jQuery选择器

管理科学与工程学科
耿方方

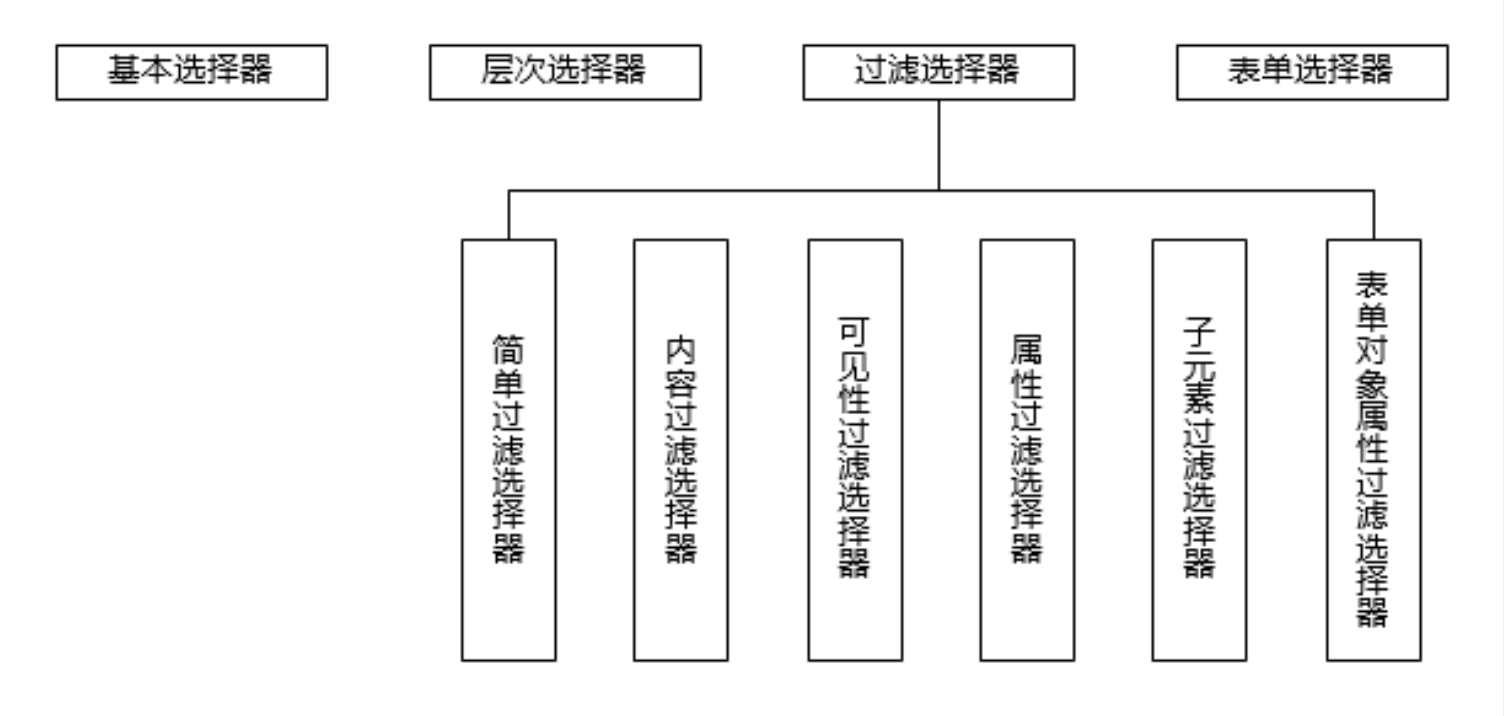
主要内容

- 什么是选择器
- 选择器的优势
- 选择器详述

什么是选择器

- 在页面中为某个元素添加属性或事件时，必须先准确地找到元素——在jQuery库中，可以通过选择器实现这一重要核心功能。
- 选择器是jQuery的根基，在jQuery中无论是对事件处理、遍历DOM还是Ajax操作都依赖于选择器，熟练的使用选择器不仅能简化代码，而且可以达到事半功倍的效果。
- 根据所获取页面中元素的不同，可以将jQuery选择器分为：基本选择器、层次选择器、过滤选择器、表单选择器四大类。而过滤选择器又可分为：简单过滤选择器、内容过滤选择器、可见性过滤选择器、属性过滤选择器、子元素过滤选择器、表单对象过滤选择器六种。

什么是选择器


基本选择器
层次选择器
过滤选择器
表单选择器
简单过滤选择器
内容过滤选择器
可见性过滤选择器
属性过滤选择器
子元素过滤选择器
表单对象属性过滤选择器

选择器的优势

- 与传统的JavaScript获取页面元素和编写事务相比，jQuery选择器具有明显的优势，具体表现在以下两个方面：
- 代码更简单

由于在jQuery库中，封装了大量可以通过选择器直接调用的方法或函数，使编写代码更加简单轻松，简单几行代码就可实现较为复杂的功能：

选择器的优势

- 完善的检测机制

在传统的JavaScript代码，给页面中某元素设置事务时必须先找到该元素，否则，浏览器将提示出错信息，影响后续代码的执行。因此，为避免出错，先检测元素是否存在，然后再运行代码，从而导致代码冗余，影响执行效率。

在jQuery选择器定位页面元素时，无需考虑所定位的元素在页面中是否存在，即使该元素不存在，浏览器也不提示出错信息，极大地方便了代码的执行效率。

选择器详述

- 基本选择器

基本选择器是jQuery中使用最频繁的选择器，它由元素id、class、元素名、多个选择符组成，通过基本选择器可以实现大多数页面元素的查找，其具体使用说明如下表15-01所示：

表 15-01 基本选择器语法

选择器	描述	返回值
#id	根据提供的 id 属性值匹配一个元素	单个元素
element	根据提供的元素名匹配所有的元素	元素集合
.class	根据提供的类名称匹配所有的元素	元素集合
*	匹配所有元素	元素集合
selector1, selectorN	将每一个选择器匹配到的元素合并后一起返回	元素集合

选择器详述

- id

ID选择器是“#id”，顾名思义就是利用DOM元素的id属性值来匹配的元素，并以jQuery包装集的形式返回给对象。

ID选择器的使用方法：

```
$("#id");
```

其中id为要查元素的id属性值。例如，要查询id属性值为username的元素，可以使用下面的jQuery代码：

$(“#username”)

选择器详述

基本选择器

实例1：

```
<html>
    <head>
        <title>ID选择器</title>
        <meta charset="UTF-8">
            <script src="jQuery/jquery-3.2.1.js" type="text/javascript"></script>
        <script>
            $(function(){
                $("input[type='button']").click(function(){
                    var inputValue=$("#test").val();
                    alert(inputValue);
                });
            });
        </script>
    </head>
    <body>
        <input type="text" id="test" name="test" value=""/>
        <input type="button" value="查看输入的值"/>
    </body>
</html>
```

选择器详述

- 元素选择器

元素选择器是根据元素名称匹配相应的元素。

ID选择器的使用方法：

```
$("element");
```

其中element为要查询元素的标记名。例如，要查询全部的div元素，可以使用下面的jQuery代码：

$(“div”)

选择器详述

案例2：<html>

```
<head>
    <title>元素选择器</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <script src="jQuery/jquery-3.2.1.js" type="text/javascript"></script>
    <script>
        $(function(){
            $("input[type='button']").click(function(){
                $("div").eq(0).html("<img src='Images/10-strawberry1.jpg'/>这里长出了一片草莓");
                $("div").get(1).innerHTML="<img src='Images/10-fish1.jpg'/>这里的鱼没有了";
            });
        });
    </script>
</head>
```

```
<body>
    <div><img src="Images/10-strawberry.jpg"/>这里种了一棵草莓</div>
    <div><img src="Images/10-fish.jpg"/>这里养了一条鱼</div>
    <input type="button" value="几年后" />

</body>
```

选择器详述

- 类名选择器

元素选择器是通过元素拥有的CSS类的名称查找匹配的DOM元素。

ID选择器的使用方法：

```
$(".class");
```

其中class为要查询元素的类名。例如，要查询使用CSS类名为mybook的元素，可以使用下面的jQuery代码：

$(“.mybook”)

选择器详述

案例3:

```
<html>
    <head>
        <title>元素选择器</title>
        <meta charset="UTF-8">
         <script src="jQuery/jquery-3.2.1.js" type="text/javascript"></script>
        <script>
            $(function(){
                $(".myClass").css("background-color","#C50210");
                $(".myClass").css("color","#FFF");
            })
        </script>
    </head>
    <body>
        <div>默认的样式</div>
        <div class="myClass">更改后的样式</div>
    </body>
</html>
```

选择器详述

- 复合选择器

复合选择器是将多个选择器（可以是ID选择器、元素选择器或是类名选择器）组合在一起；两个选择器之间以逗号“，”分隔，只要符合其中的任何一个筛选条件就会被匹配，返回的是一个集合形式的jQuery包装集，利用jQuery索引器可以取得集合中的jQuery对象。

复合选择器的使用方法：

```
$(" selector1,selector2,selectorN");
```

选择器详述

案例4：

```
<script>
            $(function(){
                $("input[type=button]").click(function(){
                    $("div,#span").addClass("change");
                });
            });
            </script>
    </head>
    <body>
        <div class="default">div元素</div>
          <p class="default">p元素</p>
         <span class="default" id="span">id为span的元素</span>
         <input type="button" value="为div元素和id为span的元素换肤" />
    </body>
```

选择器详述

- 通配符选择器

所谓通配符，就是指符号“*”，它代表页面上的每一个元素，也就是使用$(“*”),将取得页面上所有的DOM元素集合的jQuery包装集。

选择器详述

- 层次选择器
- 通过DOM元素间的层次关系获取元素，其主要的层次关系包含后代、父子、相邻、兄弟关系，通过其中某类关系可以方便快捷的定位元素，其具体使用说明如下表15-02所示：

表 15-02 层次选择器语法

选择器	描述	返回值
ancestor descendant	根据祖先元素匹配所有的后代元素	元素集合
parent>child	根据父元素匹配所有的子元素	元素集合
prev+next	匹配所有紧接在 prev 元素后的相邻元素	元素集合
prev~siblings	匹配 prev 元素之后的所有兄弟元素	元素集合

选择器详述

- ancestor descendan选择器
- ancestor代表祖先，descendan代表子孙，用于在给定的祖先元素下匹配所有的后代元素。ancestor descendan选择器的使用方法：

```
$("ancestor descendant");
```

- 例如要匹配ul元素下的li元素，可以使用以下代码：
- $(“ul li”)

选择器详述

- 案例5:
- <script>
- $(function(){
- $("div ul").addClass("copyright");
- });
- </script>

选择器详述

- parent > child选择器
- parent代表父元素，child代表子元素，用于在给定的父元素下匹配所有的子元素。使用该选择器只能选择父元素的直接子元素。使用方法：

```
$("parent > child");
```

- 例如要匹配表单中的所有子元素input，可以使用以下代码：
- $(“form>input”)

选择器详述

- 案例6:
- <script>
- $(function(){
- $("#change").click(function(){
- $("form>input").addClass("input");
- });
- $("#default").click(function(){
- $("form>input").removeClass("input");
- });
- });
- </script>

选择器详述

- prev+next选择器
- prev+next选择器用于匹配所有紧接在prev元素后的next元素。其中，prev和next是两个相同级别的元素。使用方法：

```
$("prev + next");
```

- 例如要匹配 标记后的<img>标记，可以使用以下代码：
- $(“div+img”)

- 注意：选择的是紧接着的那个元素。

选择器详述

- 案例7：

```
<script>
    $(function(){
        $("label+p").addClass("background");
    });
 </script>
```

```
<body>
    <div>
  <label>第一个label标记</label>
  <p>第一个p标记</p>
  <fieldset>
    <label>第二个label标记</label>
    <p>第二个p标记</p>
  </fieldset>
</div>
<p>div外面的p标记</p>
  </body>
```

选择器详述

- prev ~ siblings选择器
- prev ~ siblings选择器用于匹配prev元素后的所有sibling元素。其中，prev和next是两个相同级别的元素。使用方法：

```
$("prev ~ siblings");
```

- 例如要匹配标记后的<ul>标记，可以使用以下代码：
- $(“span~ul”)

- 注意：选择的是所有紧接着的元素。

选择器详述

- 案例8：

```
<script>
            $(function(){
                $("div~p").addClass("background");
            });
        </script>
    </head>

    <body>
<div>
    <p>第一个p标记</p>
    <p>第二个p标记</p>
</div>
<p>div外面的p标记</p>
<p>div外面的p标记</p>
```

选择器详述

- 简单过滤选择器
- 过滤选择器根据某类过滤规则进行元素的匹配，书写时都以冒号（:）开头，而简单过滤器便是过滤器当中使用最为广泛的一种，其具体使用说明如下表15-03所示：

选择器详述

- 简单过滤选择器

过 滤 器	说　明	示　例
:first	匹配找到的第一个元素，它是与选择器结合使用的	$("tr:first")　//匹配表格的第一行
:last	匹配找到的最后一个元素，它是与选择器结合使用的	$("tr:last")　//匹配表格的最后一行
:even	匹配所有索引值为偶数的元素，索引值从0开始计数	$("tr:even")　//匹配索引值为偶数的行
:odd	匹配所有索引值为奇数的元素，索引从0开始计数	$("tr:odd")　//匹配索引值为奇数的行
:eq(index)	匹配一个给定索引值的元素	$("div:eq(1)") //匹配第二个div元素
:gt(index)	匹配所有大于给定索引值的元素	$("div:gt(0)") //匹配第二个及以上的div元素
:lt(index)	匹配所有小于给定索引值的元素	$("div:lt(2)") //匹配第二个及以下的div元素
:header	匹配如 h1, h2, h3……之类的标题元素	$(":header")　//匹配全部的标题元素
:not(selector)	去除所有与给定选择器匹配的元素	$("input:not(:checked)") //匹配没有被选中的input元素
:animated	匹配所有正在执行动画效果的元素	$(":animated ") //匹配所有正在执行的动画

选择器详述

- 案例9:
- <script>
- $(function(){
- $("tr:even").addClass("even");
- $("tr:odd").addClass("odd");
- $("tr:first").removeClass("even");
- $("tr:first").addClass("th");
- });
- </script>

选择器详述

- 内容过滤选择器
- 内容过滤选择器根据元素中的文字内容或所包含的子元素特征获取元素，其文字内容可以模糊或绝对匹配进行元素定位，其具体使用说明如下表15-04所示：

过 滤 器	说 明	示 例
contains(text)	匹配包含给定文本的元素	$("li:contains('DOM')") //匹配含有“DOM”文本内容的li元素
:empty	匹配所有不包含子元素或者文本的空元素	$("td:empty") //匹配不包含子元素或者文本的单元格
:has(selector)	匹配含有选择器所匹配元素的元素	$("td:has(p)") //匹配表格的单元格中含有<p>标记的单元格
:parent	匹配含有子元素或者文本的元素	$("td: parent") //匹配不为空的单元格，即在该单元格中还包括子元素或者文本

选择器详述

- 案例10:
- <script>
- $(function(){
- $("td:parent").css("background-color","#E8F3D1");
- $("td:empty").html("暂无内容"); //为空的单元格添加默认内容
- $("td:contains('JavaScript')").css("color","red"); //将含有文本wgh的单元格的文字颜色设置为红色
- });
- </script>

选择器详述

- 可见性过滤选择器
- 可见性过滤选择器根据元素是否可见的特征获取元素，其具体使用说明如下表15-05所示：

表 15-05 可见性过滤选择器语法

选择器	描述	返回值
:hidden	获取所有不可见元素，或者 type 为 hidden 的元素	元素集合
:visible	获取所有的可见元素	元素集合

选择器详述

过滤选择器

- 案例11:
- <script type="text/javascript">
- $(document).ready(function() {
- var visibleVal = $("input:visible").val(); //取得显示的input的值
- var hiddenVal1 = $("input:hidden:eq(0)").val(); //取得隐藏的input的值
- var hiddenVal2 = $("input:hidden:eq(1)").val(); //取得隐藏的input的值
- alert(visibleVal+"\n\r"+hiddenVal1+"\n\r"+hiddenVal2); //alert取得的信息
- });
- </script>

```
<body>
    <input type="text" value="显示的input元素">
    <input type="text" value="不显示的input元素" style="display:none">
    <input type="hidden" value="隐藏域">

</body>
```

选择器详述

- 表单对象属性过滤选择器
- 表单对象属性过滤选择器通过表单中的某个对象属性特征获取该元素，其具体使用说明如下表15-08所示：

过 滤 器	说　明	示　例
:checked	匹配所有选中的被选中元素	$("input:checked")　//匹配checked属性为checked的input元素
:disabled	匹配所有不可用元素	$("input:disabled")　//匹配disabled属性为disabled的input元素
:enabled	匹配所有可用的元素	$("input:enabled ")　//匹配enabled属性为enabled的input元素
:selected	匹配所有选中的option元素	$("select option:selected")　//匹配select元素中被选中的option

选择器详述

过滤选择器

- 案例12:
- <script>
- $(document).ready(function() {
- $("input:disabled").val("被禁用的按钮"); //为灰色不可用按钮赋值
- });
- function selectVal(){ //下拉列表框变化时执行的方法
- alert($("select option:selected").val()); //显示选中的值
- }
- </script>

不可用按钮： <input type="button" value="不可用按钮" disabled>
下拉列表框：
<select onchange="selectVal()">
<option value="列表项1">列表项1</option>
<option value="列表项2">列表项2</option>
<option value="列表项3">列表项3</option>
</select>

选择器详述

- 子元素过滤选择器
- 在页面开发过程中，常常遇到突出指定某行的需求。虽然使用基本过滤选择器:eq(index)可实现单个表格的显示，但并不能满足大量数据和多个表格的选择需求。为了实现这样的需求，jQuery中可以通过子元素过滤选择器十分轻松的获取所有父元素中的某个元素，其具体使用说明如下：

选 择 器	说　明	示　例
:first-child	匹配所有给定元素的第一个子元素	$("ul li:first-child") //匹配ul元素中的第一个子元素li
:last-child	匹配所有给定元素的最后一个子元素	$("ul li:last-child") //匹配ul元素中的最后一个子元素li
:only-child	匹配元素中唯一的子元素	$("ul li:only-child") //匹配只含有一个li元素的ul元素中的li
:nth-child(index/even/odd/equation)	匹配其父元素下的第index个子元素或奇偶元素，index从1开始，而不是从0开始	$("ul li:nth-child(even)") //匹配ul中索引值为偶数的li元素 $("ul li:nth-child(3)") //匹配ul中第3个li元素

选择器详述

- 案例13：

```
<script>
        $(function(){
            $("li:nth-child(2)").addClass("GetFocus");
            $("li:only-child").addClass("GetFocus");
        })
    </script>
```

```
<body>
    <ul>
        <li>Item1-0</li>
        <li>Item1-1</li>
        <li>Item1-2</li>
        <li>Item1-3</li>
    </ul>
    <ul>
        <li>Item2-0</li>
        <li>Item2-1</li>
        <li>Item2-2</li>
        <li>Item2-3</li>
    </ul>
    <ul>
        <li>Item3-0</li>
    </ul>
</body>
```

选择器详述

- 属性过滤选择器
- 属性过滤选择器根据元素的某个属性获取元素，如id属性值或匹配属性值的内容，并用中括号包裹，如下表所示：

选 择 器	说 明	示 例
[attribute]	匹配包含给定属性的元素	$("div[name]") //匹配含有name属性的div元素
[attribute=value]	匹配给定的属性是某个特定值的元素	$("div[name='test']") //匹配name属性是test的div元素
[attribute!=value]	匹配所有含有指定的属性，但属性不等于特定值的元素	$("div[name!='test']") //匹配name属性不是test的div元素
[attribute*=value]	匹配给定的属性是以包含某些值的元素	$("div[name*='test']") //匹配name属性中含有test值的div元素
[attribute^=value]	匹配给定的属性是以某些值开始的元素	$("div[name^='test']") //匹配name属性以test开头的div元素
[attribute$=value]	匹配给定的属性是以某些值结尾的元素	$("div[name$='test']") //匹配name属性以test结尾的div元素
[selector1][selector2][selectorN]	复合属性选择器，需要同时满足多个条件时使用	$("div[id][name^='test']") //匹配具有id属性并且name属性是以test开头的div元素

选择器详述

- 表单选择器
- 无论是提交还是传递数据，表单在页面中的作用是显而易见的。通过表单进行数据的提交或处理，在前端页面开发中占据重要地位。因此，为了使用户能够更加方便的、高效的使用表单，在jQuery选择器中引入了表单选择器，该选择器专为表单量身打造，通过它可以在页面中快速定位某表单对象，其具体使用说明如下表所示：

选择器详述

- 表单选择器

选 择 器	说 明	示 例
:input	匹配所有的input元素	$(":input") //匹配所有的input元素 $("form :input") //匹配<form>标记中的所有input元素，需要注意，在form和:之间有一个空格
:button	匹配所有的普通按钮，即type="button"的input元素	$(":button") //匹配所有的普通按钮
:checkbox	匹配所有的复选框	$(":checkbox") //匹配所有的复选框
:file	匹配所有的文件域	$(":file") //匹配所有的文件域
:hidden	匹配所有的不可见元素，或者type为hidden的元素	$(":hidden") //匹配所有的隐藏域
:image	匹配所有的图像域	$(":image") //匹配所有的图像域
:password	匹配所有的密码域	$(":password") //匹配所有的密码域
:radio	匹配所有的单选按钮	$(":radio") //匹配所有的单选按钮
:reset	匹配所有的重置按钮，即type=" reset "的input元素	$(":reset") //匹配所有的重置按钮
:submit	匹配所有的提交按钮，即type=" submit "的input元素	$(":submit ") //匹配所有的提交按钮
:text	匹配所有的单行文本框	$(":text ") //匹配所有的单行文本框

选择器详述

- 案例14：

```
<script type="text/javascript">
    $(document).ready(function() {
        $(":checkbox").attr("checked","checked");                    //选中复选框
        $(":radio").attr("checked","true");                                  //选中单选框
        $(":image").attr("src","Images/10-fish1.jpg");                 //设置图片路径
        $(":file").hide();                                                          //隐藏文件域
        $(":password").val("123");                                             //设置密码域的值
        $(":text").val("文本框");                                               //设置文本框的值
        $(":button").attr("disabled","disabled");                      //设置按钮不可用
        $(":submit").val("提交按钮");                                       //设置提交按钮的值
        $(":reset").val("重置按钮");                                        //设置重置按钮的值
        $("#testDiv").append($("input:hidden:eq(1)").val());     //显示隐藏域的值
    });
</script>
```