MIS 233

**WEB BASED APPLICATION PROGRAMMING**

**PROJECT 1**

Zeynep Sena Tınaz

2019502168

09/12/2021



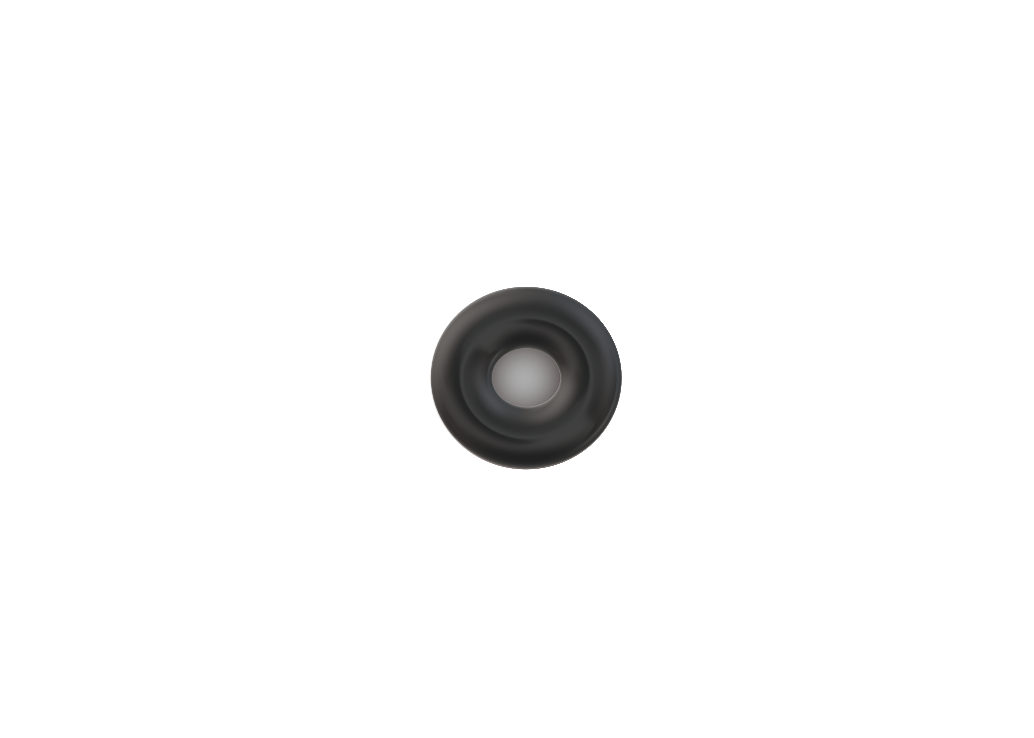
**INTRODUCTION**

The Project asked us to code a checkers game using HTML, CSS, JavaScript and preferably JQuery.

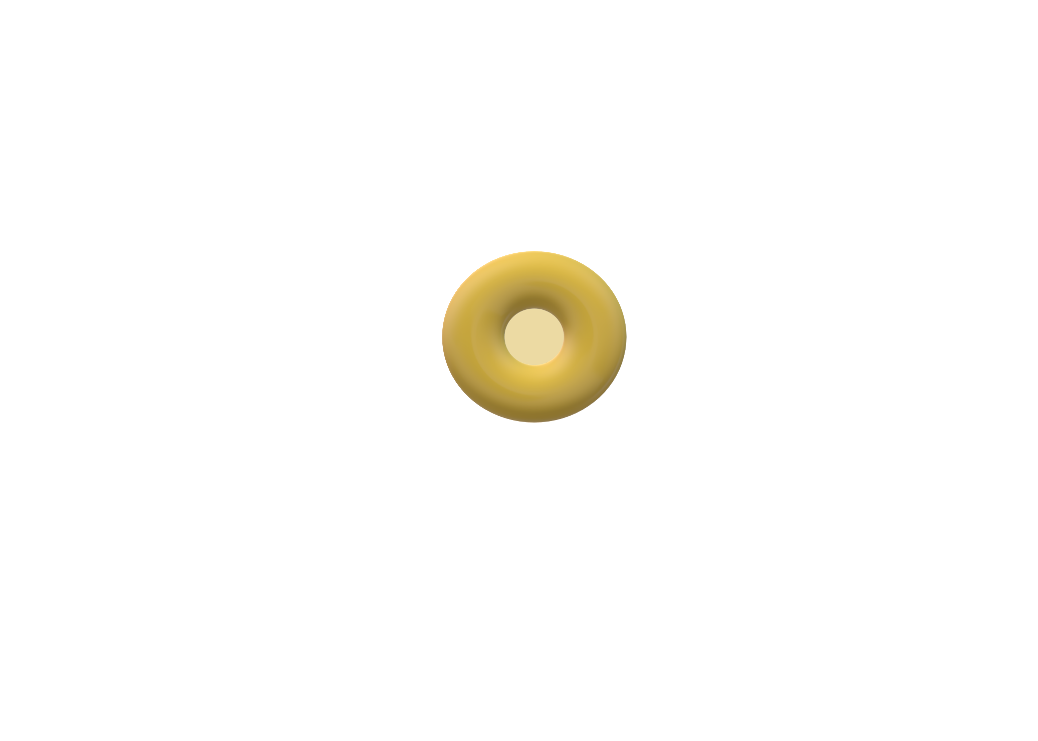
After understanding the logic of the game given in the Project description, I have come up with the algorithm needed to code the game and got to work. Roughly what I needed to do was, first create the checker board with HTML using tables, and then design it. Secondly, I had to create the checkers (pieces). There were two ways I could do that:

* I could use an ‘img’ tag and use checkers images off the internet or design them myself using Adobe Photoshop.
* I could design my own pieces with CSS.

*Even though I have designed my own checkers with Adobe Photoshop as show down below, later on I decided to design them using CSS as it felt better for me.*

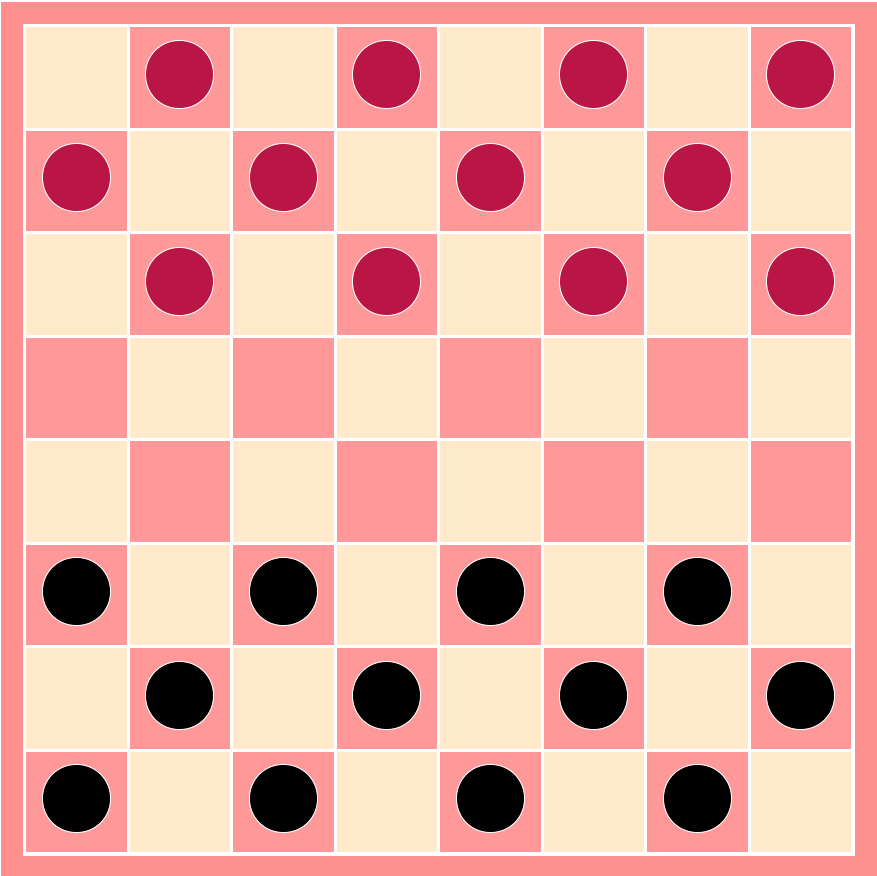


*Checker 1*



*Checker 2*

I ended up with a board looking like this after designing my board and my checkers;



Now for the hard part, all I had to do was use JavaScript and give my website/my game functionality.

**ALGORITHM/PROGRAM DESIGN**

* Created a 8x8 board using table element with HTML.
* Designed the board and the checker pieces using CSS.
* Created a representation of the board on the back and with a 64 item array.
* Established variables that will be used to reference things.
* Saved the user variables (like score) to keep track of the game state.
* Created the object that will hold the properties of the pieces.
* Created the function that will give the pieces a click event listener.
* Called the function at the bottom of the file that initializes the event listeners.
* Created two functions called removeCellonclick (dynamically adding and removing attributes) and resetBorder (gave the selected piece a border of green but if the user clicks on any other piece it needs to get reset) and called them inside another function called getPlayerPieces.
* Created a function called resetSelectedPieceProperties to reset every time a piece is clicked.
* Created getSelectedPiece and findPiece functions to find where the piece is located on the board.
* Created the function isPieceKing to find out if the selected piece is a king or not.
* Created getAvailableSpaces function with necessary if statements to analyze the surrounding cells that a piece can make without jumping another piece. (pieces can move either -/+7 cells or -/+9 cells.)
* Created the checkAvailableJumpSpace function with if statements to analyze the jumping spaces for both red and the black pieces. (+ / - 14 or + / - 18)
* Created the checkPieceConditions function.
* Created the givePieceBorder function to give a green border and highlight the selected pieces.
* Created a function called giveCellsClick for the pieces when they do not have a possible move and gave them an onclick attribute of our next function: makeMove.
* For the makeMove function;
  + first removed the piece on the front end by using .remove() attribute. Then changed the .innerHTML of the cell to empty strings. These make it look like nothing was ever on the cell.
  + Created the necessary if else statements and reset the redPieces variable to querySelectorAll(“p”).
  + Duplicate the code for the else statement without the class of king.
* Created the changeData function to change the data on the back end to reflect the board from the front end.
* Created removeEventListeners and checkForWin functions.
* Lastly, created the changePlayer function. This one is self-explanatory.

**STUDENT FEEDBACK**

I have really struggled while writing this code. I needed to use a lot of references and it took me a lot of time to finish it. It was overwhelming in my opinion for this course’s level. However, on the bright side, having written this code I feel a lot more confident with JavaScript now.

**REFERENCES**

1. <https://levelup.gitconnected.com/creating-a-board-game-checkers-with-javascript-ecd562f985c2>
2. <https://alvin-almazov.com/rules/checkers-rules-explained-everything-about-the-game-for-the-smartest/>