Laporan Program Game "Catch the Ball" dalam Bahasa Java

Nama : Muhamamd Zakky Ghufron

NIM : 2222105151

Kelas : 2TI03

MK : Pemrograman Berorientasi Objek

Dosen : Novan Zulkarnain, S.T., M.Kom

Pendahuluan

Program "Catch the Ball" adalah permainan sederhana yang dikembangkan menggunakan bahasa pemrograman Java dengan menggunakan pustaka Swing untuk antarmuka grafis. Permainan ini mengharuskan pemain untuk menggerakkan paddle ke kiri atau kanan guna menangkap bola yang jatuh dari atas layar. Setiap bola yang berhasil ditangkap akan menambah skor pemain, sementara bola yang terlewat akan mengurangi jumlah nyawa. Permainan berakhir ketika nyawa pemain habis.

Kode Sumber

Berikut adalah kode sumber lengkap dari program:

```
import java.awt.*;
import java.awt.event.*;
import java.awt.image.BufferedImage;
import java.io.File;
import java.io.IOException;
import javax.imageio.ImageIO;
import javax.swing.*;

public class Main extends JPanel implements ActionListener, KeyListener {
    private static final long serialVersionUID = 1L;

    private Timer timer;
    private int paddleX = 250;
    private int ballX = 100;
    private int ballY = 0;
    private int ballSpeedY = 3;
    private int score = 0;
    private int lives = 3;
    private boolean gameOver = false;
    private BufferedImage backgroundImage;
```

```
    public Main() {
        try {
            backgroundImage = ImageIO.read(new
File("path/to/your/background/image.jpg")); // Path to your background image
        } catch (IOException e) {
            e.printStackTrace();
        }
        timer = new Timer(10, this);
        timer.start();
        addKeyListener(this);
        setFocusable(true);
        setFocusTraversalKeysEnabled(false);
    }

    @Override
    protected void paintComponent(Graphics g) {
        super.paintComponent(g);

        // Draw background image
        if (backgroundImage != null) {
            g.drawImage(backgroundImage, 0, 0, getWidth(), getHeight(), this);
        } else {
            g.setColor(new Color(173, 216, 230)); // Light blue background
            g.fillRect(0, 0, getWidth(), getHeight());
        }

        // Game Over Screen
        if (gameOver) {
            g.setColor(new Color(255, 69, 0)); // Red color for game over text
            g.setFont(new Font("Arial", Font.BOLD, 50));
            g.drawString("Game Over", 150, 250);
            g.setFont(new Font("Arial", Font.BOLD, 30));
            g.drawString("Score: " + score, 230, 300);
            g.drawString("Press 'R' to Restart", 150, 350);
            return;
        }

        // Paddle
        g.setColor(new Color(60, 179, 113)); // Medium sea green paddle
        g.fillRoundRect(paddleX, 550, 100, 10, 10, 10);

        // Ball
        g.setColor(new Color(255, 69, 0)); // Red ball
        g.fillOval(ballX, ballY, 20, 20);

        // Score and Lives
        g.setColor(Color.BLACK);
        g.setFont(new Font("Arial", Font.BOLD, 20));
```

```
        g.drawString("Score: " + score, 10, 20);
        g.drawString("Lives: " + lives, 500, 20);
    }

    @Override
    public void actionPerformed(ActionEvent e) {
        if (gameOver) {
            return;
        }

        ballY += ballSpeedY;
        if (ballY >= 550 && ballX >= paddleX && ballX <= paddleX + 100) {
            ballSpeedY = -ballSpeedY;
            score += 10;
        }
        if (ballY < 0) {
            ballSpeedY = -ballSpeedY;
        }
        if (ballY > 600) {
            lives--;
            if (lives <= 0) {
                gameOver = true;
            } else {
                ballY = 0;
                ballX = (int) (Math.random() * 580);
            }
        }

        repaint();
    }

    @Override
    public void keyPressed(KeyEvent e) {
        int code = e.getKeyCode();
        if (code == KeyEvent.VK_LEFT) {
            paddleX = Math.max(paddleX - 20, 0);
        }
        if (code == KeyEvent.VK_RIGHT) {
            paddleX = Math.min(paddleX + 20, 500);
        }
        if (gameOver && code == KeyEvent.VK_R) {
            restartGame();
        }
    }

    @Override
    public void keyTyped(KeyEvent e) {}
    @Override
```

```
    public void keyReleased(KeyEvent e) {}

    private void restartGame() {
        ballX = 100;
        ballY = 0;
        ballSpeedY = 3;
        paddleX = 250;
        score = 0;
        lives = 3;
        gameOver = false;
        requestFocus(); // Memastikan panel kembali fokus untuk menerima input
dari keyboard
        repaint();
    }

    public static void main(String[] args) {
        JFrame frame = new JFrame("Catch the Ball");
        Main game = new Main();
        frame.add(game);
        frame.setSize(600, 600);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setVisible(true);
    }
}
```

Langkah-langkah logika program

1.Import Perpustakaan yang Diperlukan

import java.awt.*;

import java.awt.event.*;

import java.awt.image.BufferedImage;

import java.io.File;

import java.io.IOException;

import javax.imageio.ImageIO;

import javax.swing.*;

penjelasan:

- java.awt.* dan java.awt.event.*: Untuk komponen GUI dan event handling.

 java.awt.image.BufferedImage: Untuk memuat gambar.

 java.io.File dan java.io.IOException: Untuk mengelola file dan menangani pengecualian I/O.

 javax.imageio.ImageIO: Untuk membaca gambar.

 javax.swing.*: Untuk komponen GUI Swing.

2. Deklarasi Kelas Utama

public class Main extends JPanel implements ActionListener, KeyListener {

    private static final long serialVersionUID = 1L;

Penjelasan:

Kelas Main memperluas JPanel dan mengimplementasikan ActionListener dan KeyListener.

3. Deklarasi Variabel

    private Timer timer;

    private int paddleX = 250;

    private int ballX = 100;

    private int ballY = 0;

    private int ballSpeedY = 3;

    private int score = 0;

    private int lives = 3;

    private boolean gameOver = false;

    private BufferedImage backgroundImage;

Penjelasan:

 timer: Untuk mengatur interval waktu.

 paddleX: Koordinat X paddle.

 ballX, ballY, ballSpeedY: Koordinat X dan Y bola serta kecepatan vertikal bola.

 score: Skor permainan.

 lives: Jumlah nyawa pemain.

 gameOver: Status permainan.

 backgroundImage: Gambar latar belakang.

4. Konstruktor Main

```
    public Main() {
        try {
            backgroundImage = ImageIO.read(new File("path/to/your/background/image.jpg"));
// Path to your background image
        } catch (IOException e) {
            e.printStackTrace();
        }
        timer = new Timer(10, this);
        timer.start();
        addKeyListener(this);
        setFocusable(true);
        setFocusTraversalKeysEnabled(false);
    }
```

Penjelasan:

 Memuat gambar latar belakang.

 Menginisialisasi dan memulai Timer.

 Mengatur KeyListener dan memastikan panel fokus untuk menerima input.

5. Metode paintComponent

```
    @Override
    protected void paintComponent(Graphics g) {
        super.paintComponent(g);
        // Draw background image
        if (backgroundImage != null) {
            g.drawImage(backgroundImage, 0, 0, getWidth(), getHeight(), this);
```

```
} else {
    g.setColor(new Color(173, 216, 230)); // Light blue background
    g.fillRect(0, 0, getWidth(), getHeight());
}
// Game Over Screen
if (gameOver) {
    g.setColor(new Color(255, 69, 0)); // Red color for game over text
    g.setFont(new Font("Arial", Font.BOLD, 50));
    g.drawString("Game Over", 150, 250);
    g.setFont(new Font("Arial", Font.BOLD, 30));
    g.drawString("Score: " + score, 230, 300);
    g.drawString("Press 'R' to Restart", 150, 350);
    return;
}
// Paddle
g.setColor(new Color(60, 179, 113)); // Medium sea green paddle
g.fillRoundRect(paddleX, 550, 100, 10, 10, 10);
// Ball
g.setColor(new Color(255, 69, 0)); // Red ball
g.fillOval(ballX, ballY, 20, 20);
// Score and Lives
g.setColor(Color.BLACK);
g.setFont(new Font("Arial", Font.BOLD, 20));
g.drawString("Score: " + score, 10, 20);
g.drawString("Lives: " + lives, 500, 20);
```

```
}
```

Penjelasan:

- Menggambar gambar latar belakang.
- Menampilkan layar "Game Over" jika permainan berakhir.
- Menggambar paddle dan bola.
- Menampilkan skor dan jumlah nyawa.

6. Metode actionPerformed

```
@Override
public void actionPerformed(ActionEvent e) {
    if (gameOver) {
        return;
    }
    ballY += ballSpeedY;
    if (ballY >= 550 && ballX >= paddleX && ballX <= paddleX + 100) {
        ballSpeedY = -ballSpeedY;
        score += 10;
    }
    if (ballY < 0) {
        ballSpeedY = -ballSpeedY;
    }
    if (ballY > 600) {
        lives--;
        if (lives <= 0) {
            gameOver = true;
        } else {
```

```
            ballY = 0;
            ballX = (int) (Math.random() * 580);
        }
    }
    repaint();
}
```

Penjelasan:

- Menggerakkan bola berdasarkan kecepatan.
- Memeriksa tabrakan bola dengan paddle dan mengubah arah bola serta menambah skor.
- Mengubah arah bola jika mengenai bagian atas layar.
- Mengurangi nyawa jika bola jatuh ke bawah layar.
- Mengatur ulang posisi bola jika nyawa masih ada.
- Mengakhiri permainan jika nyawa habis.

7. Metode keyPressed

```
@Override
public void keyPressed(KeyEvent e) {
    int code = e.getKeyCode();
    if (code == KeyEvent.VK_LEFT) {
        paddleX = Math.max(paddleX - 20, 0);
    }
    if (code == KeyEvent.VK_RIGHT) {
        paddleX = Math.min(paddleX + 20, 500);
    }
    if (gameOver && code == KeyEvent.VK_R) {
        restartGame();
    }
```

```
    }
```

Penjelasan:

- Menggerakkan paddle ke kiri atau kanan berdasarkan input keyboard.
- Mereset permainan jika tombol 'R' ditekan setelah game over.

8. Metode keyTyped dan keyReleased

```
    @Override
    public void keyTyped(KeyEvent e) {}
    @Override
    public void keyReleased(KeyEvent e) {}
```

Penjelasan:

Metode kosong yang diperlukan oleh KeyListener tetapi tidak digunakan.

9. Metode restartGame

```
    private void restartGame() {
        ballX = 100;
        ballY = 0;
        ballSpeedY = 3;
        paddleX = 250;
        score = 0;
        lives = 3;
        gameOver = false;
        requestFocus(); // Memastikan panel kembali fokus untuk menerima input dari keyboard
        repaint();
    }
```

Penjelasan:

- Menginisialisasi ulang variabel permainan.

 Mengatur fokus panel untuk menerima input keyboard.

 Memanggil repaint untuk memperbarui tampilan.

10. Metode main

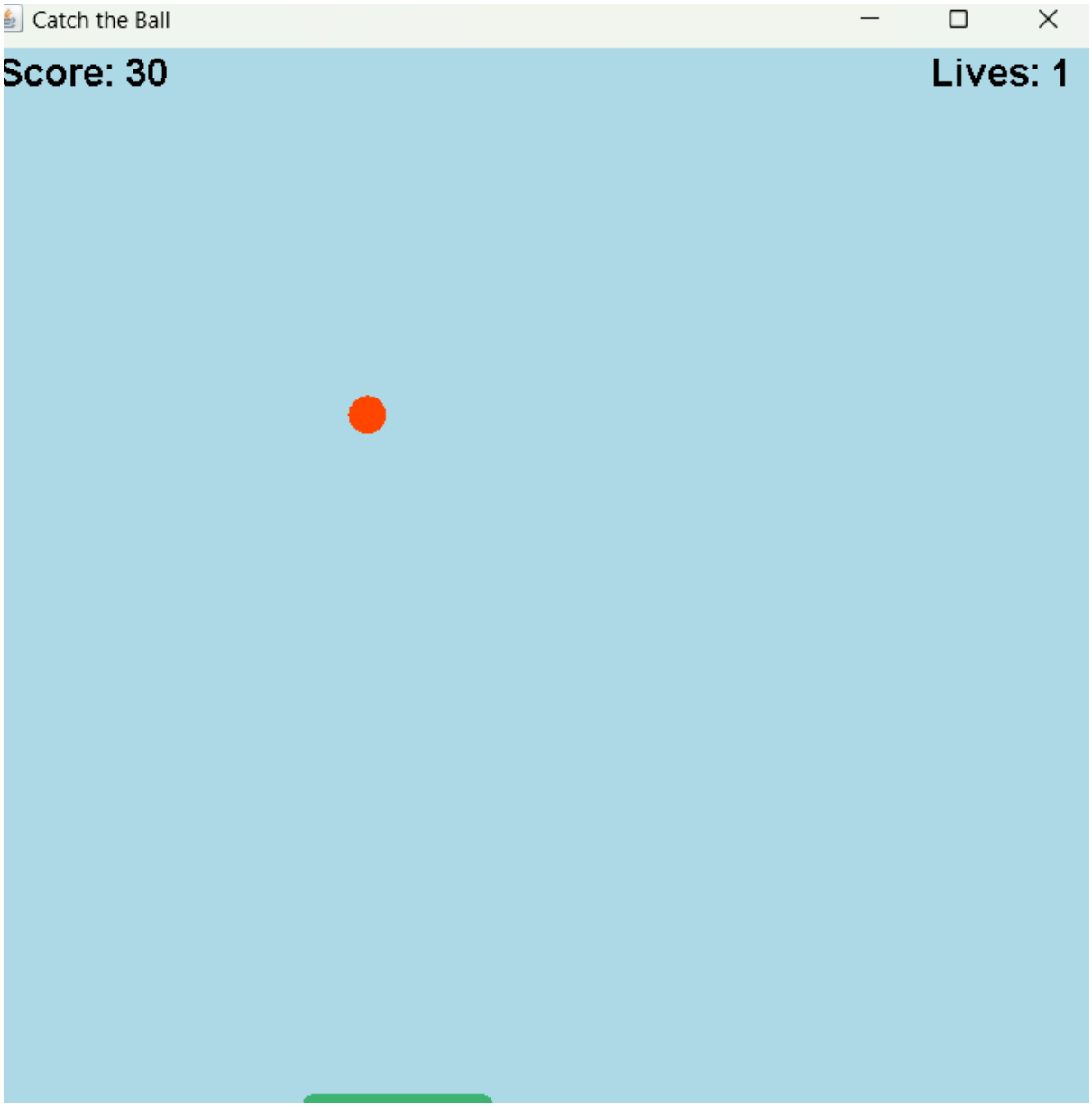
```
public static void main(String[] args) {
    JFrame frame = new JFrame("Catch the Ball");
    Main game = new Main();
    frame.add(game);
    frame.setSize(600, 600);
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setVisible(true);
}
```

Penjelasan:
 Membuat jendela JFrame dan menambahkan panel permainan.
 Mengatur ukuran jendela dan properti lainnya.
 Menampilkan jendela.

Kesimpulan

Program game "Catch the Ball" ini adalah contoh sederhana tetapi efektif untuk memperkenalkan konsep dasar pemrograman game menggunakan Java dan Swing. Program ini mencakup elemen-elemen penting seperti pengendalian animasi dengan timer, pengolahan input dari pengguna, dan manipulasi grafis. Dengan memperluas konsep-konsep ini, lebih banyak fitur dan kompleksitas dapat ditambahkan ke dalam game di masa depan.

Hasil dari output sebuah program game catch the ball dalam bahasa java




Catch the Ball
Game Over
Score: 0
Press 'R' to Restart