

Chương 6: Lập trình GUI



ThS. Trần Văn Hữu

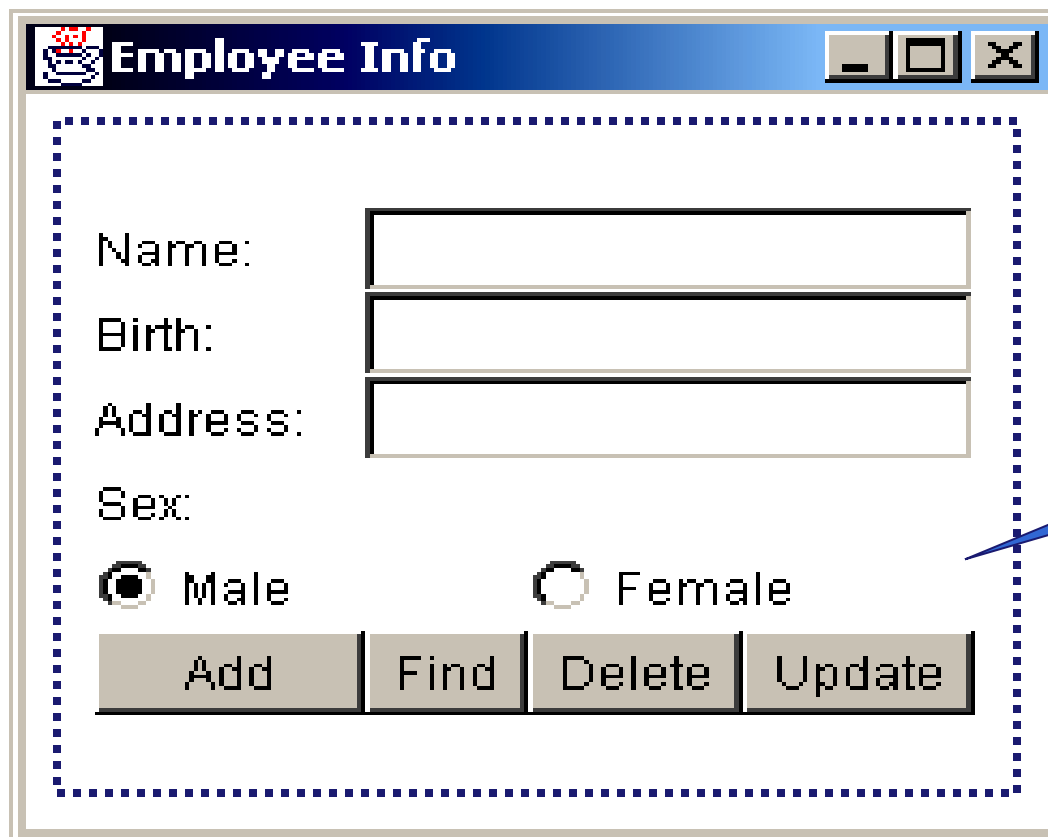


Nội dung

- ❖ Giới thiệu thiết kế GUI trong java
- ❖ Đối tượng khung chứa (Container)
- ❖ Các thành phần cơ bản (Component)
- ❖ Bộ quản lý trình bày (Layout Manager)
- ❖ Ví dụ minh họa

Giới thiệu (1)

- ❖ **GUI:** mô hình giao tiếp kiểu tương tác giữa ứng dụng và user dạng đồ họa.
- ❖ **GUI = Container + Components**



Employee Info

Name:

Birth:

Address:

Sex: ☒ Male ☐ Female

Add Find Delete Update

Container

Components



Giới thiệu về AWT (1)

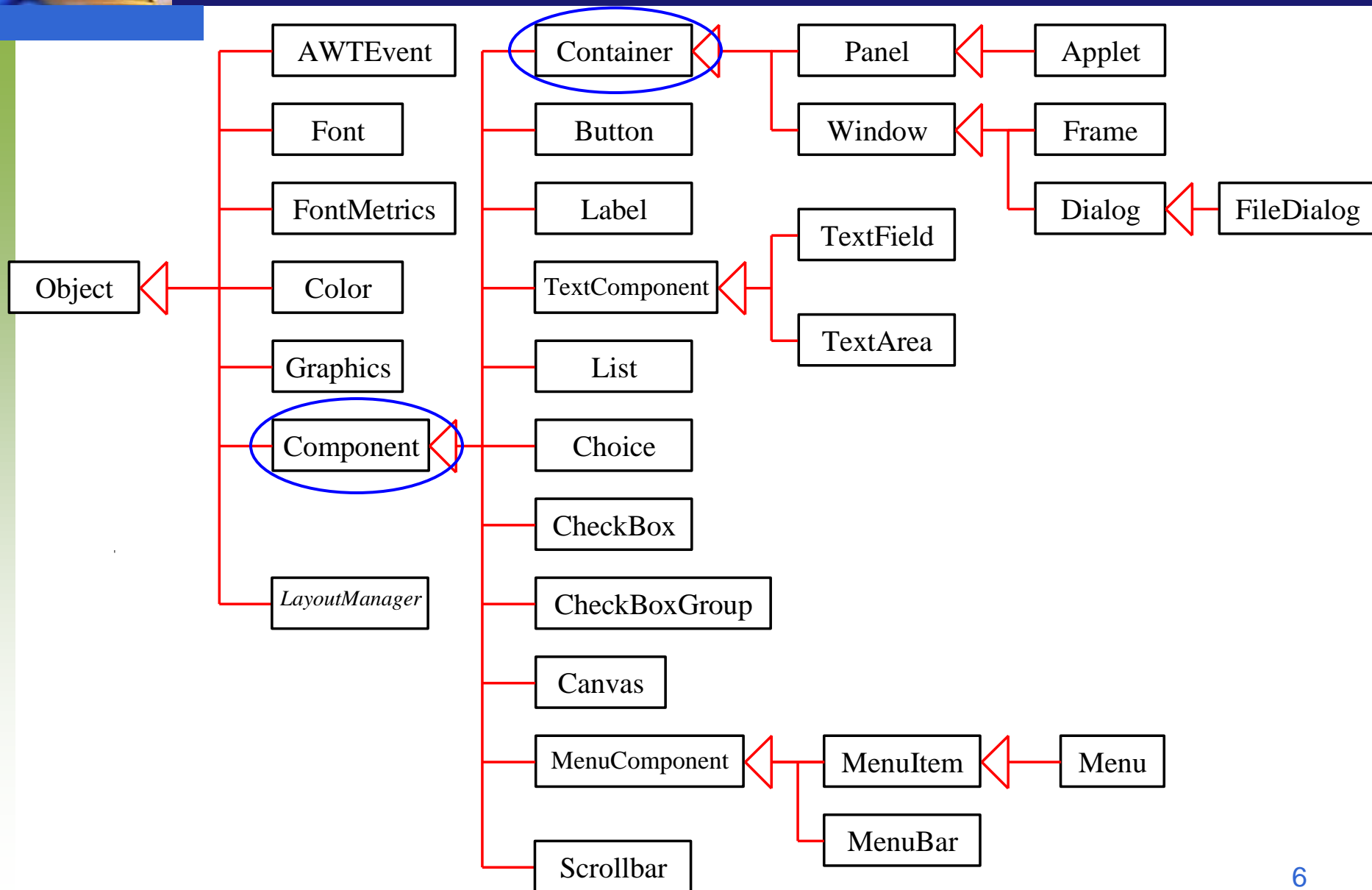
- ❖ AWT: Abstract Windowing Toolkit
- ❖ AWT là một bộ các lớp trong Java cho phép chúng ta tạo một GUI.
- ❖ Sử dụng:
 - `import java.awt.*;`
 - `import java.awt.event.*;`



Giới thiệu về AWT (2)

- ❖ AWT cung cấp các item khác nhau là:
 - Container
 - Component
 - Trình quản lý cách trình bày (Layout manager)
 - Đồ họa (Graphic) và các tính năng vẽ (draw)
 - Phong chữ (Font)
 - Sự kiện (Event)

Cấu trúc gói AWT





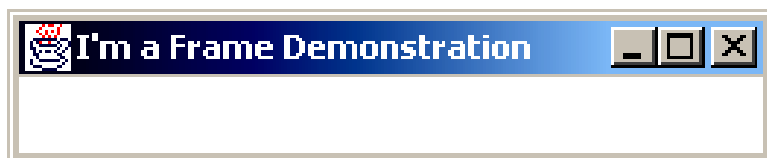
Container (1)

- ❖ Container có khả năng quản lý và nhóm các đối tượng khác lại.
 - Hai container được sử dụng phổ biến nhất là **Frame** và **Panel**.

Frame (1)

❖ Khung chứa Frame là một cửa sổ window

- Bao gồm một tiêu đề và một đường biên (border) như các ứng dụng windows thông thường khác.
- Thường được sử dụng để tạo cửa sổ chính của các ứng dụng.



java.awt

Class Frame

java.lang.Object

└ java.awt.Component

└ java.awt.Container

└ java.awt.Window

└ java.awt.Frame

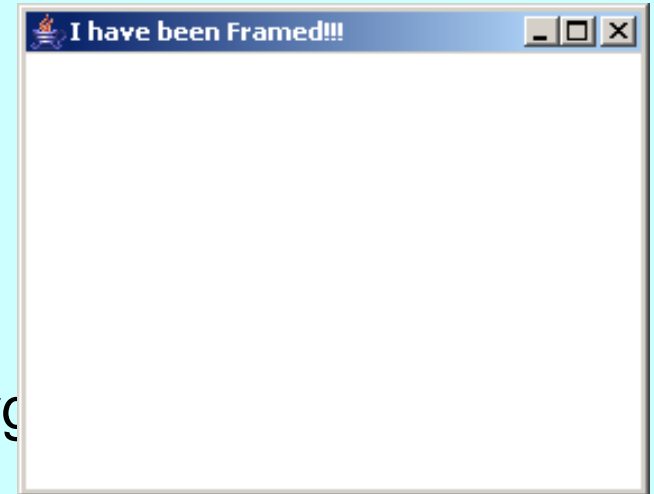


Frame (2)

- ❖ Frame có thể hoạt động như một container hay như một thành phần (component).
- ❖ Chúng ta có thể sử dụng một trong những **constructor** sau để tạo một frame:
 - **Frame()**
 - **Frame(String title)**
 - ...

Frame – Ví dụ

```
import java.awt.*;
class FrameDemo extends Frame{
    public FrameDemo(String title){
        super (title);
    }
    public static void main (String args[]) {
        FrameDemo ObjFr =
            new FrameDemo("I have been Framed!!!");
        ObjFr.setSize(500,500);
        ObjFr.setVisible(true);
    }
}
```



Panel (1)

- ❖ Đối tượng khung chứa đơn giản nhất, dùng để nhóm các đối tượng, thành phần con lại. Một Panel có thể chứa bên trong một Panel khác.
- ❖ Hàm khởi tạo
 - Panel ()
 - Panel (LayoutManager)

java.awt

Class Panel

java.lang.Object

└ java.awt.Component

└ java.awt.Container

└ java.awt.Panel



Panel (2)

- ❖ Panel không thể được nhìn thấy trực tiếp.
- ❖ Do đó, chúng ta cần thêm panel đến một frame.
- ❖ Sử dụng hai phương thức `setSize()` và `setVisible()` để thiết lập kích thước và hiển thị frame.

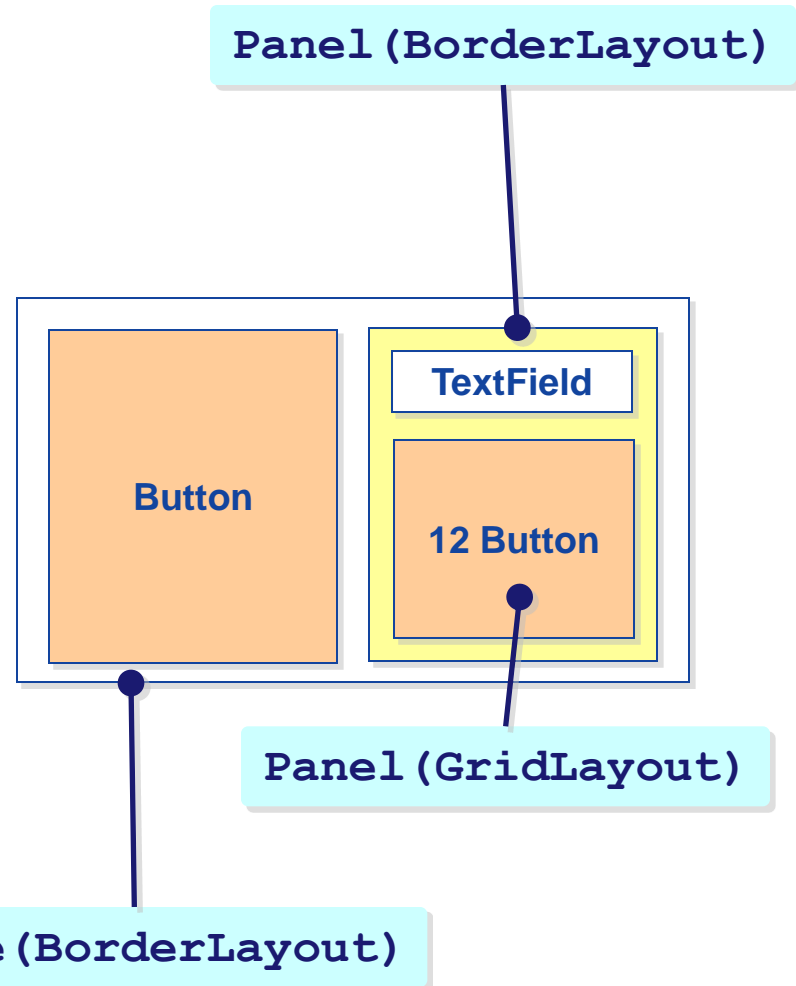
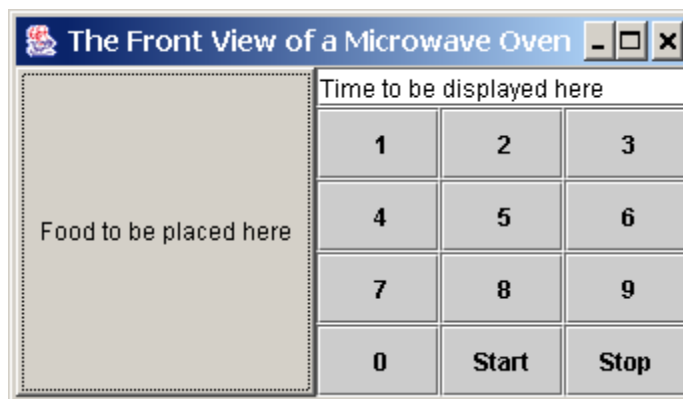
Panel – Ví dụ

```
import java.awt.*;
class Paneltest extends Panel{
    public static void main(String args[])
    {
        Paneltest p = new Paneltest();
        Frame f = new Frame("Testing a Panel");
        f.add(p);
        f.setSize(300,200);
        f.setVisible(true);
    }
    public Paneltest()
    {
    }
}
```

Panel – Ví dụ (tt)

❖ java.awt.Panel

- Ví dụ microwave GUI



Dialog (1)

- ❖ Là một cửa sổ dạng hộp hội thoại, cửa sổ dạng này thường được dùng để đưa ra thông báo, hay dùng để lấy dữ liệu nhập từ ngoài.

`java.awt`

Class Dialog

`java.lang.Object`

└ `java.awt.Component`

└ `java.awt.Container`

└ `java.awt.Window`

└ `java.awt.Dialog`



Dialog (2)

❖ Ví dụ:

```
Frame myframe = new Frame("My frame");  
String title = "Title";  
boolean modal = true;  
Dialog dlg = new Dialog(myframe, title, modal);
```


ScrollPanels

- ❖ Là một khung chứa **tương tự khung chứa Panel**, nhưng có **thêm 2 thanh trượt** giúp ta tổ chức và xem được các đối tượng lớn chiếm nhiều chỗ trên màn hình.

`java.awt`

Class ScrollPane

`java.lang.Object`

└ `java.awt.Component`

└ `java.awt.Container`

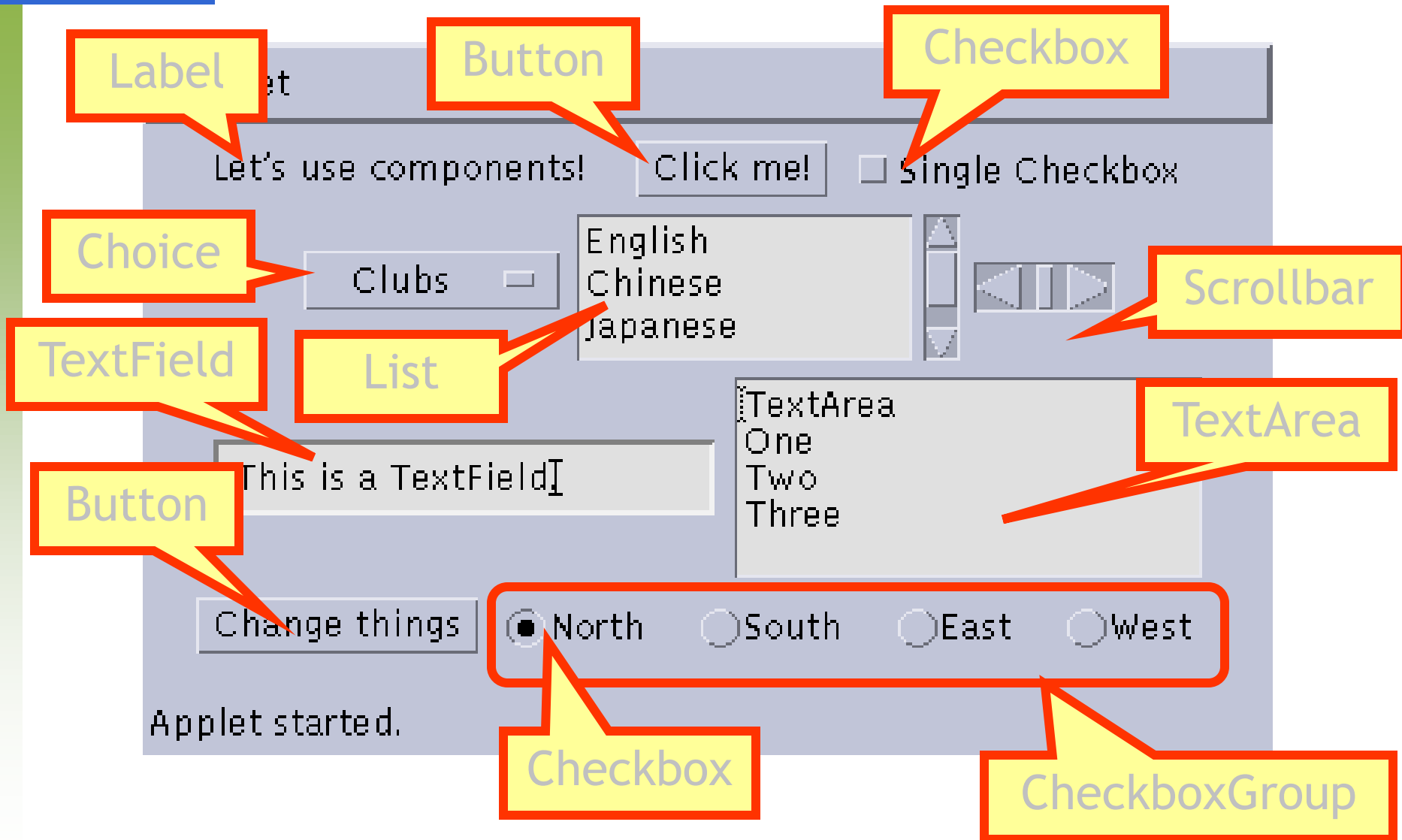
└ `java.awt.ScrollPane`



Thành phần - Components

- ❖ Tất cả các thành phần cấu tạo nên chương trình GUI được gọi là component.
- ❖ Ví dụ:
 - Containers,
 - TextFields, Labels, CheckBoxes, TextAreas
 - Scrollbars, scrollpanes, dialog

Thành phần - Components





Layout Manager

- ❖ Các bộ quản lý trình bày mà thư viện AWT cung cấp bao gồm:
 - FlowLayout
 - BorderLayout
 - GridLayout
 - GridBagLayout
 - Null Layout



FlowLayout

- ❖ Là Layout mặc định của **Panel**
- ❖ Các đối tượng được sắp xếp từ trái qua phải và từ trên xuống dưới.
 - Các đối tượng đều giữ nguyên kích thước của mình.
 - Nếu chiều rộng của Container không đủ chỗ cho các component thì chúng tự động tạo ra một dòng mới.
 - Có thể điều chỉnh khoảng cách giữa các component.
 - cách nhau theo chiều dọc (*vgap*), theo chiều ngang (*hgap*).



FlowLayout

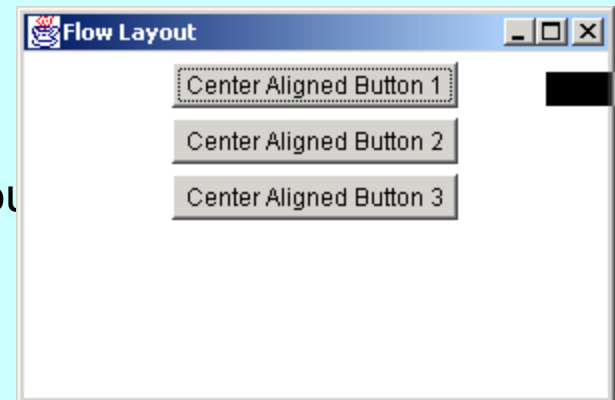
❖ Constructors:

- `FlowLayout()`
- `FlowLayout(int align)`
- `FlowLayout(int align, int hgap, int vgap)`
- Các điều khiển có thể **canh trái**, **canh phải** hay ở **giữa**. Sử dụng cú pháp sau:

```
setLayout(new FlowLayout(FlowLayout.RIGHT));
```

FlowLayout – Ví dụ

```
import java.awt.*;
class Fltest extends Frame{
    Button b1 = new Button("Center Aligned Button 1");
    Button b2 = new Button("Center Aligned Button 2");
    Button b3 = new Button("Center Aligned Button 3");
    public Fltest(String title){
        super (title);
        setLayout(new FlowLayout(FlowLayout.CENTER));
        add(b1);
        add(b2);
        add(b3);
    }
    public static void main(String args[ ]){
        Fltest t = new Fltest("Flow Layout");
        t.setSize(300,200);
        t.show();
    }
}
```

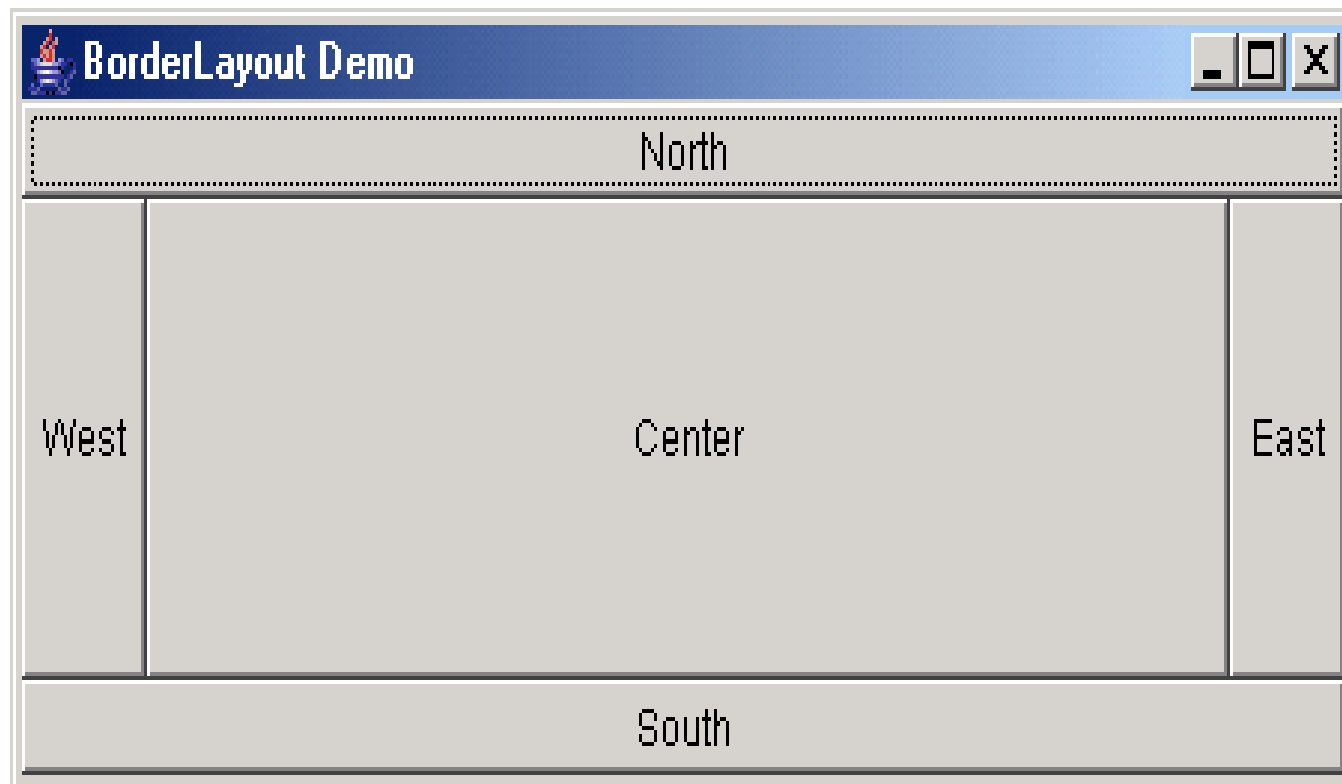




BorderLayout

- ❖ Là layout mặc định của **Window**, **Frame** và **Dialog**.
- ❖ Các đối tượng được đặt theo các **đường viền của khung chứa** theo các cạnh **West, East, South, North** và **Center** (tương ứng **Trái, Phải, Trên, Dưới** và **Giữa**).
- ❖ Nếu container chỉ có 1 component và đặt nó ở vị trí **CENTER** thì component này **phủ kín container**.

BorderLayout



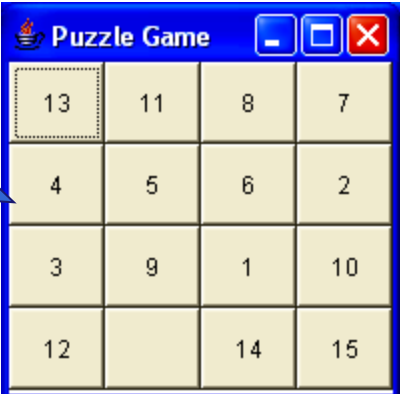
BorderLayout – Ví dụ

```
import java.awt.*;
class BorderLayoutDemo extends Frame{
    private Button north, south, east, west, center;
    public BorderLayoutDemo(String sTitle){
        super(sTitle);
        north = new Button("North");
        south = new Button("South");
        //...
        this.add(north, BorderLayout.NORTH);
        this.add(south, BorderLayout.SOUTH);
        //..
    }
    public static void main(String args[]){
        Frame fr = new BorderLayoutDemo ("BorderLayout Demo");
        fr.pack();
        fr.setVisible(true);
    }
}
```

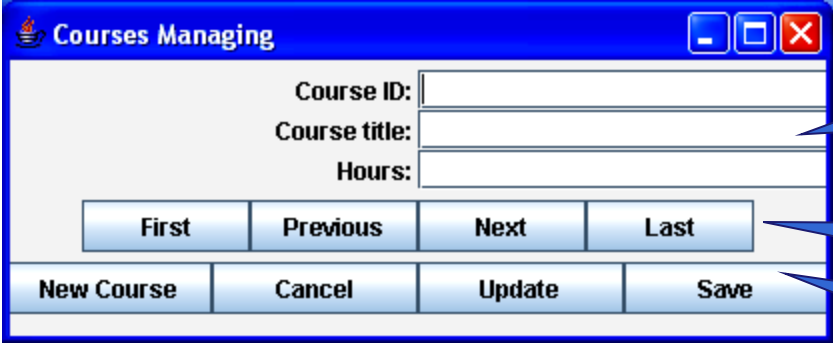
GridLayout

- ❖ Bố trí các component thành 1 lưới rows dòng, cols cột đều nhau.
- ❖ Kiểu layout này được sử dụng khi tất cả các thành phần có cùng kích thước.
- ❖ Thứ tự sắp xếp cũng từ trái qua phải và từ trên xuống dưới.

Lưới 4x4



13	11	8	7
4	5	6	2
3	9	1	10
12		14	15



Lưới 3x2

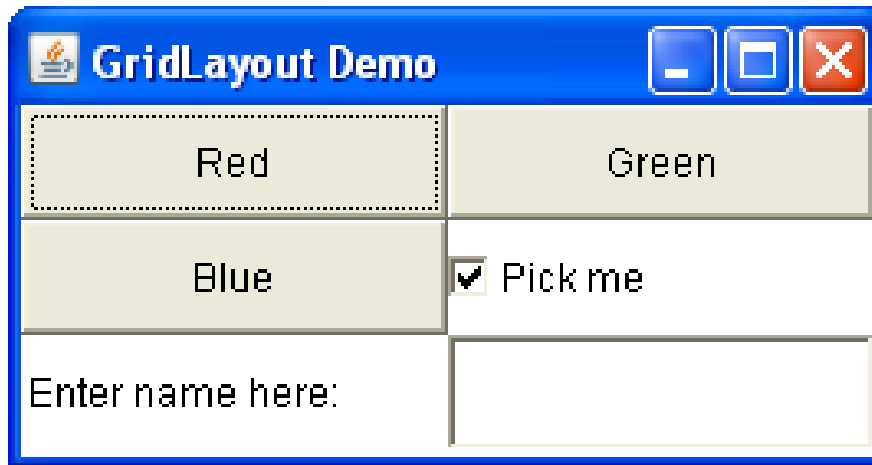
Lưới 1x4

Lưới 1x4

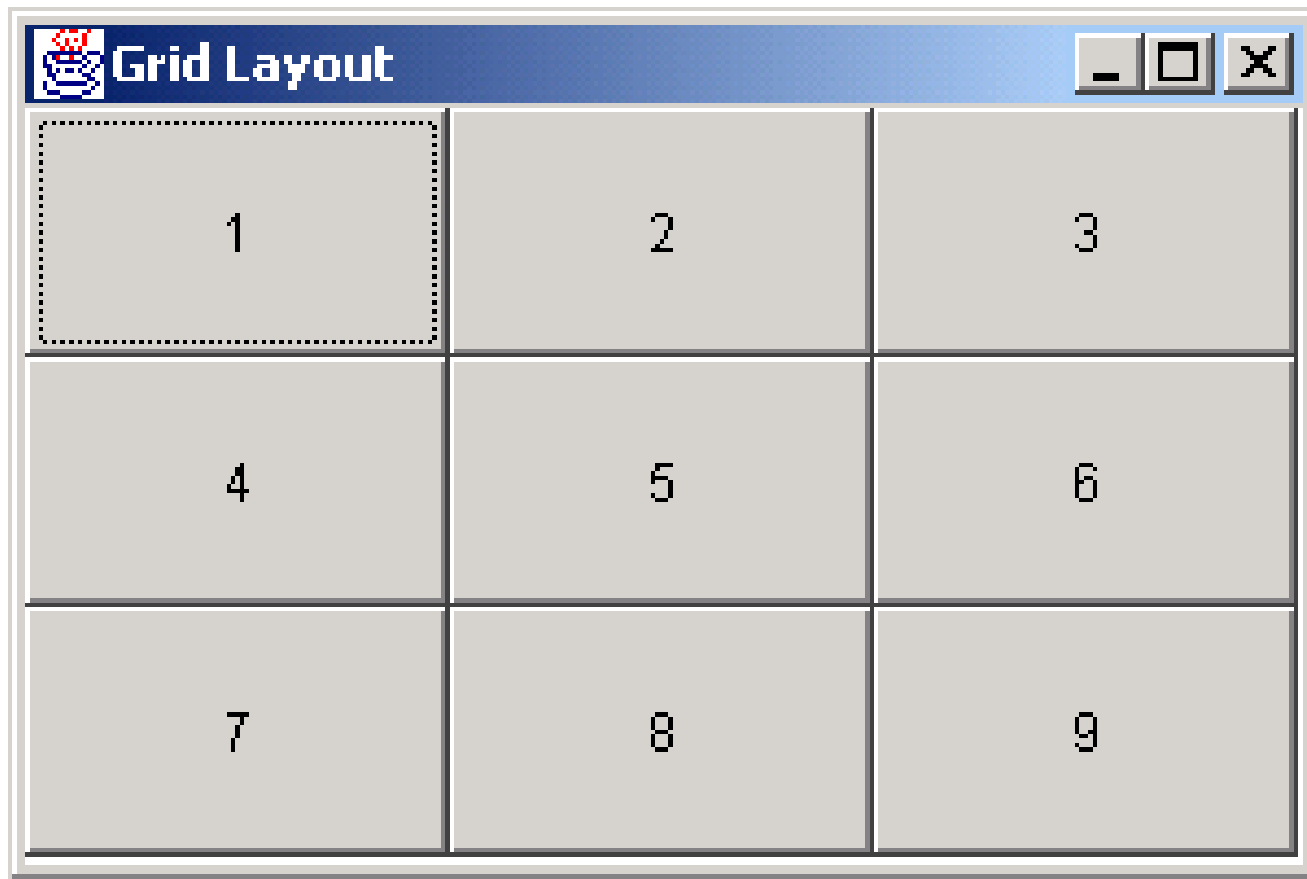
GridLayout

❖ Constructors:

- GridLayout()
- GridLayout(int **rows**, int **cols**)
- GridLayout(int **rows**, int **cols**, int **hgap**, int **vgap**)



GridLayout – Ví dụ



GridBagLayout

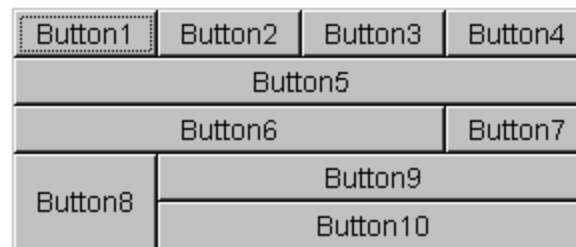
- ❖ Tương tự như GridLayout.
- ❖ Tuy nhiên kích thước các đối tượng có thể trên nhiều dòng và nhiều cột, được chỉ định thông qua đối tượng **GridBagConstraints**.



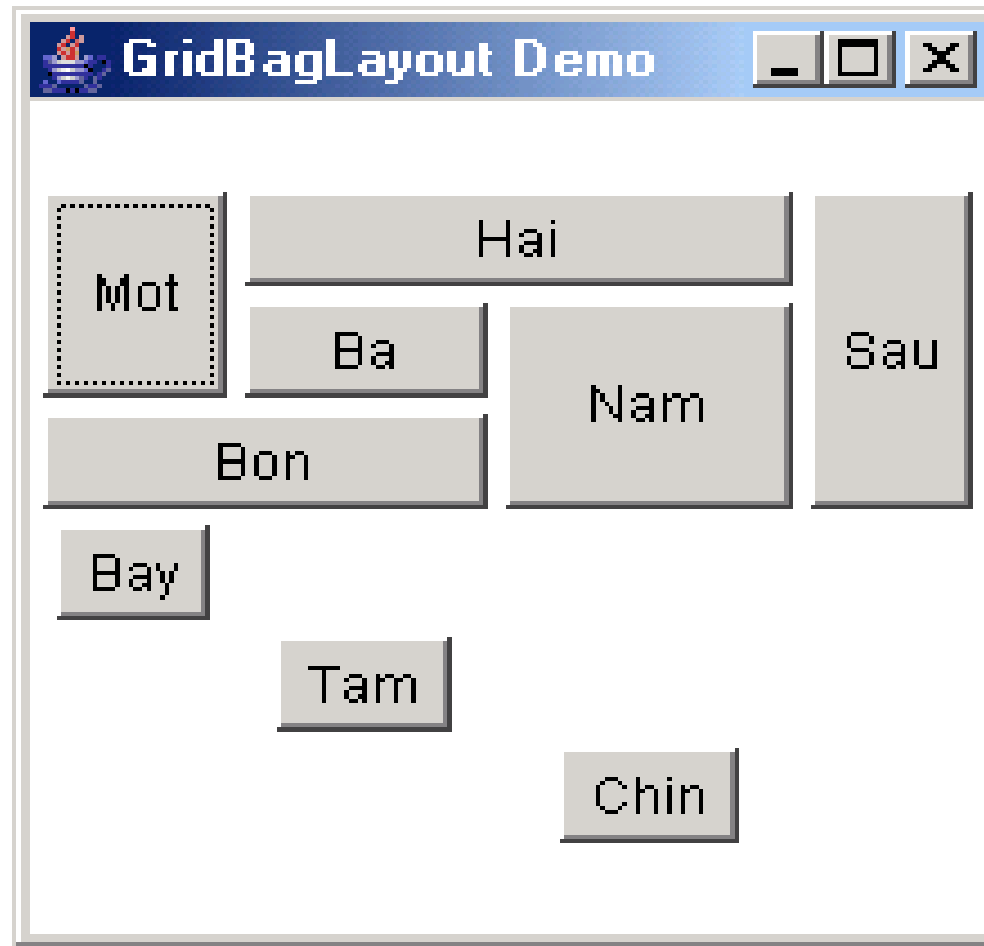
The screenshot shows a Java Swing window titled "Employee Info" with a blue title bar and standard window controls. The window contains a form with the following elements:

- Labels "Name:", "Birth:", and "Address:" followed by text input fields.
- A label "Sex:" followed by two radio buttons labeled "Male" (selected) and "Female".
- Four buttons at the bottom: "Add", "Find", "Delete", and "Update".

The components are arranged in a grid-like fashion, demonstrating the use of GridBagLayout.



GridBagLayout (3)





GridBagConstraints

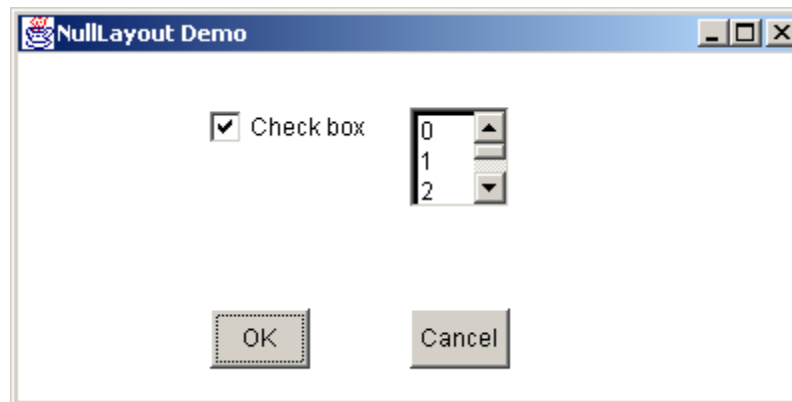
❖ Lớp `GridBagConstraints`.

- `int gridx, gridy`: Vị trí ô của khung lưới vô hình mà ta sẽ đưa đối tượng con vào
- `int gridwidth, gridheight`: Kích thước hay vùng trình bày cho đối tượng con.
- `Insets`: Là một biến đối tượng thuộc lớp `Inset` dùng để qui định khoảng cách biên.
- `double weightx, weighty`: Khoảng hở của lưới, mặc định là 0.

NullLayout

- ❖ Cách trình bày **tự do**.
- ❖ Người lập trình **phải xác định kích thước và vị trí** của các đối tượng trên màn hình.

```
Frame fr = new Frame("NullLayout Demo");  
fr.setLayout(null);
```



NullLayout – Ví dụ

```
class NullLayoutDemo{
    public static void main(String args[]){
        Frame fr = new Frame("NullLayout Demo");
        fr.setLayout(null);
        Button buttOk = new Button("OK");
        buttOk.setBounds(100, 150, 50, 30);
        Button buttCancel = new Button("Cancel");
        buttCancel.setBounds(200, 150, 50, 30);
        Checkbox checkBut = new Checkbox("Check box", true);
        checkBut.setBounds(100, 50, 100, 20);
        List li = new List();
        for (int i=0; i<5; i++)
            li.add(Integer.toString(i));
        li.setBounds(200, 50, 50, 50);
        fr.add(buttOk);                fr.add(buttCancel);
        fr.add(checkBut);              fr.add(li);
        fr.setBounds(10, 10, 400, 200);
        fr.setVisible(true);
    }
}
```



Nguyên tắc xây dựng GUI

- ❖ Lựa chọn 1 container: Frame, Dialog,...
- ❖ Tạo các điều khiển: label, button, text areas...
- ❖ Đưa các điều khiển vào vùng chứa
- ❖ Sắp xếp các điều khiển (layout)
- ❖ Thêm các xử lý sự kiện (Listeners)
 - Sẽ được giới thiệu trong chương sau



Tạo GUI cho ứng dụng

- ❖ Dữ liệu nhập/xuất → Chọn component?
- ❖ Các tác vụ → Mỗi tác vụ là một nút lệnh?

Dữ liệu	Đối tượng
Chuỗi nhập 1 dòng	TextField
Chuỗi nhiều dòng	TextArea
Chọn trong nhiều chuỗi	Choice
Chọn Yes/No (nhiều)	Checkbox
Chọn Yes/No (1/n)	CheckboxGroup + Checkbox
Dữ liệu chỉ xuất 1 dòng	Label, TextField-cắm nhập
Dữ liệu chỉ xuất nhiều dòng	TextArea – cắm nhập
Chuỗi nhập + xuất	TextField/TextArea



Tạo GUI cho ứng dụng

❖ Các cơ sở chọn Layout

- Thân thiện: chọn Layout giống mẫu hồ sơ.
- Trật tự **nhập liệu tự nhiên** của bài toán.
- Nếu GUI phức tạp thì **phân bổ các component vào nhiều panel, mỗi panel có một layout khác nhau.**

❖ Thói quen tốt khi đặt tên đối tượng

- Dùng tiếp đầu ngữ: **txt** cho TextField, **lbl** cho Label, **chk** cho Checkbox, **btn** cho Button, ...

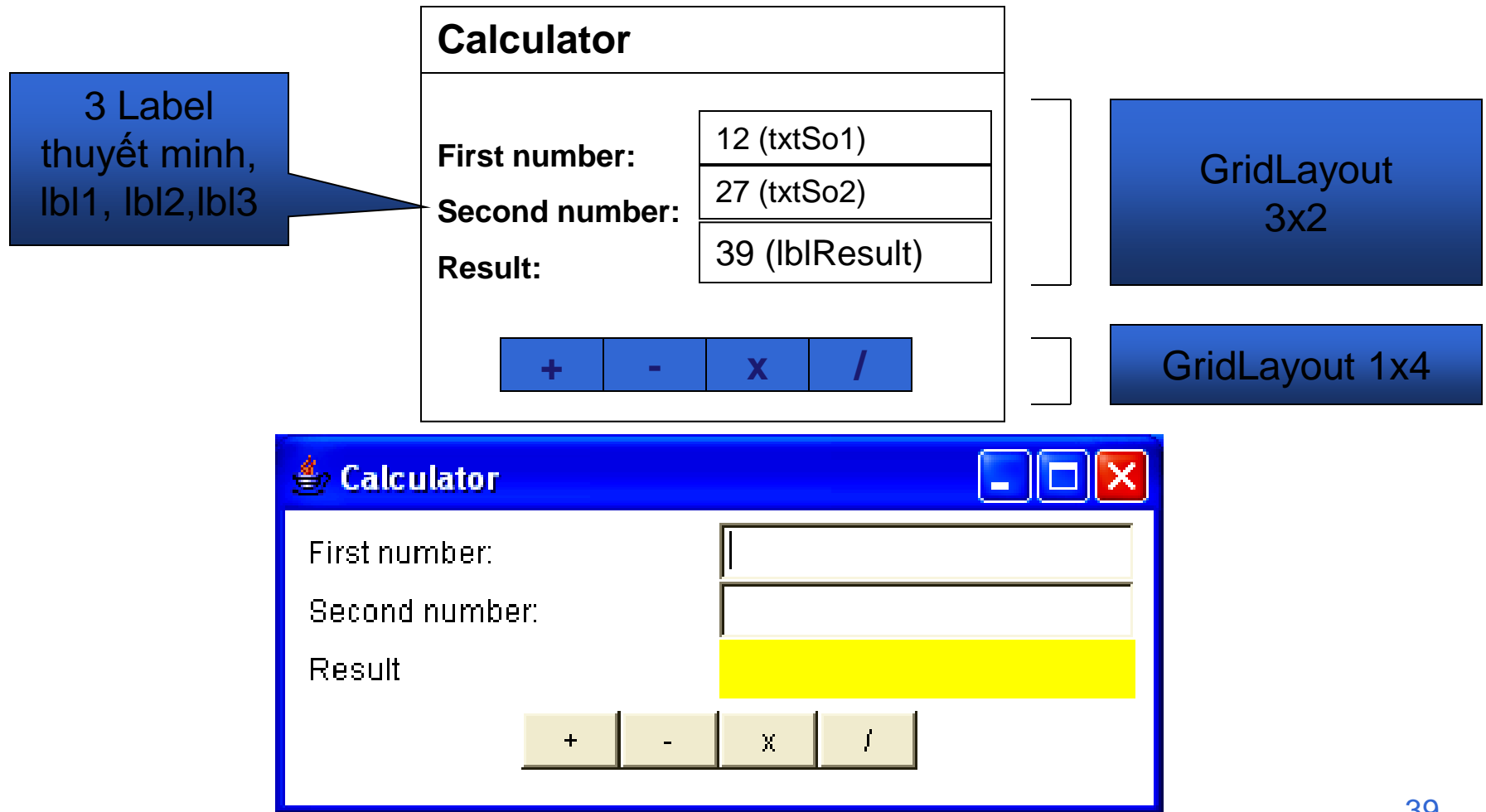


Bài tập

- ❖ Xây dựng ứng dụng cho phép thực hiện các phép tính +, -, *, /
 - Dữ liệu nhập: 2 số → 2 TextFiled, tên txtSo1, txtSo2.
 - Dữ liệu xuất: Kết quả của phép tính → Label, tên lblResult
 - Bốn tác vụ: Cộng, trừ, nhân, chia → 4 Button, tên btnAdd, btnSub, btnMul, btnDiv
 - Ba lời thuyết minh: Label, tên lbl1, lbl2, lbl3

Hướng dẫn

❖ Hình thức GUI:





Bài tập

1. Viết chương trình minh họa việc sử dụng các đối tượng components, đối tượng khung chứa container, bộ quản lý trình bày Layout Manager.
2. Viết chương trình xây dựng giao diện chương trình máy tính cá nhân tương tự chương trình Calculator trên windows.
3. Viết chương trình xây dựng giao diện tương tự giao diện của trình ứng dụng MS. WordPad trên Windows.
4. Viết chương trình xây dựng giao diện chương trình tương tự Windows Explorer.



Hỏi & đáp

