jQuery培训教程

**讲师: 张晓飞**





**请拿着<<Jquery文档>>学习和编程**

# 1 jQuery简介

* 为了简化 JavaScript 的开发, 一些 JavsScript 库诞生了. JavaScript库封装了很多预定义的对象和实用函数。能帮助使用者建立有高难度交互的页面, 并且兼容各大浏览器
* 当前流行的 JavaScript 库有



* jQuery是继prototype之后又一个优秀的Javascript框架。其宗旨是——WRITE LESS,DO MORE。优势如下:
* 轻量级(压缩后只有几十k)
* 强大的选择器
* 出色的 DOM 操作的封装
* 可靠的事件处理机制
* 完善的 Ajax
* 出色的浏览器兼容性
* 链式操作方式
* 文档说明很全.
* 可扩展插件

# 2. jQuery快速入门

## 2.1. 将jQuery的库文件加入应用

* 在开发测试时, 用的是未压缩的版本: jquery-1.7.2.js
* 在上线项目中, 会使用压缩后的版本: jquery-1.7.2.min.js

## 2.2. 在页面中导入并使用

|  |
| --- |
| <!-- 导入jQuery库-->  <script type=*"text/javascript"* src=*"script/jquery-1.7.2.js"/*>  <script type=*"text/javascript"*>  <button>  ClickMe!  </button>  //调用jQuery库定义的函数  $(**function**(){  $("button").click(**function**(){  alert("----");  });  });  </script> |

# 3. Jquery语法详解

## 3.1. jQuery源码初分析

|  |
| --- |
| (**function**(window) {  **var** jQuery = **function**(selector, context) {  **return** **new** jQuery.fn.init(selector, context, rootjQuery);  };  window.jQuery = window.$ = jQuery;  })(window); |

**说明:**

* 自执行函数
* 给window添加了两个等同的函数: jQuery()/$()
* 执行$()后,返回的是一个jQuery库包装的对象, 一般称之为:jQuery对象

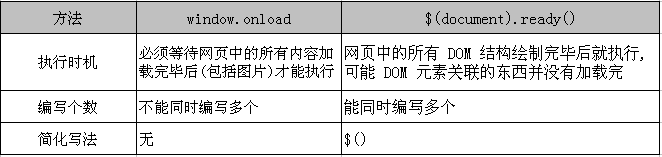
## 3.2. 核心

### 1). jQuery核心函数: $()

* jQuery库中为window对象定义了一个$函数(方法)
* $函数会根据参数数据类型的不同做不同的工作, 返回一个jQuery封装的伪数组的对象

#### 1.1). $(callback)

* 传入一个function参数, 作为回调函数
* 在DOM文档载入完成后, 回调函数自动执行
* 这个函数的作用如同$(document).ready(callback)一样
* 它与window.onload是有区别的



#### 1.2). $(selector)

* 接收一个CSS选择器格式的字符串参数
* 根据此字符串在document中去匹配一组元素, 并封装成jQuery对象返回

#### 1.3). $(domEle)

* 接收一个DOM对象类型的参数
* 返回包含这个dom对象的jQuery对象
* jQuery对象只能调用jQuery封装的方法
* DOM对象只能调用DOM中所定义的方法

#### 1.4). $(htmlString)

* 接收一个标签字符串参数
* 创建对应的标签对象, 并包装成jQuery对象
* 如: $("<div><p>Hello</p></div>")

### 2). jQuery对象访问

#### 2.1). each(callback)

* callback函数: function (index, domEle)
* 遍历jQuery对象中包含的所有DOM对象, 每拿到一个dom对象都会调用指定的回调函数, 并将下标和dom对象传入

#### 2.2). size() / length

* 得到jQuery对象中包含的所有DOM对象的个数

#### 2.3). [index] / get(index)

* 根据下标得到jQuery对象中所包含的某个dom对象
* 常用此方法来将jQuery对象转为DOM对象
* 约定：如果获取的是jQuery对象,那么要在变量前面加上$.

|  |
| --- |
| **var** nameELe = document.getElementById("name")  **var** $nameEle = $("name"); |

## 3.3. 选择器(同css)

* jQuery最牛的地方就是其强大的选择器, 使用其选择器基本可以快速轻松的找到页面的任意节点
* jquery的选择器分类
  + 基本选择器
  + 层次选择器
  + 过滤选择器
    - 基本
    - 内容
    - 可见
    - 属性
    - 子元素
    - 表单
    - 表单属性

### 1). 基本选择器

* 基本选择器是jquery中最简单，也是最常用的选择器
* 它通过标签名,id属性,class属性来查找匹配的DOM元素

#### id选择器

* 用法: $(‘#id’)
* 返回值：根据id属性匹配一个标签, 封装成jQuery对象

#### 标签选择器

* 用法: $(‘tagName’)
* 返回值：根据标签名匹配的一个或多个标签, 封装成jQuery对象

#### class选择器

* 用法: $(‘.class’)
* 返回值：根据class属性值匹配一个或多个标签, 封装成jQuery对象

#### \*选择器

* 用法: $(\*)
* 返回值: 匹配所有标签, 封装成jQuery对象

#### selector1,selector2,…

* 用法: $(”div,span,.myClass”)
* 返回值: 所有匹配选择器的标签, 封装成jQuery对象

### 2). 层次选择器

* 如果想通过DOM元素之间的层次关系来获取特定元素。例如子元素、兄弟元素等。则需要通过层次选择器。

#### 2.1). ancestor descendant

* 用法: $(”form input”)
* 说明: 在给定的祖先元素下匹配所有后代元素

#### parent > child

* 用法: $(”form > input”)
* 说明: 在指定父元素下匹配所有子元素.注意:要区分好后代元素与子元素

#### prev + next

* 用法: $(”label + input”)
* 说明: 匹配所有紧接在prev元素后的next元素

#### prev ~ siblings

* 用法: $(”form ~ input”)
* 说明: 匹配prev元素之后的所有 siblings元素, 不包含该元素在内,并且siblings匹配的是和prev同辈的元素,其后辈元素不被匹配.

### 3). 基本过滤选择器

* 过滤选择器主要是通过特定的过滤规则来筛选出所需的DOM元素, 该选择器都以 “:” 开头
* 按照不同的过滤规则, 过滤选择器可以分为基本过滤, 内容过滤, 可见性过滤, 属性过滤, 子元素过滤, 表单过滤和表单属性过滤选择器.

#### 3.1). :first

* 用法: $(”tr:first”) ;
* 说明: 匹配找到的第一个元素

#### 3.2). :last

* 用法: $(”tr:last”)
* 说明: 匹配找到的最后一个元素.与 :first 相对应

#### 3.3). :not(selector)

* 用法: $(”input:not(:checked)”)
* 说明: 去除所有与给定选择器匹配的元素.有点类似于”非”,意思是没有被选中的input(当input的type=”checkbox”).

#### 3.4). :even

* 用法: $(”tr:even”)
* 说明: 匹配所有索引值为偶数的元素，从0开始计数.js的数组都是从0开始计数的.

#### 3.5). : odd

* 用法: $(”tr:odd”)
* 说明: 匹配所有索引值为奇数的元素,和:even对应,从 0 开始计数.

#### 3.6). :eq(index)

* 用法: $(”tr:eq(0)”)
* 说明: 匹配一个给定索引值的元素.eq(0)就是获取第一个tr元素.括号里面的是索引值,不是元素排列数.

#### 3.7). :gt(index)

* 用法: $(”tr:gt(0)”)
* 说明: 匹配所有大于给定索引值的元素.

#### 3.8). :lt(index)

* 用法: $(”tr:lt(2)”)
* 说明: 匹配所有小于给定索引值的元素.

### 4). 内容过滤选择器

* 内容过滤选择器的过滤规则主要体现在它所包含的子元素和文本内容上

#### 4.1). :contains(text)

* 用法: $(”div:contains(’John’)”)
* 说明: 匹配包含给定文本的元素.这个选择器比较有用，当我们要选择的不是dom标签元素时,它就派上了用场了,它的作用是查找被标签”围”起来的文本内容是否符合指定的内容的.

#### 4.2). :empty

* 用法: $(”td:empty”)
* 说明: 匹配所有不包含子元素或者文本的空元素

#### 4.3). :has(selector)

* 用法: $(”div:has(p)”).addClass(”test”)
* 说明: 匹配含有选择器所匹配的元素的元素.这个解释需要好好琢磨,但是一旦看了使用的例子就完全清楚了:给所有包含p元素的div标签加上class=”test”.

#### 4.4). :parent

* 用法: $(”td:parent”)
* 说明: 匹配含有子元素或者文本的元素.注意:这里是”:parent”,可不是”.parent”哦!感觉与上面讲的”:empty”形成反义词.

### 5). 可见过滤选择器

* 根据元素的可见和不可见状态来选择相应的元素

#### 5.1). :hidden

* 用法: $(”tr:hidden”)
* 说明: 匹配所有的不可见元素，input 元素的 type 属性为 “hidden” 的话也会被匹配到.意思是css中display:none和input type=”hidden”的都会被匹配到.同样,要在脑海中彻底分清楚冒号”:”, 点号”.”和逗号”,”的区别.

#### 5.2). :visible

* 用法: $(”tr:visible”)
* 说明: 匹配所有的可见元素.

### 6). 属性过滤选择器

* 属性过滤选择器的过滤规则是通过元素的属性来获取相应的元素

#### 6.1). [attribute]

* 用法: $(”div[id]“)
* 说明: 匹配包含给定属性的元素. 例子中是选取了所有带id属性的div标签.

#### 6.2). [attribute=value]

* 用法: $(”input[name='newsletter']“).attr(”checked”, true)
* 说明: 匹配给定的属性是某个特定值的元素.例子中选取了所有name属性是newsletter的 input 元素.

#### 6.3). [attribute!=value]

* 用法: $(”input[name!='newsletter']“).attr(”checked”, true)。
* 说明：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.此选择器等价于:not([attr=value]),要匹配含有特定属性但不等于特定值的元素,请使用[attr]:not([attr=value]).之前看到的 :not 派上了用场.

#### 6.4). [attribute^=value]

* 用法: $(”input[name^=‘news’]“)
* 说明: 匹配给定的属性是以某些值开始的元素.,我们又见到了这几个类似于正则匹配的符号.现在想忘都忘不掉了吧?!

#### 6.5). [attribute$=value]

* 用法: $(”input[name$=‘letter’]“)
* 说明: 匹配给定的属性是以某些值结尾的元素.

#### 6.6). [attribute\*=value]

* 用法: $(”input[name\*=‘man’]“)
* 说明: 匹配给定的属性是以包含某些值的元素.

### 7). 子元素过滤选择器

#### 7.1). :nth-child(index/even/odd/equation)

* 用法: $(”ul li:nth-child(2)”)
* 说明: 匹配其父元素下的第N个子或奇偶元素.这个选择器和之前说的[基础过滤(Basic Filters)](http://flex.joelove.cn/jquery-selectors-basicfilters/)中的 eq() 有些类似,不同的地方就是前者是从0开始,后者是从1开始.

#### 7.2). :first-child

* 用法: $(”ul li:first-child”)
* 说明: 匹配第一个子元素.’:first’ 只匹配一个元素,而此选择符将为每个父元素匹配一个子元素.这里需要特别点的记忆下区别.

#### 7.3). :last-child

* 用法: $(”ul li:last-child”)
* 说明: 匹配最后一个子元素.’:last’只匹配一个元素,而此选择符将为每个父 元素匹配一个子元素.

#### 7.4). : only-child

* 用法: $(”ul li:only-child”)
* 说明: 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.意思就是:只有一个子元素的才会被匹配!

### 8). 表单选择器

#### 9.1). :input

* 用法: $(”:input”)
* 说明:匹配所有 text, textarea, select 和 button 元素

#### 9.2). :text

* 用法: $(”:text”)
* 说明: 匹配所有的单行文本框.

#### 9.3). :password

* 用法: $(”:password”)
* 说明: 匹配所有密码框.

#### 9.4). :radio

* 用法: $(”:radio”)
* 说明: 匹配所有单选按钮.

#### 9.5). :checkbox

* 用法: $(”:checkbox”)
* 说明: 匹配所有复选框

#### 9.6). :submit

* 用法: $(”:submit”)
* 说明: 匹配所有提交按钮

#### 9.7). :image

* 用法: $(”:image”)
* 说明: 匹配所有图像域.

#### 9.8). :reset

* 用法: $(”:reset”)
* 说明: 匹配所有重置按钮.

#### 9.9). :button

* 用法: $(”:button”)
* 说明: 匹配所有按钮.这个包括直接写的元素button.

#### 9.10). :file

* 用法: $(”:file”)
* 说明: 匹配所有文件域.

#### 9.11). :hidden

* 用法: $(”input:hidden”)
* 说明: 匹配所有不可见元素，或者type为hidden的元素.这个选择器就不仅限于表单了,除了匹配input中的hidden外,那些style为hidden的也会被匹配.

### 9). 表单对象属性过滤选择器

* 此选择器主要对所选择的表单元素进行过滤

#### 8.1). :enabled

* 用法: $(”input:enabled”)
* 说明: 匹配所有可用元素.意思是查找所有input中不带有disabled=”disabled”的input.不为disabled,当然就为enabled啦.

#### 8.1). :disabled

* 用法: $(”input:disabled”)
* 说明: 匹配所有不可用元素.与上面的那个是相对应的.

#### 8.1). :checked

* 用法: $(”input:checked”)
* 说明: 匹配所有选中的被选中元素(复选框、单选框等，不包括select中的option).这话说起来有些绕口.

#### 8.1). :selected

* 用法: $(”select option:selected”)
* 说明: 匹配所有选中的option元素.

## 文档处理(CRUD)

### 1). 内部插入节点

#### 1.1). append(content)

* 向每个匹配的元素的内部的结尾处追加内容

#### 1.2). appendTo(content)

* 把所有匹配的元素追加到另一个指定的元素元素集合中

#### 1.3). prepend(content)

* 向每个匹配的元素的内部的开始处插入内容

#### 1.4). prependTo(content)

* 将每个匹配的元素插入到指定的元素内部的开始处

### 2). 外部插入节点

#### 2.1). after(content) :

* 在每个匹配的元素之后插入内容

#### 2.2). before(content)

* 在每个匹配的元素之前插入内容

#### 2.3). insertAfter(content)

* 把所有匹配的元素插入到另一个、指定的元素元素集合的后面

#### 2.4). insertBefore(content)

* 把所有匹配的元素插入到另一个、指定的元素元素集合的前面

### 3).查找节点

#### 3.1). 使用jQuery选择器查询

* $(selector)
* 得到一个包含所有匹配的dom节点对象的jQuery对象

#### 3.2). 查询jQuery对象内部数据

* $object.find(selector)
* 在Jquery对象中根据selector查找其中匹配的后代节点

#### 3.3). 遍历jQuery对象包含的数据

* $(selector1).each(function(index, itemDom){ })
* 遍历jQuery对象所包含的所有节点, 每取一个dom节点对象都去调用设置的回调函数, 并将取出的节点在数组中的下标和节点对象传入函数

### 4).创建节点

* $(htmlString).
* 动态创建的新元素节点不会被自动添加到文档中, 需要使用其他方法将其插入到文档中;
* 当创建单个元素时, 需注意闭合标签和使用标准的 XHTML 格式. 例如创建一个<p>元素, 可以使用 $(“<p/>”) 或 $(“<p></p>”), 但不能使用 $(“<p>”) 或 $(“</P>”)
* 创建文本节点就是在创建元素节点时直接把文本内容写出来; 创建属性节点也是在创建元素节点时一起创建

### 5). 删除节点

#### 5.1). empty():

* 删除匹配的元素集合中所有的子节点(不包括本身)。

#### 5.2). remove():

* 删除匹配的元素及其子元素(包括本身)

### 6). 属性操作

#### 6.1). attr(name [,value])

* 根据属性名获取属性值或者设置一个属性

#### 6.2). removeAttr(name)

* 根据属性名删除对应的属性

### 7). HTML代码/值

#### 7.1). html([val])

* 得到元素的内容或者设置元素的内容

#### 7.2). val([value])

* 获得匹配元素的当前值或者设置其值

### 8) CSS

#### 8.1). addClass(className)

* 添加class属性

#### 8.2). removeClass()

* 移除class属性

#### 8.3). css(name, [value])

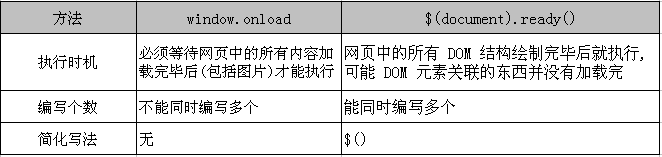
* 查看某个样式属性, 或设置某个样式属性

## 事件

### 常用的事件

#### 1.1). ready(fn)

* 当DOM载入就绪可以查询及操纵时绑定一个要执行的函数
* 它与window.onload是有区别的



#### 1.2). click([fn])

* 触发每一个匹配元素的click事件

#### 1.3). blur([fn])

* 触发每一个匹配元素的blur事件

#### 1.4). change([fn])

* 触发每一个匹配元素的change事件

### 绑定与解绑事件

#### 2.1). bind(type, fn)

* 为每个匹配元素的特定事件绑定事件处理函数。

#### 2.2). unbind(type)

* bind()的反向操作，从每一个匹配的元素中删除绑定的事件

### 事件切换

#### 3.1). hover(over,out)

* 当鼠标移动到一个匹配的元素上面时，会触发指定的第一个函数。当鼠标移出这个元素时，会触发指定的第二个函数。

### 事件冒泡

* 描述: 事件会按照 DOM 层次结构像水泡一样不断向上只止顶端
* 解决: 在事件处理函数中返回 false, 会对事件停止冒泡

## 效果

### 基本

#### show()

* 显示当前标签

#### hide()

* 隐藏当前标签

#### toggle()

* 切换当前标签的可见性

### 滑动

#### 2.1). slideDown()

* 这个动画效果只调整元素的高度，可以使匹配的元素以“滑动”的方式显示出来

#### 2.2). slideUp()

* 这个动画效果只调整元素的高度，可以使匹配的元素以“滑动”的方式隐藏起来

#### 2.3). slideToggle()

* 通过高度变化来切换所有匹配元素的可见性

### 淡入淡出

#### 3.1). fadeIn()

* 通过不透明度的变化来实现所有匹配元素的淡入效果

#### 3.2). fadeOut()

* 通过不透明度的变化来实现所有匹配元素的淡出效果

#### 3.3). fadeToggle()

* 通过不透明度的变化来开关所有匹配元素的淡入和淡出效果

## 处理XML文件

### 加载XML文件

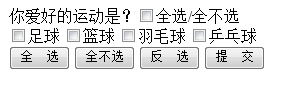
* 可以使用$函数中定义的get()函数或post()函数加载本地或远程的xml文件, 得到文件中包含的dom数据
* get(path, fn, dataType) post(path, fn, dataType)
* 例子:

|  |
| --- |
| $.get(  "cities.xml",  **function**(dom){  //dom为cities.xml包含的文档对象  },  "xml"  ); |

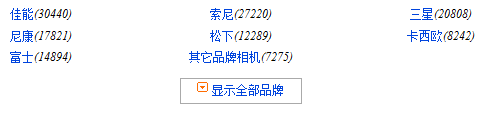
### 操作XML数据

# 练习

## 练习1: 爱好选择

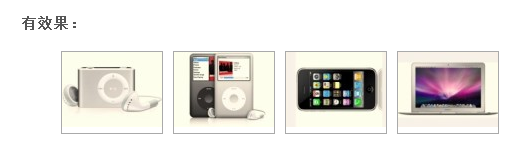


## 练习2: 分类动态展示





## 练习3: 动态查看图片





## 练习4: 员工管理



## 练习5: 二级联动