Rapport-Takenoko

Berriri Mehdi,Bounab Fayçal,Djema Sofiane,Doglio Arthur

Novembre 2018

1 Introduction

Dans ce rapport, nous allons présenter les différents concepts du génie logiciel que nous avons utilisé pour notre projet. Le projet est la réalisation d'un Takenoko avec une interface textuelle.

2 Organisation générale du code

Nous avons regroupé les classes dans les packages suivant: takenoko, moteur, personnage, objectif, joueur

Dans le package takenoko nous avons les classes principales ainsi que les packages joueur et moteur.

Dans le package moteur on a le cœur du jeu.
La classe Partie dans laquelle va se dérouler une partie , Cette classe initialise le jeu, l'exécute puis détermine qui sont les vainqueurs.
La classe Plateau va représenter le plateau du jeu. Elle contient deux Hashmap qui lient une coordonnée en 3 dimensions respectivement à une irrigation ou à une parcelle. Cette classe calcule également les emplacements dont nous avons besoin pour respecter notre système de coordonnées. Enfin, elle effectue la pose d'éléments sur le plateau.
La classe Deck s'occupe de la création des piles d'objectifs et de parcelles, ainsi que de la manière dont on les pioche.
Les classes Bambou, Irrigation et Parcelle permettent de créer des instances des objets qu'on crée sur le plateau.
La classe Parcelle contient des Bambous et des méthodes relatives à sa pousse.
La classe Affichage s'occupe d'afficher différentes informations sur le déroulement de la partie. La classe Enums contient des énumérations permettant de définir les couleurs pour les parcelles, les actions qu'il est possible d'effectuer pour un joueur dans un tour et les couleurs pour différencier les joueurs.

Le package moteur contient deux sous packages.
Le premier est le package objectifs qui contient la classe mère Objectif et trois

classes filles ObjectifJardinier ObjectifPanda et ObjectifParcelle. Le second package nommé personnages est composé de trois classes.
La classe Personnage permet de calculer les déplacements possibles d'un personnage sur le plateau. Et les classes Panda et Jardinier effectuent des actions relatives à leur rôle dans le jeu.

Le package joueur contient la classe Joueur qui effectue toutes les interactions avec le moteur du jeu pour permettre au joueur de jouer. Il possède également la classe mère Bot (qui hérite elle-même de Joueur) ainsi que les classes filles BotJardinier, BotPanda, BotParcelle et BotRandom. Ces classes de bots permettent d'effectuer des choix sur les actions à effectuer dans la partie. L'interface IA est implémentée par Joueur et toutes ses classes filles. Elle contient les définitions des méthodes relatives aux choix qu'un joueur peut avoir pour interagir avec le jeu.

3 Patrons de conception

Pour ce projet nous avons utilisé deux patrons de conception.

Le premier est le patron de conception Singleton.
Nous avons eu recours à ce patron pour les classes Jardinier, Panda, Plateau et Deck. L'utilisation de ce patron nous a permis d'accéder à ces classes de manière plus simple dans le code. En contrepartie il faut pour chaque nouvelle partie penser à réinitialiser les attributs de ces classes.

Le second patron de conception employé est le pattern Template Method. Nous l'avons employé dans la classe Joueur et ses classes filles ainsi que dans la classe Personnage et ses classes filles. Grâce à ce patron de conception nous avons pu redéfinir les actions des deux personnages Panda et Jardinier lors de leur déplacement sur une case sans réécrire deux fois la méthode de déplacement.

Dans la classe Joueur cela nous a permis de redéfinir les choix du joueur dans les classes des bots afin que celles-ci n'aient aucune modification à apporter sur les éléments constituants le jeu.

Enfin dans la classe Objectif nous utilisons ce patron de conception dans les classes filles afin de redéfinir les méthodes qui gèrent la validation des objectifs.

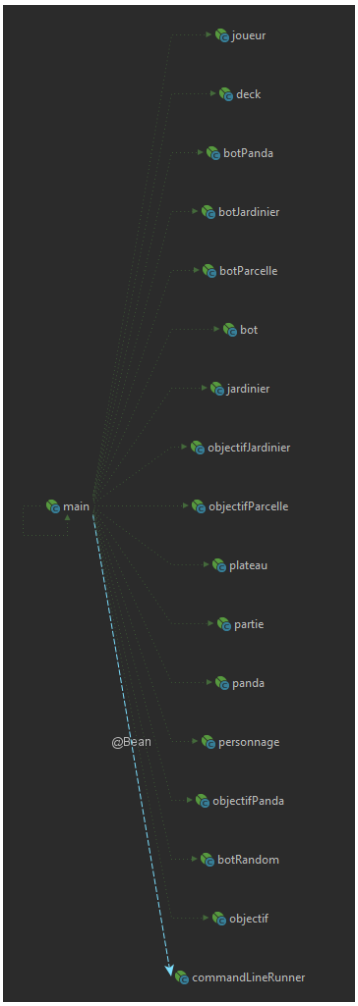
4 Spring

La migration vers Spring s'est effectuée en plusieurs étapes. Tout d'abord, il a fallut ajouter les dépendances Maven. Puis nous avons fait l'injection des dépendances par mutateur, ce qui signifie que nous avons du supprimer les constructeurs et utiliser des setters. Ce qui implique la réécriture de la créations de

certains objets tels que les objectifs ou les parcelles. Les beans de notre projet avec une portée de type singleton: -La classe Personnage annoté @Primary et ses filles, Jardinier et Panda -La classe Deck -La classe Plateau

Les beans de notre projets avec une portée de type prototype: -La classe Objectif annoté @Primary avec ses filles, ObjectifJardinier, ObjectifPanda et ObjectifParcelle -La classe Joueur annoté @Primary avec ses filles, Bot, Bot-Parcelle, BotJardinier, BotPanda et BotRandom(conservé uniquement pour le TER)

De plus, seul la classe Main est annoté @SpringBootApplication



5 Conclusion

Ce projet nous a permis de comprendre un petit mieux le concept de patrons de conceptions qui s'avèrent très utiles quand le projet contient beaucoup de code.

Au niveau de la gestion du projet en équipe, nous avons réussi à bien nous répartir les tâches afin de réaliser nos objectifs dans les temps et l'ambiance générale du groupe était très bonne, nous sommes entièrement satisfaits de ce que nous avons réalisé