Project 1: Tic Tac Toe

Introduction

Your company's task is to create a Tic Tac Toe game in Java that allows either two human players to compete against each other, or a human player to compete against the computer. When playing against computer, there are two different algorithms that you should implement. The first algorithm is for the "easy level" of play, in which the computer just randomly places its marker in an available position on the board. The second algorithm is for the "hard level" of play, in which the computer will play optimally and should be able to beat any human player who does not play optimally.

Specific Project Requirements

1. Use Java FX to develop a nice User Interface for your application
2. When the application starts, you should have two game modes: "One Player" or "Two Players"
3. Each human player should be given the opportunity to enter her name
4. If a computer player is playing (i.e. one player mode), make sure you allow the human player to set the difficulty mode.
5. Have an option to exit the game at any time
6. If the application is closed while a game is in progress (without intentionally exiting), then when the application is launched again, it should prompt the user if the previous game should be restored.

Deliverables

1. Application Demo – show a working version of the application and your collaboration tools (e.g. waffle.io board)
2. Source code – organize your source code and make sure it is well documented
3. Javadoc documentation – run the Javadoc tool against your code to generate HTML documentation
4. Application Level User Manual – write up of how to use your application from the user's perspective (not developer details).
5. Difficulty mode implementation document – writeup of how you specifically implemented the difficulty mode (i.e. design patterns, etc.) and details about the algorithms used for both the easy and difficult modes.

Guidelines

1. Use an agile approach to develop your application
2. Make sure all team members contribute (you will be penalized for not doing any work or for doing all the work)
3. Ask questions – we are trying to simulate a real-world environment. If you have implementation questions, ask the client (i.e. the instructor).
4. Use Design Patterns where appropriate.