

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 “Математичні та алгоритмічні основи комп’ютерної графіки”

|  |  |  |  |
| --- | --- | --- | --- |
| Виконав  студент III курсу  групи КП-83  Мар’яненко Роман Геннадійович  (*прізвище, ім’я, по батькові*)  варіант № 11 |  |  | Зарахована  “\_\_\_\_” “\_\_\_\_\_\_\_\_\_\_\_\_” 20\_\_\_ р.  викладачем  Шкурат Оксаною Сергіївною (*прізвище, ім’я, по батькові*) |

Київ 2020

**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 у списку групи імпортують моделі формату .obj, парний варіант – .lwo.

**Варіант:** Формат .obj

**Лістинг коду програми**

|  |
| --- |
| Main.java |
| package com.company;  import com.sun.j3d.utils.universe.\*;  import javax.media.j3d.\*;  import javax.vecmath.\*;  import javax.media.j3d.Background;  import com.sun.j3d.loaders.\*;  import com.sun.j3d.loaders.objectfile.ObjectFile;  import com.sun.j3d.utils.image.TextureLoader;  import java.awt.\*;  import java.io.FileReader;  import java.io.IOException;  import java.util.Map;  import javax.swing.JFrame;  public class Main extends JFrame {  private static Canvas3D canvas;  private static TransformGroup t90;  private final static String \_\_ASSETS\_\_ = System.getProperty("user.dir") + "\\src\\assets\\";  public Main() throws IOException {  setUpWindow();  canvas = new Canvas3D(SimpleUniverse.getPreferredConfiguration());  canvas.setDoubleBufferEnable(true);  getContentPane().add(canvas, BorderLayout.CENTER);  SimpleUniverse universe = new SimpleUniverse(canvas);  universe.getViewingPlatform().setNominalViewingTransform();  BranchGroup root = new BranchGroup();  setUpBackground(root, "background.jpg");  setUpLight(root);  t90 = getModelTransformGroup("t90.obj", "t90.jpg", "t90");  root.addChild(t90);  root.compile();  universe.addBranchGraph(root);  }  public static void main(String[] args) {  try {  Main window = new Main();  Animation tankMovement = new Animation(t90);  canvas.addKeyListener(tankMovement);  window.setVisible(true);  } catch (IOException e) {  System.out.println(e.getMessage());  }  }  private void setUpWindow() {  setTitle("lab5");  setExtendedState(JFrame.MAXIMIZED\_BOTH);  setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);  }  private void setUpLight(BranchGroup root) {  BoundingSphere bounds = new BoundingSphere();  bounds.setRadius(100);  DirectionalLight dirlight = new DirectionalLight(new Color3f(1, 1, 1), new Vector3f(-1, -1, -1));  dirlight.setInfluencingBounds(bounds);  root.addChild(dirlight);  AmbientLight amblight = new AmbientLight(new Color3f(1, 1, 1));  amblight.setInfluencingBounds(new BoundingSphere());  root.addChild(amblight);  }  private void setUpBackground(BranchGroup root, String path) {  TextureLoader t = new TextureLoader(\_\_ASSETS\_\_ + path, canvas);  Background background = new Background(t.getImage());  BoundingSphere bounds = new BoundingSphere(new Point3d(0.0, 0.0, 0.0), 100.0);  background.setApplicationBounds(bounds);  background.setImageScaleMode(Background.SCALE\_FIT\_ALL);  root.addChild(background);  }  private void addAppearance(Shape3D shape, String path) {  TextureLoader loader = new TextureLoader(\_\_ASSETS\_\_ + path, "RGP", new Container());  Texture texture = loader.getTexture();  Appearance appearance = new Appearance();  appearance.setTexture(texture);  shape.setAppearance(appearance);  }  private TransformGroup getModelTransformGroup(String modelPath, String texturePath, String groupName) throws IOException {  Shape3D shape = getModelShape(modelPath, groupName);  addAppearance(shape, texturePath);  Transform3D transforms = new Transform3D();  Transform3D transformScale = new Transform3D();  transformScale.setScale(new Vector3d(0.4, 0.4, 0.4));  Transform3D transformRotX = new Transform3D();  transformRotX.rotX(-Math.PI / 2.5);  Transform3D transformRotZ = new Transform3D();  transformRotZ.rotZ(-Math.PI / 4);  transforms.mul(transformScale);  transforms.mul(transformRotX);  transforms.mul(transformRotZ);  TransformGroup group = getModelGroup(shape);  group.setTransform(transforms);  return group;  }  private TransformGroup getModelGroup(Shape3D shape) {  TransformGroup group = new TransformGroup();  group.setCapability(TransformGroup.ALLOW\_TRANSFORM\_WRITE);  group.addChild(shape);  return group;  }  private Shape3D getModelShape(String path, String groupName) throws IOException {  ObjectFile file = new ObjectFile(ObjectFile.RESIZE);  file.setFlags(ObjectFile.RESIZE | ObjectFile.TRIANGULATE | ObjectFile.STRIPIFY);  Scene scene = file.load(new FileReader(\_\_ASSETS\_\_ + path));  Map<String, Shape3D> map = scene.getNamedObjects();  Shape3D shape = map.get(groupName);  scene.getSceneGroup().removeChild(shape);  return shape;  }  } |

|  |
| --- |
| Animation.java |
| package com.company;  import java.awt.event.ActionEvent;  import java.awt.event.ActionListener;  import java.awt.event.KeyEvent;  import java.awt.event.KeyListener;  import javax.media.j3d.\*;  import javax.swing.Timer;  import javax.vecmath.\*;  class KeyCodes {  final static int ROTATE\_UP = 38;  final static int ROTATE\_DOWN = 40;  final static int ROTATE\_LEFT = 37;  final static int ROTATE\_RIGHT = 39;  final static int MOVE\_UP = 87;  final static int MOVE\_DOWN = 83;  final static int MOVE\_LEFT = 65;  final static int MOVE\_RIGHT = 68;  }  public class Animation implements ActionListener, KeyListener {  private TransformGroup group;  private Transform3D transform3D = new Transform3D();  private float x = 0;  private float y = 0;  private float dx = 0.02f;  private float dy = 0.02f;  private float da = 0.05f;  private boolean mu = false;  private boolean md = false;  private boolean ml = false;  private boolean mr = false;  private boolean ru = false;  private boolean rd = false;  private boolean rl = false;  private boolean rr = false;  Animation(TransformGroup group) {  this.group = group;  this.group.getTransform(this.transform3D);  Timer timer = new Timer(20, this);  timer.start();  }  private void Move() {  if (ml) x -= dx;  if (mr) x += dx;  if (mu) y += dy;  if (md) y -= dy;  transform3D.setTranslation(new Vector3f(x, y, 0));  Transform3D rotation = new Transform3D();  if (ru) rotation.rotX(-da);  if (rd) rotation.rotX(da);  if (rl) rotation.rotZ(da);  if (rr) rotation.rotZ(-da);  transform3D.mul(rotation);  group.setTransform(transform3D);  }  @Override  public void actionPerformed(ActionEvent e) {  Move();  }  @Override  public void keyPressed(KeyEvent ev) {  switch (ev.getKeyCode()) {  case KeyCodes.MOVE\_UP -> mu = true;  case KeyCodes.MOVE\_DOWN -> md = true;  case KeyCodes.MOVE\_LEFT -> ml = true;  case KeyCodes.MOVE\_RIGHT -> mr = true;  case KeyCodes.ROTATE\_UP -> ru = true;  case KeyCodes.ROTATE\_DOWN -> rd = true;  case KeyCodes.ROTATE\_LEFT -> rl = true;  case KeyCodes.ROTATE\_RIGHT -> rr = true;  }  }  @Override  public void keyTyped(KeyEvent e) {}  @Override  public void keyReleased(KeyEvent ev) {  switch (ev.getKeyCode()) {  case KeyCodes.MOVE\_UP -> mu = false;  case KeyCodes.MOVE\_DOWN -> md = false;  case KeyCodes.MOVE\_LEFT -> ml = false;  case KeyCodes.MOVE\_RIGHT -> mr = false;  case KeyCodes.ROTATE\_UP -> ru = false;  case KeyCodes.ROTATE\_DOWN -> rd = false;  case KeyCodes.ROTATE\_LEFT -> rl = false;  case KeyCodes.ROTATE\_RIGHT -> rr = false;  }  }  } |

**Результат**

****

****

****