Chapter-9

Creating Label, Dropdown box, Combo box and Menu

Dulal Acharjee

In previous chapter-8, some GUI features like: Frame, Buttons, Checkbox and Radiobuttons were discussed. Now, in this chapter, different other advanced GUI features like Label, Dropdown box, Combo box and Menu creations are discussed. These toolboxes are known as GUI tools for user interfacing which are used with the forms for inputting user selections from different options. These tools have made the life of the users easier than inputting all data by typing as texts. One by one these advanced GUI tools are discussed below:

JLabel

JLabel is inherited from java **swing** package. It is a class under Abstract Windowing Toolkit(AWT) used for setting label of any GUI component. It can set test, an icon as figure and alignment of icon in different directions with the label. The old one is **Label** which can give only text but **JLabel** can add both text and graphics with that text. Inheritance of it is as:
javax.swing.**JLabel**
For window GUI program four components are important:

UI components: this type of UI(User Interface) will contain Buttons, Checkbox, Radiobuttons, lists, menus and other components you may have seen within a Window.
Containers: it is another form of AWT component and holds JPanel which is a type of container. Applets are example of Panel.
Canvases: Within frame, graphic components are drawn on either canvas or on panel.
Window construction components: some special components of window are windows, frames, menubars, dialogs, close, maximize and minimize buttons. An window may contain one or more windows within it.

JLabel class has different overloaded constructors as:

JLabel () : without argument with constructor having no text to display.
JLabel (String s**)** : any string type of texts as argument to be displayed.
JLabel (Icon i**)** : a icon as argument which will be displayed as label.
JLabel (Icon i, int horizontalAlignment): A picture with horizontal alignment will be drawn within the panel/canvas.
JLabel (String s, Icon i, int align**)**: an icon with texts and alignment are defined with label for making the label more user friendly.

An example of using labels with alignment:

```
add(new Label("it will display as centered", Label.CENTER));
```

Different methods of JLabel class:

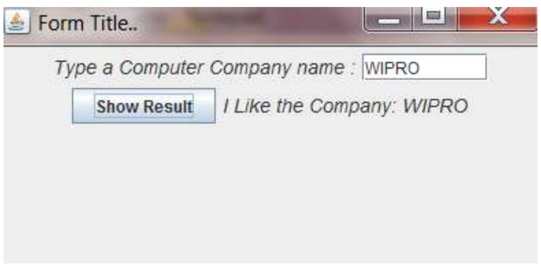
For programming purposes, sometimes, it is required to read the text and to set another text dynamically at the time of program running, some methods are mentioned below:

getText() : will get the text of the label of a window component
setText(String) : current text will be reset by new text of the button.
getAlignment() : it will get the alignment of the label
setAlignment(int): alignment is set as : 0 = LEFT, 1= CENTER, 2 = RIGHT.

```
//FormDemo.java: it is a class file having no main() module. All designing
//of Form components are here; it has 'ActionListener' of a Button having
//label "Show Result"; another program FormCall.java will use this program.
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
public class FormDemo extends JFrame implements ActionListener
{
  final int x = 275;
  final int y= 225;
  JLabel name = new JLabel("Type a Computer Company name :");
  Font fnt = new Font("Roman", Font.ITALIC, 14);
  JTextField an = new JTextField(8);
  JButton sh = new JButton("Show Result");
  JLabel txt = new JLabel("");
  public FormDemo()
  {
        super("Form Title..");
        setSize(x, y);
        setLayout(new FlowLayout());
        name.setFont(fnt);
        txt.setFont(fnt);
        add(name);
        add(an);
        add(sh);
        add(txt);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        sh.addActionListener(this);
  }

  public void actionPerformed(ActionEvent e)
  {
        String name = an.getText();
        String greet = "I Like the Company: " + name;
        txt.setText(greet);
  }
}

//FormCall.java: it is the main java program which invoke a class, FormDemo.class,
//which is a designed form.
public class FormCall
  {
        public static void main(String[] args)
     {
          FormDemo frame = new FormDemo();
          frame.setVisible(true);
     }
  }
```



I Like the Company: WIPRO

Discussion: this program demonstrates event based programming where ‘Show Result’ push button registers the event by the statement ‘sh.addActionListener(this);’ and then the action is performed by the method ‘actionPerformed(ActionEvent e)’. This example also shows how to use a form class by another program. To run this program, first of all ‘FormDemo.java’ should be compiled by the command:

```
E:\example>javac FormDemo.java
```

Then ‘FormDemo.class’ is compiled and bytecode is generated. This bytecode is nothing but our form which contains other libraries required to run this program. So, if you look at the size of class file it is 4 to 5 times larger than that of .java file. Reusability is one of the main feature of JAVA, classes written by us can be called and used by another java program. This concept is to design a form and then use that form in different programs. It is known as reusability of program module. So, to call ‘FormDemo.class’, we have written another small program by name ‘FormCall.java’ as:

```
E:\example>javac FormCall.java
E:\example> java FormCall
```

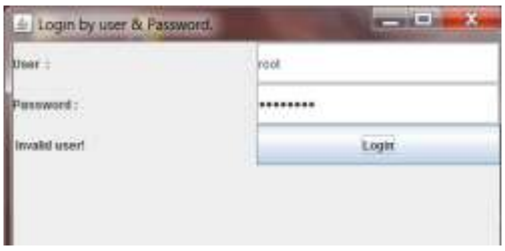
Then, we got the form as shown above to input data and to do some job under the Button ‘Show Result’.

```
//UserLogin.java: this program demonstrates how to create a user login page of a
project.
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
public class UserLogin extends JFrame implements ActionListener
{
   JLabel ul, pl, ms;
   JTextField txt;
   JPasswordField ptxt; // to create invisible password column
   JButton submit, cancel;
   UserLogin()
  {
     JPanel pn = new JPanel(new GridLayout(5, 2));
     ul = new JLabel();
     ul.setText("User  :");
     pn.add(ul);
     txt = new JTextField();
     pl = new JLabel();
     pl.setText("Password :");
     pn.add(txt);
     ptxt = new JPasswordField();
     submit = new JButton("Login");
     pn.add(pl);
     pn.add(ptxt);
     ms = new JLabel();
     pn.add(ms);
     pn.add(submit);
     setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
     submit.addActionListener(this); //event listener
     add(pn, BorderLayout.CENTER);
     setTitle("Login by user & Password.");
     setSize(300,250);
     setVisible(true);
  }
  public void actionPerformed(ActionEvent ee)
  {
     String user = txt.getText();
     String code = ptxt.getText();
```

```
        if (user.trim().equals("root") && code.trim().equals("root"))
        {
            ms.setText(" Valid User: " + user + "");
         }
         else
         {
            ms.setText(" Invalid user! ");
         }
    }

// main module
  public static void main(String[] st)
{
     new UserLogin();
   }
}
```

OUTPUT:



```
Login by user & Password.
User :          root
Password :      ••••••••
Invalid user!   Login
```

Discussion: this program has three components like: user, password and login and for each of component to work three steps of statements are written as:

```
         ul = new JLabel();
        ul.setText("User  :");
        pn.add(ul);
```

Then, to create password field as unseen code, we have used JpasswordField class at the beginning of the program and here we have initialized predefined text variable ‘ptxt’ by this statement as:

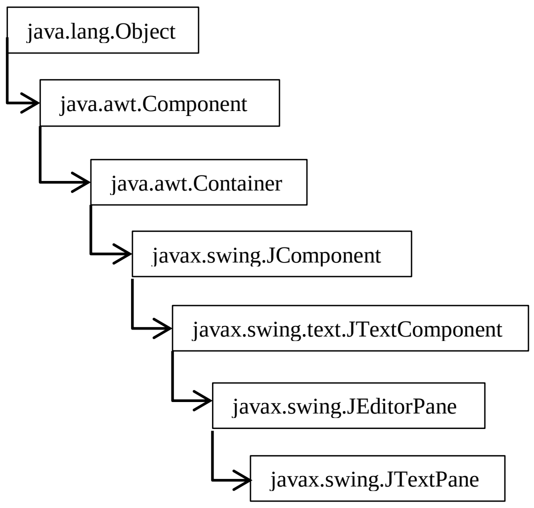
```
            ptxt = new JPasswordField();
```

Here, JpasswordField() is the constructor of the same class. There are two important methods of the class ‘JButton’ are used and these are:

```
            addActionListener(this);
            actionPerformed(ActionEvent ee)
```

The first method hears or registers the action performed and the second method executes what to be performed by that action. In this example, when push button is clicked by mouse it will check for both user and code are ‘root’ then it will do some work, if username and passcode are not same, then it will do another work. Users use this logic every day when they login any software either standalone or web based. For checking user name and password, there is a datafile which contains these info, for writing customised application, you have to open that password data file, search for user name, if found within the datafile, then read two fields i.e, username and password, if these are same then software will open else will give error message.

Hierarchy of JTextPane class:



Creating Dropdown box and JComboBox

It is another feature to select an option from listed options which is displayed within a dropdown box. If the top of the box is clicked, then, options are displayed, else, it remains to display as a single option. Hierarchy of JComboBox is as:

javax.swing.JComboBox

There are different ways to create a list of ComboBox. Using String class as array of strings, the technique of creating ComboBox is shown below:

String [ ] list = {"Bread", "Fish", "Meat", “Vegitable”};

When we create a list of data within the memory, we should attach this data to the first memory as:

JComboBox abc = new JComboBox(list);

To work with different options of ComboBox, there are different methods such as:

addItem(), removeItem(), getItemAt() etc.

Items or components of a ComboBox start from index number 0 then 1, 2 ..etc. Using the method getSelectedIndex() any index number can be reached and them program can do next steps that selecting the position 0 for ‘Bread’, 1 for ‘Fish’ etc. what to be done.

//ExCombo.java: demonstrating ComboBox and selection

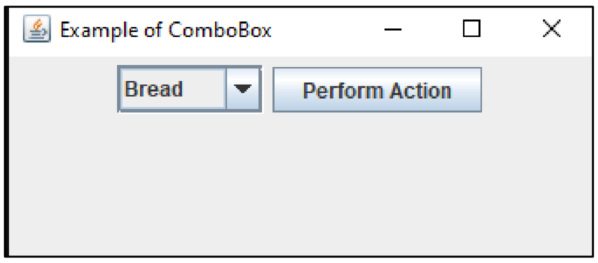
```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class ExCombo {
   JFrame fm;
   JLabel Sb;
   JPanel Sp;
    public ExCombo(){
     frameShow();
                              }
  public static void main(String[] args){
     ExCombo  obC = new ExCombo();
     obC.comboShow();
  }
  private void frameShow(){
     fm = new JFrame("Example of ComboBox");
     fm.setSize(350,320);
     fm.setLayout(new GridLayout(4, 1));
     fm.addWindowListener(new WindowAdapter() {
        public void windowClosing(WindowEvent obWin){
           System.exit(0);
        }
     });
     Sb = new JLabel("",JLabel.LEFT);
     Sb.setSize(250,120);
     Sp = new JPanel();
     Sp.setLayout(new FlowLayout());
     fm.add(Sp);
     fm.add(Sb);
     fm.setVisible(true);
  } // end of method
  //another user defined method
  private void comboShow(){
     final DefaultComboBoxModel food = new DefaultComboBoxModel();
     food.addElement("Bread");
     food.addElement("Fish");
     food.addElement("Meat");
     food.addElement("Vigitable");
     final JComboBox foodC = new JComboBox(food);
     foodC.setSelectedIndex(0);
     JScrollPane sPanel = new JScrollPane(foodC);
     JButton sB = new JButton("Perform Action");
      sB.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {
           String data = "";
           if (foodC.getSelectedIndex() != -1) {
                data = "Option selected is: "
                + foodC.getItemAt
                 (foodC.getSelectedIndex());
           }
```

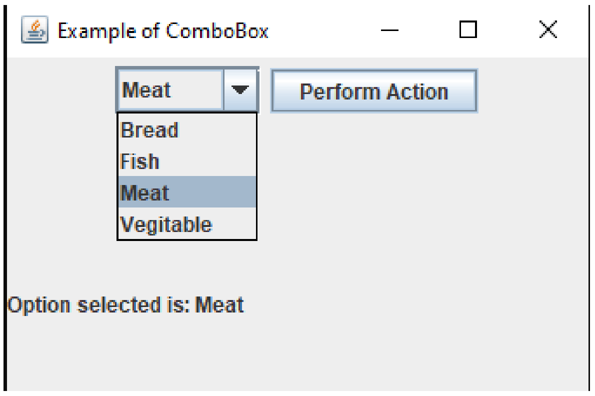
```
          Sb.setText(data);
        }
    });
     Sp.add(sPanel);
     Sp.add(sB);
     fm.setVisible(true);
  }
}//end of class
```

OUTPUT:

Discussion: when program is run it displays screen as shown below. It has two boxes, first one is the combo box which contains list of options and the second one 'perform action' is the pushbutton after clicking it another action will be performed.



When the down-icon is pressed, it will list all options within combo box. Programs are done to scroll the options with shade marks of selection. If any option is clicked, name of that option is displayed at top of list as shown below.



Option selected is: Meat

This statement collects the name of the option as example 'Meat'
foodC.getSelectedIndex()

This program demonstrate for JScrollPane, JComboBox and JButton classes.

If we delete this line from the program as shown below, it will not show the comboBox panel within the frame.

Panel is added and made visible by this statement:
    Sp.add(sPanel);
If the upper line is deleted from the end of the above program, then combobox panel will not be displayed within the frame.

If the below line is deleted from the end of the above program, then 'Perform Action' button will not be displayed.
Sp.add(sB);

If the below line is deleted at end of the program, then, both combobox and pushbutton will be disappeared.

```
        fm.setVisible(true);
```

Creating Menu:

Menu is a UGI tool to select different options which are listed under different group of sub options. Most of the computer users have used MSword, MSexcel etc. and have seen menus like options: File, Home, Insert, Page Layout, References etc. If you click any option, lists are displayed within box. It has multiple ways of display modes and hope users have worked with some of options.

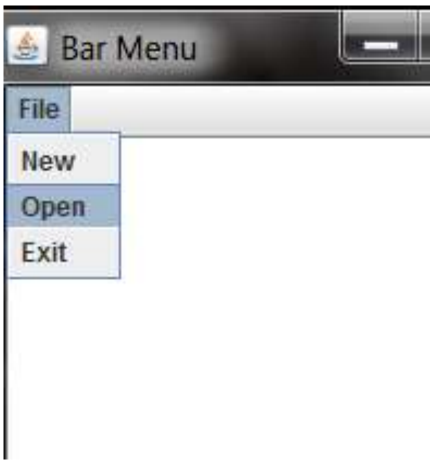
Here, a simple menu 'File' is created having three options as 'New', 'Open' and 'Exit' having names 'bar1', 'bar2' and 'bar3'. For selection of any option, action listener is registered as: bar1.addActionListener(this);

The full program is shown below:

```
//mymenu.java: demonstration of creating a small menu list
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.text.*;
public class mymenu extends JApplet implements ActionListener
{
   Container cont;
   JTextPane txtpane;
   public mymenu()
   {
      cont = getContentPane();
      txtpane = new JTextPane();
      txtpane.setEditable(true);
      JScrollPane ptext = new JScrollPane(txtpane);
      cont.add(ptext);
      JMenuBar bar = new JMenuBar();
      setJMenuBar(bar);
      JMenu title = new JMenu("File");
      bar.add(title);
      JMenuItem bar1, bar2, bar3;
      bar1 = new JMenuItem("New");
      title.add(bar1);
      bar2 = new JMenuItem("Open");
      title.add(bar2);
      bar3 = new JMenuItem("Exit");
```

```
        title.add(bar3);
        //register for action requested for each options
        bar1.addActionListener(this);
        bar2.addActionListener(this);
        bar3.addActionListener(this);
     }
    public void actionPerformed(ActionEvent e)
    {
        String action = e.getActionCommand();
        JFrame jf=new JFrame();
        try
        {
           if (action.equals("New")) txtpane.setText("");
            else if (action.equals("Open")) openFile();
           else if (action.equals("Exit")) System.exit(0);
        }
            catch (BadLocationException e1)
            {}
     }
     void openFile() throws BadLocationException
     {
        JFileChooser d = new JFileChooser("Open a file");
        if (d.showOpenDialog(cont) = =JFileChooser.APPROVE_OPTION)
        {
           StyledDocument s = txtpane.getStyledDocument();
            s.insertString(s.getLength(),"Open file \""
+d.getSelectedFile().getAbsolutePath() +"\"\n", null);
        }
     }
     public static void main(String[] args)
     {
            mymenu bMenu = new mymenu();
            JFrame fm = new JFrame("Bar Menu ");
            fm.addWindowListener(new WindowAdapter()
          {
               public void windowClosing(WindowEvent e)
              {
                   System.exit(0);
               }
          });
        fm.getContentPane().add(bMenu,BorderLayout.CENTER);
        fm.setVisible(true);
     }
}
```

OUTPUT:



Discussions: an important class, JTextPane, is used in this example for marking the text components “File” with some attributes. When that text is clicked it performs some job. In this case, when ‘File’ is clicked, it displays dropdown menu having three options like ‘New’, ‘Open’ and ‘Exit’. Two methods are used, setEditable(true) and getStyledDocument(). Due to the first method it allows to input text within the text area like any text editor. It has two arguments ‘true and false’. If the argument parameter is ‘false’ then user can’t input text within the text area. The getStyledDocument(), gets documents or texts from the panel area and then patterns or colors can be applied on those texts.
Another important class name is used, i.e, Container class which contains a gathered information. In this example, an object ‘cont’ is created from the class ‘Container’ and used as an argument to the method ‘showOenDialog()’ In this way:

d.showOpenDialog(cont) = = JFileChooser.APPROVE_OPTION)

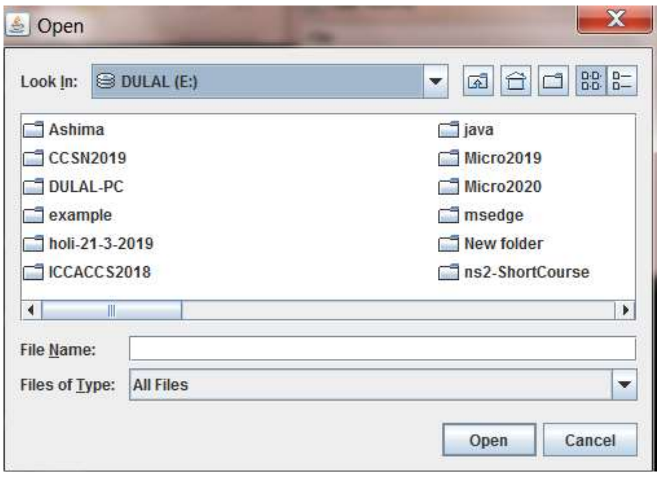
(Note: this statement is used with ‘if’ condition, so, two equal sign is used. The syntax of JAVA is like C/C++ as:

```
 if (x = = y)
    {
       ……
     }
```

But, for assignment of any value to a parameter, only one equal sign is used as:

```
   x = 10;
   x = y +5; etc.
```

By this method it opens a window of ‘file open’ generally happens for all software as:



Selection based Menu:

In this example, the class **Arrays** is used. In the website of JAVA, about 105 method names are mentioned under the class **Arrays.** Some method names are mentioned below:

Return type	Method name	Descriptions
static int	binarySearch(byte[] a, byte key)	Searches the **key** value within the list byte[] type of list-a using binary search algorithm.
static int	binarySearch(int[] a, int key)	Searches the key number from the list of integers
static boolean	deepEquals(Object[] a1, Object[] a2)	If both arrays are equal, returns true else false.
static boolean	equals(char[] a, char[] a2)	If both character arrays are equal, returns true else false
static void	fill(char[] a, char val)	Fills all positions of character array by val
static void	sort(char[] a)	Sorts the array by ascending numerical order
static string	toString(byte[] a)	Converts all components of array to string

Questions and Answers:

Descriptive questions:

1. What are the advantages of GUI tools?
2. Have you used any GUI tools in window or MSword? Name some tools used for different works.
3. How many ways can you input data? Think and answer.
4. Mouse, Microphone, digital pen, scanner are of what types of devices?
5. How do wireless mouse work?
6. JLabel is a class under which package?
7. Explain the statement :

   ```
   final int x = 275;
   ```

8. What is the meaning of the keyword 'final'?
9. Explain the statement:

   ```
   JLabel name = new JLabel("Type a Computer Company name :");
   ```

10. What is the meaning of the keyword 'new'?
11. Explain the statement:

    ```
    name.setFont(fnt);
    ```

12. What is the 'fnt' in above statement?
13. What is the meaning of the statement?

    ```
    sh.addActionListener(this);
    ```

14. What is the meaning of the keyword 'this'?
15. In the above example of 'FormDemo.java' what is the role of 'FormDemo()'?
16. Is 'FormDemo()' a user defined method or a constructor?
17. In the program mentioned above, 'mymenu.java', what is the meaning of the statement?

    ```
    cont = getContentPane();
    ```

18. In above statement what is 'cont' ?
19. Why is not 'new' operator used after the '=' sign in the below statemet?

    ```
    String action = e.getActionCommand();
    ```

20. Explain the statement given below used in 'mymenu.java' for file-open purpose:

    ```
    if (action.equals("Open")) openFile();
    ```

21. What is the meaning of 'throws' in this statement used in 'mymenu.java'?

    ```
    void openFile() throws BadLocationException
    ```

Link of PPT of this book and all source codes:

`https://drive.google.com/drive/folders/1bMxMCaqPe0W35COAdn_-BrnV_uzQ-tNO?usp=sharing`

(Note: to get access, copy and paste this link to your browser. Both PPT and PDF version of presentation of this book are uploaded in Google Drive.)