

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 №** **4**

з теми “Побудова найпростіших тривимірних об'єктів за допомогою бібліотеки Java3D та їх анімація”

|  |  |  |  |
| --- | --- | --- | --- |
| Виконала  студентка III курсу  групи КП-82  Лахман Ксенія  (*прізвище, ім’я, по батькові*)  варіант № 11 |  |  | Зарахована  “\_\_\_\_” “\_\_\_\_\_\_\_\_\_\_\_\_” 20\_\_\_ р.  викладачем  Шкурат Оксаною Сергіївною (*прізвище, ім’я, по батькові*) |

Київ 2021

**Мета:**

1) вивчення стандартних засобів Java3D для візуалізація зображення;

2) вивчення засобів анімації примітивів та складених об'єктів в Java3D.

**Завдання**:

За допомогою засобів, що надає бібліотека Java3D, побудувати тривимірний об’єкт. Для цього скористатися основними примітивами, що буде доцільно використовувати згідно варіанту: сфера, конус, паралелепіпед, циліндр. Об'єкт має складатися з 5-15 примітивів. Задати матеріал кожного примітиву, в разі необхідності накласти текстуру. В сцені має бути мінімум одне джерело освітлення.

Виконати анімацію сцени таким чином, щоб можна було розглянути об'єкт з усіх сторін. За бажанням можна виконати інтерактивні взаємодію з об'єктом за допомогою миші та клавіатури.

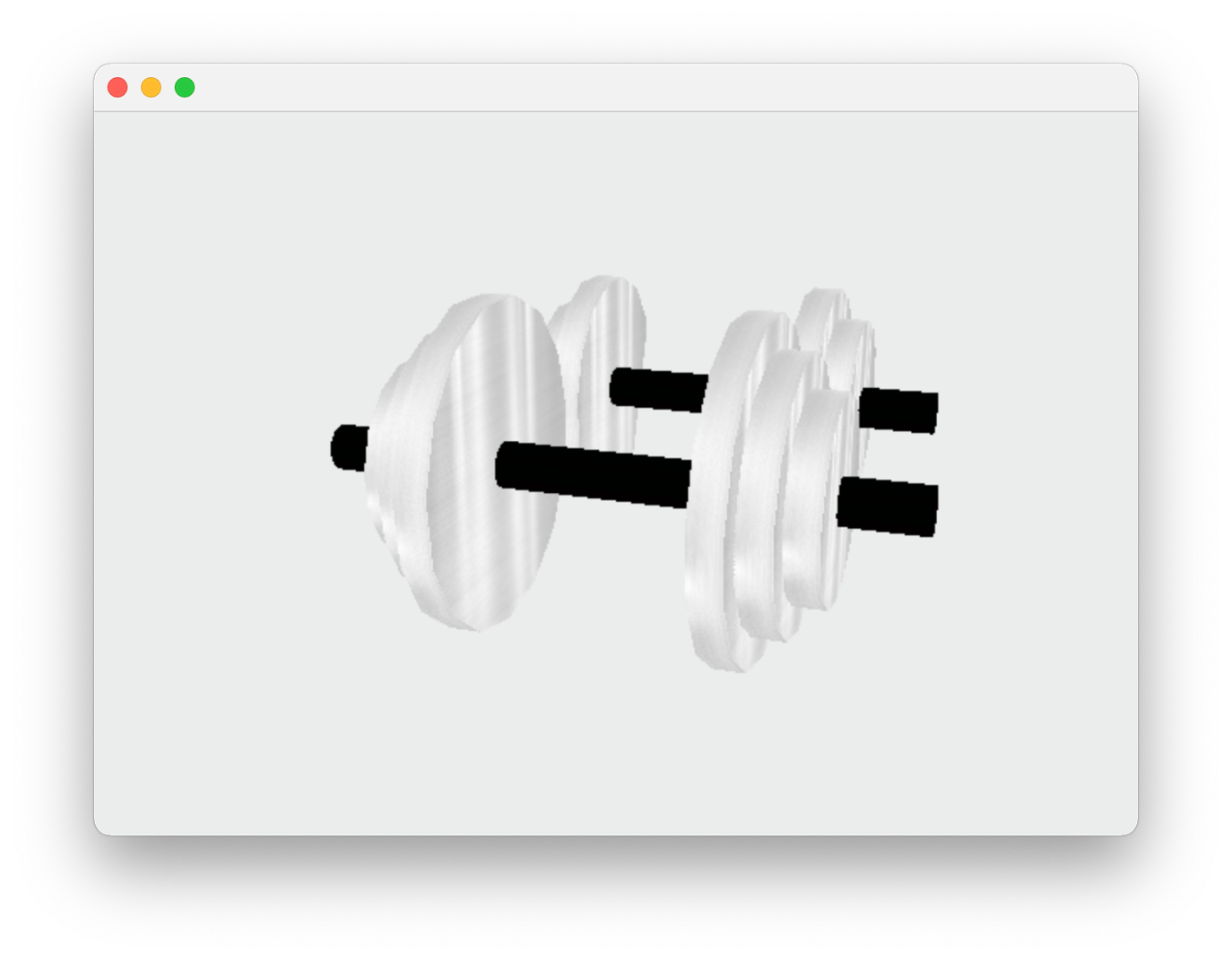
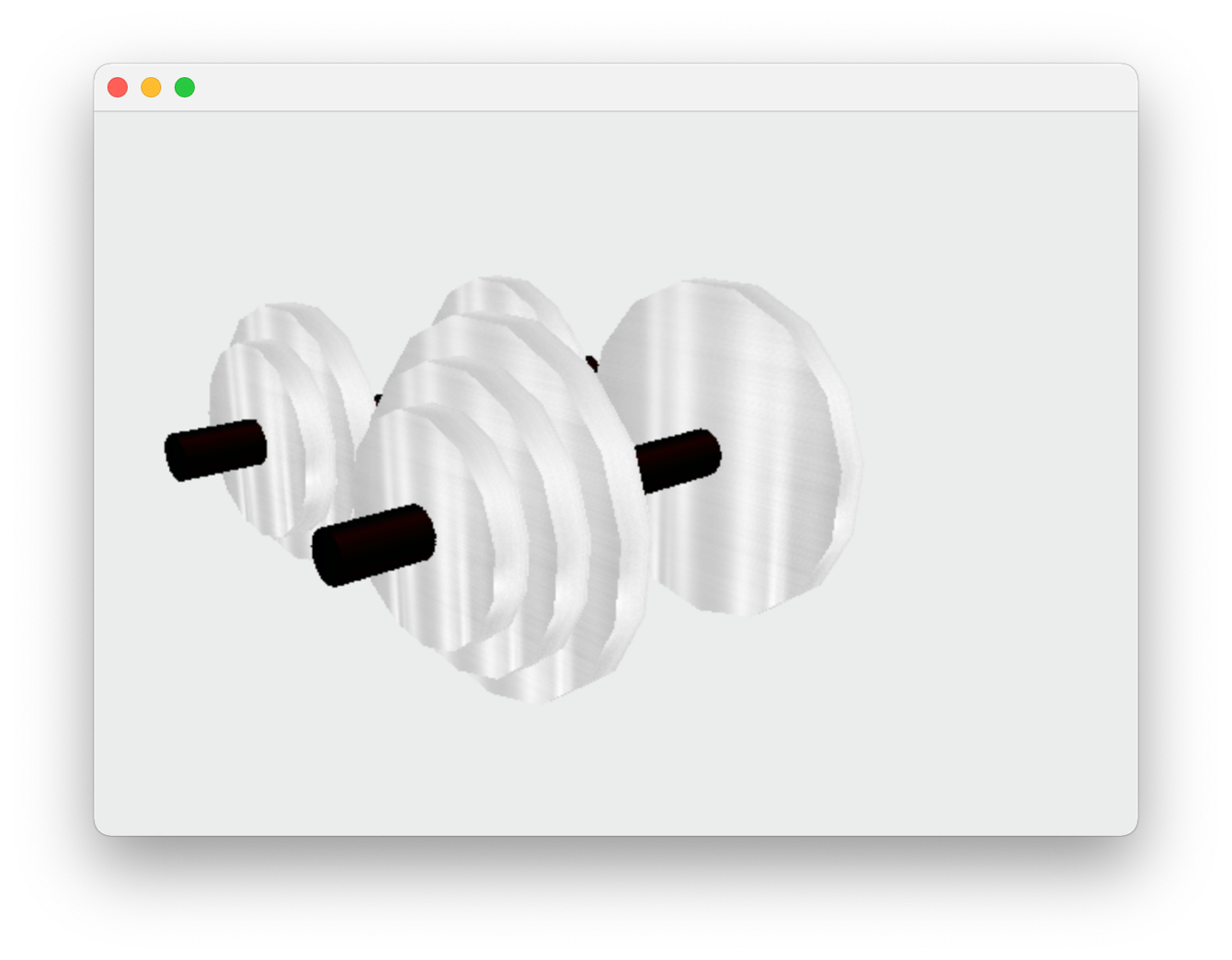
**Варіант:** гантелі

**Лістинг коду програми**

|  |
| --- |
| Dumbbell.java |
| package org.example; import com.sun.j3d.utils.geometry.Cylinder; import com.sun.j3d.utils.geometry.Primitive; 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;  public class Dumbbell implements ActionListener {  private float upperEyeLimit = 15.0f;  private float lowerEyeLimit = 8.0f;  private float farthestEyeLimit = 28.0f;  private float nearestEyeLimit = 22.0f;   private TransformGroup treeTransformGroup;  private TransformGroup viewingTransformGroup;  private Transform3D treeTransform3D = new Transform3D();  private Transform3D viewingTransform = new Transform3D();  private float angle = 0;  private float eyeHeight;  private float eyeDistance;  private boolean descend = true;  private boolean approaching = true;   public static void main(String[] args) {  new Dumbbell();  }   private Dumbbell() {  Timer timer = new Timer(50, this);  SimpleUniverse universe = new SimpleUniverse();    viewingTransformGroup = universe.getViewingPlatform().getViewPlatformTransform();  universe.addBranchGraph(createSceneGraph());   eyeHeight = upperEyeLimit;  eyeDistance = farthestEyeLimit;  timer.start();  }   private BranchGroup createSceneGraph() {  BranchGroup objRoot = new BranchGroup();   treeTransformGroup = new TransformGroup();  treeTransformGroup.setCapability(TransformGroup.*ALLOW\_TRANSFORM\_WRITE*);  buildDumbbell();  objRoot.addChild(treeTransformGroup);   Background background = new Background(new Color3f(new Color(235, 236, 236)));  BoundingSphere sphere = new BoundingSphere(new Point3d(100, 1000, 0), 1000);  background.setApplicationBounds(sphere);  objRoot.addChild(background);   BoundingSphere bounds = new BoundingSphere(new Point3d(0.0, 0.0, 0.0), 100.0);  Color3f light1Color = new Color3f(1.0f, 0, 0);  Vector3f light1Direction = new Vector3f(4.0f, -7.0f, -12.0f);  DirectionalLight light1 = new DirectionalLight(light1Color, light1Direction);  light1.setInfluencingBounds(bounds);  objRoot.addChild(light1);   Color3f ambientColor = new Color3f(1.0f, 1.0f, 1.0f);  AmbientLight ambientLightNode = new AmbientLight(ambientColor);  ambientLightNode.setInfluencingBounds(bounds);  objRoot.addChild(ambientLightNode);  return objRoot;  }   private TransformGroup getTransformGroup(Vector3f translation) {  Transform3D weightT = new Transform3D();  TransformGroup weightTG = new TransformGroup();  weightT.setTranslation(translation);  weightTG.setTransform(weightT);  return weightTG;  }   private void buildDumbbell() {  int primflags = Primitive.*GENERATE\_NORMALS* + Primitive.*GENERATE\_TEXTURE\_COORDS*;   *// Dumbbell 1* Transform3D stick1T = new Transform3D();  TransformGroup handle1Group = new TransformGroup();  handle1Group.setTransform(stick1T);   Cylinder handle1 = new Cylinder(0.7f, 18, primflags, Utils.*getHandleAppearance*("black.png"));  handle1Group.addChild(handle1);   Cylinder disk1 = new Cylinder(5, 1, primflags, Utils.*getDiskAppearance*("metal.jpg"));  Cylinder disk2 = new Cylinder(5, 1, primflags, Utils.*getDiskAppearance*("metal.jpg"));  Cylinder disk3 = new Cylinder(4, 1, primflags, Utils.*getDiskAppearance*("metal.jpg"));  Cylinder disk4 = new Cylinder(4, 1, primflags, Utils.*getDiskAppearance*("metal.jpg"));  Cylinder disk5 = new Cylinder(3, 1, primflags, Utils.*getDiskAppearance*("metal.jpg"));  Cylinder disk6 = new Cylinder(3, 1, primflags, Utils.*getDiskAppearance*("metal.jpg"));   buildDisk(disk1, handle1Group, new Vector3f(0, 4f, 0));  buildDisk(disk2, handle1Group, new Vector3f(0, -4f, 0));  buildDisk(disk3, handle1Group, new Vector3f(0, -5f, 0));  buildDisk(disk4, handle1Group, new Vector3f(0, 5f, 0));  buildDisk(disk5, handle1Group, new Vector3f(0, -6f, 0));  buildDisk(disk6, handle1Group, new Vector3f(0, 6f, 0));   *// Dumbbell 2* Transform3D stick2T = new Transform3D();  TransformGroup handle2Group = new TransformGroup();  stick2T.setTranslation(new Vector3f(10, 0, 0));  handle2Group.setTransform(stick2T);   Cylinder handle2 = new Cylinder(0.7f, 16, primflags, Utils.*getHandleAppearance*("black.png"));   handle2Group.addChild(handle2);   Cylinder disk7 = new Cylinder(4, 1, primflags, Utils.*getDiskAppearance*("metal.jpg"));  Cylinder disk8 = new Cylinder(4, 1, primflags, Utils.*getDiskAppearance*("metal.jpg"));  Cylinder disk9 = new Cylinder(3, 1, primflags, Utils.*getDiskAppearance*("metal.jpg"));  Cylinder disk10 = new Cylinder(3, 1, primflags, Utils.*getDiskAppearance*("metal.jpg"));   buildDisk(disk7, handle2Group, new Vector3f(0, 4f, 0));  buildDisk(disk8, handle2Group, new Vector3f(0, -4f, 0));  buildDisk(disk9, handle2Group, new Vector3f(0, -5f, 0));  buildDisk(disk10, handle2Group, new Vector3f(0, 5f, 0));   *//* treeTransformGroup.addChild(handle1Group);  treeTransformGroup.addChild(handle2Group);  }   public void buildDisk(Cylinder disk, TransformGroup handleGroup, Vector3f v) {  TransformGroup diskGroup = getTransformGroup(v);  diskGroup.addChild(disk);  handleGroup.addChild(diskGroup);  }   @Override  public void actionPerformed(ActionEvent e) {  float delta = 0.01f;   treeTransform3D.rotZ(angle);  treeTransformGroup.setTransform(treeTransform3D);  angle += delta;   if (eyeHeight > upperEyeLimit) {  descend = true;  } else if (eyeHeight < lowerEyeLimit) {  descend = false;  }  if (descend) {  eyeHeight -= delta;  } else {  eyeHeight += delta;  }   if (eyeDistance > farthestEyeLimit) {  approaching = true;  } else if (eyeDistance < nearestEyeLimit) {  approaching = false;  }  if (approaching) {  eyeDistance -= delta;  } else {  eyeDistance += delta;  }   Point3d eye = new Point3d(eyeDistance, eyeDistance, eyeHeight);  Point3d center = new Point3d(.0f, .0f, 0.1f);  Vector3d up = new Vector3d(.0f, .0f, 1.0f);  viewingTransform.lookAt(eye, center, up);  viewingTransform.invert();  viewingTransformGroup.setTransform(viewingTransform);  } } |

|  |
| --- |
| Utils.java |
| package org.example;  import com.sun.j3d.utils.image.TextureLoader; import javax.media.j3d.\*; import javax.vecmath.Color4f; import java.awt.\*; import java.util.Objects;  public class Utils {   public static Texture loadTexture(String picture) {  TextureLoader loader = new TextureLoader(Objects.*requireNonNull*(Utils.class.getClassLoader().getResource(picture)), "RGB", new Container());  Texture texture = loader.getTexture();   *// задаємо властивості границі* texture.setBoundaryModeS(Texture.*WRAP*);  texture.setBoundaryModeT(Texture.*WRAP*);  texture.setBoundaryColor(new Color4f(0.0f, 1.0f, 1.0f, 0.0f));   return texture;  }   public static Appearance getDiskAppearance(String picture) {  Appearance ap = new Appearance();   if (picture != "") {  Texture texture = *loadTexture*(picture);  TextureAttributes texAttr = new TextureAttributes();  texAttr.setTextureMode(TextureAttributes.*MODULATE*);   ap.setTexture(texture);  ap.setTextureAttributes(texAttr);  }  return ap;  }   public static Appearance getHandleAppearance(String picture) {  Appearance ap = new Appearance();  Material material = new Material(); *// create texture* ap.setMaterial(material);   if (picture != "") {  Texture texture = *loadTexture*(picture);  TextureAttributes texAttr = new TextureAttributes();  texAttr.setTextureMode(TextureAttributes.*MODULATE*);   ap.setTexture(texture);  ap.setTextureAttributes(texAttr);  }  return ap;  } } |

**Результат**

****