编写程序《外星人入侵》

共有7个py文件来存储游戏所需编码内容：

1、alien\_invasion.py为主运行文件

2、settings.py游戏需要设置的变量参数

3、game\_functions.py所有游戏运行所需要的效果代码

4、ship.py设置一个飞船类

5、bullet.py设置绘制一个子弹类

6、alien.py设置一个外星人类

7、geme\_stats.py设置一个统计信息类

1、首先建立一个alien\_invasion.py文件，为主运行文件，添加模块：

import pygame

from pygame.locals import \*

import sys

2、定义一个游戏运行函数def run\_game()，最后调用run\_game()

将游戏运行所需内容写在函数内

1）初始化游戏：pygame.init()

2）绘制窗口大小：screen=pygame.display.set\_mode（）

3）设置游戏窗口标题名称：pygame.display.set\_caption('Alien Invasion')

4）将游戏需要设置的变量参数等单独放到一个文件settings.py设置类来去调用

class Settings():

'''存储《外星人入侵》的所有设置类'''

def \_\_init\_\_(self): '''初始化屏幕设置'''

#屏幕设置

self.screen\_width=800

self.screen\_height=600

self.bg\_color=(230,230,230)

#子弹设置

self.bullet\_speed\_factor=1

self.bullet\_width=3

self.bullet\_height=15

self.bullet\_color=(60,60,60)

#外星人设置

self.alien\_speed\_factor=0.3

self.fleet\_drop\_speed=10

self.fleet\_direction=1

self.ship\_limit=3

5）游戏运行：while True:

编写退出事件 event

编写所需投放到屏幕上的背景颜色，图片等blit

编写投放到屏幕运行pygame.display.flip()

3、将游戏主运行的所有内容单独放在一个game\_functions.py文件

1）'''将退出事件封装函数，在主程序中调用'''

def check\_event(ai\_settings,screen,ship,bullets):

for event in pygame.event.get():

if event.type==pygame.QUIT:

pygame.quit()

sys.exit()

if event.type==pygame.KEYDOWN:

check\_keydown\_events(ai\_settings,screen,event,ship,bullets)

if event.type==pygame.KEYUP:

check\_keyup\_events(event,ship)

2）'''将填充与投放到屏幕封装函数，在主程序中调用'''

def update\_screen(ai\_settings,screen,ship,bullets,aliens):

screen.fill(ai\_settings.bg\_color)

aliens.draw(screen) #alien.blitme()单个显示外星人

ship.blitme()

ship.update()

for bullet in bullets.sprites():

bullet.draw\_bullet()

pygame.display.flip()

4、ship.py添加飞船图片，设置参数，移动

class Ship():

'''初始化飞船并设置其初始位置'''

def \_\_init\_\_(self,screen):

self.screen=screen

self.move\_right=False

self.move\_left=False

self.image=pygame.image.load('/python/ship.png') #加载飞船图像并获取外接矩形

self.rect=self.image.get\_rect()

self.screen\_rect=self.screen.get\_rect()

self.rect.centerx=self.screen\_rect.centerx #将飞船放在屏幕底部中央

self.rect.bottom=self.screen\_rect.bottom

def blitme(self):

'''在指定位置绘制飞船'''

self.screen.blit(self.image,self.rect)

def update(self):

'''定义飞船左右移动'''

if self.move\_right and self.rect.centerx<self.screen\_rect.right-self.rect.width/2:

self.rect.centerx+=1

elif self.move\_left and self.rect.centerx>self.rect.width/2:

self.rect.centerx-=1

def center\_ship(self):

'''新一轮重新将飞船放在屏幕底部中央'''

self.rect.centerx=self.screen\_rect.centerx

self.rect.bottom=self.screen\_rect.bottom

5、设置飞船按键左右移动在game\_functions.py内

1） '''当按下键盘时触发的事件'''

def check\_keydown\_events(ai\_settings,screen,event,ship,bullets):

if event.key==pygame.K\_RIGHT:

ship.move\_right=True

if event.key==pygame.K\_LEFT:

ship.move\_left=True

if event.key==pygame.K\_SPACE: #按下空格键发射子弹

fire\_bullet(ai\_settings,screen,ship,bullets)

2）'''当按键弹起时'''

def check\_keyup\_events(event,ship):

if event.key==pygame.K\_RIGHT:

ship.move\_right=False

if event.key==pygame.K\_LEFT:

ship.move\_left=False

6、建立bullet.py存储子弹代码

from pygame.sprite import Sprite #引入精灵包，可以管理多个对象，这样就不用建列表循环

class Bullet(Sprite):

'''一个对飞船发射的子弹进行管理的对象'''

def \_\_init\_\_(self,ai\_settings,screen,ship):

'''在飞船所处的位置创建一个子弹对象'''

super(Bullet,self).\_\_init\_\_()

self.screen=screen

#在（0，0）处创建一个表示子弹矩形，在设置正确的位置

self.rect=pygame.Rect(0,0,ai\_settings.bullet\_width,ai\_settings.bullet\_height)

self.rect.centerx=ship.rect.centerx

self.rect.top=ship.rect.top

#存储用小数表示子弹的位置

self.y=float(self.rect.y)

self.color=ai\_settings.bullet\_color

self.speed\_factor=ai\_settings.bullet\_speed\_factor

def update(self):

'''向上移动的子弹'''

#更新表示子弹位置的小数

self.y-=self.speed\_factor

#更新表示子弹的rect的位置

self.rect.y=self.y

def draw\_bullet(self):

'''在屏幕上绘制子弹'''

pygame.draw.rect(self.screen,self.color,self.rect)

7、在game\_function.py设置子弹效果

def fire\_bullet(ai\_settings,screen,ship,bullets):

'''发射多个子弹'''

new\_bullet=Bullet(ai\_settings,screen,ship)

bullets.add(new\_bullet)

def update\_bullets(bullets,aliens):

'''将已出屏幕的子弹删除'''

bullets.update() #引入子弹

for bullet in bullets.copy(): #删除拷贝的，避免出错

if bullet.rect.bottom<0:

bullets.remove(bullet)

collsions=pygame.sprite.groupcollide(bullets,aliens,True,True)

#sprite精灵中固有的碰撞方法，True,True当碰撞时删除

8、建立allien.py编写外星人

rom pygame.sprite import Sprite

class Alien(Sprite):

'''表示单个外星人的类'''

def \_\_init\_\_(self,ai\_settings,screen):

super().\_\_init\_\_()

self.screen=screen

self.ai\_settings=ai\_settings

#加载外星人图片，并设置其rect属性

self.image=pygame.image.load('/python/alien.bmp')

self.rect=self.image.get\_rect()

self.rect.x=self.rect.width

self.rect.y=self.rect.height

def blitme(self):

'''在屏幕上绘制外星人'''

self.screen.blit(self.image,self.rect)

def check\_edges(self):

'''检查外星人是否到屏幕边缘'''

screen\_rect=self.screen.get\_rect()

if self.rect.right>=screen\_rect.right:

return True

elif self.rect.left<=0:

return True

def update(self):

'''向右移动'''

self.x+=self.ai\_settings.alien\_speed\_factor\*self.ai\_settings.fleet\_direction

self.rect.x=self.x

9、在game\_function.py设置外星人效果

def create\_alien(ai\_settings,screen,aliens,row\_number,alien\_number):

'''创建一个外星人并放入当前行'''

alien=Alien(ai\_settings,screen)

alien\_width=alien.rect.width

alien.x=alien\_width+2\*alien\_width\*alien\_number

alien.rect.x=alien.x

alien.rect.y=alien.rect.height+2\*alien.rect.height\*row\_number

aliens.add(alien)

def create\_fleet(ai\_settings,screen,aliens,ship):

'''建立一个外星人群'''

alien=Alien(ai\_settings,screen)

number\_aliens\_x=get\_number\_aliens\_x(ai\_settings,alien.rect.width)

number\_rows=get\_num\_rows(ai\_settings,ship.rect.height,alien.rect.height)

for row\_number in range(number\_rows):

for alien\_number in range(number\_aliens\_x):

create\_alien(ai\_settings,screen,aliens,row\_number,alien\_number)

def get\_number\_aliens\_x(ai\_settings,width):

'''计算每行可容纳多少外星人'''

available\_space\_x=ai\_settings.screen\_width-2\*width

number\_aliens\_x=int(available\_space\_x/(2\*width))

return number\_aliens\_x

def get\_num\_rows(ai\_settings,ship\_height,alien\_height):

'''计算屏幕可容纳多少行外星人'''

available\_space\_y=ai\_settings.screen\_height-3\*alien\_height-ship\_height

number\_rows=int(available\_space\_y/(2\*alien\_height))

return number\_rows

def change\_fleet\_direction(ai\_settings,aliens):

'''改变外星人的移动方向，垂直移动'''

for alien in aliens.sprites():

alien.rect.y+=ai\_settings.fleet\_drop\_speed

ai\_settings.fleet\_direction\*=-1

def check\_fleet\_edges(ai\_settings,aliens):

'''检测边缘，到屏幕边缘时调用change\_fleet\_direction改变移动方向'''

for alien in aliens.sprites():

if alien.check\_edges():

change\_fleet\_direction(ai\_settings,aliens)

break

def check\_aliens\_bottom(ai\_settings,stats,screen,ship,aliens,bullets):

'''检测每次外星人位于屏幕底部时'''

screen\_rect=screen.get\_rect()

for alien in aliens.sprites():

if alien.rect.bottom>screen\_rect.bottom:

ship\_hit(ai\_settings,stats,screen,ship,aliens,bullets)

break

def update\_aliens(ai\_settings,stats,screen,ship,aliens,bullets):

'''将外星人绘制到屏幕'''

check\_fleet\_edges(ai\_settings,aliens)

aliens.update()

if pygame.sprite.spritecollideany(ship,aliens):

ship\_hit(ai\_settings,stats,screen,ship,aliens,bullets)

check\_aliens\_bottom(ai\_settings,stats,screen,ship,aliens,bullets)

10、在game\_function.py中编写函数来检测飞船与外星人碰撞，删除外星人

def ship\_hit(ai\_settings,stats,screen,ship,aliens,bullets):

stats.ships\_life-=1

if stats.ships\_life>0:

#清空外星人列表和子弹列表

aliens.empty()

bullets.empty()

#创建一群新的外星人并将飞船放在屏幕底部中央

create\_fleet(ai\_settings,screen,aliens,ship)

ship.center\_ship()

sleep(0.5)

else:

stats.game\_active=False

11、建立geme\_stats.py来设置分值及游戏开局效果

class GameStats():

'''跟踪游戏的统计信息'''

def \_\_init\_\_(self,ai\_settings):

'''初始化统计信息'''

self.ai\_settings=ai\_settings

self.reset\_stats()

self.game\_active=True #游戏刚启动时处于活动状态

def reset\_stats(self):

'''初始化随游戏进行可能变化的统计信息'''

self.ships\_life=self.ai\_settings.ship\_limit

12、主运行程序alien\_invasion.py调用文件包与函数

1）文件包调用

from settings import Settings

from ship import Ship

import game\_functions as gf #引入函数包并附简化名字gf

from pygame.sprite import Group #Group用来存储精灵sprite

from geme\_stats import GameStats

2）函数调用

def run\_game():

ai\_settings=Settings() #创建一个设置类

'''初始化游戏并创建屏幕对象'''

pygame.init()

screen=pygame.display.set\_mode((ai\_settings.screen\_width,ai\_settings.screen\_height)) #绘制窗口大小，调用settings文件包

pygame.display.set\_caption('Alien Invasion')

ship=Ship(screen)#创建一个飞船对象

aliens=Group()#创建一个外星人群

gf.create\_fleet(ai\_settings,screen,aliens,ship)

stats=GameStats(ai\_settings)

bullets=Group()#创建一个用于存储子弹的编组

while True: #主程序运行

gf.check\_event(ai\_settings,screen,ship,bullets) #调用函数用来实现退出事件

if stats.game\_active:

gf.update\_bullets(bullets,aliens) #调用删除出屏幕子弹函数

gf.update\_aliens(ai\_settings,stats,screen,ship,aliens,bullets)

gf.update\_screen(ai\_settings,screen,ship,bullets,aliens)#调用绘制屏幕函数

run\_game()