BUOYWAVES APPLICATION
Version 1.2

Jay A. Albrecht
Lead Forecaster,
National Weather Service Forecast Office
Seattle-Tacoma Washington

1. INTRODUCTION

BuoyWaves Version 1.2 is a desktop application used to thoroughly diagnose the state of the sea as reported by National Oceanic and Atmospheric Administration (NOAA) data buoys. BuoyWaves is a Java desktop application; it can be installed on any computer that has the Java 1.6 runtime environment and has access to high-speed internet. BuoyWaves has been tested on the Microsoft Windows XP, Vista and Windows 7 operating systems and on the Fedora core 4+ Linux distribution using both the Gnome and KDE desktop environments. While the application has not been tested on other operating systems, theoretically there should not be a problem running it on various Apple Macintosh OS, Linux, Unix, BSD, or Windows distributions -- assuming the Java 1.6 runtime environment is available and a high-speed internet connection is possible.

A network of NOAA data buoys surrounds the United States coastline; NOAA data buoys are deployed in the waters of Hawaii, the Bering Sea, the eastern North Pacific including the Gulf of Alaska, the Gulf of Mexico, the Caribbean Sea, the tropical and western North Atlantic Ocean, and the Great Lakes. In addition to measuring standard meteorological parameters, NOAA data buoys are equipped with accelerometers that measure buoy movement with respect to time. Through a complex Fourier analysis of the buoy's vertical and horizontal accelerations, distributions of wave energy with respect to frequency and direction are determined. These energy distributions can be used to compute significant wave height – the height of the highest 1/3 of the waves defining the sea state that is typically reported by a trained observer. An analysis of the energy distribution makes it easy to distinguish between wind wave and swell. Understanding the wave distribution is important in the forecast of the sea state, and wave distribution information is used to validate wave model output.

Potential BuoyWaves users include Marine Weather forecasters, Oceanographers, Incident Meteorolgists, civil engineers, and wave enthusiasts.

2. INSTALLATION AND CONFIGURATION

STEP 1:

Ensure that you have the Java 1.6 runtime installed on your computer. Verifying or installing the Java Virtual Machine is beyond the scope of this document. However, if you visit http://java.sun.com/javase/downloads/index.jsp you can download and receive instructions on how to install Java on your system. Mac users will need to visit http://www.apple.com/downloads/ and search for Java downloads.

STEP 2:

Create an application directory on your computer and dump the contents of the supplied BuoyWaves distribution into it. Include all files – libraries, the executable, and the supplied icon.

STEP 3:

How you set the program to run will depend on which operating system you use. On Windows systems that have the Java runtime installed, just create a desktop shortcut to the BuoyWaves.jar file in the application directory, then set the icon to the one supplied (SPECWV.ico) or to one of your choosing.

On Linux or Unix-based systems, assuming you have the environment variable entry for Java properly configured, create a shortut that employs the command:

java -jar BuoyWaves.jar

then set the icon as appropriate.

You can also run the application directly from a "terminal" or "command line" by typing ***[java]* -jar *[dir]*BuoyWaves.jar** where [java] is the complete path to the binary java run time program and [dir] is the complete path to the BuoyWaves application.

3. Using BuoyWaves

Start the program in the manner described above. The first time you start the application, you may need to enlarge the window to expose the entire GUI. Once this is done, the window state will be remembered by Java environment.

Figure 1 shows the BuoyWaves GUI. The following sections provide a description of each of the components.

COMPONENTS AND CONTROLS:

***Application Menu Bar**:* **LABEL A on Fig 1-** The Application Menu Bar provides a means to select a buoy of interest to be viewed. Buoys are grouped by area or ocean basin. Within each group, a menu separator is used to create subgroups by state or region within the basin. When a buoy is selected by clicking on a menu item, window labels are updated and information for the most recent observation from that buoy is displayed.

***Buoy Name Label**:* **LABEL B on Fig 1-** This is the buoy number, name, and location for the currently selected buoy as chosen using the Application Menu Bar.

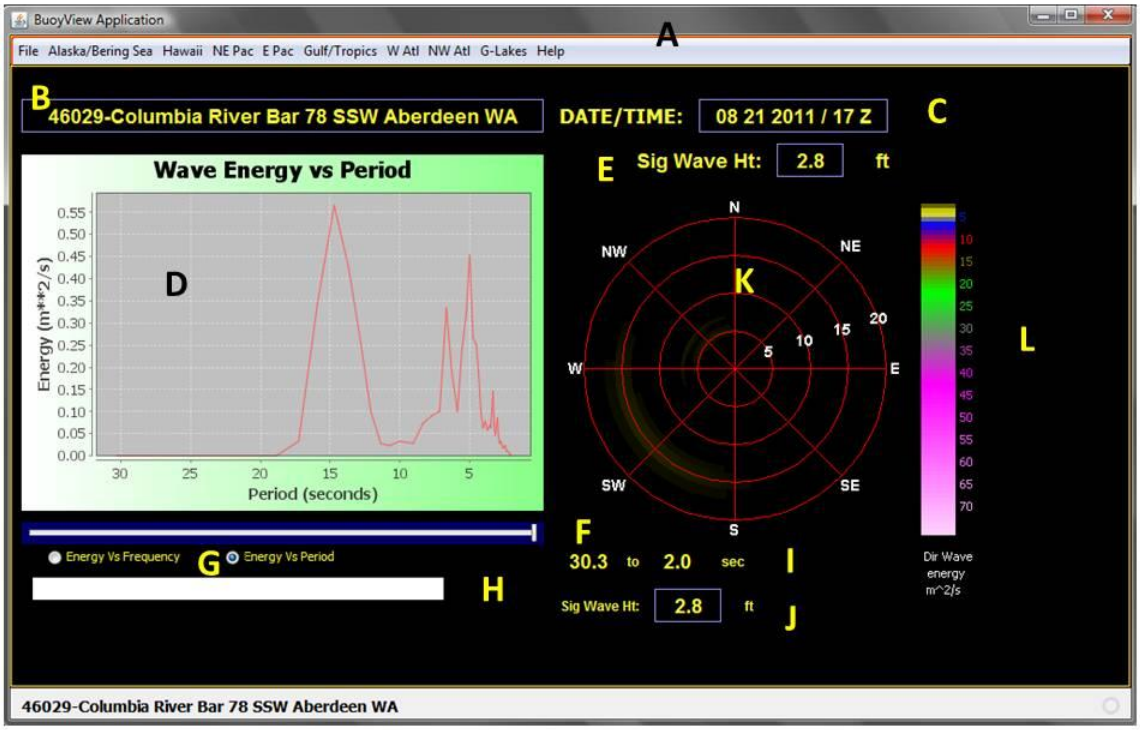

Figure 1. Sample view of BuoyWaves with components and controls labeled.

Observation Date-Time Label: **LABEL C on Fig 1-** Shows the date and time of the displayed observation. When a buoy is first selected from the Application menu bar, the most recent wave observation will be shown. When the Time-Control Slider Bar (**LABEL F on Fig 1**) is moved by clicking on the slider control or sliding the thumb on the control, older or more recent observations are selected and the date-time label and all output controls are updated. Up to 5 days of observations are available for each buoy.

Energy Chart:: **LABEL D on Fig 1-** The chart type that is shown is dependent on the selection of the Chart Type radio buttons (**LABEL G on Fig 1**). The default selection is Energy vs Frequency. In this case wave energy in meters squared per second (in a dynamic scale) is plotted with respect to wave frequency in hertz. Most users will want to select Energy vs Period as shown in Figure 1. Swell is high period (on the left) while wind wave is low period to the right

Signifant Wave Height Label: **LABEL E on Fig 1-** This label contains the total significant wave height for the buoy as computed for the entire wave energy spectrum. The significant wave height, defined as the height of the highest 1/3 of the waves, is the value typically reported by trained observers and is defined as

$$H_{1/3} = 4 \sqrt{\sum(E_f \ \Delta f)}$$

where $H_{1/3}$ is the significant wave height, E_f is the wave energy contained within an individual frequency bin, and Δf is the width of the frequency bin containing that wave energy.

Time Selection Slider Control: **LABEL F on Fig 1-** This control allows you to scroll through a set of 5 days of wave observations (the most recent observation is displayed by default when a buoy is selected). The most current observation is displayed when the slider thumb is all the way to the right, while the oldest observation is displayed when the slider thumb is all the way to the left. To change the displayed observation time, you can grab and drag the thumb on the slider control with the left mouse button or click the thumb and use the arrow keys.

Chart Type Radio Button Selection Controls: **LABEL G on Fig 1-** By default, the energy versus frequency radio buton is selected. These radio buttons determine the kind of energy chart and directional energy chart will be displayed (energy vs frequency or energy vs period).

Period or Frequency Band Selection Control: **LABEL H on Fig 1-** This control is the most complicated one to use in the application; it gives you the ability to visually compute significant wave height for the frequency or period band of your choosing. Left click and slowly drag the mouse on the control. This action dynamically sets the upper (lower) bounds of the period (frequency) band you are interested in. You will notice that the color of the control will change to blue to the left of your click point and yellow to the right. You'll also notice that the left Period (Frequency) band label (**LABEL I on Fig 1**) will be set to your selected value as you move the mouse. If the Chart Type Radio Button Selection is currently set to period space (radio buttons 2 and 3) the left value will be the higher period of the selected period band; if it is set to frequency space (radio button 1) it will be the lower frequency value of the selected frequency band. Now click using the right mouse button to the right of the previous selection. You will now be selecting your upper frequency (lower period) band value. As you make your selection, not only with the Period (Frequency) band labels change, the significant wave height for the selected band will be displayed in the lower Sig Wave Ht label (**LABEL J on Fig 1**). Your selections will not alter the total significant wave height shown by **LABEL E on Fig 1**. Your frequency band selection will be lost if you change observation times or buoys.

Selected Frequency Band Labels: **LABEL I on Fig 1-** These labels display the currently selected period (frequency) band for determining significant wave height over a user-defined period (frequency) interval. By default the label is set to the entire range reported by the buoy – and the significant wave height cooresponding to this band is equal to the total significant wave height for the buoy. These values change dynamically as the user left and right clicks and drags on the Period or Frequency Band Selection Control (**LABEL H on Fig 1**).

Signifant Wave Height for Selected Frequency Band Label: **LABEL J on Fig 1-** The value shown in this label cooresponds to the significant wave height for the user chosen period or frequency band. By default the significant wave height is equal to the value determined for the entire wave energy spectrum. This value is updated as the user selects period or frequency bands of interest using the Period or Frequency Band Selection Control (**LABEL H on Fig 1**).

Directional Wave Energy Plot: **LABEL K on Fig 1-** This plot shows the wave energy distribution with respect to direction and period (or frequency depending on the state of the Chart Type Radio Buttons. Period is easier to interpret, but remember that the buoy natively computes energy in frequency space. Swell will be farther from the origin than wind wave. The direction shown is the direction from which the energy is coming (direction here is in a meteorological sense instead of an oceanographic sense). Directional energy spread is rather large, so take a visual mean to obtain a direction for an individual swell or wind wave train. You will notice that with larger swell there is some energy coming from the opposite direction than the actual swell. This is a mathematical artifact of the Fourier transforms and the recomposition algorithms..

Directional Wave Energy Legend: **LABEL L on Fig 1-** This graphical label is simply a legend of colors used in the Directional Wave Energy Plot.

4. PROGRAM SPECIFICATIONS

This simple application was built using:

Netbeans 7.0.1 Integrated Development Environment.
http://www.netbeans.org

5. PROGRAM SOUCE CODE

-- SpectralOb.java

```
package buoywaves;

/**
 *
 * @author Jay Albrecht, NOAA/National Weather Service, Seattle WA
 *
 * This class defines a single spectral piece of a wave observation
 *   from a NOAA data buoy.  Each spectral piece contains a frequency in hz,
 *   the frequency bandwidth in hz, the spectral density (energy) contained
 *   within the spectral band in m^2/s, r1 and r2 are the first and second
 *   normalized polar coordinates of the Fourier coefficients and are
 *   nondimensional, and alpha1 and alpha2 are the mean and pricipal wave directions
 *   in terms of Longuet-Higgins Fourier Coefficients.
```

```
 */
public class SpectralOb {
    private double frequency;         // Frequency band - hz
    private double period;            // central period - sec
    private double spectralWidth;     // width of frequency bin - hz
    private double energy;            // spectral wave density (m^2/s)
    private double r1;                // first normalized polar coord of Fourier coef
    private double r2;                // 2nd normalized polar coord of Fourier coef
    private int alpha1;               // mean wave direction (Fourier coef)
    private int alpha2;               // principal wave direction (Fourier coef)
    private double sigWaveHeight;     // contribution to sig wave height (ft)

    public SpectralOb(String L) {

        // Constructor, takes a line of data, each line contains a single frequency.
        //  The constructor parses the data into the components and computes
        //  the significant wave height contribution for this frequency band.

        this.frequency = Double.parseDouble(L.substring(0, 5).trim());
        this.period = 1./this.frequency;
        String SpectralWidth = L.substring(5);
        this.spectralWidth = Double.parseDouble(SpectralWidth.substring(0, 6).trim());
        String Energy = SpectralWidth.substring(6);
        this.energy = Double.parseDouble(Energy.substring(0, 8).trim());
        int b1 = Energy.indexOf(" ", 3);
        String R1 = Energy.substring(b1);
        this.r1 = Double.parseDouble(R1.substring(0, 7).trim());
        b1 = R1.indexOf(" ", 5);
        String R2 = R1.substring(b1);
        this.r2 = Double.parseDouble(R2.substring(0, 7).trim());
        b1 = R2.indexOf(" ", 5);
        String Alpha1 = R2.substring(b1);
        this.alpha1 = Integer.parseInt(Alpha1.substring(0, 4).trim());
        b1 = Alpha1.indexOf(" ", 3);
        String Alpha2 = Alpha1.substring(b1);
        this.alpha2 = Integer.parseInt(Alpha2.trim());
        this.sigWaveHeight = 3.28084 *
                (4.0 * Math.sqrt(this.energy * this.spectralWidth));
    }

    // Public getter methods for the various members of the class.

    public double getFrequency() { return this.frequency; }
    public double getPeriod() { return this.period; }
    public double getSpectralWidth() { return this.spectralWidth; }
    public double getEnergy() { return this.energy; }
    public double getR1() { return this.r1; }
    public double getR2() { return this.r2; }
    public int getAlpha1() { return this.alpha1; }
    public int getAlpha2() { return this.alpha2; }
    public double getSigWaveHeight() { return this.sigWaveHeight; }
}
```

-- *BuoyOb.java*

```
package buoywaves;

import java.util.GregorianCalendar;
import java.util.ArrayList;
import java.awt.Color;

/**
 *
 * @author Jay Albrecht, NOAA/National Weather Service, Seattle WA
 *  This class defines a single complete wave observation from a NOAA data buoy.
 *   Each  observation contains a date time group, the number of spectral bands
 *   that the observation is composed of, and an array lift of individual
 *   spectral observations that define the individual spectral components that
 *   make up the complete observation.  The constructor takes an array list of
 *   lines of data (in the form of strings). The first line contains the date/time
 *   group and spectral band information while the subsequent strings are individual
 *   spectral band data. Each spectral band is a SpectralOb that defines individual
 *   bins of frequency, the bandwidth of the bin, spectral wave density, and
 *   various Longuet-Higgins directional Fourier Coefficients.
 */
public class BuoyOb {

    private GregorianCalendar DTG;                 // Observation date/time group
```

```
private int bands;                              // number of frequency bands
private ArrayList<SpectralOb> spectralData
        = new ArrayList<SpectralOb>();          // Individual spectral bands.

public BuoyOb (BuoyOb O) {                      // Copy constructor
    this.DTG = O.DTG;
    this.bands = O.bands;
    this.spectralData.clear();
    for (int i = 0; i < O.spectralData.size(); i++) {
       SpectralOb s = O.spectralData.get(i);
       this.spectralData.add(s);
    }
}

public BuoyOb (ArrayList<String> RAW) {

    // The first element of the array list contains date/time information
    //  and the number of bands in the wave observation.  These are decoded
    //  and placed into the DTG and bands elements.

    this.spectralData.clear();
    String Header = RAW.get(0);
    int year = Integer.parseInt(Header.substring(0, 4).trim());
    int mon = Integer.parseInt(Header.substring(5, 7).trim()) - 1;
    int day = Integer.parseInt(Header.substring(8, 10).trim());
    int hour = Integer.parseInt(Header.substring(11, 13).trim());
    int minute = Integer.parseInt(Header.substring(14, 16).trim());
    this.DTG = new GregorianCalendar(year, mon, day, hour, minute);

    this.bands = Integer.parseInt(Header.substring(17).trim());

    //  Now get the remaining elements to construct the individual
    //  spectral bands and place in the spectralData array list.

    for (int i = 1; i < RAW.size(); i++) {

        String SpecObLine = RAW.get(i);
        SpectralOb Ob = new SpectralOb(SpecObLine);
        spectralData.add(Ob);
    }
}

//  Public getter methods for getting dates, band number, and spectral data.


public GregorianCalendar getDate() { return this.DTG; }
public int getBands() { return this.bands; }
public ArrayList<SpectralOb> getSpectralData() { return spectralData; }
public double getDirEnergyPd(double per, double dir) {
    // Given a period in seconds and direction in deg, this function computes
    //  and returns the directional energy for that point. The value that is
    //  returned will be an approximate value -- one that is representative
    //  for the frequency/period bin for which the request is made, and for the
    //  directional bin as well. This should be good enough as it is well
    //  within instrument error. If the directional energy value is null
    //  -- either because it is outside the rhelm of the instrument package
    //  of the particular buou, or missing, or the buoy does not report
    //  direction, the returned value is zero.

    double dEnergy = 0.0;

    // First check if the band is within range. if not return 0.0
    if (per > this.spectralData.get(0).getPeriod()) return 0.0;
    if (per < this.spectralData.get(this.spectralData.size()-1).getPeriod()) return
        0.0;

    // Now determine which band to pull in

    int bandnum = 0;

    for (int i = this.spectralData.size() - 1; i >= 0; i--) {

        if (per > this.spectralData.get(i).getPeriod()) bandnum = i;

    }

    // retrieve r1, r2, alpha1 and alpha2 for the found band
    double r1 = this.spectralData.get(bandnum).getR1();
    double r2 = this.spectralData.get(bandnum).getR2();
    int a1 = this.spectralData.get(bandnum).getAlpha1();
```

```
    int a2 = this.spectralData.get(bandnum).getAlpha2();
    double Omni = this.spectralData.get(bandnum).getEnergy();

    // if any of the fourier coeficients are null return 0.0
    if (r1 > 990.0 || r2 > 990.0 || a1 > 360 || a2 > 360) return 0.0;
    if (Omni < 0.01) return 0.0;

    if ((dir - a1) > 180) a1 = a1 + 360;
    if ((dir - a1) < -180) dir = dir + 360;

    double D1 = (Math.abs(dir - a1)) * Math.PI/180.0;
    double D2 = (Math.abs(dir - a2)) * Math.PI/180.0;

    // Finally, compute directional energy
    dEnergy = Omni * (1 / Math.PI) * (0.5 +
            r1 * Math.cos(D1) +
            r2 * Math.cos(2.0 * D2));

    return dEnergy;
}
public double getDirEnergyFr(double freq, double dir) {
    // Given a frequency in hz and direction in deg, this function computes
    //  and returns the directional energy for that point. The value that is
    //  returned will be an approximate value -- one that is representative
    //  for the frequency bin for which the request is made, and for the
    //  directional bin as well. This should be good enough as it is well
    //  within instrument error. If the directional energy value is null
    //  -- either because it is outside the rhelm of the instrument package
    //  of the particular buou, or missing, or the buoy does not report
    //  direction, the returned value is zero.

    double dEnergy = 0.0;

    // First check if the frequency band is within range. if not return 0.0
    if (freq < this.spectralData.get(0).getFrequency()) return 0.0;
    if (freq > this.spectralData.get(this.spectralData.size()-1).getFrequency())
          return 0.0;

    // Now determine which band to pull in

    int bandnum = 0;

    for (int i = 0; i < this.spectralData.size(); i++) {

        if (freq > this.spectralData.get(i).getFrequency()) bandnum = i;

    }

    // retrieve r1, r2, alpha1 and alpha2 for the found band
    double r1 = this.spectralData.get(bandnum).getR1();
    double r2 = this.spectralData.get(bandnum).getR2();
    int a1 = this.spectralData.get(bandnum).getAlpha1();
    int a2 = this.spectralData.get(bandnum).getAlpha2();
    double Omni = this.spectralData.get(bandnum).getEnergy();

    if ((dir - a1) > 180) a1 = a1 + 360;
    if ((dir - a1) < -180) dir = dir + 360;

    // if any of the fourier coeficients are null return 0.0
    if (r1 > 990.0 || r2 > 990.0 || a1 > 360 || a2 > 360) return 0.0;
    if (Omni < 0.01) return 0.0;

    double D1 = (Math.abs(dir - a1)) * Math.PI/180.0;
    double D2 = (Math.abs(dir - a2)) * Math.PI/180.0;

    // Finally, compute directional energy
    dEnergy = Omni * (1 / Math.PI) * (0.5 +
            r1 * Math.cos(D1) +
            r2 * Math.cos(2.0 * D2));

    return dEnergy;
}
public Color getColor(double E) {
    // if there is no energy, return black
    if (E < 0.1) return new Color(0, 0, 0);

    // if the energy is less than 2.55 m2/s, use yellow tones
    if (E < 2.56) {
       int x = (int)(E * 100.0);
       return new Color(x, x, 0);
```

```
        }

        // if E is now between 2.56 and 5.12 trend to blue
        if (E < 5.12) {
           int r = 256 - (int)((E - 2.55) * 100.0);
           int g = 256 - (int)((E - 2.55) * 100.0);
           int b = (int)((E - 2.56) * 100.0);
           return new Color(r, g, b);
        }

        // if E is now between 5.12 and 10.24 trend to red
        if (E < 10.24) {
           int r = (int)((E - 5.12) * 50.0);
           int g = 0;
           int b = 255 - (int)((E - 5.12) * 50.0);
           return new Color(r, g, b);
        }

        // if E is now between 10.24 and 20.48 trend to green
        if (E < 20.48) {
            int r = 255 - (int)((E - 10.24) * 25.0);
            int g = (int)((E - 10.24) * 25.0);
            int b = 0;
            return new Color(r, g, b);
        }

        // if E is now between 20.48 and 40.96 trend to cyan
        if (E < 40.96) {
            int r = (int)((E - 20.48) * 12.5);
            int g = 255 - (int)((E - 20.48) * 12.5);
            int b = (int)((E - 20.48) * 12.5);
            return new Color(r, g, b);
        }

        // if E is now between 40.96 and 81.92 trend to white
        if (E < 81.92) {
            int r = 255;
            int g = (int)((E - 40.96) * 6.25);
            int b = 255;
            return new Color(r, g, b);
        }
        if (E < 163.84) {
            int x = 255 - (int)((E - 81.92) * 3.125);
            return new Color(x, x, x);
        }

        return new Color(255, 255, 255);


     }

  }
```

-- *BuoyWavesView.java*

```
/*
 * BuoyWavesView.java
 */

package buoywaves;

import org.jdesktop.application.Action;
import org.jdesktop.application.ResourceMap;
import org.jdesktop.application.SingleFrameApplication;
import org.jdesktop.application.FrameView;
import org.jdesktop.application.TaskMonitor;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.Timer;
import javax.swing.Icon;
import javax.swing.JDialog;
import javax.swing.JFrame;
import java.util.ArrayList;
import java.net.*;
import java.io.*;
import java.awt.*;
import org.jfree.chart.ChartFactory;
import org.jfree.chart.ChartPanel;
import org.jfree.chart.JFreeChart;
import org.jfree.chart.plot.PlotOrientation;
```

```
import org.jfree.data.xy.XYDataset;
import org.jfree.data.xy.XYSeries;
import org.jfree.data.xy.XYSeriesCollection;
import java.awt.image.BufferedImage;

/**
 * The application's main frame.
 */
public class BuoyWavesView extends FrameView {

    // Global variables:
    //  CurrentBuoyObs is an array list that holds up to 5 days of hourly
    //   observations for the currently selected buoy
    //  O is an individual buoy observation for the currently selected hour
    //  CurrentBuoy is a String that holds the number of the currently selected buoy
    //  BuoyLabel is a String that holds the long name of the currently selected buoy
    //  LeftPoint and RightPoint are integer values that hold the current
    //   mouse click locations on the cSel canvas that define user selections
    //   of frequency or period bands of interst.

    private ArrayList<BuoyOb> CurrentBuoyObs = new ArrayList<BuoyOb>();
    private BuoyOb O;
    private String CurrentBuoy = "";
    private String BuoyLabel = "";
    private int LeftPoint = 0;
    private int RightPoint = 370;
    private BufferedImage dirPlot;
   // private BufferedImage blankImage = new BufferedImage(200, 200, 1);


    public BuoyWavesView(SingleFrameApplication app) {
        super(app);

        initComponents();

        // status bar initialization - message timeout, idle icon and busy animation, etc
        ResourceMap resourceMap = getResourceMap();
        int messageTimeout = resourceMap.getInteger("StatusBar.messageTimeout");
        messageTimer = new Timer(messageTimeout, new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                statusMessageLabel.setText("");
            }
        });
        messageTimer.setRepeats(false);
        int busyAnimationRate = resourceMap.getInteger("StatusBar.busyAnimationRate");
        for (int i = 0; i < busyIcons.length; i++) {
            busyIcons[i] = resourceMap.getIcon("StatusBar.busyIcons[" + i + "]");
        }
        busyIconTimer = new Timer(busyAnimationRate, new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                busyIconIndex = (busyIconIndex + 1) % busyIcons.length;
                statusAnimationLabel.setIcon(busyIcons[busyIconIndex]);
            }
        });
        idleIcon = resourceMap.getIcon("StatusBar.idleIcon");
        statusAnimationLabel.setIcon(idleIcon);
        progressBar.setVisible(false);

        // connecting action tasks to status bar via TaskMonitor
        TaskMonitor taskMonitor = new TaskMonitor(getApplication().getContext());
        taskMonitor.addPropertyChangeListener(new java.beans.PropertyChangeListener() {
            public void propertyChange(java.beans.PropertyChangeEvent evt) {
                String propertyName = evt.getPropertyName();
                if ("started".equals(propertyName)) {
                    if (!busyIconTimer.isRunning()) {
                        statusAnimationLabel.setIcon(busyIcons[0]);
                        busyIconIndex = 0;
                        busyIconTimer.start();
                    }
                    progressBar.setVisible(true);
                    progressBar.setIndeterminate(true);
                } else if ("done".equals(propertyName)) {
                    busyIconTimer.stop();
                    statusAnimationLabel.setIcon(idleIcon);
                    progressBar.setVisible(false);
                    progressBar.setValue(0);
                } else if ("message".equals(propertyName)) {
                    String text = (String)(evt.getNewValue());
                    statusMessageLabel.setText((text == null) ? "" : text);
                    messageTimer.restart();
```

```
            } else if (”progress”.equals(propertyName)) {
                int value = (Integer)(evt.getNewValue());
                progressBar.setVisible(true);
                progressBar.setIndeterminate(false);
                progressBar.setValue(value);
            }
        }
    });
}

@Action
public void showAboutBox() {
    if (aboutBox == null) {
        JFrame mainFrame = BuoyWavesApp.getApplication().getMainFrame();
        aboutBox = new BuoyWavesAboutBox(mainFrame);
        aboutBox.setLocationRelativeTo(mainFrame);
    }
    BuoyWavesApp.getApplication().show(aboutBox);
}

/** 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.
 */
@SuppressWarnings(”unchecked”)
// <editor-fold defaultstate=”collapsed” desc=”Generated Code”>
private void initComponents() {

    mainPanel = new javax.swing.JPanel();
    pnlEnergy = new javax.swing.JPanel();
    jLabel1 = new javax.swing.JLabel();
    lblDTG = new javax.swing.JLabel();
    jLabel2 = new javax.swing.JLabel();
    lblSigWaveHt = new javax.swing.JLabel();
    jLabel3 = new javax.swing.JLabel();
    rbFrequency = new javax.swing.JRadioButton();
    rbPeriod = new javax.swing.JRadioButton();
    sliderTime = new javax.swing.JSlider();
    cSel = new java.awt.Canvas();
    lblLowerVal = new javax.swing.JLabel();
    lblUnits = new javax.swing.JLabel();
    lblUpperVal = new javax.swing.JLabel();
    lblLowerVal2 = new javax.swing.JLabel();
    jLabel4 = new javax.swing.JLabel();
    lblSubHt = new javax.swing.JLabel();
    jLabel5 = new javax.swing.JLabel();
    pnlDir = new javax.swing.JPanel() {
        public void paint(Graphics g) {
            PaintPnlDir(g);
        }
    };
    pnlLegend = new java.awt.Panel();
    lblBuoyName = new javax.swing.JLabel();
    menuBar = new javax.swing.JMenuBar();
    javax.swing.JMenu fileMenu = new javax.swing.JMenu();
    javax.swing.JMenuItem exitMenuItem = new javax.swing.JMenuItem();
    jSeparator1 = new javax.swing.JSeparator();
    mnuAlaska = new javax.swing.JMenu();
    mnu46070 = new javax.swing.JMenuItem();
    mnu46035 = new javax.swing.JMenuItem();
    mnu46073 = new javax.swing.JMenuItem();
    jSeparator2 = new javax.swing.JSeparator();
    mnu46071 = new javax.swing.JMenuItem();
    mnu46072 = new javax.swing.JMenuItem();
    mnu46075 = new javax.swing.JMenuItem();
    jSeparator3 = new javax.swing.JSeparator();
    mnu46077 = new javax.swing.JMenuItem();
    mnu46066 = new javax.swing.JMenuItem();
    mnu46078 = new javax.swing.JMenuItem();
    mnu46080 = new javax.swing.JMenuItem();
    mnu46001 = new javax.swing.JMenuItem();
    mnu46082 = new javax.swing.JMenuItem();
    mnu46085 = new javax.swing.JMenuItem();
    jSeparator4 = new javax.swing.JSeparator();
    mnu46083 = new javax.swing.JMenuItem();
    mnu46084 = new javax.swing.JMenuItem();
    mnuHawaii = new javax.swing.JMenu();
    mnu51000 = new javax.swing.JMenuItem();
    mnu51001 = new javax.swing.JMenuItem();
```

```
mnu51003 = new javax.swing.JMenuItem();
mnu51002 = new javax.swing.JMenuItem();
mnu51004 = new javax.swing.JMenuItem();
mnuNW = new javax.swing.JMenu();
mnu46005 = new javax.swing.JMenuItem();
mnu46089 = new javax.swing.JMenuItem();
mnu46002 = new javax.swing.JMenuItem();
mnu46006 = new javax.swing.JMenuItem();
mnu46059 = new javax.swing.JMenuItem();
jSeparator5 = new javax.swing.JSeparator();
mnu46088 = new javax.swing.JMenuItem();
mnu46087 = new javax.swing.JMenuItem();
mnu46041 = new javax.swing.JMenuItem();
jSeparator6 = new javax.swing.JSeparator();
mnu46029 = new javax.swing.JMenuItem();
mnu46050 = new javax.swing.JMenuItem();
mnu46015 = new javax.swing.JMenuItem();
jSeparator7 = new javax.swing.JSeparator();
mnu46027 = new javax.swing.JMenuItem();
mnu46022 = new javax.swing.JMenuItem();
mnu46014 = new javax.swing.JMenuItem();
mnu46013 = new javax.swing.JMenuItem();
mnu46026 = new javax.swing.JMenuItem();
mnu46012 = new javax.swing.JMenuItem();
mnu46042 = new javax.swing.JMenuItem();
mnuSW = new javax.swing.JMenu();
mnu46028 = new javax.swing.JMenuItem();
mnu46011 = new javax.swing.JMenuItem();
mnu46023 = new javax.swing.JMenuItem();
mnu46063 = new javax.swing.JMenuItem();
mnu46069 = new javax.swing.JMenuItem();
mnu46047 = new javax.swing.JMenuItem();
jSeparator8 = new javax.swing.JSeparator();
mnu46054 = new javax.swing.JMenuItem();
mnu46053 = new javax.swing.JMenuItem();
mnu46025 = new javax.swing.JMenuItem();
mnu46086 = new javax.swing.JMenuItem();
mnuGulf = new javax.swing.JMenu();
mnu42020 = new javax.swing.JMenuItem();
mnu42019 = new javax.swing.JMenuItem();
mnu42035 = new javax.swing.JMenuItem();
jSeparator9 = new javax.swing.JSeparator();
mnu42002 = new javax.swing.JMenuItem();
mnu42055 = new javax.swing.JMenuItem();
mnu42001 = new javax.swing.JMenuItem();
mnu42003 = new javax.swing.JMenuItem();
jSeparator10 = new javax.swing.JSeparator();
mnu42007 = new javax.swing.JMenuItem();
mnu42012 = new javax.swing.JMenuItem();
mnu42040 = new javax.swing.JMenuItem();
jSeparator11 = new javax.swing.JSeparator();
mnu42039 = new javax.swing.JMenuItem();
mnu42036 = new javax.swing.JMenuItem();
mnu42080 = new javax.swing.JMenuItem();
jSeparator12 = new javax.swing.JSeparator();
mnu42056 = new javax.swing.JMenuItem();
mnu42057 = new javax.swing.JMenuItem();
mnu42058 = new javax.swing.JMenuItem();
mnu42059 = new javax.swing.JMenuItem();
mnu42060 = new javax.swing.JMenuItem();
jSeparator13 = new javax.swing.JSeparator();
mnu41043 = new javax.swing.JMenuItem();
mnu41040 = new javax.swing.JMenuItem();
mnu41041 = new javax.swing.JMenuItem();
mnu41044 = new javax.swing.JMenuItem();
mnuWAtl = new javax.swing.JMenu();
mnu41046 = new javax.swing.JMenuItem();
mnu41047 = new javax.swing.JMenuItem();
mnu41048 = new javax.swing.JMenuItem();
mnu41049 = new javax.swing.JMenuItem();
jSeparator14 = new javax.swing.JSeparator();
mnu41010 = new javax.swing.JMenuItem();
mnu41009 = new javax.swing.JMenuItem();
mnu41012 = new javax.swing.JMenuItem();
mnu41008 = new javax.swing.JMenuItem();
jSeparator15 = new javax.swing.JSeparator();
mnu41001 = new javax.swing.JMenuItem();
mnu41002 = new javax.swing.JMenuItem();
mnu41013 = new javax.swing.JMenuItem();
jSeparator16 = new javax.swing.JSeparator();
```

```
        mnu41004 = new javax.swing.JMenuItem();
        mnu41035 = new javax.swing.JMenuItem();
        mnu41036 = new javax.swing.JMenuItem();
        mnu41025 = new javax.swing.JMenuItem();
        mnuNWAtl = new javax.swing.JMenu();
        mnu44014 = new javax.swing.JMenuItem();
        mnu44009 = new javax.swing.JMenuItem();
        jSeparator17 = new javax.swing.JSeparator();
        mnu44004 = new javax.swing.JMenuItem();
        mny44066 = new javax.swing.JMenuItem();
        mnu44008 = new javax.swing.JMenuItem();
        mnu44018 = new javax.swing.JMenuItem();
        mnu44005 = new javax.swing.JMenuItem();
        mnu44011 = new javax.swing.JMenuItem();
        jSeparator18 = new javax.swing.JSeparator();
        mnu44025 = new javax.swing.JMenuItem();
        mnu44017 = new javax.swing.JMenuItem();
        mnu44065 = new javax.swing.JMenuItem();
        mnu44013 = new javax.swing.JMenuItem();
        mnu44020 = new javax.swing.JMenuItem();
        mnu44007 = new javax.swing.JMenuItem();
        mnu44027 = new javax.swing.JMenuItem();
        mnuLakes = new javax.swing.JMenu();
        mnu45006 = new javax.swing.JMenuItem();
        mnu45001 = new javax.swing.JMenuItem();
        mnu45004 = new javax.swing.JMenuItem();
        jSeparator19 = new javax.swing.JSeparator();
        mnu45002 = new javax.swing.JMenuItem();
        mnu45007 = new javax.swing.JMenuItem();
        jSeparator20 = new javax.swing.JSeparator();
        mnu45003 = new javax.swing.JMenuItem();
        mnu45008 = new javax.swing.JMenuItem();
        jSeparator21 = new javax.swing.JSeparator();
        mnu45005 = new javax.swing.JMenuItem();
        jSeparator22 = new javax.swing.JSeparator();
        mnu45012 = new javax.swing.JMenuItem();
        javax.swing.JMenu helpMenu = new javax.swing.JMenu();
        javax.swing.JMenuItem aboutMenuItem = new javax.swing.JMenuItem();
        statusPanel = new javax.swing.JPanel();
        javax.swing.JSeparator statusPanelSeparator = new javax.swing.JSeparator();
        statusMessageLabel = new javax.swing.JLabel();
        statusAnimationLabel = new javax.swing.JLabel();
        progressBar = new javax.swing.JProgressBar();
        btnGrpEnergyPlotControls = new javax.swing.ButtonGroup();

        org.jdesktop.application.ResourceMap resourceMap =
org.jdesktop.application.Application.getInstance(buoywaves.BuoyWavesApp.class).getContext
().getResourceMap(BuoyWavesView.class);
        mainPanel.setBackground(resourceMap.getColor("mainPanel.background")); // NOI18N

mainPanel.setBorder(javax.swing.BorderFactory.createEtchedBorder(resourceMap.getColor("ma
inPanel.border.highlightColor"), resourceMap.getColor("mainPanel.border.shadowColor")));
// NOI18N
        mainPanel.setMaximumSize(new java.awt.Dimension(600, 400));
        mainPanel.setMinimumSize(new java.awt.Dimension(600, 400));
        mainPanel.setName("mainPanel"); // NOI18N
        mainPanel.setPreferredSize(new java.awt.Dimension(600, 400));
        mainPanel.setLayout(new org.netbeans.lib.awtextra.AbsoluteLayout());

        pnlEnergy.setBackground(resourceMap.getColor("pnlEnergy.background")); // NOI18N
        pnlEnergy.setBorder(new
javax.swing.border.SoftBevelBorder(javax.swing.border.BevelBorder.RAISED,
resourceMap.getColor("pnlEnergy.border.highlightOuterColor"),
resourceMap.getColor("pnlEnergy.border.highlightInnerColor"),
resourceMap.getColor("pnlEnergy.border.shadowOuterColor"),
resourceMap.getColor("pnlEnergy.border.shadowInnerColor"))); // NOI18N
        pnlEnergy.setForeground(resourceMap.getColor("pnlEnergy.foreground")); // NOI18N
        pnlEnergy.setMaximumSize(new java.awt.Dimension(470, 320));
        pnlEnergy.setMinimumSize(new java.awt.Dimension(470, 320));
        pnlEnergy.setName("pnlEnergy"); // NOI18N
        pnlEnergy.setPreferredSize(new java.awt.Dimension(470, 320));

        javax.swing.GroupLayout pnlEnergyLayout = new javax.swing.GroupLayout(pnlEnergy);
        pnlEnergy.setLayout(pnlEnergyLayout);
        pnlEnergyLayout.setHorizontalGroup(

pnlEnergyLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGap(0, 464, Short.MAX_VALUE)
        );
        pnlEnergyLayout.setVerticalGroup(
```

```
pnlEnergyLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGap(0, 314, Short.MAX_VALUE)
        );

        mainPanel.add(pnlEnergy, new org.netbeans.lib.awtextra.AbsoluteConstraints(10,
80, 470, 320));

        jLabel1.setFont(resourceMap.getFont("jLabel1.font")); // NOI18N
        jLabel1.setForeground(resourceMap.getColor("jLabel1.foreground")); // NOI18N
        jLabel1.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        jLabel1.setText(resourceMap.getString("jLabel1.text")); // NOI18N
        jLabel1.setName("jLabel1"); // NOI18N
        mainPanel.add(jLabel1, new org.netbeans.lib.awtextra.AbsoluteConstraints(490, 30,
120, 30));

        lblDTG.setFont(resourceMap.getFont("lblDTG.font")); // NOI18N
        lblDTG.setForeground(resourceMap.getColor("lblDTG.foreground")); // NOI18N
        lblDTG.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        lblDTG.setText(resourceMap.getString("lblDTG.text")); // NOI18N
        lblDTG.setBorder(javax.swing.BorderFactory.createLineBorder(new
java.awt.Color(153, 153, 255)));
        lblDTG.setName("lblDTG"); // NOI18N
        mainPanel.add(lblDTG, new org.netbeans.lib.awtextra.AbsoluteConstraints(620, 30,
170, 30));

        jLabel2.setFont(resourceMap.getFont("jLabel2.font")); // NOI18N
        jLabel2.setForeground(resourceMap.getColor("jLabel2.foreground")); // NOI18N
        jLabel2.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        jLabel2.setText(resourceMap.getString("jLabel2.text")); // NOI18N
        jLabel2.setName("jLabel2"); // NOI18N
        mainPanel.add(jLabel2, new org.netbeans.lib.awtextra.AbsoluteConstraints(560, 70,
120, 30));

        lblSigWaveHt.setFont(resourceMap.getFont("lblSigWaveHt.font")); // NOI18N
        lblSigWaveHt.setForeground(resourceMap.getColor("lblSigWaveHt.foreground")); //
NOI18N
        lblSigWaveHt.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        lblSigWaveHt.setBorder(javax.swing.BorderFactory.createLineBorder(new
java.awt.Color(153, 153, 255)));
        lblSigWaveHt.setName("lblSigWaveHt"); // NOI18N
        mainPanel.add(lblSigWaveHt, new
org.netbeans.lib.awtextra.AbsoluteConstraints(690, 70, 60, 30));

        jLabel3.setFont(resourceMap.getFont("jLabel3.font")); // NOI18N
        jLabel3.setForeground(resourceMap.getColor("jLabel3.foreground")); // NOI18N
        jLabel3.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        jLabel3.setText(resourceMap.getString("jLabel3.text")); // NOI18N
        jLabel3.setName("jLabel3"); // NOI18N
        mainPanel.add(jLabel3, new org.netbeans.lib.awtextra.AbsoluteConstraints(770, 70,
30, 30));

        rbFrequency.setBackground(resourceMap.getColor("rbFrequency.background")); //
NOI18N
        btnGrpEnergyPlotControls.add(rbFrequency);
        rbFrequency.setForeground(resourceMap.getColor("rbFrequency.foreground")); //
NOI18N
        rbFrequency.setSelected(true);
        rbFrequency.setText(resourceMap.getString("rbFrequency.text")); // NOI18N
        rbFrequency.setName("rbFrequency"); // NOI18N
        rbFrequency.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                ChangeChartType(evt);
            }
        });
        mainPanel.add(rbFrequency, new org.netbeans.lib.awtextra.AbsoluteConstraints(30,
430, -1, -1));

        rbPeriod.setBackground(resourceMap.getColor("rbPeriod.background")); // NOI18N
        btnGrpEnergyPlotControls.add(rbPeriod);
        rbPeriod.setForeground(resourceMap.getColor("rbPeriod.foreground")); // NOI18N
        rbPeriod.setText(resourceMap.getString("rbPeriod.text")); // NOI18N
        rbPeriod.setName("rbPeriod"); // NOI18N
        rbPeriod.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                ChangeChartType(evt);
            }
        });
        mainPanel.add(rbPeriod, new org.netbeans.lib.awtextra.AbsoluteConstraints(190,
430, -1, -1));
```

```
        sliderTime.setBackground(resourceMap.getColor("sliderTime.background")); //
NOI18N
        sliderTime.setForeground(resourceMap.getColor("sliderTime.foreground")); //
NOI18N
        sliderTime.setValue(100);
        sliderTime.setBorder(javax.swing.BorderFactory.createEtchedBorder());
        sliderTime.setDoubleBuffered(true);
        sliderTime.setMaximumSize(new java.awt.Dimension(400, 27));
        sliderTime.setMinimumSize(new java.awt.Dimension(400, 27));
        sliderTime.setName("sliderTime"); // NOI18N
        sliderTime.setPreferredSize(new java.awt.Dimension(400, 27));
        sliderTime.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mousePressed(java.awt.event.MouseEvent evt) {
                sliderTimeMouseHandler(evt);
            }
            public void mouseReleased(java.awt.event.MouseEvent evt) {
                sliderTimeMouseHandler(evt);
            }
        });
        sliderTime.addMouseMotionListener(new java.awt.event.MouseMotionAdapter() {
            public void mouseDragged(java.awt.event.MouseEvent evt) {
                sliderTimeMouseHandler(evt);
            }
        });
        sliderTime.addKeyListener(new java.awt.event.KeyAdapter() {
            public void keyPressed(java.awt.event.KeyEvent evt) {
                ArrowKeys(evt);
            }
        });
        mainPanel.add(sliderTime, new org.netbeans.lib.awtextra.AbsoluteConstraints(10,
410, 470, 20));

        cSel.setBackground(resourceMap.getColor("cSel.background")); // NOI18N
        cSel.setName("cSel"); // NOI18N
        cSel.addMouseMotionListener(new java.awt.event.MouseMotionAdapter() {
            public void mouseDragged(java.awt.event.MouseEvent evt) {
                SetBandLimits(evt);
            }
        });
        mainPanel.add(cSel, new org.netbeans.lib.awtextra.AbsoluteConstraints(20, 460,
370, 20));

        lblLowerVal.setFont(resourceMap.getFont("lblLowerVal.font")); // NOI18N
        lblLowerVal.setForeground(resourceMap.getColor("lblLowerVal.foreground")); //
NOI18N
        lblLowerVal.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        lblLowerVal.setText(resourceMap.getString("lblLowerVal.text")); // NOI18N
        lblLowerVal.setName("lblLowerVal"); // NOI18N
        mainPanel.add(lblLowerVal, new org.netbeans.lib.awtextra.AbsoluteConstraints(500,
430, 40, 30));

        lblUnits.setFont(resourceMap.getFont("lblUnits.font")); // NOI18N
        lblUnits.setForeground(resourceMap.getColor("lblUnits.foreground")); // NOI18N
        lblUnits.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        lblUnits.setText(resourceMap.getString("lblUnits.text")); // NOI18N
        lblUnits.setName("lblUnits"); // NOI18N
        mainPanel.add(lblUnits, new org.netbeans.lib.awtextra.AbsoluteConstraints(630,
430, 40, 30));

        lblUpperVal.setFont(resourceMap.getFont("lblUpperVal.font")); // NOI18N
        lblUpperVal.setForeground(resourceMap.getColor("lblUpperVal.foreground")); //
NOI18N
        lblUpperVal.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        lblUpperVal.setText(resourceMap.getString("lblUpperVal.text")); // NOI18N
        lblUpperVal.setName("lblUpperVal"); // NOI18N
        mainPanel.add(lblUpperVal, new org.netbeans.lib.awtextra.AbsoluteConstraints(580,
430, 40, 30));

        lblLowerVal2.setFont(resourceMap.getFont("lblLowerVal2.font")); // NOI18N
        lblLowerVal2.setForeground(resourceMap.getColor("lblLowerVal2.foreground")); //
NOI18N
        lblLowerVal2.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        lblLowerVal2.setText(resourceMap.getString("lblLowerVal2.text")); // NOI18N
        lblLowerVal2.setName("lblLowerVal2"); // NOI18N
        mainPanel.add(lblLowerVal2, new
org.netbeans.lib.awtextra.AbsoluteConstraints(550, 430, 20, 30));

        jLabel4.setFont(resourceMap.getFont("jLabel4.font")); // NOI18N
        jLabel4.setForeground(resourceMap.getColor("jLabel4.foreground")); // NOI18N
```

```
        jLabel4.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        jLabel4.setText(resourceMap.getString("jLabel4.text")); // NOI18N
        jLabel4.setName("jLabel4"); // NOI18N
        mainPanel.add(jLabel4, new org.netbeans.lib.awtextra.AbsoluteConstraints(490,
470, 80, 30));

        lblSubHt.setFont(resourceMap.getFont("lblSubHt.font")); // NOI18N
        lblSubHt.setForeground(resourceMap.getColor("lblSubHt.foreground")); // NOI18N
        lblSubHt.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        lblSubHt.setBorder(javax.swing.BorderFactory.createLineBorder(new
java.awt.Color(153, 153, 255)));
        lblSubHt.setName("lblSubHt"); // NOI18N
        mainPanel.add(lblSubHt, new org.netbeans.lib.awtextra.AbsoluteConstraints(580,
470, 60, 30));

        jLabel5.setFont(resourceMap.getFont("jLabel5.font")); // NOI18N
        jLabel5.setForeground(resourceMap.getColor("jLabel5.foreground")); // NOI18N
        jLabel5.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        jLabel5.setText(resourceMap.getString("jLabel5.text")); // NOI18N
        jLabel5.setName("jLabel5"); // NOI18N
        mainPanel.add(jLabel5, new org.netbeans.lib.awtextra.AbsoluteConstraints(650,
470, 30, 30));

        pnlDir.setBackground(resourceMap.getColor("pnlDir.background")); // NOI18N

pnlDir.setBorder(javax.swing.BorderFactory.createEtchedBorder(resourceMap.getColor("pnlDi
r.border.highlightColor"), resourceMap.getColor("pnlDir.border.shadowColor"))); // NOI18N
        pnlDir.setMaximumSize(new java.awt.Dimension(305, 305));
        pnlDir.setMinimumSize(new java.awt.Dimension(305, 305));
        pnlDir.setName("pnlDir"); // NOI18N

        javax.swing.GroupLayout pnlDirLayout = new javax.swing.GroupLayout(pnlDir);
        pnlDir.setLayout(pnlDirLayout);
        pnlDirLayout.setHorizontalGroup(
            pnlDirLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGap(0, 301, Short.MAX_VALUE)
        );
        pnlDirLayout.setVerticalGroup(
            pnlDirLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGap(0, 301, Short.MAX_VALUE)
        );

        mainPanel.add(pnlDir, new org.netbeans.lib.awtextra.AbsoluteConstraints(500, 120,
305, 305));

        pnlLegend.setBackground(resourceMap.getColor("pnlLegend.background")); // NOI18N
        pnlLegend.setName("pnlLegend"); // NOI18N

        javax.swing.GroupLayout pnlLegendLayout = new javax.swing.GroupLayout(pnlLegend);
        pnlLegend.setLayout(pnlLegendLayout);
        pnlLegendLayout.setHorizontalGroup(

pnlLegendLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGap(0, 60, Short.MAX_VALUE)
        );
        pnlLegendLayout.setVerticalGroup(

pnlLegendLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGap(0, 370, Short.MAX_VALUE)
        );

        mainPanel.add(pnlLegend, new org.netbeans.lib.awtextra.AbsoluteConstraints(820,
120, 60, 370));

        lblBuoyName.setFont(resourceMap.getFont("lblBuoyName.font")); // NOI18N
        lblBuoyName.setForeground(resourceMap.getColor("lblBuoyName.foreground")); //
NOI18N
        lblBuoyName.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        lblBuoyName.setBorder(javax.swing.BorderFactory.createLineBorder(new
java.awt.Color(153, 153, 255)));
        lblBuoyName.setName("lblBuoyName"); // NOI18N
        mainPanel.add(lblBuoyName, new org.netbeans.lib.awtextra.AbsoluteConstraints(10,
30, 470, 30));

        sliderTime.setEnabled(false);

        menuBar.setBackground(resourceMap.getColor("menuBar.background")); // NOI18N
        menuBar.setBorder(new
javax.swing.border.SoftBevelBorder(javax.swing.border.BevelBorder.RAISED,
resourceMap.getColor("menuBar.border.highlightOuterColor"),
```

```
resourceMap.getColor("menuBar.border.highlightInnerColor"), null, null)); // NOI18N
        menuBar.setForeground(resourceMap.getColor("menuBar.foreground")); // NOI18N
        menuBar.setDoubleBuffered(true);
        menuBar.setMaximumSize(new java.awt.Dimension(1000, 27));
        menuBar.setMinimumSize(new java.awt.Dimension(600, 27));
        menuBar.setName("menuBar"); // NOI18N
        menuBar.setPreferredSize(new java.awt.Dimension(600, 27));

        fileMenu.setText(resourceMap.getString("fileMenu.text")); // NOI18N
        fileMenu.setName("fileMenu"); // NOI18N

        javax.swing.ActionMap actionMap =
org.jdesktop.application.Application.getInstance(buoywaves.BuoyWavesApp.class).getContext
().getActionMap(BuoyWavesView.class, this);
        exitMenuItem.setAction(actionMap.get("quit")); // NOI18N
        exitMenuItem.setName("exitMenuItem"); // NOI18N
        fileMenu.add(exitMenuItem);

        jSeparator1.setName("jSeparator1"); // NOI18N
        fileMenu.add(jSeparator1);

        menuBar.add(fileMenu);

        mnuAlaska.setText(resourceMap.getString("mnuAlaska.text")); // NOI18N
        mnuAlaska.setName("mnuAlaska"); // NOI18N

        mnu46070.setText(resourceMap.getString("mnu46070.text")); // NOI18N
        mnu46070.setActionCommand(resourceMap.getString("mnu46070.actionCommand")); //
NOI18N
        mnu46070.setName("mnu46070"); // NOI18N
        mnu46070.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuAlaska.add(mnu46070);

        mnu46035.setText(resourceMap.getString("mnu46035.text")); // NOI18N
        mnu46035.setActionCommand(resourceMap.getString("mnu46035.actionCommand")); //
NOI18N
        mnu46035.setName("mnu46035"); // NOI18N
        mnu46035.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuAlaska.add(mnu46035);

        mnu46073.setText(resourceMap.getString("mnu46073.text")); // NOI18N
        mnu46073.setActionCommand(resourceMap.getString("mnu46073.actionCommand")); //
NOI18N
        mnu46073.setName("mnu46073"); // NOI18N
        mnu46073.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuAlaska.add(mnu46073);

        jSeparator2.setName("jSeparator2"); // NOI18N
        mnuAlaska.add(jSeparator2);

        mnu46071.setText(resourceMap.getString("mnu46071.text")); // NOI18N
        mnu46071.setActionCommand(resourceMap.getString("mnu46071.actionCommand")); //
NOI18N
        mnu46071.setName("mnu46071"); // NOI18N
        mnu46071.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuAlaska.add(mnu46071);

        mnu46072.setText(resourceMap.getString("mnu46072.text")); // NOI18N
        mnu46072.setActionCommand(resourceMap.getString("mnu46072.actionCommand")); //
NOI18N
        mnu46072.setName("mnu46072"); // NOI18N
        mnu46072.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
```

```
            }
        });
        mnuAlaska.add(mnu46072);

        mnu46075.setText(resourceMap.getString("mnu46075.text")); // NOI18N
        mnu46075.setActionCommand(resourceMap.getString("mnu46075.actionCommand")); //
NOI18N
        mnu46075.setName("mnu46075"); // NOI18N
        mnu46075.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuAlaska.add(mnu46075);

        jSeparator3.setName("jSeparator3"); // NOI18N
        mnuAlaska.add(jSeparator3);

        mnu46077.setText(resourceMap.getString("mnu46077.text")); // NOI18N
        mnu46077.setActionCommand(resourceMap.getString("mnu46077.actionCommand")); //
NOI18N
        mnu46077.setName("mnu46077"); // NOI18N
        mnu46077.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuAlaska.add(mnu46077);

        mnu46066.setText(resourceMap.getString("mnu46066.text")); // NOI18N
        mnu46066.setActionCommand(resourceMap.getString("mnu46066.actionCommand")); //
NOI18N
        mnu46066.setName("mnu46066"); // NOI18N
        mnu46066.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuAlaska.add(mnu46066);

        mnu46078.setText(resourceMap.getString("mnu46078.text")); // NOI18N
        mnu46078.setActionCommand(resourceMap.getString("mnu46078.actionCommand")); //
NOI18N
        mnu46078.setName("mnu46078"); // NOI18N
        mnu46078.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuAlaska.add(mnu46078);

        mnu46080.setText(resourceMap.getString("mnu46080.text")); // NOI18N
        mnu46080.setActionCommand(resourceMap.getString("mnu46080.actionCommand")); //
NOI18N
        mnu46080.setName("mnu46080"); // NOI18N
        mnu46080.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuAlaska.add(mnu46080);

        mnu46001.setText(resourceMap.getString("mnu46001.text")); // NOI18N
        mnu46001.setActionCommand(resourceMap.getString("mnu46001.actionCommand")); //
NOI18N
        mnu46001.setName("mnu46001"); // NOI18N
        mnu46001.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuAlaska.add(mnu46001);

        mnu46082.setText(resourceMap.getString("mnu46082.text")); // NOI18N
        mnu46082.setActionCommand(resourceMap.getString("mnu46082.actionCommand")); //
NOI18N
        mnu46082.setName("mnu46082"); // NOI18N
        mnu46082.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
```

```
            }
        });
        mnuAlaska.add(mnu46082);

        mnu46085.setText(resourceMap.getString("mnu46085.text")); // NOI18N
        mnu46085.setActionCommand(resourceMap.getString("mnu46085.actionCommand")); //
NOI18N
        mnu46085.setName("mnu46085"); // NOI18N
        mnu46085.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuAlaska.add(mnu46085);

        jSeparator4.setName("jSeparator4"); // NOI18N
        mnuAlaska.add(jSeparator4);

        mnu46083.setText(resourceMap.getString("mnu46083.text")); // NOI18N
        mnu46083.setActionCommand(resourceMap.getString("mnu46083.actionCommand")); //
NOI18N
        mnu46083.setName("mnu46083"); // NOI18N
        mnu46083.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuAlaska.add(mnu46083);

        mnu46084.setText(resourceMap.getString("mnu46084.text")); // NOI18N
        mnu46084.setActionCommand(resourceMap.getString("mnu46084.actionCommand")); //
NOI18N
        mnu46084.setName("mnu46084"); // NOI18N
        mnu46084.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuAlaska.add(mnu46084);

        menuBar.add(mnuAlaska);

        mnuHawaii.setText(resourceMap.getString("mnuHawaii.text")); // NOI18N
        mnuHawaii.setName("mnuHawaii"); // NOI18N

        mnu51000.setText(resourceMap.getString("mnu51000.text")); // NOI18N
        mnu51000.setActionCommand(resourceMap.getString("mnu51000.actionCommand")); //
NOI18N
        mnu51000.setName("mnu51000"); // NOI18N
        mnu51000.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuHawaii.add(mnu51000);

        mnu51001.setText(resourceMap.getString("mnu51001.text")); // NOI18N
        mnu51001.setActionCommand(resourceMap.getString("mnu51001.actionCommand")); //
NOI18N
        mnu51001.setName("mnu51001"); // NOI18N
        mnu51001.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuHawaii.add(mnu51001);

        mnu51003.setText(resourceMap.getString("mnu51003.text")); // NOI18N
        mnu51003.setActionCommand(resourceMap.getString("mnu51003.actionCommand")); //
NOI18N
        mnu51003.setName("mnu51003"); // NOI18N
        mnu51003.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuHawaii.add(mnu51003);

        mnu51002.setText(resourceMap.getString("mnu51002.text")); // NOI18N
        mnu51002.setActionCommand(resourceMap.getString("mnu51002.actionCommand")); //
```

```
NOI18N
        mnu51002.setName("mnu51002"); // NOI18N
        mnu51002.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuHawaii.add(mnu51002);

        mnu51004.setText(resourceMap.getString("mnu51004.text")); // NOI18N
        mnu51004.setActionCommand(resourceMap.getString("mnu51004.actionCommand")); //
NOI18N
        mnu51004.setName("mnu51004"); // NOI18N
        mnu51004.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuHawaii.add(mnu51004);

        menuBar.add(mnuHawaii);

        mnuNW.setText(resourceMap.getString("mnuNW.text")); // NOI18N
        mnuNW.setName("mnuNW"); // NOI18N

        mnu46005.setText(resourceMap.getString("mnu46005.text")); // NOI18N
        mnu46005.setActionCommand(resourceMap.getString("mnu46005.actionCommand")); //
NOI18N
        mnu46005.setName("mnu46005"); // NOI18N
        mnu46005.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuNW.add(mnu46005);

        mnu46089.setText(resourceMap.getString("mnu46089.text")); // NOI18N
        mnu46089.setActionCommand(resourceMap.getString("mnu46089.actionCommand")); //
NOI18N
        mnu46089.setName("mnu46089"); // NOI18N
        mnu46089.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuNW.add(mnu46089);

        mnu46002.setText(resourceMap.getString("mnu46002.text")); // NOI18N
        mnu46002.setActionCommand(resourceMap.getString("mnu46002.actionCommand")); //
NOI18N
        mnu46002.setName("mnu46002"); // NOI18N
        mnu46002.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuNW.add(mnu46002);

        mnu46006.setText(resourceMap.getString("mnu46006.text")); // NOI18N
        mnu46006.setActionCommand(resourceMap.getString("mnu46006.actionCommand")); //
NOI18N
        mnu46006.setName("mnu46006"); // NOI18N
        mnu46006.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuNW.add(mnu46006);

        mnu46059.setText(resourceMap.getString("mnu46059.text")); // NOI18N
        mnu46059.setActionCommand(resourceMap.getString("mnu46059.actionCommand")); //
NOI18N
        mnu46059.setName("mnu46059"); // NOI18N
        mnu46059.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuNW.add(mnu46059);
```

```
        jSeparator5.setName("jSeparator5"); // NOI18N
        mnuNW.add(jSeparator5);

        mnu46088.setText(resourceMap.getString("mnu46088.text")); // NOI18N
        mnu46088.setActionCommand(resourceMap.getString("mnu46088.actionCommand")); //
NOI18N
        mnu46088.setName("mnu46088"); // NOI18N
        mnu46088.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuNW.add(mnu46088);

        mnu46087.setText(resourceMap.getString("mnu46087.text")); // NOI18N
        mnu46087.setActionCommand(resourceMap.getString("mnu46087.actionCommand")); //
NOI18N
        mnu46087.setName("mnu46087"); // NOI18N
        mnu46087.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuNW.add(mnu46087);

        mnu46041.setText(resourceMap.getString("mnu46041.text")); // NOI18N
        mnu46041.setActionCommand(resourceMap.getString("mnu46041.actionCommand")); //
NOI18N
        mnu46041.setName("mnu46041"); // NOI18N
        mnu46041.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuNW.add(mnu46041);

        jSeparator6.setName("jSeparator6"); // NOI18N
        mnuNW.add(jSeparator6);

        mnu46029.setText(resourceMap.getString("mnu46029.text")); // NOI18N
        mnu46029.setActionCommand(resourceMap.getString("mnu46029.actionCommand")); //
NOI18N
        mnu46029.setName("mnu46029"); // NOI18N
        mnu46029.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuNW.add(mnu46029);

        mnu46050.setText(resourceMap.getString("mnu46050.text")); // NOI18N
        mnu46050.setActionCommand(resourceMap.getString("mnu46050.actionCommand")); //
NOI18N
        mnu46050.setName("mnu46050"); // NOI18N
        mnu46050.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuNW.add(mnu46050);

        mnu46015.setText(resourceMap.getString("mnu46015.text")); // NOI18N
        mnu46015.setActionCommand(resourceMap.getString("mnu46015.actionCommand")); //
NOI18N
        mnu46015.setName("mnu46015"); // NOI18N
        mnu46015.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuNW.add(mnu46015);

        jSeparator7.setName("jSeparator7"); // NOI18N
        mnuNW.add(jSeparator7);

        mnu46027.setText(resourceMap.getString("mnu46027.text")); // NOI18N
        mnu46027.setActionCommand(resourceMap.getString("mnu46027.actionCommand")); //
NOI18N
        mnu46027.setName("mnu46027"); // NOI18N
        mnu46027.addActionListener(new java.awt.event.ActionListener() {
```

```
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuNW.add(mnu46027);

        mnu46022.setText(resourceMap.getString("mnu46022.text")); // NOI18N
        mnu46022.setActionCommand(resourceMap.getString("mnu46022.actionCommand")); //
NOI18N
        mnu46022.setName("mnu46022"); // NOI18N
        mnu46022.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuNW.add(mnu46022);

        mnu46014.setText(resourceMap.getString("mnu46014.text")); // NOI18N
        mnu46014.setActionCommand(resourceMap.getString("mnu46014.actionCommand")); //
NOI18N
        mnu46014.setName("mnu46014"); // NOI18N
        mnu46014.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuNW.add(mnu46014);

        mnu46013.setText(resourceMap.getString("mnu46013.text")); // NOI18N
        mnu46013.setActionCommand(resourceMap.getString("mnu46013.actionCommand")); //
NOI18N
        mnu46013.setName("mnu46013"); // NOI18N
        mnu46013.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuNW.add(mnu46013);

        mnu46026.setText(resourceMap.getString("mnu46026.text")); // NOI18N
        mnu46026.setActionCommand(resourceMap.getString("mnu46026.actionCommand")); //
NOI18N
        mnu46026.setName("mnu46026"); // NOI18N
        mnu46026.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuNW.add(mnu46026);

        mnu46012.setText(resourceMap.getString("mnu46012.text")); // NOI18N
        mnu46012.setActionCommand(resourceMap.getString("mnu46012.actionCommand")); //
NOI18N
        mnu46012.setName("mnu46012"); // NOI18N
        mnu46012.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuNW.add(mnu46012);

        mnu46042.setText(resourceMap.getString("mnu46042.text")); // NOI18N
        mnu46042.setActionCommand(resourceMap.getString("mnu46042.actionCommand")); //
NOI18N
        mnu46042.setName("mnu46042"); // NOI18N
        mnu46042.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuNW.add(mnu46042);

        menuBar.add(mnuNW);

        mnuSW.setText(resourceMap.getString("mnuSW.text")); // NOI18N
        mnuSW.setName("mnuSW"); // NOI18N

        mnu46028.setText(resourceMap.getString("mnu46028.text")); // NOI18N
        mnu46028.setActionCommand(resourceMap.getString("mnu46028.actionCommand")); //
NOI18N
```

```
        mnu46028.setName("mnu46028"); // NOI18N
        mnu46028.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuSW.add(mnu46028);

        mnu46011.setText(resourceMap.getString("mnu46011.text")); // NOI18N
        mnu46011.setActionCommand(resourceMap.getString("mnu46011.actionCommand")); //
NOI18N
        mnu46011.setName("mnu46011"); // NOI18N
        mnu46011.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuSW.add(mnu46011);

        mnu46023.setText(resourceMap.getString("mnu46023.text")); // NOI18N
        mnu46023.setActionCommand(resourceMap.getString("mnu46023.actionCommand")); //
NOI18N
        mnu46023.setName("mnu46023"); // NOI18N
        mnu46023.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuSW.add(mnu46023);

        mnu46063.setText(resourceMap.getString("mnu46063.text")); // NOI18N
        mnu46063.setActionCommand(resourceMap.getString("mnu46063.actionCommand")); //
NOI18N
        mnu46063.setName("mnu46063"); // NOI18N
        mnu46063.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuSW.add(mnu46063);

        mnu46069.setText(resourceMap.getString("mnu46069.text")); // NOI18N
        mnu46069.setActionCommand(resourceMap.getString("mnu46069.actionCommand")); //
NOI18N
        mnu46069.setName("mnu46069"); // NOI18N
        mnu46069.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuSW.add(mnu46069);

        mnu46047.setText(resourceMap.getString("mnu46047.text")); // NOI18N
        mnu46047.setActionCommand(resourceMap.getString("mnu46047.actionCommand")); //
NOI18N
        mnu46047.setName("mnu46047"); // NOI18N
        mnu46047.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuSW.add(mnu46047);

        jSeparator8.setName("jSeparator8"); // NOI18N
        mnuSW.add(jSeparator8);

        mnu46054.setText(resourceMap.getString("mnu46054.text")); // NOI18N
        mnu46054.setActionCommand(resourceMap.getString("mnu46054.actionCommand")); //
NOI18N
        mnu46054.setName("mnu46054"); // NOI18N
        mnu46054.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuSW.add(mnu46054);

        mnu46053.setText(resourceMap.getString("mnu46053.text")); // NOI18N
        mnu46053.setActionCommand(resourceMap.getString("mnu46053.actionCommand")); //
NOI18N
```

```
        mnu46053.setName("mnu46053"); // NOI18N
        mnu46053.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuSW.add(mnu46053);

        mnu46025.setText(resourceMap.getString("mnu46025.text")); // NOI18N
        mnu46025.setActionCommand(resourceMap.getString("mnu46025.actionCommand")); //
NOI18N
        mnu46025.setName("mnu46025"); // NOI18N
        mnu46025.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuSW.add(mnu46025);

        mnu46086.setText(resourceMap.getString("mnu46086.text")); // NOI18N
        mnu46086.setActionCommand(resourceMap.getString("mnu46086.actionCommand")); //
NOI18N
        mnu46086.setName("mnu46086"); // NOI18N
        mnu46086.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuSW.add(mnu46086);

        menuBar.add(mnuSW);

        mnuGulf.setText(resourceMap.getString("mnuGulf.text")); // NOI18N
        mnuGulf.setName("mnuGulf"); // NOI18N

        mnu42020.setText(resourceMap.getString("mnu42020.text")); // NOI18N
        mnu42020.setActionCommand(resourceMap.getString("mnu42020.actionCommand")); //
NOI18N
        mnu42020.setName("mnu42020"); // NOI18N
        mnu42020.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuGulf.add(mnu42020);

        mnu42019.setText(resourceMap.getString("mnu42019.text")); // NOI18N
        mnu42019.setActionCommand(resourceMap.getString("mnu42019.actionCommand")); //
NOI18N
        mnu42019.setName("mnu42019"); // NOI18N
        mnu42019.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuGulf.add(mnu42019);

        mnu42035.setText(resourceMap.getString("mnu42035.text")); // NOI18N
        mnu42035.setActionCommand(resourceMap.getString("mnu42035.actionCommand")); //
NOI18N
        mnu42035.setName("mnu42035"); // NOI18N
        mnu42035.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuGulf.add(mnu42035);

        jSeparator9.setName("jSeparator9"); // NOI18N
        mnuGulf.add(jSeparator9);

        mnu42002.setText(resourceMap.getString("mnu42002.text")); // NOI18N
        mnu42002.setActionCommand(resourceMap.getString("mnu42002.actionCommand")); //
NOI18N
        mnu42002.setName("mnu42002"); // NOI18N
        mnu42002.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
```

```
        mnuGulf.add(mnu42002);

        mnu42055.setText(resourceMap.getString("mnu42055.text")); // NOI18N
        mnu42055.setActionCommand(resourceMap.getString("mnu42055.actionCommand")); //
NOI18N
        mnu42055.setName("mnu42055"); // NOI18N
        mnu42055.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuGulf.add(mnu42055);

        mnu42001.setText(resourceMap.getString("mnu42001.text")); // NOI18N
        mnu42001.setActionCommand(resourceMap.getString("mnu42001.actionCommand")); //
NOI18N
        mnu42001.setName("mnu42001"); // NOI18N
        mnu42001.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuGulf.add(mnu42001);

        mnu42003.setText(resourceMap.getString("mnu42003.text")); // NOI18N
        mnu42003.setActionCommand(resourceMap.getString("mnu42003.actionCommand")); //
NOI18N
        mnu42003.setName("mnu42003"); // NOI18N
        mnu42003.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuGulf.add(mnu42003);

        jSeparator10.setName("jSeparator10"); // NOI18N
        mnuGulf.add(jSeparator10);

        mnu42007.setText(resourceMap.getString("mnu42007.text")); // NOI18N
        mnu42007.setActionCommand(resourceMap.getString("mnu42007.actionCommand")); //
NOI18N
        mnu42007.setName("mnu42007"); // NOI18N
        mnu42007.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuGulf.add(mnu42007);

        mnu42012.setText(resourceMap.getString("mnu42012.text")); // NOI18N
        mnu42012.setActionCommand(resourceMap.getString("mnu42012.actionCommand")); //
NOI18N
        mnu42012.setName("mnu42012"); // NOI18N
        mnu42012.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuGulf.add(mnu42012);

        mnu42040.setText(resourceMap.getString("mnu42040.text")); // NOI18N
        mnu42040.setActionCommand(resourceMap.getString("mnu42040.actionCommand")); //
NOI18N
        mnu42040.setName("mnu42040"); // NOI18N
        mnu42040.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuGulf.add(mnu42040);

        jSeparator11.setName("jSeparator11"); // NOI18N
        mnuGulf.add(jSeparator11);

        mnu42039.setText(resourceMap.getString("mnu42039.text")); // NOI18N
        mnu42039.setActionCommand(resourceMap.getString("mnu42039.actionCommand")); //
NOI18N
        mnu42039.setName("mnu42039"); // NOI18N
        mnu42039.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
```

```
                MenuHandler(evt);
            }
        });
        mnuGulf.add(mnu42039);

        mnu42036.setText(resourceMap.getString("mnu42036.text")); // NOI18N
        mnu42036.setActionCommand(resourceMap.getString("mnu42036.actionCommand")); //
NOI18N
        mnu42036.setName("mnu42036"); // NOI18N
        mnu42036.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuGulf.add(mnu42036);

        mnu42080.setText(resourceMap.getString("mnu42080.text")); // NOI18N
        mnu42080.setActionCommand(resourceMap.getString("mnu42080.actionCommand")); //
NOI18N
        mnu42080.setName("mnu42080"); // NOI18N
        mnu42080.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuGulf.add(mnu42080);

        jSeparator12.setName("jSeparator12"); // NOI18N
        mnuGulf.add(jSeparator12);

        mnu42056.setText(resourceMap.getString("mnu42056.text")); // NOI18N
        mnu42056.setActionCommand(resourceMap.getString("mnu42056.actionCommand")); //
NOI18N
        mnu42056.setName("mnu42056"); // NOI18N
        mnu42056.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuGulf.add(mnu42056);

        mnu42057.setText(resourceMap.getString("mnu42057.text")); // NOI18N
        mnu42057.setActionCommand(resourceMap.getString("mnu42057.actionCommand")); //
NOI18N
        mnu42057.setName("mnu42057"); // NOI18N
        mnu42057.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuGulf.add(mnu42057);

        mnu42058.setText(resourceMap.getString("mnu42058.text")); // NOI18N
        mnu42058.setActionCommand(resourceMap.getString("mnu42058.actionCommand")); //
NOI18N
        mnu42058.setName("mnu42058"); // NOI18N
        mnu42058.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuGulf.add(mnu42058);

        mnu42059.setText(resourceMap.getString("mnu42059.text")); // NOI18N
        mnu42059.setActionCommand(resourceMap.getString("mnu42059.actionCommand")); //
NOI18N
        mnu42059.setName("mnu42059"); // NOI18N
        mnu42059.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuGulf.add(mnu42059);

        mnu42060.setText(resourceMap.getString("mnu42060.text")); // NOI18N
        mnu42060.setActionCommand(resourceMap.getString("mnu42060.actionCommand")); //
NOI18N
        mnu42060.setName("mnu42060"); // NOI18N
        mnu42060.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
```

```
                MenuHandler(evt);
            }
        });
        mnuGulf.add(mnu42060);

        jSeparator13.setName("jSeparator13"); // NOI18N
        mnuGulf.add(jSeparator13);

        mnu41043.setText(resourceMap.getString("mnu41043.text")); // NOI18N
        mnu41043.setActionCommand(resourceMap.getString("mnu41043.actionCommand")); //
NOI18N
        mnu41043.setName("mnu41043"); // NOI18N
        mnu41043.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuGulf.add(mnu41043);

        mnu41040.setText(resourceMap.getString("mnu41040.text")); // NOI18N
        mnu41040.setActionCommand(resourceMap.getString("mnu41040.actionCommand")); //
NOI18N
        mnu41040.setName("mnu41040"); // NOI18N
        mnu41040.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuGulf.add(mnu41040);

        mnu41041.setText(resourceMap.getString("mnu41041.text")); // NOI18N
        mnu41041.setActionCommand(resourceMap.getString("mnu41041.actionCommand")); //
NOI18N
        mnu41041.setName("mnu41041"); // NOI18N
        mnu41041.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuGulf.add(mnu41041);

        mnu41044.setText(resourceMap.getString("mnu41044.text")); // NOI18N
        mnu41044.setActionCommand(resourceMap.getString("mnu41044.actionCommand")); //
NOI18N
        mnu41044.setName("mnu41044"); // NOI18N
        mnu41044.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuGulf.add(mnu41044);

        menuBar.add(mnuGulf);

        mnuWAtl.setText(resourceMap.getString("mnuWAtl.text")); // NOI18N
        mnuWAtl.setName("mnuWAtl"); // NOI18N

        mnu41046.setText(resourceMap.getString("mnu41046.text")); // NOI18N
        mnu41046.setActionCommand(resourceMap.getString("mnu41046.actionCommand")); //
NOI18N
        mnu41046.setName("mnu41046"); // NOI18N
        mnu41046.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuWAtl.add(mnu41046);

        mnu41047.setText(resourceMap.getString("mnu41047.text")); // NOI18N
        mnu41047.setActionCommand(resourceMap.getString("mnu41047.actionCommand")); //
NOI18N
        mnu41047.setName("mnu41047"); // NOI18N
        mnu41047.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuWAtl.add(mnu41047);

        mnu41048.setText(resourceMap.getString("mnu41048.text")); // NOI18N
```

```
        mnu41048.setActionCommand(resourceMap.getString("mnu41048.actionCommand")); //
NOI18N
        mnu41048.setName("mnu41048"); // NOI18N
        mnu41048.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuWAtl.add(mnu41048);

        mnu41049.setText(resourceMap.getString("mnu41049.text")); // NOI18N
        mnu41049.setActionCommand(resourceMap.getString("mnu41049.actionCommand")); //
NOI18N
        mnu41049.setName("mnu41049"); // NOI18N
        mnu41049.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuWAtl.add(mnu41049);

        jSeparator14.setName("jSeparator14"); // NOI18N
        mnuWAtl.add(jSeparator14);

        mnu41010.setText(resourceMap.getString("mnu41010.text")); // NOI18N
        mnu41010.setActionCommand(resourceMap.getString("mnu41010.actionCommand")); //
NOI18N
        mnu41010.setName("mnu41010"); // NOI18N
        mnu41010.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuWAtl.add(mnu41010);

        mnu41009.setText(resourceMap.getString("mnu41009.text")); // NOI18N
        mnu41009.setActionCommand(resourceMap.getString("mnu41009.actionCommand")); //
NOI18N
        mnu41009.setName("mnu41009"); // NOI18N
        mnu41009.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuWAtl.add(mnu41009);

        mnu41012.setText(resourceMap.getString("mnu41012.text")); // NOI18N
        mnu41012.setActionCommand(resourceMap.getString("mnu41012.actionCommand")); //
NOI18N
        mnu41012.setName("mnu41012"); // NOI18N
        mnu41012.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuWAtl.add(mnu41012);

        mnu41008.setText(resourceMap.getString("mnu41008.text")); // NOI18N
        mnu41008.setActionCommand(resourceMap.getString("mnu41008.actionCommand")); //
NOI18N
        mnu41008.setName("mnu41008"); // NOI18N
        mnu41008.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuWAtl.add(mnu41008);

        jSeparator15.setName("jSeparator15"); // NOI18N
        mnuWAtl.add(jSeparator15);

        mnu41001.setText(resourceMap.getString("mnu41001.text")); // NOI18N
        mnu41001.setActionCommand(resourceMap.getString("mnu41001.actionCommand")); //
NOI18N
        mnu41001.setName("mnu41001"); // NOI18N
        mnu41001.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
```

```
        mnuWAtl.add(mnu41001);

        mnu41002.setText(resourceMap.getString("mnu41002.text")); // NOI18N
        mnu41002.setActionCommand(resourceMap.getString("mnu41002.actionCommand")); //
NOI18N
        mnu41002.setName("mnu41002"); // NOI18N
        mnu41002.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuWAtl.add(mnu41002);

        mnu41013.setText(resourceMap.getString("mnu41013.text")); // NOI18N
        mnu41013.setActionCommand(resourceMap.getString("mnu41013.actionCommand")); //
NOI18N
        mnu41013.setName("mnu41013"); // NOI18N
        mnu41013.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuWAtl.add(mnu41013);

        jSeparator16.setName("jSeparator16"); // NOI18N
        mnuWAtl.add(jSeparator16);

        mnu41004.setText(resourceMap.getString("mnu41004.text")); // NOI18N
        mnu41004.setActionCommand(resourceMap.getString("mnu41004.actionCommand")); //
NOI18N
        mnu41004.setName("mnu41004"); // NOI18N
        mnu41004.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuWAtl.add(mnu41004);

        mnu41035.setText(resourceMap.getString("mnu41035.text")); // NOI18N
        mnu41035.setActionCommand(resourceMap.getString("mnu41035.actionCommand")); //
NOI18N
        mnu41035.setName("mnu41035"); // NOI18N
        mnu41035.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuWAtl.add(mnu41035);

        mnu41036.setText(resourceMap.getString("mnu41036.text")); // NOI18N
        mnu41036.setActionCommand(resourceMap.getString("mnu41036.actionCommand")); //
NOI18N
        mnu41036.setName("mnu41036"); // NOI18N
        mnu41036.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuWAtl.add(mnu41036);

        mnu41025.setText(resourceMap.getString("mnu41025.text")); // NOI18N
        mnu41025.setActionCommand(resourceMap.getString("mnu41025.actionCommand")); //
NOI18N
        mnu41025.setName("mnu41025"); // NOI18N
        mnu41025.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuWAtl.add(mnu41025);

        menuBar.add(mnuWAtl);

        mnuNWAtl.setText(resourceMap.getString("mnuNWAtl.text")); // NOI18N
        mnuNWAtl.setName("mnuNWAtl"); // NOI18N

        mnu44014.setText(resourceMap.getString("mnu44014.text")); // NOI18N
        mnu44014.setActionCommand(resourceMap.getString("mnu44014.actionCommand")); //
NOI18N
        mnu44014.setName("mnu44014"); // NOI18N
```

```
        mnu44014.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuNWAtl.add(mnu44014);

        mnu44009.setText(resourceMap.getString("mnu44009.text")); // NOI18N
        mnu44009.setActionCommand(resourceMap.getString("mnu44009.actionCommand")); //
NOI18N
        mnu44009.setName("mnu44009"); // NOI18N
        mnu44009.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuNWAtl.add(mnu44009);

        jSeparator17.setName("jSeparator17"); // NOI18N
        mnuNWAtl.add(jSeparator17);

        mnu44004.setText(resourceMap.getString("mnu44004.text")); // NOI18N
        mnu44004.setActionCommand(resourceMap.getString("mnu44004.actionCommand")); //
NOI18N
        mnu44004.setName("mnu44004"); // NOI18N
        mnu44004.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuNWAtl.add(mnu44004);

        mny44066.setText(resourceMap.getString("mny44066.text")); // NOI18N
        mny44066.setActionCommand(resourceMap.getString("mny44066.actionCommand")); //
NOI18N
        mny44066.setName("mny44066"); // NOI18N
        mny44066.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuNWAtl.add(mny44066);

        mnu44008.setText(resourceMap.getString("mnu44008.text")); // NOI18N
        mnu44008.setActionCommand(resourceMap.getString("mnu44008.actionCommand")); //
NOI18N
        mnu44008.setName("mnu44008"); // NOI18N
        mnu44008.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuNWAtl.add(mnu44008);

        mnu44018.setText(resourceMap.getString("mnu44018.text")); // NOI18N
        mnu44018.setActionCommand(resourceMap.getString("mnu44018.actionCommand")); //
NOI18N
        mnu44018.setName("mnu44018"); // NOI18N
        mnu44018.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuNWAtl.add(mnu44018);

        mnu44005.setText(resourceMap.getString("mnu44005.text")); // NOI18N
        mnu44005.setActionCommand(resourceMap.getString("mnu44005.actionCommand")); //
NOI18N
        mnu44005.setName("mnu44005"); // NOI18N
        mnu44005.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuNWAtl.add(mnu44005);

        mnu44011.setText(resourceMap.getString("mnu44011.text")); // NOI18N
        mnu44011.setActionCommand(resourceMap.getString("mnu44011.actionCommand")); //
NOI18N
        mnu44011.setName("mnu44011"); // NOI18N
```

```
        mnu44011.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuNWAtl.add(mnu44011);

        jSeparator18.setName("jSeparator18"); // NOI18N
        mnuNWAtl.add(jSeparator18);

        mnu44025.setText(resourceMap.getString("mnu44025.text")); // NOI18N
        mnu44025.setActionCommand(resourceMap.getString("mnu44025.actionCommand")); //
NOI18N
        mnu44025.setName("mnu44025"); // NOI18N
        mnu44025.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuNWAtl.add(mnu44025);

        mnu44017.setText(resourceMap.getString("mnu44017.text")); // NOI18N
        mnu44017.setActionCommand(resourceMap.getString("mnu44017.actionCommand")); //
NOI18N
        mnu44017.setName("mnu44017"); // NOI18N
        mnu44017.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuNWAtl.add(mnu44017);

        mnu44065.setText(resourceMap.getString("mnu44065.text")); // NOI18N
        mnu44065.setActionCommand(resourceMap.getString("mnu44065.actionCommand")); //
NOI18N
        mnu44065.setName("mnu44065"); // NOI18N
        mnu44065.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuNWAtl.add(mnu44065);

        mnu44013.setText(resourceMap.getString("mnu44013.text")); // NOI18N
        mnu44013.setActionCommand(resourceMap.getString("mnu44013.actionCommand")); //
NOI18N
        mnu44013.setName("mnu44013"); // NOI18N
        mnu44013.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuNWAtl.add(mnu44013);

        mnu44020.setText(resourceMap.getString("mnu44020.text")); // NOI18N
        mnu44020.setActionCommand(resourceMap.getString("mnu44020.actionCommand")); //
NOI18N
        mnu44020.setName("mnu44020"); // NOI18N
        mnu44020.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuNWAtl.add(mnu44020);

        mnu44007.setText(resourceMap.getString("mnu44007.text")); // NOI18N
        mnu44007.setActionCommand(resourceMap.getString("mnu44007.actionCommand")); //
NOI18N
        mnu44007.setName("mnu44007"); // NOI18N
        mnu44007.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuNWAtl.add(mnu44007);

        mnu44027.setText(resourceMap.getString("mnu44027.text")); // NOI18N
        mnu44027.setActionCommand(resourceMap.getString("mnu44027.actionCommand")); //
NOI18N
        mnu44027.setName("mnu44027"); // NOI18N
```

```
        mnu44027.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuNWAtl.add(mnu44027);

        menuBar.add(mnuNWAtl);

        mnuLakes.setText(resourceMap.getString("mnuLakes.text")); // NOI18N
        mnuLakes.setName("mnuLakes"); // NOI18N

        mnu45006.setText(resourceMap.getString("mnu45006.text")); // NOI18N
        mnu45006.setActionCommand(resourceMap.getString("mnu45006.actionCommand")); //
NOI18N
        mnu45006.setName("mnu45006"); // NOI18N
        mnu45006.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuLakes.add(mnu45006);

        mnu45001.setText(resourceMap.getString("mnu45001.text")); // NOI18N
        mnu45001.setActionCommand(resourceMap.getString("mnu45001.actionCommand")); //
NOI18N
        mnu45001.setName("mnu45001"); // NOI18N
        mnu45001.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuLakes.add(mnu45001);

        mnu45004.setText(resourceMap.getString("mnu45004.text")); // NOI18N
        mnu45004.setActionCommand(resourceMap.getString("mnu45004.actionCommand")); //
NOI18N
        mnu45004.setName("mnu45004"); // NOI18N
        mnu45004.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuLakes.add(mnu45004);

        jSeparator19.setName("jSeparator19"); // NOI18N
        mnuLakes.add(jSeparator19);

        mnu45002.setText(resourceMap.getString("mnu45002.text")); // NOI18N
        mnu45002.setActionCommand(resourceMap.getString("mnu45002.actionCommand")); //
NOI18N
        mnu45002.setName("mnu45002"); // NOI18N
        mnu45002.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuLakes.add(mnu45002);

        mnu45007.setText(resourceMap.getString("mnu45007.text")); // NOI18N
        mnu45007.setActionCommand(resourceMap.getString("mnu45007.actionCommand")); //
NOI18N
        mnu45007.setName("mnu45007"); // NOI18N
        mnu45007.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuLakes.add(mnu45007);

        jSeparator20.setName("jSeparator20"); // NOI18N
        mnuLakes.add(jSeparator20);

        mnu45003.setText(resourceMap.getString("mnu45003.text")); // NOI18N
        mnu45003.setActionCommand(resourceMap.getString("mnu45003.actionCommand")); //
NOI18N
        mnu45003.setName("mnu45003"); // NOI18N
        mnu45003.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
```

```
            }
        });
        mnuLakes.add(mnu45003);

        mnu45008.setText(resourceMap.getString("mnu45008.text")); // NOI18N
        mnu45008.setActionCommand(resourceMap.getString("mnu45008.actionCommand")); //
NOI18N
        mnu45008.setName("mnu45008"); // NOI18N
        mnu45008.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuLakes.add(mnu45008);

        jSeparator21.setName("jSeparator21"); // NOI18N
        mnuLakes.add(jSeparator21);

        mnu45005.setText(resourceMap.getString("mnu45005.text")); // NOI18N
        mnu45005.setActionCommand(resourceMap.getString("mnu45005.actionCommand")); //
NOI18N
        mnu45005.setName("mnu45005"); // NOI18N
        mnu45005.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuLakes.add(mnu45005);

        jSeparator22.setName("jSeparator22"); // NOI18N
        mnuLakes.add(jSeparator22);

        mnu45012.setText(resourceMap.getString("mnu45012.text")); // NOI18N
        mnu45012.setActionCommand(resourceMap.getString("mnu45012.actionCommand")); //
NOI18N
        mnu45012.setName("mnu45012"); // NOI18N
        mnu45012.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                MenuHandler(evt);
            }
        });
        mnuLakes.add(mnu45012);

        menuBar.add(mnuLakes);

        helpMenu.setText(resourceMap.getString("helpMenu.text")); // NOI18N
        helpMenu.setName("helpMenu"); // NOI18N

        aboutMenuItem.setAction(actionMap.get("showAboutBox")); // NOI18N
        aboutMenuItem.setName("aboutMenuItem"); // NOI18N
        helpMenu.add(aboutMenuItem);

        menuBar.add(helpMenu);

        statusPanel.setMaximumSize(new java.awt.Dimension(600, 25));
        statusPanel.setMinimumSize(new java.awt.Dimension(600, 25));
        statusPanel.setName("statusPanel"); // NOI18N

        statusPanelSeparator.setName("statusPanelSeparator"); // NOI18N

        statusMessageLabel.setFont(resourceMap.getFont("statusMessageLabel.font")); //
NOI18N
        statusMessageLabel.setName("statusMessageLabel"); // NOI18N

        statusAnimationLabel.setHorizontalAlignment(javax.swing.SwingConstants.LEFT);
        statusAnimationLabel.setName("statusAnimationLabel"); // NOI18N

        progressBar.setName("progressBar"); // NOI18N

        javax.swing.GroupLayout statusPanelLayout = new
javax.swing.GroupLayout(statusPanel);
        statusPanel.setLayout(statusPanelLayout);
        statusPanelLayout.setHorizontalGroup(

statusPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addComponent(statusPanelSeparator, javax.swing.GroupLayout.DEFAULT_SIZE,
924, Short.MAX_VALUE)
            .addGroup(statusPanelLayout.createSequentialGroup()
                .addContainerGap()
                .addComponent(statusMessageLabel)
```

```
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 904,
Short.MAX_VALUE)
                .addComponent(statusAnimationLabel)
                .addContainerGap())
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
statusPanelLayout.createSequentialGroup()
                .addContainerGap(764, Short.MAX_VALUE)
                .addComponent(progressBar, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(14, 14, 14))
        );
        statusPanelLayout.setVerticalGroup(

statusPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(statusPanelLayout.createSequentialGroup()
                .addComponent(statusPanelSeparator,
javax.swing.GroupLayout.PREFERRED_SIZE, 2, javax.swing.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

.addGroup(statusPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELIN
E)
                    .addComponent(statusMessageLabel)
                    .addComponent(statusAnimationLabel)
                    .addComponent(progressBar, javax.swing.GroupLayout.PREFERRED_SIZE,
14, javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(3, 3, 3))
        );

        setComponent(mainPanel);
        setMenuBar(menuBar);
        setStatusBar(statusPanel);
    }// </editor-fold>

private void MenuHandler(java.awt.event.ActionEvent evt) {

    // Handle selections of buoys


    // CurrentBuoy is global and represents the current operator selection

    this.CurrentBuoy = evt.getActionCommand();

    // Now retrieve the long name of the object that was clicked. This is the
    //  name of the component. Place this into the BuoyLabel global String variable
    //  for later use and give a message to the user that data are being retrieved
    //  and processed.

    String Label = evt.getSource().toString();
    int pos = Label.indexOf("text=") + 5;
    this.BuoyLabel = Label.substring(pos, Label.length() - 1);
    statusMessageLabel.setText("Getting data for " + this.BuoyLabel);
    lblBuoyName.setText(this.BuoyLabel);

    // Get the buoy observations for the selected location.

    this.CurrentBuoyObs.clear();

    this.CurrentBuoyObs = getBuoyObs(this.CurrentBuoy);

     statusMessageLabel.setText(this.BuoyLabel);
     if (CurrentBuoyObs.size() > 0) {

         // if there are observations in the array list, this block is executed.

         // Set the bounds of the time slider to the number of obs available for the
buoy.

         sliderTime.setEnabled(true);
         sliderTime.setMaximum(CurrentBuoyObs.size() - 1);
         sliderTime.setValue(CurrentBuoyObs.size() - 1);

         // For convenience set an individual buoy ob for computing/viewing.
         //  By default, the last observation in the array list is set as current.

         O = CurrentBuoyObs.get(CurrentBuoyObs.size() - 1);

         // Plot all labels and graphs for the current buoy.
         PlotGraphicsAndLabels(O);
     }
```

```
     else {

         // If for some reason NDBC can't be reached, or if the selected buoy is
         //  not available, this block is executed.

         CurrentBuoyObs.clear();
         O = null;
         statusMessageLabel.setText("No Data Available for: " + this.BuoyLabel);
         lblDTG.setText("");
         lblSigWaveHt.setText("");
         pnlEnergy.removeAll();
         sliderTime.setEnabled(false);

     }
}

private void PlotDirectionalLegend(BuoyOb O) {

    Graphics g;
    g = pnlLegend.getGraphics();

    for (int i = 0; i < 300; i++) {
        double E = (double) (i / 4);
        g.setColor(O.getColor(E));
        g.drawRect(0, i, 30, 2);
        if (i % 20 == 0) g.drawString(Integer.toString((int)E), 35, i);
    }
    g.setColor(Color.WHITE);
    g.drawString("Dir Wave", 3, 325);
    g.drawString(" energy",3, 340);
    g.drawString("m^2/s", 5, 355);

}

private void PlotGraphicsAndLabels(BuoyOb O) {

    // Update and plot the date time group and significant wave height labels.
   // Initialize the Spectral sig wave height selector bar

    PlotDirectionalLegend(O);

    if (rbFrequency.isSelected() == true) {

        // If the user wants to view wave energy data in frequency space, then
        //  get the frequency bounds of the selected buoy's payload (found in the first
        //  and last spectral observation for the selected buoy).

        lblLowerVal.setText(ValToHundredths(O.getSpectralData().get(0).getFrequency()));

lblUpperVal.setText(ValToHundredths(O.getSpectralData().get(O.getSpectralData().size()-
1).getFrequency()));
        lblUnits.setText("Hz");

    }
    else {

        // If the user wants to view wave energy data in period space, then
        //  get the period bounds of the selected buoy's payload (found in the first
        //  and last spectral observation for the selected buoy). Note that this has
        //  been computed and is simply the inverse of the frequency.

        lblLowerVal.setText(ValToTenths(O.getSpectralData().get(0).getPeriod()));

lblUpperVal.setText(ValToTenths(O.getSpectralData().get(O.getSpectralData().size()-
1).getPeriod()));
        lblUnits.setText("sec");

    }

    // In all cases, compute and display the significant wave height for the buoy and
    //  the band specific sub value (which is initially equal to the total value). Then
create
    //  and display the energy chart for the current buoy observation.

    lblSubHt.setText(ValToTenths(ComputeSigWaveHeight(O, 0, O.getBands())));
    lblSigWaveHt.setText(ValToTenths(ComputeSigWaveHeight(O, 0, O.getBands())));
    CreateEnergyChart(O);
    CreateDirChart(O);
    UpdateDTGLabel(O);
```

```
    // cSel is a canvas on the bottom of the GUI that allows the user to select frequency
or
    //  period bands of interest for the current observation. Initially, this canvas
    //  is drawn in white.  Its color will change based on user interaction.

    this.LeftPoint = 0;
    this.RightPoint = 370;
    Graphics g = cSel.getGraphics();
    g.setColor(Color.WHITE);
    g.fillRect(0, 0, cSel.getWidth(), cSel.getHeight());

}

public String ValToTenths(double V) {

    // This is a utility method that takes a double value, rounds it to the nearest
tenth,
    //  then returns it as a string for eventual use as a label.

    double x = ((int)(V * 100.0) / 10) / 10.0;
    return Double.toString(x);
}

public String ValToHundredths(double V) {

    // This is a utility method that takes a double value, rounds it to the nearest
    //  hundredth, then returns it as a string for eventual use as a label.

    double x = ((int) (V * 1000.0) / 10) / 100.0;
    return Double.toString(x);
}

private double ComputeSigWaveHeight(BuoyOb O, int startbin, int endbin) {

    // this method takes a buoy observation, itegrates the area under the wave energy
    //  curve, and uses that value to determine significant wave height.  The significant
    //  wave height for a frequncy interval is the square root of the integral
    //  of the wave energy times the frequency band width times 4. This value is
    //  multiplied by 3.28084 to convert it from meters to feet.

    double m = 0.0;

    for (int i = startbin; i < endbin; i++) {
        SpectralOb S = O.getSpectralData().get(i);
        m += S.getEnergy() * S.getSpectralWidth();
    }
    return 3.28084 * 4.0 * Math.sqrt(m);
}



private void UpdateDTGLabel(BuoyOb O) {

    // This method takes a the date from the current buoy observation, converts
    //  it to a date/time string, and sets that string into the Date Time Group label
    //  on the GUI.

    String DATE = "";
    String D = DATE.format("%1$tm %1$te %1$tY / %1$tk Z", O.getDate());
    lblDTG.setText(D);
}

private void sliderTimeMouseHandler(java.awt.event.MouseEvent evt) {

    // When the user slides the time slider, this method responds to the mouse
    //  drag event by retrieving the value of the slider, using that to
    //  get a buoy observation for the currently selected buoy, then
    //  calling PlotGraphicsAndLabels(O) to do just that...

    if (this.CurrentBuoyObs.size() == 0)
        statusMessageLabel.setText("Select a buoy from the drop down menu");

    int thumbPos = sliderTime.getValue();
    O = CurrentBuoyObs.get(thumbPos);
    PlotGraphicsAndLabels(O);

}

private void ChangeChartType(java.awt.event.ActionEvent evt) {
```

```
    // This method responds to the user changing chart type (wave energy vs freq,
    //  wave energy vs pd, or sig wave height vs pd). It simply calls the method
    //  that redraws all graphics and labels.

    PlotGraphicsAndLabels(O);

}

private void SetBandLimits(java.awt.event.MouseEvent evt) {

    // This method responds to mouse drags on the cSel canvas area on the
    //  bottom of the GUI. The method paints the canvas based on user requests,
    //  determines the frequency bands corresponding to the click positions,
    //  determines the significant wave height for the selected band,
    //  then fills in the labels on the GUI as appropriate.

    // Get a graphics context of the cSel canvas to draw on.

    Graphics g = cSel.getGraphics();

    // get the locations of the drag on the canvas. the MetaDown
    //  event handler handles the click of the right mouse button
    //  while the else statement handles the drag of the left mouse button.
    //  The x-positions of the retrieved points are placed in the global
    //  integer RightPoint and LeftPoint variables as appropriate.

    if (evt.isMetaDown() == true) {

        this.RightPoint = evt.getX();

    }
    else {
        this.LeftPoint = evt.getX();
    }

    // Color everything to the left of LeftPoint and to the Right of RightPoint Blue
    // Color everything in between yellow.

    g.setColor(Color.BLUE);
    g.fillRect(0, 0, LeftPoint, cSel.getHeight());
    g.fillRect(RightPoint, 0, 370-RightPoint, cSel.getHeight());
    g.setColor(Color.YELLOW);
    g.fillRect(LeftPoint, 0, RightPoint-LeftPoint, cSel.getHeight());

    // if frequency is selected, figure out the bounding frequency bands associated
    // with the selected values.  These computations are handled in the body of the
    // following main if statement. The else statement below handles things in a
    // similar manner, but makes computations with respect to Period instead of
    // frequency.

    if (rbFrequency.isSelected() == true) {

        double botFreq = O.getSpectralData().get(0).getFrequency();
        double topFreq = O.getSpectralData().get(O.getSpectralData().size()-
1).getFrequency();
        double f1 = (topFreq/370.) * ((double) this.LeftPoint) + botFreq;
        double f2 = (topFreq/370.) * ((double) this.RightPoint) + botFreq;
        if (f1 < botFreq) f1 = botFreq;
        if (f2 > topFreq) f2 = topFreq;
        lblLowerVal.setText(ValToHundredths(f1));
        lblUpperVal.setText(ValToHundredths(f2));

        // Determine the bands that coorespond to the current selection.
        int lowerband = 0;
        int upperband = 0;
        boolean foundLower = false;
        boolean foundUpper = false;

        // First determine the lower band number

        for (int i = 0; i < O.getBands(); i++) {
            double freq = O.getSpectralData().get(i).getFrequency();
            if (freq >= f1) {
                if (foundLower == false) lowerband = i - 1;
                foundLower = true;
            }
        }
        if (lowerband < 0) lowerband = 0;

        // Now the upper band number
```

```
        for (int i = lowerband + 1; i < O.getBands(); i++) {
            double freq = O.getSpectralData().get(i).getFrequency();
            if (freq >= f2) {
                if (foundUpper == false) upperband = i - 1;
                foundUpper = true;
            }
        }
        if (upperband < lowerband) upperband = lowerband;

        // Now compute the significant wave height between these bands

        double wave = ComputeSigWaveHeight(O, lowerband, upperband);
        lblSubHt.setText(ValToTenths(wave));

        }
    else {

        double botPer = O.getSpectralData().get(0).getPeriod();
        double topPer = O.getSpectralData().get(O.getSpectralData().size()-
1).getPeriod();
        double p1 = (((double) this.LeftPoint)/370.0) * (topPer - botPer) + botPer;
        double p2 = (((double) this.RightPoint)/ 370.0) * (topPer - botPer) + botPer;
        if (p1 > botPer) p1 = botPer;
        if (p2 < topPer) p2 = topPer;
        lblLowerVal.setText(ValToTenths(p1));
        lblUpperVal.setText(ValToTenths(p2));

        // Determine the bands that coorespond to the current selection.
        int lowerband = 0;
        int upperband = 0;
        boolean foundLower = false;
        boolean foundUpper = false;

        // First the lower band number

        for (int i = 0; i < O.getBands(); i++) {
            double per = O.getSpectralData().get(i).getPeriod();
            if (per <= p1) {
                if (foundLower == false) lowerband = i - 1;
                foundLower = true;
            }
        }
        if (lowerband < 0) lowerband = 0;

        // Now the upper band number

        for (int i = lowerband + 1; i < O.getBands(); i++) {
            double per = O.getSpectralData().get(i).getPeriod();
            if (per <= p2) {
                if (foundUpper == false) upperband = i - 1;
                foundUpper = true;
            }
        }
        if (upperband < lowerband) upperband = lowerband;

        // Now compute the significant wave height between these bands

        double wave = ComputeSigWaveHeight(O, lowerband, upperband);
        lblSubHt.setText(ValToTenths(wave));

    }

}

private void ArrowKeys(java.awt.event.KeyEvent evt) {

    if (this.CurrentBuoyObs.size() == 0)
        statusMessageLabel.setText("Select a buoy from the drop down menu");

    int thumbPos = sliderTime.getValue();
    int key = evt.getKeyCode();
    if (key == evt.VK_LEFT) thumbPos--;
    if (key == evt.VK_RIGHT) thumbPos++;

    if (thumbPos < 0) thumbPos = 0;
    if (thumbPos >= this.CurrentBuoyObs.size()) thumbPos = this.CurrentBuoyObs.size() -
1;

    O = CurrentBuoyObs.get(thumbPos);
```

```
    PlotGraphicsAndLabels(O);
}

private ArrayList<BuoyOb> getBuoyObs(String L) {

    // This method takes a string that defines the buoy the user wants to view,
    //  then goes to tbe NDBC web site and pulls in 5 days of spectral wave
    //  data for the requested buoy. The file is read as a list of lines of data
    //  with each line representing a spectral band, and groups of 30-45 spectral
    //  bands representing an observation. The header line of a single observation
    //  defines the data and number of available spectral bands. RAW groups of strings\
    //  defining a single observation are sent to the BuoyOb constructor for decoding
    //  and construction. This function returns an ArrayList of buoy observations
    //  to the calling function.

    BufferedReader br = null;
    ArrayList<String> RAW = new ArrayList<String>();
    ArrayList<BuoyOb> decodedBuoyData = new ArrayList<BuoyOb>();

    decodedBuoyData.clear();

    // Read the 5 day file for the selected observation.

    try {
        URL url = new
                URL("http://www.ndbc.noaa.gov/data/5day2/" + L + "_5day.spectral");
        InputStream is = url.openStream();
        br = new BufferedReader(new InputStreamReader(is));
        String line = null;
        line = br.readLine();
        line = br.readLine();
        while ((line = br.readLine()) != null) {

            // Now Construct a RAW observation. This contains one header line
            //  that defines the date and number of spectral bands, then
            //  the frequency band, spectral width, wave energy, and
            //  Fourier coefficients of directional spectral.

            // Clear the RAW Array List
            RAW.clear();
            // Add the header line
            RAW.add(line);
            // determine the number of spectal obs there are to read:
            int spectralNumber = Integer.parseInt(line.substring(17).trim());
            for (int i = 0; i < spectralNumber; i++) {
                line = br.readLine();
                RAW.add(line);
            }
            // Create a buoy observation
            BuoyOb OB = new BuoyOb(RAW);
            decodedBuoyData.add(0, OB);
        }

    }
    catch (MalformedURLException e) {
        statusMessageLabel.setText("Can't get data for " + this.BuoyLabel);
    }
    catch (IOException e) {
        statusMessageLabel.setText("Problem reading data for " + this.BuoyLabel);
    }

    return decodedBuoyData;
}
private void CreateEnergyChart(BuoyOb O) {

    //This method creates an energy chart and place it onto the main application panel.

    // Note: The data set that is plotted will be dependent on the state of the
    //  radio button currently clicked. It can be wave energy vs frequency,
    //  wave energy vs period, or significant wave height (per frequency band) with
    //  respect to period.  O is the buoy observation to be worked with.

    // XYDataset is the actual data set plotted.
        XYDataset dataset;
        JFreeChart chart;
        if (rbFrequency.isSelected() == true) {
            // Create a data set that contains wave energy wrt frequency, then
            //  create the chart.

            dataset = createEnergyDataset(O, 0);
```

```
            chart = createChart(dataset, 0);
        }
        else if (rbPeriod.isSelected() == true) {
            // Create a data set that contains wave energy wrt period, then
            //  create the chart.

            dataset = createEnergyDataset(O, 1);
            chart = createChart(dataset, 1);
            chart.getXYPlot().getDomainAxis().setInverted(true);
        }
        else {
            // Create a data set that contains significant wave height with
            //  respect to period, then create the chart.

            dataset = createEnergyDataset(O, 2);
            chart = createChart(dataset, 2);
            chart.getXYPlot().getDomainAxis().setInverted(true);
             }

        // To make things pretty, paint a gradient paint behind the chart
        //  and set the antialias property for the chart to true.

        Paint P = new GradientPaint(0, 0, Color.white, 1000, 0, Color.green);
        chart.setBackgroundPaint(P);
        chart.setAntiAlias(true);

        ChartPanel chartPanel = new ChartPanel(chart);

        // remove any charts from the pnlEnergy container then add the chartpanel
        //  to pnlEnergy.  The size of the chart Panel is equal to the container.
        //  The removal prior to placing a new component is done to ensure no
        //  memory leaks or to ensure against excessive resource usage by the
        //  java virtual machine.

        pnlEnergy.removeAll();
        pnlEnergy.add(chartPanel);
        chartPanel.setSize(470, 320);
        chartPanel.setVisible(true);

}

private void CreateDirChart(BuoyOb O) {
    //  This method is called to Create a directional wave spectra display in the
    //    form of a Buffered Image

    // Instantiate a the dirPlot BufferedImage size 200 x 200
    dirPlot = new BufferedImage(300, 300, 1);
    Graphics2D d = dirPlot.createGraphics();

    if (rbFrequency.isSelected() == false) {

        for (int i = 0; i < 300; i++) {
            for (int j = 0; j < 300; j++) {
                int Direction = getDirAtPos(j, i);
                double Period = getPeriodAtPos(j, i);
                double Energy = O.getDirEnergyPd(Period, Direction);
                Color EColor = O.getColor(Energy);
                d.setColor(EColor);
                d.fillRect(j, i, 1, 1);

            }
        }
        PaintCompassLabels(d);
        PaintPeriodLabels(d);
    }
    else {

        for (int i = 0; i < 300; i++) {
            for (int j = 0; j < 300; j++) {
                int Direction = getDirAtPos(j, i);
                double Frequency = getFreqAtPos(j, i);
                double Energy = O.getDirEnergyFr(Frequency, Direction);
                Color EColor = O.getColor(Energy);
                d.setColor(EColor);
                d.fillRect(j, i, 1, 1);

            }
        }
        PaintCompassLabels(d);
        PaintFreqLabels(d);
```

```
    }

    Graphics P = pnlDir.getGraphics();
    pnlDir.paint(P);
}

private void PaintCompassLabels(Graphics2D d) {

    d.setColor(Color.WHITE);
    d.setFont(new Font("ARIAL", Font.BOLD, 14));
    d.drawString("N", 145, 10);
    d.drawString("S", 145, 300);
    d.drawString("E", 290, 155);
    d.drawString("W", 0, 155);
    d.drawString("NW", 30, 50);
    d.drawString("NE", 245, 45);
    d.drawString("SW", 30, 260);
    d.drawString("SE", 245, 260);
}

private void PaintPeriodLabels(Graphics2D d) {

    d.setColor(Color.RED);
    d.drawOval(116, 116, 68, 68);
    d.drawOval(82, 82, 136, 136);
    d.drawOval(48, 48, 204, 204);
    d.drawOval(14, 14, 272, 272);
    d.drawLine(150,15, 150, 285);
    d.drawLine(15, 150, 285, 150);
    d.drawLine(55, 55, 245, 245);
    d.drawLine(55, 245, 245, 55);
    d.setColor(Color.WHITE);
    d.drawString("5", 177, 140);
    d.drawString("10", 205, 130);
    d.drawString("15", 239, 120);
    d.drawString("20", 272, 110);
}

private void PaintFreqLabels(Graphics2D d) {

    d.setColor(Color.RED);
    d.drawOval(98, 98, 104, 104);
    d.drawOval(56, 56, 188, 188);
    d.drawOval(25, 25, 250, 250);
    d.drawOval(9, 9, 282, 282);
    d.drawLine(150, 15, 150, 285);
    d.drawLine(15, 150, 285, 150);
    d.drawLine(55, 55, 245, 245);
    d.drawLine(55, 245, 245, 55);
    d.setColor(Color.WHITE);
    d.drawString("3", 197, 140);
    d.drawString("5", 236, 130);
    d.drawString("10", 260, 120);
    d.drawString("20", 280, 115);

}

private void PaintPnlDir(Graphics g) {
    if (dirPlot != null) g.drawImage(dirPlot, 2, 2, pnlDir);
}

private int getDirAtPos(int x, int y) {

    // Given an x, y position on a 300 x 300 directional wave chart,
    //  this function computes the direction from the center of the
    //  chart in degrees from true north.  the following was
    //  derived from A dot B = |A||B| cos (theta)

    if ((x == 150) && (y <= 150)) return 0;
    if ((x == 150) && (y > 150)) return 180;

    double dir = (180.0/Math.PI) * (Math.acos((150.0 - (double)y) /
            Math.sqrt((150.0 - (double)x) * (150.0 - (double)x) + (150.0 - (double)y) *
(150.0 - (double)y))));
    if (x < 150) return 360 - (int) dir;
    else return (int) dir;

}

private double getPeriodAtPos(int x, int y) {
```

```
    // given an x, y position on a 300 x 300 directional wave chart,
    //  this function computes the period at that position, given that the
    //  period at the center of the display (150, 150) is 0 and the period
    //  at the outer edge to the North (150, 0) is 22 seconds.

    double DOrg = Math.sqrt((150 - x) * (150 - x) + (150 - y) * (150 - y));
    return (22.0 / 150.0) * DOrg;
}

private double getFreqAtPos(int x, int y) {

    // given an x, y position on a 300 x 300 directional wave chart,
    // this function computes the frequency at that position given that the
    // frequency at the center of the display (150, 150) is 0.50 hz and
    // the frequency at the outer edge to the North (150, 0) is 0.00 hz.

    double DOrg = Math.sqrt((150 - x) * (150 - x) + (150 - y) * (150 - y));
    return 0.50 - ((0.48 / 150.0) * DOrg);

}

private static XYDataset createEnergyDataset(BuoyOb O, int button) {

    // This method is a subclass of the XYDataset createDataSet method
    //  that is given in the sample code provided by JFreeChart. It takes
    //  a buoy observation (O) and an integer that defines what type of data
    //  (wave energy vs freq, wave energy vs pd, or wave ht vs pd) to be
    //  plotted.

    if (button == 0) {
        // Create a data set (series) that holds wave energy vs Freq.
        //  If energy data are missing (999.0) for a given spectral band,
        //  insert a null value into the data set.

        XYSeries series = new XYSeries("Wave Energy vs Frequency");
        for (int i = 0; i < O.getBands(); i++) {
            if (O.getSpectralData().get(i).getEnergy() >= 999.0)
                series.add(O.getSpectralData().get(i).getFrequency(),
                        null);
            else
                series.add(O.getSpectralData().get(i).getFrequency(),
                        O.getSpectralData().get(i).getEnergy());
        }
        return new XYSeriesCollection(series);
    }
    else if (button == 1) {
        // Create a series that defines wave energy vs period.

        XYSeries series = new XYSeries("Wave Energy vs Period");
        for (int i = 0; i < O.getBands(); i++) {
            if (O.getSpectralData().get(i).getEnergy() >= 999.0)
                series.add(O.getSpectralData().get(i).getPeriod(),
                        null);
            else
                series.add(O.getSpectralData().get(i).getPeriod(),
                        O.getSpectralData().get(i).getEnergy());

        }
        return new XYSeriesCollection(series);
    }
    else {
        // create a series that defines wave height vs period.

        XYSeries series = new XYSeries("Wave Period vs Period");
        for (int i = 0; i < O.getBands(); i++) {
            series.add(O.getSpectralData().get(i).getPeriod(),
                    O.getSpectralData().get(i).getSigWaveHeight());
        }
        return new XYSeriesCollection(series);
    }
}

private static JFreeChart createChart(XYDataset dataset, int button) {

    if (button == 0) {
        JFreeChart chart = ChartFactory.createXYLineChart(
            "Wave Energy vs Frequency",
            "Frequency (Hz)",
            "Energy (m**2/s)",
```

```
            dataset,
            PlotOrientation.VERTICAL,
            false,
            true,
            false
        );
       return chart;
    }

    else if (button == 1) {
        JFreeChart chart = ChartFactory.createXYLineChart(
            ”Wave Energy vs Period”,
            ”Period (seconds)”,
            ”Energy (m**2/s)”,
            dataset,
            PlotOrientation.VERTICAL,
            false,
            true,
            false
        );
    return chart;
    }
    else {
        JFreeChart chart = ChartFactory.createXYLineChart(
            ”Significant Wave Height”,
            ”Period (seconds)”,
            ”Significant Wave Height (ft)”,
            dataset,
            PlotOrientation.VERTICAL,
            false,
            true,
            false
        );
        return chart;
    }

}


    // Variables declaration - do not modify
    private javax.swing.ButtonGroup btnGrpEnergyPlotControls;
    private java.awt.Canvas cSel;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JLabel jLabel5;
    private javax.swing.JSeparator jSeparator1;
    private javax.swing.JSeparator jSeparator10;
    private javax.swing.JSeparator jSeparator11;
    private javax.swing.JSeparator jSeparator12;
    private javax.swing.JSeparator jSeparator13;
    private javax.swing.JSeparator jSeparator14;
    private javax.swing.JSeparator jSeparator15;
    private javax.swing.JSeparator jSeparator16;
    private javax.swing.JSeparator jSeparator17;
    private javax.swing.JSeparator jSeparator18;
    private javax.swing.JSeparator jSeparator19;
    private javax.swing.JSeparator jSeparator2;
    private javax.swing.JSeparator jSeparator20;
    private javax.swing.JSeparator jSeparator21;
    private javax.swing.JSeparator jSeparator22;
    private javax.swing.JSeparator jSeparator3;
    private javax.swing.JSeparator jSeparator4;
    private javax.swing.JSeparator jSeparator5;
    private javax.swing.JSeparator jSeparator6;
    private javax.swing.JSeparator jSeparator7;
    private javax.swing.JSeparator jSeparator8;
    private javax.swing.JSeparator jSeparator9;
    private javax.swing.JLabel lblBuoyName;
    private javax.swing.JLabel lblDTG;
    private javax.swing.JLabel lblLowerVal;
    private javax.swing.JLabel lblLowerVal2;
    private javax.swing.JLabel lblSigWaveHt;
    private javax.swing.JLabel lblSubHt;
    private javax.swing.JLabel lblUnits;
    private javax.swing.JLabel lblUpperVal;
    private javax.swing.JPanel mainPanel;
    private javax.swing.JMenuBar menuBar;
    private javax.swing.JMenuItem mnu41001;
```

```
private javax.swing.JMenuItem mnu41002;
private javax.swing.JMenuItem mnu41004;
private javax.swing.JMenuItem mnu41008;
private javax.swing.JMenuItem mnu41009;
private javax.swing.JMenuItem mnu41010;
private javax.swing.JMenuItem mnu41012;
private javax.swing.JMenuItem mnu41013;
private javax.swing.JMenuItem mnu41025;
private javax.swing.JMenuItem mnu41035;
private javax.swing.JMenuItem mnu41036;
private javax.swing.JMenuItem mnu41040;
private javax.swing.JMenuItem mnu41041;
private javax.swing.JMenuItem mnu41043;
private javax.swing.JMenuItem mnu41044;
private javax.swing.JMenuItem mnu41046;
private javax.swing.JMenuItem mnu41047;
private javax.swing.JMenuItem mnu41048;
private javax.swing.JMenuItem mnu41049;
private javax.swing.JMenuItem mnu42001;
private javax.swing.JMenuItem mnu42002;
private javax.swing.JMenuItem mnu42003;
private javax.swing.JMenuItem mnu42007;
private javax.swing.JMenuItem mnu42012;
private javax.swing.JMenuItem mnu42019;
private javax.swing.JMenuItem mnu42020;
private javax.swing.JMenuItem mnu42035;
private javax.swing.JMenuItem mnu42036;
private javax.swing.JMenuItem mnu42039;
private javax.swing.JMenuItem mnu42040;
private javax.swing.JMenuItem mnu42055;
private javax.swing.JMenuItem mnu42056;
private javax.swing.JMenuItem mnu42057;
private javax.swing.JMenuItem mnu42058;
private javax.swing.JMenuItem mnu42059;
private javax.swing.JMenuItem mnu42060;
private javax.swing.JMenuItem mnu42080;
private javax.swing.JMenuItem mnu44004;
private javax.swing.JMenuItem mnu44005;
private javax.swing.JMenuItem mnu44007;
private javax.swing.JMenuItem mnu44008;
private javax.swing.JMenuItem mnu44009;
private javax.swing.JMenuItem mnu44011;
private javax.swing.JMenuItem mnu44013;
private javax.swing.JMenuItem mnu44014;
private javax.swing.JMenuItem mnu44017;
private javax.swing.JMenuItem mnu44018;
private javax.swing.JMenuItem mnu44020;
private javax.swing.JMenuItem mnu44025;
private javax.swing.JMenuItem mnu44027;
private javax.swing.JMenuItem mnu44065;
private javax.swing.JMenuItem mnu45001;
private javax.swing.JMenuItem mnu45002;
private javax.swing.JMenuItem mnu45003;
private javax.swing.JMenuItem mnu45004;
private javax.swing.JMenuItem mnu45005;
private javax.swing.JMenuItem mnu45006;
private javax.swing.JMenuItem mnu45007;
private javax.swing.JMenuItem mnu45008;
private javax.swing.JMenuItem mnu45012;
private javax.swing.JMenuItem mnu46001;
private javax.swing.JMenuItem mnu46002;
private javax.swing.JMenuItem mnu46005;
private javax.swing.JMenuItem mnu46006;
private javax.swing.JMenuItem mnu46011;
private javax.swing.JMenuItem mnu46012;
private javax.swing.JMenuItem mnu46013;
private javax.swing.JMenuItem mnu46014;
private javax.swing.JMenuItem mnu46015;
private javax.swing.JMenuItem mnu46022;
private javax.swing.JMenuItem mnu46023;
private javax.swing.JMenuItem mnu46025;
private javax.swing.JMenuItem mnu46026;
private javax.swing.JMenuItem mnu46027;
private javax.swing.JMenuItem mnu46028;
private javax.swing.JMenuItem mnu46029;
private javax.swing.JMenuItem mnu46035;
private javax.swing.JMenuItem mnu46041;
private javax.swing.JMenuItem mnu46042;
private javax.swing.JMenuItem mnu46047;
private javax.swing.JMenuItem mnu46050;
```

```
    private javax.swing.JMenuItem mnu46053;
    private javax.swing.JMenuItem mnu46054;
    private javax.swing.JMenuItem mnu46059;
    private javax.swing.JMenuItem mnu46063;
    private javax.swing.JMenuItem mnu46066;
    private javax.swing.JMenuItem mnu46069;
    private javax.swing.JMenuItem mnu46070;
    private javax.swing.JMenuItem mnu46071;
    private javax.swing.JMenuItem mnu46072;
    private javax.swing.JMenuItem mnu46073;
    private javax.swing.JMenuItem mnu46075;
    private javax.swing.JMenuItem mnu46077;
    private javax.swing.JMenuItem mnu46078;
    private javax.swing.JMenuItem mnu46080;
    private javax.swing.JMenuItem mnu46082;
    private javax.swing.JMenuItem mnu46083;
    private javax.swing.JMenuItem mnu46084;
    private javax.swing.JMenuItem mnu46085;
    private javax.swing.JMenuItem mnu46086;
    private javax.swing.JMenuItem mnu46087;
    private javax.swing.JMenuItem mnu46088;
    private javax.swing.JMenuItem mnu46089;
    private javax.swing.JMenuItem mnu51000;
    private javax.swing.JMenuItem mnu51001;
    private javax.swing.JMenuItem mnu51002;
    private javax.swing.JMenuItem mnu51003;
    private javax.swing.JMenuItem mnu51004;
    private javax.swing.JMenu mnuAlaska;
    private javax.swing.JMenu mnuGulf;
    private javax.swing.JMenu mnuHawaii;
    private javax.swing.JMenu mnuLakes;
    private javax.swing.JMenu mnuNW;
    private javax.swing.JMenu mnuNWAtl;
    private javax.swing.JMenu mnuSW;
    private javax.swing.JMenu mnuWAtl;
    private javax.swing.JMenuItem mny44066;
    private javax.swing.JPanel pnlDir;
    private javax.swing.JPanel pnlEnergy;
    private java.awt.Panel pnlLegend;
    private javax.swing.JProgressBar progressBar;
    private javax.swing.JRadioButton rbFrequency;
    private javax.swing.JRadioButton rbPeriod;
    private javax.swing.JSlider sliderTime;
    private javax.swing.JLabel statusAnimationLabel;
    private javax.swing.JLabel statusMessageLabel;
    private javax.swing.JPanel statusPanel;
    // End of variables declaration

    private final Timer messageTimer;
    private final Timer busyIconTimer;
    private final Icon idleIcon;
    private final Icon[] busyIcons = new Icon[15];
    private int busyIconIndex = 0;

    private JDialog aboutBox;
}
```

-- *BuoyWavesApp.java*

```
/*
 * BuoyWavesApp.java
 */

package buoywaves;

import org.jdesktop.application.Application;
import org.jdesktop.application.SingleFrameApplication;

/**
 * The main class of the application.
 */
public class BuoyWavesApp extends SingleFrameApplication {

    /**
     * At startup create and show the main frame of the application.
     */
    @Override protected void startup() {
```

```
        show(new BuoyWavesView(this));
    }

    /**
     * This method is to initialize the specified window by injecting resources.
     * Windows shown in our application come fully initialized from the GUI
     * builder, so this additional configuration is not needed.
     */
    @Override protected void configureWindow(java.awt.Window root) {
    }

    /**
     * A convenient static getter for the application instance.
     * @return the instance of BuoyWavesApp
     */
    public static BuoyWavesApp getApplication() {
        return Application.getInstance(BuoyWavesApp.class);
    }

    /**
     * Main method launching the application.
     */
    public static void main(String[] args) {
        launch(BuoyWavesApp.class, args);
    }
}
```

-- *BuoyWavesAboutBox.java*

```
/*
 * BuoyWavesAboutBox.java
 */

package buoywaves;

import org.jdesktop.application.Action;

public class BuoyWavesAboutBox extends javax.swing.JDialog {

    public BuoyWavesAboutBox(java.awt.Frame parent) {
        super(parent);
        initComponents();
        getRootPane().setDefaultButton(closeButton);
    }

    @Action public void closeAboutBox() {
        setVisible(false);
    }

    /** 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">
    private void initComponents() {

        closeButton = new javax.swing.JButton();
        javax.swing.JLabel appTitleLabel = new javax.swing.JLabel();
        javax.swing.JLabel versionLabel = new javax.swing.JLabel();
        javax.swing.JLabel appVersionLabel = new javax.swing.JLabel();
        javax.swing.JLabel vendorLabel = new javax.swing.JLabel();
        javax.swing.JLabel appVendorLabel = new javax.swing.JLabel();
        javax.swing.JLabel homepageLabel = new javax.swing.JLabel();
        javax.swing.JLabel appHomepageLabel = new javax.swing.JLabel();
        javax.swing.JLabel appDescLabel = new javax.swing.JLabel();
        javax.swing.JLabel imageLabel = new javax.swing.JLabel();

        setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE_ON_CLOSE);
        org.jdesktop.application.ResourceMap resourceMap =
          org.jdesktop.application.Application.getInstance(buoywaves.
            BuoyWavesApp.class).getContext().getResourceMap(BuoyWavesAboutBox.class);
        setTitle(resourceMap.getString("title"));
        setBackground(resourceMap.getColor("aboutBox.background"));
        setForeground(resourceMap.getColor("aboutBox.foreground"));
        setModal(true);
        setName("aboutBox");
```

```
setResizable(false);

javax.swing.ActionMap actionMap =
  org.jdesktop.application.Application.getInstance(buoywaves.BuoyWavesApp.class)
  .getContext().getActionMap(BuoyWavesAboutBox.class, this);
closeButton.setAction(actionMap.get("closeAboutBox"));
closeButton.setName("closeButton");

appTitleLabel.setFont(appTitleLabel.getFont().deriveFont(appTitleLabel
  .getFont().getStyle() | java.awt.Font.BOLD, appTitleLabel.getFont()
  .getSize()+4));
appTitleLabel.setText(resourceMap.getString("Application.title"));
appTitleLabel.setName("appTitleLabel");

versionLabel.setFont(versionLabel.getFont().deriveFont(versionLabel
  .getFont().getStyle() | java.awt.Font.BOLD));
versionLabel.setText(resourceMap.getString("versionLabel.text"));
versionLabel.setName("versionLabel");

appVersionLabel.setText(resourceMap.getString("Application.version"));
appVersionLabel.setName("appVersionLabel");

vendorLabel.setFont(vendorLabel.getFont().deriveFont(vendorLabel.getFont()
  .getStyle() | java.awt.Font.BOLD));
vendorLabel.setText(resourceMap.getString("vendorLabel.text"));
vendorLabel.setName("vendorLabel");

appVendorLabel.setText(resourceMap.getString("Application.vendor"));
appVendorLabel.setName("appVendorLabel");

homepageLabel.setFont(homepageLabel.getFont().deriveFont(homepageLabel
  .getFont().getStyle() | java.awt.Font.BOLD));
homepageLabel.setText(resourceMap.getString("homepageLabel.text"));
homepageLabel.setName("homepageLabel");

appHomepageLabel.setText(resourceMap.getString("Application.homepage"));
appHomepageLabel.setName("appHomepageLabel");

appDescLabel.setText(resourceMap.getString("appDescLabel.text"));
appDescLabel.setName("appDescLabel");

imageLabel.setIcon(resourceMap.getIcon("imageLabel.icon"));
imageLabel.setName("imageLabel");

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
getContentPane().setLayout(layout);
layout.setHorizontalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(layout.createSequentialGroup()
        .addComponent(imageLabel)
        .addGap(28, 28, 28)
        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout
        .Alignment.TRAILING)
            .addGroup(javax.swing.GroupLayout.Alignment.LEADING,
               layout.createSequentialGroup()
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout
                  .Alignment.LEADING)
                    .addComponent(versionLabel)
                    .addComponent(vendorLabel)
                    .addComponent(homepageLabel))
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement
                   .RELATED)
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout
                     .Alignment.LEADING)
                    .addComponent(appVersionLabel)
                    .addComponent(appVendorLabel)
                    .addComponent(appHomepageLabel)))
            .addComponent(appTitleLabel,
               javax.swing.GroupLayout.Alignment.LEADING)
            .addComponent(appDescLabel,
                javax.swing.GroupLayout.Alignment.LEADING,
                javax.swing.GroupLayout.DEFAULT_SIZE, 418, Short.MAX_VALUE)
            .addComponent(closeButton))
        .addContainerGap())
);
layout.setVerticalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(layout.createSequentialGroup()
        .addContainerGap()
        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout
```

```
                    .Alignment.LEADING)
                     .addComponent(imageLabel, javax.swing.GroupLayout.PREFERRED_SIZE,
                        146, javax.swing.GroupLayout.PREFERRED_SIZE)
                     .addGroup(layout.createSequentialGroup()
                         .addComponent(appTitleLabel)
                         .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement
                            .RELATED)
                         .addComponent(appDescLabel)
                         .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement
                           .RELATED)
                         .addGroup(layout.createParallelGroup(javax.swing.GroupLayout
                            .Alignment.BASELINE)
                             .addComponent(versionLabel)
                             .addComponent(appVersionLabel))
                         .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement
                            .RELATED)
                         .addGroup(layout.createParallelGroup(javax.swing.GroupLayout
                            .Alignment.BASELINE)
                             .addComponent(vendorLabel)
                             .addComponent(appVendorLabel))
                         .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement
                            .RELATED)
                         .addGroup(layout.createParallelGroup(javax.swing.GroupLayout
                            .Alignment.BASELINE)
                             .addComponent(homepageLabel)
                             .addComponent(appHomepageLabel))
                         .addGap(125, 125, Short.MAX_VALUE)
                         .addComponent(closeButton)))
                .addContainerGap())
        );

        pack();
    }// </editor-fold>

    // Variables declaration - do not modify
    private javax.swing.JButton closeButton;
    // End of variables declaration

}
```