МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО”

Факультет прикладної математики
Кафедра програмного забезпечення комп’ютерних систем

Лабораторна робота №2

з дисципліни “Математичні та алгоритмічні основи комп’ютерної графіки”

тема: “Побудова та анімація зображень за допомогою Java2D”

Виконав
студент III курсу
групи КП-83
Ландо Максим Юрійович
варіант № 9

Зарахована
“____” “_____” 20___ р.
викладачем
Шкурат Оксаною Сергіївною

Київ 2021

Мета: Ознайомитися з можливостями побудови зображень та їх анімації у Java2D

Задання на лабораторну роботу

За допомогою Java 2D намалювати картинку з лабораторної роботи №1 (за варіантом).

Додатково виконати:

1. Хоча б 1 стандартний примітив, та хоча б 1 фігуру, побудовану по точкам (ламаною).
2. Хоча б 1 фігуру залити градієнтною фарбою за вибором (в цьому випадку колір може не співпадати з варіантом із лабораторної роботи № 1).
3. На достатній відстані від побудованого малюнку намалювати прямокутну рамку, всередині якої відбуватиметься анімація. Тип лінії рамки задано за варіантом.
4. Виконати анімацію малюнку, за варіантом. При цьому рамка повинна залишатися статичною. Взаємодія з рамкою не обов’язкова, якщо не передбачено варіантом.

Варіант типу анімації:

Рух по квадрату проти годинникової стрілки

Обертання навколо кута малюнка за годинниковою стрілкою

Варіант типу ліній рамки:

JOIN_ROUND

Лістинг коду програми

Main.java

```
package sample;

import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.geom.GeneralPath;

import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.Timer;

@SuppressWarnings("serial")
public class Main extends JPanel implements ActionListener{

    Timer timer;

    private static int maxWidth;
    private static int maxHeight;

    int rectStartPoints[] = {600,100};
    static final int rectWidth = 300;
    static final int rectHeight = 300;

    private double angle = 0;

    private int ovalR = 50;
    private int dx = 0;
    private int tx = 0;
    private int dy = 1;
    private int ty = 1;

    public Main() {
        timer = new Timer(10, this);
        timer.start();
    }

    public void paint(Graphics g) {
        Graphics2D g2d = (Graphics2D) g;
        g2d.setBackground(Color.green);
        g2d.clearRect(0, 0, maxWidth, maxHeight);

        Double[] big_star_points = new Double[10];
        Double[] small_star_points = new Double[10];;

        double big_star_r = 200;
        double small_star_r = 75;

        double center_x = 325;
        double center_y = 300;

        for (int i=0; i<5; i++) {
            big_star_points[i*2] = center_x + big_star_r*Math.cos(Math.PI*(1.0/2.0
+ 2.0*i/5.0));
            big_star_points[i*2+1] = center_y - big_star_r*Math.sin(Math.PI*(1.0/2.
0 + 2.0*i/5.0));
            small_star_points[i*2] = center_x + small_star_r*Math.cos(Math.PI*(-
1.0/2.0 + 2.0*(i+2.0)/5.0));
```

```
            small_star_points[i*2+1] = center_y - small_star_r*Math.sin(Math.PI*(-
1.0/2.0 + 2.0*(i+2.0)/5.0));
        }
        BasicStroke bs3 = new BasicStroke(16, BasicStroke.CAP_SQUARE,
                BasicStroke.JOIN_BEVEL);
        g2d.setStroke(bs3);
        g2d.setColor(new Color(150,150,150));
        g2d.drawLine(
                small_star_points[6].intValue(),
                small_star_points[7].intValue(),
                small_star_points[6].intValue(),
                small_star_points[7].intValue()+200
        );
        g2d.setStroke(new BasicStroke(0));
        GradientPaint gp = new GradientPaint(
                5, 25,
                new Color(255,255,0),
                20, 2,
                new Color(0,0,255), true);
        g2d.setPaint(gp);

        GeneralPath star = new GeneralPath();
        star.moveTo(small_star_points[0], small_star_points[1]);
        for (int k = 1; k < 5; k++)
            star.lineTo(small_star_points[2*k], small_star_points[2*k+1]);
        star.closePath();
        g2d.fill(star);
        g2d.draw(star);
        for (int i=0; i<5;i++){
            GeneralPath star_part = new GeneralPath();
            star_part.moveTo(small_star_points[2*i], small_star_points[2*i+1]);
            star_part.lineTo(big_star_points[2*i], big_star_points[2*i+1]);
            star_part.lineTo(small_star_points[((i+1)*2) % 10], small_star_points[(
(i+1)*2+1) % 10]);
            star_part.closePath();
            g2d.fill(star_part);
            g2d.draw(star_part);
        }
        BasicStroke bs4 = new BasicStroke(4, BasicStroke.CAP_ROUND,
                BasicStroke.JOIN_ROUND);
        g2d.setStroke(bs4);

        g2d.drawRect(rectStartPoints[0], rectStartPoints[1], rectWidth, rectHeight)
;
        g2d.translate(tx-ovalR/2, ty-ovalR/2);
        g2d.fillOval(rectStartPoints[0], rectStartPoints[1], ovalR, ovalR);
        g2d.translate(-tx+ovalR/2, -ty+ovalR/2);
        g2d.rotate(angle, rectStartPoints[0],
                rectStartPoints[1]);
        g2d.fillOval(rectStartPoints[0]+36, rectStartPoints[1], ovalR, ovalR);
    }
    public static void main(String[] args) {
        JFrame frame = new JFrame("lab2");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(1000, 600);
        frame.setLocationRelativeTo(null);
        frame.setResizable(false);
        frame.add(new Main());
        frame.setVisible(true);
        Dimension size = frame.getSize();
        Insets insets = frame.getInsets();
        maxWidth = size.width - insets.left - insets.right - 1;
```

```
        maxHeight = size.height - insets.top - insets.bottom - 1;
    }

    public void actionPerformed(ActionEvent e) {
        if (tx >= rectWidth && ty <= 0){
            dx=-1;
            dy=0;
        }
        if (ty>=rectHeight && tx >= rectWidth){
            dx=0;
            dy=-1;
        }
        if (tx<=0 && ty >=rectHeight){
            dx=1;
            dy=0;
        }
        if (ty<=0 && tx <=0){
            dx=0;
            dy=1;
        }
        ty+=dy;
        tx+=dx;
        repaint();
        angle+=0.01;
    }
}
```

Результат