

problem in java swings

Source: <http://coding.derkeiler.com/Archive/Java/comp.lang.java.programmer/2006-05/msg00509.html>

- *From:* "sweta" <sweta_gupta_56@xxxxxxxxxxxx>
 - *Date:* 5 May 2006 06:08:11 -0700
-

I Have problem related to Internal Frame
actually In Internal Frame one list is attached in which 1 to 19
options exists
In Internal Frame one graph is displaying in which changes occurs with
respect to only that option which we have already selected in
list3.java and peakContainer.java in which Instance of list3 is passed
file but I Want changes with respect to every Option(1 to 19)in
Internal Frame's graph ,sir help me.Actually I am working in
NMR(nuclear Magnetic Resonance) project I am attaching code Below
(NMRUI.java is main file) actually when we click on file menu and then
in open option in browsing window we select p38400.7 type file and then
internal frame will open

```
import java.awt.*;
import javax.swing.*;
import javax.swing.event.*;
public class list3 extends JPanel
{
public int Frame_index;
public list3(int Frame_index)
{
this.Frame_index = Frame_index;
initComponents();
}

private void initComponents() {
scrollPane1 = new JScrollPane();
list = new JList();

//===== this =====

setLayout(new FlowLayout());

//===== scrollPane1 =====
{

//---- list ----
list.setSelectionMode(ListSelectionModel.SINGLE_SELECTION);
list.setModel(new AbstractListModel() {
String[] values = {
```

problem in java swings

```
"1","2","3","4",
"5",
"6",
"7",
"8",
"9",
"10",
"11",
"12",
"13",
"14",
"15",
"16","17","18","19"
};
public int getSize() { return values.length; }
public Object getElementAt(int i) { return values[i]; }
});
list.setVisibleRowCount(19);
list.setSelectedIndex(Frame_index);
list.addListSelectionListener(new ListSelectionListener() {
public void valueChanged(ListSelectionEvent e) {
listValueChanged(e);
}
});
scrollPane1.setViewportView(list);
}
add(scrollPane1);
// JFormDesigner – End of component initialization
//GEN-END:initComponents
}

public void listValueChanged(ListSelectionEvent e) {
// TODO add your code here
JList jl = (JList)e.getSource();
this.Frame_index = jl.getSelectedIndex();
}

private JScrollPane scrollPane1;
private JList list;
}

//peakContainer.java
/*
 * Created on Aug 26, 2004
 *
 */
//package nmrui.mri;

import java.awt.BorderLayout;
import java.awt.event.ActionEvent;
```

problem in java swings

```
import java.awt.event.ActionListener;
import javax.swing.ButtonGroup;
import javax.swing.JInternalFrame;
import javax.swing.JMenu;
import javax.swing.JMenuBar;
import javax.swing.JMenuItem;
import javax.swing.JPopupMenu;
import javax.swing.JRadioButtonMenuItem;
import javax.swing.JRootPane;
import javax.swing.Icon;
import javax.swing.ImageIcon;
import java.io.*;
import java.awt.event.*;
import java.awt.*;
import javax.swing.JOptionPane;//new code for pope-up
import javax.swing.*;
import javax.swing.event.*;
/*
 * @ Author ANADI MISHRA
 *
 * @ version 1.00
 */
public class PeakContainer extends JInternalFrame
{
    JMenuBar menuBar;
    JMenu fileMenu, dispMenu, psMenu;
    JMenuItem[] fmi;
    JRadioButtonMenuItem[] mmi, tmi, psi;
    JRootPane rootPane;
    public final int inset = 50;
    ButtonGroup mbg, tbg, psbg;
    JRadioButtonMenuItem sb; //new code
    static final Integer DOCLAYER = new Integer(5);
    GraphGE gr;
    GraphDataGE1 gd;
    PeakSupres ps;
    //XAxismod ax; // for pope-up

    JPopupMenu m_menu = new JPopupMenu(); //new codes
    double max, min; // for pope-up
    double xvalue;
    boolean linePresent = false;
    boolean popupPresent = false;
    String msg=" ";
    public int frameindex = 6;
    public PeakContainer()
    {
        super();
    }
}
/*
 * @ s the name of the GE File
```

problem in java swings

```
* @ data the double array containing file's values
*/
public PeakContainer(String s, double[][] data)
{
    super(s);
    final int point = data.length/19;
    rootPane = this.getRootPane();
    //JPopupMenu.setDefaultLightWeightPopupEnabled(false);

    menuBar = new JMenuBar();
    menuBar.setOpaque(true);

    /******* codes for pope-up
    *****/
    JMenuItem m_pv = new JMenuItem("Previous view");
    JMenuItem m_iv = new JMenuItem("Initial view");
    JMenuItem m_szr = new JMenuItem("Set zero reference");
    JMenuItem m_sr = new JMenuItem("Set reference");

    m_menu.add (m_pv);
    m_menu.add (m_iv);

    // Separator
    m_menu.add (new JMenuItem("-"));

    m_menu.add (m_szr);
    m_menu.add (m_sr);
    add(m_menu);

    addMouseListener ( new MouseAdapter ()
    {
        public void mouseReleased(MouseEvent e)
        {
            // Check for a right click pop-up
            if (e.isPopupTrigger() )
            {
                int x = e.getX();
                int y = e.getY();
                popupPresent = true;

                genppm_frq pmfr = new genppm_frq(point);
                double[] ppm = pmfr.returnppm();
                max_min mm = new max_min();
                max = mm.getMax(ppm);
                min = mm.getMin(ppm);
                xvalue = -(min+((720-(x-80))*(max-min)/720));
                // Show the popup meun
                m_menu.show (rootPane, x, y );
            }
        }
    });
```

problem in java swings

```
m_sr.addActionListener(new PopupMenuListener(this));
```

```
/**************************************************************************
```

```
fileMenu = new JMenu("File");  
fileMenu.setMnemonic('F');
```

```
fmi = new JMenuItem[3];
```

```
fmi[0] =  
new JMenuItem(  
"Open",  
(Icon) new ImageIcon("./resources/images/open.gif"));  
fmi[1] =  
new JMenuItem(  
"Save",  
(Icon) new ImageIcon("./resources/images/save.gif"));  
fmi[2] = new JMenuItem("Exit");
```

```
for (int i = 0; i < 3; i++)  
fileMenu.add(fmi[i]);
```

```
for (int i = 0; i < 3; i++)  
fmi[i].addActionListener(new GraphContainerListener(this));
```

```
dispMenu = new JMenu("Display");  
dispMenu.setMnemonic('D');
```

```
mbg = new ButtonGroup();  
tbg = new ButtonGroup();
```

```
mmi = new JRadioButtonMenuItem[2];  
tmi = new JRadioButtonMenuItem[7];
```

```
mmi[0] = new JRadioButtonMenuItem("FID");  
mmi[0].setActionCommand("fid");  
mmi[1] = new JRadioButtonMenuItem("Spectrum", true);  
mmi[1].setActionCommand("sp");  
tmi[0] = new JRadioButtonMenuItem("Real", true);  
tmi[0].setActionCommand("r");  
tmi[1] = new JRadioButtonMenuItem("Imaginary");  
tmi[1].setActionCommand("i");  
tmi[2] = new JRadioButtonMenuItem("Absolute");  
tmi[2].setActionCommand("a");  
tmi[3] = new JRadioButtonMenuItem("Real^2");  
tmi[3].setActionCommand("r2");  
tmi[4] = new JRadioButtonMenuItem("Imaginary^2");  
tmi[4].setActionCommand("i2");  
tmi[5] = new JRadioButtonMenuItem("Absolute^2");  
tmi[5].setActionCommand("a2");  
tmi[6] = new JRadioButtonMenuItem("Mag[Mag-real]");
```

problem in java swings

problem in java swings

```
tmi[6].setActionCommand("msr");

//MyListSelectionListener l2=new MyListSelectionListener();
//JListSimpleExample1 l1=new JListSimpleExample1();

for (int i = 0; i < 2; i++)
{
dispMenu.add(mmi[i]);
mbg.add(mmi[i]);
}

for (int i = 0; i < 2; i++)
mmi[i].addActionListener(new GraphContainerListener(this));

dispMenu.addSeparator();

for (int i = 0; i < 7; i++)
{
dispMenu.add(tmi[i]);
tbg.add(tmi[i]);
}

for (int i = 0; i < 7; i++)
{
tmi[i].addActionListener(new GraphContainerListener(this));
}

//new code
//tmi[i].addMouseListener(new handelmouse(this));
//tmi[i].addMouseMotionListener(new handelmouse(this));
}

////###////////////////////////////////////
psMenu = new JMenu("PeakSup");
psMenu.setMnemonic('P');

psbg = new ButtonGroup();
psi = new JRadioButtonMenuItem[2];

psi[0] = new JRadioButtonMenuItem("Water Peak");
psi[0].setActionCommand("wps");
psi[1] = new JRadioButtonMenuItem("Viewps");
psi[1].setActionCommand("vps");

for (int i = 0; i < 2; i++)
{
psMenu.add(psi[i]);
psbg.add(psi[i]);
}

for (int i = 0; i < 2; i++)
```

problem in java swings

problem in java swings

```
psi[i].addActionListener(new GraphContainerListener(this));
```

```
////###////////////////////////////////////
```

```
//new code
```

```
sb = new JRadioButtonMenuItem("SaveData", (Icon) new
```

```
ImageIcon("./resources/images/save.gif"));
```

```
sb.setActionCommand("savedata");
```

```
sb.addActionListener(new GraphContainerListener(this));
```

```
menuBar.add(fileMenu);
```

```
menuBar.add(dispMenu);
```

```
menuBar.add(psMenu);
```

```
menuBar.add(sb);//new code
```

```
rootPane.setJMenuBar(menuBar);
```

```
setClosable(true);
```

```
setIconifiable(true);
```

```
setMaximizable(false);
```

```
setResizable(false);
```

```
setVisible(true);
```

```
this.setBounds(10,10,600,400);
```

```
/*
```

```
* Adding the default Graph
```

```
*/
```

```
list3 l3 = new list3(frameindex);
```

```
rootPane.getContentPane().add(l3, BorderLayout.EAST);
```

```
gd = new GraphDataGE1(data.length);
```

```
if (data != null)
```

```
{
```

```
//choicetest t=new choicetest();
```

```
//msg+=t.cbg.getSelectedCheckbox().getLabel();
```

```
//System.out.println("index @nkur:"+l1.getIndex());
```

```
//System.out.println(l3.index);
```

```
gd.addSignals(data,frameindex);
```

```
gd.doPhaseFID();
```

```
gd.doFFT();
```

```
gd.doPhase();
```

```
}
```

```
else
```

```
{
```

```
System.out.println("Cannot Proceed\n System Shall Exit");
```

```
System.exit(0);
```

```
}
```

```
gr = new GraphGE(gd.getPoint(), s);
```

problem in java swings

problem in java swings

```
gr.clearValues();
gr.setValues(gd.getReal());
gr.reDraw();

rootPane.getContentPane().add(gr, BorderLayout.CENTER);
//ax = new XAxismod(gd.getPoint())

}

public GraphGE getGraphGE()
{
return this.gr;
}

public GraphDataGE1 getGraphDataGE()
{
return this.gd;
}

/*public XAxismod gettrans()
{
return this.ax; //pope-up
}*/

/*public PeakSupres getPeakSupres()
{
return this.ps;
}*/

public void message()
{
System.out.println("Hello!");
}

public PeakContainer getPeakContainer()
{
return this;
}

public int[] getSelectedMenu()
{
int [] idx = new int[4];

if(mmi[0].isSelected())
idx[0] = 0;
else
idx[0] = 1;

for(int i=0;i<7;i++)
{
if(tmi[i].isSelected())
```

problem in java swings

```
{
idx[1] = i;
}
}

if(sb.isSelected())
idx[2] = 0;

for(int i=0;i<2;i++)
{
if(psi[i].isSelected())
{
idx[3] = i;
}
}
return idx;
}

}

#####//////////////////////////////////new code for
popup//////////////////////////////////
class PopupMenuListener implements ActionListener
{
PeakContainer cp;
int [] idx = new int[2];
public PopupMenuListener(PeakContainer cp)
{
this.cp = cp;
}

public void actionPerformed(ActionEvent e)
{
GraphGE grp = cp.getGraphGE();
GraphDataGE1 gdp = cp.getGraphDataGE();
int pt = gdp.getPoint();

if(cp.popupPresent)
{
String Input = JOptionPane.showInputDialog("Enter first integer");
double inputnumber = Double.parseDouble(Input);
double d = cp.xvalue;
System.out.println(inputnumber);
System.out.println(d);
double numsub = inputnumber - d;
double numsubfrq = (inputnumber * 63.86) - d;

ppm_freq_save pfs = new ppm_freq_save(pt, numsub,
numsubfrq);
pfs.writefrq();
```

problem in java swings

```
idx = cp.getSelectedMenu();
//System.out.println("Pope-up present"+idx[0]);

if(idx[0] == 0)
{
int ch = idx[1];
switch (ch)
{
case 0: // Real
grp.clearValues();
grp.setValues(gdp.getRealFID());
grp.settrans(numsub);
cp.repaint();
grp.reDraw();
break;
case 1: //Imag
grp.clearValues();
grp.setValues(gdp.getImagFID());
grp.settrans(numsub);
cp.repaint();
grp.reDraw();
break;
case 2: //Absolute
grp.clearValues();
grp.setValues(gdp.getMagnitudeFID());
grp.settrans(numsub);
cp.repaint();
grp.reDraw();
break;
case 3: //Real Square
grp.clearValues();
grp.setValues(gdp.getRealFIDSq());
grp.settrans(numsub);
cp.repaint();
grp.reDraw();
break;
case 4 : // Imag Square
grp.clearValues();
grp.setValues(gdp.getImagFIDSq());
grp.settrans(numsub);
cp.repaint();
grp.reDraw();
break;
case 5:
grp.clearValues();
grp.setValues(gdp.getMagnitudeFIDSq());
grp.settrans(numsub);
cp.repaint();
grp.reDraw();
break;
}
```

problem in java swings

```
}
else if(idx[0] == 1)
{
int ch = idx[1];

switch (ch)
{
case 0: // Real
grp.clearValues();
grp.setValues(gdp.getReal());
grp.settrans(numsub);
cp.repaint();
grp.reDraw();
break;
case 1: //Imag
grp.clearValues();
grp.setValues(gdp.getImag());
grp.settrans(numsub);
cp.repaint();
grp.reDraw();
break;
case 2: //Absolute
grp.clearValues();
grp.setValues(gdp.getMagnitude());
grp.settrans(numsub);
cp.repaint();
grp.reDraw();
break;
case 3: //Real Square
grp.clearValues();
grp.setValues(gdp.getRealSq());
grp.settrans(numsub);
cp.repaint();
grp.reDraw();
break;
case 4 : // Imag Square
grp.clearValues();
grp.setValues(gdp.getImagSq());
grp.settrans(numsub);
cp.repaint();
grp.reDraw();
break;
case 5:
grp.clearValues();
grp.setValues(gdp.getMagnitudeSq());
grp.settrans(numsub);
cp.repaint();
grp.reDraw();
break;
}
}
```

problem in java swings

problem in java swings

```
}  
  
}  
  
}  
#####  
  
class GraphContainerListener implements ActionListener  
{  
    PeakContainer cg;  
    String str = new String("");  
    public GraphContainerListener(PeakContainer cg)  
    {  
        this.cg = cg;  
    }  
  
    public String getActionString()  
    {  
        return str;  
    }  
  
    public void actionPerformed(ActionEvent evt)  
    {  
        str = evt.getActionCommand();  
        GraphGE gr = cg.getGraphGE();  
        GraphDataGE1 d = cg.getGraphDataGE();  
        //PeakSupres s = cg.getPeakSupres();  
  
        fileHandling fw = new fileHandling();  
        File fp = new File(".");  
        String savepath = fp.getAbsolutePath();  
        String foldername = "\\Result\\";  
  
        if (str.equals("fid"))  
        {  
            return;  
        }  
        else if (str.equals("sp"))  
        {  
            return;  
        }  
        if (str.equals("r"))  
        {  
            gr.clearValues();  
            if(cg.mmi[1].isSelected())  
                gr.setValues(d.getReal());  
            else  
                gr.setValues(d.getRealFID());  
            cg.repaint();  
            gr.reDraw();  
        }  
    }  
}
```

```
}
else if (str.equals("i"))
{
gr.clearValues();
if(cg.mmi[1].isSelected())
gr.setValues(d.getImag());
else
gr.setValues(d.getImagFID());
cg.repaint();
gr.reDraw();
}
else if (str.equals("a"))
{
gr.clearValues();
if(cg.mmi[1].isSelected())
gr.setValues(d.getMagnitude());
else
gr.setValues(d.getMagnitudeFID());
cg.repaint();
gr.reDraw();
}
else if (str.equals("r2"))
{
gr.clearValues();
if(cg.mmi[1].isSelected())
gr.setValues(d.getRealSq());
else
gr.setValues(d.getRealFIDSq());
cg.repaint();
gr.reDraw();
}
else if (str.equals("i2"))
{
gr.clearValues();
if(cg.mmi[1].isSelected())
gr.setValues(d.getImagSq());
else
gr.setValues(d.getImagFIDSq());
cg.repaint();
gr.reDraw();
}
else if (str.equals("a2"))
{
gr.clearValues();
if(cg.mmi[1].isSelected())
gr.setValues(d.getMagnitudeSq());
else
gr.setValues(d.getMagnitudeFIDSq());
cg.repaint();
gr.reDraw();
}
```

```

}

else if (str.equals("msr"))
{
gr.clearValues();
if(cg.mmi[1].isSelected())
gr.setValues(d.getMagofMagSubRe());
else
gr.setValues(d.getMagofMagSubReFID());
cg.repaint();
gr.reDraw();
}

#####
else if (str.equals("wps"))
{

PeakSupres s = new PeakSupres(d.getReal(),d.getImag());
double[] pcreal = s.getpsreal();

//System.out.println(pcreal[0]);
gr.clearValues();
if(cg.mmi[1].isSelected())
gr.setValues(pcreal);
//System.out.println("Error1");
else
System.out.println("Error2");
cg.repaint();
gr.reDraw();
}

else if (str.equals("vps"))
{
Frame_ParalleDisplay pf = new Frame_ParalleDisplay(d.getReal());
//Frame f = new Frame_ParalleDisplay();
System.out.println("ViewWP");
}

else if (str.equals("savedata"))
{
/*try
{
//File fp = new File(".");
//String savepath = fp.getAbsolutePath();
String foldername1 = "\\BMP_Graph\\";
File file = new File(savepath+foldername1);
file.mkdir();
File filef = new File(savepath+foldername1+"Mag.bmp");
Filesave sf = new Filesave();
sf.saveToFile(cg, filef);
}
}

```

problem in java swings

```
System.out.println("Done saving File!");
}
catch(Exception e)
{ }*/

if(cg.sb.isSelected())
{
double[] pcreal = d.getReal();
double[] pcimag = d.getImag();
double[] pcrealtd = d.getRealFID();
double[] pcimagtd = d.getImagFID();
fw.filewrite(savepath, pcreal,"pcreal",foldername);
fw.filewrite(savepath, pcimag,"pcimag",foldername);
fw.filewrite(savepath, pcrealtd,"pcrealtd",foldername);
fw.filewrite(savepath, pcimagtd,"pcimagtd",foldername);
System.out.println("Done saving File!");
}
}
}
}
```

```
//NMRGUI.java
```

```
/*
```

```
* Created on Aug 25, 2004
```

```
*
```

```
*/
```

```
//package nmru.gui;
```

```
import java.awt.BorderLayout;
import java.awt.Container;
import java.awt.Color;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.KeyEvent;
import java.awt.event.KeyListener;
import java.awt.event.WindowAdapter;
import java.awt.event.WindowEvent;
import java.awt.GraphicsConfiguration;
import java.awt.HeadlessException;
import java.awt.Insets;
import java.io.File;
import javax.swing.border.EtchedBorder;
import javax.swing.Icon;
import javax.swing.ImageIcon;
import javax.swing.JButton;
import javax.swing.JComponent;
import javax.swing.JCheckBoxMenuItem;
import javax.swing.JDesktopPane;
import javax.swing.JFrame;
import javax.swing.JMenuBar;
import javax.swing.JMenu;
```

problem in java swings

```
import javax.swing.JMenuItem;
import javax.swing.JPanel;
import javax.swing.JRootPane;
import javax.swing.JToolBar;
import javax.swing.UIManager;
import java.awt.*;
import java.awt.Toolkit;
//import nmrui.mrs.GraphContainer;
//import nmrui.mri.PeakContainer;
//import views.*;
/**
 * @author ANADI MISHRA
 */
public class NMRGui extends JFrame {
    private JRootPane rootPane;
    private Container cot;
    private JMenuBar menuBar;
    private JMenu[] menu;
    private JMenuItem[] fmi, emi, dmi, qmi, wmi;
    private JPanel iPanel, up;
    private JToolBar tBar;
    private JButton[] tlb = new JButton[7];
    private JDesktopPane dpane;
    private PeakContainer p;
    static final Integer layer1 = new Integer(0);
    static final Integer layer2 = new Integer(1);
    static final Integer layer3 = new Integer(2);
    /**
     * @throws java.awt.HeadlessException
     */
    public NMRGui() throws HeadlessException {
        super();
    }

    /**
     * @param gc
     */
    public NMRGui(GraphicsConfiguration gc) {
        super(gc);
    }

    /**
     * @param title
     * @param gc
     */
    public NMRGui(String title) {
        super(title);
        try
        {
            UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());
        }
    }
}
```

problem in java swings

```
catch (Exception e)
{
System.err.println("Error In Look And Feel :" + e);
e.printStackTrace();
}

rootPane = getRootPane();
//extracting the content-pane
cot = rootPane.getContentPane();
cot.setBackground(Color.white);
//cot.setLayout(new BorderLayout());

//activating close window button
addWindowListener(new WindowAdapter() {
public void windowClosing(WindowEvent we) {
System.exit(0);
}
});

addWindowListener(new WindowAdapter() {
public void windowActivated(WindowEvent we) {
}
});

/*
@The following Code is to Add both the Menu bar and the Tool Bar
* to this upper panel in BorderLayout so that they cover the entire
* panel

*/
//creating the menu bar
menuBar = new JMenuBar();
menuBar.setBorder(new EtchedBorder(EtchedBorder.RAISED));
rootPane.setJMenuBar(menuBar);

//creating upper panel
JPanel up = new JPanel(new BorderLayout(0, 1));
cot.add(up, BorderLayout.NORTH,0);

// creating Menus
menu = new JMenu[8];

menu[0] = new JMenu("File", true);
menu[0].setMnemonic('F');

menu[1] = new JMenu("Edit");
menu[1].setMnemonic('E');

menu[2] = new JMenu("Display");
menu[2].setMnemonic('D');
```

problem in java swings

```
menu[3] = new JMenu("Preprocessing");
menu[3].setMnemonic('P');

menu[4] = new JMenu("Quantification");
menu[4].setMnemonic('Q');

menu[5] = new JMenu("Options");
menu[5].setMnemonic('O');

menu[6] = new JMenu("Window");
menu[6].setMnemonic('W');

menu[7] = new JMenu("Help");
menu[7].setMnemonic('H');

for (int i = 0; i < 8; i++)
    menuBar.add(menu[i]);

//Creating Submenus for File

JMenuItem[] fmi = new JMenuItem[6];

fmi[0] = new JMenuItem("Show Wizard ");
fmi[0].setActionCommand("Show");
fmi[1] = new JMenuItem("Open Ctrl-o", KeyEvent.VK_O);
fmi[1].setActionCommand("Open");
/*fmi[1].addActionListener(new ActionListener()
{
    public void actionPerformed(ActionEvent e)
    {
        final Frame c=new choicetest();
        final Frame d=new Test();
        Dimension screensize=Toolkit.getDefaultToolkit().getScreenSize();
        d.setBounds(0,0,screensize.width-150,screensize.height-150);
        d.setVisible(true);

        c.setBounds(200,200,200,200);
        c.setVisible(true);

    }
});*/
fmi[2] = new JMenuItem("Open as ", KeyEvent.VK_P);
//fmi[2].setEnabled(false);
fmi[2].setActionCommand("Openas");
fmi[3] = new JMenuItem("Save Ctrl-s", KeyEvent.VK_S);
//fmi[3].setEnabled(false);
fmi[3].setActionCommand("Save");
fmi[4] = new JMenuItem("Save as ", KeyEvent.VK_A);
//fmi[4].setEnabled(false);
fmi[4].setActionCommand("Saveas");
fmi[5] = new JMenuItem("Exit Ctrl-x", KeyEvent.VK_X);
```

problem in java swings

```
fmi[5].setActionCommand("Exit");

for (int i = 0; i < 6; i++) {
    menu[0].add(fmi[i]);
    if (i == 0 || i == 4)
        menu[0].addSeparator();
    }
//adding Listeners to the MenuItems

for (int i = 0; i < 6; i++) {
    fmi[i].addActionListener(new NMRGuiListener(this));
    fmi[i].addKeyListener(new NMRGuiKeyListener(this));
    }

//Creating Submenus for edit

JMenuItem[] emi = new JMenuItem[2];

emi[0] = new JMenuItem(" Undo ", KeyEvent.VK_U);
menu[1].add(emi[0]);
emi[1] = new JMenuItem(" Redo ", KeyEvent.VK_R);
menu[1].add(emi[1]);

// adding listeners
for (int i = 0; i < 2; i++) {
    emi[i].addActionListener(new NMRGuiListener(this));
    emi[i].addKeyListener(new NMRGuiKeyListener(this));
    }

//Creating SubMenu For Display

JCheckBoxMenuItem[] dmi = new JCheckBoxMenuItem[4];

dmi[0] = new JCheckBoxMenuItem("Phase Window", false);
dmi[1] = new JCheckBoxMenuItem("Aphodize Window", false);
dmi[2] = new JCheckBoxMenuItem("Zero Fitting Window", false);
dmi[3] = new JCheckBoxMenuItem("Preprocessing Window", false);

for (int i = 0; i < 4; i++) {
    menu[2].add(dmi[i]);
    if (i == 2)
        menu[2].addSeparator();
    }

//adding listsners
for (int i = 0; i < 4; i++) {
    dmi[i].addActionListener(new NMRGuiListener(this));
    dmi[i].addKeyListener(new NMRGuiKeyListener(this));
    }

//creating submenu for Preprocessing
```

problem in java swings

```
JMenuItem[] pmi = new JMenuItem[2];

//creating submenus for Quantification

JMenuItem[] qmi = new JMenuItem[2];

qmi[0] = new JMenuItem(" PSVD ", KeyEvent.VK_P);
menu[4].add(qmi[0]);
qmi[1] = new JMenuItem(" HSVD ", KeyEvent.VK_H);
menu[4].add(qmi[1]);

//adding Listeners
for (int i = 0; i < 2; i++) {
qmi[i].addActionListener(new NMRGuiListener(this));
qmi[i].addKeyListener(new NMRGuiKeyListener(this));
}

//creating submenu for options menu

JMenuItem[] omi = new JMenuItem[2];

//creating submenu for Window Menu

JMenuItem[] wmi = new JMenuItem[2];

wmi[0] = new JMenuItem("Minimize All Windows", KeyEvent.VK_N);
menu[6].add(wmi[0]);
wmi[1] = new JMenuItem("Close All Windows", KeyEvent.VK_C);
menu[6].add(wmi[1]);

//adding listeners
for (int i = 0; i < 2; i++) {
wmi[i].addActionListener(new NMRGuiListener(this));
wmi[i].addKeyListener(new NMRGuiKeyListener(this));
}

//creating the icon

Icon open = new ImageIcon("./resources/images/OPEN.GIF");
Icon save = new ImageIcon("./resources/images/SAVE.GIF");
Icon print = new ImageIcon("./resources/images/PRINT.GIF");
Icon wiz = new ImageIcon("./resources/images/wizardMenu2.gif");
Icon phase = new ImageIcon("./resources/images/phase.gif");
Icon apodize = new ImageIcon("./resources/images/apodize.gif");
Icon zero = new ImageIcon("./resources/images/zero.gif");

// adding the toolbar

JToolBar tBar = new JToolBar();
tBar.setMargin(new Insets(5, 5, 5, 5));
```

problem in java swings

problem in java swings

```
tBar.setFloatable(false);
up.add(tBar, BorderLayout.NORTH);
//adding button to the toolbar

tlb[0] = new JButton(wiz);
tlb[0].setToolTipText("Show Wizard");
tlb[0].setActionCommand("wiz");
tlb[1] = new JButton(open);
tlb[1].setToolTipText(" Open ");
tlb[1].setActionCommand("open");
tlb[2] = new JButton(save);
tlb[2].setToolTipText(" Save ");
tlb[2].setActionCommand("save");
tlb[3] = new JButton(print);
tlb[3].setToolTipText(" Print ");
tlb[3].setActionCommand("print");
tlb[4] = new JButton(phase);
tlb[4].setToolTipText(" Phase Window ");
tlb[4].setActionCommand("phase");
tlb[5] = new JButton(apodize);
tlb[5].setToolTipText(" Apodize Window ");
tlb[5].setActionCommand("apo");
tlb[6] = new JButton(zero);
tlb[6].setToolTipText(" Zero Fitting Window ");
tlb[6].setActionCommand("zero");

for (int i = 0; i < 7; i++) {
    tBar.add(tlb[i]);
    if (i == 3)
        for (int j = 0; j < 3; j++)
            tBar.addSeparator();
}

//adding listeners
for (int i = 0; i < 7; i++)
    tlb[i].addActionListener(new NMRGuiButtonListener(this));

//create and add the DesktopPane
dpane = new JDesktopPane();
rootPane.getContentPane().add(dpane,1);
dpane.setBackground(Color.white);
}
public void addScreenGraph(String s, int type, double[][] data) {
    if (type == 1) {
        System.out.println("Text file Container is not here");
    } else {
        PeakContainer pfr = new PeakContainer(s, data);
        if(cot.getComponentCount() < 2){

rootPane.getContentPane().add(dpane,1);
dpane.setBackground(Color.white);
```

problem in java swings

```
}
p = pfr.getPeakContainer();
dpane.add(pfr, layer1);
}
}
/**
 *
 * @param p
 */
public void addView(Phaseview p){
dpane.add(p,layer2);
}
/**
 *
 * @param a
 */
public void addView(Apodizeview a){
dpane.add(a,layer2);
}

/**
 *
 * @return
 */
public PeakContainer getPeakContianer(){

if(p == null){
System.out.println("Fatal : Object Not Initialized");
}
return p;
}

/**
 *
 *
 */
public void activateControls() {
for (int i = 2; i < 6; i++)
fmi[i].setEnabled(true);
for (int i = 2; i < 6; i++)
tlb[i].setEnabled(true);
}
}

class NMRGuiListener implements ActionListener {

private NMRGui cot;
private File file;
private FileHandler fh;
private Phaseview p;
private Apodizeview a;
```

problem in java swings

problem in java swings

```
private PeakContainer pc;

public NMRGuiListener(NMRGui cot) {
this.cot = cot;
}

public void actionPerformed(ActionEvent evt) {
JComponent comp = (JComponent) evt.getSource();
fh = new FileHandler(cot);

if ((evt.getActionCommand().equals("Open"))
|| (evt.getActionCommand().equals("Openas"))) {
//cot.activateControls();
fh.openFile();
} else if (evt.getActionCommand().equals("Save")) {
if (file == null)
fh.callSave();
else
fh.saveFile(file);
} else if (evt.getActionCommand().equals("Saveas"))
fh.callSaveAs();
else if(evt.getActionCommand().equals("Phase Window")){

pc = cot.getPeakContianer();
p= new Phaseview("Phasing",pc);
cot.addView(p);
}
else if(evt.getActionCommand().equals("Apodize Window")){
pc = cot.getPeakContianer();
a= new Apodizeview("Apodize",pc);
cot.addView(a);
}
else if (evt.getActionCommand().equals("Exit"))
System.exit(0);

}
}

class NMRGuiButtonListener implements ActionListener {
private NMRGui cot;
private FileHandler fh;
private Phaseview p;
private Apodizeview a;
private PeakContainer pc;

public NMRGuiButtonListener(NMRGui cot) {
this.cot = cot;
}

public void actionPerformed(ActionEvent evt) {
String str = evt.getActionCommand();
```

problem in java swings

```
fh = new FileHandler(cot);
```

```
if (str.equals("open"))
fh.openFile();
else if (str.equals("save"))
fh.callSave();
else if(str.equals("phase")){
pc = cot.getPeakContianer();
p = new Phaseview("Phasing",pc);
cot.addView(p);
}
else if(str.equals("apo")){
pc = cot.getPeakContianer();
a= new Apodizeview("Apodize",pc);
cot.addView(a);
}
}
}
```

```
class NMRGuiKeyListener implements KeyListener
{
```

```
private NMRGui cot;
private FileHandler fh;
```

```
public NMRGuiKeyListener(NMRGui cot) {
this.cot = cot;
}
```

```
public void keyPressed(KeyEvent ke) {
}
public void keyReleased(KeyEvent ke) {
}
public void keyTyped(KeyEvent ke) {
}

}
```

```
main file is NMRUI.java
//import nmrui.gui.nmrgui;
import java.awt.toolkit;
import java.awt.dimension;
```

```
public class nmru {
private static final int inset = 50;
```

```
public static void main(string[] args) {
```

```
nmrgui win = new nmrgui("nmr interface");
dimension screensize = toolkit.getDefaultToolkit().getscreensize();
```

problem in java swings

problem in java swings

```
win.setBounds(0, 0, screensize.width, screensize.height);
win.setVisible(true);
}
}
```

```
/*
 * Created on Aug 26, 2004
 *
 */
//package nmrui.mri;

import java.util.Vector;
//import java.io.*;
import fft.*;
/*
 * @author ANADI MISHRA
 *
 * @version 1.00
 */

public class GraphDataGE1
{
double[] real;
double[] imag;
double[] real_fd;
double[] imag_fd;
double[] realTD;
double[] imagTD;
double [][] signal;
double phi;
int originalLength = 0;
Vector setOfFIDs,setOFSpectra;
////////###////////////////////////////////////
double a;
double dw;
double d;
////////###////////////////////////////////////
public GraphDataGE1(int size)
{
originalLength = size/19;
//originalLength = size/11;
System.out.println("originalLength"+originalLength);
//originalLength = size;
real = new double[originalLength];
imag = new double[originalLength];
real_fd = new double[originalLength];
imag_fd = new double[originalLength];
realTD = new double[originalLength];
imagTD = new double[originalLength];
}
```

problem in java swings

problem in java swings

```
signal = new double[originalLength][2];
phi=0.0D;
///////#####
a=0.0D;
dw = 0.4;
d = 0;
///////#####
setOfFIDs = new Vector();
setOfSpectra = new Vector();
}

/**
 * @param s
 */
/**public void addSignals(double[][] s)
 {
 for(int i=0;i<originalLength;i++)
 {
 signal[i][0] = s[i][0];
 signal[i][1] = s[i][1];
 }
 }*/

public void addSignals(double[][] s,int frameindex)
 {
 //int ctr = 0;
 //int ctr = originalLength;
 int ctr = (frameindex-1)*originalLength;
 //int ctr = 4*originalLength;
 for(int k=0;k<1;k++)
 //for(int k=0;k<1;k++)
 //for(int k=1;k<3;k++)
 for (int l = 0; l < originalLength; l++)
 {
 signal[l][0] += s[ctr][0];
 signal[l][1] += s[ctr][1];
 ctr++;
 }

 }

public void doPhaseFID()
 {
 for(int i=0; i<originalLength; i++)
 {
 realTD[i]= (Math.cos(Math.toRadians(phi))) * signal[i][0] +
 (Math.sin(Math.toRadians(phi))) * signal[i][1];
 imagTD[i]= -(Math.sin(Math.toRadians(phi))) * signal[i][0] +
 (Math.cos(Math.toRadians(phi))) * signal[i][1];
 }
 }
 }
```

problem in java swings

```
#####  
/*public void doApodizeFID()  
{  
for(int i=0; i<originalLength; i++)  
{  
  
realTD[i]= (Math.exp(-a*i*dw)) * realTD[i];  
imagTD[i]= (Math.exp(-a*i*dw)) * imagTD[i];  
}  
}*/  
  
#####  
  
public void doFFT()  
{  
Fft fft = new Fft();  
double [][] input_fft = new double[2][originalLength];  
for(int i=0; i<input_fft[0].length; i++)  
{  
input_fft[0][i] = signal[i][0];  
input_fft[1][i] = signal[i][1];  
}  
  
double [][] estimate = fft.direct_1D_Estim_fft(input_fft);  
//double [][] output_fft = fft.calcFFTD(estimate,true);  
for (int i = 0; i < originalLength; i++)  
{  
real_fd[i] = estimate[0][i];  
imag_fd[i] = estimate[1][i];  
}  
}  
  
public int getPoint()  
{  
return originalLength;  
}  
#####  
  
public double geta()  
{  
return a;  
}  
  
public void seta(double a)  
{  
this.a = a;  
}  
  
public void doApodize()  
{
```

problem in java swings

```
Fft fft = new Fft();
double [][] input_fft_ap = new double[2][originalLength];
for(int i=0;i<input_fft_ap[0].length;i++)
{
d = 1/(a * dw);
//double bell = (d*d)/(((Math.PI)*i*i) + d*d); // For Lorentzian
//double bell = Math.exp((-i*i)/(2*d*d)); // For Gaussian
double bell = Math.exp((i*i)/(2*d*d));
//input_fft_ap[0][i] =(Math.exp(-a*i*dw))*realTD[i];
//input_fft_ap[1][i] = (Math.exp(-a*i*dw))*imagTD[i];
input_fft_ap[0][i] = bell*realTD[i];
input_fft_ap[1][i] = bell*imagTD[i];
}
double [][] estimate_ap =
fft.direct_1D_Estim_fft(input_fft_ap);
for (int i = 0; i < originalLength; i++)
{
real[i] = estimate_ap[0][i];
imag[i] = estimate_ap[1][i];
}
}

#####
public double getPhi()
{
return phi;
}

public void setPhi(double phi)
{
this.phi = phi;
}

public void doPhase()
{
for(int i=0; i<originalLength; i++)
{
real[i]= (Math.cos(Math.toRadians(phi))) * real_fd[i] +
(Math.sin(Math.toRadians(phi))) * imag_fd[i];
imag[i]= -(Math.sin(Math.toRadians(phi))) * real_fd[i] +
(Math.cos(Math.toRadians(phi))) * imag_fd[i];
}
}

public double[][] getSignal()
{
return signal;
}

public double[] getReal()
{
```

```
return real;
}

public double[] getRealFID()
{
return realTD;
}

public double[] getImag()
{
return imag;
}

public double[] getImagFID()
{
return imagTD;
}

public int getoriginalLength()
{
return originalLength;
}

public double[] getRealSq()
{
double[] sq = new double[real.length];

for (int i = 0; i < real.length; i++)
sq[i] = Math.pow(real[i], 2.0);

return sq;
}

public double[] getRealFIDSq()
{
double[] sq = new double[realTD.length];

for (int i = 0; i < realTD.length; i++)
sq[i] = Math.pow(realTD[i], 2.0);

return sq;
}

public double[] getImagSq()
{
double[] sq = new double[imag.length];

for (int i = 0; i < imag.length; i++)
sq[i] = Math.pow(imag[i], 2.0);

return sq;
}
```

```

}

public double[] getImagFIDSq()
{
double[] sq = new double[imagTD.length];

for (int i = 0; i < imagTD.length; i++)
sq[i] = Math.pow(imagTD[i], 2.0);

return sq;
}

public double[] getMagnitude()
{
double[] mag = new double[originalLength];
for (int i = 0; i < originalLength; i++)
mag[i] = Math.sqrt((Math.pow(real[i], 2.0) + Math.pow(imag[i],
2.0)));

return mag;
}

public double[] getMagnitudeFID()
{
double[] mag = new double[originalLength];
for (int i = 0; i < originalLength; i++)
mag[i] = Math.sqrt((Math.pow(realTD[i], 2.0) + Math.pow(imagTD[i],
2.0)));

return mag;
}

public double[] getMagnitudeSq()
{
double[] mag = getMagnitude();
double[] magsq = new double[originalLength];
for (int i = 0; i < originalLength; i++)
{
magsq[i] = Math.pow(mag[i], 2.0);
}

return magsq;
}

public double[] getMagnitudeFIDSq()
{
double[] mag = getMagnitudeFID();
double[] magsq = new double[originalLength];
for (int i = 0; i < originalLength; i++) {
magsq[i] = Math.pow(mag[i], 2.0);
}
}

```

problem in java swings

```
return magsq;
}

public double[] getMagofMagSubRe()
{
double[] magmsr = new double[originalLength];
double[] mag = new double[originalLength];
for (int i = 0; i < originalLength; i++)
{
mag[i] = Math.sqrt((Math.pow(real[i], 2.0) + Math.pow(imag[i],
2.0)));
magmsr[i] = Math.sqrt(Math.pow((mag[i] - real[i]), 2.0));
}

return magmsr;
}

public double[] getMagofMagSubReFID()
{
double[] magmsr = new double[originalLength];
double[] mag = new double[originalLength];
for (int i = 0; i < originalLength; i++)
{
mag[i] = Math.sqrt((Math.pow(realTD[i], 2.0) + Math.pow(imagTD[i],
2.0)));
magmsr[i] = Math.sqrt(Math.pow((mag[i] - real[i]), 2.0));
}

return magmsr;
}

}
/*
 * Created on Aug 26, 2004
 *
 */
//package nmruim.mri;
/*
 * @ Author Manoj Sarma
 *
 * @version 1.00
 */

import javax.swing.JPanel;
import java.awt.Color;
import java.awt.Graphics;
import java.awt.event.*;
import java.io.*;
```

problem in java swings

```
public class GraphGE extends JPanel
{
// handelmouse hm;
//int mouseX, mouseY;
//String msg;
int point = 0;
double numsub = 0D;
double xmax, ymax, xmin, ymin;

double [] yVal;
double[] yVal1;
int[] yVal2;
int[] yVal3;

double[] xVal;
int[] xVal1;
int[] xVal2;

String name;
//XAxis xaxis;
XAxismod xaxis;
//YAxis yaxis;
YAxismod yaxis;

////###////////////////////////////////////

boolean tracking = false;
boolean mouseresele = false;
int startX = 0;
int startY = 0;
int currentX = 0;
int currentY = 0;
////###////////////////////////////////////

/*
 * @param pt gives the length of the array
 * @param s gives the name of the GE File
 */
GraphGE(int pt, String s)
{
point = pt;
System.out.println("Point"+point);
name = s;
yVal = new double[point];
yVal1 = new double[point];
yVal2 = new int[point];
yVal3 = new int[point];
xVal = new double[point];
xVal1 = new int[point];
xVal2 = new int[point];
```

problem in java swings

```
xmax = ymax = xmin = ymin = 0.0;

setBackground(new Color(255, 255, 255));
setForeground(new Color(0, 0, 0));
setOpaque(true);

/*###//////////////////////////////////////
this.addMouseListener (new mousePressHandler());
this.addMouseListener (new mouseReleaseHandler());
this.addMouseMotionListener (new mouseMotionHandler());

//###////////////////////////////////////*/
}

/*###//////////////////////////////////////
void mouseMotion(int x, int y)
{
if(tracking)
{
currentX = x;
currentY = y;
}
requestFocus();
repaint();
}

void startMotion(int x, int y)
{
tracking = true;
startX = x;
startY = y;
currentX = x+4;
currentY = y+4; // nonzero size, may choose to ignore later
requestFocus();
repaint();
}

void stopMotion(int x, int y)
{
tracking = false; // no more rubber_rect
// save final figure data for 'display' to draw
currentX = x;
currentY = y;
requestFocus();
repaint();
}

class mousePressHandler extends MouseAdapter
{
public void mousePressed (MouseEvent e)
```

problem in java swings

```
{
int b, x, y;
b = e.getButton();
x = e.getX();
y = e.getY();
System.out.println("press x="+x+" y="+y+" b="+b); // debug
print
if(b==1) startMotion(x, y);
}
}
```

```
class mouseReleaseHandler extends MouseAdapter
{
public void mouseReleased (MouseEvent e)
{
int b, x, y;
b = e.getButton();
x = e.getX();
y = e.getY();
System.out.println("release x="+x+" y="+y+" b="+b); //
debug print
System.out.println("Mouse Starting:"+(startX-80));
System.out.println("Mouse Ending:"+(x-80));
int sp = (int)((2047/900)*(startX-80));
int ep = (int)((2047/900)*(x-80)); // Check these instead of
2047 put point
//scx = new double[(ep-sp)+1];
//scy = new double[(ep-sp)+1];
//System.out.println(scx.length);
for(int i= sp; i<=ep; i++)
{
//scx[i-sp] = ppm[i];
//scy[i-sp] = real[i];
//System.out.println(scx[i-sp]);
}

if(b==1) stopMotion(x, y);
mouseRelease = true;
}
}
```

```
class mouseMotionHandler extends MouseMotionAdapter
{
public void mouseDragged (MouseEvent e)
{
int b, x, y;
b = e.getButton();
x = e.getX();
y = e.getY();
System.out.println("motion x="+x+" y="+y+" b="+b); // debug
print
```

problem in java swings

```
mouseMotion(x, y);
}
}

void rubberRect(Graphics g, int x0, int y0, int x1 , int y1)
{
// can apply to all figures
// draw a rubber rectangle, mouse down, tracks mouse
int x,y,x2,y2,x3,y3; // local coordinates
x2=x0;
x3=x1;
if(x1<x0) {x2=x1; x3=x0;};
y2=y0;
y3=y1;
if(y1<y0) {y2=y1; y3=y0;};
g.setColor(Color.black);
for(x=x2; x<x3-3; x=x+8)
{
g.drawLine(x, y0, x+4, y0);
g.drawLine(x, y1, x+4, y1);
}
for(y=y2; y<y3-3; y=y+8)
{
g.drawLine(x0, y, x0, y+4);
g.drawLine(x1, y, x1, y+4);
}
}
}
////////###////////////////////////////////////*/

public void paint(Graphics g)
{
g.setColor(Color.black);
g.drawLine(80, 440, 800, 440);
g.drawLine(80, 35, 80, 440);
//new code
//yaxis = new YAxis(this);
//yaxis.displayY(g);
xaxis = new XAxismod(point, numsub);
yaxis = new YAxismod(ymax, ymin);

//x-axis scale plotting
xaxis.displayppm(g);

//y-axis scale plotting
yaxis.displayY(g);

g.setColor(Color.blue);
//if(tracking) rubberRect(g, startX, startY, currentX,
currentY);
```

problem in java swings

```
for (int p = 0; p <(point-1); p++)
{
g.drawLine((xVal2[p]),yVal3[p),(xVal2[p + 1]),yVal3[p + 1]);
}
}

public void update(Graphics g)
{
paintImmediately(0,0,800,500);
paint(g);
}
/*
* @param g The current graphics object context
*/
private void drawMargins(Graphics g)
{
}
/*
* @param g the current graphics object context
*/
private void drawValues(Graphics g)
{
}

public void reDraw()
{
MinMax();
scale();
repaint();
}

public void clearValues()
{

for(int x = 0; x < point ; x++)
{
yVal1[x] = 0;
yVal2[x] = 0;
yVal3[x] = 0;
xVal[x] = 0.0;
xVal1[x] = 0;
xVal2[x] = 0;
}
ymax = ymin = 0;
xmax = xmin =0.0;
}
/*
* @param arr the double array whose value is to be stored for y –
axis
*/
public void setValues(double[] arr)
```

problem in java swings

```
{
for(int i=0;i<point;i++)
yVal[i] = arr[i];

for (int x = 0; x < point; x++)
{
xVal[x] = (double)x;
}

for (int l = 0; l < point; l++)
{
yVal1[l] = arr[l];
}

}

public double gettrans()
{
return numsub;
}
public void settrans(double numsub)
{
this.numsub = numsub;
//numb = numsub;
}

/*
 * @ Provides with the minimum and maximum values
 */
private void MinMax()
{
ymin = ymax = yVal1[0];
for (int i = 0; i < point; i++)
{
if (yVal1[i] > ymax)
ymax = yVal1[i];
if (yVal1[i] < ymin)
ymin = yVal1[i];

xmin = 0.0;
xmax = 2047.0;
}

}
/*
 * @ scales the array to screen size
 */
private void scale() {
for (int k = 0; k < point; k++) {
yVal2[k] = (int) (((yVal1[k] - ymin) / (ymax - ymin)) * 400);
xVal1[k] = (int) (((xVal[k] - xmin) / (xmax - xmin)) * 720);
```

problem in java swings

```
}

for (int l = 0; l < point; l++)
{
yVal3[l] = 440 - yVal2[l];
xVal2[l] = 80 + xVal1[l];
}

}

public double[] getMarginY()
{
return yVal;
}
//new code
/* public void start()
{
}
class handelmouse extends MouseAdapter implements MouseMotionListener
{
public void mouseMoved(MouseEvent me)
{
mouseX = me.getX();
mouseY = me.getY();
repaint();
reDraw();
}
public void mouseDragged(MouseEvent me)
{
}
public void mousePressed(MouseEvent me)
{
}
}
*/

}import java.awt.image.*;
import java.awt.*;
import java.util.*;
import java.io.*;

public class PeakSupres
{
GraphDataGE gd;
int totalpoint = 2048;
int point = 2048;
//int point = 300, p =750;
int pointarpk = point;
//int pointarpk = 3;
int iter = 15;
int p = 0;
```

problem in java swings

```

int l= 0;
int countloop = 0;
double[] iterreal;
double[] iterimag;
double[] ppm;
double ampl, dc, fr, ph;

public PeakSupres(double[] realpart, double[] imagpart)
{
File fp = new File(".");
String savepath = fp.getAbsolutePath();
String foldername = "\\Result\\";
try
{
PrintStream ps1 = new PrintStream(new
FileOutputStream(savepath+foldername+"\\Out.log", true) ,true);
System.setOut(ps1);
PrintStream ps2 = new PrintStream(new
FileOutputStream(savepath+foldername+"\\Error.txt", true) ,true);
//Out.Err
System.setErr(ps2);
}
catch(IOException e){ };

iterreal = new double[totalpoint];
iterimag = new double[totalpoint];
double x1=0, x2=0, x3=0, x4=0;
double max, min, max1, min1, max2, min2;
double[] real = new double[point];
double[] imag = new double[point];
double[] imag1 = new double[pointarpk];
double[] R = new double[pointarpk];
double[] Rdesh = new double[pointarpk];
double[] real1 = new double[pointarpk];
double[] qt = new double[pointarpk];
double[] ee = new double[pointarpk];
double[] f = new double[pointarpk];
double[] g = new double[pointarpk];
double[] h= new double[pointarpk];
double[] realdiffsq = new double[pointarpk];
double[] realdiffsqsum = new double[iter];
double[] realsq = new double[pointarpk];

double[] itRp = new double[totalpoint];
double[] itIp = new double[totalpoint];
double[] itratpeaks = new double[totalpoint];
double[] itiatpeaks = new double[totalpoint];

double[] I = new double[pointarpk];
double[] Idesh = new double[pointarpk];
double[] imag2 = new double[point];

```

problem in java swings

```
double[] imag3 = new double[point];
double[] imagdiffsq = new double[pointarpk];
double[] realimagdiffsqadd = new double[pointarpk];
double[] imagdiff = new double[pointarpk];
double[] imagsq = new double[pointarpk];
double[] imagdiffsqsum = new double[iter];
```

```
double[] edevwamp = new double[pointarpk];
double[] edevwdc = new double[pointarpk];
double[] edevwfreq = new double[pointarpk];
double[] edevwphase = new double[pointarpk];
double[] fdevwamp = new double[pointarpk];
double[] fdevwdc = new double[pointarpk];
double[] fdevwfreq = new double[pointarpk];
double[] fdevwphase = new double[pointarpk];
double[] gdevwamp = new double[pointarpk];
double[] gdevwdc = new double[pointarpk];
double[] gdevwfreq = new double[pointarpk];
double[] gdevwphase = new double[pointarpk];
double[] hdevwamp = new double[pointarpk];
double[] hdevwdc = new double[pointarpk];
double[] hdevwfreq = new double[pointarpk];
double[] hdevwphase = new double[pointarpk];
```

```
double[] amplp = new double[iter+1];
double[] dcp = new double[iter+1];
double[] frdp = new double[iter+1];
double[] phasp = new double[iter+1];
```

```
double[] Fdesh = new double[iter];
double[] FuncE = new double[iter];
double[] FuncF = new double[iter];
double[] FuncG = new double[iter];
double[] FuncH = new double[iter];
double[] deltaamp = new double[iter];
double[] deltadcp = new double[iter];
double[] deltafrd = new double[iter];
double[] deltaphas = new double[iter];
```

```
double[] DevFuncE = new double[4];
double[] DevFuncF = new double[4];
double[] DevFuncG = new double[4];
double[] DevFuncH = new double[4];
double[][] J = new double[4][4];
double[][] Func = new double[4][1];
double[][] b = new double[4][1];
```

```
double[] Fdeshexp = new double[iter+1];
double[] ppm = new double[totalpoint];
double[] freq = new double[totalpoint];
```

problem in java swings

```
double[] output = new double[4];
double[] realdata = new double[realpart.length];
double[] imagdata = new double[realpart.length];

/*if (data != null)
{
gd.addSignals(data);
gd.doPhaseFID();
gd.doFFT();
gd.doPhase();
}
else
{
System.out.println("Cannot Proceed\n System Shall Exit");
System.exit(0);
}*/

/*public GraphDataGE getGraphDataGE()
{
return this.gd;
}*/

//double[] realdata = gd.getReal();
//double[] imagdata = gd.getImag();
for(int i = 0 ; i <realpart.length; i++)
{
realdata[i] = realpart[i];
imagdata[i] = imagpart[i];
}

File fp1 = new File(".");
String path = fp1.getAbsolutePath();

mainfileread mffr = new mainfileread("C:\\Documents and
Settings\\manoj\\Desktop\\ApodizeMRS\\Result\\nfreq.txt");
freq = mffr.returnarray();

mainfileread mfp = new mainfileread("C:\\Documents and
Settings\\manoj\\Desktop\\ApodizeMRS\\Result\\nppm.txt");
ppm = mfp.returnarray();

for(int i = 0 ; i <point; i++)
{
real[i] = realdata[i+p];
imag[i] = imagdata[i+p];
}
for(int i = 0 ; i <point; i++)
{
```

problem in java swings

```
real[i] = realdata[i+p];
imag[i] = imagdata[i+p];
}
```

```
max_min mm = new max_min();
max = mm.getMax(real);
min = mm.getMin(real);
```

```
int count=0;
int w=0, w1=0, w2;
double frd=0.0, length;
for(int i = 0 ; i < point; i++)
{
if(real[i] == max)
{
w= count+p;
System.out.println(" array no" + " " +w);
System.out.println("imaginary" + " " + imag[count]);
frd = freq[count+p];
System.out.println("Height" + " " + max);
System.out.println("Frequency" + " " + frd);
}
count++;
}
```

```
ArrayList X = new ArrayList();
ArrayList Y = new ArrayList();
System.out.println("");
double frahh=0.0, frahh1=0.0, frahh2=0.0;
```

```
for(int i = 0 ; i <point-1; i++)
{
if(((real[i] - (max/2))>0) && ((real[i+1] - (max/2))<0))
{
X.add(new Integer(Math.abs(w-(i+p))));
}
```

```
if(((real[i] - (max/2))<0) && ((real[i+1] - (max/2))>0))
{
Y.add(new Integer(Math.abs(w-(i+p))));
}
}
```

```
//*****
```

```
Object[] X1 = X.toArray();
int min3 =((Integer) X1[0]).intValue();
for(int i = 0 ; i <X1.length;i++)
{
if( ((Integer) X1[i]).intValue()<min3)
min3 = ((Integer) X1[i]).intValue();
}
```

problem in java swings

```

//*****
int r = X1.length;
double frahh5 = 0.0;
for(int i = 0 ; i <point-1; i++)
{
if(((real[i] - (max/2D))>0) && ((real[i+1] - (max/2D))<0) &&
(Math.abs(w-(i+p))) == min3)
{
w1 = i+p;
frahh5 = (freq[i+p] + freq[i+p+1])/2D;
System.out.println(" array no1" + " " + w1 + " " + " array
no2" + " " + (w1+1));
}
}

//*****
Object[] Y1 = Y.toArray();
int min4 =((Integer) Y1[0]).intValue();
for(int i = 0 ; i <Y1.length;i++)
{
if( ((Integer) Y1[i]).intValue(<min4)
min4 = ((Integer) Y1[i]).intValue();
}
}
//*****
int r1 = Y1.length;
double frahh6 = 0.0;
for(int i = 0 ; i <point-1; i++)
{
if(((real[i] - (max/2))<0) && ((real[i+1] - (max/2))>0) &&
(Math.abs(w-(i+p))) == min4)
{
w2 = i+p;
frahh6 = (freq[i+p] + freq[i+p+1])/2D;
System.out.println(" array no1" + " " + w2 + " " + " array
no2" + " " + (w2 +1));
}
}

//*****
double wabh = Math.abs(frahh5 - frahh6);
System.out.println("FWHM" + " " + wabh);
length = max;
System.out.println("length" + " " + length);

double dc = wabh * (Math.PI) ;
double ampl = length * dc;
System.out.println("DecayConst" + " " + dc);
System.out.println("Amplitude" + " " + ampl);
System.out.println("Frequency" + " " + frd);

amplp[0] = ampl;

```

problem in java swings

```
dcp[0] = dc;
//frdp[0] = -frd;
frdp[0] = frd;
//phasp[0] = Math.toRadians(90.0);
phasp[0] = 0.0;

for(int i=0; i<iter ; i++ )
{
Fdesh[i]= 0.0;
FuncE[i] = 0.0;
for(int m=0; m<4; m++)
{
DevFuncE[m] = 0.0;
}

FuncF[i] = 0.0;
for(int m=0; m<4; m++)
{
DevFuncF[m] = 0.0;
}

FuncG[i] = 0.0;
for(int m=0; m<4; m++)
{
DevFuncG[m] = 0.0;
}

FuncH[i] = 0.0;
for(int m=0; m<4; m++)
{
DevFuncH[m] = 0.0;
}

for(int k=0; k< pointarpk; k++)
{
//int j = k + w-p-24;
int j = k ;
//int j = k + w-p-((pointarpk-1)/2);
real1[k] = real[j+1];
realsq[k] = real[j+1]*real[j+1];
imag1[k] = imag[j+1];
imagsq[k] = imag[j+1] * imag[j+1];
double
devRwrtamp=0.0,devRwrtdc=0.0,devRwrtfreq=0.0,devIwrtamp=0.0,devIwrtdc=0.0,devIwrtfreq=0.0;
double term1=0.0, term2=0.0, term3=0.0, term22=0.0;
//term1 = 2.0*(Math.PI)*(frdp[i] + freq[j+p]);
term1 = 2.0*(Math.PI)*(frdp[i] - freq[j+p]);
term2 = term1 * term1;
term3 = ((dcp[i] * dcp[i]) + term2) * ((dcp[i] * dcp[i]) +
term2);
```

problem in java swings

```
term22 = ((dcp[i]* dcp[i]) + term2) * ((dcp[i] * dcp[i]) +
term2) * ((dcp[i] * dcp[i]) + term2);
R[k] = (amplp[i]*dcp[i])/((dcp[i] * dcp[i]) + term2);
devRwrtamp= dcp[i]/((dcp[i] * dcp[i]) + term2);
double term4=0.0,term5=0.0;
term4 = amplp[i]/((dcp[i] * dcp[i]) + term2);
term5 = (2.0*amplp[i]*dcp[i] * dcp[i])/term3;
devRwrtdc = term4 - term5;
devRwrtfreq =
-(4.0*(Math.PI)*amplp[i]*dcp[i]*term1)/term3;
return !value;
}
```

```
public synchronized boolean waitUntilTrue(long msTimeout)
throws InterruptedException {
```

```
return waitUntilStateIs(true, msTimeout);
}
```

```
public synchronized boolean waitUntilFalse(long msTimeout)
throws InterruptedException {
```

```
return waitUntilStateIs(false, msTimeout);
}
```

```
public synchronized boolean waitUntilStateIs(boolean state, long
msTimeout)
throws InterruptedException {
```

```
if (msTimeout == 0L) {
```

```
while (value != state)
wait();
```

```
return true;
}
```

```
long endTime = System.currentTimeMillis() + msTimeout;
```

```
long msRemaining = msTimeout;
```

```
while ((value != state) && (msRemaining > 0L)) {
```

```
wait(msRemaining);
msRemaining = endTime - System.currentTimeMillis();
}
```

```
return (value == state);
}
```

```
}
```

problem in java swings

problem in java swings

```
/*
 * Created on Aug 25, 2004
 *
 */
//package nmruui.gui;

/**
 * @author ANADI MISHRA
 *
 * @version 1.00
 */

import java.io.File;
import java.io.IOException;
import java.io.DataInputStream;
import java.io.FileInputStream;
import javax.swing.JFileChooser;
//import nmruui.readers.*;

public class FileHandler {

    private File file;
    private File currentDir = new File("UNTITLED");
    private JFileChooser dialog;
    private NMRGui cot;

    public FileHandler(NMRGui cot) {
        this.cot = cot;
    }

    public void openFile() {
        dialog = new JFileChooser();
        int open = 0;
        open = dialog.showOpenDialog(cot);

        if (open == JFileChooser.APPROVE_OPTION) {
            File f = dialog.getSelectedFile();
            String s = f.getPath();
            String s1 = f.getPath();
            System.out.println("*****File Path " + s1);

            if (s.endsWith(".txt") || s.endsWith(".TXT")) {
                try {
                    TXTFilesReader txtFile = new TXTFilesReader(s1);
                    double ans[][] = txtFile.loadTxt();
                    String title = f.getName();
                    cot.addScreenGraph(title, 1, ans);
                }
            }
        }
    }
}
```

problem in java swings

problem in java swings

```
} catch (IOException e) {
System.err.println(e);
e.printStackTrace();
}
} else {
boolean flag = false;
boolean flag1 = false;
boolean flag2 = false;
boolean flag3 = false;
FileInputStream fileinputstream;
DataInputStream datainputstream;
byte abyte0[];
String fileName = f.getName();

try {
fileinputstream = new FileInputStream(f);
datainputstream = new DataInputStream(fileinputstream);

String tp;
int type;
abyte0 = new byte[4];
for (int j = 0; j != 4; j++)
abyte0[j] = datainputstream.readByte();
double[][] ans;

GEFilesReader geFile = new GEFilesReader(f.getPath());

type = geFile.findType();
switch (type) {
case 0 :
geFile.loadGe(false, 4, true, true); #####Changed for Phantom
data
//geFile.loadGe(false, 4, true, false);
tp = "4X";
break;
case 1 :
geFile.loadGe(false, 4, true, true);
tp = "5X";
break;
case 2 :
geFile.loadGe(false, 5, true, true);
tp = "P5";
break;
case 3 :
default :
geFile.loadGe(false, 8, true, true);
tp = "LX";
break;

}
ans = geFile.getResult();
```

problem in java swings

problem in java swings

```
cot.addScreenGraph(
"1D Mode " + f.getName() + " " + tp,
2,
ans);
} catch (Exception e) {
e.getMessage();
}
return;
}
} else if (open == JFileChooser.CANCEL_OPTION) {
return;
//do nothing
}

}
public void callSave() {
dialog = new JFileChooser();
int save = dialog.showS
```