Rapport « Projet Sokoban »

M2103 2021

Ce rapport décrit la conception et la réalisation d'un programme d'un jeu d'un jeu de Sokoban, de l'implémentation d'une lecture de jeu depuis un fichier et de la réalisation d'une base de données.

Dans ce jeu, le joueur doit avancer son personnage représenté par un « P » sur un plateau de taille variable. L'objectif du jeu est d'amener les caisses (« c ») sur les cibles (« X ») placé un peu partout sur le plateau. S'il a réussi à amener toutes les caisses sur les cibles alors la partie et gagné. Mais s'il voit que la partie ne peut pas être terminer ou qu'il n'y arrive pas, il peut quitter à tout moment le jeu.

Analyse :

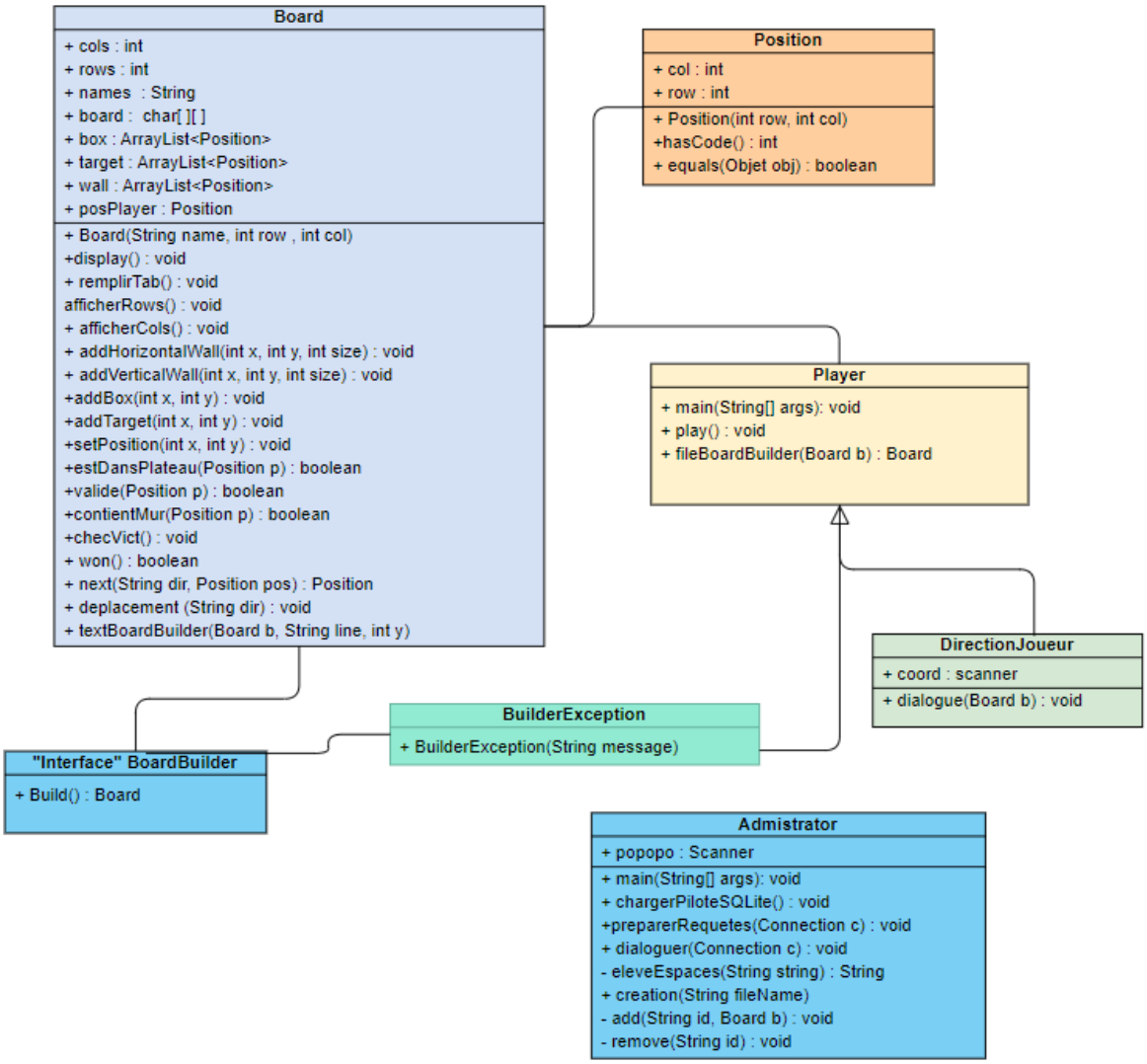
Spécification des besoins :

Le programme fonctionne dans la console NetBeans ou le plateau s'affichera et ou le joueur pourras jouer. Les éléments du plateau sont stockés dans des ArrayList.

Conception :

Informatique / iut de BORDEAUX

Rapport Projet Sokoban — Chagnaud Romain S2B



Rapport Projet Sokoban Chagnaud Romain S2B

Données et traitements :

Le programme principal consiste alors à :

- Construire un plateau
- A pouvoir afficher un plateau depuis un fichier
- Construire la base de données
- Mettre des fonctionnalités à la base de données

Pour une partie :

- Pouvoir stocker les positions des éléments du plateau
- Faire une boucle de jeu (début, fin)

Réalisation

La répartition des responsabilités entre les classes.

Dans la conception de ce jeu, j'ai utilisé un seul package qui gère tout le projet. Le package contient des classes ayant des rôles différents.

Le package projetSokoban: Le package projetSokoban est responsable de la gestion du projet. Il contient :

- **La classe Board :** La classe Board est responsable de l'affichage du plateau de jeu. Elle permet aussi de gérer les méthodes agissant sur le contenu des cases du plateau de jeu, tout ce qui est déplacement du joueur avec la caisse, la victoire du joueur et aussi l'affichage pour la lecture du fichier texte.

- **La classe Player :** est la classe principale, la classe contenant le main. C'est elle qui fait appel à la classe Board pour lancer une partie et afficher le plateau du fichier texte. Elle gère la boucle du jeu avec la méthode play.

- **La classe DirectionJoueur:** La classe DirectionJoueur est responsable du déplacement du joueur. Cette classe est composé d'une méthode dialogue qui gère le dialogue entre le joueur et le jeu.

- **La classe BuilderException:** La classe InvalidPositionException étend l'interface Exception et permet de lever des exceptions lorsqu'une position n'est pas valide. Cette classe est utilisée dans la classe Position et Board.

- **La classe Position :** La classe position est responsable de la position actuelle d'un caractère du plateau. Elle permet aussi de pouvoir utiliser les positions sur lesquelles le joueur veut se déplacer.

- **La classe Administrator :** cette classe gère la création de la base de donnée, avec différente option comme la création d'un nouveau plateau dedans, de lister les plateau, de les voir et les supprimer.

Les choix effectués pour la programmation.

Pour la réalisation de ce jeu, j'ai effectué certains choix. En effet, le plateau de jeu est un tableau à deux dimensions de caractères. Le joueur est associé au carcatère « P ». De cette façon, j'affiche le caractère «. » lorsque les cases du plateau ne sont pas occupées. Alors le caractère croix « X » représente la cible ou il faut amener la caisse et le "C" est associé à la caisse.

Quand la position du caractère « C » est égale à la position du caractère « X » la partie est finie.

Le joueur peut choisir la position de son « pion » au départ, combien de caisse il veut mettre et où. L'emplacement des murs et des cibles avant de lancer une partie.

La classe Player contient une méthode run() responsable du lancement du jeu. D'une méthode play qui prend en paramètre un board crée et un dialogue (méthode de la classe Direction joueur) qui gère la boucle de jeu. Et d'un fileBoardBuilder qui gère l'affiche d'un fichier texte choisi.

Lorsque le joueur saisit une position à laquelle il souhaite jouer, il saisit une chaîne de caractères « LRUD » avec une lettre majuscule pour donner la direction. Cette chaîne de caractère est gérée par deux switch un dans la class Board (méthode next) et l'autre dans DirectionJoueur (méthode dialogue).

Rapport Projet Sokoban Chagnaud Romain S2B

La classe Administrator crée une base de données dans laquelle on peut crée de nouveau plateau, lire des plateaux qui sont des fichier texte, supprimer

J'ai fait l'implémentation d'une commande pour quitter le jeu : « /quit » à tout moment le joueur peut partir et un message s'affiche : « Abandon du joueur ».

Quelque image du jeu :

```
coucou

    0 1 2 3 4 5 6 7 8 9
0   # # # # # # # # # #
1   # . . . . . . . . #
2   # C . C . . . . . #
3   # X X . P . . . . #
4   # . . . . . . . . #
5   # . . . . . . . . #
6   # . . . . . . . . #
7   # . . . . . . . . #
8   # . . . . . . . . #
9   # # # # # # # # # #
Commande : [LRUD]


    0 1 2 3 4 5 6 7 8 9
0   # # # # # # # # # #
1   # P . . . . . . . #
2   # C . . . . . . . #
3   # X . . . . . . . #
4   # . . . . . . . . #
5   # . . . . . . . . #
6   # . . . . . . . . #
7   # . . . . . . . . #
8   # . . . . . . . . #
9   # # # # # # # # # #
Commande : [LRUD]
D


     0 1 2 3 4 5 6 7 8 9
 0   # # # # # # # # # #
 1   # . . . . . . . . #
 2   # P . . . . . . . #
 3   # C . . . . . . . #
 4   # . . . . . . . . #
 5   # . . . . . . . . #
 6   # . . . . . . . . #
 7   # . . . . . . . . #
 8   # . . . . . . . . #
 9   # # # # # # # # # #
 Vous avez Gagné
```

```
Commande : [LRUD]
u

Pas bonne lettre
    0 1 2 3 4 5 6 7 8 9
0   # # # # # # # # # #
1   # . . . . . . . . #
2   # C . . . . . . . #
3   # X . . P . . . . #
4   # . . . . . . . . #
5   # . . . . . . . . #
6   # . . . . . . . . #
7   # . . . . . . . . #
8   # . . . . . . . . #
9   # # # # # # # # # #
```

```
     0 1 2 3 4 5 6 7 8 9
0    # # # # # # # # # #
1    # . . . . . . . . #
2    # C . . . . . . . #
3    # X . . P . . . . #
4    # . . . . . . . . #
5    # . . . . . . . . #
6    # . . . . . . . . #
7    # . . . . . . . . #
8    # . . . . . . . . #
9    # # # # # # # # # #
Commande : [LRUD]
/quit
Abadon du Joueur
```

Conclusion :

Ce projet m'a permis de bien revoir les dernières notions que nous avons vu en cours de programmation et de réaliser autre un jeu en partant de 0 tout seul. J'aurais pu améliorer la partie concernant la base de données, mais un manque de temps et connaissance, m'ont « empêché » de finir cette partie. Cependant, j'ai pris le parti de prendre soin de réaliser correctement les fonctionnalités principales du jeu et de bien faire la seconde partie. Ainsi, en étant confrontés à différentes difficultés j'ai fait mon maximum et j'ai bien progressé.