CSE 1203

Object Oriented Programming [C++]

Chapter 5:
Java Programming -03

Java: Swing

Java: swing

Java Swing is a part of Java Foundation Classes (JFC) that is *used to create window-based applications*. It is built on the top of AWT (Abstract Windowing Toolkit) API and entirely written in java.
The `javax.swing` package provides classes for java swing API such as `JButton`, `JTextField`, `JTextArea`, `JRadioButton`, JCheckbox, `JMenu`, `JColorChooser` etc.

Java: awt vs swing

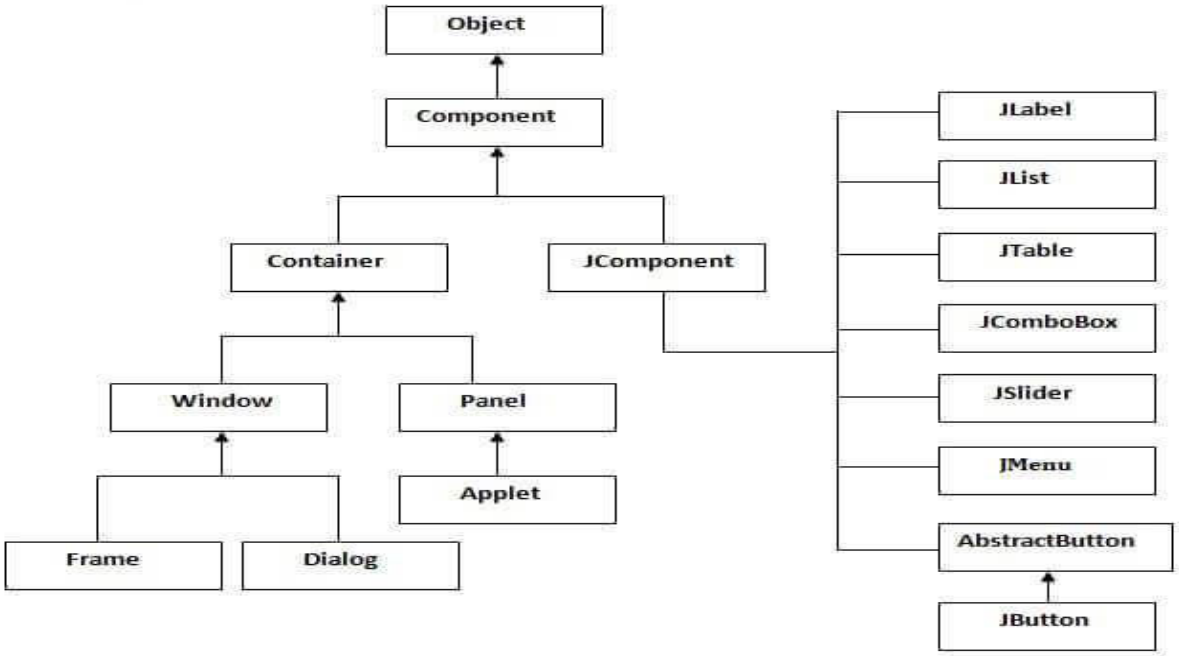
AWT (Abstract Window Toolkit)

- >It was platform-dependent
- >It was heavy-weight
- >It has a limited set of components

Swing

- >It is platform-independent
- >It is light-weight
- >It has more & powerful components

Java: Hierarchy of Swing classes



Java: swing topics

- >Basic Components
- >Applications
- >Event Listeners
- >Layouts Managers
- >Some Advance Components
- >Layers Of JFrame
- >Look and Feel
- >Projects

Java: inheritance

Java swing: JFrame class

The class **JFrame** is an extended version of **java.awt.Frame** that adds support for the JFC/Swing component architecture. It is like a board where you can add your swing objects

Java swing Frame: ways to create frame

Methods:

1. By creating the object of Frame class (association)
2. By extending Frame class (inheritance)
3. Create a frame using Swing inside main()

Java: swing methods

setVisible()
setDefaultCloseOperation()
setSize()
setLocation()
setBounds()
setIconImage()
setTitle()
setBackground()
setResizable()

Java swing Frame: way 1 to create frame

```
package CSE1203;
import javax.swing.JFrame;
import javax.swing.*;

public class First {
    JFrame frame;
    First()
    {
        frame=new JFrame("first way");
        // setting close operation
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        // sets 500 width and 600 height
        frame.setSize(500, 600);

        // makes the frame visible
        frame.setVisible(true);
    }
    public static void main(String[] args) {
        new First(); //anonymous object
        //First f=new First() is similar
    }
}
```

Notes A frame frame object is created in the constructor of First class. For JFrame class import `javax.swing.JFrame`;

Java: inheritance

Java swing Frame: way 2 to create frame

```
package CSE1203;
import javax.swing.*;

public class First extends JFrame {

First()
    {
        setTitle("this is also a title");
// setting close operation
  setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        setSize(400, 500);
        setVisible(true);
    }
    public static void main(String[] args)
    {
        First f=new First() is similar
    }
}
```

Notes here First class is inherited by Jframe class. So object of Jframe class can use the methods of Jframe class. The methods includes `setTitle(), setSize(), setVisible` etc.

Java swing Frame: way 3 to create frame

```
package CSE1203;
import javax.swing.JFrame;
public class First {
 public static void main(String[] args) {
   JFrame frame=new JFrame("CSE 1203");
   frame.setSize(600,400);
   frame.setLocation(200,100);
   //frame.setBounds(200,100,200,300);
   frame.setVisible(true);
   frame.setDefaultCloseOperation(frame.EXIT_ON_C
LOSE);
 }
}
```

Notes A frame of size 600x400 will be created at location display point (200,100). The frame is visible and it would be closed when x button is clicked.

Java swing : ImageIcon class

```
ImageIcon icon=new
ImageIcon("F:\\Zaman\\Image\\login.png");
frame.setIconImage(icon.getImage());
```

Notes Adding a new icon at the top upper corner of the frame.

Java: inheritance

Java swing : Container & Color class

```
Container c=frame.getContentPane();
c.setBackground(Color.RED);
//create color object
//Color color=new Color(255,0,0)
//c.setBackground(color);
```

Notes To change the background of the frame, first create a **Container** class object then set its background color using **Color** class

Java swing : JFrame setResizable()

```
frame.setResizable(false);
```

Notes If the argument is `false`, the frame cann't be resized.

Java swing : Jlabel class

```
JFrame frame=new JFrame("JLabel");
        frame.setBounds(200, 100,400,300);
        frame.setVisible(true);

frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        Container c=frame.getContentPane();
        c.setLayout(null);
        JLabel label=new JLabel("Username:");
        //label.setText("User Name:");
        label.setBounds(50,70,100,30);
        c.add(label);
```

Java swing : Font class

```
Font font=new Font("Arial",Font.PLAIN,12);
label.setFont(font);
```

Notes Font class constructor requires three parameters like Font name, Font type and Font size

Java swing : ImageIcon with JLabel class

```
ImageIcon icon=new
ImageIcon("F:\\Zaman\\Image\\Apple.png");
        JLabel label=new
JLabel("Apple",icon,JLabel.LEFT);
        label.setBounds(50,70,500,200);
        c.add(label);
        frame.setVisible(true);
```

Notes Jlabel constructor take 3 arguments text, icon and position

Java swing : JTextField

```
JTextField text=new JTextField();
text.setText("type here");
text.setBounds(50,30,100,20);
text.setForeground(Color.blue);
text.setBackground(Color.cyan);
text.setEditable(true);
c.add(text);
```

Notes: `JTextField()` is a input box

Java: inheritance

Java swing : JPasswordText class

```
JPasswordField text=new JPasswordField();
        text.setBounds(50,30,100,20);
        text.setForeground(Color.blue);
        text.setBackground(Color.cyan);
        text.setEchoChar('*');
        text.setEchoChar((char)0); //show
        text.setEditable(true);
        c.add(text);
```

Java swing : JButton & Cursor class

```
JButton btn=new JButton("Submit");
btn.setBounds(100,50,100,30);
Cursor cursor=new Cursor(Cursor.HAND_CURSOR);
btn.setCursor(cursor);
btn.setEnabled(false);
c.add(btn);
```

Notes when cursor moves inside button Hand cursor is displayed. To disable the button use `setEnabled()` to `false`

Java swing : ActionListener

The Java ActionListener is notified whenever you click on the button or menu item. It is notified against ActionEvent. The ActionListener interface is found in java.awt.event package. It has only one method: actionPerformed().

Java swing : actionPerformed() method

The `actionPerformed()` method is invoked automatically whenever you click on the registered component.

public abstract void actionPerformed(ActionEvent e);

Java swing :How to implement ActionListener

ActionListener class, you need to follow 3 steps:

1) Implement the ActionListener interface in the class:

```
public class ActionListenerExample Implements
     ActionListener
```

2) Register the component with the Listener:

component.addActionListener(instanceOfListenerclass);

3) Override the actionPerformed() method:

```
public void actionPerformed(ActionEvent e){
      //Write the code here
}
```

Java: inheritance

Java swing : use of ActionListener interface

```
package CSE1203;

import java.awt.Color;
import java.awt.Container;
import java.awt.Cursor;
import java.awt.Font;
import java.awt.Image;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.util.List;

import javax.swing.*;
class MyFrame extends JFrame implements
ActionListener{
Container c;
JButton btn1,btn2;
JLabel label;
public MyFrame() {
c=this.getContentPane();
        c.setLayout(null);
        btn1=new JButton("Submit");
        btn1.setBounds(100,50,100,30);
        btn2=new JButton("Cancel");
        btn2.setBounds(210,50,100,30);
        label=new JLabel();
        label.setBounds(110,110,400,30);
        label.setText("Output Displays here");
        btn1.addActionListener(this);
        btn2.addActionListener(this);
        c.add(btn1);
```

```
  public void actionPerformed(ActionEvent e) {
     if(e.getSource()==btn1)
     label.setText("Submit Button Pressed");
     if(e.getSource()==btn2)
     label.setText("Cancel Button Pressed");
  }
}
public class First {
    public static void main(String[] args)
    {
        MyFrame frame=new MyFrame();
        frame.setBounds(200, 100,400,500);
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setVisible(true);

    }
}
```

Notes: When a button is clicked, through addActionListener() method it invokes actionPerformed() automatically. Using object e inside the actionPerformed() method check which button is clicked and setText() in label accordingly.

Java: inheritance

Java swing : use of ActionListener interface

```
package CSE1203;

import java.awt.Color;
import java.awt.Container;
import java.awt.Cursor;
import java.awt.Font;
import java.awt.Image;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.util.List;

import javax.swing.*;
class MyFrame extends JFrame{
Container c;
JButton btn1,btn2;
JLabel label;
public MyFrame() {
c=this.getContentPane();
        c.setLayout(null);
        btn1=new JButton("Submit");
        btn1.setBounds(100,50,100,30);
        btn2=new JButton("Cancel");
        btn2.setBounds(210,50,100,30);
        label=new JLabel();
        label.setBounds(110,110,400,30);
        label.setText("Output Displays here");
 btn1.addActionListener(new ActionListener() {
   public void actionPerformed(ActionEvent e) {
     label.setText("Submit is pressed");
   }
 });
 btn2.addActionListener(new ActionListener() {
   public void actionPerformed(ActionEvent e) {
     label.setText("Cancel is pressed");
   }
 });
        c.add(btn1);
        c.add(btn2);
        c.add(label);
}
}
public class First {
    public static void main(String[] args)  {
        MyFrame frame=new MyFrame();
        frame.setBounds(200, 100,400,500);
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setVisible(true);
    }
}
```

Notes: When AddActionListener is called, its parameter is the object of interface and in the same time abstract method actionPermored() is overrriden. Here no need to create a class to override actionPerformed().

Java: Swing Example

Java swing : login form

```
package CSE1203;

import java.awt.Color;
import java.awt.Container;
import java.awt.Cursor;
import java.awt.Font;
import java.awt.Image;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.util.List;

import javax.swing.*;
class MyFrame extends JFrame
implements ActionListener{
Container c;
JButton btn1,btn2;
JLabel label1,label2,label9;
JTextField text1,text2;
JPasswordField pass1;

String[] username= {"zaman","kamal"};
String[] password= {"123","user"};

public MyFrame() {
    c=this.getContentPane();
        c.setLayout(null);

        label1=new JLabel("Username");

label1.setBounds(50,50,100,30);
        text1=new JTextField();

text1.setBounds(120,50,180,30);
```

```
 label2=new JLabel("Password");

label2.setBounds(50,80,100,30);
        pass1=new JPasswordField();

pass1.setBounds(120,80,180,30);

          btn1=new JButton("Cancel");

btn1.setBounds(100,150,100,30);
          btn2=new JButton("Login");

btn2.setBounds(210,150,100,30);

          label9=new JLabel();

label9.setBounds(120,200,180,30);

        btn1.addActionListener(this);
        btn2.addActionListener(this);

        c.add(label1);
        c.add(text1);
        c.add(label2);
        c.add(pass1);
        c.add(btn1);
        c.add(btn2);
        c.add(label9);
}
```

```
public void actionPerformed(ActionEvent
e) {
  int flag=0;
  String s=new
String(pass1.getPassword());
  if(e.getSource()==btn1)
  dispose(); //close the frame
  if(e.getSource()==btn2) {
  for(int i=0;i<username.length;i++) {
if(username[i].equals(text1.getText()))
if(password[i].equals(s))
 flag++;
  }
  if(flag==1)
    label9.setText("Valid User");
  else
label9.setText("Worng
username/password");
  }
 }
}
public class First {
    public static void main(String[]
args) {
        MyFrame frame=new MyFrame();
        frame.setBounds(200,
100,400,500);
frame.setDefaultCloseOperation(JFrame.E
XIT_ON_CLOSE);
        frame.setVisible(true);
    }
}
```

Java: Swing

Java swing : JTextArea class

```
c=this.getContentPane();
c.setLayout(null);
JTextArea area=new JTextArea();
area.setBounds(100,50,200,150);
area.setBackground(Color.BLUE);
area.setForeground(Color.white);
area.setLineWrap(true);
c.add(area);
```

Notes This method is used to input a text in a text area.

Java swing : JRadioButton & ButtonGroup class

```
package CSE1203;

import java.awt.Color;
import java.awt.Container;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.*;

class MyFrame extends JFrame implements
ActionListener{
Container c;
JRadioButton rb1,rb2;
public MyFrame() {
        c=this.getContentPane();
        c.setLayout(null);
        rb1=new JRadioButton("Male");
        rb1.setBounds(100,50,200,30);
        rb1.setSelected(true)
        rb2=new JRadioButton("Female");
        rb2.setBounds(100,80,200,30);
        ButtonGroup bg=new ButtonGroup();
        bg.add(rb1);  bg.add(rb2);
        rb1.addActionListener(this);
        rb2.addActionListener(this);
        c.add(rb1);  c.add(rb2);
}
public void actionPerformed(ActionEvent e) {
if(rb1.isSelected()){
JOptionPane.showMessageDialog(this,"You are
Male.");
}
if(rb2.isSelected()){
JOptionPane.showMessageDialog(this,"You are
Female.");
}
}
}
public class First {
    public static void main(String[] args) {
        MyFrame frame=new MyFrame();
        frame.setBounds(200, 100,400,500);
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setVisible(true);
    }
}
```

Notes ButtonGroup class combines the buttons in one group ie only one can be selected from the group.

Java: Swing

Java swing : JcheckBox class

```
package CSE1203;
import java.awt.Container;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.*;
class MyFrame extends JFrame implements
ActionListener{
 Container c;
 JCheckBox check;
 JButton btn1,btn2;
 public MyFrame() {
        c=this.getContentPane();
        c.setLayout(null);
        check=new JCheckBox("I agree");
        check.setBounds(100,80,200,30);
        btn1=new JButton("Cancel");
        btn1.setBounds(100,130,80,30);
        btn2=new JButton("Next");
        btn2.setBounds(180,130,80,30);
        btn2.setEnabled(false);
        check.addActionListener(this);
        c.add(check); c.add(btn1); c.add(btn2);
}
public void actionPerformed(ActionEvent e) {
   if(check.isSelected())
      btn2.setEnabled(true);
   else
      btn2.setEnabled(false);
 }
}
```

```
public class First {
    public static void main(String[] args)
    {
        MyFrame frame=new MyFrame();
        frame.setBounds(200, 100,400,500);
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE
);
        frame.setVisible(true);

    }
}
```

Notes Initially Next Button is disabled. When check box is selected then Next button would be enabled.

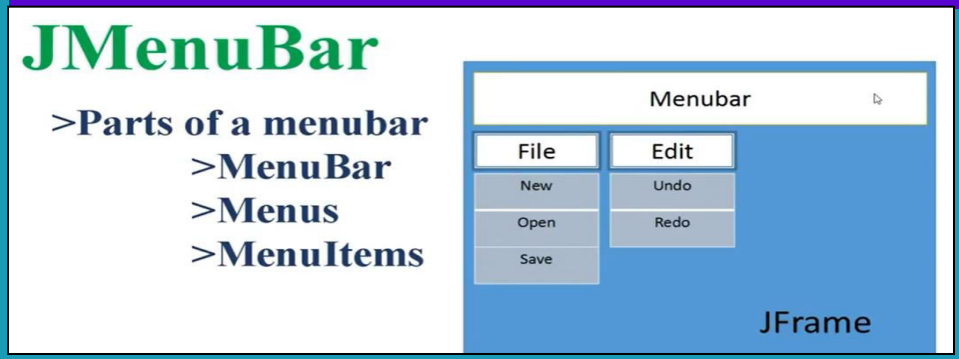
Java: Swing

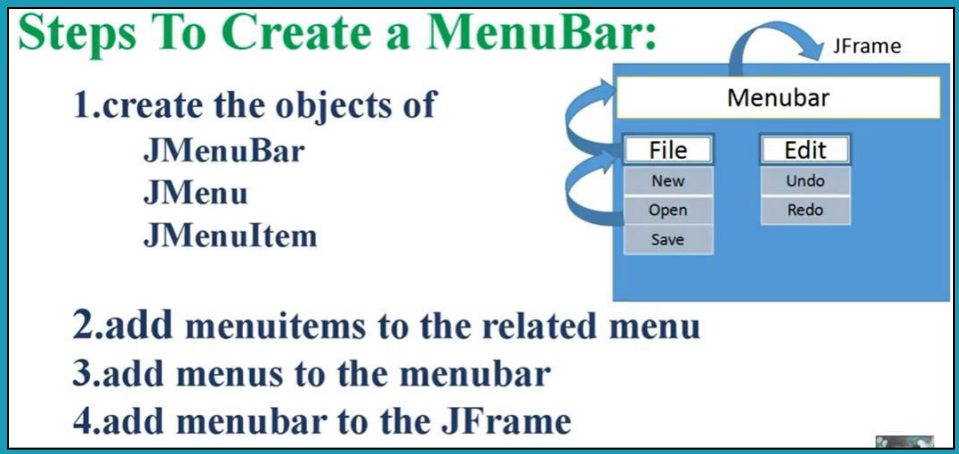
Java swing : JComboBox class

```
package CSE1203;
import java.awt.Container;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.*;
class MyFrame extends JFrame implements
ActionListener{
Container c;
JComboBox combo;
String[] ax=
{"Australia","Banglades","India","Japan","Malays
ia"};
String[] bx= {"Canberra","Dhaka","New
Delhi","Tokyo","Kualalumpur"};
public MyFrame() {
    c=this.getContentPane();
        c.setLayout(null);
        combo=new JComboBox(ax);
        combo.setSelectedIndex(1);
        combo.setBounds(100,80,200,30);
combo.addActionListener(this);
        c.add(combo);
}
public void actionPerformed(ActionEvent e) {
//String s=(String) combo.getSelectedItem();
int i=combo.getSelectedIndex();
JOptionPane.showMessageDialog(this,"Capital:"+bx
[i]);
}
}
```

Notes There are two arrays ax and bx contains contry and capital name respectively. When user select a country from combobox then in the actionPerformed() method selected country index is stored in i. Finally, capital bx[i] is displayed in dialog box.
The main() method is NOT written here, it is similar to the previous one.

Java swing : JMenuBar class





THANK YOU