Match nul et améliorations

Temps de lecture : 2 minutes

Match nul

Classe `TicTacToe`

Ajout de la méthode `checkDraw` :

```
public boolean checkDraw() {
    for (char[] line : grid) {
        for (char cell : line) {
            if (cell == '.') {
                return false;
            }
        }
    }
    return true;
}
```

Si au moins un caractère `'.'` est encore présent, on renvoie false. Si aucun `'.'` n'est trouvé, alors on renvoie true, ce qui correspond à un match nul.

Classe `Main`

```
public class Main {
    public static void main(String[] args) {

        final var scanner = new Scanner(System.in);
        final var game = new TicTacToe();

        var player = Player.FIRST;

        while (true) {
            System.out.println(game);
            System.out.println("Veuillez saisir un des chi
```

```
ffres [1-9] :");
            final var playerInput = scanner.nextInt();

            game.processInput(player, playerInput);
            if (game.checkWin()) {
                System.out.println(game);
                System.out.println("Le joueur " + player +
" a gagné la partie ! :");
                break;
            }
            if (game.checkDraw()) {
                System.out.println(game);
                System.out.println("Match nul. Personne
n'a gagné !");
                break;
            }

            player = nextPlayer(player);
        }
    }

    private static Player nextPlayer(Player player) {
        if (player.equals(Player.FIRST)) {
            return Player.SECOND;
        } else {
            return Player.FIRST;
        }
    }
}
```

On ajoute l'appel à `checkDraw` dans l'algorithme du jeu. Si la méthode renvoie true, alors on quitte le jeu et on affiche le match nul.

Aller plus loin

Voici quelques missions à réaliser :

1. Ajouter de la gestion d'erreurs pour s'assurer que les joueurs font bien la saisie de nombres entre 1 et 9.
2. Ajouter de la gestion d'erreurs si la case a déjà été jouée par un joueur.
3. Donnez la possibilité de quitter le jeu en cours de partie en entrant par exemple `quit` ou `exit`.

4. Laissez la possibilité aux joueurs de rentrer leur nom avant le début du jeu.

Voici un exemple de solution :

Classe TicTacToe

```
public class TicTacToe {

    private char[][] grid = new char[][]{
        {'.', '.', '.'},
        {'.', '.', '.'},
        {'.', '.', '.'}
    };

    public void processInput(Player player, int playerInpu
t) throws TicTacToeInvalidInputException {
        final var row = (playerInput - 1) / 3;
        final var column = (playerInput - 1) % 3;
        if (grid[row][column] == '.') {
            if (player.equals(Player.FIRST)) {
                grid[row][column] = 'X';
            } else {
                grid[row][column] = 'O';
            }
        } else {
             throw new TicTacToeInvalidInputException("La
case est déjà occupée");
        }
    }

    public boolean checkWin() {
        for (int i = 0; i < 3; i++) {
            var checkWinLine = grid[i][0] == grid[i][1] &&
grid[i][1] == grid[i][2] && grid[i][2] != '.';
            var checkWinColumn = grid[0][i] == grid[1][i]
&& grid[1][i] == grid[2][i] && grid[2][i] != '.';
            if (checkWinLine || checkWinColumn) {
                return true;
            }
        }
        var checkWinDiagonal1 = grid[0][0] == grid[1][1] &
& grid[1][1] == grid[2][2] && grid[2][2] != '.';
        var checkWinDiagonal2 = grid[0][2] == grid[1][1] &
```

```
        & grid[1][1] == grid[2][0] && grid[2][0] != '.';
        if (checkWinDiagonal1 || checkWinDiagonal2) {
            return true;
        }
        return false;
    }

    public boolean checkDraw() {
        for (char[] line : grid) {
            for (char cell : line) {
                if (cell == '.') {
                    return false;
                }
            }
        }
        return true;
    }

    @Override
    public String toString() {
        final var builder = new StringBuilder();
        builder.append("Grille du Morpion : ").append(LINE
_SEPARATOR);
        for (char[] line : grid) {
            for (char cell : line) {
                builder.append(SPACE).append(cell).append
(SPACE);
            }
            builder.append(LINE_SEPARATOR);
        }
        return builder.toString();
    }
}
```

Classe `Main`

```
public class Main {
    public static void main(String[] args) {

        final var game = new TicTacToe();
        var player = Player.FIRST;
```

```
        final var players = initPlayers();
        System.out.println(game);

        while (true) {
            System.out.println("Joueur : " + players.get(p
layer) + " / Veuillez saisir un des chiffres [1-9] :");
            try {
                final var playerInput = scanInput();
                game.processInput(player, playerInput);
                System.out.println(game);
                if (game.checkWin()) {
                    System.out.println("Le joueur " + play
ers.get(player) + " a gagné la partie ! :");
                    break;
                }
                if (game.checkDraw()) {
                    System.out.println("Match nul. Personn
e n'a gagné !");
                    break;
                }
                player = nextPlayer(player);
            } catch (TicTacToeInvalidInputException e) {
                System.out.println("Le nombre saisi doit ê
tre entre 1 et 9");
            } catch (Exception e) {
                System.out.println("Un nombre entier doit
être saisi");
            }
        }
    }

    private static HashMap<Player, String> initPlayers() {
        var players = new HashMap<Player, String>();
        final var scanner = new Scanner(System.in);
        do {
            System.out.println("Nom du joueur 1 :");
            players.put(Player.FIRST, scanner.nextLine());
        } while (players.get(Player.FIRST).equals(BLANK));
        do {
            System.out.println("Nom du joueur 2 :");
            players.put(Player.SECOND, scanner.nextLine
());
        } while (players.get(Player.SECOND).equals(BLAN
```

```
K));
        return players;
    }

    private static Integer scanInput() throws TicTacToeInv
alidInputException {
        final var scanner = new Scanner(System.in);
        final var input = scanner.nextLine();
        if (input.equals("exit") || input.equals("quit"))
{
            System.out.println("Vous quittez la partie");
            System.exit(0);
        }
        final var inputInt = Integer.parseInt(input);
        if (inputInt < 1 || inputInt > 9) throw new TicTac
ToeInvalidInputException("Le chiffre doit être entre 1 et
9");
        return inputInt;
    }

    private static Player nextPlayer(Player player) {
        if (player.equals(Player.FIRST)) {
            return Player.SECOND;
        } else {
            return Player.FIRST;
        }
    }
}
```

Classe TicTacToeInvalidInputException

```
public class TicTacToeInvalidInputException extends Except
ion {}
```

Classe StringConstant

```
public class StringConstant {

    public static final String LINE_SEPARATOR = System.lin
eSeparator();
```

```
        public static final String SPACE = " ";
        public static final String BLANK = "";
    }
```