Ly Mickael

Parcours Développeur web/applications Python

Soutenance du projet 3 : Aidez Macgyver à s'échapper !

Lien code source vers github : https://github.com/Goro-Majima/Macgyver

Dans le cadre de ce projet, j'ai pour but de créer un jeu de labyrinthe en 2D. Cela peut paraître facile car ce jeu est d'apparence simple, mais la conception demande une certaine technique et architecture qui seront déployés méthodiquement. J'ai opté pour l'approche en programmation objet. J'ai eu pendant le projet quelques blocages dus à un algorithme pas assez complet ou encore des variables non définies. Mais ce que je retiendrai est surtout le moment de joie quand je trouve finalement la solution à mon cas.

Ce projet est développé sous sublime text 3 sous windows 10 et versionné sur github et respecte la PEP8.

Versions python et librairies :

Python 3.7

Pygame , random

Structure du programme :

Scripts

escape.py : Fichier qui organise la structure et le déroulement du programme.

classes.py : Emplacement des classes et des fonctions créés pour apporter plus de clarté.

Constants.py : Déclaration des variables afin de pouvoir les modifier plus simplement. +(images et fichier « labyrinthe »)

La démarche

Création de la fenêtre : division de la dimension en 15 cases pour créer des coordonnées.

Création du fond : choisir une image, la charger et l'afficher

Création du labyrinthe : créer l'algorithme qui va lire le fichier maze et répartir les murs et le gardien sur la fenêtre.

Création du héro : initialiser le héro au point de départ et lui créer ses fonctions pour ses déplacements en tenant compte des murs et des dimensions du jeu.

Dispersion des objets : disperser les objets grâce au module random en tenant compte des murs et points de départ et arrivée.

Récupération des objets par le héro : disparation des objets lors du ramassage du héro

Message de fin : Détermine si le héro a gagné ou perdu en fonction des éléments utilisés face au gardien.

A propos de l'algorithme :

Affichage d'un menu ouvert tant que le bouton ECHAP ou ENTREE n'est pas appuyée.

Ensuite créer une boucle avec un booléen qui conditionnera la fin de la partie. Tant que le héro n'est pas sur le gardien, la variable reste la même et le jeu continuera.

L'algorithme pour la création est la plus complexe du jeu : un fichier (représentation du labyrinthe) sera parcouru. Il faudra le parcourir ligne par ligne. Il faudra donc une boucle qui lira la dernière lettre d'une ligne et une autre boucle qui permettra d'aller la ligne suivante et ainsi de suite. Dans la même logique, pour afficher le labyrinthe, on parcourt les listes et on affiche l'image selon la lettre.

Les déplacements de Macgyver sont soumis deux conditions, la fonction s'assure qu'il ne sortira pas du jeu en gardant ses déplacements horizontaux et verticaux cohérents avec la taille de la fenêtre.

La dispersion des objets se fait avec le module random. Un nombre aléatoire entre 0 et 15 sera attribué pour chaque coordonnée de chaque objet(x, y) . On s'assurera que ces coordonnées soient hors du mur et des autres objets.

Pour les faire disparaître lors du ramassage de Macgyver, une variable s'incrémente et l'affichage lors de la prochaine boucle n'aura pas lieu.

Enfin, lors de la fin du jeu, une variable totalisera le nombre d'objets ramassés. S'il est positionné sur le gardien avec un total de 3 il gagne, sinon il perd.

Difficultés rencontrées:

L'architecture du labyrinthe est plutôt complexe à organiser.

Solutions : Tutoriels sur openclassrooms et vidéos sur Youtube.

Lors de certains blocages, je n'ai pas pu trouver de réponses sur google et notamment sur Stackoverflow.

Solutions : Réflexion sur les tests à apporter, j'ai parfois procéder par tâtonnements afin d'éliminer certaines hypothèses. Relecture approfondie du code lors de modifications ou de création.