

МIНIСТЕРСТВО ОСВIТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ”

Факультет прикладної математики

Кафедра програмного забезпечення комп’ютерних систем

**Лабораторна робота № 6**

з дисципліни “ МАОКГ”

|  |  |  |
| --- | --- | --- |
| Виконав  студент III курсу  групи КП-82  Залізний Захар Юрійович  (*прізвище, ім’я, по батькові*)  Варіант № 7 |  | Зарахована  “\_\_\_\_” “\_\_\_\_\_\_\_\_\_\_\_\_” 2021 р.  викладачем  Шкурат О. С.  (*прізвище, ім’я, по батькові*) |

Київ-2021

**Завдання** Виконати анімацію тривимірної сцени за варіантом.

**Результат виконання програми :**



**Код програми:**

|  |
| --- |
| **Mike.java** |
| package Lab6.src.com.company;  import com.sun.j3d.loaders.Scene;  import com.sun.j3d.loaders.objectfile.ObjectFile;  import com.sun.j3d.utils.behaviors.vp.OrbitBehavior;  import com.sun.j3d.utils.image.TextureLoader;  import com.sun.j3d.utils.universe.SimpleUniverse;  import javax.media.j3d.\*;  import javax.swing.\*;  import javax.vecmath.Color3f;  import javax.vecmath.Color4f;  import javax.vecmath.Point3d;  import javax.vecmath.Vector3f;  import java.awt.\*;  import java.util.Enumeration;  import java.util.Hashtable;  public class Mike extends JFrame {  //The canvas to be drawn upon.  public Canvas3D myCanvas3D;  public Mike() {  this.setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);  myCanvas3D = new Canvas3D(SimpleUniverse.*getPreferredConfiguration*());  SimpleUniverse simpUniv = new SimpleUniverse(myCanvas3D);  simpUniv.getViewingPlatform().setNominalViewingTransform();  createSceneGraph(simpUniv);  addLight(simpUniv);  OrbitBehavior ob = new OrbitBehavior(myCanvas3D);  ob.setSchedulingBounds(new BoundingSphere(new Point3d(0.0, 0.0, 0.0), Double.*MAX\_VALUE*));  simpUniv.getViewingPlatform().setViewPlatformBehavior(ob);  setTitle("Lab6");  setSize(700, 700);  getContentPane().add("Center", myCanvas3D);  setVisible(true);  }  public static void main(String[] args) {  Mike mike = new Mike();  }  public void createSceneGraph(SimpleUniverse su) {  ObjectFile f = new ObjectFile(ObjectFile.*RESIZE*);  Scene mikeScene = null;  try {  mikeScene = f.load("G:\\Maokg\\src\\main\\java\\Lab6\\src\\com\\company\\mike.obj");  } catch (Exception e) {  System.*out*.println("File loading failed:" + e);  }  Transform3D scaling = new Transform3D();  scaling.setScale(1.0 / 6);  Transform3D tfRoach = new Transform3D();  tfRoach.rotY(Math.*PI* / 6);  tfRoach.mul(scaling);  TransformGroup tgRoach = new TransformGroup(tfRoach);  TransformGroup sceneGroup = new TransformGroup();  Hashtable roachNamedObjects = mikeScene.getNamedObjects();  Enumeration enumer = roachNamedObjects.keys();  String name;  while (enumer.hasMoreElements()) {  name = (String) enumer.nextElement();  System.*out*.println("Name: " + name);  }  Appearance lightApp = new Appearance();  *setToMyDefaultAppearance*(lightApp, new Color3f(Color.*GREEN*));  TextureLoader loader = new TextureLoader("G:\\Maokg\\src\\main\\java\\Lab6\\src\\com\\company\\texture.jpg", "LUMINACE", new Container());  Texture texture = loader.getTexture();  texture.setBoundaryModeS(Texture.*WRAP*);  texture.setBoundaryModeT(Texture.*WRAP*);  texture.setBoundaryColor(new Color4f(1.0f, 0.0f, 0.0f, 0.0f));  TextureAttributes texAttr = new TextureAttributes();  texAttr.setTextureMode(TextureAttributes.*MODULATE*);  Appearance textureAP = new Appearance();  textureAP.setTexture(texture);  textureAP.setTextureAttributes(texAttr);  Shape3D leftLeg = (Shape3D) roachNamedObjects.get("left\_leg");  leftLeg.setAppearance(lightApp);  leftLeg.setAppearance(textureAP);  Shape3D rightLeg = (Shape3D) roachNamedObjects.get("right\_leg");  rightLeg.setAppearance(lightApp);  rightLeg.setAppearance(textureAP);  Shape3D leftHand = (Shape3D) roachNamedObjects.get("left\_hand");  leftHand.setAppearance(lightApp);  leftHand.setAppearance(textureAP);  Shape3D rightHand = (Shape3D) roachNamedObjects.get("right\_hand");  rightHand.setAppearance(lightApp);  rightHand.setAppearance(textureAP);  Shape3D body = (Shape3D) roachNamedObjects.get("monstr");  body.setAppearance(lightApp);  body.setAppearance(textureAP);  TransformGroup mike = new TransformGroup();  mike.addChild(body.cloneTree());  TransformGroup leftleggr = new TransformGroup();  TransformGroup rightleggr = new TransformGroup();  TransformGroup lefthandgr = new TransformGroup();  TransformGroup righthandgr = new TransformGroup();  leftleggr.addChild(leftLeg.cloneTree());  rightleggr.addChild(rightLeg.cloneTree());  lefthandgr.addChild(leftHand.cloneTree());  righthandgr.addChild(rightHand.cloneTree());  BoundingSphere bounds = new BoundingSphere(new Point3d(120.0, 250.0, 100.0), Double.*MAX\_VALUE*);  BranchGroup theScene = new BranchGroup();  Transform3D tCrawl = new Transform3D();  Transform3D tCrawl1 = new Transform3D();  tCrawl.rotY(-90D);  tCrawl1.rotX(-90D);  long crawlTime = 10000;  Alpha crawlAlpha = new Alpha(1,  Alpha.*INCREASING\_ENABLE*,  0,  0, crawlTime, 0, 0, 0, 0, 0);  float crawlDistance = 3.0f;  PositionInterpolator posICrawl = new PositionInterpolator(crawlAlpha,  sceneGroup, tCrawl, -9.0f, crawlDistance);  long crawlTime1 = 30000;  Alpha crawlAlpha1 = new Alpha(1,  Alpha.*INCREASING\_ENABLE*,  3000,  0, crawlTime1, 0, 0, 0, 0, 0);  float crawlDistance1 = 15.0f;  PositionInterpolator posICrawl1 = new PositionInterpolator(crawlAlpha1,  sceneGroup, tCrawl1, -9.0f, crawlDistance1);  Transform3D leftLegRotationAxis = new Transform3D();  leftLegRotationAxis.rotZ(Math.*PI* / 2);  int timeStart = 500;  int timeRotationHour = 500;  Alpha leftLegRotationAlpha = new Alpha(-1, Alpha.*INCREASING\_ENABLE* | Alpha.*DECREASING\_ENABLE*, timeStart, 0,  timeRotationHour, 0, 0, timeRotationHour, 0, 0);  RotationInterpolator leftLegRotation = new RotationInterpolator(leftLegRotationAlpha, leftleggr,  leftLegRotationAxis, (float) Math.*PI* / 4, 0.0f);  RotationInterpolator rightHandRotation = new RotationInterpolator(leftLegRotationAlpha, righthandgr,  leftLegRotationAxis, (float) Math.*PI* / 4, 0.0f);  BoundingSphere bounds\_leg = new BoundingSphere(new Point3d(0.0, 0.0, 0.0), Double.*MAX\_VALUE*);  leftLegRotation.setSchedulingBounds(bounds\_leg);  leftleggr.setCapability(TransformGroup.*ALLOW\_TRANSFORM\_WRITE*);  leftleggr.addChild(leftLegRotation);  rightHandRotation.setSchedulingBounds(bounds\_leg);  righthandgr.setCapability(TransformGroup.*ALLOW\_TRANSFORM\_WRITE*);  righthandgr.addChild(rightHandRotation);  Transform3D rightLegRotationAxis = new Transform3D();  rightLegRotationAxis.rotZ(Math.*PI* / 2);  Alpha rightLegRotationAlpha = new Alpha(-1, Alpha.*INCREASING\_ENABLE* | Alpha.*DECREASING\_ENABLE*, 0, 0,  timeRotationHour, 0, 0, timeRotationHour, 0, 0);  RotationInterpolator rightLegRotation = new RotationInterpolator(rightLegRotationAlpha, rightleggr,  rightLegRotationAxis, (float) Math.*PI* / 4, 0.0f);  RotationInterpolator leftHandRotation = new RotationInterpolator(rightLegRotationAlpha, lefthandgr,  rightLegRotationAxis, (float) Math.*PI* / 4, 0.0f);  rightLegRotation.setSchedulingBounds(bounds\_leg);  leftHandRotation.setSchedulingBounds(bounds\_leg);  rightleggr.setCapability(TransformGroup.*ALLOW\_TRANSFORM\_WRITE*);  rightleggr.addChild(rightLegRotation);  lefthandgr.setCapability(TransformGroup.*ALLOW\_TRANSFORM\_WRITE*);  lefthandgr.addChild(leftHandRotation);  BoundingSphere bs = new BoundingSphere(new Point3d(0.0, 0.0, 0.0), Double.*MAX\_VALUE*);  posICrawl.setSchedulingBounds(bs);  posICrawl1.setSchedulingBounds(bs);  sceneGroup.setCapability(TransformGroup.*ALLOW\_TRANSFORM\_WRITE*);  sceneGroup.addChild(posICrawl);  Alpha upRamp = new Alpha();  upRamp.setIncreasingAlphaDuration(800);  upRamp.setLoopCount(-1);  leftleggr.setCapability(TransformGroup.*ALLOW\_TRANSFORM\_WRITE*);  rightleggr.setCapability(TransformGroup.*ALLOW\_TRANSFORM\_WRITE*);  lefthandgr.setCapability(TransformGroup.*ALLOW\_TRANSFORM\_WRITE*);  righthandgr.setCapability(TransformGroup.*ALLOW\_TRANSFORM\_WRITE*);  sceneGroup.addChild(mike);  sceneGroup.addChild(leftleggr);  sceneGroup.addChild(rightleggr);  sceneGroup.addChild(lefthandgr);  sceneGroup.addChild(righthandgr);  tgRoach.addChild(sceneGroup);  theScene.addChild(tgRoach);  Background bg = new Background(new Color3f(0.5f, 0.5f, 0.5f));  bg.setApplicationBounds(bounds);  theScene.addChild(bg);  theScene.compile();  su.addBranchGraph(theScene);  }  public static void setToMyDefaultAppearance(Appearance app, Color3f col) {  app.setMaterial(new Material(col, col, col, col, 150.0f));  }  public void addLight(SimpleUniverse su) {  BranchGroup bgLight = new BranchGroup();  BoundingSphere bounds = new BoundingSphere(new Point3d(0.0, 0.0, 0.0), 100.0);  Color3f lightColour1 = new Color3f(0.5f, 1.0f, 1.0f);  Vector3f lightDir1 = new Vector3f(-1.0f, 0.0f, -0.5f);  DirectionalLight light1 = new DirectionalLight(lightColour1, lightDir1);  light1.setInfluencingBounds(bounds);  bgLight.addChild(light1);  su.addBranchGraph(bgLight);  }  } |