# Experiment No: 8

## AIM: To learn JAVA FX UI Controls.

**Date:**

**CO mapped: CO-5**

**Objectives:**

a) To gain proficiency in JavaFX UI controls, including understanding their features and capabilities, and developing the ability to create interactive and visually appealing user interfaces for Java applications. This knowledge will enable the design and development of user-friendly, responsive, and feature-rich graphical user interfaces (GUIs) in Java applications.
b) Learning JavaFX UI controls is essential for creating modern and engaging graphical user interfaces for Java applications. This objective emphasizes not only understanding the various UI controls but also the practical skills to design and implement user interfaces effectively.

## Background:

Every user interface considers the following three main aspects –

UI elements − These are the core visual elements that the user eventually sees and interacts with. JavaFX provides a huge list of widely used and common elements varying from basic to complex, which we will cover in this practical.
Layouts − They define how UI elements should be organized on the screen and provide a final look and feel to the GUI (Graphical User Interface). This part will be covered in the Layout chapter.
Behavior − These are events that occur when the user interacts with UI elements.
JavaFX provides several classes in the package javafx.scene.control. To create various GUI components (controls), JavaFX supports several controls such as date picker, button text field, etc.
Each control is represented by a class; you can create a control by instantiating its respective class.

Common elements in a JavaFX application

All JavaFX applications contain the following elements:

1. A main window, called a stage in JavaFX.
2. At least one Scene in the stage.
3. A system of panes and boxes to organize GUI elements in the scene.
4. One or more GUI elements, such as buttons and labels.

The usual procedure for setting up a scene is to build it from the bottom up. First, we make the GUI elements, then we make boxes and panes to organize the elements, and finally, we put everything in the scene.

All JavaFX elements such as boxes and panes that are meant to contain other elements have a child list that we can access via the getChildren() method. We put elements inside other

elements by adding things to child lists. In the code above you can see the button and the label objects being added as children of a VBox, and the VBox, in turn, is set as the child of a StackPane.

In addition to setting the structure for the window, we also call methods designed to set the properties of various elements. For example, the code in this example uses the button's setText() method to set the text the button will display.

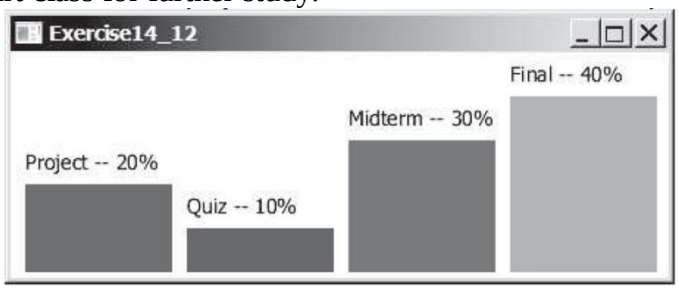
Follow the procedure outlined in the section above to make a new JavaFX application. Replace the start() method in the App class with the following code:

```
public void start(Stage primaryStage) {
   Button btn = new Button();
btn.setText("Say 'Hello World'");

StackPane root = new StackPane();
VBox box = new VBox();
box.getChildren().add(btn);
   Label label = new Label();
box.getChildren().add(label);
root.getChildren().add(box);

btn.setOnAction(new ClickHandler(label));

   Scene scene = new Scene(root, 300, 250);

primaryStage.setTitle("Hello World!");
primaryStage.setScene(scene);
primaryStage.show();
}
```

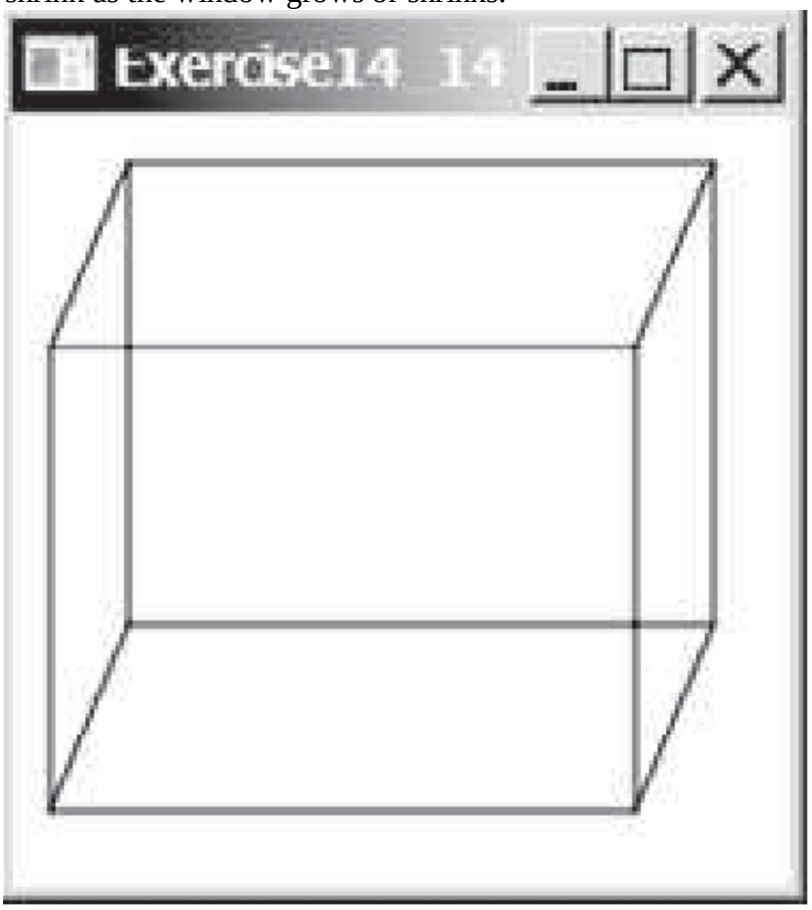
## Practical questions:

1. Write a program that displays five texts vertically, as shown in Figure. Set a random color and opacity for each text and set the font of each text to Times Roman, bold, italic, and 22 pixels.


2. Write a program that uses a bar chart to display the percentages of the overall grade represented by projects, quizzes, midterm exams, and the final exam, as shown in Figure b. Suppose that projects take 20 percent and are displayed in red, quizzes take 10 percent and are displayed in blue, midterm exams take 30 percent and are displayed in green, and the final exam takes 40 percent and is displayed in orange. Use the Rectangle class to display the bars. Interested readers may explore the JavaFXBarChart class for further study.




3. Write a program that displays a rectanguloid, as shown in Figure a. The cube should grow and shrink as the window grows or shrinks.



**Observations:** Put Output of the program

**Conclusion:** (Sufficient space to be provided)

**Quiz:** (Sufficient space to be provided for the answers)

1. Explain the evolution of Java GUI technologies since awt,swing and JavaFX.
2. What is the purpose of a TextField control, and how can it be used to collect user input?
3. How to create an ImageView from an Image, or directly from a file or a URL?.
4. What are the primary layout controls in JavaFX, and how do they impact the arrangement of UI components?
5. What is CSS, and how is it used for styling JavaFX UI controls?

**Suggested Reference:**


1. https://www.tutorialspoint.com/java/
2. https://www.geeksforgeeks.org/
3. https://www.w3schools.com/java/
4. https://www.javatpoint.com/