Sokoban

M2013 - Mannarelli, S2A

Introduction

Ce rapport détaille la réalisation d'un programme permettant de jouer au jeu Sokoban en mode texte.

Rappel des règles : Le joueur peut se déplacer sur les 4 directions de base. Dans son mouvement, il peut bouger une ou plusieurs caisses. Son but est de mettre ces caisses sur des cases cibles.

Plan du rapport

1. Analyse et conception

1.1. Détails des contraintes

Le programme doit pouvoir fonctionner en mode texte et permettre au joueur de choisir son plateau parmi une liste contenue dans une base de donnée

1.2. Choix de conception

On utilise quelques objets pour le stockage de données comme:

Case : Qui stocke un couple de coordonnées (x,y) ainsi que ses types (apparents et réels).

Board : Un objet contenant un tableau deux dimensions de Position.

Game : Qui s'occupe du fonctionnement du jeu.

1.3. Analyse rapide du fonctionnement

Le programme principal (que nous appellerons "Play" pour le reste du rapport) est donc en charge de :

-Afficher la liste du plateau.

-Demander le plateau sur lequel le joueur veut jouer

-Lancer la boucle de partie.

Dans une partie Play réalise les actions suivantes:

-Exécuter une boucle tant que la partie n'est pas finie.

-Afficher le plateau à chaque étape.

-Demander un déplacement et voir si c' est possible.

2. Réalisation et choix de programmation

2.1. Découpage en classes

On a découpé le programme en classe et défini les mêmes responsabilités à chaque classe comme spécifié dans la partie 1.2.

2.2. Classe Case

La classe Case stocke donc un couple de coordonnées ainsi que deux types de cases comme dit plus haut.

Elle dispose d'un constructeur :

-public Position(int row, int col , CaseType type) qui prend en paramètre deux entiers les affecte directement comme couple de coordonnées et un type de case et l'affecte également. En plus de cela à la création le second type est automatiquement défini à vide et sera modifié par la suite

2.3. Classe Board

La classe Board abrite un double tableau de Case permettant ainsi de créer un plateau de jeu à deux dimensions.

Son constructeur fait appel à la méthode createBoard() dont voici la signature : *public ArrayList<ArrayList<Case>>* createBoard*s()* comme on peut le voir cette méthode construit un double tableau de Case, totalement initialisés avec des Position avec des types vides.

Elle abrite également la méthode pour totalement afficher un plateau sous forme de texte. *public void printBoard().*

Ainsi que diverses méthodes permettant d'ajouter des éléments au plateau (murs, cases objectis, caisses, position joueur).

Cette classe dispose aussi d'une méthode qui transforme un Board en une ArrayList de String ce qui sera utile dans la classe Administrator.

2.4. Classe Game

La classe Game constitue le cœur du jeu. Elle assure l'interaction avec le joueur, gère les déplacements (tout d'abord leur possibilité puis leur réalisation) et est aussi en charge des conditions de victoire du jeu

2.5. Classe Player

La classe Player est une classe ne contenant qu' un main contenant l'appel à la méthode Play() de Game, qui elle-même lance le jeu, la boucle de jeu et propose au joueur de choisir son plateau.

2.6. Classe Administrator

La classe Administrator est une classe contenant un deuxième main, ainsi qu'une variété de méthodes permettant la gestion de la base de données contenant les plateaux.

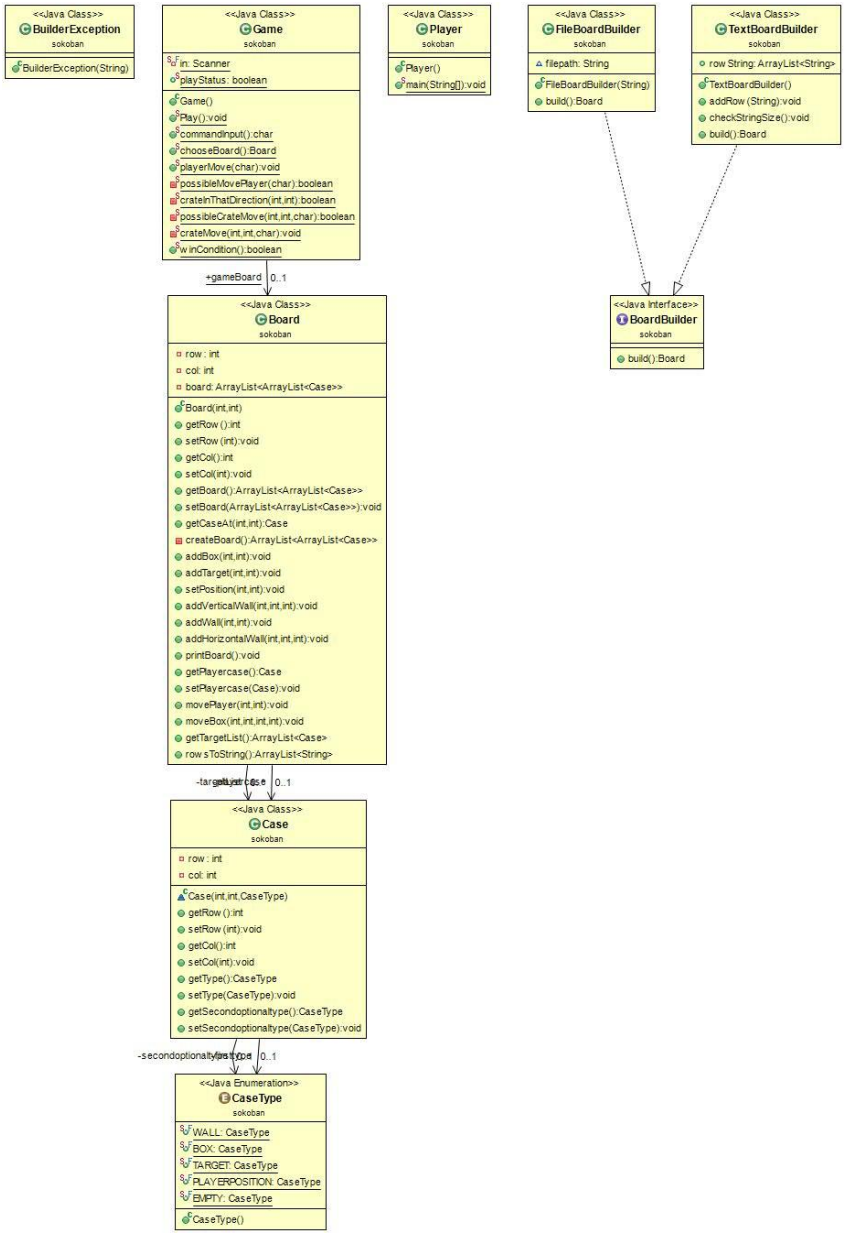
Comme par exemple la création de la base donnée avec createDatabase(), qui vérifie dans la base de données si une table contenant un booléen unique existe et si non créer le fichier ainsi que les tables correspondantes.

L'ajout d'un plateau à partir d'un fichier texte, qui passe par un FileBoardBuilder pour convertir un fichier en Board avant d'utiliser la méthode de celui-ci qui le transforme en ArrayList de String pour ensuite le stocker dans la table row de la base de donnée (en charge de stocker les lignes de chaques plateaux).

Une autre méthode gère son affichage en lisant ligne par ligne dans la table row. Et une affiche la liste des plateaux en prenant ses données dans la table board

Et enfin une méthode et en charge de la suppression du plateau de la base de données.

2.7. Diagramme de classe



3. Remarques et difficultés rencontrées

Difficultés rencontrées:

J’ai rencontré quelques problèmes avec sqlLite et Git, surtout avec le fait qu’il faille installer la librairie à chaque fois. Sinon je n’ai pas rencontré de difficultés particulières.