

МIНIСТЕРСТВО ОСВIТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ”

Факультет прикладної математики

Кафедра програмного забезпечення комп’ютерних систем

**Лабораторна робота № 5**

з дисципліни “ МАОКГ”

|  |  |  |
| --- | --- | --- |
| Виконав  студент III курсу  групи КП-72  Божко Володимир Сергійович  (*прізвище, ім’я, по батькові*)  Варіант № 3 |  | Зарахована  “\_\_\_\_” “\_\_\_\_\_\_\_\_\_\_\_\_” 2020  р.  викладачем  Шкурат О. С.  (*прізвище, ім’я, по батькові*) |

Київ-2020

Результат: :



|  |  |  |
| --- | --- | --- |
| |  | | --- | | Лістинг коду програми (class FirstMainClass): | | package lab;  import com.sun.j3d.utils.universe.\*;  import java.awt.Color; import javax.media.j3d.\*; import javax.media.j3d.Material; import javax.vecmath.\*; import javax.media.j3d.Background;  import com.sun.j3d.loaders.\*; import com.sun.j3d.loaders.objectfile.ObjectFile; import com.sun.j3d.loaders.lw3d.Lw3dLoader; 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 FirstMainClass extends JFrame {  static SimpleUniverse *universe*;  static Scene *scene*;  static Map<String, Shape3D> *nameMap*;  static BranchGroup *root*;  static Canvas3D *canvas*;   static TransformGroup *wholeCybertruck*;  static Transform3D *transform3D*;   public FirstMainClass () throws IOException{  configureWindow();  configureCanvas();  configureUniverse();  addModelToUniverse();  setCybertruckElementsList();  addAppearance();  addImageBackground();  addLightToUniverse();  addOtherLight();  ChangeViewAngle();  *root*.compile();  *universe*.addBranchGraph(*root*);  }   private void configureWindow() {  setTitle("Elon Musk animates his Cybertruck");  setSize(760,640);  setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);  }   private void configureCanvas(){  *canvas*=new Canvas3D(SimpleUniverse.*getPreferredConfiguration*());  *canvas*.setDoubleBufferEnable(true);  getContentPane().add(*canvas*,BorderLayout.*CENTER*);  }   private void configureUniverse(){  *root*= new BranchGroup();  *universe*= new SimpleUniverse(*canvas*);  *universe*.getViewingPlatform().setNominalViewingTransform();  }   private void addModelToUniverse() throws IOException{  *scene* = *getSceneFromFile*("3dModels//cebertruck2020.obj");  *root*=*scene*.getSceneGroup();  }   private void addLightToUniverse(){  Bounds bounds = new BoundingSphere();  Color3f color = new Color3f(65/255f, 30/255f, 25/255f);  Vector3f lightdirection = new Vector3f(1f,1f,1f);  DirectionalLight dirlight = new DirectionalLight(color,lightdirection);  dirlight.setInfluencingBounds(bounds);  *root*.addChild(dirlight);  }   private void printModelElementsList(Map<String,Shape3D> nameMap){  for (String name : nameMap.keySet()) {  System.*out*.printf("Name: %s\n", name);}  }   private void setCybertruckElementsList() {  *nameMap*=*scene*.getNamedObjects();  //Print elements of your model:  printModelElementsList(*nameMap*);   *wholeCybertruck* = new TransformGroup();  *transform3D* = new Transform3D();  *transform3D*.setScale(new Vector3d(1,1,1));  *wholeCybertruck*.setTransform(*transform3D*);  *root*.removeChild(*nameMap*.get("cybertruck2019"));  *wholeCybertruck*.addChild(*nameMap*.get("cybertruck2019"));  *wholeCybertruck*.setCapability(TransformGroup.*ALLOW\_TRANSFORM\_WRITE*);  *root*.addChild(*wholeCybertruck*);  }   Texture getTexture(String path) {  TextureLoader textureLoader = new TextureLoader(path,*canvas*);  Texture texture = textureLoader.getTexture();  texture.setBoundaryModeS(Texture.*WRAP*);  texture.setBoundaryModeT(Texture.*WRAP*);  texture.setBoundaryColor( new Color4f( 0.0f, 5.0f, 3.0f, 0.0f ) );  return texture;  }   Material getMaterial() {  Material material = new Material();  material.setAmbientColor ( new Color3f( 1f, 1f, 1f ) );  material.setDiffuseColor ( new Color3f( 0.5f, 0.5f, 0.5f ) );  material.setSpecularColor( new Color3f( 0.5f, 0.5f, 0.5f ) );  material.setShininess( 0.9f );  material.setLightingEnable(true);  return material;  }   private void addAppearance(){  Appearance cybertruckAppearance = new Appearance();  cybertruckAppearance.setTexture(getTexture("3dModels//metal.png"));  TextureAttributes texAttr = new TextureAttributes();  texAttr.setTextureMode(TextureAttributes.*COMBINE*);  cybertruckAppearance.setTextureAttributes(texAttr);  cybertruckAppearance.setMaterial(getMaterial());  Shape3D plane = *nameMap*.get("cybertruck2019");  plane.setAppearance(cybertruckAppearance);  }    private void addImageBackground(){  TextureLoader t = new TextureLoader("3dModels//Essay-Roads-Featured.jpg", *canvas*);  Background background = new Background(t.getImage());  background.setImageScaleMode(Background.*SCALE\_FIT\_ALL*);  BoundingSphere bounds = new BoundingSphere(new Point3d(0.0, 0.0, 0.0),100.0);  background.setApplicationBounds(bounds);  *root*.addChild(background);  }   private void ChangeViewAngle(){  ViewingPlatform vp = *universe*.getViewingPlatform();  TransformGroup vpGroup = vp.getMultiTransformGroup().getTransformGroup(0);  Transform3D vpTranslation = new Transform3D();  Vector3f translationVector = new Vector3f(0F, 1F, 0F);  vpTranslation.setTranslation(translationVector);  vpGroup.setTransform(vpTranslation);  }   private void addOtherLight(){  Color3f directionalLightColor = new Color3f(Color.*WHITE*);  Color3f ambientLightColor = new Color3f(Color.*WHITE*);  Vector3f lightDirection = new Vector3f(0F, 2F, 0F);   AmbientLight ambientLight = new AmbientLight(ambientLightColor);  DirectionalLight directionalLight = new DirectionalLight(directionalLightColor, lightDirection);   Bounds influenceRegion = new BoundingSphere(new Point3d(0.0, 0.0, 0.0),100.0);   ambientLight.setInfluencingBounds(influenceRegion);  directionalLight.setInfluencingBounds(influenceRegion);  *root*.addChild(ambientLight);  *root*.addChild(directionalLight);  }   public static Scene getSceneFromFile(String location) throws IOException {  ObjectFile file = new ObjectFile(ObjectFile.*RESIZE*);  file.setFlags (ObjectFile.*RESIZE* | ObjectFile.*TRIANGULATE* | ObjectFile.*STRIPIFY*);  return file.load(new FileReader(location));  }   //Not always works  public static Scene getSceneFromLwoFile(String location) throws IOException {  Lw3dLoader loader = new Lw3dLoader();  return loader.load(new FileReader(location));  }   public static void main(String[]args){  try {  FirstMainClass window = new FirstMainClass();  AnimationCybertruck cybertruckMovement = new AnimationCybertruck(*wholeCybertruck*, *transform3D*, window);  window.addKeyListener(cybertruckMovement);  window.setVisible(true);  }  catch (IOException ex) {  System.*out*.println(ex.getMessage());  }  } } | |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
| |  | | --- | | Лістинг коду програми (class FirstMainClass): | | package lab;  import javax.media.j3d.Transform3D; import javax.media.j3d.TransformGroup; import javax.swing.\*; import javax.vecmath.Vector3d; import javax.vecmath.Vector3f; import java.awt.\*; import java.awt.event.ActionEvent; import java.awt.event.ActionListener; import java.awt.event.KeyEvent; import java.awt.event.KeyListener;  public class AnimationCybertruck implements ActionListener, KeyListener {  private Button go;  private TransformGroup wholeCybertruck;  private Transform3D translateTransform;   private Transform3D rotateTransformX;  private Transform3D rotateTransformY;  private Transform3D rotateTransformZ;   private JFrame mainFrame;   private float a = 0.0f;  private float zoom = 1f;  private float xloc = 0.8f;  private float yloc = 0.0f;  private float zloc = 0.0f;  private Timer timer;   public AnimationCybertruck(TransformGroup wholeCybertruck, Transform3D trans, JFrame frame) {  go = new Button("Go");   rotateTransformX= new Transform3D();  rotateTransformY= new Transform3D();  rotateTransformZ= new Transform3D();   this.wholeCybertruck = wholeCybertruck;  this.mainFrame = frame;   wholeCybertruck.setTransform(trans);   translateTransform = trans;   zoom = (float) trans.getScale();   FirstMainClass.*canvas*.addKeyListener(this);  timer = new Timer(5, this);   Panel p = new Panel();  p.add(go);  mainFrame.add("North", p);  go.addActionListener(this);  go.addKeyListener(this);  }   @Override  public void actionPerformed(ActionEvent e) {  // start timer when button is pressed  if (e.getSource() == go) {  if (!timer.isRunning()) {  timer.start();  go.setLabel("Stop");  }  else {  timer.stop();  go.setLabel("Go");  }  }  else {  Move();  translateTransform.setScale(new Vector3d(zoom, zoom, zoom));  translateTransform.setTranslation(new Vector3f(xloc, yloc, zloc));  wholeCybertruck.setTransform(translateTransform);  }  }   private void Move() {  yloc -= (0.0055 \* a);  zloc -= (0.1 \* a);   if (zloc <= -31f) {  a = 0.0f;  zloc = -31f;  yloc = -1.7f;  }  if(zloc >= 2.84f) {  zloc = 2.84f;  yloc = 0.16f;  }   if(a >= 0.6f) {  a = 0.6f;  }  if(a <= -0.3f) {  a = -0.3f;  }  }   @Override  public void keyTyped(KeyEvent e) {  //Invoked when a key has been typed.  }   @Override  public void keyPressed(KeyEvent e) {  if (e.getKeyChar()=='w') {  a += 0.1;  }  if (e.getKeyChar()=='s') {  a -= 0.1;  }  if (e.getKeyChar()=='1') {  rotateTransformX.rotX(Math.*PI*/2);  translateTransform.mul(rotateTransformX);  }  if (e.getKeyChar()=='2') {  rotateTransformY.rotY(Math.*PI*/2);  translateTransform.mul(rotateTransformY);  }  if (e.getKeyChar()=='3') {  rotateTransformZ.rotZ(Math.*PI*/2);  translateTransform.mul(rotateTransformZ);  }  }      @Override  public void keyReleased(KeyEvent e) {  // Invoked when a key has been released.  } } | |  |