JavaFX – Packages and Layout

Lecture - 03

Minoli de Silva

FAQ: How does a JavaFX program start?

The JavaFX program starts execution in the main method similar to a simple Java Application.

1. The class containing the main method (a.k.a. the main class) inherits from a class in the JavaFX packages named Application.

2. Application class provides 4 method.
   1. launch
   2. Init
   3. start
   4. stop

3. These methods run in sequence from the start to end of a JavaFX application

4. Start method is where the Primary Stage is configured and displayed

FAQ: Can we code inside the Start Method?

The simple answer is yes.

You can create JavaFX elements inside the start method without using any FXML code or SceneBuilder to do so.

```java
@Override
public void start(Stage primaryStage) throws Exception{
    //Set title to stage
    primaryStage.setTitle("TITLE");

    //Create new AnchorPane
    AnchorPane anchorPane = new AnchorPane();

    //Create new Schene with anchorPane as the rootNode
    Scene myScene = new Scene(anchorPane, 300, 200); // width, height

    //Create a label and add to anchorPane as a child node
    Label label = new Label("A Label");
    anchorPane.getChildren().add(label);

    //Set scene and display stage
    primaryStage.setScene(myScene);
    primaryStage.show();
}
```

FAQ: How do we create a JavaFX window from Java code?

```
//class extends javafx application class to override start method
public class Test extends Application {
    //main method - calls the testMethod
    public static void main(String[] args) {
        System.out.println("I'm in the main method");
        testMethod();
        System.out.println("I'm back in the main method. Going to end the application");
    }

    //another method named testMethod - launches the application
    public static void testMethod(){
        System.out.println("I'm now in the test method. Going to launch the UI");
        launch();
        System.out.println("Window closed, going to leave the test method");
    }

    @Override
    public void start(Stage primaryStage) throws Exception {
        Stage stage = primaryStage;
        Parent root = FXMLLoader.load(getClass().getResource("login.fxml"));
        stage.setScene(new Scene(root, 500, 500));
        stage.show();
    }
}
```

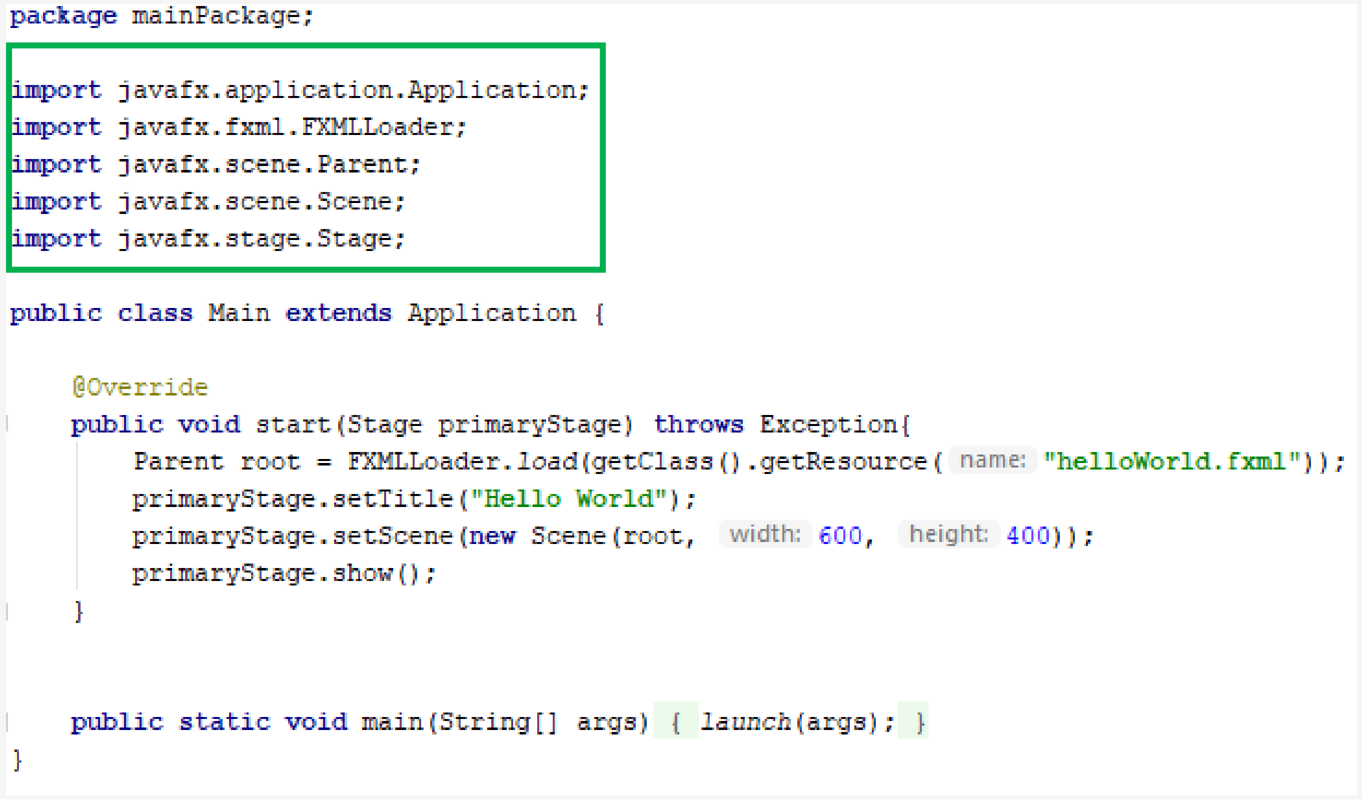
JavaFX Packages

- JavaFX module is composed of over 30 packages
- Each starting with the pre-fix javafx
- These packages must be imported to your code to use their classes in your code

- Ex:

import javafx.scene.control.Label;

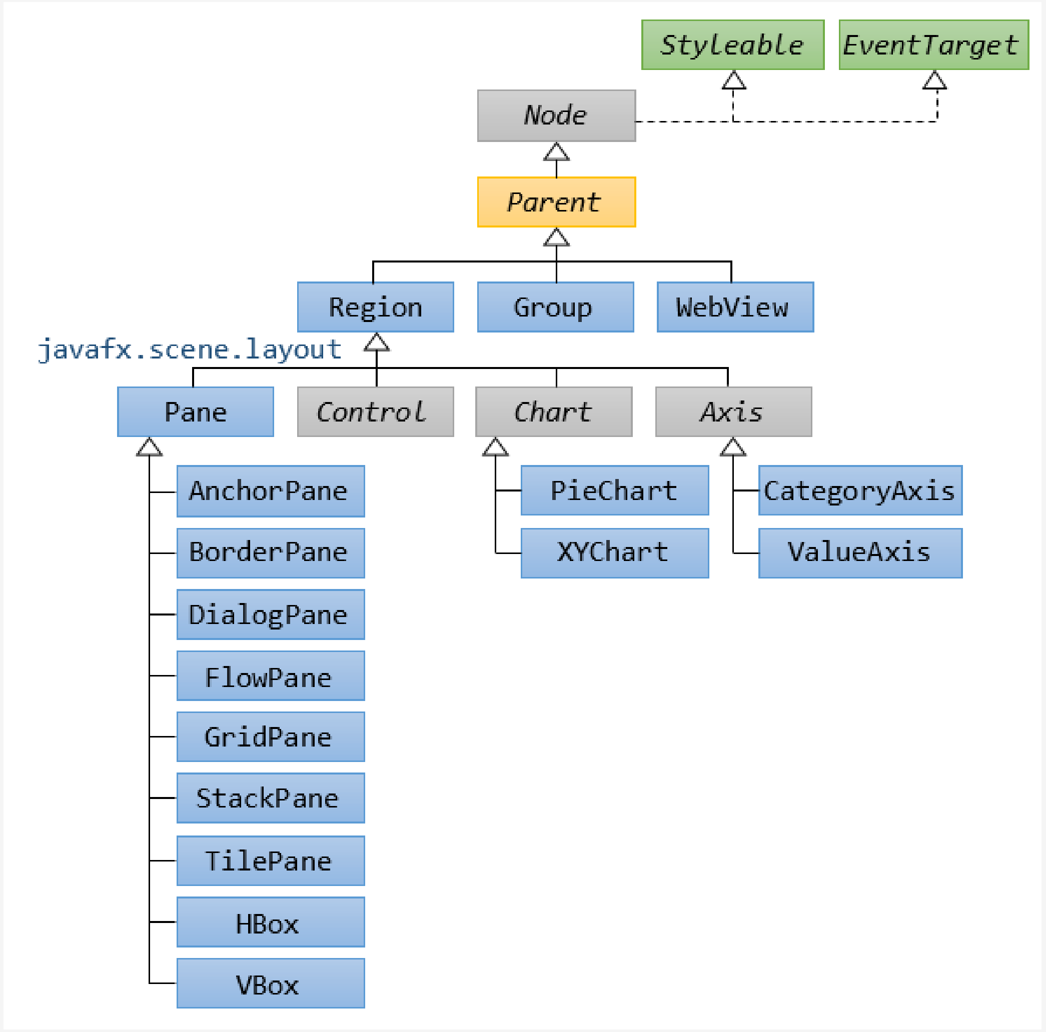
Allows the use of Label class of the control package.



```
package mainPackage;

import javafx.application.Application;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.stage.Stage;

public class Main extends Application {

    @Override
    public void start(Stage primaryStage) throws Exception{
        Parent root = FXMLLoader.load(getClass().getResource( name: "helloWorld.fxml"));
        primaryStage.setTitle("Hello World");
        primaryStage.setScene(new Scene(root,  width: 600,  height: 400));
        primaryStage.show();
    }


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

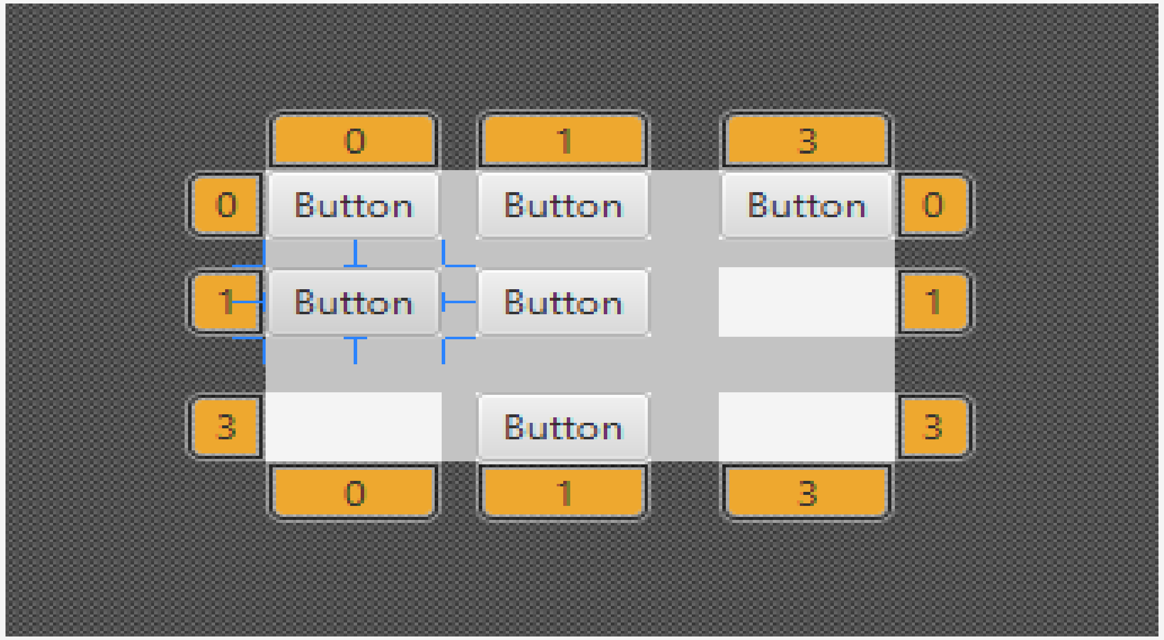
JavaFX Layouts - Containers

- JavaFX provides many types of panes for organizing nodes in a container
- Containers in JavaFX are inherited by Pane class.
- Some of the main types of containers are given below.
  - AnchorPane
  - BorderPane
  - FlowPane
  - GridPane
  - HBox
  - Vbox
  - SplitPane
  - TabbedPane



GridPane

- The GridPane layout enables the creation of a flexible grid of rows and columns in which nodes can be laid out. Nodes can be placed in any cell in the grid and can span multiple cells as needed.
- Useful layout when creating forms and grids in JavaFX applications.
- Nodes may span multiple rows or columns.



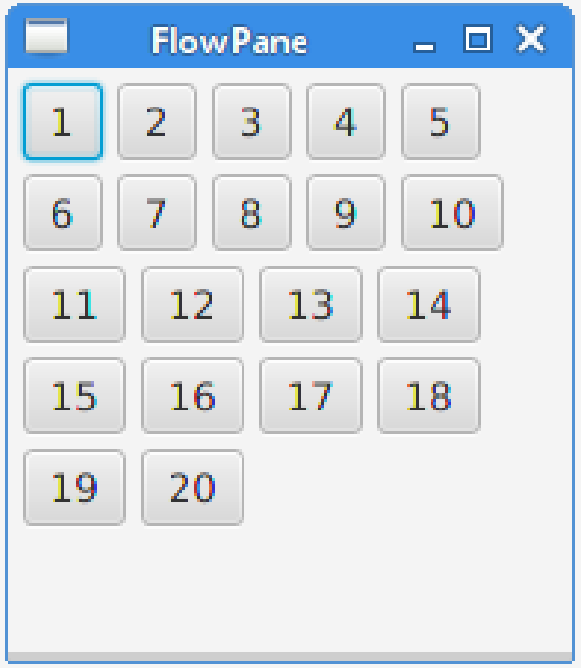
AnchorPane

- The Anchor pane layout anchors the nodes in our application at a particular distance from the pane.
- AnchorPane anchors the edges of child nodes to an offset from the anchor pane's edges.
- If the anchor pane has a border or padding set, the offsets will be measured from the inside edge of those insets.

```
<AnchorPane prefHeight="340.0" prefWidth="481.0" xmlns="http://javafx.com/javafx/8.0.172-ea"
        xmlns:fx="http://javafx.com/fxml/1">
   <children>
      <Button layoutX="274.0" layoutY="200.0" mnemonicParsing="false" text="Button" />
      <Label layoutX="209.0" layoutY="107.0" text="Label" />
   </children>
</AnchorPane>
```

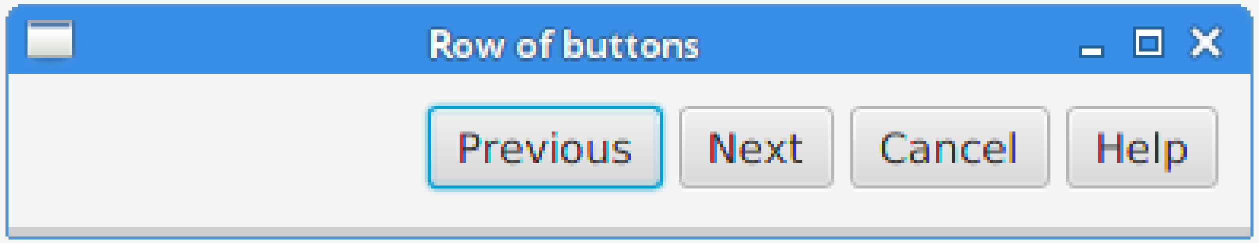
FlowPane

- FlowPane positions nodes in a row or a column, where the nodes are wrapped when they all cannot be shown.
- The default orientation of a flow pane is horizontal.
- FlowPane has a very limited usage.



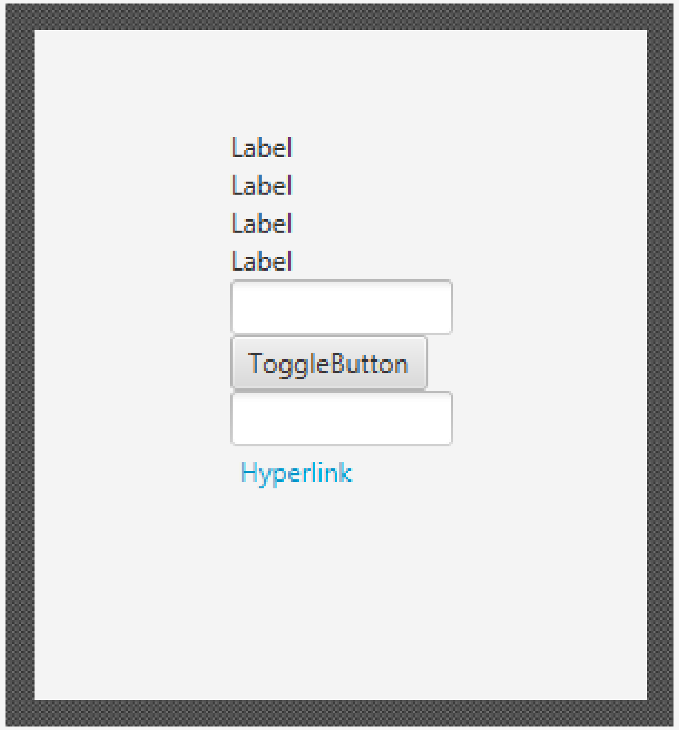
HBox

- HBox lays out its children in a single horizontal row.
- This pane is used in cooperation with other layout managers to create layouts
- It is suited for doing basic layouts.



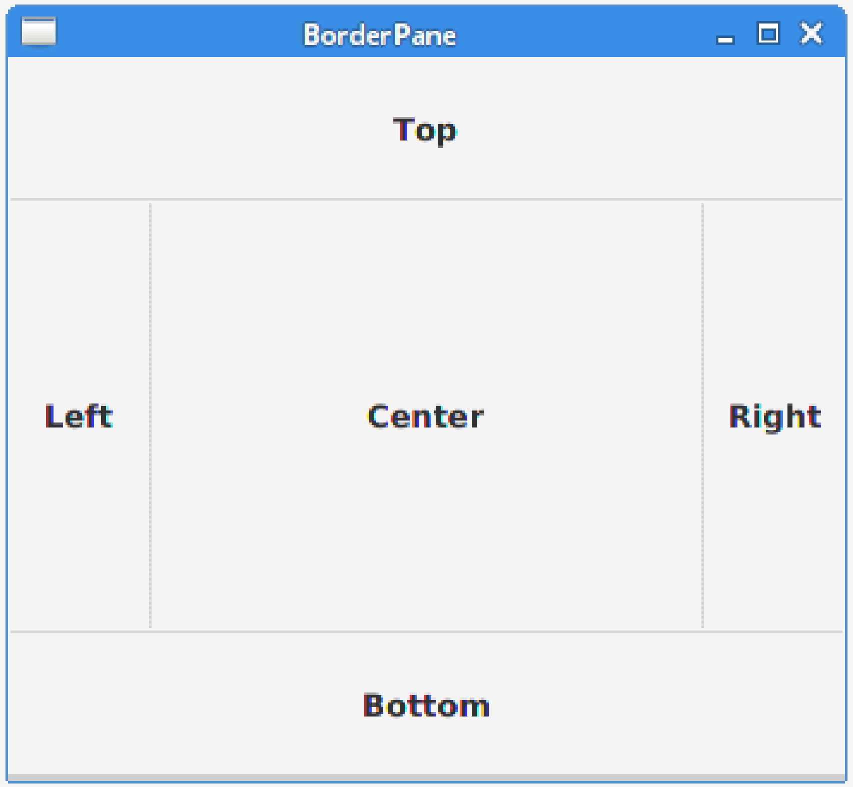
VBox

- VBox lays out its children in a single vertical column.
- This pane is also used in cooperation with other layout managers to create layouts



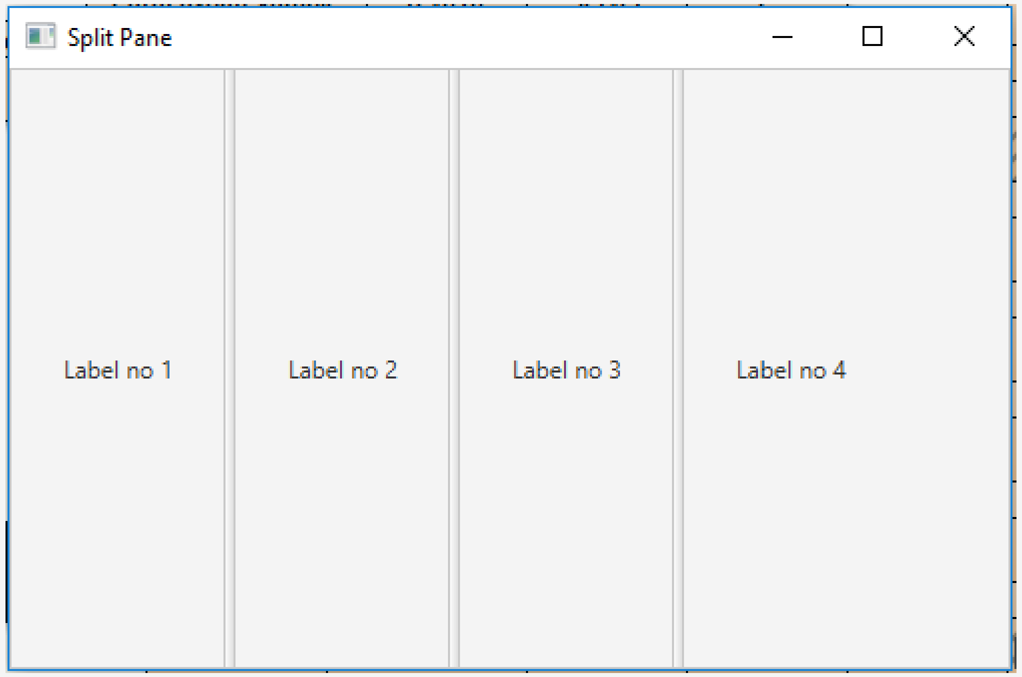
BorderPane

- BorderPane lays out children in top, left, right, bottom, and center positions.
- It can be used to create the classic looking application layouts.

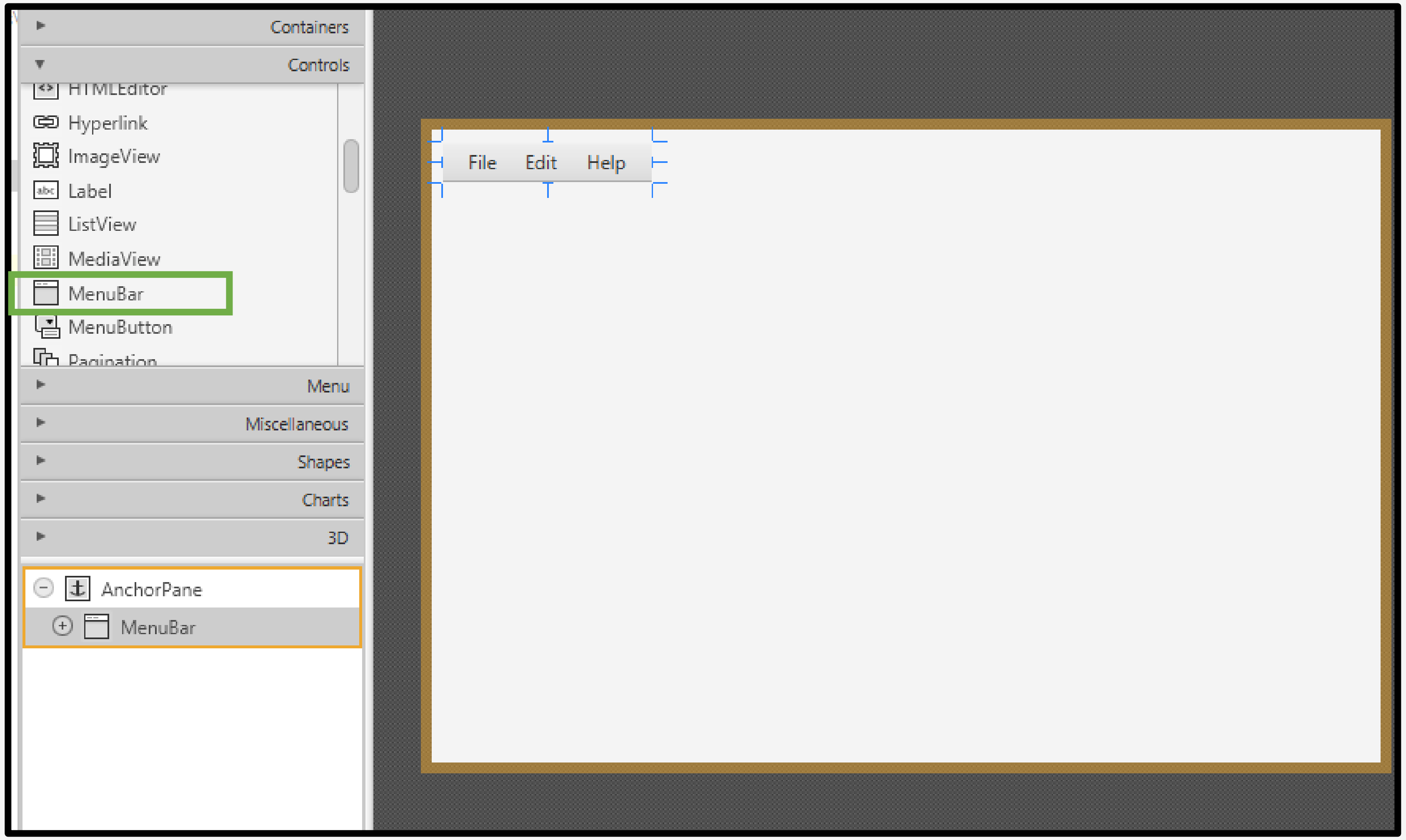


SplitPane

- SplitPane contains two or more sides separated by a divider.
- Sides can be dragged by the user to give more space to one of the sides which cause the other side to shrink by an equal amount.

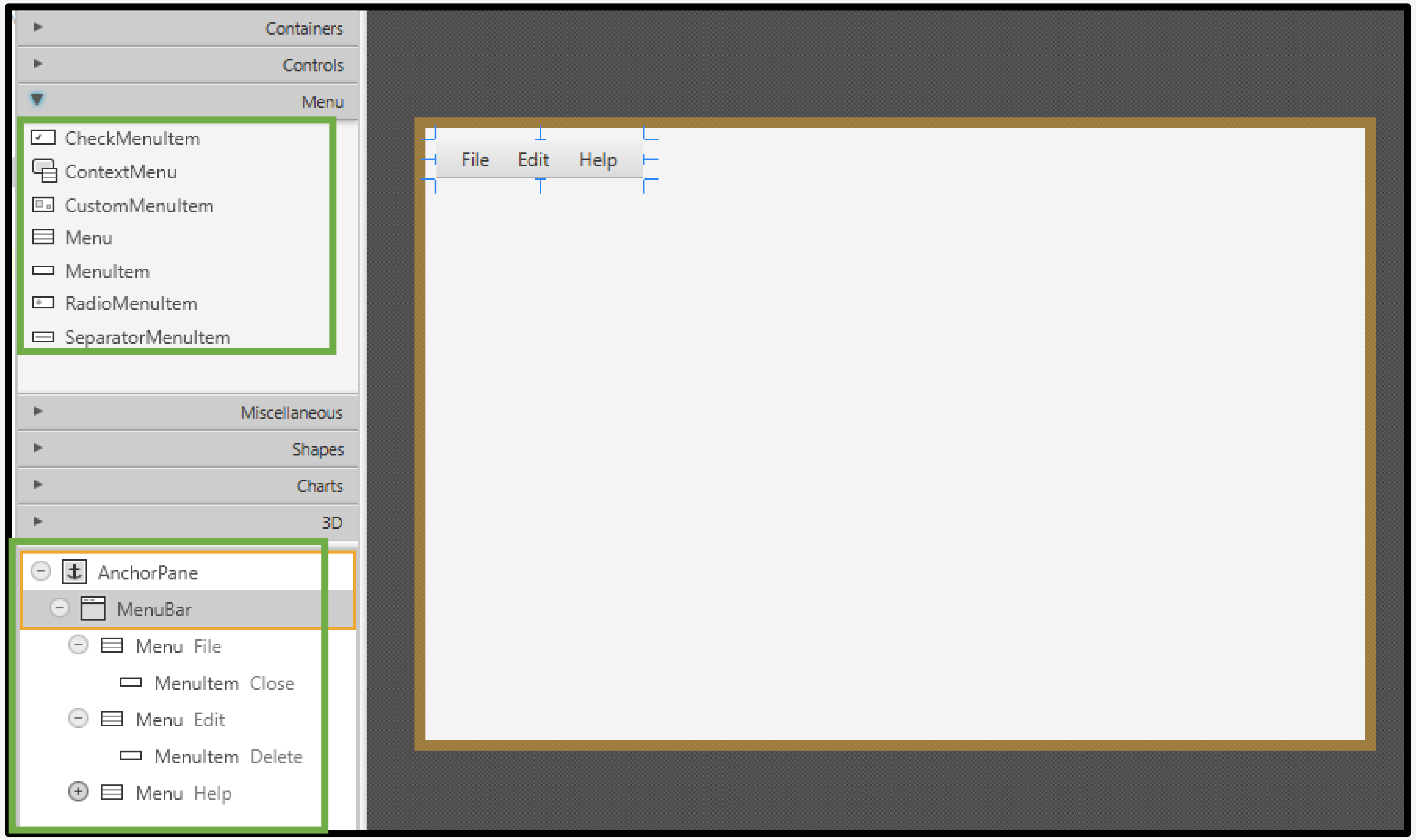


Working with Menus - 1



- Create a FXML file and a container of your choice.
- Add Menu Bar from the Controls

Working with Menus - 2



- You can view the available Menus and Menu Items.
- New Menu Items can be added from the Menu section of the JavaFX toolkit.

Thank You

Thank You