

中国计量大学现代科技学院

本科毕业设计（论文）

基于 layabox 的微信端魔塔小游戏开发

**Development of magic tower game on
wechat based on layabox**

学生姓名朱铸杰 学号1630332228

学生专业计算机科学与技术 班级计算机 162

系（部）信息工程 指导教师黄 俊

中国计量大学现代科技学院

2020 年 6 月

郑重声明

本人呈交的毕业设计论文，是在导师的指导下，独立进行研究工作所取得的成果，所有数据、图片资料真实可靠。尽我所知，除文中已经注明引用的内容外，本学位论文的研究成果不包含他人享有著作权的内容。对本论文所涉及的研究工作做出贡献的其他个人和集体，均已在文中以明确的方式标明。本学位论文的知识产权归属于培养单位。

学生签名：_____

日期：_____

分类号: TP311.1
UDC: 004

密 级: 公开
学校代码: 13292

中国计量大学现代科技学院 本科毕业设计（论文）

基于 layabox 的微信端魔塔小游戏开发

**Development of magic tower game on
wechat based on layabox**

作 者 朱铸杰 学 号 1630332228

申请学位 工学学士 指导教师 黄 俊

学科专业 计算机科学与技术 培养单位 中国计量大学现代科技学院

答辩委员会主席 黄 俊 评 阅 人 陈光平

2020 年 6 月

致 谢

大学生活一晃而过，回首走过的岁月，心中倍感充实，当我写完这篇毕业论文的时候，有一种如释重负的感觉，感慨良多。

首先诚挚的感谢我的毕业设计教师黄俊老师。他在忙碌的教学工作中挤出时间来审查、修改我的论文。还要感谢教过我的所有老师们，你们严谨细致、一丝不苟的作风一直是我工作、学习中的榜样；他们循循善诱的教导和不拘一格的思路给予我无尽的启迪。感谢四年中陪伴在我身边的同学、朋友、感谢他们为我提出的有益的建议和意见，有了他们的支持、鼓励和帮助，我才能充实的度过了四年的学习生活。

基于 layabox 的微信端魔塔小游戏开发

摘要：在社会发展的今天，人们对于物质上的追求已经基本满足，开始寻求更多精神上的追求。疫情当下，游戏这一娱乐模式逐渐成为人们习惯的一种行为方式。而相对于大型网络游戏与普通 APP 手游的需要长期积累升级，下载后才能进行游戏的模式，小游戏体验轻便，无需下载，寄托于客户端的方式无疑为玩家开辟了一条新的快捷的方式，自微信小游戏推出后，头条，V0 快应用等各大厂商紧随其后，掀起了中国全新 5G 时代下游戏的新浪潮，也是 H5 取代 Flash 游戏的大势所趋。本文旨在借助微信平台环境下，利用 laya 引擎，开发一套改编后的传统游戏—魔塔。

目前国内主流的 H5 游戏引擎主要有 Cocos2d-js，Egret 与 Layabox。其中，Cocos2d-js 在语言上仅支持 js，且不支持 3D 渲染与 VR 发布，对日后向 3D 转换带来不便，而 Layabobx 在性能渲染上的综合实力较强，且内部支持多渠道游戏的发布。本文就是采用 Layabox 引擎下支持的 TypeScript 来设计的小游戏，利用 FairyGUI 进行游戏内部界面的设计，利用 xlsxConvert 工具将写在 xlsx 表中的游戏数据转换成 json 格式，利用 Web Storage（web 端），setStorage、getStorage（微信端）进行在玩家客户端中进行读取、存储玩家的基础数据。通过监听与取消监听相结合的方式进行游戏中的摇杆设计来操作角色移动，游戏地图采用数组的方式来记录，在玩家数据中存入各个地图的道具，怪物等信息，移动时进行事件判断。核心玩法为玩家与各种怪物战斗，战斗胜利后获得经验与金币提升自身属性，最后击杀最终 BOSS 取得胜利营救公主。最终完成的功能有地图怪物设计绘制，摇杆操作系统，地图跳跃系统，金币商城经验商城系统，角色与各种不同的属性类的碰撞事件系统，背包系统，战斗系统，数据存储系统。

关键词：智力游戏;魔塔;微信小游戏;LayaBox

中图分类号：TP311.1

Development of magic tower game on wechat based on layabox

Abstract:In today's social development, people's pursuit of material has been basically satisfied, and began to seek more spiritual pursuit. At present, game as an entertainment mode has gradually become a way of behavior that people are used to. Compared with the large-scale online games and ordinary app mobile games, which need to be accumulated and upgraded for a long time, the game mode can only be carried out after downloading. The small game experience is light and does not need to be downloaded. The way entrusted to the client undoubtedly opens up a new and fast way for the players. Since the launch of wechat small game, the top news, VO fast application and other major manufacturers have followed closely, setting off a new 5g era in China The new wave of game is also the trend of H5 replacing flash game. This paper aims to develop a set of adapted traditional game magic tower by using the Laya engine under the wechat platform environment.

At present, the mainstream H5 game engines in China mainly include cocos2d JS, egret and layabox. Among them, cocos2d JS only supports JS in language, and does not support 3D rendering and VR publishing, which is inconvenient for future 3D conversion, while layabobx has strong comprehensive strength in performance rendering, and internal support for multi-channel game publishing. This paper is a small game designed by typescript supported by layabox engine. Fairygui is used to design the internal interface of the game. Xlsxconvert tool is used to convert the game data written in xlsx table into JSON format. Web storage (web side), setstorage and getstorage (wechat side) are used to read and store the basic data of the player in the player client. Through the combination of monitoring and canceling monitoring, the rocker design in the game is designed to operate the role movement. The game map is recorded in the way of array, and the props, monsters and other information of each map are stored in the player's data, and the event judgment is carried out when moving. The core play method is that players fight with various monsters, gain experience and gold coins to improve their own attributes after winning the battle, and finally kill the boss to win and rescue the princess. Final functions include map Monster Design and drawing, rocker operating system, map jumping system, gold coin shopping mall experience shopping mall system, collision event system of roles and various attribute classes, backpack system, combat system and data storage system.

Keywords:Intellectual games;Magic tower;Wechat games;LayaBox

Classification: TP311.1

目 次

摘要.....	I
目次.....	III
1 绪 论.....	1
1.1 研究背景及意义.....	1
1.2 国内外研究现状.....	1
1.3 本文主要工作.....	2
1.4 本文组织结构.....	3
2 游戏的组成、玩家属性分析.....	4
2.1 游戏组成.....	4
2.1.1 机制 (Mechanics)	4
2.1.2 故事 (Story)	4
2.1.3 美学 (Esthetics)	5
2.1.4 技术 (Technology)	5
2.2 玩家属性分析.....	5
2.2.1 方块——成就型玩家.....	5
2.2.2 黑桃——探索型玩家.....	6
2.2.3 红桃——社交型玩家.....	6
2.2.4 梅花——杀手型玩家.....	7
2.2.5 总结.....	7
3 游戏分析与设计.....	8
3.1 需求分析.....	8
3.1.1 功能需求分析.....	8
3.1.2 非功能需求.....	9
3.2 可行性分析.....	9
3.2.1 技术可行性分析.....	9
3.2.2 经济可行性分析.....	9
3.3 游戏的总设计.....	9
3.4 功能模块设计.....	11
3.4.1 游戏存储模块.....	11
3.4.2 游戏分享模块.....	11
3.4.3 操作模块.....	11
3.4.4 界面模块.....	12
3.4.5 商城模块.....	12
3.4.6 碰撞模块.....	12
3.4.7 战斗模块.....	12
3.4.8 跳跃模块.....	12
3.4.9 音乐模块.....	12
3.4.10 属性查看模块.....	12
3.4.11 背包模块.....	12

3.5	xlsx 表设计.....	13
4	系统实现.....	15
4.1	数据初始化.....	15
4.2	功能模块实现.....	15
4.2.1	游戏存储模块.....	15
4.2.2	游戏分享模块.....	16
4.2.3	操作模块.....	17
4.2.4	界面模块.....	18
4.2.5	商城模块.....	18
4.2.6	碰撞模块.....	21
4.2.7	战斗模块.....	22
4.2.8	跳跃模块.....	23
4.2.9	音乐模块.....	25
4.2.10	属性查看模块.....	21
4.2.11	背包模块.....	22
5	游戏测试.....	27
5.1	软件测试的重要性和目的.....	27
5.2	基于 layabox 的微信端魔塔小游戏功能测试.....	27
5.2.1	游戏外部功能测试.....	27
5.2.2	游戏内部功能测试.....	28
6	总结.....	31
	参考文献.....	32
	作者简历.....	33
	学位论文数据集.....	34

1 绪 论

1.1 研究背景及意义

随着 5G 时代的发展，一种无需下载，即玩即走的游戏模式——微信小游戏模式越来越起到举足轻重的地步。这是一个小游戏高速发展的时代，自从微信推出第一款小游戏“跳一跳”之后，短短一个月内吸引了 3.1 亿玩家，远超市场上其它游戏行业的平均水平。而在微信正式向第三方开发者开放后，越来越多的经典游戏开始接入微信渠道。而魔塔小游戏正是我们这一代人儿时一大经典的游戏。

魔塔最早由两位日本人所做，传入中国后由胖老鼠制作了 21 层魔塔与 24 层魔塔，这正是大多数人所接触的第一款魔塔，之后由 0ksh 制作了 RMXF 用的魔塔样板，随之，越来越多的人参与制作了魔塔，魔塔种类也越来越多。但在制作技术上没有跟进，反而因为使用了样板的原因，制作的技术成本大大降低。而由于技术不更新换代的原因，魔塔游戏已经逐渐跟不上微信小游戏的版本。

本文旨在使用新的 H5 技术重构老的 Flash 制作的魔塔，并在其中的战斗数据增加了暴击率命中率闪避率来增加游戏的可玩性，给玩家增加一个背包系统来增加玩家的游戏体验。

1.2 国内外研究现状

2017 年底，微信更新的 6.6.1 版本开放了小游戏，跳一跳小游戏横空出世，拉开了中国市场下 H5 小游戏取代 Flash 小游戏的序幕。正是因为小游戏的投入成本小、开发周期短、技术门槛低等优势，对于中小型开发公司而言十分有吸引力。而微信小游戏的模式，一种无需下载点击即玩的手机游戏，玩家无需像 MMORPG 与其它网游一样长时间在游戏中也能取得较大的游戏乐趣，也非常适合学生党与办公党，而这部分用户具备一定的消费水平，在中国市场的潜力巨大。目前微信小游戏的大小最高仅支持 8M，开屏加载速度快，微信自带具备分享功能，微信用户数量大，因此易于传播。微信自带的商业广告模式，为开发者直接带来收益，开发者只需关注自身的游戏内容，无需联系其它商业渠道，接入模式十分容易，可在游戏中添加 Banner 广告、插屏广告与激励视频广告等为开发者带来收益。此外，除了微信渠道，开发者还可以接入头条、抖音、OV 快应用、手 Q、小米、UC 等多种渠道，一款游戏多次收益，真正为开发者带来便利。

相比国内的小游戏环境，国外的小游戏开发者并没有国内多渠道的小游戏带来的收益，也没有平台自带的商业模式接入，需要开发者自身去找商业价值来为自身盈利，但国外小游戏的开发者大多不以此为盈利，甚至以开发游戏为副业。对比于国内以商业价值为主的游戏模式，国外的开发者更多关注游戏机制与玩家体验，制作的游戏也

颇为精良，但由于受限于游戏收益不大，后期维护成本过高，多数开发者仅仅制作了最初的游戏模型就放弃了，许多优秀的游戏因此没能在国内爆火。国内许多游戏厂商也会在各大国外的游戏网址寻找这一批游戏游戏来重构并维护。

我国游戏的发展远晚于国外，但在近几年的发展速度愈发迅速，特别是小游戏模式。微信自带的用户数量庞大，自身主动为小游戏引流，为开发者建立了一个良好的环境。但是由于国内的一些血腥暴力的内容被和谐，在游戏的内容表现力上不如国外，另一方面，国内的游戏厂商更加注重盈利，往往会在游戏中增加一些人民币的礼包来增强玩家的游戏属性，这种做法破坏了游戏的公平性，游戏制作的质量与国外相比还有很大的差距，需要不断努力提升，争取早日打开国内游戏向国外市场输出的渠道，在全球游戏市场占领一席之地。

1.3 本文主要工作

本课题实现基于 LayaBox 的微信端魔塔小游戏。小游戏大致划分七个模块功能，分别为地图模块，摇杆操作模块，楼层跳转模块，碰撞模块，战斗模块，商城交易模块，背包模块。

地图模块又细分为普通墙路地图，NPC 地图，道具地图，可拆卸的墙门地图，怪物地图 5 种不同的地图。地图采用数组方式记录，每块地图上必须拥有普通墙路的地图，且可以拥有一种其它类型的地图构成。在游戏过程中，需要对玩家现在的地图进行存储来比对原本绘制的地图信息来判断玩家进行到了哪一步。

摇杆操作模块是玩家操作角色移动的核心，通过监听玩家的触屏事件来判断玩家移动的事件，保障玩家移动的合理性。

楼层跳转模块是玩家在后期快速移动的关键，由于 21 层的设计，来回可能过于麻烦，可以通过该模块进行跳转，同时需要注意跳转后的玩家的初始位置。

碰撞模块是玩家体验的核心，玩家在行动过程会存在各种事件，如拾取道具，遭遇怪物，触碰 NPC，触碰商店，触碰到不可移动的墙壁等，在玩家移动之前，先要进行游戏内的判断。如触碰怪物则触发战斗模块。

战斗模块是游戏结算的核心，玩家与怪物拥有攻击防御命中暴击回避等属性，同时拥有生命值，玩家生命值耗尽，则游戏失败结束，最终 BOSS 生命值归零则玩家通关游戏。战斗采用回合制，由于回避暴击等属性的存在，大大增强了游戏的不确定性与可玩性。

商城交易模块是玩家对自身属性提升的关键所在，游戏内设有金币、经验两种提升手段，金币可以在商城中对玩家属性进行提升或购买钥匙，经验可以在大师 NPC 中对玩家属性进行提升。

背包模块是本次新增的魔塔模块，与传统魔塔中直接使用道具的方法不同，玩家在拾取道具后，可存入背包之后使用，一方面节省了故意不捡血瓶而来回穿梭楼层的

时间，另一方面增加了一个玩家可操作的内容，增加游戏玩法。

整个游戏的难点在于多种事件的判断与事件结束后的地图刷新机制，游戏地图采用列表模式刷新，一层一个数组，拥有 121 个元素，每个元素中有两个地图属性。事件分为拾取道具，怪物战斗，对话 NPC，打开商城等，不同事件对于不同的回调。部分事件需要销毁地图上的元素，同一种事件也存在不同回调情况，可能存在回调动画等多种情况。需要对各种不同情况进行不同处理，本文后续将详细描述各种事件的回调情况。

1.4 本文组织结构

本文共分为 6 章。

第一章主要介绍了微信魔塔游戏的背景和意义，以及国内外研究的现状，同时讲述了本文研究的内容和工作。

第二章则是介绍了有关于游戏组成，玩家属性的研究，包含游戏机制，游戏故事，游戏美学与游戏技术，玩家属性可简单分为成就型玩家、探索型玩家、社交型玩家与杀手型玩家。

第三章对游戏进行了分析和设计，其中有需求分析和可行性分析，最后讲解了游戏的总设计流程，11 个模块的功能设计和 xlsx 表格设计。

第四章主要是阐述了游戏的实现，有数据的初始化服务，11 个模块的具体实现和实际效果图，当然也包括各种事件的回调处理与刷新地图的逻辑。

第五章主要是对整个游戏的所有模块进行测试，同时也提到了测试的重要性的目的。

第六章是对本文所研究的基于 Layabox 的微信端魔塔小游戏进行总结。

2 游戏的组成、玩家属性分析

2.1 游戏组成

在 Jesse Schell 所著的《游戏设计艺术》中，将游戏的组成归类为了四大基本元素。分别是：机制、故事、美学和技术。

2.1.1 机制（Mechanics）

什么是游戏机制。游戏机制，英文名为 Game Mechanics。机制是指游戏中的过程和规则。它详细的描述了玩家应该怎样才能完成你的游戏的目标，当他们尝试的时候会发生什么。游戏机制是游戏真正的核心。是剥离了故事、美学与技术之后，剩下的互动和关系，就是游戏机制。简而言之，就是游戏中玩家用于交互与游戏世界的基础功能，是一种游戏规则，但是，这种规则却并不像其它商品一样如同印在说明书上一般，对于玩家来说，游戏的机制很多时候是隐藏的，需要玩家自己去发现。而对一个游戏而言最重要的规则就是核心机制，比如说本文中的魔塔小游戏，战斗就是核心机制，但是获取道具购买属性也是机制，移动角色是机制，对话 NPC，楼层跳转等也都是机制，任何游戏都是由许多的机制组成的。就如同最早的游戏《太空侵略者》一般，游戏都具有突破性质的机制，在《太空侵略者》中，随着消灭敌人的数量越来越多，敌人的前进速度也越来越快，这让游戏更加兴奋，增加了紧张感。而在魔塔游戏中，随着楼层的升高，怪物的属性越来越高，玩家需要不断提升自身属性去挑战更高更难怪物，这正是魔塔游戏的突破性质的机制。

游戏机制和游戏的美术和故事完全无关，哪怕是刨除游戏的所以背脊设定和视觉元素，游戏机制依然在那里，忠实地提供者可玩性，有时候哪怕设定完全不同，但游戏机制是完全一样的，虽然游戏机制是游戏可玩性的来源，游戏机制做得好就是游戏设计的好，但是这并不是说只要你游戏设计好就是一款好游戏，也并不是说如果游戏设计的差游戏就会很失败，更多时候，爆款的游戏并没有出色的游戏设计，很多的让游戏好玩或者上瘾的地方和现有体系下的游戏设计的理解完全无关，甚至完全是不经意中形成的。游戏的制作和定义目前在血书上还是一个很初级的阶段，并不能够非常公式化的制作好的游戏，游戏是一门很复杂的学科，同样的机制，套上不同的设定和包装会有截然不同的游戏体验，有些游戏发明了一个新的机制，但是这个机制却在另一些游戏上才被记住。一个在纸面分析上极其合理的机制和玩法，却有可能因为对于玩家心理学上的认知误区甚至引导的不完善而被玩家讨厌。

2.1.2 故事（Story）

在目前的游戏行业中，并没有一定要求一个游戏必须从游戏机制出发，有些游戏设计师是最先先写好了游戏的故事脚本，以叙事为游戏的重心去展开项目，像橙光游

戏，则是完全以叙事的方式去展现一个游戏。对于任何一个游戏来说，它都有自己的背景故事。一个好的背景故事，可以吸引玩家，激发玩家的探求欲，推进游戏发展。

在魔塔游戏中，勇士为了营救公主，进入了魔塔，在刚进魔塔之时，就被魔塔的首领暗算，被夺取了武器，只能在塔里不断打怪升级，在经历了一番战斗后，勇士变得更加强大，在无路可走的情况下，小偷打通了一条密道，和勇士一起逃出了监牢，并打败了魔王，救出了被囚禁的公主。

2.1.3 美学 (Esthetics)

当然也有一些游戏设计师一开始只敲定了美术设计风格，试图以视觉渲染为主来展开游戏，甚至用一张概念原图铺开游戏设定来开始。而游戏的背景音乐也是游戏烘托气氛的关键。

魔塔游戏的美术风格是像素风且偏阴暗的，这是为了配合游戏的故事而设定的美学，音乐也同样是偏暗系风格。

2.1.4 技术 (Technology)

技术是让游戏变得可行的材料和交互手段。本文使用的 Layabox 引擎下的 TypeScript 语言。TypeScript 是一种面向对象的高级脚本语言，在项目开发管理中与项目开发的工具环境的成熟度都明显优于 JavaScript 脚本语言。另一方面，性能是 H5 游戏的关键所在，下图中是各大引擎的性能对比，可以看出来，Layabox 在性能的综合实力非常强，这也是我选择使用 Layabox 引擎的关键所在。

引擎	2D性能 (Canvas)	2D性能 (webGL)	3D性能 (webGL)	runtime性能
Pixi.js	不支持	★★★★★	不支持	不支持
Three.js	不支持	不支持	★★★★★	不支持
PlayCanvas	不支持	不支持	★★★★★	不支持
Layabox	★★★★★	★★★★★	★★★★★	★★★★★
Egret	★★★★☆	★★★★☆	★★★★☆	★★★★☆
Cocos2d-js	★★★	★★★	不支持	★★★★★
Hilo	★★	★★	不支持	不支持

图 2.1 各大引擎的性能对比

2.2 玩家属性分析

理查德巴特尔在 1996 年发表了一篇名为《牌上的花色——MUD 中的玩家》的论文，这篇论文将 MUD 游戏中玩家的行为用扑克牌的四种花色分成了四种基本类别，而这种基本分类也成为了目前玩家分类理论的基础，尽管因为这个模型因为过于二元化和简单，并不是一个完善的玩家分类模型，后来也出现了比如 Marczewski 模型之类的更复杂的分类方法，风法完善了玩家分类的拓扑图谱，但是巴特模型始终以其简单和广泛性受到了游戏制作者的欢迎和认可。

2.2.1 方块——成就型玩家

这些玩家主要关注在游戏中达成某些特定的目标和成就，比如说通关和满级就是最基本的目标之一。哪怕他们并不享受他们正在玩的游戏，但他们就是想通关，很多

硬核的成就型玩家更愿意为了达成特定的成就，会愿意反复性的刷一些毫无趣味性可言的游戏内容，他们更多不在意游戏过程有多无聊或多困难，有时更加相反，这过程越无聊越困难，在达成成就之后的成就感就越强。游戏设计的本质之一就是给与玩家游玩的成就感，哪怕游戏中没有设置这种成就属性的类型，他们也会想办法迫害他们自己。对于他们而言，可能玩游戏本身的乐趣并不重要，重要的是他们在玩游戏以后达成的成就，重要的是完成目标的那一瞬间的满足感，而成就型玩家的终击形态就是速通玩家，第一次通关是为了玩，第二次通关可能是为了挑战或是回味，而第三第四次反复通关，那完全就是为了成就感，这就是成就型玩家。

在魔塔游戏中，玩家在许多论坛贴吧上都会晒上自己的通关属性或者通关时间，而通关属性越高，通关时间越短，他们获得的关注量就越高，获得的成就感就越强。而成就型玩家会想法设法反复通关争取更高的通关属性，更短的通关时间来给自身带来成就感。

2.2.2 黑桃——探索型玩家

这是一群有着天生探索欲的玩家，他们总是希望在游戏中探索新东西，他们希望了解游戏内外的故事，探索有关这个游戏的一切，对于这些玩家来说，很多时候游戏的乐趣就是探索的乐趣，有关探索的过程和结果，他们会探索游戏的每一个房间和每一个 NPC 对话，了解他们的故事，他们热衷于寻找游戏的暗线剧情，对游戏进行剖析和研究，研究哪些装备能最大化他们的利益，尝试不同的打法，搭配不同的组合套路，有时候很多东西也不是他们自己找到的，他们可能只是看了一下攻略，然后知道了这个游戏还能这么玩，然后自己去试了一下，会感觉到开心和新奇。很多速通类玩家是成就型玩家的同时也是探索型玩家，他们需要找到新的路线或 BUG，而这些都需要对游戏机制和关卡本身的探索，支撑这种玩家的往往是植根于人类 DNA 里的探索欲和发现欲，这种发现欲让这些玩家费尽心血，要玩完他们喜欢游戏的每一个角落，那么这就是探索型玩家。

在魔塔游戏中，在每一层有一种不同的新的道具与各种不同的 NPC，吸引着探索型玩家不断继续探索这个游戏。而魔塔游戏的故事背景也是版本众多，有传统的英雄打败恶魔救公主的故事，也有恶魔故意吸引有实力的勇士来挑战自己的反向逻辑版本。NPC 的各种对话也都暴露了自身一些性格，吸引玩家去遐想这背后的故事。

2.2.3 红桃——社交型玩家

一群很有爱的玩家类型，顾名思义，这类玩家就是一些喜欢交朋友喜欢在游戏中构建人与人关系的玩家们，因为这些玩家爱交朋友的特性，无论是怎么样的玩家，都应该或多或少认识一两个社交型玩家，他们在游戏里啥也不干，就在 YY 里或游戏聊天工具中和公会里的兄弟聊天，这就是社交型玩家。

像魔塔这类单机游戏，游戏内部并没有社交功能，但是玩家自发性的组织了魔塔

吧，游戏论坛等，为社交型玩家带来了一种社交渠道，他们可能并没有玩过这款游戏，但长时间在这些地方活跃，最终会有一部分玩家向其它类型转换，变成真正的玩家群体。

2.2.4 梅花——杀手型玩家

可以说是目前最普遍的一种玩家类型了，对于这类玩家来说，就是喜欢把自己的快乐建立在他人的痛苦之上，是热衷于 PVP 的一些人，他们相信与人斗其乐无穷的道理，对于他们来说最大的爽快感就是胜利，且这种胜利必须建立在另一个玩家的失败之上，他们才能感觉到成就感，他们喜欢与人博弈。本文设计的魔塔游戏不涉及到此类玩家，在此只做简单的介绍。

2.2.5 总结

这只是一种最宽泛的，最普世的玩家模型，下图能很好的概括巴特尔模型，X 轴从左到右是玩家的世界，Y 轴从上到下是交互于和作用于，所以方块成就型玩家倾向于作用于游戏世界的玩家，黑桃探索型玩家倾向于世界，红桃社交型玩家倾向于交互于其它玩家，梅花杀手型玩家倾向于作用于其他玩家。在游戏设计的基本应用层面上，绝大部分的情况这种四分法完全够用了。其实很多玩家都会觉得自己起码有两到三种混合的玩家类型，会享受各种不同游戏的不同成就感。



图 2.2 巴特尔模型

3 游戏分析与设计

3.1 需求分析

3.1.1 功能需求分析

本游戏的主要功能需求分为六块，分别为摇杆操作，碰撞，楼层跳转，背包设计，战斗，商城交易。

1) 摇杆操作

一个游戏最基础的是移动角色的位置，本次设计手机端魔塔游戏采用传统摇杆的操作模式。摇杆需要实时监听玩家点下摇杆盘的事件，并判断摇杆盘的移动方向，在按下后才开始实时监听玩家的点住事件，并在没有离开摇杆盘或手指弹起前，不断判断手指在摇杆盘的哪部分位置，进行角色的移动，在离开摇杆盘或手指弹起后，需要取消监听玩家的点住事件。另一方面，由于监听事件的频率比行走频率要高太多，在一次行走完成之前，再次监听到行走事件必须要打断，否则会出现角色错位等问题。以行走事件为基础，玩家才能在游戏中进行其它操作。而一些事件触发后，如战斗，购买道具等，又需要不能触发行走事件，否则也会出现多次触发事件的错误。

2) 碰撞

玩家行走过程中，会碰撞上各种不同的种类，碰撞上墙壁，则无法继续行走，碰撞上道具，则拾取道具增加属性或存入背包，碰撞上 NPC，有对话，则触发对话机制，碰撞上怪物，则触发战斗系统。在碰撞之后，需要根据不同的属性进行判断地图上的元素是否需要被销毁并保存。

3) 楼层跳转

楼层跳转是玩家在拾取到跳转工具后获得是一个技能，可以在无事件处理时使用，跳转到各个楼层，而不用反复行走于楼层之间，给用户更好的游戏体验。

4) 背包设计

背包是本次系统新增的一种玩法，在传统魔塔中，玩家可能会为了存血而放弃前期的一些血瓶后续回头拾取，本次增加背包就是为了避免这种情况的发生，让玩家节省时间。同时，在玩家拾取到可存入背包的道具时，有动画，并随着动画的轨迹存入背包。

5) 战斗

战斗是核心玩法，本次增加了闪避暴击回避等战斗属性，为了给玩家带来更多的不确定性，增加游戏可玩性，每个怪的属性均存在表格数据中，需要用到时这条属性时调用读取，战斗结束若胜利，则清除怪物在地图上的位置。

6) 商城交易

金币与经验两种交易途径，均能从战斗获得。其中，经验只能在商店消耗，不能反向使用属性兑换经验；但金币购买的钥匙，以及在战斗路途中获得的钥匙，均可以在商店内低价出售，给玩家更高的策略性。

3.1.2 非功能需求

首先是各机型的适配问题，现在手机的类型非常多，分辨率各不相同，但需要一次发布兼容市面上大部分可使用该小游戏的手机。

其次是小游戏发布后的维护性。后续如果有新增内容，在游戏进入之前，可以进行判断用户属性，如果用户是老用户，则需要将新的内容存入用户数据中，如果用户是新用户，则调用新的用户创建接口，最终将两种用户类型统一合并为同一种且不影响原有的功能。

最后是安全性，要求在任何时刻都不能泄露玩家的信息，让玩家放心进行游玩。

3.2 可行性分析

3.2.1 技术可行性分析

本次魔塔游戏采用的是 Layabox 引擎，上手简单，在官网中拥有丰富的教程与案例，可供开发者学习使用。Laya 项目可由 Laya 自带的编辑器进行调试运行，也可以根据教程自行设置配置，在不打开 ide 的情况下也能使用 Laya 环境进行编译调试。

项目大部分的参数设计在 xlsx 表中，即使是不懂代码的人员，也可以参与制作到游戏之中。在项目中可使用 xlsxConvert 工具，经配置后可将表中的数据自动转换为 json 格式，供开发人员使用。

Laya 项目在发布导出时，可选择转化为微信小游戏，发布时可设置分包与引擎插件，配合使用微信提供的微信开发者工具，在官网阅读后，可轻松将 Laya 项目转换为微信可使用的微信小游戏。注意：微信小游戏暂最高支持单包 4M，整包 8M，部分资源若过大，可以在不影响游戏质量的情况下进行压缩处理。

上述分析表明从技术上来说开发微信端魔塔小游戏完全是可行的。

3.2.2 经济可行性分析

本项目所有使用到的引擎、技术及开发工具都是开源的，不需要额外付费。同时基本是属于腾讯公司，搜游公司，花谷软件，在开源的同时性能仍很强悍，所以在经济方面也是可行的。

3.3 游戏的总设计

从下面这张流程图发现，实现了游戏的整体设计，包含了从角色建号开始直到游戏失败或胜利的全部流程。

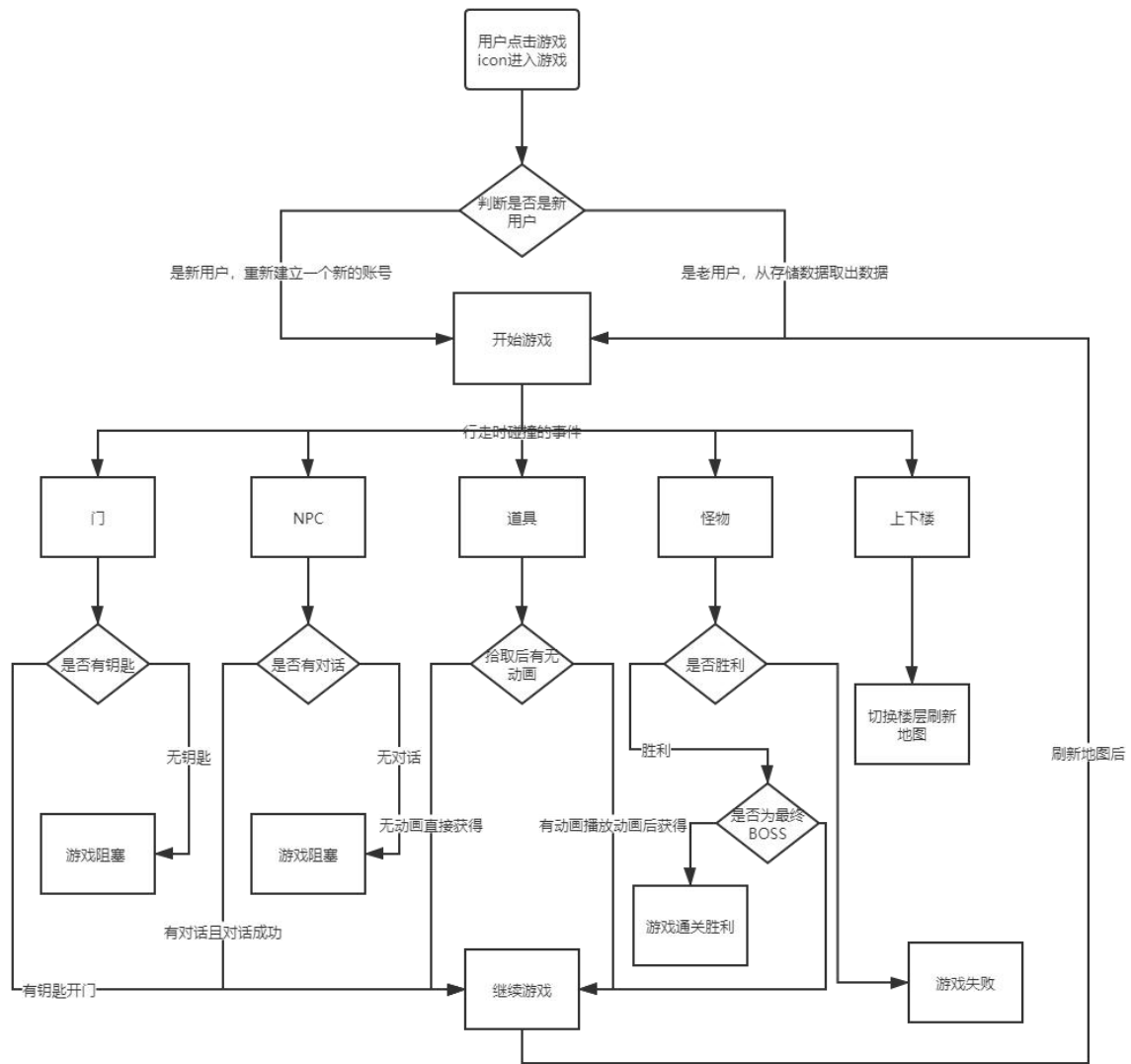


图 3.1 游戏流程图

新老用户判断：主要基于当前用户是否有数据已经存储在本地，通过 `wx.getStorage` 与 `wx.getStorageSync`（同步方法）获取，同时，对于有数据存储的老用户，也要进行对比有无新版本增加的必要的数据类型，若没有，也需要对该数据类型进行重置。

地图刷新：采用循环渲染 `fgui` 列表方式，从获得的角色层数数据中，获得列表中每个元素需要渲染的图片 `url`，由地图背景，角色，楼梯，NPC，门，怪物，道具的顺序进行渲染。

行走：项目采用角色在数组的位置变化进行。左是数组位置减一，右是加一，上是减 11，下是加 11，由于数组的图层问题，往右下走与往左上走的逻辑不同，否则会走到地图图层下。

门的种类：主要是三个玩家可用钥匙开的门，与一个小偷开的门，与两种不同的监狱门组成。两种监狱门的开门的逻辑不同。

NPC 的种类：除了常用 NPC 以外，将商店等不可删除的地图元素均归为 NPC，保证这些元素不会意外删除。常用 NPC 基本只有对话功能，商店等特殊 NPC 有特殊逻辑。部分 NPC 对话结束就被删除。

道具的种类：其中常用道具为红蓝宝珠与红蓝血瓶，血瓶为背包道具，背包内的道具拾取后调用进入背包动画。另外还有游戏内的各种其它道具。

怪物的种类：部分怪物拥有两套属性，在玩家登上一定层数第一次击杀魔王之后，部分怪物会变强。给玩家带来挑战。

3.4 功能模块设计

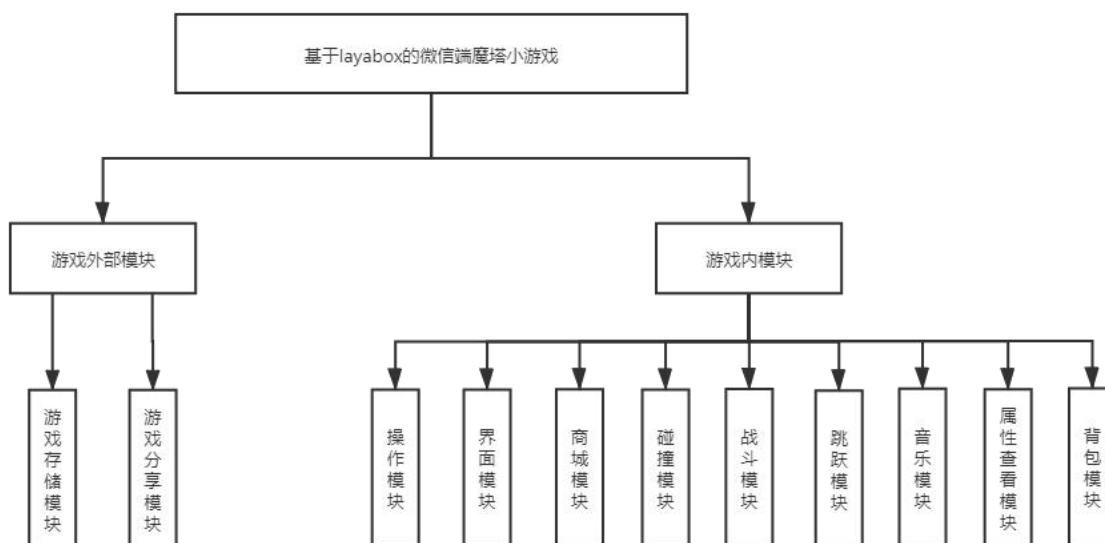


图 3.2 系统功能模块设计

上图 3.2 是功能模块设计，下面将进行详细介绍。

3.4.1 游戏存储模块

此模块负责存储玩家已玩过的数据，取出已玩过玩家的数据，删除失败的玩家的数据。

3.4.2 游戏分享模块

此模块负责微信端用户将小游戏分享出去。

3.4.3 操作模块

此模块用于用户在地图上的行走操作。并在有高级的事件触发时，操作模块进行隐藏不可用。

3.4.4 界面模块

此模块用于显示游戏内各种界面的切换与显示逻辑。

3.4.5 商城模块

此模块用于玩家在游戏内进行游戏道具的收售，增加玩家的游戏属性与道具数量。

3.4.6 碰撞模块

此模块用于玩家在游戏内行走时碰撞到墙、门、道具、怪物等时是否进行特殊逻辑。并在执行完碰撞后是否对游戏内的地图元素进行销毁。碰撞通过数组间的数组位置进行判断，防止因为存储数的二进制问题带来的位置不同。

3.4.7 战斗模块

此模块是游戏的核心，胜利与失败的判定点，战斗内有攻击力，防御力，暴击率，闪避率，命中率，其中怪物的暴击闪避命中是玩家不可知的，防止玩家在一刷时就完全了解怪物的全机制找到攻略。在玩家战斗开始之后，不能使用背包内的血药，当战斗到玩家生命力不足时且背包内还有血药时，战斗失败，怪物血量回满。若背包无血药，游戏失败，账号销毁。战斗胜利，则游戏继续。战胜最终 BOSS，则游戏胜利。

3.4.8 跳跃模块

该模块当玩家拾取到游戏特定道具——风之轮盘后开启。玩家可以通过该道具进行楼层跳跃。提升玩家体验。

3.4.9 音乐模块

该模块是玩家在体验游戏时获得的一种放松方式。进行播放音乐与音效。

3.4.10 属性查看模块

该模块当玩家拾取到游戏特定道具——圣光徽后开启。玩家可以通过该道具进行查看怪物属性，其中，怪物的命中暴击闪避不展示。提升玩家体验的同时给游戏增加更多的不确定性。

3.4.11 背包模块

该模块当玩家拾取到血瓶道具后，进行一个贝塞尔曲线运动的运动轨迹存入背包，减轻玩家为了存血来回走动的负担。

3.5 xlsx 表设计

在本项目中，数据读取采用 xlsx 表转 json 的方法进行读取。表中也需要有主键，方便程序设计时寻找需要的数据。采用 xlsx 表而不适用数据库在本次系统中的好处有：小游戏的体量比较小，采用数据库读取地图的方式增加了一层不必要的逻辑；xlsx 表不需要专业的懂程序开发的人员也能进行修改，且比数据库要更加清晰。本次设计的表中除了主键为必填项，其它都可以选填。

xlsx 表的功能和设计如下：

1. event 表： 主要保存基础地图的 icon 信息。
2. map 表： 主要保存地图上的其它道具、角色位置等信息。
3. monster 表： 主要保存怪物的信息。

表 3.1 event 表

字段名	数据类型	是否主键	描述
id	number	是	总编号
wall	number		墙编号
prop	number		道具编号（后弃用）
door	number		门编号（后弃用）
monster	number		怪物编号（后弃用）
npc	number		NPC 编号（后弃用）
way	number		路编号
floor	number		楼层编号（后弃用）
icon	string		图标地址

event 表中有许多弃用字段，这是因为在游戏前期准备使用图片的方式进行渲染地图，图片方式渲染地图则该表完全可以使用，而在游戏进行时发现图片的方式带来的体验效果不好，后将其改成使用有动效的 UI，故将 event 表中的这些数据弃用。

表 3.2 map 表

字段名	数据类型	是否主键	描述
layer	number	是	层数
map	Array<number>		具体地图的墙与路
character	Array<number>		角色位置（上楼，下楼）
npc	Array<number>		NPC 位置（位置，ID）
door	Array<number>		门位置（位置，ID）
monster	Array<number>		怪物位置（位置，ID）
prop	Array<number>		道具位置（位置，ID）

floor	Array<number>		上下楼位置（上楼，下楼）
-------	---------------	--	--------------

NPC，门，怪物，道具的数据要在代码里进行分离，将一个数组以单双位置拆分为两个数组，一个记录位置，一个记录 ID。

表 3.3 monster 表

字段名	数据类型	是否主键	描述
id	number	是	怪物 id
name	string		怪物名称
life	number		怪物生命值
attack	number		怪物攻击力
defense	number		怪物防御力
hit	number		怪物命中率
crit	number		怪物暴击率
dodge	number		怪物闪避率
gold	number		击杀怪物后可获得的金币数
experience	number		击杀怪物后可获得的经验值

4 系统实现

4.1 数据初始化

每一个项目都是用数据作为支撑的，所以第一步就是获取数据。

当一个新玩家进入游戏时，建立一套游戏内需要的数据。若是一个老玩家进入游戏，判定玩家存储的数据与这个版本的数据是否有新的数据，若有，则建立新的数据的基础数据，保证玩家在进入游戏时拥有全部完整的能进行游戏的数据结构，至此，整个项目的基础数据基本完成。

下图 4.1 为初始玩家数据结构图，属性不为初始值，只做结构示范。

```
▼ {grade: 1, life: 10000, attack: 1000, defense: 1000, hit: 1, crit: 1, dodge: 1, gold: 1000,...}
  attack: 1000
  blood_blue: 0
  blood_red: 0
  characterIndex: 0
  crit: 1
  defense: 1000
  dodge: 1
  experience: 1000
  gold: 1000
  grade: 1
  hit: 1
  ▶ key: [10, 10, 10]
  layer: 19
  life: 10000
  ▼ map: [{npc: [1], npcIndex: [92], door: [1], doorIndex: [82], monster: [], monsterIndex: [], prop: [],...},...]
    ▶ 0: {npc: [1], npcIndex: [92], door: [1], doorIndex: [82], monster: [], monsterIndex: [], prop: [],...}
    ▶ 1: {npc: [], npcIndex: [], door: [3, 1, 1, 1, 1, 1], doorIndex: [93, 97, 78, 45, 25, 60],...}
    ▶ 2: {npc: [10, 11], npcIndex: [117, 119], door: [1, 1, 1, 1, 1, 4, 1, 2, 5, 5],...}
    ▶ 3: {npc: [2, 3, 4], npcIndex: [4, 5, 6], door: [1, 1, 1], doorIndex: [27, 34, 84],...}
    ▶ 4: {npc: [12], npcIndex: [5], door: [1, 1, 1, 1, 5, 3, 2], doorIndex: [11, 13, 19, 21, 27, 60, 93],...}
    ▶ 5: {npc: [8, 13], npcIndex: [78, 43], door: [1, 1, 1, 2, 1, 1], doorIndex: [30, 34, 93, 95, 97, 107],...}
    ▶ 6: {npc: [], npcIndex: [], door: [2, 2, 3, 1, 1, 1, 1, 2, 1, 1],...}
    ▶ 7: {npc: [], npcIndex: [], door: [2, 6, 2, 6, 6, 2, 6, 2, 3, 1, 1],...}
    ▶ 8: {npc: [], npcIndex: [], door: [1, 1, 2, 1, 1], doorIndex: [16, 18, 28, 85, 103],...}
    ▶ 9: {npc: [], npcIndex: [], door: [1, 1, 1, 3, 2, 1, 1, 2, 1, 1],...}
    ▶ 10: {npc: [], npcIndex: [], door: [1, 1, 1, 5, 1, 1, 3], doorIndex: [15, 17, 62, 69, 74, 84, 104],...}
    ▶ 11: {npc: [5, 6, 7], npcIndex: [92, 93, 94], door: [1, 1, 1, 1, 2, 1, 2, 2, 1, 3, 3],...}
    ▶ 12: {npc: [14], npcIndex: [0], door: [1, 2, 2, 1, 1, 2], doorIndex: [16, 56, 64, 90, 96, 104],...}
    ▶ 13: {npc: [9], npcIndex: [70], door: [1, 3, 5, 5, 2, 1], doorIndex: [17, 35, 59, 69, 113, 119],...}
    ▶ 14: {npc: [], npcIndex: [], door: [5, 2, 2, 2], doorIndex: [49, 93, 103, 105],...}
    ▶ 15: {npc: [15, 16], npcIndex: [37, 39], door: [1, 1, 3], doorIndex: [81, 83, 104], monster: [],...}
    ▶ 16: {npc: [], npcIndex: [], door: [3], doorIndex: [38], monster: [22], monsterIndex: [60], prop: [],...}
    ▶ 17: {npc: [], npcIndex: [], door: [], doorIndex: [],...}
    ▶ 18: {npc: [17], npcIndex: [49], door: [5, 3, 3], doorIndex: [60, 71, 82], monster: [], monsterIndex: [],...}
    ▶ 19: {npc: [], npcIndex: [], door: [5, 5], doorIndex: [68, 74], monster: [22, 22, 12],...}
    ▶ 20: {npc: [], npcIndex: [], door: [], doorIndex: [],...}
    ▶ 21: {npc: [], npcIndex: [], door: [6, 6], doorIndex: [81, 83], monster: [12, 25, 25],...}
```

图 4.1 初始玩家数据的结构图

4.2 功能模块实现

4.2.1 游戏存储模块

整个模块共分为 3 部分。

1. 存储数据

微信端可使用 `setStorage` 或 `setStorageSync` (同步方法) 进行，在游戏中存储数据的使用频率比较频繁，故采用异步方法，防止游戏卡在同步存储的进度上。使用方法如下：

```
wx.setStorage({
  key: "data",
  data: saveData,
  success: function () {
    console.log("成功")
  },
  fail: function () {
    console.log("失败")
  },
  complete: function () {
    console.log("完成")
  }
});
```

Web 端可使用 `Laya.LocalStorage.setJSON` 或 `setItem` 方法进行存储。

2. 读取数据

微信端可使用 `getStorage` 或 `getStorageSync` (同步方法) 进行，在游戏中读取数据仅执行一次，且玩家应必须得到数据才能进入游戏，故采用同步方法，使用时，需要将数据由 json 格式转换为键值对的形势。防止游戏没有读取到数据就进入。使用方法如下：

```
let value = JSON.parse(JSON.stringify(wx.getStorageSync("data")));
return value;
```

Web 端可使用 `Laya.LocalStorage.getJSON` 或 `getItem` 方法进行读取。

3. 删除数据

微信端可使用 `removeStorage` 或 `removeStorageSync` (同步方法) 进行，在游戏中删除数据可随意使用，但应该注意玩家在删除时可能已经因某种原因丢失了该数据，需要进行处理防止报错。使用方法如下：

```
try {
  wx.removeStorageSync(key);
} catch (e) {
  console.log(e);
}
```


Web 端可使用 `Laya.LocalStorage.removeItem` 方法进行删除。

4.2.2 游戏分享模块

微信端独有功能，分享分为两种，一种是界面自带的分享功能，但需要开发者调用开启，调用方法为：

```
wx.showShareMenu({  
    withShareTicket: true  
})
```

另一种是需要向微信后台申请通过图片后调用 `wx.shareAppMessage` 进行，由于本项目暂未进行到微信后台申请的步骤，暂时无法完成。

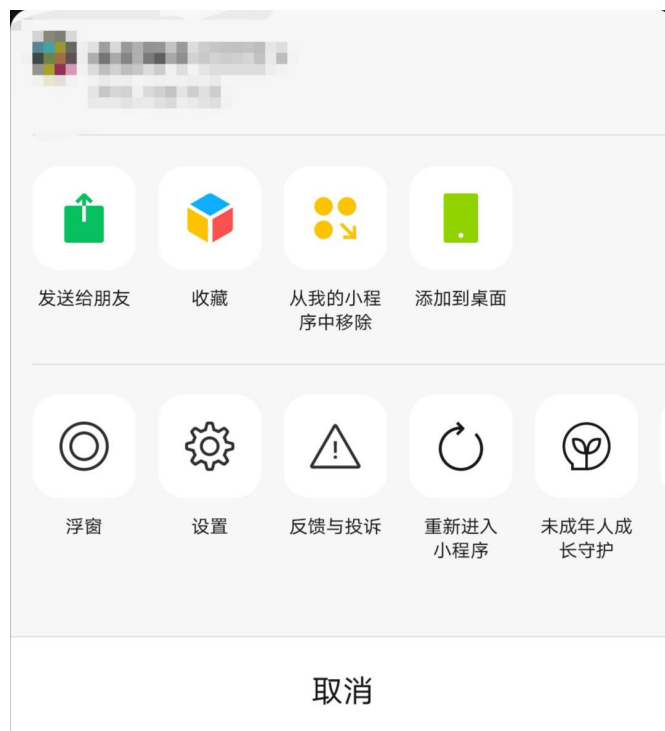


图 4.2 微信自带的分享界面

4.2.3 操作模块

用户的主要操作，用户未点击时，实时监听玩家点下摇杆盘的事件（`MOUSE_DOWN` 事件），并判断用户点击摇杆盘的区域，即移动方向，在按下后才开始实时监听玩家的点住事件（`MOUSE_OVER`），取消监听点下摇杆盘事件防止多指操作，并在没有离开摇杆盘或手指弹起前，实时判断手指在摇杆盘的哪部分位置，进行角色的移动，在离开摇杆盘（`MOUSE_OUT`）或手指弹起（`MOUSE_UP`）后，需要取消监听玩家的点住事件并重新监听玩家点下摇杆盘的事件。

实现方法如下：

```

/** 操作上下左右 */
protected initOperation() {
    this._view.m_operation.getChild('btnUp').asGraph.on(Laya.Event.MOUSE_DOWN, this, this._moveStartUp);
    this._view.m_operation.getChild('btnDown').asGraph.on(Laya.Event.MOUSE_DOWN, this, this._moveStartDown);
    this._view.m_operation.getChild('btnRight').asGraph.on(Laya.Event.MOUSE_DOWN, this, this._moveStartRight);
    this._view.m_operation.getChild('btnLeft').asGraph.on(Laya.Event.MOUSE_DOWN, this, this._moveStartLeft);
}

protected _beforeMove() {
    this._view.m_operation.getChild('btnUp').asGraph.off(Laya.Event.MOUSE_DOWN, this, this._moveStartUp);
    this._view.m_operation.getChild('btnDown').asGraph.off(Laya.Event.MOUSE_DOWN, this, this._moveStartDown);
    this._view.m_operation.getChild('btnRight').asGraph.off(Laya.Event.MOUSE_DOWN, this, this._moveStartRight);
    this._view.m_operation.getChild('btnLeft').asGraph.off(Laya.Event.MOUSE_DOWN, this, this._moveStartLeft);
    this._view.m_operation.on(Laya.Event.MOUSE_OUT, this, this._moveReturn);
    this._view.m_operation.on(Laya.Event.MOUSE_UP, this, this._moveReturn);
    this._view.m_operation.getChild('btnUp').asGraph.on(Laya.Event.MOUSE_OVER, this, this._moveUp);
    this._view.m_operation.getChild('btnDown').asGraph.on(Laya.Event.MOUSE_OVER, this, this._moveDown);
    this._view.m_operation.getChild('btnRight').asGraph.on(Laya.Event.MOUSE_OVER, this, this._moveRight);
    this._view.m_operation.getChild('btnLeft').asGraph.on(Laya.Event.MOUSE_OVER, this, this._moveLeft);
}

```

图 4.3 操作盘的实现 1

```

protected _moveReturn() {
    this._moveStop();
    this.initOperation();
}

protected _moveStop() {
    this._view.m_operation.off(Laya.Event.MOUSE_OUT, this, this._moveReturn);
    this._view.m_operation.off(Laya.Event.MOUSE_UP, this, this._moveReturn);
    this._view.m_operation.getChild('btnUp').asGraph.off(Laya.Event.MOUSE_OVER, this, this._moveUp);
    this._view.m_operation.getChild('btnDown').asGraph.off(Laya.Event.MOUSE_OVER, this, this._moveDown);
    this._view.m_operation.getChild('btnRight').asGraph.off(Laya.Event.MOUSE_OVER, this, this._moveRight);
    this._view.m_operation.getChild('btnLeft').asGraph.off(Laya.Event.MOUSE_OVER, this, this._moveLeft);
    this._view.m_operation.getController('type').selectedIndex = 0;
}

```

图 4.4 操作盘的实现 2

在进行其它高级事件时，需要进行_moveReturn 事件，结束后执行 initOperation 事件。

4.2.4 界面模块

游戏内采用两种打开与关闭的方式，一种是 Laya 引擎下的 Laya.Scene.open()与 Laya.Scene.close()打开与关闭。传参为界面路径+是否关闭当前界面参数。另一种是使用 fgui 下的管理器，fairygui.GRoot.inst.addChild()与 remove 方式，传参为界面节点。

游戏内界面图片使用 PhotoShop CS6 进行 P 图处理，使用 FaryGUI 搭建界面内全部 UI 设计。

4.2.5 商城模块

金币、经验商城的设计逻辑相似，均为多个购买按钮与一个关闭按钮。将其绑定事件：

```

initScene(cb: Function,finish: Function) {
    this._callBack = cb;
}

```

```
this._finish = finish;
this.m_btnLife.onClick(this, ()=>{
    this.useMoney(()=>{
        DataUtil.player.life += 800;
        if (!!this._callBack) this._callBack();
    })
})
this.m_btnAttack.onClick(this, ()=>{
    this.useMoney(()=>{
        DataUtil.player.attack += 4;
        if (!!this._callBack) this._callBack();
    })
})
this.m_btnDefense.onClick(this, ()=>{
    this.useMoney(()=>{
        DataUtil.player.defense += 4;
        if (!!this._callBack) this._callBack();
    })
})
this.m_btnHit.onClick(this, ()=>{
    this.useMoney(()=>{
        DataUtil.player.hit += 0.2;
        if (!!this._callBack) this._callBack();
    })
})
this.m_btnCrit.onClick(this, ()=>{
    this.useMoney(()=>{
        DataUtil.player.crit += 0.2;
        if (!!this._callBack) this._callBack();
    })
})
this.m_btnDodge.onClick(this, ()=>{
    this.useMoney(()=>{
        DataUtil.player.dodge += 0.2;
        if (!!this._callBack) this._callBack();
```

```
    })  
  })  
  this.m_btnClose.onClick(this, ()=>{  
    this.recover();  
  })  
}
```



图 4.5 金币商城



图 4.6 经验商城

4.2.6 碰撞模块

玩家在行走过程中，需要进行碰撞检测，如果碰撞到了墙、门、道具、怪物等，需要进行特殊逻辑，下面以最为特殊的道具为例，在拾取道具时，是有检测到碰撞，但是随即该道具被销毁，并不阻止玩家移动。逻辑如下：

```

/** 判断碰撞 */
protected _judgeCollision(index: number) {
    if (this._view.m_mapList._children[index].asCom._children.length == 1) return false;
    if (this._judgeStairs(index)) return true;
    if (this._judgeNPC(index)) return true;
    if (this._judgeProp(index)) return true;
    if (this._judgeDoor(index)) return true;
    if (this._judgeMonster(index)) return true;
    return false;
}

/** 判断道具 */
protected _judgeProp(index: number) {
    for (let i = 0; i < DataUtil.player.map[DataUtil.player.layer].propIndex.length; i++) {
        if (index == DataUtil.player.map[DataUtil.player.layer].propIndex[i]) {
            if (PropUtil.addProp(DataUtil.player.map[DataUtil.player.layer].prop[i])) {
let { x, y } = this._view.m_mapList._children[index].asCom.getChildAt(1).asCom.localToGlobal();
let temp = this._view.m_mapList._children[index].asCom.getChildAt(1).asCom;
this._view.m_mapList._children[index].asCom.getChildAt(1).asCom.removeFromParent();

                temp.setXY(x, y)
                this._view.addChild(temp);
                let targetComp = this._view.m_blood_red;
                let rewardType = 0;
                if (DataUtil.player.map[DataUtil.player.layer].prop[i] == 5) {
                    targetComp = this._view.m_blood_blue;
                    rewardType = 1;
                }
                FlyUtil.flyObject(temp, targetComp, () => {
                    if (rewardType) {
                        DataUtil.player.blood_blue++;
                    }
                })
            }
        }
    }
}

```

```

        else {
            DataUtil.player.blood_red++;
        }
        this.flushPlayerPanel();
    })
}
else {
    this._view.m_mapList._children[index].asCom.getChildAt(1).asCom.removeFromParent();
    this.flushPlayerPanel();
}
DataUtil.player.map[DataUtil.player.layer].propIndex.splice(i, 1);
DataUtil.player.map[DataUtil.player.layer].prop.splice(i, 1);
return false;
}
}
return false;
}

```

4.2.7 战斗模块

战斗每 500ms 攻击一次，大部分怪物无先手，由勇士先出手，250ms 后才开始执行怪物攻击勇士的逻辑，其中需要进行暴击回避命中的判定。攻击的逻辑判断如下：

```

fightSelf() {
    if (this.data.attack <= DataUtil.player.defense) {
        return;
    }
    if (this.data.hit < DataUtil.player.dodge) {
        if (Math.floor(Math.random() * 100) < (DataUtil.player.dodge - this.data.hit)) {
            LifeComp.LossLife(0, 0, this.getChild('n13'));
            return;
        }
    }
    let times = 1;
    if (Math.floor(Math.random() * 100) < this.data.crit) {
        times = 2;
    }
}

```

```

let life = Number(this.m_myLife.text) - ((this.data.attack - DataUtil.player.defense) *
times);

LifeComp.LossLife((this.data.attack - DataUtil.player.defense) * times, times,
this.getChild('n13'));

if (life > 0) {
    DataUtil.player.life = life;
    this.m_myLife.text = "" + life;
}
else {
    Laya.timer.clearAll(this);
    DataUtil.removeNow();
}
}
}

```

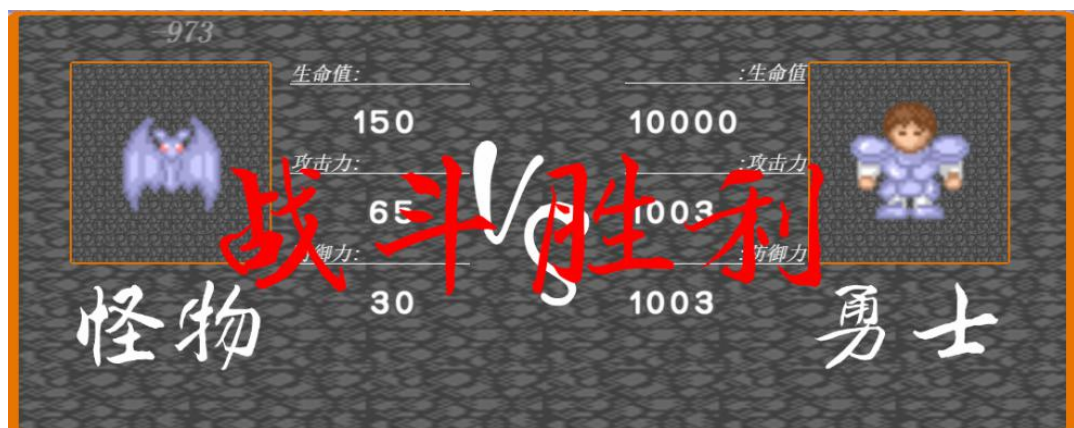


图 4.7 战斗胜利

4.2.8 跳跃模块

玩家点击风之轮盘道具，进行判断玩家是否已经拥有了该道具，若有，玩家进行选择跳转到哪个楼层，然后调用 `initMapList` 方法进行重绘地图即可。`initMapList` 方法逻辑如下：

```

protected initMapList(type?: number) {
    this._view.m_mapList.removeChildren(0, 120);
    let mapList = DataUtil.getMap(DataUtil.player.layer).map;
    for (let i = 0; i < mapList.length; i++) {
        let item = this._view.m_mapList.addItemFromPool() as UI_MapItem;
        item.getChild('map').asLoader.url=ResUrlUtil.getEventUrl(DataUtil.getEvent(mapList[i]).icon);
        if (!type && DataUtil.getMap(DataUtil.player.layer).character[0] == i) {

```



```

        let character = <UI_character>fairygui.UIPackage.createObjectFromURL(UI_character.URL,
UI_character);

        character.setScale(2, 2);
        item.addChild(character);
        DataUtil.player.characterIndex = i;
    }
    if (type && DataUtil.getMap(DataUtil.player.layer).character[1] == i) {
        let character = <UI_character>fairygui.UIPackage.createObjectFromURL(UI_character.URL,
UI_character);

        character.setScale(2, 2);
        item.addChild(character);
        DataUtil.player.characterIndex = i;
    }

    if(DataUtil.getMap(DataUtil.player.layer).floor[0]>=0&&DataUtil.getMap(DataUtil.player.layer).f
loor[0] == i) {
        let upStairs= <UI_UpStairs>fairygui.UIPackage.createObjectFromURL(UI_UpStairs.URL,
UI_UpStairs);

        item.addChild(upStairs);
    }

    if(DataUtil.getMap(DataUtil.player.layer).floor[1]>=0&&DataUtil.getMap(DataUtil.player.layer).f
loor[1] == i) {
        let downStairs =<UI_DownStairs>fairygui.UIPackage.createObjectFromURL(UI_DownStairs.URL,
UI_DownStairs);

        item.addChild(downStairs);
    }
    this._addNPC(i, item);
    this._addDoor(i, item);
    this._addMonster(i, item);
    this._addProp(i, item);
}
}

```




图 4.8 楼层跳跃

4.2.9 音乐模块

调用 Laya.SoundManager.playMusic 播放背景音乐，Laya.SoundManager.playSound 播放音效。其中背景音效可以叠加播放，背景同时音乐只能存在一个。

4.2.10 属性查看模块

玩家点击圣光徽道具，进行判断玩家是否已经拥有了该道具，若有，判断当前玩家层数是否有怪物，若有，则显示该层数剩余怪物的属性，通过 filter 函数过滤怪物属性的 json 即可。









	名称	骷髅队长	攻击	90	金·经	15·12
	生命	400	防御	50	损失	126
	名称	麻衣法师	攻击	120	金·经	20·17
	生命	250	防御	70	损失	355
	名称	大蝙蝠	攻击	65	金·经	10·8
	生命	150	防御	30	损失	0
	名称	青头怪	攻击	35	金·经	5·5
	生命	200	防御	10	损失	0
	名称	红蝙蝠	攻击	160	金·经	25·20
	生命	550	防御	90	损失	1911
	名称	初级卫兵	攻击	150	金·经	22·19
	生命	450	防御	90	损失	1377
	名称	怪王	攻击	250	金·经	32·30
	生命	700	防御	125	损失	???
	名称	白衣武士	攻击	300	金·经	40·35
	生命	1300	防御	150	损失	???

图 4.9 属性查看

4.2.11 背包模块

背包内的消耗品拥有使用 CD，在 fgui 制作一个进度条，使用进度条刷新 CD 的逻辑如下：

```
protected _useBloodBlue() {
    if (this._isFighting) return;
    if (this._view.m_blood_blue.value > 0) return;
    if (DataUtil.player.blood_blue <= 0) return;
    DataUtil.player.blood_blue--;
    DataUtil.player.life += 500;
    this._view.m_blood_blue.value = 100;
    this.flushPlayerPanel();
    Laya.timer.loop(20, this, this.cdBloodBlue);
}

protected cdBloodBlue() {
    this._view.m_blood_blue.value--;
    if (this._view.m_blood_blue.value <= 0) {
        Laya.timer.clear(this, this.cdBloodBlue);
    }
}
```

5 游戏测试

5.1 软件测试的重要性的目的

1996 年 6 月 4 日，阿丽亚娜 5 型火箭的首航，因软件引发问题导致在发射后 39 秒偏离轨道，激活了火箭自毁装置，原因为代码重用，损失 3.7 亿美元。

测试的最终目的是为了提升用户体验，但测试人员不肯发现系统中所有的缺陷，反复测试就是为了预防风险，发现更多的缺陷。

软件测试的目的是：

1. 为了证明程序有错误，但不仅仅是唯一目的。
2. 提高软件质量、保障安全性。
3. 降低开发成本。
4. 没有发现错误的测试也是有价值的。

5.2 基于 layabox 的微信端魔塔小游戏功能测试

5.2.1 游戏外部功能测试

游戏外部的功能测试总共分为游戏存储模块，游戏分享模块的系统功能测试。

表 5.1 游戏存储功能测试表

测试模块	用例描述	期望输出	测试结论
游戏存储	新玩家进入	log 中有数据输出，且数据与预想一致	正常
	每次获取道具打完怪物后关闭游戏后重新打开	数据有被保存，第二次打开游戏时与下线前一致	正常
	老玩家进入	log 中有数据输出，且数据与预想一致	正常
	游戏失败后查看内部存储	内部存储为空	正常

表 5.2 游戏分享功能测试表

测试模块	用例描述	期望输出	测试结论
游戏分享	点击分享	分享后有调起分享界面	正常

5.2.2 游戏内部功能测试

表 5.3 操作功能测试表

测试模块	用例描述	期望输出	测试结论
操作模块	玩家点击操作摇杆板随意操作角色行走	角色不会出现卡墙，移动前后左右的无问题	正常
	在战斗，在商店购买道具等事件发生时点击操作摇杆板	角色不会移动，摇杆板不会亮起	正常

表 5.4 界面功能测试表

测试模块	用例描述	期望输出	测试结论
界面模块	各界面之间来回点击关闭	无异常报错信息	正常
	界面关闭后查看log	无上个界面正在执行的逻辑，变量回收正常	正常

表 5.5 商城功能测试表

测试模块	用例描述	期望输出	测试结论
商城模块	金币购买物品	玩家金币减少，属性按购买的物品描述增加	正常
	经验购买物品	玩家经验减少，属性按购买的物品描述增加	正常
	出售多余钥匙	玩家钥匙余量减少，金币增加	正常

表 5.6 碰撞功能测试表

测试模块	用例描述	期望输出	测试结论
碰撞模块	玩家碰撞到墙壁	无反应	正常
	玩家碰撞到门	有钥匙则开门，无钥匙则无反应	正常
	玩家碰撞到道具	道具消失，玩家背包道具数量内或玩家属性提升	正常
	玩家碰撞到上下楼	过渡界面后刷新地图	正常
	玩家碰撞到怪物	打开战斗界面	正常

表 5.7 战斗功能测试表

测试模块	用例描述	期望输出	测试结论
战斗模块	战斗胜利	被战胜的怪物从地图上被销毁，玩家数据保持在战斗后的数据	正常
	战斗失败有血瓶	怪物回满血，怪物不消失	正常
	战斗失败无血瓶	账号销毁	正常
	特殊怪物	先手攻击	正常

表 5.8 跳跃功能测试表

测试模块	用例描述	期望输出	测试结论
跳跃模块	没获得道具前点击	提示不能使用	正常
	获得道具后点击使用跳跃楼层	出现过渡界面后刷新该层界面地图	正常
	跳跃为当前楼层	无反应	正常

表 5.9 音乐功能测试表

测试模块	用例描述	期望输出	测试结论
音乐模块	打开声音接收游戏播放的音乐音效	游戏声音正常显示	正常

表 5.10 属性查看功能测试表

测试模块	用例描述	期望输出	测试结论
属性查看模块	没获得道具前点击	提示不能使用	正常
	获得道具后点击使用属性查看	怪物显示的是本层且属性正常	正常
	获得道具后点击使用属性查看但该层无怪物	无反应	正常

表 5.11 背包功能测试表

测试模块	用例描述	期望输出	测试结论
背包模块	背包内无消耗品点击	无反应	正常
	背包内有消耗品点击	玩家角色的血量上升	正常
	背包内消耗品在CD	无反应	正常

经过上面几个模块的测试，发现整个游戏的功能完善。

6 总结

本论文首先介绍了基于 layabox 的微信端魔塔小游戏的背景和意义以及游戏行业在国内外的发展现状，然后详细说明了游戏的重要组成部分与游戏玩家的属性。之后介绍了系统中使用的关键技术。在中间部分，通过需求分析介绍了系统需要完成的功能，通过可行性分析表明系统可以实现，接着分析游戏的功能设计。之后对游戏的各个模块进行实现，同时也讲解了最重要的部分：各种事件的回调处理与刷新地图的逻辑。最后，对游戏进行了功能测试，所有模块都能按照预期运行。

基于 layabox 的微信端魔塔小游戏完成的工作如下：

1. 游戏的界面设计，界面之间的显示逻辑。
2. 游戏角色的基础行走，碰撞到墙壁后无法继续移动。
3. 游戏内碰撞到各种其它物体的事件逻辑。
4. 特殊道具的逻辑。
5. 商店界面的逻辑。

整个游戏的完成经历了前期美术资源收集，策划方案，需求分析，界面搭建，系统设计，详细设计，编码，测试等步骤。在这其中我阅读了大量的文献，并结合了现在流行的技术来完成，在不懂的地方得到了黄老师的指教。虽然能够正常运行，但还是存在很多问题，比如手机的适配，游戏的加载速度，界面的美化，部分逻辑的时间复杂度过高，希望在以后能够改善。

参考文献

- [1] Tencent腾讯. 微信官方文档·小游戏 快速上手|微信开放文档[EB/OL].
<https://developers.weixin.qq.com/minigame/dev/guide/>, 2019. 12. 12.
- [2] 搜游网络. LAYABOX技术文档[EB/OL]<https://ldc2.layabox.com/doc/>, 2019. 03. 13.
- [3] [美]Wendy Despain. 游戏设计的100个原理[M]. 人民邮电出版社: 北京, 2015-02.
- [4] 李楚墨, 马佛栋. 浅议中国游戏产业的现状与前景[J]. 时代金融, 2018, 12:264-272.
- [5] 百度百科“魔塔”词条
[EB]/[OL]. <https://baike.baidu.com/item/%E9%AD%94%E5%A1%94/861619?fr=aladdin>. 2019.
06. 12
- [6] 菠萝小笨笨. 取代Flash的HTML5技术（H5编写游戏的优点）
[EB/OL]. <https://blog.csdn.net/xiaokunzhang/article/details/80713744>. 2018.06.16.
- [7] 2020微信公开课PRO版<http://v.qq.com/detail/m/mzc00200fyleel7.html>
- [8] 刘为. 现代游戏开发引擎解析[J]. 企业技术开发(下旬刊), 2016. 10.
- [9] FairyGUI教程[EB]/[OL]. <https://www.fairygui.com/docs/guide/>. 2020. 1. 14
- [10] Martens C, Hammer M A. Languages of Play: Towards semantic foundations for game interfaces[J]. 2017
- [11] Christopher Totten. Teaching Serious Game App Design Through Client-based Projects[J]. 2017
- [12] Nicolas Ducheneaut, Nicholas Yee. Alone Together? Exploring the Social Dynamics of Massively Multiplayer Online Games[J]. 2016

作者简历

教育经历：

2010.9-2013.6 就读于浙江省温州市乐清市城东二中

2013.9-2016.6 就读于浙江省温州市乐清市乐清中学

2016.9-2020.6 就读于中国计量大学现代科技学院信息工程系计算机科学与技术专业

学习方面，本人学习了计算机专业的基础课程，包括数据结构，操作系统，计算机网络等。熟练使用C、Java、JavaScript、TypeScript等语言，对前端开发的html、css、jquery、vue框架、node.js有一定的了解，熟悉canvas开发，了解市面上主流的游戏引擎，熟练掌握Layabox引擎，对全渠道的小游戏发布有一定的心得体会。于2019年12月在本市一家游戏公司进行小游戏开发的实习。

获奖情况：

本科期间没有发表过论文，也没有其他的获奖经历。

学位论文数据集

关键词*		密级*	中图分类号*	UDC
推荐系统；游戏；Layabox		公开	TP311.1	004
论文赞助				
学位授予单位*		学位授予单位代码*	学位类别*	学位级别*
中国计量大学现代科技学院		13292	工学	学士
论文题名*	基于 layabox 的微信端魔塔小游戏开发			论文语种*
并列题名*	Development of magic tower game on wechat based on layabox			简体中文
作者姓名*	朱铸杰	学号*	1630332228	
培养单位名称*	培养单位代码*	培养单位地址		邮编
中国计量大学 现代科技学院	13292	浙江省杭州下沙高教园区学源街		310018
学科专业*	研究方向*		学制*	学位授予年*
计算机科学与技术	程序研究 游戏开发		4 年	2020
论文提交日期*	2020.6			
导师姓名*	黄俊	职称*	教授	
评阅人	陈光平	答辩委员会主席*	黄俊	
答辩委员会成员	黄俊，刘晓芳，余骞，唐立梅，唐文彬			
电子版论文提交格式 文本（ <input checked="" type="checkbox"/> ）图像（ <input type="checkbox"/> ）视频（ <input type="checkbox"/> ）音频（ <input type="checkbox"/> ）多媒体（ <input type="checkbox"/> ）其他（ <input type="checkbox"/> ） 推荐格式：application/msword； application/pdf				
电子版论文出版（发布者）	电子版论文出版（发布）地		权限声明	
论文总页数*	42			
注：共 40 项，其中带“*”为必填数据。				