UNIT – 4

JAVAFX BASICS AND EVENT-DRIVEN PROGRAMMING AND ANIMATIONS

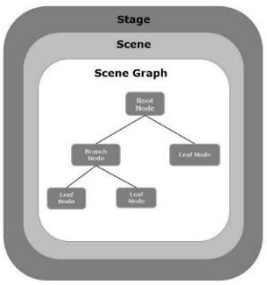
What is JavaFX ?

JavaFX is a free , open-source framework that lets you create application with rich graphics and media.

What is the Basic Structure of JavaFX ?

In General JavaFX application will have three major components.

1. Stage
2. Scene
3. Node



1. Stage :

   A stage (window) contains all the objects of JavaFX application. It's represent stage class of the package ***javafx.stage*** . you have to call the **show()** method to display the contents of stage.

2. Scene :

   A scene graph is a data structure similar to a tree , in modern graphical application , it's a collection of nodes. The scene class holds all the contents of a scene graph

3. Node :

   A Node is a visual / graphical primitive object of a JavaFX application. In other words Node is a widget or a shape which is used to represent data in JavaFX application. Each node in scene graph has a single parent , and the node which doesn't contain any parent is known as **root node**.

   The Following are various kinds of root nodes in JavaFX application.

   GROUP : collective node that contains list of child node. If any effect apply to group it will be applied to all child node.

   REGION : it is based class of all JavaFX node-based UI Controls , such as Chart , Pane and Control.

   WebView : This node manages web engine and displays its contents.

   Leaf Node : The node without Child node is known as leaf node . For Ex. Rectangle , ImageView , MediaView are example of leaf node.

Pane in JavaFX

Pane is used store or holds the node. The Following are some layout pane

1. **HBox :**
   This layout arranges all the nodes in single horizontal row.
2. **VBox :**
   This layout arranges all the nodes in single vertical column.
3. **BorderPane :**
   This layout arranges all the nodes in top , left , right , bottom and center position.
4. **StackPane :**
   This layout arranges all the node like stack one node on the another node.
5. **TextFlow :**
   This layout arranges multiple text node in single flow.

6. **AnchorPane :**
   This layout anchors the nodes in our application at a particular distance from the pane.
7. **TilePane :**
   This layout arranges its child nodes in uniformly sized tiles, either vertically or horizontally .
8. **GridPane :**
   This layout arrange all the nodes as grid of rows and columns .
9. **FlowPane :**
   This layout wraps all the nodes in a flow.

What is UI Controls and Shapes ?

UI stands for ***User Interface*** . In simple word UI Controls means widget like TextField , PasswordField , Slider etc. All UI controls are available in ***javafx.scene.control*** package . Following is list of Common Controls used by designing the GUI in JavFX :

1. **Label :**
   A Label object is a component for placing text.
2. **Button :**
   This class create a labelled button.
3. **ColorPicker :**
   ColorPicker provide a pane of controls designed to allow a user to manipulate and select a color.
4. **CheckBox :**
   It's a graphical component that can be either on(true) or off(false) state.
5. **RadioBox :**
   It's a graphical component that can be either on(true) or off(false) state in a group
6. **ListView :**
   It's a component present the user with a scrolling list of text items.
7. **TextField :**
   A TextField object is a text component that allow for the editing of a single line of text.
8. **PasswordField :**
   A PasswordField is a text component specialized for password entry.
9. **Scrollbar :**
   A scrollbar control represent the scroll bar component in order to enable user to select from range of values.
10. **FileChooser :**
    A FileChooser control represent a dialog window from which the user can select a file.
11. **ProgressBar :**
    As the task progresses towards completion , the progress bar display the task's percentage of completion.
12. **Slider :**
    A slider lets the user graphically select a value by sliding a knob within a bounden interval.

Also JavaFX provides a shapes like Circle , Polygon , Arc etc . All shapes are available in ***javafx.scene.shape*** package. Following is list of shapes provided by JavaFX.

1. **Line :**
   It's a geometrical structure joining two point.
2. **Rectangle :**
   In General Rectangle is four-sided polygon that has two pairs of parallel and concurrent sides with all interior angles as right angles.

3. **Rounded Rectangle :**

   You can draw rectangle either with sharp edges or with arched edges and the one with arched edges is known as Rounded Rectangle.

4. **Circle :**

   A circle is a line forming a closed loop , every point on which Is a fixed distance from a center point.

5. **Ellipse :**

   An ellipse is defined by two points, each called a focus. If any point on the ellipse is taken, the sum of the distances to the focus points is constant. The size of the ellipse is determined by the sum of these two distances.

6. **Polygon :**

   A closed shape formed by a number of coplanar line segments connected end to end.

7. **Polyline :**

   A polyline is same a polygon except that a polyline is not closed in the end. Or, continuous line composed of one or more line segments.

8. **Cubic Curve :**

   A cubic curve is a Bezier parametric curve in the XY plane is a curve of degree 3.

9. **Quad Curve :**

   A quadratic curve is a Bezier parametric curve in the XY plane is a curve of degree 2.

10. **Arc :**

    An arc is part of a curve.

11. **SVGPath :**

    In JavaFX, we can construct images by parsing SVG paths.

What is Property binding ? and how to bind it ?

Property binding means apply an event to a any property(Data Type). In Java we can not bind any event with normal data types like Integer , String , Double etc. so we have to user property binding class. Property binding class have capacity to bind an event with self. Some of property binding class name is StringProperty , IntegerProprty etc.

Color and Font class

COLOR : Color class is used to provide a color in UI.using Color class we can provide a color in our application by default text has black color. For provide a color we can write :

```
Color clr=Color.rgb(0,0,255);
Text t=new Text("Demo");
t.setFill(clr);
```

FONT : Font Class is used to provide a different font in UI.

```
Text t=new Text("Demo");
t.setFont(Font.font("arial",FontWeight.BOLD, FontPosture.REGULAR,20));
```

Image and Image-View class

Image class is used to load image / store the path of image(using FileInputStream) and ImageView class in used to display the image in our application. In ImageView class we pass the object of Image class which holds the path of image.

```
Image path=new Image(new FileInputStream("C:\\User\\XYZ\\Pictures\\img1.png"));
ImageView img=new ImageView(path);
img.setX(20);
img.setY(20);
```

```
img.setFitWidth(250);
img.setFitHeight(250);
img.setPreserveRatio(true);
```

Events , Events sources and Handling Events

There Are mainly two types of events (1) Foreground Event and (2)Background Event. Using this event we can perform any action. Foreground Event means user can perform a event like click on button , keypress , mouse move etc. Background Event means we don't require any interaction like software failure , timer Expire etc.

Handling events means when user click on a button or perform any action on widgets perform an action.

```
Button btn=new Button("CLICK");
Btn.setOnAction(new EventHandler<ActionEvent>(){
        System.out.println("Click");
});
```

Now, Let's convert it into lambda Expression

```
Button btn=new Button("Click");
Btn.setOnAction(event -> System.out.println("Click"));
```

Animation

Animation means change or move the any control or shape. Like move circle from one location to another , Fill color etc. Following are animation classes.

Fade Transition , Fill Transition , Rotate Transition , Scale Transition , Stroke Transition , Translate Transition , Path Transition , Sequential Transition , Pause Transition , Parallel Transition .

UNIT – 5

JAVAFX UI CONTROLS AND MULTIMEDIA

In this unit we learn control . Which is label , checkbox , Button , Radiobutton , Textfield , Textarea , Video , Combobox etc.

Some of examples are below.

1) **Label**

```
Label lbl = new Label("USERNAME");
lbl.setX(20);
lbl.setY(20);
lbl.setTextFill(Color.GREEN);
```

2) **Button**

```
Button btn=new Button("login");
btn.setX(20);
btn.setY(20);
btn.setOnAction(event-> System.out.println("Click");
```

3) **Video**

```
File Video=new File("C:\\User\\XYZ\\video\\demo.mp4");
Media md=new Media(video.toURI().toString());
MediaPlayer mp=new MediaPlayer(md);
MediaView mv=new mediaView(mp);
mv.setFitHeight(500);
mv.setFitWidth(500);
```

4) **RadioButton**

```
RadioButton rb1=new RadioButton("JAVA");
RadioButton rb2=new RadioButton("C#");
RadioButton rb3=new RadioButton("SAD");
ToggleGroup tg=new ToggleGroup();
rb1.setToggleGroup(tg);
rb2.setToggleGroup(tg);
rb3.setToggleGroup(tg);
```

5) **ComboBox**

```
ComboBox<String> sub=new ComboBox<String>();
sub.setPromptText("Select Subject");
ObservableList<string> list=sub.getItems();
list.add("JAVA");
list.add("C#");
list.add("PYTHON");
```

6) **ListView**

```
ObservableList<String> list=FXCollections.observableArrayList("JAVA","C#","SAD");
ListView<String> lv=new ListView<string>(list);
lv.setMaxSize(200,160);
```

7) **Slider**

```
Slider s=new Slider(1,100,20); //1=startvalue , 100=endvalue , 20= increment by
```

8) **Scrollbar**

```
ScrollBar sb=new ScreollBar();
//by default its horizontal if you want to add vertical Scrollbar Write following code
sb.setOrientation(Orientation.VERTICAL);
```

EXAMPLES

1)HBox

```
TextField t=new textField();
Button b1=new Button("Click");
Button b2=new Button("Reset");
HBox hb=new HBox();
hb.getChildern().addAll(t,b1,b2);
Scene sc=new Scene(hb);
stage.setScene(sc);
stage.show();
```

2)VBox

```
TextField t=new textField();
Button b1=new Button("Click");
Button b2=new Button("Reset");
VBox vb=new VBox();
vb.getChildern().addAll(t,b1,b2);
Scene sc=new Scene(hb);
stage.setScene(sc);
stage.show();
```

3) BorderPane

```
BorderPane b=new BorderPane();
b.setTop(new Buton("TOP"));
b.setBottom(new Button("BOTTOM"));
Scene sc=new Scene(b);
stage.setScene(sc);
stage.show();
```

4) TextFlow

```
Button b1=new Button("btn1");
Button b2=new Button("btn2");
Button b3=new Button("btn3");
TextFlow t=new textFlow(b1,b2,b3);
Scene sc=new Scene(b);
stage.setScene(sc);
stage.show();
```

5)GridPane

```
Text t1=new Text("Username");
Text t2=new Text("Password");
TextField txt1=new TextField();
TextField txt2=new TextField();
GridPane gp=new GridPane();
gp.add(t1,0,0);  // widget , column , row
gp.add(txt1,1,0);
gp.add(t2,0,1);
gp.add(txt2,1,1);
Scene sc=new Scene(b);
stage.setScene(sc);
stage.show();
```