```java
/*
 * DisplayViewPanel.java
 *
 * Created on January 22, 2007, 2:36 PM
 */

package Vista2;

import Controlador2.DefaultController;
import java.awt.AlphaComposite;
import java.awt.BorderLayout;
import java.awt.Color;
import java.awt.Dimension;
import java.awt.Font;
import java.awt.Graphics;
import java.awt.Graphics2D;
import java.awt.RenderingHints;
import java.awt.Shape;
import java.awt.font.TextAttribute;
import java.awt.font.TextLayout;
import java.awt.geom.AffineTransform;
import java.beans.PropertyChangeEvent;
import java.util.HashMap;
import javax.swing.JComponent;

/**
 * This is a custom view panel that displays a "document" consisting of a single
 * text element. Both the document and the text element respond to changes in
 * the model state.
 *
 * @author Robert Eckstein
 */

public class DisplayViewPanel extends AbstractViewPanel
{

    //  The controller used by this view

    private DefaultController controller;

    //  A private inner class that performs the painting of the text
    //  component

    private TextElementComponent paintedString;


    /**
     * Creates new form TextElementDisplayPanel
     * @param controller An object implenting the controller interface that
     *        this view can use to process GUI actions
     */
    public DisplayViewPanel(DefaultController controller) {

        this.controller = controller;

        initComponents();
        localInitialization();
    }

    // <editor-fold defaultstate="collapsed" desc=" Local Initialization ">

    /**
     * Initialization method called from the constructor
     */
    public void localInitialization() {

        paintedString = new TextElementComponent();

        displayPanel.add(paintedString, BorderLayout.CENTER);
        displayPanel.repaint();
    }
    // </editor-fold>

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    // <editor-fold defaultstate="collapsed" desc=" Generated Code ">//GEN-BEGIN:initComponents
    private void initComponents() {
        documentNameLabel = new javax.swing.JLabel();
        documentScrollPane = new javax.swing.JScrollPane();
        resizablePanel = new javax.swing.JPanel();
        displayPanel = new javax.swing.JPanel();

        documentNameLabel.setFont(new java.awt.Font("Dialog", 1, 14));
        documentNameLabel.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        documentNameLabel.setText("title");

        displayPanel.setLayout(new java.awt.BorderLayout());

        displayPanel.setBackground(new java.awt.Color(255, 255, 255));
        displayPanel.setBorder(javax.swing.BorderFactory.createEtchedBorder());
        displayPanel.setMaximumSize(new java.awt.Dimension(200, 200));
        displayPanel.setMinimumSize(new java.awt.Dimension(200, 200));
        displayPanel.setPreferredSize(new java.awt.Dimension(200, 200));
        resizablePanel.add(displayPanel);

        documentScrollPane.setViewportView(resizablePanel);

        javax.swing.GroupLayout layout = new javax.swing.GroupLayout(this);
        this.setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()
                .addContainerGap()
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
                    .addComponent(documentScrollPane, javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE, 450, Short.MAX_VALUE)
                    .addComponent(documentNameLabel, javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE, 450, Short.MAX_VALUE))
                .addContainerGap())
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(documentNameLabel)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(documentScrollPane, javax.swing.GroupLayout.DEFAULT_SIZE, 391,
Short.MAX_VALUE)
                .addContainerGap())
        );
    }// </editor-fold>//GEN-END:initComponents

    // <editor-fold defaultstate="collapsed" desc=" Model Event Handling Code ">

    /**
     * Called by the controller when it needs to pass along a property change
     * from a model.
     * @param evt The property change event
     */
    public void modelPropertyChange(PropertyChangeEvent evt) {

        if (evt.getPropertyName().equals(DefaultController.DOCUMENT_NAME_PROPERTY)) {
            documentNameLabel.setText((String)evt.getNewValue());
        }

        else if (evt.getPropertyName().equals(DefaultController.DOCUMENT_WIDTH_PROPERTY)) {

            Dimension newSize =
                    new Dimension((Integer)evt.getNewValue(), displayPanel.getHeight());
```

```java
            displayPanel.setPreferredSize(newSize);
            displayPanel.setSize(newSize);

        }

        else if (evt.getPropertyName().equals(DefaultController.DOCUMENT_HEIGHT_PROPERTY)) {

            Dimension newSize =
                    new Dimension(displayPanel.getWidth(), (Integer)evt.getNewValue());

            displayPanel.setPreferredSize(newSize);
            displayPanel.setSize(newSize);

        }

        else if (evt.getPropertyName().equals(DefaultController.ELEMENT_X_PROPERTY)) {
            paintedString.xPosition = (Integer)evt.getNewValue();
        }

        else if (evt.getPropertyName().equals(DefaultController.ELEMENT_Y_PROPERTY)) {
            paintedString.yPosition = (Integer)evt.getNewValue();
        }

        else if (evt.getPropertyName().equals(DefaultController.ELEMENT_TEXT_PROPERTY)) {
            paintedString.text = (String)evt.getNewValue();
        }

        else if (evt.getPropertyName().equals(DefaultController.ELEMENT_FONT_PROPERTY)) {
            paintedString.font = (Font)evt.getNewValue();
        }

        else if (evt.getPropertyName().equals(DefaultController.ELEMENT_ROTATION_PROPERTY)) {
            paintedString.rotation = (Integer)evt.getNewValue();
        }

        else if (evt.getPropertyName().equals(DefaultController.ELEMENT_OPACITY_PROPERTY)) {
            paintedString.opacity = (Integer)evt.getNewValue();
        }

        revalidate();
        repaint();

    }

    // </editor-fold>



    //  An inner class that we can use to paint the text component. A real
    //  drawing program would have a more sophisticated solution, but this
    //  is presented as is for brevity.

    private class TextElementComponent extends JComponent {

        public int xPosition;
        public int yPosition;
        public Font font;
        public String text;
        public int rotation;
        public int opacity;

        public void paintComponent(Graphics g) {

            Graphics2D g2 = (Graphics2D)g;

            //  If the current string in the model is empty, then we can return
            //  without painting anything.

            if (text.equals(""))
                return;

            //  Set up our rendering hints
            HashMap renderingHints = new HashMap();
            renderingHints.put(RenderingHints.KEY_ANTIALIASING, RenderingHints.VALUE_ANTIALIAS_ON);
            renderingHints.put(RenderingHints.KEY_FRACTIONALMETRICS, 
RenderingHints.VALUE_FRACTIONALMETRICS_ON);
            renderingHints.put(RenderingHints.KEY_COLOR_RENDERING, 
RenderingHints.VALUE_COLOR_RENDER_QUALITY);
            renderingHints.put(RenderingHints.KEY_STROKE_CONTROL, RenderingHints.VALUE_STROKE_PURE);
            g2.addRenderingHints(renderingHints);

            //  Set up an attributes HashMap to represent the current font

            HashMap attributes = new HashMap();
            attributes.put(TextAttribute.FONT, font);

            //  Create an AlphaComposite using the current opacity value

            AlphaComposite composite = AlphaComposite.getInstance(
                AlphaComposite.SRC_OVER, (float)(opacity / 100.0));
            g2.setComposite(composite);

            //  Create an AffineTransform to handle the X and Y position, as
            //  well as the rotation.

            AffineTransform affineTransform = new AffineTransform();
            affineTransform.translate(xPosition, yPosition);
            affineTransform.rotate(rotation*(Math.PI / 180.0));
            g2.transform(affineTransform);

            //  Obtain the textShape using the proper attributes, then draw it and fill it

            TextLayout textLayout = new TextLayout(
                    text, attributes, g2.getFontRenderContext());
            Shape textShape = textLayout.getOutline(null);

            g2.setPaint(Color.black);
            g2.fill(textShape);

            g2.setPaint(Color.black);
            g2.draw(textShape);

        }

    }

    // Variables declaration - do not modify//GEN-BEGIN:variables
    private javax.swing.JPanel displayPanel;
    private javax.swing.JLabel documentNameLabel;
    private javax.swing.JScrollPane documentScrollPane;
    private javax.swing.JPanel resizablePanel;
    // End of variables declaration//GEN-END:variables

}
```