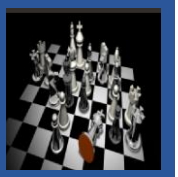
RAPPORT

Documentation détaillée du code et guide d'utilisation.



J'ai choisi comme nom du jeu AfoRA-Echec-Game

(Afo Mon nom en famille, premier garçon).

L'entete du jeu .

Guide d'utilisation

Installation et lancement

1- Prerequis :
   Java JDK 11 ou supérieur installe sur votre machine
   Environnement de développements (Eclipse).
2- Compilation du projet

   Depuis le répertoire recine du projet :

   mkdir -Force bin (en cas de probleme d'execution si non passe au directement au point 3)

   javac -d bin src\com\chess\Main.java src\com\chess\gui\EcranAccueil.java src\com\chess\gui\ChessGUI.java src\com\chess\model*.java src\com\chess\ai\ChessAI.java

3- Exécution du jeu

   Depuis le répertoire racine du projet

   java -cp bin com.chess.Main

```
C:\Users\USER\Desktop\Examen-Java_POO\ChessSimulator>java -cp bin com.chess.MainGUI
java.lang.NullPointerException
        at java.base/java.util.Objects.requireNonNull(Objects.java:233)
        at java.desktop/javax.sound.sampled.AudioSystem.getAudioInputStream(AudioSystem.java:1030)
        at com.chess.gui.ChessGUI.jouerSon(ChessGUI.java:483)
        at com.chess.gui.ChessGUI.effectuerCoup(ChessGUI.java:398)
        at com.chess.gui.ChessGUI.gererClic(ChessGUI.java:274)
        at com.chess.gui.ChessGUI.lambda$0(ChessGUI.java:92)
        at java.desktop/javax.swing.AbstractButton.fireActionPerformed(AbstractButton.java:1972)
        at java.desktop/javax.swing.AbstractButton$Handler.actionPerformed(AbstractButton.java:2314)
        at java.desktop/javax.swing.DefaultButtonModel.fireActionPerformed(DefaultButtonModel.java:407)
        at java.desktop/javax.swing.DefaultButtonModel.setPressed(DefaultButtonModel.java:262)
        at java.desktop/javax.swing.plaf.basic.BasicButtonListener.mouseReleased(BasicButtonListener.java:279)
        at java.desktop/java.awt.AWTEventMulticaster.mouseReleased(AWTEventMulticaster.java:297)
        at java.desktop/java.awt.Component.processMouseEvent(Component.java:6621)
        at java.desktop/javax.swing.JComponent.processMouseEvent(JComponent.java:3398)
        at java.desktop/java.awt.Component.processEvent(Component.java:6386)
```

INTERFACE D'ACCUEIL

L'écran d'accueil vous permet de configurer votre partie :

1. Mode de jeu

   2 Joueurs : Jouer contre un autre joueur sur le même ordinateur

   Contre l'ordinateur : Jouer contre l'IA

2. Niveau de difficulté (pour le mode contre l'ordinateur)

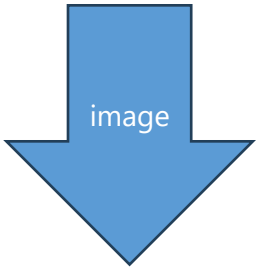
   Facile : IA avec une profondeur d'analyse de 2 coups

   Moyen : IA avec une profondeur d'analyse de 3 coups

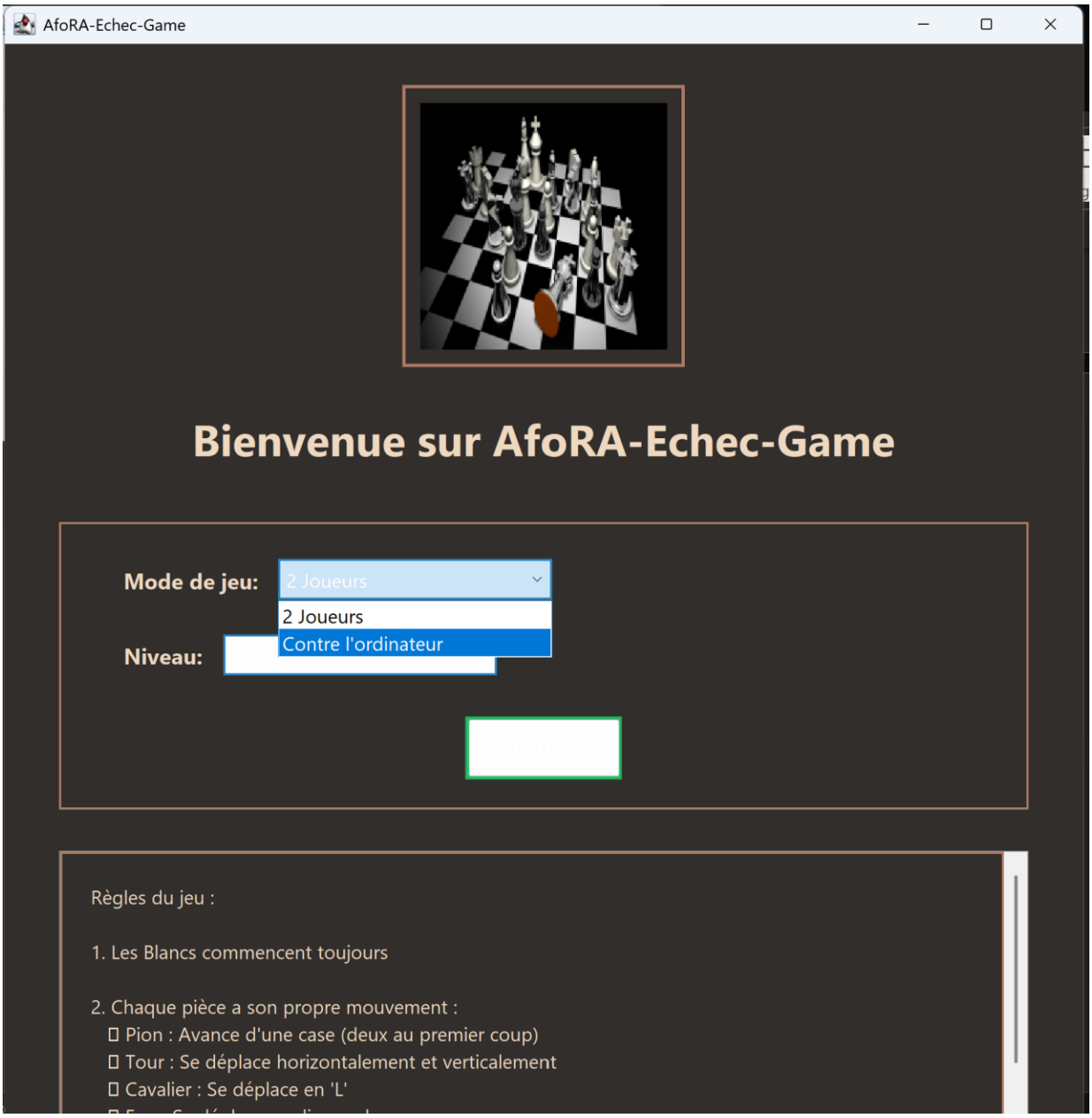
   Difficile : IA avec une profondeur d'analyse de 4 coups

3. Bouton JOUER

   - Cliquez pour lancer la partie avec les paramètres sélectionnés



AFORA-ECHECS-GAME



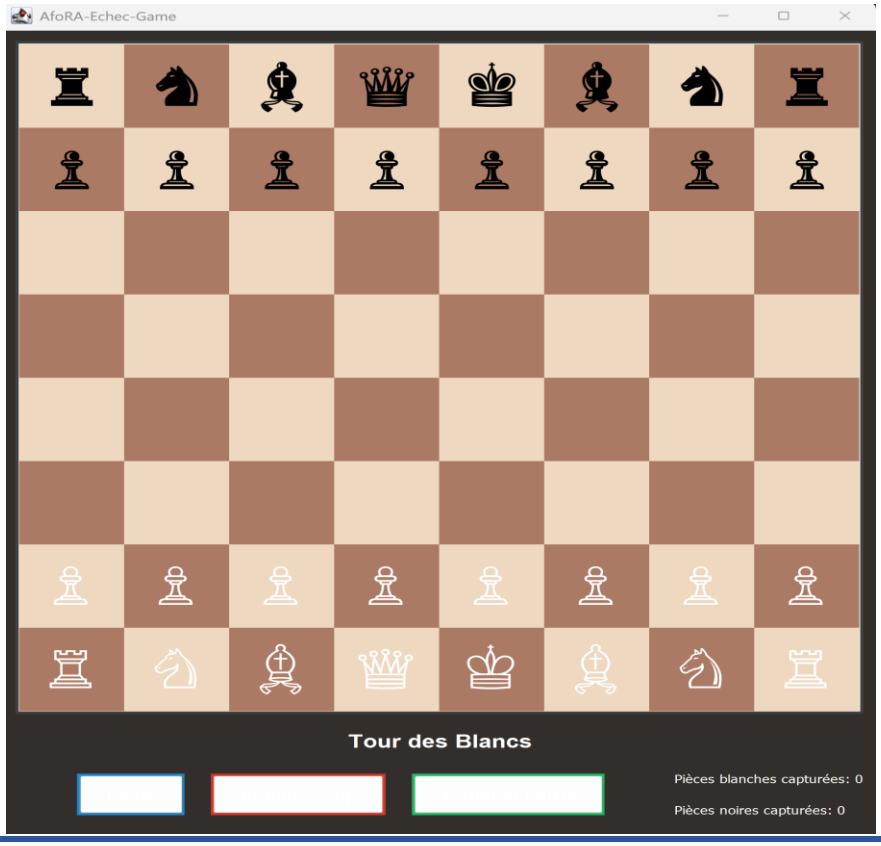
INTERFACE DE JEU

1. Plateau d'échecs
   - Les pièces blanches commencent toujours
   - Cliquez sur une pièce pour la sélectionner (elle sera surlignée en vert)
   - Les cases de destination possibles seront surlignées en jaune
   - Cliquez sur une case de destination pour déplacer la pièce

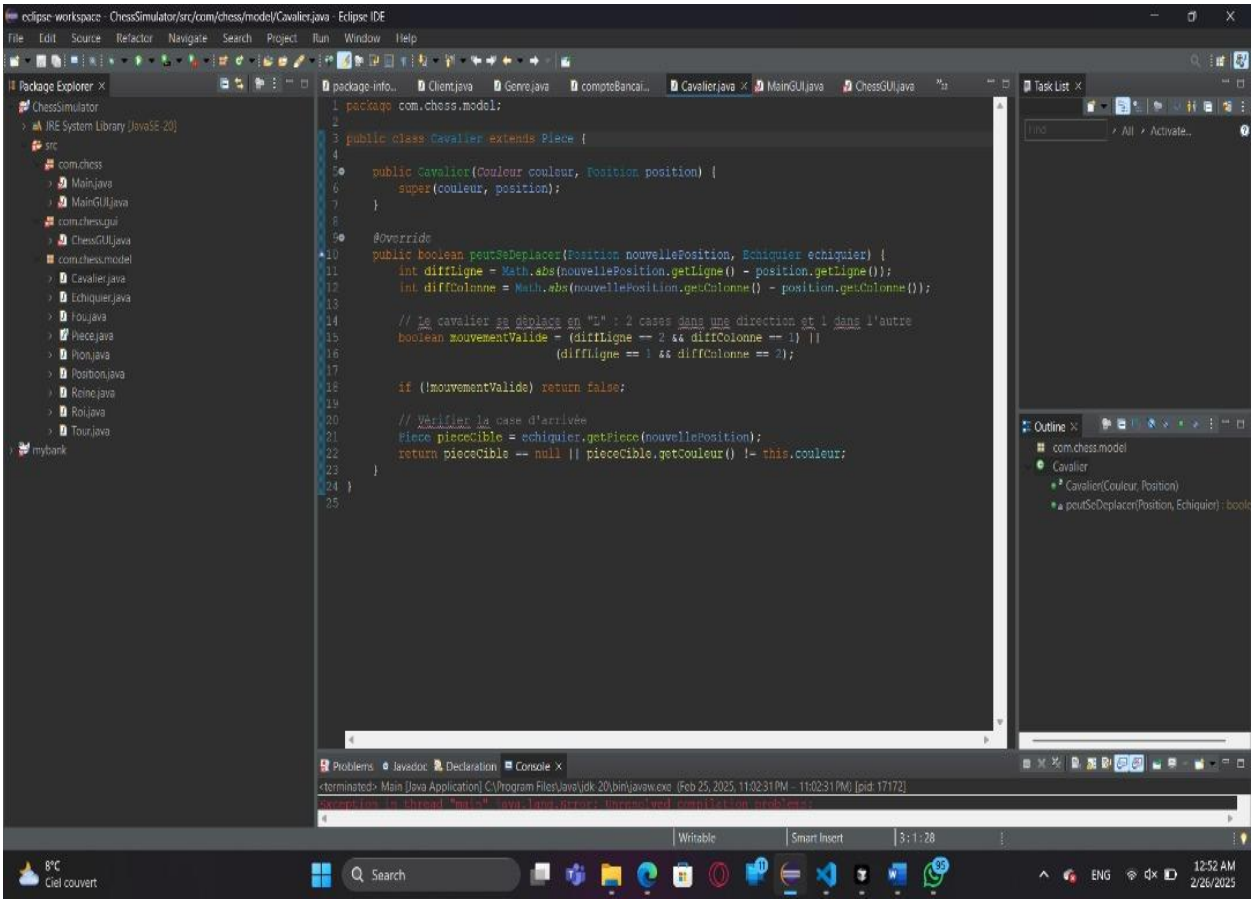
2. Contrôles de jeu
   - Pause : Met la partie en pause (En bleu)
   - Annuler coup : Annule le dernier coup joué (En rouge)
   - Terminer partie : Quitte la partie et retourne à l'écran d'accueil (En vert)

3. Informations
   - Le joueur actuel est indiqué en haut
   - Le nombre de pièces capturées est affiché sur le côté

4. Fin de partie
   - Un message s'affiche en cas d'échec et mat ou de pat
   - Vous pouvez alors retourner à l'écran d'accueil



Documentation technique



STRUCTURE DU PROJET :

Package com.chess

-*Main.java : Point d'entrée de l'application

- main(String[] args) : Initialise l'interface graphique

Package com.chess.gui

- EcranAccueil.java : Interface d'accueil du jeu

- Attributs principaux :

  - modeJeuCombo : Sélection du mode de jeu

  - niveauDifficulteCombo : Sélection du niveau de difficulté

- Méthodes principales :

  - initComposants() : Initialise les composants graphiques

  - createStyledComboPanel() : Crée les menus déroulants stylisés

  - styleButton() : Applique le style aux boutons

  - demarrerPartie() : Lance une nouvelle partie

- ChessGUI.java : Interface principale du jeu

- Attributs principaux :

  - echiquier : Référence au modèle d'échiquier

  - casesBoutons : Tableau des boutons représentant les cases

  - positionSelectionnee : Position de la pièce sélectionnée

  - joueurActuel : Joueur dont c'est le tour

- Méthodes principales :

  - initGUI() : Initialise l'interface du jeu

  - actualiserEchiquier() : Met à jour l'affichage

  - gererClic() : Gère les clics sur l'échiquier

  - surlignerCasesValides() : Surligne les déplacements possibles

  - effectuerCoup() : Exécute un mouvement

  - jouerCoupIA() : Fait jouer l'IA

Package com.chess.model

- Piece.java (classe abstraite)
  - Attributs :
    - position : Position actuelle de la pièce
    - couleur : Couleur de la pièce (BLANC/NOIR)
  - Méthodes :
    - peutSeDeplacer() : Vérifie si un déplacement est légal
    - getPosition(), setPosition() : Accesseurs
    - getCouleur() : Retourne la couleur
- *Classes des pièces* (héritent de Piece)
  - Pion.java, Tour.java, Cavalier.java, Fou.java, Reine.java, Roi.java
  - Chacune implémente sa propre logique de déplacement
- *Echiquier.java* : Gestion du plateau
  - Attributs :
    - pieces : Tableau 2D contenant les pièces
  - Méthodes :
    - initialiser() : Place les pièces
    - deplacerPiece() : Déplace une pièce
    - getPiece() : Récupère une pièce à une position
    - estPositionMenacee() : Vérifie si une position est menacée
- *Position.java* : Représente une position sur l'échiquier
  - Attributs :
    - ligne : Numéro de ligne (0-7)
    - colonne : Numéro de colonne (0-7)
- EtatPartie.java : Gère l'état de la partie
  - Méthodes :
    - verifierEtat() : Vérifie l'état actuel (échec, mat, pat)
    - estEnEchec() : Vérifie si un roi est en échec
    - estEnEchecEtMat() : Vérifie s'il y a échec et mat
    - estPat() : Vérifie s'il y a pat

Règles du jeu d'échecs

1. Déplacement des pièces :

- Pion : Avance d'une case (deux au premier coup), capture en diagonale
- Tour : Se déplace horizontalement et verticalement
- Cavalier : Se déplace en 'L' (2 cases puis 1 perpendiculairement)
- Fou : Se déplace en diagonale
- Reine : Combine les mouvements de la tour et du fou
- Roi : Se déplace d'une case dans toutes les directions

2. Règles spéciales :

- Promotion du pion : Un pion qui atteint la dernière rangée est promu
- Échec : Le roi est menacé par une pièce adverse
- Échec et mat : Le roi est en échec et aucun coup ne peut l'en sortir
- Pat : Le joueur ne peut jouer aucun coup légal mais son roi n'est pas en échec

3. Fin de partie :

- Victoire par échec et mat
- Match nul par pat
- Match nul par accord mutuel en mettant fin au jeu manuellement

4. Notes de développement

- L'interface utilise Swing avec un look and feel moderne
- L'IA utilise l'algorithme Negamax avec élagage Alpha-Bêta
- Les images et sons sont chargés depuis le dossier resources/
- Le projet suit une architecture MVC (Modèle-Vue-Contrôleur)

Conclusion

AfoRA-Echec-Game est un projet **100% Java**, développé sans framework, en appliquant les principes fondamentaux de la **programmation orientée objet (POO)**. Il offre une expérience immersive avec une **interface graphique intuitive**, un **système d'IA performant** et une **architecture bien structurée** suivant le modèle **MVC**.

Grâce à son **moteur d'échecs basé sur l'algorithme Negamax**, le jeu propose un défi adapté à tous les niveaux, du débutant au joueur avancé. Son **code modulaire et optimisé** permet une évolution facile, que ce soit pour enrichir l'IA, améliorer l'interface ou ajouter de nouvelles fonctionnalités.

Ce projet démontre ainsi la puissance et la flexibilité de Java pour développer des applications interactives sans dépendre de frameworks externes. Il constitue une excellente base pour quiconque souhaite approfondir ses compétences en Java, en intelligence artificielle et en développement d'applications graphiques.