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

The Brick Breaker which was originally called the Breakout was developed in 1976. The popular game Pong which was released 4 years earlier, had a big influence on the Breakout game. In the late 70's computers were only available for the universities and only for studies, therefore the first brick breaker game was created as an arcade game.
As the breakout came out it became immediately famous. As a result, there were released over a twenty famous Breakout games. One popular game is the Arkanoid that spawned many clones of its own. Breakout's clones actually were released to every possible platform from the arcade console machine in 1976 to PS3 in 2009. Although each game is unique the main concept is the same in each breakout game- keep the ball away from a border and break bricks.
The big change came in 2007 when brick breaker was released. Brick Breaker is the first game that was created for the cell phone platform- BlackBerry. The Brick Breaker was record-breaking in sales. The Brick Breaker has re-founded the Breakout style genre, In fact, many people think that the brick breaker is the original name of this genre. Simultaneously, as the internet evolved many breakout style games were uploaded to the internet and the fast successful of the brick breaker promoted it as well.
Over the time as many users moved to the internet , the online Brick Breaker games have the winning formula; easy and accessible. Therefore this game is the most common. There are a lot of different and interesting versions.

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

First and foremost, I wish to thank the Almighty God for his grace and benediction tocomplete this Project work successfully. I would like to convey my special thanks from the bottom of my heart to my dear Parents and affectionate Family members for their honest support for the completion of this Project work.

I express deep sense of gratitude to "Kalvivallal" Thiru. T. Kalasalingam B.com., Founder Chairman, "Ilayavallal" Dr.K.Sridharan Ph.D., Chancellor, Dr.S.ShasiAnand, Ph.D., Vice President (Academic) , Mr.S.ArjunKalasalingam M.S., Vice President (Administration) , Dr.R.Nagaraj,Vice-Chancellor,Dr.V.Vasudevan Ph.D., Registrar , Dr.P.Deepalakshmi Ph.D., Dean (School of Computing) . And also a special thanks to Dr. A. FRANCIS SAVIOUR DEVARAJ. Head Department of CSE, Kalasalingam Academy of Research and Education forgranting the permission and providing necessary facilities to carry out Project work.

I would like to express my special appreciation and profound thanks to my enthusiastic Project Supervisor Dr.R.Ramalakshmi Ph.D, Associate Professor at Kalasalingam Academy of Research and Education [KARE] for her inspiring guidance, constant encouragement with my work during all stages. I am extremely glad that I had a chance to do my Project under her Guide, who truly practices and appreciates deep thinking. I will be forever indebted to my Guide for all the time he has spent with me in discussions. And during the most difficult times when writing this report, he gave me the moral support and the freedom I needed to move on. Also, i thank my friends for helping me out in completing the project successfully.

SHARIK ANWAR Z
9918004110
RAJA DHANANJEYAN V
9918004098

TABLE OF CONTENTS

LIST OF FIGURES

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: Classes In Swing

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.

Chapter 3

CLASSES USED IN THE CODE

ActionEvent
A semantic event which indicates that a component-defined action occured. This high-level event is generated by a component (such as a Button) when the component-specific action occurs (such as being pressed). The event is passed to every every ActionListener object that registered to receive such events using the component's addActionListener method. The object that implements the ActionListener interface gets this ActionEvent when the event occurs. The listener is therefore spared the details of processing individual mouse movements and mouse clicks, and can instead process a "meaningful" (semantic) event like "button pressed".

ActionListener
ActionListener in Java is a class that is responsible in handling all action events such as when the user clicks on a component. Mostly, action listeners are used for JButtons. An ActionListener can be used by the implements keyword to the class definition. It can also be use separately to the class by creating a new class that implements it. It should also be imported to your project. The method actionPerformed handles the all the actions, and also here, you will defined or write your own codes what will happened when an action occured.

KeyListener

The Java KeyListener is notified whenever you change the state of key. It

is notified against KeyEvent. The KeyListener interface is found in java.awt.event package. It has three methods.

Methods of KeyListener interface The signature of 3 methods found in KeyListener interface are given below:

public abstract void keyPressed(KeyEvent e); public abstract void keyReleased(KeyEvent e); public abstract void keyTyped(KeyEvent e);

KeyEvents

It is a subclass of the abstract InputEvent class and is generated when the user presses or releases a key or does both i.e. types a character.When a key is pressed, the event generated is KEY-PRESSED

When a key is released, the event generated is KEY-RELEASED

When a character is typed on the keyboard, that is a key is pressed and then released, the event generated is KEY-TYPED

Following are the types of methods provided by KeyEvent class

int getKeyCode() It is used for getting the integer code associated with a key. It is used for KEY-PRESSED and KEY-RELEASED events. The keycodes are defined as constants in KeyEvent class.

BorderLayout

The BorderLayout is used to arrange the components in five regions: north, south, east, west and center. Each region (area) may contain one component only. It is the default layout of frame or window. The BorderLayout provides five constants for each region:

public static final int NORTH public static final int SOUTH public static final int EAST public static final int WEST public static final int CENTER.

Color

The Color class states colors in the default sRGB color space or colors in arbitrary color spaces identified by a ColorSpace.

Class declaration Following is the declaration for java.awt.Color class:

public class Color extends Object implements Paint, Serializable.

Graphics

The Graphics class is the abstract super class for all graphics contexts which allow an application to draw onto components that can be realized on various devices, or onto off-screen images as well.

A Graphics object encapsulates all state information required for the basic rendering operations that Java supports. State information includes

the following properties.

The Component object on which to draw.

A translation origin for rendering and clipping coordinates.

The current clip.

The current color.

The current font.

The current logical pixel operation function.

The current XOR alternation color

Class declaration Following is the declaration for java.awt.Graphics class:

public abstract class Graphics extends Object.

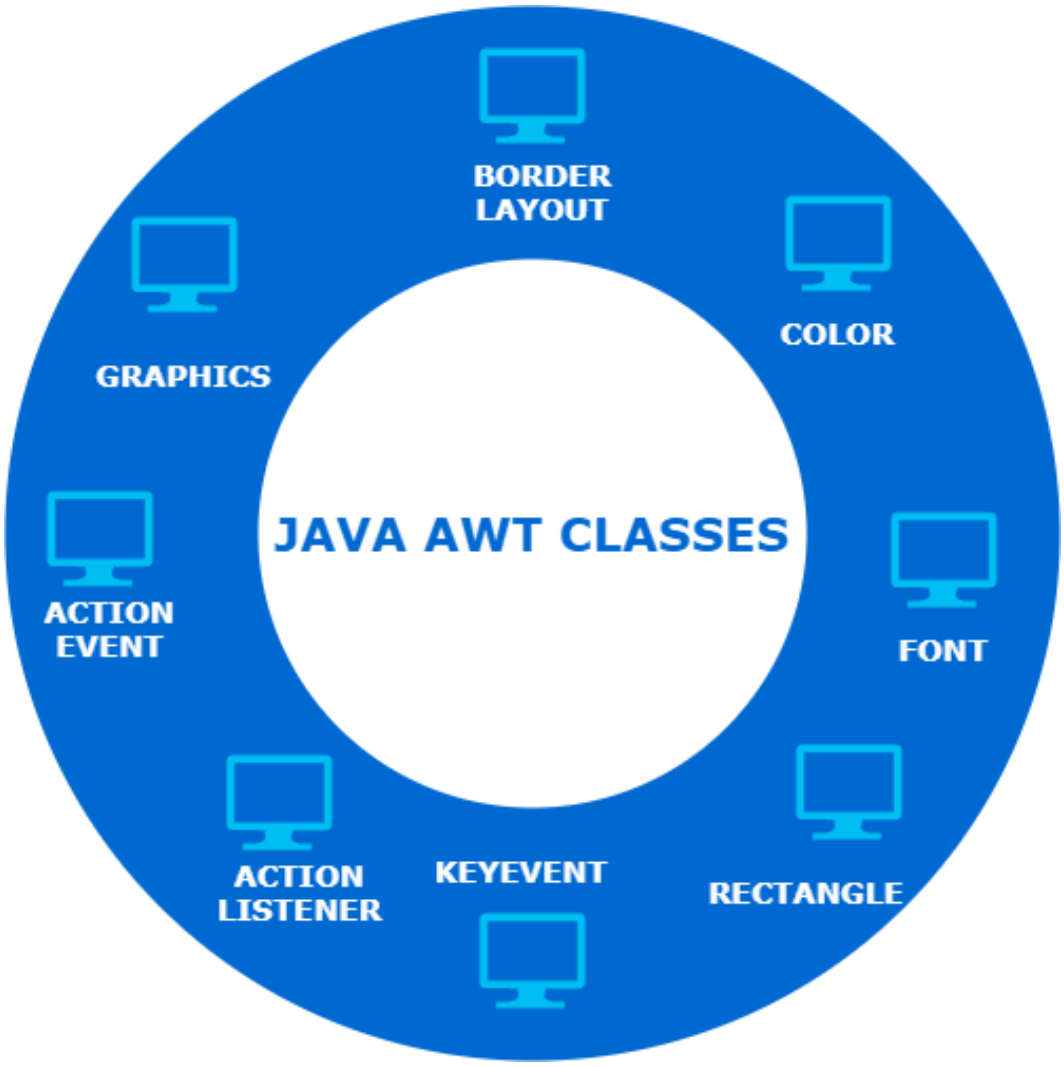

Figure 3.1: Classes In AWT

Chapter 4

IMAGES OF THE PROJECT

Figure 4.1: Actual Brick Breaker Game

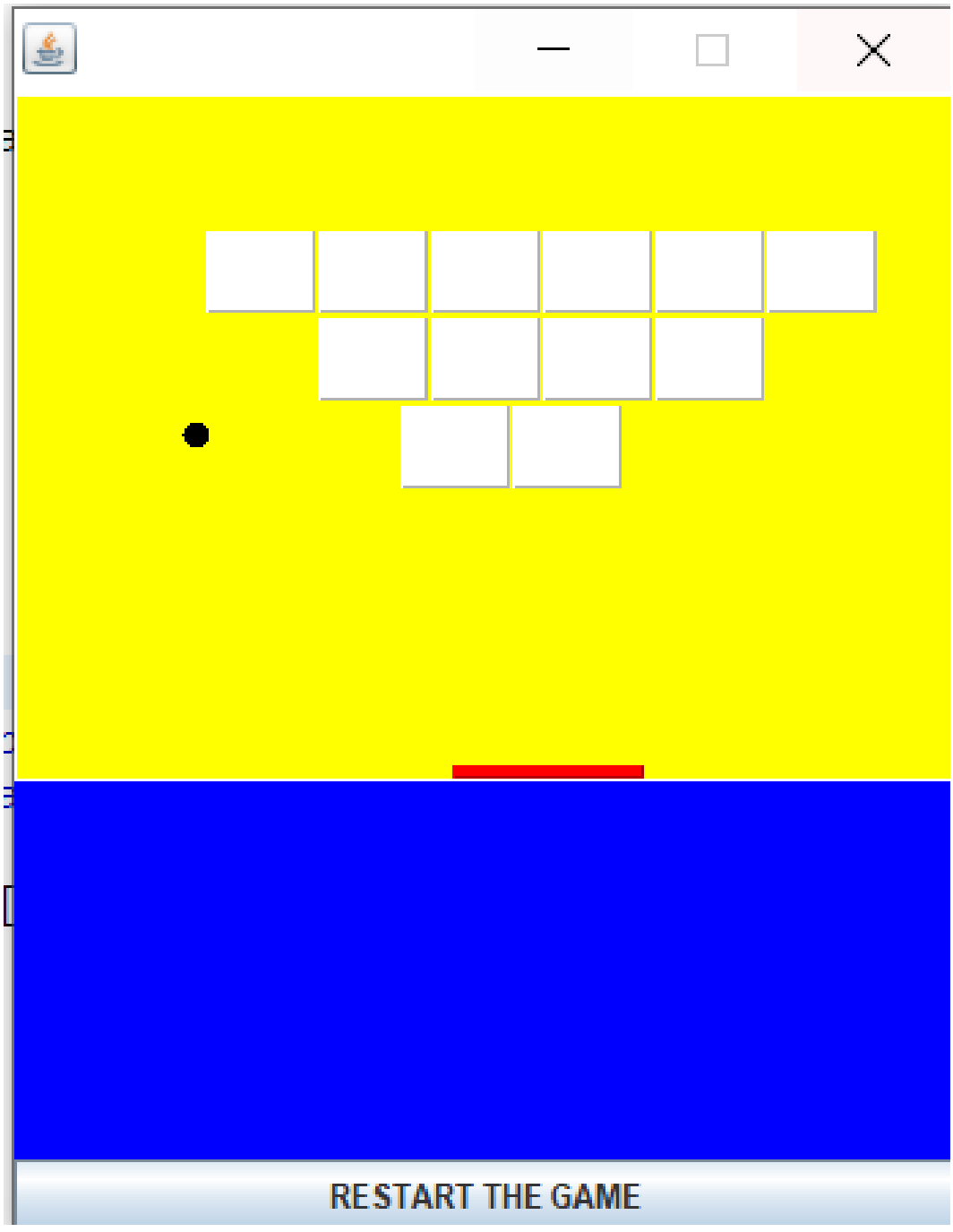

Figure 4.2: Brick Breaker Game

Figure 4.3: After winning the game

Figure 4.4: After loosing the game

Chapter 5

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.

SOURCE CODE

```
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;
```

```
 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();

 }
```

```
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) {
```

```
//    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) {
```

```
    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(){
```

```
        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);
       }
 }

}
```