logo.png

jquery

H5 web前端

[www.bufanui.com](http://www.bufanui.com)

# 一、jquery

## 1.1 为什么要学jQuery？

window.onload 事件有个事件覆盖的问题，我们只能写一个

代码容错性差 document.~~getElementByClassName~~

浏览器兼容性问题 ….各种兼容问题，bom更严重！

书写很繁琐，代码量多

动画效果，我们很难实现 （jquery的动画 敬请期待！）

都是原生js操作dom的痛点

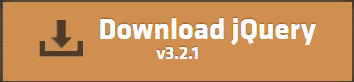
jQuery 是js对dom操作的一个封装，我们可以通过使用jQuery快速简单的操作dom元素.

## 1.2 jQuery是什么？

jQuery是一个快速、简洁的JavaScript框架，是继Prototype之后又一个优秀的JavaScript代码库（或JavaScript框架）。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。它封装JavaScript常用的功能代码，提供一种简便的JavaScript设计模式，优化HTML文档操作、事件处理、动画设计和Ajax交互。

目前这个阶段，主要学习如何来使用jQuery操作DOM，其实就是学习jQuery封装好的那些功能方法，这些方法叫做API（Application Programming Interface应用程序编程接口）。

# 二、jQuery的基本使用



Jquery 主要版本

1.xx 1版本 支持ie5678.。。

2.xx 2 版本 放弃了对ie5678的支持

3.xx 3 版本 没向前兼容

这个版本并不是越新越好。根据项目需求。

但是各个版本的基础功能都是一样的。常用api也是一样的。

Jquery的版本选择 需要看项目中依赖的或者使用的jquery插件来决定

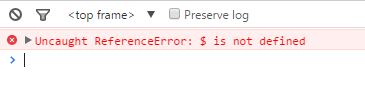


在开发中 一般使用开发版本，非压缩版本。可以查看源码（比如api不明确的时候，不知道怎么用的时候）。

但是对于jquery 没必要。Jquery很成熟文档很全面，也不存在修改源码的情况。

1. 获取源文件
2. 引入页面

问题：



如果遇到这种问题，那肯定是没有引用我们的jQuery源文件，或者先了代码 后引入了jqury

用jQuery之前，先引入jQuery，然后，再去写我们的jQuery代码。

如果使用了第三方jquery插件（日历、滚动、瀑布流等），必须先引入jquery，再引入该插件。

### 2.1 入口函数

### $(document).ready(function(){});

### 2.$(function(){}); 推荐

### 2.2 事件处理

* 事件源

1. 类选择器 .d1 类 是为了告诉程序员这个标签是什么，它跟谁是一样（类）的，类一般所有的标签都有
2. Id选择器 #div1 id=’goods\_1’ id是为了区分为一，是js中使用的，id是在需要的时候才有，多用于列表等重复性标签
3. 标签选择器 div

* 事件

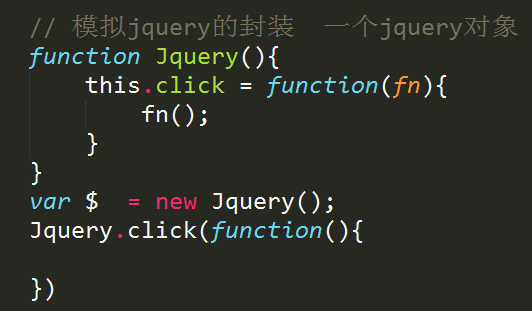
Js: onclick onmouseover onmousemove …

Jquery: click mouseover mousemove …

* 事件处理程序

Js: .onclick = function(){ }

Jquery: click(function(){ });



注意： jquery中 的事件处理都是方法 ，原生js的事件处理是属性

# 三、详细介绍

### 3.1 $

a)命名归法：下划线、字母、$、数字

b)但是不能以数字作为开头

jQuery的两个变量：$ 和 jQuery

注意：不能再使用这两个变量 防止引起冲突，什么时候使用jQuery？当有其他框架中使用$造成冲突的时候使用jQuery。

### 3.2 js入口函数跟jQuery入口函数的区别：

**1.执行时间**

window.onload必须等到页面内包括图片的所有元素加载完毕后才能执行。

$(document).ready()是DOM结构绘制完毕后就执行，不必等到加载完毕。只加载了dom框架，对于大的图片需要时间，这个不等。 <img scr=’’>

**2.编写个数不同**

window.onload不能同时编写多个，如果有多个window.onload方法，只会执行一个

$(document).ready()可以同时编写多个，并且都可以得到执行

**3.简化写法**

window.onload没有简化写法

$(document).ready(function(){})可以简写成$(function(){});

## 3.3 jQuery基本选择器

### 3.3.1 回顾CSS选择器

|  |  |  |
| --- | --- | --- |
| **CSS选择器回顾** | | |
| **符号** | 说明 | 用法 |
| **#id** | Id选择器 | #id{ color:red; } |
| **.class** | 类选择器 | .class{ //} |
| **Element** | 标签选择器 | P { //} |
| **\*** | 通配符选择器 | 配合其他选择器来使用 |
| **,** | 并集选择器 | div,p{} |
| **空格** | 后代选择器 | div span{}  选择div下面所有后代的span |

### 3.3.2 jQuery基本选择器

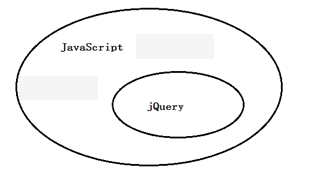
|  |  |  |
| --- | --- | --- |
| **基本选择器** | | |
| **符号** | 说明 | 用法 |
| **$(“#demo”)** | 选择id为demo的第一个元素 | $(“#demo”).css(“background”,”red”) |
| **$(“.d1”)** | 选择所有类名（样式名）为dl的元素 | $(“.d1”). css(“background”,”red”); |
| **$(“div”)** | 选择所有标签名字为div的元素 | $(“div”). css(“background”,”red”); |
| **$(“\*”)** | 选择所有元素 | $(“\*”). css(“background”,”red”) |
| **$(“.arr.arr-l”)** | 选择多个指定的元素，这个地方是选择出了 .liItem元素和div元素 | $(“.arr.arr-l”). css(“background”,”red”) |

总结：对比之后发现 jquery的选择器跟css是一模一样的

## 3.4 Javascipt跟jQuery的区别：

Javascript是一门编程语言，我们用它来编写客户端浏览器脚本。

jQuery是javascript的一个库，包含多个可重用的函数，用来辅助我们简化javascript开发



jQuery能做的javascipt都能做到，而javascript能做的事情，jQuery不一定能做到。

所有的前段框架的基础都是javascript，动画基础都是前面跟大家讲过的动画原理。包括以后会讲的vue.js

Angular.js 甚至市面上各种前段框架，基础都是JavaScript。

只要框架能实现，js原生写法都能实现。

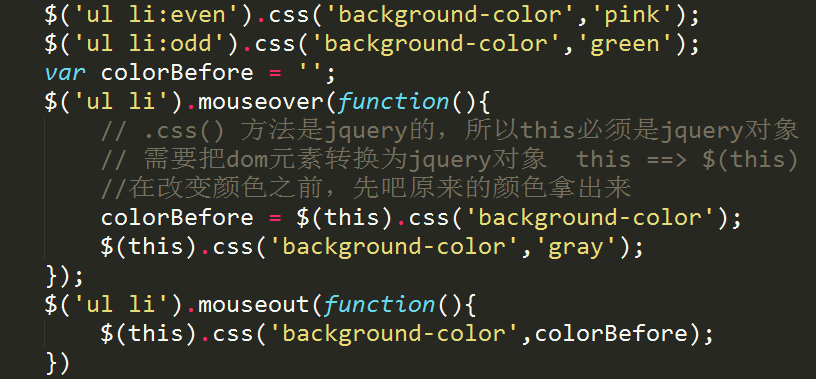
## 3.5 jQuery其他选择器

### 3.5.1层级选择器

|  |  |  |
| --- | --- | --- |
|  | | |
| **符号** | 说明 | 用法 |
| **空格** | 后代选择器  选择所有的后代元素 | $(“div span”). css(“background”,”red”); |
| **>** | 子代选择器  选择所有的子代元素 | $(“div > span”). css(“background”,”red”) |
| **+** | 紧邻选择器  选择紧挨着的下一个元素 | $(“div + p”). css(“background”,”red”) |
| **~** | 兄弟选择器  选择后面的所有的兄弟元素 | $(“div ~ p”). css(“background”,”red”) |

### 3.5.2过滤选择器

|  |  |  |
| --- | --- | --- |
| **基本过滤选择器** | | |
| **符号** | 说明 | 用法 |
| **:eq(index)** | index是从0开始的一个数字，选择序号为index的元素。选择第一个匹配的元素。 | $(“li:eq(1)”). css(“background”,”red”) |
| **:gt(index)** | Index 是从0开始的一个数字，选择序号大于index的元素 | $(“li:gt(2)”). css(“background”,”red”) |
| **:lt(index)** | Index是从0开始的一个数字，选择小于index 的元素 | $(“li:lt(2)”). css(“background”,”red”) |
| **:odd** | 选择所有序号为奇数行的元素 | $(“li:odd”). css(“background”,”red”) |
| **:even** | 选择所有序号为偶数的元素 | $(“li:even”). css(“background”,”red”) |
| **:first** | 选择匹配第一个元素 | $(“li:first”). css(“background”,”red”) |
| **:last** | 选择匹配的最后一个元素 | $(“li:last”). css(“background”,”red”) |



### 3.5.3属性选择器

|  |  |  |
| --- | --- | --- |
| **属性选择器** | | |
| **符号** | 说明 | 用法 |
| **$(“a[href]”) title type …** | 选择所有包含href属性的元素 | $(“a[href]”). css(“background”,”red”) |
| **$(“a[href=’baidu’]”)** | 选择href属性值为baidu的所有a标签 | $(“a[href=’baidu]”). css(“background”,”red”) |
| **$(“a[href!=’baidu’]”)** | 选择所有href属性不等baidu的所有元素，包括没有href的元素 | $(“a[href!=’baidu’]”). css(“background”,”red”) |
| **$(“a[href^=’web’]”)** | 选择所有以web开头的元素 | $(“a[href^=’web’]”). css(“background”,”red”) |
| **$(“a[href$=’cn’]”)** | 选择所有以cn结尾的元素 | $(“a[href$=’cn’]”). css(“background”,”red”) |
| **$(“a[href\*=’i’]”)** | 选择所有包含i这个字符的元素，可以是中英文 | $(“a[href\*=’i’]”). css(“background”,”red”) |
| **$(“a[href][title=’内容’]”)** | 选择所有符合指定属性规则的元素，都符合才会被选中。 | $(“a[href][title=’内容’]”). css(“background”,”red”) |

### 3.5.4筛选选择器（方法）

|  |  |  |
| --- | --- | --- |
| **符号(名称)** | **说明** | **用法** |
| **find(selector)** | 查找指定元素的所有后代元素（子子孙孙） | $(“#j\_wrap”).find(“li”).css(“color”, “red”);  选择id为j\_wrap的所有后代元素li |
| **children()** | 查找指定元素的直接子元素（亲儿子元素） | $(“#j\_wrap”).children(“ul”).css(“color”, “red”);  选择id为j\_wrap的所有子代元素ul |
| **siblings()** | 查找所有兄弟元素（不包括自己） | $(“#j\_liItem”).siblings().css(“color”, “red”);  选择id为j\_liItem的所有兄弟元素 |
| **parent()** | 查找父元素（亲的） | $(“#j\_liItem”).parent(“ul”).css(“color”, “red”);  选择id为j\_liItem的父元素 |
| **eq(index)** | 查找指定元素的第index个元素，index是索引号，从0开始 | $(“li”).eq(2).css(“color”, “red”);  选择所有li元素中的第二个 |

## mouseover事件跟mouseenter事件的区别：

mouseover/mouseout事件，鼠标经过的时候会触发多次，每遇到一个子元素就会触发一次。

mouseenter/mouseleave事件，鼠标经过的时候只会触发一次

## 四、DOM对象跟jQuery对象相互转换

Div 🡺 对div的包装，使其具有各种各样已经封装好的方法。

jQuery对象转换成DOM对象:

方式一：$(“#btn”)[0]

方式二：$(“#btn”).get(0)

DOM对象转换成jQuery对象：

$(document) -> 把DOM对象转成了jQuery对象

var btn = document.getElementById(“bt n”);

btn -> $(btn);