```java
/* This is the main driver for the Dot-Plot program.
 * Compilation:  javac DotPlotUI.java
 * Usage:        java DotPlotUI
 *
 */
import java.awt.*;
import javax.swing.*;
import java.util.*;
import java.io.*;
import java.awt.event.*;

public class DotPlotUI implements ActionListener
{
    // the following are the components used to make
    // up the application UI
    GraphPane graphPane;
    JTabbedPane tabbedPane;
    TextField tfDirectory = new TextField();
    TextField tfFile = new TextField();
    TextField tfFilter = new TextField();
    FileDialog fd;
    JFrame f;
    final JButton button1;
    final JButton button2;
    final JButton button3;
    TextArea textArea1;
    TextArea textArea2;
    JLabel window_label;
    JTextField window;
    JLabel threshold_label;
    JTextField threshold;
    TextArea status;
    JLabel status_label;
    JPanel pre_plot;

   /*  The is the sole constructionfor the DotPlotUI
    *  In this constructor all the components are added
    *  sequentially/
    */
    public DotPlotUI()
    {
        f = new JFrame("Dot-plot Comparative Sequence Analysis");
        f.getContentPane().setLayout(null);
        f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        GraphicsEnvironment env = GraphicsEnvironment.getLocalGraphicsEnvironment();
        Rectangle bounds = env.getMaximumWindowBounds();

        button1 = new JButton("Fasta File 1");
        button2 = new JButton("Fasta File 2");
        button3 = new JButton("Create Dot-plot");
        textArea1 = new TextArea("Fasta File 1", 10,10,TextArea.SCROLLBARS_NONE);
        textArea2 = new TextArea("Fasta File 2", 10,10,TextArea.SCROLLBARS_NONE);
        status = new TextArea("", 10,bounds.width,TextArea.SCROLLBARS_VERTICAL_ONLY);
        status.setEditable(false);
        window_label = new JLabel("Windows Size");
        window = new JTextField(10);
        threshold_label = new JLabel("Threshold Size");
        threshold = new JTextField(10);
        status_label = new JLabel("Process Status/Error Info");
        pre_plot = new JPanel();
        pre_plot.setBackground(new Color(235,235,235));

        f.getContentPane().add(textArea1);
        f.getContentPane().add(button1);
        f.getContentPane().add(textArea2);
        f.getContentPane().add(button2);
        f.getContentPane().add(window_label);
        f.getContentPane().add(window);
        f.getContentPane().add(threshold_label);
        f.getContentPane().add(threshold);
        f.getContentPane().add(button3);
        f.getContentPane().add(status);
        f.getContentPane().add(status_label);
        f.getContentPane().add(pre_plot);

        button1.addActionListener(this);
        button2.addActionListener(this);
        window.addActionListener(this);
        threshold.addActionListener(this);
        button3.addActionListener(this);

        textArea1.setBounds(10,10,182,150);
        Dimension size = button1.getPreferredSize();
        button1.setBounds(10,170,size.width,size.height);
        textArea2.setBounds(10,180+size.height,182,150);
        size = button1.getPreferredSize();
        button2.setBounds(10,340+size.height,size.width,size.height);
        size = threshold_label.getPreferredSize();
        window_label.setBounds(10,420+size.height,size.width,size.height );
        window.setBounds(10+size.width+10,420+size.height,size.width,size.height );
        threshold_label.setBounds(10,445+size.height,size.width,size.height );
        threshold.setBounds(10+size.width+10,445+size.height,size.width,size.height );
        size = button3.getPreferredSize();
        button3.setBounds(10,485+size.height,size.width+5,size.height+5);
        button3.setBackground(new Color(153,153,153));
        size = status_label.getPreferredSize();
        status_label.setBounds((int)((bounds.width/2)-(size.width/2)),bounds.height-115,size
        status.setBounds(10,bounds.height-100,bounds.width-30,65);
        status.setBackground(new Color(255,255,204));
        status.setForeground(Color.red);
        pre_plot.setBounds(300,10, 600, 600);

        f.setSize((int)bounds.getWidth(),(int)bounds.getHeight());
        f.setLocation(0,0);
        f.setVisible(true);
    }

    /* This handles all action events to the Application UI.
    *  It poulates the Fasta File TextFields, and ultimatley
    *  produces the dot-plot.
    */
    public void actionPerformed(ActionEvent evt)
    {
        // load the first fasta file
        if ((evt.getActionCommand()).equals("Fasta File 1")) {
            fd = new FileDialog(f, null, FileDialog.LOAD);
            fd.setDirectory(".\\");
        }
        // load the second fasta file
        else if ((evt.getActionCommand()).equals("Fasta File 2"))
        {
            fd = new FileDialog(f, null, FileDialog.LOAD);
            fd.setDirectory(".\\");
        }
        // display the dot-plot of the previously loaded data
        else if ((evt.getActionCommand()).equals("Create Dot-plot"))
        {
            graphPane = new GraphPane(textArea1.getText(),textArea2.getText(),getWindow(),ge
            f.getContentPane().add(graphPane);
            graphPane.setBounds(300,10, 600, 600);
            // display program status
            status.append("Info: Dot-plot sequence analysis diagram created!");
        }
        // Populate textArea_1 with sequences
        if ((evt.getActionCommand()).equals("Fasta File 1"))
        {
            fd.show();
            String sequence = this.readFile(fd.getDirectory()+fd.getFile());
            textArea1.replaceRange(sequence,0,textArea1.getText().length());
            // display program status
            status.append("Info: Fast File 1 sequence data loaded, " + sequence.length()
                          + " characters in length.\n");
        }
        // Populate textArea_2 with sequences
        else if((evt.getActionCommand()).equals("Fasta File 2"))
        {
            fd.show();
            String sequence = this.readFile(fd.getDirectory()+fd.getFile());
            textArea2.replaceRange(sequence,0,textArea2.getText().length());
            // display program status
            status.append("Info: Fast File 2 sequence data loaded, " + sequence.length()
                                +" characters in length.\n");
        }

    }

    /* This method is used to collect the window size in the
    *  text-fields. If they are empty or erroneous a default value
    *  of 100 is chosen, and a warning message outputted.
    */
    public int getWindow()
    {

        String win_val = window.getText();
        if(win_val.equals("") || win_val == null)
        {
            status.append("Warning: No window size selected, using default window size of 10
            return 100;
        }
        win_val.trim();
        boolean is_number = true;

        for(int i=0; i<win_val.length(); i++)
        {
            if(Character.isDigit(win_val.charAt(i)) == false)
            {
                is_number = false;
                break;
            }
        }
        if(is_number == false)
        {
            status.append("Error: The chosen window size is not a number, using default wind
            return 100;
        }
        else
        {
            status.append("Info: Window size set at: " + win_val + ".\n");
            return Integer.parseInt(win_val);
        }
    }

    /* This method is used to collect the threshold size in the
    *  text-fields. If they are empty or erroneous a default value
    *  of 100 is chosen, and a warning message outputted.
    */
    public int getThreshold()
    {

        String thres_val = window.getText();
        if(thres_val.equals("") || thres_val == null)
        {
            status.append("Warning: No threshold size selected, using default threshold size
            return 100;
        }
        thres_val.trim();
        boolean is_number = true;

        for(int i=0; i<thres_val.length(); i++)
        {
            if(Character.isDigit(thres_val.charAt(i)) == false)
            {
                is_number = false;
                break;
            }
        }
        if(is_number == false)
        {
            status.append("Error: The chosen threshold size is not a number, using default t
            return 100;
        }
        else
        {
            status.append("Info: Threshold size set at: " + thres_val + ".\n");
            return Integer.parseInt(thres_val);
        }
    }

    /* This method reads the genomic sequences from the file
    *  passed in the argument. The file must be of FASTA formatting.
    */
    public String readFile(String file)
    {
        BufferedReader inFile = null;
        String sequence = "";

        try{
            inFile = new BufferedReader(new FileReader(file));

            String str = null;

            while((str = inFile.readLine()) != null)
            {
                if(!str.equals(""))
                {
                    if(str.substring(0,1).equals(">"))
                    {
                        while((str = inFile.readLine()) != null && !str.equals(""))
                            sequence = sequence.concat(str);
                    }
                }
            }
        }
        catch (IOException e)
        {
            System.out.println("INPUT ERROR: Input file not recognized at DotPlotUI.readFile
            System.exit(1);
        }
        return sequence;
    }

    public static void main(String[] args)
    {
        new DotPlotUI();
    }
}


```