# Assignment 2

Munkhbat Nandin

HDVC5Y

nandin2310@gmail.com

## Task 6 – Four Game

This a two-player game is played on a board consists of n x n fields, where each field contains a value between 0 and 4. Initially, all the fields contain the value of 0. If a player chooses a field, then the value of the field and its neighbours incremented by one (if the value is less than 4). The player's score represents how many fields did he make to have the value of 4. If a value of a field reaches 4, then the field is colorized with the color of the actual player (red or blue). The game ends, when all fields have the value of 4. The player having the higher score wins. Implement this game, and let the board size be selectable (3x3, 5x5, 7x7). The game should recognize if it is ended, and it has to show in a message box which player won. After this, a new game should be started automatically.

**Operations:**

1. showExitConfirmation(): void

Displays an exit confirmation dialog when the window is closed by the user "Do you really want to quit?" with YES/NO options.

2. doUponExit()

Close the current window

3. getActionListener(final int size): ActionListener

When a button is clicked, it creates a new Window of the given size (3/5/7) and adds it to the list of gameWindows, making the window visible.

4. actionPerformed(ActionEvent e): void

If a player chooses a field, the value of the field and its neighbors are incremented by one (if the value is less than 4). If the value of a cell reaches 4, it is colored to indicate a winning move and the score for the respective player is updated. The current player's turn is switched, and the method checks if the game is over, displaying the winner starting a new game and updates the score labels.

5. isGameOver(): Boolean

Returns true if all the numbers of in the table are 4

6. showWinner(): void

Show a message dialog with winner's name(color)

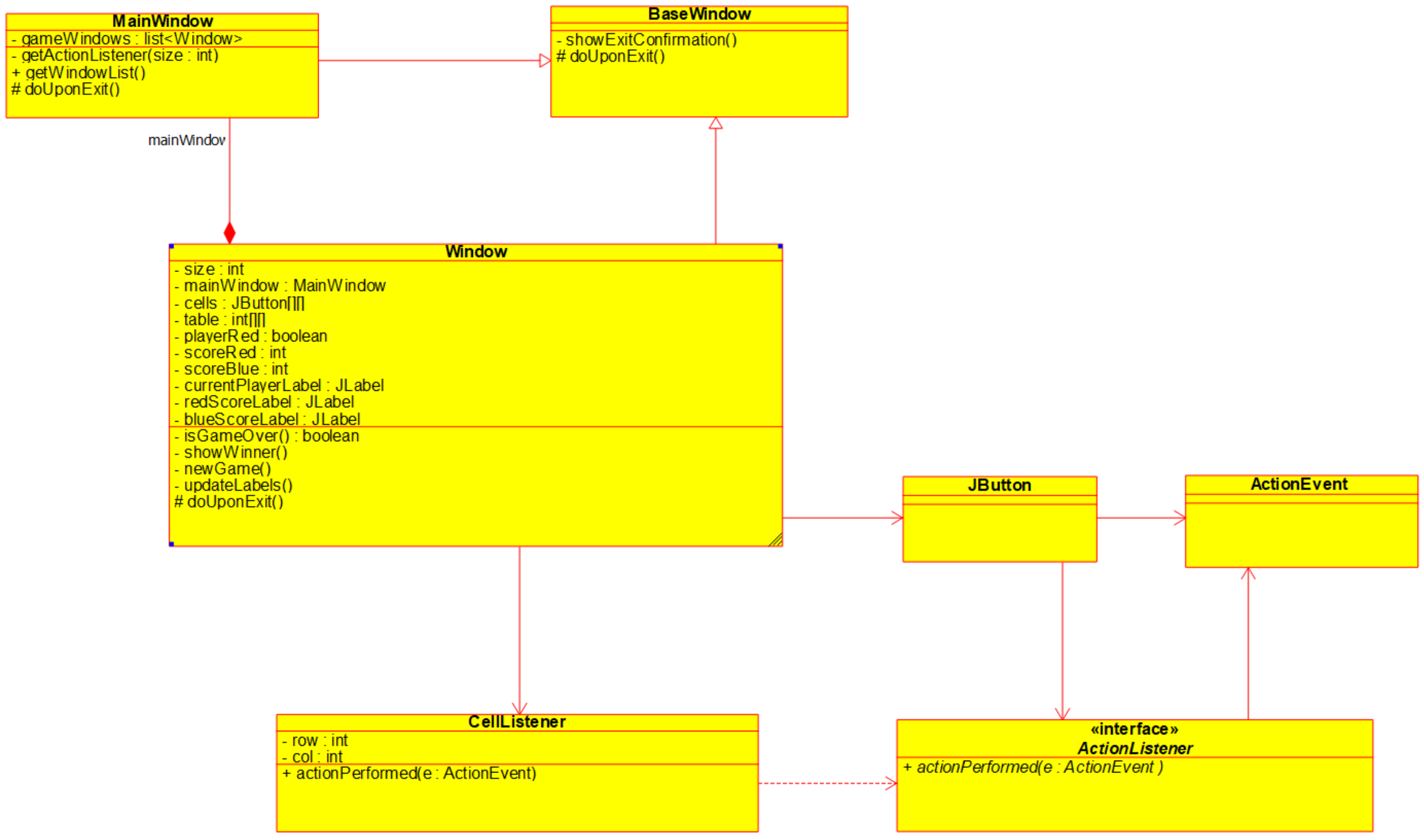
7. newGame(): void

Creates a new game window and close the current one.

8. updateLabels(): void

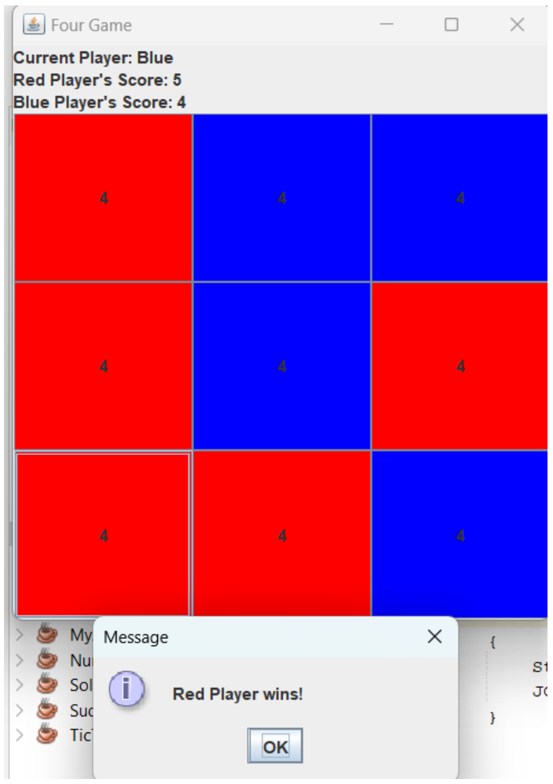
Update labels of the current table, and scores.

**UML class diagram:**



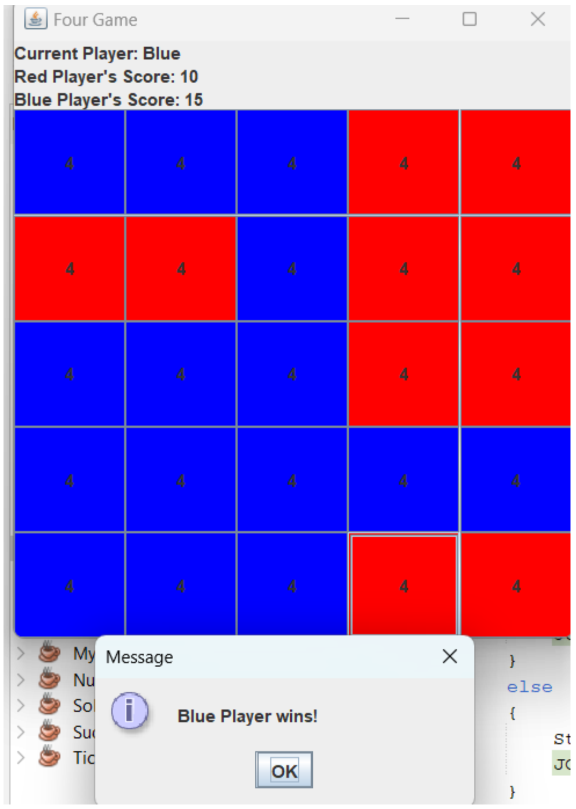
**Test cases:**

1. On 3x3 table, red player wins with 5 scores



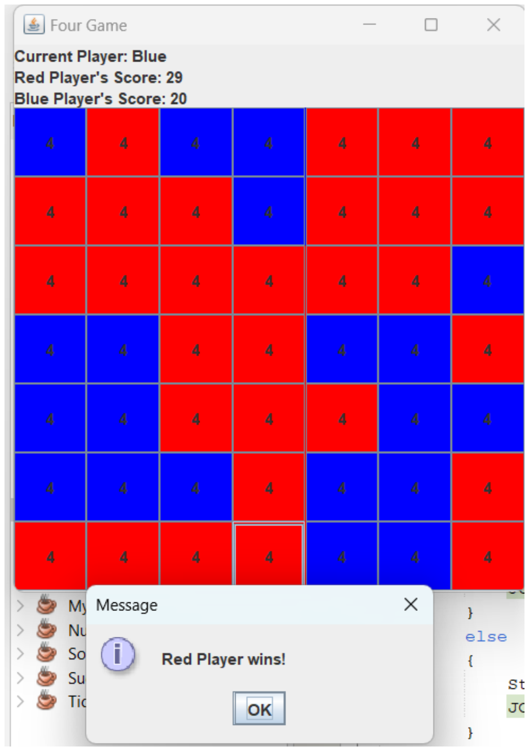
```
Four Game
Current Player: Blue
Red Player's Score: 5
Blue Player's Score: 4
4 4 4
4 4 4
4 4 4
Message
Red Player wins!
OK
```

2. On 5x5 table, Blue player wins with 15 scores



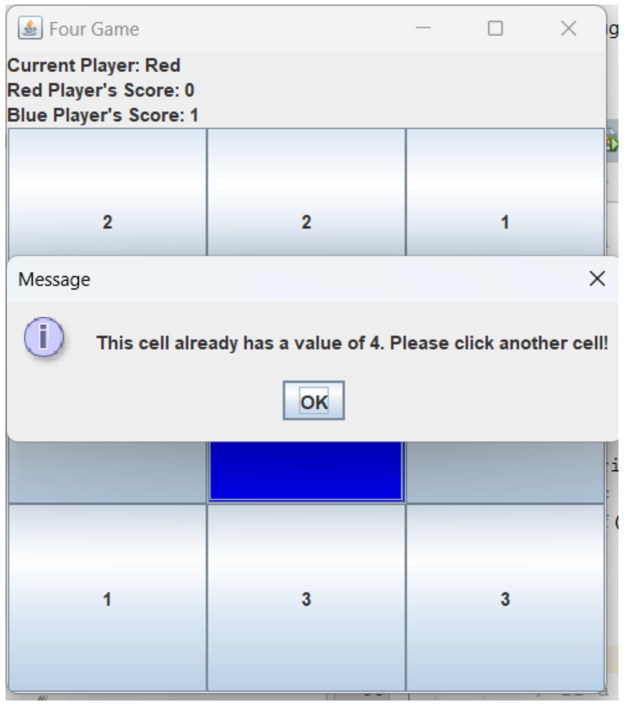
```
Four Game
Current Player: Blue
Red Player's Score: 10
Blue Player's Score: 15
4 4 4 4 4
4 4 4 4 4
4 4 4 4 4
4 4 4 4 4
4 4 4 4 4
Message
Blue Player wins!
OK
```

3. On 7x7 table, blue player wins with 5 scores



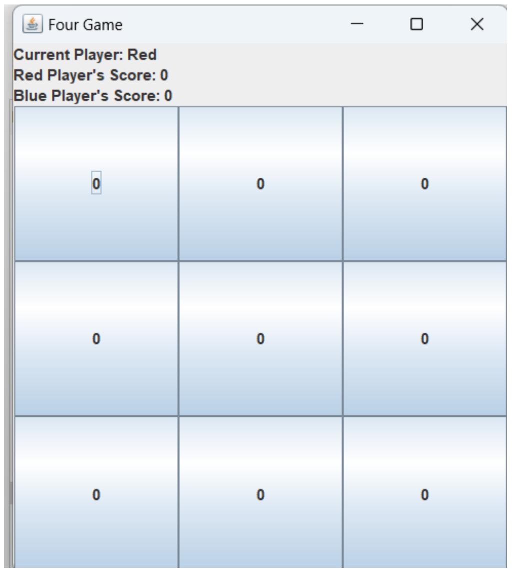
```
Four Game
Current Player: Blue
Red Player's Score: 29
Blue Player's Score: 20
4 4 4 4 4 4 4
4 4 4 4 4 4 4
4 4 4 4 4 4 4
4 4 4 4 4 4 4
4 4 4 4 4 4 4
4 4 4 4 4 4 4
4 4 4 4 4 4 4
Message
Red Player wins!
OK
```

5. If a player clicks on a cell with value of 4, players are not switched and it asks the current player to click another cell.



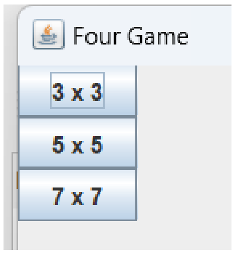
```
Four Game
Current Player: Red
Red Player's Score: 0
Blue Player's Score: 1
2 2 1
Message
This cell already has a value of 4. Please click another cell!
OK
1 3 3
```

6. If a user clicks 'OK' after finishing a game, a new game with the same size automatically begins.

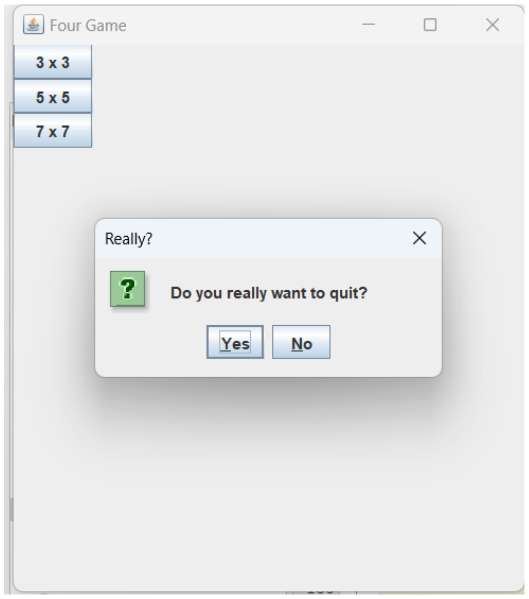


Current Player: Red
Red Player's Score: 0
Blue Player's Score: 0

7. If players want to play on a different size of board after finishing a game, they can close the current window and choose a board size for the new game.



8. When a player closes a gaming window or a main window, it displays an exit confirmation dialog.



Do you really want to quit?