

单位代码： 10293 密 级： 公开

南京邮电大学

# 专业学位硕士学位论文



论文题目： 基于 HTML5 与 Node.js 的游戏引擎的设计与开发

学 号 1213012003

姓 名 徐佳宾

导 师 曹士珂

专业学位类别 工程硕士

类 型 全 日 制

专业（领域） 电子与通信工程

论文提交日期 二零一六年四月

# **The Design and Development of Game Engine Based On HTML5and Node.js**

Thesis Submitted to Nanjing University of Posts and  
Telecommunications for the Degree of  
Master of Engineering



By

Xu Jiabin

Supervisor: Prof. Cao Shike

April 2016

## 南京邮电大学学位论文原创性声明

本人声明所呈交的学位论文是我个人在导师指导下进行的研究工作及取得的研究成果。尽我所知，除了文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得南京邮电大学或其它教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示了谢意。

本人学位论文及涉及相关资料若有不实，愿意承担一切相关的法律责任。

研究生签名：\_\_\_\_\_ 日期：\_\_\_\_\_

## 南京邮电大学学位论文使用授权声明

本人授权南京邮电大学可以保留并向国家有关部门或机构送交论文的复印件和电子文档；允许论文被查阅和借阅；可以将学位论文的全部内容编入有关数据库进行检索；可以采用影印、缩印或扫描等复制手段保存、汇编本学位论文。本文电子文档的内容和纸质论文的内容相一致。论文的公布（包括刊登）授权南京邮电大学研究生院办理。

涉密学位论文在解密后适用本授权书。

研究生签名：\_\_\_\_\_ 导师签名：\_\_\_\_\_ 日期：\_\_\_\_\_

# 摘要

伴随着 HTML5 标准的正式定稿，HTML5 游戏也在如火如荼的发展。当微信游戏《围住神经猫》如病毒般在朋友圈中传播开来，无论是游戏开发者还是普通游戏用户亦或是互联网投资者，都感受到了这一次 HTML5 游戏的强势来袭必定会带来一波游戏革命的新浪潮。而目前游戏市场上的各类产品鱼龙混杂，且以轻量级单机小游戏为主，玩家不能实时交流、在线邀请好友一同游戏。为了解决此类问题，实现用户体验更佳、功能更完善的射击类游戏，本文设计并实现一款基于 HTML5 与 Node.js 的平面射击类的游戏引擎。基于此游戏引擎进行平面射击类游戏的开发，可以比之前更加高效、快速。并且实现玩家实时交流以及在线邀请一同游戏，增强游戏的趣味性。

本文首先阐述了课题研究的来源及背景，简单的介绍了目前 HTML5 游戏的发展现状，以及 HTML5 游戏引擎在国内外发展的现状。以此明确了开发一款基于 HTML5 与 Node.js 的平面射击类的游戏引擎课题研究的可行性。该游戏引擎以 B/S 模式架构，通过浏览器与服务器搭配实现。而服务器端则是采用 Node.js 来搭建、管理 HTTP Server，静态文件访问通过 Apache 服务器实现。由于我们的目标是实现轻量级、碎片化时间游戏的开发，所以我们没有采用数据库进行辅助开发。但是如果想做一款做中度、重度的 HTML5 游戏，使游戏与用户间有更高的黏性，那么我们无可避免的要借助于数据库来存储游戏数据。

本游戏引擎可以为所有开发基于 HTML5 平面射击类游戏的开发者提供一系列 API 和工具，使得此类游戏的开发更加快速、便捷。为了更好的观察此游戏引擎的功能，在此游戏引擎上，我们实例化了一个游戏《坦克大战》，通过一系列的测试，游戏可以正常运行，并且实现实时在线交流和在线邀请功能，基本达到了预期的效果。

**关键词：** HTML5，Node.js ， WebSocket，游戏引擎

## Abstract

With the official announcement of the HTML5 standard, the HTML5 game is in full swing. In the we chat, the game named "trap the nerve cat" spreads such as viruses in the circle of friends. Developers, ordinary users or other investors, they all feel HTML5 game will comeback with a wave of game revolution at this time. In the market products are mixed with various quality and light-weight single small games account for most. Most games can not take a real-time communication, or invite friends to play together online. In order to break the technical barriers, achieve a better user experience, design a more perfect shooting game engine, it designs and realizes a plane shooting game engine based on HTML5 and Node.js in this thesis. Through this game engine, you can develop the game more efficient and faster. And you can achieve a real-time exchange between players and games together online to enhance the fun of the game.

In this thesis, it describes the source and background of the topic, introduces the current development trend of HTML5 game, tells the future effect that HTML5 game will achieve, as well as the development of the HTML5 game engine at home and abroad. Therefore, it is feasibility to develop a plane shooting game engine based on HTML5 and Node.js. The game engine is based on Browser/Sever mode architecture to achieve collocation between the browser and server. It is convenient to build HTTP server by Node.js, and to achieve static file server through the Apache software. Because to achieve a light-weight game engine using the fragmentation of time need not with the development of database. However, to develop moderate HTML5 games, there is unavoidable that the game would get support from database development to store data.

This game engine can provide a series of API and tools for all developers to develop plane shooting games based on the HTML5, to make the development of the game more rapid and convenient. In order to observe the function of the game engine, it instantiates a game "battle of tanks". Through a series of tests, the game can be operated normally , and realize real-time communication and invites online, basically reached the expected effect.

**Key words:** HTML5, Node.js, WebSocket, Game engine

# 目录

第一章 绪论.....	1
1.1 课题背景、来源及意义.....	1
1.2 国内外研究现状.....	2
1.3 课题主要工作.....	3
1.4 论文组织结构.....	3
第二章 游戏引擎的技术支持.....	5
2.1 HTML5.....	5
2.1.1 HTML5 应用新的市场机会.....	5
2.1.2 HTML5 行业的发展预测.....	6
2.2 JavaScript.....	7
2.2.1 JavaScript 的特点.....	7
2.2.2 JavaScript 的框架.....	8
2.3 Node.js.....	9
2.3.1 Node.js 的特点.....	9
2.3.2 Node.js 发展前景.....	10
2.3.3 Node.js 的社区.....	11
2.4 开发工具.....	11
2.5 浏览器.....	11
2.6 本章小结.....	11
第三章 Game Engine.....	12
3.1 游戏引擎的定义.....	12
3.2 开发 Game Engine 的意义.....	13
3.3 Game Engine 的架构以及包含的模块.....	14
3.4 国内商用 HTML5 游戏引擎——白鹭引擎.....	16
第四章 游戏引擎的架构分析与设计.....	18
4.1 游戏引擎架构分析.....	18
4.2 平面射击类游戏架构分析.....	20
4.3 平面射击类游戏功能流程分析.....	22
4.4 平面射击类游戏玩法分析.....	25
4.5 平面射击类游戏主要模块分析.....	26
4.6 本章小结.....	28
第五章 游戏引擎的功能设计与实现.....	29
5.1 游戏引擎服务器端功能设计.....	29
5.2 游戏引擎地图功能设计.....	32
5.3 游戏引擎英雄模块功能设计.....	33
5.4 游戏引擎子弹模块功能设计.....	37
5.5 游戏引擎玩家控制模块功能设计.....	39
5.6 游戏引擎游戏结束模块功能设计.....	40
5.7 本章小结.....	40
第六章 游戏引擎的功能与性能测试.....	41
6.1 测试环境的搭建.....	41
6.2 玩家登陆与即时交流功能测试.....	41
6.3 玩家单人模式下的功能测试.....	44
6.4 玩家双人模式下的功能测试.....	45

6.5 游戏结束的功能测试..... 47

6.6 游戏引擎文件的加载性能测试..... 48

6.7 局域网与互联网环境下的游戏性能与功能测试..... 48

6.8 本章小结..... 49

第七章 总结与展望..... 50

参考文献..... 52

致谢..... 54

# 第一章 绪论

## 1.1 课题背景、来源及意义

自 2014 年 10 月 29 日，W3C 官网正式宣布 HTML5 的标准制定完毕，越来越多的开发者把目光聚向了 HTML5。而关于 HTML5 在游戏中的应用也受到了国内外越来越多的个人、社区、公司的关注。随之，国内外也出现了许多优秀的 HTML5 游戏引擎。著名游戏引擎网站 <http://www.html5gameengine.com> 分析了目前国内外全部的 HTML5 游戏引擎，并对游戏引擎的使用情况做了一个综合排名，前五位的分别是 Construct 2、ImpactJS、EaselJS、pixi.js、Phaser。

表 1.1 html5 游戏活跃人数排名

序号	游戏名称	游戏人数（万）	所用引擎	公司名称
1	空间花藤	1357	Dom	台战涛
2	来消星星的你	1006	Cocos	乐天城科技
3	一起猜猜	1003	Dom	佛山闪吧科技
4	谁是色盲	748	Dom	榴莲科技
5	最强大脑	684	Dom	佛山闪吧科技
6	开心 2048	630	Dom	开开心心工作室
7	一起跳跳	613	Dom	上海晟游网络
8	一个不能死	597	Createjs	佛山闪吧科技
9	摩天高楼	496	Createjs	佛山闪吧科技
10	我偏不	495	Dom	叮当工作室

根据腾讯玩吧 2015 年 3 月 27 日的前 10 名的 HTML5 游戏的数据统计，我们可以看到有 3 款游戏用了游戏引擎，其他的开发都是用的原生 dom。那么我们再对比前 50 名游戏，有 22 款游戏运用游戏引擎开发，其中有 4 款是游戏开发公司自己来封装的游戏引擎，28 款游戏用原生的 dom 进行开发。那么为什么还有超过一半的游戏没有用游戏引擎进行开发呢？主要原因有三个：其一，部分游戏在设计、交互时，功能超出了现有游戏引擎的覆；其二，对于开发单个 HTML5 游戏，现有的游戏引擎太过于繁重、复杂，文件加载慢，影响游戏性能；其三，游戏开发者或者开发团队的喜好，偏向于使用原生 JavaScript 来进行开发。



作者本人先后在欧朋浏览器南京研发中心、金智教育实习过，所做的工作均与 HTML5 及 HTML5 游戏相关。实习过程中，作者在进行 HTML5 游戏的二次开发时，发现了现有 HTML5 游戏的一些不足之处，比如每款游戏的代码冗余率很高，只能在单人模式下，不能邀请其他玩家进行对战。所以，作者吸取国内外其他 HTML5 游戏引擎的优点，并加入自己的特点来开发一款易用、通用、可对战的平面射击类游戏引擎。

本篇论文，则是从最基本最常见的游戏功能进行设计，开发一款通用型最强，代码量最少的游戏引擎。并且，对于目前 HTML5 的游戏大多是单人模式，本论文实现了实时在线邀请的双人模式，趣味性更强。

## 1.2 国内外研究现状

2010 年，苹果 Chief Executive Officer 史蒂夫·乔布斯在 Apple 官网发表了一篇文章，题目为《Thought on Flash》，文章细数了 flash 技术存在的六大问题，并提到了一门可以代替 flash 的语言——HTML5，这大概是人们第一次开始关注 HTML5。

到 2014 年底，将有百分之八十的手机应用将全部或部分基于 HTML5 技术进行开发的。此数据表明绝大多数的手机 app 应用，其内容的展现都将是 web 页面的形式，此类型的应用包括淘宝、微信、脸书、推特等。Chrome 浏览器宣布于 2015 年 9 月初起不再支持自动播放 Flash 动画。零售巨头亚马逊也证实 Amazon 门户以及其他网络在内的所有广告将不再使用 Flash 技术实现。在不久的未来，我们可以预料基于 HTML5 的广告终将代替全部的 Flash 广告。

2014 年，是 HTML5 在中国发展的元年。这一年智能手机硬件设备极大的提升、微信平台的成熟都为 HTML5 的发展铺平了道路，而 HTML5 游戏也正是借助此优秀的富媒体属性，一跃成为当年最受关注的热点之一。正是在这一年，国内涌现了众多的 HTML5 游戏开发团队、论坛社区、游戏引擎、游戏分发渠道，与此同时其他传统游戏公司也开始在 HTML5 领域发力布局。这其中包含腾讯玩吧、360 游戏、白鹭时代、猎豹游戏、触控科技、台战涛、佛山闪吧科技、乐天成科技、榴莲科技、叮当工作室、蝴蝶互动、开开心心工作室、比悦科技、黑桃互动、上海晟游网络等。而这些公司创作开发了一大批新颖的小游戏，例如《谁是色盲》、《一个都不能死》、《寻找房祖名》、《愚公移山》、《貂蝉有妖气》、《时尚都市》、《来消星星的你》。除此之外，还有一系列的病毒式传播的小游戏，它们大多有同一个特点：较低的每用户平均收入，较高的日活跃用户数量。而此时国外也涌现了诸多基于 HTML5 的游戏引擎，其中有很多优秀的作品，比如 Impact、Crafty、Playcraft、Jaws。

### 1.3 课题主要工作

课题是在浏览器环境下开发基于 HTML5 和 Node.js 的游戏引擎。该游戏引擎主要涉及到前端和后端开发的诸多技术,比如 canvas 标签的绘制,客户端之间的通信,客户端与后端数据的交互,定时刷新技术。本课题设计的游戏引擎有以下几个特点:设计的引擎具有常用功能,引擎文件小,加载速度快;功能逻辑清晰,调用简单。此外,本课题设计的游戏引擎还可以邀请玩家一同对战,解决了多数 HTML5 游戏只能单人模式的尴尬。

本课题主要工作是在浏览器环境下开发游戏引擎,在开发的过程中所做的工作如下:

- (1) 熟悉目前国内外 HTML5 游戏引擎的发展现状。
- (2) 熟悉 Chrome 浏览器开发者工具和 HTML5、CSS3、JavaScript 编程语言。
- (3) 研究 Node.js 中封装的 Socket.IO 库,熟练掌握编写网络应用程序的方法。
- (4) 设计并实现游戏引擎的结构,科学合理的设计游戏引擎的要解决哪些问题,设计哪些模块,各模块之间的联系。
- (5) 设计并实现游戏引擎的功能,实现了预期的功能,成功应用在基于 HTML5 坦克大战的游戏中。

### 1.4 论文组织结构

本论文主要从五个章节来介绍游戏引擎的设计与实现,论文的整体结构安排如下:

第一章,绪论。介绍了本课题研究背景、意义及来源,国内外研究现状。通过课题来源引出对课题的思考,进而构思课题的主要工作。最后介绍了及论文的组织结构。

第二章,游戏引擎的开发环境及语言。首先介绍了开发语言,分别是前端语言的 HTML5、CSS、JavaScript,服务器端语言 Node.js,及其相关的其他知识。开发环境则是 PC 端浏览器(谷歌、火狐、IE)、Apache。开发工具是 Sublime、WebStrom、谷歌浏览器开发者工具、Node.js 客户端。

第三章,游戏引擎。首先介绍了游戏引擎的起源、定义。再通过介绍开发游戏引擎的目的,开发前应该如何准备,来引出如何组织与设计游戏引擎的架构,如何设计游戏引擎的功能模块。最后则是介绍了一款国内目前流行的 HTML5 游戏引擎--白鹭引擎。

第四章,游戏引擎结构的设计与实现。针对平面射击类游戏引擎,我们从引擎架构到联网分析再到游戏引擎模块都进行了设计。这为我们第五章具体的实现功能指明了方向。

第五章,游戏引擎功能的设计与实现。这一章则是针对第四章的设计,通过程序设计来

具体的实现功能。完成具体的游戏引擎模块逻辑。

第六章，游戏引擎的应用与测试。我们通过开发的游戏引擎封装了一款游戏，坦克大战，分别在本地、局域网、互联网的环境下进行功能与性能的测试。

第七章，总结与展望。对本次课题做了总结，分析了该游戏引擎的优点及不足之处，并对该游戏引擎的未来发展做了展望。

## 第二章 游戏引擎的技术支持

本章主要介绍了开发游戏引擎所要运用的语言、环境、工具等等。首先介绍 HTML5 与 JavaScript 相关的语言定义以及它们的新特点、发展前景、选择此技术实现的原因。其次介绍游戏引擎开发的主要工具。最后介绍开发过程用到的浏览器、服务器环境。

### 2.1 HTML5

HTML5 是指超文本标记语言（HTML）的第五次重大修改。2014 年 10 月底，W3C 组织宣称，经过长达八年的艰苦努力，HTML5 标准规范终于制定完成并公布。HTML5 的愿景是依此来开发出更简单的 web 程序，编写出更简洁的 HTML 的代码，取得更好的用户体验。

#### 2.1.1 官方规范的新特性

HTML5 新增的四种重点特性：

（1）新的语义标签，有助于搜索引擎的抓取或者辅助技术对 web 页面的理解，增强页面自身的可访问性。

（2）新的表单属性，新增邮件、url 地址、数字、数字值范围、日期、搜索域等输入类型，并设置了一些常见属性对表单字段进行调整。

（3）新增多媒体元素标签，如 video 和 audio 标签，使 web 页面无需安装第三方的插件的情况下就可以播放音频视频等，且带暂停、播放、快进等功能。此外通过设置 draggable 等属性，可以让图像等素材具备可拖放效果。

（4）定义 canvas 标签，实现通过 JavaScript 编程来控制在画布上绘制指定图案，可以支持 2d 和 3d 效果。

#### 2.1.2 HTML5 应用新的市场机会

尽管 HTML5 的新特性具有很大吸引力，但是它的应用并不及技术发展得那么迅速。从硬件角度来看，国内手机和平板两种移动设备使用最广泛，其次是 PC 端，也有少量电视和游戏设备画面采用 HTML5 技术。但这对于 HTML5 而言，还存在很大发挥余地。然而，在多方因素的影响下，在中国 HTML5 的传播效果却是最火热的，这与 HTML5 的跨平台特性密

切相关。从 2014 年起，国内省市城市开始大范围布局 Wi-Fi 和 4G 信号，以此来为用户日常提供高速的上网速度；除此之外，手机软、硬件设备升级换代，手机、平板等设备对 HTML5 新特性支持良好，国内浏览器开发团队也逐渐升级浏览器内核，以支持更加丰富的 HTML5 特性。国内手机应用里面，排名第一的微信拥有大概 80% 的安装率，而微信平台易于分享页面的特点也促使了各种类型的 HTML5 小游戏作品病毒式传播。这促使更多 HTML5 游戏公司和游戏的发芽生根，也推动了国内面向 PC 端、移动端的 HTML5 游戏引擎的快速发展。

在 2015 年，移动端仍是 HTML5 游戏传播的主要阵地。小公司的自媒体、广告公司、大型公司的广告部、企业新媒体部，都会或多或少通过 HTML5 小游戏进行各种宣传与内容营销。成本低，维护简单，易于传播并且传播量大，都是 HTML5 小游戏吸引他们的地方。

### 2.1.3 HTML5 行业的发展预测

根据国外发展现状以及国内的 HTML5 发展环境，可以预测国内 HTML5 行业在未来五年的三个发展方向：

(1) 传统网页游戏逐渐向交互性 HTML5 游戏方向发展。通过中国游戏产业年会上公布的《2015 年中国游戏产业报告》，我们了解到 2015 年我国国内游戏市场规模高达一千四百多亿，其中移动游戏占比 31.5%，而网页游戏占比 15.6%，约为前者的一半。而根据 2015 年 DataEye 公司整理的 HTML5 游戏数据报告中显示，在网页游戏中 HTML5 游戏所占比例并不高，并且很少用户为之付费。虽然 HTML5 游戏拥有无需安装下载、易于社交分享、传播速度快、开发成本低、开发周期短等优点，但如果未能有效利用 HTML5 的新特点，未能重视 HTML5 游戏的交互性，它很难成为游戏市场中的黑马。因此，未来的网页游戏可以去尝试与拥有优良通信特点 HTML5 相结合，往跨平台等交互特征更好的形式进行发展。

(2) 市场垂直领域的解决方案往 HTML5 在线应用方向发展。市场垂直领域与 HTML5 关系最紧密的主要有三种类型：在线教育、电商和流媒体。由艾瑞公司整理的《2015 年中国在线教育平台研究报告》显示，国内的在线教育公司或者平台正逐渐将主要业务与流量入口由电脑端转移到移动端。因为移动端可以打破线下教育的时间、空间限制，可以让用户通过更灵活的碎片化时间进行学习，而基于 HTML5 的在线教育平台无疑是实现移动端课程教学的唯一方法。基于 HTML5 技术能更方便地对视频播放时长进行记录，并且在无需下载安装任何插件的情况下实现多种反馈机制，有助于对学习的课程进行线上考核。电商、流媒体的情况亦是相同，根本原因是用户消费行为趋于移动端。而且国内网民多数通过手机访问网页进入各类电商或新闻媒体站点，因而这些市场垂直领域更迫切的需要基于 HTML5 技术支撑

的移动互联网解决方案。

(3) 传媒业营销往 HTML5 模版应用直接填充内容方向发展。目前,在国内通过 HTML5 进行开发的软件大多都提供样式模板服务,这与 HTML5 网页优秀的跨平台特性有关。HTML5 网页基础框架开发设计完成后,通过修改源代码中标签元素的属性、参数便能替换为目标素材,形成一个内容属于自己的网页。因此 HTML5 网页可以用来制作填充内容的模板。

由此可见,国内浏览器厂家或 HTML5 web 程序开发公司只有从 HTML5 新特性入手,着眼于底层交互、功能才可能应对各行业的需求发展。HTML5 取代 Flash 已经是势不可挡的趋势,但伴随着虚拟现实技术(VR)、增强现实技术(AR)的发展,可预见的是 HTML5 能走得更远。

## 2.2 JavaScript

JavaScript 是一门程序设计语言,同时也是一门脚本语言,其特点是可以使用动画、多媒体元素、交互式的响应效果来丰富拓展 web 程序的功能。JavaScript 从 1995 年创立出来,截止到现在已经有 20 年左右的发展历史了,但是在刚发展的时候并不受到人们的重视,一度被程序开发者戏称为“丑小鸭”。而随着服务器端技术的快速发展,JavaScript 凭借独有的跨平台、学习成本低等特点快速跟进。JavaScript 获得 2014 年 Tiobe 编程语言年度语言的桂冠,其背后 jQuery、React.js、Bootstrap、Node.js、AngularJs 等框架的推动功不可没。

### 2.2.1 JavaScript 的特点

与 Perl、Python、Ruby 等脚本语言类似,JavaScript 也是一种解释性语言,其弱语言化的语法特征,为 web 程序的开发提供了一个简单高效的路径。JavaScript 的基本语法以及结构形式与其他编程语言大同小异。我们在使用 JavaScript 时,不需要像 C 语言那样先编译再执行,而是在执行过程中,JavaScript 解释器对程序依次逐行地解析。常见的是将 JavaScript 与 HTML 标签结合在一起使用,以此来更加快速、方便的响应用户的操作,让用户觉得自己在操作一个桌面应用一样。

对于用户的操作响应,JavaScript 则是通过事件驱动的模式进行的。那什么是事件驱动?用户在浏览网页时执行了一个操作而产生的动作,被称为“事件”(Event)。比如鼠标单击、关闭网页、弹出下拉菜单、敲击键盘等都可以被视为事件。而所谓的事件驱动就是当此类事件发生以后,有可能会导致执行相应的事件响应,或者执行某些处理此事件的脚本的这一过程。

JavaScript 的执行主要是通过浏览器内核进行解析,与操作系统具体是 Linux 还是 Windows 无关,这就要求计算机能成功安装浏览器,并且支持 JavaScript 语言,JavaScript 程序就可正确执行,进而极大的减少工作量,只需要编写一次,就可以在任何平台、系统使用。JavaScript 另外一个优点是即时性。JavaScript 使得 web 网页可以瞬间对于用户和网页的交互的操作做出响应,比如点击一张图片、提交一个表格,在页面上随意的移动鼠标等等。与 PHP 服务器端编程运行时的延迟不同,JavaScript 对于页面上事件的响应则不会存在令人尴尬的延迟,因为这种延迟是只存在于 Web 浏览器和 Web 服务器之间,页面自身不会存在。因为在使用 JavaScript 语言开发时,我们不会允许浏览器频繁加载或者重新载入 Web 页面这种糟糕的用户体验方式,使用 JavaScript 创建的 web 程序给使用者的感觉更像是桌面应用,而不是一个的普普通通的静态网页。

### 2.2.2 JavaScript 的框架

JavaScript 还有一个令开发人员纠结的地方:编写起来比较困难。虽然 JavaScript 比 C、C++、Java、PHP 等编程语言的学习成本要低一些,但不可否认 JavaScript 仍是一种编程语言。而且包括网页设计师在内的许多人,发现 web 编程是个艰苦的工作。其中更令人沮丧的是一些网页设计师对 JavaScript 的兼容性理解不深,导致部分在谷歌浏览器中运行正常的程序,可能在 Internet Explore 7 浏览器中毫无效果。浏览器开发厂商的不同,导致浏览器内核解析效果不同,甚至同一浏览器版本不同解析效果也会有差异,这种浏览器兼容性问题导致开发者要花一定的时间对网页进行测试,如此才能实现目标网站对所有访问者都能很好的展现出来。

而正是因为不同浏览器存在解析差异以及原生 JavaScript 写起来太过繁琐,jQuery 库应运而生。jQuery 是基于 JavaScript 封装的一个库,其意义是使编写 JavaScript 程序更简单,更快捷,更有趣。其实我们可以把 jQuery 看作是一个比较复杂的 JavaScript 程序,其底层则是采用原生的 JavaScript 编写的,最大的两个特点是简化了复杂的编程书写,解决了 JavaScript 在跨浏览器中的兼容性问题。那么在 jQuery 解决了使用 JavaScript 最尴尬的两个问题,即不同浏览器内核对于解析的复杂和繁琐特性之后,我们也许就可以用单独一行代码来完成之前多行代码所做的工作,而且也不用再去担心自己编写的 JavaScript 代码兼容性不够,还要再花很多时间进行浏览器型号与版本的测试。

除了 jQuery 之外,也有很多其他著名的 JavaScript 框架,比如 Angular.js(一个谷歌公司内部开发的 JavaScript 框架,可以迅速构建企业级的程序应用)、React.js(可提供复用的 web 组

件。React.js 是美国脸谱公司开发并实现的虚拟 DOM，其封装的组件易于复用，可以简单的接入到现有项目中去）等等。

## 2.3 Node.js

我们可以将 Node.js 理解为是一个 JavaScript 语言的执行环境。它其实是对 Google 浏览器 V8 引擎（应用于谷歌浏览器内核）进行了封装,使用 C++进行开发的。Node.js 新增了一些在特定环境下的用例，并对其进行版本的迭代优化，提供易于理解使用的 API，以让 V8 引擎在浏览器之外的环境下运行的更快更稳定。

### 2.3.1 Node.js 的特点

创建 Node.js 的意义在于可以在服务器端环境中高效的执行和运行 JavaScript 代码。自 JavaScript 被公布以来，其都被定义为一种在浏览器环境下才能运行的客户端脚本语言，但是现在通过将 JavaScript 运行环境剥离出来，并安装到服务器上，那么我们就可以在服务器环境下执行 JavaScript 代码，而不再是单纯的依赖于浏览器解释器的解析，进而就可以在服务器端功能开发的时候使用 JavaScript 语言。由于 Node.js 其拥有异步非阻塞特性，所以 Node.js 在处理长连接、多请求的情况下很有优势。Node.js 的编程是以 JavaScript 的语法为基础的，并且封装了一些特殊的 API 接口，接口文档在官方网站上有具体的说明，所以我们在编写 Node.js 时其实就是设计编写 JavaScript 程序，参照并使用 Node.js 的 API 接口来实现服务器端具体功能的开发。

Node.js 的最大的特点就是对程序执行性能的提升。谷歌浏览器 V8 引擎的开发运用了语言编译领域的最新技术，以此使得用 JavaScript 语言开发的程序在执行速度上能够近似于 C++等底层语言开发的程序，这使得有效的降低了程序的开发成本。其次，与主流的服务器端开发语言相比，比如 PHP 和 Java，每当网页进行 HTTP 请求时，请求链接都会使服务器产生一个线程，服务器会为每个线程都需要分配一定的内存，当一台服务器主机承受不了用户的并发请求数时，就需要增加服务器的数量。而 Node.js 可以有效处理 http 多请求问题的特殊之处在于 Node.js 解决连接服务器的模式。对于每个请求的 HTTP 链接，Node.js 都会自动生成一个事件，而此事件则会被推送到 Node.js 的引擎进程中，而不是像主流服务器端的解决方法，即为每个请求建立一个新的 OS 进程，并给每一个 OS 进程配备一整套的内容。

由此可见，我们可以清楚的认识到 Node.js 与 PHP 相比较来优势在于能合理有效的解决高并发请求问题，因为 Node.js 是基于事件驱动机制，因而相对与传统服务器语言的做法能够



更多的节约服务器的内存空间。Node.js 能够简单的搭建出一个独立 server，只需要一句代码，这是非常快捷方便的，对于此类简单的服务器，使用 Node.js 比使用 C 语言来搭建会更快捷、更方便。

### 2.3.2 Node.js 发展前景

现代的程序开发人员具有很强的创造性，他们会在一两个月的时间内发明一种新技术。而另外一些没有很好管理、衍生的技术则就会慢慢消失。但是 Node.js 独树一帜，推出来的六年里，其技术已经发展的相当成熟，并且受到越来越多的程序爱好者的喜欢！

Node.js 意在成为一个开源、开放的平台，在其上，程序开发者可以熟练的使用 JavaScript 来为服务器环境和浏览器环境开发应用。如今，使用 JavaScript 的程序开发者数量与日俱增，并对当下互联网开发领域产生了较大的影响，在此现状下，Node.js 也成为越来越多码农的重要开发工具之一。Node.js 客户端内封装有谷歌浏览器的 V8 引擎，因此能够合理的利用 web 程序的可压缩性和高性能来加速程序的执行运行速度。并且 Node.js 是基于事件驱动机制的，可以更好的服务于网络，另外 Node.js 内部还使用高速的网络服务器。除此之外，Node.js 还通过使用许多“非阻塞”接口来实现事件循环的特点，提高代码的执行效率。

伴随着 Node.js 所受到的关注度与日俱增，程序开发人员使用和学习 Node.js 的频率也在逐渐增长。如图 2.3.2 是 Google 公司绘制的重要编程语言职位占比图，我们可以明显的发现与 Groovy 和 Ruby on Rails 等流行技术相比较，对于 Node.js 的需求可谓是如日中天、独占鳌头！有许多国际知名软件、互联网公司诸如微软、谷歌、雅虎、亚马逊等都看好 Node.js 的发展前景。

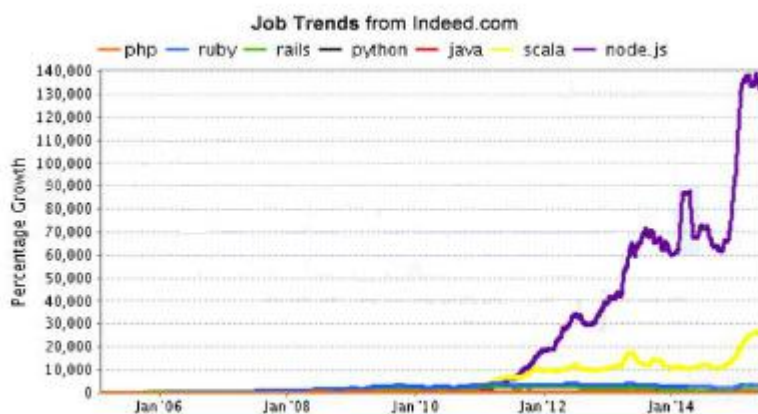


图 2.1 不同语言职位数量变化图

### 2.3.3 Node.js 的社区

另外 Node.js 的中、英文社区里面有大量的活跃粉丝，虽然 Node.js 社区才成立几年而已，但是却意外地受到特别多的程序开发者的积极关注。无论是 Node.js 的入门者还是资深专家都会聚在社区里，并且将个人使用 Node.js 感悟的心得、碰到的困难问题等等一一反馈贡献到社区中，致力于把 Node.js 社区打造成为每一份子都能在社区内愉快的学习、分享经验并获取帮助的乐土。

## 2.4 开发工具

关于游戏引擎的开发工具主要是代码编辑器，调试工具。常用的代码编辑器有 sublime text3 、 WebStrom 。调试工具则有 Chrome 浏览器的开发者工具（F12）、Firefox 火狐浏览器的 FireBug 等。另外辅助以 Apache 服务器，浏览器等等。

## 2.5 浏览器

以 Net Market Share 2015 年发布的浏览器市场占有率为例，全球 PC 端浏览器市场占有率从高到低依次为，IE 浏览器，Chrome 浏览器，Firefox 火狐浏览器，Safari 浏览器，Opera 浏览器。五家共占有 99.58% 的全球总市场。而根据百度流量研究院的统计，2015 年国内浏览器市场份额最大的仍是对 HTML5 性能支持不够良好的 IE 浏览器，约占 40%；Chrome 浏览器以微弱劣势次之，约占 35%（实际份额主要源自 360 浏览器）。但对比分时间段的数据可见，IE 浏览器占有率呈下降趋势，Chrome 浏览器则呈上升趋势。而对于 HTML5 新特性的支持上，Chrome 浏览器最优，其次是 Firefox 火狐浏览器、Safari 浏览器、Opera 浏览器，最差的是 IE 浏览器。但是最新的 IE11 或者 IE10 浏览器对于 HTML5 大部分的新特性也能支持。

## 2.6 本章小结

本章主要介绍了本次课题中所涉及到的开发语言、开发技术、开发环境、开发工具。介绍了近些年 HTML5 的发展特别是在 HTML5 游戏的发展，并对其未来满怀信心。Node.js 实现 JavaScript 在服务器端运行，使得前后端开发可以公用一套语言，极大的减少了开发成本和程序兼容性问题。这一技术领域的发展，也促进了游戏引擎领域的发展。本章的理论知识将为下面的设计实现工作打下良好的基础。

## 第三章 Game Engine

本章主要介绍了游戏引擎的起源、定义。再通过介绍开发游戏引擎的目的，开发前应该如何准备，来引出如何组织与设计游戏引擎的架构，如何设计游戏引擎的功能模块。最后则是介绍了一款国内目前流行的 HTML5 游戏引擎--白鹭引擎。

### 3.1 游戏引擎的定义

“游戏引擎（game engine）”此称谓最早出现在 20 世纪 90 年代中期，是在开发第一人称射击游戏（first person shooter, FPS）《毁灭战士（Doom）》时所出现的表述。这款一经发布就受到全球玩家一致欢迎的游戏是由 id Software 公司独立研发的。《毁灭战士》游戏的组织架构是特别清晰的，其主要部分可以归纳为：核心软件模块（例如游戏图形渲染模块、碰撞检测模块和音频设置模块等）、UI 设计、游戏场景、游戏规则（rule of play）、游戏外设控制。这种就架构体系是很有特点的，比如其他游戏开发商取得这类游戏引擎的授权，只需要设计好游戏 UI、地图布局、武器样式、角色、操作游戏规则等，对引擎软件做出很少的修改，不用去修改引擎核心，就可以把游戏打造成新产品。这一游戏架构的诞生也引发 MOD 社区的发展。MOD 的含义是指，某一游戏的众多玩家参与的兴趣小组，或着是小型的独立游戏工作室，借助于游戏原开发商提供的 API 或者游戏工具箱，对现有的游戏进行创作、修复或者更改等等，从而制作出不同于原来的游戏。

在一个游戏中，游戏引擎完全控制着游戏内容、游戏进度、画面 UI、游戏音乐、外设操作等部分，将游戏中所包含的各多媒体元素结合起来。但是，其实游戏和其引擎之间是无法分割开来的。有些引擎有比较清晰的划分，另外的则没有尝试把二者分开。也许在某一游戏中，游戏的渲染代码可以熟练画出一只“妖怪”；但是在另一款游戏中，渲染模块可能只提供各种各样的材质和配色功能，“妖怪”也许完全是用数据定义出来的。不存在游戏开发商可以完全区分出游戏和游戏引擎之间的差别。因为在不同的游戏设计下，游戏与游戏引擎的定义也会经常变化。

通常情况下，研发的游戏引擎都是为某一特定类型的游戏而设立的。就像为拳击格斗类游戏而研发的游戏引擎，与大型多玩家网络游戏、第一人称射击类游戏或者实时策略游戏引擎是存在很大区别的。但是各类引擎很多功能是重复的，就像任何种类的 3D 游戏，都是通过玩家输入或者操作（例如键盘、鼠标）、渲染画面、平视显示器、性能良好的音频系统等。比如，尽管虚幻引擎是为射击类游戏而开发的，但是游戏开发团队仍可以用它来开发其他种

类的游戏，像英佩游戏公司研发的大型射击类游戏《战争机器》，上海麻辣马公司全力打造的有的角色动作冒险游戏《格林童话惊魂记》，还有韩国 Acro Games 公司设计研发的高清赛车游戏《速度之星》。

在 HTML5 游戏出来之前，游戏引擎都是用从 C/C++ 开发实现的。本文所开发的游戏引擎是基于 HTML5 与 Node.js 实现的，并且主要适配的是平面射击类游戏。尽管开发的语言不一样，但是游戏引擎的设计思路、实现的功能是一致的。在 3.5 小节中，我们会介绍一下目前国内比较流行的 HTML5 游戏引擎，白鹭引擎。

## 3.2 开发 Game Engine 的意义

开发游戏引擎的意义在于极大的降低游戏开发者编写代码时的重复劳动，并且提高游戏的可移植性。游戏引擎相当于游戏的模板，授权后的开发者只需往游戏引擎中填充内容就能够制作出一个新游戏。游戏引擎可以理解为是一条游戏生产链，在其内部集成了许许多多的模块开发工具，比较著名的游戏引擎 Unreal 3 其使用起来非常简单快捷，不但可以支持多种语言的开发，其关卡设置以及场景设计器都十分好用。事实上一款优秀的游戏引擎可以把关卡设置与物件流水线顺利的分离出来，极大的减少游戏代码的开发量。

随着显卡制造工艺技术的与日提升，现在的 GPU 不仅可以应用固定的变换与光照管道线，而且也可以应用顶点着色和像素着色方式，因此对于游戏开发团队，拥有了更大的自主研发空间，也因此可以使游戏场景更接近现实的绘制效果。在此技术背景下，资金殷实的游戏公司都开始设计研发新一代 3D 游戏引擎。其中市场上开发者评价最好的两个引擎是 Quake III 与 Unreal II 引擎。其他方面，随着游戏引擎社区的不断发展、不断壮大，活跃的开发者们也开发出一款不错的开源 3D 游戏引擎，例如 OGRE。通过使用这些游戏引擎中内嵌的场景渲染模块，我们可以在游戏中生成更加真实的水波、天空白云、一望无际的森林、流熔岩的喷发火山等效果。随着现代大型游戏场景规模的不断扩大，创新的顶点动态光照技术也开始逐渐被应用在了游戏的室外场景中。

实现超大规模场景和真实感更强的虚拟世界是 3D 游戏发展的下一个目标。超大规模游戏场景与游戏虚拟仿真技术，不可置否的也成为了全球三维游戏行业特别受到关注的两个研究地方。例如著名游戏公司暴雪的《魔兽世界》和韩国 NCSoft 公司的《天堂 2》已经在超大规模场景与虚拟仿真方面研究了多年并且目前已经实现了相当的技术突破和进展。在《魔兽世界》游戏中，如果按照人体的比例来衡量，游戏中世界地图的面积已然达到了  $35\text{km} \times 35\text{km}$ 。

随着电子娱乐业的渐入繁荣，国内研发 3D 游戏引擎的公司也有诸多作品展现。其中比较

不错有盛大游戏公司的 3D 引擎，网易游戏部的 3D 引擎，锦大科技的 AURORA 引擎等，其他有独立游戏工作室设计研发的 3D 引擎，例如 WIN3D 系列引擎，Origo 系列引擎，TUAM9 系列引擎等。但实事求是的说，国内的公司目前还处在使用和模仿优秀引擎的阶段，很多都是基于国外优秀引擎的基础之上拓展而来的。真正严格意义上拥有自主知识产权的游戏引擎还没有，或者说自主开发产品的质量性能和欧美日韩还存在一定差距。

### 3.3 Game Engine 的架构以及包含的模块

一般情况下，通过游戏引擎开发的游戏，大概会存在这样一个关系：

游戏 = 引擎(程序) + 资源(图像、声音、动画等)

目前游戏引擎的架构都是按照 Model、View、Controller 原则进行设计的，其中将逻辑部分和显示部分分开，再通过一个逻辑控制流程来配合 Client 的请求和 Server 的响应。View 层主要是负责页面的展示，Controller 层则是负责控制工作流程的逻辑执行，外设的输入输出，对各类事件的响应，Model 层则是模型、逻辑程序的功能实现。而游戏绘制的基本流程则会遵循传递消息、更新数据、绘制各节点等环节。

现在，市面上的游戏引擎种类繁多，其主要区别在于游戏逻辑封装的不同而不是区别于哪种语言开发的游戏引擎。例如，通过 C 语言开发的引擎则有数据与函数相分离的特点。而以 C++ 开发的游戏引擎则大量运用类结构。以及现在光环日耀的 CBSE，则是从基于组件的架构来开发引擎。通常，一款游戏引擎需要几个主要模块：基本引擎框架，资源管理，渲染，基本逻辑，物理引擎。

下面我们重点解释一下比较重要的模块。

#### 3.3.1 渲染器

渲染器的主要功能是使游戏场景可视化，让玩家可以看见游戏场景，进而让玩家能够根据屏幕上所看到的场景作出适当的决断。当构造一个游戏引擎的时候，第一步就是开发渲染器。因为如果看不见场景，那么又如何判断自己的程序代码在正常工作呢？有超过一半的 CPU 处理时间是消耗在渲染器渲染页面的；通常也是此开发部分，游戏开发人员可能会受到最苛刻的评判。如果游戏引擎的渲染器渲染出的效果很差，事情将会变得很棘手。包括我们的程序技术，我们的游戏和我们的公司很有可能将成为业界的笑话。渲染器也是我们最依赖于外部厂商的地方，在这里他们将最大限度的处理潜在游戏类型。如此说来，建造一个渲染器的确是一件劳苦功高的任务，如果没有一个好的渲染器，游戏或许永远不会跻身于排行榜前列。

### 3.3.2 碰撞检测

碰撞检测模块在 3D 游戏中非常重要。作为一个优秀的碰撞检测模块，游戏人物在游戏场景中可以平滑走动是正常要求。如果遇到正常高度的台阶，人物可以自动走上去，而再高的台阶则就不能通过。如果遇到坡度较小的斜坡可以，游戏人物可以走上去，坡度过大则无法通过。如果存在前进方向被挡住的环境下，碰撞检测模块都会尽可能地使人物沿合理的方向运动而不是原地静止。碰撞检测模块不仅要满足上面这些要求，还要做到移动足够精确和稳定，防止人物在某些情况下穿墙而掉出场景。优秀的碰撞检测模块反而不会引起玩家的注意，因为这符合物理运动中的常识。相反，如果做得差了，玩家很明显的就会发现，比如，人物经常卡住不能前进或者人物穿越了障碍物。所以很多开发者觉得写碰撞检测代码是一个艰巨的任务，不仅碰撞检测的算法复杂，而且测试容易出问题，就算写的好也不容易出彩。

### 3.3.3 动画系统

目前游戏引擎中普遍采用的动画系统有两种，一是骨骼动画系统，另一个是模型动画系统。骨骼动画系统用内置的骨骼带动物体产生运动的效果，这种比较常见。模型动画系统则是在模型的基础上直接进行变形。游戏引擎则会将这两类动画系统预先配置到游戏中，以此动画设计师可以更方便的为角色设计各种各样的动作造型。

### 3.3.4 光影处理系统

光影效果则是指场景中的光源对场景内的人和物的影响方式。游戏内的光影效果全部都是由游戏引擎控制产生的，其中包括光的折射、散射、反射等基本的光学原理以及动态光源、彩色光源等高级效果。我们都可以游戏引擎的开发上通过技术手段来实现。

### 3.3.5 人体学输入设备通信系统

游戏引擎还有一个重要的功能就是实现玩家与游戏之间的沟通，处理来自键盘、鼠标、手柄等等外设的信号。如果游戏定位是非单机游戏，则游戏需要支持联网的特性，我们会在游戏引擎中开发网络连接模块，以此来处理客户端与服务器之间的通信。

另外游戏引擎还会包含一些独立的模块，例如游戏 UI、背景音乐、游戏音效、网络、脚本（有些类型的游戏引擎需要脚本和逻辑的关联性非常强，有些脚本则比较独立）等等。

### 3.4 国内商用 HTML5 游戏引擎——白鹭引擎

自微信游戏《围住神经猫》如病毒式在朋友圈散播开来，其创造了 3 天 PV（访问量）破亿的奇迹，使得人们的视野再次汇聚在 HTML5 游戏上，与此同时也引领了 HTML5 “即点即玩”轻游戏潮流的开始。作为《围住神经猫》的游戏引擎--白鹭引擎（即 Egret Engine，以下简称 Egret）也渐渐被人了解。Egret Engine（白鹭引擎）是白鹭时代公司发布的一款开源免费的移动游戏引擎。其开发环境是基于 Typescript 语言，JavaScript 语言，和 ES6 语言。借助于白鹭引擎，游戏开发人员可以方便地制作出可以运行在手机应用 WebView 或者手机浏览器中的 HTML5 移动游戏，也可以通过白鹭引擎的开发工具编译输出成可以在 Android、IOS、Windows Phone 等平台运行的原生移动游戏。通过 Egret 引擎设计研发的 HTML5 移动游戏，不仅能让 HTML5 游戏具备超强的性能表现，并且拥有较高的运行效率。

白鹭引擎有六大功能点：

- （1）灵活的显示对象，通过内置显示列表，可方便管理游戏中可视化元素。
- （2）矢量图/位图双重支持，内置矢量绘图功能，可实时绘制矢量图形，与位图搭配使用。
- （3）方便的资源管理，通过资源管理模块，可同步或异步加载，让你的游戏体验流畅。
- （4）多种屏幕适配策略，内置四种适配策略，让游戏完美适配不同分辨率。
- （5）高效的物理引擎，内置 P2 物理引擎，可快速搭建类似《愤怒的小鸟》游戏。
- （6）绚丽的粒子系统，通过内置几十种参数，可配置千余种粒子效果，满足各种游戏特效。

白鹭引擎的 egret.js 的部分功能代码：

显示对象添加到舞台,可以传入两个参数，前者是舞台的对象。

```
p.onAddToStage = function (stage, nestLevel) {  
    _super.prototype.$onAddToStage.call(this, stage, nestLevel);  
    var bitmapData = this.$Bitmap[0 /* bitmapData */];  
    if (bitmapData) {  
        egret.Texture.$addDisplayObject(this, bitmapData);  
    }  
};
```

显示对象从舞台移除

```
p.$onRemoveFromStage = function () {  
  
    _super.prototype.$onRemoveFromStage.call(this);  
  
    var bitmapData = this.$Bitmap[0 /* bitmapData */];  
  
    if (bitmapData) {  
  
        egret.Texture.$removeDisplayObject(this, bitmapData);  
  
    }  
  
}
```

### 3.5 本章小结

本章通过对游戏引擎概念的了解，对游戏引擎的作用、功能、优势的介绍，以及目前游戏引擎中所常见的模块，来为我们设计平面射击类游戏引擎打好基础。通过介绍，尽管我们设计的游戏引擎基于的语言环境不同，但是很多设计思路值得借鉴的。最后通过对国内比较流行的游戏引擎白鹭引擎的部分功能进行分析与使用，明确了开发基于 HTML5 游戏引擎的逻辑思路。



## 第四章 游戏引擎的架构分析与设计

本章针对课题游戏引擎架构进行分析与设计。由于不可能存在一种游戏引擎可以适配任何游戏的情况，所以我们本次设计的游戏引擎主要是针对平面射击类游戏。设计的重点在于以怎样的一种方式进行游戏引擎的架构组织，方便引擎开发，方便未来的开发者使用此引擎。另外一重点就是引擎要有什么样的功能，以及各功能的切割与组合。

### 4.1 游戏引擎架构分析

如前文所述，我们要开发基于 HTML5 与 Node.js 的、平面射击类的游戏引擎，那我们来分析一下引擎的结构。我们采用 MVC 模式进行架构，分为 Model 层、View 层、Controller 层。MVC 模式的工作流程如下：首先在 View（界面）层触发某个事件，其次 Controller（业务）层响应事件处理，然后导致数据更新，再次寻找确认 Model 层的数据载体，接下来 Model 层携带数据回到了 View 层，最后 View 层更新数据。前端架构除了 MVC 模式以外还有 MVP（Model - View - Presenter）、MVVM（Model - View - ViewModel）。

MVP 模式则是将 Controller 层改名为 Presenter 层。名称改变的同时也改变其模式三部分的通信方向。MVP 模式有如下特点：

- （1）Model，View，Presenter 三部分之间的通信，都是双向的；
- （2）View 与 Model 不发生直接联系，都通过 Presenter 层进行传递；
- （3）View 非常薄，不会部署任何业务逻辑，我们称之为"被动视图"（Passive View），也就是无任何主动性，而相应的 Presenter 层非常厚，全部业务逻辑都部署在这边。

MVVM 模式则是将 Presenter 层改名为 ViewModel 层，其与 MVP 模式基本相同。不同之处在于，MVVM 模式采用双向绑定（data-binding）：如果 View 层发生变动，则会自动反映在 ViewModel 层，反之也是如此。而常见的 Angular 框架和 Ember 框架都采用 MVVM 模式。MVP 模式与 MVVM 模式更适合在数据与页面交互频繁的项目里。所以我们在这里选择 MVC 模式。

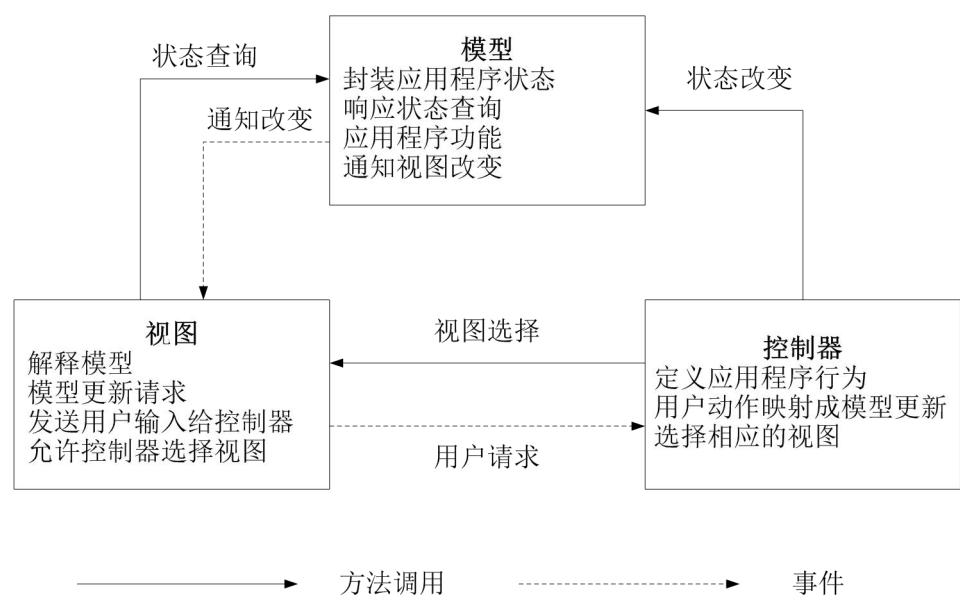


图 4.1 MVC 模式

**Model** 层的主要工作是用来固化数据的，而在我们的平面射击类游戏中，页面 **HTML** 就相当于我们的 **Model** 层，而页面的某一区域就是我们游戏所要展现的舞台。另外在 **Model** 层，也会有一些页面的其他元素，辅助来让玩家来更好的理解游戏的操作。

**View** 层的主要工作是用来展现数据，在我们的平面射击类游戏中，相当于样式控制层（**CSS**）和数据展示层(部分 **JavaScript**)。包括射击者、被射击物体（或人物）、障碍的样式，也包括他们位置的移动、重绘等等。**View** 层也是与用户进行交互的一层，当用户进行某个动作的时候，比如点击、聚焦，**View** 层需要去请求新的数据并且更新到页面上。

**Controller** 层的主要工作是控制业务逻辑执行，调控 **Model** 层和 **View** 层的联系，它控制程序的工作流程，对触发的事件作处理和响应，而这里的事件不但包括用户的行为还有数据模型上的改变。**Controller** 层通过获取用户事件的类型，通知 **Model** 层作出相应的更新处理，同时将 **Model** 层的更新和改变通知给 **View** 层，使 **View** 层作出相应的改变，因此，**Controller** 层确保 **View** 层和 **Model** 层的一致性。这一层主要是由前端 **JavaScript** 和 **Node.js** 来共同完成。比如，当一颗子弹射向某个敌人，子弹会不会射中敌人，未射中会怎样，射中又会怎样，这些都是需要 **Controller** 层去处理的。

## 4.2 平面射击类游戏架构分析

### 4.2.1 平面射击类游戏功能分析

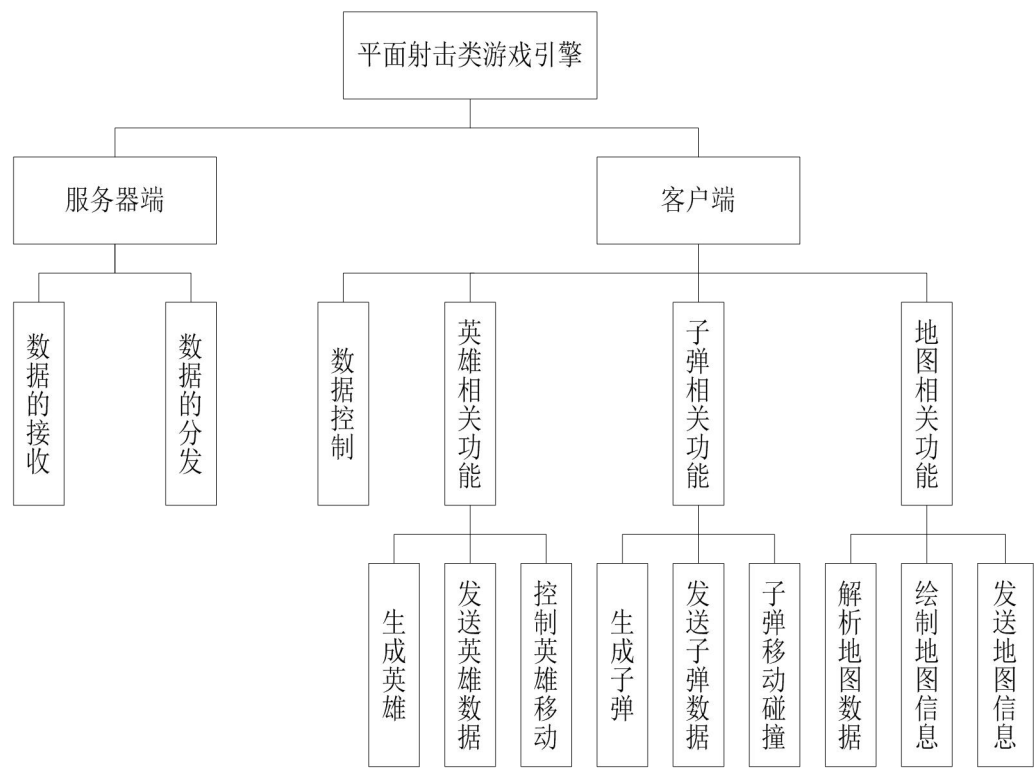


图 4.2 平面射击类游戏引擎架构

针对服务器端进行功能分析，主要是接收来自客户端的数据，包括玩家、子弹、地图等数据，将数据进行处理后分发给所有的用户，监听用户是否在线。针对客户端，则要实现地图的相关功能（包括发送地图信息、解析地图信息、绘制地图）、子弹的相关功能（发送子弹信息，子弹的运动，子弹的碰撞，解析子弹信息，绘制子弹）、英雄的相关功能（英雄的运动信息、发送位置数据、绘制英雄、出生的动画效果）、地图障碍物或者敌人的爆炸效果、打包数据发给服务器端。

### 4.2.2 平面射击类游戏玩家控制分析

玩家主要具备以下几种功能：控制英雄的前进、后退、向左、向右、停止；控制英雄射击，包括选择射击方向，控制射击时间；玩家可以在舞台上看到控制的英雄、敌人、障碍物；玩家可以击毁障碍物或者敌人；玩家可以查看爆炸效果；玩家可以查看自己的生命数和比分，查看剩余敌人数量。

### 4.2.3 平面射击类游戏地图共享分析

在双人模式下的对战类地图中，要求双方在相同的地图上对战，并且每方的移动或者射击都要实时更新在页面中，地图应该有较快的加载速度，更新的延迟要尽量缩短，不然会破坏游戏体验。每当一方进行英雄的移动、射击、发射子弹或者爆炸，都会重新发送数据到服务器，服务器再把信息反馈给双方，再进行同步更新至页面上。

### 4.2.4 平面射击类游戏音频控制分析

根据游戏的进程与动作，设计或配置不同的游戏音效。包括，英雄出生，英雄移动，射击子弹，击中敌人，击中障碍物等等。另外游戏配有背景音乐，可以流畅、清晰的进行全程播放，增加游戏趣味性。当游戏结束时，也会根据游戏的结果播放不同类型的音频。

### 4.2.5 平面射击类游戏联网对战分析

联网对战是指玩家在浏览器打开游戏页面之后，进行邀请其他玩家一同来进行游戏。双方可以实时看到对方。为实现此客户端之间数据的通信，我们先来分析一下现有的处理客户端数据交互的网络架构：

对等网络架构（Peer-to-Peer）。在对等网络架构中，所有的信息都是在两个客户端之间直接传递，不需要通过服务器。但首先需要所有的客户端都已成功的连接服务器，这样某一客户端才能通过对等架构网络找到目标客户端。而对等网络架构又分为两种，如下：

完整连接拓扑架构。在完整连接拓扑架构中，任意两个客户端之间都存在连接，而信息的传递就是直接作用在它们之间。

(1)环状拓扑架构。在环状拓扑架构中，所有的客户端依次排列成一个环状，传递信息时需要通过两客户端中间的所有客户端。

对等网络架构架构的特点在于延时较小。在客户端/服务器架构中，信息的流向首先是从请求客户端到服务器，然后再从服务器端传递到目标客户端。而对等网络架构则是在两客户端之间建立通信链路，使信息直接传递，如此一来其传递时间就会缩减为客户端/服务器架构的一半。另外，对等网络架构不需要服务器，这就意味着无论是游戏公司还是工作室，都不再去维护游戏的中央处理服务器，也不必再去支付高昂的主机托管和管理费用。

与此同时，对等网络架构也存在缺陷。在游戏玩家数量较少的时候，对等网络架构的延时很小并且无需服务器端的参与。但是当游戏的玩家数量过大的时候，对等网络架构的缺陷就很明显了，因为在对等网络架构中，任意一个客户端都和其他客户端维持着一个特有的链路，因此建立了很多链接，通信链路利用率很低，这些也都是对等网络架构得游戏可扩展性教差的原因。

(2) 客户端/服务器架构 (Client/Server, C/S)。在此架构中，客户端只负责将信息传送至服务器上，然后服务器依据目标地址再把信息传送出去。Socket 服务器主要负责运行并监听客户端的连接尝试，接受和管理来自客户端的请求，合理安排客户端之间的信息传送路径。每一个 Socket 服务器都会有与之对应的一个 IP 地址或主机名称，并且至少某一端口上监听客户端请求。如果客户端与 Socket 服务器成功建立起连接，那客户端也就和服务端建立了不间断的 Socket 连接。通过此 Socket 连接，任意客户端就可以将信息传递给服务器，同样，服务器端也可以将信息传递给任意客户端。此类 Socket 连接是可以随时建立与可用的，因此通过 Socket 连接进行信息的传递是特别方便的，并且客户端数据的更新是以 Socket 连接事件驱动为依托并实时更新。

鉴于我们课题的实际情况，我们采用 C/S 架构的 Socket 服务器方式来实现客户端数据的交互。

### 4.3 平面射击类游戏功能流程分析

按照上述的分析，我们可以把游戏的主要流程划分为三个：服务器端流程、客户端页面刷新流程、客户端发送数据流程。

#### 4.3.1 服务器端更新数据流程设计

服务器端更新数据流程的触发是通过某一客户端向服务器发送数据，服务器端在接收到数据之后，对数据进行处理再广播发送给所有的客户端。

服务器端流程图设计如下：

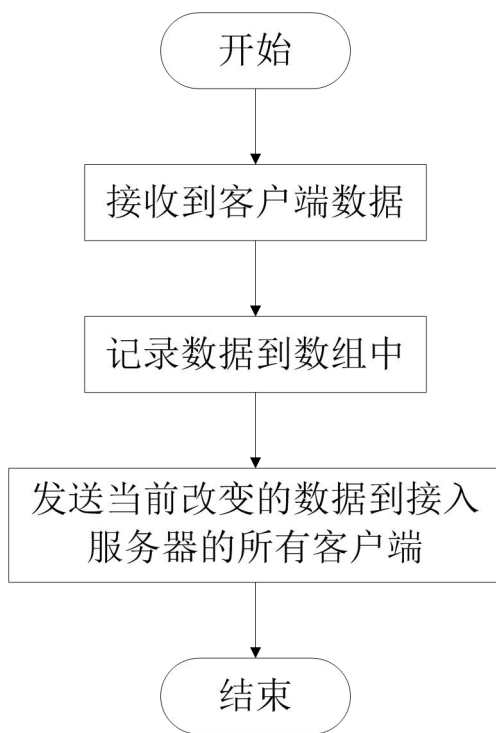


图 4.3 服务器端流程

首先是服务器接收到客户端发送的数据，并将其储存在内存中的一个数组中，再发送改变的数据到所有的客户端，客户端接收并处理。

### 4.3.2 客户端页面刷新流程设计

客户端页面的刷新是通过 JavaScript 语言的定时刷新技术来实现的。每隔 20ms，定时器会去执行绘制地图、英雄、敌人等元素。玩家首次进行连接的时候，定时器会去获取地图信息，再绘制地图、生成英雄敌人。

客户端页面刷新流程图设计如下：

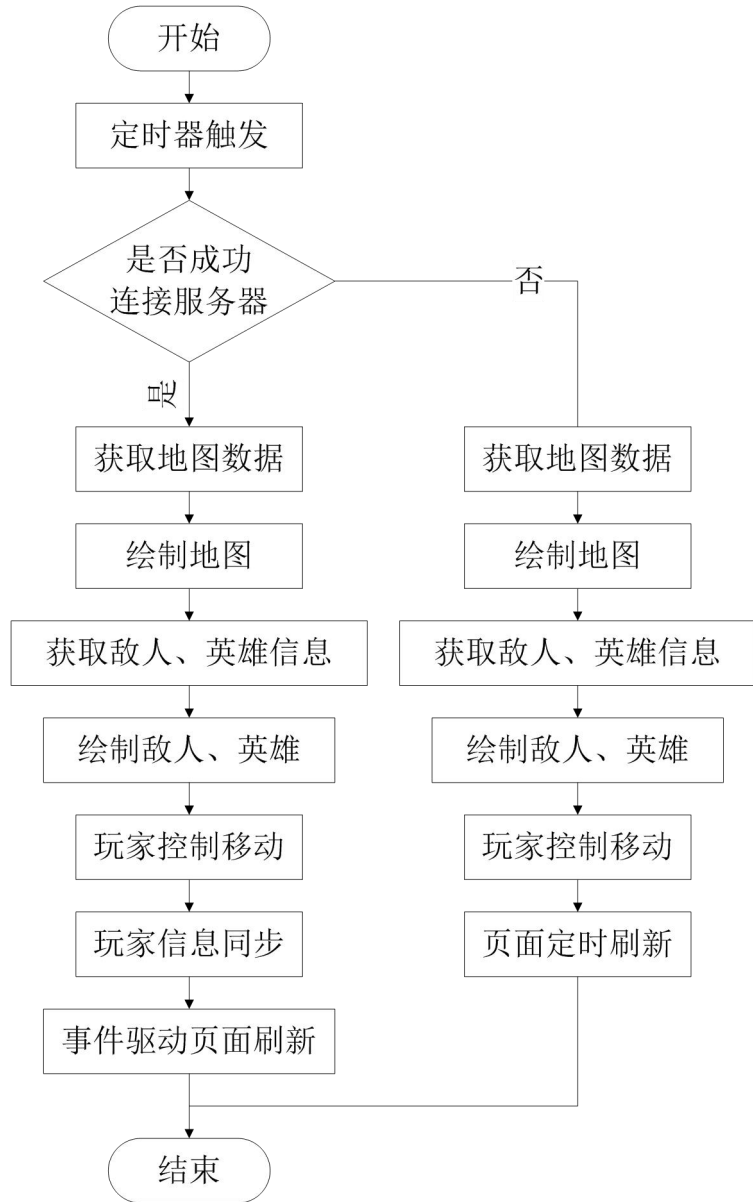


图 4.4 客户端刷新流程

客户端开始游戏，触发定时器工作，每隔 20ms 会去检查是否连接上服务器，如果未连接上，则结束。如果连接上，则获取地图信息，绘制地图。再获取英雄（敌人）数据，生成实例，绘制英雄（敌人）。最后把英雄（敌人）当前的数据发送给服务器，以备下次请求。

4.3.3 客户端发送数据流程设计

每当英雄或者敌人的位置发生变化，或者子弹移动或者地图上障碍发生变化都需要客户端重新向服务器发送数据。

客户端发送数据流程图设计如下：

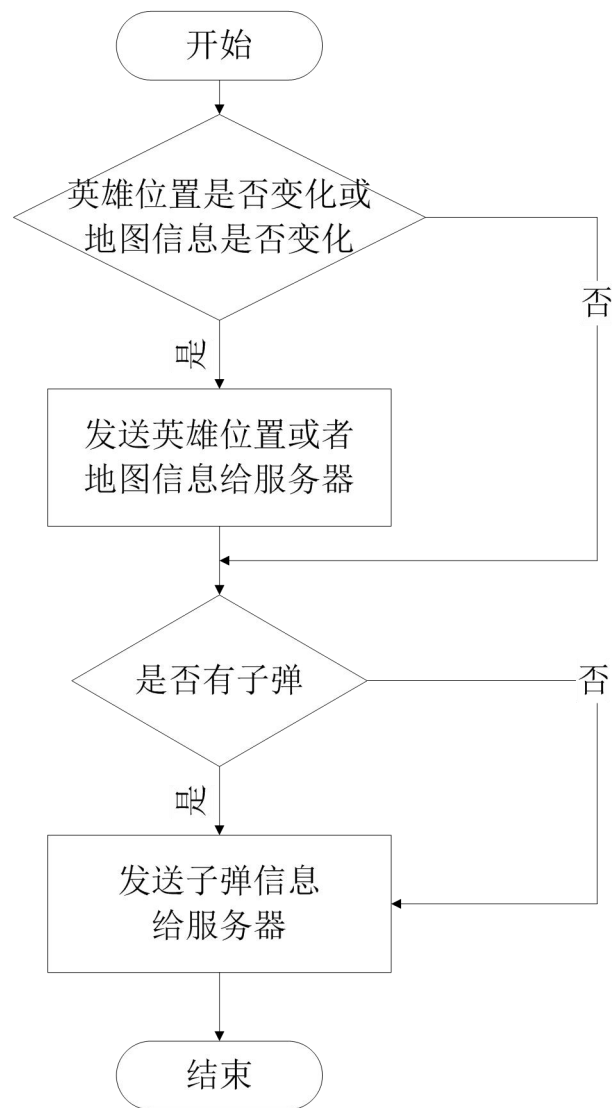


图 4.5 客户端发送数据流程

首先客户端会检测英雄位置坐标或者地图样式信息是否发生改变，若存在变化，则客户端需要把改变后的信息发送给服务器，若位置和地图信息无变化，则客户端需要检查是否有子弹信息。如果有子弹，则把子弹信息发给服务器，无，则结束。

## 4.4 平面射击类游戏玩法分析

平面射击类游戏会分为单人模式，双人模式两类。进入页面前，需提示用户输入用户名。玩家自己选择是单人模式还是多人模式，不管在何种模式下，游戏都提供所有在线玩家的一个聊天室，大家可以在里面发布消息、交流心得。



### 4.4.1 单人模式

在单人模式下，定义玩家射击者为英雄，对面射击者为敌人，另有障碍物若干。至于具体游戏开发中，有没有障碍物，有的话障碍物有多少，敌人数量多少，这些都是可控的。

英雄需要做的就是射击消灭敌人的同时，守护自己的家不让敌人破坏或者是保护好自己不能死掉。当家配破坏或者自己的生命值降为 0,则游戏结束，敌人取得胜利。在当前关卡取得胜利以后，游戏会提示是否愿意挑战下一关卡，如果是“是”会继续挑战下一关卡，如果是“否”则会从头再来。

### 4.4.2 双人模式

双人模式为对战模式，为两方玩家之间的对战，对方是敌人。双人模式下的胜利与单人模式类似，以击中对方或者击毁其老巢为胜利。在地图上，两方玩家可以各自发挥自己的谋略、反应能力来击败对方。比之单人模式，双人模式更加有竞技性，娱乐性，单人模式下的电脑玩家毕竟比不上真实玩家的操控。

### 4.4.3 地图的选择

在单人模式下是不允许自主选择地图的，必须是每一个关卡闯过来才能进入下一个关卡。而在双人模式下，是允许选择地图的，但是双人模式下的对战是每一局作为结束，每一局结束，双方玩家可以选择再重新开一局，不再有“是否愿意挑战下一局”的字样。

## 4.5 平面射击类游戏主要模块分析

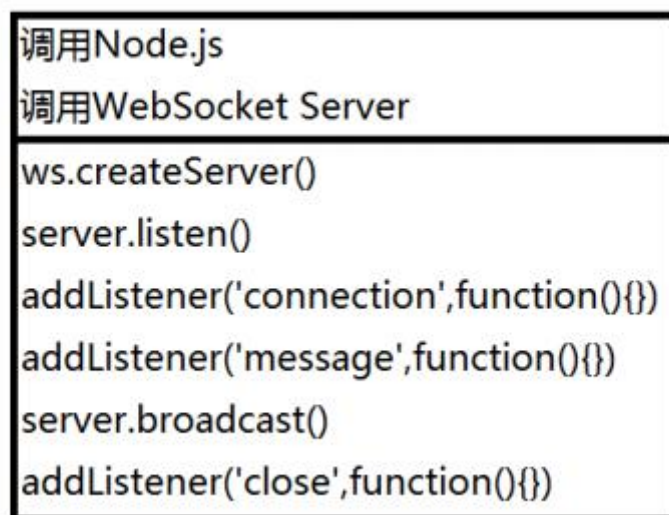
模块分析可以从服务器端、客户端两方面进行分析。

### 4.5.1 服务器端模块

服务器端环境使用的是在 window7 操作系统下安装 Node.js 客户端，命令行安装包管理 NPM，在 Node.js 客户端中通过编程简单的建立一个 Sever。除了 Node.js 客户端之外，我们还需要一个 Web 发布环境，本文使用 Apache 作为 Web 页面的预发布程序。

服务器端调用的模块都是通过 Node.js 的包管理来实现的。主要有创建服务器、监听服务器端口、加载 socket.io 模块。然后通过具体封装的方法来处理接收、处理、发送数据，监听事件。

服务器端模块所包含的具体方法如下：



```
调用Node.js  
调用WebSocket Server  
ws.createServer()  
server.listen()  
addListener('connection',function(){})  
addListener('message',function(){})  
server.broadcast()  
addListener('close',function(){})
```

图 4.6 服务器模块主要方法

## 4.5.2 客户端模块

游戏引擎中的客户端模块负责玩家控制游戏，绘制游戏人物画面、实现玩家实时在线交流以及发送和接收游戏信息数据等功能。客户端模块主要分为基础工具类模块、游戏主函数模块、地图设置生成模块、游戏英雄模块和音乐系统模块。

(1)基础工具类模块中有加载图片方法，通过此方法来预加载图片，并且在图片完成加载之后，通过异步调用传入的其他方法。

(2)游戏主函数模块是整个游戏运行的主逻辑住调用的类，其中有主入口函数 `init()`、当服务器器打开时触发方法 `onopen()`、接收数据方法 `onmessage()`、键盘事件控制方法 `control()`、游戏开始 `start()`和游戏时间触发器 `move()`。

(3)地图设置生成模块是地图相关方法模块，其中主要有定点清除地图方法 `clean()`、清除全部地图方法 `cleanAll()`、绘制定点位置地图方法 `draw()`、绘制全部地图方法 `drawAll()`、取得定点位置 `z-index` 数值方法 `getZ_index()`等。

(4)游戏英雄模块是玩家控制的英雄的主要方法的类，其中包括英雄的 `birth` 方法、进行下一步的方法 `getNextStep()`和绘制英雄图像 `drawHero()`等方法。

(5)音乐系统模块主要是控制播放游戏音乐，其中有基本的音乐控制方法，播放音乐 `play()` 和暂停音乐播放 `pause()` 等方法。

## 4.6 本章小结

本章从如何设计一款平面射击类游戏引擎出发，引出游戏引擎的组织架构，平面射击类游戏玩法分析，再到游戏引擎功能模块组成、前后端数据如何交互、客户端服务器端该分别如何设计等等。其核心在于如何设计如何架构，一个好的组织架构可以达到事半功倍的效果，不仅节省人力物力也可以提高开发速度，本章为第五章的具体功能模块的实现指明了方向。

## 第五章 游戏引擎的功能设计与实现

本章针对课题设计的游戏引擎进行具体功能模块的逻辑开发。主要分为服务器端、与客户端的开发。服务器端主要实现对客户端事件的监听，对客户端发送数据的处理，并且再对所有客户端发送响应事件的一个过程。客户端则主要实现地图模块、子弹模块、英雄模块、玩家控制操作模块、游戏结束模块的逻辑开发。

### 5.1 游戏引擎服务器端功能设计

#### 5.1.1 服务器端同步的信息

在双人模式的对战地图下，需要双方进行数据信息的同步，主要是以下四点：

地图信息的同步。对战地图的选择是在客户端完成的，但是当双方对战时，地图上的障碍存在被击毁的变化，需要将地图元素的变化发送给对战的双方，并且重新绘制地图。但是在此之前，要检查对战的双方是否都连接着服务器。若否，则提示玩家已经断开，无法继续游戏。

对战英雄的位置同步。双方都可以控制自己的英雄在地图上移动，所以，每当有一方移动英雄，就要把英雄的信息发给服务器，服务器再推送到所有的客户端上，在对战双方的页面上进行更新。

子弹信息的同步。若对战的一方发射子弹，那么子弹的运动轨迹要在页面上绘制出来，同样需要吧子弹信息发送到服务器，再推送到所有的客户端上。

爆炸信息的同步。若是子弹击中障碍物或者对方，会有一个爆炸效果。这里客户端会发送爆炸的对象给服务器，再推送到所有的客户端后，绘制爆炸效果。

#### 5.1.2 服务器端的事件监听

服务器端主要使用事件触发机制的几个关键方法来完成数据共享，方法如下：

监听用户是否已经连接服务器：

```
io.on('connection', function(socket){});
```

监听用户是否登陆：

```
socket.on('login', function(obj){});
```

监听用户是否退出:

```
socket.on('disconnect', function(){});
```

监听用户是否发送信息

```
socket.on('message', function(obj){});
```

向所有客户端推送信息

```
io.emit('tankInfoSym',info,invoter,beInvoter);
```

### 5.1.3 服务器端部分代码

#### (1)监听新用户加入

服务器监听 login 事件, 并将新加入用户的唯一标识当作 socket 的名称, 此标识后面退出的时候会用到。检查在线列表, 如果用户不在里面就加入到在线数组中。并将在线人数加 1, 向所有的客户端广播有新用户加入。

```
socket.on('login', function(obj){  
    socket.name = obj.userid;  
    if(!onlineUsers.hasOwnProperty(obj.userid)) {  
        onlineUsers[obj.userid] = obj.username;  
        onlineCount++;  
    }  
    io.emit('login',{onlineUsers:onlineUsers, onlineCount:onlineCount, user:obj});  
});
```

#### (2)监听用户退出

服务器监听 disconnect 事件, 如果 onlineUsers 里面无 socket.name 属性, 则表示用户已经退出。将用户从在线列表中删除, 并且将在线人数减 1,向所有的客户端广播哪位用户退出。

```
socket.on('disconnect', function(){  
    if(onlineUsers.hasOwnProperty(socket.name)) {  
        var obj = {userid:socket.name,username:onlineUsers[socket.name]};
```

```
        delete onlineUsers[socket.name];

        onlineCount--;

        io.emit('logout',{onlineUsers:onlineUsers,onlineCount:onlineCount, user:obj});

    }

});
```

### (3)监听用户发布聊天内容

服务器监听 `message` 事件，并且收到一个参数，`obj` 对象，其含有信息发送者的 `id` 与 `name` 信息。再向所有的客户端发布 `message` 的一个事件，所有的客户端都能接收到此事件，作为向所有客户端广播发布的消息功能。

```
socket.on('message', function(obj){

    io.emit('message', obj);

});
```

### (4)监听用户发布游戏邀请

服务器监听 `invotion` 事件，并且收到一个参数，`playerInfo` 对象，其含有邀请者与被邀请者的 `id` 与 `name` 信息。再向所有的客户端发布 `invotionTo+userid`（变量）的一个事件，只有目标 `userid` 的客户端才能接收到此事件，作为邀请用户一起游戏功能。

```
socket.on('invotion', function(playerInfo){

    io.emit('invotionTo'+playerInfo.beInvoter.userid, playerInfo);

});
```

### (5)监听用户是否接受游戏邀请

服务器监听 `acceptInvotion` 事件，并且收到一个参数，`playerInfo` 对象，其含有邀请者与被邀请者的 `id` 与 `name` 信息。再向所有的客户端发布 `acceptInvotion+userid`（变量）的一个事件，只有目标 `userid` 的客户端才能接收到此事件，作为接受游戏邀请功能。

```
socket.on('acceptInvotion', function(playerInfo){

    io.emit('acceptInvotion'+playerInfo.invoter.userid, playerInfo);

});
```

### (6)监听用户多人游戏是否开始

服务器监听 multiGames 事件，并且收到一个参数，playerInfo 对象，其含有邀请者与被邀请者的信息。再向所有的客户端发布 multiGamesStart 的一个事件，只有目标 userid 的客户端才能接收到此事件，作为多人游戏开始功能。

```
socket.on('multiGames', function(playerInfo){  
  
    io.emit('multiGamesStart',playerInfo);  
  
});
```

5.2 游戏引擎地图功能设计

地图模块是控制游戏难度、体现游戏设计的重要组成部分，此地图模块则是从地图元素的生成与绘制设计的。地图的选择是在双人游戏开始前进行的，此时选定好地图类型，游戏中就会加载此地图。而地图的生成是通过函数设置的。

5.2.1 障碍格、障碍块的生成设计

在代码中我们封装了两个对象，障碍格（Hamper）与障碍块（HamperBlock），障碍格是最小的障碍，障碍块由障碍格构成。

```
var block_1 = new HamperBlock(blockInfo);  
  
block_1.SetBlock();
```

代码如上所示，我们定义了一个 HamperBlock 的对象，并且传入了一个参数 blockInfo。blockInfo 是一个对象，有六个属性，分别是：

表 5.1 blockInfo 对象属性

参数	start_x	start_y	hamperStyl e	length_x	length_y	hamperNu m
注释	代 表 障 碍 块的起始 x 坐标	代 表 障 碍 块的起始 y 坐标	代 表 障 碍 块的类型	代 表 障 碍 块的长度	代 表 障 碍 块 的 高 度 (2d 平面)	hamper 数 组 的 起 始 位置

而 SetBlock()方法则是将障碍块细分为障碍格，根据起始坐标与长度、高度，定义 Hamper 对象，并将其推入到 hamper 数组内（每一类型的障碍格的 UI 都是设计好的，有着固定的长与宽），hamperNum 代表障碍块在 hamper 数组的起始位置（hamper 数组是存放地图所有障碍的数组）。

每一种地图，都是由众多的障碍格与障碍块组成的，障碍格类型、位置的不同造就了不同的地图闯关、对战难度。

### 5.2.2 障碍格、障碍块的绘制设计

每一个 Hamper 对象都存在有一个 Draw()的方法，当页面进行定时刷新的时候，浏览器会获取到 hamper 数组，将遍历 hamper 数组内的每个 Hamper 对象，并调用他们的 Draw()方法。

```
this.Draw = function() {  
    switch (this.style) {  
        case 1:  
            cxt.drawImage(brike,this.x,this.y,hamperLength,hamperHeight);  
            break;  
        case 2:  
            cxt.drawImage(ice,this.x,this.y,hamperLength, hamperHeight);  
            break;  
        case 3:  
            break;  
    }  
}
```

在进行 Draw()方法绘制的时候，我们会首先检查 hamper 数组内 Hamper 对象的类型（Hamper 对象的类型遗传自 HamperBlock 对象），比如，障碍格类型为石头的时候，我们在制定的位置绘制石头的 UI 图，是土墙的时候，绘制土墙的 UI 图。

在双人模式下，我们会同步更新地图上的障碍信息，则就是将 hamper 数组内的信息进行重新整理，去掉已经被射击破坏掉的元素，将新的数组重新发布给两边玩家，再绘制出新的同步的地图。

## 5.3 游戏引擎英雄模块功能设计

英雄模块是玩家所能直接控制的模块，是游戏引擎中与用户交互最重要的一环，用户操作的流畅、体验都是此模块性能的体现，是本次论文重点设计的模块之一。



5.3.1 英雄原形模块的功能设计

在单人模式下，会存在英雄与敌人两个角色，所以我们封装一个英雄的原形对象，在英雄原形对象的基础上创建出英雄对象与敌人对象。原型对象定义为 HeroPrototype()对象，存在六个参数，分别是：

表 5.2 HeroPrototype 对象参数

参数	x	y	speed	direct	tankColor	isLive
注释	英雄原形的起始 x 坐标	英雄原形的起始 y 坐标	英雄原形的移动单位移动速度	英雄原形的当前移动方向	英雄原形的颜色	英雄原型是否活着

HeroPrototype()对象存在四个与移动相关的方法，分别是向上移动、向下移动、向左移动、向右移动。我们以其中一个为例，来看一下设计。

```
this.moveUp = function() {
    this.y = this.y - this.speed,
    this.y <= 0 && (this.y = 0),
    this.direct = 0;
}
```

在用户控制英雄进行移动的时候，系统会首先获取到英雄的坐标分别是 x，y（画布左上角为原点）。再次根据用户按键的不同（w：向上，s：向下，a：向左，d：向右），来确定执行的函数。本例是向上，则在英雄的 y 坐标上进行一个 speed 的相减运算，再重新赋值给 this.y。并且把英雄的方向调整为向上。

5.3.2 英雄出生模块的功能设计

英雄出生模块是在游戏刚开始或者英雄死后复活时运行的，意在通过动画的效果将英雄绘制在页面上，而让玩家觉的不会显得如此突兀。

游戏引擎为英雄出生模块封装了 Born()对象，该英雄出生对象有两个参数，分别是出生区域的 x 与 y 坐标。另外还有两个私有属性，一个私有方法，分别是：

表 5.3 Born()对象属性

参数	this.time	this.born	this.drawBorn
初始值	0	0!	function() {}
注释	模块内的计数变量	是否出生	出生效果绘制

This.time 初始值为 0，为模块内的计数变量，在不同的时刻，出生动画是不一样的，每次绘制完都会执行 this.time++。所以通过判断 this.time 来控制动画。而 this.born 则是是否出生的标识，如果 this.born 为 true 则会绘制出生动画，false，则就跳过绘制。

drawBorn()方法则就是在不同的 this.time 值时，执行绘制不同的图片，我们预先定义了四个不同的出生状态，分别是 born1、born2、born3、born4 状态。页面在执行定时刷新的时候就会依次绘制此四个状态，持续两个回合，最后成功将英雄绘制出来。

```
this.time <= 1 && cxt.drawImage(born1, this.x, this.y, tankWidth, tankHeight);  
  
this.time > 1 && this.time <= 3 && cxt.drawImage(born2, this.x, this.y, tankWidth,  
tankHeight);  
  
this.time >= 4 && this.time <= 5 && cxt.drawImage(born3, this.x, this.y, tankWidth,  
tankHeight);  
  
this.time >= 6 && this.time <= 7 && cxt.drawImage(born4, this.x, this.y, tankWidth,  
tankHeight);
```

画布的 drawImage()方法就是用来绘制出生图案的函数，其五个参数分别是：

表 5.4 drawImage 方法参数

参数	born4	this.x	this.y	heroWidth	heroHeight
注释	图片对象，状态四	绘制图片的起始 x 坐标	绘制图片的起始 y 坐标	英雄对象的宽度	英雄对象的高度

5.3.3 英雄行为模块的功能设计

英雄对象（Hero）是继承自 HeroPrototype()对象,在此基础上另外定义英雄对象的两个方法,分别是英雄对象射击、英雄对象碰撞检测。英雄射击函数是英雄的本身技能之一，碰撞检测模块则是让英雄按照现实生活中的物理规则进行运动。

在英雄对象的射击状态下，我们会给英雄新建一个子弹对象：

```
var hero1bullet = new Bullet();
```

因为英雄对象方向的不同，导致射击时，子弹的位置也会不一样。所以，引擎要获取英雄对象的方向，根据方向，新建出子弹对象。当子弹对象被新建出来，那我们要让子弹“飞”起来，通过 JavaScript 的 `setInterval()` 函数进行定时执行。

```
hero1bullet.time = window.setInterval("hero1bullet.run('hero1')", 40);
```

`setInterval()` 函数会返回一个数字，作为取消定时操作的句柄，可以通过 `window.clearInterval(hero1bullet.time)` 来取消。而新建的子弹对象则会运行本身的 `run()` 函数，进行飞行。

英雄在进行移动时，每次移动都会触发英雄对象的碰撞检测函数。英雄的碰撞检测对象分两种，一种是跟敌人或者队友的碰撞检测，另外一种是与地图上的障碍物之间的碰撞检测。我们以某一障碍物为例：

```
if (hero1.x > c.x - hero.width && hero1.x < c.x + hamper.width && (hero1.y == c.y + 7 || hero1.y == c.y + 8 || hero1.y == c.y + 9 || hero1.y == c.y + 10)) return ! 0;
```

`c` 代表某一障碍物，我们会比照英雄对象的坐标与障碍物的坐标，如果有 `if` 条件中有一项不满足，比如前进一步之后，英雄的 `x` 坐标会位于障碍物之间。我们会判断英雄与障碍物在前进一步之后会发生碰撞，此时设置函数返回 `0` 来阻止英雄的继续前进。依照此方法，我们每次移动后都会检测英雄与其他所有敌人、障碍物的距离。对于不同类型的障碍物，我们在进行碰撞检测前游戏引擎会获取其障碍的类型，设置不同的碰撞检测函数，以此来满足游戏设置不同类型的障碍物。

### 5.3.4 英雄绘制模块的功能设计

英雄绘制模块主要实现的功能是将英雄在画布上绘制出来。不过，由于设计的英雄各型各样，颜色、样子都很难统一，所以游戏引擎在在模块上只是封装一个简单的绘制函数，其余的细节绘制，需要游戏的开发人员进一步补充。

### 5.3.5 英雄爆炸模块的功能设计

英雄爆炸模块主要应用于敌人击中英雄，或者英雄击中敌人会导致坦克爆炸。此模块的设计类似于出生模块。

```
var b = new TankBomb(hero1.x, hero1.y);  
tankbombs.push(b);
```

假设敌人击中英雄，则会生成一个 TankBomb 对象，复制给临时变量 b，再将 b 推入到 tankbombs 数组中。游戏引擎每次刷新会去检测 tankbombs 数组是否有数据，有的话则取出来，在画布上绘制出英雄爆炸效果图。

TankBomb 对象存在两个参数，和两个私有属性以及一个公有方法，如下：

表 5.5 TankBomb 对象属性

参数	this.x	this.y	this.time	this.isLive	this.drawBomb
初始值	无	无	0	0!	无
注释	英雄爆炸区域的 x 坐标	英雄爆炸区域的 y 坐标	爆炸动画的计数变量	英雄爆炸是否存在	绘制英雄爆炸函数

与出生模块的 drawBorn 方法类似，drawBomb()函数的部分代码如下：

```
this.time <= 2 && cxt.drawImage(bomb3, this.x, this.y, tankWidth, tankHeight);
```

主要区别在于定义了不同的图片状态变量，也就是说，我们只是简单的替换一下图片资源就可以改变爆炸的效果。

5.4 游戏引擎子弹模块功能设计

在 5.3 节中，我们设计过英雄对象的射击功能，射击的时候会新建一个子弹对象，本节我们会就子弹对象的新建、飞行、碰撞检测、爆炸做详细的设计。

5.4.1 新建子弹的功能设计

在游戏引擎中，我们封装了子弹对象，Bullet()对象。

```
hero1bullet = new Bullet(x,y,speed,direct);
```

我们来看一下 Bullet()对象的四个参数：

表 5.6 Bullet()对象属性

参数	x	y	speed	direct
注释	子弹对象的 x 坐标	子弹对象的 y 坐标	子弹对象飞行中的速度	子弹对象的飞行、生成方向

另外子弹对象还有两个私有属性，分别是 this.time 取消子弹对象刷新的句柄,this.isLive 子弹对象是否存在。子弹对象只有在用户控制射击的时候才会被新建出来。

### 5.4.2 子弹飞行的功能设计

当子弹对象被新建出来之后，按照现实，子弹会沿着既定的方向以一个既定的速度进行飞行。

```
hero1bullet.time = window.setInterval("hero1bullet.run('hero1')", 40);  
case 0:this.y = this.y - this.speed;
```

例如，每隔 40ms，子弹都会执行飞行函数，首先对子弹对象做一次碰撞检测，如果子弹对象未碰撞到任何敌人或者障碍物，引擎会获取子弹对象的方向，当子弹飞行的方向是向上的时候，会对子弹对象的 y 坐标做一个 speed 的减运算并重新赋值。如果碰撞检测结果为子弹对象坐标在障碍物或者敌人坐标范围内，则会执行新建子弹爆炸对象。

### 5.4.3 子弹爆炸的功能设计

对子弹对象进行碰撞检测时，如果子弹对象的坐标范围在障碍物或者敌人的坐标范围内，则子弹对象被设为 null，并且新建出子弹爆炸对象 bulletBomb()。

```
hero1bulletBomb = new bulletBomb(x,y);
```

构造函数 bulletBomb() 有两个参数，分别是绘制爆炸效果区域的 x 与 y 坐标。另外 bulletBomb 对象还具有一个方法：drawBomb()。此方法是用来具体的在画布上会指出爆炸的效果图。

```
var buBoImg = new Image();  
buBoImg.src = "images/blast2.gif";  
cxt.drawImage(buBoImg, this.x, this.y);
```

此方法有三个参数，分别是图片对象、绘制区域 x 坐标、绘制区域 y 坐标。buBoImg 是一个我们事先定义好的图片对象，即爆炸的效果图。这里调用画布的 drawImage() 方法就可以成功将爆炸效果图画出来。

### 5.4.3 子弹绘制的功能设计

在对子弹对象进行碰撞检测之后，如果子弹存在，我们则要在画布上重新绘制出子弹。

首先要判断子弹对象是否存在，如果存在子弹对象的 isLive 属性是否为 true，如果都满足条件，则会运行：

```
cxt.fillRect(hero1bullet.x, hero1bullet.y, bulletWidth, bulletHeight);
```

游戏引擎会调用画布的 fillRect()方法，此方法是在画布上绘制一个长方形，需要传入四个参数，分别是：

表 5.7 fillRect()方法参数

参数	hero1bullet.x	hero1bullet.y	bulletWidth	bulletHeight
注释	绘制子弹对象起始的 x 坐标	绘制子弹对象起始的 y 坐标	子弹对象的长度	子弹对象的高度

5.5 游戏引擎玩家控制模块功能设计

设计本模块主要目的是游戏引擎实时的获取玩家通过键盘或者其他外接设备对英雄的控制。我们约定好 a。当玩家通过键盘进行操作的时候，游戏引擎要操作正确的指令，并且在画布上呈现出来。

```
if (null != hero1 && 0 != hero1.islive && !gameover) {
    var a = event.keyCode || event.which;
    switch (a) {
        case 65:
            65 == lastcode ? heroTankCollision() || hero1.moveLeft() : (lastcode = 65,
hero1.direct = 3);
            CHAT.requestTankInfo();
            break;
        case 74: //J 键，发射子弹
            hero1.attackEnemy(window.heroType);
            break;
    }
}
```

当玩家在键盘上按下某一个键的时候，会触发此模块。游戏引擎会去判断英雄对象是否存在，是否活着，游戏是否结束。如果条件都满足，则会获取所按键的 keyCode，并且判断是否是 W、S、A、D 中的某一个，若是，则执行英雄碰撞检测，根据碰撞检测的结果来决定是原地不动还是向某一方向前进。如果是 J，则会执行 attackEnemy(window.heroType)方法，并且传入一个参数，即当前玩家的英雄标识。

## 5.6 游戏引擎游戏结束模块功能设计

游戏结束模块是判断当前游戏的进度，游戏是否结束，胜利还是失败，根据游戏的不同状态执行不同的动作。

游戏引擎定义了两个全局变量，分别是 `verygood` 与 `gameover`。只有当游戏胜利时，`verygood` 被设为 `true`，而 `gameover` 是初始值 `undefined`。当游戏失败的时候，`gameover` 被设为 `true`，`verygood` 为初始值 `undefined`。当游戏正在运行中，还没有结束的时候，`verygood` 与 `gameover` 都还是初始值 `undefined`。

```
if (verygood) {  
    cxt.drawImage(winImg,canvasLength/3,canvasHeight/3,image.width,image.height);  
}else{  
    if (gameover) {  
        cxt.drawImage(loseImg,canvasLength/3,canvasHeight/3,image.width,image.height);  
    }  
}
```

游戏引擎封装了 `GameOver()` 函数，此函数在画布每次刷新时进行变量 `verygood` 与 `gameover` 的检查。如果两者都是初始值 `undefined`，则函数 `GameOver()` 不执行动作。如果玩家取得胜利，则 `GameOver()` 函数会执行绘制胜利提示符，如果玩家失败，则 `GameOver()` 函数会执行绘制失败提示符。

## 5.7 本章小结

本章详细设计了平面射击类游戏引擎中的重要的几个组成模块，分别有服务器端监听模块，地图功能的模块设计，英雄功能的模块设计，子弹功能的模块设计，玩家控制功能的模块设计，游戏结束功能的模块设计。其中以服务器端监听模块，地图功能的模块设计，英雄功能的模块设计，子弹功能的模块设计最为重要，又分别从它们的具体功能入手，划分为不同的部分进行详细的逻辑、代码设计。

## 第六章 游戏引擎的功能与性能测试

本章针对课题封装的游戏引擎实例化了一个游戏，坦克大战。对坦克大战游戏的各项功能与性能进行测试。主要有玩家登陆功能、玩家即时交流功能、玩家单人游戏功能、玩家在线邀请功能、玩家双人游戏功能、玩家控制英雄功能、单人模式随机敌人功能、多人模式游戏画面同步功能。另外就是游戏引擎内，所有与逻辑处理相关的模块之间协同是否正常的功能测试。

### 6.1 测试环境的搭建

本次测试环节主要是从三种情况进行测试，本地、局域网、互联网。

本地测试搭建的环境是，一台笔记本电脑或者台式机，配置有浏览器、本地服务器、Apache 服务器。局域网测试搭建的环境是两台笔记本电脑，一个局域网，其中一台笔记本电脑配置有 Apache 服务器和本地服务器。互联网测试环境下需要两台笔记本电脑并且都可以联网，一个外网服务器空间和域名。

浏览器选择我们开发游戏引擎时所用的谷歌浏览器，安装 Node.js，并启动监听 8080 端口。安装 Apache 服务器作为静态文件访问的服务器。接下来进行具体的测试，我们先以本次测试为主，辅以局域网环境测试，真正上线之后再进行互联网环境的测试。

### 6.2 玩家登陆与即时交流功能测试

玩家在打开游戏首页之后，页面会提示玩家输入用户名，用户在输入完用户名后，点击提交进入到游戏准备页面，左侧为游戏的画布区域，右侧是玩家们的聊天室。在左侧的聊天室区域，会在聊天室顶部显示当前玩家的姓名，以及目前聊天室中所有的游戏玩家姓名，大家可以自由的在聊天室内发送消息，所有玩家都会看到。

登陆界面如下图：



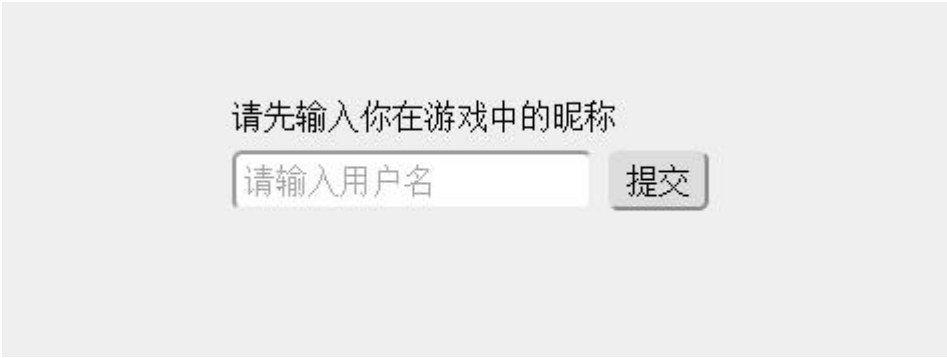


图 6.1 用户登陆界面

准备游戏界面如下图：



图 6.2 游戏准备界面

在玩家进入到游戏准备页面的时候，服务器也会把此玩家的登陆信息发布在聊天室中，所有玩家均可以看到。当玩家退出此聊天室，或者关闭游戏下线的时候，服务器也会监听到用户的行为，然后在聊天室中发布消息，某某用户退出游戏。如下图：



图 6.3 玩家在线列表

玩家在聊天室发送的内容所有人都是可以看见，并且回复，玩家 Hevid 聊天截图，玩家 Tom 聊天截图,玩家 Jary 聊天截图：



图 6.4 游戏聊天室页面 1



图 6.5 游戏聊天室页面 2



图 6.6 游戏聊天室页面 3

6.3 玩家单人模式下的功能测试

在游戏准备页面，点击单人游戏，则开始单人游戏模式。在单人游戏模式下，敌坦克自动刷新出生、移动，射击等等，我们在游戏中设定敌坦克的总数量，并且页面上敌坦克不超过五个，低于五个的时候敌坦克会自动出生，当所有的敌坦克都被消灭的时候，确认玩家取得胜利。玩家坦克则可以在画布上自由移动，完成射击功能。子弹飞行正常，击中效果正常。

## HTML5——经典坦克大战



图 6.7 单人模式游戏开始图

### 6.4 玩家双人模式下的功能测试

在双人模式下，游戏不仅要具备可以查看在线玩家并且邀请的功能，还要具备一起游戏的功能，双方的画面同步，游戏进度同步。

#### 6.4.1 双人模式邀请功能的测试

首先玩家 Tom 选择双人模式，点击双人模式按钮之后，弹出选择项，里面分别是在线玩家列表与选择地图种类，如图 6.8。在选择邀请的玩家和地图种类之后，点击邀请按钮，发出邀请函。而邀请函只有被邀请者才能收到，如图 6.9。地图种类可以不选择，开始游戏是会按照游戏引擎默认的地图进行加载。如果被邀请者接受了邀请，那么邀请者会收到一个成功邀请的回复，如图 6.10。此时整个邀请功能测试完毕。



图 6.8 玩家邀请图

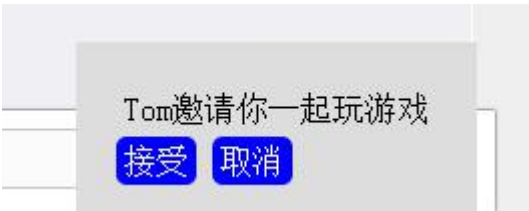


图 6.9 玩家是否接受邀请

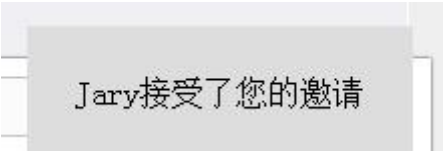


图 6.10 玩家接受邀请

6.4.2 双人模式控制功能的测试

在双人模式下，当邀请者成功邀请要到其他人一起游戏时，开始游戏的控制权在邀请者，被邀请者是无法点击开始游戏按钮的。在邀请者点击开始游戏以后，对战双方的游戏同时开始。无论哪一方的移动或者是射击都会在另一方的页面上展现出来，如图 6.11，左侧是玩家 Tom 的游戏画面，右侧是玩家 Jary 的游戏画面。



图 6.11 双人模式对战图

6.5 游戏结束的功能测试

在单人模式下，当敌坦克击中玩家的老巢，亦或是玩家消灭掉所有的敌坦克；或者在双人对战模式下，一方率先击败另一方，都会导致游戏进入结束功能，结果只有两个，胜利或者失败。游戏结束功能侧视图如下图 6.12：



图 6.12 游戏胜利图

6.6 游戏引擎文件的加载性能测试

我们通过谷歌浏览器自带的开发者工具进行调试游戏的整体加载速度，如图 6.13 所示：

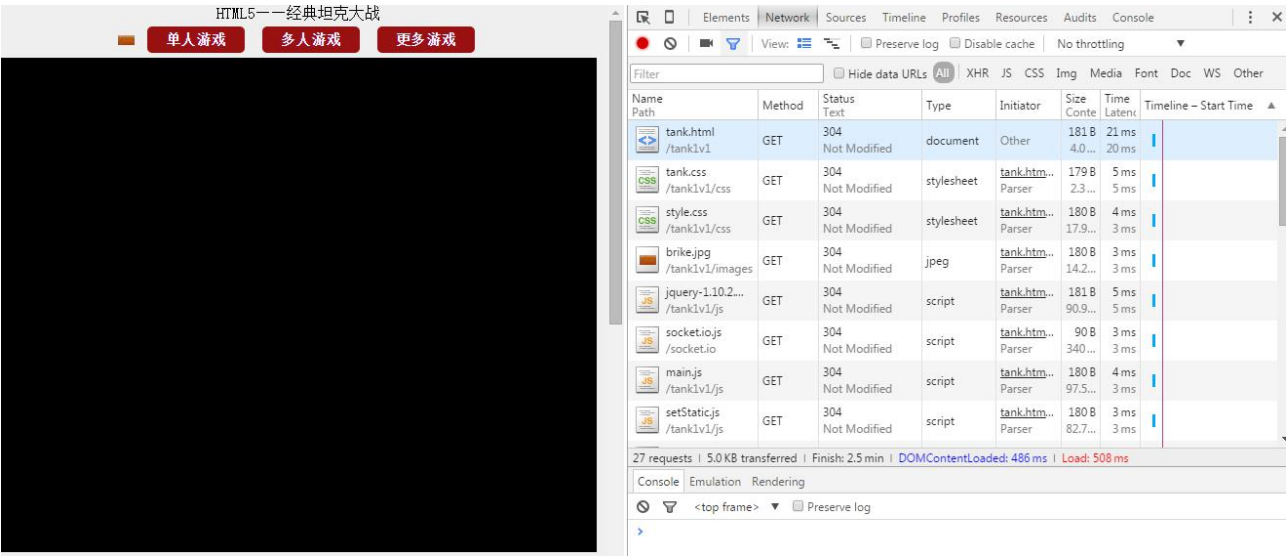


图 6.13 游戏性能测试图

从上图中我们可以看出各个文件的加载速度，整个游戏的打开耗时 508ms，在用户的可等待时间范围以内。主要的时间消耗在图片的加载时间上，所以我们可以适当压缩图片的质量，或者替换为效果相同，但是图片大小较小的。整个游戏引擎执行的时间是可以接受的。

6.7 局域网与互联网环境下的游戏性能与功能测试

在局域网下，主要测试的是双人对战模式，相互攻击对方的效果。此效果在本地测试环境下无法实现，必须用两台电脑完成。经过在局域网环境的测试下，一切游戏功能运行正常。

在互联网环境下，游戏的打开速度不稳定，我们进行了九次测试。测试结果如下表：

表 6.1 游戏速度性能测试

测试次序	1	2	3	4	5	6	7	8	9
运行速度	557ms	421ms	696ms	1134ms	856ms	505ms	1265ms	793ms	843ms

从表中的数据我们可以看到，页面打开的速度还是差别很大的。只有一次在 500ms 以内打开，在 500ms-1000ms 中打开的次数是 6 次，在 1000ms 已上的打开的次数是 2 次。其主要原因是服务器的连接速度、服务器是否稳定、当时的网络连接速度。总体上，游戏打开的时间在良好的用户体验打开时间以内。

## 6.8 本章小结

本章主要从搭建本地测试环境出发，开始逐一测试游戏实例功能的具体实现，是否完成，是否符合逻辑。其主要是登陆会话功能、从单人模式、双人模式、游戏结束功能进行具体的功能测试。又从游戏打开的速度进行分析，得出游戏引擎性能的具体测试结果。最后在局域网环境与互联网环境对游戏实例进行测试，并分析其打开时间差异性的原因。



## 第七章 总结与展望

随着移动、PC 端设备硬件的不断提升,将会更加促进 HTML5 游戏的火热。相比于其他 app 游戏、页游、端游,HTML5 游戏优势在于可以更好的利用用户的碎片化时间,更快的游戏节奏,更加人性化的打开关闭,不占用设备的太多资源,只要有一款浏览器即可。目前的游戏行业发展,HTML5 游戏已是一块赤手可热的领域。

在此游戏行业背景下,加快 HTML5 游戏开发的速度与质量,成了各个游戏开发团队必须要面临的问题。通过试玩当下较热门的 HTML5 游戏,例如《围住神经猫》《一个都不能死》《寻找房祖名》,我们可以发现,这类游戏都是单人模式的小游戏,没有玩家交流的途径,也没有多人游戏的模式,当然部分游戏在设计时也只是设计为单人游戏。本课题封装的基于 HTML5 与 Node.js 的游戏引擎,是第一款为平面射击类游戏封装的游戏引擎。可以提供其他游戏没有的实时在线交流与邀请在线玩家共同游戏功能。此游戏引擎的主要优势就是帮助游戏的开发者更快更便捷的进行开发,而不用再进行一系列的技术调研与论证。

本课题开发的游戏引擎具有如下特点:

(1) 基于 HTML5 与 Node.js,前后端开发公用一套 JavaScript 语法,减少前后端因不同语言开发而带来的学习成本、开发成本问题。

(2) 基本覆盖了基于 HTML5 平面射击类游戏的开发,游戏引擎本身封装了大量的常用模块功能,使得开发更加快速、简捷。

(3) 基于 HTML5 的 WebSocket 技术实现玩家在线实时交流与邀请在线玩家游戏等功能模块,极大的改善了现有游戏模式单一的特点。

(4) 通过本游戏引擎封装游戏实例简单快捷,主要是进行资源图片、声音等替换。游戏引擎结构易于理解。

在本次课题的具体开发过程中,也发现了此游戏引擎存在的一些问题以及需要后续改进的地方如下:

(1) 由于本课题设计的游戏引擎时针对于平面射击类的游戏,所以其他类型的 HTML5 游戏并不能通过此引擎开发。后续可以对游戏引擎再开发模块,来适应开发更多类型的平面小游戏。

(2) 在游戏引擎的具体模块代码层面,需要进一步进行代码评审与精简,目前游戏引擎文件内,存在部分代码冗余度较高,可以进一步封装成工具函数。

(3) 在游戏引擎具体的实时在线交流模块上, 目前只开发了聊天室功能, 如需用于商业开发, 则需要补充小纸条模块或者个人到个人的聊天模块。

(4) 基于 HTML5 游戏优势之一就是可以运行在移动端, 目前游戏引擎未做移动端的处理, 所以今后会增加对于移动端的适配性, 增加各 PC 端、移动端浏览器的兼容情况。

因此, 在后续的修改中, 我会就以上发现的问题继续做优化与开发, 以保证本次课题尽量打造一个高质量的游戏引擎。

## 参考文献

- [1] 黄永慧, 陈程凯. HTML5 在移动应用开发上的应用前景[J]. 计算机技术与发展, 2013.
- [2] 刘华星, 杨庚. HTML5——下一代 Web 开发标准研究[J]. 计算机技术与发展, 2011.
- [3] 2015 年 8 月全球及国内浏览器市场份额排行榜. <http://tools.yesky.com/420/93749420.shtml>, 2015.09.04.
- [4] 张卫国. 基于 HTML5 的 2D 动画的设计与实现[D]. 中山大学, 2014.
- [5] 李慧云, 何震苇, 李丽, 陆钢. HTML5 技术与应用模式研究[J]. 电信科学, 2012.
- [6] 朱文. 基于 HTML5Canvas 技术的在线图像处理方法的研究[D]. 华南理工大学, 2013.
- [7] 刘天寅. HTML5 与未来的 WEB 应用平台[J]. 阴山学刊(自然科学), 2010.
- [8] 赵剑. HTML5 动画引擎技术的研究与实现[D]. 北京邮电大学, 2014.
- [9] 董霖, 杨丁宁, 史德年. 基于 HTML5 技术的移动智能终端应用及安全问题研究[M]. 现代电信科技, 2012.
- [10] Matti Anttonen, Arto Salminen, Tommi Mikkonen, Antero Taivalsaari. Transforming the web into a real application platform: new technologies, emerging trends and missing pieces. ACM, 2011.
- [11] Hyun Lock Choo, Sanghwan Oh, Jonghun Jung, Hwankuk Kim. The Behavior-Based Analysis Techniques for HTML5 Malicious Features[J]. Innovative Mobile and Internet Services in Ubiquitous Computing, IEEE, 2015.
- [12] 李强. 基于 HTML5 的网页围棋游戏的开发和研究[D]. 北京邮电大学, 2014.
- [13] 黎志雄, 黄彦湘, 陈学中. 基于 HTML5 游戏开发的研究与实现[J]. 东莞理工学院学报, 2014.
- [14] James Teng Kin Lo, Eric Wohlstadter, Ali Mesbah. Imagen: runtime migration of browser sessions for Java Script web applications[J]. ACM, 2013.
- [15] Xianfeng Li, Zhiqiang Bao. Performance Characterization of Web Applications with HTML5 Enhancements[J]. Dependable, Autonomic and Secure Computing (DASC), 2014 IEEE 12th. 2014.
- [16] 冯科融, 王和兴, 连加美, 刘伟, 贾贺杰. 基于 HTML5 的 3D 多人网页游戏实现方案[J]. 微型机与应用, 2013.
- [17] 赵晓蕾. 基于 HTML5 的帝国战争网游的设计与部分实现[D]. 北京工业大学, 2013.
- [18] 刘再明. 国产 HTML5 游戏路在何方?[J]. 互联网周刊, 2013.
- [19] 马少雄. 基于 HTML5 的教育游戏设计与开发——以“Impact”打字游戏为例[D]. 上海师范大学, 2013.
- [20] 赵帅. 移动平台下基于 HTML5 的教育游戏研究与设计——以小学一年级数学为例[D]. 四川师范大学, 2015.
- [21] 杜宁军. 基于 HTML5 的 KDS 系统研究及实现[D]. 中国科学院大学, 2015.
- [22] 张奇伟. 基于 HTML5 的移动应用的研究与开发. 北京邮电大学, 2012.
- [23] 郑逸凡. 基于 HTML5 技术的 Web 游戏设计[J]. 计算机光盘软件与应用, 2014.
- [24] 白培发, 王阳千. HTML5 游戏开发技巧[J]. 电脑编程技巧与维护, 2012.
- [25] 郭宋. 一种支持 IOS-HTML5 规范的游戏引擎的设计与实现[D]. 吉林大学, 2013.
- [26] 郑逸凡. 基于 HTML5Canvas 的射击类游戏设计[J]. 林区教学, 2014.
- [27] 陈俊羲. 交互式广灵剪纸游戏设计研究[D]. 哈尔滨工业大学, 2015.
- [28] Hazra, T.K., Ghosh, A., Sengupta, A., Mukherjee, N. Mitigating the adversities of social media through real time tweet extraction system[J]. Computing and Communication (IEMCON), 2015 International Conference and Workshop on, IEEE, 2015.
- [29] 冯科融, 王和兴. HTML5 网页游戏分析[J]. 电脑编程技巧与维护, 2012.
- [30] Di Liu, Dong Pan. Message transport system of distributed stream computing[J]. Wavelet Active Media Technology and Information Processing (ICCWAMTIP), 2014 11th International Computer Conference on. IE

- EE, 2014.
- [31] 李雪昆. HTML5: 移动游戏的下一个机会? [N]. 中国新闻出版报, 2015.
- [32] 张伟. 基于 Web 的学生课下学习系统的设计与实现[D]. 河北师范大学, 2014.
- [33] 娄启林. 基于 Webkit 内核的桌面应用程序通用框架的构建与应用[D]. 北京交通大学, 2014.
- [34] Pavel Arapov, Michel Buffa. WikiNext, a JavaScript wiki with semantic features[J]. ACM. 2012.
- [35] Xixi Gong, Yuehui Jin, Yidong Cui, Tan Yang. Web visualization of distributed network measurement system based on HTML5[J]. Cloud Computing and Intelligent Systems (CCIS), 2012 IEEE 2nd International Conference on, IEEE, 2012.
- [36] 张少博. 双目立体网页的生成方法研究[D]. 上海交通大学, 2013.
- [37] 张露. 移动多平台跨平台开发工具集的设计与实现[D]. 华中科技大学, 2013.
- [38] 龙云. 基于 HTML5 的 WebGIS 研究[D]. 江西理工大学, 2013.
- [39] 平淑文, 潘珏羽, 张学金, 杜晓荣. 基于 HTML5 和 JavaScript 轻量级动画框架开发[J]. 计算机技术与发展, 2013.
- [40] 陈钢, 邓纯雅. H5 手机游戏迎来爆发年[J]. 中外管理, 2015.
- [41] 巴特尔. 基于 Node.js 的理财应用的设计与实现[D]. 吉林大学, 2014.
- [42] 许会元, 何利力. Node.js 的异步非阻塞 I/O 研究[J]. 工业控制计算机, 2015.
- [43] 黄扬子. 基于 Node.js 平台搭建 REST 风格 Web 服务[J]. 无线互联科技, 2015.
- [44] W3C 组织官网 [http://www.w3school.com.cn/tags/html\\_ref\\_httpmethods.asp](http://www.w3school.com.cn/tags/html_ref_httpmethods.asp).
- [45] 卢晓阳. JSON 数据交换语言在 Ajax 技术中的应用[J]. 河南科技, 2013, (20).
- [46] D. Crockford. The application/json Media Type for JavaScript Object Notation (JSON). 2006, 6.
- [47] 柯肇丰, 曾霞霞. 基于 HTML5+Node.js+MongoDB 构建在线图像编辑器系统[J]. 福建电脑, 2015.
- [48] 李志秀, 张军, 陈光. jQuery Ajax 异步处理 JSON 数据在项目管理系统中的应用[J]. 云南大学学报(自然科学版), 2011(33): 247-250.
- [49] 李兴华. 基于 WebSocket 的移动即时通信系统[D]. 重庆大学, 2013.
- [50] 荣林林. 自适应浏览器的高性能 Web 扩展应用开发系统的设计与实现[D]. 北京邮电大学, 2014.
- [51] 刘派. 基于 Node.js 的热计量监控预警系统的设计与实现[D]. 北京邮电大学, 2014.
- [52] MVC 框架. [http://baike.baidu.com/link?url=kgg5-620iaduRDEkyoixUdZK43SfUHOcy1o8iAWrbr4UPm6kXUsta0EzXvIJ1Ttc1E56vwql88mld0fSTKXP\\_1S-CW\\_0-oFOqjlkB02zYbD-0x4th5sYWPcCWdaQDwTkIW0NGCeLiQFV1PmWjseCtLqHQzNGVcna2RW0zmfW0NIX8QEFuOIeRf9sTAnLCzI1CopO9aiFozV7Oq8i-TGhxK](http://baike.baidu.com/link?url=kgg5-620iaduRDEkyoixUdZK43SfUHOcy1o8iAWrbr4UPm6kXUsta0EzXvIJ1Ttc1E56vwql88mld0fSTKXP_1S-CW_0-oFOqjlkB02zYbD-0x4th5sYWPcCWdaQDwTkIW0NGCeLiQFV1PmWjseCtLqHQzNGVcna2RW0zmfW0NIX8QEFuOIeRf9sTAnLCzI1CopO9aiFozV7Oq8i-TGhxK).
- [53] 黄丹华. Node.js 开发实战详解[M]. 北京: 清华大学出版社, 2014. 04.
- [54] Tom Hughes-Croucher, Mike Wilson 著 郑达韡译. Node 即学即用: up and running, scalable server-side code with JavaScript[M]. 北京: 人民邮电出版社, 2013.
- [55] 陆凌牛. HTML 5 与 CSS 3 权威指南[M]. 北京: 机械工业出版社, 2013.
- [56] 肖在昌, 杨文晖, 刘兵. 基于 WebSocket 的实时技术[J]. 电脑与电信, 2012.

## 致谢

不知不觉间又到了说再见的时候了，对所有敬爱的老师，对所有亲爱的同学，对母校南京邮电大学。依然清晰的记得三年前的那场毕业，转眼间，又到了毕业，这次是真的毕业了。在南京邮电大学的这三年里，我学会了很多事情，比如确立了自己的研究方向、找到了适合自己就业的方向、找到了一份比较满意的工作。在刚来到南京邮电大学时，我是一片迷茫的。在贴心的辅导员、耐心的导师、热情的师兄师姐的帮助下，自己也慢慢的融入了这个校园。那时，一切都是新鲜的，一切都是欣喜的，一切都是值得期待的。正像现在，未来很值得期待！

首先我要感谢我的导师曹士珂教授。从一个懵懂入学的研究生，到现在即将进入公司工作的战士，一路走来，我的成长离不开曹教授的谆谆教诲。无论是从研究生学习过程中理论知识的储备，还是对于研究生期间实习岗位与公司的建议，亦或是对研究生毕业论文的指导，曹老师的关心总是无微不至。人生得一知己足以，但是现在，我想说，人生得一如此导师，亦足矣。对于自己今天取得的成绩，再次向曹老师表示感谢。

其次，我要向南京北界软件有限公司、江苏金智教育信息股份有限公司表示感谢。正是在贵公司的实习期间，我渐渐了解到了前端，并且喜欢上前端，并决定以此为职业。也正是在这一期间，我掌握了前端开发的技术，HTML5、CSS、JavaScript、Node.js。正是基于上述技术的学习，我才能顺利的完成毕业论文的设计。感谢两个公司为我提供的实习机会，不仅让我学习到了工作技能，也让我真正参与到了团队合作当中，培养了我发现问题、解决问题的能力，以及为人处事的技巧等等。

再次，对一直在背后默默付出的父亲母亲表示感谢。是你们的疼爱在鼓励着我一步步前行，曾经也有过很多心灰意冷想要放弃的时候，是你们的温暖驱散我心中的雾霾，让我重新鼓起勇气面对。也感谢关心、帮助我的所有的朋友同学，是你们在我困难时，及时给我帮助与支持，我才能克服一个又一个的问题。

最后，感谢各位专家、教授在百忙之中抽空审阅我的论文及参加我的毕业论文答辩，感谢你们为我的论文提出的宝贵建议，我一定听取各位老师的意见，认真修改，达到要求。各位老师，你们辛苦了！

祝大家身体健康，工作顺利。