Name: Trung Nguyen Huynh

Case ID: tnn18

TESTING REPORT – Project 5

I. Test helper methods

1. isValid

- Test when idRow or idColumn is out of bound
    - Above the board
      (line 19)
    - Below the board
      (line 23)
    - On the left side of board
      (line 27)
    - On the right side of board
      (line 31)
- Test when idRow and idColumn is inside the game board
  In each case of row, I will test the column is at the first-middle-last column
    - The row is the first row (with first-middle-last column)
      (line 36)
    - The row is in the middle of the board (with first-middle-last column)
      (line 40)
    - The row is the last row (with first-middle-last column)
      (line 44)

2. numberInLine

Because the numberInLine method counts the number of squares which are continuous and have the same color, it **always returns at least 1** (including the case in which the currently played square is empty)

- Test right direction
    - There is only 1 piece
      (line 58)
    - There are some pieces
      (line 62)
    - The method reaches the edge of the board
      (line 66)
- Test left direction

    - There is only 1 piece
      (line 70)
    - There are some pieces
      (line 74)
    - The method reaches the edge of the board
      (line 78)
- Test down direction
    - There is only 1 piece
      (line 82)
    - There are some pieces
      (line 86)
    - The method reaches the edge of the board
      (line 90)
- Test up direction
    - There is only 1 piece
      (line 94)
    - There are some pieces
      (line 98)
    - The method reaches the edge of the board
      (line 102)
- Test down-right direction
    - There is only 1 piece
      (line 106)
    - There are some pieces
      (line 110)
    - The method reaches the edge of the board
      (line 114)
- Test up-left direction
    - There is only 1 piece
      (line 119)
    - There are some pieces
      (line 123)
    - The method reaches the edge of the board
      (line 127)
- Test down-left direction
    - There is only 1 piece
      (line 132)

    - There are some pieces
      (line 136)
    - The method reaches the edge of the board
      (line 140)
- Test up-right direction
    - There is only 1 piece
      (line 145)
    - There are some pieces
      (line 149)
    - The method reaches the edge of the board
      (line 154)

3. isOpen

- Test right direction
    - The method returns true
      (line 167)
    - The method returns false (the square is occupied)
      (line 170)
    - The method returns false (it reaches the edge of the board)
      (line 173)
- Test left direction
    - The method returns true
      (line 176)
    - The method returns false (the square is occupied)
      (line 180)
    - The method returns false (it reaches the edge of the board)
      (line 183)
- Test down direction
    - The method returns true
      (line 187)
    - The method returns false (the square is occupied)
      (line 190)
    - The method returns false (it reaches the edge of the board)
      (line 193)
- Test up direction
    - The method returns true
      (line 197)

    - The method returns false (the square is occupied)
      (line 200)
    - The method returns false (it reaches the edge of the board)
      (line 203)
- Test down-right direction
    - The method returns true
      (line 207)
    - The method returns false (the square is occupied)
      (line 210)
    - The method returns false (it reaches the edge of the board)
      (line 213)
- Test up-left direction
    - The method returns true
      (line 217)
    - The method returns false (the square is occupied)
      (line 220)
    - The method returns false (it reaches the edge of the board)
      (line 223)
- Test down-left direction
    - The method returns true
      (line 227)
    - The method returns false (the square is occupied)
      (line 230)
    - The method returns false (it reaches the edge of the board)
      (line 233)
- Test up-right direction
    - The method returns true
      (line 237)
    - The method returns false (the square is occupied)
      (line 240)
    - The method returns false (it reaches the edge of the board)
      (line 243)

4. isWin

- The player can win
  I will also test the method in 4 different directions in the cases below.
    - The last move is at the endpoints of the series
      (line 256)

    - The last move is at the middle of the series
      (line 261)
- The player cannot win
    - The length of series is less than numNeedToWin
      (line 279)
    - Overline situation
      (line 286)

5. isFourFour

- The player violates 4-4 rule
  I will also try to test differently to cover 4 directions
    - Case 1: Closed rows
      (line 304)
    - Case 2: Open rows
      (line 311)
    - Case 3: Closed and open rows
      (line 319)
- The player does not violate 4-4 rule (There are less than 2 '4-4' rows)
    - A row with more than (numNeedToWin – 1) continuous pieces
      (line 328)
    - A row with less than (numNeedToWin – 1) continuous pieces
      (line 335)

6. isThreeThree

- The player violates 3-3 rule
  I will also try to test differently to cover 4 directions
    - There are exactly 2 open '3-3' rows
      (line 352)
    - There are more than 2 open '3-3' rows
      (line 359)
- The player does not violate 3-3 rule
    - The number of **open** '3-3' rows are less than 2
      (line 367)
    - There are not enough rows that has **exactly** (numNeedToWin – 2) pieces
      (line 374)

7. processMove

In this test, I will test the configuration of game board and the current player before and after running processMove method

- The game has already had the winner
  (line 392)
- The square to process is already occupied by another player
  (line 399)
- The move makes the current player win the game
  (line 406)
- The move violates '4-4' rule
  (line 421)
- The move violates '3-3' rule
  (line 432)
- The move is completely valid
  (line 447)

8. resetGame

- A board with one row and one column
  (line 486)
- A board with many rows and one column
  (line 493)
- A board with one rows and many columns
  (line 500)
- A board with many rows and many columns
  (line 507)

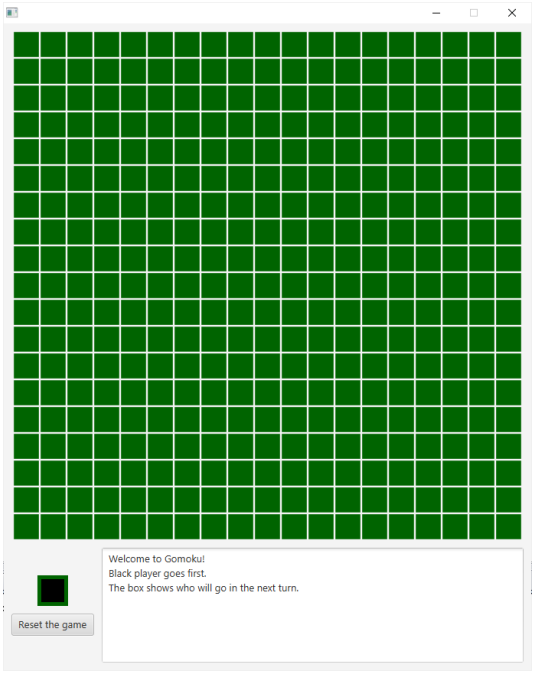
II. <u>Test GUI</u>

1. When the game is started

There will be 4 main features:

+ The game board (including the buttons to be clicked)
+ A box to display color of the current player (showCurrentPlayer)
+ A box to inform players of game messages (notificationBar)
+ A button to reset the game

The width of notificationBar is flexible to the size of the board. However, it should be at least 300.



(19x19 game board)

(19x7 game board)

2. **Visual effects for buttons**

When we move the mouse over the button, the button's background changes the color (into green)

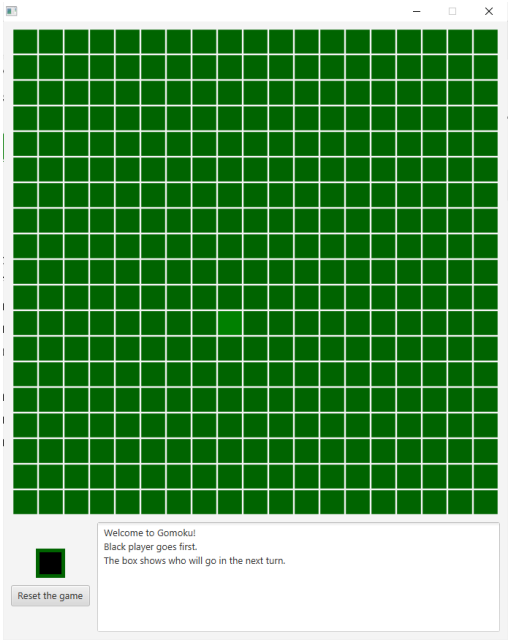
When we click the mouse, the buttn's background changes the color (into light green) 

When we move the mouse outside the button, the button's background changes the color back to the original color (into dark green) 

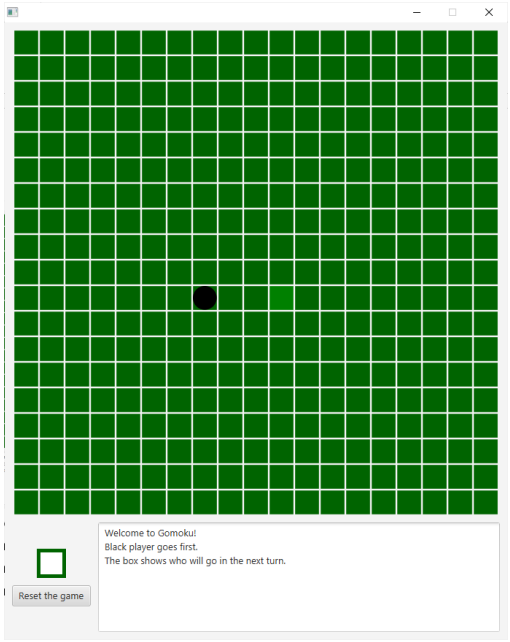
3. **After the button is clicked**

- The move is valid

The square on the board will be set the color to black or white (based on the current player) and the showCurrentPlayer box will switch the color



(before the move)



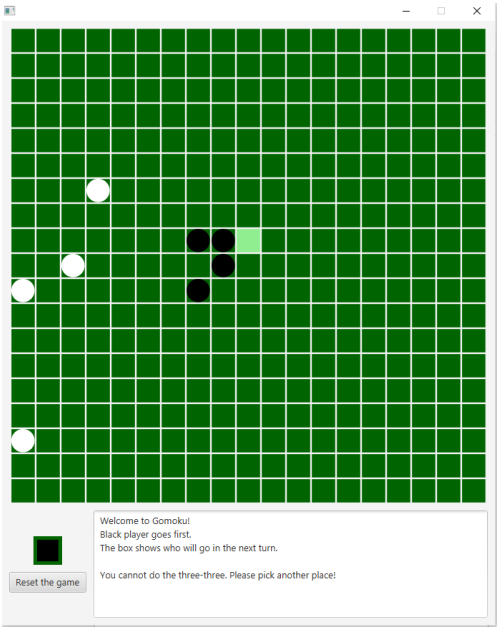
(after the move)

- The move violates 4-4 or 3-3 rule

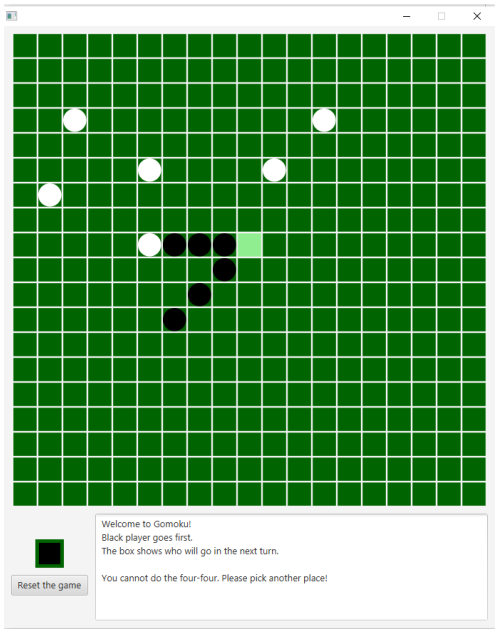
The game board and showCurrentPlayer box do not change.

The notificationBar will announce:

+ For 4-4 rule: “You cannot do the four-four. Please pick another place!”
+ For 3-3 rule: “You cannot do the three-three. Please pick another place!”



(3-3 rule)



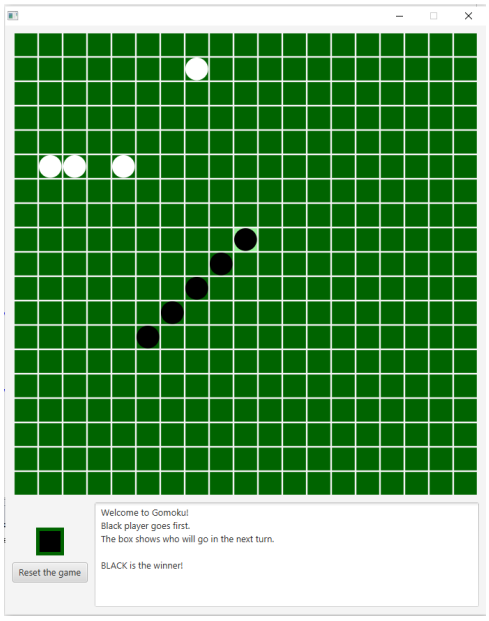
(4-4 rule)

- The player wins the game

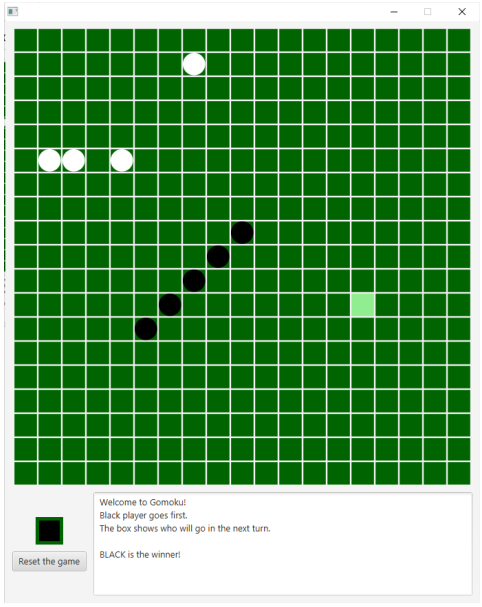
The notificationBar will announce who is the winner.

The showCurrentPlayer box does not change color.

When another button is clicked, nothing happens



(Black player wins!)



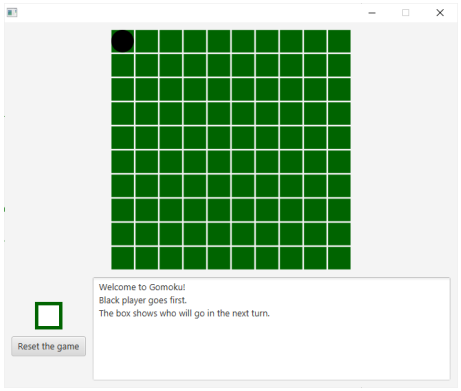
(Nothing happens after that)

4. **Test the loop to find idRow and idColumn (inside the setOnAction)**

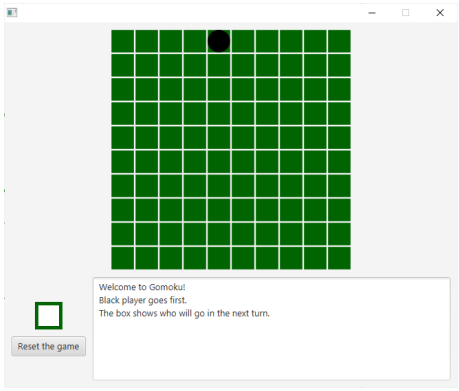
If the loop behaves correctly, the GUI and other helper methods should then behave correctly.

I will test the loop on 10x10 board

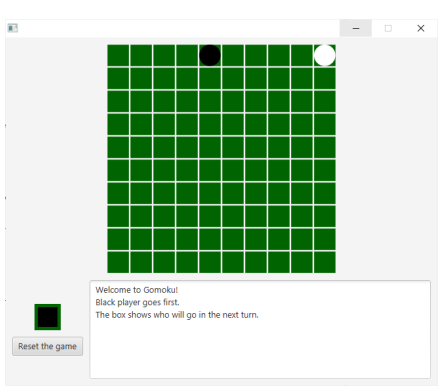
- Click the button in the first row



(1st row, 1st column)

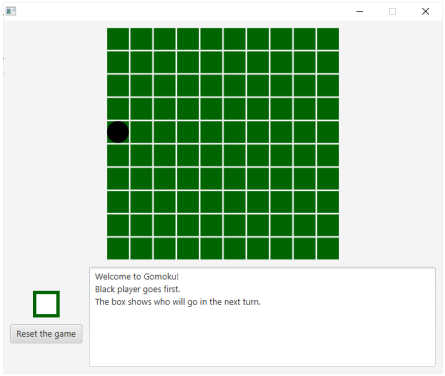


(1st row, 5th column)

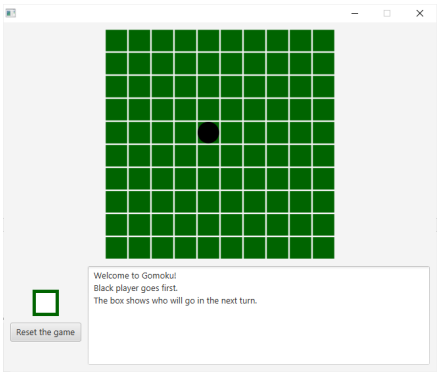


(1st row, 10th column)

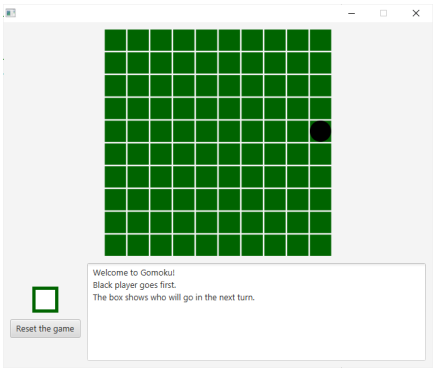
- Click the button in the middle row



(5th row, 1st column)

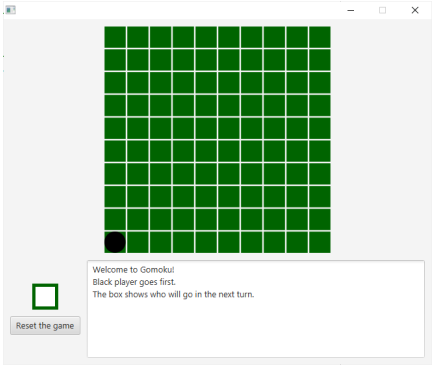


(5th row, 5th column)

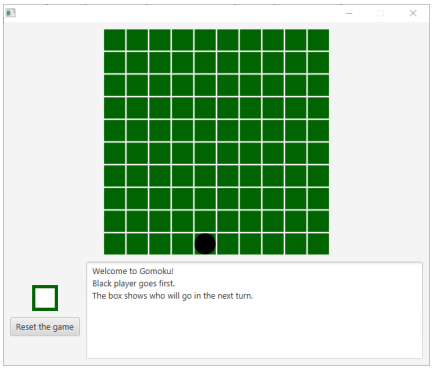


(5th row, 10th column)

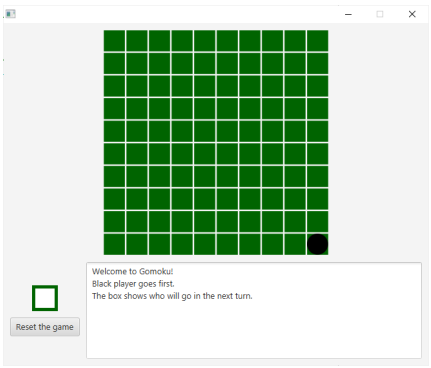
- Click the button in the last row



(10th row, 1st column)



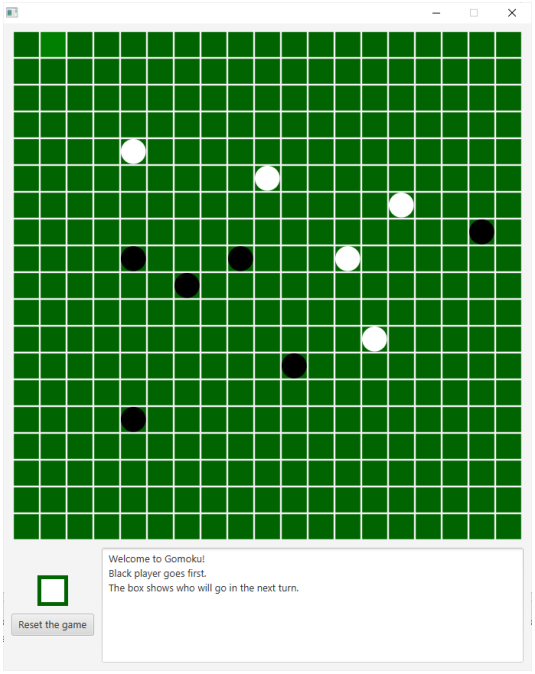
(10th row, 5th column)



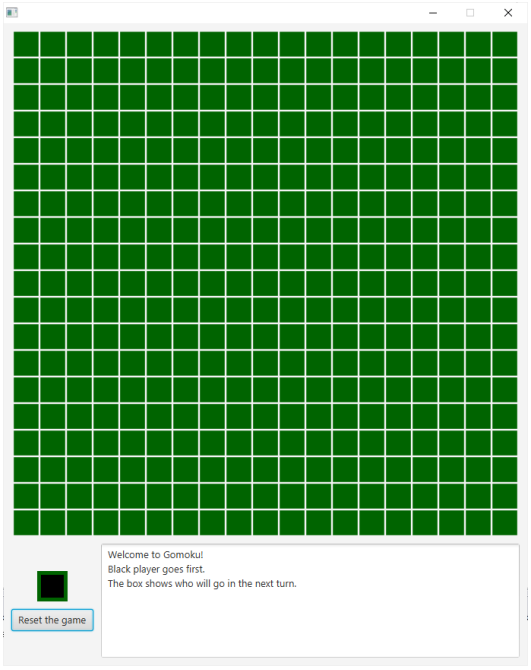
(10th row, 10th column)

5. Test the reset button

Case 1: I will launch the game and do some moves on the board.

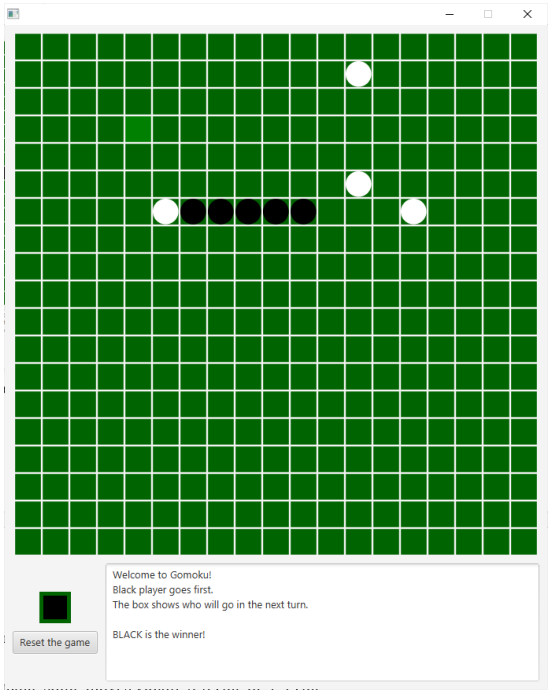


(before resetting)

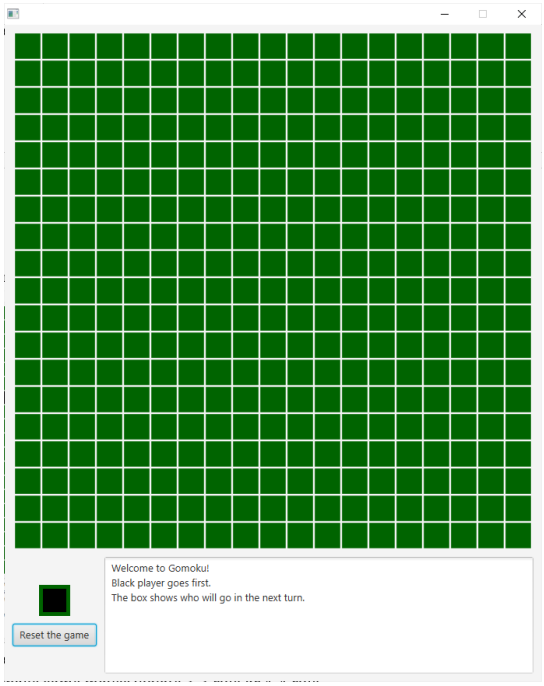


(after resetting)

Case 2: I will make a player win this game

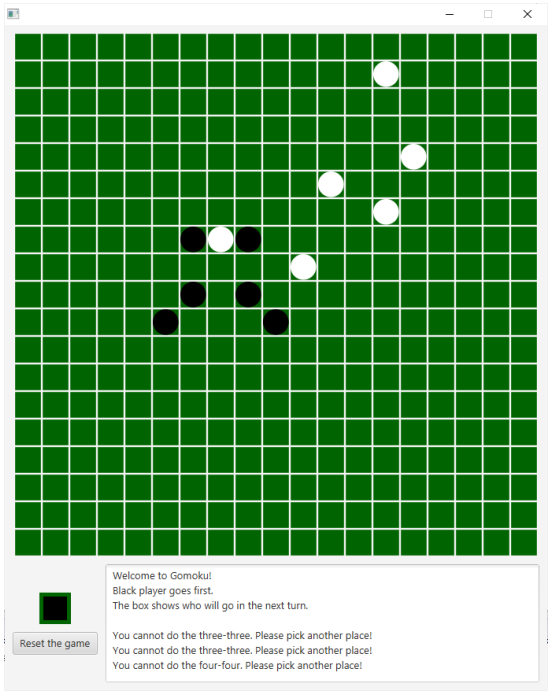


(before resetting)

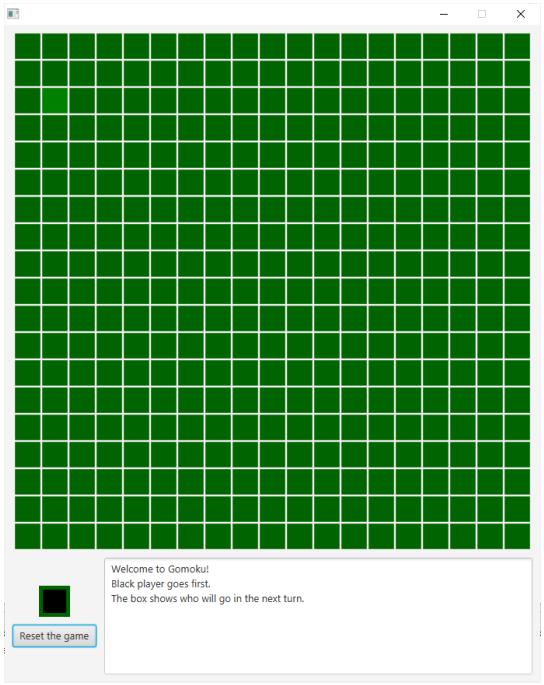


(after resetting)

Case 3: I will make some moves violate 3-3 rule or 4-4 rule



(before resetting)



(after resetting)