PROJET HEX

BOUIZEGARENE Hachimi
ESTRADE Yoan
BENYETTOU Imane
201

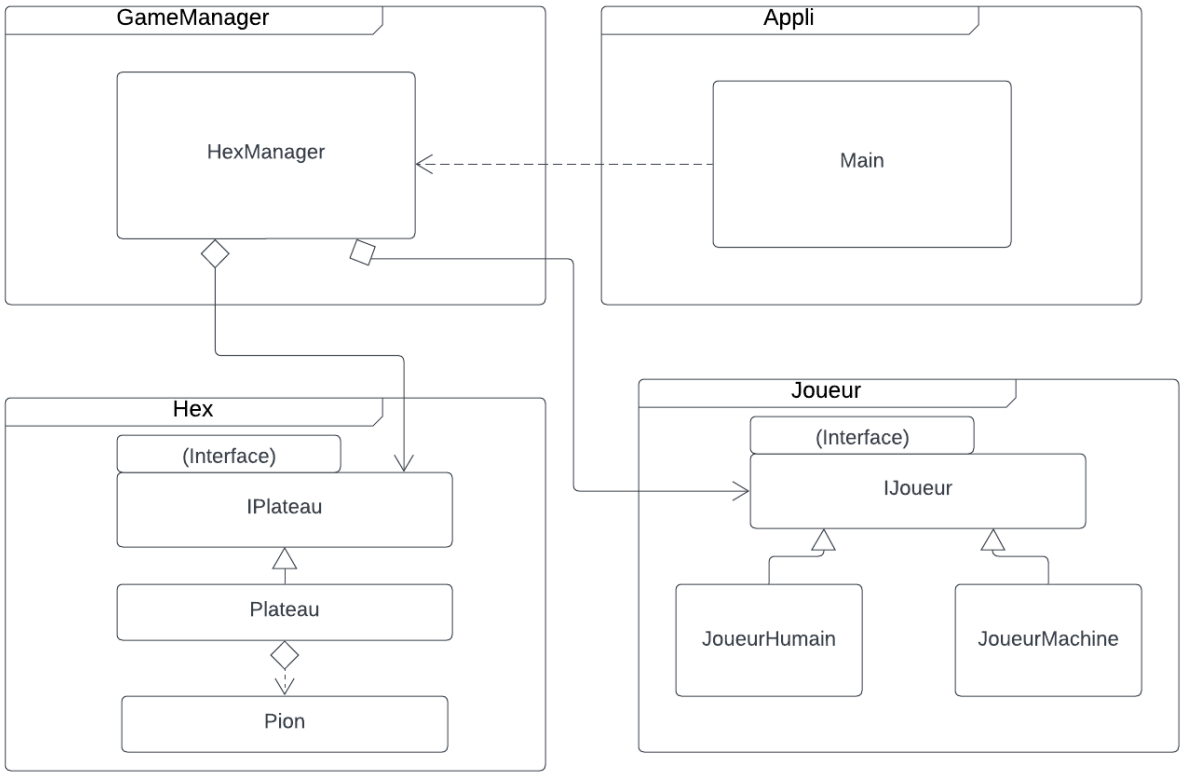
Table des matières

Introduction

Ce projet consiste en le développement d'un jeu fonctionnel (hex), permettant à deux joueurs de s'affronter dans une partie complète, les joueurs pouvant être soit humain, soit un algorithme, avec des coups choisis aléatoirement à chaque fois. L'interface à été réalisé en mode texte plutôt qu'en mode graphique, par simplicité et gain de temps.

Le projet comporte plusieurs classes Pion et Plateau, ainsi qu'une classe principale Appli, comportant la fonction main.

Diagramme d'architecture



Liste des tests unitaires

Classe PionTest.java :

- Le constructeur Pion(**char** symbole)
- La methode toString()
- La methode static get(**char** c)

Classe PlateauTest.java :

- Le constructeur Plateau(**int** taille)
- La méthode toString()
- La méthode jouer(String coord)
- La méthode estValide(String coord)
- La méthode getCase(String coord)
- Les méthode coordEnDessous(String coord), coordAuDessus(String coord), coordAdroite(String coord), coordAgauche(String coord)
- La méthodecoordAutour(String coord)
- La méthodeestCheminVers (String coord, ArrayList<String> array, **char** direction)
- La méthodeVerifPartie()

Classe JoueurMachineTest.java :

- Le constructeur JoueurMachine(String nom)
- La méthode getNom()

Classe JoueurHumainTest :

- Le constructeur JoueurHumain(String nom)
- La méthode getNom()

Explication des tâches à accomplir

Pour recréer le jeu en java :

- Il faut tout d'abord une classe Pion, ayant pour attributs un symbole (croix, rond, ou vide) pour représenter les coups des joueurs

- Une classe Plateau ayant pour attributs une taille, un nombre de joueurs, et une première ligne et colonne pour définir les coordonnées des cases.

    - Le constructeur et la méthode redéfinie toString() pour formater l'apparence du plateau
    - La méthode jouer() pour gérer la partie (mouvement des pions et tour des joueurs
    - Les méthodes coordEnDessous(), coordAuDessus(), coordADroite() et coordAGauche(), qui renvoient respectivement la coordonnée sur chaque cardinalité. ainsi que AllCoordsEnDessous(), AllCoordsAuDessus(), AllCoordsAGauche() et AllCoordsADroite(), qui renvoient respectivement l'ensemble des coordonnées sur chaque cardinalité.
    - Les méthodes estCheminVersBas() et estCheminVersDroite() pour tester l'existence d'un chemin vers la droite ou vers le bas, qui servira pour définir le gagnant de la partie
    - Et enfin la méthode VerifPartie(), qui détermine à l'aide des méthodes estCheminVersLaDroite() et estCheminVersLeBas() le gagnant de la partie.

- Des fichiers tests, pour effectuer les tests unitaires et vérifie que chaque méthode renvoi ce qu'elle est censée renvoyer (un fichier test pour le Plateau, et un pour le Pion)
- Une classe Appli contenant le main, afin de pouvoir jouer les parties

Bilan du projet : Ce qui est réussi, ce qui peut être amélioré

Difficultées

Les principales difficultées rencontrées durant la réalisation de ce projet ont été de déterminer lorsque la partie est terminée, qui à remporté la victoire : nous avons mis du temps à trouver l'idée d'implémenter une fonction récursive pour déterminer l'existence de chemins, et donc la fin de la partie. Nous avons aussi eu du mal à appliquer à la lettre les principes SOLID, car pour chaque fonction, il fallait penser à si elle respectait ces principes, et la modifier si ce n'était pas le cas, ajoutant du travail supplémentaire.

Améliorations

Nous pensons que le code peut être optimisé, et rendu plus court : certaines fonctions pourraient être assemblées et différenciées par un paramètre, rendant le code plus lisible, ainsi que d'autres détails pouvant être améliorés.