Université de Mons
Faculté des sciences
Département d'Informatique

Sokoban

Rapport de projet

Professeur :
Hadrien Melot

Auteurs :
Pignozi Agbenda
Theo Godin
Ugo Proietti

Année académique 2020-2021

Table des matières

1 Introduction

Dans cette section, vous rappelez l'objectif de votre projet et donnez un aperçu de comment vous allez procéder.

1.1 Objectif

Ce projet est réalisé pour le cours de projet d'informatique donné en BAC 1 a l'université de Mons. Il s'agit d'un jeu de sokoban écrit en java sur la version 11. Cette version n'est pas la plus récente (java 15 était disponible lors de la réalisation du projet début 2021) mais c'est la version LTS (long term support) la plus récente. Voulant rendre le jeu open source sur GitHub, il nous semblait important d'utiliser une version LTS afin que d'autres étudiants puissent utiliser et/ou analyser notre code dans les années a venir.

1.2 Procédure

2 Mode d'emploi

Le projet utilise Gradle comme système d'automatisation permettant de gérer facilement les dépendances et la compilation.

2.1 Gradle

Nous avons ajoute deux task a Gradle, trouvable a la fin du fichier *build.gradle*
— checkMap
— movReplay

2.2 Dépendances

ffmpeg - nécessaire sur les systèmes UNIX afin d'afficher correctement la video de fond de l'écran principal.

3 Algorithme

Dans cette section, vous allez expliquer les différents algorithmes qui vous paraissent importants pour votre projet. (Pour l'explication : son principe, les grandes lignes de comment il s'exécute, sa complexité,...)

3.1 Programmation orientee objet

3.2 Chargement d'un niveau

3.3 Déplacements

3.4 Options utilisateur

3.5 Pack de textures

3.6 Génération aléatoire de niveau

3.6.1 Placement aléatoire

3.6.2 Backward induction

3.6.3 A* path finding

4 Points forts et points faibles

4.1 Points forts

4.1.1 Graphismes

Textures jolies et interfaces intuitives et soignees

4.1.2 Performances

Jeu fluide partout sauf dans le builder

4.1.3 Editeur de niveau

4.1.4 Enregistreur de niveau aleatoires

4.1.5 Menu d'options complet

4.1.6 Facilite de changer les textures

4.2 Points faibles

4.2.1 Generation aleatoire

4.2.2 Interface non responsive

4.2.3 Taille des niveaux limitee

5 Erreurs restantes

5.1 Generation

La generation aleatoire de niveaux aleatoires ne fonctionne pas dans tous des cas

5.2 Touches

6 Choix personnels

Dans cette section, vous allez expliquer et justifier les choix que vous aurez fait (par exemple pourquoi utiliser un tri à la place d'un autre).

6.1 Open source

6.2 Github

6.3 FXML

6.4 Tools

Pourquoi on a fait des tools pour nous faciliter la vie

6.5 Easter eggs

7 Problèmes survenus

7.1 Problèmes logistiques

Nous avons rencontre deux problèmes "logistiques" durant toute la durée du projet. En effet nous sommes deux étudiants sur Linux et un sur Windows. De plus les étudiants sur Linux utilisent Vim et VSCode comme éditeur de texte alors que l'étudiant sur Windows utilise Intellij.
Les problèmes rencontres ont étés lies a la façon dont Intellij gere Git et Gradle, ce n'a pas ete facile de bien le configurer, et encore moins facile de gérer les surprises que Intellij a push sur le repo GitHub. A plusieurs reprises l'étudiant en charge du repo a du gérer des conflits et parfois revenir sur des commits précédents pour annuler ce que Intellij fait en fond sans en avertir l'utilisateur.

7.2 Problèmes algorithmiques

Nous avons du faire des rework

8 Conclusion

Enfin dans cette section, vous ferez un bref rappel du sujet de votre projet, de comment vous avez fait pour résoudre le problème, des résultats s'il y en a(s'il y a beaucoup de résultats préférer une section à part entière) et enfin ce que le projet vous a apporté.