|  |
| --- |
| **基于Cocos引擎的微信小游戏架构和设计** |

|  |
| --- |
| WeChat Game Architecture and Design Based on Cocos Engine |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 学生姓名 | 梁庭嘉 | 学号 | | 201411921513 | | |
| 所在学院 | 数学与计算机学院 | | 班级 | | | 计数1142 |
| 所在专业 | 信息与计算科学 | | | | | |
| 申请学位 | 理学学士 | | | | | |
| 指导教师 | 周永雄 | | 职称 | | 讲师 | |
| 副指导教师 |  | | 职称 | |  | |
| 答辩时间 | 2018年 6 月 3 日 | | | | | |

**目 录**

[设计总说明 I](#_Toc515828511)

[**introduction** II](#_Toc515828512)

[1 绪论 3](#_Toc515828513)

[1.1 设计背景及意义 3](#_Toc515828514)

[1.2 小游戏的现状和发展趋势 3](#_Toc515828515)

[2 游戏开发及相关环境 4](#_Toc515828516)

[2.1 开发环境 4](#_Toc515828517)

[2.2 开发工具 4](#_Toc515828518)

[2.3 开发相关技术 4](#_Toc515828519)

[3 Cocos create技术分析 5](#_Toc515828520)

[3.1 功能特性 5](#_Toc515828521)

[3.2 架构特色 5](#_Toc515828522)

[4 游戏布局与分析 6](#_Toc515828523)

[4.1 游戏总体开发流程与设计 6](#_Toc515828524)

[4.2 游戏UI设计 7](#_Toc515828525)

[4.3 音效设计 7](#_Toc515828526)

[4.4 游戏场景结构与交互设计 8](#_Toc515828527)

[4.5 游戏目录结构 9](#_Toc515828528)

[5 游戏系统实现 9](#_Toc515828529)

[5.1 资源渲染 9](#_Toc515828530)

[5.2 界面设计 10](#_Toc515828531)

[5.3 碰撞设计 16](#_Toc515828532)

[6 测试与发布 18](#_Toc515828533)

[6.1 微信开发软件打包 18](#_Toc515828534)

[6.2 游戏测试 21](#_Toc515828535)

[7 设计总结 23](#_Toc515828536)

[鸣 谢 24](#_Toc515828537)

[参考文献 25](#_Toc515828538)

# 设计总说明

随着智能手机的快速发展，人们使用手机所进行的娱乐活动也越来越丰富，在追求简单方便又快捷的条件下，微信小游戏应运而生：不用下载，不占内存，随时随地可以开始游戏，其便捷性等不言而喻。因为微信丰富的生态系统和海量的用户群体，也为小游戏带来了的全新的发展机遇。

本文提出了一种基于Cocos2d-x的微信小游戏，该游戏使用物理碰撞检测系统来处理游戏进程，基于Cocos2d-x的游戏也具有更好的性能及较小的体积。任何使用微信的手机用户都可以轻松的进行游戏，覆盖范围及其广阔。在这个游戏中，玩家通过释放小球，消灭砖块来进行娱乐。看似简单，但只要接触，你就会发现小小游戏里面的大惊喜：游戏的关键点在于运用物体的运动角度，在游戏中可以产生叫做黑洞的砖块和三角形砖块提供更多的碰撞角度，增加游戏体验和乐趣。该游戏以软件工程思想为指导，分析了游戏策划方案，以经典的打砖块游戏为基础，采用数据驱动模式和MVC模式等设计模式，设计用一款基于Cocos2d-x引擎的打砖块游戏。论文主要工作包括游戏程序结构设计和游戏场景结构设计的完成过程，以及游戏系统的开发。游戏的界面制作优美大气，运行过程流畅，功能设计基本达到了预期目标。

论文在撰写过程当中，以实践为基石，以理论为框架，在对微信小游戏开发理论举行论述的同时，配合思维导图以现实程序和游戏源码上加以说明，以期达到对Javascipt和Cocos2d-x的更深层次的理解与认识。

关键词：打砖块；CocosCreate；Javascript；微信小游戏

# **introduction**

With the rapid development of smart phone, there is a common-seen phenomenon that more and more people use it to do many entertaining activities. Under the condition that people nowadays pursuit a simple, convenient and quick way of life, WeChat games arise at the historic moment as there's no need to download and take memory. Since at anytime and everywhere you can start playing, its convenience speaks for itself. As WeChat has a rich ecosystem and a large number of users, it in turn offers a new opportunity for these games.

Based on Cocos2d - x, this paper puts forward a kind of such game, which uses a physical collision detection system to handle the process that makes a better performance and smaller volume, and every mobile phone user can play WeChat games easily as there is a wide range of coverage. In this game, players enjoy themselves by releasing balls and destroying bricks, which may look simple, but as long as you keep playing, you will find big surprises in it. The key point of the game is to use the angle of object to make movement, in which it can produce bricks called black hole and triangle bricks to provide more collision angles, which adds much more pleasure to the game. Software engineering ideas are the guidance and planning scheme of the game. Meanwhile, based on the classical game of playing bricks and using the designed patterns, such as the observer pattern and factory pattern shown as the leisure elimination game, which is also based on Cocos2d-x. The main work of the thesis includes the process of game program structure design and the completion of game scene structure design, as well as the development of game system. The interface of the game is simple and beautiful, running smoothly, and the functional design has basically achieved the expected goal.

During the process of writing this paper, with the practice-based theories as the framework, discussion of its development as well as cooperation with mind mapping explained by the actual application and game source code so as to reach the JavaScript and create a deep understanding and profound knowledge.

**Keywords:** Bricks; CocosCreate; Javascript; WeChat Game

**基于cocos引擎的微信小游戏架构和设计**

信息与计算科学，201411921513，梁庭嘉

指导教师：周永雄

# 1 绪论

## 1.1 设计背景及意义

微信小游戏在2017年12月正式上线(不对个人开发者开放)。2018年4月全面开放。相对于传统的基于H5的游戏，微信小游戏的优势非常明显，拥有微信巨大的用户量以及很好的兼容性。**小游戏实质是Runtime游戏，它跟小程序相似，内部结构是小程序+游戏库API。**在这个架构之上的应用，会有更多优势。第一，它会比H5游戏更稳定；第二，小游戏切换出重进后，用户不会像在H5游戏中那样需要重新登录；第三，小游戏可以调用微信的用户资料，在转发、支付、分享等方面也有对应接口，这对整个微信的生态系统发展是有很大推动作用的。

## 1.2 小游戏的现状和发展趋势

未来微信小游戏的发展将会是高活跃的游戏广告和高收入的虚拟道具并驾齐驱的局面：在前期主要是发展高活跃用户，等到小游戏这个平台发展强大后，再转向高收入的游戏。未来小游戏将会和App游戏互联互通，用户可以在APP游戏中分享数据给微信好友，也可以借助小游戏来唤醒App游戏，并且这一功能将会开放给所有的厂商和个人开发者。基于“即点即玩”的特点，等到小游戏和微信其他业务完全互通之后，会有更大的发展空间。比如公众号文章下面添加小游戏二维码或者链接，用户点进去就能玩；或者在小程序下拉界面可以系统上架几款游戏模式等。而且小游戏在朋友圈和公众号文章的信息流投放广告，这也是一种有效且快捷吸收新用户的方法。

小游戏平台的定位，在前期会侧重于“高传播性”与“评价较好”的游戏类型。官方的初衷是在前期给小游戏树立“小而美”的形象，通过游戏的品质提高整个小游戏平台的水准。官方也表示已经和许多著名游戏厂商建立了合作关系，将来会有多款育碧旗下 Ketchapp 游戏接入到微信小游戏的平台。但是目前来看，小游戏也不乏短板。微信小游戏基于小程序，所以缺少微信过去“发现-游戏”这样一个传播平台，传播主要靠分享和排行榜，这就导致新的游戏传播是被动的，里面的人为因素对小游戏甚至起到了一定的决定作用，这也使得小游戏不能得到充足的传播保障。一个典型的例子就是，微信目前至少上线了几千款小游戏，但有很大一部分人还仅只知道“跳一跳”这一日常款的小游戏。这种单一的游戏模式也就加大了马太效应：火爆的小游戏“一传十，十传百”，会越来越火爆；而得不到分享的游戏，因为缺乏其他传播手段，就会慢慢“死”在摇篮里。

如何在发展前期得到最大程度的分享和传播，这是微信小游戏必须思考的问题。但可以预见的是，背靠用户活跃量超过9亿的微信，小游戏也会在2018年继续蓬勃发展，到达一个新的高度。

# 2 游戏开发及相关环境

## 2.1 开发环境

微信小游戏和cocos引擎及相关开发工具在windows 10运行，前期游戏测试在浏览器Crome中。

## 2.2 开发工具

1. VSCode

[VSCode](https://www.iplaysoft.com/visual-studio-code.html)(全称[Visual Studio Code](https://www.iplaysoft.com/visual-studio-code.html))是微软在2015年推出的一款免费、[开源](https://www.iplaysoft.com/tag/%E5%BC%80%E6%BA%90)、跨平台的轻量级代码编辑器，支持语法高亮、代码自动补全、编辑器调试、控制台工具、代码对比 Diff、GIT 等功能，支持多种插件扩展和主题，并针对网页[开发](https://www.iplaysoft.com/tag/%E5%BC%80%E5%8F%91)和云端应用开发做了特殊优化。

1. CocosCreate

Cocos Creator 是雅基软件在2014年推出的一个支持 Windows 和 Mac 跨平台运行的游戏开发编辑器，双击便可进入游戏开发。相比传统的 Cocos2d-x 复杂工作流程和编译程序，将配置开发环境的要求完全免除，运行之后就可以立刻开始游戏内容制作或功能开发。

1. PhotoShop CS6

Photoshop CS6 它是由Adobe在2013年推出的新一代的图像处理软件。集图像扫描、编辑修改、图像制作、广告创意，图像输入与输出于一体的图形图像处理软件，深受广大平面设计人员和电脑美术爱好者的喜爱。它功能强大,操作界面友好,得到了广大第三方开发厂家的支持,从而也赢得了众多的用户的青睐。

## 2.3 开发相关技术

1. Javascript

Javascript是一种基于原型、面向对象脚本语言，具有其独特的垃圾回收等机制，他的解释器被称为Javascript引擎，是浏览器的一部分，目前最新的Javascript引擎为V8基于C++开发，并在谷歌Chrome浏览器和Node.js服务端中使用。Javascript主要用于客户端的脚本语言，最早用来给HTML网页增加动态功能。目前他是很多脚本语言中使用人数最多应用最广的一种，随着HTML5的出现，Javascript的功能已经扩展到服务端开发。配合HTML5就能实现多平台web软件开发。

# 3 Cocos create技术分析

## 3.1 功能特性

1. 可以随时的调整在脚本中定义的属性，也可以在编辑器中定义属性通过脚本调整。
2. 支持智能画布自适应的UI系统和自定义的对齐功能可以完美地自适应多种分辨率的设备。
3. 专门为2D游戏制作的动画系统，支持动画轨迹预览和编辑复杂曲线。
4. 支持动态语言编写脚本进行开发，使得动态调整和移动设备的远程调整变得非常容易。
5. 使用Cocos2d-x引擎，既可以享受可编写脚本开发的便利性，还可以通过单击构建在各种桌面平台和移动终端平台上发布游戏，并保留原来的编辑功能。
6. 动态脚本和开放插件系统的组件化为开发人员提供了在不同深度定制工作流程的方法。组件化开发可以适应于不同层次和项目的需求

## 3.2 架构特色

1. Cocos Creator包含游戏开发所需的所有功能，包括：游戏引擎，资源管理，场景编辑，游戏预览和游戏发布。 所有这些都在一个单一的，统一的工具中，可以让您的游戏开发变得简单。
2. 它以数据驱动设计和组件化成为核心游戏开发方法，并无缝集成了Cocos引擎的成熟JavaScript 脚本系统。 它可以适应开发者的习惯，并且继续提供前所未有的内容创建水平，并为艺术工作者和导演提供实时预览和测试环境。
3. 这款游戏开发软件不仅集成了强大的工具包，而且还提供了一个开放式插件框架。 开发人员可以使用Javascript等通用前端技术来方便地扩展编辑器的功能和提供定制的个性化工作流程。

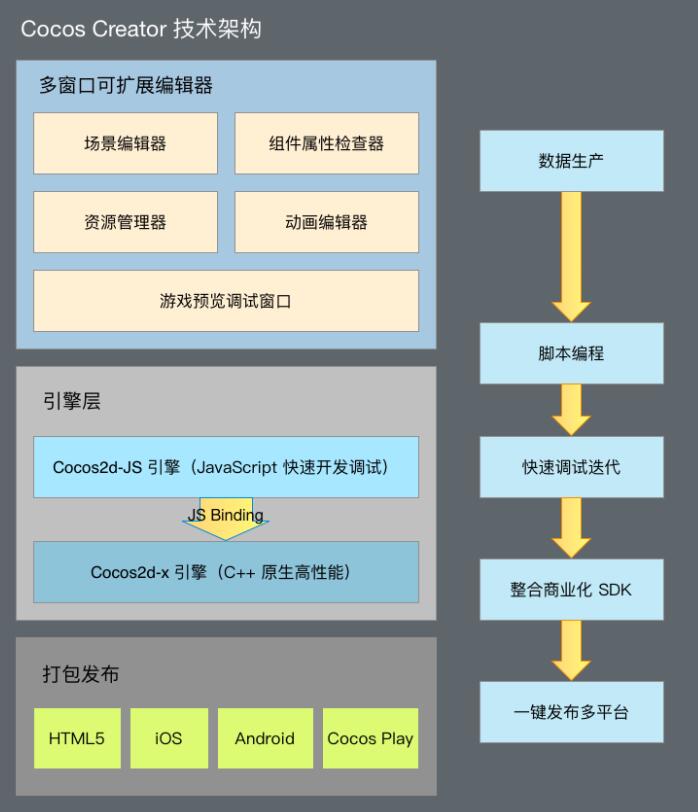


图1 CC架构设计

# 4 游戏布局与分析

## 4.1 游戏总体开发流程与设计

图1 总体开发流程与设计

图2 总体开发流程与设计

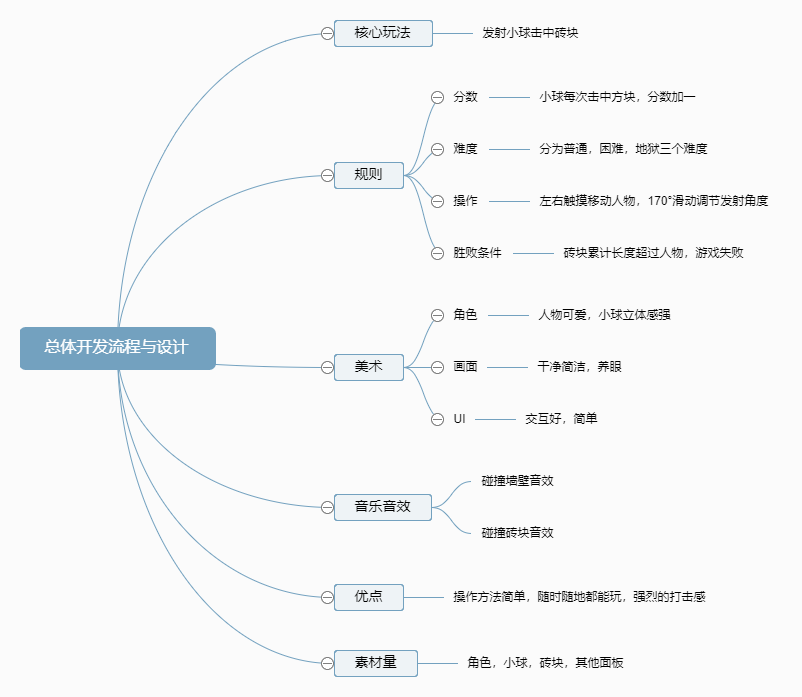


图2 总体流程设计图

## 4.2 游戏UI设计

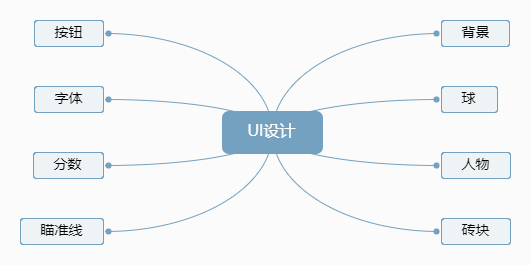


图3 游戏UI设计

## 4.3 音效设计

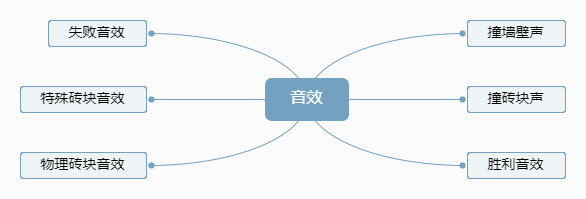


图4 音效设计

## 4.4 游戏场景结构与交互设计

图5 游戏场景与交互设计

## 4.5 游戏目录结构

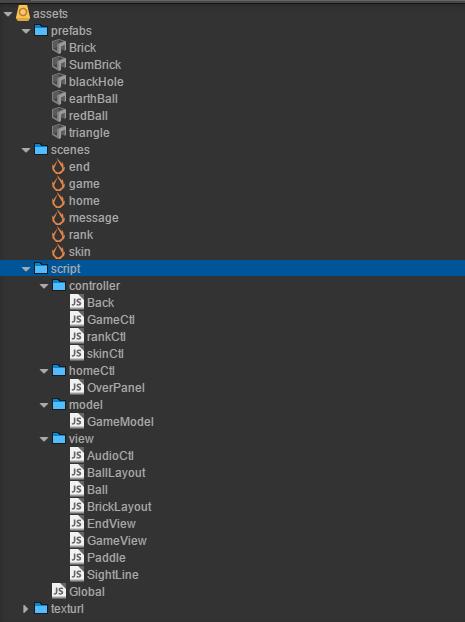


图6 游戏目录结构

# 5 游戏系统实现

## 5.1 资源渲染

游戏中存在大量的资源渲染，砖块200+，小球70+及其他资源50+大量的渲染导致游戏所占内存越来越多，导致游戏越来越卡。为了更好的解决性能问题，游戏中渲染使用对象池提高性能，减少内存占用。节点池是可重用节点对象的集合。我们可以使用New的cc.NodePool（）为某个节点模板创建节点池实例。如果我们有多个需要实例化的预制件，我们应该为每个预制件创建多个节点池实例。每当我们需要实例化节点时，我们首先会尝试从节点池“请求”一个，如果至少有一个节点可用，请求方法get将从池中返回一个节点。然后我们可以使用parentNode.addChild（newNode）将其添加到节点树中。

当不再需要节点时，我们调用节点池实例的put（newNode）方法将节点返回到池而不是销毁它。这个方法也会从父节点中删除节点，所以我们不需要显式地调removeFromParent。将节点返回池是非常重要的，因为只有这样，我们才能将节点的库存保留在我们的池中供将来请求。

假如玩家在游戏中渲染成千上万个砖块，但同时出现的不超过100个，那我们就只需要生成 100个节点大小的对象池，通过使用对象池里的资源，循环调用就可以了。

1. 建立对象池
2. onLoad(){
3. **if**(Global.gameBall=="redBall"){
4. **this**.gameBall=**this**.redBall
5. }**else** **if**(Global.gameBall=="earthBall"){
6. **this**.gameBall=**this**.earthBall
7. }**else**{
8. **this**.gameBall=**this**.redBall
9. }
10. **this**.ballPool = **new** cc.NodePool();
11. let initCount = Global.difficult;
12. **for** (let i = 0; i < initCount; ++i) {
13. let ball = cc.instantiate(**this**.gameBall);
14. **this**.ballPool.put(ball);        }
15. },
16. 使用对象池渲染资源
17. init(initBalls) {
18. **this**.initBalls=initBalls;
19. **this**.node.removeAllChildren();
20. let ballX=window.GameCtl.sightLine.node.x;
21. **for**(let i=0;i<initBalls;i++){
22. let \_this=**this**;
23. setTimeout(() => {
24. let ballNode=**null**;
25. **if**(**this**.ballPool.size()>0){
26. ballNode=**this**.ballPool.get();
27. }**else**{
28. ballNode=cc.instantiate(\_this.gameBall);
29. }
30. ballNode.parent=\_this.node;
31. ballNode.position = cc.v2(ballX,210);
32. ballNode.getComponent(cc.RigidBody).linearVelocity = cc.v2(Math.round(1000/(Math.tan((90-window.GameCtl.sightLine.node.rotation)/57.2956))),1000);
33. }, i\*100);
34. }
35. },

## 5.2 界面设计

1. Home场景



图7 Home场景

Home场景是小游戏的主页，主要进行难度选择、个人记录查看、系统消息提示和皮肤选择该场景主函数OverPanel控制该页面所有按钮。

1. //  难度选择按钮，普通对应绿色，困难对应橙色，地狱对应红色
2. onBtnPutong(event){
3. **this**.initBtn()
4. event.currentTarget.color=**new** cc.Color(128, 243, 123,255)
5. Global.difficult=70
6. },
7. onBtnKunnan(event){
8. **this**.initBtn()
9. event.currentTarget.color=**new** cc.Color(240,181,56, 255)
10. Global.difficult=80
11. },
12. onBtnDiyu(event){
13. **this**.initBtn()
14. event.currentTarget.color=**new** cc.Color(248, 68, 68, 255)
15. Global.difficult=90
16. },
17. //功能选项按钮，进行场景切换。
18. onBtnRank(event){
19. cc.director.loadScene("rank")
20. },
21. onBtnSkin(event){
22. cc.director.loadScene("skin")
23. },
24. onBtnMessage(event){
25. cc.director.loadScene("message")
26. }
27. Game场景

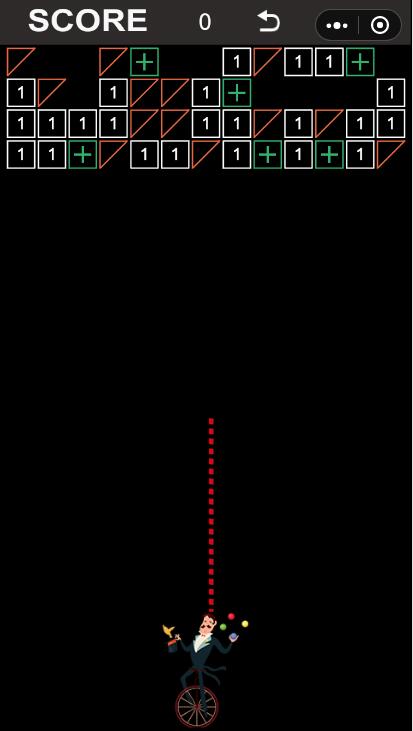


图8 Game场景

Game场景是进行游戏的主要场景。通过控制瞄准线发射小球，击中砖块得到分数。该场景主函数GameCtl控制本场景初始化及碰撞函数调用，即时更新游戏分数，调用小球碰撞砖块函数，游戏胜负判断，实现游戏功能。

1. //游戏初始化函数
2. onLoad() {
3. **this**.physicsManager = cc.director.getPhysicsManager();
4. //获取到数据，并且把该脚本传递给全局变量
5. **this**.gameModel = **new** GameModel();
6. window.GameCtl=**this**;
7. //初始化游戏
8. **this**.init();
9. },
10. init() {
11. //开启物理碰撞，加载函数和游戏布局
12. **this**.physicsManager.enabled = **true**;
13. **this**.gameModel.init();
14. **this**.gameView.init(**this**);
15. **this**.paddle.init();
16. **this**.sightLine.init();
17. },
18. //初始化小球
19. ballRestart(){
20. **this**.ballLayout.init(**this**.gameModel.initBalls);
21. },
22. //游戏结束函数，结束后加载end场景
23. gameOver(){
24. **this**.physicsManager.enabled = **false**;
25. cc.director.loadScene("end")
26. },
27. //游戏暂停函数，关闭物理功能
28. stopGame() {
29. **this**.physicsManager.enabled = **false**;
30. },
31. //砖块移动函数
32. moveBricks(){
33. **this**.brickLayout.moveBricks();
34. },
35. //添加砖块函数
36. addBricks(n){
37. **this**.brickLayout.addBricks(n);
38. },
39. //砖块碰撞函数
40. onBallContactBrick(ballNode, brickNode) {
41. let brickNodeSum=brickNode.getChildByName('brickSum').getComponent(cc.Label)
42. brickNodeSum.string-=1
43. **if**(brickNodeSum.string==0){
44. **this**.brickLayout.brickPool.put(brickNode);
45. Global.score+=1;
46. **this**.gameView.updateScore(Global.score);
47. }
48. **this**.audioCtl.brickPlay();
49. },
50. //砖块碰撞地函数
51. onBallContactGround(ballNode, groundNode) {
52. **this**.ballLayout.ballPool.put(ballNode)
53. **this**.gameModel.groundBall(1,ballNode.position.x)
54. },
55. //砖块碰撞多功能砖块函数
56. onBallContactSumBrick(ballNode,sumBrick){
57. **this**.audioCtl.brickPlay();
58. **this**.brickLayout.sumBrickPool.put(sumBrick);
59. **this**.gameModel.addBall(1);
60. },
61. //砖块碰撞三角形函数
62. onBallContactTriangle(ballNode,triangle){
63. **this**.audioCtl.brickPlay();
64. setTimeout(() => {
65. **this**.brickLayout.trianglePool.put(triangle);
66. }, 2000);
67. },
68. //砖块碰撞黑洞函数
69. onBallContactBlackHole(ballNode,blackHole){
70. ballNode.parent=**null**;
71. **this**.gameModel.initBalls-=1;
72. **this**.gameModel.groundBall(0,ballNode.position.x)
73. **if**(**this**.gameModel.initBalls==0){
74. **this**.gameOver();
75. }
76. setTimeout(() => {
77. **this**.brickLayout.blackHolePool.put(blackHole);
78. }, 10000);
79. }
80. End场景



图9 End场景

End场景是游戏结束场景，加载完后会显示本局玩家分数和重新开始按钮。该场景的主函数是EndView。

1. //加载函数，在游戏结束时显示分数
2. onLoad(){
3. **this**.endScore.string=Global.score;
4. },
5. //返回Home场景
6. backHome(){
7. cc.director.loadScene("home")
8. }
9. Rank场景



图10 Rank场景

Rank场景是排行榜场景，会显示玩家Id和最高分数。该场景主函数是RankCtl.

1. //读取微信本地缓存，获取用户姓名和分数
2. onLoad(){
3. **var** \_userInfo=wx.getStorageSync('\_userInfo')
4. **this**.userName.string=\_userInfo.userName
5. **this**.userScore.string=\_userInfo.userScore
6. **}**
7. Skin场景



图11 Skin场景

Skin场景是皮肤选择场景，选择后小球会上下震动，表示选择该小球。

1. //初始化函数，停止所有动画
2. init(){
3. **for**(let i=0;i<**this**.node.childrenCount;i++){
4. **this**.node.children[i].stopAllActions();
5. }
6. },
7. //选择函数，选择皮肤后会触发动画
8. checkSkin(event){
9. **this**.init()
10. Global.gameBall=event.currentTarget.name;
11. let positionCache=event.currentTarget.position;
12. event.currentTarget.runAction(cc.repeatForever(cc.sequence(cc.moveBy(0.1,0,30),cc.moveBy(0.1,0,-30))))
13. }

## 5.3 碰撞设计

1. 根据碰撞物体不同分为7中不同碰撞函数
2. //小球碰撞时触发回调函数
3. onLoad() {
4. **this**.gameCtl = window.GameCtl;
5. },
6. onBeginContact(contact, self, other) {
7. **switch** (other.tag) {
8. **case** 1: //小球碰到砖块回调函数
9. **this**.gameCtl.onBallContactBrick(self.node, other.node);
10. **break**;
11. **case** 2: //球碰到地面回调函数
12. **this**.gameCtl.onBallContactGround(self.node, other.node);
13. **break**;
14. **case** 5://球碰到多功能砖块回调函数
15. **this**.gameCtl.onBallContactSumBrick(self.node, other.node);
16. **break**;
17. **case** 6://球碰到三角砖块回调函数
18. **this**.gameCtl.onBallContactTriangle(self.node, other.node);
19. **break**
20. **case** 7://球碰到黑洞砖块回调函数
21. **this**.gameCtl.onBallContactBlackHole(self.node, other.node);
22. **break**
23. }
24. },
25. 根据小球碰撞的物体调用相应的回调函数
26. //碰撞砖块触发的函数
27. onBallContactBrick(ballNode, brickNode) {
28. let brickNodeSum=brickNode.getChildByName('brickSum').getComponent(cc.Label)
29. brickNodeSum.string-=1
30. **if**(brickNodeSum.string==0){
31. brickNode.parent = **null**;
32. Global.score+=1;
33. **this**.gameView.updateScore(Global.score);
34. }
35. **this**.audioCtl.brickPlay();
36. },
37. //碰撞到地面触发的函数
38. onBallContactGround(ballNode, groundNode) {
39. ballNode.parent=**null**;
40. **this**.gameModel.groundBall(1,ballNode.position.x)
41. },
42. //碰撞到多功能砖块触发的函数
43. onBallContactSumBrick(ballNode,sumBrick){
44. **this**.audioCtl.brickPlay();
45. sumBrick.parent=**null**;
46. **this**.gameModel.addBall(1);
47. },
48. //碰撞到三角形砖块触发的函数
49. onBallContactTriangle(ballNode,triangle){
50. **this**.audioCtl.brickPlay();
51. setTimeout(() => {
52. triangle.parent=**null**
53. }, 3000);
54. },
55. //碰撞到黑洞触发的函数
56. onBallContactBlackHole(ballNode,blackHole){
57. ballNode.parent=**null**;
58. **this**.gameModel.initBalls-=1;
59. **this**.gameModel.groundBall(0,ballNode.position.x)
60. **if**(**this**.gameModel.initBalls==0){
61. **this**.gameOver();
62. }
63. setTimeout(() => {
64. blackHole.parent=**null**
65. }, 10000);
66. }

# 6 测试与发布

## 6.1 微信开发软件打包

Cocos Creator v1.8 开始，支持 Cocos Creator 游戏一键发布为微信小游戏，下面是详细的发布步骤：

1) 在[微信公众平台](https://mp.weixin.qq.com/debug/wxagame/dev/devtools/download.html" \t "_blank)下载微信开发者工具

2) 在 "Creator 偏好设置" > "原生开发环境" 中设置微信开发者工具路径

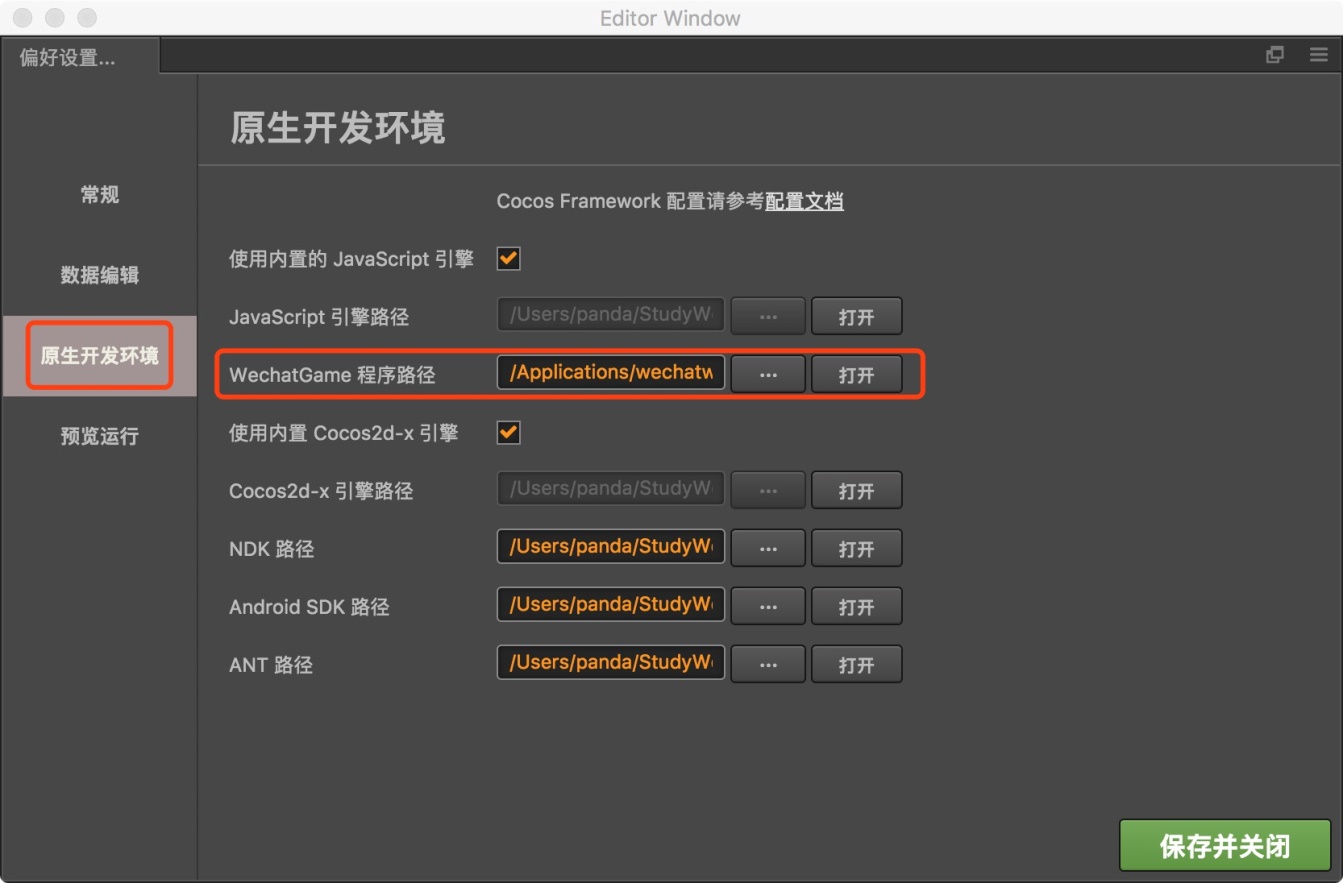


图12 偏好设置

3) 登陆微信公众平台，找到 appid



图13 获取AppID

4) 在构建面板中设置项目名和 AppID

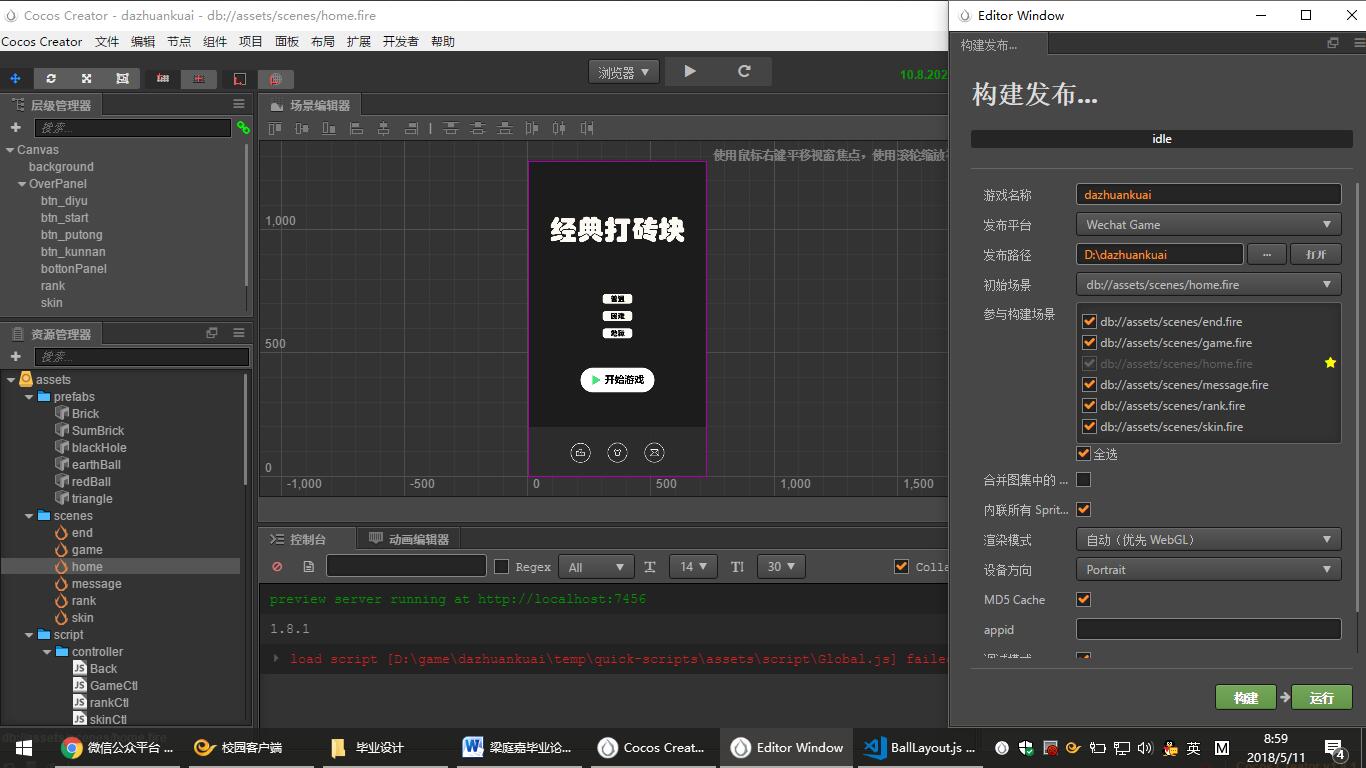


图14 构建面板设置

5) 构建发布

6) 点击 play 打开微信开发者工具

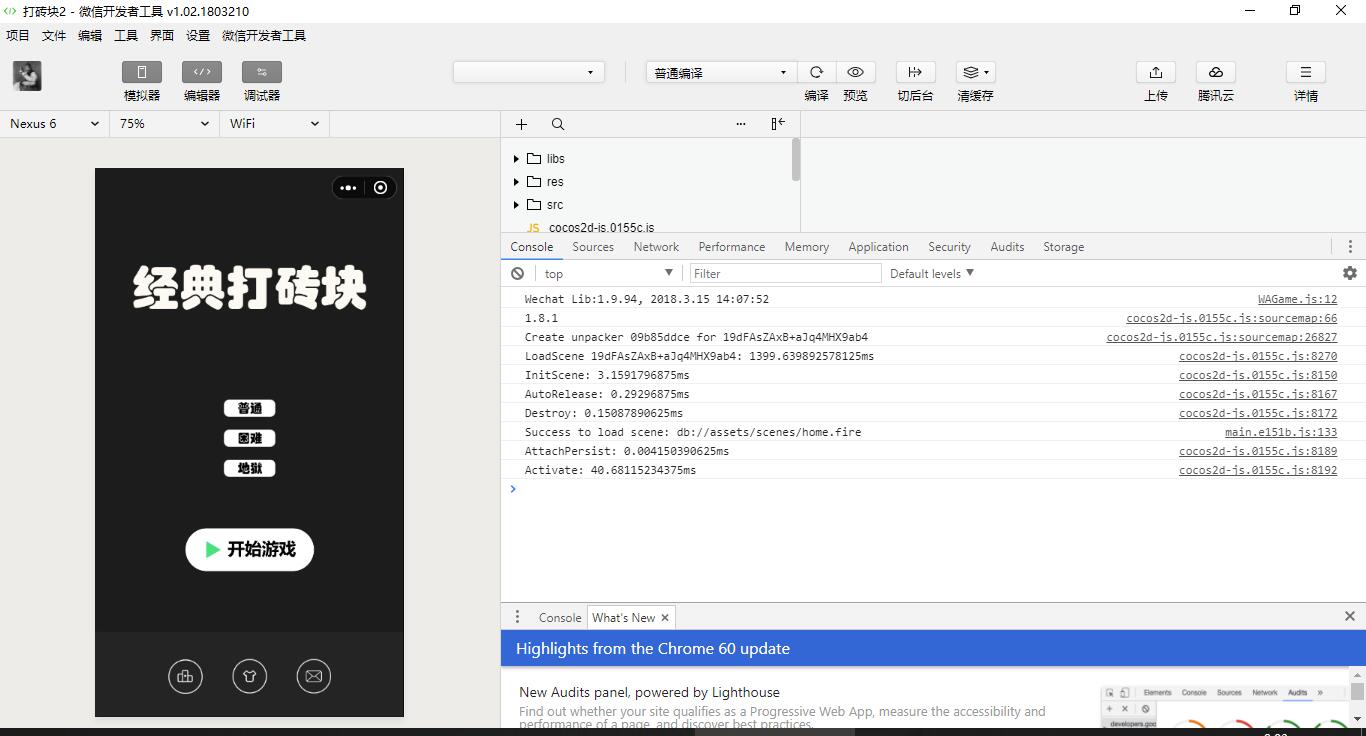


图15 微信开发者工具

7) 预览部署

按照这样的流程，项目的 build 目录下就会生成一个微信小游戏的发布包，其中已经包含了微信小游戏环境的配置文件：game.json 和 project.config.json

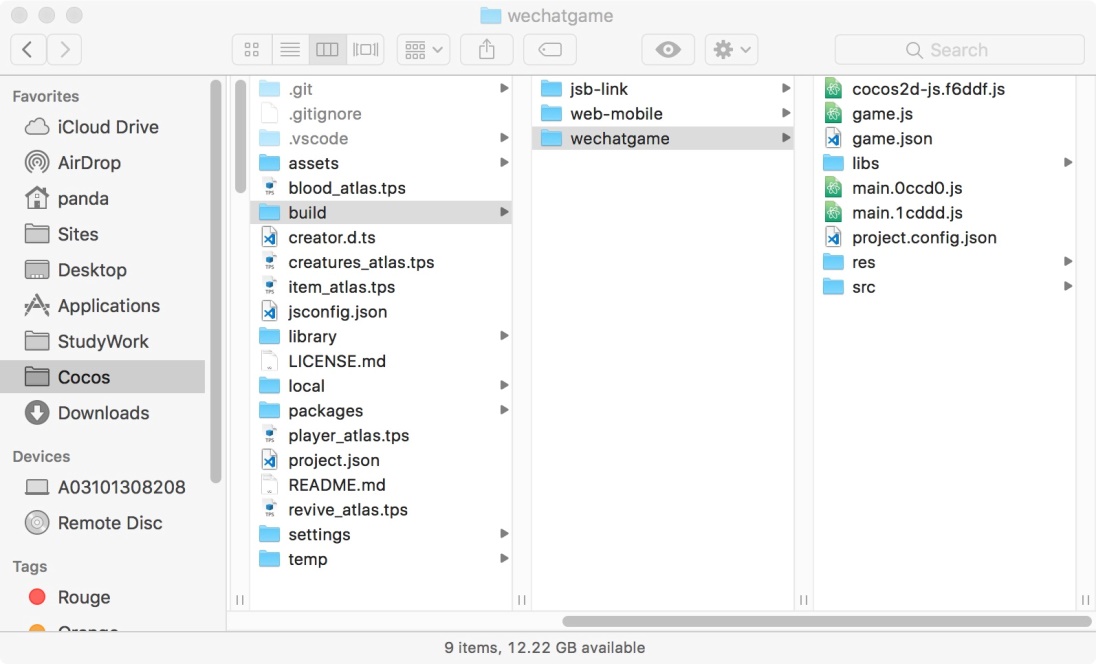


图16 项目目录

接下来就可以把游戏上传到微信小游戏（测试版）。在网页小程序管理中心可以添加测试成员，获取测试权限。

## 6.2 游戏测试

游戏主程序完成之后，为了确保游戏系统能够完整和运行可靠，对系统源程序进行调试和修改时必需的工作。由于在游戏程序的编写过程中已经对各个子程序进行的调试和修正，因此在这不用再对程序做白盒测试，只进行黑盒测试。

按照整体框架的设计，需逐一对系统功能，输入和输出的正确性，性能的可靠性进行测试。

手机端运行微信，开始测试小游戏。

游戏开始界面：

在这里我们可以通过控制小球的发射角度，达到消灭砖块的目的。

游戏结束界面：

当砖块超过人物游戏失败，显示分数和重新开始按钮。

1) 设备A

智能手机：小米6；

CPU：高通骁龙835；

RAM：6GB；

操作系统：Android 8.0；

微信版本：6.6.6；

2) 设备B

智能手机：iphone 8；

Cpu：A11;

RAM:2GB

操作系统：IOS11；

微信版本:6.6.6

3) 设备C

笔记本电脑：华硕C40；

Cpu：i5-3337U;

RAM:8GB

操作系统：win10;

Chrome版本： 61.0.3163.100

图17 性能测试图

测试结果及分析：

如图17所示，通过三个平台的测试，平均帧数都在55以上，可以流畅使用。在win10系统中因为硬件原因，游戏中都能在最高帧数运行。因为ios系统优化问题，帧数也比安卓的高，稳定性也表现得比较好，安卓系统在随着游戏的进行会出现掉帧问题，还会表现出音画不同步的问题，但是不影响游戏的进行。

# 7 设计总结

在Cocos Creator（以下简称“CC”）的学习过程中，由于使用群体比较少，因而学习此的资源也相对较少，缺少好的“学习路径”，以至于在CC的学习道路上，开始之初迷茫与不知所措完全战胜了我对于其学习的热情和动力。好在有CC的社区活跃以及优秀的引擎开发团队，使我在短暂迷茫之后迅速进入状态，根据他们所提供的交流平台，由慢慢摸索到一步步走入学习正轨。

通过使用CC开发微信小游戏，我全面的了解了游戏开发过程，以及游戏性能的重要性；资源的加载和重复使用时至关重要的；资源加载和回调函数决定游戏的流畅性等；一个好游戏的UI设计更能吸引玩家的注意力，游戏中的至关重要的动画效果更能让玩家爱不释手；此外如何搭配和使用颜色更是以后要学习的方向。CC帮助我解决了很多我难以理解的问题，最主要的就是物理碰撞方面，自己写个物理碰撞引擎需要耗费很大的精力，性能也比较差，而CC具备完整的物理解决方案，让我能够全身心投入到游戏逻辑里面。

从开始只有一百多行的基础游戏，到一个月后，一步步的增加功能，提高性能，最终使其具备了一个单机游戏的功能和特点，我学习到了很多东西，但仅仅这样是肯定不够的，接下我准备在服务器上建立数据库，把微信小游戏的用户和数据库的用户数据绑定起来，带来更加个性化的游戏体验，让游戏体验不再单一，开发出能够和其他玩家进行分数比较、互赠礼物等功能，让这个游戏能够成为一个网络游戏。同时在性能上也会继续优化，例如在资源加载使用对象池时，能减少内存损耗。但是目前还不能有效的解决回调函数的卡顿问题，还有碰撞函数和移动砖块的掉帧问题，这也是后期需要努力的目标。

现在，游戏中还存在不少的BUG，但由于时间的原因不能全部解决，但我相信只要继续学习努力，日后定能使其更吸引人、也更完美。

# 鸣 谢

在此，我要特别感谢学校四年来对我的培育，在这短暂的时间里，我学到了很多东西，这些东西也将会是我一生宝贵的财富。也谢谢学院为我提供的良好的学习环境，丰富的学习资源，给予我走入社会的坚实知识基础。

此外，论文能顺利完成，还要特别感谢周老师给予的悉心指导，从选题到完成，是老师一直鼓励与引导；在完成论文的途中，有不少的困难，也是老师不断给予建议让我在完善论文的基础上更开阔了自己的知识储备。为此请允许我向周老师致以诚挚的感谢和敬意!

在毕业设计过程中我也得到了同学们无私的帮助和鼓励，与他们的交流使我不断地调整并清晰了自己的思路，让我更好的完成了毕业设计。

也感谢Cocos社区，给我的毕业设计提供了诸多帮助，让我对CocosCreate更加的了解，写出性能更好的程序。

在此，我向给予过我帮助的每个人 每个团体都致以诚挚的谢意，是你们的鼓励和支持才让我走到今天，长路漫漫，唯记恩于心中

# 参考文献

1. Douglas Crockford. JavaScript语言精粹[M]. 电子工业出版社, 2012.
2. Muzykov K. Learning iPhone game development with Cocos2D 3.0 harness the power of Cocos2D to create your own stunning and engaging games iOS[J]. 2014.
3. Itterheim S. Learn iPhone and iPad cocos2d Game Development[M]. Apress, 2010.
4. 齐兵辉. JavaScript 技术的现状与前景[J]. 科教导刊:电子版, 2017(10):137-137.
5. Addy Osmani. JavaScript设计模式[M]. 人民邮电出版社, 2013.
6. 叶梦轩. 危双丰,张冬梅. 基于HTML5和WebGL的三维点云可视化方法[J]. 工程勘察,2017.
7. 凌建风. Cocos2d-JS游戏开发[M]. 人民邮电出版社, 2016.
8. 黄伊琳, 张征. 基于HTML5的微信游戏设计[J]. 计算机系统应用, 2017, 26(8):261-266.
9. 陈俊建,蒋伟杰,罗康,何勇. 基于HTML5的图结构演示系统[J]. 湖南城市学院学报（自然科学版）,2016
10. 彭小红, 郑琼璇,邓淦辉,林界. 基于HTML5&amp;JavaScript的游戏设计和实现[J]. 现代计算机（专业版）, 2016
11. 丛婷婷. 基于HTML5轻游戏的新媒体推广策略的分析研究[J]. 科技传播,2016.
12. 劳斯三世, 劳斯三世, RouseⅢ,等. 游戏设计[M]. 电子工业出版社, 2003.
13. 瞿苏. 基于HTML5构建离线Web应用的研究与探讨[J]. 佛山科学技术学院学报（自然科学版）,2016.
14. 泽卡斯丁琛. 高性能JavaScript[M]. 电子工业出版社, 2015.
15. 刘剑卓, 郑光龙. Cocos2D-X游戏开发技术精解[M]. 人民邮电出版社, 2015.
16. 黄海玲. HTML5 Web技术下的APP快速开发分析[J]. 无线互联科技,2016.
17. Abdulkarim A, Karlsson L. Game Development Project with Cocos2D-X[J]. Metropolia Ammattikorkeakoulu, 2014.
18. Itterheim S, Löw A. Learn cocos2D 2[J]. 2012.
19. 张筱毅. COCOS2D-X强交互网络游戏架构设计与实践[D]. 西安电子科技大学, 2014.
20. 韦娅彬. 基于HTML5的移动端游戏新闻系统的设计与实现[D].浙江工业大学,2015.
21. Tom Barker. JavaScript性能优化[M]. 机械工业出版社, 2014.
22. 周飞龙. Cocos2d-X引擎中MVC框架的设计与实现[D]. 华中科技大学, 2013.
23. 沈大海. Cocos2d-x手机游戏开发与项目实战详解[M]. 清华大学出版社, 2014.