Développement orientée objet - Java
TP 08 - Partie 3 (pas relu, pas corrigé)

Lionnel Conoir, Isabelle Delignières, Wajdi Elleuch et Rémi Synave

Dans ce TP, vous allez mettre en place toutes les classes nécessaires pour créer le jeu de plateau.

Le jeu

C'est un jeu de plateau à deux joueurs qui pourrait être étendu à 4 joueurs facilement. Il se joue en temps à tour. Chaque joueur va jouer un coup : un déplacement ou une attaque.

Le plateau

Le jeu se joue sur un plateau type échiquier de 9 cases x 9 cases.
Les cases peuvent être repérées par leurs coordonnées allant de (0,0) à (8,8) mais aussi par un code composé d'une lettre et d'un chiffre. Les colonnes sont désignées par une lettre. La colonne la plus à gauche porte la lettre 'A' et la colonne la plus à droite, la lettre 'I'. Les lignes sont désignées par des chiffres. La ligne la plus en bas est désignée par 1, la ligne la plus haute par 9. La case en bas à gauche est donc désignée par A1 ou (0,0). La case centrale sera notée (4,4) ou E5.

Les pièces, leur déplacement et les prises

Les pièces sont des pokemons. Une pièce est située sur une case et le pokemon possède toutes les caractéristiques habituelles.
Chaque joueur peut poser, en début de partie des pokemons sur les 3 premières de son côté (à adapter pour 4 joueurs). Chaque joueur a les choix des pokemons qu'il pose. Toutefois, la somme des PV de tous les pokemons de son équipe ne doit pas excéder 1500. Une autre contrainte est que la pièce centrale sur la première ligne doit être mewtwo et qu'il n'est plus possible d'en ajouter d'autre.

Un exemple de plateau acceptable est montré sur la figure **??**.

FIGURE 1 – Plateau acceptable pour lequel chaque joueur a des pokemons dont la somme des PV est égal à 1404. Il serait encore possible d'en positionner un ou deux supplémentaires de chaque côté. il est à noter que les deux côtés auraient pu avoir des pokemons totalement différents. La seule contrainte est posée sur le total des PV.

Un joueur, lorsque c'est son tour, peut soit déplacer une pièce, soit attaquer une pièce adverse.
Une pièce peut se déplacer de une (et une seule) case seulement dans les 8 directions. Le déplacement n'est possible que sur une case vide.

Il n'y a pas de prise comme aux échecs ou aux dames mais des combats de pokemons. Lorsqu'une pièce de trouve à côté d'une pièce adverse (selon les 8 directions), lorsque c'est son tour, un joueur peut lancer une attaque sur une pièce adverse. L'attaque suit les règles des combats pokemon : le plus rapide attaque en premier et le second contre attaque s'il est encore vivant. L'attaque s'arrête une fois l'attaque et la contre-attaque lancée.

Le but du jeu

Le vainqueur est le joueur qui arrive à tuer le mewtwo adverse. (ou le dernier joueur à avoir un mewtwo vivant pour l'extension à 4 joueurs).

TODO

Le `Plateau` aura une taille et un ensemble de `Piece` seront posées dessus. Chaque pièce sera représentée par un `Pokemon`, appartiendra à un joueur et aura une `Position` particulière.

La plateau sera donc une collection de pièces. Les pièces seront composées d'un pokemon, d'un identifiant de joueur (un entier pour le numéro) et une position qu'il faudra pouvoir manipuler par ses coordonnées ou son équivalent lettre/numéro. Les pièces sont des pokemons (classe que nous avons) et sont situés sur une case particulière qui est repérée par une position.

La classe `Position`

Cette classe se compose simplement de deux attributs `x` et `y` permettant de stocker les valeurs de composantes. Cette classe est spécifique à ce jeu. Ainsi les valeurs des composantes ne peuvent être comprises qu'entre 0 et 8. Si l'utilisateur essaie de mettre une valeur incohérente, un message d'erreur devra s'afficher et le programme devra s'arrêter.

Développez la classe `Position`. Elle doit comporter :

- Deux attributs privés entiers `x` et `y` permettant de repérer une position particulière. Attention, il ne doit pas être possible d'y mettre autre chose que des chiffres de 0 à 8.
- Un constructeur par défaut qui intialisera les attributs à 0.
- Un constructeur par copie.
- Un constructeur prenant deux entiers en paramètres.
- Un constructeur prenant une chaine de caractère en paramètre. La chaine de caractère en paramètre sera de la forme : lettre en majuscule suivie d'un chiffre (identification d'une case comme sur un échiquier).
- Les getter et setter.

— La méthode `equals` prenant un `Object` en paramètre.
— La méthode `toString` retournant la position sous la forme de du repérage échiquier (type "E4").

N'hésitez pas à tester votre classe en écrivant un programme principal dans cette classe.

La classe `Piece`

Cette classe permet de définir une pièce. Sur ce jeu, les pièces sont des pokemons. Chaque pièce a également un numéro de joueur associé et une position. Toutes les classes ont été écrites. Il est donc possible de développer la classe `Piece`.

Développez la classe `Piece`. Elle doit comporter :
— Trois attributs privés :
  — `pokemon`, un pokemon.
  — `joueur`, un entier pour définir le joueur à qui appartient la pièce.
  — `position`, un objet de type `Position`.
— Un constructeur par défaut qui créera un pokemon par défaut pour le joueur 1 en A1.
— Un constructeur par copie.
— Un constructeur prenant en paramètre le type de la pièce (le pokemon), un numéro de joueur et une position sous la forme de deux entiers.
— Un constructeur prenant en paramètre le type de la pièce (le pokemon), un numéro de joueur et une position sous la forme d'un objet de type `Position`.
— Un constructeur prenant en paramètre le type de la pièce (le pokemon), un numéro de joueur et une position sous la forme d'une chaine de caractère identifiant une case à la manière d'un échiquier (type "A4" ou "E3").
— Les getter et setter (si nécessaire).
— La méthode `equals` prenant un `Object` en paramètre.
— La méthode `toString` retournant une description simplifiée de la pièce du type : "Bulbizarre du joueur 2 en E4".

La classe `Plateau`

Le plateau sera modélisé par un tableau dynamique contenant les pièces.
Cette classe `Plateau` doit comporter :
— Un attribut privé de type tableau dynamique stockant des objets de type `Piece`.
— Un constructeur par défaut initialisant le plateau tel que montré en figure **??**. Ce constructeur ne laisse pas le choix des pokemons au joueur.
— Un constructeur que vous développerez à la fin permettant de prendre en paramètre deux fichiers décrivant les pokemons de départ du joueur. Vous définirez vous-mêmes la structure du fichier.
— Trois méthodes `getCase`. L'une prend en paramètre deux entiers, la seconde un objet de type `Position` et une dernière prenant une chaine de caractère indiquant une position à la manière de l'échiquier ("E4" par exemple). Cette méthode retournera

un objet de type `Piece`. Si une pièce se trouve à la position donnée, cette pièce sera retournée. Si aucune pièce ne se trouve sur la case, la méthode retournera la valeur `null` indiquant ainsi que la case est vide.

— Une méthode `getPiecesJoueur1` retournant un tableau dynamique rempli des pièces du joueur 1 encore présentes sur le plateau.

— Une méthode `getPiecesJoueur2` retournant un tableau dynamique rempli des pièces du joueur 2 encore présentes sur le plateau.

— Une méthode `toString()` qui retourne une chaine de caractères contenant l'ensemble des informations des pièces.

La classe `MainGraphique`

Développer une méthode principale utilisant `MG2D` et affichant le plateau. Vous pouvez utiliser les images fournies. Le nom des images est leur numéro de dans le pokedex suivi de l'extension `png`. Par exemple, l'image `37.png` représente goupix.