div")

6. js对象和jQuery对象的相互转换
   a. js对象->jQuery对象

      将已有的js对象使用$()包起来即可由js对象变为jQuery对象。

   ```
   var div = document.getElementById("test");
   ```

   转换为jQuery对象：

   $(div).val()

   b. jQuery对象->js对象

   ```
   //jQuery对象->js对象
   ```

   jQuery对象一般利用选择器选中页面中的元素，选择时可能会选中很多个元素，所以这些元素被装入到了一个数组当中，将数组中的元素取出，这是取出的元素就是一个js对象。数组中有几个元素，就可以取出几个js对象。

```
var div = $("div")[0];
        = $("div").get(0);
div.innerText= "ccc";
```

注意：

```
var $div = $("div");
var div = $("div");
```

此处$div和div表示的内容是同一个jQuery对象，$div这种命名的方式只是想表示当前值是一个jQuery的对象。

7. jQuery主要用法：
   a. 在jQuery中书写一个CSS选择器 --- $("div")，这样就可以将这种形式作为jQuery选择器来使用。
   b. 动态创建dom元素---直接书写HTML元素名称来创建代表HTML元素的jQuery对象

```
$("<div><p>Hello</p></div>").appendTo("body");
```

   c. 利用jQuery控制css样式

```
$(document.body).css( "background", "black" );
    选中的元素      .css.(  "属性名称","属性值");
```

   d. 文档就绪事件 --- 在整个HTML页面加载完成之后再执行其中的js代码。

js：

```
window.onload =function(){

}
```

jQuery： ready()函数就是文档就绪事件

```
$(document).ready(function (){

});
<==>
$().ready(function(){

})
<==>
$(funtion(){

});
```

jQuery选择器

2019年5月9日 星期四 9:31

1. 基本选择器（selector案例一）

   元素名选择器，类选择器，id选择器，多元素选择器，*选择器

   1. 元素名选择器

      $("div") – 匹配所有的div元素

   2. class选择器

      $(".c1") – 匹配所有class值为c1 的元素

      $("div.c1") – 匹配所有class值为c1的div元素

   3. id选择器

      $("#d1") – 匹配所有id值为d1的元素

      $("div#d1") – 匹配所有id值为d1的div元素

   4. *号匹配符

      $("*") – 匹配所有的元素

   5. 多元素选择器

      $("div,span,#d1,.c1") – 匹配所有的div/span元素以及id值为d1的元素和class值为c1的元素。。。

2. 层级选择器（selector案例二）

   如果想通过DOM元素之间的层次关系来获取特定元素。例如子元素、兄弟元素等。则需要通过层次选择器。

   $("div span") – 匹配div下所有的span元素

   $("div>span") – 匹配div下所有的span子元素

   $("div+span") – 匹配div后面紧邻的span兄弟元素

   $("div~span") – 匹配div后面所有的span兄弟元素

3. 基本过滤选择器（selector案例三）

   过滤选择器主要是通过特定的过滤规则来筛选出所需的DOM元素，该选择器都以 ":" 开头

   $("div:first") – 匹配所有div中的第一个div元素

   $("div:last") – 匹配所有div中的最后一个div元素

   $("div:even") – 匹配所有div中索引值为偶数的div元素，0开始

   $("div:odd") – 匹配所有div中索引值为奇数的div元素，0开始

   $("div:eq(n)") – 匹配所有div中索引值为n的div元素，0开始

   $("div:lt(n)") – 匹配所有div中索引值小于n的div元素，0开始

   $("div:gt(n)") – 匹配所有div中索引值大于n的div元素，0开始

$("div:not(.one") – 匹配所有class值不为one的div元素

4. 内容选择器（selector案例四）

$("div:contains('abc')") – 匹配所有div中包含abc内容的div元素

如: xxxabcxx

$("div:has(p)") – 匹配所有包含p元素的div元素

如:

$("div:empty") – 匹配所有内容为空的div元素

如:

$("div:parent") – 匹配所有内容不为空的div元素

如: xxxxx

5. 可见选择器

$("div:hidden") – 匹配所有隐藏的div元素

$("div:visible") – 匹配所有可见的div元素

6. 属性选择器（selector案例五）

$("div[id]") – 匹配所有具有id属性的div元素

$("div[id='d1']") – 匹配所有具有id属性并且值为d1的div元素

$("div[id!='d1']") – 匹配所有id属性值不为d1的div元素

$("div[id^='d1']") – 匹配所有id属性值以d1开头的div元素 id=d1xxxxx id=d1zzszzz

$("div[id$='d1']") – 匹配所有id属性值以d1结尾的div元素 id=xxxxxd1 id = zzzzzd1

7. 子元素选择器

$("div:nth-child(n)") – n从1开始, 匹配div中第n个子元素。

$("div:first-child") –匹配div中第1个子元素。

$("div:last-child") –匹配div中最后一个子元素。。。

8. 表单选择器（selector案例六）

$(":input") – 匹配所有的input文本框、密码框、单选框、复选框、select框、textarea、button。

$(":password") – 匹配所有的密码框

$(":radio") – 匹配所有的单选框

$(":checkbox") – 匹配所有的复选框

$(":checked") – 匹配所有的被选中的单选框/复选框/option

$("input:checked") – 匹配所有的被选中的单选框/复选框

$(":selected") – 匹配所有被选中的option选项

注意：

$("input")表示元素名选择器，选择的元素为input元素，只能匹配名称为input的元素。

文档操作

2019年5月9日 星期四 11:39

1. 文档操作

parent()

$("#d1").parent() – 获取id为d1元素的父元素

parents()

$("#d1").parents() – 获取id为d1元素的祖先元素

$("#d1").parents("tr") – 获取id为d1元素的tr祖先元素

```
<div>
    <tr>
        <div>
            <span id="d1">
```

next()

```
<a></a>
<div></div>
<a></a>
<span></span>
```

$("a").next() – 获取所匹配元素后面紧邻的兄弟元素

$("div").next("span") – 获取所匹配元素后面紧邻的span兄弟元素

nextAll()

```
<div></div>
<span></span>
<p></p>
<span></span>
```

$("div").nextAll() – 获取所匹配元素后面所有的兄弟元素

$("div").nextAll("span") – 获取所匹配元素后面所有的span兄弟元素

prev()

$("div").prev() – 获取所匹配元素前面紧邻的兄弟元素

$("div").prev("span") – 获取所匹配元素前面紧邻的span兄弟元素

prevAll()

$("div").prevAll() – 获取所匹配元素前面所有的兄弟元素

$("div").prevAll("span") – 获取所匹配元素前面所有的span兄弟元素

siblings()

$("div").siblings() – 获取所匹配元素前后所有的兄弟元素

$("div").siblings("span") – 获取所匹配元素前后所有的span兄弟元素

append()

$("div").append("") –为所匹配元素追加一个span子元素

remove()

$("div").remove() – 删除所匹配元素

html()

$("div").html() – 获取所匹配元素的html内容

$("div").html("xxx") – 为所匹配元素设置html内容

text()

$("div").text() – 获取所匹配元素的文本内容

$("div").text("xxx") – 为所匹配元素设置文本内容

attr()

$("div").attr("id") – 获取所匹配元素的id属性值

$("div").attr("id", "xx") – 为所匹配元素设置id属性

css

$("div").css("width") – 获取所匹配元素的width样式属性值

$("div").css("width", "200px") – 为所匹配元素设置width样式属性

$("div").css({"width":"200px", "color":"red","font-size":"24px" }); – 为所匹配元素设置width样式属性

2. 事件

click()

$("div").click(function(){}) – 为所匹配元素绑定点击事件

blur()

$("input").blur(function(){}) – 为所匹配元素绑定失去输入焦点事件

focus()

$("input").focus(function(){}) – 为所匹配元素绑定获得输入焦点事件

change()

$("select").change(function(){}) – 为所匹配元素绑定选项切换事件

ready()

$(document).ready(function(){}) – 文档就绪事件

其作用相当于: window.onload = function(){}

简写形式为:

$(function(){}) – 在整个文档加载完成后立即执行

3. 效果

.css("display","block");

show()

$("div").show() – 将隐藏元素设置为显示(底层操作的是display);

.css("display","none");

hide()

$("div").hide() – 将显示元素设置为隐藏(底层操作的是display);

toggle()

$("div").toggle() – 切换元素的可见状态, 如果元素显示则设置为隐藏, 如果元素隐藏则设置为可见.

(Exercise案例)