Politechnika Wrocławska

Platformy programistyczne .Net i Java

Projekt własnej aplikacji w języku Java

Prowadzący: Mgr. inż. Michał Jaroszczuk

Grupa Środa, 18:55-20:40

Jan Klisowski 263485

18.06.2024

Spis treści

1. Opis Projektu

Celem projektu było stworzenie własnej aplikacji w języku Java. Ja wybrałem stworzenie klasycznej gry Snake w języku Java z użyciem biblioteki Swing do tworzenia interfejsu graficznego oraz z wielowątkowością dla równoległego i sprawnieszego wykonywania się procesów podczas gry. Gra polega na sterowaniu wężem, który porusza się po planszy, zbiera jedzenie, rośnie, a jednocześnie unika przeszkód oraz kolizji z samym sobą.

2. Struktura Projektu

Projekt składa się z następujących plików:

- `Main.java` - klasa główna uruchamiająca aplikację.
- `GamePanel.java` - panel gry zawierający główną logikę gry.
- `Food.java` - klasa reprezentująca jedzenie.
- `Obstacle.java` - klasa reprezentująca przeszkody.

3. Opis Klas i Metod

3.1. Main.java

```java
package org.example;

import javax.swing.JFrame;

public class Main extends JFrame {

    public Main() {
        this.add(new GamePanel());
        this.setTitle("Snake Game");
        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        this.setResizable(false);
        this.pack();
        this.setLocationRelativeTo(null);
        this.setVisible(true);
    }

    public static void main(String[] args) {
        new Main();
    }
}
```

Klasa `Main` dziedziczy po `JFrame`, co oznacza, że jest głównym oknem aplikacji. W konstruktorze dodawany jest panel gry (`GamePanel`), ustawiane są podstawowe właściwości okna, takie jak tytuł, rozmiar, zamykanie aplikacji po kliknięciu "X", itd.

3.2. GamePanel.java

Klasa `GamePanel` zawiera całą logikę gry. Zawiera metody odpowiedzialne za inicjalizację, rysowanie na ekranie, sprawdzanie kolizji, poruszanie się węża i obsługę zdarzeń. W metodzie `run` za pomocą `ExecutorService` uruchamiane są równolegle zadania dla jedzenia i przeszkód. Nie zamieściłem kodu w sprawozdaniu ze względu na to, że jest bardzo długi, ale znajduje się on w repozytorium.

3.3. Food.java

Listing 1: Food.java

```
package org.example;

import java.awt.*;
import java.util.Random;
import java.util.concurrent.TimeUnit;

public class Food implements Runnable {
    private int x;
    private int y;
    private final Color color;
    private final GamePanel panel;
    private static final Random random = new Random();

    public Food(GamePanel panel) {
        this.panel = panel;
        this.color = Color.red;
        this.spawn();
    }

    public void spawn() {
        this.x = random.nextInt((int) (panel.getScreenWidth() /
            panel.getUnitSize())) * panel.getUnitSize();
        this.y = random.nextInt((int) (panel.getScreenHeight() /
            panel.getUnitSize())) * panel.getUnitSize();
    }

    public void draw(Graphics g) {
        g.setColor(this.color);
        g.fillRect(this.x, this.y, panel.getUnitSize(), panel.getUnitSize());
    }

    public int getX() {
        return x;
    }
```

```
    public int getY() {
        return y;
    }

    @Override
    public void run() {
        while (true) {
            try {
                TimeUnit.MILLISECONDS.sleep(GamePanel.FOOD_MOVE_INTERVAL * 100);
                this.spawn();
            } catch (InterruptedException e) {
                Thread.currentThread().interrupt();
            }
        }
    }
}
```

Klasa `Food` reprezentuje jedzenie. Ma metody do losowego generowania pozycji, rysowania i przesuwania się na planszy. Implementuje interfejs `Runnable`, dzięki czemu może być uruchamiana jako wątek.

3.4. Obstacle.java

Listing 2: Obstacle.java

```
package org.example;

import java.awt.*;
import java.util.Random;
import java.util.concurrent.TimeUnit;

public class Obstacle implements Runnable {
    private int x;
    private int y;
    private final Color color;
    private final GamePanel panel;
    private static final Random random = new Random();

    public Obstacle(GamePanel panel) {
        this.panel = panel;
        this.color = Color.blue;
        this.spawn();
    }

    public void spawn() {
        this.x = random.nextInt((int) (panel.getScreenWidth() /
            panel.getUnitSize())) * panel.getUnitSize();
```

```
        this.y = random.nextInt((int) (panel.getScreenHeight() /
            panel.getUnitSize())) * panel.getUnitSize();
    }

    public void draw(Graphics g) {
        g.setColor(this.color);
        g.fillRect(this.x, this.y, panel.getUnitSize(), panel.getUnitSize());
    }

    public int getX() {
        return x;
    }

    public int getY() {
        return y;
    }

    @Override
    public void run() {
        while (true) {
            try {
                TimeUnit.MILLISECONDS.sleep(GamePanel.FOOD_MOVE_INTERVAL * 100);
                this.spawn();
            } catch (InterruptedException e) {
                Thread.currentThread().interrupt();
            }
        }
    }
}
```

Klasa `Obstacle` reprezentuje przeszkody w grze. Podobnie jak klasa `Food`, ma metody do losowego generowania pozycji, rysowania i przesuwania się na planszy. Implementuje interfejs `Runnable`.

4. Repozytorium

Link do repozytorium projektu: *https://github.com/jachoofrachoo/netjavaKlis*

5. Podsumowanie

Projekt gra Snake w języku Java z użyciem biblioteki Swing został zrealizowany zgodnie z założeniami. Wprowadzono podstawowe elementy gry takie jak jedzenie, przeszkody oraz kontrolę węża. Gra działa płynnie, a interfejs użytkownika jest prosty oraz wygodny w użyciu.