DOCUMENTATION DU PROJET PUISSANCE 4 EN JAVA

Sommaire

1. Introduction

Le projet **Puissance 4** est une implémentation du jeu classique en Java. Le jeu se joue à deux joueurs sur une grille de 7 colonnes et 6 lignes. Chaque joueur place à tour de rôle un pion dans une colonne, et le premier à aligner 4 pions horizontalement, verticalement ou en diagonale gagne la partie.

Ce document explique en détail la conception, l'implémentation et le fonctionnement du programme.

2. Règles du jeu

- La grille de jeu est de taille 7 colonnes × 6 lignes.
- Deux joueurs s'affrontent : Joueur 1 (o) et Joueur 2 (x).
- Chaque joueur place un pion dans une colonne de son choix. Le pion tombe au fond de la colonne ou sur le premier pion rencontré.
- Le premier joueur à aligner 4 pions (horizontalement, verticalement ou en diagonale) gagne la partie.
- Si la grille est remplie sans qu'aucun joueur n'ait aligné 4 pions, la partie se termine par un match nul.

3. Structure du projet

Le projet est structuré autour d'une seule classe principale Puissance4, qui contient toutes les méthodes nécessaires pour gérer le jeu. Voici les composants principaux :

- **Initialisation de la grille** : Création d'une grille vide.
- **Affichage de la grille** : Visualisation de l'état actuel du jeu.
- **Jouer un coup** : Placement d'un pion dans une colonne.
- **Vérification de la victoire** : Détection d'un alignement de 4 pions.
- **Alternance des joueurs** : Passage du tour entre les deux joueurs.

4. Explication du code

4.1 Initialisation de la grille

La grille est représentée par un tableau à deux dimensions int[][] grid, où :

- 0 représente une case vide.
- 1 représente un pion du Joueur 1 (o).
- 2 représente un pion du Joueur 2 (x).

```
private void initializeGrid() {
    for (int col = 0; col < COLS; col++) {
        for (int row = 0; row < ROWS; row++) {
            grid[col][row] = EMPTY; // Initialisation à 0 (case vide)
        }
    }
}
```

4.2 Affichage de la grille

La méthode printGrid() affiche la grille en utilisant des symboles :

- . pour une case vide.
- o pour un pion du Joueur 1.
- x pour un pion du Joueur 2.

```
private void printGrid() {
    for (int row = 0; row < ROWS; row++) {
        for (int col = 0; col < COLS; col++) {
            switch (grid[col][row]) {
                case EMPTY:  System.out.print(". "); break;
                case PLAYER1: System.out.print("o "); break;
                case PLAYER2: System.out.print("x "); break;
            }
        }
        System.out.println();
    }
    System.out.println();
}
```

4.3 Jouer un coup

La méthode dropPiece(int col, int player) permet de placer un pion dans une colonne. Le pion tombe au fond de la colonne ou sur le premier pion rencontré.

```
private boolean dropPiece(int col, int player) {
    for (int row = ROWS - 1; row >= 0; row--) {
        if (grid[col][row] == EMPTY) {
            grid[col][row] = player; // Placement du pion
            return true;
        }
    }
    return false; // Colonne pleine
}
```

4.4 Vérification de la victoire

La méthode checkWin(int player) vérifie si un joueur a aligné 4 pions horizontalement, verticalement ou en diagonale.

```
private boolean checkWin(int player) {
    for (int col = 0; col < COLS; col++) {
        for (int row = 0; row < ROWS; row++) {
            if (grid[col][row] == player) {
                if (checkDirection(col, row, 1, 0, player) || // Horizontal
                    checkDirection(col, row, 0, 1, player) || // Vertical
                    checkDirection(col, row, 1, 1, player) || // Diagonal down-right
                    checkDirection(col, row, 1, -1, player)) { // Diagonal up-right
                    return true;
                }
            }
        }
    }
    return false;
}
```

La méthode checkDirection(int col, int row, int colDir, int rowDir, int player) vérifie un alignement dans une direction donnée.

```
private boolean checkDirection(int col, int row, int colDir, int rowDir, int player) {
    int count = 0;
    for (int i = 0; i < 4; i++) {
        int newCol = col + i * colDir;
        int newRow = row + i * rowDir;
        if (newCol >= 0 && newCol < COLS && newRow >= 0 && newRow < ROWS && grid[newCol][newRo
w] == player) {
            count++;
        } else {
            break;
        }
    }
    return count == 4;
}
```

4.5 Alternance des joueurs

La méthode switchPlayer() alterne les tours entre les deux joueurs.

```
private void switchPlayer() {
    currentPlayer = (currentPlayer == PLAYER1) ? PLAYER2 : PLAYER1;
}
```

5. Fonctions principales

- **initializeGrid()** : Initialise la grille vide.
- **printGrid()** : Affiche la grille.
- **dropPiece(int col, int player)** : Place un pion dans une colonne.
- **checkWin(int player)** : Vérifie si un joueur a gagné.
- **switchPlayer()** : Alterne les tours entre les joueurs.
- **play()** : Gère le déroulement du jeu.

7. Conclusion

Ce projet est une implémentation simple mais complète du jeu Puissance 4 en Java. Il peut être étendu avec des fonctionnalités supplémentaires pour améliorer l'expérience utilisateur. Le code est modulaire et facile à comprendre, ce qui permet d'ajouter des extensions sans difficulté.

Annexes

- **Code source complet** : Voir le fichier Puissance4.java.
- **README** : Instructions pour compiler et exécuter le programme.