CS 2820

Final Project Assessment

Spencer Gritton

INSTRUCTIONS TO RUN

Running the Tic tac toe game takes just a few steps

1. Clone the project from https://github.uiowa.edu/spgritton/Project2820
2. Import the project into IntelliJ
3. Navigate to ./src/main/kotlin/edu/uiowa/Main.kt
4. Run the main method

FEATURES

The game has two main features and many additional quality of life features.

MAIN FEATURES

1. Player Vs Player
    a. This game type allows players to play their friends in head to head tic tac toe. Users can choose their names, piece colors, and the size of the board from 3x3 to 5x5. I anticipate this will be the most used feature of the application as the reason people play games like tic tac toe is to be competitive with their friends.
2. Player Vs Computer
    a. This game type allows players to play against the computer to hone their tic tac toe skills. There are three varieties of computer skill so that users can play at a level they feel comfortable at and can be competitive. This is a very important feature as if you don't have anyone to play with you still can have the chance to play!

ADDITIONAL FEATURES/QUALITY OF LIFE

These additional features were added to improve the overall experience of the game.

1. Background music and other sounds – Although not necessary to make the game work I feel these sounds significantly add to the experience of playing the game and make play time more enjoyable.
2. Controls/Rules menus – These menus are helpful for the user to not only see how the game works and what the controls are but also for them to understand the rules. I added two unique variants of tic tac toe where the board can be 4x4 or 5x5 so knowing the rules for those is essential.
3. Colored Pieces – This is nice aesthetically and helps make your pieces feel more like they are your own.
4. Winning/Losing indicators (Sounds/Visuals) – While I believe all games should have an indicator of winning or losing I feel mine are a unique feature in that they generate randomly moving confetti each time. Although this is a small touch it's unique and gives the game some character.
5. Pause menu (Esc key during game) – This is a nice way to pause the game to have a meal or get back to the main menu at any time.
6. Easy to use menus and buttons – Although some might not call these a feature I feel that my game looks exceptionally well put together and this allows the user to easily navigate the menus and begin to play with no prior experience.

TIME SPENT/OBSTACLES

I would say that I spent a significant amount of time on this project and I'm mostly happy with the end result. The first iteration of my project was smooth sailing as I had already completed the write up for a previous assignment and I knew generally which data structures I was going to use to make the game run. The real troubles came after I had to change the underlying data structures into a game complete with GUI and menus. In fact, for the first five hours of iteration two I was having such troubles creating and centering a button on the screen that I was worried I would be unable to finish the project. After this I tried to make FXML work

but even after looking at the FXML sample we were given I still could not figure it out. After watching a few more Kotlin-JavaFX videos I figured out the general premise behind programming using the JavaFX framework and it became a bit easier from there. One of the bigger struggles that I had was setting up the computer player and thinking of how I could give them different difficulties to challenge the player in new and unique ways. Since my original player vs player game was not running on a game loop of any kind I wasn't sure how I would make the computer always play after the user had. Eventually after much time spent wondering I set up a click driven environment where the computer will play its moves directly after the user. From this I set up a computer player and I believe that the hard difficulty is a sufficient challenge for all but the best tic tac toe players.

One of the biggest lessons that I learned was to make things inherit or extend frequently. There were many times between my two games (Player vs Computer and Player vs Player) that I found myself using the same methods but not having any good way to call them from the same spot, so I had to define the same method twice in separate places. I quickly found that if you are making a project where multiple things will be calling the same resource it's best to set up one centralized resource for those calls. After this I created the BoardBuilder class to house all the centralized JavaFX display methods I would call between each game and it made organization much cleaner and my entire project more concise.

THE EXPERIENCE

Overall I learned a lot and I'm happy that we had a project with a framework we were unfamiliar with. I think that this project helped me gain valuable skills in using my resources to the best of my advantage to get information about something I don't know. At the start of this project I couldn't have told you how to make even a blank screen in JavaFX and now I can say I've built an entire tic tac toe application from the ground up. I also learned a bit about pacing in a software environment and focusing on what's important. At the start, I had tons and tons of features laid out that I wanted to work on but I soon realized that as a one man team I needed to pick and choose what the best features would be for my game within the time

constraints that I had. From here I settled on the core features and perfected them so that my game looks and feels clean with the features it has implemented.

UML DIAGRAM