**Instructions pour implémenter le jeu « Airplane Wars »**

La qualité du code fourni n’est pas optimale. Une des taches à faire dans ce projet est d’améliorer la qualité du code et de réaliser le re-factoring quand nécessaire. Dans les étapes suivantes, il ne faut pas juste copier le code mais il faut améliorer le code.

Attention, l’objectif du projet est d’assurer la qualité du code et pratiquer le Git-Flow dans une équipe

1. La première étape consiste à afficher l'arrière-plan (code existant dans le code de base)

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

import pygame

from pygame.locals import \*

from sys import exit

SCREEN\_WIDTH = 480

SCREEN\_HEIGHT = 800

# Initialize the game

pygame.init()

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

pygame.display. set\_caption('Airplane Wars')

# Load the background map

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

while True:

# Draw the background

screen.fill (0)

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

# Update the screen

pygame.display.update()

# Process game exits

for event in pygame.event.get():

if event.type == pygame.QUIT:

pygame.quit()

exit()

1. La deuxième étape consiste à montrer l’avion (code existant dans le code de base)

# Load the picture of the plane

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

# Select the position of the plane in the big picture, generate subsurface, and then initialize the position of the plane.

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

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

player\_pos = [200, 600]

while True:

# Draw the background

screen.fill (0)

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

# Draw an airplane

screen.blit (player, player\_pos)

1. La troisième étape consiste à laisser l'avion se déplacer

# Load the picture of the plane

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

# Select the position of the plane in the big picture, generate subsurface, and then initialize the position of the plane.

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

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

player\_pos = [200, 600]]

clock = pygame.time.Clock()

running = True

while running:

# Contrôlez la fréquence d'images maximale du jeu

clock.tick(45)

# Draw the background

screen.fill (0)

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

# Draw an airplane

screen.blit (player, player\_pos)

# Update the screen

pygame.display.update()

# Monitor keyboard events

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

1. **Dessiner les avions ennemis et les faire avancer vers notre avion**

# Enemy class

class Enemy(pygame.sprite.Sprite):

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

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

self.image = enemy\_img

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

self.rect.topleft = init\_pos

self.speed = 2

self.down\_index = 0

def move (self):

self.rect.top += self.speed

# Définir les paramètres liés à la surface utilisés par l'objet avion ennemi

enemy1\_rect = pygame.Rect(534, 612, 57, 43)

enemy1\_img = plane\_img.subsurface(enemy1\_rect)

enemies1 = pygame.sprite.Group()

enemy\_frequency = 0

code à ajouter dans la boucle principale

# Faire apparaître des avions ennemis :

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

code à ajouter dans la boucle principale

# Déplacez l'avion ennemi, s'il dépasse la plage de la fenêtre, supprimez-le

for enemy in enemies1:

enemy.move()

if enemy.rect.top > SCREEN\_HEIGHT:

enemies1.remove(enemy)

enemies1.draw(screen)

# Update the screen

pygame.display.update()

# Process game exits

for event in pygame.event.get():

if event.type == pygame.QUIT:

pygame.quit()

exit()

1. **Re-factoring : créer une classe pour l’avion du joueur**

On va créer une classe Player qui représentes l’ avion du joueur.

Les caracteristiques de l’avion sont : 1) les images, 2) la position de l’avion, 3) la vitesse de l’avion

L’avion peut se deplacer en 4 directions donc on va ajouter 4 methodes de la classe Player : move\_up, move\_down, move\_left, move\_right.

# Players

class Player (pygame.sprite.Sprite):

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

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

self.image = [] # List of pictures of player object wizard

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

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

self.rect = player\_rect[0] # Initialize the rectangle where the picture is located

self.rect.topleft = init\_pos # Initialize the upper left corner coordinates of the rectangle

self.speed = 8 # Initialize the player speed, here is a definite value.

self.img\_index = 0 # Player Wizard Image Index

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

On va définir les images de l’avion. Ces images sont présentées dans le fichier *resources/image/shoot.png.* Les coordonnes de ces images sont présentées dans le fichier *resources/image/shoot.pack* . Ici, on va ajouter des images de l’avion, et des images de l’avion explosé.

# Définir les paramètres liés au joueur

player\_rect = []

player\_rect.append(pygame.Rect(0, 99, 102, 126)) # Zone d'image du sprite du joueur

player\_rect.append(pygame.Rect(165, 360, 102, 126))

player\_rect.append(pygame.Rect(165, 234, 102, 126)) # Zone d'image du sprite d'explosion du joueur

player\_rect.append(pygame.Rect(330, 624, 102, 126))

player\_rect.append(pygame.Rect(330, 498, 102, 126))

player\_rect.append(pygame.Rect(432, 624, 102, 126))

player\_pos = [200, 600]

player = Player(plane\_img, player\_rect, player\_pos)

Il faut noter que vous devez copier tous le code et le mettre à un bon endroit, il n’est pas difficile à trouver le bon endroit.

# Draw an airplane

# screen.blit (player, player\_pos)

screen.blit(player.image[player.img\_index], player.rect)

Attention, c’est un cours de la qualité. Il faut alors bien structurer votre code et améliorer ces exemples de code que je vous fournies.

# # Monitor keyboard events

# 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

# Écouter les événements du clavier

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

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()

1. **Dessiner les puces (bullets) envoyés par notre avion**

# Bullets

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

On va modifier la classe Player pour ajouter l’attribut *bullets* et la méthode *shoot*

# Players

class Player (pygame.sprite.Sprite):

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

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

self.image = [] # List of pictures of player object wizard

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

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

self.rect = player\_rect[0] # Initialize the rectangle where the picture is located

self.rect.topleft = init\_pos # Initialize the upper left corner coordinates of the rectangle

self.speed = 8 # Initialize the player speed, here is a definite value.

self.bullets = pygame.sprite.Group() # Collection of bullets fired by the player's aircraft

self.img\_index = 0 # Player Wizard Image Index

def shoot (self, bullet\_img):

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

self.bullets.add (bullet)

On doit charger le son et l’image de la puce (bullets)

bullet\_sound = pygame.mixer.Sound('resources/sound/bullet.wav')

bullet\_sound.set\_volume(0.3)

# Définir les paramètres liés à la surface utilisés par l'objet puce (bullets)

bullet\_rect = pygame.Rect(1004, 987, 9, 21)

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

shoot\_frequency = 0

Ensuite il faut modifier la boucle principale pour contrôler la fréquence des tirs et dessiner les puces. On doit aussi faire avancer les puces.

while True:

# Draw the background

screen.fill (0)

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

# Contrôler la fréquence des tirs de balles et des balles de feu

if shoot\_frequency % 15 == 0:

bullet\_sound.play()

player.shoot(bullet\_img)

shoot\_frequency += 1

if shoot\_frequency >= 15:

shoot\_frequency = 0

# Déplacer la puce, la supprimer si elle dépasse le cadre de la fenêtre

for bullet in player.bullets:

bullet.move()

if bullet.rect.bottom < 0:

player.bullets.remove(bullet)

# Draw an airplane

# screen.blit (player, player\_pos)

screen.blit(player.image[player.img\_index], player.rect)

player.bullets.draw(screen)

1. **Avion ennemies exploser si touchés par une puce**

On doit modifier la classe Enemy pour ajouter les images de l’explosition des avions ennemies *enemy\_down\_imgs* .

# Enemy class

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

On va charger le son et les images des ennemies explosées

enemy1\_down\_sound = pygame.mixer.Sound('resources/sound/enemy1\_down.wav')

enemy1\_down\_sound.set\_volume(0.3)

enemy1\_down\_imgs = []

enemy1\_down\_imgs.append(plane\_img.subsurface(pygame.Rect(267, 347, 57, 43)))

enemy1\_down\_imgs.append(plane\_img.subsurface(pygame.Rect(873, 697, 57, 43)))

enemy1\_down\_imgs.append(plane\_img.subsurface(pygame.Rect(267, 296, 57, 43)))

enemy1\_down\_imgs.append(plane\_img.subsurface(pygame.Rect(930, 697, 57, 43)))

# Stockez des avions détruits pour le rendu d'animations de sprites d'épaves

enemies\_down = pygame.sprite.Group()

Ensuite, on doit modifier le code ou on ajouter des ennemies pour ajouter *enemy1\_down\_imgs*

# Faire apparaître des avions ennemis

if enemy\_frequency % 50 == 0:

enemy1\_pos = [random.randint(0, SCREEN\_WIDTH - enemy1\_rect.width), 0]

enemy1 = Enemy(enemy1\_img,

enemy1\_down\_imgs, #Ajouter ceci car on a modifie la classe Enemy

enemy1\_pos)

enemies1.add(enemy1)

enemy\_frequency += 1

if enemy\_frequency >= 100:

enemy\_frequency = 0

On va supprimer l'ennemi en cas de collision, ajouter *enemy\_down* pour le dessiner

# Ajoutez l'objet avion ennemi touché au groupe d'avions ennemis détruits, utilisé pour rendre l'animation de destruction

enemies1\_down = pygame.sprite.groupcollide(enemies1, player.bullets, 1, 1)

for enemy\_down in enemies1\_down:

enemies\_down.add(enemy\_down)

En fin, il faut dessiner les ennemies explosees

# Dessinez l'animation de l'épave

for enemy\_down in enemies\_down:

if enemy\_down.down\_index == 0:

enemy1\_down\_sound.play()

if enemy\_down.down\_index > 7:

enemies\_down.remove(enemy\_down)

# score += 1000

continue

screen.blit(enemy\_down.down\_imgs[enemy\_down.down\_index // 2], enemy\_down.rect)

enemy\_down.down\_index += 1

1. Game Over si l'ennemi touche l'avion

On va déterminez si l’avion touche des ennemies. On doit ajouter egalement un attribut pour la classe Player : is\_hit

# Déplacez l'avion ennemi, s'il dépasse la plage de la fenêtre, supprimez-le

for enemy in enemies1:

enemy.move()

# Déterminez si le joueur a été touché

if pygame.sprite.collide\_circle(enemy, player):

enemies\_down.add(enemy)

enemies1.remove(enemy)

player.is\_hit = True

game\_over\_sound.play()

break

if enemy.rect.top > SCREEN\_HEIGHT:

enemies1.remove(enemy)

self.img\_index = 0 # Player Wizard Image Index

self.is\_hit = False # Is the player hit?

self.bullets = pygame.sprite.Group() # Collection of bullets fired by the player's aircraft

On va replacer le code pour dessiner le Player.

Au debut on a

screen.blit(player.image[player.img\_index], player.rect)

On va replacer par ceci, plus complex

# dessiner l'avion du joueur

if not player.is\_hit:

screen.blit(player.image[player.img\_index], player.rect)

# Changer l'index de l'image pour animer l'avion

player.img\_index = shoot\_frequency // 8

else:

player.img\_index = player\_down\_index // 8

screen.blit(player.image[player.img\_index], player.rect)

player\_down\_index += 1

if player\_down\_index > 47:

running = False

on doit ajouter cette ligne

player\_down\_index = 16

On charge le photo pour afficher quand le jeu est gameover

game\_over = pygame.image.load('resources/image/gameover.png')

game\_over\_sound = pygame.mixer.Sound('resources/sound/game\_over.wav')

game\_over\_sound.set\_volume(0.3)

Ajouter ce code à la fin du programme pour afficher le GameOver ne pas quiter

screen.blit(game\_over, (0, 0))

while 1:

for event in pygame.event.get():

if event.type == pygame.QUIT:

pygame.quit()

exit()

pygame.display.update()

1. **Afficher le nombre de points gagnés**

On ajoute une variable globale *score* et augmenter le *score* si une ennemie est explosé

score = 0

# Dessinez l'animation de l'épave

for enemy\_down in enemies\_down:

if enemy\_down.down\_index == 0:

enemy1\_down\_sound.play()

if enemy\_down.down\_index > 7:

enemies\_down.remove(enemy\_down)

score += 1000

continue

screen.blit(enemy\_down.down\_imgs[enemy\_down.down\_index // 2], enemy\_down.rect)

enemy\_down.down\_index += 1

On dessine le score dans la boucle principale

# dessiner le score

score\_font = pygame.font.Font(None, 36)

score\_text = score\_font.render(str(score), True, (128, 128, 128))

text\_rect = score\_text.get\_rect()

text\_rect.topleft = [10, 10]

screen.blit(score\_text, text\_rect)

… et le code pour dessiner le score lorsque GameOVER

font = pygame.font.Font(None, 48)

text = font.render('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)  
while 1:

for event in pygame.event.get():

if event.type == pygame.QUIT:

pygame.quit()

exit()

1. Les fonctionnalités complémentaires vous pouvez ajouter dans le jeu
   1. Faire bouger le background comme l’avion avance bien dans l’espace
   2. Changer le niveau du jeu
      1. Changer la vitesse des puces
      2. Changer la vitesse des ennemies
      3. Ajouter les bombes délivrées par des ennemies
   3. Changer la puissant de l’avion
      1. Changer l’avion (voir des images d’avion dans resources/image/shoot.png) si on arrive à 50 000 points
      2. Ajouter d’autres types d’armes pour l’avion

**Annexes**

1. resources/image/shoot.png

­­ Une image contenant texte, candélabre

Description générée automatiquement