**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：** 邓邵伟 讲师

**起止日期：** 2017.06-2018.05

**2018 年 5 月 10 日**

**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

目 录

[摘　要 I](#_Toc8598690)

[关键词 I](#_Toc8598691)

[Abstract I](#_Toc8598692)

[Key words I](#_Toc8598693)

[1 前言 1](#_Toc8598694)

[1.1研究背景与意义 1](#_Toc8598695)

[1.1.1 研究背景 1](#_Toc8598696)

[1.1.2 研究意义 1](#_Toc8598697)

[1.2 研究现状 1](#_Toc8598698)

[1.2.1 国内研究现状 1](#_Toc8598699)

[2 关键技术概述 1](#_Toc8598700)

[2.1 HTML5简介 1](#_Toc8598701)

[2.2 Canvas框架概述 2](#_Toc8598702)

[2.3 开发环境 2](#_Toc8598703)

[3 需求分析 2](#_Toc8598704)

[3.1 系统功能需求 2](#_Toc8598705)

[3.2 系统非功能需求 3](#_Toc8598706)

[4 系统的设计 3](#_Toc8598707)

[4.1 系统的体系结构 3](#_Toc8598708)

[4.1.1 游戏界面绘制模块 3](#_Toc8598709)

[4.1.2 游戏操作模块 4](#_Toc8598710)

[4.1.3 游戏调试模块 5](#_Toc8598711)

[5 系统的实现 6](#_Toc8598712)

[5.1 游戏框架的设计 6](#_Toc8598713)

[5.2 scene场景的实现 8](#_Toc8598714)

[5.3主场景控件的实现 11](#_Toc8598715)

[5.4 按键监听模块实现 13](#_Toc8598716)

[6.1 测试计划 15](#_Toc8598717)

[6.1.1 测试进度 16](#_Toc8598718)

[6.1.2 测试方案 16](#_Toc8598719)

[6.1.3 测试目的和原则 16](#_Toc8598720)

[6.1.4 测试环境 16](#_Toc8598721)

[6.2 测试说明 16](#_Toc8598722)

[6.2.1 游戏界面显示测试 16](#_Toc8598723)

[6.2.2 挡板移动控制 18](#_Toc8598724)

[6.2.3 小球移动控制 19](#_Toc8598725)

[7 测试及性能分析 19](#_Toc8598726)

[7.1 测试结果概述 19](#_Toc8598727)

[7.2 测试结果评价 20](#_Toc8598728)

[8 总结与展望 20](#_Toc8598729)

[8.1 本论文研究成果 20](#_Toc8598730)

[参考文献 20](#_Toc8598731)

[致 谢 21](#_Toc8598732)

基于Canvas的web版打砖块游戏的设计与实现

摘　要

近几年来社会不断地快速发展，也让人们在工作中产生了很多这样那样的压力，随着科技的进步，人们更希望通过网络、通过游戏来释放自己的压力，以往的技术都只是让游戏在笔记本或者移动产品上单独运行，并不可以跨平台操作，这也增加了开发者的工作量，因此经过人们不懈的努力，终于在原有基础上对html进行了改革，这就是HTML5。  HTML5具有很多新的特性，最主要的还是它具有跨平台兼容性，不仅可以在电脑上运行，还可以在移动终端运行，不仅如此HTML5中的Canvas元素可以使浏览器直接创建并处理图像，减轻了开发人员的负担，而且使界面更加美观，具有很好的用户体验。减少了用户刷新页面的时间，得到了更多人的认可,因此基于HTML5所制作的游戏也就成了很多人的研究课题。  本文通过介绍HTML5的一些特性，并通过当下比较流行的打砖块游戏的研发过程分析来使HTML5的内容更加形象，并对HTML5的前景做了一些分析。

关键词

科技社区；HTML5；Canvas；JSP；

Design and Implementation of Arkanoid game Based on Canvas Framework

Abstract

In recent years, the rapid and rapid development of society has also caused people to exert a lot of pressures in their work. With the advancement of technology, people are more willing to release their pressure through the Internet and through games. The past technology only makes the game. Running separately on a notebook or mobile product, it is not possible to operate across platforms. This also increases the workload of the developer. Therefore, after unremitting efforts, the html has been reformed on the basis of the original. This is HTML5. HTML5 has many new features, the most important of which is cross-platform compatibility, not only can run on the computer, but also can be run on the mobile terminal, not only the Canvas element in HTML5 allows the browser to directly create and process images, reducing The burden on the developer, and the interface is more beautiful, with a good user experience. It has reduced the time for users to refresh the page and has been recognized by more people. Therefore, the game based on HTML5 has become a research topic for many people. This article introduces some of the features of HTML5.

Key words

Technology Community；HTML5；Canvas；JSP；

1 前言

1.1研究背景与意义

1.1.1 研究背景

当今社会，随着经济的不断发展，人们对精神领域的追求也在不断加强，人们不再满足于只在笨拙的台式机上进行工作，而是越来越倾向于使用移动电子产品，并且人们对于网页的设计，刷新的速度都有了更高的要求，在这样的一个时代要求下，以前的网页制作语言HTML已经不能满足人们的要求，需要在移动、跨平台等新特性下研发更新html语言。[1]

先前的几个html版本，主要针对的是静态的文本网页，用于显示文档和共享一些数据，后来动态网站和应用程序的出现，使得网页更具有灵动性，但是这些都是基于第三方的插件或者Adobe Flash来完成的，这些插件可以使网站更加丰富，可以提供人机交互的功能。随着网店和网页游戏的不断兴起，Web早就不能只满足于静态的文档，但是Web本身又不具备对视频，音频，图像处理的开发功能，想要在网页中添加这些就必须借助第三方插件，另外有时还需要其他的技术来支持HTML，让它支持多种媒体，这就加重了开发人员的工作量，不仅如此，人们再刷新网页时由于内容过多，耗费时间较长，用户体验差。因此通过不懈的努力，HTML5终于应运而生。

1.1.2 研究意义

此项目的目的在于通过完成游戏项目，加深对JSP的语法、算法等多方面的学习，深入了解图形用户界面的设计，并且熟练的使用编程软件，将书本上的理论知识应用到实践当中，来拿这款打砖块小游戏来验证下自己的学习的理论知识是否扎实。希望通过这一次的毕业设计，让我在程序的设计和算法的设计上有进一步的认识，然后获得一定提高。通过设计增加了我们的实际动手能力，对于我们以后工作能力和信心的培养有重要意义，进而与社会所需的人才相接轨。更好的完善自己！

游戏是一种艺术，而JSP是一种技术，集合不同的艺术门类，并把艺术与技术相结合，完成游戏的开发，游戏可以让人在疲倦或休闲之时获得轻松而又愉快的享受。所以，开发出大家都比较喜欢的，高品质的，简单容易好玩的休闲游戏，将会受到人们的普遍欢迎。让人们在工作学习之余，享受游戏的快乐，也是一款游戏真正成功的意义。

1.2 研究现状

1.2.1 国内研究现状

HTML5对旧的Web网页制作技术做了大量的改进、创新。它增加了很多新的元素，将一些模块动态化，另外HTML5的核心目标是增加了比如<audio>、<video>等新的媒体元素，这样就可以不用第三方插件来播放多媒体，操作更加简单方便。不仅如此新增加的Canvas元素，可以使浏览器直接创建并处理图形，使得网页布局更加美观大方，也方便了开发人员，代码更加简洁。HTML5不仅是对互联网的改革，它更增加跨平台性，兼容性更好，它将互联网带向了一个更加成熟的平台，使用HTML5开发的程序，不仅可以在电脑上运行，同样人们也可以在移动产品上使用程序，人们获取信息更加方便，快捷。大大的满足了人们对了解各种信息的迫切要求，无论是查看文档信息，还是加载动画、视频，都增加了页面的刷新速度，具有良好的用户体验。  HTML5新增加的这些功能，也为网页游戏提供了开发上的便利条件，不再需要第三方插件来支持游戏的音效，动画，给网页游戏开创了一个新的时代。在国内外也掀起了一股学习开发HTML5游戏的热潮，之前的网页游戏开发成本高，即需要服务器端有需要客户端     的支持，而现在HTML5可以跨平台操作，可以在更多的移动产品上进行操作，更能满足人们对于便捷的要求。同时开发者只需要在原有基础上了解一下增加的新的HTML5和JS语言，当然对于游戏少不了的还有各种引擎，这些会在后面的文章中有所涉及。

2 关键技术概述

2.1 HTML5简介

HTML5是万维网html语言的第五个版本，目前比较流行的版本是HTML4.01，虽然相较先前的版本有了很大的改善，但是随着移动应用的发展，以前的那些版本已经远远达不到人们对于科学技术的要求，Html需要在包括语言方面，语法方面和UI、API等各方面的更新，因此就产生了现在这个新的版本。  在HTML5之前人们更注重用html来制作一些静态的网页，那时候人们通过网络的交互还不是那么发达，从网络上了解外界信息的心情也不是那么迫切，博客，网店也没有兴起。但是近几年随着科技和经济的不断发展，人们更倾向于从网上获取信息，在网上购买商品，节省了人们很多的时间与精力。  因此再2010年HTML5正式被大家所认可，实现了Web领域近十几年来质的飞跃，将Web推向了一个更高、更成熟、更稳定的平台。[3]  在此基础上HTML5将音频，视频，动画等内容融为一体，更是增加了很多新的元素标签，去除了一些繁琐，冗长的标签，将一些相似标签融为一体，比如增加了表单验证标签，节省了表单输入时的麻烦，这些新的标签是网页结构更加紧凑合理。比如hgroup可以将标题进行分组，合成一个个的整体，使开发页面也更加整洁；还有新添加的导航标签nav，有利于搜索引擎的结果整理。HTML5相较其他版本的以大亮点就是将audio和video引入到标签中，这样网页播放音频和视频时就不必借助第三方插件，本身就可以做到了。  当然对于游戏来说最主要的还是图形动画的界面要美观，这样才可以吸引更多的人来玩，对于这方面不得不提的就是Canvas标签了，它与其他元素不同，不像audio等元素直接将现有元素插到网页中，而是可以独立的处理或创建2D图形，不仅如此，还可以通过JavaScript语言来控制Canvas的图形来响应与用户交互的动态图形与动画。

2.2 Canvas框架概述

<canvas> 是 HTML5 新增的元素，可用于通过使用JavaScript中的脚本来绘制图形。例如，它可以用于绘制图形，制作照片，创建动画，甚至可以进行实时视频处理或渲染。

Mozilla 程序从 Gecko 1.8 (Firefox 1.5) 开始支持 <canvas>。它首先是由 Apple 引入的，用于 OS X Dashboard 和 Safari。Internet Explorer 从IE9开始支持<canvas> ，更旧版本的IE可以引入 Google 的 Explorer Canvas 项目中的脚本来获得<canvas>支持。Chrome和Opera 9+ 也支持 <canvas>。

WebGL也使用<canvas>元素在网页上绘制硬件加速的3D图形。

2.3 开发环境

Atom 是专门为程序员推出的一个跨平台文本编辑器。具有简洁和直观的图形用户界面，并有很多有趣的特点：支持 CSS，HTML，JavaScript 等网页编程语言。它支持宏，自动完成分屏功能，集成了文件管理器。

git（/ɡɪt/[4]，关于这个音频文件 音频 帮助·信息）是一个分布式版本控制软件，最初由林纳斯·托瓦兹创作，于2005年以GPL发布。最初目的是为更好地管理Linux内核开发而设计。应注意的是，这与GNU Interactive Tools[5]（一个类似Norton Commander界面的文件管理器）有所不同。

git最初的开发动力来自于BitKeeper和Monotone[6][7]。git最初只是作为一个可以被其他前端（比如Cogito或Stgit[8]）包装的后端而开发的，但后来git内核已经成熟到可以独立地用作版本控制[9]。很多著名的软件都使用git进行版本控制[10]，其中包括Linux内核、X.Org服务器和OLPC内核等项目的开发流程[11]。

3 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，确定了游戏的功能性需求与非功能性需求。

3.1 系统功能需求

通过分析调查结果，打砖块游戏系统界面的布局应尽可能的做到合理、美观。 具体针对本次设计，可以概括出功能大致如下：

（1）游戏基本功能（方块的操作）：挡板左右移动、发射小球，通过键盘操作。

（2）游戏界面：开始画面，得分，关卡，结束画面等。

（3）游戏随关卡等级提升使难度有所增加。

（4）游戏分数的统计。

（5）暂停游戏/继续游戏、退出游戏。

（6）砖块消除，挡板与小球碰撞，等功能。

（7）游戏能流畅的运行 ，通过小球碰撞砖块，消灭砖块同时能增加得分，能够手动跳掉下一关。通过最后一关后游戏胜利，如果挡板没有在小球下落时挡住小球，游戏结束。

3.2 系统非功能需求

 （1）界面布局简洁美观

良好的界面布局，简洁美观的网站显示效果能使用户更愿意使用此网站。

 （2）模块功能清晰

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

网站处理的用户操作十分复杂，因此很可能出现异常的情况。要在正常的网站运行流程外，增加异常捕获机制，避免网站出现奔溃的情况。要合理设计业务流程和测试用例，提高网站的可靠性。

 （4）系统数据正确

网站运行时，需要接收的数据要保证正确，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，保证用户输入合法，尽量避免异常地出现。

4 系统的设计

4.1 系统的体系结构

系统设计是经过系统分析，然后由抽象至具体的过程，本阶段的主要目标是把在分析阶段中的抽象概念化作能够实现，使之成为有具体功能的模块，提供支持给系统代码设计。在系统设计的阶段种，应该尽可能全面的思考全系统的功能与需求，由于情况的不同，有时还要考虑到系统性能。打砖块游戏的画面简单，功能也不怎么复杂，系统资源占用很少，所以，在这里不过于考虑性能需求，下面对游戏的详细设计过程做一个说明。

4.1.1 游戏界面绘制模块

界面绘制模块的作用是负责游戏界面的显示。

游戏界面为一矩形窗口。 游戏界面内容：

1. 游戏信息：小球生命值显示、游戏得分、游戏等级，随着游戏不断更新；

2.游戏功能：砖块、小球、挡板；

3. 游戏开始场景，当游戏开始后会隐藏。

游戏界面如图4.1，



图4.1 游戏界面

4.1.2 游戏操作模块

游戏操作模块，包括砖块的生成，挡板的生成，挡板的移动，小球的运动，消除砖块，游戏得分，游戏等级等。要特别注意小球与挡板移动后会不会超出操作区域等问题。  游戏过程中，玩家是通过键盘或鼠标控制游戏。玩家输入操作命令，程序接收并做出响应。下面列出鼠标、键盘的按键和对应的功能：

（1）A，D键控制挡板的左右移动；

（2）F键弹出小球，K箭开始游戏；

（3）R键：游戏重新开始；

 游戏操作如图4.2，

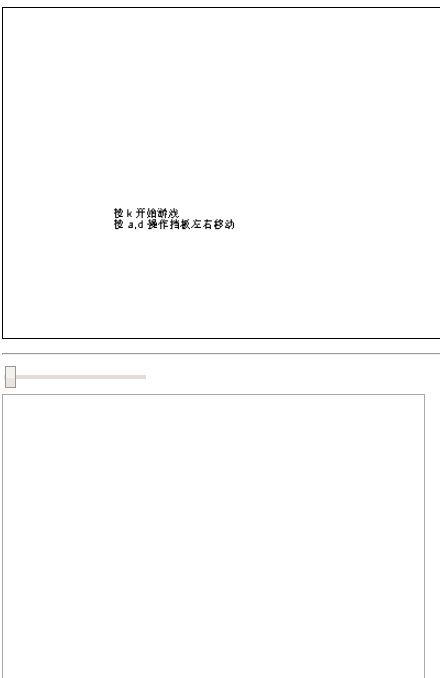


图4.2 游戏操作

4.1.3 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调节游戏速率

（2）点击小球可拖拽小球；

（3）P键：暂停游戏；

（4）自定义log窗口；

 游戏调试如图4.3，



图4.3 游戏调试

5 系统的实现

5.1 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。guaGmae是游戏运行的配置函数

是游戏的主入口。在guaGmae的基础上，我们在一个个scene来切分游戏。

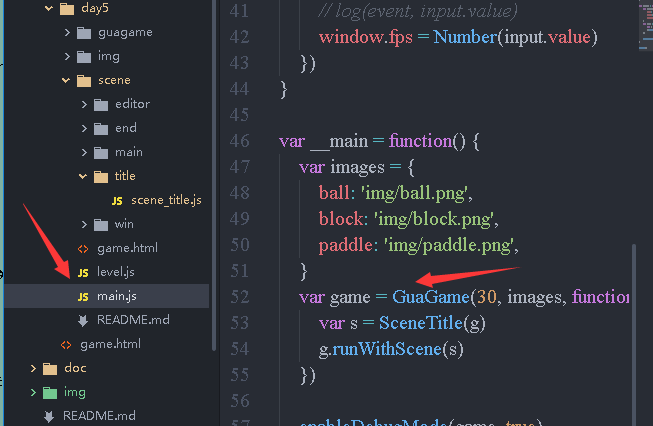


图5.1 main.js

关键代码：

var loadLevel = function (game,n) {

var n = n - 1

var level = levels[n]

var blocks = []

for (let i = 0; i < level.length; i++) {

var p = level[i]

var b = Block(game, p)

blocks.push(b)

// log('blocks.n',blocks.n)

}

return blocks

}

score = 0

userLife = 3

nowLevel = null || 1

var blocks = []

var enableDebugMode = function (game, enable) {

if (!enable) {

return

}

window.paused = false

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

let k = event.code

// log(k)

if (k == 'KeyP') {

//debug 暂停

window.paused = !window.paused

} else if ('Digit1Digit2Digit3Digit4Digit5Digit6Digit7'.includes(k)) {

//debug 载入关卡

// log('k[5]',k[5])

blocks = loadLevel(game, Number(k[5]))

alives = blocks.length

// log('blocks', blocks)

}

})

// 控制速度

document.querySelector('#id-input-speed').addEventListener('input', function(event) {

var input = event.target

// log(event, input.value)

window.fps = Number(input.value)

})

}

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()

5.2 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进入main scene，win scene，end scene等等的场景 scene是游戏模块的基本单位。

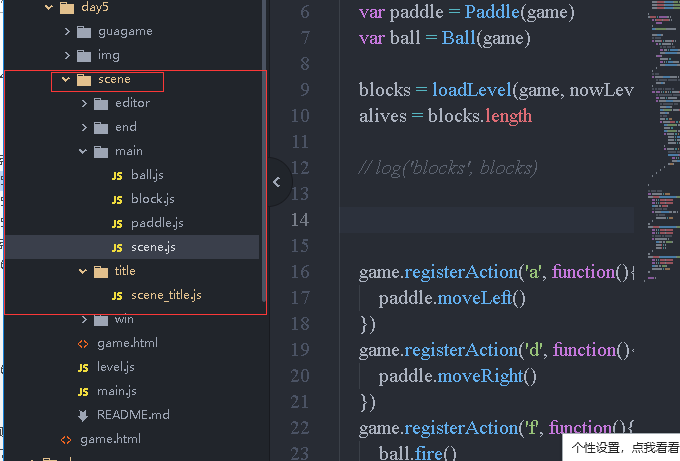


图5.2 scene

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

// log('blocks', blocks)

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

paddle.moveLeft()

})

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

paddle.moveRight()

})

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

ball.fire()

})

s.draw = function() {

// draw 背景

game.context.fillStyle = "#554"

game.context.fillRect(0, 0, 400, 300)

// draw

game.drawImage(paddle)

game.drawImage(ball)

// draw blocks

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

var block = blocks[i]

if (block.alive) {

game.drawImage(block)

}

}

// draw labels

game.context.fillStyle = "white"

game.context.fillText('分数: ' + score, 5, 280)

game.context.fillText('剩余生命: ' + userLife, 5, 290)

game.context.fillText('level: ' + blocks.length, 180, 20)

}

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)

}

}

}

}9

}

// mouse event

var enableDrag = false

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

var x = event.offsetX

var y = event.offsetY

log(x, y, event)

// 检查是否点中了 ball

if (ball.hasPoint(x, y)) {

// 设置拖拽状态

enableDrag = true

}

})

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

var x = event.offsetX

var y = event.offsetY

// log(x, y, 'move')

if (enableDrag) {

log(x, y, 'drag')

ball.x = x

ball.y = y

}

})

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

var x = event.offsetX

var y = event.offsetY

log(x, y, 'up')

enableDrag = false

})

return s

}

5.3主场景控件的实现

主场景的控件主要是ball，paddle，blocks等等。分别通过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。

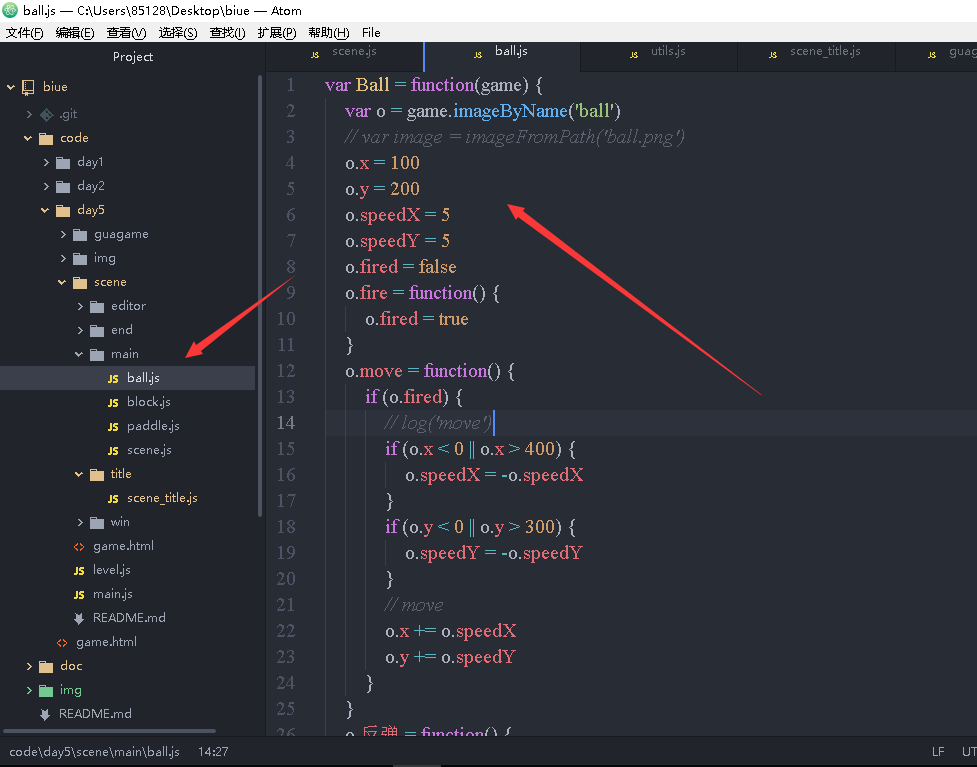


图5.3 ball.js

关键代码：

@AuthPassport(isEditor = true)

@ResponseBody

@RequestMapping("/sendNews")

public Result sendNews(String title, String content, HttpServletRequest request) {

User user = (User) request.getSession().getAttribute("User");

return newsService.sendNews(title, content, user.getId().longValue());

}

@Override

public Result sendNews(String title, String content, long userId) {

try {

NewsWithBLOBs news = new NewsWithBLOBs(title, content, userId);

String str = Utils.delHTMLTag(news.getNewsContent()).replace(" ", "").replace("　", "").trim();

news.setNewsIntro(str.length() > 150 ? str.substring(0, 150) : str);

if(newsMapper.insert(news) > 0) {

return new Result(ResultInfo.SUCCESS);

}

} catch (Exception e) {

e.printStackTrace();

}

return new Result(ResultInfo.ERROR);

}

5.4 按键监听模块实现

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。

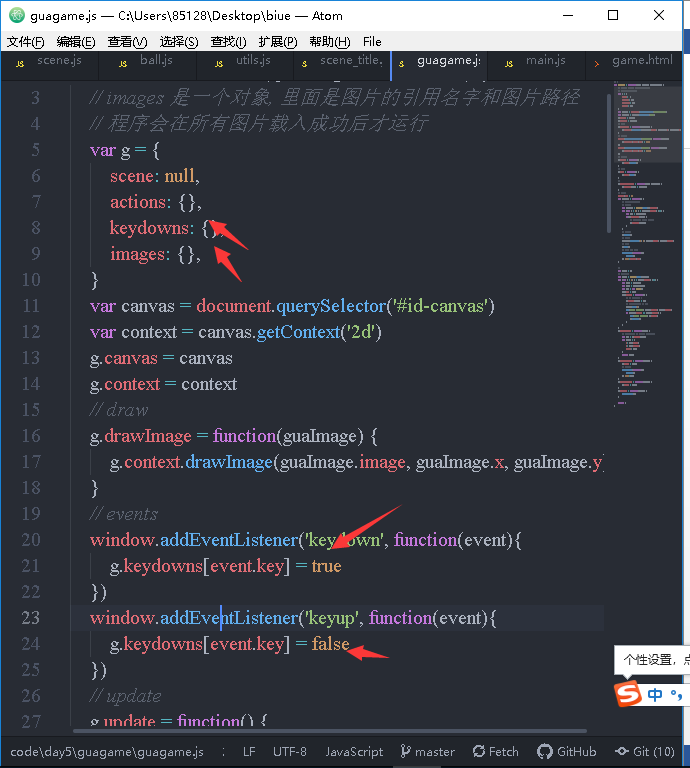


图5.4 按键监听模块

关键代码：

// 瓜

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.update = function() {

g.scene.update()

}

// draw

g.draw = function() {

g.scene.draw()

}

//

g.registerAction = function(key, callback) {

g.actions[key] = callback

}

// timer

window.fps = 30

var runloop = function() {

// log(window.fps)

// events

var actions = Object.keys(g.actions)

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

var key = actions[i]

if(g.keydowns[key]) {

// 如果按键被按下, 调用注册的 action

g.actions[key]()

}

}

// update

g.update()

// clear

context.clearRect(0, 0, canvas.width, canvas.height)

// draw

g.draw()

// next run loop

setTimeout(function(){

runloop()

}, 1000/window.fps)

}

//

var loads = []

// 预先载入所有图片

var names = Object.keys(images)

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

let name = names[i]

var path = images[name]

let img = new Image()

img.src = path

img.onload = function() {

// 存入 g.images 中

g.images[name] = img

// 所有图片都成功载入之后, 调用 run

loads.push(1)

log('load images', loads.length, names.length)

if (loads.length == names.length) {

log('load images', g.images)

g.\_\_start()

}

}

}

g.imageByName = function(name) {

// log('image by name', g.images)

var img = g.images[name]

var image = {

w: img.width,

h: img.height,

image: img,

}

return image

}

g.runWithScene = function(scene) {

g.scene = scene

// 开始运行程序

setTimeout(function(){

runloop()

}, 1000/fps)

}

g.replaceScene = function(scene) {

g.scene = scene

}

g.\_\_start = function(scene) {

runCallback(g)

}

return g

}

6 系统测试

6.1 测试计划

单元测试（类测试）：25天、集成测试（交互测试）：10天、系统测试：8天。

6.1.1 测试进度

单元测试：2018.01.01-2018.01.25

集成测试：2018.02.10-2018.02.20

系统测试：2018.02.20-2018.02.28

6.1.2 测试方案

软件工程理论要求任何程序开发都不能离开软件测试，游戏项目的开发当然也是如此。要想开发的游戏被玩家接受与喜爱，最基本的是游戏能正确合理运行，所以，游戏的测试就变得非常重要了。

6.1.3 测试目的和原则

本次测试要找出游戏的各种错误、缺陷，发现游戏中不合理的地方，进一步完善游戏内容，增强用户的体验，确保游戏正常运行。  根据测试目的，本测试的测试原则如下：

（1）测试用例的设计，要同时包含合理的和不合理的条件输入。

（2）测试用例要有输入数据与预期输出的结果两部分。

（3）测试计划必须严格执行。

（4）对测试结果检查和记录。

（5）多人测试，增强测试的全面性。

（6）为了后续游戏完善的方面，应该将测试材料归档，包括测试的计划、 用例、错误统计与最终分析结果。

6.1.4 测试环境

系统硬件需求：

Intel Core i3-4000M或更高主频CPU，512MB以上内存。

系统软件需求：

Windows XP 以及更高版本的操作系统、Chrome 63.0.3239.132以上的版本浏览器。

6.2 测试说明

6.2.1 游戏界面显示测试

游戏的功能测试主要对游戏初始界面、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，如图6.1 结果：正常



图6.1 游戏初始界面

1. 小球生命值显示 初始状态：显示剩余3条生命 输入：挡板没有挡住小球 输出：生命值显示减少，如图6.2 结果：正常



图6.2 游戏开始界面

1. 游戏等级升级显示 初始状态：等级显示为1 输入：清除第一关所有砖块 输出：等级显示为2，如图5-4 结果：正常



图6.3 Level升级界面

1. 游戏通关后界面显示 输入：通过所有关卡  输出：“You Win,Press R To Restart”，如图6.4 结果：正常



图6.4 游戏通关界面

6.2.2 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6.5



图6.5挡板测试

6.2.3 小球移动控制测试

（1）小球发射检测

输入：进入游戏后按下“空格” 输出：小球开始运动 结果：正常

输入：进入游戏后点击鼠标“左键”

输出：小球开始运动

结果：正常，

（2）小球碰撞检测

①测试小球与墙壁、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

输入：开始游戏观察碰撞

输出：小球与碰撞后，消除砖块并正常反弹

结果：正常

如图6.6



图6.6小球发射

7 测试及性能分析

7.1 测试结果概述

测试过程中，程序出现了一些错误，界面不能正常加载、小球不能移动、操作命令无响应等，测试和调试后，解决了这些问题，完成了这个可以正常运行的版本，当前游戏的开发结束。

7.2 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。

8 总结与展望

8.1 本论文研究成果

通过此次毕业设计，我更深入的了解了JSP，而且能用他做一些有趣的小应用。之前学过JSP，但是没用它做过游戏，这次设计使我从另一个方面了解了JSP，也是对学习的进一步加深。  打砖块游戏的开发，我认为最主要的是一些游戏算法的设计和游戏界面界面的设计，或者说所有的小游戏都是以这些为核心。JSP语言ES6的语法是一种面向对象的语言，非常符合我们平时的思维习惯，所以只要思考了游戏的流程，游戏的算法实现并不困难。当然，游戏设计实现的过程中还是遇到了很多困难，因为以前没有用JSP做过游戏，所以很多方面不熟悉。  从最初学习JSP语法，到现在完成简单的打砖块游戏，我感觉到了自己在进步，学习和编程的能力都有很大的增强，同时，这也增加了我对游戏项目的实战经验和技巧，相信以后可以更轻松的完成此类项目。  在做项目这段时间里，我学到了许多新知识，同时复习并巩固了之前学习的旧知识，通过对它们的实践，我对我自己的专业软件工程也有了更深刻的理解。一个打砖块游戏的开发，贯穿了整个软件工程的过程，从开始的项目选定，到需求分析，到算法的设计和实现，直到最后的软件测试，把整个大学里学习的很多专业知识综合起来，构成了一个完整的体系。

参考文献

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

[2] ChristianHeilmann[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] 百度百科.JSP .  <https://baike.baidu.com/item/JSP/141543?fr=aladdin>.

[6] 百度百科.Canvas .  https://baike.baidu.com/item/canvas/8928454.

[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，笔者要特别感谢邓邵伟导师的热情关怀和悉心指导。撰写毕业论文的过程中，邓邵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，笔者都得到了邓邵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！