Rapport « Gomoku »

M2103 2021

Ayman KACHMAR et Mathieu RAKOTOARISOA, groupe S2A”

Ce rapport décrit la réalisation du jeu Gomoku, un jeu ressemblant au Morpion, consistant à aligner 5 pions de cases voisines du plateau de jeu pour gagner.

Plan du rapport

1) Choix effectués

Pour effectuer ce programme, nous nous sommes inspirés des TP's que nous avons réalisés (le menu, par exemple, vient du TP bibliothèque) et du projet Towa (coordonnées, case...). Nous avons tenté de réaliser un code cohérent en utilisant les notions apprises en cours : les classes abstraites, les ArrayList, la récursivité, les exceptions et évidemment les tests unitaires. Nous n'avons pas créé de collection. En effet, bien que nous voulions appliquer toutes les nouvelles notions apprises, nous n'avons pas trouvé d'utilité à utiliser une collection dans cette situation.

2) Répartition entre classes

a) Conception

Notre méthode **main()** se trouve dans la classe **Jeu**. Cette classe permet de créer une instance de la classe partie depuis la méthode **menuPrincipal()** Une partie est constituée de deux joueurs et d'un plateau, entre autres attributs nécessaires au déroulement d'une partie.

Joueur est une classe abstraite, mère de deux classes : **JoueurHumain** et **JoueurIA**, ayant en commun la méthode abstraite **jouer()**. **JoueurIA** génère des coups dans cette méthode, qu'il va envoyer dans la classe **Partie**. **JoueurHumain** saisit lui-même les coups qu'il veut jouer. Un **Plateau** est constitué, entre autres, d'un double tableau de **Case**. Une case a plusieurs propriétés, dont des coordonnées. La classe **Coordonnees** sert pour l'association d'une ligne et d'une colonne dans le plateau, et pour faciliter l'utilisation d'une **Case**. Cette classe est nécessaire pour, notamment, trouver des coordonnées voisines avec l'aide de la classe **Direction**.

b) Organisation en packages

Jeu

- Jeu.java (le main)
- Partie.java (gestion de partie)

Joueurs (création de joueurs)

- Joueur.java
- JoueurHumain.java

- JoueurIA.java

Positions (utilisation de cases)

- Case.java
- Coordonnees.java
- Direction.java
- Plateau.java

Package Jeu

- La classe Jeu est la classe principal “main” qui permet de lancer le menu principal menuPrincipal() et par conséquent le jeu ; elle pose des questions à l’utilisateur s’il ne s’agit pas d’une IA, pour entrer un nombre de tours, le nom, la taille du plateau à l’aide de questionInt() et questionString().
- La classe Partie gère une partie, en faisant jouer les deux joueurs tours à tours tant que le coup gagnant ou le nombre de tours ne remettent pas en question la fin de partie. La partie gère aussi l’entrée de l’utilisateur, en vérifiant que le coup joué est valide et en demandant une nouvelle saisie en cas d’erreur.

Package Joueurs

- La classe abstraite Joueur permet la création d’un joueur, qui possède un nom et un booléen estIA, et la méthode abstraite jouer() qui servira pour les deux prochaines classes dérivées de celle-ci.
- La classe JoueurHumain sert à l’utilisateur qui joue, la fonction jouer() est implémentée pour jouer le coup, ajouter ce coup dans la liste des coups joués, et modifier le plateau.
- La classe JoueurIA permet la création d’une IA et la fonction jouer() implémentée par un programme récursif, cherchant un coup présent dans la liste des coups jouables.

Package Positions

- La classe Case définit, pour chaque case une coordonnée, deux attributs booléens jouable et gagnant. Elle permet de définir la Couleur de chaque case et de modifier cette Couleur en fonction du coup joué. Deux fonctions servent pour la partie, actualiseCaseJouable() permet, après un coup joué, de trouver les voisines de ce coup et puis de mettre le booléen jouable de ces cases voisines à vrai. setGagnante() permet quant à elle de trouver s’il y a 4 cases de directions

complémentaires et de même Couleur alignées, dans ce cas l'attribut gagnant est vrai et la partie se termine.

- La classe Coordonnees contient une coordonnée composé de ligne et colonne, et permet la conversion de chaine de caractère du coup joué en coordonnée, et inversement. Elle est composée de fonctions qui permettent de trouver les cases voisines proches, et les cases voisines concernant des directions complémentaires.
- L'énumération Direction permet l'utilisation de directions pour parcourir les cases voisines à l'aide des fonctions mvtHoriz() et mvtVertic()
- La classe Plateau permet l'initialisation de cases et de coordonnées associées à ces cases, elle permet l'affichage du plateau et des cases, et contient des fonctions utilisées lors d'une partie : victoire(String coup) permettant de retourner vrai s'il y a victoire, actualiserPlateau(String coup) qui actualise le plateau après un coup joué et modifPlateau(boolean estNoir, String coup) permettant de modifier la Couleur de la classe après le coup joué.

Un package est réservé pour les tests unitaires : il est composé de CoordonneesTest.java, CoupTest.java, VictoireTest.java

- CoordonneesTest permet de tester la présence de coordonnées voisines et la conversion de chaîne en coordonnée
- CoupTest sert à tester, après le jeu d'un coup, si une case est jouable (si elle est voisine d'une case déjà jouée)
- VictoireTest vérifie s'il y a détection d'une victoire après un dernier coup clé joué :

```
46          plat = new Plateau(10, 10);
47
48          // sur ligne
49          j1.jouer("A0", coupsJoues, plat, p, estNoir, !estIA);
50          j1.jouer("A1", coupsJoues, plat, p, estNoir, !estIA);
51          j1.jouer("A2", coupsJoues, plat, p, estNoir, !estIA);
52          j1.jouer("A3", coupsJoues, plat, p, estNoir, !estIA);
53          j1.jouer("A4", coupsJoues, plat, p, estNoir, !estIA);
54
55          boolean fini3 = plat.victoire("A4");
56          assertTrue(fini3);
57
58          plat = new Plateau(10, 10);
59          // pas de victoire
60          j1.jouer("A0", coupsJoues, plat, p, estNoir, !estIA);
61          j1.jouer("A1", coupsJoues, plat, p, estNoir, !estIA);
62          j2.jouer("A2", coupsJoues, plat, p, !estNoir, !estIA);
63          j1.jouer("A3", coupsJoues, plat, p, estNoir, !estIA);
64          j1.jouer("A4", coupsJoues, plat, p, estNoir, !estIA);
65
66          boolean fini4 = plat.victoire("A4");
67          assertFalse(fini4);
```