python 作业 level5 实验报告

舒文炫

2021 年 7 月 17 日

目录

1 实验介绍 **2**

1.1 实验内容 2

1.2 实验环境 2

2 实验实现 **3**

3 实验结果 **8**

Chapter 1

实验介绍

1.1 实验内容

这一部分我使用了 pygame 库实现了一个类似的打飞机游戏的一部分，准确来说是移动，更多功能放在大作业中实现。

1.2 实验环境

python 版本 3.7

Chapter 2

实验实现

游戏名叫刻晴大战派蒙，这两个是原神里面的角色，贴图是在网络上寻找的。这里代码量很大，我一步一步分析。

```python
import pygame
from settings import Settings
from kq import Kq
import game_functions as gf
from pygame.sprite import Group
from pm import PM
def run_game():
    pygame.init()
    game_settings=Settings()

    screen=pygame.display.set_mode((game_settings.width,game_settings.height)) ##the screen is (800width 800height)
    pygame.display.set_caption("刻晴大战派蒙")
    keqing=Kq(game_settings,screen)
    bullets=Group()
    pimeng=Group()
    gf.create_fleet(game_settings,screen,pimeng)


    while True:

        gf.check_events(game_settings,screen,keqing,bullets)
        keqing.update()
        gf.update_bullets(bullets)
        gf.update_pm(game_settings,pimeng)
        gf.update_screen(game_settings,screen,keqing,bullets,pimeng)
run_game()
```

这里是游戏的启动模块，要打开这个游戏，输入运行这个程序即可。先初始化 pygame，创建一个窗口，keqing 即刻晴，是玩家操控的人物。bullets 是子弹，玩家攻击派蒙的主要方式。pimeng 即为游戏的敌人派蒙。后面通过一个 while 循环，不懂进行，进行游戏界面的更新，比如角色移动，射出子弹。

```python
class Settings():
    def __init__(self):
        self.width=1200 ##screen width
        self.height=800 ##screen height
        self.bg_color=(200,150,120) ##background color RGB mode
        self.kq_speed_factor=1  ##the speed of the character
        self. (variable) bullet_width: Literal[2]
        self.bullet_width=2
        self.bullet_height=10
        self.bullet_color=(10,10,10)
        self.bullet_allowed=5
        self.pm_speed_factor=0.8
        self.fleet_drop_speed=10
        self.fleet_direction=1  ##1 means right -1 menas left

```

这里是游戏的初始设置模块，我在这里给出了一些参数 width 和 height 是游戏窗口的大小，bg_color 是游戏背景的颜色，这里是 RGB 形式表示，这里我没找到特别好的背景图片。还有子弹大小，速度，派蒙的速度等等。

```python
import pygame

class Kq():

    def __init__(self,game_settings,screen):
        self.screen=screen
        self.game_settings=game_settings
        self.image=pygame.image.load('images/kq.bmp') ##load the image
        self.rect=self.image.get_rect()      #the rectangular of the image
        self.screen_rect=screen.get_rect()   #the rectangular of the screen
        self.rect.centerx=self.screen_rect.centerx
        self.rect.bottom=self.screen_rect.bottom
        self.centerx=float(self.rect.centerx)
        self.top=float(self.rect.top)
        self.moving_right=False   ##to show the character is moving or not
        self.moving_left=False
        self.moving_up=False
        self.moving_down=False
    def update(self):
        if self.moving_right and self.rect.right < self.screen_rect.right:
            self.centerx+=self.game_settings.kq_speed_factor
        if self.moving_left and self.rect.left>0:
            self.centerx-=self.game_settings.kq_speed_factor
        if self.moving_up and self.rect.top>self.screen_rect.top:
            self.top-=self.game_settings.kq_speed_factor
        if self.moving_down and self.rect.bottom<self.screen_rect.bottom:
            self.top+=self.game_settings.kq_speed_factor
        self.rect.centerx=self.centerx
        self.rect.top=self.top
    def blitme(self):
        self.screen.blit(self.image,self.rect)
```

这里是对角色刻晴的设置模块，初始化刻晴在屏幕中的位置，update 方法用来更新她的位置，blitme 方法用来在游戏窗口绘制刻晴

```python
import pygame
from pygame.sprite import Sprite
class PM(Sprite):

    def __init__(self,game_settings,screen): ##paimeng
        Sprite.__init__(self)
        super(PM).__init__()
        self.screen=screen
        self.game_settings=game_settings

        self.image=pygame.image.load('images/pm.bmp') ##the position
        self.rect=self.image.get_rect()
        self.rect.x=self.rect.width
        self.rect.y=self.rect.height
        self.x=float(self.rect.x)

    def blitme(self):
        self.screen.blit(self.image,self.rect)

    def check_edge(self):
        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):
        self.x+=self.game_settings.pm_speed_factor*self.game_settings.fleet_direction
        self.rect.x=self.x
```

这里是对角色派蒙的设置模块，使用了 pygame 中的 sprites 模块，用来对一群派蒙进行管理，这里每次生成一行的派蒙，然后派蒙的移动是先向右，移动到头后向下移动一段，再向左移动这样循环，使用了 check_edge 来检测是否碰到了左右的边界, 碰到边界的话，返回一个真值，以便后续函数处理。

```python
import pygame
from pygame.sprite import Sprite

class Bullet(Sprite):
    def __init__(self,game_settings,screen,keqing):
        super(Bullet,self).__init__()
        self.screen=screen

        self.rect=pygame.Rect(0,0,game_settings.bullet_width,game_settings.bullet_height) ##create bullet rect and initiali
        self.rect.centerx=keqing.rect.centerx
        self.rect.top=keqing.rect.top
        self.y=float(self.rect.y)
        self.color=game_settings.bullet_color
        self.speed_factor=game_settings.bullet_speed_factor
    def update(self): ##use speed_factor show the pixel it goes each time
        self.y-=self.speed_factor
        self.rect.y=self.y

    def draw_bullet(self): ##draw the bullet in the screen
        pygame.draw.rect(self.screen,self.color,self.rect)

```

这里是对子弹这个对象的设置模块，我设置了子弹大小，颜色，初始位置是在刻晴的头部发出。它具有 update 方法，更新它的位置，已经 draw 方法，用于再 screen 上显示。

```python
import sys ##exit the game
import pygame
from bullet import Bullet
from pm import PM
def check_keydown(event,game_settings,screen,keqing,bullets):
    if event.key==pygame.K_RIGHT:
        keqing.moving_right=True
    if event.key==pygame.K_LEFT:
        keqing.moving_left=True
    if event.key==pygame.K_UP:
        keqing.moving_up=True
    if event.key==pygame.K_DOWN:
        keqing.moving_down=True
    elif event.key==pygame.K_SPACE:
        fire(game_settings,screen,keqing,bullets)
    elif event.key==pygame.K_q:
            sys.exit()
def check_keyup(event,keqing):
    if event.key==pygame.K_RIGHT:
        keqing.moving_right=False
    if event.key==pygame.K_LEFT:
        keqing.moving_left=False
    if event.key==pygame.K_UP:
        keqing.moving_up=False
    if event.key==pygame.K_DOWN:
        keqing.moving_down=False

def fire(game_settings,screen,keqing,bullets): ##shoot the hun, you are allowed to shoot 5 at ine time
    if len(bullets)<game_settings.bullet_allowed:
            new_bullet=Bullet(game_settings,screen,keqing)
            bullets.add(new_bullet)
```

这里开始，是游戏具体操作的实现部分，通过 check_keydown 方法，检测键盘的输入，从而对应控制角色向对应方向移动，但是仅仅移动不够，这里我实现了长按对应方向键可以连续向一个方向移动，速度为 game_setting 中所设 kq_speed_factor，这里就需要通过 check_keyup 方法，这个的用处在于，保存当前状态知道松开对应键，即通过 keydown 开始，keyup 结束。fire 方法即射击，射出子弹，传入的 bullets 可以看成一个子弹数组，里面储存了射出的子弹。那么射出子弹就相当于往这个数组里面添加子弹元素。

```
def check_events(game_settings,screen,keqing,bullets):
    for event in pygame.event.get():##get the event such as mice and keybox input
            if event.type==pygame.QUIT:
                sys.exit()
            elif event.type==pygame.KEYDOWN:  ##to move continuously of you press the buttom
                check_keydown(event,game_settings,screen,keqing,bullets)

            elif event.type==pygame.KEYUP:
                check_keyup(event,keqing)

def get_num_pimeng_x(game_settings,pm_width): ##get the max mun one row can supply
    available_spacce_x=game_settings.width-2*pm_width
    num_pimeng_x=int(available_spacce_x/2/pm_width)
    return num_pimeng_x

def create_pm(game_settings,screen,pimeng,pm_num): ##create one pimeng
    pimengs=PM(game_settings,screen)
    pm_width=pimengs.rect.width
    pimengs.x=pm_width+2*pm_width*pm_num
    pimengs.rect.x=pimengs.x
    pimeng.add(pimengs)

def change_fleet_direction(game_settions,pimeng): ##change the direction if pm comes to the edge
    for pimengs in pimeng.sprites():
        pimengs.rect.y+=game_settions.fleet_drop_speed
    game_settions.fleet_direction*=-1
```

check_event 方法在 start 中使用，里面包含了 check_keydown 和 check_keyup，即对这两个方法再进行封装，这样可以使 start 函数更简洁。后面开始创造派蒙这个角色，get_num_pimeng_x 方法可以返回每行最多容纳多少个派蒙，然后在 create_pm 方法中，用来按顺序创造单个派蒙，其中的 pm_num 参数即表示这是该行第几个派蒙。change_fleet_direction 方法则是用来使得这一行派蒙移动。

```
def check_fleet_edges(game_settings,pimeng): ##check whether pm comes to the edge
    for pimengs in pimeng.sprites():
        if pimengs.check_edge():
            change_fleet_direction(game_settings,pimeng)
            break



def create_fleet(game_settings,screen,pimeng): ##creat a group of pimeng
    pimengs=PM(game_settings,screen)
    number_pimeng_x=get_num_pimeng_x(game_settings,pimengs.rect.width)
    for pm_num in range(number_pimeng_x):
        create_pm(game_settings,screen,pimeng,pm_num)




def update_pm(game_settings,pimeng):  ##update pimeng
    check_fleet_edges(game_settings,pimeng)
    pimeng.update()
```

check_fleet_edges 方法则是去判断这一个派蒙组里面是否有任何一只派蒙碰到边界 create_fleet 方法就是创造每行最大数量的派蒙，里面使用循环。update_pm 方法则是调用了 check_fleet_edges 方法，查看是否到边界，是否需要改变方向。我设置向右方向为 1，向左方向为-1, 这在 update 方法里面反应为只需要乘以这个方向，就能是派蒙朝着对应方向移动。

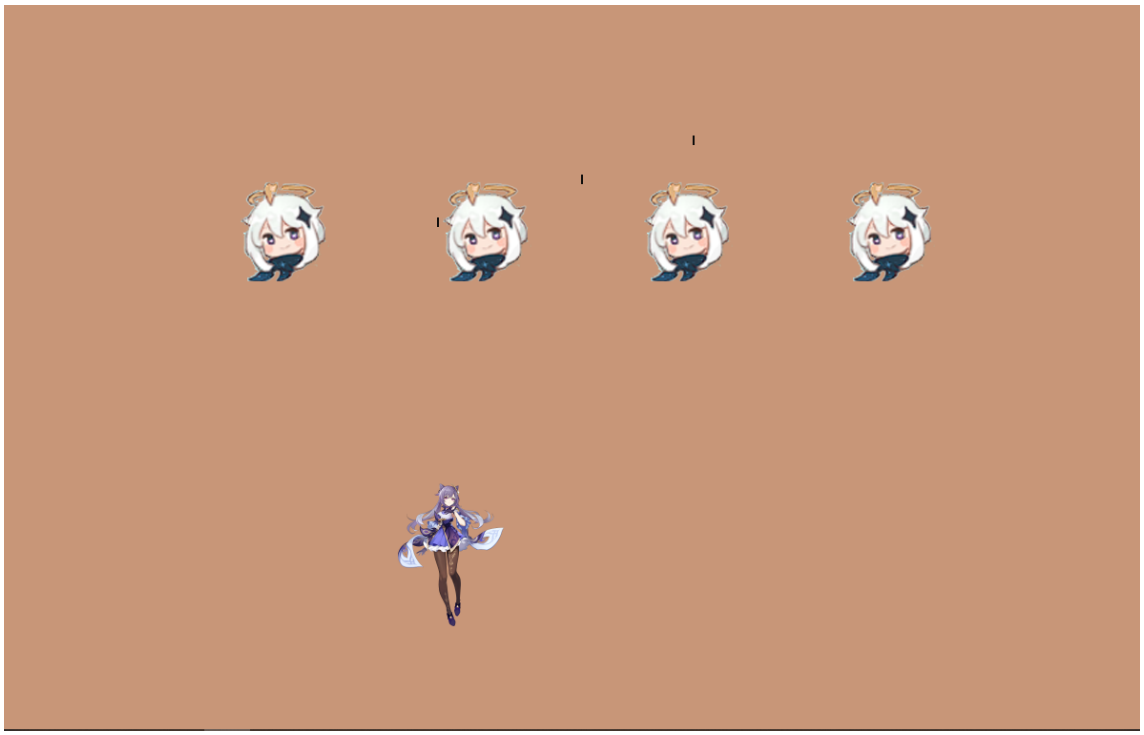
```
def update_screen(game_settings,screen,keqing,bullets,pimeng):
    screen.fill(game_settings.bg_color)
    for bullet in bullets.sprites():
        bullet.draw_bullet()

    keqing.blitme()
    pimeng.draw(screen)
    pygame.display.flip()

def update_bullets(bullets):
    bullets.update()
    for bullet in bullets.copy():
        if bullet.rect.bottom<=0:
            bullets.remove(bullet)
```

这里就是更新屏幕的显示，同时删除了多余的子弹，即子弹如果碰到边界即消失，从子弹这个组里面

删去，这样可以使得总的子弹数有个很好的控制。

Chapter 3

实验结果



这个就是我目前的输出，上面四个是派蒙，下面紫发的妹子是刻晴，黑色的是子弹，不过目前尚未实现子弹的攻击功能，只能和派蒙擦身而过，目前也只能做到显示刻晴，派蒙，子弹，同时可以控制刻晴移动这样。剩下的功能，我会尝试在大作业中完善，毕竟作业五本身并没有要求过高的复杂度。