

МIНIСТЕРСТВО ОСВIТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені Ігоря Сікорського”

Факультет прикладної математики

Кафедра програмного забезпечення комп’ютерних систем

**Лабораторна робота № 4**

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

|  |  |
| --- | --- |
| Виконав  студент III курсу  групи КП-73  Булаєвський Ігор Олегович  (*прізвище, ім’я, по батькові*)  Варіант №3 |  |

Київ - 2020Лістинг програми

**PingPongTable.java**

import com.sun.j3d.utils.applet.MainFrame;

import com.sun.j3d.utils.geometry.Box;

import com.sun.j3d.utils.geometry.Cylinder;

import com.sun.j3d.utils.geometry.Sphere;

import com.sun.j3d.utils.image.TextureLoader;

import com.sun.j3d.utils.universe.SimpleUniverse;

import javax.media.j3d.\*;

import javax.swing.\*;

import javax.swing.event.MouseInputListener;

import javax.vecmath.\*;

import java.applet.Applet;

import java.awt.\*;

import java.awt.event.\*;

public class PingPongTable extends Applet implements ActionListener, MouseInputListener, MouseWheelListener {

private TransformGroup objTrans;

private TransformGroup ballGroup;

private float z\_ball = 0.2f;

private float z\_diff = -0.005f;

BranchGroup objRoot;

private Transform3D trans = new Transform3D();

private Point2d mouseLastPosition = null;

private double angleX = 0;

private double angleY = 0;

private double scale = 1;

public static void main(String[] args) {

PingPongTable bb = new PingPongTable();

bb.addMouseMotionListener(bb);

bb.addMouseListener(bb);

bb.addMouseWheelListener(bb);

MainFrame mf = new MainFrame(bb, 800, 800);

System.out.println("Program Started");

}

Appearance createAppearance(Color3f color) {

Appearance ap = new Appearance();

ap.setColoringAttributes(new ColoringAttributes(color, 1));

return ap;

}

Appearance loadTexture(String path) {

Texture tex = new TextureLoader(path, this).getTexture();

tex.setBoundaryModeS(Texture.WRAP);

tex.setBoundaryModeT(Texture.WRAP);

TextureAttributes texAttr = new TextureAttributes();

texAttr.setTextureMode(TextureAttributes.DECAL);

Appearance ap = new Appearance();

ap.setTexture(tex);

ap.setTextureAttributes(texAttr);

Material material = new Material();

material.setSpecularColor(new Color3f(Color.WHITE));

material.setDiffuseColor(new Color3f(Color.WHITE));

ap.setMaterial(material);

return ap;

}

BranchGroup createTable() {

BranchGroup root = new BranchGroup();

Box table = new Box(0.5f, 0.2f, 0.02f, loadTexture("wood.jpg"));

Box net = new Box(0.007f, 0.19f, 0.05f, createAppearance(new Color3f(0f, 1f, 1f)));

moveObject(net, 0f, 0f, 0.035f);

root.addChild(table);

createFoot(1f, 1f);

createFoot(-1f, 1f);

createFoot(1f, -1f);

createFoot(-1f, -1f);

createBall();

return root;

}

void createBall() {

ballGroup = new TransformGroup();

ballGroup.setCapability(TransformGroup.ALLOW\_TRANSFORM\_WRITE);

Sphere ball = new Sphere(.01f, createAppearance(new Color3f(1f, 0f, 0f)));

ballGroup.addChild(ball);

moveObject(ballGroup, 0.4f, 0.15f, 0.4f);

}

void createFoot(float i1, float i2) {

Cylinder foot = new Cylinder(0.007f, 0.25f, createAppearance(new Color3f(0.5f, 1f, 0f)));

TransformGroup tg = new TransformGroup();

Transform3D transform = new Transform3D();

transform.rotX(Math.PI / 2);

tg.setTransform(transform);

tg.addChild(foot);

moveObject(tg, 0.45f \* i1, 0.18f \* i2, -0.12f);

}

public BranchGroup createSceneGraph() {

objRoot = new BranchGroup();

objTrans = new TransformGroup();

BranchGroup shape = createTable();

objTrans.setCapability(TransformGroup.ALLOW\_TRANSFORM\_WRITE);

objTrans.addChild(shape);

objRoot.addChild(objTrans);

Color3f light1Color = new Color3f(1f, 1f, 1f);

BoundingSphere bounds = new BoundingSphere(new Point3d(0.0, 0.0, 0.0),

100.0);

Vector3f light1Direction = new Vector3f(3f,-20f,10f);

DirectionalLight light1 = new DirectionalLight(light1Color,

light1Direction);

light1.setInfluencingBounds(bounds);

objRoot.addChild(light1);

AmbientLight ambientLight = new AmbientLight(new Color3f(.5f, .5f, .5f));

ambientLight.setInfluencingBounds(bounds);

objRoot.addChild(ambientLight);

Background background = new Background(new Color3f(new Color(126, 87, 194)));

background.setApplicationBounds(bounds);

objRoot.addChild(background);

return objRoot;

}

public PingPongTable() {

//налаштовуємо вікно

setLayout(new BorderLayout());

GraphicsConfiguration config = SimpleUniverse.getPreferredConfiguration();

Canvas3D c = new Canvas3D(config);

// розміщуємо сцену в центрі фрейму

add("Center", c);

// підписуємо об'єкт поточного класу на подію натиснення кнопки клавіатури

c.addMouseMotionListener(this);

c.addMouseListener(this);

c.addMouseWheelListener(this);

// створюємо об'єкт таймеру з інтервалом 100 мілісекунд та підписуємо

Timer timer = new Timer(10, this);

// створюємо просту сцену та додаємо її до простору

BranchGroup scene = createSceneGraph();

SimpleUniverse u = new SimpleUniverse(c);

u.getViewingPlatform().setNominalViewingTransform();

u.addBranchGraph(scene);

timer.start();

}

void moveObject(Node o, float x, float y, float z) {

TransformGroup tg = new TransformGroup();

Transform3D transform = new Transform3D();

Vector3f vector = new Vector3f(x, y, z);

transform.setTranslation(vector);

tg.setTransform(transform);

tg.addChild(o);

objTrans.addChild(tg);

}

public void actionPerformed(ActionEvent e) {

z\_ball += z\_diff;

if (z\_ball < 0f || z\_ball > 0.38f) z\_diff = -z\_diff;

Transform3D move = new Transform3D();

move.setTranslation(new Vector3d(0f, 0f, -z\_ball));

ballGroup.setTransform(move);

Transform3D temp = new Transform3D();

temp.rotX(angleY);

trans.rotY(angleX);

trans.mul(temp);

trans.setScale(scale);

objTrans.setTransform(trans);

}

@Override

public void mouseDragged(MouseEvent e) {

if (e.getID() != MouseEvent.MOUSE\_DRAGGED) System.out.println(e.getID());

Point2d mouseCurrentPosition = new Point2d(e.getX(), e.getY());

if (mouseLastPosition != null) {

double dx = mouseCurrentPosition.x - mouseLastPosition.x;

double dy = mouseCurrentPosition.y - mouseLastPosition.y;

// if (dy < 0) System.out.println(dy / 150 + " " + angleY + " ");

angleX += dx / 150;

angleY += dy / 150;

}

mouseLastPosition = mouseCurrentPosition;

}

@Override

public void mouseReleased(MouseEvent e) {

mouseLastPosition = null;

}

@Override

public void mouseMoved(MouseEvent e) {

}

@Override

public void mouseClicked(MouseEvent e) {

}

@Override

public void mousePressed(MouseEvent e) {

}

@Override

public void mouseEntered(MouseEvent e) {

}

@Override

public void mouseExited(MouseEvent e) {

}

@Override

public void mouseWheelMoved(MouseWheelEvent e) {

scale \*= 1 - e.getPreciseWheelRotation() / 10;

}

}

Скріншоти результатів

