Package edu.cmu.cs780.hw3

Class Connect4

java.lang.Object
    edu.cmu.cs780.hw3.Connect4

```
public class Connect4
extends Object
```

Represents the game logic and state for Connect4.

Connect4 is a two-player connection game in which the players first choose a color and then take turns dropping one colored disc from the top into a seven-column, six-row vertically suspended grid. The objective of the game is to connect four of one's own discs of the same color next to each other vertically, horizontally, or diagonally before the opponent.

This class provides methods to:

- Initialize a new game session.
- Handle player moves and input.
- Check for game-winning conditions.
- Determine the current state of the game board.
- Display the current player.
- Display the current state of the game board.

Example usage:

```
// Start a new game
Connect4 game = new Connect4();

// Get the current player
int currentPlayer = game.getCurrentPlayer();
System.out.println("It's " + currentPlayer + "'s turn.");

// Make a move in the third column
game.placeChecker(2);

// Print the current game board
game.toString();

// Displays the current game status
// If the game has not yet ended, it indicates which player's turn it is to play
displayGameStatus();

// Check if the game is over
if (game.isGameOver()) {
    // If the game has concluded, the game status (either a win or a draw) is displayed
    displayGameStatus();
}
```

Author:

Zihao Yang (zihaoy2@andrew.cmu.edu), Yun Lee (yunl3@andrew.cmu.edu)

Constructor Summary

Constructors

Constructor	Description
`Connect4()`	Initializes the game with a randomly chosen starting player and an empty board.
`Connect4(int startPlayer)`	Initializes the game with a specified starting player and an empty board.

Method Summary

All Methods | **Instance Methods** | **Concrete Methods**

Modifier and Type	Method	Description
`void`	`displayGameStatus()`	Prints out the current game status to the console.
`int`	`getCurrentPlayer()`	Retrieves the current player's identifier.
`boolean`	`isGameOver()`	Determines if the game has reached a conclusion, either due to a win condition or a draw.
`boolean`	`placeChecker(int columnNum)`	Appends a checker to the specified column.
`String`	`toString()`	Transforms the board into a `String` to be printed in the terminal.

Methods inherited from class java.lang.Object

`clone`, `equals`, `finalize`, `getClass`, `hashCode`, `notify`, `notifyAll`, `wait`, `wait`, `wait`

Constructor Details

Connect4

```
public Connect4()
```

Initializes the game with a randomly chosen starting player and an empty board. The board is represented as a 2D array of integers, where:

- 0 represents an empty cell.
- 1 represents a checker from player 1.
- 2 represents a checker from player 2.

Connect4

```
public Connect4(int startPlayer)
```

Initializes the game with a specified starting player and an empty board. The board is represented as a 2D array of integers, where:

- 0 represents an empty cell.
- 1 represents a checker from player 1.
- 2 represents a checker from player 2.

Parameters:

`startPlayer` - The number of player id to start the game. Valid values are 1 or 2.

Throws:

`IllegalArgumentException` - if the `startPlayer` is neither 1 nor 2.

Method Details

isGameOver

```
public boolean isGameOver()
```

Determines if the game has reached a conclusion, either due to a win condition or a draw.

The game is considered to be over under the following conditions:

- A player forms a horizontal, vertical, or diagonal line of 4 checkers.
- The board is full, signifying a draw, even if there isn't a distinct winner.

The method returns `true` if any of the above conditions are met, indicating the game's conclusion. Otherwise, it returns `false`, implying the game can still proceed.

Returns:

`true` if the game has reached a conclusion, either due to a win condition or a draw; `false` if the game can continue.

displayGameStatus

```
public void displayGameStatus()
```

Prints out the current game status to the console.

getCurrentPlayer

```
public int getCurrentPlayer()
```

Retrieves the current player's identifier.

Returns:

An integer representing current player (either 1 or 2).

toString

```
public String toString()
```

Transforms the board into a `String` to be printed in the terminal.

Overrides:

`toString` in class `Object`

Returns:

a `String` of formulated board.

placeChecker

```
public boolean placeChecker(int columnNum)
```

Appends a checker to the specified column. If the chosen column is valid (not full), the current player's checker is added to the game board, the role is switched to the other player, and true is returned. If the chosen column is invalid or full, appropriate exceptions are thrown.

Parameters:

`columnNum` - the index-based column number (0 to 6 inclusive) where the current player wants to place their checker.

Returns:

true if the checker was successfully placed, otherwise exceptions are thrown.

Throws:

`IndexOutOfBoundsException` - if the columnNum is outside the valid range.

`IllegalArgumentException` - if the chosen column is already full.