



## LẬP TRÌNH JAVA 3

### BÀI 3: LAYOUT, TIMER, JSLIDER

#### PHẦN 1

## 🎯 Kết thúc bài học này bạn có khả năng

### ❖ Layout Manager

- FlowLayout
- BorderLayout
- CardLayout
- TabbedPaneLayout
- GridLayout
- GridBagLayout
- BoxLayout
- GroupLayout

### ❖ Timer - ProgressBar

### ❖ JSlider



- ❑ Các component trong Java Swing được chia làm hai loại là container và children, trong đó container là loại component dùng để chứa các component khác, ví dụ như JFrame, JPanel... còn children là các component như JLabel, JButton, JTextField... khi chúng ta đặt các component con lên các container thì việc đặt và sắp xếp theo vị trí nào đó và nó được gọi là Layout.
- ❑ Mỗi Container có một đối tượng Layout Manager
- ❑ Layout Manager là một đối tượng quyết định cách sắp xếp vị trí của các Component bên trong một Container.
- ❑ Các Layout Manager "implements" từ interface LayoutManager hoặc LayoutManger2.

- ❑ Flow Layout bố trí các Component trong Container theo dòng, từ trái sang phải theo thứ tự thêm vào.
- ❑ Tạo dòng mới khi kích thước dòng còn lại không đủ chứa Component thêm vào.
- ❑ Flow Layout bố trí vị trí các Component phụ thuộc vào kích thước của Container.
- ❑ Mỗi dòng của các Component được window mặc định canh giữa theo chiều ngang . Có thể điều chỉnh canh trái hoặc phải



## ❑ Khởi tạo

### ❖ public FlowLayout ()

- align: FlowLayout.CENTER
- vgap: 5px, hgap: 5px

### ❖ FlowLayout (int align)

- align: canh lề
- FlowLayout.CENTER : Canh giữa
- FlowLayout.LEFT; : Canh trái
- FlowLayout.RIGHT; : Canh phải

### ❖ FlowLayout(int align, int vgap, int hgap)

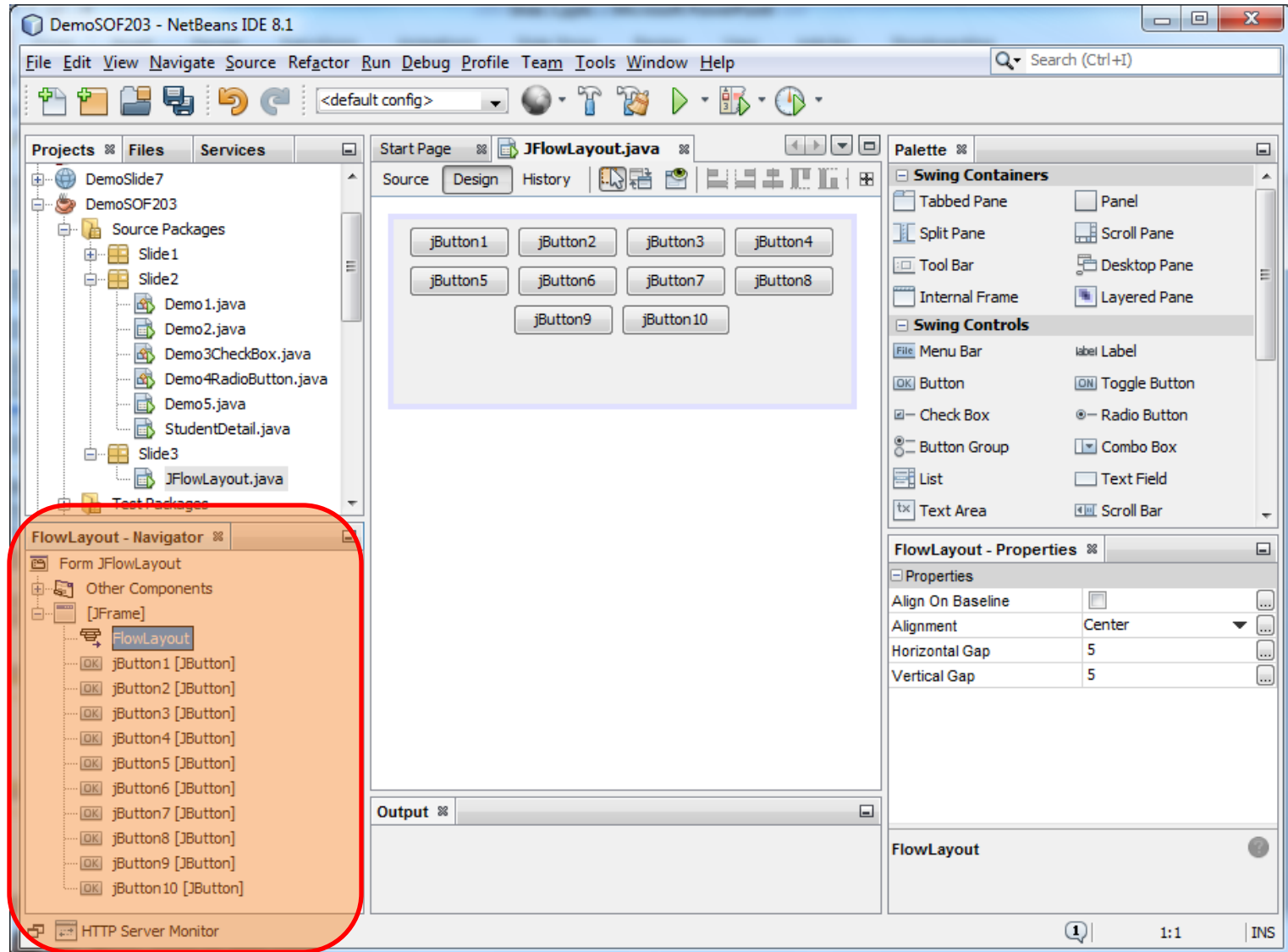
- align : canh lề
- vgap : kích thước chiều ngang
- hgap: chiều dọc



## ❑ Phương thức

- ❖ `public void setAlignment(int align)`
- ❖ `public void setHgap(int hgap)`
- ❖ `public void setVgap (int vgap)`
- ❖ `public int getAlignment()`
- ❖ `public int getHgap ()`
- ❖ `public int getVgap ()`

## □ Demo: FlowLayout





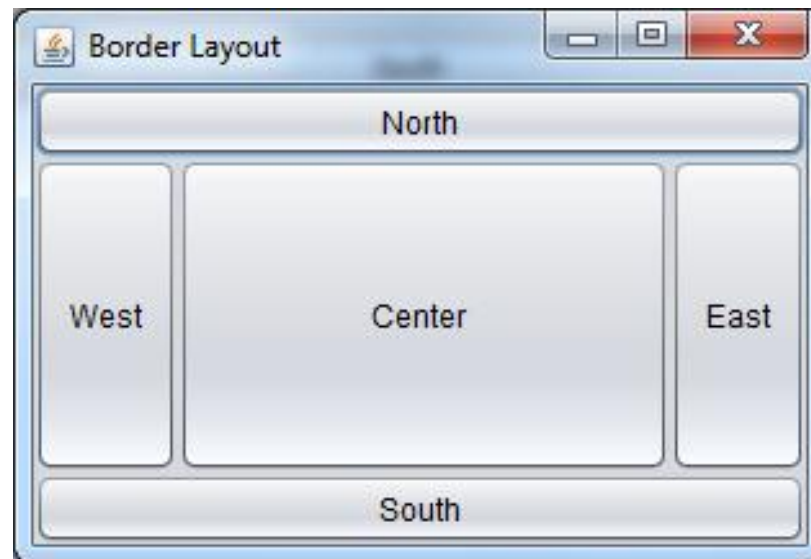
# DEMO

Chạy và giải thích





- ❑ Border Layout bố trí các Component bên trong Container theo 5 vùng: "North", "South", "East", "West", "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.



□ khởi tạo:

❖ `public BorderLayout ()`

➤ `hgap = 0`

➤ `vgap = 0`

❖ `public BorderLayout (int hgap, int vgap)`

➤ `hgap`: chiều ngang

➤ `vgap` : chiều dọc

## Demo: Border Layout

The screenshot shows the NetBeans IDE 8.1 interface. The main editor displays a Java Swing window titled "JBorderLayout.java" with a BorderLayout. The window is divided into four regions: North, South, West, and East, surrounding a central area. The "BorderLayout - Navigator" window is open, showing the hierarchy of components. The "Palette" window shows various Swing containers and controls. The "Output" window shows a successful build message.

**BorderLayout - Navigator**

- Form JBorderLayout
  - Other Components
    - [JFrame]
      - BorderLayout
        - jButton1 [JButton]
        - jButton2 [JButton]
        - jButton3 [JButton]
        - jButton4 [JButton]
        - jButton5 [JButton]

**BorderLayout - Properties**

Properties	
Horizontal Gap	0
Vertical Gap	0

**Output - DemoSOF203 (run)**

```
BUILD SUCCESSFUL (total time: 1 minute 10 seconds)
```

- ❑ Card Layout quản lý nhiều Card cùng một không gian hiển thị, 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.
- ❑ Card Layout giúp quản lý hai hay nhiều Component (thường là JPanel) để chia sẻ cùng một không gian hiển thị.
- ❑ Chỉ duy nhất Top Card được hiển thị.
- ❑ Mỗi "Card" có thể sử dụng Layout Manager riêng.
- ❑ Card nào cũng có thể là Top Card
- ❑ Có thể sử dụng **JTabbedPane** để thay cho **Card Layout**

❑ khởi tạo :

❖ `public CardLayout ()`

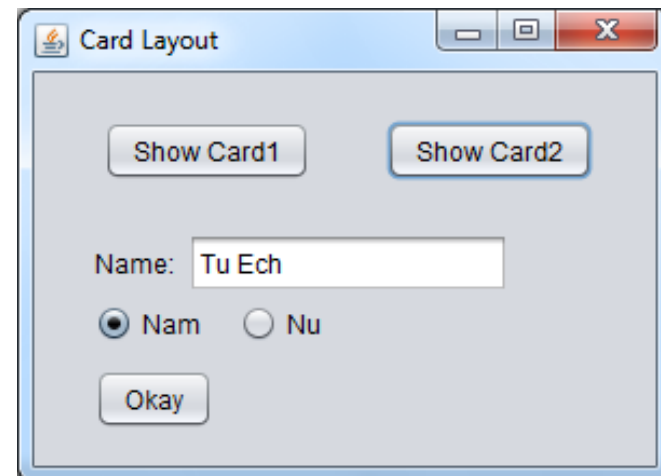
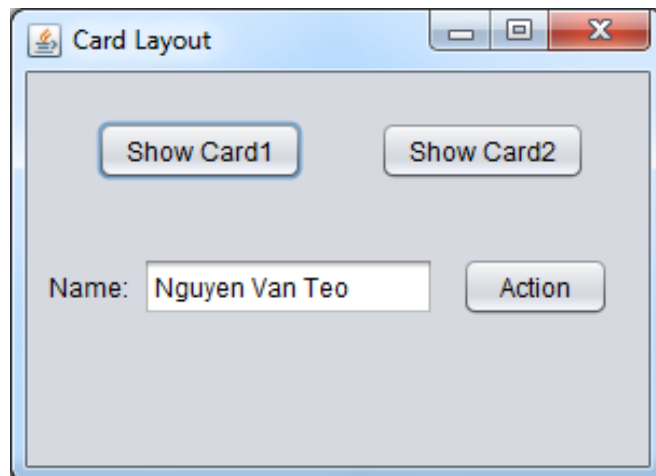
➤ `hgap = 0`

➤ `vgap = 0`

❖ `public CardLayout (int hgap, int vgap)`

➤ `hgap`: chiều ngang

➤ `vgap` : chiều dọc



## □ Phương thức

- ❖ `public void setHgap(int hgap)`
- ❖ `public void setVgap(int vgap)`
- ❖ `public int getHgap()`
- ❖ `public int getVgap()`
- ❖ `public void next (Container parent)`
- ❖ `public void previous(Container parent)`
- ❖ `public void first(Container parent)`
- ❖ `public void last(Container parent)`
- ❖ `public void show(Container parent, String name)`

## ❑ Demo: Card Layout

The screenshot displays the NetBeans IDE 8.1 interface for a project named 'DemoSOF203'. The main workspace is in 'Design' mode, showing a visual representation of a Java Swing window titled 'JPanel1 [JPanel]'. The window contains two buttons, 'Show Card1' and 'Show Card2', and a card layout area below them. The card layout area includes a text field labeled 'Name:', two radio buttons labeled 'Nam' and 'Nu', and an 'Okay' button. The 'Projects' window on the left shows the project structure, with 'JPanel1 [JPanel]' selected and its components listed in a tree view. The 'Palette' window on the right shows various Swing components like Tabbed Pane, Split Pane, etc. The 'Properties' window at the bottom right shows the properties of the selected component. The 'Output' window at the bottom shows the successful build message.

**Projects Window:**

- Slide3
  - JBorderLayout.java
  - JCardLayout.java
  - JFlowLayout.java
- Test Packages
- Libraries
- Test Libraries
- DemoStruts2
- MCVDemo
- Ngay2Strut

**jPanel1 [JPanel] - Navigator:**

- Form JCardLayout
  - Other Components
    - buttonGroup1 [ButtonGroup]
  - [JFrame]
    - jPanel1 [JPanel]
      - CardLayout
        - jPanel2 [JPanel]
          - label1 [JLabel]
          - jTextField1 [JTextField]
          - OK jButton1 [JButton]
        - jPanel3 [JPanel]
          - jRadioButton5 [JRadioButton]
          - OK jButton4 [JButton]
          - label4 [JLabel]
          - jRadioButton6 [JRadioButton]
          - jTextField4 [JTextField]
      - btnCard1 [JButton]
      - btnCard2 [JButton]

**Palette:**

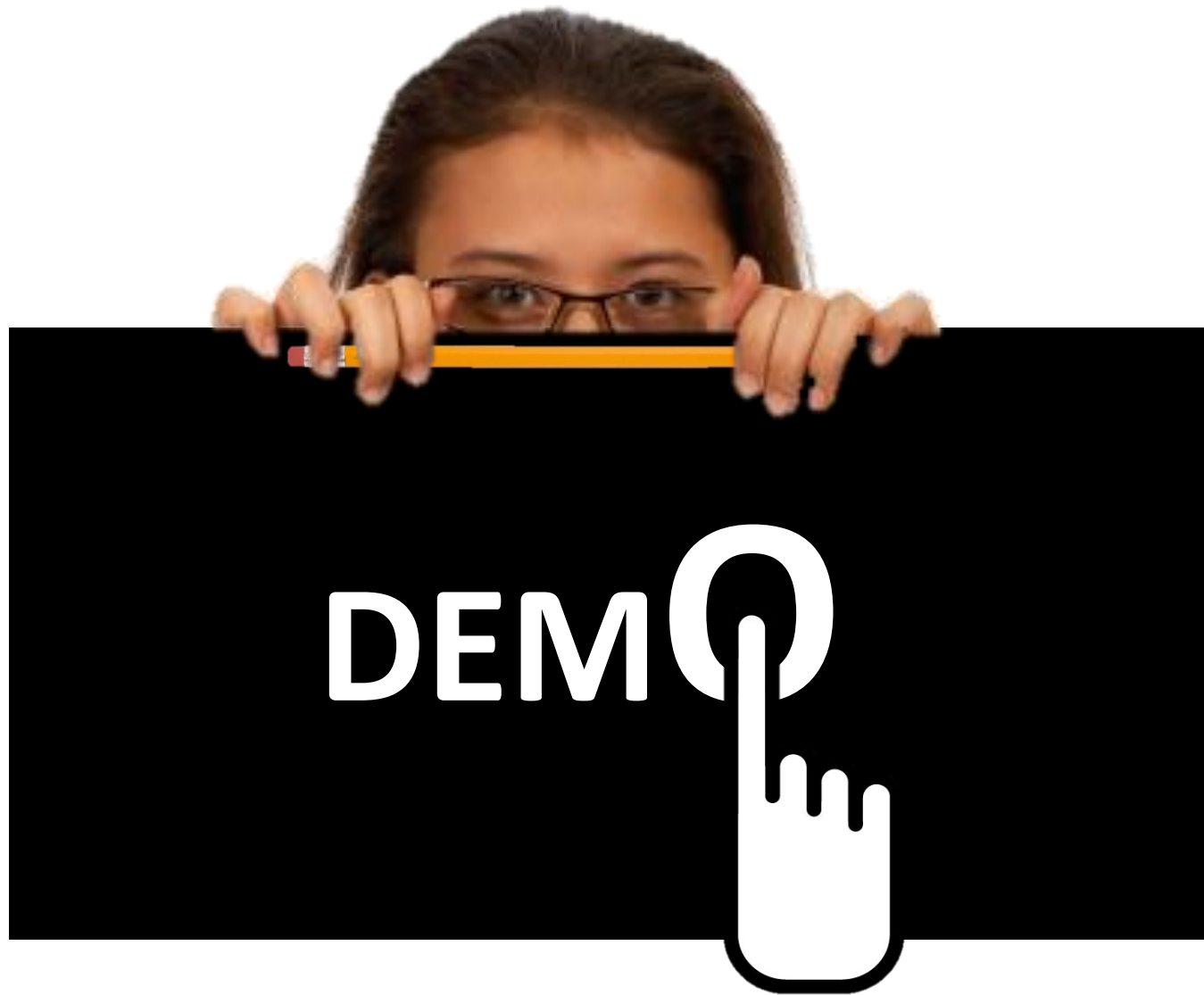
- Swing Containers**
  - Tabbed Pane
  - Split Pane
  - Tool Bar
  - Internal Frame
  - Panel
  - Scroll Pane
  - Desktop Pane
  - Layered Pane
- Swing Controls**
  - Menu Bar
  - Button
  - Check Box
  - Button Group
  - List
  - Label
  - Toggle Button
  - Radio Button
  - Combo Box
  - Text Field

**jPanel1 [JPanel] - Properties:**

Properties	Binding
background	[240,240,240]
border	(No Border)
foreground	[0,0,0]
toolTipText	
<b>Other Properties</b>	
UIClassID	PanelUI
alignmentX	0.5
alignmentY	0.5

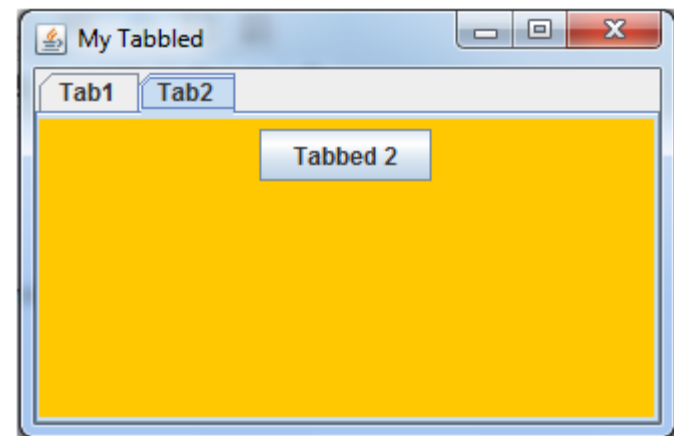
**Output - DemoSOF203 (run):**

```
run:
BUILD SUCCESSFUL (total time: 21 s)
```





- ❑ Ngoài ra ta có thể dùng JTabbedPane để thay thế cho CardLayout, JTabbedPane có giao diện đẹp mắt và thân thiện với người sử dụng



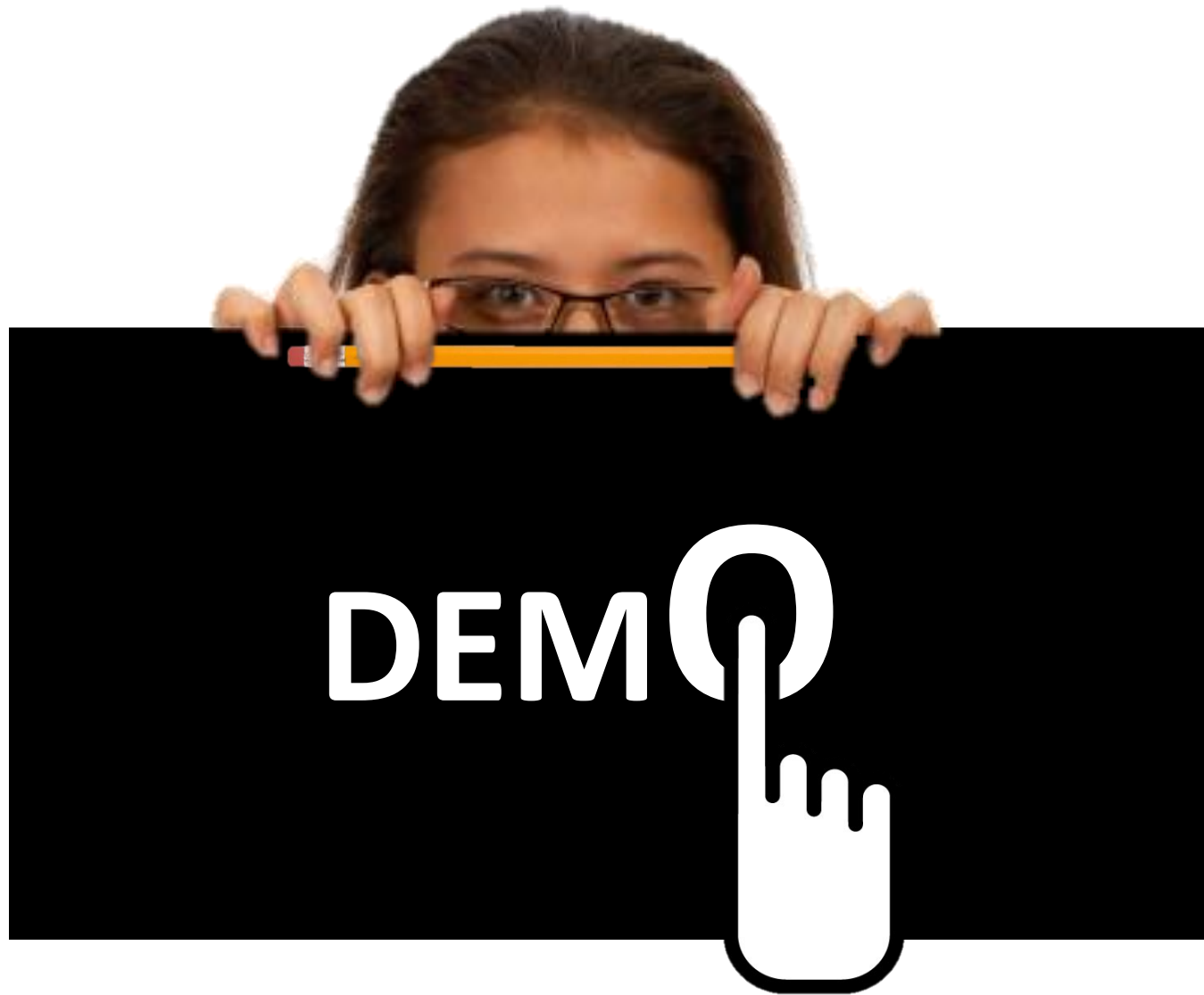
## □ Demo: JTabbedPane

The screenshot displays the NetBeans IDE 8.1 environment. The main window shows the **JTabbedPane.java** file in the **Design** view, featuring three tabs labeled **Tab1**, **Tab2**, and **Tab3**. The **Tab1** tab is active, showing a text field labeled **Name:** and an **Active** button.

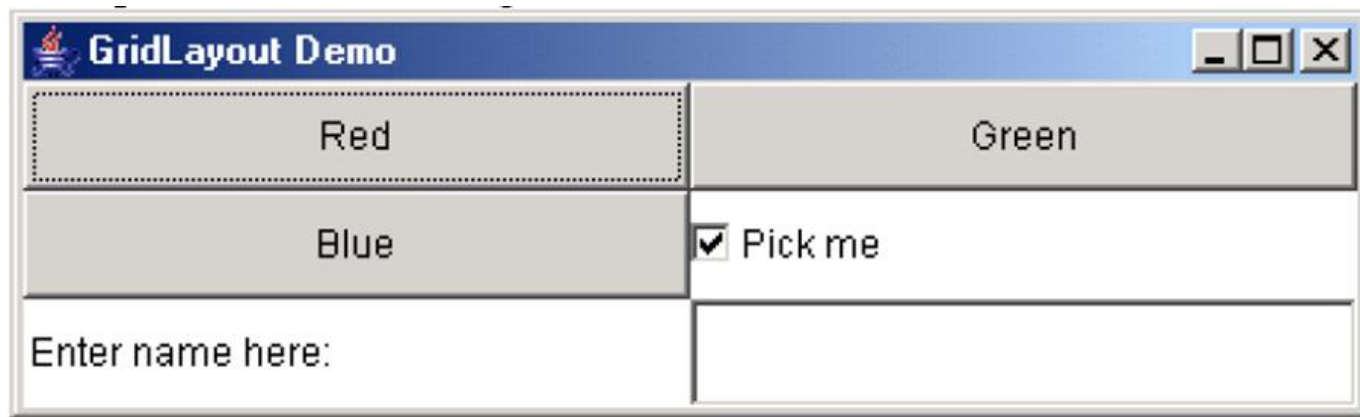
The **Project Explorer** on the left shows the project structure, including **Slide3** and **Test Packages**. The **Palette** on the right lists various **Swing Containers** and **Swing Controls**. The **Properties** window for **jTabbedPane1 [JTabbedPane]** is open, showing the following properties:

Property	Value
background	[240,240,240]
border	(No Border)
foreground	[0,0,0]
selectedIndex	0
tabLayoutPolicy	WRAP_TA...
tabPlacement	TOP
toolTipText	

The **Output** window at the bottom shows the message: **BUILD SUCCESSFUL (total time: 7 minutes)**.



- ❑ Grid Layout bố trí các Component của Container vào bên trong một Grid với các Row và Column.
- ❑ Mỗi Component được bố trí trong 1 Cell của lưới.
- ❑ Tất cả các Cell có cùng kích thước bên trong Grid.
- ❑ Các Component được thêm vào Container theo thứ tự từ trái sang phải, từ trên xuống dưới (mặc định).



## □ Khởi tạo

- ❖ `GridLayout()`
- ❖ `GridLayout(int rows, int cols)`
- ❖ `GridLayout(int rows, int cols, int hgap, int vgap)`

## Demo: GridLayout

- ❑ GridBag Layout bố trí các Component trong một Grid với các Row và Column.
- ❑ Mỗi Component bên trong Grid được RowSpan và ColumnSpan (giống table HTML)
- ❑ Width và Height của các Row/Column có thể khác nhau.
- ❑ GridBag Layout là một Layout Manager rất linh động cho việc bố trí các Component bên trong Container theo dạng Grid.
- ❑ GridBag Layout là một trong các Layout Manager **thường sử dụng nhất** mà Java Platform cung cấp.

## ❑ Các thuộc tính của GridBagConstraints

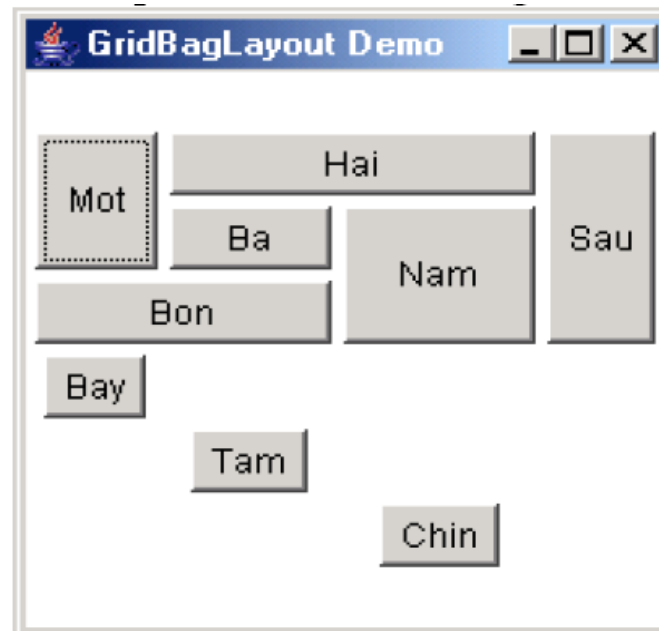
- ❖ gridx ,gridy : vị trí dòng, vị trí cột
- ❖ gridheight , gridwidth : số lượng dòng, số lượng cột
- ❖ ipadx, ipady :
- ❖ Insets
- ❖ weightx, weighty
- ❖ fill :
  - NONE, HORIZONTAL, VERTICAL, BOTH



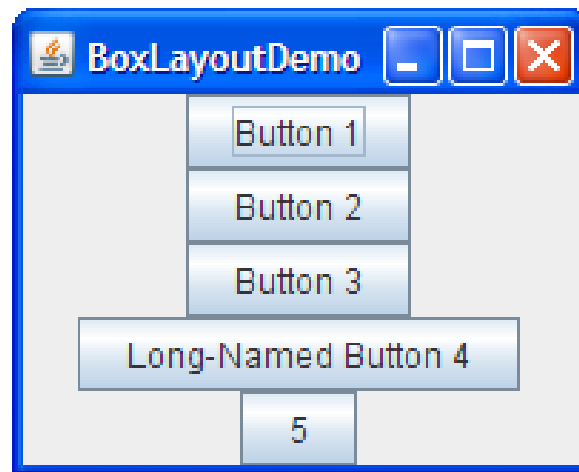
## ❑ Các thuộc tính của GridBagConstraints

❖ anchor:

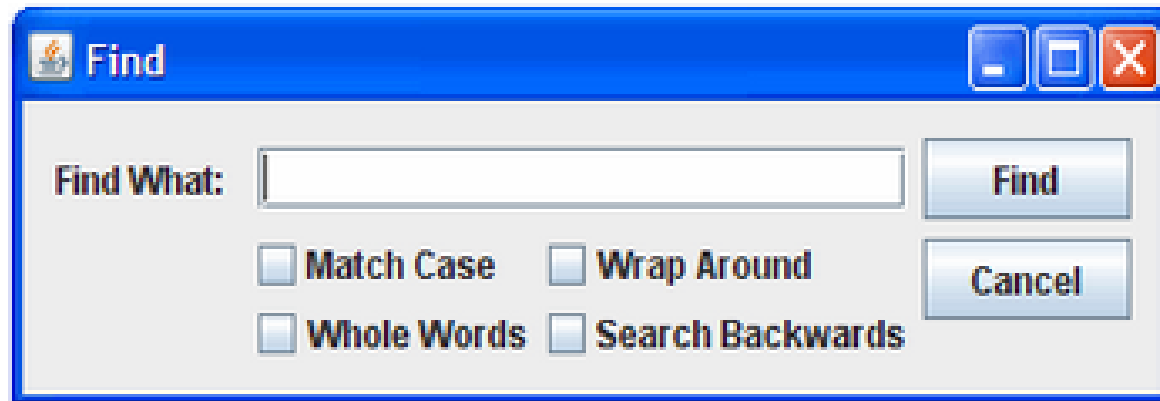
```
-----  
| FIRST_LINE_START   PAGE_START   FIRST_LINE_END |  
|                   |             |                   |  
| LINE_START         CENTER        LINE_END          |  
|                   |             |                   |  
| LAST_LINE_START    PAGE_END      LAST_LINE_END     |  
-----
```

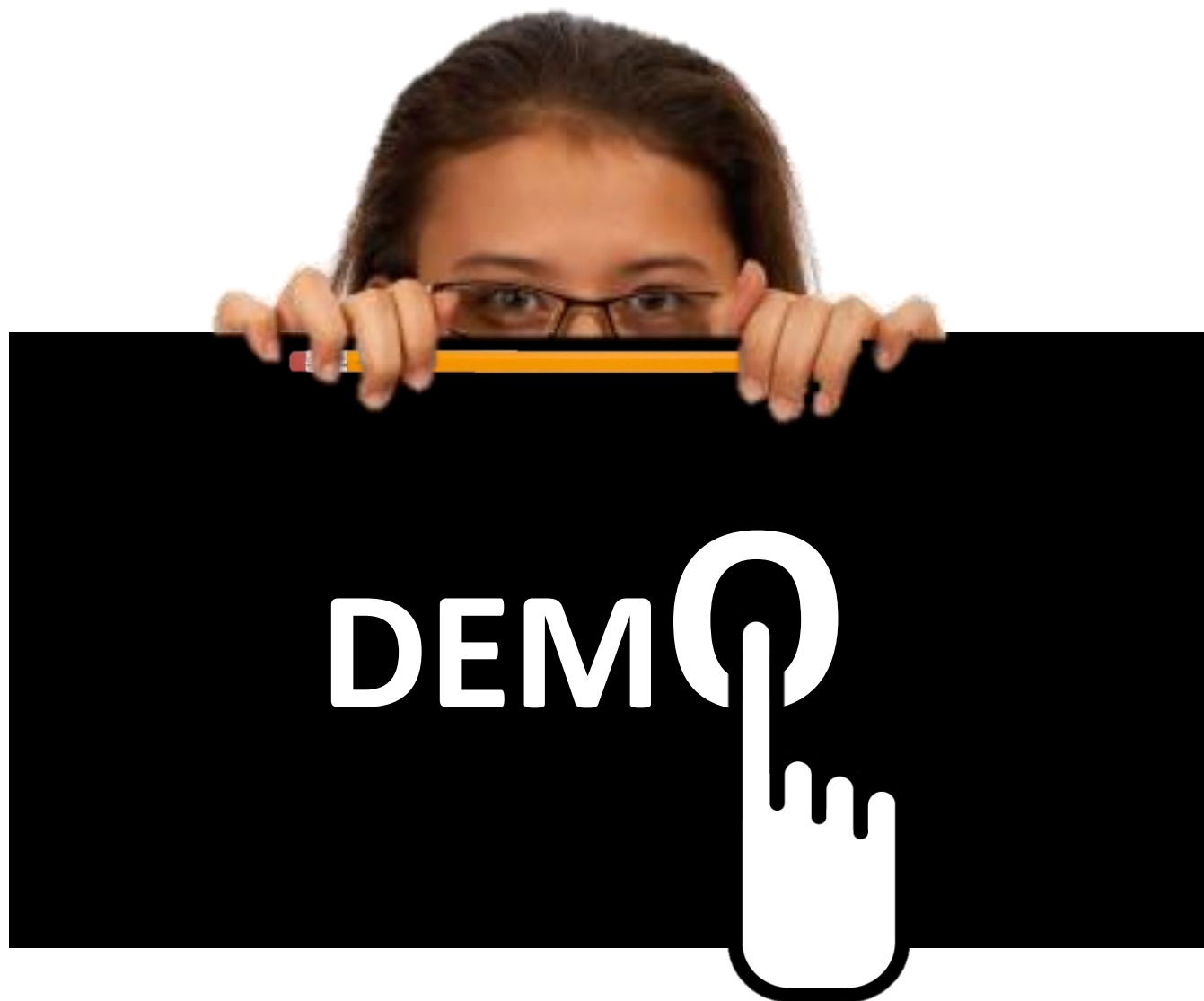


- ❑ Box Layout bố trí các Component bên trong Container theo 1 dòng theo trục X, hoặc là trục Y.
- ❑ `BoxLayout(Container container, int align)`
- ❑ `container`: chứa các Component
- ❑ `axis`:
- ❑ `BoxLayout.X_AXIS` : Trục X
- ❑ `BoxLayout.Y_AXIS` : Trục Y



- ☐ Group Layout bố trí các Component bên trong Container theo chiều ngang và chiều dọc.
- ☐ Sự bố trí được thực hiện theo mỗi chiều riêng lẻ







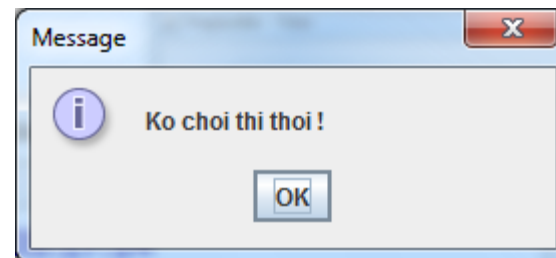
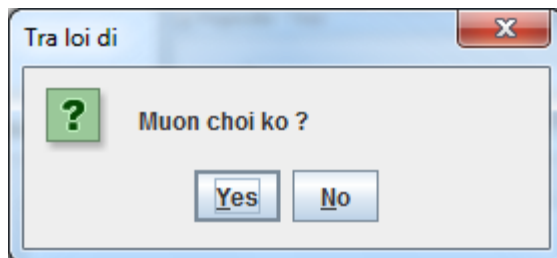
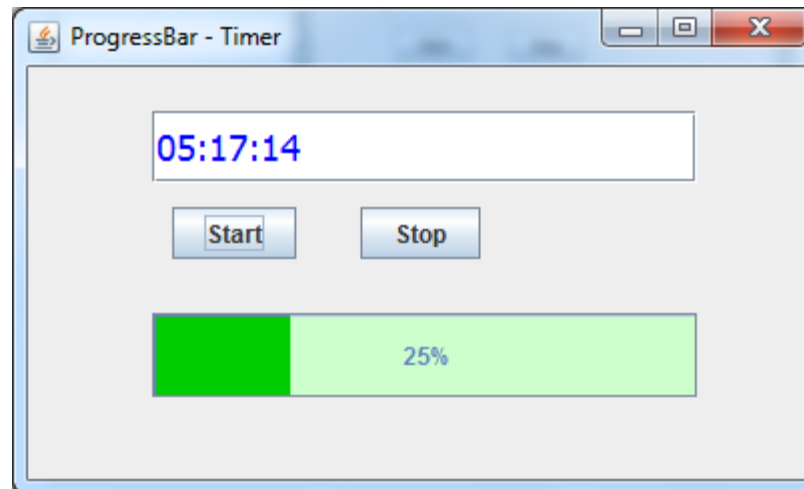
## LẬP TRÌNH JAVA 3

### BÀI 3: LAYOUT, TIMER, JSLIDER

#### PHẦN 2

- ❑ Timer dùng để kích hoạt một hoặc nhiều sự kiện ActionEvents sau một khoảng thời gian nhất định.
- ❑ Hàm tạo(constructor):  
Timer(int delay, ActionListener listener)  
Hàm tạo này sẽ tạo một sự kiện sau mỗi khoảng thời gian tính theo đơn vị mili giây

## □ Demo

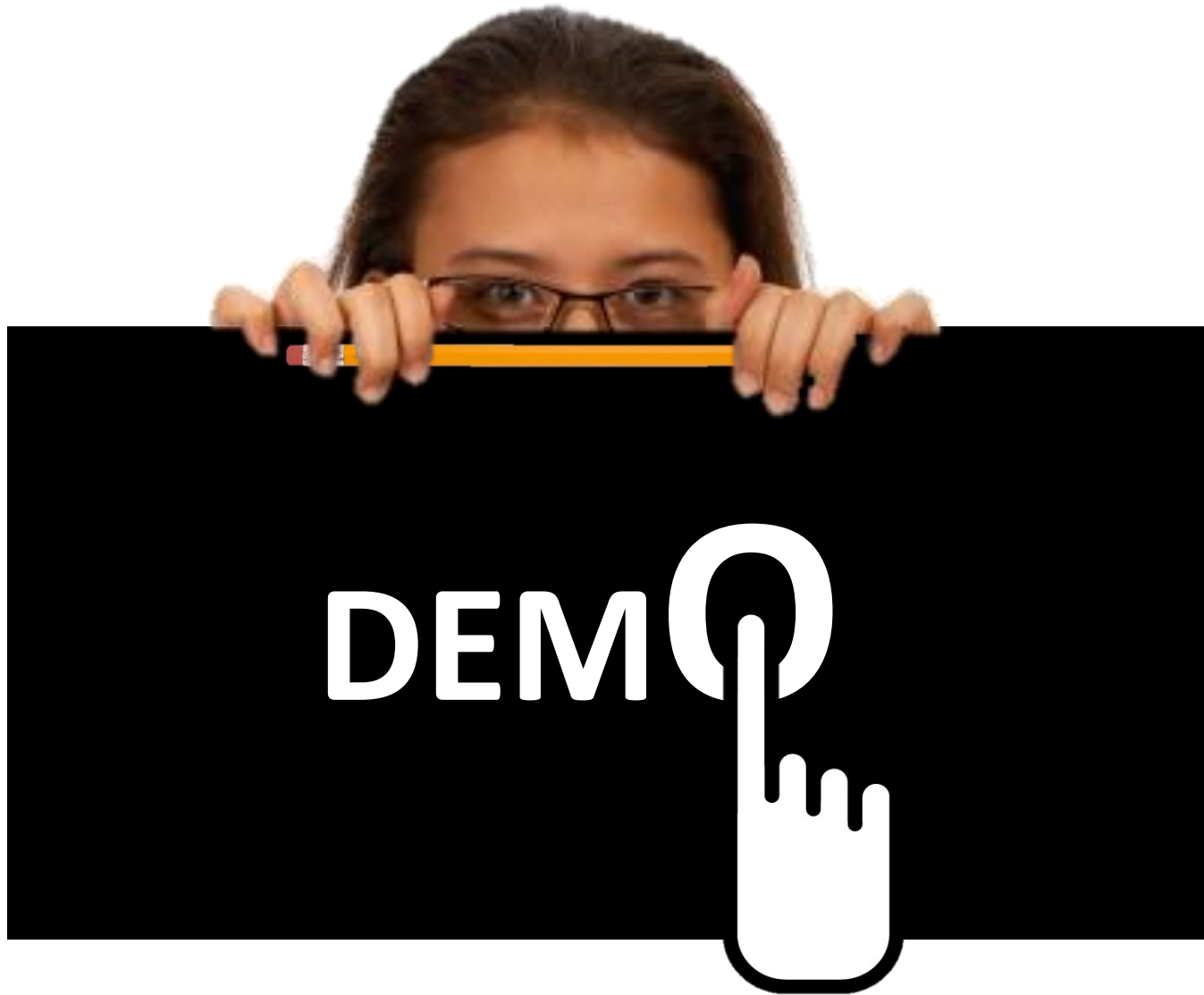


```
public class JProgressBarTimer extends javax.swing.JFrame {
    javax.swing.Timer t;
    public JProgressBarTimer() {
        initComponents();
        ActionListener a = new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                SimpleDateFormat sdf = new SimpleDateFormat("hh:mm:ss");
                txtClock.setText(sdf.format(new Date()));
                int value = prgTienDo.getValue()+1;
                prgTienDo.setValue(value<=100?value:0);
            }
        };
        t = new javax.swing.Timer(1000, a);
    }
    //.....
}
```

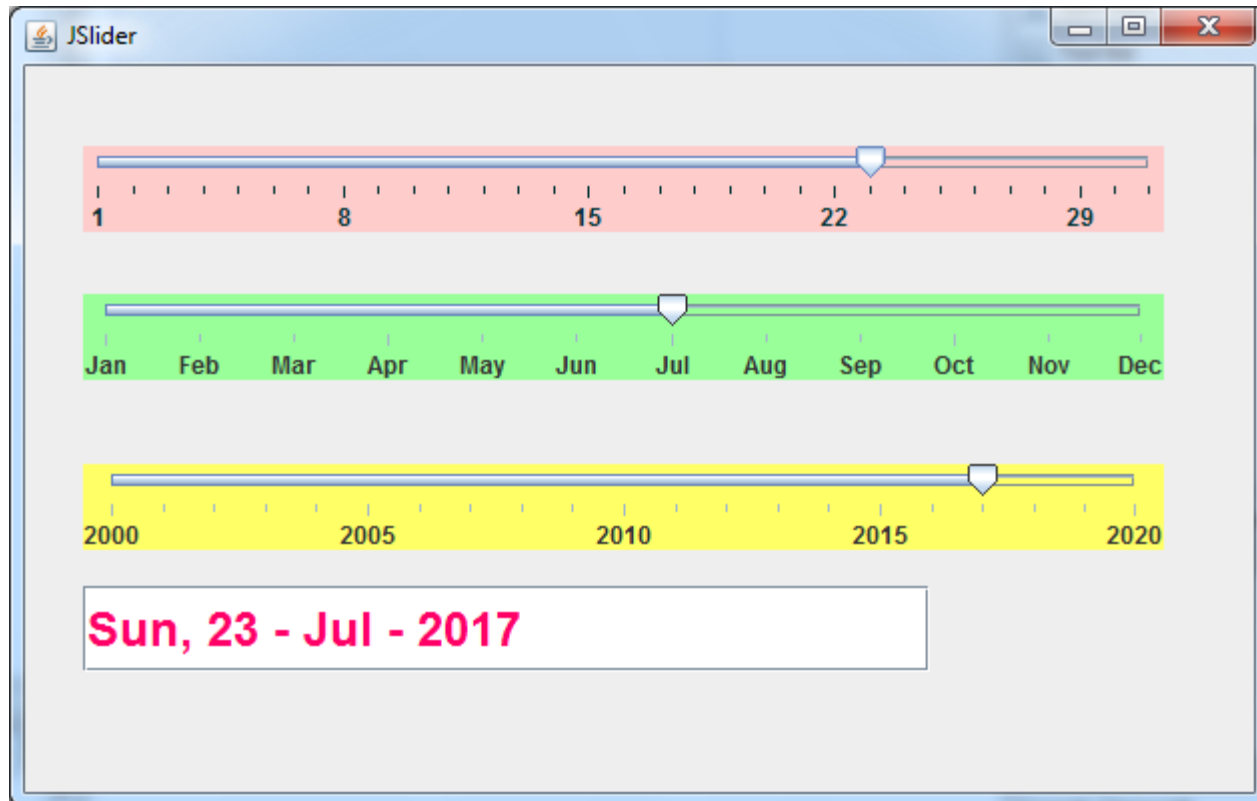
```
private void btnStartActionPerformed(java.awt.event.ActionEvent evt) {
    int i = JOptionPane.showConfirmDialog(null, "Muon choi ko ?", "Tra loi di",
        JOptionPane.YES_NO_OPTION);
    if(i==0)
        t.start();
    else
        JOptionPane.showMessageDialog(null, "Ko choi thi thoi !");
}
```

```
private void btnStopActionPerformed(java.awt.event.ActionEvent evt) {
    t.stop();
}
```



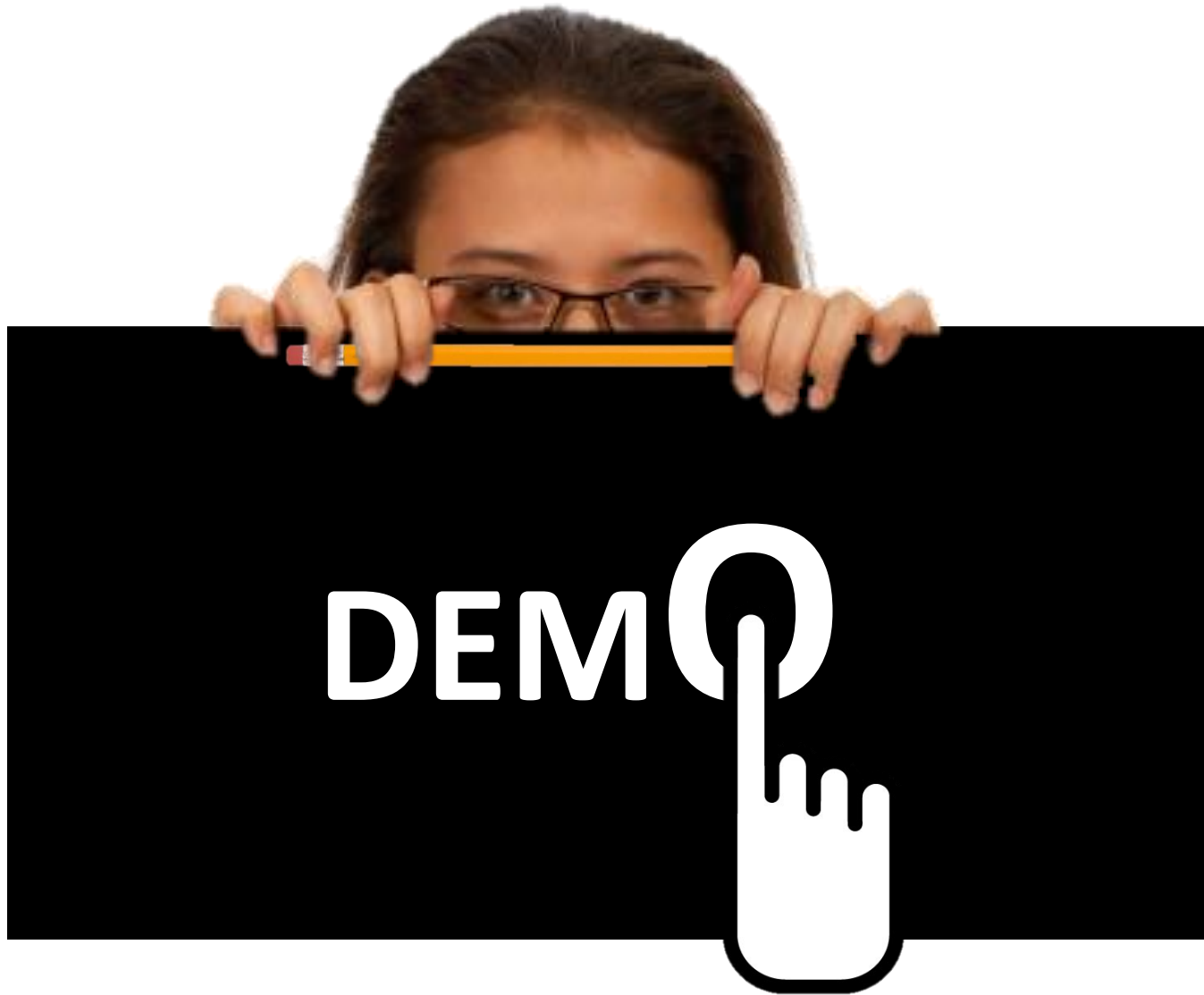


## □ Demo JSlider



```
public class JSsliderDemo extends javax.swing.JFrame {
    public JSsliderDemo() {
        initComponents();
        Hashtable data = new Hashtable();
        String [] sMonths= {"Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul", "Aug",
            "Sep", "Oct", "Nov", "Dec"};
        for(int i =0; i<12; i++){
            data.put(i, new JLabel(sMonths[i]));
        }
        sldThang.setLabelTable(data);
        Calendar can = Calendar.getInstance();
        sldNgay.setValue(can.get(Calendar.DAY_OF_MONTH));
        sldThang.setValue(can.get(Calendar.MONTH));
        sldNam.setValue(can.get(Calendar.YEAR));
        displayDate();
    }
    void displayDate(){
        SimpleDateFormat sdf = new SimpleDateFormat("EEE, dd - MMM - yyyy");
        int dd = sldNgay.getValue();
        int mm = sldThang.getValue();
        int yy = sldNam.getValue()-1900;
        Date d = new Date(yy, mm, dd);
        txtNgay.setText(sdf.format(d));
    }
    //.....

    private void sldNgayStateChanged(javax.swing.event.ChangeEvent evt) {
        displayDate();
    }
    private void sldThangStateChanged(javax.swing.event.ChangeEvent evt) {
        displayDate();
    }
    private void sldNamStateChanged(javax.swing.event.ChangeEvent evt) {
        displayDate();
    }
}
```



- ❖ Layout Manager
  - FlowLayout
  - BorderLayout
  - CardLayout
  - TabbedPaneLayout
  - GridLayout
  - GridBagLayout
  - BoxLayout
  - GroupLayout
- ❖ Timer – ProgressBar
- ❖ JSlider





**Cảm ơn**