项目报告

周晴阳，赵麒安

2025 年 1 月 12 日

角色：

主要角色 Player:

在本游戏中，Player 主要有以下互动行为：

根据坐标在场景中渲染（见镜头与渲染部分）

通过 update 函数实现按 WASD 在场景中移动和墙体碰撞监测

向着对应方向每刻移动等同于玩家速度大小的距离（这会涉及到场景相机的迁移，在本文其他部分会详细说明），同时在每一个游戏刻内若检测到玩家刚才的移动使玩家将要碰撞到墙体的碰撞体积，执行回退一步的操作来实现墙体碰撞功能。

按 I 键打开/关闭物品背包：

当玩家按下 I 键时切换背包的开闭状态，在 show_inventory 函数中通过预设好的背包背景（淡蓝色半透明方块）并读取玩家此时的属性和库存将玩家的物品和各种详细信息打印在屏幕上。为了防止游戏后期过多物品显示不全，加入了使用 pgup/pgdn(通过测试，部分电脑还需要按住 Fn 键再按上述按键)来改变文字 y 轴偏移值从而实现视觉上的翻页效果。

获取物品：玩家的 add_item 函数在玩家通过购买或击败 NPC 获取新物品时被调用，效果是为玩家的属性值加上对应物品的属性。

通过 draw_item_message 在当前游戏窗口的中心位置短暂的显示刚才获取的物品的名字，达到提醒玩家获得新物品的功能

在 stage1 函数下，有两行全物品指令供测试用。

关于玩家与 NPC，Transparent，Portal 类的交互，在他们对应的模块将会详细阐释

NPC:

在本游戏中，NPC 主要有以下互动行为：

根据坐标在场景中渲染（见镜头与渲染部分），每个 NPC 都拥有独立的场景内像素小人贴图

在与玩家 Player 的欧式距离小于设定数值时，启用所有交互，包括：

1. 聊天气泡循环显示：通过读取 setting.py 中对应 NPC 的聊天气泡列表，每个 NPC 在初始化后都具有独一无二的气泡文本内容。通过 pygame.time 调用当前时间，在主循环中使用 switch_bubble 函数每隔 120 帧切换一次气泡内容并调用 scenemanager 相关函数进行渲染（见 scenemanager）

2. 对话，战斗，商店窗口：在主循环中监听玩家的交互行为，将非交互状态下的所有输入交由 npc 的 handle_input 函数进行处理：使用 self.dialogue/fight/buy_active 标记自身的交互状态，当自身未被激活时，根据对应的功能键 e/f/b 激活交互状态，在主循环中监听到对应交互状态启用时，调用相关函数

对话相关：通过抓取键盘输入作为对话输入，Esc 为退出对话指令（即关闭对话激活），enter 为提交文本。所有 NPC 通过 setting.py 中预设好的人物性格，剧情内容和游戏通识内容作为 openai 预设输入，并将玩家的对话输入提交至 openai 得到反馈，将问答文本整理在 dialogue_history 列表中并在 scenemanager 中根据 NPC 激活对话状态用调用 draw_dialogue 函数，根据对应的 NPC 聊天背景（同样每个 NPC 有个性化的背景）和对话内容绘制对话窗

口。同时，为了防止对话次数过多导致的显示问题，同样也加入了基于监听 pgup/pgdn 的上下翻页功能。

战斗相关：监测到玩家在自己附近按 f 且自身未被击败时，激活战斗状态，并在主循环中根据激活状态绘制对应的战斗窗口：每个回合由以下事件构成：先在 handle_input 函数中监听 q 键和 e 键的点按次数并计数，当次数到达回合行动上限时调用 fight_calculate 计算玩家和 NPC 下一回合的血量值。回合结束，回合数+1，当玩家血量小于等于 0 时触发 fight_failed 函数，退出战斗窗口并初始化所有战斗参数（和打开战斗以前不会有任何区别），当玩家血量大于 0 且 NPC 血量小于 0 时触发 fight_succeeded 函数，将 npc 设置为 defeated 并结算战斗奖励同时初始化战斗参数；战斗窗口的绘制由 scenemanager 根据战斗激活状态调用 draw_fight 函数执行：在屏幕左右分别是玩家和 NPC 的贴图，血量数值，红白条血量百分比图像，中央会显示回合数，玩家的下方会根据 q,e 键的监听次数显示该回合已经打出的攻击牌数目和防御牌数目（为了可视化，采用两张画有盾牌和剑的贴图，贴图数目代表打出的牌数）同时界面正下方有提示文字引导玩家攻击与防御的按键是 e 和 q
战斗部分有独立背景音乐，见背景音乐部分

商店相关：监测到玩家在自己附近按 b 时激活商店状态，并在主循环中根据激活状态绘制对应的商店窗口：每个 NPC 读取 setting.py 中自己的商品列表作为商品，调用数字键 1，2，3 作为购买对应物品的快捷键，同时在窗口上绘制半透明的黑色商店背景和物品文字信息，按照“快捷键--物品信息--价格”的结构从上到下显示所有商品，同时当玩家购买了上层物品时会重新排列，即若购买物品 1，则物品 2，3 的快捷键和显示行数将变为 1，2
同样是使用 scenemanager 根据商店激活状态调用 draw_buy 函数进行绘制

Transparent 类：可变 alpha 值类，作为屋顶使用，通过 check_transparent 函数监测是否与玩家重合，若重合则提高透明度以实现“玩家在屋顶下能看到屋子内的场景，玩家在屋子外只能看到屋顶”的视觉效果，渲染通过 scenemanager 实现

Portal 类：通过 check_telepotation 和 check_item 函数实现当玩家与自身重合并按下交互键 e 时，若玩家库存中所需的剧情物品不够，则调用 tp_failed 函数弹出白色窗口提示玩家还缺少哪些物品，若玩家已经具备足够条件，则激活自身的 tp_succeeded,并在主循环中监听此特征值并跳出当前循环切换到下一个 stage

Items 类:游戏中所有物品通过此类定义， 初始值包括数值列表 statistics 和描述 discription, 其中 statistics 包括对玩家 attack,defend,health,moves 的更改量，通过玩家的 add_item 函数实现，而 discription 是我们创作时对各个物品的吐槽和简介，在打开 i 键背包时能够看到所有物品的 discription