

**TRƯỜNG ĐẠI HỌC NAM CÀN THƠ**  
**KHOA CÔNG NGHỆ THÔNG TIN**

Trung tâm Đào tạo nguồn nhân lực & Phát triển Chuẩn Đầu Ra



**LẬP TRÌNH  
GIAO DIỆN ĐỒ HỌA (GUI)  
VỚI SWING**

ThS. Bùi Thị Diễm Trinh  
Khoa CNTT, Đại học Nam Cần Thơ  
Email: [trinhbtd2012@gmail.com](mailto:trinhbtd2012@gmail.com)  
Fone: 0932.907.666

# NỘI DUNG

1

Giới thiệu

2

Các lớp vật chứa (Container)

3

Các thành phần giao diện Swing

4

Bố cục giao diện

5

Xử lý sự kiện

6

Trình đơn, thanh công cụ

7

Thiết kế giao diện đồ họa với Eclipse

# Thư viện lập trình đồ họa

## ❑ Abstract Window Toolkit (AWT)

- 2 gói thường dùng là `java.awt` (điều khiển) và `java.awt.event` (sự kiện).
- Cung cấp giao diện phụ thuộc vào nền **GUI** của hệ điều hành (thay đổi theo hệ điều hành).
- Các thành phần được gọi là **heavyweight components**.

## ❑ Swing

- Bản nâng cấp của AWT
- Là 1 phần trong JFC (Java Foundation Classes)
- Giao diện **độc lập** với nền **GUI** của hệ điều hành (không thay đổi theo hệ điều hành, viết hoàn toàn bằng Java).
- Các thành phần được gọi là **lightweight components**.
- Tất cả các Swing component có tên bắt đầu với **J...**

## ❑ Gói **java.awt** bao gồm các lớp

- Thành phần GUI (*Button, TextField, and Label, ...*)
- Vật chứa GUI (*Frame, Panel, Dialog, ScrollPane, ...*)
- Sắp xếp bố cục (*FlowLayout, BorderLayout, GridLayout, ...*)
- Tùy chọn (*Graphics, Color, Font, ...*)

## ❑ Gói **java.awt.event** bao gồm các lớp

- Sự kiện (*ActionEvent, MouseEvent, KeyEvent, WindowEvent*)
- Lắng nghe sự kiện (*ActionListener, MouseListener, KeyListener, WindowListener, ...*)
- Các lớp Adapter (*MouseAdapter, KeyAdapter, and WindowAdapter*)

## Khái niệm

### ❑ Lớp vật chứa cấp cao (*top-level*):

- JFrame
- JDialog
- JOptionPane
- JFileChooser
- JColorChooser

### ❑ Lớp vật chứa thứ cấp (*secondary*)

- Được sử dụng để nhóm và sắp xếp bố cục các thành phần
- Thường dùng là JPanel, JScrollPane, JTabbedPane, JToolBar, ...

### ❑ Các thành phần GUI không thể thêm trực tiếp vào 1 vật chứa cấp cao mà chỉ được thêm vào ô nội dung (content-pane) của nó

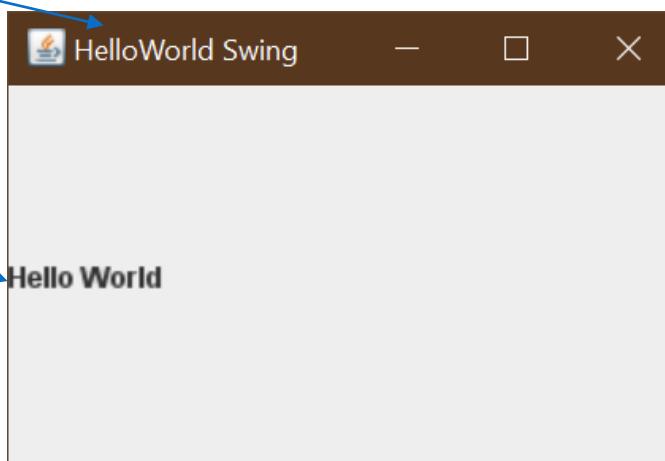
# Ví dụ: Tạo cửa sổ với Swing

- Ứng dụng HelloWorld cơ bản
- Tạo một cửa sổ với “HelloWorldString” trong thanh tiêu đề và hiển thị label “Hello World”

```

public class HelloWorldSwing extends JFrame{
    public HelloWorldSwing() {
        setTitle("HelloWorldSwing"); //Tiêu đề của JFrame
        setSize(300, 200); //Kích thước của JFrame
        setDefaultCloseOperation(EXIT_ON_CLOSE); //Thoát chương trình khi click nút exit
        setLocationRelativeTo(null); //Canh giữa màn hình
        setResizable(true); //Cho phép thay đổi kích thước
        JLabel label = new JLabel("Hello World");
        getContentPane().add(label);
    }
    public static void main(String[] args) {
        new HelloWorldSwing().setVisible(true);
    }
}

```

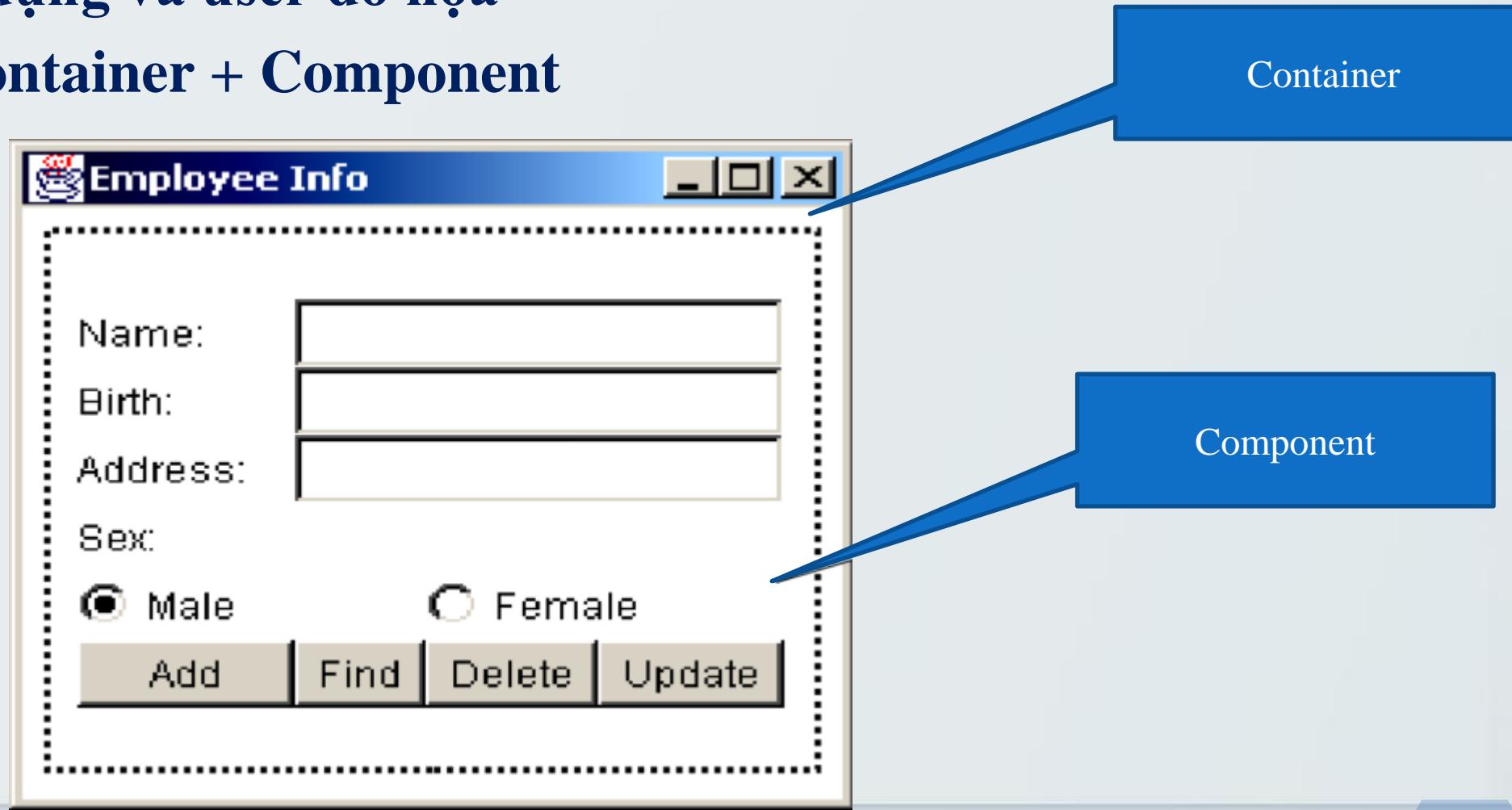


## Cơ bản về thiết kế GUI

- ❑ Khái niệm xây dựng GUI rất đơn giản. Những thành phần (**component**) được bố trí trong một bộ chứa (**container**) theo cách thức có tổ chức nào đó.
- ❑ Những component có thể là các đối tượng (**Button**, **Menu**, **Label**, **Textbox**, **Slider**, **Chechbox**, **Radio button**,...) hoặc có thể là các bộ chứa lồng nhau, ...
- ❑ Những thành phần được tổ chức trong những bộ chứa sử dụng bộ quản lý bố cục (**Layout Manager**)

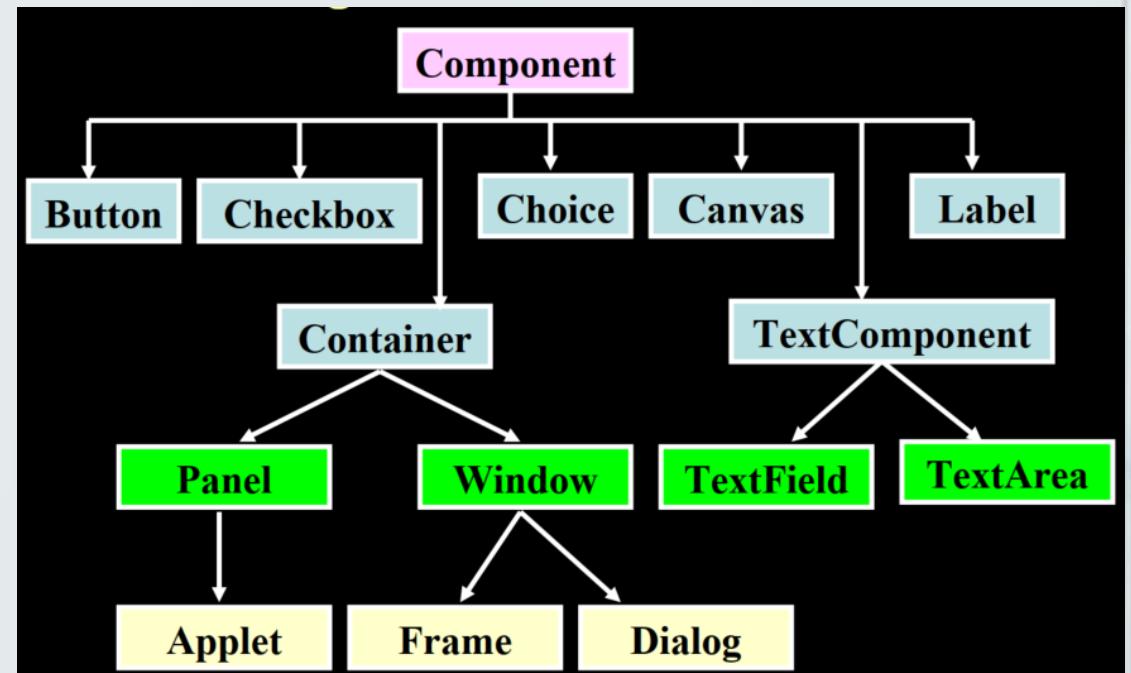
# Thư viện lập trình đồ họa

- ❑ GUI = **Graphic User Interface** – mô hình giao tiếp kiểu tương tác giữa ứng dụng và user đồ họa
- ❑ GUI = Container + Component



# Component

- ❑ Tất cả các **thành phần** cấu tạo nên chương trình GUI được gọi là **component**.
- ❑ Là các đối tượng có **biểu diễn đồ họa** được **hiển thị** lên **màn hình** mà người dùng tương tác được (**nút nhấn, checkbox, scrollbar,...**)
- ❑ **Ví dụ**
  - Containers,
  - textfields, labels, checkboxes, textareas
  - scrollbars, scrollpanes, dialog



# Đưa 1 component vào GUI

## ❑ Các bước để đưa 1 component vào GUI (*viết code*)

- Tạo 1 đối tượng component phù hợp
- Xác định hình thức bên ngoài lúc đầu của component.
- Định vị component này trên GUI
- Thêm component này vào GUI.
- Ví dụ



**Container**

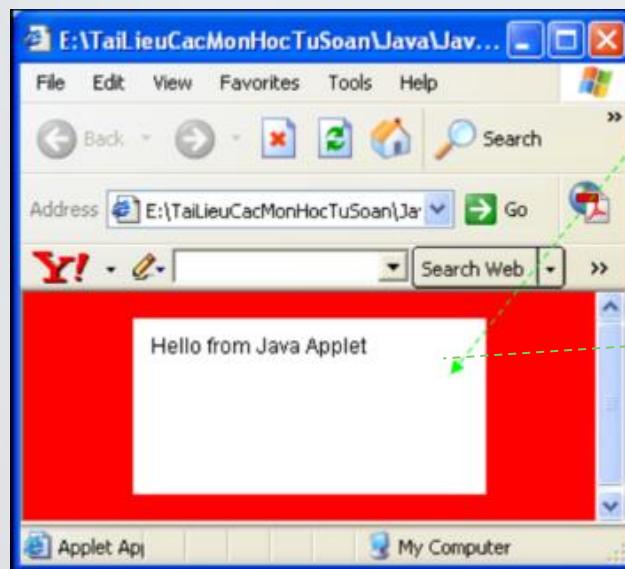
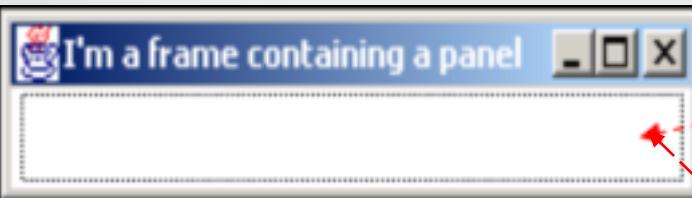
### Component

- 3 label
- 3 textField
- 1 checkbox groups  
chứa 2 check box
- 4 button

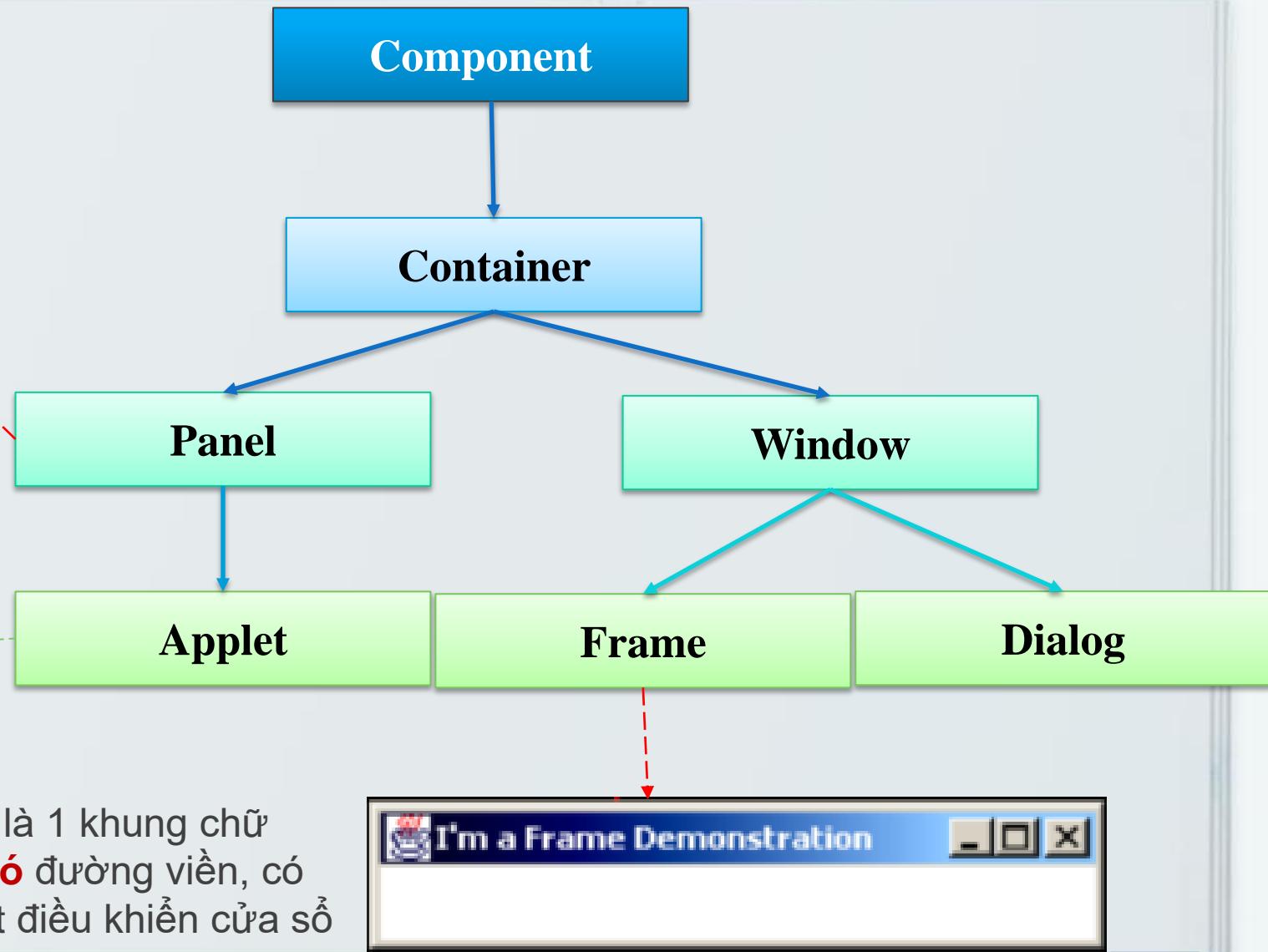
# Container

- ❑ Đối tượng vật chứa hay những đối tượng có khả năng quản lý và nhóm các đối tượng khác, hay nói cách khác Container là thành phần mà có thể chứa các thành phần khác
- ❑ Một số đối tượng container trong java
  - Panel: thường dùng để chứa các element trong 1 GUI phức tạp
  - ScrollPanes: tương tự Panel nhưng có thêm 2 thanh trượt.
  - Dialogs: cửa sổ dạng hộp thoại, dùng để đưa ra các thông báo, lấy dữ liệu nhập từ người dùng.
- ❑ Java.awt chứa một lớp có tên là **Container**. Lớp này dẫn xuất trực tiếp và không trực tiếp theo 2 cách là:
  - Frame
  - Panel

Panel là 1 vùng chữ nhật,  
không có đường viền



Frame là 1 khung chữ  
nhật, **có** đường viền, có  
các nút điều khiển cửa sổ



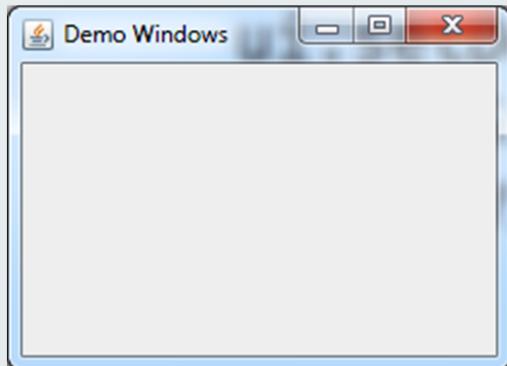
## Giới thiệu gói Swing, lớp JFrame, JPanel

- ❑ Gói tạo giao diện, tạo cửa sổ, tạo lớp chứa control, các thành phần liên quan tới giao diện

```
import java.awt.*;
```

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

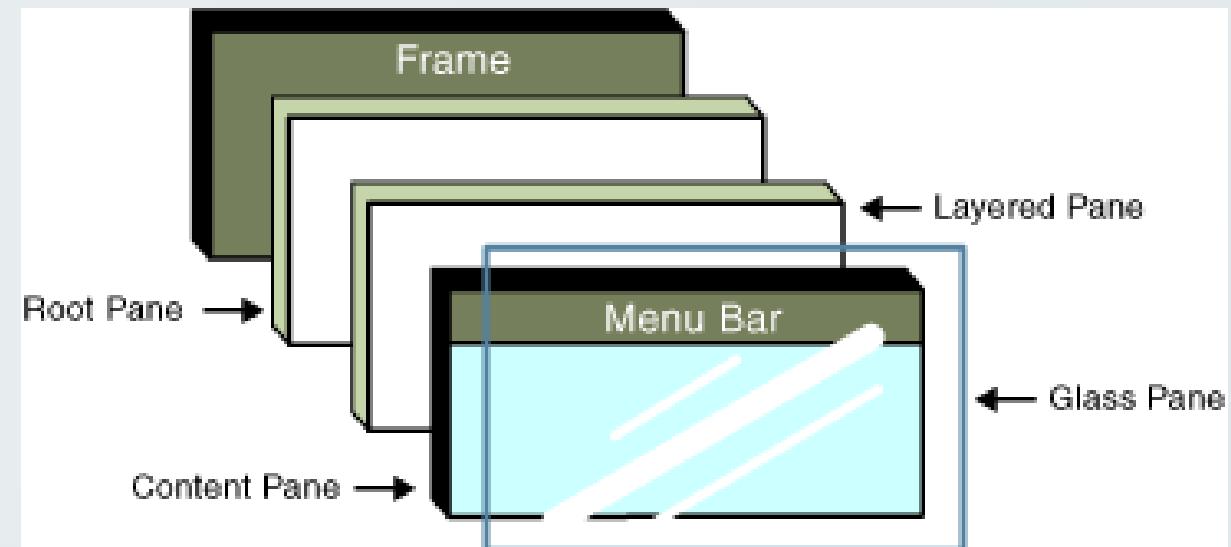
```
import javax.swing.*;
```



# JFrame

## ❑ JFrame

- Có thể được tạo ra bởi nhiều tầng (layer)
- Các tầng có thể chứa nhiều thành phần và được thêm vào JFrame
- Các tầng được sử dụng để tùy biến hiển thị của cửa sổ.



# JFrame

## ❑ JFrame

- Sử dụng lớp `javax.swing.JFrame`
- Là một cửa sổ có khung, icon, tiêu đề, các nút điều khiển (thu nhỏ, phóng to, đóng cửa sổ).
- Có thể di chuyển, thay đổi kích thước.
- Có thanh menu (tùy chọn), ô nội dung.
- Được dùng để chứa các thành phần khác.



Cấu trúc JFrame

## JFrame

### ❑ Các hàm tạo thông dụng

- **JFrame()**: khởi tạo một frame mới invisible.
- **JFrame(String title)**: khởi tạo một frame mới invisible với tiêu đề được chỉ định.

# JFrame

## ❑ Các phương thức cơ bản

- **setSize(int width, int height)**: đặt kích thước cho JFrame
- **setLocation(int x, int y)**: đặt vị trí cho JFrame (mặc định thì một JFrame sẽ hiển thị ở vị trí góc trên – trái của màn hình)
- **setVisible(boolean b)**: cho phép JFrame ẩn/hiện
- **setDefaultCloseOperation(int operation)**: đặt hành động mặc định sẽ xảy ra khi người dùng “đóng” Frame. Mặc định là **HIDE\_ON\_CLOSE**, các lựa chọn khác gồm **DO NOTHING ON CLOSE**, **DISPOSE ON CLOSE**, **EXIT ON CLOSE**.
- **setTitle(String title)**: đặt tiêu đề cho JFrame
- **setResizable(boolean b)**: đặt JFrame có được thay đổi kích thước hay không
- **getContentPane()**: nhận về ô nội dung của JFrame để thêm các thành phần GUI vào

## JFrame

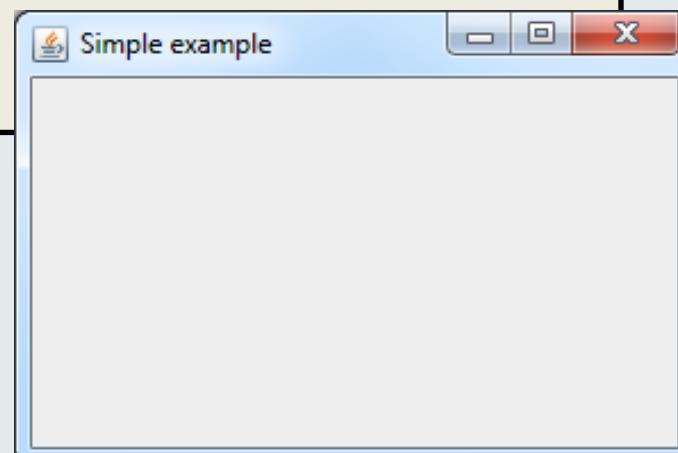
### ❑ Các bước cơ bản để tạo một cửa sổ JFrame

- Khởi tạo một đối tượng của lớp JFrame
- Thiết lập kích thước cho JFrame
- Thiết lập tiêu đề cho JFrame (*nếu không đặt thì thanh tiêu đề sẽ trắng*)
- Thiết lập hành động cho việc “đóng” JFrame
- Cho phép JFrame hiển thị

# Ví dụ JFrame

```
import javax.swing.JFrame;
public class JFrameExample extends JFrame {
    public JFrameExample() {
        setTitle("Simple example");
        setSize(300, 200);
        setLocationRelativeTo(null); ←
        setDefaultCloseOperation(EXIT_ON_CLOSE);
    }
    public static void main(String[] args) {
        JFrameExample ex = new JFrameExample();
        ex.setVisible(true);
    }
}
```

Frame sẽ xuất hiện ở  
giữa màn hình desktop



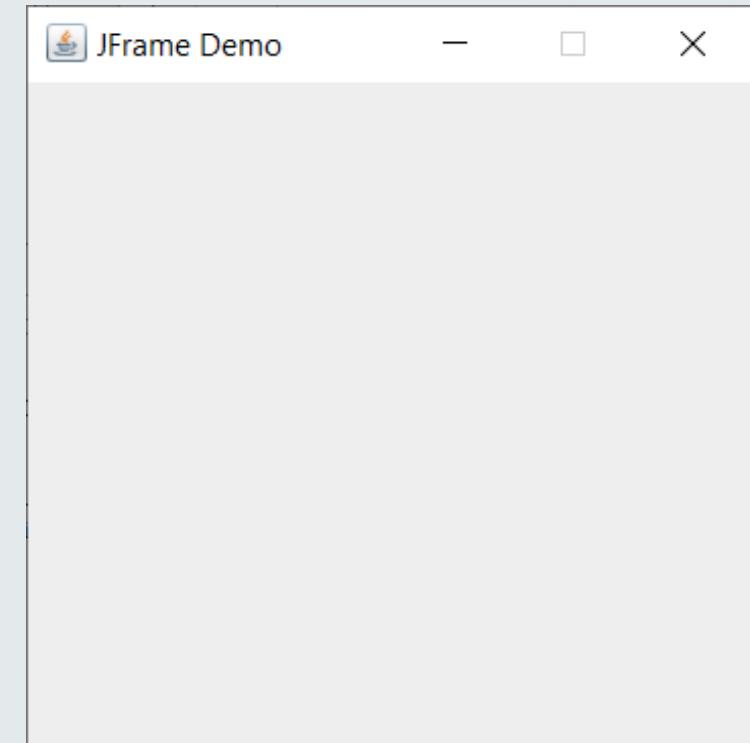
# Ví dụ tạo JFrame đơn giản

```
import javax.swing.JFrame;

public class viDuJFrame {

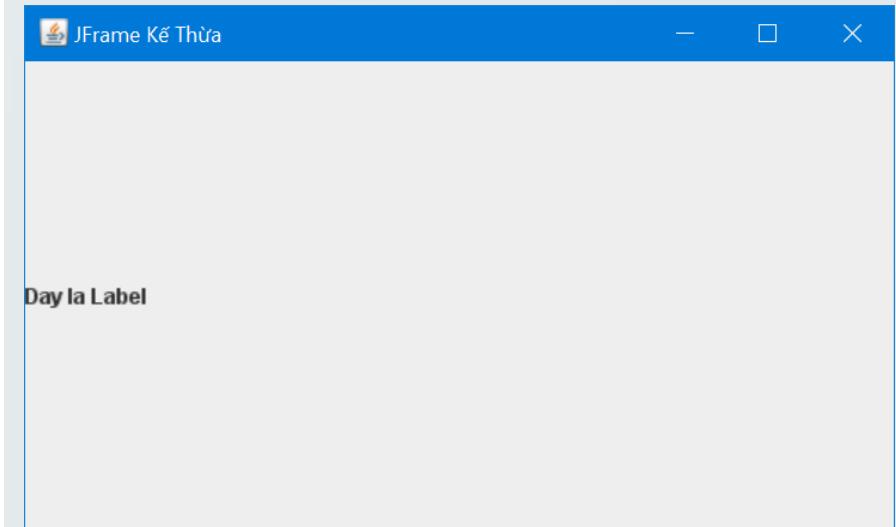
    public viDuJFrame() {
        // khởi tạo frame
        JFrame frame = new JFrame();
        // đặt tiêu đề
        frame.setTitle("JFrame Demo");
        // đặt vị trí hiển thị frame
        frame.setLocation(300, 300);
        // không cho phép thay đổi kích thước
        frame.setResizable(false);
        // thiết lập kích thước cho frame
        frame.setSize(300, 300);
        // thiết lập hành động khi đóng frame
        frame.setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        // hiển thị frame
        frame.setVisible(true);
    }

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



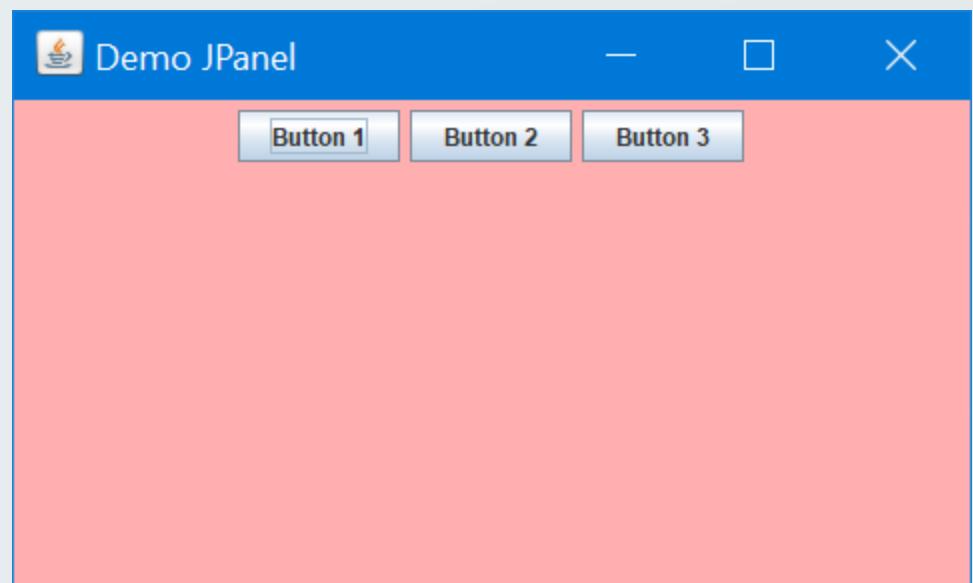
# Ví dụ tạo JFrame đơn giản

```
public class JFrameKeThuaDemo extends JFrame{  
    public JFrameKeThuaDemo() {  
        setTitle("JFrame Kế Thừa");  
        setSize(500, 300);  
        setDefaultCloseOperation(EXIT_ON_CLOSE);  
        setLocationRelativeTo(null);  
        setVisible(true);  
        //  
        JLabel lb = new JLabel();  
        lb.setText("Day la Label");  
        add(lb);  
    }  
    public static void main(String[] args) {  
        new JFrameKeThuaDemo();  
    }  
}
```



## JPanel

- ❑ JPanel là một container trong Swing dùng để chứa và sắp xếp các component khác bên trong nó
- ❑ Tổ chức các thành phần, nhiều bộ cục khác nhau có thể được thiết lập trong JPanel giúp tổ chức các thành phần tốt hơn



# JPanel – Các hàm tạo thông dụng

## ❑ JPanel() :

- Khởi tạo JPanel với FlowLayout.

## ❑ JPanel(LayoutManager l) :

- Khởi tạo một JPanel với Layout được chỉ định.

## ❑ JPanel(boolean isDoubleBuffered) :

- tạo một JPanel mới với một chiết lược đệm cụ thể

## ❑ JPanel(LayoutManager l, boolean isDoubleBuffered) :

- Khởi tạo một JPanel với Layout và chiết lược đệm cụ thể.

# JPanel – Một số phương thức cơ bản

## **add(Component c):**

- Thêm một component vào JPanel

## **setLayout(LayoutManager l) :**

- Điều chỉnh lại layout của JPanel

## **updateUI() :**

- cập nhật thuộc tính giao diện người dùng với một giá trị từ giao diện hiện tại

## **setUI(PanelUI ui) :**

- thiết lập giao diện của một đối tượng hiển thị thành phần này.

## **getUI() :**

- trả về giao diện đối tượng hiển thị thành phần này.

## **paramString() :**

- trả về đại diện chuỗi của JPanel này.

## **getUIClassID() :**

- trả về tên của lớp Giao diện hiển thị thành phần này.

## **getAccessibleContext() :**

- lấy AccessibleContext được liên kết với JPanel này.

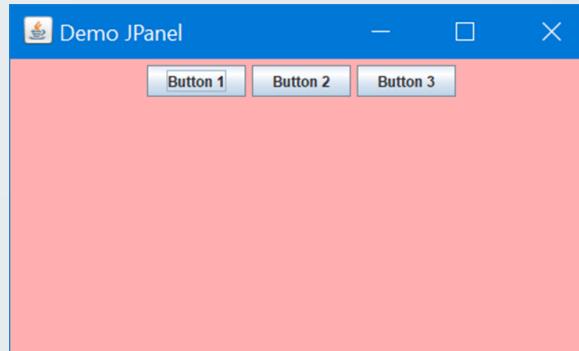
# Ví dụ JPanel đơn giản

```

public class ViduJPanelDonGian extends JFrame{
static JFrame frame;
static JButton button1, button2, button3;

public ViduJPanelDonGian() {
frame = new JFrame("Demo JPanel");
button1 = new JButton("Button 1");
button2 = new JButton("Button 2");
button3 = new JButton("Button 3");
//create JPanel to add button
JPanel pn = new JPanel();
pn.add(button1);
pn.add(button2);
pn.add(button3);
}
}

```



```

//set background
pn.setBackground(Color.PINK);
//add panel to frame
frame.add(pn);
frame.setSize(500, 300);
frame.setVisible(true);
}
}

```

```

public static void main(String[] args) {
new ViduJPanelDonGian();
}
}

```

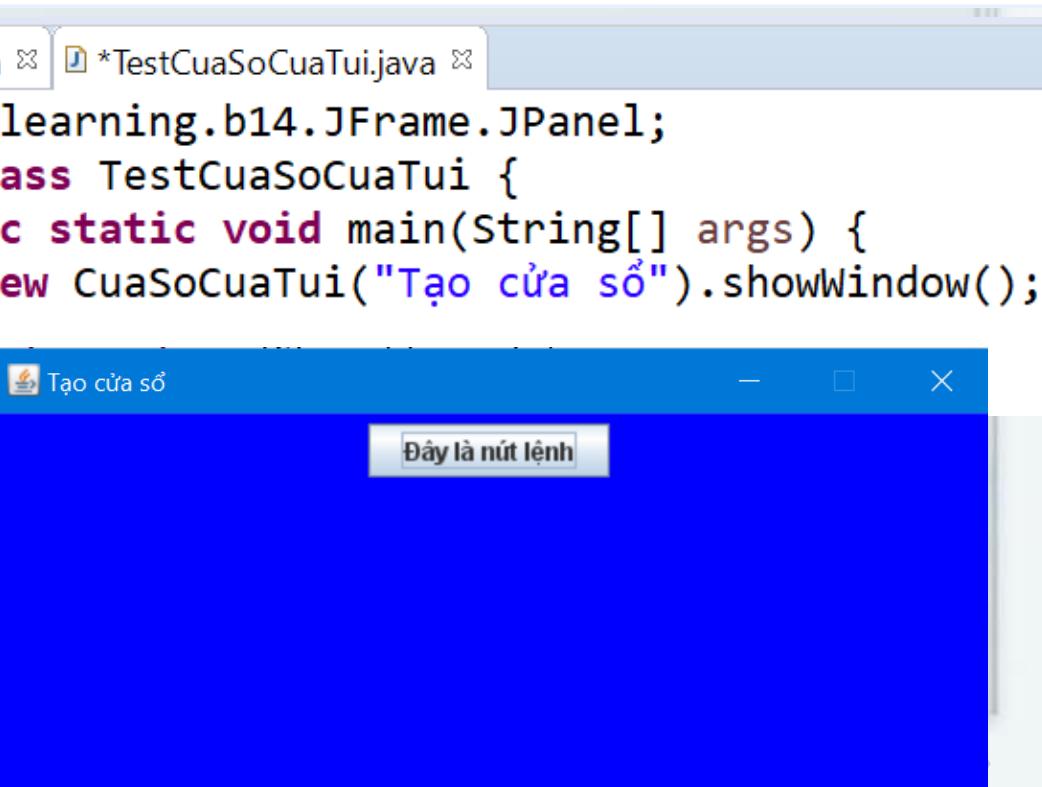
# Ví dụ

```
*CuaSoCuaTui.java ✘
1+import java.awt.*;
2
3 public class CuaSoCuaTui extends JFrame{
4     public CuaSoCuaTui(String tieude) {
5         this.setTitle(tieude);
6         addControls();
7     }
8
9
10    public void addControls() {
11        //Lay lop chua cua so ra
12        Container con = getContentPane();
13        //Tao 1 lop chua Control
14        JPanel pn = new JPanel();
15        pn.setBackground(Color.BLUE);
16
17        JButton btn = new JButton("Đây là nút lệnh");
18        pn.add(btn);
19        //dua panel len giao dien
20        con.add(pn);
21
22    }
23 }
```

```
24+    public void showWindow(){
25        this.setSize(500, 400);
26        this.setDefaultCloseOperation(EXIT_ON_CLOSE);
27        this.setLocationRelativeTo(null);
28        this.setResizable(false);
29        this.setVisible(true);
30    }
31 }
```

\*CuaSoCuaTui.java ✘ \*TestCuaSoCuaTui.java ✘

```
1 package elearning.b14.JFrame.JPanel;
2 public class TestCuaSoCuaTui {
3     public static void main(String[] args) {
4         new CuaSoCuaTui("Tạo cửa sổ").showWindow();
5     }
6 }
```



The screenshot shows two code editors at the top. The left editor contains the code for `CuaSoCuaTui.java`, which creates a blue-colored frame with a single button labeled "Đây là nút lệnh". The right editor contains the code for `TestCuaSoCuaTui.java`, which creates an instance of `CuaSoCuaTui` and calls its `showWindow` method. Below the editors is a screenshot of a Java application window titled "Tạo cửa sổ". Inside the window, there is a single button with the same label: "Đây là nút lệnh".

## Sắp xếp bố cục

❑ AWT và Swing cung cấp nhiều kiểu sắp xếp bố cục (cho phép xác định vị trí và kích thước của các thành phần):

- java.awt.BorderLayout
- java.awt.FlowLayout
- java.awt.GridLayout
- java.awt.GridBagLayout
- javax.swingBoxLayout
- javax.swing.GroupLayout
- javax.swing.SpringLayout

## Sắp xếp bố cục

- Bố cục mặc định là FlowLayout**
- Cài đặt bố cục**
  - Khởi tạo bố cục
  - Sử dụng phương thức **setLayout** của vật chứa để thiết lập bố cục
- Thêm thành phần GUI vào vật chứa với tham số vị trí thích hợp với bố cục được thiết lập ở bước trên.**
- Có thể sử dụng NetBeans, Eclipse để có thể bố trí các thành phần GUI một cách dễ dàng hơn.**

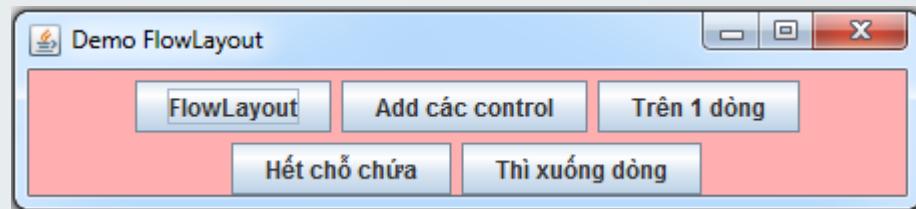
## Các loại Layout Manager quan trọng thường dùng

- Dùng để xác định kích thước và vị trí của các thành phần GUI**
- Mỗi thành phần sẽ có 1 Layout manager mặc định**
- Các Layout manager Java hỗ trợ:**
  - `java.awtFlowLayout` (bố trí dạng dòng chảy)
  - `java.awt.BoxLayout` (bố trí dạng dòng hộp)
  - `java.awt.GridLayout` (bố trí dạng lưới đều nhau)
  - `java.awt.BorderLayout` (bố trí về biên khung)
  - `java.awt.CardLayout` (bố trí dạng lưng quân bài)
  - ....

# Các loại Layout Manager quan trọng thường dùng

## ❑ Flowlayout

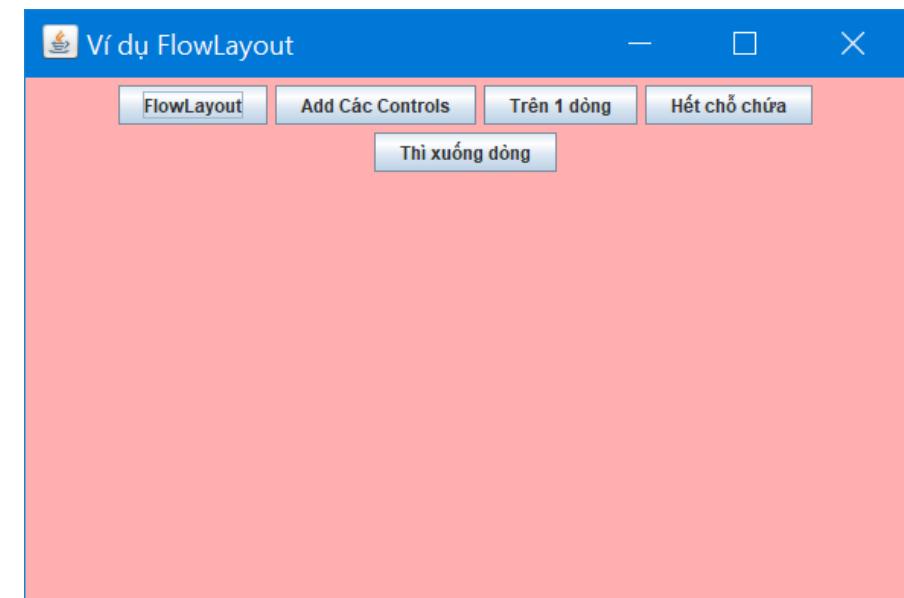
- Là cách bố cục mặc định của mỗi JPanel
- Các thành phần được đặt trong khung theo thứ tự được thêm vào vật chứa
- Nếu vượt quá chiều ngang thì sẽ tự động chuyển xuống hàng
- Mặc định khi một JPanel được khởi tạo thì bản thân lớp chứa này sẽ có kiểu Layout là FlowLayout.



Resize the Width

```
public class ViduFlowLayout extends JFrame{  
public ViduFlowLayout() {  
setTitle("Ví dụ FlowLayout");  
Container con = getContentPane();  
//tạo 1 JPanel:  
JPanel pnFlowLayout=new JPanel();  
pnFlowLayout.setLayout(new FlowLayout());  
JButton btn1=new JButton("FlowLayout");  
JButton btn2=new JButton("Add Các Controls");  
JButton btn3=new JButton("Trên 1 dòng");  
JButton btn4=new JButton("Hết chỗ chứa");  
JButton btn5=new JButton("Thì xuống dòng");  
  
pnFlowLayout.add(btn1);  
pnFlowLayout.add(btn2);  
pnFlowLayout.add(btn3);  
pnFlowLayout.add(btn4);  
pnFlowLayout.add(btn5);  
pnFlowLayout.setBackground(Color.PINK);  
con.add(pnFlowLayout);  
}
```

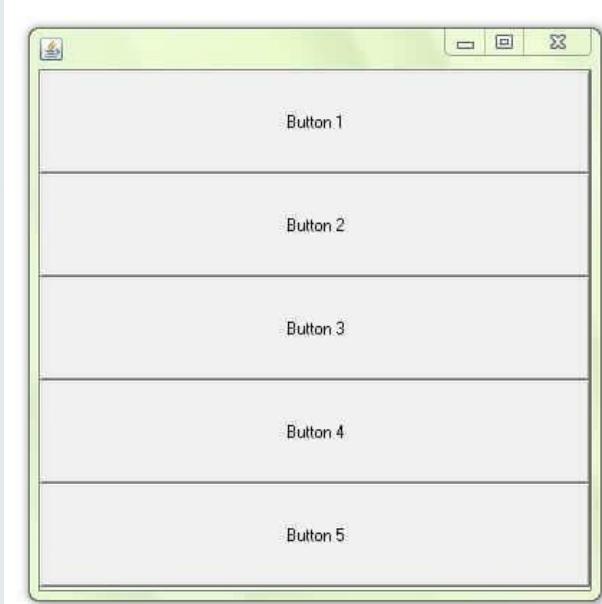
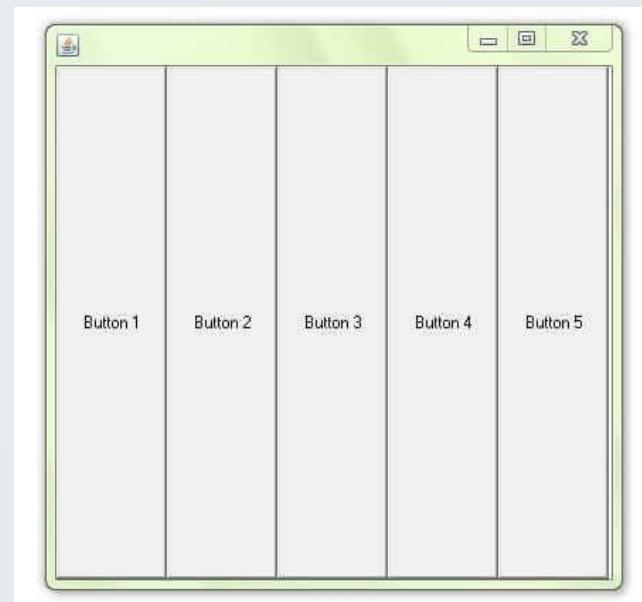
```
public void showWindow(){  
this.setSize(600, 400);  
this.setDefaultCloseOperation(EXIT_ON_CLOSE);  
this.setLocationRelativeTo(null);  
this.setVisible(true);  
}  
  
public static void main(String[] args) {  
new ViduFlowLayout().showWindow();  
}  
}
```



# Các loại Layout Manager quan trọng thường dùng

## Boxlayout

- Được sử dụng để sắp xếp các thành phần hoặc theo chiều dọc hoặc theo chiều ngang.
- Cách bố trí được xác định bởi 4 hàng số
  - X\_AXIS
  - Y\_AXIS
  - LINE\_AXIS
  - PAGE\_AXIS



# Các loại Layout Manager quan trọng thường dùng

## ❑ Boxlayout

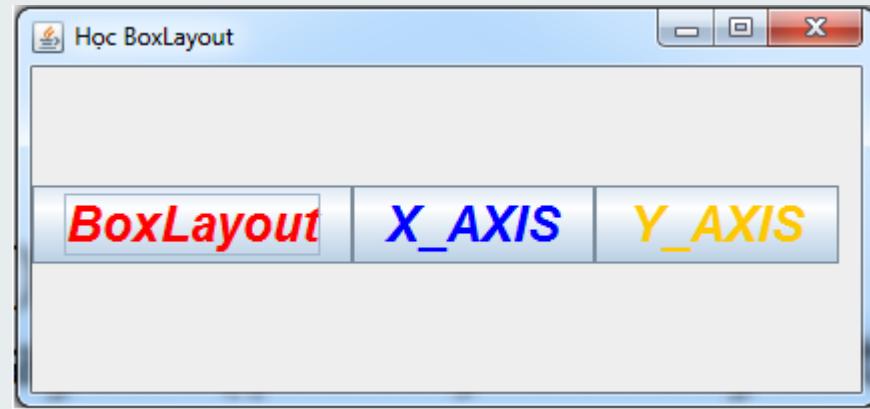
- Cho phép add các control theo dòng hoặc cột
- Tại mỗi vị trí add chỉ chấp nhận 1 control (*do đó muốn xuất hiện nhiều control tại một vị trí thì nên add vị trí đó là 1 JPanel rồi sau đó add các control khác vào JPanel này*)
- **BoxLayout.X\_AXIS** cho phép add các control theo hướng từ trái qua phải
- **BoxLayout.Y\_AXIS** cho phép add các control theo hướng từ trên xuống

**❑ BoxLayout sẽ không tự động xuống dòng khi hết chỗ chứa, tức là các control sẽ bị che khuất nếu như thiếu không gian chứa nó**

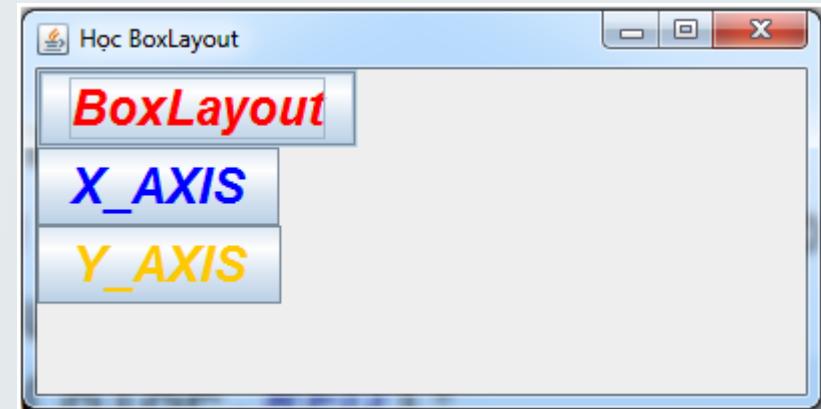
# Các loại Layout Manager quan trọng thường dùng

## Boxlayout

### BoxLayout.X\_AXIS



### BoxLayout.Y\_AXIS



No wrap row when  
resize dimension

# Các loại Layout Manager quan trọng thường dùng

## ❑ Boxlayout

```
JPanel pnBox=new JPanel();
pnBox.setLayout(new BoxLayout(pnBox, BoxLayout.X_AXIS));
JButton btn1=new JButton("BoxLayout");
btn1.setForeground(Color.RED);
Font font=new Font("Arial",Font.BOLD | Font.ITALIC,25);
btn1.setFont(font);pnBox.add(btn1);
JButton btn2=new JButton("X_AXIS");
btn2.setForeground(Color.BLUE);
btn2.setFont(font);pnBox.add(btn2);
JButton btn3=new JButton("Y_AXIS");
btn3.setForeground(Color.ORANGE);
btn3.setFont(font);pnBox.add(btn3);

Container con=getContentPane();
con.add(pnBox);
```

## ❑ Boxlayout

```
pnBox.setLayout(new BoxLayout(pnBox, BoxLayout.X_AXIS));
```

*Setup BoxLayout for pnBox with X\_AXIS*

```
btn1.setForeground(Color.RED);
```

*Setup TextColor for button btn1*

```
Font font=new Font("Arial",Font.BOLD | Font.ITALIC,25);
```

*creat font object with Font.BOLD and Font.ITALIC*

```
btn1.setFont(font);
```

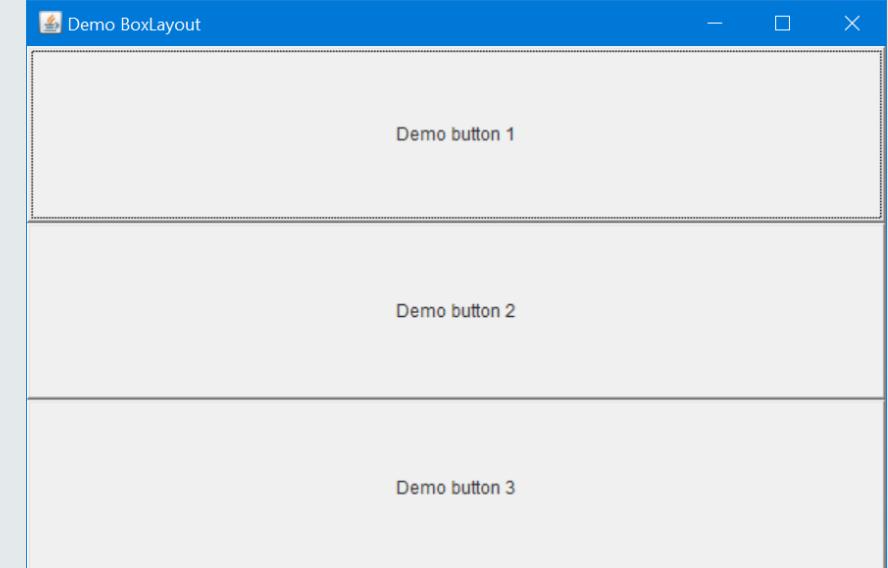
*setup font for button btn1*

# Ví dụ BoxLayout

```
public class BoxLayoutDemo extends JFrame {  
    public BoxLayoutDemo(String tieude) {  
        this.setTitle(tieude);  
        addControls();  
    }  
    public void addControls(){  
        //  
    }  
    public void showWindow(){  
        this.setSize(600, 400);  
        this.setDefaultCloseOperation(EXIT_ON_CLOSE);  
        this.setLocationRelativeTo(null);  
        this.setVisible(true);  
    }  
}
```



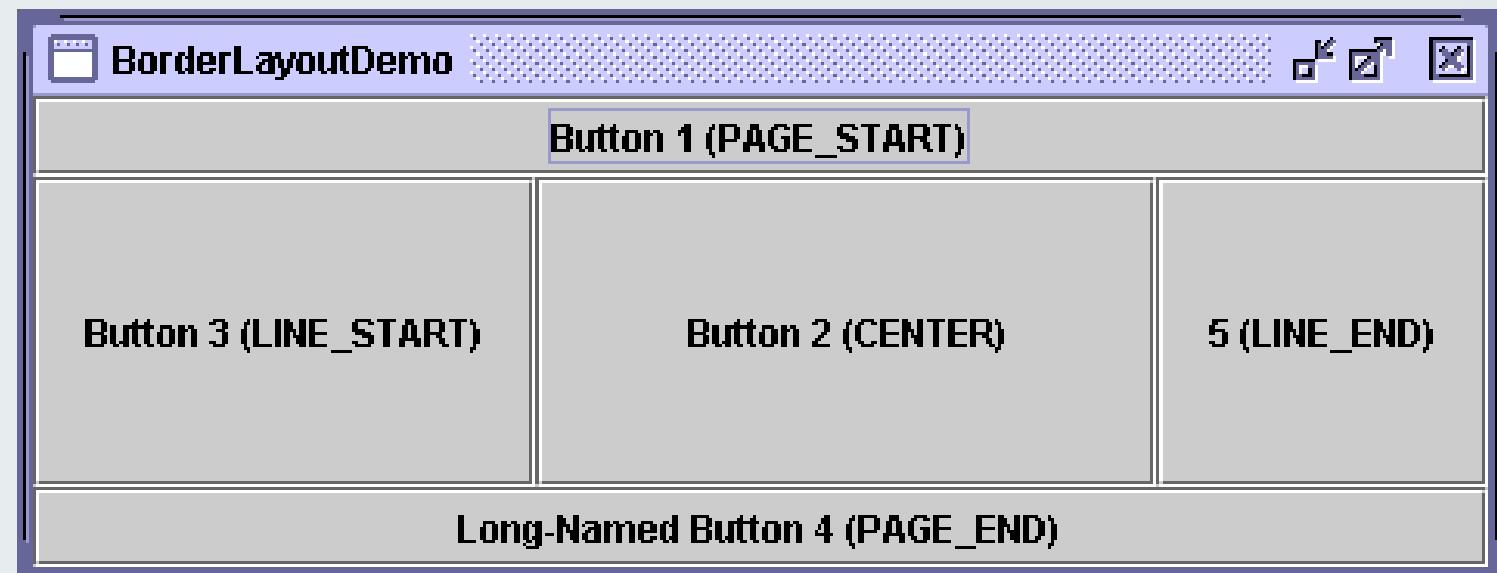
```
public void addControls(){  
    Container con = getContentPane();  
    JPanel pnBox=new JPanel();  
    pnBox.setLayout(new BoxLayout(pnBox, BoxLayout.Y_AXIS));  
    Button btn1=new Button("Demo button 1");  
    Button btn2=new Button("Demo button 2");  
    Button btn3=new Button("Demo button 3");  
  
    pnBox.add(btn1);  
    pnBox.add(btn2);  
    pnBox.add(btn3);  
  
    con.add(pnBox);  
}
```



# Các loại Layout Manager quan trọng thường dùng

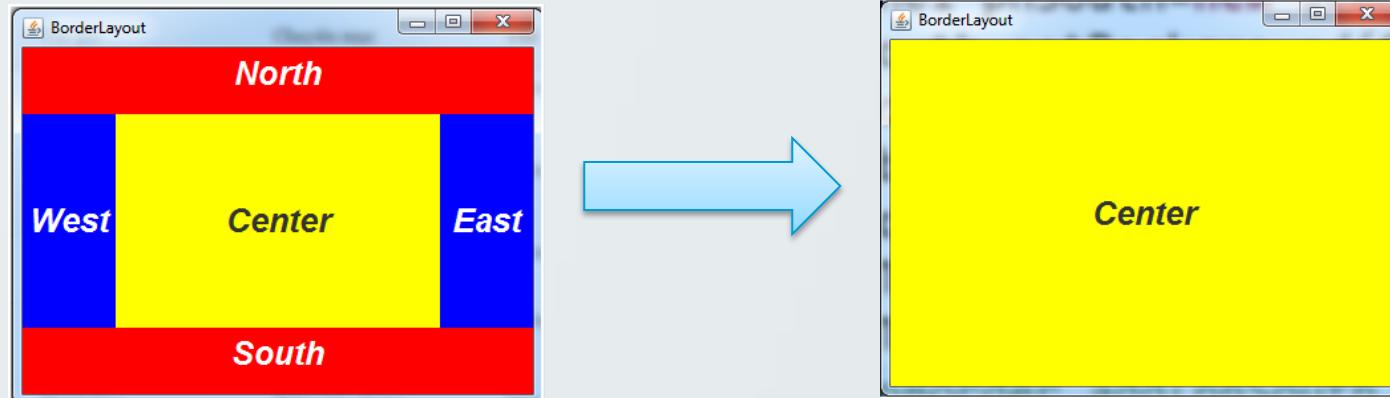
## ❑ Borderlayout

- Mỗi contentpane luôn được khởi tạo với bố cục BorderLayout
- JToolBar khi tạo ra phải thuộc BorderLayout
- Có 5 vị trí: xung quanh và ở giữa



## ❑ Borderlayout

- Giúp hiển thị các control theo 5 vùng: North, South, West, East, Center
- 



Nếu như không có 4 vùng : North, West, South, East. Thì vùng Center sẽ tràn đầy cửa sổ, thông thường khi đưa các control JTable, JTree, ListView, JScrollPane... ta thường đưa vào vùng Center để nó có thể tự co giãn theo kích thước cửa sổ giúp giao diện đẹp hơn.

## ❑ Borderlayout

```
JPanel pnBorder=new JPanel();
pnBorder.setLayout(new BorderLayout());
JPanel pnNorth=new JPanel();
pnNorth.setBackground(Color.RED);
pnBorder.add(pnNorth,BorderLayout.NORTH);
JPanel pnSouth=new JPanel();
pnSouth.setBackground(Color.RED);
pnBorder.add(pnSouth,BorderLayout.SOUTH);
JPanel pnWest=new JPanel();
pnWest.setBackground(Color.BLUE);
pnBorder.add(pnWest,BorderLayout.WEST);
JPanel pnEast=new JPanel();
pnEast.setBackground(Color.BLUE);
pnBorder.add(pnEast,BorderLayout.EAST);
JPanel pnCenter=new JPanel();
pnCenter.setBackground(Color.YELLOW);
pnBorder.add(pnCenter,BorderLayout.CENTER);
getContentPane().add(pnBorder);
```

## ❑ Borderlayout

```
pnBorder.setLayout(new BorderLayout());
```

**Setup BorderLayout for pnBorder**

```
pnBorder.add(pnNorth,BorderLayout.NORTH);
```

**Add pnNorth into the NORTH side**

```
pnBorder.add(pnSouth,BorderLayout.SOUTH);
```

**Add pnSouth into the SOUTH side**

```
pnBorder.add(pnWest,BorderLayout.WEST);
```

**Add pnWest into the WEST side**

```
pnBorder.add(pnEast,BorderLayout.EAST);
```

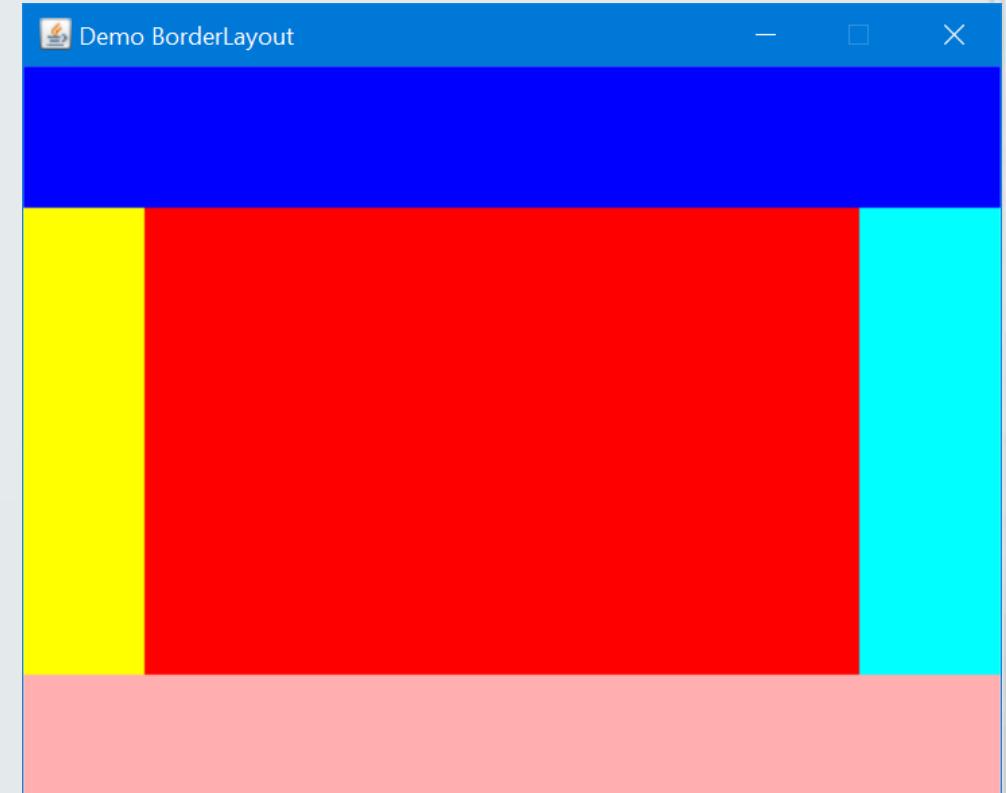
**Add pnEast into the EAST side**

```
pnBorder.add(pnCenter,BorderLayout.CENTER);
```

**Add pnCenter into the CENTER side**

# Ví dụ BorderLayout

```
public class BorderLayoutDemo extends JFrame{  
    public BorderLayoutDemo(String tieude) {  
        //sv code  
    }  
    public void AddControls(){  
        //code ở đây  
    }  
    public void showWindow(){  
        //sv code  
    }  
}  
  
public static void main(String[] args) {  
    new BorderLayoutDemo("Demo BorderLayout").showWindow();  
}
```

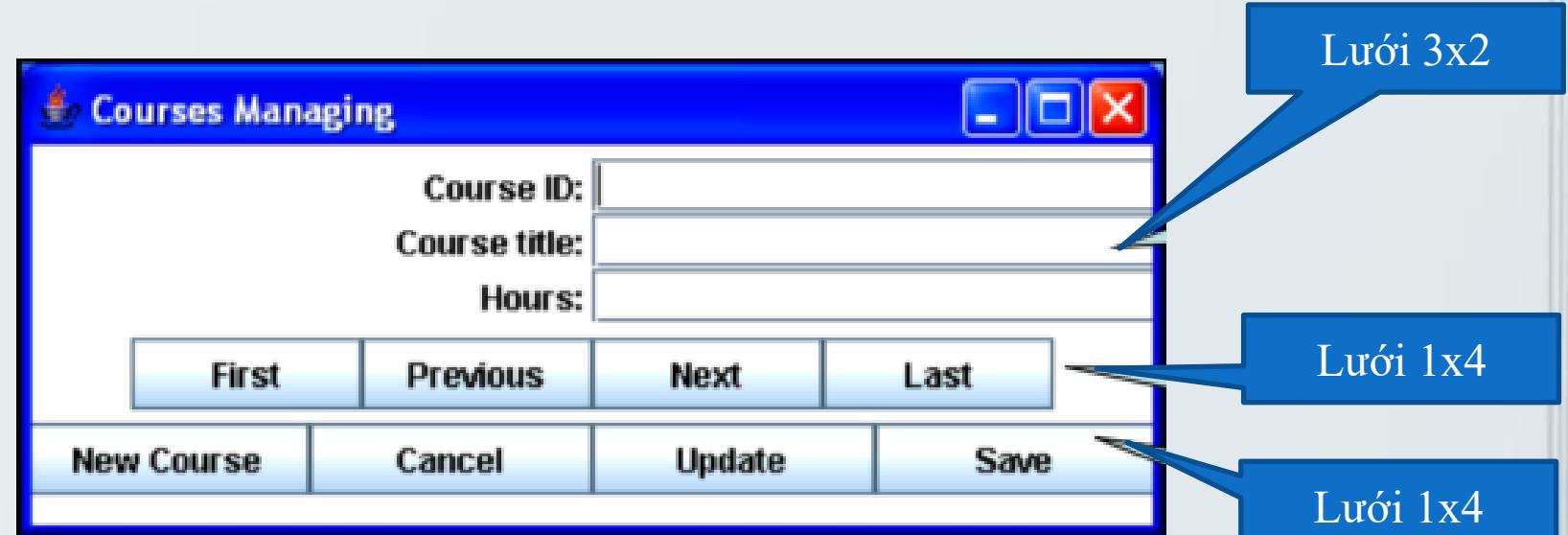


# Ví dụ BorderLayout

```
public void AddControls(){
    Container con = getContentPane();
    JPanel pnBorderLayout = new JPanel();
    pnBorderLayout.setLayout(new BorderLayout());
    JPanel pnNorth = new JPanel();
    pnNorth.setBackground(Color.BLUE);
    pnBorderLayout.add(pnNorth,BorderLayout.NORTH);
    pnNorth.setPreferredSize(new Dimension(0, 70));
    JPanel pnSouth=new JPanel();
    pnSouth.setBackground(Color.PINK);
    pnBorderLayout.add(pnSouth,BorderLayout.SOUTH);
    pnSouth.setPreferredSize(new Dimension(0, 60));
    JPanel pnWest=new JPanel();
    pnWest.setBackground(Color.YELLOW);
    pnBorderLayout.add(pnWest,BorderLayout.WEST);
    pnWest.setPreferredSize(new Dimension(60, 0));
    JPanel pnEast=new JPanel();
    pnEast.setBackground(Color.CYAN);
    pnBorderLayout.add(pnEast,BorderLayout.EAST);
    pnEast.setPreferredSize(new Dimension(70, 0));
    JPanel pnCenter=new JPanel();
    pnCenter.setBackground(Color.RED);
    pnBorderLayout.add(pnCenter,BorderLayout.CENTER);
    con.add(pnBorderLayout);
}
```

## ❑ GridLayout

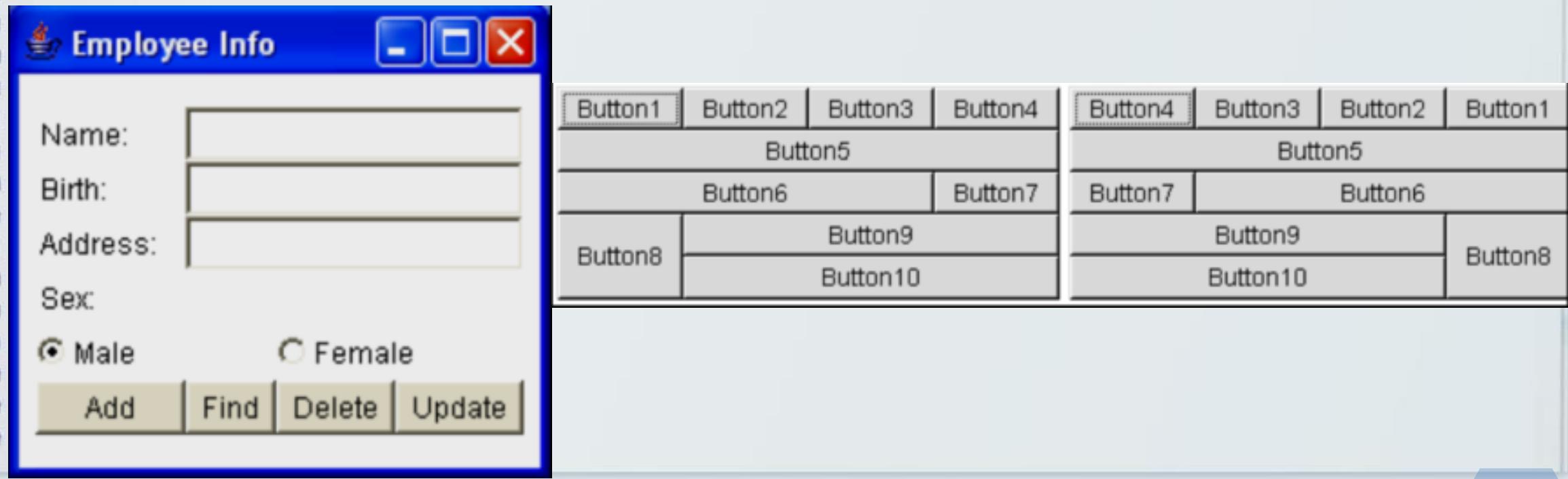
- Kiểu bố trí các component thành 1 lưới rows, cols đều nhau
- Thêm các đối tượng tuần tự từ trái sang phải, từ trên xuống dưới vào các ô định sẵn



## □ Constructor

- **GridLayout():** Tạo grid layout mặc định 1x1
- **GridLayout(int rows, int cols)** Tạo grid layout rows x cols
- **GridLayout(int rows, int cols, int hgap, int vgap)**

- ❑ Thêm các đối tượng vào các ô lưới đã định sẵn, nhưng **cho phép người dùng mở rộng chỗ chứa cho các component** (không chỉ 1 ô)
- ❑ Layout dạng lưới cho phép 1 component chiếm 1 số ô kề nhau theo cả 2 chiều.



- ❑ Constructor: **GridLayout()**
- ❑ Áp đặt GridLayout cho 1 container

```
GridLayout gb= new GridLayout();
```

```
FrameName.setLayout(gb);
```

```
PanelName.setLayout(gb);
```

- ❑ Viết ngắn gọn

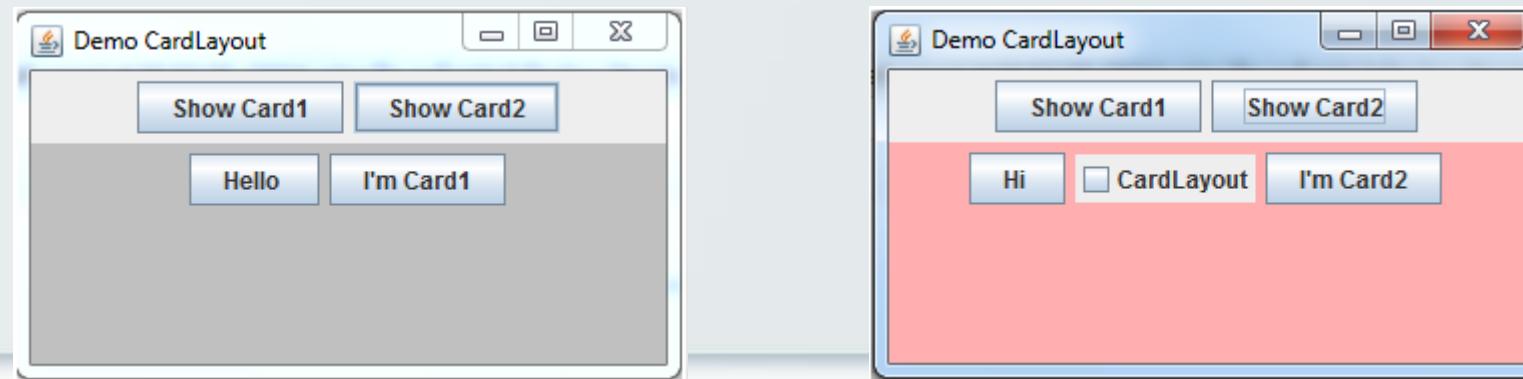
```
FrameName.setLayout(new GridLayout());
```

```
PanelName.setLayout(new GridLayout());
```

## ❑ CardLayout

- Bố trí các component thành từng lớp như lưng các quân bài (card).
- Thường dùng Panel để chứa các component
- Tại 1 thời điểm chỉ có 1 panel hiện hành (quân bài trên cùng).
- Có thể chuyển qua lại giữa các Panel.

CardLayout cho phép chia sẻ vị trí hiển thị của các control, tức là ứng với cùng 1 vị trí hiển thị đó thì ta có thể cho các control khác hiển thị tại những thời điểm khác nhau, mặc định control được add đầu tiên sẽ hiển thị.



## □ CardLayout

```
final JPanel pnCenter=new JPanel();
pnCenter.setLayout(new CardLayout());
final JPanel pnCard1=new JPanel();
pnCard1.setBackground(Color.LIGHT_GRAY);
final JPanel pnCard2=new JPanel();
pnCard2.setBackground(Color.PINK);
pnCenter.add(pnCard1, "mycard1");
pnCenter.add(pnCard2, "mycard2");
btnShowCard1.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent arg0) {
    CardLayout cl=(CardLayout)pnCenter.getLayout();
    cl.show(pnCenter, "mycard1");
}});
```

## ❑ CardLayout

```
pnCenter.setLayout(new CardLayout());
```

**Setup CardLayout for pnCenter**

```
pnCenter.add(pnCard1, "mycard1");
```

**Add pnCard1 into pnCenter with mycard1 name**

```
CardLayout cl=(CardLayout)pnCenter.getLayout();
```

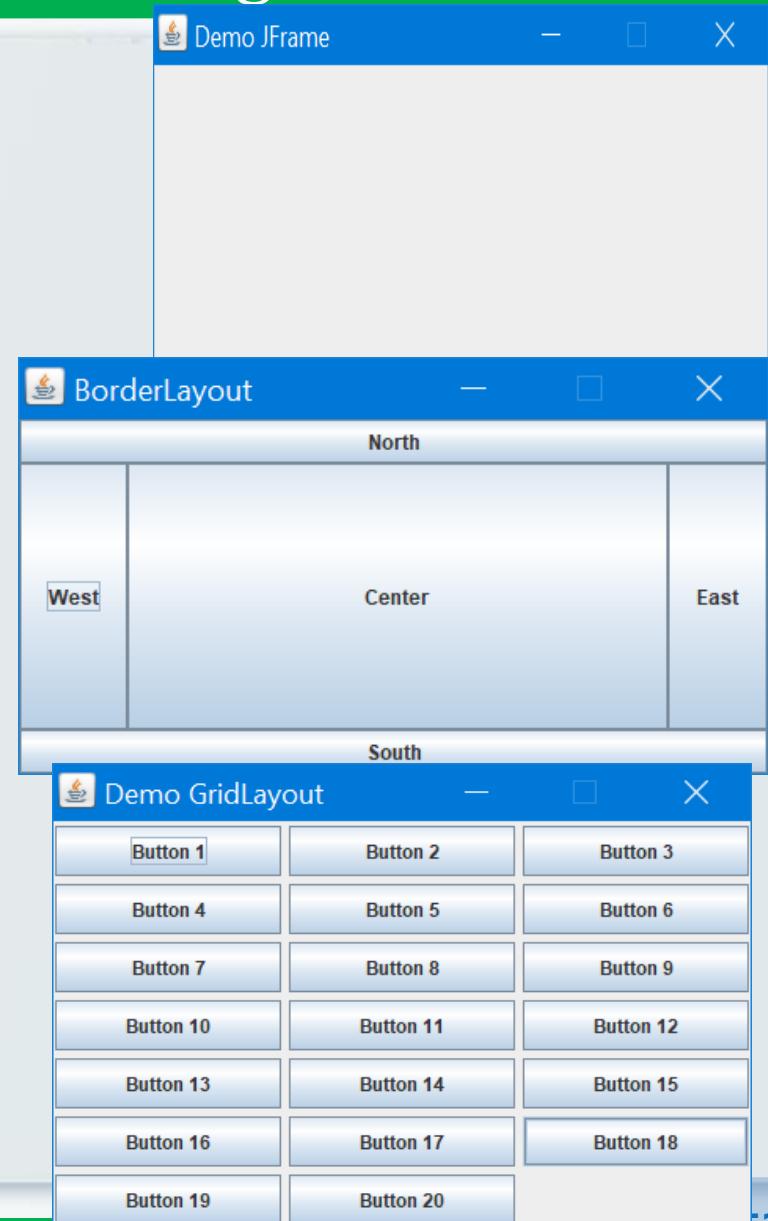
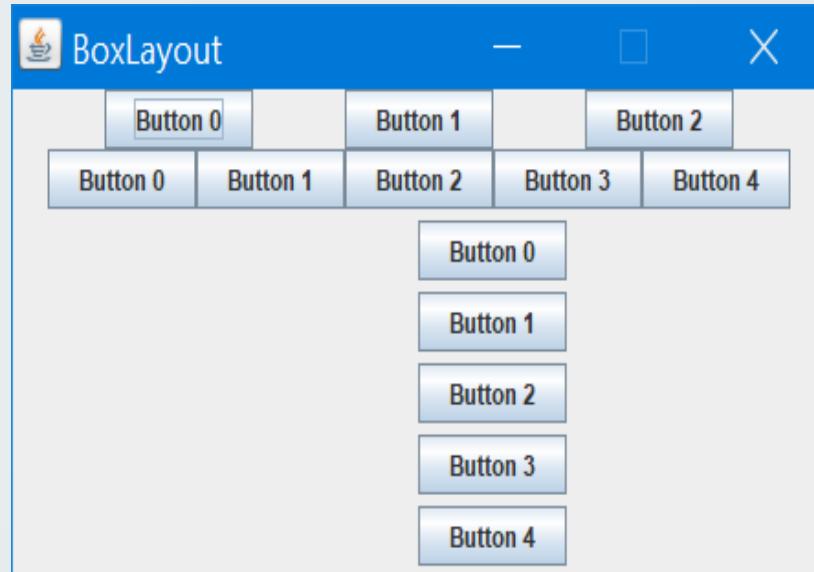
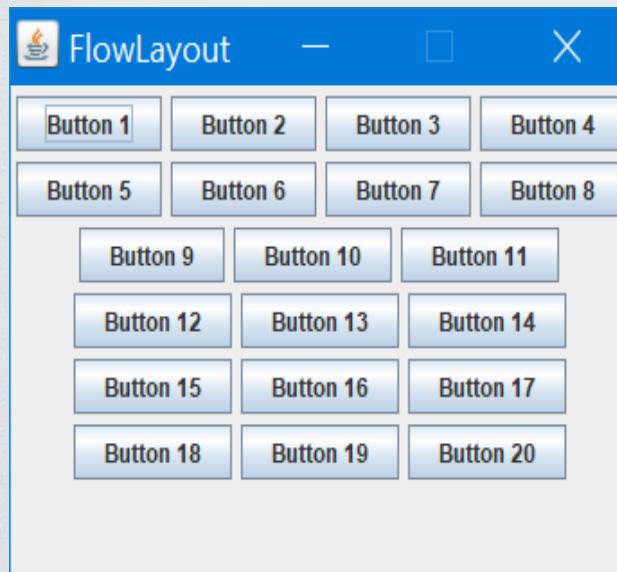
**get CardLayout from pnCenter**

```
cl.show(pnCenter, "mycard1");
```

**Show component with mycard1 name, that  
we define from above**

# Bài tập Thực hành Layout Manager

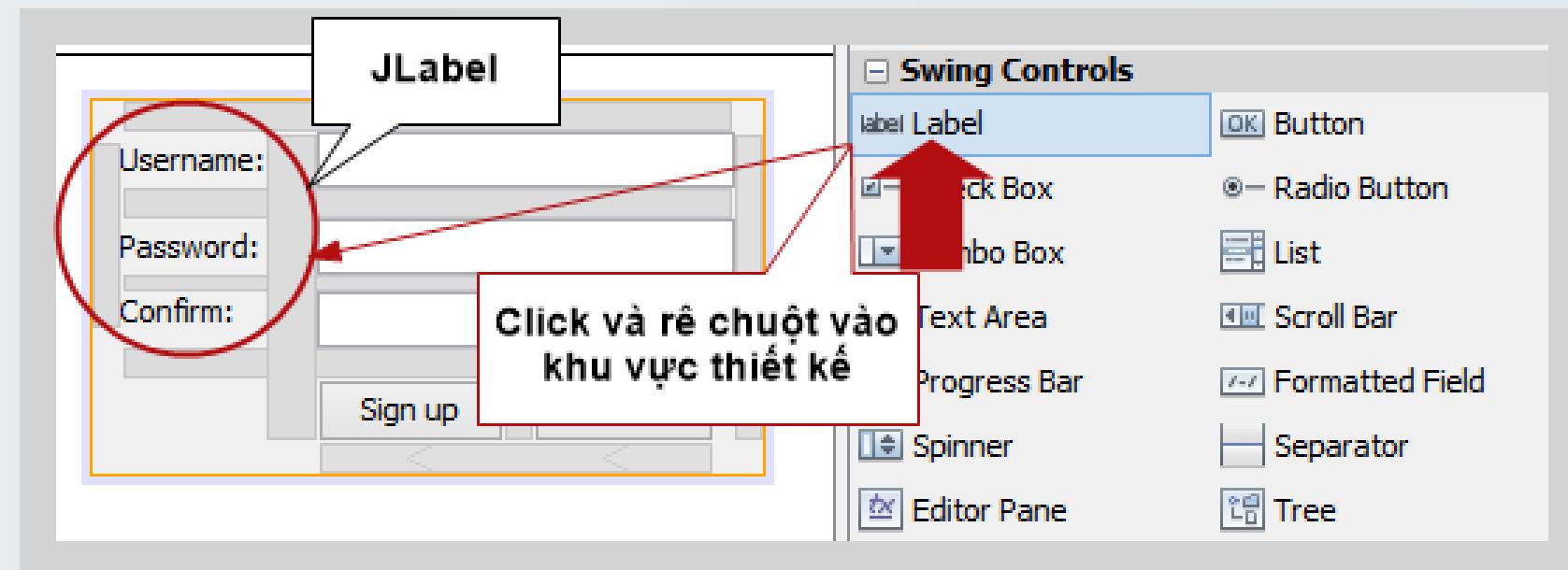
- 1) Thực hành Container – Hiển thị Jframe
- 2) Thực hành LayoutManager – FlowLayout
- 3) Thực hành Layout Manager – BoxLayout
- 4) Thực hành Layout Manager – BorderLayout
- 5) Thực hành Layout Manager – GridLayout



- ✓ JLabel
- ✓ JTextField
- ✓ JButton

- Các kỹ thuật gán sự kiện

- Được dùng để **tạo một nhãn văn bản**, có nguồn gốc từ JComponent
- Cung cấp **text** chỉ dẫn hoặc thông tin trên giao diện người dùng
- Hiển thị một dòng đơn text chỉ đọc
- Một hình ảnh hoặc cả text hoặc hình ảnh



**❑ JLabel():**

- Tạo một thẻ hiện JLabel không có hình ảnh và một chuỗi rỗng

**❑ JLabel(Icon image, int horizontalAlignment):**

- Tạo một thẻ hiện JLabel chỉ định một hình ảnh và horizontal alignment

**❑ JLabel(String text):**

- Tạo một thẻ hiện JLabel chỉ định text

**❑ JLabel(String text, Icon icon, int horizontalAlignment):**

- Tạo một thẻ hiện JLabel chỉ định text, image và horizontal alignment

**❑ JLabel(String text, int horizontalAlignment):**

- Tạo một thẻ hiện JLabel chỉ định text và horizontal alignment

## ❑ **setText(String text):**

- Đặt giá trị text cho JLabel

## ❑ **getText():**

- Lấy giá trị text của JLabel

## ❑ **setToolTipText(String Text):**

- Đặt tooltip cho Jlabel (Khi di chuột trên Label sẻ hiển thị text tip)

## ❑ **setForeground(Color fg):**

- Đặt màu cho chữ

## ❑ **setIcon(Icon icon) :**

- Đặt icon cho Jlabel

## ❑ **setSize(int width, int height) :**

- Đặt kích thước cho JLable

## □ JLabel

```
//Display Text, not editable  
JLabel lbl=new JLabel(“Tách cafe”);  
lbl.setForeground(Color.RED);
```

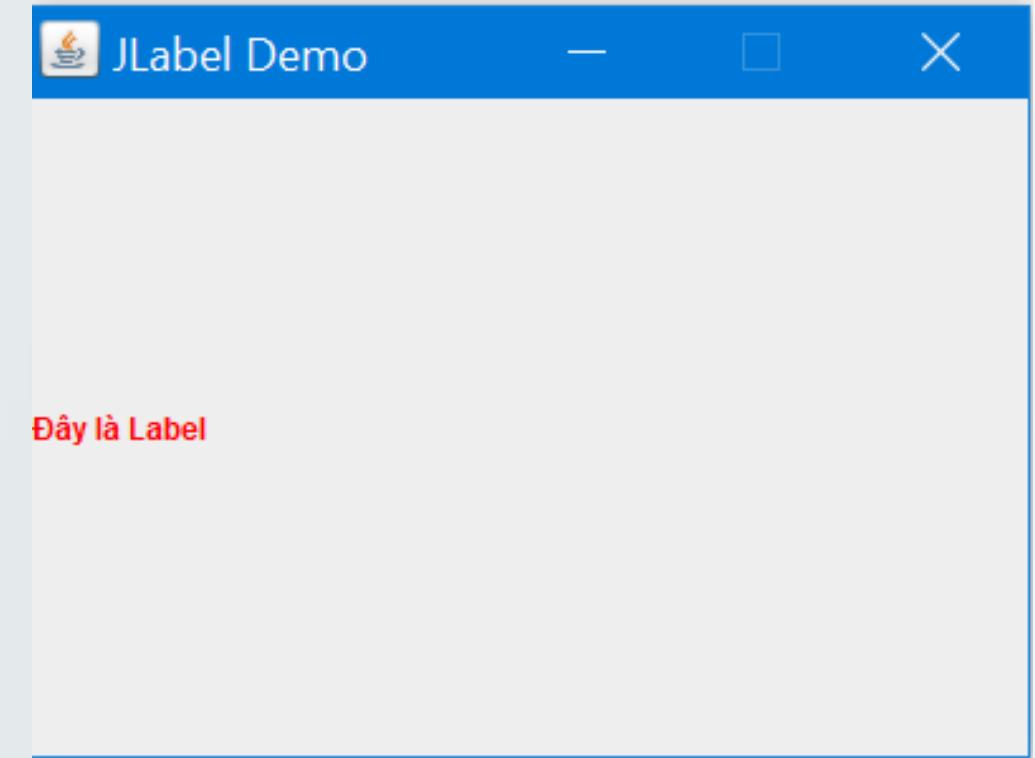
Tách cafe

```
//Could add Icon for JLabel  
ImageIcon icon=new ImageIcon(“cf.png”);  
lbl.setIcon(icon);
```



# Ví dụ 1 Tạo JLabel đơn giản

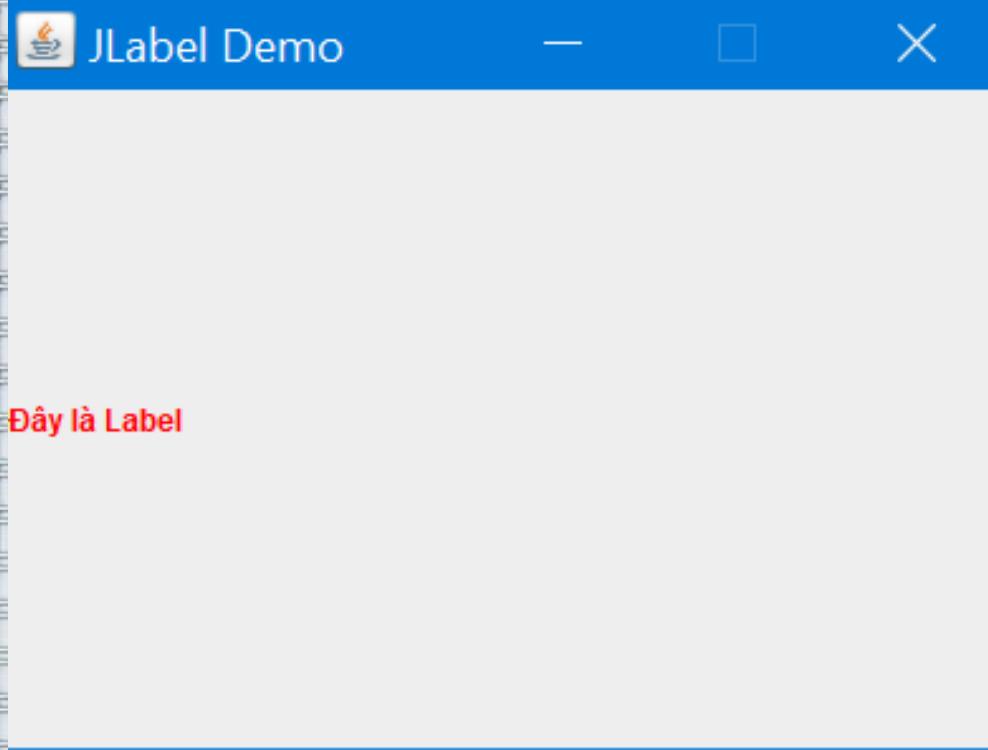
```
public class ViduJLabel extends JFrame{  
    public ViduJLabel() {  
        //create JFrame  
        //create Jlabel  
        ...  
        add(label);  
    }  
    public static void main(String[] args) {  
        ViduJLabel lableDemo = new ViduJLabel();  
    }  
}
```



# Ví dụ 1 Tạo Jlabel đơn giản

```
public class ViduJLabel extends JFrame{  
  
    public ViduJLabel() {  
        // Create JFrame  
        setSize(450, 300);  
        setVisible(true);  
        setLocationRelativeTo(null);  
        //setDefaultCloseOperation(EXIT_ON_CLOSE);  
        //setLocation(500, 500);  
        setResizable(false);  
        setTitle("JLabel Demo");  
        //create JLabel  
        JLabel label = new JLabel();  
        // setText  
        label.setText("Đây là Label");  
        // set Tool tip  
        label.setToolTipText("Tooltip");  
        // set foreground  
        label.setForeground(Color.RED);  
        add(label);  
    }  
  
    public static void main(String[] args) {  
        ViduJLabel lableDemo = new ViduJLabel();  
    }  
}
```

## Ví dụ 2 Tạo Jlabel đơn giản



```
public class ViduJLabel2 {  
    public ViduJLabel2(String title) {  
        //create JFrame  
        //create JLabel  
    }  
    public static void main(String[] args) {  
        new ViduJLabel2("JLabel Demo");  
    }  
}
```

## Ví dụ 2 Tạo JLabel đơn giản

```
import javax.swing.JLabel;
public class ViduJLabel2 {
public ViduJLabel2(String title) {
//create JFrame
JFrame frame = new JFrame();
frame.setTitle(title);
frame.setResizable(false);
frame.setSize(400, 300);
frame.setDefaultCloseOperation(
javax.swing.WindowConstants.EXIT_ON_CLOSE);
frame.setLocationRelativeTo(null);
frame.setVisible(true);
//Create JLabel
JLabel label = new JLabel();
label.setSize(450, 300);
label.setVisible(true);
// setText
label.setText("Đây là Label");
// set Tool tip
label.setToolTipText("Tooltip");
// set foreground
label.setForeground(Color.RED);
//add(label);
frame.add(label);
}
public static void main(String[] args) {
new ViduJLabel2("JLabel Demo");
}
}
```

# JLabel hiển thị hình ảnh

- Sinh viên thực hiện kế một Jlabel theo mẫu như yêu cầu, Jlabel đổi tên lại thành tên Sinh viên



//tao image

```
Icon icon = new ImageIcon(getClass().getResource("cf.PNG"));
```

```
//create JFrame
```

```
setLayout(new GridLayout(1, 2, 5, 5));
```

```
Doituong.setFont(new Font("VNI",Font.PLAIN,24));  
Doituong.setForeground(Color.blue);
```

- Là đối tượng cho phép người dùng nhập một dòng văn bản**
- Cho phép chỉnh sửa/hiển thị một dòng text đơn**
- Có thể căn lề trái, phải hoặc giữa hay đặt font cho text**
- Cung cấp nhiều event listener**

**JTextField:**

- tạo mới 1 JTextField

 **JTextFiled(Document doc, String text, int columns):**

- Tạo 1 JTextField sử dụng mô hình lưu trữ văn bản với đoạn text và số cột (columns)

 **JTextFiled(int columns):**

- Tạo JTextField trống với độ rộng là columns

 **JTextField(String text):**

- Tạo JTextField với text cho trước

 **TextField(String text, int columns):**

- Tạo JTextField với text và độ rộng cho trước.

## ❑ **getText():**

- Lấy dữ liệu là một chuỗi

## ❑ **setText(String text):**

- Đặt dữ liệu

## ❑ **requestFocus():**

- giúp con trỏ nhảy (tập trung) đến JTextField đó.

## ❑ **JTextField(String text):**

- Tạo JTextField với text cho trước

## ❑ **setEditable(boolean edit):**

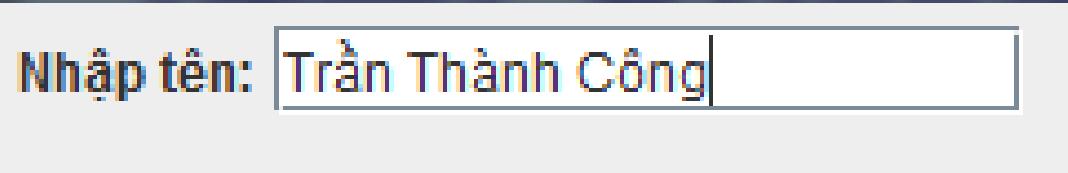
- để đặt có được nhập dữ liệu cho JTextField hay không.



## ❑ JTextField

Display data, Input data

```
getContentPane().setLayout(new FlowLayout());  
JLabel lblTen=new JLabel("Nhập tên:");  
JTextField txtTen=new JTextField(15);  
add(lblTen);add(txtTen);
```



Nhập tên: Trần Thành Công

If you don't allow input data, please call setEditable method and take **false** value

```
txtTen.setEditable(false);
```

## ❑ JTextField

To set Text in code behind for JTextField:

```
txtTen.setText("Hello Tèo");
```

To get Text from JTextField:

```
String s=txtTen.getText();
```

We could convert data:

```
int n=Integer.parseInt(s);  
double d=Double.parseDouble(s);  
float f=Float.parseFloat(s);
```

To set focus:txtTen.requestFocus();

# Ví dụ JTextField



```
public class ViduJTextField extends JFrame{  
    public ViduJTextField() { //sv tự code}  
    public void showWindow() { //sv tự code}  
    public static void main(String[] args) {  
        new ViduJTextField().showWindow();  
    }  
}
```

# Ví dụ JTextField

```
public VíduJTextField() {  
    Container con = getContentPane();  
    JPanel pn = new JPanel();  
    pn.setLayout(new FlowLayout());  
    //  
    JLabel lb = new JLabel("Nhập họ tên: ");  
    lb.setForeground(Color.RED);  
    lb.setSize(150, 180);  
    lb.setFont(new Font("VNI", Font.PLAIN, 18));  
  
    JTextField tf = new JTextField(30);  
    tf.setSize(150, 180);  
    tf.setFont(new Font("VNI", Font.PLAIN, 18));  
    tf.setForeground(Color.BLUE);  
  
    pn.add(lb);  
    pn.add(tf);  
    //  
    con.add(pn);  
}  
  
private void showWindow() {  
    setTitle("Demo JTextField");  
    setSize(600, 300);  
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
    setLocationRelativeTo(null);  
    setVisible(true);  
}
```

## ❑ Sử dụng lớp javax.swing.JDialog.

- Cửa sổ dạng hộp thoại thông báo (có khung viền và thanh tiêu đề).
- Có thể được khởi tạo dạng modal hoặc không
- Dạng “modal”: khi hiển thị hộp thoại thì khóa truy xuất của người dùng đến các cửa sổ khác

**❑ JDialog(Frame owner):**

- tạo dialog dạng không “modal” với vật chủ frame được chỉ định, không có tiêu đề.

**❑ JDialog(Frame owner, boolean modal):**

- có thể tạo dialog dạng “modal” với vật chủ frame được chỉ định, không có tiêu đề.

**❑ JDialog(Frame owner, String title):**

- tạo dialog dạng không “modal” với vật chủ frame được chỉ định và tiêu đề được chỉ định.

**❑ JDialog(Frame owner, String title, boolean modal):**

- có thể tạo dialog dạng “modal” với vật chủ frame được chỉ định và có tiêu đề được chỉ định.

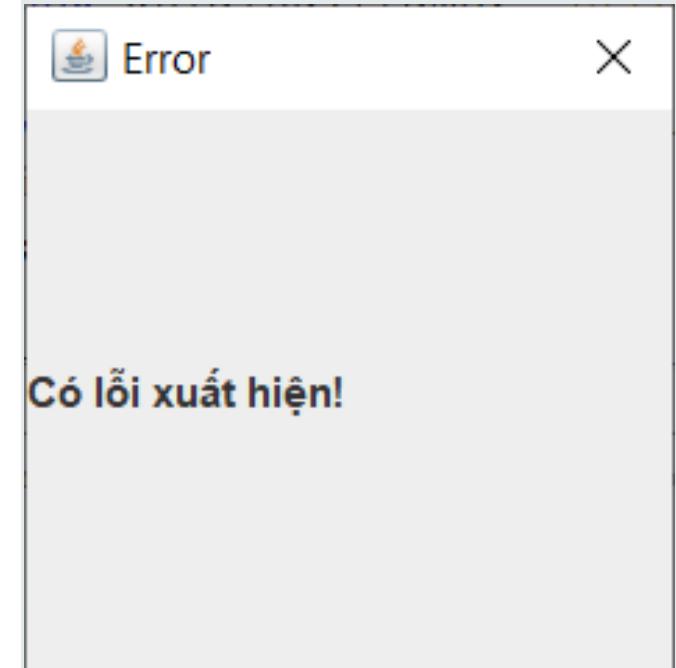
**❑ Một số phương thức cơ bản (Như phương thức của JFrame)**

# Tạo JDialog hiển thị thông báo đơn giản

```
import javax.swing.*;
public class viDuJDialog {

    public viDuJDialog(JFrame frame) {
        // Tạo dialog với tiêu đề dạng modal
        JDialog dialog = new JDialog(frame, "Error", true);
        // tạo nhãn label
        JLabel label = new JLabel("Có lỗi xuất hiện!");
        //thêm nhãn vào dialog
        dialog.add(label);
        // thiết lập kích thước cho dialog
        dialog.setSize(200, 200);
        // hiển thị dialog
        dialog.setVisible(true);
    }

    public static void main(String args[]) {
        JFrame frame = new JFrame(); //tạo frame làm vật chủ cho dialog
        viDuJDialog dialog = new viDuJDialog(frame);
    }
}
```



- ❑ Là đối tượng cho phép người dùng khi **click chuột** vào sẽ thực hiện **một công việc** gì đó.
- ❑ Kế thừa từ các phương thức
  - javax.swing.AbstractButton
  - javax.swing.Jcomponent
  - java.awt.Container
  - java.awt.Component
  - java.lang.Object

## ❑ JButton():

- Tạo một Button không có text hoặc icon

## ❑ JButton(Action a) :

- Tạo một Button có thuộc tính được lấy từ một Action

## ❑ JButton(Icon icon) :

- Tạo một Button chỉ định một icon

## ❑ JButton(String text) :

- Tạo một Button chỉ định text

## ❑ JButton(String text, Icon icon) :

- Tạo một Button chỉ định text và icon

**public void setText(String s):**

- được sử dụng để thiết lập text đã cho trên button.

 **public String getText():**

- được sử dụng để trả về text của button

 **public void setEnabled(boolean b):**

- được sử dụng để kích hoạt hoặc vô hiệu hóa button.

 **public void setIcon(Icon b):**

- được sử dụng để thiết lập Icon đã cho trên button

 **public Icon getIcon():**

- được sử dụng để lấy Icon của button

 **public void setMnemonic(int a):**

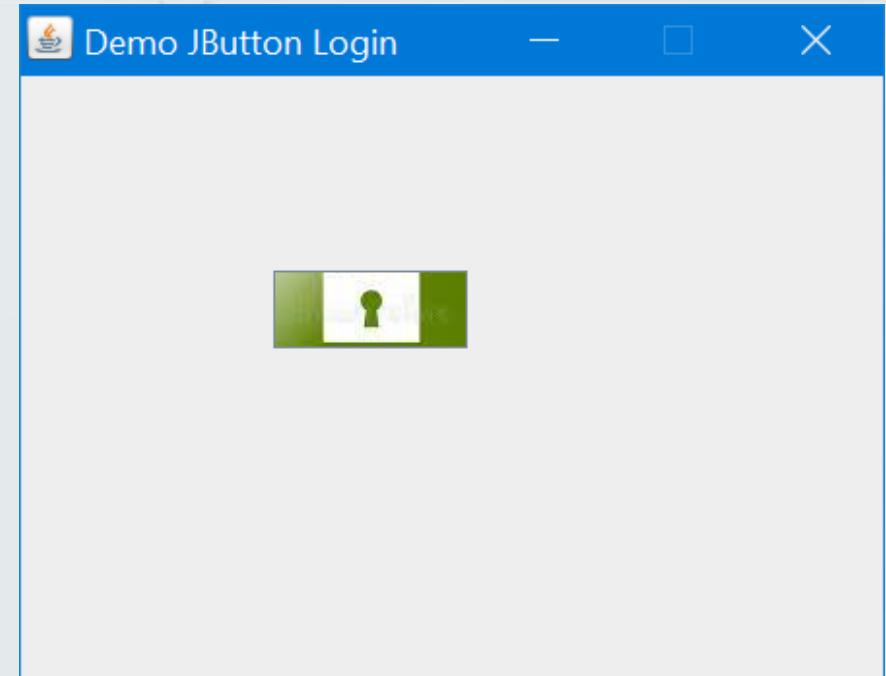
- được sử dụng để thiết lập thuộc tính mnemonic trên button.

 **public void addActionListener(ActionListener a):**

- được sử dụng để thêm action listener tới đối tượng này.

# JButton – Ví dụ

```
public class ViduJButton {  
    public ViduJButton() {  
        JFrame f = new JFrame();  
        f.setTitle("Demo JButton Login");  
        JButton b = new JButton();  
        b.setIcon(new ImageIcon("hinh/login.jpg"));  
        b.setBounds(130, 100, 100, 40);  
        f.add(b);  
        f.setSize(400, 350);  
        f.setLayout(null);  
        f.setVisible(true);  
        f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
    }  
    public static void main(String[] args) {  
        new ViduJButton();  
    }  
}
```



## □ JButton



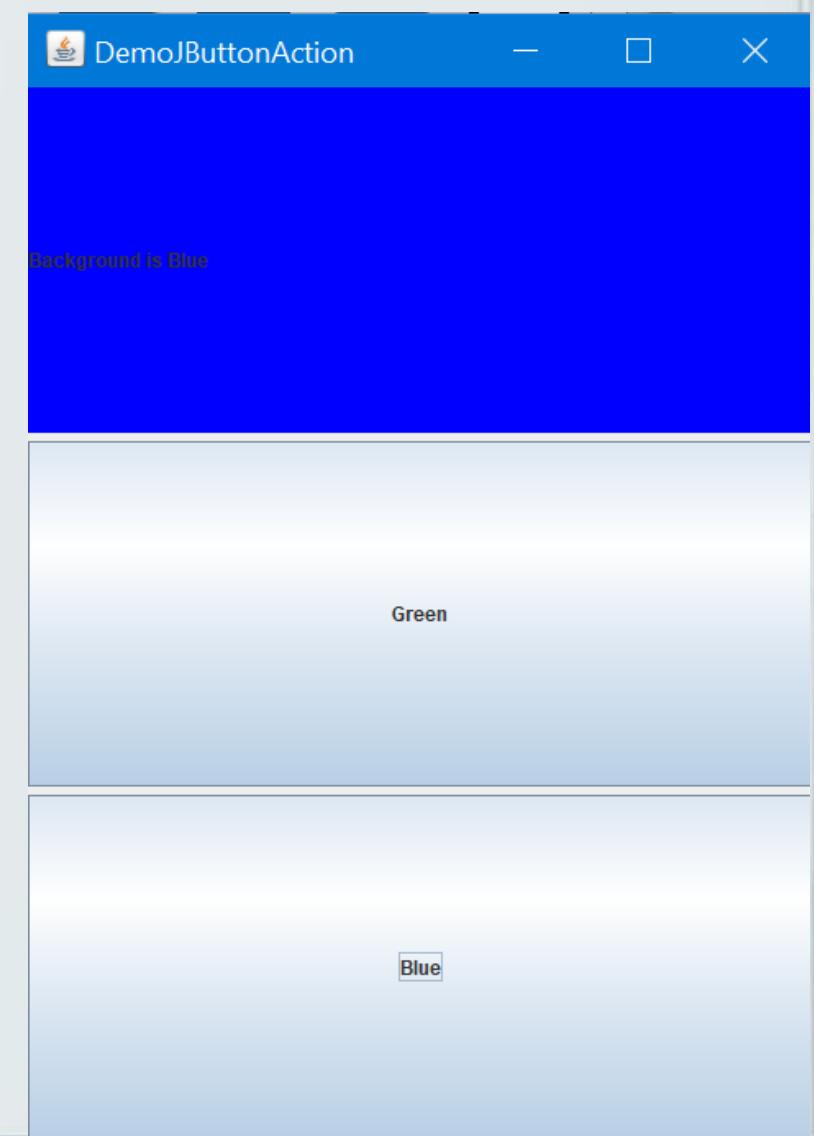
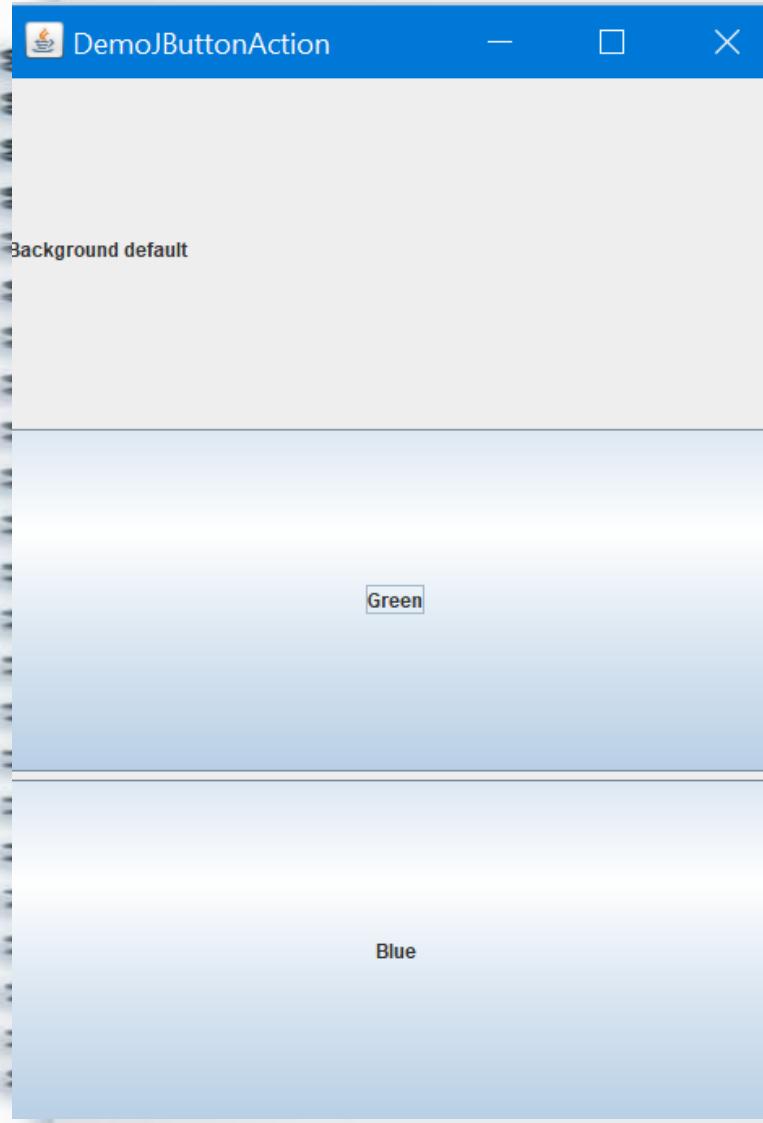
It is very important, attach event to do something that you want.

```
JButton btn=new JButton("Watch");
```

```
btn.setIcon(new ImageIcon("mywatch.png"));
```

```
btn.setMnemonic('W');  →Alt+W to call btn command
```

```
btn.addActionListener(new ActionListener() {  
    public void actionPerformed(ActionEvent arg0) {  
        //do something here →coding here  
    }  
});
```



```
public class DemoJButtonAction extends JFrame implements ActionListener {  
  
    private JButton btnGreen;  
    private JLabel lb;  
    public DemoJButtonAction() {  
        //create JFrame  
        setTitle("DemoJButtonAction");  
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
        setSize(500, 700);  
        setLayout(new GridLayout(3, 1, 5, 5));  
        //add Framecontent  
        lb = new JLabel("Background default");  
        lb.setOpaque(true); //mau nen mac dinh la false  
        add(lb);  
  
        btnGreen = createJButton("Green");  
        add(btnGreen);  
        add(createJButton("Blue"));  
    }
```

```
//display JFrame  
setLocationRelativeTo(null);  
setVisible(true);  
}  
//create JButton with text is title  
private JButton createJButton(String title) {  
    JButton btn = new JButton(title);  
    //add action for JButton  
    btn.addActionListener(this);  
    return btn;  
}  
//change text and background of  
JLabel when click button  
private void changeBackgroundColorJLabel  
(Color bgcolor, String namebg) {  
    lb.setBackground(bgcolor);  
    lb.setText("Background is "+namebg);  
}
```

```
//change text and background of JLabel when
click button

private void changeBackgroundJLabel(Color
bgcolor, String namebg) {
lb.setBackground(bgcolor);
lb.setText("Background is "+namebg);
}

@Override
public void actionPerformed(ActionEvent e) {
// click button green
if(e.getSource() == btnGreen) {
changeBackgroundJLabel(Color.green, "Green");
}
//click button blue
if(e.getActionCommand() == "Blue") {
changeBackgroundJLabel(Color.blue, "Blue");
}
}
```

```
public static void main(String[] args) {
// TODO Auto-generated method stub
new DemoJButtonAction();
}
```

- Sử dụng lớp `javax.swing.JComboBox`

❑ Là một thành phần có sự kết hợp giữa một button, một trường có thể chỉnh sửa và một danh sách xổ xuống. Tại một thời điểm chỉ có một phần tử có thể được lựa chọn từ danh sách.

❑ **Hàm tạo thông dụng**

- **JComboBox():** tạo một JComboBox với danh sách phần tử rỗng.
- **JComboBox(Object[] items):** tạo một JComboBox chứa các phần tử trong mảng tham số.
- **JComboBox(Vector items):** tạo một JComboBox chứa các phần tử trong Vector tham số.

## □ Phương thức thông dụng

- **void addItem(Object anObject)**: được sử dụng để thêm một phần tử mới vào danh sách phần tử của JComboBox.
- **void removeItem(Object anObject)**: được sử dụng để xóa một phần tử ra khỏi danh sách phần tử của JComboBox.
- **void removeAllItems()**: được sử dụng để xóa tất cả phần tử của danh sách của JComboBox.
- **void setEditable(boolean b)**: được sử dụng để xác định xem có hay không JComboBox là được nhập vào phần tử không có danh sách.
- **Object getSelectedItem()**: trả về phần tử được chọn.

# JCombobox

```
JComboBox cbo=new JComboBox();
cbo.addItem("Xuất sắc");
cbo.addItem("Giỏi");
cbo.addItem("Khá");
cbo.addItem("Trung bình");
add(cbo);
```



---

```
String arr[]={ "Xuất sắc" , "Giỏi"
,"Khá" , "Trung bình"};
JComboBox cbo=new JComboBox(arr);
add(cbo);
```

```
class Person
{
    private String Id;
    private String Name;
    public Person(String id, String name){
        Id=id;
        Name=name;
    }
    public String toString() {
        return Name;
    }
}
```

How to add Array Person to JComboBox?



```
Person []list={  
    new Person("1", "Trần Thành Công"),  
    new Person("2", "Nguyễn Đại Thắng"),  
    new Person("3", "Hoàng Thành Đạt")};  
JComboBox cbo2=new JComboBox(list);  
add(cbo2);
```

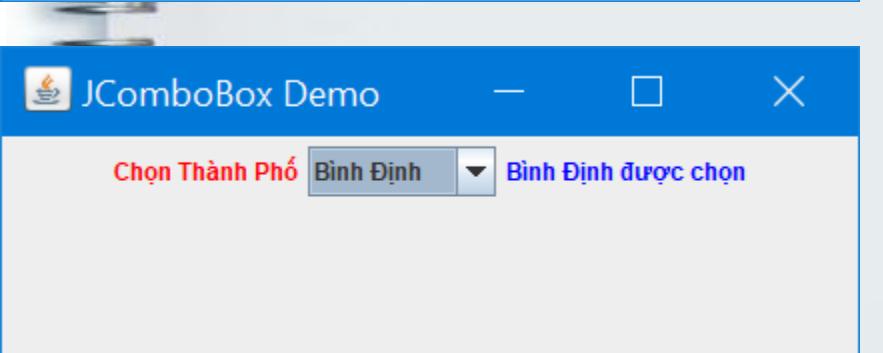
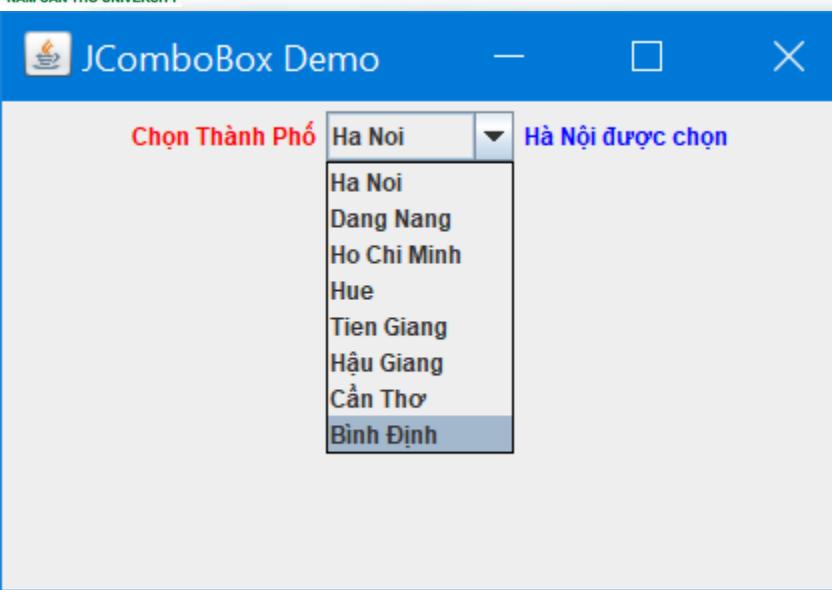
How to add Array Person to JComboBox?

We get Person from selected:

Person

```
p=(Person)cbo2.getSelectedItem();
```

# JCombobox – Ví dụ



```
public class ViduJComboBox extends JFrame{  
  
    public ViduJComboBox() {  
        //create a new Panel  
        //Create checkbox  
    }  
    public void showWindow(String tieude){  
        //sv code  
    }  
    public static void main(String[] args) {  
        new ViduJComboBox()  
            .showWindow("JComboBox Demo");  
    }  
}
```

```

public class ViduJComboBox extends JFrame{
public ViduJComboBox() {
//create a new Panel
JPanel pn = new JPanel(new FlowLayout());
String ds[] = {"Ha Noi", "Dang Nang", "Ho Chi
Minh", "Hue", "Tien Giang", "Hậu Giang",
"Cần Thơ", "Bình Định"};
JLabel lbChonTP = new JLabel("Chọn Thành Phố");
JLabel lbTPChon = new JLabel("Hà Nội được
chọn");
lbChonTP.setForeground(Color.red);
lbTPChon.setForeground(Color.BLUE);
//Create checkbox
JComboBox jcb = new JComboBox(ds);
jcb.addItemListener(new ItemListener() {
@Override
public void itemStateChanged(ItemEvent e) {
if(e.getSource() == jcb) {
lbTPChon.setText(jcb.getSelectedItem()+" được
chọn");
}
}
});

```

```

pn.add(lbChonTP);
pn.add(jcb);
pn.add(lbTPChon);
this.add(pn);
}

public void showWindow(String tieude) {
this.setTitle(tieude);
this.setSize(450, 300);
this.setDefaultCloseOperation(EXIT_ON_CLOSE);
this.setLocationRelativeTo(null);
this.setVisible(true);
}

public static void main(String[] args) {
new ViduJComboBox().showWindow("JComboBox
Demo");
}
}

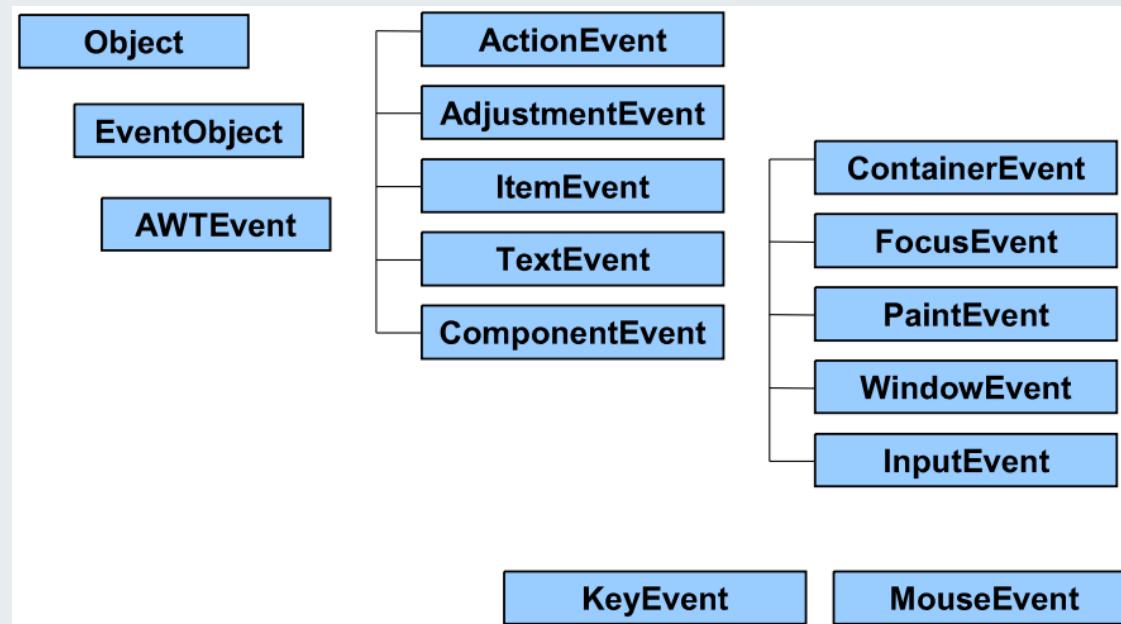
```



## □ GUI là hệ thống hướng sự kiện (event-driven)

- Chuột nhấn và chuyển động, nút nhấn và văn bản nhập thông qua bàn phím, nhấn vào các mục menu,...
- Thao tác mong muốn sinh ra một hành động trên mỗi các sự kiện này

## □ Gói `java.awt.event.*; java.swing.event.*;`



## ❑ Xử lý sự kiện sẽ liên quan đến 3 đối tượng

- Đối tượng nguồn (source), đối tượng phát sinh sự kiện: Button, TextField, ...
- Sự kiện (event): sự kiện phát sinh khi 1 đối tượng nguồn bị tác động.
- Ví dụ
  - *1 Button được bấm,*
  - *1 ComboBox thay đổi giá trị được chọn,*
  - *1 cửa sổ được đóng, ...*

## ❑ Bộ lắng nghe sự kiện (listener):

- Khi sự kiện được tạo ra, nó sẽ gửi thông báo đến các bộ nghe (đã được đăng ký). Phương thức xử lý sự kiện tương ứng sẽ được kích hoạt.

# Các loại bộ sự kiện và bộ lắng nghe tương ứng

Event Classes	Listener Interfaces
ActionEvent	ActionListener
MouseEvent	MouseListener and MouseMotionListener
MouseWheelEvent	MouseWheelListener
KeyEvent	KeyListener
ItemEvent	ItemListener
TextEvent	TextListener
AdjustmentEvent	AdjustmentListener
WindowEvent	WindowListener
ComponentEvent	ComponentListener
ContainerEvent	ContainerListener
FocusEvent	FocusListener

## □ Button, MenuItem

- **void addActionListener(ActionListener a):** lắng nghe sự kiện click trên nút lệnh, mục menu.

## □ TextField, TextArea

- **void addTextListener(TextListener a):** lắng nghe sự kiện nội dung text bị thay đổi.
- **void addKeyListener(KeyListener a):** lắng nghe sự kiện mỗi khi một phím trên bàn phím được nhấn, nhả.

## □ Checkbox, RadioButton, ComboBox, ListBox

- **void addItemListener(ItemListener a):** lắng nghe sự kiện một thành phần được nhấn.

## □ ...

## 1. Tạo lớp lắng nghe sự kiện có cài đặt (implements) bộ lắng nghe sự kiện tương ứng

- class ButtonClickListener implements ActionListener.
- *Tuy nhiên có thể sử dụng các bộ lắng nghe nguyên thủy do Java cài đặt.*

## 2. Đăng ký bộ lắng nghe cho nguồn (source)

- okButton.addActionListener(new ButtonClickListener());
- okButton.addActionListener(this);

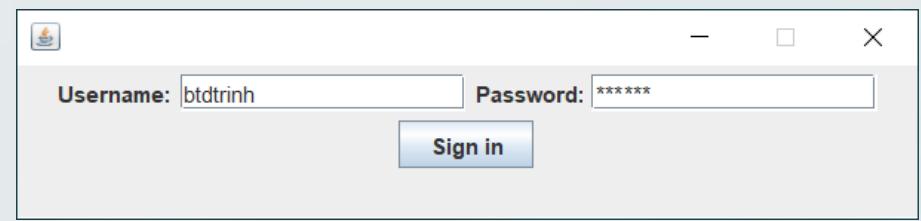
## 3. Tái định nghĩa các hàm xử lý sự kiện

```
public void actionPerformed(ActionEvent e) {  
    String command = e.getActionCommand();  
    if( command.equals( "OK" ) ) {  
        // ...  
    }  
}
```

❑ Sử dụng Frame được bố cục theo FlowLayout với 1 trường TextField nhập vào User name và 1 trường PasswordField để nhập Password cùng nút lệnh có nhãn Sign in. Khi người dùng nhấn nút lệnh thì sử dụng Dialog hiển thị thông tin đã được nhập vào.

❑ Hướng dẫn

- Tạo một frame
- Thiết lập kiểu bố cục FlowLayout, canh giữa (ví dụ)
- Tạo các label, textfield, passwordfield, button
- Đăng ký lắng nghe sự kiện click cho nút lệnh
- Thêm vào frame
- Tái định nghĩa hàm xử lý sự kiện



# Ví dụ 1 - Xử lý sự kiện

```
public class ViDuXuLySuKien_1 implements
ActionListener{
JFrame frame;
JTextField userText;
JPasswordField passText;
public ViDuXuLySuKien_1() {
    frame = new JFrame();
    frame.setSize(500,120);
    frame.setTitle("ViDuXuLySuKien_1");
    FlowLayout layout = new
    FlowLayout(FlowLayout.CENTER);
    frame.setLayout(layout);
    JLabel userLabel = new
    JLabel("Username:");
    JLabel passLabel = new
    JLabel("Password:");
    userText = new JTextField(15);
    passText = new JPasswordField(15);
    passText.setEchoChar('*');//đặt ký
    tự hiển thị cho ô nhập mật khẩu
}
```

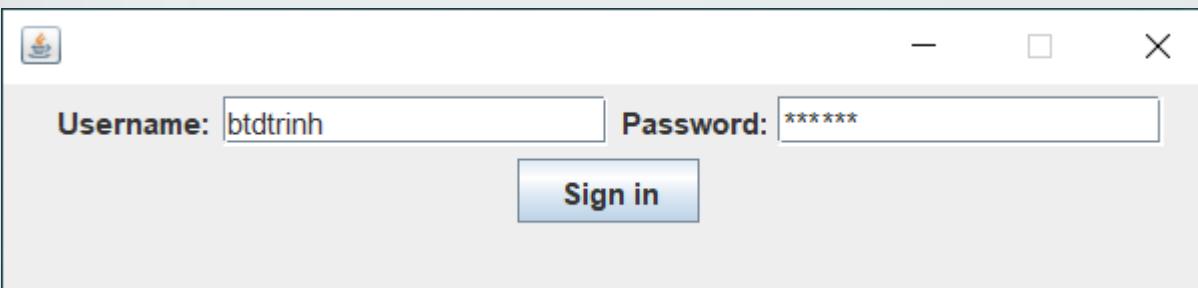
```
JButton button = new JButton("Sign in");
button.addActionListener(this);//đăng ký
lắng nghe sự kiện click
frame.add(userLabel);
frame.add(userText);
frame.add(passLabel);
frame.add(passText);
frame.add(button);
frame.setLocation(200, 150);

frame.setDefaultCloseOperation(javax.swing
.WindowConstants.EXIT_ON_CLOSE);
frame.setResizable(false);
frame.setVisible(true);
}

public static void main(String[] args) {
// TODO Auto-generated method stub
ViDuXuLySuKien_1 frame = new
ViDuXuLySuKien_1();
}
```

# Ví dụ 1 - Xử lý sự kiện

```
@Override
public void actionPerformed(ActionEvent e) {
    // TODO Auto-generated method stub
    if(e.getActionCommand().equals("Sign in")) {
        String message = "Bạn nhập vào username là
                        "+userText.getText()+
                        " và password là
                        "+String.valueOf(passText.getPassword());
        JOptionPane.showMessageDialog(frame, message);
    }
}
```

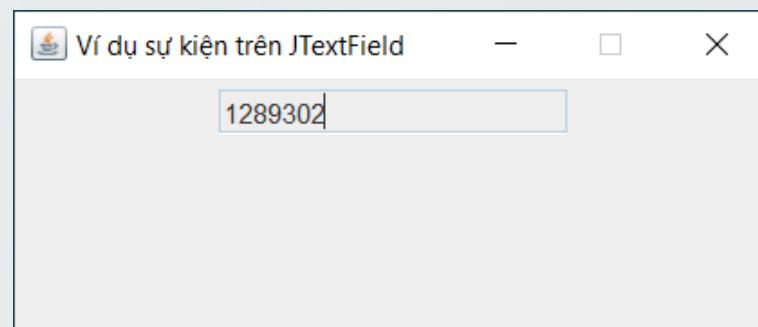


- Sử dụng sự kiện phù hợp để **không cho phép** người dùng nhập vào **textfield ký tự khác với ký tự số.**
- Hướng dẫn
  - Tạo một frame
  - Thiết lập kiểu bố cục FlowLayout
  - Tạo textfield
  - Đăng ký lắng nghe sự kiện bàn phím cho textfield
  - Thêm textfield vào frame
  - Tái định nghĩa hàm xử lý sự kiện keyPressed

## Ví dụ 2 - Xử lý sự kiện

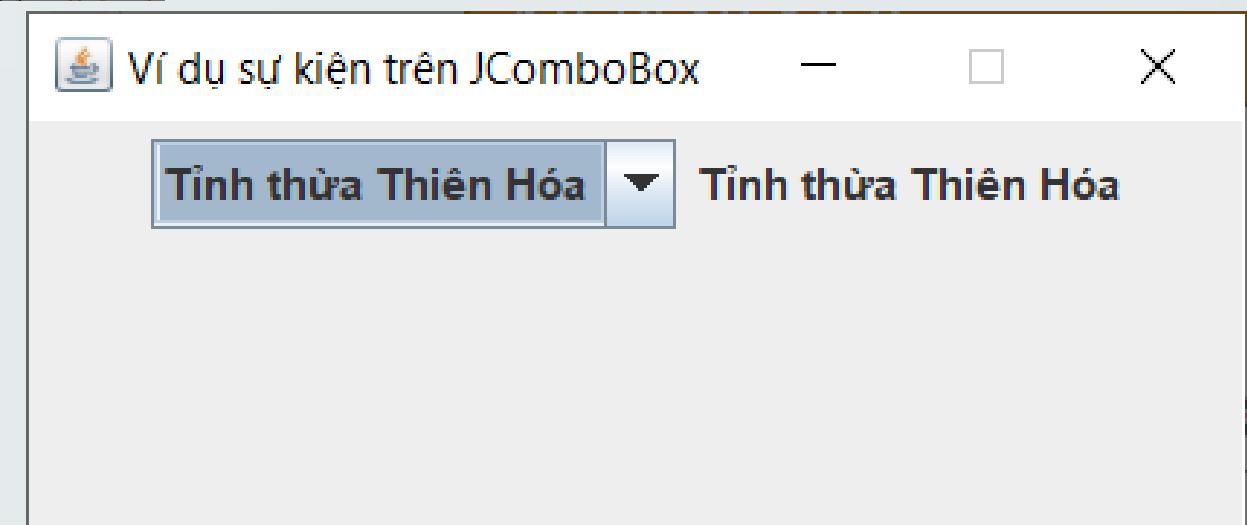
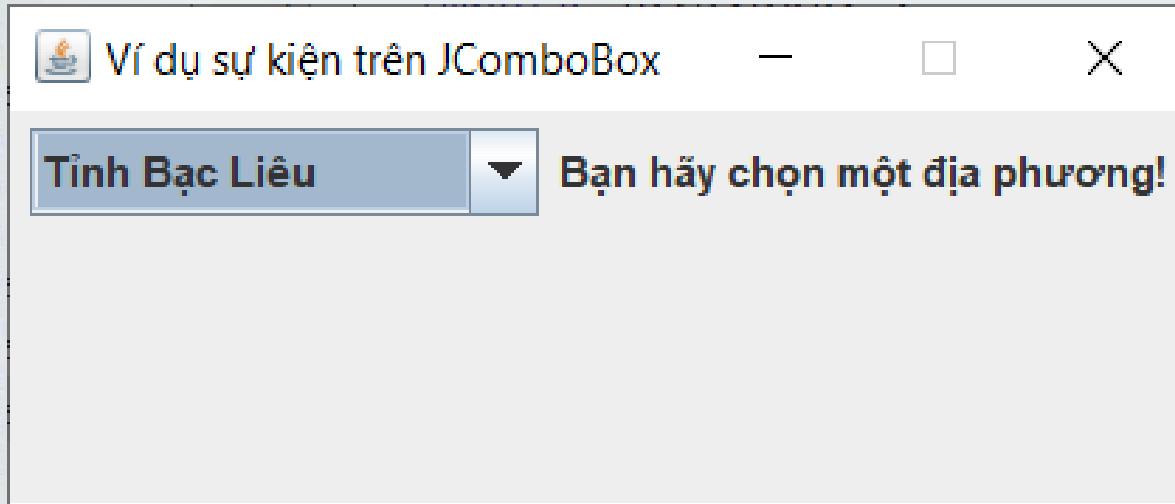
```
JFrame frame;
JTextField text;
public void viDuXuLySuKien () {
    frame = new JFrame();
    frame.setTitle("Ví dụ sự kiện trên JTextField");
    frame.setLayout(new FlowLayout());
    frame.setSize(350, 150);
    frame.setLocation(200, 200);
    frame.setResizable(false);
    frame.setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
    text = new JTextField(15);
    text.addKeyListener(this); //đăng ký sự kiện lắng nghe bàn phím
    frame.add(text);
    frame.setVisible(true);
}
```

```
//tái định nghĩa các phương thức xử lý sự kiện
//phải tái định nghĩa cả 3 phương thức
public void keyPressed(KeyEvent e){
    if(e.getKeyChar() >= '0' && e.getKeyChar() <= '9'){
        text.setEditable(true);
    }
    else{
        text.setEditable(false);
    }
}
public void keyReleased(KeyEvent e) {
}
public void keyTyped(KeyEvent e) {
}
```



- ❑ Thiết kế một Combobox bao gồm các phần tử được định nghĩa trước.
- ❑ Hãy sử dụng sự kiện phù hợp để hiển thị phần tử hiện được chọn của ComboBox bằng một Label mỗi khi người dùng thay đổi phần tử hiện hành của ComboBox.
- ❑ Hướng dẫn
  - Tạo một frame
  - Thiết lập kiểu bố cục FlowLayout
  - Tạo combobox, label
  - Đăng ký lắng nghe sự kiện phần tử được chọn cho combobox
  - Thêm combobox, label vào frame
  - Tái định nghĩa hàm xử lý sự kiện itemStateChanged để đặt lại nahnx của label

## Ví dụ 3 – Xử lý sự kiện



# Ví dụ xử lý sự kiện

```
public class viDuXuLySuKien_3 extends JFrame implements ItemListener {  
    JFrame frame;  
    JComboBox combo;  
    JLabel label;  
    public viDuXuLySuKien_3() {  
        frame = new JFrame();  
        frame.setTitle("Ví dụ sự kiện trên JComboBox");  
        frame.setLayout(new FlowLayout());  
        frame.setSize(350, 150);  
        frame.setLocation(200, 200);  
        frame.setResizable(false);  
  
        frame.setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);  
        String [] list = {"Tỉnh Bạc Liêu",  
                        "Tỉnh Cà Mau", "TP Cần Thơ", "Tỉnh thừa  
Thiên Hóa"};
```

```
combo = new JComboBox(list);  
combo.addItemListener(this);  
frame.add(combo);  
label = new JLabel("Bạn hãy chọn một  
địa phương!");  
frame.add(label);  
frame.setVisible(true);  
}  
public static void main(String[] args) {  
    new viDuXuLySuKien_3();  
}  
  
@Override  
public void itemStateChanged(ItemEvent e) {  
    Object selectedItem =  
        ((JComboBox)(e.getSource())).getSelectedItem();  
    label.setText(selectedItem.toString());  
}  
}
```

☐ Sinh viên thực hiện kế theo mẫu như yêu cầu, JTextField nhập tên của Sinh viên

btdTrinh XuLyJTextField

Tiền gửi:	<input type="text"/>
Lãi suất/Tháng:	<input type="text"/>
Tháng:	<input type="text"/>
Tiền Lãi:	<input type="text"/>
<input type="button" value="Tính"/> <input type="button" value="Nhập Lại"/>	

btdTrinh XuLyJTextField

Tiền gửi:	50000
Lãi suất/Tháng:	0.03
Tháng:	7
Tiền Lãi:	10500.0
<input type="button" value="Tính"/> <input type="button" value="Nhập Lại"/>	

## Bài tập 2

 Giải PT Bậc 1-btdtrinh - □ ×

### Giải phương trình bậc 1

Hệ số a:

Hệ số b:

Giải  Thoát Help

Kết quả :

```

7 public class PhuongTrinhBac1UI extends JFrame implements ActionListener{
8
9     JTextField txtHeSoa,txtHeSob;
10    JButton btnGiai,btnThoat,btnHelp;
11    JTextField txtKetqua;
12
13    ActionListener eventGiai=new ActionListener() {
14
15        public void actionPerformed(ActionEvent e) {
16            // TODO Auto-generated method stub
17            xuLyGiaiPhuongTrinh();
18        }
19    };
20
21    public PhuongTrinhBac1UI(String title){}
22
23    protected void xuLyGiaiPhuongTrinh() {
24        String hsa=txtHeSoa.getText();
25        String hsb=txtHeSob.getText();
26        double a=Double.parseDouble(hsa);
27        double b=Double.parseDouble(hsb);
28        if(a==0 && b==0)
29        {
30            //gán giá trị lên JTextField
31            txtKetqua.setText("Vô số nghiệm");
32        }
33        else if(a==0 &&b!=0)
34        {
35            txtKetqua.setText("Vô nghiệm");
36        }
37        else
38        {
39            double x=-b/a;
40            txtKetqua.setText("x="+x);
41        }
42    }
43
44
45
46
47    public void addEvents()
48    {
49        btnThoat.addActionListener(new ActionListener() {
50
51            public void actionPerformed(ActionEvent e) {
52                // TODO Auto-generated method stub
53                System.exit(0);
54            }
55        });
56        btnGiai.addActionListener(eventGiai);
57        btnHelp.addActionListener(new HelpEvent());
58    }

```

```
public void addControls()
{
    Container con=getContentPane();
    JPanel pnMain=new JPanel();
    pnMain.setLayout(new BoxLayout(pnMain,
BoxLayout.Y_AXIS));
    con.add(pnMain);

    JPanel pnTitle=new JPanel();
    pnTitle.setLayout(new FlowLayout());
    JLabel lblTieuDe=new JLabel("Giải phương trình bậc 1");
    pnTitle.add(lblTieuDe);
    pnMain.add(pnTitle);
    lblTieuDe.setForeground(Color.BLUE);
    Font fontTieuDe=new Font("arial", Font.BOLD, 20);
    lblTieuDe.setFont(fontTieuDe);
}
```

```
JPanel pnHeSoa=new JPanel();
pnHeSoa.setLayout(new FlowLayout());
JLabel lblHeSoa=new JLabel("Hệ số a:");
txtHeSoa=new JTextField(15);
pnHeSoa.add(lblHeSoa);
pnHeSoa.add(txtHeSoa);
pnMain.add(pnHeSoa);
```

```
JPanel pnHeSob=new JPanel();
pnHeSob.setLayout(new FlowLayout());
JLabel lblHeSob=new JLabel("Hệ số b:");
txtHeSob=new JTextField(15);
pnHeSob.add(lblHeSob);
pnHeSob.add(txtHeSob);
pnMain.add(pnHeSob);

JPanel pnButton=new JPanel();
pnButton.setLayout(new FlowLayout());
btnGiai=new JButton("Giải");
btnThoat=new JButton("Thoát");
btnThoat.setIcon(new ImageIcon("hinh/exit.png"));
btnHelp=new JButton("Help");

pnButton.add(btnGiai);
pnButton.add(btnThoat);
pnButton.add(btnHelp);
pnMain.add(pnButton);
```

```
JPanel pnKetQua=new JPanel();
pnKetQua.setLayout(new FlowLayout());
JLabel lblKetQua=new JLabel("Kết quả :");
txtKetqua=new JTextField(15);
pnKetQua.add(lblKetQua);
pnKetQua.add(txtKetqua);
pnMain.add(pnKetQua);
}

class HelpEvent implements ActionListener
{
    public void actionPerformed(ActionEvent e) {
        // TODO Auto-generated method stub
        JOptionPane.showMessageDialog(null,
            "Chi tiết xin liên hệ");
    }
}
```

```
public void showWindow()
{
    this.setSize(400, 250);
    this.setDefaultCloseOperation(EXIT_ON_CLOSE);
    this.setLocationRelativeTo(null);
    this.setVisible(true);
}

@Override
public void actionPerformed(ActionEvent e) {
    // TODO Auto-generated method stub
}
```

- Border
- Border Title
- JTextArea
- JScrollPane
- JCheckBox
- JRadioButton-ButtonGroup

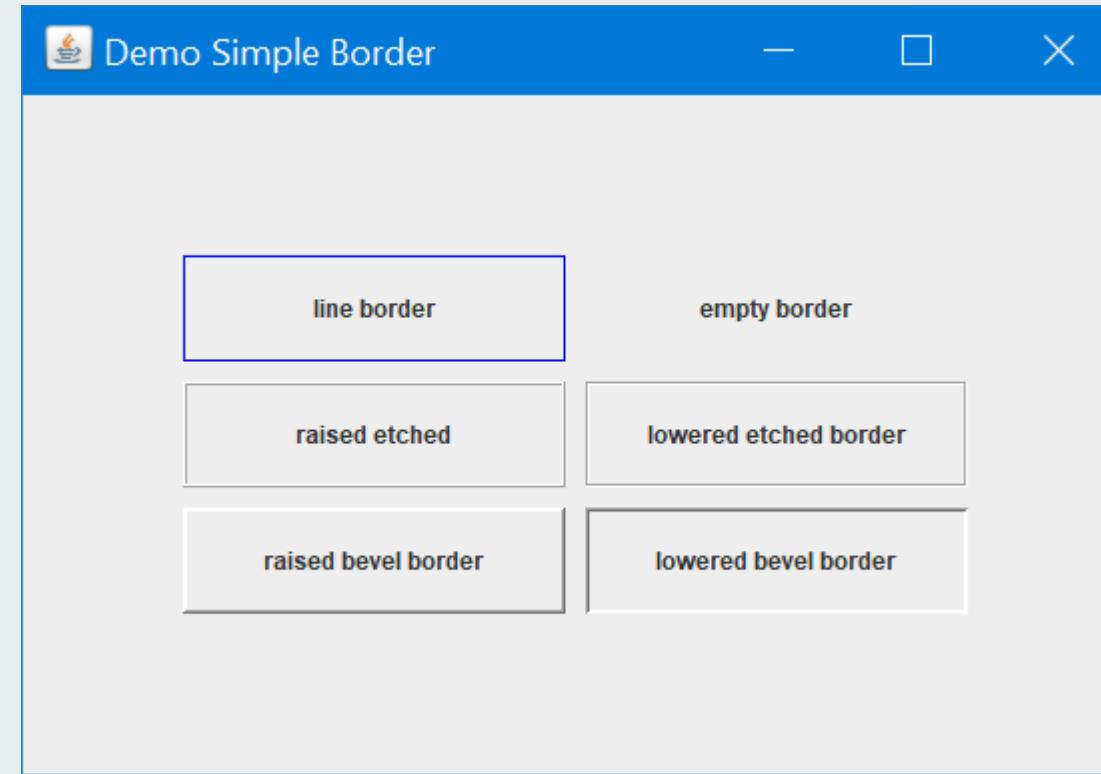
## ❑ Cú pháp tạo các loại border

Border border = BorderFactory.create...

## ❑ Một số loại border đơn giản

- **Line border** là loại border nét liền với màu sắc nào đó
- **Empty border** là border trống, nó thường được dùng để tạo khoảng trống, công dụng như sử dụng padding để đặt khoảng cách giữa nội dung và bên ngoài.
- **Raised etched border** là dạng border nét nổi, khắc nổi lên trên.
- **Lowered etched border** là border khắc nhưng chìm xuống dưới.
- **Raised bevel border** là border dạng tạo góc xiên nổi lên
- **Lowered bevel border** là border dạng tạo góc chìm xuống.

# Ví dụ Border



# Border, Border Title

Quản lý sinh viên- Oracle

### Chương trình quản lý sinh viên

Mã sinh viên:

Tên sinh viên:

Tuổi:

**Lưu** **Mới** **Xóa** **Thoát**

Danh sách

Mã	Tên	Tuổi
sv1	Nguyễn Thị An	19
sv2	Hồ Ngọc TÀO LAO	18
sv3	Trần Thị Hạnh	20
sv4	Nguyễn Văn Phúc	22
sv5	Phạm Văn Giải	19
sv6	Hồ Cố Thoát	21
sv7	Trần Phạm Mẫn Nhi	15

# Border, Border Title

```
JPanel pnTable=new JPanel();
Border border=
    BorderFactory.createLineBorder(Color.RED);
TitledBorder borderTitle=
    BorderFactory.createTitledBorder(
        border, "Danh sách");
pnTable.setBorder(borderTitle);
```

- Sử dụng lớp javax.swing.JPasswordField.**
- Thừa kế từ JTextField.**
- Cho phép nhập dòng text đơn nhưng các ký tự hiển thị là ký tự đại diện.**
- Hàm tạo thông dụng**
  - **JPasswordField():** tạo một JPasswordField rỗng.
  - **JPasswordField(int columns):** tạo một JPasswordField rỗng với số cột đã cho.
  - **JPasswordField(String text):** tạo một JPasswordField với text đã cho.
  - **JPasswordField(String text, int columns):** tạo một JPasswordField với text và các cột đã cho.
- Phương thức cần dùng**
  - **void setEchoChar(char c):** thiết lập ký tự hiển thị của JPasswordField.

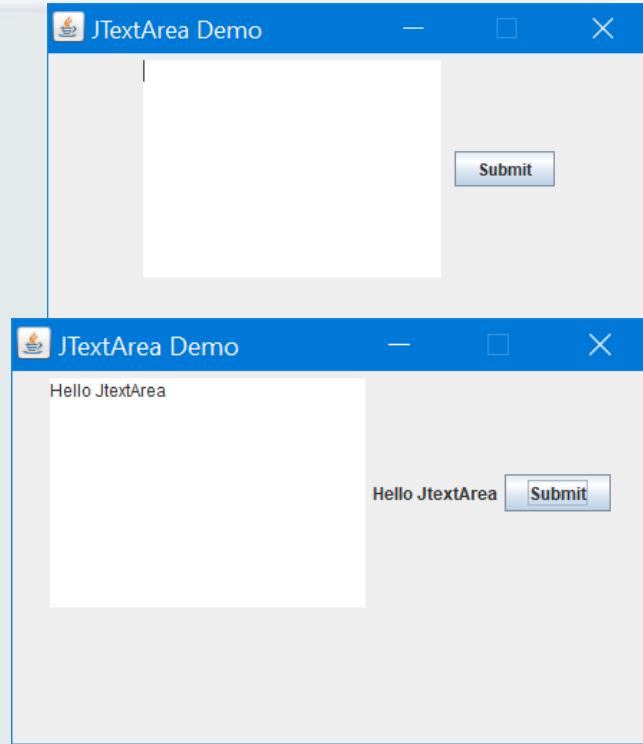
- ❑ Cho phép **hiển thị** và **chỉnh sửa** văn bản (bao gồm nhiều dòng)
- ❑ Thường được sử dụng để **nhập**, **chỉnh sửa** hoặc **hiển thị** các văn bản chứa **nội dung dài**: mô tả, thông tin liên hệ,...
- ❑ Một số hàm tạo JTextArea
  - **JTextArea()** : Tạo JTextArea rỗng
  - **JTextArea(String s)** : Tạo JTextArea với nội dung ban đầu là chuỗi s được truyền vào..
  - **JTextArea(int row, int column)** : Khởi tạo JTextArea không chứa văn bản, có số dòng row và số cột column như giá trị truyền vào.
  - **JTextArea(String s, int row, int column)** : Khởi tạo JTextArea với văn bản, số dòng row và cột column được truyền vào.

## ❑ Một số phương thức thường sử dụng

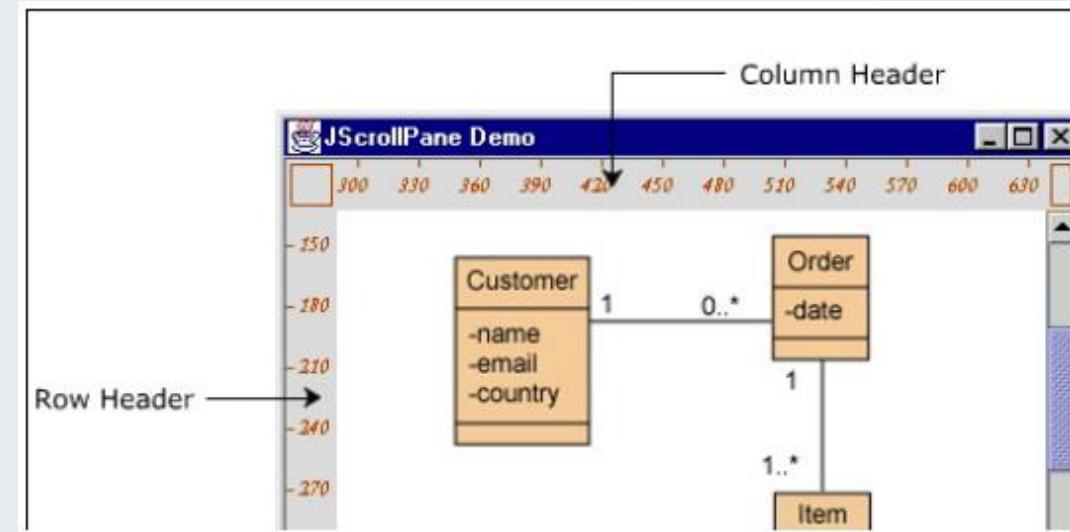
- **append(String s)** : Gắn một s vào văn bản hiện tại của JTextArea.
- **getLineCount()** : Lấy số dòng của văn bản trong JTextArea ( số dòng mà văn bản đang chiếm giữ, có thể nhỏ hơn tổng số dòng có thể có của JTextArea).
- **setFont(Font f)** : Thay đổi font chữ.
- **setColumns(int c)** : Thay đổi số cột
- **setRows(int r)** : Thay đổi số dòng.
- **getColumns()** : Lấy tổng số cột của JTextArea
- **getRows()** : Lấy tổng số dòng của JTextArea

# Ví dụ tạo JTextArea đơn giản

```
public class ViduJTextArea extends JFrame{  
    public ViduJTextArea() {}  
    public void showWindow(String tieude) {}  
    public static void main(String[] args) {  
        new ViduJTextArea().showWindow("JTextArea Demo");  
    }  
    public ViduJTextArea() {  
        JPanel pn = new JPanel();  
        JLabel label = new JLabel();  
        JButton button = new JButton("Submit");  
        JTextArea jta = new JTextArea(10,20);  
        pn.add(jta);  
        pn.add(label);  
        pn.add(button);  
        add(pn);  
        //Bắt sự kiện onclick  
    }  
    // Bat su kien button click  
    button.addActionListener(new ActionListener() {  
        @Override  
        public void actionPerformed(ActionEvent e) {  
            label.setText(jta.getText());  
        }  
    });
```



- ❑ Cung cấp một view cho phép cuộn một thành phần
- ❑ Một ScrollPane cung cấp cả hai thanh cuộn ngang và dọc



### ❑ JScrollPane():

- Khởi tạo JScrollPane rỗng, thanh cuộn có thể xuất hiện theo cả chiều ngang và chiều rộng khi cần thiết.

### ❑ JScrollPane(Component component)

- Khởi tạo JScrollPane cho một component được chỉ định, khi nội dung của component này lớn hơn kích thước được phép, thì thanh cuộn ngang và dọc sẽ xuất hiện.

### ❑ JScrollPane(int vsPolicy, int hsPolicy)

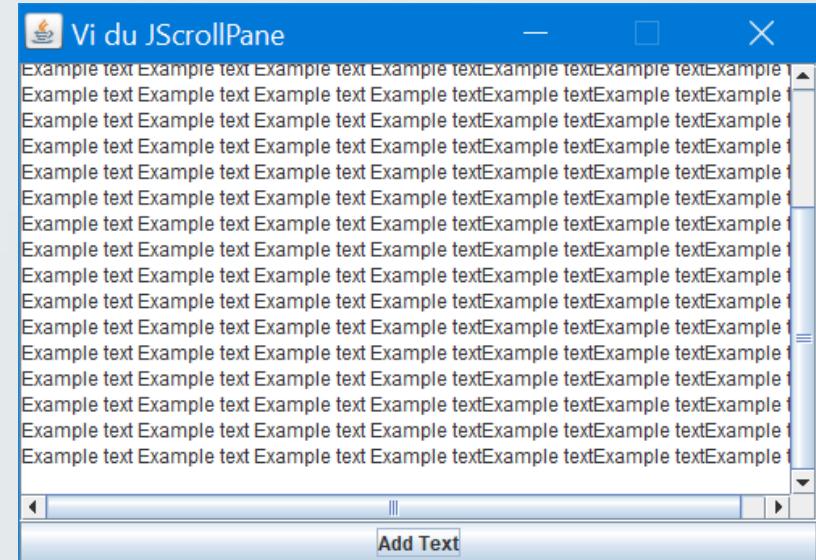
- Khởi tạo JScrollPane với vsPolicy và hsPolicy được chỉ định.

### ❑ ScrollPane(Component c, int vsPolicy, int hsPolicy)

- Khởi tạo JScrollPane cho một component và vsPolicy, hsPolicy được chỉ định.

# JScrollPane – Ví dụ

```
public ViduJScrollPane() {  
    JScrollPane jscpane = new JScrollPane();  
    JTextArea txtMain = new JTextArea();  
    jscpane.setViewportView(txtMain);  
    add(jscpane, BorderLayout.CENTER);  
  
    JButton btnAddText = new JButton("Add Text");  
    btnAddText  
        .addActionListener(e -> {  
            txtMain.setText(txtMain.getText()  
                + "Example text Example  
text Example text Example textExample  
textExample textExample text\n");  
            String text = txtMain.getText();  
            txtMain.setCaretPosition(text !=  
null ? text.length() : 0);  
        });  
    add(btnAddText, BorderLayout.SOUTH);  
}
```

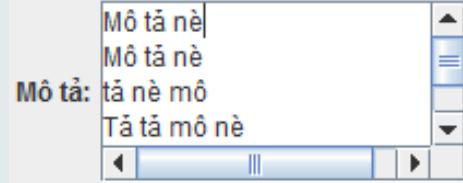


```
public static void main(String[] args) {  
    new ViduJScrollPane().showWindow("Vi du  
JScrollPane");  
}
```

- Sử dụng lớp **javax.swing.JTextArea**.
- Có tính năng tương tự như JTextField nhưng có thể bao gồm nhiều dòng.
- **Hàm tạo thông dụng**
  - **JTextArea()**: tạo một TextArea rỗng.
  - **JTextArea(String s)**: tạo một TextArea với text đã cho.
  - **JTextArea(int row, int column)**: tạo một TextArea rỗng với số hàng và cột đã cho.
  - **JTextArea(String s, int row, int column)**: tạo một TextArea với text, số hàng và cột đã cho.

- ❑ Thêm JTextArea vào vật chứa giống như cách thêm JTextField.
- ❑ Phương thức thông dụng
  - **void textArea.setEditable (Boolean b):** cho phép hoặc không người dùng sửa nội dung.
  - **JScrollPane scrollPane = new JScrollPane (textArea):** bổ sung thanh cuộn.
  - **void textArea.append (String s):** cộng thêm chuỗi vào nội dung của JTextArea.
  - **void textArea.setLineWrap (Boolean b):** cho phép nội dung của JTextArea tự động xuống hàng hay không.

# JTextArea, JScrollPane



Input data multi line

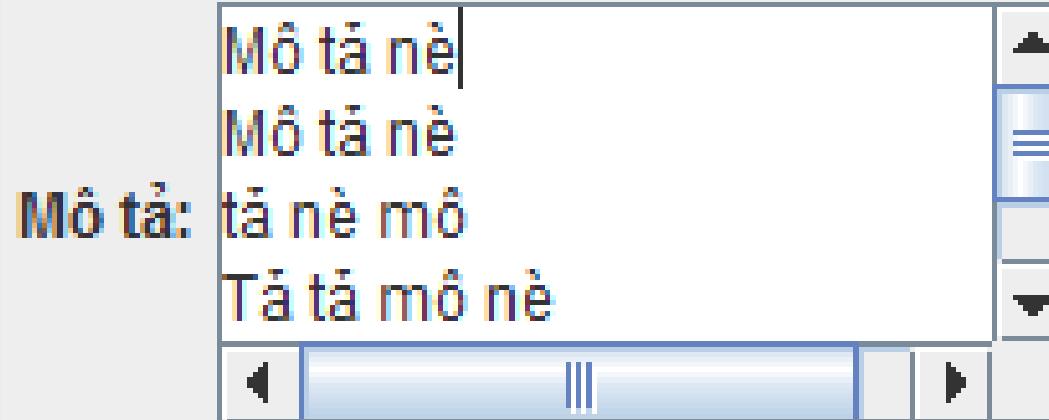
```
JLabel lblDes=new JLabel("Mô tả:");
JTextArea are=new JTextArea(5, 15);
JScrollPane sc=new JScrollPane(are);
add(lblDes); add(sc);
```

---

```
JTextArea are=new JTextArea(5, 15);
```

→ 5 rows, 15 columns

We should use JScrollPane to create Scroll  
for JTextArea when user input data over  
limit row or column



```
JScrollPane sc=new JScrollPane(are,  
JScrollPane.VERTICAL_SCROLLBAR_ALWAYS,  
JScrollPane.HORIZONTAL_SCROLLBAR_ALWAYS);
```

# JTextArea, JScrollPane

```
ImageIcon img=new ImageIcon("baby.jpg");
JLabel lblImg=new JLabel(img);
JScrollPane scimg=new JScrollPane(lblImg,
JScrollPane.VERTICAL_SCROLLBAR_ALWAYS,
JScrollPane.HORIZONTAL_SCROLLBAR_ALWAYS);
scimg.setPreferredSize(new Dimension(600, 500));
add(scimg);
```



JCheckBox kế thừa từ AbstractButton và JTogglleButton Một số hàm tạo thường dùng

- **JCheckBox()** – Tạo JCheckBox mặc định không chứa bất kỳ văn bản và icon nào
- **JCheckBox(Icon i)** – Tạo JCheckBox với icon được chỉ định.
- **JCheckBox(Icon i, boolean selected)** – Tạo CheckBox với Icon được chỉ định và giá trị boolean – selected là giá trị mặc định ban đầu xem checkbox này có được chọn hay không.
- **JCheckBox(String text, boolean selected)** – Tạo CheckBox với văn bản được chỉ định và giá trị boolean – selected là giá trị mặc định ban đầu xem checkbox này có được chọn hay không.
- **JCheckBox(String text, Icon)** – Tạo JCheckBox với Icon và văn bản được chỉ định
- **JCheckBox(String text, Icon, boolean selected)** – Tạo JCheckBox với Icon và văn bản được chỉ định và giá trị boolean – selected là giá trị mặc định ban đầu xem checkbox này có được chọn hay không.

## Một số hàm xử lý sự kiện

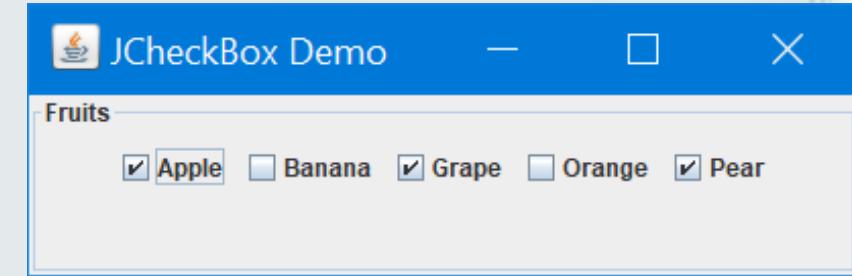
- **addActionListener(ItemListener l):** Thêm một ItemListener
- **itemStateChanged(ItemEvent e) :** Một abstract method, được gọi khi trạng thái của CheckBox thay đổi.
- **getItem() :** Trả về một đối tượng liên quan đến Item đang được chọn.
- **getStateChange() :** Trả về trạng thái mới nhất của Checkbox,
- **getSource():** Trả về component đã kích hoạt item event.

## Một số hàm thông dụng trên JCheckBox

- **setIcon(Icon i)** – Chỉ định Icon cho checkbox.
- **setText(String text)** – Chỉ định văn bản hiển thị trên checkbox.
- **setSelected(boolean selected)** – Chuyển trạng thái của checkbox thành selected tương ứng nếu TRUE thì checkbox ở trạng thái được chọn ngược lại là không chọn.
- **getIcon()** – Trả về Icon hiện tại của checkbox.
- **getText()** – Trả về văn bản được hiển thị trên checkbox.
- **updateUI()** – Cập nhập giao diện của checkbox với các giá trị mới.
- **getUI()** – Trả về giao diện hiển thị checkbox.
- **paramString()** – Trả về một biểu diễn chuỗi của JCheckBox này.
- **getUIClassID()** – Trả về tên của lớp Giao diện hiển thị thành phần này.
- **getAccessibleContext()** – Lấy AccessibleContext được liên kết với JCheckBox này
- **isBorderPaintedFlat()** – Nhận giá trị của thuộc tính borderPaintedFlat
- **setBorderPaintedFlat(boolean b)** – Đặt thuộc tính borderPaintedFlat.

# JCheckBox – Ví dụ

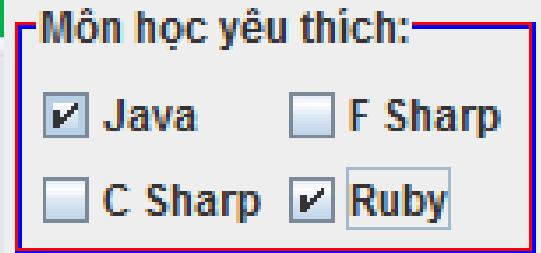
```
public class ViduJCheckBox extends JFrame{  
    public ViduJCheckBox() {  
        JPanel pn = new JPanel();  
        pn.setBorder(BorderFactory.createTitledBorder("Fruits"));  
        //create JCheckBox  
        JCheckBox cb1 = new JCheckBox("Apple",true);  
        JCheckBox cb2 = new JCheckBox("Banana");  
        JCheckBox cb3 = new JCheckBox("Grape",true);  
        JCheckBox cb4 = new JCheckBox("Orange");  
        JCheckBox cb5 = new JCheckBox("Pear",true);  
        //Add the checkbox into the panels  
        pn.add(cb1);  
        pn.add(cb2);  
        pn.add(cb3);  
        pn.add(cb4);  
        pn.add(cb5);  
        // Add the panel into the frame  
        this.add(pn);  
    }  
  
    public void showWindow(String tieude) {  
        this.setTitle(tieude);  
        this.setSize(300, 150);  
        this.setDefaultCloseOperation(EXIT_ON_CLOSE);  
        this.setLocationRelativeTo(null);  
        this.setVisible(true);  
    }  
    public static void main(String[] args) {  
        new ViduJCheckBox().showWindow("JCheckBox Demo");  
    }  
}
```



# JCheckBox

Make multi choice

```
 JPanel pnCheck=new JPanel();
pnCheck.setLayout(new GridLayout(2, 2));
Border bor2=BorderFactory
    .createEtchedBorder(Color.BLUE, Color.RED);
TitledBorder titlebor2=
    new TitledBorder(bor2, "Môn học yêu thích:");
pnCheck.setBorder(titlebor2);
JCheckBox chk1=new JCheckBox("Java");
JCheckBox chk2=new JCheckBox("F Sharp");
JCheckBox chk3=new JCheckBox("C Sharp");
JCheckBox chk4=new JCheckBox("Ruby");
pnCheck.add(chk1);pnCheck.add(chk2);
pnCheck.add(chk3);pnCheck.add(chk4);
add(pnCheck);
```



# JCheckBox

Make multi choice

Set grid layout 2 rows and 2 column

```
pnCheck.setLayout(new GridLayout(2, 2));
```

Create JCheckBox:

```
JCheckBox chk1=new JCheckBox("Java");
```

Add chk1 into the pnCheck:

```
pnCheck.add(chk1);
```

Add pnCheck into the Window:

```
add(pnCheck);
```

Create border with 2 color: Blue, Red:

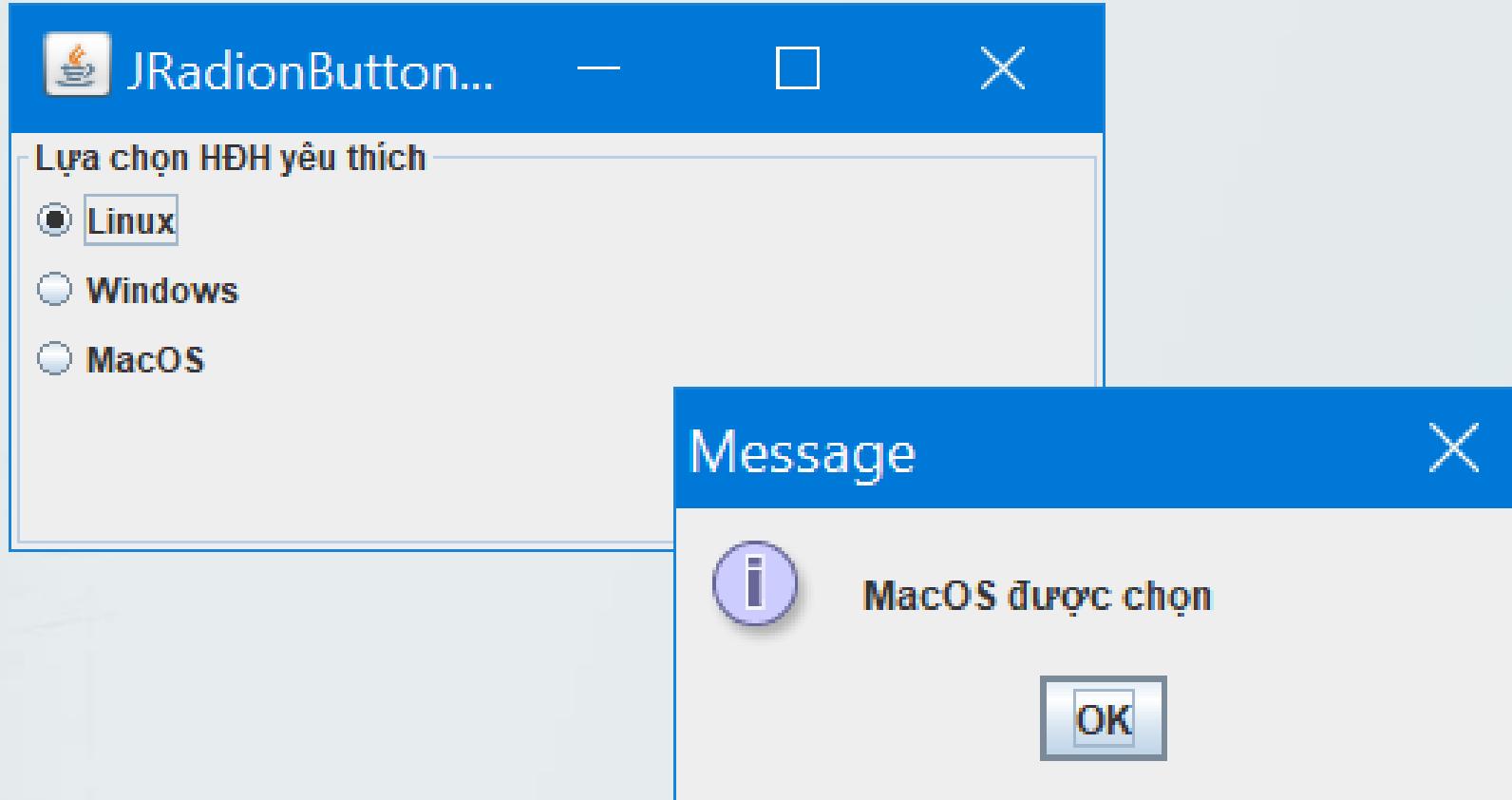
```
BorderFactory.createEtchedBorder(Color.BLUE, Color.RED);
```



```
if(chk1.isSelected())
{
//do something
}
```

- ❑ JRadioButton được dùng để tạo ra các Radio Button **cho phép** người dùng có thể **lựa chọn một trong số các tùy chọn** được cung cấp.
  - ❑ Nó thường được dùng trong các khảo sát về mức độ hài lòng, hay các bài thi trắc nghiệm, ...
- 
- ❑ Khởi tạo RadioButton
    - **JRadioButton()** – Khởi tạo JRadioButton rỗng
    - **JButton(String s)** – Khởi tạo JRadioButton với nhãn được chỉ định.
- 
- ❑ Để sử dụng các JRadioButton chúng ta cần thêm chúng vào một **ButtonGroup** và sử dụng **add()** method để thêm các radio button vào.

# Ví dụ RadioButton đơn giản



# Ví dụ RadioButton đơn giản

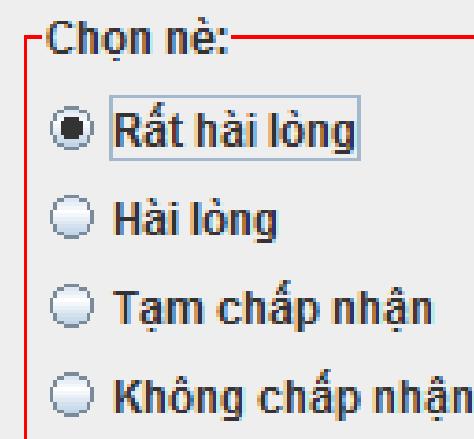
```
public void JRadioButton() {  
    JPanel pn = new JPanel(new FlowLayout());  
    pn.setLayout(new BoxLayout(pn, BoxLayout.Y_AXIS));  
    pn.setBorder(BorderFactory.createTitledBorder("Lựa chọn HDH  
yêu thích"));  
    //pn.setBorder(BorderFactory.createEtchedBorder(Color.BLUE,  
Color.RED));  
    //create JRadioButton  
    JRadioButton rdLinux = new JRadioButton("Linux");  
    JRadioButton rdWindows = new JRadioButton("Windows");  
    JRadioButton rdMacOS = new JRadioButton("MacOS");  
    // Gom nhom 3 JRadioButton vao ButtonGroup  
    ButtonGroup group = new ButtonGroup();  
    group.add(rdLinux);  
    group.add(rdWindows);  
    group.add(rdMacOS);
```

# Ví dụ RadioButton đơn giản

```
ActionListener listener = new ActionListener() {  
    @Override  
    public void actionPerformed(ActionEvent e) {  
        if(e.getSource()==rdLinux) {  
            JOptionPane.showMessageDialog(null, "Linux được chọn");  
        }  
        if(e.getSource()==rdWindows) {  
            JOptionPane.showMessageDialog(null, "Windows được chọn");  
        }  
        if(e.getSource()==rdMacOS) {  
            JOptionPane.showMessageDialog(null, "MacOS được chọn");  
        }  
    }  
};  
    rdLinux.addActionListener(listener);  
    rdWindows.addActionListener(listener);  
    rdMacOS.addActionListener(listener);  
    //Add the 3 JRadioButton into the panels  
    pn.add(rdLinux);  
    pn.add(rdwindows);  
    pn.add(rdMacOS);  
    this.add(pn); //add vao JFrame  
}
```

Make single choice

Must add **JRadioButton** into the **ButtonGroup**



```
if(rad1.isSelected())
{
}
```

# Radio Button - ButtonGroup

```
JPanel pnGroup=new JPanel();
pnGroup.setLayout(new BoxLayout(pnGroup,
BoxLayout.Y_AXIS));
Border bor=BorderFactory.createLineBorder(Color.RED);
TitledBorder titlebor=new TitledBorder(bor, "Chọn nè:");
pnGroup.setBorder(titlebor);
JRadioButton rad1=new JRadioButton("Rất hài lòng");
JRadioButton rad2=new JRadioButton("Hài lòng");
JRadioButton rad3=new JRadioButton("Tạm chấp nhận");
JRadioButton rad4=new JRadioButton("Không chấp nhận");

ButtonGroup group=new ButtonGroup();
group.add(rad1);group.add(rad2);
group.add(rad3);group.add(rad4);
pnGroup.add(rad1);pnGroup.add(rad2);
pnGroup.add(rad3);pnGroup.add(rad4);
add(pnGroup);
```

# Radio Button - ButtonGroup

Create Border with title:

```
Border bor=BorderFactory.createLineBorder(Color.RED);
TitledBorder titlebor=new TitledBorder
                    (bor, "Chọn nè:");
pnGroup.setBorder(titlebor);
```

Define a buttongroup to add all radio:

```
ButtonGroup group=new ButtonGroup();
group.add(rad1);
```

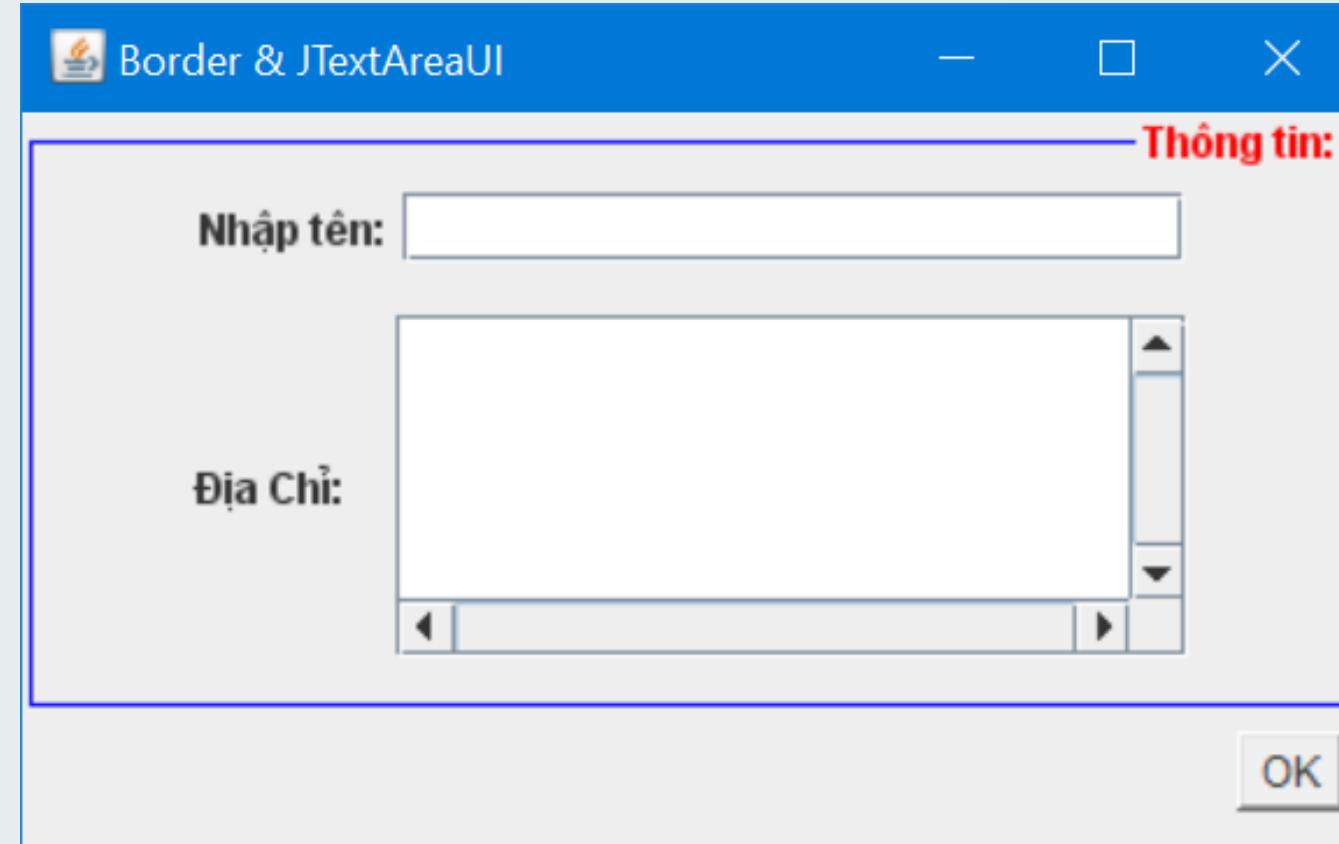
And add all Radio into the pnGroup:

```
pnGroup.add(rad1);
```

Add pnGroupd into the Window:

```
add(pnGroup);
```

## ☐ Thiết kế giao diện như mô tả ảnh



# Bài tập 4

Dòng 1

Dòng 1.1

Thông tin

Nhập tên:

Nguyễn Văn Tèo

Dòng 1.2

Địa Chỉ:

Số 55 đường 22 xóm x , phường y .....

Dòng 2

Sở thích

- ✓ Đi bơi
- ✓ Đi xem phim

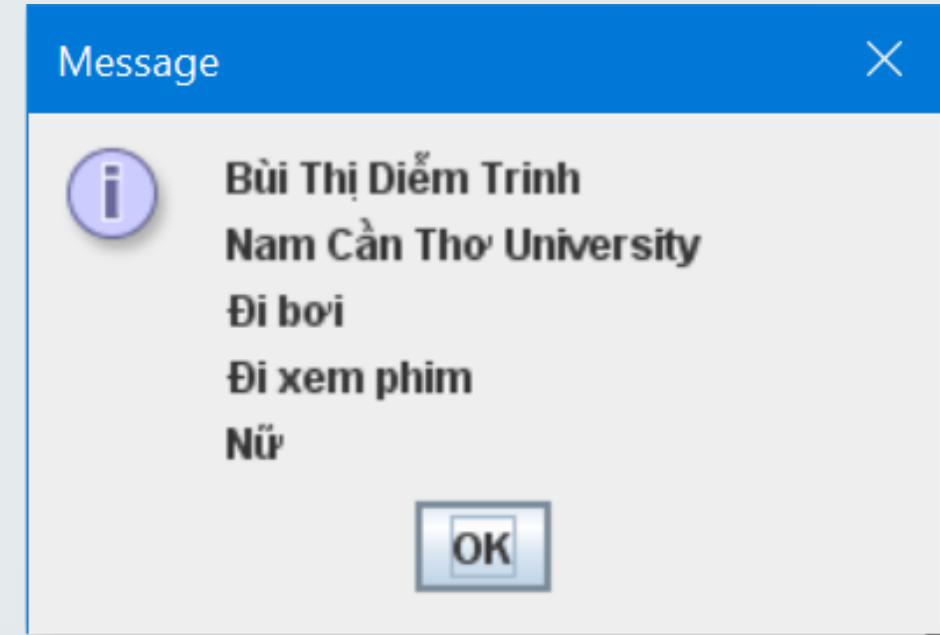
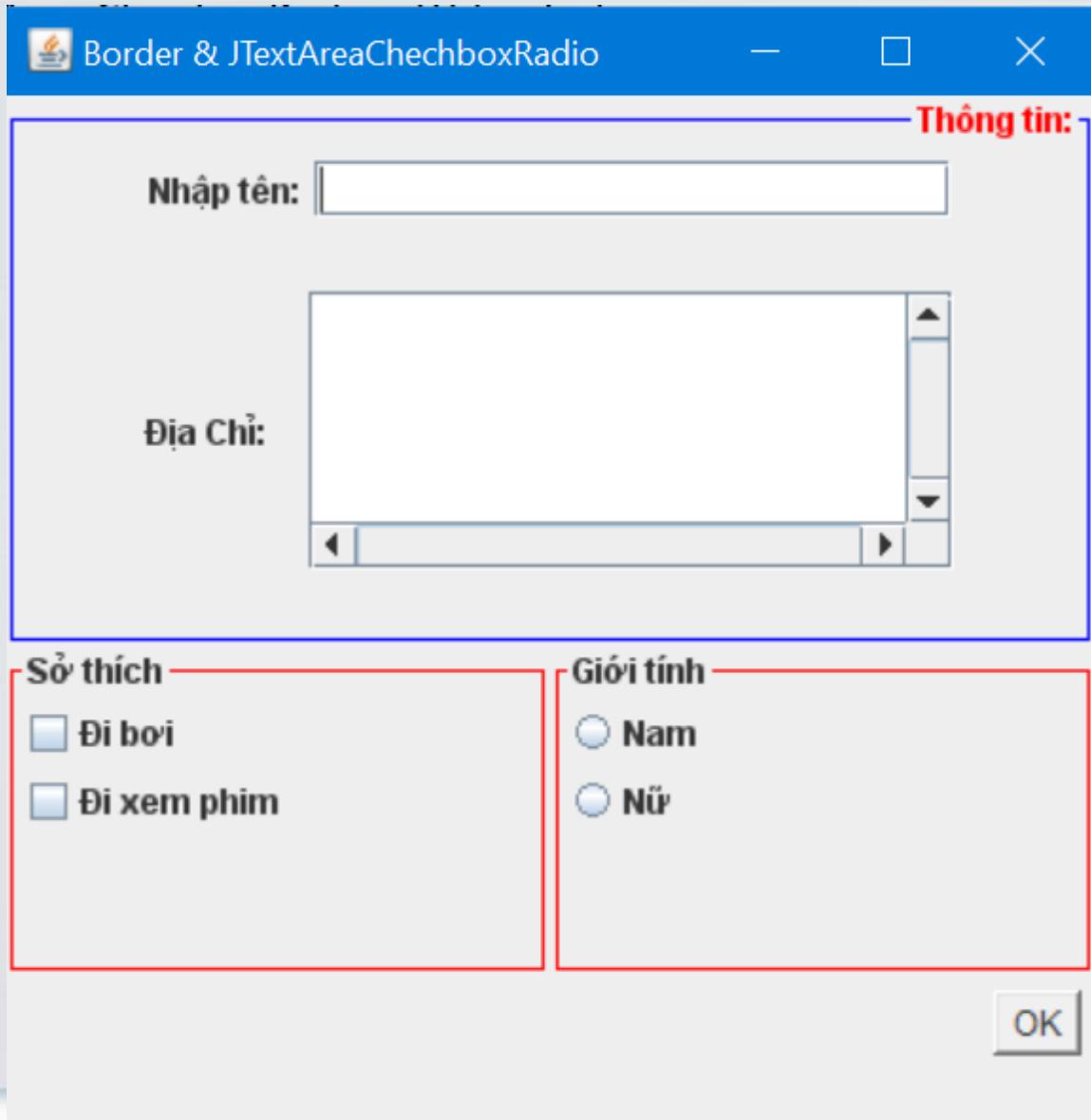
Giới tính

- Nam
- Nữ

Dòng 3

OK

# Bài tập 4



## □ Sử dụng lớp javax.swing.JOptionPane.

- Cung cấp các phương thức chuẩn để gọi một hộp thoại dialog chuẩn cho việc nhận một giá trị hoặc thông báo người dùng về một vấn đề gì đó.
- Được sử dụng rộng rãi hơn Jdialog vì có nhiều phương thức và nhiều tùy chọn hơn.

## □ Các hàm tạo thông dụng

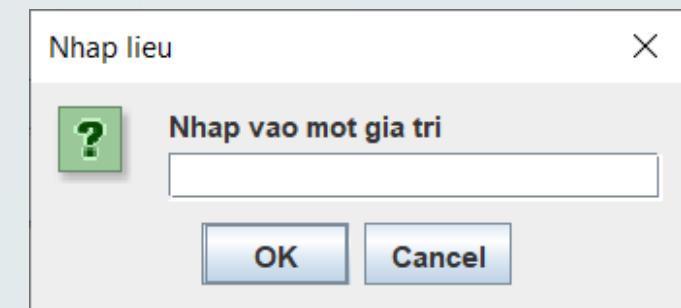
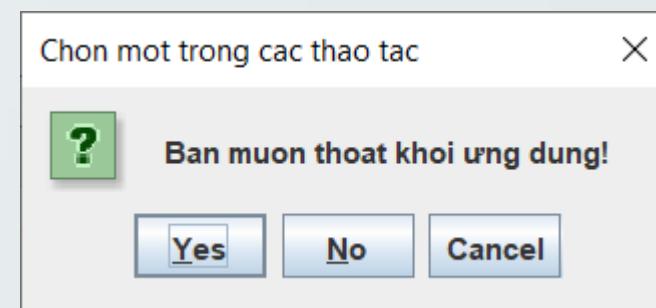
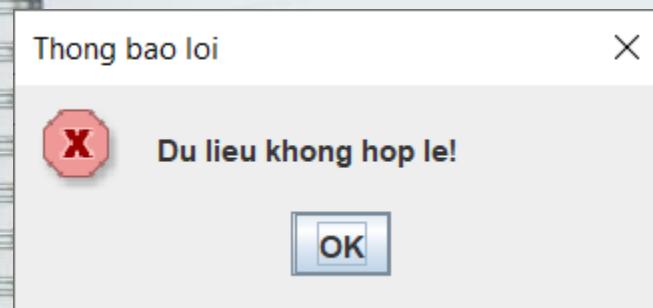
- **JOptionPane(Object message, int messageType):** tạo một thẻ hiện của JOptionPane để hiển thị một thông điệp với kiểu thông điệp đã cho và các tùy chọn mặc định.
- **JOptionPane(Object message, int messageType, int optionType):** tạo một thẻ hiện của JOptionPane để hiển thị một thông điệp với kiểu thông điệp đã cho và các tùy chọn được xác định.
- **JOptionPane(Object message, int messageType, int optionType, Icon icon):** tạo một thẻ hiện của JOptionPane để hiển thị một thông điệp với kiểu thông điệp, tùy chọn và icon đã cho.

## □ Phương thức thông dụng

- **showMessageDialog(Component parentComponent, Object message, String title, int messageType):** hiển thị thông báo với thông điệp, tiêu đề và kiểu hộp thoại cho trước.
- **showConfirmDialog(Component parentComponent, Object message, String title, int optionType):** hiển thị hộp thoại xác nhận với tiêu đề và thông điệp, có thể bao gồm các nút lệnh Yes, No và Cancel tùy theo giá trị biến tùy chọn.
- **showInputDialog(Component parentComponent, Object message, String title, int messageType):** hiển thị hộp thoại nhập liệu với thông điệp, tiêu đề và kiểu hộp thoại cho trước

# JOptionPane – ví dụ

```
import javax.swing.*;  
public class viDuJOptionPane {  
    public viDuJOptionPane() {  
        JFrame frame = new JFrame();  
        JOptionPane.showMessageDialog(frame, "Du lieu khong hop le!", "Thong bao loi", JOptionPane.ERROR_MESSAGE);  
        JOptionPane.showConfirmDialog(frame, "Ban muon thoat khoi ung dung!", "Chon mot trong cac thao tac", JOptionPane.YES_NO_CANCEL_OPTION);  
        JOptionPane.showInputDialog(frame, "Nhap vao mot gia tri", "Nhap lieu", JOptionPane.QUESTION_MESSAGE);  
    }  
    public static void main(String args[]) {  
        viDuJOptionPane option = new viDuJOptionPane();  
    }  
}
```



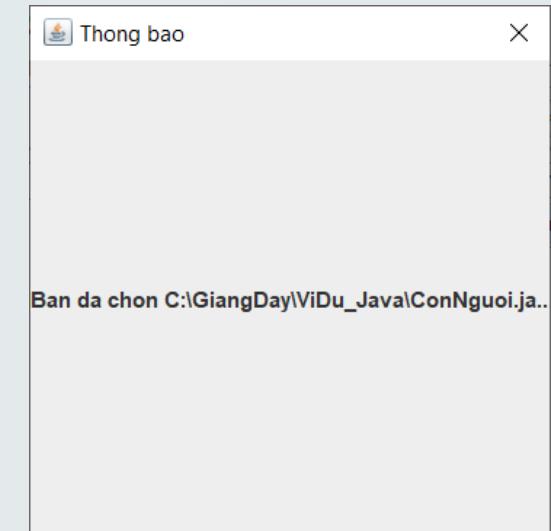
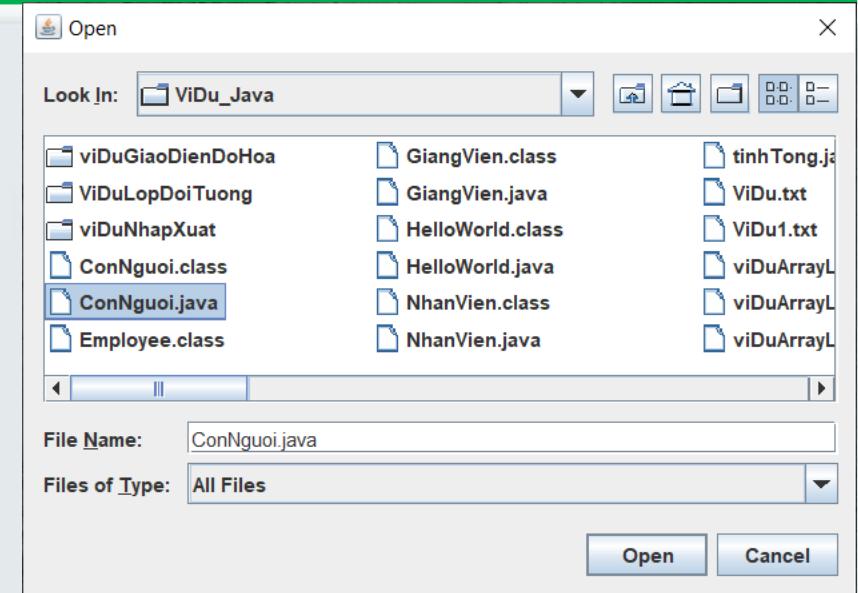
- **Sử dụng lớp Javax.swing.JFileChooser.**
  - Cho phép tạo một hộp thoại để người dùng chọn nhanh một tập tin.
- **Các hàm tạo thông dụng**
  - **JFileChooser()**: xây dựng một JFileChooser trả về thư mục hiện hành của người dùng.
  - **JFileChooser(String currentDirectoryPath)**: xây dựng một JFileChooser trả về thư mục được xác định bởi đường dẫn do người dùng cung cấp.

## □ Phương thức thông dụng

- **int showOpenDialog(Component parent)**: hiển thị hộp thoại dạng “Open file”.
- **int showSaveDialog(Component parent)**: hiển thị hộp thoại dạng “Save file”.
- **void setCurrentDirectory(File dir)**: cho phép thiết lập thư mục mặc định của hộp thoại được xác định bởi dir.

# JFileChooser – ví dụ

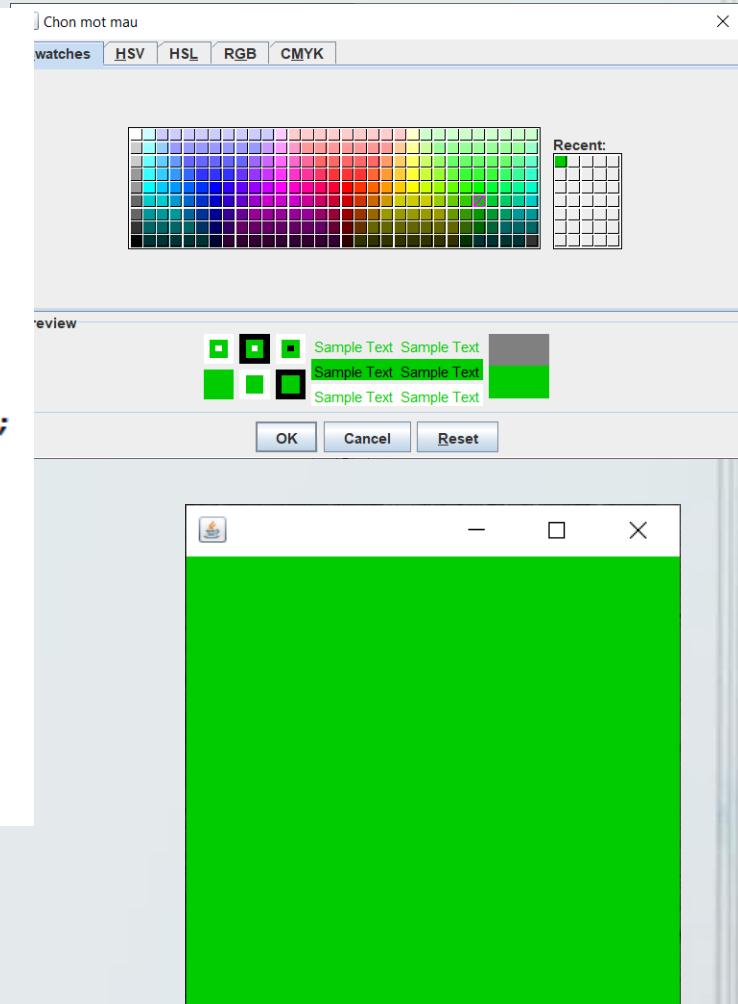
```
import javax.swing.*;
import java.io.*;
public class viDuJFilechooser {
    public viDuJFilechooser(){
        JFrame frame = new JFrame();
        JFileChooser filechooser = new JFileChooser();
        //mở hộp thoại dạng Open file
        int result = filechooser.showOpenDialog(frame);
        if(result == JFileChooser.APPROVE_OPTION){
            File selectedFile = filechooser.getSelectedFile();
            //thiết lập dialog để hiển thị tập tin được chọn
            JDialog dialog = new JDialog(frame, "Thong bao", true);
            JLabel label = new JLabel("Ban da chon "+selectedFile.getAbsolutePath());
            dialog.add(label);
            dialog.setSize(300, 300);
            dialog.setVisible(true);
        }
    }
    public static void main(String args[]) {
        viDuJFilechooser filechooser = new viDuJFilechooser();
    }
}
```



- Sử dụng lớp **Javax.swing.JColorChooser**.
- Cho phép tạo một hộp thoại để người dùng chọn nhanh một màu (color).
- Các hàm tạo thông dụng
  - **JColorChooser()**: xây dựng một JColorChooser với màu được chọn mặc định là màu trắng.
  - **JColorChooser(color initialcolor)**: xây dựng một JColorChooser với màu mặc định được chọn do người dùng cung cấp.
- Phương thức thông dụng
  - **Color showDialog(Component c, String title, Color initialColor)**: cho phép hiển thị hộp thoại chọn màu.

# JColorChooser - ví dụ

```
import javax.swing.*;
import java.awt.*;
public class viDuJColorChooser {
    public viDuJColorChooser() {
        JFrame frame = new JFrame();
        frame.setSize(300, 300);
        Color init = Color.GRAY;
        JColorChooser colorchooser = new JColorChooser();
        Color selectedColor = colorchooser.showDialog(frame, "Chon mot mau", init);
        //đổi màu nền cho frame
        frame.getContentPane().setBackground(selectedColor);
        frame.setVisible(true);
    }
    public static void main(String args[]) {
        viDuJColorChooser colorchooser = new viDuJColorChooser();
    }
}
```



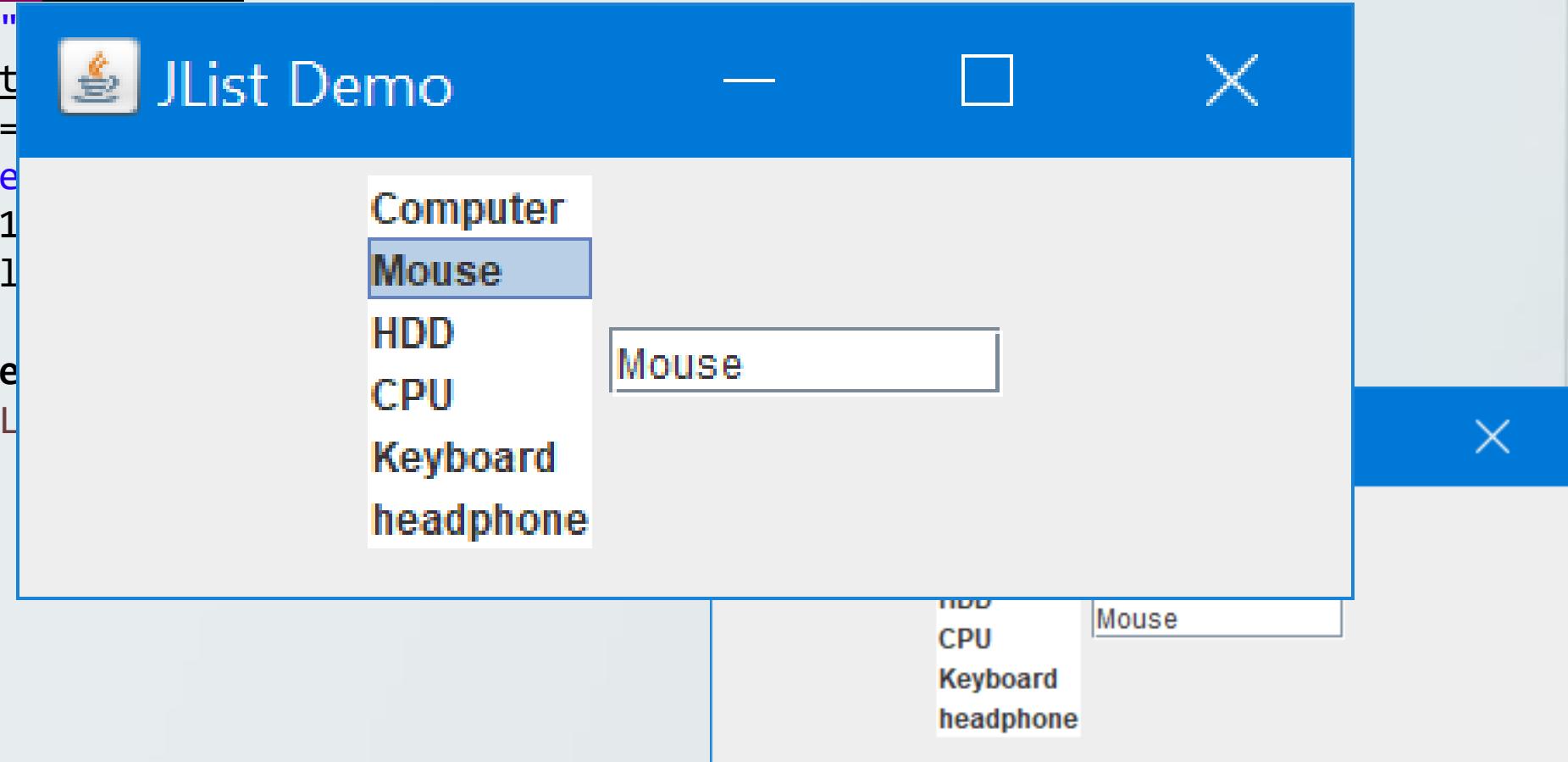
## □ Phương thức thông dụng

- Các phương thức **tương tự** như của lớp **JComboBox**.
- **JList():** tạo một JList rỗng
- **JList(E[] listData):** tạo 1 JList với một mảng dữ liệu
- **List(ListModel dataModel):** tạo JList với một model đã có
- **JList(Vector listData):** tạo JList với dữ liệu trong vecto
- **List getSelectedValuesList():** trả về danh sách các phần tử được chọn.
- **int [] getSelectedValuesList():** trả về mảng chỉ số của các phần tử được chọn.
- **Object getElementAt(int index):** trả về phần tử tại vị trí index của JList.

## ❑ Cách bắt sự kiện

- **addListSelectionListener:** chọn một item trong list
- **valueChaned:** thực hiện viết đè phương thức để hiện công việc mong muốn
- **getSelectedIndex:** để lấy vị trí được chọn trong JList

```
public void ViduJList3() {  
    JPanel pn = new JPanel();  
    JList jList = new JList();  
    String ds[] = {"Computer", "Mouse", "HDD", "CPU", "Keyboard", "headphone"};  
    jList.setListData(ds);  
    JTextField jtf = new JTextField("jText");  
    jtf.setColumns(10);  
    jList.addListSelectionListener(new ListSelectionListener() {  
        @Override  
        public void valueChanged(ListSelectionEvent e) {  
            jtf.setText(ds[jList.getSelectedIndex()]);  
        }  
    });  
    pn.add(jList);  
    pn.add(jtf);  
    this.add(pn);  
}
```



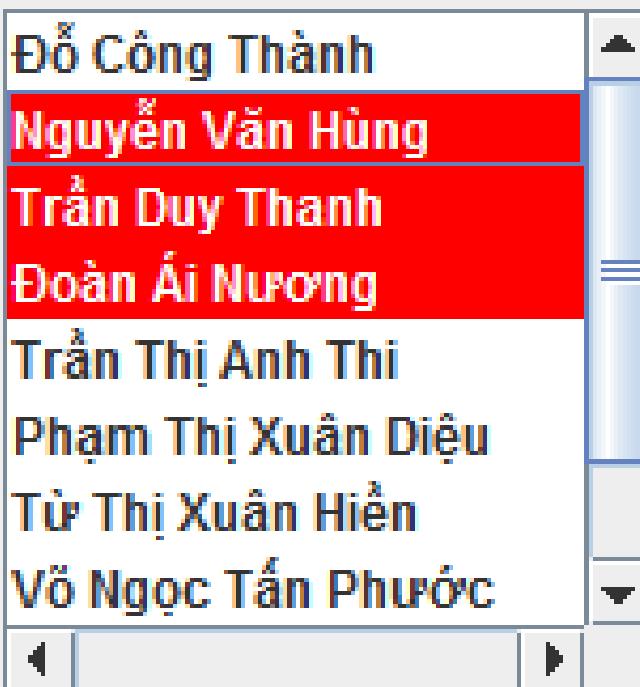
## JList

```
Person []list={  
    new Person("1", "Đỗ Công Thành"),  
    new Person("2", "Nguyễn Văn Hùng"),  
    new Person("3", "Trần Duy Thanh"),  
    new Person("4", "Đoàn Ái Nương"),  
    ...  
    new Person("10", "Đào Cẩm Hằng")  
};  
  
JList jl=new JList(list);  
jl.setSelectionBackground(Color.RED);  
jl.setSelectionForeground(Color.WHITE);  
JScrollPane scjl=new JScrollPane(jl,  
    JScrollPane.VERTICAL_SCROLLBAR_ALWAYS,  
    JScrollPane.HORIZONTAL_SCROLLBAR_ALWAYS);  
add(scjl);
```



## ListSelectionModel

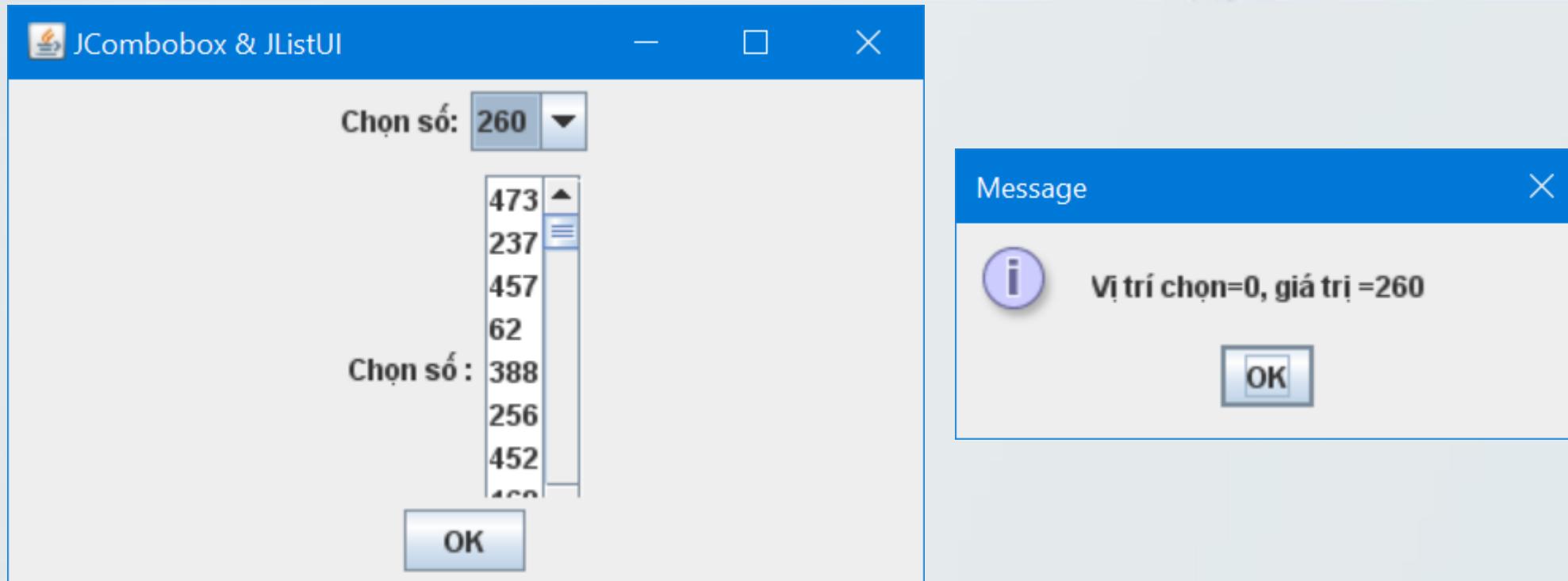
```
ListSelectionModel.SINGLE_SELECTION;  
ListSelectionModel.SINGLE_INTERVAL_SELECTION;  
ListSelectionModel.MULTIPLE_INTERVAL_SELECTION;
```



```
jl.setSelectionMode(ListSelectionMode  
l.MULTIPLE_INTERVAL_SELECTION)
```

```
int n=jl.getSelectedIndex();  
int m[]=jl.getSelectedIndices();  
Object o=jl.getSelectedValue();  
Object arr[]=jl.getSelectedValues();
```

# Bài Tập ? JComboBox, JList



**Công ty có nhiều phòng ban**

- một phòng ban thì có nhiều nhân viên
- Thông tin phòng ban: mã PB, tên PB
- Nhân viên: Mã NV, Tên NV, Ngày Sinh, Ngày vào làm việc

 **Thiết kế giao diện để hiển thị danh sách phòng ban → đưa vào Jcombobox** **Danh sách nhân viên ứng với mỗi phòng ban được đưa vào Jlist** **Cho phép thao tác:**

- Thêm Phòng ban, nhân viên viên, truy vấn tìm kiếm, lưu đọc file

# Bài Tập ?

Dòng 1

Dòng 2

Dòng 3

Chọn phòng ban:

Phòng hợp tác giảng viên

Danh sách nhân viên

Nguyễn văn an  
Trần thị bình  
Hồ văn giải  
Trần đình thoát

Chi tiết

Mã:

NV1:

Tên:

Nguyễn văn an

Ngày  
làm:

01/01/2016

Năm  
sinh:

01/01/1991

Dòng 2.1

Dòng 2.2

Dòng 2.3

Dòng 2.4

Lưu

Xóa

Thoát

# Bài Tập ?

Quản lý nhân viên

Chọn phòng ban: Phòng hợp tác giảng viên

**Danh sách**

Bùi Thị Diễm Trinh
Trần Văn Bình
Võ Văn Phúc

**Thông tin chi tiết**

Mã: NV1

Tên: Bùi Thị Diễm Trinh

Ngày vào: 01/02/2016

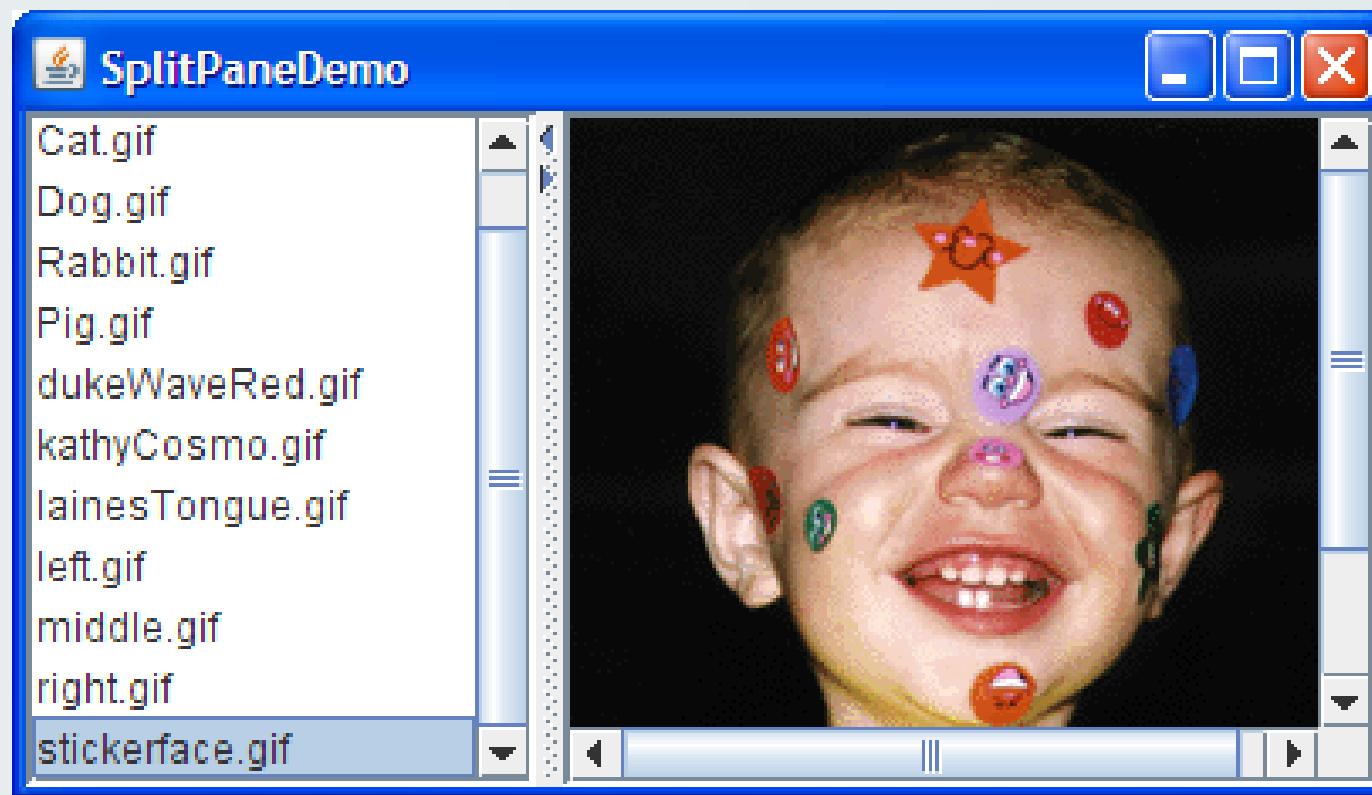
Năm sinh: 01/02/1990

**Chọn chức năng**

Lưu Xóa Thoát

# Split Panes, Jtable, JTree

Chia màn hình theo hướng đứng hoặc nằm, có thể di động nội dung bên trong màn hình

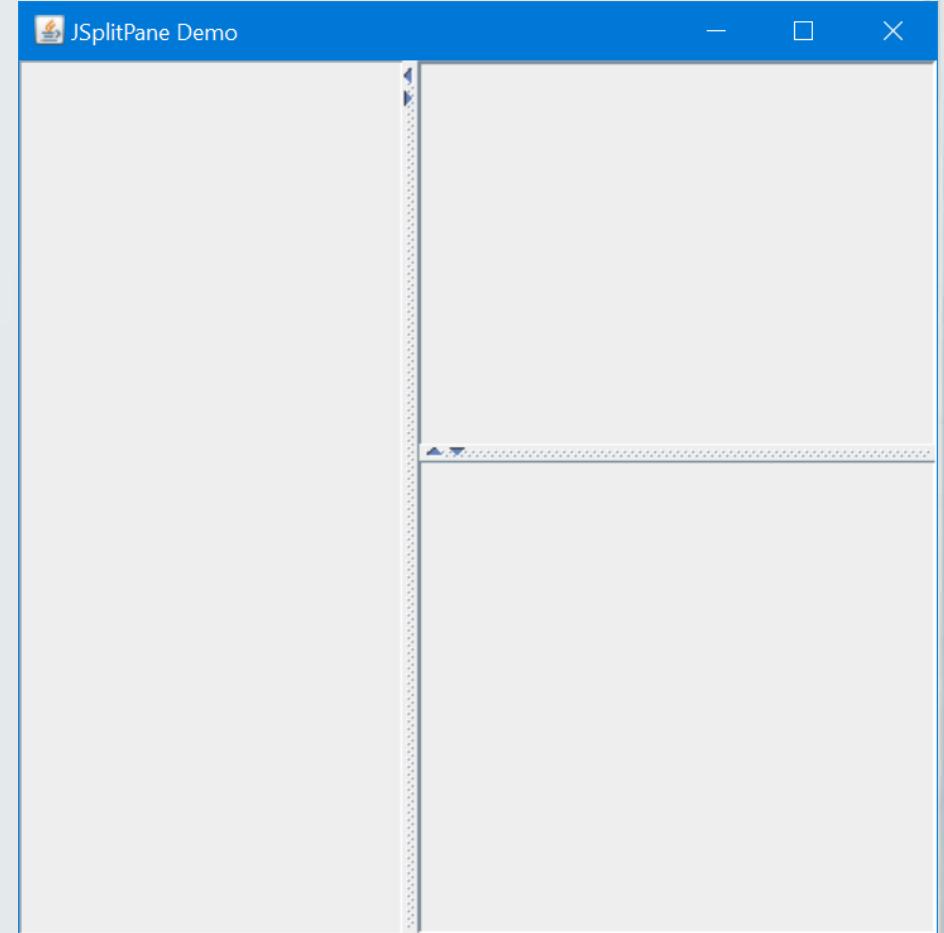


# Split Panes

```
JSplitPane splitPane = new JSplitPane(JSplitPane.HORIZONTAL_SPLIT,
                                         listScrollPane,
                                         pictureScrollPane);
splitPane.setOneTouchExpandable(true);
splitPane.setDividerLocation(150);

//Provide minimum sizes for the two components in
the split pane
Dimension minimumSize = new Dimension(100, 50);
listScrollPane.setMinimumSize(minimumSize);
pictureScrollPane.setMinimumSize(minimumSize);
```

```
public void addControls(){  
    Container con=getContentPane();  
    JPanel pnLeft=new JPanel();  
    pnLeft.setPreferredSize(new Dimension(200, 0));  
    JPanel pnRight=new JPanel();  
    pnRight.setLayout(new BorderLayout());  
    JSplitPane sp=new  
    JSplitPane(JSplitPane.HORIZONTAL_SPLIT,pnLeft,pnRi  
ght);  
    sp.setOneTouchExpandable(true);  
    con.add(sp);  
    JPanel pnRight1=new JPanel();  
    JPanel pnRight2=new JPanel();  
    JSplitPane spRight=new JSplitPane(  
    JSplitPane.VERTICAL_SPLIT, pnRight1, pnRight2);  
    spRight.setOneTouchExpandable(true);  
    pnRight1.setPreferredSize(new Dimension(0, 200));  
    pnRight.add(spRight, BorderLayout.CENTER);  
}  
public void addEvents(){}  
}
```



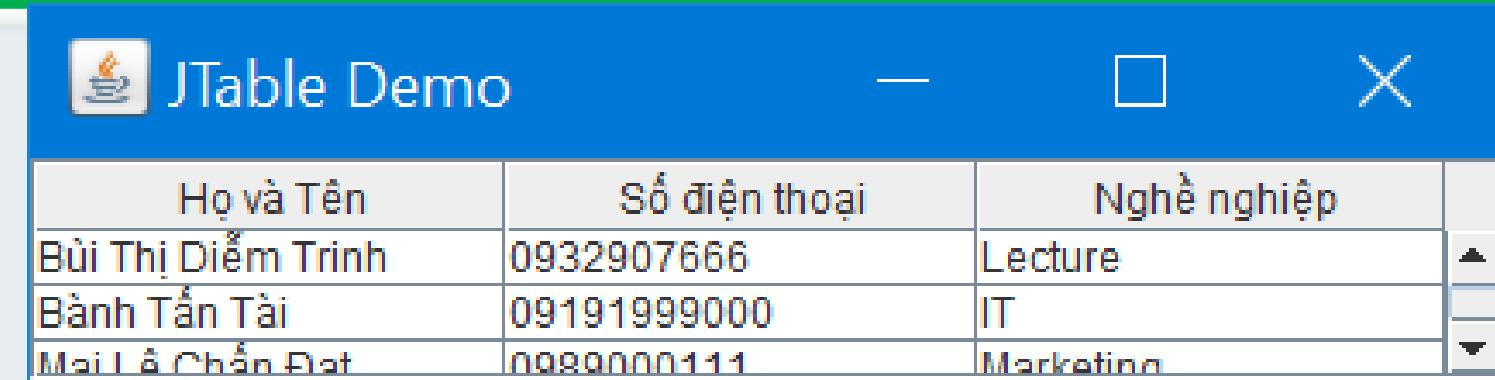
- ❑ **JTable là một trong những thành phần quan trọng của Java Swing package**
- ❑ **Được sử dụng để hiển thị và chỉnh sửa dữ liệu theo dạng Table (Dữ liệu được trình bày theo dạng hàng và cột).**
- ❑ **Một số hàm tạo**
  - **JTable()** – Khởi tạo một JTable rỗng.
  - **JTable(int rows, int cols)** – Khởi tạo một JTable với số dòng và cột được chỉ định.
  - **JTable(Object[][] data, Object []Column)** – Khởi tạo một Jtable với các tên cột được chỉ định trong Object[] Column và dữ liệu trong Object[][] data.

## ❑ Một số phương thức thường dùng

- **addColumn(TableColumn column)** – Thêm một cột vào cuối bảng.
- **editCellAt(int row, int col)** – Chỉnh sửa ô giao nhau của cột số col và hàng số row, nếu các chỉ số đã cho hợp lệ và ô tương ứng có thể chỉnh sửa được.
- **setValueAt(Object value, int row, int col)** – Cập nhật giá trị tạo ô có số cột là col và hàng row thành giá trị value mới.
- **clearSelection()** – Bỏ chọn tất cả các cột và hàng trước đó đã được chọn.

## ❑ Ví dụ: Tạo một bảng gồm 3 cột **Họ Tên**, **Phone** và **Nghề Nghiệp**.

## Jtable - Ví dụ



Họ và Tên	Số điện thoại	Nghề nghiệp
Bùi Thị Diễm Trinh	0932907666	Lecture
Bành Tấn Tài	09191999000	IT
Mai Lê Chấn Đạt	0989000111	Marketing

```
public VíduJTable() {  
    //data hien thi trong table  
    String[][] data = {{"Bùi Thị Diễm Trinh", "0932907666", "Lecture"},  
    {"Bành Tấn Tài", "09191999000", "IT"},  
    {"Mai Lê Chấn Đạt", "0989000111", "Marketing"}};  
    //ten cot  
    String[] col = {"Họ và Tên", "Số điện thoại", "Nghề nghiệp"};  
    //Initializing the JTable  
    JTable tb = new JTable(data,col);  
    tb.setBounds(30, 40, 200, 300);  
    //add into ScrollPane  
    JScrollPane sp = new JScrollPane(tb);  
    this.add(sp);  
}
```

- Sinh viên thực hiện tạo table như mẫu đã thiết kế, tiến hành sắp xếp dữ liệu trong trong table

Họ và Tên	Tuổi	Địa Chỉ
Trần Trọng Khôi	20	Sóc Trăng
Bành Tân Tài	22	Vĩnh Long
Mai Lê Chấn Đạt	30	Trà Vinh
Bùi Thị Điểm Trinh	32	Nam Cần Thơ

Gợi ý sd thuộc tính

//Cho phép table sắp xếp dữ liệu  
table.setAutoCreateRowSorter(true);

# Hiển thị dữ liệu theo dạng lưới

The Header contains  
Column labels

First Name	Last Name	Sport	# of Years	Vegetarian
Kathy	Smith	Snowboarding	5	<input type="checkbox"/>
John	Doe	Rowing	3	<input checked="" type="checkbox"/>
Sue	Black	Knitting	2	<input type="checkbox"/>
Jane	White	Speed reading	20	<input checked="" type="checkbox"/>

Each Cell displays  
a data item

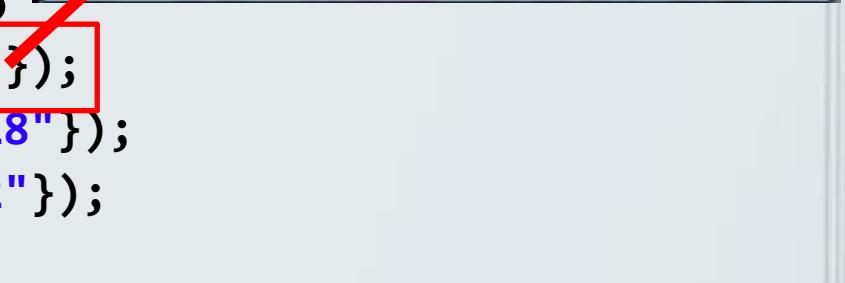
Each Column displays  
one type of data

## JTable

```
DefaultTableModel dm=new DefaultTableModel();  
dm.addColumn("Mã"),  
dm.addColumn("Tên");  
dm.addColumn("Tuổi");
```

```
final JTable tbl=new JTable(dm);  
dm.addRow(new String[]{"112","Ngô văn Bắp","21"});  
dm.addRow(new String[]{"113","Nguyễn Thị Tý","18"});  
dm.addRow(new String[]{"114","Trần Văn Tèo","22"});
```

```
JScrollPane sc=new JScrollPane(tbl);  
Container con=getContentPane();  
con.setLayout(new BorderLayout());  
con.add(sc,BorderLayout.CENTER);
```



Mã	Tên	Tuổi
112	Ngô văn Bắp	21
113	Nguyễn Thị Tý	18
114	Trần Văn Tèo	22

# How to handle event : Mouse, Key from JTable?

```
tbl.addMouseListener(new MouseListener() {  
    public void mouseReleased(MouseEvent e) {}  
    public void mousePressed(MouseEvent e) {}  
    public void mouseExited(MouseEvent e) {}  
    public void mouseEntered(MouseEvent e) {}  
    public void mouseClicked(MouseEvent e) {  
        int row=tbl.getSelectedRow();  
        int col=tbl.getSelectedColumn();  
        String s=(String)tbl.getValueAt(row, col);  
        JOptionPane.showMessageDialog(null, s);  
    }});
```

# How to handle event : Mouse, Key from JTable?

```
tbl.addKeyListener(new KeyListener() {  
    public void keyTyped(KeyEvent arg0) {}  
    public void keyReleased(KeyEvent arg0) {  
        int row=tbl.getSelectedRow();  
        int col=tbl.getSelectedColumn();  
        String s=(String)tbl.getValueAt(row, col);  
        JOptionPane.showMessageDialog(null, s);  
    }  
    public void keyPressed(KeyEvent arg0) {}  
});
```

dm.setRowCount(0); Clear all data from JTable

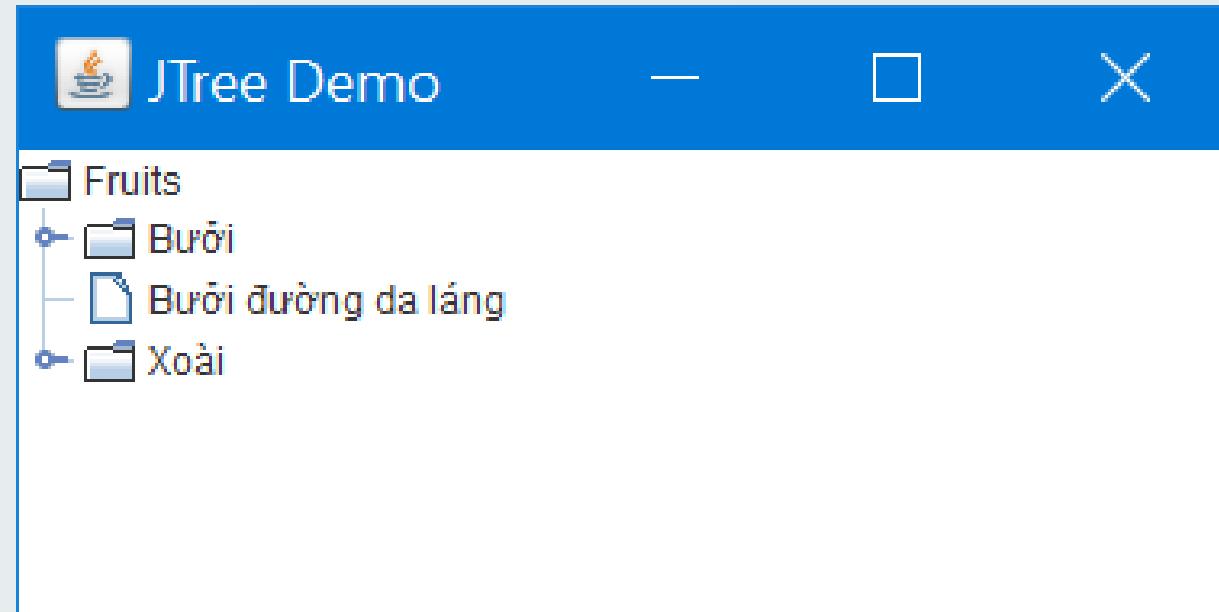
dm.getRowCount(); Get total row from JTable

## 2 Ways to add new Row:

```
dm.addRow(  
    new String[]{"ID_002", "Võ Tòng", "32"});  
Vector<String>vec=new Vector<String>();  
vec.add("ID_003");  
vec.add("Lâm Sung");  
vec.add("30");  
dm.addRow(vec);
```

Mã	Tên	Tuổi
112	Ngô văn Bắp	21
113	Nguyễn Thị Tý	18
114	Trần Văn Tèo	22
ID_002	Võ Tòng	32
ID_003	Lâm Sung	30

- ❑ JTree là một thành phần Swing **hiển thị dữ liệu phân cấp**, nó là một component khá phức tạp với một nút cha được xem là cha (gốc) của tất cả các node con trong cấu trúc cây. Một nút là một phần tử trong một cây.
- ❑ Một nút có thể có nhiều nút con. Bản thân các nút con này có thể có các nút con khác.
- ❑ Nếu một nút không có bất kỳ nút con nào, nó được gọi là nút lá.



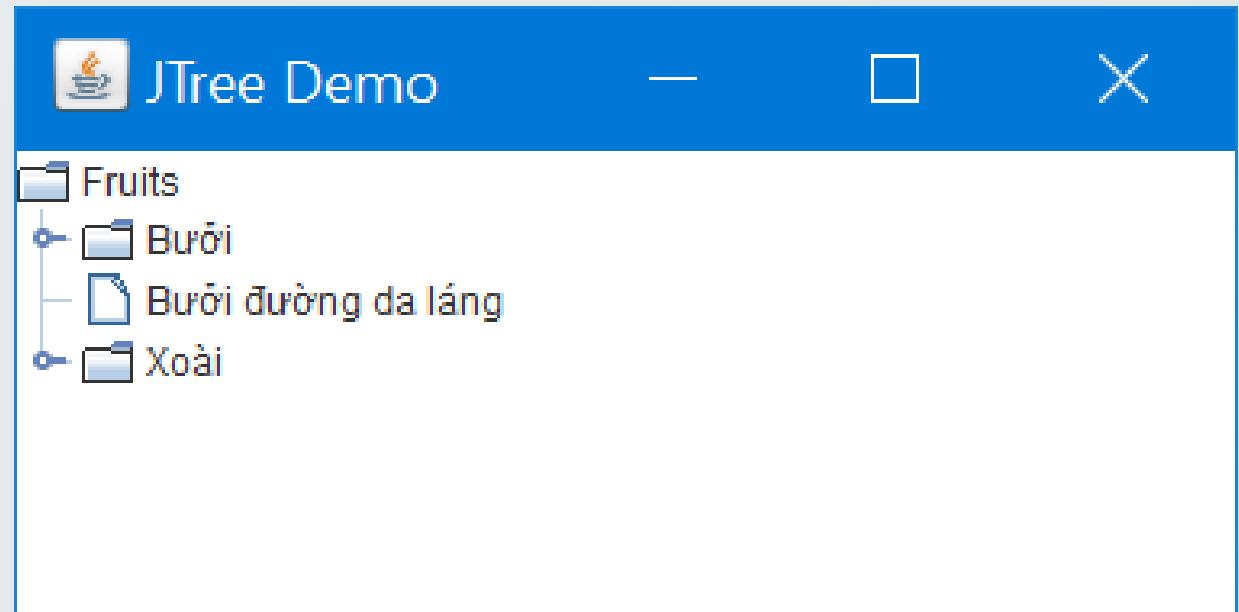
# JTree - ví dụ

```
DefaultMutableTreeNode root=null;  
JTree tree;  
  
public void JTree() {  
    Container con=getContentPane();  
    JPanel pnTree=new JPanel();  
    pnTree.setLayout(new BorderLayout());  
    root=new DefaultMutableTreeNode("Fruits");  
    tree=new JTree(root);  
  
    DefaultMutableTreeNode nodeBuoi=new  
    DefaultMutableTreeNode("Bưởi");  
    root.add(nodeBuoi);  
    DefaultMutableTreeNode node1=new  
    DefaultMutableTreeNode("Bưởi da xanh");  
    nodeBuoi.add(node1);  
    DefaultMutableTreeNode node2=new  
    DefaultMutableTreeNode("Bưởi nǎm roi");  
    nodeBuoi.add(node2);
```

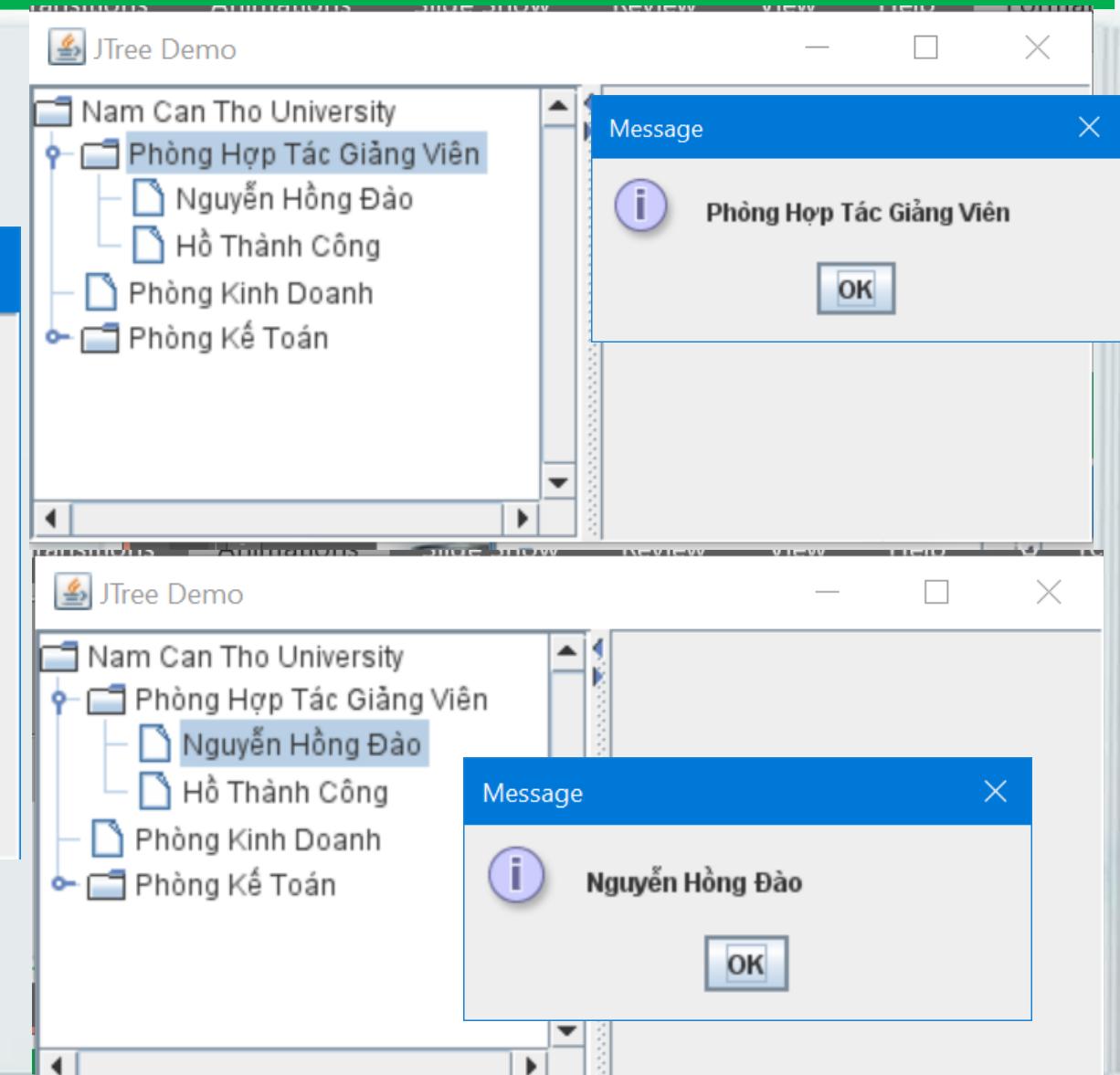
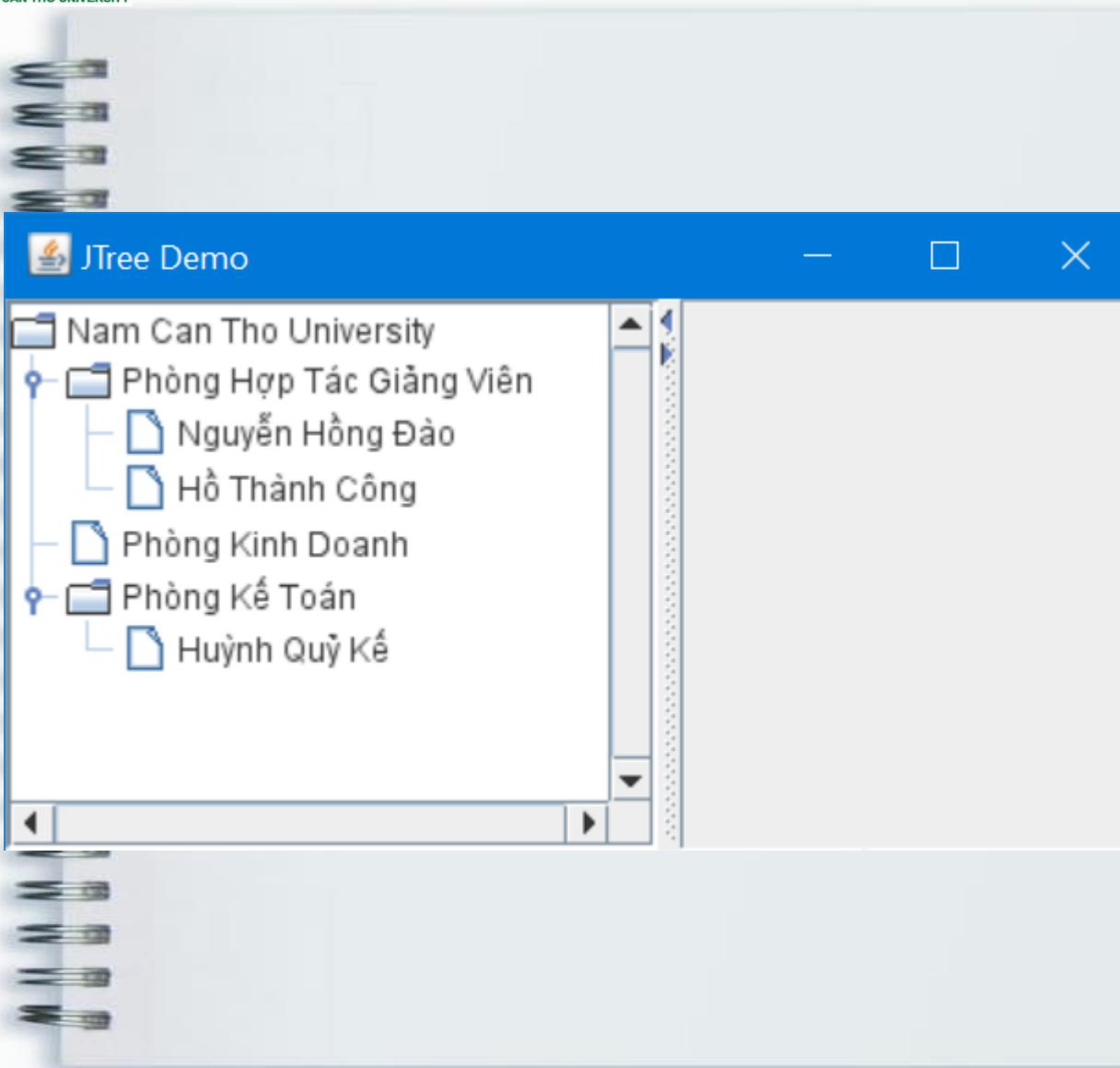
```
DefaultMutableTreeNode nodePhKd=new  
DefaultMutableTreeNode("Bưởi đường da láng");  
root.add(nodePhKd);  
  
DefaultMutableTreeNode nodeXoài=new  
DefaultMutableTreeNode("Xoài");  
root.add(nodeXoài);  
  
DefaultMutableTreeNode nodeX1=new  
DefaultMutableTreeNode("Xoài Cát Hòa Lộc");  
nodeXoài.add(nodeX1);  
DefaultMutableTreeNode nodeX2=new  
DefaultMutableTreeNode("Xoài Tứ Quý");  
nodeXoài.add(nodeX2);  
DefaultMutableTreeNode nodeX3=new  
DefaultMutableTreeNode("Xoài Thanh Ca");  
nodeXoài.add(nodeX3);  
tree.expandRow(0);  
con.add(tree);  
}
```

## JTree - ví dụ

```
public void showWindow(){  
    this.setTitle("JTree Demo");  
    this.setSize(400, 200);  
    this.setDefaultCloseOperation(EXIT_ON_CLOSE);  
    this.setLocationRelativeTo(null);  
    this.setVisible(true);  
}  
public static void main(String[] args) {  
    new VieuJTree().showWindow();  
}
```



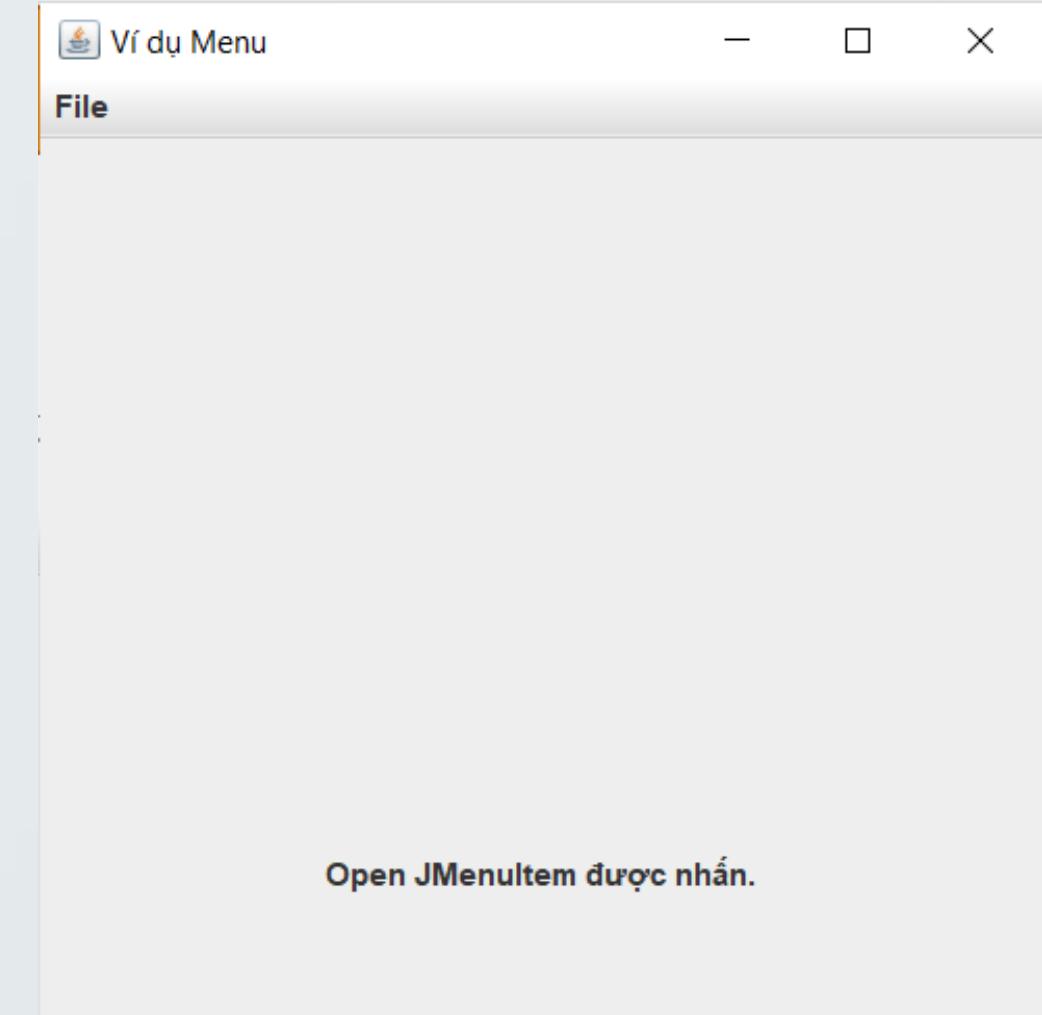
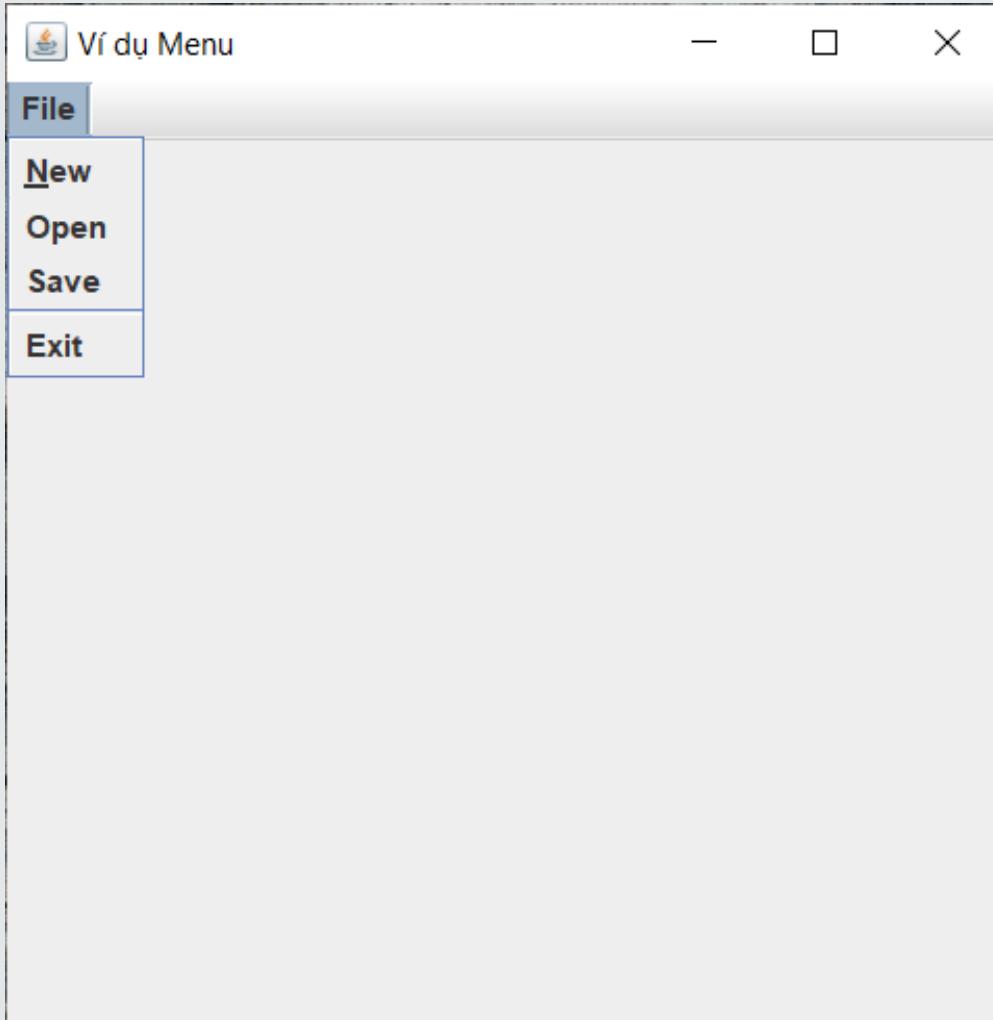
# Bài tập



# JMenuBar-ContextMenu-JToolbar

- Tạo 1 trình đơn sử dụng đối tượng Jmenubar.**
  - Sử dụng phương thức khởi tạo JMenuBar().
- Thêm các đối tượng JMenu vào JMenubar.**
  - Sử dụng phương thức khởi tạo JMenu(String menuName).
- Thêm các đối tượng JMenuItem vào Jmenu.**
  - Sử dụng phương thức khởi tạo **JMenuItem(String menuItemName)**.
- Sử dụng phương thức setActionCommand(String menuItemName) để xác định mục menu nào được nhấn.**
  - Có thể thiết lập phím tắt bằng phương thức setMnemonic(char key).
- Gắn JMenubar vào JFrame.**
- Mặc định các JMenuItem là enabled.**
- Có thể vô hiệu hóa JMenuItem bằng cách gọi phương thức **setEnabled(boolean b)**

# Ví dụ Tạo trình đơn JMenuBar



# Ví dụ Tạo trình đơn JMenuBar

```
private void prepareGUI() {
    mainFrame = new JFrame("Ví dụ Menu");
    mainFrame.setSize(400, 400);
    mainFrame.setLayout(new GridLayout(3, 1));

    headerLabel = new JLabel("", JLabel.CENTER);
    statusLabel = new JLabel("", JLabel.CENTER);

    statusLabel.setSize(350, 100);
    //sử dụng sự kiện để thoát hệ thống khi frame được đóng
    mainFrame.addWindowListener(new WindowAdapter() {
        public void windowClosing(WindowEvent windowEvent) {
            System.exit(0);
        }
    });
    controlPanel = new JPanel();
    controlPanel.setLayout(new FlowLayout());

    mainFrame.add(headerLabel);
    mainFrame.add(controlPanel);
    mainFrame.add(statusLabel);
    mainFrame.setVisible(true);
}
```

```
private void showMenuDemo() {
    //Tạo menu bar
    final JMenuBar menuBar = new JMenuBar();
    //Tạo các menu
    JMenu fileMenu = new JMenu("File");
    //Tạo các menu item
    JMenuItem newItem = new JMenuItem("New");
    newItem.setMnemonic(KeyEvent.VK_N);
    newItem.setActionCommand("New");
    JMenuItem openMenuItem = new JMenuItem("Open");
    openMenuItem.setActionCommand("Open");
    JMenuItem saveMenuItem = new JMenuItem("Save");
    saveMenuItem.setActionCommand("Save");
    JMenuItem exitMenuItem = new JMenuItem("Exit");
    exitMenuItem.setActionCommand("Exit");
    //đăng ký lắng nghe sự kiện click cho các menu item
    MenuItemListener menuItemListener = new MenuItemListener();
    newItem.addActionListener(menuItemListener);
    openMenuItem.addActionListener(menuItemListener);
    saveMenuItem.addActionListener(menuItemListener);
    exitMenuItem.addActionListener(menuItemListener);
    //Thêm các menu item vào menu
    fileMenu.add(newItem);
    fileMenu.add(openMenuItem);
    fileMenu.add(saveMenuItem);
    fileMenu.addSeparator();
    fileMenu.add(exitMenuItem);
    //Thêm menu vào menu bar
    menuBar.add(fileMenu);
    //Thêm menubar toi frame
    mainFrame.setJMenuBar(menuBar);
    mainFrame.setVisible(true);
}
```

## Bước 1: Tạo 1 Lớp PopUp kế thừa từ JPopupMenu

```
class PopUpDemo extends JPopupMenu {  
    JMenuItem anItem;  
    public PopUpDemo () {  
        anItem = new JMenuItem("Click Me!");  
        add(anItem);  
    }  
}
```

## Bước 2 Tạo 1 Lớp để lắng sự kiện nhấn chuột

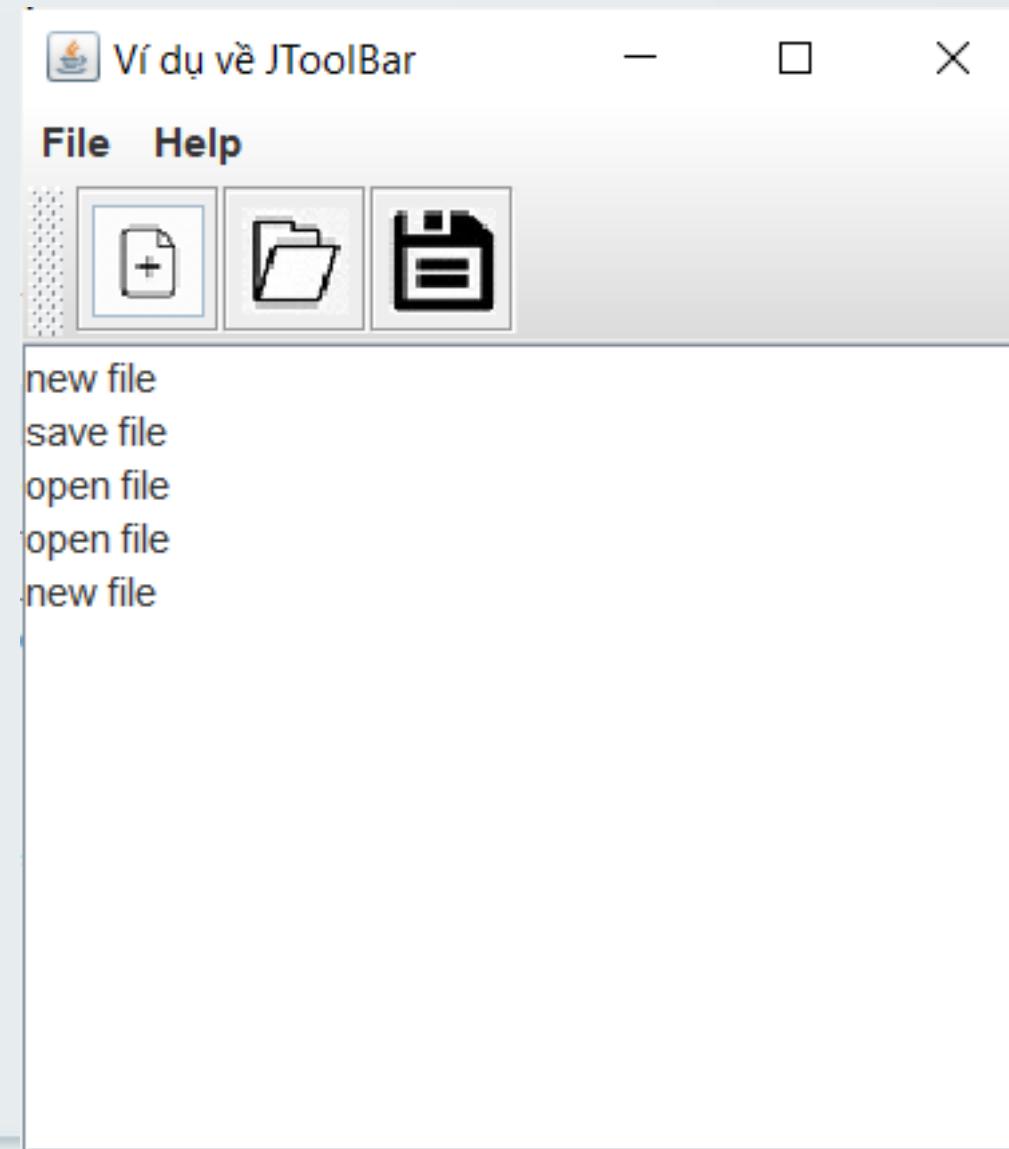
```
class PopClickListener extends MouseAdapter {  
    public void mousePressed(MouseEvent e) {  
        if (e.isPopupTrigger())  
            doPop(e);  
    }  
  
    public void mouseReleased(MouseEvent e) {  
        if (e.isPopupTrigger())  
            doPop(e);  
    }  
  
    private void doPop(MouseEvent e) {  
        PopUpDemo menu = new PopUpDemo();  
        menu.show(e.getComponent(), e.getX(), e.getY());  
    }  
}
```

# ContextMenu

```
component.addMouseListener(new PopClickListener());
```

- ❑ Sử dụng lớp JToolbar.
- ❑ JToolBar là một thanh công cụ dài gồm các nút với biểu tượng tương ứng thường nằm dưới menu bar.
- ❑ Thành phần GUI thường được thêm vào thanh công cụ là:
  - Button
  - Textfield
- ❑ Thường gắn Toolbar vào cạnh trên (trang đầu) của BorderLayout.
- ❑ Có thể thêm vào ngăn cách bằng cách gọi phương thức addSeparator().

# JToolBar



# JToolBar

```
//Tạo tool bar
private JToolBar createToolBar() {
    JToolBar tb = new JToolBar();
    tb.add(createButton(iconNew, "New", NEW, "Create a new file"));
    tb.add(createButton(iconOpen, "Open", OPEN, "Open a file"));
    tb.add(createButton(iconSave, "Save", SAVE, "Save this file"));
    return tb;
}

//Tạo các nút lệnh với icon
private JButton createButton(String iconLink, String text, String command, String toolTip) {
    JButton btn = new JButton();
    btn.setActionCommand(command);
    btn.setToolTipText(toolTip);

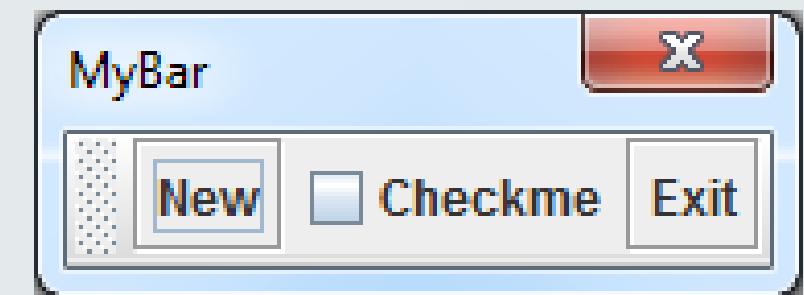
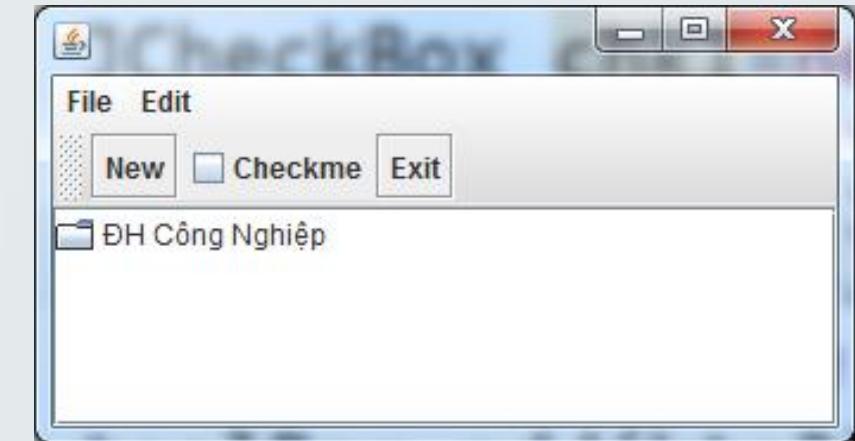
    ImageIcon icon = createIcon(iconLink);
    if (icon == null) {
        btn.setText(text);
    } else {
        btn.setIcon(icon);
    }
    btn.setActionCommand(command);
    btn.addActionListener(this);
    return btn;
}

//Tạo các icon cho nút lệnh
private ImageIcon createIcon(String iconLink) {
    if (iconLink != null) {
        return new ImageIcon(iconLink);
    } else {
        System.out.println("image " + iconLink + " not found");
        return null;
    }
}
```

# JToolBar

```
JToolBar toolBar=new JToolBar("MyBar");
JButton btn1=new JButton("New");
JCheckBox chk1=new JCheckBox("Checkme");
toolBar.add(btn1);
toolBar.add(chk1);
JButton btn2=new JButton("Exit");
toolBar.add(btn2);
add(toolBar,BorderLayout.NORTH);
```

Handle event as the same JButton



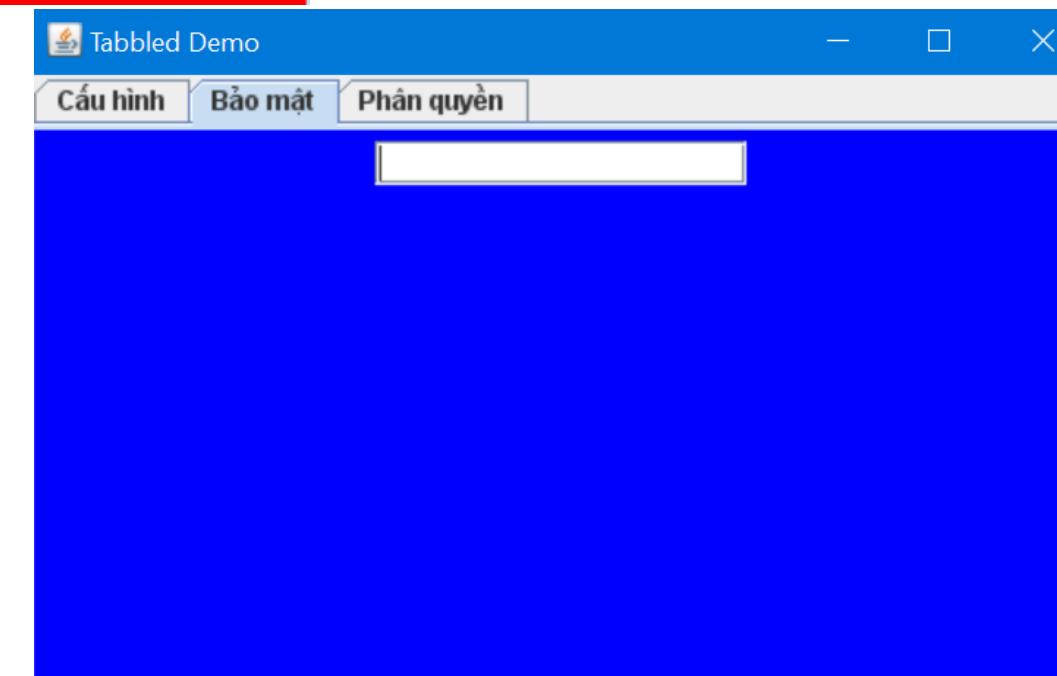
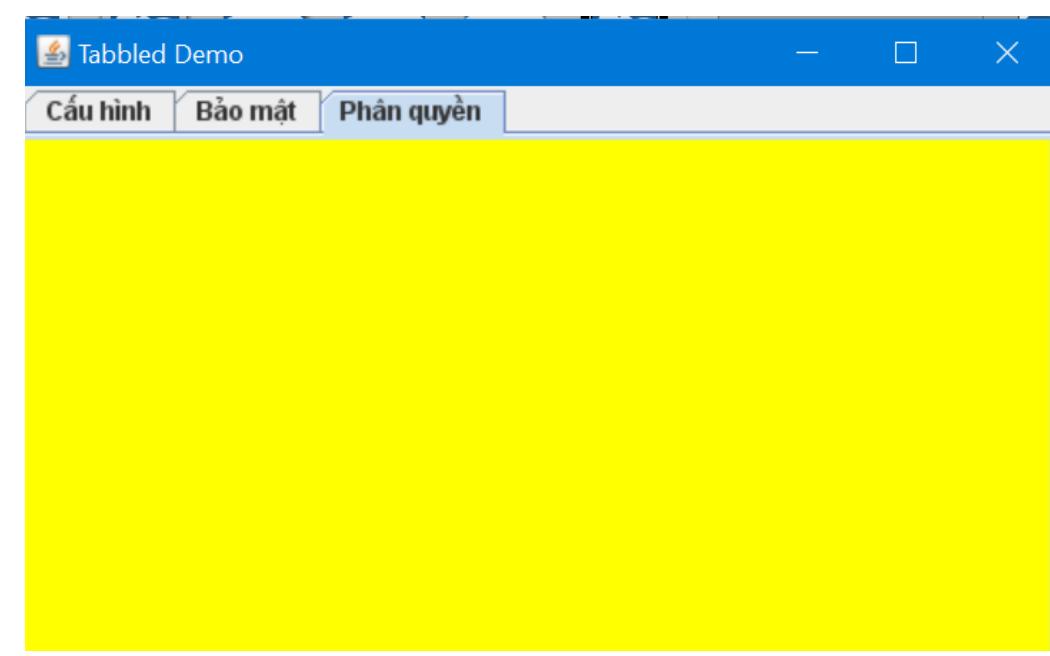
# JTabbedPane

JTabbedPane có giao diện đẹp mắt và thân thiện với người sử dụng, dùng để chia màn hình sử dụng:



# JTabbedPane

```
JTabbedPane myTabled=new JTabbedPane();
JPanel pnTab1=new JPanel();
pnTab1.setBackground(Color.BLUE);
pnTab1.add(new JButton("Tabbed 1"));
JPanel pnTab2=new JPanel();
pnTab2.setBackground(Color.ORANGE);
pnTab2.add(new JButton("Tabbed 2"));
myTabled.add(pnTab1,"Tab1");
myTabled.add(pnTab2,"Tab2");
Container con=ContentPane();
con.add(myTabled);
```



```
public class HocJTabbedPaneUI extends JFrame{  
  
JTabbedPane tab;  
  
public HocJTabbedPaneUI(String title)  
{  
super(title);  
addControls();  
addEvents();  
}  
private void addEvents() {  
// TODO Auto-generated method stub  
}  
  
private void addControls() {  
Container con=getContentPane();  
tab=new JTabbedPane();  
con.add(tab);
```

```
JPanel pnTab1=new JPanel();  
JPanel pnTab2=new JPanel();  
JPanel pnTab3=new JPanel();  
tab.add(pnTab1,"Cấu hình");  
tab.add(pnTab2,"Bảo mật");  
tab.add(pnTab3,"Phân quyền");  
  
pnTab1.setBackground(Color.RED);  
pnTab2.setBackground(Color.BLUE);  
pnTab3.setBackground(Color.YELLOW);  
  
pnTab1.add(new JButton("Đây là button"));  
pnTab2.add(new JTextField(15));  
}  
  
public void showWindow()  
{  
this.setSize(500, 500);  
this.setDefaultCloseOperation(EXIT_ON_CLOSE);  
this.setLocationRelativeTo(null);  
this.setVisible(true);  
}  
}
```

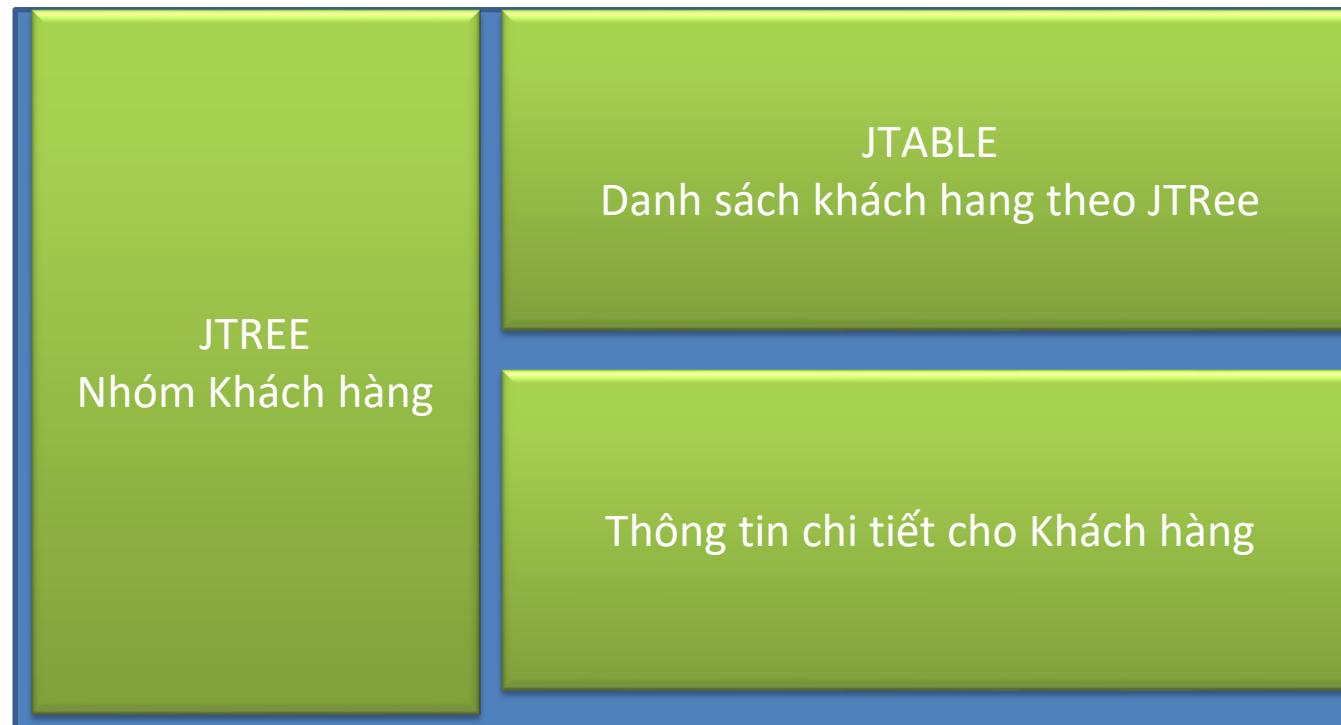
# Quản Lý Khách Hàng

## Nhóm khách hàng

- Khách hàng VIP
- Khách hàng tiềm năng

Khách : mã khách, tên khách, phone, email

Bên trái là Jtree, bên phải JTable





- 📁 Công ty TNHH Nam Cần Thơ
- ↳ 📁 Khách hàng VIP
- ↳ 📁 Khách hàng Tiềm Năng
- ↳ 📁 Khách hàng Khó Tính hay nhăn nhít

Mã khách hàng	Tên khách hàng	Số điện thoại	Thư điện tử
k1	An	0981234567	an@gmail.com
k2	Bình	0971234567	binh@gmail.com
k3	Giải	0951234567	giai@gmail.com
k4	Thoát	0961234567	thoat@gmail.com
k5	Trinh	0932907666	trinh@gmail.com

Mã khách hàng:

Tên khách hàng:

Phone khách hàng:

Email khách hàng:

Lưu khách hàng

Xóa khách hàng



- 📁 Công ty TNHH Nam Cần Thơ
- ↳ 📁 Khách hàng VIP
- ↳ 📁 Khách hàng Tiềm Năng
- ↳ 📁 Khách hàng Khó Tính hay nhăn nhít

Mã khách hàng	Tên khách hàng	Số điện thoại	Thư điện tử
TN01	Bành Tẩn Tài	09099090999	tai@gmail.com
TN02	Nguyễn Thế Hưng	09099090999	hung@gmail.com

Mã khách hàng:

Tên khách hàng:

Phone khách hàng:

Email khách hàng:

Lưu khách hàng

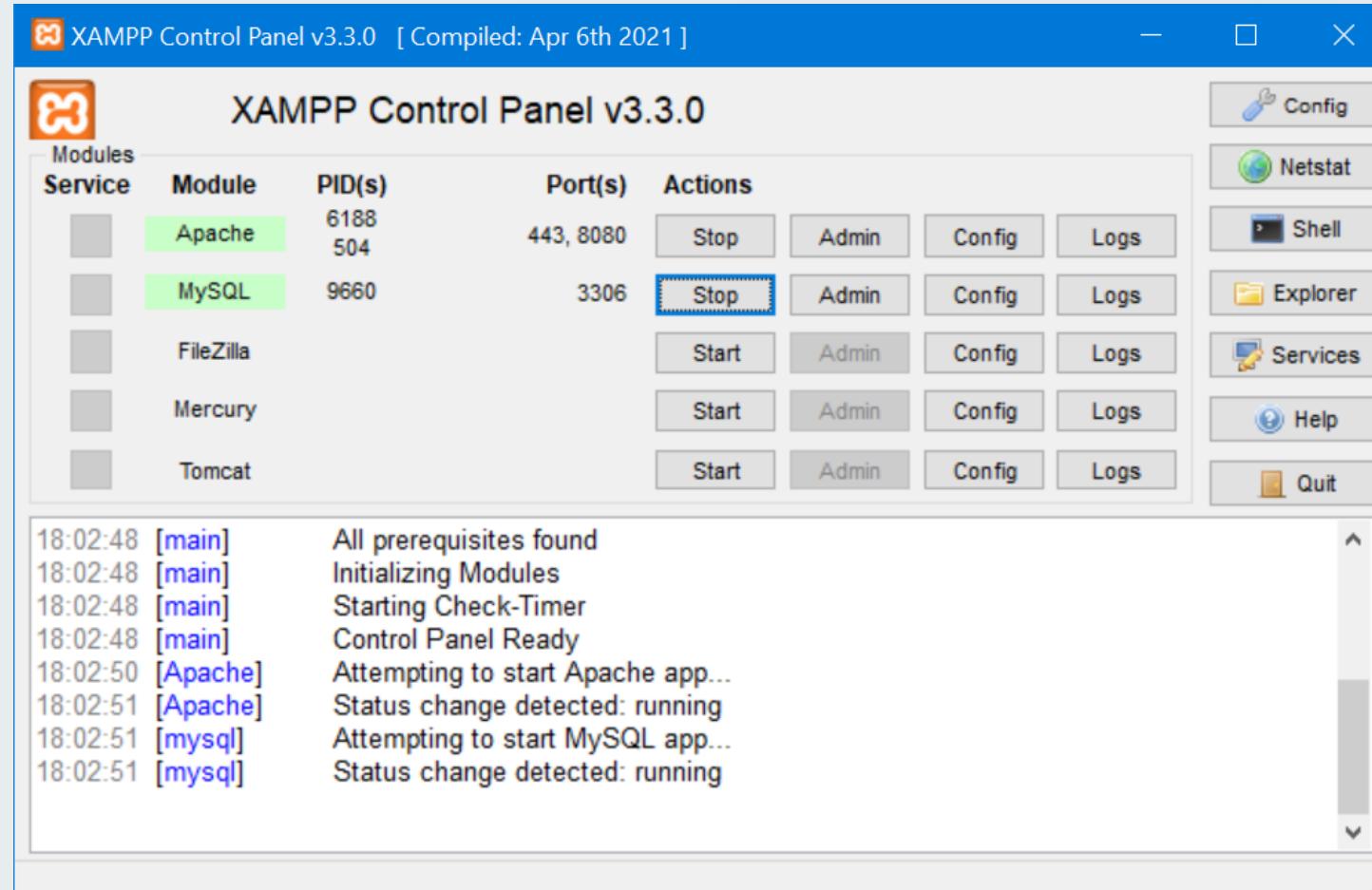
Xóa khách hàng

# Giới thiệu về hệ quản trị CSDL MySQL

- ❑ **Cách cài đặt XAMPP**
- ❑ **Cách tạo CSDL trong PHP Myadmin: Cách tạo cơ sở dữ liệu, tạo bảng,nhập liệu**
- ❑ **Cách sử dụng Query builder trong PHP MyAdmin: Viết select,insert,update,delete**
- ❑ **Cách backup và restore CSDL Mysql**

# Cài đặt Xampp

❑ <https://www.apachefriends.org/download.html>



## ☐ Tạo CSDL, Tạo bảng, nhập liệu

The screenshot shows the phpMyAdmin interface for MySQL management. The left sidebar lists databases: csdlmau, csdltest, gschool, hello, information\_schema, javacrud, mysql, nhanvien, performance\_schema, phpmyadmin, qlsanphan, quanlysinhvien, and student. The 'nhomkhachhang' table under the 'csdlmau' database is selected.

The main panel displays the 'Table structure' tab for the 'nhomkhachhang' table. It has two columns: 'Ma' (Type: varchar(3), Collation: utf8mb4\_unicode\_ci) and 'Ten' (Type: varchar(30), Collation: utf8mb4\_unicode\_ci). Below the table structure, there are buttons for 'Check all', 'With selected:', and various actions like 'Browse', 'Change', 'Drop', 'Primary', 'Unique', 'Index', 'Spatial', and 'Fulltext'. A 'Print' button is also present.

The 'Indexes' section shows a single primary key index named 'PRIMARY' on the 'Ma' column. Below it, there is a form to 'Create an index on' the 'Ma' column.

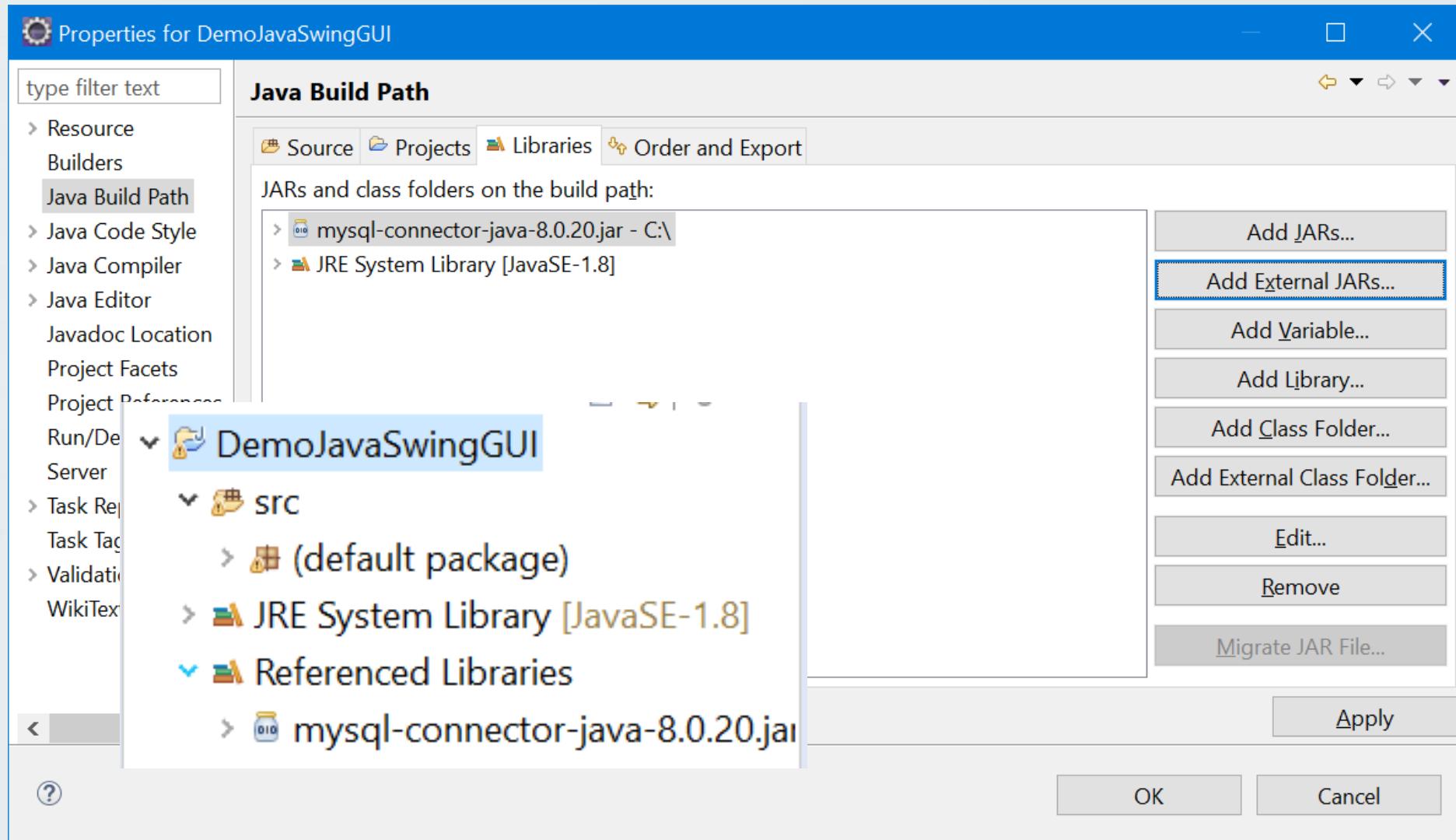
The bottom of the interface includes tabs for 'Structure', 'SQL', 'Search', 'Insert', 'Export', 'Import', 'Privileges', 'Operations', and 'Triggers'.

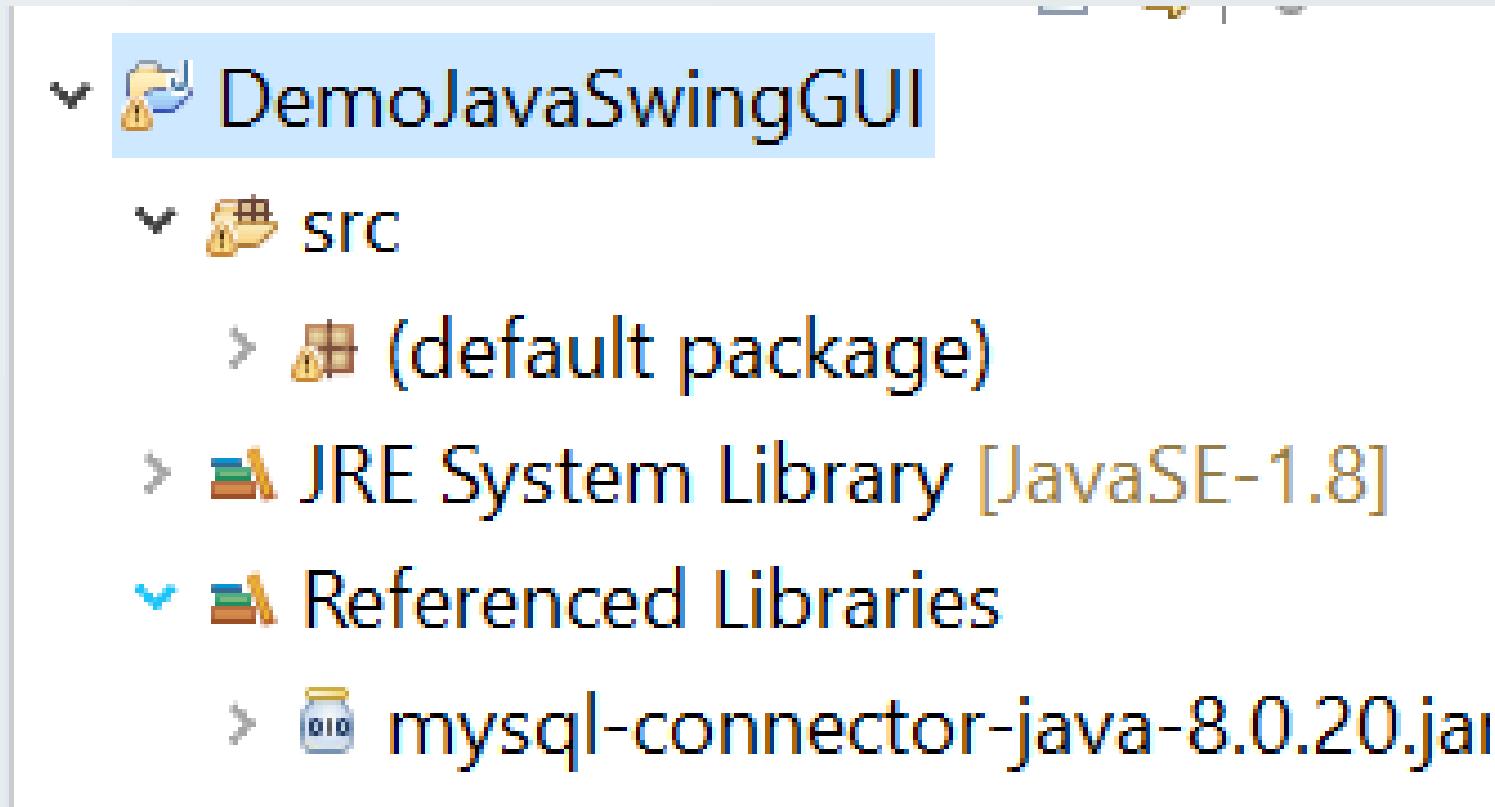
# Cách kết nối MySQL bằng MySql JDBC

❑ <https://dev.mysql.com/downloads/connector/j/>

 apache-maven-3.3.9-bin	20/06/2021 15:47	WinRAR ZIP archive	8,416 KB
 jaxb-api-2.3.0	18/06/2021 17:49	Executable Jar File	123 KB
 mysql-connector-java-8.0.20	16/06/2021 20:46	Executable Jar File	2,330 KB
 rs2xml	18/06/2021 12:44	Executable Jar File	168 KB

# Tải driver kết nối





```
public class MyConnect {  
    public static Connection getConnection() {  
        Connection conn = null;  
        String url = "jdbc:mysql://localhost/quanlysinhvien?characterEncoding=UTF-8";  
        String user = "root";  
        String password = "";  
        if(conn != null) {  
            return conn;  
        }else {  
            try {  
                Class.forName("com.mysql.cj.jdbc.Driver").newInstance();  
                conn = DriverManager.getConnection(url, user, password);  
                System.out.println("connect successful!");  
                System.out.println("Kết nối thành công!!!ye ye");  
            } catch (Exception er) {  
                System.out.println("connect failure!");  
                er.printStackTrace();  
            }  
        }  
        return conn;  
    }  
}
```

## Kiểm tra kết nối

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

```
<terminated> MyConnect [Java Application] C:\Program Files\Java  
connect successful!  
Kết nối CSDL thành công!!!!
```

# Sử dụng **Statement** để truy vấn dữ liệu trong MySql

# Truy vấn dữ liệu trong MySQL – select

```
try {
    Statement statement=conn.createStatement();
    ResultSet result=statement.executeQuery
        ("select * from NguoiDung");
    while(result.next())
    {
        System.out.println(result.getString("HoTen"));
    }
} catch (Exception e) {
    e.printStackTrace();
}
```

# Truy vấn dữ liệu trong MySQL – select

ResultSet.CONCUR\_READ\_ONLY

ResultSet.CONCUR\_UPDATABLE

```
try {
    Statement
    statement=conn.createStatement(ResultSet.TYPE_SCROLL_INSE
NSITIVE,ResultSet.CONCUR_READ_ONLY);
    ResultSet result=
        statement.executeQuery("select * from NguoiDung");
    while(result.next())
    {
        System.out.println(result.getString("HoTen"));
    }
} catch (Exception e) {
    e.printStackTrace();
}
```

# Sử dụng **Statement** để thêm mới dữ liệu trong MySql

# Thêm mới dữ liệu – insert

```
try
{
    String sql="insert into sinhvien values(
        '"+txtMa.getText()+"','"+txtTen.getText()+"',
        "
        +txtTuoi.getText()+")";
    Statement statement=conn.createStatement();
    int x=statement.executeUpdate(sql);
    if(x>0)
    {
        JOptionPane.showMessageDialog(null, "Lưu OK");
    }
}
catch(Exception ex)
{
    ex.printStackTrace();
}
```

# Sử dụng **Statement** để thay đổi dữ liệu trong MySQL

# Thay đổi dữ liệu – update

```
try
{
    String sql="update sinhvien set
Ten='"+txtTen.getText()+"',Tuoi='"+txtTuoi.getText()+"'
where Ma='"+txtMa.getText()+"';
Statement statement=conn.createStatement();
int x=statement.executeUpdate(sql);
if(x>0)
{
    JOptionPane.showMessageDialog(null, "Cập nhật
OK");
}
catch(Exception ex)
{
    ex.printStackTrace();
}
```

# Sử dụng **Statement** để xóa bỏ dữ liệu trong MySQL

```
try
{
    String sql="delete from sinhvien where
Ma='"+txtMa.getText()+"'";
    Statement statement=conn.createStatement();
    int x=statement.executeUpdate(sql);
    if(x>0)
    {
        JOptionPane.showMessageDialog(null, "Xóa
OK");
    }
}
catch(Exception ex)
{
    ex.printStackTrace();
}
```

# Sử dụng PreparedStatement để truy vấn dữ liệu trong MySql

```
try {
    PreparedStatement statement=conn.prepareStatement
        ("select * from sinhvien where tuoi=?",
        ResultSet.TYPE_SCROLL_INSENSITIVE,
        ResultSet.CONCUR_READ_ONLY);
    statement.setInt(1, 19);

    ResultSet result=statement.executeQuery();

    while(result.next())
    {
        System.out.println(result.getString("HoTen"));
    }

} catch (Exception e) {
    e.printStackTrace();
}
```



# Stop!

