

Broadview®  
www.broadview.com.cn



魂世界  
SOUL GAME STUDIO

我所理解的

# COCOS2D-X

秦春林 著



电子工业出版社  
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY  
<http://www.phei.com.cn>

# 我所理解的 **COCOS2D-X**

秦春林 著



电子工业出版社

Publishing House of Electronics Industry

北京·BEIJING

## 内 容 简 介

本书针对最新的 Cocos2d-x 3.x 版本,介绍了 Cocos2d-x 游戏引擎的基本架构、渲染机制,以及各个子模块的功能和原理,并结合 OpenGL ES 图形渲染管线,深入探讨了游戏开发中涉及的图形学知识,内容包括 Cocos2d-x 的基本架构及渲染机制、纹理的相关知识、OpenGL ES 2.0 渲染管线、计算机字体的绘制、多分辨率支持、事件分发、碰撞与物理引擎、游戏对象模型架构、Genius-x 开源框架、脚本在游戏引擎中的架构等。本书偏重讲解每个知识模块的概念及原理,使读者能够透过现象看到其背后的工作机制,所以,本书并不是围绕 Cocos2d-x 接口的使用组织内容,而是按照通用游戏引擎架构及图形学的知识组织内容。

本书面向中、高级开发者,可作为初级开发者进阶的教程,也可作为高校游戏引擎架构及图形学相关专业的参考教材。

未经许可,不得以任何方式复制或抄袭本书之部分或全部内容。

版权所有,侵权必究。

### 图书在版编目(CIP)数据

我所理解的 Cocos2d-x / 秦春林著. —北京:电子工业出版社, 2014.12

ISBN 978-7-121-24625-8

I. ①我… II. ①秦… III. ①游戏程序—程序设计 IV. ①TP311.5

中国版本图书馆 CIP 数据核字 (2014) 第 244907 号

责任编辑:潘昕

印 刷:三河市鑫金马印装有限公司

装 订:三河市鑫金马印装有限公司

出版发行:电子工业出版社

北京市海淀区万寿路 173 信箱 邮编 100036

开 本:787×980 1/16 印张:28.75 字数:520 千字

版 次:2014 年 12 月第 1 版

印 次:2014 年 12 月第 1 次印刷

印 数:4 000 册 定价:79.00 元

凡所购买电子工业出版社图书有缺损问题,请向购买书店调换。若书店售缺,请与本社发行部联系,联系及邮购电话:(010) 88254888。

质量投诉请发邮件至 [zltz@phei.com.cn](mailto:zltz@phei.com.cn), 盗版侵权举报请发邮件至 [dbqq@phei.com.cn](mailto:dbqq@phei.com.cn)。

服务热线:(010) 88258888。

此书献给

这个世界上最伟大的爸爸、妈妈

你们对我的支持与包容

造就了我自由的思想

这是我今生最有意义的财富

# 前言

我与 Cocos2d-x 结缘于 2011 年，那个时候我所在的公司 OpenXLive 与 Cocos2d-x 团队合作移植 Cocos2d-x 游戏引擎到 WP7 平台，它采用 C#语言基于 XNA 来实现，我是该项目的负责人。然而不久，微软就用支持 C++原生语言的 WP8 替代了 WP7，该项目也逐渐被开发者淡忘。没想到三年后我却会为 Cocos2d-x 写一本书，这三年，我和 Cocos2d-x 都在成长。

开源是我认为软件最富魅力的部分，它给予我们阅读和修改源代码的自由。开源对人类和科技进步的贡献是巨大的，如今，大多数软件系统都有着各种各样开源软件的影子。开源代码对技术人员的成长有着难以估量的价值，例如，没有开源的 Cocos2d-x，我就难以写出本书中的很多内容。

本书定位为了一本进阶的书籍，它着重于讲述 Cocos2d-x 引擎各个功能及组件背后的实现原理。因此，本书并没有严格按照 Cocos2d-x 引擎本身的功能展开描述，而是从这些功能中抽象出一些设计或者架构层面的内容进行讨论。例如本书没有分别讲述精灵、地图、粒子特效等的接口使用，而是从纹理、渲染方式等多个方面来讲述这些元素背后的工作机制；而 OpenGL ES、物理引擎和脚本等内容也几乎是完全可以脱离于 Cocos2d-x 进行学习和理解的。

这就是本书标题的来源，也是本书与同类书籍在内容编排上的最大不同。我希望写一本书，它可以让读者站在一个系统性的高度对游戏引擎的一些架构设计及实现进行理解，从而不但提升和扩充自己的知识结构，更能够在实际开发中去灵活解决各个层面的技术问题。本书同时追求系统性和实践性的平衡，例如对于纹理部分，除了总结纹理相关的所有知识，也通过一些示例来演示它在各个层面的具体使用。

本书共分 18 章。

第 1~4 章介绍了 Cocos2d-x 引擎的基本架构及新的绘制系统。这部分内容包括：Cocos2d-x 的内存管理机制，UI 树的遍历及结构，应用程序的生命周期，游戏循环的各个阶段，以及在 Cocos2d-x 中驱动各个子系统进行逻辑更新的机制和工作原理。这部分也介绍了 Cocos2d-x 3.x 新的数据结构，并详细描述了新的渲染系统。总之，在这一部分，读者可以对 Cocos2d-x 的基本架构有比较系统的了解。

第 5~10 章围绕 OpenGL ES 图形渲染管线进行介绍。这部分从纹理讲起,详细讲述了纹理的存储格式、传输、缩放、压缩及多重纹理等相关知识;第 8 章详细讲述了顶点数组的结构、顶点属性的绑定及传输、着色器程序的编译及链接,以及 Cocos2d-x 新的着色器子系统,并举例在 Cocos2d-x 中使用着色器的流程和怎样使用多重纹理。这部分也对顶点着色器和片段着色器两个阶段在图形渲染管线中的作用进行了详细描述。

第 9、10 章讲述了 OpenGL ES 图形渲染管线的最后两个阶段:帧缓冲和片段操作,并以 Cocos2d-x 中对这两个阶段的应用 RenderTexture 和 ClippingNode 为例进行讲解。这样,读者将对整个渲染管线的每一个阶段都能有所了解,并且能够结合 Cocos2d-x 中的使用去思考每一个阶段的意义和作用,从而对图形渲染管线有更深刻的了解。这也是本书最具特色的部分。

第 10~15 章讲述了 Cocos2d-x 的一些子系统,包括事件分发、多分辨率支持、动画系统及物理引擎整合。其中物理引擎部分也是我比较喜欢的章节,这部分讲述了一些通用的物理引擎的架构及其使用,以及怎样和游戏引擎进行整合。

第 16~18 章探讨应该怎样去设计和管理游戏世界中的对象。第 16 章讲述了常见的对象模型、组件模型、属性模型之间的概念、区别及优缺点;第 17 章则以属性模型为例,讲述了一个游戏对象模型应该怎样设计;第 18 章探讨了时下最流行的脚本相关的内容,但是与仅仅讨论脚本使用不同的是,我们站在一个游戏引擎的高度去讨论脚本的架构,这样读者甚至能设计自己的脚本模型。这一部分内容具有对前端架构设计的高度总结性与实践性,不管是对经验丰富的读者还是初学者,相信都具有一定的启发性。

另外,本书在写作之初并没有构建出整本书的内容结构,在写前面的章节时我甚至不知道最后几章的内容和结构会是怎样的。在开始写每一章的时候,我不会首先去给自己描绘一个目录结构,而是首先把所有相关的内容列出来,如果之中有我不熟悉的内容,或者有 Cocos2d-x 以外的一些知识,我则会首先停下写作去把这些内容整理出来。例如关于 ECS 部分的内容就是基于 Cocos2d-x 社区的一些讨论,我花了很多时间去学习和理解。当最后所有相关的内容整理为一个列表时,我就知道该怎么去写作这部分的内容。因此,在这本书的目录结构中,读者找不到任何同类书籍类似的内容结构,因为这完全是基于个人的理解及知识结构体系去写作的一本书。

所以,对我而言这本书还有一个有意义的目标,我希望透过这样的方式去探讨一种系统性的学习方法。当你开始学习某个知识点时,不要把自己限定在该知识点处,而是首先从该知识点进行适当的扩展,列出该知识点相关的所有内容,再逐个深入学习和研究细节,最后反过来整理该知识点的结构体系。这样不仅能够更深入地学习该知识点,还能够延伸知识的广度,所以本书有大量超出 Cocos2d-x 以外的内容,这些内容又是与 Cocos2d-x 紧密相关的。希望读者在阅读过程中能够感受到这样一种写作风格和方法。

本书是一本进阶类书籍，阅读本书，你需要对 Cocos2d-x 和 C++ 有基本的了解。如果你是一名新手，可以首先阅读同类书籍中相对比较入门的书籍，否则本书的一些章节可能会给你造成一定的困惑。

学习知识最好的方法永远是多读几遍，书中关于 OpenGL ES 和游戏对象模型部分的内容可能需要多读几次才能更加深入地理解相关的概念。当然，如果你在阅读过程中遇到什么障碍，随时可以通过电子邮件或者我的博客等多种方式和我进行探讨。

本书可以作为 Cocos2d-x 开发学习的书籍，也可以作为单独的 OpenGL ES 相关的教材，游戏对象模型部分的内容也可以作为设计游戏引擎架构的参考资料。本书也可以作为高校游戏及图形学相关专业的参考教程。

最后，由于个人经验有限、思维有限，书中难免会出现一些错误和疏漏。真诚地希望读者可以将这些错误的地方反馈给出版社，我们会及时地列出勘误，以帮助更多的读者更好地学习。

秦春林

2014 年 7 月 25 日

# 致 谢

人类永远都不是孤立存在的，每个人的学习、工作、经历及思想都受到他人的影响，有些人可能会改变你的一生。我希望向那些曾经帮助和鼓励过我的同事、朋友及领导表示最真心的感谢。

首先，我要感谢本书的策划编辑符隆美，没有她的信任、支持与鼓励，我就不可能完成此书。今年春节期间留京写书，她还邀请我与她们家人一起度过除夕，让我感受到团圆的气氛。

我要感谢毕业后在第一家公司 K2 遇到了 Ruddy 和 Michael，他们始终把技术当作自己一生的工作和爱好，并展现出技术在经验积累下的魅力，这对我以后的发展影响是很大的。当然还有 ChoonNgee、Kevin、Evan、Bob、Karon、Cathy、Ruby、Kaiboo 等，我一直都很想念他们。

在北京的这几年，我的好友杨亨冉对我帮助是很大的，他给我介绍了很多工作和学习的机会，同时也在我迷茫的时候给予了我很多鼓励和教诲。

因为前公司 OpenXLive 的老板马宁对我的信任及支持，我才能够有勇气进入游戏行业，并且他还在一直鼓励着我向更远的方向前行。

我要特别感谢现在的公司北京魂世界的 CEO 刘哲，他从一开始就非常支持这本书的写作。虽然他不懂技术，可是还经常很有兴致地和我讨论一些前端架构的细节（关于这点我总是感到很惊讶和困惑）。他对于我因为写作此书而对工作产生影响的无限包容，令我非常感动，这也保证了本书能够按时顺利出版。他还同意将我在公司内部开发的前端框架 Genius-x 开源出来作为本书的内容，并供社区交流学习。

还有魂世界前端技术团队，他们对本书的内容提出的建议，保证了本书的质量，同时也展现了他们的实力，他们永远是最棒的，很荣幸能成为他们中的一员。当然还有魂世界其他所有的同事，这一年来，他们都在支持和鼓励着我。

我要感谢 Cocos2d-x 团队一直以来的努力，他们为 Cocos2d-x 3.x 加入了如此丰富及实用的特性，这让我在写作过程中感到非常愉快。在本书的写作过程中，我有幸到厦门和引擎团队一起工作了一个星期，他们对纹理、着色器、内存管理等内容都有非常重要的建议。

本书写作后期在 CocoaChina 论坛发起讨论，有很多朋友给予了很多的建议和支持，

Net Fly, Hanrea, 泽宇, 海浪, 高雷, 王泽, 大尾巴狼, 臭脚'维, 若尘, 矢乐镇, Balzac 等, 没有写到的朋友请拿到书后把自己的名字填在前面的空格中。

还要特别感谢我的同事兼好友朱启星, 他为本书设计了非常精美的封面, 使得本书增色不少, 相信所有读者都会喜欢他的杰作。

最后, 我要感谢两本书: 初中时阅读的培根的《人生论》散文集, 其精练而内容丰富的词句影响了我的写作风格, 并且培根对于知识的论述支撑着我一生的信仰; 《游戏引擎架构》这本书具有广泛而有深度的内容, 是游戏开发者必读的书籍之一, 本书很多章节在内容组织上都受其启发, 并且它帮助我更清晰地理解了很多概念及思路。书籍对人类的影响是巨大的, 我非常期望本书也能给读者带来一定的帮助。

# 推荐序

2014 年，对于 Cocos2d-x 可以说是具有里程碑意义的一年：3.x 大版本更新、向 3D 领域正式进军、北大/浙大等重点高校开课、与 ARM/微软/Intel 巨头深入合作、在日本等市场实现后进者逆袭……引擎团队的不懈努力得到了丰厚的回报。

Cocos2d-x 3.0 版本的这一次飞跃主要实现了三大优化：一是新的渲染器，可创建渲染队列，可简单地实现批量渲染，跨平台也非常容易；二是新修改过的 API，将语言用法改为 C++ 程序员更习惯的使用方式，稳定性与可读性得到提升；三是性能再次提升优化，利用自动裁剪、自动批处理技术，性能得到很大提升，兼容性方面在中国安卓平台上也做到非常好用。

在版本跟进方面，秦春林的速度是最快的。现在你看到的这本《我所理解的 Cocos2d-x》是国内第一本直接从 Cocos2d-x 3.0 版本入手的书籍，直接从更高的起点起步，可以说是国内最新。

本书由浅入深，共分 18 个章节，从 Cocos2d-x 3.0 及架构开始，逐块剖析引擎的各个特性，同时搭配 OpenGL ES 的相关使用。在最后一章，秦春林还介绍了他自己基于组件架构模型写的开源框架，可以说是一本很好的 Cocos2d-x 高阶书籍，值得一读。

另外，秦春林的团队“魂世界”采用 Cocos2d-x 开发的作品《天降》在今年也取得了非常好的成绩，已被乐逗游戏以 2000 万元的高价代理，是一款很酷的电竞手游。我本人非常期待，希望这款 Cocos2d-x 游戏能够取得更好的成绩。

最后，祝秦春林的书本大卖、游戏火爆，也祝更多的团队能够从 Cocos 引擎中受益。

Cocos2d-x 游戏引擎作者

王哲

# 目 录

|                               |    |
|-------------------------------|----|
| 第 1 章 全新的 Cocos2d-x 3.0 ..... | 1  |
| 1.1 Cocos2d-x 3.0 的历史意义 ..... | 1  |
| 1.1.1 回归 C++ 风格 .....         | 2  |
| 1.1.2 更灵活的渲染架构 .....          | 2  |
| 1.1.3 更自由的发展 .....            | 2  |
| 1.2 Cocos2d-x 3.0 的新特性 .....  | 3  |
| 1.2.1 使用 C++ 风格 .....         | 3  |
| 1.2.2 跨平台的 Label .....        | 6  |
| 1.2.3 新的渲染系统 .....            | 6  |
| 1.2.4 统一的消息分发 .....           | 8  |
| 1.2.5 物理引擎集成 .....            | 9  |
| 1.2.6 新的数据结构 .....            | 10 |
| 1.2.7 其他 .....                | 14 |
| 1.3 Cocos2d-x 引擎展望 .....      | 15 |
| 1.3.1 3D .....                | 16 |
| 1.3.2 Cocos Code IDE .....    | 16 |
| 1.4 本章小结 .....                | 17 |
| 第 2 章 Cocos2d-x 架构一瞥 .....    | 19 |
| 2.1 Cocos2d-x 引擎系统总览 .....    | 19 |
| 2.2 Cocos2d-x 内存管理机制 .....    | 21 |
| 2.2.1 C++ 显式堆内存管理 .....       | 21 |
| 2.2.2 C++ 11 中的智能指针 .....     | 22 |
| 2.2.3 为什么不使用智能指针 .....        | 23 |
| 2.2.4 垃圾回收机制 .....            | 23 |
| 2.2.5 Cocos2d-x 内存管理机制 .....  | 24 |
| 2.2.6 Cocos2d-x 中的智能指针 .....  | 29 |

|       |                         |    |
|-------|-------------------------|----|
| 2.2.7 | 怎样进行内存管理.....           | 35 |
| 2.3   | UI 树及运行时游戏对象.....       | 35 |
| 2.3.1 | 位置与变换.....              | 35 |
| 2.3.2 | 坐标系.....                | 36 |
| 2.3.3 | UI 树.....               | 39 |
| 2.3.4 | UI 元素与内存管理.....         | 45 |
| 2.4   | 应用程序架构.....             | 46 |
| 2.4.1 | 游戏生命周期.....             | 46 |
| 2.4.2 | 窗口尺寸.....               | 48 |
| 2.4.3 | 场景管理.....               | 48 |
| 2.4.4 | 游戏循环.....               | 49 |
| 2.5   | 实时更新游戏对象.....           | 52 |
| 2.5.1 | 帧率.....                 | 52 |
| 2.5.2 | Scheduler.....          | 53 |
| 2.5.3 | 时间线.....                | 54 |
| 2.5.4 | 逻辑更新优先级.....            | 55 |
| 2.5.5 | 性能问题.....               | 56 |
| 2.6   | Cocos2d-x 的主线程.....     | 56 |
| 2.6.1 | 在主线程中执行异步处理.....        | 57 |
| 2.6.2 | 纹理的异步加载.....            | 58 |
| 2.6.3 | 异步处理的单元测试.....          | 60 |
| 2.7   | 本章小结.....               | 60 |
| 第 3 章 | OpenGL ES 2.0 概览.....   | 62 |
| 3.1   | 图形处理器简介.....            | 62 |
| 3.2   | 什么是 OpenGL ES.....      | 63 |
| 3.3   | OpenGL ES 2.0 渲染管线..... | 64 |
| 3.3.1 | 顶点数组.....               | 65 |
| 3.3.2 | 顶点着色器.....              | 66 |
| 3.3.3 | 图元装配.....               | 66 |
| 3.3.4 | 光栅化.....                | 69 |
| 3.3.5 | 片段着色器.....              | 70 |

|                                |           |
|--------------------------------|-----------|
| 3.3.6 片段测试.....                | 70        |
| 3.4 渲染管线中的并行计算.....            | 71        |
| 3.5 构建高性能的渲染引擎.....            | 72        |
| 3.6 帧缓冲.....                   | 73        |
| 3.7 本章小结.....                  | 74        |
| <b>第4章 全新的绘制系统.....</b>        | <b>76</b> |
| 4.1 新绘制系统的特点.....              | 76        |
| 4.2 绘制系统概览.....                | 77        |
| 4.3 RenderCommand.....         | 79        |
| 4.4 RenderQueue.....           | 80        |
| 4.5 GroupCommand.....          | 81        |
| 4.6 Render.....                | 84        |
| 4.6.1 RenderCommand 的排序.....   | 84        |
| 4.6.2 QuadCommand.....         | 85        |
| 4.7 元素的可见性.....                | 87        |
| 4.8 绘制的时机.....                 | 89        |
| 4.9 示例: 自定义 RenderCommand..... | 91        |
| 4.10 本章小结.....                 | 95        |
| <b>第5章 纹理.....</b>             | <b>96</b> |
| 5.1 光栅化.....                   | 96        |
| 5.1.1 多重采样.....                | 97        |
| 5.1.2 纹理坐标.....                | 98        |
| 5.2 像素矩形.....                  | 99        |
| 5.2.1 像素存储模式.....              | 99        |
| 5.2.2 纹理数据的传输.....             | 99        |
| 5.2.3 解包.....                  | 100       |
| 5.3 客户端图像格式.....               | 103       |
| 5.3.1 纹理格式的对应关系.....           | 104       |
| 5.3.2 图像数据格式转换.....            | 105       |
| 5.4 纹理对象和加载纹理.....             | 107       |
| 5.5 纹理单元与多重纹理.....             | 109       |

|        |                      |     |
|--------|----------------------|-----|
| 5.6    | 纹理缩放                 | 110 |
| 5.6.1  | 纹理缩小                 | 110 |
| 5.6.2  | 纹理放大                 | 111 |
| 5.6.3  | 在 Cocos2d-x 中设置过滤模式  | 112 |
| 5.7    | 多级纹理                 | 113 |
| 5.7.1  | 多级纹理过滤模式             | 113 |
| 5.7.2  | 多级纹理的上传              | 114 |
| 5.7.3  | 多级纹理的生成              | 115 |
| 5.8    | 纹理压缩                 | 116 |
| 5.8.1  | 压缩纹理的特点              | 116 |
| 5.8.2  | 压缩纹理的实现              | 117 |
| 5.8.3  | 在 Cocos2d-x 中使用压缩纹理  | 118 |
| 5.8.4  | PVRTC 和 PVRTC2       | 120 |
| 5.8.5  | ETC                  | 122 |
| 5.8.6  | 针对不同设备使用不同的压缩纹理      | 123 |
| 5.9    | 纹理缓存管理               | 123 |
| 5.9.1  | 纹理的生命周期              | 124 |
| 5.9.2  | 用 TextureCache 来管理纹理 | 125 |
| 5.9.3  | 场景过渡中的资源管理           | 127 |
| 5.9.4  | Android 下的纹理恢复处理     | 130 |
| 5.10   | 纹理所占内存的计算            | 131 |
| 5.11   | 使用纹理最佳实践             | 133 |
| 5.11.1 | 硬件层面                 | 133 |
| 5.11.2 | 程序层面                 | 133 |
| 5.11.3 | 资源层面                 | 134 |
| 5.12   | 本章小结                 | 135 |
| 第 6 章  | 精灵                   | 137 |
| 6.1    | 用 Sprite 绘制一个矩形区域    | 137 |
| 6.1.1  | V3F_C4B_T2F_Quad 结构体 | 138 |
| 6.1.2  | 使用 QuadCommand 进行绘制  | 139 |
| 6.1.3  | 将 Sprite 作为子元素       | 140 |

|                                    |            |
|------------------------------------|------------|
| 6.2 Sprite 的绘制属性 .....             | 140        |
| 6.2.1 颜色混合 .....                   | 141        |
| 6.2.2 颜色叠加 .....                   | 144        |
| 6.3 Alpha 预乘 .....                 | 148        |
| 6.4 精灵表 .....                      | 150        |
| 6.5 精灵动画 .....                     | 152        |
| 6.6 批绘制还是自动批绘制 .....               | 154        |
| 6.6.1 SpriteBatchNode .....        | 154        |
| 6.6.2 TextureAtlas .....           | 156        |
| 6.6.3 SpriteBatchNode 的特点和限制 ..... | 157        |
| 6.7 部分拉伸：九宫格 .....                 | 157        |
| 6.8 本章小结 .....                     | 158        |
| <b>第 7 章 OpenGL ES 着色语言 .....</b>  | <b>161</b> |
| 7.1 概览 .....                       | 161        |
| 7.2 基础类型 .....                     | 162        |
| 7.2.1 空类型 .....                    | 163        |
| 7.2.2 布尔类型 .....                   | 163        |
| 7.2.3 整型 .....                     | 163        |
| 7.2.4 浮点型 .....                    | 163        |
| 7.2.5 矢量 .....                     | 164        |
| 7.2.6 矩阵 .....                     | 164        |
| 7.2.7 采样器 .....                    | 164        |
| 7.2.8 结构体 .....                    | 165        |
| 7.2.9 数组 .....                     | 165        |
| 7.3 存储限定符 .....                    | 166        |
| 7.3.1 默认限定符 .....                  | 167        |
| 7.3.2 常量限定符 .....                  | 167        |
| 7.3.3 属性限定符 .....                  | 167        |
| 7.3.4 全局限定符 .....                  | 168        |
| 7.3.5 易变量限定符 .....                 | 169        |
| 7.4 构造器 .....                      | 170        |

|              |                              |            |
|--------------|------------------------------|------------|
| 7.4.1        | 标量的转换构造 .....                | 170        |
| 7.4.2        | 矢量和矩阵构造器 .....               | 171        |
| 7.4.3        | 结构体构造器 .....                 | 172        |
| 7.5          | 矢量的分量 .....                  | 173        |
| 7.6          | 矩阵的分量 .....                  | 174        |
| 7.7          | 结构体和成员 .....                 | 174        |
| 7.8          | 矢量和矩阵操作符 .....               | 175        |
| 7.9          | 本章小结 .....                   | 176        |
| <b>第 8 章</b> | <b>OpenGL ES 着色程序 .....</b>  | <b>177</b> |
| 8.1          | 顶点和顶点数组 .....                | 177        |
| 8.1.1        | 图元与顶点 .....                  | 178        |
| 8.1.2        | 顶点属性状态 .....                 | 178        |
| 8.1.3        | 顶点数组 .....                   | 180        |
| 8.2          | 顶点缓冲对象 .....                 | 185        |
| 8.2.1        | 顶点数组缓冲对象 .....               | 187        |
| 8.2.2        | 索引数组缓冲对象 .....               | 188        |
| 8.2.3        | 使用 VAO 缓存顶点数组状态 .....        | 190        |
| 8.3          | 着色器程序 .....                  | 192        |
| 8.3.1        | 着色器程序字符串 .....               | 192        |
| 8.3.2        | 着色器的加载和编译 .....              | 193        |
| 8.3.3        | 着色器程序对象 .....                | 195        |
| 8.4          | Cocos2d-x 着色器子系统 .....       | 196        |
| 8.4.1        | GLProgram 与 Node 之间的关系 ..... | 197        |
| 8.4.2        | GLProgramState .....         | 199        |
| 8.4.3        | 着色器程序的使用过程 .....             | 202        |
| 8.4.4        | GLProgramState 的管理 .....     | 204        |
| 8.4.5        | GLProgramState 的限制 .....     | 205        |
| 8.5          | 顶点着色器 .....                  | 206        |
| 8.5.1        | 输入参数 .....                   | 206        |
| 8.5.2        | 顶点坐标输出参数 .....               | 207        |
| 8.5.3        | 易变量输出参数 .....                | 208        |

|               |                            |            |
|---------------|----------------------------|------------|
| 8.6           | 片段着色器.....                 | 209        |
| 8.6.1         | 输入参数.....                  | 210        |
| 8.6.2         | 纹理采样.....                  | 210        |
| 8.6.3         | 输出参数.....                  | 211        |
| 8.7           | 着色器编辑工具.....               | 211        |
| 8.8           | 示例.....                    | 213        |
| 8.8.1         | 使用 ETC 压缩纹理.....           | 214        |
| 8.8.2         | 动态设置着色器参数.....             | 217        |
| 8.9           | 着色器程序最佳实践.....             | 220        |
| 8.10          | 本章小结.....                  | 220        |
| <b>第 9 章</b>  | <b>帧缓冲 .....</b>           | <b>223</b> |
| 9.1           | GroupCommand.....          | 223        |
| 9.1.1         | 使用 GroupCommand 的流程.....   | 224        |
| 9.1.2         | GroupCommand 的限制.....      | 225        |
| 9.2           | 帧缓冲.....                   | 226        |
| 9.2.1         | 绑定和管理帧缓冲.....              | 227        |
| 9.2.2         | 将图像附加到帧缓冲.....             | 228        |
| 9.2.3         | 渲染缓冲对象.....                | 228        |
| 9.2.4         | 将 Renderbuffer 附加到帧缓冲..... | 230        |
| 9.2.5         | 将 Texture 附加到帧缓冲.....      | 230        |
| 9.2.6         | 帧缓冲完成状态.....               | 231        |
| 9.3           | RenderTexture.....         | 234        |
| 9.3.1         | RenderTexture 的初始化.....    | 235        |
| 9.3.2         | RenderTexture 的绘制.....     | 238        |
| 9.3.3         | ReadPixels.....            | 239        |
| 9.3.4         | RenderTexture 的限制及用途.....  | 240        |
| 9.4           | 本章小结.....                  | 240        |
| <b>第 10 章</b> | <b>片段操作.....</b>           | <b>242</b> |
| 10.1          | 片段操作简述.....                | 242        |
| 10.2          | 逻辑缓冲区.....                 | 243        |
| 10.2.1        | 位平面.....                   | 244        |