A Mini Project Report

On

BRICK BREAKER GAME

Submitted in partial fulfillment of requirements for the Course
CSE18R272 - JAVA PROGRAMMING

Bachelor's of Technology

In

Computer Science and Engineering

Submitted By

SHARIK ANWAR Z

9918004110

RAJA DHANANJEYAN V

9918004098

Under the guidance of

Dr. R. RAMALAKSHMI

(Associate Professor)

Department of Computer Science and Engineering

Kalasalingam Academy of Research and Education

Anand Nagar, Krishnankoil-626126

APRIL 2020

ABSTRACT

Brick Breaker (platformer) is a Breakout game in which the player must smash a wall of bricks by deflecting a bouncing ball with a paddle. The paddle may move horizontally and is controlled with the arrow keys of the keyboard.

DECLARATION

I hereby declare that the work presented in this report entitled "**BRICK BREAKER GAME**", in partial fulfilment of the requirements for the course CSE18R272- Java Programming and submitted in **Department of Computer Science and Engineering, Kalasalingam Academy of Research and Education (Deemed to be University)** is an authentic record of our own work carried out during the period from **Jan 2020** under the guidance of Mr. **Dr. R. Ramalakshmi** (Associate Professor).

The work reported in this has not been submitted by me for the award of any other degree of this or any other institute.

SHARIK ANWAR Z
9918004110
RAJA DHANANJEYAN V
9918004098

ACKNOWLEDGEMENT

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

INTRODUCTION

Brick Breaker is incredibly simple and classic brick breaking puzzle game. We made an interactive game based upon the classic game brick breaker. The object of brick breaker is to break the bricks that are distributed around the top of the game screen. The bricks are broken after coming in contact with a ball that bounces around the screen. At the bottom is a paddle that in the classic game moves based on user input.

   The user has to make sure the ball bounces off the paddle without going off the bottom of the screen.Once all the bricks are broken, the player wins the game else,the player loses it.

1.0.1   Objectives

1. To develop a code for the most loved BRICK BREAKER GAME

2. To implement a project for developing the BRICK BREAKER GAME

Chapter 2

PROJECT DESCRIPTION

Modules and packages used in the project
The two main packages used in the project are,
1.Swing Package
2.AWT(Abstract Windowing Toolkit) Package)

AWT PACKAGE
AWT stands for Abstract Window Toolkit. It is a platform dependent API for creating Graphical User Interface (GUI) for java programs.e java.awt package provides classes for AWT api such as TextField, Label, TextArea, RadioButton, CheckBox, Choice, List etc.

SWING PACKAGE
Java Swing tutorial is a part of Java Foundation Classes (JFC) that is used to create window-based applications. It is built on the top of AWT (Abstract Windowing Toolkit) API and entirely written in java.

Unlike AWT, Java Swing provides platform-independent and lightweight components.

JPanel
JPanel, a part of Java Swing package, is a container that can store a group of components. The main task of JPanel is to organize components, various layouts can be set in JPanel which provide better organisation of components, however it does not have a title bar.

JFrame
The javax.swing.JFrame class is a type of container which inherits the java.awt.Frame
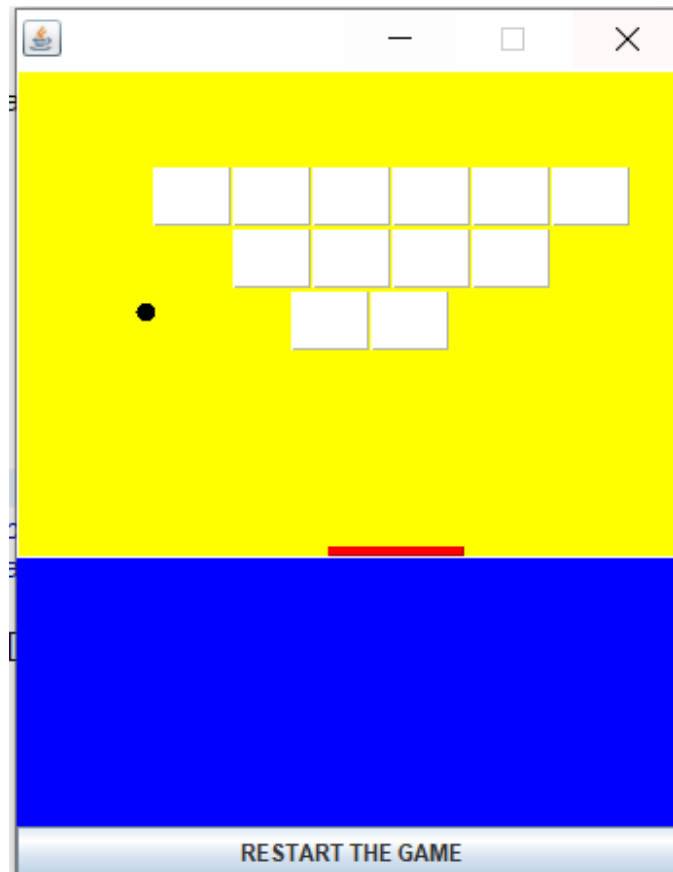
Figure 2.1: Brick Breaker Game

class. JFrame works like the main window where components like labels, buttons, textfields are added to create a GUI.

JButton
The JButton class is used to create a labeled button that has platform independent implementation. The application result in some action when the button is pushed. It inherits AbstractButton class.

Figure 2.2: After winning the game

Figure 2.3: After loosing the game

Chapter 3

CONCLUSION

In all, me and Anwar had a lot of fun creating and playing the classic Brick Breaker. Brick Breaker was a successful educational experience that solidified our understanding of java and its tools. Brick Breaker demonstrated our learning and new found expertise in Java coding.

Appendices

SOURCE CODE

```java
import java.awt.BorderLayout;
import java.awt.Color;
import java.awt.Font;
import java.awt.Graphics;
import java.awt.Rectangle;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.KeyEvent;
import java.awt.event.KeyListener;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JPanel;




public class BrickBreaker extends JPanel implements
    ↪ KeyListener,
  ActionListener, Runnable {

 static boolean right = false;
 static boolean left = false;

 int ballx = 160;
 int bally = 218;

 int batx = 160;
 int baty = 245;

 int brickx = 70;
 int bricky = 50;

 int brickBreadth = 40;
 int brickHeight = 30;
```

```java
  Rectangle  Ball  =  new  Rectangle(ballx,  bally,  10,  10);
  Rectangle  Bat  =  new  Rectangle(batx,  baty,  70,  5);
  // Rectangle  Brick;// = new Rectangle(brickx,  bricky,
      ↪ 30,  10);
  Rectangle[]  Brick  =  new  Rectangle[12];
//reverses......==>
int  movex  =  −1;
int  movey  =  −1;
boolean  ballFallDown  =  false;
boolean  bricksOver  =  false;
int  count  =  0;
String  status;

  BrickBreaker()  {

  }

  public static void  main(String[]  args)  {
   JFrame  frame  =  new  JFrame();
   BrickBreaker  game  =  new  BrickBreaker();
   JButton  button  =  new  JButton("RESTART_THE_GAME");
   frame.setSize(350,  450);
   frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

   frame.add(game);
   frame.add(button,  BorderLayout.SOUTH);
   frame.setLocationRelativeTo(null);
   frame.setResizable(false);
   frame.setVisible(true);
   button.addActionListener(game);

   game.addKeyListener(game);
   game.setFocusable(true);
   Thread  t  =  new  Thread(game);
   t.start();

  }
```

```java
public void paint(Graphics g) {
 g.setColor(Color.LIGHT_GRAY);
 g.fillRect(0, 0, 500, 450);
 g.setColor(Color.white);
 g.fillOval(Ball.x, Ball.y, Ball.width, Ball.height);
 g.setColor(Color.yellow);
 g.fill3DRect(Bat.x, Bat.y, Bat.width, Bat.height,
   ↪ true);
 g.setColor(Color.BLACK);
 g.fillRect(0, 251, 450, 200);
 g.setColor(Color.blue);
 g.drawRect(0, 0, 343, 250);
 for (int i = 0; i < Brick.length; i++) {
  if (Brick[i] != null) {
   g.fill3DRect(Brick[i].x, Brick[i].y, Brick[i].width
     ↪ ,
     Brick[i].height, true);
  }
 }

 if (ballFallDown == true || bricksOver == true) {
  Font f = new Font("Arial", Font.BOLD, 20);
  g.setFont(f);
  g.drawString(status, 70, 120);
  ballFallDown = false;
  bricksOver = false;
 }

}


public void run() {


    createBricks();



 while (true) {
```

```java
//    if (gameOver == true){return;}
   for (int i = 0; i < Brick.length; i++) {
    if (Brick[i] != null) {
     if (Brick[i].intersects(Ball)) {
      Brick[i] = null;
      // movex = -movex;
      movey = -movey;
      count++;

    }
    }



    if (count == Brick.length) {
     bricksOver = true;
     status = "CONGRATULATIONS! YOU_WON_THE_GAME";
     repaint();
    }

    repaint();
    Ball.x += movex;
    Ball.y += movey;

    if (left == true) {

     Bat.x -= 3;
     right = false;
    }
    if (right == true) {
     Bat.x += 3;
     left = false;
    }
    if (Bat.x <= 4) {
     Bat.x = 4;
    } else if (Bat.x >= 298) {
     Bat.x = 298;
    }

    if (Ball.intersects(Bat)) {
```

```
    movey = −movey;
    // if(Ball.y + Ball.width >=Bat.y)
  }


  if (Ball.x <= 0 || Ball.x + Ball.height >= 343) {
   movex = −movex;
  }
  if (Ball.y <= 0) {// ////////////////|| bally + Ball
    ↪ .height >= 250
   movey = −movey;

  if (Ball.y >= 250) {// when ball falls below bat
    ↪ game is over...
   ballFallDown = true;
   status = "YOU_LOST_THE_GAME";
   repaint();
// System.out.print("game");
  }
  try {
   Thread.sleep(10);
  } catch (Exception ex) {
  }

 }

}




@Override
public void keyPressed(KeyEvent e) {
 int keyCode = e.getKeyCode();
 if (keyCode == KeyEvent.VK_LEFT) {
  left = true;
  // System.out.print("left");
 }

 if (keyCode == KeyEvent.VK_RIGHT) {
```

```java
    right = true;
    // System.out.print("right");
  }
}

@Override
public void keyReleased(KeyEvent e) {
 int keyCode = e.getKeyCode();
 if (keyCode == KeyEvent.VK_LEFT) {
  left = false;
 }

 if (keyCode == KeyEvent.VK_RIGHT) {
  right = false;
 }
}

@Override
public void keyTyped(KeyEvent arg0) {

}

@Override
public void actionPerformed(ActionEvent e) {
 String str = e.getActionCommand();
 if (str.equals("RESTART_THE_GAME")) {
  this.restart();

 }
}

public void restart() {

 requestFocus(true);
 initializeVariables();
 createBricks();
 repaint();
}

public void initializeVariables(){
```

```java
        ballx = 160;
        bally = 218;
        batx = 160;
        baty = 245;



        brickx = 70;
        bricky = 50;



        Ball = new Rectangle(ballx, bally, 10, 10);
        Bat = new Rectangle(batx, baty, 70, 5);
        // Rectangle Brick;// = new Rectangle(brickx,
            ↪ bricky, 30, 10);
        Brick = new Rectangle[12];

        movex = -1;
        movey = -1;
        ballFallDown = false;
        bricksOver = false;
        count = 0;
        status = null;


}
public void createBricks(){


        for (int i = 0; i < Brick.length; i++) {
         Brick[i] = new Rectangle(brickx, bricky,
             ↪ brickBreadth, brickHeight);
          if (i == 5) {
           brickx = 70;
           bricky = (bricky + brickHeight + 2);


          }
```

```
        if (i == 9) {
         brickx = 100;
         bricky = (bricky + brickHeight + 2);


        }
        brickx += (brickBreadth+1);
      }
  }

}
```