Unit 6:
GUI with JavaFX

Er. Shankar pd. Dahal
pdsdahal@gmail.com

JavaFX :

- JavaFX is a Java library used to build Rich Internet Applications. The applications written using this library can run consistently across multiple platforms.

- The applications developed using JavaFX can run on various devices such as Desktop Computers, Mobile Phones, Tablets, etc..

- To develop GUI Applications using Java programming language, the programmers rely on libraries such as Advanced Windowing Tool kit and Swing. After the advent of JavaFX, these Java programmers can now develop GUI applications effectively with rich content.

- JavaFX allows developers to create applications with features such as multimedia, 2D and 3D graphics, animations, and more.

- Used to create both desktop and web application.

- JavaFX is intended to replace Swing as the standard GUI library for java software environment.

- Link Swing draws its own components, less communication with OS.

JavaFX Vs. Swing:

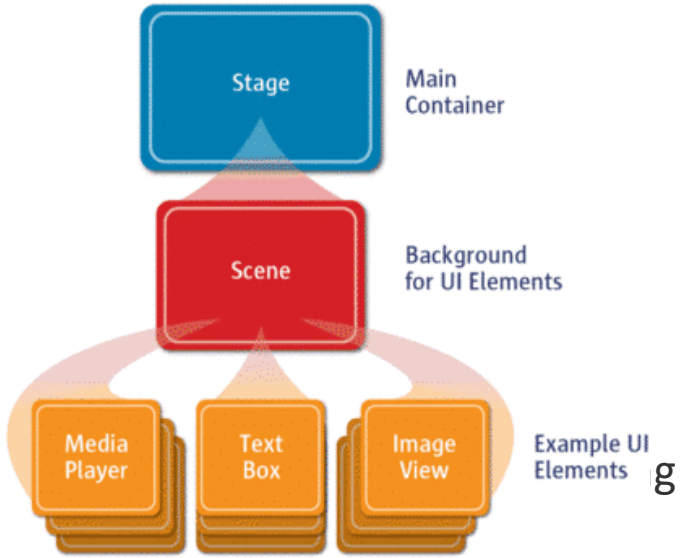
Basis of Comparison Between Java Swing vs Java FX	Java Swing	Java FX
Components	A large number of components provided.	Less number of Components.
User Interface	Standard UI components can be designed with Swing	Rich UI can be designed with JavaFX.
Development	Swing APIs are being used to write UI components	JavaFX scripts and fast UI development with screen builder
Functionality	No new functionality introduction for future	JavaFX has a rich new toolkit, expected to grow in future
Category	Legacy UI library fully featured	Up and coming to feature-rich UI components
MVC Support	MVC support across components lack consistency	Friendly with MVC pattern.

JavaFX Application Structure

- JavaFX application is divided hierarchically into three main components known as Stage, Scene and nodes.
- We need to import **javafx.application.Application** class in every JavaFX application.

In order to create a basic JavaFX application, we need to:

1.Import **javafx.application.Application** into our code.
2.Inherit **Application** into our class.
3.Override **start()** method of Application class.



Stage :

- It is a top-level container that represents the main window of a graphical applicatior
- It serves as the primary container for JavaFX applications and provides a platform-sp the graphical user interface (GUI) of the application.

Scene :

- It is a container for the graphical content of a user interface.
- It represents the content that is displayed within a Stage (main window) of a JavaFX application.
- The Scene class in JavaFX is part of the scene graph, which is a hierarchical tree-like structure of nodes representing the visual elements of the user interface.

Example:

```
import javafx.application.Application;
import javafx.scene.Scene;
import javafx.scene.control.Label;
import javafx.scene.layout.BorderPane;
import javafx.stage.Stage;


public class Sample extends Application {
@Override
public void start(Stage primaryStage) throws Exception {

primaryStage.setTitle("JavaFx Demo");
primaryStage.show();

}


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

JavaFX Layouts:

- In JavaFX, Layout defines the way in which the components are to be seen on the stage.
- JavaFX provides various built-in Layouts that are :

1. BorderPane
2. FlowPane
3. Vbox
4. Hbox
5. GridPane
6. StackPane

1. BorderPane :

Organizes nodes in top, left, right, centre and the bottom of the screen.

Constructors

There are the following constructors in the class.

- BorderPane() : create the empty layout
- BorderPane(Node Center) : create the layout with the center node
- BorderPane(Node Center, Node top, Node right, Node bottom, Node left) : create the layout with all the nodes


```
BorderPane borderPane = new BorderPane();
borderPane.setTop(new Label("Top"));
borderPane.setLeft(new Label("Left"));
borderPane.setCenter(new Label("Center"));
borderPane.setRight(new Label("Right"));
borderPane.setBottom(new Label("Bottom"));
```

2. FlowPane :

- ❖ Organizes the nodes in the horizontal rows according to the available horizontal spaces.
- ❖ Wraps the nodes to the next line if the horizontal space is less than the total width of the nodes.

Constructors:

There are 8 constructors in the class that are given below.

- ➢ FlowPane()
- ➢ FlowPane(Double Hgap, Double Vgap)
- ➢ FlowPane(Double Hgap, Double Vgap, Node? children)
- ➢ FlowPane(Node... Children)
- ➢ FlowPane(Orientation orientation)
- ➢ FlowPane(Orientation orientation, double Hgap, Double Vgap)
- ➢ FlowPane(Orientation orientation, double Hgap, Double Vgap, Node? children)
- ➢ FlowPane(Orientation orientation, Node... Children)

```
FlowPane flowPane = new FlowPane();
Button btn1 = new Button("Button ");
flowPane.getChildren().addAll(btn1);
```


3. Vbox :

- ❖ VBox (Vertical Box) arranges its children in a single vertical column.

Constructors :

- ➢ VBox() : creates layout with 0 spacing
- ➢ Vbox(Double spacing) : creates layout with a spacing value of double type
- ➢ Vbox(Double spacing, Node? children) : creates a layout with the specified spacing among the specified child nodes
- ➢ Vbox(Node? children) : creates a layout with the specified nodes having 0 spacing among them

```
VBox vBox = new VBox();
Button btn1 = new Button("Button ");
vBox.getChildren().addAll(btn1);
```

4. Hbox :

❖ HBox (Horizontal Box) arranges its children in a single horizontal row.

Constructors:

➢ HBox() : create HBox layout with 0 spacing

➢ Hbox(Double spacing) : create HBox layout with a spacing value

```
HBox hBox = new HBox();
Label lblUserName  = new Label("UserName : ");
hBox.getChildren().addAll(lblUserName);
```


5. GridPane :

❖ Organizes the nodes in the form of rows and columns.

Constructors

➢ GridPane(): creates a gridpane with 0 hgap/vgap.

```
GridPane gridPane = new GridPane(5,5);
gridPane.add(lblUserName, 0, 0);// column , row
gridPane.add(txtUserName, 1, 0);
```

JavaFx UI Controls :

- The package **javafx.scene.control** provides all the necessary classes for the UI components like Button, Label, etc. Every class represents a specific UI control and defines some methods for their styling.

Label:

- Label is a component that is used to define a simple text on the screen.

Constructors:

1. Label(): creates an empty Label
2. Label(String text): creates Label with the supplied text
3. Label(String text, Node graphics): creates Label with the supplied text and graphics

TextField:

- Text Field is basically used to get the input from the user in the form of text.

Constructors:

1. TextField() : creates an empty textfield
2. TextField(String text): creates TextField with the supplied text

Button:

- Button is a component that controls the function of the application.

Constructors:

1. Button() : creates an empty Button
2. Button(String text): creates Button with the supplied text

RadioButton :

- The Radio Button is used to provide various options to the user. The user can only choose one option among all. A radio button is either selected or deselected.
- you can group them using the ToggleGroup class to ensure that only one radio button within the same group can be selected at a time.

Constructors :

1. RadioButton(): creates an empty radioButton
2. RadioButton(String text): creates radioButton with the supplied text

CheckBox:

- The Check Box is used to provide more than one choices to the user.
- It can be used in a scenario where the user is prompted to select more than one option or the user wants to select multiple options.

Constructors :

1. CheckBox(): creates an empty checkBox
2. CheckBox(String text): creates checkBox with the supplied text

Hyperlink:

- Hyper-links to refer the web pages.
- It is similar to anchor links in HTML.

Constructors:

1. Hyperlink(): creates an empty hyperlink
2. Hyperlink(String text): creates hyperlink with the supplied text

Menu:

- JavaFX provides a Menu class to implement menus.
- Menu is the main component of any application.

```
Menu  menu1= new Menu(”Go");
```

MenuBar:

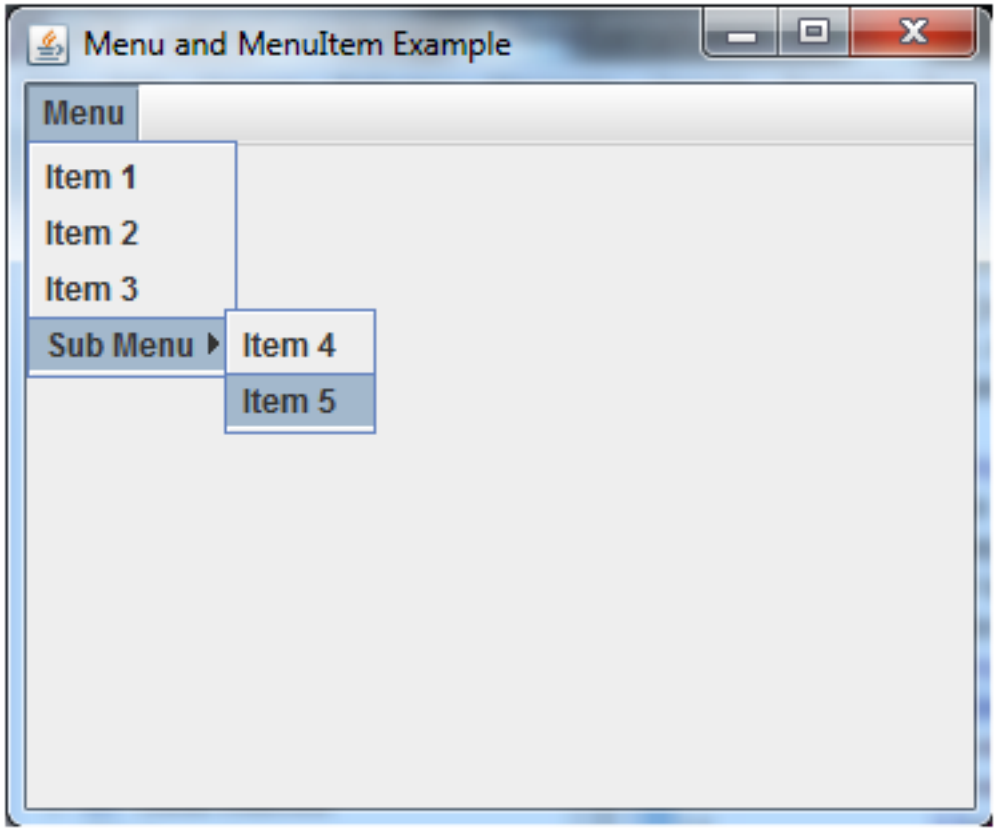
- It is used to create a menu bar that can contain menus.

```
MenuBar menuBar = new MenuBar();
menuBar.getMenus().addAll(menu1,menu2);
```

MenuItem:

- It is a class that represents an item within a menu.

```
MenuItem cutItem = new MenuItem("Cut");
menu1.getItems().addAll(cutItem);
```



ToolTip:

- It is used to provide hint to the user about any component.
- It is mainly used to provide hints about the text fields or password fields being used in the application.