Ce code est un jeu simple de "Space Invaders" réalisé avec la bibliothèque Pygame en Python.

1. Importations et Initialisation

import pygame : Importe la bibliothèque Pygame, essentielle pour créer des jeux vidéo en Python. Elle fournit des fonctions pour gérer les graphismes, le son, les événements, etc.

import random : Permet de générer des nombres aléatoires, utilisés pour positionner les ennemis et déterminer des événements aléatoires dans le jeu.

import math : Utilisé pour effectuer des calculs mathématiques, comme la distance entre deux points, essentielle pour détecter les collisions et les effets physiques.

import sys : Fournit un accès à certaines variables et fonctions qui interagissent avec l'interprète Python, comme quitter le programme.

import os : Permet d'interagir avec le système d'exploitation, notamment pour gérer les chemins de fichiers.

pygame.init() : Initialise tous les modules Pygame nécessaires au jeu. C'est une étape essentielle avant d'utiliser toute fonctionnalité de Pygame.

1. Configuration de l'Écran

screen\_width et screen\_height : Définissent la taille de la fenêtre du jeu, ici 880 pixels de large et 600 pixels de haut.

pygame.display.set\_mode(...) : Crée la fenêtre du jeu avec la taille spécifiée. Retourne un objet de surface utilisable pour dessiner sur l'écran.

pygame.display.set\_caption(...) : Définit le titre de la fenêtre du jeu, affiché dans la barre de titre.

1. Chargement des Ressources

resource\_path : Cette fonction s'assure que les ressources (images, sons, polices) se chargent correctement, que ce soit dans un environnement de développement ou dans un exécutable créé avec des outils comme PyInstaller.

sys.\_MEIPASS : Variable définie lorsqu'un programme empaqueté est exécuté. Si elle n'existe pas, le chemin actuel est utilisé.

os.path.join(...) : Combine le chemin de base avec le chemin relatif de la ressource pour assurer une gestion correcte sur différents systèmes d'exploitation.

1. Chargement des Images et Sons

Chargement des images : Plusieurs images sont chargées pour le jeu, comme l'arrière-plan, l'icône du jeu, l'image du joueur, les balles et les images des ennemis.

pygame.image.load(...) : Charge une image depuis un fichier et retourne un objet de surface utilisable sur l'écran.

pygame.display.set\_icon(...) : Définit l'icône de la fenêtre du jeu.

Chargement du son : La musique de fond est chargée et son volume est ajusté. Elle est jouée en boucle infinie (-1).

Toutes les ressources (sprites, sons, musiques, tailles et styles de police) ont un chemin fixe. Vous devrez modifier ce chemin selon l'emplacement des fichiers sur votre ordinateur.

1. Variables Globales

Variables du joueur : Définissent les coordonnées initiales du joueur (playerx, playery) et la variable playerx\_change qui contrôle le mouvement horizontal. Cette partie du code définit où le personnage apparaîtra.

Variables des ennemis : Listes stockant les positions et les images des ennemis, ainsi que leurs vitesses de déplacement.

1. Fonctions du Jeu

a. Dessin et Collisions

draw\_player : Dessine l'image du joueur à la position (x, y) sur l'écran. Cette fonction est appelée en continu pour donner l'illusion de mouvement.

screen.blit(...) : Dessine une surface (ici, l'image du joueur) sur l'écran aux coordonnées spécifiées.

isCollision : Calcule la distance entre deux points (ex. la position d'un ennemi et d'une balle) et détermine s'il y a collision. Si la distance est inférieure à 27 pixels, il y a collision.

1. Boucle Principale du Jeu

gameloop : La boucle principale du jeu, qui s'exécute tant que running est True. Elle gère la logique du jeu, met à jour les positions et affiche les éléments.

1. Gestion des Événements

Les événements (fermeture de la fenêtre, appui sur des touches) sont gérés ici. Si un événement de fermeture est détecté, le jeu se termine.

1. Mouvement et Tir

Le mouvement du joueur est contrôlé par les touches flèches gauche et droite. La vitesse peut être ajustée en modifiant playerx\_change.

1. Limites du Terrain

Si un ennemi atteint une position critique (440 pixels sur l'axe Y), l'écran "Game Over" s'affiche et le jeu redémarre.

1. Collisions et Score

Si une balle touche un ennemi, la balle est réinitialisée, le score augmente et l'ennemi change de position.

1. Tir des Ennemis

Une probabilité est définie pour que le premier ennemi tire une balle. Augmenter cette probabilité augmente la difficulté.

1. Mouvement des Balles

Les balles des ennemis descendent, celles du joueur montent. Si une balle ennemie touche le joueur, "Game Over" s'affiche.

1. Mise à Jour de l'Écran

Le joueur et le score sont dessinés, puis l'écran est mis à jour avec pygame.display.update().

1. Écran de Game Over

game\_over\_screen : Affiche "Game Over" et le score final. Attendre une touche pour redémarrer.

1. Menu Principal

main\_menu : Affiche un menu d'accueil, attendant l'appui sur "Entrée" pour commencer le jeu.

1. Fonction Principale

main : Gère le flux du jeu, entre le menu principal et la boucle du jeu, permettant un redémarrage après "Game Over".