Rapport individuel Thibaut LABROUCHE 1H (projet "Zeldo")

Objectifs

L'objectif initial était de réaliser un jeu rpg avec une vue de dessus avec si possible quelques fonctionnalités en ligne pour rendre le jeu amusant à jouer en multijoueur. les inspirations se trouvent dans les jeux suivants : Minecraft pour le côté bac à sable, Zelda pour les mécaniques de gameplay et le style graphique.

Mise en place du projet

Tout le code du projet est stocké dans une repository sur github.com : https://github.com/Thithan77/Zeldo (repository privée)
J'ai choisi le language python en dépis de son manque de performance (voir annexe 1: performance) dans le but de rester dans la continuité du cours de l'année.
Après quelques recherches et à partir de ce que je connaissais déjà j'ai choisi la librairie pygame pour réaliser la partie graphique du projet car cela semble être la meilleure solution en python.
Je voulais réaliser le jeu le plus modulable possible avec des outils de développement simples pour pouvoir rajouter des *tiles* , des *items* le plus simplement possible et sans toucher au code. Par exemple lorsque j'ai rajouté l'objet des water babouches permettant de marcher sur l'eau il m'a suffit de créer le fichier de configuration `waterbabouche.yml` dans `content/items` :

```
name: "Water Babouche" # Le nom de la texture
fileName : "water_babouches.png" # Le fichier contenant la texture
effects: # La liste des effets de cet item
  - "water_walking" # marcher sur l'eau
textureonperso: "baboucheonperso.png" # le fichier contenant la texture à appliquer au personna
```

Le jeu est basé sur de la programmation orientée objet
Pour prendre en exemple le système de *Tile*
Un objet *Tile* contient à la fois des variables de classe et des variables d'instance:

Variables de classe

Les variables de classes ont pour objectiver de stocker les variables communes à tout les objets comme le nombre d'objets instanciés qui permet d'attribuer un nouvel id unique à la création d'un nouvel objet. La variable *nameToNumber* qui est un dictionnaire permet de trouver cet id à partir du nom de l'objet et le tableau *tiles* permet de faire l'inverse

Variables d'instance

Les variables d'instance sont propres à chaque objet *Tile*.

Mon rôle dans l'équipe

J'ai écrit la plupart du code présent dans le projet à l'exception du système de gestion des mobs et du système de combat qui ont été réalisé respectivement par Léo et Loïc.

Annexes

Annexe 1: Performance

Sur mon ordinateur personnel le jeu tourne fluidement à 60FPS (avec un temps de 6ms nécéssaire par frame cela laisse une marge). Néanmoins sur les ordinateurs moins puissants de la salle informatique, le framerate moyen est d'environ 40 FPS , ce qui a pour effet non seulement de rendre le jeu moins fluide à la vue , mais de plus par la gestion des évènements qui est faite sur chaque frame , rend le déplacement du personnage plus lent.
J'estime que les parties coutant le plus de performance comme étant:

- La gestion de la map par l'objet Map (ou multiMap) qui est constituée de multiples doubles boucles très lentes qui ralentissent considérablement l'éxécution.
- Les effets post-processing qui sont malheuresement impossible à réaliser avec une accélération GPU en python sauf en modifiant quasiment la totalité du code pour l'adapter à OpenGL et qui sont donc très lents sur le cpu. La plupart de ces effets sont les lumières (système jour/nuit et torches)
  En recherchant de la documentation sur internet j'ai trouvé une solution intéressante à ce problème: utiliser un autre language pour ces opérations répétitives. J'ai donc commencé à étudier rapidement le fonctionnement du module Cython. Après une petite heure de légère transformation des fichiers .py en fichiers .pyx ensuite compilés j'ai pu diviser par deux le temps nécéssaire pour l'affichage d'une frame en n'adaptant uniquement que l'objet Map. Néanmoins , cette compilation nécéssaire rendant le dévellopement plus difficile et n'en voyant que peu l'avantage j'ai choisi de ne pas intégrer cython dans la version finale du projet.

Annexe 2: Prolongement et futur du projet

Je n'ai pas l'intention d'abandonner le projet dans le futur et je compte continuer de travailler dessus autant que possible dans le futur dans le but de continuer à apprendre l'utilisation de python et de pygame. Si ce n'est pas le cas ce projet pourra toujours servir de base pour un autre dans le futur.

Annexe 3: Jouer à Zeldo

Télécharger les dépendances avec

```
python -m pip install -r requirements.txt
```

ajouter

```
python -m pip install tkinter
```

sur linux car il n'y est pas présent de base
Puis lancer le jeux

```
python main.py
```