# 概述

jQuery是一个JavaScript代码仓库，或称之为框架。jQuery功能强大。能快速方便的完成以下任务：

选择页面元素：jQuery有强大的选择器。

动态更改页面样式：由于浏览器对css样式的兼容性不容，js处理一些特殊的样式时需要考虑兼容性。jQuery弥补这一不足，提供了一套跨浏览器的标准解决方案。

动态更改页面内容：jQuery能使用少量代码改变文档内容。

控制响应事件：jQuery提供了丰富的页面事件。

提供基本网页特效：淡入、淡出、移动等。

快速实现通信：jQuery封装了Ajax的操作。

扩展JavaScript内核：如迭代和数组操作。

jQuery有1.x、2.x、和3.x多个版本。

1.x：兼容ie678,使用最为广泛的，官方只做BUG维护，功能不再新增。因此一般项目来说，使用1.x版本就可以了。

2.x：不兼容ie678，很少有人使用，官方只做BUG维护，功能不再新增。如果不考虑兼容低版本的浏览器可以使用2.x。

3.x：不兼容ie678，只支持最新的浏览器。除非特殊要求，一般不会使用3.x版本的，很多老的jQuery插件不支持这个版本。目前该版本是官方主要更新维护的版本。

使用jQuery时，在页面头部引用jQuery类库：

<script type="text/javascript" src="jquery-1.11.3.min.js"></script>

# 选择器

## 基本选择器

### ID选择器

根据元素的id值选择元素。使用js时有一个document.getElementById()方法。jQuery简化了这个操作使用$(“#”)代表根据元素的id值选择元素。

语法：$(“#id”)

示例：

<input id="aa" value="1"/>  
<script type="text/javascript">  
 //dom对象  
 var ***dom*** = document.getElementById("aa");  
 alert("dom=" + ***dom***.value);  
 //jQuery对象  
 var ***jq*** = $("#aa");  
 alert("jq=" + ***jq***.val())  
</script>

注：理论上id的值在同一个页面中应该是唯一的。但如果id值冲突，jQuery会返回第一个对象。

### 标签选择器

根据html标签名可以选择元素。js提供了document.getElementsByName(“标签名”)方法获得元素。jQuery中使用$(“标签名”)。

<input id="aa" value="1"/>  
<script type="text/javascript">  
 //dom对象  
 var ***dom*** = document.getElementsByName("input");  
 //jQuery对象  
 var ***jq*** = $("input");  
</script>

### 类选择器

根据元素的class值选择元素。js中使用document.getElementsByClassName(“类名”)选择元素。

jQuery中使用$(“.类名”)。

<input id="aa" value="1" class="bb"/>  
<script type="text/javascript">  
 var ***dom*** = document.getElementsByClassName("bb");//dom对象  
 var ***jq*** = $(".bb");//jQuery对象  
</script>

$(“div.aa”)注意中间没空格，选择的是class=”aa”的div

### 通配选择器

通配选择器$(“\*”)选择所有元素。js中使用document.getElementsByName(“\*”)

### 组选择器

$(“selector1,selector2,selectorN”)表示多个选择器，选择器可以是任何类型，没有数量限制。

如$(“input,div,#aa”)表示选择所有的input标签和div还有id=aa的元素。

## 层级选择器

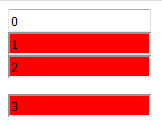
层级选择器在DOM嵌套结构中非常实用。

### 包含选择器

包含选择器是一个祖先下匹配的所有后代节点。祖先和后代选择器之间用空格隔开。选择器可以是任意类型。

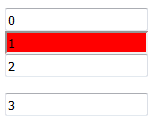
$(“ancestor descendant”) ancestor选择器可以是任意类型，空格之后是后代节点。注意空格很重要！！

示例：



<!--这个节点不在form里，不会选中-->  
<input value="0" class="bb"/>  
<form>  
 <input value="1"/>  
 <div><input value="2"/></div>  
 <div><p><input value="3"/></p></div>  
</form>  
<script type="text/javascript">  
 //选择form下的所有input，子节点、孙节点等所有后代  
 var ***input***=$("form input");  
 ***input***.css("background","red");//改变背景色  
</script>

### 子选择器



$(“parent>child”)只选择parent下子级的child，非子级的后代就不选中了。

<!--这个节点不在form里，不会选中-->  
<input value="0" class="bb"/>  
<form>  
 <!--这个是form的子元素，可以选中-->  
 <input value="1"/>  
 <!--这两个input不是子级，是孙子和重孙子，不会选中-->  
 <div><input value="2"/></div>  
 <div><p><input value="3"/></p></div>  
</form>  
<script type="text/javascript">  
 //选择form下的所有子级的input元素  
 var ***input*** = $("form > input");  
 ***input***.css("background", "red");//改变背景色  
</script>

### 相邻选择器

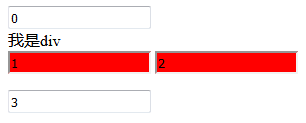
$(“prev+next”)找到所有prev元素后的next元素。



<div>我是div</div>  
<input value="1"/><!--这个元素紧随div会选中-->  
<input value="2"/><!--这个元素没有紧随div，不会选中-->  
<script type="text/javascript">  
 //选择form下的所有子级的input元素  
 var ***input*** = $("div + input");  
 ***input***.css("background", "red");//改变背景色  
</script>

### 兄弟选择器

$(“prev~siblings”)选择所有在prev之后且与prev同级的siblings。



<!--虽然与div同级，但不在div后面-->  
<input value="0"/>  
<div>我是div</div>  
<input value="1"/><!--在div之后且同级，会选中-->  
<input value="2"/><!--在div之后且同级，会选中-->  
<p><input value="3"/><!--在div之后但不同级，不会选中--></p>  
<script type="text/javascript">  
 //选择form下的所有子级的input元素  
 var ***input*** = $("div ~ input");  
 ***input***.css("background", "red");//改变背景色  
</script>

## 属性选择器

|  |  |
| --- | --- |
| 选择器 | 说明 |
| [attribute] | 匹配包含特定属性的元素，如$(“div[id]”)选择所有含有id属性的div。 |
| [attribute=value] | 匹配属性值等于特定值的元素。如$("a[href='#']")匹配所有 href 属性的值等于 "#" 的a标签 |
| [attribute!=value] | $("a[href!='#']")匹配所有 href 属性的值不等于 "#" 的a标签 |
| [attribute^=value] | $(" input [name^=test]")所有name以test开头的input标签 |
| [attribute$=value] | $("a[href$='.jpg']") 所有 href 属性的值包含以 ".jpg" 结尾的a标签 |
| [attribute\*=value] | $(" input [name\*='test]")所有name值包含test的input标签 |
| [selector1] [selector2] [selectorN] | $(" input [name\*='test][id]") 所有name值包含test的且含有id属性的input标签 |

## 伪类选择器

伪类可以看做是特殊的类型选择，它的特征是选择器中的冒号”:”

|  |  |  |
| --- | --- | --- |
| 特定位置的伪类选择器 | | |
| :first | $("p:first") | 第一个 <p> 元素 |
| :last | $("p:last") | 最后一个 <p> 元素 |
| :eq(index) | $("ul li:eq(3)") | 列表中的第四个元素（index 从 0 开始） |
| 指定范围的伪类选择器 | | |
| :even | $("tr:even") | 所有偶数 <tr> 元素 |
| :odd | $("tr:odd") | 所有奇数 <tr> 元素 |
| :gt(no) | $("ul li:gt(3)") | 列出ul下 索引 大于 3 的li元素 |
| :lt(no) | $("ul li:lt(3)") | 列出 ul下 索引小于 3 的li元素 |
| 排除选择器 | | |
| :not(*selector*) | $("input:not(:empty)") | 所有不为空的 input 元素 |
| 特殊选择器 | | |
| :header | $(":header") | 所有标题元素 <h1> - <h6> |
| :animated |  | 所有动画元素 |
| 与内容相关的伪类选择器 | | |
| :contains(text) | $("div:contains('W3')") | 内容包含W3的div |
| :empty | $("div:empty") | 无子（元素）节点的所有div元素 |
| :has(selector) | $("div:has(p)") | 包含p标签的div |
| :parent | $("div:parent") | 匹配非空的div |
| 与显示相关的伪类选择器 | | |
| :hidden | $("p:hidden") | 所有隐藏的 <p> 元素 |
| :visible | $("a:visible") | 所有可见的a标签 |
| 匹配子元素的伪类选择器 | | |
| :first-child | $("div span:first-child") | 给每个匹配的 div 中查找第一个 span |
| :first-of-type |  | 选择所有相同的元素名称的第一个兄弟元素。 |
| :last-child |  | 选择所有父级元素下的最后一个子元素 |
| :last-of-type |  | 选择的所有元素之间具有相同元素名称的最后一个兄弟元素。 |
| :nth-child() |  | 选择的他们所有父元素的第n个子元素，n从1开始 |
| :nth-last-child() |  | 选择所有他们父元素的第n个子元素。计数从最后一个元素开始到第一个。 |
| :nth-last-of-type() |  | 选择的所有他们的父级元素的第n个子元素，计数从最后一个元素到第一个。 |
| :nth-of-type() |  | 选择同属于一个父元素之下，并且标签名相同的子元素中的第n个 |
| :only-child |  | 如果某个元素是其父元素的唯一子元素，那么它就会被选中。 |
| :only-of-type |  | 选择所有没有兄弟元素，且具有相同的元素名称的元素。 |
| 表单相关的伪类选择器 | | |
| :input | $(":input") | 所有 <input> 元素 |
| :text | $(":text") | 所有 type="text" 的 <input> 元素 |
| :password | $(":password") | 所有 type="password" 的 <input> 元素 |
| :radio | $(":radio") | 所有 type="radio" 的 <input> 元素 |
| :checkbox | $(":checkbox") | 所有 type="checkbox" 的 <input> 元素 |
| :submit | $(":submit") | 所有 type="submit" 的 <input> 元素 |
| :reset | $(":reset") | 所有 type="reset" 的 <input> 元素 |
| :button | $(":button") | 所有 type="button" 的 <input> 元素 |
| :image | $(":image") | 所有 type="image" 的 <input> 元素 |
| :file | $(":file") | 所有 type="file" 的 <input> 元素 |
| 表单属性相关的伪类 | | |
| :enabled | $(":enabled") | 所有激活的 input 元素 |
| :disabled | $(":disabled") | 所有禁用的 input 元素 |
| :selected | $(":selected") | 所有被选取的 input 元素 |
| :checked | $(":checked") | 所有被选中的 input 元素 |

## 选择器优化

1. 多用ID 选择器，如果不存在ID选择器，可以在父元素中添加ID，这样可以缩短访问路径
2. 少直接使用class选择器，可以使用标签.class。避免表达式中的冗余，如div#aa，可以直接简化成#aa。
3. 多用父子关系parent>child,少用parent child因为>是子选择器，不递归，而“ ”是后代选择器，后递归遍历子元素及子元素的子元素（所有后代）…
4. 缓存jQuery对象，如果选出的结果不发生变化，可以缓存jQuery对象，如：

for(var ***i***=0;***i***<100;***i***++){  
 var ***aa***=$("#aa");  
 ***aa***.append(***i***);  
}  
//优化后  
var ***aa***=$("#aa");  
for(var ***i***=0;***i***<100;***i***++){  
 ***aa***.append(***i***);  
}

# jQuery和DOM

jQuery对象对DOM对象进行了封装，本质上jQuery对象是一个普通的JavaScript对象，它包含DOM对象的集合。即DOM对象是一个独立的个体，而jQuery对象是多个DOM对象的集合。

<input id="aa"/>  
<script type="text/javascript">  
 var ***dom*** = document.getElementById("aa");  
 alert(***dom***);//[object HTMLInputElement]  
 var ***jq***=$("#aa");  
 alert(***jq***);//[object Object]  
</script>

<input id="aa"/>  
<script type="text/javascript">  
 var ***dom*** = document.getElementById("aa");  
 alert(***dom***.length);//未定义  
 var ***jq***=$("#aa");  
 alert(***jq***.length);//长度是1  
</script>

如果选择一个不存在的对象，dom会是null，而jQuery对象是一个长度等于0的集合，所以它不会有空指针的异常。

## DOM对象转jQuery对象

$(dom)用$()把dom对象括起来，就会返回jQuery对象。

## jQuery对象转DOM对象

使用[index]或get(index)可以将jQuery对象转换成DOM对象。如$(“li”)[0]或$(“li”).get(0)

## window.onload和document.ready

浏览器解析大致有以下几个步骤：

1. 解析HTML结构
2. 加载外部脚本和样式
3. 解析并执行脚本代码
4. 构造HTML DOM 模型
5. 加载图片等外部文件
6. 界面加载完成

window.onload会在页面所有内容加载完成之后执行(第6步之后)，比如图片等对媒体文件。如果媒体文件较多，即时网页文档已经显示，但load事件不会触发。

document.ready会在DOM绘制完毕后执行(第4步之后)，能确保文档呈现和脚本初始化同时完成。

window.onload只能写一次，而document.ready可以写多次。

<script type="text/javascript">  
 //只会显示load2  
 window.onload=function () {  
 alert("load1")  
 }  
 window.onload=function () {  
 alert("load2")  
 }  
</script>

<script type="text/javascript">  
 //都会执行  
 $(document).ready(function () {  
 alert("ready1")  
 })  
 $(document).ready(function () {  
 alert("ready2")  
 })  
 //简化形式  
 $(function () {  
 alert("ready3")  
 })  
</script>

# 过滤器

jQuery提供了选择器和过滤器两种方式选择文档元素，选择器主要模仿CSS语法，提供高效准确匹配元素的一般方法，而过滤器是建立在选择器基础上进行二次筛选的。

## 过滤

|  |  |  |
| --- | --- | --- |
| 过滤方法 | 示例 | 说明 |
| eq(index) | $(“ul>li”).eq(3) | 找到第n个元素,n从0开始 |
| hasClass(class) | $(“#aa”).hasClass(“bb”) | 判断id=aa的元素是否有class=”bb”的属性，如果有就返回true |
| filter(expr) | $(“div”).filter(“.red,.blue”)  $(“div”) .filter(function(index){  return $(“span”,this).length==2  }) | 选择class=”red”或class=”blue”的div  找到包含两个span标签的div |
| is(expr) | $(“#aa”).is(“input”) | 判断id=aa的元素是不是一个input标签 |
| map(callback) |  | map能把一组元素转化成其它数组 |
| has(expr) | $(“div”).has(“span.red”) | 找到包含<span class=”red”>的div |
| not(expr) |  | 和is相反 |
| slice(start,[end]) |  | 截取指定范围内的元素 |

<input value="a"/>  
<input value="b"/>  
<input value="c"/>  
<script type="text/javascript">  
 $(function () {  
 var vals = $("input").map(function () {  
 return $(this).val()  
 });  
 for (var i = 0; i < vals.length; i++) {  
 alert(vals[i])  
 }  
 })  
</script>

<input value="a"/>  
<input value="b"/>  
<input value="c"/>  
<input value="d"/>  
<script type="text/javascript">  
 $(function () {  
 $("input").slice(1,4).val(0)  
 })  
</script>

## 查找

|  |  |
| --- | --- |
| **函数** | **描述** |
| .add() | 将元素添加到匹配元素的集合中。 |
| .andSelf() | 把堆栈中之前的元素集添加到当前集合中。 |
| .children() | 获得匹配元素集合中每个元素的所有子元素。 |
| .closest() | 从元素本身开始，逐级向上级元素匹配，并返回最先匹配的祖先元素。 |
| .contents() | 获得匹配元素集合中每个元素的子元素，包括文本和注释节点。 |
| .each() | 对 jQuery 对象进行迭代，为每个匹配元素执行函数。 |
| .end() | 结束当前链中最近的一次筛选操作，并将匹配元素集合返回到前一次的状态。 |
| .eq() | 将匹配元素集合缩减为位于指定索引的新元素。 |
| .filter() | 将匹配元素集合缩减为匹配选择器或匹配函数返回值的新元素。 |
| .find() | 获得当前匹配元素集合中每个元素的后代，由选择器进行筛选。 |
| .first() | 将匹配元素集合缩减为集合中的第一个元素。 |
| .has() | 将匹配元素集合缩减为包含特定元素的后代的集合。 |
| .is() | 根据选择器检查当前匹配元素集合，如果存在至少一个匹配元素，则返回 true。 |
| .last() | 将匹配元素集合缩减为集合中的最后一个元素。 |
| .map() | 把当前匹配集合中的每个元素传递给函数，产生包含返回值的新 jQuery 对象。 |
| .next() | 获得匹配元素集合中每个元素紧邻的同辈元素。 |
| .nextAll() | 获得匹配元素集合中每个元素之后的所有同辈元素，由选择器进行筛选（可选）。 |
| .nextUntil() | 获得每个元素之后所有的同辈元素，直到遇到匹配选择器的元素为止。 |
| .not() | 从匹配元素集合中删除元素。 |
| .offsetParent() | 获得用于定位的第一个父元素。 |
| .parent() | 获得当前匹配元素集合中每个元素的父元素，由选择器筛选（可选）。 |
| .parents() | 获得当前匹配元素集合中每个元素的祖先元素，由选择器筛选（可选）。 |
| .parentsUntil() | 获得当前匹配元素集合中每个元素的祖先元素，直到遇到匹配选择器的元素为止。 |
| .prev() | 获得匹配元素集合中每个元素紧邻的前一个同辈元素，由选择器筛选（可选）。 |
| .prevAll() | 获得匹配元素集合中每个元素之前的所有同辈元素，由选择器进行筛选（可选）。 |
| .prevUntil() | 获得每个元素之前所有的同辈元素，直到遇到匹配选择器的元素为止。 |
| .siblings() | 获得匹配元素集合中所有元素的同辈元素，由选择器筛选（可选）。 |
| .slice() | 将匹配元素集合缩减为指定范围的子集。 |

# DOM操作

## 创建节点

<script type="text/javascript">  
 //js的写法  
 var ***div***=document.createElement("div");  
 var ***txt***=document.createTextNode("DOM");  
 ***div***.appendChild(***txt***);  
 ***div***.setAttribute("title","盒子");  
 document.body.appendChild(***div***);  
 //jquery写法1  
 //var div2=$('<div title="盒子">JQUERY</div>');  
 //jquery写法2  
 var ***div2***=$("<div></div>");  
 ***div2***.text("JQUERY");  
 ***div2***.attr("title","盒子")  
 $("body").append(***div2***)  
</script>

生成的代码：

<div title="盒子">DOM</div>  
<div title="盒子">JQUERY</div>

## 插入节点

|  |  |
| --- | --- |
| 方法 | 说明 |
| append() | a.append(b)在a内部加入b，b是a的last child |
| appendTo() | a.appendTo(b)等同于b.append(a) |
| prepend() | a.prepend(b)在a内部加入b，b是a的first child |
| prependTo() | a.prependTo (b)等同于b.prepend(a) |
| after() | a.after(b)在a外部加入b，b是紧随a的兄弟节点 |
| insertAfter() | a.insertAfter(b)等同于b.after(a) |
| before() | a.before(b)在a外部加入b，b是在a之前紧邻的兄弟节点 |
| insertBefore() | a.insertBefore(b)等同于b.before(a) |

位置示意图：

<!--before-->  
<div>  
 <!--prepend-->  
 内容  
 <!--append-->  
</div>  
<!--after-->

## 删除元素

|  |  |
| --- | --- |
| 方法 | 说明 |
| remove([selector]) | 移除元素 |
| empty() | 清空元素内部 |
| detach([selector]) | 与remove方法类似，元素移除，但元素绑定的事件会被保留。 |

<input type="button" onclick="*removeDiv*()" value="移除">  
<input type="button" onclick="*addDiv*()" value="添加">  
<div style="width: 100px;height: 100px;border: 1px solid #ccc">  
 内容  
</div>  
  
<script type="text/javascript">  
 $("div").click(function () {  
 alert(1)  
 })  
 var ***aa***;//缓存被移除的元素  
  
 function *removeDiv*() {  
 ***aa*** = $("div").remove();  
 }  
  
 function *addDiv*() {  
 $("body").append(***aa***);  
 }  
</script>

## 克隆节点

clone([withDataAndEvents],[deepWithDataAndEvents])

withDataAndEvents表示是否处理事件

deepWithDataAndEvents设置是否复制处理事件和克隆元素的所有子元素进行数据复制。

<input id="aa" onclick="alert(1)">  
<input type="button" value="克隆" onclick="*cloneTest*()"/>  
<script type="text/javascript">  
 $(function () {  
 $("#aa").change(function () {  
 alert(2)  
 })  
 })  
  
 function *cloneTest*() {  
 //clone中设置true，change事件会被复制  
 $("body").append($("#aa").clone(true))  
 }  
</script>

## 替换内容

a.replaceWith(b) 用b替换a。等同于b.replaceAll(a)

<div id="aa">aa</div>  
<input type="button" value="替换" onclick="$('#aa').replaceWith('<div>bb</div>')"/>

## 包裹内容

包裹有三个方法：wrap(),wrapInner(),wrapAll()

目标元素：

<div class="aa">aa</div>  
<div class="aa">aa</div>

<!--$(".aa").wrap("<a></a>")-->  
<a><div class="aa">aa</div></a>  
<a><div class="aa">aa</div></a>  
<!--$(".aa").wrapInner("<a></a>")-->  
<div class="aa"><a>aa</a></div>  
<div class="aa"><a>aa</a></div>

<!--$(".aa").wrapAll("<a></a>")-->  
<a>  
 <div class="aa">aa</div>  
 <div class="aa">aa</div>  
</a>

卸包unwrap()删除匹配元素的父级元素，保留自身。

## 操作属性

prop()和attr()都用来操作属性。传递一个参数时是获取属性值，传递两个参数时是给属性赋值。

只设置属性名，不需要设置属性值的时候或属性值只有true和false的时候，使用prop()方法。

removeProp()和removeAttr()用于删除属性。

## 添加类属性

addClass()添加class。removeClass()删除class。toggleClass()自动切换class

## 读写内容

html()、text()、val()

## 样式操作

css()设置或获取样式。

offset()获取绝对定位，即相对于浏览器左上角的定位。返回包括top和left两个属性的对象。

position()获得相对定位，即相对于距离最近的父定位元素的左上角便宜。

width(),height()设置大小。

innerHeight(),innerWidth(),outerHeight(),outetWidth()

# 事件处理

bind 绑定一个事件

one 绑定一个事件，但只生效一次

unbind 解除绑定事件

<input type="button" value="one" id="aa"/>  
<input type="button" value="bind" id="bb"/>  
<input type="button" value="unbind" id="cc"/>  
<script type="text/javascript">  
 $(function () {  
 $("#aa").one("click", function () {  
 alert("one");  
 });  
 $("#bb").bind("click", function () {  
 alert("bind");  
 });  
 $("#cc").bind("click",function () {  
 $("#bb").unbind("click")  
 });  
 })  
</script>

常用事件：

blur,dblclick,keydown,select,change,error,keypress,submit,click,focus,keyup

为未来对象绑定事件：

<div style="width: 100%;border: 1px solid #ccc"></div>  
<input type="button" value="one" id="aa"/>  
<script type="text/javascript">  
 $(function () {  
 $("#aa").bind("click", function () {  
 $("div").append('<input type="button" class="bb"/>');  
 });  
 $("div").on("click",'.bb', function () {  
 alert("new");  
 });  
 })  
</script>

选择器选中的是父元素。

# 动画设计

显示：show()

隐藏：hide()

显隐切换：toggle()

滑动显示：slideDown()

滑动隐藏：slideUp()

显隐滑动切换：slideToggle()

淡入：fadeIn()

淡出：fadeOut()

设置淡出透明效果：fadeTo(time,opacity)

自动切换淡入淡出：fadeToggle(time)

自定义动画：animate

停止动画：stop([clearQueue],[jumpToEnd])

# 工具函数

trim()去除字符串前后的空格

param()把对象转换成url参数

$.each(collection,function(index,valueOfElement){})遍历一个集合