Wield——作业报告

# 游戏设计

玩法设计：

具体玩法仍然与之前设计的一致，是2D俯视角平面游戏，采取的为类似于吸血鬼幸存者的玩法。玩家需要在一张随机生成的有限地图上通过移动来躲避各式怪物的攻击，并通过不断围绕自身旋转的武器对敌人造成伤害。敌人的数量无限生成，且随时间增长将越来越难对付，玩家也可以通过击杀敌人获得的经验值来升级提升自己的能力。

设计思路：

首先确定本次设计参考的对象，即吸血鬼幸存者，随后试图在此基础上做出创新。联想到某小程序的海投广告，便决定将武器旋转作为攻击方式实现。随后便得出此设计。

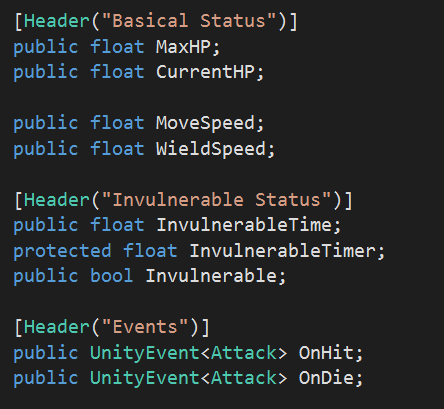
设计目的：

通过独立实现这么一个想法，来加深对2D游戏制作过程的了解，提升自己的能力。

# 程序实现

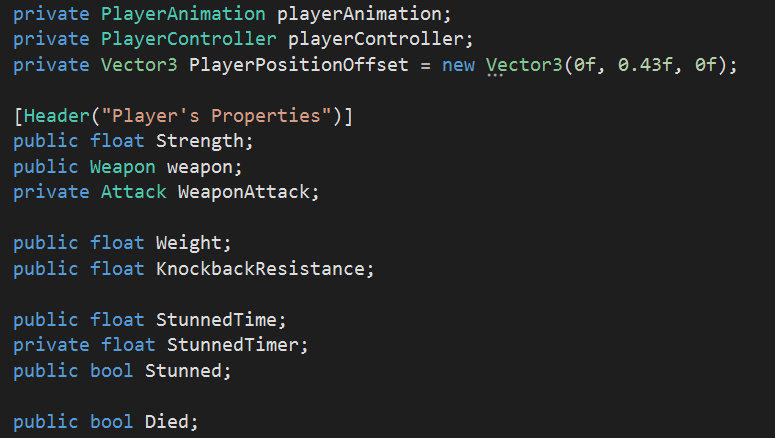
1. 角色类

角色类全部继承于Character基类，该类包含玩家和敌人的一些基本方法和属性，以及受击和死亡的事件。

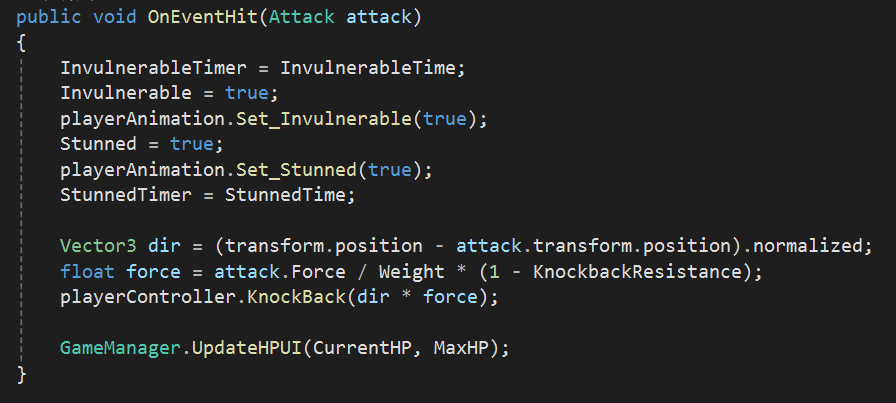


1. 玩家类

玩家类在基础角色类的基础上，增加了玩家特定的属性，以及玩家动画、玩家控制等组件，和对武器的交互逻辑实现。

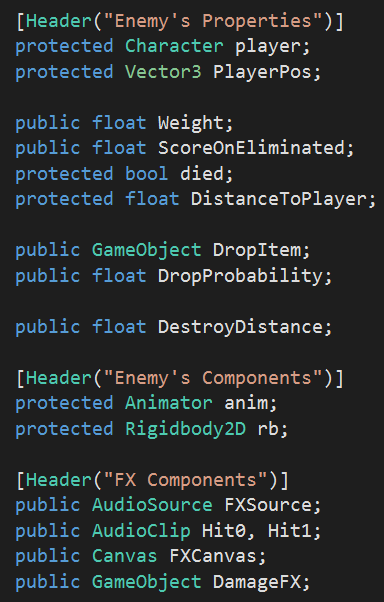


玩家类会通过自身的力量值时时更新自己的武器的攻击类，并且在受击和死亡逻辑中也会和其他类联动来修改UI和游戏状态等信息。



1. 敌人类

敌人类也有自己特定的属性与方法实现，来满足敌人类的通用需求。



还有受击死亡事件处理器，敌人状态逻辑设计，掉落物逻辑等等具体方法的实现。

各种敌人都继承于这个敌人基类，并在此基础上实现敌人种类所特色的方法。

敌人总共有三种，蝙蝠，大蝙蝠和蘑菇人。





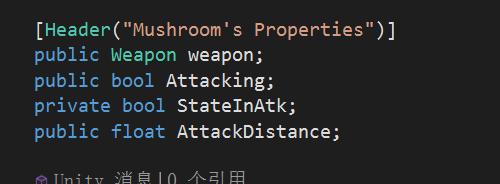


其中大蝙蝠是小蝙蝠的变体，只在体型和属性方面有所改变。

对于蝙蝠类，它的特殊技能是一个冲刺攻击。它会在远离玩家的时候进行充能，当接近玩家时概率触发进行冲刺来造成伤害。

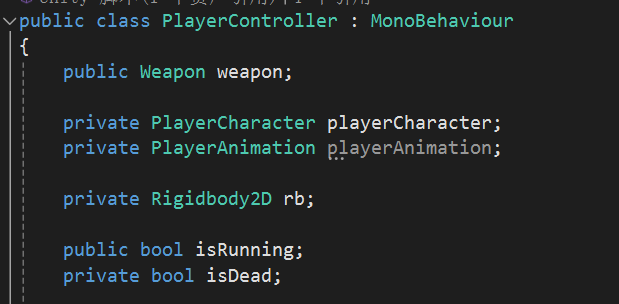


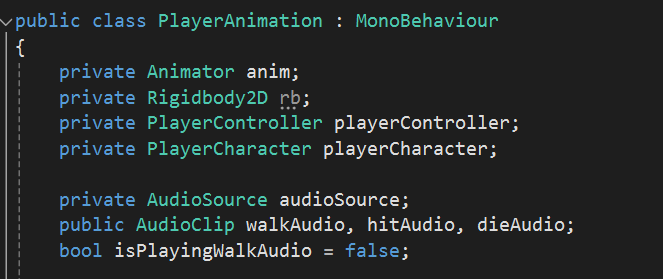
而对于蘑菇人，它除了自身的攻击手段之外另外也有它的武器来围绕它转动，并会在接触玩家的时候造成伤害。



1. 其他玩家相关类

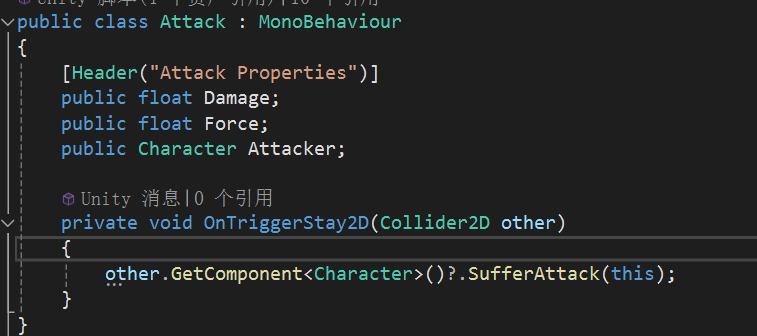
剩下的玩家类为Controller和Animator，前者操作玩家移动之类的逻辑，后者则是控制玩家的动画系统和音频系统。



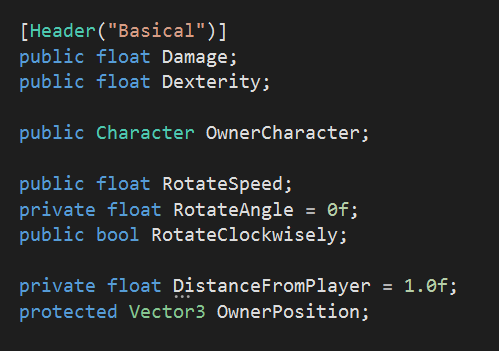


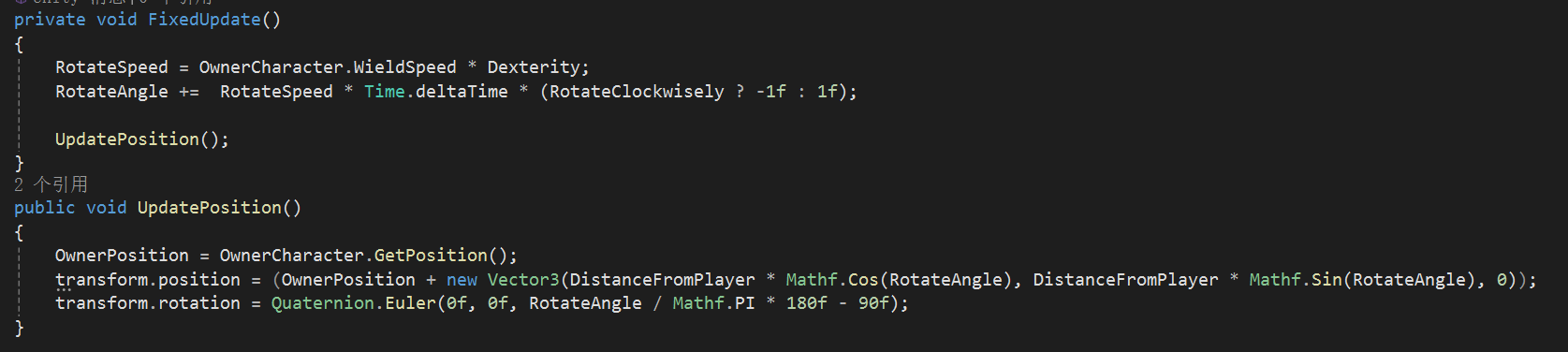
1. 武器和攻击类

攻击类包含攻击力，受力大小和攻击者属性，所绑定的物体具有碰撞体属性，而攻击类则只负责检测碰撞，并将自身作为攻击判定的参数传出。



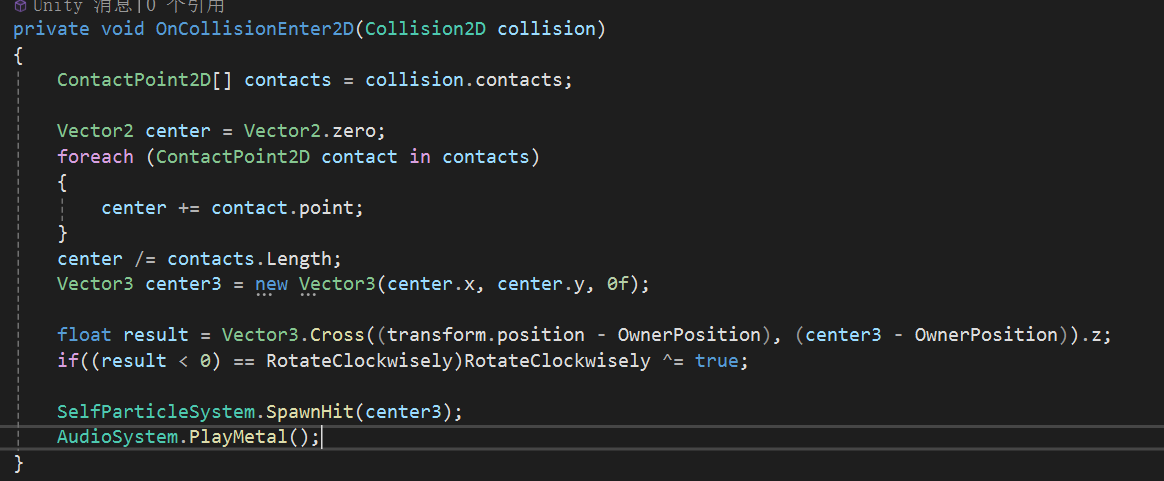
武器类则维护了它的攻击者，并在每次更新时围绕攻击者进行转动。



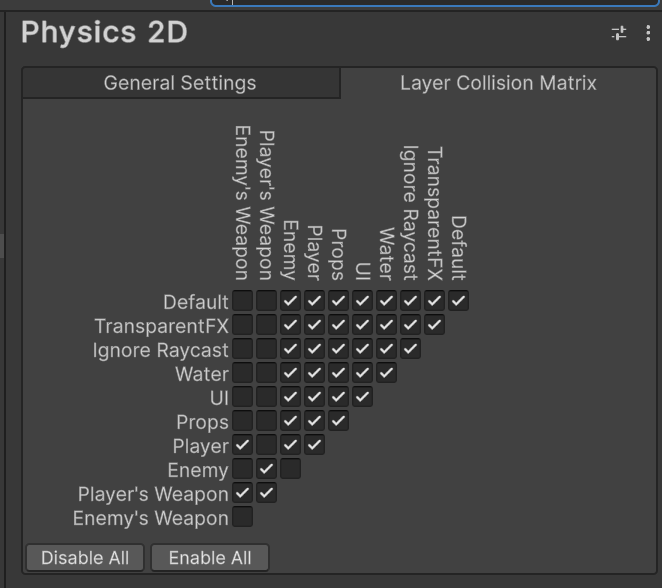


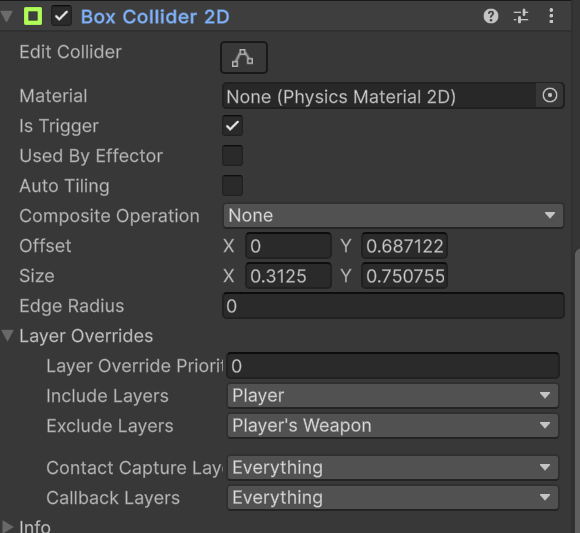
另外，武器类绑定的武器身上有两个碰撞体，其中一个为trigger，负责提供给攻击类进行攻击判定，另一个为碰撞体，和武器的刚体属性一起用来检测武器和敌人武器之间产生的碰撞。

当武器之间的碰撞发生时，计算出所有接触点的中心生成粒子特效和音频特效，并将接触点、旋转中心和当前旋转方向进行计算，来判断武器是否需要反转旋转方向。



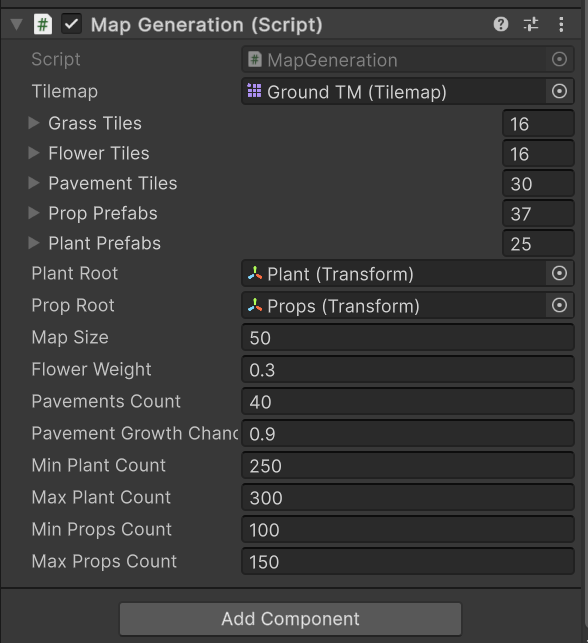
由于实现在Physics中设定过层之间的碰撞并且也在两个碰撞体上设置了层的遮罩，因此攻击类和武器类只会在正确的对象层中产生碰撞。



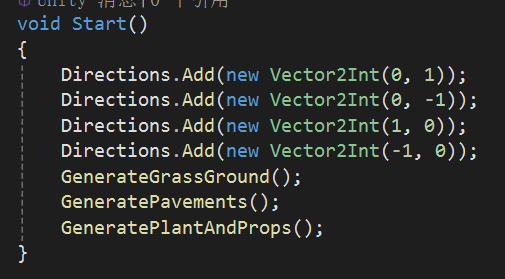


1. 地图相关

整个2D地图由瓦片地图构成。在最开始，会有个MapGenerator对象，通过它挂载的地图生成脚本来对地图进行生成。生成完毕后，这个对象就会自动销毁。在地图最边缘的城墙部分则是手动绘制的，并将其位置作为地图大小传给地图生成脚本。



而地图生成的逻辑如下：



第一步，将地面的花草瓦片随机填充满整个地图。

接下来，在地图上随机生成人行地砖。地砖生成的逻辑为在地图上随机选取Pavement Count个点作为地砖生成起点，然后从起点开始向四个方向不断随机生长，生长后的地砖则向另外三个方向继续随机生长。这样便能生成一堆路径。

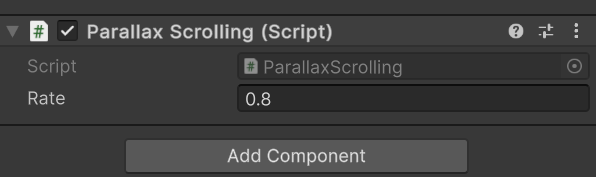
第三步，就是随机生成各类植被和物件。

三步完成后，地图就算生成完毕，MapGenerator销毁。

在地图的最边缘，有天空盒显示的场景和一个手工放置的浮岛群。

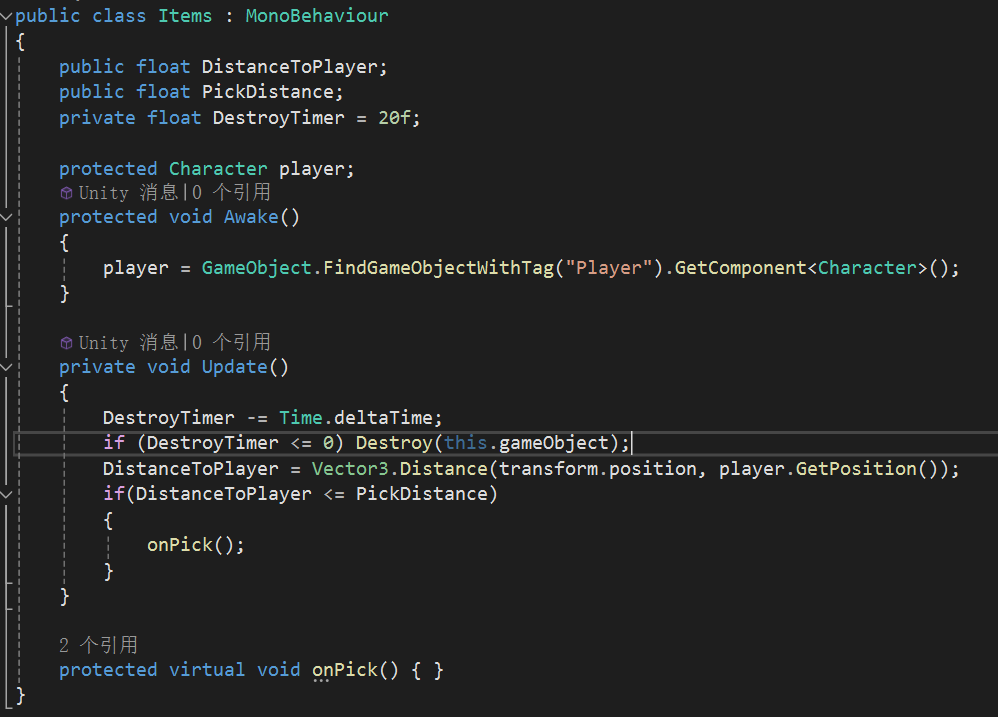


浮岛群中上下部分的石头用来做视察卷轴。通过脚本控制随玩家x方向移动的倍率。

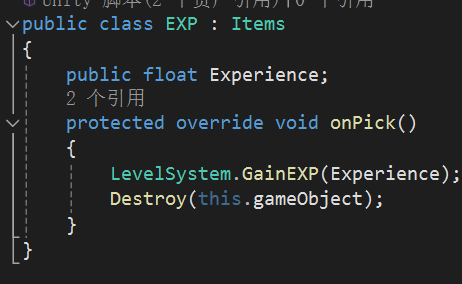


1. 等级系统

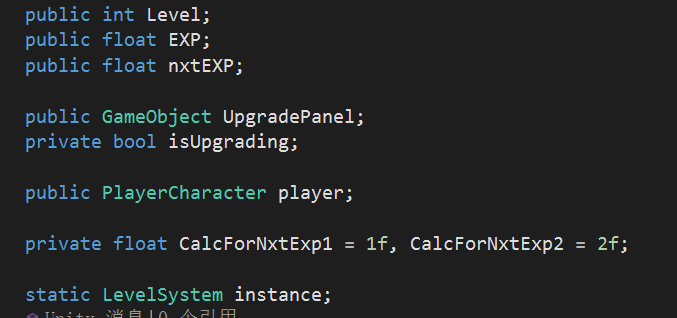
掉落物类会判断与玩家之间的距离来调用捡起方法。自身也会在超出时间限制后自我销毁。



经验类则继承于掉落物类，它的捡起方法实现为获得经验。



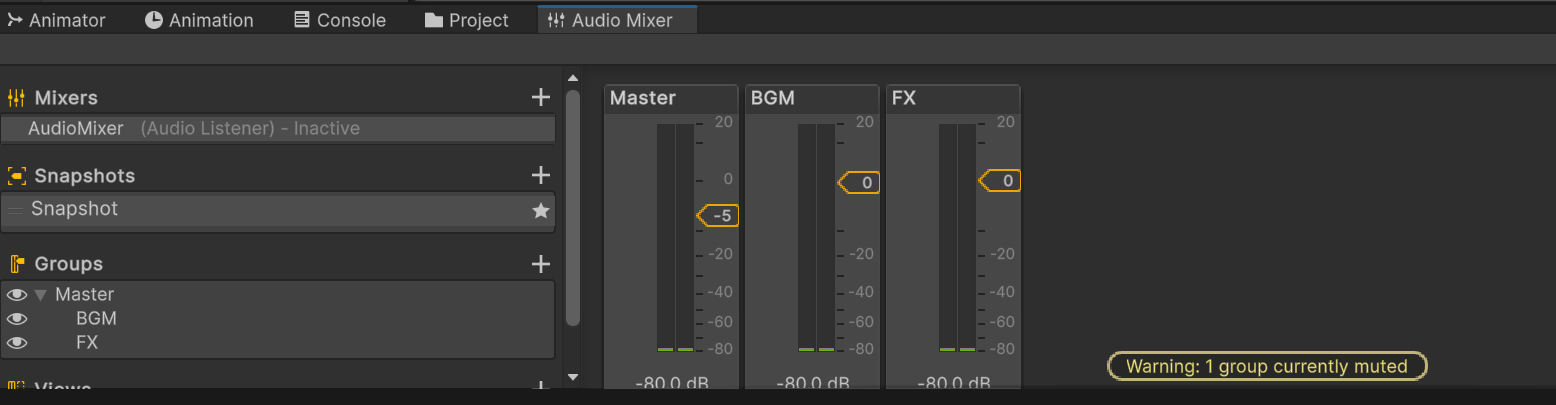
对于等级系统，它的实现为单例模式。它负责在获得经验时更新UI，当升级时暂停并跳出加点菜单，并在玩家加点后更新玩家的属性。当玩家等级提升后，升级所需的经验会越来越高。



1. 音频系统

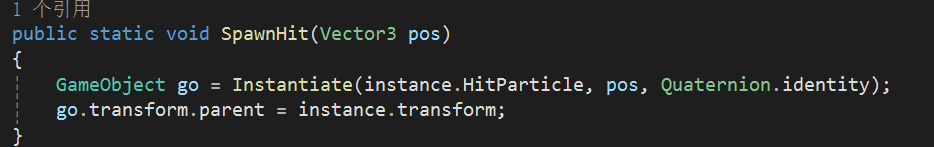
音频系统也为单例模式，负责一个BGM的播放和一个音效的播放。音频系统负责播放的音效大多为系统音效。各类角色自身的例如死亡、受击、移动等的音效为角色自己的脚本来控制角色自己的声源进行播放。

另外，外部也有一个混音器控制总音量，bgm音量和各类音效音量。



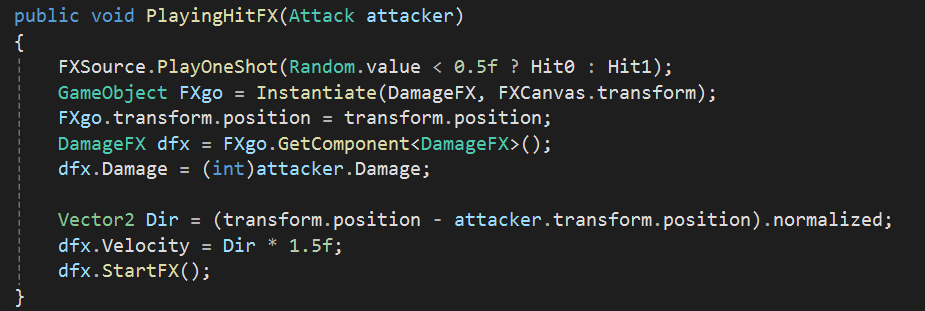
1. 视觉特效系统

视觉特效系统包括粒子特效系统。粒子特效系统的实现为单例模式。它实现了一个静态方法用来在特定位置生成一个特效的预制体。



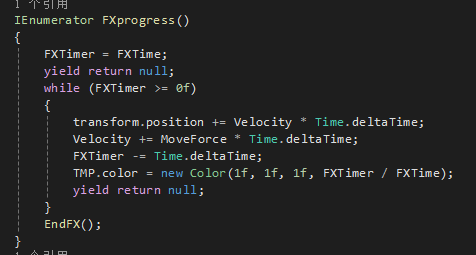
此处它用来生成武器碰撞时产生的撞击特效。

另外，还有一个敌人受击时的数字特效。

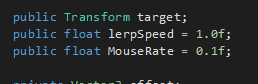


此部分特效为手工制作。设计的特效为在敌人受击部位产生一个显示伤害数值的数字并产生动画效果。

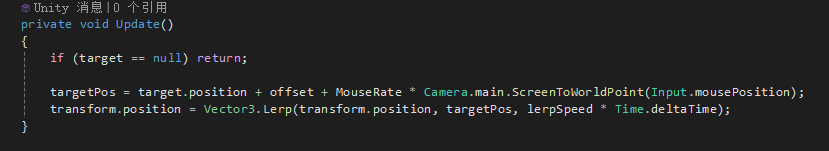
为实现此效果，首先需要一个canvas来显示特效中的text部分。该canvas与其他ui的canvas不同，设置为世界坐标显示。随后创建特效的预制体，并将特效写为协程，作为脚本挂载在预制体上。这样预制体在生成的时候就会自动执行预期的动画并销毁。



1. 镜头跟踪

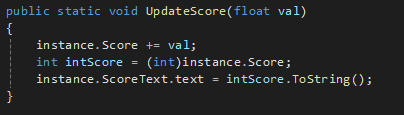


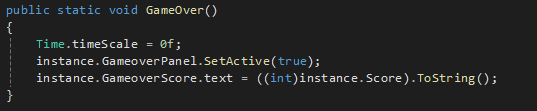
将镜头的目的坐标设置为角色坐标加上鼠标坐标按一定权重的偏移。这样在每次update时就只用通过Lerp来移动镜头就可以达到镜头缓慢跟踪的效果。



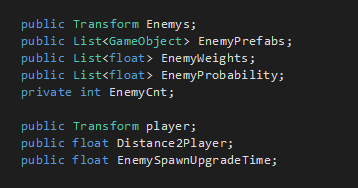
1. 游戏管理

游戏由GameManger对象进行管理。该管理也为单例模式。主要功能有实现对UI调用修改的接口，分数的计算，以及游戏整体的暂停重开之类的实现。





另外，由于游戏设计为无限的战斗模式，因此怪物生成系统也需随时间改变生成逻辑。



基本逻辑为通过协程来每搁一段固定的时间生成怪物。在每次生成的时候，根据当前游戏进行的时间来获得一个生成权重总和，作为当前时刻生成的怪物的衡量。越强力的怪物生成所需要的权重越高。当生成时，会从高级怪物中遍历，通过一次概率检定后若是成功生成则更新剩余权重总和，以此来在此刻生成符合时间所对应强度的怪物集合。



生成时，只需在以玩家为中心一定半径的圆上随机挑选一个点生成即可。

# 参考资料

1.《勇士传说》横版动作类游戏开发教程-https://www.bilibili.com/video/BV1zY411B7KH/?share\_source=copy\_web&vd\_source=59a8d2af3e8686dd62c829f07183caf4

2. Unity官方手册