Game.xml

```
<?xml version="1.0" encoding="UTF-8"?>


<?import javafx.geometry.Insets?>
<?import javafx.scene.control.Button?>
<?import javafx.scene.control.Label?>
<?import javafx.scene.control.TextField?>
<?import javafx.scene.layout.ColumnConstraints?>
<?import javafx.scene.layout.GridPane?>
<?import javafx.scene.layout.Pane?>
<?import javafx.scene.layout.Region?>
<?import javafx.scene.layout.RowConstraints?>
<?import javafx.scene.layout.VBox?>
<?import javafx.scene.text.Font?>


<GridPane fx:id="rootGridPane" maxHeight="-Infinity" maxWidth="-Infinity"
prefHeight="400.0" style="-fx-background-color: #D9F7F0;"
xmlns="http://javafx.com/javafx/8.0.121" xmlns:fx="http://javafx.com/fxml/1"
fx:controller="com.internsha.connectfour.Controller">
  <columnConstraints>
    <ColumnConstraints />
    <ColumnConstraints maxWidth="298.0" minWidth="225.0" prefWidth="225.0" />
  </columnConstraints>
  <rowConstraints>
    <RowConstraints minHeight="15.0" prefHeight="30.0" />
```

```
    <RowConstraints />
  </rowConstraints>
   <children>
      <Pane GridPane.columnSpan="2" />
      <Pane fx:id="insertedDiscPane" prefHeight="385.0" prefWidth="218.0"
GridPane.rowIndex="1" />
      <VBox prefWidth="323" style="-fx-background-color: #2B3B4C;"
GridPane.columnIndex="1" GridPane.rowIndex="1">
         <children>
            <TextField fx:id="playerOneTextField" focusTraversable="false" prefHeight="23.0"
prefWidth="50.0" promptText="Player One Name">
               <VBox.margin>
                  <Insets left="15.0" right="15.0" top="35.0" />
               </VBox.margin></TextField>
            <TextField fx:id="playertwoTextField" focusTraversable="false" promptText="Player
Two Name">
               <VBox.margin>
                  <Insets left="15.0" right="15.0" />
               </VBox.margin>
            </TextField>
            <Button fx:id="setNamesButton" alignment="CENTER" contentDisplay="CENTER"
mnemonicParsing="false" prefHeight="25.0" prefWidth="337.0" text="Set Names">
               <VBox.margin>
                  <Insets bottom="5.0" left="15.0" right="15.0" top="5.0" />
               </VBox.margin>
            </Button>
```

```
        <Label fx:id="playerNameLabel" alignment="CENTER" prefHeight="43.0"
prefWidth="366.0" text="Player One" textFill="#f5ebeb">
          <font>
            <Font name="System Bold" size="29.0" />
          </font>
          <VBox.margin>
            <Insets top="70.0" />
          </VBox.margin>
        </Label>
        <Label alignment="CENTER" prefHeight="40.0" prefWidth="365.0" text="Turn"
textFill="#f5eeee">
          <font>
            <Font size="28.0" />
          </font>
        </Label>
        <Region prefHeight="200.0" prefWidth="200.0" />
      </children></VBox>
  </children>
</GridPane>
```

Controller. Java

```
package com.internsha.connectfour;


import javafx.animation.TranslateTransition;
import javafx.application.Platform;
import javafx.event.ActionEvent;
import javafx.event.EventHandler;
import javafx.fxml.FXML;
import javafx.fxml.Initializable;
import javafx.geometry.Point2D;
import javafx.scene.control.Alert;
import javafx.scene.control.Button;
import javafx.scene.control.ButtonType;
import javafx.scene.control.Label;
import javafx.scene.control.TextField;
import javafx.scene.layout.GridPane;
import javafx.scene.layout.Pane;
import javafx.scene.paint.Color;
import javafx.scene.shape.Circle;
import javafx.scene.shape.Rectangle;
import javafx.scene.shape.Shape;
import javafx.util.Duration;
import sun.dc.pr.PRError;
```

```
import java.awt.*;
import java.net.URL;
import java.util.ArrayList;
import java.util.List;
import java.util.Optional;
import java.util.ResourceBundle;
import java.util.stream.Collectors;
import java.util.stream.IntStream;

public class Controller implements Initializable {
    private static final int COLUMNS = 7;
    private static final int ROWS = 6;
    private static final int CIRCLE_DIAMETER = 82;
    private static final String discColor1 = "#24303E";
    private static final String discColor2 = "#4CAA88";
    public TextField playerOneTextField, playertwoTextField;

    private static String PLAYER_ONE ;
    private static String PLAYER_TWO;

    private boolean isPlayerOneTurn = true;
    private Disc[][] insertedDiscsArray = new Disc[ROWS][COLUMNS];
```

```
    @FXML
    public GridPane rootGridPane;
    @FXML
    public Pane insertedDiscPane;
    @FXML
    public Label playerNameLabel;
    @FXML
public Button setNamesButton;

    private boolean isAllowedtoInsert = true;

    public void createPlayground() {
        Shape rectangleWithholes = createGridStructuralGrid();



        rootGridPane.add(rectangleWithholes, 0, 1);
        List<Rectangle> rectangleList = createClickableColumns();
        for (Rectangle rectangle : rectangleList) {
            rootGridPane.add(rectangle, 0, 1);
        }
setNamesButton.setOnAction(event -> setNames());
    }
```

```
    private void setNames() { // Set Player one and player two names
        PLAYER_ONE = playerOneTextField.getText();
        PLAYER_TWO = playertwoTextField.getText();
        if (isPlayerOneTurn)

            playerNameLabel.setText(PLAYER_ONE);
        else
            playerNameLabel.setText(PLAYER_TWO);
    }

    private Shape createGridStructuralGrid() {
        Shape rectangleWithholes = new Rectangle((COLUMNS + 1) *
CIRCLE_DIAMETER, (ROWS + 1) * CIRCLE_DIAMETER);
        for (int row = 0; row < ROWS; row++) {
            for (int col = 0; col < COLUMNS; col++) {
                Circle circle = new Circle();
                circle.setRadius(CIRCLE_DIAMETER / 2);
                circle.setCenterX(CIRCLE_DIAMETER / 2);
                circle.setCenterY(CIRCLE_DIAMETER / 2);
                circle.setSmooth(true);
                circle.setTranslateX(col * (CIRCLE_DIAMETER + 5) +
CIRCLE_DIAMETER / 4);
```

```
            circle.setTranslateY(row * (CIRCLE_DIAMETER + 5) +
CIRCLE_DIAMETER / 4);
            rectangleWithholes = Shape.subtract(rectangleWithholes, circle);
        }
}
    rectangleWithholes.setFill(Color.WHITE);
    return rectangleWithholes;
  }

  private List<Rectangle> createClickableColumns() {
    List<Rectangle> rectangleList = new ArrayList<>();
    for (int col = 0; col < COLUMNS; col++) {
        Rectangle rectangle = new Rectangle(CIRCLE_DIAMETER, (ROWS + 1)
* CIRCLE_DIAMETER);
        rectangle.setFill(Color.TRANSPARENT);
        rectangle.setTranslateX(col * (CIRCLE_DIAMETER + 5) +
CIRCLE_DIAMETER / 4);
        rectangle.setOnMouseEntered(event ->
rectangle.setFill(Color.valueOf("#eeeeee26")));
        rectangle.setOnMouseExited(event ->
rectangle.setFill(Color.TRANSPARENT));
        final int column = col;
        rectangle.setOnMouseClicked(event -> {
            if (isAllowedtoInsert) {
```

```
                isAllowedtoInsert = false;

                insertDisc(new Disc(isPlayerOneTurn), column);
            }
        });

        rectangleList.add(rectangle);
    }
return rectangleList;
  }

  private void insertDisc(Disc disc, int column) {
    int row = ROWS - 1;
    while (row >= 0) {
        if ( getDiscIfPresent(row,column)== null)
            break;
        row--;
    }
    if (row < 0)
        return;
    insertedDiscsArray[row][column] = disc;
    insertedDiscPane.getChildren().add(disc);
```

```
        disc.setTranslateX(column * (CIRCLE_DIAMETER + 5) +
CIRCLE_DIAMETER / 4);

        int currentRow = row;

        TranslateTransition translateTransition = new
TranslateTransition(Duration.seconds(0.5), disc);

        translateTransition.setToY(row * (CIRCLE_DIAMETER + 5) +
CIRCLE_DIAMETER / 4);

        translateTransition.setOnFinished(event -> {

            isAllowedtoInsert=true;

            if (gameEnded(currentRow, column)) {

            gameOver();

            return;

            }

            isPlayerOneTurn = !isPlayerOneTurn;

            playerNameLabel.setText(isPlayerOneTurn ? PLAYER_ONE :
PLAYER_TWO);

        } );

        translateTransition.play();

    }


    private boolean gameEnded(int row, int column) {

        List<Point2D> verticalPoints =  IntStream.rangeClosed(row-3,row+3)

                .mapToObj(r->new Point2D(r,column))

                .collect(Collectors.toList());
```

```
List<Point2D> horizontalPoints =  IntStream.rangeClosed(column-3,column+3)
        .mapToObj(col->new Point2D(row,col))
        .collect(Collectors.toList());

Point2D startPoint1 = new Point2D(row-3,column+3);

List<Point2D> diagnol1Points = IntStream
        .rangeClosed(0,6)
        .mapToObj(i->startPoint1.add(i,-i)).
            collect(Collectors.toList());

Point2D startPoint2 = new Point2D(row-3,column-3);

List<Point2D> diagnol2Points = IntStream
        .rangeClosed(0,6)
        .mapToObj(i->startPoint2.add(i,i)).
            collect(Collectors.toList());

boolean isEnded = checkCombinations(verticalPoints)|| checkCombinations(horizontalPoints)
        || checkCombinations(diagnol1Points) || checkCombinations(diagnol2Points);
```

```
        return isEnded;
    }

    private boolean checkCombinations(List<Point2D> points) {
        int chain =0;
        for (Point2D point : points)
        {
            int rowIndexForArray = (int) point.getX();
            int columnIndexForArray = (int) point.getY();

            Disc disc = getDiscIfPresent(rowIndexForArray,columnIndexForArray);
            if(disc!=null && disc.isPlayerOneMove==isPlayerOneTurn)
            {
                chain++;
                if (chain==4)
                {
                  return true;
                }
            } else
            {
                chain=0;
            }
        }
```

```
        return false;
    }
private Disc getDiscIfPresent(int row, int column)
{
    if(row>=ROWS || row<0 || column>=COLUMNS || column<0 )
        return null;

    return insertedDiscsArray[row][column];
}
    private void gameOver()
    {
       String winner = isPlayerOneTurn? PLAYER_ONE:PLAYER_TWO;
        System.out.println("Winner is:"+"    "+winner);

        Alert alert = new Alert(Alert.AlertType.INFORMATION);
        alert.setTitle("Connect Four");
        alert.setHeaderText("The Winner is :" +"    " +winner);
        alert.setContentText("Want to Play Again?");
        ButtonType yesButton = new ButtonType("Yes");
        ButtonType noButton = new ButtonType("No, Exit");
        alert.getButtonTypes().setAll(yesButton,noButton);
            Platform.runLater(()->{
                Optional<ButtonType> buttonClicked = alert.showAndWait();
```

```
            if (buttonClicked.isPresent()&& buttonClicked.get()==yesButton)
            {
                resetGame();
            }
            else
            {
                Platform.exit();
                System.exit(0);
            }

        });
}

    public void resetGame() {
        insertedDiscPane.getChildren().clear();
        for (int row =0; row<insertedDiscsArray.length; row++)
        {
            for (int col =0; col<insertedDiscsArray[row].length; col++)
            {
              insertedDiscsArray[row][col]=null;
            }
        }
        isPlayerOneTurn=true;
```

```
        playerNameLabel.setText(PLAYER_ONE);
        createPlayground();
    }


    private static class Disc extends  Circle
    {
        private final boolean isPlayerOneMove;


        public Disc(boolean isPlayerOneMove)
        {
            this.isPlayerOneMove = isPlayerOneMove;
            setRadius(CIRCLE_DIAMETER/2);
            setFill(isPlayerOneMove?
Color.valueOf(discColor1):Color.valueOf(discColor2));
            setCenterX(CIRCLE_DIAMETER/2);
            setCenterY(CIRCLE_DIAMETER/2);


        }
    }
    @Override
    public void initialize(URL location, ResourceBundle resources) {
```

```
    }
}
```

Main. java

```
 package com.internsha.connectfour;

import javafx.application.Application;
import javafx.application.Platform;
import javafx.event.ActionEvent;
import javafx.event.EventHandler;
import javafx.fxml.FXML;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.scene.control.*;
import javafx.scene.layout.GridPane;
import javafx.scene.layout.Pane;
import javafx.stage.Stage;

public class Main extends Application {
```

```
    private Controller controller;


    @Override
    public void start(Stage primaryStage) throws Exception{
        FXMLLoader fxmlLoader = new
FXMLLoader(getClass().getResource("game.fxml"));
        GridPane rootGridPane = fxmlLoader.load();
       controller= fxmlLoader.getController();
       controller.createPlayground();
       MenuBar menuBar = createMenu();
        Pane menuPane = (Pane) rootGridPane.getChildren().get(0);
        menuPane.getChildren().addAll(menuBar);
        menuBar.prefWidthProperty().bind(primaryStage.widthProperty());
       Scene scene = new Scene(rootGridPane);
       primaryStage.setScene(scene);
       primaryStage.setTitle("Connect Four");
       primaryStage.setResizable(false);
       primaryStage.show();
    }
    private MenuBar createMenu()
    {
     // File Menu
        Menu fileMenu = new Menu("File");
```

```
        MenuItem newGame = new MenuItem("New Game");


        newGame.setOnAction(event -> controller.resetGame());
        MenuItem resetGame = new MenuItem("Reset Game");
        resetGame.setOnAction(event ->  controller.resetGame());
        SeparatorMenuItem separatorMenuItem = new SeparatorMenuItem();
        MenuItem exitGame = new MenuItem("Exit Game");
        exitGame.setOnAction(event -> exitGame());

fileMenu.getItems().addAll(newGame,resetGame,separatorMenuItem,exitGame);
        // Help Menu
        Menu helpMenu = new Menu("Help");
        MenuItem aboutGame = new MenuItem("About Connect 4");
        aboutGame.setOnAction(event -> aboutConnect4());
        SeparatorMenuItem smi = new SeparatorMenuItem();
        MenuItem aboutMe = new MenuItem("About Me");
        aboutMe.setOnAction(new EventHandler<ActionEvent>() {
            @Override
            public void handle(ActionEvent event) {
                aboutMe();
            }
        });
        helpMenu.getItems().addAll(aboutGame,smi,aboutMe);
```

```
        MenuBar menuBar = new MenuBar();
        menuBar.getMenus().addAll(fileMenu,helpMenu);
        return menuBar;

    }

    private void aboutMe() {
        Alert alert = new Alert(Alert.AlertType.INFORMATION);
        alert.setTitle("About the Developer");
        alert.setHeaderText("Mohammed Asfaar Uddin Shareef");
        alert.setContentText("Developing Applications since 2019.");
        alert.show();
    }

    private void aboutConnect4() {
        Alert alert = new Alert(Alert.AlertType.INFORMATION);
        alert.setTitle("About Connect Four");
        alert.setHeaderText("How to Play?");
        alert.setContentText("Connect Four is a two-player connection game in which
the players first choose a color and then take turns dropping colored discs from the
top into a seven-column, six-row vertically suspended grid. The pieces fall straight
down, occupying the next available space within the column. The objective of the
game is to be the first to form a horizontal, vertical, or diagonal line of four of
one's own discs. Connect Four is a solved game. The first player can always win by
playing the right moves.");
```

```
        alert.show();
    }

    private void exitGame() {
        Platform.exit();
        System.exit(0);
    }



    public static void main(String[] args) {
        launch(args);
    }
}
```