**学科分类号** 0809



**本科生毕业设计**

**题目（中文）:** 基于Canvas的web版打砖块

游戏的设计与实现

**（英文）:** Design and Implementation of

Arkanoid game based on

Canvas Framework

**学生姓名：** 叶志文

**学 号：** 1500130604

**院 别：** 计算机科学与工程学院

**专 业：** 软件工程

**指导教师：** 彭小宁 教授

**指导教师：** 邓绍伟 助教

**起止日期：** 2018.06-2019.05

**怀化学院本科毕业设计诚信声明**

作者郑重声明：所呈交的本科毕业设计，是在指导老师的指导下，独立进行研究所取得的成果，成果不存在知识产权争议。除文中已经注明引用的内容外，设计不含任何其他个人或集体已经发表或撰写过的成果。对设计的研究做出重要贡献的个人和集体均已在文中以明确的方式标明。本声明的法律结果由作者承担。

本科毕业论文设计作者签名：

年 月 日

**2019 年 5 月 10 日**

目 录

[摘　要 I](#_Toc9020690)

[关键词 I](#_Toc9020691)

[Abstract I](#_Toc9020692)

[Key words I](#_Toc9020693)

[1 前言 1](#_Toc9020694)

[1.1研究背景与意义 1](#_Toc9020695)

[1.1.1 研究背景 1](#_Toc9020696)

[1.1.2 研究意义 1](#_Toc9020697)

[1.2 研究现状 1](#_Toc9020698)

[1.2.1 国内研究现状 1](#_Toc9020699)

[2 需求分析 2](#_Toc9020700)

[2.1 系统功能需求 2](#_Toc9020701)

[2.2 系统非功能需求 2](#_Toc9020702)

[3 系统的设计 3](#_Toc9020703)

[3.1 系统的体系结构 3](#_Toc9020704)

[3.1.1 游戏界面绘制模块 3](#_Toc9020705)

[3.1.2 游戏操作模块 4](#_Toc9020706)

[3.1.3 游戏调试模块 5](#_Toc9020707)

[3.2 关键技术概述 5](#_Toc9020708)

[3.2.1 HTML5简介 5](#_Toc9020709)

[3.2.2 Canvas框架概述 6](#_Toc9020710)

[3.2.3开发环境 6](#_Toc9020711)

[4 系统的实现 7](#_Toc9020712)

[4.1 游戏框架的设计 7](#_Toc9020713)

[4.2 scene场景的实现 7](#_Toc9020714)

[4.3基础类的实现 10](#_Toc9020715)

[4.4按键监听模块的实现 11](#_Toc9020716)

[4.5 关卡编辑模块实现 12](#_Toc9020717)

[4.6 游戏效果实现 14](#_Toc9020718)

[5 系统测试 16](#_Toc9020719)

[5.1 测试计划 16](#_Toc9020720)

[5.1.1 测试进度 16](#_Toc9020721)

[5.1.2 测试方案 16](#_Toc9020722)

[5.1.3 测试目的和原则 16](#_Toc9020723)

[5.1.4 测试环境 16](#_Toc9020724)

[5.2 测试说明 16](#_Toc9020725)

[5.2.1 游戏界面显示测试 16](#_Toc9020726)

[5.2.2 挡板移动控制测试 18](#_Toc9020727)

[5.2.3 小球移动控制测试 19](#_Toc9020728)

[5.3 测试性能及分析 20](#_Toc9020729)

[5.3.1测试结果概述 20](#_Toc9020730)

[5.3.2测试结果评价 20](#_Toc9020731)

[6 总结与展望 21](#_Toc9020732)

[6.1 本论文研究成果 21](#_Toc9020733)

[6.2 进一步工作 21](#_Toc9020734)

[参考文献 21](#_Toc9020735)

[致 谢 22](#_Toc9020736)

基于Canvas的web版打砖块游戏的设计与实现

摘　要

随着时代进步，科技发展，人们的精神娱乐需求日益增长，而游戏无疑是一个很好的选择。在过去，因技术限制，游戏只能在本地客户端运行且无法方便地跨多平台，而这就给开发人员增加了许多无效率工作。经过技术的反复迭代，在现有的HTML的基础上，最终诞生了HTML5。HTML5中的新标签Canvas可以直接通过浏览器来绘制和处理画面，这样的好处是显而易见的 ，不但降低了用户的等待时间，也让游戏开发变得简单方便。最重要的是其免去了下载与安装客户端的繁琐，也因为这些优势。基于H5和Canvas的页游市场涌现了越来越多优秀的作品。本文试着介绍H5和ES6的一些新特性，并通过设计开发打砖块这个游戏，加深我对JSP这门脚本语言以及H5的Canvas标签的理解。

关键词

游戏；HTML5；Canvas标签；JSP；

Design and Implementation of Arkanoid game Based on Canvas Framework

Abstract

With the advancement of the times and the development of science and technology, people's spiritual entertainment needs are growing, and games are undoubtedly a good choice. In the past, due to technical limitations, games could only be run on local clients and could not easily span multiple platforms, which added a lot of inefficient work to developers. After repeated iterations of technology, based on the existing HTML, HTML5 was finally born. The new tag Canvas in HTML5 can draw and process images directly through the browser. The benefits are obvious, which not only reduces the user's waiting time, but also makes game development simple and convenient. The most important thing is that it eliminates the tedious task of downloading and installing the client, and because of these advantages. More and more excellent works have emerged in the page-play market based on H5 and Canvas. This article tries to introduce some of the new features of H5 and ES6, and through the design and development of the brick-breaking game, deepen my understanding of the JSP scripting language and H5's Canvas tag.

Key words

Game；HTML5；Canvas；JSP；

1 前言

1.1研究背景与意义

1.1.1 研究背景

H5游戏越来越受欢迎，出现了越来越多的优秀作品。 与一般的客户端游戏的不同在于，H5游戏并没有客户端的限制，即点即玩。无需反复下载，重复安装，占用的硬件资源和内存都较小，基于这些优势，基于H5的网页游戏渐渐成为跻身为游戏行业的香饽饽。所以我想开发一个基于Canvas的打砖块游戏加深对网页游戏原理的理解[1]。

1.1.2 研究意义

现在手机和PC的性能越来越好，同时随着5G时代即将到来，我们的网络带宽也将会大大增加。网页的一些传统限制不再是限制，过去因为资源加载慢，无法承载过大资源而导致画面粗糙等造成的种种限制，我们无法开发出一些重量级的网页游戏，而随着硬件性能和网络带宽的大大加强，这些将不再是限制。而制作这个项目就是想是通过设计并开发打砖块这个小游戏，一方面可以了解一些图形学的基本知识，另外也可以加深对JSP这门脚本语言的理解以及Canvas的理解，当然开发过程中，也加强自己的编程能力，让我对开发工具的使用（ATOM,Git）更加熟练；同时将之前看过的H5，图形学，及游戏设计的相关知识落实到实处，看看自己学得是否扎实，当然也希望通过这一个项目，让我对编写游戏本身有一个全新的认识，也让自己的代码能力得到增强。

游戏本身是一个艺术品，然而Canvas，H5作为一种技术，我相信，以H5技术为依托，能做出很棒的作品，但是Canvas本身的学习成本高，开发思维和一般意义的前端不同，所以市面上并没有太多类似的项目作为参考。不过我还是认为H5作为一项兴欣技术，网页，简单方便，做游戏有其不能替代的优势。所以，我认为这方面的研究比较有意义。

1.2 研究现状

1.2.1 国内研究现状

JSP语言有很多弊病，但却又有其不可替代的优势，它的成功并不是偶然，JS不用配置环境，不需要编译，可直接在浏览器上运行[2]。

JSP现在发展得非常好（甚至将JS迁移到本地的NODE.JS也发展得很好，脚本语言简单方便）。电子游戏是近几十年出来新兴的娱乐方式，它在当今的娱乐领域占了很大比例，是画面，声音，交互的结合。而网页游戏更是有其鲜明的优势，在拥有H5技术之前，一个有影响的web游戏受到了诸多限制。 HTML5与传统的HTML相比做了许多改进，创新。在我们原先的认知中，HTML是一个笨重的静态语言，需要JSP的配合才能做出稍微好看一些的页面，而用H5做游戏简直就是天方夜谭，但是经过几年的发展与积累，H5游戏的产出越来越丰富。但不得不说，还是因为网页游戏的限制，其作品都不是特别着名的大CP。因为H5作为一种标签语言，JS作为一种脚本语言，对游戏是有一定的影响和限制、比如说帧率，画面的流畅性，H5制作的游戏体验确实和客户端有差距。当然这实际上并不是一个大问题，技术更新换代之后就可以了。所以目前国内的H5游戏总来来看还是处于快速发展状态。

2 需求分析

通过调查互联网上已有的打砖块游戏、对游戏爱好者与开发者进行问卷调查，分析出网站所需要拥有的功能需求，以此来划分游戏的核心功能需求与次要需求[4]。

2.1 系统功能需求

经过分析设计，基于Canvas的Web版打砖块游戏系统的界面应尽量做到简约、美观，合乎逻辑。 具体针对本次设计，可以概括出功能大致如下：

（1）游戏通过键盘A,D操作挡板左右移动、按F弹出小球。

（2）游戏界面绘制：开始场景，主场景，结束场景等。

（3）游戏随关卡等级提升游戏难度增加。

（4）游戏分数的统计。

（5）暂停游戏/继续进行游戏、重新开始游戏。

（6）小球与挡板，边框以及砖块的碰撞，关卡编辑等功能。

（7）游戏运行要求及具体逻辑，能流畅的运行 ，小球与砖块，墙壁，挡板碰撞分别会得分，反弹，且不同的砖块生命值不同，能够手动跳掉下一关。在有限的生命值内完成了全部指定关卡后，游戏胜利。不然，当生命值用完，跳转到游戏结束场景。

2.2 系统非功能需求

 （1）可拓展性

产品在当前状态下，根据产品业务需求的改变，可以增加或减少模块功能的能力，拓展性强的产品应对变化的能力更强[5] 。

 （2）模块划分清晰

每个模块功能相互独立，方便用户的使用。

 （3）网站可靠性良好

网页游戏处理的用户操作十分复杂，因此很可能出现异常的情况。产品的不稳定性是一个隐形杀手，不稳定的产品就像性格不稳定的人，完全没有安全感！要在正常的网站运行流程外，增加异常捕获机制，避免网站卡死或者奔溃的情况。并要合理设计游戏流程和黑盒测试用例，以此来提高网站的健壮性。

 （4）系统健壮性

网站运行时，需要接收的数据要保证正确性，在网站运行中在需要用户输入数据的地方，需要对用户的输入进行判断，防止一些常用的攻击手段，保证用户输入合法，尽量避免异常地出现。

3 系统的设计

3.1 系统的体系结构

系统的体系结构是指，当我们将需求划分并记录之后，将需求的可实现部分细化拆分，将具体的需求抽象为软件的逻辑架构，并划分出模块，这样能让我们将具体化为抽象，将具体化为软件的设计架构[6]。当划分好各种模块之后，才能方便游戏的开发与设计，因为目前的程序还较为简单，所以我决定打砖块游戏的前几个版本可暂且不考虑性能需求，只需将模块划分清晰方便日后扩展即可。以下是我游戏的设计方案。

游戏系统设计如图3.1所示

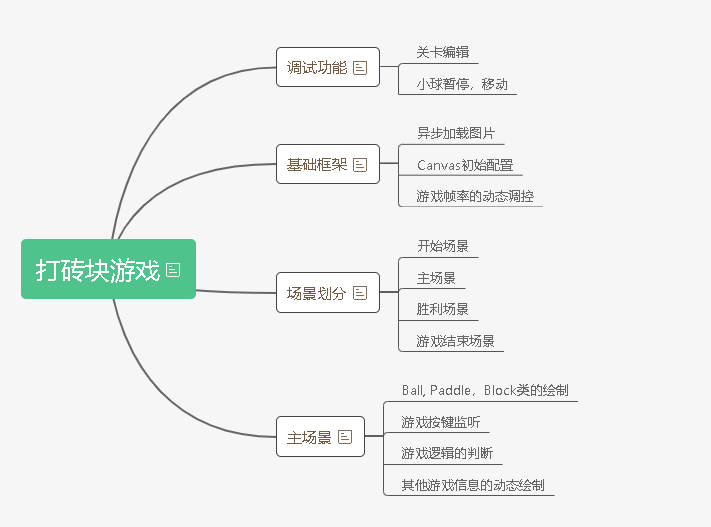


图3.1 游戏系统设计

3.1.1 游戏界面绘制模块

游戏界面绘制模块的作用是负责绘制游戏界面的窗口。

游戏界面是为一400×300的矩形窗口。

游戏界面内容：

1. 游戏基本信息：包括弹球、挡板、砖块等等，分数、游戏Level，随着游戏Level升高而增加的难度等等；

2. 游戏功能：通过移动挡板，操作小球，完成“打砖块操作”，并随着游戏进行，关卡升级，游戏难度会做微调。

3. 游戏各个场景，比如开始场景，编辑场景，胜利场景，失败场景等等，当游戏开始后会切换画面。

游戏绘制组成如图3.2所示

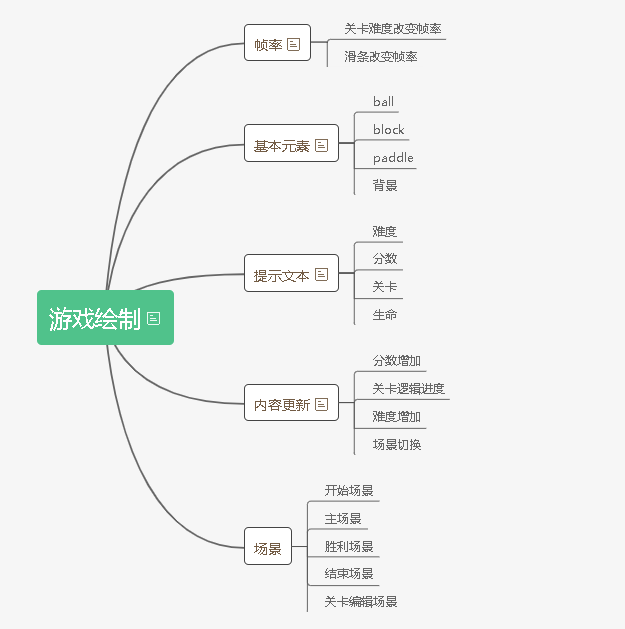


图3.2 游戏绘制组成

3.1.2 游戏操作模块

打砖块操作的基本介绍，挡板的左右调节，小球的碰撞和运动，砖块的消除与碰撞，游戏加分，游戏Level改变等。以下是游戏操作的一些基本键位：

（1）A，D键控制挡板的左右移动；

（2）F键弹出小球，K键开始游戏；

（3）R键：游戏重新开始；

（4）E键：游戏编辑界面；

（5）鼠标拖动：在游戏界面点击小球并拖拽能移动小球；

（6）鼠标左键：游戏编辑界面添加砖块；

（7）鼠标右键：游戏编辑界面去除砖块；

 游戏键鼠操作设计如图3.3所示

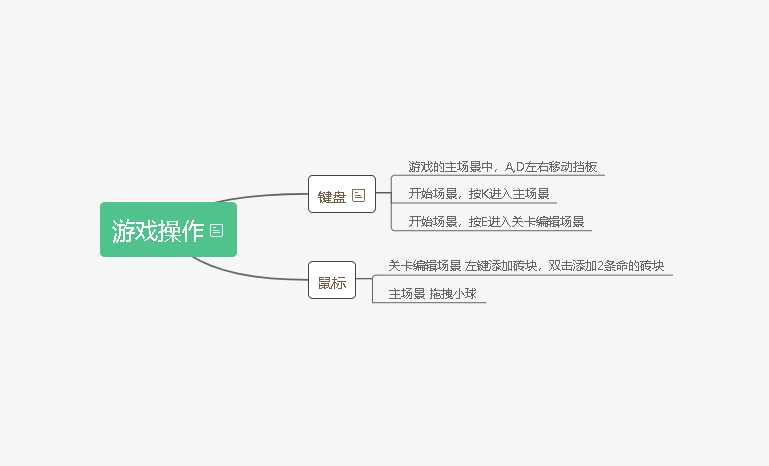


图3.3游戏键鼠操作设计

3.1.3 游戏调试模块

游戏调试模块，包括调节速率，游戏暂停，移动小球等功能。

（1）点击游戏下方滚动条可调节游戏难度；

（2）点击小球可拖拽小球；

（3）P键：暂停游戏；

（4）自定义log窗口；

（5）自定义关卡界面；

 游戏调试模块如图3.4，

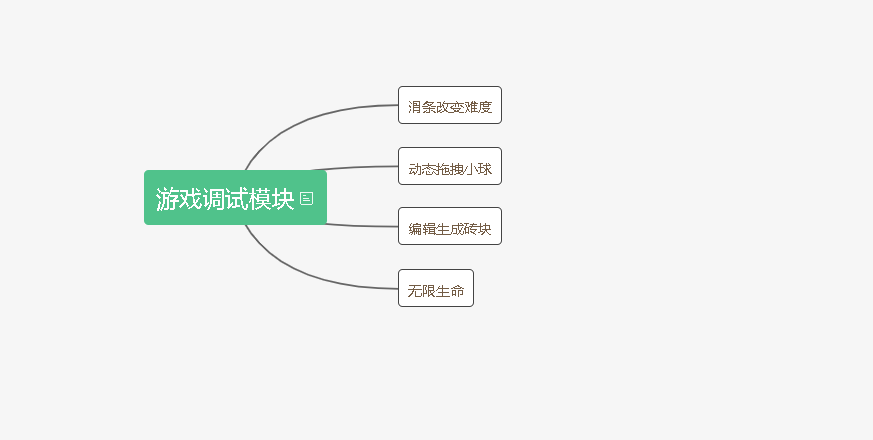


图3.4游戏调试模块

3.2 关键技术概述

3.2.1 HTML5简介

HTML本身是一种标记语言，H5是其第五个版本。

3.2.2 Canvas框架概述

< HTML5>中的<canvas> 元素可以用于画面的绘制，帧率的刷新，图像的展示等等功能， 当然canvas本身只是个标签，你需要用一种脚本语言来操控绘制，我们比较常用的就JS.

如今市面上也有许多基于Canvas的库，可以让我们更加方便地开发游戏[3]。

3.2.3开发环境

Windows10操作系统

Atom编辑器。

Git版本管理。

Chrome浏览器。

4 系统的实现

4.1 游戏框架的设计

Main.js是游戏的入口文件。里面存放了一些全局函数和全局变量。主要是得分，当前关卡，剩余生命等等。guaGame是游戏运行的配置函数。在guaGame的基础上，我们构建出一个个场景，scene根据游戏逻辑来切换不同的场景（开始场景，主场景，胜利场景，关卡编辑场景，以及结束场景）[7]。

游戏架构如图4.1，

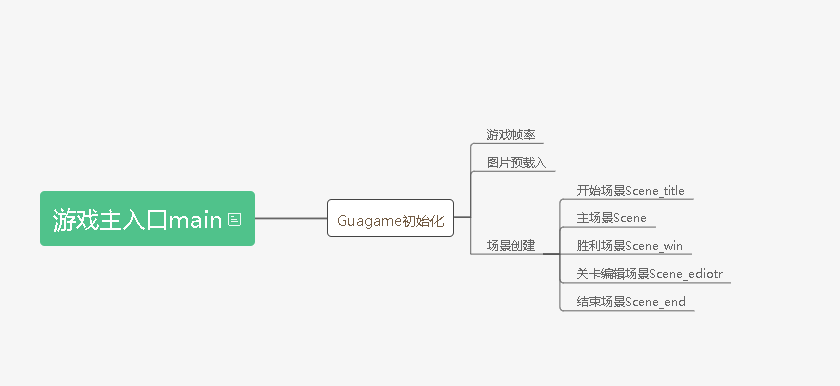


图4.1 游戏架构

关键代码：

var \_\_main = function() {

var images = {

ball: 'img/ball.png',

block: 'img/block.png',

paddle: 'img/paddle.png',

}

var game = GuaGame(30, images, function(g){

var s = SceneTitle(g)

g.runWithScene(s)

})

enableDebugMode(game, true)

}

\_\_main()

4.2 scene场景的实现

游戏初始化后，先从scene\_title开始，一步步依据游戏逻辑而跳转到main scene，win scene，end scene等场景 scene是游戏模块的基本单位。以main scene为例，介绍一下scene的基本组成

Main Scene的组成如图4.2

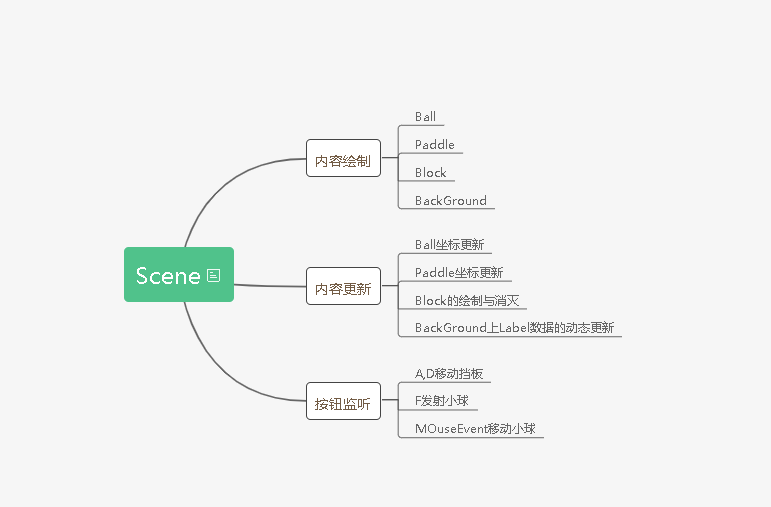


图4.2 Main Scene的组成

关键代码：

var Scene = function(game) {

var s = {

game: game,

}

// 初始化

var paddle = Paddle(game)

var ball = Ball(game)

blocks = loadLevel(game, nowLevel)

alives = blocks.length

game.registerAction('a', function(){

paddle.moveLeft()

})

game.registerAction('d', function(){

paddle.moveRight()

})

game.registerAction('f', function(){

ball.fire()

})

s.update = function() {

if (window.paused) {

return

}

log("nowLevel",nowLevel)

ball.move()

// 判断游戏结束

if (ball.y > paddle.y) {

if (! userLife) {

// 跳转到 游戏结束 的场景

var end = SceneEnd(game)

game.replaceScene(end)

}else{

userLife --

var s = Scene(game)

game.replaceScene(s)

}

}

// 判断相撞

if (paddle.collide(ball)) {

// 这里应该调用一个 ball.反弹() 来实现

ball.反弹()

}

// 判断 ball 和 blocks 相撞

for (var i = 0; i < blocks.length; i++) {

// log('alives', alives)

var block = blocks[i]

if (block.collide(ball)) {

// log('block 相撞')

block.kill()

log('alive', block.alive)

ball.反弹()

// 更新分数

score += 100

if (block.alive == false && alives>0 ) {

log('alives', alives)

alives --

log('alives', alives)

if (alives == 0 && blocks.length<=2) {

blocks = loadLevel(game, blocks.length + 1)

alives = blocks.length

nowLevel = blocks.length

}

if (alives == 0 && blocks.length >2) {

//跳转到you win 场景

setTimeout(function () {

var s = SceneWin(game)

game.replaceScene(s)

},500)

}

}

}

}

}

4.3基础类的实现

主场景的控件主要是ball，paddle，blocks，背景等等。通过var ball = Ball() 这样的形式初始化实例。实例中包含实例的一些基本变量和函数，比如在画面中的初始位置，速度，碰撞函数等等。以Ball为例介绍一下实例的组成。

Ball实体类的设计如图4.3

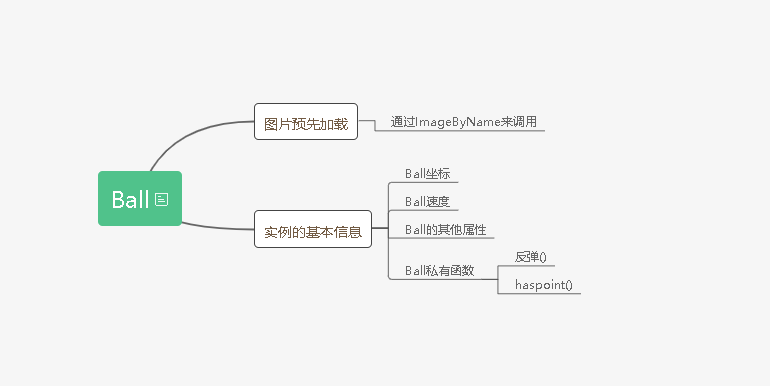


图4.3 Ball实体类

关键代码：

var Ball = function(game) {

var o = game.imageByName('ball')

// var image = imageFromPath('ball.png')

o.x = 100

o.y = 200

o.speedX = 5

o.speedY = 5

o.fired = false

o.fire = function() {

o.fired = true

}

o.move = function() {

if (o.fired) {

// log('move')

if (o.x < 0 || o.x > 400) {

o.speedX = -o.speedX

}

if (o.y < 0 || o.y > 300) {

o.speedY = -o.speedY

}

// move

o.x += o.speedX

o.y += o.speedY

}

}

o.反弹 = function() {

o.speedY \*= -1

}

o.hasPoint = function(x, y) {

var xIn = x >= o.x && x <= o.x + o.w

var yIn = y >= o.y && y <= o.y + o.h

return xIn && yIn

}

return o

}

4.4按键监听模块的实现

首先在游戏的框架类guagame里定义两个对象actions: {}, keydowns: {} 。actions里定义按键对应的callback函数。而keydowns记录按键的按下状态。最后调用全局函数window.addEventListener监听按键，改变两个对象的状态。

按键监听模块如图4.4

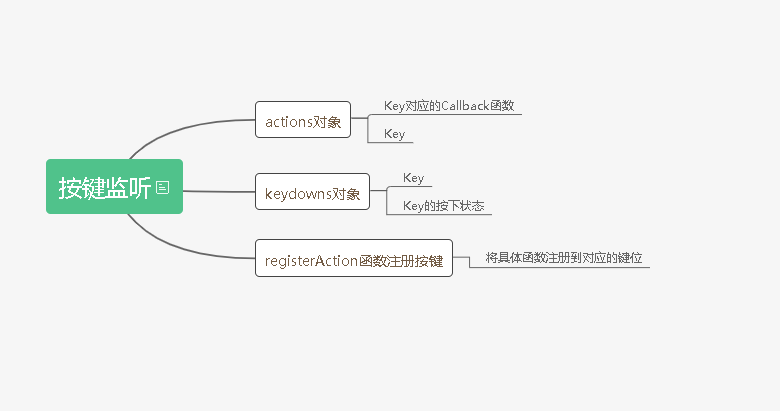


图4.4 按键监听模块

关键代码：

var GuaGame = function(fps, images, runCallback) {

// images 是一个对象, 里面是图片的引用名字和图片路径

// 程序会在所有图片载入成功后才运行

var g = {

scene: null,

actions: {},

keydowns: {},

images: {},

}

var canvas = document.querySelector('#id-canvas')

var context = canvas.getContext('2d')

g.canvas = canvas

g.context = context

// draw

g.drawImage = function(guaImage) {

g.context.drawImage(guaImage.image, guaImage.x, guaImage.y)

}

// events

window.addEventListener('keydown', function(event){

g.keydowns[event.key] = true

})

window.addEventListener('keyup', function(event){

g.keydowns[event.key] = false

})

// update

g.registerAction = function(key, callback) {

g.actions[key] = callback

4.5 关卡编辑模块实现

关卡编辑模块是我在老师的指导下构思设计的模块，其核心的思路在于砖块落点判定，重复点击判定，以及左右按键的监听，最后会将数据存储到localstorage，另外还有值得注意的是，因为画面可能重复绘制，所以要关注生成的block数组的长度有没有增加，而不仅仅是画面上绘制的砖块。

关卡编辑模块如图4.5

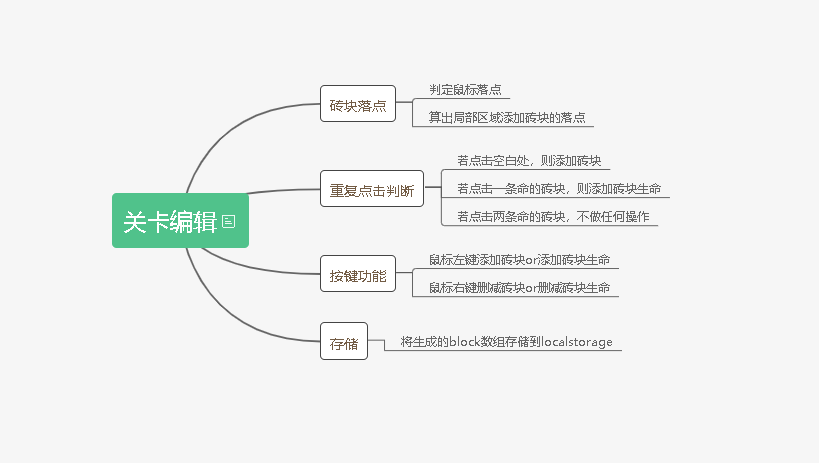


图4.5关卡编辑模块

game.canvas.addEventListener('mousedown',function(event) {

var button = event.button

var x = event.offsetX

var y = event.offsetY

var needx = (parseInt(x / 40) \* 40)

var needy = (parseInt(y / 19) \* 19)

var n = 1

var p = [needx, needy, n]

if(button == 0){

if (editorBlocks.length == 0){

// log("editorBlocks.length == 0")

let block = Block(game, p)

editorBlocks.push(block)

}else {

let haddosomething = false

for (var i = 0; i < editorBlocks.length; i++) {

if (editorBlocks[i].hasPoint(x, y)) {

if (editorBlocks[i].lifes == 1) {

editorBlocks[i].lifes += 1

}

haddosomething = true

break;

}

else if (editorBlocks[i].hasPoint(x, y) == false) {

continue;

}

}

if (haddosomething == false) {

let block = Block(game, p)

editorBlocks.push(block)

haddosomething = true

}

}

log('editorBlocks', editorBlocks)

}

else if(button == 2){

let haddosomething = false

for (var i = 0; i < editorBlocks.length; i++) {

if (editorBlocks[i].hasPoint(x, y)) {

if (editorBlocks[i].lifes == 2) {

editorBlocks[i].lifes -= 1

}else if (editorBlocks[i].lifes == 1) {

editorBlocks.splice(i,1)

}

haddosomething = true

break;

}

else if (editorBlocks[i].hasPoint(x, y) == false) {

continue;

}

}

}

save()

})

return s

4.6 游戏效果实现

经过反复的调试与改进，游戏主场景的画面如图4.6所示



图4.6游戏主场景

5 系统测试

5.1 测试计划

基本类测试：15天、游戏流畅度测试：8天。

5.1.1 测试进度

基本类测试：2019.01.01-2019.01.15

游戏流畅度测试：2019.02.12-2019.02.20

5.1.2 测试方案

游戏最重要的是稳定性与合理性，游戏对程序的稳定性有着极高的要求，起码在游戏的运行中不能出一些致命的bug，所以，测试游戏的稳定性是最基本，也是最重要的环节，我想要我设计的游戏有良好的健壮性和稳定性。这样才能在其上扩展，所以我主要是测试基本类是否方便扩展，另外就只能通过简单的黑盒测试游戏本身是否与设计需求相符。最后便是我加入了一个调试模块，可以让我更方便地测试和调试游戏[8] [9] 。

5.1.3 测试目的和原则

这次测试为了测试出打砖块中的各种BUG、和一些不太符合逻辑的地方，为了更好的进行黑盒测试，我们应该有独立在游戏逻辑之外的游戏调试功能[10]。根据我们定下的目的，这次测试的原则如下：

（1）测试方式的构思，必须得全面覆盖一些不常出现的情况。

（2）游戏测试最好有一些方便调试的功能，不然会难以测试，非常不方便。

（3）测试计划必须严格地单步执行。

（4）对测试结果分析比较，检查和记录。

（5）为了方便程序的后续扩展，应该将相关文档归档备份。

5.1.4 测试环境

系统硬件需求：

较高主频的CPU，512MB以上内存。

系统软件需求：

支持H5Canvas版本的浏览器，推荐用火狐开发者版本，或者Chrome。

5.2 测试说明

5.2.1 游戏界面显示测试

游戏的功能测试主要是对游戏初始界面、小球生命值显示、游戏等级显示、游戏调试等模块进行测试。

1. 游戏初始界面显示

输入：刷新游戏

输出：游戏初始界面

结果：正常

游戏初始界面如图5.1

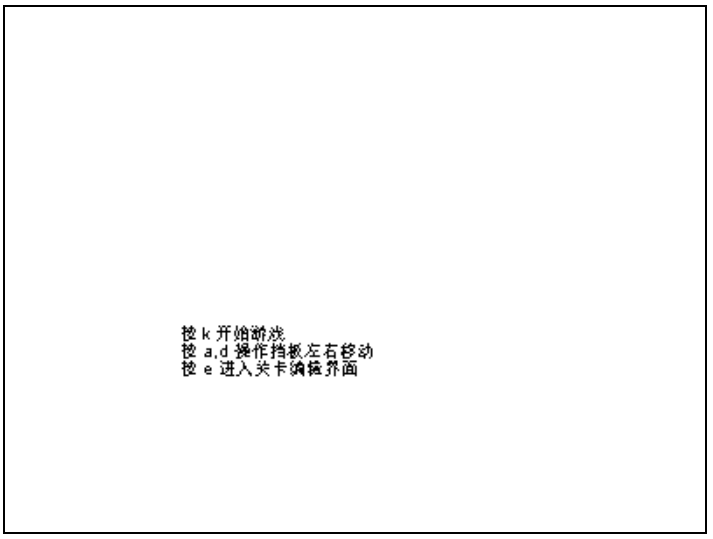


图5.1 游戏初始界面

1. 游戏的主场景界面， 初始状态，小球剩余有三条命，

输入：当小球的y坐标低于挡板的y坐标时

输出：剩余生命值减少1

结果：正常

游戏主场景如图5.2



图5.2 游戏主场景

1. 游戏Level提升

输入：等级显示为Level1当清除了Level1的所有砖块后

输出：游戏界面顶部显示为Level2

结果：正常

Level升级测试如图5.3



图5.3 Level升级测试

1. 根据游戏逻辑的设计

输入：按要求通过第三关后

输出：“You Win,按R键重新开始游戏”

结果：正常

游戏通关界面如图5.4



图5.4 游戏通关界面

5.2.2 挡板移动控制测试

（1）按键控制挡板

输入：分别按下“a”、“d”按键

输出：挡板按照游戏规则移动，挡板移动正常，如图5.5



图5.5挡板测试

5.2.3 小球移动控制测试

（1）小球发射检测

输入：进入游戏后按下“F” 输出：小球开始运动 结果：正常

输出：小球开始运动

结果：正常，

（2）小球碰撞检测

①测试小球与边框、挡板碰撞

输入：开始游戏观察碰撞

输出：小球碰撞后进行反弹

结果：正常

②测试小球与砖块碰撞反弹

输入：开始游戏观察碰撞

输出：小球与砖块碰撞后，砖块会小时并小球会反弹

结果：正常

小球碰撞测试如图5.6



图5.6小球碰撞测试

5.3 测试性能及分析

5.3.1测试结果概述

在反复调试打砖块游戏的过程中，我也遇到了很多难以解决的问题以及各种各样的BUG，比如挡板越界，操作没反应，以及一些让我毫无头绪的BUG,不过在经过反复的测试和修改之后，终于大体完成了这个打砖块游戏，特别是游戏的关卡编辑器，在通宵达旦的调试中，才初步完成。

5.3.2测试结果评价

经测试，系统功能需求的核心功能都已经实现，并且运行良好。

6 总结与展望

6.1 本论文研究成果

在大体完成这次毕业设计之后，我对H5，Canvas有了更加深入的了解,在完成这个打砖块游戏之后，我对Canvas有了更深的理解。之前虽然也学过JSP和H5，但并没有用其做过任何项目，这次小砖块游戏的设计与制作让我对JSP这门脚本语言有了更深的认识，同时也让我发觉自己编程水平的提升。打砖块游戏的核心，在于基础框架的构建，比如游戏几个基本元素类的设计，我想大部分小游戏的设计应该都是差不多的。JSP语言经过反复迭代，新出的ES6已经可以较好的使用面向对象的编程思想，所以也该考虑代码的复用，善用函数式编程，能让我们更好地完成项目。打砖块小游戏的算法本身实现并不困难。不过，在打砖块游戏的设计开发重构与实现过程中，我也确确实实是边学便做，别查边做才完成的。从最初学习JSP语法，到现在以此来制作一个基于Canvas的打砖块游戏，我欣喜地发现了自己的进步，各方面的能力都有或多或少的提高，与此同时，这也让我对基于H5的游戏的开发有了更多的经验和技巧，在以后的项目实践中，希望我能更加从容地应对类似的项目。其实在实习期间，我做过基于Cocos相关的棋牌类的小游戏开发，当时也是基于JS语言，Cocos提供了许多类库，让游戏制作更加方便简单，如今我在Canvas的基础上设计与制作游戏，着实让我觉得对游戏设计有了更深的理解。另外通过这次游戏的设计与开发，从需求分析到现在完成整个项目，把软件工程的相关知识用到一起，形成了一个完整的链路，让我对软件工程有了更加清晰的认识。

6.2 进一步工作

制作这个基于Canvas的打砖块游戏的过程中，从最开始往Canvas画布上添加一个挡板，再到后来将整个游戏丰满起来。我体会到了一个项目，从无到有，从简单到复杂，代码从刚开始面向过程的逻辑流水，到最后不得不采用面向对象的方式重构封装。我真真切切地体会到了我代码能力的提升。当然，这简简单单的一个打砖块系统还有许多问题，比如砖块碰撞的情况考虑得不够全面，比如关卡编辑后的关卡无法合理地加载。再比如场景的抽象不够细化导致程序的扩展性不太好。所以接下来的工作，我打算将我想加的功能继续抽离细化，把一些不够好的算法重新编写，把游戏的界面做得更加美观。

参考文献

[1] Peter Lubbers[美]，Brian Albers [美]，Frank Salim[美][著]，李杰，柳靖，刘淼[译].HTML5高级序设计[M]. 北京，人民邮电出版社，2011.

[2] Christian Heilmann[德][著].牛海彬[译]. 深入浅出JavaScript[M]. 北京,人民邮电出版社,2008.

[3] Rob Hawkes[英][著]，周广新，曾少宁，盛海艳[译].HTML5 Canvas基础教程[M]. 北京，人民邮电出版社, 2012.1.

[4] 百度百科.打砖块.http://baike.baidu.com/view/28513.htm.

[5] 罗彪,张宏涛.基于HTML5的移动互联网应用发展趋势[J].信息与电脑(理论版),2018(10):129-131.

[6] 和歆雨.基于HTML5的网页游戏的设计与开发[J].中国战略新兴产业,2018(04):94-95.

[7] 史济民，顾春华，郑红.软件工程:原理、方法与应用[M].北京:高等教育出版社,2009:33-68.

[8] 网页设计教程/张月玲，韩毅娜主编.——北京：清华大学出版社;北京交通大学出版社,2006.8.

[9] Web程序设计——JSP/孙延鹏,吕晓鹏编著.——北京:人民邮电出版社,2008.5.

[10] IT项目管理:从理论到实践/王如龙主编.——北京:清华大学出版社,2008.9.

致 谢

在毕业论文完成之际，我要特别感谢邓绍伟导师的热情关怀和悉心指导。撰写毕业论文以及修改项目的过程中，邓绍伟导师倾注了大量的心血和汗水，无论是在毕业论文选题、构思和资料的收集方面，还是在毕业论文的研究方法以及成文定稿方面，笔者都得到了邓绍伟导师悉心细致的教诲和热情无私的帮助。本次毕业设计的每一个地方都凝聚着邓老师的心血，没有邓老师的悉心指导和帮助，就没有本论文的顺利完成和我的成长。

衷心感谢邓老师所给予的无私帮助，他渊博的学识、敏锐而超前的学术意识、民主而严谨的治学作风将使我受益终生，并激励我在今后的工作、学习和生活中，时刻保持勤奋刻苦的工作态度和求实创新的敬业精神。

最后，向在百忙之中抽出时间对本文进行评审并提出宝贵意见的各位老师表示衷心的感谢！