Tic-Tac-Toe

Objective

Tic-Tac-Toe is a staple of childhood games. The rules are simple and the logic that governs the game is straightforward. This project helps you to create a functioning version of Tic-Tac-Toe that can be played by two humans. The graphical part of the project has been taken care of for you; your focus will be to develop the logic that governs the gameplay of the grid (a 2D array). By the end of the project you will be proficient at implementing methods that access and evaluate conditionals with elements in a 2D array.

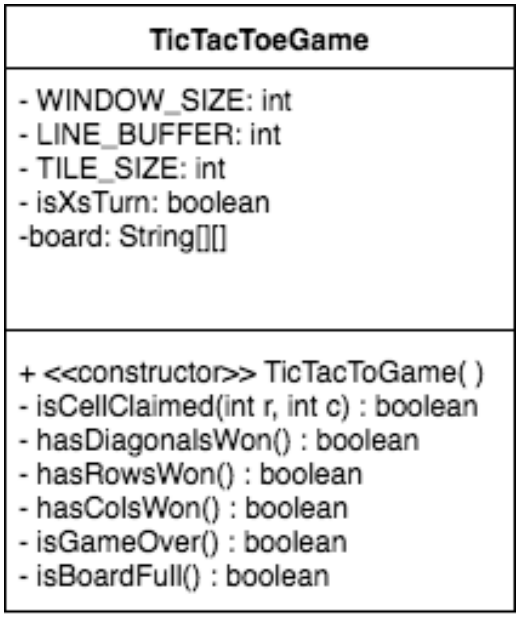
Collaboration

This assignment is to be completed in pairs with a Housemate and you are to employ the Pair Programming technique. You and your partner are to honor the Collaboration Policy throughout this assignment. For this project, you may choose your own partner. **You will need to join a group on Canvas**.

Project Setup

Go to the Unit 6 page on Canvas and download the `TicTacToeStarterCode.zip` folder and place its contents inside the `AP CS/Unit 6 - 2D Arrays/TicTacToe Project` directory. When you unzip `TicTacToeStarterCode`, you'll see that there is one Java file in the folder--the `TicTacToeGame` class, and a `README.txt`. You'll implement five methods to develop a functioning game of Tic-Tac-Toe in `TicTacToeGame`. To play a game of Tic-Tac-Toe, construct a `TicTacToeGame` object and call the `playAGame` method.

Project Structure



The UML diagram should help you get an understanding of how the project is organized.

The first section lists all of the fields in the class.

The second region has methods that you will need to implement.

The class already has some methods that I wrote for you. **Do not modify or alter these methods**. EVER. LIKE AT ALL. K THANKS. Those methods are responsible for handling the graphics and the user interacting with the game board.

TIC-TAC-TOE

Project Structure

- **Fields**
  - `isXsTurn` : a `boolean` that is `true` when the active turn belongs to Player X, and `false` otherwise
  - `board` : a 2D String array of size 3 by 3 that represents the markings of a Tic-Tac-Toe board.
    - If the square has been claimed by Player X, its value will be the String literal "`X`".
    - If the square has been claimed by Player O, its value will be the String literal "`O`".
    - If the square has not yet been claimed by either player, then it will be the default uninstantiated String value, `null`. Remember that! Calling methods on `null` objects results in a `NullPointerException`.
- **Methods**
  - `public booolean isCellClaimed(int r, int c)`
    - Returns `true` if `board[r][c]` is claimed by either Player X or Player O, and returns `false` otherwise.
  - `public booolean hasDiagonalsWon()`
    - Returns `true` if a diagonal of `board` has been claimed by Player X or Player O. The method returns `false` otherwise.
  - `public booolean hasRowsWon()`
    - Returns `true` if a row of `board` has been claimed by Player X or Player O. The method returns `false` otherwise.
  - `public booolean hasColsWon()`
    - Returns `true` if a column of `board` has been claimed by Player X or Player O. The method returns `false` otherwise.
  - `public booolean isGameOver()`
    - Returns `true` when the game has finished. A game of Tic-Tac-Toe is finished when either a player has won or the game has resulted in a tie. `isGameOver()` should call other methods in `TicTacToeGame` to determine if the game is finished. This can be accomplished in one single Java statement.
  - `public booolean isBoardFull()`
    - Returns `true` if all spaces in `board` have been claimed. If a square is still unoccupied, `isBoardFull` returns `false`. You need not concern yourself with identifying if the game will end in a tie before all 9 squares have been played.

Tic-Tac-Toe

1. By reading the UML diagram, identify the name of the variable that stores the Tic-Tac-Toe board. What is its data type?

2. What are the three types of values that can be assigned to a cell in the game board?

3. Consider the `hasRowsWon` method. Identify the three cases in which this method returns `true`. What condition(s) must be met for `hasRowsWon` to return `true`?

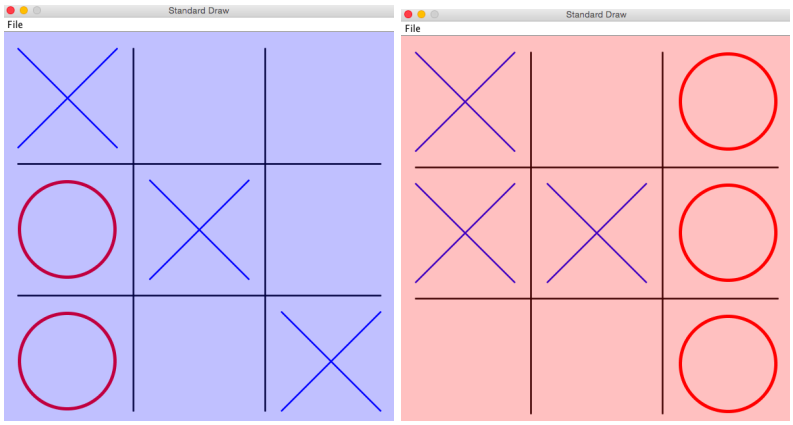
4. What should your program do in the `hasRowsWon` method if the first row doesn't yield a winner?

5. What conditions must be satisfied in order for `isGameOver` to return `true`?

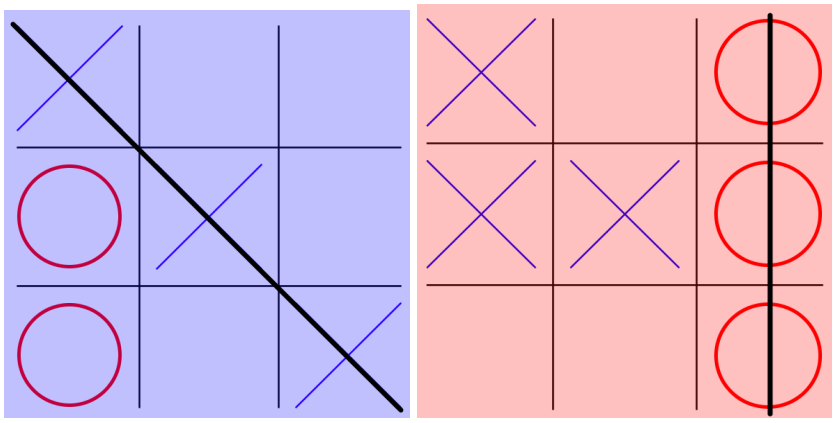
TIC-TAC-TOE

Challenge for the Bored (Challenge for the Board??? Get it??)

1. Currently the game simply stops when a player has won the game. Add a nice visual effect of coloring the game board to match the color of the player (X is blue and O is red). See below for an example. This will require you to do a little more diving into the original source code as well as exploring the StdDraw API.



2. Traditionally when the game of Tic-Tac-Toe is played on paper, the winner strikes a line through their winning path. Add a feature to do the same. Again, you'll need to rely on the StdDraw API.



3. Add a feature at the conclusion of the game which prompts the user if they would like to play another game. If yes, restart a game of Tic-Tac-Toe. If not, wish them farewell.

4. Add a naive AI. The AI should randomly play an open cell on the board. The AI does not consider a best strategy, but rather chooses a cell at random.

If you're curious for a conceptual understanding of a smarter AI, Google "Minmax Algorithm" and "Alpha Beta Pruning." Do be aware that programming the Minimax Algorithm into your game is beyond the scope of this course. That Data Structures and Algorithms class, tho. 👍