

Bài 9

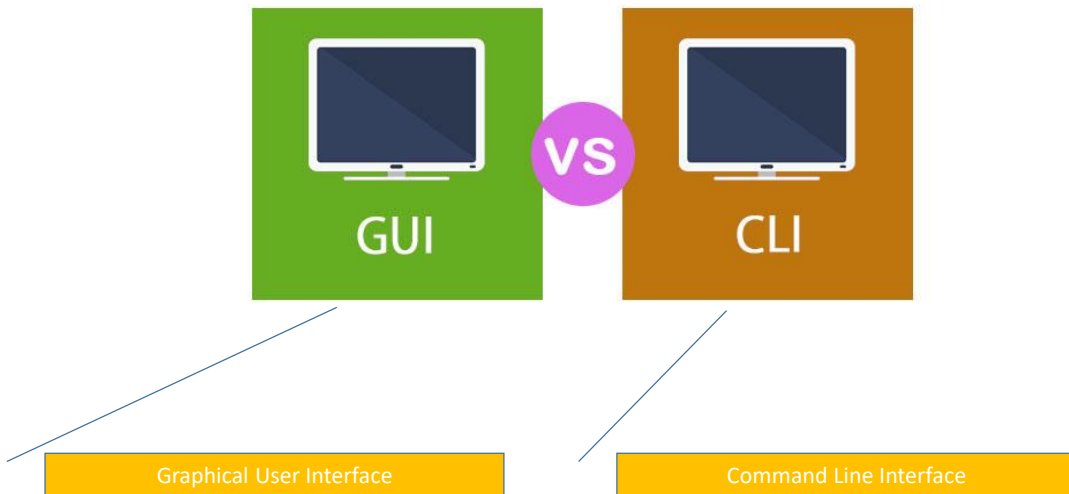
LẬP TRÌNH GIAO DIỆN

NỘI DUNG:

- Giới thiệu GUI của Java
- GUI với AWT
- GUI với Swing
- Bài tập

1. Giới thiệu GUI trong Java:

1.1. Sự khác nhau giữa giao diện đồ họa và text:



1.2. Các công cụ để xây dựng GUI của Java:

Hai bộ công cụ chính:

- JFC: Java Foundation Classes
- JavaFX

JFC là gì ?

Java Foundation Classes

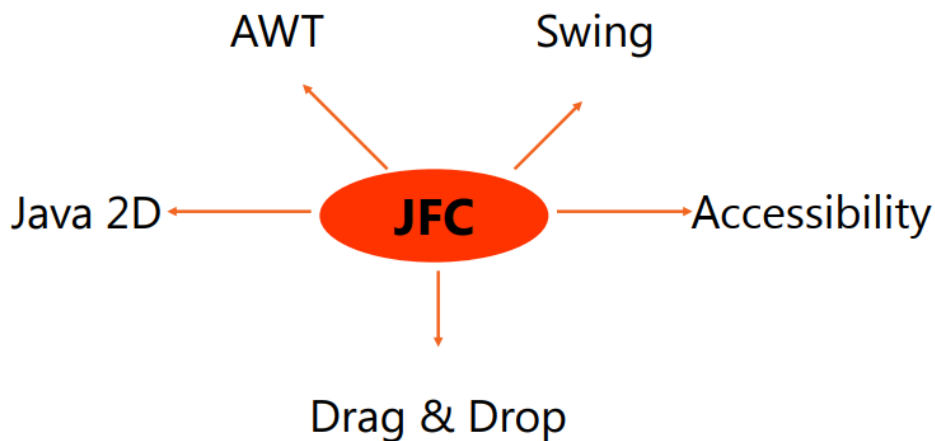
Tập hợp các tính năng để xây dựng giao diện người dùng đồ họa – GUI, để tạo ra các chức năng đồ họa phong phú và tương tác với các ứng dụng Java

JFC là một dự án phối hợp giữa Netscape's Internet Foundation Classes(IFC) và IBM's Taligent division and Lighthouse Design

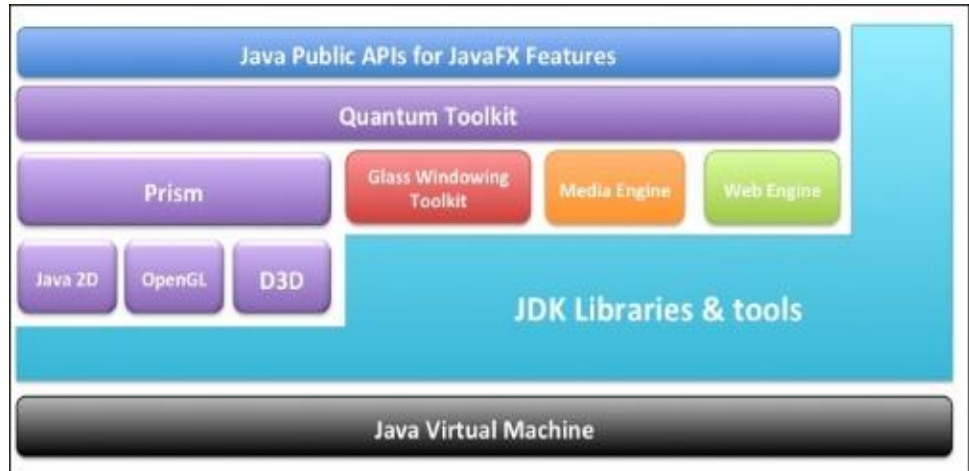
JFC bao gồm 5 thư viện:

- Swing GUI Components
- AWT
- Pluggable Look and Feel Support
- Accessibility API
- Java 2D API

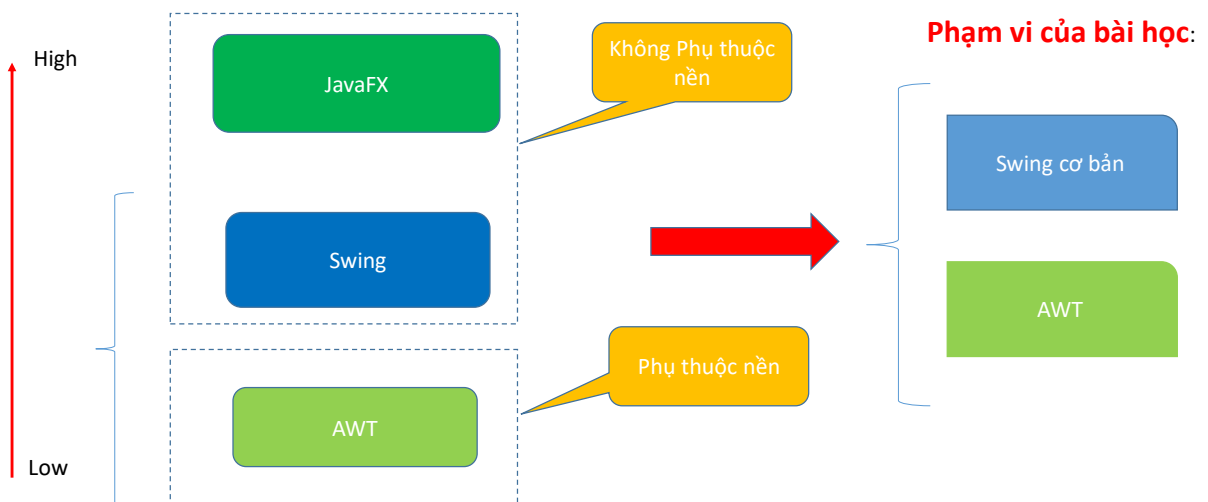
Java Foundation Classes



JavaFX:



1.3. So sánh AWT, Swing và JavaFX:



1.4. Xây dựng giao diện GUI:

- Dùng các thư viện hỗ trợ:
 - AWT:
`java.awt,`
`java.awt.event`
 - Swing:
`javax.swing,`
`java.swing.event`

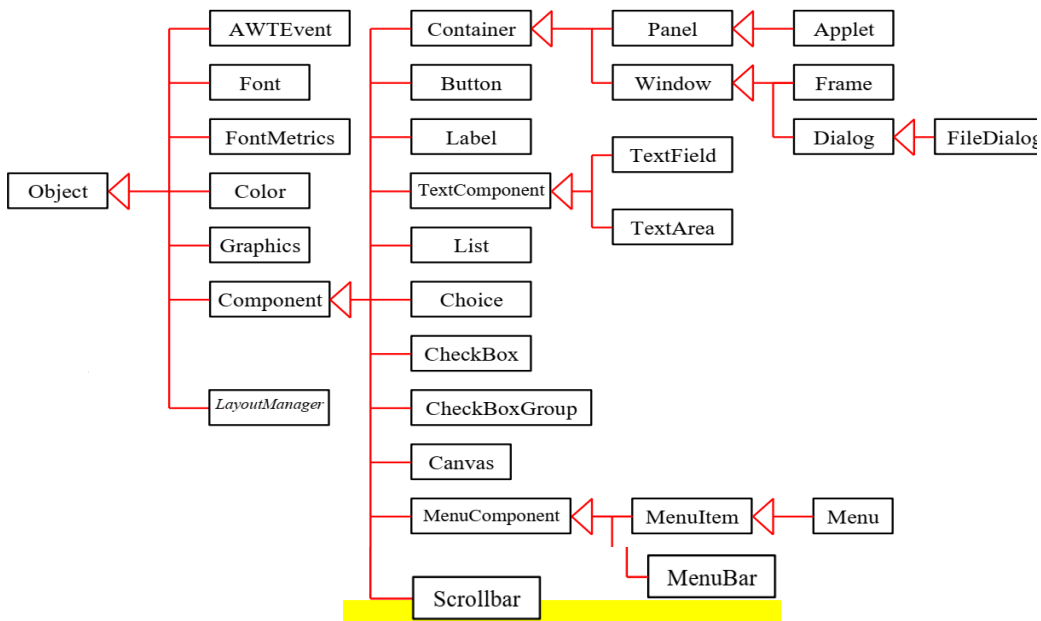
AWT

Giới thiệu chung

(Abstract Windows Toolkit)

- Thư viện API cung cấp các đối tượng GUI
- Tạo liên kết giao diện giữa ứng dụng Java và OS
- Chiếm nhiều tài nguyên hệ thống (Heavy-weight component)
- Package `java.awt`
- Gồm nhiều phần tử (class) để tạo GUI.
- Có các lớp quản lý việc bố trí các phần tử.
- Có **(event-oriented application)** mô hình ứng dụng hướng sự kiện.
- Có các công cụ xử lý đồ họa và hình ảnh.
- Các lớp sử dụng các tác vụ với clipboard (vùng nhớ đệm) như cut, paste.

Gói AWT của Java



SWING

Giới thiệu chung

SWING không phải là một từ viết tắt, Swing là một sản phẩm của gia đình lớn Java, một phần của dự án JFC

Swing bắt đầu được phát triển từ bản beta của JDK 1.1, khoảng mùa xuân năm 1997, đến tháng 3 năm 1998 thì bắt đầu được công bố rộng rãi

Khi phát hành thư viện Swing 1.0 chứa 250 class và 80 interface, đến nay phiên bản Swing 1.4 chứa 451 class và 85 interface

Mặc dù Swing được phát triển riêng rẽ với phần lõi của Java Development Kit, nó đòi hỏi ít nhất JDK 1.1.5 để chạy

- Sự xuất hiện thêm Swing từ Java 1.2 nhằm giúp khắc phục sự khó khăn của AWT khi cần bổ sung thêm các widget mới
- Đặc biệt, khắc phục bộ mặt nghèo nàn của chương trình viết bằng AWT so với các giao diện đẹp đẽ khác (chẳng hạn với MFC (Microsoft Foundation Classes) của Microsoft

Các thành phần của Swing

- MVC
- JFrame
- JDesktopPane
- JInternalFrame
- JLabel
- JButton
- JPasswordField
- JCheckBox
- JRadioButton
- JPanel
- JComboBox, JList
- JTable
- JMenu
- JToolBar
- JOptionPane
- JFileChooser

Swing Components

- Các thành phần của Swing bắt nguồn từ AWT.
- Được viết hoàn toàn bằng JAVA
- Chứa đựng cảm quan (Look and Feel): sự thể hiện và cách người dùng tương tác với chương trình.
- Cách dùng: `import javax.swing.*`

Swing Components

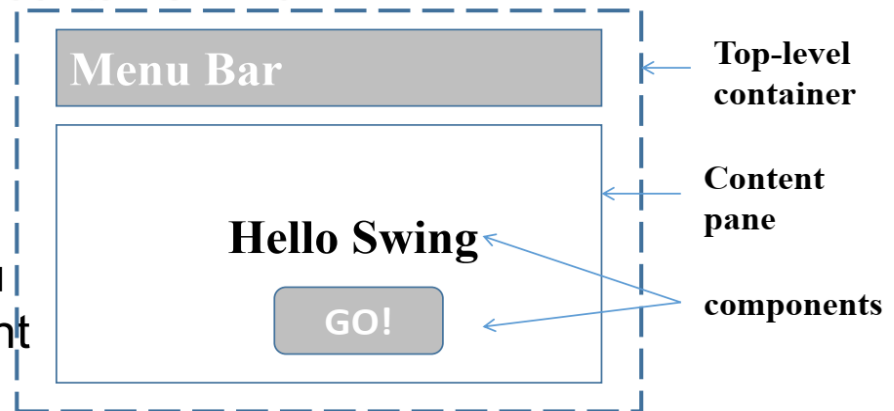
- Có các thành phần tương ứng với AWT. Tên của các thành phần của Swing thường bắt đầu bằng “J”. VD: JFrame, JLabel, JButton...
- Swing có các thành phần phức tạp hơn mà AWT không có như: tabbed panes, scroll panes, trees, tables...
- Các thành phần của swing có thể hỗ trợ nhiều tính năng hơn so với AWT.
- Để sử dụng các thành phần của Swing cần có chỉ thị: `import javax.swing.*`

Swing Components

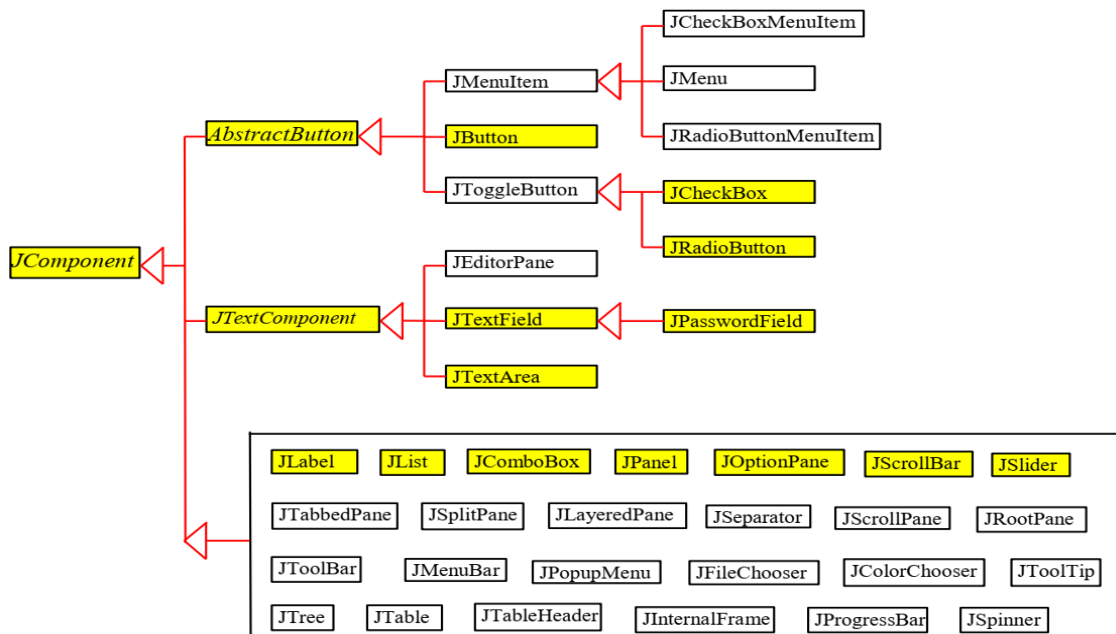
- Tất cả các ứng dụng của Swing phải có ít nhất một top-level container như:

- Frame
- Dialog
- Applet

- Mỗi top-level container đều có một content pane – chứa tất cả các thành phần visible còn lại, ngoại trừ menu bar.



Core Swing Components



So sánh AWT và SWING

AWT	SWING
<ul style="list-style-type: none"> ▪ Xây dựng bằng native code ▪ Khó phát triển thêm các linh kiện(widget) mới 	<ul style="list-style-type: none"> ▪ Xây dựng hoàn toàn bằng JAVA API ▪ Dễ phát triển các linh kiện ▪ Có thể thay đổi diện mạo của linh kiện lúc runtime ▪ Mô hình MVC (Model – View – Controller)

So sánh AWT và SWING

AWT	SWING
	<p>Có thể bổ sung thêm biểu tượng bên cạnh dòng chữ vào Label, Button, Menu item</p> <ul style="list-style-type: none"> ▪ Tạo đường viền và ghi tựa cho các thành ▪ phần Swing

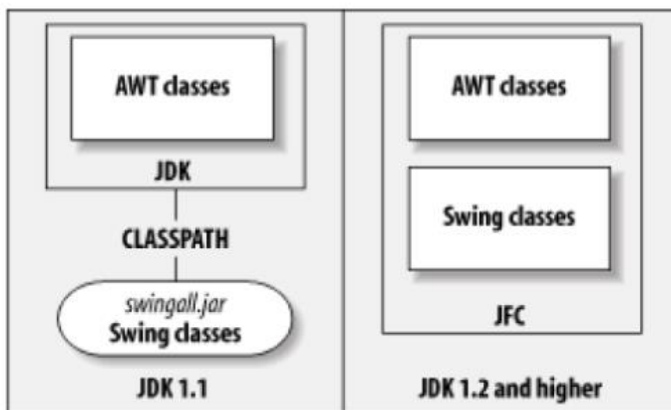
So sánh AWT và SWING

AWT	SWING
	<ul style="list-style-type: none"> ▪ Có thể chọn một thành phần Swing bằng ▪ Cách dùng phím thay cho chuột ▪ Các thành phần Swing sử dụng các nhãn Unicode, vì vậy có thể đưa tiếng Việt vào các thành phần này

Vậy SWING là sự thay thế của AWT?

Không. Swing thực tế được xây dựng trên phần lõi của AWT, bởi vì Swing không chứa bất kỳ mã dành cho nền tảng nào (native code)

Mô tả mối quan hệ giữa AWT, SWING, và JDK:



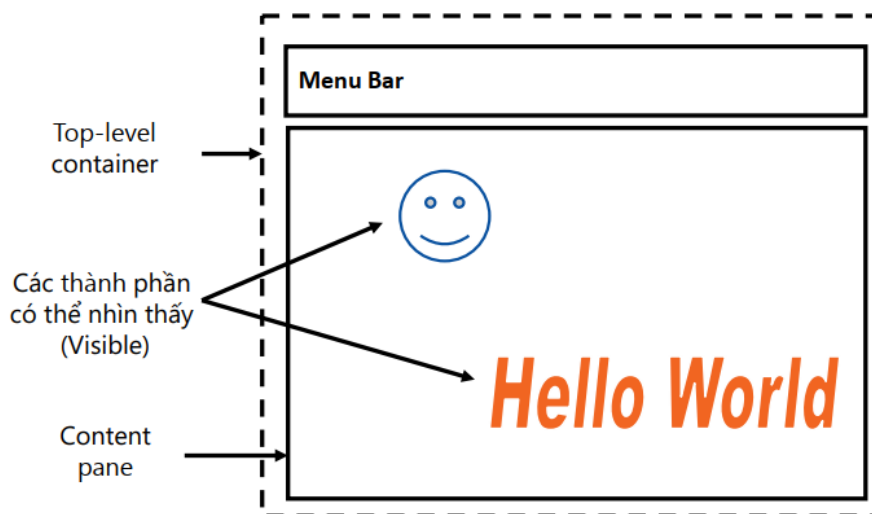
2. Lập trình GUI với Swing

Containers

Một container là một thành phần đặc biệt có thể chứa các thành phần khác.

- **Top-level container** là một container cấp cao nhất, ở trên cùng của một hệ thống phân cấp.
- Swing cung cấp **4 container top-level container** là: `JFrame`, `JDialog`, `JWindow` và `JApplet`.
- Để hiển thị trên màn hình, mỗi một thành phần GUI phải là một phần của hệ thống phân cấp. Mỗi hệ thống phân cấp sẽ có một **top-level** là gốc. Trong đó **JFrame** hay được sử dụng nhất.

Top-Level Containers

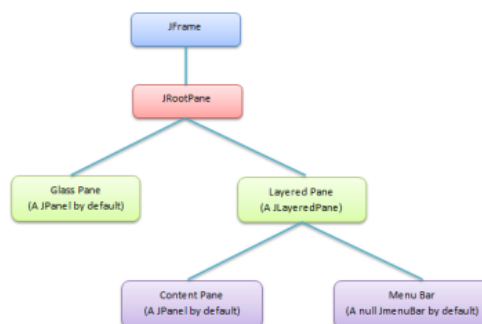


Container trong Java Swing

- 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?

- **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

- JFrame Là top-level container.
- JFrame được dùng để tạo ra 1 cửa sổ.
 - `JFrame()`
 - `JFrame(String title)`
- Các thành phần khác sẽ được add vào cửa sổ JFrame đó:
 - `frame.getContentPane().add(b);` // cũ
 - `frame.add(b)` // mới

JFrame

Đóng JFrame

```
public int getDefaultCloseOperation()
public void setDefaultCloseOperation(int operation)
```

Các hằng số khi đóng 1 JFrame:

- `DISPOSE_ON_CLOSE` (value = 1)
- `DO_NOTHING_ON_CLOSE` (value = 2)
- `EXIT_ON_CLOSE` (value = 3)
- `HIDE_ON_CLOSE` (value = 4) // mặc định

JFrame

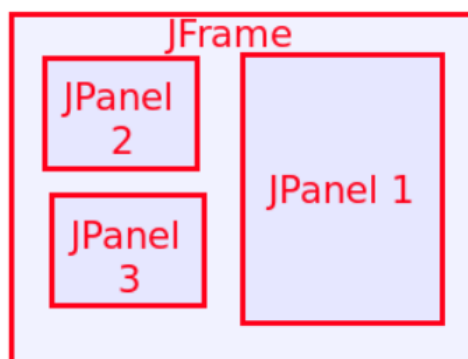
```
import javax.swing.*;

class FrameTest {

    public static void main(String args[] {
        JFrame frame = new JFrame("My Frame");//tiêu đề
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(300,200);// đặt kích thước của sổ
        frame.show();// hoặc setVisible(true)
    }
}
```

JPanel là gì?

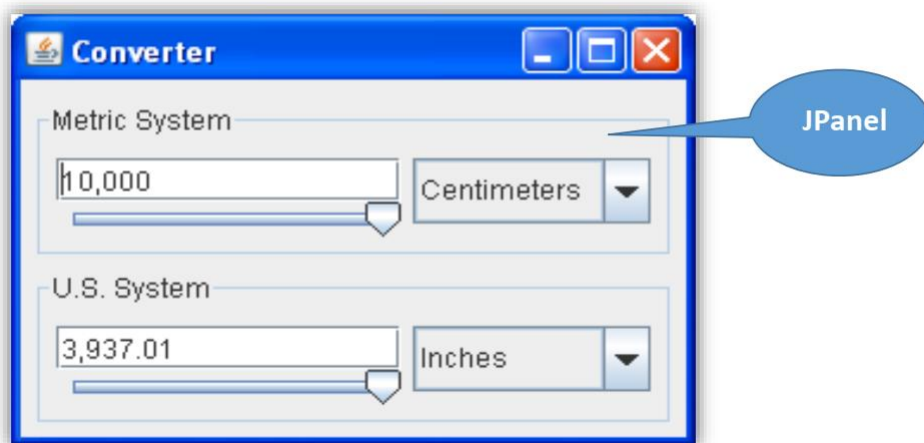
- JPanel là một container dùng để chứa các thành phần đồ họa khác (tương tự như JFrame tuy nhiên nó không phải là 1 JFrame).
- Trong một JFrame chứa các JPanel, trong mỗi JPanel lại có thể chứa các đối tượng hoặc thậm chí là các JPanel khác.



JPanel

- Giúp tổ chức các thành phần
- Lớp **JPanel** là lớp con của **JComponent**, **Container** và **Component**.
- Có thể có nhiều thành phần (và các khung chứa panel khác) được thêm vào chúng.
- Layout mặc định là **FlowLayout**
- Constructors
 - `JPanel ();`
 - `JPanel (LayoutManager lm);`
- Methods: có các phương thức của các lớp **JComponent**, **Container**, **Component**.

JPanel



JPanel

- Methods:
 - **setBorder** (Border border)
 - **add** (Component comp)
- Events:
 - **mouseClicked**

JPanel

- JPanel là container được dùng để nhóm các thành phần với nhau.
- JPanel sử dụng FlowLayout là mặc định.
- JPanel có các constructor:
 - `JPanel()`
 - `JPanel(LayoutManager mg)`

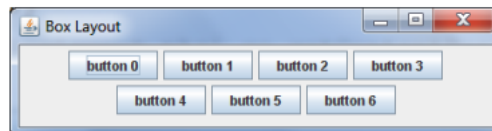
Layout Manager:

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

AWT

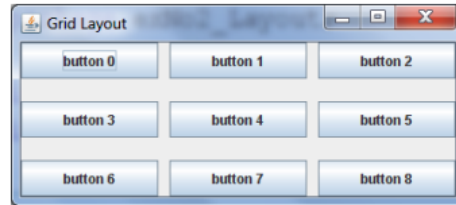
FlowLayout

- Là Layout đơn giản và là mặc định cho Panel.
- Các thành phần được sắp xếp theo hàng.
- Hướng của layout:
 - FlowLayout.LEFT
 - FlowLayout.CENTER
 - FlowLayout.RIGHT
- Khoảng cách giữa các thành phần:
 - Có giá trị mặc định = 5
 - Có thể đặt lại qua phương thức setHgap và setVgap.



GridLayout

- Là tập hợp các ô lưới có cùng kích thước
- Khoảng cách giữa các ô:
 - Mặc định = 5
 - Có thể đặt lại qua phương thức `setHgap` và `setVgap`.
- Thay đổi kích thước của các ô:
 - Các ô có thể có kích thước to hơn phụ thuộc vào kích thước của cửa sổ.



BorderLayout

- Có 5 vùng:
 - NORTH, SOUTH, EAST, WEST, CENTER
 - Không phải tất cả 5 vùng đều được sử dụng
- Để đặt các thành phần vào một vị trí:
 - `Panel.setLayout(new BorderLayout())`
 - `Panel.add(new JButton("Click here"), BorderLayout.NORTH)`
- Đặt lại khoảng cách giữa các thành phần:
 - Có thể đặt lại qua phương thức `setHgap` và `setVgap`.
 - Sử dụng `BorderLayout(int horGap, int verGap)`



BoxLayout

- Khi đặt các component vào container, BoxLayout dựa vào các thông tin minimum size, prefer size và maximum size.
- Khi BoxLayout đặt các component từ trên xuống dưới, nó sẽ cố đặt các component với kích thước chiều cao xác định trong preferred size.
- Vùng không gian trống sẽ đặt ở phía bên dưới của container

BoxLayout

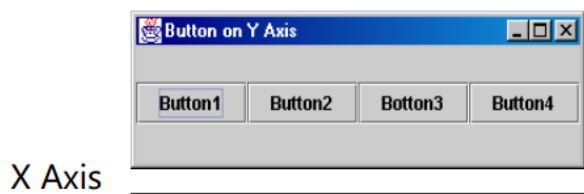
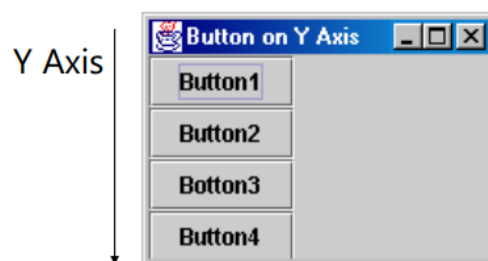
Các thành phần được sắp xếp:

- X_AXIS: Theo chiều ngang từ trái qua phải
- Y_AXIS: Theo chiều dọc từ trên xuống dưới
- `panel.setLayout(new BoxLayout(panel, BoxLayout.X_AXIS));`
- `panel.setLayout(new BoxLayout(panel, BoxLayout.Y_AXIS));`

BoxLayout

- X Alignment.
 - Giá trị của các X Alignment này sẽ từ 0.0 tới 1.0.
 - 0.0 ứng với `Component.LEFT_ALIGNMENT`
 - 0.5 ứng với `Component.CENTER_ALIGNMENT`
 - 1.0 ứng với `Component.RIGHT_ALIGNMENT`.

BoxLayout

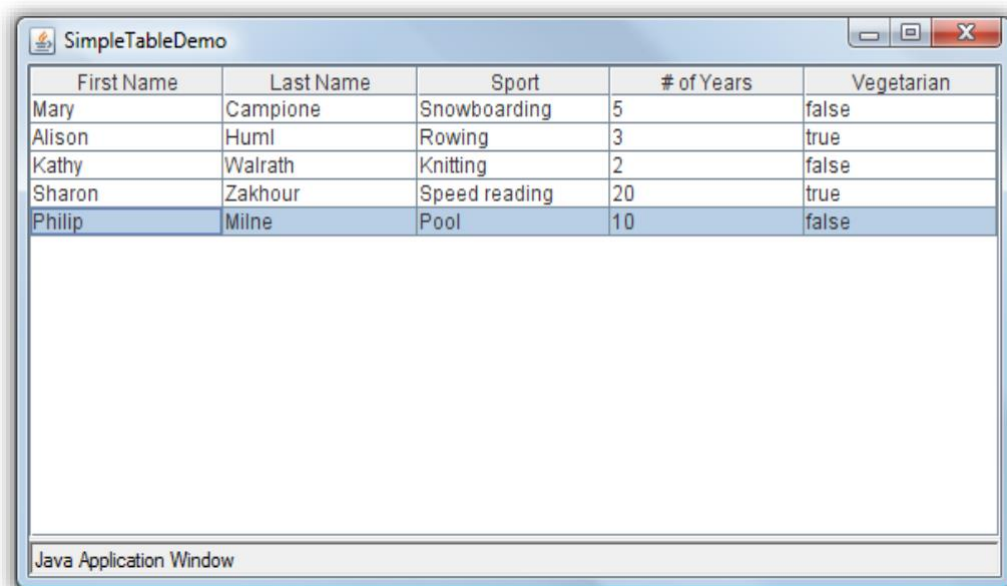


JOptionPane –Hàm hiển thị Dialog

- Hai hàm dùng để hiển thị Dialog
 - void showMessageDialog(...)
 - void showDialog(...) *//can display a customized dialog*
- Hàm thường sử dụng:


```
void showMessageDialog (Component  
parentComponent, ObjectMessage);
```

JTable



First Name	Last Name	Sport	# of Years	Vegetarian
Mary	Campione	Snowboarding	5	false
Alison	Huml	Rowing	3	true
Kathy	Walrath	Knitting	2	false
Sharon	Zakhour	Speed reading	20	true
Philip	Milne	Pool	10	false

JTable

- JTable không chứa hoặc lưu trữ dữ liệu, nó chỉ cung cấp cách hiển thị dữ liệu.
- Constructor
 - JTable (Object [][] cells, String [] colName);
 - JTable(TableModel model);

JTable

- DefaultTableModel
 - **addColumn** (Object obj)
 - **addRow** (Object obj)
 - **getColumnCount** ()
 - **getRowCount** ()
 - **getValueAt** (int row, int col)
 - **setValueAt** (Object obj, int row, int col)

JTable

- **Methods:**

- **setModel** (TableModel tm)
- **getModel** ()
- **getValueAt** (int row, int col)
- **getRowCount** ()
- **getColumnCount** ()

- **Events:**

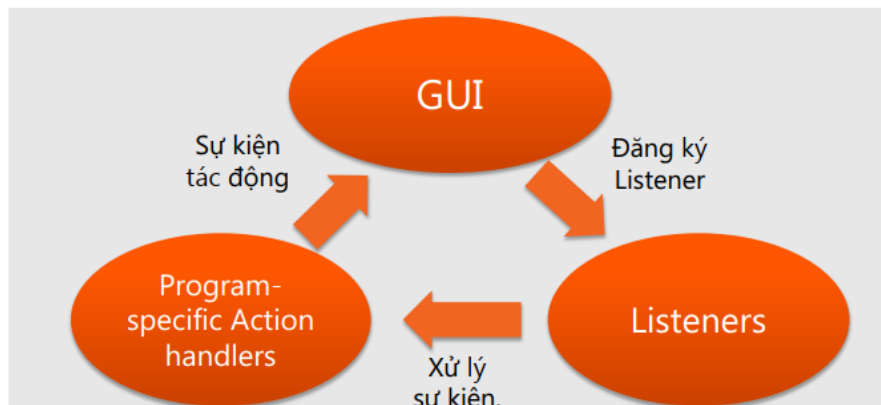
- **mouseClicked**

3.5. Xử lý sự kiện của Swing

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	<i>Được tạo bởi sự kiện Mouse của bất kỳ thành phần GUI nào.</i>

Event Handling

- Các thành phần của swing có thể phát sinh các sinh các sự kiện được định nghĩa trong `java.awt.event` và `javax.swing.event`.
- Hàm đăng ký sự kiện: `addABCListener()`
- Hàm hủy đăng ký sự kiện:
`removeABCListener()`

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.

