DON BOSCO INSTITUTE OF TECHNOLOGY

Skill Lab: C++ and Java Programming
MINI PROJECT REPORT

On

"Brick Breaker Game"
2021-22

Submitted By:

Sarika Galphade	36
Sakshi Morey	34
Russel Dmello	32
Shruti Gokhale	40

Under the guidance of
Ms. Deepali Kayande

Mini Project Title **:** Brick Breaker Game

Institute Name **:** Don Bosco Institute of Technology.

Institute Address **:** Premier Automobiles Road,
Kurla (West), Mumbai – 400070

Department **:** Electronics and Telecommunication

Class **:** Second Year (Sem 3)

Project Group Members :

	Names of students	**Roll No.**
1.	Russel Dmello	32
2.	Sakshi Morey	34
3.	Sarika Galphade	36
4.	Shruti Gokhale	40

Date of Submission **: 10|12|2021**

Guide : Ms. Deepali Kayande

TABLE OF CONTENTS

SR. NO.	CONTENT	PAGE NO.
CHAPTER 1	INTRODUCTION	4
CHAPTER 2	PROBLEM DEFINITION MODULES (IF ANY)	5
CHAPTER 3	IMPLEMENTATION	5
CHAPTER 4	RESULTS(SNAPSHOTS)	8
CHAPTER 5	CONCLUSION	9
CHAPTER 6	REFERENCES	

CHAPTER 1
INTRODUCTION

1.1 Overview

Java is a general-purpose, concurrent, class-based, object-oriented computer programming language that is specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere" (WORA), meaning that code that runs on one platform does not need to be recompiled to run on another. Java applications are typically compiled to byte code (class file) that can run on any Java virtual machine (JVM) regardless of computer architecture.

1.2 The Basic GUI (graphical user interface) Application

A **JFrame** is an independent window that can, for example, act as the main window of an application. One of the most important things to understand is that a JFrame object comes with many of the behaviors of windows already programmed in. In particular, we has one Jframe “InterfaceForm” that contents all the components, which enables the user to work on our system easily and draw any shape in Panel and the ability to be opened and closed.

JPanel is another of the fundamental classes in Swing [6]. The basic JPanel is, again, just a blank rectangle. There are two ways to make a useful JPanel : The first is to add other components to the panel; the second is to draw something in the panel. Both of these techniques are illustrated in our sample project. In fact, you will find JPanel in the program: content, display Panel, which is used as a drawing surface.

Events and Listeners: In order to program the behavior of a GUI, you have to write event-handling methods to respond to the events that you are interested in. The most common technique for handling events in Java is to use event listeners. A listener is an object that includes one or more event-handling methods. When an event is detected by another object, such as a button or menu, the listener object is notified and it responds by running the appropriate event-handling method. An event is detected or generated by an object.

- **PROBLEM DEFINITION**

To create a Brick breaker game using GUI interface

IMPLEMENTATION

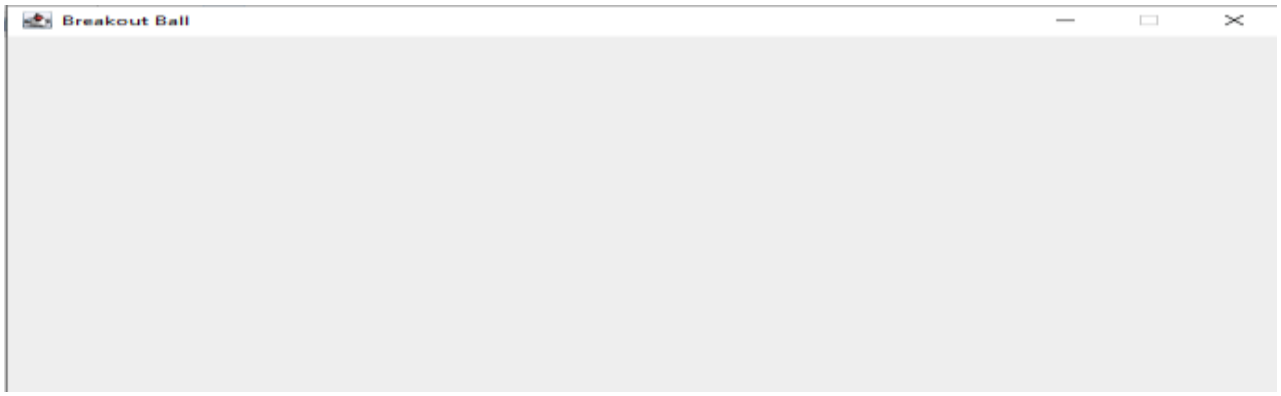
In a Java GUI program, each GUI component in the interface is represented by an object in the program. One of the most fundamental types of component is the window. Windows have many behaviors. They can be opened and closed.
They can be resized. 5 They have “titles” that are displayed in the title bar above the window. And most important, they can contain other GUI components such as buttons and menus. Java, of course, has a built-in class to represent windows.
There are actually several different types of window, but the most common type is represented by the JFrame class (which is included in the package javax.swing)

JFrame Code:

```
package brickBreaker;
import javax.swing.JFrame;
public class Main {

	public static void main(String[] args) {
		JFrame obj= new JFrame();
		Gameplay gameplay = new Gameplay();
	  obj.setBounds(10,10,700,600);
	  obj.setTitle("Breakout Ball");
	  obj.setResizable(false);
	  obj.setVisible(true);
	  obj.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
	  obj.add(gameplay);
	}

}
```



Shapes

The Graphics class includes a large number of instance methods for drawing various shapes, such as lines, rectangles, and ovals. The shapes are specified using the (x,y) coordinate system described above. They are drawn in the current drawing color of the graphics context. The current drawing color is set to the foreground color of the component when the graphics context is created, but it can be changed at any time using the setColor() method

drawOval(int x, int y, int width, int height): Draws the outline of an oval. The oval is one that just fits inside the rectangle specified by x, y, width, and height. If width equals height, the oval is a circle.

fillOval(int x, int y, int width, int height): Draws a filled-in oval.

```
// background
 g.setColor(Color.black);
 g.fillRect(1,1, 692, 592);

 //drawing map
 map.draw((Graphics2D)g);

 // borders
 g.setColor(Color.yellow);
 g.fillRect(0,0,3,592);
 g.fillRect(0,0,692,3);
 g.fillRect(691,0,3,592);

 // scores
 g.setColor(Color.white);
 g.setFont(new Font("serif", Font.BOLD, 25));
 g.drawString(""+score, 590, 30);

//the paddle
 g.setColor(Color.green);
 g.fillRect(playerX, 550,100, 8);

  //the ball
 g.setColor(Color.yellow);
 g.fillOval(ballposX, ballposY, 20 ,20);
```

Painting in AWT and Swing

In a graphical system [10], a windowing toolkit is usually responsible for providing a framework to make it relatively painless for a graphical user interface (GUI) to render the right bits to the screen at the right time.
Both the AWT (abstract windowing toolkit) and Swing provide such a framework.
But the APIs that implement it are not well understood by some developers -- a problem that has led

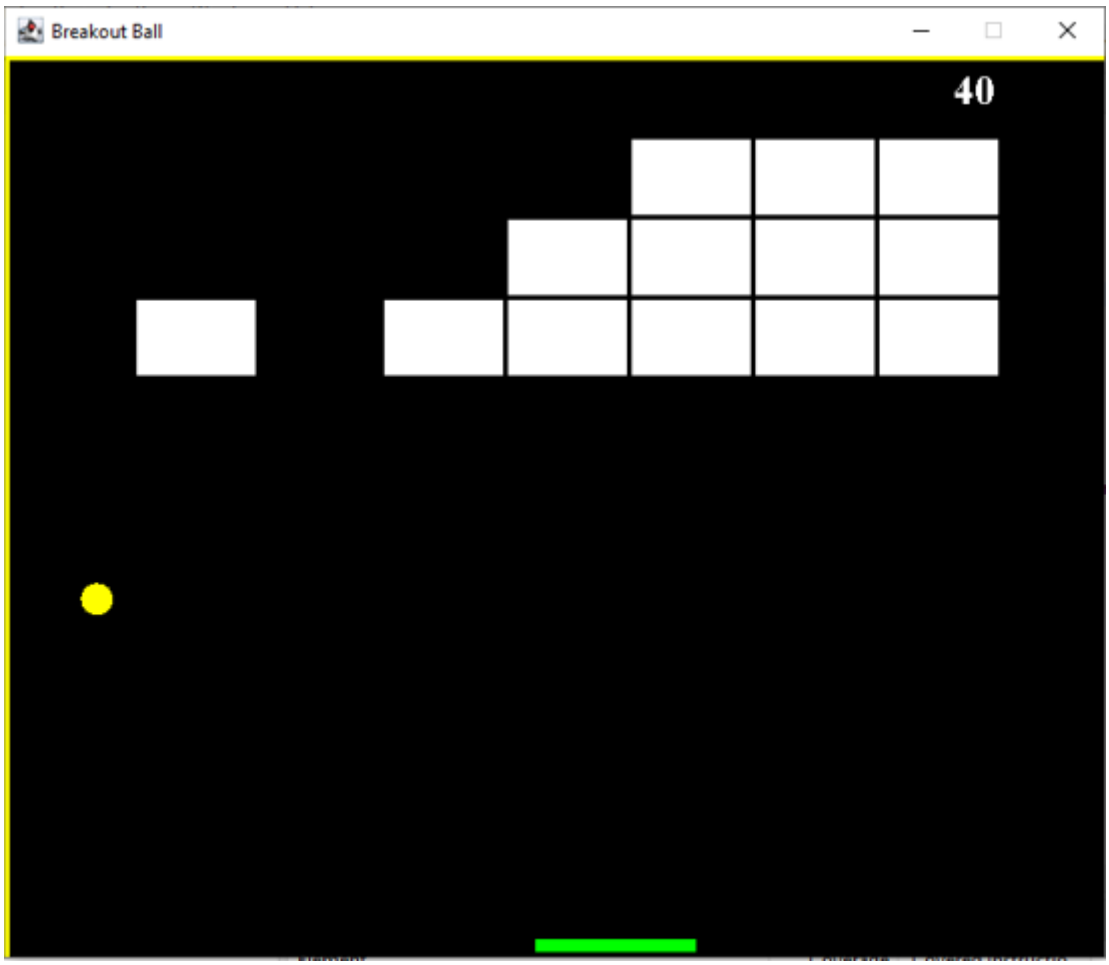
to programs not performing as well as they could. This section explains the AWT and Swing paint mechanisms in detail. Its purpose is to help developers write correct and efficient GUI painting code.

```
import java.awt.Color;
import java.awt.Font;
import java.awt.Graphics;
import java.awt.Graphics2D;
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.Timer;

import javax.swing.JPanel;
```

RESULT


Breakout Ball
40
Game Over, Scores:
Press Enter to Restart


Breakout Ball
40

CONCLUSION

In conclution our group had a lot of fun creating the game and playing it. Brick Breaker was a successful educational experience that solidified our understanding of Java and its various tools.
Breaker successfully ran at the end and provided a fun experience for all of us.

REFERENCES

https://www.researchgate.net/publication/306918533_Final_Report_Java_Programming_Language_A_Simple_project_to_Draw_Paint_Java_language

https://www.daitm.org.in/wp-content/uploads/2019/04/Gr.-06library-project-report.pdf

https://www.academia.edu/13719598/Project_report_on_game_development