Camara Nelson

Kaouane Issam

Manuel d’architecture du projet Java TheBigAdventure

Sommaire :

Introduction :

L’objectif de ce projet est de réaliser un meta-jeu se rapprochant de Zelda.

Nous devions donc implémenter des personnages,des objets et des environnement.

Une grosse partie du projet consiste également à pouvoir créer sa propre carte.

Architecture :

Afin de réaliser se projet en respectant les principes de la programmation objet.

Une grosse partie du temps consacré à celui-ci a été investi dans la conception de l'architecture.

L'objectif était de concevoir une structure respectant au mieux les principes SOLID ce qui fut très difficile car cela fait quelques mois seulement que nous apprenons la programmation objet.

Le projet est divisé en plusieurs packages :

-***characterEntities*** pour gérer les entités.

-***graphic*** pour gérer l'affichage sur la fenêtre.

-***ground entities*** pour les entités au sol type décoration Obstacles etc.

-*Main*

-***mapBuilder*** avec toutes les classes de création et gestion de la grille.

-***usableEntities*** avec tous les objets activables.

Les classes principales :

GameMap gère le plateau de jeu,et toutes les entitiés présentes dessus.

Case représente une unité de la grille et peut stocker plusieurs entités comme :

Environnement (Interface) Obligatoirement présent pour créer une case

cette entité représente le sol et les décorations de la carte

Character (Interface) en optionnel représentant un personnage ennemi,joueur ou allié (non implémenté par manque de temps)

InsideInventory(Interface) Toutes les entités stockables dans un inventaire et pouvant être ramassées par le joueur.

Player cette classe gère tous les mécanismes liés au joueur.

Inventory gère le concept d'inventaire et les opérations réalisables dans celui-ci.

Graph contenant toutes les méthodes graphiques.

Interfaces :

Character une interface scellé permettant uniquement les types Enemy,Ally,Player

Environnement une interface représentant les décorations,biomes,obstacles de la carte.

InsideInventory une interface représentant tous les items stockables dans un inventaire.

J'ai décidé de ne sceller que character car il ne peut y avoir qu'un type ennemi, un allié et un joueur dans ce type de jeu donc par la suite pas besoin de rajouter un nouveau type de personnage.

Or ce n'est pas le cas pour Environnement et InsideInventory peuvent contenir une grande variété de types.(En effet j'aurais pu les sceller pour ce rendu mais j'ai préféré laisser comme ça).

Choix d'implantations et difficultés rencontrées :

La grille est une MAP d'un record Position en clé et d'une classe Case.

Cette grille est contenue dans la classe GameMap.

En effet cela me permet d'accéder à tous types d'entitiés sur chaque position de la grille car une Case peut contenir n'importe quel entité.

C'est la Case qui gère si elle est un obstacle ou non en fonction des entités qu'elle contient cette gestion me permet de modifier simplement l'état d'une case et l'action possible avec celle-ci.

L'action possible sur une Case est stockée dans ActionType un enum gérant les actions comme une case contenant un Enemy,un Item,une DOOR etc.

Les types Enemy et Player ne savent pas qu'ils se déplacent en effet le déplacement est géré par la même méthode dans GameMap MoveMovableEntity qui déplace les types Character.

Ainsi après avoir crée le type Player et l'avoir fait se déplacer,il m'a été rapide d'implémenter de déplacement d'un Enemy car la même méthode gère Character.

Lors du déplacement d'un Enemy ou d'un Player la Position de la tuile devant lui en fonction de sa direction est calculable dans une méthode de Character.

Ce choix est pratique pour savoir si par exemple le Player se situe devant une porte ou un Enemy.

Les grandes difficultés ont été la partie graphique,en effet j'ai passé beaucoup de temps dans l'affichage rien que des points de vie par exemple à cause des différentes échelles du AffineTransform que j'ai implémenté pour mettre l'affichage zoomé sur le joueur.

Il peut y avoir quelques bugs d'affichage notamment si vous allez dans les coins.

La partie graphique a réellement été un facteur limitant pour moi dans l'avancement de ce projet.

La structure que j'avais mis plusieurs jours à définir me permettait finalement d'implanter les nouveau mécanismes demandés dans le phases de développement plus rapidement que prévu,mais le graphique m'empêchait de me focaliser sur ce développement.

Des mécanismes comme transformer un TREE en BOX,enflammer une épée,prendre des moyens de transports auraient largement pu être implantés sans nécessiter beaucoup de code j'ai d'ailleurs grandement apprécié cette philosophie du langage (ça change du C !).

Le Lexer a également été une source de difficultés.

Améliorations depuis la soutenance :

Vous nous aviez dit de recommencer notre Lexer cela a été fait avec la validation des .map également.

Les packages ont été organisés par sémantiques comme demandé.

Le joueur peut manger une pizza récupérer une épée se battre avec un Monster

Le joueur peut récupérer une clé et ouvrir une porte.

Le joueur possède un inventaire et peut l'organiser et choisir quel item mettre en main.

Affichage centré sur le joueur.

Répartition du travail :

Une autre difficulté a été crée par la répartition du travail.

Issam ,bien que motivé à travailler sur le projet bloquais sur plusieurs points de la validations des map qui lui a pris beaucoup de temps.

J'ai donc dû m'occuper du développement de tous le reste c'est-à-dire Lexer,passage du fichier map en objects (après validation) ainsi que tous les autres mécanismes du projet.

En effet tous les .java hors MapValidate ont étés développés par de moi-même ce qui a été également une grande limitation dans l'avancement de notre projet.