LẬP TRÌNH JAVA

Bài 10:
Lập trình giao diện Java – Các thành phần GUI cơ bản

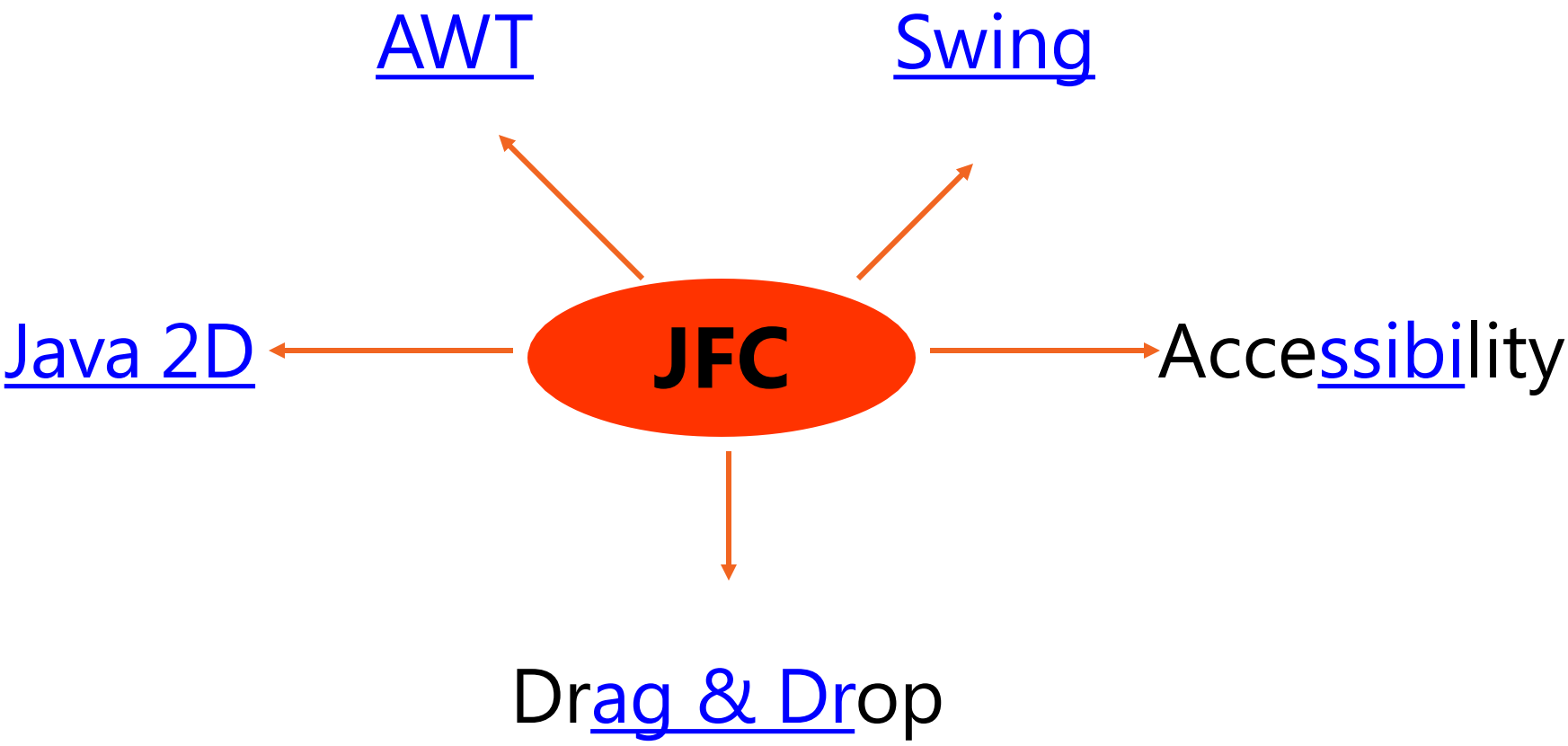
Mục tiêu bài học

- Giới thiệu gói thư viện AWT
- So sánh AWT và Swing
- Một số Swing component (JTextfield, JLabel,JButton, JCheckBox, JRadioButton,JTextArea, JPasswordField)
- Modal và Non – Modal Dialog
- Custom Dialog
- Các lớp quản lý bố cục (Layout)

Java Foundation Classes (JFC)

- FC (Foundation Classes) là một nhóm các thư viện được thiết kế để hỗ trợ lập trình viên tạo ra các ứng dụng GUI trên Java.
- FC đơn giản hóa quá trình thiết kế và làm giảm thời gian thực hiện viết mã.
- Swing chỉ là một trong năm thư viện tạo nên JFC. JFC cũng chứa Abstract Window Toolkit (AWT),Accessibility API, 2D API và tăng cường hỗ trợ khả năng kéo thả (Drag and Drog).

Java Foundation Classes



Abstract Windowing Toolkit (AWT)

AWT chứa nhiều **class** và **method** cho phép thiết kế, quản lý cửa sổ và font chữ **trên giao diện đồ họa**.

Mục đích chính của awt là **hỗ trợ cho các ứng dụng applet** nhưng cũng được dùng để thiết kế các chương trình có **giao diện đồ họa độc lập**.

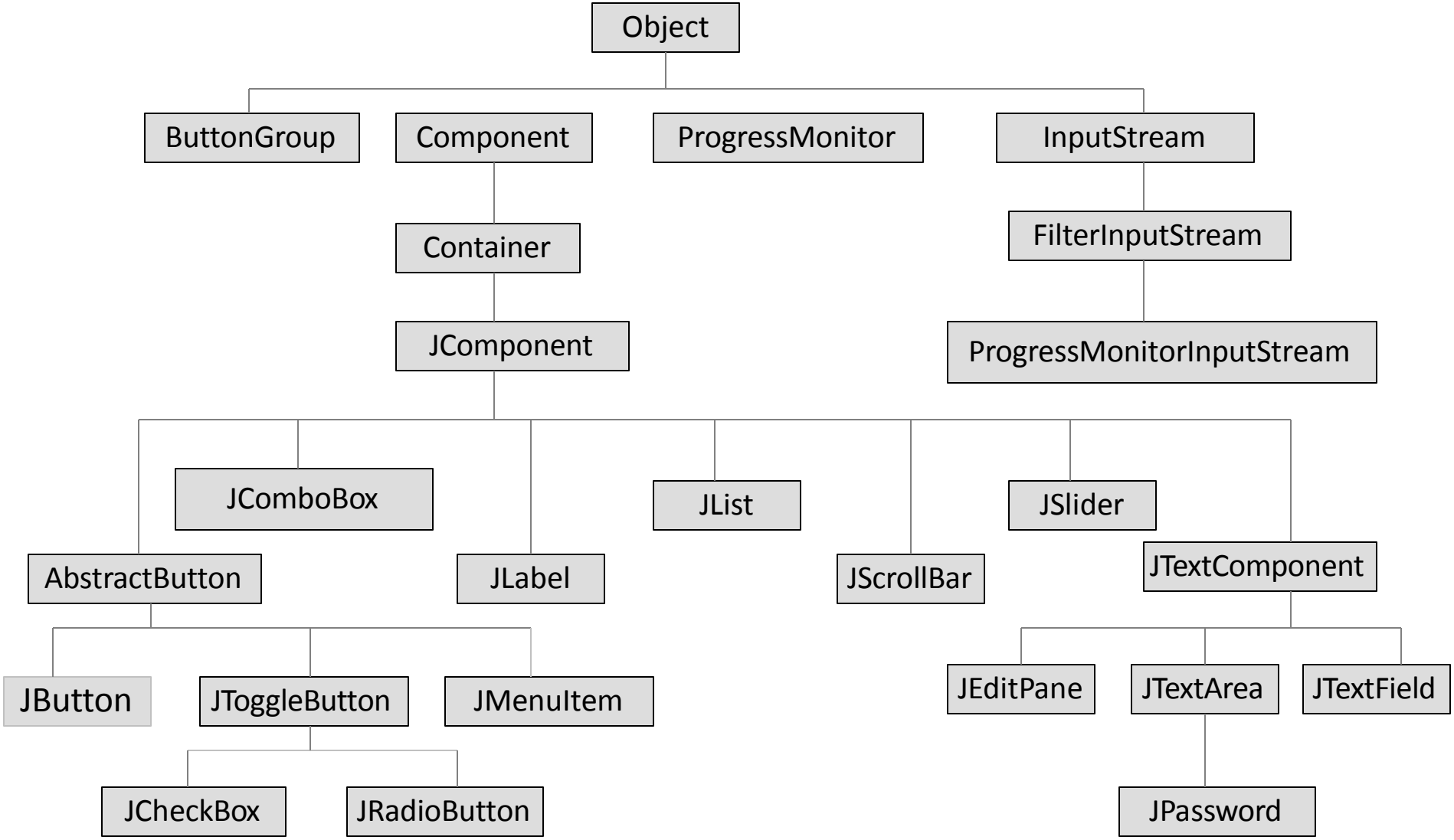
Tuy nhiên AWT có hạn chế:

- Các thành phần GUI có thể có hình dạng/hành động khác nhau trên các hệ điều hành khác nhau (heavyweight)
- Look and Feel của mỗi thành phần không thể (dễ dàng) thay đổi.

Swing

- Swing giải quyết các hạn chế liên quan đến các thành phần của AWT thông qua việc sử dụng 2 tính năng: các thành phần *lightweight* và *pluggable look and feel*.
- Các thành phần trong swing là **lightweight**: Các thành phần này được viết hoàn toàn bằng Java, do đó nó không phụ thuộc vào một hệ điều hành cụ thể nào và nó cũng rất hiệu quả và linh hoạt.
- Các class Swing có khả năng viết những cảm quan (**Look&Feels**) cho mỗi thành phần, và có thể thay đổi cảm quan vào thời điểm chạy.
- Swing có rất nhiều những thành phần mới như table, tree, slider, spinner, progress bar, internal frame và text.

Các thành phần của Swing

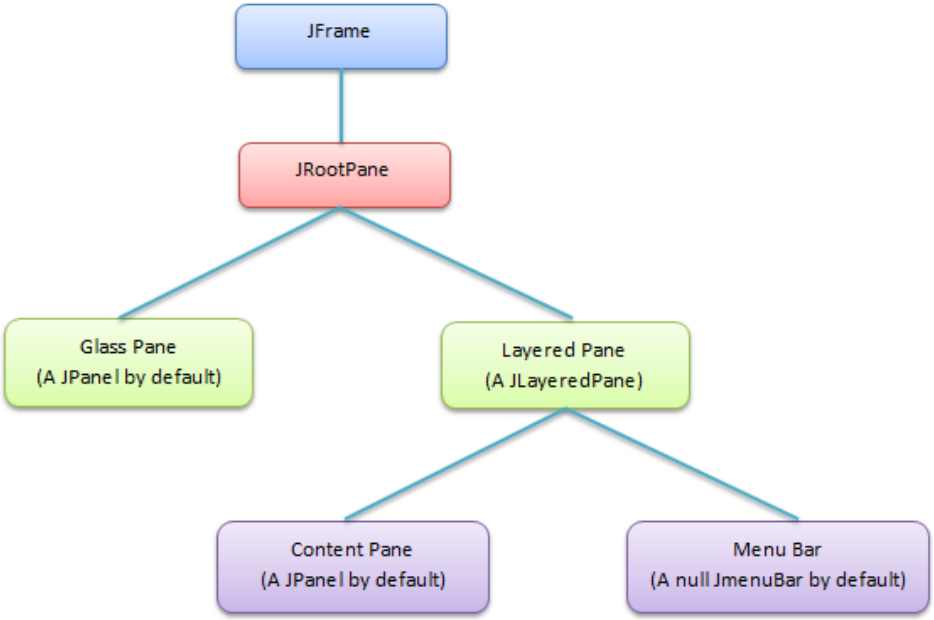


Container trong Java Swing

- Thành phần chứa trong Swing, hay còn gọi là Container
- Có 2 kiểu Container trong Swing, đó là Top-level Container và Multi-purpose Container
- Swing cung cấp cho chúng ta 3 loại Top-level Container đó là:
  - JFrame
  - JDialog
  - JApplet: được sử dụng cho ứng dụng web
  - JWindow: loại này không có đặc điểm gì cả, chỉ là một màn hình chờ được bật lên trong lúc khởi động (Splash – Screen)
- General-purpose thì gồm có: JPanel, JLayered, JInternalFrame, và JDesktopPane

JFrame là gì? Chương trình Swing đầu tiên.

- **JFrame** là một Top-level Container thường được sử dụng để tạo các giao diện ứng dụng người dùng. Tất cả các đối tượng liên quan tới **JFrame** được quản lý bởi đứa con duy nhất của nó, một thể hiện (instance) của JRootPane. JRootPane có 4 phần chính là GlassPane, LayeredPane, ContentPane và MenuBar.



JFrame là gì? Chương trình Swing đầu tiên.

- Khi thêm các thành phần (component) vào JFrame chúng ta không thêm trực tiếp, thay vào đó, phần lớn các thành phần sẽ thêm vào ContentPane bằng cách gọi phương thức:
  - getContentPane().add(component);
- Có hai cách tạo JFrame như sau:
  - **Thứ nhất: Thừa kế javax.swing.Jframe**

```
1   Import javax.swing.JFrame;
2   public class DemoJframe extends JFrame {
3       // TO DO code here
4       public DemoJframe(){
5       ……
6       ……
7       }
8       public static void main (String [] args){
9       DemoJframe objFrame = new DemoJframe();
10      ……
11      ……
12      }
13  }
14
```

JFrame là gì? Chương trình Swing đầu tiên.

- Điều kiện sử dụng cách tạo **JFrame** kiểu này là khi lớp chúng ta muốn tạo một **JFrame** không được kế thừa từ bất cứ lớp nào khác. Sự thuận lợi của cách này là chúng ta có thể gọi các phương thức của lớp Cha một cách trực tiếp
- **Thứ hai: Khai báo javax.swing.Jframe**

```
1   import javax.swing.JFrame;
2   public class DemoJframe extends ClassAbc {
3       // TO DO code here
4       ……
5       ……
6       public static void main (String [] args){
7       DemoJframe objFrame = new DemoJframe();
8       JFrame objFrame = new JFrame();     //tao Jfram khong co tieu de
9       ……
10      ……
11      }
12  }
```

JFrame là gì? Chương trình Swing đầu tiên.

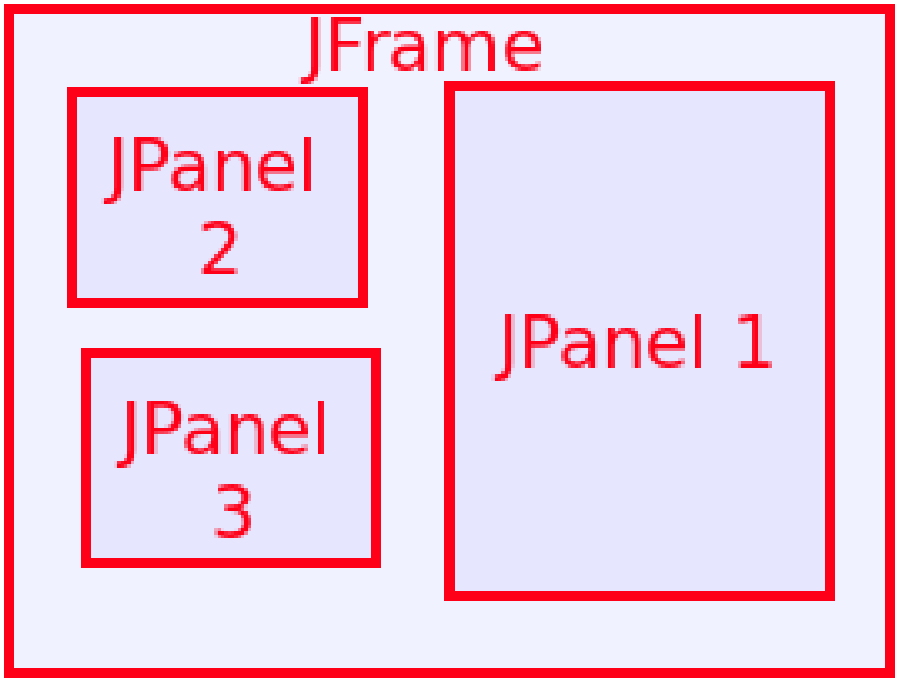
- JFrame sau khi được tạo thì chúng ta sẽ không nhìn thấy được nó. Để nhìn được nó chúng ta phải nhờ đến phương thức
  - **setVisible(boolean value)**.
- Thiết lập độ rộng và cao của JFrame.
  - **setSize(width, heigh)**
- Khi **JFrame** đã được nhìn thấy, muốn đóng lại dùng phương thức
  - **setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);**
  - Nếu không dùng hàm này để đặt, thì mặc định là: HIDE_ON_CLOSE: Khi đóng frame sẽ bị ẩn đi chứ hoàn toàn không đóng lại.
  - Các lựa chọn khác gồm:
    - DO_NOTHING_ON_CLOSE (0) – không làm gì cả
    - DISPOSE_ON_CLOSE (2) – Chỉ đóng frame đó, các frame khác liên quan sẽ không bị đóng.
    - EXIT_ON_CLOSE (3) – Đóng toàn bộ các frame liên quan tới nó.

JFrame là gì? Chương trình Swing đầu tiên.

- Đặt vị trí xuất hiện của JFrame trên màn hình.
  - **setLocation(x,y)**
- Đặt JFrame xuất hiện chính giữa màn hình
  - **setLocationRelativeTo(null)**
- Đặt kích thước Jframe vừa đủ với nội dung
  - **pack()**
- Đặt JFrame có thể được thay đổi kích thước hay không,mặc định là True.
  - **setResizable(boolean)**
- Đặt màu nền cho JFrame
  - **getContentPane().setBackground(Color.”Color”)**

JPanel là gì?

- JPanel là một container dùng để chứa các thành phần đồ họa khác (tương tự như JFrame tuy nhiên nó không phải là 1 JFrame).
- Trong một JFrame chứa các JPanel, trong mỗi JPanel lại có thể chứa các đối tượng hoặc thậm chí là các JPanel khác.



JPanel là gì?

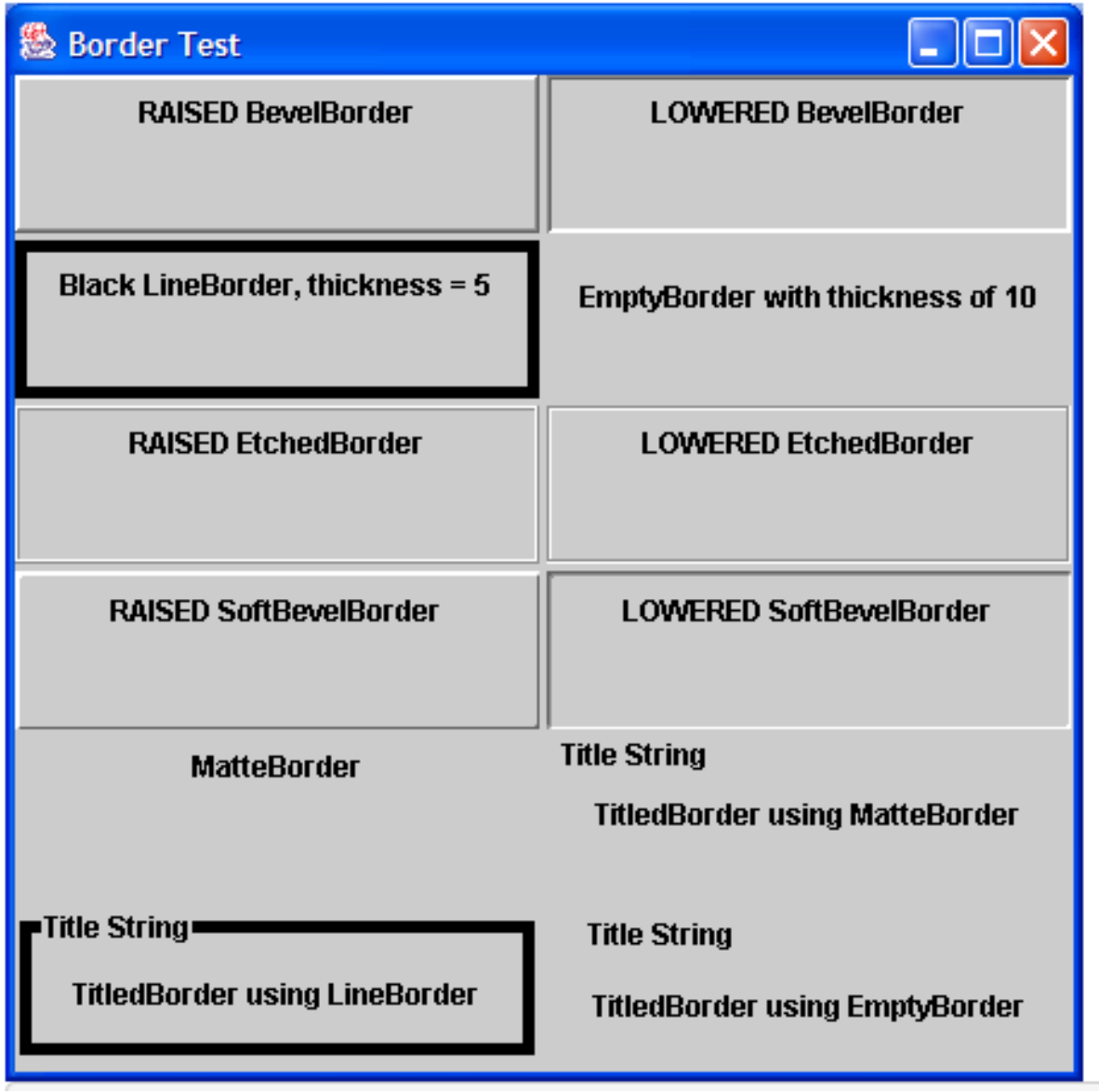
- Chúng ta có 2 Phương thức khởi tạo JPanel đó là:
  - JPanel(): Tạo 1 JPanel với Layout **mặc định** là FlowLayout(cách bố trí mà các đối tượng nối tiếp nhau).
  - JPanel(LayoutManager layout): Tạo 1 JPanel với Layout được **chỉ định**.
- Ví dụ:

```
// create panel1 with layout default is FlowLayout
JPanel panel1 = new JPanel();


// create panel2 with GridLayout
JPanel panel2 = new JPanel(new GridLayout());
```

JPanel là gì?

- Đặt Layout cho JPanel
  - **setLayout(“layout”)**
- Thiết lập vị trí và kích thước cho JPanel
  - **setBound(x,y,width,heigh)**
- Đặt màu nền cho JPanel
  - **setBackground(Color.”color”)**
- Đặt đường viền cho JPanel
  - Có nhiều kiểu đường viền
  - Sử dụng phương thức **setBorder(new “kiểu-border”)**
  - Cùng xem ví dụ

JPanel là gì?


Border Test
RAISED BevelBorder
LOWERED BevelBorder
Black LineBorder, thickness = 5
EmptyBorder with thickness of 10
RAISED EtchedBorder
LOWERED EtchedBorder
RAISED SoftBevelBorder
LOWERED SoftBevelBorder
MatteBorder
Title String
TitledBorder using MatteBorder
Title String
TitledBorder using LineBorder
Title String
TitledBorder using EmptyBorder

JPanel là gì?

```
p.setBorder(new BevelBorder(BevelBorder.RAISED));

p.setBorder(new BevelBorder(BevelBorder.LOWERED));

p.setBorder(new LineBorder(Color.black, 5));

p.setBorder(new EmptyBorder(10, 10, 10, 10));

p.setBorder(new EtchedBorder(EtchedBorder.RAISED));

p.setBorder(new EtchedBorder(EtchedBorder.LOWERED));

p.setBorder(new SoftBevelBorder(SoftBevelBorder.RAISED));

p.setBorder(new SoftBevelBorder(SoftBevelBorder.LOWERED));

p.setBorder(new MatteBorder(new ImageIcon("BALL.GIF")));

p.setBorder(new TitledBorder(new MatteBorder(new ImageIcon("java2sLogo.gif")), "Title String"));

p.setBorder(new TitledBorder(new LineBorder(Color.black, 5),"Title String"));

p.setBorder(new TitledBorder(new EmptyBorder(10, 10, 10, 10),"Title String"));
```

JComponent

- JComponent là lớp cơ sở abstract của tất cả các Swing components (có tên bắt đầu chữ J - ngoại trừ top-container)
- Kế thừa từ Component và Container
- Đóng gói hầu hết các đặc điểm và thao tác cơ bản của 1 thành phần GUI
- Lớp JComponent
  - Cảm quan khả kiến (Pluggable)
  - Phím tắt (tính dễ nhớ)
  - Khả năng xử lý sự kiện chung

JComponent

Bao gồm:

- JButton, JList, JLabel, JTextField,JTextArea
- JComboBox, JRadioButton,
- JCheckBox… extend functionality existing in AWT Components.
- JProgressBar, JSlider, JTable, JToolBar, JTree … provide new components.
- JInternalFrame, JScrollPane,
- JSplitPane, JTabbedPane … provide new ways to combine components.

JTextField

- Để hiển thị dữ liệu, nhập dữ liệu
  Khởi tạo
- JTextField(): text field trống
- JTextField(int): A text field with the specified width
- JTextField(String): A text field with text
- JTextField(String, int): A text field with the specified text and width
- JTextField(int cols): khởi tạo JtextField với số cột quy định.
- JTextField(String text, int cols): khởi tạo JTextField với dòng text và số cột quy định.

JTextField

Ví dụ:

- JTextField textField = new JTextField(20);
  Và sau đó setText():
  textField.setText("Hello!");

JTextField

- Methods:
  - void setText(String t) set Text in code behind for JTextField
  - String getText()

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

  - void setEditable(boolean b): chỉnh sửa nội dung
  - setColumns()
  - setFont

```
textField.setFont(new Font("Arial", Font.BOLD,12));
```

  - requestFocus();
- Events:
  - –caretUpdate

JLabel

- JLabel thường được dùng để hiển thị text hoặc hình ảnh để tạo các chỉ dẫn, hướng dẫn trên giao diện người dùng.
  Khai báo:
- Label() : An empty label
- JLabel(String) : A label with the specified text
- JLabel(String, int) : A label with the specified text and alignment LEFT, CENTER, and RIGHT.
- JLabel(String, Icon, int) : A label with the specified text, icon, and Alignment

JLabel

- Methods:
  - void setFont (Font f)
  - void setText(String S) quy định chuỗi văn bản.
  - String getText()
  - void setIcon(Icon) quy định Icon
  - getLength(): đưa ra chiều dài của chuỗi text.
- Events:
- mouseClicked

JLabel

- Example:
  - JLabel lbl=new JLabel("Họ và tên:");
  - JLabel lbl=new JLabel("Ngày sinh:");
- Sử dụng HTML để tạo ra các JLabel nhiều dòng, nhiều định dạng

```
JLabel lbHoten = new JLabel("<html>Dòng 1<p
style=\"color:red;font-size:20\">Dòng
2</p></html>");
```

JButton

- là một đối tượng mà cho phép chúng ta khi click chuột vào sẽ thực hiện một việc gì đó
- Khai báo
  - JButton() Creates a button with no set text or icon.
  - JButton(Action a) Creates a button where properties are taken from the Action supplied.
  - JButton(Icon icon) Creates a button with an icon.
  - JButton(String text) Creates a button with text.
  - JButton(String text, Icon icon) Creates a button with initial text and an icon.

JButton

- Methods:
  - setText (String text)
  - getText ()
  - setForeground (Color fg)
  - setFocusCycleRoot (boolean b)
- Events
  - actionPerformed
  - mousePressed

JButton

Các cách tạo và bắt sự kiện JButton

```
JButton bt=new JButton("Watch");
bt.setIcon(new ImageIcon("mywatch.png"));
bt.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent arg0)
{
    //do something here
}
});
```

JButton

Ví dụ

```
14   public TestButtonIcons() {
15     ImageIcon usIcon = new ImageIcon("image/usIcon.gif");
16     ImageIcon caIcon = new ImageIcon("image/caIcon.gif");
17     ImageIcon ukIcon = new ImageIcon("image/ukIcon.gif");
18
19     JButton jbt = new JButton("Click it", usIcon);
20     jbt.setPressedIcon(caIcon);
21     jbt.setRolloverIcon(ukIcon);
22
23     add(jbt);
24   }
25 }
```

(a) Default icon (b) Pressed icon (c) Rollover icon

JCheckBox

- là đối tượng cho phép chúng ta chọn nhiều thuộc tính.
- Ví dụ như khi điền thông tin một người xem có tiền, có nhà, có xe hơi không chẳng hạn. Người đó có thể có cả 3 hoặc không có một cái nào cả
- Khai báo
  - JCheckBox() Creates an initially unselected check box button with no text, no icon.
  - JCheckBox(Action a) Creates a check box where properties are taken from the Action supplied.
  - JCheckBox(Icon icon) Creates an initially unselected check box with an icon.
  - JCheckBox(Icon icon, boolean selected) Creates a check box with an icon and specifies whether or not it is initially selected.
  - JCheckBox(String text) Creates an initially unselected check box with text.
  - JCheckBox(String text, boolean selected)
  - JCheckBox(String text, Icon icon)
  - JCheckBox(String text, Icon icon, boolean selected)

JCheckBox

- Methods:
  - –setSelected (boolean b)
  - –isSelected ()
  - –setText (String text)
  - –getText ()
- Events:
  - –actionPerformed

JCheckBox

Ví dụ:

```
panelCheck=new JPanel();
panelCheck.setLayout(new GridLayout(2, 2));
JCheckBox chk1=new JCheckBox("C/C++");
JCheckBox chk2=new JCheckBox("C#");
JCheckBox chk3=new JCheckBox("PHP");
JCheckBox chk4=new JCheckBox("Java");
panelCheck.add(chk1);
panelCheck.add(chk2);
panelCheck.add(chk3);
panelCheck.add(chk4);
```

JCheckBox

JRadioButton

- là một đối tượng cho phép chúng ta chọn lựa các thuộc tính giống như JCheckBox. Tuy nhiên chúng ta hay sử dụng JRadioButton khi mà muốn người dùng chỉ chọn được một trong các thuộc tính.
- Ví dụ chọn giới tính thì người dùng chỉ được chọn là Nam hoặc Nữ.
- Khai báo
  - JRadioButton() Creates an initially unselected radio button with no set text.
  - JRadioButton(Action a) Creates a radiobutton where properties are taken from the Action supplied.
  - JRadioButton(Icon icon) Creates an initially unselected radio button with the specified image but no text.
  - JRadioButton(Icon icon, boolean selected) Creates a radio button with the specified image and selection state, but no text.
  - JRadioButton(String text) Creates an unselected radio button with the specified text.
  - JRadioButton(String text, boolean selected)
  - JRadioButton(String text, Icon icon) Creates a radio button that has the specified text and image, and that is initially unselected.
  - JRadioButton(String text, Icon icon, boolean selected)
- Must add JRadioButton into the ButtonGroup

JRadioButton

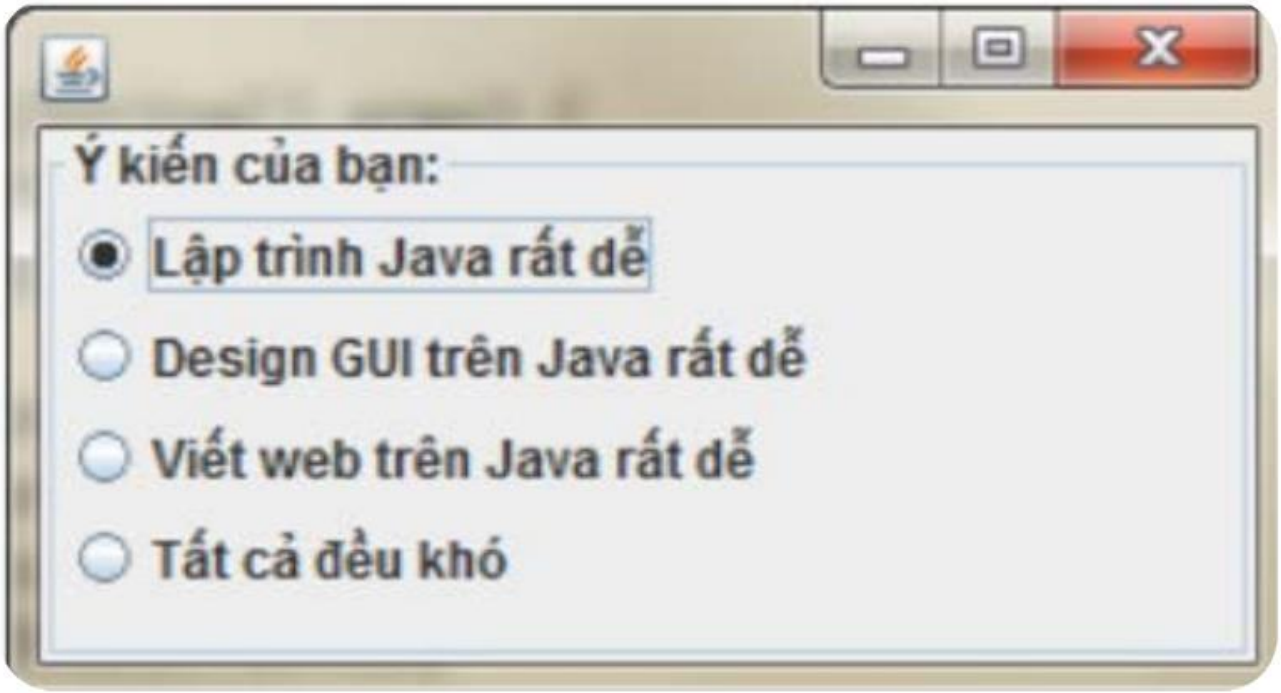
- Methods:
  - setSelected (boolean b)
  - isSelected ()
  - setText (String text)
  - getText ()
- Events:
  - actionPerformed

JRadioButton

Ví dụ:

```
JPanel panelGroup=new JPanel();
panelGroup.setBorder(new TitledBorder("Ý kiến của bạn:"));
JRadioButton rad1=new JRadioButton("Lập trình Java rất dễ");
JRadioButton rad2=new JRadioButton("Design GUI Java rất dễ");
JRadioButton rad3=new JRadioButton("Viết web trên Java rất dễ");
JRadioButton rad4=new JRadioButton("Tất cả đều khó");
ButtonGroup group=new ButtonGroup();
group.add(rad1);
group.add(rad2);
group.add(rad3);
group.add(rad4);
panelGroup.add(rad1);
panelGroup.add(rad2);
panelGroup.add(rad3);
panelGroup.add(rad4);
add(panelGroup);
```

JRadioButton



JTextArea

- là một thành phần cho phép hiển thị nhiều dòng văn bản đồng thời người dùng có thể chỉnh sửa văn bản
- Khai báo
- JTextArea(): Create a default text area.
- JTextArea(int rows, int columns): Create a text area with the specified number of rows and columns.
- JTextArea(String text)
- JTextArea(String text, int rows, int columns)
- JTextArea(Document doc): Create a text area that uses the specified Document.
- JTextArea(Document doc, String text, int rows, int columns)

JTextArea

- Methods
- setWrapStyleWord(true);
- setLineWrap(true);
- setText("");
- setFont(font);
- setForeground(Color.BLUE);
- void append(String str)
- Append the given text to the end of the document.
- void insert(String str, int pos)
- Insert the specified text at the given position . To insert text at the beginning of the document, use a position of 0.

JTextArea

- void replaceRange(String str, int start, int end)
- Replace a section of the document
- public int getLineStartOffset(int line) throws BadLocationException
- Return the character offset (from the beginning) that marks the beginning of the specified line number.
- public int getLineEndOffset(int line) throws BadLocationException
- Return the character offset (from the beginning) that marks the end of the specified line number. This is actually the offset of the first character of the next line.
- public int getLineOfOffset(int offset) throws BadLocationException
- Return the line number that contains the given character offset (from the beginning of the document).
- - Kết hợp với thanh cuộn:
  textArea = new JTextArea(8, 40);
  JScrollPane scrollPane = new JScrollPane(textArea);

JTextArea

Ví dụ

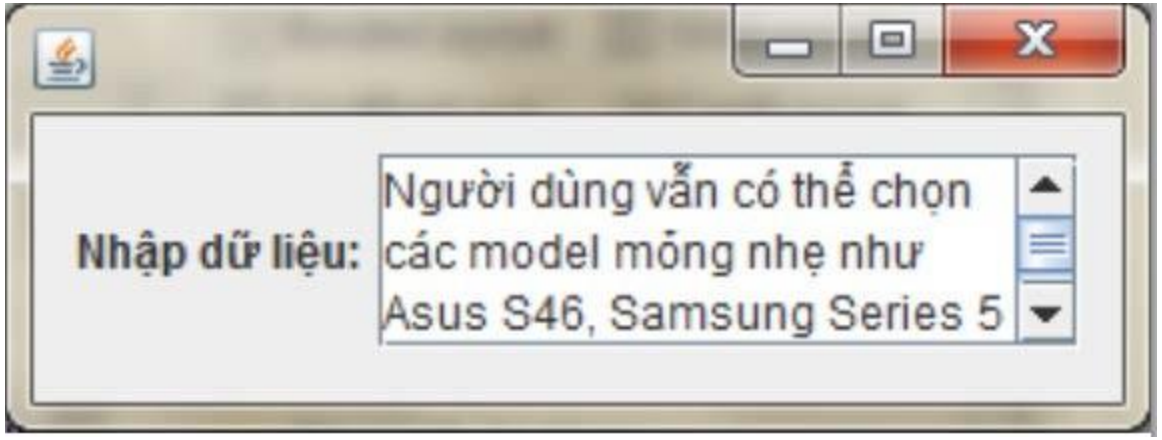
```
JPanel contentPane = new JPanel();
 JLabel lblNewLabel = new JLabel("Nhập dữ liệu:");
 contentPane.add(lblNewLabel);
JTextArea textArea = new JTextArea(3,15);
 textArea.setWrapStyleWord(true);
 textArea.setLineWrap(true);
 JScrollPane scrollPane = new JScrollPane(textArea);
contentPane.add(scrollPane);
 JTextArea class MyJTextArea extends JFrame {
private JPanel contentPane;
public MyJTextArea() {
setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
setBounds(100, 100, 257, 128);
contentPane = new JPanel();
```

JTextArea

Ví dụ

```
contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
 setContentPane(contentPane);
contentPane.setLayout(new FlowLayout(FlowLayout.CENTER, 5, 5));
 JLabel lblNewLabel = new JLabel("Nhập dữ liệu:");
 contentPane.add(lblNewLabel);
JScrollPane scrollPane = new JScrollPane();
JTextArea textArea = new JTextArea(3,15);
textArea.setWrapStyleWord(true);
textArea.setLineWrap(true);
scrollPane.setViewportView(textArea);
contentPane.add(scrollPane);
 }
```

JTextArea


Nhập dữ liệu:
Người dùng vẫn có thể chọn
các model mỏng nhẹ như
Asus S46, Samsung Series 5

JPasswordField

- là đối tượng cho phép chúng ta nhập vào một dòng text giống như JTextField nhưng được ẩn bởi các dấu sao (*) hoặc chấm tròn để tạo nên mật khẩu (password)
- JPasswordField class, a subclass of JTextField.
- JPasswordField constructor methods take the same arguments as
- those of its parent class.
- Khởi tạo
  - JPasswordField(): Khởi tạo JPasswordField không có text và độ rộng là 0 cột
  - JPasswordField(int columns): Khởi tạo JPasswordField không có text và độ rộng là columns cột
  - JPasswordField(String text): Khởi tạo JPasswordField vớit text ban đầu
  - JPasswordField(String text, int columns): Khởi tạo JPasswordField vớit text ban đầu và rộng columns cột

JPasswordField

- Methods
- JPasswordField(String text, int columns)
- char[] getPassword(): returns the text contained in this password field JPasswordField setEchoChar(char): replacing each input character with the specified character
- JPasswordField pass = new JPasswordField(20);
- pass.setEchoChar('#');

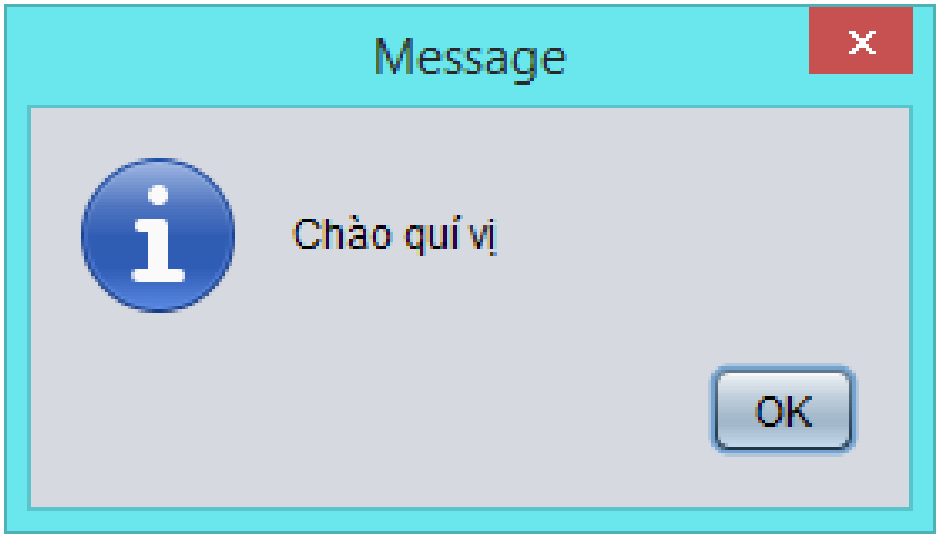
JDialog

- Hộp thoại thường là cửa sổ phụ trình bày danh sách tùy chọn hoặc hiển thị thông điệp
- Hộp thoại thường cung cấp nút xác nhận thay đổi hoặc trường nhập vào thành phần, hủy bỏ thay đổi...
- Có 3 dạng hộp thoại:
- Hộp thoại thông điệp
- Hộp thoại xác nhận
- Hộp thoại tập tin (JFileChooser)
- JDialog có 2 trạng thái
  - Modal: Khi Jdialog thực hiện xong mới được phép thao tác lên form cha .
  - Modeless: Sau khi hiển thị dialog, người dùng có thể thao tác lên form cha
- JDialog thường được sử dụng với trạng thái Modal
- Khởi tạo
- public JDialog()
- public JDialog(Dialog owner, String title, boolean modal, GraphicsConfiguration gc)
- public JDialog(Frame owner, String title, boolean modal, GraphicsConfiguration gc)

JDialog

Ví dụ:

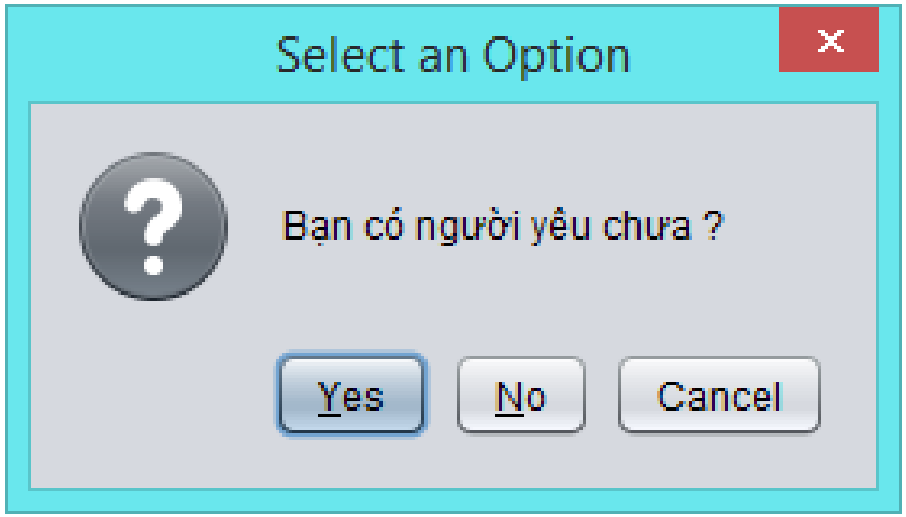
JOptionPane.showMessageDialog(this,"Chào quí vị");



JDialog

Ví dụ 2:

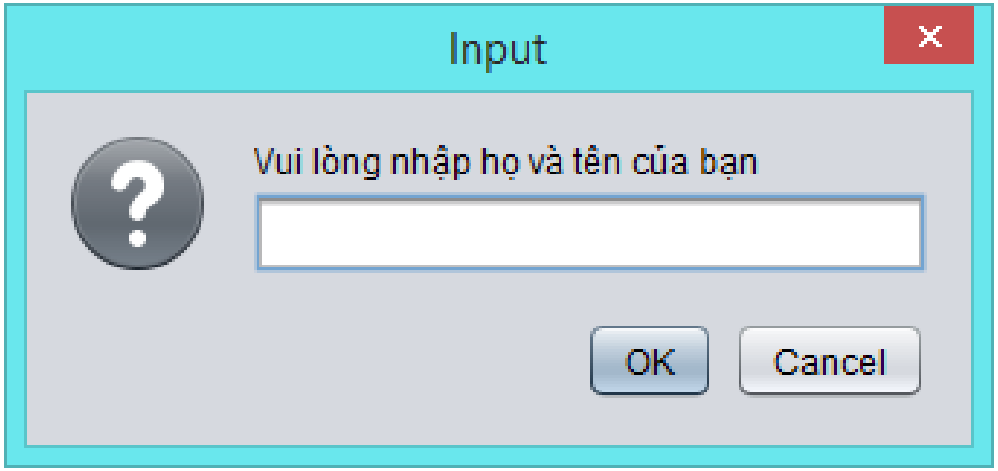
```
int choice=JOptionPane.showConfirmDialog(this,"Bạn có người yêu chưa?");
If(JOptionPane.YES_OPTION==choice) {
//do something
}
else if(JOptionPane.NO_OPTION=choice){
//do something
}
else{


}
```



JDialog

Ví dụ 3

String input =JOptionPane.showInputDialog(this,"Vui lòng nhập họ và tên của bạn");



JDialog

Methods

Property	Data type	get	is	set	Default value
acce3SibleContext°	AccessibleContext				JDialog.AccessibleJDialog( )
contentPane0	Container				From rootPane
defaultCloseOperation	int				HIDE_ON_CLOSE
defaultLookAndFeelDecorateds' 1-4	boolean				Depends on L&.F, often false
glassPane®	Component				From rootPane
JMenuBar°	JMenuSar				null
layeredPane0	JLayeredPane				From rootPane
layout0	LayoutManager				BorderLayout
modal*	boolean				false
parent*	Container				SwingUtilities.get-SharedOwnerFrame( )
rootPane	JRootPane				JRootPane
title*	String				n n

Dialog tùy biến

```
public class LoginJDialog extends javax.swing.JDialog{
}

LogginJDialog dialog=new LoginJDialog()
dialog.setDefaultCloseOperation(JDialog.DIPOSE_ON_CLOSE);
dialog.setTitle(“Đăng nhập”);
dialog.setAlwaysOnTop(true);
dialog.setIconImage(new
ImageIcon(getClass().getResource(“/icons/Login.png”)).getImage()
```

Dialog tùy biến

Layout Manager

- Một Container là một Component có thể chứa các Component khác
- JFrame, JDialog, JScollPane, Jpanel, JDesktopPane, JInternalFrame
- getContentPane().add để thêm Component vào Container
- 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.

FlowLayout


FlowLayout
Button 0
Button 1
Button 2
Button 3
Button 4
Button 5
Button 6
Button 7
Button 8
Button 9

FlowLayout

- 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

FlowLayout

- 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

FlowLayout

- 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 ()

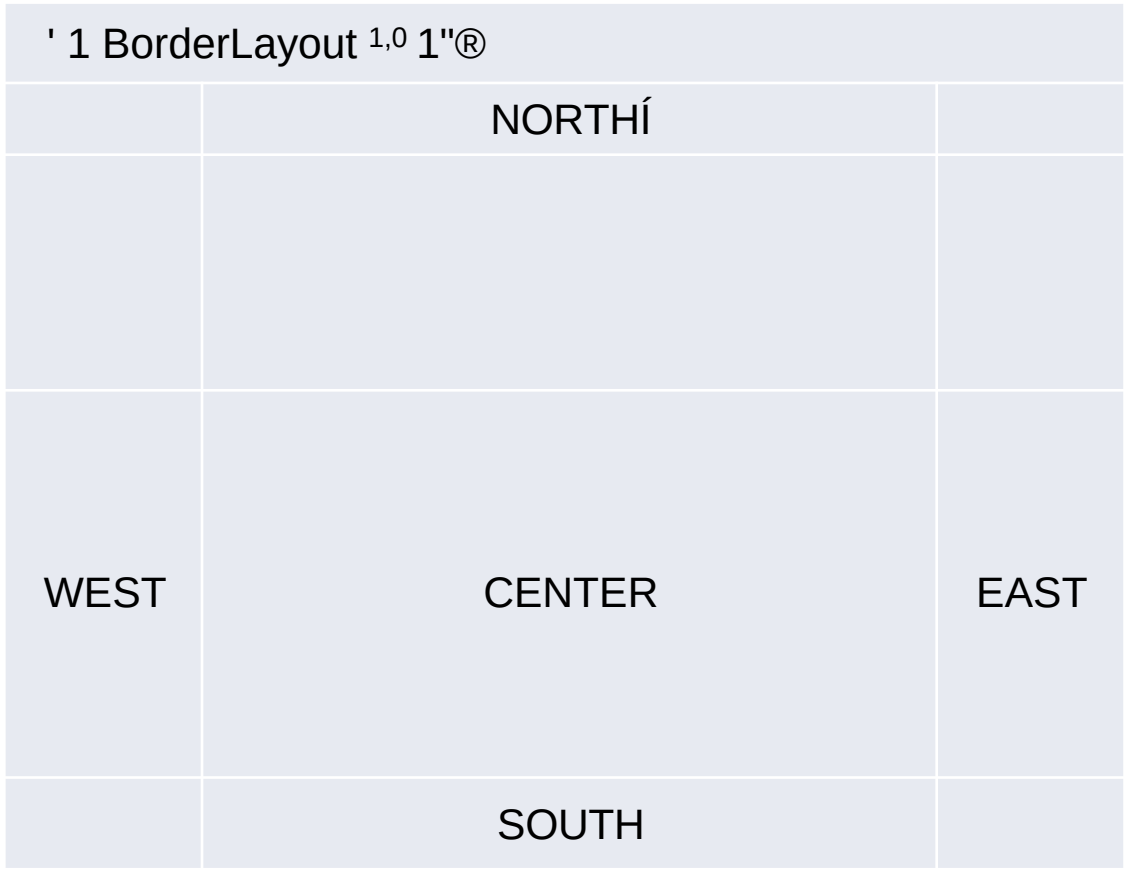
FlowLayout

- Ví dụ:

```
1    FlowLayout layout=new FlowLayout();
2    layout.setAlignment(FlowLayout.LEFT);
3    this.setLayout(layout );
4
```

BorderLayout

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



BorderLayout

- khởi tạo:
  - public BorderLayout ()
    - hgap = 0
    - vgap = 0
  - public BorderLayout (int hgap, int vgap)
    - hgap: chiều ngang
    - vgap : chiều dọc

BorderLayout

```
4   private JButton jbtNorth; private JButton jbtSouth;
5   private JButton jbtEast; private JButton jbtWest;
6   private JButton jbtCenter;
7   //Phương thức khởi tạo public BorderLayoutFrame()    {
8   this.initComponents();
9   //Phương thức khởi tạo các thành phần
10  private void initComponents() {
11  //MainFrame
12      this.setDefaultCloseOperation(WindowConstants.EXIT ON CLOSE);
13      this.setTitle("BorderLayout");
14      BorderLayout layout = new BorderLayout();
15      this.setLayout(layout) ;
16  //jbtNorth
17      this.jbtNorth = new JButton("NORTH");
18      this.getContentPane().add(this.jbtNorth,BorderLayout.NORTH);
19  //jbtWest
20      this.jbtWest = new JButton("WEST");
21      this.getContentPane().add(this.jbtWest,BorderLayout.WEST);
22  //jbtEast
23      this.jbtEast = new JButton(MEASTM);
24      this.getContentPane().add(this.jbtEast,BorderLayout.EAST);
25  //jbtSouth
26      this.jbtSouth = new JButton(MSOUTHM);
27      this.getContentPane().add(this.jbtSouth,BorderLayout.SOUTH);
28  //jbtCenter
29      this.jbtCenter = new JButton(MCENTERM);
30      this.getContentPane().add(this.jbtCenter,BorderLayout.CENTER);
31      pack();
32  }
```

CardLayout

- Card Layout quản lý nhiều Card cùng một không gian hiển thị
- Card Layout giúp quản lý hai hay nhiều Component (thường là JPanel) để chia sẽ 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**

CardLayout

- khởi tạo :
  - public CardLayout ()
    - hgap = 0
    - vgap = 0
  - public CardLayout (int hgap, int vgap)
    - hgap: chiều ngang
    - vgap : chiều dọc

CardLayout

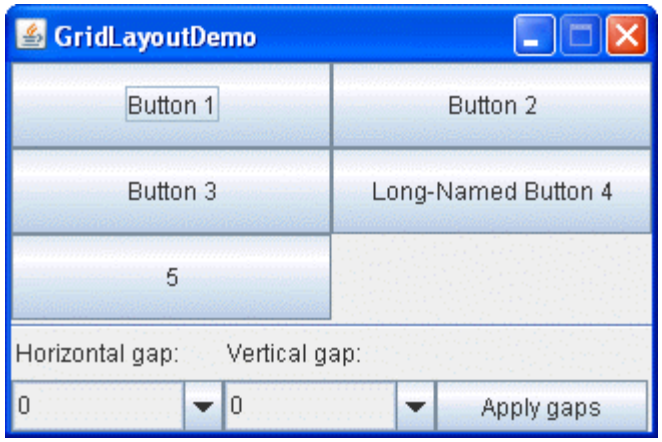
- Phương thức
  - public void setHgap(int hgap)
  - public void setVgap(int vgap)
  - public int getHgap()
  - public int getVgap()
  - public void next (Container parent)
  - public void previous(Container parent)
  - public void first(Container parent)
  - public void last(Container parent)
  - public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

CardLayout


CardLayoutDemo
Card with JButtons
Button 1
Button 2
Button 3
CardLayoutDemo
Card with JTextField
TextField

GridLayout

- 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).
- Kích thước của mỗi Cell được xác định bởi kích thước của Container.



GridLayout

- Khởi tạo
  - GridLayout()
    - rows: 1
    - -1 Component / 1 Column
  - GridLayout(int rows, int cols)
  - GridLayout(int rows, int cols, int hgap, int vgap)

GridLayout

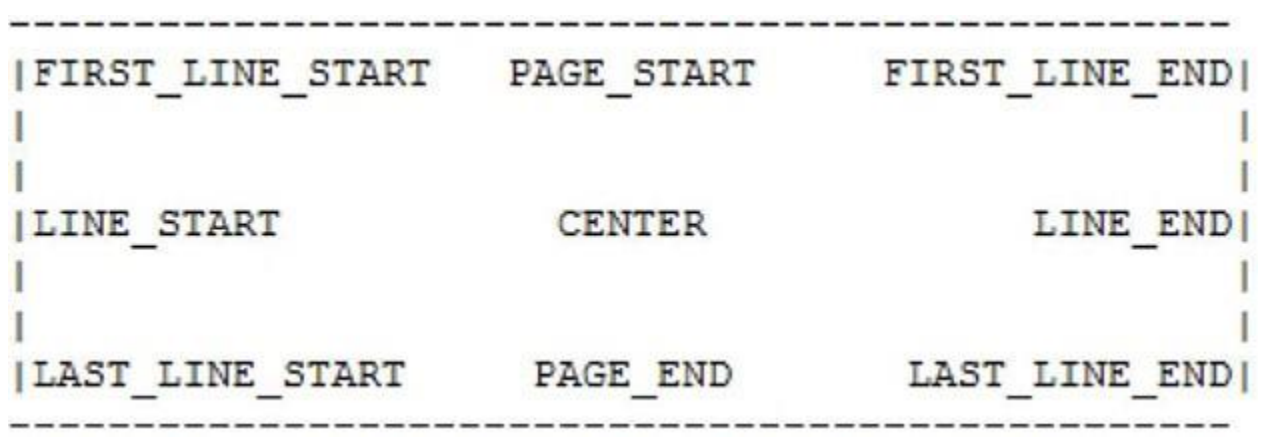
- GridBag Layout bố trí các Component trong một Grid với các Row và Column.
- Mỗi Component bên trong Grid được RowSpan và ColumnSpan (giống table HTML)
- Width và Height của các Row/Column có thể khác nhau.
- GridBag Layout là một Layout Manager rất linh động cho việc bố trí các Component bên trong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

GridLayout

- Các thuộc tính của GridBagContraints
  - 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

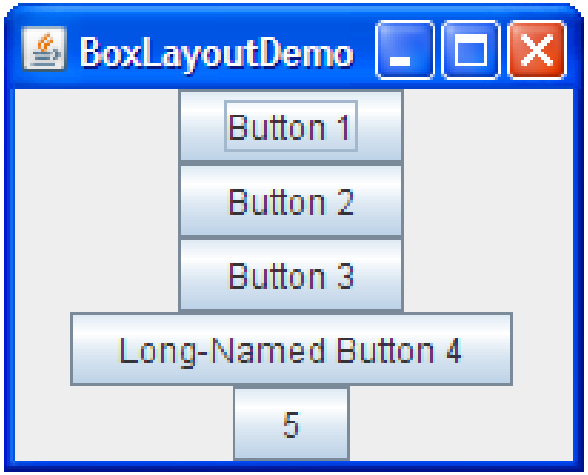
GridLayout

- Các thuộc tính của GridBagContraints
  - anchor:



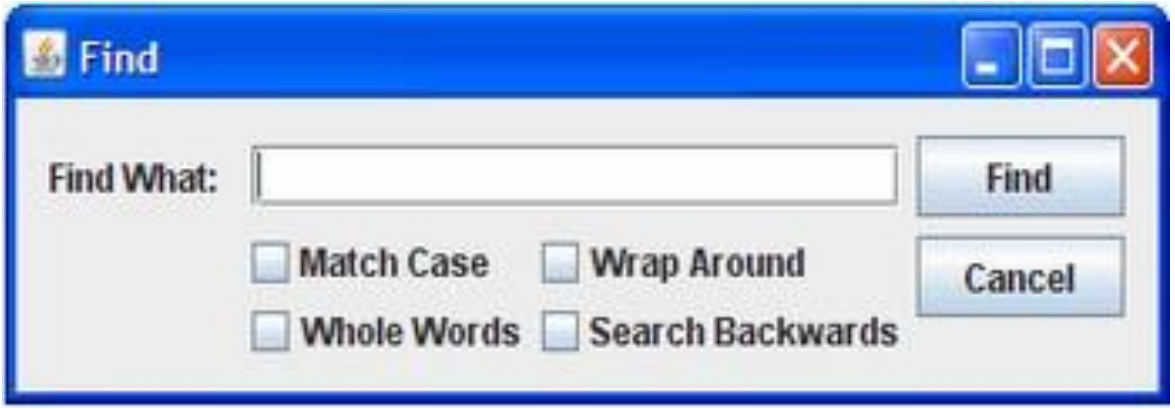
BoxLayout

- 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



GroupLayout

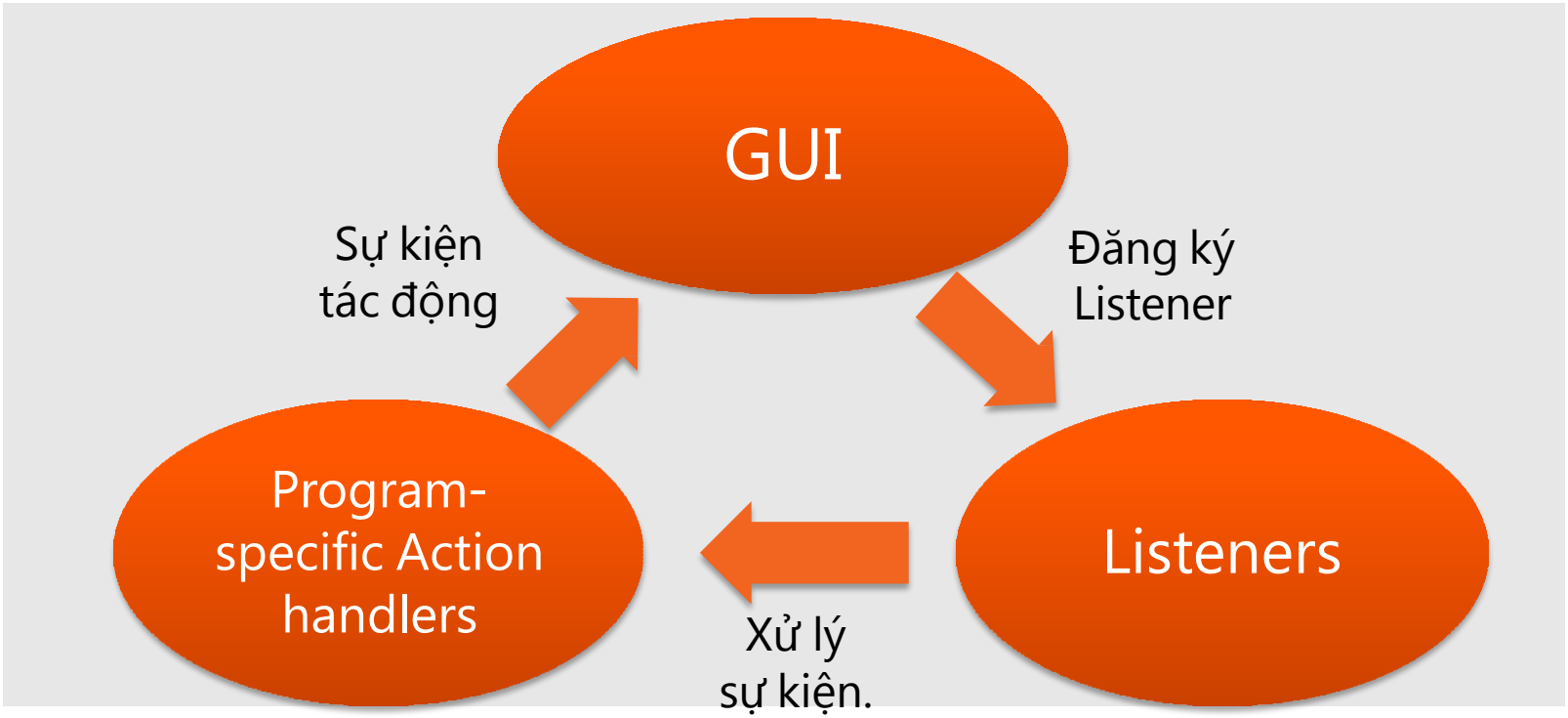
- Group Layout bố trí các Component bên trong Container theo chiều ngang và chiều dọc.
- Sự bố trí được thực hiện theo mỗi chiều riêng lẽ
-



Xử lý sự kiện

- Một sự kiện thường xảy ra khi có sự thay đổi trong giao diện người dùng đồ họa. Ví dụ như khi người dùng click chuột vào một button, click vào một mục trong combo box… và như vậy một sự kiện sẽ được kích hoạt.
- Các thành phần đồ họa (components) tạo ra các sự kiện này được gọi là ‘nguồn sự kiện’ (event source). Các nguồn sự kiện sẽ gửi đi một đối tượng sự kiện (event object).
- Các sự kiện được xử lý bởi một sự kiện lắng nghe (event listener) được gán cho nguồn sự kiện. Các thành phần khác nhau của GUI sẽ có các event listener khác nhau.

Xử lý sự kiện



- Xây dựng GUI và kết nối (còn gọi là đăng ký) tới các listeners.
- Listener sẽ thực thi (implements) các interface thích hợp với từng loại sự kiện.
- Thực hiện các lệnh (trong xử lý sự kiện) phù hợp với nguồn sự kiện.

Xử lý sự kiện

Event Object	Interface và method	Các method liên kết	Event Source
ActionEvent	interface **ActionListener** actionPerformed(ActionEvent)	addActionListener removeActionListener	JButton JCheckBox JComboBox JTextField JRadioButton
ItemEvent	interface **ItemListener** itemStateChanged(ItemEvent)	addItemListener removeItemListener	JCheckBox JRadioButton JComboBox
MouseEvent	interface **MouseListener** mousePressed(MouseEvent) mouseReleased(MouseEvent) mouseEntered(MouseEvent) mouseExited(MouseEvent) mouseClicked(MouseEvent)	addMouseListener removeMouseListener	*Được tạo bởi sự kiện Mouse của bất kỳ thành phần GUI nào.*

Xử lý sự kiện

- Sử dụng Inner Class Listener có tên: Là cách thường dùng để viết các chương trình nhỏ. Nhiều component có thể dùng chung Inner Class này.
- Anonymous Inner Class Listeners: Là inner class không đặt tên, thường được dùng cho một component. Class dạng này không được linh hoạt.
- Top-level Listeners (this): Thường được sử dụng khi có một event source.

Xử lý sự kiện

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

class SomePanel extends JPanel {
private JButton myGreetingButton = new JButton("Hello");
private JTextField myGreetingField = new JTextField(20);
public SomePanel() {
        ActionListener doGreeting = new GreetingListener();
        myGreetingButton.addActionListener(doGreeting);
        myGreetingField.addActionListener(doGreeting);
        // Các lệnh khác
}
// Inner class tên là GreetingListener
private class GreetingListener implements ActionListener {
    public void actionPerformed(ActionEvent e) {
        myGreetingField.setText("Guten Tag");
        }
     }
}
```

Xử lý sự kiện

```
class myPanel extends JPanel {
. . .
   public MyPanel() {
        JButton b1 = new JButton("Hello");

        b1.addActionListener(
        new ActionListener(){
          public void actionPerformed(ActionEvent e){
              // Các lệnh xử lý sự kiện cho b1
          }
        }
       );
...
}
```

Xử lý sự kiện

```
...
class Hello extends JFrame implements ActionListener{
        JButton b;
        ...
        public Hello() {
                JButton b = new JButton("Hello");
                b.addActionListener(this);
                ...
        }
        public void actionPerformed(ActionEvent e) {
                ...
        }
        ...
}
```

XIN CẢM ƠN!