Projet libre

Projet : ‘Attack on Titan - The Game’ – Jeu en 2D basé sur l’univers de l’attaque des titans
Auteur.e.s : IMBERT Violette et BIHET Lucie
Langage utilisé : Python 3.9 avec les bibliothèques : pygame, math, random et os, réalisé sur Pyzo

Fonctionnalités :
Accueil :
Un menu d’accueil avec le titre du jeu, le meilleur score tiré du fichier de meilleur score en haut à gauche, un bouton pour obtenir les règles du jeu en haut à droite et 4 images de personnages pour choisir lequel est joué.
Une fois la page de règles ouvertes, il suffit de re-cliquer sur le bouton pour retourner au menu.
Cliquer sur un des 4 personnages pour le choisir et lancer la partie.
Le meilleur score est initialement à zéro et évolue selon les parties du joueur. De plus il est sauvegardé dans un fichier « meilleur score ».

Jeu partie vague de titans (phase 1) :
Des titans vont arriver sur le joueur par la droite.
Il suffit de se rapprocher d’eux et d’appuyer sur la barre espace pour les attaquer (on peut soit cliquer plusieurs fois sur la barre ou la maintenir appuyer sans bouger le personnage).
Les titans font perdre de la vie au joueur lorsqu’ils sont en contact mais plus lorsqu’on les attaque.
Les titans et le joueur ont une barre de vie.
Chaque titan tué rapporte 1 point.
Le titan bestial (au fond à gauche) lance quelques cailloux sur le joueur durant cette phase.
La phase se fini après avoir vaincu une dizaine de titan.

Jeu partie esquive de cailloux (phase 2) :
Une fois la première phase terminée, une seconde commence.
Le titan bestial lance beaucoup plus de cailloux qu’ils faut esquiver jusqu’à ce que la barre orange en haut de l’écran se termine (une dizaine de seconde).
La première phase recommence alors et le jeu continue ainsi jusqu’à la mort du joueur.
Le but est donc de faire le meilleur score possible.

Musique :
Une musique de fond est jouée en continue.
De plus il y a un bruitage lors du lancement du jeu et un bruitage lorsque le joueur meurt.
- Bruitage de lancement : « Shinzou wo sasageyo » : signifie « Dévouons notre cœur » en japonais, cette réplique est dite par le général Erwin Smith dans l’anime de l’attaque des titans.
- Bruitage de mort : « Eren !! » : prénom dit par le personnage de Mikasa également dans l’anime de l’attaque des titans.
- Musique : OST tiré de l’anime de l’attaque des titans (sous copyright)

Contexte du jeu : l’univers de l’attaque des titans (Shingeki no Kyojin (SNK) en japonais) :
SNK est d’abord un manga de 139 chapitres qui a été adapté en anime.
Tout se passe dans un univers où les titans existent et menacent de détruire l’humanité.
Une population humaine vit entouré de murs pour se protéger des titans à l ‘extérieur.
Un jeune homme nommé Eren découvre posséder le pouvoir des titans et va l’utiliser à bon escient pour tenter de vaincre la menace des titans. Il intègre donc le bataillon d’exploration qui est un groupe de reconnaissance destiné à explorer l’extérieur des murs.

D’autres humains possèdent eux aussi un pouvoir de titan tel que le titan bestial mais dont les intentions vont à l’encontre de celle d’Eren et de sa bande. C’est ainsi qu’ils se retrouvent tous à devoir combattre.

Touches :
Barre espace : attaquer
Flèches droite/gauche : se déplacer

Détails :
class Titan :
- deplacement_titan : déplace le sprite de quelques pixels à gauche pour obtenir un mouvement fluide
- degats : inflige les dégâts au titan et gère sa mort (pour le score par exemple)
- barre_de_vie : dessine la barre de vie selon la vie restante du titan

class Caillou :
- supprimer_caillou : enlève le caillou du groupe de sprite lorsqu’il touche le sol
- chute : déplace le sprite de quelques pixels en bas pour obtenir un mouvement fluide

class CaillouTombant :
- pluie_de_cailloux : ajoute un caillou dans le groupe de sprite pour la gestion de la chute de cailloux
- barre_evenement_chute_rapide : dessine la barre qui définie la durée de la phase 2 du jeu

class Joueur :
- droite : déplace le sprite de quelques pixels à droite pour obtenir un mouvement fluide
- gauche : déplace le sprite de quelques pixels à gauche pour obtenir un mouvement fluide
- barre_de_vie : dessine la barre de vie selon la vie restante du joueur
- degats : inflige les dégâts au joueur et gère sa mort

class Son :
- play_son : joue le son entré en paramètre

class Score :
- lire_fichier : lit le fichier
- sauvegarder_fichier : écrit et sauvegarde dans le fichier
- afficher_meilleur_score : affiche le score inscrit dans le fichier meilleur score sur l’écran d’accueil
- update_score : augmente le score du joueur de 1, est appelé dès qu’un titan est tué
- afficher_score : affiche le score du joueur en direct lors de la partie

class Jeu :
- apparition_titan : ajoute un titan au groupe de sprite
- check_collision : vérifie si deux objets sont en collision
- start : fait apparaître des titans à l’écran, peut être utilisé pour le renouvellent de la phase 1 et le lancement du jeu
- remise_a_zero : réinitialise toutes les variables qui doivent être réinitialisé à la fin du jeu ou à la fin de la phase 2
- game_over : gère la fin de la partie quand le joueur a perdu
- afficher_regles : charge l’image pour afficher les règles
- gestion_chute_caillou : gère toute la chute de caillou et les phase de celle-ci
- update : fonction met à jour à chaque tour dans la boucle principale tous les paramètres nécessaires tel que les barres de vie, le déplacement, ect...

def lancement_jeu : importe les images nécessaire à l'accueil du jeu, contient la boucle principale gérant l'appui des touches et le clique de la souris et permet de garder pygame ouvert

def main : crée la fenêtre, lance la musique de fond et lance le jeu

PS : le __init__ de chaque classe permet d'en initialiser les éléments tel que la quantité de vie, l'image du sprite ou la vitesse de déplacement par exemple

Difficultés rencontrées :
- apprendre une toute nouvelle façon de programmer (orientée objet) a été compliqué
- maîtriser une interface graphique est aussi nouveau même si facilité avec pygame
- beaucoup de problème lié au fait que l'on soit novice dans ces domaines
- difficile de créer un cahier des charges clairs à partir de rien
- le seul moyen de fermer le jeu est de cliquer sur la croix de la fenêtre car il n'y a plus assez de place sur l'écran d'accueil pour créer un bouton pour quitter
- la fonction lancement_jeu est très longue car contient le chargement de toutes les images utilisées pour l'accueil du jeu et contient également la boucle principale

Aides utilisées :
- https://pub.phyks.me/sdz/sdz/interface-graphique-pygame-pour-python.html pour les bases
- documentation pygame
- forum d'aide tel que OpenClassroom pour certains détails (implémentation de la musique par exemple)