East West University
Department of Computer Science and Engineering

CSE 110: LAB 10 (Handout)
Course Instructor: Dr. Mohammad Rezwanul Huq

JavaFX Basics and Installing Scene Builder 2.0

Lab Objective

Familiarize students with JavaFX basics to develop Java GUI Programs.

Lab Outcomes

After completing this lab successfully, students will be able to:

1. **Understand** the basic structure of a JavaFX program.
2. **Implement** a basic JavaFX program.
3. **Implement** a few Java GUI programs using the concept of Layout Panes, Shapes, Color and Font class.

Psychomotor Learning Levels

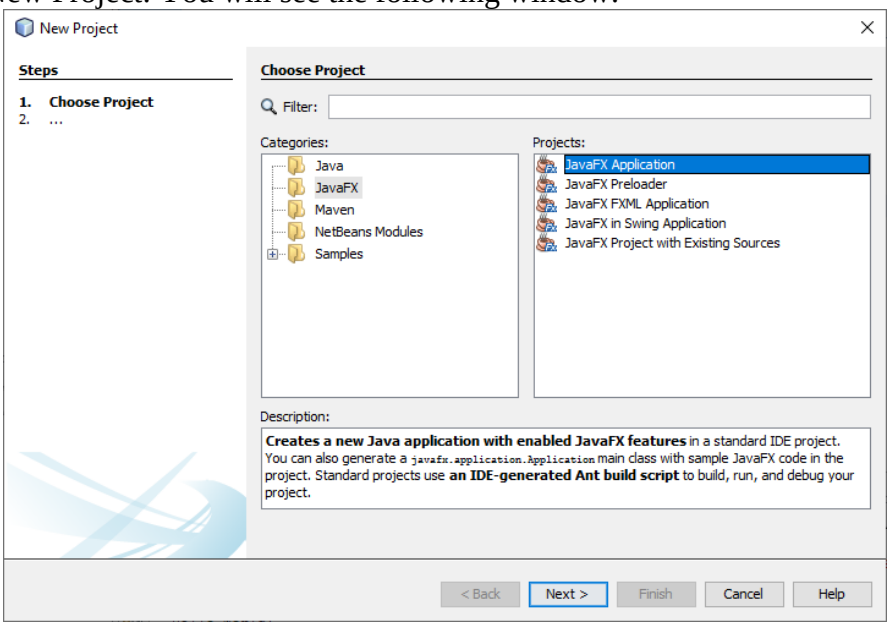
This lab involves activities that encompass the following learning levels in psychomotor domain.

Level	Category	Meaning	Keywords
P1	Imitation	Copy action of another; observe and replicate.	Relate, Repeat, Choose, Copy, Follow, Show, Identify, Isolate.
P2	Manipulation	Reproduce activity from instruction or memory	Copy, response, trace, Show, Start, Perform, Execute, Recreate.

Lab Activities

A. Creating a "Hello World" JavaFX Application

- Open NetBeans IDE.
- Go to File > New Project. You will see the following window.



- Choose JavaFX and JavaFX Application and then click Next.
- Do the rest of the things as per the usual procedure.
- The following codes will be automatically generated.

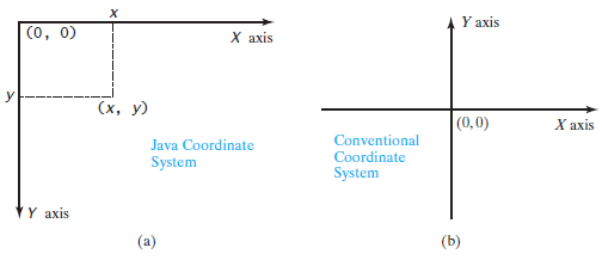
```
import javafx.application.Application;
import javafx.event.ActionEvent;
import javafx.event.EventHandler;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.layout.StackPane;
import javafx.stage.Stage;

public class JavaFXApplication1 extends Application {
    @Override
    public void start(Stage primaryStage) {
        Button btn = new Button();
        btn.setText("Say 'Hello World'");
        btn.setOnAction(new EventHandler<ActionEvent>() {
            @Override
            public void handle(ActionEvent event) {
                System.out.println("Hello World!");
            }
        });
        StackPane root = new StackPane();
        root.getChildren().add(btn);
        Scene scene = new Scene(root, 300, 250);
        primaryStage.setTitle("Hello World!");
        primaryStage.setScene(scene);
        primaryStage.show();
    }
    public static void main(String[] args) {
        launch(args);
    }
}
```

B. Creating a Circle

- **Circle** is a shape that can be displayed in a Java GUI Application.
- **Create** a new JavaFX Application.
- **[Lab Task] Add the following code to display a circle.**

```
Circle circle = new Circle();
circle.setCenterX(100);
circle.setCenterY(100);
circle.setRadius(50);
circle.setStroke(Color.BLACK);
circle.setFill(Color.WHITE);
```

C. Understanding Java Coordinate System



D. Using Colors through Color Class

- A color instance can be constructed using the following constructor:
  `public Color(double r, double g, double b, double opacity);`
- Example: **`Color color = new Color (0.25, 0.14, 0.333, 0.51);`**

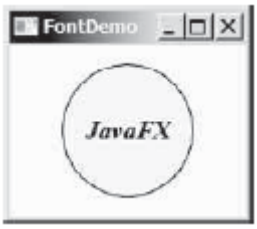
- You can also create a Color object using the static methods color(r, g, b), color(r,g, b, opacity), rgb(r, g, b), and rgb(r, g, b, opacity).
- Example: **`Color color = Color.rgb(0,0,255);`**

- Alternatively, you can use one of the many standard colors such as BEIGE, BLACK, BLUE, BROWN, CYAN, DARKGRAY, GOLD, GRAY, GREEN, LIGHTGRAY, MAGENTA, NAVY, ORANGE, PINK, RED, SILVER, WHITE, and YELLOW.
- Example: **`Color color = Color. BLUE;`**

E. Using Fonts through Font Class

- A Font instance can be constructed in different ways.
- **`Font font1 = new Font("SansSerif", 16);`**
- **`Font font2 = Font.font("Times New Roman", FontWeight.BOLD, FontPosture.ITALIC, 12);`**

- **[Lab Task] Create a JavaFX Application that shows the following:**



F. Displaying an Image through ImageView Class

- The Image class represents a graphical image.
- The ImageView class can be used to display an image.
- **`Image image = new Image("image/us.gif");`**
- **`ImageView imageView = new ImageView(image);`**

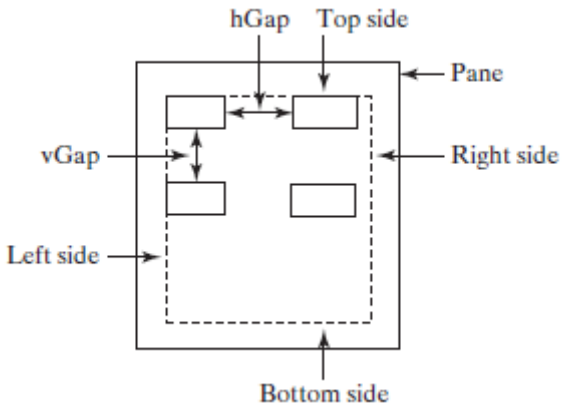
G. Controlling the positioning of nodes through Layout Panes

TABLE 14.1 Panes for Containing and Organizing Nodes

Class	*Description*
`Pane`	Base class for layout panes. It contains the `getChildren()` method for returning a list of nodes in the pane.
`StackPane`	Places the nodes on top of each other in the center of the pane.
`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.
`BorderPane`	Places the nodes in the top, right, bottom, left, and center regions.
`HBox`	Places the nodes in a single row.
`VBox`	Places the nodes in a single column.

- **Specifying Padding, Hgap and Vap:**

```
pane.setPadding(new Insets(11, 12, 13, 14));
pane.setHgap(5);
pane.setVgap(5);
```



- **[Lab Task] Create a window like the following using a GridPane:**

H. Installing JavaFX Scene Builder 2.0

- JavaFX Scene Builder is a visual layout tool that lets users quickly design JavaFX application user interfaces, without coding.
- Users can drag and drop UI components to a work area, modify their properties, apply style sheets, and the the result is an FXML file that can then be combined with a Java project by binding the UI to the application's logic.
- Download from this link: https://www.oracle.com/technetwork/java/javase/downloads/javafxscenebuilder-info-2157684.html
- **Integration with NetBeans:** Tools > Options> Java and then select JavaFX and choose the installation folder
- Useful Links for Scene Builder and JavaFX:

  https://netbeans.org/kb/trails/matisse.html?utm_source=netbeans&utm_campaign=welcomepage

  https://www.oracle.com/technetwork/java/javase/downloads/javafxscenebuilder-info-2157684.html

  https://code.makery.ch/library/javafx-tutorial/part1/