

网站开发路线图

犀利开发 jQuery 内核详解与实践

- 第一部破解jQuery核心技术的先锋图书
- 生动诠释jQuery与JavaScript技术执行效率和实现方法之异同
- 使用通俗易懂的语言深度剖析jQuery框架设计模式和选择器实现原理
- 潜心研究jQuery多年，耗时近一年，国内著名原创IT作者又一力作

CD-ROM

- 本书实例素材和源代码
- CSS参考手册
- jQuery参考手册
- Ajax参考手册
- JavaScript参考手册
- HTML参考手册

清华大学出版社

朱印宏 编 著



网站开发线路图

犀利开发 ——jQuery 内核详解与实践

朱印宏 编 著

清华大学出版社

北 京

内 容 简 介

本书循序渐进地讲解了jQuery高效开发的方法和技巧,内容包括jQuery框架的设计模式和思路、Sizzle选择器的构成和工作机制、DOM文档操作、事件处理、动画设计、Ajax异步通信、插件扩展和辅助工具等。

执行效率是JavaScript脚本的第一要务,本书在详细讲解jQuery基础知识和技巧用法的同时,重点讲解了如何提高jQuery工作效率,如何混合使用jQuery和JavaScript进行高效开发。此外,本书还深入剖析了jQuery框架的设计模式和选择器的工作机制。全书理论结合实践,通过大量生动的示例帮助读者快速上手。

本书适合Web开发人员阅读和参考,同时也适合广大网页制作和设计爱好者阅读和学习。

本书封面贴有清华大学出版社防伪标签,无标签者不得销售。

版权所有,侵权必究。侵权举报电话:010-62782989 13701121933

图书在版编目(CIP)数据

犀利开发——jQuery内核详解与实践/朱印宏编著. —北京:清华大学出版社,2010.8
(网站开发路线图)

ISBN 978-7-302-23111-0

I. ①犀… II. ①朱… III. ①JAVA语言—程序设计 IV. ①TP312

中国版本图书馆CIP数据核字(2010)第144391号

责任编辑:应 勤 张 瑜

封面设计:杨玉兰

责任校对:王 晖

责任印制:孟凡玉

出版发行:清华大学出版社

地 址:北京清华大学学研大厦A座

<http://www.tup.com.cn>

邮 编:100084

社 总 机:010-62770175

邮 购:010-62786544

投稿与读者服务:010-62776969, c-service@tup.tsinghua.edu.cn

质 量 反 馈:010-62772015, zhiliang@tup.tsinghua.edu.cn

印 刷 者:北京密云胶印厂

装 订 者:三河市新茂装订有限公司

经 销:全国新华书店

开 本:185×260 印 张:26.25 字 数:620千字

附光盘1张

版 次:2010年8月第1版 印 次:2010年8月第1次印刷

印 数:1~4000

定 价:49.00元

产品编号:034795-01

前言

\$表示美元符号，就是这个小符号让无数网页设计师和 Web 开发人员为之着迷，甚至折腰。实际上在 jQuery 诞生之前，已经有很多语言和框架在使用\$符号了，如 Prototype 和 DWR 等。使用\$符号代替 document.getElementById()函数应该是 DOM 访问中最简捷的操作方法了。没错，jQuery 是后起之秀，是跟风者，但正是因为 jQuery，才让更多的读者、学者、设计师和开发人员把\$符号铭记于心。

jQuery 确实有其无穷的魅力，我也是在这个魅力的潮流中开始研究 jQuery 并利用它进行开发的。从 2006 年到现在，不到 4 年的时间，jQuery 竟然能够聚焦全球亿万开发人员的眼球，实在是 Web 开发史上的一大奇迹。这又是为什么呢？

从 jQuery(JavaScript+Query)名称也可以看出，使用 CSS+XPath 选择器查询页面元素是该框架赖以起家的绝活。提及脚本选择器，最早可以追溯到 Dean Edwards 的 cssQuery 和 Simon Willison 的 getElementsBySelector，但是在早期的 Web 编程环境中，面对恶劣的浏览器环境和粗糙的 Web 应用，这些选择器的实现只能被当作内部实验，没有应用市场。而到了 2005 年，互联网已经进入 Web 2.0 和 Ajax 的新时代，1984 年出生的天才少年 John Resig 在先行者的启发下开始探索 JavaScript 选择器技术，并在一年后正式发布了 jQuery 1.0，迅速红遍全球，流行程度堪比摇滚巨星。

人都有一种惰性，或者说是习惯吧，用惯了 Java 作为开发语言，就不习惯.NET 的编程环境。同样，如果你习惯了 CSS 的用法及其选择网页标签的方式，再来学习 jQuery 的选择器你会倍感亲切，甚至于入迷。jQuery 完全仿制了 CSS 选择器的设计思路和用法，让广大的初学者和整日为获取网页节点而搔首的开发人员快速上手。

jQuery 灵巧、便捷，并能够按着人的思维去编写代码，这实在是一件很令人兴奋的事情。难怪但凡接触过 jQuery 的初学者都会对它爱不释手，也难怪 jQuery 在众多 JavaScript 流行框架中能快速杀出一条血路，并得到广大草根编程人员的喜爱。

很多读者和开发人员接触 jQuery 之后，都会身不由己地迷恋于 jQuery 的优雅和便捷，错误地认为它就是一种高效的 JavaScript 编程方法，于是在前端开发中毫无节制地使用 jQuery 匹配和操作网页元素。这种行为本无可厚非，但是我們不应忘记，jQuery 仅是 JavaScript 功能的外包装。因此在学习和使用 jQuery 的过程中，读者应该首先树立以下两种意识。

1. 执行效率是编写脚本的第一要务

jQuery 经过多个版本的磨炼，特别是新的独立开发的 Sizzle 选择器引擎，使 jQuery 匹配元素的速度达到了新的高度，可以毫不夸张地说 jQuery 已是业界第一。但是，无论 Sizzle 选

择器的速度如何优化,它都是在 JavaScript 原生方法的基础上进行打包的,这种打包过程虽然节省了用户编写代码的工作量,但是却增加了代码的执行速度。因此,在可能的条件下,建议读者不要完全抛弃 JavaScript 原生的选择器方法,适当混合使用 jQuery、DOM 和 JavaScript 能够提升程序的执行效率。例如,offset(获取页面上的各种尺寸和位置数据)、创建和插入 DOM 节点的方法(如 append, before)都是 JQuery 速度的瓶颈,在必要的情况下,读者完全不用它们,而直接使用 DOM 原生方法会更加高效。正是由于这个原因,本书在讲解 JQuery 选择器及其各种 DOM 元素的操作过程中,都会对比 JavaScript 设计相同效果或功能的实现方法,让读者能够在学习后辈技术的同时,不要忘记老一辈技术的实现途径。很多初学者由于习惯了 JQuery 开发方式,竟然不会使用 JavaScript 进行开发,这实在是件很不幸的事情。

2. 理解 JQuery 设计模式和工作机制是学习和提升 JQuery 开发水平的关键

JQuery 框架的源代码虽然仅有 4000 多行,但是它设计得精巧和复杂,非一般读者所能够读懂,但是即便如此,对于每一位初学者来说,我们都应该尽力理解框架中每一行代码的功能,以及各行代码之间的逻辑关系,这对于正确和高效地使用 JQuery 是至关重要的。同时当程序发生错误时,我们也能够通过浏览器简单的错误提示快速找到症结所在。正所谓知其然,也应知其所以然。本书在讲解 JQuery 框架的使用方法的同时,把更多的精力投入到 JQuery 框架的内部,揭秘 JQuery 框架的内部逻辑关系。相信读者在阅读本书的过程中,会对 JQuery 内核有更深入的认识。

关于本书

本书从前台开发人员的角度进行选材,主要研究 JQuery 框架的设计模式、实现机制和 JQuery 框架的一般用法、扩展应用及其实战演练。全书共分为 10 章,简单介绍如下。

第 1 章 JQuery 起步,重点介绍 JavaScript 及其框架发展概述,如何使用 JQuery,如何编写自己的第一个示例,以及 JQuery 框架的基本特性。

第 2 章 JQuery 解密技术,重点介绍 JQuery 框架的原型设计思路和模式,详细分解选择器接口和选择器引擎 Sizzle 的结构和工作机制,另外还详细讲解 JQuery 对象的数据结构和基本操作。

第 3 章 高效选择的技巧与原理,重点讲解 JQuery 的 CSS 选择器用法,以及这些选择器类型的实现方法。

第 4 章 文档对象的操作及其高效实践,重点讲解 JQuery 如何操作文档结构和元素,如何控制页面 CSS 的样式。同时,还就 JavaScript 实现方法进行详细讲解,以方便读者横向比较两者实现途径的异同,从而更加深刻地理解 JQuery 的操作机制。

第 5 章 事件封装机制与解析,重点讲解 JQuery 如何封装 JavaScript 的事件处理机制,并介绍如何使用 JavaScript 来定义 JQuery 的事件处理方法。

第 6 章 动画效果设计及其高效实践,重点讲解 JQuery 在设计动画方面的优势,以及如何使用 JQuery 设计常规动画方法,并就 JavaScript 实现相同动画效果进行讲解。

第 7 章 Ajax 异步通信高效实践, 讲解 Ajax 异步通信的原理, 并根据这个原理解析 jQuery 是如何实现这些异步通信的方法。

第 8 章 高效开发和使用插件, 重点讲解如何扩展 jQuery 公共函数、jQuery 对象方法和 jQuery 选择器, 并通过几个典型案例演示这些扩展方法。

第 9 章 jQuery 辅助工具, 重点讲解 jQuery 公共函数工具, 以及附加的缓存数据和数据队列处理, 还介绍一些辅助工具的使用。

第 10 章 使用 jQuery 打造 Ajax 异步交互式动态网站, 通过一个动态演示网站的架设帮助读者认识 jQuery 在网站开发中的角色。

读者对象

本书适合网页制作的初学者, 广大网页制设计师和前端技术人员。

读者在阅读本书之前, 应该初步掌握 HTML、CSS 和 JavaScript 语言的基础知识, 特别是能够初步使用 JavaScript。

本书约定

(1) 考虑到版面限制, 展示出来的代码仅包含 JavaScript 脚本和必要的结构代码。读者在学习测试时, 应该把这些 JavaScript 脚本输入到网页。例如:

```
<script src="jquery-1.3.2.js" type="text/javascript" ></script>
<script type="text/javascript" >
$(function(){
    $("div").after("<p>最后一段文本</p>");
    $("div").before("<p>第一段文本</p>");
});
</script>

<div>
    <p>段落文本</p>
</div>
```

则应该按如下代码格式输入上面示例代码, 并另存为 html 网页文件, 然后在浏览器中测试即可。

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title></title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<script src="jquery-1.3.2.js" type="text/javascript" ></script>
<script type="text/javascript" >
$(function(){
    $("div").after("<p>最后一段文本</p>");
```

```

    $("div").before("<p>第一段文本</p>");
  })
</script>
</head>
<body>
<div>
  <p>段落文本</p>
</div>
</body>
</html>

```

(2) 本书以 jQuery 1.3.2 版本为基础进行介绍和演示,能够兼容 jQuery 3.0 以后的任何版本。

(3) 在默认情况下, jQuery 1.3.2 库文件都会自动导入文档,如果没有特别说明,我们会在示例中省略以下命令。注: jquery-1.3.2.js 存放在 images 文件夹中。

```
<script src="images/jquery-1.3.2.js" type="text/javascript" ></script>
```

(4) 在默认情况下,我们都会使用 jQuery 的别名 \$ 来表示 jQuery 名字空间,同时直接把调用的函数放在 \$() 函数中,该函数实际上是 \$("document").ready() 方法的简写,它相当于 JavaScript 中的 window.onload = function () {} 事件处理函数。例如:

```

<script type="text/javascript" >
$(function(){
    $("div").after("<p>最后一段文本</p>");
    $("div").before("<p>第一段文本</p>");
})
</script>

```

实际上上面的示例代码应该是下面的形式。

```

<script type="text/javascript" >
$( "document" ).ready (function(){
    $("div").after("<p>最后一段文本</p>");
    $("div").before("<p>第一段文本</p>");
})
</script>

```

它相当于 JavaScript 中下面的示例代码,当然两者之间还存在细微的区别,详细讲解请参阅本书第 1、2 章中的讲解。

```

<script type="text/javascript" >
window.onload =function(){
    $("div").after("<p>最后一段文本</p>");
    $("div").before("<p>第一段文本</p>");
}
</script>

```

(5) 由于 jQuery 与 JavaScript 变量之间存在区别,默认情况下当定义 jQuery 对象变量时,应在该变量的前面附加一个 \$ 前缀,以便与 JavaScript 变量进行区分。例如:

```

<script type="text/javascript" >
var $div = $("div"); //jQuery 对象变量

```

```
var div = document.getElementsByTagName("div"); //JavaScript 对象变量  
</script>
```

关于作者

本书由朱印宏编写，参与资料整理及编写的还有袁衍明、常才英、袁祚寿、袁衍明、张敏、袁江、田明学、唐荣华、毛荣辉、卢敬孝、刘玉凤、李坤伟、旷晓军、陈万林、陈锐、钱佩林、苏敬波、冉东林、杨龙贵、张炜、王慧明、涂怀清、卢国才、苏恢定、司成向、胡体清、陈宗亮、徐清银、周秀成、颜昌学、王幼平、冉原洲、李经键、胡厚成等，在此对大家的辛勤工作表示衷心的感谢。

由于作者水平有限，书中难免会有疏漏之处，恳请广大读者提出宝贵意见，也希望广大读者能够把个人的意见、建议或问题发送到 zhuyinhong@css8.cn，以便与作者进行交流。

编 者

目 录

第 1 章 jQuery 起步	1	2.2.4 生长——跨域访问	31
1.1 认识 jQuery	2	2.2.5 成熟——选择器	32
1.1.1 JavaScript 及其库	2	2.2.6 延续——迭代器	33
1.1.2 选用 jQuery 框架的理由	6	2.2.7 延续——功能扩展	36
1.2 jQuery 初步体验	7	2.2.8 延续——参数处理	38
1.2.1 安装 jQuery 库	7	2.2.9 涅槃——名字空间	39
1.2.2 导入 jQuery 库	8	2.3 破解 jQuery 选择器接口	41
1.2.3 编写 jQuery 代码	9	2.3.1 简单但很复杂的黑洞	42
1.2.4 区分 jQuery 对象和 DOM 对象	10	2.3.2 盘根错节的逻辑关系	43
1.2.5 jQuery 对象和 DOM 对象的 相互转换	11	2.3.3 jQuery 构造器	44
1.2.6 ready 事件和 load 事件比较	13	2.3.4 生成 DOM 元素	47
1.3 jQuery 核心特性	15	2.3.5 引用 DOM 元素	50
1.3.1 jQuery 构造函数	16	2.4 解析 jQuery 选择器引擎 Sizzle	53
1.3.2 jQuery 链式语法	17	2.4.1 回顾 CSS 的选择器	54
1.3.3 jQuery 选择器	19	2.4.2 解析 jQuery 选择器引擎的 设计思路	54
1.3.4 jQuery 扩展性	20	2.4.3 选择器和过滤器	55
第 2 章 jQuery 技术解密	23	2.4.4 Sizzle 引擎结构	56
2.1 jQuery 框架设计概述	24	2.4.5 Sizzle 构造器	59
2.1.1 设计目标	24	2.4.6 Sizzle 选择器	63
2.1.2 目标实现	25	2.4.7 Sizzle 过滤器	65
2.2 jQuery 原型技术分解	26	2.4.8 jQuery 选择器应用优化	68
2.2.1 起源——原型继承	26	2.5 类数组	70
2.2.2 生命——返回实例	27	2.5.1 定义类数组	70
2.2.3 学步——分隔作用域	29	2.5.2 操作类数组	72

第 3 章 高效选择的技巧与原理.....79	第 4 章 文档对象的操作及其 高效实践..... 125
3.1 选择器是什么.....80	4.1 DOM 标准..... 126
3.1.1 从 CSS 选择器说起.....80	4.1.1 分解 DOM..... 126
3.1.2 jQuery 盗了谁的版.....83	4.1.2 HTML DOM..... 126
3.1.3 认识 cssQuery 选择器.....84	4.1.3 DOM Core..... 127
3.1.4 使用 cssQuery 选择器.....85	4.1.4 DOM 文档树..... 128
3.1.5 初步接触 jQuery 选择器.....87	4.2 创建节点..... 129
3.2 简单选择器.....89	4.2.1 创建元素..... 129
3.2.1 选择指定 ID 元素.....89	4.2.2 创建文本..... 131
3.2.2 选择指定类型元素.....93	4.2.3 创建属性..... 132
3.2.3 选择指定类元素.....95	4.3 插入元素..... 134
3.2.4 选择所有元素及其优化.....98	4.3.1 jQuery 实现..... 135
3.2.5 选择多组元素及其实现.....99	4.3.2 JavaScript 实现..... 138
3.3 关系选择器.....100	4.3.3 自定义 JavaScript 扩展 DOM 功能函数..... 139
3.3.1 层级选择器.....100	4.3.4 使用 JavaScript 自定义 appendTo()和 prependTo() 方法..... 142
3.3.2 层级选择器的实现方法.....102	4.3.5 使用 JavaScript 自定义 after()和 before()方法..... 143
3.3.3 子元素选择器.....105	4.3.6 使用 JavaScript 自定义 insertAfter()和 insertBefore() 方法..... 144
3.3.4 子元素选择器的实现方法.....106	4.4 删除元素..... 144
3.4 过滤选择器.....107	4.4.1 jQuery 实现..... 145
3.4.1 定位过滤器.....108	4.4.2 JavaScript 实现..... 146
3.4.2 定位过滤器的实现方法.....110	4.4.3 使用 JavaScript 自定义 empty()方法..... 146
3.4.3 内容过滤器.....111	4.5 复制元素..... 147
3.4.4 内容过滤器的实现方法.....112	4.5.1 jQuery 实现..... 147
3.4.5 可见过滤器.....113	4.5.2 JavaScript 实现..... 148
3.4.6 可见选择器的实现方法.....114	4.6 替换元素..... 148
3.5 属性选择器.....114	
3.5.1 使用属性选择器.....114	
3.5.2 属性选择器的实现方法.....116	
3.6 表单选择器.....118	
3.6.1 基本表单选择器.....118	
3.6.2 高级表单选择器.....120	
3.6.3 表单选择器的实现方法.....121	

4.6.1	jQuery 实现	149	4.12.2	JavaScript 实现的元素遍历 方法	186
4.6.2	JavaScript 实现	149	第 5 章 事件封装机制与解析		
4.6.3	使用 JavaScript 自定义 replaceWith()和 replaceAll() 方法	150	5.1	事件模型	190
4.7	包裹元素	151	5.1.1	0 级事件模型	190
4.7.1	jQuery 实现	151	5.1.2	事件模型中的 Event 对象	190
4.7.2	使用 JavaScript 自定义 wrap()、wrapAll()和 wrapInner()方法	152	5.1.3	事件模型中的冒泡现象	192
4.8	操作属性	153	5.1.4	事件流控制与 默认事件动作	194
4.8.1	设置属性	153	5.1.5	2 级 DOM 标准事件模型	194
4.8.2	获取属性	154	5.1.6	IE 事件模型	198
4.8.3	删除属性	155	5.2	jQuery 事件模型	200
4.9	操作类样式	156	5.2.1	绑定事件	201
4.9.1	追加样式	156	5.2.2	注销事件	203
4.9.2	移出样式	157	5.2.3	jQuery 事件模型中的 Event 对象	204
4.9.3	切换样式	159	5.2.4	jQuery 事件触发	205
4.9.4	判断样式	162	5.2.5	jQuery 事件切换	207
4.10	操作 HTML、文本和值	162	5.2.6	jQuery 事件委派	211
4.10.1	读写 HTML 字符串	163	5.2.7	jQuery 事件命名空间	213
4.10.2	读写文本内容	164	5.2.8	jQuery 的多事件绑定	214
4.10.3	读写表单值	165	5.2.9	jQuery 自定义事件	216
4.11	操作样式表	167	5.3	jQuery 页面初始化	216
4.11.1	通用 CSS 样式读写方法	167	5.3.1	使用 jQuery 的 ready()方法	216
4.11.2	绝对偏移位置	174	5.3.2	ready 事件的触发时机	218
4.11.3	相对偏移位置	176	5.3.3	初始化事件的多次调用	219
4.11.4	扩展 DOM 操作函数	179	5.3.4	使用 JavaScript 自定义 addLoadEvent()方法	220
4.11.5	元素的宽和高	182	5.4	使用 JavaScript 自定义 jQuery 事件方法	221
4.12	元素遍历操作	185	5.4.1	JavaScript 与 jQuery 的 执行效率比较	222
4.12.1	jQuery 实现的元素遍历 方法	185	5.4.2	自定义 ready()方法	223

5.4.3	自定义 bind() 方法.....	224	7.2.1	jQuery 实现.....	264
5.4.4	自定义 one() 方法.....	226	7.2.2	JavaScript 实现.....	265
第 6 章 动画效果设计及其高效实践..... 227			7.3 从 JavaScript 角度分析		
6.1	直接显示和隐藏.....	228	XMLHttpRequest 对象..... 266		
6.1.1	jQuery 实现显隐效果.....	228	7.3.1	XMLHttpRequest 对象成员和用法.....	266
6.1.2	JavaScript 实现显隐效果.....	229	7.3.2	建立异步连接.....	267
6.1.3	折叠效果.....	230	7.3.3	发送请求.....	268
6.1.4	树形结构.....	233	7.3.4	发送 GET 请求.....	269
6.1.5	Tab 选项卡.....	237	7.3.5	发送 POST 请求.....	270
6.1.6	显隐切换.....	239	7.3.6	跟踪响应状态.....	272
6.2	滑动显示和隐藏.....	241	7.3.7	获取响应信息.....	273
6.2.1	jQuery 实现的滑动显隐效果.....	241	7.4 从 jQuery 角度分析		
6.2.2	JavaScript 实现的滑动显示效果.....	242	XMLHttpRequest 对象..... 275		
6.2.3	JavaScript 实现的滑动隐藏效果.....	245	7.4.1	使用 GET 方式请求.....	276
6.2.4	jQuery 设计的滑动显隐切换.....	246	7.4.2	使用 POST 方式请求.....	278
6.3	渐隐和渐显.....	247	7.4.3	使用 ajax() 方法请求.....	279
6.3.1	jQuery 实现的渐隐渐显效果.....	247	7.4.4	跟踪响应状态.....	281
6.3.2	JavaScript 实现的渐显效果.....	249	7.4.5	载入网页文件.....	283
6.3.3	JavaScript 实现的渐隐效果.....	251	7.4.6	预设 Ajax 选项.....	285
6.4	自定义动画.....	252	7.4.7	预处理请求的字符串.....	286
6.4.1	jQuery 自定义动画.....	252	第 8 章 高效开发和使用插件..... 291		
6.4.2	使用 jQuery 停止动画.....	255	8.1 创建 jQuery 插件..... 292		
6.4.3	使用 jQuery 关闭动画.....	256	8.1.1	jQuery 插件的类型.....	292
6.4.4	使用 JavaScript 实现滚动动画.....	256	8.1.2	解析 jQuery 插件机制.....	292
第 7 章 Ajax 异步通信高效实践..... 261			8.1.3	创建 jQuery 全局函数.....	295
7.1	Ajax 应用准备.....	262	8.1.4	使用 jQuery.fn 对象创建 jQuery 对象方法.....	296
7.1.1	Ajax 应用利弊分析.....	262	8.1.5	使用 extend() 方法创建 jQuery 对象方法.....	299
7.1.2	安装虚拟服务器.....	263	8.1.6	创建自定义选择器.....	300
7.2	Ajax 应用的第一个示例.....	263	8.1.7	优化 jQuery 默认选择器.....	302
			8.1.8	封装 jQuery 插件.....	305

8.1.9 优化 jQuery 插件 ——开放公共参数	307	9.3 数组处理	350
8.1.10 优化 jQuery 插件 ——开放部分功能	309	9.3.1 检测数组	351
8.1.11 优化 jQuery 插件 ——保留插件隐私	310	9.3.2 遍历数组或集合对象	352
8.1.12 优化 jQuery 插件 ——非破坏性操作	312	9.3.3 转换为数组	354
8.1.13 优化 jQuery 插件 ——添加事件日志	314	9.3.4 过滤数组	356
8.1.14 创建 jQuery 插件应注意的 问题	318	9.3.5 映射数组	357
8.2 创建 jQuery 插件实战	320	9.3.6 合并数组	359
8.2.1 简化式插件设计	321	9.3.7 删除数组中的重复项	360
8.2.2 定宽输出插件	322	9.4 多库共存	361
8.2.3 Tab 选项卡插件	325	9.4.1 解决 \$ 名字冲突	361
8.3 jQuery UI 插件应用	331	9.4.2 解决 jQuery 名字冲突	363
8.3.1 如何使用外部插件	332	9.5 数据缓存	364
8.3.2 认识 UI 插件	335	9.5.1 jQuery 数据缓存的作用	364
8.3.3 调整大小	337	9.5.2 定义缓存数据	366
8.3.4 日期选择器	338	9.5.3 获取缓存数据	367
第 9 章 jQuery 辅助工具	341	9.5.4 删除缓存数据	368
9.1 检测浏览器特性	342	9.5.5 jQuery 数据缓存的 JavaScript 实现原理	368
9.1.1 jQuery 检测浏览器的类型	342	9.5.6 jQuery 数据缓存的 使用规范	371
9.1.2 JavaScript 检测浏览器 的类型	343	9.6 数据队列	372
9.1.3 更灵巧的浏览器检测方法	345	9.6.1 添加队列	372
9.1.4 检测浏览器的版本号	345	9.6.2 获取队列	374
9.1.5 检测浏览器的盒模型	346	9.6.3 替换队列	375
9.1.6 浏览器特性综合测试	347	9.6.4 删除队列函数	376
9.2 字符串处理	348	9.7 内核工具	377
9.2.1 修剪字符串	348	9.7.1 遍历 jQuery 对象	377
9.2.2 序列化字符串	349	9.7.2 遍历 jQuery 对象的 JavaScript 实现	378
		9.7.3 获取 jQuery 对象的 元素个数	379
		9.7.4 获取选择器和选择范围	380
		9.7.5 获取 jQuery 对象的元素	380

第 10 章 使用 jQuery 打造 Ajax 异步

交互式动态网站 383

10.1 案例背景介绍 384

10.2 网站设计思路 385

10.3 结构设计 385

10.4 样式设计 387

10.4.1 基本样式 387

10.4.2 主题皮肤样式 392

10.5 网站脚本设计 393

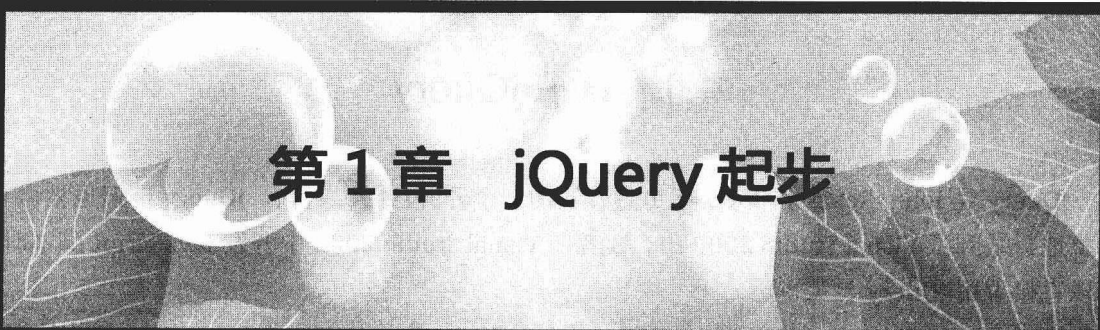
10.5.1 主题样式动态控制 393

10.5.2 导入外部数据 394

10.5.3 分类导航设计 395

10.5.4 缩微图显示 397

10.5.5 灯箱广告 400



第1章 jQuery 起步

jQuery 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。

jQuery 封装了 JavaScript 常用的功能代码，提供了一种简洁、快捷的 JavaScript 设计模式，优化了 HTML 文档操作、事件处理、动画设计和 Ajax 交互。可以说，jQuery 改变了用户编写 JavaScript 代码的方式。由于 jQuery 能够兼容 CSS 3，并且能兼容各种主流浏览器，如 IE 6.0+、FF 1.5+、Safari 2.0+ 和 Opera 9.0+等，因此它正被越来越多的开发人员喜爱和选用。

1.1 认识 jQuery

jQuery 是 John Resig 于 2006 年 1 月开发的一个开源项目。目前, jQuery 框架也被微软完整地封装到了 Visual Studio 2008 中, 读者在 Visual Studio 2008 中开发 JavaScript 脚本时, 可以直接调用 jQuery 框架。

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)四部分, 团队成员由核心库开发、UI 开发、插件开发人员, 以及网站设计、运营和推广人员组成。团队核心人物主要是 John Resig、Brandon Aaron 和 Jorn Zaefferer。

- jQuery 框架官网: <http://jquery.com>。
- jQuery 项目组官网: <http://jquery.org>。
- John Resig 个人网站: <http://ejohn.org>。

John Resig 目前供职于 Mozilla 公司, 专门负责开发 JavaScript 开发工具。这个小伙子出生于 1984 年 6 月 8 日, 居住在美国波士顿, 今年 26 岁, 也就是说 John Resig 在 22 岁时就已经独立开发出了 jQuery 技术框架。先来欣赏一下这个白白的小胖子, 如图 1.1 所示。



图 1.1 John Resig

1.1.1 JavaScript 及其库

JavaScript 最先是由 Netscape 公司开发的一种脚本语言, 用来在网页中设计交互、动态的效果。随后, 微软公司也推出了自己的 JavaScript 版本, 并将其命名为 JScript。后来, ECMA 标准化组织推出了 ECMAScript 标准, 并试图规范 JavaScript 语言的语法和用法。目前, 我们所介绍、学习和使用的 JavaScript 版本都是以 ECMAScript 标准为基础的。

由于 JavaScript 版本之间的差异性, 以及各大浏览器对于 JavaScript 和 DOM(Document Object Model, 文档对象模型)解析的不统一, 给开发人员带来了很大麻烦。正当 JavaScript 逐渐被开发人员弃用时, 一种基于 JavaScript 语言的 Ajax 技术横空出世, 重新唤起了人们对于 JavaScript 开发的热情。

为了简化 JavaScript 开发, 一些 JavaScript 代码库诞生了。这些代码库封装了很多预定义

的对象和实用函数，能够简化开发人员的工作，提高代码的执行效率。JavaScript 代码库的诞生，标识了真正的 Web 2.0 应用开发的到来，为富客户端开发奠定了基础。

随着代码库的不断演化和升级，技术框架逐渐变得逻辑严密。所谓框架，就是指一套包含工具、函数库、约定，以及尝试从常用任务中抽象出可以复用的通用模块，其目标是使设计师和开发人员把重点放在任务项目所特有的方面，避免重复开发。通常地讲，框架就是最常用的 JavaScript 框架和 Web 应用框架，当然，还有 CSS 框架。下面分别介绍几个比较流行的 JavaScript 技术框架。

1. Dojo

Dojo(<http://dojotoolkit.org>)是一个强大的面向对象的 JavaScript 框架，主要由三大模块组成：Core、Dijit 和 DojoX。Core 提供 Ajax、events、packaging、CSS-based querying、animations 和 JSON 等相关操作 API。Dijit 是一个可更换皮肤，且基于模板的 Web UI 控件库。DojoX 包括一些创新/新颖的代码和控件：DateGrid、charts、离线应用和跨浏览器矢量绘图等，其官网如图 1.2 所示。



图 1.2 Dojo 官网

2. YUI

YUI(<http://developer.yahoo.com/yui>)是 Yahoo! User Interface (YUI) Librar 库的简称，它是一组采用 DOM Scripting、Dhtml 和 Ajax 等技术开发出的 Web UI 控件和工具，中文翻译就是 Yahoo 用户界面库，其官网如图 1.3 所示。

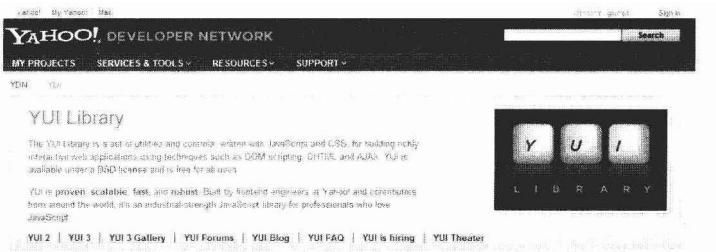


图 1.3 YUI 官网