成都东软学院

基本能力实训报告

|  |  |
| --- | --- |
| **实训项目**： | 基于Python重构的植物大战僵尸 |
| **指导教师：** | 贺敏 |
| **学 院** | 计算机与软件学院 |
| **系 别：** | 数据科学与大数据技术系 |
| **专业班级：** | 大数据22205 |
| **学 号：** | 22010820502 |
| **学生姓名：** | 陈兆昂 |
| **实 训 期：** | 2022-2023学年第3学期 |

**2023 年 9**

**实习周报**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 姓名 | 陈兆昂 | 班级 | 大数据22205 | 学号 | 22010820502 |
| 项目  名称 | 基于Python重构的植物大战僵尸 | | | 时间 | 第1周 |
| 本周学习内容 | 周一：pygame.event(一)  周二：pygame.event(二)  周三：pygame.event(三)  周四：pygame.mouse  周五：pygame.key | | | | |
| 本周完成任务 | 周一：①项目资源的收集与整理  ②植物的相关属性收集  周二：①整体框架的搭建  周三：①背景类的创建  ②植物父类的创建  ③植物父类射击逻辑的引入  ③创建植物子类--太阳花，豌豆射手，寒冰射手  ④子弹父类的创建  ⑤创建子弹子类--普通豌豆，寒冰豌豆  ⑤阳光父类的创建  ⑥创建阳光子类--太阳花产生，天空飘落  周四：①创建植物子类--双发豌豆，坚果墙，土豆地雷  ②创建子弹子类--双发豌豆类  周五：游戏细节的优化   1. 解决动画帧播放速度过快的问题 2. 添加太阳花产光前短暂变亮的提示动画 3. 射击函数的优化 4. 土豆地雷状态切换时添加土壤动画 5. 将植物相关属性以字典的形式写入配置文件 6. 解决了当局阳光数无法居中的问题 | | | | |
| 心得体会 | 总结：本周项目整体难度较低，在项目中消耗的时间更多是在丰富专业知识方面，例如对pygame中sprite，rect，mouse这些类的学习，对面向对象编程中类的继承有了更多见解。  针对难点1—①通过传入的子弹数这一参数创建一个类似“弹夹”的列表，列表前n-1发子弹对应为True，第n发为False，故当每次发射子弹时就索引这发子弹索引的前一位，True为短延迟，而False则为长延迟  针对难点1—③在索引为False后，将参数last\_fire减n-1个短延迟  针对难点3我在配置文件中添加了一个帧数计数器（主循环每循环一次，计数器加一），然后在需要调整动画帧的精灵中，设置专属的fps，在其本身的update中当计数器模自己的fps不等于零时，则跳出update方法  难点：1.统一不同射击模式的函数   1. 单发豌豆的射击延迟固定不变，多发豌豆的射击延迟先短后长如何仅通过一个参数（单发射击数）合理调整射击延迟 2. 射击类植物的首发子弹不计射击延迟，（与太阳花首颗阳光的产生延迟较短一样）如何进行首发判定 3. 多发每次的第一发理论上应与单发的每一发射击时间大致相同，如何调整多发豌豆的射击时间 4. 将豌豆的射击延迟设置为区间（在原版游戏中，与阳光的产生延迟类似都是区间形式）每次的射击延迟在区间内随机产生   2.植物与天空的不同产生、消失逻辑   1. 植物产生的需要随机左右位置 2. 天空飘落的需要随机飘出的位置以及位置检测，防止飘出游戏场景外 3. 不同类型阳光的存在时间计算   3.游戏各精灵动画帧过快 | | | | |
| 备注 |  | | | | |

**实习周报**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 姓名 | 陈兆昂 | 班级 | 大数据22205 | 学号 | 22010820502 |
| 项目  名称 | 基于Python重构的植物大战僵尸 | | | 时间 | 第2周 |
| 本周学习内容 | 周一：pygame.surface(一)  周二：pygame.surface(二)  周三：pygame.surface(三)  周四：pygame.image  周五：pygame.rect | | | | |
| 本周完成任务 | 周一：①小车，卡片栏，铲子等元素的引入  ②飘落阳光的具体实现  周二：①引入阳光的拾取功能  ②引入当前阳光数元素，并使其能通过实时监测来修改当前阳光数  周三：创建卡片父类并添加以下功能  ①冷却阶段覆盖动态阴影  ②种植判定（可种后卡片恢复原色）  ③为卡片添加对应的植物精灵  周四：①选中卡片卡片后植物跟随鼠标  ②植物的种植位置提示  ③种植植物的逻辑  ④添加铲子铲除植物功能  ⑤添加卡片子类（所有植物卡片）  周五：游戏细节优化   1. 引入回收的阳光存在图像缩小消失效果 2. 调整回收的阳光速度，使其先快后慢 3. 调整阳光数的居中效果 4. 添加代码注释 5. 实现单一方块土地对应单一植物 6. 在Sever中添加检测鼠标是否被选中的方法，（精灵Card、Sun、Shole返回选中状态） 7. 将卡片的ready判断中CD与sunnum的检测分开处理 | | | | |
| 心得体会 | 总结：本周工作的整体难度相较第一周有所提升，因实现功能需要，接触了更多从前未接触的板块。但苦难之余也因此丰富了自己的技术。例如：我需要了解pygame中的event板块来做事件处理，学习pygame中rect的碰撞检测来实现阳光拾取、植物铲除  针对难点1—①我学习了pygame中的event板块，来实时检测我的输入状态，当获取到的事件满足回收条件时，修改参数self\_recycle为True，否则为False  针对难点1—③我最终在sunlight中先添加handle\_event方法来实现具体的参数修改，随后在sever中添加handle\_event方法，使其实时调用sunlight中的handle\_event方法，并传入事件队列，紧接着在scene中添加handle\_event方法来调用sever中的handle\_event方法，最终在frame的主循环中循环调用scene中的handle\_event方法  针对难点2我尝试如果参数self\_recycle为True就实时检测sun的位置当其进入缩小区域时，每次的大小是前一次的大小-8像素  针对难点3我尝试如果参数self\_recycle为True就实时检测当前rect到回收位置的距离，回收速度就是两点间的距离除10  针对难点5我根据不同植物卡片的CD计算出阴影面积每减小一像素所需时间，然后每达到改刷新间隔，让阴影在卡片上的y轴绘制位置减一像素；其次在每次刷新阴影时先绘制卡片再绘制阴影，而不是在卡片上重复绘制阴影  针对难点6，我在Card中创建了三个参数分别是is\_ready、isselected、is\_grow\_plant当满足一定条件时当前者也同时满足时，更新自身  难点：1.回收阳光的逻辑  ①如何合理的实现回收的逻辑（判断事件、执行回收等行为写在哪里更为合适）  ②阳光的handle\_event逻辑写好后，具体的回收逻辑如何实现  ③sunlight—scene—sever—frame之间如何关联起handle\_event  2.阳光回收图像逐渐缩小的效果  ①在原版游戏中阳光在飞向左上角的回收框内时，图像是逐渐缩小的回收效果  3.阳光回收速度由快变慢  ②在原版游戏中阳光在飞向左上角的回收框过程中，速度是逐渐放慢的  4.铲除植物的逻辑（与回收阳光类似）  5.卡片的阴影在刷新中无法变小，且随时间颜色逐渐加深直到纯黑  6.植物跟随、位置提示、种植植物三者之间的关联 | | | | |
| 备注 |  | | | | |

**实习周报**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 姓名 | 陈兆昂 | 班级 | 大数据22205 | 学号 | 22010820502 |
| 项目  名称 | 基于Python重构的植物大战僵尸 | | | 时间 | 第3周 |
| 本周学习内容 | 周一：pygame.sprite(一)  周二：pygame.sprite(二)  周三：pygame.sprite(三)  周四：pygame.music  周五：pygame.time | | | | |
| 本周完成任务 | 周一：①不同状态下僵尸图片的重新整理  ②僵尸的相关属性的收集  周二：创建僵尸父类并添加以下功能  ①移动（正常移动、受伤移动）  ②啃食（正常啃食、受伤啃食）  周三：①引入僵尸的普遍受击逻辑  周四：①引入僵尸的特殊受击逻辑  ②僵尸碰撞小车逻辑  ③引入寒冰豌豆子弹的功能特性  ④引入僵尸遭遇爆炸性伤害后的特殊死亡动画  ⑤引入小车接触后被压扁的动画  周五：游戏细节优化   1. 调整豌豆出射位置，防止僵尸啃食时碰撞效果异常 2. 修正了植物遭遇啃食时子弹无法正常破碎的bug 3. 为植物父类添加抽象攻击方法，方便服务器调用植物组攻击 4. 为子弹添加成功攻击后的破碎效果 5. 将植物与僵尸取消关联（创建类时无需传入对方精灵组） 6. 将子弹类与僵尸，铲子类与植物取消关联 7. 优化配置文件中不属于游戏框架的常量位置（将有关常量写在各自对应的类中，通过python中的装饰器，使在各文件中调用） 8. 现在豌豆类植物仅在该路存在僵尸且在植物之前时才进行射击 9. 引入僵尸血量低于90后添加尸头的落地动画 10. 引入僵尸血量到达0时添加僵尸的倒地动画 11. 重写坚果墙状态的判断代码 | | | | |
| 心得体会 | 总结：本周整体上的工作难度较上周相似，项目内容稳定更新。其中难点中的1、3、7、8在实现上花了不少时间，需要经常查阅方法、资料来尝试实现自己的想法，最恶心之处在于，在运用新方法时，常常面对报错而不知所云，不过也正是通过一点一点的学习和尝试的过程，自己对相关方法运用更加熟悉，理解更加深入，学习到了很多心得知识。  针对难点1我尝试在类里创建内部类实现  针对难点3我学习了自定义碰撞是如何实现的，并且复习了这学期学习的匿名函数这一知识点  针对难点7我尝试将类中需要其他类参数的方法重新定义一个类方法，在调用类方法时出入需要的参数，而不是在初始化时传入这些参数  针对难点8我学习了python中的各种装饰器，最终选择通过@staticmethod这一装饰器，实现需求  难点：1.在image上添加额外surface时，此时如果surface的大小大于image时无法将完整的surface贴到image上（僵尸掉落的头，土豆地雷生长时冒出的土壤）  2.僵尸精灵的4大基本状态之间的切换（移动，受伤移动，啃食，受伤啃食）  3.由于僵尸的image图片原比单单一个僵尸要大，所以此时需要自定义碰撞的位置逻辑（def或者lambda的两种实现）  4.如何设置触发条件以实现在满足对应条件时，切换僵尸不同状态的动画  5.寒冰豌豆特性的实现（减速），以及如何正确更新僵尸的移动速度  6.土豆地雷的攻击逻辑（触发为单个僵尸碰撞，伤害为范围伤害）  7.如何在实例类时，尽量不用或者少用其他类对象的参数，以达到削弱类与类之间的直接关联  8.优化配置文件（如何将各种类的常量配置，在保证其他文件在不实例该类的前提下，访问到需要的常量信息） | | | | |
| 备注 |  | | | | |

**实习周报**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 姓名 | 陈兆昂 | 班级 | 大数据22205 | 学号 | 22010820502 |
| 项目  名称 | 基于Python重构的植物大战僵尸 | | | 时间 | 第4周 |
| 本周学习内容 | 周一：pygame.sound  周二：pygame.mixer。  周三：pygame. transform  周四：pygame.font  周五：pygame.display | | | | |
| 本周完成任务 | 周一：①引入僵尸的生成逻辑  周二：①引入关卡波次提示动画  ②引入游戏结束提示动画  ③引入关卡进度的UI动画  周三：添加游戏场景  ①加载场景，欢迎场景，主场景  周四：①暂停场景，获胜场景，失败场景  ②引入按钮类  周五：游戏细节优化   1. 使某些植物卡片在首次种植时不计CD 2. 解决植物被铲子铲除、土豆地雷爆炸消失后当前地格无法继续种植植物的bug 3. 调整各精灵的图层绘制顺序 4. 调整僵尸头的前进逻辑（由最开始的参照时间，调整为参照僵尸的死亡数） 5. 设置僵尸在不同状态下采用不同fps 6. 引入当进入下一波时，小旗升起的动画 7. 调整僵尸的生成逻辑（由最开始的所有僵尸同时生成，调整为当上一波僵尸全部死亡后，再刷新下一波） 8. 调整子弹、小车与僵尸的碰撞检测逻辑（自定义专属的碰撞检测区域） 9. 优化小旗的生成逻辑（由硬编码，调整为传入关卡列表自动生成） 10. 修改子弹类的代码逻辑（使初始化无需碰撞僵尸参数，与僵尸的碰撞检测调整到植物的sttack方法中） 11. 添加游戏音频 12. 播放开场动画时，暂停sever的update 13. 服务器添加游戏结束事件 14. 优化按钮类的代码逻辑，使其更加通用化 | | | | |
| 心得体会 | 总结：本周为工作强度最高的一周，同时也是游戏功能的扩展得最丰富的一周。  针对难点1中僵尸的生成位置，需要结合游戏帧数、僵尸移速、僵尸自身刷新率、窗口大小、波次间隔时间等进行综合计算，导致由于参数较多运算比较复杂，最终计算出的就是具体的像素位置  针对难点2，解决方法是由最开始的在创建的等大小surface表面上bilt图片后再transfrom图片尺寸，改为直接transfrom图片尺寸，既简化了代码，同时也美化了显示效果  针对难点3，实现上是在init时，需要传入按钮的显示图片或文字内容，初始化阶段会检测传入的是哪种类型，当传入图片时直接在按钮的背景图片上bilt按钮的显示图片，当传入的是文字则先渲染文字，然后再在按钮的背景图片上bilt文字内容。按钮选中的状态切换设计为，图片类型需传入包含，选中与未选中的图片列表，文字类型则为字体颜色选中为红，未选中为绿  针对难点4，则是使用pygame.sprite. LayeredUpdates()来创建全精灵组，因为在往这个组中添加精灵时，可以设置其所在的画布层级  针对难点5，由最开始的随时间移动，调整为随僵尸的死亡数移动，这样做的好处在于，在原版游戏中，当前一波还有未被击杀的僵尸时，后一波是不会刷新的，这就使得僵尸的生成应取决于僵尸而不是时间。故我在sever中根据玩家游玩的关卡，先自动计算出单个僵尸对应僵尸头移动的像素长度，然后实时记录玩家当前的僵尸击杀数，并用击杀数\*单个僵尸的像素长度作为僵尸头的移动距离，来更新僵尸头的移动  针对难点6，与难点5类似，只有当当前波次中，僵尸组中的僵尸数为零时才会刷新下一波僵尸  针对难点7，由于图片在旋转时其本身的尺寸会发生变化，这就导致图片的topleft会因为旋转而发生变化，故我们不采用原图片的topleft作为参照点，而是将centerx在每次循环中加上精灵的speed，就可解决  针对难点9，为所有地格添加初始为True的种植状态参数，当该地格上种植植物后，状态参数修改为False，并将植物kill的条件设置为blood等于0，当植物blood等于0时，当前地格的种植状态改为True  难点：  1.游戏关卡的设计需要考虑非常多因素，例如：僵尸种类的分配、生成位置的计算、生成位置是否合理、关卡整体的趣味性等等  2.图片放大，缩小后，图案比例异常  3.按钮类的通用化，按钮分为—文字类、图案类。如何把这两类按钮统一起来，实现不管是文字还是图案，都可以用这一按钮类来创建  4.画布上精灵的正确绘制顺序  5.僵尸头的移动逻辑  6.不同波次僵尸的生成逻辑  7.图片的平滑滚动  8.把音效加入sound队列  9.植物在非啃食死亡的情况下，所种植的地格无法重复种植植物 | | | | |
| 备注 |  | | | | |

# 实习内容及知识应用

## 1.1实训内容以及进程安排

|  |  |  |
| --- | --- | --- |
| 时间段 | 学习、工作安排 | 完成情况 |
| 第1周 | 学习pygame库 | 已完成 |
| 第2周 | 开发小游戏——飞机大战 | 已完成 |
| 第3周 | 搭建游戏项目框架、植物部分 | 已完成 |
| 第4周 | UI部分 | 已完成 |
| 第5周 | 僵尸部分 | 已完成 |
| 第6周 | 代码的整理、重构、优化 | 已完成 |
| 第7周 | 创建项目所需的场景 | 已完成 |
| 第8周 | 游戏细节优化、音频添加 | 已完成 |

## 1.2知识（点）应用以及经验总结

知识点：pygame库、游戏开发框架、面向对象的程序设计方法、状态机的设计模式

（以下是一些涉及知识点的代码切片）

游戏开发框架

config-配置、entry-入口、frame-主程序、resources-资源、scene-场景、sever-服务器

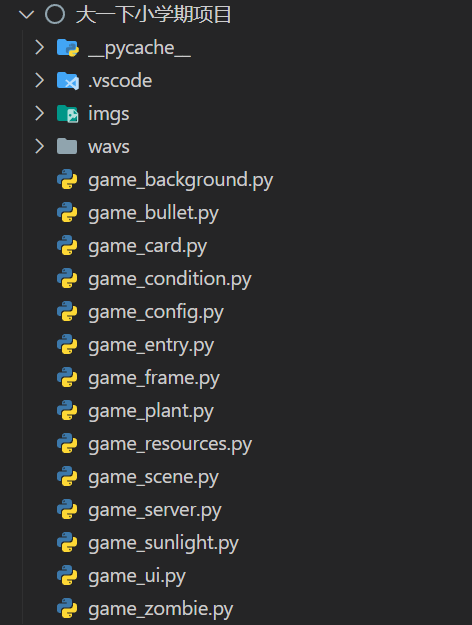


图1.1

面向对象

\_Plant-植物父类、各植物子类

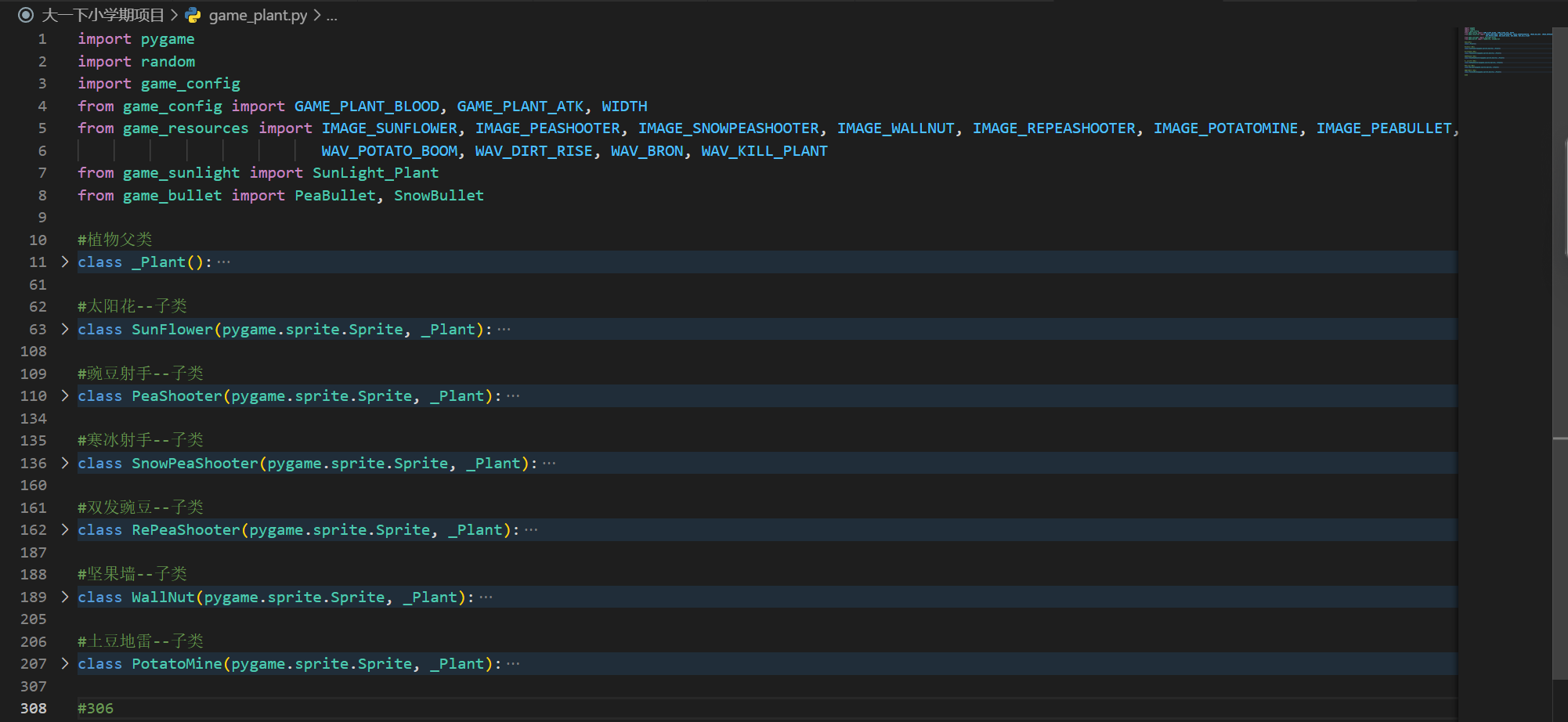


图1.2

状态机

加载场景、欢迎场景、主场景、游戏场景、获胜场景、失败场景

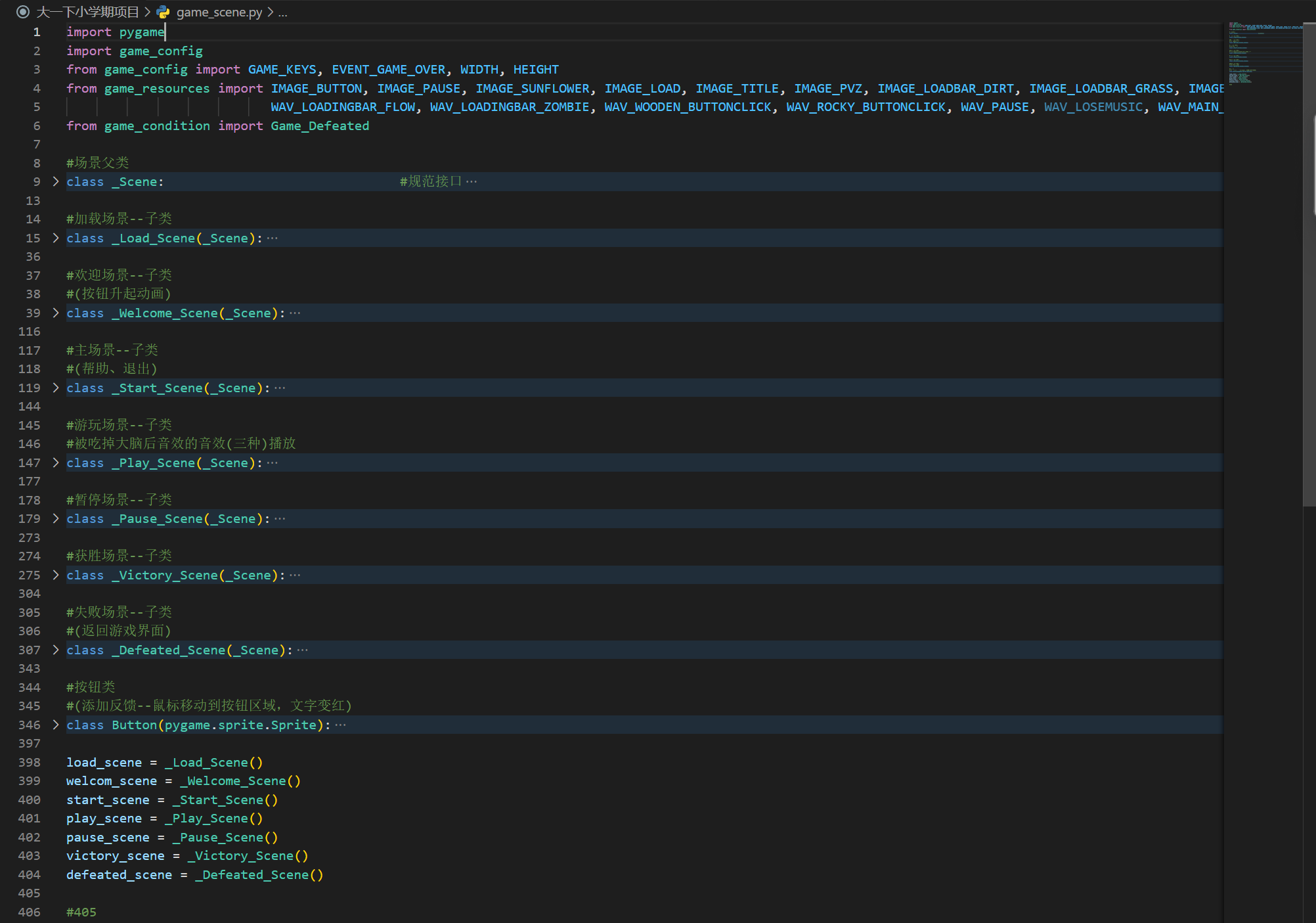


图1.3

个人总结：通过本次项目的开发，提高了编程水平，拓展了编程思想，对大体量代码的掌控能力也有了很大进步----详见 2.学习心得

（并附两张暑假在家期间的游戏开发动态）



图1.4

图形用户界面

描述已自动生成

图1.5

# 学习心得

一、实训第一周，在指导老师的带领下学习了python的pygame库，并在课上随指导老师，完成了很多小项目。在每个小项目的完成过程中都会学习到很多新的游戏开发思想与方法。例如：

1.在draw\_olympic\_rings中学习了

①实现动画的主体框架

②如何创建游戏窗口

③创建画布以及线条、图案的绘制等

2.在draw\_polygon中主要是对update这一方法有了更深的认识

3.在imagedemo中

①接触到了pygame中sprite这一重要板块

②image和rect这两个非常重要类

4.实训第二周，跟随老师完成了“飞机大战”这一游戏项目，在开发过程中，学习了如何搭建游戏框架（涉及配置文件、资源文件、游戏主要元素、服务器、场景、主程序、游戏入口）。对上一周所学的内容有了更加深刻的认识，接触到了很多新的板块例如：font、time、event、mixer、transfrom等。更为难得可贵的是在思想认知上的进步。这使我对面向对象的软件开发方法逐步开始有了认识、还有单件模式、状态机的了解与学习

二、暑假期间，从7月底开始着手准备本项目的开发，到开学之前，除僵尸的生成外，大体上完成了游戏场景中主要的功能开发，代码行数达到了1200行。

（以下是一些本人认为比较重要的实训收获内容）

1.游戏框架——在暑期的开发过程中，入门最为困难。难点在于，实训的前两周因为有老师带领，根本无需考虑开发的项目需要什么、功能在哪里实现、以及服务器、场景、主程序之间“千丝万缕”的联系——换而言之，大型项目的编程不仅要有局部的技术，更需要立足于项目全局的统筹。但随着每天的学习，以及功能添加的过程，对整体框架的认识也开始逐渐清晰：

（以下是一些对所学游戏框架的个人理解）

* 1. 主循环位于头部，系切换场景之场所
  2. 下一层为游戏场景，乃设计游戏需要之场所
  3. 接下来为服务器，一切游戏所需参数的管理场所
  4. 最后、最基本的单位为游戏中的元素，充斥一切场所的构成之物

因此在后续的开发、改错效率上有了显著提高

2.面向对象——在开发过程中切实的感受到了这种发法在逻辑实现上体现出的便捷，举个例子：

因为很多植物都有发射子弹的这一设定，所以我需要创建一个子弹类，在子弹类里我们把子弹共有的属性、功能抽象出来写到这个类里面

属性：子弹图片、飞行速度、攻击伤害

方法：hit\_zombie()

在子弹类中对于子弹本身我只仅考虑子弹rect位置的更新，而不需要考虑其他；当需要与其他元素（僵尸）关联时，也很简单，只需要创建相应方法（hit\_zombie），思考需要传入的参数就可以了（这里参数就应该是与子弹发生碰撞的僵尸精灵了），然后我会在植物父类里的fire方法中创建子弹实例，在植物的attack方法中调用该子弹类的hit\_zombie方法，并传入与其碰撞的僵尸精灵作为参数，最终服务器会每帧刷新子弹的rect，调用hit\_zombie方法

（以下是对面向对象这一设计方法的个人理解）

面向对象的这一设计方法在开发过程中，能使程序的设计逻辑更接近人的思考逻辑，类与类之间的界限更加清晰、关联更加自然。

3.状态机——整个项目，总共开发了6个场景（包括加载场景、欢迎场景、主菜单场景、游戏场景、获胜场景以及失败场景）在开发的过程中遇到最大的困难就是如何设计不同场景之间的切换，以游戏场景切换到获胜场景为例，获胜场景在原版游戏中是游戏场景停止刷新，然后在此之上刷新获胜场景。我最后的复现思路就是当游戏达到获胜条件时，就切换到获胜场景，然后在每次调用获胜场景的draw方法时，先调用一下sever的draw方法到canvens上，因为此时sever的update是不会调用的（故sever不会继续更新），所以我们可以确保draw上的就是游戏场景的最后一帧画面，然后我们再draw获胜场景的画面。

（以下是对状态机这一设计模式的个人理解）

这种设计模式最大的好处就是，在游戏的设计上不需要大量复杂的游戏状态判断，而是场景与场景之间互相独立，互不干扰，当游戏处在哪种状态时就切换到对应的场景，而其他场景就好像原地不动一样，在原地等待调用，其实也和面向对象一样------在开发的过程中逻辑将会十分清晰。

三、实训的最后两周，拜军训所赐，工程量巨大，至于时间捉襟见肘。不得不昼夜鏖战。然可喜的是，有了老师的指导帮助，得以对暑假期间遗留的难点，疑点进行重新学习，也是在最后的这两周，项目才真正在意义上实现了从“能玩”到“好玩”的跃进。

四、总结，在为期4周的小学期和暑期的学习开发中，我的编程水平有了明显提高、对编程语言的掌握能力、知识面的扩展、开发设计思想上都有了明显进步等。项目的完成，其意义远不止是总计不到2500行的Python代码。它为我大一学习生活的结束画上了圆满句号，同时也成为了送给自己大二的开学礼物，它将一直激励着我，提醒我在计算机科学的漫长攀岩中获得的热情与快乐！这个项目和它的影响将会很快结束，但从其中获得的思维方式和知识将成为我漫长求学旅途中的永远伴侣。

While(True) enthusiasm++;

（一个在01数字世界畅游的孩子）

|  |  |  |  |
| --- | --- | --- | --- |
| 指导教师  评阅意见 | 不少于两行 | | |
| 成 绩 | 报告成绩 | | |
| 指导教师签字 |  | 评 阅 日 期 | 年 月 日 |