**西南交通大学**

**嵌入式系统实验日志**



**实验名称：**Pygame 小游戏制作实验

**实验序号：**8

**实验组号：**3组-6队

**使用语言：**Python

**姓名学号：**杨志成 2023211042

**实验时间：**2023年12月8日

**指导老师：**徐图

**一、实验目的：**

1、了解 Python 程序的特点与运行方式

2、掌握 Python 程序开发的基础知识

3、掌握运用 Python 第三方库开发的基础能力

4、了解小游戏开发的基本原理和构成

5、了解 Pygame 库的基本功能

6、掌握 Pygame 库的基本使用

7、掌握用 Pygame 库完成一个飞机大战小游戏的基本能力

**二、实验步骤：**

1、 导入 Pygame 模块并实现基本开发框架

2、 实现玩家类的创建并能控制其移动

3、 实现敌机类的创建及其自动移动

4、 实现敌机和玩家的碰撞检测

5、 实现子弹类的创建及子弹和敌机、子弹和玩家的碰撞检测

6、 通过 python 的列表元素实现敌机的随机创建和销毁

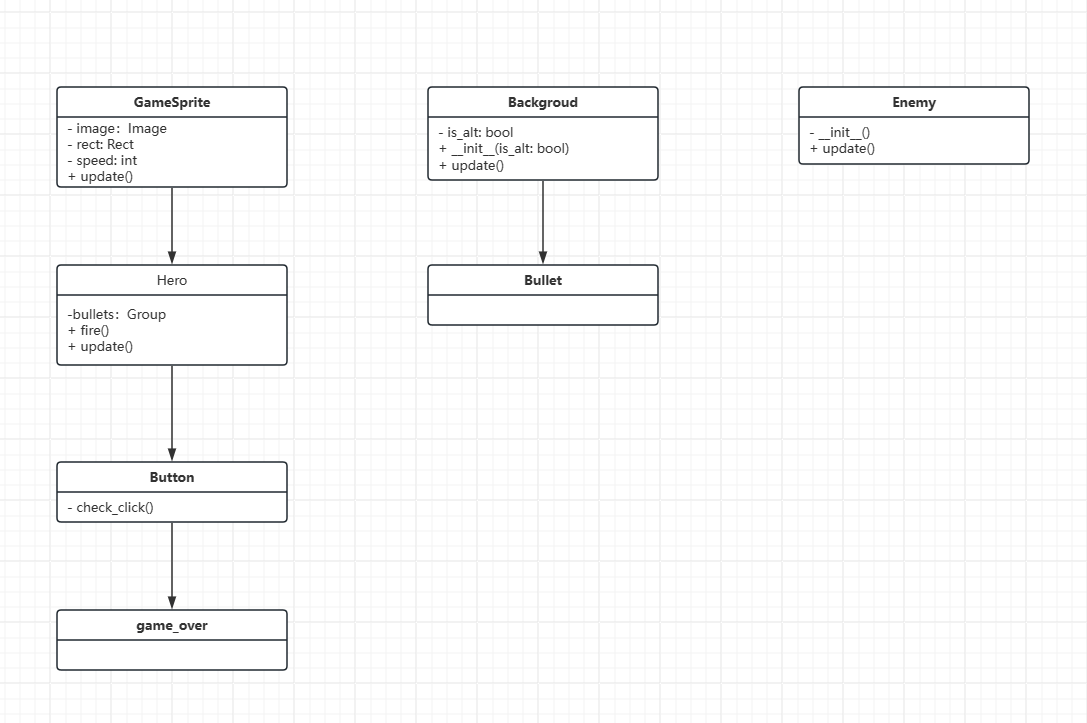
7、 通过 python 的列表元素实现子弹的创建和销毁

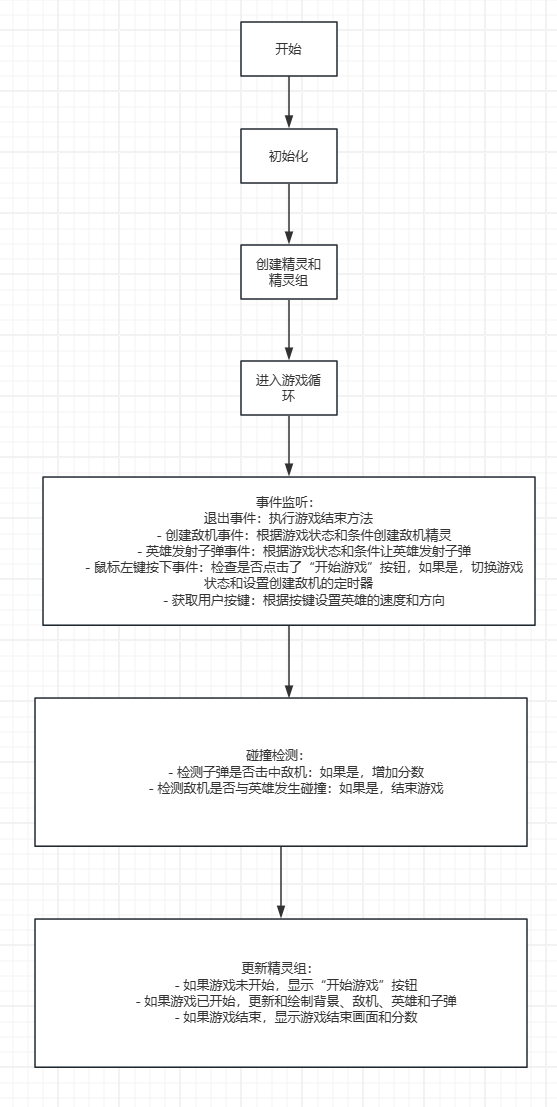
8、 实现起始界面、游戏界面、结束界面三种界面的创建和切换

9、 添加文字和背景图片，完成基本的界面美化 10、 以碰撞检测的方式实现按钮效果

**三、硬件连接图，软件流程图：**

软件流程图：





**四、关键代码分析：**

import pygame

from plane\_sprites import \*

Score = 0

class PlaneGame(object):

"""飞机大战主游戏"""

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

print("游戏初始化..")

pygame.init()

# 1.创建游戏窗口

self.screen = pygame.display.set\_mode(SCREEN\_RECT.size)

# 2.创建游戏的时钟

self.clock = pygame.time.Clock()

#3.调用私有方法，精灵和精灵组的创建

self.\_\_create\_sprites()

def \_\_create\_sprites(self):

"""创建精灵和精灵组"""

# 1.创建背景精灵和精灵组

bg1 = Background()

bg2 = Background(True)

#bg2.rect.y = -bg2.rect.height

self.back\_group = pygame.sprite.Group(bg1, bg2)

# 2.创建敌机精灵组

self.enemy\_group = pygame.sprite.Group()

# 3.创建英雄精灵组

self.hero = Hero()

self.hero\_group = pygame.sprite.Group(self.hero)

self.play\_button = Button()

self.button\_group = pygame.sprite.Group(self.play\_button)

self.over = game\_over()

self.game\_over\_group = pygame.sprite.Group(self.over)

def start\_game(self):

#print("游戏开始...")

while True:

# 1.设置刷新帧率

self.clock.tick(FRAME\_PER\_SEC)

# 2.事件监听

self.\_\_event\_handle()

# 3.碰撞检测

self.\_\_check\_collide()

# 4.更新精灵组

self.\_\_update\_sprites()

# 5.更新屏幕显示

pygame.display.update()

def \_\_event\_handle(self):

"""事件监听"""

for event in pygame.event.get():

# 监听游戏是否退出

if event.type == pygame.QUIT:

PlaneGame.\_\_game\_over()

# 监听敌机的出现

elif event.type == CREATE\_ENEMY\_EVENT:

if not Judge[0]:

if Judge1[0]:

print("敌机出场...")

if Judge1[0]:

# 创建敌机精灵

enemy = Enemy()

self.enemy\_group.add(Enemy())

# 让英雄发射子弹

elif event.type == Hero\_FIRE\_EVENT:

if Judge1[0]:

self.hero.fire()

if pygame.mouse.get\_pressed()[0]:#左键按下

if self.play\_button.check\_click(pygame.mouse.get\_pos()):

Judge[0] = False

# 4.设置定时器－－创建敌机出现 1s

pygame.time.set\_timer(CREATE\_ENEMY\_EVENT, 500)

print("游戏开始...")

# 获取用户按键

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

if keys\_pressed[pygame.K\_RIGHT]:

self.hero.speed = 2

Direction[0] = True

elif keys\_pressed[pygame.K\_LEFT]:

self.hero.speed = -2

Direction[0] = True

elif keys\_pressed[pygame.K\_UP]:

self.hero.speed = -2

Direction[0] = False

elif keys\_pressed[pygame.K\_DOWN]:

self.hero.speed = 2

Direction[0] = False

else:

self.hero.speed = 0

def \_\_check\_collide(self):

"""碰撞检测"""

global Score

# 1.设置子弹摧毁敌机

time = pygame.sprite.groupcollide(self.hero.bullets,self.enemy\_group, True, True)

if time:

Score += 1

print(Score)

# 2.敌机摧毁英雄

enemies = pygame.sprite.spritecollide(self.hero,self.enemy\_group, True)

# 2.1判断列表是否有内容

if len(enemies) > 0:

# 让英雄牺牲

Judge1[0] = False

self.hero.kill()

self.hero.bullets.empty()

self.enemy\_group.empty()

# 结束游戏

#PlaneGame.\_\_game\_over()

def \_\_update\_sprites(self):

"""更新精灵组"""

# 1.背景更新渲染显示

self.back\_group.update()

self.back\_group.draw(self.screen)

if Judge[0]:#鉴定是否开始 false开始，true没开始

self.button\_group.draw(self.screen)

ft = pygame.font.Font("Fonts/AaJiuZhongXiao-2.ttf", 40)

self.Font = ft.render("开始游戏", True, (191, 62, 255))

self.screen.blit(self.Font,(SCREEN\_RECT.centerx-86,SCREEN\_RECT.centery+50))

else:

# 2.敌机渲染更新显示

self.enemy\_group.update()

self.enemy\_group.draw(self.screen)

# 3.英雄渲染更新显示

self.hero\_group.update()

self.hero\_group.draw(self.screen)

# 4.子弹渲染更新显示

self.hero.bullets.update()

self.hero.bullets.draw(self.screen)

if not Judge1[0]:

self.game\_over\_group.draw(self.screen)

ft\_1 = pygame.font.Font("Fonts/AaJiuZhongXiao-2.ttf", 40)

self.Font = ft\_1.render("您的得分", True, (191, 62, 255))

self.screen.blit(self.Font, (SCREEN\_RECT.centerx - 86, SCREEN\_RECT.centery))

ft\_2 = pygame.font.Font("Fonts/AaJiuZhongXiao-2.ttf", 60)

self.Font = ft\_2.render(str(Score), True, (255 ,64, 64))

self.screen.blit(self.Font, (SCREEN\_RECT.centerx - 35, SCREEN\_RECT.centery + 60))

@staticmethod

def \_\_game\_over():

"""游戏结束"""

pygame.quit()

print("游戏结束")

exit()

if \_\_name\_\_ == '\_\_main\_\_':

# 创建游戏对象

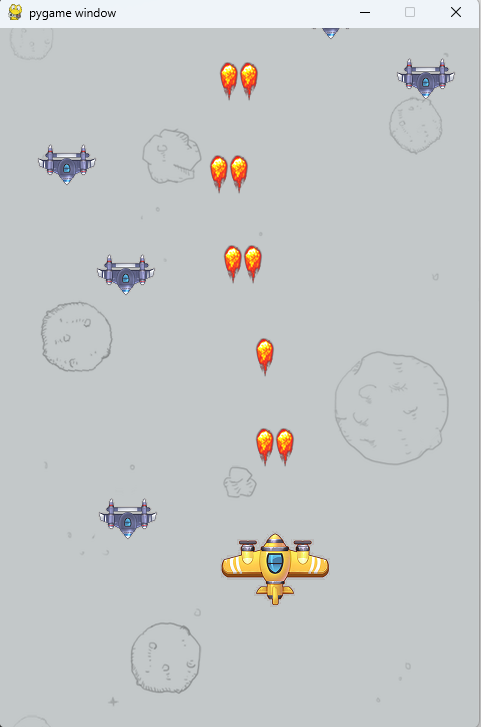
game = PlaneGame()

# 开始游戏

game.start\_game()

**五、实验结果：**

实现了游戏界面的生成和背景的布置，角色和敌机的生成、移动与显示，实现子弹的生成、移动与显示，实现键盘对角色的移动控制和子弹发射控制，实现攻击碰撞的判定，实现字体的设置和文字的显示，利用碰撞的原理实现按键的效果，实现初始和结束画面，实现计分功能

**六、实验中遇到的问题及处理方法：**

根据实验指导书上的步骤图片加载不出来，解决：检查图像文件路径是否正确。定时器设置错误：解决方法：确保定时器事件的类型和触发时间设置正确。检查定时器事件是否被正确绑定到事件处理方法。

Sprite组管理错误：

错误： 精灵组未正确管理，导致内存泄漏或未更新的精灵。

解决方法： 在适当的时候使用kill()方法删除不再需要的精灵，确保精灵组的update()方法得到调用。

碰撞检测问题：

错误： 碰撞检测不准确，导致游戏逻辑错误。

解决方法： 确保使用适当的碰撞检测方法，并检查精灵的rect属性是否正确设置。在调用碰撞检测前确保更新精灵的位置。

事件处理错误：

错误： 事件处理方法未正确响应用户输入或其他事件。

解决方法： 检查事件处理方法是否正确绑定，并确保事件类型被正确处理。注意处理键盘、鼠标等用户输入的方式。

游戏逻辑错误：

错误： 游戏逻辑不按预期工作，得分计算、游戏结束条件等有问题。

解决方法： 仔细检查游戏逻辑的实现，确保各个组件之间的协作正确。打印调试信息有助于追踪问题。

**七、实验中，自己的创意部分：**

按钮启动游戏： 使用一个按钮来启动游戏，而不是传统的键盘输入。这种方式使得用户可以通过点击按钮来启动游戏，增加了用户界面的互动性。

多线程定时器： 使用 pygame.time.set\_timer 来创建定时器事件，这种方式可以在主循环中异步处理事件，而不需要显式地在主循环中等待。这有助于提高游戏的响应性。

字体使用： 使用字体文件来渲染文字，而不是简单的默认字体。这样可以增强游戏的视觉效果。

**八、实验中的收获：**

1、了解了 Python 程序的特点与运行方式

2、掌握了 Python 程序开发的基础知识

3、掌握了运用 Python 第三方库开发的基础能力

4、了解了小游戏开发的基本原理和构成

5、了解了 Pygame 库的基本功能

6、掌握了 Pygame 库的基本使用

7、掌握了用 Pygame 库完成一个飞机大战小游戏的基本能力