Première partie du PROJET ECHECS

SHAHZAD SAAD

Fonctionnalités de la classe Échiquier :

Je commence par déclarer les variables publiques / privées de ma classe **Echiquier()** , le nombre de lignes et de colonnes de mon tableau ➔ **public static int nbDeLigneEtCol = 8;** ensuite, je déclare un Boolean ➔ **jeuxEnCours ;** qui me permettront de faire durer la partie. Je déclare également une variable privée ➔ **private static int departL, departC, arriveL, arriveC;** qui correspond au départ et a l'arrivée de mes pièces en fonction des lignes et colonnes ce qui me serait utile pour le déplacement de mes pièces. Pour ce qui concerne l'échiquier, j'ai décidé de coder un tableau de char 8X8 avec un affichage de console ainsi qu'une chaine de caractère avec l'entrée utilisateurs pour les instructions de déplacement. ➔ **Mouvement() ;** Les instructions de déplacement sont saisies dans mon scanner ➔ **Scanner saisie = new Scanner(System.in);** qui seront ensuite lu par le programme.

Par la suite, je commence à initialiser mon tableau ➔ **creerTableau** d'une dimension 8X8, je fais une boucle I qui correspond à la colonne du tableau ainsi qu'une boucle J qui correspond à la ligne du tableau. Je remplis mon tableau de case vide en faisait appel à ma classe ➔ **Case0()** avec une boucle for, ensuite a des emplacements bien définis je fais apparaître mon Roi ➔ **new Roi() ;** et la Dame ➔ **new Dame() ;**
J'affiche l'échiquier ➔ **afficherTableau()** avec les numéros 8 à 1 le long des lignes et les lettres A-H (ASCII TABLE) le long des colonnes avec des boucles for , je fais également apparaître le damier en respectant le format demandé dans le sujet avec les différents caractères "---" , "|"

Ma fonction mouvement ➔ **public void mouvement()** me sert à faire déplacer mes pièces. Pour saisir les coordonnés de départ vers les coordonnées d'arrivée par exemples
A1 a A3 ma chaîne de caractère ➔ **String mouvement ;** est divisé avec la méthode split avec un caractère prédéfini ici un espace et analyse ma chaîne en compensant ➔ **composants[0].charAt(0) = 'A'** et **composants[0].charAt(1) = '1'** la méthode **charAt** renvoie le caractère **'A'** à l'index spécifié.
La condition if ➔ **if (mouvValide())** vérifie si le déplacement est possible alors je place ma nouvelle pièce dans sa nouvelle case ➔ **tableau[arriveL][arriveC] = tableau[departL][departC];** et je supprime ma pièce de son ancienne case en plaçant la case vide ➔ **tableau[departL][departC] = new Case0();**
La fonction ➔ **private boolean mouvValide()** permet de recenser les déplacements possibles ou non par exemple les coordonnées saisies sont supérieures et/ou inférieur au tableau (N1 a M1) alors j'affiche un message et je renvoie false sinon je renvoie true. De même , si les coordonnées rentrées ne correspondent pas à la façon d'on se déplace la pièce déclarée dans deplacementOK alors j'affiche un message et je renvoie false sinon je renvoie true.

Fonctionnalités de la classe Pièces :

Ma classe Pièces est une abstract class c'est-à-dire que toute les classes hérités de celle -ci sont forcement surcharger. La fonction ➔ **public String toString()** est utilisé pour afficher le nom de la pièce (R , D).

La deuxième fonction, quant à elle vérifie si le déplacement est valide ➔ **public abstract boolean deplacementOK().**

Fonctionnalités des classes Roi , Dame et Case0 :

Les trois classes suivantes sont toutes héritées de la classe Pieces. Je déclare une variable char **'R'** pour le Roi , **'D'** pour la Dame et **' '** pour la Case0. La fonction ➔ **public String toString ()** renvoie une représentation sous forme de String des valeurs. Ensuite , j'utilise la méthode ➔ **String.valueOf()** pour renvoie la représentation sous forme de chaîne.

Je saisis les coordonnées de déplacement dans la fonction hérité ➔ **public boolean deplacementOK()** pour chaque classe par exemple dans la classe Roi ➔ **public boolean deplacementOK(int departL, int departC, int arriveL, int arriveC) {**
return Math.abs(arriveL - departL) <= 1 && Math.abs(arriveC - departC) <= 1;
Si la différence entre les coordonnées arriveL departL et arriveC departC est 0 ou 1 je peux bouger mon roi.

Fonctionnalités de la classe Main :

J'instancie un nouvel échiquier que j'appelle mon échiquier ➔ **Echiquier monEchiquier = new Echiquier()** ensuite je demande au plateau de m'afficher le tableau ➔ **monEchiquier.afficherTableau();.** Pour finir je fais une boucle while tant que le jeu est en cours d'afficher le tableau ainsi que le Scanner ➔ **mouvement()**.

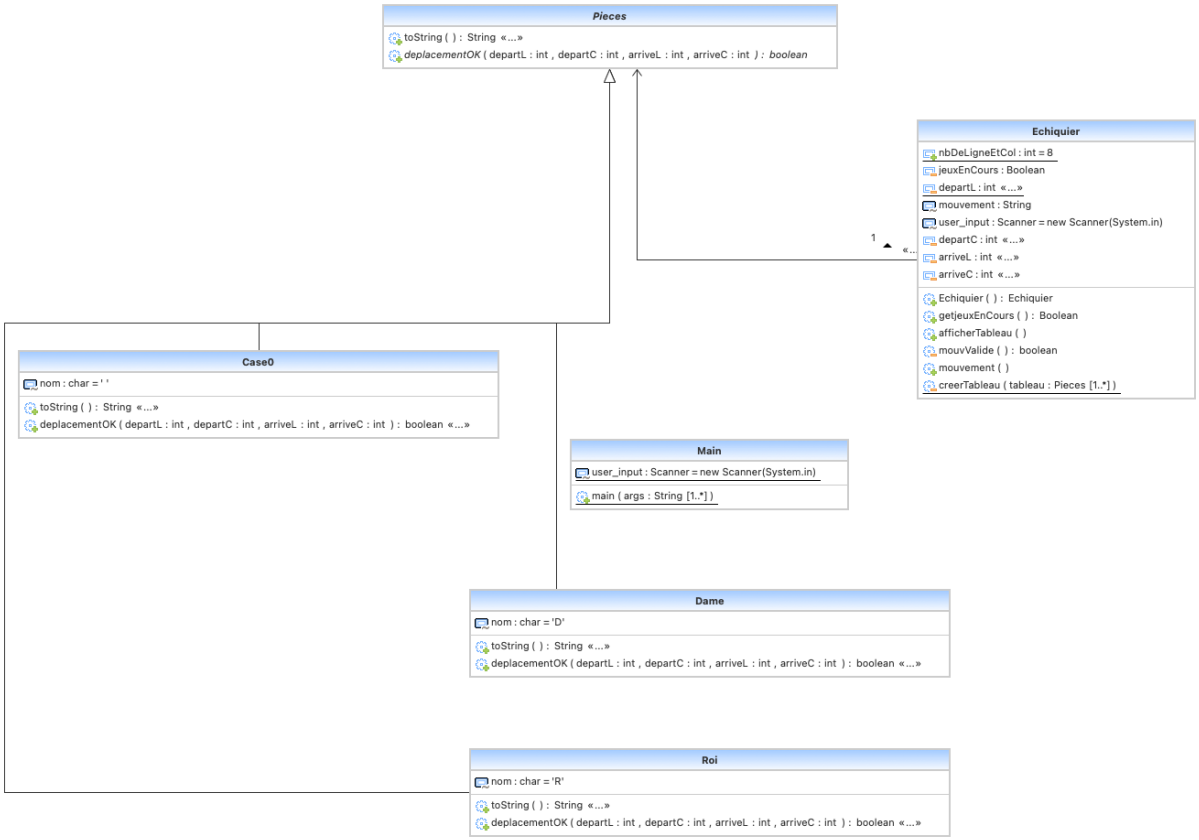

Diagramme de l'architecture (UML)