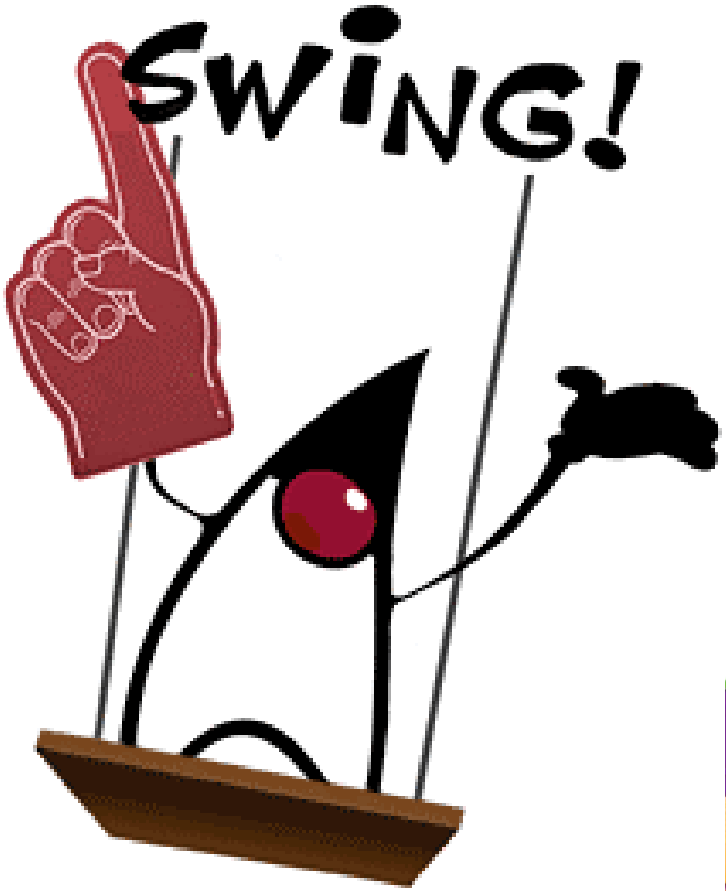
BACHKHOA EDUCATION APTECH

Bài 1

Giới thiệu tổng quan

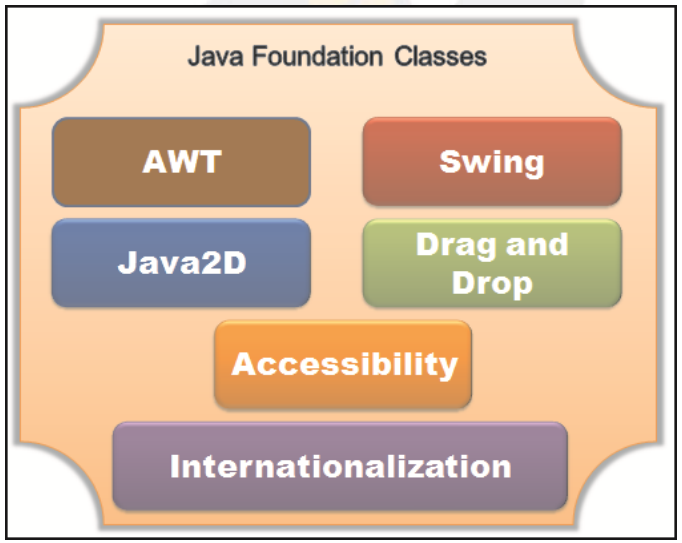
Mục tiêu

- Giới thiệu Java Swing
- Kiến trúc MVC
- Look & Feel
- Thành phần điều khiển
- Jframe
- Jpanel
- Ứng dụng Java Swing
- Control cơ bản



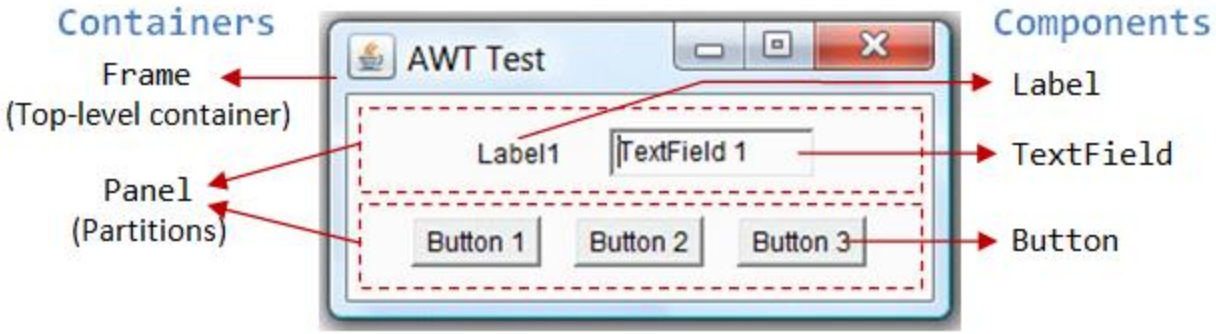
Giới thiệu

- **Java Foundation Class** (JFC) là framework đồ họa giới thiệu từ phiên bản JDK 2.0
- Thư viện này bao gồm **AWT** cũ, **Swing** và lớp **Java2D**
- Thư viện cung cấp **giao diện** chương trình Java một cách **nhất quán** cho dù HĐH sử dụng khác nhau.



Giới thiệu

- Thư viện **AWT** là thư viện đồ họa người dùng (**Graphical User Interface** - GUI).
- Nó được sử dụng xây dựng ứng dụng và applet.
- Nó chứa các lớp thành phần cơ bản như **Button**, **CheckBox**, **TextField**….
- Giao diện ứng dụng sử dụng AWT sẽ khác nhau trên các OS.

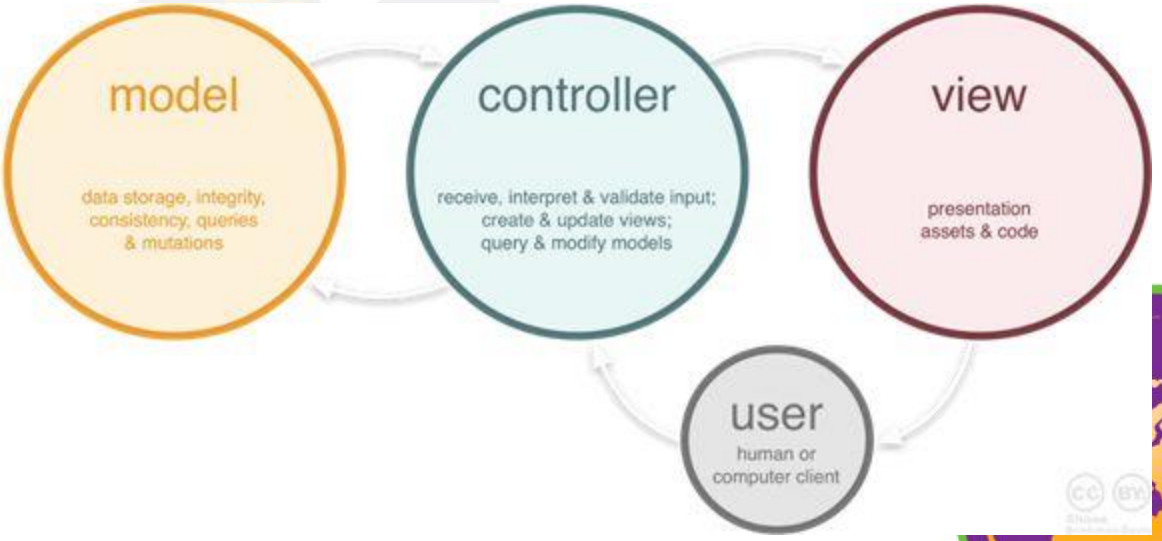


Giới thiệu

- Thư viện **SWING** là thư viện có kiến trúc MVC (Model-View-Controller).
- Hỗ trợ công nghệ "Pluggable-Look-And-Feel".

Kiến trúc MVC

- Là phương pháp tách các thành phần: giao diện, tương tác, dữ liệu.
- Là mẫu thiết kế hướng đối tượng giới thiệu vào cuối những năm 1970.
- Phân rã 3 chức năng thành 3 đối tượng riêng biệt:
  - Model
  - View
  - Controller



Kiến trúc MVC

Model

- Đại diện cho tất cả dữ liệu và trạng thái khác nhau của thành phần.
- Thành phần sẽ có trạng thái khác nhau do có tương tác của người dùng.
- Thông tin của trạng thái lưu trữ trong Model.
- Thực hiện giao tiếp với View và Controller.

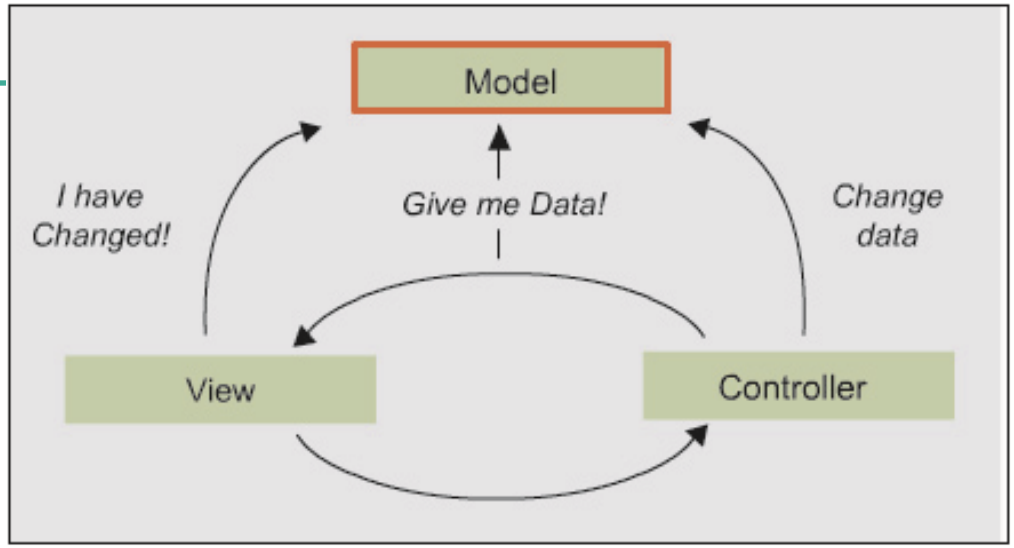
View

- Mô tả đồ họa lên màn hình.
- Lấy dữ liệu và trạng thái khác nhau từ Model đẻ hiển thị đồ họa thành phần.
- Hiển thị thành phần đồ họa trên màn hình và cập nhật khi có yêu cầu gián tiếp từ Model hoặc trực tiếp từ Controller.

Kiến trúc MVC

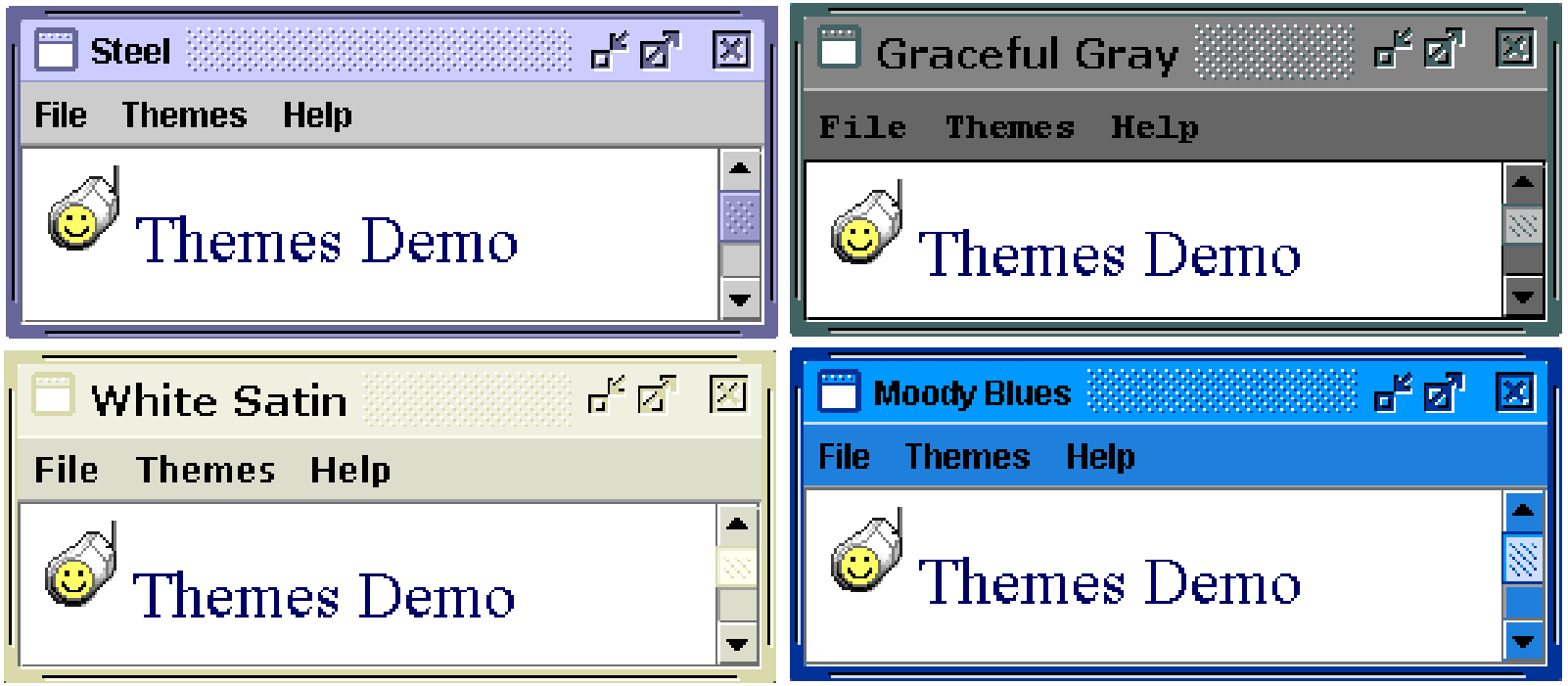
Controller

- Có trách nhiệm xác định các thành phần có phản ứng với bất kỳ sự kiện đầu vào từ các thiết bị như bàn phím, chuột.
- Xác định hành động được thực hiện khi sử dụng thành phần.
- Có thể nhận thông điệp trực tiếp từ View và gián tiếp từ Model.



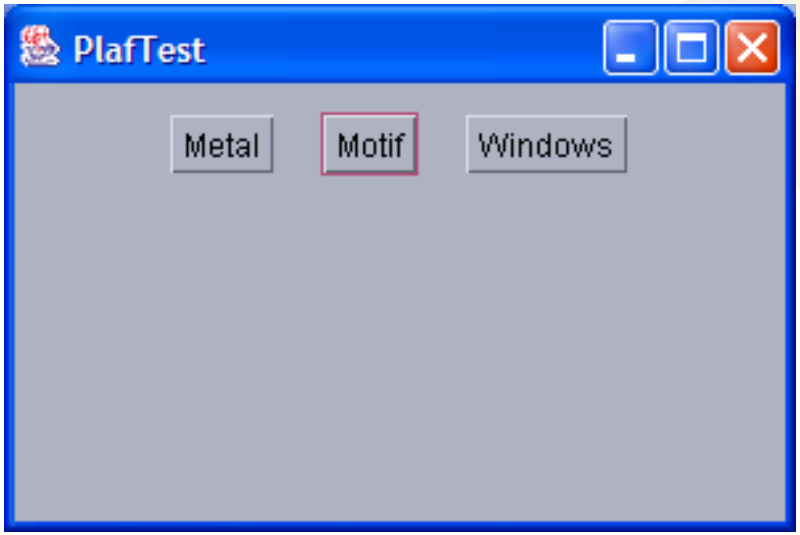
Look & Feel

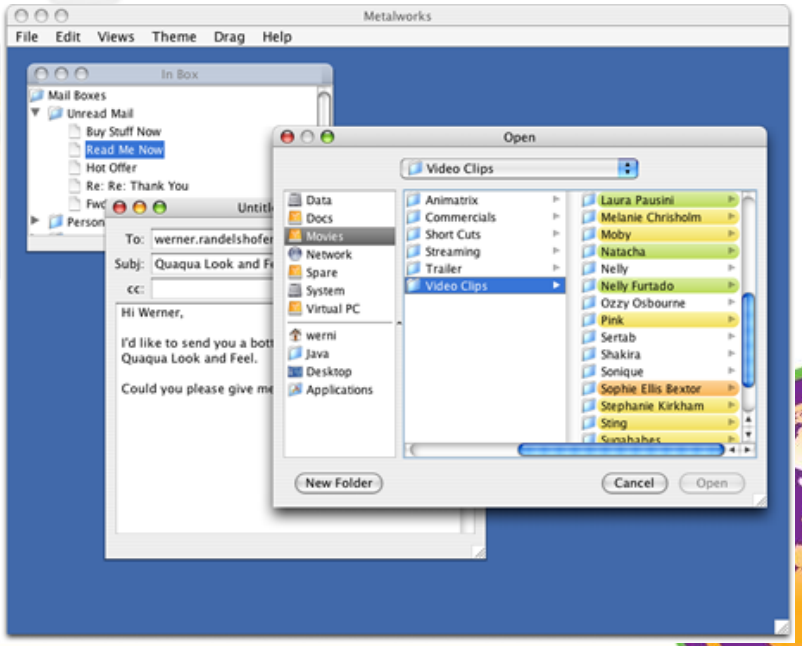
Giao diện ứng dụng Java Swing có thể thay đổi không phụ thuộc vào nền tảng OS.



Look & Feel

- Swing cung cấp mặc định sẵn 3 Lool & Feel: **Metal**, **Windows**, **Motif**.
- Look & Feel đặc trưng **Macintosh** cần phải tải riêng biệt.
- Look & Feel Windows chỉ giới hạn trên OS Windows vì lý do bản quyền.





Look & Feel

Phương thức **setLookAndFeel**() của **javax.swing.UIManager** sử dụng để cài đặt giao diện. Xem code mẫu sau để cài đặt.

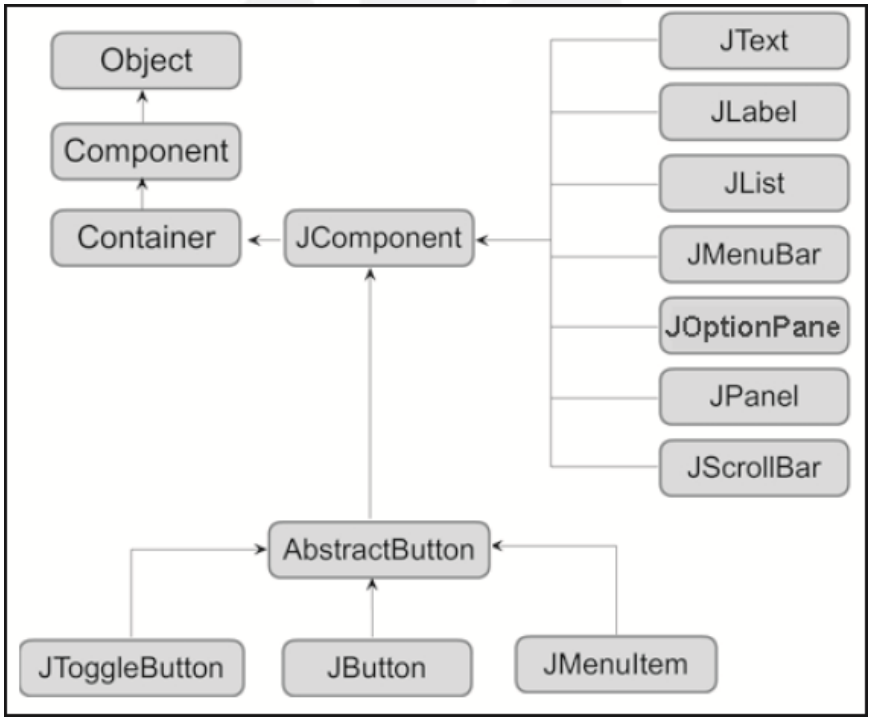
Code Snippet

```
try {
UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
 } catch( ClassNotFoundException ex) {
            System.out.println("Exception : " + ex.getMessage());
 } catch( UnsupportedLookAndFeelException ex){
            System.out.println("Exception : " +  ex.getMessage());
 }
```

Thành phần điều khiển

Java Swing cung cấp 2 thành phần GUI:

- **Component**: thành phần điều khiển đồ họa độc lập
- **Container**: cửa sổ GUI ảo sử dụng giữ component.



Thành phần điều khiển

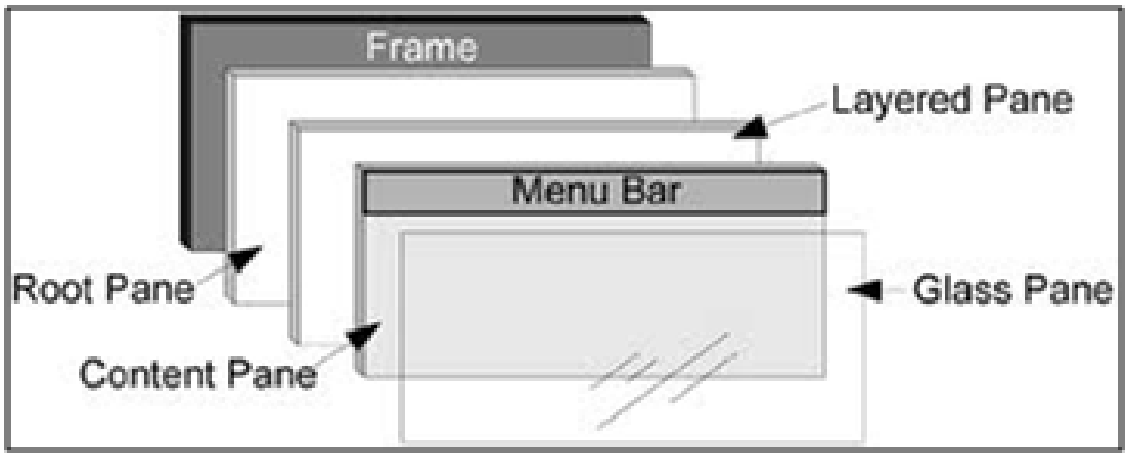
- Có 3 kiểu container trong Swing
- Mỗi container ở mức **top-level** gọi là **root-pane**.

Kiểu Container	Ví dụ	Mô tả
Top Level Containers	`JApplet`, `JDialog`, `JFrame`	Là thành phần bắt buộc có trong ứng dụng swing. Nó là gốc chứa các phân cấp container bên trong.
General Purpose Containers	`JPanel`, `JScrollPane`, `JTabbedPane`, `JToolBar`	Phổ biến nhất trong ứng dụng swing.
Special Purpose Containers	`JInternalFrame`, `JLayeredPane`, `JRootPane`	Có vai trò cụ thể trong giao diện người dùng.

Thành phần điều khiển

Root-pane có 4 phần: **GlassPane**, **LayeredPane**, **ContentPane** và **MenuBar**

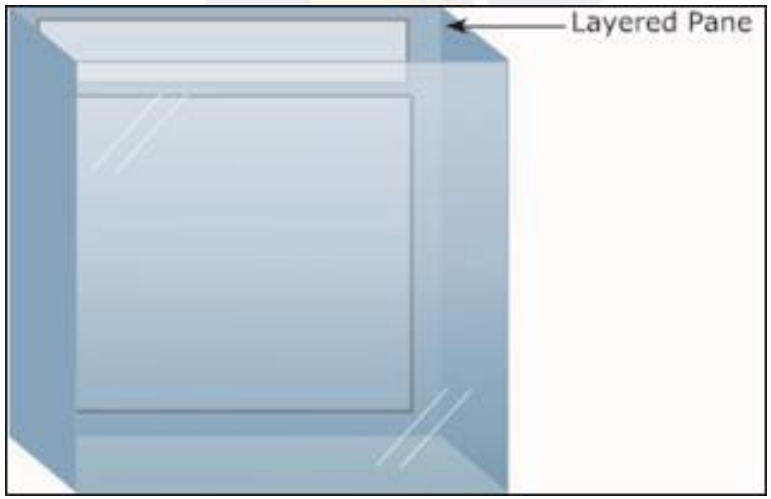
- GlassPane: mặc định ẩn, như tấm kính và hoàn toàn trong suốt.



Thành phần điều khiển

LayeredPane:

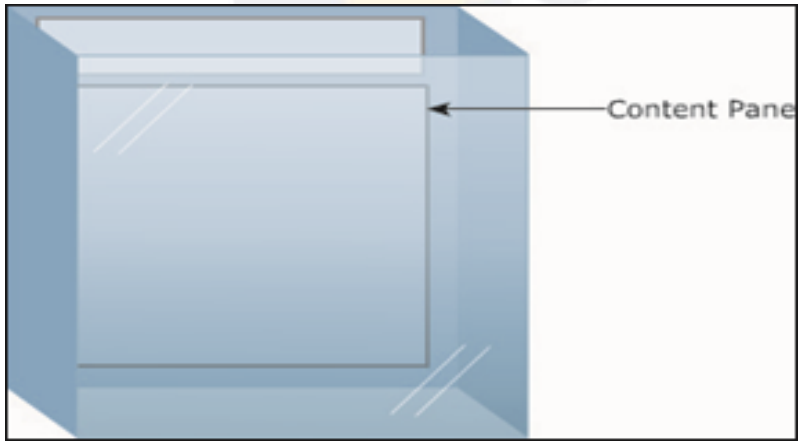
- Phục vụ định vị nội dung bao gồm phần nội dung và thanh menu.
- Nó có thể chứa các thành phần khác theo thứ tự chiều sâu.
- Chiều sâu (trục Z) cho phép cung cấp hành vi hiển thị như là popup menu trên thành phần khác.



Thành phần điều khiển

ContentPane:

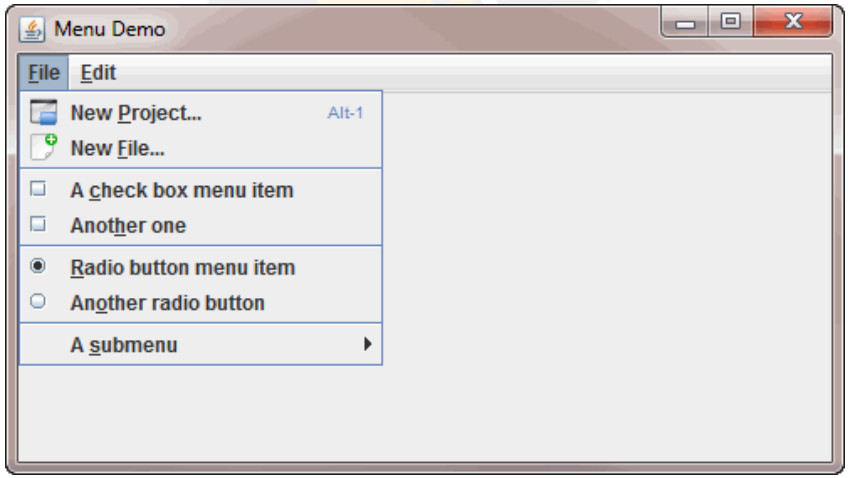
- Vùng chứa các thành phần nhìn thấy được trong pane gốc.
- Đa số thành phần GUI đều được thêm vào pane này.
- Bố cục (layout) mặc định là BorderLayout.
- Phương thức getContentPane() trả về container ở mức top-level để tiếp đó có thể thêm các thành phần.



Thành phần điều khiển

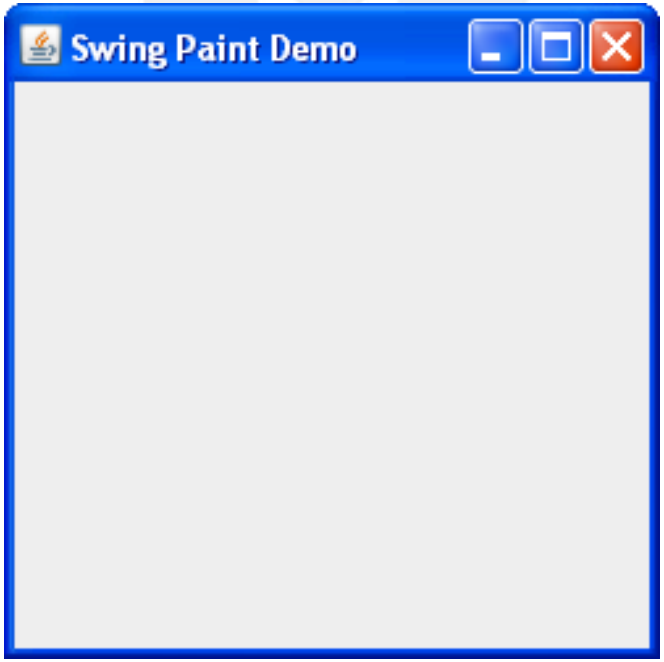
MenuBar:

- Tất cả container top-lever có thanh menu. Tuy nhiên thực tế nó chỉ xuất hiện ở Jframe và Japplet.
- Để thêm menu, một JMenuBar được tạo ra và cài đặt bằng setJMenuBar().
- Thanh menu có thể thêm các thành phần nhẹ khác như Jbutton, JTextField.



JFrame

- **JFrame** là container mức top-level sử dụng để tạo ứng dụng GUI căn bản.
- Có 2 cách tiếp cận: kế thừa Jframe và tạo đối tượng Jframe.



JFrame

Kế thừa JFrame.

Code Snippet

```
import javax.swing.*;
public class WinApp extends JFrame {
// Instance Data
    . . .
    . . .
  public WinApp() {
            . . .
  }
public static void main(String[] args)     {
    WinApp app = new WinApp();
            . . .
}
 }
```

JFrame

Tạo đối tượng JFrame.

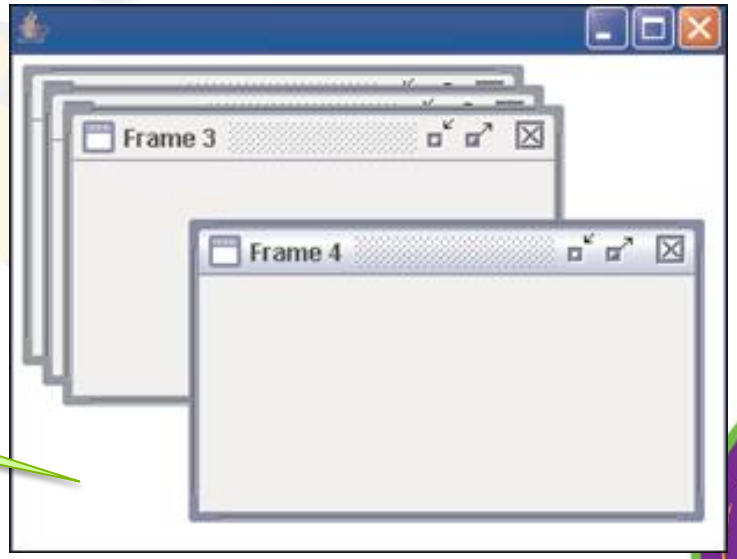
Code Snippet

```
import javax.swing.*;
public class WinApp extends SomeOtherClass
{
        // Instance Data    .
         . .
        public static void main(String[] args)    {
        JFrame frame = new JFrame();
         . . .
 }
}
```

JFrame

- Có thể hiển thị Jframe như cửa sổ window bên trong cửa sổ window khác.
- Cửa sổ có thể đóng, phóng to/nhỏ.
- Cửa sổ này có thể add thêm các thành phần con khác.

Content pane of the `JInternalFrame` container where child components are added.



JFrame

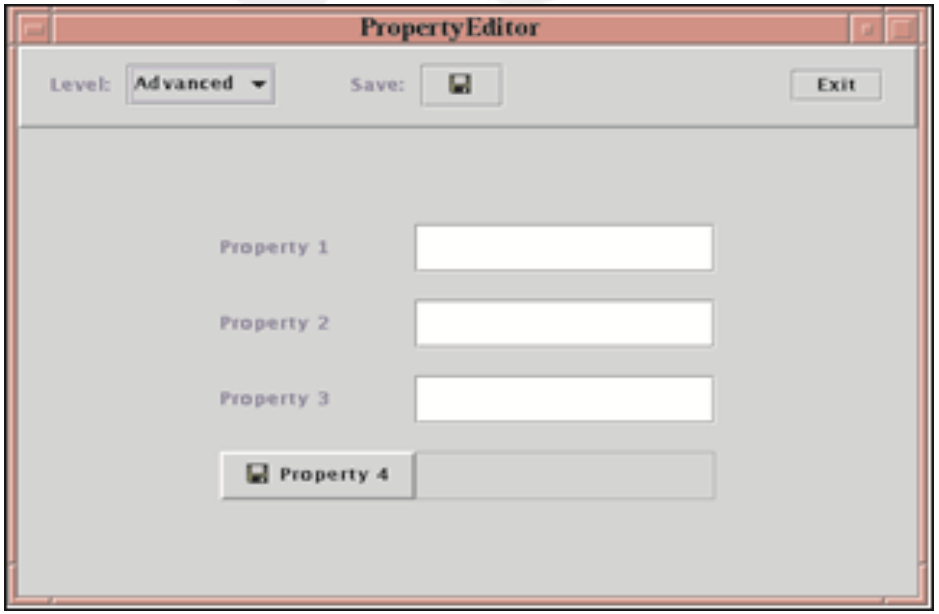
Code mẫu:

Code Snippet

```
. . .
JDesktopPane desk1 = new JDesktopPane();
createInternalFrame();
setContentPane(desk1);
 protected void createInternalFrame()
 {
    MyInternalFrame fm1 = new MyInternalFrame();
    fm1.setVisible(true);
    desk1.add(fm1);
    . . .
    fm1.setSelected(true);
 }
. . .
```

JPanel

- Jpanel đồng thời vừa là container và component
- Nó có thể như là container mức top-level hoặc được thêm vào bên trong Jpanel khác.
- Có dạng hình chữ nhật không có đường viền (mặc định). Jpanel có thể thêm các thành phần control khác.



JPanel

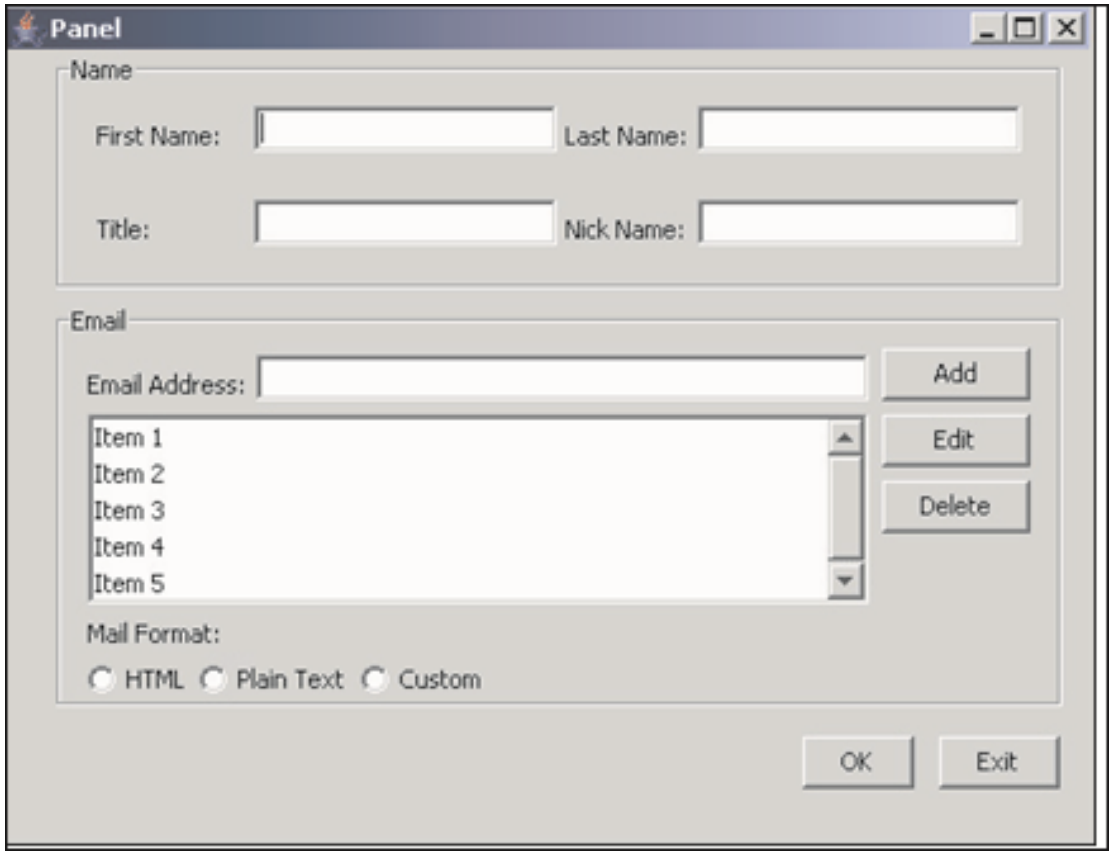
- Jpanel có layout mặc định là FlowLayout.
- Mặc định Jpanel ở trạng thái visible là true.
- Code:

Code Snippet

```
import javax.swing.JPanel;
. . .
JPanel myPanel;
myPanel = new JPanel();
. . .
. . .
```

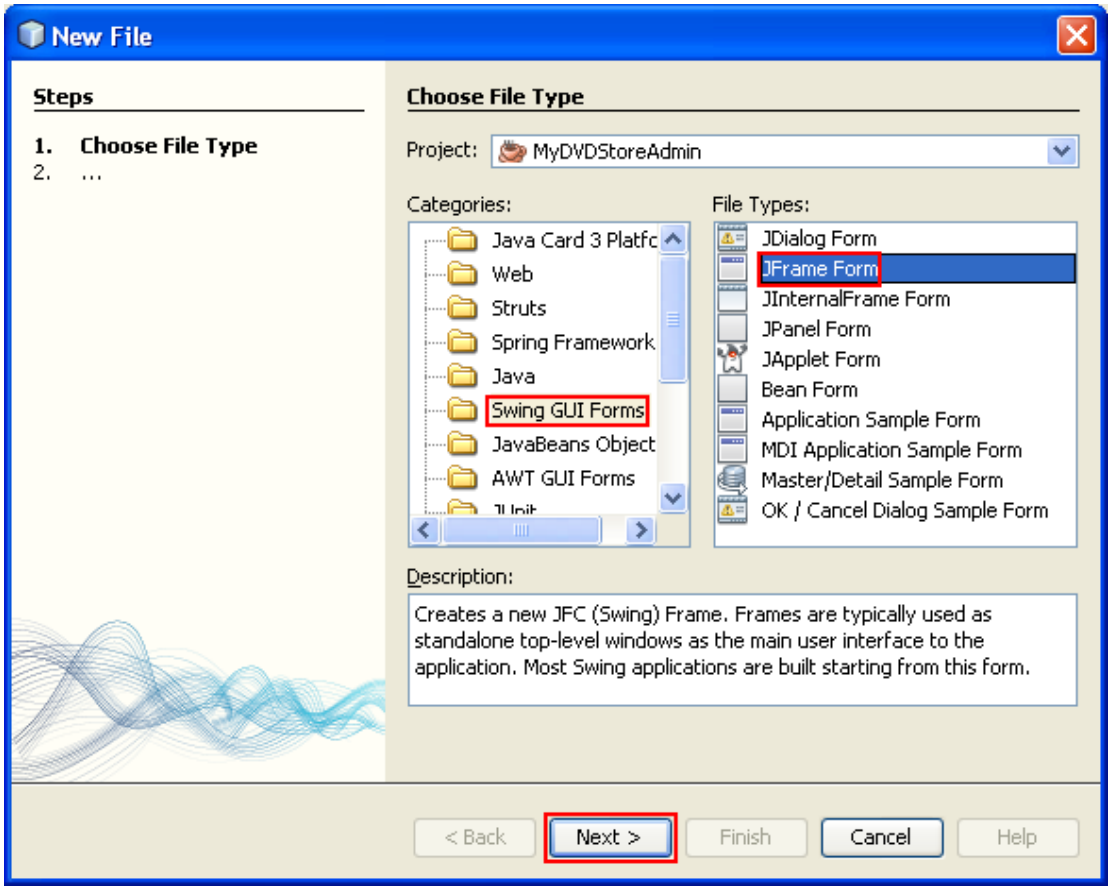
JPanel

Jpanel có viền, chứa các thành phần điều khiển.



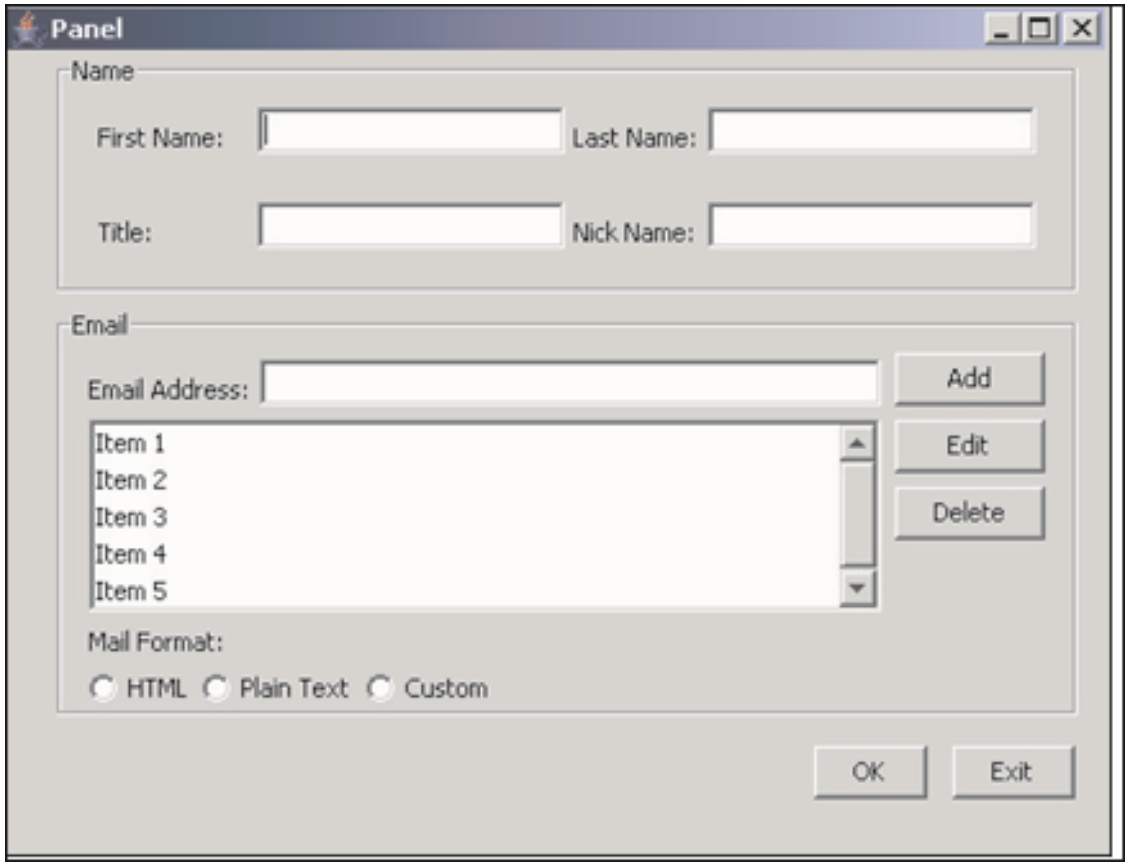
Ứng dụng Java Swing

Tạo mới ứng dụng và khởi tạo frame



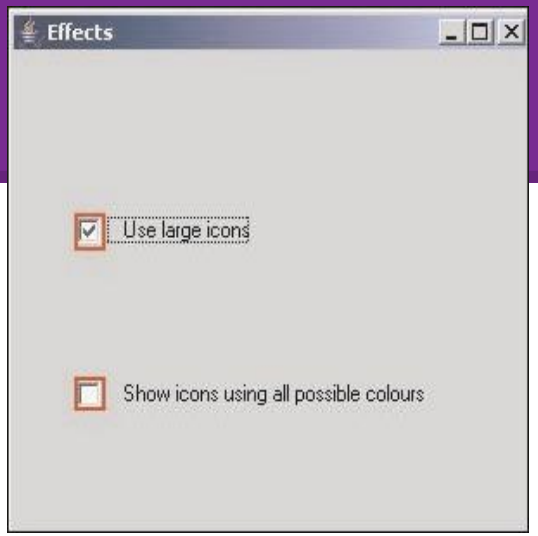
Control cơ bản

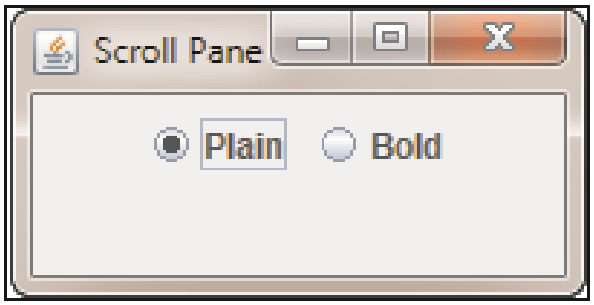
Lightweight Component

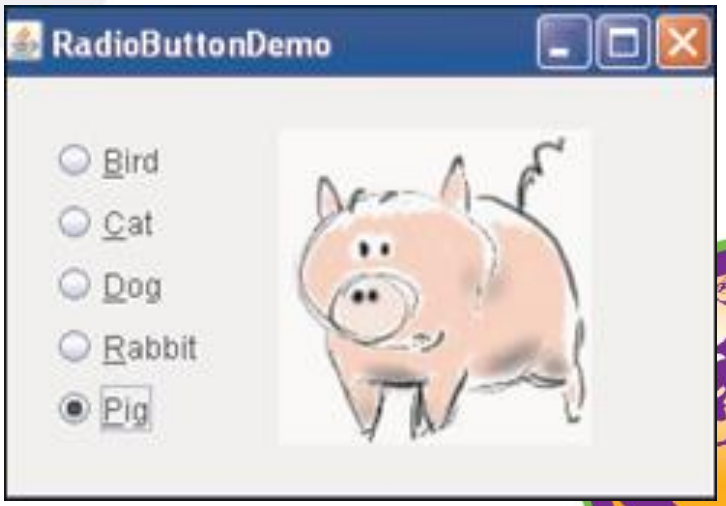


Control cơ bản

- Jlabel
- Jbutton
- Jcheckbox
- JRadioButton – ButtonGroup
- Jtextfield
- JtextArea
- JPasswordField

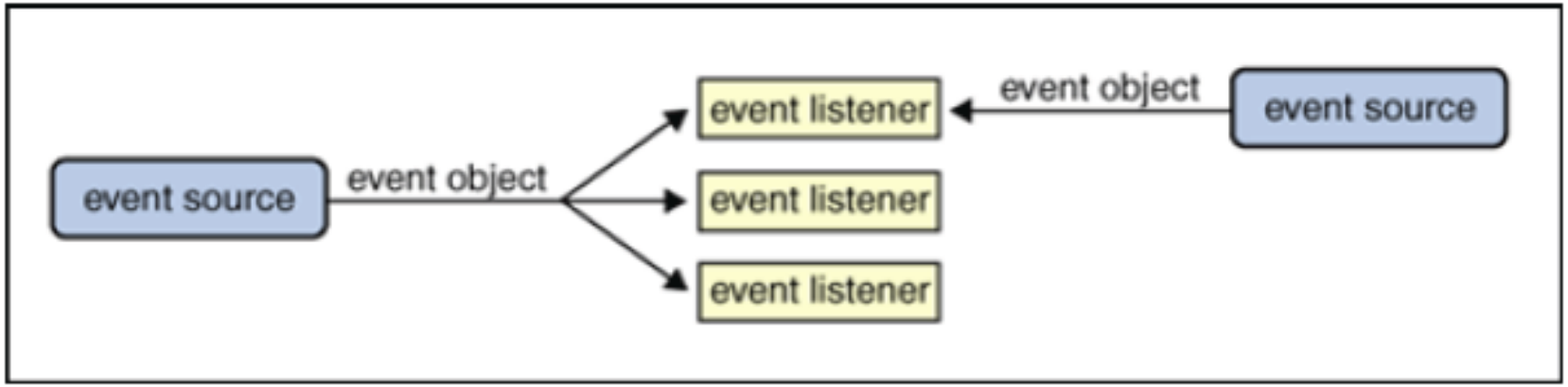






Control cơ bản

Quản lý sự kiện



Control cơ bản

Quản lý sự kiện thành phần

Events	Source	Listeners	Interface Methods
`ActionEvent`	Button, Menu, List	`ActionListener`	`void actionPerformed( ActionEvent ae)`
`AdjustmentEvent`	Scrollbar	`AdjustmentListener`	`void adjustmentValueChanged(AdjustmentEvent ae)`
`ItemEvent`	Check box, List	`ItemListener`	`void itemStateChanged (ItemEvent ie)`

Control cơ bản

Quản lý sự kiện chuột

Events	Source	Listeners	Interface Methods

`MouseEvent`	Mouse	`MouseListener`
		`MouseMotionListener`
			`void mouseClicked(MouseEvent me)`
			`void mouseEntered(MouseEvent me)`
			`void mouseExited(MouseEvent me)`
			`void mousePressed(MouseEvent me)`
			`void mouseReleased(MouseEvent me)`
			`void mouseDragged(MouseEvent me)`
			`void mouseMoved(MouseEvent me)`

Control cơ bản

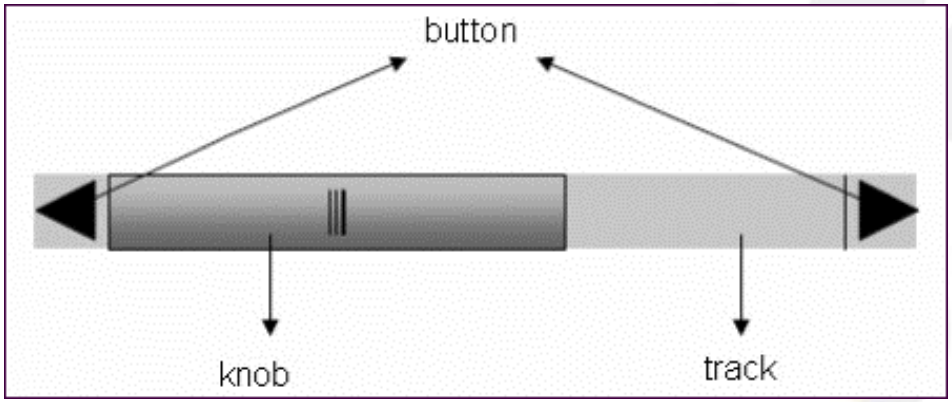
Quản lý sự kiện window

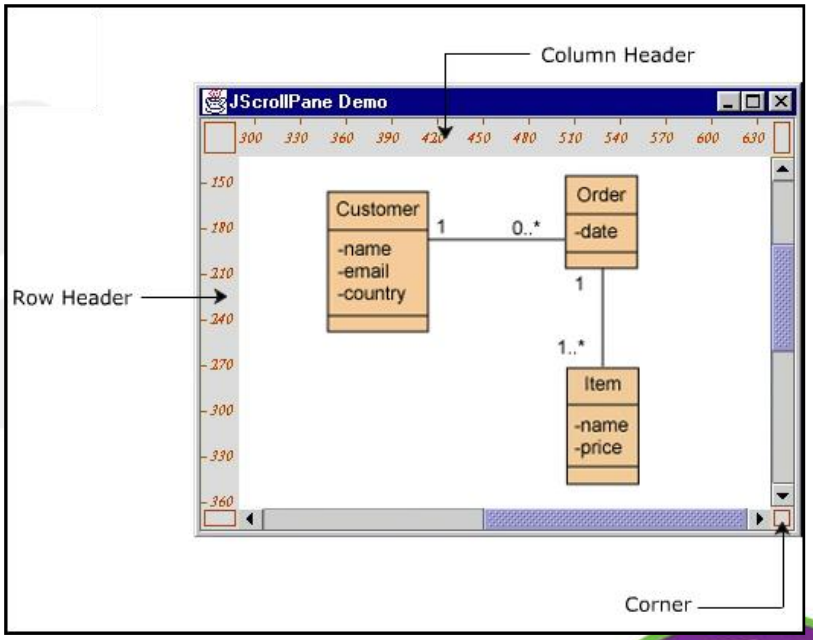
Events	Source	Listeners	Interface Methods

`WindowEvent`	Window	`WindowListener`	`void windowActivated(WindowEvent we)`
			`void windowClosed(WindowEvent we)`
			`void windowClosing(WindowEvent we)`
			`void windowDeactivated(WindowEvent we)`
			`void windowDeiconified(WindowEvent we)`
			`void windowIconified(WindowEvent we)`
			`void windowOpened(WindowEvent we)`

Control cơ bản

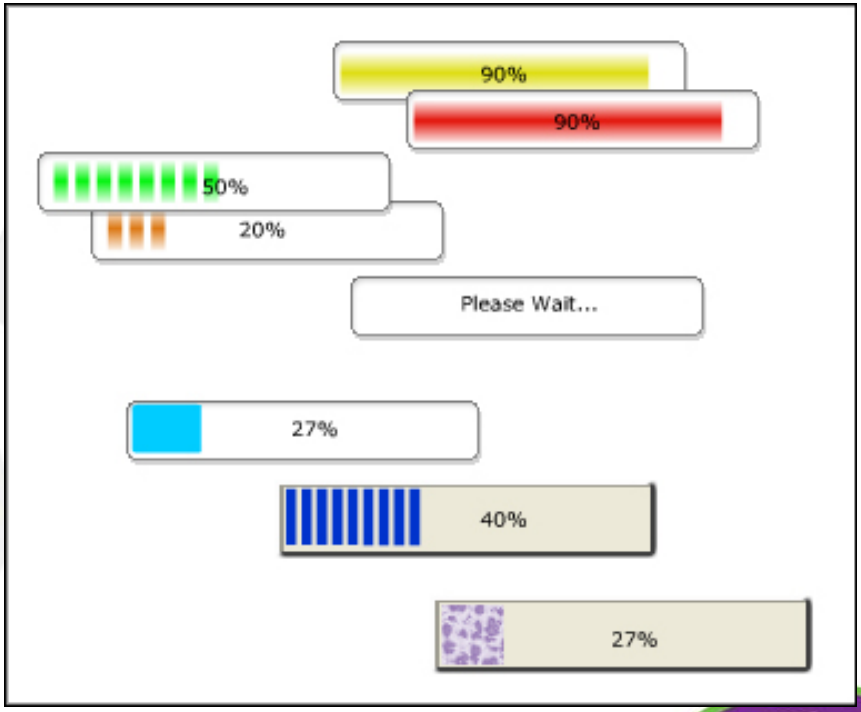
- Jscrollpane
- Jslider

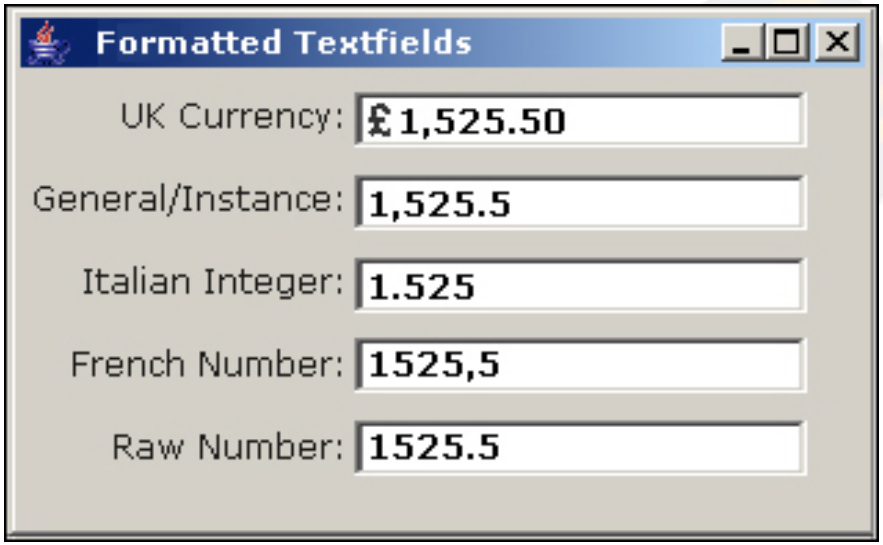




Control cơ bản

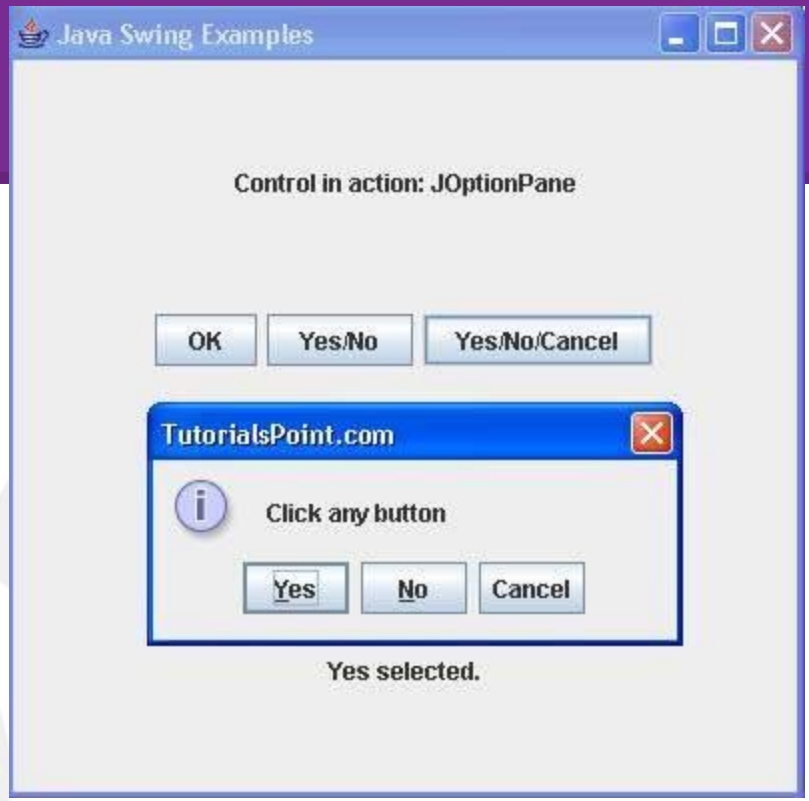
- Jprogressbar
- JFormattedTextField
- JEditorPane - JTextPane
- ImageIcon và Border API

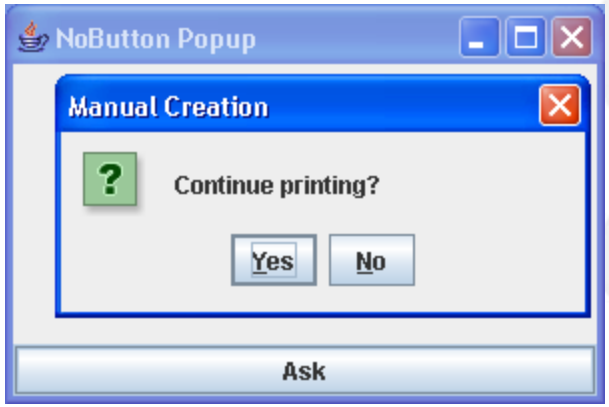


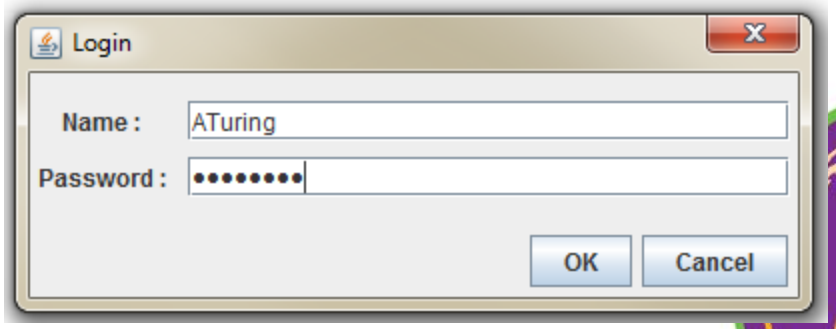


Control cơ bản

- JOptionPane
- JDialog







Tóm tắt bài học

- ✓ **Java Swing** hỗ trợ xây dựng lên ứng dụng Java có giao diện window.
- ✓ Thư viện Swing có ưu điểm hơn thư viện **AWT** cũ vì nó có thể độc lập giao diện, nhẹ hơn và hiệu năng hơn.
- ✓ Thành phần GUI trong Swing gồm: **container** và **component**.
- ✓ **Jframe** là thành phần gốc hiển thị lên cửa sổ.
- ✓ **Jpanel** mặc định là hiển thị và không có viền.
- ✓ Các control trong Swing có thể cài đặt các sự kiện tương ứng.

HẾT
XIN CẢM ƠN!