Github: https://github.com/WyzzR/Search

Compte Rendu

L'idée du Projet

L'idée du projet n'a pas changé, nous avons bien un labyrinthe, des joueurs et des ennemis qui peuvent bouger selon les 4 points cardinaux.

Fonctionnalités Ajoutée

Dans notre cahier des charges, nous avons pensé à ajouter plusieurs fonctionnalités en plus du mouvement de base. Les fonctionnalités que nous avons mit en place sont :

- La téléportation, au lieu de faire téléporter le joueur d'un côté à l'autre de la carte, nous avons ajouté des portails.
- L'animation, nous avons utilisé des "sprites" pour toutes les entités. De plus, le dossier *src/img* contient pleins d'autres images donc c'est facile de personnaliser le jeu.
- Le combat, les joueurs et les ennemis ont des points de vie et peuvent se battre.
- Multijoueur, notre jeu est jouable par deux personnes sur le même clavier. Ils peuvent même s'entre-tuer.

Structuration du Code

Comme indiqué dans le cahier des charges, le jeu est divisé en plusieurs classes. La classe principale *Game* et celle pour la carte *Board*. Ces classes sont restées inchangées. Cependant, au lieu d'avoir une seule classe *Person* nous avons créé une classe mère *Entitiy* et les classes filles *Player, Enemy, Portal* et *Bullet* . Ceci nous a permis d'intégrer des fonctionnalités communes comme les "sprites" (animations) et à faciliter l'implémentation du mode multijoueur.

Contraintes Imposées sur le Projet

Concernant projet, il y avait certaines contraintes obligatoires et optionnelles que nous avons suivi :

- Nous avons deux algorithmes non trivial, une pour la génération du labyrinthe et l'autre pour calculer le chemin le plus court entre l'ennemi et le joueur.
- Nous utilisons la lecture et l'écriture de fichier pour pouvoir stocker et charger des cartes dans notre jeu, puis pour l'animation des entités.
- Un dictionnaire est employé pour stocker les différentes entités.
- Nous avons bien une interface graphique.

Auteurs : KUSNO Louis, SUN Jixiang, GUO Yizuo

Github: https://github.com/WyzzR/Search

- Plusieurs timers sont utilisés dans le code. Il y a un timer pour changer l'image de chaque entité (l'animation), il y en a un autre pour son déplacement, puis un dans le titre de l'onglet qui affiche le temps écoulé.
- Pour calculer le chemin le plus court, nous avons utilisé l'algorithm A*, un algorithme de graph pondéré avec une heuristique.
- Comme détaillé dans la partie *Structuration du Code* nous utilisons bien des classes.

Organisation

L'utilisation de GitHub a permis à tous les membres du groupe de travailler sur une partie du code et à faciliter l'implémentation de différentes fonctionnalités. Nous avons suivi comme règle de seulement pouvoir commit le code sur GitHub s'il marche sans bug. Ceci nous a permis d'avoir un code fonctionnel à tout moment du développement pour pouvoir ajouter des fonctionnalités.

Conclusion

Tout compte fait, il n'y a pas beaucoup de différences entre le cahier des charges et notre projet final. En effet, nous étions déjà bien avancé en terme de code quand nous avons écrit le cahier des charges. Par la suite, il n'y avait pas trop de déviation des idées principales.