

游戏设计书

北理幼儿园的猪猪公寓

[](https://image.baidu.com/search/detail?ct=503316480%26z=%26tn=baiduimagedetail%26ipn=d%26word=%E7%8C%AA%E7%8C%AA%E5%85%AC%E5%AF%93%26step_word=%26ie=utf-8%26in=%26cl=2%26lm=-1%26st=-1%26hd=%26latest=%26copyright=%26cs=3070737240,3647073579%26os=1487377672,2462725930%26simid=4075641986,577481344%26pn=7%26rn=1%26di=36440%26ln=620%26fr=%26fmq=1583650913895_R%26ic=%26s=undefined%26se=%26sme=%26tab=0%26width=%26height=%26face=undefined%26is=0,0%26istype=2%26ist=%26jit=%26bdtype=11%26spn=0%26pi=0%26gsm=0%26objurl=http%3A%2F%2Fnewsimg.5054399.com%2Fuploads%2Fuserup%2F2002%2F03114J35F5.jpg%26rpstart=0%26rpnum=0%26adpicid=0%26force=undefined)



目录

[1.概述 1](#_Toc1194)

[1.1项目简介 1](#_Toc17430)

[1.2本文结构 1](#_Toc11992)

[1.3团队的成员及分工 1](#_Toc21623)

[1.4游戏类型与同类型游戏比较 2](#_Toc20250)

[1.5游戏分级 2](#_Toc1314)

[2.游戏故事 3](#_Toc30691)

[2.1游戏故事设定 3](#_Toc3331)

[2.2游戏故事梗概 3](#_Toc16815)

[2.2.1故事背景 3](#_Toc21238)

[2.2.2主要角色 6](#_Toc12013)

[2.2.3游戏反派 6](#_Toc21136)

[2.2.4胜利条件 6](#_Toc19516)

[3.游戏元素设定 7](#_Toc20458)

[3.1游戏角色NPC设定 7](#_Toc22396)

[3.2游戏物体及机关 7](#_Toc6884)

[4. 玩法设计 7](#_Toc19588)

[4.1游戏机制 7](#_Toc25992)

[4.1.1游戏胜负判定 7](#_Toc27780)

[4.1.2升级体系 8](#_Toc3692)

[4.2游戏世界行为规则 8](#_Toc15694)

[4.2.1场景行为 8](#_Toc1866)

[4.2.2非玩家角色行为（怪物行为） 8](#_Toc10274)

[5.游戏交互设计 8](#_Toc291)

[6.界面设计 9](#_Toc23479)

[6.1界面设计原则 9](#_Toc4772)

[6.2界面设计图与美工设计 9](#_Toc29580)

[6.3跳转关系图 12](#_Toc29364)

[7.游戏进程和关卡设计 12](#_Toc14028)

[7.1游戏进程设计 12](#_Toc28741)

[7.2关卡设计 13](#_Toc4937)

[7.3素材来源 14](#_Toc23011)

[8.程序总体设计 15](#_Toc23896)

[8.1软件设计 15](#_Toc23195)

[8.2主要功能设计 16](#_Toc25501)

[8.2.1动画设计 16](#_Toc16169)

[8.2.2图形设计 16](#_Toc7675)

[8.2.3音效设计 16](#_Toc28053)

[8.2.4AI设计 16](#_Toc14664)

[8.2.5关卡完成逻辑 17](#_Toc27294)

[8.3主要脚本和算法 17](#_Toc3452)

[8.3.1锁定视角在玩家(target)身上 17](#_Toc25516)

[8.3.2切换场景脚本 18](#_Toc14960)

[8.3.3玩家控制脚本 19](#_Toc10525)

[8.3.4实现血条显示 21](#_Toc18272)

[8.3.5战斗脚本实现 23](#_Toc9252)

# 1.概述

## 1.1项目简介

来自北理幼儿园的四名成员时常慨叹自己不是主角没有主角光环。偶然的一次机会他们穿越到了他们喜欢的电视剧猪猪公寓的世界中，原本安静祥和的世界被辣酱大魔王入侵，此时他们惊讶地发现自己拥有了“主角光环”，各自不同的能力让他们各怀绝技，拯救被污染的世界的重任落在了他们头上。

## 1.2本文结构

见目录

1. 概述
2. 游戏故事
3. 游戏元素设定

## 1.3团队的成员及分工

龙锴：组长

负责整体项目的把控，并与老师及时沟通

陈敬天：程序员

负责代码的整体编写把控，尽可能完成预设功能实现

王翌丰：测试员 程序员

负责部分项目的编码，同时测试已完善的功能

窦义通：美工设计

主要负责游戏背景设计和美工处理

## 1.4游戏类型与同类型游戏比较

猪猪公寓是一款集冒险RPG为一体的3D地牢探险游戏

《元气骑士》：是2.5D的地牢探险游戏，玩家通过结合不同的正副手武器与技能来进行闯关。

《失落城堡》：横版地牢探险游戏，支持4人线上联机，通过闯关和收集特殊物品来闯关配合。

相比于以上两款游戏，我们的游戏题材比较新颖

同时3D的游戏体验带给玩家们更大的惊喜。

## 1.5游戏分级

[](https://baike.baidu.com/pic/ç½ç»æ¸¸æåçº§å¶/22638120/0/e7cd7b899e510fb31ab48da2db33c895d1430c51?fr=lemma%26ct=single)

属于E10+级 面向10岁以上的人群

# 2.游戏故事

## 2.1游戏故事设定

地点时代：爱情公寓5的伴生动画猪猪公寓，生活着猪猪化的爱情公寓5原住民，和一些公寓中拟人化的物品，异界传送门的开启，异界生物的降临消灭了所有的猪猪公寓原住民，原本的公寓成为了4位勇士的主战场。

特殊规则：猪猪公寓是一座封闭的“地牢”，玩家需要通过一个区域才能够前往下一个区域；上一场冒险中获得的东西，不会累积到下一局，每一次冒险都是全新的开始。

## 2.2游戏故事梗概

### 2.2.1故事背景

（1序幕）4个同寝室的好兄弟在一起津津有味地看爱情公寓5 “我的青春回来啦！”陈说到。“确实，特别是这集猪猪公寓拍的特别有寓意啊这游戏我也挺想玩的。”“我搜了一下网上全是盗版要不我们自己开发怎么样。”“好啊好啊，正好当做完成作业了。”四人达成共识。很快四个人开发的游戏步入尾声，深夜当四个人还在熟睡的时候，熟悉的白光闪过，四个人掉入了游戏世界。

（2开始）“这是哪啊？”，眼前陌生的环境让4个男孩子恐慌起来。“是我将你们召唤来的。”突然一个声音从四个人的身后响起。“是谁？”四个人被吓了一跳，回头一看却是一头可爱的猪。“我是猪小黑，是你们创造出来的生命。”“猪小黑？那这里不是”龙惊异地说道“我们创建的游戏世界！”四个人异口同声。“没错，这正是你们创建的世界，但是很不幸，一个邪恶的传送门降临，异界生物抢走了我们的居所，并将其他的猪猪及居民消灭了，我有幸躲过，所以我召唤了你们来拯救这个世界。”“害，我以为什么呢，那是不是闯完了，我们就能离开了？”陈不放在心上地笑笑“我们可是世界的创造者还会怕你们吗？”猪小黑不放心地托付给我们，我们需要打败并解救被镇守在公寓中的异界生物，最后解锁区域打败大魔王消灭传送门。“好！我们去去就回。”

（3迷茫）四个人走向了公寓的入口，猪小黑提示到每次只能进入一个人，闯关的过程中只会受伤，失败了也不会损失生命，但只有穿过关卡才能离开这里。“那身为组长，我先去吧！”王说，说着拿起了猪小黑准备的武器穿过了光门。光门过后，是猪猪公寓的广场，“这里镇守的会是谁呢？”他一边自问自答，一边往前去，这时不断的敌人从四面八方向他围绕过来，他生涩地通过走位和武器击退着来犯的敌人，“撕拉”，他被敌人碰触到，身体受到了损伤，疼痛让他的动作更加缓慢，很快他就被送出了战场。回到了初始地的他惊魂未定，只有身上隐隐作痛的伤口，在提醒他失败了，其余三人默不作声地替他包扎，少年们内心刚刚燃起的信心之火很快就熄灭了。一个不眠之夜过去，四人各怀心事，其余三人也很快做出了尝试，可都是乘兴而去败兴而归，一次次的失败狠狠地打击了他们。

（4坚定）“这样下去总不是办法，我们得有更好的策略”窦说。“我们应该先强化自己，”于是他们分工明确，三个人负责学习更好的技艺强化、辅助、治疗一个人，让他去攻克难关。时间很快的过去，新的征程再次开启。窦背负着三人的希望走上了战场，他通过精巧的走位武器配合击退了一波又一波的敌人，很快他面临了第一个镇守BOSS。身为BOSS的异界生物极为强大，并且拥有着其他小怪没有的技能属性，这场战斗无疑是艰难的。“背负着所有人的希望我要战斗下去”窦默默念到。一场艰难的战斗之后，他摇摇晃晃地站在猪小贤倒下的身体前，“终于结束了”，当他要挥下武器的一刻，他突然想到自己的任务是击败拯救而不是毁灭，此时类似主神的声音响起，“会否需要继续闯关”，窦非常地坚定脚步冲向了下一关的大门，“感谢你勇者，我赞赏你的勇气，请收下我的赠予”，主神的电子合成声音第一次附带感情说到。“不用客气，毕竟我们是你们的创造者也应该拯救你们”，窦一边说着一边选择好自己的奖励来强化自己，加快了迈向了下一关的步伐。

（5战斗）四人的精心配合下，连战连捷，击退了守在公寓各个位置的异界生物，获得了不菲的奖励与强化，最终决战在即，4人汇合在一起鼓劲加油“加油！马上就可以回家了！”四个人很感动，他们克服了独在异乡的孤独，克服了战斗的恐惧，攻克了一个又一个难关。窦冲向了最后一关，依旧是熟悉的敌人，心境却是不同的心境。“上吧”心中默念到，冲到关卡的最末，大魔王就屹立在那里。“多说无益，手底下见真招吧”伴随着窦的怒吼，两人战在一起。许久过后，三人紧紧地盯着关卡的入口，等待着奇迹出现，窦摇摇晃晃在白光中闪现出来。“还是失败了吗”三人有些失落，窦坚定了身影握紧了拳头，“赢了！”四人紧紧地相拥在一起，此时公寓已经完全的回复了原样，充满了过去的欢乐气息，猪猪原住民都围绕过来，向四位勇者表达谢意。“不客气，你们让我们也懂得了要珍惜时间与生命”“我们一定会将自己的游戏完善地更好的。”四人高兴地回复说，欢声笑语中，度过了一夜。

（6终章）四人回到了现实世界，而现实世界好像刚流动了5分钟一样。四个人面面相觑，围绕在电脑桌前，看着自己开发的游戏，一个个生动活泼的形象跃然于屏幕上，“加油！让我们的游戏走出学校，走出世界！”四个人在漆黑一片中，眼里迸发出璀璨的骄阳。

### 2.2.2主要角色

四位来自于北理幼儿园的成员

### 2.2.3游戏反派

异界传送门降临下的异界怪物以及BOSS魔王

### 2.2.4胜利条件

战胜镇守在不同区域的异界生物，通往天台打败大魔王，摧毁传送门

# 3.游戏元素设定

## 3.1游戏角色NPC设定

BOSS大魔王：身为污染猪猪公寓的大魔王，拥有强大的能力，坐镇于整个猪猪公寓最末端的位置，等待着下一位勇者的挑战。

异界怪物：身为最底层的小怪，拥有不同的攻击方式与种类，等待着玩家探索。

## 3.2游戏物体及机关

游戏物体：树木、草丛、野怪等

机关：陷阱等

# 玩法设计

## 4.1游戏机制

《北理幼儿园的猪猪公寓》本质上是一个RPG冒险地牢游戏，所以玩家能采取的行动就是与怪物进行战斗来达到通关和强化自己的目的。特色机制是玩家和怪物的攻击都有前后摇，玩家可以通过巧妙的时机选择来进行规避。

### 4.1.1游戏胜负判定

胜利：玩家需要战胜当前场景下的所有敌人来达到通关的目的

失败：玩家的血量为0，游戏结束

### 4.1.2升级体系

我们游戏中设计比较特色的就是升级体系，也是现在很火的一个设计理念。玩家通过战胜敌人后，自己选择攻击力，护甲值（伤害减免），生命值，以及攻速四个方面进行升级，比较自由。

## 4.2游戏世界行为规则

### 4.2.1场景行为

我们制作的场景是静态的，目前没有提供，只设定了有的地方不可以通过，以及提供光照。

### 4.2.2非玩家角色行为（怪物行为）

我们设计的怪物会自动刷新在场景中，并有自己的领地范围，当玩家踏入警戒范围时，会受到怪物的追杀，而且没有OT（超出仇恨）的范围，所以玩家需要小心谨慎选择行进路线。

# [5.游戏交互设计](file:///E:\\我仍然爱学习\\网课集锦\\课程设计作业3-游戏设计书-游戏原型-游戏演示.docx" \l "_Toc309653030)

游戏采用鼠标交互，左键选择到达地点并且会自动选择最佳路径，右键进行攻击。

# 6.界面设计

## 6.1界面设计原则

我们设计的界面有开始界面 游戏介绍 游戏关卡 死亡界面 升级界面 最终界面 关于我们

原则是每一个界面互相之间有联系，能互相进行跳转，尽量美观易懂。

## 6.2界面设计图与美工设计

开始界面：



游戏介绍：



升级系统：



死亡界面：



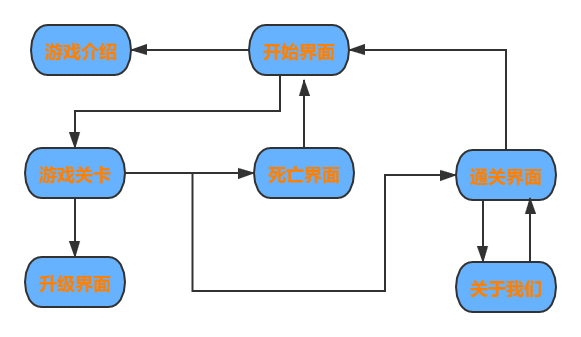
通关界面：



关于我们：



## 6.3跳转关系图



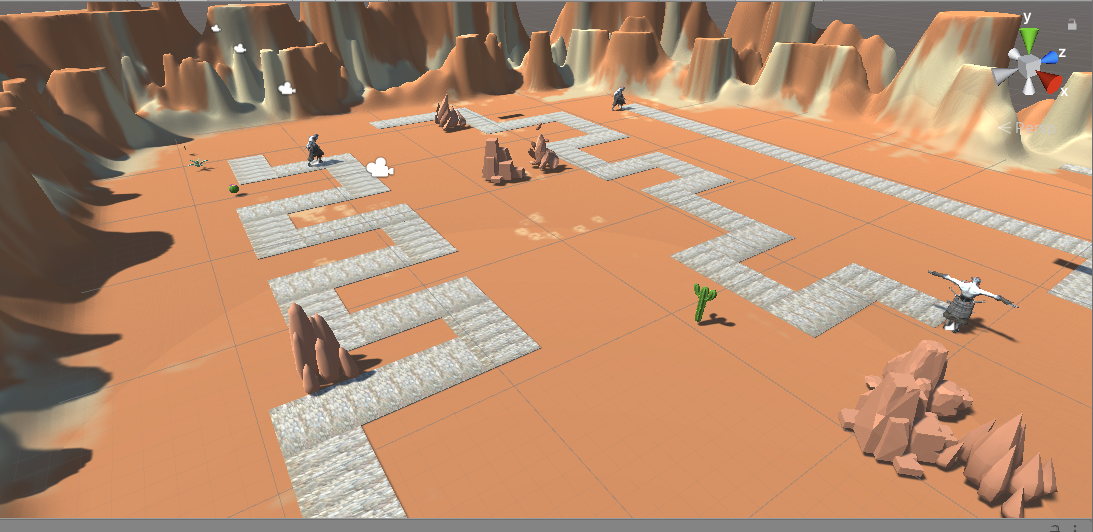
# 7.游戏进程和关卡设计

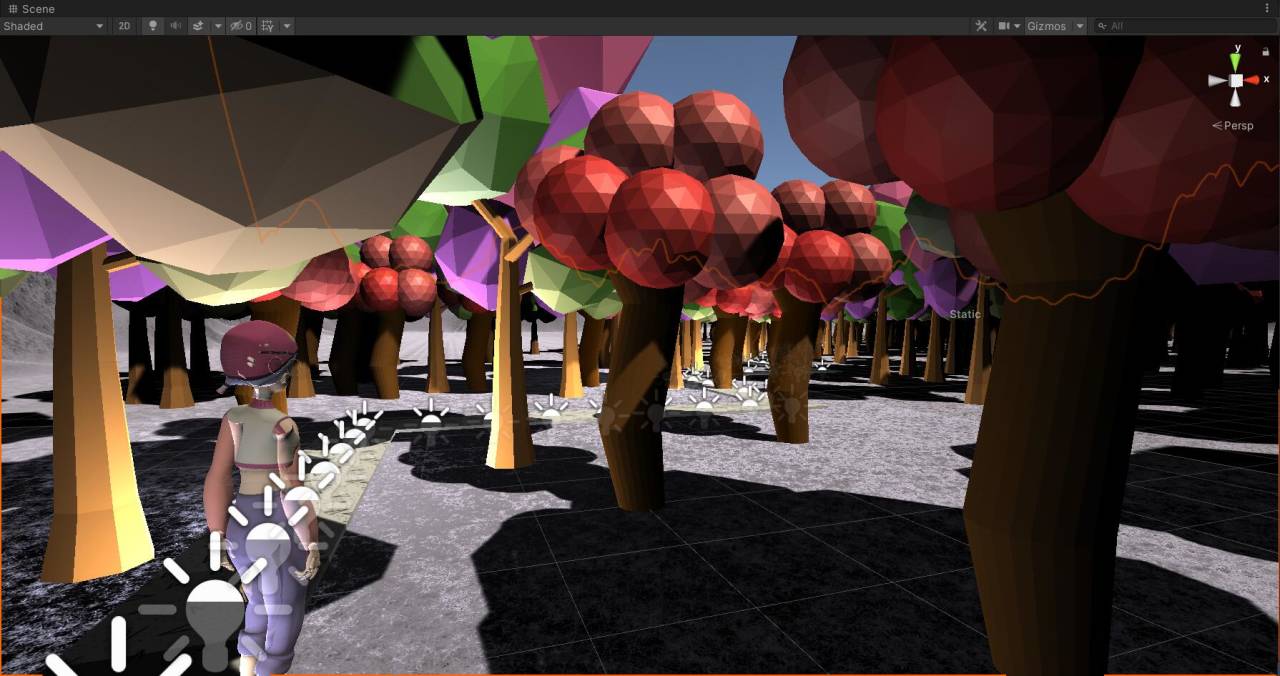
## 7.1游戏进程设计

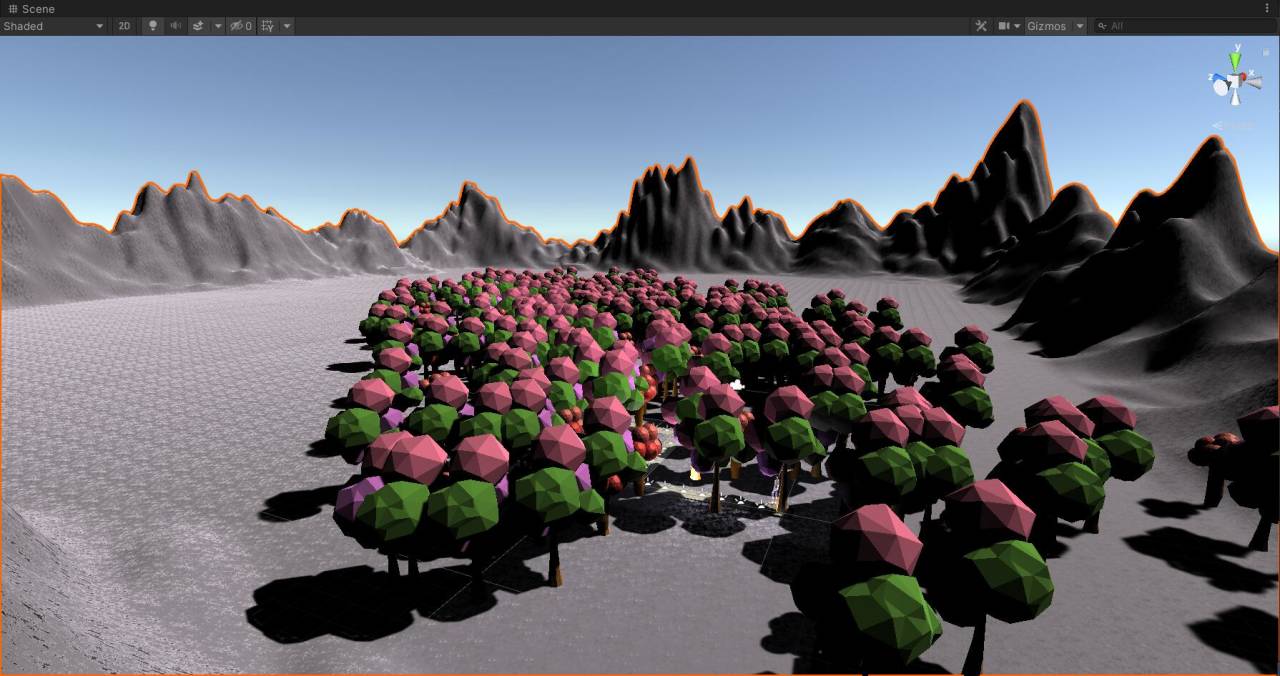
我们设计了多个关卡。剧情的分析构想是我们的主角经过层层磨练，踏入真正的公寓战场来面对异界怪物的厮杀决斗。主角过关斩将，踏入了公寓的顶层天台，面对强大的敌人和BOSS，以及造成这一切的罪魁祸首传送门，展开恶战的勇气信心。

## 7.2关卡设计

整体结构：







场景的分布如上图

敌人有BOSS,场景中的道具不可行走，是障碍物

胜利条件：击败敌人

失败条件：玩家死亡

## 7.3素材来源

1.素材包、地形都是在unity store中寻找的；

2.贴图和音频都是百度上下载的，具体链接找不到了；

3.人物建模

人物建模：<https://www.mixamo.com>

具体名称：

Jackie：https://www.mixamo.com/#/?page=1&query=Jackie&type=Character

Mutant：https://www.mixamo.com/#/?page=1&query=Mutant&type=Character

Vampire A Lusth：<https://www.mixamo.com/#/?page=1&query=vampire&type=Character>

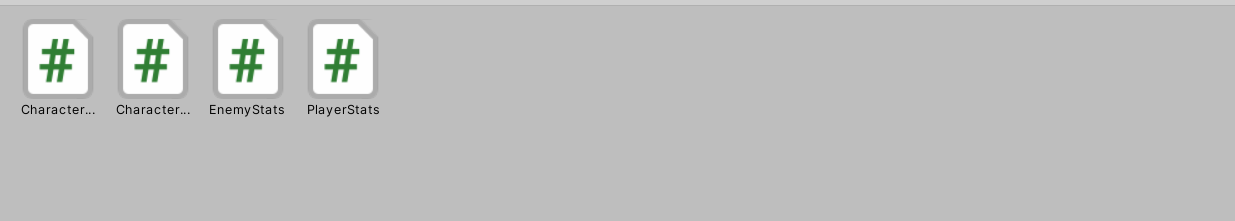
人物动画：https://www.mixamo.com/#/?page=1&query=&type=Motion%2CMotionPack

# 8.程序总体设计

## 8.1软件设计

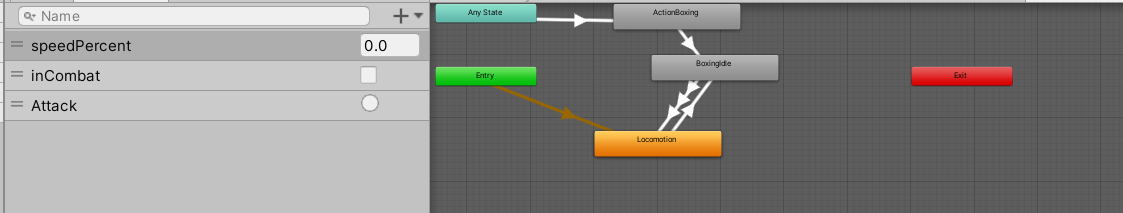
在文件夹Scripts、stats中有我们需要的脚本





## 8.2主要功能设计

### 8.2.1动画设计



动画设计主要是应用UNITY3D里的ANIMATION进行切换，设定特定的布尔代数在TURE OR FALSE 进行动画状态的切换。

### 8.2.2图形设计

运用的是probuilder，属于unity3d中的额外脚本，进行场景的搭建。

### 8.2.3音效设计

音效分为两种：BGM和战斗音效。BGM当进入关卡的时候和切换场景都会有背景音乐；战斗音效主要是在战斗系统中进行添加，在击打受伤时的音效。

### 8.2.4AI设计

主要是调节敌人的视角始终面对玩家，然后对玩家发动攻击



### 8.2.5关卡完成逻辑

我们的游戏是击杀场景中的所有怪物才会通关，所以设定了一个参数存储最开始的怪物数量，然后每有一个怪物死亡执行DIE函数的时候，数量-1，直到数量为0，切换场景到下一个升级界面。

## 8.3主要脚本和算法

### 8.3.1锁定视角在玩家(target)身上

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class CameraControl : MonoBehaviour

{

public Transform target;

public Vector3 offset;

public float currentZoom = 10f;//视角缩放

public float minZoom = 5f;

public float maxZoom = 15f;

public float ZoomSpeed = 4f;

public float currentYaw = 0f;//视角移动

public float YawSpeed = 100f;

public float height = 2f;

// Start is called before the first frame update

void Start()

{

}

// Update is called once per frame

void Update()

{

currentZoom -= Input.GetAxis("Mouse ScrollWheel") \* ZoomSpeed;//滚轮控制视角缩放

currentZoom = Mathf.Clamp(currentZoom, minZoom, maxZoom);

currentYaw -= Input.GetAxis("Horizontal") \* YawSpeed \* Time.deltaTime;

}

private void LateUpdate()

{

transform.position = target.position - offset \* currentZoom;

transform.LookAt(target.position + Vector3.up \* height);

transform.RotateAround(target.position, Vector3.up, currentYaw);

}

}

### 8.3.2切换场景脚本

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class ChangeScene : MonoBehaviour

{

// Start is called before the first frame update

void Start()

{

}

// Update is called once per frame

void Update()

{

}

public void OnChangeScene(string SceneName)

{

Application.LoadLevel(SceneName);

}

public void ChangeToLast()//切换至上一个场景

{

string name = PlayerPrefs.GetString("name");

Application.LoadLevel(name);

}

public void ChangeToNext()//切换至下一个场景

{

string name = PlayerPrefs.GetString("name");

if (name == "Instruction")

name = "Level1";

else

name = "Level2";

Application.LoadLevel(name);

}

}

### 8.3.3玩家控制脚本

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

[RequireComponent(typeof(PlayerMotor))]

public class PlayerControl : MonoBehaviour

{

public Interactable focus;

//public LayerMask movementmask;

Camera cam;

PlayerMotor motor;

// Start is called before the first frame update

void Start()

{

cam = Camera.main;

motor = GetComponent<PlayerMotor>();

}

// Update is called once per frame

void Update()

{

if(Input.GetMouseButtonDown(0))//鼠标左键点击

{

Ray ray = cam.ScreenPointToRay(Input.mousePosition);

RaycastHit hit;

if(Physics.Raycast(ray,out hit))//指向点击的位置

{

motor.Moveto(hit.point);//移动到鼠标点击位置

RemoveFocus();

}

}

if (Input.GetMouseButtonDown(1))//鼠标右键点击

{

Ray ray = cam.ScreenPointToRay(Input.mousePosition);

RaycastHit hit;

if (Physics.Raycast(ray, out hit))//指向点击的位置

{

Interactable interactable = hit.collider.GetComponent<Interactable>();

if(interactable != null)

{

SetFocus(interactable);

}

}

}

}

void SetFocus(Interactable newfocus)//设置焦点（即玩家指向点）

{

if(newfocus!=focus)

{

if(focus!=null)

focus.Defocused();

}

focus = newfocus;

newfocus.Onfocused(transform);

//motor.Moveto(newfocus.transform.position);//交互的同时移动向该物品

motor.FollowTarget(newfocus);

}

void RemoveFocus()

{

if (focus != null)

focus.Defocused();

focus = null;

motor.StopFollow();

}

}

### 8.3.4实现血条显示

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

[RequireComponent(typeof(CharacterStats))]

public class HealthUI : MonoBehaviour

{

public GameObject uiPrefab;

public Transform target;

float existingtime = 5;//血条持续时间

float lastchangetime;//上一次血条改变的时间点

Transform ui;

Image healthSlider;

Transform cam;

// Start is called before the first frame update

void Start()

{

cam = Camera.main.transform;

foreach(Canvas c in FindObjectsOfType<Canvas>())

{

if(c.renderMode == RenderMode.WorldSpace)//找到指定的canvas

{

ui = Instantiate(uiPrefab, c.transform).transform;

healthSlider = ui.GetChild(0).GetComponent<Image>();

ui.gameObject.SetActive(false);

break;

}

}

GetComponent<CharacterStats>().OnHealthChanged += OnHealthChanged;

}

void OnHealthChanged(int maxHealth,int currentHealth)

{

if (ui != null)

{

ui.gameObject.SetActive(true);

lastchangetime = Time.time;

float HealthPercent = (float)currentHealth/maxHealth;

healthSlider.fillAmount = HealthPercent;

if (currentHealth <= 0)

{

Destroy(ui.gameObject);//死亡时去除血条

}

}

}

private void LateUpdate()

{

if (ui != null)

{

ui.position = target.position;

ui.forward = -cam.forward;

if(Time.time - lastchangetime >= existingtime)

{

ui.gameObject.SetActive(false);//超出时间时不显示

}

}

}

// Update is called once per frame

void Update()

{

}

}

### 8.3.5战斗脚本实现

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

[RequireComponent(typeof(CharacterStats))]

public class CharacterCombat : MonoBehaviour

{

public float attackSpeed = 1f;

private float attackCD = 0f;

public float delaytime = 1f;//攻击延迟,根据后续动画调整延迟时间

const float combatCD = 5;

float lastAttackTime;

public bool InCombat { get; private set; }

public event System.Action OnAttack;

CharacterStats mystats;

CharacterStats opponantStats;

private void Start()//初始化该角色，将属性设置为mystats

{

mystats = GetComponent<CharacterStats>();

attackSpeed = 10f;

}

private void Update()

{

attackCD -= Time.deltaTime;//攻击CD随时间递减

if(Time.time - lastAttackTime > combatCD)

{

InCombat = false;

}

}

public void Attack(CharacterStats targetstats)//攻击实现函数

{

if (attackCD <= 0f)

{

//targetstats.TakeDamage(mystats.attack);

//StartCoroutine(DoDamage(targetstats, delaytime));

//

opponantStats = targetstats;

if (OnAttack != null)

OnAttack();

Debug.Log("攻击速度为" + (mystats.atspeed + 0.1 \* attackSpeed));

attackCD = 2f / (attackSpeed + mystats.atspeed);//攻击过后更新攻击CD

InCombat = true;

lastAttackTime = Time.time;//记录上次攻击时间，以方便攻击CD的实现

//mystats.atAS.Play();

}

}

//IEnumerator DoDamage (CharacterStats stats, float delay)

//{

// yield return new WaitForSeconds(delay);

// stats.TakeDamage(mystats.attack);

// if(stats.currentHealth <= 0)

// {

// InCombat = false;

// }

//}

public void AttackHit\_AnimationEvent()

{

float Distance = Vector3.Distance(opponantStats.transform.position , mystats.transform.position);

mystats.atAS.Play();

if (Distance <= mystats.range)

opponantStats.TakeDamage(mystats.attack);

if (opponantStats.currentHealth <= 0)

{

InCombat = false;

}

}

}