# Write Less, Do More！

## jQuery介绍

### 解决的问题

* 消除浏览器差异：你不需要自己写冗长的代码来针对不同的浏览器来绑定事件，编写AJAX等代码；
* 简洁的操作DOM的方法：写$('#test')肯定比document.getElementById('test')来得简洁；
* 轻松实现动画、修改CSS等各种操作。

## 选择器

### 什么是jQuery对象？

jQuery对象类似数组，它的每个元素都是一个引用了DOM节点的对象。

### 按ID查找

var a = $('.red'); // 所有节点包含`class="red"`都将返回

### 按tag查找

var ps = $('p');

### 按class查找

var a = $('.red');

var a = $('.red.green'); // 注意没有空格！

### 按属性查找

var a = $('[items="A B"]'); // 找出<??? items="A B"> //当属性的值包含空格等特殊字符时，需要用双引号括起来

* 按属性查找还可以使用前缀查找或者后缀查找：

var icons = $('[name^=icon]'); // 找出所有name属性值以icon开头的DOM

* 这个方法尤其适合通过class属性查找，且不受class包含多个名称的影响：

var icons = $('[class^="icon-"]'); // 找出所有class包含至少一个以`icon-`开头的DOM

// 例如: class="icon-clock", class="abc icon-home"

### 组合查找

var emailInput = $('input[name=email]'); // 不会找出<div name="email">

var tr = $('tr.red'); // 找出<tr class="red ...">...</tr>

### 多项选择器

多项选择器就是把多个选择器用,组合起来一块选.选出来的元素是按照它们在HTML中出现的顺序排列的，而且不会有重复元素。

### 层级选择器

$('ul.lang li');

#### 子选择器（Child Selector）

子选择器$('parent>child')类似层级选择器，但是限定了层级关系必须是父子关系

#### 过滤器（Filter）

过滤器一般不单独使用，它通常附加在选择器上，帮助我们更精确地定位元素。

$('ul.lang li'); // 选出JavaScript、Python和Lua 3个节点

$('ul.lang li:first-child'); // 仅选出JavaScript

$('ul.lang li:last-child'); // 仅选出Lua

$('ul.lang li:nth-child(2)'); // 选出第N个元素，N从1开始

$('ul.lang li:nth-child(even)'); // 选出序号为偶数的元素

$('ul.lang li:nth-child(odd)'); // 选出序号为奇数的元素

### 表单相关

针对表单元素，jQuery还有一组特殊的选择器：

:input：可以选择<input>，<textarea>，<select>和<button>；

:file：可以选择<input type="file">，和input[type=file]一样；

:checkbox：可以选择复选框，和input[type=checkbox]一样；

:radio：可以选择单选框，和input[type=radio]一样；

:focus：可以选择当前输入焦点的元素，例如把光标放到一个<input>上，用$('input:focus')就可以选出；

:checked：选择当前勾上的单选框和复选框，用这个选择器可以立刻获得用户选择的项目，如$('input[type=radio]:checked')；

:enabled：可以选择可以正常输入的<input>、<select> 等，也就是没有灰掉的输入；

:disabled：和:enabled正好相反，选择那些不能输入的。

### 其他

$('div:visible'); // 所有可见的div

$('div:hidden'); // 所有隐藏的div

### 查找和过滤

#### 查找

* 用find()查找子节点：

var ul = $('ul.lang'); // 获得<ul>

var dy = ul.find('.dy'); // 获得JavaScript

* 从当前节点开始向上查找，使用parent()方法
* 对于位于同一层级的节点，可以通过next()和prev()方法

#### 过滤

* filter()方法可以过滤掉不符合选择器条件的节点：
* map()方法把一个jQuery对象包含的若干DOM节点转化为其他对象：

#### 操作DOM

1. 获取文本：text();获取节点内的HTML代码html();

eg:<ul id=”test-ul”>

<li class=”js”>JavaScript</li>

<li name=”book”>Java &amp; JavaScript</li>

</ul>

获取name属性为book的节点的文本

$(‘#test-ul li[name=book]’).text();//Java & JavaScript

获取name属性为book的节点内的HTML

$(‘#test-ul li[name=book]’).html();//Java &amp; JavaScript

1. 设置文本：text(‘想要设置文本’);设置节点内的HTML代码html(‘想要设置的HTML代码’);

eg:$(‘#test-ul li.js’).text(‘c#’);

$(‘#test-ul li[name=js]’).html(‘<i>c#</i>’);

1. 一个jQuery对象可以包含0个或任意个DOM对象，它的方法实际上会作用在对应的每个DOM节点上.

eg:$(‘#test-ul li’).text(‘JS’);

1. 即使选择器没有返回任何DOM节点，调用jQuery对象的方法仍然不会报错。
2. jQuery对象的所有方法都返回一个jQuery对象（可能是新的也可能是自身），这样我们可以进行链式调用，非常方便。

eg:$('#test-css>li.dy>span').css('background-color', 'blue').css('color',"yellow");

1. css方法：

var div=$(‘#test-div’);

div.css(‘color’);//#fff,获取color属性的值

div.css(‘color’,’#e598fa’);//设置color属性的值

div.css(‘color’,’’);//清除color属性

1. 为了和JavaScript保持一致，CSS属性可以用’background-color’和’backgroundgroundColor’两种格式。
2. css()方法将作用于DOM节点的style属性，具有最高优先级。
3. 修改class属性的方法：

var div=$(‘#test-div’);

div.hasClass(‘highlight’);//false,判断class是否包含highlight

div.addClass(‘highlight’);//添加hightlight这个class

div.removeClass(‘highlight’);//删除highlight这个class

1. 显示和隐藏DOM show()和hide()

var a=$(‘a[target=\_blank]’);

a.hide();//隐藏

a.show();//显示

1. 获取DOM的高宽

//浏览器可视窗口大小

$(window).width();

$(window).height();

//HTML文档大小

$(document).width();

$(document).height();

//设置宽高

var div=$(‘#test-div’);

$div.widhth(400);

$div.height(‘200px’);

attr()和removeAttr()方法用于操作DOM节点的属性：

//<div id=”test-div” name =”Test” start = “1”></div>

var div =$(‘#test-div’);

div.attr(‘data’);//获取属性的值，data属性不存在

div.attr(‘name’,’hello’);//将attr的name属性设置为hello

div.removeAttr(‘name’);//删除name属性

div.attr(‘name’);//name属性不存在

prop()方法和attr()类似，但是HTML5规定有一种属性在DOM节点中可以没有值，只有出现何不出现两种情况。

eg:<input id=”test-radio” radio = “radio” name=”test” checked value = “1”>

等价于：

<input id = “test-radio” radio=”radio” name = “test” checked=”checked” value = “1”>

attr()和prop()对于属性checked处理有所不同

var radio = $(‘#test-radio’);

radio.attr(‘checked’);//’checked’

radio.prop(‘checked’);//true

prop()返回值更合理一些，不过，用is()方法判断更好。

var radio=$(‘#test-radio’);

radio.is(‘:checked’);//true

类似的属性还有selected,处理时最好用is(‘:selected’)

1. 操作表单

val()方法获取和设置对应的value属性

var input = $(‘#test-input’);

input.val();/’test’

input.val(‘xuanmiao@xuanmiao.com’);

##### 添加DOM

1. 要添加新的DOM节点，除了通过jQuery的html()这种暴力方法外，还可以用append()方法，

eg：

<div id="test-div">

<ul>

<li><span>JavaScript</span></li>

<li><span>Python</span></li>

<li><span>Swift</span></li>

</ul>

</div>

var ul =$(#test-div ul);

然后调用append()传入HTML片段

ul.append(‘<li><span>Java</span></li>’);

1. append()还可以传入原始的DOM对象，jQuery对象和函数对象

//创建DOM对象：

var ps =document.createElement(‘li’);

ps.innerHTML=’<span>Pascal</span>’;

//添加DOM对象：

ul.append(ps);

//添加jQuery

ul.append($(‘#scheme’));

//添加函数对象；

ul.append(function(index,html){

return ‘<li><span>Language-‘+index+’</span></li>’;

});

1. 传入函数时，要求返回一个字符串、DOM对象或者jQuery对象。因为jQuery的append()可能作用于一组DOM节点，只有传入函数才能针对每个DOM生成不同的子节点
2. append把DOM节点添加到最后，prepend()把DOM节点添加在最前面。
3. 如果要添加的DOM节点已经存在于HTML文档中，它会首先从文档移除，然后再添加，也就是说，用append()，你可以移动一个DOM节点。
4. 如果要把新节点插入到指定位置，例如，JavaScript和Python之间，那么，可以先定位到JavaScript，然后用after()方法：
5. var js=$(‘#test-div>ul>li:first-child’);
6. js.after(‘<li><span>Lua</span></li>’);

也就是说，同级节点可以用after()或者before()方法。

1. 删除节点

要删除DOM节点，拿到jQuery对象后直接调用remove()方法就可以了。

### 事件

鼠标事件：

click:单击事件

dbclick:双击事件

mouseenter:移入事件

mouseleave:移出事件

mousemove:在DOM内部移动事件

hover:鼠标进入和推出时触发两个函数，相当于mouseenter和mouseleave;

键盘事件：

键盘事件仅作用于当前焦点上的DOM，通常是input或者textarea。

keydown:键盘按下时触发

keyup:键盘抬起时触发

keypress:按一次键后触发

其他事件：

focus:当DOM获得焦点时触发。

blur:当DOM 失去焦点时触发。

change:当input select 或textarea 的内容改变时触发

submit:

ready:

eg:

$(document).on(‘ready’,function(){

$(‘#testForm’).on(‘submit’,’function(){

alert(‘submit’);

}’);

});

简化：

$(document).ready(function(){

$(‘#testForm’).submit(function(){

alert(‘submit’);

});

});

再简化为：

$(function(){

//init…

});

上面的这种写法最为常见。如果你遇到$(function(){

//init…

});牢记这是document对象的ready事件处理函数。

完全可以反复绑定处理事件函数，他们会依次执行：

$(function(){

console.log(‘init A…’);

});

$(function(){

console.log(‘init B…’);

});

$(function(){

console.log(‘init B…’);

});

事件参数

所有事件都会传入Event对象作为参数，可以从Event对象上获取到更多的信息。

eg:

$(function(){

$(‘#testMouseMoveDiv’).mousemove(function(e) {

$(‘#testMouseMoveSpan’).text(‘pageX=’+e.pageX+’,pageY=’+e.pageY);

});

});

获取当前页面的横纵坐标。

取消绑定

eg:

function hello(){

alert(‘hello’);

}

1. click(hello);

setTimeout(function(){

a.off(‘click’，hello);

},1000);

//1秒后取消绑定

需要特别注意的是:

eg:

1. click(function(){

alert(‘hello’);

});

1. off(‘click’,function(){  
   alert(‘hello’);

});

这种写法是无效的。

虽然这两个匿名函数一模一样，但是他们是两个不同的函数对象。

为了实现移出效果可以使用off(‘click’);来一次性移除所有click事件。

同理，无参数的off()将一次性移除所有类型的事件处理函数。

事件触发条件

一个需要注意的问题是，事件的触发总是由用户的操作引发的。

eg:

我们监控文本框的内容改动当文本在文本框中输入时，就会触发change事件。但是，如果用JavaScript代码去改动文本框的值，将不会触发change事件。

### 动画

show / hide/toggle（数字/’(fast/slow(=0.6))’）;从左上角展开或者收缩

slideUp / slideDown/slideToggle; 从顶部展开或者收缩

fadeIn / fadeOut/fadeToggle;淡入和淡出

自定义动画：animate()

eg:

var div = $('#test-animate');

div.animate({

opacity: 0.25,

width: '256px',

height: '256px'

}, 3000, function () {

console.log('动画已结束');

// 恢复至初始状态:

$(this).css('opacity', '1.0').css('width', '128px').css('height', '128px');

});

### AJAX

1. ajax(url,settings)函数需要接收一个URL和一个可选的settings对象，常用的选项如下：

async:是否异步执行AJAX请求，不要指定为false.

method:发送的method,缺省为‘GET’，可指定为‘POST’，‘PUT’等

contentType:发送POST请求的格式，默认值为：‘application/x-www-form-urlencoded;charset=UTF-8’,也可以指定为：text/plain、application/json

data:发送的数据，可以是字符串、数组或object。如果是GET请求，data将被转换为query附加到URL上，如果是POST请求，根据contentType把data序列化成合适的格式；

headers：发送的额外的HTTP头，必须是一个object。

dataType:接收的数据格式，可以指定为‘HTML’、‘xml’、‘json’、’text’等，缺省情况下根据响应的Content-Type猜测。

eg:

下面的例子发送一个GET请求，并返回一个JSON格式的数据；

var jqxhr = $.ajax(‘/api/categories’,{dataType:’json’});

1. 链式写法来处理各种回调
2. $.get,$post,$getJSON
3. 如果需要使用JSONP，可以在ajax()中设置jsonp: 'callback'，让jQuery实现JSONP跨域加载数据。

### 扩展

1. 编写jQuery插件

方法一：

给jQuery对象绑定一个新方法是通过扩展$.fn对象实现的。

$.fn.highlight1=function(){

//this已经绑定当前对象

this.css(‘backgroundColor’,’#fffceb’).css(‘color’,’#d85030’);

return this;//使该方法能够链式操作。

}

注意到函数内部的this在调用时被绑定为jQuery对象，所以函数内部代码可以正常调用所有jQuery对象的方法。

改进：

加参数，可以指定颜色

$.fn.highlight1=function(options){

//this已经绑定当前对象

var bgColor = options&&options.backgroundColor||’#fffceb’;

var color =option&&options.color||’#d85030’;

this.css(‘backgroundColor’,bgColor).css(‘color’, color);

return this;//使该方法能够链式操作。

}

$(‘#test-highlight2 span’).highlight2({

backgroundColor:’#00a8e6’,

color:’#ffffff’  
});

方法二：

$.fn.highlight =function(options){

var opts = $.extends({},$.fn.highlight.defaults,options);

this.css(‘backgroundColor’,opts.backgroundColor).css(‘color’,opts.color);

return this;

}

$.fn.highlight.defaults = {

color:”#d85030”,

backgroundColor:”#fff8d3”

}

针对特定元素的扩展：

我们知道jQuery对象的有些方法只能针对特定DOM元素，比如submitt()方法只能针对form。如果我们编写的扩展只能针对某些类型的元素该怎么写？

利用filter（）方法来过滤。

eg:给所有超链接标签加上跳转提示

$(‘#mian a’).external扩展

$.fn.prompt=function(){

return this.filter(a).each(function(){

var a = $(this);

var url = a.atrr(‘href’);

if(url&&url.indexOf(‘http://’)===0||url.indexOf(‘https://’)===0){

1. attr(‘href’,’#0’).removeAttr(‘target’).append(‘<I class=”uk-icon-external-link”></i>’).click(function(){if(confirm(‘你确定要前往’+url+’?’)){

window.open(url);

}

});

}

});

}

### 错误处理

try{

…

}catch(e){

…

}finally{

…

}

### 错误传播