# SWING COMPONENTS

## CS – 25 ADVANCED JAVA PROGRAMMING (J2EE)
## BCA 5TH SEMESTER
## 2018 - 19

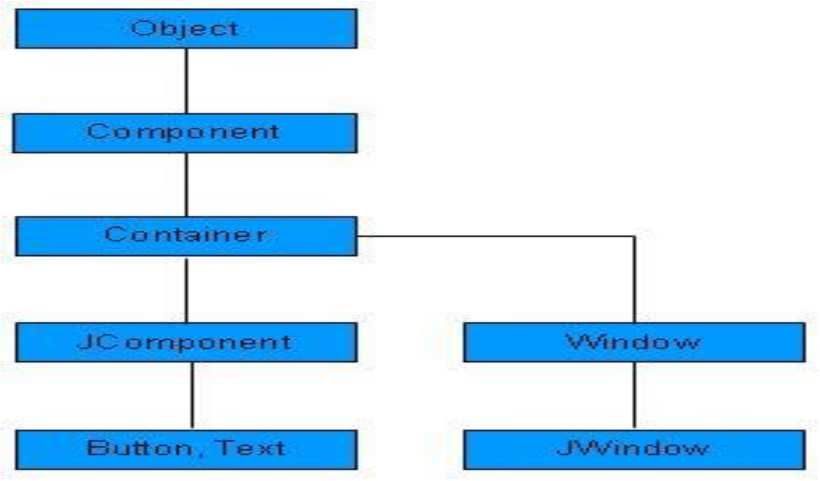
Dr. Sharon V. Mohtra

Assistant Professor, Christ College, Rajkot

## user interface

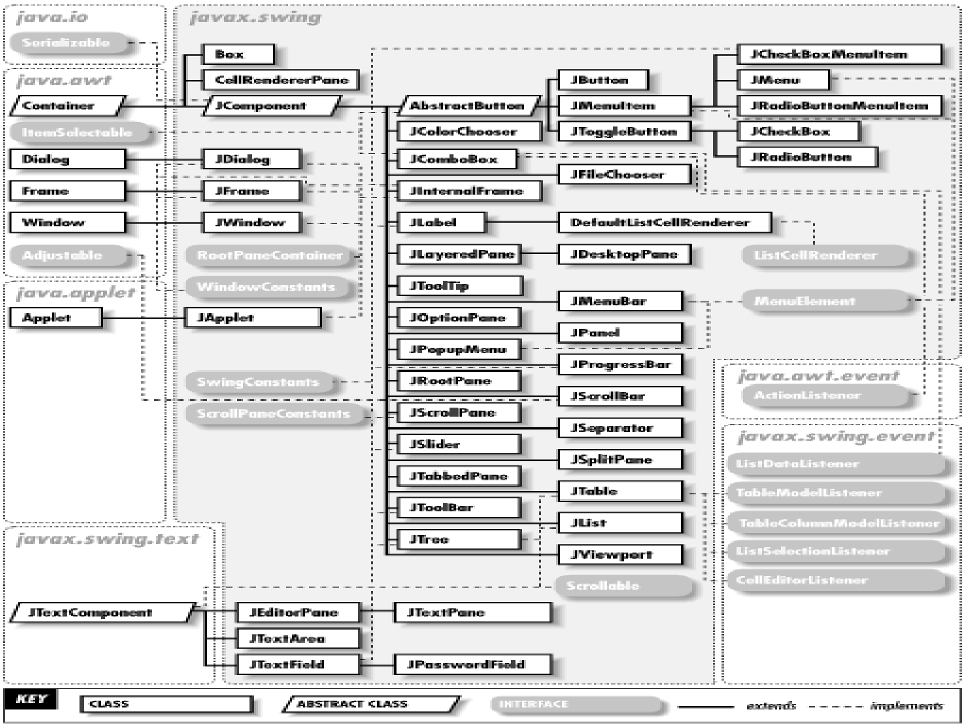
- □ **UI elements**
- □ **Layouts**
- □ **Behavior**

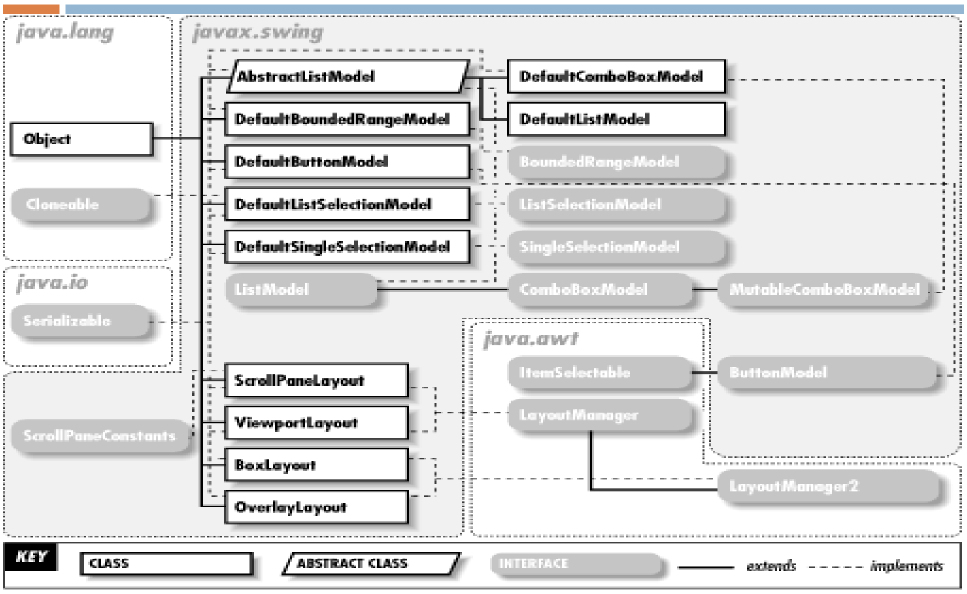
# UI hierarchy

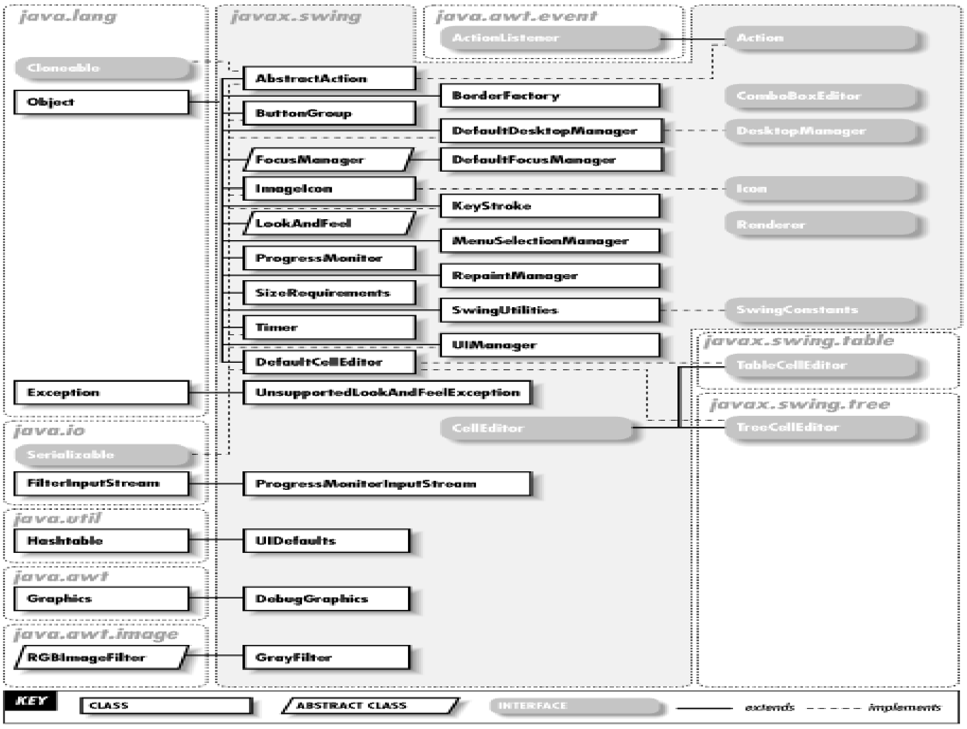


# Class & Description

- **Component**-A Container is the abstract base class for the non menu user-interface controls of SWING. Component represents an object with graphical representation
- **Container-**A Container is a component that can contain other SWING components.
- **JComponent -**A JComponent is a base class for all swing UI components. In order to use a swing component that inherits from JComponent, component must be in a containment hierarchy whose root is a top-level Swing container.


java.io
Serializable
java.awt
Container
ItemSelectable
Dialog
Frame
Window
Adjustable
java.applet
Applet
javax.swing.text
JTextComponent
javax.swing
Box
CellRendererPane
JComponent
JDialog
JFrame
JWindow
RootPaneContainer
WindowConstants
JApplet
SwingConstants
ScrollPaneConstants
AbstractButton
JColorChooser
JComboBox
JInternalFrame
JLabel
JLayeredPane
JToolTip
JOptionPane
JPopupMenu
JRootPane
JScrollPane
JSlider
JTabbedPane
JToolBar
JTree
JButton
JMenuItem
JToggleButton
JFileChooser
DefaultListCellRenderer
JDesktopPane
JMenuBar
JPanel
JProgressBar
JScrollBar
JSeparator
JSplitPane
JTable
JList
JViewport
Scrollable
JCheckBoxMenuItem
JMenu
JRadioButtonMenuItem
JCheckBox
JRadioButton
ListCellRenderer
MenuElement
java.awt.event
ActionListener
javax.swing.event
ListDataListener
TableModelListener
TableColumnModelListener
ListSelectionListener
CellEditorListener
JEditorPane
JTextArea
JTextField
JTextPane
JPasswordField
KEY
CLASS
ABSTRACT CLASS
INTERFACE
extends
implements


java.lang
Object
Cloneable
java.io
Serializable
ScrollPaneConstants
javax.swing
AbstractListModel
DefaultBoundedRangeModel
DefaultButtonModel
DefaultListSelectionModel
DefaultSingleSelectionModel
ListModel
ScrollPaneLayout
ViewportLayout
BoxLayout
OverlayLayout
DefaultComboBoxModel
DefaultListModel
BoundedRangeModel
ListSelectionModel
SingleSelectionModel
ComboBoxModel
MutableComboBoxModel
java.awt
ItemSelectable
LayoutManager
ButtonModel
LayoutManager2
KEY
CLASS
ABSTRACT CLASS
INTERFACE
extends
implements



# Swing Components

- □ Jlabel
- □ JTextField
- □ Jbutton
- □ JCheckBox
- □ JComboBox
- □ JRadioButton
- □ JScrollPane
- □ JList
- □ JTree
- □ JTable

# JLabel

## JLabel

- **Constructors**
- JLabel(Icon *icon)*
- JLabel(String *str)*
- JLabel(String *str, Icon icon,* int *align)*
- JLabel is a passive component.

## Parameter Description

□ Icon:

–An interface.

–Implemented by **ImageIcon class**

□ and the icon image is encapsulated.

–ImageIcon(String *filename)*

–Icon getIcon()

–void setIcon(Icon *icon)*

## Parameter Description

□ String:

–Displayed as the label text.

–String getText()

– void setText(String *str)*

□ Align:

–To set the horizontal alignment of the string or icon of label.

– LEFT, RIGHT, CENTER, LEADING,TRAILING.

# JLabel

Swing Examples\JLabelDemo.java

# JTextField

## JTextField

- **Constructors**
- JTextFIeld(int cols)
- JTextField(String *str, int cols)*
- JTextField(String *str)*

## Parameter Description

- String:

– Text to be initially presented.

– setText(String *str)*

- Cols:

– Number of columns (ultimately the width) in text field.

## JTextField

Swing Examples\JTextFieldDemo.java

## JButton

# JBuilder

- **Constructors**
- JButton(Icon *icon)*
- JButton(String *str)*
- JButton(String *str, Icon icon)*

# Associated Methods

- String getActionCommand()
- public void setActionCommand(String *str)*

## JButton

Swing Examples\JButtonDemo.java

## JCheckBox

## JCheckBox

- **Constructors**
- JCheckBox(String *str)*
- JCheckBox(Icon *i, boolean state)*
- JCheckBox(String *s)*
- JCheckBox(String *s, boolean state)*
- JCheckBox(String *s, Icon i)*
- JCheckBox(String *s, Icon i, boolean state)*

## Parameter Description

- State:

–Either true or false.

–True implies the checked state and false implies unchecked state.

## Associated Methods

- void setText(String *str)*
- String getText()
- boolean isSelected()
- void setSelected(boolean *state*)
- public void itemStateChanged()

## JCheckBox

Swing Examples\JCheckBoxDemo.java

# JRadioButton

## JRadioButton

- **Constructors**
- JRadioButton(Icon *i)*
- JRadioButton(Icon *i, boolean state)*
- JRadioButton(String *s)*
- JRadioButton(String *s, boolean state)*
- JRadioButton(String *s, Icon i)*
- JRadioButton(String *s, Icon i, boolean state)*

## Important Note

- Radio buttons must be configured into a group.
- Only one of the buttons in that group can be selected at any time.
- Use of ButtonGroup class.
- void add(AbstractButton *ab)*
- ActionListener is associated with its events.

## JRadioButton

Swing Examples\JRadioButtonDemo.java

# JComboBox

## JComboBox

- □ **Constructors**
- □ JComboBox( )
- □ JComboBox(Vector v)

## Parameter Description

□ Vector:

–v *is a vector that initializes the* combo box.

## Associated Methods

□ void addItem(Object *obj)*

□ int getSelectedIndex()

□ void setSelectedIndex(int)

□ Object getSelectedItem()

□ void setSelectedItem(Object)

# JComboBox

Swing Examples\JComboBoxDemo.java

# JScrollPane

# JScrollPane

- □ **Constructors**
- □ JScrollPane(Component *comp)*
- □ JScrollPane(int *vsb, int hsb)*
- □ JScrollPane(Component *comp, int vsb, int hsb)*

# Parameter Description

□ Comp:

–Component to be added inside scroll pane.

□ Vsb:

–Vertical scroll bar.

□ Hsb:

– Horizontal scroll bar.

## Parameter Description

- □ HORIZONTAL_SCROLLBAR_ALWAYS

- Always provide horizontal scroll bar.

- □ HORIZONTAL_SCROLLBAR_AS_NEEDED

- Provide horizontal scroll bar if needed.

- □ VERTICAL_SCROLLBAR_ALWAYS

- Always provide vertical scroll bar.

- □ VERTICAL_SCROLLBAR_AS_NEEDED

- Provide vertical scroll bar, if needed.

## Steps

- □ Create a **JComponent object.**
- □ Create a **JScrollPane object. (The arguments to the constructor specify the component and the** policies for vertical and horizontal scroll bars)
- □ Add the scroll pane to the content pane of the applet.

# JScrollPane

Swing Examples\JScrollPaneDemo.java

# JList

## JList

□ **Constructors**

□ JList()

□ JList(Object[] items)

– Here, object array can be in terms of String or any other relevant data type.

## Selection Modes

□ SINGLE_SELECTION

□ SINGLE_INTERVAL_SELECTION

□ MULTIPLE_INTERVAL_SELECTION

## Associated Methods

- void valueChanged(ListSelectionEvent le)
- void setSelectionMode(int *mode*)
- int getSelectedIndex()
- Object getSelectedValue()

## JList

Swing Examples\JListDemo.java

# JTree

## JTree

- **Constructors**
- JTree(Hashtable *ht)*
- JTree(Object *obj[ ])*
- JTree(TreeNode *tn)*
- JTree(Vector *v)*

## Parameter Description

- The objects of Hashtable class, Object class and Vector class are placed as child nodes where as the object of TreeNode class is placed as a root node in the tree.

## Associated Methods

- void addTreeExpansionListener(TreeExpa nsionListener *tel)*
- void removeTreeExpansionListener(TreeE xpansionListener *tel)*
- TreePath getPathForLocation(int *x, int y)*

## Declaring Tree Nodes

□ DefaultMutableTreeNode:

– The **DefaultMutableTreeNode class**

□ implements the **MutableTreeNode interface.**

– The **MutableTreeNode interface extends**

□ **TreeNode interface. It declares methods that** can insert and remove child nodes or change the parent node.

– **It represents a node in a tree. One of its** constructors is shown here:

– **DefaultMutableTreeNode(Object *obj*)**

## Associated Methods

□ void add(MutableTreeNode *child*)

□ TreePath getPath( )

□ void treeCollapsed(TreeExpansionEvent *tee*)

□ void treeExpanded(TreeExpansionEvent *tee*)

## How to create a Tree??

- Create a **JTree object.**
- Create a **JScrollPane object. (The arguments to the constructor specify the tree and the policies for** vertical and horizontal scroll bars.)
- Add the tree to the scroll pane.
- Add the scroll pane to the content pane of the applet.

## JTree

Swing Examples\JTreeDemo.java

# JTable

## JTable

- **Constructors**
- JTable(Object *data[ ][ ]*, Object *colHeads[ ]*)

## Parameter Description

- Object *data[ ][ ]:*
- –A two-dimensional array of the
- information to be presented.
- • *Object colHeads[ ]:*
- –A one-dimensional array with the
- column headings.

## How to create a Table?

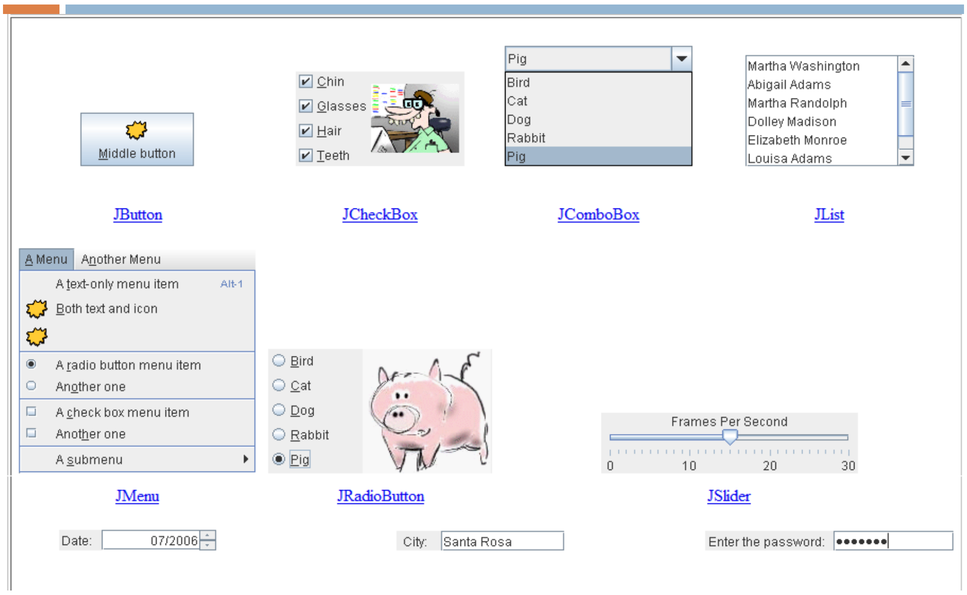
- Create a **JTable object.**
- Create a **JScrollPane object. (The arguments to the constructor specify the table and the policies for vertical** and horizontal scroll bars.)
- Add the table to the scroll pane.
- Add the scroll pane to the content pane of the applet.

## JTable

Swing Examples\JTableDemo.java

## Visual Guide

Basic Controls = Simple components that are used primarily to get input from the user; they may also show simple state.



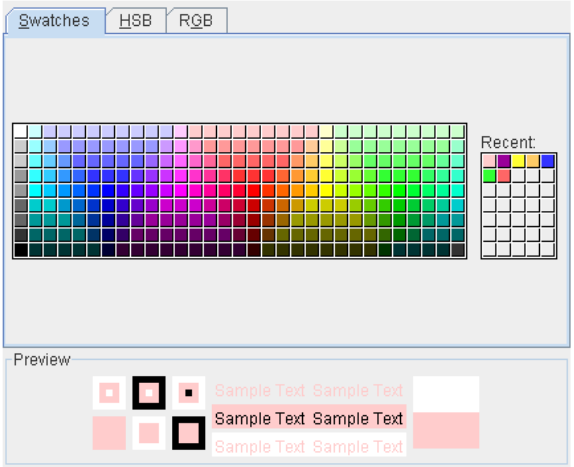
Date: 07/2006

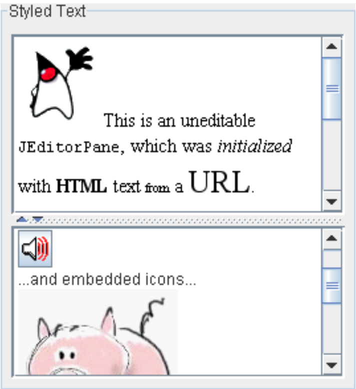
City: Santa Rosa

Enter the password: •••••••

## Interactive Displays of Highly Formatted Information

These components display highly formatted information that (if you choose) can be modified by the user.



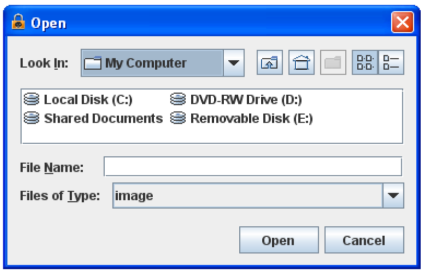


This is an uneditable JEditorPane, which was *initialized* with **HTML** text from a URL.

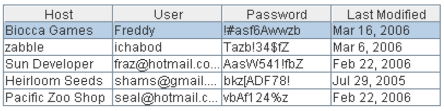
...and embedded icons...

JColorChooser

JEditorPane and JTextPane

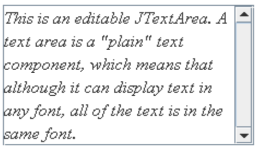


JFileChooser



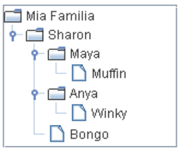
| Host | User | Password | Last Modified |
|---|---|---|---|
| Biocca Games | Freddy | !#asf6Awwzb | Mar 16, 2006 |
| zabble | ichabod | Tazb!34$fZ | Mar 6, 2006 |
| Sun Developer | fraz@hotmail.co... | AasW541!fbZ | Feb 22, 2006 |
| Heirloom Seeds | shams@gmail.... | bkz[ADF78! | Jul 29, 2005 |
| Pacific Zoo Shop | seal@hotmail.c... | vbAf124%z | Feb 22, 2006 |

JTable



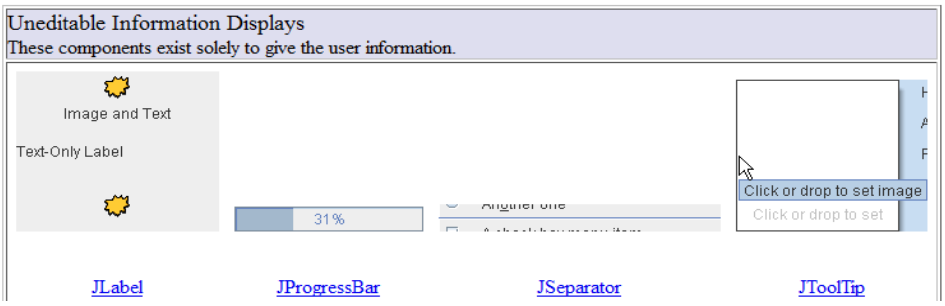
*This is an editable JTextArea. A text area is a "plain" text component, which means that although it can display text in any font, all of the text is in the same font.*

JTextArea



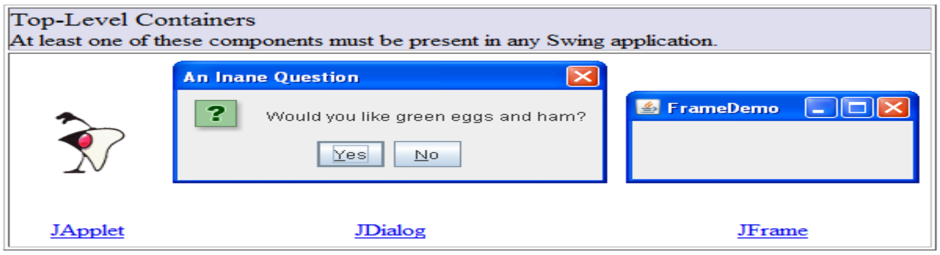
JTree

# Uneditable and Top-level



Uneditable Information Displays

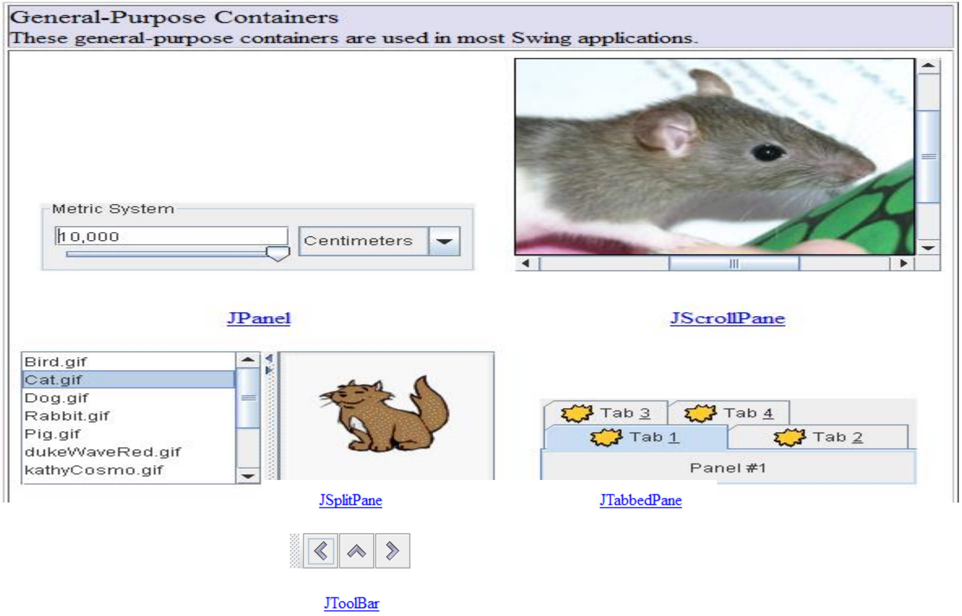
These components exist solely to give the user information.



Top-Level Containers

At least one of these components must be present in any Swing application.

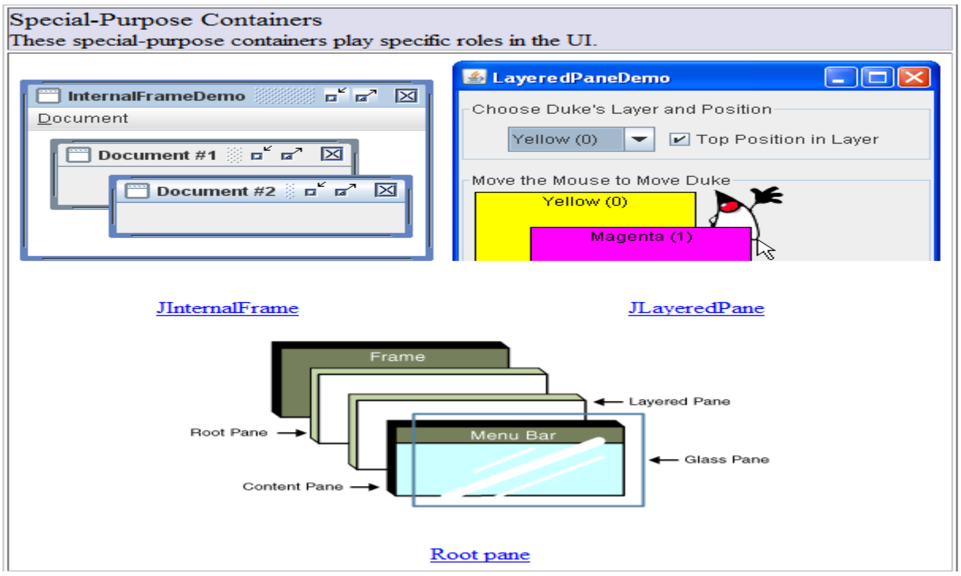
# General-Purpose



General-Purpose Containers

These general-purpose containers are used in most Swing applications.

# Special-Purpose



Special-Purpose Containers

These special-purpose containers play specific roles in the UI.