université de Caen
Basse-Normandie

Complément POO
Rapport du Devoir

Beauchamp Aymeric 21301016
Chagneux Dimitri 21606807
Mori Baptiste 21602052
Leblond Valentin 21609038

L2-Info-groupe-4A

Table des matières

Introduction

L'objectif de ce projet étant de réaliser un Taquin qui est un casse-tête consistant à déplacer des cases d'un plateau afin de les replacer dans l'ordre et de reconstituer l'image ou le paterne souhaité.

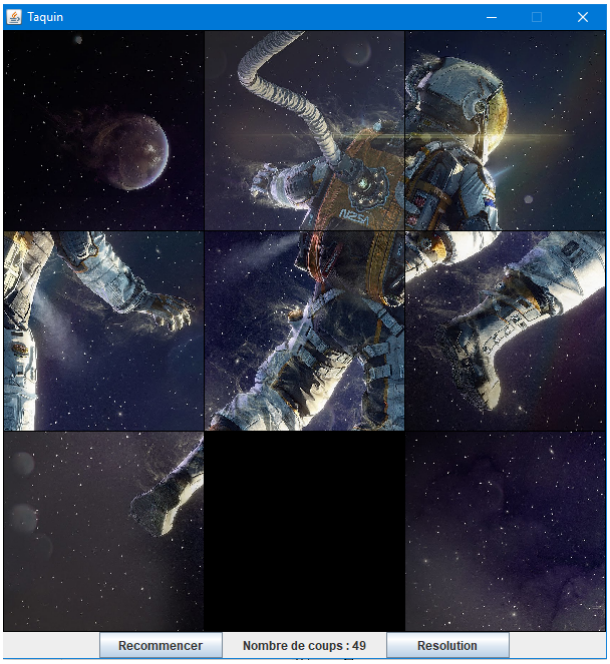

FIGURE 1 – Jeu du Taquin

Nous avons séparé le projet en deux packages, le premier comporte la version console avec la structure du jeu (le package **model**) et le deuxième contient toute la partie graphique (le package **GUI**), nous avons un troisième dossier qui possède les images pour l'interface graphique (le package **ressources**).

1 La conception du package model

1.1 Organisation des classes

Tout d'abord, nous avons représenté les cases par deux types de classe, FullTile pour les cases pleines et EmptyTile pour la zone vide qu'on déplacera. Ces deux classes possède des attributs en commun qui sont les coordonnées X et Y de la case c'est pourquoi on a fait une classe abstraite Tile qui possède ces coordonnées.
La classe FullTile possède un identifiant sous forme d'entier qui la caractérise des autres cases, on s'arrangera donc à leur mettre une valeur différente pour chaqu'une de ces cases.

Ensuite, nous avons une classe Board qui représentera l'état du jeux et qui fera en sorte de déplacer la case vide, résoudre le niveau avec un solveur, mélanger le jeu, etc.

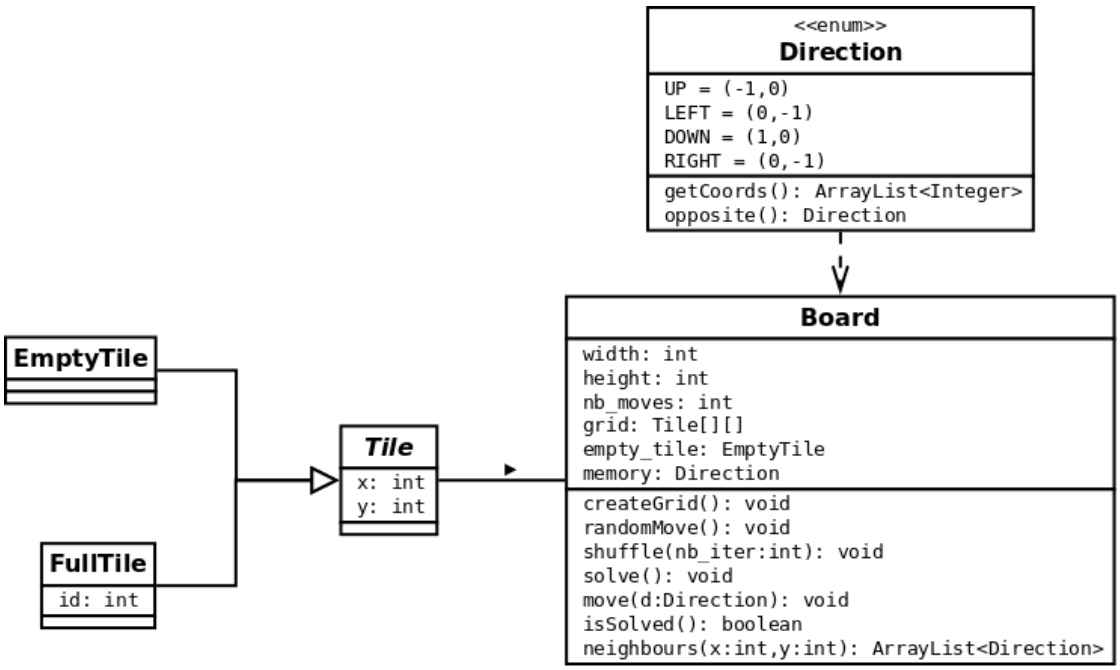

FIGURE 2 – Diagramme de classe du package model

1.2 Algorithmes du Board

Tout d'abord, nous avons une fonction **createGrid** qui permet d'initialiser une grille avec des FullTile et une EmptyTile et une fonction **toString** qui permet de l'afficher.

```
$ java model/Main
0 1 2
3 4 5
6 7
```

FIGURE 3 – Affiche d'une grille 3x3 initialisée

Ensuite, nous avons créer les déplacements de la case vide, nous avons d'abord commencer par créer des énums UP, DOWN, LEFT, RIGHT qui représentent à la direction de déplacement de la case vide.

Pour mélanger le jeu, on choisi un mouvement aléatoire selon les mouvements possible de la case vide (sous forme d'énums) puis on déplace la case vide jusqu'en bas à droite. On garde en mémoire le coup aléatoire effectué pour ne pas faire le coup opposé pour qui ne s'annulent pas comme par exemple UP o DOWN o UP o DOWN qui ne modifie pas le jeu une fois le mélange appliqué. Pour effectuer un mouvement utilise une énum donnée et on effectue le mouvement.

La récupération des coups jouables de la case vide est un voisinage de taille 4 de rayon 1 qui correspond aux quatre direction haut, bas, gauche, droite et la fonction **neighbours** qui selon une coordonnée dans la grille du jeu donne toutes les possibilité des directions.

2 Conception du package GUI

2.1 Organisation des classes

2.2 Fonctionnement GUI