Checkers - CheckerBoard

Description: Create a CheckerBoard class that generates a board for the game of checkers in JavaFX.

Purpose: This challenge provides experience in building a class to generate a JavaFX UI. It provides experience with JavaFX UI hierarchies and in creating an algorithm to generate the checkerboard layout.

Requirements:

Project Name: Checkers

For all challenges in this course you are to use the project naming scheme that was presented in the first challenge, Hello World. The project name is to be preceded by your pawprint with the first letter capitalized. Example if pawprint is abcxyz9 then the project name is to be: Abcxyz9Checkers.

IDE: NetBeans
Language: Java
JDK: 8
UI: JavaFX (do not use Scene Builder or FXML for this challenge)

In this challenge you are to create a CheckerBoard class that builds a UI hierarchy that is the board in the game of checkers. The following are examples of boards generated by the CheckerBoard class.

Fig. 1: Default Colored Board

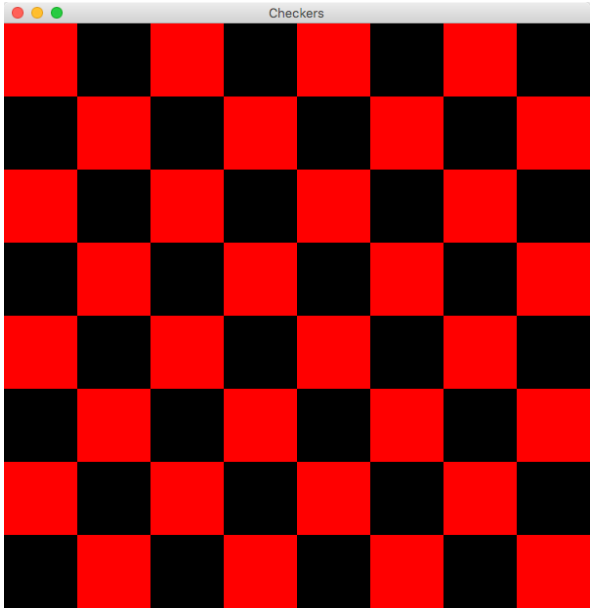
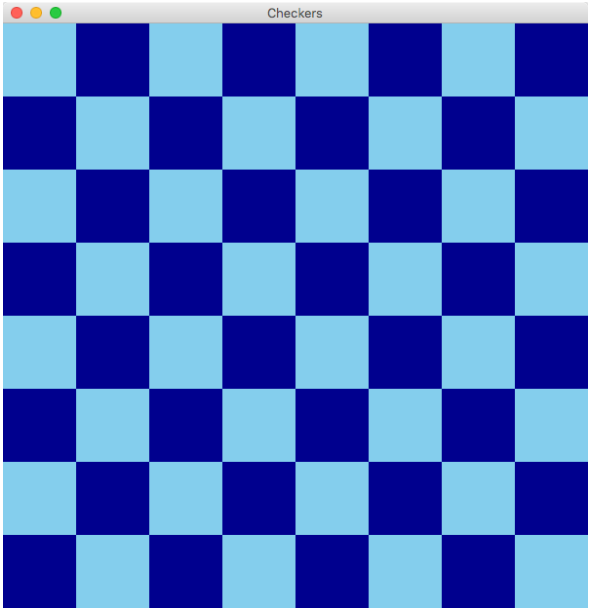


Fig. 2: Custom Colored Board



Checkers / English draughts: https://en.wikipedia.org/wiki/English_draughts

Checkers - CheckerBoard

The default colored board is generated when colors are not specified via the constructor. Boards are comprised of squares that are a dark color or a light color. For the default colored board the light color is Color.RED and the dark color is Color.BLACK.

A normal board is 8 x 8. The top left square is a light color and in each direction the squares alternate between a light color and a dark color. CheckerBoard instances, via two constructors, are to be configurable in the following ways:

- Number of rows and number of columns.
- Board width and the board height in pixels.
- Colors for the light color and the dark color.
- One constructor allows the colors to be specified and the other does not. If the colors are not specified, the default board colors are to be used: Color.RED and Color.BLACK.

CheckerBoard Class Requirements

The following are requirements for the CheckerBoard class.

- Its fields are to be private.
- It has two constructors.
    - First constructor receives: int numRows, int numCols, int boardWidth, int boardHeight
    - Second constructor receives: int numRows, int numCols, int boardWidth, int boardHeight, Color lightColor, Color darkColor
- Code is not to be duplicated in the constructors. One constructor is to use the other constructor.
- It has a public build() method that builds the board UI and returns a GridPane as the root object.
- It has a public getBoard() method that returns the GridPane generated by build() or null if one hasn't been built yet.
- It has the following additional public getters: getNumRows(), getNumCols(), getWidth(), getHeight(), getLightColor(), getDarkColor(), getSquareWidth(), getSquareHeight()

Building the Board UI

The following is information about building the board UI.

The board is to be procedurally generated from the data held in the CheckerBoard fields and is to fill the height and width. The UI is comprised of a GridPane that is the root object that contains Rectangle objects that represent the squares. The colors of the Rectangle objects follow the rules for a checker board: the top left square is a light color and in each direction the squares alternate between a light color and a dark color. The result is to look like the examples shown in Figures 1 and 2. The algorithm for determining the color choice for each square (light or dark) as it is generated should be intelligently implemented. Hint: there is no need for flags or flip-flopping states.

Testing the CheckerBoard Class

The following is information about testing the CheckerBoard class.

The main class, Checkers, is to have the following fields:

```
private final int numRows = 8;
private final int numCols = 8;
private final int boardWidth = 600;
private final int boardHeight = 600;
private final Color lightColor = Color.SKYBLUE;
private final Color darkColor = Color.DARKBLUE;
private CheckerBoard checkerBoard;
```

In the start() method of the main class a CheckerBoard instance is to be created based on the information provided in the fields:

```
checkerBoard = new CheckerBoard(numRows, numCols, boardWidth, boardHeight, lightColor,
darkColor);
```

The build() method is to be called on the instance and assigned to a variable called **board**.

The pane referenced by **board** that resulted from calling build() is to be used as the root element in creating the Scene object that is set as the Scene for the Stage passed to the start() method in the main class. The Scene object is to use boardWidth and boardHeight to establish its width and height.

The title for the State object is to be set to **Checkers** so that **Checkers** is displayed in the bar at the top of the application window.

If you do the above and run the application, you should see a window that has **Checkers** at the top and contains the custom colored (sky blue / dark blue) 8 x 8 checker board that is 600 px x 600 px.

If you comment out the line of code that creates the instance of the CheckerBoard shown above and write a new line that creates an instance where no colors are specified, you should get the default colored (red / black) checker board:

```
checkerBoard = new CheckerBoard(numRows, numCols, boardWidth, boardHeight);
```

Leave both version of new CheckerBoard(…) in the code so they can be commented/uncommented to test that they work.

Submit a zip file of your NetBeans project.