Projet 03 : Aidez MacGyver à s'échapper

Vincent Houillon - Formation "Développeur d'application - Python"

- Projet 03 : Aidez MacGyver à s'échapper
  - Introduction
  - Arborescence du code
  - Extrait du code
  - Difficultés rencontrées
  - Conclusion

Introduction

Le projet consiste à développer un jeu de labyrinthe en 2D avec les fonctionnalités suivantes :

- un seul niveau, mais dont la structure doit être dans un fichier facilement modifiable;
- le personnage doit être contrôlé par les touches directionnelles du clavier;
- les objets doivent réparti aléatoirement dans le labyrinthe à chaque démarrage du jeu;
- le labyrinthe doit être un carré de 15x15 cases;
- les objets sont récupérés en passant dessus;
- le programme s'arrête quand on est face au gardien. Si on a tous les objets on gagne, sinon on perd.

Attention : Le programme doit être 'standalone' et codé uniquement en anglais et dans le respect des bonnes pratiques de la PEP8.

Code source : https://github.com/vincenthouillon/macgyver-maze-game"

Arborescence du code

- **includes** *(dossier contenant les éléments nécessaires au jeu)*
  - **font** *(contient la police d'écriture et sa licence)*
    - `IndieFlower.ttf` *(police utilisée dans je jeu pour la liste des objets et les messages de fin du jeu)*
    - `OFF.txt` *(licence de police d'écriture)*
  - **img** *(dossier contenant les images utilisées dans le jeu (fournis avec le cahier des charges))*
    - `aiguille.png`, `ether.png`, `floor-tiles-20x20.png`, `Gardien.png`, `MacGyver.png`, `tube_plastique.png`, `etc.`
  - `classes.py` *(génération du labyrinthe à partir du fichier et placement aléatoire des objets à collecter)*
  - `constants.py` *(contient les paramètres du jeu)*
  - `labyrinth_scheme.txt` *(fichier texte définissant la structure du labyrinthe)*
  - `macgyver.py` *(gestion des déplacements de MacGyver)*
- `macgyver_maze_game.py` *(programme principal)*
- `README.md` *(fiche de présentation du jeu notamment pour Github)*
- `requirements.txt` *(pour installer les packages externes nécessaires au jeu)*
- `setup.py` *(script pour la création d'un exécutable pour Windows, voir `README.md`)*

Extrait du code

Dans `classes.py` // `class Maze` :

Utilisation de randint(random) pour définir les positions aléatoires des objets et vérification que les cases ne soient pas des murs ou un gardien :

```
    loot = 0
    while loot < len(OBJECTS):
        x_object = randrange(0, SPRITE_NUMBER)
        y_object = randrange(0, SPRITE_NUMBER)

        if maze_structure[y_object][x_object] == " ":
            maze_structure[y_object][x_object] = OBJECTS[loot]
            loot += 1
```

Vient ensuite, une méthode pour récupérer la case de départ de MacGyver dans le fichier texte `'labyrinth_scheme.txt'`, permet de configurer facilement la position de départ du labyrinthe :

```
def __get_position_mac(self):
"""Private method for get the macgyver position."""
    line_number = 0
    for line in self.structure:
        case_number = 0
        for sprite in line:
            pos_x = case_number
            pos_y = line_number
            if sprite == "s":
                self.macgyver_pos = (pos_x, pos_y)
            case_number += 1
                line_number += 1
```

Difficultés rencontrées

La bonne compréhension de gestion des différents affichages, événements, etc. de pygame.

Les échanges de données entre les méthodes, attributs et propriétés des différentes classes du programme.

Le choix de l'algorythme le plus simple pour le positionnement des objets, soit à partir d'une liste des cases disponibles, soit en vérifiant les coordonnées en sortie du `randint`.

Conclusion

Pour un débutant, c'est un exercice qui réclame que l'on ait une bonne vue d'ensemble de son projet. Qui demande de bien assimiler le fonctionnement de pygame, à savoir quand gérer les différents affichages, événements, etc. Et nous démontre bien la puissance et la complexité qu'est la Programmation Orientée Objets.