BIDAULT Matthieu
GAUTHIER Julien

TP1B

Rapport de Projet

POA

Sommaire :

Table des images :

Présentation

Ce projet avait pour but de coder un jeu de chasse au trésor en java avec l'IDE eclipse.

Le Jeu :

Le but du jeu est de déplacer plusieurs personnages sur une grille, tous à la recherche du trésor qui se trouve dans une case de la grille inconnue des personnages. La grille est délimitée par des bords, et comprend également des murs intérieurs qui font obstacle aux déplacements des personnages, ainsi que des cases spéciales pouvant ralentir les personnages, ou leur fournir les outils pour franchir les murs sans les contourner.

Les personnages :

- Les Hunters : ce sont les chasseurs de trésor (les personnages jouables). Ils sont les seuls personnages pouvant interagir avec les obstacles et les outils placés sur la grille
- Les WiseMan : ce sont les sages. Ces personnages permettent de rediriger les Hunters vers le trésor
- Les Cheater : ce sont les “ennemis”. Leur but est de donner une mauvaise direction aux Hunters qui croisent leur chemin.

Les obstacles :

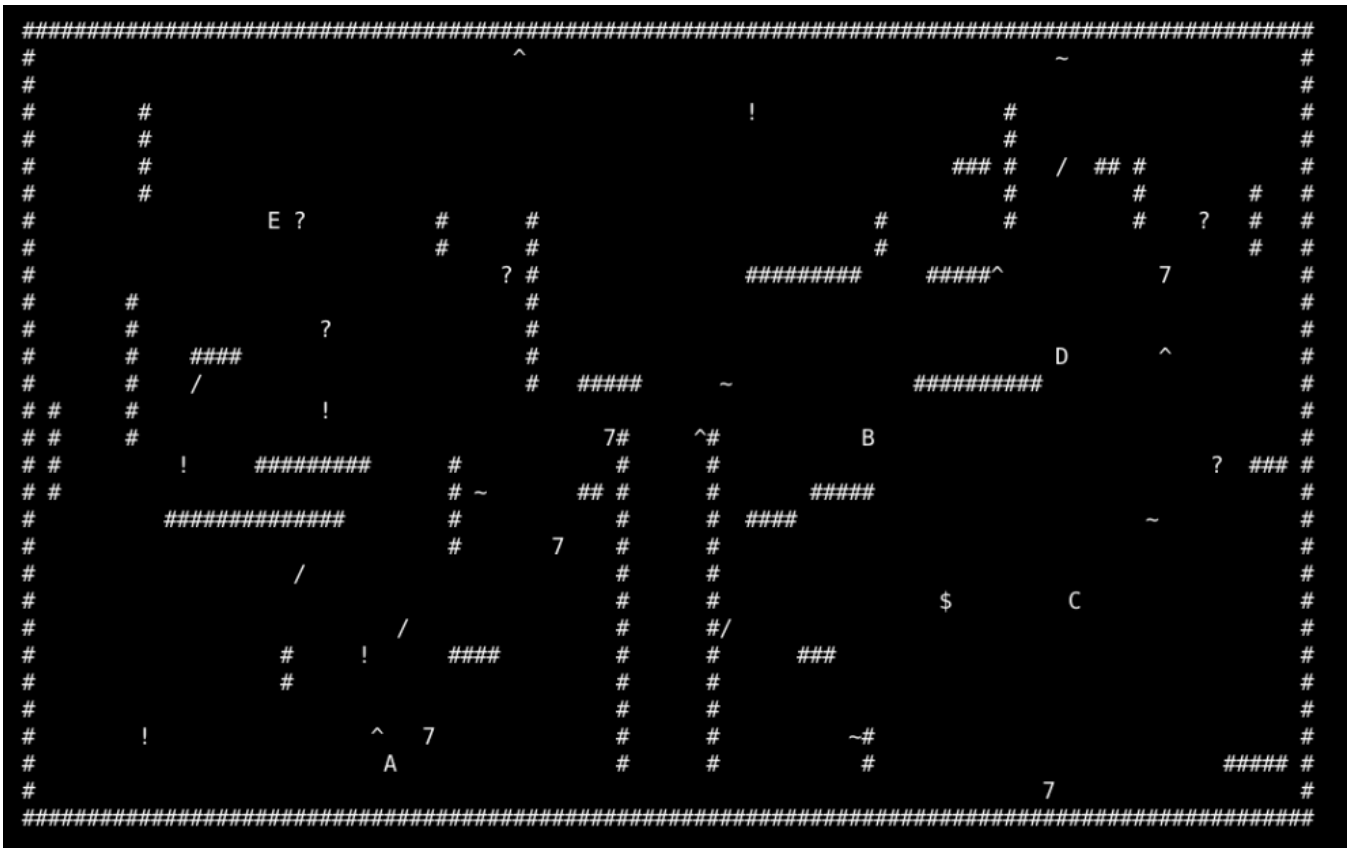
- Les murs : Ils sont l'obstacle principal des Hunter, car ils bloquent leur progression en les obligeant à les contourner s'ils n'ont pas d'outils.
- Les flaques de glu : elles agissent uniquement sur le hunter qui tombe dessus, elle le bloque pendant 2 tours.

Les outils :

- L'échelle : elle permet de grimper en haut d'un mur.
- La Pioche : elle permet de casser une case d'un mur pour pouvoir le traverser.
- La carte au trésor : elle indique la direction à suivre pour s'approcher su trésor.

Pour la version console, on a utilisé les symboles suivant pour afficher le jeu :

- Hunter : les lettres de l'alphabet en majuscule
- WiseMan : ^
- Cheater : “
- Les murs : #
- La glu : ~
- Les échelles : /
- Les pioches : 7
- La carte au trésor : ?
- Le trésor : $



1 - Version_console

Pour la version graphique :

2 - Les_assets

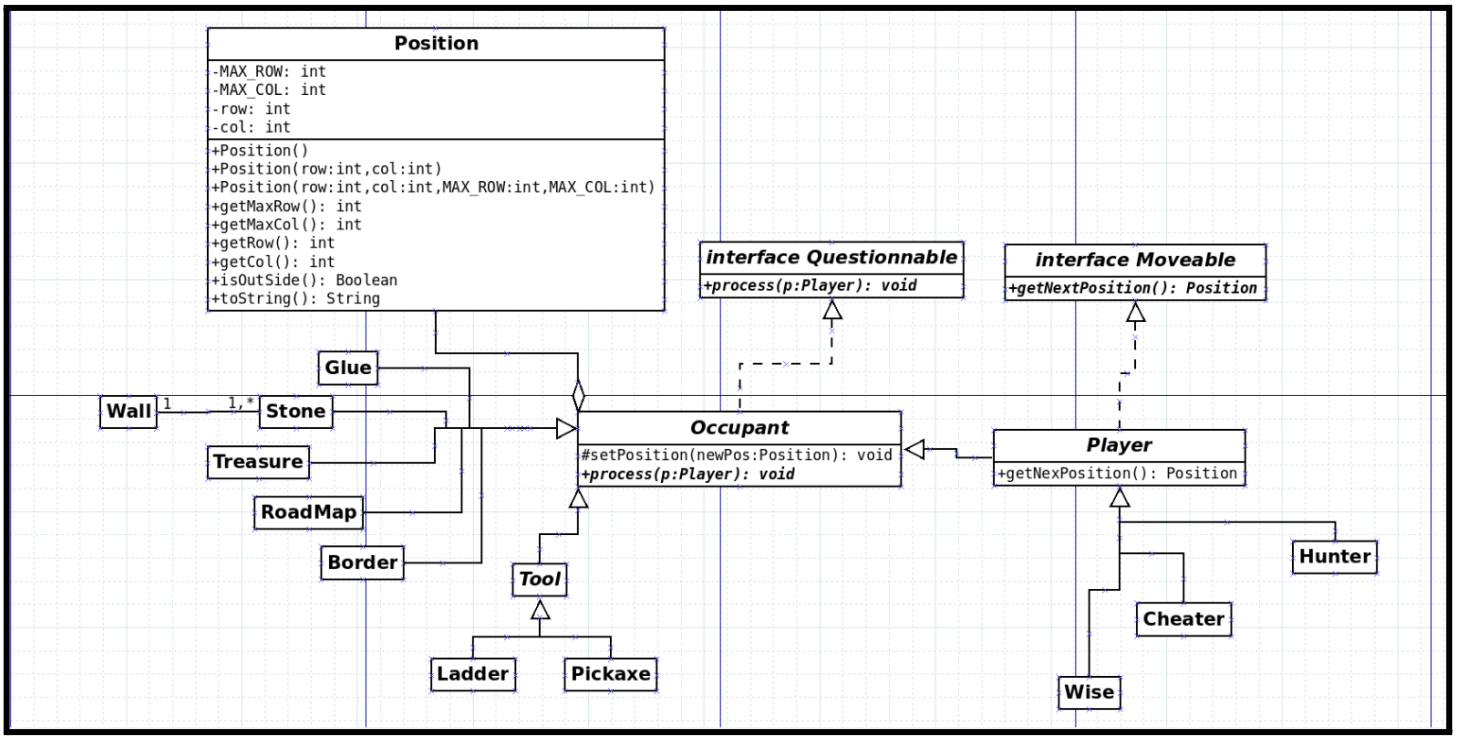
l'image est à lire de gauche à droite en partant du haut :

- hunter yellow
- hunter red
- hunter grey
- hunter blue
- wiseman
- cheater
- pickaxe
- ladder
- chest
- roadmap
- grass
- glu
- stone

On a choisi de se limiter à 4 variations de hunter et une variation de wiseman et de cheater par manque de temps. On aurait aimé créer un asset spécifique quand le hunter est en possession d'une pioche, mais ça aurait doublé le nombre d'asset de hunter.

Conception de l'application

Pour concevoir l'application, nous avons d'abord réalisé un diagramme de classe :



3 - Diagramme_UML__conception_de_l_application

Choix de conception

Dans les choix intéressant de conception, il y a :

- La génération aléatoire des murs : Pour créer un mur, on prend aléatoirement une case de départ vide avec ses voisins vides aussi. On choisit ensuite si le mur sera vertical ou horizontal. Tant que l'on peut placer une pierre à la position suivante, on a 80% de chance de placer une nouvelle pierre, sinon on s'arrête de construire le mur.
- La redirection des hunters : Pour pouvoir au mieux rediriger les hunters, nous avons décidé de calculer les angles entre le hunter et le trésor pour pouvoir au mieux le rediriger. Simplement choisir de le diriger vers le trésor par des différences entre l'axe x et y aurait parfois amené à un éloignement.

avec différenciel axe X et axe Y:
H(1,1) -> T(7,2)
7-1 = 6
2- 1 = 1
6 > 0 et 1 > 0
donc la direction est vers le nord-est

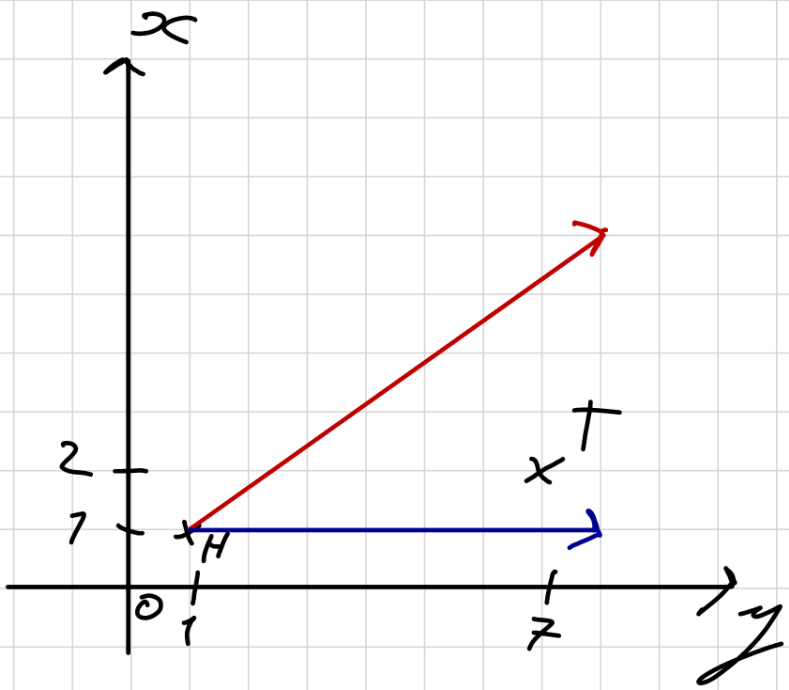
avec un calcule d'angle grace aux vecteurs:
v1 = (1,0) // axe X
v2 = (6,1) // Vecteur HT
angle = ARCCOS((v1.v2) / (||v1||.||v2||))

v1.v2 = 1x6 + 0x1 = 6

||v1||.||v2|| = sqrt(1² + 0²) x sqrt(6² + 1²)
||v1||.||v2|| = 1 x sqrt(37)

angle = arccos(6 / sqrt(37)) ~ 0,1651486774

0 < 0,1651486774 < pi/2
donc la direction est vers l'est



4 - calcule_meilleur_direction

Développement de l'application

Répartition des taches :

Matthieu :

- les interfaces et la classe position
- la classe Wall, Stone, glue et border
- la classe Occupant et player
- La classe treasur
- Une partie des classes hunter, cheater et wiseman
- les classes Jeu (version console et graphique)

Julien :

- La classe roadMap
- La classe Tools et ses enfants (Ladder et Pickaxe)
- une partie des classes hunter, cheater et wiseman
- choix des symboles de la version console
- tous les assets (les images PNG en pixel-art) de la version graphique

Outils techniques utilisés :

Nous avons utilisé GitHub pour pouvoir gérer notre avancement plus facilement et nous avons préféré utiliser l'IDE IntelliJ plutôt que eclipse pour son ergonomie plus facile à prendre en main et pour la gestion des plugins intégrée (ceux de GitHub par exemple)

Nous avons aussi utilisé eclipse pour vérifier que le projet fonctionnait bien dessus.

Pour générer les assets en pixel art, nous avons utilisé le logiciel piskel et des asset gratuit et libre de droit trouvé sur itch.io

Conclusion :

Ce projet nous aura permis de vraiment exploiter chaque aspect de la programmation orienté objet et ce qui en ressort est qu'il est très intéressant et important de séparer les classes en sous classe et d'implanter des interfaces et des classes abstraites.

L'implantation de classe abstraite et d'interface nous a fait gagner énormément de temps, car quand l'un d'entre nous faisait une interface, il était facile pour l'autre de reprendre ce code pour créer les ce qui a facilité la mise en commun du travail.

L'un des soucis que nous avons eu avec ce projet est que nous avons sous-estimé le temps nécessaire pour la version graphique. Les assets ont été long à faire et la limitation technique de l'interface graphique ne nous à pas permit de charger les asset par couche ce qui nous a contraint à faire une image par case et par cas particulier ce qui a multiplié le nombre de PNG à faire. Avec plus de temps, nous aurions pu faire d'autre asset pour, par exemple, représenter un hunter avec une pioche.