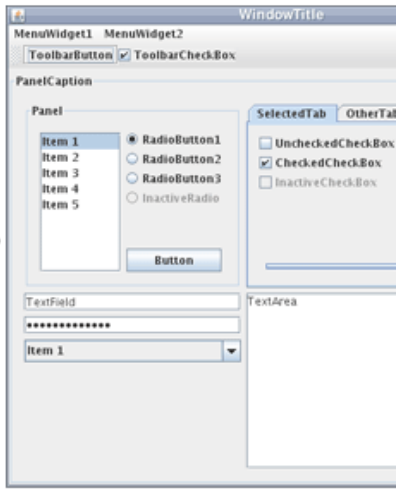
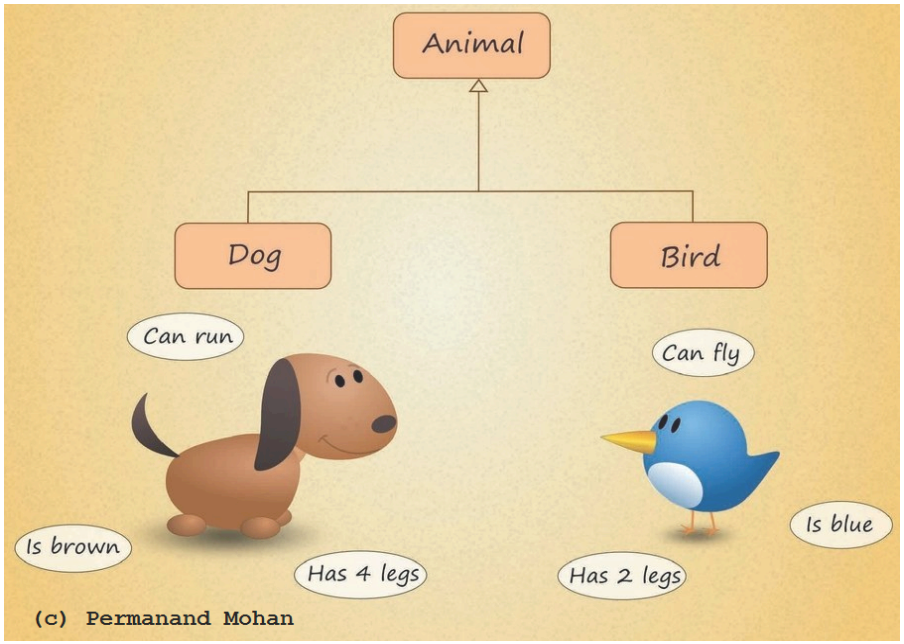




Chapter 5

Lập trình giao diện đồ họa

CT176 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

Mục tiêu

Chương này nhằm giới thiệu
cách thức xây dựng giao diện đồ họa trong Java

Nội dung

- Giới thiệu
- Tạo 1 ứng dụng với giao diện đồ họa
- Các lớp vật chứa
- Các thành phần giao diện Swing
- Sắp xếp bố cục
- Xử lý sự kiện
- Trình đơn, thanh công cụ
- Mô hình MVC

Sắp xếp bố cục (Layout managers)

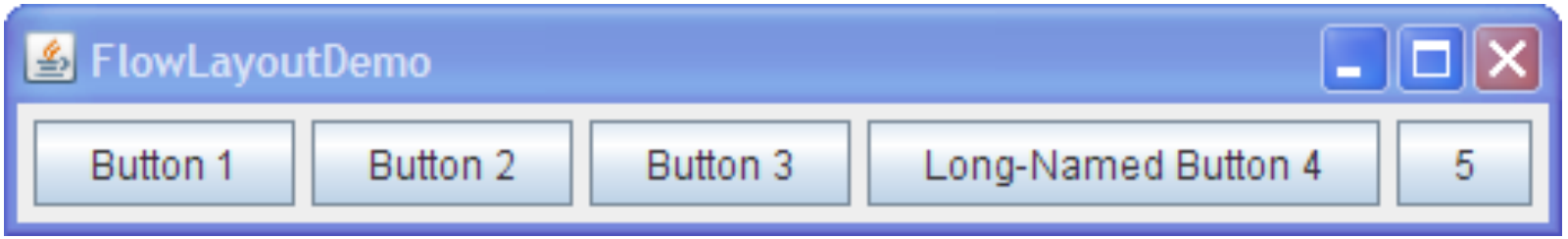
- AWT và Swing cung cấp nhiều kiểu sắp xếp bố cục (xác định vị trí và kích thước của các thành phần):
  - java.awt.BorderLayout
  - java.awt.FlowLayout
  - java.awt.GridLayout
  - java.awt.CardLayout
  - java.awt.GridBagLayout
  - javax.swing.BoxLayout
  - javax.swing.GroupLayout
  - javax.swing.ScrollPaneLayout
  - javax.swing.SpringLayout

Sắp xếp bố cục (tt)

- Sắp xếp bố cục mặc định là FlowLayout
- Cài đặt
  - `JPanel panel = new JPanel(new BorderLayout());`
  - `Container contentPane = frame.getContentPane();`
    `contentPane.setLayout(new FlowLayout());`
- Thêm 1 thành phần vào 1 container
  - `pane.add(aComponent, BorderLayout.PAGE_START);`
- Có thể dùng NetBeans để sắp xếp bố cục dễ dàng và trực quan hơn so với cấu hình bằng dòng lệnh.

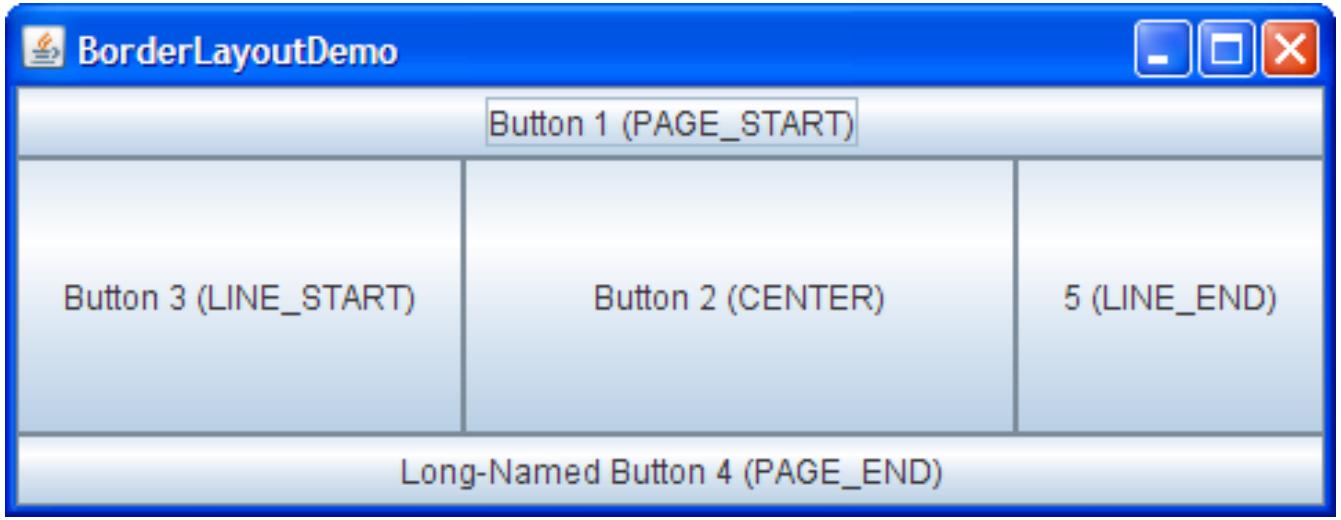
FlowLayout

- Là cách sắp xếp mặc định của mỗi JPanel
- Đặt các thành phần trong khung, nếu vượt quá chiều ngang thì sẽ chuyển xuống hàng dưới.



BorderLayout

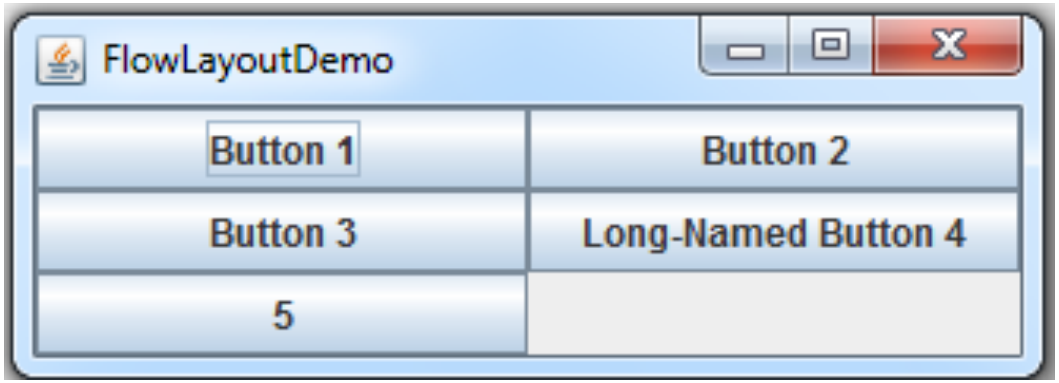
- Mỗi content pane luôn được khởi tạo với BorderLayout
- JToolBar khi tạo ra phải thuộc BorderLayout
- Có 5 vị trí: xung quanh và ở giữa.



GridLayout

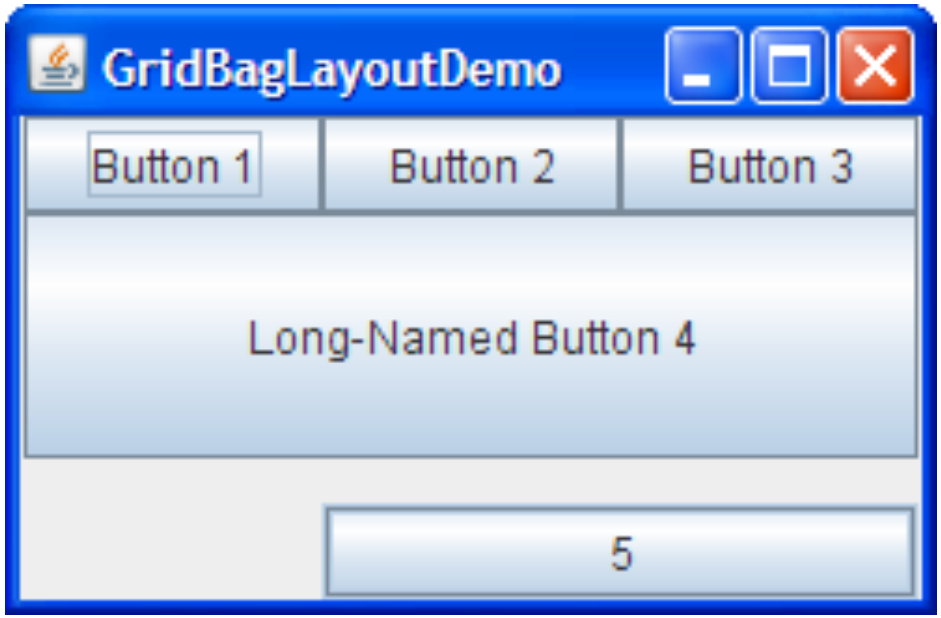
- Dạng lưới: có cùng số hàng số cột.
- Kích thước các ô trong lưới là bằng nhau.

VD: `GridLayout layout1= new GridLayout(3,2);`



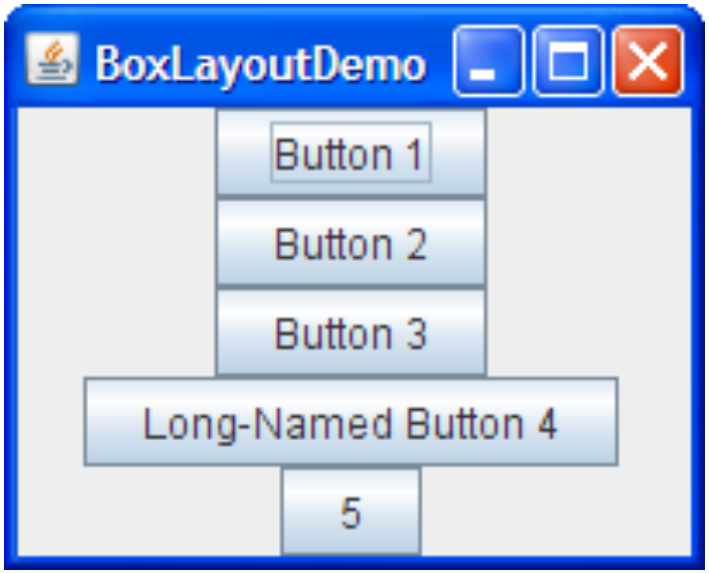
GridBagLayout

- Phức tạp, mạnh mẽ và mềm dẻo hơn.
- Dạng lưới với các hàng có thể có chiều cao khác nhau, các cột có thể có độ rộng khác nhau.



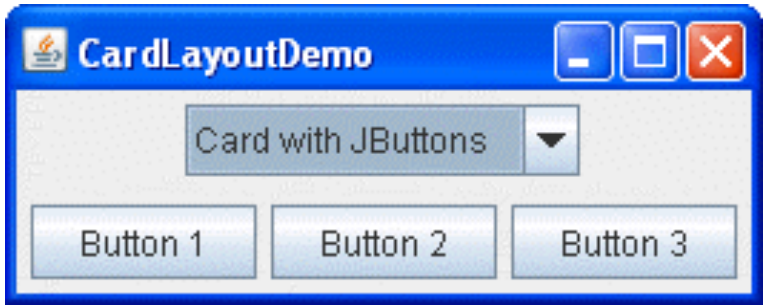
BoxLayout

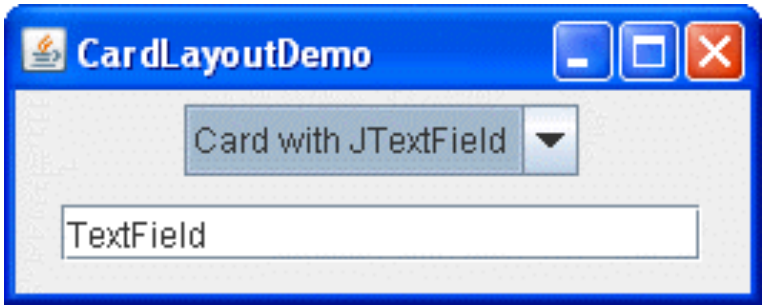
- Sắp xếp trên 1 dòng hoặc 1 cột
  - `pane.setLayout(new BoxLayout(pane, BoxLayout.Y_AXIS));`
  - `pane.setLayout(new BoxLayout(pane, BoxLayout.X_AXIS));`



CardLayout

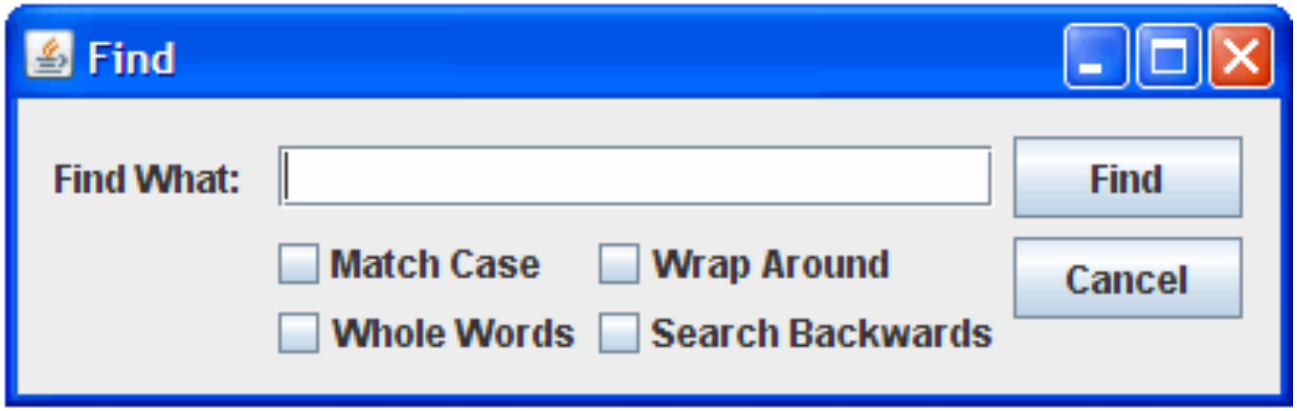
- Cài đặt 1 vùng chứa nhiều loại thành phần giao diện khác nhau tùy vào từng thời điểm.
- Điều khiển bởi 1 comboBox
- Có thể dùng kết hợp với Tabbed Pane





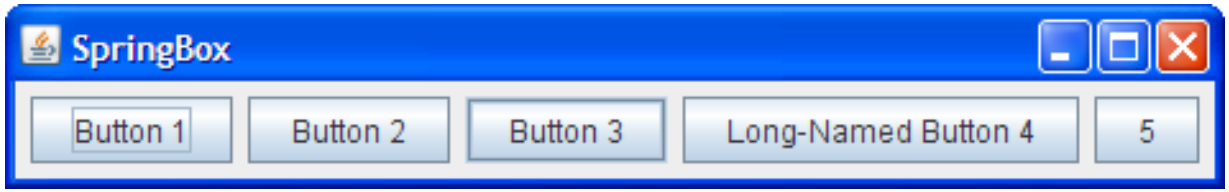
GroupLayout

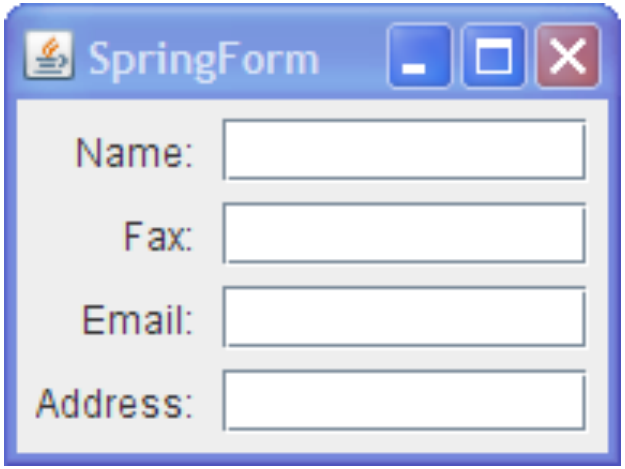
- Được phát triển cho việc sử dụng công cụ NetBeans.
- Các dòng và cột được bố trí riêng biệt.
- Linh hoạt, kích thước và vị trí mỗi thành phần độc lập.



SpringLayout

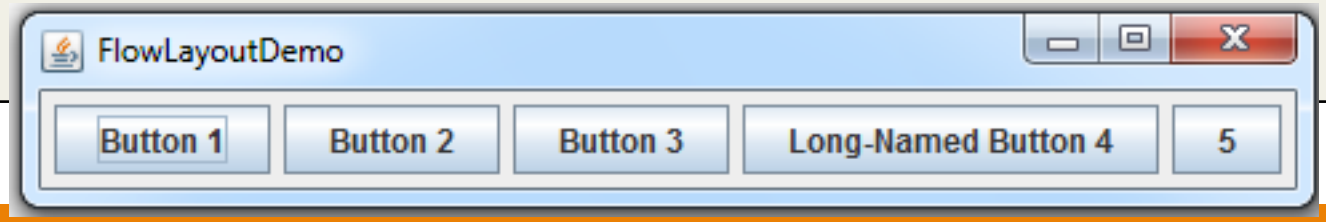
- Được đưa vào từ JDK 1.4
- Rất mềm dẻo và mạnh mẽ.
- Thích hợp cho việc sử dụng công cụ GUI Builder.





Ví dụ 1 về sắp xếp bố cục

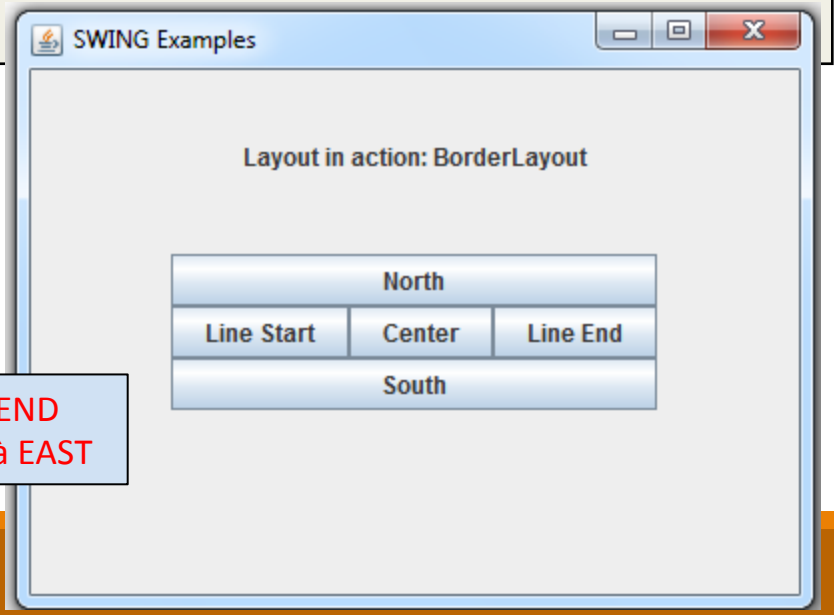
```
import java.awt.*;
import javax.swing.*;
public class FlowLayoutDemo extends JFrame{
    FlowLayout layout1 = new FlowLayout();
    public FlowLayoutDemo(String name) { super(name); }
    public void addComponentsToPane(final Container pan) {
        pan.setLayout(layout1);
        pan.add(new JButton("Button 1"));
        pan.add(new JButton("Button 2"));
        pan.add(new JButton("Button 3"));
        pan.add(new JButton("Long-Named Button 4"));
        pan.add(new JButton("5"));
        pan.setComponentOrientation(ComponentOrientation.LEFT_TO_RIGHT);
    }
    public static void main(String[] args) {
        FlowLayoutDemo frame = new FlowLayoutDemo("FlowLayoutDemo");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.addComponentsToPane(frame.getContentPane());
        frame.pack(); frame.setVisible(true);
    }
}
```



Ví dụ 2 về sắp xếp bố cục

```
import java.awt.*;
import javax.swing.*;
public class SwingLayoutDemo {
   private JFrame mainFrame;
   private JLabel headerLabel;
   private JLabel statusLabel;
   private JPanel controlPanel;
   private JLabel msglabel;
   public SwingLayoutDemo(){
      prepareGUI();
   }
   public static void main(String[] args){
      SwingLayoutDemo swlodm =
                        new SwingLayoutDemo();
      swlodm.showBorderLayoutDemo();
   }
   private void prepareGUI() {
      mainFrame = new JFrame(”SWING Examples");
      mainFrame.setSize(400,300);
      mainFrame.setLayout(new GridLayout(3, 1));
      headerLabel = new JLabel("",JLabel.CENTER );
      statusLabel = new JLabel("",JLabel.CENTER);
      statusLabel.setSize(350,50);
      controlPanel = new JPanel();
      controlPanel.setLayout(new FlowLayout());
      mainFrame.add(headerLabel);
      mainFrame.add(controlPanel);
      mainFrame.add(statusLabel);
      mainFrame.setVisible(true);
   }
```

```
private void showBorderLayoutDemo(){
    headerLabel.setText("Layout in action: BorderLayout");
    JPanel panel = new JPanel();
    panel.setBackground(Color.darkGray);
    panel.setSize(300,200);
    BorderLayout layout = new BorderLayout();
    panel.setLayout(layout);
    panel.add(new JButton("Center"),BorderLayout.CENTER);
    panel.add(new JButton("Line Start"),BorderLayout.LINE_START);
    panel.add(new JButton("Line End"),BorderLayout.LINE_END);
    panel.add(new JButton("East"),BorderLayout.EAST);
    panel.add(new JButton("West"),BorderLayout.WEST);
    panel.add(new JButton("North"),BorderLayout.NORTH);
    panel.add(new JButton("South"),BorderLayout.SOUTH);
    controlPanel.add(panel);
    mainFrame.setVisible(true);
   }
}
```



LINE_START và LINE_END
sẽ ưu tiên hơn WEST và EAST

Xử lý sự kiện (Event - handling)

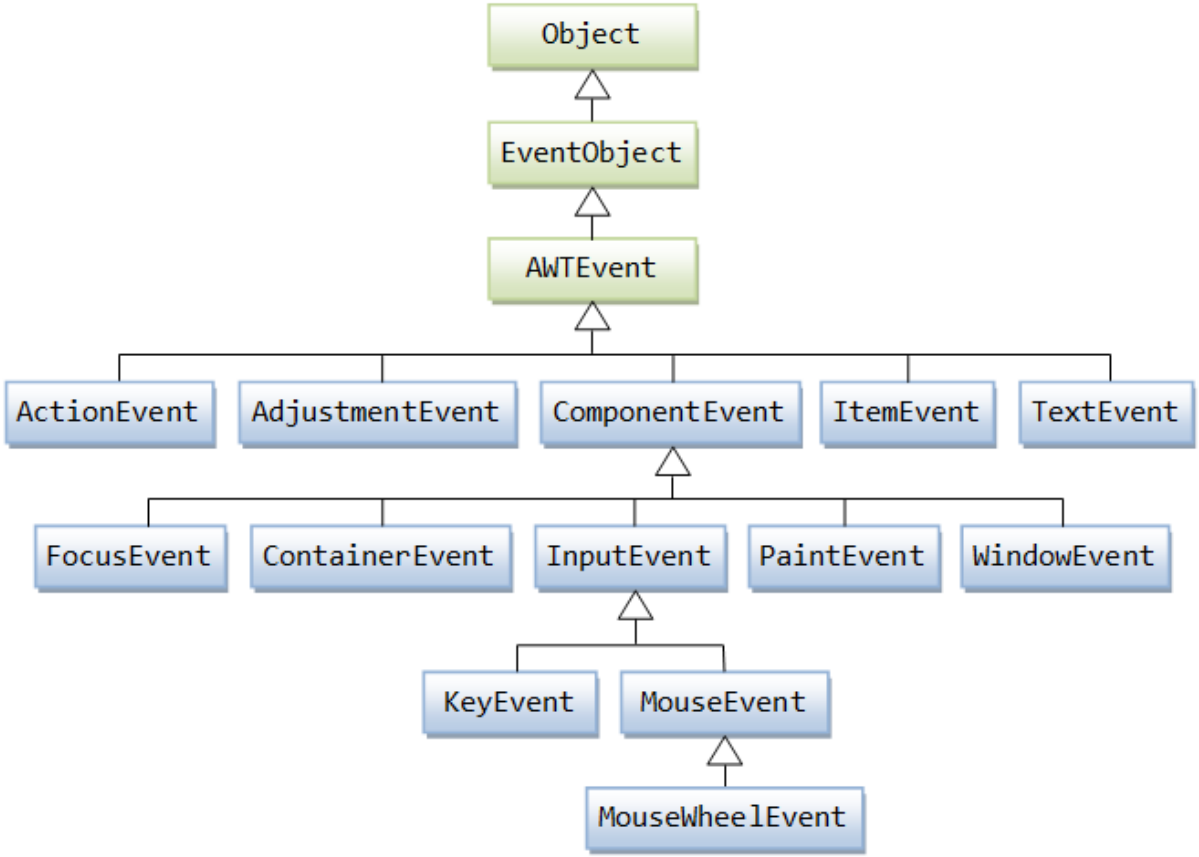
- Xử lý sự kiện bao gồm 3 đối tượng có liên quan:
  - **Đối tượng nguồn** (**source**): Button, TextField, …
  - **Sự kiện** (**event**): khi 1 đối tượng nguồn bị tác động sẽ tạo ra 1 sự kiện.

    Chẳng hạn: 1 Button được bấm, cửa sổ được đóng, …
  - **Bộ nghe** (**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.

Thao tác người dùng	Event	Event listener interface
Click JButton	ActionEvent	ActionListener
Mở, đóng JFrame	WindowEvent	WindowListener
Click 1 JComponent	MouseEvent	MouseListener
Đổi text của 1 JTextField	TextEvent	TextListener
Gõ 1 phím	KeyEvent	KeyListener
Chọn 1 mục Checkbox, …	ItemEvent, ActionEvent	ItemListener, ActionListener

Xử lý sự kiện (Event - handling)

- Các lớp xử lý sự kiện được lưu trong gói java.awt.event



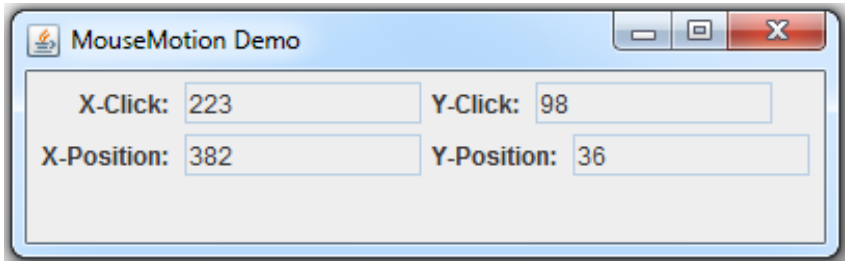
Các bước xử lý sự kiện

- Tạo lớp implements (các) bộ nghe sự kiện
  - `class ButtonClickListener implements ActionListener`
- Cho nguồn (source) đăng ký các bộ nghe
  - `okButton.addActionListener(new ButtonClickListener());`
- Tái định nghĩa các hàm xử lý sự kiện đã mô tả trong các bộ nghe sự kiện chuẩn (*VD: ActionListener*)

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

Ví dụ 1 - xử lý sự kiện

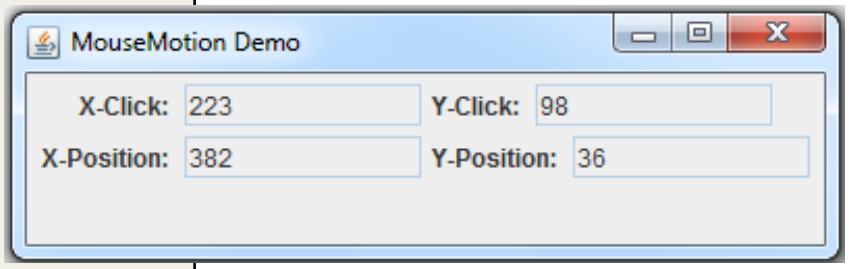
```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
public class MouseMotionDemo
             extends JFrame
      implements MouseListener,
                 MouseMotionListener {
   private JTextField tfMouseClickX;
   private JTextField tfMouseClickY;
   private JTextField tfMousePositionX;
   private JTextField tfMousePositionY;

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




```
   public MouseMotionDemo() {
      setLayout(new FlowLayout());
      add(new JLabel("X-Click: "));
      tfMouseClickX = new JTextField(10);
      tfMouseClickX.setEditable(false);
      add(tfMouseClickX);
      add(new JLabel("Y-Click: "));
      tfMouseClickY = new JTextField(10);
      tfMouseClickY.setEditable(false);
      add(tfMouseClickY);
      add(new JLabel("X-Position: "));
      tfMousePositionX = new JTextField(10);
      tfMousePositionX.setEditable(false);
      add(tfMousePositionX);
      add(new JLabel("Y-Position: "));
      tfMousePositionY = new JTextField(10);
      tfMousePositionY.setEditable(false);
      add(tfMousePositionY);
      addMouseListener(this);
      addMouseMotionListener(this);
      setTitle("MouseMotion Demo");
      setSize(400, 120);
      setVisible(true);
   }
```

Ví dụ 1 - xử lý sự kiện (tt)

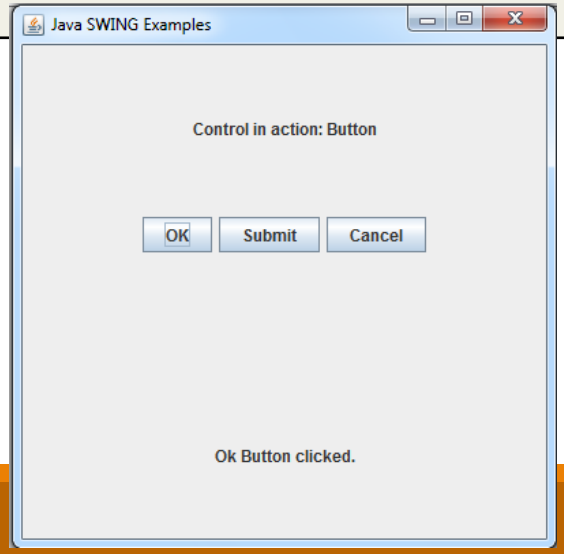
```
/** MouseListener handlers */
   // Called back when a mouse-button has been clicked
   @Override
   public void mouseClicked(MouseEvent e) {
      tfMouseClickX.setText(e.getX() + "");
      tfMouseClickY.setText(e.getY() + "");
   }
   public void mousePressed(MouseEvent e) { }
   public void mouseReleased(MouseEvent e) { }
   public void mouseEntered(MouseEvent e) { }
   public void mouseExited(MouseEvent e) { }
   /** MouseMotionEvent handlers */
   @Override
   public void mouseMoved(MouseEvent e) {
      tfMousePositionX.setText(e.getX() + "");
      tfMousePositionY.setText(e.getY() + "");
   }
   public void mouseDragged(MouseEvent e) { }
}
```



Ví dụ 2 - xử lý sự kiện

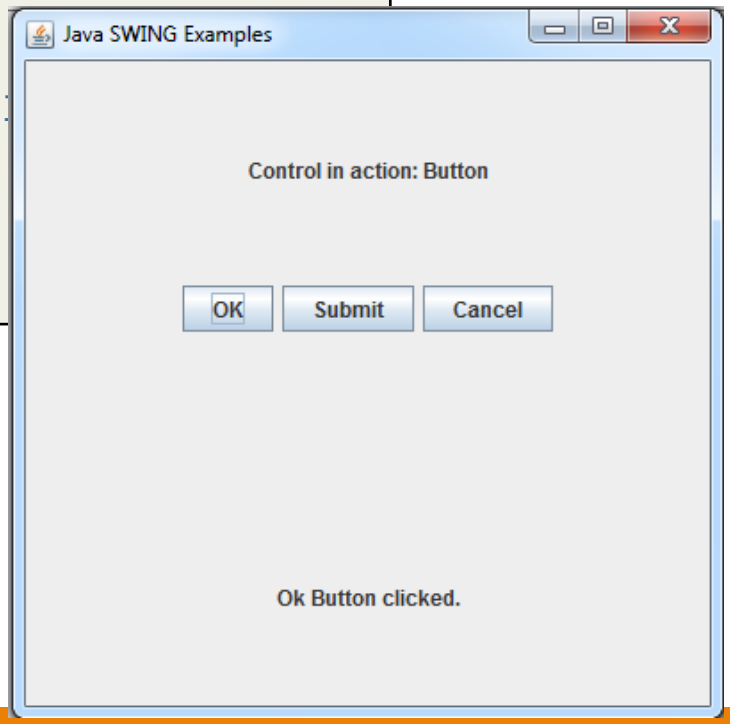
```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
public class SwingControlDemo {
   private JFrame mainFrame;
   private JLabel headerLabel;
   private JLabel statusLabel;
   private JPanel controlPanel;
   public SwingControlDemo(){ prepareGUI(); }
   public static void main(String[] args){
      SwingControlDemo scdm= new SwingControlDemo();
      scdm.showEventDemo();
   }
   private void prepareGUI(){
      mainFrame = new JFrame("Java SWING Examples");
      mainFrame.setSize(400,400);
      mainFrame.setLayout(new GridLayout(3, 1));
      headerLabel = new JLabel("",JLabel.CENTER );
      statusLabel = new JLabel("",JLabel.CENTER);
      statusLabel.setSize(350,100);
      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 showEventDemo(){
      headerLabel.setText("Control in action: Button");
      JButton okButton = new JButton("OK");
      JButton submitButton = new JButton("Submit");
      JButton cancelButton = new JButton("Cancel");
      okButton.setActionCommand("OK");
      submitButton.setActionCommand("Submit");
      cancelButton.setActionCommand("Cancel");
      okButton.addActionListener(
               new ButtonClickListener());
      submitButton.addActionListener(
               new ButtonClickListener());
      cancelButton.addActionListener(
               new ButtonClickListener());
      controlPanel.add(okButton);
      controlPanel.add(submitButton);
      controlPanel.add(cancelButton);
      mainFrame.setVisible(true);
   }
```



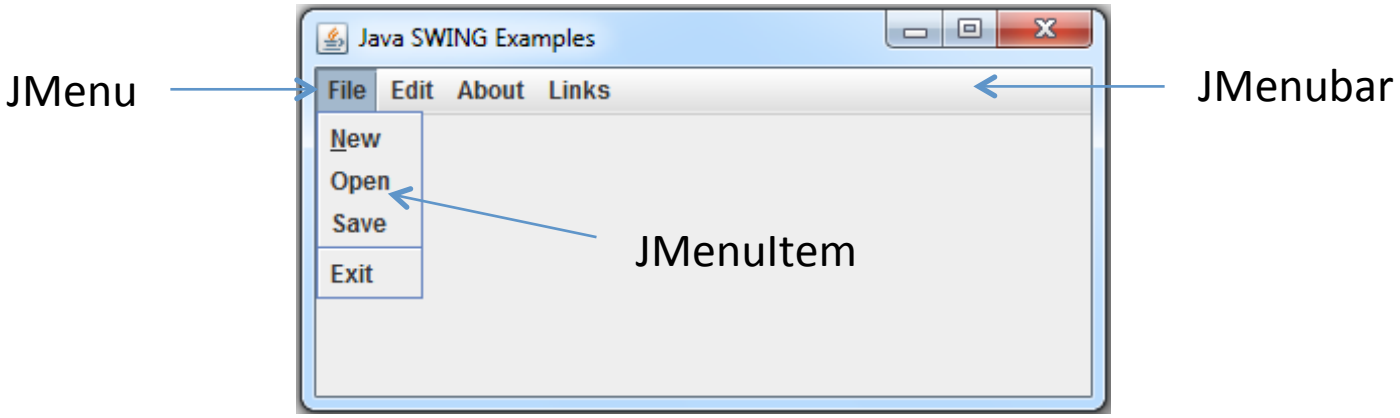
Ví dụ 2 - xử lý sự kiện (tt)

```
private class ButtonClickListener implements ActionListener {
      public void actionPerformed(ActionEvent e) {
         String command = e.getActionCommand();
         if( command.equals( "OK" ))  {
            statusLabel.setText("Ok Button clicked.");
         }
         else if( command.equals( "Submit" ) )  {
            statusLabel.setText("Submit Button clicked.");
         }
         else  {
            statusLabel.setText("Cancel Button c
         }
      }
   }
}
```



Trình đơn (Menu)

- Tạo 1 trình đơn: đối tượng JMenubar
- Gắn các đối tượng JMenu vào JMenubar.
- Gắn các JMenuItem vào Jmenu.
- Thêm JMenubar vào 1 JFrame.
- Mặc định các JMenuItem là enabled.
- Có thể vô hiệu hóa JMenuItem bằng cách gọi hàm:
  - `void setEnabled(boolean b)`



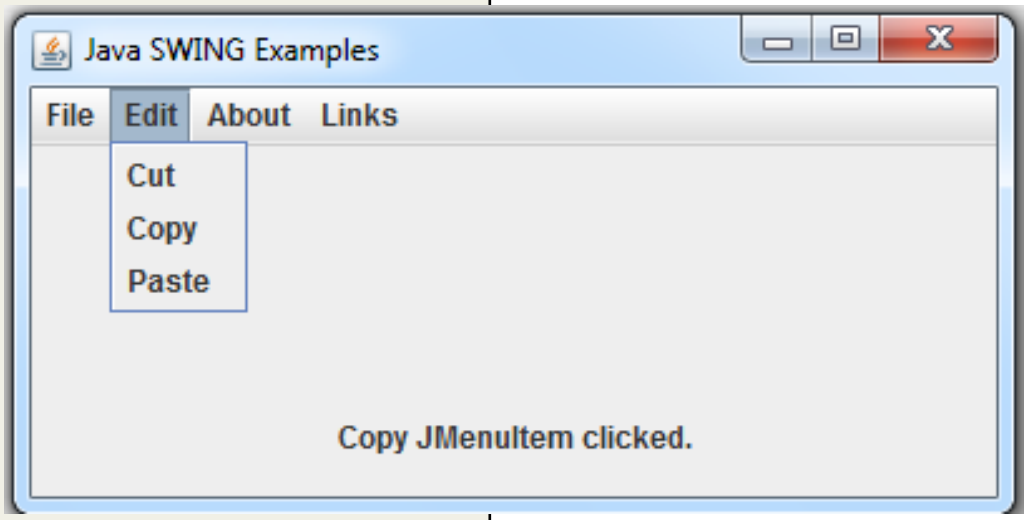
Ví dụ về Menu

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
public class SwingMenuDemo {
   private JFrame mainFrame;
   private JLabel headerLabel;
   private JLabel statusLabel;
   private JPanel controlPanel;
   public SwingMenuDemo(){ prepareGUI(); }
   public static void main(String[] args){
      SwingMenuDemo  swingMenuDemo = new SwingMenuDemo()
      swingMenuDemo.showMenuDemo();
   }
   private void prepareGUI(){
      mainFrame = new JFrame("Java SWING Examples");
      mainFrame.setSize(400,200);
      mainFrame.setLayout(new GridLayout(3, 1));
      headerLabel = new JLabel("",JLabel.CENTER );
      statusLabel = new JLabel("",JLabel.CENTER);
      statusLabel.setSize(350,100);
      mainFrame.addWindowListener(new WindowAdapter() {
         public void windowClosing(WindowEvent windowEve
            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(){
      JMenuBar menuBar = new JMenuBar();

      JMenu fileMenu = new JMenu("File");
      JMenu editMenu = new JMenu("Edit");
      JMenu aboutMenu = new JMenu("About");
      JMenu linkMenu = new JMenu("Links");

      JMenuItem newMenuItem = new JMenuItem("New");
      newMenuItem.setMnemonic(KeyEvent.VK_N);
      newMenuItem.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");
      JMenuItem cutMenuItem = new JMenuItem("Cut");
      cutMenuItem.setActionCommand("Cut");
      JMenuItem copyMenuItem = new JMenuItem("Copy");
      copyMenuItem.setActionCommand("Copy");
      JMenuItem pasteMenuItem = new JMenuItem("Paste");
      pasteMenuItem.setActionCommand("Paste");

      MenuItemListener menuItemListener =
                        new MenuItemListener();
      newMenuItem.addActionListener(menuItemListener);
      openMenuItem.addActionListener(menuItemListener);
      saveMenuItem.addActionListener(menuItemListener);
      exitMenuItem.addActionListener(menuItemListener);
      cutMenuItem.addActionListener(menuItemListener);
      copyMenuItem.addActionListener(menuItemListener);
      pasteMenuItem.addActionListener(menuItemListener);
```

Ví dụ về Menu (tt)

```
   //add menu items to menus
    fileMenu.add(newMenuItem);
    fileMenu.add(openMenuItem);
    fileMenu.add(saveMenuItem);
    fileMenu.addSeparator();
    fileMenu.add(exitMenuItem);
    editMenu.add(cutMenuItem);
    editMenu.add(copyMenuItem);
    editMenu.add(pasteMenuItem);
    //add menu to menubar
    menuBar.add(fileMenu);
    menuBar.add(editMenu);
    menuBar.add(aboutMenu);
    menuBar.add(linkMenu);
    //add menubar to the frame
    mainFrame.setJMenuBar(menuBar);
    mainFrame.setVisible(true);
  }
class MenuItemListener implements ActionListener {
      public void actionPerformed(ActionEvent e) {
         statusLabel.setText(e.getActionCommand()
         + " JMenuItem clicked.");
      }
   }
}
```

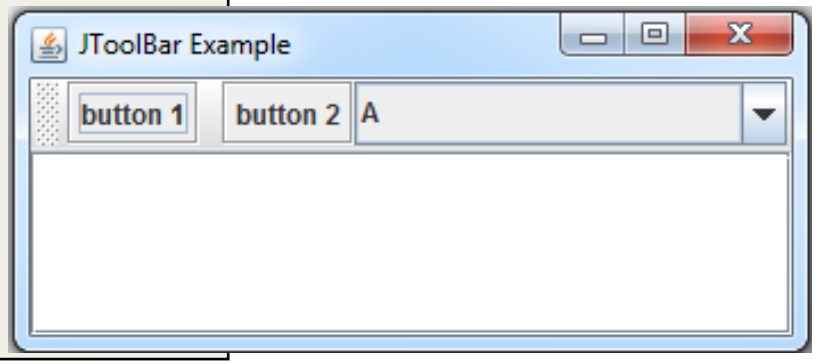


Thanh công cụ (Toolbar)

- Lớp JToolbar
- Toolbar hỗ trợ 2 dạng: ngang và đứng
- Có thể thêm các thành phần vào thanh công cụ như:
  - Nút bấm
  - ComboBox
  - Menu
- Thường gắn Toolbar vào cạnh trên BorderLayout.
- Có thể thêm vào ngăn cách bằng cách gọi hàm:
  - `toolbar.addSeparator();`

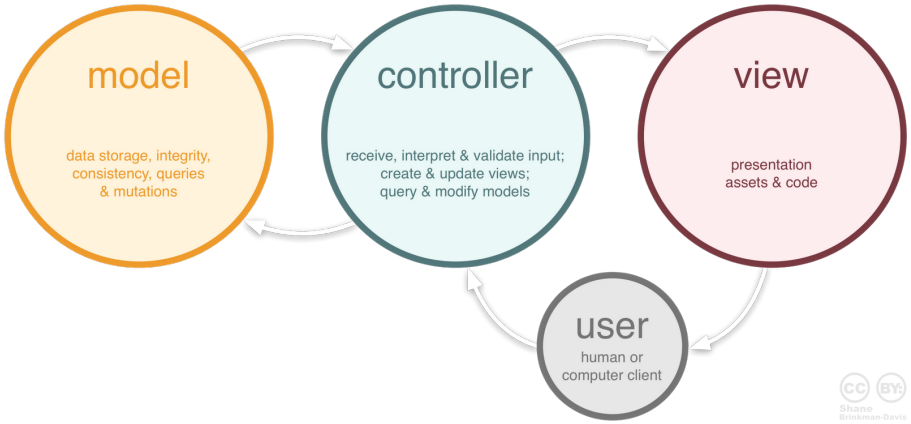
Ví dụ về thanh công cụ

```
import java.awt.*;
import javax.swing.*;

public class ToolBarSample {
  public static void main(final String args[]) {
    JFrame frame = new JFrame("JToolBar Example");
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    JToolBar toolbar = new JToolBar();
    toolbar.setRollover(true);
    JButton button = new JButton("button 1");
    toolbar.add(button);
    toolbar.addSeparator();
    toolbar.add(new JButton("button 2"));
    toolbar.add(new JComboBox(new String[]{"A","B","C"}));
    Container contentPane = frame.getContentPane();
    contentPane.add(toolbar, BorderLayout.NORTH);
    JTextArea textArea = new JTextArea();
    JScrollPane pane = new JScrollPane(textArea);
    contentPane.add(pane, BorderLayout.CENTER);
    frame.setSize(350, 150);
    frame.setVisible(true);
  }
}
```



Mô hình MVC

- Model – View – Controller: là mẫu thiết kế được áp dụng rộng rãi trong các ngôn ngữ hướng đối tượng hiện nay.
- Mục tiêu là chia tách phần Giao diện – Code để dễ quản lý, phát triển và bảo trì.
- Chia ứng dụng ra làm 3 phần:
  - Model: lớp chứa thông tin đối tượng (dữ liệu).
  - View: giao diện tương tác với người dùng.
  - Controller: Code điều khiển tương tác giữa Model-View và nghiệp vụ (Business).



Ví dụ về mô hình MVC

MODEL

```
package mvc.models;

public class Model {

    private int x;

    public Model(){
        x = 0;
    }

    public Model(int x){
        this.x = x;
    }

    public void incX(){
        x++;
    }

    public int getX(){
        return x;
    }
}
```

Ví dụ về mô hình MVC

VIEW

```
package mvc.views;
import javax.swing.*;
import java.awt.BorderLayout;
public class View {
    private JFrame frame;
    private JLabel label;
    private JButton button;
    public View(String text){
        frame = new JFrame("View");
        frame.getContentPane().setLayout(new BorderLayout());
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(200,200);
        frame.setVisible(true);
        label = new JLabel(text);
        frame.getContentPane().add(label, BorderLayout.CENTER);
        button = new JButton("Button");
        frame.getContentPane().add(button, BorderLayout.SOUTH);
    }
    public JButton getButton(){
        return button;
    }
    public void setText(String text){
        label.setText(text);
    }
}
```

Ví dụ về mô hình MVC

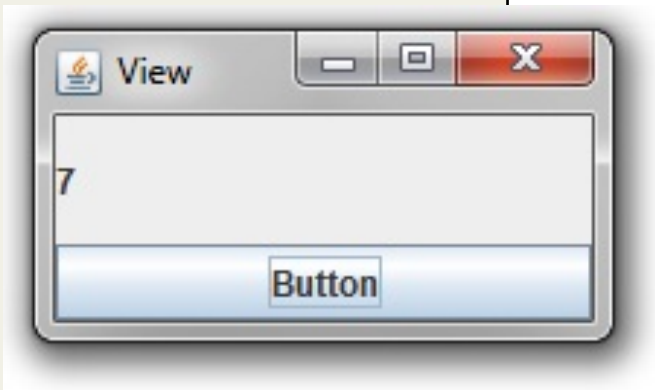
CONTROLLER

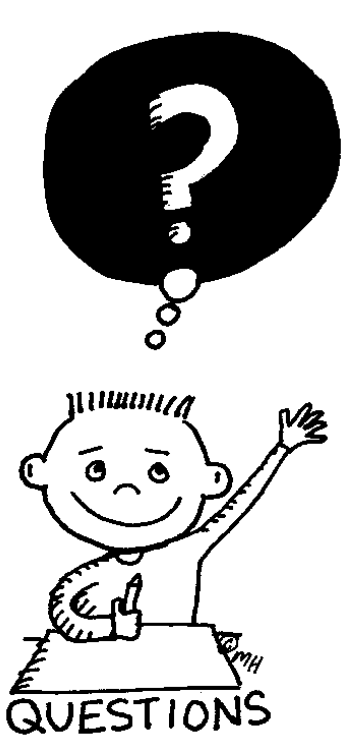
```
package mvc.controllers;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import mvc.models.*;
import mvc.views.*;
public class Controller {
    private Model model;
    private View view;
    private ActionListener actionListener;
    public Controller(Model model, View view){
        this.model = model; this.view = view;
    }
    public void control(){
        actionListener = new ActionListener() {
              public void actionPerformed(ActionEvent actionEvent) {
                  linkBtnAndLabel();
              }
        };
        view.getButton().addActionListener(actionListener);
    }
    private void linkBtnAndLabel(){
        model.incX();
        view.setText(Integer.toString(model.getX()));
    }
}
```

Ví dụ về mô hình MVC

MAIN CLASS

```
package mvc;

import javax.swing.SwingUtilities;

import mvc.models.*;
import mvc.views.*;
import mvc.controllers.*;

public class Main
{
    public static void main(String[] args) {
        SwingUtilities.invokeLater(new Runnable() {
            @Override
            public void run() {
                Model model = new Model(0);
                View view = new View("-");
                Controller controller = new Controller(model,view);
                controller.control();
            }
        });
    }
}
```





Question?

CT176 – LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG