

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 №** **3**

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

|  |  |  |  |
| --- | --- | --- | --- |
| Виконав  студент III курсу  групи КП-02  Товстига Максим  (*прізвище, ім’я, по батькові*)  варіант № 14 |  |  | Зарахована  “\_\_\_\_” “\_\_\_\_\_\_\_\_\_\_\_\_” 20\_\_\_ р.  викладачем  Шкурат Оксаною Сергіївною (*прізвище, ім’я, по батькові*) |

Київ 2023

**Варіант завдання**

**Завдання**: За допомогою примітивів JavaFX максимально реально зобразити персонажа за варіантом та виконати його 2D анімацію. Для анімації скористатися стандартними засобами бібліотеки JavaFX. Обов’язковою є реалізація таких видів анімації:

1) переміщення;

2) поворот;

3) масштабування.

Студентам пропонується скористатися розглянутими класами для читання, обробки та збереження зображень формату .bmp з метою використання рисунку для створення траєкторії руху або меж, в яких дозволений рух об’єктів. В даному випадку рекомендується використовувати кольори великої контрастності для різних призначень (наприклад, чорний колір відповідатиме за траєкторію руху, а інші кольори – заборонятимуть рух).

**Варіант:**



**Лістинг коду програми**

package lab3;

import javafx.animation.\*;

import javafx.application.Application;

import javafx.scene.Group;

import javafx.scene.Scene;

import javafx.scene.paint.Color;

import javafx.scene.shape.\*;

import javafx.stage.Stage;

import java.awt.geom.RoundRectangle2D;

import javafx.util.Duration;

public class Umbrella extends Application {

@Override

public void start(Stage primaryStage){

Group root = new Group();

Scene scene = new Scene(root, 1000, 500);

Ellipse ellipse1 = new Ellipse(228, 167, 80.5, 80.5);

ellipse1.setFill(Color.rgb(192,98,133));

root.getChildren().add(ellipse1);

Ellipse ellipse2 = new Ellipse(177, 187, 44, 35);

ellipse2.setFill(Color.WHITE);

root.getChildren().add(ellipse2);

Ellipse ellipse3 = new Ellipse(231, 178, 55, 35);

ellipse3.setFill(Color.WHITE);

root.getChildren().add(ellipse3);

Ellipse ellipse4 = new Ellipse(277, 170, 38, 30);

ellipse4.setFill(Color.WHITE);

root.getChildren().add(ellipse4);

Ellipse ellipse5 = new Ellipse(231, 218, 85, 70);

ellipse5.setFill(Color.WHITE);

root.getChildren().add(ellipse5);

MoveTo mt3 = new MoveTo(220, 120);

LineTo lt2 = new LineTo(235, 220);

QuadCurveTo qt8 = new QuadCurveTo(240, 226, 246, 220);

LineTo lt3 = new LineTo(245, 212);

Path handle = new Path();

handle.setStrokeWidth(2);

handle.setStroke(Color.rgb(213, 214, 204));

handle.getElements().addAll(mt3, lt2, qt8, lt3);

root.getChildren().add(handle);

MoveTo mt1 = new MoveTo(146, 161.5);

QuadCurveTo qt1 = new QuadCurveTo(160, 100, 211, 85);

QuadCurveTo qt2 = new QuadCurveTo(277, 119, 250, 135);

QuadCurveTo qt3 = new QuadCurveTo(217, 117, 190, 143);

QuadCurveTo qt4 = new QuadCurveTo(170, 130, 146, 163);

QuadCurveTo qt5 = new QuadCurveTo(280, 75, 307, 150);

Path top = new Path();

top.setStrokeWidth(2);

top.setStroke(Color.rgb(213, 214, 204));

top.setFill(Color.rgb(121, 30, 62));

top.getElements().addAll(mt1, qt1, qt5, qt2, qt3, qt4);

root.getChildren().add(top);

MoveTo mt2 = new MoveTo(215, 86);

LineTo lt1 = new LineTo(215, 70);

Path pin = new Path();

pin.setStrokeWidth(2);

pin.setStroke(Color.rgb(213, 214, 204));

pin.getElements().addAll(mt2, lt1);

root.getChildren().add(pin);

QuadCurveTo qt6 = new QuadCurveTo(185, 110, 190, 142);

QuadCurveTo qt7 = new QuadCurveTo(250, 100, 250, 134);

Path lines = new Path();

lines.setStrokeWidth(2);

lines.setStroke(Color.rgb(213, 214, 204));

lines.getElements().addAll(mt2, qt6, mt2, qt7);

root.getChildren().add(lines);

int cycleCount = 2;

int time = 3000;

TranslateTransition translateTransition = new TranslateTransition(Duration.millis(time), root);

translateTransition.setFromY(50);

translateTransition.setToY(200);

translateTransition.setFromX(100);

translateTransition.setToX(400);

translateTransition.setCycleCount(cycleCount + 1);

translateTransition.setAutoReverse(true);

ScaleTransition scaleTransition = new ScaleTransition(Duration.millis(time), root);

scaleTransition.setToX(3);

scaleTransition.setToY(3);

scaleTransition.setAutoReverse(true);

RotateTransition rotateTransition = new RotateTransition(Duration.millis(time), root);

rotateTransition.setByAngle(180f);

rotateTransition.setCycleCount(cycleCount);

rotateTransition.setAutoReverse(true);

ScaleTransition scaleTransition2 = new ScaleTransition(Duration.millis(time), root);

scaleTransition2.setToX(0.3);

scaleTransition2.setToY(0.3);

scaleTransition2.setCycleCount(cycleCount);

scaleTransition2.setAutoReverse(true);

ParallelTransition parallelTransition = new ParallelTransition();

parallelTransition.getChildren().addAll(

translateTransition,

scaleTransition,

rotateTransition,

scaleTransition2

);

parallelTransition.setCycleCount(Timeline.INDEFINITE);

parallelTransition.play();

primaryStage.setResizable(false);

primaryStage.setTitle("lab3");

primaryStage.setScene(scene);

primaryStage.show();

}

public static void main(String[] args) {

launch(args);

}

}

**Результат**



