移动开发经典丛书

Learn cocos2d 2 Game Development for iOS

iOS cocos2d 2 游戏开发实战(第3版)

[美] Steffen Itterheim 著
[德] Andreas Löw 著
同济大学苹果俱乐部 译

Apress®

清华大学出版社

移动开发经典丛书

iOS cocos2d 2 游戏开发实战

(第 3 版)

[美] Steffen Itterheim
[德] Andreas Löw
著

同济大学苹果俱乐部 译

清华大学出版社

北 京

Steffen Itterheim，Andreas Löw
Learn cocos2d 2: Game Development for iOS
EISBN：978-1-4302-4416-5

北京市版权局著作权合同登记号 图字：01-2013-1842

图书在版编目(CIP)数据

iOS cocos2d 2 游戏开发实战(第 3 版)/(美)伊特海姆(Itterheim, S.)，(德)勒夫(Löw, A.) 著；同济大学苹果俱乐部 译. —北京：清华大学出版社，2013.5
(移动开发经典丛书)
书名原文：Learn cocos2d 2: Game Development for iOS

ISBN 978-7-302-31892-7

Ⅰ. ①i… Ⅱ. ①伊… ②勒… ③同… Ⅲ. ①移动终端—游戏程序—程序设计 Ⅳ. ①TN929.53 ②TP311.5

中国版本图书馆 CIP 数据核字(2013)第 074841 号

责任编辑：王 军 李维杰
装帧设计：牛静敏
责任校对：成凤进
责任印制：刘海龙

出版发行：清华大学出版社
网 址：http://www.tup.com.cn，http://www.wqbook.com
地 址：北京清华大学学研大厦 A 座 **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 装 者：清华大学印刷厂
经 销：全国新华书店
开 本：185mm×260mm **印 张**：30 **字 数**：730 千字
版 次：2011 年 12 月第 1 版 2013 年 5 月第 3 版 **印 次**：2013 年 5 月第1 次印刷
印 数：1～4000
定 价：59.80 元

产品编号：051097-01

作 者 简 介

Steffen Itterheim 从 20 世纪 90 年代开始就一直热衷于游戏开发。他在 Doom 和 Duke Nukem 3D 社区表现活跃，并因此获得了他的第一份自由职业，成为 3D Realms 的一名 beta 测试人员。作为职业游戏开发者，Steffen 拥有 10 多年的丰富经验，其中大部分时间担任 Electronic Arts Phenomic 的游戏和工具程序员。2009 年 Steffen 第一次接触 cocos2d，那时他与其他人共同创办了一家 iOS 游戏公司——Fun Armada。他乐于将自己的宝贵经验传授给其他游戏开发者，以帮助他们更上一层楼。有机会你可能会在白天看到他在住所附近茂密的葡萄园周围散步，也可能在晚上看到他在 Nevada 沙漠收集瓶盖。

Andreas Löw 在 10 岁的时候有了一台 Commodore C16，从那时起他就对计算机产生了狂热的兴趣。他自学了编写游戏的技术，并在 1994 年发布了自己的第一款游戏 Gamma Zone，这是一款针对 Commodore Amiga 平台的游戏，用纯汇编语言编写完成。在获得电子工程学的学位后，他进入 Harman International 公司，负责为汽车行业开发具有语音识别功能的导航和娱乐系统。他开发了自己的编程语言和开发工具，现在世界上采用语音识别技术的每辆汽车都在使用他的编程语言和开发工具。

iPhone 出现后，他有了回归本行的打算，开始开发一款叫做 TurtleTrigger 的游戏。他意识到 cocos2d 社区存在对好的开发工具的强烈需求。于是，利用自己在游戏和工具开发方面的知识，他开发出了 TexturePacker 和 PhysicsEditor，它们迅速成为 cocos2d 用户进行开发时必不可少的工具。

技术编辑简介

Boon Chew 是 Nanaimo Studio 的执行董事。Nanaimo Studio 位于西雅图和中国上海，是一个专注于互联网和移动游戏的工作室。Boon 拥有丰富的游戏开发和交互型媒体经验，曾就职于 Vivendi Universal、Amazon、Microsoft 以及其他游戏工作室和广告代理商。他热衷于创造，喜欢与出色的人们一起工作。您可以通过 boon@nanaimostudio.com 与 Boon 取得联系。

Tony Hillerson 是一位移动开发人员，也是 Tack Mobile 的共同创始人之一。他毕业于 Ambassador University，获得了管理信息系统学士学位。他涉猎广泛，擅长 Objective-C、Java、Ruby、CoffeeScript、JavaScript、HTML 或 shell 脚本。Tony 曾在 RailsConf、AnDevCon 和 360|Flex 上发表过演讲。他创建了颇受欢迎的 O'Reilly Android 截屏。

在空闲时间，Tony 喜欢弹贝斯和 Warr 吉他，以及制作电子乐。Tony 在科罗拉多州的丹佛市郊居住，陪在他身边的是他的妻子 Lori 以及两个儿子 Titus 和 Lincoln。

致　谢

本书的这一部分使我有一点担心，我不想忘记每一位对本书的编写有指导和帮助的人，但是我知道不可能提到你们中的每一位。如果这里没有提到你，并不代表我不感谢你！给我一支笔，我将马上在书上写下你的名字，并且为我没有提及你抱以诚挚的歉意！

首先我要感谢的是你们，亲爱的读者。没有你们，这本书将没有任何意义。如果没有想到你们可能喜爱本书并且希望从中获得知识，我可能从一开始就不会想到要写这本书。在写这本书的过程中，我博客的读者和其他一些人以见面或邮件的方式向我提出了一些很有价值的建议和要求。感谢你们！感谢所有人！

我要感谢 Jack Nutting，他最先使我有了编写这本书的想法。我很感激他没有掩饰写书所需要的工作量，所以我编写本书并不是毫无准备。

Clay Andres，我不得不感谢这么善良的一个人。他对于书中章节所提出的建议是不可估价的，并且也十分恰当。他还帮助我整理写本书的思路，与他交谈让我感到很愉快。Clay，我希望暴风雨没有冲走你的房子。

很感谢 Kelly Moritz、Corbin Collins 和 Brigid Duffy，他们是本书的流程编辑。当事情变得糟糕而混乱时，他们总是能让每件事情回到正轨。

我收到了 Brian MacDonald、Chris Nelson(他们是本书的开发编辑)和 Boon Chew(本书的技术编辑)很多的回复和建议。他们使我更有信心。Brian 帮助我理解了写作本书的纷繁难懂之处，而 Boon 则指出了很多技术点不明确的地方和需要详细解释的地方。非常感谢两位！Chris 为本书的第 2 版提供了重要的帮助，他指出了许多细微但却很关键的地方，使我能够进行修正。

感谢编辑 Kim Wimpsett。如果没有你的帮助，本书的文字——用我们程序员的说法——将充满各种语法错误和编译警告。

我还想感谢 Bernie Watkins，他负责管理 Alpha Book 的反馈以及我的合同。同样感谢 Chris Guillebeau，他是一位出色的富有灵感的博客作者，并且是我的榜样。

当然我的朋友和家人也都参与了这本书的编写，他们给了我一些反馈意见，并忍受我写作时的过分投入。在此谢谢你们！

译 者 序

随着苹果公司不断地创新与发展，新的 iPhone 5、iPad 4 以及 iPad mini 产品相继问世，包括 iOS 与 Xcode 在内的开发环境和开发工具也都有了更新和进步。相信有不少开发团队正紧锣密鼓地在 iPhone 5 和 iPad mini 上部署自己的应用，一切都是那么令人激动！而随着 iOS 6 的推出，cocos2d 游戏引擎又有了新的发展。在 IT 行业，可谓唯一不变的便是变化。日新月异的技术需要同为开发者的你我保持强盛的好奇心和完美的学习状态，只有这样，才能不断进步，开发出更酷的游戏或应用。相信这本书的再版，一定能使你受益匪浅，帮助你实现梦想。

此次再版，在保留原有内容的基础上，进行了一些全面彻底的检查与更新，目的就是为了适应 iOS 6 和 Xcode 4.4 的开发环境与工具的新发展。cocos2d 2.0 的发布引入了 ARC(自动引用计数)技术，并修改了一些函数重载方式。还有更多的细节，等待着你去探索和尝试。相信你已经跃跃欲试了吧！初次接触 cocos2d 的人也不用担心，本书会通过一些基于 App Store 上热门游戏的原型改编示例，逐步引导你动手实践，向你打开 cocos2d 2.0 精彩世界的大门！

和读者一样，我一直期待着本书的出版。在同济大学苹果俱乐部前辈们的工作基础上，我利用这个寒假完成本书第 3 版的翻译。在这里还要特别感谢同济大学苹果俱乐部主席徐本源、杨元女士以及俱乐部各成员的大力支持和帮助。

由于时间仓促，加之水平有限，书中难免会有错漏之处，希望读者能够提供反馈，我们将不胜感激。

杨　明

2013 年 3 月 10 日

前　言

2009年5月，我第一次接触了Mac OS平台，并且学习了Xcode、Objective-C和cocos2d。即便是对于经验丰富的程序员来说，这也是一个不小的挑战。就在那段时间，我意识到cocos2d真的很棒。但是，相比我当时学习的其他技术，cocos2d的教程、文档和说明文章实在是太匮乏了。

转眼就到了2010年5月。在这一年里，我完成了4个cocos2d项目。我对Objective-C和cocos2d的使用都更加娴熟了。但是，我发现其他很多的程序员还在为一些基本的问题感到困惑，甚至产生误解，这些情景让我想到了一年前痛苦的自己。有关cocos2d的文档依然处于严重缺乏的状态。

今天，有不少使用cocos2d的开发者在博客上发布cocos2d教程、分享他们的使用心得，并因此引起了广泛的关注。大家都在积极地撰写着cocos2d的文档，只可惜你一言我一语，太过分散，读者很难对cocos2d有一个系统的理解。所以，这时候就需要有一个网站来整合这些散落在网络上的宝贵资料。

为此，我创建了一个网站(www.learn-cocos2d.com)来分享我对cocos2d和游戏开发的理解。这个网站上有一些教程，列出了一些常见问题的解答，并提供链接以便对cocos2d感兴趣的读者能够找到所有与cocos2d有关的重要资料。相应的，我也会出售一些与cocos2d相关的产品，希望有一天它能助我达到经济独立的终极目标。我知道这个网站可以令所有人受益。

在网站发布后的24小时内，Jack Nutting就问我是不是考虑写一本有关cocos2d的书。于是，在经历了一系列小故事之后，就有了现在你手上的这本书。

我把我所知道的一切都放在了我的网站上，也写进了这本书里，但这些内容最多也就占全书的1/4。我希望这本书可以以前所未有的详细叙述向大家介绍cocos2d的工作原理和使用方法，如果真是这样，那么我这5个月夜以继日的辛勤劳动就真的值了！

在这本书的撰写过程中，尤其是在本书第2版和第3版更新内容的过程中，我学到了很多东西。我最大的期待就是你能够从本书中学习到所有你想要知道的cocos2d和游戏开发知识!

写作cocos2d图书的过程让我意识到cocos2d还有改进的空间。我强烈感觉到需要一个更好的、让游戏开发初学者更容易上手的cocos2d，因而创建了Kobold2D。本书第16章和www.kobold2d.com上都介绍了Kobold2D。与cocos2d不同的是，Kobold2D有一个安装程序，提供了完整的文档，包含了重要的库，启用了ARC，并且还附带了几十个示例项目。在此基础上，Kobold2D还包含了一些额外的功能。本书第3版的目标是与Kobold2D 2.0完全兼容。

目　录

第 1 章　简介……1
1.1　第 3 版中的新增内容……2
1.2　选择 iOS 版 cocos2d 的理由……3
1.2.1　免费……3
1.2.2　开源……4
1.2.3　Objective-C……4
1.2.4　2D 游戏引擎……4
1.2.5　物理引擎……4
1.2.6　技术难度较低……5
1.2.7　依然需要编程……5
1.2.8　超棒的 cocos2d 社区……5
1.3　为什么要用 Kobold2D 取代 cocos2d-iphone……6
1.4　其他 cocos2d 游戏引擎……6
1.5　本书读者对象……7
1.6　阅读前提……8
1.6.1　编程经验……8
1.6.2　Objective-C……8
1.7　本书内容……9
1.7.1　iOS 游戏开发新手将学会什么……9
1.7.2　iOS 应用程序开发者将学会什么……10
1.7.3　cocos2d 开发者将学会什么…10
1.8　章节介绍……10
1.9　本书的源代码……11
1.10　问题和反馈……12
第 2 章　入门……13
2.1　准备工作……13
2.1.1　系统要求……13
2.1.2　注册成为 iOS 开发者……14
2.1.3　证书与授权文件……14
2.1.4　下载并安装 Xcode 与 iOS SDK……15
2.1.5　下载 cocos2d 或 Kobold2D …16
2.1.6　安装 Kobold2D……16
2.1.7　创建 Kobold2D 项目……16
2.1.8　安装 cocos2d 及其 Xcode 项目模板……18
2.1.9　创建 cocos2d 项目的方式……19
2.1.10　如何在 cocos2d 项目中支持 ARC……21
2.2　cocos2d 和 Kobold2D 应用程序剖析……28
2.3　支持 ARC 的内存管理……34
2.4　改变世界……35
2.5　你还应该知道的……37
2.5.1　iOS 设备……37
2.5.2　关于内存的使用……38
2.5.3　iOS 模拟器……39
2.5.4　关于性能和日志……40
2.6　本章小结……41
第 3 章　基础知识……43
3.1　cocos2d 场景图……43
3.2　CCNode 类的层次结构……46
3.3　CCNode 类……47
3.3.1　节点的处理方式……47
3.3.2　动作的处理方式……48
3.3.3　消息调度……49
3.4　Director 类、场景和层……53
3.4.1　Director 类……53
3.4.2　CCScene 类……53
3.4.3　场景和内存……54

3.4.4 推进和弹出场景……55
3.4.5 CCTransitionScene 类……56
3.4.6 CCLayer 类……58
3.5 CCSprite 类……64
3.6 CCLabelTTF 类……65
3.7 菜单……66
3.8 动作……71
3.8.1 延时动作……72
3.8.2 瞬时动作……77
3.9 方向、单例、测试、API 参考……80
3.10 本章小结……86

第 4 章 你的第一个游戏……87
4.1 创建 DoodleDrop 项目……88
4.2 从一个支持 ARC 的 cocos2d 项目开始……88
4.3 创建 DoodleDrop 场景……90
4.4 添加 Player Sprite……93
4.5 加速计输入……96
4.6 首次测试运行……97
4.7 玩家速度……97
4.8 添加障碍物……100
4.9 碰撞检测……105
4.10 标签和位图字体……107
4.10.1 添加得分标签……107
4.10.2 CCLabelBMFont 简介……108
4.10.3 使用 Glyph Designer 创建位图字体……109
4.11 播放音频……111
4.12 iPad 开发注意事项……113
4.12.1 支持 Retina 高清显示屏的 iPad……113
4.12.2 单个通用的应用程序亦或两款独立的应用程序……113
4.12.3 限定支持设备……114
4.13 本章小结……115

第 5 章 游戏组件……117
5.1 使用多个场景……117
5.1.1 添加多个场景……117
5.1.2 正在加载下一段，请做好准备……120
5.2 使用多个层……122
5.2.1 实现关卡的最佳方法……128
5.2.2 CCLayerColor 和 CCLayerGradient……129
5.3 从 CCSprite 类继承游戏对象……130
5.4 使用 CCSprite 复合游戏对象……131
5.5 奇妙的 CCNode 派生类……135
5.5.1 CCProgressTimer……136
5.5.2 CCParallaxNode……137
5.5.3 CCMotionStreak……139
5.6 本章小结……141

第 6 章 深入了解精灵……143
6.1 Retina 显示屏……143
6.2 CCSpriteBatchNode……146
6.2.1 何时使用 CCSpriteBatchNode……148
6.2.2 Sprites01 示例项目……148
6.3 精灵动画初体验……154
6.4 用于创建动画的辅助类别……155
6.5 使用纹理图册……158
6.5.1 何为纹理图册……158
6.5.2 TexturePacker 工具介绍……158
6.5.3 为 TexturePacker 准备项目……159
6.5.4 使用 TexturePacker 创建纹理图册……160
6.5.5 在 cocos2d 中使用纹理图册……163
6.5.6 改进 CCAnimation 辅助类别……165
6.5.7 将所有图像都放入一个纹理图册中……166
6.6 本章小结……167

第 7 章 滚屏射击游戏(上)……169
7.1 高级视差滚屏……169
7.1.1 将背景创建为条纹……169
7.1.2 在代码中重建背景……171

7.1.3 移动 ParallaxBackground····· 174
7.1.4 视差滚动的速度因素 ····· 175
7.1.5 实现背景的无限滚动 ····· 177
7.1.6 消除闪烁 ····· 180
7.1.7 重复贴图 ····· 181
7.2 虚拟手柄 ····· 182
7.2.1 SneakyInput 简介 ····· 182
7.2.2 触摸按钮产生射击 ····· 185
7.2.3 为按钮添加皮肤 ····· 187
7.2.4 控制动作 ····· 189
7.2.5 数字控制 ····· 192
7.3 本章小结 ····· 193

第 8 章 滚屏射击游戏(下) ····· 195
8.1 添加 BulletCache 类 ····· 195
8.2 添加敌人 ····· 202
8.2.1 Enemy 类 ····· 203
8.2.2 EnemyCache 类 ····· 208
8.3 组件类 ····· 212
8.4 射击开火 ····· 215
8.5 大怪物的生命条 ····· 218
8.6 本章小结 ····· 221

第 9 章 粒子效果 ····· 223
9.1 粒子效果实例 ····· 223
9.2 用复杂方法创建粒子效果 ····· 226
9.2.1 继承 CCParticleSystem ····· 227
9.2.2 CCParticleSystem 属性 ····· 229
9.3 Particle Designer ····· 238
9.3.1 Particle Designer 介绍 ····· 238
9.3.2 使用 Particle Designer 生成的粒子效果 ····· 240
9.3.3 分享粒子效果 ····· 241
9.4 在射击游戏中添加粒子效果 ····· 243
9.5 本章小结 ····· 245

第 10 章 瓦片地图 ····· 247
10.1 瓦片地图简介 ····· 247
10.2 使用 TexturePacker 处理图像 ····· 250
10.3 Tiled(Qt)地图编辑器 ····· 251
10.3.1 创建新的瓦片地图 ····· 251
10.3.2 设计瓦片地图 ····· 254
10.4 在 cocos2d 中使用直角瓦片地图 ····· 256
10.4.1 定位被触摸的瓦片 ····· 260
10.4.2 使用对象层 ····· 263
10.4.3 绘制对象层矩形 ····· 264
10.4.4 滚动瓦片地图 ····· 266
10.5 本章小结 ····· 268

第 11 章 斜角瓦片地图 ····· 269
11.1 设计斜角瓦片地图图形 ····· 270
11.2 使用 Tiled 编辑斜角瓦片地图 ····· 272
11.2.1 新建斜角瓦片地图 ····· 272
11.2.2 创建新的斜角瓦片集 ····· 274
11.2.3 设计斜角瓦片地图的基本规则 ····· 274
11.3 将斜角瓦片地图应用到游戏编程中 ····· 276
11.3.1 在 cocos2d 中加载斜角瓦片地图 ····· 276
11.3.2 在 cocos2d 中设置斜角瓦片地图 ····· 276
11.3.3 定位斜角瓦片 ····· 278
11.3.4 滚动斜角瓦片地图 ····· 281
11.3.5 斜角瓦片地图的边界问题 ····· 282
11.3.6 增加可移动的玩家角色 ····· 285
11.4 在游戏中加入更多内容 ····· 293
11.5 本章小结 ····· 294

第 12 章 物理引擎 ····· 295
12.1 物理引擎的基本概念 ····· 295
12.2 物理引擎的局限性 ····· 296
12.3 Box2D 与 Chipmunk ····· 296
12.4 Box2D ····· 297
12.4.1 Box2D 眼中的世界 ····· 298

12.4.2 把移动范围限制在屏幕内 ····· 302
12.4.3 转换点 ····· 304
12.4.4 在 Box2D 世界中添加盒子 ····· 304
12.4.5 更新 Box2D 世界 ····· 306
12.4.6 碰撞检测 ····· 307
12.4.7 连接刚体 ····· 309
12.5 Chipmunk ····· 310
12.5.1 构建 Chipmunk 物理空间 ····· 311
12.5.2 将盒子添加到物理空间中 ····· 313
12.5.3 添加小盒子 ····· 314
12.5.4 更新 Chipmunk 物理空间 ····· 315
12.5.5 Chipmunk 碰撞实践 ····· 316
12.5.6 Chipmunk 中的关节 ····· 317
12.6 本章小结 ····· 319

第 13 章 弹球游戏 ····· 321
13.1 图形：凸多边形和逆时针方式 ····· 321
13.2 使用 PhysicsEditor ····· 322
13.2.1 定义发射器形状 ····· 324
13.2.2 定义弹球桌形状 ····· 326
13.2.3 定义挡板 ····· 328
13.2.4 定义反弹器和球 ····· 329
13.2.5 保存并发布 ····· 330
13.3 编写弹球游戏 ····· 330
13.3.1 强制纵向显示 ····· 331
13.3.2 BodySprite 类 ····· 331
13.3.3 创建弹球桌 ····· 334
13.3.4 Box2D 调试绘制 ····· 340
13.3.5 添加球 ····· 341
13.3.6 使球动起来 ····· 343
13.3.7 添加反弹器 ····· 346
13.3.8 发射器 ····· 348
13.3.9 挡板 ····· 356
13.4 本章小结 ····· 360

第 14 章 Game Center ····· 361
14.1 激活 Game Center ····· 361
14.1.1 在 iTunes Connect 中创建应用程序 ····· 362
14.1.2 建立排行榜和成就 ····· 362
14.1.3 AppController 和 NavigationController ····· 363
14.1.4 配置 Xcode 项目 ····· 363
14.1.5 小结 ····· 366
14.2 Game Kit 编程 ····· 366
14.2.1 GameKitHelper 委托 ····· 367
14.2.2 检查 Game Center 是否可用 ····· 368
14.2.3 验证本地玩家身份 ····· 369
14.2.4 block 对象 ····· 372
14.2.5 接收本地玩家的好友列表 ····· 374
14.2.6 排行榜 ····· 376
14.2.7 成就 ····· 382
14.2.8 联机 ····· 387
14.2.9 收发数据 ····· 391
14.3 本章小结 ····· 395

第 15 章 cocos2d 与 UIKit 视图 ····· 397
15.1 Cocoa Touch 是什么 ····· 397
15.2 同时使用 Cocoa Touch 和 cocos2d ····· 398
15.2.1 为什么将 Cocoa Touch 和 cocos2d 混合在一起 ··· 398
15.2.2 混合 Cocoa Touch 和 cocos2d 的局限性 ····· 398
15.2.3 Cocoa Touch 和 cocos2d 的区别 ····· 399
15.3 注意：你在 cocos2d 中的第一个 UIKit 视图 ····· 400
15.4 在 cocos2d 应用程序中嵌入 UIKit 视图 ····· 403

15.4.1 在 cocos2d 视图的前面添加视图 …… 403
15.4.2 使用 UIImage 改变 UITextField 的皮肤 …… 405
15.4.3 在 cocos2d 视图的后面添加视图 …… 407
15.4.4 添加利用 Interface Builder 的视图设计 …… 413
15.5 在 Cocoa Touch 应用程序中嵌入 cocos2d 视图 …… 415
15.5.1 用 cocos2d 创建基于视图的应用程序项目 …… 415
15.5.2 设计混合应用程序的用户界面 …… 417
15.5.3 启动 cocos2d 引擎 …… 418
15.5.4 改变场景 …… 421
15.6 本章小结 …… 423

第 16 章 Kobold2D 入门 …… 425
16.1 使用 Kobold2D 的好处 …… 425
16.1.1 准备使用 Kobold2D …… 426
16.1.2 免费使用 Kobold2D …… 426
16.1.3 Kobold2D 升级简单 …… 426
16.1.4 Kobold 包含流行的类库 … 426
16.1.5 Kobold2D 的跨平台性 …… 427
16.2 Kobold2D 的工作空间 …… 428
16.3 Hello-Kobold2D 模板项目 …… 429
16.3.1 HelloWorld 项目文件 …… 429
16.3.2 Kobold2D 如何启动应用程序 …… 431
16.3.3 Hello Kobold2D 场景和层 …… 434
16.3.4 用 iSimulate 运行 Hello World …… 438
16.4 使用 KKInput 编写的针对 Mac 的 DoodleDrop …… 439
16.5 本章小结 …… 441

第 17 章 番外篇 …… 443
17.1 其他学习和工作资源 …… 444
17.1.1 寻求帮助 …… 444
17.1.2 从源码项目中受益 …… 446
17.1.3 Cocos2D Podcast …… 450
17.1.4 工具介绍 …… 450
17.1.5 cocos2d 参考应用程序 …… 451
17.2 游戏行业 …… 453
17.2.1 与出版商合作 …… 454
17.2.2 寻找自由职业者 …… 455
17.2.3 寻找免费的艺术品和音频 …… 455
17.2.4 寻找相关工具 …… 456
17.2.5 营销 …… 456
17.2.6 使用更多技术获得更多收入 …… 459
17.3 本章小结 …… 463

第1章

简　介

可曾想象，有朝一日你会自己写一个计算机游戏然后靠它赚钱？有了 Apple 的 iTunes App Store 及其配套移动设备，如 iPhone、iPod Touch 和 iPad，要实现这个梦想不再是一件难事。当然，这并不表示开发一个游戏有多简单，你仍然需要学习很多游戏开发和编程的知识。不过，既然你选择了阅读本书，我就有理由相信你已经下定决心踏上这条游戏开发之路了。恭喜你选择了一个可能是全世界最有趣的游戏开发引擎——iOS 版 cocos2d。

使用 cocos2d 的开发者可能有很多不同的专业背景，来自不同的领域。有些人(比如我)可能是已经从事游戏开发好几年甚至几十年的专业人士，还有一些人可能是刚开始接触 iOS 平台开发，或者刚刚进入游戏开发这一充满激情的领域。不管你属于哪一类人，我保证，看完这本书一定能够有所收获。

是这样一种信念让 cocos2d 开发人员走到一起：我们都热爱游戏，也热爱设计和编写游戏。本书十分推崇这种信念，并会向读者介绍一些能够帮助简化游戏开发过程的工具。最重要的是，这本书会教大家写一些很有借鉴意义的小游戏，从中你可以学会如何把一些理论知识运用到现实的游戏开发中。

有些书会整页整页地教读者怎样用一些特定的游戏编程 API 来写一个无聊的战机类太空游戏(Asteroid)，我读到这种书的时候总是觉得特别没劲。我觉得一本好书应该向大家介绍游戏编程的理念和开发工具，因为这些东西是永恒的，不会随着 API 或编程喜好的变化而变化。我读编程书籍和游戏开发书籍已经有 20 多年了，我认为最有价值的书是那些高于技术本身的，是能够让我明白为什么这个地方会这样设计、这样编程、这样做有什么好处的书。所以本书不仅会关注游戏代码的含义，更会关注它的工作原理以及在哪些处理上需要根据情况权衡利弊。

我希望你能学着写出一些有价值的、能在 App Store 上热卖并且受玩家欢迎的游戏。我会介绍这本书里的示例游戏背后深藏的思想和技术理念，当然，我也会告诉你在游戏编程中如何使用 cocos2d 和 Objective-C。本书源代码中有大量注释，它们可以帮助你正确地理解代码的含义。

学习别人的源代码并且根据注释去关注一些重要设计对我来说是学习新知识的最好方法(我想它对你来说也会是一个很棒的方法)。你可以对这本书的随书源代码加以修改，进而做出自己的游戏。我非常期待在不久的将来能够玩到你的游戏！完成你的游戏千万别忘了告诉我！

你可以在 Cocos2D Central(www.cocos2d-central.com)上分享你的游戏或者咨询一些问题，也可以通过我的邮箱(steffen@learn-cocos2d.com)联系我。一定要访问本书的配套网站，网址为 http://www.learn-cocos2d.com。从本书的第 1 版开始，我就在用 Kobold2D 项目改进 cocos2d，你可以在阅读本书的过程中使用它。下面这个网址包含关于 Kobold2D 的更多信息：www. kobold2d.com。

1.1 第 3 版中的新增内容

第 3 版是一次全面彻底的检查与更新，旨在与包括 iOS 6 和 Xcode 4.4 在内的开发环境与工具的新发展保持同步。

首先，cocos2d 2.0 已经摆脱限制，获得了新生。所有的源代码和描述都已经适配于 cocos2d 的最新版本和 OpenGL ES 2.0。尽管不能在第一代和第二代设备上部署 cocos2d 2.0 应用，但其实这些设备在 2012 年 5 月也就只占 iOS 设备销售以来总量的 16%。在你阅读本书时，这个比率就会不出意外地降到 10%左右，并且随着新一代设备持续增高的销售量，这个比率还在不断下降。

有很多读者问我类似这样的问题：这本书还可以适用于 Kobold2D 吗？之前我也回答过这个问题："是的，这本书可以，但是有些许地方会有略微不同。"但现在，这本书的第 3 版改变了我的回答："是的，完全可以。"Kobold2D 所有的不同之处我都会在这本书中提出来。而且我还会强调那些你在使用 Kobold2D 时不需要再做的工作，因为"让 cocos2d 用起来更简单方便"正是 Kobold2D 的所有目的。

如果你读过本书的第 2 版，这里有一份在第 2 版中更新并且仍旧适用于第 3 版的内容的扼要重述。一开始 Andreas Löw 作为合作者参与了本书的撰写，他的 TexturePacker 和 PhysicsEditor 工具的操作指导为这本书的改进提供了很有价值的帮助。我们一起彻底检查了大量的图形，通过添加一些代码以及一些新特性，有效地改进了部分章节。还有新添加的两章内容：其中一章是关于如何将 cocos2d 整合到 UIKit 应用中；另外一章是关于(新发布的)Kobold2D 的介绍。

关于 ARC 的一切

下面介绍 ARC 是什么？ARC？没错，自动引用计数(Automatic Reference Counting，ARC)是苹果公司为 Objective-C 应用程序简化内存管理而提出的新技术。从本质上讲，ARC 摒弃了引用计数，也就意味着你不必自己再去记住你或别人的代码保留了多少次对象，也不再需要你自己去准确地释放对象这么多次。自动释放对象让问题更加复杂化。一旦你不保留对象，对象就会被释放，而且每一次释放都能做到 100%的准确，这就会导致内存泄漏或应用程序易于崩溃。

另外，知道 ARC 并不是垃圾回收也是件好事。ARC 的表现具有完全确定性，也就是说，如果你通过相同的过程运行相同的程序，那么 ARC 每次都会准确地做出一样的表现。ARC 并不是运行时组件——是指编译器在需要时自动插入保留、释放和自动释放的声明。ARC 遵守之前 Objective-C 程序员都要遵守的一些简单规定，以免内存泄漏和程序崩溃。你可以想象如果是一个人在做这道工序，那么会有许多错误悄悄潜藏在程序里，但是 ARC 不但可以表现得丝毫不出差错，同时还能优化你的代码。例如，手动做引用计数就要求你有能力多次保留对象。相比之下，ARC 则会知道什么时候多余的保留是不必要的，然后就会删掉它们。

现在，编译器已经接管了保留和释放内存中对象的乏味工作，而这本书中所有的源代码都已经转为使用 ARC。这意味着更少的代码量、更少的潜在致命病毒以及 bug。

Objective-C 程序员关于 ARC 最关心的两大问题是失控和学习新的编程范例。这两个问题都是针对使用 ARC 的争论话题。在此我提供一种分析：把 ARC 想象成汽车里的自动变速器，而手动引用就像是手动换挡。是不是你离开离合器和换挡摇杆就真的失去控制了？确实是，但那些都不是你真正需要控制的。那是不是要学习一些新内容呢？并非如此，你只需要学习少许内容就可以了，事实上你要做的比以前更少了。

使用手动变速器，即使有多年驾驶经验的司机也很容易忘记换挡，那样就可能会损耗引擎。这就像过度释放或保留对象一样。你可能还会偶尔在使用离合器时直接抛锚。这就像在过度释放对象时导致应用程序崩溃。相比之下，使用自动变速器时引擎总是会在最合适的时候换挡，相比于同档次的手动换挡汽车可以显著降低燃油消耗率。内存泄漏和燃油浪费的问题是相似的。

总之，ARC 并没有剥夺你的控制权。你依然可以掌控对于应用程序来说最重要的一切。那些相比于在 ARC 控制下对内存管理控制要求更高的代码将不再出现。失控纯粹是心理作用所致，要不然就可能是误解了 ARC 的工作方式。至于学习方面，确实有些关于 ARC 的东西你应该学一学，相信你能够学会。相比于一名程序员一天要学习的内容来说这算是很少的，而且比 Objective-C 新手学会(更别提掌握)手动引用计数要容易得多。你只需要阅读苹果公司相对简短的 Transitioning to ARC Release Notes 总结即可：http://developer.apple.com/library/ios/#releasenotes/ObjectiveC/RN-TransitioningToARC/Introduction/Introduction.html。也许你还想阅读我撰写的关于自动引用技术所有内容的博客：www.learn-cocos2d.com/2011/11/everything-know-about-arc。

1.2 选择 iOS 版 cocos2d 的理由

游戏开发者在选择游戏引擎时首先会对他们要选择的产品做一些评估。综合很多因素之后，我认为 cocos2d 对许多开发者来说会是一个非常棒的选择。

1.2.1 免费

首先，cocos2d 是免费的。不需要花钱就可以用它来进行开发。你可以随心所欲地开发 iPhone、iPod 和 iPad 应用，无论免费还是收费都可以。甚至还可以用它开发 Mac OS X 应

用。说真的，这是完全没有附加限制的。

1.2.2 开源

cocos2d 的第二个好处就是它是开源的，这就意味着可以自由地学习游戏引擎的源代码，或者在需要时对引擎做些改动。这使得 cocos2d 既可以扩展，又十分灵活。

1.2.3 Objective-C

另外，cocos2d 是用 Objective-C 编写而成的，Objective-C 是苹果公司用于开发 iOS 应用程序的原生编程语言(native programming language)。由于 iOS SDK 也是用 Objective-C 编写而成的，因此对于使用 cocos2d 的开发者来说，要理解苹果公司的官方文档和使用 iOS SDK 提供的 API 并不困难。

其他很多有用的 API，如 Facebook Connect 和 OpenFeint，也是用 Objective-C 编写而成的，所以要集成它们也非常容易。

注意：

相对于 Objective-C 而言，你可能更喜欢其他编程语言，但我还是建议你学习 Objective-C。我原本有很深的 C++和 C#背景，而且 Objective-C 语法乍一看还挺古怪，所以一开始我并不情愿去学这个据说是又陈旧又过时的编程语言。果不其然，有段时间我的日子过得相当挣扎，我必须摒弃已经养成的编程习惯和思维模式才能弄清楚怎样用 Objective-C 来写程序。

但是，千万不要因为困难就放弃学习 Objective-C。你确实需要花点时间去习惯它，但是这样的付出马上就会得到回报(只要教程和文档足够充分)。所以，再多努力都是值得的!

1.2.4 2D 游戏引擎

显然，cocos2d 中的“2d”已经表明它是一个专注于开发 2D 游戏的引擎，这在当今众多的 iOS 游戏引擎中算是少见的。

cocos2d 也可以用于加载并显示 3D 对象。事实上，现在它已经有了一个完整的增件，叫做 cocos3d，这是一个开源项目，用于为 cocos2d 添加 3D 渲染支持。遗憾的是，cocos3d 目前与 cocos2d 2.0 不兼容，因为 cocos3d 仍然在使用 OpenGL ES 1.1。

但是我想说的是，iOS 设备是非常理想的 2D 游戏平台。时至今日，在 iTunes App Store 中发布的新游戏绝大多数仍然是全 2D 的。许多 3D 游戏在玩法上仍然是 2D 的。2D 游戏通常比较容易开发，算法也比较容易理解和实现。很多情况下，它们对硬件的要求比较低，因此你可以创建色彩更鲜明、更细致的图形。

1.2.5 物理引擎

目前有两种集成在 cocos2d 中的物理引擎可供选择：Chipmunk 和 Box2D。这两种物理引擎仅仅在编写它们的语言上有一些细微的差别：Chipmunk 是用 C 语言编写而成的，而