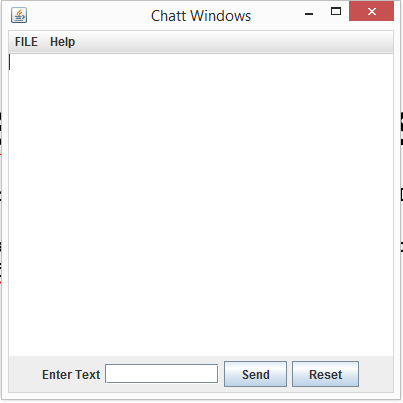
**LabExercise: Graphics**

**Exercise : Graphics, Creating and using GUI Components:**

**Example1:** Creating GUI Interface for Chat Application with MenuBar, Textfiled, TextArea and buttons:



**Program:**

package Example;

import javax.swing.\*;

import java.awt.\*;

class **MenuBarExample** extends JFrame{

public **MenuBarExample**(){

JMenuBar mb = new JMenuBar();

JMenu m1 = new JMenu("FILE");

JMenu m2 = new JMenu("Help");

mb.add(m1);

mb.add(m2);

JMenuItem m11 = new JMenuItem("Open");

JMenuItem m22 = new JMenuItem("Save as");

m1.add(m11);

m1.add(m22);

//Creating the panel at bottom and adding components

JPanel panel = new JPanel(); // the panel is not visible in output

JLabel label = new JLabel("Enter Text");

JTextField tf = new JTextField(10); // accepts upto 10 characters

JButton send = new JButton("Send");

JButton reset = new JButton("Reset");

panel.add(label); // Components Added using Flow Layout

panel.add(tf);

panel.add(send);

panel.add(reset);

// Text Area at the Center

JTextArea ta = new JTextArea();

add(BorderLayout.*SOUTH*, panel);

add(BorderLayout.*NORTH*, mb);

add(BorderLayout.*CENTER*, ta);

}

public static void ***main***(String args[]) {

//Creating the Frame

MenuBarExample frame = new MenuBarExample();

frame.setTitle("Chatt Windows");

frame.setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);

frame.setSize(400, 400);

frame.setVisible(true);

}

}

**Exercise:** Add another menue and menu items into menu bar.

**Example 2:A simple Example to Add some controls to GUI window with event handling.**

import java.awt.FlowLayout;

import java.awt.event.ActionListener;

import java.awt.event.ActionEvent;

import javax.swing.JFrame;

import javax.swing.JTextField;

import javax.swing.JPasswordField;

import javax.swing.JOptionPane;

import javax.swing.JButton;

public class **GUIExercise** extends JFrame {

private JTextField textField1; // text field with set siz

private JTextField textField2; // text field with set siz

private JPasswordField passwordField; // password field with text

private JButton btn1;

private JButton btn2;

public **GUIExercise**(){

super( "My First GUI Application" );

setLayout( new FlowLayout() ); // set frame layout

// construct textfield with 10 columns

textField1 = new JTextField( "Enter some text here",15 );

add( textField1 ); // add textField1 to JFrame

// construct textfield with default text and 21 columns

textField2 = new JTextField( "Uneditable text field2", 21 );

textField2.setEditable( false ); // disable editing

add( textField2 ); // add textField3 to Container

passwordField = new JPasswordField( "Hidden text" );

add( passwordField ); // add passwordField to JFrame

btn1=new JButton("Click me");

add(btn1);

btn2=new JButton("Copy Text");

add(btn2);

// register event handlers

MyEventHandler handler = new MyEventHandler();

textField1.addActionListener( handler );

textField2.addActionListener( handler );

passwordField.addActionListener( handler );

btn1.addActionListener(handler);

btn2.addActionListener(handler);

}

// private inner class for event handling

private class **MyEventHandler** implements ActionListener

{

// process textfield events

String str, str2;

public void **actionPerformed**( ActionEvent event )

{

String string = ""; // declare string to display

// user pressed Enter in JTextField textField1

if ( event.getSource() == textField1 ){

string = String.*format*( "textField1: %s",

event.getActionCommand() );

str= event.getActionCommand();

str2=event.getActionCommand();

}

// user pressed Enter in JTextField passwordField

else if ( event.getSource() == passwordField ){

string = String.*format*( "passwordField: %s",

event.getActionCommand() );

str=event.getActionCommand();

}

else if (event.getSource()==btn1){

JOptionPane.*showMessageDialog*( null, str );

}

else if ( event.getSource() == btn2 )

textField2.setText(str2);

JOptionPane.*showMessageDialog*( null,string );

}

}

public static void ***main***( String[] args )

{

GUIExercise app = new GUIExercise();

app.setDefaultCloseOperation( JFrame.*EXIT\_ON\_CLOSE* );

app.setSize( 450, 450 ); // set frame size

app.setVisible( true ); // display frame

} // end main

}

Add another text box and button to above example, when user enter text in text box and click the button the text should be shown in a dialog box. (see the next example)

**Example#2: Adding Button and performing Event handling when user click the button**

**First stage: Adding Two buttons into GUI and Displaying the Button:**

package graphics;

import javax.swing.JFrame;

import java.awt.FlowLayout;

import java.awt.event.ActionListener;

import java.awt.event.ActionEvent;

import java.awt.event.ItemListener;

import javax.swing.JButton;

import javax.swing.Icon;

import javax.swing.ImageIcon;

import javax.swing.JOptionPane;

/\*\*

\*

\* **@author** Waqas

\*/

public class **ButtonExample** extends JFrame {

private JButton plainButton;

private JButton fancyButton;

public **ButtonExample**(){

super( "Testing Buttons" );

setLayout( new FlowLayout() ); // set frame layout

plainButton=new JButton("Plain Button");

add(plainButton);

Icon bug1 = new ImageIcon( getClass().getResource( "bug1.png" ) );

Icon bug2 = new ImageIcon( getClass().getResource( "bug2.GIF" ) );

fancyButton=new JButton("Fancy Button",bug1);

fancyButton.setRolloverIcon(bug2);

add(fancyButton);

}

public static void ***main***( String[] args )

{

ButtonExample buttonFrame = new ButtonExample(); // create ButtonFrame

buttonFrame.setDefaultCloseOperation( JFrame.*EXIT\_ON\_CLOSE* );

buttonFrame.setSize( 250, 400 ); // set frame size

buttonFrame.setVisible( true ); // display frame

} // end main

}

Second Step: Event handler class and Registering Events to handler

**This is inner class and should be written inside the outer class before main method:**

private class **Hanler** implements ActionListener

{

// handle button event

public void **actionPerformed**( ActionEvent event )

{

JOptionPane.*showMessageDialog*( ButtonExample.this, String.*format*("You Pressed %s", event.getActionCommand()));

} // end method actionPerformed

**Register buttons to handler in the constructor of outer class:**

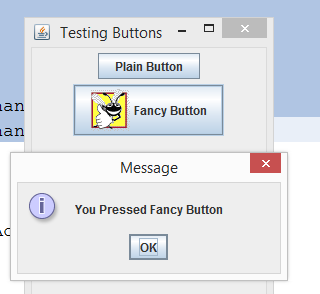
add(fancyButton);

Hanler myhandler=new Hanler();

plainButton.addActionListener(myhandler);

fancyButton.addActionListener(myhandler);

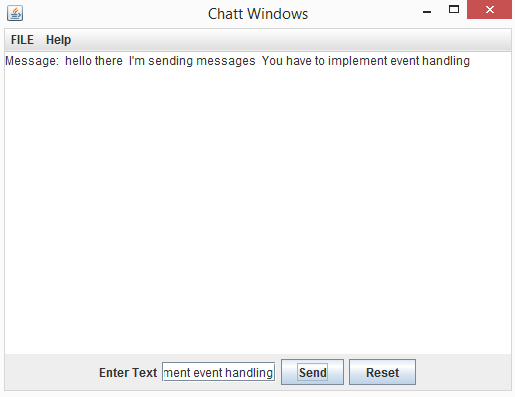
**OUTPUT:**



**Lab Exercise:**

In example 1, if you enter text in text area and press enter or click send button then the text should be displayed in text area.

Clicking Reset button will erase all text from text area.



**Example#3: Example program to demonstrate check box event handling… Add a button and another check box in this example.**

**If button is clicked it should display the message in a dialog window written in text field. Check box should change the font to some other font.**

import java.awt.FlowLayout;

import java.awt.Font;

import java.awt.event.ActionListener;

import java.awt.event.ActionEvent;

import java.awt.event.ItemListener;

import java.awt.event.ItemEvent;

import javax.swing.JFrame;

import javax.swing.JButton;

import javax.swing.Icon;

import javax.swing.ImageIcon;

import javax.swing.JOptionPane;

import javax.swing.JCheckBox;

import javax.swing.JTextField;

public class **ButtonFrame** extends JFrame

{

private JTextField textarea;

private JCheckBox bold;

private JCheckBox italic;

// ButtonFrame adds JButtons to JFrame

public **ButtonFrame**()

{

super( "Testing Buttons" );

setLayout( new FlowLayout() ); // set frame layout

textarea =new JTextField("Simple Event Handling Example",20);

textarea.setFont(new Font("serif",Font.*PLAIN*,14));

add(textarea);

bold =new JCheckBox("Bold");

italic =new JCheckBox("Italic");

add(bold);

add(italic);

// create new ButtonHandler for button event handling

ButtonHandler handler = new ButtonHandler();

italic.addItemListener(handler);

bold.addItemListener(handler);

} // end ButtonFrame constructor

// inner class for button event handling

private class **ButtonHandler** implements ActionListener, ItemListener

{

// handle button event

public void **actionPerformed**( ActionEvent event )

{

} // end method actionPerformed

public void **itemStateChanged**(ItemEvent event){

Font font=null;

if(bold.isSelected() && italic.isSelected())

font=new Font("serif",Font.*BOLD* +Font.*ITALIC*,14);

else if(bold.isSelected())

font=new Font("serif",Font.*BOLD* ,14);

else if(italic.isSelected())

font=new Font("serif",Font.*ITALIC*,14);

else

font=new Font("serif",Font.*PLAIN*,14);

textarea.setFont(font);

}

} // end private inner class ButtonHandler

public static void ***main***( String[] args )

{

ButtonFrame buttonFrame = new ButtonFrame(); // create ButtonFrame

buttonFrame.setDefaultCloseOperation( JFrame.*EXIT\_ON\_CLOSE* );

buttonFrame.setSize( 250, 400 ); // set frame size

buttonFrame.setVisible( true ); // display frame

} // end main

} // end class ButtonFrame