SAE R201
Modélisation et implémentation d'un programme arbitrant une partie d'échecs entre 2 joueurs

BOURG Solène & JEGATHASAN Amirtavarsini

Notice d'utilisation du jeu d'échecs

> document associé : *SAE_JAVA_bourg_jegathasan_CERYNIE.zip*

Les classes .java provenant du fichier .zip sont déjà compilées. Vous pouvez donc commencer à jouer dès à présent.

1. Pour lancer le jeu, ouvrez un terminal et entrez la commande ' java Main '.

2. Il vous sera demandé de donner le nom du premier et du deuxième joueur qui joueront respectivement les pièces blanches et noires.

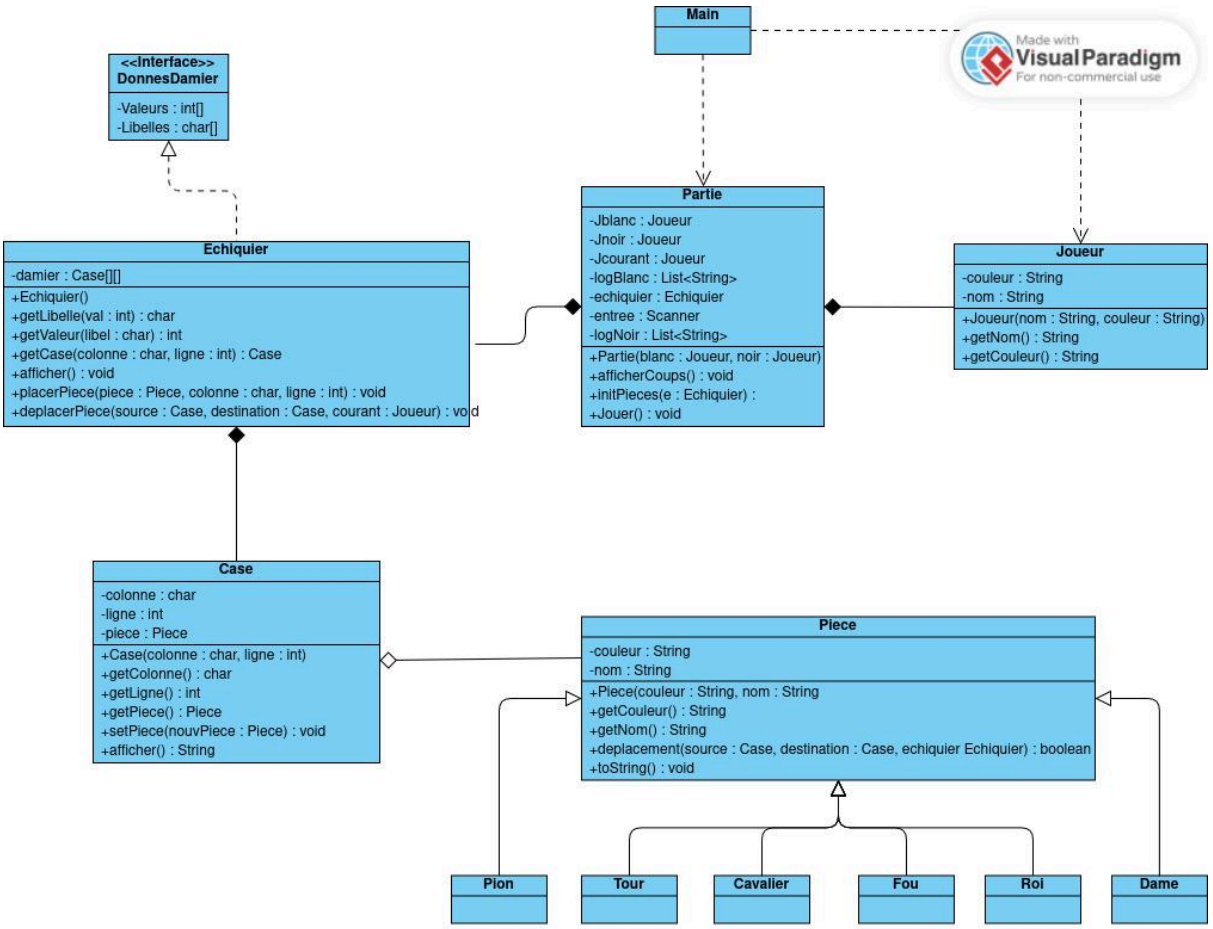
3. Votre échiquier apparaît dans le terminal : les pièces blanches sont positionnées en bas de l'échiquier représentées par des emojis blanches. Les pièces noires sont en haut et représentées par les mêmes émojies mais en noir.
   Le joueur "blanc" commence la partie.

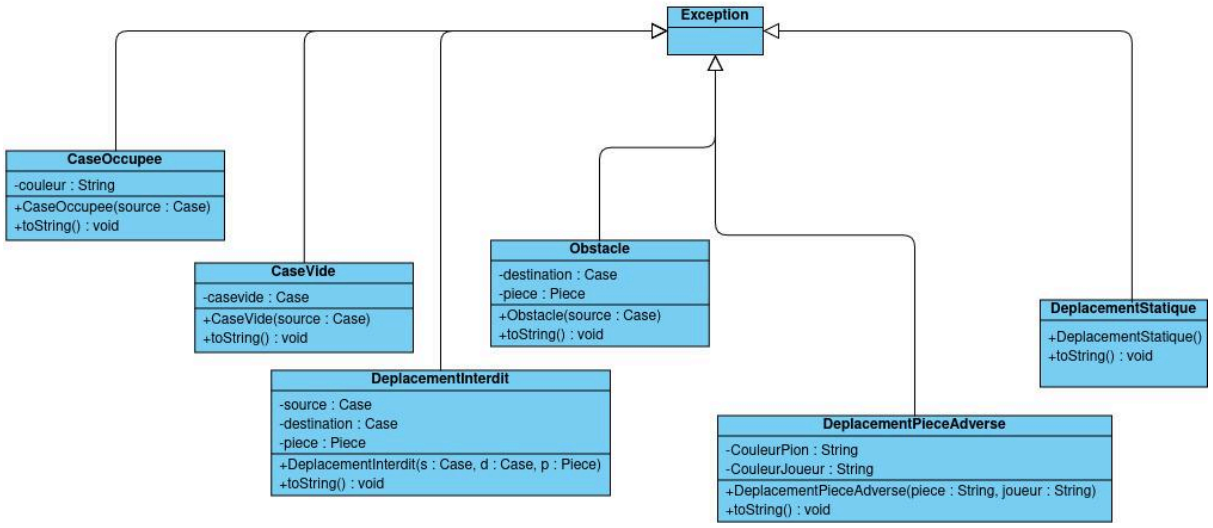
4. Pour déplacer une pièce, entrez les coordonnées de la case ou se trouve la pièce à bouger et faites ENTRER. Puis entrez les coordonnées de la case destination et faites ENTRER. LEs coordonnées doivent être sous le format 'ColonneLigne ' (par exemple: a2 a4).

   (Vous pouvez également entrer les deux coordonnées sur la même ligne en veillant à laisser un espace entre la source et la destination).

5. Si le coup est valide, c'est à l'adversaire de jouer. Sinon, un message d'erreur apparaîtra et vous devrez réessayer.

Diagramme UML du projet





Bilan de l’implémentation

Cette implémentation du jeu d’échecs respecte le déplacement de chaque pièce ainsi que les limites d’un échiquier et lève des exceptions lorsque le joueur ne les respecte pas. 6

exceptions ont été créées pour détecter différentes erreurs au cours du déroulement de la partie :

- CaseVide,
  Les coordonnées sources correspondent à une case sans pièce (la méthode getPiece() de la case renvoie null).

- DeplacementPieceAdverse,
  Le joueur essaie de déplacer une pièce adverse (la sortie de la méthode getCouleur() du joueur au trait ne correspond pas à la sortie de la méthode getCouleur() de la piece à déplacer), les deux couleurs sont effectivement différentes.

- DeplacementInterdit,
  Le déplacement voulu sur une pièce est contraire aux règles (la méthode deplacement() de la pièce renvoie False).

- DeplacementStatique,
  Les coordonnées sources et destinations sont identiques, faire du surplace n'est pas un déplacement valide.

- Obstacle,
  Une pièce est placée entre la source et la destination. Qu'elle soit de la même couleur que le joueur au trait ou non, aucune pièce (à part le Cavalier) ne peut passer par-dessus une pièce.

- CaseOccupee
  La case destination est occupée par une pièce de la même couleur que le joueur au trait.

En plus de celà, les exceptions '*StringIndexOutOfBoundsException'* et '*ArrayIndexOutOfBoundsException'* ont été utilisées pour déterminer si les coordonnées respectent les limites de l'échiquier et que leur format est correct.

L'implémentation enregistre également les coups joués durant la partie et les affichent dans le terminal sous le format ' Pièce Source->Destination ' au-dessus de l'échiquier.

La gestion facultative de la pendule et des tests unitaires ne sont pas implémentés tout comme les conditions de victoires et de défaites. De ce fait, la partie est en réalité une boucle qui ne se termine jamais.
La seule manière d'en sortir est de détruire le processus.

Répartition du travail

- En duo

Compréhension du sujet et définition du diagramme UML sur papier.

- JEGATHASAN Amirtavarsini

Commentaires du code des 19 classes de ce projet.
Classe Echiquier : Constructeur, définition du format graphique de l'échiquier dans le terminal
Classe Pion : définition d'un déplacement valide pour un pion mathématiquement, méthode toString()
Classe Roi : définition d'un déplacement valide pour un roi mathématiquement, méthode toString()
Classe Tour : méthode toString()
Classe Cavalier : méthode toString()
Classe Fou : méthode toString()
Classe Dame : méthode deplacement(), méthode toString()
Classe Joueur : Implémentation totale
Classe Main : Implémentation totale
Classe Case : méthode toString()
Classe Partie : implémentation de la pendule (infructueux)

- BOURG Solène

Implémentation du reste des classes.
Diagramme UML version numérique.