Introduction to JavaFX

What is JavaFX?

JavaFX is a set of graphics and media packages that enable developers to design, create, test, debug, and deploy desktop applications and Rich Internet Applications (RIA) that operate consistently across diverse platforms. The applications built in JavaFX can run on multiple platforms including Web, Mobile, and Desktops.

Features of JavaFX:

Feature	Description
Java Library	It consists of many classes and interfaces that are written in Java.
FXML	FXML is the XML based Declarative markup language. The coding can be done in FXML to provide the more enhanced GUI to the user.
Scene Builder	Scene Builder generates FXML mark-up which can be ported to an IDE.
Web view	Web View uses WebKitHTML technology to embed web pages into the Java Applications.
Built in UI controls	Built-in controls are not dependent on operating system. The UI components are used to develop a full featured application.
CSS like styling	JavaFX code can be embedded with the CSS to improve the style and view of the application.
Swing interoperability	The JavaFX applications can be embedded with swing code using the Swing Node class. We can update the existing swing application with the powerful features of JavaFX.
Canvas API	Canvas API provides the methods for drawing directly in an area of a JavaFX scene.
Rich Set of APIs	JavaFX provides a rich set of API's to develop GUI applications.

Integrated Graphics Library	It is provided to deal with 2D and 3D graphics.
Graphics Pipeline	JavaFX graphics are based on Graphics rendered pipeline(prism). It offers smooth graphics which are hardware accelerated.
High Performance Media Engine	The media pipeline supports the playback of web multimedia on a low latency. It is based on a Gstreamer Multimedia framework.
Self-contained application deployment model	Self-Contained application packages have all of the application resources and a private copy of Java and JavaFX Runtime.

JavaFX Application Structure:

A JavaFX application will have three major components namely

1) Stage
2) Scene and
3) Nodes

as shown in the following diagram.

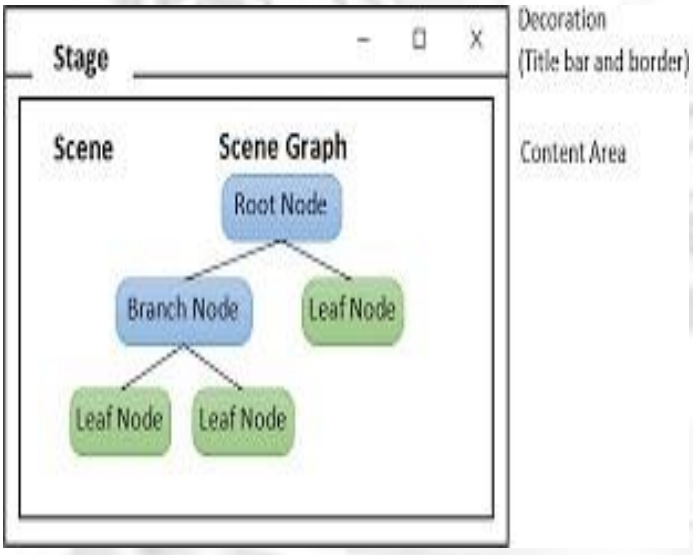


Figure: JavaFX App.lication Structure

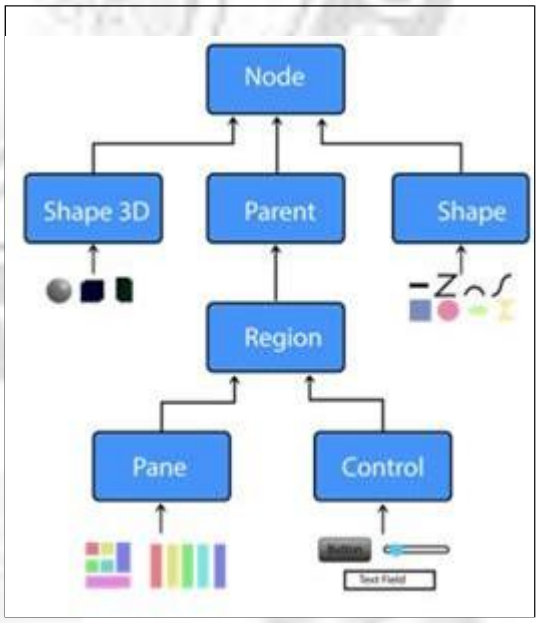


Figure: Scene Graph and Nodes

1) Stage

- ✓ Stage(a window) in a JavaFX application is similar to the Frame in a Swing Application. It acts like a container for all the JavaFX objects.
- ✓ Primary Stage is created internally by the platform. Other stages can further be created by the application.

- ✓ A stage has two parameters determining its position namely **Width** and **Height**. It is divided as Content Area and Decorations (Title Bar and Borders).
- ✓ There are five types of stages available –
  - o Decorated
  - o Undecorated
  - o Transparent
  - o Unified
  - o Utility
- ✓ We have to call the **show()** method to display the contents of a stage.

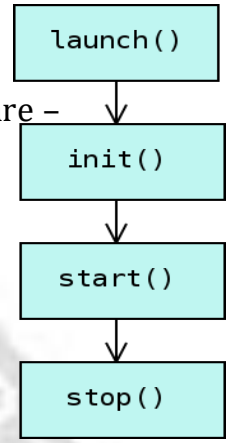
2) Scene

- ✓ A scene represents the physical contents of a JavaFX application. It contains all the contents of a scene graph.
- ✓ The class **Scene** of the package **javafx.scene** represents the scene object. At an instance, the scene object is added to only one stage.

3) Scene Graph and Nodes

- ✓ A **scene graph** is a tree-like data structure (hierarchical) representing the contents of a scene. In contrast, a **node** is a visual/graphical object of a scene graph.
- ✓ A node may include –
  - **Geometrical** (Graphical) objects (2D and 3D) such as – Circle, Rectangle, Polygon, etc.
  - **UI Controls** such as – Button, Checkbox, Choice Box, Text Area, etc.
  - **Containers** (Layout Panes) such as Border Pane, Grid Pane, Flow Pane, etc.
  - **Media elements** such as Audio, Video and Image Objects.
- ✓ A node is of three types –
  - o **Root Node** – The first Scene Graph is known as the Root node.
  - o **Branch Node/Parent Node** – the node with child nodes are known as branch/parent nodes. The parent nodes will be of the following types –
    - ➢ **Group** – A group node is a collective node that contains a list of children nodes. Whenever the group node is rendered, all its child nodes are rendered in order. Any transformation, effect state applied on the group will be applied to all the child nodes.
    - ➢ **Region** – It is the base class of all the JavaFX Node based UI Controls, such as Chart, Pane and Control.
    - ➢ **WebView** – This node manages the web engine and displays its contents.
  - o **Leaf Node** – The node without child nodes is known as the leaf node.

Lifecycle of a JavaFX Application:



The JavaFX Application class has the following life cycle methods, which are –

1) **launch()** - to launch JavaFX application.
2) **init()** – An empty method which can be overridden, but you cannot create a stage or scene in this method.
3) **start()** – The entry point method where the JavaFX graphics code is to be written.
4) **stop()** – An empty method which can be overridden, here we can write the logic to stop the application.

General Rules for writing JavaFX Application:

- ✓ A JavaFX Application must extend javafx.application.Application.
- ✓ The main() method should call Application.launch()
- ✓ The start() method is the main entry point for all JavaFX applications
  - Start() is called when a Stage is connected to the Operating System's window
- ✓ The content of the scene is represented as a hierarchical scene graph of nodes:
  - Stage is the top-level JavaFX Container
  - Scene is the container for all content

Minimal example

```
public class HelloWorld extends Application {
    public static void main(String[] args) {
        launch(args);
    }

    @Override
    public void start(Stage primaryStage) {
        primaryStage.setTitle("Hello World!");

        StackPane root = new StackPane();

        Button btn = new Button();
        btn.setText("Say 'Hello World'");

        root.getChildren().add(btn);

        primaryStage.setScene(new Scene(root, 300, 250));
        primaryStage.show();
    }
}
```