МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО
Факультет прикладної математики
Кафедра програмного забезпечення комп'ютерних систем

ЗВІТ

з лабораторної роботи № 5

Варіант 8

Виконав:
студент 3-го курсу,
групи КП-82,
спеціальності 121 –
Інженерія
програмного
забезпечення
Клапатюк Олександр
Петрович

Київ – 2020

Тема: Імпорт тривімирних моделей у середовище програмування java3D, обробка та маніапуляція цих зображень.
Мета: Здобути навички імпорту моделей, побудованих у тривимірних редакторах, (об’єктів форматів .obj, .lwo, .3ds) до бібліотеки java3D

AnimationDuck.java

```
public AnimationDuck(TransformGroup wholeDuck, Transform3D trans, JFrame frame){
    go = new Button("Go");
    this.wholeDuck =wholeDuck;
    this.translateTransform=trans;
    this.mainFrame=frame;

    rotateTransformX= new Transform3D();
    rotateTransformY= new Transform3D();
    rotateTransformZ= new Transform3D();

    Duck.canvas.addKeyListener(this);
    timer = new Timer(100, this);

    Panel p =new Panel();
    p.add(go);
    mainFrame.add("North",p);
    go.addActionListener(this);
    go.addKeyListener(this);
}

private void initialBoteState(){
    xloc=0.0f;
    yloc=0.0f;
    zloc=0.0f;
    zoom=1.0f;
    sign=1.0f;
    if(timer.isRunning()){timer.stop();}
    go.setLabel("Go");
}

@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));
        wholeDuck.setTransform(translateTransform);
    }
}

private void Move(){
    xloc += 0.1 * sign;
    if (Math.abs(xloc *2) >= 2 ) {
        sign = -1.0f * sign;
        rotateTransformZ.rotZ(Math.PI);
        translateTransform.mul(rotateTransformZ);
    }
}

@Override
public void keyTyped(KeyEvent e) {
    //Invoked when a key has been typed.
}
```

```
    @Override
    public void keyPressed(KeyEvent e) {
        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);
        }
        if (e.getKeyChar()=='0'){
            rotateTransformY.rotY(Math.PI/2.8);
            translateTransform.mul(rotateTransformY);
        }
    }

    @Override
    public void keyReleased(KeyEvent e) {
        // Invoked when a key has been released.
    }
}
```

Duck.java

```
public class Duck extends JFrame {
    static SimpleUniverse universe;
    static Scene scene;
    static Map<String, Shape3D> nameMap;
    static BranchGroup root;
    static Canvas3D canvas;

    static TransformGroup wholeDuck;
    static Transform3D transform3D;

    public Duck() throws IOException{
        configureWindow();
        configureCanvas();
        configureUniverse();
        addModelToUniverse();
        setDuckElementsList();
        addAppearance();
        addImageBackground();
        addLightToUniverse();
        addOtherLight();
        ChangeViewAngle();
        root.compile();
        universe.addBranchGraph(root);
    }

    private void configureWindow()  {
        setTitle("Duck Animation");
        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("source_folder//10602_Rubber_Duck_v1_L3.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 setDuckElementsList() {
    nameMap=scene.getNamedObjects();
    //Print elements of your model:
    printModelElementsList(nameMap);

    wholeDuck = new TransformGroup();

    transform3D = new Transform3D();
    transform3D.rotX(-Math.PI / 2);
    wholeDuck.setTransform(transform3D);
    transform3D.setTranslation(new Vector3f(0, -1.3f, 0));
    wholeDuck.setTransform(transform3D);
    transform3D.setScale(0.5f);
    wholeDuck.setTransform(transform3D);

    root.removeChild(nameMap.get("10602_rubber_duck_v1"));
    wholeDuck.addChild(nameMap.get("10602_rubber_duck_v1"));
    wholeDuck.setCapability(TransformGroup.ALLOW_TRANSFORM_WRITE);
    root.addChild(wholeDuck);
}

Texture getTexture(String path) {
    TextureLoader textureLoader = new TextureLoader(path,"LUMINANCE",canvas);
    Texture texture = textureLoader.getTexture();
    texture.setBoundaryModeS(Texture.WRAP);
    texture.setBoundaryModeT(Texture.WRAP);
    texture.setBoundaryColor( new Color4f( 0.0f, 1.0f, 0.0f, 0.0f ) );
    return texture;
}

Material getMaterial() {
    Material material = new Material();
    material.setAmbientColor ( new Color3f( 0.9f, 0.9f, 0.0f) );
    material.setDiffuseColor ( new Color3f( 1f, 1f, 1f ) );
    material.setSpecularColor( new Color3f( 1f, 1f, 1f ) );
    material.setShininess( 0.3f );
    material.setLightingEnable(true);
    return material;
}

private void addAppearance(){
    Appearance duckAppearance = new Appearance();
    duckAppearance.setTexture(getTexture("source_folder//app.jpg"));
    TextureAttributes texAttr = new TextureAttributes();
    texAttr.setTextureMode(TextureAttributes.COMBINE);
    duckAppearance.setTextureAttributes(texAttr);
    duckAppearance.setMaterial(getMaterial());
    Shape3D duck = nameMap.get("10602_rubber_duck_v1");
    duck.setAppearance(duckAppearance);
}

private void addColorBackground(){
    Background background = new Background(new Color3f(Color.CYAN));
    BoundingSphere bounds = new BoundingSphere(new Point3d(0.0, 0.0, 0.0),100.0);
    background.setApplicationBounds(bounds);
    root.addChild(background);
}
```

```
    private void addImageBackground(){
        TextureLoader t = new TextureLoader("source_folder//lake.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(0.0F, -1.2F, 6F);
        vpTranslation.setTranslation(translationVector);
        vpGroup.setTransform(vpTranslation);
    }

    private void addOtherLight(){
        Color3f directionalLightColor = new Color3f(Color.BLACK);
        Color3f ambientLightColor = new Color3f(Color.WHITE);
        Vector3f lightDirection = new Vector3f(-1F, -1F, -1F);

        AmbientLight ambientLight = new AmbientLight(ambientLightColor);
        DirectionalLight directionalLight = new DirectionalLight(directionalLightColor,
lightDirection);

        Bounds influenceRegion = new BoundingSphere();

        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));
    }


    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 {
            Duck window = new Duck();
            AnimationDuck duckMovement = new AnimationDuck(wholeDuck, transform3D, window);
            window.setVisible(true);
        }
        catch (IOException ex) {
            System.out.println(ex.getMessage());
        }
    }
}
```

Приклад роботи програми: