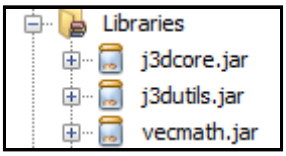
3D Cover Maker with Java

Do you have a need for a tool that allows you to create book covers? With the help of Java 3D API, you can build a simple book cover maker. You can modify the program so that it produces more sophisticated images.

To start with, you need to install the required library. Get the Java 3D development kit here: http://www.sun.com/desktop/java3D.

The jar files that you need to add to your project:



Here is the full code:

```
import java.awt.Color;
import java.awt.Dimension;
import java.awt.Toolkit;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.IOException;
import javax.swing.BorderFactory;
import javax.swing.BoxLayout;
```

```
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JTextField;
import javax.swing.SwingUtilities;
import com.sun.j3d.utils.universe.SimpleUniverse;
import com.sun.j3d.utils.geometry.Primitive;
import com.sun.j3d.utils.geometry.Box;
import com.sun.j3d.utils.image.TextureLoader;
import java.awt.Container;
import java.awt.Font;
import java.awt.GraphicsConfiguration;
import java.awt.image.BufferedImage;
import java.io.File;
import java.io.FileInputStream;
import javax.imageio.ImageIO;
import javax.media.j3d.Appearance;
import javax.media.j3d.Background;
import javax.media.j3d.BoundingSphere;
import javax.media.j3d.BranchGroup;
import javax.media.j3d.Canvas3D;
import javax.media.j3d.Font3D;
import javax.media.j3d.FontExtrusion;
import javax.media.j3d.ImageComponent;
import javax.media.j3d.ImageComponent2D;
import javax.media.j3d.Material;
import javax.media.j3d.Screen3D;
import javax.media.j3d.Shape3D;
import javax.media.j3d.Text3D;
import javax.media.j3d.Texture;
import javax.media.j3d.Transform3D;
import javax.media.j3d.TransformGroup;
import javax.swing.JFileChooser;
import javax.swing.filechooser.FileNameExtensionFilter;
import javax.vecmath.*;

class MainPanel extends JPanel{
    private final int mainPanelW;
    private final int mainPanelH;

    private final InputPanel inputPanel;
    private final OutputPanel outputPanel;
    private final ControlPanel controlPanel;

    private JTextField tfTitle;
    private File frontImage;

    private Canvas3D canvas=null;
    private SimpleUniverse su=null;

    private TransformGroup tgFront=null;
    private Transform3D t3dFront=null;
    private Transform3D rotFront;

    public MainPanel() throws IOException{
        Dimension screenSize = Toolkit.getDefaultToolkit().getScreenSize();
        mainPanelW = ((int)screenSize.getWidth())*40/100;
        mainPanelH = ((int)screenSize.getHeight())*80/100;
        setPreferredSize(new Dimension(mainPanelW,mainPanelH));
        setLayout(new BoxLayout(this,BoxLayout.Y_AXIS));
        setBorder(BorderFactory.createEmptyBorder(1,1,1,1));

        inputPanel=new InputPanel(mainPanelW,mainPanelH);
        add(inputPanel);

        outputPanel=new OutputPanel(mainPanelW,mainPanelH);
        add(outputPanel);

        controlPanel=new ControlPanel(mainPanelW,mainPanelH);
        add(controlPanel);
```

```
    }

    class InputPanel extends JPanel implements ActionListener{
        //private JTextField tfTitle;
        private JTextField tfImage;
        private JButton btnImage;
        //private File frontImage;
        private JButton btnSubmit;

        public InputPanel(int width, int height){
            setPreferredSize(new Dimension(width, height * 5/100));
            setBorder(BorderFactory.createLineBorder(Color.BLACK));

            add(new JLabel('Title:'));
            tfTitle = new JTextField(15);
            add(tfTitle);

            tfImage = new JTextField(15);
            add(tfImage);

            btnImage = new JButton('Image');
            btnImage.addActionListener(this);
            add(btnImage);

            btnSubmit = new JButton('OK');
            btnSubmit.addActionListener(this);
            add(btnSubmit);

            frontImage=new File('');
        }

        @Override
        public void actionPerformed(ActionEvent ae) {
            if(ae.getSource()==btnImage){
                JFileChooser openFile=new JFileChooser();
                FileNameExtensionFilter filter = new FileNameExtensionFilter('Image Files',
'jpg', 'png', 'gif', 'jpeg');
                openFile.setFileFilter(filter);

                int result = openFile.showOpenDialog(null);
                if (result == JFileChooser.APPROVE_OPTION){
                    frontImage = openFile.getSelectedFile();
                    tfImage.removeAll();
                    tfImage.setText(frontImage.getPath().toString());
                }
            }else if(ae.getSource()==btnSubmit){
                try {
                    FileInputStream fis = new FileInputStream(frontImage);
                    outputPanel.removeAll();
                    outputPanel.draw3DCover();
                    outputPanel.revalidate();
                } catch (IOException ex) {

                }
            }
        }
    }

    class OutputPanel extends JPanel{
        private int outputPanelW;
        private int outputPanelH;

        private GraphicsConfiguration config=null;
        private BranchGroup bg=null;
        //private Canvas3D canvas=null;
        //private SimpleUniverse su=null;

        //private TransformGroup tgFront=null;
        //private Transform3D t3dFront=null;
        //private Transform3D rotFront;
        private Box boxFront=null;
```

```
        private Appearance apFront=null;
        private TextureLoader loaderFront=null;
        private Texture textureFront=null;

        private float boxFrontW;
        private float boxFrontH;
        private float boxFrontD;
        private float boxFrontX;
        private float boxFrontY;
        private float boxFrontZ;

        public OutputPanel(int width, int height) throws IOException{
            outputPanelW = width;
            outputPanelH = height * 90/100;
            setPreferredSize(new Dimension(outputPanelW, outputPanelH));
            setBorder(BorderFactory.createLineBorder(Color.BLACK));

            boxFrontW = 0.65f;              //book width
            boxFrontH = 0.75f;              //book height
            boxFrontD = 0.02f;              //book depth (thickness)
            boxFrontX = 0f;
            boxFrontY = 0f;
            boxFrontZ = 0f;

            draw3DCover();
        }

        private void draw3DCover() throws IOException{
            config=SimpleUniverse.getPreferredConfiguration();
            canvas=new Canvas3D(config);
            canvas.setPreferredSize(new Dimension(outputPanelW, outputPanelH));
            add(canvas);

            su = new SimpleUniverse(canvas);
            su.getViewingPlatform().setNominalViewingTransform();
            bg = new BranchGroup();

            int primflags = Primitive.GENERATE_NORMALS + Primitive.GENERATE_TEXTURE_COORDS;
            boxFront=new Box(boxFrontW,boxFrontH,boxFrontD,primflags,null);
            boxFront.setCapability(Box.ENABLE_APPEARANCE_MODIFY);
            boxFront.setCapability(Box.GEOMETRY_NOT_SHARED);

            apFront = new Appearance();
            apFront.setCapability(Appearance.ALLOW_TEXTURE_WRITE);
            apFront.setCapability(Appearance.ALLOW_TEXGEN_WRITE);
            if(frontImage.exists()){
                loaderFront = new TextureLoader(frontImage.getCanonicalPath(),'RGB', new
Container());
            }else{
                loaderFront = new TextureLoader('src/Images/BlankFront.png','RGB', new
Container());
            }
            textureFront = loaderFront.getTexture();
            apFront.setTexture(textureFront);
            boxFront.getShape(Box.FRONT).setAppearance(apFront);

            t3dFront = new Transform3D();
            t3dFront.setTranslation(new Vector3f(boxFrontX, boxFrontY, boxFrontZ));

            tgFront = new TransformGroup();
            tgFront.setCapability(TransformGroup.ALLOW_TRANSFORM_READ);
            tgFront.setCapability(TransformGroup.ALLOW_TRANSFORM_WRITE);
            tgFront.setTransform(t3dFront);
            tgFront.addChild(boxFront);

            bg.addChild(tgFront);

            Background background = new Background(new Color3f(1f,1f,1f));
            BoundingSphere sphere = new BoundingSphere(new Point3d(0,0,0), 100000);
            background.setApplicationBounds(sphere);
            bg.addChild(background);
```

```
            //TEXT
            Vector3f posText = new Vector3f(-0.50f,0.55f,boxFrontD);
            tgFront.addChild(draw3DText(tfTitle.getText(),posText));
            //TEXT

            bg.compile();
            su.addBranchGraph(bg);
        }

        private TransformGroup draw3DText(String tempText, Vector3f tempPos){
            FontExtrusion fontEx = new FontExtrusion(new
java.awt.geom.Line2D.Double(0,0,0.4,0.0)) ;
            Font3D font3d = new Font3D(new Font('Arial', Font.BOLD, 2), fontEx);
            Text3D text3D = new Text3D(font3d, tempText);
            text3D.setCapability(Text3D.ALLOW_STRING_READ);
            text3D.setCapability(Text3D.ALLOW_STRING_WRITE);
            Shape3D shape3D = new Shape3D(text3D);
            Color3f black = new Color3f(0.0f,0.0f,0.0f);
            Color3f white = new Color3f(1.0f,1.0f,1.0f);
            Appearance apText = new Appearance();
            Material matText = new Material(black,black,white,white,1.0f);          //text color
            apText.setMaterial(matText);
            shape3D.setAppearance(apText);
            TransformGroup tgText = new TransformGroup();
            Transform3D t3dText = new Transform3D();
            t3dText.setScale(0.040);                        //font size
            t3dText.setTranslation(tempPos);                //test position
            tgText.setTransform(t3dText);
            tgText.addChild(shape3D);

            return tgText;
        }
    }

    class ControlPanel extends JPanel implements ActionListener{
        private JButton btnRotateL;
        private JButton btnRotateR;
        private JButton btnSave;

        public ControlPanel(int width, int height){
            setPreferredSize(new Dimension(width, height * 5/100));
            setBorder(BorderFactory.createLineBorder(Color.BLACK));

            btnRotateL = new JButton('Rotate Left');
            btnRotateL.addActionListener(this);
            add(btnRotateL);

            btnRotateR = new JButton('Rotate Right');
            btnRotateR.addActionListener(this);
            add(btnRotateR);

            btnSave = new JButton('Save');
            btnSave.addActionListener(this);
            add(btnSave);

            rotFront = new Transform3D();
        }

        @Override
        public void actionPerformed(ActionEvent ae) {
            if(ae.getSource()==btnRotateL){
                rotFront.rotY(-0.1);
                t3dFront.mul(rotFront);
                tgFront.setTransform(t3dFront);
            }else if(ae.getSource()==btnRotateR){
                rotFront.rotY(0.1);
                t3dFront.mul(rotFront);
                tgFront.setTransform(t3dFront);
            }else if(ae.getSource()==btnSave){
                JFileChooser saveFile=new JFileChooser();
                FileNameExtensionFilter filter = new FileNameExtensionFilter('Image Files',
'jpg', 'png', 'gif', 'jpeg');
```

```
                saveFile.setFileFilter(filter);

                int result=saveFile.showSaveDialog(null);

                if (result == JFileChooser.APPROVE_OPTION) {
                    File targetFile = saveFile.getSelectedFile();

                    if (!targetFile.exists()) {
                        try{
                            GraphicsConfiguration config =
SimpleUniverse.getPreferredConfiguration();
                            Canvas3D canvas2 = new Canvas3D(config, true);

                            Screen3D sOn = canvas.getScreen3D();            //SELF: set size
                            Screen3D sOff = canvas2.getScreen3D();
                            sOff.setSize(sOn.getSize());
                            sOff.setPhysicalScreenWidth(sOn.getPhysicalScreenWidth());
                            sOff.setPhysicalScreenHeight(sOn.getPhysicalScreenHeight());

                            su.getViewer().getView().addCanvas3D(canvas2);  //SELF: copy the
content to canvas2

                            BufferedImage bImage = new
BufferedImage(outputPanel.outputPanelW,outputPanel.outputPanelH,BufferedImage.TYPE_INT_ARGB);
                            ImageComponent2D buffer = new
ImageComponent2D(ImageComponent.FORMAT_RGBA,bImage);
                            buffer.setCapability(ImageComponent2D.ALLOW_IMAGE_READ);
                            canvas2.setOffScreenBuffer(buffer);
                            canvas2.renderOffScreenBuffer();
                            canvas2.waitForOffScreenRendering();

ImageIO.write(canvas2.getOffScreenBuffer().getRenderedImage(),'png',targetFile);
                        }catch(Exception ex){
                            ex.printStackTrace();
                        }
                    }
                }
            }
        }
    }
}



public class SimpleCoverMaker {
    private void createAndShowGUI() throws IOException{
        JFrame frame=new JFrame('Simple Cover Maker');
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        frame.setResizable(false);
        frame.add(new MainPanel());

        frame.pack();
        frame.setVisible(true);
    }

    public static void main(String[] args){
        SwingUtilities.invokeLater(new Runnable(){
            @Override
            public void run() {
                SimpleCoverMaker ap;
                ap = new SimpleCoverMaker();
                try {
                    ap.createAndShowGUI();
                } catch (IOException ex) {
                }
            }
        });
    }
```

```
}
```

Explanation:

The program contains one main panel and three sub panels (*inputPanel*, *outputPanel*, and *controlPanel*). The *inputPanel*, as the name implies, deals with user inputs (book title and cover image). The output will be displayed on the *outputPanel*. Through the use of two extra buttons in the *controlPanel*, the generated image can be rotated with respect to Y axis. The output can then be saved to your hard drive.

The function *draw3DCover()* is the core of the program. It will be called everytime a new user input is submitted. When the function is called, it will call another function, *draw3DText*, which is responsible to draw book title on the output panel.

```
canvas=new Canvas3D(config); canvas.setPreferredSize(new Dimension(outputPanelW, outputPanelH));
add(canvas);
```

The drawing area or the region for the rendered image is determined by a Canvas3D. The Canvas3D component is added to the Swing Panel.

```
su = new SimpleUniverse(canvas);
su.getViewingPlatform().setNominalViewingTransform();
```

The class *SimpleUniverse* build the standard objects for the 3D environment. The Canvas3D object is associated with this class.

```
bg = new BranchGroup();
```

The class *BranchGroup* is the place to attach child nodes.

```
boxFront=new Box(boxFrontW,boxFrontH,boxFrontD,primflags,null);
boxFront.setCapability(Box.ENABLE_APPEARANCE_MODIFY);
boxFront.setCapability(Box.GEOMETRY_NOT_SHARED);
```

The object of class *Box* is used as the base for the book cover. We can modify the size of the book cover by modifying this class.

```
apFront = new Appearance();
apFront.setCapability(Appearance.ALLOW_TEXTURE_WRITE);
apFront.setCapability(Appearance.ALLOW_TEXGEN_WRITE);
```

The *Appearance* object is used to set the components' attributes. It will be associated with the *Box* object so that the texture of the book cover can be changed.

```
if(frontImage.exists()){
    loaderFront = new TextureLoader(frontImage.getCanonicalPath(),'RGB', new Container());
}else{
    loaderFront = new TextureLoader('src/Images/BlankFront.png','RGB', new Container());
}
textureFront = loaderFront.getTexture();
apFront.setTexture(textureFront);
boxFront.getShape(Box.FRONT).setAppearance(apFront);
```

An image is set as the texture for the book cover. Default image is 'BlankFront.png'. As you can see, the value is then passed to the *Appearance* object.

```
t3dFront = new Transform3D();
t3dFront.setTranslation(new Vector3f(boxFrontX, boxFrontY, boxFrontZ));
```

Utilizing class *Transform3D, we set the initial position of the objet Box.*

```
tgFront = new TransformGroup();
tgFront.setCapability(TransformGroup.ALLOW_TRANSFORM_READ);
tgFront.setCapability(TransformGroup.ALLOW_TRANSFORM_WRITE);
```

```
tgFront.setTransform(t3dFront);
tgFront.addChild(boxFront);
bg.addChild(tgFront);
```

The class *TransformGroup* connects the object of the class *Transform3D with the object of the class Box . Put it simply, whether we use the class Transform3D for translation, scaling, or rotation , it will affect the child nodes in the TransformGroup.*

```
Background background = new Background(new Color3f(1f,1f,1f));
BoundingSphere sphere = new BoundingSphere(new Point3d(0,0,0), 100000);
background.setApplicationBounds(sphere);
bg.addChild(background);
```

The class *background* sets the background color of the image ((1f,1f,1f) = white). The class *BoundingSphere* sets influencing bounds. Background is then added to the branch.

```
Vector3f posText = new Vector3f(-0.50f,0.55f,boxFrontD);
tgFront.addChild(draw3DText(tfTitle.getText(),posText));
```

It is the code that calls *draw3DText*, the function to draw the title of the book cover.

```
bg.compile();
su.addBranchGraph(bg);
```

All child nodes under the *BranchGroup* are added to *SimpleUniverse*.

Now, have a look at what is included in the function *draw3DText.* As you can see, the function has two parameters, one for book title and the other for the text position. The return value is an object of the class *TransformGroup*.

```
FontExtrusion fontEx = new FontExtrusion(new java.awt.geom.Line2D.Double(0,0,0.4,0.0)) ;
Font3D font3d = new Font3D(new Font('Arial', Font.BOLD, 2), fontEx);
```

The above code is very intuitive. Basically, it describes text depth and font type.

```
t3dText.setScale(0.040);                        t3dText.setTranslation(tempPos);
```

If you want to modify the font size and the text position, just change the two values above.

```
if(ae.getSource()==btnRotateL){
                rotFront.rotY(-0.1);
                t3dFront.mul(rotFront);
                tgFront.setTransform(t3dFront);
 }else if(ae.getSource()==btnRotateR){
                rotFront.rotY(0.1);
                t3dFront.mul(rotFront);
                tgFront.setTransform(t3dFront);
 }
```

When the button *btnRotateL* is clicked, the image will be rotated to the left. The opposite occurs when the button *btnRotateR* is clicked.

```
GraphicsConfiguration config = SimpleUniverse.getPreferredConfiguration();
Canvas3D canvas2 = new Canvas3D(config, true);
.
.
ImageIO.write(canvas2.getOffScreenBuffer().getRenderedImage(),'png',targetFile);
```

When saving the image, an off-screen rendering process is performed by creating another *Canvas3D* and setting an off-screen buffer that temporarily holds the rendered image. The rendered 2D image is then assigned back to the the *Canvas3D*.