package luce;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.awt.event.KeyAdapter;

import java.awt.event.KeyEvent;

import java.util.ArrayList;

import java.util.Random;

public class SnakeGame extends JPanel implements ActionListener {

private final int WIDTH = 800;

private final int HEIGHT = 600;

private final int DOT\_SIZE = 20;

private final int ALL\_DOTS = (WIDTH \* HEIGHT) / (DOT\_SIZE \* DOT\_SIZE);

private final ArrayList<Point> snake = new ArrayList<>();

private Point food;

private char direction = 'R'; // Initial direction

private boolean running = false;

private Timer timer;

public SnakeGame() {

setPreferredSize(new Dimension(WIDTH, HEIGHT));

setBackground(Color.BLACK);

setFocusable(true);

addKeyListener(new KeyAdapter() {

@Override

public void keyPressed(KeyEvent e) {

switch (e.getKeyCode()) {

case KeyEvent.VK\_UP:

if (direction != 'D') direction = 'U';

break;

case KeyEvent.VK\_DOWN:

if (direction != 'U') direction = 'D';

break;

case KeyEvent.VK\_LEFT:

if (direction != 'R') direction = 'L';

break;

case KeyEvent.VK\_RIGHT:

if (direction != 'L') direction = 'R';

break;

}

}

});

startGame();

}

private void startGame() {

running = true;

snake.clear();

snake.add(new Point(5, 5)); // Start position

spawnFood();

timer = new Timer(100, this);

timer.start();

}

private void spawnFood() {

Random rand = new Random();

int x = rand.nextInt(WIDTH / DOT\_SIZE);

int y = rand.nextInt(HEIGHT / DOT\_SIZE);

food = new Point(x, y);

}

@Override

protected void paintComponent(Graphics g) {

super.paintComponent(g);

if (running) {

g.setColor(Color.RED);

g.fillRect(food.x \* DOT\_SIZE, food.y \* DOT\_SIZE, DOT\_SIZE, DOT\_SIZE);

g.setColor(Color.GREEN);

for (Point p : snake) {

g.fillRect(p.x \* DOT\_SIZE, p.y \* DOT\_SIZE, DOT\_SIZE, DOT\_SIZE);

}

g.setColor(Color.WHITE);

g.drawString("Score: " + (snake.size() - 1), 10, 20);

} else {

showGameOver(g);

}

}

private void showGameOver(Graphics g) {

g.setColor(Color.WHITE);

g.drawString("Game Over!", WIDTH / 2 - 30, HEIGHT / 2);

g.drawString("Score: " + (snake.size() - 1), WIDTH / 2 - 30, HEIGHT / 2 + 20);

}

@Override

public void actionPerformed(ActionEvent e) {

if (running) {

move();

checkCollision();

checkFood();

repaint();

}

}

private void move() {

Point head = snake.get(0);

Point newHead = new Point(head);

switch (direction) {

case 'U': newHead.y--; break;

case 'D': newHead.y++; break;

case 'L': newHead.x--; break;

case 'R': newHead.x++; break;

}

snake.add(0, newHead); // Add new head

snake.remove(snake.size() - 1); // Remove tail

}

private void checkCollision() {

Point head = snake.get(0);

// Check wall collision

if (head.x < 0 || head.x >= WIDTH / DOT\_SIZE || head.y < 0 || head.y >= HEIGHT / DOT\_SIZE) {

running = false;

timer.stop();

}

// Check self-collision

for (int i = 1; i < snake.size(); i++) {

if (head.equals(snake.get(i))) {

running = false;

timer.stop();

break;

}

}

}

private void checkFood() {

Point head = snake.get(0);

if (head.equals(food)) {

snake.add(new Point(-1, -1)); // Grow the snake

spawnFood(); // Spawn new food

}

}

public static void main(String[] args) {

JFrame frame = new JFrame("Snake Game");

SnakeGame gamePanel = new SnakeGame();

frame.add(gamePanel);

frame.pack();

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

frame.setVisible(true);

frame.setResizable(false);

}

}