Task 6

2019年8月16日 14:18

飞机大战游戏设计

- 游戏整体框架：

1. 设置游戏界面大小、标题、背景图片、飞机图片（正常&爆炸）、子弹图片；
2. 设置两个list，分布存储敌机和被击毁的飞机；
3. 初始化分数、射击频率、敌机移动频率，并设置游戏循环帧率；
4. 进入游戏的主循环部分；

- 游戏的主循环部分包括以下内容：
  a. 按一定频率发射子弹；
  b. 按一定频率生成敌机；
  c. 移动子弹；
  d. 移动敌机；
  e. 敌机与玩家飞机相撞处理方法；
  f. 敌机被子弹击中处理方法；
  g. 一系列绘制、显示的方法，包括：绘制背景、绘制玩家飞机、显示子弹、显示敌机、绘制得分、更新屏幕；
  h. 处理退出游戏。
- 一共需建立三个类：
  a. 子弹类；
  b. 玩家飞机类；
  c. 敌机类；

5. 在“gameover”后显示最终得分；
6. 处理游戏退出。

- 所涉及的函数和类：

1. 子弹类

- 子弹类里包含两个函数：
  a. 一个用来定义子弹的基本属性：子弹图片、位置、移动速度；
  b. 另一个用来计算子弹位置

```
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
```

2. 玩家飞机类：

- 主要有三个函数：
  a. 定义基本属性：设置飞机的图片、大小、位置、速度、是否被撞、并建立了存储飞机发射子弹的集合；
  b. 如何发射子弹：调用了子弹类，给子弹类传递了实参，包括子弹的图片和位置；
  c. 如何移动：设置了飞机上、下、左、右移动的方法，并防止飞机出界面；

```
class Player(pygame.sprite.Sprite):
    def __init__(self, plane_img, player_rect, init_pos):
        pygame.sprite.Sprite.__init__(self)
        self.image = []                                 # 用来存储玩家飞机图片的列表
        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                                  # 初始化玩家飞机速度，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.is_hit = False                             # 玩家是否被击中
```

```
    # 发射子弹
    def shoot(self, bullet_img):
        bullet = Bullet(bullet_img, self.rect.midtop)
        self.bullets.add(bullet)

    # 向上移动，需要判断边界
    def moveUp(self):
        if self.rect.top <= 0:
            self.rect.top = 0
        else:
            self.rect.top -= self.speed

    # 向下移动，需要判断边界
    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

    # 向左移动，需要判断边界
    def moveLeft(self):
        if self.rect.left <= 0:
            self.rect.left = 0
        else:
            self.rect.left -= self.speed

    # 向右移动，需要判断边界
    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
```

3. 敌机类：

- 主要有三个函数：
  a. 定义基本属性：敌机图片、敌机坠毁图片、敌机位置、敌机速度；
  b. 如何移动：计算敌机位置；

```
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

    # 敌机移动，边界判断及删除在游戏主循环里处理
    def move(self):
        self.rect.top += self.speed
```

4. 主函数：

```
while running:
    # 控制游戏最大帧率为 60
    clock.tick(60)

    # 生成子弹，需要控制发射频率
    # 首先判断玩家飞机没有被击中
    # 循环15次发射一个子弹
    if not player.is_hit:
        if shoot_frequency % 15 == 0:
            player.shoot(bullet_img)
        shoot_frequency += 1
        if shoot_frequency >= 15:
            shoot_frequency = 0

    # 生成敌机，需要控制生成频率
    # 循环50次生成一架敌机
    if enemy_frequency % 50 == 0:
        enemy1_pos = [random.randint(0, SCREEN_WIDTH - enemy1_rect.width), 0]
        enemy1 = Enemy(enemy1_img, enemy1_down_imgs, enemy1_pos)
        enemies1.add(enemy1)
    enemy_frequency += 1
    if enemy_frequency >= 100:
        enemy_frequency = 0

    for bullet in player.bullets:
        # 以固定速度移动子弹
        bullet.move()
        # 移动出屏幕后删除子弹
        if bullet.rect.bottom < 0:
            player.bullets.remove(bullet)
```

```
    for enemy in enemies1:
        #2. 移动敌机
        enemy.move()
        #3. 敌机与玩家飞机碰撞效果处理
        if pygame.sprite.collide_circle(enemy, player):
            enemies_down.add(enemy)
            enemies1.remove(enemy)
            player.is_hit = True
            break
        #4. 移动出屏幕后删除敌人
        if enemy.rect.top < 0:
            enemies1.remove(enemy)

    #敌机被子弹击中效果处理
    #将被击中的敌机对象添加到击毁敌机 Group 中
    enemies1_down = pygame.sprite.groupcollide(enemies1, player.bullets, 1, 1)
    for enemy_down in enemies1_down:
        enemies_down.add(enemy_down)

    # 绘制背景
    screen.fill(0)
    screen.blit(background, (0, 0))

    # 绘制玩家飞机
    if not player.is_hit:
        screen.blit(player.image[0], player.rect) #将正常飞机画出来
    else:
        # 玩家飞机被击中后的效果处理
        screen.blit(player.image[1], player.rect) #将爆炸的飞机画出来
        running = False
```

```
    # 敌机被子弹击中效果显示
    for enemy_down in enemies_down:
        enemies_down.remove(enemy_down)
        score += 1
        screen.blit(enemy_down.down_imgs, enemy_down.rect) #将爆炸的敌机画出来


    # 显示子弹
    player.bullets.draw(screen)
    # 显示敌机
    enemies1.draw(screen)

    # 绘制得分
    score_font = pygame.font.Font(None, 36)
    score_text = score_font.render('score: '+str(score), True, (128, 128, 128))
    text_rect = score_text.get_rect()
    text_rect.topleft = [10, 10]
    screen.blit(score_text, text_rect)

    # 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()

    # 获取键盘事件（上下左右按键）
    key_pressed = pygame.key.get_pressed()
```

```
    # 处理键盘事件（移动飞机的位置）
    if key_pressed[K_w] or key_pressed[K_UP]:
        player.moveUp()
    if key_pressed[K_s] or key_pressed[K_DOWN]:
        player.moveDown()
    if key_pressed[K_a] or key_pressed[K_LEFT]:
        player.moveLeft()
    if key_pressed[K_d] or key_pressed[K_RIGHT]:
        player.moveRight()

# 游戏 Game Over 后显示最终得分
font = pygame.font.Font(None, 64)
text = font.render('Final Score: '+ str(score), True, (255, 0, 0))
text_rect = text.get_rect()
text_rect.centerx = screen.get_rect().centerx
text_rect.centery = screen.get_rect().centery + 24
screen.blit(game_over, (0, 0))
screen.blit(text, text_rect)

# 显示得分并处理游戏退出
while 1:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            pygame.quit()
            exit()
    pygame.display.update()
```