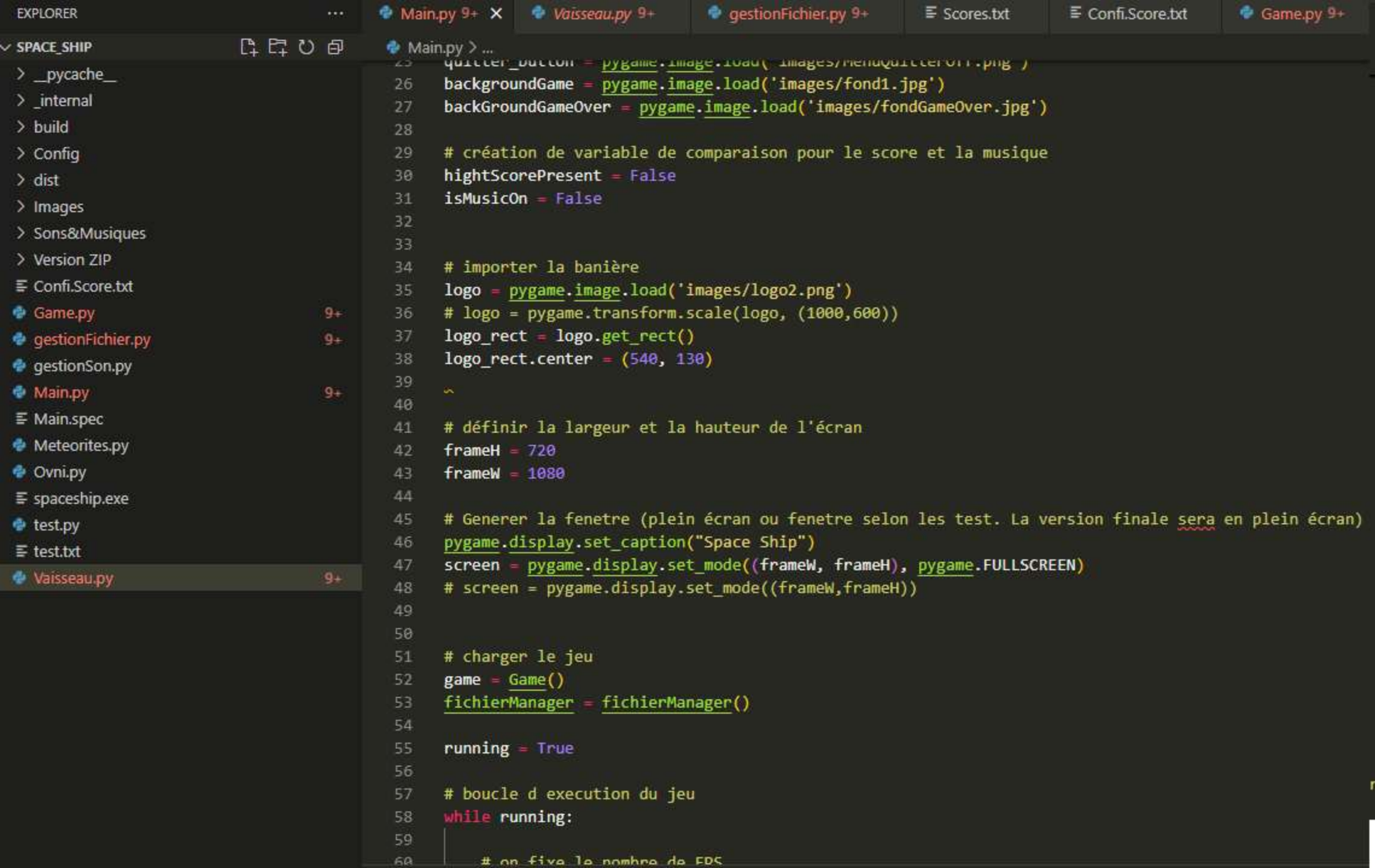
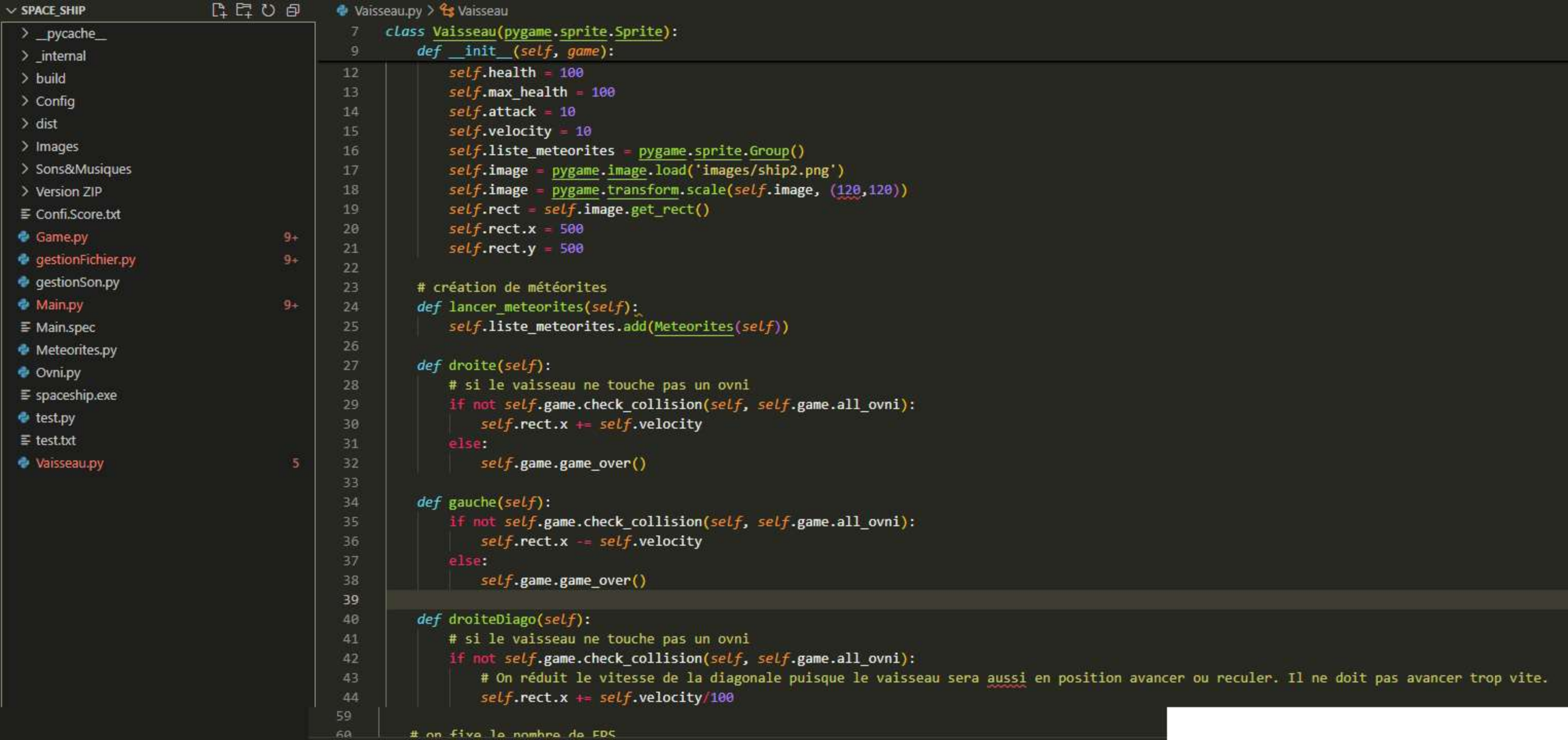

EXPLORER
SPACE_SHIP
pycache
_internal
build
Config
dist
Images
Sons&Musiques
Version ZIP
Confi.Score.txt
Game.py 9+
gestionFichier.py 9+
gestionSon.py
Main.py 9+
Main.spec
Meteorites.py
Ovni.py
spaceship.exe
test.py
test.txt
Vaisseau.py 9+
Main.py 9+
Vaisseau.py 9+
gestionFichier.py 9+
Scores.txt
Confi.Score.txt
Game.py 9+
Main.py > ...
backgroundGame = pygame.image.load('images/fond1.jpg')
backGroundGameOver = pygame.image.load('images/fondGameOver.jpg')
# création de variable de comparaison pour le score et la musique
hightScorePresent = False
isMusicOn = False
# importer la banière
logo = pygame.image.load('images/logo2.png')
# logo = pygame.transform.scale(logo, (1000,600))
logo_rect = logo.get_rect()
logo_rect.center = (540, 130)
# définir la largeur et la hauteur de l'écran
frameH = 720
frameW = 1080
# Generer la fenetre (plein écran ou fenetre selon les test. La version finale sera en plein écran)
pygame.display.set_caption("Space Ship")
screen = pygame.display.set_mode((frameW, frameH), pygame.FULLSCREEN)
# screen = pygame.display.set_mode((frameW,frameH))
# charger le jeu
game = Game()
fichierManager = fichierManager()
running = True
# boucle d execution du jeu
while running:
r ou reculer. Il ne doit pas avancer trop vite.


SPACE_SHIP
__pycache__
_internal
build
Config
dist
Images
Sons&Musiques
Version ZIP
Confi.Score.txt
Game.py 9+
gestionFichier.py 9+
gestionSon.py
Main.py 9+
Main.spec
Meteorites.py
Ovni.py
spaceship.exe
test.py
test.txt
Vaisseau.py 5
Vaisseau.py > Vaisseau
class Vaisseau(pygame.sprite.Sprite):
def __init__(self, game):
self.health = 100
self.max_health = 100
self.attack = 10
self.velocity = 10
self.liste_meteorites = pygame.sprite.Group()
self.image = pygame.image.load('images/ship2.png')
self.image = pygame.transform.scale(self.image, (120,120))
self.rect = self.image.get_rect()
self.rect.x = 500
self.rect.y = 500
# création de météorites
def lancer_meteorites(self):
self.liste_meteorites.add(Meteorites(self))
def droite(self):
# si le vaisseau ne touche pas un ovni
if not self.game.check_collision(self, self.game.all_ovni):
self.rect.x += self.velocity
else:
self.game.game_over()
def gauche(self):
if not self.game.check_collision(self, self.game.all_ovni):
self.rect.x -= self.velocity
else:
self.game.game_over()
def droiteDiago(self):
# si le vaisseau ne touche pas un ovni
if not self.game.check_collision(self, self.game.all_ovni):
# On réduit le vitesse de la diagonale puisque le vaisseau sera aussi en position avancer ou reculer. Il ne doit pas avancer trop vite.
self.rect.x += self.velocity/100

Game.py > Game > _init_

```python
from Vaisseau import Vaisseau
from Ovni import Ovni
from gestionSon import SoundManager
import pygame
import time
# import random


# generation du jeu
class Game:

    def __init__(self):

        # Définir les variables permettant de détecter dans quelles phase de jeu nous nous trouvons (Menu, Game ou Game Over). On commence par le menu
        self.is_playing = False
        self.is_GameOver = False
        self.is_Menu = True

        # Définir le nombre d'Ovni à lancer
        self.nb_Ovni = 0

        # Vitesse de scroll
        self.vitesseScroll = 5

        # Définir le timer
        self.timer_deb = time.time()

        # Définir la texture de fond d'écran ainsi que la variable de déplacement
        # self.background = pygame.image.load('images/fond'+str(random.randint(1, 3))+'.jpg')
        self.background = pygame.image.load('images/fond3.jpg')
        self.scroll=5

        # generer notre vaisseau
        self.all_vaiseaux = pygame.sprite.Group()
        self.Vaisseau = Vaisseau(self)


    # on fixe le nombre de FPS
```