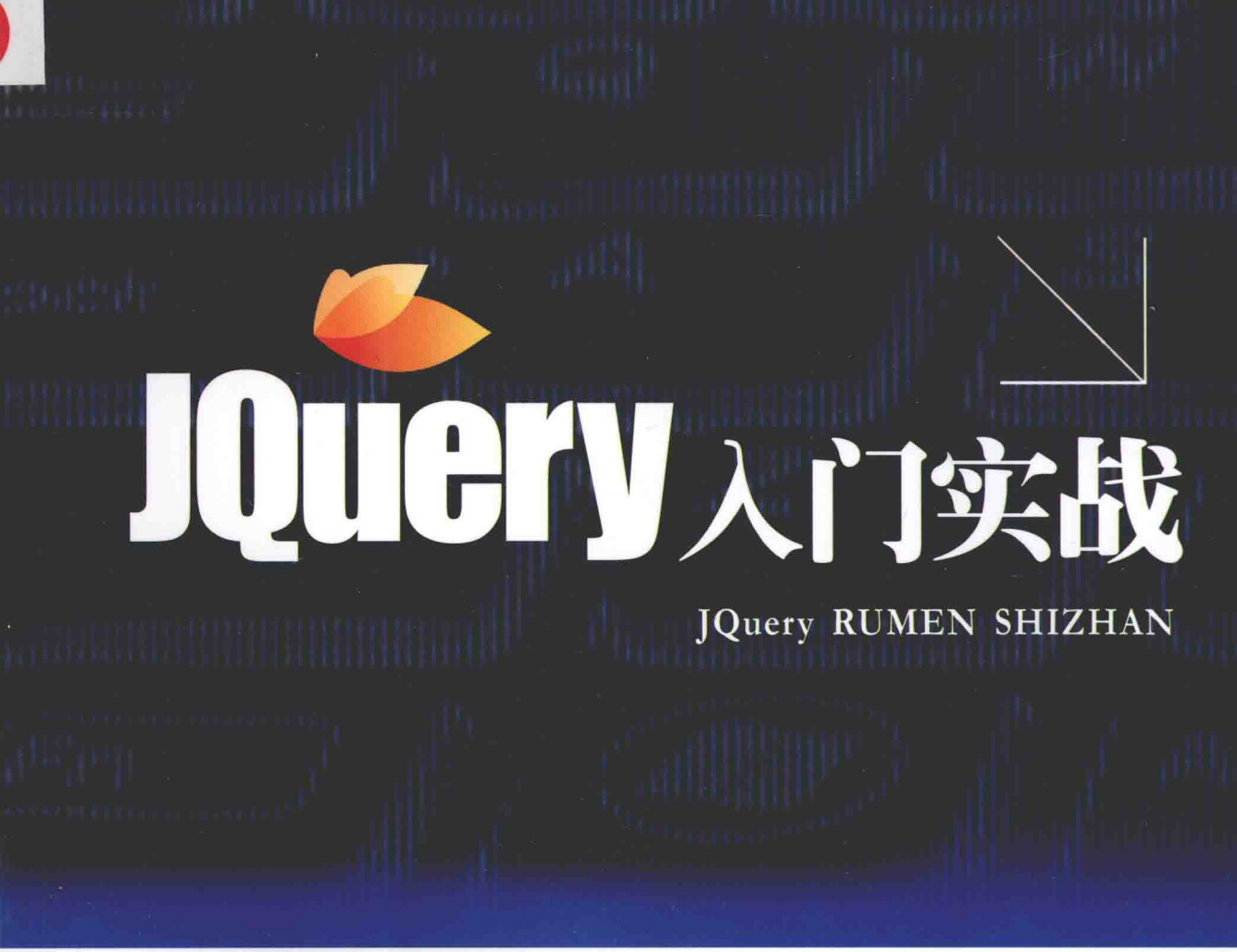

JQuery入门实战

JQuery RUMEN SHIZHAN

主编 汤 东 张富银

JQuery是一个快速、简单的JavaScript library，
它简化了HTML文件的traversing、事件处理、动画、AJAX互动，
从而方便了网页制作的快速发展。

本书详细地讲解了JQury的各种方法和使用技巧，
读者可以系统地掌握JQuery中关于
DOM操作、事件、动画效果、表单操作、Ajax以及插入方面的知识点。
本书适合所有对JQuery技术感兴趣的
WEB设计的前端开发人员、后端开发人员阅读学习。

西南财经大学出版社
Southwestern University of Finance & Economics Press

JQuery入门实战

JQuery RUMEN SHIZHAN

主编 汤 东 张富银

西南财经大学出版社
Southwestern University of Finance & Economics Press

图书在版编目(CIP)数据

JQuery 入门实战/汤东,张富银主编. —成都:西南财经大学出版社,2015.7

ISBN 978-7-5504-2006-9

Ⅰ.①J… Ⅱ.①汤…②张… Ⅲ.①JAVA 语言—程序设计
Ⅳ.①TP312

中国版本图书馆 CIP 数据核字(2015)第 146070 号

JQuery 入门实战

主编:汤 东 张富银

责任编辑:邓克虎

助理编辑:傅倩宇

封面设计:何东琳设计工作室

责任印制:封俊川

出版发行	西南财经大学出版社(四川省成都市光华村街 55 号)
网 址	http://www.bookcj.com
电子邮件	bookcj@foxmail.com
邮政编码	610074
电 话	028-87353785 87352368
照 排	四川胜翔数码印务设计有限公司
印 刷	四川五洲彩印有限责任公司
成品尺寸	185mm×260mm
印 张	10.75
字 数	260 千字
版 次	2015 年 7 月第 1 版
印 次	2015 年 7 月第 1 次印刷
印 数	1—2000 册
书 号	ISBN 978-7-5504-2006-9
定 价	25.00 元

前　言

一、编写本书的目的

市场上 JQuery 的书籍全部都只写了基础的使用，并没有利用 JQuery 开发一套完整的 demo 案例，造成新手入门难、熟手不愿意看的局面。本书打破了传统编写手法，全部采用真实案例讲解，并且保证所有源代码均能正常运行。

二、本书主要讲解的内容

本书详细地讲解了 JQuery 的各种方法和使用技巧，读者可以系统地掌握 JQuery 中关于 DOM 操作、事件、动画效果、表单操作、AJAX 以及插入方面的知识点，并且在本书的第十六章我们会参考成熟案例详细讲解 JQuery 项目的开发，为新手入门打下坚实基础。

本书共分十六章。

第一章首先介绍了什么是 JQuery、学习 JQuery 的条件、JQuery 的版本、JQuery 的功能和优势、其他 JavaScript 库、是否兼容低版本 IE、下载及运行 JQuery。

第二章介绍了 JQuery 的基础核心内容，包含代码风格、加载模式、对象互换、多个库之间的冲突。

第三章主要讲解常规选择器，本章节是 JQuery 入门的关键，主要由简单选择器、进阶选择器、高级选择器组成。

第四章主要讲解过滤选择器，包括基本过滤器、内容过滤器、可见性过滤器、子元素过滤器、其他方法。

第五章主要讲解基础 DOM 和 CSS 操作，包括 DOM 简介、设置元素及内容、元素属性操作、元素样式操作、CSS 方法。

第六章主要讲解 DOM 节点操作，包括创建节点、插入节点、包裹节点、节点操作。

第七章主要讲解表单选择器，包括常规选择器、表单选择器、表单过滤器。

第八章主要讲解事件中的基础事件，包括绑定事件、简写事件、复合事件。

第九章主要讲解事件对象中的基础事件对象和冒泡与默认行为。

第十章主要讲解事件中的高级事件部分，包括使用最多的模拟操作、命名空间、事件委托、On、Off 和 One。

第十一章主要讲解 JQuery 中的动画效果，包括动画的显示、隐藏、滑动、卷动、淡入、淡出、自定义动画、列队动画方法、动画相关方法、动画全局属性。

前　言

第十二章讲解 JQuery 中的 AJAX 应用，首先介绍了 AJAX 的优势与不足，讲解了 load（）方法、$. get（）和 $. post（）、$. getScript（）和 $. getJSON（）、$. ajax（）方法、表单序列化。

第十三章讲解 AJAX 的进阶应用，主要解决了具体的 AJAX 使用中最常遇到的问题及解决方法。AJAX 加载请求、AJAX 错误处理、AJAX 请求全局事件、AJAX 的跨域 JSON 和 JSONP、jqXHR 对象。

第十四章讲解 JQuery 工具虑数，如字符串操作、数组和对象操作、测试操作、URL 操作、浏览器检测、其他操作。

第十五章讲解 JQuery 的插件机制以及开发自己想要的插件，主要包括插件概述、验证插件、自动完成插件、自定义插件。

第十六章将用前面第一章至第十五章全部知识开发一套完成的项目，项目主要参考知乎网（http: //www. zhihu. com）。在项目中将使用到 JQuery UI、邮箱自动补全、日历 UI、验证插件（validate）、form 表单插件、cookie 插件、AJAX 登录、AJAX 提问、AJAX 显示问题、AJAX 提交评论、AJAX 显示评论、AJAX 加载更多评论、处理错误与屏蔽低版 IE 等。

三、本书适合您吗

本书适合所有对 JQuery 技术感兴趣的 WEB 设计的前端开发人员、后端开发人员阅读学习。

阅读此书需要一定的 HTML、CSS 和 JavaScript 基础。

四、本书约定

本书所有例子都是基于 JQuery1. 10 版而编写。

如果没有特殊说明，JQuery 默认是导入的。

如果没有特殊说明，程序中的 $ 符号都是 JQuery 的一个简写行为。

如果没有特殊说明，所有网页的头部都必须有标准的 DOCTYPE 声明。

如果没有特殊说明，所有网页的编码都是 UTF-8 无 BOM 格式。

五、读者反馈与示例下载

我们十分欢迎来自读者的宝贵意见与建议，这些建议可以是您感兴趣的内容，或者是本书没有介绍到而又是您十分需要的知识。

作者

2015 年 5 月

目　录

第一章
JQuery 入门

教学要点：

1. 什么是 JQuery
2. 学习 JQuery 的必要条件
3. JQuery 的版本
4. JQuery 的功能和优势
5. 其他 Javascript 库
6. 是否兼容低版本的 IE
7. 下载及运行 JQuery

教学重点：

1. 理解框架的概念
2. 掌握新技术的学习方法
3. 练习 JQuery 的编程手感

教学难点：

JQuery 的编程手感

一、什么是 JQuery

JQuery 是一个 Javasript 库，通过封装原生的 Javascript 函数得到一套定义好的方法。JQuery 是 John Resig 于 2006 年创建的一个开源项目。随着越来越多的开发者加入，JQuery 已经集成了 Javascript，CSS，DOM，AJAX 于一体的强大功能，它可以用更少的代码完成更多更强大更复杂的功能，从而得到开发者的青睐。

二、学习 JQuery 的必要条件

JQuery 是 Javascript 库，所以 JQuery 在使用上要比 Javascript 简单，但对于网页编程来说，有些通用的基础知识是必备的：

(1)XHTML 或 HTML5(含 CSS)；

(2)Javascrtpt；

(3)服务器端语言(如 PHP、JAVA、.NET)。

三、JQuery 的版本

2006 年 8 月正式发了 JQuery1.0 版本，第一个稳定版本，具有对 CSS 选择符、事件处理以及 AJAX 的交互支持。

2007 年 1 月发布了 JQuery1.1 版本，极大地简化了 API，合并了许多极少使用的方法。

2007 年 7 月发布了 JQuery1.1.3 版本，优化了 JQuery 选择符引擎的执行速度。

2007 年 9 月发布了 JQuery1.2 版本，优化了 Xpath 选择器，增加了命名空间事件。

2009 年 1 月发布了 JQuery1.3，使用了全新的选择符引擎 Sizzle，性能得到进一步提升。

2010 年 1 月发布了 JQuery1.4，进行了一次大规模更新，提供了 DOM 操作，增加了很多新的方法或是增强了原有的方法。

2010 年 2 月发布了 JQuery1.4.2，添加了.delegate()和.undelegate()两个新方法，提升了灵活性和浏览器的一致性，对事件系统进行了升级。

2011 年 1 月发布了 JQuery1.5，重写了 AJAX 组件，增强了扩展性和性能。

2011 年 5 月发布了 JQuery1.6，重写了 Attribute 组件，引入了新对象和方法。

2011 年 11 月发布了 JQuery1.7，引入了.on()和.off()简介的 API 解决事件绑定及委托容易混淆的问题。

2012 年 3 月发布了 JQuery1.7.2，进行一些优化和升级。

2012 年 7 月发布了 JQuery1.8，8 月发布了 JQuery1.8.1，9 月发布了 JQuery1.8.2，重写了选择符引擎，修复了一些问题。

2013 年 1 月发布了 JQuery1.9，CSS 的多属性设置，增强了 CSS3。

2013 年 5 月发布了 JQuery1.10，增加了一些功能。2013 年 4 月发布了 JQuery2.0，5 月发布了 JQuery2.0.2，一个重大更新版本，不再支持 IE6/7/8，体积更小、速度更快。

本书我们使用的是最新的中文版的 API 文档(1.8 版本)，有在线和离线两种手段查阅：①在线的 AP 文档可以访问：http://t.mb5u.com/jquery/。②离线的 AP 文档将打包提供给大家。

其中，版本号升级主要有三种：第一种是大版本升级，比如 1.×.× 升级到 2.×.×，这种升级规模是最大的，改动的地方是最多的，周期也是最长的。比如 JQuery 从 1.×.×到 2.×.×用了 7 年。第二种是小版本更新，比如从 1.7 升级到 1.8，改动适中，增加或减少了一些功能，一般周期半年到一年左右。第三种是微版本更新，比如从 1.8.1 升级到 1.8.2，修复一些 bug 或错误之类。

版本的内容升级也主要有三种：第一种是核心库的升级，比如优化选择符、优化 DOM 或者 AJAX 等；这种升级不影响开发者的使用。第二种是功能性的升级，比如剔除一些过时的方法、新增或增强一些方法等。这种升级需要了解和学习。第三种就是 BUG 修复之类的升级，对开发者使用没有影响。

学习者有一种担忧，比如学了 1.3 版本的 JQuery，那么以后升级新版本是不是还需要重学？没必要，因为并不是每次升级一个版本都会增加或剔除功能的，一半左右都是内部优化，升级到新版本并不需要任何学习成本。就算在新的版本增加了一些功能，只需要几分钟了解一下即可使用，无需清零之前的知识，只需后续累加。当然，在早期的 JQuery 版本都创建了最常用的功能，而新版本中增加的功能，也不是最常用的，无需立即

学习,立马用起。

四、JQuery 的功能和优势

JQuery 作为封闭的 JavaScript 库,其目的就是简化开发者使用 JavaSript。主要的功能有以下几点:

(1)像 CSS 哪样访问和操作 DOM;

(2)修改 CSS 控制页面外观;

(3)简化 JavaSript 代码操作;

(4)事件处理更加容易;

(5)各种动画效果使用方便;

(6)让 AJAX 技术更加完美;

(7)基于 JQuery 的大量插件;

(8)自动扩展功能插件。

JQuery 最大的优势就是使用特别方便,比如模仿 CSS 获取 DOM 对象,比原生的 JS 要简单和方便得多,并且在多个 CSS 的集中处理上非常舒服。而最常用的 CSS 功能又封装了单独的方法,感觉非常有心。最重要的是 JQuery 的代码兼容性非常好,你不需要总是去考虑不同浏览器的兼容问题。

五、其他的 Javascript 库,

目前除了 JQuery 外还有五个库比较流行,分别是 YUI,Prototype,MooTools,DOJO 和 ExtJs。

YUI 是雅虎公司开发的一套完备的、扩展性良好的富交互网页工具集。

Prototype 是最早成型的 JavaSctipt 库之一,对 JavaScript 内置对象做了大量的封装和扩展。

MooTools 是一个简洁、模块化的面向对象的 JavaScript 框架。

DoJo 最强大的在于提供其他库没有的功能,如离线存储、图标资源等。

EXtjs 简称 Ext,原本是对 YUI 的一个扩展,主要用于创建前端用户界面(收费)。

六、是否兼容 IE 低版本

这次 JQuery 发布了大版本 2.×.×,完全放弃了兼容 IE6/7/8。不仅如此,很多国际上的大型站点也开始逐步不再支持 IE6/7/8。但对于国内而言,比较大型的网站最多只是抛弃 IE6,或者部分功能不支持 IE6 的警示框,还没可能一下子把 IE6/7/8 全面抛弃。这里我们就谈一谈你的项目是否有必要兼容 IE6/7/8。

完全不支持 IE6 的示例:网易云课堂——http://study.163.com

完全不支持的做法,就是检测到是否为 IE6 或 IE6、IE8,然后直接跳转到一个信息错误界面,让你更换或升级浏览器,否则无法访问使用。

部分功能不支持的做法,就是判断你是 IE6 或 IE6、IE8,然后给一个警示条或弹出窗,告诉你使用此款浏览器性能降低或部分功能使用不正常或不能使用的提示,但还可以访问使用。

虽然大部分国内网站用 IE6 去运行都能基本兼容,但很多细节上还是有些问题,导致不能流畅的去使用。

这个问题争论很久,支持兼容的人会拿国情和使用率来证明。不支持兼容的人会用技术落后导致整个水平落后来证明。其实这两种说法都有值得商榷的地方。

首先,传统行业失败率为97%,而新的IT行业的失败率更高达99%以上(数据可能不精确,但可以说明失败率很高)。那么站在更高的角度去看你的项目,你不管是付出3倍成本去完成一个用户体验一般但兼容性很好的项目,还是付出正常成本去完成用户体验很好但不兼容低版本浏览器。这两种情况不管是哪一种,最终可能都会失败。那么你愿意选择哪种?

是否兼容IE6或IE6、IE8并不单纯是用户基数和国情的问题,而很多项目发起人只一味地用这种理由去判定需求,那么失败也在所难免。除此之外,我们还应该考虑以下几个方面的问题:

1. 成本控制

很多项目往往在6、12、18、32个月就会发生财务问题,比如资金紧缩甚至断裂。所以,成本控制尤为重要。项目如果不是老站升级,也不是大门户的新闻站,成本控制和尽快上线测试才是最重要的。而如果新站一味要求全面兼容,会导致成本增加(随着功能多少,成本倍率增加)。为了抓紧时间,就不停地加班再加班,又导致员工产生抵触情绪,工作效率降低,人员流动开始频繁,新员工又要接手开发一半的项目。这样成本不停地在累加。最终不少项目根本没上线就失败了。

2. 用户选择

一般可以分为两种用户:高质量用户和低质量用户。所谓高质量用户,就是为了一款最新的3D游戏去升级一块发烧级的显卡,或直接换一台整机。所谓低质量用户,就是发现不能玩最新的3D游戏,就放弃了,去玩"植物大战僵尸"解解馋算了。在用户选择上有一个很好的案例,就是移动互联网。网易和腾讯在它们的新闻应用上,几乎兼容了所有的手机平台,比如IOS、安卓、黑莓、塞班等,因为新闻应用的核心在新闻,而新闻的用户基数巨大,需要兼顾高质量用户和低质量用户。而腾讯在IOS上的几十个应用,除了新闻、QQ、浏览器,其他的基本都只有IOS和安卓,在塞班和黑莓及其他应用上就没有了。

所以,你的应用核心是哪方面?兼容的成本有多大?会不会导致成本控制问题?用户选择尤为重要,放弃低质量用户也是一种成本控制。在用户基数庞大的项目上,放弃低质量用户就有点愚笨,比如某个新闻站有1亿用户,2 000万为使用低版本浏览器的低质量用户,而面对2 000万用户,你兼容它或单独为2 000万用户做个低版本服务,成本虽然可能还是3倍,但从庞大的用户基数来看,这种成本又非常低廉。而你的用户基数只有1 000人,而低质量用户有50人,那么为了这50人去做兼容,那么3倍的成本就变得非常昂贵。

3. 项目的侧重点

你的项目重点在哪里?是为了看新闻?是为了宣传线下产品?那么你其实有必要兼容低版本浏览器。首先这种类型的站不需要太好的用户体验,不需要太多的交互操作,只是看,而兼容的成本比较低,并且核心在新闻或产品!但如果你的项目有大量的交互、大量的操作,比如全球最大的社交网已经不兼容IE6、IE7,最大的微博也不再兼容IE6、IE7,就是这个原因。所以,项目并不是一味地全面兼容,或者全面不兼容,主要看你的项目侧重点在哪里!

4. 用户体验

如果你的项目在兼容低版本浏览器成本巨大,比如社交网,有大量的JS和AJAX操作。那么兼容IE6、IE7的成本确实很高,如果兼容,用户体验就会很差。兼容有两种:一

种是高版本浏览器用性能好,体验好的模式;低版本的自动切换到兼容模式。另一种就是,不管高版本或低版本都用统一的兼容模式。这两种成本都很高。用户体验好的模式,能增加用户黏度,增加付费潜在用户,而用户体验差的总是被用户归纳为心目中的备胎(所谓备胎就是实在没有了才去访问,如果有,很容易被抛弃)。

5. 数据支持

如果对某一种类型的网站项目有一定的研究,那么手头必须有支持的数据分析。有数据分析可以更好地进行成本控制,更有魄力地解决高质量用户和低质量用户的取舍。

6. 教育用户

很多项目可能是有固定客户群,或者使用该项目人员质量普遍较高。那么,面对零星一点的低质量用户,我们不能再去迎合他。因为迎合他,就无法用高质量用户体验去粘住忠实用户,同时也不能获取低质量用户的芳心。所以,我们应有的策略是:牢牢把握住高质量的忠诚用户,做到他们心目中的第一;教育那部分低质量用户(比如企业级开发项目,可以直接做企业培训,安装高版本浏览器等。互联网项目,就给出提示安装高版本浏览器即可)。那么一部分低质量用户被拉拢过来,还有一小撮死性不改的用户就只有放弃。切不可捡了芝麻丢了西瓜,不要贪大求全。

7. 经验之谈

以上我们讨论了是否需要兼容 IE6 或 IE7、IE8,结论就是必须根据实际情况,即成本情况、人员情况、用户情况和项目本身类型情况来制定,没有一刀切的兼容或不兼容。

七、下载及运行 JQuery

目前最新的版本,是 JQuery1. 10. 1 和 JQuery2. 0. 2,我们下载开发版,可以顺便读一读源代码。如果你需要引用到你线上的项目,就必须使用压缩版,去掉注释和空白,使容量最小。

本课程使用的软件是:Nodepad++使用测试的浏览器为:Firefox3. 6. 8、Firefox21+、Chrome、IE6/7/8/9、Opera 和 Safari。

使用的版本为:JQuery1. 10. 1 和 JQuery2. 02。

使用的 html 版本为:xhtml1. 0,在必要的时候将会使用 html5。

使用的调试工具为:Firefox 下的 firebug。

测试代码

```
//单击按钮弹窗
$(function () {
$('input').click(function () {
    alert('第一个 JQuery 程序!');
});
});
```

第二章
基础核心

教学要点：

1. 代码风格；
2. 加载模式；
3. 对象互换；
4. 多个库之间的冲突。

教学重点：

1. 熟悉 JQuery 的代码风格；
2. 了解 JQuery 的加载模式；
3. 对象互换；
4. 多个库之间的冲突。

教学难点：

对象互换、多个库之间的冲突。

开篇：本节课我们简单地介绍一下 JQuery 一些核心的问题，这些问题为后续课程展开提供了帮助。对于 JavaScript 课程已经学完的同学，这些概念会非常清晰，而对于 JavaScript 薄弱的同学可能会有一些模糊，但不必太担心，后续会慢慢展开。而对于完全没有 JavaScript 基础的同学，就无法学习了。

一、JQuery 代码风格

在 JQuery 程序中，不管是页面元素的选择还是内置的功能函数，都是从美元符号“$”来起始的。而这个“$”就是 JQuery 当中最重要且独有的对象，所以我们在页面元素选择或执行功能函数的时候可以这么写：

```
$(function(){});//执行一个匿名函数
$('#box');//进行执行的 ID 元素选择
$('#box').css('color','red');//执行功能函数
```

由于 $ 本身就是 JQuery 对象的缩写形式，那么也就是说上面的三段代码也可以写成如下形式：

JQuery(function ( ) {});

JQuery('#box');

JQuery('#box').css('color','red')。

在执行功能函数的时候,我们发现.css( )这个功能函数并不是直接被“$”或 JQuery 对象调用执行的,而是先获取元素后,返回某个对象再调用.css( )这个功能函数。那么也就是说,这个返回的对象其实也就是 JQuery 对象。

$( ).css('color', 'red'); //理论上合法,但实际上缺少元素而报错。

值得一提的是,执行了.css( )这个功能函数后,最终返回的还是 JQuery 对象,那么也就是说,JQuery 的代码模式是采用的连缀方式,可以不停地连续调用功能函数。

$('#box').css('color', 'red').css('font-size', '50px'); //连缀

JQuery 中代码注释和 JavaScript 是保持一致的,有两种最常用的注释:单行使用“//...”;多行使用“/* ... */”。// $('#box').css('color', 'red')。

二、加载模式

我们在之前的代码一直在使用 $(function ( ) {});这段代码进行首尾包裹,那么为什么必须要包裹这段代码呢? 原因是 JQuery 库文件是在 body 元素之前加载的,我们必须等待所有的 DOM 元素加载后,延迟支持 DOM 操作,否则就无法获取到。在延迟等待加载,JavaScript 提供了一个事件为 load。其方法如下:

window.onload = function ( ) {}; //JavaScript 等待加载。

$(document).ready(function ( ) {}); //JQuery 等待加载。

表 2-1　　onload 和 ready 的区别

	window.onload	$(document).ready( )
执行时机	必须等待网页全部加载完毕(包括图片等),然后再执行包裹代码。	只需要等待网页中的 DOM 结构加载完毕,就能执行包裹的代码。
简写方案	无	$(function( ){});

在实际应用中,我们都很少直接去使用 window.onload,因为它需要等待图片之类的大型元素加载完毕后才能执行 JS 代码。所以,最头疼的就是在网速较慢的情况下,页面已经全面展开,图片还在缓慢加载,这时页面上的 JS 交互功能全部处在假死状态。

三、对象互换

JQuery 对象虽然是 JQuery 库独有的对象,但它也是通过 JavaScript 进行封装而来的。我们可以直接输出来得到它的信息。

alert($); //JQuery 对象方法内部。

alert($ ( )); //JQuery 对象返回的对象还是 JQuery。

alert($ ('#box')); //包裹 ID 元素返回的对象还是 JQuery。

从上面三组代码我们发现:只要使用了包裹后,最终返回的都是 JQuery 对象。这样的好处显而易见,就是可以连缀处理。但有时我们也需要返回原生的 DOM 对象,比如:

alert(document.getElementById('box')); //[object HTMLDivElement]

JQuery 想要达到获取原生的 DOM 对象,可以这么处理:

alert($ ('#box').get(0)); //ID 元素的第一个原生 DOM

从上面 get(0)可以看出,JQuery 是可以进行批量处理 DOM 的,这样可以在很多需要循环遍历的处理上更加得心应手。

四、多个库之间冲突的解决

当一个项目中引入多个第三方库的时候,由于没有命名空间的约束(命名空间就好比同一个目录下的文件夹一样,名字相同就会产生冲突),库与库之间发生冲突在所难免。

那么,既然有冲突的问题,为什么要使用多个库呢? 原因是 JQuery 只不过是 DOM 操作为主的库,方便我们日常 Web 开发。但有时我们的项目有更多特殊的功能需要引入其他的库,比如用户界面 UI 方面的库,游戏引擎方面的库等一系列。

而很多库,比如 prototype 和 Base 库,都使用“$”作为基准起始符,如果想和 JQuery 共容有两种方法:

(1)将 JQuery 库在 Base 库之前引入,那么“$”的所有权就归 Base 库所有,而 JQuery 可以直接用 JQuery 对象调用,或者创建一个“$ $”符给 JQuery 使用。

```
var $ $ = JQuery; //创建一个 $ $ 的 JQuery 对象
$(function () { //这是 Base 的 $
    alert( $('#box').ge(0)); //这是 Base 的 $
    alert( $ $('#box').width()); //这是 JQuery 的 $ $
});
```

(2)如果将 JQuery 库在 Base 库之后引入,那么“$”的所有权就归 JQuery 库所有,而 Base 库将会冲突而失去作用。这里,JQuery 提供了一个方法:

```
JQuery.noConflict(); //将 $ 符所有权剔除
var $ $ = JQuery;
$(function () {
    alert( $('#box').ge(0));
    alert( $ $('#box').width());
});
```

第三章
常规选择器

教学要点：

1. 简单选择器；
2. 进阶选择器；
3. 高级选择器。

教学重点：

1. 简单选择器；
2. 进阶选择器；
3. 高级选择器。

教学难点：

理解什么是选择器，多种选择器组合使用。

开篇：JQuery 最核心的组成部分就是选择器引擎。它继承了 CSS 的语法，可以对 DOM 元素的标签名、属性名、状态等进行快速、准确的选择，并且不必担心浏览器的兼容性。JQuery 选择器除实现了 CSS1～CSS3 的大部分规则之外，还实现了一些自定义的选择器，用于各种特殊状态的选择。备注：学习本课程必须有(X)html+CSS 基础。

一、简单选择器

在使用 JQuery 选择器时，我们首先必须使用“$()”函数来包装我们的 CSS 规则。而 CSS 规则作为参数传递到 JQuery 对象内部后，再返回包含页面中对应元素的 JQuery 对象。随后，我们就可以对这个获取到的 DOM 节点进行行为操作了。

```
#box { //使用 ID 选择器的 CSS 规则：
    color:red; //将 ID 为 box 的元素字体颜色变红
}
```

在 JQuery 选择器里，我们使用如下方式来获取同样的结果：

```
$('#box').css('color', 'red'); //获取 DOM 节点对象，并添加行为。
```

那么除了 ID 选择器之外，还有两种基本的选择器：元素标签名和类(class)。

表 3-1

选择器	CSS 模式	JQuery 模式	描述
元素标签名	Div{}	$('div')	获取所有 div 元素的 DOM 对象
ID	#box {}	$('#box')	获取一个 ID 为 box 元素的 DOM 对象
类(class)	.box{}	$('.box')	获取所有 class 为 box 的所有 DOM 对象

```
$('div').css('color', 'red'); //元素选择器,返回多个元素。
$('#box').css('color', 'red'); //ID 选择器,返回单个元素。
$('.box').css('color', 'red'); //类(class)选择器,返回多个元素。
```

为了证明 ID 返回的是单个元素,而元素标签名和类(class)返回的是多个,我们可以采用 JQuery 核心自带的一个属性 length 或 size()方法来查看返回的元素个数。

```
alert( $('div').size()); //3 个。
alert( $('#box').size()); //1 个,后面两个失明了。
alert( $('.box').size()); //3 个。
```

同理,你也可以直接使用 JQuery 核心属性来操作:

```
alert( $('#box').length); //1 个,后面失明了。
```

警告:有个问题特别要注意,ID 在页面只允许出现一次,我们一般都是要求开发者要遵守和保持这个规则。但如果你在页面中出现三次,并且在 CSS 使用样式,那么这三个元素还会执行效果。但如果你想要 JQuery 这么去做,那么就会遇到失明的问题。所以,开发者必须养成良好的遵守习惯,在一个页面仅使用一个 ID。

```
$('#box').css('color', 'red'); //只有第一个 ID 变红,后面两个 ID 失明了。
```

JQuery 选择器的写法与 CSS 选择器十分类似,只不过它们的功能不同。CSS 找到元素后添加的是单一的样式,而 JQuery 则添加的是动作行为。最重要的一点是:CSS 在添加样式的时候,高级选择器会对部分浏览器不兼容,而 JQuery 选择器在添加 CSS 样式的时候却不必为此烦恼。

```
#box > p {//CSS 子选择器,IE6 不支持
    color:red;
}
$('#box > p').css('color','red'); //JQuery 子选择器,兼容了 IE6。
```

JQuery 选择器支持 CSS1、CSS2 的全部规则,支持 CSS3 部分实用的规则,同时它还有少量独有的规则。所以,对于已经掌握 CSS 的开发人员,学习 JQuery 选择器几乎是零成本。而 JQuery 选择器在获取节点对象的时候不但简单,还内置了容错功能,这样避免像 JavaScript 那样每次对节点的获取需要进行有效判断。

```
$('#pox').css('color', 'red'); //不存在 ID 为 pox 的元素,也不报错。
document.getElementById('pox').style.color = 'red'; //报错了。
```

因为 JQuery 内部进行了判断,而原生的 DOM 节点获取方法并没有进行判断,所以导致了一个错误,原生方法可以这么判断解决这个问题:

```
if (document.getElementById('pox')) { //先判断是否存在这个对象
    document.getElementById('pox').style.color = 'red';
```

}

那么对于缺失不存在的元素，我们使用 JQuery 调用的话，怎么去判断是否存在呢？因为本身返回的是 JQuery 对象，可能会导致不存在元素存在与否，都会返回 true。

```
if ( $ ('#pox ').length > 0) { //判断元素包含数量即可
    $ ('#pox ').css(' color ', ' red ');
}
```

除了这种方式之外，还可以用转换为 DOM 对象的方式来判断，例如：

if ($ ('#pox ').get(0)) {}或 if ($ ('#pox ')[0]) {} //通过数组下标也可以获取 DOM 对象。

二、进阶选择器

在简单选择器中，我们了解了最基本的三种选择器：元素标签名、ID 和类(class)。那么在基础选择器外，还有一些进阶和高级的选择器方便我们进行更精准的选择元素。

表 3-2

选择器	CSS 模式	JQuery 模式	描述
群组选择器	span,em,.box {}	$ ('span,em,.box ')	获取多个选择器的 DOM 对象
后代选择器	ul li a {}	$ ('ul li a ')	获取追溯到的多个 DOM 对象
通配选择器	* {}	$ ('* ')	获取所有元素标签名的 DOM 对象

```
//群组选择器
span, em, .box { //多种选择器添加红色字体
    color:red;
}
$ ('span, em, .box ').css(' color ', ' red '); //群组选择器 JQuery 方式
//后代选择器
ul li a { //层层追溯到的元素添加红色字体
    color:red;
}
$ ('ul li a ').css(' color ', ' red '); //群组选择器 JQuery 方式
//通配选择器
* { //页面所有元素都添加红色字体
    color:red;
}
$ ('* ').css(' color ', ' red '); //通配选择器
```

目前介绍的六种选择器，在实际应用中，我们可以灵活地搭配，使得选择器更加精准和快速：

```
$ ('#box p, ul li  * ').css(' color ', ' red '); //组合了多种选择器。
```

警告：在实际使用上，通配选择器一般用得并不多，尤其是在大通配上，比如：$ ('* ')。这种使用方法效率很低，影响性能，建议尽可能少用。还有一种选择器，可以在 ID 和类(class)中指明元素前缀，比如：