# 项目总结

## 实现的功能

|  |  |
| --- | --- |
| 玩家左右攻击 | √ |
| 玩家生命值 | √ |
| 玩家得分制 | √ |
| 僵尸随机移动 | √ |
| 僵尸定时生成 | √ |

## 核心代码说明

* 游戏模块

游戏添加了两个模块，Player模块和Zombie模块，用来创建新的Player和新的Zombie，使用闭包形式。（PS：使用元表一直出问题）

* Zombie模块

在构建时赋值：僵尸在table中的索引、图片ID、宽高、Y轴位置、左右方向、动画帧速度、行走速度。

函数返回：Render, GetPos, SetDead, isDead, CheckCollision。

Render：渲染僵尸到游戏画面

GetPos：获取僵尸位置

SetDead：设置僵尸死亡

isDead：判断僵尸是否死亡

CheckCollision：判断僵尸碰撞

* Player模块

在构建时赋值：图片ID、XY轴位置、宽高、动画帧速度。

函数返回：Render, KeyListener, isAttack, GetDirection, Destroy。

Render：渲染Player到游戏画面

KeyListener：对 A D键进行监听，如果按下改变人物方向并进入攻击模式

isAttack：返回人物攻击状态

GetDirection：获取人物方向

Destroy：销毁人物

* LuaGameHost

定义collider表存放左右攻击范围和人物受击范围。

定义zombies表存放Zombie模块生成的僵尸。

定义playerMethods表存放Player模块生成的游戏玩家。

在HandleFrame函数中进行对三种游戏场景的判断：开始游戏、进入游戏、游戏结束。

在进入游戏（HandleGameScene）函数中依次进行了：渲染得分、渲染生命值、创建僵尸、渲染僵尸、渲染玩家。

创建僵尸：使用定时器实现，定时1秒创建一只僵尸，通过对Zombie模块的调用，将返回的值用table.insert方法加入zombies表中。

渲染僵尸：通过使用迭代器对zombies表遍历，如果僵尸没有死亡则渲染，并对其与collider进行碰撞检测。

渲染玩家：如果playerMethods中没有渲染方法，则对Player进行调用，其返回值加入playerMethods。如果存在渲染方法，则对人物进行渲染并监听键盘。

## 耗费工作量

从图片素材编辑到以上所述功能完成，大概耗时6小时，每天大概1小时左右的时间编写代码。还有一个小时用来写项目总结。

## 完善计划

后续如果有时间会考虑加入“大招功能”，击败多少僵尸可获得1点充能、充能满5点可使用大招，大招可一次性消灭指定方向僵尸。另外还会考虑加入游戏音效。

## 学习体会

使用这个游戏框架写代码甚至不用编译，实现了所谓开箱即用。但是还是觉得C++加上Lua这个组合有点麻烦，可能是没什么大型项目开发经历。

许景智

2220631128

2220级数字媒体技术1班

2022.12.22