

MÔN HỌC: CHUYÊN ĐỀ JAVA

Giảng Viên: ThS. Giang Hào Côn



odaz2a5

22DTH1A - CDJava

LT: CT4 - TH: CT7

 Sao chép đường liên kết mời



BÀI 1: Lập trình giao diện với Swing (3 tiết)

BÀI 2: Lớp Jframe, Jpanel (6 tiết)

BÀI 3: Các loại Layout Manager thường dùng (6 tiết)

BÀI 4: Các loại Layout Manager thường dùng (tt) (3 tiết)

BÀI 5: Lập trình CSDL với SQL server (6 tiết)

BÀI 6: Các thao tác trên CSDL (6 tiết)

■ 03 tín chỉ

- Lý thuyết: 30 tiết (2 tín chỉ)
- Thực hành: 30 tiết (1 tín chỉ)
- Thời gian tự học: 90 tiết

Đánh giá kết quả học tập

- **Chuyên cần:** cấm thi đối với sinh viên nghỉ > 1 buổi
- **Đánh giá kết quả:**
 - Kiểm tra thường xuyên: 20%
 - Kiểm tra giữa kỳ: 20%
 - Thực hành: 10%
 - Thi kết thúc môn/đồ án: 50%

■ Tài liệu chính

- [1] Bài giảng chuyên đề Java, khoa CNTT, trường ĐH Nguyễn Tất Thành
- [2] Paul Deitel, Harvey Deitel, 2017, Java: How to program, Early Objects, 11th edition, Pearson

■ Tài liệu tham khảo

- [1] Reges, Stuart, 2018, Building Java Programs, Pearson
- [2] Mark Lassoﬀ, 2017, Java Programming for Beginners: Learn the fundamentals of programming with Java, Packt Publishing

Kiến thức yêu cầu

- Kiến thức về thuật toán và logic
- Lập trình hướng đối tượng Java (cơ bản)
- Kiến thức về cơ sở dữ liệu
- Máy vi tính có cấu hình tốt

Bài 01: LẬP TRÌNH GIAO DIỆN VỚI SWING

Giảng Viên: ThS. Giang Hào Côn

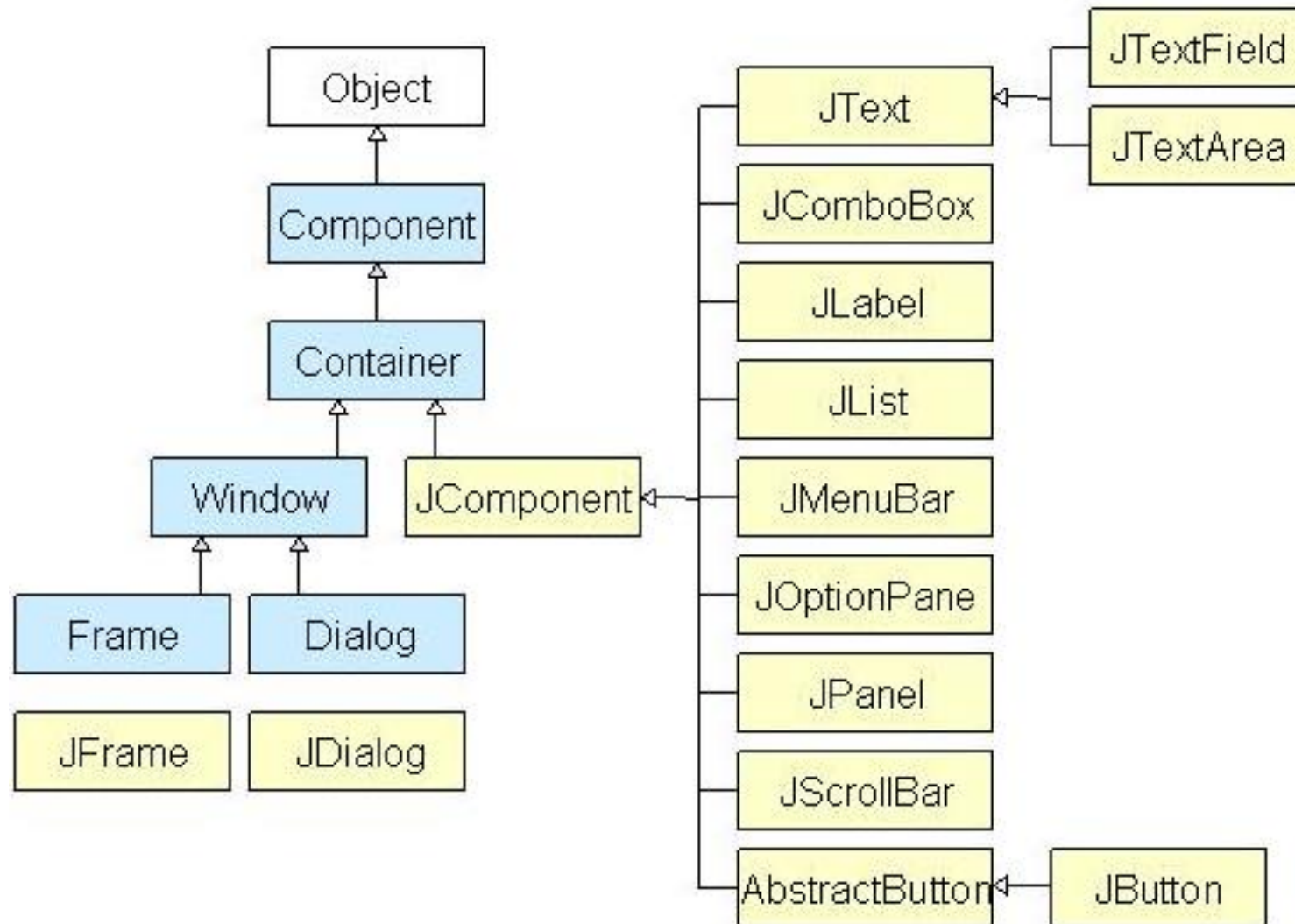
Giới thiệu gói Swing

- **Swing** là thư viện các đối tượng để lập trình giao diện đồ hoạ trong Java. Trước đây thư viện **AWT** là thư viện tiêu chuẩn cho lập trình giao diện, sau này **Swing** được phát triển kế thừa một số lớp của AWT, hoạt động nhẹ hơn và độc lập với nền tảng thiết bị, và bổ sung thêm nhiều lớp hiển thị mạnh mẽ hơn.
- Mỗi thành phần trong **Swing** được gọi là **component**. Component được chia làm 2 loại:
 - **Loại khung chứa**
 - **Loại hiển thị**

Điểm khác nhau giữa Java Swing và Java AWT

Java AWT	Java Swing
Các thành phần AWT là phụ thuộc nền tảng	Các thành phần Java Swing là độc lập nền tảng
Các thành phần AWT là nặng	Các thành phần Swing là gọn nhẹ
AWT cung cấp ít thành phần hơn Swing	Swing cung cấp các thành phần mạnh mẽ hơn như table, list, scrollpanes, colorchooser, tabbedpane ...
AWT không theo sau MVC (Model View Controller), ở đây model biểu diễn dữ liệu, view biểu diễn sự trình bày và controller hoạt động như một Interface giữa model và view	Swing theo sau MVC

Sơ đồ phân cấp Java Swing



Tính năng của Java Swing

- 1) **Trọng lượng nhẹ** - Các thành phần Swing độc lập với API của hệ điều hành gốc do các điều khiển API Swing được kết xuất chủ yếu bằng cách sử dụng mã Java thuần túy thay vì các cuộc gọi hệ điều hành cơ bản.
- 2) **Rich Controls** - Swing cung cấp một bộ điều khiển nâng cao phong phú như Tree, TabbedPane, thanh trượt, colorpicker và điều khiển bảng.
- 3) **Tùy biến cao** - các điều khiển xoay có thể được tùy chỉnh theo một cách rất dễ dàng và độc lập với biểu diễn bên trong.
- 4) **Pluggable look-and-feel** - Swing dựa nhìn GUI Application và có thể thay đổi thời gian chạy, dựa trên các giá trị có sẵn

Container Class

Container class (lớp vùng chứa) là các class có thể chứa các component khác trên đó. Vì vậy, để tạo Java GUI, chúng ta cần ít nhất một container object (đối tượng vùng chứa). Java Swing có 3 loại container.

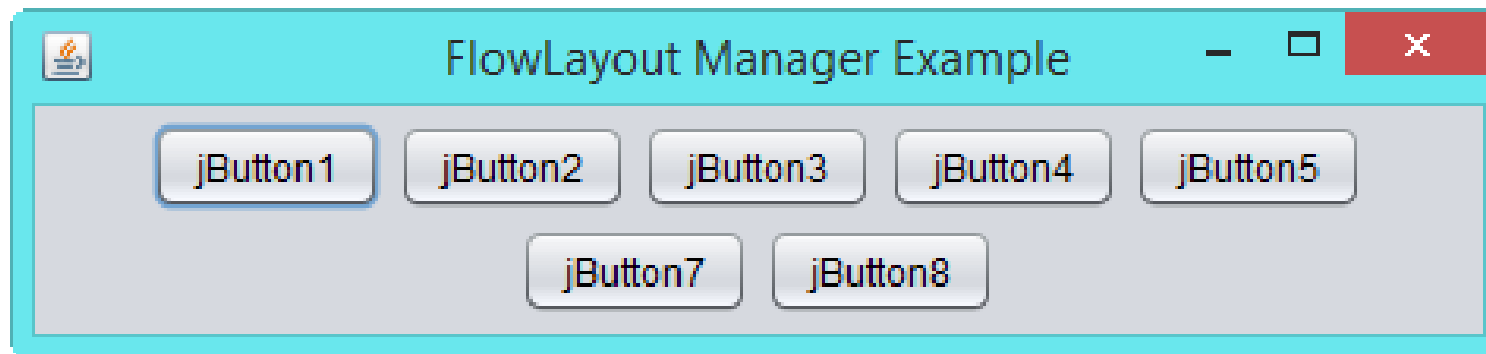
- **Panel (bảng điều khiển)**: là một container thuần túy và thực chất không phải là một window (cửa sổ). Mục đích duy nhất của Panel là tổ chức các component trên một window.
- **Frame (khung)**: là một window hoàn chỉnh có đầy đủ title (tiêu đề) và các icon (biểu tượng).
- **Dialog (hộp thoại)**: có thể được coi như một pop-up window sẽ bật ra khi một tin nhắn cần được hiển thị. Nó không phải là window hoàn chỉnh như Frame.

GUI trong Java là gì ?

- GUI (Graphical User Interface) trong Java là một trình tạo trải nghiệm trực quan cho các ứng dụng Java. Nó chủ yếu được làm bằng các thành phần đồ họa (graphical component) như button (nút), label (nhãn), window (cửa sổ), v.v. mà qua đó người dùng có thể tương tác với ứng dụng.
- GUI dễ sử dụng và đóng một vai trò quan trọng để xây dựng giao diện cho các ứng dụng Java.

Layout Managers (Quản lý bố cục)

- Quản lý bố cục trong java (**Layout Managers**) có nhiệm vụ qui định vị trí và kích thước của các thành phần trong giao diện đồ hoạ (GUI). Tất cả các container có một bộ quản lý bố cục mặc định, ví dụ JFrame mặc định là BorderLayout.
- Khi container được thay đổi kích cỡ, quản lý bố cục có nhiệm vụ tính toán vị trí mới và kích thước của các thành phần theo yêu cầu.



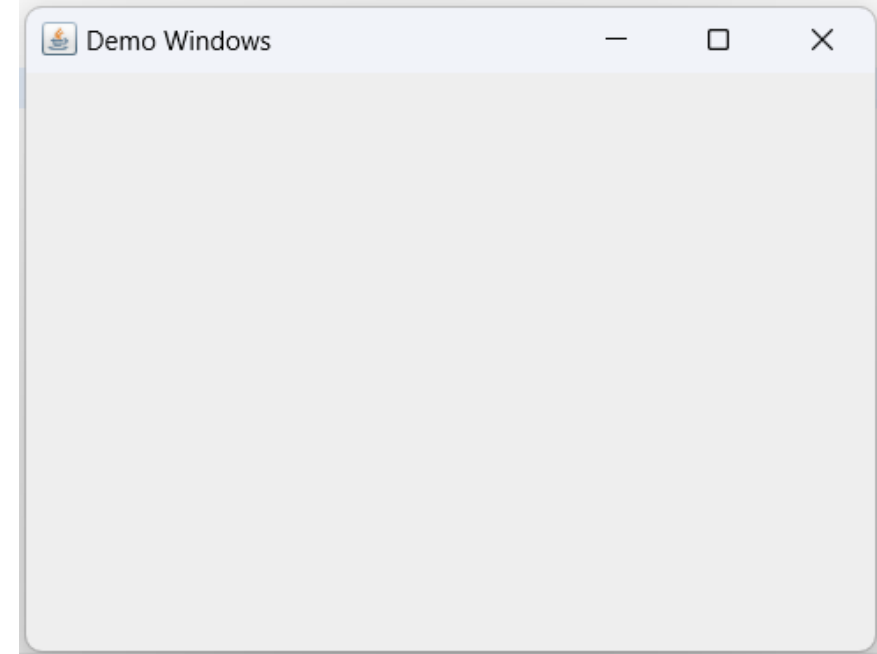
Ví dụ: Tạo cửa sổ giao diện

```
import javax.swing.JFrame;

public class DemoWindow {

    public static void main(String[] args)
    {
        JFrame frame = new JFrame("Demo Windows");
        frame.setDefaultCloseOperation(frame.EXIT_ON_CLOSE);
        frame.setSize(400, 300);
        frame.setLocationRelativeTo(null);
        frame.setVisible(true);

    }
}
```



Demo

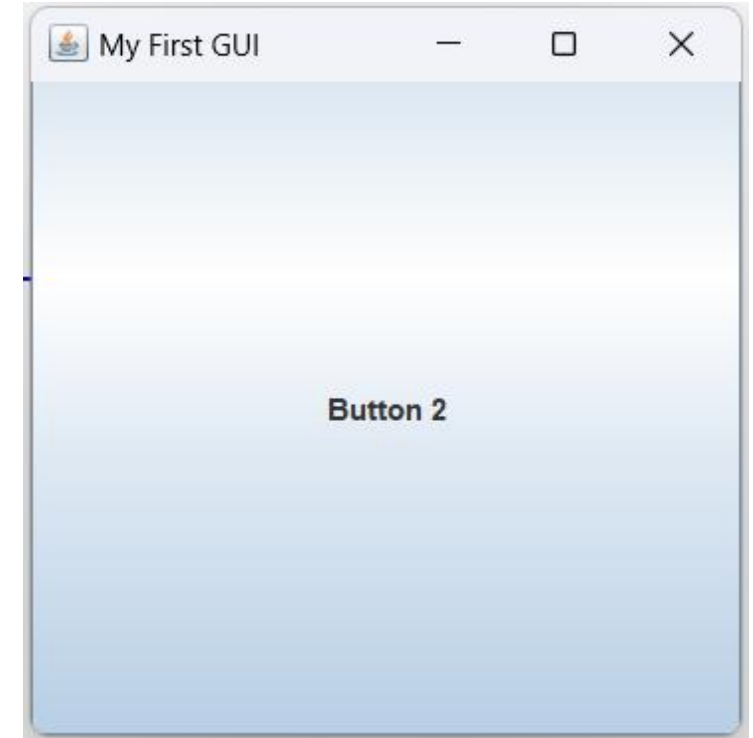
Ví dụ 01: tạo 1 nút

```
import javax.swing.JButton;  
import javax.swing.JFrame;  
  
public class Vidu01 {  
    public static void main(String args[]) {  
  
        JFrame frame = new JFrame("My First GUI");  
  
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
  
        frame.setSize(300,300);  
  
        JButton button = new JButton("Press");  
  
        frame.add(button); // Adds Button to content pane of frame  
  
        frame.setVisible(true);  
  
    }  
}
```



Ví dụ 02: tạo 2 nút vào frame

```
import javax.swing.JButton;  
import javax.swing.JFrame;  
  
public class Vidu01 {  
    public static void main(String args[]){  
  
        JFrame frame = new JFrame("My First GUI");  
  
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
  
        frame.setSize(300,300);  
  
        JButton button1 = new JButton("Button 1");  
  
        JButton button2 = new JButton("Button 2");  
  
        frame.getContentPane().add(button1);  
  
        frame.getContentPane().add(button2);  
  
        frame.setVisible(true);  
    }  
}
```

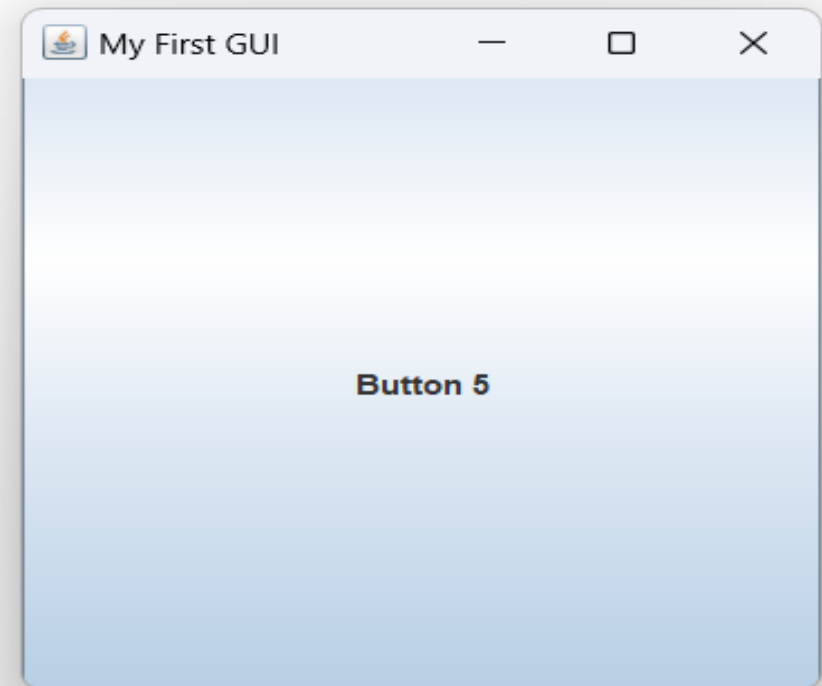


Nhận xét ?

Ví dụ 03: tạo 5 nút vào frame

```
import javax.swing.JButton;  
import javax.swing.JFrame;  
  
public class Vidu01 {  
    public static void main(String args[]){  
  
        JFrame frame = new JFrame("My First GUI");  
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
        frame.setSize(300,300);  
  
        JButton button1 = new JButton("Button 1");  
        JButton button2 = new JButton("Button 2");  
        JButton button3 = new JButton("Button 3");  
        JButton button4 = new JButton("Button 4");  
        JButton button5 = new JButton("Button 5");  
  
        frame.add(button1);  
        frame.add(button2);  
        frame.add(button3);  
        frame.add(button4);  
        frame.add(button5);  
  
        frame.setVisible(true);  
    }  
}
```

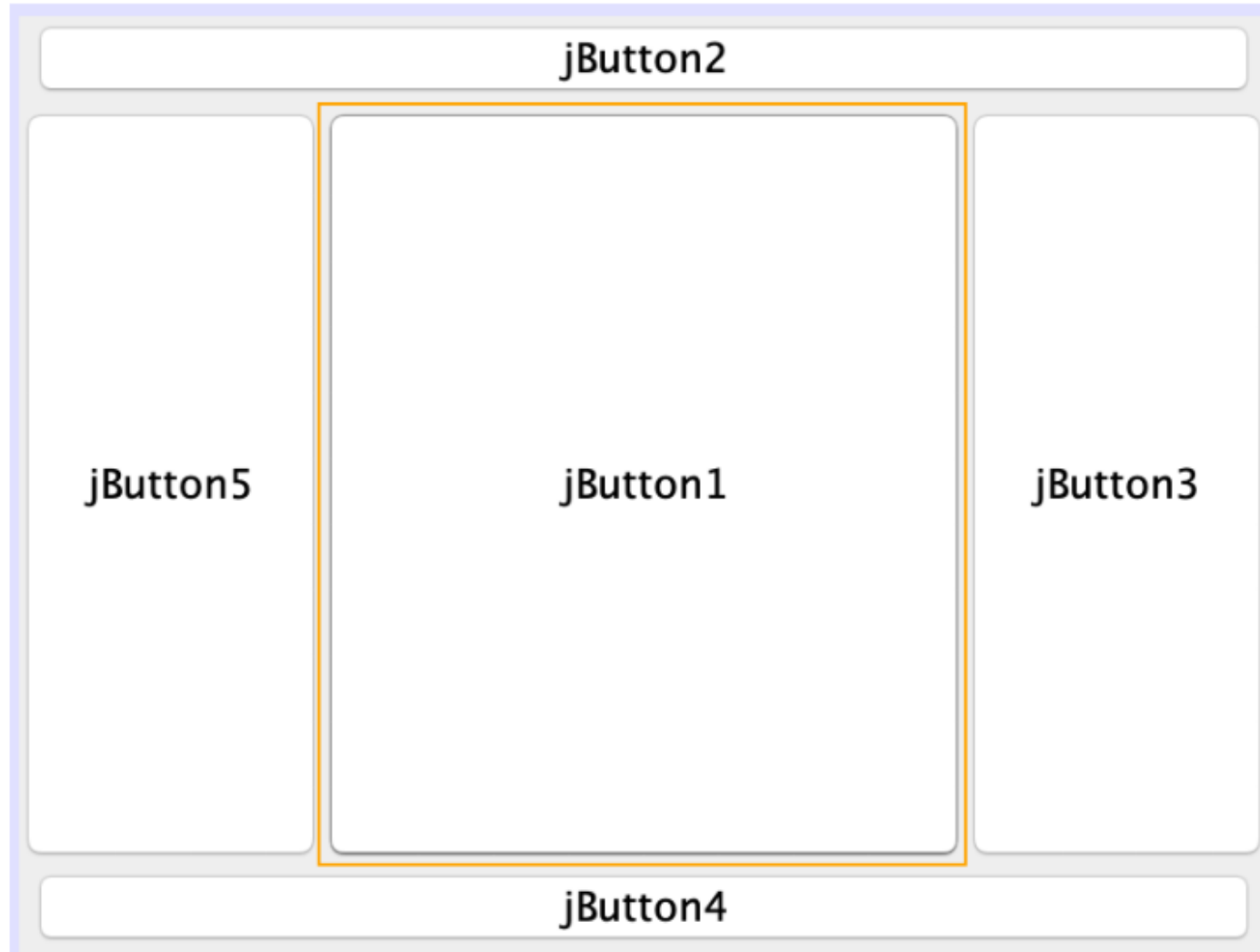
Nhận xét ?



Layout trong Java Swing

- **Layout** có tác dụng tự động bố trí các **component** trên container.
- Nói cách khác, chúng ta đặt các thành phần tại một vị trí cụ thể bên trong Container. Tác vụ bố trí này được thực hiện tự động bởi **LayoutManager**.
- **LayoutManager** được liên kết với mỗi đối tượng Container. Mỗi **LayoutManager** là một đối tượng của lớp mà triển khai **LayoutManager Interface**.

Border Layout



Ví dụ: sử dụng BorderLayout

```
import javax.swing.JButton;
import javax.swing.JFrame;

public class Vidu01 {
    public static void main(String args[]) {

        JFrame frame = new JFrame("My First GUI");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(300,300);

        JButton button1 = new JButton("Button 1");
        JButton button2 = new JButton("Button 2");
        JButton button3 = new JButton("Button 3");
        JButton button4 = new JButton("Button 4");
        JButton button5 = new JButton("Button 5");

        frame.add(button1, BorderLayout.NORTH);
        frame.add(button2, BorderLayout.EAST);
        frame.add(button3, BorderLayout.SOUTH);
        frame.add(button4, BorderLayout.WEST);
        frame.add(button5, BorderLayout.CENTER);

        frame.setVisible(true);
    }
}
```

Nhận xét ?



Ví dụ: sử dụng BorderLayout có khoảng cách

```
import java.awt.BorderLayout;  
import javax.swing.JButton;  
import javax.swing.JFrame;  
  
public class Vidu01 {  
    public static void main(String args[]) {  
  
        JFrame frame = new JFrame("My First GUI");  
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
        frame.setSize(300,300);  
  
        JButton button1 = new JButton("Button 1");  
        JButton button2 = new JButton("Button 2");  
        JButton button3 = new JButton("Button 3");  
        JButton button4 = new JButton("Button 4");  
        JButton button5 = new JButton("Button 5");  
  
        frame.setLayout(new BorderLayout(20,15));  
        frame.add(button1, BorderLayout.NORTH);  
        frame.add(button2, BorderLayout.EAST);  
        frame.add(button3, BorderLayout.SOUTH);  
        frame.add(button4, BorderLayout.WEST);  
        frame.add(button5, BorderLayout.CENTER);  
  
        frame.setVisible(true);  
    }  
}
```

Nhận xét ?



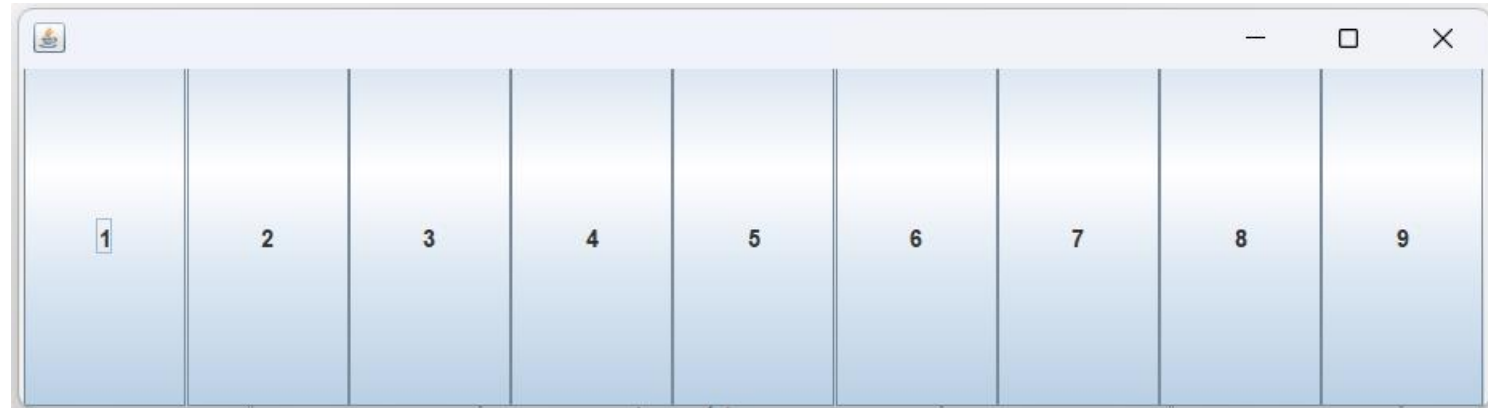
GridLayout

GridLayout được sử dụng để bố trí các thành phần trong một lưới hình chữ nhật, lưới này đã được chia thành các hình chữ nhật có kích thước bằng nhau và một thành phần được đặt trong mỗi hình chữ nhật. Chúng ta có thể khởi tạo GridLayout thông qua các constructor sau:

- **GridLayout()** – Tạo GridLayout với duy nhất một dòng và một cột
- **GridLayout(int row, int col)** – Tạo GridLayout với row dòng và col cột.
- **GridLayout(int row, int col, int hgap, int vgap)** – Tạo GridLayout với row dòng và col cột, các phần tử trên cùng một dòng cách nhau hgap và trên cùng một cột cách nhau vgap.

Ví dụ: sử dụng GridLayout không tham số

```
Vidu02() {  
    frameObj = new JFrame();  
  
    JButton btn1 = new JButton("1");  
    JButton btn2 = new JButton("2");  
    JButton btn3 = new JButton("3");  
    JButton btn4 = new JButton("4");  
    JButton btn5 = new JButton("5");  
    JButton btn6 = new JButton("6");  
    JButton btn7 = new JButton("7");  
    JButton btn8 = new JButton("8");  
    JButton btn9 = new JButton("9");  
  
    frameObj.add(btn1); frameObj.add(btn2); frameObj.add(btn3);  
    frameObj.add(btn4); frameObj.add(btn5); frameObj.add(btn6);  
    frameObj.add(btn7); frameObj.add(btn8); frameObj.add(btn9);  
  
    frameObj.setLayout(new GridLayout());  
  
    frameObj.setSize(300, 300);  
    frameObj.setVisible(true);  
}
```



Ví dụ: sử dụng GridLayout 3 dòng, 3 cột

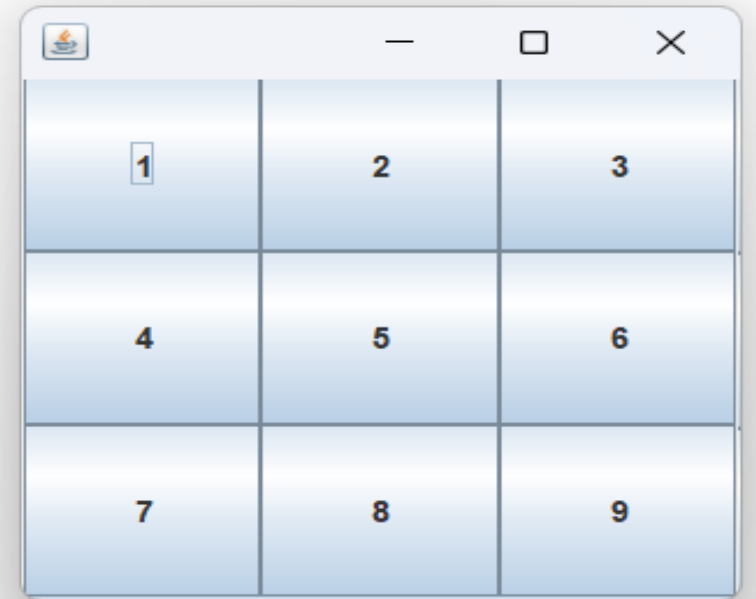
```
// constructor
Vidu02() {
    frameObj = new JFrame();

    JButton btn1 = new JButton("1");
    JButton btn2 = new JButton("2");
    JButton btn3 = new JButton("3");
    JButton btn4 = new JButton("4");
    JButton btn5 = new JButton("5");
    JButton btn6 = new JButton("6");
    JButton btn7 = new JButton("7");
    JButton btn8 = new JButton("8");
    JButton btn9 = new JButton("9");

    frameObj.add(btn1); frameObj.add(btn2); frameObj.add(btn3);
    frameObj.add(btn4); frameObj.add(btn5); frameObj.add(btn6);
    frameObj.add(btn7); frameObj.add(btn8); frameObj.add(btn9);

    frameObj.setLayout(new GridLayout(3, 3));

    frameObj.setSize(300, 300);
    frameObj.setVisible(true);
}
```



Ví dụ: sử dụng GridLayout có khoảng cách

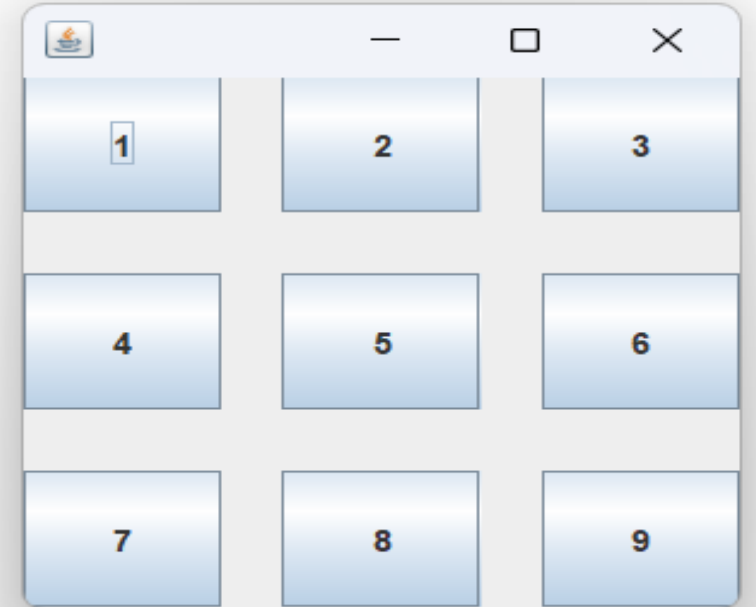
```
// constructor
Vidu02() {
    frameObj = new JFrame();

    JButton btn1 = new JButton("1");
    JButton btn2 = new JButton("2");
    JButton btn3 = new JButton("3");
    JButton btn4 = new JButton("4");
    JButton btn5 = new JButton("5");
    JButton btn6 = new JButton("6");
    JButton btn7 = new JButton("7");
    JButton btn8 = new JButton("8");
    JButton btn9 = new JButton("9");

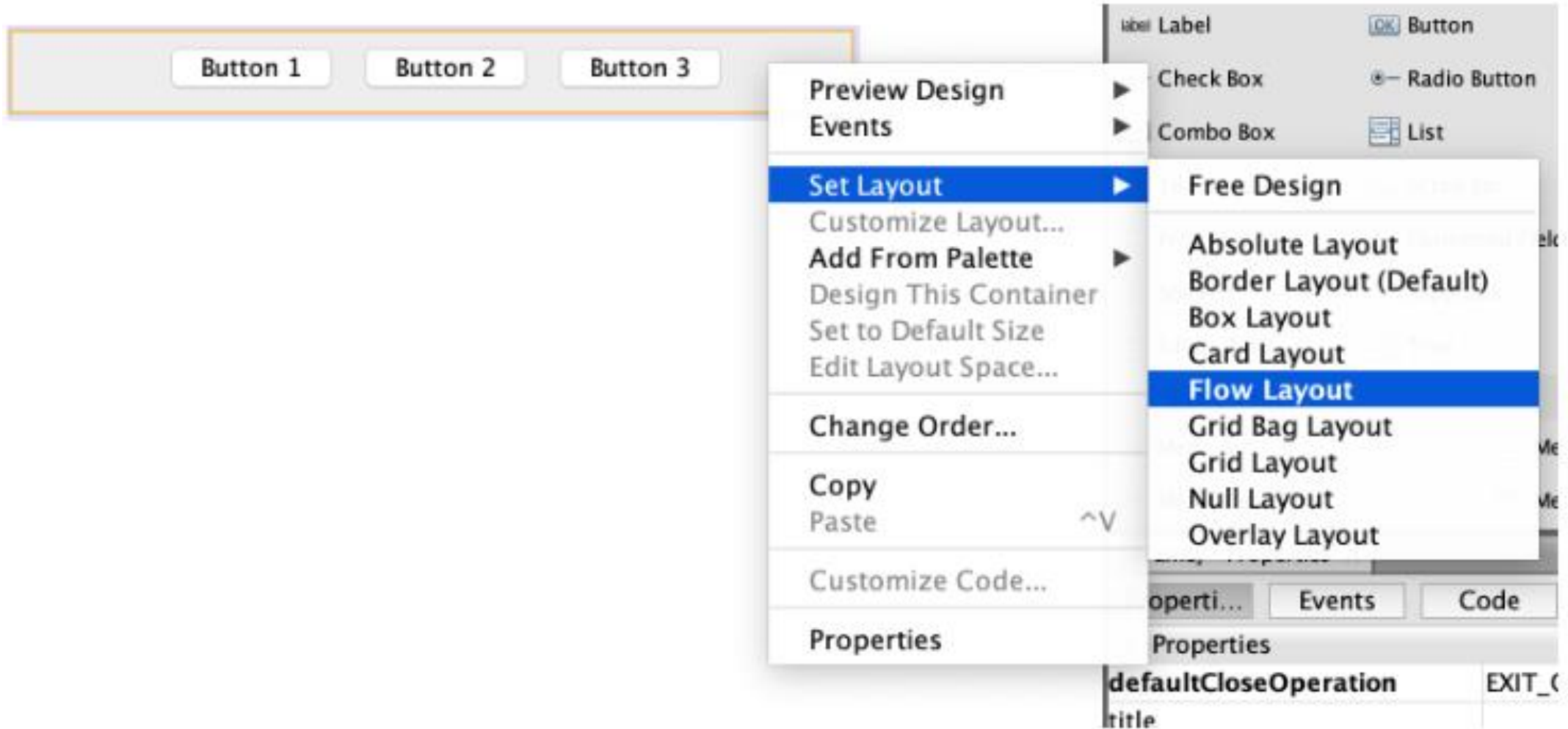
    frameObj.add(btn1); frameObj.add(btn2); frameObj.add(btn3);
    frameObj.add(btn4); frameObj.add(btn5); frameObj.add(btn6);
    frameObj.add(btn7); frameObj.add(btn8); frameObj.add(btn9);

    frameObj.setLayout(new GridLayout(3, 3, 20, 25));

    frameObj.setSize(300, 300);
    frameObj.setVisible(true);
}
```



FlowLayout



Ví dụ: sử dụng FlowLayout không tham số

```
// constructor
Vidu02() {
    frameObj = new JFrame();

    JButton btn1 = new JButton("1");
    JButton btn2 = new JButton("2");
    JButton btn3 = new JButton("3");
    JButton btn4 = new JButton("4");
    JButton btn5 = new JButton("5");
    JButton btn6 = new JButton("6");
    JButton btn7 = new JButton("7");
    JButton btn8 = new JButton("8");
    JButton btn9 = new JButton("9");

    frameObj.add(btn1); frameObj.add(btn2); frameObj.add(btn3);
    frameObj.add(btn4); frameObj.add(btn5); frameObj.add(btn6);
    frameObj.add(btn7); frameObj.add(btn8); frameObj.add(btn9);

    frameObj.setLayout(new FlowLayout());

    frameObj.setSize(300, 300);
    frameObj.setVisible(true);
}
```



Ví dụ: sử dụng FlowLayout có tham số

```
// constructor
Vidu02() {
    frameObj = new JFrame();

    JButton b1=new JButton("1");
    JButton b2=new JButton("2");
    JButton b3=new JButton("3");
    JButton b4=new JButton("4");
    JButton b5=new JButton("5");

    // adding buttons to the frame
    frameObj.add(b1); frameObj.add(b2); frameObj.add(b3);
    frameObj.add(b4); frameObj.add(b5);

    // setting flow layout of right alignment
    frameObj.setLayout(new FlowLayout(FlowLayout.RIGHT));

    frameObj.setSize(300, 300);
    frameObj.setVisible(true);
}
```



Ví dụ: sử dụng FlowLayout có tham số

```
// constructor
Vidu02() {
    frameObj = new JFrame();

    JButton b1 = new JButton("1");
    JButton b2 = new JButton("2");
    JButton b3 = new JButton("3");
    JButton b4 = new JButton("4");
    JButton b5 = new JButton("5");
    JButton b6 = new JButton("6");
    JButton b7 = new JButton("7");
    JButton b8 = new JButton("8");
    JButton b9 = new JButton("9");
    JButton b10 = new JButton("10");

    frameObj.add(b1); frameObj.add(b2); frameObj.add(b3); frameObj.add(b4);
    frameObj.add(b5); frameObj.add(b6); frameObj.add(b7); frameObj.add(b8);
    frameObj.add(b9); frameObj.add(b10);

    frameObj.setLayout(new FlowLayout(FlowLayout.LEFT, 20, 25));

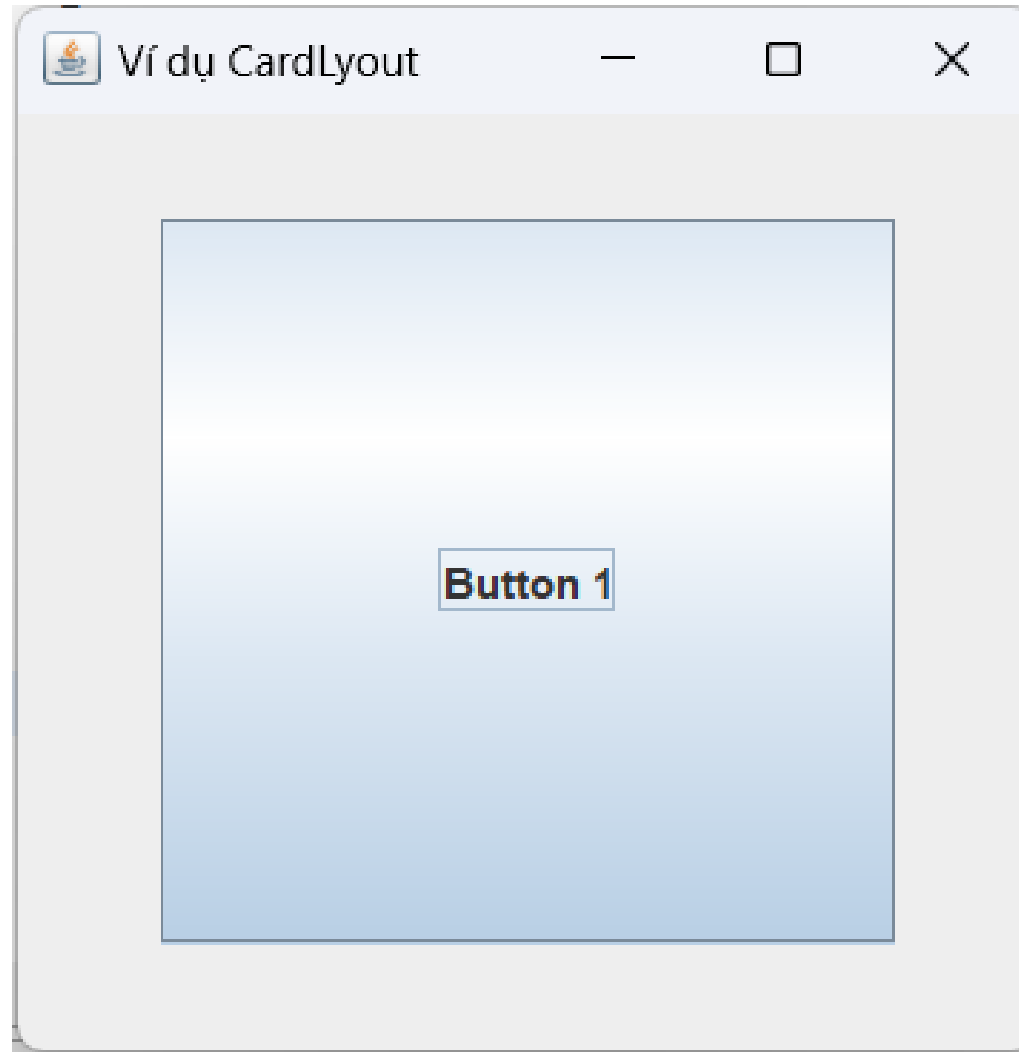
    frameObj.setSize(300, 300);
    frameObj.setVisible(true);
}
```



CardLayout

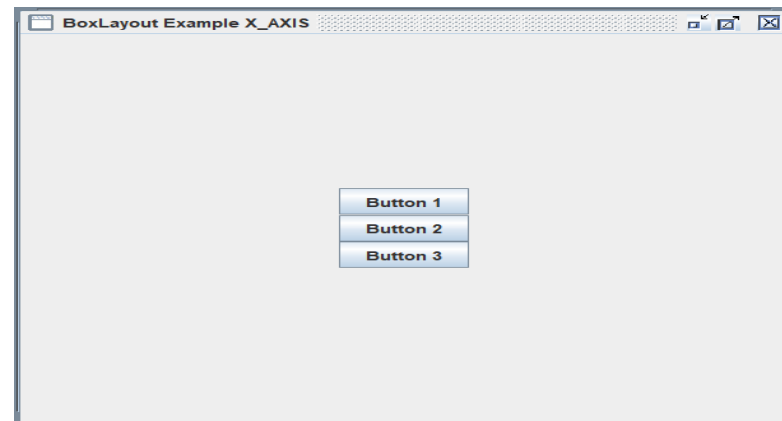
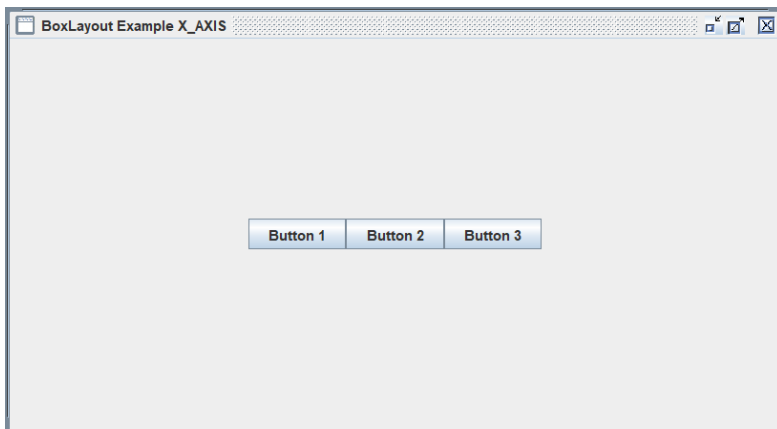
- **CardLayout** sắp xếp các thành phần con thành một Stack, chúng sẽ chồng lên nhau và tại mỗi thời điểm chúng ta chỉ có thể thấy một thành phần con duy nhất cho đến khi CardLayout chuyển sang thành phần con tiếp theo.
- Ví dụ tạo một JFrame với layout được chỉ định là CardLayout, chứa 3 Button sắp chồng lên nhau. Cứ mỗi khi click vào một Button thì chúng ta sẽ xử lý sự kiện để chuyển sang hiện thị Button tiếp theo và xoay vòng.

▪ Ví dụ:



BoxLayout

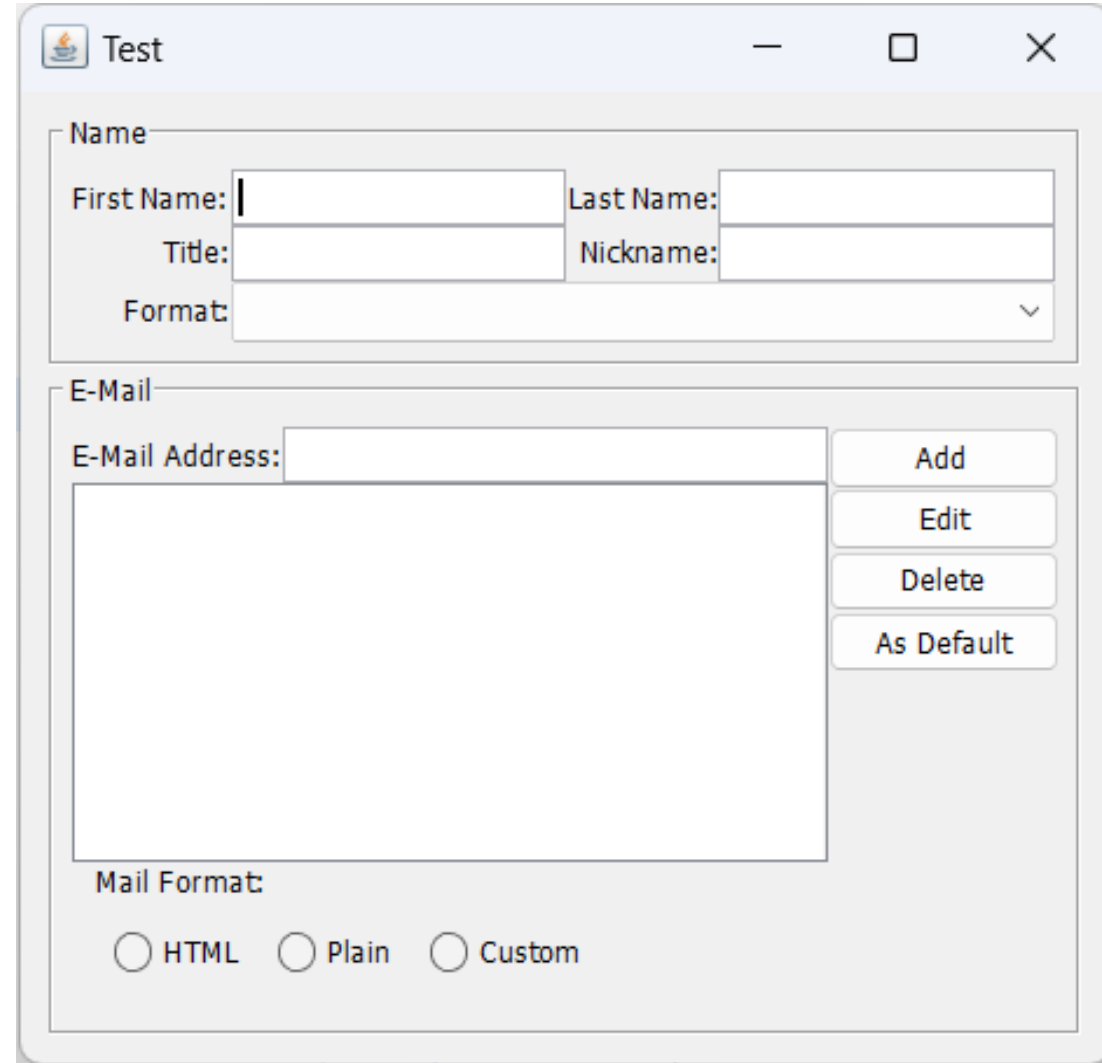
- **BoxLayout** được sử dụng để sắp xếp các thành phần con **theo chiều dọc** hoặc **theo chiều ngang**.
- Trong BoxLayout cung cấp các hằng số **X_AXIS**, **Y_AXIS** trong đó **X_AXIS** dùng để sắp xếp các thành phần con theo chiều ngang từ **trái sang phải**, còn **Y_AXIS** thì theo chiều dọc từ **trên xuống dưới**.



Các layout khác

- GridBagLayout
- GroupLayout
- SpringLayout
- ScrollPanelLayout

Sinh viên thảo luận
Và cho ví dụ



Test

Name

First Name: Last Name:

Title: Nickname:

Format:

E-Mail

E-Mail Address:

Add

Edit

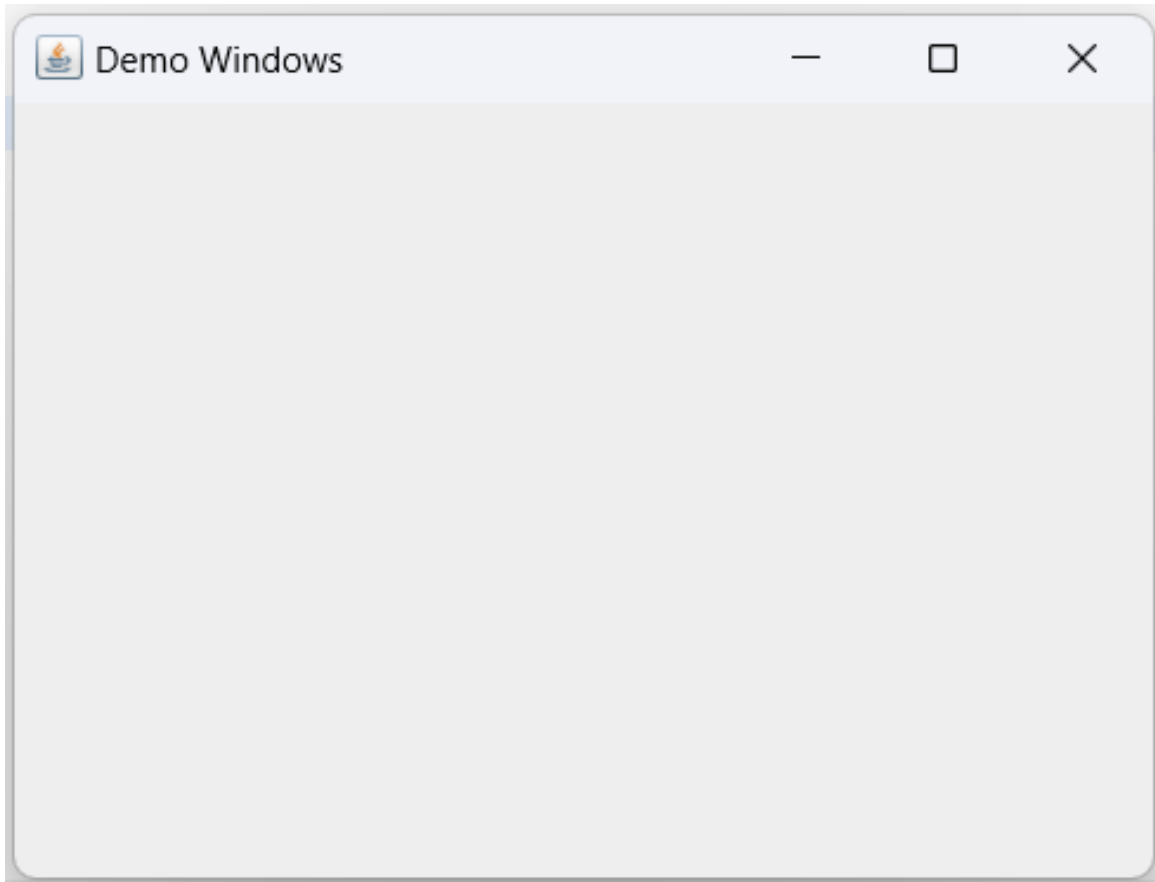
Delete

As Default

Mail Format:

☐ HTML ☐ Plain ☐ Custom

- Tạo cửa sổ như sau:



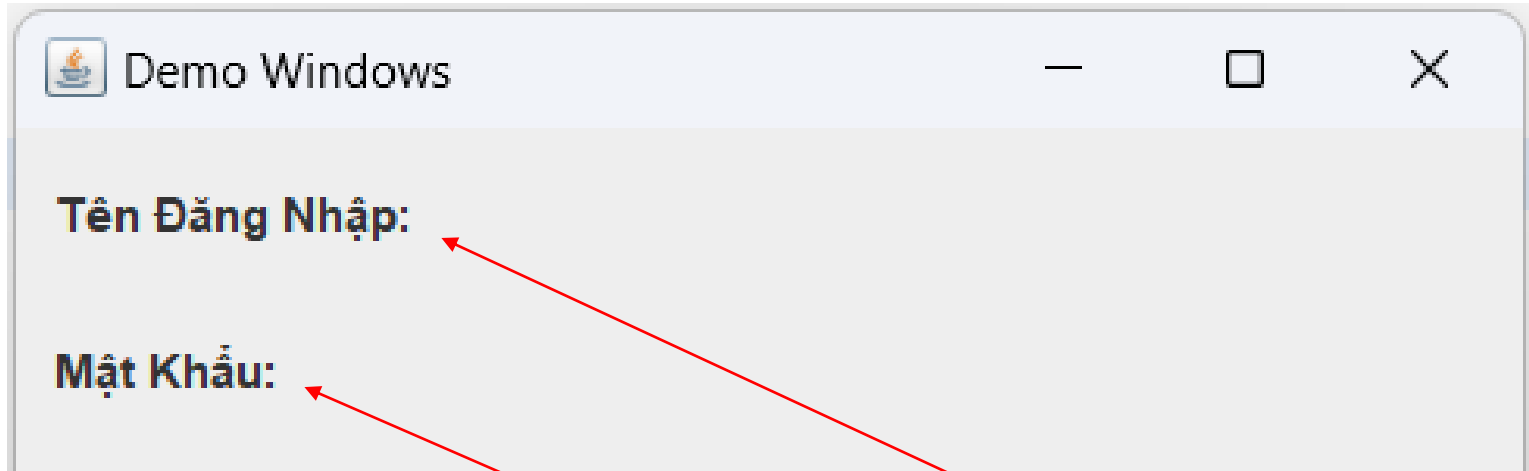
```
import javax.swing.JFrame;

public class DemoWindow {

    public static void main(String[] args)
    {
        JFrame frame = new JFrame("Demo Windows");
        frame.setDefaultCloseOperation(frame.EXIT_ON_CLOSE);
        frame.setSize(400, 300);
        frame.setLocationRelativeTo(null);
        frame.setVisible(true);
    }
}
```

Ví dụ không sử dụng layout và dùng setBound

- Tạo cửa sổ như sau: thêm 2 label vào cửa sổ



```
//Tạo Label  
JLabel ul1 = new JLabel("Tên Đăng Nhập: ");  
ul1.setBounds(10,10,100,25);  
frame.add(ul1);
```

```
JLabel ul2 = new JLabel("Mật Khẩu: ");  
ul2.setBounds(10,50,100,25);  
frame.add(ul2);
```

Ví dụ không sử dụng layout và dùng setBound

- Tạo cửa sổ như sau: thêm 2 Text Field vào cửa sổ

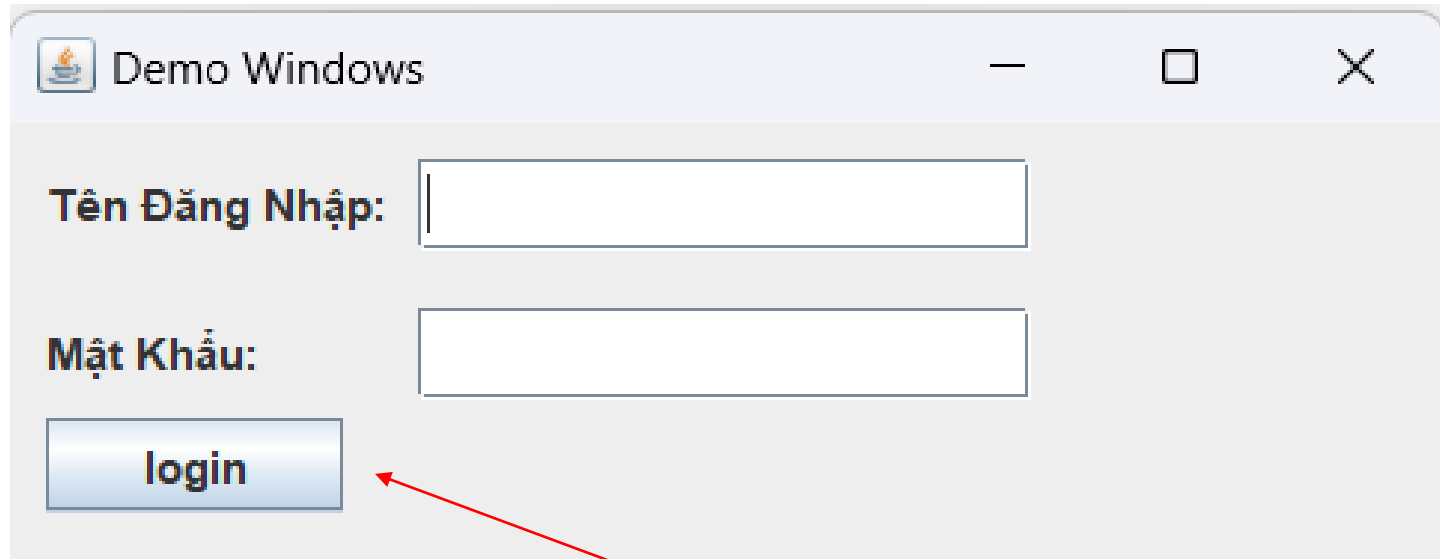


```
JTextField userText = new JTextField(20);  
userText.setBounds(110, 10, 165, 25);  
frame.add(userText);
```

```
JPasswordField passwordText = new JPasswordField(20);  
passwordText.setBounds(110, 50, 165, 25);  
frame.add(passwordText);
```

Ví dụ không sử dụng layout và dùng setBound

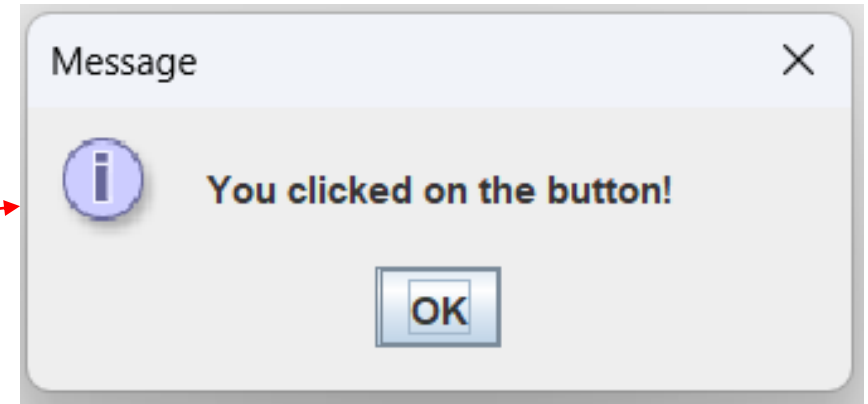
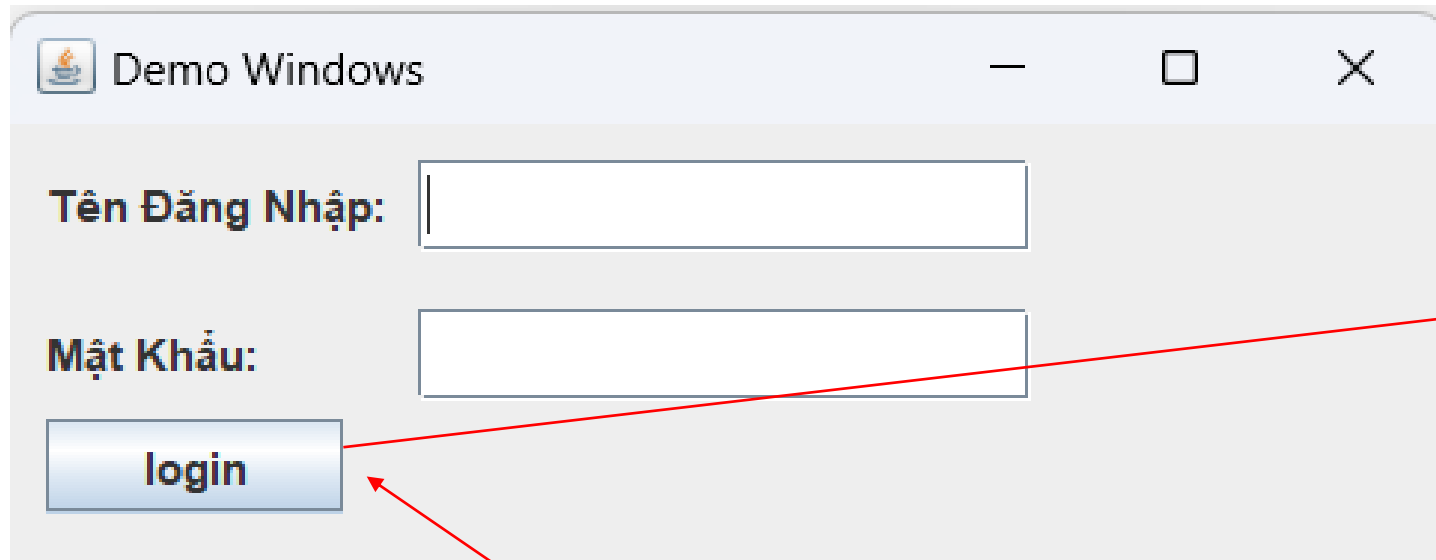
- Tạo cửa sổ như sau: thêm nút lệnh vào cửa sổ



```
JButton loginButton = new JButton("login");  
loginButton.setBounds(10, 80, 80, 25);  
frame.add(loginButton);
```

Ví dụ không sử dụng layout và dùng setBound

- Tạo cửa sổ như sau: thêm sự kiện click cho nút



```
loginButton.addActionListener(new ActionListener() {  
    @Override  
    public void actionPerformed(ActionEvent e) {  
        JOptionPane.showMessageDialog(null, "You clicked on the button!");  
    }  
});
```


Câu hỏi thảo luận

- 1) Gói Swing là gì ?
- 2) Trình bày điểm khác nhau giữa Java Swing và Java AWT ?
- 3) Trình bày Tính năng của Java Swing
- 4) Java LayoutManagers là gì ?