Chapter 14. JavaFX Basics

Objectives:

To write a simple JavaFX program and understand the relationship among **stages**, **scenes**, and **nodes** (§14.3).
To create user interfaces using **panes**, **UI controls**, and **shapes** (§14.4).
To create colors using the **Color class** (§14.7).
To create fonts using the **Font class** (§14.8).
To display text using the **Text** class and create shapes using **Line**, **Circle**, **Rectangle**, **Ellipse**, **Arc**, **Polygon**, and **Polyline** (§14.11).
To develop the reusable GUI component **ClockPane** for displaying an analog clock (§14.12).

Problem A
Draw an Ellipse in JavaFX.

1) Create a Java class (name of your choice) and inherit the Application class of the package javafx.application and implement the start() method of this class.

Ex:
```
public void start(Stage primaryStage){
 }
```

2) Create an instance of the Ellipse class, which belongs to the javafx.scene.shape package.
3) By using four setter methods, specify the x and y coordinates of the center of the Ellipse. The width of the Ellipse along the x axis and y axis (major and minor axises)of the circle by setting the properties CenterX, CenterY, RadiusX and RadiusY.
4) In the start() method, create an object(named root) of the class Group, which belongs to the package javafx.scene. Pass the Ellipse (node) object created in the previous step as a parameter to the constructor of the Group class. This should be done in order to add it to the group.
5) Create a Scene object of the class Scene which belongs to the package javafx.scene. To this class pass the Group object (root) created in the previous step. Also pass two double parameters representing height and width of the screen along with the object of the Group class.

Ex: `Scene scene = new Scene(group ,600, 300);`

6) Set the title to the stage using the setTitle() method of the Stage class.
7) Add a Scene object to the stage using the method setScene() of the class named Stage. Add the Scene object prepared in the previous step using this method.

Ex: `primaryStage.setScene(scene);`

8) Display the contents of the scene using the method show() of the Stage class.
9) Launch the JavaFX application by calling the static method launch() of the Application class from the main method.

Problem B

Cont. Problem A

Using Classes Text, Font and Color

1) Create a text by instantiating this class as follows −

Ex: Text text = new Text();

The class Text contains a property named text of string type, which represents the text that is to be created. After instantiating the Text class, you need to set value to this property using the **setText()** method.

2) Set text “This is my ellipse”

Also set the position (origin) of the text by specifying the values to the properties x and y using their respective setter methods namely **setX()** and **setY().**

By default, the text created by text class is of the font…, size…, and black in color.

You can change the font size and color of the text using the **setFont()** method. This method accepts an object of the **Font** class.

The class named **Font** of the package javafx.scene.text is used to define the font for the text. This class contains a static method named font().

This method accepts four parameters namely −

- family − This is of a String type and represents the family of the font that we want to apply to the text.
- weight − This property represents the weight of the font. It accepts 9 values, which are − FontWeight.BLACK, FontWeight.BOLD, FontWeight.EXTRA_BOLD, FontWeight.EXTRA_LIGHT, LIGHT, MEDIUM, NORMAL, SEMI_BOLD, THIN.
- posture − This property represents the font posture (regular or italic). It accepts two values FontPosture.REGULAR and FontPosture.ITALIC.
- size − This property is of type double and it represents the size of the font.

3) Set the font of your choice(different from example) to the text by using the following method −

Ex: text.setFont(Font.font("verdana", FontWeight.BOLD, FontPosture.REGULAR, 20)

4) Set the color to the text and the ellipse by using the setFill(), setStroke() methods.
5) Add text decorations (ex. Underline text)

Problem C

.Write two GUI programs using JavaFX to finish the following sub-tasks:

(a) Draw a detailed clock: Modify the **ClockPane** class provided to draw the clock with more details on the hours and minutes, as shown in Figure 1. Save the java file as ProblemCa.java.

(b) Displays two clocks. The hour, minute, and second values are 4, 20, 45 for the first clock and 22, 46, 15 for the second clock, as shown in Figure 2. Save the java file as ProblemCb.java.

The figures below show an example GUI of this program for subtask 1 and subtask 2.

If time allows, you can also create animation for a running clock.

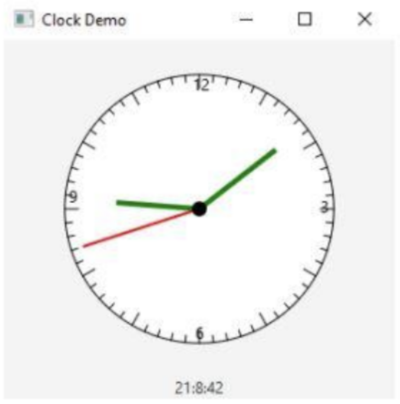

Figure 1: A detailed clock

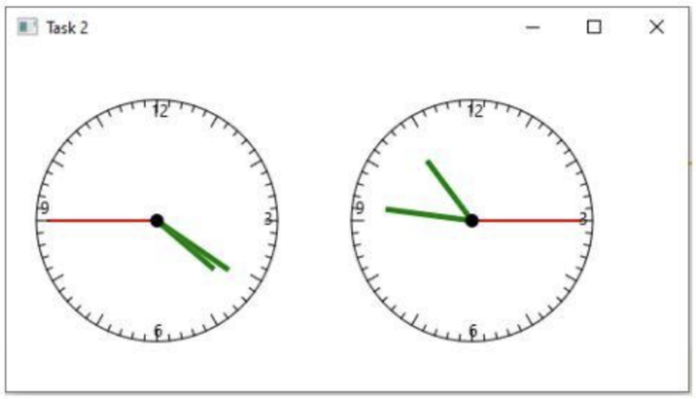

Figure 2: Two clocks