# jQuery

# jQuery概述

jQuery诞生于2005年，是JavaScript框架之一。它可以帮助用户使用很少的JavaScritpt代码，创建出漂亮的页面效果。jQuery设计的宗旨是“Write Less,Do More”，即倡导写更少的代码，做更多的事情。

## jQuery能做什么

1. 选择页面元素

使用JavaScript遍历DOM树，以及查找HTML文档结构中的某个元素，必须编写很多行代码。在jQuery中提供了可靠而富有效率的选择器，只需要一个简单的选择器字符串，即可准确获取需要检查或操作的文档元素。

1. 动态改变页面样式

使用JavaScript控制CSS受限于不同浏览器的兼容性处理，而jQuery可以弥补这一不足，它提供了跨浏览器的标准解决方案。而且即使在页面已经呈现之后，jQuery仍然能够改变文档中某个部分的类或者个别样式属性。

1. 动态改变页面内容

jQuery能改变文档的内容。使用少量的代码，即可改变页面内容，对HTML文档的整个结构都能重写或者扩展。使用起来远比JavaScript直接控制便捷。

1. 控制响应事件

jQuery提供了丰富的页面事件，这些事件使用简单，易记，易用，不需要考虑浏览器兼容性问题，但是如果使用JavaScript直接控制用户行为，需要考虑的问题就很多，既要考虑HTML文档结构与事件处理函数的合成，还要开率浏览器的不一致性。

1. 提供基本网页特效

jQuery内置了一批淡入、擦除、移动之类的效果，以及制作新效果的工具包，用户只需要简单地调用动画函数，就可以快速设计出高级动画效果。

1. 快速实现通行

jQuery对Ajax技术的支持很缜密，它通过消除这一过程中浏览器特定的复杂性，使用户得以专注于服务器端的功能设计。

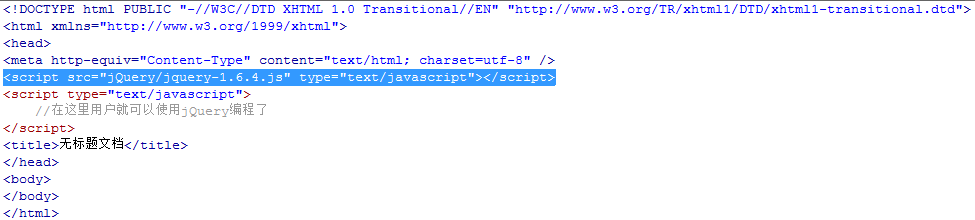
1. 扩展JavaScript内核

jQuery提供了对JavaScript核心功能的扩展，如迭代和数组操作等，增加对客户端、数据存储及JavaScript扩展的支持。

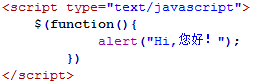
# 使用jQuery

jQuery项目主要包括jQuery Core（核心库）、jQuery UI（界面库）、Sizzle（CSS选择器）和Qunit（测试套件）4部分。

在项目中使用jQuery之前需在官网下载jQuery库文件，下载的库文件以.js后缀。然后将库文件放在项目中即可。当我们想要在某个页面上使用jQuery时，只需要在HTML文档头部简单地引用该库文件即可。



第一个jQuery程序



在jQuery库中，$是jQuery的别名，如$()等效于jQuery()。jQuery()函数是jQuery()库文件的接口函数，所有jQuery操作都必须从该接口函数切入。jQuery()函数相当于页面初始化事件处理函数，当页面加载完毕，会执行jQuery()函数包含的函数。

# jQuery框架核心功能

jQuery框架的核心就是从HTML文档中匹配元素并对其执行操作。

## 对外接口单一让使用更简单

jQuery把所有的操作都封装在一个jQeury()函数中，形成统一（也是唯一）的操作入口，这为jQuery操作降低了门槛。jQuery()构造函数能够接收任意类型的数据，但是能够解析的参数主要有以下4中类型。

1. jQuery(expression,context)

参数expression为一个表达式，该表达式可以是ID、DOM元素名、CSS表达式、XPath表达式等，jQuery将根据表达式匹配文档中的元素，然后把找到的元素包装到一个jQuery对象中返回。



在表达式字符串中，div#wrap表示id为wrap的div元素，然后在该元素中匹配子元素p，最后筛选出第一个p元素。”div#wrap>p:first”是CSS表达式，如果使用Xpath表达式表示，则应该为”div#wrap/p:first”，:first是一个伪类，表示其中的第一个。addClass()为jQuery对其用来添加CSS样式类的方法，相反操作的方法为removeClass()。

1. jQuery(html)

参数html表示一个HTML结构的字符串，此时jQuery将创建一个对应结构的HTML文档片段。



$(‘<li>new item</li>’)将其中的字符串转换为DOM对象，然后通过append()方法加入ul元素的最后。

1. jQuery(elements)

参数elements是一个DOM元素对象或者集合，此时jQuery将把DOM元素或集合中的DOM元素封装为jQuery对象



其中，jQuery构造函数把document对象封装为一个jQuery对象，然后调用ready()方法。Ready事件处理函数为document对象绑定一个事件，当页面初始化后，将ul的颜色设置为红色。

1. jQuery(fn)

参数fn是一个处理函数。$(document).ready()由于使用频繁，所以jQuery又使用$(fn)来代替，fn表示处理函数。



## 链式语法让编码更快速、优雅

jQuery的代码是非常优雅，也是非常灵巧的。它允许用户连续编写各种行为，从而实现按人的惯性思维进行快速开发，这种代码形式成为链式语法。

链式语法是一种比较时尚的编程方法，但是在使用该种方法时，为了方便阅读，读者应该注意以下几个问题。

1. 如果在同一个jQuery对象上执行不超过3个方法，则可以在同一行内书写。
2. 如果在同一个jQuery对象上执行很多操作，则应该分行进行书写，以方便阅读和修改。
3. 对于多个对象执行少量的操作，则可以为每一个对象书写一行代码。如果设计子元素操作，可以考虑使用缩进进行设计，这样就能够区分层次。
4. 如果对于多个对象执行很多连续的操作，则可以考虑结合上面几种方法同时进行设计。

## 模仿CSS选择器让选取元素更精确、灵活

jQuery选择器令DOM操作优雅而艺术，它支持ID、tagName、CSS1~3表达式、XPath表达式。jQuery不仅模仿CSS和XPath选择器的用法和功能，还自定义了很多过滤方法，综合利用这些选择器，可以随心所欲地选择HTML结构中的任意元素。

jQuery选择器按照功能主要分为选择和过滤，并允许配合使用，可以同时使用组合成一个选择器字符串。两者主要的区别是：

1. 过滤作用的选择器是指定条件从前面匹配的内容中筛选。过滤选择器也可以单独使用，此时表示从全部\*元素中筛选。

$(“:[title]”) 等同于 $(“\*:[title]”)

1. 选择功能的选择器则不会有默认的范围，因为作用是选择而不是过滤。

## 扩展接口让jQuery更开放、富有活力

jQuery定义了庞杂的公共函数和jQuery方法，但是也无法满足所有人的需求。jQuery努力实现了扩展性，在核心库里仅定义了基础的方法和函数，但是特意流出了使用jQuery易于扩展的方法和接口。

# 容易混淆的几个概念

* jQuery对象不等于DOM对象

DOM文档对象模型，是HTML与XML的应用程序接口，DOM将整个页面映射为一个由层次节点组成的文件。而jQuery对象是通过jQuery框架包装DOM对象之后产生的一个新的对象，本质上它是一个普通的Javascript对象，该对象中包含了DOM对象的集合，因此可以把DOM对象看作是一个独立的个体，而jQuery可以是多个DOM对象组成的数据集合。

jQuery框架为jQuery对象定义了独立使用的方法和属性，它无法直接调用DOM对象的方法。相反，DOM对象也无法直接调用jQuery对象的方法或属性。

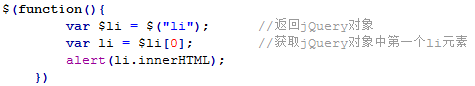
* jQuery对象与DOM对象之间的转换

jQuery对象和DOM对象是可以相互转换的，因为他们所操作的对象都是DOM元素，只不过jQuery对象包含了多个DOM元素。简单地说，jQuery对象是DOM元素的数据，也称为类数组，而DOM对象就是一个单个DOM元素

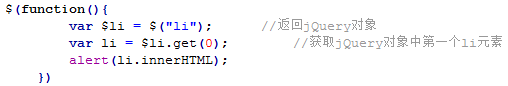
1. 把jQuery对象转换为DOM对象

jQuery对象不能够使用DOM对象的方法，但是如果需要，就应该先把jQuery对象转换为DOM对象。转换的方法有以下两种

1. 借助数组下标来读取jQuery对象集合中的某个DOM元素对象

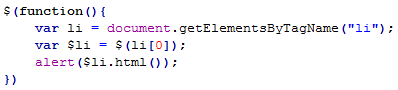


1. 借助jQuery对象方法，如get()方法，为get()方法传递一个下标值，即可从jQuery对象中取出一个DOM对象元素。

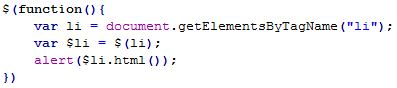


1. 把DOM对象转换为jQuery对象

对于DOM对象来说，直接把它传递给$()函数即可，jQuery对象会自动把它封装为jQuery对象，然后就可以自由调用jQuery定义的方法。



或者直接将DOM对象数组传递给jQuery对象



注意：使用document.getElementsByTagName()方法获取的DOM元素集合是一个真正的数组类型，而jQuery对象仅是一个类数组，不是真正意义上的数组类型数据。

* jQuery的ready不等于JavaScript的load

jQuery定义的ready事件与JavaScript定义的load事件都表示页面初始化行为，但是它们之间并非完全相同。

1. 执行时机不同

Load事件必须等到网页中所有内容全部加载完毕之后，才被执行。而jQuery的ready事件是在DOM结构绘制完毕之后就执行，也就是说它先于外部文件的加载就被执行了。这样就能够确保文档呈现和脚本初始化设置保持同步。

1. 用法不同

JavaScript的load事件只能够被编写一次。而对于jQuery的ready事件类型来说，可以在同一个文档中多次定义。

# 使用选择器

jQuery选择器是jQuery框架的基础，jQuery对事件的处理、DOM操作、CSS动态控制、Ajax通信、动画设计都是在选择器基础上进行的。jQuery选择器采用CSS和XPath选择符的能力，能够满足用户在DOM中会计而轻松地获取元素或元素组。

## 基本选择器

基本选择器主要包括ID选择器、标签选择器、类选择器、通配选择器和组选择器5中类型，这与CSS基本选择器类型相一致。

|  |  |  |
| --- | --- | --- |
| **选择器** | **说明** | **返回值** |
| #id | 根据给定的ID匹配一个元素。如果选择器中包含特殊字符，可以使用两个斜杠转义 | 单个元素 |
| element | 根据指定的元素类型名称选择该类型所有元素 | 同类型集合元素 |
| .class | 根据指定的类名选择所有同类元素 | 集合元素 |
| \* | 在所限定的范围内选择所有元素 | 所有元素的集合 |
| selector1,selector2,selectorN | 分别选择选择器组中每个选择器匹配的元素，然后合并返回所有元素 | 集合元素 |

1. ID选择器

JavaScript提供了原生方法getElementById()，实现在DOM中选择指定ID值的元素。jQuery简化了JavaScript原生方法的操作，通过一个简单的“#”标识前缀快速匹配指定ID的元素对象。

jQuery(“#id”);

1. 标签选择器

JavaScript提供了原生方法getElementsByTagName()，用来在DOM中选择指定类型的元素。而在jQuery中直接在jQuery()构造函数中指定标签名称既可以。

jQuery(“ul”);

1. 类选择器

JavaScript没有原生的类选择方法，此时使用jQuery的类选择器会更加便捷

jQuery(“.className”);

1. 通配符选择器

jQuery定义了通配符选择器，该选择器能够匹配指定上下文中所有元素。

jQuery(“\*”);

其实使用JavaScript的原生方法getElementsByTagName(“\*”)也可以实现同样的效果。

1. 组合选择器

jQuery支持CSS的分组选择器，通过这种方式可以扩大选择器的选择范围，同时增强jQuery选择器引擎的应用能力。

jQuery(“selector1,selector2,selectorN”);

## 层级选择器

层级选择器是通过DOM嵌套关系结构来实现准确匹配，层级选择器主要有包含选择器、子选择器、相邻选择器和兄弟选择器4种类型。

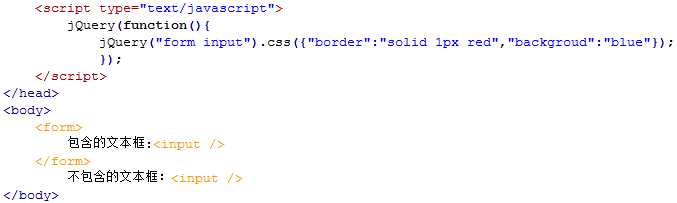
|  |  |
| --- | --- |
| 选择器 | 说明 |
| ancestor descendant | 在给定的祖先元素下匹配所有的后代元素。Ancestore表示任何有效选择器，descendant表示用以匹配元素的选择器，并且它是第一个选择器的后代元素。  例如：$(“form input”)可以匹配表单下所有的input元素 |
| parent > child | 在给定的父元素下匹配所有的子元素。Parent表示任何有效选择器，child表示用以匹配元素的选择器，并且它是第一个选择器的子元素。  例如：$(“form > input”)可以匹配表单下所有的子级input元素 |
| prev + next | 匹配所有紧接在prev元素后面的next元素。Prev表示任何有效选择器，next表示一个有效选择器并且紧接着第一个选择器  例如：$(“label + input”)可以匹配所有更在label后面的input元素 |
| prev ~ siblings | 匹配prev元素之后的所有siblings元素。Prev表示任何有效选择器，siblings表示一个选择器，并且它作为第一个选择器的同级结构。  例如：$(“form ~ input”)可以匹配所有与表单同级结构的input元素 |

1. 包含选择器

包含选择器就是在给定的祖先元素下匹配所有的后代元素

jQuery("ancestor descendant”);

参数ancestor和descendant为字符串，其中，ancestor表示包含元素，descendant表示被包含元素。



注意：包含选择不受DOM层级结构的限制，jQuery会返回所有包含选元素下的所有被包含元素

1. 子选择器

子选择器就是在匹配的父元素下选择所有匹配的直接子元素

jQuery(“parent > child”);

参数parent和child为字符串，其中parent为父元素,child为被包含的直接子元素，“>”为子选择器的标识符。



注意：子选择器受DOM层级结构的影响，只会返回父元素直接的子元素，不会返回嵌套的子元素。

1. 相邻选择器

相邻选择器就是在所有匹配的元素后选择同级的相邻元素

jQuery(“prev + next”);

参数prev和next为字符串，其中prev表示相邻的前一个元素，next表示相邻的后一个元素，“+”为相邻选择器的标识符。



注意：相邻选择器返回的是前一个元素DOM同级结构中紧挨着的下一个元素。

1. 兄弟选择器

兄弟选择器就是在所有匹配的元素后选择同级的所有元素。

jQuery(“prev ~ siblings”);

参数prev和siblings为字符串，其中prev表示相邻的前一个元素，siblings表示相邻的同级元素，“~”为兄弟选择器的标识符。



注意：与相邻选择器相比，兄弟选择返回的是与后一个元素相匹配并且与前一元素同级的所有元素。

## 简单伪类选择器

伪类可以看作是一种能被浏览器自动识别的特殊选择符。伪类选择器的最大语法特征就是在选择符中添加“：“标识符。

|  |  |
| --- | --- |
| 选择器 | 说明 |
| :first | 匹配找到的第一个元素。  例如：$(“tr:first”)表示匹配表格的第一行 |
| :last | 匹配找到的最后一个元素。  例如：$(“tr:last”)表示匹配表格的最后一行 |
| :not | 去除所有与给定选择器匹配的元素。  例如：$(“input:not(:cheched)”)可以匹配所有未选中的input元素 |
| :even | 匹配所有索引值为偶数的元素，从0开始计数。  例如：$(“tr:even”)可以匹配表格的1、3、5行（索引值0、2、4…..） |
| :odd | 匹配所有索引值为奇数的元素，从0开始计数。  例如：$(“tr:odd”)可以匹配表格的2、4、6行（索引值1、3、5…..） |
| :eq | 匹配一个给定索引值的元素，从0开始计数。  例如：$(“tr:eq(0)”)可以匹配第一行表格行 |
| :gt | 匹配所有大于给定索引值的元素，从0开始计数。  例如：$(“tr:gt(0)”)可以匹配第二行及其后面的所有行 |
| :lt | 匹配所有小于给定索引值的元素。  例如：$(“tr:lt(1)”)可以匹配第一行 |
| :header | 匹配如h1、h2、h3之类的标题元素 |
| :animated | 匹配所有正在执行动画效果的元素 |

简单伪类选择也被称为定位过滤器，因为它们主要根据编号和排位筛选特定位置上的元素或者过滤掉特定元素

1. 特定位置选择器

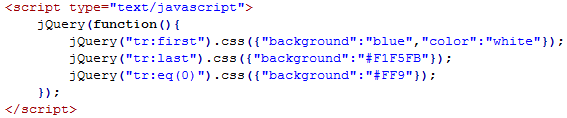
特定位置选择主要包含:first、:last和:eq(index)3种。:first能够在匹配的集合中选择第一个元素，:last能够在匹配的集合中选择最后一个元素，:eq(index)能够根据index索引值确定指定位置的元素。

jQuery(“selector:first”);

jQuery(“selector:last”);

jQuery(“selector:eq(indx)”);

参数selector、eq(index)为字符串，其中selector表示任意形式的选择器，“:“为伪类选择器的标识符。eq(index)虽然以字符串形式进行传递，但是eq()自身却是一个匹配函数，该函数接收一个index索引值，这个索引值从0开始计数，与数组下标位置一一对应。



注意：特定位置选择器是针对伪类分隔符前面的选择器元素所匹配的结果基础进行第2次过滤，如果没有指定匹配的范围，则将视为是整个文档范围。

1. 指定范围选择器

特定范围选择器主要包括:even、:odd、:gt(index)和:lt(index)4种。:even能都在匹配的集合中选择所有偶数行元素，:odd能够在匹配的集合中选择所有奇数行元素，:gt(index)能够在匹配的集合中选择大于给定索引值的元素，:lt(index)能够在匹配的集合中选择小于给定索引值的元素

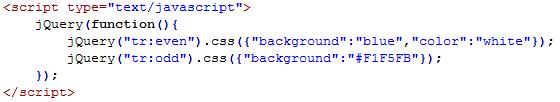
jQuery(“selector:even”);

jQuery(“selector:odd”);

jQuery(“selector:gt(index)”);

jQuery(“selector:lt(index)”);

参数selector、even、odd为字符串，其中selector表示任意形式的选择器，even表示在匹配元素中选择偶数行元素，odd表示在匹配元素中选择奇数行元素，匹配元素集合第一个元素下标从0开始计数。“：“为伪类选择器的标识符。index表示数字索引值，从0开始。



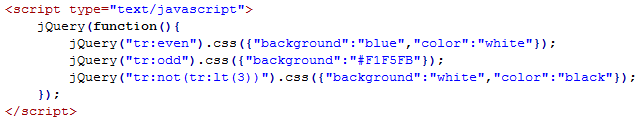
注意：:get(index)和:lt(index)选择器能够匹配连续的多个元素，而:even和:odd仅能够匹配非连续的多个元素。这些选择器在表格样式和列表样式设计中应用价值较大

1. 排除选择器

:not选择器比较特殊，它能够在匹配元素集合中排除符合特定匹配规则的元素，也就是说它可以反向方式快速过滤掉不需要的元素。

jQuery(“selector1:not(selector2)”);

参数selector1、selector2和not为字符串，其中selector1和selector2表示任意形式的选择器，“：“为伪类选择器的标识符，not表示配出函数标识符。



1. 特殊选择器

:animated和:header选择器是两个比较特殊的选择器，他们分别用来匹配动画元素和标题元素。

jQuery(“selector:animated”);

jQuery(“selector:header”);

参数selector、animated和header为字符串。其中selector表示任意形式的选择器，“：为伪类选择器的标识符，animated表示正在执行动画效果的元素，header表示标题元素，如h1、h2、h3等。

## 与内容相关的伪类选择器

内容伪类选择器主要是根据元素包含内容作为筛选条件进行匹配，这类选择器主要包括4种

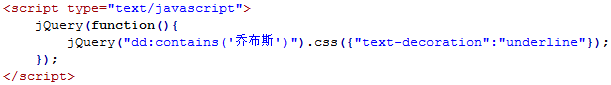
|  |  |
| --- | --- |
| 选择器 | 说明 |
| :contains | 匹配包含给定文本的元素。  例如：$(“div:contains(‘图片’)”)匹配所有包含“图片“的div元素 |
| :empty | 匹配所有不包含子元素或者文本的空元素 |
| :has | 匹配含有选择所匹配的元素的元素。  例如：$(“div:has(p)”)匹配所有包含p元素的div元素 |
| :parent | 匹配含有子元素或者文本的元素 |

1. 匹配包含文本选择器

:contains(text)选择器是一个比较实用的针对网页文本过滤的选择器，它能够根据指定的文本在所能够匹配的元素集合中搜索含特定关键字的元素

jQuery(“selector:contains(text)”);

参数selector、contains、text为字符串。其中selector表示任意形式的选择器，“：“为伪类选择器的标识符，contains()表示文本匹配函数，text表示一个用以查找的字符串。



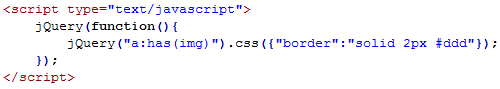
注意：contains()过滤函数中必须使用单引号包含文本关键字。

1. 匹配包含元素选择器

:has(selector)选择器与:contains(text)选择器使用类似，但是:has(selector)主要搜索匹配元素所包含的元素进行过滤。

jQuery(“selector1:has(selector2)”);

参数selector1、selector2、has为字符串。其中，selector1和selecotr2表示任意形式的选择器，“：“为伪类选择器的标识符，has()表示包含元素匹配函数。



1. 包含判断选择器

:empty和:parent选择器比较特殊，他们专门用来检测匹配元素是否包含内容。其中:empty选择器过滤出匹配元素中包含空内容的元素，而:parent选择器过滤出匹配元素中包含非空内容的元素。

jQuery(“selector:empty”);

jQuery(“selector:parent”);

参数selector、empty、parent为字符串。其中selector表示任意形式的选择器，“：“为伪类选择器的标识符，empty表示包含空内容的匹配元素，parent表示包含内容的匹配元素。

## 与元素显示状态相关的伪类选择器

jQuery为了方便用户铺货显示或者隐藏元素，专门提供了两个显示隐藏伪类选择器。他们能够根据元素的可见或者隐藏进行快速过滤

|  |  |
| --- | --- |
| 选择器 | 说明 |
| :hidden | 匹配所有不可见元素，或者type为hidden的元素 |
| :visible | 匹配所有的可见元素 |

## 匹配子元素的伪类选择器

子元素伪类选择器就是通过匹配元素选择该元素包含的特定子元素。

|  |  |
| --- | --- |
| 选择器 | 说明 |
| :nth-child | 匹配其父元素下的第N个字或奇偶元素 |
| :first-child | 匹配第一个子元素  :first选择器只匹配一个元素，而:first-child选择符将为每个父元素匹配一个子元素 |
| :last-child | 匹配最后一个子元素  :last只匹配一个元素，而:last-child选择符将为每个父元素匹配一个子元素 |
| :only-child | 如果某个元素是父元素中唯一的子元素，那将会被匹配；如果父元素中含有其他元素，那将不会被匹配 |

注意：后面将介绍到:eq(index)选择器，该选择器只能够匹配一个元素，而:nth-child能够为每一个父元素匹配子元素。:nth-child是从1开始的，而:eq()是从0算起的。

## 与表单对象相关的伪类选择器

|  |  |
| --- | --- |
| 选择器 | 说明 |
| :input | 匹配所有input、textarea、select和button元素 |
| :text | 匹配所有的单行文本框 |
| :password | 匹配所有密码框 |
| :radio | 匹配所有单选按钮 |
| :checkbox | 匹配所有复选框 |
| :submit | 匹配所有提交按钮 |
| :image | 匹配所有图像域 |
| :reset | 匹配所有重置按钮 |
| :button | 匹配所有按钮 |
| :file | 匹配所有文件域 |
| :hidden | 匹配所有不可见元素，或者type为hidden的元素 |

## 与表单属性相关的伪类选择器

|  |  |
| --- | --- |
| 选择器 | 说明 |
| :enabled | 匹配所有可用元素 |
| :disabled | 匹配所有不可用元素 |
| :checked | 匹配所有选中的被选中元素（复选框、单选框，不包括select中的option） |
| :selected | 匹配所有选中的option元素 |

## 属性选择器

属性选择器就是根据元素的属性及其值作为过滤条件，来匹配对应的DOM元素。属性选择一般都是以中括号作为起止分界符。

|  |  |
| --- | --- |
| 选择器 | 说明 |
| [attribute] | 匹配包含给定属性的元素  例如：$(“div[id]”)表示查找所有含有id属性的div元素 |
| [attribute=value] | 匹配属性等于特定值的元素。属性值得引号在大多数情况下是可选的，如果属性值中包含“]“时，需要加引号用以避免冲突  例如：$(“input[name=’text’]”)表示查找所有name属性值为text的input元素 |
| [attribute!=value] | 匹配所有不含有指定的属性，或者属性不等于特定值的元素。该选择器等价于:not([attr=value])  要匹配含有特定属性单不等于特定值的元素，可以使用[attr]:not([attr=value])  例如：$(“input[name!=’text’]”)表述查找所有name属性但值不是text的input元素 |
| [attribute^=value] | 匹配给定的属性是以某些值开始的元素  例如：$(“input[name^=’text’]”)表示所有name属性值以text开始的input元素 |
| [attribute$=value] | 匹配给定的数值是以某些值结尾的元素  例如：$(“input[name$=’text’]”)表示所有name属性值以text结束的input元素 |
| [attribute\*=value] | 匹配给定的属性是以包含某些值的元素  例如：$(“input[name\*=’text’]”)包含所有name属性值包含text字符串的input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  例如：$(“input[name\*=’text’] [id]”)表示所有name属性值包含text字符串，且包含了id属性的input元素 |

模糊匹配属性值选择器是一类特殊的属性选择器，类似于正则表达式的匹配模式，也是属性选择器中功能最强大的一部分，主要包括以下几种匹配模式

1. [!=]（非匹配）：匹配不等于特定属性值
2. [^=]（前缀匹配）：匹配属性值中的起始字符
3. [$=]（后缀匹配）：匹配属性值中的结束字符
4. [\*=]（子字符串匹配）：匹配属性值存在的指定字符

## jQuery选择器应用优化

正确使用选择器引擎对于页面性能起了至关重要的作用。使用合适的选择器表达式可以提高性能、增强语义并简化逻辑。在传统用法中，最常用的简单选择器包括ID选择器、Class选择器、类型标签选择器。其中ID选择器是速度最快的，其次是类型选择器(因为都是使用了JavaScript内置函数)，速度最慢的是Class选择器（需要通过解析HTML文档树，并需要在浏览器内核外递归）。

优化建议：

1. 多使用ID选择器
2. 少直接使用Class选择器，可以用复合选择器代替
3. 摒除表达式中的冗余部分，对于不必要的复合表达式应该进行简化。
4. 多用父子关系，少用嵌套关系
5. 缓存jQuery对象

缓存jQuery对象可以提高系统性能。



# 使用过滤器

jQuery提供了两种选择文档元素的方法：选择器和过滤器。选择器主要是模仿CSS和XPath语法，提供高效、准确匹配元素的一般方法和用法，而过滤器是建立在选择器基础上进行的二次筛选。选择器是符合一定规律的字符串组合，而过滤器是一系列简单、使用的jQuery方法。在jQuery框架中，过滤器通过Sizzle.filter子类来实现，它包含过滤、查找和串联3类操作行为。

## 过滤

过滤是现有的jQuery对象所包含元素进行再筛选的操作。

|  |  |
| --- | --- |
| 过滤方法 | 说明 |
| eq(index) | 获取第N个元素 |
| hasClass(class) | 检查当前的元素是否含有某个特定的类，如果有，则返回ture |
| filter(expr) | 筛选出于指定表达式匹配的元素集合 |
| filter(fn) | 筛选出与指定函数返回值匹配的元素集合 |
| is(expr) | 用一个表达式来检查当前选择的元素集合，如果其中至少有一个元素符合这个给定的表达式就返回true |
| map(callback) | 将一组元素转换成其他数组（不论是否是元素数组） |
| has(expr) | 保留包含特定后代的元素，，去掉那些不含有指定后代的元素 |
| not(expr) | 删除与指定表达式匹配的元素 |
| slice(start,[end]) | 选取一个匹配的子集 |

1. 类过滤

类过滤就是根据元素的类属性来进行过滤操作，jQuery通过hasClass()方法来实现。

hasClass(className)

参数className是一个字符串，表示元素的类名。该方法适合执行判断操作，判断当前jQuery对象中的某个元素是否包含了指定类名，如果包含则返回true，否则返回false。但是该方法无法过滤包含特定类名的元素。



1. 下标过滤

类过滤还仅是一个条件检测，无法真正过滤出符合指定类名的元素，但是使用下标过滤就可以精确选取出jQuery对象中指定下标的元素

eq(index)

参数index是一个整数值，从0开始，用来指定元素在jQuery对象中的下标位置。



1. 表达式过滤

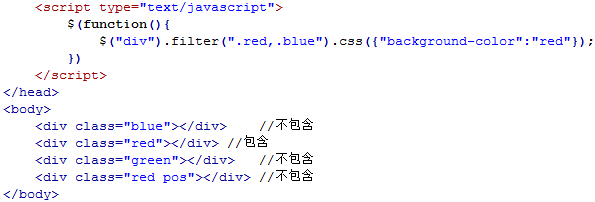
表达式过滤是最强大的过滤工具，因为表达式具有较大的灵活性，用户可以根据需要自定义表达式的形式，只要表达式符合jQuery选择器语法形式即可。

1. filter()方法

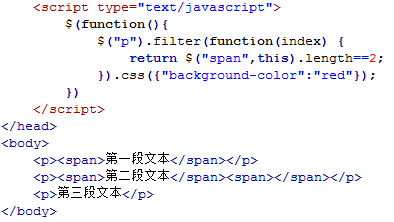
filter()方法时功能最强大的表达式过滤器，同时也可以接收函数参数，并根据函数的返回值来确定要过滤的元素

filter(expr)

该方法可以同时带多个表达式，表达式之间用逗号分隔。



filter(fn)



其中$(“span”,this)将匹配当前元素内部的所有span元素，然后计算它们的长度。filter()方法将根据参数返回值决定是否保留每个匹配元素。

在这个参数函数中包含一个index参数（默认的），该参数存储当前对象在jQuery对象中的下标位置，在函数体内，this关键字指向当前元素对象，而不是jQuery对象。

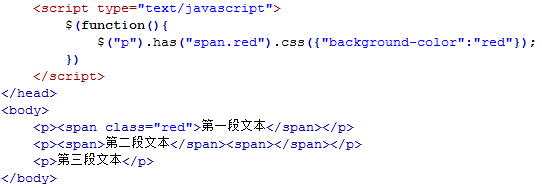
1. has()方法

has()方法也是一个过滤工具，不过它没有filter()方法功能强大，但是使用敏捷，更方便用户把它作为专业过滤工具，专门负责保留包含特定后代的元素，去掉那些包含有指定后代的元素。

has()方法将会从给定的jQuery对象中重新创建一组匹配的对象

has(expr)

参数expr可以是一个jQuery选择器表达式字符串，也可以是一个元素或者一组元素。提供的选择器会一一测试原先那些对象的后代，含有匹配后代的对象将得以保留。也就是说，如果元素包含了与expr表达式相匹配的子元素，则将保留该元素，否则就会删除该元素。



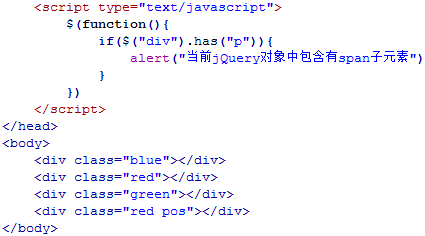
1. 判断

表达式判断的方法不直接过滤元素，仅作为一个检测工具为其他代码判断当前jQuery对象是否包含特定符合条件的元素提供方便

is(expr)

参数expr为一个jQuery选择器表达式，用来筛选符合特定条件的元素

工作原理：用一个表达式来检测当前选择的元素集合，如果其中至少有一个元素符合这个给定的表达式就返回ture。如果没有元素符合或者表达式无效，则返回false。



1. 映射

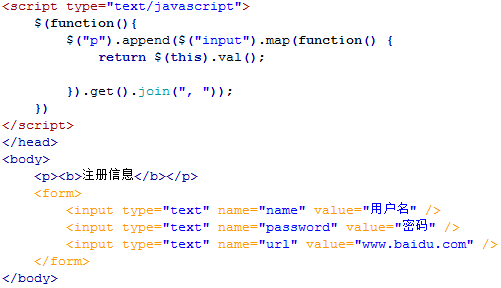
通常情况下，映射是指两个元素几个之间元素相互“对应”的关系，实际上映射是一种间接引用。map()方法通过映射关系，把jQuery对象中每个元素映射到一个数组中，也就是将一组元素转换成其他数组（不论是都是元素数组）

map(callback)

参数callback表示一个回调函数，将给每个元素执行的函数。用户可以用这个函数来建立一个列表，不论是值、属性还是CSS样式，或者其他特别形式，都可以用$.map()方法建立。

实例：

通过map()方法把所有匹配的input元素的value属性值映射为一个新集合，然后调用get()方法把集合内的数据转换为数组，再调用数组的join()方法把几个元素连接为字符串，最后调用jQuery的append()方法把这个字符串附加到<p>标签中的末尾。





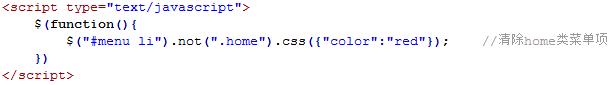
map()方法根据调用的jQuery作为映射原对象，返回的结果也是一个jQuery对象，不过映射后的jQuery对象并非都是元素集合，也可能是其他任意类型的数据，这主要根据map()的参数函数返回值来确定。

1. 清洗

与eq()、filter()、has()方法的过滤器思路不同，not()方法执行的是反向操作，它能够从jQuery对象中删除符合条件的元素，并返回这个清洗后的jQuery。

not(expr)

参数expr表示一个jQuery选择器表达式字符串，当然也可以是一个元素或者多个元素。

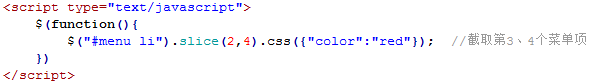


1. 截取

jQuery模仿Array对象的slice()方法也定义了一个slice()方法，用来在当前jQuery对象中截取部分元素，并把这个被截取的元素集合装在一个新的jQuery对象中返回。

slice(start,[end])

参数start和end都是一个整数，其中start表示开始选取子集的位置，第一个元素是0，如果该参数为负数，则表示从集合的尾部开始选起。end是一个可选参数，表示结束选取的位置，如果不指定，则表示到集合的结尾，但是被截取的元素中不包含end所指定位置的元素。



## 查找

查找主要是以当前jQuery对象为基础，查找父级、同级或者下级相关元素，以便实现延伸筛选，从而增强对文档的控制能力。

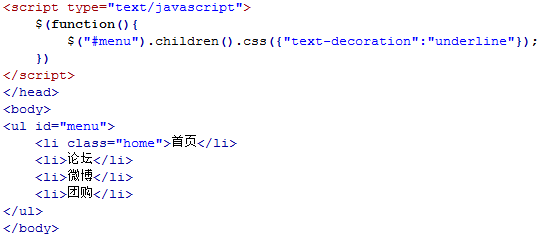
|  |  |
| --- | --- |
| 查找方法 | 说明 |
| add(expr,[context]) | 把与表达式匹配的元素添加到jQuery对象中 |
| children([expr]) | 取得一个包含匹配的元素集合中每一个元素的所有子元素的元素集合 |
| closest(expr,[context]) | 从元素本身开始，逐级向上级元素匹配，并返回最先匹配的元素 |
| contents() | 查找匹配元素内部所有的子节点（包括文本节点） |
| find(expr) | 搜索所有与指定表达式匹配的元素 |
| next([expr]) | 取得一个包含匹配的元素集合中每一个元素紧邻的后面同辈元素的元素集合 |
| nextAll([expr]) | 查找当前元素之后所有的同辈元素 |
| nextUntil([selector]) | 查找当前元素之后所有的同辈元素，直到遇到匹配的那个元素位置 |
| offsetParent() | 返回第一个匹配元素用于定位的父节点 |
| parent([expr]) | 取得一个包含所有匹配元素的唯一父元素的元素集合 |
| parents([expr]) | 取得一个包含所有匹配元素的祖先元素的元素几个（不包含根元素） |
| parentsUntil([selector]) | 查找当前元素的所有的父辈元素，知道遇到匹配的那个元素为止 |
| prev([expr]) | 取得一个包含匹配的元素集合中每一个元素紧邻的前一个同辈元素的元素集合 |
| prevAll([expr]) | 查找当前元素之前所有的同辈元素 |
| prevUntl([selector]) | 查找当前元素之前所有的同辈元素，直到遇到匹配的那个元素为止 |
| siblings([expr]) | 取得一个包含匹配的元素集合中每一个元素的所有唯一同辈元素的元素集合 |

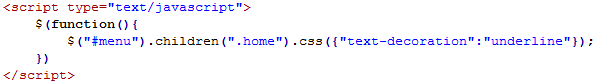
1. 向下查找后代元素
2. children()方法

children()方法能够查找当前元素的所哟或者部分子元素，实际上它是childNodes集合的另一种用法

children([expr])

参数expr表示jQuery选择器表达式字符串，用以过滤子元素。该参数为可选，如果省略，则将匹配所有的子元素，功能类似于DOM的childNodes，不过返回值形式不同。children()方法返回的是一个jQuery，而childNodes返回的是一个数组。





1. contents()方法

与children()方法相似，contents()方法也是用来查子内容的，但是它不仅获取子元素，还可以获取文本节点、注释节点等，

contents()

该方法没有参数，功能等同于DOM的childNodes，不过返回的值形式不同。contents()方法返回的是一个jQuery对象，而childNodes返回的是一个数组。

1. find()方法

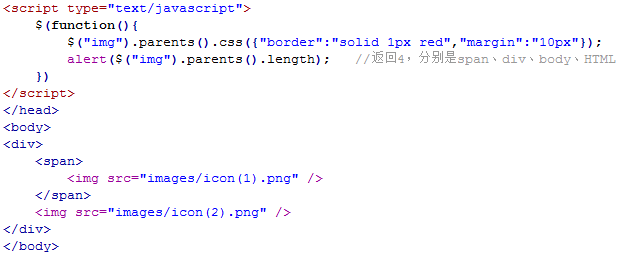
find()方法与children()方法相似，都是用来向下查找元素的，但是find()方法能够查找所有后代元素，而children()方法仅能够查找子元素。find()查找仅限于元素。

1. 向上查找祖先元素
2. parents()方法

parents()方法能够查找所有匹配元素的祖先元素

parents([expr])

参数expr表示jQuery选择器表达式字符串，用以过滤祖先元素。该参数为可选，如果省略，则将匹配所有元素的祖先元素。



parents()方法将查找所有匹配元素的祖先元素，如果存在重合的祖先元素，则仅记录一条。当然也可以在oarents()参数中定义一个过滤表达式，过滤出符合条件的祖先元素。

1. parent()方法

parent()方法时对parents()方法的延伸，它可以取得一个包含所有匹配元素的直接唯一父元素的元素集合。

parent([expr])

参数expr表示jQuery选择器表达式字符串，用以过滤父元素。该参数可选，如果省略，则将匹配所有元素的直接唯一父元素



1. parentsUntil()方法

parentsUntil()方法是对parents()方法的一个有益补充，它可以查找指定范围的所有祖先元素，相当于在parents()方法返回集合中截取部分祖先元素

parentsUntil([selector])

参数selector表示jQuery选择器表达式字符串，用以确定范围的祖先元素，该参数为可选，如果省略，则将匹配所有祖先元素。



1. offsetParent()方法

offsetParent()方法能够查找到当前元素的第一个定位祖先元素，使用该方法可以快速找到当前元素的定位框，并据此设置元素的绝对位置

offsetParent()

该方法没有参数，使用该方法可以返回包含祖先元素中第一个定位元素的jQuery对象（定位元素就是元素的position属性被设置为relative或者absolute的元素）。offsetParent()方法仅对可见元素有效。

1. closest()方法

closest()方法用来查找指定的父元素。它主要为事件处理而设计，处理事件委派非常有用。

closest(expr,[context])

参数expr可以是字符串，也可以是数组，用以过滤元素的表达式。也可以传递一个字符串数组，用于查找多个元素。context是一个可选参数，表示一个元素，用来设置待查找的DOM元素集、文档或jQuery对象。如果省略，则表示待查找所有祖先元素。

closest()方法被解析时，会首先检查当前元素是否匹配，如果匹配则直接返回元素本身。如果不匹配则向上查找父元素，一层一层往上，直到找到匹配选择器的元素。如果什么都没有找到则返回一个空的jQuery对象

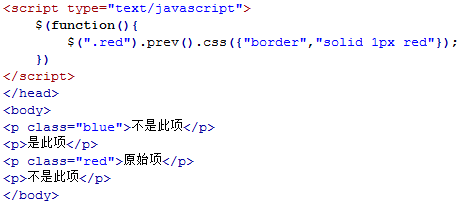
closest()方法与parents()方法的区别：

1. closest()方法从当前元素开始匹配寻找，而parents()方法从父元素开始匹配寻找
2. closest()方法逐级向上查找，知道发现匹配的元素后就停止了，而parents()方法一直向上查找直到根元素，然后把这些元素放进一个临时集合中，再用给定的选择器表达式过滤。
3. closest()方法返回0或1个元素，而parents()方法可能返回0个、1个，或者多个元素。
4. 向前查找兄弟元素
5. prev()方法

prev()方法用于匹配上一个相邻的同辈元素

prev([expr])

参数expr表示jQuery选择器表达式字符串，用以过滤匹配元素。该参数为可选，如果省略，则将匹配所有上一个相邻的元素。

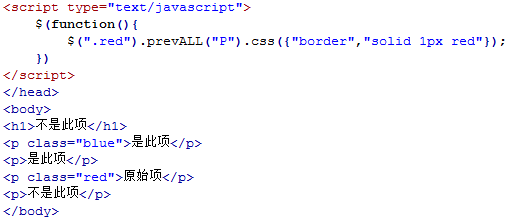


1. prevAll()方法

prevAll()方法比prev()方法查找的范围要大，不仅包括相邻的兄弟元素，还包括所有同辈元素。

prevAll([expr])

参数expr表示jQuery选择器表达式字符串，用以过滤匹配元素。该参数为可选，如果省略，则将匹配所有上面同辈元素

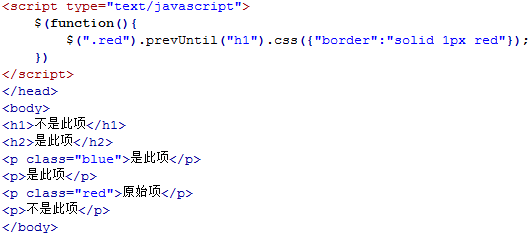


1. prevUntil()方法

prevUntil()方法介于prevAll()方法和prev()方法之间。prevUntil()方法选取指定范围的相邻同辈元素。

prevUntil([selector])

参数selector表示jQeury选择器表达式字符串，用以过滤匹配元素。该参数为可选，如果省略，则将匹配所有上面同辈元素。



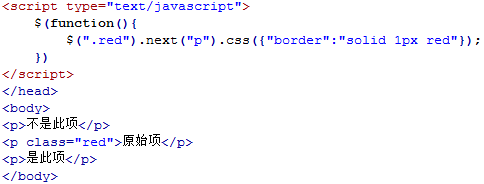
查找范围为给定两个条件之间的所有同辈元素

1. 向后查找兄弟元素
2. next()方法

next()方法用于匹配下一个相邻的同辈元素。

next([expr])

参数expr表示jQuery选择器表达式字符串，用以过滤匹配元素。该参数为可选，如果省略，则将匹配所有下一个相邻的元素

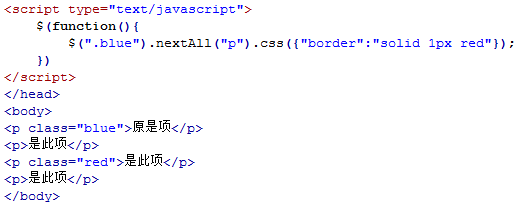


1. nextAll()方法

nextAll()方法比next()方法查找的范围要大，不仅包括相邻的兄弟元素，还包括所有同辈元素。

nextAll([expr])

参数expr表示jQuery选择器表达式字符串，用以过滤匹配元素。该参数为可选，如果省略，则将匹配所有下面的同辈元素

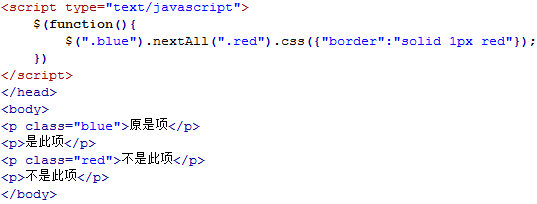


1. nextUntil()方法

nextUntil()方法介于nextAll()方法和next()方法之间。nextUntil()方法选取指定范围的相邻同辈元素

nextUntil([selector])

参数selector表示jQeury选择器表达式字符串，用以过滤匹配元素。该参数为可选，如果省略，则将匹配所有下面同辈元素。



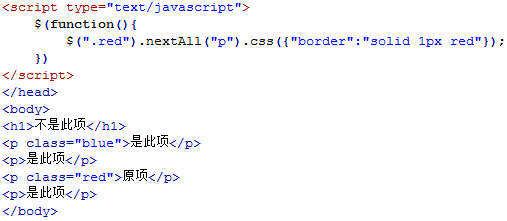
查找范围为给定两个条件之间的所有同辈元素

1. 查找兄弟元素
2. siblings([expr])

siblings()方法用于查找所有兄弟元素，不管其位置在前还是在后。

siblings([expr])

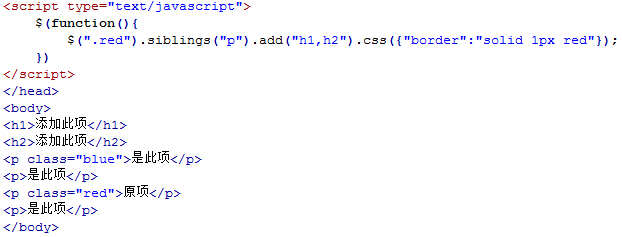
参数expr表示jQuery选择器表达式字符串，用以过滤匹配元素。该参数为可选，如果省略，则将匹配所有同辈兄弟元素



1. 添加查找对象
2. add()方法用于将指定的元素添加到jQuery对象中。

add(expr,[context])

1. 参数expr表示jQuery选择器表达式字符串，用于匹配元素并添加表达式字符串，或者用于动态生成的HTML代码，如果是一个字符串数组则返回多个元素。
2. context为可选参数，可以是待查的DOM元素集、文档或jQuery对象。



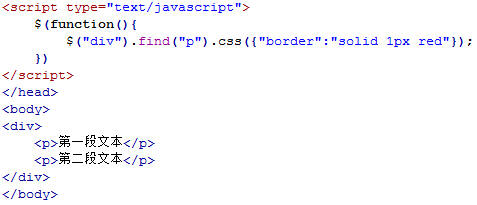
## 串联

jQuery语法的最大亮点就是链式语法，它能够实现在一行代码中完成各种复杂的任务。这种连续书写的代码显得灵巧、优雅，因此称之为链式语法。不过这种语法有时候会感觉不方便，因此很多方法通过点运算符串联在一次，操作的jQeury对象不能够前后照应，但是很多情况下我们希望用不同的方法操作不同的jQuery对象，或者前后方法能够相互影响，为此jQueyr提供了两个串联专用工具。

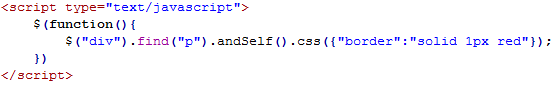
1. 绑定前后jQuery对象

andSelf()方法

例句：



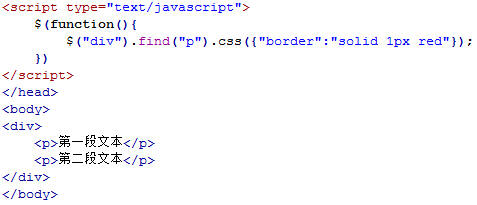
如果想为div同时也设置样式，则需要单独写一段代码来实现。这样就与jQuery的链式语法设置原则相违背了。此时就要用到串联工具andSelf()方法将$(“div”)和find(“p”)两个不同的jQuery对象链接在一起，最后再为他们定义统一的样式



1. 返回前一个jQuery对象

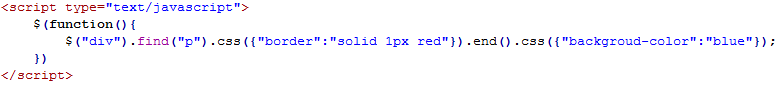
end()方法

例句：



如果为<p>标签定义了边框样式，然后再为<div>标签定义背景颜色，简单的方法时重新换一行为<div>标签定义样式。

不过利用jQuery定义的end()方法，可以保持在一行内完成两行任务，即调用find(“p”).css()后，再调用end()方法返回$(“div”)方法匹配的jQuery对象，而不是find()方法所查找的jQuery



首先为$(“div”).find(“p”)定义的jQuery所包含的元素定义边框样式，然后调用end()方法返回上一次匹配的jQuery对象，即$(“div”)定义的jQuery对象，再为该对象调用css()方法定义背景样式。

# DOM操作

## 创建节点

节点是DOM中有效而完整的结构中的最小单元，包括元素、属性、文本、文档、注释等。在实际开发中，要创建动态内容，主要操作的节点包括元素、属性和文本。

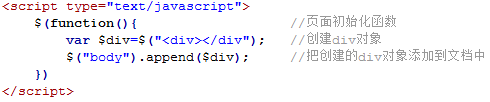
1. 创建元素

在jQuery中，直接使用jQuery构造函数$()创建元素对象。

$(html)

该函数能够根据参数html所传递的HTML标记字符串，创建一个DOM对象，并将该对象包装在jQeury对象返回。

创建好元素对象后，需要将元素插入到文档中。此时可以使用jQuery的append()方法把创建的元素添加到文档中去。

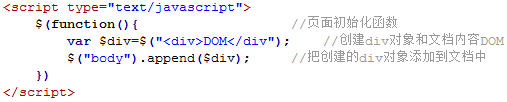


虽然jQuery创建元素代码比较简单，但是从执行效率角度分析，JavaScript要比jQuery快很多。

1. 输入文本

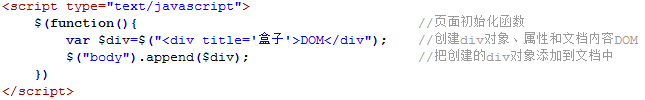
文本节点无法独立存在，必须包裹在元素节点内部，因此在创建文本之前，应确保新建或者选择元素节点。

jQuery在创建文本节点比较简单，直接把文本字符串添加到元素标记字符串中，然后使用append()等方法把它添加到DOM文档结构树中



1. 设置属性

jQuery创建属性节点与创建文本节点类似，简单而又方便。



## 插入内容

jQuery提供了众多在文档中插入内容的方法，这些方法从不同角度和用途进行设计，极大地方便了用户操作。

1. 内部插入

jQuery定义了4种方法用来在节点内部插入内容

|  |  |
| --- | --- |
| 方法 | 说明 |
| append() | 向每个匹配的元素内部追加内容 |
| appendTo() | 把所有匹配的元素追加到另一个指定的元素集合中。实际上，该方法颠倒了append()的方法。  例如：$(A).append(B)与$(B).appendTo(A)是等价的 |
| prepend() | 向每个匹配的元素内部前置内容 |
| prependTo() | 把所有匹配的元素前置到另一个指定的元素集合中。实际上，该方法颠倒了prepend()方法。  例如：$(A).prepend(B)与$(B).prependTo(A)是等价的 |

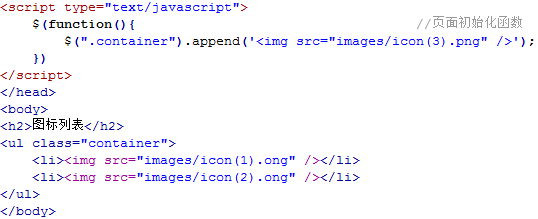
1. append()方法

append()方法能够把参数指定的内容插入到指定的节点中，并返回一个jQuery对象。指定的内容被插入到每个匹配元素里面的最后面，作为它的最后一个子元素。

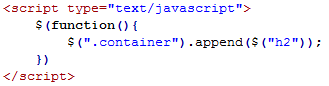
append(content)

append(function(index,html))

参数content可以是一个元素、HTML字符串或jQuery对象，用来插在每个匹配元素里的末尾。参数function(index,html)是一个返回HTML字符串的函数，该字符串用来插入到匹配元素的末尾。



append()方法不仅接收HTML字符串，还可以接受jQuery对象和DOM对象。如果把jQuery对象追加到当前元素尾部，则将阐述原来位置的jQuery匹配对象，此操作相当于移动，而不是复制。



1. appendTo()方法

appendTo()方法将匹配的元素插入到目标元素的最后面（里面的后面）

appendTo(target)

参数target表示一个选择符、元素、HTML字符串或者jQuery对象；符合的元素会被插入到由参数指定的目标末尾



可以改写为



1. prepend()方法

prepend()方法能够把参数指定的内容插入到指定的节点中，并返回一个jQuery对象。指定的内容被插入到每个匹配元素里面的最前面，作为它的第一个子元素。

prepend(content)

prepend(function(index,html))

参数content可以是一个元素、HTML字符串或jQuery对象，用来插在每个匹配元素里的最前面。参数function(index,html)是一个返回HTML字符串的函数，该字符串用来插入到匹配元素的最前面。



1. prependTo()方法

prependTo()方法将匹配的元素插入到目标元素的最前面（里面的前面）

prependTo (target)

参数target表示一个选择符、元素、HTML字符串或者jQuery对象；符合的元素会被插入到由参数指定的目标最前面。



可以改写为



1. 外部插入

|  |  |
| --- | --- |
| 方法 | 说明 |
| after() | 在每个匹配的元素之后插入内容 |
| before() | 在每个匹配的元素之前插入内容 |
| insertAfter() | 把所有匹配的元素插入到另一个指定的元素集合的后面 |
| insertBefore() | 把所有匹配的元素插入到另一个指定的元素集合的前面 |

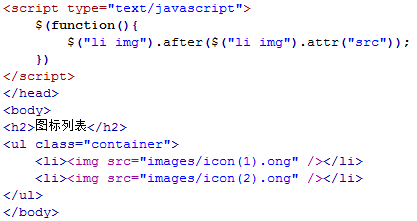
1. after()方法

after()方法能够根据参数设定在每一个匹配的元素之后插入内容。

after(content)

after(function(index))

参数content表示一个元素、HTML字符串或jQuery对象，用来插在每个匹配元素的后面。参数function(index)表示一个返回HTML字符串的函数，这个字符串会被插入到每个匹配元素的后面





1. insertAfter()方法

insertAfter()与after()方法具有相同的功能，主要是语法不同，特别是内容和目标的位置。



可以改写为



1. before()方法

before()方法将在每个匹配的元素之前插入内容。

before(content)

before(function(index))

参数content表示一个元素、HTML字符串或jQuery对象，用来插在每个匹配元素的前面。参数function(index)表示一个返回HTML字符串的函数，这个字符串会被插入到每个匹配元素的前面

1. insertBefore()方法

insertBefore()方法与before()方法具有相同的功能，只是语法不同，特别是内容和目标的位置。需要插入的内容在前面，选择表达式在函数的后面。



可以改写为



## 删除内容

DOM内置了removeChild()方法，该方法可以删除指定的节点及其包含的所有子节点，并返回这些删除的内容。

## 克隆内容

## 替换内容

## 包裹内容

## 属性操作

## 类操作

## 读写文本和值

## 样式表操作

## 访问文档树