

---

# JQUERY基础

---

素材来源: [w3school jQuery](#)

起始时间: 2022/01/01

笔记作者: iVikey

## 1.JQUERY基础

### H3 1.jQuery基础语法

基础语法是: `$(selector).action()`

- 美元符号定义 jQuery
- 选择符 (selector) “查询”和“查找” HTML 元素
- jQuery 的 `action()` 执行对元素的操作

### H3 2.文档就绪函数

您也许已经注意到在我们的实例中的所有 jQuery 函数位于一个 document ready 函数中:

```
$(document).ready(function(){  
  
    --- jQuery functions go here ---  
  
});  
//相当于  
window.onload = function(){  
    --- javascript functions go here ---  
}
```

这是为了防止文档在完全加载（就绪）之前运行 jQuery 代码。

如果在文档没有完全加载之前就运行函数，操作可能失败。下面是两个具体的例子：

- 试图隐藏一个不存在的元素
- 获得未完全加载的图像的大小

## H3 3.jQuery选择器

### H4 1. jQuery 元素选择器

jQuery 使用 CSS 选择器来选取 HTML 元素。

`$("p")` 选取

元素。

`$("p.intro")` 选取所有 `class="intro"` 的

元素。

`$("#demo")` 选取所有 `id="demo"` 的

元素。

## H5 2. jQuery 属性选择器

jQuery 使用 XPath 表达式来选择带有给定属性的元素。

`$("[href]")` 选取所有带有 href 属性的元素。

`$("[href='#']")` 选取所有带有 href 值等于 "#" 的元素。

`$("[href!='#']")` 选取所有带有 href 值不等于 "#" 的元素。

`$("[href$='.jpg']")` 选取所有 href 值以 ".jpg" 结尾的元素。

## H5 3. jQuery CSS 选择器

jQuery CSS 选择器可用于改变 HTML 元素的 CSS 属性。

下面的例子把所有 p 元素的背景颜色更改为红色：

### 实例



```
$("p").css("background-color", "red");
```

## H5 4. jQuery 事件

下面是 jQuery 中事件方法的一些例子：

Event 函数	绑定函数至
<code>\$(document).ready(function)</code>	将函数绑定到文档的就绪事件（当文档完成加载时）
<code>\$(selector).click(function)</code>	触发或将函数绑定到被选元素的点击事件
<code>\$(selector).dblclick(function)</code>	触发或将函数绑定到被选元素的双击事件

Event 函数	绑定函数至
<code>\$(selector).focus(function)</code>	触发或将函数绑定到被选元素的获得焦点事件
<code>\$(selector).mouseover(function)</code>	触发或将函数绑定到被选元素的鼠标悬停事件

如需完整的参考手册，请访问我们的 [jQuery 事件参考手册](#)。

---

## 2.JQUERY效果

### H3 1.显示 / 隐藏

#### H4 jQuery hide() 和 show()

通过 jQuery，您可以使用 `hide()` 和 `show()` 方法来隐藏和显示 HTML 元素：

```
$( "#hide" ).click(function(){
    $( "p" ).hide();
});

$( "#show" ).click(function(){
    $( "p" ).show();
});
```

[试一试](#)

## 语法:

```
$(selector).hide(speed,callback);  
  
$(selector).show(speed,callback);
```

可选的 speed 参数规定隐藏/显示的速度，可以取以下值："slow"、"fast" 或毫秒。

可选的 callback 参数是隐藏或显示完成后所执行的函数名称。

下面的例子演示了带有 speed 参数的 hide() 方法：

## 实例

```
$("#button").click(function(){  
    $("#p").hide(1000);  
});
```

[试一试](#)

## H4 jQuery toggle()

通过 jQuery，您可以使用 toggle() 方法来切换 hide() 和 show() 方法。

显示被隐藏的元素，并隐藏已显示的元素：

## 实例

```
$("#button").click(function(){  
    $("#p").toggle();  
});
```

[试一试](#)

语法:

```
$(selector).toggle(speed, callback);
```

可选的 speed 参数规定隐藏/显示的速度，可以取以下值："slow"、"fast" 或毫秒。

可选的 callback 参数是 toggle() 方法完成后所执行的函数名称。

## H3 2.淡入淡出

### H4 jQuery Fading 方法

通过 jQuery，您可以实现元素的淡入淡出效果。

jQuery 拥有下面四种 fade 方法：

- fadeIn()
- fadeOut()
- fadeToggle()
- fadeTo()

#### H5 1. jQuery fadeIn()

jQuery fadeIn() 用于淡入已隐藏的元素。

语法:

```
$(selector).fadeIn(speed, callback);
```

可选的 speed 参数规定效果的时长。它可以取以下值："slow"、"fast" 或毫秒。

可选的 callback 参数是 fading 完成后所执行的函数名称。

下面的例子演示了带有不同参数的 fadeIn() 方法：

### 实例

```
$("button").click(function(){  
    $("#div1").fadeIn();  
    $("#div2").fadeIn("slow");  
    $("#div3").fadeIn(3000);  
});
```

[试一试](#)

## H5 2. jQuery fadeOut()

jQuery fadeOut() 方法用于淡出可见元素。

### 语法：

```
$(selector).fadeOut(speed, callback);
```

可选的 speed 参数规定效果的时长。它可以取以下值："slow"、"fast" 或毫秒。

可选的 callback 参数是 fading 完成后所执行的函数名称。

下面的例子演示了带有不同参数的 fadeOut() 方法：

### 实例

```
$("button").click(function(){  
    $("#div1").fadeOut();  
    $("#div2").fadeOut("slow");  
    $("#div3").fadeOut(3000);  
});
```

[试一试](#)

## H5 3. jQuery fadeToggle()

jQuery fadeToggle() 方法可以在 fadeIn() 与 fadeOut() 方法之间进行切换。

如果元素已淡出，则 fadeToggle() 会向元素添加淡入效果。

如果元素已淡入，则 fadeToggle() 会向元素添加淡出效果。

语法：

```
$(selector).fadeToggle(speed,callback);
```

可选的 speed 参数规定效果的时长。它可以取以下值："slow"、"fast" 或毫秒。

可选的 callback 参数是 fading 完成后所执行的函数名称。

下面的例子演示了带有不同参数的 fadeToggle() 方法：

实例

```
$("#button").click(function(){
    $("#div1").fadeToggle();
    $("#div2").fadeToggle("slow");
    $("#div3").fadeToggle(3000);
});
```

[试一试](#)

## H5 4. jQuery fadeTo()

jQuery fadeTo() 方法允许渐变为给定的不透明度（值介于 0 与 1 之间）。

语法：





```
$(selector).fadeTo(speed,opacity,callback);
```

必需的 speed 参数规定效果的时长。它可以取以下值："slow"、"fast" 或毫秒。

fadeTo() 方法中必需的 opacity 参数将淡入淡出效果设置为给定的不透明度（值介于 0 与 1 之间）。

可选的 callback 参数是该函数完成后所执行的函数名称。

下面的例子演示了带有不同参数的 fadeTo() 方法：

### 实例



```
$("#button").click(function(){  
    $("#div1").fadeTo("slow",0.15);  
    $("#div2").fadeTo("slow",0.4);  
    $("#div3").fadeTo("slow",0.7);  
});
```

试一试

## H3 3.滑动

### H4 jQuery 滑动方法

通过 jQuery，您可以在元素上创建滑动效果。

jQuery 拥有以下滑动方法：

- slideDown()
- slideUp()
- slideToggle()

## H5 1. jQuery slideDown()

jQuery slideDown() 方法用于向下滑动元素。

语法：

```
$(selector).slideDown(speed,callback);
```

可选的 speed 参数规定效果的时长。它可以取以下值："slow"、"fast" 或毫秒。

可选的 callback 参数是滑动完成后所执行的函数名称。

下面的例子演示了 slideDown() 方法：

实例

```
$("#flip").click(function(){  
    $("#panel").slideDown();  
});
```

[试一试](#)

## H5 2. jQuery slideUp()

jQuery slideUp() 方法用于向上滑动元素。

语法：

```
$(selector).slideUp(speed,callback);
```

可选的 speed 参数规定效果的时长。它可以取以下值："slow"、"fast" 或毫秒。

可选的 callback 参数是滑动完成后所执行的函数名称。

下面的例子演示了 slideUp() 方法：

## 实例



```
$("#flip").click(function(){  
    $("#panel").slideUp();  
});
```

[试一试](#)

## H5 3. jQuery slideToggle()

jQuery slideToggle() 方法可以在 slideDown() 与 slideUp() 方法之间进行切换。

如果元素向下滑动，则 slideToggle() 可向上滑动它们。

如果元素向上滑动，则 slideToggle() 可向下滑动它们。



```
$(selector).slideToggle(speed,callback);
```

可选的 speed 参数规定效果的时长。它可以取以下值："slow"、"fast" 或毫秒。

可选的 callback 参数是滑动完成后所执行的函数名称。

下面的例子演示了 slideToggle() 方法：

## 实例



```
$("#flip").click(function(){  
    $("#panel").slideToggle();  
});
```

[试一试](#)

## H3 4. 动画

### H4 jQuery 动画 - animate()

jQuery animate() 方法用于创建自定义动画。

语法：

```
$(selector).animate({params}, speed, callback);
```

必需的 params 参数定义形成动画的 CSS 属性。

可选的 speed 参数规定效果的时长。它可以取以下值："slow"、"fast" 或毫秒。

可选的 callback 参数是动画完成后所执行的函数名称。

下面的例子演示 animate() 方法的简单应用；它把

元素移动到左边，直到 left 属性等于 250 像素为止：

实例

```
$("#button").click(function(){
    $("#div").animate({left: '250px'});
});
```

试一试

**提示：**默认地，所有 HTML 元素都有一个静态位置，且无法移动。

如需对位置进行操作，要记得首先把元素的 CSS position 属性设置为 relative、fixed 或 absolute！

## H4 animate() - 操作多属性

请注意，生成动画的过程中可同时使用多个属性：

### 实例

```
$("button").click(function(){  
    $("div").animate({  
        left: '250px',  
        opacity: '0.5',  
        height: '150px',  
        width: '150px'  
    });  
});
```

试一试

**提示：** 可以用 animate() 方法来操作所有 CSS 属性吗？

是的，几乎可以！不过，需要记住一件重要的事情：当使用 animate() 时，必须使用 Camel 标记法书写所有的属性名，比如，必须使用 paddingLeft 而不是 padding-left，使用 marginRight 而不是 margin-right，等等。

同时，色彩动画并不包含在核心 jQuery 库中。

如果需要生成颜色动画，您需要从 jQuery.com 下载 Color Animations 插件。

## H4 animate() - 使用相对值

也可以定义相对值（该值相对于元素的当前值）。需要在值的前面加上 += 或 -=：

### 实例

```
$("button").click(function(){
    $("div").animate({
        left: '250px',
        height: '+=150px',
        width: '+=150px'
    });
});
```

[试一试](#)

## H4 animate() - 预定义的值

您甚至可以把属性的动画值设置为 "show"、"hide" 或 "toggle"：

### 实例

```
$("button").click(function(){
    $("div").animate({
        height: 'toggle'
    });
});
```

[试一试](#)

## H4 animate() - 队列功能

默认地，jQuery 提供针对动画的队列功能。

这意味着如果您在彼此之后编写多个 animate() 调用，jQuery 会创建包含这些方法调用的“内部”队列。然后逐一运行这些 animate 调用。

### 实例 1

隐藏，如果您希望在彼此之后执行不同的动画，那么我们要利用队列功能：



```
$("#button").click(function(){  
    var div=$("#div");  
    div.animate({height: '300px', opacity: '0.4'}, "slow");  
    div.animate({width: '300px', opacity: '0.8'}, "slow");  
    div.animate({height: '100px', opacity: '0.4'}, "slow");  
    div.animate({width: '100px', opacity: '0.8'}, "slow");  
});
```

试一试

## 实例 2

下面的例子把

元素移动到右边，然后增加文本的字号：



```
$("#button").click(function(){  
    var div=$("#div");  
    div.animate({left: '100px'}, "slow");  
    div.animate({fontSize: '3em'}, "slow");  
});
```

试一试

## H3 5.stop()

## H4 jQuery stop() 方法

jQuery stop() 方法用于停止动画或效果，在它们完成之前。

stop() 方法适用于所有 jQuery 效果函数，包括滑动、淡入淡出和自定义动画。

语法



```
$(selector).stop(stopAll,goToEnd);
```

可选的 `stopAll` 参数规定是否应该清除动画队列。默认是 `false`，即仅停止活动的动画，允许任何排入队列的动画向后执行。

可选的 `goToEnd` 参数规定是否立即完成当前动画。默认是 `false`。

因此，默认地，`stop()` 会清除在被选元素上指定的当前动画。

下面的例子演示 `stop()` 方法，不带参数：

### 实例



```
$("#stop").click(function(){  
    $("#panel").stop();  
});
```

## H3 6. Callback

许多 jQuery 函数涉及动画。这些函数也许会将 `speed` 或 `duration` 作为可选参数。

例子： `$("p").hide("slow")`

`speed` 或 `duration` 参数可以设置许多不同的值，比如 "slow", "fast", "normal" 或毫秒。

由于 JavaScript 语句（指令）是逐一执行的 - 按照次序，动画之后的语句可能会产生错误或页面冲突，因为动画还没有完成。

为了避免这个情况，您可以以参数的形式添加 Callback 函数。

### 语法



```
$(selector).hide(speed,callback)
```



当动画执行100%后将会执行返回函数。

实例：

```
$("p").hide(1000,function(){  
    alert("The page is now hidden");  
})
```

## H3 7.Chaining

### jQuery 方法链接

有时我们需要写很多语句来操作一条效果。

但实际上我们只需要使用方法链接就可以使多条语句相连。

#### 实例1

下边的例子把css()、slideUp()、slideDown()链接到一起。"P标签将会变为红色然后向上滑动，再向下滑动

```
$("p").css("color","red").slideUp(2000).slideDown(2000);
```

#### 实例2

也可以这样写

```
$("p").css("color","red")  
    .slideUp(2000)  
    .slideDown(2000);
```

## H3 jQuery 效果参考手册

如需全面查阅 jQuery 效果，请访问我们的 [jQuery 效果参考手册](#)。

---

# 3. DOM操作

## H3 1.DOM元素的获取

三个简单实用的用于 DOM 操作的 jQuery 方法：

- `text()` - 设置或返回所选元素的文本内容
- `html()` - 设置或返回所选元素的内容（包括 HTML 标记）
- `val()` - 设置或返回表单字段的值

下面的例子演示如何通过 jQuery `text()` 和 `html()` 方法来获得内容：

### 实例

```
$("#btn1").click(function(){
    alert("Text: " + $("#test").text());
});
$("#btn2").click(function(){
    alert("HTML: " + $("#test").html());
});
```

[试一试](#)

下面的例子演示如何通过 jQuery `val()` 方法获得输入字段的值：

## 实例

```
$("#btn1").click(function(){
    alert("Value: " + $("#test").val());
});
```

[试一试](#)

## H4 获取属性 - attr()

jQuery attr() 方法用于获取属性值。

下面的例子演示如何获得链接中 href 属性的值：

## 实例

```
$("#button").click(function(){
    alert($("#w3s").attr("href"));
});
```

[试一试](#)

## H3 2.DOM元素的设置

## H4 设置内容

我们将使用前一章中的三个相同的方法来设置内容：

- text() - 设置或返回所选元素的文本内容
- html() - 设置或返回所选元素的内容（包括 HTML 标记）
- val() - 设置或返回表单字段的值

下面的例子演示如何通过 text()、html() 以及 val() 方法来设置内容：

## 实例

```
$("#btn1").click(function(){
    $("#test1").text("Hello world!");
});
$("#btn2").click(function(){
    $("#test2").html("<b>Hello world!</b>");
});
$("#btn3").click(function(){
    $("#test3").val("Dolly Duck");
});
```

试一试

## H4 text() 的回调函数

上面的三个 jQuery 方法：text()、html() 以及 val()，同样拥有回调函数。回调函数由两个参数：被选元素列表中当前元素的下标，以及原始（旧的）值。然后以函数新值返回您希望使用的字符串。

下面的例子演示带有回调函数的 text() 和 html()：

## 实例

```
$("#btn1").click(function(){
    $("#test1").text(function(i,origText){
        return "Old text: " + origText + " New text: Hello world!
        (index: " + i + ")";
    });
});

$("#btn2").click(function(){
    $("#test2").html(function(i,origText){
        return "Old html: " + origText + " New html: Hello <b>world!</b>
        (index: " + i + ")";
    });
});
```

试一试

## H4 设置属性 - attr()

jQuery attr() 方法也用于设置/改变属性值。

下面的例子演示如何改变（设置）链接中 href 属性的值：

### 实例

```
$("button").click(function(){  
    $("#w3s").attr("href", "http://www.w3school.com.cn/jquery");  
});
```

试一试

attr() 方法也允许您同时设置多个属性。

下面的例子演示如何同时设置 href 和 title 属性：

### 实例

```
$("button").click(function(){  
    $("#w3s").attr({  
        "href" : "http://www.w3school.com.cn/jquery",  
        "title" : "W3School jQuery Tutorial"  
    });  
});
```

试一试

## H4 attr() 的回调函数

jQuery 方法 attr(), 也提供回调函数。回调函数由两个参数：被选元素列表中当前元素的下标, 以及原始（旧的）值。然后以函数新值返回您希望使用的字符串。

下面的例子演示带有回调函数的 attr() 方法：

#### 实例

```
$("button").click(function(){  
    $("#w3s").attr("href", function(i,origValue){  
        return origValue + "/jquery";  
    });  
});
```

[试一试](#)

### H3 3.DOM元素的添加

我们将学习用于添加新内容的四个 jQuery 方法：

- append() - 在被选元素的结尾插入内容
- prepend() - 在被选元素的开头插入内容
- after() - 在被选元素之后插入内容
- before() - 在被选元素之前插入内容

#### H4 append() 方法

jQuery append() 方法在被选元素的结尾插入内容。

#### 实例

```
$("p").append("Some appended text.");
```

[试一试](#)

## H4 prepend() 方法

jQuery prepend() 方法在被选元素的开头插入内容。

### 实例

```
$("#p").prepend("Some prepended text.");
```

[试一试](#)

## H4 通过 append() 和 prepend() 方法添加若干新元素

在上面的例子中，我们只在被选元素的开头/结尾插入文本/HTML。

不过，append() 和 prepend() 方法能够通过参数接收无限数量的新元素。可以通过 jQuery 来生成文本/HTML（就像上面的例子那样），或者通过 JavaScript 代码和 DOM 元素。

在下面的例子中，我们创建若干个新元素。这些元素可以通过 text/HTML、jQuery 或者 JavaScript/DOM 来创建。然后通过 append() 方法把这些新元素追加到文本中（对 prepend() 同样有效）：

### 实例

```
function appendText()
{
    var txt1("<p>Text.</p>");           // 以 HTML 创建新元素
    var txt2($("#<p></p>").text("Text."); // 以 jQuery 创建新元素
    var txt3=document.createElement("p"); // 以 DOM 创建新元素
    txt3.innerHTML="Text.";
    $("#p").append(txt1,txt2,txt3);      // 追加新元素
}
```

[试一试](#)

## H4 after() 和 before() 方法

jQuery after() 方法在被选元素之后插入内容。

jQuery before() 方法在被选元素之前插入内容。

### 实例

```
$("#img").after("Some text after");

$("#img").before("Some text before");
```

[试一试](#)

## H4 通过 after() 和 before() 方法添加若干新元素

after() 和 before() 方法能够通过参数接收无限数量的新元素。可以通过 text/HTML、jQuery 或者 JavaScript/DOM 来创建新元素。

在下面的例子中，我们创建若干新元素。这些元素可以通过 text/HTML、jQuery 或者 JavaScript/DOM 来创建。然后我们通过 after() 方法把这些新元素插到文本中（对 before() 同样有效）：

### 实例

```
function afterText()
{
    var txt1("<b>I </b>");           // 以 HTML 创建新元素
    var txt2=$("#<i></i>").text("love "); // 通过 jQuery 创建新元素
    var txt3=document.createElement("big"); // 通过 DOM 创建新元素
    txt3.innerHTML="jQuery!";
    $("#img").after(txt1,txt2,txt3);    // 在 img 之后插入新元素
}
```

[试一试](#)



## H3 4.DOM元素的删除

如需删除元素和内容，一般可使用以下两个 jQuery 方法：

- `remove()` - 删除被选元素（及其子元素）
- `empty()` - 从被选元素中删除子元素

## H4 jQuery `remove()` 方法

jQuery `remove()` 方法删除被选元素及其子元素。

实例

```
$("#div1").remove();
```

[试一试](#)

## H4 jQuery `empty()` 方法

jQuery `empty()` 方法删除被选元素的子元素。

实例

```
$("#div1").empty();
```

[试一试](#)

## H4 过滤被删除的元素

jQuery `remove()` 方法也可接受一个参数，允许您对被删元素进行过滤。

该参数可以是任何 jQuery 选择器的语法。

下面的例子删除 class="italic" 的所有

元素：

实例

```
$("#p").remove(".italic");
```

[试一试](#)

## H3 5. 操作css

jQuery 拥有若干进行 CSS 操作的方法。我们将学习下面这些：

- addClass() - 向被选元素添加一个或多个类
- removeClass() - 从被选元素删除一个或多个类
- toggleClass() - 对被选元素进行添加/删除类的切换操作
- css() - 设置或返回样式属性

## H4 jQuery addClass() 方法

下面的例子展示如何向不同的元素添加 class 属性。当然，在添加类时，您也可以选取多个元素：


实例

```
$("#button").click(function(){  
    $("#h1,h2,p").addClass("blue");  
    $("#div").addClass("important");  
});
```

[试一试](#)

您也可以在 addClass() 方法中规定多个类：

## 实例

```

$("button").click(function(){
    $("#div1").addClass("important blue");
});
```

[试一试](#)

## H4 jQuery removeClass() 方法

下面的例子演示如何在不同的元素中删除指定的 class 属性：

## 实例

```

$("button").click(function(){
    $("h1,h2,p").removeClass("blue");
});
```

[试一试](#)

## H4 jQuery toggleClass() 方法

下面的例子将展示如何使用 jQuery toggleClass() 方法。该方法对被选元素进行添加/删除类的切换操作：

## 实例

```

$("button").click(function(){
    $("h1,h2,p").toggleClass("blue");
});
```

[试一试](#)

### H3 6.css()方法

#### H4 jQuery css() 方法

css() 方法设置或返回被选元素的一个或多个样式属性。

#### H4 返回 CSS 属性

如需返回指定的 CSS 属性的值，请使用如下语法：

```
css("propertyname");
```

下面的例子将返回首个匹配元素的 background-color 值：

实例

```
$("#p").css("background-color");
```

试一试

#### H4 设置 CSS 属性

如需设置指定的 CSS 属性，请使用如下语法：

```
css("propertyname", "value");
```

下面的例子将为所有匹配元素设置 background-color 值：

实例



```
$("#p").css("background-color", "yellow");
```

试一试

## H4 设置多个 CSS 属性

如需设置多个 CSS 属性，请使用如下语法：



```
css({"propertyname": "value", "propertyname": "value", ... });
```

下面的例子将为所有匹配元素设置 background-color 和 font-size：

实例



```
$("#p").css({"background-color": "yellow", "font-size": "200%"});
```

试一试

# 遍历

所谓遍历，与数组的遍历并不相同，此处的遍历指的是从dom元素开始搜索，查找父级、子级、或是同级寻找到自己想要操作的元素

## H3 1.向上遍历DOM树

## H4 1.parent()方法

获取被选元素的直接父元素，只会对上一级的DOM数进行遍历。

### 实例



```
$(document).ready(function(){  
    $("span").parent();  
});
```

## H4 2. parents()方法

获取被选元素的全部祖先元素，会一路向上直到根元素（）

### 实例



```
$(document).ready(function(){  
    $("span").parents();  
});
```

## H4 3. parentsUntil() 方法

获取两个给定元素之间的所有祖先元素。

下边的例子将会获取介于与

元素之间所有的祖先元素。

### 实例



```
$(document).ready(function(){  
    $("span").parentsUntil("div");  
});
```

## H3 向下遍历DOM树

向下遍历DOM树，查找指定元素的后代。

使用下列两个方法：

- children()
- find()

### H4 1. children() 方法


children() 方法返回所选元素的所有直接子元素。

该方法只对下一级的DOM树进行遍历。

下面的例子返回每个

元素的直接子元素。

#### 实例

```
  
$(document).ready(function(){  
    $("div").children();  
});
```

也可以传入参数对结果进行筛选。

下面的例子返回类名为 "1" 的所有

元素，并且它们是

的直接子元素：

#### 实例



```
$(document).ready(function(){  
  $("div").children("p.1");  
});
```

## H4 2.find() 方法

find() 方法返回所选元素的所有后代元素，一路向下直到最后。

下面的例子返回属于

后代的所有 元素：

### 实例



```
$(document).ready(function(){  
  $("div").find("span");  
});
```

下面的例子返回

的所有后代：

### 实例



```
$(document).ready(function(){  
  $("div").find("*");  
});
```

## H3 同级遍历DOM树

有许多有用的方法让我们在 DOM 树进行水平遍历：

- siblings()



- next()
- nextAll()
- nextUntil()
- prev()
- prevAll()
- prevUntil()

## H4 jQuery siblings() 方法

siblings() 方法返回被选元素的所有同胞元素。

下面的例子返回

的所有同胞元素：  
实例

```
$(DOCUMENT).READY(FUNCTION(){  
    $("H2").SIBLINGS();  
});
```

您也可以使用可选参数来过滤对同胞元素的搜索。  
下面的例子返回属于

# 的所有同胞元素 元素： 实例

```
$(DOCUMENT).READY(FUNCTION(){  
    $("H2").SIBLINGS("P");  
});
```

## H4 jQuery next() 方法

next() 方法返回被选元素的下一个同胞元素。

该方法只返回一个元素。

下面的例子返回

# 的下一个同胞元素： 实例

```
$(DOCUMENT).READY(FUNCTION(){  
    $("H2").NEXT();  
});
```

## H4 jQuery nextAll() 方法

nextAll() 方法返回被选元素的所有跟随的同胞元素。

下面的例子返回

的所有跟随的同胞元素：  
实例

```
$(DOCUMENT).READY(FUNCTION(){  
    $("H2").NEXTALL();  
});
```

## H4 jQuery nextUntil() 方法

nextUntil() 方法返回介于两个给定参数之间的所有跟随的同胞元素。

下面的例子返回介于

与

## H6 元素之间的所有同胞元素：

实例

```
$(document).ready(function(){  
    $("h2").nextUntil("h6");  
});
```

## H4 jQuery prev(), prevAll() & prevUntil() 方法

prev(), prevAll() 以及 prevUntil() 方法的工作方式与上面的方法类似，只不过方向相反而已：它们返回的是前面的同胞元素（在 DOM 树中沿着同胞元素向后遍历，而不是向前）。

## H3 过滤

三个最基本的过滤方法是：first(), last() 和 eq()，它们允许您基于其在一组元素中的位置来选择一个特定的元素。

其他过滤方法，比如 filter() 和 not() 允许您选取匹配或不匹配某项指定标准的元素。

## H4 jQuery first() 方法

first() 方法返回被选元素的首个元素。

下面的例子选取首个

元素内部的第一个

元素：

实例



```
$(document).ready(function(){  
    $("div p").first();  
});
```

## H4 jQuery last() 方法

last() 方法返回被选元素的最后一个元素。

下面的例子选择最后一个

元素中的最后一个

元素：

### 实例

```
$(document).ready(function(){  
    $("div p").last();  
});
```

## H4 jQuery eq() 方法

eq() 方法返回被选元素中带有指定索引号的元素。

索引号从 0 开始，因此首个元素的索引号是 0 而不是 1。下面的例子选取第二个

元素（索引号 1）：

### 实例

```
$(document).ready(function(){  
    $("p").eq(1);  
});
```

## H4 jQuery filter() 方法

filter() 方法允许您规定一个标准。不匹配这个标准的元素会被从集合中删除，匹配的元素会被返回。

下面的例子返回带有类名 "intro" 的所有

元素：

### 实例



```
$(document).ready(function(){  
    $("p").filter(".intro");  
});
```

## H4 jQuery not() 方法

not() 方法返回不匹配标准的所有元素。

**提示：** not() 方法与 filter() 相反。

下面的例子返回不带有类名 "intro" 的所有

元素：

**实例**



```
$(document).ready(function(){  
    $("p").not(".intro");  
});
```

---

# 5.JQUERY AJAX

## H3 1. jQuery load() 方法

jQuery load() 方法是简单但强大的 AJAX 方法。

load() 方法从服务器加载数据，并把返回的数据放入被选元素中。

**语法：**



```
$(selector).load(URL,data,callback);
```

必需的 `URL` 参数规定您希望加载的 URL。

可选的 `data` 参数规定与请求一同发送的查询字符串键/值对集合。

可选的 `callback` 参数是 `load()` 方法完成后所执行的函数名称。

这是示例文件 ("demo\_test.txt") 的内容：



```
<h2>jQuery and AJAX is FUN!!!</h2>
<p id="p1">This is some text in a paragraph.</p>
```

下面的例子会把文件 "demo\_test.txt" 的内容加载到指定的

元素中：

### 示例



```
$("#div1").load("demo_test.txt");
```

试一试

也可以把 jQuery 选择器添加到 URL 参数。

下面的例子把 "demo\_test.txt" 文件中 id="p1" 的元素的内容，加载到指定的

元素中：

### 实例



```
$("#div1").load("demo_test.txt #p1");
```

## 试一试

可选的 `callback` 参数规定当 `load()` 方法完成后所要允许的回调函数。回调函数可以设置不同的参数：

- `responseTxt` - 包含调用成功时的结果内容
- `statusTxt` - 包含调用的状态
- `xhr` - 包含 XMLHttpRequest 对象

下面的例子会在 `load()` 方法完成后显示一个提示框。如果 `load()` 方法已成功，则显示“外部内容加载成功！”，而如果失败，则显示错误消息：

### 实例

```
$("button").click(function(){
    $("#div1").load("demo_test.txt",function(responseTxt,statusTxt,xhr){
        if(statusTxt=="success")
            alert("外部内容加载成功！");
        if(statusTxt=="error")
            alert("Error: "+xhr.status+": "+xhr.statusText);
    });
});
```

## 试一试

## H3 2.HTTP 请求：GET vs. POST

两种在客户端和服务端进行请求-响应的常用方法是：GET 和 POST。

- `GET` - 从指定的资源请求数据
- `POST` - 向指定的资源提交要处理的数据

GET 基本上用于从服务器获得（取回）数据。注释：GET 方法可能返回缓存数据。

POST 也可用于从服务器获取数据。不过，POST 方法不会缓存数据，并且常用于连同请求一起发送数据。



如需学习更多有关 GET 和 POST 以及两方法差异的知识，请阅读我们的 [HTTP 方法 - GET 对比 POST](#)。

## H4 jQuery \$.get() 方法

\$.get() 方法通过 HTTP GET 请求从服务器上请求数据。

语法：

```
$.get(URL, callback);
```

必需的 `URL` 参数规定您希望请求的 URL。

可选的 `callback` 参数是请求成功后所执行的函数名。

下面的例子使用 \$.get() 方法从服务器上的一个文件中取回数据：

实例

```
$("#button").click(function(){
    $.get("demo_test.asp", function(data, status){
        alert("Data: " + data + "\nStatus: " + status);
    });
});
```

[试一试](#)

\$.get() 的第一个参数是我们希望请求的 URL ("demo\_test.asp") 。

第二个参数是回调函数。第一个回调参数存有被请求页面的内容，第二个回调参数存有请求的状态。

**提示：** 这个 ASP 文件 ("demo\_test.asp") 类似这样：



```
<%  
response.write("This is some text from an external ASP file.")  
%>
```

## H4 jQuery \$.post() 方法

\$.post() 方法通过 HTTP POST 请求从服务器上请求数据。

语法：



```
$.post(URL,data,callback);
```

必需的 **URL** 参数规定您希望请求的 URL。

可选的 **data** 参数规定连同请求发送的数据。

可选的 **callback** 参数是请求成功后所执行的函数名。

下面的例子使用 \$.post() 连同请求一起发送数据：

实例



```
$("button").click(function(){  
  $.post("demo_test_post.asp",  
  {  
    name:"Donald Duck",  
    city:"Duckburg"  
  },  
  function(data,status){  
    alert("Data: " + data + "\nStatus: " + status);  
  });  
});
```

试一试

`$.post()` 的第一个参数是我们希望请求的 URL ("demo\_test\_post.asp")。

然后我们连同请求 (name 和 city) 一起发送数据。

"demo\_test\_post.asp" 中的 ASP 脚本读取这些参数，对它们进行处理，然后返回结果。

第三个参数是回调函数。第一个回调参数存有被请求页面的内容，而第二个参数存有请求的状态。

**提示：** 这个 ASP 文件 ("demo\_test\_post.asp") 类似这样：

```
<%  
dim fname,city  
fname=Request.Form("name")  
city=Request.Form("city")  
Response.Write("Dear " & fname & ". ")  
Response.Write("Hope you live well in " & city & ".")  
%>
```

### H3 ajax ()

```
$.ajax({  
    url:'', //file url  
    method:'', //get or post  
    datatype:'', //file datatype  
    success:function(data){}, //获取成功的时候执行，data为获取到的数据  
    error:function(err){}, //获取失败时运行  
})
```