L3 informatique
Hugo Robert, Arthur Haye, Alexis Bottin

Rapport projet diablotin

Sommaire :

- **Réalisation du projet**
- **Présentation du projet**
- **Diagramme de classes**
- **Interface Homme Machine**
- **Modèle BD**
- **Lancer/jouer au jeu**

Réalisation du projet :

Pour réaliser ce projet, nous avons décidé de réaliser tout le code en java, cela nous paraissait le plus simple pour concilier la partie orienté objet l'IHM.
Nous avons réfléchi à quoi ressemblerait ce projet et la conception de celui-ci, à la partie orientée objet et l'IHM ainsi qu'à la base de données. Nous avons réalisé les différents schéma et modèles permettant de réaliser ce projet ainsi que beaucoup de code, cela nous a permis de développer ce projet au mieux.
Nous avons réalisé ce projet en essayant de nous rapprocher le plus possible d'un jeu Diablo dans la mesure du possible.
Ce que nous avons réalisé a été conforme à nos attentes, nous avons développé la conception du jeu pour qu'il ait beaucoup d'interface et de fonctionnalités, ainsi nous avons une base solide pour développer celui-ci.
Vous trouverez ci-dessous ce que nous avons réalisés pour la conception du jeu, cela comprend plusieurs schémas et idées qui nous ont permis d'arriver à développer le code du projet que nous avons réalisé.
Cela comprend un diagramme de classes assez complexe que nous avons réussi à améliorer au fil du temps, comprenant de nombreuses classes.
La partie IHM de ce rapport sera composé de captures d'écrans de l'interface que nous avons codés étant donné que celle-ci correspond à ce que voulions réaliser.

Présentation du projet :

Dans notre projet, on incarne Lucifer, celui-ci peut se trouve sur une carte de 10x10 cases, le héros peut se déplacer d'une case à une autre dans 8 directions différentes grâce à des boutons de déplacements.
Sur la carte se trouve le décor, cela peut être simplement le sol sur lequel le héros marche mais également des obstacles.
Il y a également des items qui peuvent être trouvés au sol qui peuvent être ramassés par le héros et qui seront placés dans un inventaire qui sera ouvrable par le joueur avec un bouton, l'inventaire affichera la liste des items détenus par le héros et les actions que l'on peut faire avec(l'utiliser ou le jeter).
Les unités autre que le héros sont les ennemis, ceux-ci viendront attaquer le héros et lui faire baisser sa vie, si le héros veut vaincre les ennemis, il doit se mettre à portée d'eux ou utiliser des compétences qui coûteront du mana au héros, ces compétences seront sous forme de boutons.
Il y a plusieurs types d'ennemis avec chacun des statistiques différentes.

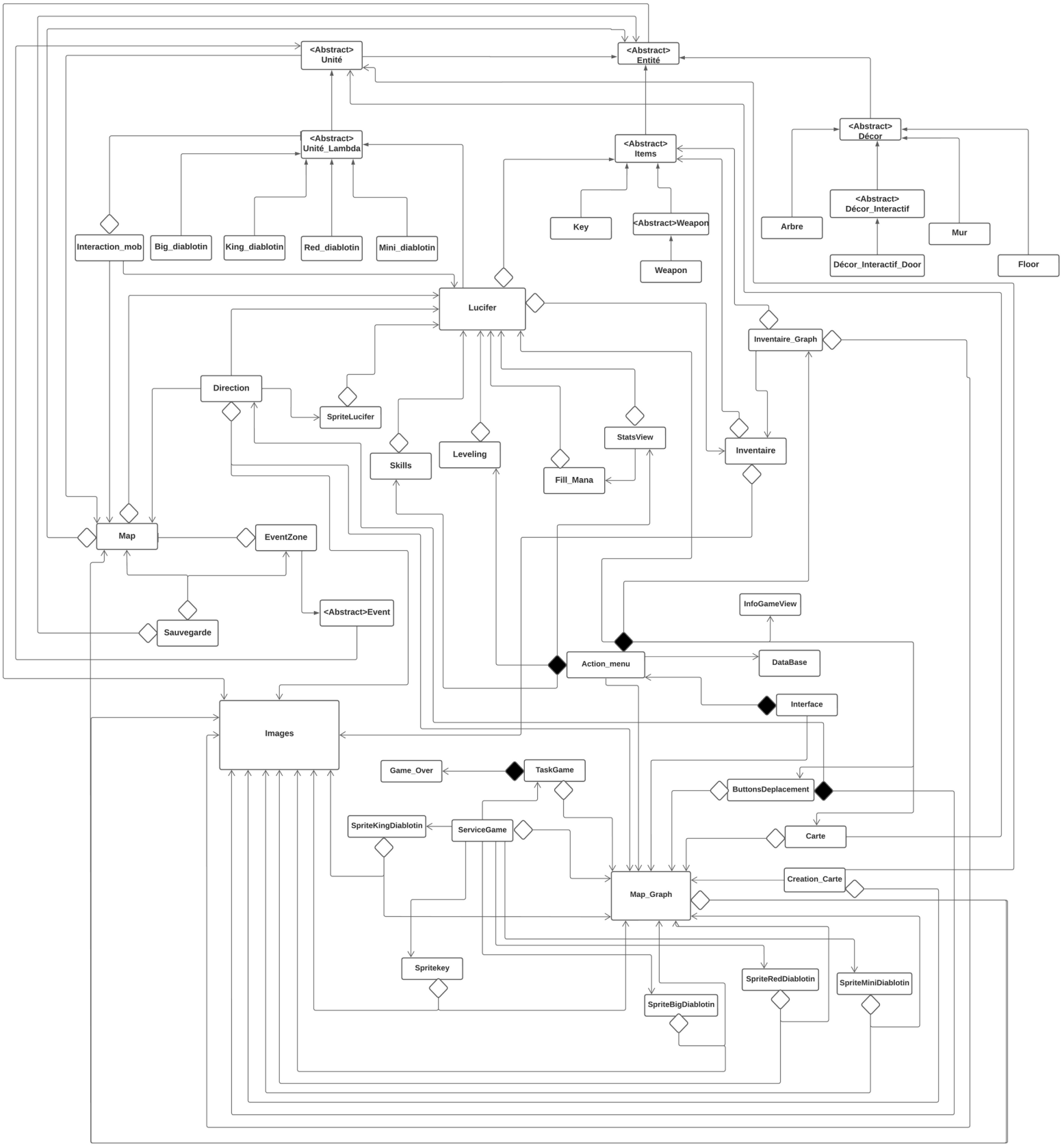
Le héros possède un système de niveau, si il tue des ennemis, sa barre d'expérience montera si elle atteint le montant maximum, elle se reset et le héros gagne un niveau. La barre d'expérience et le niveau du héros sont affichés à coté de la carte, certaines compétences du héros se débloqueront en fonction du niveau de celui-ci, il gagnera également des statistiques en montant de niveau.

Les statistiques du héros seront affichés dans l'interface, le héros a des points de vie, de mana, une portée, une vitesse et des dégâts. Le mana du héros se remplit progressivement avec le temps.
Il y a également un menu dans l'interface qui répertorie toutes les actions, par exemple si le héros ramasse une clé, un message apparaîtra dans ce menu, il permet également de suivre les actions pendant un combat, quand un ennemi attaque le héros et inversement, des messages apparaîtront et indiqueront les dégâts infligés et les PV restant des unités qui combattent.

Le joueur peut ouvrir une carte pour avoir une vue plus en profondeur sur ce qui entoure le héros, il peut l'ouvrir grâce à un bouton sur l'interface. Il y a aussi un bouton qui permet au joueur de sauvegarder sa partie pour pouvoir la reprendre plus tard.
Chacune des images du jeu sont dans une classe Images qui possède toutes les images du jeu, chacune des classes nécessitant ces images pourront y accéder et ainsi constituer l'aspect graphique du jeu.

Diagramme de classes :

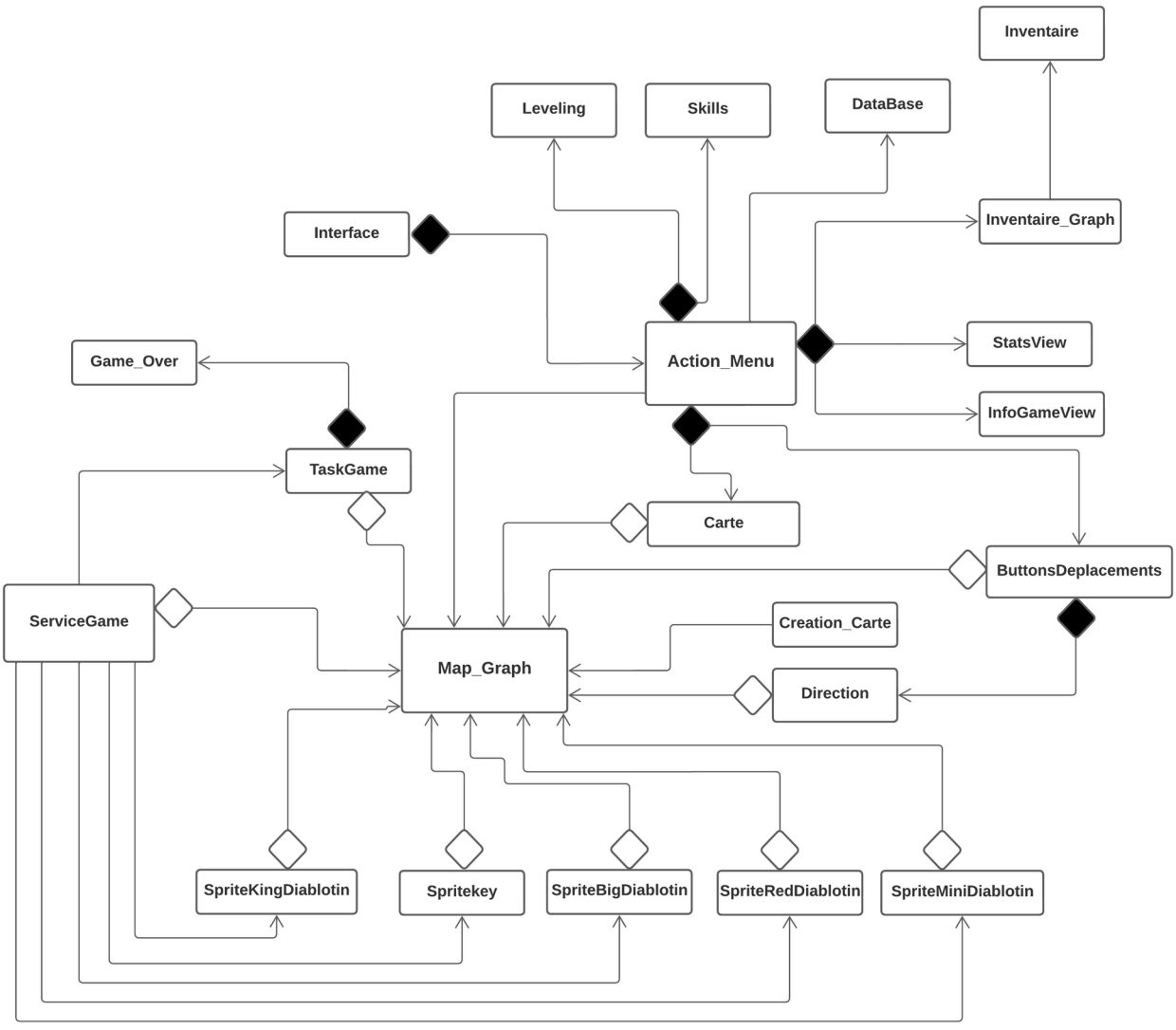
Diagramme du projet complet :



Schémas simplifiés :
(il n’y a pas toutes les classes mais les meilleurs pour une compréhension globales)

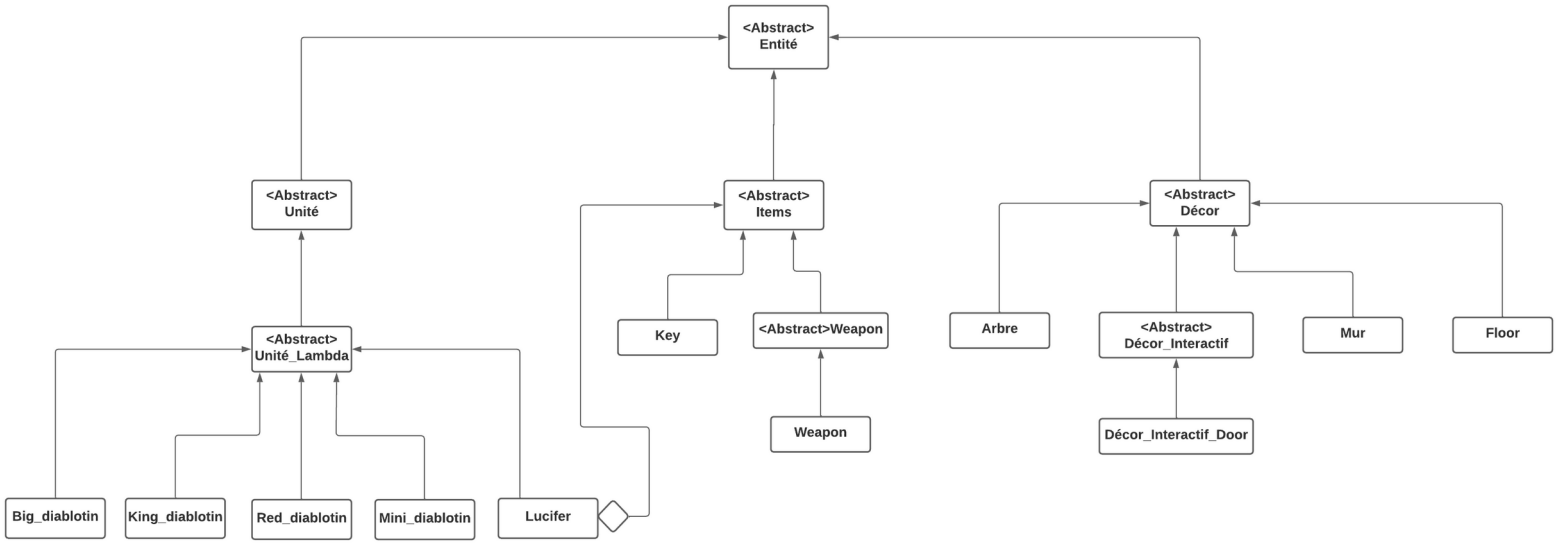
Principales classes de l’interface du jeu :

Map_graph correspond a la carte du jeu, Action_Menu représente l’interface en jeu avec tous ses éléments comme les menus de stats, les skills, la carte, etc.
Interface utilise Action_Menu qui sera ensuite utilisé dans le Main pour lancer le projet.



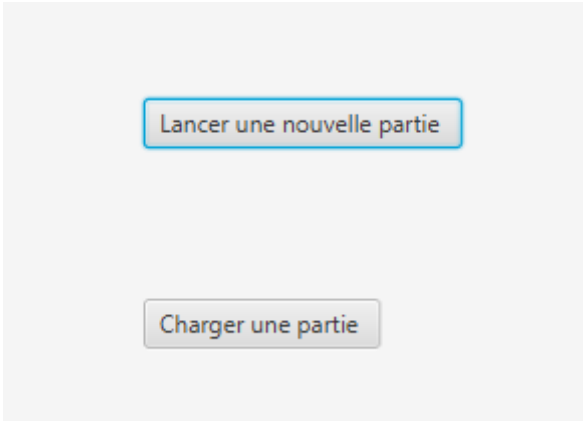
Hiérarchie de la classe entité :

La classe entité représente toutes les entités qui seront présentes dans le jeu, avec les unités, que ce soit le héros ou les ennemis, les items qui seront sur la map ou dans l'inventaire du héros et les décors qui composeront la map, dont ceux avec lesquels on pourra interagir comme les portes.

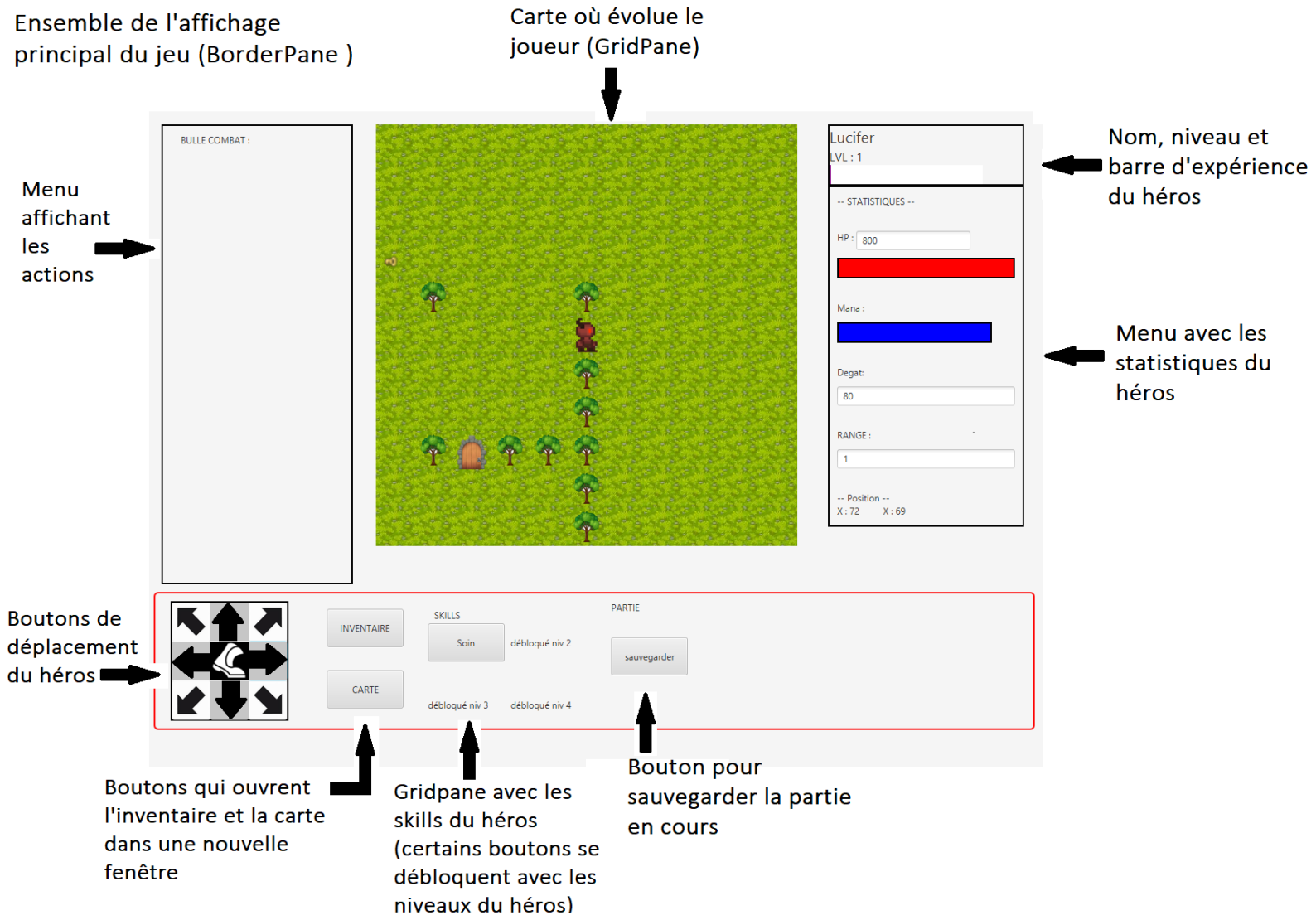


Interface homme-machine :

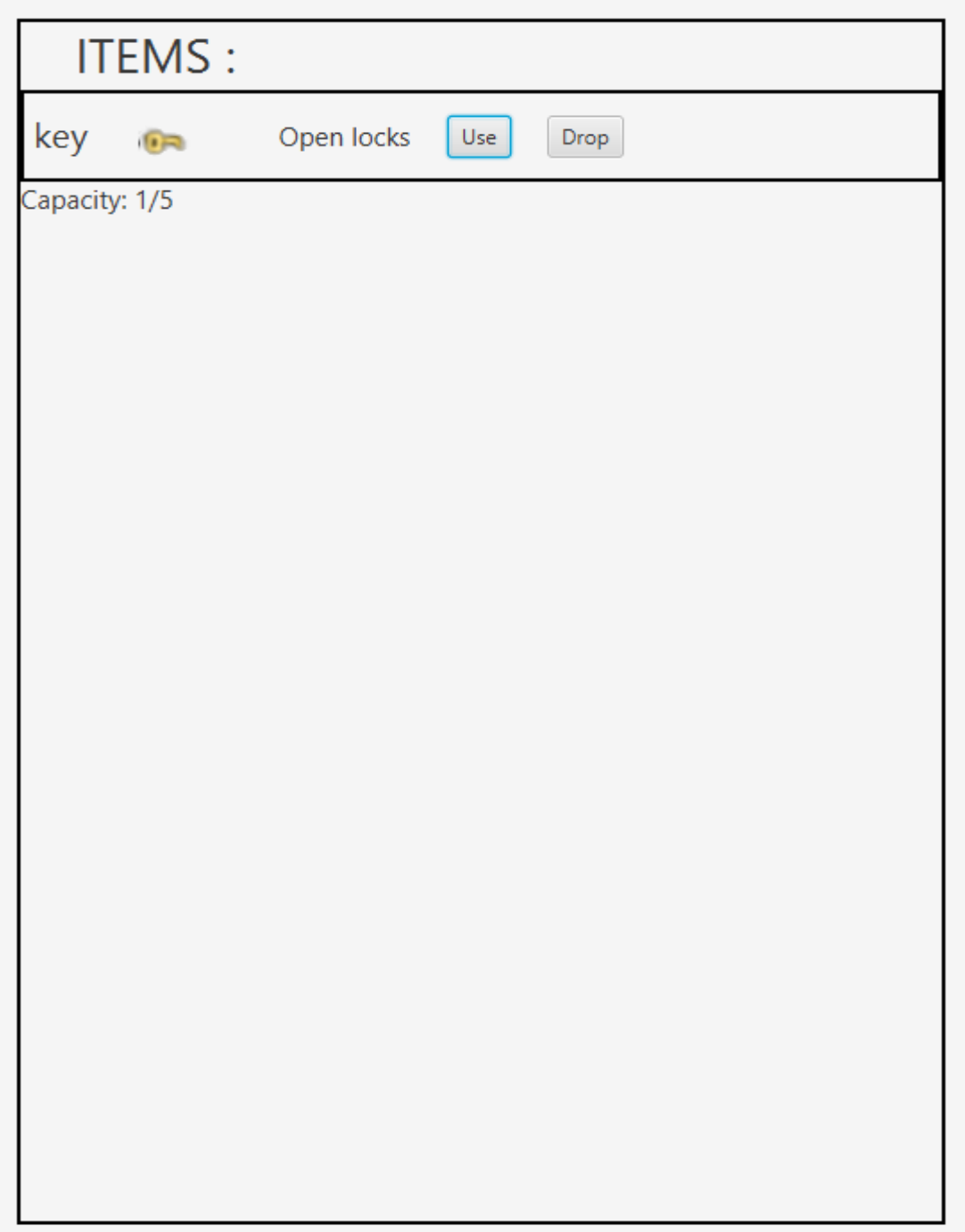
Boutons permettant soit de lancer une nouvelle partie soit d'en charger une que l'on a préalablement sauvegardée :



Aperçu de l'affichage principal :

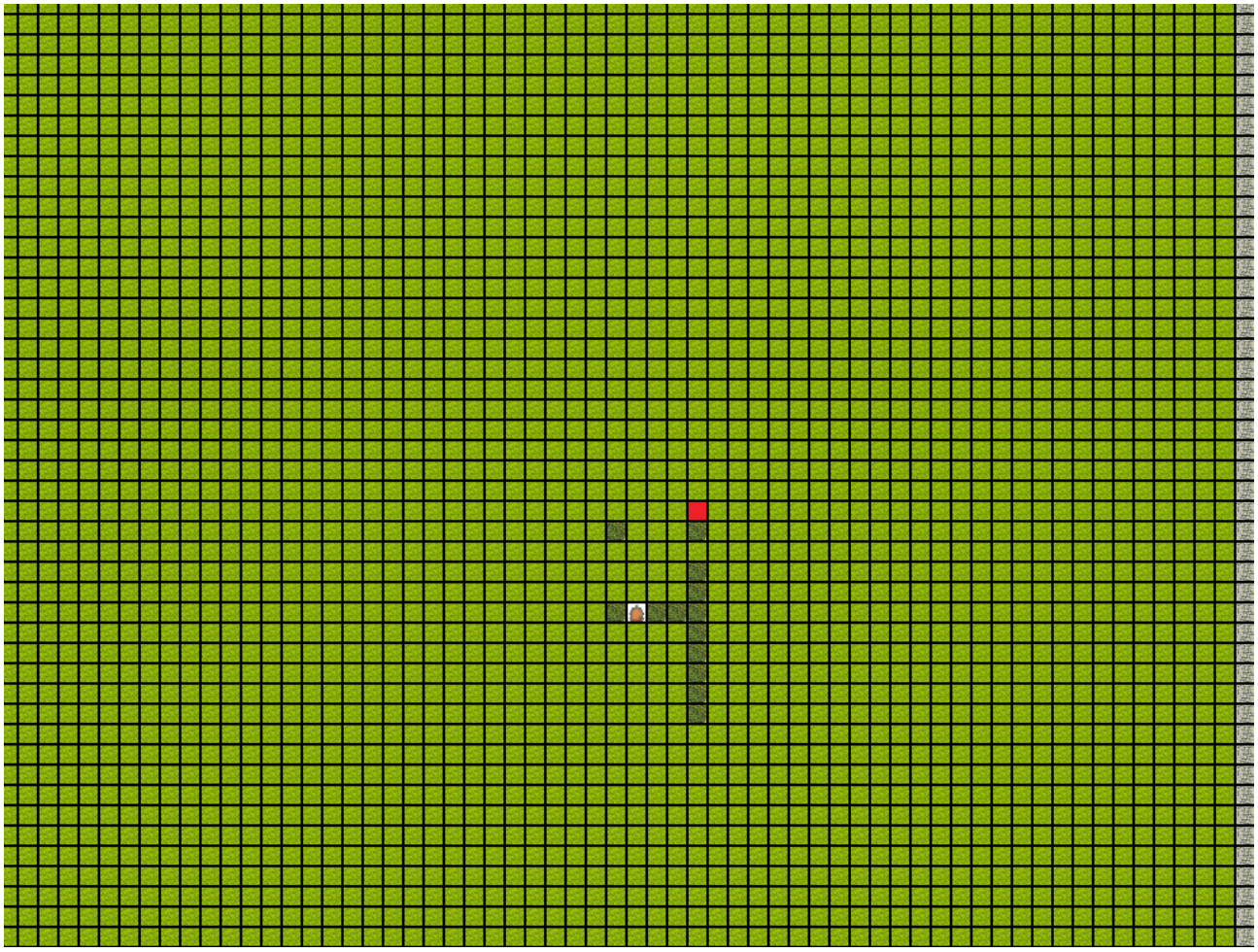


Inventaire ouvert avec le bouton inventaire :



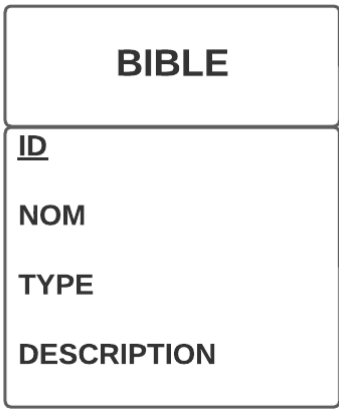
L’inventaire est un BorderPane avec « Items » en haut et le stockage en bas, le milieu est composé des objets, chaque objet sera représenté par son nom suivi de son image, de sa description puis des boutons Use et Drop (comme ici avec la clé).
Le bouton Use sert à utiliser un item et le bouton Drop à en lâcher un, il y a un maximum de 5 items dans cet inventaire. L’inventaire se remplira des objets ramassés par le héros.

Aperçu de la map obtenu en appuyant sur le bouton Carte :



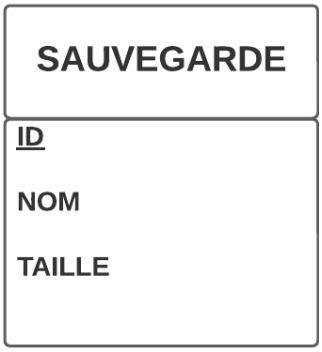
La carte est un GridPane représentant la matrice du jeu avec tous ses éléments, on reprend la carte de l’affichage principal et on l’affiche en montrant plus de cases. Cette carte est centrée sur le héros et affiche tous ce qu’il y a autour de lui mais on peut faire défiler la carte avec la souris si l’on veut voir plus loin.

Modèle BD :



BIBLE :

La colonne ID (INTEGER) est la clé primaire de la classe BIBLE.
La colonne Nom (STRING) correspond au nom de l'item Bible.
La colonne Type (STRING) correspond au type d'objet.
La colonne Description (STRING) correspond à la description de l'objet.



SAUVEGARDE :

La colonne ID (INTEGER) est la clé primaire de la classe SAUVEGARDE.
La colonne nom (STRING) correspond au nom de la sauvegarde.
La colonne taille (STRING) correspond à la taille du fichier de sauvegarde.

Lancer/Jouer au jeu :

Pour jouer au jeu, lancer la compilation du projet, on se retrouve devant un menu, en appuyant sur lancer une nouvelle partie, vous pourrez jouer au jeu, il suffit de cliquer sur les flèches directionnelles pour déplacer le héros, vous pouvez ainsi ramasser des objets, tuer des ennemis en se mettant à portée d'eux et progresser dans le niveau.
Le héros peut utiliser des skills pour aider la progression.

Le joueur peut ouvrir son inventaire (bouton inventaire) pour consulter les objets qu'il a en sa possession, il peut également ouvrir la carte (bouton carte) pour voir où se situe le héros et ce qu'il y a aux alentours. Il peut également sauvegarder la partie en cours pour pouvoir la reprendre plus tard, pour ce faire il doit appuyer sur « charger une partie » au moment de la compilation du projet.

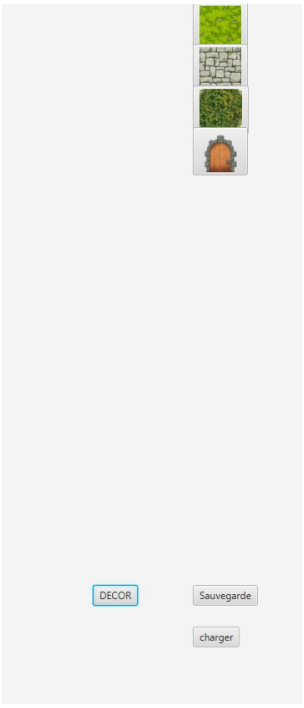
Si il y a des problèmes de compilation du projet, cela peut être lié à la version de la JDK, pour régler ce problème, il suffit de changer la version de la JDK dans les propriétés du projet.

Lancer Creation_Carte :

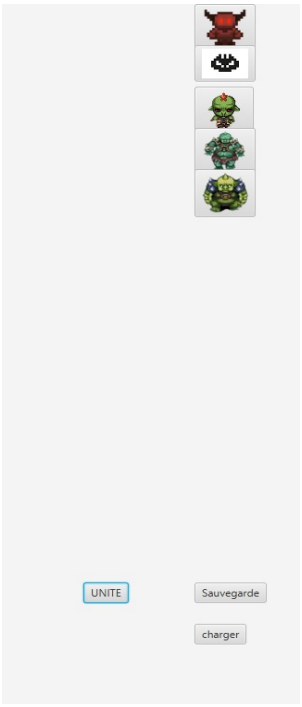
Ceci est une classe avec son propre main qu'on lance pour pouvoir créer des cartes pour notre jeu diablotin. C'est un outils de développement (à l'heure actuelle) car il n'est pas simple d'utilisation pour des joueurs.

A gauche la carte (vierge) , a droite des boutons pour sélectionner des objets à placer sur la carte.

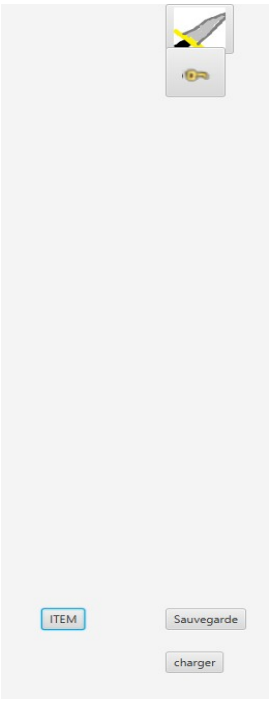
Le bouton DÉCOR qui permet de changer de type d’objet (DÉCOR → UNIT → ITEM → ÉVENT) .



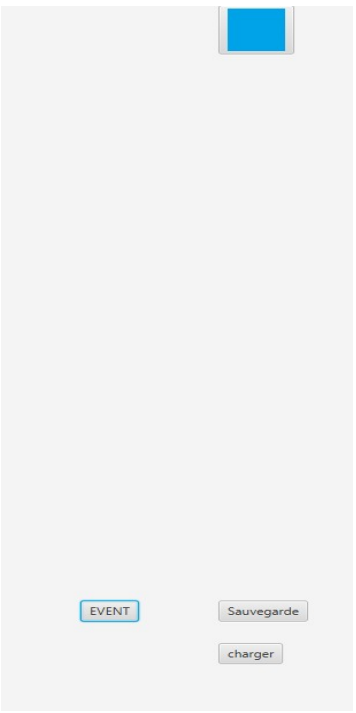
DÉCOR



UNIT



ITEM



ÉVENT

Si on sélectionne un décor on peux cliquer/drag sur la carte pour les placers , pour UNIT,ITEM,EVENT on ne peux que cliquer pour ne pas faire de doublons .

Pour EVENT, c’est une zone ou on doit tuer des unités pour gagner une épée donc pour placer la zone, il faut cliquer une première fois dans le coin en haut à gauche puis une deuxième fois dans le coin en bas à droite pour créer une zone de coordonnée du coin en haut à gauche au coin en bas à droite.

On peux aussi sauvegarder la carte alors elle sera dans le fichier CRÉATION sous le nom sauvegarde_création.txt . On peux donc aussi charger une carte mais elle doit être sauvegarder par création_carte.java et doit être dans le fichier CRÉATION sous le nom sauvegarde_création.txt .

Le but est de créer un carte ou d’améliorer une carte déjà faite pour notre jeu, après la carte finis il faudra la mettre dans le dossier NIVEAU et la nommé STAGE1.txt (on peux l’appeler STAGE2.txt pour faire une suite de niveau mais faudra adapter le code).

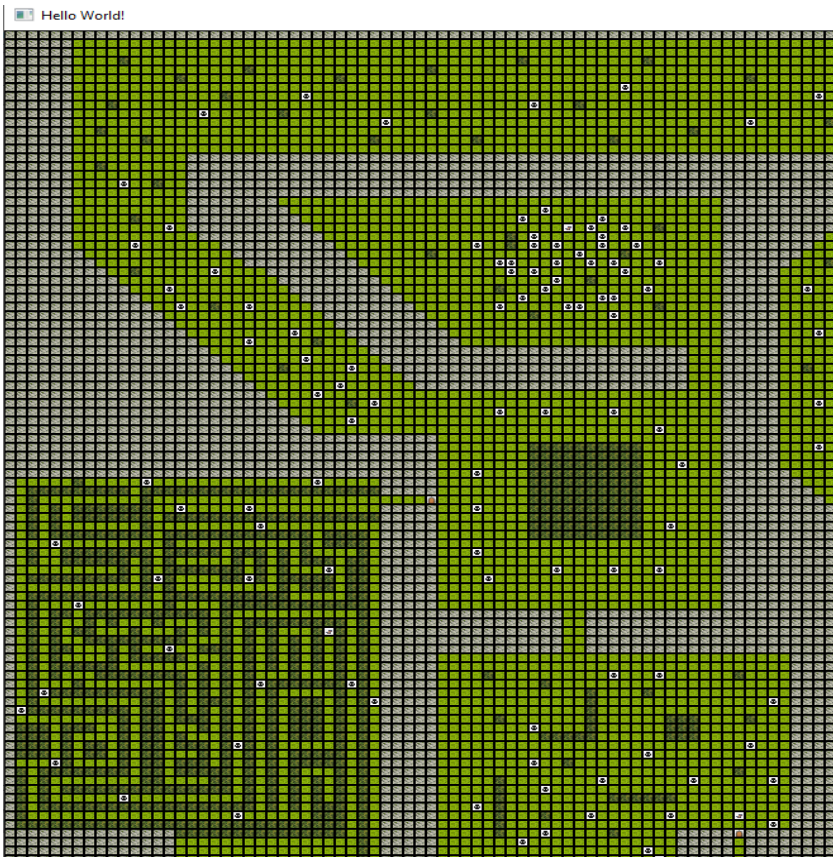
Pour supprimer une UNIT,ITEMS il faut mettre un DÉCOR par dessus .

Contrainte à ne pas faire pour une carte de jeu correct :

Il est possible de mettre plusieurs Lucifer ou aucun (notre héro en rouge) mais pour un jeu correct il faut en mettre un seul. (Aucun héro , en posera un quand même en 70 70)

On peux poser plusieurs UNIT au même endroit si on clique plusieurs fois.

On peux enlever les murs des bordures , mettre des portes sans clés et créer une carte impossible à finir.



Exemple d’une partie créé

Arborescence du Fichier :

Voici L’arborescence de notre fichier de façons a ce que l’utilisateur puisse le consulter sans problèmes :

build/ : Ce répertoire contient les fichiers générés lors de la construction du projet, tels que les fichiers compilés, les bibliothèques externes et autres ressources nécessaires à l'exécution de l'application.

CREATION/ : Ce répertoire contient des fichiers de carte créés. Il peut s'agir de fichiers de données ou de fichiers de configuration spécifiques à notre application de création de cartes.

dist/ : Ce répertoire contient les fichiers distribuables de votre application, tels que les fichiers **JAR** exécutables ou les packages d'installation.

map-Test/ : Ce répertoire contient des fichiers de test de cartes. Il peut être utilisé pour tester les fonctionnalités de votre application avec des cartes spécifiques.

NIVEAU/ : Ce répertoire contient les niveaux de notre jeu ou de notre application basée sur les cartes. Chaque niveau peut avoir ses propres fichiers et ressources associés.

Sauvegarde/ : Ce répertoire contient les Sauvegarde effectué par le joueur lors d'une partie

Src/ :

1. **Dossier "Diablo"** : Ce dossier contient le fichier principal de notre application. Il est nommé **"Main"** ou quelque chose de similaire. Ce fichier contient le point d'entrée de votre programme, c'est-à-dire la méthode **"main" où l'exécution de votre application commence.**

2. **Dossier "Image"** : Ce dossier contient des fichiers d'images utilisés dans notre application. Ces images peuvent être des icônes, des arrière-plans ou tout autre élément graphique nécessaire à votre jeu.

3. **Dossier "Jeu"** : Ce dossier contient deux autres fichiers importants pour notre jeu :

   Fichier "Model" : Ce fichier représente la partie modèle de notre application. Il peut contenir des classes et des logiques qui décrivent la structure des objets de votre jeu, les règles du jeu, la gestion des collisions, etc. Le modèle est généralement responsable de la gestion des données et de la logique métier.

   Fichier "Vue" : Ce fichier représente la partie vue de notre application. Il peut contenir des classes et des logiques qui gèrent l'affichage graphique du jeu, l'interaction avec les utilisateurs, les menus, les animations, etc. La vue est généralement responsable de la présentation des données au joueur.

Ces fichiers et dossiers constituent une structure de base pour notre projet Java. Cependant, veuillez noter que la structure exacte varie en fonction des conventions établie entre nous lors de choix de conception spécifiques que nous avons faits.

MANUEL D'INSTALLATION :

Assurez-vous d'avoir une machine Java correctement configurée :

Vérifiez que Java Development Kit (JDK) est installé sur votre machine. Vous pouvez le télécharger depuis le site officiel d'Oracle et suivre les instructions d'installation appropriées pour votre système d'exploitation.
Placez-vous dans le répertoire où se trouve le fichier .jar de votre projet, normalement dans le dossier "dist".

Ouvrez une fenêtre de terminal ou de ligne de commande dans ce répertoire.

Utilisez la commande suivante pour exécuter le fichier .jar de votre projet :

java -jar Jeu.jar
ou

java --module-path /chemin/vers/javafx/lib --add-modules javafx.controls,javafx.fxml -jar Projet/dist/Jeu.jar

Appuyez sur la touche "Entrée" pour exécuter la commande. Le lancement de l'application JavaFX devrait commencer.

Assurez-vous que toutes les dépendances requises par votre projet JavaFX sont présentes et accessibles à partir du fichier .jar. Cependant normalement notre projet contient bien les bibliothèques externes et ressources nécessaires, assurez-vous sinon de les inclure dans le répertoire approprié et de les référencer correctement dans notre projet.

PLANNING

Le Planning ce Situe dans le Dossier Planning et Détaille les Etapes de Développement .