1813030 王少乾

外星人入侵项目

一.项目前的准备

1.安装pygame

二.开始游戏项目

1.创建pygame窗口以及相应用户输入

（1）创建alien\_invasion.py（import sys和import pygame）

（2）方法run\_game()

初始化游戏并创建一个屏幕对象（是该项目的主程序入口）

while True :开始游戏主循环 event :监视键盘和鼠标事件

pygame.display.flip() :让绘制的屏幕可见

2.创建设置类Settings（负责所有设置）

（1）屏幕设置 （2）飞船设置（3）子弹设置

（4）外星人设置 （5）速度设置（6）等级设置

（7）加分设置

3.添加飞船图像

（1）创建Ship类（负责管理飞船大部分行为）ship.py

（2）方法\_\_init\_\_() :

初始化飞船并设置其初始位置（self.screen）

加载飞船图像并获取其外接矩形（self.image、self.rect、self.screen）

将每艘新飞船放在屏幕底部的中央（self.rect.centerx、self.rect.bottom）

方法blitme() ：在指定位置绘制飞船（self.screen.blit(self.image,self.rect)）

4.重构：模块game\_function

（1）创建game\_function.py（用来存储项目的大部分函数）

（2）函数check\_evemts() ：响应按键和鼠标事件

（3）函数update\_screen() ：更新屏幕上的图像并切换到新屏幕

5.驾驶飞船

每当用户按键时，都在pygam中注册一个KEYDOWN事件，事件都是通过方法pygame.event.get()获取的。

（1）响应按键（如果按下右箭头键则增大飞船的 rect.centerx 值）

（2）允许飞船不断移动（设置标志self.moving\_right，按下为true，松开为false）

（3）左右移动 ：类似飞船右移动

（4）调整飞船速度

settings.py中添加飞船的速度属性ship\_speed\_factor

\_\_init\_\_()中初始化一个ai\_settings的属性在飞船的属性center中存储小数值，更新飞船的center值，而不是rect。根据self.center更新rect对象。

（5）限制飞船的活动范围

ship.py：对移动标志进行if判断时添加self.rect.right和self.rect.left，即要保证外接矩形的右边缘的x坐标小于屏幕右边缘，左边缘的x坐标要保证大于0。

（6）重构check\_events()

将check\_events()的代码放在两个函数中，一个处理KEYDOWN事件，一个处理KEYUP事件。

6.射击

（1）添加子弹设置

settings.py ：\_\_init\_\_()方法储存子弹设置，包括bullet\_width,bullet\_height,bullet\_color,bullet\_speed\_factor

（2）创建Bullet类

它是一个对飞船发射的子弹进行管理的类，继承了我们从模块pygame.sprite中导入的sprite类，通过使用精灵将游戏中相关的元素编组，进而同时操作编组中的所有元素。

bullet.py：\_\_init\_\_()方法：传递ai\_settings，screen，ship实例，调用super继承Sprite创建子弹属性rect将子弹的centerx设置为飞船的centerx，表示子弹的rect的top属性设置为飞船的rect的top属性，让子弹看起来像是从飞船中射出的 ，将子弹的y坐标存储为小数值，以便能够微调子弹的速度 ，将子弹的颜色和速度设置粉笔存储在self.color和self.speed\_factor中。

方法update():向上移动子弹更新表示子弹位置的小数值更新表示子弹的rect的位置。

方法draw\_bullet()：在屏幕上绘制子弹pygame.draw.rect()。

（3）将子弹存储到编组

alien\_invasion.py：创建一个编组，用于存储所有有效的子弹，以便能够管理发射出去的所有子弹 ，创建一个Group实例，命名为bullets，将其传递给check\_events()和update\_screen()，并在check\_events()后添加bullet.update()。

（4）开火

game\_functions.py:修改check\_keydown\_events()：以便在玩家按空格键时发射一颗子弹:创建一颗子弹，bullets.add(new\_bullet)将其加入到编组bullets中。

（5）删除已消失的子弹 ：判断是否在界外（bottom属性是否为0）

（6）限制子弹数量 ：检查未消失的子弹数len(bullets)

（7）创建函数update\_bullets() ：编写并检查子弹管理代码。

（8）创建函数fire\_bullet() ：发射子弹的代码移到改函数中（代码优化）

二.外星人

1.创建第一个外星人

（1）创建Alien类

alien.py ：创建外星人类

方法\_\_init\_\_() ：初始化外星人并设置其起始位置 ，加载外星人图像，并设置rect属性 ，每个外星人最初都在屏幕左上角附近（rect.x，rect.y） ，存储外星人的准确位置（self.x）

方法blitme() ：在指定位置绘制外星人（screen.blit()）

（2）创建Alien实例 ：在主程序里创建一个Alien实例

（3）让外星人出现在屏幕上：调用方法blitme()

2.创建一群外星人

（1）确定一行可容纳多少外星人

（2）创建多行外星人

（3）创建外星人群

（4）方法creat\_fleet() ：创建外星人并将其放在当前行

（5）添加行：方法get\_number\_rows() ：计算屏幕可容纳多少行外星人。

3.让外星人群移动

（1）向右移动外星人

（2）创建表示外星人移动方向的标志：添加fleet\_drop\_speed属性（外星人下移速度）。

（3）方法check\_edges() ：检查外星人是否撞到了屏幕左右边缘。

（4）方法check\_fleet\_direction() ：向下移动外星人群并改变移动方向

（5）方法update\_aliens() ：更新外星人位置

4.射杀外星人

（1）方法update\_bullets()：检查子弹与外星人的碰撞

（2）设置子弹的速度：调整bullet\_speed\_factor 的值

（3）重构update\_bullets() ：将处理子弹和外星人碰撞的代码移到一个函数中：check\_bullet\_alien\_collisions() 在update\_bullets()中调用

5.结束游戏

（1）pygame.sprite.spritecollideany(sprite，sprite\_group)：检测外星人和飞船碰撞

（2）响应外星人和飞船碰撞：创建用于跟踪游戏统计信息的新类 —— GameStats（创建game\_stats.py）初始化统计信息。

暂停：从模块time中导入sleep()函数

方法ship\_hit() ：响应被外星人撞到的飞船，暂停，清空外星人、子弹列表并创建一群新的外星人和新飞船。

6.确定应运行游戏的哪些部分

alien\_invasion.py ：确定哪些部分仅在游戏状态时运行，哪些在任何状态下运行。

7.计分

添加一个Play按钮，用于根据需要启动游戏以及在游戏结束后重启游戏。

我们还将修改这个游戏，使其在玩家的等级提高时加快节奏，并实现一个记分系统。

（1）添加play按钮

game\_stats.py：设置game\_active为False让游戏一开始处于非活动状态

（2）创建button类：button.py : 导入pygame.font模块

init()方法：初始化按钮的属性 ，设置按钮的尺寸和其他属性 ，创建按钮的rect对象，并使其居中，按钮的标签纸创建一次（调用方法prep\_msg(msg)）

prep\_msg()方法 ：接受实参self和要渲染为图像的文本msg ，将msg渲染为图像，并使其在按钮上居中：font.render()将存储在msg中的文本转换为图像，然后将图像存储在msg\_image中，还有一个布尔实参，和两个文本颜色和背景色实参。

方法draw\_button.py：绘制一个用颜色填充的按钮（screen.fill()），绘制文本（screen.blit()）。

（3）在屏幕上绘制按钮

alien\_invasion.py：导入button，创建一个名为play\_button的实例将其传递给update\_screen()方法。

game\_fuctions.py：修改update\_screen()：如果游戏处于非活动状态，就绘制play按钮，此时运行，就会在屏幕中央看到一个play按钮。

（4）开始游戏

game\_fuctions.py：修改check\_events()：添加形参stats和play\_button ；添加elif模块，判断事件类型是否是MOUSEBUTTONDOWN，若是，使用pygame.mouse.get\_pos()，返回一个包含玩家单击时鼠标的x和y坐标的元组，并将这些值传递给函数check\_play\_button()

check\_play\_button()：

在玩家单击play按钮时开始新游戏：

使用collidepoint()检查鼠标单击位置是否在play按钮的rect内。若是，就将game\_active设置为True，让游戏就此开始。

alien\_invasion.py：调用check\_events()

（5）重置游戏

（6）将play按钮切换到非活动状态

（7）隐藏光标：在check\_play\_button()的if模块中：添加隐藏光标的语句：pygame.mouse.set\_visible(False)

ship\_hit()： else模块中，在游戏进入非活动状态后，立即让光标可见。