JavaFX

A brief reminder

- In JavaFX, a node is an object that represents a graphical element in the scene graph. A scene graph is a hierarchical representation of all the graphical elements in the scene, where each node in the graph represents a visual component such as a button, label, or image.
- There are many types of nodes in JavaFX, each with their own specific functionality. Here are some examples:
  1. Shapes: JavaFX provides a number of shape nodes, such as Rectangle, Circle, and Polygon, which can be used to draw geometrical shapes.
  2. Text: The Text node can be used to display text on the screen. It supports rich text formatting, including different fonts, styles, and colors.
  3. Images: The ImageView node can be used to display images in various formats such as JPG, PNG, or GIF.
  4. UI Controls: JavaFX provides a wide range of UI control nodes, such as Button, Label, TextField, TextArea, ComboBox, and more. These nodes allow developers to create interactive user interfaces.
  5. Containers: JavaFX provides a number of container nodes, such as HBox, VBox, StackPane, and GridPane, which can be used to lay out other nodes in the scene graph.
  6. Media: JavaFX provides nodes for playing and displaying media, such as MediaView, MediaPlayer, and MediaControl.
- So far we have learned the following JavaFX classes:
  1. Color class
  2. Font class
  3. Text class
  4. ImageView class
  5. Shape classes : Text, Line, Rectangle, Circle, Ellipse, Polygon, Polyline, Arc
- We have also learned how to use the following JavaFX layout panes:
  1. FlowPane
  2. BorderPane
  3. GridPane
  4. HBox
  5. VBox

Critical Thinking

1. Which of the following cannot be the root node of a scene graph in JavaFX?

   a) GridPane
   b) Canvas
   c) HBox
   d) VBox

answer: b

---

2. How does a BorderPane arrange the components that it contains?

   a) Horizontally next to each other
   b) Vertically next to each other
   c) On top of each other
   d) Center, right, left, top and bottom

answer: d

---

3. Which of the following image formats is not supported by the Image class?

   a) PNG
   b) SVG
   c) JPEG
   d) GIF

answer: b

---

4. Which of the following is not a valid way of creating a Color object?

   a) `Color c = Color.color(red, green, blue);`
   b) `Color c = Color.color(red, green, blue, opacity);`
   c) `Color c = new Color(red, green, blue);`
   d) `Color c = new Color(red, green, blue, opacity);`

answer: c

---

5. Which of the following is the signature of the start method?

   a) `public void start(Stage stage)`
   b) `public void start()`
   c) `public void start(Scene scene)`
   d) `public void start(Stage stage, Scene scene)`

answer: a

Practice

1. Modify the following code to display the text “Hello World” in bigger scene.

```
package application;

import javafx.application.Application; I
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.stage.Stage;
import javafx.scene.layout.StackPane;

public class Main extends Application {
    @verride
    public void start(Stage primaryStage) {
        try {
            Button btn1=new Button("Say, Hello World");
            StackPane root=new StackPane();
            root.getChildren().add(btn1);
            Scene scene=new Scene(root);
            primarystage.setScene(scene);
            primaryStage.setTitle("First JavaFX Application");
            primaryStage.show();
        } catch(Exception e) {
            e.printStackTrace();
        }
    }

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

answer:

```
package application;
import javafx.application.Application;
import javafx.event.ActionEvent;
import javafx.event.EventHandler;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.stage.Stage;
import javafx.scene.layout.StackPane;
public class Main extends Application{

    @Override
    public void start(Stage primaryStage) throws Exception {
        Button btn1=new Button("Say, Hello World");
        StackPane root=new StackPane();
        root.getChildren().add(btn1);
        Scene scene=new Scene(root,600,400);
        primaryStage.setScene(scene);
        primaryStage.setTitle("First JavaFX Application");
        primaryStage.show();
    }
}
```

---

2. Write an event handler for the following app so that pressing the button increments the number displayed by the label by one.

```java
public void start(Stage stage) {
    Button button = new Button("count");
    Label label = new Label("0");

    VBox root = new VBox(label, button);
    root.setAlignment(Pos.CENTER);

    button.setOnAction(e -> {
        // your code
    });

    Scene scene = new Scene(root, 100, 100);
    stage.setScene(scene);
    stage.setResizable(false);
    stage.show();
}
```

answer:

```java
button.setOnAction(e -> {
    int cur = Integer.parseInt(label.getText());
    label.setText(cur + 1 + "");
});
```

---

3. Write a simple app which consists of a single button. A new stage should be created every time the button is pressed.

answer:

```java
public void start(Stage stage) {
    Button button = new Button("new window");
    BorderPane root = new BorderPane(button);

    button.setOnAction(e -> {
    new Stage().show();
    });

    Scene scene = new Scene(root, 100, 100);
    stage.setScene(scene);
    stage.show();
}
```

Project

1. The following app annoys its user by tempting user to click a button and moves the button as soon as cursor hovers over it.

Modify it and make it even more annoying. Make sure that the button escapes to a random corner of the window instead of just right-left.

```java
public void start(Stage stage) {
    Button b = new Button("Click if you can!");
    HBox root = new HBox(b);
    root.setAlignment(Pos.CENTER_RIGHT);
    b.setOnMouseEntered(new EventHandler<Event>() {
        @Override
        public void handle(Event e) {
            if(root.getAlignment() == Pos.CENTER_RIGHT)
                root.setAlignment(Pos.CENTER_LEFT);
            else
                root.setAlignment(Pos.CENTER_RIGHT);
        }
    });
    Scene scene = new Scene(root, 500, 300);
    stage.setScene(scene);
    stage.setResizable(false);
    stage.show();
}
```

answer:

```java
@Override
public void start(Stage stage) {
    Button b = new Button("Click if you can!");
    HBox root = new HBox(b);
    root.setAlignment(Pos.CENTER_RIGHT);
    Set<Pos> unwanted = new HashSet<>();
    unwanted.add(Pos.BASELINE_CENTER);
    unwanted.add(Pos.BASELINE_LEFT);
    unwanted.add(Pos.BASELINE_RIGHT);
    b.setOnMouseEntered(new EventHandler<MouseEvent>() {
        @Override
        public void handle(MouseEvent e) {
            Pos initP = root.getAlignment();
            Pos p;
            unwanted.add(initP);
            while(unwanted.contains( p = pickRandom(Pos.values()) ));
            root.setAlignment(p);
            unwanted.remove(initP);
        }
    });
    Scene scene = new Scene(root, 500, 300);
    stage.setScene(scene);
    stage.setResizable(false);
    stage.show();
}

public <T> T pickRandom(T[] arr) {
    int randIndex = (int)
        (Math.random()*arr.length);
    return arr[randIndex];
}
```

2. The following app shows a ball tracing your cursor while leaving mark behind. Observe that the top-left corner of the ball coincides with the cursor at the stable state. Modify the program so that stabilization occurs at the center of the ball instead.

```
double mouseX, mouseY;
double ballX, ballY;
double ballSpeed = .1;
double ballSize = 30;
Color bg = Color.WHITE;
Color ballColor = Color.BLACK;
public void start(Stage stage) {
    Pane root = new Pane();
    Scene scene = new Scene(root);
    Canvas canvas = new Canvas(300, 300);
    root.getChildren().add(canvas);
    GraphicsContext g = canvas.getGraphicsContext2D();
    canvas.setOnMouseMoved(e -> {
        mouseX = e.getX();
        mouseY = e.getY();
    });
    AnimationTimer timer = new AnimationTimer() {
        public void handle(long now) {
            // erase old ball
            g.clearRect(0, 0, canvas.getWidth(), canvas.getHeight());
            // draw new ball
            ballX += (mouseX - ballX) * ballSpeed;
            ballY += (mouseY - ballY) * ballSpeed;
            g.setFill(ballColor);
            g.fillOval(ballX, ballY, ballSize, ballSize);
        }
    };
    timer.start();
    stage.setScene(scene);
    stage.show();
}
```

answer:

```
double mouseX, mouseY;
double ballX, ballY;
double ballSpeed = .1;
double ballSize = 30;
Color bg = Color.WHITE;
Color ballColor = Color.BLACK;
public void start(Stage stage) {
    Pane root = new Pane();
    Scene scene = new Scene(root);
    Canvas canvas = new Canvas(300, 300);
    root.getChildren().add(canvas);
    GraphicsContext g = canvas.getGraphicsContext2D();
    canvas.setOnMouseMoved(e -> {
        mouseX = e.getX();
        mouseY = e.getY();
    });
    AnimationTimer timer = new AnimationTimer() {
        public void handle(long now) {
            // erase old ball
            g.clearRect(0, 0, width, height);
            // draw new ball
            ballX += (mouseX - ballX) * ballSpeed;
            ballY += (mouseY - ballY) * ballSpeed;
            g.setFill(ballColor);
            g.fillOval(ballX-ballSize/2, ballY-ballSize/2, ballSize, ballSize);
        }
    };
    timer.start();
    stage.setScene(scene);
    stage.show();
}
```

---

3. Observe that the speed of the ball depends on its distance from the cursor. Modify the app in the previous question so that the ball approaches the cursor with a constant speed instead.

answer:

```
double mouseX, mouseY;
double ballX, ballY;
double ballSpeed = .1;
double ballSize = 30;
Color bg = Color.WHITE;
Color ballColor = Color.BLACK;
public void start(Stage stage) {
    Pane root = new Pane();
    Scene scene = new Scene(root);
    Canvas canvas = new Canvas(300, 300);
    root.getChildren().add(canvas);
    GraphicsContext g = canvas.getGraphicsContext2D();
    canvas.setOnMouseMoved(e -> {
        mouseX = e.getX();
        mouseY = e.getY();
    });
    // avoid divide by zero
    ballX = 150;
    ballY = 150;
    AnimationTimer timer = new AnimationTimer() {
        public void handle(long now) {
            // erase old ball
            g.clearRect(0, 0, width, height);
            // draw new ball
            double normalizer = Math.sqrt(Math.pow((mouseX - ballX), 2) + Math.pow((mouseY -
                            ballY), 2));
            ballX += (mouseX - ballX)/normalizer * ballSpeed;
            ballY += (mouseY - ballY)/normalizer * ballSpeed;
            g.setFill(ballColor);
            g.fillOval(ballX-ballSize/2, ballY-ballSize/2, ballSize, ballSize);
        }
    };
    timer.start();
    stage.setScene(scene);
```

---

4. The following is a simple explorer app which traverses a directory tree and displays it visually. Modify it so that the font size decreases with deeper hierarchy level.

```
@Override
public void start(Stage stage) {
    Pane root = new Pane();
    Scene scene = new Scene(root, 500, 500);
```

```java
    File f = new File("path-to-a-dir");
    Accordion accordion = new Accordion();
    root.getChildren().add(accordion);
    dirTraversal(f, accordion);
    stage.setScene(scene);
    stage.show();
}
public void dirTraversal(File f, Accordion a) {
    if(f.isDirectory()) {
        Accordion aNew = new Accordion();
        a.getPanes().add(new TitledPane(f.getName(), aNew));
        for(File i: f.listFiles())
            dirTraversal(i, aNew);
    }
    else {
        TitledPane t = new TitledPane(f.getName(), null);
        t.setTextFill(Color.BROWN);
        a.getPanes().add(t);
    }
}
```

answer:

```java
@Override
public void start(Stage stage) {
    Pane root = new Pane();
    Scene scene = new Scene(root, 500, 500);
    File f = new File("path-to-a-dir");
    Accordion accordion = new Accordion();
    root.getChildren().add(accordion);
    dirTraversal(f, accordion, 0);
    stage.setScene(scene);
    stage.show();
}
public double computeFont(int depth) {
    return 16 - 1.5*depth;
}
public void dirTraversal(File f, Accordion a, int depth) {
    if(f.isDirectory()) {
        Accordion aNew = new Accordion();
        TitledPane t = new TitledPane(f.getName(), aNew);
        t.setFont(Font.font(computeFont(depth)));
        a.getPanes().add(t);
        for(File i: f.listFiles())
            dirTraversal(i, aNew, depth+1);
    }
    else {
        TitledPane t = new TitledPane(f.getName(), null);
        t.setTextFill(Color.BROWN);
        t.setFont(Font.font(computeFont(depth)));
        a.getPanes().add(t);
    }
}
```

Extra

1. modify the following code so user can pick custom font.

```
class TextEditor extends Application {
    final double width = 400;
    final double height = 300;

    TextArea text;
    BorderPane root;
    boolean ctrlPressed;

    public void start(Stage stage) {
        text = new TextArea();
        root = new BorderPane(text);
        ctrlPressed = false;

        text.setOnKeyPressed(e -> {
            if( e.getCode() == KeyCode.CONTROL )
                ctrlPressed = true;

            if( ctrlPressed)
                if(e.getCode() == KeyCode.S)
                    try {
                        save();
                    } catch (Exception e1) {
                        e1.printStackTrace();
                    }
                else if(e.getCode() == KeyCode.O)
                    try {
                        load(stage);
                    } catch (Exception e1) {
                        e1.printStackTrace();
                    }

        });

        text.setOnKeyReleased(e -> {
            if(e.getCode() == KeyCode.CONTROL)
                ctrlPressed = false;
        });

        finalize(stage);
    }

    public void load(Stage stage) throws Exception {
        FileChooser fc = new FileChooser();
        fc.setTitle("Open a file");
        File f = fc.showOpenDialog(stage);
        if(f == null)
            return;
        Scanner sc = new Scanner(f);
        StringBuilder s = new StringBuilder();
        while(sc.hasNextLine())
            s.append(sc.nextLine());

        text.setText(s.toString());
    }
    public void save() throws IOException {
        FileWriter out = new FileWriter("text.txt");
        out.write(text.getText());
        out.close();
    }

    public void finalize(Stage stage) {
        Scene scene = new Scene(root, width, height);
        stage.setScene(scene);
        stage.setTitle("Text Editor");
        stage.show();
    }
}
```

answer:

```
...
ComboBox<String> box = new ComboBox<>();
box.getItems().addAll("times new roman", "verdana",
        "arial");
root.setBottom(box);
BorderPane.setAlignment(box, Pos.CENTER);
box.setOnAction(e -> {
    String fontName = box.getValue();
    double fontSize = text.getFont().getSize();
    Font font = new Font(fontName, fontSize);
    text.setFont(font);
});
...
```