**大连理工大学本科毕业设计（论文）**

**一款酷跑手机游戏的设计与实现**

**Design and Implementation of Parkour Mobile Game**

学 院（系）： 软件学院

专 业： 网络工程

学 生 姓 名： 孔程程

学 号： 201092301

指 导 教 师： 刘文杰

评 阅 教 师： 朱 明

完 成 日 期： 2014年6月5日

大连理工大学

Dalian University of Technology

# 摘 要

随着iOS，Android等智能手机操作系统的飞速发展，人们逐步告别Symbian手机，迎来了智能手机的时代。而智能手机除了满足人们打电话，发短信的普通需求外，也为人们的闲暇时光带来了无穷的乐趣，手机游戏在这方面功不可没。由此，手机市场也出现了异常火爆的景象。本文设计的酷跑游戏利用Cocos2D-x游戏引擎设计实现，适用于Android平台，Android平台正是目前发展最快，使用人数最多，最具有市场前景的智能手机操作系统。

本文介绍了Cocos2D-x游戏引擎的基本知识及几个核心类的内容，对酷跑游戏进行了详细的需求分析，并全面的介绍了酷跑游戏的总体设计，包括酷跑游戏的设计原则，层次架构，功能模块图以及酷跑游戏的图片及音频资源。之后通过对酷跑游戏四个模块的详细介绍阐述了该游戏的详细设计与实现部分的内容。最后对该游戏进行系统测试，以保证该游戏的质量。

本文设计的酷跑游戏使用了Cocos2D-x游戏引擎，在VS2010中编写，并用eclipse工具进行打包，因考虑到屏幕适配问题，之后生成apk文件可安装于任意屏幕分辨率的Android手机，是一款便于操作，界面美观，具有趣味性的酷跑游戏。

关键词**：** Cocos2D-x；酷跑游戏；Android

**Design and Implementation of Parkour Mobile Game**

# Abstract

With the rapid development of iOS smartphone operating system and Android smartphone operating system, people gradually farewell Symbian phones, ushered in the era of smart phones. And smart phone in addition to meet people’s demand on the phone, texting, also has brought the infinite pleasure for people's leisure time, mobile game played in this regard. Thus, the mobile phone market also appears the unusually hot. The parkour game in this paper using Cocos2D - x game engine to design and implement, suitable for the Android platform, the Android platform is the intelligent mobile phone operating system of fastest growing, having the largest number of people using and the market prospect.

This paper introduces the basic knowledge of the Cocos2D - x game engine and a few core class content, has carried on the detailed demand analysis of this parkour game, and comprehensive introduction to the overall design of the game, including the design principle of this parkour game, the hierarchical architecture, function module diagram and pictures and audio resources to the game. And then through four modules ,the paper introduce the content of the elaborated design and implementation of the game in detail. Finally ,the paper uses system test to ensure the quality of the game.

The parkour game design in this paper using the Cocos2D - x game engine, programming in VS2010, and use the eclipse tools to pack, because the screen adaptation into consideration, then the generated apk can be installed on any screen resolution of the Android mobile phone, The game is a fascinating game ,which is easy to operate, and has interface aesthetics.

**Key Words：**Cocos2D-x；parkour game；Android

目 录

[摘 要 I](#_Toc389331511)

[Abstract II](#_Toc389331512)

[引 言 1](#_Toc389331513)

[1 项目背景及Cocos2D-x游戏引擎介绍 2](#_Toc389331514)

[1.1 项目背景 2](#_Toc389331515)

[1.2 项目目的 2](#_Toc389331516)

[1.3 Cocos2D与Cocos2D-x手游引擎简介 2](#_Toc389331517)

[1.4 论文结构 10](#_Toc389331518)

[2 酷跑游戏的分析及设计 12](#_Toc389331519)

[2.1 酷跑游戏的可行性分析 12](#_Toc389331520)

[2.1.1 社会可行性分析 12](#_Toc389331521)

[2.1.2 经济可行性分析 12](#_Toc389331522)

[2.1.3 技术可行性分析 12](#_Toc389331523)

[2.2 酷跑游戏的需求分析 12](#_Toc389331524)

[3 酷跑游戏的总体设计 19](#_Toc389331525)

[3.1 酷跑游戏的设计原则 19](#_Toc389331526)

[3.2 酷跑游戏的层次架构 19](#_Toc389331527)

[3.3 酷跑游戏的功能模块图 22](#_Toc389331528)

[3.4 酷跑游戏的图片及音频资源 23](#_Toc389331529)

[4 酷跑游戏主要模块的详细设计与实现 26](#_Toc389331530)

[4.1 游戏主菜单模块的详细设计与实现 26](#_Toc389331531)

[4.1.1 游戏主菜单模块的设计图 26](#_Toc389331532)

[4.1.2 游戏主菜单模块代码详细介绍 26](#_Toc389331533)

[4.2 游戏中模块的详细设计与实现 29](#_Toc389331534)

[4.2.1 GameMainScene类 29](#_Toc389331535)

[4.2.2 Role类 30](#_Toc389331536)

[4.3 游戏帮助模块的详细设计与实现 33](#_Toc389331537)

[4.4 游戏结束模块的详细设计与实现 33](#_Toc389331538)

[4.4.1 游戏结束模块设计图 33](#_Toc389331539)

[4.4.2 游戏结束模块代码详细介绍 34](#_Toc389331540)

[4.5 游戏开发环境及打包环境的搭建 35](#_Toc389331541)

[4.5.1 游戏开发环境的搭建 35](#_Toc389331542)

[4.5.2 打包环境的搭建 35](#_Toc389331543)

[5 系统测试 36](#_Toc389331544)

[5.1 测试环境 36](#_Toc389331545)

[5.2 测试方法 36](#_Toc389331546)

[5.3 功能测试 36](#_Toc389331547)

[5.3.1 游戏主菜单界面测试 37](#_Toc389331548)

[5.3.2 游戏帮助界面测试 38](#_Toc389331549)

[5.3.3 游戏中界面测试 38](#_Toc389331550)

[5.3.4 游戏结束界面测试 39](#_Toc389331551)

[5.4 性能测试 40](#_Toc389331552)

[5.5 本章小结 41](#_Toc389331553)

[结 论 42](#_Toc389331554)

[参 考 文 献 43](#_Toc389331555)

[致 谢 44](#_Toc389331556)

# 引 言

随着科技的发展及人们生活水平的提高，人们对手机的要求也越来越高，只能够打电话，发短信的手机已无法满足人们的需求，随着这样的大潮，智能手机进入了人们的视线。Android系统一经问世，凭借其强大的功能和便于操作的特点，吸引了越来越多的人来使用。与iOS系统相比，其最大的优势莫过于免费，开源。人们不需要花钱即可下载适合于自己机型的软件，升级自己的手机系统，这对于Android手机市场的发展带来了极大的推动作用。同时由于这种免费性Android手机的游戏市场也异常火爆，游戏成了每个人手中手机上不可或缺的软件。然而很多游戏公司为了更大的盈利化，常常会在游戏中插播一些广告或者把某些关卡设置成只有花钱购买道具才可以通过。这种种行为导致游戏失去了其原本的意义。

本项目开发的这款酷跑游戏完全弥补了以上不足，不仅毫无广告，也不需要购买道具，是一款界面美观清新，易于操作的酷跑游戏。该酷跑游戏的设计原则如下：

（1） 软件模块层次划分合理，充分考虑其耦合性和可扩展性。

（2） 组件划分合理，组件内部逻辑的划分和封装合理，具有良好的复用性，便于今后复用。

（3） 通信方式设计合理。主要包含各个场景之间的通信，定时器的合理调度。

（4） 界面设计美观简洁易于操作，界面的切换合理。

（5） 应用程序与其他应用程序的交互问题，作为一个手机应用程序，它必须可以监听手机中返回键等按键的动作。

（6） 滚动的背景：因为玩家的主角一直在控制在屏幕范围内，所以让玩家感觉到“移动”的方式就是背景的前后滚动。可以使用缓冲背景并移动的方式来达到滚动的效果。同时，还要处理主角与地图的碰撞。

（7） 主角：由玩家控制的对象，玩家控制它的移动，通过点击屏幕以实现主角的跳跃，连续点击屏幕应可实现主角的连续跳跃。

（8） 处理碰撞对象：包括炸弹和星星等对象。

该酷跑游戏按照界面大致可以分为四个模块，分别为游戏主菜单模块，游戏中模块，游戏帮助模块和游戏结束模块。游戏中模块是整个游戏的主要部分，也是整个游戏最难实现的部分。

程序在经过设计和实现后，对其进行了非常全面的白盒和黑盒测试，并请同学试玩，听取他们的宝贵建议进行改进。由于开发时间较为紧张，还有很多地方做的不够完善，今后会继续对该游戏做完善工作。

# 1 项目背景及Cocos2D-x游戏引擎介绍

## 1.1 项目背景

随着科技的发展，人们迎来了移动互联网时代，传统的大哥大，传呼机已随着时代与我们说再见，在科技的大潮中，智能手机发展为新一代备受欢迎的科技产品。其功能不再只局限于打电话，发短信，形形色色的手机软件比比皆是。而在这些手机软件中，手机游戏更成为许多用户的挚爱。提到游戏不得不说游戏引擎，Cocos2D-x游戏引擎在众多游戏引擎中脱颖而出，本文的酷跑类手机游戏即是利用该引擎而设计的。

“神庙逃亡”，“地铁跑酷”，“天天酷跑”等酷跑类游戏更是在手机游戏中备受欢迎，但是当前很多游戏公司为了最大化盈利，会在游戏中夹杂一些广告或者让玩家购买道具，玩家可能在不经意之间就被扣除了话费，同时也有很多游戏因为操作比较复杂，不得不使很多玩家选择放弃，使游戏丧失了最本质的目的。针对以上问题我设计并实现了本文所描述的酷跑类游戏，其拥有清新，卡通的风格，并且操作简单，让人们真正体会到手机游戏带来的乐趣。

1.2 项目目的

无论是在地铁上，公交上还是在马路上，随处可见目前被称为“低头族”的一群人，手机成为了人们日常生活中的必需品。而手机除了最原始的打电话发短信功能外，越来越多的功能被人们所需求，手机游戏应运而生。本项目的目的是设计一款酷跑类的手机游戏，以便人们在无聊等公交或者工作闲暇之余放松紧绷的神经，为繁忙的生活带来更多的乐趣。

本项目是一款利用Cocos2D-x游戏引擎开发的酷跑类手机游戏，活泼生动的界面，紧张刺激的游戏过程，相信会使手机显得更生动灵活化，使玩家为之兴奋，也使设计者更加熟练使用Cocos2D-x引擎，更加了解目前手游市场的发展趋势。

1.3 Cocos2D与Cocos2D-x手游引擎简介

还记得曾经风靡一时的“捕鱼达人”吗？它可以说是利用Cocos2D系列引擎所开发出的游戏的典型代表，正是因为App Store中那些火爆的游戏才使我们对于这个游戏引擎越来越熟悉。Cocos2D游戏引擎最早是用Python语言所开发的，后来使用Object-C被移植为iOS平台的Cocos2D-iPhone版本。下面我们就来了解一下Cocos2D家族的各个成员以及成员之间的联系。



图1.1 Cocos家族成员说明

以下为Cocos2D一些主要版本的介绍：

（1） ShinyCocos：Cocos2D-iPhone绑定Ruby的实现。

（2） Cocos2D-android：Cocos2D的Java实现，且能够在Android操作系统上运行。

（3） Cocos2D-android1：此分支是由一位国内资深开发者建立发展的。这是另一个关于Android操作系统的Cocos2D引擎实现。

（4） Cocos2D-x：Cocos2D的C++语言移植版，支持 WOPhone, iOS , Android 2.1 及更高版本, WindowsXP & Windows7，WindowsPhone 8。

（5） Cocos2D-html5：由Cocos2D-x团队开发的分支，支持HTML5 Canvas技术，获得Google等公司的资助。

（6） Cocos2D-XNA：由Cocos2D-x团队开发的分支，支持WP（Windows Phone）。

（7） Cocos2D-javascript：Cocos2D的JavaScript语言实现。由于HTML5技术的发展，相信Cocos2D-javascript会有更广阔的前景。

（8） CocosNet：Cocos2D的.Net实现，运行在Mono上。

Cocos2D是一个开源框架，主要用于开发二维游戏、演示程序和其他图形界面交互应用等。最初是由Ricardo和他的一些朋友们用Python语言开发。该引擎的名字最早源自开发地Los Cocos。

Cocos2D是一款非常易于使用，且灵活高效的游戏引擎，游戏开发者可以把关注焦点放在游戏设置本身，而不必消耗大量时间学习晦涩难懂的OpenGL ES。从不断壮大的Cocos2D家族成员图中我们不难看出它还是一个方便扩展，易于集成第三方库的引擎。但是我想这还不是很多草根爱好者喜爱它的原因，基于MIT协议的免费开源框架才是它最吸引人之处，用户可以放心使用，而不用再去担心那扰人的商业授权问题。同时强大的社区支持也为许多初学者带来了便利，开发者在开发过程中如果有问题会得到很快的解决。而且如今互联网非常发达，我们可以轻松的从网上查到一些开发者们分享的经验和知识。

由之前介绍的Cocos2D家族成员图我们可知Cocos2D-x是其家族成员中的一个重要分支。Cocos2D-x的跨平台特性使得开发者只需编写一套代码就可以横跨两大平台，即Android和iOS，这大大提高了开发者的开发效率。不仅如此，它还保留了Cocos2D的全部特性，使之前学习过Cocos2D的开发者不至于从头学起，而我们这篇文章所要介绍的酷跑游戏就是使用Cocos2D-x手游引擎所编写的。

随着智能手机的出现，移动手机市场出现了三足鼎立的现象，iPhone，Android手机，Windows Phone彼此在不断竞争着市场，智能平台的应用使得手机游戏异常火爆。但是另一方面，如今智能手机平台的多样性也给开发者带来了许多不便之处。如何将一款手机游戏稍作改动就可以发布到多个平台也是开发者们一直想要解决的的问题，而Cocos2D-x便成了解决这一问题的产物。其中“x”包含两个意思：一方面是它代表了C++的文件扩展CXX，“X”表示这个项目是由C++语言编写的；另一方面意味着交叉，正符合开源项目让用户创建跨平台代码的目标。

Cocos2D-x继承了Cocos2D的全部特点，具体内容如下：

（1） 流程控制：不同场景之间的流程控制易于管理。

（2） 精灵：快速而方便的精灵。

（3） 动作：可组合精灵移动、旋转和缩放等动作，使精灵不再静止。

（4） 特效：包括很多特性，例如波浪、旋转和透镜等。

（5） 平面地图：支持平面地图和45度角地图。

（6） 转换：从一个场景移动到另外一个不同的场景。

（7） 菜单：创建主菜单和游戏菜单等内部菜单。

（8） 文本渲染：支持文本渲染标签。

（9） 文档：包含有简单的例子可以用来测试，例如包括编程指南、API参考、视频教学等。

（10） BSD许可：BSD（Berkly Software Distribution，伯克利软件套件）开源协议让使用者拥有很大的自由。

（11） 基于OpenGL：支持硬件加速。

下面简要介绍一下Cocos2D-x中的几个核心类。

Cocos2D-x是一款手游引擎，它的设计思路当然与游戏密不可分，游戏的各个部分可以抽象成几个概念，包括导演、场景、布景层和人物精灵，各部分之间的关系如下图所示。



图1.2 Cocos2D-x基本部分间的关系图

几乎每一款游戏都是由这些概念组合而成，下面详细解释一下各个概念的具体含义。

（1） 导演（CCDirector）：对于电影导演这一角色每个人都不会感觉陌生，Cocos2D-x中的导演类与电影导演有着相似之处，它是游戏中的组织者和领导者，是整个游戏的负责人、总指挥。游戏的运行规则是由导演类所制定，游戏内的场景、布景类和精灵类能够有序地进行也都是导演类的功劳。

（2） 场景（CCScene）：场景用通俗的话解释就是游戏中的关卡，游戏界面。整个游戏就是由许许多多的场景所构成的。

（3） 布景层（CCLayer）：布景层是场景的组成部分。通常在游戏中一个场景可以根据游戏内容的需要包含多个不同类型的布景层。

（4） 人物精灵（CCSprite）：人物精灵可以通俗的理解为游戏中由玩家控制的主角它们是一个游戏中的关键要素。

下面介绍一下Cocos2D-x中的几个核心类。

（1） 节点类

节点类（CCNode）是Cocos2D-x中的主要类，它继承自CCObject，CCObject类为Cocos2D-x中所有对象的基类，它的地位相当于java中的Object类。它封装了对象的引用计数和自动释放功能。CCNode类的继承关系如图所示。



图1.3 CCNode类的继承关系

我们最常使用的节点类包括场景类（CCScene）、布景层类（CCLayer）、人物精灵类（CCSprite）、菜单类（CCMenu）。

CCNode类包括的主要功能如下：

① 每个节点都可以含有子节点。

② 节点中包含周期性回调函数（Schedule、Unschedule等）。

③ 包含有动作（CCAction）。

有时为了满足自己游戏的需要，会自己定义一个类，这时候让自己新定义的这个类继承自CCNode。一个类如果可以继承自CCNode类，说明它有如下特点：

① 初始化方法和周期性回调方法需重写。

② 在时间线上控制回调。

③ 重写渲染的绘制方法。

CCNode类的主要函数，如表1.1所示。

表1.1 CCNode类的主要函数

|  |  |  |
| --- | --- | --- |
| 函数名 | 返回类型 | 描述 |
| getZOrder | 整型 | 获得兄弟节点间z轴顺序 |
| getChildren | 数组 | 获得子节点数组 |
| getCamera | 摄像机 | 获得节点摄像机 |
| getIsVisible | 布尔型 | 获得是否可见 |
| setAnchorPoint | 空 | 设置锚点 |
| getScale | 浮点型 | 获得缩放系数 |
| setTag | 空 | 设置标签值 |
| onEnter | 空 | 进入节点（场景类）的对象 |
| onExit | 空 | 离开节点（场景类）的对象 |
| addChild | 空 | 添加子节点 |
| runAction | 动作 | 运行动作 |
| getPosition | 点坐标 | 获得坐标位置 |
| stopAllActions | 空 | 停止所有动作 |
| schedule | 空 | 定义一个定时器 |
| unschedule | 空 | 取消一个定时器 |
| visit | 空 | 访问节点 |

在构建游戏时，一般是一个场景中添加一个或者多个布景层，一个布景层中又添加多个精灵或者菜单项。通过这种方式构成一棵渲染树，Cocos2D-x通过遍历这棵树使图像在屏幕上显示。

（2） 导演类

CCDirector类是整个Cocos2D-x游戏引擎的核心，主屏幕创建，控制及显示，都由导演类所控制。一个游戏里一般只存在一个导演。调用CCDirector类的方法控制游戏的开始、结束、暂停。CCDirector类具有如下功能。

① 初始化OpenGL会话。

② 设置OpenGL的一些参数和方式。

③ 对场景的访问和改变以及对Cocos2D-x配置细节的访问。

④ 访问视图。

⑤ 设置投影和朝向。

CCDirector类采用的是单例模式，如下所示为调用CCDirector方法的标准方式：

CCDirector::sharedDirector( ) 函数名

CCDirecor类的继承关系如图。



图1.4 CCDirector类的继承关系

Cocos2D-x采用的是典型的面向对象和分层的设计原则，导演就是整个游戏流程的总指挥，游戏全过程的场景切换均是由它负责。

（3） 场景类

CCScene类继承自CCNode类。相比于CCNode类，它只是添加了一个拥有自己锚点的特性，默认状态下位置在屏幕的正中央。除此之外，到现在为止还没有其它的功能。CCScene类有CCTransitionScene（切换场景类），并且CCTransitionScene类有很多子类，这些类都用于切换场景的特效。切换场景动画大致分为以下步骤：

① 新建场景。

② 根据游戏需要及个人审美观点选择选择CCTransitionScene子类进行切换动画，使用create方法传入之前建的场景，并设置其他参数。

③ 通过调用CCDirector::sharedDirector( ) replaceScene方法将新场景替换掉。

（4） 布景层类

布景层类CCLayer同样继承自CCNode类，其除了实现CCNode类的功能外，增加了触屏事件代理（TouchEventsDelegate）协议，并且可以处理输入，包括触屏和加速度传感器。

CCLayer类的继承关系如图。



图1.5 CCLayer类继承关系

由上图可以看出CCLayer类继承自CCNode类，并且CCLayer类还遵循触屏代理协议、加速度传感器代理协议、键盘事件代理协议。

（5） 精灵类

可以把精灵类CCSprite理解成一张二维的图片对象。它的继承关系如图。



图1.6 CCSprite类继承关系

谈到精灵不得不提到精灵批处理类CCSpriteBatchNode，当我们需要显示两个或两个以上相同的精灵时，如果逐个渲染精灵，每一次渲染都会调用OpenGL的函数，这样帧率就会下降15%左右或者更多，而如果我们使用CCSpriteBatchNode类的精灵，同一张贴图只进行一次准备，一次渲染，一次清理，可以大大提高程序的效率。

1.4 论文结构

本文共分为五章，内容安排如下所示。

第一章为项目背景及Cocos2D-x游戏引擎介绍，介绍本项目的项目背景，Cocos2D-x游戏引擎的主要特点，几个核心类以及论文结构。

第二章为该酷跑游戏的分析及设计，一方面从社会可行性、经济可行性及技术可行性三个方面对酷跑游戏进行可行性分析，另一方面对酷跑游戏进行功能和性能方面的需求分析。

第三章为酷跑游戏的总体设计，首先介绍酷跑游戏的设计原则，然后介绍酷跑游戏的层次架构和功能模块图，最后介绍了酷跑游戏的图片及音频资源。

第四章为酷跑游戏主要模块的详细设计与实现，分别介绍了游戏主菜单模块的详细设计与实现，游戏中模块的详细设计与实现，游戏帮助模块的详细设计与实现，游戏结束模块的详细设计与实现以及游戏开发环境及打包环境的搭建。

第五章为系统测试，分别从功能测试和性能测试两方面对该游戏进行了测试。

# 2 酷跑游戏的分析及设计

对软件需求的合理分析是软件开发周期中非常重要的一个环节，本节将对酷跑游戏做较为全面的可行性分析和需求分析，主要围绕“酷跑游戏需要做什么”这一问题进行分析说明。

2.1 酷跑游戏的可行性分析

2.1.1 社会可行性分析

随着Symbian操作系统的淘汰，移动手机市场成了智能手机的天下。无论是尚在学校的孩子，还是步入社会的成年人，智能手机成了每天生活的一部分。而人们对于它如此喜爱的原因，当然少不了智能手机中种类繁多的游戏，从“神庙逃亡”再到“天天酷跑”，不难看出酷跑类游戏是人们非常钟爱的一种手机游戏类型。

2.1.2 经济可行性分析

利用Cocos2D-x引擎开发的该酷跑类游戏只需要安装有VS2010（或其他版本），Python，Tiled地图编辑器，PhotoShop，eclipse及Android模拟器的电脑即可。

2.1.3 技术可行性分析

设计一款可以实现酷跑游戏基本功能的游戏需要了解Cocos2D-x游戏引擎的基本框架及基本语法知识，因为Cocos2D-x是由Cocos2D-iPhone发展而来，所用语言与Object-C非常类似，这对于曾经有过iOS或者C/C++开发经历的程序员来说学习起来并没有太大难度。同时PhotoShop做为一款如今非常流行的图像处理软件使用起来也并不是特别复杂。

2.2 酷跑游戏的需求分析

首先介绍酷跑游戏的功能需求。

对于一款酷跑类手机游戏来说，它必须要具备与玩家交互的界面，且界面要简洁美观易于玩家操作，给玩家提供良好的用户体验。酷跑游戏的用户界面可分为三部分：游戏的主菜单，游戏中界面，游戏结束界面。

（1） 游戏主菜单

游戏主菜单是用户进入游戏后的主界面。此界面实现的主要功能有：开始新游戏、进入游戏帮助界面、打开关闭游戏背景音乐。游戏主菜单用例图如下所示。



图2.1 游戏主菜单用例图

下面对游戏主菜单进行用例说明。

① 开始新游戏用例说明如表2.1所示。

表2.1 开始新游戏用例说明

|  |  |
| --- | --- |
| 用例信息 | 说明 |
| 用例名 | 开始新游戏 |
| 参与者 | 游戏玩家 |
| 目标 | 玩家点击New Game按钮进入到游戏中界面 |
| 前置条件 | 1.玩家进入游戏 |
|  | 2.玩家点击New Game按钮 |
| 基本事件流 | 1.玩家进入游戏 |
|  | 2.玩家点击New Game按钮  3.游戏开始 |
| 后置条件 | 游戏开始 |

② 关于界面用例说明如表2.2所示。

表2.2 关于界面用例说明

|  |  |
| --- | --- |
| 用例信息 | 说明 |
| 用例名 | 关于界面 |
| 参与者 | 游戏玩家 |
| 目标 | 玩家点击About按钮进入到游戏关于界面 |
| 前置条件 | 1.玩家进入游戏 |
|  | 2.玩家点击About按钮 |
| 基本事件流 | 1.玩家进入游戏 2. 玩家点击About按钮 |
| 后置条件 | 玩家进入到游戏关于界面 |

③ 打开或关闭声音用例说明如表2.3所示。

表2.3 打开或关闭声音用例说明

|  |  |
| --- | --- |
| 用例信息 | 说明 |
| 用例名 | 打开或关闭声音 |
| 参与者 | 游戏玩家 |
| 目标 | 玩家点击声音控制按钮打开或关闭游戏背景音乐 |
| 前置条件 | 1.玩家进入游戏 |
|  | 2.玩家点击声音控制按钮 |
| 基本事件流 | 1.玩家进入游戏 |
| 后置条件 | 2.玩家点击声音控制按钮  3.若声音为打开状态则关闭，若声音为关闭状态则打开  游戏声音打开或者关闭 |

（2）游戏中界面

游戏中界面是整个游戏的核心部分，也是衡量一个游戏制作好坏的关键之处。在该界面处我使用了当前酷跑游戏中非常流行的点击屏幕使主角跳跃的方式，同时保证在人物跳跃时整个画面流畅。在游戏进行的过程中，可以点击屏幕中的暂停游戏按钮，对游戏进行暂停操作，也可以点击返回键返回到游戏主菜单。当主角从陆地上掉落或者触碰到炸弹时，游戏则进入结束画面。游戏中界面用例图如下。



图2.2 游戏中界面用例图

下面对游戏中界面进行用例说明。

① 暂停游戏用例说明如表2.4所示。

表2.4 暂停游戏用例说明

|  |  |
| --- | --- |
| 用例信息 | 说明 |
| 用例名 | 暂停游戏 |
| 参与者 | 游戏玩家 |
| 目标 | 玩家点击暂停游戏按钮则游戏暂停 |
| 前置条件 | 1.玩家开始新游戏 |
|  | 2.玩家点击暂停游戏按钮 |
| 基本事件流 | 1.玩家进入游戏中界面 |
| 后置条件 | 2.玩家点击暂停按钮  3.游戏暂停，除恢复游戏按钮可用外，其余按钮均不可用，且屏幕不可触摸游戏声音打开或者关闭  游戏暂停 |

② 恢复游戏用例说明如表2.5所示。

表2.5 恢复游戏用例说明

|  |  |
| --- | --- |
| 用例信息 | 说明 |
| 用例名 | 恢复游戏 |
| 参与者 | 游戏玩家 |
| 目标 | 玩家点击恢复游戏按钮，游戏继续 |
| 前置条件 | 1.玩家开始新游戏 |
|  | 2.玩家点击暂停游戏按钮  3.玩家点击恢复游戏按钮 |
| 基本事件流 | 1.玩家进入游戏 |
| 后置条件 | 2.玩家点击暂停游戏按钮  3.玩家点击恢复游戏按钮  4.游戏继续  游戏从暂停处继续开始 |

③ 返回主菜单用例说明如表2.6所示。

表2.6 返回主菜单用例说明

|  |  |
| --- | --- |
| 用例信息 | 说明 |
| 用例名 | 返回游戏主菜单 |
| 参与者 | 游戏玩家 |
| 目标 | 玩家点击返回主菜单按钮，返回到游戏主菜单 |
| 前置条件 | 1.玩家开始新游戏 |
|  | 2.玩家点击返回主菜单按钮 |
| 基本事件流 | 1.玩家开始新游戏 |
| 后置条件 | 2.玩家点击返回主菜单按钮  3.返回到游戏主菜单  返回到游戏主菜单 |

（3）游戏结束界面

当主角从陆地掉落或者碰到炸弹后则会切换到游戏结束界面，该界面会显示玩家所得分数，同时有再玩一次，返回主菜单，退出游戏三个按钮，玩家可以任意选择后续操作。游戏结束界面用例图如下所示。



图2.3 游戏结束界面用例图

下面对游戏结束界面进行用例说明。

① 再玩一次用例说明如表2.7所示。

表2.7 再玩一次用例说明

|  |  |
| --- | --- |
| 用例信息 | 说明 |
| 用例名 | 再玩一次 |
| 参与者 | 游戏玩家 |
| 目标 | 玩家点击try again按钮，游戏重新开始 |
| 前置条件 | 1.主角死亡进入游戏结束界面 |
|  | 2.玩家点击try again按钮 |
| 基本事件流 | 1.主角死亡进入游戏结束界面 |
| 后置条件 | 2.玩家点击try again按钮  3.进入游戏中界面，游戏重新开始  游戏重新开始 |

② 返回主菜单用例说明如表2.8所示。

表2.8 返回主菜单用例说明

|  |  |
| --- | --- |
| 用例信息 | 说明 |
| 用例名 | 返回主菜单 |
| 参与者 | 游戏玩家 |
| 目标 | 玩家点击Menu按钮，返回到游戏主菜单 |
| 前置条件 | 1.主角死亡进入游戏结束界面 |
|  | 2.玩家点击Menu按钮 |
| 基本事件流 | 1.主角死亡进入游戏结束界面 |
| 后置条件 | 2.玩家点击Menu按钮  3.返回到游戏主菜单  返回到游戏主菜单 |

下面介绍酷跑游戏的性能需求。

对于所有的手机游戏来说最忌讳的即是游戏对于用户的响应较慢，游戏画面很卡，对于本款酷跑游戏有以下性能需求：

（1） 当点击游戏图标进入游戏时，程序响应时间最长不超过2秒钟。

（2） 当点击New Game按钮时，程序响应时间最长不超过1秒钟。

（3） 当点击游戏暂停按钮时，程序响应时间最长不超过1秒钟。

（4） 当点击恢复游戏按钮时，程序响应时间最长不超过1秒钟。

（5） 当点击 try again按钮时，程序响应时间最长不超过1秒钟。

（6） 当在主菜单页点击Android手机返回键时，程序响应时间最长不超过2秒钟。

（7） 保证玩家在游戏过程中，地图滚动自然，主角跳跃动作流畅，吃到星星及碰到炸弹时音效实时播放。

# 3 酷跑游戏的总体设计

经过需求分析，已经确定软件必须做什么了，总体设计阶段就是决定软件应该怎么做。本章将从三个方面来阐述酷跑游戏的总体设计。首先，根据手机软件和横版酷跑类游戏的特点列出该游戏的设计原则。然后，对酷跑游戏设计架构进行说明并根据功能需求分析来划分酷跑游戏的功能模块。最后，对酷跑游戏的音频及图片资源进行说明，简要介绍一下游戏核心部分地图的制作方法。

3.1 酷跑游戏的设计原则

对于一款酷跑类的手机游戏来说我们不仅要考虑到它是一个手机软件，更有考虑到它还是一个游戏，所以酷跑游戏既要具有一款实用性软件的特点，也要具有一款横版跑酷类游戏的所有特点。根据这两部分要求，酷跑游戏的设计原则如下：

（1） 软件模块层次划分合理，充分考虑其耦合性和可扩展性。

（2） 组件划分合理，组件内部逻辑的划分和封装合理，具有良好的复用性，便于今后复用。

（3） 通信方式设计合理。主要包含各个场景之间的通信，定时器的合理调度。

（4） 界面设计美观简洁易于操作，界面的切换合理。

（5） 应用程序与其他应用程序的交互问题，作为一个手机应用程序，它必须可以监听手机中返回键等按键的动作。

（6） 滚动的背景：因为玩家的主角一直在控制在屏幕范围内，所以让玩家感觉到“移动”的方式就是背景的前后滚动。可以使用缓冲背景并移动的方式来达到滚动的效果。同时，还要处理主角与地图的碰撞。

（7） 主角：由玩家控制的对象，玩家控制它的移动，通过点击屏幕以实现主角的跳跃，连续点击屏幕应可实现主角的连续跳跃。

（8） 处理碰撞对象：包括炸弹和星星等对象。

3.2 酷跑游戏的层次架构

了解一个程序的层次结构会使我们对这个程序有更加清晰的了解和认识，下面将通过层次结构图对酷跑游戏的层次结果进行详细的介绍。酷跑游戏的层次结构图如下图3.1所示。



图3.1 酷跑游戏层次结构图

根据酷跑游戏层次结构图，下面对不同层次中的各个模块进行介绍。

场景层是酷跑游戏的核心部分，但在介绍场景层之前我们需要先了解一下一个Cocos2D-x游戏是怎样跑起来的，以samples文件夹下最简单的例子HelloWorld为例。这个项目由AppDelegate.cpp，AppDelegate.h，HelloWorldScene.cpp，HelloWorldScene.h， main.cpp，main.h，外部依赖项等几个文件组成。像我们所熟知的C/C++等程序一样，main函数依然是程序的入口点。在main函数中，AppDelegate继承于CCApplication，是整个游戏的程序类。CCEGLView封装了使用OpenGL作为显示底层API的一个基本的Windows窗体的创建和控制。可以简单的认为一个控制了程序的逻辑，而另一个负责游戏的显示。整个Cocos2D-x游戏的运行过程可以分为以下几步：

（1） 程序运行进入main函数，在main函数中调用CCEGLView::sharedOpenGLView()，在sharedOpenGLView()中调用create函数来创建窗体。创建一个窗体的步骤如下：

① 填充窗口类结构WNDCLASS

② 注册窗口

③ 创建窗口

④ 显示窗口

（2） 接下来在main函数中调用CCApplication::sharedApplication()->run(),该函数有以下作用：

① 获取当前系统的频率和计数，利用一个LARGE\_INTEGER型变量来存储。

② 调用子类的applicationDidFinishLauching()执行进入程序后的一些初始化工作。在这个函数中会初始化导演这个单例类，设置窗口属性及创建场景和运行场景，经过这个函数以后我们的游戏就跑起来了。

③ 获取CCEGLView单例，显示窗口。

④ 进入循环while(1)，重绘每一帧的场景。

图3.2 Cocos2D-x游戏运行步骤

（1） 场景层：场景层主要由主菜单场景模块，游戏主界面场景模块，游戏帮助场景模块，游戏结束场景模块几部分构成。游戏主界面场景模块包含了游戏中界面的实现部分，例如地图，游戏背景，主角及道具的初始化以及触摸函数的实现。游戏帮助场景模块主要实现给玩家游戏帮助的界面。游戏结束场景模块主要实现当主角死亡后游戏结束画面的实现。

（2） 精灵层：精灵层中主要包括主角精灵模块以及由敌人星星组成的道具模块。除相关精灵的实现外，还包含主角的动画，碰撞检测，碰撞修复，镜头移动，主角运动状态更新定时器函数，镜头部分定时器函数，分数更新定时器函数。是整个程序的一个主要部分。

（3） 分数层：考虑到使用Cocos2D-x中字体显示分数的话无法达到非常美观的效果，在程序中我使用一张包含有0~9数字的图片来显示游戏的分数，在程序中控制每次显示图片中相应数字，这样不仅使得游戏更加美观，也没有影响程序的可扩展性，如果游戏风格改变，可以任意切换图片，而不用修改代码。

3.3 酷跑游戏的功能模块图

对酷跑游戏的功能性需求分析，可按照功能对软件划分出功能模块图，如图3.3.所示。大方框框起来的部分是可与用户交互的界面实现功能。

图3.3 酷跑游戏功能模块图

（1） 游戏主菜单包含的内容如下：

① 开始新游戏按钮：当用户点击此按钮时，游戏以翻页的形式切换到游戏中界面。

② 帮助页面按钮：当用户点击此按钮时，游戏以翻页形式切换到帮助页面。

③ 声音控制按钮：若游戏正在播放背景音乐点击该图标则将游戏切换成静音模式，按钮图标切换为静音图标，若游戏正处于静音状态，点击该按钮切换为有音乐状态，按钮图标切换为有声音状态。

④ 对手机返回键进行监听：当用户点击手机返回键时，以弹窗形式弹出一个对话框，询问玩家是否真的要退出游戏，如果玩家选择“是”，则游戏退出；如果玩家选择“否”，则游戏仍停留在该界面。

（2） 游戏中界面包含的内容如下：

① 屏幕的触摸实现：当点击屏幕任意位置时即可控制主角跳跃来吃星星加分或者躲避炸弹，本游戏可支持连续点击屏幕实现主角连跳。

② 暂停、恢复游戏按钮：当游戏为进行中状态时点击暂停按钮弹出暂停页面游戏暂停，将屏幕触摸事件以及除恢复按钮外的其它按键屏蔽，使其失效。当点击游戏暂停页面的恢复按钮后，游戏继续，失效控件重新有效。

③ 返回游戏主菜单按钮：点击该按钮后切换到游戏主菜单。

（3） 游戏帮助界面内容如下：

① 游戏规则：通过一个制作精美的界面向玩家介绍游戏的玩法。

② 返回主菜单按钮：当玩家理解了游戏的规则后，通过点击此按钮可以返回到游戏主菜单界面。

（4） 游戏结束界面内容如下：

① 显示游戏得分：利用图片资源来显示玩家的得分，整个分数为卡通风格，与游戏的风格相一致。

② 再玩一次按钮：如果玩家对于游戏的失败不服气可以点击该按钮，重新开始一次游戏。

③ 返回游戏主菜单按钮：点击该按钮后即可切换到游戏的主菜单界面。

④ 退出游戏按钮：点击该按钮即可退出游戏。

3.4 酷跑游戏的图片及音频资源

一款成功的游戏不仅仅需要有刺激的游戏情节，良好的用户交互，能吸引人眼球的界面也是重中之重。由于本人不是学美工出身，所以游戏的图片及音频资源有部分是引用自已有的游戏。图片的制作主要是利用PhotoShop进行裁剪或者修复等操作，在此主要想介绍一下酷跑游戏的核心地图的制作。

在最初的设计中地图是想采用随机生成的方式，但后来考虑到这种实现方法的复杂性，便使用了在Cocos2D-x游戏中非常流行的Tiled地图编辑器。

（1） Tiled地图编辑器简要介绍

Cocos2D-x支持Tiled地图编辑器生成的地图数据文件。Tiled地图编辑器的目标是成为普遍使用的地图编辑器，它便于使用并且可以轻松地在不同的引擎中使用。目前最新版本使用Qt框架进行开发，之前也有Java版本，目的就是可以使编辑器跨平台。这点在Cocos2D-x的开发中也比较重要。因为Cocos2D-x的跨平台特性，可能导致用户所使用的操作系统各不相同。为了让大家都可以看到地图的效果，编辑器的“跨平台”也是必要的。

Tiled地图编辑器的特性如下：

① 使用基于XML编码形式的地图数据文件使其可以在不同的游戏引擎中通用。

② 支持普通和45度角两种视角。

③ 对象的放置位置可以精确到像素。

④ 支持撤销、重做和复制、粘贴的操作。

⑤ 支持图素、层次和对象等通用的概念。

⑥ 自动重新载入图素集。

⑦ 可以重置图素的大小和偏移。

⑧ 支持图章刷和填充等高效工具。

⑨ 支持以通用的格式输入输出来打开和存储文件。

（2） 使用Tiled地图编辑器制作地图

下面简单介绍一下如何使用Tiled地图编辑器编辑地图。

① 选择“文件 新地图”，新建地图工程，在弹出的对话框中设置地图的高度和宽度，图块的大小以及地图的方向。

② 选择“地图 新图块”导入图素文件，在弹出的对话框中设置图块的大小、边距和偏移量等。

③ 选择完图块后，右下角的部分就显示了目前的图块，选择相应图块便可以填充某图了。

④ 可以右键选择图块添加属性，图块的数据可以在程序中获得。

⑤ 工具栏中提供了不同工具填充，包括图章刷、填充。橡皮擦和选择矩形等。可以使用图章刷为每个格填充图素，填充是批量的填充图素，橡皮擦可以擦掉之前的填充图素。

⑥ 通过图层编辑部分可以选择我们需要编辑的图层，并且可以排除其他层的干扰。另外，选择菜单中的图层部分也可以帮助编辑修改图层。

⑦ 地图的相关参数在地图菜单中修改。

⑧ 通过视图菜单修改地图在编辑器中的缩放比例。

⑨ 最后通过另存把TMX文件保存在指定的位置。

（3） 酷跑游戏地图的设计

本酷跑游戏的地图由两个图层以及四个对象层构成。图层map是地图的陆地部分，图层trees是地图的背景部分。由于每个图层只能有唯一的图块资源，所以两个图层的图块分别来自于两张不同的图片。为了使游戏具有可玩性陆地的设置采用高低参差不齐排列的方式，且每块陆地的长度也有分别，但为了保持主角在跳跃后可以成功着陆，陆地之间间隔的大小大致一致，保持在主角跳跃距离以内。四个对象层分别为Role，enemy，items，coll。Role代表主角，将它的初始位置设置在地图的开始位置，且其中只有一个对象。enemy代表炸弹，随机分布在地图中，整张地图有多个。items代表星星，与炸弹一样随机分布在地图中且有多个。coll代表碰撞对象，为了使碰撞检测更加准确，将其设置为每块陆地的边缘部分。

为了保证游戏的可玩性及有趣性，该酷跑游戏的地图制作的相对较长，以保证玩家在游戏过程中体验到不同的游戏画面。

# 4 酷跑游戏主要模块的详细设计与实现

本章主要阐述酷跑游戏主要模块的设计与实现，即游戏主菜单模块、游戏中模块、游戏帮助模块、游戏结束模块以及游戏开发及打包环境的搭建。

4.1 游戏主菜单模块的详细设计与实现

4.1.1 游戏主菜单模块的设计图

游戏主菜单模块主要由几个按钮及背景组成。该模块的详细设计和具体实现如下。图4.1即为游戏主菜单模块的设计图。



图4.1 游戏主菜单模块设计图

4.1.2 游戏主菜单模块代码详细介绍

主菜单模块的详细设计和实现如下。

该模块的内容主要集中在HelloWorldScene.cpp与HelloWorldScene.h中。在HelloWorldScene.h中主要有CCMenuItemImage的声明，主菜单中按钮的回调函数声明以及android返回键监听函数的声明。

（1） CCMenuItemImage是CCMenuItemSprite的子类，关于菜单类的继承关系如图所示。



图4.2 CCMenuItem类的继承关系

（2） 精灵菜单项CCMenuItemSprite的特点是可以封装图片进入菜单项，相关用法如下：首先定义精灵类菜单项CCMenuItemSprite，前三个参数分别是普通状态下的按钮、选中状态下的按钮、无效状态下的按钮，后两个参数是执行回调函数的目标类和回调函数的名称传入选择器中。CCMenuItemImage和CCMenuItemSprite的区别是图片菜单项直接以图片的路径定义。

（3） 开始新游戏按钮的回调函数是menuNewGameCallback，在该函数中实现了从主菜单场景到游戏中场景的切换。在Cocos2D-x中场景切换的动画有许多种，在该函数中采用的是翻页式切换动画。常见的切换动画有跳跃式，淡入淡出，翻页等多种形式，我们可以根据程序需要的效果任意选择。

（4） 游戏关于按钮的回调函数是menuAboutCallback，在该函数中实现了从主菜单场景到游戏关于场景的切换，同样采用的是翻页的动画效果。

（5） 控制声音按钮的回调函数是menuSoundCallback，在该函数中实现了对背景音乐开关的控制。具体实现方法为：在HelloWorld类中定义了一个静态全局变量issound用来表示当前背景音乐状态。当issound为true时表示当前背景音乐为播放状态，则点击按钮使音乐关闭，并将issound置为false；当issound为false时表示当前背景音乐为关闭状态，则点击按钮播放音乐，并将issound置为true。在这里需要提一下Cocos2D-x中采用SimpleAudioEngine类来实现跨平台的声音引擎。而在声音处理方面，Cocos2D-x中包含CocosDenshion库。关于背景音乐，Cocos2D-x在Android平台支持MP3，WAV，OGG等格式。CocosDenshion库中，最常用到的一层是SimpleAudioEngine类，它的头文件在libCocosDenshion文件夹下的Header中。其常用函数如表4.1所示。

表4.1 SimpleAudioEngine类的常用函数

|  |  |  |
| --- | --- | --- |
| 函数名 | 返回类型 | 描述 |
| preloadBackgroundMusic | 空 | 预加载背景音乐 |
| playBackgroundMusic | 空 | 播放背景音乐 |
| stopBackgroundMusic | 空 | 停止背景音乐 |
| pauseBackgroundMusic | 空 | 暂停背景音乐 |
| resumeBackgroundMusic | 空 | 重新开始背景音乐 |
| setBackgroundMusicVolume | 空 | 设置背景音乐音量 |
| playEffect | 整型 | 播放音效，参数为文件路径和是否循环 |
| pauseAllEffects | 空 | 暂停所有音效 |
| resumeAllEffects | 空 | 重新开始所有音效 |
| stopEffect | 空 | 停止音效，参数为播放时获得的ID号 |
| preloadEffect | 空 | 预加载音效 |

在该回调函数中主要使用了播放和停止背景音乐两个函数。

（6） 监听Android手机返回键的回调函数是buttonCallback。为了达到点击Android手机返回键有弹窗跳出提示的效果，又封装了一个自定义类PopupLayer。在该类中主要实现了以下内容：

① 重写触摸注册函数，重新给定触摸级别。

② 重写触摸函数，返回true屏蔽其它层，达到“模态”效果。

③ 静态创建函数，创建一个弹出层，设置背景图片。

④ 自定义设置标题函数，参数有标题内容及字体大小。

⑤ 设置文本内容，参数分别为文本内容，字体大小，文字到对话框两边预留的距离，距上方的距离，这些值均是可控的。

⑥ 设置上层对象和上层回调函数，用于回调时传递CCNode参数。

⑦ 添加MenuItem按钮，封装了一个函数，传入些必要的参数。

⑧ onEnter函数，为了在显示层时的属性生效，选择在onEnter里动态生成。

⑨ 定义int型变量表示文字内容两边的空白区域。

⑩ 定义CCMenu\*类型变量m\_pMenu,并且直接定义默认的set/get方法。

调用PopupLayer中addButton函数为该弹窗添加按钮，用tag表示每个按钮，tag=0表示用户同意退出的按钮，tag=1表示用户不同意退出的按钮。在回调函数中通过tag来判断用户选择进行不同的处理，若tag=0直接退出程序，若tag=1则仍停留在主菜单页面。

4.2 游戏中模块的详细设计与实现

游戏中模块的实现较为复杂，是整个游戏的核心部分。这里将该模块根据实现的类拆分成几个部分来介绍。

4.2.1 GameMainScene类

GameMainScene类的设计图如下所示。



图4.3 GameMainScene类设计图

由该设计图可以看出GameMainScene类主要进行的初始化工作，为了使用Role类中的内容，在GameMainScene类中对Role类进行声明且包含了Role.h.。下面详细介绍每一部分的具体内容。

（1） 初始化地图：该函数中调用瓦片地图的create函数，对地图进行初始化，加入到游戏中场景中。

（2） 初始化主角：该函数调用Role类中的静态成员函数，将主角加入到地图中并获得该主角对象。

（3） 初始化道具：因为一个游戏中会出现许多道具，所以使用CCSpriteBatchNode类来放置星星及炸弹对象。之后调用resetComponents函数，在该函数中获得地图中的items对象层以及enemy对象层。通过查看TMX格式的地图文件可知，对象层中的每一个对象都有自己的名字及x坐标和y坐标，以键值对的形式存储。我们在程序中可以由此获得每一个道具对象的坐标并为其赋上图片，这样我们就成功的把星星及炸弹图片加入到地图的合适位置上，再将它们分别加入到数组中以备后来消除使用。

（4） 初始化游戏背景：游戏背景由两张图片构成，一张做为底部背景，另一张覆盖其上。为了使背景可以无限循环滚动采用的是酷跑游戏中最常用的做法：两个背景精灵循环交替。背景的移动利用的是Cocos2D-x中的动画效果，使用CCMoveBy函数，当一张图片移动到边界后便调用回调函数，重置这张图片的位置，使另一张图片开始移动。

（5） 暂停游戏按钮：暂停按钮使用CCMenu实现，其回调函数为onPause。在该回调函数中首先新建一个暂停界面覆盖在屏幕上，再为该界面添加恢复按钮并使上级按钮暂时失效，并调用导演类使该场景暂停。暂停恢复按钮的回调函数onResume恰好与该函数功能相反，其使上级暂停按钮恢复有效并移出暂停界面，调用导演类恢复场景运行。

（6） 返回主菜单按钮：返回主菜单按钮同样是使用CCMenu实现，其回调函数为BackToMenu。该回调函数中首先新建一个主菜单界面的场景，并调用replaceScene函数，用新建场景以翻页形式替换当前场景。

（7） 触摸函数的实现：为了实现屏幕的可触摸性，在该类中重写了ccTouchBegan，ccTouchMoved，ccTouchEnded三个函数。当玩家点击屏幕的时候，在ccTouchBegan中调用dispatchRoletTouchMove函数，在该函数中判断如果主角当前是奔跑或者站立状态，则设置主角状态为跳跃，否则调用jumpjump函数实现连续跳跃。

4.2.2 Role类

Role类是酷跑游戏主角及道具的实现部分，为了使程序结构更加清晰，道具中的星星和炸弹分别新建了两个类Gold和Enemy来实现，分数结算部分新建了类GameMark来实现。星星及炸弹两个道具在游戏中的作用很相似，所以采用的实现方法也比较接近，GameMark类为自己封装的一个类，具有很强的移植性，在以后的游戏开发中可以直接使用此类所封装的代码。

Role类的设计图如下。

 图4.4 Role类设计图

下面将详细介绍Role类的各个部分内容。

（1） 主角初始化部分：上节提到在GameMainScene类中使用了Role类中的主角对象，但是主角的真正初始化部分是在Role类中实现的。在extraInit函数中，先从地图中获得对象层Role，再根据对象的名字获得主角的坐标。在这里为了实现主角奔跑的效果，为主角添加了动画。与之前背景移动的动画不同，这里主角动画使用的是plist文件，该文件可使用TexturePackerGUI制作。在plist文件中会按照动画的播放顺序存有每一张图片的名字，我们就可以根据名字获取plist文件中的每一张图片。

（2） 镜头部分：在酷跑游戏中主角需要不断的奔跑，即坐标需要不断改变，这样的话主角就会跑出屏幕，为了解决这个问题，就引入了酷跑游戏中的镜头。当主角移动到的位置大于屏幕四分之一时，镜头就开始移动，即屏幕中的地图开始移动，这样就可以保持主角一直处于屏幕的中央位置。因为该游戏中地图为有限长度，在移动的过程中总有一刻，地图要全部跑完，为了解决这个问题，在地图将要跑出屏幕时，我将主角及地图又重新置于游戏开始时的位置。而主角及地图位置的实时更新，都是依赖于定时器函数实现的，在程序中将定时器函数设置为每一帧及0.016s调用一次。

（3） 主角不同状态：在程序中使用枚举变量表示主角的不同运动状态，分别有无效状态、奔跑、向上跳、下落、站立和自由落体。同样是使用定时器函数每隔一帧调用一次roleUpdate函数，根据主角的不同状态进行不同的处理。若主角当前是奔跑状态则调用run\_logic函数。在该函数中先进行向右的碰撞检测，再进行向下的碰撞检测，然后重新设置主角的位置。若主角当前是向上跳，则先播放跳跃动画，再调用jumpUp\_logic函数。在该函数中先进行向上的碰撞检测，再进行向右的碰撞检测，然后设置主角的位置。若主角当前是下落状态，则调用jumpDown\_logic函数。在该函数中先进行向下的碰撞检测，再进行向右的碰撞检测，如果主角的y坐标已小于0，则调用gameOver函数，表示主角已死亡，反之重新设置主角的坐标。若主角为自由落体状态，则调用roleFreeFallLogic函数。在该函数中先判断主角y坐标是否小于0，小于则直接进行场景的切换，然后进行向下的碰撞检测，重新设置主角的坐标。

（4） 与山体碰撞检测：碰撞检测部分可以说是酷跑游戏中重要组成部分。为了进行碰撞检测，需要先获得山体部分。在该游戏中使用一个数组mCollisionArray来存储需要进行碰撞检测的山体。具体步骤如下：先从地图中获取对象层coll，然后得到其中的对象，并对对象的x坐标进行由小到大的排序，排序之后存储于mCollisionArray中。每当一个山体进行碰撞检测完之后则将其从数组中移出，否则获得当前山体的碰撞矩形区域。在isCollWithCollArea函数中根据碰撞检测位置的不同，碰撞检测的方式也有所不同。当进行向上的碰撞检测时调用isCollisionWithTop函数。在该函数中获得主角矩形区域的x坐标中间点及最大y坐标，并判断该点是否在山体的矩形区域内，若在则表示发生了碰撞，反之则未发生碰撞。当进行向下的碰撞检测时调用isCollisionWithBottom函数。在该函数中获得主角矩形区域的x坐标中间点和最小y坐标，并判断该点是否在山体的矩形区域内，若在则表示发生了碰撞，反之则未发生碰撞。当进行向左的碰撞检测时调用isCollisionWithLeft函数。在该函数中获得主角矩形区域的最小x坐标和y坐标中间点，并判断该点是否在山体的矩形区域内，若在则表示发生了碰撞，反之则未发生碰撞。当进行向右的碰撞检测时调用isCollisionWithRight函数。在该函数中获得主角矩形区域的最大x坐标和y坐标中间点，并判断该点是否在山体矩形区域内，若在则表示发生了碰撞，反之则未发生碰撞。

（5） 碰撞修复：为了使碰撞检测更加的准确，增加了这一部分的内容。在该函数中先获得主角矩形区域与山体区域x，y坐标的最大最小值，并求得x坐标最大值与最小值之差width及y坐标最大值与最小值之差height。当向下碰撞时，将主角的y坐标设置为当前y坐标与height之和；当向上碰撞时，将主角的y坐标设置为当前y坐标与height之差；当主角向左碰撞时，将主角的x坐标设置为当前x坐标与width之和；当主角向右碰撞时，将主角的x坐标设置为当前x坐标与width之差。

（6） 与道具碰撞检测：与道具碰撞检测包括两部分，一部分是与星星的碰撞检测，另一部分是与炸弹的碰撞检测，但是二者的原理是相似的。与星星的碰撞检测调用collsionWithGold函数。在该函数中先从存储星星的数组中获得Gold对象，调用Cocos2D-x的封装函数intersectRect进行星星与主角的碰撞检测，若发生碰撞则播放音效并播放粒子效果，同时从数组中将该星星移除。在该游戏中设置成每吃到一颗星星则加10分，所以之后调用addnumber函数进行加分。与炸弹的碰撞检测调用collisionWithEnemy函数。在该函数中先从存储炸弹的数组中获得Enemy对象，调用Cocos2D-x的封装函数intersectRect进行炸弹与主角的碰撞检测，若发生碰撞则播放音效并设置主角为站立状态，将屏幕设置为不可触摸，以解决在主角死亡后玩家点击屏幕主角仍可跳动的bug。同时，将该炸弹从数组中移出，用一张炸弹爆炸效果的图片代替当前炸弹，播放主角被炸死的动画后调用gameOver函数进行后续处理。

4.3 游戏帮助模块的详细设计与实现

游戏帮助模块主要是用来为用户提供游戏帮助的部分，实现部分较为简单，下图所示为游戏帮助模块GameAbout类的设计图。



图4.5 GameAbout类设计图

帮助界面的背景是用PhotoShop制作的一张有关游戏规则的图片。返回主菜单按钮同样采用CCMenu，其回调函数为menuBackCallBack，该函数中首先新建一个HelloWorld类的场景，之后调用replaceScene函数以翻页的形式跳转到游戏主菜单界面。

4.4 游戏结束模块的详细设计与实现

4.4.1 游戏结束模块设计图

游戏结束模块是酷跑游戏的最后一个页面，主要由几个按钮及其回调函数构成。GameOverScene类的设计图如下所示。



图4.6 GameOverScene类设计图

4.4.2 游戏结束模块代码详细介绍

（1） 显示分数：为了使游戏更加美观，在显示游戏分数的时候该酷跑游戏未直接使用Cocos2D-x中的字体，而是重新封装了一个类GameMark来显示分数。该类主要由onEnter和addnumber两个函数构成。在onEnter函数中，首先将分数标题加入场景中，之后将包含有数字0~1的图片的每一位读出存入数组bits中。函数addnumber则分别获得分数的每一位，根据每一位的数字选择bits数组中图片。为了使分数的获得更加公正，该游戏的分数除与吃到星星有关外，还与主角奔跑的距离有关。该分数在游戏中界面实时更新，游戏结束界面的分数由该分数读得，并通过一个CCLableTTF对象显示在屏幕上。

（2） 按钮及回调函数：该界面中的按钮也均是使用CCMenu实现的。点击再玩一次游戏的按钮，游戏则重新开始，其回调函数为TryAgain函数，在该函数中首先新建一个GameMainScene类的场景，然后调用replaceScene函数以翻页形式跳转到游戏中界面。点击返回主菜单按钮，则返回游戏主菜单，其回调函数为BackToMainMenu函数，在该函数中首先新建一个HelloWorld类的场景，然后调用replaceScene函数以翻页形式跳转到游戏主菜单。点击退出游戏按钮，则退出该游戏，其回调函数为QuitGame函数，在该函数中直接使用导演类调用end函数以结束游戏。

4.5 游戏开发环境及打包环境的搭建

4.5.1 游戏开发环境的搭建

下面介绍在Windows中进行开发，开发环境的搭建步骤：

（1） 安装VS2010或者其他版本的VS及Python。

（2） 进行环境变量的配置，在环境变量PATH中添加Python的安装路径。

（3） 创建新项目：在开始菜单中输入cmd弹出命令行界面，并将路径切换到cocos2dx文件夹中project-creator文件夹下。输入命令：python create\_project.py -project 项目名 -package org.cocos2dx.项目名 -language cpp

经过以上几步，我们就成功的新建了一个工程，之后在该工程下进行项目的开发即可。

4.5.2 打包环境的搭建

在使用Cocos2D-x进行项目开发的时候，打包这个环节一直令许多程序员头痛，在进行环境搭建的时候常会遇到许多非常奇怪的问题，下面简单介绍一下打包环境的搭建过程。

（1） 首先需要在电脑上安装Android版的eclipse以及Android NDK。

（2） 将Cocos2D-x项目导入到eclipse中。

（3） 在eclipse的配置选项中添加两个变量分别是NDK\_ROOT和COCOS2DX，NDK\_ROOT的路径为X:\cocos2d-x\android-ndk-r9b，COCOS2DX路径为X:\ cocos2d-x\cocos2d-x-2.2。

（4） 这一步未使用普遍流行的安装Cygwin的方法，而是修改C++ Build的属性如下：Build Command：X:\cocos2d-x\android-ndk-r9b\ndk-build.cmd。

（5） 在Android.mk文件中添加如下语句：

$(call import-add-path, X:\cocos2d-x\cocos2d-x-2.2)，

$(call import-add-path, X:\cocos2d-x\cocos2d-x-2.2\cocos2dx\platform\ third\_party\android\prebuilt)。

（6） 在Android.mk文件中添加所有cpp文件的声明。

经过以上步骤就可以成功在模拟器上运行我们的程序了，并可以在bin文件夹下获得apk文件，将其安装到Android手机上运行。

5 系统测试

本章将对酷跑游戏使用的测试方法进行介绍，并在之后分别进行功能测试和性能测试，确保软件各项指标均满足系统需求。

5.1 测试环境

该酷跑游戏的测试环境主要由以下几部分组成：

（1） 一台装有该游戏的安卓智能手机

（2） 一台电脑

（3） 一根USB数据线

对酷跑游戏的测试主要是通过直接对手机中该软件进行操作。通过eclipse或者USB数据线将该游戏安装到Android手机中，进而对酷跑游戏的游戏主菜单，游戏中界面，游戏结束界面进行测试。并且通过持续操作测试其稳定性。具体的测试方法和测试内容下文进行详细介绍。

5.2 测试方法

任何的工程产品都可以使用如下两种方式进行测试：白盒测试和黑盒测试。

白盒测试：白盒测试就是将软件看成一个透明的盒子，对产品的内部工作原理要非常了解，经过测试证明其内部操作是否符合需求设计规格的要求。白盒测试通常又被称为结构测试、基于程序本身的测试或逻辑驱动测试。

黑盒测试：黑盒测试与白盒测试相反，它不要求测试人员对产品的内部有任何了解，只是把产品看作一个黑色的盒子，从接口处测试产品的功能。因此黑盒测试又叫功能测试或数据驱动测试

在该游戏的开发过程中，白盒测试的方法只要是在编码测试和集成测试阶段被采用。通常白盒测试的工作主要是由程序的开发人员在编码完成之后所编代码进行测试。在进行测试的时候，并不能单纯使用白盒测试，还需要选用黑盒测试对测试方案进行补充，这样测试效果会更加的好。

5.3 功能测试

功能测试适用于检测软件实现功能是否满足需求，这里采用黑盒测试的方法分别对酷跑游戏的四个界面进行功能测试。

5.3.1 游戏主菜单界面测试

（1） 测试目的

检测主菜单界面的按钮是否可用，对Android手机返回键监听是否正确。

（2） 测试内容

检测点击按钮时是否可以正确实现相应功能，可点击区域是否正确，主菜单背景是否正确显示，控制声音按钮是否可以正确打开或者关闭背景音乐，是否可以监听Android手机返回键。

（3） 测试结果

图5.1为游戏主菜单界面测试效果图。分别为游戏主菜单界面和点击Android手机返回键时的弹出对话框。





图5.1 游戏主菜单界面测试效果图

5.3.2 游戏帮助界面测试

（1） 测试目的

检测帮助界面是否正常显示，按钮是否可用。

（2） 测试内容

检测点击返回按钮时是否正确返回到主菜单界面。

（3） 测试结果

图5.2为游戏关于界面测试效果图。



图5.2 游戏关于界面测试效果图

5.3.3 游戏中界面测试

（1） 测试目的

检测游戏是否可以正常运行，音效播放是否正确，按钮是否可用。

（2） 测试内容

检测点击屏幕主角是否可以正常跳跃，碰撞检测是否正确，碰到道具时音效是否正确播放，地图及背景是否正常移动，主角动画是否正确播放，屏幕中按钮是否可用，主角是否一直处于屏幕中央，游戏分数是否可以正确的实时更新。

（3） 测试结果

图5.3为游戏中界面测试效果图。分别为主角跳跃画面，游戏暂停画面以及主角死亡画面。







图5.3 游戏中界面测试效果图

5.3.4 游戏结束界面测试

（1） 测试目的

检测游戏结束界面是否正常显示，检测按钮是否可用。

（2） 测试内容

点击每个按钮，检测是否正确实现其功能，检测最后得分是否正确显示。

（3） 测试结果

图5.4为游戏结束界面测试效果图。



图5.4 游戏结束界面测试效果图

5.4 性能测试

对该游戏进行的性能测试如表5.1所示。

表5.1 酷跑游戏性能测试表

|  |  |  |
| --- | --- | --- |
| 测试项 | 测试内容 | 测试结果 |
| 时间特性  时间特性  时间特性 | 点击游戏中按钮，切换场景需要的时间  点击屏幕主角相应时间  背景音乐及音效播放所需时间 | 小于1秒  小于1秒  小于0.5秒 |
| 时间特性 | 对Android手机返回键的响应时间 | 小于0.5秒 |
| 稳定性 | 设备正常使用时不断打开关闭游戏，持续玩 | 未发现异常 |

通过测试结果，我们可以发现该酷跑游戏各项指标均满足系统需求。

5.5 本章小结

本章开头首先对白盒及黑盒测试方法进行了简要的介绍，并在之后分别对酷跑游戏进行了功能和性能测试，在测试中给出了部分游戏界面的截图，并且以表格的形式对各项性能测试进行说明分析，结果表明该游戏各项指标均满足系统需求，成功通过测试。

# 结 论

本文第一章对该项目的背景及Cocos2D-x手机游戏引擎做了详细的介绍，主要是利用相关类之间的继承关系图，向读者清晰的展示出Cocos2D-x引擎的核心内容，便于读者理解本文中所用到的一些Cocos2D-x引擎的相关知识。第二章主要介绍酷跑游戏的分析及设计，主要从社会可行性，经济可行性，技术可行性三个方面进行可行性分析，之后又介绍了酷跑游戏的需求分析，分别从功能需求与性能需求两个方面进行介绍。该部分进行分析时将整个游戏分成三个界面，即游戏主菜单界面，游戏中界面，游戏结束界面，并为每个界面绘制用例图，可视化的表示系统需求，以明确系统内部与系统外部（也就是角色）的交互。第三章主要介绍酷跑游戏的总体设计，包括酷跑游戏的设计原则，酷跑游戏的层次架构，酷跑游戏的功能模块图，以及酷跑游戏的图片和音频资源四个方面。之所以介绍酷跑游戏的图片和音频资源是因为在制作地图时使用到了一款叫做Tiled地图编辑器的软件，这款软件在游戏开发中常常会被用到。第四章主要介绍酷跑游戏主要模块的详细设计与实现，包括游戏主菜单模块的详细设计与实现，游戏中模块的详细设计与实现，游戏帮助模块的详细设计与实现，游戏结束模块的详细设计与实现，游戏开发环境及打包环境的搭建。该部分以程序中的类为单位对整个程序的架构进行详细的介绍。

本文的相关研究已经取得了一定的成果，但是由于时间及技术上的原因还有很多地方可以进行进一步的完善。首先，在游戏界面方面，由于作者本身非美工出身，对于PhotoShop等图像处理软件的使用还有些生疏，所以图片资源方面借鉴了如今已经发布的一些游戏，这对于一款想要发布到Android市场的游戏是不可取的。在这方面可以进行一下创新，设计出属于自己的游戏人物及游戏场景。其次，目前的酷跑游戏只有一种难度，可能对于一些玩家会觉得没有挑战性，游戏可以再设置一些其他关卡或者随着分数的提高适当增加游戏难度，这样的话可以增加游戏的趣味性及挑战性。最后，在游戏中还可以增加其他的道具，例如：可以让主角加速的道具，变大的道具，或者增加带有攻击性的敌人，并使主角可以发动攻击等。还有目前的游戏只支持主角进行两连跳，可以再进行扩展，使支持多次跳跃，并增加主角的跳跃动画等。

参 考 文 献

[1] 满硕泉.Cocos2D-x权威指南[M].北京:机械工业出版社,2013.

[2] 徐松林，黄猛.Cocos2d-x手机游戏开发[M].北京:人民邮电出版社,2012.

[3] 刘凯威.Coco2D-x for XNA游戏开发指南（下）《TweeJump》项目实战[J].程序员,2012,7:87-91.

[4] 秦春林.Cocos2D-x数据驱动游戏开发实践[J].程序员,2013,10:83-87.

[5] 舒沂.基于cocos2d-x引擎的手机游戏快速开发工具的设计与实现[D].南京：南京大学,2013.

[6] 沈大海.Cocos2d-x手机游戏开发与项目实战详解[M].北京:清华大学出版社,2014.

[7] Shi Qianqian.Design and Implementation of Game Framework for cross-platform on Smart device[J].Electronic Test,2014,1:90-107.

[8] Xing Fang Zhang Xiaoqin.Design of Eliminate Class Game based on Cocos2d-x[J].Science Mosaic,2013,8:93-112.

[9] 火烈鸟网络科技.Cocos2d-x高级开发教程：制作自己的《捕鱼达人》[M].北京:人民邮电出版社,2013.

[10] 刘剑卓.游戏设计与开发技术丛书：Cocos2D-x游戏开发技术精解[M].北京:人民邮电出版社,2013.

[11] 蒋恒恒，章国稳.利用粒子系统实现喷泉模拟[J].重庆邮电大学学报,2008，20：749-753.

[12] Hayit Greenspan.Image Enhancement by Nonlinear Extrapolation in Frequency Space[J].IEEE Transactions on Image Processing,2000,9:1035-1048.

[13] Bjame Stroustrup.The C++ Programming Language[J].Addison-Wesley,2000,8:78-89.

[14] Weidenhaupt K,Pohl K,Jarke M,Haumer P.Scenarios in system development:current practice[J].IEEE Software,2011,6:110-121.

[15] 牛犇.基于ios的2D第三人称射击游戏引擎的研究与实现[D].苏州:苏州大学,2012.

# 致 谢

经过这段时间的不懈努力，论文终于即将完成。整个过程中我也遇到了不少挫折和难题。在这里我要向所有指导过我的老师，帮助过我的同学，一直支持着我的家人以及评阅老师表示深深的谢意！

首先，我非常的感谢我的指导老师，从开始的选题、开题到撰写，都离不开指导老师给予我的关心与指导，是老师孜孜不倦的教导使得我的论文可以顺利完成。同时老师严谨的教学态度与认真的工作作风对我产生了深刻的影响，在以后的学习生活中我也将沿袭这种良好的作风。

在此，我还要感谢为我提供帮助的同学们，在我遇到困难止步不前时，是他们的帮助让我打起精神，继续努力下去。

感谢我的家人，他们在背后默默的给予我支持与鼓励，不论我遇到多大的困难，他们的一声鼓励都会帮助我重新树立起信心。

感谢各位专家在百忙中审阅我的论文，感谢专家们的宝贵建议。

最后，再次向我的老师、同学、家人和专家们致以最衷心的谢意。