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 № 5

з дисципліни “Імпорт тривимірних моделей у середовище програмування java 3D, обробка та маніпуляція цих зображень.”

Виконав
студент III курсу
групи КП-82
Шило Андрій Володимирович
(*прізвище, ім’я, по батькові*)

Зарахована
“____” “_____” 20___ р.
викладачем
Шкурат Оксаною Сергіївною
(*прізвище, ім’я, по батькові*)

варіант № 20

Київ 2021

Варіант завдання

Завдання: Імпортувати моделі тривимірних об’єктів форматів, що визначені варіантом. Створити реалістичну анімацію об’єкту. Додати до сцени фон, інші об’єкти для надання сцені реалістичного вигляду. Для цього використати текстури, матеріали, імпортувати додаткові об’єкти з відкритих бібліотек, за бажанням створити прості об’єкти у графічному редакторі.Студенти, які мають непарний номер варіанту списку групи імпортують моделі формату .obj, парний варіант – .lwo.

Варіант: 20

літак

Лістинг коду програми

```
package sample;

import com.sun.j3d.loaders.Scene;
import com.sun.j3d.loaders.objectfile.ObjectFile;
import com.sun.j3d.utils.image.TextureLoader;
import com.sun.j3d.utils.universe.SimpleUniverse;

import javax.media.j3d.*;
import javax.swing.*;
import javax.vecmath.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.KeyEvent;
import java.awt.event.KeyListener;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.util.Map;

public class Main extends JFrame implements ActionListener, KeyListener {
	private final static String planeModelLocation = "plane.obj";
	private final static String backgroundLocation = "sky.jpg";
	private final BranchGroup root = new BranchGroup();
	private final Canvas3D canvas = new
Canvas3D(SimpleUniverse.getPreferredConfiguration());
	private final TransformGroup planeGroup = new TransformGroup();
	private final Transform3D transform3D = new Transform3D();
	private final Transform3D rotateTransformX = new Transform3D();
	private final Transform3D rotateTransformY = new Transform3D();
	private final Transform3D rotateTransformZ = new Transform3D();
	private final ClassLoader classLoader =
Thread.currentThread().getContextClassLoader();
	private SimpleUniverse universe;
	private Scene plane;
	private Background background;
	private Map<String, Shape3D> nameMap;

	private final float z_location_current = -7;
	private double angle = -5;


	public static void main(String[] args) {
	try {
		var window = new Main();
		window.addKeyListener(window);
		window.setVisible(true);
	} catch (IOException e) {
		System.err.println(e.getMessage());
	}
	}

	public Main() throws IOException {
	initialize();
	addTexture();
	addAppearance();
	addImageBackground();
	addLight();
	setInitialLocation();
	setInitialViewAngle();
	Timer timer = new Timer(100, this);
	timer.start();
	root.compile();
	universe.addBranchGraph(root);
	}

	private void setInitialLocation() {

	transform3D.setTranslation(new Vector3f(0, 0, z_location_current));
```

```
float scale_cur = 1;
transform3D.setScale(scale_cur);
rotateTransformY.rotY(-Math.PI/2);
rotateTransformX.rotX(Math.PI/2);
transform3D.mul(rotateTransformY, rotateTransformX);
planeGroup.setTransform(transform3D);
}

private void initialize() throws IOException {
// window settings
setTitle("Lab #5");
setSize(1000, 1000);
setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

canvas.setDoubleBufferEnable(true);
getContentPane().add(canvas, BorderLayout.CENTER);

universe = new SimpleUniverse(canvas);
universe.getViewingPlatform().setNominalViewingTransform();

canvas.addKeyListener(this);
plane = getSceneFromFile();
}

private void addLight() {
var dirLight = new DirectionalLight(
       new Color3f(Color.WHITE),
       new Vector3f(4.0f, -7.0f, -12.0f)
);

dirLight.setInfluencingBounds(new BoundingSphere(new Point3d(), 1000));
root.addChild(dirLight);

var ambientLight = new AmbientLight(new Color3f(Color.WHITE));
var directionalLight = new DirectionalLight(
       new Color3f(Color.BLACK),
       new Vector3f(-1F, -1F, -1F)
);
var influenceRegion = new BoundingSphere(new Point3d(), 1000);
ambientLight.setInfluencingBounds(influenceRegion);
directionalLight.setInfluencingBounds(influenceRegion);
root.addChild(ambientLight);
root.addChild(directionalLight);
}

private TextureLoader getTextureLoader(String path) throws IOException {
var textureResource = classLoader.getResource(path);
if (textureResource == null) {
       throw new IOException("Couldn't find texture: " + path);
}
return new TextureLoader(textureResource.getPath(), canvas);
}

private Material getMaterial() {
var material = new Material();
material.setAmbientColor(new Color3f(new Color(243, 242, 221)));
material.setDiffuseColor(new Color3f(new Color(255, 233, 207)));
material.setSpecularColor(new Color3f(new Color(255, 203, 195)));
material.setLightingEnable(true);
return material;
}

private void addTexture() throws IOException {
nameMap = plane.getNamedObjects();
planeGroup.addChild(plane.getSceneGroup());
planeGroup.setCapability(TransformGroup.ALLOW_TRANSFORM_WRITE);
root.addChild(planeGroup);

}

private void addAppearance() throws IOException {
```

```
        Appearance rexAppearance = new Appearance();
        rexAppearance.setTexture(getTextureLoader("texture.png").getTexture());
        TextureAttributes texAttr = new TextureAttributes();
        texAttr.setTextureMode(TextureAttributes.COMBINE);
        rexAppearance.setTextureAttributes(texAttr);
        rexAppearance.setMaterial(getMaterial());
        Shape3D rex = nameMap.get("default");
        rex.setAppearance(rexAppearance);
        }

        private void addImageBackground() throws IOException {
        background = new Background(getTextureLoader(backgroundLocation).getImage());
        background.setImageScaleMode(Background.SCALE_FIT_MAX);
        background.setApplicationBounds(new BoundingSphere(new Point3d(),1000));
        background.setCapability(Background.ALLOW_IMAGE_WRITE);
        root.addChild(background);
        }

        private void setInitialViewAngle() {
        Transform3D lookAt = new Transform3D();
        lookAt.lookAt(new Point3d(0.0, 0.0, 3.0), new Point3d(0.0, 0.0, 0.0), new
Vector3d(1.0, 0.0, 0.0));
        lookAt.invert();
        universe.getViewingPlatform().getViewPlatformTransform().setTransform(lookAt);
        }

        private Scene getSceneFromFile() throws IOException {
        ObjectFile file = new ObjectFile(ObjectFile.RESIZE);
        file.setFlags(ObjectFile.RESIZE | ObjectFile.TRIANGULATE |
ObjectFile.STRIPIFY);
        var inputStream = classLoader.getResourceAsStream(planeModelLocation);
        if (inputStream == null) {
                throw new IOException("Resource " + planeModelLocation + " not found");
        }
        return file.load(new BufferedReader(new InputStreamReader(inputStream)));
        }

        @Override
        public void actionPerformed(ActionEvent e) {
        angle += .1;
        float x = (float) (2*(1-Math.pow(angle, 2))/(1+Math.pow(angle, 2)));
        float y = (float) (4*angle*(1-Math.pow(angle, 2))/(1+Math.pow(angle, 2)));
        if (x > 0) {
                rotateTransformX.rotX(0.25);
        } else {
                rotateTransformX.rotX(0.1);
        }
        transform3D.setTranslation(new Vector3f(x, y, z_location_current));
        transform3D.mul(rotateTransformX);
        transform3D.mul(rotateTransformZ);
        planeGroup.setTransform(transform3D);
        }

        @Override
        public void keyPressed(KeyEvent e) {  }

        @Override
        public void keyReleased(KeyEvent e) { }

        @Override
        public void keyTyped(KeyEvent e) { }
}
```

Результат