**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.

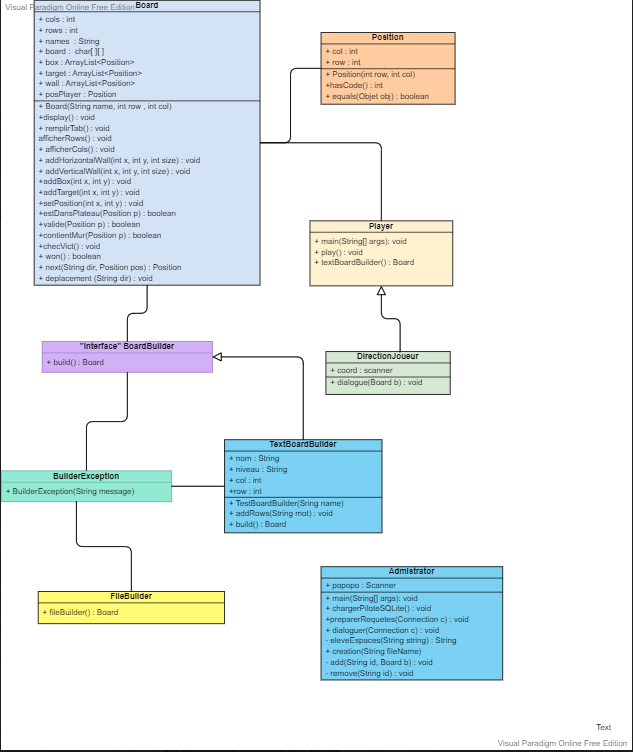
Dans ce jeu, le joueur doit avancer son personnage représenté par un « P » sur un plateau de taille variable. L’objectif du jeu est d’amener les caisses (« c ») sur les cibles (« X ») placé un peu partout sur le plateau. S’il a réussi à amener toutes les caisses sur les cibles alors la partie et gagné. 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 :**



**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 board est responsable de la gestion du plateau de jeu.

* **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.
* **L’ énumération Color :** L’énumération Color permet d’associer des couleurs à chaque joueur.
* **La classe Game :** La classe Game est responsable de l’état de la partie. Cette classe indique si une partie est finie ou non et si ce n’est pas le cas, au joueur de jouer. Elle utilise des méthodes de la classe Position indiquant l’état du plateau de jeu.
* **La classe InvalidPositionException :**  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 l’état du plateau de jeu. Elle permet de connaître les cases et lignes occupées. Elle permet aussi de pouvoir utiliser les positions sur lesquelles l’utilisateur veut poser un pion.

**Le package Game :**

* **L’interface Player :** L’interface Player permet d’implémenter des méthodes communes aux joueurs humains de la classe HumanPlayer et aux joueurs de la classe IA.
* **La classe HumanPlayer :** La classe HumanPlayer permet au joueur de choisir des paramètres de jeu tels que la taille du plateau ou les positions auxquelles il veut poser des pions. Cette classe est utilisée dans la classe Match qui fait appel aux joueurs.
* **La classe IA :** La classe IA permet de créer des joueurs IA.
* **La classe Match :** LA classe Match est responsable du lancement d’une partie. Elle est utilisée dans la classe ProjetPOOGomoku. Cette classe fait appel aux classes Board, et HumanPlayer pour la création d’un match et utilise les méthodes implémentées dans les classes Game et Position pour le lancement du jeu.

**Le package Main :**

* La classe ProjetPooGomoku est la classe principale. C’est elle qui fait appel à la classe Match pour lancer un match et donc une partie entre deux joueurs.

**Le diagramme de classe “général” illustrant l’architecture de notre programme.**

Game

Board

Color

Position

ProjetPOOGomoku

Match

Player

HumanPlayer

IA

**Board**

**Match**

**Main**

**Les choix effectués pour la programmation.**

Pour la réalisation de ce jeu, nous avons effectué certains choix. En effet, le plateau de jeu est un tableau à deux dimensions de couleurs. Chaque joueur est associé à une couleur. De cette façon, on affiche le caractère espace lorsque les cases du plateau ne sont pas occupées. Alors caractère croix “x”, ou rond “o” associé au joueur courant est affiché sur le plateau quand un nouveau pion est placé sur le plateau.

La classe Match contient principalement une méthode run() responsable du lancement du jeu. La classe ProjetPOOGomoku est chargée de créer une nouvelle instance de match et lancer le run. De cette façon l'exécution d’une partie se déroule avec l’appel à la méthode run

Nous avons fait en sorte que le joueur puisse choisir la taille du plateau de jeu. Nous vérifions alors la validité de la taille de plateau de jeu entrée par le joueur. De plus, lorsque le joueur saisit une position à laquelle il souhaite jouer, il saisit une chaîne de caractère avec une lettre majuscule pour les lettres et un chiffre pour les colonnes. Cette chaîne de caractères est convertie en entiers par la suite de façon à pouvoir l’utiliser comme une position de deux entiers ligne et colonne.

**Conclusion :**

Ce projet nous a permis de nous confronter aux dernières choses que nous avons vu en cours de programmation et de réaliser un jeu en partant de 0. Nous avons aussi véritablement appris à utiliser git par la pratique quotidienne de dépôts. De plus, nous avons appris à travailler à plusieurs sur un même projet et à nous organiser sur la répartition des différentes tâches à effectuer. Nous aurions pu améliorer la partie concernant le joueur IA et la commande permettant de quitter le jeu. Cependant, nous avons pris le parti de prendre soin de réaliser correctement les fonctionnalités principales du jeu. Ainsi, en étant confrontés à ces difficultés nous avons fait notre maximum et nous avons progressé.