INDRAPRASTHA INSTITUTE *of*
INFORMATION TECHNOLOGY
DELHI

Department
of
Computer Science & Engineering

Subject

AP ENDSEM PROJECT

Authors:

Aditya Upadhyay (2022040)
Sanyam Barwar (2022447)

This Is a write up for game made Stick Hero.

We have made this using JavaFx

```
package com.example.stick_hero_final_project;

//import com.sun.javafx.menu.MenuItemBase;
import javafx.animation.*;
import javafx.application.Platform;
import javafx.event.ActionEvent;
import javafx.event.EventHandler;
import javafx.fxml.FXML;
import javafx.fxml.FXMLLoader;
import javafx.scene.Node;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.control.Label;
import javafx.scene.image.Image;
import javafx.scene.image.ImageView;
import javafx.scene.input.KeyCode;
import javafx.scene.input.MouseEvent;
import javafx.scene.layout.*;
import javafx.scene.paint.Color;
import javafx.scene.shape.Rectangle;
import javafx.stage.Stage;
import javafx.util.Duration;
import javafx.scene.media.Media;
import javafx.scene.media.MediaPlayer;

import java.io.*;
import java.util.ArrayList;
import java.util.Random;
import java.util.concurrent.CountDownLatch;
import java.util.concurrent.atomic.AtomicInteger;
import java.util.concurrent.atomic.AtomicReference;

import static java.lang.Thread.sleep;
```

These are the imports which we need to work upon the Project, we also mentioned these in our Pom.xml.

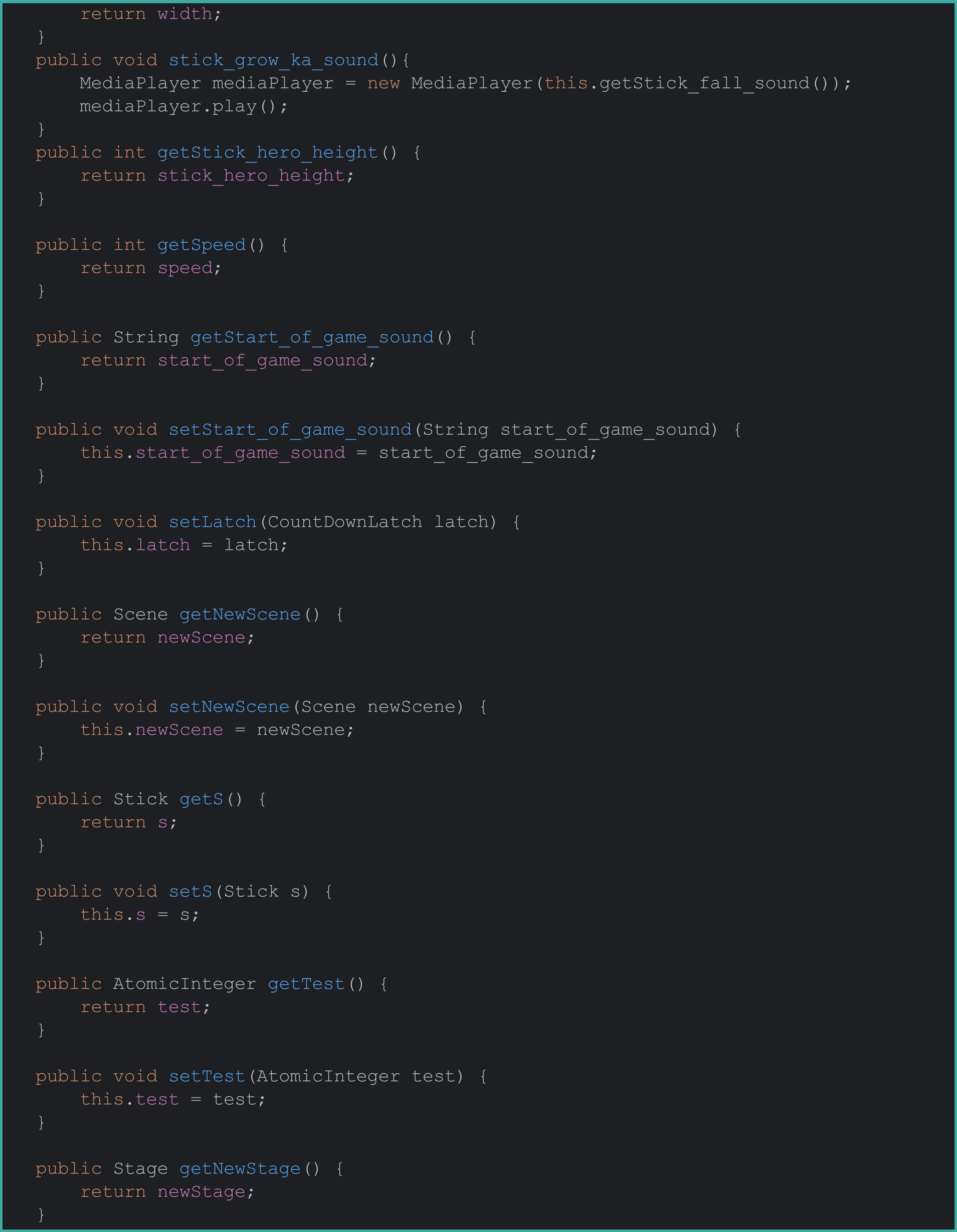
The Pom.xml contains the dependencies to be needed in the code.

It is simple code extending Application Class that is generated to work upon.

```java
package com.example.stick_hero_final_project;

import javafx.application.Application;
import javafx.fxml.FXMLLoader;
import javafx.scene.Scene;
import javafx.stage.Stage;

import java.io.IOException;

public class Game extends Application {
   @Override
   public void start(Stage stage) throws IOException {
       FXMLLoader fxmlLoader = new
FXMLLoader(Game.class.getResource("hello-view.fxml"));
       Scene scene = new Scene(fxmlLoader.load(), 320, 240);
       stage.setTitle("Stick Hero Ninja !!!");
       stage.setScene(scene);
       stage.setResizable(true);
       stage.setOnCloseRequest(event -> {
           // Perform actions when the user attempts to close the window will cork
when press red cross buttomn although not needed
           System.out.println("Closing the application...");

       });
       stage.setWidth(600);
       stage.setHeight(800);
       stage.setMaxWidth(600);
       stage.setMaxHeight(730);
//         stage.lo
       stage.show();
   }

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

   @Override
   public String toString() {
       return "Main{}";
   }
}
```

the main here is simple that can be run directly from the terminal.

```java
package com.example.stick_hero_final_project;

//import com.sun.javafx.menu.MenuItemBase;
import javafx.animation.*;
import javafx.application.Platform;
import javafx.event.ActionEvent;
import javafx.event.EventHandler;
import javafx.fxml.FXML;
import javafx.fxml.FXMLLoader;
import javafx.scene.Node;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.control.Label;
import javafx.scene.image.Image;
import javafx.scene.image.ImageView;
import javafx.scene.input.KeyCode;
import javafx.scene.input.MouseEvent;
import javafx.scene.layout.*;
import javafx.scene.paint.Color;
import javafx.scene.shape.Rectangle;
import javafx.stage.Stage;
import javafx.util.Duration;
import javafx.scene.media.Media;
import javafx.scene.media.MediaPlayer;

import java.io.*;
import java.util.ArrayList;
import java.util.Random;
import java.util.concurrent.CountDownLatch;
import java.util.concurrent.atomic.AtomicInteger;
import java.util.concurrent.atomic.AtomicReference;

import static java.lang.Thread.sleep;

public class StickHero extends Thread implements ScoreInterface, Cherries,
Points,Serializable{
   public void run() {
       try {
           back_create();
       } catch (IOException e) {
           throw new RuntimeException(e);
       }
   }
   public Rectangle rect123;
   public int getStick_speed_fllag() {
       return stick_speed_fllag;
   }

   public void setStick_speed_fllag(int stick_speed_fllag) {
       this.stick_speed_fllag = stick_speed_fllag;
   }
   private int postion_face = 0; // 0 means up and 1 means down
   private boolean load;
   //    private
   private int stick_speed_fllag = 0;
   private ImageView cherry_1;
   private String start_of_game_sound =
"src/Main/resources/com/example/stick_hero_final_project/Sounds/Start_of_Game.mp4";

//"D:\Stick_Hero_Final_Project\src\Main\resources\com\example\stick_hero_final_project
\Sounds\Start_of_Game.mp4"
   private String stick_grow_sound =
("src/Main/resources/com/example/stick_hero_final_project/Sounds/stick_grow_loop.wav")
;

   public Rectangle getRect123() {
       return rect123;
   }

   public void setRect123(Rectangle rect123) {
       this.rect123 = rect123;
   }

   public String getStick_grow_sound() {
       return stick_grow_sound;
   }

   public void setStick_grow_sound(String stick_grow_sound) {
       this.stick_grow_sound = stick_grow_sound;
   }

   public Rectangle getLayoutforcherry() {
       return layoutforcherry;
   }

   public void setLayoutforcherry(Rectangle layoutforcherry) {
       this.layoutforcherry = layoutforcherry;
   }

   public String getStart_of_game() {
       return start_of_game_sound;
   }
   private CountDownLatch latch = new CountDownLatch(0);
   private Scene newScene;
   Stick s = new Stick(10, 10, 5, 0);
   int brahmastra = 0; //to move synchronous
   public void setStart_of_game(String start_of_game_sound) {
       this.start_of_game_sound = start_of_game_sound;
   }

   public CountDownLatch getLatch() {
       return latch;
   }

   public void setLatch() {
       latch = new CountDownLatch(1);
   }

   public void setTest(int test) {
       this.test.set(test);
   }

   private AtomicInteger test = new AtomicInteger();

   public Media getSound() {
       return sound;
   }

   public void setSound(Media sound) {
       this.sound = sound;
   }

   private Media sound = new Media(new File(start_of_game_sound).toURI().toString());

   public Media getStick_fall_sound() {
       return stick_fall_sound;
   }

   public void setStick_fall_sound(Media stick_fall_sound) {
       this.stick_fall_sound = stick_fall_sound;
   }

   private Media stick_fall_sound =  new Media(new
File(stick_grow_sound).toURI().toString());
   private PlayerCreate player;
   private final int height = 80;

   public PlayerCreate getPlayer() {
       return player;
   }

   public void setPlayer(PlayerCreate player) {
       this.player = player;
   }

   public int getHeight() {
       return height;
   }

   public int getWidth() {
```

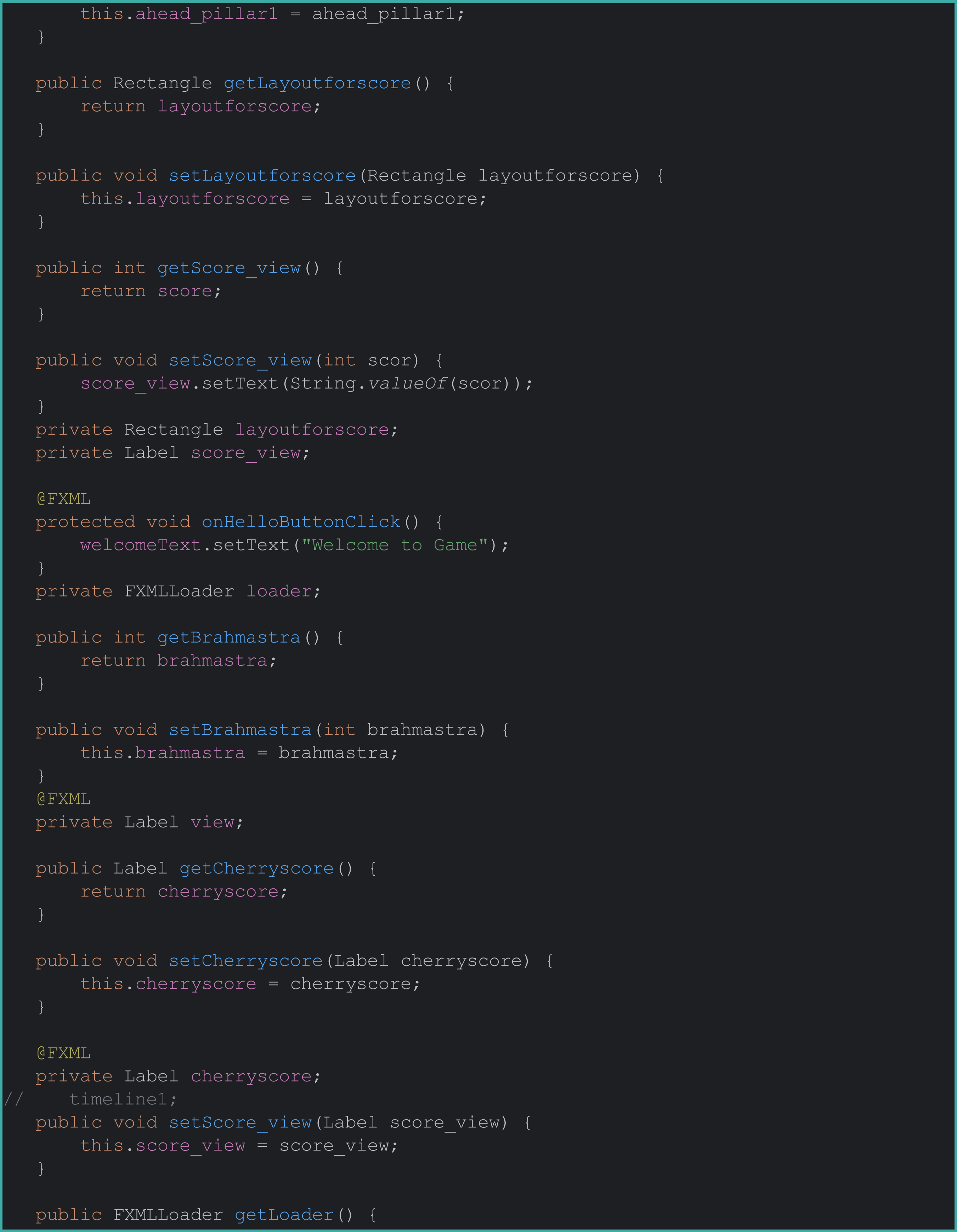


```java
        return width;
    }
    public void stick_grow_ka_sound(){
        MediaPlayer mediaPlayer = new MediaPlayer(this.getStick_fall_sound());
        mediaPlayer.play();
    }
    public int getStick_hero_height() {
        return stick_hero_height;
    }

    public int getSpeed() {
        return speed;
    }

    public String getStart_of_game_sound() {
        return start_of_game_sound;
    }

    public void setStart_of_game_sound(String start_of_game_sound) {
        this.start_of_game_sound = start_of_game_sound;
    }

    public void setLatch(CountDownLatch latch) {
        this.latch = latch;
    }

    public Scene getNewScene() {
        return newScene;
    }

    public void setNewScene(Scene newScene) {
        this.newScene = newScene;
    }

    public Stick getS() {
        return s;
    }

    public void setS(Stick s) {
        this.s = s;
    }

    public AtomicInteger getTest() {
        return test;
    }

    public void setTest(AtomicInteger test) {
        this.test = test;
    }

    public Stage getNewStage() {
        return newStage;
    }
```

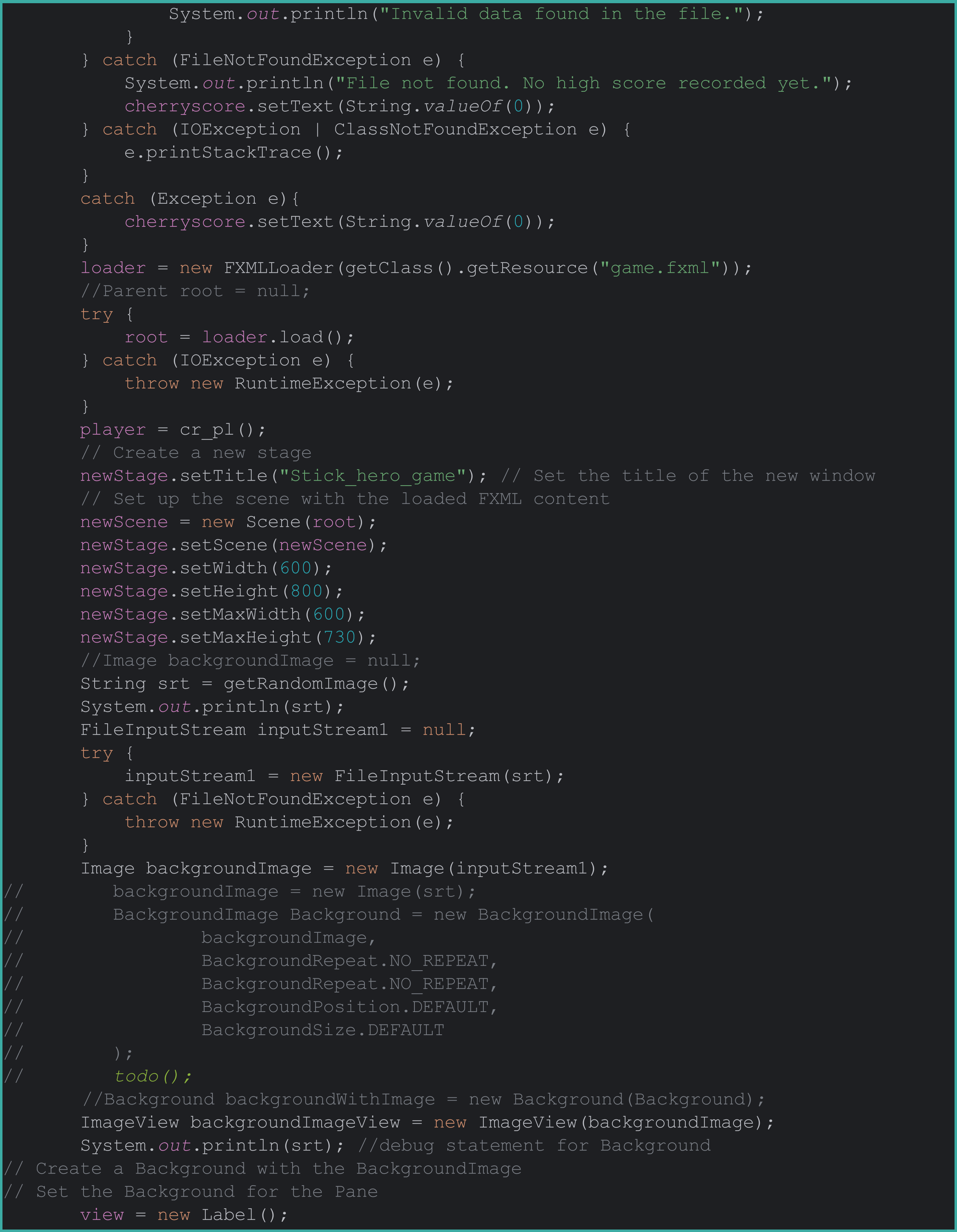
```
public void setNewStage(Stage newStage) {
    this.newStage = newStage;
}

public Timeline getTimeline() {
    return timeline;
}

public void setTimeline(Timeline timeline) {
    this.timeline = timeline;
}

public Parent getRoot() {
    return root;
}

public void setRoot(Parent root) {
    this.root = root;
}

public void setAdi_flag(int adi_flag) {
    this.adi_flag = adi_flag;
}

public void setSpeed(int speed) {
    this.speed = speed;
}

public int getScore() {
    return score;
}

public void setScore(int score) {
    this.score = score;
}

public int getPillar_length() {
    return pillar_length;
}

public void setPillar_length(int pillar_length) {
    this.pillar_length = pillar_length;
}

public int getGetPillar_width() {
    return getPillar_width;
}

public void setGetPillar_width(int getPillar_width) {
    this.getPillar_width = getPillar_width;
}
```

```java
   public int getCherries() {
       return cherries;
   }

   public void setCherries(int cherries) {
       this.cherries = cherries;
   }

   public boolean isKeyIsPressed() {
       return keyIsPressed;
   }

   public void setKeyIsPressed(boolean keyIsPressed) {
       this.keyIsPressed = keyIsPressed;
   }

   public boolean isClick_flag() {
       return click_flag;
   }

   public void setClick_flag(boolean click_flag) {
       this.click_flag = click_flag;
   }

   public Label getWelcomeText() {
       return welcomeText;
   }

   public void setWelcomeText(Label welcomeText) {
       this.welcomeText = welcomeText;
   }

   public void start_song(){
       MediaPlayer mediaPlayer = new MediaPlayer(this.getSound());
       mediaPlayer.play();
   }
   public StickHero(PlayerCreate player, int speed, int score, int pillar_length, int
getPillar_width, int cherries, boolean keyIsPressed, boolean click_flag, Label
welcomeText,boolean revived,boolean load) {
       this.player = null;
       this.speed = speed;
       this.score = score;
       this.pillar_length = pillar_length;
       this.getPillar_width = getPillar_width;
       this.cherries = cherries;
       this.keyIsPressed = keyIsPressed;
       this.click_flag = click_flag;
       this.welcomeText = welcomeText;
       this.load = load;
       this.revived=revived;
       //cherryadapt.add(new
StickHero(player,speed,score,pillar_length,getPillar_width,cherries,keyIsPressed,click
_flag,welcomeText));
```

```
        setTest(1);
    }

    private final int width = 800;
    private final int stick_hero_height = 50;
    private int speed = 0;
    private Stage newStage = new Stage();
    private int score = 0;
    private Pillar current_pillar;
    private Pillar ahead_pillar;

    public Pillar getCurrent_pillar() {
        return current_pillar;
    }

    public void setCurrent_pillar(Pillar current_pillar) {
        this.current_pillar = current_pillar;
    }

    public Pillar getAhead_pillar() {
        return ahead_pillar;
    }
    private Timeline timeline;
    public void setAhead_pillar(Pillar ahead_pillar) {
        this.ahead_pillar = ahead_pillar;
    }
    private int adi_flag=100; //yeh flag prevent for any code to get run more than 1
time if not needed
    private int  pillar_length = 100;
    private int getPillar_width = 30;
    private int cherries=0;
    private boolean keyIsPressed = false;
    private Parent root;
    private boolean click_flag = true; // will make it True once key is pressed
    private boolean revived;
    private Rectangle layoutforcherry;
    @FXML
    private Label welcomeText;
    private Pillar ahead_pillar1;

    public ImageView getCherry_1() {
        return cherry_1;
    }

    public void setCherry_1(ImageView cherry_1) {
        this.cherry_1 = cherry_1;
    }

    public Pillar getAhead_pillar1() {
        return ahead_pillar1;
    }

    public void setAhead_pillar1(Pillar ahead_pillar1) {
```



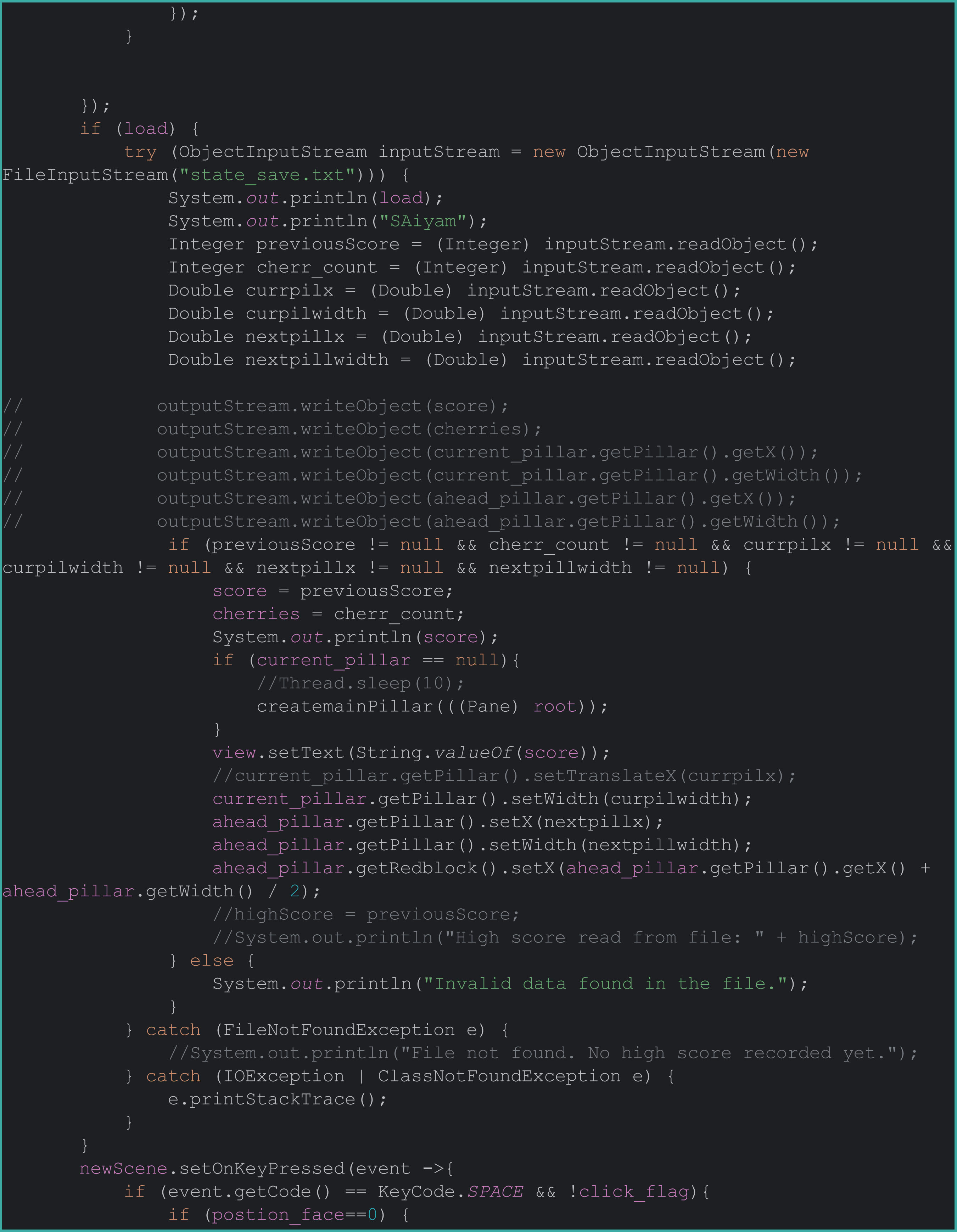
```java
        this.ahead_pillar1 = ahead_pillar1;
    }

    public Rectangle getLayoutforscore() {
        return layoutforscore;
    }

    public void setLayoutforscore(Rectangle layoutforscore) {
        this.layoutforscore = layoutforscore;
    }

    public int getScore_view() {
        return score;
    }

    public void setScore_view(int scor) {
        score_view.setText(String.valueOf(scor));
    }
    private Rectangle layoutforscore;
    private Label score_view;

    @FXML
    protected void onHelloButtonClick() {
        welcomeText.setText("Welcome to Game");
    }
    private FXMLLoader loader;

    public int getBrahmastra() {
        return brahmastra;
    }

    public void setBrahmastra(int brahmastra) {
        this.brahmastra = brahmastra;
    }
    @FXML
    private Label view;

    public Label getCherryscore() {
        return cherryscore;
    }

    public void setCherryscore(Label cherryscore) {
        this.cherryscore = cherryscore;
    }

    @FXML
    private Label cherryscore;
//     timeline1;
    public void setScore_view(Label score_view) {
        this.score_view = score_view;
    }

    public FXMLLoader getLoader() {
        return loader;
    }

    public void setLoader(FXMLLoader loader) {
        this.loader = loader;
    }

    public Label getView() {
        return view;
    }

    public void setView(Label view) {
        this.view = view;
    }
    public int getAdi_flag() {
        return adi_flag;
    }
    private int clickCount =0;
    //private Button button2 = new Button("Masti");
    public void back_create() throws IOException {


        timeline1.setCycleCount(-1);
        brahmastra = 0;
        score_view = new Label(String.valueOf(score));
        cherryscore = new Label(" ");
//          Button button = new Button("Masti");
//
//          // Set action for the button
//          button.setOnAction(asdfghjkl -> {
//              clickCount++;
//              if (clickCount % 2 == 1) {
//                  System.out.println("First action performed");
//                  ahead_pillar.ofpillar(ahead_pillar);
//
//                  // First action on odd click
//              } else {
//                  System.out.println("Second action performed");
//                  ahead_pillar.onpillar(ahead_pillar);
//                  // Second action on even click
//              }
//          });
//
//          button.setLayoutX(300);
//          button.setLayoutY(100);

        try (ObjectInputStream inputStream = new ObjectInputStream(new
FileInputStream("cherry_data.txt"))) {
            Integer chr = (Integer) inputStream.readObject();
            if (chr != null) {
                cherryscore.setText(String.valueOf(chr));
                cherries = chr;
            } else {
```



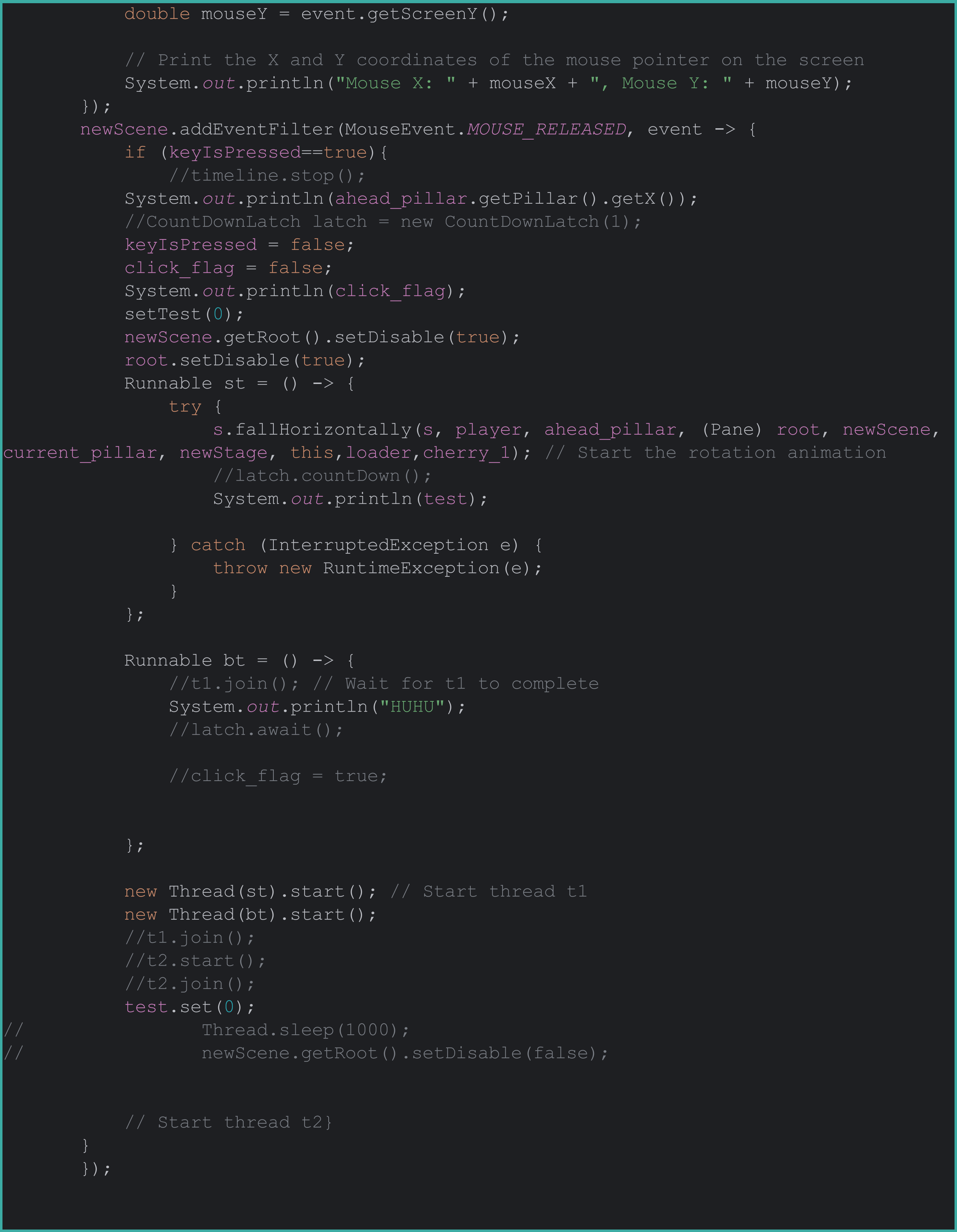
```java
                System.out.println("Invalid data found in the file.");
            }
        } catch (FileNotFoundException e) {
            System.out.println("File not found. No high score recorded yet.");
            cherryscore.setText(String.valueOf(0));
        } catch (IOException | ClassNotFoundException e) {
            e.printStackTrace();
        }
        catch (Exception e){
            cherryscore.setText(String.valueOf(0));
        }
        loader = new FXMLLoader(getClass().getResource("game.fxml"));
        //Parent root = null;
        try {
            root = loader.load();
        } catch (IOException e) {
            throw new RuntimeException(e);
        }
        player = cr_pl();
        // Create a new stage
        newStage.setTitle("Stick_hero_game"); // Set the title of the new window
        // Set up the scene with the loaded FXML content
        newScene = new Scene(root);
        newStage.setScene(newScene);
        newStage.setWidth(600);
        newStage.setHeight(800);
        newStage.setMaxWidth(600);
        newStage.setMaxHeight(730);
        //Image backgroundImage = null;
        String srt = getRandomImage();
        System.out.println(srt);
        FileInputStream inputStream1 = null;
        try {
            inputStream1 = new FileInputStream(srt);
        } catch (FileNotFoundException e) {
            throw new RuntimeException(e);
        }
        Image backgroundImage = new Image(inputStream1);
//         backgroundImage = new Image(srt);
//         BackgroundImage Background = new BackgroundImage(
//                 backgroundImage,
//                 BackgroundRepeat.NO_REPEAT,
//                 BackgroundRepeat.NO_REPEAT,
//                 BackgroundPosition.DEFAULT,
//                 BackgroundSize.DEFAULT
//         );
//         todo();
        //Background backgroundWithImage = new Background(Background);
        ImageView backgroundImageView = new ImageView(backgroundImage);
        System.out.println(srt); //debug statement for Background
// Create a Background with the BackgroundImage
// Set the Background for the Pane
        view = new Label();
        backgroundImageView.setFitWidth(600);
        layoutforscore = new Rectangle(216, 76, 100, 100);
        layoutforscore.setFill(Color.SKYBLUE); // Set fill color
        layoutforscore.setStroke(Color.BLACK); // Set stroke color
        layoutforcherry = new Rectangle(0, 0, 100, 100);
        layoutforcherry.setFill(Color.SKYBLUE); // Set fill color
        layoutforcherry.setStroke(Color.BLACK);
        player.getNode().setTranslateX(0);

        backgroundImageView.setFitHeight(700);
        score_view.setStyle("-fx-font-size: 40px; -fx-font-weight: bold;");

        ((Pane) root).getChildren().add(backgroundImageView);
        ((Pane) root).getChildren().add(layoutforscore);
        ((Pane) root).getChildren().add(score_view);
        String imageUrl =
("src/Main/resources/com/example/stick_hero_final_project/Images/cherry.png");
        FileInputStream inputStream = new FileInputStream(imageUrl);
        Image image = new Image(inputStream);
        ImageView imageView = new ImageView();
        // Setting the image to the ImageView
//        ((Pane) root).getChildren().add(button2);
//        button2.setLayoutX(200);
//        button2.setLayoutY(300);
        imageView.setImage(image);
        imageView.setX(0);
        imageView.setY(0);
        imageView.setFitHeight(50);
        imageView.setFitWidth(50);
//        ((Pane) root).getChildren().add(view); //view ko niche lagaya hai
        view.setLayoutX(100);
        cherryscore.setLayoutX(60);
        cherryscore.setLayoutY(15);
        ((Pane) root).getChildren().add(layoutforcherry);
        ((Pane) root).getChildren().add(imageView);
        ((Pane) root).getChildren().add(cherryscore);
        cherryscore.setStyle("-fx-font-size: 40px; -fx-font-weight: bold;");
        view.setLayoutY(300);
        view.setText("hu");
//        view.setVisible(false);
        score_view.setLayoutX(258); // Position from the right
        score_view.setLayoutY(100);
        if (revived==true) {
            try (ObjectInputStream inputStream78 = new ObjectInputStream(new
FileInputStream("state_save.txt"))) {
                Integer chr = (Integer) inputStream78.readObject();
                if (chr != null) {
                    view.setText(String.valueOf(chr));
                    score = chr;
                } else {
                    System.out.println("Invalid data found in the file.");
                }
            } catch (FileNotFoundException e) {
                System.out.println("File not found. No high score recorded yet.");
                view.setText(String.valueOf(0));
            } catch (IOException | ClassNotFoundException e) {
                e.printStackTrace();
            } catch (Exception e) {
                view.setText(String.valueOf(0));
            }
        }
//        initialize1((Pane) root);
        // Show the new stage
        ((Pane) root).getChildren().add(player.getNode());
        player.getNode().setX(0);
        player.getNode().setY(500);
        start_song();
        newStage.show();
//        while (newScene==null){}
        if (current_pillar!=null){
        current_pillar.onpillar(current_pillar);}
        init();
    }
    public void rem(){
        if (ahead_pillar!=null && ((Pane)
root).getChildren().contains(ahead_pillar.getRedblock())){
            ((Pane) root).getChildren().remove(ahead_pillar.getRedblock());
        }
    }
    AtomicInteger khtm = new AtomicInteger();
    Timeline timeline1 = timeline1 = new Timeline( //thoda pdhna padega timeline ke
baare me

            new KeyFrame(Duration.millis(80), e -> {
                if (keyIsPressed && click_flag) {
                    s.extend(10);
                    stick_grow_ka_sound();
//                            latch.countDown();
                    //ext.updateAndGet(v -> new Double((double) (v * 0.9)));
//                            System.out.println("HI"); Debug statement
                }
                else {
                    khtm.set(1);
                    System.out.println("kopkopkop[]");
                    tstop();
                }

            })
    );

    private void tstop() {
        timeline1.stop();

        System.gc();
        System.out.println("yeh");
    }
```

```java
    public void remcherry(){
        if (((Pane) root).getChildren().contains(cherry_1)){
            ((Pane) root).getChildren().remove(cherry_1);
            cherry_1 = null;
        }
    }
    public int getPostion_face() {
        return postion_face;
    }

    public void setPostion_face(int postion_face) {
        this.postion_face = postion_face;
    }

    public void init() throws IOException {
        System.gc();
        newStage.setOnCloseRequest(event -> {
            // Handle the close request here
            System.out.println("Close button pressed");
            // You can perform any cleanup or other actions here before exiting
            Platform.exit(); // Exit the application
        });
//        player.getNode().setX(0);
//        player.getNode().setY(500);
//        if (((Pane) root).getChildren().contains(cherry_1)){
//            ((Pane) root).getChildren().remove(cherry_1);
//        }
//        player.getNode().setX(0);
//        player.getNode().setY(500);
        cherry_1 =null;
//        view.setVisible(false);
        setScore_view(score);
//        ((Pane) root).getChildren().add(view);
        view.setLayoutX(300);
        view.setLayoutY(300);
        setClick_flag(true);
        keyIsPressed = true;
//        if (ahead_pillar!=null && ((Pane)
root).getChildren().contains(ahead_pillar.getRedblock())){
//            ((Pane) root).getChildren().remove(ahead_pillar.getRedblock());
//        }
//        ahead_pillar = getAhead_pillar1();
        if (brahmastra==0){
            current_pillar = getAhead_pillar();
        createRandomPillar((Pane) root);}
//        createRandomPillar2((Pane) root);

view.setLayoutX(ahead_pillar.getPillar().getX()+ahead_pillar.getPillar().getWidth()/2)
;
        view.setLayoutY(480);
        view.setStyle("-fx-font-size: 20;");
        if (((Pane)root).getChildren().contains(view)){
```

```java
       ((Pane)root).getChildren().remove(view);
       view.setOpacity(1.0);
       view.setTextFill(Color.BLACK);
       view.setVisible(false);
       }
//        view = new Label("+1");
       view.setText("+1");
       view.setVisible(false);

       ((Pane)root).getChildren().add(view);
       if (current_pillar!= null){
       current_pillar.onpillar(current_pillar);}
//        view.setVisible(false);
// Create a BackgroundImage

       newScene.addEventFilter(MouseEvent.MOUSE_PRESSED, event -> {
           System.out.println(click_flag);
           try {
               latch.await();
           } catch (InterruptedException e) {
               throw new RuntimeException(e);
           }
           if (click_flag) {
               keyIsPressed = true;
               double x = event.getSceneX();
               double y = event.getSceneY();

               s = new Stick(0, 0, 5, 0);
               if (!((Pane) root).getChildren().contains(s.getStick())) {
                   ((Pane) root).getChildren().add(s.getStick());
               }

               System.out.println("Start and end " + player.getNode().getX() + "     "
+ player.getNode().getY());
               s.getStick().setX(player.getNode().getTranslateX() + (double) 40);

               s.getStick().setY(player.getNode().getY() + (double) 30);

               AtomicReference<Double> ext = new AtomicReference<>((double) 5);
               speed++;
//                  if (timeline1.getStatus() != Animation.Status.STOPPED) {
//                      timeline1.stop(); // Stop the previous timeline if it's not
already stopped
//                      //timeline1.getKeyFrames().clear(); // Clear existing keyframes
//                  }

//                  Timeline timeline89 = new Timeline(
//                          new KeyFrame(Duration.millis(10), eop ->{
//                      if (khtm.get()==1){
//                          System.out.println("dil jeet liya");
//                          khtm.set(0);
//                          timeline1.stop();
//
//                    }
//                    else {
//                        System.out.println("bhao");
//                    }
//            })
//                );
               //AnimationTimer checkKhtm = new AnimationTimer() {
//                    @Override
//                    public void handle(long now) {
//                        if (khtm.get() == 1) {
//                            khtm.set(0);
//                            timeline1.stop();
//                            System.out.println("chandan boodha");
//                            stop(); // Stop the AnimationTimer when condition met
//                        }
//                    }
//                };
//                checkKhtm.start();
//                timeline1.setCycleCount(10000);
               //
               Runnable indefiniteExecution = () -> {
                   while (keyIsPressed && click_flag) {
                       System.out.println("");
                       timeline1.play();
                       try {
                           Thread.currentThread().sleep(10);
                       } catch (InterruptedException e) {
                           throw new RuntimeException(e);
                       }

                   }
               };

               Thread indefiniteThread = new Thread(indefiniteExecution);
               indefiniteThread.start();
               //timeline89.play();


               timeline1.setOnFinished(actionEvent -> {
                   //timeline1.stop();
                   //timeline89.stop();
                   //timeline89.setCycleCount(1);
                   //timeline1.setCycleCount(0);
               });
               newScene.addEventFilter(MouseEvent.MOUSE_RELEASED, evenyt -> {
                   keyIsPressed = false; // Reset keyIsPressed on mouse release
                   timeline1.stop();

                   //indefiniteThread.interrupt();
                   //System.gc();
                   //timeline89.stop();
                   //timeline1.setCycleCount(0);
                   //timeline89.setCycleCount(0);
```

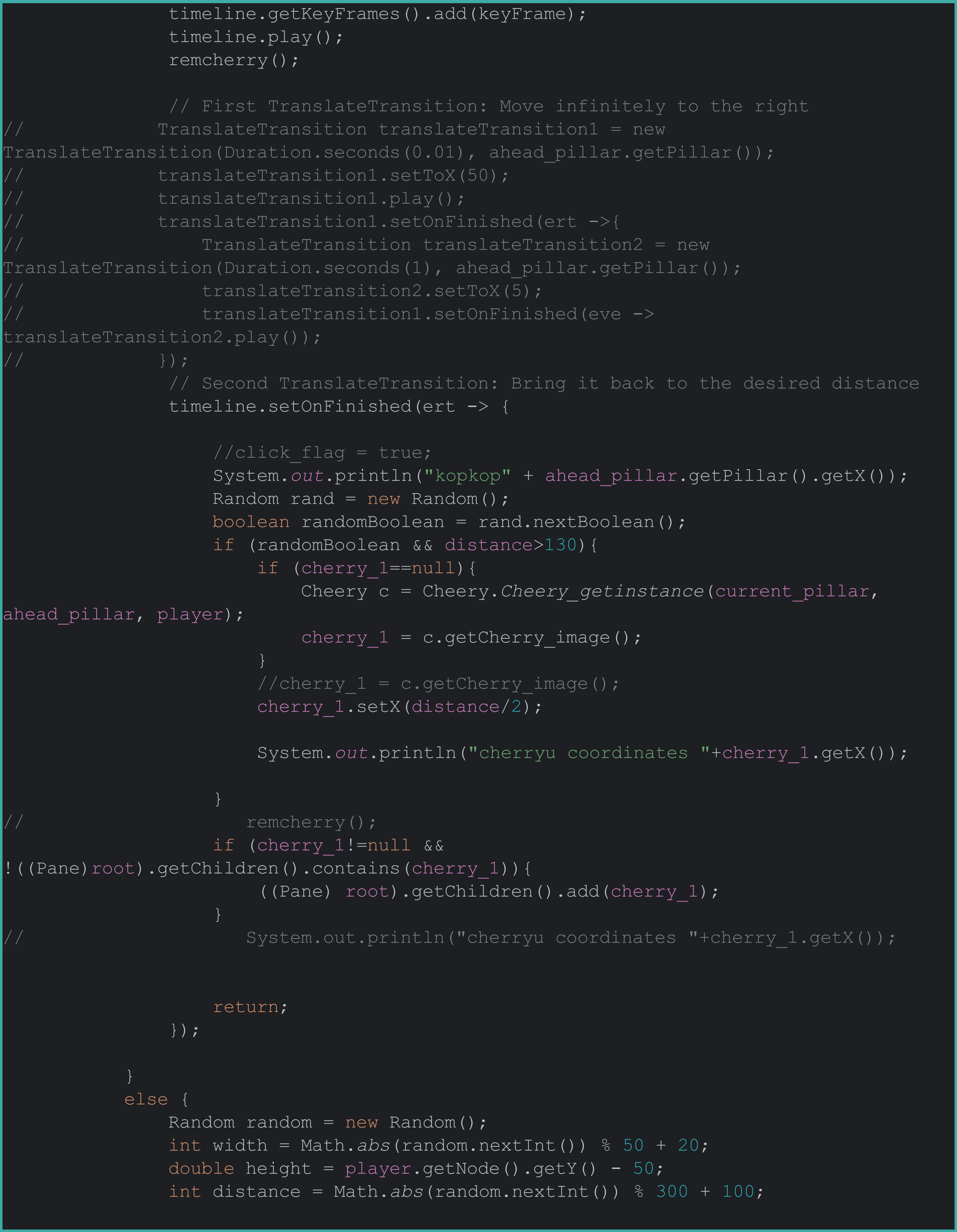


```
                });
            }


        });
        if (load) {
            try (ObjectInputStream inputStream = new ObjectInputStream(new
FileInputStream("state_save.txt"))) {
                System.out.println(load);
                System.out.println("SAiyam");
                Integer previousScore = (Integer) inputStream.readObject();
                Integer cherr_count = (Integer) inputStream.readObject();
                Double currpilx = (Double) inputStream.readObject();
                Double curpilwidth = (Double) inputStream.readObject();
                Double nextpillx = (Double) inputStream.readObject();
                Double nextpillwidth = (Double) inputStream.readObject();

//            outputStream.writeObject(score);
//            outputStream.writeObject(cherries);
//            outputStream.writeObject(current_pillar.getPillar().getX());
//            outputStream.writeObject(current_pillar.getPillar().getWidth());
//            outputStream.writeObject(ahead_pillar.getPillar().getX());
//            outputStream.writeObject(ahead_pillar.getPillar().getWidth());
                if (previousScore != null && cherr_count != null && currpilx != null &&
curpilwidth != null && nextpillx != null && nextpillwidth != null) {
                    score = previousScore;
                    cherries = cherr_count;
                    System.out.println(score);
                    if (current_pillar == null){
                        //Thread.sleep(10);
                        createmainPillar(((Pane) root));
                    }
                    view.setText(String.valueOf(score));
                    //current_pillar.getPillar().setTranslateX(currpilx);
                    current_pillar.getPillar().setWidth(curpilwidth);
                    ahead_pillar.getPillar().setX(nextpillx);
                    ahead_pillar.getPillar().setWidth(nextpillwidth);
                    ahead_pillar.getRedblock().setX(ahead_pillar.getPillar().getX() +
ahead_pillar.getWidth() / 2);
                    //highScore = previousScore;
                    //System.out.println("High score read from file: " + highScore);
                } else {
                    System.out.println("Invalid data found in the file.");
                }
            } catch (FileNotFoundException e) {
                //System.out.println("File not found. No high score recorded yet.");
            } catch (IOException | ClassNotFoundException e) {
                e.printStackTrace();
            }
        }
        newScene.setOnKeyPressed(event ->{
            if (event.getCode() == KeyCode.SPACE && !click_flag){
                if (postion_face==0) {
                    player.getNode().setScaleY(-1);
                    System.out.println("GUIII");
                    // Simulating very fast movement and scaling using Timeline
                    player.getNode().setLayoutY(player.getNode().getLayoutY() + 20);
                    System.out.println(player.getNode().getY());
                    // Scale the player vertically

                    postion_face = 1;

                }
                else{
                    player.getNode().setLayoutY(player.getNode().getLayoutY()-20);
                    System.out.println("BY");
                    // Scale the player vertically
                    player.getNode().setScaleY(1);// Start the animation
                    postion_face = 0;

                }
            }
            if (event.getCode() == KeyCode.H){
                clickCount++;
            if (clickCount % 2 == 1) {
                System.out.println("First action performed");
                ahead_pillar.ofpillar(ahead_pillar);

                // First action on odd click
            } else {
                System.out.println("Second action performed");
                ahead_pillar.onpillar(ahead_pillar);
                // Second action on even click
            }
            }
            if (event.getCode() == KeyCode.S){
                try (ObjectOutputStream outputStream = new ObjectOutputStream(new
FileOutputStream("state_save.txt"))) {
                    System.out.println(load);
                    System.out.println("SAiyam");
                    outputStream.writeObject(score);
                    outputStream.writeObject(cherries);

outputStream.writeObject(current_pillar.getPillar().getBoundsInParent().getMaxX());
                    outputStream.writeObject(current_pillar.getPillar().getWidth());

outputStream.writeObject(ahead_pillar.getPillar().getBoundsInParent().getMaxX());
                    outputStream.writeObject(ahead_pillar.getPillar().getWidth());
                }
                catch (Exception e){
                    e.printStackTrace();
                }
            }
        });
        newScene.addEventFilter(MouseEvent.MOUSE_MOVED,event -> {
            double mouseX = event.getScreenX();
```



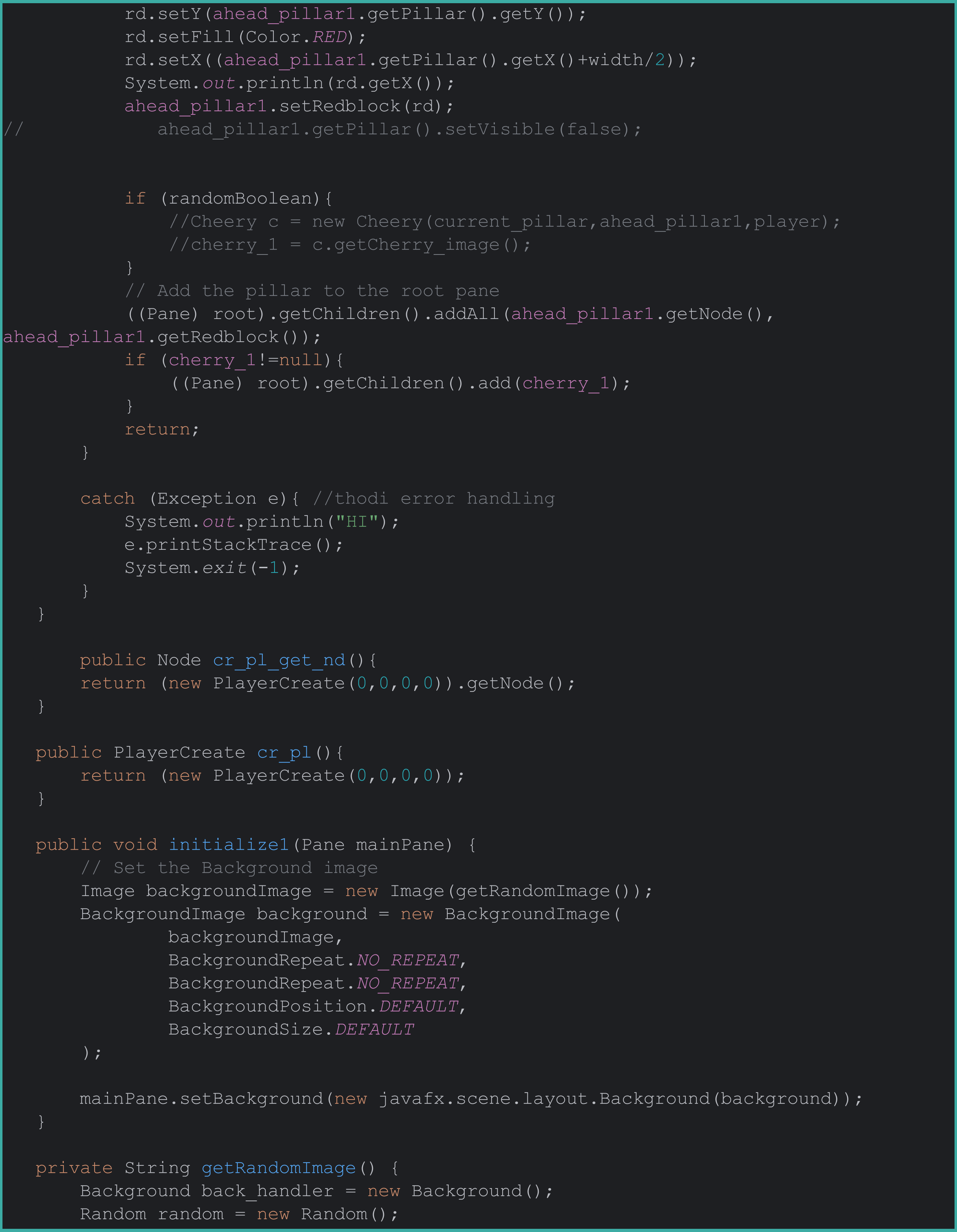
```
           double mouseY = event.getScreenY();

           // Print the X and Y coordinates of the mouse pointer on the screen
           System.out.println("Mouse X: " + mouseX + ", Mouse Y: " + mouseY);
       });
       newScene.addEventFilter(MouseEvent.MOUSE_RELEASED, event -> {
           if (keyIsPressed==true){
               //timeline.stop();
           System.out.println(ahead_pillar.getPillar().getX());
           //CountDownLatch latch = new CountDownLatch(1);
           keyIsPressed = false;
           click_flag = false;
           System.out.println(click_flag);
           setTest(0);
           newScene.getRoot().setDisable(true);
           root.setDisable(true);
           Runnable st = () -> {
               try {
                   s.fallHorizontally(s, player, ahead_pillar, (Pane) root, newScene,
current_pillar, newStage, this,loader,cherry_1); // Start the rotation animation
                   //latch.countDown();
                   System.out.println(test);

               } catch (InterruptedException e) {
                   throw new RuntimeException(e);
               }
           };

           Runnable bt = () -> {
               //t1.join(); // Wait for t1 to complete
               System.out.println("HUHU");
               //latch.await();

               //click_flag = true;


           };

           new Thread(st).start(); // Start thread t1
           new Thread(bt).start();
           //t1.join();
           //t2.start();
           //t2.join();
           test.set(0);
//                Thread.sleep(1000);
//                newScene.getRoot().setDisable(false);


           // Start thread t2}
       }
       });
```

```
        //Yha pr pillar ko dalna hai

        if (adi_flag==100){
            if (current_pillar==null) {
                createmainPillar((Pane) root);
            }
//              createRandomPillar((Pane) root);
            adi_flag=101; // setting this so also not providing any setter function
such that by any chance it cant re do and edit this intentionally not providing setter
        }
    }
    public void createmainPillar(Pane root){
        //will come on start Ninja jispr khara rhega
        current_pillar = new Pillar(player.getNode().getX(),530,40,500);
        root.getChildren().add(current_pillar.getNode());
        return;
    }
    public void createRandomPillar(Pane root) {
        try {
            if (brahmastra == 1) {
                click_flag = false;
                Random random = new Random();
                int width = Math.abs(random.nextInt()) % 50 + 40;
                double height = player.getNode().getY() - 50;
                int distance = Math.abs(random.nextInt()) % 300 + 100;

                ahead_pillar = new Pillar(player.getNode().getX() + 200, 530, width,
height);
                Rectangle rd = new Rectangle(5, 2, Color.RED);
                rd.setY(ahead_pillar.getPillar().getY());
                rd.setFill(Color.RED);
                System.out.println("LOPLOP" + distance);
                ahead_pillar.getPillar().setX(500);
                rd.setX(500 + width / 2);
                ahead_pillar.setRedblock(rd);

                // Start the pillar off-screen to the right
//                 rem();
                if (((Pane)root).getChildren().contains(ahead_pillar.getPillar())){
                    root.getChildren().remove(ahead_pillar.getPillar());
                }
                root.getChildren().addAll(ahead_pillar.getNode(),
ahead_pillar.getRedblock());
                Timeline timeline = new Timeline();

                // Create KeyValue to gradually change the x position of the rectangle
                KeyValue keyValue = new KeyValue(ahead_pillar.getPillar().xProperty(),
distance);
                KeyValue keyValue2 = new KeyValue(rd.xProperty(), distance + width /
2);
                KeyFrame keyFrame = new KeyFrame(Duration.seconds(0.2), keyValue,
keyValue2);
```



```
                timeline.getKeyFrames().add(keyFrame);
                timeline.play();
                remcherry();

                // First TranslateTransition: Move infinitely to the right
//              TranslateTransition translateTransition1 = new
TranslateTransition(Duration.seconds(0.01), ahead_pillar.getPillar());
//              translateTransition1.setToX(50);
//              translateTransition1.play();
//              translateTransition1.setOnFinished(ert ->{
//                  TranslateTransition translateTransition2 = new
TranslateTransition(Duration.seconds(1), ahead_pillar.getPillar());
//                  translateTransition2.setToX(5);
//                  translateTransition1.setOnFinished(eve ->
translateTransition2.play());
//              });
                // Second TranslateTransition: Bring it back to the desired distance
                timeline.setOnFinished(ert -> {

                    //click_flag = true;
                    System.out.println("kopkop" + ahead_pillar.getPillar().getX());
                    Random rand = new Random();
                    boolean randomBoolean = rand.nextBoolean();
                    if (randomBoolean && distance>130){
                        if (cherry_1==null){
                            Cheery c = Cheery.Cheery_getinstance(current_pillar,
ahead_pillar, player);
                            cherry_1 = c.getCherry_image();
                        }
                        //cherry_1 = c.getCherry_image();
                        cherry_1.setX(distance/2);

                        System.out.println("cherryu coordinates "+cherry_1.getX());

                    }
//                      remcherry();
                    if (cherry_1!=null &&
!((Pane)root).getChildren().contains(cherry_1)){
                        ((Pane) root).getChildren().add(cherry_1);
                    }
//                      System.out.println("cherryu coordinates "+cherry_1.getX());


                    return;
                });

            }
            else {
                Random random = new Random();
                int width = Math.abs(random.nextInt()) % 50 + 20;
                double height = player.getNode().getY() - 50;
                int distance = Math.abs(random.nextInt()) % 300 + 100;
```

```
               ahead_pillar = new Pillar(player.getNode().getX() + 200, 530, width,
height);
               Rectangle rd = new Rectangle(5, 2, Color.RED);
               rd.setY(ahead_pillar.getPillar().getY());
               rd.setFill(Color.RED);
               System.out.println("LOPLOP" + distance);
               ahead_pillar.getPillar().setX(distance);
               rd.setX(distance + width / 2);
               ahead_pillar.setRedblock(rd);

               // Start the pillar off-screen to the right


               root.getChildren().addAll(ahead_pillar.getNode(),
ahead_pillar.getRedblock());
               brahmastra++;
               Random rand = new Random();
               boolean randomBoolean = rand.nextBoolean();
               if (randomBoolean && distance>130){
                   Cheery c =
Cheery.Cheery_getinstance(current_pillar,ahead_pillar,player);
                   cherry_1 = c.getCherry_image();
               }
               if (cherry_1!=null){
                   ((Pane) root).getChildren().add(cherry_1);
               }
           }
       } catch(Exception e){
               System.out.println("HI");
               e.printStackTrace();
               System.exit(-1);
           }

   }



   public void createRandomPillar2(Pane root) {
       try {
//             click_flag=false;
           Random random = new Random();
           int width = Math.abs(random.nextInt()) % 50 + 20; // width is random
setting as likha tha assignment me
           double height =player.getNode().getY()-50; // yha pr basically uski hieght
acc to ninja niklegi wese to yeh ek constant as uska y axis will not change never
           int distance = Math.abs(random.nextInt()) % 100 + 100;
           ahead_pillar1 = new Pillar(player.getNode().getX()+200+100, 530, width,
height);
           Random rand = new Random();
           boolean randomBoolean = rand.nextBoolean();
           // Set pillar position at a random distance
           ahead_pillar1.getPillar().setX(distance);
           Rectangle rd = new Rectangle(5, 2, Color.RED);
```



```java
            rd.setY(ahead_pillar1.getPillar().getY());
            rd.setFill(Color.RED);
            rd.setX((ahead_pillar1.getPillar().getX()+width/2));
            System.out.println(rd.getX());
            ahead_pillar1.setRedblock(rd);
//             ahead_pillar1.getPillar().setVisible(false);


            if (randomBoolean){
                //Cheery c = new Cheery(current_pillar,ahead_pillar1,player);
                //cherry_1 = c.getCherry_image();
            }
            // Add the pillar to the root pane
            ((Pane) root).getChildren().addAll(ahead_pillar1.getNode(),
ahead_pillar1.getRedblock());
            if (cherry_1!=null){
                ((Pane) root).getChildren().add(cherry_1);
            }
            return;
        }

        catch (Exception e){ //thodi error handling
            System.out.println("HI");
            e.printStackTrace();
            System.exit(-1);
        }
    }

        public Node cr_pl_get_nd(){
        return (new PlayerCreate(0,0,0,0)).getNode();
    }

    public PlayerCreate cr_pl(){
        return (new PlayerCreate(0,0,0,0));
    }

    public void initialize1(Pane mainPane) {
        // Set the Background image
        Image backgroundImage = new Image(getRandomImage());
        BackgroundImage background = new BackgroundImage(
                backgroundImage,
                BackgroundRepeat.NO_REPEAT,
                BackgroundRepeat.NO_REPEAT,
                BackgroundPosition.DEFAULT,
                BackgroundSize.DEFAULT
        );

        mainPane.setBackground(new javafx.scene.layout.Background(background));
    }

    private String getRandomImage() {
        Background back_handler = new Background();
        Random random = new Random();
        int index = random.nextInt(back_handler.getBackgroundImages().size());
        return back_handler.getBackgroundImages().get(index);
    }

    private void onMouseDragged(MouseEvent event) {
        double xCoordinate = event.getX();
        double yCoordinate = event.getY();
        System.out.println("Mouse dragged at coordinates: (" + xCoordinate + ", " +
yCoordinate + ")");

        // You can perform any action or logic based on these coordinates here
    }

    @Override
    public void viewscore() {
        System.out.println(score);;
    }

    @Override
    public void setscore() {
        score++;
    }

    @Override
    public void increase_score() {
        score++;
    }

    @Override
    public void set_cherries() {

    }

    @Override
    public void inc_cherries() throws InterruptedException {
        if (cherry_1!=null) {
//              Thread.sleep(10);
            cherryscore.setText(String.valueOf(cherries + 1));
            cherries++;
        }
    }

    @Override
    public void view_cherries() {
        System.out.println(cherries);
    }

    @Override
    public void revive_cherries() {
        cherries-=5;
    }
    @Override
    public void display() {
        System.out.println("hi");
    }

    @Override
    public void set_score() {

    }

    @Override
    public void increment() {

    }

    @Override
    public void perfect_increment() {

    }
}
```

this is the main code to be worked upon that works as a heart here everything works created in smaller chunks.

The code here specifies it.

We have included 2 design patterns 1st Singleton and second Decorator.

Here the Singleton is used with cherry as cherry here needs to be instantiated once and after that we can access it by just resetting the X and Y coordinates.

```
package com.example.stick_hero_final_project;

import javafx.scene.image.Image;
import javafx.scene.image.ImageView;

import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.util.Random;

public class Cheery {
   public Double min(Double a,Double b){
       if (a<b){
           return a;
       }
       else {
           return b;
       }


   }
   private static ImageView cherry_image;
   private static Cheery cheery;
   public ImageView getCherry_image() {
       return cherry_image;
   }
   public static Cheery Cheery_getinstance(Pillar p1, Pillar p2, PlayerCreate player){
       if (cheery == null){
           cheery = new Cheery(p1,p2,player);
           return cheery;
       }
       else{
           double minX = 0;
           if(p1!=null){
               minX = p1.getPillar().getX()+p1.getPillar().getWidth();
           }
           else {
               minX = 60;
           }
           double maxX = p2.getPillar().getX();
           System.out.println("Max min "+minX+" "+p2.getPillar().getTranslateX());
           String cherr =
"src/Main/resources/com/example/stick_hero_final_project/Images/cherry.png";
           FileInputStream inputStream = null;
           try {
               inputStream = new FileInputStream(cherr);
           } catch (FileNotFoundException e) {
               throw new RuntimeException(e);
           }
           if (cherry_image!=null){
               Image cherry = new Image(inputStream) ;
               cherry_image = new ImageView(cherry);
           };
           cherry_image.setFitWidth(30);
           cherry_image.setFitHeight(30);
           Random random = new Random();
           double randomX = (minX + maxX)/2;
           cherry_image.setX(randomX);
           System.out.println("vhhv"+cherry_image.getX());
           //boolean dirn = false;
           cherry_image.setY(500 + player.getNode().getFitHeight() + 5);
           return cheery;
       }
   }
   private Cheery(Pillar p1, Pillar p2, PlayerCreate player){
       double minX = 0;
       if(p1!=null){
           minX = p1.getPillar().getX()+p1.getPillar().getWidth();
       }
       else {
           minX = 60;
       }
       double maxX = p2.getPillar().getX();
       System.out.println("Max min "+minX+" "+p2.getPillar().getTranslateX());
       String cherr =
"src/Main/resources/com/example/stick_hero_final_project/Images/cherry.png";
       FileInputStream inputStream = null;
       try {
           inputStream = new FileInputStream(cherr);
       } catch (FileNotFoundException e) {
           throw new RuntimeException(e);
       }
       Image cherry = new Image(inputStream) ;
       cherry_image = new ImageView(cherry);
       cherry_image.setFitWidth(30);
       cherry_image.setFitHeight(30);
       Random random = new Random();
       double randomX = (minX + maxX)/2;
       cherry_image.setX(randomX);
       System.out.println("vhhv"+cherry_image.getX());
       //boolean dirn = false;
       cherry_image.setY(500 + player.getNode().getFitHeight() + 5);
   };
}
```

(cherry code)

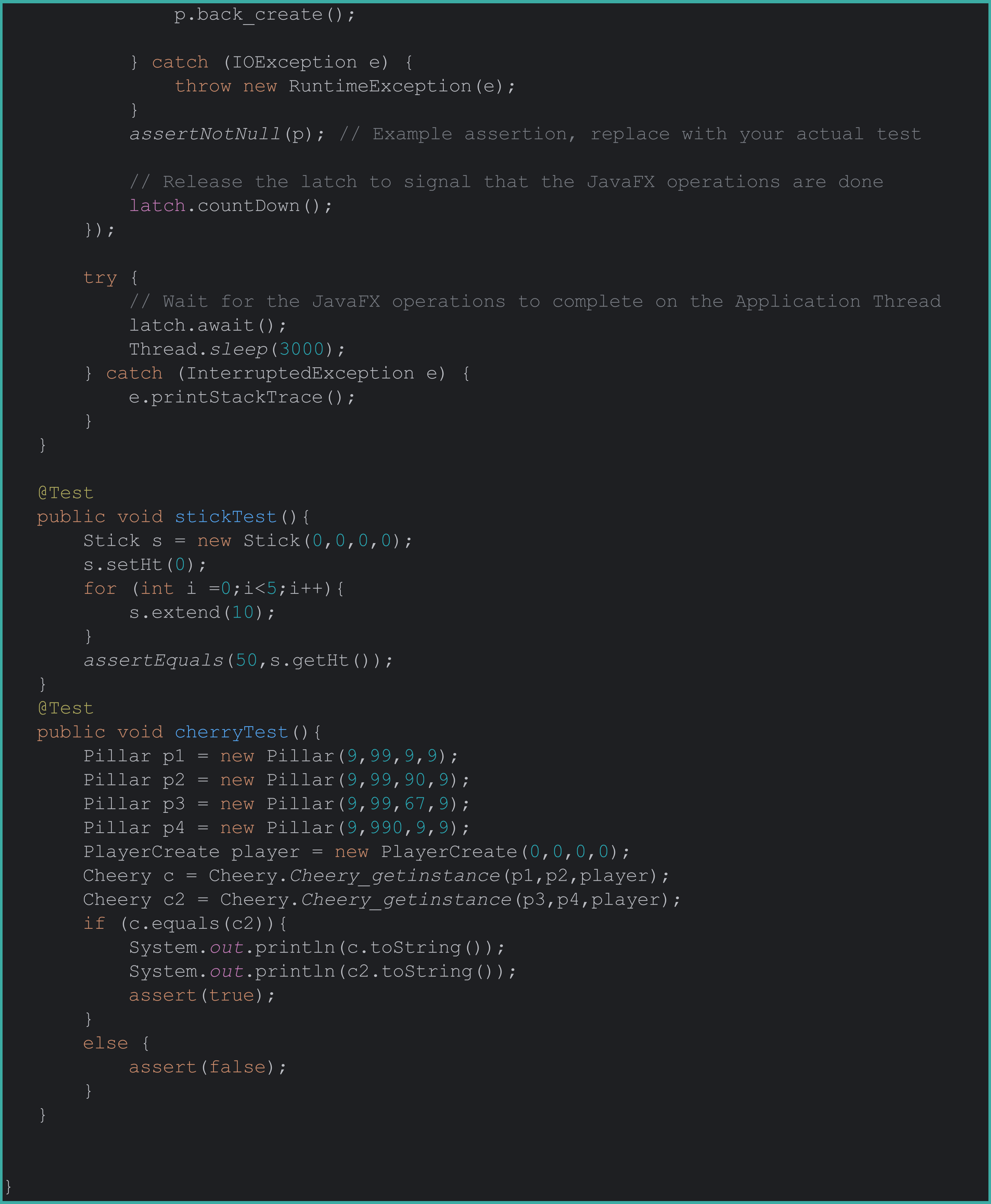
```
    @Override
    public void onpillar(Pillar p) {
        p.getPillar().setVisible(true);
        p.getRedblock().setVisible(true);
    }

    @Override
    public void ofpillar(Pillar p) {
        p.getPillar().setVisible(false);
        p.getRedblock().setVisible(false);
    }
}
```

Decorator is managed with Pillar for UI to additional feature as to disappear the pillar as the new one.

Now for Junit testings

We have made 3 tests for it as follows

```
package com.example.stick_hero_final_project;

import org.junit.jupiter.api.*;
import javafx.application.Platform;
import javafx.scene.control.Label;

import java.io.IOException;
import java.nio.FloatBuffer;
import java.util.concurrent.CountDownLatch;

import static org.junit.jupiter.api.Assertions.*;

public class StickHeroTest {

   @BeforeAll
   public static void initJFX() {
       // Initialize JavaFX Toolkit
       Platform.startup(() -> {
           // Any initialization code if needed
       });
       Platform.setImplicitExit(false);
   }

   @AfterAll
   public static void cleanUp() {
       // Clean up JavaFX Toolkit
       Platform.exit();
   }
   @Test
   public void StickheroNullity() {
       CountDownLatch latch = new CountDownLatch(1);

       Platform.runLater(() -> {
           StickHero p = new StickHero(new PlayerCreate(0, 0, 0, 0), 10, 0, 100, 30,
0, false, true, new Label("HI"),false, false);
           try {
```



```
                p.back_create();

            } catch (IOException e) {
                throw new RuntimeException(e);
            }
            assertNotNull(p); // Example assertion, replace with your actual test

            // Release the latch to signal that the JavaFX operations are done
            latch.countDown();
        });

        try {
            // Wait for the JavaFX operations to complete on the Application Thread
            latch.await();
            Thread.sleep(3000);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }

    @Test
    public void stickTest(){
        Stick s = new Stick(0,0,0,0);
        s.setHt(0);
        for (int i =0;i<5;i++){
            s.extend(10);
        }
        assertEquals(50,s.getHt());
    }
    @Test
    public void cherryTest(){
        Pillar p1 = new Pillar(9,99,9,9);
        Pillar p2 = new Pillar(9,99,90,9);
        Pillar p3 = new Pillar(9,99,67,9);
        Pillar p4 = new Pillar(9,990,9,9);
        PlayerCreate player = new PlayerCreate(0,0,0,0);
        Cheery c = Cheery.Cheery_getinstance(p1,p2,player);
        Cheery c2 = Cheery.Cheery_getinstance(p3,p4,player);
        if (c.equals(c2)){
            System.out.println(c.toString());
            System.out.println(c2.toString());
            assert(true);
        }
        else {
            assert(false);
        }
    }


}
```

Cherry singleton test
Stickhero not null test

stickextend test

Credits help

1) Geeks for Geeks
2) stack exchange
3) Professor Koteswar Rao Jerripothula
4) Harshita Goyal

Thank You