Saraswati Vandana

या कुन्देन्दु तुषार हार धवला
या शुभ्र वस्त्रान्विता ।
या वीणा वर दंड मंडितकरा
या श्वेत पद्मासना ॥

या ब्रह्मा अच्युत शंकर प्रभ्रतिभिः
देवै सदा पूजिता ।
सा मां पातु सरस्वती भगवती
निःश्येश जाड्यापह ॥

SANKALCHAND PATEL UNIVERSITY
।। अथातो ज्ञानजिज्ञासा ।।

Faculty of Engineering & Technology
Sankalchand Patel College of Engineering, Visnagar

Java Programming
(1ET1030406)

Unit-10 : Event and GUI programming

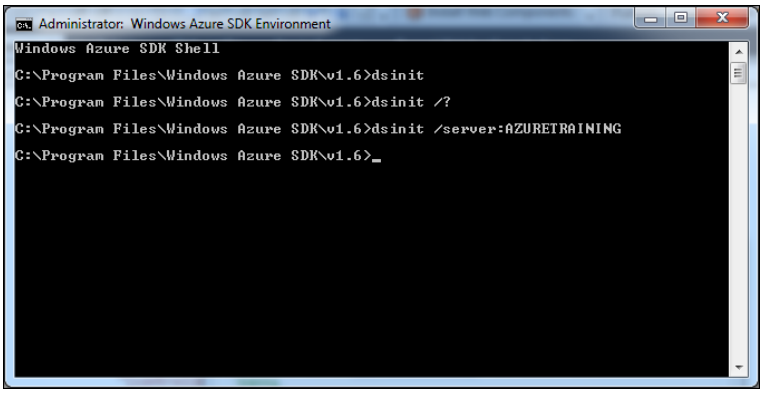
Prepared By
Mr. Mehul S. Patel
Department of Computer Engineering & Information Technology

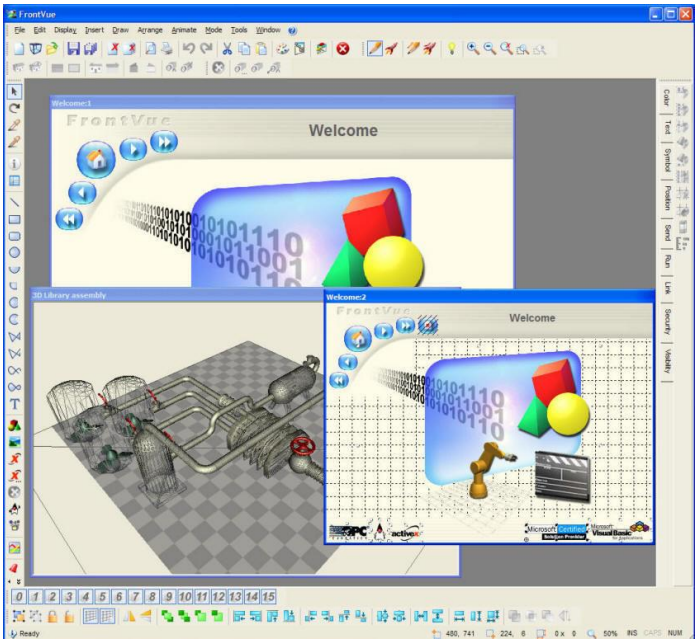
Content

- Event handling in java
- Event types
  - Mouse and key events
- GUI Basics
  - Panels
  - Frames
- Layout Managers:
  - Flow Layout
  - Border Layout
  - Grid Layout
- GUI components
  - Buttons
  - Check Boxes
  - Radio Buttons
  - Labels
  - Text Fields
  - Text Areas
  - Combo Boxes
  - Lists
  - Scroll Bars
  - Sliders
  - Windows
  - Menus
  - Dialog Box
- Applet and its life cycle
- Introduction to swing

Concepts

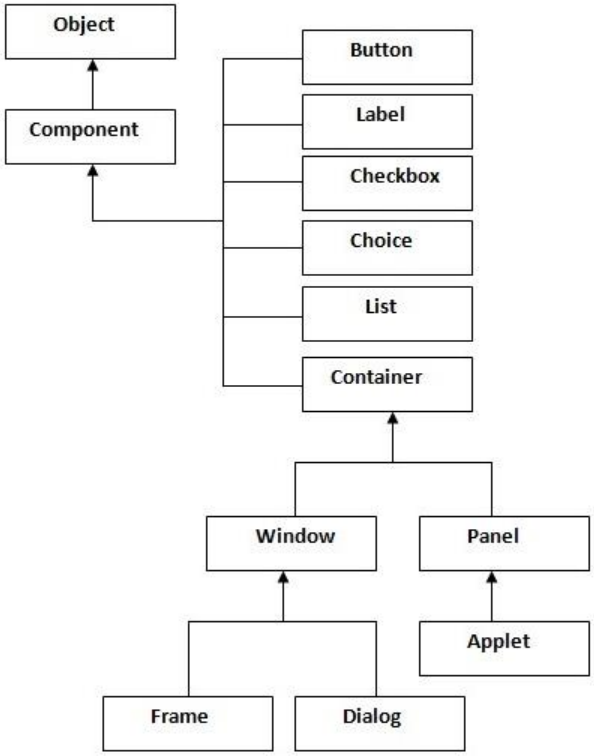
- Conventional Programming
- Event-driven Programming



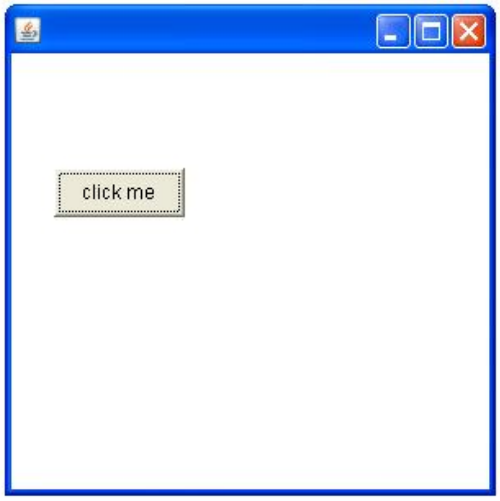


AWT

- Abstract Window Toolkit
- package:
  - java.awt
  - java.awt.event
- heavyweight components using native GUI system elements
- Like : Frame, Dialog, Label, Button or Applet …



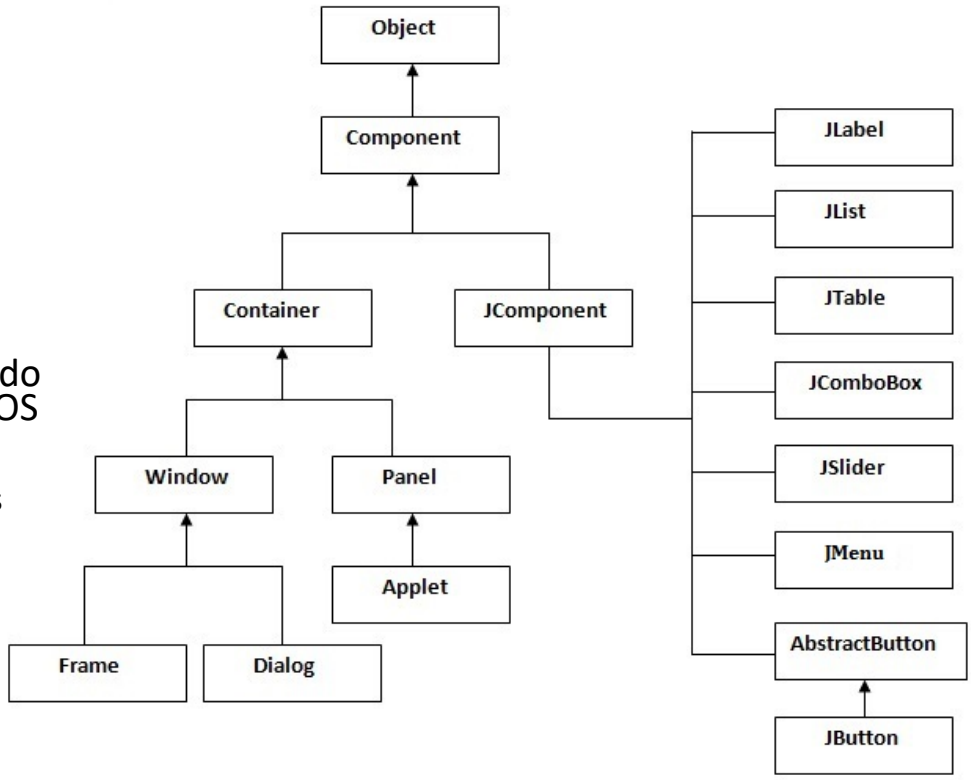
AWT Example




```
import java.awt.*;
class First extends Frame{
First(){
Button b=new Button("click me");
b.setBounds(30,100,80,30);// setting button position
add(b);//adding button into frame
setSize(300,300);//frame size 300 width and 300 height
setLayout(null);//no layout manager
setVisible(true);//now frame will be visible, by default not visible
}
public static void main(String args[]){
First f=new First();
}}
```

Swing

- JDL 1.2+
- package:
  - javax.swing
  - javax.swing.event
- lightweight components that do not rely on the native GUI or OS
- “look and feel”
  - identical on different platforms
  - customized
- inherits from AWT
- AWT still used for events, layouts
- Like JFrame, JDialog, or JApplet



Swing Example

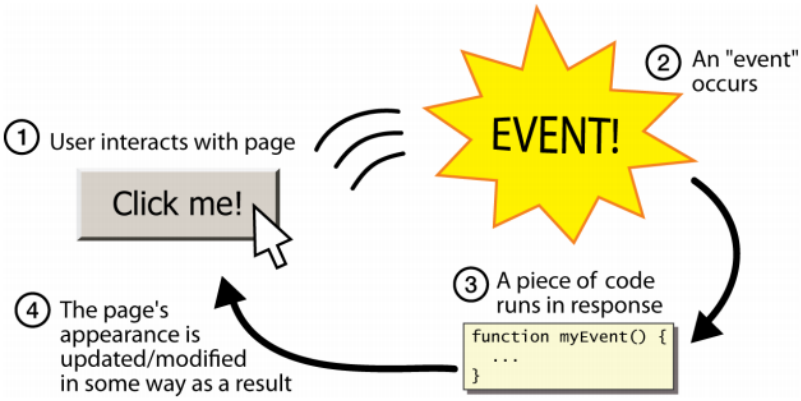

```
import javax.swing.*;
public class FirstSwingExample {
public static void main(String[] args) {
JFrame f=new JFrame();//creating instance of JFrame
JButton b=new JButton("click");//creating instance
b.setBounds(130,100,100, 40);//x axis, y axis, width, height

f.add(b);//adding button in JFrame
f.setSize(400,500);//400 width and 500 height
f.setLayout(null);//using no layout managers
f.setVisible(true);//making the frame visible
}
}
```

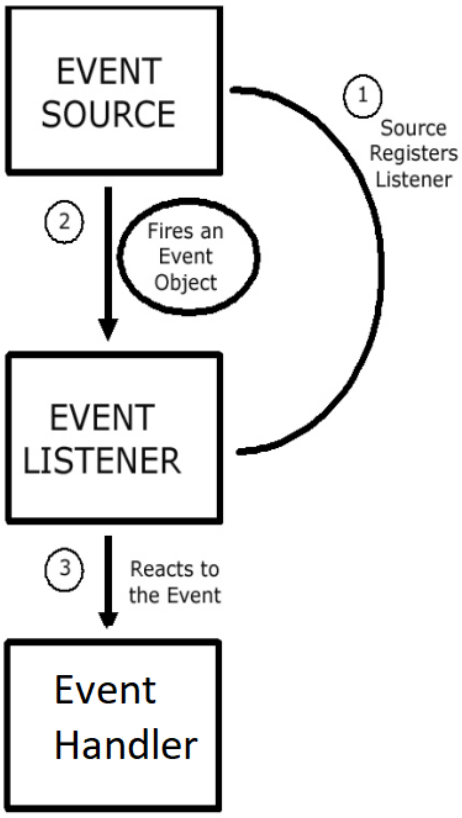
AWT Vs Swing

Java AWT	Java Swing
platform-dependent.	**platform-independent**.
heavyweight.	**lightweight**.
doesn't support pluggable look and feel.	**supports pluggable look and feel**.
less components than Swing.	**more powerful components**such as tables, lists, scrollpanes, colorchooser, tabbedpane etc.
doesn't follows MVC (Model View Controller)	**follows** MVC.

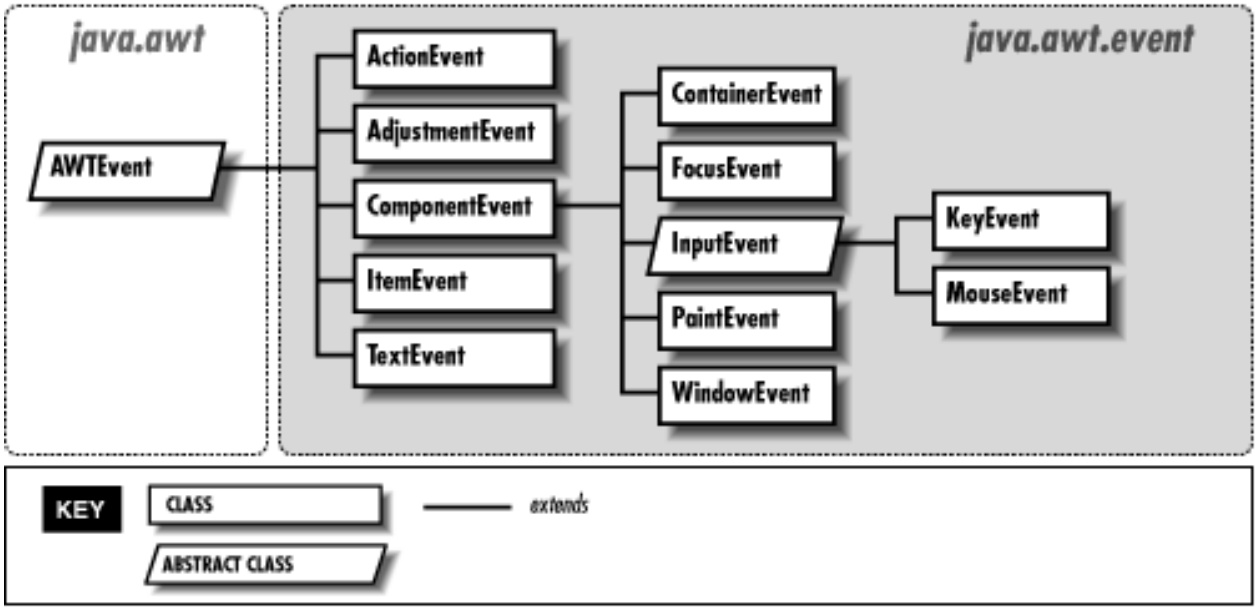
Event

- Action
- Listener
- Handler





Event Classes

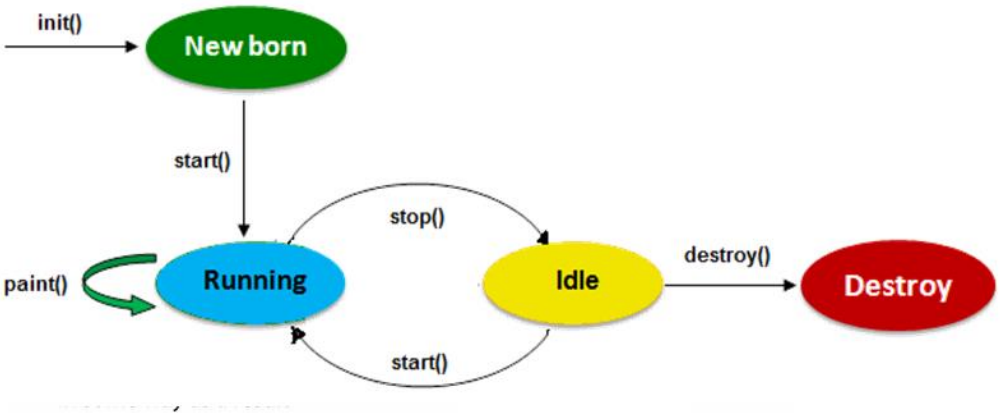


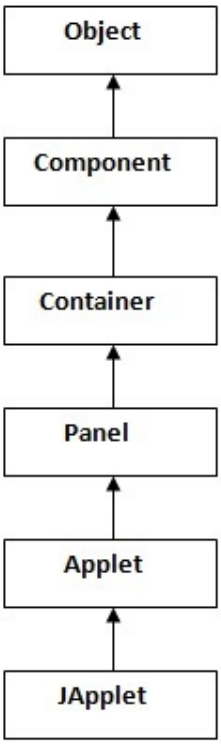
Event Listeners

EVENTS	SOURCE	LISTENERS
Action Event	Button, List, MenuItem, Text field	ActionListener
Component Event	Component	Component Listener
Focus Event	Component	FocusListener
Item Event	Checkbox, CheckboxMenuItem, Choice, List	ItemListener
Key Event	when input is received from keyboard	KeyListener
Text Event	Text Component	TextListener
Window Event	Window	WindowListener
Mouse Event	Mouse related event	MouseListener

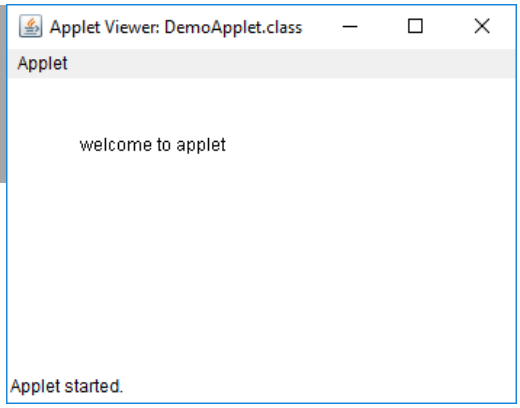
Applet - Life Cycle

- Applet - embedded in the webpage to generate the dynamic content





Example




```
//DemoApplet.java
import java.applet.Applet;
import java.awt.Graphics;
public class DemoApplet extends Applet{
        public void paint(Graphics g){
                g.drawString("welcome to applet",50,50);
        }
  }
/*
<applet code="DemoApplet.class" width="300" height="300">
</applet>
*/
```

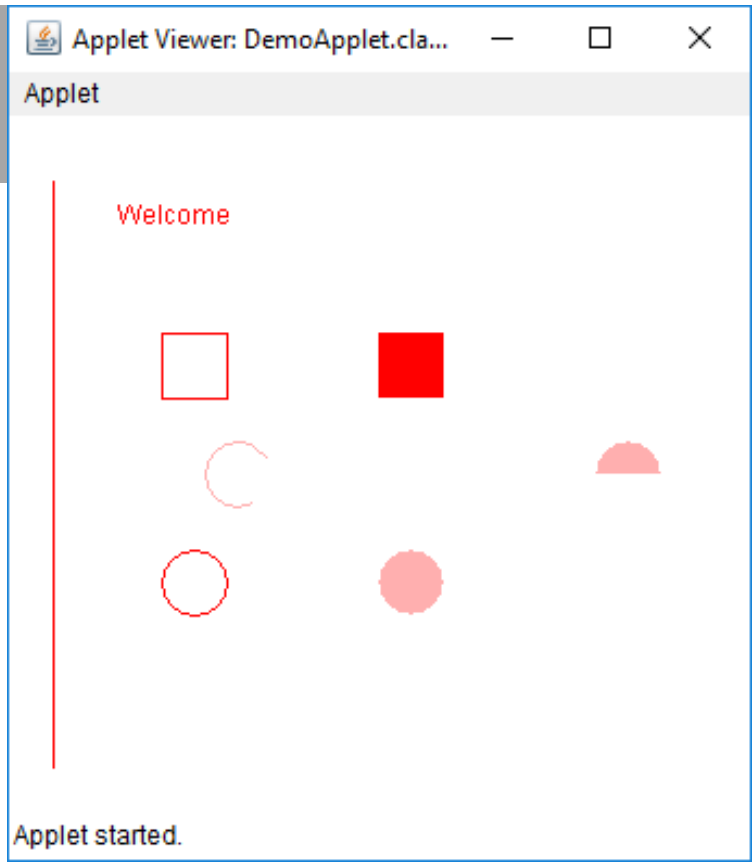
Run Applet

```
<html>
<body>
<applet code="DemoApplet.class"
                    width="300" height="300">
</applet>
</body>
</html>
```

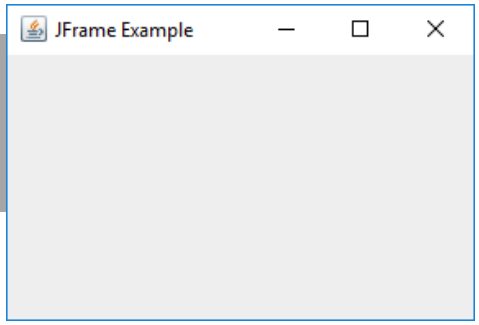
```
c:\>javac First.java
c:\>appletviewer First.java
```

Graphics

```
import java.applet.Applet;
import java.awt.*;
public class GraphicsDemo extends Applet{
public void paint(Graphics g){
g.setColor(Color.red);
g.drawString("Welcome",50, 50);
g.drawLine(20,30,20,300);
g.drawRect(70,100,30,30);
g.fillRect(170,100,30,30);
g.drawOval(70,200,30,30);
g.setColor(Color.pink);
g.fillOval(170,200,30,30);
g.drawArc(90,150,30,30,30,270);
g.fillArc(270,150,30,30,0,180);
}}
```



JFrame




```
import javax.swing.JFrame;
public class JFrameExample {
    public static void main(String s[]) {
        JFrame frame = new JFrame("JFrame Example");
        frame.setSize(300, 200);
        frame.setLocationRelativeTo(null);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setVisible(true);
    }
}
```

JLabel

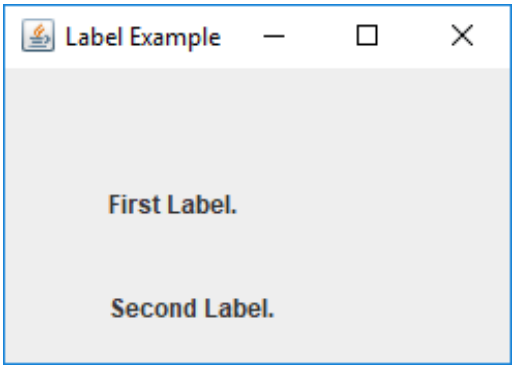
Constructor	Description
JLabel()	Creates a JLabel instance with no image and with an empty string for the title.
JLabel(String s)	Creates a JLabel instance with the specified text.
JLabel(Icon i)	Creates a JLabel instance with the specified image.
JLabel(String s, Icon i, int horizontalAlignment)	Creates a JLabel instance with the specified text, image, and horizontal alignment.

JLabel (Cont.)

Methods	Description
String getText()	t returns the text string that a label displays.
void setText(String text)	It defines the single line of text this component will display.
void setHorizontalAlignment(int alignment)	It sets the alignment of the label's contents along the X axis.
Icon getIcon()	It returns the graphic image that the label displays.
int getHorizontalAlignment()	It returns the alignment of the label's contents along the X axis.

JLabel (Cont.)

```
import javax.swing.*;
class JLabelExample {
public static void main(String args[])      {
    JFrame f= new JFrame("Label Example");
    JLabel l1,l2;
    l1=new JLabel("First Label.");
    l1.setBounds(50,50, 100,30);
    l2=new JLabel("Second Label.");
    l2.setBounds(50,100, 100,30);
    f.add(l1); f.add(l2);
    f.setSize(300,300);
    f.setLayout(null);
    f.setVisible(true);
    }  }
```



JButton

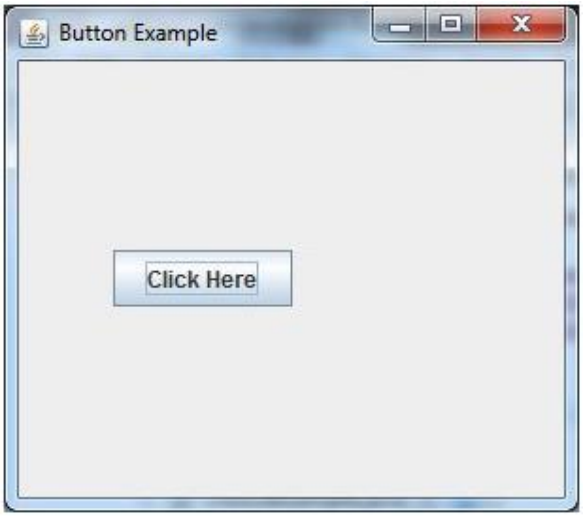
Constructor	Description
JButton()	It creates a button with no text and icon.
JButton(String s)	It creates a button with the specified text.
JButton(Icon i)	It creates a button with the specified icon object.

JButton (Cont.)

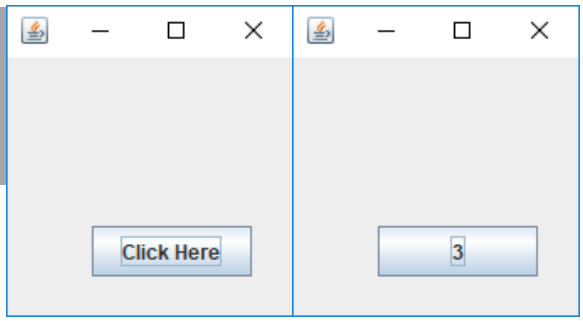
Methods	Description
void setText(String s)	It is used to set specified text on button
String getText()	It is used to return the text of the button.
void setEnabled(boolean b)	It is used to enable or disable the button.
void setIcon(Icon b)	It is used to set the specified Icon on the button.
Icon getIcon()	It is used to get the Icon of the button.
void setMnemonic(int a)	It is used to set the mnemonic on the button.
void addActionListener(ActionListener a)	It is used to add the action listener to this object.

JButton (Cont.)

```
import javax.swing.*;
public class ButtonExample {
public static void main(String[] args) {
    JFrame f=new JFrame("Button Example");
    JButton b=new JButton("Click Here");
    b.setBounds(50,100,95,30);
    f.add(b);
    f.setSize(400,400);
    f.setLayout(null);
    f.setVisible(true);
}
}
```



JButton (Cont.)




```
import java.awt.event.*;
import javax.swing.*;
public class ButtonExample extends JFrame
                    implements ActionListener
{
        JButton b;
        int counter=0;

ButtonExample(){
        b=new JButton("Click Here");
        b.setBounds(50,100,95,30);
        b.addActionListener(this);
        add(b);
}
void actionPerformed(ActionEvent e){
   counter++;
   b.setText(String.valueOf(counter));
   }
public static void main(String[] args) {
   ButtonExample f=new ButtonExample();
   f.setSize(400,400);
   f.setLayout(null);
   f.setVisible(true);
}}
```

JTextField

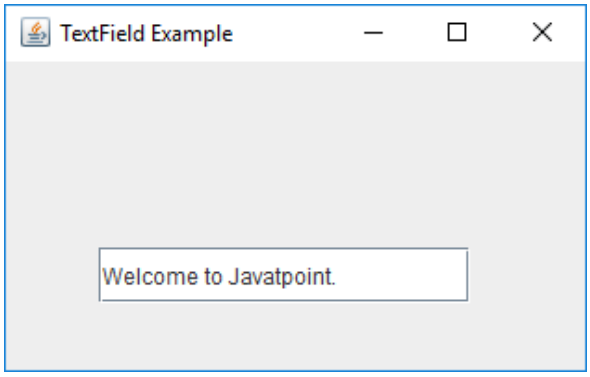
Constructor	Description
JTextField()	Creates a new TextField
JTextField(String text)	Creates a new TextField initialized with the specified text.
JTextField(String text, int columns)	Creates a new TextField initialized with the specified text and columns.
JTextField(int columns)	Creates a new empty TextField with the specified number of columns.

JTextField (Cont.)

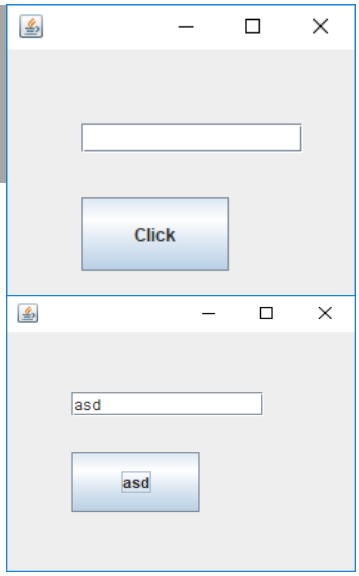
Methods	Description
void addActionListener(ActionListener l)	It is used to add the specified action listener to receive action events from this textfield.
Action getAction()	It returns the currently set Action for this ActionEvent source, or null if no Action is set.
void setFont(Font f)	It is used to set the current font.
void removeActionListener(ActionListener l)	It is used to remove the specified action listener so that it no longer receives action events from this textfield.

JTextField (Cont.)

```
import javax.swing.*;
class JLabelExample  {
public static void main(String args[])      {
JFrame f= new JFrame("TextField Example");
    JTextField t1;
    t1=new JTextField("Welcome to Javatpoint.");
    t1.setBounds(50,100, 200,30);
    f.add(t1);
    f.setSize(400,400);
    f.setLayout(null);
    f.setVisible(true);
    }
    }
```



JTextField (Cont.)




```
import javax.swing.*;
import java.awt.event.*;
public class TextFieldExample
implements ActionListener{
    JTextField tf1;
    JButton b1;
    TextFieldExample(){
        JFrame f= new JFrame();
        tf1=new JTextField();
        tf1.setBounds(50,50,150,20);
        b1=new JButton("Click");
        b1.setBounds(50,100,100,50);
        b1.addActionListener(this);
        f.add(tf1);
        f.add(b1);
        f.setSize(300,300);
        f.setLayout(null);
        f.setVisible(true);
    }
    public void actionPerformed(ActionEvent e){
            String s1=tf1.getText();
            b1.setText(s1);  }
    public static void main(String[] args) {
        new TextFieldExample();
    } }
```

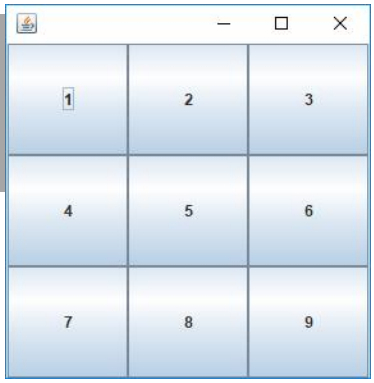
LayoutManagers

1. BorderLayout
2. BoxLayout
3. CardLayout
4. FlowLayout
5. GridBagLayout
6. GridLayout
7. GroupLayout
8. SpringLayout

BorderLayout

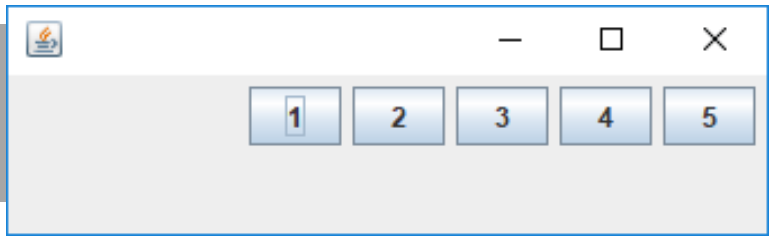

```
import java.awt.*;  import javax.swing.*;
public class Border {
JFrame f;
Border(){
   f=new JFrame();
   JButton b1=new JButton("NORTH");;
   JButton b2=new JButton("SOUTH");;
   JButton b3=new JButton("EAST");;
   JButton b4=new JButton("WEST");;
   JButton b5=new JButton("CENTER");;
   f.add(b1,BorderLayout.NORTH);
   f.add(b2,BorderLayout.SOUTH);
   f.add(b3,BorderLayout.EAST);
   f.add(b4,BorderLayout.WEST);
   f.add(b5,BorderLayout.CENTER);
   f.setSize(300,300);
   f.setVisible(true);
}
public static void main(String[] args) {
   new Border();
} }
```

GridLayout



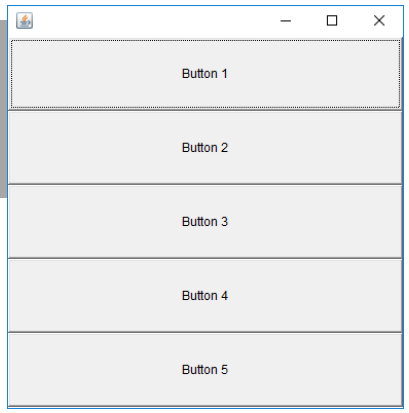

```
import java.awt.*;  import javax.swing.*;
public class MyGridLayout{
JFrame f;
MyGridLayout(){
    f=new JFrame();
    JButton b1=new JButton("1");
    JButton b2=new JButton("2");
    JButton b3=new JButton("3");
    JButton b4=new JButton("4");
    JButton b5=new JButton("5");
    JButton b6=new JButton("6");
    JButton b7=new JButton("7");
    JButton b8=new JButton("8");
    JButton b9=new JButton("9");
    f.add(b1);f.add(b2);f.add(b3);
    f.add(b4);f.add(b5);  f.add(b6);
    f.add(b7);f.add(b8);f.add(b9);
    f.setLayout(new GridLayout(3,3));
     f.setSize(300,300);
     f.setVisible(true);
}
public static void main(String[] args) {
    new MyGridLayout();
}  }
```

FlowLayout



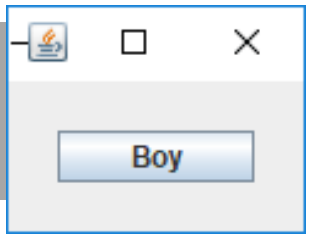

```
import java.awt.*;
import javax.swing.*;

public class MyFlowLayout{
JFrame f;
MyFlowLayout(){
    f=new JFrame();

    JButton b1=new JButton("1");
    JButton b2=new JButton("2");
    JButton b3=new JButton("3");
    JButton b4=new JButton("4");
    JButton b5=new JButton("5");

f.add(b1);f.add(b2);f.add(b3);
f.add(b4);f.add(b5);

f.setLayout(new FlowLayout(FlowLayout.RIGHT));


    f.setSize(300,300);
    f.setVisible(true);
}
public static void main(String[] args) {
    new MyFlowLayout();
} }
```

BoxLayout




```
import java.awt.*;
import javax.swing.*;
 public class BoxLayoutExample1
                      extends Frame {
 Button buttons[];
  public BoxLayoutExample1 () {
   buttons = new Button [5];

   for (int i = 0;i<5;i++) {
   buttons[i] = new Button ("Button " + (i + 1));
   add (buttons[i]);
    }
setLayout (new BoxLayout (this, BoxLayout.Y_AXIS));

setSize(400,400);
setVisible(true);
}

public static void main(String args[]){
BoxLayoutExample1 b=new BoxLayoutExample1();
} }
```

CardLayout




```
import java.awt.*;  import java.awt.event.*;
import javax.swing.*;
ublic class CardLayoutExample extends JFrame
                    implements ActionListener{
CardLayout card;
JButton b1,b2,b3;
Container c;
CardLayoutExample(){
        c=getContentPane();
        card=new CardLayout(40,30);
        c.setLayout(card);
        b1=new JButton("Apple");
        b2=new JButton("Boy");
        b3=new JButton("Cat");
        b1.addActionListener(this);
        b2.addActionListener(this);
        b3.addActionListener(this);
        c.add("a",b1);c.add("b",b2);c.add("c",b3);
    }
        public void actionPerformed(ActionEvent e) {
                card.next(c);
    }
public static void main(String[] args) {
CardLayoutExample cl=new CardLayoutExample();
        cl.setSize(400,400);
        cl.setVisible(true);
        cl.setDefaultCloseOperation(EXIT_ON_CLOSE);
    } }
```

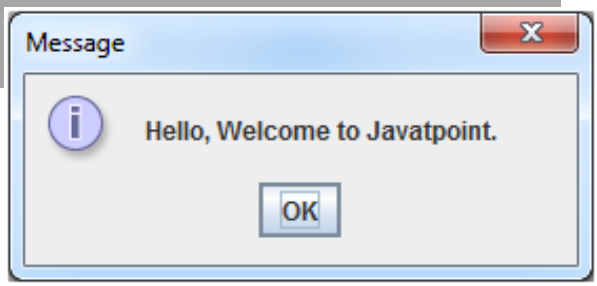
JOptionPane

Constructor	Description
JOptionPane()	It is used to create a JOptionPane with a test message.
JOptionPane(Object message)	It is used to create an instance of JOptionPane to display a message.
JOptionPane(Object message, int messageType)	It is used to create an instance of JOptionPane to display a message with specified message type and default options.

JOptionPane (Cont.)

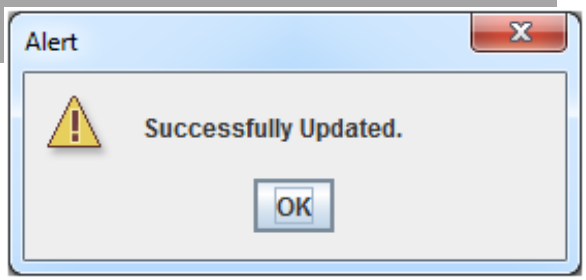
Methods	Description
JDialog createDialog(String title)	It is used to create and return a new parentless JDialog with the specified title.
static void showMessageDialog(Component parentComponent, Object message)	It is used to create an information-message dialog titled "Message".
static void showMessageDialog(Component parentComponent, Object message, String title, int messageType)	It is used to create a message dialog with given title and messageType.
static int showConfirmDialog(Component parentComponent, Object message)	It is used to create a dialog with the options Yes, No and Cancel; with the title, Select an Option.
static String showInputDialog(Component parentComponent, Object message)	It is used to show a question-message dialog requesting input from the user parented to parentComponent.
void setInputValue(Object newValue)	It is used to set the input value that was selected or input by the user.

JOptionPane (Cont.)



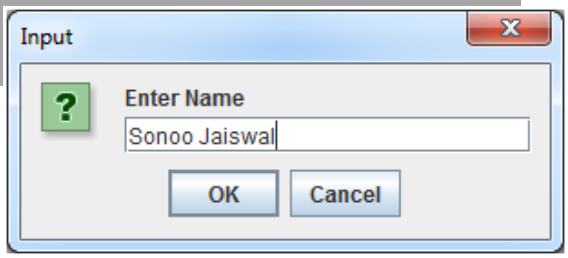

```
import javax.swing.*;
public class OptionPaneExample {
JFrame f;
OptionPaneExample(){
    f=new JFrame();
    JOptionPane.showMessageDialog(f,"Hello, Welcome to Javatpoint.");
}
public static void main(String[] args) {
    new OptionPaneExample();
}
}
```

JOptionPane (Cont.)



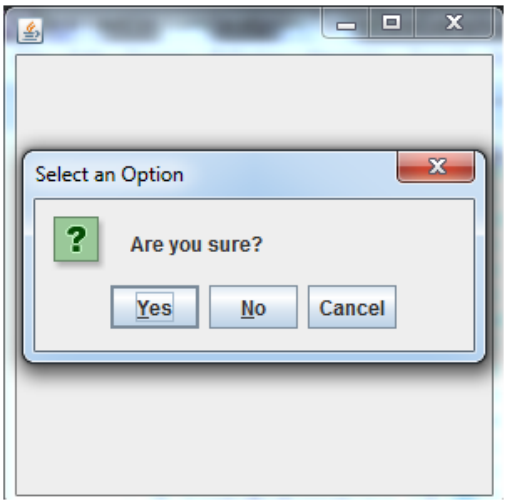

```
import javax.swing.*;
public class OptionPaneExample {
JFrame f;
OptionPaneExample(){
    f=new JFrame();
    JOptionPane.showMessageDialog(f,"Successfully Updated.","Alert",
                                                    JOptionPane.WARNING_MESSAGE);
}
public static void main(String[] args) {
    new OptionPaneExample();
}
}
```

JOptionPane (Cont.)




```
import javax.swing.*;
public class OptionPaneExample {
JFrame f;
OptionPaneExample(){
    f=new JFrame();
    String name=JOptionPane.showInputDialog(f,"Enter Name");
}
public static void main(String[] args) {
    new OptionPaneExample();
}
}
```

JOptionPane (Cont.)




```
import javax.swing.*;  import java.awt.event.*;
public class OptionPaneExample extends WindowAdapter{
   JFrame f;
   OptionPaneExample(){
    f=new JFrame();
    f.addWindowListener(this);
    f.setDefaultCloseOperation(JFrame.DO_NOTHING_ON_CLOSE);
    f.setVisible(true);  }
public void windowClosing(WindowEvent e) {
    int a=JOptionPane.showConfirmDialog(f,"Are you sure?");
    if(a==JOptionPane.YES_OPTION)
            f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
}
public static void main(String[] args) {
    new  OptionPaneExample();  }     }
```

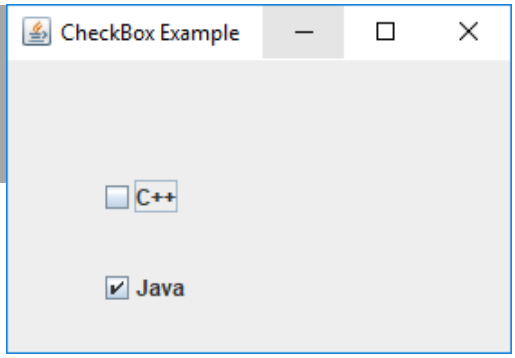
JCheckBox

Constructor	Description
JJCheckBox()	Creates an initially unselected check box button with no text, no icon.
JChechBox(String s)	Creates an initially unselected check box with text.
JCheckBox(String text, boolean selected)	Creates a check box with text and specifies whether or not it is initially selected.
JCheckBox(Action a)	Creates a check box where properties are taken from the Action supplied.

JCheckBox (Cont.)

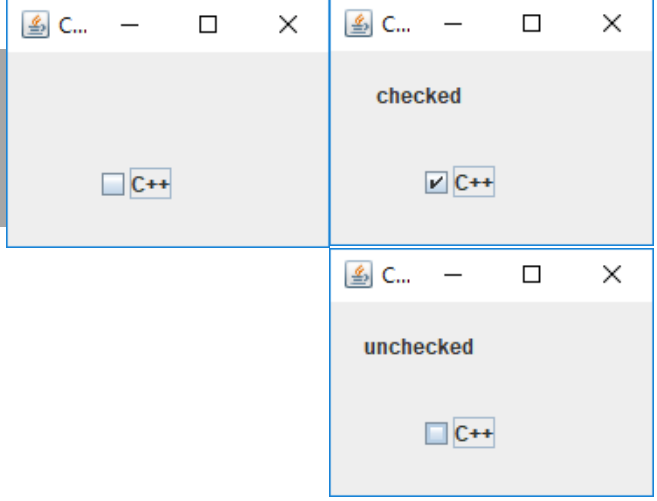
Methods	Description
AccessibleContext getAccessibleContext()	It is used to get the AccessibleContext associated with this JCheckBox.
protected String paramString()	It returns a string representation of this JCheckBox.

JCheckBox (Cont.)




```
import javax.swing.*;
public class CheckBoxExample  {
   CheckBoxExample(){
        JFrame f= new JFrame("CheckBox Example");
        JCheckBox checkBox1 = new JCheckBox("C++");
        checkBox1.setBounds(50,50, 150,50);
        JCheckBox checkBox2 = new JCheckBox("Java", true);
        checkBox2.setBounds(50,100, 150,50);
        f.add(checkBox1);          f.add(checkBox2);
        f.setSize(200,200);           f.setLayout(null);
        f.setVisible(true);        }
public static void main(String args[])      {
    new CheckBoxExample();     }     }
```

JCheckBox (Cont.)




```
import javax.swing.*;import java.awt.event.*;
public class CheckBoxExample
                implements ItemListener {
JLabel label ;
CheckBoxExample(){
  JFrame f= new JFrame("CheckBox Example");
  label = new JLabel();
  label.setSize(100,100);
  JCheckBox checkbox1 = new JCheckBox("C++");
  checkbox1.setBounds(50,100, 50,50);
  f.add(checkbox1); f.add(label);
  checkbox1.addItemListener(this);
  f.setSize(200,200);  f.setLayout(null);
  f.setVisible(true);         }
public void itemStateChanged(ItemEvent e) {
String state =
e.getStateChange()==1?"checked":"unchecked";
label.setText(state);
}
public static void main(String args[])    {
    new CheckBoxExample();
}    }
```

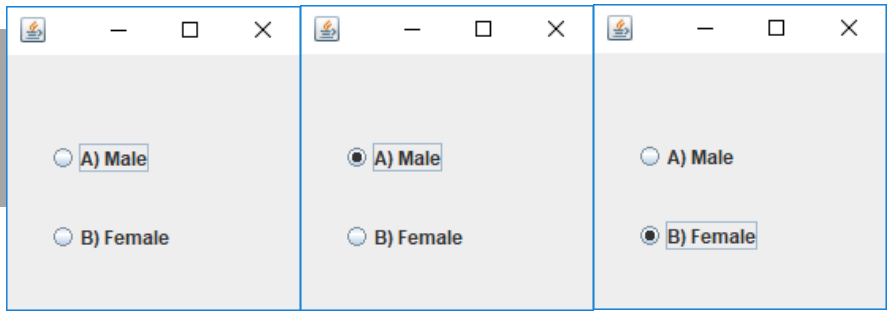
JRadioButton

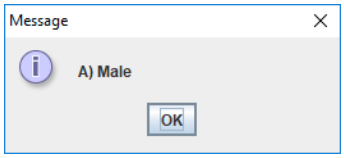
Constructor	Description
JRadioButton()	Creates an unselected radio button with no text.
JRadioButton(String s)	Creates an unselected radio button with specified text.
JRadioButton(String s, boolean selected)	Creates a radio button with the specified text and selected status.

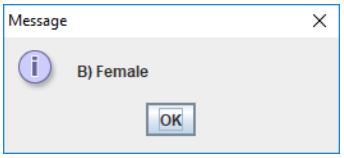
JRadioButton(Cont.)

Methods	Description
void setText(String s)	It is used to set specified text on button.
String getText()	It is used to return the text of the button.
void setEnabled(boolean b)	It is used to enable or disable the button.
void setIcon(Icon b)	It is used to set the specified Icon on the button.
Icon getIcon()	It is used to get the Icon of the button.
void setMnemonic(int a)	It is used to set the mnemonic on the button.
void addActionListener(ActionListener a)	It is used to add the action listener to this object.

JRadioButton(Cont.)








```
import java.awt.event.*; import javax.swing.*;
public class RadioButtonExample
                implements ItemListener{
JFrame f;
RadioButtonExample() {
f=new JFrame();
JRadioButton r1=new JRadioButton("A) Male");
JRadioButton r2=new JRadioButton("B) Female");
r1.setBounds(75,50,100,30); r2.setBounds(75,100,100,30);
ButtonGroup bg=new ButtonGroup();
bg.add(r1);bg.add(r2);
r1.addItemListener(this); r2.addItemListener(this);
f.add(r1);      f.add(r2);
f.setSize(200,200); f.setLayout(null); f.setVisible(true); }
```

```
public void itemStateChanged(ItemEvent e) {
JRadioButton r=(JRadioButton)e.getItem();
if(r.isSelected())
JOptionPane.showMessageDialog(f,r.getText());
}
public static void main(String[] args) {
    new RadioButtonExample(); } }
```

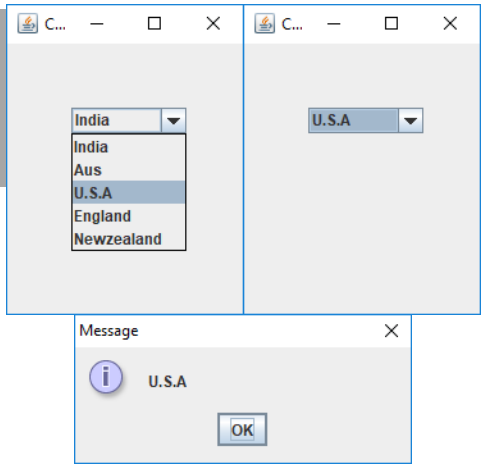
JComboBox

Constructor	Description
JComboBox()	Creates a JComboBox with a default data model.
JComboBox(Object[] items)	Creates a JComboBox that contains the elements in the specified array.
JComboBox(Vector<?> items)	Creates a JComboBox that contains the elements in the specified Vector.

JComboBox (Cont.)

Methods	Description
void addItem(Object anObject)	It is used to add an item to the item list.
void removeItem(Object anObject)	It is used to delete an item to the item list.
void removeAllItems()	It is used to remove all the items from the list.
void setEditable(boolean b)	It is used to determine whether the JComboBox is editable.
void addActionListener(ActionListener a)	It is used to add the ActionListener.
void addItemListener(ItemListener i)	It is used to add the ItemListener.

JComboBox (Cont.)




```
import java.awt.event.*;import javax.swing.*;
public class ComboBoxExample
        implements ItemListener {
JFrame f;
ComboBoxExample(){
    f=new JFrame("ComboBox Example");
    String country[]={"India","Aus",
       "U.S.A","England","Newzealand"};
    JComboBox cb=new JComboBox(country);
    cb.setBounds(50, 50,90,20);
    cb.addItemListener(this);
    f.add(cb);
    f.setLayout(null);   f.setSize(200,250);
    f.setVisible(true); }
public void itemStateChanged(ItemEvent e) {
if(e.getStateChange()==1)
        JOptionPane.showMessageDialog
                        (f, e.getItem());
}
public static void main(String[] args) {
    new ComboBoxExample();   }
}
```

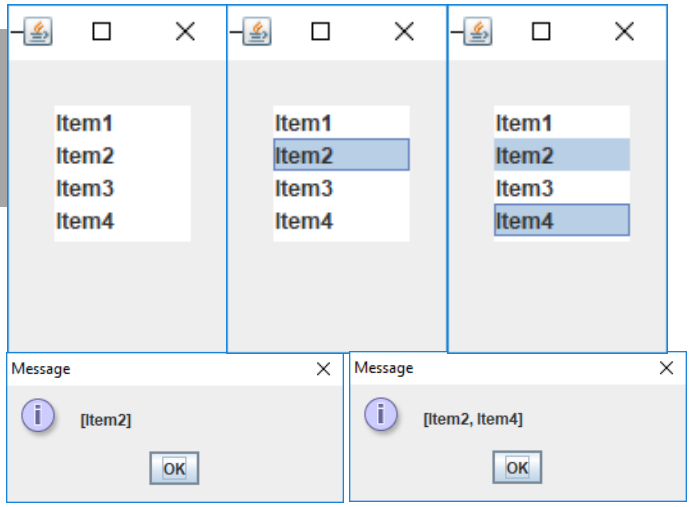
JList

Constructor	Description
JList()	Creates a JList with an empty, read-only, model.
JList(ary[] listData)	Creates a JList that displays the elements in the specified array.
JList(ListModel<ary> dataModel)	Creates a JList that displays elements from the specified, non-null, model.

JList (Cont.)

Methods	Description
Void addListSelectionListener(ListSelectionListener listener)	It is used to add a listener to the list, to be notified each time a change to the selection occurs.
int getSelectedIndex()	It is used to return the smallest selected cell index.
ListModel getModel()	It is used to return the data model that holds a list of items displayed by the JList component.
void setListData(Object[] listData)	It is used to create a read-only ListModel from an array of objects.

JList (Cont.)

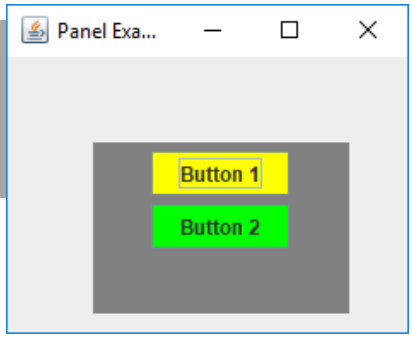



```
import javax.swing.*;import javax.swing.event.*;
public class ListExample
       implements ListSelectionListener  {
JFrame f;      JList<String> list;
ListExample(){  f= new JFrame();
DefaultListModel<String> l1 =
              new DefaultListModel<>();
l1.addElement("Item1");l1.addElement("Item2");
l1.addElement("Item3");l1.addElement("Item4");
list = new JList<>(l1);
list.setBounds(25,25, 75,75);
list.addListSelectionListener(this);
f.add(list); f.setSize(100,200);
          f.setVisible(true);       }
public void valueChanged(ListSelectionEvent e) {
java.util.List<String>
l=list.getSelectedValuesList();
JOptionPane.showMessageDialog(f, l.toString());
}
public static void main(String args[]){
   new ListExample();     }
}
```

JPanel

Constructor	Description
JPanel()	It is used to create a new JPanel with a double buffer and a flow layout.
JPanel(boolean isDoubleBuffered)	It is used to create a new JPanel with FlowLayout and the specified buffering strategy.
JPanel(LayoutManager layout)	It is used to create a new JPanel with the specified layout manager.

Jpanel (Cont.)

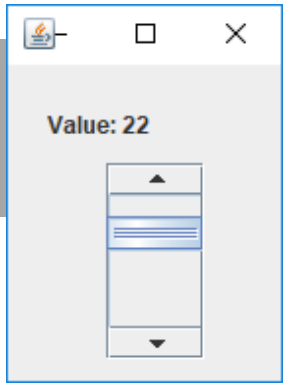



```
import java.awt.*;
import javax.swing.*;
public class PanelExample {
    PanelExample()        {
      JFrame f= new JFrame("Panel Example");
      JPanel panel=new JPanel();
      panel.setBounds(40,80,200,200);
      panel.setBackground(Color.gray);
      JButton b1=new JButton("Button 1");
      b1.setBounds(50,100,80,30);
      b1.setBackground(Color.yellow);
      JButton b2=new JButton("Button 2");
      b2.setBounds(100,100,80,30);
      b2.setBackground(Color.green);
      panel.add(b1); panel.add(b2);
      f.add(panel);
      f.setSize(400,400);
      f.setVisible(true);
      }
      public static void main(String args[]) {
      new PanelExample();
      }
}
```

JScrollBar

Constructor	Description
JScrollBar()	Creates a vertical scrollbar with the initial values.
JScrollBar(int orientation)	Creates a scrollbar with the specified orientation and the initial values.
JScrollBar(int orientation, int value, int extent, int min, int max)	Creates a scrollbar with the specified orientation, value, extent, minimum, and maximum.

JScrollBar (Cont.)




```
import javax.swing.*;
import java.awt.event.*;
class ScrollBarExample
{
ScrollBarExample(){
    JFrame f= new JFrame("Scrollbar Example");
    final JLabel label = new JLabel();
    label.setBounds(20,20,100,20);
    final JScrollBar s=new JScrollBar();
    s.setBounds(50,50, 50,100);
    s.setUnitIncrement(1);
    s.setMinimum(1); s.setMaximum(100);
    f.add(label); f.add(s);
    f.setSize(200,200);
    f.setVisible(true);
    s.addAdjustmentListener(
        new AdjustmentListener() {
    public void
    adjustmentValueChanged(AdjustmentEvent e){
       label.setText("Value: "+ s.getValue());
    }
 });
}
public static void main(String args[])
{
   new ScrollBarExample();
}}
```

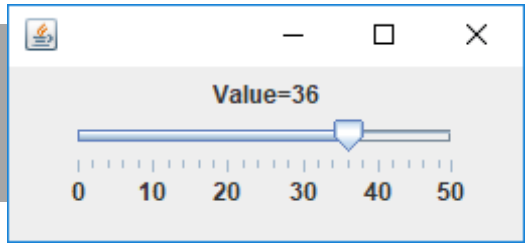
JSlider

Constructor	Description
JSlider()	creates a slider with the initial value of 50 and range of 0 to 100.
JSlider(int orientation)	creates a slider with the specified orientation set by either JSlider.HORIZONTAL or JSlider.VERTICAL with the range 0 to 100 and initial value 50.
JSlider(int min, int max)	creates a horizontal slider using the given min and max.
JSlider(int min, int max, int value)	creates a horizontal slider using the given min, max and value.
JSlider(int orientation, int min, int max, int value)	creates a slider using the given orientation, min, max and value.

JSlider (Cont.)

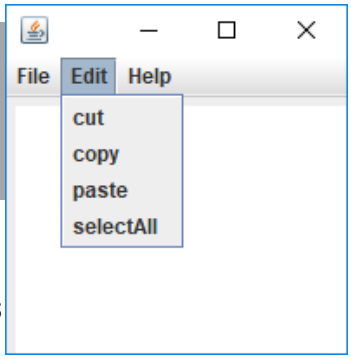
Method	Description
public void setMinorTickSpacing(int n)	is used to set the minor tick spacing to the slider.
public void setMajorTickSpacing(int n)	is used to set the major tick spacing to the slider.
public void setPaintTicks(boolean b)	is used to determine whether tick marks are painted.
public void setPaintLabels(boolean b)	is used to determine whether labels are painted.
public void setPaintTracks(boolean b)	is used to determine whether track is painted.

JSlider (Cont.)




```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.event.*;
public class SliderExample
                        extends JFrame{
public SliderExample() {
    JLabel label=new JLabel("Slider");
    add(label);
    JSlider slider = new
        JSlider(JSlider.HORIZONTAL,0,50,25);
    slider.setMinorTickSpacing(2);
    slider.setMajorTickSpacing(10);
    slider.setPaintTicks(true);
    slider.setPaintLabels(true);
slider.addChangeListener(
        new ChangeListener() {
    public void stateChanged(ChangeEvent e) {
    label.setText("Value="+slider.getValue());
        }
    });
add(slider);
}
public static void main(String s[]) {
SliderExample frame=new SliderExample();
frame.setLayout(new FlowLayout());
frame.pack();
frame.setVisible(true);
}  }
```

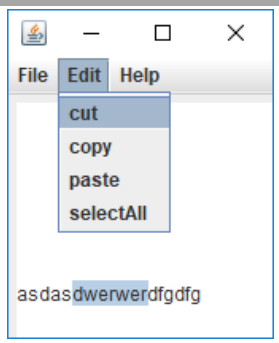
JMenuBar, JMenu and JMenuItem

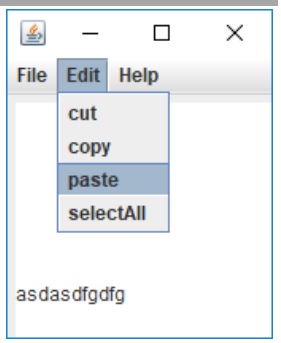


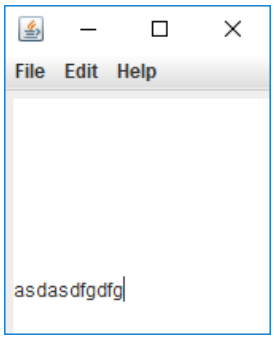

```
import javax.swing.*;
import java.awt.event.*;
public class MenuExample
                    implements ActionListener{
JFrame f;     JMenuBar mb;
JMenu file,edit,help;
JMenuItem cut,copy,paste,selectAll;
JTextArea ta;
MenuExample(){
f=new JFrame();
cut=new JMenuItem("cut");
copy=new JMenuItem("copy");
paste=new JMenuItem("paste");
selectAll=new JMenuItem("selectAll");
cut.addActionListener(this);
copy.addActionListener(this);
paste.addActionListener(this);
selectAll.addActionListener(this);
mb=new JMenuBar();
file=new JMenu("File");
edit=new JMenu("Edit");
help=new JMenu("Help");
edit.add(cut);    edit.add(copy);
edit.add(paste);
edit.add(selectAll);
mb.add(file);     mb.add(edit);     mb.add(help);
ta=new JTextArea();
ta.setBounds(5,5,360,320);
f.add(mb);        f.add(ta);
f.setJMenuBar(mb);
f.setVisible(true);     }
```

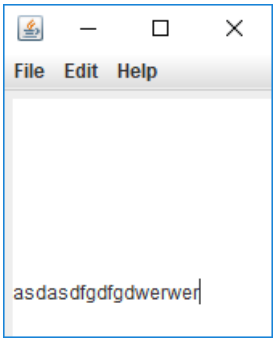
JMenuBar, JMenu and JMenuItem

```
public void actionPerformed(ActionEvent e) {
if(e.getSource()==cut)
        ta.cut();
if(e.getSource()==paste)
        ta.paste();
if(e.getSource()==copy)
        ta.copy();
if(e.getSource()==selectAll)
        ta.selectAll();
}
public static void main(String[] args) {
    new MenuExample();    }}
```









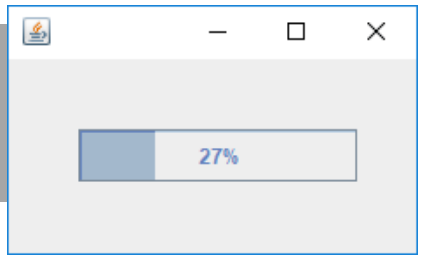
JProgressBar

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.

JProgressBar (Cont.)

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.

JProgressBar (Cont.)

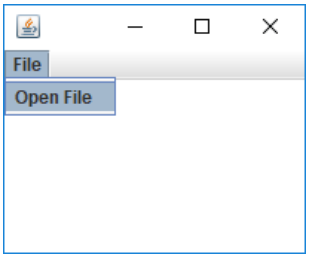


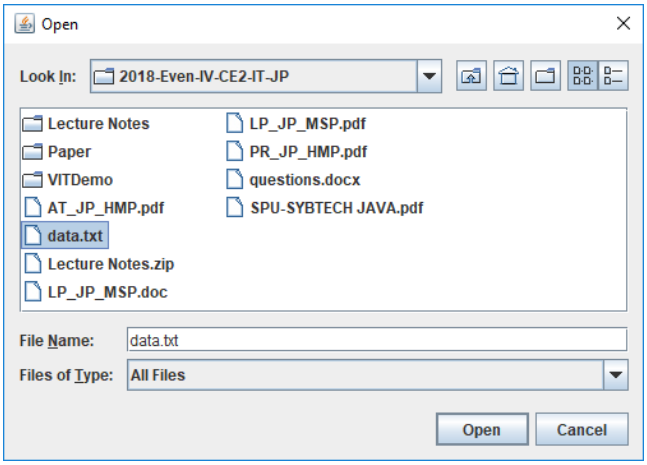

```
import javax.swing.*;
public class ProgressBarExample
                extends JFrame{
JProgressBar jb;
int i=0,num=0;
ProgressBarExample(){
        jb=new JProgressBar(0,2000);
        jb.setBounds(40,40,160,30);
        jb.setValue(0);
        jb.setStringPainted(true);
        add(jb);
        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();
}    }
```

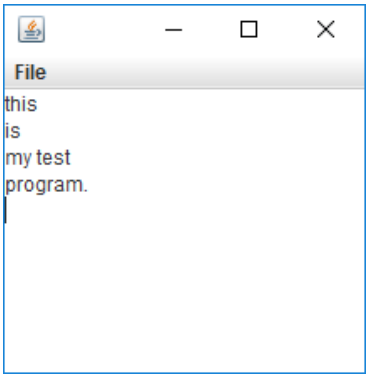
JFileChooser

Constructor	Description
JFileChooser()	Constructs a JFileChooser pointing to the user's default directory.
JFileChooser(File currentDirectory)	Constructs a JFileChooser using the given File as the path.
JFileChooser(String currentDirectoryPath)	Constructs a JFileChooser using the given path.

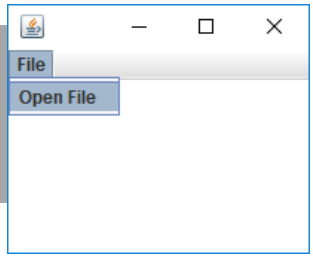
JFileChooser (Cont.)







JFileChooser (Cont.)



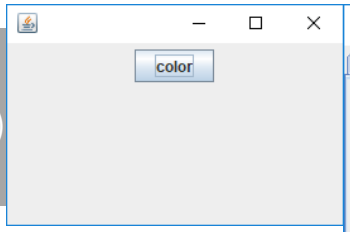

```
import javax.swing.*;import java.awt.event.*;
import java.io.*;
public class FileChooserExample
        extends JFrame implements ActionListener{
JMenuBar mb;  JMenu file;  JMenuItem open;  JTextArea ta;
FileChooserExample(){
open=new JMenuItem("Open File");open.addActionListener(this);

file=new JMenu("File");file.add(open);
mb=new JMenuBar(); mb.setBounds(0,0,800,20);mb.add(file);

ta=new JTextArea(800,800);ta.setBounds(0,20,800,800);
add(mb);   add(ta);
}
public void actionPerformed(ActionEvent e) {
if(e.getSource()==open){
   JFileChooser fc=new JFileChooser();
int i=fc.showOpenDialog(this);
   if(i==JFileChooser.APPROVE_OPTION){
      File f=fc.getSelectedFile();  String filepath=f.getPath();
      try{
BufferedReader br=new BufferedReader(new FileReader(filepath));
      String s1="", s2="";
      while((s1=br.readLine())!=null)
            s2+=s1+"\n";
      ta.setText(s2);
      br.close();
      }catch (Exception ex) {ex.printStackTrace();  }
   }  }  }
public static void main(String[] args) {
   FileChooserExample om=new FileChooserExample();
        om.setSize(500,500);    om.setVisible(true);
        om.setDefaultCloseOperation(EXIT_ON_CLOSE);
}  }
```

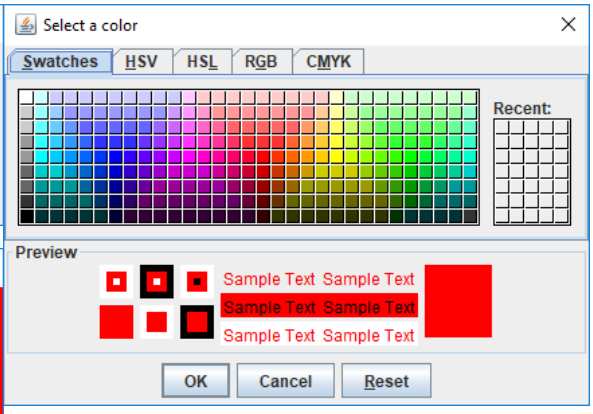
JColorChooser

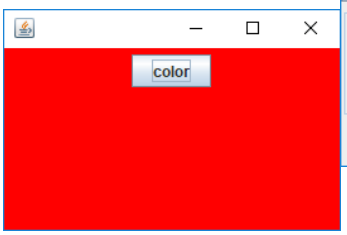
Constructor	Description
JColorChooser()	It is used to create a color chooser panel with white color initially.
JColorChooser(color initialcolor)	It is used to create a color chooser panel with the specified color initially.

Method	Description
void addChooserPanel(AbstractColorChooserPanel panel)	It is used to add a color chooser panel to the color chooser.
static Color showDialog(Component c, String title, Color initialColor)	It is used to show the color chooser dialog box.

JColorChooser (Cont.)



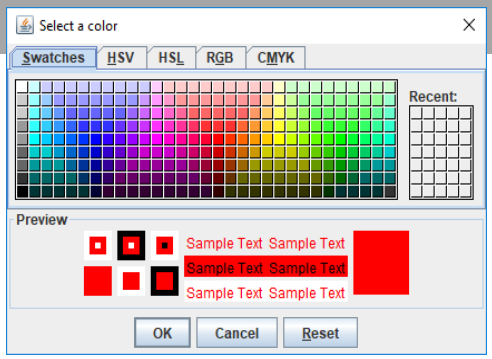


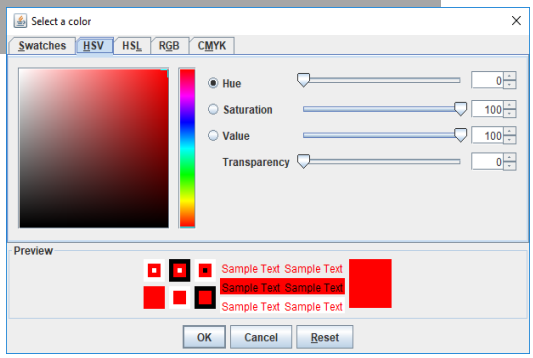


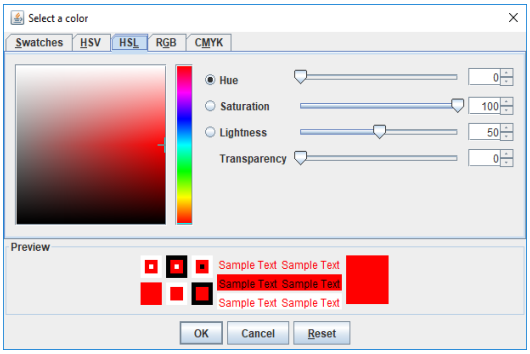

```
import java.awt.event.*;
import java.awt.*;
import javax.swing.*;
public class ColorChooserExample
extends JFrame implements ActionListener {
JButton b;    Container c;
ColorChooserExample(){
    c=getContentPane();
    c.setLayout(new FlowLayout());
    b=new JButton("color");
    b.addActionListener(this);
    c.add(b);
}
public void actionPerformed(ActionEvent e) {
    Color initialcolor=Color.RED;
    Color color=JColorChooser.showDialog(this,
                    "Select a color",initialcolor);
c.setBackground(color);
}
```

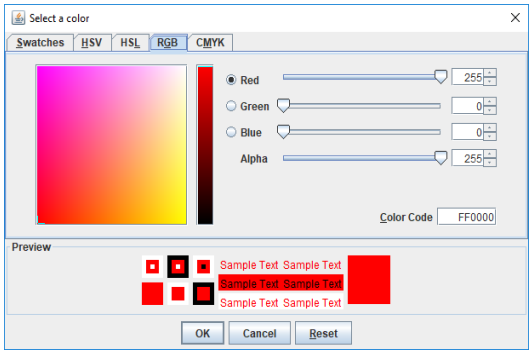
```
public static void main(String[] args) {
    ColorChooserExample ch=
                    new ColorChooserExample();
    ch.setSize(400,400);
    ch.setVisible(true);
    ch.setDefaultCloseOperation(EXIT_ON_CLOSE);
}
}
```

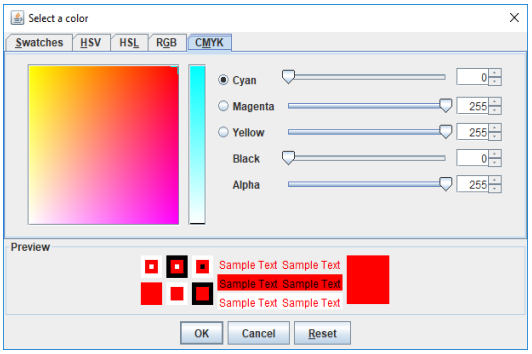
JColorChooser (Cont.)



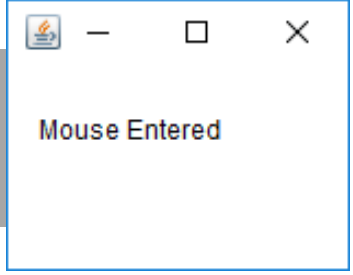








MouseListener




```
import java.awt.*;  import java.awt.event.*;
public class MouseListenerExample
extends Frame implements MouseListener{
    Label l;
    MouseListenerExample(){
        addMouseListener(this);
        l=new Label();
        l.setBounds(20,50,100,20);
        add(l);
        setSize(300,300);
        setVisible(true);
    }
    public void mouseClicked(MouseEvent e) {
        l.setText("Mouse Clicked");
}
    public void mouseEntered(MouseEvent e) {
        l.setText("Mouse Entered");
    }
    public void mouseExited(MouseEvent e) {
        l.setText("Mouse Exited");
    }
    public void mousePressed(MouseEvent e) {
        l.setText("Mouse Pressed");
    }
    public void mouseReleased(MouseEvent e) {
        l.setText("Mouse Released");
    }
public static void main(String[] args) {
    new MouseListenerExample();  }  }
```

MouseMotionListener

```
import java.awt.*;
import java.awt.event.*;
public class MouseMotionListenerExample
extends Frame implements MouseMotionListener{
    MouseMotionListenerExample(){
        addMouseMotionListener(this);

        setSize(300,300);
        setLayout(null);
        setVisible(true);
    }
public void mouseMoved(MouseEvent e) {
}
public void mouseDragged(MouseEvent e) {
    Graphics g=getGraphics();
    g.setColor(Color.BLUE);
    g.fillOval(e.getX(),e.getY(),20,20);
}
public static void main(String[] args) {
    new MouseMotionListenerExample();
}
}
```

References:

- http://java.sun.com/docs/books/tutorial/uiswing/events/index.html
- http://java.sun.com/docs/books/tutorial/uiswing/learn/example2.html#handlingEvents
- https://www.javatpoint.com

Questions/Comments