Gaël Riolland, 44 ans

Parcours Développeur d'application – Python /Django

Lien vers le code source https://github.com/donkansigui/OCR_P3.git

PROJET 3 : AIDEZ MAC GYVER A S'ECHAPPER

Introduction

Ayant débuté le parcours de Développeur d'application Python sans avoir la moindre expérience dans le domaine du développement, le projet qui doit être réalisé me parait semblable à l'ascension de l'Everest, autant dire que j'ai l'impression que le ciel me tombe sur la tête.

Après avoir validé les cours liés au projet je ne sais toujours pas comment organiser ce programme, j'entreprends de suivre des cours sur d'autres supports (sites web) et d'effectuer des exercices sur « France IOI » pour augmenter ma compréhension du développement, des algorithmes et plus particulièrement du langage Python.

Après avoir changé de mentor accompagnateur nous entreprenons de reprendre l'initiation au développement en Python à sont début tout en jetant les bases de notre programme, Le début de mon programme se met en place et je décide de prendre un virage relativement serré, quitter Windows 10 et Pycharm pour m'initier à Linux et l'éditeur de texte Atom, chose qui n'a pas été aisée dans la mesure où j'étais bien plus à l'aise avec une souris qu'avec un terminal, mais il faut avouer que le choix était judicieux pour travailler sur ce projet.

I. Démarrage du projet

Choix des outils pour mener à bien le projet

J'ai développé ce projet en utilisant le langage de programmation Python pour la réalisation des algorithmes nécessaires au programme, le module random pour le placement aléatoire des objets ainsi que la bibliothèque Pygame pour la création de la fenêtre de jeu (sa taille et son apparence) et les différents éléments qui la compose, les déplacements du personnage et la gestion des collisions.

II. Etablissement de la structure de départ

Pour construire le squelette du jeu, j'ai fait le choix de débuter à partir d'un labyrinthe dont les éléments (murs, McGyver, garde) sont représentés par des caractères, ces chaines de caractères (ou string) sont ensuite traités par un algorithme (une fonction qui à été mise au point pour vérifier la longueur des chaines de caractères et si les caractères remplissaient les conditions établies) les chaines de caractères sont ensuite ajoutés à une liste contenue dans une variable qui sera appelée dans le script principal pour l'affichage de la structure de l'interface graphique et l'emplacement des différents éléments.

III. Création des classes

Le code doit être réparti en plusieurs sections, en plusieurs objets, en différentes classes pour réunir les données et les fonctionnalités de chaque objet (attributs et méthodes) qui pourront ainsi être instanciés dans la boucle principale de notre programme, les Classes gèrent les catégories suivantes :

Affichage de l'interface utilisateur

La première étape consistait à afficher la fenêtre du jeu et les différents éléments (ou sprites) qui la compose, leur affichage est géré grâce au commandes du module Pygame, qui fait appel à un algorithme pour reconnaitre les emplacements des murs, personnages et espaces vides du labyrinthe représentés à la base par des chaines de caractères qui seront ajoutés à une liste qui sera parsée par notre algorithme.

Affichage des éléments de l'inventaire

Pour afficher les trois éléments nécessaires à la validation de la sortie de notre héro dans l'inventaire en bas de l'écran, une méthode a été créée, méthode qui fait appel à la commande « get_rect » du module Pygame pour transformer nos images en rectangles et les positionner.

Déplacement et gestion des collisions

Le déplacement du personnage et les collisions avec les objets qu'il doit ramasser ainsi qu'avec le garde est également géré grâce aux commandes du module Pygame, gestion des évènements (EVENT & EVENT_KEY) conditions et modifications de variables si ces conditions sont remplies et « Colliderect » (collisions entre deux rectangles).

Conditions de victoire ou d'échec

Idem que pour le déplacement et la gestion des collisions, des conditions ont étés établies et sont validées par les valeurs booléennes True et False, si les trois conditions sont validées par la valeur Booléenne True en cas de collision entre les sprites des deux personnages principaux du jeu (Mc Gyver & Murdock) l'écran de succès s'affiche, dans le cas contraire l'écran de défaite s'affiche.

P.-S. :

Une musique à été ajoutée, si vous n'avez pas été un grand fan de la série télévisée Mac Gyver, rien ne vous empêche de mettre le volume à zéro sur votre système audio ou de modifier la ligne de code

self.pygame.mixer.music.set_volume(0.6) en mettant le volume à (0.0) dans les paramètres

cette ligne se trouve dans la méthode «def init_music » de la classe Gui du fichier gui.py