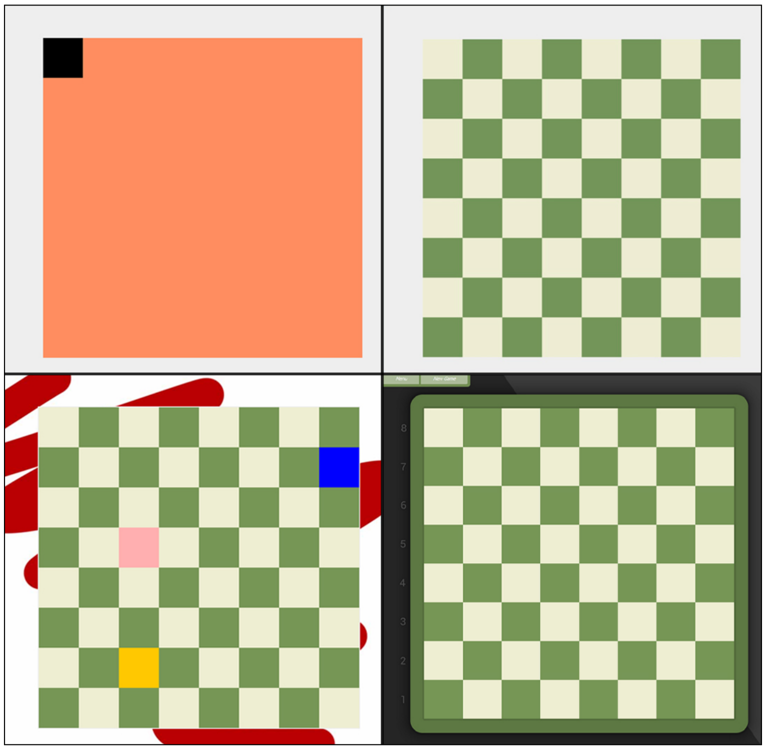
CP222 | Checkpoint Two: Implementation Details

Grace Mun, Caleb Peimann, Vladimir Palma

Progression of The Interface



Current Functionality and Goal Adjustments

Our goals have not been altered. Currently, we are on track to complete our project on Tuesday, perhaps even implementing extra content if possible. This is assuming that we promptly find a way to implement the AI player. Although we do not support the movement of chess pieces yet, we predict the implementation will be added by the end of or early tomorrow. Besides that, we have successfully put all the necessary pieces in their default board placements and have a semi-completed interface. From an outside perspective, we still have a significant amount of coding to do, but we are confident we can execute it shortly.

Submitted to Ben Nye *February 18, 2023*

Data Structures and Accessing Pieces

For our board, we utilized a 2D array of pieces. We mainly utilized an array because the board has a set size. An array works when representing a list that is a set size, as we do not need to create or delete elements of the array but rather change their value. 2D arrays or matrices are commonly used to represent boards, and we were familiar with them in our previous computer science class.

When choosing a data structure for the possible moves, we thought about what we wanted from the data. We needed a data structure that allowed us to add and remove an unknown number of legal moves. This being said, we did not need an extremely large number of moves, and we did not need to be able to remove things from the middle. Thus, an ArrayList made sense as it is a structure we are comfortable with and can use to easily add elements to the end of the list of undetermined sizes.

For the chessboard itself, we created a JPanel that takes in eight rows and eight columns. We wrote a loop afterward that created separate JPanels within each square. This increases the game's functionality paired with our two data structures above, allowing us to easily access piece positions and their potential moves.

Relationships Between Classes

We connected the classes we individually worked on by creating methods that utilize variables in other classes. For example, to implement our pieces onto the board, we have a method that can store all of the pieces in a 2D array and then another method that uses that array to paint all of our pieces onto our chessboard. In our classes for our pieces, we have also created a getter method to access each piece's positions and attacks and use them in our main game class to calculate the current game status.

Smooth Integration?

Integration has gone quite smoothly, especially when it comes to knowing where each of our components fit into the project. We were also able to work off of each other's code. However, successfully integrating components has been a little difficult, especially with our unfamiliarity with the Swing platform. Specifically, we have had issues repainting our pieces to different squares when moving them.

Original Plan Compared to Current Implementation

We have followed our original plan of integrating all our parts closely. Our original plan was to split our game into three parts: the board, pieces, and the main game (such as detecting checks, checkmate, and so on). We created the main board that we have been able to integrate the pieces into, and we are now working on implementing the pieces into the main game part.

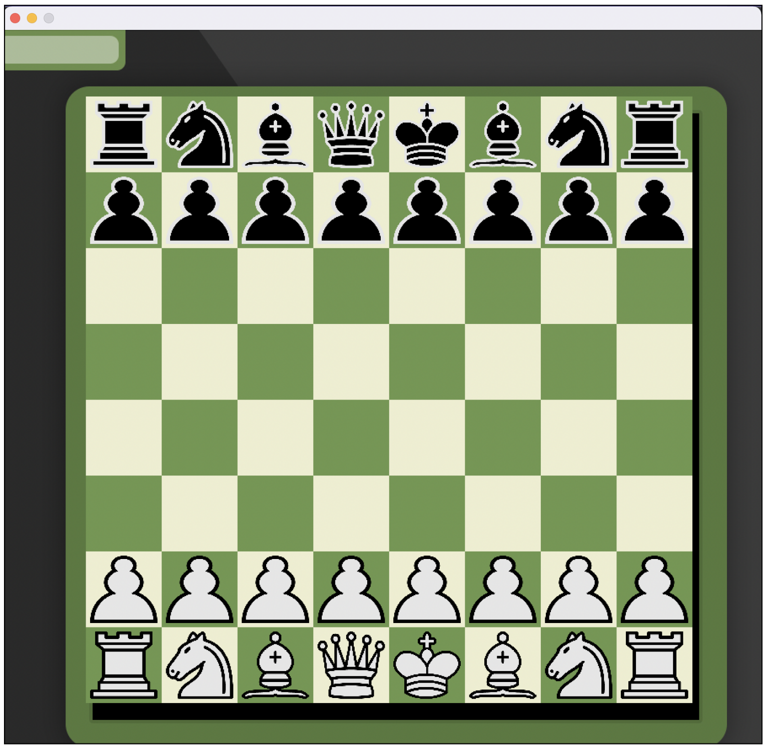


Figure 1: Chessboard with Pieces

Difficult Implementations and Our Navigation

Each of us has encountered challenging issues, such as Swing components not appearing when called, which slowed our production. We broke our project into three parts but underestimated the time it took to complete each. Each of our pieces builds off the other. Creating the main interface made it difficult for the pieces to be created. With pieces not implemented, we cannot write the game's rules. It may not sound too challenging, but the process was certainly time-consuming. We have navigated these issues by completing our parts as soon as possible, using our in-class and online resources when needed.