Dossier du Projet Revenge of the Student

Conception d'un jeu vidéo a l'aide de JAVAFX

Alloune Ayoub - Houssian David - Mehadji Sofiane / TPA1

- I. Architecture (page1)
- II. Diagrammes de classe (page2)
- III. Diagrammes de séquence (page3)
- IV. Structure de données dans le fichier csv et dans sa lecture (page4)
- V. Algorithme (page5)

I. Architecture

Nous avons gardé les choses simples pour l'organisation de nos classes, nous avons opter pour une architecture MVC. Lorsque l'on considérait qu'une classe été assez importante pour lui créer un répertoire a part on le faisait. Lorsque des classes dépendaient d'autre classes on les mettait toute dans le même sous répertoire. Pour le répertoire ressources il y a des sous répertoires pour les différent sprite, icone d'arme pour l'inventaire et le csv.

```
Hero.ucls
application
               application.css
Main.java
Sample.fxml
               SampleController.java
 Controleur
               Controleur.java
               ControleurDeplacement.java
OberservableDirection.java
               OberservablePersonnage.java
                ObservableEnnemie.java
               ObservableInventaire.java
ObservableTerrain.java
             —GameLoop
BoucleDeJeu.java
               ChercheImage.java
               Collision.java
Consommable.java
               Epee.java
Fleche.java
Inventaire.java
              The law and the law and law an
               -Accessoire
                                Accessoire.java
                               EcharpeHomps.java
                              m
Arme.java
Block.java
Epee.java
Item.java
Katana.java
Outil.java
              -Movable
                               GraviterMovable.java
                              Moove.java
Movable.java
                              Ennemie.java
                               Homps.java
                              Personnages.java
Simonot.java
                             Hero
                                               Hero.java
                             -Item
                                               Arme.java
                                               Rety.java
               coeur.png
Grassland_Terrain_Tileset.png
              main.csv
MehtodeUtil.java
               outdoors.png
              test_extra.csv
test_main.csv
test_test.csv
vide.png
              -Armes
New Piskel.png
             -ImagePerso
archerDroite.png
                                archerGauche.png
                              epeeDroite.png
epeeGauche.png
flecheDroite.png
flecheGauche.png
                               HompsDroite.png
                               HompsGauche.png
                               Katana.png
katanaDroite.png
                              katanaGauche.png
NewPiskel(1).png
                              NewPiskel.png
SimonotDroite.png
SimonotGauche.png
               ChargeurDImage.java
               Vue.java
VueEnnemie.java
               VueFleche.java
VueInventaire.java
                vueJeu.fxml
               vuePersonnage.java
               VueStat.java
VueTerrain.java
```

II. Diagramme de classe

Diagramme de classe représentant l'architecture de la classe abstraite Movable.

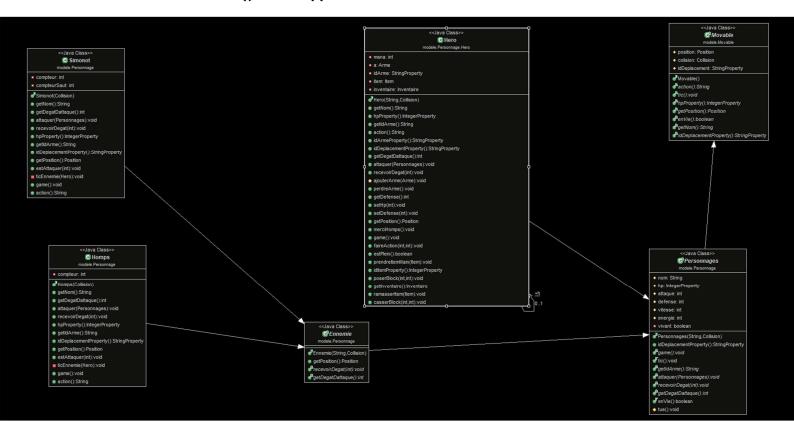
On a choisi cette architecture pour regrouper toutes les entités du jeux (à savoir les flèches, les ennemis et le héros) tout en permettant de les rendre bien distinct l'un de l'autre. Movable étant la super classe abstraite elle contient une méthode "tic" qui corresponds à toutes les actions que les movables effectuerons pendant le tour.

La difficulté de cette méthode est qu'il fallait un moyen de savoir si le movable était mort et donc ne devait plus faire d'action (que ce soit une flèche ou une ennemie morte), Voilà pourquoi une méthode abstract Game à était rajouté à la super Classe Personnage qui extende de Movable, ainsi qu'un attribut booléen vivant.

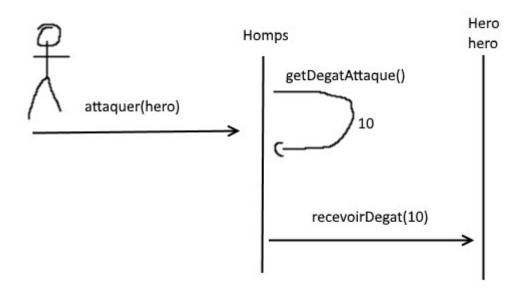
Ainsi dans la méthode tic est activé si vivant est à true.

Il y a la méthode ticEnnemie() qui affecte les mouvement et action spécifique à chaque ennemie, détermine quand il attaque est quand il est tué(attribut vivant qui est set à false).

Enfin ticEnnemie() est appelé dans la méthode Game.



III. Diagramme de séquence



IV. Structure de données dans le fichier csv et dans sa lecture On retrouve également une partie algortithmique dans notre projet qui est la lecture de la map ainsi que du tileset.

Le tileset de la map est formé d'une manière assez specifique elle ne dépasse pas les 10 blocs de haut mais a une longueur infinie. On a choisi cette forme pour que chaque blocs ai comme nom dans le csv ses coordonnées sur le tileset. Ainsi on récupère le x et le y grave a une division et un modulo et on affiche le bon block correspondant dans le tileset