Java JPopupMenu

PopupMenu can be dynamically popped up at specific position within a component. It inherits the JComponent class.

JPopupMenu class declaration

Let's see the declaration for javax.swing.JPopupMenu class.

```
public class JPopupMenu extends JComponent implements Accessible, MenuElement
```

Commonly used Constructors:

Constructor	Description
JPopupMenu()	Constructs a JPopupMenu without an "invoker".
JPopupMenu(String label)	Constructs a JPopupMenu with the specified title.

Java JPopupMenu Example

```
import javax.swing.*;
import java.awt.event.*;
class PopupMenuExample
{
    PopupMenuExample()
    {
        final JFrame f= new JFrame("PopupMenu Example");
        final JPopupMenu popupmenu = new JPopupMenu("Edit");
        JMenuItem cut = new JMenuItem("Cut");
        JMenuItem copy = new JMenuItem("Copy");
        JMenuItem paste = new JMenuItem("Paste");
        popupmenu.add(cut); popupmenu.add(copy); popupmenu.add(paste);
        f.addMouseListener(new MouseAdapter() {
           public void mouseClicked(MouseEvent e) {
               popupmenu.show(f , e.getX(), e.getY());
           }
        });
        f.add(popupmenu);
        f.setSize(300,300);
        f.setLayout(null);
        f.setVisible(true);
    }
public static void main(String args[])
{
        new PopupMenuExample();
}}
```

Java JPopupMenu Example with MouseListener and ActionListener

```java
import javax.swing.*;
import java.awt.event.*;
class PopupMenuExample
{
     PopupMenuExample(){
         final JFrame f= new JFrame("PopupMenu Example");
         final JLabel label = new JLabel();
         label.setHorizontalAlignment(JLabel.CENTER);
         label.setSize(400,100);
         final JPopupMenu popupmenu = new JPopupMenu("Edit");
         JMenuItem cut = new JMenuItem("Cut");
         JMenuItem copy = new JMenuItem("Copy");
         JMenuItem paste = new JMenuItem("Paste");
         popupmenu.add(cut); popupmenu.add(copy); popupmenu.add(paste);
         f.addMouseListener(new MouseAdapter() {
            public void mouseClicked(MouseEvent e) {
                popupmenu.show(f , e.getX(), e.getY());
            }
         });
        cut.addActionListener(new ActionListener(){
         public void actionPerformed(ActionEvent e) {
             label.setText("cut MenuItem clicked.");
         }
        });
        copy.addActionListener(new ActionListener(){
            public void actionPerformed(ActionEvent e) {
                label.setText("copy MenuItem clicked.");
            }
           });
        paste.addActionListener(new ActionListener(){
            public void actionPerformed(ActionEvent e) {
                label.setText("paste MenuItem clicked.");
            }
           });
         f.add(label); f.add(popupmenu);
         f.setSize(400,400);
         f.setLayout(null);
         f.setVisible(true);
     }
public static void main(String args[])
{
        new PopupMenuExample();
} }
```

Java JSeparator

The object of JSeparator class is used to provide a general purpose component for implementing divider lines. It is used to draw a line to separate widgets in a Layout. It inherits JComponent class.

JSeparator class declaration

public class JSeparator **extends** JComponent **implements** SwingConstants, Accessible

Commonly used Constructors of JSeparator

Constructor	Description
JSeparator()	Creates a new horizontal separator.
JSeparator(int orientation)	Creates a new separator with the specified horizontal or vertical orientation.

Commonly used Methods of JSeparator

Method	Description
void setOrientation(int orientation)	It is used to set the orientation of the separator.
int getOrientation()	It is used to return the orientation of the separator.

Java JSeparator Example 1

```
import javax.swing.*;
class SeparatorExample
{
        JMenu menu, submenu;
        JMenuItem i1, i2, i3, i4, i5;
        SeparatorExample()  {
        JFrame f= new JFrame("Separator Example");
        JMenuBar mb=new JMenuBar();
        menu=new JMenu("Menu");
        i1=new JMenuItem("Item 1");
        i2=new JMenuItem("Item 2");
        menu.add(i1);
        menu.addSeparator();
        menu.add(i2);
        mb.add(menu);
        f.setJMenuBar(mb);
        f.setSize(400,400);
        f.setLayout(null);
        f.setVisible(true);
}
public static void main(String args[])
{
new SeparatorExample();
}}
```

Java JSeparator Example 2

```
import javax.swing.*;
import java.awt.*;
public class SeparatorExample
{
   public static void main(String args[]) {
      JFrame f = new JFrame("Separator Example");
      f.setLayout(new GridLayout(0, 1));
      JLabel l1 = new JLabel("Above Separator");
      f.add(l1);
      JSeparator sep = new JSeparator();
      f.add(sep);
      JLabel l2 = new JLabel("Below Separator");
      f.add(l2);
      f.setSize(400, 100);
      f.setVisible(true);
    }
  }
```

Java JProgressBar

The JProgressBar class is used to display the progress of the task. It inherits JComponent class.

JProgressBar class declaration

Let's see the declaration for javax.swing.JProgressBar class.

```
public class JProgressBar extends JComponent implements SwingConstants, Accessible
```

Commonly used Constructors:

Constructor	Description
JProgressBar()	It is used to create a horizontal progress bar but no string text.
JProgressBar(int min, int max)	It is used to create a horizontal progress bar with the specified minimum and maximum value.
JProgressBar(int orient)	It is used to create a progress bar with the specified orientation, it can be either Vertical or Horizontal by using SwingConstants.VERTICAL and SwingConstants.HORIZONTAL

	constants.
JProgressBar(int orient, int min, int max)	It is used to create a progress bar with the specified orientation, minimum and maximum value.

Commonly used Methods:

Method	Description
void setStringPainted(boolean b)	It is used to determine whether string should be displayed.
void setString(String s)	It is used to set value to the progress string.
void setOrientation(int orientation)	It is used to set the orientation, it may be either vertical or horizontal by using SwingConstants.VERTICAL and SwingConstants.HORIZONTAL constants.
void setValue(int value)	It is used to set the current value on the progress bar.

Java JProgressBar Example

```
import javax.swing.*;
public class ProgressBarExample extends JFrame{
JProgressBar jb;
int i=0,num=0;
ProgressBarExample(){
jb=new JProgressBar(0,2000);
jb.setBounds(40,40,160,30);
jb.setValue(0);
jb.setStringPainted(true);
add(jb);
setSize(250,150);
setLayout(null);
}
public void iterate(){
while(i<=2000){
  jb.setValue(i);
  i=i+20;
  try{Thread.sleep(150);}catch(Exception e){}
}
}
public static void main(String[] args) {
    ProgressBarExample m=new ProgressBarExample();
```

```
        m.setVisible(true);
        m.iterate();
    }
    }
```

Java JDialog

The JDialog control represents a top level window with a border and a title used to take some form of input from the user. It inherits the Dialog class. Unlike JFrame, it doesn't have maximize and minimize buttons.

JDialog class declaration

Let's see the declaration for javax.swing.JDialog class.

```
public class JDialog extends Dialog implements WindowConstants, Accessible, RootPaneC
ontainer
```

Commonly used Constructors:

Constructor	Description
JDialog()	It is used to create a modeless dialog without a title and without a specified Frame owner.
JDialog(Frame owner)	It is used to create a modeless dialog with specified Frame as its owner and an empty title.
JDialog(Frame owner, String title, boolean modal)	It is used to create a dialog with the specified title, owner Frame and modality.

Java JDialog Example

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
public class DialogExample {
    private static JDialog d;
    DialogExample() {
        JFrame f= new JFrame();
        d = new JDialog(f , "Dialog Example", true);
        d.setLayout( new FlowLayout() );
        JButton b = new JButton ("OK");
        b.addActionListener ( new ActionListener()
        {
            public void actionPerformed( ActionEvent e )
            {
                DialogExample.d.setVisible(false);
            }
        });
        d.add( new JLabel ("Click button to continue."));
        d.add(b);
        d.setSize(300,300);
        d.setVisible(true);
    }
    public static void main(String args[])
    {
        new DialogExample();
    }
}
```