Programing Technology

Kamila Kubala
GUM6BS

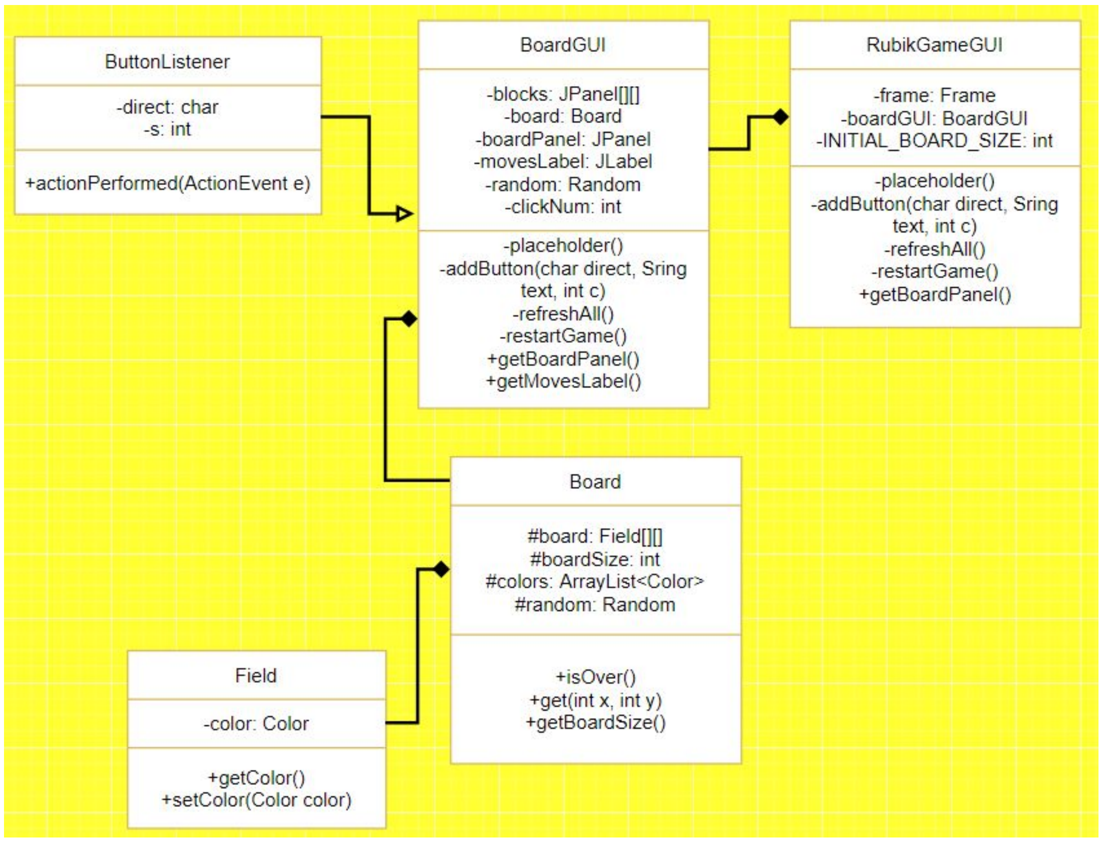
2nd Assignments - Rubik Board

10.11.2020

Description

This game is a two dimension variant of the Rubik cube. The board of the game consists of n x n fields. There are n different colors, and exactly n fields have the same color. The fields are shuffled initially. A player can move cyclically the colors of a row or column (e.g. moving a row to the right, the color of the first field will be the color the last field) to make homogenous rows and columns. The game ends, when each row (or column) contains one color. Implement this game, and let the board size be selectable (2x2, 4x4, 6x6). The game should recognize if it is ended, and it has to show in a message box how much steps did it take to solve the game. After this, a new game should be started automatically.

Class Diagram



Classes

1. **Field**

   Attributes:

   - **color**

   Methods:

   - **getColor() -** this method returns the color of this field.
   - **setColor(Color color)** - it is setting a color of a field.

2. **Board**

   Attributes:

   - **board** - a two-dimensional array composed of fields.
   - **boardSize -** size of board.
   - **colors -** array of colors in this game.

   Methods:

   - **initializer -** The method creates a board of the selected size filled with fields. Then it randomizes n colors and assigns them to the fields so that each color appears exactly n times.
   - **isOver()** - the method checks if the same colors appear in all rows or columns.
   - **get(int x, int y)** - the method returns field from position x,y.
   - **getBoardSize()** - the method returns the size of the board.

3. BoardGUI

Attributes:

- **blocks**
- **board**
- **boardPanel**
- **movesLabel**

Methods:

- **initializer -** the method organizes the arrangement of buttons and fields. So that the fields appear in the n x n table, and the buttons for the rows and columns are right next to them.
- **placeholder()** - allows you to create a panel of an appropriate size to fill the gap in the corners
- **addButton(char direct, String text, int c)** - creates a button and adds its ButtonListeren. The button will operate in the selected direction (direct) and display the text. The constant c is the number of the row or column it will handle.
- **refreshAll()** - refreshes the field colors and moves label.
- restartGame() - shuffles the colors of the fields and sets the move counter to 0.
- getBoardPanel() - returns BoardPanel.

ButtonListener

Attributes:

- **direct**
- **s**

Methods:

- **actionPerformed(ActionEvent e) -** When one of the buttons is pressed, the method will shift the colors of the corresponding fields and increase the number of moves.

4. RubikGameGUI

Attributes:

- **frame**
- **boardGUI**
- **INITIAL_BOARD_SIZE**

Methods:

- **initializer** - creates the main application window and operates the menu bar.

Events:

When the user selects "Game" from the MenuBar he will be able to:

- **New** - start a new game with the selected size. The method will clear the current window and start a new one with new settings.
- **Exit** - exit the game and close the application window.