

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-83  Клименко Ярослав  (*прізвище, ім’я, по батькові*)  варіант № 7 |  |  | Зарахована  “\_\_\_\_” “\_\_\_\_\_\_\_\_\_\_\_\_” 20\_\_\_ р.  викладачем  Шкурат Оксаною Сергіївною (*прізвище, ім’я, по батькові*) |

Київ 2020

**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 7:**

* 

**Лістинг коду програми**

package sample;

import javafx.application.Application;

import javafx.scene.Group;

import javafx.scene.Scene;

import javafx.scene.paint.Color;

import javafx.scene.shape.Circle;

import javafx.scene.shape.Polygon;

import javafx.stage.Stage;

import javafx.scene.paint.Stop;

import javafx.animation.\*;

import javafx.util.Duration;

import javafx.scene.shape.\*;

import javafx.scene.transform.Rotate;

import javafx.scene.paint.CycleMethod;

import javafx.scene.paint.LinearGradient;

public class Main extends Application {

public static void main(String args[]) {

launch(args);

}

@Override

public void start(Stage primaryStage) {

Group root = new Group();

Scene scene = new Scene(root, 470, 310);

var spearColor = Color.rgb(255,241,164);

// ARROW

var colorDarker =Color.rgb(150, 90, 36);

var colorLighter = Color.rgb(253, 242, 171);

var colorMedium = Color.rgb(217,151,79);

Stop[] stops = new Stop[] { new Stop(0, colorDarker), new Stop(0.25, colorLighter), new Stop(0.5, colorDarker) ,new Stop(0.75, colorMedium), new Stop(1, colorDarker)};

LinearGradient arrowColor = new LinearGradient(0, 0, 1, 0, true, CycleMethod.NO\_CYCLE, stops);

Line arrowTop = new Line();

arrowTop.setStartX(280.0);

arrowTop.setStartY(90.0);

arrowTop.setEndX(150.0);

arrowTop.setEndY(200);

arrowTop.setStroke(spearColor);

arrowTop.setStrokeWidth(5.0);

root.getChildren().add(arrowTop);

// FEATHER 1

Polygon featherOne = new Polygon();

featherOne.getPoints().addAll(new Double[]{

280.0, 88.0,

270.0, 95.0,

273.0, 85.0,

280.0, 78.0,

});

featherOne.setFill(arrowColor);

root.getChildren().add(featherOne);

// FEATHER 2

Polygon featherTwo = new Polygon();

featherTwo.getPoints().addAll(new Double[]{

283.0, 90.0,

270.0, 100.0,

283.0, 98.0,

290.0, 93.0,

});

featherTwo.setFill(arrowColor);

root.getChildren().add(featherTwo);

// spear

Polygon spear = new Polygon();

spear.setFill(arrowColor);

spear.getPoints().addAll(new Double[]{

140.0, 207.0,

150.0, 192.0,

160.0, 203.0,

});

root.getChildren().add(spear);

// spear circle 1

Circle circle1 = new Circle(152, 195, 3);

circle1.setFill(arrowColor);

root.getChildren().add(circle1);

// spear circle 2

Circle circle2 = new Circle(157, 200, 3);

circle2.setFill(arrowColor);

root.getChildren().add(circle2);

// serpentine Adjusters

Polygon adjusterOne = new Polygon();

adjusterOne.getPoints().addAll(new Double[]{

158.0, 122.0,

170.0, 118.0,

160.0, 135.0

});

adjusterOne.setFill(colorDarker);

root.getChildren().add(adjusterOne);

Polygon adjusterTwo = new Polygon();

adjusterTwo.getPoints().addAll(new Double[]{

273.0, 168.0,

240.0, 170.0,

270.0, 150.0

});

adjusterTwo.setFill(colorDarker);

root.getChildren().add(adjusterTwo);

// heart 1

Ellipse ellipse = new Ellipse();

ellipse.setCenterX(225.0f);

ellipse.setCenterY(150.0f);

ellipse.setRadiusX(35.0f);

ellipse.setRadiusY(50.0f);

ellipse.setFill(Color.rgb(218, 26, 32));

root.getChildren().add(ellipse);

Rotate rotate = new Rotate();

rotate.setPivotX(225.0f);

rotate.setPivotY(150.0f);

rotate.setAngle(30);

ellipse.getTransforms().add(rotate);

// heart 2

Ellipse ellipse2 = new Ellipse();

ellipse2.setCenterX(205.0f);

ellipse2.setCenterY(140.0f);

ellipse2.setRadiusX(35.0f);

ellipse2.setRadiusY(50.0f);

ellipse2.setFill(Color.rgb(218, 26, 32));

root.getChildren().add(ellipse2);

Rotate rotate2 = new Rotate();

rotate2.setPivotX(225.0f);

rotate2.setPivotY(150.0f);

rotate2.setAngle(-30);

ellipse2.getTransforms().add(rotate2);

// Serpentine

Polygon serpentine = new Polygon();

serpentine.getPoints().addAll(new Double[]{

160.0, 125.0,

190.0, 135.0,

230.0, 125.0,

270.0, 135.0,

270.0, 165.0,

230.0, 155.0,

190.0, 165.0,

160.0, 155.0,

});

serpentine.setStrokeWidth(5);

serpentine.setStroke(arrowColor);

serpentine.setStrokeLineJoin(StrokeLineJoin.ROUND);

serpentine.setFill(arrowColor);

root.getChildren().add(serpentine);

// Animation

int cycleCount = 2;

int time = 1000;

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

scaleTransition.setToX(2);

scaleTransition.setToY(2);

scaleTransition.setCycleCount(cycleCount);

scaleTransition.setAutoReverse(true);

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

rotateTransition.setByAngle(-360f);

rotateTransition.setCycleCount(cycleCount);

rotateTransition.setAutoReverse(true);

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

translateTransition.setFromX(150);

translateTransition.setToX(50);

translateTransition.setCycleCount(cycleCount);

translateTransition.setAutoReverse(true);

ParallelTransition parallelTransition = new ParallelTransition();

parallelTransition.getChildren().addAll(

rotateTransition,

scaleTransition,

translateTransition

);

parallelTransition.setCycleCount(Timeline.INDEFINITE);

parallelTransition.play();

primaryStage.setTitle("lab3");

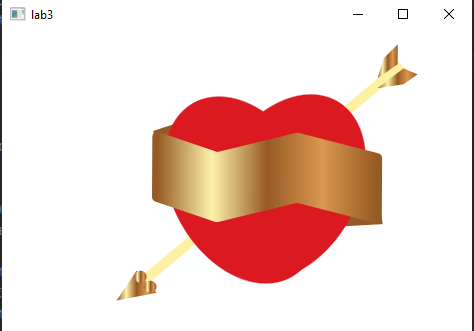
primaryStage.setScene(scene);

primaryStage.show();

}

}

**Результат**

****