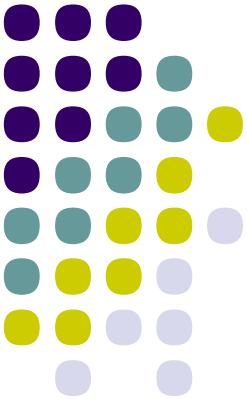
GUI components



Nguyễn Thị Tú Mi
Email: nttmi@hcmuaf.edu.vn

Các thành phần GUI Swing

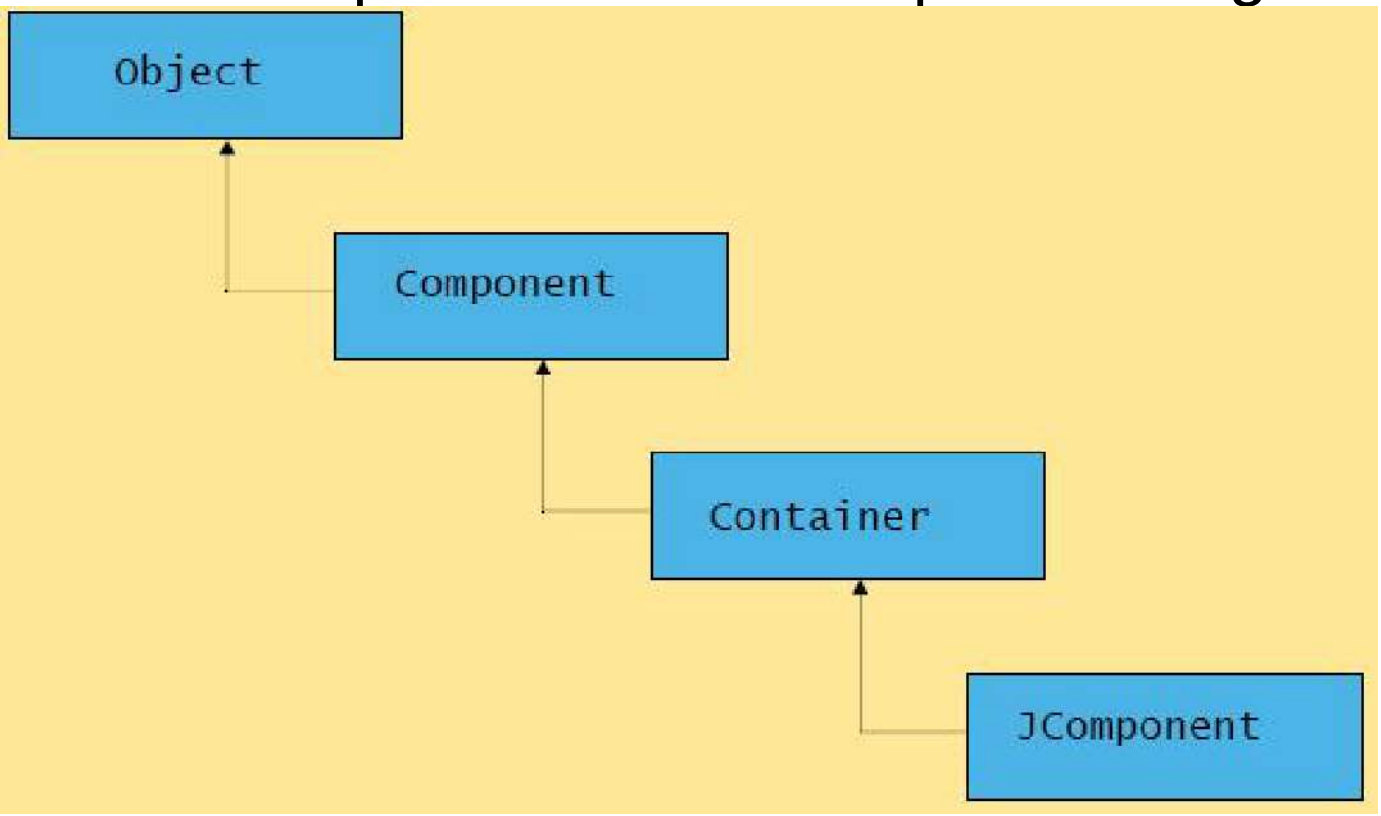
- Gói javax.swing.*
- Các thành phần bắt nguồn từ AWT (gói java.awt.*)
- Chứa đựng cảm quan (look and feel)
  - Sự thể hiện và cách người sử dụng tương tác với chương trình
- Những thành phần nhẹ (lightweight)
  - Được viết hoàn toàn bằng Java

Các thành phần GUI Swing

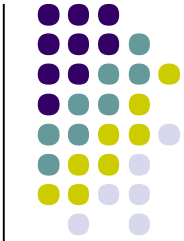
- Các thành phần
  - Chứa phương thức paint() để vẽ thành phần trên màn hình
- Các bộ chứa
  - Tập hợp các thành phần liên quan
  - Chứa phương thức add() để thêm các thành phần
- Lớp JComponent
  - Cảm quan khả kiến (Pluggable)
  - Phím tắt (tính dễ nhớ)
  - Khả năng xử lý sự kiện chung

Các thành phần GUI Swing

- Các siêu lớp của nhiều thành phần Swing



JComponent Class



Property	Data type	get	is	set	Default value (if applicable)
background	Color	.		.	
colorModel	ColorModel	.			
component[i]	Component	.			
componentCount	int	.			
components	Component[]	.			
cursor	Cursor	.		.	Cursor.DEFAULT_CURSOR
enabled	boolean		.	.	true
font	Font	.		.	
foreground	Color	.		.	
insets	Insets	.			Insets(0,0,0,0)
layout	LayoutManager	.		.	BorderLayout( )
locale	Locale	.		.	
location	Point	.		.	
locationOnScreen	Point	.		.	
name	String	.		.	""
parent	Container	.		.	null
size	Dimension	.		.	
showing	boolean		.		true
valid	boolean		.		
visible	boolean		.	.	true

[i]indexed

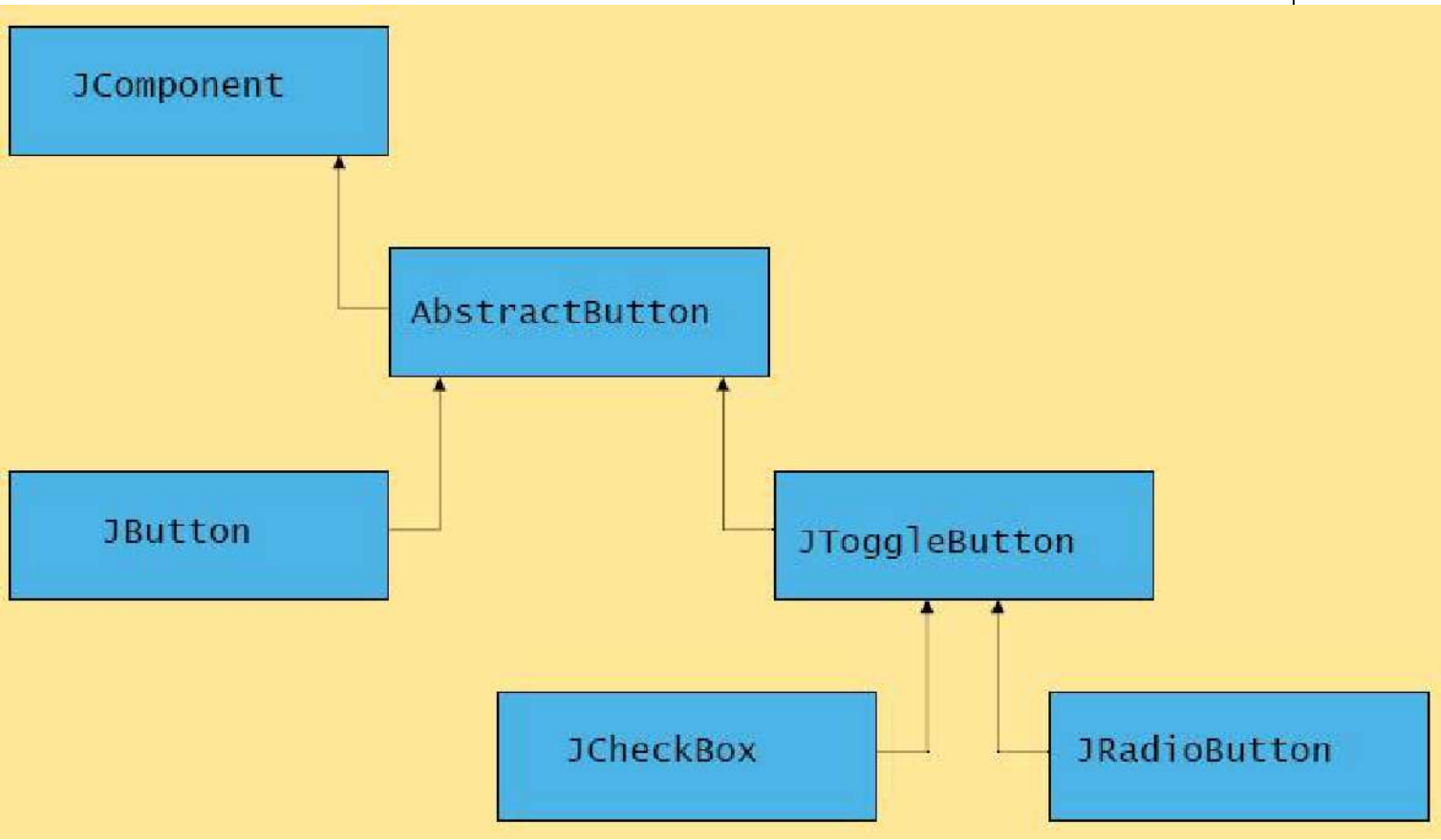
Các thành phần GUI cơ bản

- JButton
- JToggleButton
- JRadioButton
- JCheckBox
- JColorChooser
- JLabel
- JTextField
- JTexArea
- JList
- JComboBox
- JPopupMenu
- JToolBar

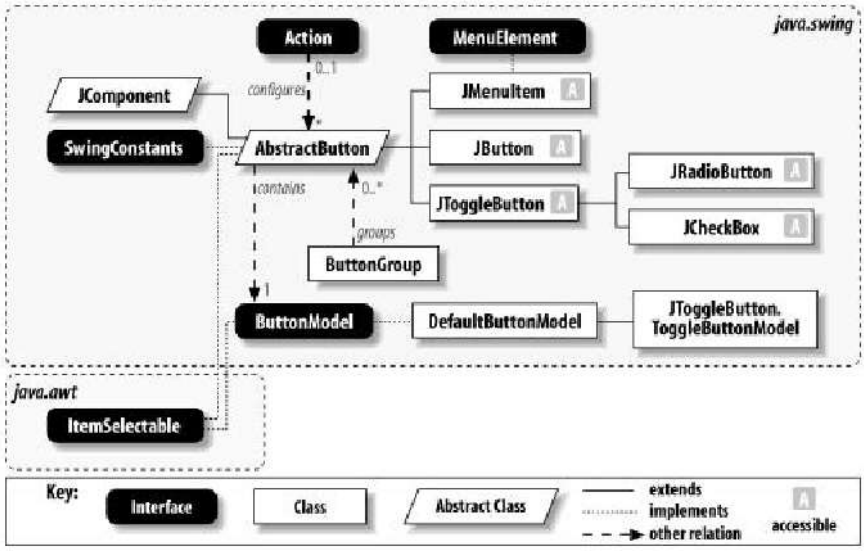
JButton

- Nút nhấn - thành phần người sử dụng nhấp để kích hoạt một hành động cụ thể.
- Một vài kiểu khác nhau
  - Command Button
  - Check Box
  - Radio Button
  - JToggle Button
  - …
- Các lớp dẫn xuất javax.swing.AbstractButton
  - Command Button được tạo với lớp JButton
- Sinh ra một ActionEvent khi người sử dụng nhấn trên nút.

Cây thừa kế các JButton



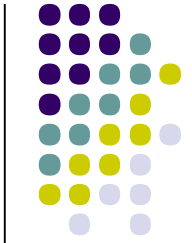
Button class diagram



ButtonModel Interface

- Đối tượng ButtonModel dùng để quản lý trạng thái của các Button.
- ButtonModel dùng để đọc và ghi trạng thái của các nút cũng như thêm và xóa các sự kiện trên nút đó.

Thuộc tính của ButtonModel



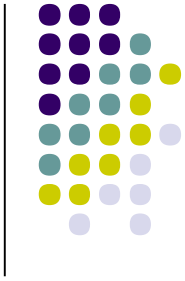
Property	Data type	get	is	set	Default value
actionCommand	String	·		·	
armed	boolean		·	·	
enabled	boolean		·	·	
group	ButtonGroup			·	
mnemonic	int	·		·	
pressed	boolean		·	·	
rollover	boolean		·	·	
selected	boolean		·	·	

ButtonModel Events

Event	Description
ActionEvent	The button is pressed.
ChangeEvent	A change has occurred in one or more properties of the button model.
ItemEvent	The button is toggled on or off.

```
public void addActionListener(ActionListener l)
public void removeActionListener(ActionListener l)
public void addItemListener(ItemListener l)
public void removeItemListener(ItemListener l)
public void addChangeListener(ChangeListener l)
public void removeChangeListener(ChangeListener l)
```

DefaultButtonModel Class

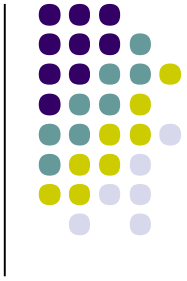


- Các thuộc tính

Property	Data type	get	is	set	Default value
actionCommand°	String	·		·	null
armed°	boolean		·	·	false
enabled°	boolean		·	·	true
group°, *	ButtonGroup	·		·	null
mnemonic°	int	·		·	KeyEvent.VK_UNDEFINED
pressed°	boolean		·	·	false
rollover°	boolean		·	·	false
selected°	boolean		·	·	false
selectedObjects	Object[]	·			null

°overridden, *getter was introduced in SDK 1.3

DefaultButtonModel Class



- Events

  public void addActionListener(ActionListener l)
  public void removeActionListener(ActionListener l)
  public ActionListener[] getActionListeners( ) (added in SDK 1.3)
  public void addItemListener(ItemListener l)
  public void removeItemListener(ItemListener l)
  public ItemListener[] getItemListeners( ) (added in SDK 1.3)
  public void addChangeListener(ChangeListener l)
  public void removeChangeListener(ChangeListener l)
  public ChangeListener[] getChangeListeners( ) (added in SDK 1.3)
  public EventListener[] getListeners(Class listenerType) (SDK 1.4)

DefaultButtonModel Class

- Constants of state

Constant	Type
ARMED	int
ENABLED	int
PRESSED	int
ROLLOVER	int
SELECTED	int

- Constructor
  - public DefaultButtonModel( )

AbstractButton Class

- Thuộc tính

Property	Data type	get	is	set	Default value
action[1.3]	Action	·		·	null
actionCommand	String	·		·	null
borderPainted[b]	boolean		·	·	true
contentAreaFilled[b]	boolean		·	·	true
disabledIcon[b]	Icon	·		·	null
disabledSelectedIcon[b]	Icon	·		·	null
displayedMnemonicIndex[1.4]	int	·		·	-1
enabled[e]	boolean		·	·	true
focusPainted[b]	boolean		·	·	true
horizontalAlignment[b]	int	·		·	CENTER
horizontalTextPosition[b]	int	·		·	TRAILING[1.4]
icon[b]	Icon	·		·	null
iconTextGap[1.4]	int	·		·	4
labelId	String	·		·	Same as `text`
margin[b]	Insets	·		·	null
mnemonic[b]	int	·		·	KeyEvent.VK_UNDEFINED
model[b]	ButtonModel	·		·	null
multiClickThreshhold[1.4]	long	·		·	0
pressedIcon[b]	Icon	·		·	null
rolloverEnabled[b]	boolean		·	·	false
rolloverIcon	Icon	·		·	null
rolloverSelectedIcon[b]	Icon	·		·	null
selected	boolean		·	·	false
selectedIcon[b]	Icon	·		·	null
selectedObjects	Object[]	·			null
text[b]	String	·		·	""
UI[b]	ButtonUI	·		·	From L&F
verticalAlignment[b]	int	·		·	CENTER

AbstractButton Class

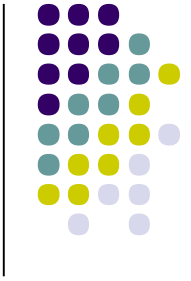
- Events
  - public void addActionListener(ActionListener l)
  - public void removeActionListener(ActionListener l)
  - public ActionListener[] getActionListeners( ) (Added in SDK 1.4)
  - public void addItemListener(ItemListener l)
  - public void removeItemListener(ItemListener l)
  - public ItemListener[] getItemListeners( ) (Added in SDK 1.4)
  - public void addChangeListener(ChangeListener l)
  - public void removeChangeListener(ChangeListener l)
  - public ChangeListener[] getChangeListeners( ) (Added in SDK 1.4)

AbstractButton Class

- Constants of properties

Constant	Type	Description
BORDER_PAINTED_CHANGED_PROPERTY	String	`borderPainted` property has changed
CONTENT_AREA_FILLED_CHANGED_PROPERTY	String	`contentAreaFilled` property has changed
DISABLED_ICON_CHANGED_PROPERTY	String	`disabledIcon` property has changed
DISABLED_SELECTED_ICON_CHANGED_PROPERTY	String	`disabledSelectedIcon` property has changed
FOCUS_PAINTED_CHANGED_PROPERTY	String	`focusPainted` property has changed
HORIZONTAL_ALIGNMENT_CHANGED_PROPERTY	String	`horizontalAlignment` property has changed
HORIZONTAL_TEXT_POSITION_CHANGED_PROPERTY	String	`horizontalTextPosition` property has changed
ICON_CHANGED_PROPERTY	String	`icon` property has changed
MARGIN_CHANGED_PROPERTY	String	`margin` property has changed
MNEMONIC_CHANGED_PROPERTY	String	`mnemonic` property has changed
MODEL_CHANGED_PROPERTY	String	`model` property has changed
PRESSED_ICON_CHANGED_PROPERTY	String	`pressedIcon` property has changed
ROLLOVER_ENABLED_CHANGED_PROPERTY	String	`rolloverEnabled` property has changed
ROLLOVER_ICON_CHANGED_PROPERTY	String	`rolloverIcon` property has changed
ROLLOVER_SELECTED_ICON_CHANGED_PROPERTY	String	`rolloverSelectedIcon` property has changed
SELECTED_ICON_CHANGED_PROPERTY	String	`selectedIcon` property has changed
TEXT_CHANGED_PROPERTY	String	`text` property has changed
VERTICAL_ALIGNMENT_CHANGED_PROPERTY	String	`verticalAlignment` property has changed
VERTICAL_TEXT_POSITION_CHANGED_PROPERTY	String	`verticalTextPosition` property has changed

JButton class



- Thuộc tính

Property	Data type	get	is	set	Default value
accessibleContext°	AccessibleContext	·			JButton.AccessibleJButton( )
defaultButton	boolean		·		false
defaultCapable	boolean		·	·	true
model°	ButtonModel	·		·	DefaultButtonModel( )
UIClassID°	String	·			"ButtonUI"
°overridden					

- Constructors
  - public JButton( )
  - public JButton(Action a)
  - public JButton(Icon icon)
  - public JButton(String text)
  - public JButton(String text, Icon icon)

JToggleButton Class

- Thuộc tính

Property	Data type	get	is	set	Default value
accessibleContext°	AccessibleContext	·			JToggleButton.AccessibleJToggleButton( )
model°	ButtonModel	·		·	ToggleButtonModel( )
UIClassID°	String	·			"ToggleButtonUI"
°overridden					

- Constructors
  - public JToggleButton( )
  - public JToggleButton(Action a)
  - public JToggleButton(Icon icon)
  - public JToggleButton(Icon icon, boolean selected)
  - public JToggleButton(String text)
  - public JToggleButton(String text, boolean selected)
  - public JToggleButton(String text, Icon icon)
  - public JToggleButton(String text, Icon icon, boolean selected)

JCheckBox Class

- Thuộc tính

Property	Data type	get	is	set	Default value
accessibleContext°	AccessibleContext	·			AccessibleJCheckBox
borderPainted°	boolean		·	·	false
borderPaintedFlat[1.3, b]	boolean		·	·	false
horizontalAlignment°	int	·		·	LEADING[1.4]
UIClassID°	String	·			"CheckBoxUI"

Constant	Type	Description
BORDER_PAINTED_FLAT_CHANGED_PROPERTY	String	`borderPaintedFlat` property has changed

JCheckBox Class

- Constructors
  - public JCheckBox( )
  - public JCheckBox(Action a)
  - public JCheckBox(Icon icon)
  - public JCheckBox(Icon icon, boolean selected)
  - public JCheckBox(String text)
  - public JCheckBox(String text, boolean selected)
  - public JCheckBox(String text, Icon icon)
  - public JCheckBox(String text, Icon icon, boolean selected)

JRadioButton Class

- Thuộc tính

Property	Data type	get	is	set	Default value
accessibleContext°	AccessibleContext	·			JRadioButton.AccessibleJRadioButton( )
borderPainted°	boolean		·	·	false
horizontalAlignment°	int	·		·	LEADING[1.4]
UIClassID°	String	·			"RadioButtonUI"

[1.4]since 1.4, °overridden

- Constructors
  - public JRadioButton( )
  - public JRadioButton(Action a)
  - public JRadioButton(Icon icon)
  - public JRadioButton(Icon icon, boolean selected)
  - public JRadioButton(String text)
  - public JRadioButton(String text, boolean selected)
  - public JRadioButton(String text, Icon icon)
  - public JRadioButton(String text, Icon icon, boolean selected)

ButtonGroup Class

- Dùng để nhóm các nút lại với nhau, đảm bảo sẽ có tối đa 1 nút được chọn
- ButtonGroups thường được dùng để nhóm các JRadioButtons hay JRadioButtonMenuItems
- Thuộc tính

Property	Data type	get	is	set	Default value
buttonCount	int	·			0
elements	Enumeration	·			Empty
selection	ButtonModel	·			null

ButtonGroup Class

- Constructor
  - public ButtonGroup( )
- Methods
  - public void add(AbstractButton b)
  - public void remove(AbstractButton b)
  - public void setSelected(ButtonModel m, boolean b)
  - public boolean isSelected(ButtonModel m)

Ví dụ sử dụng JButton

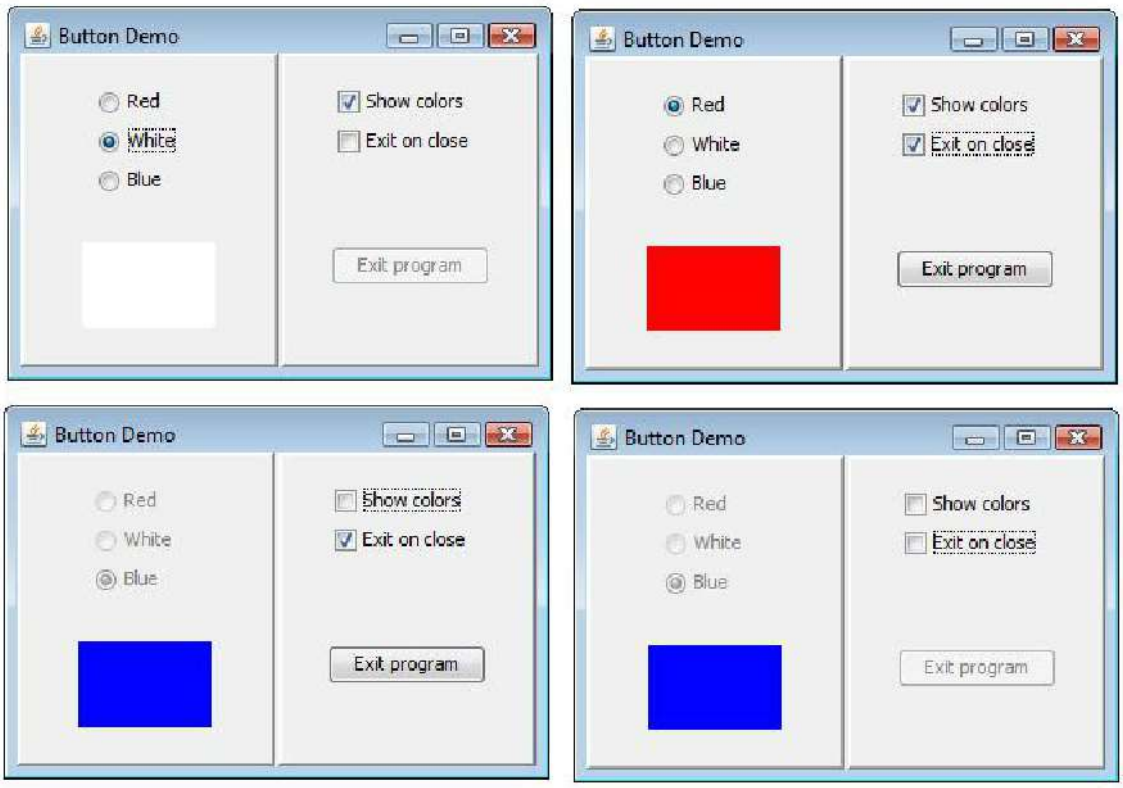
```
public class ButtonIcon extends JFrame {
    public ButtonIcon() {
        try {
            UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());
        } catch (Exception e) {
            e.printStackTrace();        }
        addWindowListener(new WindowAdapter() {
            @Override
            public void windowClosing(WindowEvent e) {
                System.exit(0);
            }
        });
        ImageIcon prevIcon = new ImageIcon("prev.gif");
        JButton prevButton = new JButton("Previous", prevIcon);
        prevButton.setToolTipText("Move to the previous page");
        ImageIcon nextIcon = new ImageIcon("next.gif");
        JButton nextButton = new JButton("Next", nextIcon);
        nextButton.setToolTipText("Move to the next page");
        Container content = getContentPane();
        content.setLayout(new FlowLayout());
        content.add(prevButton);
        content.add(nextButton);
    }
    public static void main(String[] args) {
        ButtonIcon app = new ButtonIcon();
        app.setTitle("Button Icon Demo");
        app.setSize(320, 80);
        app.setVisible(true);    }}
```

Ví dụ sử dụng JToogleButton

Ví dụ sử dụng JToogleButton

```
public class ToogleDemo extends JFrame {
    ImageIcon successIcon;
    ImageIcon unsuccesIcon;
    JToggleButton success_unsuccess_Button;

    public ToogleDemo() {
        ....
        successIcon = new ImageIcon("success.gif");
        unsuccesIcon = new ImageIcon("unsuccess.gif");
        success_unsuccess_Button = new JToggleButton("Success", successIcon);
        success_unsuccess_Button.addChangeListener(new ChangeListener() {

            @Override
            public void stateChanged(ChangeEvent e) {
                if (success_unsuccess_Button.isSelected()) {
                    success_unsuccess_Button.setIcon(successIcon);
                    success_unsuccess_Button.setText("Success");
                } else {
                    success_unsuccess_Button.setIcon(unsuccesIcon);
                    success_unsuccess_Button.setText("Unsuccess");
                }
            }
        });
        Container content = getContentPane();
        content.setLayout(new FlowLayout());
        content.add(success_unsuccess_Button);
    }
...
```

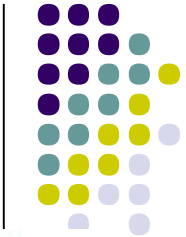
Ví dụ sử dụng JRadioButton & JCheckBox



Ví dụ sử dụng JRadioButton & JCheckBox

```
public class ButtonDemo extends JFrame implements ActionListener {
    ButtonGroup group;
    JRadioButton redButton;
    JRadioButton whiteButton;
    JRadioButton blueButton;
    JPanel colorBox;
    JCheckBox showColorsButton; // o chon thu 1
    JCheckBox exitOnClose; // o chon thu 2
    JButton exitButton;

    public ButtonDemo() {
        try {
            UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());
        } catch (Exception e) {
            e.printStackTrace();
        }
        addWindowListener(new WindowAdapter() {
            @Override
            public void windowClosing(WindowEvent e) {
                System.exit(0);
            }
        });
```

Ví dụ sử dụng JRadioButton & JCheckBox

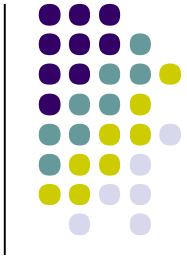


```
// tao bang nut radio va khung ben trong de giup hien thi gon gang hon
JPanel radioPane = new JPanel();
JPanel innerRadioPane = new JPanel();
radioPane
        .setBorder(BorderFactory.createBevelBorder(BevelBorder.RAISED));
innerRadioPane
        .setLayout(new BoxLayout(innerRadioPane, BoxLayout.Y_AXIS));
innerRadioPane.setBorder(BorderFactory
        .createEmptyBorder(10, 10, 10, 10));

// xay dung nhom radio va cac nut cua no
// tat ca bien co den ActionListener cua chuong trinh
group = new ButtonGroup();
redButton = new JRadioButton("Red");
whiteButton = new JRadioButton("White");
blueButton = new JRadioButton("Blue");
whiteButton.setSelected(true);// chon 1 nut
redButton.addActionListener(this);// xem actionPerformed
whiteButton.addActionListener(this);
blueButton.addActionListener(this);
group.add(redButton); // nhom dam bao khi 1 nut da chon thi cac nut khac
// se tat
group.add(whiteButton);
group.add(blueButton);
```

Ví dụ sử dụng JRadioButton & JCheckBox

```
// xay dung khung nho hien thi mau duoc chon
colorBox = new JPanel();
colorBox.setBackground(Color.white);
colorBox.setPreferredSize(new Dimension(50, 50));

// them doi tuong GUI vao khung radio trong
innerRadioPane.add(redButton);
innerRadioPane.add(whiteButton);
innerRadioPane.add(blueButton);
innerRadioPane.add(Box.createRigidArea(new Dimension(0, 25)));
innerRadioPane.add(colorBox);

// them khung trang vao khung radio noi ben trai
radioPane.add(innerRadioPane);

// tao khung o chon va khung trong
JPanel checkPane = new JPanel();
JPanel innerCheckPane = new JPanel();
checkPane
        .setBorder(BorderFactory.createBevelBorder(BevelBorder.RAISED));
innerCheckPane
        .setLayout(new BoxLayout(innerCheckPane, BoxLayout.Y_AXIS));
innerCheckPane.setBorder(BorderFactory
        .createEmptyBorder(10, 10, 10, 10));
```

Ví dụ sử dụng JRadioButton & JCheckBox



```
// tao doi tuong o chon "show colors" va
// kich hoat hoac vo hieu hoa cac nut radio chon mau
showColorsButton = new JCheckBox("Show colors");
showColorsButton.setSelected(true);
showColorsButton.addChangeListener(new ChangeListener() {
    @Override
    public void stateChanged(ChangeEvent e) {
        boolean t = showColorsButton.isSelected();
        redButton.setEnabled(t);// kich hoat hoac vo hieu hoa
        whiteButton.setEnabled(t);// cac nut radio tuy theo
        blueButton.setEnabled(t);// trang thai cua o chon
    }
});
// tao doi tuong o chon "Exit on close" va khoi hoat hay vo hieu hoa nut
// Exit
exitOnClose = new JCheckBox("Exit on close");
exitOnClose.addChangeListener(new ChangeListener() {
    @Override
    public void stateChanged(ChangeEvent e) {
        boolean t = exitOnClose.isSelected();
        exitButton.setEnabled(t);
    }
});
```

Ví dụ sử dụng JRadioButton & JCheckBox

```
    // tao nut tron "Exit program" va lop tiep nhan bien co thao tac
    exitButton = new JButton("Exit program");
    exitButton.setEnabled(false);
    exitButton.addActionListener(new ActionListener() {

        @Override
        public void actionPerformed(ActionEvent e) {
            System.exit(0);
        }
    });

    // them cac nut vao o ben trong
    innerCheckPane.add(showColorsButton);
    innerCheckPane.add(exitOnClose);
    innerCheckPane.add(Box.createRigidArea(new Dimension(0, 50)));
    innerCheckPane.add(exitButton);
    // them o ben trong vao bang o chon noi
    checkPane.add(innerCheckPane);

    // them khung va doi tuong GUI vao noi dung cua khung
    Container content = getContentPane();
    content.setLayout(new GridLayout(1, 3, 2, 2));
    content.add(radioPane);
    content.add(checkPane);
}
```

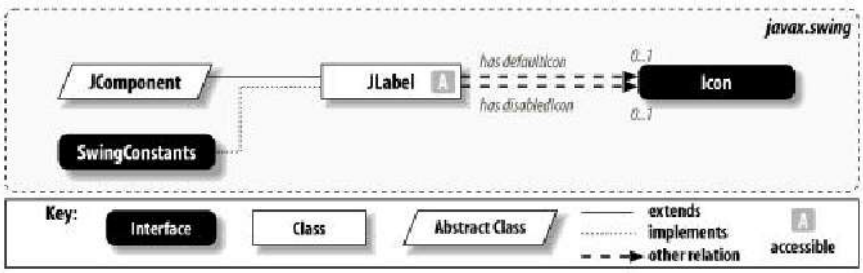
Ví dụ sử dụng JRadioButton & JCheckBox

```
    @Override
    public void actionPerformed(ActionEvent e) {
        Color c;
        if (redButton.isSelected())
            c = Color.red;
        else if (whiteButton.isSelected())
            c = Color.white;
        else
            c = Color.blue;
        colorBox.setBackground(c);
    }

    public static void main(String[] args) {
        ButtonDemo app = new ButtonDemo();
        app.setTitle("Button Demo");
        app.setSize(320, 220);
        app.setVisible(true);
    }
}
```

JLabel

- Cung cấp văn bản trên GUI
- Được định nghĩa với lớp JLabel
- Có thể trình bày :
  - Dòng văn bản chỉ đọc
  - Hình ảnh
  - Văn bản và hình ảnh

JLabel class diagram



Các thuộc tính của JLabel

Property	Data type	get	is	set	Default value
UI	LabelUI	•		•	From L&F
UIClassID[o]	String				"LabelUI"
accessibleContext[o]	AccessibleContext	•			JLabel.AccessibleJLabel
disabledIcon[b]	Icon	•		•	null
displayedMnemonic[b]	int	•		•	KeyEvent.VK_UNDEFINED
displayedMnemonicIndex[1.4, b]	int				-1
font[o]	Font	•		•	From L&F
horizontalAlignment[b]	int	•		•	LEADING[1.3]
horizontalTextPosition[b]	int	•		•	TRAILING[1.3]
icon[b]	Icon	•		•	null
iconTextGap[b]	int	•		•	4
labelFor[b]	Component	•		•	null
text[b]	String	•		•	null
verticalAlignment[b]	int	•		•	CENTER
verticalTextPosition[b]	int	•		•	CENTER

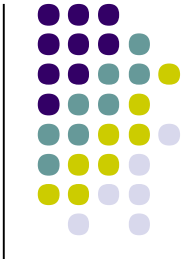
[1.3]since 1.3, [1.4]since 1.4, [b]bound, [o]overridden

Constructors

- JLabel( )
- JLabel(Icon image)
- JLabel(Icon image, int horizontalAlignment)
- JLabel(String text)
- JLabel(String text, int horizontalAlignment)
- JLabel(String text, Icon image, int horizontalAlignment)
  - The horizontal alignment: `LEADING`, `TRAILING`, `LEFT`, `RIGHT`, `CENTER`.

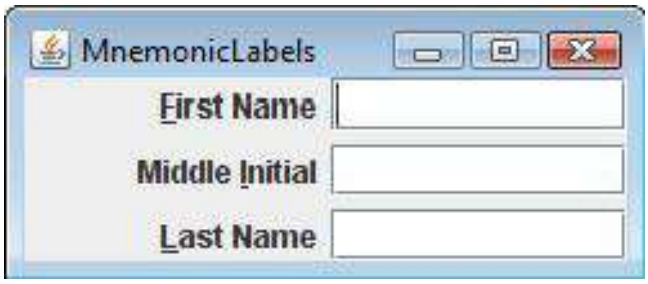
Ví dụ sử dụng JLabel

Ví dụ sử dụng JLabel



```
public class LabelDemo extends JFrame {
    public LabelDemo() {
        ...
        // sap xep cac thanh phan trong o
        JPanel pane = new JPanel();
        pane.setLayout(new GridLayout(3, 1, 2, 2));
        pane.setBorder(BorderFactory.createEmptyBorder(10, 10, 10, 10));
        // tao nhan don gian bang van ban mac dinh
        JLabel titleLabel = new JLabel("Three title labels");
        // them doi tuong hinh anh vao nhan
        ImageIcon mailIcon = new ImageIcon("email.jpg");
        JLabel mailLabel = new JLabel(mailIcon, JLabel.LEADING);
        mailLabel.setText("Your new mail");
        // dung HTML de dinh dang font va co chu cua nhan
        String s = "<html>"
                + "<font size = +2><b><i>Check this out!</i></b></font>"
                + "</html>";
        ImageIcon handIcon = new ImageIcon("hand.jpg");
        JLabel htmlLabel = new JLabel(s, handIcon, JLabel.CENTER);
        //them cac thanh phan vao o va them o vao lop noi dung
        pane.add(titleLabel);
        pane.add(mailLabel);
        pane.add(htmlLabel);
        getContentPane().add(pane);
    }
    ....
}
```

Ví dụ sử dụng JLabel



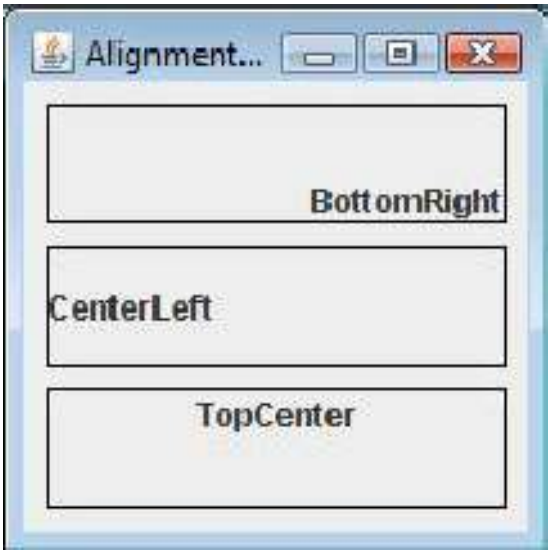
Ví dụ sử dụng JLabel

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

// Shows how displayedMnemonic and labelFor properties work together
public class MnemonicLabels {
    public static void main(String[] args) {
        JTextField firstField = new JTextField(10);
        JTextField middleField = new JTextField(10);
        JTextField lastField = new JTextField(10);

        // Create labels and mnemonics.
        JLabel firstLabel = new JLabel("First Name", JLabel.RIGHT);
        firstLabel.setDisplayedMnemonic('F');
        firstLabel.setLabelFor(firstField);

        JLabel middleLabel = new JLabel("Middle Initial", JLabel.RIGHT);
        middleLabel.setDisplayedMnemonic('I');
        middleLabel.setDisplayedMnemonicIndex(7); // Requires 1.4
        middleLabel.setLabelFor(middleField);

        JLabel lastLabel = new JLabel("Last Name", JLabel.RIGHT);
        lastLabel.setDisplayedMnemonic('L');
        lastLabel.setLabelFor(lastField);
```

Ví dụ sử dụng JLabel

```
        // Layout and display
        JPanel p = new JPanel();
        p.setLayout(new GridLayout(3, 2, 5, 5));
        p.add(firstLabel);
        p.add(firstField);
        p.add(middleLabel);
        p.add(middleField);
        p.add(lastLabel);
        p.add(lastField);

        JFrame f = new JFrame("MnemonicLabels");
        f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        f.setContentPane(p);
        f.pack();
        f.setVisible(true);
    }
}
```

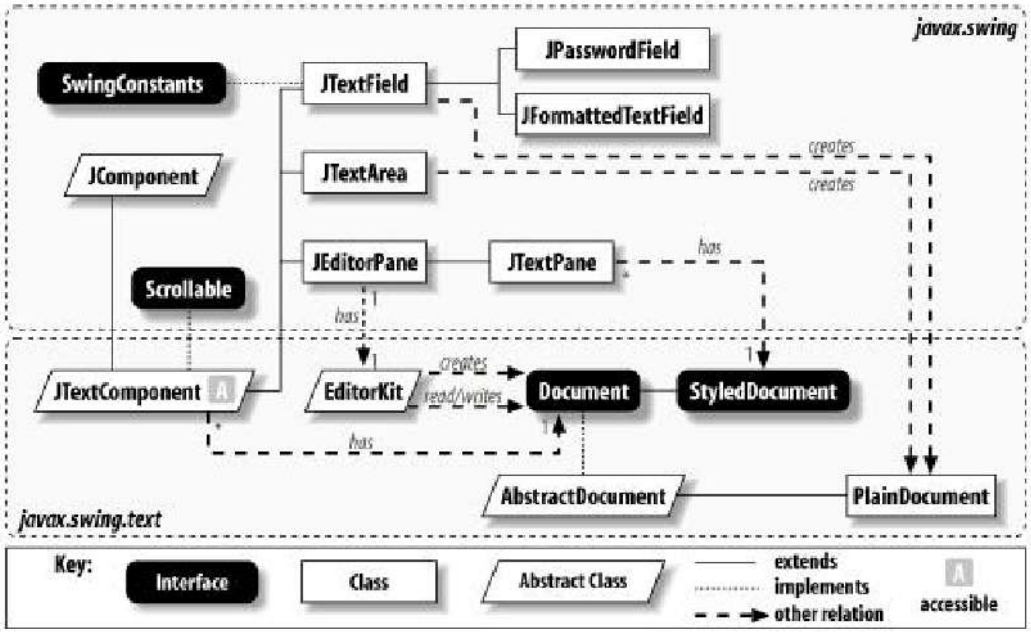
Canh lề - Alignment



Canh lề - Alignment

```
public class AlignmentExample {
    public static void main(String[] args) {
        // Create the labels and set alignment.
        JLabel label1 = new JLabel("BottomRight", SwingConstants.RIGHT);
        JLabel label2 = new JLabel("CenterLeft", SwingConstants.LEFT);
        JLabel label3 = new JLabel("TopCenter", SwingConstants.CENTER);
        label1.setVerticalAlignment(SwingConstants.BOTTOM);
        label2.setVerticalAlignment(SwingConstants.CENTER);
        label3.setVerticalAlignment(SwingConstants.TOP);

        // Add borders to the labels (more on Borders later in the book).
        label1.setBorder(BorderFactory.createLineBorder(Color.black));
        label2.setBorder(BorderFactory.createLineBorder(Color.black));
        label3.setBorder(BorderFactory.createLineBorder(Color.black));

        // Put it all together.
        JFrame frame = new JFrame("AlignmentExample");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        JPanel p = new JPanel(new GridLayout(3, 1, 8, 8));
        p.add(label1);
        p.add(label2);
        p.add(label3);
        p.setBorder(BorderFactory.createEmptyBorder(8, 8, 8, 8));
        frame.setContentPane(p);
        frame.setSize(200, 200);
        frame.setVisible(true);
    }
}
```

JTextComponent Class



Các thuộc tính

Property	Data type	get	is	set	Default value
accessibleContext°	AccessibleContext	·			AccessibleJTextComponent
actions	Action[]	·			From the UI's EditorKit
caret[b]	Caret	·		·	null
caretColor[b]	Color	·		·	null
caretPosition	int	·		·	From caret
disabledTextColor[b]	Color	·		·	null
document[b]	Document	·		·	null
dragEnabled[1.4]	boolean	·		·	false
editable[b]	boolean		·	·	true
focusAccelerator[b]	char	·		·	'\0'
highlighter[b]	Highlighter	·		·	null
keymap[b]	Keymap	·		·	null
layout°	LayoutManager	·		·	null
margin[b]	Insets	·		·	From UI
navigationFilter[1.4]	NavigationFilter	·		·	null
preferredScrollableViewportSize°	Dimension	·			Preferred size of component
scrollableTracksViewportHeight°	boolean	·			See below
scrollableTracksViewportWidth°	boolean	·			See below
selectedText	String	·			From document
selectedTextColor[b]	Color	·		·	null
selectionColor[b]	Color	·		·	null
selectionEnd	int	·		·	From caret
selectionStart	int	·		·	From caret
text	String	·		·	From document
UI[b]	TextUI	·		·	From L&F

[1.4]since 1.4, [b]bound, °overridden

Events, constants & constructor

- public void addCaretListener(CaretListener listener)
- public void removeCaretListener(CaretListener listener)
- public CaretListener[] getCaretListeners( )
- public JTextComponent( )

Constant	Type	Description
DEFAULT_KEYMAP	String	The name of the default keymap used by all text components. This string can be used as a parameter to the static `getKeymap( )` method.
`FOCUS_ACCELERATOR_KEY`	`String`	The bound property name for the focus accelerator, used when firing property change events.

Methods

- public void copy( )
- public void cut( )
- public void paste( )
- public void moveCaretPosition(int pos)
- public void replaceSelection(String content)
- public void select(int selectionStart, int selectionEnd)
- public void selectAll( )
- public int getScrollableUnitIncrement(Rectangle visibleRect, int orientation, int direction)
- public int getScrollableBlockIncrement(Rectangle visibleRect, int orientation, int direction)
- public Rectangle modelToView(int pos) throws BadLocationException
- public int viewToModel(Point pt)
- public String getText(int offset, int len) throws BadLocationException
- public void write(Writer out) throws IOException
- public void read(Reader in, Object desc) throws IOException
- public void updateUI( )

JTextField class

- Hộp văn bản trong đó người sử dụng có thể nhập dữ liệu từ bàn phím

Property	Data type	get	is	set	Default value
accessibleContext[o]	AccessibleContext	·			AccessibleJTextField
action[1.3]	Action	·		·	null
actionCommand	String			·	null
actions[o]	Action[]	·			From superclass plus NotifyAction
columns	int	·		·	0
document[b, o]	Document	·		·	PlainDocument( )
font[b, o]	Font	·		·	From superclass
horizontalAlignment[b]	int	·		·	LEADING
horizontalVisibility	BoundedRangeModel	·			DefaultBoundedRangeModel
preferredSize[b, o]	Dimension	·		·	Width based on columns and font
scrollOffset	int	·		·	From horizontal visibility
UIClassID[o]	String	·			"TextFieldUI"
validateRoot[o]	boolean		·		true

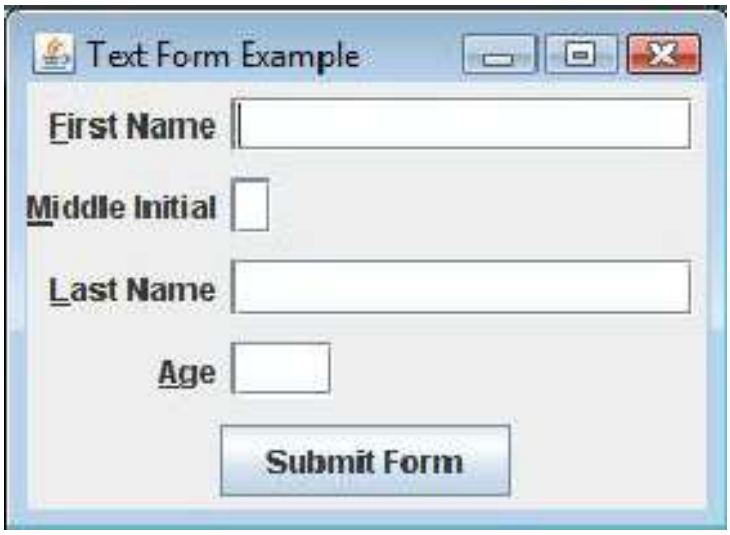
[1.3]since 1.3, [b]bound, [o]overridden

Events Constructor & Constant , methods

- public void addActionListener(ActionListener l)
- public void removeActionListener(ActionListener l)
- public ActionListener[] getActionListeners( )
- public JTextField( )
- public JTextField(String text)
- public JTextField(int columns)
- public JTextField(String text, int columns)
- public JTextField(Document doc, String text, int columns)
- public void scrollRectToVisible(Rectangle r)

Constant	Type	Description
notifyAction	String	The name of the action used to send notification that the field's contents have been accepted

Ví dụ sử dụng JTextField



Ví dụ sử dụng JTextField

```
public class TextForm extends JPanel {
    private JTextField[] fields;
    public TextForm(String[] labels, char[] mnemonics, int[] widths,
            String[] tips) {
        super(new BorderLayout());
        JPanel labelPanel = new JPanel(new GridLayout(labels.length, 1));
        JPanel fieldPanel = new JPanel(new GridLayout(labels.length, 1));
        add(labelPanel, BorderLayout.WEST);
        add(fieldPanel, BorderLayout.CENTER);
        fields = new JTextField[labels.length];

        for (int i = 0; i < labels.length; i += 1) {
            fields[i] = new JTextField();
            if (i < tips.length)
                fields[i].setToolTipText(tips[i]);
            if (i < widths.length)
                fields[i].setColumns(widths[i]);
            JLabel lab = new JLabel(labels[i], JLabel.RIGHT);
            lab.setLabelFor(fields[i]);
            if (i < mnemonics.length)
                lab.setDisplayedMnemonic(mnemonics[i]);
            labelPanel.add(lab);
            JPanel p = new JPanel(new FlowLayout(FlowLayout.LEFT));
            p.add(fields[i]);
            fieldPanel.add(p);
        }
    }
```

JPasswordField Class

- Mở rộng JTextField
- Che giấu các ký tự mà người sử dụng nhập vào

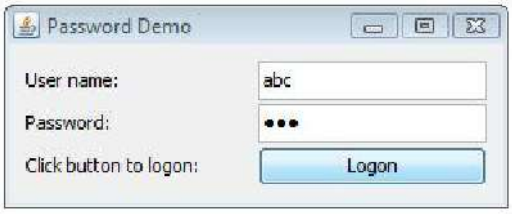
Property	Data type	get	is	set	Default value
accessibleContext°	AccessibleContext	·			AccessibleJPasswordField
echoChar	char	·		·	'*'
password	char[]	·			
text°	string	·		·	
UIClassID°	String	·			"PasswordFieldUI"

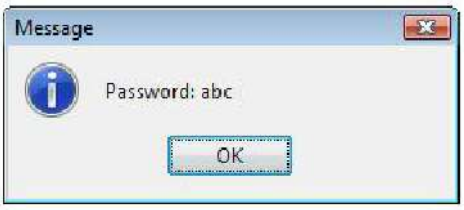
°overridden

Constructor & methods

- public JPasswordField( )
- public JPasswordField(String text)
- public JPasswordField(int columns)
- public JPasswordField(String text, int columns)
- public JPasswordField(Document doc, String text, int columns)
- public boolean echoCharIsSet( )
- Protected methods
  - public void cut( )
  - public void copy( )
  - public string getText(int offs, int len) throws BadLocationException

Ví dụ sử dụng JTextField & JPasswordField





Ví dụ sử dụng JTextField & JPasswordField

```
public class Password extends JFrame implements ActionListener {
    JTextField userName;
    JPasswordField password;
    JButton logon;

    public Password() {
        try {
            UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());
        } catch (Exception e) {
            e.printStackTrace();
        }
        addWindowListener(new WindowAdapter() {
            @Override
            public void windowClosing(WindowEvent e) {
                System.exit(0);
            }
        });
        JPanel pane = new JPanel();
        pane.setLayout(new GridLayout(3, 2, 2, 2));
        pane.setBorder(BorderFactory.createEmptyBorder(10, 10, 10, 10));
        userName = new JTextField(16);
        password = new JPasswordField(16);
        logon = new JButton("Logon");
        logon.addActionListener(this);

        // ngan nguoi dung dieu chinh kich thuoc frame
        setResizable(false);
```

Ví dụ sử dụng JTextField & JPasswordField

```
        // them cac thanh phan vao o va them o vao lop noi dung
        pane.add(new JLabel("User name:"));
        pane.add(userName);
        pane.add(new JLabel("Password:"));
        pane.add(password);
        pane.add(new JLabel("Click button to logon:"));
        pane.add(logon);
        getContentPane().add(pane);
    }
    @Override
    public void actionPerformed(ActionEvent e) {
        Object source = e.getSource();
        if (source.equals(logon)) {
            // bao mat gap nguy hiem
            char[] ptext = password.getPassword();
            String s = new String(ptext);
            JOptionPane.showMessageDialog(this, "Password: " + s);
            for (int i = 0; i < ptext.length; i++)
                ptext[i] = 0;
        }
    }
    public static void main(String[] args) {
        Password app = new Password();
        app.setTitle("Toogle Demo");
        app.pack();
        app.setVisible(true);
    }
}
```

JTextArea

- Vùng văn bản cho phép thao tác soạn thảo nhiều dòng văn bản.
- Sử dụng JScrollPane bên dưới

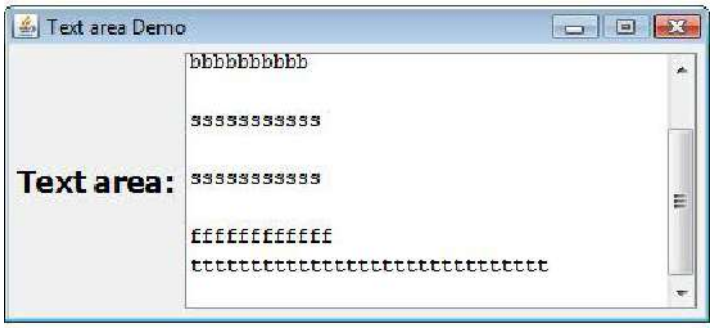
Property	Data type	get	is	set	Default value
accessibleContext[o]	AccessibleContext	·			AccessibleJTextArea
columns	int	·		·	0
font[b, o]	Font	·		·	From superclass
lineCount	int	·			From document
lineWrap[b]	boolean	·		·	false
preferredScrollableViewportSize[o]	Dimension	·			See comments below
preferredSize[b, o]	Dimension	·		·	See comments below
rows	int	·		·	0
scrollableTracksViewportWidth[o]	boolean	·			See comments below
tabSize[b]	int	·		·	8
UIClassID[o]	String	·			"TextAreaUI"
wrapStyleWord[b]	boolean	·		·	false

[b]bound, [o]overridden

Methods

- public JTextArea( )
- public JTextArea(int rows, int columns)
- public JTextArea(String text)
- public JTextArea(String text, int rows, int columns) public JTextArea(Document doc)
- public JTextArea(Document doc, String text, int rows, int columns)
- public void append(String str)
- public void insert(String str, int pos)
- public void replaceRange(String str, int start, int end)
- public int getLineStartOffset(int line) throws BadLocationException
- public int getLineEndOffset(int line) throws BadLocationException
- public int getLineOfOffset(int offset) throws BadLocationException

Ví dụ sử dụng JTextArea



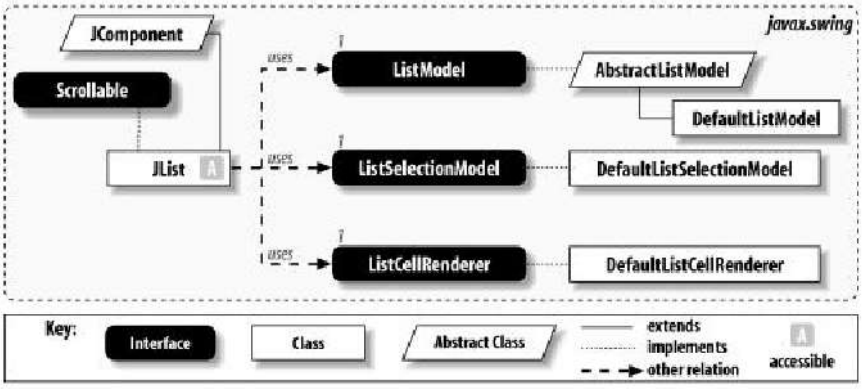
Ví dụ sử dụng JTextArea

```
public class TextDemo extends JFrame {
    public TextDemo() {
        ...
        JPanel pane = new JPanel();
        // tao doi tuong vung van ban
        JTextArea theText = new JTextArea();
        theText.setFont(new Font("Courier", Font.PLAIN, 12));
        theText.setLineWrap(false);
        // them thanh cuon vao vung van ban
        JScrollPane scroller = new JScrollPane(theText);
        scroller.setPreferredSize(new Dimension(300, 150));
        scroller
                .setVerticalScrollBarPolicy(JScrollPane.VERTICAL_SCROLLBAR_ALWAYS);
        // tao nhan cho doi tuong vung van ban
        String s = "<html>" + "<font size = +1><b>Text area:</b></font>"
                + "</html>";
        JLabel inputLabel = new JLabel(s);
        inputLabel.setLabelFor(theText);
        inputLabel.setAlignmentX(Component.LEFT_ALIGNMENT);
        // them tat ca cac thanh phan vao khung noi dung
        pane.add(inputLabel);
        pane.add(scroller);
        getContentPane().add(pane);
        setResizable(false);
    }
...
}
```

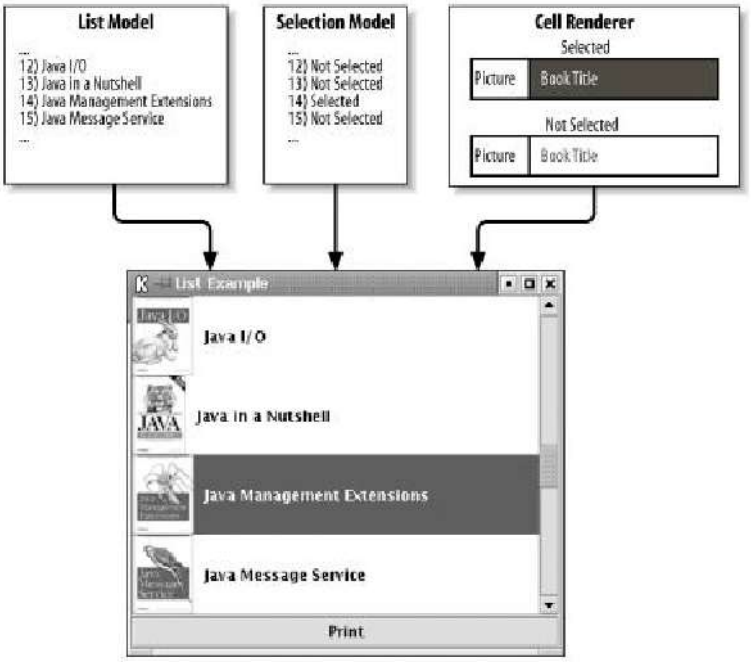
Danh sách

- Thường dùng để liệt kê
- Java co 2 loại danh sách cơ bản:
  - JList:
    - dùng cho danh sách có các khoản mục cố định
    - Người sử dụng có thể chọn một hoặc nhiều mục
  - JComboBox:
    - dùng cho danh sách mà người dùng có thể lựa chọn mục nhập hay

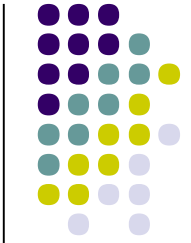
Swing list class diagram



The three parts of a Swing list



Hiển thị danh sách dữ liệu



- ListModel Interface
  - Thuộc tính

Property	Data type	get	is	set	Default value
elementAt[i]	Object	·			
size	int	·			
[i]indexed					

  - Events
    - public abstract void addListDataListener(ListDataListener l)
    - public abstract void removeListDataListener(ListDataListener l)

Hiển thị danh sách dữ liệu

- AbstractListModel Class
  - protected void fireContentsChanged(Object source, int index1, int index2)
  - protected void fireIntervalAdded(Object source, int index1, int index2)
  - protected void fireIntervalRemoved(Object source, int index1, int index2)
  - public EventListener[] getListeners(Class listenerType)
  - public ListDataListener[] getListDataListeners( )

Hiển thị danh sách dữ liệu

- DefaultListModel Class
  - Là hiện thực mặc định của AbstractListModel
  - Thuộc tính

Property	Data type	get	is	set	Default value
elementAt[i]	Object	·		·	
empty	boolean		·		true
size	int	·		·	0
[i]indexed					

  - Constructor
    - public DefaultListModel( )

Hiển thị danh sách dữ liệu

- Methods
  - public void copyInto(Object anArray[])
  - public void trimToSize( )
  - public void ensureCapacity(int minCapacity)
  - public int capacity( )
  - public int size( )
  - public Enumeration elements( )
  - public boolean contains(Object elem)
  - public int indexOf(Object elem)
  - public int indexOf(Object elem, int index)
  - public int lastIndexOf(Object elem)
  - public int lastIndexOf(Object elem, int index)
  - public Object elementAt(int index)
  - public Object firstElement( )
  - public Object lastElement( )

- Methods
  - public void removeElementAt(int index)
  - public void insertElementAt(Object obj, int index)
  - public void addElement(Object obj)
  - public boolean removeElement(Object obj)
  - public void removeAllElements( )
  - public String toString( )
  - public Object[] toArray( )
  - public Object get(int index)
  - public Object set(int index, Object element)
  - public void add(int index, Object element)
  - public Object remove(int index)
  - public void clear( )
  - public void removeRange(int fromIndex, int toIndex)

Hiển thị danh sách dữ liệu

- ListDataEvent là lớp mở rộng của java.util.EventObject nắm dữ thông tin về sự thay đổi dữ liệu trên danh sách
  - Sửa đổi các phần tử
  - Thêm phần tử
  - Xóa phần tử
  - Thuộc tính

Property	Data type	get	is	set
index0	int	·		
index1	int	·		
source°	Object	·		
type	int	·		
°overridden				

Hiển thị danh sách dữ liệu

- Constants

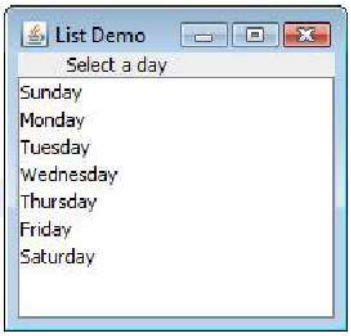
Constant	Data type	Description
CONTENTS_CHANGED	int	The elements between the two indices (inclusive) have been altered.
INTERVAL_ADDED	int	The elements now between the two indices (inclusive) have just been inserted into the list.
INTERVAL_REMOVED	int	The elements previously between the two indices (inclusive) have now been removed from the list.

- Constructor
  - public ListDataEvent(Object source, int type, int index0, int index1)
- Method
  - public String toString( )

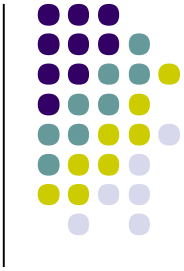
Quản lý lựa chọn

- ListDataListener Interface
  - ListSelectionModel Interface liệt kê các phương thức cần thiết để quản lý các lựa chọn

Ví dụ sử dụng JList



Ví dụ sử dụng JList

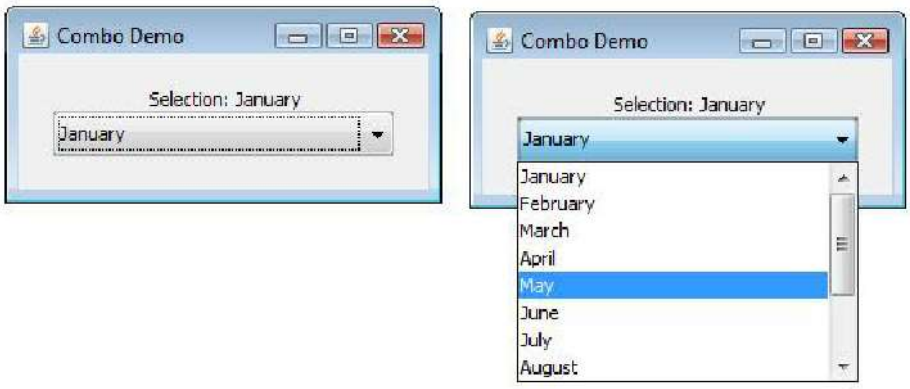


```
public class ListDemo extends JFrame {
    JFrame frame;// tham chieu cua so frame nay
    JLabel label;// cho biet lua chon hien hanh

    public ListDemo() {
        try {
            UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());
        } catch (Exception e) {
            e.printStackTrace();
        }
        addWindowListener(new WindowAdapter() {
            @Override
            public void windowClosing(WindowEvent e) {
                System.exit(0);
            }
        });
        frame = this;
        String[] items = { "Sunday", "Monday", "Tuesday", "Wednesday",
                "Thursday", "Friday", "Saturday" };
        JList dayList = new JList(items);
        dayList.setSelectionMode(ListSelectionModel.SINGLE_SELECTION);
        dayList.setAlignmentX(Component.CENTER_ALIGNMENT);

        JScrollPane listScroller = new JScrollPane(dayList);
```

Ví dụ sử dụng JList

```
        dayList.addListSelectionListener(new ListSelectionListener() {
            @Override
            public void valueChanged(ListSelectionEvent e) {
                JList list = (JList) e.getSource();
                if (!list.isSelectionEmpty()) {
                    int i = list.getSelectedIndex();
                    String s = (String) list.getModel().getElementAt(i);
                    label.setText("Selection: " + i);
                }
            }
        });
        label = new JLabel("Select a day");
        Container content = getContentPane();
        content.setLayout(new BoxLayout(content, BoxLayout.Y_AXIS));
        content.add(label);
        content.add(listScroller);
    }

    public static void main(String[] args) {
        ListDemo app = new ListDemo();
        app.setTitle("Toogle Demo");
        app.pack();
        app.setVisible(true);
    }
}
```

Ví dụ sử dụng JCompoBox



Ví dụ sử dụng JCompoBox

```
public class ComboDemo extends JFrame {
    JFrame frame;
    JLabel label;
    public ComboDemo() {
        ...
        frame = this;
        label = new JLabel("Select a month");
        String[] items = { "January", "February", "March", "April", "May",
                "June", "July", "August", "September", "October", "November", "December" };
        JComboBox months = new JComboBox(items);
        months.setSelectedIndex(0);
        months.setEditable(false);
        months.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                JComboBox box = (JComboBox) e.getSource();
                String s = (String) box.getSelectedItem();
                label.setText("Selection: " + s);
            }
        });
        JPanel pane = new JPanel();
        pane.setLayout(new BoxLayout(pane, BoxLayout.Y_AXIS));
        pane.setBorder(BorderFactory.createEmptyBorder(20, 20, 20, 20));
        pane.add(label);
        pane.add(months);
        getContentPane().add(pane, BorderLayout.CENTER);
    }
...
```

JColorChooser

- Giúp bạn dễ dàng lựa chọn các màu bằng cách nhấp vào các ô màu

Thuộc tính

Property	Data type	get	is	set	Default value
accessibleContext	AccessibleContext	·			JColorChooser.AccessibleJCol-orChooser( )
chooserPanels[b]	AbstractColorChooser-Panel[]	·		·	null
color	Color	·		·	Color.white
dragEnabled[1.4]	boolean	·		·	false
previewPanel[b]	JComponent	·		·	null
selectionModel[b]	ColorSelectionModel	·		·	DefaultColorSelectionModel
UI[b]	ColorChooserUI	·		·	From L&F
UIClassID[o]	String	·			"ColorChooserUI"

[1.4]since 1.4, [b]bound, [o]overridden

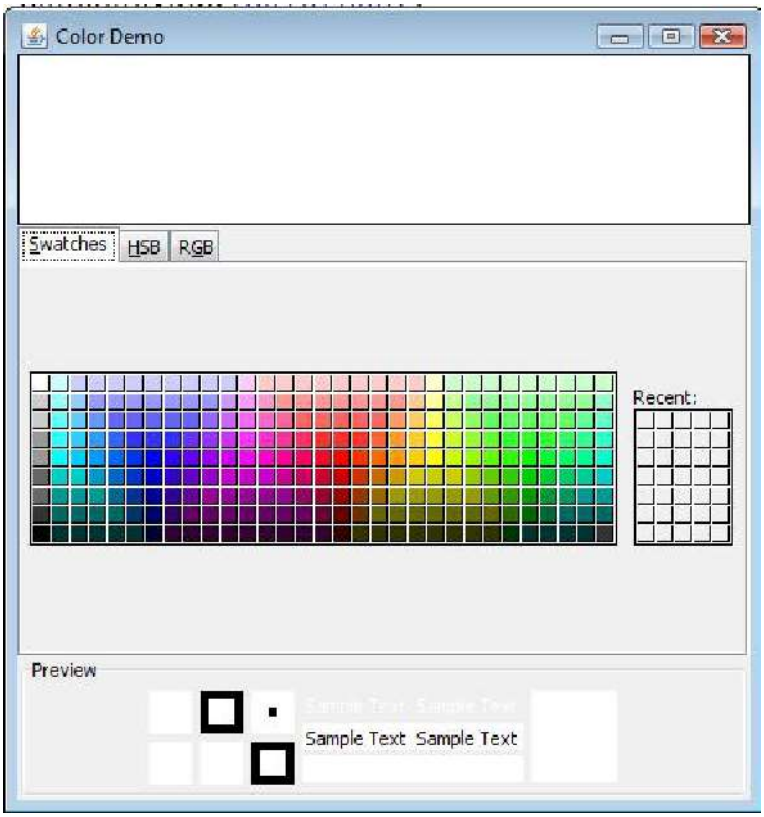
Constants for the property names

Constant	Type	Description
CHOOSER_PANELS_PROPERTY	String	The name of the `chooserPanels` property
PREVIEW_PANEL_PROPERTY	String	The name of the `previewPanel` property
SELECTION_MODEL_PROPERTY	String	The name of the `selectionModel` property

Constructors

- public JColorChooser( )
- public JColorChooser(Color initialColor)
- public JColorChooser(ColorSelectionModel model)

Ví dụ sử dụng JColorChooser

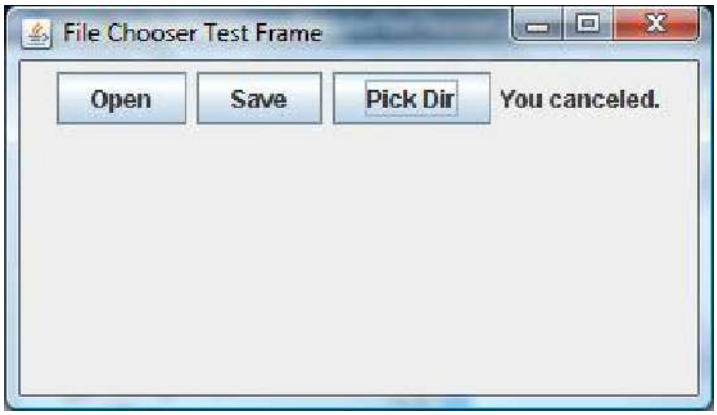


Ví dụ sử dụng JColorChooser

```
public class ColorDemo extends JFrame {
    JColorChooser colorChooser;// thanh phan chon mau
    JPanel colorBox;// mau mau
    Color selectedColor;// mau duoc chon
    public ColorDemo() {
        ...
        selectedColor = Color.white;
        colorBox = new JPanel();
        colorBox.setBackground(selectedColor);
        colorBox.setPreferredSize(new Dimension(150, 100));
        colorBox.setBorder(BorderFactory.createLineBorder(Color.black));
        // tao thanh phan chon mau va thay doi lop tiep nhan bien co
        colorChooser = new JColorChooser(selectedColor);
        colorChooser.getSelectionModel().addChangeListener(
                new ChangeListener() {
                    @Override
                    public void stateChanged(ChangeEvent e) {
                        selectedColor = colorChooser.getColor();
                        colorBox.setBackground(selectedColor);
                    }
                });
        // them cac thanh phan
        Container content = getContentPane();
        content.setLayout(new BoxLayout(content, BoxLayout.Y_AXIS));
        content.add(colorBox);
        content.add(colorChooser);
    }
...
```

JFileChooser Class

- Giúp bạn lựa chọn 1 file bất kì



```
public class SimpleFileChooser extends JFrame {
    public SimpleFileChooser() {
        super("File Chooser Test Frame");
        setSize(350, 200);
        setDefaultCloseOperation(EXIT_ON_CLOSE);
        Container c = getContentPane();
        c.setLayout(new FlowLayout());
        JButton openButton = new JButton("Open");
        JButton saveButton = new JButton("Save");
        JButton dirButton = new JButton("Pick Dir");
        final JLabel statusbar = new JLabel(
                "Output of your selection will go here");
        // Create a file chooser that opens up as an Open dialog.
        openButton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent ae) {
                JFileChooser chooser = new JFileChooser();
                chooser.setMultiSelectionEnabled(true);
                int option = chooser.showOpenDialog(SimpleFileChooser.this);
                if (option == JFileChooser.APPROVE_OPTION) {
                    File[] sf = chooser.getSelectedFiles();
                    String filelist = "nothing";
                    if (sf.length > 0)
                        filelist = sf[0].getName();
                    for (int i = 1; i < sf.length; i++)
                        filelist += ", " + sf[i].getName();
                    statusbar.setText("You chose " + filelist);
```

```
            } else
                statusbar.setText("You canceled.");
        }
    });

    // Create a file chooser that opens up as a Save dialog.
    saveButton.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent ae) {
            JFileChooser chooser = new JFileChooser();
            int option = chooser.showSaveDialog(SimpleFileChooser.this);
            if (option == JFileChooser.APPROVE_OPTION) {
                statusbar.setText("You saved "
                        + ((chooser.getSelectedFile() != null) ? chooser
                                .getSelectedFile().getName() : "nothing"));
            } else {
                statusbar.setText("You canceled.");
            }
        }
    });

    // Create a file chooser that allows you to pick a directory
    // rather than a file.
    dirButton.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent ae) {
            JFileChooser chooser = new JFileChooser();
            chooser.setFileSelectionMode(JFileChooser.DIRECTORIES_ONLY);
            int option = chooser.showOpenDialog(SimpleFileChooser.this);
            if (option == JFileChooser.APPROVE_OPTION) {
```

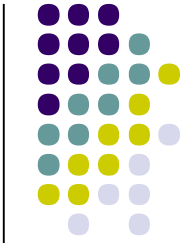
```
                if (option == JFileChooser.APPROVE_OPTION) {
                    statusbar.setText("You opened "
                            + ((chooser.getSelectedFile() != null) ? chooser
                                    .getSelectedFile().getName() : "nothing"));
                } else {
                    statusbar.setText("You canceled.");
                }
            }
        });

        c.add(openButton);
        c.add(saveButton);
        c.add(dirButton);
        c.add(statusbar);
    }

    public static void main(String args[]) {
        SimpleFileChooser sfc = new SimpleFileChooser();
        sfc.setVisible(true);
    }
}
```

Property	Data type	get	is	set	Default value
acceptAllFileFilter	FileFilter	·			From L&F
acceptAllFileFilterUsed[b, 1.3]	boolean		·	·	true
accessibleContext	AccessibleContext	·			FileChooser.accessibleJFileChooser( )
accessory[b]	JComponent	·		·	null
actionListeners[1.4]	ActionListener[]	·			Empty array
approveButtonMnemonic[b, #]	int	·		·	0 (no mnemonic)
approveButtonText[b]	String	·		·	null
approveButtonToolTipText[b]	String	·		·	null
choosableFileFilters[b, *]	FileFilter[]	·			{getAcceptAllFileFilter( )}
controlButtonsAreShown[b, 1.3]	boolean	·		·	true
currentDirectory[b]	File	·		·	User's home directory
dialogTitle	String	·		·	null
dialogType[b]	int	·		·	OPEN_DIALOG
directorySelectionEnabled[*]	boolean		·		false
dragEnabled[1.4]	boolean	·		·	false
fileFilter[b]	FileFilter	·		·	AcceptAllFileFilter( )
fileHidingEnabled[b]	boolean		·	·	true
fileSelectionEnabled+	boolean		·		true
fileSelectionMode[b]	int	·		·	FILES_ONLY
fileSystemView[b]	FileSystemView	·		·	FileSystemView.getFileSystemView( )
fileView[b]	FileView	·		·	null
multiSelectionEnabled[b, $]	boolean		·	·	false
selectedFile[b]	File	·		·	null
selectedFiles[b]	File[]	·		·	Empty array
UI[b]	SplitPaneUI	·		·	From L&F
UIClassID[o]	String	·			"FileChooserUI"

Events

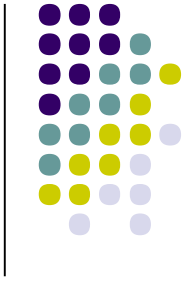
- public void addActionListener(ActionListener l)
- public void removeActionListener(ActionListener l)
- public void approveSelection( )
- public void cancelSelection( )
- protected void fireActionPerformed(String command)

Constant for change events



Constant	Type	Description
ACCEPT_ALL_FILE_FILTER_USED_CHANGED_PROPERTY	String	The name used for the `acceptAllFileFilterUsed` property
ACCESSORY_CHANGED_PROPERTY	String	The name used for the `accessory` property
APPROVE_BUTTON_MNEMONIC_CHANGED_PROPERTY	String	The name used for the `approveButtonMnemonic` property
APPROVE_BUTTON_TEXT_CHANGED_PROPERTY	String	The name used for the `approveButtonText` property
APPROVE_BUTTON_TOOL_TIP_TEXT_CHANGED_PROPERTY	String	The name used for the `approveButtonToolTipText` property
CHOOSABLE_FILE_FILTER_CHANGED_PROPERTY	String	The name used for the `choosableFileFilters` property
CONTROL_BUTTONS_ARE_SHOWN_CHANGED_PROPERTY	String	The name used for the `controlButtonsAreShown` property
DIALOG_TITLE_CHANGED_PROPERTY	String	The name used for the `dialogTitle` property
DIALOG_TYPE_CHANGED_PROPERTY	String	The name used for the `dialogType` property
DIRECTORY_CHANGED_PROPERTY	String	The name used for the `currentDirectory` property
FILE_FILTER_CHANGED_PROPERTY	String	The name used for the `fileFilter` property
FILE_HIDING_CHANGED_PROPERTY	String	The name used for the `fileHidingEnabled` property
FILE_SELECTION_MODE_CHANGED_PROPERTY	String	The name used for the `fileSelectionMode` property
FILE_SYSTEM_VIEW_CHANGED_PROPERTY	String	The name used for the `fileSystemView` property
FILE_VIEW_CHANGED_PROPERTY	String	The name used for the `fileView` property
MULTI_SELECTION_ENABLED_CHANGED_PROPERTY	String	The name used for the `multiSelectionEnabled` property
SELECTED_FILE_CHANGED_PROPERTY	String	The name used for the `selectedFile` property
SELECTED_FILES_CHANGED_PROPERTY	String	The name used for the `selectedFiles` property

Constant dialog

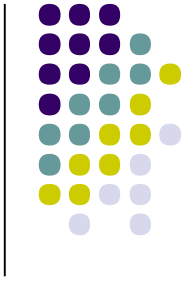


Constant	Type	Description
APPROVE_OPTION	int	The return value from the `showDialog( )` methods, indicating that the user selected the approve option
APPROVE_SELECTION	String	The string to be used for the `actionCommand` property of the `ActionEvent` generated when the user approves the current selection
CANCEL_OPTION	int	The return value from the `showDialog( )` methods, indicating that the user selected the cancel option
CANCEL_SELECTION	String	The string to be used for the `actionCommand` property of the `ActionEvent` generated when the user cancels the current selection
CUSTOM_DIALOG	String	A valid option for the `dialogType` property, indicating that this dialog supports a user-defined operation
DIRECTORIES_ONLY	int	A valid option for the `fileSelectionMode` property, indicating that only directories can be selected
ERROR_OPTION	int	The return value from the `showDialog( )` methods, indicating that an error occurred
FILES_AND_DIRECTORIES	int	A valid option for the `fileSelectionMode` property, indicating that both files and directories can be selected
FILES_ONLY	int	A valid option for the `fileSelectionMode` property, indicating that only files can be selected
OPEN_DIALOG	int	A valid option for the `dialogType` property, indicating that this dialog is selecting files to be opened
SAVE_DIALOG	int	A valid option for the `dialogType` property, indicating that this dialog is selecting a file to be saved

Constructors

- public JFileChooser( )
- public JFileChooser(File currentDirectory)
- public JFileChooser(String currentDirectoryPath)
- public JFileChooser(FileSystemView fsv)
- public JFileChooser(File currentDirectory, FileSystemView fsv)
- public JFileChooser(String currentDirectoryPath, FileSystemView fsv)

Other methods



- public void addChoosableFileFilter(FileFilter filter)
- public void removeChoosableFileFilter(FileFilter filter)
- public void resetChoosableFileFilters( )
- public boolean accept(File f)
- public void changeToParentDirectory( )
- public void ensureFileIsVisible(File f)
- public String getDescription(File f)
- public Icon getIcon(File f)
- public String getName(File f)
- public String getTypeDescription(File f)
- public boolean isTraversable(File f)
- public void rescanCurrentDirectory( )
- protected JDialog createDialog(Component parent) throws HeadlessException
- public int showDialog(Component parent, String approveButtonText)
- public int showOpenDialog(Component parent)
- public int showSaveDialog(Component parent)

Spinners

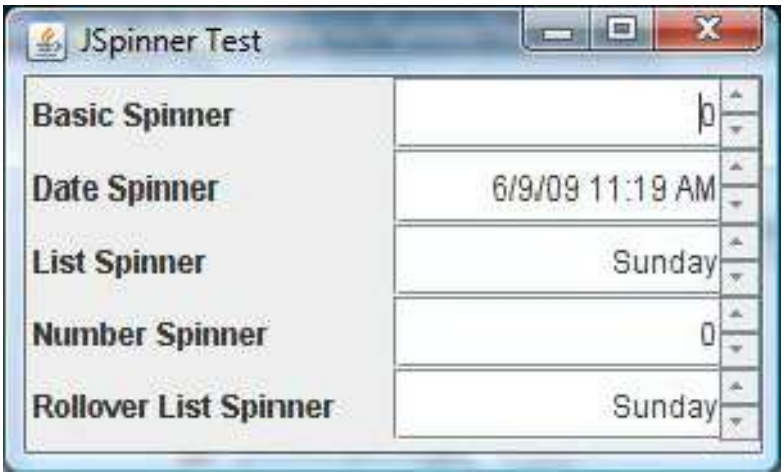
Property	Data type	get	is	set	Default value
changeListeners	ChangeListener[]	·			Empty array
editor[b]	JComponent	·		·	JPspinner.NumberEditor( )
model[b]	SpinnerModel	·		·	SpinnerNumberModel( )
nextValue	Object	·			
previousValue	Object	·			
UI	SpinnerUI	·			L&F-dependent
UIClassID	String	·			"SpinnerUI"
value	Object	·		·	

[b]bound

Events & Constructors

- public void addChangeListener(ChangeListener l)
- public void removeChangeListener(ChangeListener l)
- public JSpinner( )
- public JSpinner(SpinnerModel model)
- protected JComponent createEditor(SpinnerModel model)
- public void commitEdit( )

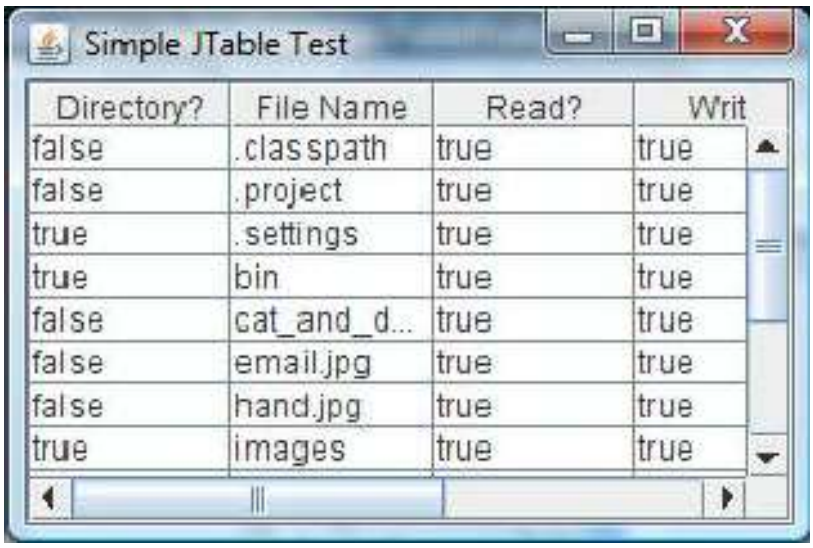
Ví dụ sử dụng Spinner



Ví dụ sử dụng Spinner

```
public SpinnerTest() {
    super("JSpinner Test");
    setSize(300, 180);
    setDefaultCloseOperation(EXIT_ON_CLOSE);

    Container c = getContentPane();
    c.setLayout(new GridLayout(0, 2));

    c.add(new JLabel(" Basic Spinner"));
    c.add(new JSpinner());

    c.add(new JLabel(" Date Spinner"));
    c.add(new JSpinner(new SpinnerDateModel()));

    String weekdays[] = new String[] { "Sunday", "Monday", "Tuesday",
            "Wednesday", "Thursday", "Friday", "Saturday" };
    c.add(new JLabel(" List Spinner"));
    c.add(new JSpinner(new SpinnerListModel(weekdays)));

    c.add(new JLabel(" Number Spinner"));
    c.add(new JSpinner(new SpinnerNumberModel(0, 0, 100, 5)));

    c.add(new JLabel(" Rollover List Spinner"));
    c.add(new JSpinner(new RolloverSpinnerListModel(weekdays)));

    setVisible(true);
}
```

JTable Class



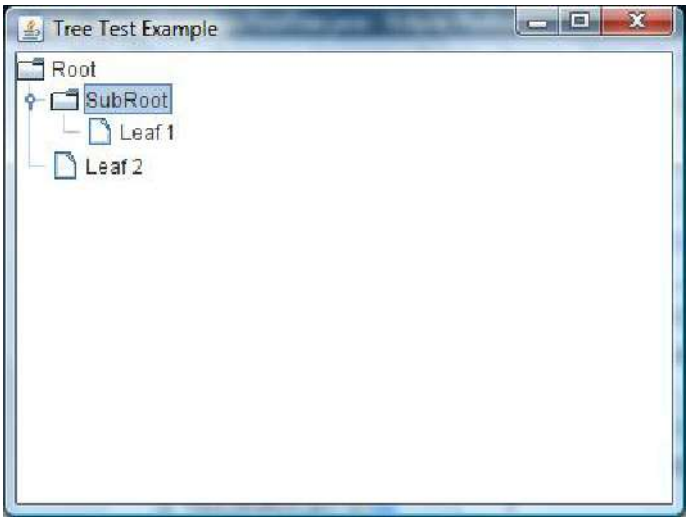
Ví dụ sử dụng JTable

```
public class TableFeature extends JFrame {

    String titles[] = new String[] { "Directory?", "File Name", "Read?",
            "Write?", "Size", "Last Modified" };

    public TableFeature() {
        super("Simple JTable Test");
        setSize(300, 200);
        setDefaultCloseOperation(EXIT_ON_CLOSE);

        File pwd = new File(".");
        Object[][] stats = getFileStats(pwd);

        JTable jt = new JTable(stats, titles);
        jt.setAutoResizeMode(JTable.AUTO_RESIZE_OFF);
        jt.setColumnSelectionAllowed(true);

        JScrollPane jsp = new JScrollPane(jt);
        getContentPane().add(jsp, BorderLayout.CENTER);
    }

    public Object[][] getFileStats(File dir) {
        String files[] = dir.list();
        Object[][] results = new Object[files.length][titles.length];
```

Ví dụ sử dụng JTable

```
        for (int i = 0; i < files.length; i++) {
            File tmp = new File(files[i]);
            results[i][0] = new Boolean(tmp.isDirectory());
            results[i][1] = tmp.getName();
            results[i][2] = new Boolean(tmp.canRead());
            results[i][3] = new Boolean(tmp.canWrite());
            results[i][4] = new Long(tmp.length());
            results[i][5] = new Date(tmp.lastModified());
        }
        return results;
    }

    public static void main(String args[]) {
        TableFeature tf = new TableFeature();
        tf.setVisible(true);
    }
}
```

JTree

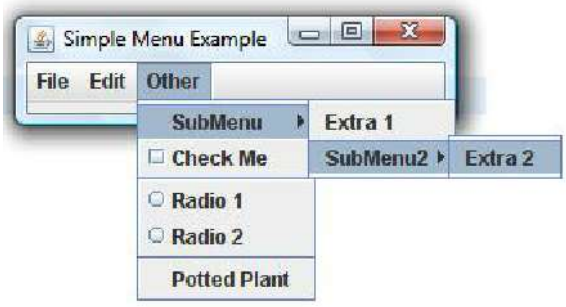


```
public class TestTree extends JFrame {
    JTree tree;
    DefaultTreeModel treeModel;
    public TestTree() {
        super("Tree Test Example");
        setSize(400, 300);
        setDefaultCloseOperation(EXIT_ON_CLOSE);
    }
    public void init() {
        // Build up a bunch of TreeNodes. We use DefaultMutableTreeNode because
        // the DefaultTreeModel can use it to build a complete tree.
        DefaultMutableTreeNode root = new DefaultMutableTreeNode("Root");
        DefaultMutableTreeNode subroot = new DefaultMutableTreeNode("SubRoot");
        DefaultMutableTreeNode leaf1 = new DefaultMutableTreeNode("Leaf 1");
        DefaultMutableTreeNode leaf2 = new DefaultMutableTreeNode("Leaf 2");
        // Build our tree model starting at the root node, and then make a JTree
        // out of it.
        treeModel = new DefaultTreeModel(root);
        tree = new JTree(treeModel);
        // Build the tree up from the nodes we created.
        treeModel.insertNodeInto(subroot, root, 0);
        // Or, more succinctly:
        subroot.add(leaf1);
        root.add(leaf2);
        // Display it.
        getContentPane().add(tree, BorderLayout.CENTER);
    }
```

JMenu

- JMenu có thể được dùng giống như một layout dùng để quản lý các Component
- Chú ý:
  - Chỉ được phép có 1 lựa chọn tại 1 thời điểm
  - Icon có thể dùng để thay thế cho các menu items
  - Hầu hết các component chuẩn đều có thể là Menu Item (radio button…)
  - Có thể gán phím tắt cho các Menu Item

Ví dụ về JMenu



SingleSelectionModel Interface

- Có chứa 1 mảng các lựa chọn có thể, và tại mỗi thời điểm chỉ được chọn duy nhất 1 lựa chọn.
- SingleSelectionModel sẽ nắm giữ vị trí của chọn lựa hiện tại, nếu có sự thay đổi ChangeEvent sẽ được bắt.

Thuộc tính, sự kiện, phương thức

- Các thuộc tính của lớp SingleSelectionModel

Property	Data type	get	is	set
selected	boolean		·	
selectedIndex	int	·		·

- Events: ChangeEvent
  - void addChangeListener(ChangeListener listener)
  - void removeChangeListener(ChangeListener listener)
- Methods
  - public void clearSelection( )

DefaultSingleSelectionModel Class

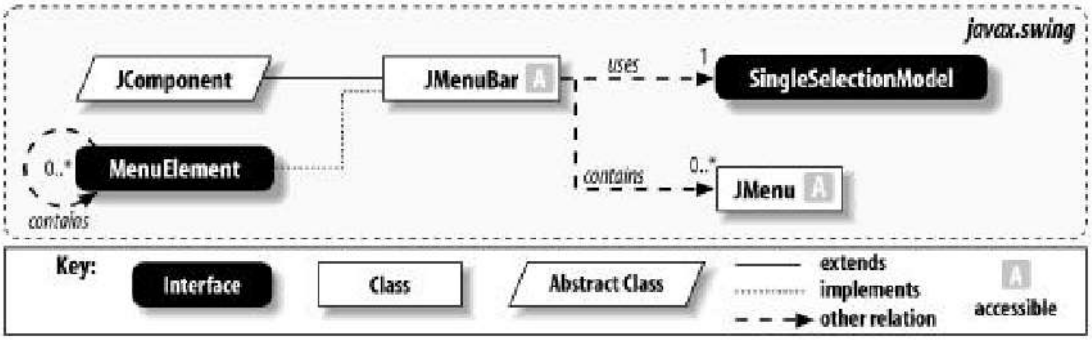
- Là một hiện thực của interface SingleSelectionModel

Property	Data type	get	is	set	Default value
selected	boolean		·		false
selectedIndex	int	·		·	-1

- Các phương thức và sự kiện giống với lớp cha

JMenuBar Class

- Dùng để tạo ra 1 Menu bar theo chiều ngang của component với 0, 1 hoặc nhiều phần tử gắn lên đó.



Chú ý

- Bạn sẽ dùng phương thức add để thêm vào các JMenu trên JMenuBar
- JMenubar sẽ hiển thị các JMenu theo thứ tự từ trái sang phải.
- Dùng sự kiện ActionListener
- Bạn có thể gắn 1 menu bar lên frame theo 1 trong 2 cách sau:
  - Sử dụng setJMenuBar( )

```
JFrame frame = new JFrame("Menu");
JMenuBar menuBar = new JMenuBar( );
frame.setJMenuBar(menuBar);
```

  - Dùng layout để định vị

```
menuBar.setBorder(new BevelBorder(BevelBorder.RAISED));
frame.getContentPane( ).add(menuBar, BorderLayout.SOUTH);
```

Thuộc tính, Constructor

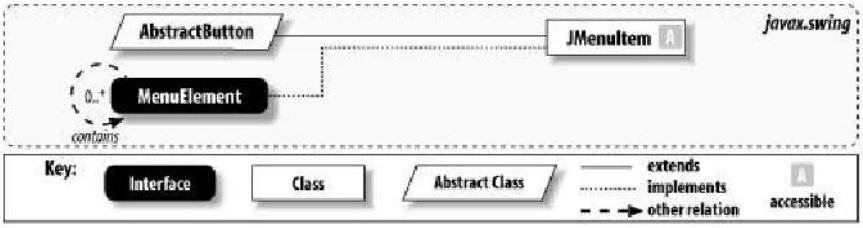
Property	Data type	get	is	set	Default value
accessibleContext[o]	AccessibleContext	·			JMenuBar.AccessibleJMenuBar( )
borderPainted[b]	boolean		·	·	true
component	Component	·			this
componentAtIndex[i]	Component	·			true
helpMenu[u]	JMenu	·		·	Throws an Error
layout[o]	LayoutManager	·		·	BoxLayout(X_AXIS)
margin[b]	Insets	·		·	null
menuCount	int	·			0
menu[i]	JMenu	·			null
selected	boolean		·		false
selectionModel[b]	SingleSelectionModel	·		·	DefaultSingleSelectionModel( )
subElements	MenuElement[]	·			
UI[b]	MenuBarUI	·		·	From L&F
UIClassID[o]	String	·			"MenuBarUI"

[b]bound, [i]indexed, [o]overridden, [u]unimplemented

public JMenuBar( )

JMenuItem Class

- Cung cấp nhiều lớp bao dùng để chuyển 1 String hay 1 icon thành 1 JMenuItem
- JMenuItem là 1 loại nút đặc biệt(xử lý mouseListenner)



- Bạn có thể cung cấp Keyboard accelerators hay mnemonics cho JMenuItem

Các thuộc tính trên JMenuItem

Property	Data type	get	is	set	Default value
accelerator[b]	KeyStroke	·		·	null
accessibleContext[o]	Accessible Context	·			JMenuItem.AccessibleJMenuItem( )
armed[b, o]	boolean		·	·	false
borderPainted[o]	boolean		·	·	false
component[o]	Component	·			
enabled[o]	boolean		·	·	true
focusPainted[o]	boolean		·	·	false
horizontalAlignment[o]	int	·		·	JButton.LEFT
horizontalTextPosition[o]	int	·		·	JButton.LEFT
menuDragMouseListeners[1.4]	MenuDragMouseListener[]	·			
menuKeyListeners[1.4]	MenuKeyListener[]	·			
model[o]	ButtonModel	·		·	DefaultButtonModel( )
subElements[o]	MenuElement[]	·			
UI[b]	MenuItemUI			·	From L&F
UIClassID[o]	String	·			"MenuItemUI"

[1.4]since 1.4, [b]bound, [o]overridden

Constructors

- JMenuItem( )
- JMenuItem(Action action)
- JMenuItem(Icon icon)
- JMenuItem(String string)
- JMenuItem(String string, Icon icon)
- JMenuItem(String string, int mnemonic)

Events

- ActionEvents, ChangeEvents, MenuDragMouseEvent, MenuKeyEvent
  - *addMenuDragMouseListener (MenuDragMouseListener 1)*
  - *removeMenuDragMouseListener (MenuDragMouseListener 1)*
  - *public void processMenuDragMouseEvent (MenuDragMouseEvent e)*
  - *addMenuKeyListener (MenuKeyListener 1)*
  - *removeMenuKeyListener (MenuKeyListener 1)*
  - *public void processMenuKeyEvent (MenuKeyEvent e)*

Ví dụ sử dụng JMenuBar

```
public class IntroExample extends JMenuBar {

    String[] fileItems = new String[] { "New", "Open", "Save", "Exit" };
    String[] editItems = new String[] { "Undo", "Cut", "Copy", "Paste" };
    char[] fileShortcuts = { 'N', 'O', 'S', 'X' };
    char[] editShortcuts = { 'Z', 'X', 'C', 'V' };

    public IntroExample() {

        JMenu fileMenu = new JMenu("File");
        JMenu editMenu = new JMenu("Edit");
        JMenu otherMenu = new JMenu("Other");
        JMenu subMenu = new JMenu("SubMenu");
        JMenu subMenu2 = new JMenu("SubMenu2");

        // Assemble the File menus with mnemonics.
        ActionListener printListener = new ActionListener() {
            public void actionPerformed(ActionEvent event) {
                System.out.println("Menu item [" + event.getActionCommand()
                        + "] was pressed.");
            }
        };
        for (int i = 0; i < fileItems.length; i++) {
            JMenuItem item = new JMenuItem(fileItems[i], fileShortcuts[i]);
            item.addActionListener(printListener);
            fileMenu.add(item);
        }
```

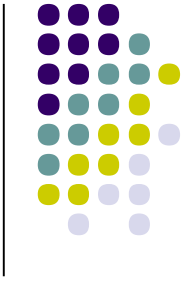
Ví dụ sử dụng JMenuBar

```
// Assemble the File menus with keyboard accelerators.
for (int i = 0; i < editItems.length; i++) {
    JMenuItem item = new JMenuItem(editItems[i]);
    item.setAccelerator(KeyStroke
            .getKeyStroke(editShortcuts[i], Toolkit.getDefaultToolkit()
                    .getMenuShortcutKeyMask(), false));
    item.addActionListener(printListener);
    editMenu.add(item);
}
// Insert a separator in the Edit menu in Position 1 after "Undo".
editMenu.insertSeparator(1);
// Assemble the submenus of the Other menu.
JMenuItem item;
subMenu2.add(item = new JMenuItem("Extra 2"));
item.addActionListener(printListener);
subMenu.add(item = new JMenuItem("Extra 1"));
item.addActionListener(printListener);
subMenu.add(subMenu2);
// Assemble the Other menu itself.
otherMenu.add(subMenu);
otherMenu.add(item = new JCheckBoxMenuItem("Check Me"));
item.addActionListener(printListener);
otherMenu.addSeparator();
```

Ví dụ sử dụng JMenuBar

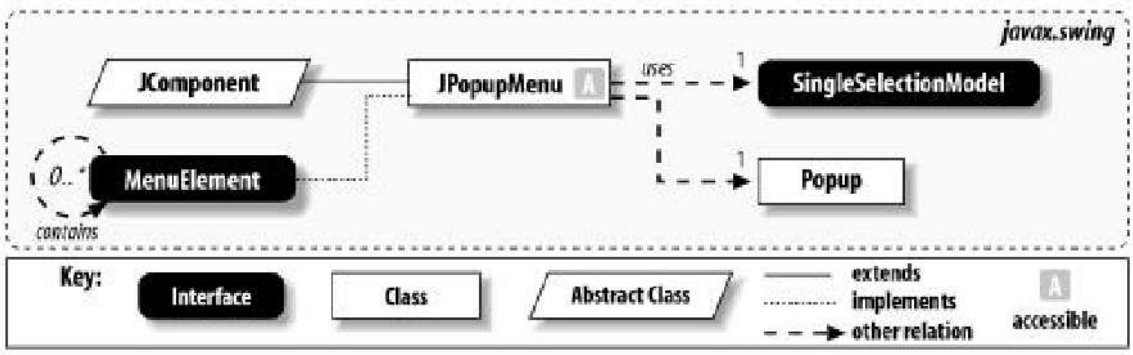
```
        ButtonGroup buttonGroup = new ButtonGroup();
        otherMenu.add(item = new JRadioButtonMenuItem("Radio 1"));
        item.addActionListener(printListener);
        buttonGroup.add(item);
        otherMenu.add(item = new JRadioButtonMenuItem("Radio 2"));
        item.addActionListener(printListener);
        buttonGroup.add(item);
        otherMenu.addSeparator();
        otherMenu.add(item = new JMenuItem("Potted Plant", new ImageIcon(
                "image.gif")));
        item.addActionListener(printListener);

        // Finally, add all the menus to the menu bar.
        add(fileMenu);
        add(editMenu);
        add(otherMenu);
    }

    public static void main(String s[]) {
        JFrame frame = new JFrame("Simple Menu Example");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setJMenuBar(new IntroExample());
        frame.pack();
        frame.setVisible(true);
    }
}
```

JPopupMenu Class



- Là một loại menu đặc biệt mà không cần phải gắn vào menu bar
- Nó có thể được hiển thị ở bất cứ vị trí nào trên thành phần chứa.
- Bạn có thể thêm, chèn một JMenuItem, 1 component hay 1 Action tùy ý vào popup menu này với phương thức add() và insert()
- JPopupMenu sẽ gán cho mỗi menu item 1 số thứ tự rồi gắn chúng vào popup menu theo layout mà nó có
- Bạn cũng có thể thêm 1 separator vào popup menu với phương thức addSeparator()

Class diagram



Hiển thị popup menu

- Bạn có thể hiển thị popup menu bằng phương thức show()

```
public void processMouseEvent(MouseEvent e) {
        if (e.isPopupTrigger( )) {
                popup.show(this, e.getX( ), e.getY( ));
        } else {
                super.processMouseEvent(e);
        }
}
```

Các thuộc tính

Property	Data type	get	is	set	Default value
accessibleContext°	AccessibleContext	·			JPopupMenu.accessibleJPopupMenu( )
borderPainted	boolean		·	·	true
component	Component	·			
componentAtIndex[i]	Component	·			
invoker	Component	·		·	
label[b]	String	·		·	""
layout°	LayoutManager	·		·	GridBagLayout( )
lightWeightPopupEnabled	boolean		·	·	getDefaultLightWeightPop-upEnabled( )
location°	Point			·	
margin	Insets	·			
popupMenuListeners[1.4]	PopupMenuListener[]	·			
popupSize	Dimension			·	
selectionModel	SingleSelectionModel	·			DefaultSingleSelectionMo-del( )
subElements	MenuElement[]	·			
UI[b]	PopupMenuUI	·		·	BasicPopupMenuUI( )
UIClassID°	String	·			"PopupMenuUI"
visible[b, °]	boolean		·	·	false

[1.4]since 1.4, [b]bound, [i]indexed, °overridden

Events

- Sử dụng PopupMenuEvent trong 2 trường hợp sau:
  - Menu hiện ⇔ ẩn
  - Loại bỏ 1 lựa chọn
  - public void addPopupMenuListener(PopupMenuListener l)
  - public void removePopupMenuListener(PopupMenuListener l)

Constructor, MenuItem

- public JPopupMenu( )
- public JPopupMenu(String title)
- public JMenuItem add(JMenuItem menuItem)
- public Component add(Component c)
- public JMenuItem add(Action a)
- public JMenuItem insert(Action a, int index)
- public Component insert(Component component, int index)
- public void addSeparator( )

Một số phương thức khác

- public void show(Component invoker, int x, int y)
- public void setPopupSize(int width, int height)
- public int getComponentIndex(Component c)
- public static boolean getDefaultLightWeightEnabled ()
- public boolean isPopupTrigger(MouseEvent e)
- public static void setDefaultLightWeightPopupEnabled(boolean aFlag)
- public void setSelected(Component c)
- public void updateUI( )

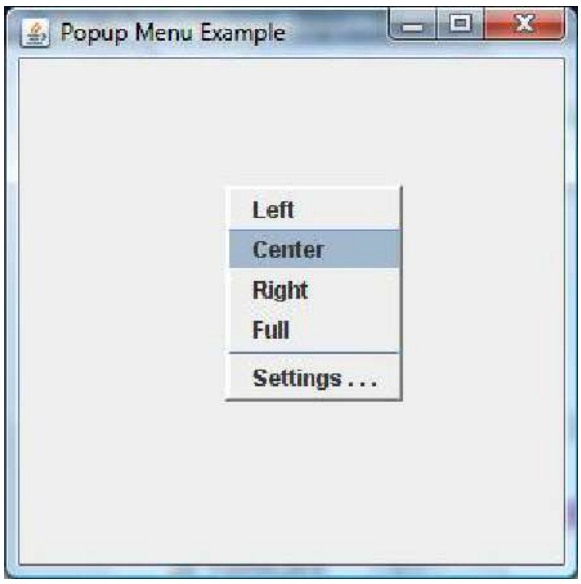
PopupMenuEvent Class

- Dùng để báo cho các listener về việc ẩn, hiện hay hủy bỏ popup menu
- Constructor
  - public PopupMenuEvent(Object source)

PopupMenuListener Interface

- Là nơi tiếp nhận PopupMenuEvent
  - public abstract void popupMenuCanceled(PopupMenuEvent e)
  - public abstract void popupMenuWillBecomeInvisible(PopupMenuEvent e)
  - public abstract void popupMenuWillBecomeVisible(PopupMenuEvent e)

Ví dụ sử dụng JPopupMenu



Ví dụ sử dụng JPopupMenu

```
public class PopupMenuExample extends JPanel {
    public JPopupMenu popup;

    public PopupMenuExample() {
        popup = new JPopupMenu();
        ActionListener menuListener = new ActionListener() {
            public void actionPerformed(ActionEvent event) {
                System.out.println("Popup menu item ["
                        + event.getActionCommand() + "] was pressed.");
            }
        };
        JMenuItem item;
        popup.add(item = new JMenuItem("Left", new ImageIcon("left.gif")));
        item.setHorizontalTextPosition(JMenuItem.RIGHT);
        item.addActionListener(menuListener);
        popup.add(item = new JMenuItem("Center", new ImageIcon("center.gif")));
        item.setHorizontalTextPosition(JMenuItem.RIGHT);
        item.addActionListener(menuListener);
        popup.add(item = new JMenuItem("Right", new ImageIcon("right.gif")));
        item.setHorizontalTextPosition(JMenuItem.RIGHT);
        item.addActionListener(menuListener);
        popup.add(item = new JMenuItem("Full", new ImageIcon("full.gif")));
        item.setHorizontalTextPosition(JMenuItem.RIGHT);
        item.addActionListener(menuListener);
        popup.addSeparator();
        popup.add(item = new JMenuItem("Settings . . ."));
        item.addActionListener(menuListener);
```

Ví dụ sử dụng JPopupMenu

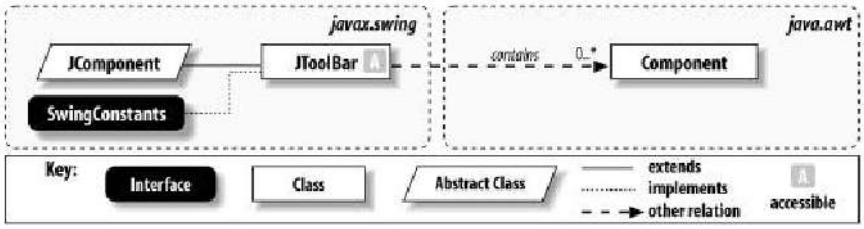
```
        popup.setLabel("Justification");
        popup.setBorder(new BevelBorder(BevelBorder.RAISED));
        popup.addPopupMenuListener(new PopupPrintListener());

        addMouseListener(new MousePopupListener());
    }

    // An inner class to check whether mouse events are the pop-up trigger
    class MousePopupListener extends MouseAdapter {
        public void mousePressed(MouseEvent e) {
            checkPopup(e);
        }

        public void mouseClicked(MouseEvent e) {
            checkPopup(e);
        }

        public void mouseReleased(MouseEvent e) {
            checkPopup(e);
        }

        private void checkPopup(MouseEvent e) {
            if (e.isPopupTrigger()) {
                popup.show(PopupMenuExample.this, e.getX(), e.getY());
            }
        }
    }
```

Ví dụ sử dụng JPopupMenu

```
    // An inner class to show when pop-up events occur
    class PopupPrintListener implements PopupMenuListener {
        public void popupMenuWillBecomeVisible(PopupMenuEvent e) {
            System.out.println("Popup menu will be visible!");
        }

        public void popupMenuWillBecomeInvisible(PopupMenuEvent e) {
            System.out.println("Popup menu will be invisible!");
        }

        public void popupMenuCanceled(PopupMenuEvent e) {
            System.out.println("Popup menu is hidden!");
        }
    }

    public static void main(String s[]) {
        JFrame frame = new JFrame("Popup Menu Example");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setContentPane(new PopupMenuExample());
        frame.setSize(300, 300);
        frame.setVisible(true);
    }
}
```

JToolBar

- Là lớp chứa cho nhiều thành phần khác
- Khi 1 component được gắn vào tool bar nó sẽ được định vị từ trái sang phải theo chỉ mục của nó



Các thuộc tính

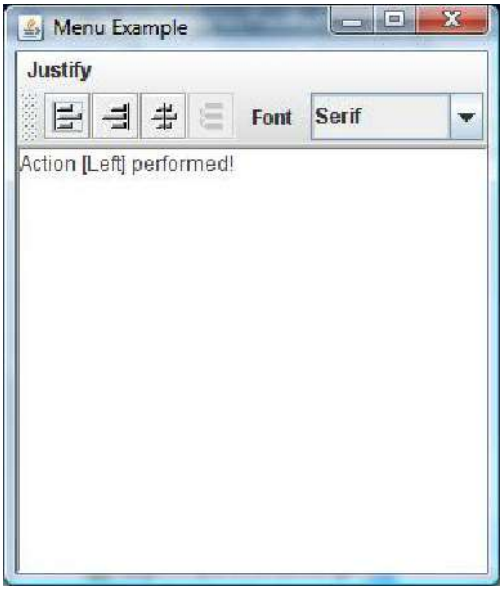
Property	Data type	get	is	set	Default value
accessibleContext	AccessibleContext	·			JToolBar.AccessibleJToolBar( )
borderPainted[b]	boolean		·	·	true
componentAtIndex[i]	Component	·			
floatable[b]	boolean		·	·	true
layout[o]	LayoutManager	·		·	BoxLayout(X_AXIS)
margin[b]	Insets	·		·	Insets(0,0,0,0)
orientation[b]	int	·		·	SwingConstants.HORIZONTAL
rollover[1.4, b]	boolean		·	·	false
UI[b]	ToolBarUI	·		·	From L&F
UIClassID[o]	String	·			"ToolBarUI"

[1.4]since 1.4, [b]bound

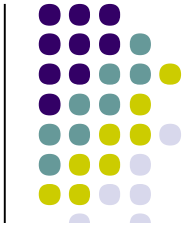
Event, constructor

- Dùng sự kiện PropertyChangeEvent khi có 1 sự thay đổi giá trị của các thuộc tính
- public JToolBar( )
- public JToolBar(int orientation)
- public JToolBar(String name)
- public JToolBar(String name, int orientation)

Ví dụ sử dụng JToolBar



Ví dụ sử dụng JToolBar



```
public class ToolBarExample extends JPanel {

    public JTextPane pane;
    public JMenuBar menuBar;
    public JToolBar toolBar;
    String fonts[] = {"Serif","SansSerif","Monospaced","Dialog","DialogInput"};

    public ToolBarExample( ) {
        menuBar = new JMenuBar( );

        // Create a set of actions to use in both the menu and toolbar.
        DemoAction leftJustifyAction = new DemoAction("Left",
            new ImageIcon("align_h_left.gif"), "Left justify text", 'L');
        DemoAction rightJustifyAction = new DemoAction("Right",
            new ImageIcon("align_h_right.gif"), "Right justify text", 'R');
        DemoAction centerJustifyAction = new DemoAction("Center",
            new ImageIcon("align_h_centers.gif"), "Center justify text", 'M');
        DemoAction fullJustifyAction = new DemoAction("Full",
            new ImageIcon("align_v_space_disabled.gif"), "Full justify text", 'F');

       JMenu formatMenu = new JMenu("Justify");
       formatMenu.add(leftJustifyAction);
       formatMenu.add(rightJustifyAction);
       formatMenu.add(centerJustifyAction);
       formatMenu.add(fullJustifyAction);
       menuBar.add(formatMenu);
```

Ví dụ sử dụng JToolBar

```
    toolBar = new JToolBar("Formatting");
    toolBar.add(leftJustifyAction);
    toolBar.add(rightJustifyAction);
    toolBar.add(centerJustifyAction);
    toolBar.add(fullJustifyAction);

    toolBar.addSeparator( );
    JLabel label = new JLabel("Font");
    toolBar.add(label);

    toolBar.addSeparator( );
    JComboBox combo = new JComboBox(fonts);
    combo.addActionListener(new ActionListener( ) {
        public void actionPerformed(ActionEvent e) {
            try { pane.getStyledDocument( ).insertString(0,
                "Font [" + ((JComboBox)e.getSource( )).getSelectedItem( ) +
                "] chosen!\n", null);
            } catch (Exception ex) { ex.printStackTrace( ); }
        }
    });
    toolBar.add(combo);

    // Disable one of the Actions.
    fullJustifyAction.setEnabled(false);
}
```

Ví dụ sử dụng JToolBar

```
public static void main(String s[]) {
    ToolBarExample example = new ToolBarExample( );
    example.pane = new JTextPane( );
    example.pane.setPreferredSize(new Dimension(250, 250));
    example.pane.setBorder(new BevelBorder(BevelBorder.LOWERED));
    example.toolBar.setMaximumSize(example.toolBar.getSize( ));
    JFrame frame = new JFrame("Menu Example");
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setJMenuBar(example.menuBar);
    frame.getContentPane( ).add(example.toolBar, BorderLayout.NORTH);
    frame.getContentPane( ).add(example.pane, BorderLayout.CENTER);
    frame.pack( );
    frame.setVisible(true);
}
class DemoAction extends AbstractAction {
    public DemoAction(String text, Icon icon, String description,
                      char accelerator) {
        super(text, icon);
        putValue(ACCELERATOR_KEY, KeyStroke.getKeyStroke(accelerator,
                 Toolkit.getDefaultToolkit( ).getMenuShortcutKeyMask( )));
        putValue(SHORT_DESCRIPTION, description);
    }
    public void actionPerformed(ActionEvent e) {
        try { pane.getStyledDocument( ).insertString(0,
              "Action [" + getValue(NAME) + "] performed!\n", null);
        } catch (Exception ex) { ex.printStackTrace( ); }
    }
}
```

JSlider Class

- JSlider là lớp dùng để hiển thị con chạy
- Giống như scrollbar, slider có con chạy theo cả hai chiều dọc và ngang
- Slider thường được dùng để hiển thị 1 giá trị trong khoảng nào đó.
- JSlider cho phép chúng ta lựa chọn khoảng cách giữa 2 điểm tick

Các thuộc tính

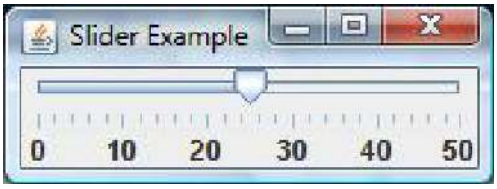
Property	Data type	get	is	set	Default value
accessibleContext°	AccessibleContext	·			JSlider.AccessibleJSlider( )
changeListeners[1,4]	ChangeListener[]	·			Empty array
extent	int	·		·	0
inverted[b]	boolean	·		·	false
labelTable[b]	Dictionary	·		·	null
majorTickSpacing[b]	int	·		·	10
maximum[b]	int	·		·	100
minimum[b]	int	·		·	0
minorTickSpacing[b]	int	·		·	2
model[b]	BoundedRangeModel	·		·	`DefaultBoundedRangeModel`
orientation[b]	int	·		·	JSlider.HORIZONTAL
paintLabels[b]	boolean	·		·	false
paintTicks[b]	boolean	·		·	false
paintTrack[b]	boolean	·		·	true
snapToTicks[b]	boolean	·		·	true
UI[b]	SliderUI	·		·	From L&F
UIClassID°	String	·			"SliderUI"
value	int	·		·	50
valueIsAdjusting	boolean	·		·	false

Events, Constructors & Labels

- public void addChangeListener(ChangeListener l)
- public void removeChangeListener(ChangeListener l)
- public JSlider( )
- public JSlider(int orientation)
- public JSlider(int min, int max)
- public JSlider(int min, int max, int value)
- public JSlider(int orientation, int minimum, int maximum, int value)
- public JSlider(BoundedRangeModel brm)
- public Hashtable createStandardLabels(int increment)
- public Hashtable createStandardLabels(int increment, int start)

Ví dụ sử dụng JSlider

```
public class SliderExample extends JPanel {
    public SliderExample() {
        super(true);
        this.setLayout(new BorderLayout());
        JSlider slider = new JSlider(JSlider.HORIZONTAL, 0, 50, 25);
        slider.setMinorTickSpacing(2);
        slider.setMajorTickSpacing(10);
        slider.setPaintTicks(true);
        slider.setPaintLabels(true);
        // We'll use just the standard numeric labels for now.
        slider.setLabelTable(slider.createStandardLabels(10));
        add(slider, BorderLayout.CENTER);
    }
    public static void main(String s[]) {
        JFrame frame = new JFrame("Slider Example");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setContentPane(new SliderExample());
        frame.pack();
        frame.setVisible(true);
    }
}
```



JProgressBar Class

- Thanh progess bar thường được dùng để chỉ trạng thái thời gian thực thi của chương trình
- Thanh progress bar thực chất chỉ là 1 hình chữ nhật có chiều dài tùy ý, tỷ lệ chiều dài của nó sẽ được làm đầy dựa trên giá trị của model.
- Có 2 dạng progress bar trong swing:
  - Thanh dọc: sẽ được vẽ từ trái sang phải
  - Thanh ngang: sẽ được vẽ từ dưới lên trên

Thuộc tính

Property	Data type	get	is	set	Default value
accessibleContext[o]	AccessibleContext	·			JProgressBarAccessibleJProgressBar( )
borderPainted[b]	boolean		·	·	true
changeListeners[1.4]	ChangeListener[]	·			Empty array
indeterminate[b, 1.4]	boolean		·	·	false
maximum	int	·		·	100
minimum	int	·		·	0
model	BoundedRangeModel	·		·	`DefaultBoundedRangeModel( )`
orientation[b]	int	·		·	JProgressBar.HORIZONTAL
percentComplete	double		·	·	
string[b]	String	·		·	null
stringPainted[b]	boolean	·		·	false
`UI`[b]	progressBarUI	·		·	From L&F
`UIClassID`[o]	String	·			"ProgressBarUI"
value[b]	int	·		·	0

Events & constructor

- public void addChangeListener(ChangeListener l)
- public void removeChangeListener(ChangeListener l)
- public JProgressBar( )
- public JProgressBar(BoundedRangeModel model)
- public JProgressBar(int orient, int min, int max)
- public JProgressBar(int min, int max)
- public JProgressBar(int orient)

Ví dụ sử dụng JProgessBar

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

public class ProgressBarExample extends JPanel {
  JProgressBar pbar;
  static final int MY_MINIMUM=0;
  static final int MY_MAXIMUM=100;

  public ProgressBarExample( ) {
     pbar = new JProgressBar( );
     pbar.setMinimum(MY_MINIMUM);
     pbar.setMaximum(MY_MAXIMUM);
     add(pbar);
  }

  public void updateBar(int newValue) {
    pbar.setValue(newValue);
  }
```

Ví dụ sử dụng JProgessBar

```
public static void main(String args[]) {

    final ProgressBarExample it = new ProgressBarExample( );

    JFrame frame = new JFrame("Progress Bar Example");
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setContentPane(it);
    frame.pack( );
    frame.setVisible(true);

    for (int i = MY_MINIMUM; i <= MY_MAXIMUM; i++) {
      final int percent=i;
      try {
        SwingUtilities.invokeLater(new Runnable( ) {
            public void run( ) {
              it.updateBar(percent);
            }
        });
        java.lang.Thread.sleep(100);
      } catch (InterruptedException e) {;}
    }
  }
}
```

JDialog Class

- Hộp thoại thường là cửa sổ phụ trình bày danh sách tùy chọn hoặc hiển thị thông điệp
- Hộp thoại thường cung cấp nút xác nhận thay đổi hoặc trường nhập vào thành phần, hủy bỏ thay đổi…
- Có 3 dạng hộp thoại:
  - Hộp thoại thông điệp
  - Hộp thoại xác nhận
  - Hộp thoại tập tin (JFileChooser)

Thuộc tính

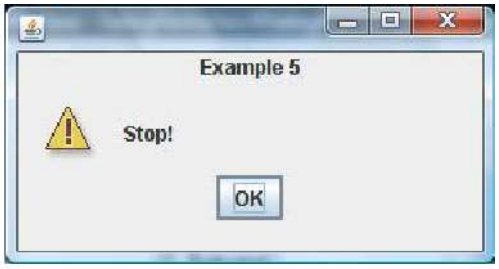
Property	Data type	get	is	set	Default value
accessibleContext°	AccessibleContext	.			JDialog.AccessibleJDialog( )
contentPane°	Container	.		.	From rootPane
defaultCloseOperation	int	.		.	HIDE_ON_CLOSE
defaultLookAndFeelDecorated[5, 1.4]	boolean		.	.	Depends on L&F, often false
glassPane°	Component	.		.	From rootPane
JMenuBar°	JMenuBar	.		.	null
layeredPane°	JLayeredPane	.		.	From rootPane
layout°	LayoutManager	.		.	BorderLayout
modal*	boolean		.	.	false
parent*	Container	.			SwingUtilities.get-SharedOwnerFrame( )
rootPane	JRootPane	.			JRootPane
title*	String	.		.	""

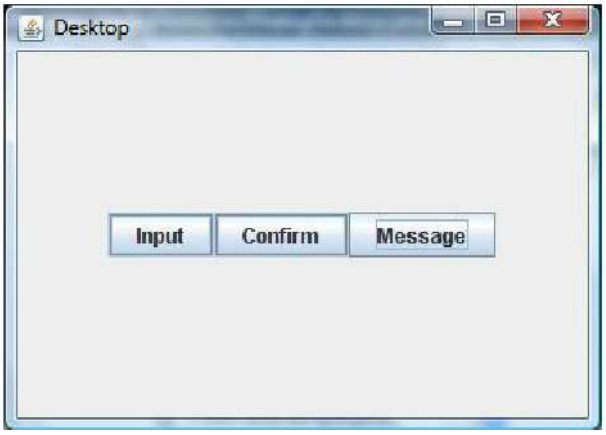
Constructors

- public JDialog( )
- public JDialog(Dialog owner, String title, boolean modal, GraphicsConfiguration gc)
- public JDialog(Frame owner, String title, boolean modal, GraphicsConfiguration gc)

Một vài ví dụ về JDialog

Một vài ví dụ về JDialog

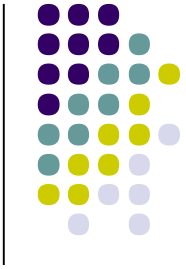




Một vài ví dụ về JDialog

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

public class JDialogExample extends JFrame {
    public static void main(String[] args) {
        //ex1
        JOptionPane.showInputDialog(null, "Please choose a name",
        "Example 1",
        JOptionPane.QUESTION_MESSAGE, null, new Object[] {
        "Amanda", "Colin", "Don", "Fred", "Gordon", "Janet", "Jay", "Joe",
        "Judie", "Kerstin", "Lotus", "Maciek", "Mark", "Mike", "Mulhern",
        "Oliver", "Peter", "Quaxo", "Rita", "Sandro", "Tim", "Will"}, "Joe");

        //ex2
        JOptionPane.showInputDialog(null, "Please enter your name",
        "Example 2",
        JOptionPane.QUESTION_MESSAGE, null, null, "Shannon");

        //ex3
        JOptionPane.showMessageDialog(null, "Have a nice day.", "Example 3",
        JOptionPane.INFORMATION_MESSAGE, new ImageIcon(
        "hand.jpg"));
```

Một vài ví dụ về JDialog



```
        //ex4
        JOptionPane.showConfirmDialog(null, "Are you sure?", "Example 4",
        JOptionPane.YES_NO_CANCEL_OPTION);

        // ex5
        JFrame f = new JFrame();
        Container c = f.getContentPane();
        c.setLayout(new BorderLayout());
        JOptionPane op = new JOptionPane("Stop!", JOptionPane.WARNING_MESSAGE);
        JPanel p = new JPanel(new FlowLayout());
        p.add(op);
        c.add(p);
        c.add(new JLabel("Example 5", JLabel.CENTER), BorderLayout.NORTH);
        f.pack();
        f.setVisible(true);
    }
}
```

Summary