

全面讲解HTML5游戏开发所需要掌握的各种最新技术、工具和框架，以及开发的思维和方法

包含大量实例，实战性和可操作性极强，理论与实践完美结合

Learning HTML5 Game Programming

A Hands-on Guide to Building Online Games Using Canvas, SVG, and WebGL

HTML5游戏开发 实践指南

(美) James L. Williams 著

黄敏 李彬 何元媛 等译 郝刚 审校



机械工业出版社
China Machine Press

HTML5游戏开发实践指南

Learning HTML5 Game Programming

A Hands-on Guide to Building Online Games Using
Canvas, SVG, and WebGL

(美) James L. Williams 著

黄敏 李彬 何元媛 等译
郝刚 审校



机械工业出版社
China Machine Press

本书以实践为导向,全面讲解了HTML5游戏开发所需掌握的各种最新技术、可以使用的各种工具和框架,以及开发的思维和方法,是系统学习HTML5游戏开发的最佳选择之一。

本书内容全面而且极具前瞻性,不仅重点阐述了HTML5和JavaScript的核心技术,而且还仔细讲解了WebGL、SVG、ProcessingJS、Inkscape、RaphaëlJS、PhoneGap、SGF、CoffeeScript和Node.js等与HTML5开发相关的各种技术、工具、框架、类库等的功能和使用方法,为HTML5应用开发者描绘了一幅完整的技术全景图。更为重要的是,它还深入讲解了HTML5游戏的运作机制、游戏开发与设计的思维和方法、游戏的性能优化方法,及其在各种平台上的发布方法。本书的实战性和可操作性也非常强,包含大量示例,有助于读者对理论知识进行融会贯通。

全书一共11章:第1章详细讲解了Canvas等HTML5的核心技术;第2章首先介绍了如何搭建一个完善的HTML5开发环境,然后讲解了如何部署HTML5应用开发需要的Web服务器端工具、浏览器端工具以及HTML5工具;第3章讲解了JavaScript的核心技术以及jQuery框架的使用;第4章探讨了游戏的设计思路和游戏的运行机制;第5~8章以实例的方式讲解了如何通过Canvas标签、SVG、RaphaëlJS、WebGL、Three.js、GWT、CoffeeScript等最新流行的技术和框架来开发游戏;第9章讲解了如何利用现今流行的Node.js来构建多人游戏服务器;第10章讲解了在Android、iOS、Windows Phone等主流移动平台上开发HTML5游戏的核心技术和PhoneGap等移动开发框架;第11章探讨了HTML5游戏的资源优化以及在各种平台的完整发布过程。

Authorized translation from the English language edition, entitled *Learning HTML5 Game Programming: A Hands-on Guide to Building Online Games Using Canvas, SVG, and WebGL*, 9780321767363 by James L. Williams, published by Pearson Education, Inc., Copyright © 2012 Pearson Education, Inc.

All rights reserved. No part of this book may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage retrieval system, without permission from Pearson Education, Inc.

CHINESE SIMPLIFIED language edition published by PEARSON EDUCATION ASIA LTD., and CHINA MACHINE PRESS Copyright © 2012.

本书封底贴有Pearson Education(培生教育出版集团)激光防伪标签,无标签者不得销售。

封底无防伪标均为盗版

版权所有,侵权必究

本书法律顾问 北京市展达律师事务所

本书版权登记号:图字:01-2011-7734

图书在版编目(CIP)数据

HTML5游戏开发实践指南/(美)威廉姆斯(Williams, J.L.)著;黄敏等译.—北京:机械工业出版社,2012.5
书名原文:Learning HTML5 Game Programming: A Hands-on Guide to Building Online Games Using Canvas, SVG, and WebGL

ISBN 978-7-111-38034-4

I. H… II. ①威… ②黄… III. 超文本标记语言, HTML—游戏—程序设计—指南 IV. TP312-62

中国版本图书馆CIP数据核字(2012)第068926号

机械工业出版社(北京市西城区百万庄大街22号 邮政编码 100037)

责任编辑:高婧雅

印刷

2012年5月第1版第1次印刷

186mm×240mm·13.5印张

标准书号:ISBN 978-7-111-38034-4

定价:59.00元

凡购本书,如有缺页、倒页、脱页,由本社发行部调换

客服热线:(010) 88378991; 88361066

购书热线:(010) 68326294; 88379649; 68995259

投稿热线:(010) 88379604

读者信箱:hzjsj@hzbook.com

译者序

本书的作者James L.Williams不仅是一位经验丰富的技术专家，同时也是一位优秀的技术作家，相信熟悉Java和游戏编程的读者对Williams不会感到陌生。

很荣幸有机会翻译这本书。这是一本非常棒的书，如果读者致力于HTML5游戏编程方面的技术研究，那么本书是很不错的选择。本书结构严密、条理清晰、内容丰富，涵盖了大量使用HTML5进行游戏编程的技术关键点。书中从HTML5游戏开发环境的安装配置开始介绍，重点讲解了实现复杂交互、SGF框架、Canvas标签、SVG矢量图形应用、RaphaëlJS库、WebGL 3D、GWT、CoffeeScript、Node.js、优化游戏性能的方法。在作者的精心安排下，这些内容分布于11章中，每章都具有独立鲜明的主题，并提供了实践性很强的应用示例和练习。读者在使用本书时，可以逐章阅读，也可以直接查阅指定内容，这种编排方式对初级读者的帮助不言而喻，也能为具备一定基础的读者获取需要的内容提供方便。

本书适合有志于掌握HTML5游戏编程技术的人员学习参考。无论是对经验丰富的游戏开发人员，还是初次接触Web或者移动游戏开发的读者，本书都很实用。书中讲解的内容涉及范围广泛，且有一定难度。为此，作者设计了大量的示例和练习帮助读者理解和掌握相关内容。建议读者在学习本书的过程中，亲自编码实现书中的示例，认真思考每章后面的练习，从而加深对这些技术关键点的理解和掌握，相信本书能够帮助读者迅速掌握HTML5游戏编程技术。

参与本书翻译和审校工作的人员有：黄敏、李彬、何元媛、郝刚、赵忠庆、唐弋涵、屈凤英和倪锦屏等。限于水平，书中翻译不妥或错误之处在所难免，敬请广大读者批评指正。

前言

笔者撰写本书是为了解一时之痒，同时也是由于看到了目前新生的HTML5游戏社区的发展潜力。笔者希望引领读者，通过最佳实践遨游陌生的HTML5领域，并掌握Canvas、WebGL和SVG这些技术。

开发人员已经认识到HTML5并不仅仅是一个玩具。他们惊喜地发现：HTML5拥有和精美的桌面应用程序一样丰富的内容——例如，双缓冲区处理、硬件加速、无须插件支持的浏览器内置缓存等。而此前，部分开发人员还认为，只有Flash技术才能提供类似支持。可以看到一种有趣的转变，即由“一切使用Flash”到“仅当使用HTML5无法实现时才使用Flash”的技术趋势。

在笔者写这本书时，HTML5游戏编程环境已经发展得相当成熟。笔者相信这项技术将会继续发展下去，同时期待它将来取得更大进展。

本书特点

本书内容涵盖“宽泛”意义上的HTML5技术，这意味着包括HTML5规范、WebGL、SVG和JavaScript，因为它们均与游戏编程相关。书中各章讨论了实现流行游戏效果背后的数学内容，并先向读者说明努力的方向，然后再展示一些代码示例。对于那些习惯使用JavaScript编程的读者而言，本书有一章讲解创建游戏时使用的其他语言，包括在JavaScript引擎中直接运行的语言，能够被编译成JavaScript的语言，以及二者混合的语言。近几个月，服务器端JavaScript席卷了编程世界，这意味着在构建游戏过程中增加了一个额外灵活的层次。程序逻辑仍然从自包含的客户端实例开始，之后在基本不需要修改代码的情况下，由服务器端实例进行扩展。本书结尾部分讲解了发布游戏的具体途径和方法。开发人员有多种游戏引擎和开发库可选择。在处理数据过程中，书中涉及的所有开发库都具有通用性，读者可通过本书轻松掌握这些知识，并将其用于其他库中。本书不讲解WebGL的底层细节，而是选择使用高层库的方式按需访问底层API。本书的目的是使读者迅速上手，而不是讲解关于WebGL的所有内容，那需要一整本书的篇章才能讲完。

本书读者对象

本书为那些愿意学习或使用HTML5及相关Web技术开发交互式游戏的应用开发者而编写。本书假设读者具有一些编程语言知识和基本的数学知识。

本书代码示例及练习

读者可以访问<http://www.informit.com/title/9780321767363>下载各章的代码示例和练习答案。另外，也可以访问Github网站<https://github.com/jwill/html5-game-book>找到代码示例。

致谢

对于本书而言，我有很多人要感谢。首先是在本书出版过程中提供了无法估量的贡献的Pearson团队（包括Trina MacDonald、Songlin Qiu和Olivia Basegio），他们使本书更棒，我认为他们成功做到了。撰写本书时，主题在迅速地变化发展，技术编辑Romin Irani、Pascal Rettig和Robert Schwentker对市场走向进行了特别的考量，我很高兴与他们合作，他们分享我的激情，给予我快速、明确的反馈，验证我的猜想并使我修正偏离的方向。最后，我要感谢的是我的家人和朋友，他们很有耐心而没有怨言，当我疲惫的时候让我放松，让我休息。

关于作者

James L. Williams是硅谷的一名开发人员，并频繁在很多国内外会议中发言。他成功地参与了2007年Google夏日代码大赛（Google Summer of Code），并找到了使用Groovy语言访问SwingLabs UI组件的简单方法。他还是Griffon项目的联合创始人之一。该项目是一个采用Java技术的富桌面框架应用程序。他和他的WalkIN团队曾经构建过一个运行在SXSW上的教练车产品，并在StartupBus 2011大赛中获奖。他的第一个视频游戏是运行在Coleco Adam（一款古老的家用计算机）上的Buck Rogers: Planet of Zoom（行星飞行）。这款计算机真是一台“怪兽”，其CPU为3.58MHz，采用高速磁带驱动，启动时可能会擦除软盘。他的博客地址是<http://jameswilliams.be/blog>。

目 录

译者序

前言

第 1 章 HTML5 概述 1

1.1 超越基本的 HTML 1

1.1.1 JavaScript 2

1.1.2 AJAX 2

1.2 填补空白 2

1.2.1 Google Gears 3

1.2.2 Chrome Frame 3

1.3 使用 WebSockets 和 Web Workers 编程 4

1.3.1 WebSockets 4

1.3.2 Web Workers 4

1.4 应用程序缓存 5

1.5 数据库 API 6

1.5.1 WebSQL API 6

1.5.2 IndexedDB API 7

1.6 Web Storage 7

1.7 定位 8

1.8 提示用户消息 10

1.8.1 为显示提示请求权限 11

1.8.2 创建提示 12

1.8.3 与提示信息实现交互 13

1.9 媒体元素 13

1.9.1 控制媒体播放 14

1.9.2 处理不支持的格式 15

1.10 绘画 API 16

1.10.1 Canvas 16

1.10.2 SVG 16

1.10.3 WebGL 16

1.11 使用 Microdata 传达信息 17

1.12 小结 18

第 2 章 配置开发环境 19

2.1 开发工具 19

2.1.1 安装 Java 19

2.1.2 安装 Eclipse IDE 和 Google 插件 20

2.1.3 Google Web Toolkit 21

2.2 Web 服务器端工具和选项 22

2.2.1 Google App Engine 23

2.2.2 Opera Unite 23

2.2.3 Node.js 和 RingoJS 23

2.3 浏览器端工具 24

2.3.1 Chrome 开发人员工具 24

2.3.2 Chrome 扩展程序 25

2.3.3 Safari 开发人员工具 26

2.3.4 Firebug 26

2.4 HTML5 工具 27

2.4.1 ProcessingJS 27

2.4.2 Inkscape 27

2.4.3 SVG-edit 27

2.4.4 Raphaël 28

2.5 3D 建模工具 29

2.6 小结 29

第 3 章 JavaScript 概述 30

3.1 什么是 JavaScript 30

3.1.1 JavaScript 基本类型	30	4.3 理解 SGF 中的 API	53
3.1.2 理解算术运算符	31	4.3.1 核心 API	53
3.1.3 理解 JavaScript 函数	31	4.3.2 组件 API	54
3.1.4 first-class 对象函数	32	4.3.3 资源 API 及网络 API	55
3.1.5 比较运算符	33	4.4 使用 SGF 构建乒乓球游戏	55
3.2 条件和循环语句	34	4.4.1 设置应用程序	56
3.2.1 使用循环控制程序流	35	4.4.2 绘制游戏配件	57
3.2.2 使用 setTimeout 和 setInterval 函 数实现延迟执行	36	4.5 碰撞检测和响应	59
3.3 使用继承和多态创建复杂对象	37	4.5.1 理解牛顿三大定律	59
3.4 JQuery 简介	39	4.5.2 让球动起来	61
3.4.1 使用选择器操作 DOM 元素	40	4.5.3 高级碰撞检测和粒子系统	62
3.4.2 JQuery 事件	41	4.6 通过人工智能增加游戏难度	63
3.4.3 JQuery 中的 AJAX	41	4.6.1 将人工智能应用到乒乓球游戏	63
3.4.4 跨站点脚本	42	4.6.2 将人工智能应用到井字棋游戏	64
3.5 JSON : 另一种 JavaScript 格式	42	4.7 小结	66
3.6 浏览器外的 JavaScript	43	4.8 练习	66
3.6.1 移动平台	43	第 5 章 使用 Canvas 标签构建游戏	67
3.6.2 将 JavaScript 作为中间语言	43	5.1 Canvas 基础	67
3.6.3 桌面应用中的 JavaScript	44	5.1.1 绘制路径	68
3.7 服务器端 JavaScript	45	5.1.2 绘制井字棋游戏的棋子	69
3.8 小结	46	5.2 使用变换绘制 Canvas 中的对象	70
3.9 练习	47	5.2.1 变换规则	72
第 4 章 游戏运行机制	48	5.2.2 保存和恢复 Canvas 绘图状态	72
4.1 游戏设计	48	5.3 在 Canvas 中使用图片	73
4.1.1 撰写基本设计文档	48	5.3.1 使用 Data URI 访问图片	73
4.1.2 确定游戏类型	49	5.3.2 使用 Spritesheets 访问图片	74
4.1.3 游戏循环	50	5.3.3 在 Canvas 中绘制图片	74
4.1.4 获取用户输入	50	5.4 使用 Trident.js 创建动画对象	75
4.2 使用高级数据结构实现游戏对象	50	5.4.1 创建时间线	75
4.2.1 使用 Set 构建唯一数据列表	50	5.4.2 使用关键帧实现动画	76
4.2.2 使用链表创建对象图	52	5.4.3 使用缓冲函数创建非线性时间线	77
		5.4.4 使用 Spritesheets 创建游戏对象	

动画·····	78	6.5.3 curveto ·····	103
5.5 在 2D 空间模拟 3D ·····	79	6.5.4 从 SVG 文件中导出路径 ·····	104
5.5.1 透视投影 ·····	79	6.5.5 在路径中使用动画 ·····	104
5.5.2 视差 ·····	80	6.6 使用插件扩展 Raphaël ·····	105
5.5.3 使用 JavaScript 创建视差效果 ·····	80	6.6.1 添加函数 ·····	105
5.6 创建 Copy Me 游戏 ·····	82	6.6.2 SVG 滤镜 ·····	105
5.6.1 绘制游戏对象 ·····	82	6.7 权衡速度 ·····	106
5.6.2 创建游戏音调 ·····	82	6.8 小结 ·····	106
5.6.3 在浏览器中播放 MIDI 文件 ·····	84	6.9 练习 ·····	106
5.6.4 同时播放多个音频 ·····	84	第 7 章 使用 WebGL 和 Three.js 构建	
5.6.5 顺序播放音频 ·····	85	游戏 ·····	108
5.6.6 绘制游戏文本 ·····	86	7.1 三维 ·····	109
5.6.7 使用 CSS 设置文本字体 ·····	86	7.2 用材质和光源美化对象 ·····	110
5.7 小结 ·····	87	7.2.1 了解照明 ·····	110
5.8 练习 ·····	87	7.2.2 使用材质及着色器 ·····	111
第 6 章 使用 SVG 和 RaphaëlJS 构建		7.3 创建第一个 Three.js 场景 ·····	113
游戏 ·····	88	7.3.1 设置视角 ·····	113
6.1 SVG 概述 ·····	88	7.3.2 观察世界 ·····	118
6.2 使用 RaphaëlJS ·····	89	7.4 使用 Three.js 加载 3D 模型 ·····	119
6.2.1 设置游戏开发环境 ·····	90	7.5 着色及纹理处理的编程 ·····	121
6.2.2 创建游戏板 ·····	91	7.6 使用纹理 ·····	123
6.2.3 绘制游戏文本 ·····	92	7.7 使用 Three.js 创建游戏 ·····	125
6.3 自定义字体 ·····	92	7.8 用游戏模拟真实世界 ·····	126
6.3.1 设置颜色 ·····	95	7.9 重温粒子系统 ·····	129
6.3.2 加载游戏资源 ·····	96	7.10 创建场景 ·····	130
6.3.3 将 SVG 文件转换成位图 ·····	97	7.11 模型动画设计 ·····	131
6.4 创建游戏类 ·····	98	7.12 3D 模型资源 ·····	131
6.4.1 洗牌 ·····	99	7.13 校准游戏 ·····	132
6.4.2 绘制及卡片动画 ·····	100	7.13.1 用 Stats.js 检查帧速率 ·····	133
6.5 制作高级动画 ·····	102	7.13.2 运用 WebGL Inspector ·····	133
6.5.1 路径 ·····	102	7.14 小结 ·····	134
6.5.2 moveto 和 lineto ·····	103	7.15 练习 ·····	134

第 8 章 不使用 JavaScript 构建游戏	135
8.1 Google Web Toolkit	135
8.1.1 了解 GWT 部件及布局	136
8.1.2 使用 JSNI 向 GWT 展示 Java- Script 库	137
8.1.3 RaphaëlGWT	138
8.1.4 使用 gwt-html5-media 创建声音	139
8.1.5 使用 GWT 访问绘图 API	139
8.2 CoffeeScript	140
8.2.1 安装 CoffeeScript	141
8.2.2 编译 CoffeeScript 文件	141
8.3 CoffeeScript 快速指南	142
8.3.1 基础	142
8.3.2 函数和调用	142
8.3.3 别名、条件语句和循环语句	144
8.3.4 增强的 for 循环和集合视图	144
8.3.5 类和继承	145
8.4 替代技术	146
8.4.1 Cappuccino	146
8.4.2 Pyjamas	146
8.5 小结	147
8.6 练习	147
第 9 章 构建多人游戏服务器	148
9.1 Node.js 简介	148
9.1.1 使用 Node 包管理器扩展 Node	149
9.1.2 Node 的多版本管理	149
9.2 使用 ExpressJS 简化 Web App 生成	149
9.2.1 使用 URL 路由满足服务请求	150
9.2.2 会话管理	152
9.2.3 理解 ExpressJS 应用程序的结构	152
9.2.4 使用 CoffeeKup 定制 HTML 模板	153
9.3 使用缓存存储数据	155
9.4 管理 C/S 通信	155
9.4.1 使用 Socket.IO 通信	156
9.4.2 使用 Express 安装一个简单的 Socket.IO 应用程序	156
9.4.3 使用 NowJS 简化 Web Sockets	157
9.5 调试 Node 应用程序	158
9.6 创建游戏服务器	159
9.6.1 创建游戏大厅	159
9.6.2 使用 NowJS Groups 创建游戏 房间	160
9.6.3 管理游戏玩家和在不同的游戏 房间中移动	161
9.7 管理游戏	162
9.8 小结	163
9.9 练习	163
第 10 章 开发移动游戏	164
10.1 选择移动平台	164
10.1.1 iOS	164
10.1.2 Android	164
10.1.3 WebOS	165
10.1.4 Windows Phone 7	165
10.2 flick、tap 和 swipe：移动手势 简介	166
10.3 选择应用程序还是 Web 站点	166
10.4 在移动设备上存储数据	168
10.4.1 从 Lawnchair 中解脱：一种相对 容易的数据存储方法	168
10.4.2 开始使用 Lawnchair	169
10.5 使用 JQuery 和 Zepto 简化客户端 脚本	170
10.5.1 使用 JQuery 变种	170

10.5.2 使用 Zepto.js	171	游戏	189
10.6 使用 JoApp 构建应用程序	172	11.4 在 Chrome Web Store 上发布应用	190
10.7 选择应用程序框架	173	11.4.1 描述应用的元数据	191
10.7.1 PhoneGap	173	11.4.2 部署主机应用	191
10.7.2 深入 PhoneGap API	173	11.4.3 部署应用程序包	193
10.7.3 Appcelerator Titanium	175	11.4.4 在本地测试应用程序	193
10.7.4 深入 Appcelerator Titanium API	175	11.4.5 上传应用程序到 Chrome Web Store	193
10.8 使用 Titanium 和 PhoneGap 打包 Android 应用程序	176	11.4.6 配置应用程序	194
10.8.1 使用 Titanium 打包应用程序	177	11.4.7 确定使用包应用还是使用 Chrome 主机应用	195
10.8.2 使用 PhoneGap 打包应用程序	179	11.5 使用 TapJS 发布应用	197
10.9 小结	182	11.5.1 创建 TapJS 应用	197
10.10 练习	183	11.5.2 打包 TapJS 应用程序	199
第 11 章 游戏发布	184	11.5.3 将 TapJS 应用程序发布到 Facebook	199
11.1 优化游戏资源	184	11.6 使用 Kongregate 发布游戏	201
11.1.1 使用 Google Closure 编译器进行 代码小型化	184	11.7 将 HTML5 应用程序发布到桌面	201
11.1.2 使用缓存运行离线程序	185	11.8 小结	202
11.2 服务器的主机服务	188	11.9 练习	202
11.3 在部署 Node.js 的服务器上发布			

第 ① 章

HTML5 概述

HTML5是HTML下一版本的规范草案，它突破了其前身，即HTML4 和XHTML技术。在HTML5中移除了部分元素，同时HTML5也不再基于旧的文本标记标准SGML。HTML5比HTML4在错误语法方面具有更好的兼容性。同时，还允许不同浏览器以相同方式解析和展示格式错误的文档。这个版本的HTML增加了许多值得注意的内容，例如，内建原生图片支持以及视听元素。本章讲解HTML5的新增特性和相关的JavaScript API。

1.1 超越基本的HTML

Tim Berners-Lee 发明的 HTML（Hypertext Markup Language，超文本标记语言）从 1990 年至今已有相当长一段历史。图 1-1 简要展示了 HTML 的发展历程（来自 HTML5Rocks slides，<http://slides.html5rocks.com/#slide3>）。

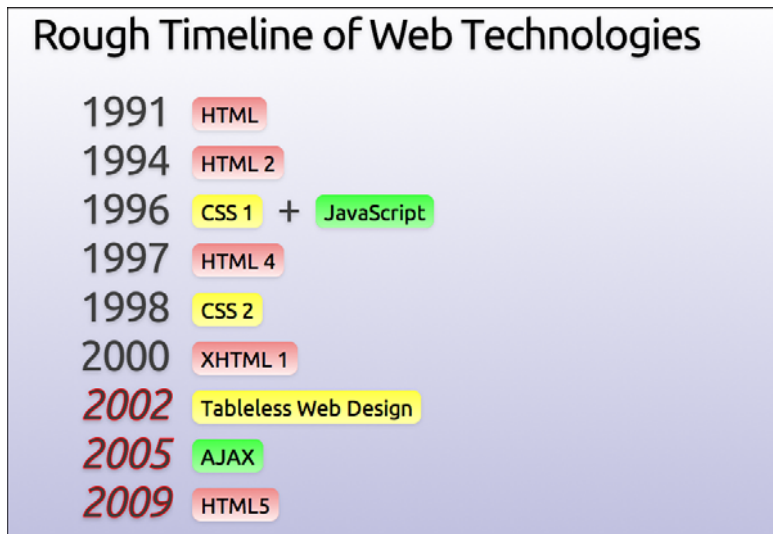


图1-1 HTML的发展历程

虽然所有的进步对于推动标准前进都至关重要，但笔者最感兴趣的是1996年出现的

JavaScript和2005年出现的AJAX。这些新特性使得Web页面，从一个表示静态单向数据的媒体（例如，报纸和图书）转变为一个允许双向交流的媒体。

1.1.1 JavaScript

JavaScript（前身为LiveScript，正式名为ECMAScript）由Netscape公司开发的浏览器脚本语言发展而来。它是一种基于原型的弱类型脚本语言，具有面向对象或者面向函数的特性。尽管名为JavaScript，也继承了Java语言的某些方面，但是它与C语言更为相似。

作为Sun Microsystems公司（现为Oracle公司）和Netscape公司之间，关于推广脚本语言和Java Applet技术市场协议的一部分，该语言被重命名为JavaScript。JavaScript语言广泛用于编写客户端Web页面脚本。同时，由于Sun公司已将“JavaScript”名称注册成为商标，所以微软公司将发布的与其兼容的脚本语言命名为JScript，并对其添加了一些新内容，并进行了一些修改。

1.1.2 AJAX

AJAX（Asynchronous JavaScript and XML，异步JavaScript和XML）曾经一度被认为是非专业人员和脚本新人的玩具，但是它却帮助开发人员解决了很多复杂问题。因此，AJAX重新激起了人们对JavaScript编程的兴趣。

AJAX的核心是微软公司在20世纪90年代后期发明的XMLHttpRequest对象。XMLHttpRequest对象可使站点连接到远程服务器，并获取结构化数据。与建立一系列静态页面不同，开发人员使用XMLHttpRequest对象可创建高度动态的应用程序。Gmail、Twitter和Facebook都是这种类型应用的示例。

目前，JavaScript正处于一个复兴过程中。主流浏览器开发商都将JavaScript引擎性能作为相互比较的指标。JavaScript作为一种主要的编程语言已经渗透到服务器端Web组件，例如Node.js，以及移动应用框架WebOS和PhoneGap等。

1.2 填补空白

即使是最好的标准，被广泛采用也需要一段时间。为了不让缺失的特性限制创新，Google公司通过创建Chrome Frame和Google Gears（以下简称Gears），将先进的特性引入旧版浏览器中。

1.2.1 Google Gears

2007年5月首次发布的Google Gears，实现了HTML5规范草案中的一些先进特性。在HTML5真正到来之前，许多应用程序在一定程度上使用了Gears，包括Google的应用程序（Gmail、YouTube、Doc、Reader等）、MySpace、Remember the Milk、WordPress和其他一些程序。Gears由若干个模块组成，这些模块将桌面应用程序的典型功能引入浏览器中。下面讲解一下相关功能特性。

在发布的第一版Gears中，引入了Database、LocalServer、WorkerPool三个模块。Gears的Database API使用类似SQLite语法来为Web应用创建关系型数据存储。数据本地化存储在特定应用程序中，并使用规范化的跨站点脚本规则进行编译，从而保证该应用程序无法访问其域之外的数据。LocalServer模块允许Web应用程序在本地缓存中保存和获取资源，即使是在没有连接因特网的情况下也是如此。在站点的manifest文件中，可以指定本地缓存所存储的资源。当请求的资源与manifest文件中某个URL匹配时，LocalServer模块将拦截请求，并从本地存储中提供请求的资源。

WorkerPool模块可帮助解决包含大量JavaScript脚本的站点所面临的普遍问题：即长时间运行的JavaScript脚本会阻塞站点交互。默认情况下，使用单个线程来处理一个站点的运行工作。对于可快速返回的短暂突发任务（例如简单的DOM操作）而言，使用这种方式通常可以轻松应对。但对于需要长时间执行的任务，例如文件输入/输出，或者试图从一台缓慢的服务器获取资源而言，可能阻塞交互，并使得浏览器认为脚本响应过慢，应该强制停止。WorkerPool模块将多线程计算的概念引进了浏览器，其创建的工作线程能执行任意JavaScript代码。只要工作线程处于同一个WorkPool中，那么工作线程就能彼此发送和接收消息，进行任务协作。同时，根据所获取继承的策略，工作线程还可以跨源执行。工作线程无法访问window对象的一些属性，例如Timer和HttpRequest，为了解决这个问题，Gears提供了自己的实现方法。

另一个有趣的API是Geolocation API。Geolocation API可试图根据一些可用数据来定位访问者的位置，例如IP地址、已知位置的Wi-Fi路由器、通信基站和其他相关数据。

2009年11月，Google停止了Gears的主要开发工作，并将注意力转移到实现HTML5的相关特性。幸运的是，这里讲解的这些特性在某些层面上和HTML5是一致的。

1.2.2 Chrome Frame

Chrome Frame是Google的一个项目，它将Google Chrome浏览器作为插件内嵌到不支

持HTML5的Internet Explorer 6以及更高版本的浏览器中。通过识别一个meta标签可以激活Chrome Frame。目前，即使在没有管理员权限的系统上也能安装和使用Chrome Frame。读者可在<http://code.google.com/chrome/chromeframe/>找到关于Chrome Frame的更多信息。

1.3 使用WebSockets和Web Workers编程

HTML5增加了一组API，可帮助实现Web应用程序通信和协同工作。WebSockets允许Web应用程序建立一条与Web服务交互的通道。Web Workers则使得Web应用程序能够执行多项重要任务，而避免锁死浏览器的现象发生。

1.3.1 WebSockets

WebSockets能够在Web应用程序和URI端点之间建立一条双向通道。socket对象能够发送和接收消息，也可以响应WebSocket对象的打开或者关闭。还有其他几种方式也能够实现双向通信，包括Comet（使用long polling技术的AJAX）、Bayeux和BOSH，不过这些方式未包含在HTML5规范中。

代码清单1-1创建了一个WebSocket对象，用于与回显服务器端点进行通信。在创建socket对象之后，代码接着创建了一些函数，分别在socket打开、关闭、接收到消息或者抛出异常时执行。最后，socket发送“Hello World!”消息。当浏览器接收到这条返回的消息后，将显示“Hello World!”。

代码清单1-1 显示消息的WebSocket代码

```
var socket = new WebSocket(ws://websockets.org:8787/echo);
socket.onopen = function(evt) { console.log("Socket opened");};
socket.onclose = function(evt) {console.log("Socket closed");};
socket.onmessage = function(evt){console.log(evt.data);};
socket.onerror = function(evt) {console.log("Error: "+evt.data);};

socket.send("Hello World!");
```

1.3.2 Web Workers

HTML5中的Web Workers类似于Google Gears中的WorkerPools。与WorkerPools不同的是，开发人员不必创建池来存放Web Workers。代码清单1-2说明了创建简单worker的方法，以及为其接收消息后，设置处理函数的方法。代码清单1-2和代码清单1-3说明如何使用Web Worker创建一个能每隔2秒显示当前日期和时间的函数。

代码清单1-2 请求当前时间的Web页面

```
<!DOCTYPE HTML>
<html>
  <head>
    <title>Web Worker example</title>
  </head>
  <body>
    <p>The time is now: <span id="result" /></p>
    <script>
      var worker = new Worker('worker.js');
      worker.onmessage = function (event) {
        document.getElementById('result').innerText = event.data;
      };
    </script>
  </body>
</html>
```

在代码清单1-3中显示了与上相关的JavaScript文件worker.js。

代码清单1-3 获取日期和时间的worker.js文件

```
setInterval(function() {w
  postMessage(new Date());
}, 2000);
```

读者通过以上两个代码清单可以看到，worker能使用postMessage()方法发送消息，并利用onmessage事件来监听消息。另外，通过在函数中执行terminate()方法，并将该函数绑定到onerror事件，可以实现响应错误消息，并终止worker对象。

worker可以共享，还能通过MessagePorts发送消息。但与Web Worker规范的其他方面一样，这部分功能也还没形成稳定版本，而本书示例也未曾用到这些功能。因此，worker共享的用法将作为练习留给读者研究。

1.4 应用程序缓存

应用程序缓存（Application Cache）提供了一种使应用程序离线运行的方法。这有些类似于Gears中的LocalServer特性。二者之间的区别是，应用程序缓存不使用JSON格式文件，而使用纯文本文件来设置缓存哪些文件。代码清单1-4给出了一个简单的用于设置缓存资源的manifest文件。

代码清单1-4 应用程序manifest文件示例

```
CACHE MANIFEST
# above line is required, this line is a comment
mygame/game.html
mygame/images/image1.png
mygame/assets/sound2.ogg
```

应用程序缓存能够响应以下几个事件：onchecking、error、cached、noupdate、progress、updateready和obsolete。开发人员可以监听这些事件，并将应用程序状态通知给用户。使用应用程序缓存，可使游戏很好地应对断电情况。同时，通过缓存资源，游戏能够更快速启动，使玩家获得愉快的体验。如果需要，还可以利用应用程序缓存让玩家在离线状态下玩游戏。本章暂且不讨论这些复杂的技术，在第11章中，笔者将更加深入地讲解使用应用程序缓存的细节。

1.5 数据库API

目前，HTML5提供了多种方式存储结构化数据，包括Webkit浏览器提供的WebSQL API，以及与之竞争的，由Firefox领衔的IndexedDB API。

1.5.1 WebSQL API

WebSQL实现了一种类似于SQL的语法，用以存储结构化的数据。目前，WebSQL以SQLite为核心实现方式，但这并不是规范的要求。

WebSQL未提供创建数据库的函数。然而，通过调用openDatabase()函数并指定参数可以连接一个数据库。如果指定的数据库不存在，那么一个新数据库将会被自动创建。例如，通过下面的代码可创建一个名为myDB的数据库。

```
var db = openDatabase("myDB", "1.0", "myDB Database", 100000);
```

此处，代码传递参数“myDB”作为名称，并设置版本号“1.0”，同时还设置显示名称为“myDB Database”，最后指定一个100KB的估计容量。另外，还可以设置一个在创建数据库时执行的回调函数，但不是必需的。图1-2展示的是，Chrome浏览器开发人员工具中的Storage面板。在第2章中，将执行上面那行代码，并详细介绍Storage面板。

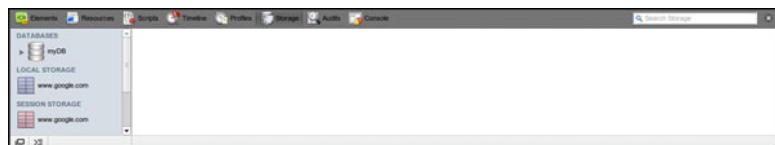


图1-2 显示已创建数据库的Storage面板

在窗口的右边部分，开发人员可以执行任意SQL代码。如图1-3所示，该操作创建了一个数据库表后，插入一些数据，并执行了一条查询语句。



图1-3 在Storage面板中执行SQL语句

虽然未被广泛支持，但HTML5规范还是要求同时支持异步和同步的数据库连接与事务处理。上文的示例使用了异步数据库连接。如果调用`openDatabaseSync`方法，并传递和上面示例相同的参数，那么可创建同步数据库连接。在初始化连接之后，在数据库操作上二者之间没有什么差别。只要了解调用`transaction(...)`方法可创建用于数据读/写的事务，调用`readTransaction`方法可创建只读事务即可。

注意：目前还未很好地支持同步数据库连接。通常来讲，开发人员编写代码时应该使用异步数据库连接。

1.5.2 IndexedDB API

IndexedDB直接将对象存储在对象存储中。这样可轻松实现NOSQL数据库的JavaScript版本。IndexedDB类似于对象数据库MongoDB、CouchDB和SimpleDB等。在撰写本书时，IndexedDB API的实现还未统一，其使用不同的命名模式，对于规范的遵守也不相同。例如，Internet Explorer浏览器使用IndexedDB时，要求安装ActiveX插件。笔者建议读者查阅<http://nparashuram.com/trialtool/index.html#example=/ttd/IndexedDB/all.html>，了解该组API分别在Firefox、Chrome和Internet Explorer浏览器中的应用。大多数情况下，为Chrome浏览器编写的代码可在Safari中无缝使用。

1.6 Web Storage

Web Storage提供了一组在客户端存储数据的API，其方式类似于在浏览器中存储

cookie。localStorage对象存储应用程序在启动时用到的数据；sessionStorage对象存储与会话相关的数据，且在会话结束时，将相关数据清除。数据以键/值对的方式存储。这两个对象实现了表1-1中列举的函数。

表1-1 Web Storage函数

函 数 名	说 明
setItem(key:String, value)	使用指定值创建键/值对。value值有时可设置为一个字符串
getItem(key:String)	根据给定参数key，返回对应键/值对
removeItem(key:String)	根据给定参数key，移除对应键/值对
clear()	删除Storage对象存储的所有键/值对
key(index:long)	返回索引对应的键

Storage对象使用length属性来表示键/值对数量。

Web Storage还提供了其他一些API，可以用来替代表1-1中列举的getItem和setItem函数。这个替代的API使用类似于数组的方式引用键key。例如，以下代码使用家乡的报纸名作为value值，来设置localStorage对象中的键/值对：

```
localStorage['newspaper'] = 'The Baltimore Sun';
```

同样地，也可以使用上面语句中的等号左边部分来获取键/值对中的值：

```
localStorage['newspaper'];
```

在编写游戏的时候，开发人员可以使用Web Storage保存游戏得分或者游戏数据。

1.7 定位

为了跟踪用户的地理位置信息，必须首先要求获得用户的授权，而Geolocation API未明确提供相关授权函数。然而，浏览器可透明地替用户处理授权。当Geolocation API首次向未获得授权的站点请求用户位置信息时，浏览器将自动弹出一个上下文窗口，以请求用户授权。

开发人员可通过以下代码检查浏览器是否支持Geolocation API：

```
navigator.geolocation
```

如果geolocation是一个非null（空）值，那么可以用其进行定位。

Position对象包括名为coords的Coordinates对象和一个用于表示获取位置数据的时间戳，通过Position对象可以计算出用户的位置信息。表1-2列出了coords对象的属性。

表1-2 Coordinates对象属性

属 性 名	值 类 型	说 明
latitude	double	定位数据中的经度
longitude	double	定位数据中的纬度
altitude	double	定位数据中的高度，单位：米。如果得不到高度数据，那么该值为null
accuracy	double	经纬度的误差范围，单位：米。如果得不到经纬度的误差范围数据，那么该值为null
altitudeAccuracy	double	高度值的误差范围。如果得不到高度值的误差范围信息，那么该值为null
heading	double	设备移动的方向，单位为度（0° ~ 360°，包括边界）。如果得不到移动方向信息，那么该值为 NaN
speed	double	设备移动的速度，单位为米。如果得不到移动速度信息，那么该值为null

当开发人员验证可使用定位特性后，就可很方便地获取设备的定位数据。开发人员仅需调用getCurrentPosition函数，并传入一个、两个或者三个参数即可。其中，前两个参数都是函数，分别在成功获取位置信息和获取失败时执行，另一个参数是位置请求的选项设置。

代码清单1-5展示了获取位置信息的方法，并通过marker对象在地图上标注该位置，同时以定位位置为中心绘制一个圆来突出定位的误差范围。

代码清单1-5 使用Geolocation绘制地图

```
if(navigator.geolocation) {
    navigator.geolocation.getCurrentPosition(function(pos) {
        var latitude = pos.coords.latitude;
        var longitude = pos.coords.longitude;

        var options = {
            position:new google.maps.LatLng(latitude, longitude)
            ,title:"Your location"};

        var marker = new google.maps.Marker(options);

        var circle = new google.maps.Circle({
            map:map, radius:pos.coords.accuracy
        });
        circle.bindTo('center', marker, 'position');

        marker.setMap(map);

        map.setCenter( new google.maps.LatLng(latitude, longitude));
    },
    function(error) {
```

```
    console.log(error.message);  
  });  
}
```

在确认定位特性可用之后，开发人员可首先试图获取设备的位置信息。在这个例子中，向`getCurrentPosition`传入两个函数作为参数，它们分别在成功获取地理位置信息以及获取失败或者用户拒绝授权定位时执行。在获取纬度和经度数据后，在该经纬度位置上创建一个marker标识，并为其设置“Your location”标签。然后在该标识上添加一个圆，该圆的半径等于此次定位的经纬度误差范围。最后，如果出现错误，那么错误处理函数将错误信息输出并显示到控制台。图1-4展示的是使用了OpenStreetMap标记集的定位示例。

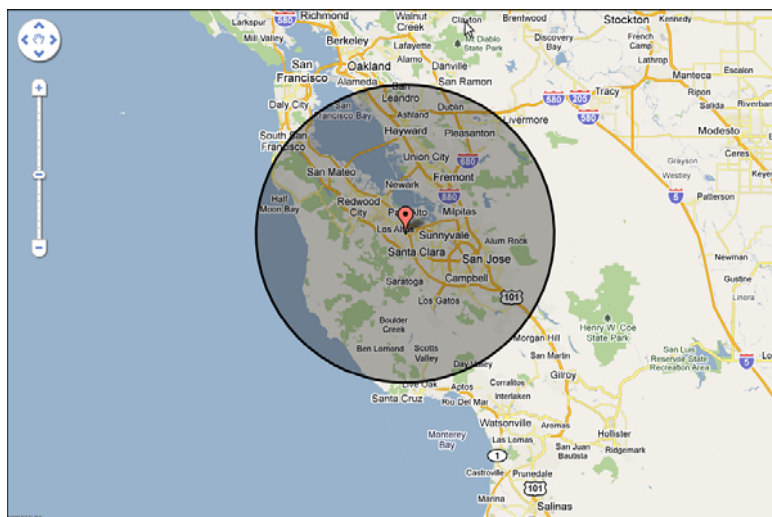


图1-4 在浏览器中显示定位

在上面的示例中，没有将`options`对象作为参数进行设置，但如果需要，可以将其作为第三个参数，以便设置所要获取的数据。同时，还可以设置一个函数作为监听器，在`watchPosition`函数每次返回位置变化时执行。Geolocation是很耗资源的API，使用它要谨慎，并注意使用缓存。

开发人员可以使用Geolocation来创建地区游戏者领先排行榜，也可以在多玩家游戏中，使用它匹配地理位置相对接近的玩家。

1.8 提示用户消息

在HTML4中，与用户进行消息交互的方式有限。例如，显示Alert窗口提示消息，或者

通过div元素显示消息。所有浏览器均支持弹出Alert窗口方式，但这种方式会给用户造成混乱，即弹出窗口将引起用户直接的关注。用户必须首先关掉弹出窗口，然后才能继续完成其他操作。如果弹出的消息窗口遮挡了游戏玩家的视线，而使玩家所扮演的游戏角色死去，那么他将非常生气。使用div元素显示消息会相对好一些，但这种方式没有标准的实现方法，并且这种方式很容易被用户忽略。第一种消息提示方式过于唐突，而第二种则可能难以引起用户注意，因此必须采取一种折中方式。

在原生Mac OS X系统、Ubuntu系统或者安装相关插件的Windows系统中，应用程序可向用户发送可配置的消息，以便通知用户重要的事件或变化。图1-5展示了这种消息提示方式的示例。

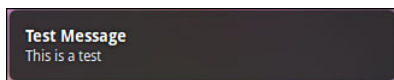


图1-5 桌面应用程序的消息提示

与桌面应用程序的消息提示一样，Web程序下的提示方式允许在显示相关消息时设置图片。

1.8.1 为显示提示请求权限

在向用户显示提示之前，必须首先获取用户授权。明确要求用户授权可以避免用户受到恶意提示的骚扰。以下代码实现在显示提示之前请求授权。

```
window.webkitNotifications.requestPermission();
```

如图1-6所示，执行以上代码将在浏览器中显示相关消息，以便用户允许或者拒绝显示提示。在调用以上方法时，开发人员可以不传递参数，也可以传递一个函数作为参数，用以在用户确认或者拒绝授权时执行。



图1-6 Web notifications授权提示

同时，开发人员还可以通过执行以下代码验证授权。

```
window.webkitNotifications.checkPermission();
```

如表1-3所示，checkPermission()函数返回一个整数，以便说明授权级别。

表1-3 Notification授权级别

常量名称	值
PERMISSION_ALLOWED	0
PERMISSION_UNKNOWN	1
PERMISSION_DENIED	2

看到webkitNotifications这个对象名称，读者可能猜想notifications能够在基于Webkit的浏览器中执行，例如，Google的Chrome浏览器和苹果的Safari浏览器。事实上，虽然Safari浏览器以Webkit为核心，但此浏览器并没有实现Notification API。如果按照规范来实现，那么以上对象不应叫做webkitNotifications，而应该简称为notifications。

1.8.2 创建提示

开发人员可以创建两种类型的提示：基本型和HTML型。基本型提示用以显示简单消息以及可选的标题和图标，而HTML型提示能够显示任意URL设定的资源。例如，执行以下代码可创建一个基本型提示。

```
var msg = window.webkitNotifications.createNotification(
    '', 'Test Notification', 'Hello World'
);
```

上面创建了一个标题为“Test Notification”，提示内容为“Hello World”的提示。由于传递了一个空字符串作为图标参数，所以API将忽略该参数。对于其他参数也可以这样设置。如果不想显示该参数，那么可以采取这种处理方式。然而，有时向函数传递无值参数，可能会显示“undefined”文本消息，或者无链接的图片。图1-7展示了浏览器中显示的提示。正如所看到的那样，提示窗口外观非常简洁，除了显示参数中设置的消息之外，开发人员不能设置外观。

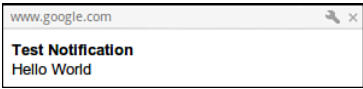


图1-7 基本型Web提示

如前文所述，HTML型提示可以从任意URL获取资源作为内容，例如获取站点页面或者图片。以下代码创建了一个HTML型提示，它从指定的URL中获取资源作为提示内容。

```
var msg = window.webkitNotifications.createHTMLNotification(
    'http://example.com'
);
```

开发人员无法修改HTML型提示所显示的尺寸。对于小屏幕而言，除非设置的URL所对应的资源能够根据屏幕大小调整自身尺寸，否则将出现滚动条。对于1 680×1 050的屏幕，默认尺寸是宽度300像素，高度50像素。在编写本书时，与提示相关的API仍然处于草案阶段，所以上面提到的尺寸很可能会发生变化。在宽度、高度属性正式纳入规范之前，开发人员最好使用简单型提示。

1.8.3 与提示信息实现交互

开发人员可使用两个基本函数来设置提示消息：show() 函数用于向用户显示提示，cancel() 函数用于隐藏提示（如果当前是可见状态）或阻止提示信息显示出来（如果当前为不可见状态）。另外，Web 提示还可在发生提示事件时执行函数。表 1-4 列举了可响应的事件。

表1-4 Web提示事件

函 数 名	说 明
onclick	如果底层平台支持，当单击提示时执行。尽量避免使用该事件
onclose	当触发close事件时执行。例如，用户或者执行代码关闭了提示
ondisplay	当调用show()函数，提示显示给用户后执行
onerror	当调用show()函数出现错误时执行

读者可以访问以下网址查阅当前规范草案进展。<http://dev.chromium.org/developers/design-documents/desktop-notifications/api-specification>。

1.9 媒体元素

HTML设计之初，其重点是显示普通文本，而显示图片的功能是后来才出现的。这就不难理解为什么需要安装浏览器插件或者扩展程序才能播放音频或者视频了。多数情况下，HTML使用Flash播放音频和视频文件。而HTML5试图通过引入audio和video标签来解决安装插件的问题。

利用audio和video标签，可实现在浏览器中原生支持播放媒体内容。同时，这两个标签提供了一组属性用于控制播放状态。以下代码使用最基本HTML代码实现了内嵌媒体播放功能（本例使用一个音频文件）：

```
<audio src="song.mp3" autoplay />
```

以上代码创建了一个audio HTML标签，它将song.mp3设置为播放源，并要求页面“自

动播放”音频内容。这行代码与下面的JavaScript代码实现了同样的功能。

```
var song = new Audio();
song.src = "song.mp3";
song.autoplay = true;
song.load();
```

1.9.1 控制媒体播放

除了前面示例中用到的autoplay属性外，还有其他几个属性可以用于控制媒体播放。例如：

```
<video src="vid.avi" controls />
```

或者：

```
var vid = new Video();
vid.src = "vid.avi";
vid.controls = true;
```

以上代码要求浏览器提供一组默认的控制按钮，以便启动和暂停播放、设置音量大小和拖动播放进度。如果不使用该属性，那么开发人员可以使用表1-5和表1-6列举的JavaScript函数和属性设置自定义控制按钮。

表1-5 控制媒体播放的函数

函 数 名	说 明
play()	从当前位置开始播放媒体，并设置paused属性值为false
pause()	暂停播放媒体，并设置paused属性值为true
load()	重置媒体，并应用相关设置，例如预加载

表1-6 媒体元素属性

属 性 名	值 类 型	说 明
currentTime	integer	设置播放的媒体流的位置
duration	N/A（只读）	指示媒体资源的时间长度，单位为秒
loop	true或者false	设置当播放到媒体流末端时是否重新播放媒体
autoplay	true或者false	设置是否自动播放媒体
muted	true或者false	设置是否能够将音量设置为0.0

为简单和实用起见，表1-6只列出了最有用的属性。如果需查阅所有可用属性，可访问HTML5规范草案<http://dev.w3.org/html5/spec>。

1.9.2 处理不支持的格式

在编写本书时，不同浏览器的audio和video标签并不一定都支持相同格式的音频和视频文件。浏览器不支持某种格式的原因，可能是该种格式太老，或者该格式是其浏览器偏爱的格式的竞争者，或者是由于专利限制原因，此浏览器的所属公司不能处理此格式。为此，媒体标签提供了几种方式处理此问题。

1. 列举多种格式资源

除了指定单个资源文件外，开发人员还可以同时指定多个资源文件，以便让浏览器选择其中一个合适的资源进行播放。在下面的代码段中，video标签中列举出了两个不同格式的资源。如果这两种格式都不支持，或者浏览器连video标签都不支持，则向用户显示反馈消息。

```
<video>
  <source src="video.ogv" />
  <source src="video.avi" />
  <!-- Neither is supported, can show message or fallback to Flash -->
  <div><span>Use a modern browser</span></div>
</video>
```

针对静态内容页面，可以采用列举多种格式资源的方法，但是对于动态内容页面却不合适。在动态页面中，建议使用Modernizr工具。在第2章中，将更加详细地探讨Modernizr，不过首先要了解一些基础知识。

2. 使用Modernizr

Modernizr (www.modernizr.com) 可以在运行时检测浏览器的功能，并将相关属性注入JavaScript对象。为了检测浏览器是否支持播放音频或视频，可以通过检查Modernizr.audio或Modernizr.video的值是否为true来判断。

检查浏览器是否支持播放特定格式媒体文件的方式略有不同。通过检查Modernizr.audio.mp3的值，可以判断浏览器是否支持MP3格式文件播放，但该属性值不是true或false。HTML5规范要求浏览器的返回值要表明其针对该格式的支持程度，这个返回值可以是“probably”、“maybe”或者空字符串。然而，开发人员在条件判断子句中使用Modernizr.audio.mp3时，非空字符串的计算结果为true，空字符串的计算结果为false。

CSS3

CSS3不在本书讲解范围内。感兴趣的读者可浏览有关规范。与HTML5类似，CSS3是在其前身CSS2的基础上进行的扩展。其中添加了一些新特性和编撰了前期提案。例如，引入之前版本中未得到广泛支持的Web字体和语音特性。有关信息可参考<http://www.css3.info>。

1.10 绘画API

HTML5规范中一个有趣的部分是全新的绘画API。通过Cavas、SVG和WebGL技术分别实现位图、矢量和3D的绘画功能。

1.10.1 Canvas

canvas元素起初是Apple公司Webkit框架中的一个扩展。其布局引擎对Safari和Chrome浏览器很有帮助，例如，可用其在Safari浏览器中显示仪表盘组件。后来Opera、Firefox和其他浏览器也采用了canvas元素，并使其最终成为HTML5规范的一部分。Internet Explorer 9（以下简称IE9）的beta版也原生支持canvas元素，但在支持的完整性方面不如前面所列的几种浏览器。

可以将canvas元素简单理解为：一个具有宽度、高度属性的可绘画区域，使用JavaScript来绘制动态的复杂图表或图像。JavaScript语言提供了一组2D绘图函数集合。JavaScript和ActionScript有着密切关系。只要修改少量代码，使用ActionScript绘制的Flash图片或动画代码就可以转换为JavaScript代码。在第5章将详细讲解canvas元素。

1.10.2 SVG

SVG（Scalable Vector Graphics，可扩展矢量图形）是成熟的用于绘制静态或动态图像的W3C标准。在HTML5中，可以不使用对象或内嵌标签，而直接把SVG内联到Web页面文档中。SVG使用各种数学公式来绘制弧线、直线、路径、长方形等各种基本图形元素，并使用这些基本元素组合构建更加复杂的图像。这样的图像能够进行缩放，并且在各种尺度下保持同样的质量。相对于放大后质量下降的位图图像，矢量图像具有很大的优势。

SVG采用了与canvas元素不同的绘画机制。它使用XML文件来实现绘画，而不是单纯地使用代码。由于XML文件中包含很多重复的结点，所以XML不能简洁地表示数据。然而，通过压缩XML文件，可以很大程度上缩减文件大小。与canvas元素一样，SVG图像与用户的交互可以采用JavaScript语言编程实现。在IE9之前，其他版本的IE浏览器均支持名为VML的非标准化的矢量格式。直到IE9发布之后，所有的主流浏览器基本上都支持SVG 1.1。第6章将深入讲解SVG。

1.10.3 WebGL

WebGL是一个实现3D绘画的JavaScript API。通过WebGL，开发者能够评估图像显示

硬件，并全面、细致地控制图像渲染通道。WebGL由Khronos组织进行管理，在语法上，大部分与OpenGL 2.0 ES相同。在本书编写时，IE6及以上版本浏览器、稳定版的Opera和Safari浏览器都不支持WebGL，而稳定版的Firefox和Chrome/Chromium浏览器、开发版的Opera和Safari浏览器都支持WebGL。第7章将深入介绍WebGL。

1.11 使用Microdata传达信息

Web应用程序或解析页面的API能够解释使用了microdata的HTML标记代码，并做出响应。例如，浏览器扩展或者脚本能够解析搜索引擎返回的使用了microdata的标记，从而更好地向患有视弱或色盲的用户传达信息。在microdata之前曾出现过Microformats概念，后者用于实现同样的目的。Microformats和HTML5 microdata的关键不同是标记数据的方式。如代码清单1-6所示，microformats使用class属性来指明对象上具有哪些字段。

代码清单1-6 hCard Microformat示例

```
<div class="vcard">
  <div class="fn">James Williams</div>
  <div class="org">Some Company</div>
  <div class="tel">650-555-3055</div>
  <a class="url" href="http://example.com/">http://example.com/</a>
</div>
```

Microdata可实现同样的功能，但是使用了稍微不同的标记方式。与使用class来标记对象属性不同，microdata使用了itemprop关键字。关键字itemscope标记出一个独立的单元。microdata本质上是键/值对的集合。代码清单1-7是一个microdata示例。itemtype用于标明对象定义和设置有效属性。开发人员可以使用microdata在游戏排行页面中对用户名和得分进行编码，或者使用其来组织操作指南和截图。

代码清单1-7 Microdata 示例

```
<p itemprop="address" itemscope
  itemtype="http://data-vocabulary.org/Address">

  <span itemprop="street-address">1600 Amphitheatre Parkway</span><br>

  <span itemprop="locality">Mountain View</span>,
  <span itemprop="region">CA</span>

  <span itemprop="postal-code">94043</span><br>

  <span itemprop="country-name">USA</span>
```

</p>

1.12 小结

HTML5使得用户与浏览器的交互方式发生了巨大变化。本章介绍了一些有用的重要内容。我们学习了Chrome Frame给浏览器带来的一些HTML5特性，也学习了实现绘画的各种方式。

在本章学习HTML5的过程中，除了学习绘画API外，我们还学习了运行计算密集型任务而不阻塞浏览器的新特性、在应用程序间建立双向通信通道的方法，以及Web应用程序在离线状态下运行的实现。

第 ② 章

配置开发环境

进行HTML5编程开发时，一台具有稳定互联网连接的计算机是很重要的。开发人员所需的工具均可从互联网上免费下载。虽然其中有一些需要许可，但是还是可以找到免费的替代工具。

在本章中，读者将介绍安装构建HTML5应用程序游戏过程中用到的工具。书中将对其中一些工具进行详细介绍。

2.1 开发工具

部分开发人员认为简洁的命令行编辑器很好用，例如 emacs、vim 和 Redcar（笔者偏好的工具）。然而，对于大中型项目而言，随着文件数量的增加，使用 IDE（Integrated Development Environment，集成开发环境）则具有很大的优势，例如，更方便的关键文件和重命名，代码自动提示和语法检查，还有自动构建等。我们将安装 Eclipse IDE，以及此 IDE 运行时所需要的 Java 平台。原因在于，该 IDE 具有很好的扩展性，同时，也因为书中一些示例中使用了 Java 代码。虽然对于基本示例安装 Java 和 Eclipse 是可选的，但是部分示例使用了 Google Web Toolkit（GWT），则必须安装 Java。当然读者也可以选用自己喜欢的工具。

2.1.1 安装Java

如前文所述，Eclipse和GWT运行在Java之上。它们需要Java 5 SDK或者更高版本支持。通常情况下，大多数开发人员都会在计算机中安装最新版的Java。通过在命令行窗口中执行以下命令，可检查是否安装了合适的Java SDK。

```
$ javac -version
```

如果以上命令执行失败，则表明还未安装Java SDK。如果得到类似于下面文字的响应，即版本数字为1.5或者更高，则说明Java SDK已经安装成功。

```
javac 1.6.0_17
```

对于 Mac 系统而言，OS X 10.7（名为 Lion）之前版本都预装了 Java SDK。OS X 10.5 版本（名为 Leopard）的计算机中默认安装有 Java 1.5。那些使用 10.6 版本的计算机则默认安装了 Java 1.6（名为 Snow Leopard）。

Windows 用户可以访问 <http://java.sun.com/javase/downloads/index.jsp>，并下载可执行的 Java SDK。同时，确保选择包含了“JDK”的资源下载。下载完毕后，即可执行文件并安装 Java 了。

在 Linux 系统中安装 Java 要麻烦一些。虽然 Java 的每个发布包在安装方式上略有不同，但是所有主要的包管理器中都包含了 Java 6-compatible binaries。如果找不到，那么可前往 Oracle（收购了 Sun 公司）公司站点下载安装包来安装 Java。

2.1.2 安装 Eclipse IDE 和 Google 插件

Eclipse 是一个多用途的 IDE，但是 Java 开发人员使用较多。Eclipse 采用了模块化的设计，它的插件架构具有向 IDE 添加新特性的功能。由于插件架构的作用，该 IDE 可支持其他很多编程语言，包括 C++、Python、Ruby 和 PHP。同时，其他多个专业 IDE 也以 Eclipse 为基础。本章后面将接触到一些这样的 IDE。

在所有平台上安装 Eclipse IDE 都非常简单、直接。图 2-1 展示了 Eclipse 的加载画面。除了使用安装方式的可执行文件之外，Eclipse foundation（Eclipse 的发明者）还发布了自包含的压缩文件。开发人员可以下载该压缩文件，并将其解压到计算机中，这样可以直接启动 Eclipse。读者可以访问 <http://www.eclipse.org/downloads/> 下载最新版本的 Eclipse。注意确保下载的包是“Eclipse IDE for Java Developers”。

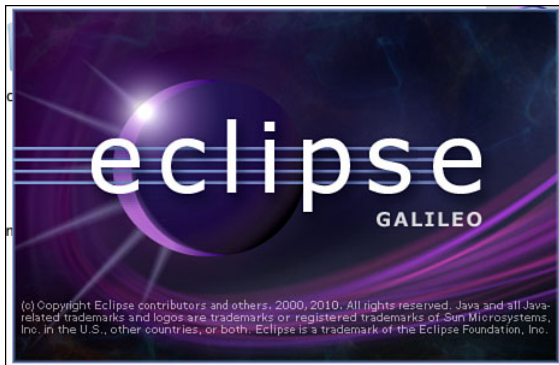


图2-1 Eclipse加载画面

Eclipse 安装完成后，双击可执行文件（一般是 eclipse.exe 或 eclipse）以启动 Eclipse。如果读者是从后文讲解的 Google App Engine 章节跳到这里查阅的，那么可能想要将默认视

图配置为Java视图。为此，需要在菜单中选择Window | Open Perspective | Java项。为了将App Engine整合到Eclipse中，读者需要安装几个软件包。以下是安装步骤：

（1）选择Help | Install New Software菜单项。也可能没有该项，而是“Software Updates”选项。

（2）单击Add按钮，添加一个新软件站点。读者也可以选择从硬盘安装软件包，然而采用添加站点的方式能够使Eclipse例行检查软件更新。

（3）为站点设置一个任意名称，因为这只是一个起组织作用的标识符。如图2-2所示，站点被命名为“Google App Engine Plugin”。在Location文本框中添加以下3个URL中的任意一个，然后单击OK按钮（可以访问code.google.com查看是否有更高的版本）：

- For Eclipse 3.5 (Galileo): <http://dl.google.com/eclipse/plugin/3.5>
- For Eclipse 3.4 (Ganymede): <http://dl.google.com/eclipse/plugin/3.4>
- For Eclipse 3.3 (Europa): <http://dl.google.com/eclipse/plugin/3.3>

（4）如图2-3所示，选中“Google Plugin for Eclipse”和“Google Web Toolkit”项对应的复选框，然后单击Next按钮。之后安装“support for Google Web Toolkit”。读者使用的版本可能比图示的版本高。

（5）确认在Install Details列表中列出了刚才安装的两个插件，然后单击Next按钮。

（6）查阅license内容，并选中相关的单选按钮，则表明读者同意有关条款。由于这个Google插件需要依赖于其他Eclipse组件才可使用，所以看到其他几个插件时不用感到奇怪。所需软件包下载完成后，Eclipse将提示开发人员重新启动程序。重启之后，则一切准备就绪。

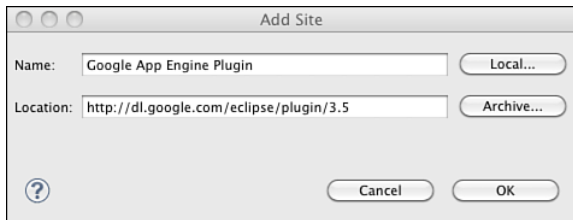


图2-2 Eclipse的Add Site对话框

2.1.3 Google Web Toolkit

Google Web Toolkit包含一组库，允许开发人员使用Java编程语言编写富互联网应用程序，并将其转换为跨平台的AJAX应用程序，而无须担心目标浏览器的不兼容或个别怪异

问题。这样开发人员可以在整个过程中使用同一种语言编写Web应用程序。

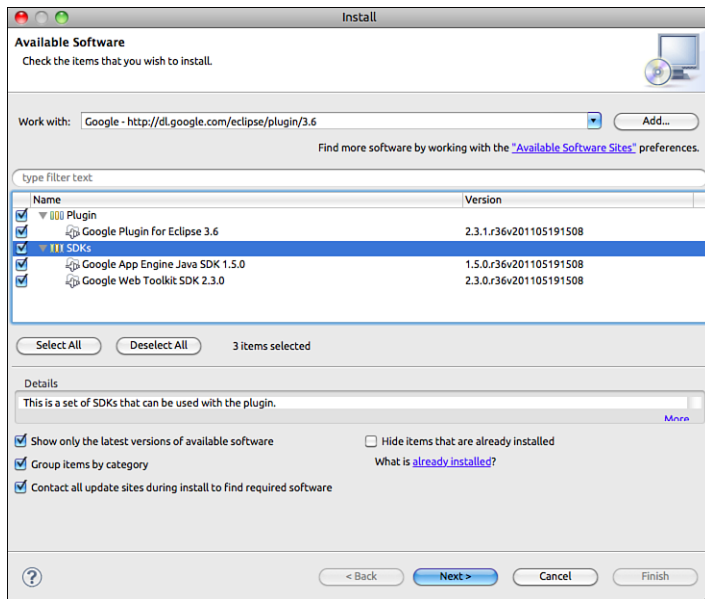


图2-3 Eclipse的Available Software对话框

如果开发人员不喜欢使用Java，那么可以使用基于Python或Ruby且开源的GWT，分别是Pyjamas和RubyJS。这两种实现往往滞后于最新版本GWT，所以笔者不会在示例中直接使用它们。然而，GWT的很多核心代码与Pyjamas代码是相似的。读者可以访问以下两个网址了解关于Pyjamas和RubyJS的更多信息：<http://code.google.com/p/pyjamas>和<http://rubyforge.org/projects/rubyjs/>。

GWT要求在目标计算机中至少安装Java 1.5。读者可以访问<http://code.google.com/webtoolkit/download.html>下载GWT，将其解压到计算机中并安装。Google针对Eclipse的插件支持GWT，读者可以回到前一节查看安装该插件的说明。

第8章将详细说明GWT以及Canvas、WebGL和SVG。

2.2 Web服务器端工具和选项

除非开发人员只是计划将所开发游戏仅供自己使用，或者将其打包成手机应用程序，否则必须采用某种Web服务器进行分发。本章将讨论几种部署游戏的方法。

2.2.1 Google App Engine

Google App Engine是Java和Python的宿主环境，允许开发人员将应用程序部署到Google的基础架构之上。理论上，这种应用程序可以无限扩展。为了使用Google App Engine需要相应的付出一些代价，例如使用非传统的数据存储、对文件系统的访问受限、需要使用特殊的API进行用户授权、收发邮件和数据抓取等。使用App Engine的优点之一是，可以使用Rhino编写应用程序。Rhino是JavaScript在Java平台下的实现，因此开发人员可以在整个开发过程中都使用Rhino。在安装Eclipse后，使用App Engine还有一点额外的好处。即利用Google插件，只需单击几个按钮即可将应用程序部署到App Engine。

如果需要安装Google App Engine或者阅读关于Eclipse插件的信息，可访问<http://code.google.com/appengine/>。

2.2.2 Opera Unite

Opera浏览器从版本10.0开始包含了Opera Unite。这是一个内嵌的Web服务。内嵌Unite的应用程序能够处理文档流、共享相片、网络聊天或者宿主网站。通常实现宿主站点，需要注册域名，找到宿主空间，然后上传应用程序文件。而使用Unite，只要单击一个按钮即可完成这些操作。当启动Unite服务后，在Opera上运行的代理服务器将注册开发人员的计算机。这样，对http://your_device.your_username.operaunite.com/（your_device 是开发人员的计算机名）的访问就可以被正确地路由到开发人员的计算机。Opera的代理服务器能够很容易地建立服务，而不必在路由器上费尽周折。同时，在Unite的Web服务器上可以运行服务端JavaScript，并能够访问文件系统。开发人员甚至可以打包并发布游戏程序，这样，就算用户不连接到对应的Unite服务实例，也可以安装所开发的游戏。

开发人员可以将Unite服务共享给朋友或者测试人员，但是要求Opera浏览器必须处于运行状态。因此，对于要求任何时候均处于服务状态的项目，Unite不是一个好的选择。第11章将讨论打包和发布游戏的方法。

2.2.3 Node.js和RingoJS

RingoJS是一个运行在Rhino平台下的Web框架，实现了CommonJS API规范和协议。考虑到JavaScript发展到现在还没有形成一个独立的规范，于是CommonJS一直在寻求为原本不属于JavaScript应用程序范畴的特性设定标准——例如，访问本地文件系统。

Node.js和RingoJS均实现了CommonJS的部分规范，其中RingoJS在API规范兼容性方面

更胜一筹。其中的一个主要差别是：Node.js可运行在Google的V8引擎上，并使用C++实现扩展，这使得Chrome浏览器对JavaScript的支持更加强大。

第9章将更加详细讲解服务端JavaScript，尤其是Node.js。

2.3 浏览器端工具

开发HTML5应用程序的重要环节是，使用一个支持HTML5规范并具有优秀代码调试工具的浏览器。Google Chrome、Mozilla Firefox（及其派生浏览器）、Apple Safari和Opera这几种浏览器都具有优秀的HTML5兼容性和相应的调试工具。2011年3月发布的IE9浏览器，比其先前版本更加符合HTML5规范，但相对于前面提到的几种浏览器，的确有些姗姗来迟。

2.3.1 Chrome开发人员工具

开发人员使用Chrome浏览器的调试工具，可以动态检查页面的DOM对象，查看资源加载时间，以及运行任意JavaScript代码。用户通过在菜单中选择View | Developer | Developer Tools项，可以打开开发人员工具控制台，从而审查当前页面。图2-4展示了Google.com网页的控制台窗口。选择Elements面板，可以看到嵌套的DOM对象树，以及各个文档元素的相关样式。如果将鼠标悬停在Elements面板中的某个元素标签上，那么将在浏览器窗口中高亮显示该元素标签。当要查看到底是哪个元素的大小或位置有所偏差时，这个功能是非常有用的。

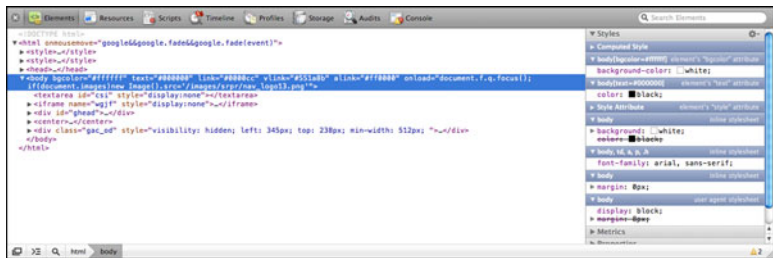


图2-4 Chrome开发人员工具的Elements面板

在开发人员工具中，另外两个对游戏开发人员很有用处面板是Resources面板和Console面板。如图2-5所示，利用Resources面板可逐条跟踪所加载的资源，同时，通过图表可诊断应用程序加载缓慢的具体原因。当使用Resources面板首次跟踪某个新站点时，开发人员可以设置只为当前会话激活面板还是永久激活。使用Resources面板跟踪资源需要一

些额外开销，这会加大页面加载所需时间，使用时应该注意。

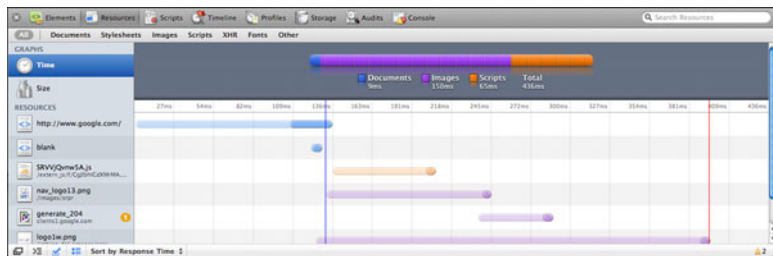


图2-5 Chrome开发人员工具的Resources面板

如图2-6所示，最后一个要介绍的面板是Console面板。该面板对于那些并非偏好直接使用JavaScript进行代码调试的开发人员而言，可以说是天赐之物。因为利用该面板中的`console.log()`命令，可使得开发人员不再编写`println`或者`alert`语句进行代码调试。开发人员还可以在该面板中采用编程方式查看DOM对象，或者运行其他JavaScript代码。

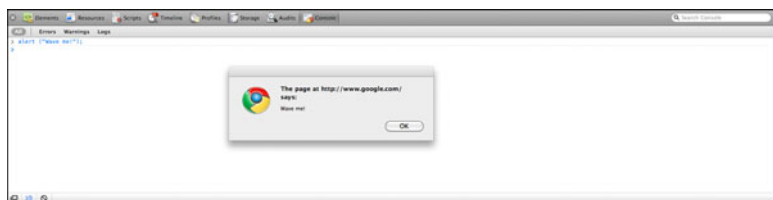


图2-6 Chrome开发人员工具的Console面板

2.3.2 Chrome扩展程序

与Eclipse IDE类似，开发人员可以使用扩展程序来增强和扩展Google Chrome浏览器的功能。这些扩展程序包括RSS订阅器、特定网站优化程序和游戏通知程序等。读者可以访问<https://chrome.google.com/extensions>查看扩展程序的完整列表。下面介绍能使开发应用变得更加容易的扩展程序。

如图2-7所示，通过单击Chrome扩展程序窗口中的Install按钮，并按照窗口提示操作，即可安装Chrome浏览器扩展程序。其中有两个插件JSONView和YSlow对开发人员非常有用。JSONView可完成格式化JSON数据，从而增强数据可读性。YSlow能够分析Web页面结构，并给出一些提高页面性能的建议。

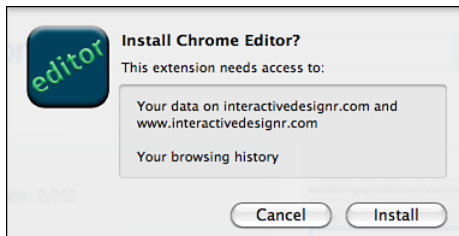


图2-7 安装Chrome扩展程序

2.3.3 Safari开发人员工具

苹果Safari浏览器的开发人员工具和Google Chrome浏览器中的很类似。默认情况下，其开发人员工具对于终端用户是隐藏的。

如图2-8所示，为启用开发人员工具，可以在菜单栏或者图标行中单击Preferences项，并导航到Advanced选项卡，注意需要确保选中“Show Develop menu in menu bar”复选框。该开发人员工具和前文所述Chrome开发人员工具类似，读者可参考前文。

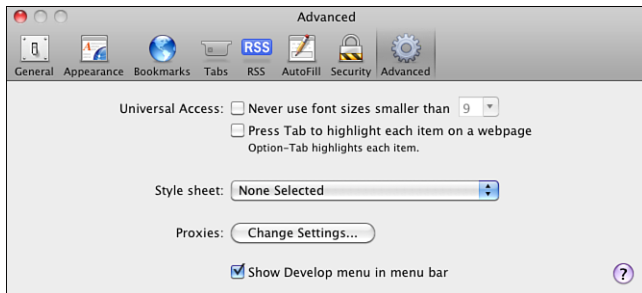


图2-8 激活开发人员菜单

2.3.4 Firebug

Firebug是Mozilla Firefox浏览器的扩展程序，开发人员可使用该工具调试网站的HTML、CSS和JavaScript代码。虽然Firebug最初只是为Firefox浏览器而设计，但是Mozilla也提供了可以在Chrome浏览器中运行的“简化”版Firebug，以作为Chrome开发人员工具的补充。

也许读者已经注意到，Firebug 核心组件及面板与 Chrome 和 Safari 开发人员工具看起来非常相似。与 Chrome 一样，为了支持使用多种编程语言（例如 PHP 和 Python）构建更多功能特性（例如 DOM 操作和解析），已经出现了一些 Firefox 和 Firebug 扩展插件。读者可以访问 <http://getfirebug.com> 查看安装 Firebug 或 Firebug 简版的说明。

2.4 HTML5工具

本节将介绍一些能帮助开发人员轻松地创建游戏资源的工具。其中，Raphaël和ProcessingJS提供了轻量级的图像编程库。

2.4.1 ProcessingJS

ProcessingJS是一个JavaScript程序库。其可作为使用canvas标签绘制基本元素，响应用户交互和绘制、管理图片的抽象层。2008年夏天，John Resig（他也是流行的JavaScript库jQuery的创建者）创建了ProcessingJS。该程序库由名称与其相似的Processing Java库发展而来。二者在函数和API名称上基本相同。同时，很多Java版示例，在不修改任何代码的情况下可以直接转换成JavaScript版本。然而，需要注意的是，ProcessingJS并没有实现Java版本的所有API，这对开发人员基本没有影响。因为对于缺失的特性，使用WebGL实现反而更加合适。

读者可以访问<http://processingjs.org>查看相关示例和API。

2.4.2 Inkscape

Inkscape是一个基于SVG的、成熟的跨平台矢量图形编辑器。其功能可与商业化应用程序Adobe Illustration和CorelDRAW相提并论。虽然最新版本号仅为0.48，但是早在2003年Inkscape发布之前，就已经建立了一个有关该软件的活跃社区。Inkscape的版本号主要用于表明其实现SVG 1.1规范中特性的数量。虽然Inkscape本身并没有完整地支持SVG规范，同时有些人也会说没有哪个应用程序可以做到，但是Inkscape社区已经通过构建插件的方式弥补和增强了相关支持。

读者可以访问<http://inkscape.org>下载Inkscape应用程序。

2.4.3 SVG-edit

SVG-edit是一个基于JavaScript语言的网页形式的SVG创建工具。SVG适用于效果不多的简单绘画。如图2-9所示，SVG-edit界面上有一些设置按钮，其可用于创建文本、基本图形和内嵌图片等。

如果创建诸如棋盘一类的图形，那么使用SVG-edit是最好不过的了。如图2-10所示，SVG-edit不包括文件管理功能，仅显示生成的SVG代码，以使用户将这些代码复制、粘贴到文本编辑器中。

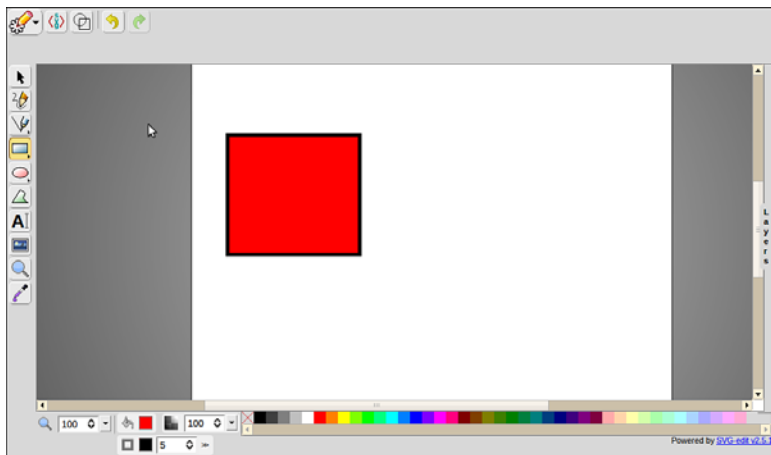


图2-9 SVG-edit界面

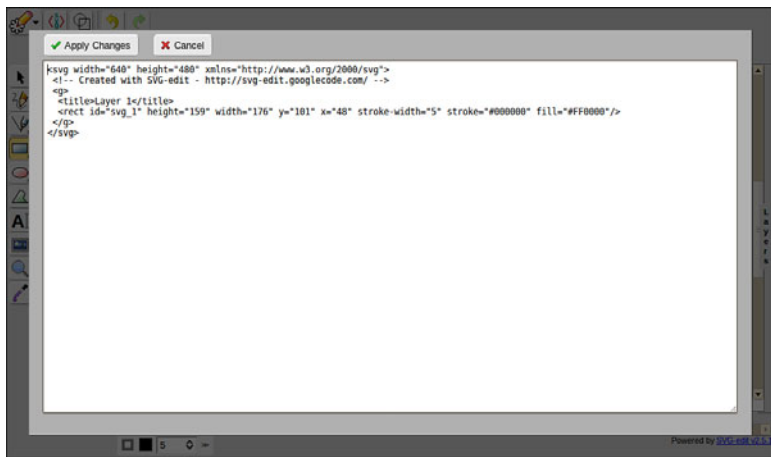


图2-10 SVG-edit代码视图

2.4.4 Raphaël

前面两节内容介绍了两款用于创建 SVG 文件的轻量高级的可视化工具。这些软件适用于构建预先配置好资源的游戏，例如国际象棋游戏。对于需要创建动态资源的 SVG 游戏，或者偏好于编写代码来创建 SVG 图形的开发人员而言，可以使用 Raphaël。Raphaël 是一个 JavaScript 库，其提供了基本图形元素、字体和动画等功能支持。使用 Raphaël 库编写的代码在运行时将被解析为内嵌的 SVG。如果浏览器仅支持矢量标记语言 VML（例如 IE9 之前的低版本浏览器），而不支持 SVG，那么 Raphaël 将使用 VML 的兼容层组件进行渲染。开发人员可以访问 <http://dmitrybaranovskiy.github.io/raphael/> 下载 Raphaël。

第6章中的相关知识将用到这些工具。

2.5 3D建模工具

第7章将讲解使用WebGL创建游戏的方法。开发人员需要使用一些工具来创建游戏资源。在游戏产业中，通常意味着需要用到如3D Studio Max或者Maya这样的软件。这些工具的价格从一千到几千美元不等。这样的价格对于业余游戏开发人员而言很难支付。当然，除了商业矢量图形软件之外，还有一些功能不错的开源软件可供业余或者专业人员选择。

Blender

如图2-11所示，Blender是一个开源的、跨平台3D建模、渲染和动画制作工具。其功能包括服装、骨骼、固定人体模拟，以及纹理、粒子动画及合成等。Blender能够导入和导出多种不同的图形文件格式。同时，还提供Python API，以用于应用扩展。Blender已经在多种电视商业广告、电视节目和专题电影的前期和后期制作中成功使用。

读者可以访问<http://blender.org>下载该软件，并阅读关于它的更多信息。

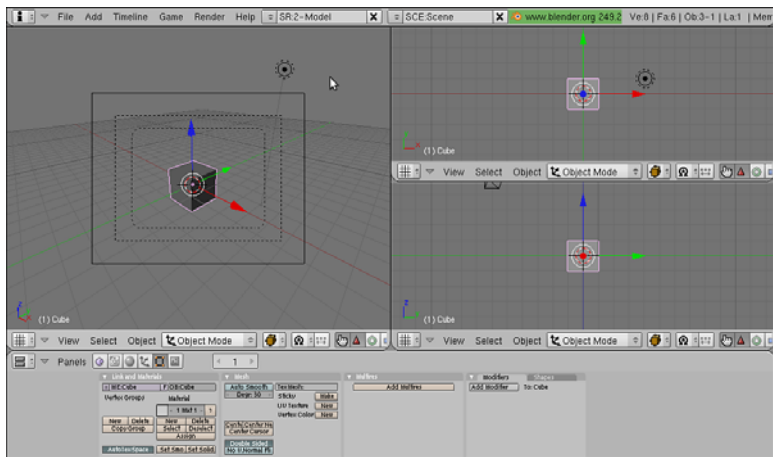


图2-11 Blender软件界面

2.6 小结

在本章中，我们了解了一些用于创建HTML5游戏的工具和框架，以便更好地为用户日后创建HTML5游戏服务。同时，还学习了Java SDK和Eclipse IDE的安装方法，并讲解了一些SVG和3D建模工具以及程序库。本书的游戏示例将用到这些内容。

第 ③ 章

JavaScript 概述

虽然本书主题主要讲解SVG、HTML5 Canvas和WebGL，但是如果没有JavaScript的帮助，那么这些技术本身并不能实现丰富的内容。SVG、Canvas和WebGL需要利用JavaScript实现用户与游戏之间的交互。JavaScript还提供了一些基础库和语言，例如本书后面讲解的GWT和CoffeeScript。另外，本书后文还将说明利用JavaScript调用Node.js来运行服务器端代码的方法。本章介绍JavaScript基础，以及一些对创建游戏有用的工具和库，并使用JavaScript库创建第一个游戏。

3.1 什么是JavaScript

JavaScript是一种弱类型动态语言，其起始于网景通信公司的LiveScript项目。后期，网景公司为其添加了一些Java编程语言的支持，从而重新命名为JavaScript。撇开名字不谈，JavaScript和Java二者之间没有很强的联系，但是二者都受到了C语言的影响，在关键字和结构上有些类似。

3.1.1 JavaScript基本类型

每段JavaScript实现代码中均包括了多个特定类型的对象。通过编写代码块还能够创建其他类型的对象。JavaScript基本类型如下：

- Array——事物的集合。
- Boolean——值为 true 或者 false。
- Function——实现功能的一段代码。
- Number——例如 42、3.54e-3 和 3.141 59。
- String——单引号或者双引号中的一些字符集合，例如 “Hello”。
- Object——由所有其他类型构成的基础类型。
- undefined——引用的对象不存在。
- null——该对象存在但无值。JavaScript 对于对象存在但无值，与不知道对象是什么

二者之间是有区别的。如果执行以下代码：

```
var x = null;
```

那么以上JavaScript代码定义了一个此刻没有值的对象，但是未来将用到该对象。另外，如果执行以下代码：

```
var x;
```

那么以上代码中的x将是undefined类型。

3.1.2 理解算术运算符

大家在很小的时候所学到的基本运算符是加+、减-、除/、乘*、等号=。通常情况下，前4种运算符用于数值计算，而很多编程语言则重新定义了使用这些运算符与用户定义对象进行交互的方式。JavaScript就是其中一种。除非所针对的是数值，开发人员应该尽量避免针对对象使用运算符，否则可能获得意外的结果。例如，利用加号将两个字符串连接起来，利用减号从第一个字符串中移除第二个字符串所包含的字符。这些例子在其他语言中可能奏效，但是在JavaScript中不能这样使用。JavaScript在运算之前，首先判断其是否为数值，如果不是，将抛出异常处理。

3.1.3 理解JavaScript函数

代码清单 3-1 列举了一个简单的 JavaScript 函数。在这段代码中，首先声明了一个名为 HelloWorld 的函数（实现在控制台打印“Hello, World”功能），然后执行该函数。

代码清单3-1 使用JavaScript编写的Hello World

```
function HelloWorld() {  
    console.log("Hello, World!");  
}  
HelloWorld();
```

关键字function要求应用程序运行由大括号封闭部分的代码，这些代码将实现一些功能。在本例中，函数未包含任何输入参数和返回参数。其他一些编程语言则要求设置函数的返回对象类型。然而，在JavaScript语言中，无论函数是否具有返回参数，函数声明的起始部分均使用关键字function。代码清单3-2实现了两个值相加的功能。

代码清单3-2 添加两个对象

```
function add(one, two) {  
    return one + two;  
}
```

以上代码说明了两个新概念：函数的参数和return关键字。函数的参数设置了函数运行时需要提供的数据。关键字return则指明在函数执行结束后返回的对象。下面调用该函数：

```
add(23, 43);
```

执行以上代码返回值是66。开发人员还可以使用其他对象，例如字符串，将两个字符串类型值连接起来以表示某个字符串。

3.1.4 first-class对象函数

在JavaScript语言中，不仅可执行函数，而且还可使用函数在运行时构建和修改内容，例如为变量赋值、返回其他函数的执行结果等。具备后面这些能力（执行代码能力）的函数称为first-class对象。first-class对象函数的结构有些类似其他编程语言（例如Java和C#）中使用关键字class构建类的结构。在那些编程语言中，类定义了基本变量，以及这些类对象包括的函数。甚至在函数中还可以包括不允许进行彼此嵌套的first-class属性。在这个功能方面，JavaScript跨越了类和函数的界限，其中一些函数可返回与自身相同类型的类。代码清单3-3列举了一个Car“类”，其包括了start、accelerate和applyBrakes函数。

代码清单3-3 使用JavaScript构建Car“对象”

```
function Car() {  
    var self = this;  
    self.speed = 0;  
    self.start = function() {  
        console.log("Car started.");  
    }  
    self.accelerate = function() {  
        self.speed = self.speed + 10;  
        console.log("Speed is now:"+self.speed+ " mph");  
    }  
    self.applyBrakes = function() {  
        self.speed = 0;  
        console.log("Brakes applied.");  
    }  
}
```

在以上代码中需要注意的是，代码在声明函数时使用了一种替代方式。即首先定义函数中将用到的变量，然后将匿名函数赋值给该变量。以下代码实例化了一个Car，然后调用start函数：

```
myCar = new Car();  
myCar.start();
```


以上代码与前面的示例之间关键的不同点在于，其没有采用执行函数的方式，而是首先创建实例，然后将其存储在全局变量myCar中。这样就可调用对象的成员函数。另一个新概念是，使用关键字var设置变量。通常使用关键字var比较随意，但其影响变量范围。变量范围决定了变量的可见性。以下代码在Car函数之外声明了名为self的变量。

```
self = "me"
```

值为“me”的self变量是一个全局变量，其在任何地方均为可见。然而，Car函数中的self变量则仅在该函数内可见。关键字this指向的对象依据调用函数的对象发生变化。使用本地self变量可确保在对象内部调用时运行正常。

3.1.5 比较运算符

比较运算符用于判断两个对象是否相等或者二者之间的相互关系。例如，开发人员可使用表3-1中的运算符判断字符串“abc”与“def”是否相等。

```
"abc" == "def"
```

执行以上代码结果为false。

表3-1 比较运算符

运 算 符	说 明
!	对对象进行逻辑反转，例如将true转换为false，或者将正转换为负
&&	表达式两边均为true，则运算返回结果为true
	如果表达式一边为true，则运算返回结果为true
==, ===	如果两个对象相等，那么返回true。使用==时，将先试图强制将两个对象转换为同一类型，然后再进行比较。使用===时，则要求两个对象是相同类型
!=, !==	如果两个对象不相等，则返回true。使用!=时，将先试图强制将两个对象转换为同一类型，然后再进行比较。使用!==时，则要求两个对象是相同类型
<	如果右边的对象值大于左边的对象值，则返回true
>	如果右边的对象值小于左边的对象值，则返回true
<=	如果右边的对象值大于或者等于左边的对象值，则返回true

在对于非数值类型使用比较运算符时有一些特别注意之处：

- true 可看做 1，比看做 0 的 false 大。
- 根据字典中字符的顺序判断是否一个字符串比另一个字符串小。
- 进行数组比较时，需要比较索引以及数组中所有的值。
- undefined == null，返回 true，但是 undefined === null 则返回 false。因为 undefined 和 null 是两种不同的类型。

多数情况下，对非字符串、数字或者布尔型对象使用==、===、!=和!==运算符，可能会得到意外的结果。因此，为了判断地址（是否两个对象指向同一个实例）是否相同，开发人员将需要自行编写函数实现。

3.2 条件和循环语句

比较运算符和表达式向开发人员提供了比较对象的方法，而条件和循环语句则提供了执行代码的方式。下面首先介绍if-else语句，其在执行过程中将进行多次比较运算。代码清单3-4展示了一些使用if语句的代码。

代码清单3-4 if-else语句

```
if (name == "John") {  
    if (age < 18)  
        console.log("Age is less than 18");  
    else if (age < 35)  
        console.log("Age is somewhere between 18 and 35");  
    else console.log("Age is greater than 35");  
} else {  
    console.log("User is not named John");  
}
```

在以上代码中，首先判断存储在变量name中的值是否等于John。如果不是，那么在控制台输出有关消息。如果变量name中的值等于John，则执行一系列的比较运算，直到表达式的值为true，从而最终确定年龄age变量的范围。当if语句比较运算结果为false，才能执行对应的else语句。switch-case语句可看做if语句的变形，可判断单个变量的值是否等于某个给定的值。开发人员可使用一组if语句实现同样的功能，但是使用switch-case语句更加简洁一些。代码清单3-5列出了常见的switch-case语句格式。

代码清单3-5 常见的switch-case语句形式

```
switch (<expression>) {  
    case value1:  
        codeToExecute1();  
        break;  
    case value2:  
        codeToExecute2();  
        break;  
    default:  
        codeToExecute3();  
        break;  
}
```

case语句中列举的值可以是string、number或者boolean类型。关键字break避免了执行后面的case语句。当没有任何case语句与表达式expression匹配时，则执行default语句部分定义的代码。

3.2.1 使用循环控制程序流

最终，程序将执行的是一个又一个的代码块。虽然复制和粘贴代码块也能够让程序执行起来，但是当需要修改代码时，情况则变得相当麻烦，并且所编写代码显得非常业余。根据前后逻辑，开发人员可使用循环语句执行特定的代码块。

1.while循环

当某个表达式不为true之前，都将执行while循环语句中的特定代码块。例如，首先创建一个初始值为0的变量count，然后将count打印输出到控制台，并每次都使count值加1直到count的值等于10。代码清单3-6给出了该while循环语句代码。

代码清单3-6 while循环语句示例

```
var count = 0;
while (count < 10) {
    console.log("The count is now:"+count);
    count++;
}
```

需要注意的是，如果首次计算的条件判断结果为false，则不执行while代码块中的语句。而另一个循环语句do-while则是传统while循环的变形，其确保至少执行一次代码块。

由于编写了一个明显为false的表达式，因此代码清单3-7中列举的do-while循环语句仅执行一次。与while循环语句中首先计算表达式的值不同，其使用关键字do作为起始，而while语句放在最后。

代码清单3-7 do-while循环语句示例

```
do {
    console.log ("This loop executes only once.");
} while (1 != 1);
```

2.for循环

在while和do-while循环中测试判断的是单个条件表达式。for循环则提供了一种更灵活的控制程序流的方式。JavaScript中的for循环语句格式与Java中的for循环语句格式相同，使用关键字for起始，紧接着是初始表达式，测试条件表达式和循环步进表达式。下面的代

码清单3-8使用for循环语句实现了与代码清单3-6中while循环语句相同的功能。

代码清单3-8 for循环语句示例

```
for (var count = 0; count < 10; count++) {  
    console.log("The count is now:"+count);  
}
```

for循环中的每个表达式都是可选的。开发人员可如同while循环语句那样，不对表达式进行初始化，例如，在实现代码中设置步进表达式，或者省略所有3个表达式而创建一个无限循环。代码清单3-9显示了这种变形之后的代码。

代码清单3-9 其他for循环语句示例

```
/* Loop 1 */  
var count = 0;  
for ( ; count<10; ) {  
    console.log("The count is now:"+count);  
    count++;  
}  
/* Loop 2 */  
for ( ; ; ) {  
    /* For demonstration purposes only, don't ever do this. */  
}
```

3.2.2 使用setTimeout和setInterval函数实现延迟执行

JavaScript实际使用单线程执行代码，这种机制影响着该语言的设计思路。读者将在本章后文了解到各种库使用事件来提示应用程序发生的值得注意的事项。为了适应这种单线程机制，JavaScript使用setTimeout函数执行一些将在未来某个时间需要的代码。代码清单3-10实现1000毫秒后在控制台打印输出当前日期和时间功能。

代码清单3-10 setTimeout函数示例

```
setTimeout(function() {  
    console.log(new Date());  
}, 1000  
);
```

执行一次setTimeout函数则实现一次其具体功能。如果需要一遍又一遍地重复执行相同代码，那么开发人员可能需要不断的使用setTimeout函数。然而，还有一种更为简洁的方法是使用setInterval函数。使用setInterval函数替代setTimeout函数时，二者的使用方法是相同的。每经过X毫秒，则执行一次setInterval函数代码。JavaScript将重点优先考虑其执行代码的请求。setInterval函数中的代码也需要合理执行。如果对于长时间执行的任务设置短暂

的时间间隔，那么在第一次执行完成之前，可能会由于执行不断的迭代造成延迟。`clearTimeout`和`clearInterval`函数分别用于取消接下来执行的`setTimeout`和`setInterval`函数。二者并不影响当前正在执行的相关代码。

3.3 使用继承和多态创建复杂对象

如前文所述，JavaScript使用函数来展现面向对象编程语言中类的概念。还有一些其他的编程语言特性，例如可创建和使用继承自其他类的属性和方法的类，能够在多个类的对象中调用相同签名的方法。这些功能特性称为继承和多态。

另外，简单的方法是将继承理解作为一种“is-a”关系。继续本章前文的示例，下面构建一个继承来自 Car 类的 Toyota 函数对象。首先，使用 JavaScript 语言中的点语法为 Car 对象添加一些属性和函数。然后，使用关键字 `prototype` 修改实例对象。如代码清单 3-11 所示，其中声明了两个类：Car 和 Toyota。在声明 Car 类后，开发人员设置了 Toyota 对象的原型 `prototype`（包括属性和函数）。然后，创建 Toyota 对象的结构 `constructor`，并重写 Car 类原本的属性和函数内容。如果在调用 Toyota 对象中的 `go` 函数，那么将执行 Toyota 对象中的 `go` 函数，而非 Car 对象中的 `go` 函数。这就是多态特性。只要创建扩展自 Car 类的对象，开发人员必须确保为基础类型中的属性和函数设置相关内容。正如后面几行代码所示，开发人员可使用关键字 `instanceof` 来确定对象所隶属的类。例如，检测 Toyota 对象是否为 Car 类，运算结果为 `true`。

代码清单3-11 JavaScript继承示例

```
function Car() {
    var self = this;
    self.type = "Car"
    self.go = function() {
        console.log("Going...");
    };
};

Toyota = function() {};
Toyota.prototype = new Car();
Toyota.prototype.constructor = function() {
    var self = this;
    self.type = "Toyota";
    self.go = function() {
        console.log("A Toyota car is going...");
    }
};

Toyota.prototype.isJapaneseCar = true;
```

```
var t = new Toyota();
console.log(t instanceof Toyota);
console.log(t instanceof Car);
```

使用Prototype库实现继承

Prototype (www.prototypejs.org) 是一个JavaScript库，可以使面向对象编程更加容易一些。该库支持继承和多态，但使用方法与已经讲解的直接使用关键字prototype不同。在实现继承特性时，使用Prototype库编码可使得代码更加简洁和易读。使用Prototype库时，主要用到了Class对象及其create函数。如代码清单3-12所示，其使用Prototype库改写了前面的继承示例。当对象实例化时，在initialize函数中写入将要执行的代码。即使将要执行的代码为空，也必须在类中设置相关的代码。当创建Toyota类时，需要向create函数添加一个额外参数，从而表示出该类与父类之间的继承关系。

代码清单3-12 使用Prototype库实现继承

```
var Car = Class.create({
  initialize: function() {
    this.type = "Car";
  },
  go: function() {
    console.log("Going...");
  }
});
var Toyota = Class.create(Car, {
  initialize: function() {
    this.type = "Toyota";
    this.isJapaneseCar = true;
  },
  go: function() {
    console.log("A Toyota car is going...");
  }
});
```

既不使用内置的关键字prototype，也不使用更为简洁的Prototype库，开发人员仍然可以通过在子类中重写的方式来调用父类中的函数。原因在于，当开发人员修改原型prototype（包括属性和函数）时，并未保存所引用对象的副本。虽然有时候在子类中整体处理输入和输出比较有用，但是在某些情况下，例如当需要使用父类来处理一些基本通用数据时，则首先要在相关子类中完成其余的任务。在面向对象语言中，例如Java和C#，开发人员可使用名为super或者base的对象来引用父类。在这些语言中，默认创建了对于父类的引用。开发人员不用修改函数签名即可使用对父类的引用。然而，当使用Prototype库调用任何父类函数时，开发人员都必须在参数列表的开始处添加\$super。幸运的是，这并

没有改变对于函数的调用方式。需要注意的重要变化是，与其他语言不同，这种方式不耗费资源，\$super对象是与父函数关联，而不是与父对象关联。子对象的go函数仅能够显式调用父对象的go函数，如果是其他重载的父函数，那么是隐式调用。代码清单3-13改写了Toyoto类，其在调用Toyota类的go函数之前调用了Car类的go函数。执行Toyota类的go函数，将首先在控制台中打印输出“Going...”，然后输入“A Toyota car is going...”。

代码清单3-13 使用Prototype库调用super函数

```
var Toyota = Class.create(Car, {
  initialize: function() {
    this.type = "Toyota";
    this.isJapaneseCar = true;
  },
  go: function($super) {
    $super();
    console.log("A Toyota car is going...");
  }
});
```

3.4 JQuery简介

JavaScript 工具集中一个重要的工具是 JQuery (<http://jquery.com/>) 及其扩展。这并不是说，JavaScript 的其他框架无法满足开发人员的需求，而是 JQuery 有一个非常活跃的开发社区支持。该社区提供了各种插件，例如部分插件是面向移动开发的，这将在第 10 章有所涉及。JQuery 是目前公认的最流行的 JavaScript 框架。JQuery 中使用了一个全局对象 \$，其可访问和修改文档对象模型 (Document Object Model, DOM)、处理事件、处理 AJAX 请求等。甚至还有一个名为 gameQuery 的插件 (<http://gamequery.onaluf.org/>)，它是一个 2D 游戏引擎。

学习 JQuery 时，其中一个很重要的函数是 ready。在页面文档完全加载完毕之前，将执行 ready 函数内包含的 JavaScript 代码。如代码清单 3-14 所示，在文档加载完毕后，实现在控制台中打印输出“Hello World!”。在实例化 DOM 元素之前，开发人员不能在 ready 函数中引用任何 DOM 元素。

代码清单3-14 JQuery的ready函数

```
<html>
  <head>
    <script type="text/javascript">
      $(document).ready(function () {
        console.log ("Hello World!");
      });
    </script>
  </head>
</html>
```



```
        </script>
    </head>
    <body></body>
</html>
```

3.4.1 使用选择器操作DOM元素

在 JQuery 中，另外一个重要概念是选择器（selector）。选择器为开发人员提供了一种通过命令深入对象树获取对象的方法，例如，获取对象中所有的 <a> 标签，或者获取 <div> 标签中表格内部的第三个 <td> 元素。选择器语法在 JQuery 特定语法的基础上结合了 CSS 和 XPath（一种查询 XML 文档的语法）的语法。表 3-2 列举了一些常见的选择器。

表3-2 一些常见的JQuery选择器

选 择 器	说 明
#id	根据特定ID返回元素
element	返回所有给定类型的元素
.class	返回所有的具有给定CSS class属性的元素
[attribute="value"]	返回所有与特定属性值匹配的元素
:eq(n)	返回给定元素集中特定索引值的元素（索引值以0为起始值）
:even	返回给定元素集中索引值为偶数的元素
:odd	返回给定元素集中索引值为奇数的元素
parent descendants	返回父元素或者选择器的子元素
parent > child	返回父元素或者选择器的一级子元素
selector1,selector2, selector3	返回所有给定选择器匹配到的结果列表

JQuery允许进行选择器链接。例如，通过执行选择器，获取div元素中id属性值为“header”的元素。选择器可写为：

```
$("div[id='header']")
```

另外，开发人员可使用如下的简化代码：

```
$(#header)
```

在通过选择器获取了单个元素或者元素集之后，开发人员可通过一些方法来操作这些元素，具体包括但不限于以下方式：

- 添加、移动或者修改 CSS 样式及属性。
- 添加或者移除子元素。
- 添加动态过渡和效果。

3.4.2 JQuery事件

与焦急的等待事情的发生不同，JavaScript使用事件来提示事情的变化情况。当没有事件需要处理时，那么释放应用程序以完成其他的工作，而不需要浪费时间来不断地检测输入数据。这是一种“不要呼唤我们，我们会叫你”的模式。由于开发人员通常不可能判断何时，何地发生事情，因此这种方式将函数与事件变化绑定起来，当事件发生时，则调用相关函数。JQuery提供了一个常用的bind函数。该函数可将任何事件类型与一些特定的函数实现绑定。常用的事件类型如单击（click）、双击（dblclick）、按下按键（keyup）等。代码清单3-15中列举了两个实现相同功能的方法，二者都为id为“menuBar”的元素绑定了单击（click）事件处理程序。该程序将显示如下HTML代码：

```
<div id="menuBar">/* Stuff here */</div>
```

读者可在第10章中了解这些内容。

代码清单3-15 JQuery事件绑定示例

```
/* Method 1 */
$("#menuBar").bind("click", function() {
    console.log("Clicked on menu bar.");
});
/* Method 2 */
$("#menuBar").click(function() {
    console.log("Clicked on menu bar.");
});
```

3.4.3 JQuery中的AJAX

最终，每个Web应用程序都需要获取非本地服务器上的各种资源。AJAX（Asynchronous JavaScript and XML，异步JavaScript和XML）允许开发人员向文档发送一个请求，同时，当完成了数据获取或者发送时得到提示。本来这种情况的处理要用到浏览器事件，但是JQuery提供了不同的处理方式。基于简化说明的目的，假设不需要使用任何高级的特性，例如验证、消息头和跨域请求。JQuery的API还为常用的服务器端框架和语言中的AJAX请求提供了一些选择。代码清单3-16列举了一个执行AJAX请求的常用方法。如代码所示，其向ajax函数传入了一组键/值对来说明AJAX请求的具体细节。JQuery中的AJAX提供了一些针对常见的数据获取方式，例如GET（get）、POST（post）、以JSON（getJSON）或者脚本（getScript）的快捷方法。同时，JQuery还可在AJAX请求生命周期的不同阶段处理事件。

代码清单3-16 JQuery AJAX: POST和GET

```
$.ajax({
    type: "GET",
    url:"request.html",
    success: function(data) {
        console.log(data);
    }
});
```

3.4.4 跨站点脚本

通常情况下，Web站点只允许来自相同域的AJAX请求，这称为同域策略。跨站点脚本是一种注入式攻击，这种攻击是由于恶意用户利用Web站点设计的缺陷而注入代码造成的，当执行这些代码时就好像是来自同一个域。当发送和接收数据时，这种从其他站点和服务获取资源的能力变得额外重要。

3.5 JSON：另一种JavaScript格式

JSON（JavaScript Object Notation）是一种比XML更为简洁的轻量级数据交换格式，其更易于传输。JSON与XML相比，更易于读写而不易出错。JSON对象由5种类型组成：

- null
- Number
- String
- Array
- Object（大括号包含的一组键/值对）

表3-3列举了一些JSON示例及其对应的XML代码。读者可以看到，JSON代码中未包含>和<符号，从而减少了很多重复性代码，同时还保持了特定的结构。

表3-3 JSON和XML对比

JSON代码	XML代码
<pre>{ "make": "Chevrolet", "model": "Cavalier", "year": 2002 }</pre>	<pre><car> <make>Chevrolet</make> <model>Cavalier</model> <year>2002</year> </car></pre>

虽然技术名没有发生改变，但是JSON作为一种返回类型已经在许多AJAX请求处理中

替代了XML。作为一种基于文本的文件格式，JSON还是有着一些与XML相同的缺点——存储二进制数据效率不高。然而，有多种方法可以化解这一问题，例如采用传递指向二进制数据的URI来替代传递数据本身，或者在有可能的情况下对数据进行base64编码。

JSONP（JSON with padding）是一种针对AJAX同域策略的变通方案。HTML中的<script>标签部分不可应用同域策略。虽然浏览器将执行任意的JavaScript代码，但是开发人员可使用JSONP构建一个不用于返回数据的URI。具有良好格式的JSON数据可视为有效的JavaScript代码。返回的内容通常被封装在被调用的函数中，其内容混合有JSON及显式JavaScript代码，或者未包含JSON数据。代码清单3-17显示了一段HTML代码中的JSONP请求。jsonp部分通常设置为封装有返回数据的函数名。query参数可替代callback函数。重要的一点是，正在请求的站点必须明确函数的命名。

代码清单3-17 JSONP示例

```
<script type="text/javascript"
  src="http://www.example.com/GetTimeLine?UserId=johndoe&jsonp=getData">
</script>
```

JSONP利用了同域策略中的漏洞，而其自身也有可能被利用。如果使用JSONP，那么任何内容都可注入站点中，从而可能泄露敏感数据。

3.6 浏览器外的JavaScript

JavaScript起初是作为一种支持浏览Web页面的工具。近年来，该语言已经扩展到了其他外部平台。下面简要介绍一下。

3.6.1 移动平台

WebOS原由Palm公司为其Pre手机开发，现在由HP（Hewlett-Packard）公司接手。该平台作为移动设备操作系统的一个重要特性是，使用JavaScript作为其主要编程语言。第10章讲解的Appcelerator公司的Titanium平台也使用JavaScript来创建内置的Android和iOS应用程序。

3.6.2 将JavaScript作为中间语言

编程语言和库的一个新发展方向是，可使用备选语言编写代码，并对其进行编译或者将其转换为JavaScript，从而得到浏览器可用的字节代码或者中间语言。这些语言和工具

包括CoffeeScript、Google Web Toolkit (GWT)、脚本语言Ruby/Python、使用Java创建AJAX应用程序的Web框架等。CoffeeScript和GWT将在第8章中讲解。下面是本书未能涉及的语言和工具的列表，笔者希望有兴趣的读者可以查阅：

- Cappuccino/Objective-J (<http://cappuccino.org>)
- Echo3 (<http://echo.nextapp.com>)
- Vaadin (<http://vaadin.com>)
- OpenLaszlo (<http://www.openlaszlo.org>)
- Pyjamas (<http://pyjs.org>)

3.6.3 桌面应用中的JavaScript

自从Mozilla Firefox的Web浏览器发布之后，JavaScript桌面应用程序正逐渐成为主流应用。开发人员现在已经开辟了浏览器的应用空间，他们正在打算开发面向桌面的相关应用程序。本节将介绍一些值得注意的来自高层的框架。

XULRunner (<https://developer.mozilla.org/en/xulrunner>) 是由Mozilla创建的运行时环境，其为Firefox的Web浏览器和许多Mozilla发布的多个应用程序提供了有力支持。这其中包括Mozilla Sunbird (日历/行程表)、Mozilla Thunderbird (电子邮件) 等。XULRunner使用一些C++代码来运行JavaScript引擎SpiderMonkey，但是所有与用户的交互管理均由JavaScript完成。还有一种名为XPI的插件格式，其允许开发人员使用JavaScript和资源来扩展应用程序。XUL和XBL (分别是XML User Interface Language, XML用户界面语言, 和XML Binding Language, XML绑定语言) 用于设置应用程序布局、界面和交互方式，二者完善了XULRunner的核心特性。一些其他的公司和开源项目将XULRunner打包到跨平台应用程序中。其中，一些流行的应用程序包括互联网电视应用程序Miro、媒体库管理应用程序Songbird (其竞争对手是iTunes) 等。

GLUEScript (<http://gluescript.sourceforge.net>) 是一个桌面应用框架，它是wxWidgets针对JavaScript的一项革命。wxWidgets是一个基于C++的跨平台桌面应用框架，其支持绑定多种不同编程语言。因此，开发人员只要学会使用一种编程语言调用wxWidgets应用程序，那么就能够在其他语言中使用wxWidgets，而降低了学习曲线。GLUEScript使用Mozilla的SpiderMonkey引擎作为JavaScript层。在该层之上，所有用户界面代码和逻辑均可使用纯JavaScript。

XULJet (<http://code.google.com/p/xuljet>) 是一个运行在XULRunner之上的桌面应用框架。虽然在后端代码调用的是XUL，但是开发人员可使用特定的基于XUL的语言。这样，

开发人员所编写的将是UI代码和逻辑代码的混合代码。这种方式是否是最合适的方法，因为与之相对的，拥有清晰分层的MVC（Model-View-Controller）架构，但已经超出了本书涉及的范围。然而，这种方式实现了更为简洁和易读的用户界面代码。代码清单3-18和代码清单3-19实现了功能相同的XUL代码和XULJet DSL代码。

代码清单3-18 XUL代码

```
<vbox>
<toolbox>
  <menubar>
<menu label="File" accesskey="f">
  <menupopup>
    <menuitem label="Close" oncommand="window.close()" />
  </menupopup>
</menu>
</menubar>
</toolbox>
<vbox align="center" pack="center", flex="1">
  <description id="descId" >Press the button</description>
  <button label="OK" oncommand='this["desc"].value = this.message' />
</vbox>
<statusbar>
<statusbarpanel flex=1 label="Ready" />
</statusbar>
</vbox>
```

代码清单3-19 XULJet DSL代码

```
vbox({flex: 1},
  toolbox(
    menubar(
      menu({label: "File", accesskey: "f"},
        menupopup(
          menuitem({label: "Close", oncommand: "window.close()"})))))
  vbox({align: "center", pack: "center", flex: 1},
    description({bind: "desc"}, "Press the button"),
    button({label: "OK", oncommand: function() {
      this["desc"].value = this.message}})),
  statusbar(
    statusbarpanel({flex: 1, label: 'Ready...'})))
```

3.7 服务器端JavaScript

许多浏览器都支持JavaScript语言。然而与其他编程语言不同，在JavaScript发展历程中，并没有专门的组织实施引导或者管理。结果，虽然很多应用程序都声称支持

JavaScript，但是其实现方面的兼容性较差。同时，作为浏览器技术，也有一些它不支持的内容，例如，与文件系统交互、文件包管理，以及创建桌面应用程序等。开发人员的目标是，提供一组通用的，开发人员可实现的规范，以便实现兼容的，能够彼此交互和共享代码的应用程序和框架。

在过去几年间，服务器端JavaScript（或者是运行在浏览器之外的JavaScript）作为一种运行Web应用程序的方法，正在变得流行起来。多亏了编程语言Rhino，它是一种可运行在JVM（Java Virtual Machine，Java虚拟机）之上的JavaScript语言，具有多个针对服务器端JavaScript的运行时特性。这样，很多Web服务器内置的运行时特性可支持Java，并允许开发人员使用类似JavaScript的代码调用这些特性。

RingoJS是一种基于JVM的相当成熟的运行时，使用Rhino作为主要编程语言。Node.js是另外一个流行的服务器端JavaScript运行时，其使用Google的V8 JavaScript引擎来执行代码。

模块（module）提供了一种将功能封装到单个文件或者命名空间中的方法，在很多应用程序均可调用模块。与为浏览器编写的函数关联window对象不同，RingoJS和Node.js函数关联的是exports对象。代码清单3-20实现了一个简单的反转字符串的函数。读者可以看到，就函数声明而言，并没有什么变化。而是仅仅添加了一行代码，以通知exports命名空间，该函数将被命名为reverseString。

代码清单3-20 模块示例

```
function reverseString(text) {  
    var reversed = "";  
    for (var i = text.length-1; i>=0; i--) {  
        reversed += text[i];  
    }  
    return reversed;  
}  
exports.reverseString = reverseString;
```

3.8 小结

本章讲解了JavaScript及其相关内容。JavaScript不再只是运行在浏览器中的客户端脚本，其可用做服务器端语言、中间语言、创建移动应用程序等。加速发展的JavaScript引擎意味着，JavaScript将继续渗透到更多的开发领域。具备良好的JavaScript基础是开发HTML5游戏的关键。即使开发人员使用替代语言，但是该语言仍然要被编译为

JavaScript。JavaScript是网络中的通用语言。

3.9 练习

1. JavaScript中实现扩展对象的关键字是什么？
2. Prototype库中的\$super对象具体访问的是什么？
3. 解释运算符==和===之间的不同点。
4. 编写一个每隔5秒钟检测当前是否是午夜12点的函数。提示：使用getHours()和getMinutes()。

Web开发敏捷之道

华章出品



第6版



第4版

JavaScript开发者圣经升级版 Jolt震撼大奖获奖图书升级版

Web开发好书云集



快速掌握HTML5游戏开发技术!

HTML5技术将改变整个互联网和移动游戏。随着新浏览器对它的迅速支持,HTML5将实现如Flash和Silverlight这样的传统技术能够完成的一切,甚至更多。在本书中,开发专家James L. Williams将向读者展示学习HTML5开发过程中遇到的所有相关技术、代码和见解。

在本书中,Williams通过示例详细说明了HTML5的关键性创新功能,其中包括两个应用程序学习示例的整个开发过程。他还向读者展示了设置最新HTML5开发环境的方法、HTML5中最常用的Canvas标签、SVG矢量图形、WebGL 3D,以及针对不同移动系统和平台的开发方法。另外,还讲解了在线游戏设计过程中涉及的性能优化的详细内容。

主要内容:

- 掌握使用HTML5技术创建神奇游戏的方法;
- 设置最新的开源HTML5游戏开发环境;
- 使用JavaScript实现用户与游戏之间的复杂交互;
- 利用基于原型的SGF框架快速构建游戏;
- 使用Canvas标签和界面构建动画和游戏活动;
- 使用SVG矢量图形和JavaScript库RaphaëlJS创建游戏;
- 使用简单的Three.js构建功能强大的WebGL 3D游戏;
- 不使用JavaScript,而是使用GWT或CoffeeScript开发游戏;
- 使用Node.js和WebSockets构建完整的多人游戏服务器
- 使用HTML5作为移动游戏开发的工具;
- 使用离线缓存、代码小型化和其他技术优化游戏性能。

本书是快速掌握HTML5游戏开发的指南——无论是对经验丰富的游戏开发人员,还是初次接触Web或者移动游戏开发的读者,本书都很实用。

客服热线: (010) 88378991, 88361066
购书热线: (010) 68326294, 88379649, 68995259
投稿热线: (010) 88379604
读者信箱: hzsj@hzbook.com

华章网站 <http://www.hzbook.com>

 网上购书: www.china-pub.com

PEARSON

www.pearson.com

上架指导: 计算机/程序设计

ISBN 978-7-111-38034-4



9 787111 380344

定价: 59.00元