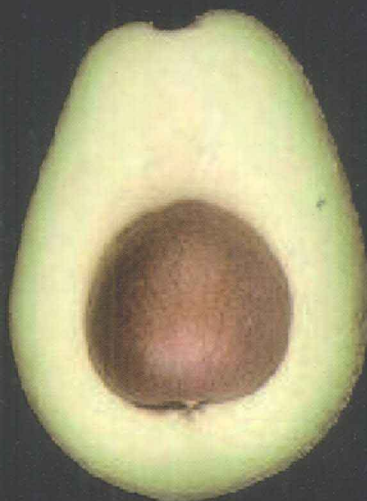


移动与嵌入式开发技术

Beginning iPhone Games Development



iPhone游戏开发 入门经典 ——也适用于iPad

iPhone 与 iPod touch 游戏开发人员的必备指南

Peter Bakhirev
(美) PJ Cabrera 等著
Ian Marsh
郑思遥 译

Apress®



清华大学出版社

移动与嵌入式开发技术

iPhone 游戏开发入门经典 ——也适用于 iPad

Peter Bakhirev
(美) PJ Cabrera 等著
Ian Marsh
郑思遥 译

清华大学出版社

北 京

Peter Bakhirev, PJ Cabrera, Ian Marsh, et al.

Beginning iPhone Games Development

EISBN: 978-1-4302-2599-7

Original English language edition published by Apress, 2855 Telegraph Avenue, #600, Berkeley, CA 94705 USA. Copyright © 2010 by Apress L.P. Simplified Chinese-Language edition copyright © 2011 by Tsinghua University Press. All rights reserved.

本书中文简体字版由 Apress 出版公司授权清华大学出版社出版。未经出版者书面许可,不得以任何方式复制或抄袭本书内容。

北京市版权局著作权合同登记号 图字: 01-2010-6236

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

iPhone 游戏开发入门经典——也适用于 iPad/(美) 巴哈雷夫(Bakhirev, P.), (美) 卡布雷拉(Cabrera, PJ) 等著; 郑思遥 译. —北京: 清华大学出版社, 2011. 7

(移动与嵌入式开发技术)

书名原文: Beginning iPhone Games Development

ISBN 978-7-302-25449-2

I. i… II. ①巴… ②卡… ③郑… III. 移动电话机—游戏—程序设计 IV. TN929.53

中国版本图书馆 CIP 数据核字(2011)第 078887 号

责任编辑: 王 军 谢晓芳 王滋润

装帧设计: 孔祥丰

责任校对: 胡雁翎

责任印制: 李红英

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: 37.25 字 数: 907 千字

版 次: 2011 年 7 月第 1 版 印 次: 2011 年 7 月第 1 次印刷

印 数: 1~4000

定 价: 79.80 元

产品编号: 039297-01

作者简介

Peter Bakhirev 是一位经验丰富的软件开发人员，在 Internet 技术和网络编程方面有十多年的经验，而且还是一名有抱负的作家和企业家。在 iPhone 出现之前，他帮助设计并实现了最大的在线纸牌网站之一。最近，他参与开发了 iPhone 的第一批多玩家游戏中的一款，该游戏名为 Scramboni。

PJ Cabrera 是一名拥有 12 年以上多行业信息系统开发经验的软件工程师，使用过的语言包括 C、C++、Java、PHP、Python、Ruby 和 Objective-C。他住在旧金山湾区，在旧金山一家机密的创业公司从事 iPhone 和 Rails 开发工作。

Ian Marsh 是独立游戏工作室 NimbleBit 的合伙创始人，这间工作室位于加利福尼亚州的圣地亚哥市。自 App Store 出现以来，他就一直在为 iPhone 开发游戏，他的成功作品包括排名第一的儿童游戏“Scoops”和排名第一的免费游戏“Hanoi”。在开发游戏之外，Ian 喜欢阅读有关科学的书籍，喜欢发布有关游戏开发的微博，还喜欢信手涂鸦。

20 世纪 70 年代，Apple II 发布后不久，**Scott Penberthy** 就开始编写代码了。他对编写软件的热爱随着 MIT 的奖学金一同绽放。在 MIT 大学，他编写了一个多玩家的在线游戏，让学校古老的计算机不堪重负。毕业之后，Scott 在 IBM 研究院获得了一份工作，IBM 的 Web 产品和服务皆诞生于此。在 20 世纪 90 年代构建大型网站的职位上升迁几年之后，2005 年他毅然选择离开，回到了他热爱的编程事业。现在作为一名成功的企业家，Scott 正在运营纽约市的一家 App 开发工作室。

Ben Britten Smith 已经在 Apple 平台上从事了 15 年的软件编写工作。最值得一提的是，他还曾经因为他的故事片作品和基于 Mac 的悬挂照相机控制系统获得过奥斯卡技术成就奖。之后，他把他的注意力从大屏幕转向了小屏幕。

他的第一款 iPhone 游戏“SnowDude”在 SDK 发布后几个月就登上了 App Store。从此以后，他就为多个客户开发了数十款应用程序，包括 Snowferno 游戏、“Mole - A quest for the Terracore Gem”获奖作品，以及“Gambook Adventure”系列游戏。Ben 和他的妻子 Leonie 生活在澳大利亚的墨尔本。

当感到充满了泡面和使用政府补贴的公交卡的生活方式过于奢侈的时候，**Eric Wing** 从加州大学圣地亚哥分校获得计算机工程专业的硕士学位毕业了，也就是“9·11”事件的前几天。在接下来充满挑战的世界中，他做过各种各样的工作，从卫星系统的自动化测试，到使用大量不同操作系统和编程语言的科学可视化。然而，由于他非凡的才能，他觉得没有钱的工作才能让他更为努力地工作，因此他开始致力于开源项目。他是很多开源项目的贡献者，例如 SDL(Simple DirectMedia Layer)、OpenSceneGraph，以及 Lua/Objective-C Bridge(及其后继项目 LuaCocoa)。当他被邀请合著本书时，他怎么会拒绝这个几乎没有报酬且更困难的工作呢？这简直就是天作之合！

关于技术审校者

本书的各个章节是由本书的 6 个作者相互审阅的。Peter Bakhirev 审阅了 Ian Marsh、PJ Cabrera 和 Ben Britten Smith 撰写的章节。Eric Wing 审阅了 Peter Bakhirev 撰写的章节。Ben Britten Smith 负责审阅了 Scott Penberthy 撰写的章节。

致 谢

Peter Bakhirev

撰写这本书是一个很多年前开始的一个旅途中的一部分。在很多比我优秀、比我善良、比我聪明的人的帮助下，这段旅途才得以顺利完成。谢谢我的父亲，谢谢您为我组装了第一台计算机，谢谢您给我 Basic 方面的磁带，谢谢您的智慧(以及所有的游戏)。谢谢 Mike，谢谢您给了我参加面试的机会，谢谢您教会了我现在所知道的全部编程知识，谢谢您一直以来的支持(您总是说“我们应该找时间再来一次”)。谢谢 Lenny，谢谢您相信我能够做得到(否则我还得过着朝九晚五的工作，而没有时间撰写这本书了)。谢谢我的母亲和 Olya，谢谢你们的鼓励，谢谢你们每天都关心这本书的进度。Lena，谢谢你的信任，我爱你。谢谢 Keith Shepherd 把我介绍给 Clay 和 Apress 出版社。谢谢 Clay Andres、Kelly Moritz、Douglas Pundick、Marilyn Smith，以及 Apress 出版社的所有员工，没有您们的帮助这本书不可能顺利出版。

PJ Cabrera

我要感谢 Peter Bakhirev、Jeff Pepper 和 Marilyn Smith，谢谢你们对我撰写的章节提供的帮助。特别要感谢 Kelly Moritz 和 Dominic Shakeshaft，谢谢你们的耐心。非常感谢 Clay Andres 和 Douglas Pundick 的支持。最后，感谢 SNAP 团队的好兄弟们 Kevin、Lenny、Ernesto、Cesar、Alejandro、Javier、Xavier 以及 Edwin。

Ben Britten Smith

Ben 要感谢 Tin Man Games 的 Neil，谢谢他允许我使用他的素材。

Eric Wing

我要向那些审阅我的章节的伟大志愿者们表示绵绵不断的谢意。

首先，我要感谢 Daniel Peacock 和 Ryan Gordon，他们利用 OpenAL 和通用音频方面的专业技术知识帮我找出了所有的错误和纰漏。还要谢谢 Ian Minett 提供了 Xbox 的 OpenAL 实现的细节，以及 Garin Hiebert 为我找出了一些 OpenAL 方面的模糊信息。

接下来我想要感谢 Josh Quick、Victor Gerth、Carlos McEvilly 和 Wang Lam 以一般读者的角度来审阅我的章节，让我在把书提交到出版社之前得以改进很多内容。

最后，还有一些不愿意透露姓名的人。不论如何我都要感谢他们，因为他们的贡献太大了。

当然，我还要感谢我的合作者们，以及 Apress 的人，谢谢他们的支持。

前言

好奇的读者，你们好！我是 Peter，我想向您介绍一下我的合著者 Ian、PJ、Scott、Ben 和 Eric。我们聚在一起只有一个原因：为了帮助您学习如何开发出色的 iPhone 游戏。您也许注意到了本书书名中的“入门”二字。下面是对入门的解释：您可以开发游戏，即使以前从来没有把自己当作“游戏开发人员”。尽管“开发游戏”一开始看上去很高深的样子(像火箭科学一样)，可实际上它并非如此(尽管本书要构建的一款游戏包含一枚火箭，以及壮观的爆炸效果，还包含外太空的声音和邪恶的外星人)。我们相信，任何人都能学到有用的技能，我们会帮助您循序渐进，并解释遇到的所有内容。

说到创意和游戏，我们有很多可以玩的游戏。本书包含了 6 款完全可玩的游戏，这些游戏有助于您的开发工作。在学习的过程中，您会懂得如何构建带有音乐、音效且支持多玩家和网络 2D 和 3D 游戏。但是“构建什么”至少与“如何构建”的问题一样重要，您也会找到很多有关如何设计好玩的游戏的讨论。

如果您以前从来没有从事过 iPhone 的开发工作，则需要一本速成教程，用于学习如何使用开发工具，我们也对 Objective-C 和 Xcode 做了简单的介绍。不过我们强烈推荐 Dave Mark 和 Jeff LaMarche 合著的 *Beginning iPhone 3 Development* 一书，在这本书中您可以更加深入地学习 iPhone 开发环境。如果您需要详细了解程序设计(特别是 C 语言和 Objective-C 语言)的相关内容，可以参阅 Dave Mark 撰著的 *Learn C on the Mac* 一书以及 Mark Darlymple 和 Scott Knaster 合著的 *Learn Objective-C on the Mac* 一书。

本书的编写、编码和调试的过程充满了乐趣，我们也希望您能够发现其中的乐趣并且参与到其中。祝您好运，在 App Store 中见！

作者团队敬上
Peter Bakhirev

目 录

第 1 章	革命性的游戏平台：随时随地，人人可以游戏	1
1.1	无处不在的 iPhone	1
1.2	iPhone 的广泛吸引力——每时每刻都在诞生新的玩家	2
1.3	iPhone 的用户界面——手柄终结者	3
1.4	iPhone 的连接性——和其他玩家一起玩	4
1.5	iPhone 中的用户数据——个性化体验	5
1.6	iPhone 设备的性能——强大的多媒体设备	6
1.7	iPhone 的开发包——人人皆可拥有	7
1.8	创新——来自小开发商的大创意	8
1.9	本章小结	9
第 2 章	iPhone 游戏开发：iPhone 工具箱一览	11
2.1	开发工具和开发环境	11
2.2	UIKit	12
2.3	Quartz 2D 和 Core Animation	13
2.4	OpenGL ES	13
2.5	音频 API	14
2.6	网络	15
2.7	本章小结	16
第 3 章	在小屏幕上移动图像——使用 UIKit 控件	17
3.1	Cocoa Touch 简介	17

3.1.1	Objective-C 语言	18
3.1.2	Cocoa Touch 和 UIKit 框架	25
3.2	构建一款简单的游戏	27
3.2.1	创建一个 Xcode 项目	27
3.2.2	创建 IVBricker 用户界面	29
3.2.3	华丽的图像才够格	34
3.2.4	完成了吗	57
3.3	应用程序委托事件	57
3.3.1	应用程序终止	58
3.3.2	应用程序中断	58
3.3.3	低内存警告	59
3.4	保存和加载游戏状态	59
3.5	动画图像	63
3.5.1	使用 UIImageView 的动画属性	63
3.5.2	通过 NSTimer 实现动画	64
3.5.3	通过 CADisplayLink 实现动画	66
3.6	本章小结	68
第 4 章	射击、命中与得分	71
4.1	Quartz 2D 游戏概述	71
4.2	所有的艺术家都需要画布	73
4.3	用 Quartz 2D 创建第一个图形	79
4.3.1	保存和还原上下文	80
4.3.2	添加颜色	82
4.4	Sprites	83
4.4.1	创建 Sprite 类	83

4.4.2	使用 Sprite 类	89
4.5	哪一面朝上	91
4.5.1	切换到水平方向	91
4.5.2	将原点居中	93
4.6	矢量图	94
4.6.1	创建 VectorSprite 类	94
4.6.2	使用 VectorSprite 类	97
4.7	翻页动画	100
4.7.1	创建 AtlasSprite 类	101
4.7.2	修改 Sprite 类	105
4.7.3	使用 AtlasSprite 类	107
4.8	抬头显示	110
4.8.1	创建 TextSprite 类	111
4.8.2	使用 TextSprite 类	116
4.9	Asteroids 游戏架构	117
4.9.1	Asteroids 游戏循环	117
4.9.2	Asteroids 模型	118
4.9.3	Asteroids 视图	120
4.9.4	Asteroids 游戏控制器	121
4.10	本章小结	123
第 5 章	通过 Core Animation 实现动画效果	125
5.1	Core Animation 示例项目概述	125
5.2	UIView 的动画	127
5.2.1	简单移动	130
5.2.2	动画曲线	131
5.2.3	反向和重复	132
5.2.4	延时、缓入和缓出	134
5.2.5	UIView 变换	135
5.2.6	UIView 过渡	137
5.3	连续变化的 Core Animation 层	139
5.3.1	隐式动画	140
5.3.2	定时函数	140
5.3.3	层动画过渡	141
5.4	本章小结	145

第 6 章	OpenGL 基础：理解 OpenGL API	147
6.1	什么是 OpenGL ES 和为什么要使用 OpenGL ES	147
6.2	理解 3D 世界	148
6.3	矩阵的基本概念	149
6.3.1	综合平移、旋转和缩放	151
6.3.2	矩阵类型	151
6.3.3	有状态的 API	152
6.4	渲染的基本概念	153
6.5	基本游戏模板	154
6.6	将 CAEAGLLayer 包装在 EAGLView 视图中	155
6.6.1	第一步：初始化方法	155
6.6.2	帧缓冲区、渲染缓冲区和深度缓冲区	158
6.6.3	进入 OpenGL 的世界	160
6.6.4	绘制和渲染	164
6.7	如何通过 OpenGL 绘制物体	165
6.7.1	场景对象和网格对象	165
6.7.2	压入和弹出矩阵	168
6.7.3	在场景中放置对象	169
6.7.4	在 3D 空间中定义对象	171
6.8	游戏循环和定时器	174
6.9	输入控制器	178
6.10	应用程序委托	179
6.11	本章小结	180
第 7 章	综合所学知识：创建一款 OpenGL 游戏	181
7.1	游戏设计——Space Rocks!	181
7.2	以模板为基础开始进行游戏设计	183
7.3	屏幕翻转	184
7.4	升级为 3D 点	186
7.5	添加按钮	188
7.5.1	创建按钮对象	189

7.5.2 使用顶点数据	190	8.2.2 其他动画	255
7.5.3 存储按钮	192	8.3 从 2D 迈向 3D	256
7.5.4 触摸检测	194	8.3.1 多了一个维度意味着 什么	256
7.5.5 连接按钮	199	8.3.2 3D 模型从何而来	257
7.6 构建更好的宇宙飞船	200	8.3.3 从建模工具到屏幕显示	258
7.6.1 移动起来	200	8.3.4 什么是法线	258
7.6.2 添加宇宙飞船	201	8.3.5 标准化为 GL_TRIANGLES	259
7.6.3 添加和删除场景对象	203	8.3.6 纹理加上模型	260
7.6.4 飞出屏幕边缘	205	8.3.7 影子展示出形状	261
7.7 Space Rocks!	206	8.3.8 深度缓冲区和背面剔除	264
7.8 添加导弹	209	8.3.9 碰撞检测的改进	266
7.8.1 发射导弹	210	8.4 粒子系统给游戏带来了 生机	267
7.8.2 删除不需要的导弹	211	8.4.1 什么是粒子系统	267
7.9 制作更好看的按钮	212	8.4.2 大量随机数	268
7.10 碰撞检测	214	8.4.3 粒子系统的根本: 粒子	268
7.10.1 什么是碰撞	214	8.4.4 粒子发射器	269
7.10.2 检测碰撞的技术	214	8.4.5 系统调整	276
7.11 碰撞石块	218	8.4.6 其他对象的粒子系统	277
7.11.1 质心和半径	218	8.4.7 What the Future Holds: Shaders	278
7.11.2 碰撞对象和被碰撞 对象	220	8.5 本章小结	278
7.11.3 再论碰撞检测	230		
7.12 本章小结	232		
第 8 章 图册、Sprite 和粒子系统	233	第 9 章 核心音频简介	279
8.1 纹理和纹理图册	233	9.1 核心音频提供的音频服务	279
8.1.1 什么是纹理, 为什么要 使用纹理	234	9.1.1 音频单元	280
8.1.2 将图像数据载入 OpenGL	234	9.1.2 音频文件服务	280
8.1.3 绑定纹理	238	9.1.3 音频文件流式服务	280
8.1.4 使用 UV 坐标	239	9.1.4 音频转换服务	280
8.1.5 获得纹理贴图的四边形	240	9.1.5 扩展的音频文件服务	281
8.1.6 认识纹理图册	241	9.1.6 音频会话服务	281
8.1.7 推陈出新	245	9.1.7 系统声音服务	281
8.1.8 更好的用户界面	246	9.1.8 音频队列服务	281
8.1.9 带有纹理的颜色	248	9.1.9 AVFoundation	281
8.2 Sprite 动画	249	9.1.10 OpenAL	282
8.2.1 与帧率无关的设计	251	9.2 核心音频框架	283
		9.3 编解码器和文件格式	284

9.3.1 核心音频所支持的 编解码器	284	10.4.5 生成声音源	322
9.3.2 核心音频支持的文件 格式	285	10.4.6 生成数据缓冲区	323
9.3.3 通过 afconvert 转换 格式	286	10.4.7 从文件中加载声音 数据	323
9.3.4 硬件加速的编解码器：受限 和非受限的编解码器组	286	10.4.8 将声音数据提交至 OpenAL 数据缓冲区	327
9.3.5 有关编解码器和文件格式 的建议	287	10.4.9 将数据缓冲区关联至 声音源	329
9.4 警告和振动：系统声音服务 介绍	288	10.4.10 播放声音	329
9.4.1 不使用系统声音服务作为 通用音效的情形	289	10.4.11 关闭并清理	329
9.4.2 系统声音服务示例	290	10.5 更多问题和细节补充	331
9.4.3 关于异步播放的注意 事项	295	10.5.1 添加更多的声音	332
9.5 设置音频策略：音频会话服务 介绍	296	10.5.2 需要注意的问题	336
9.5.1 使用音频会话服务的样本 代码和过程	297	10.5.3 OpenAL 错误检查	336
9.5.2 检测硬件并获得属性	299	10.5.4 音频会话中断	338
9.6 通过 AVFoundation 利用 Objective-C 轻松播放音频	300	10.5.5 iOS 的 OpenAL 扩展	341
9.7 本章小结	310	10.5.6 性能说明	341
第 10 章 通过 OpenAL 发出声音	311	10.5.7 OpenAL 声音源的限制	342
10.1 OpenAL 概述	311	10.6 声音资源管理器：修复 设计	344
10.1.1 OpenAL 支持的特性	311	10.6.1 资源管理器概述	345
10.1.2 OpenAL 简史	312	10.6.2 初步清理	346
10.2 Eric Wing 的故事和音频涵盖 的内容	315	10.6.3 声音文件数据库(缓存 系统)	352
10.3 内容指引	316	10.6.4 OpenAL 声音源管理(预留 和回收)	356
10.4 在 OpenAL 中设置基本 声音	316	10.6.5 与 Space Rocks! 游戏的 集成	360
10.4.1 设置音频会话	318	10.6.6 处理所有可用的声音源 都耗尽的情况	370
10.4.2 打开设备	318	10.6.7 最后的润色	371
10.4.3 创建上下文	321	10.7 本章小结	373
10.4.4 激活上下文	322	第 11 章 3D 音频——将声音变为游戏 声音	375
		11.1 OpenAL 的设计：声音源、 缓冲区和听者	375
		11.2 OpenAL 中 3D 音频的 局限性	377

11.3 将听者整合进 Space Rocks! 游戏	378	12.3 音频流式处理	418
11.3.1 创建听者类	378	12.3.1 基于 AVFoundation 的 Space Rocks! 游戏背景 音乐	419
11.3.2 挑选听者	379	12.3.2 OpenAL 缓冲区队列 介绍	424
11.4 在声音中添加位置	380	12.3.3 基于 OpenAL 的 Space Rocks! 游戏背景音乐	432
11.4.1 处理创建时的初始位置	382	12.3.4 Space Rocks! 游戏的 OpenAL 语音	445
11.4.2 禁用距离衰减	383	12.3.5 基于音频队列服务的 Space Rocks! 游戏背景音乐	454
11.5 听者朝向	384	12.3.6 完美的全连	456
11.5.1 右手坐标系统和右手 定则	385	12.4 音频捕捉	456
11.5.2 单位圆、极坐标转换直角 坐标、相位平移和三角恒 等式	386	12.4.1 音频捕捉 API	457
11.5.3 集成至 Space Rocks! 游戏	388	12.4.2 AVFoundation: 使用 AVAudioRecorder 录制 到文件	458
11.6 声音源方向和圆锥	389	12.4.3 OpenAL: 捕捉示波器	463
11.6.1 内圆锥、外圆锥和 过渡区	390	12.5 回到 OpenGL	468
11.6.2 实现	391	12.5.1 顶点缓冲区对象	469
11.7 速度和多普勒效应	392	12.5.2 有关 OpenGL 和 OpenAL 优化的注意	470
11.7.1 速度和缩放因子	393	12.6 本章小结	471
11.7.2 多普勒效应示例	394	第 13 章 iPhone 游戏联网简介	473
11.8 距离衰减	396	13.1 了解网络	473
11.8.1 衰减模型	396	13.1.1 网络接口	473
11.8.2 回到 Space Rocks! 游戏	401	13.1.2 TCP/IP	474
11.9 使用相对声音属性选择性的 禁用 3D 效果	404	13.1.3 Bonjour	475
11.10 本章小结	406	13.2 iPhone SDK 和联网	475
第 12 章 流式媒体	409	13.2.1 套接字和连接	475
12.1 音乐与其他音频	409	13.2.2 BSD 套接字 API	476
12.2 iPod 音乐库(Media Player 框架)	411	13.2.3 CFNetwork	476
12.2.1 在 Space Rocks! 游戏中 播放 iPod 音乐	413	13.2.4 NSNetServices	476
12.2.2 添加媒体项选择器	415	13.2.5 GameKit	476
12.2.3 摇晃(简易的加速度 传感器摇晃检测)	417	13.3 本章小结	477
		第 14 章 对战游戏	479
		14.1 见识 Pong	479

14.2	使用 Peer 选择器寻找一个 人类对手	480	15.5.2	选择网络消息的格式	546
14.2.1	运行界面	484	15.5.3	实现通信	548
14.2.2	工作原理	487	15.5.4	整合起来	551
14.3	建立连接	487	15.6	实现游戏服务器	552
14.4	发送和接收消息	491	15.6.1	管理玩家	552
14.4.1	投掷骰子	492	15.6.2	代码整合	555
14.4.2	准备好, 继续	500	15.6.3	初始化	561
14.4.3	击球和失球	501	15.6.4	玩家加入和离开	561
14.4.4	让板子活跃起来	510	15.6.5	开始和结束游戏回合	563
14.5	游戏结束: 处理断开 连接	515	15.6.6	收集和处理答案	565
14.6	本章小结	516	15.6.7	整合起来	566
第 15 章	多人游戏	517	15.7	本章小结	566
15.1	8 乘以 3 等于几	517	第 16 章	连接 Internet	567
15.1.1	起始点	517	16.1	挑战	567
15.1.2	目标	519	16.1.1	不能使用 Bonjour	567
15.1.3	结构	519	16.1.2	不能通过 GameKit 实现 点到点的游戏	568
15.2	创建连接	521	16.1.3	服务器中断	568
15.2.1	连接对象和流对象的 简介	521	16.1.4	滞后	568
15.2.2	连接初始化	522	16.1.5	数据格式问题	569
15.2.3	关闭和清理	524	16.1.6	其他问题	569
15.2.4	读取数据	524	16.2	在线游戏基础知识	570
15.2.5	写入数据	527	16.2.1	客户端/服务器游戏	570
15.2.6	处理流事件	528	16.2.2	不通过 Bonjour 连接到 游戏服务器	571
15.2.7	完整画面	529	16.2.3	点到点的游戏	571
15.3	套接字服务器	529	16.2.4	有关重新开发	572
15.3.1	SocketServer 类	530	16.2.5	沧海一粟	573
15.3.2	套接字服务器初始化	531	16.3	引入社交元素	573
15.3.3	通过 Bonjour 发布服务	534	16.3.1	在线共享最高得分	573
15.3.4	开始和停止	534	16.3.2	虚拟追击	574
15.4	通过 Bonjour 查找服务	536	16.3.3	聊天	575
15.4.1	查找服务器	536	16.4	本章小结	575
15.4.2	连接至服务器	538	第 17 章	整合一切, 享受乐趣	577
15.4.3	最后的细节	540	17.1	本书涵盖的内容	577
15.5	实现游戏客户端	544	17.2	一些游戏设计的技巧	578
15.5.1	跟踪逻辑	544	17.3	本章小结	578

第 1 章

革命性的游戏平台：随时随地， 人人都可以游戏

iPhone 平台已经极大地改变了下一代移动游戏的前景。iPhone 平台创造了许多第一，正是 iPhone 平台多才多艺的能力使得 iPhone 大大超越了传统的移动游戏平台。由于 iPhone 的独特性、连接性、个人集成性，流行以及新颖的界面，因此 iPhone 已经成为时下最火爆、最激动人心的开发平台。

1.1 无处不在的 iPhone

由于 iPhone 本身就是一个移动电话，而且还是数字音乐播放器，因此绝大多数 iPhone 和 iPod touch 的拥有者都会每时每刻随身携带着这个设备。由于 iPhone 将电话、音乐播放器和游戏机整合在一个纤小的软件包内，因此人们每天早上再也不需要考虑今天要携带什么设备了。因此，iPhone 可以称为一个开创性的游戏平台。

对于 iPhone 或 iPod touch 的用户，随时都可以打开游戏——不管在加油站等待加油还是在飞越大西洋的航班上。随处看看，就可以发现 iPhone 正在迅速扩张而且 iPhone 游戏的市场也在迅速扩张。对于移动中的游戏玩家，iPhone 是最适合随身携带的游戏设备了。

由于玩家随手就可以开始玩游戏，因此游戏开发人员既可以开发那种很快的“即开即玩”的游戏，也可以开发需要耗费 30 分钟或更多时间玩的“任务”型游戏。例如，Veiled Game 开发的 Up There(如图 1-1 所示)是一个非常好的、消磨时光的游戏，这个游戏每回只需要几分钟的时间即可，适合不会花大段时间玩游戏的休闲玩家。另外一类玩家喜欢消耗时间玩一些非常吸引人的游戏，例如 Electronic Art 开发的 SimCity(如图 1-1 所示)。这两类玩家都有可能拥有 iPhone，因此开发人员可以设计迎合这两类人兴趣的游戏。有一点是肯定的：作为 iPhone 游戏的开发人员，您可以期望您的用户在每一天睁开眼睛的那一刹那就开始带着您的游戏到处跑了。

尽管用户可能会一直带着您的游戏，但这并不意味着他们会花更多的时间玩游戏，不过这仍然可以帮助您开发游戏的市场。高质量 iPhone 游戏最好的广告方式之一就是玩家的口头传播。iPhone 用户通常都喜欢和其他 iPhone 用户联系。他们随身携带着他们完整的游戏库，这样玩家随时可以向其他玩家炫耀他们最喜欢的游戏，而他们的朋友也可能立即自己也购买一款。

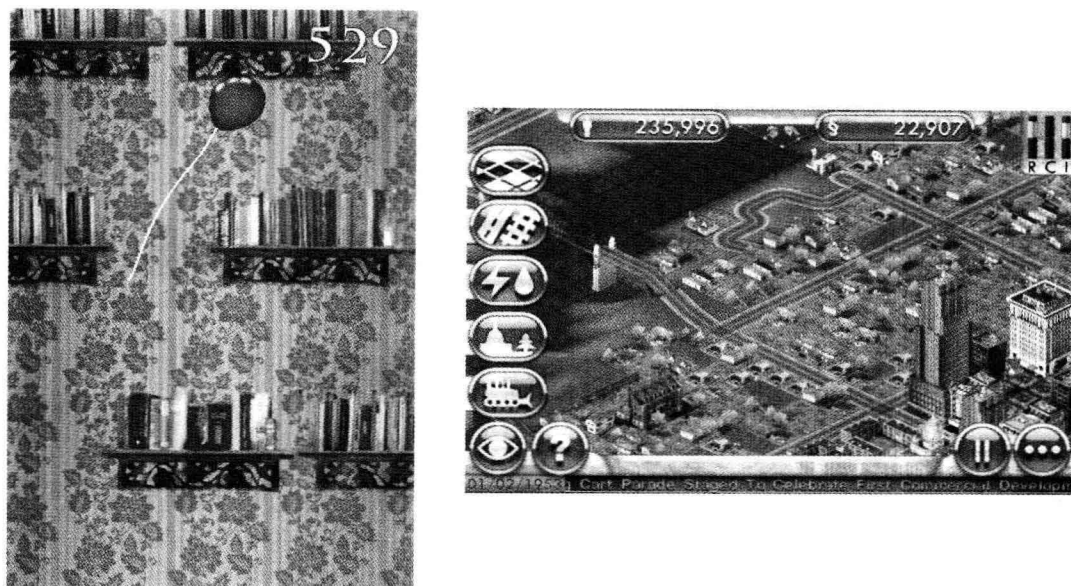


图 1-1 Up There 为玩家提供了一款快速的放气球游戏。模拟游戏的爱好者们会很容易地连续消耗大量时间玩 SimCity

1.2 iPhone 的广泛吸引力——每时每刻都在诞生新的玩家

游戏并不是 iPhone 吸引用户的唯一原因。有的人主要是对 iPhone 的 Web 浏览功能和多媒体功能感兴趣。但是即使是那些从来没有玩过视频游戏的用户也会觉得 App Store 及其成千上万款的游戏非常有吸引力。

App Store 的易用性结合了这么多游戏的可用性，很有可能将任何人转变为游戏玩家，不管是休闲玩家还是其他玩家。开发人员可以创建适合所有类型的人的游戏——从在车上玩游戏的小孩，到远离他的 Xbox 的 Halo 迷，到在椅子上休闲的老爷爷。iPhone 可以让从来不会认为重要到需要专门购买一台设备来玩游戏的人们也能接触到您的游戏。

iPhone 应用程序的多样性实际上在模糊游戏的定义。一些类似于 Snappy touch 开发的 Flower Garden、The Blimp Pilots 开发的 Koi Pond (如图 1-2 所示)，以及 Bolt Creative 开发的 Pocket God 娱乐应用程序都非常流行。虽然这些交互式体验可能并不是字面定义的游戏，但是它们和游戏有很多共性，也能够吸引大量的爱好者。

由于应用程序的类型是由大量开发人员决定的，而不是由发布者决定的，因此创新和实验非常泛滥。由于存在如此多的潜在客户和不同类型的玩家，因此几乎任何类型的游戏都会有市场——不管是经典类型的游戏还是人们从来没有见过的游戏。

作为一个 iPhone 游戏开发人员，您的潜在客户群正在日益增长，而且没有减缓的趋势。由于支持标准化的产品型号，即使是客户升级了 iPhone 或 iPod touch，它们仍然可以继续玩您的游戏——而这在传统的移动电话游戏中几乎是不可能的。您的游戏可以面向所有 iPhone 或 iPod touch 玩家，不用考虑它们拥有什么模型。即使是最大最成功的游戏，也不可能浸透整个市场，因为每时每刻都在诞生新的 iPhone 玩家。

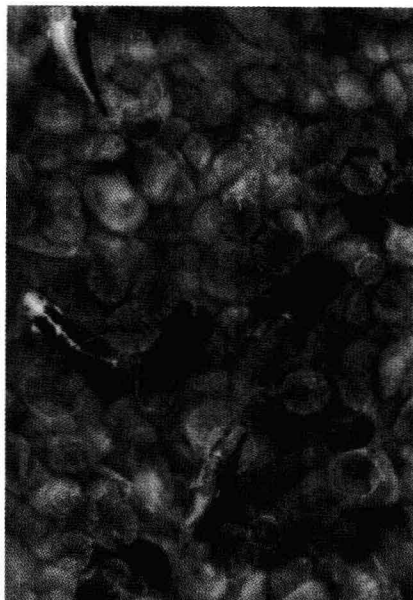


图 1-2 Flower Garden 是一款用户可以管理一个虚拟花园并向朋友送花的应用。

Koi Pond 向用户展示了一个交互式的虚拟金鱼池

1.3 iPhone 的用户界面——手柄终结者

iPhone 的用户界面是这个革命性平台的另一部分。和 Nintendo Wii 将电视游戏(console game)带给公众一样的方式，iPhone 通过触摸屏、加速度传感器、摄像头和麦克风让游戏开发人员可以为交互式体验带来直观自然的界面。

通过控制倾斜、触摸和麦克风控件，开发人员可以使控件对游戏透明，创造出逼真和丰富的用户体验。通过手指直接操作游戏对象或通过移动手中的物理设备来操作游戏对象，用户可以得到符合感官的游戏界面。用户再也不需要花时间来学习怎样将游戏按键、游戏板或游戏手柄对应到游戏动作了。非常明显 iPhone 的界面完全可用。

开发人员正在以全新且未知的方式利用这些非传统的控制方法。iPhone 独特的用户界面已经派生了一些全新类别的游戏。改变设备倾斜的角度可以控制倾斜类的游戏，例如 Lima Sky 开发的 Doodle Jump 和 NimbleBit 开发的 Scoops(如图 1-3 所示)。多点触摸的游戏(例如 Igloo Games 开发的 Bed Bugs)可以使得用户专注于游戏，因为迫使用户同时操作多个游戏对象。Smule 开发的娱乐应用程序 Ocarina 和 Leaf Trombone 允许用户通过向 iPhone 的麦克风吹气来弹唱虚拟乐器。

iPhone 向用户提供了非常吸引人且自然的交互方式，用户甚至看不见界面的存在，因此游戏的界面本身就是游戏。可以展示 iPhone 的透明用户界面功能的一个很好的示例是 NimbleBit 开发的 Hanoi(如图 1-3 所示)。Hanoi 是一款经典的汉诺塔谜题游戏，许多计算机科学的算法课程都会拿汉诺塔谜题作为示例。在其他很多平台上，这个简单的游戏可能会要求用户通过方向