Layouts – FlowPane – HBox and VBox – BorderPane – StackPane – GridPane.

In JavaFX, Layout defines the way in which the components are to be seen on the stage. It basically organizes the scene-graph nodes.

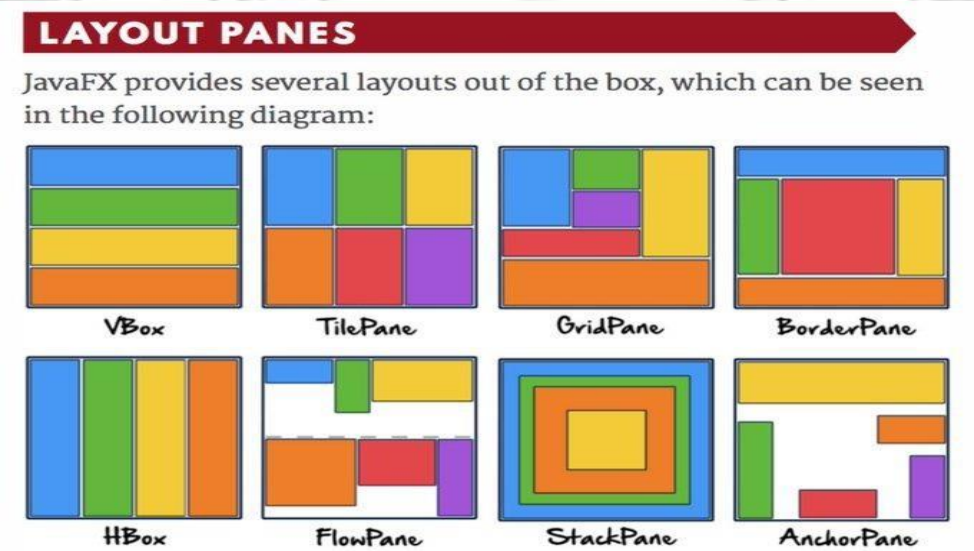
Layout Panes: Layout panes are containers which are used for flexible and dynamic arrangements of UI controls within a scene graph of a JavaFX application.

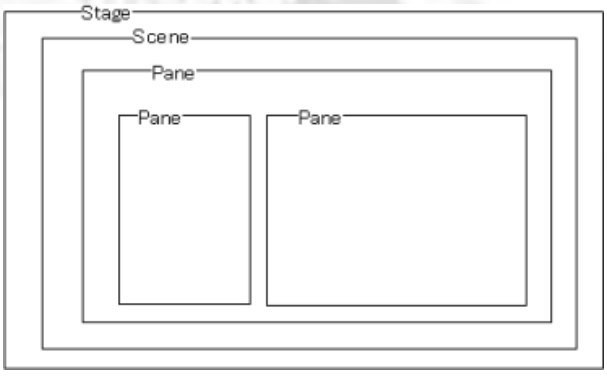
Package used: javaFX.scene.layout package

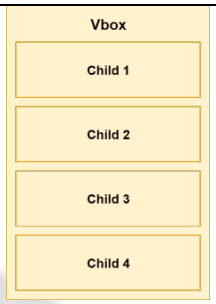
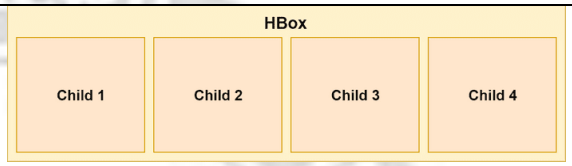
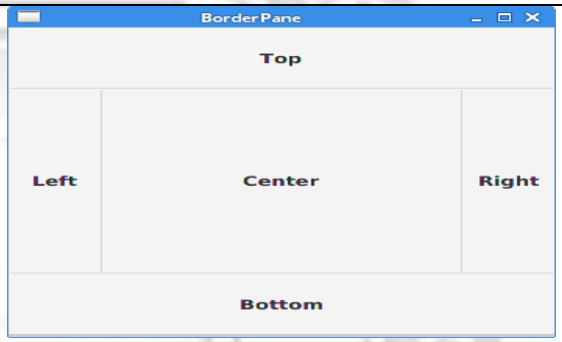
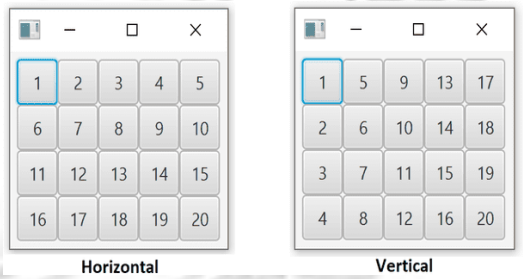
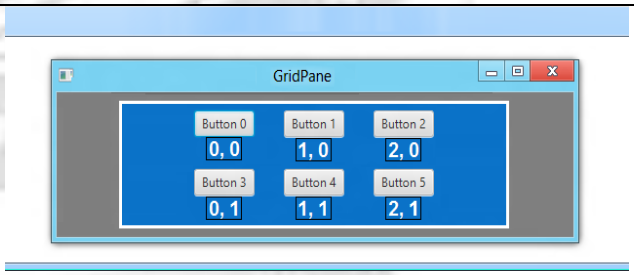
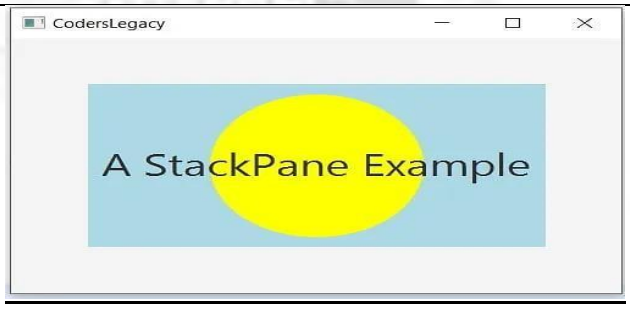
JavaFX provides various built-in Layouts that are

1. Pane
2. VBox
3. HBox
4. BorderPane
5. FlowPane
6. GridPane
7. StackPane.

JavaFX provides many types of panes for organizing nodes in a container:



Class	Description	Representation
Pane	Base class for layout panes. It containes the getChildren() method for returning a list of nodes in the pane.	

VBox	Places the nodes in a single column	
HBox	Places the nodes in a single row	
BorderPane	Places the nodes in the top, right, bottom, left and center regions	
FlowPane	Places the nodes row-by-row horizontally or column-by-column vertically	
GridPane	Places the nodes in the cells in a two-dimensional grid(like matrix)	
StackPane	Places the nodes on top of each other in the center of the pane	

Methods and Properties of different layouts:

<table>
<tr><th>Layout</th><th>Constructors</th><th>Methods/Properties</th></tr>
<tr><td>VBox</td><td>1. VBox() : creates layout with 0 spacing
2. Vbox(Double spacing) : creates layout with a spacing value of double type
3. Vbox(Double spacing, Node? children) : creates a layout with the specified spacing among the specified child nodes
4. Vbox(Node? children) : creates a layout with the specified nodes having 0 spacing among them</td><td>
<table>
<tr><th>Property</th><th>Description</th><th>Setter Methods</th></tr>
<tr><td>Alignment</td><td>This property is for the alignment of the nodes.</td><td>setAlignment(Double)</td></tr>
<tr><td>FillWidth</td><td>This property is of the boolean type. The Widtht of resizeable nodes can be made equal to the Width of the VBox by setting this property to true.</td><td>setFillWidth(boolean)</td></tr>
<tr><td>Spacing</td><td>This property is to set some spacing among the nodes of VBox.</td><td>setSpacing(Double)</td></tr>
</table>
</td></tr>
<tr><td>HBox</td><td>1. new HBox() : create HBox layout with 0 spacing
2. new Hbox(Double spacing) : create HBox layout with a spacing value</td><td>
<table>
<tr><th>Property</th><th>Description</th><th>Setter Methods</th></tr>
<tr><td>Alignment</td><td>This represents the alignment of the nodes.</td><td>setAlignment(Double)</td></tr>
<tr><td>fillHeight</td><td>This is a boolean property. If you set this property to true the height of the nodes will become equal to the height of the HBox.</td><td>setFillHeight(Double)</td></tr>
<tr><td>spacing</td><td>This represents the space between the nodes in the HBox. It is of double type.</td><td>setSpacing(Double)</td></tr>
</table>
</td></tr>
<tr><td>BorderPane</td><td>1. BorderPane() :- create the empty layout
2. BorderPane(Node Center):- create the layout with the center node
3. BorderPane(Node Center, Node top, Node right, Node bottom, Node left) :- create the layout with all the nodes</td><td>
<table>
<tr><th>Type</th><th>Property</th><th>Setter Methods</th><th>Description</th></tr>
<tr><td>Node</td><td>Bottom</td><td>setBottom()</td><td>Add the node to the bottom of the screen</td></tr>
<tr><td>Node</td><td>Centre</td><td>setCentre()</td><td>Add the node to the centre of the screen</td></tr>
<tr><td>Node</td><td>Left</td><td>setLeft()</td><td>Add the node to the left of the screen</td></tr>
<tr><td>Node</td><td>Right</td><td>setRight()</td><td>Add the node to the right of the screen</td></tr>
<tr><td>Node</td><td>Top</td><td>setTop()</td><td>Add the node to the top of the screen</td></tr>
</table>
</td></tr>
</table>

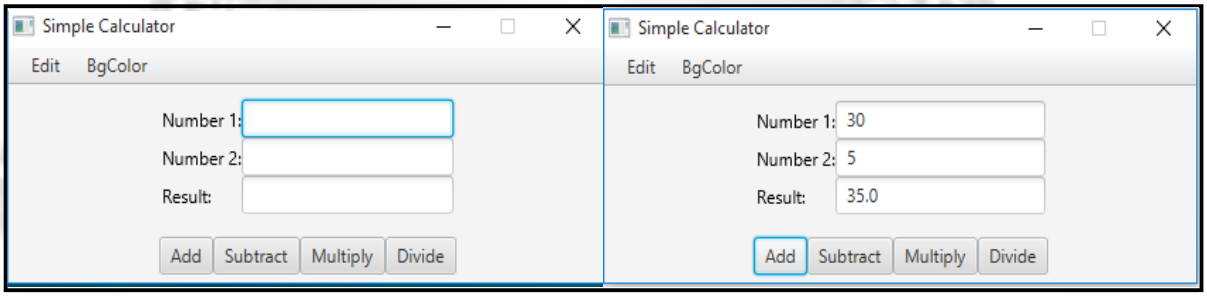
<table>
<tr>
<td>FlowPane</td>
<td>1. FlowPane()
2. FlowPane(Double Hgap, Double Vgap)
3. FlowPane(Double Hgap, Double Vgap, Node? children)
4. FlowPane(Node... Children)
5. FlowPane(Orientation orientation)
6. FlowPane(Orientation orientation, double Hgap, Double Vgap)</td>
<td>
<table>
<tr><th>Property</th><th>Description</th><th>Setter Methods</th></tr>
<tr><td>alignment</td><td>The overall alignment of the flowpane's content.</td><td>setAlignment(Pos value)</td></tr>
<tr><td>columnHalignment</td><td>The horizontal alignment of nodes within the columns.</td><td>setColumnHalignment(HPos Value)</td></tr>
<tr><td>hgap</td><td>Horizontal gap between the columns.</td><td>setHgap(Double value)</td></tr>
<tr><td>orientation</td><td>Orientation of the flowpane</td><td>setOrientation(Orientation value)</td></tr>
<tr><td>prefWrapLength</td><td>The preferred height or width where content should wrap in the horizontal or vertical flowpane.</td><td>setPrefWrapLength(double value)</td></tr>
<tr><td>rowValignment</td><td>The vertical alignment of the nodes within the rows.</td><td>setRowValignment(VPos value)</td></tr>
<tr><td>vgap</td><td>The vertical gap among the rows</td><td>setVgap(Double value)</td></tr>
</table>
</td>
</tr>
<tr>
<td>GridPane</td>
<td>Public GridPane(): creates a gridpane with 0 hgap/vgap.</td>
<td>
<table>
<tr><th>Property</th><th>Description</th><th>Setter Methods</th></tr>
<tr><td>alignment</td><td>Represents the alignment of the grid within the GridPane.</td><td>setAlignment(Pos value)</td></tr>
<tr><td>gridLinesVisible</td><td>This property is intended for debugging. Lines can be displayed to show the gidpane's rows and columns by setting this property to true.</td><td>setGridLinesVisible(Boolean value)</td></tr>
<tr><td>hgap</td><td>Horizontal gaps among the columns</td><td>setHgap(Double value)</td></tr>
<tr><td>vgap</td><td>Vertical gaps among the rows</td><td>setVgap(Double value)</td></tr>
</table>
</td>
</tr>
<tr>
<td>StackPane</td>
<td>1. StackPane()
2. StackPane(Node? Children)</td>
<td>
<table>
<tr><th>Property</th><th>Description</th><th>Setter Methods</th></tr>
<tr><td>alignment</td><td>It represents the default alignment of children within the StackPane's width and height</td><td>setAlignment(Node child, Pos value)</td></tr>
</table>
</td>
</tr>
</table>

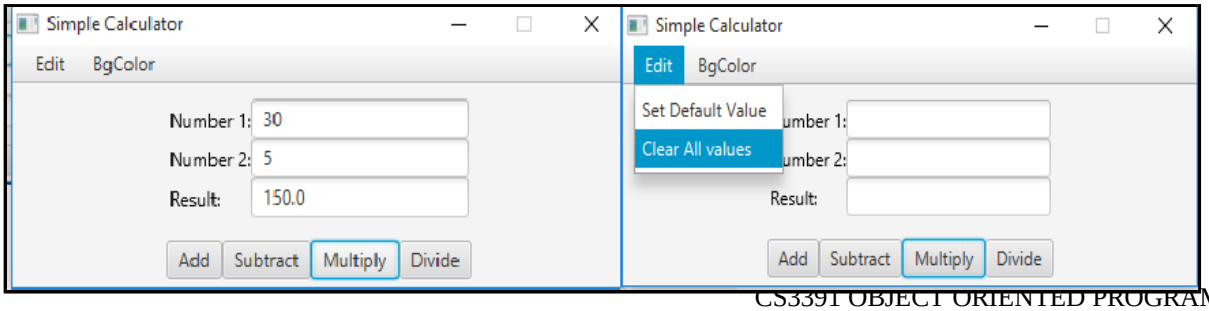
Example: Program for Layouts, Menus and MenuBars, Event Handling

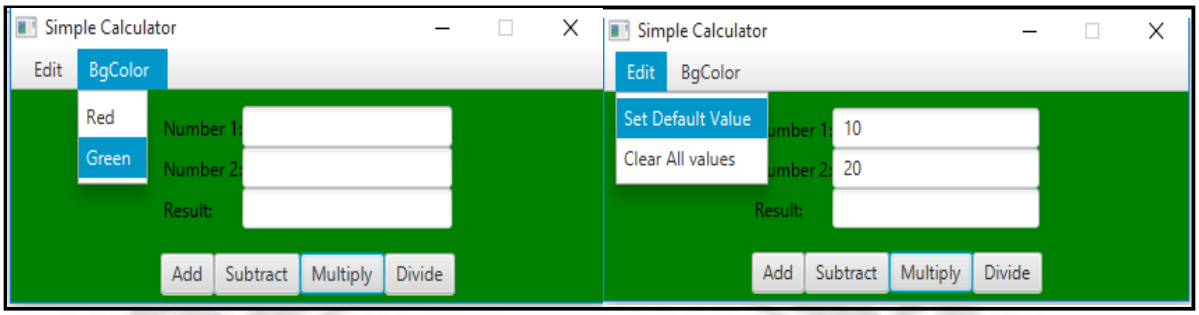
```
import javafx.application.Application;
import  javafx.geometry.Insets;
import javafx.geometry.Pos;
import javafx.scene.Scene;
import javafx.scene.control.*;
import javafx.scene.layout.BorderPane;
import javafx.scene.layout.*;
import javafx.scene.paint.Color;
import javafx.scene.text.Text;
import javafx.stage.Stage;

public class JavaFXApplication1 extends Application {
  @Override//from ww w .  ja v a2s  . co m

  public void start(Stage primaryStage) {
    // Create the top section of the UI
    Text tNumber1 = new Text("Number 1:");
    Text tNumber2 = new Text("Number 2:");
    Text tResult = new Text("Result:");
    TextField tfNumber1 = new TextField();
    TextField tfNumber2 = new TextField();
    TextField tfResult = new TextField();
    tfResult.setEditable(false);
    Menu me=new Menu("Edit");

    // create menuitems
     MenuItem m1 = new MenuItem("Set Default Value");
     MenuItem m2 = new MenuItem("Clear All values");

     // add menu items to menu
     me.getItems().add(m1);
     me.getItems().add(m2);
    Menu mc=new Menu("Bg_Color");
    MenuItem c1 = new MenuItem("Red");
    MenuItem c2 = new MenuItem("Green");
    mc.getItems().addAll(c1,c2);
    MenuBar mb = new MenuBar();

      // add menu to menubar
    mb.getMenus().add(me);
    mb.getMenus().add(mc);
    VBox vb=new VBox(mb);
    m1.setOnAction(e -> {
       tfNumber1.setText("10");
       tfNumber2.setText("20");
    });
```

```
    m2.setOnAction(e ->{
       tfNumber1.setText("");
       tfNumber2.setText("");
       tfResult.setText("");
    });

   // Create the bottom section of the UI
    Button btAdd = new Button("Add");
    Button btSubtract = new Button("Subtract");
    Button btMultiply = new Button("Multiply");
    Button btDivide = new Button("Divide");

    // Add top and bottom UI to HBox containers
    GridPane calcTop = new GridPane();
    calcTop.setAlignment(Pos.CENTER);
    calcTop.setPadding(new Insets(5));
    calcTop.add(tNumber1, 0, 0);
    calcTop.add(tfNumber1, 1, 0);
    calcTop.add(tNumber2, 0, 1);
    calcTop.add(tfNumber2, 1, 1);
    calcTop.add(tResult, 0, 2);
    calcTop.add(tfResult, 1, 2);
    FlowPane calcBottom = new FlowPane();
    calcBottom.setAlignment(Pos.CENTER);
    calcBottom.setPadding(new Insets(5));
    calcBottom.getChildren().addAll(btAdd, btSubtract, btMultiply, btDivide);

    // Add HBox containers to a BorderPane
    BorderPane pane = new BorderPane();
    pane.setTop(vb);
    pane.setCenter(calcTop);
    pane.setBottom(calcBottom);
    c1.setOnAction(e -> {
       pane.setBackground(new Background(new BackgroundFill(Color.RED,null,null)));
    });
    c2.setOnAction(e -> {
       pane.setBackground(new Background(new
BackgroundFill(Color.GREEN,null,null)));
    });

    // Register event handlers for buttons
    btAdd.setOnAction(e -> {
      double a = getDoubleFromTextField(tfNumber1);
      double b = getDoubleFromTextField(tfNumber2);
      tfResult.setText(String.valueOf(a + b));
    });
```

```
  btSubtract.setOnAction(e -> {
    double a = getDoubleFromTextField(tfNumber1);
    double b = getDoubleFromTextField(tfNumber2);
    tfResult.setText(String.valueOf(a - b));
  });
  btMultiply.setOnAction(e -> {
    double a = getDoubleFromTextField(tfNumber1);
    double b = getDoubleFromTextField(tfNumber2);
    tfResult.setText(String.valueOf(a * b));
  });
  btDivide.setOnAction(e -> {
    double a = getDoubleFromTextField(tfNumber1);
    double b = getDoubleFromTextField(tfNumber2);
    tfResult.setText(b == 0 ? "NaN" : String.valueOf(a / b));
  });

  Scene scene = new Scene(pane);
  primaryStage.setTitle("Simple Calculator");
  primaryStage.setScene(scene);
  primaryStage.setResizable(false);
  primaryStage.show();
}

private static double getDoubleFromTextField(TextField t) {
  return Double.parseDouble(t.getText());
}

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






Simple Calculator
Edit
BgColor
Red
Green
Number 1:
Number 2:
Result:
Add
Subtract
Multiply
Divide
Simple Calculator
Edit
BgColor
Set Default Value
Clear All values
10
20
Result:
Add
Subtract
Multiply
Divide