VISVESVARAYA TECHNOLOGICAL UNIVERSITY
BELGAUM-590014

An Internship Report on

"PONG GAME using java swings"

*Submitted in partial fulfillment of the requirements for the award of the degree of **Bachelor of Engineering in Artificial Intelligence and Machine Learning***

Submitted by:

BM SOMASHEKAR
(1DT21AI008)
SHREYAS H S
(1DT21AI052)

Guide	**HOD**
Mr. Vijay M Kashyap	**Dr. Sandhya N**
Asst. Professor,	**Professor & Head,**
DSATM	**DSATM**

DAYANANDA SAGAR ACADEMY OF TECHNOLOGY AND MANAGEMENT

Department of Artificial Intelligence and Machine Learning

(Accredited by NAAC with A+ Grade)

Udayapura, Kanakapura Road, Bangalore-560082

2022-2023

DAYANANDA SAGAR ACADEMY OF TECHNOLOGY AND MANAGEMENT

Department of Artificial Intelligence and Machine Learning

(Accredited by NAAC with A+ Grade)

Udayapura, Kanakapura Road, Bangalore-560082

2022-2023

CERTIFICATE

Certified that the Internship work entitled “**PongGame using java swings”** carried out by **BM SOMASHEKAR(1DT21AI008)**and **SHREYAS H S(1DT21AI052)**, are bonafide students of **Dayananda Sagar Academy of Technology and Management** in partial fulfillment for the award of Bachelor of Engineering in **Artificial Intelligence and Machine Learning** of the Visvesvaraya Technological University, Belgaum during the year 2022-23. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the report deposited in the departmental library. The internship report has been approved as it satisfies the academic requirements in respect of internship work prescribed for the said degree.

Signature of Guide
Mr. Vijay M Kashyap
Asst. Prof.
Dept. of AIML

Signature of HOD
Dr. Sandhya N
Professor & Head
Dept. of AIML

STUDENT DECLARATION

We, students **BM SOMASHEKAR, 1DT21AI008** and **SHREYAS H S, 1DT21AI052,** hereby declare that the presented report of internship titled “**Pong game using Java swings”** is uniquely prepared by us after the completion of one month internship work carried out at “**Infosys Spring Board**”.

We also confirm that the report is prepared for our academic requirement, not for any other purpose.

…………………………

BM SOMASHEKAR

1DT21AI008

Department of Artificial

Intelligence and Machine

Learning

DSATM, Bangalore

…………………………

SHREYAS H S

1DT21AI052

Department of Artificial

Intelligence and Machine

Learning

DSATM, Bangalore

Table of Contents

INTRODUCTION

The code provided consists of four classes: PongGame, GameFrame, GamePanel, and Paddle. This code is a basic implementation of a Pong game, a classic arcade game. The game has two players, each with a paddle that they can use to hit a ball back and forth. The objective of the game is to score points by getting the ball past the opponent's paddle.

PongGame is the main class, which creates an instance of GameFrame and starts the game.

GameFrame is a JFrame that contains the GamePanel, which is where the game is played. In the constructor of GameFrame, the GamePanel is added to the frame, and some properties of the frame are set, such as its size, title, background color, and whether it can be resized.

GamePanel is where the game is played. It extends JPanel and implements Runnable. It has a Thread object that runs the game loop, which updates the game state, checks for collisions, and repaints the screen. The GamePanel also has a Paddle and a Ball object, which are responsible for moving the paddles and ball respectively. The GamePanel also has a Score object that keeps track of the score.

The Paddle class is responsible for controlling the movement of the paddles. It keeps track of the position of the paddle and its dimensions. It also has methods that respond to keyboard events to move the paddle up or down.

Overall, the code provides a basic implementation of a Pong game. However, there are some areas where the code can be improved, such as handling collisions more accurately and implementing a game over state.

TOOLS AND TECHNOLOGY USED

1.**Java**:

Java is a high-level, class-based, object-oriented programming language that is designed to have as few implementation dependencies as possible. It is a general-purpose programming language intended to let programmers write once, run anywhere (WORA), meaning that compiled Java code can run on all platforms that support Java without the need to recompile. Java applications are typically compiled to bytecode that can run on any Java virtual machine (JVM) regardless of the underlying computer architecture. The syntax of Java is similar to C and C++, but has fewer low-level facilities than either of them. The Java runtime provides dynamic capabilities (such as reflection and runtime code modification) that are typically not available in traditional compiled languages. As of 2019, Java was one of the most popular programming languages in use according to GitHub, particularly for client–server web applications, with a reported 9 million developers.

2.javax.swing

javax.swing is a package in Java that provides a set of GUI (Graphical User Interface) components for building desktop applications. It is an extension of java.awt package, which is the Abstract Window Toolkit (AWT) that provides a set of classes for creating a graphical user interface. javax.swing provides a more flexible and feature-rich set of components that are designed to work with the Java look and feel (L&F) themes, such as Windows, Mac, and Metal.

In our code, javax.swing is used to create the GUI for the application. The JFrame class is used to create the main window of the application, while the JLabel, JTextField, and JButton classes are used to create the various GUI components such as the text labels, input fields, and buttons. The ActionListener interface is used to handle events such as button clicks. Overall, javax.swing provides a powerful set of tools for building desktop applications with a graphical user interface.

3.java.awt

In our code, java.awt package is used for creating and managing graphical user interfaces (GUIs) in Java. It provides a set of classes and interfaces for building user interfaces for Java applications.

Some of the key features of java.awt package are:

- Drawing: It provides classes for drawing basic shapes such as lines, rectangles, and ovals on a GUI. It also provides classes for working with images and fonts.

- Layout managers: It provides a set of classes for organizing the components of a GUI. These classes help to position and resize the components based on the size of the window.

- Event handling: It provides classes for handling events such as mouse clicks, key presses, and window events.

In our code, java.awt.Color class is used to set the background color of the JFrame window. Additionally, java.awt.Font class is used to set the font style and size of the text displayed on the JLabel.

4.INTELLIJ IDE

IntelliJ IDEA is an Integrated Development Environment (IDE) for JVM languages designed to maximize developer productivity. It does the routine and repetitive tasks for you by providing clever code completion, static code analysis, and refactorings, and lets you focus on the bright side of software development, making it not only productive but also an enjoyable experience.

CLASSES OF THE PROGRAM

PADDLE CLASS

The below code shows the implementation of a Paddle class in a typical game, which is responsible for controlling the movement and drawing of a paddle on the game screen.
Here is a breakdown of the code:
The Paddle class extends the Rectangle class, which represents a rectangle with integer coordinates and dimensions.The class has several private fields, including the paddle's ID (1 or 2), its vertical velocity (yVelocity), and its speed.
The class has a constructor that initializes the paddle's position, size, and ID.
The class has two methods, keyPressed and keyReleased, which handle the paddle's movement when the player presses or releases a key on the keyboard. The method takes a KeyEvent object as a parameter, which contains information about the key that was pressed or released.
Depending on the paddle's ID and the key that was pressed or released, the methods set the paddle's vertical velocity to move it up or down.
The class has a method called setYVelocity, which sets the paddle's vertical velocity.The class has a method called move, which updates the paddle's position based on its vertical velocity.The class has a method called draw, which draws the paddle on the screen using the fillRect method of the Graphics class. The paddle's color depends on its ID (blue for player 1 and red for player 2).Overall, the Paddle class is an essential component of any game that involves paddles, such as Pong or Breakout. It allows the player to control the movement of the paddle and handles the drawing of the paddle on the game screen.

BALL CLASS

The below code defines the Ball class, which extends the Rectangle class. It represents a ball in a game of Pong. The class contains the following member variables and methods:
random: a Random object used to generate random x and y directions for the ball
xVelocity: the current velocity of the ball in the x direction
yVelocity: the current velocity of the ball in the y direction
initialSpeed: the initial speed of the ball
Ball(int x, int y, int width, int height): a constructor method that takes in the initial position

and size of the ball, and sets its initial x and y velocities to random values
setXDirection(int randomXDirection): a method that sets the ball's x velocity to the given value
setYDirection(int randomYDirection): a method that sets the ball's y velocity to the given value
move(): a method that updates the position of the ball based on its current x and y velocities
draw(Graphics g): a method that draws the ball on the screen using the given Graphics object. It fills an oval with the color white at the current x and y position and with the specified height and width.

SCORE CLASS

This is a Java class called "Score" that extends the "Rectangle" class. It has four instance variables - "GAME_WIDTH" and "GAME_HEIGHT" which are static and shared by all instances of the class, and "player1" and "player2" which hold the scores for two players. The constructor takes two integer arguments "GAME_WIDTH" and "GAME_HEIGHT" and initializes the static variables with those values.The "draw" method takes a "Graphics" object as a parameter and uses it to draw the scores on the screen. It sets the font and color for the text and draws a line at the center of the screen. Then it uses the "player1" and "player2" scores to draw the scores for both players on either side of the line.
The scores are calculated using the modulus and division operators to separate the tens and ones digits of each score. The resulting strings are drawn on the screen at specific x and y coordinates.
Overall, this class is used to display the scores of two players in a game.

GAME FRAME

In a typical game, a game frame is a window that provides a graphical user interface (GUI) for the game. It can contain various components such as buttons, text fields, menus, and graphics. The game frame is the main entry point for the game and serves as the primary user interface for the player.
The game frame typically handles input from the player, such as mouse clicks or keyboard events, and sends this input to the game logic to update the game state. It also displays the game graphics and updates them as the game state changes. The game frame may also provide various options or settings for the player to customize the game experience.
Overall, the game frame is a crucial component of any graphical game as it provides the user interface and controls the game flow.

CONCLUSION

This code implements a Pong game using Java Swing library. It consists of three classes: GameFrame, GamePanel, and Paddle.The GameFrame class creates the game window, sets its properties such as title, size, background color, and adds the GamePanel to it.The GamePanel class handles the game logic, drawing, and user input. It initializes the game objects such as paddles, ball, and score, and starts the game loop. In the game loop, it updates the game objects' position, checks for collisions, and redraws them on the screen. It also listens to the user input through the KeyAdapter class and moves the paddles accordingly.The Paddle class defines the properties and behavior of a paddle object, such as its position, size, and direction of movement. It listens to the user input and changes its direction of movement accordingly.Overall, the code provides a simple implementation of the classic Pong game with two paddles and a ball bouncing between them, and a score keeping syste

APPENDIX

SOURCE CODE

mainclass-

```
public class PongGame {
    public static void main(String[] args){
        GameFrame frame = new GameFrame();

    }
}
```

gameframe-

```
import javax.swing.*;
import java.awt.*;


public class GameFrame extends JFrame {
     GamePanel panel ;

    GameFrame(){
       panel = new GamePanel();
       this.add(panel);
       this.setTitle("PONG GAME");
       this.setResizable(false);
       this.setBackground(Color.black);
       this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
       this.pack();
       this.setVisible(true);
       this.setLocationRelativeTo(null);

    }
}
```

gamepanel-

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.KeyAdapter;
import java.awt.event.KeyEvent;
import java.util.Random;

public class GamePanel extends JPanel implements Runnable{

    static final int GAME_WIDTH = 1000;
    static final int GAME_HEIGHT = (int)(GAME_WIDTH * (0.5555));
    static final Dimension SCREEN_SIZE = new Dimension(GAME_WIDTH,GAME_HEIGHT);
    static final int BALL_DIAMETER = 20;
    static final int PADDLE_WIDTH = 25;
    static final int PADDLE_HEIGHT = 100;
    Thread gameThread;
    Image image;
    Graphics graphics;
    Random random;
    Paddle paddle1;
    Paddle paddle2;
    Ball ball;
    Score score;

    GamePanel(){
        newPaddles();
        newBall();
        score = new Score(GAME_WIDTH,GAME_HEIGHT);
        this.setFocusable(true);
        this.addKeyListener(new AL());
        this.setPreferredSize(SCREEN_SIZE);

        gameThread = new Thread(this);
        gameThread.start();
    }

    public void newBall() {
        random = new Random();
        ball = new
Ball((GAME_WIDTH/2)-(BALL_DIAMETER/2),random.nextInt(GAME_HEIGHT-BALL_DIAMETER),B
ALL_DIAMETER,BALL_DIAMETER);
    }
    public void newPaddles() {
        paddle1 = new
Paddle(0,(GAME_HEIGHT/2)-(PADDLE_HEIGHT/2),PADDLE_WIDTH,PADDLE_HEIGHT,1);
        paddle2 = new
Paddle(GAME_WIDTH-PADDLE_WIDTH,(GAME_HEIGHT/2)-(PADDLE_HEIGHT/2),PADDLE_WIDT
H,PADDLE_HEIGHT,2);
    }
    public void paint(Graphics g) {
        image = createImage(getWidth(),getHeight());
        graphics = image.getGraphics();
        draw(graphics);
        g.drawImage(image,0,0,this);
    }
    public void draw(Graphics g) {
        paddle1.draw(g);
        paddle2.draw(g);
        ball.draw(g);
        score.draw(g);
        Toolkit.getDefaultToolkit().sync(); // I forgot to add this line of code in the video, it helps with the
animation

    }
    public void move() {
        paddle1.move();
        paddle2.move();
        ball.move();
    }
    public void checkCollision() {

        //bounce ball off top & bottom window edges
        if(ball.y <=0) {
            ball.setYDirection(-ball.yVelocity);
        }
        if(ball.y >= GAME_HEIGHT-BALL_DIAMETER) {
            ball.setYDirection(-ball.yVelocity);
        }
        //bounce ball off paddles
        if(ball.intersects(paddle1)) {
            ball.xVelocity = Math.abs(ball.xVelocity);
            ball.xVelocity++; //optional for more difficulty
            if(ball.yVelocity>0)
                ball.yVelocity++; //optional for more difficulty
            else
                ball.yVelocity--;
            ball.setXDirection(ball.xVelocity);
            ball.setYDirection(ball.yVelocity);
        }
        if(ball.intersects(paddle2)) {
            ball.xVelocity = Math.abs(ball.xVelocity);
            ball.xVelocity++; //optional for more difficulty
            if(ball.yVelocity>0)
                ball.yVelocity++; //optional for more difficulty
            else
                ball.yVelocity--;
            ball.setXDirection(-ball.xVelocity);
            ball.setYDirection(ball.yVelocity);
        }
        //stops paddles at window edges
        if(paddle1.y<=0)
            paddle1.y=0;
        if(paddle1.y >= (GAME_HEIGHT-PADDLE_HEIGHT))
            paddle1.y = GAME_HEIGHT-PADDLE_HEIGHT;
        if(paddle2.y<=0)
            paddle2.y=0;
        if(paddle2.y >= (GAME_HEIGHT-PADDLE_HEIGHT))
            paddle2.y = GAME_HEIGHT-PADDLE_HEIGHT;
        //give a player 1 point and creates new paddles & ball
        if(ball.x <=0) {
            score.player2++;
            newPaddles();
            newBall();
            System.out.println("Player 2: "+score.player2);
        }
        if(ball.x >= GAME_WIDTH-BALL_DIAMETER) {
            score.player1++;
            newPaddles();
            newBall();
            System.out.println("Player 1: "+score.player1);
        }
    }
    public void run() {
        //game loop
        long lastTime = System.nanoTime();
        double amountOfTicks =60.0;
        double ns = 1000000000 / amountOfTicks;
        double delta = 0;
        while(true) {
            long now = System.nanoTime();
            delta += (now -lastTime)/ns;
            lastTime = now;
            if(delta >=1) {
                move();
                checkCollision();
                repaint();
                delta--;
            }
        }
    }
    public class AL extends KeyAdapter{
        public void keyPressed(KeyEvent e) {
            paddle1.keyPressed(e);
            paddle2.keyPressed(e);
        }
        public void keyReleased(KeyEvent e) {
            paddle1.keyReleased(e);
            paddle2.keyReleased(e);
        }
    }
}
```

paddle-

```
import java.awt.*;
import java.awt.event.KeyEvent;

public class Paddle extends Rectangle {
    private int id;
    private int yVelocity;
    private int speed = 15;

    public Paddle(int x, int y, int width, int height, int id) {
        super(x, y, width, height);
        this.id = id;
    }

    public void keyPressed(KeyEvent e) {
        switch(id) {
            case 1:
                if (e.getKeyCode() == KeyEvent.VK_W) {
                    setYVelocity(-speed);
                } else if (e.getKeyCode() == KeyEvent.VK_S) {
                    setYVelocity(speed);
                }
                break;
            case 2:
                if (e.getKeyCode() == KeyEvent.VK_UP) {
                    setYVelocity(-speed);
                } else if (e.getKeyCode() == KeyEvent.VK_DOWN) {
                    setYVelocity(speed);
                }
                break;
        }
    }

    public void keyReleased(KeyEvent e) {
        switch(id) {
            case 1:
                if (e.getKeyCode() == KeyEvent.VK_W || e.getKeyCode() == KeyEvent.VK_S) {
                    setYVelocity(0);
                }
                break;
            case 2:
                if (e.getKeyCode() == KeyEvent.VK_UP || e.getKeyCode() == KeyEvent.VK_DOWN) {
                    setYVelocity(0);
                }
            break;
        }
    }

    public void setYVelocity(int yVelocity) {
        this.yVelocity = yVelocity;
    }

    public void move() {
        y = y + yVelocity;
    }

    public void draw(Graphics g) {
        if (id == 1) {
            g.setColor(Color.blue);
        } else {
            g.setColor(Color.red);
        }
        g.fillRect(x, y, width, height);
    }
}
```

ball-

```
import java.awt.*;
import java.util.*;

public class Ball extends Rectangle{

    Random random;
    int xVelocity;
    int yVelocity;
    int initialSpeed = 2;

    Ball(int x, int y, int width, int height){
        super(x,y,width,height);
        random = new Random();
        int randomXDirection = random.nextInt(2);
        if(randomXDirection == 0)
            randomXDirection--;
        setXDirection(randomXDirection*initialSpeed);

        int randomYDirection = random.nextInt(2);
        if(randomYDirection == 0)
            randomYDirection--;
        setYDirection(randomYDirection*initialSpeed);

    }

    public void setXDirection(int randomXDirection) {
        xVelocity = randomXDirection;
    }
    public void setYDirection(int randomYDirection) {
        yVelocity = randomYDirection;
    }
    public void move() {
        x += xVelocity;
        y += yVelocity;
    }
    public void draw(Graphics g) {
        g.setColor(Color.white);
        g.fillOval(x, y, height, width);
    }
}
```

score-

```
import java.awt.*;

public class Score extends Rectangle{

    static int GAME_WIDTH;
    static int GAME_HEIGHT;
    int player1;
    int player2;

    Score(int GAME_WIDTH, int GAME_HEIGHT){
        Score.GAME_WIDTH = GAME_WIDTH;
        Score.GAME_HEIGHT = GAME_HEIGHT;
    }
    public void draw(Graphics g) {
        g.setColor(Color.white);
        g.setFont(new Font("Consolas",Font.PLAIN,60));

        g.drawLine(GAME_WIDTH/2, 0, GAME_WIDTH/2, GAME_HEIGHT);

        g.drawString(String.valueOf(player1/10)+String.valueOf(player1%10), (GAME_WIDTH/2)-85, 50);
        g.drawString(String.valueOf(player2/10)+String.valueOf(player2%10), (GAME_WIDTH/2)+20, 50);
    }
}
```