Hierarchy of Java Swing classes

The hierarchy of java swing API is given below.



Commonly used Methods of Component class

The methods of Component class are widely used in java swing that are given below.

|  |  |
| --- | --- |
| **Method** | **Description** |
| public void add(Component c) | add a component on another component. |
| public void setSize(int width,int height) | sets size of the component. |
| public void setLayout(LayoutManager m) | sets the layout manager for the component. |
| public void setVisible(boolean b) | sets the visibility of the component. It is by default false. |

Java Swing Examples

There are two ways to create a frame:

* By creating the object of Frame class (association)
* By extending Frame class (inheritance)

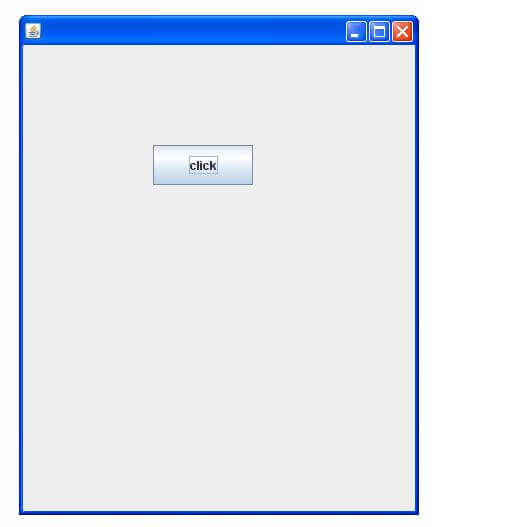
We can write the code of swing inside the main(), constructor or any other method.

Simple Java Swing Example

Let's see a simple swing example where we are creating one button and adding it on the JFrame object inside the main() method.

*File: FirstSwingExample.java*

1. **import** javax.swing.\*;
2. **public** **class** FirstSwingExample {
3. **public** **static** **void** main(String[] args) {
4. JFrame f=**new** JFrame();//creating instance of JFrame
6. JButton b=**new** JButton("click");//creating instance of JButton
7. b.setBounds(130,100,100, 40);//x axis, y axis, width, height
9. f.add(b);//adding button in JFrame
11. f.setSize(400,500);//400 width and 500 height
12. f.setLayout(**null**);//using no layout managers
13. f.setVisible(**true**);//making the frame visible
14. }
15. }



Example of Swing by Association inside constructor

We can also write all the codes of creating JFrame, JButton and method call inside the java constructor.

*File: Simple.java*

1. **import** javax.swing.\*;
2. **public** **class** Simple {
3. JFrame f;
4. Simple(){
5. f=**new** JFrame();//creating instance of JFrame
7. JButton b=**new** JButton("click");//creating instance of JButton
8. b.setBounds(130,100,100, 40);
10. f.add(b);//adding button in JFrame
12. f.setSize(400,500);//400 width and 500 height
13. f.setLayout(**null**);//using no layout managers
14. f.setVisible(**true**);//making the frame visible
15. }
17. **public** **static** **void** main(String[] args) {
18. **new** Simple();
19. }
20. }

The setBounds(int xaxis, int yaxis, int width, int height)is used in the above example that sets the position of the button.

Simple example of Swing by inheritance

We can also inherit the JFrame class, so there is no need to create the instance of JFrame class explicitly.

*File: Simple2.java*

1. **import** javax.swing.\*;
2. **public** **class** Simple2 **extends** JFrame{//inheriting JFrame
3. JFrame f;
4. Simple2(){
5. JButton b=**new** JButton("click");//create button
6. b.setBounds(130,100,100, 40);
8. add(b);//adding button on frame
9. setSize(400,500);
10. setLayout(**null**);
11. setVisible(**true**);
12. }
13. **public** **static** **void** main(String[] args) {
14. **new** Simple2();
15. }}

[download this example](https://static.javatpoint.com/src/swing/first2.zip)

*What we will learn in Swing Tutorial*

* JButton class
* JRadioButton class
* JTextArea class
* JComboBox class
* JTable class
* JColorChooser class
* JProgressBar class
* JSlider class
* Digital Watch
* Graphics in swing
* Displaying image
* Edit menu code for Notepad
* OpenDialog Box
* Notepad
* Puzzle Game
* Pic Puzzle Game
* Tic Tac Toe Game
* BorderLayout
* GridLayout
* FlowLayout
* CardLayout

# Java JButton

The JButton class is used to create a labeled button that has platform independent implementation. The application result in some action when the button is pushed. It inherits AbstractButton class.

## JButton class declaration

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

1. **public** **class** JButton **extends** AbstractButton **implements** Accessible

### Commonly used Constructors:

|  |  |
| --- | --- |
| **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. |

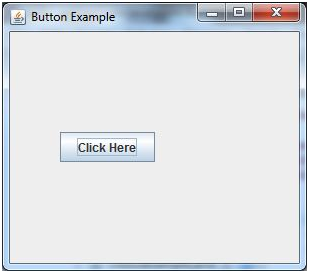
### Commonly used Methods of AbstractButton class:

|  |  |
| --- | --- |
| **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](https://www.javatpoint.com/java-actionlistener) to this object. |

## Java JButton Example

1. **import** javax.swing.\*;
2. **public** **class** ButtonExample {
3. **public** **static** **void** main(String[] args) {
4. JFrame f=**new** JFrame("Button Example");
5. JButton b=**new** JButton("Click Here");
6. b.setBounds(50,100,95,30);
7. f.add(b);
8. f.setSize(400,400);
9. f.setLayout(**null**);
10. f.setVisible(**true**);
11. }
12. }

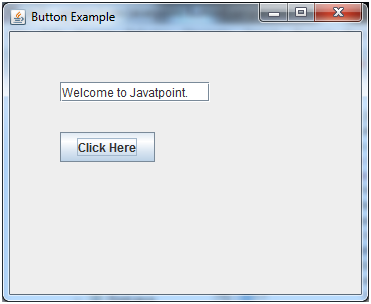
Output:



## Java JButton Example with ActionListener

1. **import** java.awt.event.\*;
2. **import** javax.swing.\*;
3. **public** **class** ButtonExample {
4. **public** **static** **void** main(String[] args) {
5. JFrame f=**new** JFrame("Button Example");
6. **final** JTextField tf=**new** JTextField();
7. tf.setBounds(50,50, 150,20);
8. JButton b=**new** JButton("Click Here");
9. b.setBounds(50,100,95,30);
10. b.addActionListener(**new** ActionListener(){
11. **public** **void** actionPerformed(ActionEvent e){
12. tf.setText("Welcome to Javatpoint.");
13. }
14. });
15. f.add(b);f.add(tf);
16. f.setSize(400,400);
17. f.setLayout(**null**);
18. f.setVisible(**true**);
19. }
20. }

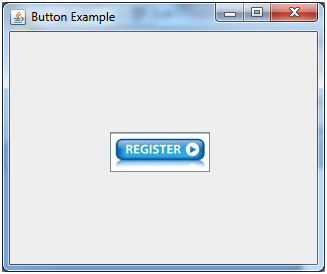
Output:



## Example of displaying image on the button:

1. **import** javax.swing.\*;
2. **public** **class** ButtonExample{
3. ButtonExample(){
4. JFrame f=**new** JFrame("Button Example");
5. JButton b=**new** JButton(**new** ImageIcon("D:\\icon.png"));
6. b.setBounds(100,100,100, 40);
7. f.add(b);
8. f.setSize(300,400);
9. f.setLayout(**null**);
10. f.setVisible(**true**);
11. f.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);
12. }
13. **public** **static** **void** main(String[] args) {
14. **new** ButtonExample();
15. }
16. }

Output:



# Java JLabel

The object of JLabel class is a component for placing text in a container. It is used to display a single line of read only text. The text can be changed by an application but a user cannot edit it directly. It inherits JComponent class.

## JLabel class declaration

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

1. **public** **class** JLabel **extends** JComponent **implements** SwingConstants, Accessible

### Commonly used Constructors:

|  |  |
| --- | --- |
| **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. |

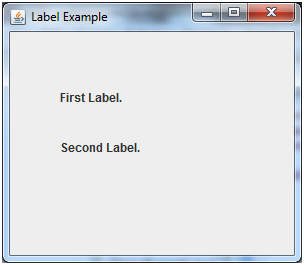
### Commonly used Methods:

|  |  |
| --- | --- |
| **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. |

## Java JLabel Example

1. **import** javax.swing.\*;
2. **class** LabelExample
3. {
4. **public** **static** **void** main(String args[])
5. {
6. JFrame f= **new** JFrame("Label Example");
7. JLabel l1,l2;
8. l1=**new** JLabel("First Label.");
9. l1.setBounds(50,50, 100,30);
10. l2=**new** JLabel("Second Label.");
11. l2.setBounds(50,100, 100,30);
12. f.add(l1); f.add(l2);
13. f.setSize(300,300);
14. f.setLayout(**null**);
15. f.setVisible(**true**);
16. }
17. }

Output:



## Java JLabel Example with ActionListener

1. **import** javax.swing.\*;
2. **import** java.awt.\*;
3. **import** java.awt.event.\*;
4. **public** **class** LabelExample **extends** Frame **implements** ActionListener{
5. JTextField tf; JLabel l; JButton b;
6. LabelExample(){
7. tf=**new** JTextField();
8. tf.setBounds(50,50, 150,20);
9. l=**new** JLabel();
10. l.setBounds(50,100, 250,20);
11. b=**new** JButton("Find IP");
12. b.setBounds(50,150,95,30);
13. b.addActionListener(**this**);
14. add(b);add(tf);add(l);
15. setSize(400,400);
16. setLayout(**null**);
17. setVisible(**true**);
18. }
19. **public** **void** actionPerformed(ActionEvent e) {
20. **try**{
21. String host=tf.getText();
22. String ip=java.net.InetAddress.getByName(host).getHostAddress();
23. l.setText("IP of "+host+" is: "+ip);
24. }**catch**(Exception ex){System.out.println(ex);}
25. }
26. **public** **static** **void** main(String[] args) {
27. **new** LabelExample();
28. } }

Output:

Play Video[](https://campaign.adpushup.com/get-started/?utm_source=banner&utm_campaign=growth_hack)



# Java JTextField

The object of a JTextField class is a text component that allows the editing of a single line text. It inherits JTextComponent class.

## JTextField class declaration

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

1. **public** **class** JTextField **extends** JTextComponent **implements** SwingConstants

### Commonly used Constructors:

|  |  |
| --- | --- |
| **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. |

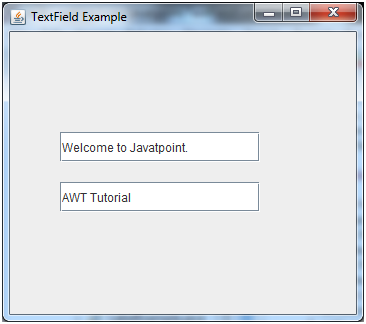
### Commonly used Methods:

|  |  |
| --- | --- |
| **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. |

## Java JTextField Example

1. **import** javax.swing.\*;
2. **class** TextFieldExample
3. {
4. **public** **static** **void** main(String args[])
5. {
6. JFrame f= **new** JFrame("TextField Example");
7. JTextField t1,t2;
8. t1=**new** JTextField("Welcome to Javatpoint.");
9. t1.setBounds(50,100, 200,30);
10. t2=**new** JTextField("AWT Tutorial");
11. t2.setBounds(50,150, 200,30);
12. f.add(t1); f.add(t2);
13. f.setSize(400,400);
14. f.setLayout(**null**);
15. f.setVisible(**true**);
16. }
17. }

Output:

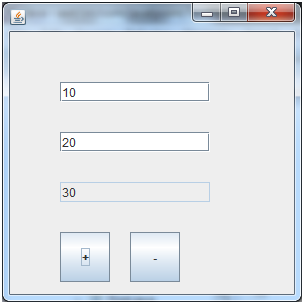


## Java JTextField Example with ActionListener

1. **import** javax.swing.\*;
2. **import** java.awt.event.\*;
3. **public** **class** TextFieldExample **implements** ActionListener{
4. JTextField tf1,tf2,tf3;
5. JButton b1,b2;
6. TextFieldExample(){
7. JFrame f= **new** JFrame();
8. tf1=**new** JTextField();
9. tf1.setBounds(50,50,150,20);
10. tf2=**new** JTextField();
11. tf2.setBounds(50,100,150,20);
12. tf3=**new** JTextField();
13. tf3.setBounds(50,150,150,20);
14. tf3.setEditable(**false**);
15. b1=**new** JButton("+");
16. b1.setBounds(50,200,50,50);
17. b2=**new** JButton("-");
18. b2.setBounds(120,200,50,50);
19. b1.addActionListener(**this**);
20. b2.addActionListener(**this**);
21. f.add(tf1);f.add(tf2);f.add(tf3);f.add(b1);f.add(b2);
22. f.setSize(300,300);
23. f.setLayout(**null**);
24. f.setVisible(**true**);
25. }
26. **public** **void** actionPerformed(ActionEvent e) {
27. String s1=tf1.getText();
28. String s2=tf2.getText();
29. **int** a=Integer.parseInt(s1);
30. **int** b=Integer.parseInt(s2);
31. **int** c=0;
32. **if**(e.getSource()==b1){
33. c=a+b;
34. }**else** **if**(e.getSource()==b2){
35. c=a-b;
36. }
37. String result=String.valueOf(c);
38. tf3.setText(result);
39. }
40. **public** **static** **void** main(String[] args) {
41. **new** TextFieldExample();
42. } }

Output:

Play Videox[](https://campaign.adpushup.com/get-started/?utm_source=banner&utm_campaign=growth_hack)



# Java JTextArea

The object of a JTextArea class is a multi line region that displays text. It allows the editing of multiple line text. It inherits JTextComponent class

## JTextArea class declaration

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

1. **public** **class** JTextArea **extends** JTextComponent

### Commonly used Constructors:

|  |  |
| --- | --- |
| **Constructor** | **Description** |
| JTextArea() | Creates a text area that displays no text initially. |
| JTextArea(String s) | Creates a text area that displays specified text initially. |
| JTextArea(int row, int column) | Creates a text area with the specified number of rows and columns that displays no text initially. |
| JTextArea(String s, int row, int column) | Creates a text area with the specified number of rows and columns that displays specified text. |

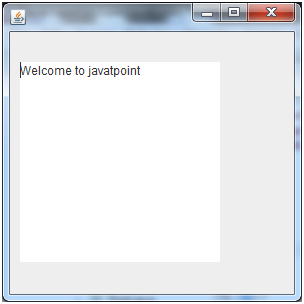
### Commonly used Methods:

|  |  |
| --- | --- |
| **Methods** | **Description** |
| void setRows(int rows) | It is used to set specified number of rows. |
| void setColumns(int cols) | It is used to set specified number of columns. |
| void setFont(Font f) | It is used to set the specified font. |
| void insert(String s, int position) | It is used to insert the specified text on the specified position. |
| void append(String s) | It is used to append the given text to the end of the document. |

## Java JTextArea Example

1. **import** javax.swing.\*;
2. **public** **class** TextAreaExample
3. {
4. TextAreaExample(){
5. JFrame f= **new** JFrame();
6. JTextArea area=**new** JTextArea("Welcome to javatpoint");
7. area.setBounds(10,30, 200,200);
8. f.add(area);
9. f.setSize(300,300);
10. f.setLayout(**null**);
11. f.setVisible(**true**);
12. }
13. **public** **static** **void** main(String args[])
14. {
15. **new** TextAreaExample();
16. }}

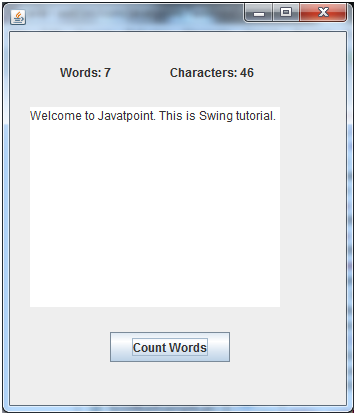
Output:



## Java JTextArea Example with ActionListener

1. **import** javax.swing.\*;
2. **import** java.awt.event.\*;
3. **public** **class** TextAreaExample **implements** ActionListener{
4. JLabel l1,l2;
5. JTextArea area;
6. JButton b;
7. TextAreaExample() {
8. JFrame f= **new** JFrame();
9. l1=**new** JLabel();
10. l1.setBounds(50,25,100,30);
11. l2=**new** JLabel();
12. l2.setBounds(160,25,100,30);
13. area=**new** JTextArea();
14. area.setBounds(20,75,250,200);
15. b=**new** JButton("Count Words");
16. b.setBounds(100,300,120,30);
17. b.addActionListener(**this**);
18. f.add(l1);f.add(l2);f.add(area);f.add(b);
19. f.setSize(450,450);
20. f.setLayout(**null**);
21. f.setVisible(**true**);
22. }
23. **public** **void** actionPerformed(ActionEvent e){
24. String text=area.getText();
25. String words[]=text.split("\\s");
26. l1.setText("Words: "+words.length);
27. l2.setText("Characters: "+text.length());
28. }
29. **public** **static** **void** main(String[] args) {
30. **new** TextAreaExample();
31. }
32. }

Output:



# Java JPasswordField

The object of a JPasswordField class is a text component specialized for password entry. It allows the editing of a single line of text. It inherits JTextField class.

## JPasswordField class declaration

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

1. **public** **class** JPasswordField **extends** JTextField

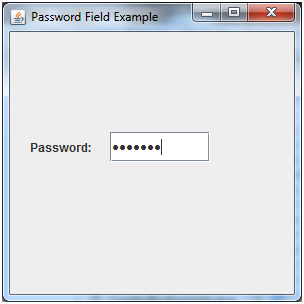
### Commonly used Constructors:

|  |  |
| --- | --- |
| **Constructor** | **Description** |
| JPasswordField() | Constructs a new JPasswordField, with a default document, null starting text string, and 0 column width. |
| JPasswordField(int columns) | Constructs a new empty JPasswordField with the specified number of columns. |
| JPasswordField(String text) | Constructs a new JPasswordField initialized with the specified text. |
| JPasswordField(String text, int columns) | Construct a new JPasswordField initialized with the specified text and columns. |

## Java JPasswordField Example

1. **import** javax.swing.\*;
2. **public** **class** PasswordFieldExample {
3. **public** **static** **void** main(String[] args) {
4. JFrame f=**new** JFrame("Password Field Example");
5. JPasswordField value = **new** JPasswordField();
6. JLabel l1=**new** JLabel("Password:");
7. l1.setBounds(20,100, 80,30);
8. value.setBounds(100,100,100,30);
9. f.add(value);  f.add(l1);
10. f.setSize(300,300);
11. f.setLayout(**null**);
12. f.setVisible(**true**);
13. }
14. }

Output:

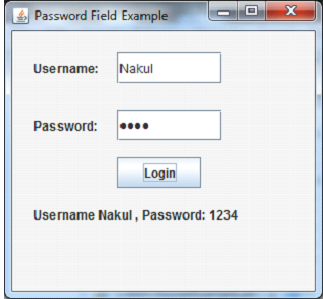


## Java JPasswordField Example with ActionListener

1. **import** javax.swing.\*;
2. **import** java.awt.event.\*;
3. **public** **class** PasswordFieldExample {
4. **public** **static** **void** main(String[] args) {
5. JFrame f=**new** JFrame("Password Field Example");
6. **final** JLabel label = **new** JLabel();
7. label.setBounds(20,150, 200,50);
8. **final** JPasswordField value = **new** JPasswordField();
9. value.setBounds(100,75,100,30);
10. JLabel l1=**new** JLabel("Username:");
11. l1.setBounds(20,20, 80,30);
12. JLabel l2=**new** JLabel("Password:");
13. l2.setBounds(20,75, 80,30);
14. JButton b = **new** JButton("Login");
15. b.setBounds(100,120, 80,30);
16. **final** JTextField text = **new** JTextField();
17. text.setBounds(100,20, 100,30);
18. f.add(value); f.add(l1); f.add(label); f.add(l2); f.add(b); f.add(text);
19. f.setSize(300,300);
20. f.setLayout(**null**);
21. f.setVisible(**true**);
22. b.addActionListener(**new** ActionListener() {
23. **public** **void** actionPerformed(ActionEvent e) {
24. String data = "Username " + text.getText();
25. data += ", Password: "
26. + **new** String(value.getPassword());
27. label.setText(data);
28. }
29. });
30. }
31. }

Output:

Play Videox[](https://campaign.adpushup.com/get-started/?utm_source=banner&utm_campaign=growth_hack)



# Java JCheckBox

The JCheckBox class is used to create a checkbox. It is used to turn an option on (true) or off (false). Clicking on a CheckBox changes its state from "on" to "off" or from "off" to "on ".It inherits [JToggleButton](https://www.javatpoint.com/java-jtogglebutton) class.

## JCheckBox class declaration

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

1. **public** **class** JCheckBox **extends** JToggleButton **implements** Accessible

### Commonly used Constructors:

|  |  |
| --- | --- |
| **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. |

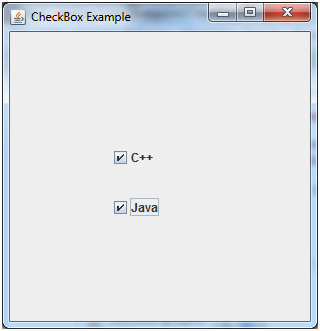
### Commonly used Methods:

|  |  |
| --- | --- |
| **Methods** | **Description** |
| AccessibleContext getAccessibleContext() | It is used to get the AccessibleContext associated with this JCheckBox. |
| protected String paramString() | It returns a [string](https://www.javatpoint.com/java-string) representation of this JCheckBox. |

## Java JCheckBox Example

1. **import** javax.swing.\*;
2. **public** **class** CheckBoxExample
3. {
4. CheckBoxExample(){
5. JFrame f= **new** JFrame("CheckBox Example");
6. JCheckBox checkBox1 = **new** JCheckBox("C++");
7. checkBox1.setBounds(100,100, 50,50);
8. JCheckBox checkBox2 = **new** JCheckBox("Java", **true**);
9. checkBox2.setBounds(100,150, 50,50);
10. f.add(checkBox1);
11. f.add(checkBox2);
12. f.setSize(400,400);
13. f.setLayout(**null**);
14. f.setVisible(**true**);
15. }
16. **public** **static** **void** main(String args[])
17. {
18. **new** CheckBoxExample();
19. }}

Output:

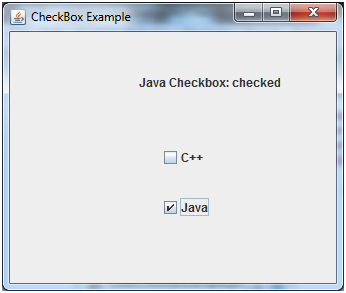


## Java JCheckBox Example with ItemListener

1. **import** javax.swing.\*;
2. **import** java.awt.event.\*;
3. **public** **class** CheckBoxExample
4. {
5. CheckBoxExample(){
6. JFrame f= **new** JFrame("CheckBox Example");
7. **final** JLabel label = **new** JLabel();
8. label.setHorizontalAlignment(JLabel.CENTER);
9. label.setSize(400,100);
10. JCheckBox checkbox1 = **new** JCheckBox("C++");
11. checkbox1.setBounds(150,100, 50,50);
12. JCheckBox checkbox2 = **new** JCheckBox("Java");
13. checkbox2.setBounds(150,150, 50,50);
14. f.add(checkbox1); f.add(checkbox2); f.add(label);
15. checkbox1.addItemListener(**new** ItemListener() {
16. **public** **void** itemStateChanged(ItemEvent e) {
17. label.setText("C++ Checkbox: "
18. + (e.getStateChange()==1?"checked":"unchecked"));
19. }
20. });
21. checkbox2.addItemListener(**new** ItemListener() {
22. **public** **void** itemStateChanged(ItemEvent e) {
23. label.setText("Java Checkbox: "
24. + (e.getStateChange()==1?"checked":"unchecked"));
25. }
26. });
27. f.setSize(400,400);
28. f.setLayout(**null**);
29. f.setVisible(**true**);
30. }
31. **public** **static** **void** main(String args[])
32. {
33. **new** CheckBoxExample();
34. }
35. }

Output:

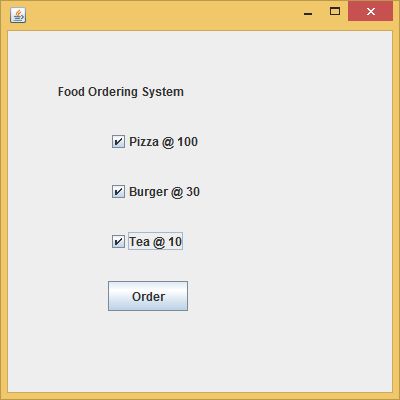
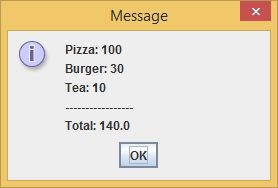
Play Videox[](https://campaign.adpushup.com/get-started/?utm_source=banner&utm_campaign=growth_hack)



## Java JCheckBox Example: Food Order

1. **import** javax.swing.\*;
2. **import** java.awt.event.\*;
3. **public** **class** CheckBoxExample **extends** JFrame **implements** ActionListener{
4. JLabel l;
5. JCheckBox cb1,cb2,cb3;
6. JButton b;
7. CheckBoxExample(){
8. l=**new** JLabel("Food Ordering System");
9. l.setBounds(50,50,300,20);
10. cb1=**new** JCheckBox("Pizza @ 100");
11. cb1.setBounds(100,100,150,20);
12. cb2=**new** JCheckBox("Burger @ 30");
13. cb2.setBounds(100,150,150,20);
14. cb3=**new** JCheckBox("Tea @ 10");
15. cb3.setBounds(100,200,150,20);
16. b=**new** JButton("Order");
17. b.setBounds(100,250,80,30);
18. b.addActionListener(**this**);
19. add(l);add(cb1);add(cb2);add(cb3);add(b);
20. setSize(400,400);
21. setLayout(**null**);
22. setVisible(**true**);
23. setDefaultCloseOperation(EXIT\_ON\_CLOSE);
24. }
25. **public** **void** actionPerformed(ActionEvent e){
26. **float** amount=0;
27. String msg="";
28. **if**(cb1.isSelected()){
29. amount+=100;
30. msg="Pizza: 100\n";
31. }
32. **if**(cb2.isSelected()){
33. amount+=30;
34. msg+="Burger: 30\n";
35. }
36. **if**(cb3.isSelected()){
37. amount+=10;
38. msg+="Tea: 10\n";
39. }
40. msg+="-----------------\n";
41. JOptionPane.showMessageDialog(**this**,msg+"Total: "+amount);
42. }
43. **public** **static** **void** main(String[] args) {
44. **new** CheckBoxExample();
45. }
46. }

Output:

# Java JRadioButton

The JRadioButton class is used to create a radio button. It is used to choose one option from multiple options. It is widely used in exam systems or quiz.

It should be added in ButtonGroup to select one radio button only.

## JRadioButton class declaration

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

1. **public** **class** JRadioButton **extends** JToggleButton **implements** Accessible

### Commonly used Constructors:

|  |  |
| --- | --- |
| **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. |

### Commonly used Methods:

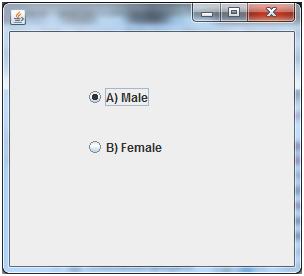
|  |  |
| --- | --- |
| **Methods** | **Description** |
| void setText(String s) | It is used to set specified text on button. |
| String getText() | It is used to return the text of the button. |
| void setEnabled(boolean b) | It is used to enable or disable the button. |
| void setIcon(Icon b) | It 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. |

## Java JRadioButton Example

1. **import** javax.swing.\*;
2. **public** **class** RadioButtonExample {
3. JFrame f;
4. RadioButtonExample(){
5. f=**new** JFrame();
6. JRadioButton r1=**new** JRadioButton("A) Male");
7. JRadioButton r2=**new** JRadioButton("B) Female");
8. r1.setBounds(75,50,100,30);
9. r2.setBounds(75,100,100,30);
10. ButtonGroup bg=**new** ButtonGroup();
11. bg.add(r1);bg.add(r2);
12. f.add(r1);f.add(r2);
13. f.setSize(300,300);
14. f.setLayout(**null**);
15. f.setVisible(**true**);
16. }
17. **public** **static** **void** main(String[] args) {
18. **new** RadioButtonExample();
19. }
20. }

Output:

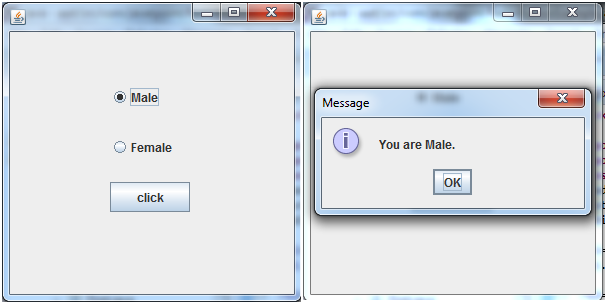
Play Videox[](https://campaign.adpushup.com/get-started/?utm_source=banner&utm_campaign=growth_hack)



## Java JRadioButton Example with ActionListener

1. **import** javax.swing.\*;
2. **import** java.awt.event.\*;
3. **class** RadioButtonExample **extends** JFrame **implements** ActionListener{
4. JRadioButton rb1,rb2;
5. JButton b;
6. RadioButtonExample(){
7. rb1=**new** JRadioButton("Male");
8. rb1.setBounds(100,50,100,30);
9. rb2=**new** JRadioButton("Female");
10. rb2.setBounds(100,100,100,30);
11. ButtonGroup bg=**new** ButtonGroup();
12. bg.add(rb1);bg.add(rb2);
13. b=**new** JButton("click");
14. b.setBounds(100,150,80,30);
15. b.addActionListener(**this**);
16. add(rb1);add(rb2);add(b);
17. setSize(300,300);
18. setLayout(**null**);
19. setVisible(**true**);
20. }
21. **public** **void** actionPerformed(ActionEvent e){
22. **if**(rb1.isSelected()){
23. JOptionPane.showMessageDialog(**this**,"You are Male.");
24. }
25. **if**(rb2.isSelected()){
26. JOptionPane.showMessageDialog(**this**,"You are Female.");
27. }
28. }
29. **public** **static** **void** main(String args[]){
30. **new** RadioButtonExample();
31. }}

Output:



# Java JComboBox

The object of Choice class is used to show popup menu of choices. Choice selected by user is shown on the top of a [menu](https://www.javatpoint.com/java-jmenuitem-and-jmenu). It inherits [JComponent](https://www.javatpoint.com/java-jcomponent) class.

## JComboBox class declaration

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

1. **public** **class** JComboBox **extends** JComponent **implements** ItemSelectable, ListDataListener, ActionListener, Accessible

### Commonly used Constructors:

|  |  |
| --- | --- |
| **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](https://www.javatpoint.com/array-in-java). |
| JComboBox(Vector<?> items) | Creates a JComboBox that contains the elements in the specified [Vector](https://www.javatpoint.com/scala-vector). |

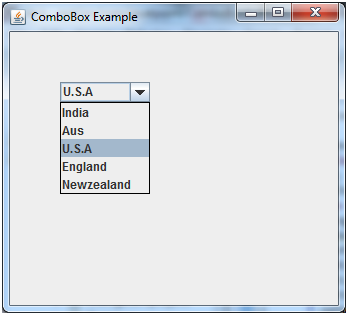
### Commonly used Methods:

|  |  |
| --- | --- |
| **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](https://www.javatpoint.com/java-actionlistener). |
| void addItemListener(ItemListener i) | It is used to add the [ItemListener](https://www.javatpoint.com/java-itemlistener). |

## Java JComboBox Example

1. **import** javax.swing.\*;
2. **public** **class** ComboBoxExample {
3. JFrame f;
4. ComboBoxExample(){
5. f=**new** JFrame("ComboBox Example");
6. String country[]={"India","Aus","U.S.A","England","Newzealand"};
7. JComboBox cb=**new** JComboBox(country);
8. cb.setBounds(50, 50,90,20);
9. f.add(cb);
10. f.setLayout(**null**);
11. f.setSize(400,500);
12. f.setVisible(**true**);
13. }
14. **public** **static** **void** main(String[] args) {
15. **new** ComboBoxExample();
16. }
17. }

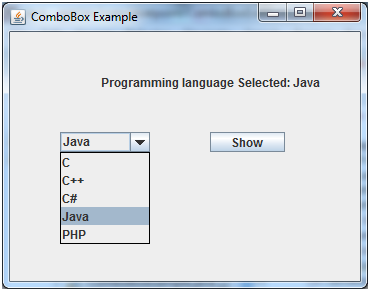
Output:



## Java JComboBox Example with ActionListener

1. **import** javax.swing.\*;
2. **import** java.awt.event.\*;
3. **public** **class** ComboBoxExample {
4. JFrame f;
5. ComboBoxExample(){
6. f=**new** JFrame("ComboBox Example");
7. **final** JLabel label = **new** JLabel();
8. label.setHorizontalAlignment(JLabel.CENTER);
9. label.setSize(400,100);
10. JButton b=**new** JButton("Show");
11. b.setBounds(200,100,75,20);
12. String languages[]={"C","C++","C#","Java","PHP"};
13. **final** JComboBox cb=**new** JComboBox(languages);
14. cb.setBounds(50, 100,90,20);
15. f.add(cb); f.add(label); f.add(b);
16. f.setLayout(**null**);
17. f.setSize(350,350);
18. f.setVisible(**true**);
19. b.addActionListener(**new** ActionListener() {
20. **public** **void** actionPerformed(ActionEvent e) {
21. String data = "Programming language Selected: "
22. + cb.getItemAt(cb.getSelectedIndex());
23. label.setText(data);
24. }
25. });
26. }
27. **public** **static** **void** main(String[] args) {
28. **new** ComboBoxExample();
29. }
30. }

Output:



# Java JTable

The JTable class is used to display data in tabular form. It is composed of rows and columns.

## JTable class declaration

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

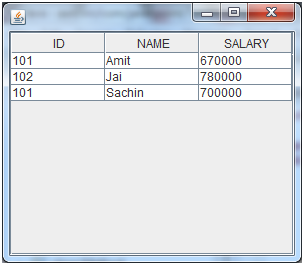
### Commonly used Constructors:

|  |  |
| --- | --- |
| **Constructor** | **Description** |
| JTable() | Creates a table with empty cells. |
| JTable(Object[][] rows, Object[] columns) | Creates a table with the specified data. |

## Java JTable Example

1. **import** javax.swing.\*;
2. **public** **class** TableExample {
3. JFrame f;
4. TableExample(){
5. f=**new** JFrame();
6. String data[][]={ {"101","Amit","670000"},
7. {"102","Jai","780000"},
8. {"101","Sachin","700000"}};
9. String column[]={"ID","NAME","SALARY"};
10. JTable jt=**new** JTable(data,column);
11. jt.setBounds(30,40,200,300);
12. JScrollPane sp=**new** JScrollPane(jt);
13. f.add(sp);
14. f.setSize(300,400);
15. f.setVisible(**true**);
16. }
17. **public** **static** **void** main(String[] args) {
18. **new** TableExample();
19. }
20. }

Output:

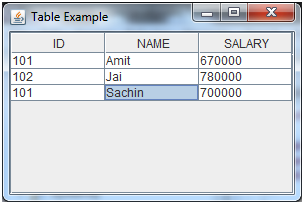


## Java JTable Example with ListSelectionListener

1. **import** javax.swing.\*;
2. **import** javax.swing.event.\*;
3. **public** **class** TableExample {
4. **public** **static** **void** main(String[] a) {
5. JFrame f = **new** JFrame("Table Example");
6. String data[][]={ {"101","Amit","670000"},
7. {"102","Jai","780000"},
8. {"101","Sachin","700000"}};
9. String column[]={"ID","NAME","SALARY"};
10. **final** JTable jt=**new** JTable(data,column);
11. jt.setCellSelectionEnabled(**true**);
12. ListSelectionModel select= jt.getSelectionModel();
13. select.setSelectionMode(ListSelectionModel.SINGLE\_SELECTION);
14. select.addListSelectionListener(**new** ListSelectionListener() {
15. **public** **void** valueChanged(ListSelectionEvent e) {
16. String Data = **null**;
17. **int**[] row = jt.getSelectedRows();
18. **int**[] columns = jt.getSelectedColumns();
19. **for** (**int** i = 0; i < row.length; i++) {
20. **for** (**int** j = 0; j < columns.length; j++) {
21. Data = (String) jt.getValueAt(row[i], columns[j]);
22. } }
23. System.out.println("Table element selected is: " + Data);
24. }
25. });
26. JScrollPane sp=**new** JScrollPane(jt);
27. f.add(sp);
28. f.setSize(300, 200);
29. f.setVisible(**true**);
30. }
31. }

Output:

Play Videox[](https://campaign.adpushup.com/get-started/?utm_source=banner&utm_campaign=growth_hack)



If you select an element in column **NAME**, name of the element will be displayed on the console:

1. Table element selected is: Sachin

# Java JList

The object of JList class represents a list of text items. The list of text items can be set up so that the user can choose either one item or multiple items. It inherits JComponent class.

## JList class declaration

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

1. **public** **class** JList **extends** JComponent **implements** Scrollable, Accessible

### Commonly used Constructors:

|  |  |
| --- | --- |
| **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. |

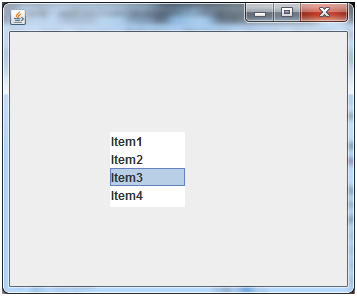
### Commonly used Methods:

|  |  |
| --- | --- |
| **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. |

## Java JList Example

1. **import** javax.swing.\*;
2. **public** **class** ListExample
3. {
4. ListExample(){
5. JFrame f= **new** JFrame();
6. DefaultListModel<String> l1 = **new** DefaultListModel<>();
7. l1.addElement("Item1");
8. l1.addElement("Item2");
9. l1.addElement("Item3");
10. l1.addElement("Item4");
11. JList<String> list = **new** JList<>(l1);
12. list.setBounds(100,100, 75,75);
13. f.add(list);
14. f.setSize(400,400);
15. f.setLayout(**null**);
16. f.setVisible(**true**);
17. }
18. **public** **static** **void** main(String args[])
19. {
20. **new** ListExample();
21. }}

Output:

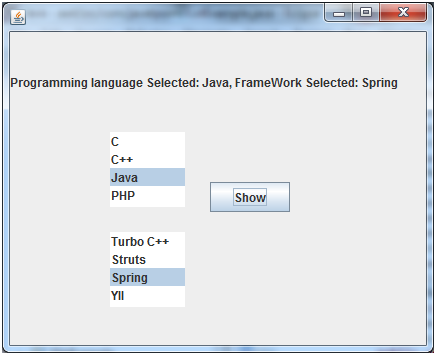


### Java JList Example with ActionListener

1. **import** javax.swing.\*;
2. **import** java.awt.event.\*;
3. **public** **class** ListExample
4. {
5. ListExample(){
6. JFrame f= **new** JFrame();
7. **final** JLabel label = **new** JLabel();
8. label.setSize(500,100);
9. JButton b=**new** JButton("Show");
10. b.setBounds(200,150,80,30);
11. **final** DefaultListModel<String> l1 = **new** DefaultListModel<>();
12. l1.addElement("C");
13. l1.addElement("C++");
14. l1.addElement("Java");
15. l1.addElement("PHP");
16. **final** JList<String> list1 = **new** JList<>(l1);
17. list1.setBounds(100,100, 75,75);
18. DefaultListModel<String> l2 = **new** DefaultListModel<>();
19. l2.addElement("Turbo C++");
20. l2.addElement("Struts");
21. l2.addElement("Spring");
22. l2.addElement("YII");
23. **final** JList<String> list2 = **new** JList<>(l2);
24. list2.setBounds(100,200, 75,75);
25. f.add(list1); f.add(list2); f.add(b); f.add(label);
26. f.setSize(450,450);
27. f.setLayout(**null**);
28. f.setVisible(**true**);
29. b.addActionListener(**new** ActionListener() {
30. **public** **void** actionPerformed(ActionEvent e) {
31. String data = "";
32. **if** (list1.getSelectedIndex() != -1) {
33. data = "Programming language Selected: " + list1.getSelectedValue();
34. label.setText(data);
35. }
36. **if**(list2.getSelectedIndex() != -1){
37. data += ", FrameWork Selected: ";
38. **for**(Object frame :list2.getSelectedValues()){
39. data += frame + " ";
40. }
41. }
42. label.setText(data);
43. }
44. });
45. }
46. **public** **static** **void** main(String args[])
47. {
48. **new** ListExample();
49. }}

Output:

Play Videox[](https://campaign.adpushup.com/get-started/?utm_source=banner&utm_campaign=growth_hack)



# Java JOptionPane

The JOptionPane class is used to provide standard dialog boxes such as message dialog box, confirm dialog box and input dialog box. These dialog boxes are used to display information or get input from the user. The JOptionPane class inherits JComponent class.

## JOptionPane class declaration

1. **public** **class** JOptionPane **extends** JComponent **implements** Accessible

### Common Constructors of JOptionPane class

|  |  |
| --- | --- |
| **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. |

### Common Methods of JOptionPane class

|  |  |
| --- | --- |
| **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. |

## Java JOptionPane Example: showMessageDialog()

1. **import** javax.swing.\*;
2. **public** **class** OptionPaneExample {
3. JFrame f;
4. OptionPaneExample(){
5. f=**new** JFrame();
6. JOptionPane.showMessageDialog(f,"Hello, Welcome to Javatpoint.");
7. }
8. **public** **static** **void** main(String[] args) {
9. **new** OptionPaneExample();
10. }
11. }

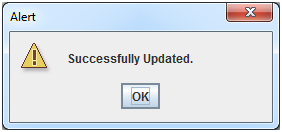
Output:



## Java JOptionPane Example: showMessageDialog()

1. **import** javax.swing.\*;
2. **public** **class** OptionPaneExample {
3. JFrame f;
4. OptionPaneExample(){
5. f=**new** JFrame();
6. JOptionPane.showMessageDialog(f,"Successfully Updated.","Alert",JOptionPane.WARNING\_MESSAGE);
7. }
8. **public** **static** **void** main(String[] args) {
9. **new** OptionPaneExample();
10. }
11. }

Output:

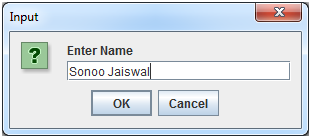


## Java JOptionPane Example: showInputDialog()

1. **import** javax.swing.\*;
2. **public** **class** OptionPaneExample {
3. JFrame f;
4. OptionPaneExample(){
5. f=**new** JFrame();
6. String name=JOptionPane.showInputDialog(f,"Enter Name");
7. }
8. **public** **static** **void** main(String[] args) {
9. **new** OptionPaneExample();
10. }
11. }

Output:

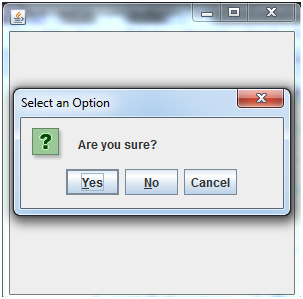
Play Videox[](https://campaign.adpushup.com/get-started/?utm_source=banner&utm_campaign=growth_hack)



## Java JOptionPane Example: showConfirmDialog()

1. **import** javax.swing.\*;
2. **import** java.awt.event.\*;
3. **public** **class** OptionPaneExample **extends** WindowAdapter{
4. JFrame f;
5. OptionPaneExample(){
6. f=**new** JFrame();
7. f.addWindowListener(**this**);
8. f.setSize(300, 300);
9. f.setLayout(**null**);
10. f.setDefaultCloseOperation(JFrame.DO\_NOTHING\_ON\_CLOSE);
11. f.setVisible(**true**);
12. }
13. **public** **void** windowClosing(WindowEvent e) {
14. **int** a=JOptionPane.showConfirmDialog(f,"Are you sure?");
15. **if**(a==JOptionPane.YES\_OPTION){
16. f.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);
17. }
18. }
19. **public** **static** **void** main(String[] args) {
20. **new**  OptionPaneExample();
21. }
22. }

Output:



# Java JScrollBar

The object of JScrollbar class is used to add horizontal and vertical scrollbar. It is an implementation of a scrollbar. It inherits JComponent class.

## JScrollBar class declaration

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

1. **public** **class** JScrollBar **extends** JComponent **implements** Adjustable, Accessible

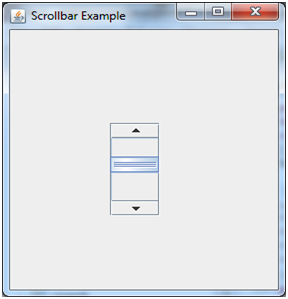
### Commonly used Constructors:

|  |  |
| --- | --- |
| **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. |

## Java JScrollBar Example

1. **import** javax.swing.\*;
2. **class** ScrollBarExample
3. {
4. ScrollBarExample(){
5. JFrame f= **new** JFrame("Scrollbar Example");
6. JScrollBar s=**new** JScrollBar();
7. s.setBounds(100,100, 50,100);
8. f.add(s);
9. f.setSize(400,400);
10. f.setLayout(**null**);
11. f.setVisible(**true**);
12. }
13. **public** **static** **void** main(String args[])
14. {
15. **new** ScrollBarExample();
16. }}

Output:

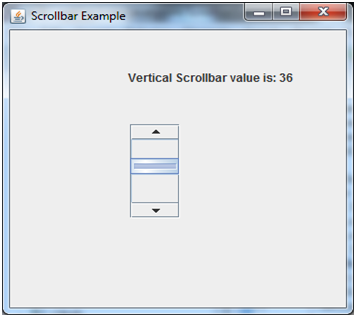


## Java JScrollBar Example with AdjustmentListener

1. **import** javax.swing.\*;
2. **import** java.awt.event.\*;
3. **class** ScrollBarExample
4. {
5. ScrollBarExample(){
6. JFrame f= **new** JFrame("Scrollbar Example");
7. **final** JLabel label = **new** JLabel();
8. label.setHorizontalAlignment(JLabel.CENTER);
9. label.setSize(400,100);
10. **final** JScrollBar s=**new** JScrollBar();
11. s.setBounds(100,100, 50,100);
12. f.add(s); f.add(label);
13. f.setSize(400,400);
14. f.setLayout(**null**);
15. f.setVisible(**true**);
16. s.addAdjustmentListener(**new** AdjustmentListener() {
17. **public** **void** adjustmentValueChanged(AdjustmentEvent e) {
18. label.setText("Vertical Scrollbar value is:"+ s.getValue());
19. }
20. });
21. }
22. **public** **static** **void** main(String args[])
23. {
24. **new** ScrollBarExample();
25. }}

Output:

Play Videox[](https://campaign.adpushup.com/get-started/?utm_source=banner&utm_campaign=growth_hack)



# Java JMenuBar, JMenu and JMenuItem

The JMenuBar class is used to display menubar on the window or frame. It may have several menus.

The object of JMenu class is a pull down menu component which is displayed from the menu bar. It inherits the JMenuItem class.

The object of JMenuItem class adds a simple labeled menu item. The items used in a menu must belong to the JMenuItem or any of its subclass.

### JMenuBar class declaration

1. **public** **class** JMenuBar **extends** JComponent **implements** MenuElement, Accessible

### JMenu class declaration

1. **public** **class** JMenu **extends** JMenuItem **implements** MenuElement, Accessible

### JMenuItem class declaration

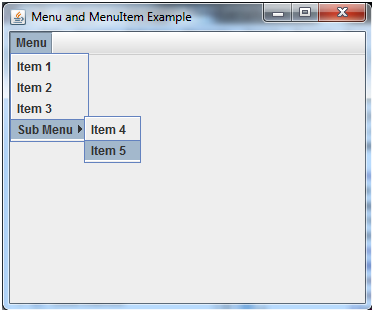
1. **public** **class** JMenuItem **extends** AbstractButton **implements** Accessible, MenuElement

## Java JMenuItem and JMenu Example

1. **import** javax.swing.\*;
2. **class** MenuExample
3. {
4. JMenu menu, submenu;
5. JMenuItem i1, i2, i3, i4, i5;
6. MenuExample(){
7. JFrame f= **new** JFrame("Menu and MenuItem Example");
8. JMenuBar mb=**new** JMenuBar();
9. menu=**new** JMenu("Menu");
10. submenu=**new** JMenu("Sub Menu");
11. i1=**new** JMenuItem("Item 1");
12. i2=**new** JMenuItem("Item 2");
13. i3=**new** JMenuItem("Item 3");
14. i4=**new** JMenuItem("Item 4");
15. i5=**new** JMenuItem("Item 5");
16. menu.add(i1); menu.add(i2); menu.add(i3);
17. submenu.add(i4); submenu.add(i5);
18. menu.add(submenu);
19. mb.add(menu);
20. f.setJMenuBar(mb);
21. f.setSize(400,400);
22. f.setLayout(**null**);
23. f.setVisible(**true**);
24. }
25. **public** **static** **void** main(String args[])
26. {
27. **new** MenuExample();
28. }}

Output:

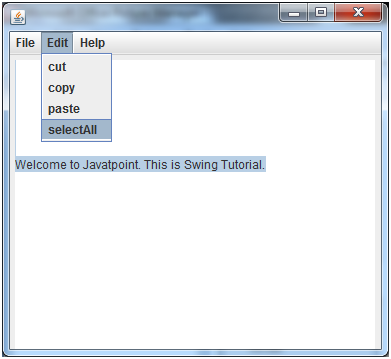
Play Videox[](https://campaign.adpushup.com/get-started/?utm_source=banner&utm_campaign=growth_hack)



## Example of creating Edit menu for Notepad:

1. **import** javax.swing.\*;
2. **import** java.awt.event.\*;
3. **public** **class** MenuExample **implements** ActionListener{
4. JFrame f;
5. JMenuBar mb;
6. JMenu file,edit,help;
7. JMenuItem cut,copy,paste,selectAll;
8. JTextArea ta;
9. MenuExample(){
10. f=**new** JFrame();
11. cut=**new** JMenuItem("cut");
12. copy=**new** JMenuItem("copy");
13. paste=**new** JMenuItem("paste");
14. selectAll=**new** JMenuItem("selectAll");
15. cut.addActionListener(**this**);
16. copy.addActionListener(**this**);
17. paste.addActionListener(**this**);
18. selectAll.addActionListener(**this**);
19. mb=**new** JMenuBar();
20. file=**new** JMenu("File");
21. edit=**new** JMenu("Edit");
22. help=**new** JMenu("Help");
23. edit.add(cut);edit.add(copy);edit.add(paste);edit.add(selectAll);
24. mb.add(file);mb.add(edit);mb.add(help);
25. ta=**new** JTextArea();
26. ta.setBounds(5,5,360,320);
27. f.add(mb);f.add(ta);
28. f.setJMenuBar(mb);
29. f.setLayout(**null**);
30. f.setSize(400,400);
31. f.setVisible(**true**);
32. }
33. **public** **void** actionPerformed(ActionEvent e) {
34. **if**(e.getSource()==cut)
35. ta.cut();
36. **if**(e.getSource()==paste)
37. ta.paste();
38. **if**(e.getSource()==copy)
39. ta.copy();
40. **if**(e.getSource()==selectAll)
41. ta.selectAll();
42. }
43. **public** **static** **void** main(String[] args) {
44. **new** MenuExample();
45. }
46. }

Output:



# 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.

1. **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

1. **import** javax.swing.\*;
2. **import** java.awt.event.\*;
3. **class** PopupMenuExample
4. {
5. PopupMenuExample(){
6. **final** JFrame f= **new** JFrame("PopupMenu Example");
7. **final** JPopupMenu popupmenu = **new** JPopupMenu("Edit");
8. JMenuItem cut = **new** JMenuItem("Cut");
9. JMenuItem copy = **new** JMenuItem("Copy");
10. JMenuItem paste = **new** JMenuItem("Paste");
11. popupmenu.add(cut); popupmenu.add(copy); popupmenu.add(paste);
12. f.addMouseListener(**new** MouseAdapter() {
13. **public** **void** mouseClicked(MouseEvent e) {
14. popupmenu.show(f , e.getX(), e.getY());
15. }
16. });
17. f.add(popupmenu);
18. f.setSize(300,300);
19. f.setLayout(**null**);
20. f.setVisible(**true**);
21. }
22. **public** **static** **void** main(String args[])
23. {
24. **new** PopupMenuExample();
25. }}

Output:

## Java JPopupMenu Example with MouseListener and ActionListener

1. **import** javax.swing.\*;
2. **import** java.awt.event.\*;
3. **class** PopupMenuExample
4. {
5. PopupMenuExample(){
6. **final** JFrame f= **new** JFrame("PopupMenu Example");
7. **final** JLabel label = **new** JLabel();
8. label.setHorizontalAlignment(JLabel.CENTER);
9. label.setSize(400,100);
10. **final** JPopupMenu popupmenu = **new** JPopupMenu("Edit");
11. JMenuItem cut = **new** JMenuItem("Cut");
12. JMenuItem copy = **new** JMenuItem("Copy");
13. JMenuItem paste = **new** JMenuItem("Paste");
14. popupmenu.add(cut); popupmenu.add(copy); popupmenu.add(paste);
15. f.addMouseListener(**new** MouseAdapter() {
16. **public** **void** mouseClicked(MouseEvent e) {
17. popupmenu.show(f , e.getX(), e.getY());
18. }
19. });
20. cut.addActionListener(**new** ActionListener(){
21. **public** **void** actionPerformed(ActionEvent e) {
22. label.setText("cut MenuItem clicked.");
23. }
24. });
25. copy.addActionListener(**new** ActionListener(){
26. **public** **void** actionPerformed(ActionEvent e) {
27. label.setText("copy MenuItem clicked.");
28. }
29. });
30. paste.addActionListener(**new** ActionListener(){
31. **public** **void** actionPerformed(ActionEvent e) {
32. label.setText("paste MenuItem clicked.");
33. }
34. });
35. f.add(label); f.add(popupmenu);
36. f.setSize(400,400);
37. f.setLayout(**null**);
38. f.setVisible(**true**);
39. }
40. **public** **static** **void** main(String args[])
41. {
42. **new** PopupMenuExample();
43. }
44. }

Output:

Play Videox[](https://campaign.adpushup.com/get-started/?utm_source=banner&utm_campaign=growth_hack)

# Java JCheckBoxMenuItem

JCheckBoxMenuItem class represents [checkbox](https://www.javatpoint.com/java-jcheckbox) which can be included on a [menu](https://www.javatpoint.com/java-jmenuitem-and-jmenu) . A CheckBoxMenuItem can have text or a graphic icon or both, associated with it. [MenuItem](https://www.javatpoint.com/java-jmenuitem-and-jmenu) can be selected or deselected. MenuItems can be configured and controlled by actions.

### Nested class

|  |  |  |
| --- | --- | --- |
| **Modifier and Type** | **Class** | **Description** |
| protected class | JCheckBoxMenuItem.AccessibleJCheckBoxMenuItem | This [class](https://www.javatpoint.com/object-and-class-in-java) implements accessibility support for the JcheckBoxMenuItem class. |

### Constructor

|  |  |
| --- | --- |
| **Constructor** | **Description** |
| JCheckBoxMenuItem() | It creates an initially unselected check box menu item with no set text or icon. |
| JCheckBoxMenuItem(Action a) | It creates a menu item whose properties are taken from the Action supplied. |
| JCheckBoxMenuItem(Icon icon) | It creates an initially unselected check box menu item with an icon. |
| JCheckBoxMenuItem(String text) | It creates an initially unselected check box menu item with text. |
| JCheckBoxMenuItem(String text, boolean b) | It creates a check box menu item with the specified text and selection state. |
| JCheckBoxMenuItem(String text, Icon icon) | It creates an initially unselected check box menu item with the specified text and icon. |
| JCheckBoxMenuItem(String text, Icon icon, boolean b) | It creates a check box menu item with the specified text, icon, and selection state. |

### Methods

|  |  |  |
| --- | --- | --- |
| **Modifier** | **Method** | **Description** |
| AccessibleContext | getAccessibleContext() | It gets the AccessibleContext associated with this JCheckBoxMenuItem. |
| Object[] | getSelectedObjects() | It returns an [array](https://www.javatpoint.com/array-in-java) (length 1) containing the check box menu item [label](https://www.javatpoint.com/java-jlabel) or null if the check box is not selected. |
| boolean | getState() | It returns the selected-state of the item. |
| [String](https://www.javatpoint.com/java-string) | getUIClassID() | It returns the name of the L&F class that renders this component. |
| protected String | paramString() | It returns a string representation of this JCheckBoxMenuItem. |
| void | setState(boolean b) | It sets the selected-state of the item. |

## Java JCheckBoxMenuItem Example

1. **import** java.awt.event.ActionEvent;
2. **import** java.awt.event.ActionListener;
3. **import** java.awt.event.KeyEvent;
4. **import** javax.swing.AbstractButton;
5. **import** javax.swing.Icon;
6. **import** javax.swing.JCheckBoxMenuItem;
7. **import** javax.swing.JFrame;
8. **import** javax.swing.JMenu;
9. **import** javax.swing.JMenuBar;
10. **import** javax.swing.JMenuItem;
12. **public** **class** JavaCheckBoxMenuItem {
13. **public** **static** **void** main(**final** String args[]) {
14. JFrame frame = **new** JFrame("Jmenu Example");
15. frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);
16. JMenuBar menuBar = **new** JMenuBar();
17. // File Menu, F - Mnemonic
18. JMenu fileMenu = **new** JMenu("File");
19. fileMenu.setMnemonic(KeyEvent.VK\_F);
20. menuBar.add(fileMenu);
21. // File->New, N - Mnemonic
22. JMenuItem menuItem1 = **new** JMenuItem("Open", KeyEvent.VK\_N);
23. fileMenu.add(menuItem1);
25. JCheckBoxMenuItem caseMenuItem = **new** JCheckBoxMenuItem("Option\_1");
26. caseMenuItem.setMnemonic(KeyEvent.VK\_C);
27. fileMenu.add(caseMenuItem);
29. ActionListener aListener = **new** ActionListener() {
30. **public** **void** actionPerformed(ActionEvent event) {
31. AbstractButton aButton = (AbstractButton) event.getSource();
32. **boolean** selected = aButton.getModel().isSelected();
33. String newLabel;
34. Icon newIcon;
35. **if** (selected) {
36. newLabel = "Value-1";
37. } **else** {
38. newLabel = "Value-2";
39. }
40. aButton.setText(newLabel);
41. }
42. };
44. caseMenuItem.addActionListener(aListener);
45. frame.setJMenuBar(menuBar);
46. frame.setSize(350, 250);
47. frame.setVisible(**true**);
48. }
49. }

Output:

# 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

1. **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. |

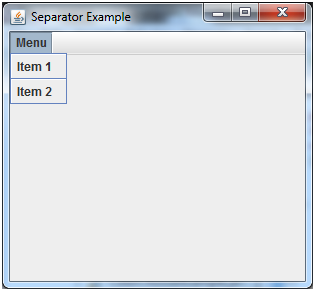
### Commonly used Methods of JSeparator

|  |  |
| --- | --- |
| **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

1. **import** javax.swing.\*;
2. **class** SeparatorExample
3. {
4. JMenu menu, submenu;
5. JMenuItem i1, i2, i3, i4, i5;
6. SeparatorExample()  {
7. JFrame f= **new** JFrame("Separator Example");
8. JMenuBar mb=**new** JMenuBar();
9. menu=**new** JMenu("Menu");
10. i1=**new** JMenuItem("Item 1");
11. i2=**new** JMenuItem("Item 2");
12. menu.add(i1);
13. menu.addSeparator();
14. menu.add(i2);
15. mb.add(menu);
16. f.setJMenuBar(mb);
17. f.setSize(400,400);
18. f.setLayout(**null**);
19. f.setVisible(**true**);
20. }
21. **public** **static** **void** main(String args[])
22. {
23. **new** SeparatorExample();
24. }}

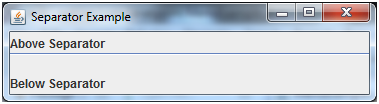
Output:



## Java JSeparator Example 2

1. **import** javax.swing.\*;
2. **import** java.awt.\*;
3. **public** **class** SeparatorExample
4. {
5. **public** **static** **void** main(String args[]) {
6. JFrame f = **new** JFrame("Separator Example");
7. f.setLayout(**new** GridLayout(0, 1));
8. JLabel l1 = **new** JLabel("Above Separator");
9. f.add(l1);
10. JSeparator sep = **new** JSeparator();
11. f.add(sep);
12. JLabel l2 = **new** JLabel("Below Separator");
13. f.add(l2);
14. f.setSize(400, 100);
15. f.setVisible(**true**);
16. }
17. }

Output:



# 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.

1. **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. |

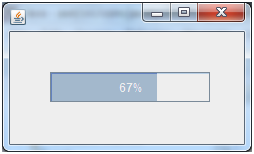
### Commonly used Methods:

|  |  |
| --- | --- |
| **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

1. **import** javax.swing.\*;
2. **public** **class** ProgressBarExample **extends** JFrame{
3. JProgressBar jb;
4. **int** i=0,num=0;
5. ProgressBarExample(){
6. jb=**new** JProgressBar(0,2000);
7. jb.setBounds(40,40,160,30);
8. jb.setValue(0);
9. jb.setStringPainted(**true**);
10. add(jb);
11. setSize(250,150);
12. setLayout(**null**);
13. }
14. **public** **void** iterate(){
15. **while**(i<=2000){
16. jb.setValue(i);
17. i=i+20;
18. **try**{Thread.sleep(150);}**catch**(Exception e){}
19. }
20. }
21. **public** **static** **void** main(String[] args) {
22. ProgressBarExample m=**new** ProgressBarExample();
23. m.setVisible(**true**);
24. m.iterate();
25. }
26. }

Output:



# Java JTree

The JTree class is used to display the tree structured data or hierarchical data. JTree is a complex component. It has a 'root node' at the top most which is a parent for all nodes in the tree. It inherits JComponent class.

## JTree class declaration

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

1. **public** **class** JTree **extends** JComponent **implements** Scrollable, Accessible

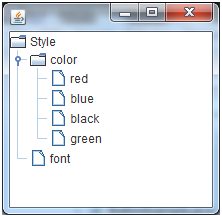
### Commonly used Constructors:

|  |  |
| --- | --- |
| **Constructor** | **Description** |
| JTree() | Creates a JTree with a sample model. |
| JTree(Object[] value) | Creates a JTree with every element of the specified array as the child of a new root node. |
| JTree(TreeNode root) | Creates a JTree with the specified TreeNode as its root, which displays the root node. |

## Java JTree Example

1. **import** javax.swing.\*;
2. **import** javax.swing.tree.DefaultMutableTreeNode;
3. **public** **class** TreeExample {
4. JFrame f;
5. TreeExample(){
6. f=**new** JFrame();
7. DefaultMutableTreeNode style=**new** DefaultMutableTreeNode("Style");
8. DefaultMutableTreeNode color=**new** DefaultMutableTreeNode("color");
9. DefaultMutableTreeNode font=**new** DefaultMutableTreeNode("font");
10. style.add(color);
11. style.add(font);
12. DefaultMutableTreeNode red=**new** DefaultMutableTreeNode("red");
13. DefaultMutableTreeNode blue=**new** DefaultMutableTreeNode("blue");
14. DefaultMutableTreeNode black=**new** DefaultMutableTreeNode("black");
15. DefaultMutableTreeNode green=**new** DefaultMutableTreeNode("green");
16. color.add(red); color.add(blue); color.add(black); color.add(green);
17. JTree jt=**new** JTree(style);
18. f.add(jt);
19. f.setSize(200,200);
20. f.setVisible(**true**);
21. }
22. **public** **static** **void** main(String[] args) {
23. **new** TreeExample();
24. }}

Output:



# Java JColorChooser

The JColorChooser class is used to create a color chooser dialog box so that user can select any color. It inherits [JComponent](https://www.javatpoint.com/java-jcomponent) class.

## JColorChooser class declaration

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

1. **public** **class** JColorChooser **extends** JComponent **implements** Accessible

### Commonly used Constructors:

|  |  |
| --- | --- |
| **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. |

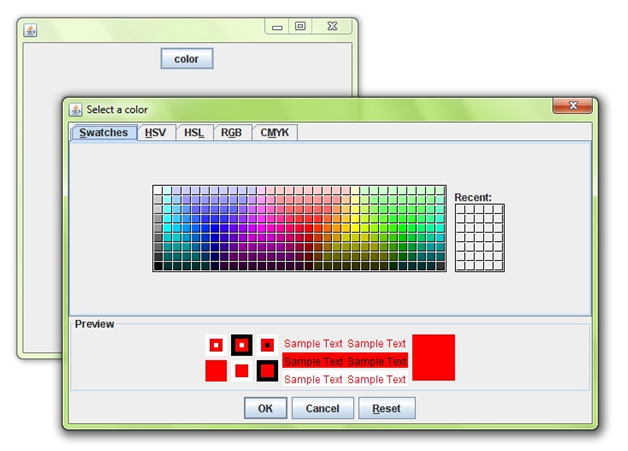
### Commonly used Methods:

|  |  |
| --- | --- |
| **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. |

## Java JColorChooser Example

1. **import** java.awt.event.\*;
2. **import** java.awt.\*;
3. **import** javax.swing.\*;
4. **public** **class** ColorChooserExample **extends** JFrame **implements** ActionListener {
5. JButton b;
6. Container c;
7. ColorChooserExample(){
8. c=getContentPane();
9. c.setLayout(**new** FlowLayout());
10. b=**new** JButton("color");
11. b.addActionListener(**this**);
12. c.add(b);
13. }
14. **public** **void** actionPerformed(ActionEvent e) {
15. Color initialcolor=Color.RED;
16. Color color=JColorChooser.showDialog(**this**,"Select a color",initialcolor);
17. c.setBackground(color);
18. }
20. **public** **static** **void** main(String[] args) {
21. ColorChooserExample ch=**new** ColorChooserExample();
22. ch.setSize(400,400);
23. ch.setVisible(**true**);
24. ch.setDefaultCloseOperation(EXIT\_ON\_CLOSE);
25. }
26. }

Output:

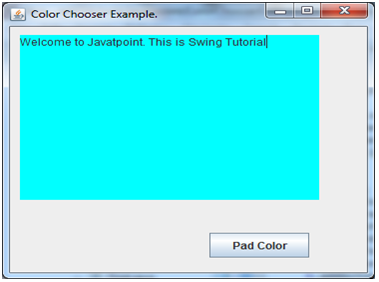


## Java JColorChooser Example with ActionListener

1. **import** javax.swing.\*;
2. **import** java.awt.\*;
3. **import** java.awt.event.\*;
4. **public** **class** ColorChooserExample **extends** JFrame **implements** ActionListener{
5. JFrame f;
6. JButton b;
7. JTextArea ta;
8. ColorChooserExample(){
9. f=**new** JFrame("Color Chooser Example.");
10. b=**new** JButton("Pad Color");
11. b.setBounds(200,250,100,30);
12. ta=**new** JTextArea();
13. ta.setBounds(10,10,300,200);
14. b.addActionListener(**this**);
15. f.add(b);f.add(ta);
16. f.setLayout(**null**);
17. f.setSize(400,400);
18. f.setVisible(**true**);
19. }
20. **public** **void** actionPerformed(ActionEvent e){
21. Color c=JColorChooser.showDialog(**this**,"Choose",Color.CYAN);
22. ta.setBackground(c);
23. }
24. **public** **static** **void** main(String[] args) {
25. **new** ColorChooserExample();
26. }
27. }

Output:

[](https://campaign.adpushup.com/get-started/?utm_source=banner&utm_campaign=growth_hack)



# Java JTabbedPane

The JTabbedPane class is used to switch between a group of components by clicking on a tab with a given title or icon. It inherits JComponent class.

## JTabbedPane class declaration

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

1. **public** **class** JTabbedPane **extends** JComponent **implements** Serializable, Accessible, SwingConstants

### Commonly used Constructors:

|  |  |
| --- | --- |
| **Constructor** | **Description** |
| JTabbedPane() | Creates an empty TabbedPane with a default tab placement of JTabbedPane.Top. |
| JTabbedPane(int tabPlacement) | Creates an empty TabbedPane with a specified tab placement. |
| JTabbedPane(int tabPlacement, int tabLayoutPolicy) | Creates an empty TabbedPane with a specified tab placement and tab layout policy. |

## Java JTabbedPane Example

1. **import** javax.swing.\*;
2. **public** **class** TabbedPaneExample {
3. JFrame f;
4. TabbedPaneExample(){
5. f=**new** JFrame();
6. JTextArea ta=**new** JTextArea(200,200);
7. JPanel p1=**new** JPanel();
8. p1.add(ta);
9. JPanel p2=**new** JPanel();
10. JPanel p3=**new** JPanel();
11. JTabbedPane tp=**new** JTabbedPane();
12. tp.setBounds(50,50,200,200);
13. tp.add("main",p1);
14. tp.add("visit",p2);
15. tp.add("help",p3);
16. f.add(tp);
17. f.setSize(400,400);
18. f.setLayout(**null**);
19. f.setVisible(**true**);
20. }
21. **public** **static** **void** main(String[] args) {
22. **new** TabbedPaneExample();
23. }}

Output:

