# Projet 3 Sauvez Mc Gyver (et le monde)

1) Introduction :

Pour le projet 3 de la formation Python d'openclassroom, il nous est demandé d'écrire une application ludique sous forme de labyrinthe. Mac Gyver doit traverser le labyrinthe en ramassant 3 objets placés aléatoirement, afin de les combiner en une arme redoutable lui permettant d'endormir le gardien.

2) Choix de l'algorithme :

J'ai choisi d'utiliser "random" afin de faire apparaître les objets lors de la création du labyrinthe, avec un retry si il s'agit d'une case "mur".
J'utilise donc les librairies "random" pour la génération des objets, pygame pour le graphisme, et time pour gérer le rafraîchissement de l'écran de fin.

3) Organisation de l'application et POO :

Suivant les conseils de mon mentor, j'ai commencé par faire le cours du labyrinthe de donkey kong, dont j'ai repris la structure pour ce projet : Un fichier "constants" rassemblant les constantes du fichier (taille du labyrinthe, emplacement des images, etc…), un fichier "classes" pour les fonctions que j'utiliserai ensuite, et un fichier "main" (mglaby.py) qui renferme l'application en tant que telle.
Il y a de plus un dossier image, et un fichier texte (map.txt) qui défini l'emplacement des murs.

4) Fonctionnement du programme

Tout d'abord, importer les librairies grâce à l'environnement virtuel (requirements.txt à importer avec virtualenv)
La class **Level** génère en se servant d'une liste les différentes tiles du labyrinthe (si la lettre correspondante n'est ni un vide ni celle servant à laisser de l'espace au sommet, c'est un mur) en se basant sur un fichier texte. Le fichier lit map.txt ligne par ligne et sprite par sprite avant de lui attribuer des coordonnées.
Le départ et le gardien (l'arrivée) sont fixes. Une génération des objets est effectuée à une place aléatoire de la grille. Je vérifie que la case n'est pas un mur, si c'est un mur, je recommence la génération (à voir pour plus tard : faire que deux objets ne puissent pas apparaître sur la même case)
Une fois que MC Gyver passe sur un objet, celui ci est changé de position et mit en haut, afin de servir de compteur. Une variable est passée en "True" pour chaque objet ramassé.
Une fois sur la case du Gardien, une vérification est effectuée. Si les trois variables sont passées en "True", affichage de l'écran de victoire. Sinon, affichage de l'écran de défaite.

La classe **Char** à principalement une fonction mooving (+1 aux coordonnées en cas de keypress) après vérification que la case suivante soit libre.

5) Problèmes rencontrés :

J’ai rencontré un certain nombre de problèmes lors des deux exercices (j’inclus les problèmes rencontrés lors du labyrinthe donkey kong qui sont une forme de “projet Alpha”)

- Nom des variables et modifications de mon code.

Je me suis aperçu que j’avais énormément de mal à rester consistant dans mes noms de variable. J’applique une nomenclature (par ex terminer mes noms se rapportant aux images par un _img), modifie mes noms de variable dans tous le projet, oublie un endroit, puis trouve une nouvelle nomenclature plus adaptée, mais oublie de changer la nomenclature d’une variable, etc. Ma solution a donc d’avoir toujours à côté de moi un cahier dans lequel je note mes nomenclature et m’y réfère à chaque fois.

- Format et taille d’image.

J’ai eu un message d’erreur à répétition du au format de mes images (obligation de passer en png-8).
J’ai aussi eu une erreur stupide : en récupérant les images, je n’ai pas vérifié qu’elles étaient de la taille de ma grille …. Cela me décalait tout tant que je ne les ai pas misent à la bonne échelle (j’ai cherché un bon moment dans le code quel était le soucis avant de comprendre que c’était externe).

- Pep8 .

Indent error et mauvais réglage d’atom qui m’a fait une erreur d’affichage sur le mglaby à cause d'espaces / tabulations.

- Écran de fin.

Le rafraîchissement des tiles affichaient les murs sur l’écran de victoire. J’ai donc importé une librairie time afin de mettre en pause mon écran de victoire ou de défaite pour une durée donnée avant de fermer le programme.

6) Améliorations possibles

- Module de changement de map
- Chronométrage du run avec Highscore