## 1，把双人游戏改为单人游戏，可以修改绘制血条的方法把两个player改为一个，把绘制飞机架数的函数也需要修改一下

|  |
| --- |
|  |
|  |

## 2.碰撞检测的代码也需要修改，因为我们取消了玩家组，那么玩家需要从外面传递进来

|  |
| --- |
|  |
|  |

## 3.然后在主程序里面创建玩家的时候，就只需要创建一个，只有已经没有玩家组了

|  |
| --- |
|  |

## 然后在游戏主循环里面也需要修改一下

|  |
| --- |
|  |

## 4，需要修改一下menu模块的命令菜单，因为wsad已经没有用了

|  |
| --- |
|  |

# 大功告成，单机项目的完整代码如下

## 10-dad-son-planewar.py

|  |
| --- |
| # 第10个版本，我方受伤  import sys  from constants import \*  import menu  from plane import \*  from funcs import \*  screen = pg.display.set\_mode(SIZE)  pg.display.set\_caption("飞机大战")  clock = pg.time.Clock()  def main():      # 1.显示启动画面      menu.menu\_display(screen) # 调用menu模块的显示菜单方法      # 2.精灵组可以直接使用constants模块里面的all\_sprites和bullets      # 3.创建玩家飞机      player1 = Plane(player\_img1,WIDTH/2,HEIGHT-30,pg.K\_LEFT,pg.K\_RIGHT,pg.K\_UP,pg.K\_DOWN)      # 4.将他们添加到精灵组,这是必须的，因为只有精灵组才有绘制方法      all\_sprites.add(player1)        # 创建敌机      for i in range(4):          new\_enemy() # 这个方法可以创建敌机并且添加到all\_sprites和enemies精灵组里面      # 调用精灵组的更新方法      all\_sprites.update()        global height      running = True      while running: # 游戏主循环            clock.tick(FPS)          for event in pg.event.get():              if event.type == pg.QUIT:                  running = False          # 背景图片向下滚动          screen.blit(bg,(0,height))          height += 2          if height > -168:              height = -936          all\_sprites.update()       # 调用精灵组的更新方法                   # 调用精灵组的更新方法          all\_sprites.draw(screen)   # 绘制精灵          bullet\_hit\_enemy() # 子弹打中敌人的碰撞检测          plane\_get\_power(player1) #我方飞机获取补给的碰撞检测          enemy\_hit\_me(player1)    #敌机子弹打中我方飞机          draw\_screen\_text(screen,player1) # 绘制血条和飞机架数            pg.display.update()        pg.quit()      sys.exit()  if \_\_name\_\_ == '\_\_main\_\_':      main() |

## bullet.py

|  |
| --- |
| import pygame as pg  from constants import \*  shooting\_sound = pg.mixer.Sound(sound\_path+'pew-gunshot-13.wav')  class Bullet(pg.sprite.Sprite):      def \_\_init\_\_(self, x,y) -> None:          super().\_\_init\_\_()          self.image = pg.transform.scale(pg.image.load(pic\_path+'pd333.png'),(15,40))          # self.image = pg.transform.scale(pg.image.load(pic\_path+'enemy\_bullet2.png'),(15,40))          self.image.set\_colorkey(BLACK)          self.rect = self.image.get\_rect()          self.rect.bottom = y          self.rect.centerx = x          self.speed = -10      def update(self, ) -> None:         self.rect.y += self.speed         if self.rect.bottom < 0:             self.kill() # 子弹出界了就销毁 |

## constants.py

|  |
| --- |
| import pygame as pg  from os import path  pg.init()       #初始化pygame模块  pg.mixer.init() #初始化pygame模块的音效模块  font\_name = pg.font.match\_font('arial') # 获取系统里面安装了的字体名称  # 实战图片和音效图片路径  pic\_path = './res/images/'  sound\_path = './res/sounds/'  WIDTH = 480      #窗口宽度  HEIGHT = 600     #窗口高度  SIZE = (WIDTH,HEIGHT) #  FPS = 30   # 帧率  # 定义颜色  BLACK = (0,0,0)  WHITE = (255,255,255)  RED = (255,0,0)  GREEN = (0,255,0)  BLUE = (0,0,255)  YELLOW = (255,255,0)  bg = pg.image.load(pic\_path+'startfield.jpg')  bg = pg.transform.scale(bg,(WIDTH,1536))  height = -936  POWERUP\_TIME = 5000  #飞机的火力持续时间  # 所有精灵的精灵组  all\_sprites = pg.sprite.Group()  # 我方子弹精灵组  bullets = pg.sprite.Group()  # 敌机精灵组  enemies = pg.sprite.Group()  # 敌人子弹精灵组  enemy\_bullets = pg.sprite.Group()  # 补给精灵组  powers = pg.sprite.Group()  # 玩家精灵组  # players = pg.sprite.Group() #单人游戏不需要玩家精灵组  # 分数  score = 0  BAR\_LENGTH =100  #血条长度  BAR\_HEIGHT = 10  #血条高度 |

## enemy.py

|  |
| --- |
| import random as rnd  import pygame as pg  from constants import \*  from funcs import \*  from enemybullet import \*  enemies\_images = [] #保存敌机图片对象的列表  enemies\_list = [      'dj1.png',      'dj2.png',      'dj3.png'  ]  # 加载敌机  for img in enemies\_list:      enemies\_img = pg.image.load(pic\_path+img)      enemies\_img = pg.transform.scale(enemies\_img,(80,60))      enemies\_images.append(enemies\_img)  # 敌机类  class Enemy(pg.sprite.Sprite):      def \_\_init\_\_(self) -> None:          pg.sprite.Sprite.\_\_init\_\_(self) # 调用父类构造函数          self.image\_orig = rnd.choice(enemies\_images) # 随机获取一张图片          self.image\_orig.set\_colorkey(BLACK)          self.image = self.image\_orig.copy()          self.rect = self.image.get\_rect()          self.radius = int(self.rect.width\*.90/2)          self.rect.x = rnd.randrange(0,WIDTH-self.rect.width)          self.rect.y = rnd.randrange(-150,-100)          self.speedy = rnd.randrange(2,5)          self.speedx = rnd.randrange(-3,3)          self.shoot\_delay = 1000          self.last\_shot = pg.time.get\_ticks()      def update(self):          self.rect.x += self.speedx          self.rect.y += self.speedy          if rnd.randrange(10) >= 6:              self.enemy\_shoot()          # 超出范围敌机重生          if(self.rect.top > HEIGHT+10) or (self.rect.left < -25) or (self.rect.right > WIDTH+20):              self.rect.x = rnd.randrange(0,WIDTH-self.rect.width)              self.rect.y = rnd.randrange(-100,-40)              self.speedy = rnd.randrange(1,5)          # 碰到两边会反弹          if self.rect.left < 0 :              self.speedx = -self.speedx          if self.rect.right > WIDTH:              self.speedx = -self.speedx      def enemy\_shoot(self):          now = pg.time.get\_ticks()          if now - self.last\_shot > self.shoot\_delay:              self.last\_shot = now              # d敌机创建子弹              enemy\_bullet = EnemyBullet(self.rect.centerx,self.rect.bottom)              all\_sprites.add(enemy\_bullet)              enemy\_bullets.add(enemy\_bullet)              ene\_shoot\_sound.play() |

## enemybullet.py

|  |
| --- |
| import pygame as pg  from constants import \*  from funcs import \*  enemy\_bullet\_img = pg.transform.scale(pg.image.load(pic\_path+'enemy\_bullet.png'),(15,25))  ene\_shoot\_sound = pg.mixer.Sound(sound\_path+'enemy\_bullet.wav')  class EnemyBullet(pg.sprite.Sprite):      def \_\_init\_\_(self,x,y) -> None:          pg.sprite.Sprite.\_\_init\_\_(self) # 这里不能用super(),必须用pg.sprite.Sprite.\_\_init\_\_(self)          self.image = enemy\_bullet\_img          self.image.set\_colorkey(BLACK)          self.rect = self.image.get\_rect()          self.rect.centerx = x          self.rect.top = y          self.speedy = 5      def update(self):          self.rect.y += self.speedy          if self.rect.top > HEIGHT: # 超出屏幕的子弹会非销毁              self.kill() |

## explosion.py

|  |
| --- |
| import pygame as pg  import random as rnd  from constants import \*  from funcs import \*  #加载爆炸图片  explosion\_anim = {}  explosion\_anim['sm'] = []  explosion\_anim['lg'] = []  explosion\_anim['player'] = []  for i in range(8): #敌机，火山石爆炸      filename = 'dd{}.png'.format(i+1)      img = pg.image.load(pic\_path + filename)      img.set\_colorkey(BLACK)      #大爆炸      img\_lg = pg.transform.scale(img,(75,75))      explosion\_anim['lg'].append(img\_lg)      #小爆炸      img\_sm = pg.transform.scale(img,(32,32))      explosion\_anim['sm'].append(img\_sm)      # 玩家爆炸      filename = 'sonic{}.png'.format(i+1)      img2 = pg.image.load(pic\_path + filename)      img2.set\_colorkey(BLACK)      explosion\_anim['player'].append(img2)  # 爆炸类  class Explosion(pg.sprite.Sprite):      def \_\_init\_\_(self,center,size):          pg.sprite.Sprite.\_\_init\_\_(self)          self.size = size          self.image = explosion\_anim[size][0]          self.rect = self.image.get\_rect()          self.rect.center = center          self.frame = 0          self.last\_update = pg.time.get\_ticks()          self.frma\_rate = 75      def update(self) -> None:          now = pg.time.get\_ticks()          if now - self.last\_update > self.frma\_rate:              self.last\_update = now              self.frame += 1              if self.frame == len(explosion\_anim[self.size]):                  self.kill()              else:                 center = self.rect.center # 获取上一帧的中心点                 self.image = explosion\_anim[self.size][self.frame]                 self.rect = self.image.get\_rect()                 self.rect.center = center   # 设置到当前帧的中心点 |

## funcs.py

|  |
| --- |
| import pygame as pg  from constants import \*  from plane import \*  from enemy import \*  from explosion import \*  import random as rnd  from power import \*  #绘制文本的函数  def draw\_text(surf,text,size,x,y):      font = pg.font.Font(font\_name,size)      text\_surface = font.render(text,True,WHITE)      text\_rect = text\_surface.get\_rect()      text\_rect.midtop = (x,y)      surf.blit(text\_surface,text\_rect)  def draw\_screen\_text(screen,player1):      draw\_text(screen,str(score),18,WIDTH/2,10) #显示分数      draw\_shield\_bar(screen,5,5,player1.shield)      draw\_lives(screen,10,20,player1.lives,player\_mini\_img1)    # 绘制血条  def draw\_shield\_bar(screen,x,y,pct):      pct = max(pct,0)      fill = (pct/100) \* BAR\_LENGTH      outline\_rect = pg.Rect(x,y,BAR\_LENGTH,BAR\_HEIGHT)      fill\_rect = pg.Rect(x,y,fill,BAR\_HEIGHT)      pg.draw.rect(screen,GREEN,fill\_rect)      pg.draw.rect(screen,WHITE,outline\_rect,2)  def draw\_lives(surf,x,y,lives,img):      for i in range(lives):          img\_rect = img.get\_rect()          img\_rect.x = x + 30\*i          img\_rect.y = y          surf.blit(img,img\_rect)  # 创建敌机的函数  def new\_enemy():      enemy = Enemy()      all\_sprites.add(enemy)      enemies.add(enemy)  def bullet\_hit\_enemy():      """我方子弹打中敌人的函数"""      # 先进行我方子弹和敌机的碰撞检测      global score      hits = pg.sprite.groupcollide(enemies,bullets,True,True)      for hit in hits:          score += 50-hit.radius          pg.mixer.Sound(sound\_path+"exp.wav").play()          #创建一个爆炸对象需要调用Explosion类          expl = Explosion(hit.rect.center,'lg')          # 将爆炸对象添加到所有精灵组          all\_sprites.add(expl)          if rnd.random()> 0.9:              pow = Power(hit.rect.center)              all\_sprites.add(pow)              powers.add(pow)          # 每消灭一个敌机，又会创建一个敌机          new\_enemy()  # 我方飞机获取补给的方法，元素碰撞检测  def plane\_get\_power(player):      sound = pg.mixer.Sound(sound\_path+'FX054\_cut.wav')      hits = pg.sprite.spritecollide(player,powers,True)      for hit in hits:          if hit.type == 'shield':              sound.play()              player.shield += rnd.randrange(20,40)              if player.shield >=100:                  player.shield = 100 # 血量不能超过100          elif hit.type == 'gun':              sound.play()              player.powerup()    #敌机子弹打中我方飞机  def enemy\_hit\_me(player):      hits = pg.sprite.spritecollide(player,enemy\_bullets,True,pg.sprite.collide\_circle)      for h in hits:          player.shield -= h.radius \*2 # 被打中会掉血          expl = Explosion(h.rect.center,'sm') # 创建爆炸对象，添加到小爆炸里集合面          all\_sprites.add(expl) # 将爆炸对象添加到所有精灵组          if player.shield <=0: # 血量掉光了就死掉了              pg.mixer.Sound(sound\_path+'exp.wav').play() #播放爆炸音效              dead\_expl = Explosion(player.rect.center,'player')              all\_sprites.add(dead\_expl)              player.hide() # 调用这个方法后几秒钟就会显示player              player.lives -= 1 # 死掉了，就要减少一条命              player.shield = 100 # 把player的血量设置位100，那么他就相当于新创建的了 |

## menu.py

|  |
| --- |
| from os import path  import pygame as pg  from constants import \*  import funcs  def main\_menu(screen):      #加载菜单音乐      pg.mixer.music.load(sound\_path + 'tgfcoder-FrozenJam-SeamlessLoop.ogg')      #循环播放菜单音乐      pg.mixer.music.play(-1)      # 加载开始图片      start\_img = pg.image.load(pic\_path+'menu.png')      start\_img = pg.transform.scale(start\_img,SIZE)      screen.blit(start\_img,(0,0))      pg.display.update()      while True:          event = pg.event.poll() # 只获取一个事件          if event.type == pg.KEYDOWN:              if event.key == pg.K\_RETURN: # 回车键                  break          elif  event.type == pg.QUIT:              pg.quit()              quit()          else:              funcs.draw\_text(screen,"Press [Enter] To Begin",30,WIDTH/2,HEIGHT/2)              funcs.draw\_text(screen,"[↑]",30,WIDTH/2,2\*HEIGHT/3-40)              funcs.draw\_text(screen,"[←] [↓]  [→] ",30,WIDTH/2,2\*HEIGHT/3)              pg.display.update()        pg.display.update()  def menu\_display(screen):      main\_menu(screen)      pg.mixer.music.stop()      pg.mixer.music.load(sound\_path+'battle.ogg')      pg.mixer.music.set\_volume(0.6)      pg.mixer.music.play(-1) |

## missile.py

|  |
| --- |
| import pygame as pg  from constants import \*  missile\_sound = pg.mixer.Sound(sound\_path+'237071-Rocket\_Launcher-02.wav')  class Missile(pg.sprite.Sprite):      def \_\_init\_\_(self, x,y) -> None:          super().\_\_init\_\_()          self.image = pg.transform.scale(pg.image.load(pic\_path+'missile3.png'),(20,55))          self.image.set\_colorkey(BLACK)          self.rect = self.image.get\_rect()          self.rect.bottom = y          self.rect.centerx = x          self.speed = -10      def update(self, ) -> None:         self.rect.y += self.speed         if self.rect.bottom < 0:             self.kill() # 子弹出界了就销毁 |

## plane.py

|  |
| --- |
| from typing import Any  import pygame as pg  from constants import \*  from bullet import \*  from missile import \*  import random as rnd  # 加载玩家飞机图片  player\_img1 = pg.image.load(pic\_path+"my1.png")  player\_mini\_img1 = pg.transform.scale(player\_img1,(20,19))  player\_mini\_img1.set\_colorkey(BLACK)  class Plane(pg.sprite.Sprite):      """"玩家飞机类"""      def \_\_init\_\_(self, playerImg,center,bottom,K\_LEFT,K\_RIGHT,K\_UP,K\_DOWN) -> None:          super().\_\_init\_\_()          self.image = pg.transform.scale(playerImg,(50,38))          self.image.set\_colorkey(BLACK)          self.rect = self.image.get\_rect()          self.rect.centerx = center          self.rect.bottom = bottom          self.speed = 5          self.shield = 100 #血量          self.redius = 20 #杀伤力          self.shoot\_delay = 250 #子弹延迟          self.last\_shot = pg.time.get\_ticks() #最后一次射击时间          self.lives = 3 # 飞机架数          self.hidden = False          self.hide\_timer = pg.time.get\_ticks()          self.power = 3          self.power\_timer = pg.time.get\_ticks() # 火力时间          self.K\_LEFT = K\_LEFT          self.K\_RIGHT = K\_RIGHT          self.K\_UP = K\_UP          self.K\_DOWN = K\_DOWN      def update(self) -> None:          # super().update()          if self.power >=2 and pg.time.get\_ticks() - self.power\_timer > POWERUP\_TIME:              self.power -= 1              self.power\_timer = pg.time.get\_ticks()          if self.hidden and  pg.time.get\_ticks() - self.hide\_timer > 1000:              self.hidden = False              self.rect.centerx = WIDTH/2              self.rect.bottom = HEIGHT - 30          self.shoot()    # 是自动发射子弹的          self.move()     # 设置玩家移动边界        def move(self):          keys = pg.key.get\_pressed() # 获取所有按下的键          if keys[self.K\_RIGHT]:              if self.rect.right > WIDTH: # 右边越界                  self.rect.right = WIDTH              else:                  self.rect.centerx += self.speed          if keys[self.K\_LEFT]:              if self.rect.left < 0 :# 左边越界                  self.rect.left = 0              else:                  self.rect.centerx -= self.speed          if keys[self.K\_UP]:              if self.rect.y < 10:                  self.rect.top = 10              else:                  self.rect.top -= self.speed          if keys[self.K\_DOWN]:              if self.rect.bottom > HEIGHT-10:                  self.rect.bottom = HEIGHT-10              else:                   self.rect.bottom += self.speed      def shoot(self):          now = pg.time.get\_ticks() # 获取现在的时间          if now - self.last\_shot > self.shoot\_delay:              self.last\_shot = now # 保存最新的时间              #单火力              if self.power ==1:                  bullet0 = Bullet(self.rect.centerx,self.rect.top)                  # bullet0 = Missile(self.rect.centerx,self.rect.top)                    # # 子弹需要添加到2个精灵组                  all\_sprites.add(bullet0)                  bullets.add(bullet0)                  shooting\_sound.set\_volume(0.7)                  shooting\_sound.play()              #双火力              if self.power ==2:                  bullet1 = Bullet(self.rect.left,self.rect.centery)                  bullet2 = Bullet(self.rect.right,self.rect.centery)                  all\_sprites.add(bullet1)                  bullets.add(bullet1)                  all\_sprites.add(bullet2)                  bullets.add(bullet2)                  shooting\_sound.set\_volume(0.7)                  shooting\_sound.play()              #三火力              if self.power >=3:                  bullet1 = Bullet(self.rect.left,self.rect.centery)                  bullet2 = Bullet(self.rect.right,self.rect.centery)                  missile1 = Missile(self.rect.centerx,self.rect.top)                  all\_sprites.add(bullet1)                  bullets.add(bullet1)                  all\_sprites.add(bullet2)                  bullets.add(bullet2)                  all\_sprites.add(missile1)                  bullets.add(missile1)                  shooting\_sound.set\_volume(0.7)                  shooting\_sound.play()                  missile\_sound.play()      def powerup(self):          self.power += 3          self.power\_timer = pg.time.get\_ticks()      def hide(self):          self.hidden = True          self.hide\_timer = pg.time.get\_ticks()          self.rect.center = (WIDTH/2,HEIGHT+200) |

## power.py

|  |
| --- |
| import pygame as pg  from constants import \*  import random as rnd  # 加载盾牌和闪电  powerup\_images = {}  powerup\_images['shield'] = pg.image.load(pic\_path+'shield.png')  powerup\_images['gun'] = pg.image.load(pic\_path+'bolt2.png')    class Power(pg.sprite.Sprite):      def \_\_init\_\_(self, center) -> None:          pg.sprite.Sprite.\_\_init\_\_(self)          self.type = rnd.choice(['shield','gun'])          self.image = powerup\_images[self.type]          self.image.set\_colorkey(BLACK)          self.rect = self.image.get\_rect()          self.rect.center = center          self.speedy = 2      def update(self):          self.rect.y += self.speedy          if self.rect.top > HEIGHT:              self.kill() |