# 程序设计报告

**一．项目介绍**

针对python课程设计，我做了一个小游戏——飞机大战，游戏代码包含到本学期所学的所有知识点。

程序运行后，进入到开始画面，鼠标单击开始游戏。敌机自上向下移动，随机出现，玩家机随鼠标移动并发射子弹，消灭敌机可以获得分数，随机出现小蜜蜂，消灭后可获得奖励。

**二．概要设计**

2.1资源需求

此游戏需要导入图片：背景图片，开始界面，玩家飞机，敌机，补给包，子弹，暂停界面，结束界面。

2.2游戏流程

显示标题界面

单击鼠标

游戏主界面

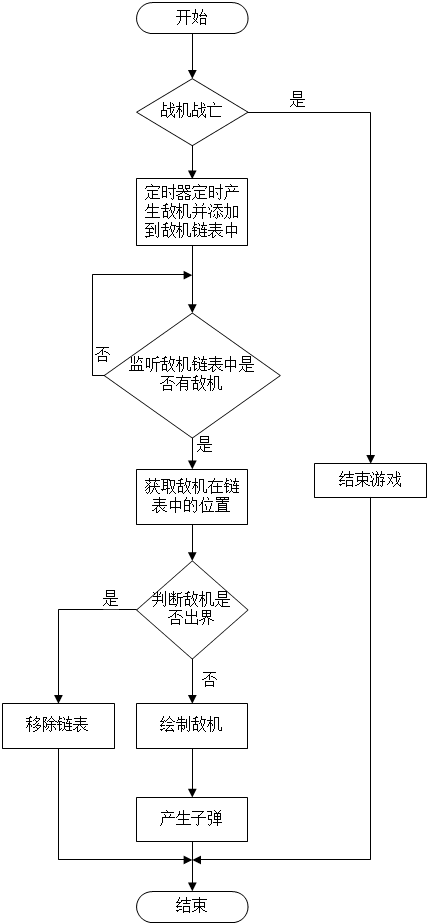
鼠标移出

暂停界面

单击鼠标

玩家死亡

游戏结束



**第一步，显示背景。**

我们先来显示出飞机大战游戏的背景图

# -\*- coding: utf-8 -\*-

import pygame

from pygame.locals import \*

from sys import exit

SCREEN\_WIDTH = 480

SCREEN\_HEIGHT = 800

# 初始化游戏

pygame.init()

screen = pygame.display.set\_mode((SCREEN\_WIDTH, SCREEN\_HEIGHT))

pygame.display.set\_caption('飞机大战')

# 载入背景图

background = pygame.image.load('resources/image/background.png')

while True:

# 绘制背景

screen.fill(0)

screen.blit(background, (0, 0))

# 更新屏幕

pygame.display.update()

# 处理游戏退出

for event in pygame.event.get():

if event.type == pygame.QUIT:

pygame.quit()

exit()

我们来分析这段代码：

1. 导入Pygame库及需要的库；
2. 初始化游戏，并根据设置好的大小生成游戏窗口；
3. 载入背景图片；
4. 进入游戏主循环。在主循环中，我们进行了以下工作：
   * 清空屏幕
   * 绘制背景
   * 更新屏幕
   * 处理游戏退出事件

运行后效果如下



**第二步，显示飞机。**

我们打开resources/image/，发现所有的飞机都在 shoot.png 一张图片中。我们在背景上显示的元素（包括飞机、子弹等）在Pygame中都是一个surface，这时可以利用pygame提供的subsurface方法，首先load一张大图，然后调用subsurface方法选取其中的一小部分生成一个新的surface。而这个小图在大图中的位置在image文件夹中shoot.pack文件里。

我们需要做的是在上述代码载入背景图后载入飞机图片，然后生成飞机的surbsurface并初始化飞机开始的位置：

# 载入飞机图片

plane\_img = pygame.image.load('resources/image/shoot.png')

# 选择飞机在大图片中的位置，并生成subsurface，然后初始化飞机开始的位置

player\_rect = pygame.Rect(0, 99, 102, 126)

player = plane\_img.subsurface(player\_rect)

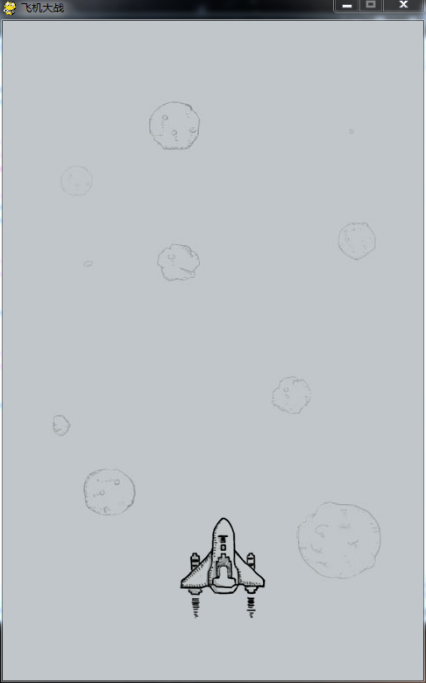
player\_pos = [200, 600]

然后在主循环中绘制背景之后插入绘制飞机的代码：

1 # 绘制飞机

2 screen.blit(player, player\_pos)

运行后就能看见玩家操作的战机了：



**第三步，让飞机动起来。**

让飞机动起来即是让游戏在主循环中响应键盘事件。上面的代码中我们知道 screen.blit(player, player\_pos) 这一句代码绘制出了飞机，这个player\_pos就是每次循环中绘制出的飞机的位置，响应键盘事件时，我们只需要改变这个位置就能移动飞机了。在主循环中加入响应键盘事件的代码：

# 监听键盘事件

key\_pressed = pygame.key.get\_pressed()

if key\_pressed[K\_UP]:

player\_pos[1] -= 3

if key\_pressed[K\_DOWN]:

player\_pos[1] += 3

if key\_pressed[K\_LEFT]:

player\_pos[0] -= 3

if key\_pressed[K\_RIGHT]:

player\_pos[0] += 3

每当一个方向键按下时，我们只需要改变一个飞机位置的坐标，就可以让飞机动起来~

这样我们知道了制作一个2D游戏的基本的原理：游戏进入主循环后，每一次循环在相应的位置绘制出图片，改变位置或者图片就可以出现动画效果；然后通过判断两张图片的矩形区域是否相交来检测碰撞。简单吧？

接下来的步骤还有显示子弹、随机生成敌机、检测碰撞、判断游戏结束、计分等。我想介绍了这些，大家应该能自己完成接下来的任务了吧？

当然，玩家、子弹、敌机都可以写成一个类，继承pygame的sprite类，实现一些动画效果，以及检测碰撞，这样使用起来更简单。给大家看一下我写的类代码

# 子弹类

class Bullet(pygame.sprite.Sprite):

def \_\_init\_\_(self, bullet\_img, init\_pos):

pygame.sprite.Sprite.\_\_init\_\_(self)

self.image = bullet\_img

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

self.rect.midbottom = init\_pos

self.speed = 10

def move(self):

self.rect.top -= self.speed

# 玩家类

class Player(pygame.sprite.Sprite):

def \_\_init\_\_(self, plane\_img, player\_rect, init\_pos):

pygame.sprite.Sprite.\_\_init\_\_(self)

self.image = [] # 用来存储玩家对象精灵图片的列表

for i in range(len(player\_rect)):

self.image.append(plane\_img.subsurface(player\_rect[i]).convert\_alpha())

self.rect = player\_rect[0] # 初始化图片所在的矩形

self.rect.topleft = init\_pos # 初始化矩形的左上角坐标

self.speed = 8 # 初始化玩家速度，这里是一个确定的值

self.bullets = pygame.sprite.Group() # 玩家飞机所发射的子弹的集合

self.img\_index = 0 # 玩家精灵图片索引

self.is\_hit = False # 玩家是否被击中

def shoot(self, bullet\_img):

bullet = Bullet(bullet\_img, self.rect.midtop)

self.bullets.add(bullet)

def moveUp(self):

if self.rect.top <= 0:

self.rect.top = 0

else:

self.rect.top -= self.speed

def moveDown(self):

if self.rect.top >= SCREEN\_HEIGHT - self.rect.height:

self.rect.top = SCREEN\_HEIGHT - self.rect.height

else:

self.rect.top += self.speed

def moveLeft(self):

if self.rect.left <= 0:

self.rect.left = 0

else:

self.rect.left -= self.speed

def moveRight(self):

if self.rect.left >= SCREEN\_WIDTH - self.rect.width:

self.rect.left = SCREEN\_WIDTH - self.rect.width

else:

self.rect.left += self.speed

# 敌人类

class Enemy(pygame.sprite.Sprite):

def \_\_init\_\_(self, enemy\_img, enemy\_down\_imgs, init\_pos):

pygame.sprite.Sprite.\_\_init\_\_(self)

self.image = enemy\_img

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

self.rect.topleft = init\_pos

self.down\_imgs = enemy\_down\_imgs

self.speed = 2

self.down\_index = 0

def move(self):

self.rect.top += self.speed

游戏截图:

