JAVA FX: Scenario: Breakout Video Game

Code

```
package com.example.breakout;

import javafx.application.*;
import javafx.scene.*;
import javafx.scene.control.*;
import javafx.stage.Stage;
import javafx.animation.*;
import javafx.event.*;
import javafx.util.Duration;
import java.util.*;
import javafx.scene.image.Image;
import javafx.scene.input.*;
import javafx.scene.layout.*;
import javafx.scene.paint.Color;
import javafx.scene.shape.*;
import javafx.scene.text.Font;
import javafx.scene.text.FontPosture;
import javafx.scene.text.FontWeight;
import javafx.scene.transform.Scale;

public class HelloApplication extends Application {

    @Override
    public void start(Stage window) {
///-----------------------------------------------Stage 1-------------------------------
--------------------///

    GridPane root = new GridPane();
    root.setHgap(60);
    root.setVgap(60);

    Scale scaleTransformation = new Scale();
    scaleTransformation.setX(2.0);
    scaleTransformation.setY(2.0);
    scaleTransformation.setPivotX(0);
    scaleTransformation.setPivotY(0);
    Font font = Font.font("Courier New", FontWeight.BOLD, 10);

    //creating Menu components
    Button b = new Button("Start");
    b.setStyle("-fx-background-color: #ff0000; ");
    b.setMaxWidth(50);
    b.setMaxHeight(50);
    b.setPrefWidth(200);
    b.setFont(font);
    b.getTransforms().add(scaleTransformation);

    Button b4 = new Button("Help");
    b4.setStyle("-fx-background-color: #ff2785; ");
    b4.setMaxWidth(50);
```

```
    b4.setMaxHeight(50);
    b4.setPrefWidth(200);
    b4.getTransforms().add(scaleTransformation);
    b4.setFont(font);

    Button b5 = new Button("Close");
    b5.setStyle("-fx-background-color: #ff7548; ");
    b5.setMaxWidth(50);
    b5.setMaxHeight(50);
    b5.setPrefWidth(200);
    b5.getTransforms().add(scaleTransformation);
    b5.setFont(font);

    Label lb3 = new Label();
    lb3.setTextFill(Color.RED);
    lb3.setFont(Font.font("Verdana", FontWeight.BOLD, FontPosture.REGULAR, 12));

    Label plbl = new Label(); //label for stage 1 to display loading message near the
progress bar
    plbl.setTextFill(Color.RED);
    plbl.setFont(Font.font("Verdana", FontWeight.BOLD, FontPosture.REGULAR, 12));


    //setting button position on GridPane
    root.add(b, 0, 1);
    root.add(b4, 0, 2);
    root.add(b5, 0, 3);
    root.add(lb3, 2, 5);//adding label to GirdPane


      Image img = new Image("G:/Semester_3/OOP
Lab/Project/Breakout_Group_Assignment_02/pics/2resized.png"); //for ghufran PC
   // Image img = new
Image("U://breakout//src//main//resources//images//2resized.png");//for faheem PC
    BackgroundImage bImg = new BackgroundImage(img,
            BackgroundRepeat.NO_REPEAT,
            BackgroundRepeat.NO_REPEAT,
            BackgroundPosition.DEFAULT,
            BackgroundSize.DEFAULT
    );
    Background bGround = new Background(bImg);
    root.setBackground(bGround);

    window.setScene(new Scene(root, 610, 500));
    window.setTitle("MENU");
    window.setResizable(false);
    window.show();

    ///-----------------------------------------Stage 2----------------------------
---------------///
    Stage second = new Stage();
    Pane layout = new Pane();
    layout.setPrefSize(610, 500);
```

```
     Image img2 = new Image("G:/Semester_3/OOP
Lab/Project/Breakout_Group_Assignment_02/pics/resized1.png");//for ghufranpc
    //Image img2 = new
Image("U://breakout//src//main//resources//images//resized1.png");//for faheem pc
    BackgroundImage bImg2 = new BackgroundImage(img2,
            BackgroundRepeat.NO_REPEAT,
            BackgroundRepeat.NO_REPEAT,
            BackgroundPosition.DEFAULT,
            BackgroundSize.DEFAULT);
    Background bGround2 = new Background(bImg2);
    layout.setBackground(bGround2);

    //code for creating bricks with specific color
    ArrayList<Rectangle> allBricks = new ArrayList<>();
    for (int x = 0; x < 10; x++) {
        for (int y = 0; y < 8; y++) {
            Rectangle brick = new Rectangle(60, 15);
            if (y < 8) {
                switch (x) {
                    case 0:
                    case 3:
                    case 6:
                    case 9:
                        brick.setFill(Color.GREEN);
                        break;
                    case 1:
                    case 4:
                    case 7:
                        brick.setFill(Color.YELLOW);
                        break;
                    case 2:
                    case 5:
                    case 8:
                        brick.setFill(Color.RED);
                        break;
                }
            }
            brick.setLayoutX(x * 62);
            brick.setLayoutY((16 * y) + 35);
            layout.getChildren().add(brick);
            allBricks.add(brick);
        }
    }

    Circle ball = new Circle(20, 20, 10, Color.BLUE);
    ball.relocate(300, 200);//starting loc of ball

    Rectangle paddle = new Rectangle(90, 7, Color.ORANGERED);
    paddle.relocate(275, 390); //starting location of paddle

    Label lb1 = new Label(); //label for displaying score and lives
    lb1.setTextFill(Color.RED);
    lb1.setFont(Font.font("Verdana", FontWeight.BOLD, FontPosture.REGULAR, 20));
```

```
    Line line = new Line(); //line below the paddle,if the ball touch it ,we will
lose the life
    line.setStartX(0);
    line.setStartY(400);
    line.setEndX(610);
    line.setEndY(400);
    line.setStroke(Color.RED);

    layout.getChildren().addAll(lb1, paddle, ball, line);//adding nodes to the pane

    Scene view = new Scene(layout);
    second.setTitle("Breakout Game!");
    second.setScene(view);
    second.setResizable(false);

    ///-------------------------------------------------Stage 3--------------------
-----------///
    Stage Third = new Stage();
    StackPane root2 = new StackPane();

    Label lb2 = new Label();//label for displaying game output
    lb2.setTextFill(Color.RED);
    lb2.setFont(Font.font("Verdana", FontWeight.BOLD, FontPosture.REGULAR, 20));
    root2.getChildren().add(lb2);

    Image img3 = new Image("G:/Semester_3/OOP
Lab/Project/Breakout_Group_Assignment_02/pics/last stage.png");//for ghufran pc
    //Image img3 = new
Image("U://breakout//src//main//resources//images//last_stage_610x500.png");//for
faheem pc

    BackgroundImage bImg3 = new BackgroundImage(img3,
            BackgroundRepeat.NO_REPEAT,
            BackgroundRepeat.NO_REPEAT,
            BackgroundPosition.DEFAULT,
            BackgroundSize.DEFAULT);
    Background bGround3 = new Background(bImg3);
    root2.setBackground(bGround3);

    Third.setScene(new Scene(root2, 610, 500));
    Third.setTitle("RESULT"); // Set the stage title // Set a scene with a button in
the stage
    Third.setResizable(false);

    ///----------------------------------handling for Stage 1 ----------------------
--------------------///

    b.setOnAction(e -> {//event handler for the start button

       ProgressBar progressBar = new ProgressBar();//progress bar
         progressBar.setPrefSize(150,20);
         root.add(progressBar,14,0);//adding to gridpane

         root.setHgap(10);//setting hgap of grid pane when button will be pressed
         plbl.setText("Loading ...!");
```

```
        root.add(plbl,10,0);//adding label plbl to gridpane

        //pause transition to pause the stage 1 for 5 second and then display stage 2
        PauseTransition delay = new PauseTransition(Duration.seconds(5));
        delay.setOnFinished( event -> second.show() );
        delay.play();

    });

    b4.setOnAction(e -> {//event handler for the help button
        lb3.setText("--Click on Start Button to play Game--\n--Move paddle so game
will start--"
                + "\n*--You have 3 lives--\n--Score as much as you can to Win--\n--
Have Fun!--*");
    });

    b5.setOnAction(e -> {//event handler for the close button
        window.close();
    });

    ///----------------------------Handling for stage 2----------------------------
---///
    //controls paddle movement
    int movement = 20;
    //creates an indefinite bouncing ball
    Timeline timeline = new Timeline(new KeyFrame(Duration.millis(20), new
EventHandler<ActionEvent>() {
        boolean var = false;
        int lives = 3;
        int score;

        double dx = 3;
        double dy = 2;

        @Override
        public void handle(ActionEvent t) {

            //ball movement
            if (allBricks.isEmpty()) {

                layout.getChildren().removeAll(ball, paddle);

                lb2.setText("CONGRATULATION YOU WON THE GAME");
                Third.show();

            }

            if (var) {

                ball.setLayoutX(ball.getLayoutX() + dx);
                ball.setLayoutY(ball.getLayoutY() + dy);

                boolean leftWall = ball.getLayoutX() <= 0;
                boolean topWall = ball.getLayoutY() < 35;
                boolean rightWall = ball.getLayoutX() >= 590;
```

```
                boolean bottomWall = ball.getLayoutY() >= 380;

                // If the top wall has been touched, the ball reverses direction.
                if (topWall) {
                    dy = dy * -1;
                }

                // If the left or right wall has been touched, the ball reverses
direction.
                if (leftWall || rightWall) {
                    dx = dx * -1;
                }
                //if ball hit bottom wall, relocate paddle and ball to its original
position
                if (bottomWall) {

                    ball.relocate(300, 200);
                    paddle.relocate(275, 390);

                    var = false;

                    lives--; //decrementing lives

                    if (lives < 3) {//setting powerup paddle to its original position
                        paddle.setWidth(90);
                    }

                    if (lives < 1) {
                        lb2.setText("Lives = " + lives + " Score = " + score + "\nYou
Lost the Game");
                        root.getChildren().removeAll(ball, paddle);
                        Third.show();
                    }
                }
            }

            if (!(paddle.getLayoutX() == 275.0 && !var)) {
                var = true;
            }

            //if ball collides with paddle,reverse ball direction
            if (collide(paddle)) {
                dy = -dy;
            }

            //if ball and brick collides, remove brick
            Rectangle temp = null;

            for (Rectangle brick : allBricks) {

                if (collide(brick)) {

                    dy = -dy;//reversing ball direction

                    if (brick.getFill().equals(Color.GREEN)) {
```

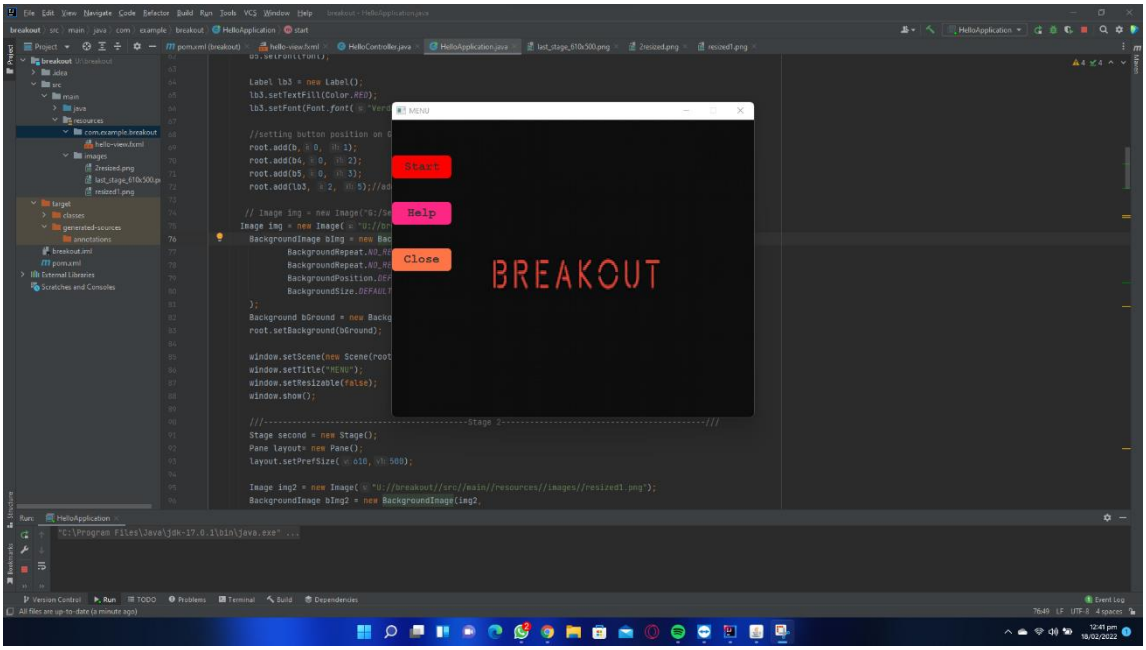
```
                        brick.setFill(Color.YELLOW);
                        paddle.setWidth(120);   //when ball hits Green brick, paddle
will increase (POWER UP) its width from 90 to 120
                    } else if (brick.getFill().equals(Color.YELLOW)) {
                        brick.setFill(Color.RED);
                    } else if (brick.getFill().equals(Color.RED)) {
                        score += 1;
                        layout.getChildren().remove(brick);
                        temp = brick;
                    }
                }
            }

            lb1.setText("Score = " + score + " Lives = " + lives);//Displaying lives
and score side by side on stage 2

            if (!(temp == null)) {
                allBricks.remove(temp);
                temp = null;
            }
        }

        public boolean collide(Rectangle other) {
            Shape collisionArea = Shape.intersect(ball, other);
            return collisionArea.getBoundsInLocal().getWidth() != -1;
        }
    }));

    timeline.setCycleCount(Timeline.INDEFINITE);

    view.setOnKeyPressed(event -> { // button pressed  on the view scene

        timeline.play(); //ball will start moving if any  of the button is pressed

        if (event.getCode() == KeyCode.LEFT) {//this will make the paddle to move
left and right

            if (paddle.getLayoutX() < 0) {//restricting paddle from going quite left
means less then 0
                paddle.setLayoutX(paddle.getLayoutX() + movement);
            }

            paddle.setLayoutX(paddle.getLayoutX() - movement);

        }

        if (event.getCode() == KeyCode.RIGHT) {

            if (paddle.getLayoutX() > 510) {//restricting paddle from going quite
right means less then 610
                paddle.setLayoutX(510);
            }

            paddle.setLayoutX(paddle.getLayoutX() + movement);
        }
```

```
        });

        }

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

    }
```

Menu

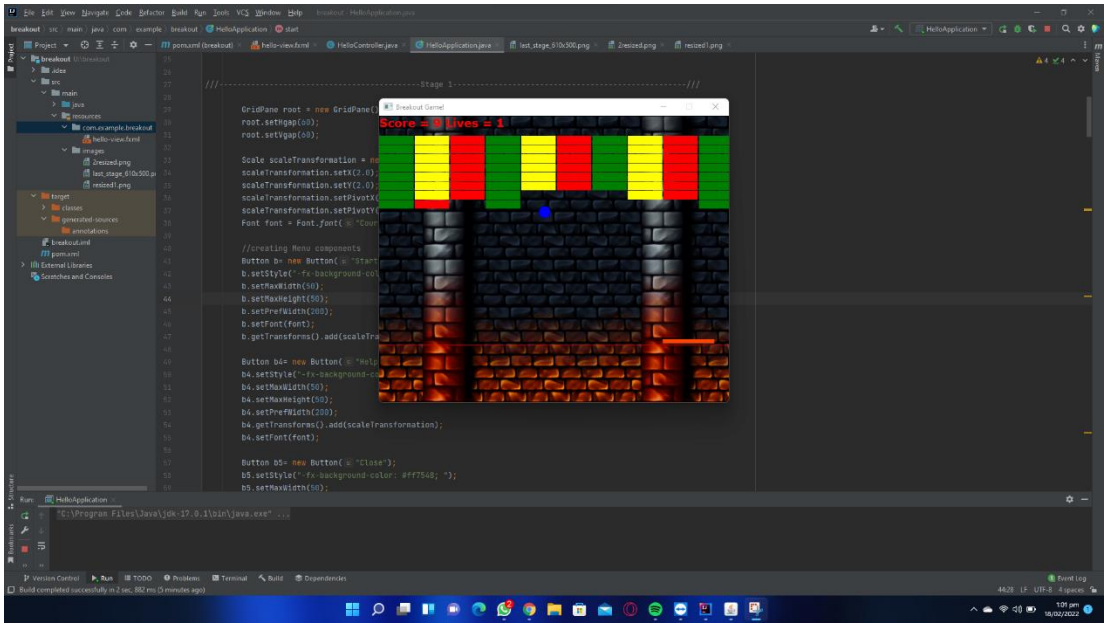
You can see three buttons on this stage.

- Clicking on **Start** button will start the game.
- Clicking on **Help** button will display some instructions about game.
- Clicking on **Close** button will exit the game immediately.



Breakout Game!

Game will begin when you click on start button, move paddle to initiate the ball movement. You can clearly see score which is equal to the number of bricks you have broken and lives you have left. Player will lose life when ball will touch bottom wall.



Color System!

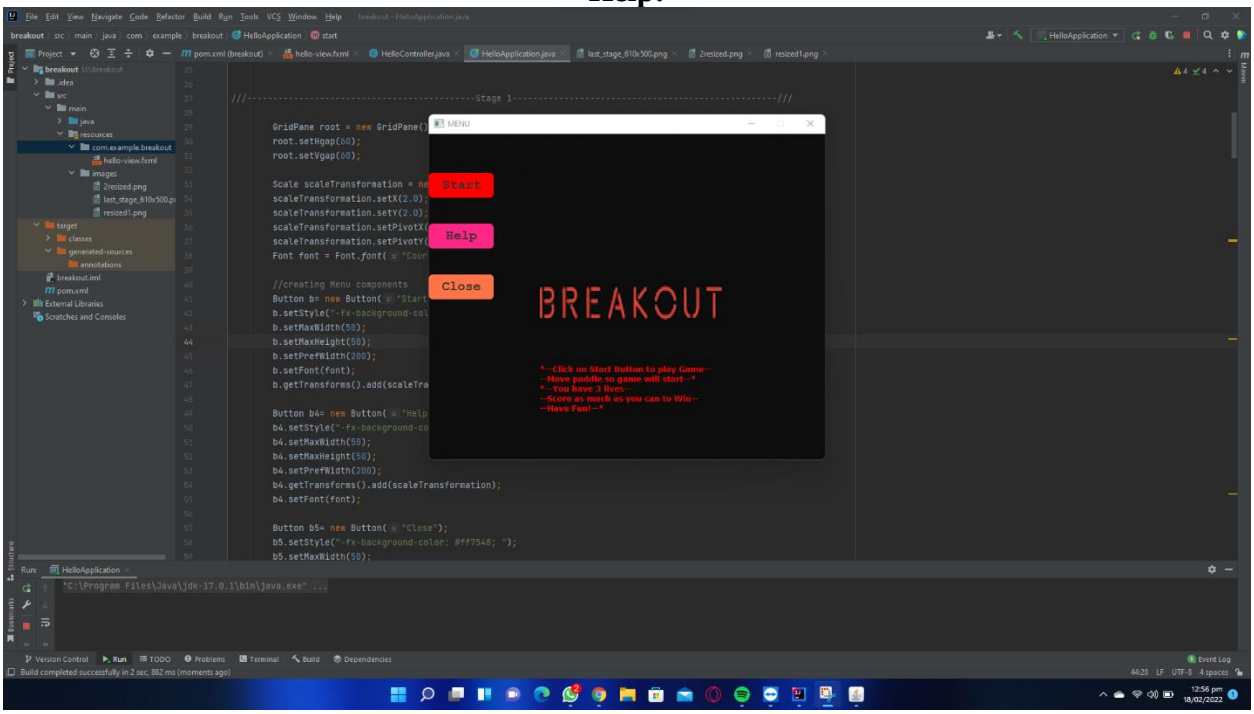
We added three color balls according to requirements.

- **Green ball has high strength**. When ball hit Green color, it will change its color to Yellow and when ball hits Yellow it will change its color to Red and third time when ball will hit Red it will be broken. Powerup system also added in this condition.
- **Yellow color has medium strength**. It will first change to Red and again if ball will hit red, it will be broken.
- **Red has Low strength**. It will break when ball hits one time. And Score will increment by 1.

For this we have added conditions that ball will hit Green color then change its color by brick.setFill(Color.RED);

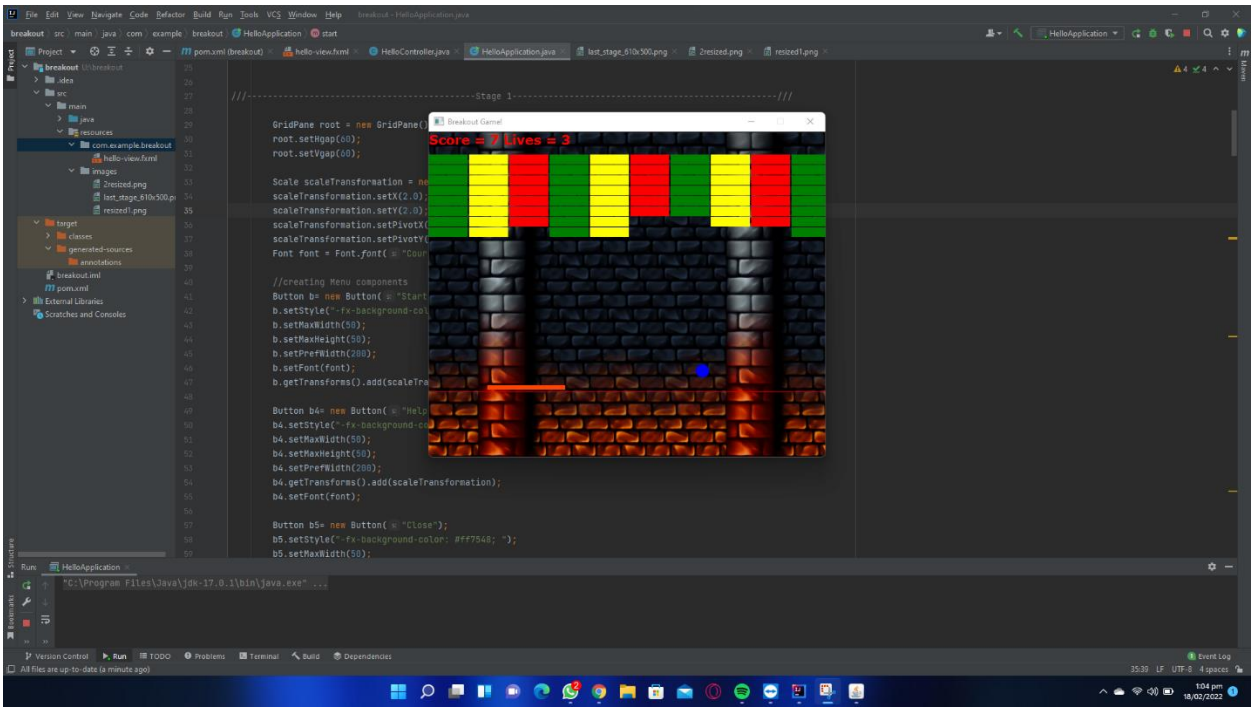
```
if (brick.getFill().equals(Color.GREEN)){
                            brick.setFill(Color.YELLOW);
                            paddle.setWidth(120);   //when ball hits Green brick,
paddle will increase (POWER UP) its width from 90 to 120
                        }else if(brick.getFill().equals(Color.YELLOW)) {
                            brick.setFill(Color.RED);
                        }
                        else if(brick.getFill().equals(Color.RED)) {
                            score+=1;
                            layout.getChildren().remove(brick);
                            temp=brick;
                        }
```

Help!



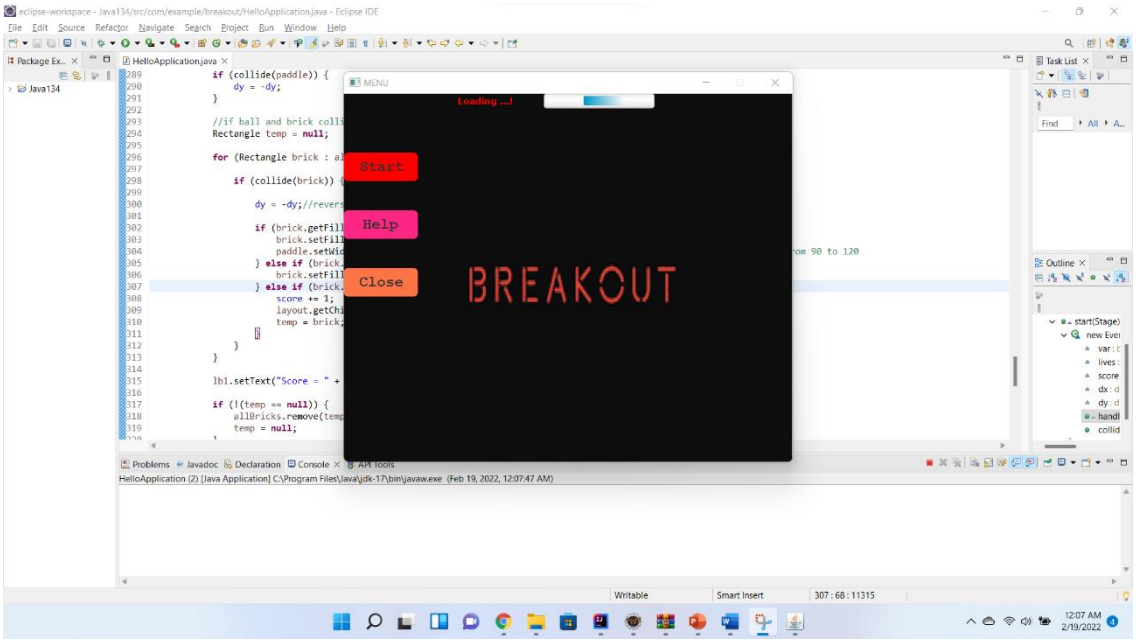
When ball will hit the green ball then paddle width will increase **(bigger paddle)** and when player lose any life paddle will come to its orginal width.

PowerUP!



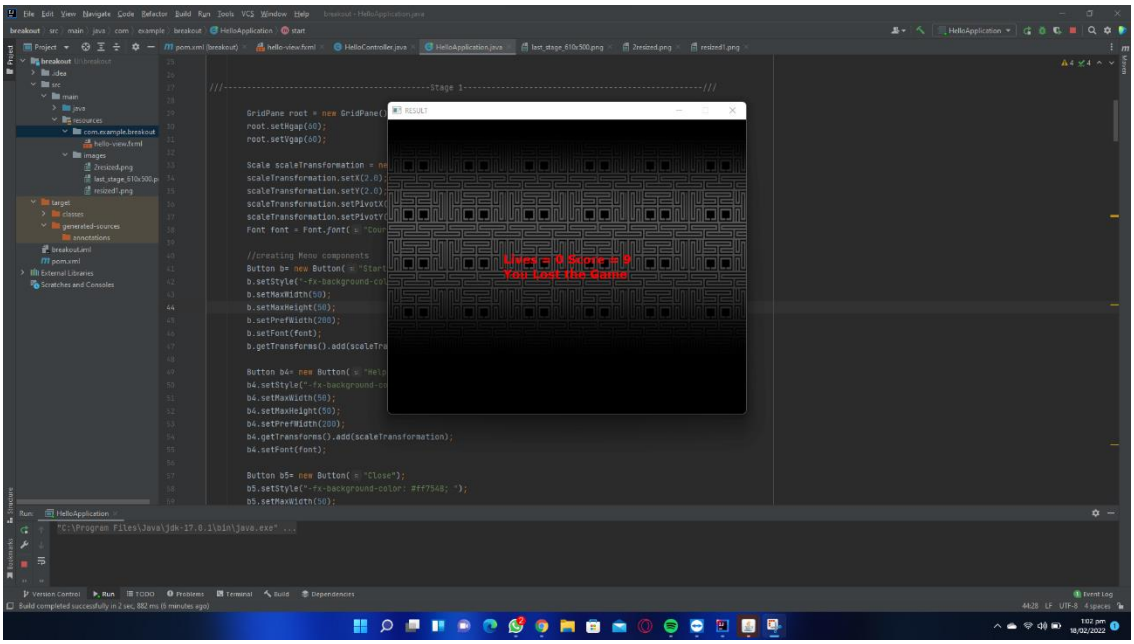
Progress Bar!

When player will click on start button progress bar will appear for 5 seconf then second stage will appear.



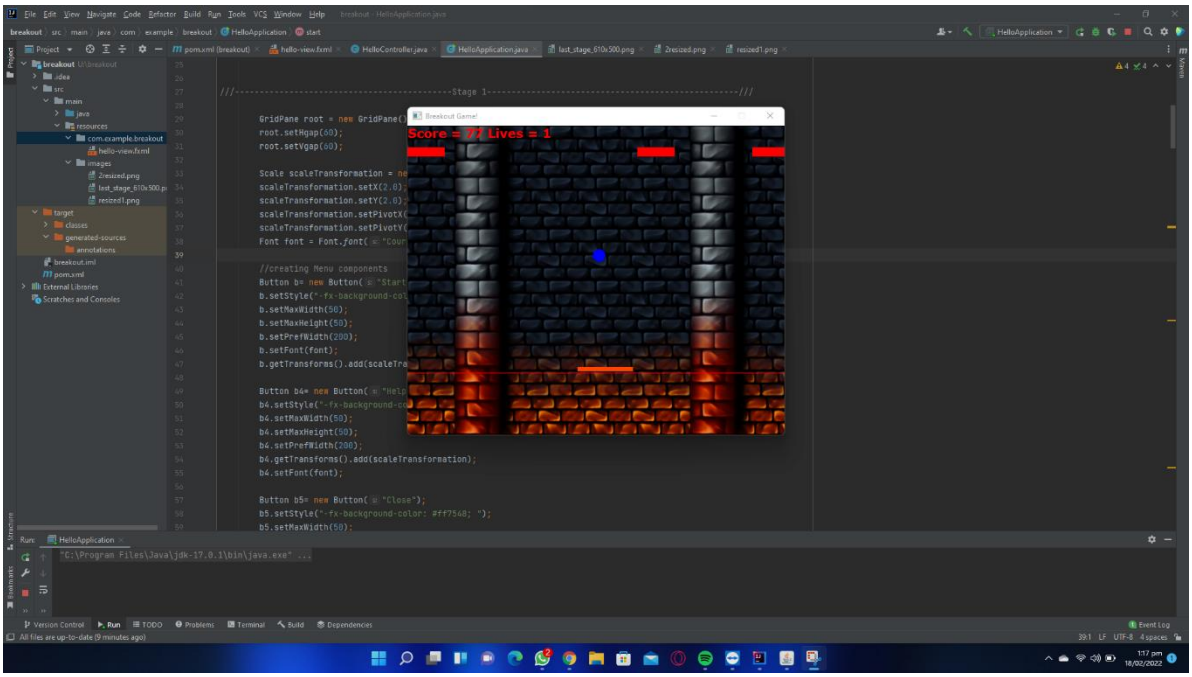
Result Stage

When player lose all of his/her lives then stage three will appear which will be showing your current score and current live and a message.



All bricks broken!

When all bricks will be broken, Stage three will appear which will be showing a message of congratulations.



Congratulation message!