**Jquery核心**

## **jQuery([selector,[context]])**

### 概述

这个函数接收一个包含 CSS 选择器的字符串，然后用这个字符串去匹配一组元素。

jQuery 的核心功能都是通过这个函数实现的。 jQuery中的一切都基于这个函数，或者说都是在以某种方式使用这个函数。这个函数最基本的用法就是向它传递一个表达式（通常由 CSS 选择器组成），然后根据这个表达式来查找所有匹配的元素。

默认情况下, 如果没有指定context参数，$()将在当前的 HTML document中查找 DOM 元素；如果指定了 context 参数，如一个 DOM 元素集或 jQuery 对象，那就会在这个 context 中查找。在jQuery 1.3.2以后，其返回的元素顺序等同于在context中出现的先后顺序。

### 示例

#### 描述:

找到所有 p 元素，并且这些元素都必须是 div 元素的子元素。

##### jQuery 代码:

$("div > p");

#### 描述:

设置页面背景色。

##### jQuery 代码:

$(document.body).css( "background", "black" );

#### 描述:

隐藏一个表单中所有元素。

##### jQuery 代码:

$(myForm.elements).hide()

#### 描述:

在文档的第一个表单中，查找所有的单选按钮(即: type 值为 radio 的 input 元素)。

##### jQuery 代码:

$("input:radio", document.forms[0]);

#### 描述:

在一个由 AJAX 返回的 XML 文档中，查找所有的 div 元素。

##### jQuery 代码:

$("div", xml.responseXML);

## **each(callback)**

### 概述

以每一个匹配的元素作为上下文来执行一个函数。

意味着，每次执行传递进来的函数时，函数中的this关键字都指向一个不同的DOM元素（每次都是一个不同的匹配元素）。而且，在每次执行函数时，都会给函数传递一个表示作为执行环境的元素在匹配的元素集合中所处位置的数字值作为参数（从零开始的整型）。 返回 'false' 将停止循环 (就像在普通的循环中使用 'break')。返回 'true' 跳至下一个循环(就像在普通的循环中使用'continue')。

### 参数

#### ****callback****FunctionV1.0

对于每个匹配的元素所要执行的函数

### 示例

#### 描述:

迭代两个图像，并设置它们的 src 属性。注意:此处 this 指代的是 DOM 对象而非 jQuery 对象。

##### HTML 代码:

<img/><img/>

##### jQuery 代码:

$("img").each(function(i){

this.src = "test" + i + ".jpg";

});

##### 结果:

[ <img src="test0.jpg" />, <img src="test1.jpg" /> ]

#### 描述:

如果你想得到 jQuery对象，可以使用 $(this) 函数。

##### HTML 代码:

<button>Change colors</button>

<span></span>

<div></div>

<div></div>

<div></div>

<div></div>

<div id="stop">Stop here</div>

<div></div>

<div></div>

<div></div>

##### jQuery 代码:

$("img").each(function(){

$(this).toggleClass("example");

});

#### 描述:

你可以使用 'return' 来提前跳出 each() 循环。

##### HTML 代码:

<button>Change colors</button>

<span></span>

<div></div>

<div></div>

<div></div>

<div></div>

<div id="stop">Stop here</div>

<div></div>

<div></div>

<div></div>

##### jQuery 代码:

$("button").click(function () {

$("div").each(function (index, domEle) {

// domEle == this

$(domEle).css("backgroundColor", "yellow");

if ($(this).is("#stop")) {

$("span").text("Stopped at div index #" + index);

return false;

}

});

});

## **size()**

### V1.0概述

jQuery 对象中元素的个数。

当前匹配的元素个数。 <span title="Core/size">size</span> 将返回相同的值。

### 示例

#### 描述:

计算文档中所有图片数量

##### HTML 代码:

<img src="test1.jpg"/> <img src="test2.jpg"/>

##### jQuery 代码:

$("img").size();

##### 结果:

2

## **length**

### V1.0概述

jQuery 对象中元素的个数。

这个函数的返回值与 jQuery 对象的'<span title="Core/length">length</span>' 属性一致。

### 示例

#### 描述:

计算文档中所有图片数量

##### HTML 代码:

<img src="test1.jpg"/> <img src="test2.jpg"/>

##### jQuery 代码:

$("img").length;

##### 结果:

2

## **get([index])**

### 概述

取得其中一个匹配的元素。 num表示取得第几个匹配的元素。

这能够让你选择一个实际的DOM 元素并且对他直接操作，而不是通过 jQuery 函数。$(this).get(0)与$(this)[0]等价。

### 参数

#### ****[index]****Number

取得第 index 个位置上的元素

#### ****get()****

取得所有匹配的 DOM 元素集合。

### 示例

#### 描述:

##### HTML 代码:

<img src="test1.jpg"/> <img src="test2.jpg"/>

##### jQuery 代码:

$("img").get(0);

##### 结果:

[ <img src="test1.jpg"/> ]

#### 描述:

选择文档中所有图像作为元素数组，并用数组内建的 reverse 方法将数组反向。

##### HTML 代码:

<img src="test1.jpg"/> <img src="test2.jpg"/>

##### jQuery 代码:

$("img").get().reverse();

##### 结果:

[ <img src="test2.jpg"/> <img src="test1.jpg"/> ]

## **data([key],[value])**

### 概述

在元素上存放数据,返回jQuery对象。

如果jQuery集合指向多个元素，那将在所有元素上设置对应数据。 这个函数不用建立一个新的expando，就能在一个元素上存放任何格式的数据，而不仅仅是字符串。

V1.4.3 新增用法NEW data(obj) 可传入key-value形式的数据。

### 参数

#### ****key****StringV1.23

存储的数据名.

#### ****key,value****String,AnyV1.2.3

****key****:存储的数据名

****value****:将要存储的任意数据

#### ****obj****objectV1.4.3

一个用于设置数据的键/值对

#### ****data()****V1.4.3

### 示例

#### 描述:

在一个div上存取数据

##### HTML 代码:

<div></div>

##### jQuery 代码:

$("div").data("blah"); // undefined

$("div").data("blah", "hello"); // blah设置为hello

$("div").data("blah"); // hello

$("div").data("blah", 86); // 设置为86

$("div").data("blah"); // 86

$("div").removeData("blah"); //移除blah

$("div").data("blah"); // undefined

#### 描述:

在一个div上存取名/值对数据

##### HTML 代码:

<div></div>

##### jQuery 代码:

$("div").data("test", { first: 16, last: "pizza!" });

$("div").data("test").first //16;

$("div").data("test").last //pizza!;

## **queue(element,[queueName])**

### 概述

显示或操作在匹配元素上执行的函数队列

### 参数

#### ****element,[queueName]****Element,StringV1.3

****element****:检查附加列队的DOM元素

****queueName****:字符串值，包含序列的名称。默认是 fx, 标准的效果序列。

#### ****element,queueName,newQueue****Element,String,ArrayV1.3

****element****:检查附加列队的DOM元素

****queueName****:字符串值，包含序列的名称。默认是 fx, 标准的效果序列。

****newQueue****:替换当前函数列队内容的数组

#### ****element,queueName,callback()****Element,StringV1.3

****element****:检查附加列队的DOM元素

****queueName****:字符串值，包含序列的名称。默认是 fx, 标准的效果序列。

****callback()****:要添加进队列的函数

### 示例

#### 描述:

显示队列长度

##### HTML 代码:

<style>

div { margin:3px; width:40px; height:40px;

position:absolute; left:0px; top:30px;

background:green; display:none; }

div.newcolor { background:blue; }

span { color:red; }

</style>

<button id="show">Show Length of Queue</button>

<span></span>

<div></div>

##### jQuery 代码:

$("#show").click(function () {

var n = $("div").queue("fx");

$("span").text("Queue length is: " + n.length);

});

function runIt() {

$("div").show("slow");

$("div").animate({left:'+=200'},2000);

$("div").slideToggle(1000);

$("div").slideToggle("fast");

$("div").animate({left:'-=200'},1500);

$("div").hide("slow");

$("div").show(1200);

$("div").slideUp("normal", runIt);

}

runIt();

#### 描述:

通过设定队列数组来删除动画队列

##### HTML 代码:

<style>

div { margin:3px; width:40px; height:40px;

position:absolute; left:0px; top:30px;

background:green; display:none; }

div.newcolor { background:blue; }

</style>

<button id="start">Start</button>

<button id="stop">Stop</button>

<div></div>

##### jQuery 代码:

$("#start").click(function () {

$("div").show("slow");

$("div").animate({left:'+=200'},5000);

$("div").queue(function () {

$(this).addClass("newcolor");

$(this).dequeue();

});

$("div").animate({left:'-=200'},1500);

$("div").queue(function () {

$(this).removeClass("newcolor");

$(this).dequeue();

});

$("div").slideUp();

});

$("#stop").click(function () {

$("div").queue("fx", []);

$("div").stop();

});

#### 描述:

插入一个自定义函数 如果函数执行后要继续队列，则执行 jQuery(this).dequeue();

##### HTML 代码:

<style>

div { margin:3px; width:40px; height:40px;

position:absolute; left:0px; top:30px;

background:green; display:none; }

div.newcolor { background:blue; }

</style>

Click here...

<div></div>

##### jQuery 代码:

$(document.body).click(function () {

$("div").show("slow");

$("div").animate({left:'+=200'},2000);

$("div").queue(function () {

$(this).addClass("newcolor");

$(this).dequeue();

});

$("div").animate({left:'-=200'},500);

$("div").queue(function () {

$(this).removeClass("newcolor");

$(this).dequeue();

});

$("div").slideUp();

});

## **dequeue([queueName])**

### 概述

从队列最前端移除一个队列函数，并执行他。

### 参数

#### ****[queueName]****StringV1.2

队列名，默认为fx

### 示例

#### 描述:

使用 dequeue() 终止一个自定义的队列函数

##### jQuery 代码:

$("div").queue(function () {

$(this).toggleClass("red");

$(this).dequeue();

});

#### 描述:

用dequeue来结束自定义队列函数，并让队列继续进行下去。

##### HTML 代码:

<style>

div { margin:3px; width:50px; position:absolute;

height:50px; left:10px; top:30px;

background-color:yellow; }

div.red { background-color:red; }

</style>

<button>Start</button>

<div></div>

##### jQuery 代码:

$("button").click(function () {

$("div").animate({left:'+=200px'}, 2000);

$("div").animate({top:'0px'}, 600);

$("div").queue(function () {

$(this).toggleClass("red");

$(this).dequeue();

});

$("div").animate({left:'10px', top:'30px'}, 700);

});

## **clearQueue([queueName])**

### 概述

清空对象上尚未执行的所有队列

如果不带参数，则默认清空的是动画队列。这跟stop(true)类似，但stop()只能清空动画队列，而这个可以清空所有通过 .queue() 创建的队列。

### 参数

#### ****queueName****BooleanV1.4

含有队列名的字符串。默认是"Fx"，动画队列。

### 示例

#### 描述:

停止当前正在运行的动画：

##### jQuery 代码:

$("#stop").click(function(){

$("#box").clearQueue();

});

## **jQuery.noConflict([extreme])**

### 概述

运行这个函数将变量$的控制权让渡给第一个实现它的那个库。

这有助于确保jQuery不会与其他库的$对象发生冲突。 在运行这个函数后，就只能使用jQuery变量访问jQuery对象。例如，在要用到$("div p")的地方，就必须换成jQuery("div p")。 '''注意:'''这个函数必须在你导入jQuery文件之后，并且在导入另一个导致冲突的库'''之前'''使用。当然也应当在其他冲突的库被使用之前，除非jQuery是最后一个导入的。

### 参数

#### ****extreme****BooleanV1.0

传入 true 来允许彻底将jQuery变量还原

### 示例

#### 描述:

将$引用的对象映射回原始的对象。

##### jQuery 代码:

jQuery.noConflict();

// 使用 jQuery

jQuery("div p").hide();

// 使用其他库的 $()

$("content").style.display = 'none';

#### 描述:

恢复使用别名$，然后创建并执行一个函数，在这个函数的作用域中仍然将$作为jQuery的别名来使用。在这个函数中，原来的$对象是无效的。这个函数对于大多数不依赖于其他库的插件都十分有效。

##### jQuery 代码:

jQuery.noConflict();

(function($) {

$(function() {

// 使用 $ 作为 jQuery 别名的代码

});

})(jQuery);

// 其他用 $ 作为别名的库的代码

#### 描述:

创建一个新的别名用以在接下来的库中使用jQuery对象。

##### jQuery 代码:

var j = jQuery.noConflict();

// 基于 jQuery 的代码

j("div p").hide();

// 基于其他库的 $() 代码

$("content").style.display = 'none';

#### 描述:

完全将 jQuery 移到一个新的命名空间。

##### jQuery 代码:

var dom = {};

dom.query = jQuery.noConflict(true);

##### 结果:

// 新 jQuery 的代码

dom.query("div p").hide();

// 另一个库 $() 的代码

$("content").style.display = 'none';

// 另一个版本 jQuery 的代码

jQuery("div > p").hide();

**Jquery中的属性**

## **attr(name|properties|key,value|fn)**

### 概述

设置或返回被选元素的属性值。

### 参数

#### ****name****StringV1.0

属性名称

#### ****properties****MapV1.0

作为属性的“名/值对”对象

#### ****key,value****String,ObjectV1.0

属性名称，属性值

#### ****key,function(index, attr)****String,FunctionV1.1

1:属性名称。

2:返回属性值的函数,第一个参数为当前元素的索引值，第二个参数为原先的属性值。

### 示例

#### 参数name 描述:

返回文档中所有图像的src属性值。

##### jQuery 代码:

$("img").attr("src");

#### 参数properties 描述:

为所有图像设置src和alt属性。

##### jQuery 代码:

$("img").attr({ src: "test.jpg", alt: "Test Image" });

#### 参数key,value 描述:

为所有图像设置src属性。

##### jQuery 代码:

$("img").attr("src","test.jpg");

#### 参数key,回调函数 描述:

把src属性的值设置为title属性的值。

##### jQuery 代码:

$("img").attr("title", function() { return this.src });

## **removeAttr(name)**

### 概述

从每一个匹配的元素中删除一个属性

1.6以下版本在IE6使用JQuery的removeAttr方法删除disabled是无效的。解决的方法就是使用$("XX").prop("disabled",false);  
1.7版本在IE6下已支持删除disabled。

### 参数

#### ****name****StringV1.0

要删除的属性名

### 示例

#### 描述:

将文档中图像的src属性删除

##### HTML 代码:

<img src="test.jpg"/>

##### jQuery 代码:

$("img").removeAttr("src");

##### 结果:

[ <img /> ]

## **prop(name|properties|key,value|fn)**

### 概述

获取在匹配的元素集中的第一个元素的属性值。

随着一些内置属性的DOM元素或window对象，如果试图将删除该属性，浏览器可能会产生错误。jQuery第一次分配undefined值的属性，而忽略了浏览器生成的任何错误

### 参数

#### ****name****StringV1.6

属性名称

#### ****properties****MapV1.6

作为属性的“名/值对”对象

#### ****key,value****String,ObjectV1.6

属性名称，属性值

#### ****key,function(index, attr)****String,FunctionV1.6

1:属性名称。

2:返回属性值的函数,第一个参数为当前元素的索引值，第二个参数为原先的属性值。

### 示例

#### 参数name 描述:

选中复选框为true，没选中为false

##### jQuery 代码:

$("input[type='checkbox']").prop("checked");

#### 参数properties 描述:

禁用页面上的所有复选框。

##### jQuery 代码:

$("input[type='checkbox']").prop({

disabled: true

});

#### 参数key,value 描述:

禁用和选中所有页面上的复选框。

##### jQuery 代码:

$("input[type='checkbox']").prop("disabled", false);

$("input[type='checkbox']").prop("checked", true);

#### 参数key,回调函数 描述:

通过函数来设置所有页面上的复选框被选中。

##### jQuery 代码:

$("input[type='checkbox']").prop("checked", function( i, val ) {

return !val;

});

## **removeProp(name)**

### 概述

用来删除由.prop()方法设置的属性集

随着一些内置属性的DOM元素或window对象，如果试图将删除该属性，浏览器可能会产生错误。jQuery第一次分配undefined值的属性，而忽略了浏览器生成的任何错误

### 参数

#### ****propertyName****StringV1.6

要删除的属性名

### 示例

#### 描述:

设置一个段落数字属性，然后将其删除。

##### HTML 代码:

<p> </p>

##### jQuery 代码:

var $para = $("p");

$para.prop("luggageCode", 1234);

$para.append("The secret luggage code is: ", String($para.prop("luggageCode")), ". ");

$para.removeProp("luggageCode");

$para.append("Now the secret luggage code is: ", String($para.prop("luggageCode")), ". ");

##### 结果:

The secret luggage code is: 1234. Now the secret luggage code is: undefined.

## **addClass(class|fn)**

### 概述

为每个匹配的元素添加指定的类名。

### 参数

#### ****class****StringV1.0

一个或多个要添加到元素中的CSS类名，请用空格分开

#### ****function(index, class)****FunctionV1.4

此函数必须返回一个或多个空格分隔的class名。接受两个参数，index参数为对象在这个集合中的索引值，class参数为这个对象原先的class属性值。

### 示例

#### 参数class 描述:

为匹配的元素加上 'selected' 类

##### jQuery 代码:

$("p").addClass("selected");

$("p").addClass("selected1 selected2");

#### 回调函数 描述:

给li加上不同的class

##### HTML 代码:

<ul>

<li>Hello</li>

<li>Hello</li>

<li>Hello</li>

</ul>

##### jQuery 代码:

$('ul li:last').addClass(function() {

return 'item-' + $(this).index();

});

## **removeClass([class|fn])**

### 概述

从所有匹配的元素中删除全部或者指定的类。

### 参数

#### ****class****StringV1.0

一个或多个要删除的CSS类名，请用空格分开

#### ****function(index, class)****FunctionV1.4

此函数必须返回一个或多个空格分隔的class名。接受两个参数，index参数为对象在这个集合中的索引值，class参数为这个对象原先的class属性值。

### 示例

#### 参数class 描述:

从匹配的元素中删除 'selected' 类

##### jQuery 代码:

$("p").removeClass("selected");

#### 参数class 描述:

删除匹配元素的所有类

##### jQuery 代码:

$("p").removeClass();

#### 回调函数描述:

删除最后一个元素上与前面重复的class

##### jQuery 代码:

$('li:last').removeClass(function() {

return $(this).prev().attr('class');

});

## **toggleClass(class|fn[,sw])**

### 概述

如果存在（不存在）就删除（添加）一个类。

### 参数

#### ****class****StringV1.0

CSS类名

#### ****class,switch****String,BooleanV1.3

1:要切换的CSS类名.

2:用于决定元素是否包含class的布尔值。

#### ****switch****BooleanV1.4

用于决定元素是否包含class的布尔值。

#### ****function(index, class,switch)[, switch]****Function,BooleanV1.4

1:用来返回在匹配的元素集合中的每个元素上用来切换的样式类名的一个函数。接收元素的索引位置和元素旧的样式类作为参数。

2: 一个用来判断样式类添加还是移除的 boolean 值。

### 示例

#### 参数class 描述:

为匹配的元素切换 'selected' 类

##### jQuery 代码:

$("p").toggleClass("selected");

#### 参数class,switch 描述:

每点击三下加上一次 'highlight' 类

##### HTML 代码:

<strong>jQuery 代码:</strong>

##### jQuery 代码:

var count = 0;

$("p").click(function(){

$(this).toggleClass("highlight", count++ % 3 == 0);

});

#### 回调函数 描述:

根据父元素来设置class属性

##### jQuery 代码:

$('div.foo').toggleClass(function() {

if ($(this).parent().is('.bar') {

return 'happy';

} else {

return 'sad';

}

});

## **html([val|fn])**

### 概述

取得第一个匹配元素的html内容。这个函数不能用于XML文档。但可以用于XHTML文档。

在一个 HTML 文档中, 我们可以使用 .html() 方法来获取任意一个元素的内容。 如果选择器匹配多于一个的元素，那么只有第一个匹配元素的 HTML 内容会被获取。

### 参数

#### ****val****StringV1.0

用于设定HTML内容的值

#### ****function(index, html)****FunctionV1.4

此函数返回一个HTML字符串。接受两个参数，index为元素在集合中的索引位置，html为原先的HTML值。

### 示例

#### 无参数 描述:

返回p元素的内容。

##### jQuery 代码:

$('p').html();

#### 参数val 描述:

设置所有 p 元素的内容

##### jQuery 代码:

$("p").html("Hello <b>world</b>!");

#### 回调函数描述:

使用函数来设置所有匹配元素的内容。

##### jQuery 代码:

$("p").html(function(n){

return "这个 p 元素的 index 是：" + n;

});

## **text([val|fn])**

### 概述

取得所有匹配元素的内容。

结果是由所有匹配元素包含的文本内容组合起来的文本。这个方法对HTML和XML文档都有效。

### 参数

#### ****val****StringV1.0

用于设置元素内容的文本

#### ****function(index, text)****FunctionV1.4

此函数返回一个字符串。接受两个参数，index为元素在集合中的索引位置，text为原先的text值。

### 示例

#### 无参数 描述:

返回p元素的文本内容。

##### jQuery 代码:

$('p').text();

#### 参数val 描述:

设置所有 p 元素的文本内容

##### jQuery 代码:

$("p").text("Hello world!");

#### 回调函数 描述:

使用函数来设置所有匹配元素的文本内容。

##### jQuery 代码:

$("p").text(function(n){

return "这个 p 元素的 index 是：" + n;

});

## **val([val|fn|arr])**

### 概述

获得匹配元素的当前值。

在 jQuery 1.2 中,可以返回任意元素的值了。包括select。如果多选，将返回一个数组，其包含所选的值。

### 参数

#### ****val****StringV1.0

要设置的值。

#### ****function(index, value)****FunctionV1.4

此函数返回一个要设置的值。接受两个参数，index为元素在集合中的索引位置，text为原先的text值。

#### ****array****Array<String>V1.0

用于 check/select 的值

### 示例

#### 无参数 描述:

获取文本框中的值

##### jQuery 代码:

$("input").val();

#### 参数val 描述:

设定文本框的值

##### jQuery 代码:

$("input").val("hello world!");

#### 回调函数 描述:

设定文本框的值

##### jQuery 代码:

$('input:text.items').val(function() {

return this.value + ' ' + this.className;

});

#### 参数array 描述:

设定一个select和一个多选的select的值

##### HTML 代码:

<select id="single">

<option>Single</option>

<option>Single2</option>

</select>

<select id="multiple" multiple="multiple">

<option selected="selected">Multiple</option>

<option>Multiple2</option>

<option selected="selected">Multiple3</option>

</select><br/>

<input type="checkbox" value="check1"/> check1

<input type="checkbox" value="check2"/> check2

<input type="radio" value="radio1"/> radio1

<input type="radio" value="radio2"/> radio2

##### jQuery 代码:

$("#single").val("Single2");

$("#multiple").val(["Multiple2", "Multiple3"]);

$("input").val(["check2", "radio1"]);