Documentatie Proiect Ball Jump

Visoiu Radu
10LF333

Introducere în proiect

Acest proiect reprezintă o implementare simplificată a unui joc de tip platformer inspirat de clasicul **Doodle Jump**, dezvoltat în Java utilizând biblioteca JavaFX pentru interfața grafică. Jucătorul controlează un personaj care sare pe platforme pentru a avansa și acumula puncte. Jocul include platforme de diferite tipuri, cum ar fi platforme normale, platforme mov (care devin inactive după o săritură) și platforme galbene (care oferă un impuls suplimentar la săritură).

Lab1: Introducere în Java (output, tipuri de valori, funcții)

Tipuri de valori:

```
private double x, y, velocityX = 0, velocityY = 0;
private final double width = 40, height = 40;
private boolean onGround = false;
```

Functii:

```
public void jump() {
    if (onGround) {
        velocityY = -7;
        onGround = false;
    }
}
```

Lab2: Introducere în Java (input, for, while, switch, if)

Input:

```java
private void setupInput() {
    scene.setOnKeyPressed(event -> {
        if (event.getCode() == KeyCode.LEFT) leftPressed = true;
        if (event.getCode() == KeyCode.RIGHT) rightPressed = true;
    });

    scene.setOnKeyReleased(event -> {
        if (event.getCode() == KeyCode.LEFT) leftPressed = false;
        if (event.getCode() == KeyCode.RIGHT) rightPressed = false;
    });
}
```

For:

```java
for (int i = 0; i < 5; i++) {
    platforms.add(new Platform(random.nextInt( bound: 340),  y: 600 - i * 120, width: 60, height: 10,PlatformType.NORMAL));
}
```

If:

```java
if (!playerLanded) {
    player.setVelocityY(player.getVelocityY());
    player.setOnGround(false);
}
```

Lab3: Colecții Java (Array, List, Map) – alegeți minimum una și folosiți câteva metode specifice (ex.: sortare, filtrare, etc.)

List:

```java
private final List<Platform> platforms = new ArrayList<>();
```

```
platforms.add(new Platform(random.nextInt( bound: 340),  y: 600 - i * 120, width: 60, height: 10,PlatformType.NORMAL));
```

```
platforms.removeIf(Platform::isOutOfScreen);
```

Lab4: Clase Java (clasă cu atribute și metode)

Clasa Platform:

```
package org.example;

import javafx.scene.canvas.GraphicsContext;
import javafx.scene.paint.Color;

public class Platform {
    private double x, y, width, height;
    private boolean outOfScreen = false;
    private boolean active = true;
    private PlatformType type;

    public Platform(double x, double y, double width, double height, PlatformType type) {
        this.x = x;
        this.y = y;
        this.width = width;
        this.height = height;
        this.type = type;
    }

    public void update(double offsetY) {
        y += offsetY;
        if (y > 600) {
            outOfScreen = true;
        }
    }

    public void render(GraphicsContext gc) {
        if (type == PlatformType.NORMAL) {
            gc.setFill(Color.GREEN);
        } else if (type == PlatformType.PURPLE) {
            gc.setFill(active ? Color.PURPLE : Color.GRAY);
        } else if (type == PlatformType.YELLOW) {
            gc.setFill(Color.YELLOW);
```

Lab5: Moștenire în Java, clase abstracte (să evidențiați relația dintre clase)

Clasa care mostenita de interfata:

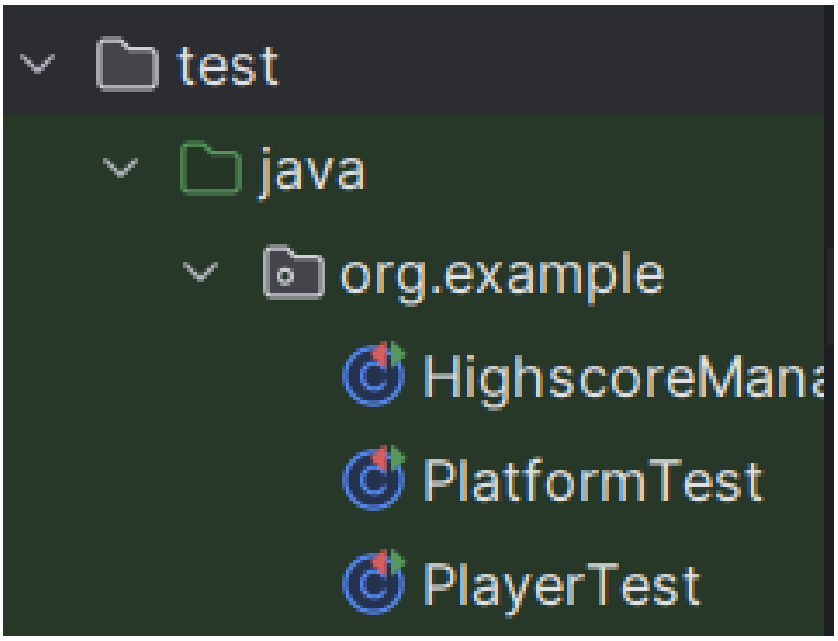
```java
package org.example;

import com.google.gson.Gson;

import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;

public class HighscoreManager implements IHighscoreManager {
    private static final String HIGHSCORE_FILE = "highscore.json";
    private int highscore;

    @Override
    public void loadHighscore() {
        try (FileReader reader = new FileReader(HIGHSCORE_FILE)) {
            Gson gson = new Gson();
            highscore = gson.fromJson(reader, Integer.class);
        } catch (IOException e) {
            highscore = 0;
        }
    }

    @Override
    public void saveHighscore() {
        try (FileWriter writer = new FileWriter(HIGHSCORE_FILE)) {
            Gson gson = new Gson();
            gson.toJson(highscore, writer);
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
```

Lab6: Interfețe în Java

Interfata:

```java
package org.example;

public interface IHighscoreManager {
    void loadHighscore();

    void saveHighscore();

    int getHighscore();

    void setHighscore(int score);
}
```

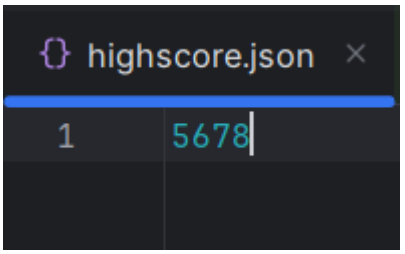
Lab7: Teste pentru fiecare metodă



```java
package org.example;

import static org.junit.jupiter.api.Assertions.*;
import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.api.Test;

class HighscoreManagerTest {

    private HighscoreManager manager;

    @BeforeEach
    void setUp() {
        manager = new HighscoreManager();
    }

    @Test
    void testInitialHighscore() {
        assertEquals( expected: 0, manager.getHighscore(), message: "Initial highscore should be 0");
    }

    @Test
    void testSetHighscore() {
        manager.setHighscore(100);
        assertEquals( expected: 100, manager.getHighscore(), message: "Highscore should be updated correctly");
    }

    @Test
    void testSaveAndLoadHighscore() {
        manager.setHighscore(150);
        manager.saveHighscore();
        HighscoreManager newManager = new HighscoreManager();
        newManager.loadHighscore();
        assertEquals( expected: 150, newManager.getHighscore(), message: "Highscore should persist after saving and loading");
    }
```

Lab8: Persistența datelor (salvare în fișiere .txt, .json sau alt format). Aplicația trebuie să încarce datele salvate anterior la repornire

```java
@Override
public void loadHighscore() {
    try (FileReader reader = new FileReader(HIGHSCORE_FILE)) {
        Gson gson = new Gson();
        highscore = gson.fromJson(reader, Integer.class);
    } catch (IOException e) {
        highscore = 0;
    }
}

@Override
public void saveHighscore() {
    try (FileWriter writer = new FileWriter(HIGHSCORE_FILE)) {
        Gson gson = new Gson();
        gson.toJson(highscore, writer);
    } catch (IOException e) {
        e.printStackTrace();
    }
}
```



Lab9: Realizați 3 diagrame UML

Diagrama de clase:

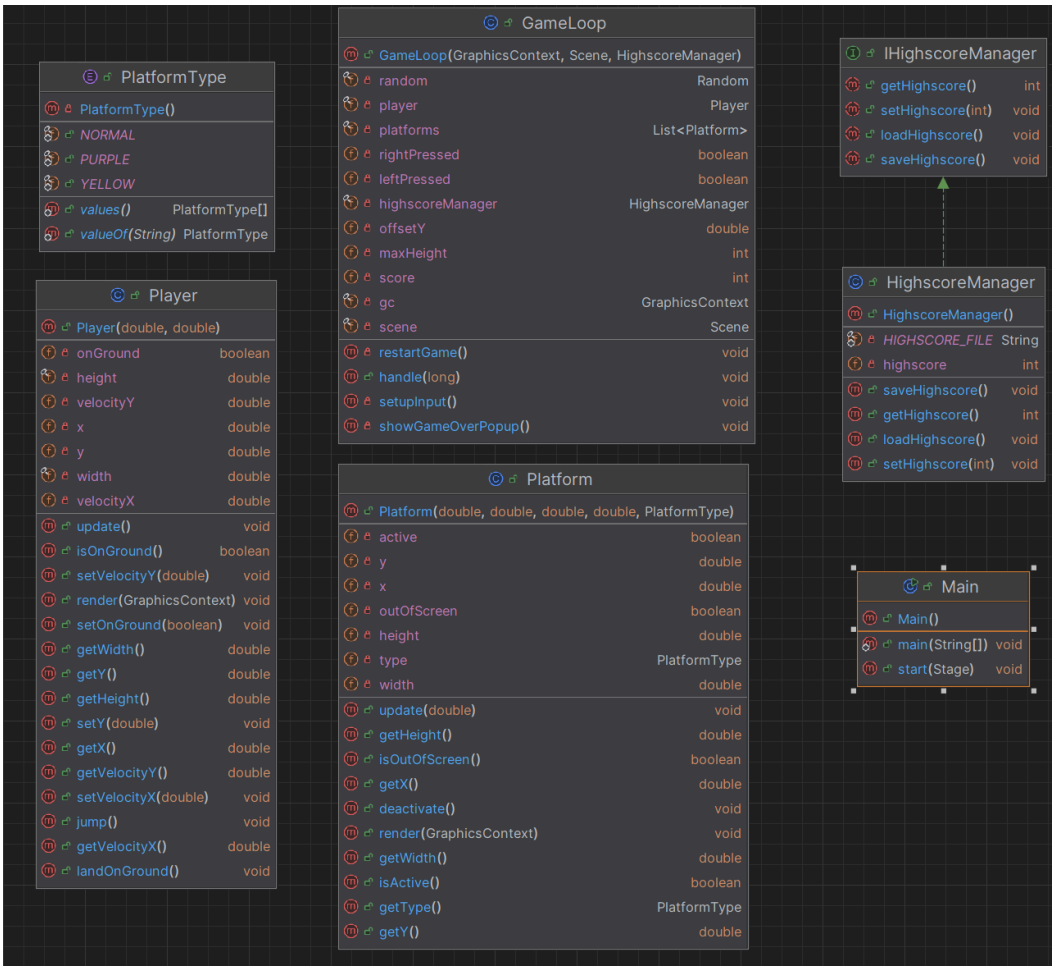


Diagrama de secventa:

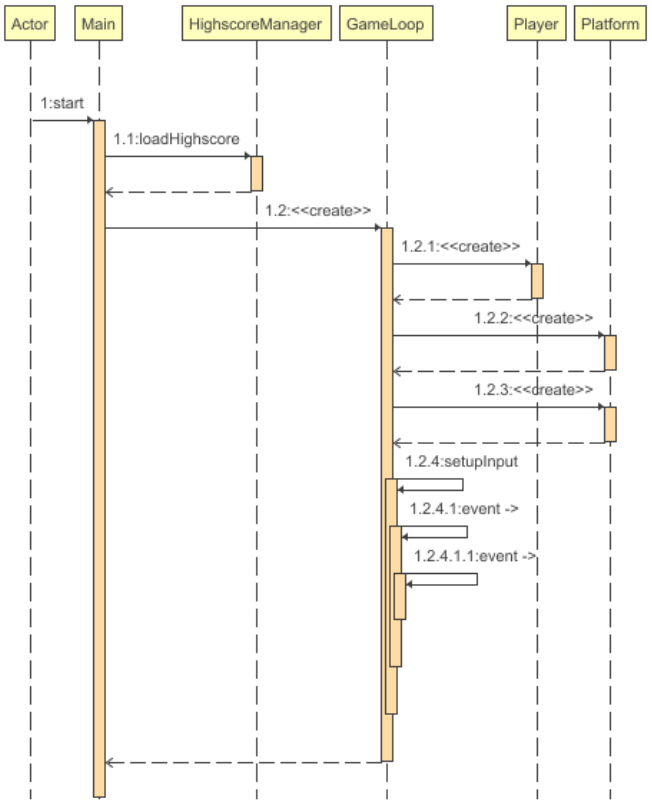
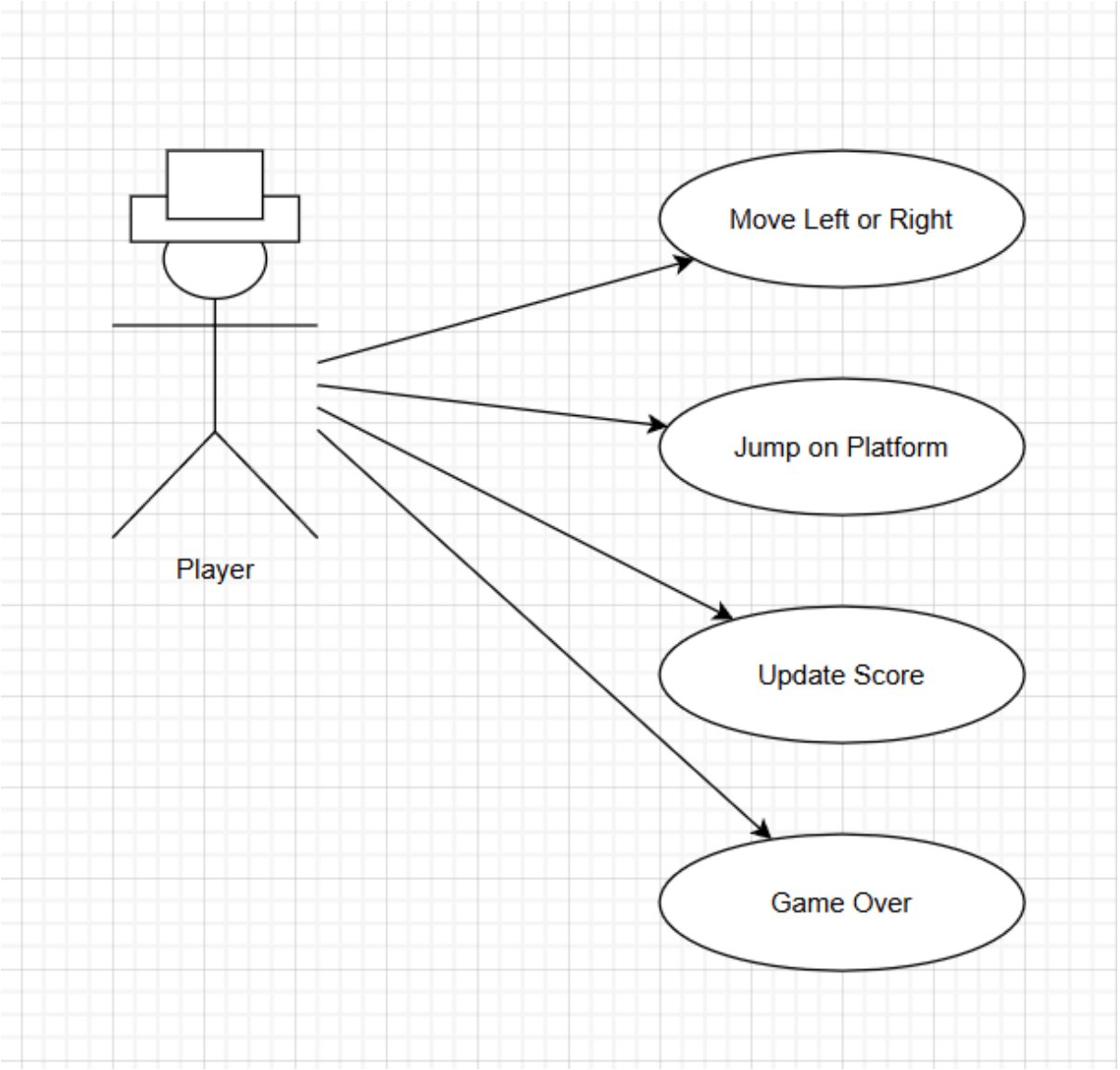


Diagrama de Usecase:


Move Left or Right
Jump on Platform
Update Score
Game Over
Player