Name: Oluwatobi Tolulope Saliu

Student-ID: S00391162

ITEC627 Lab 1 Report

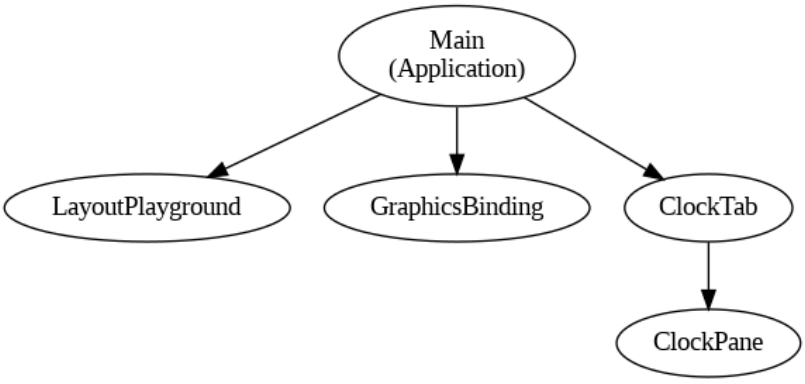
JavaFX Application Report

The project is a JavaFX app that demonstrates layouts, property binding, styling, images, fonts, and a reusable clock component. It is split into three tabs: Layout Playground, Graphics & Binding, and Clock Widget.

Design Overview

The app uses a Main class as the entry point with a TabPane for navigation.
LayoutPlayground shows layout panes, image loading, fonts, theme toggle, and a wrapping FlowPane.
 GraphicsBinding shows shapes, unidirectional and bidirectional property binding, rotation, and color changes.
ClockTab embeds the reusable ClockPane with start/stop and theme switching.

Architecture diagram



Binding Demonstrations

Unidirectional binding (circle radius in GraphicsBinding):
circle.radiusProperty().bind(radius.valueProperty());
The circle's radius changes when the slider moves.

Bidirectional binding (caption text in GraphicsBinding):
captionLabel.textProperty().bindBidirectional(captionField.textProperty());
Typing updates the label instantly, and changing the label updates the text field.

Styling & Rotate

The app supports light/dark themes in both the Layout Playground and Clock Widget tabs. The CSS file is swapped when a new theme is selected.

Theme toggle code block:

```
themes.valueProperty().addListener((o, oldV, newV) -> {
  getScene().getStylesheets().clear();
  String cssName = newV.equals("Dark") ? "/dark.css" : "/light.css";
  var css = getClass().getResource(cssName);
  if (css != null) getScene().getStylesheets().add(css.toExternalForm());
});
```

Rotation control in GraphicsBinding:

```
spinner.rotateProperty().bind(rotate.valueProperty());
```

The marker on the circle makes rotation visible.

Images & Error Handling

The image is loaded from /images/javaimage.png in resources. If the file is missing, an error alert is shown.

Image loading snippet From LayoutPlayground CLass:

```
try (InputStream is = getClass().getResourceAsStream("/images/javaimage.png")) {
  if (is == null) throw new IllegalArgumentException("Image not found.");
  imageView.setImage(new Image(is));
} catch (Exception ex) {
  new Alert(Alert.AlertType.ERROR, "Failed to load image: " +
ex.getMessage()).showAndWait();
}
```

Font loading:

```
try (InputStream fontIs = getClass().getResourceAsStream("/fonts/Quicksand-
SemiBold.ttf")) {
  Font f = Font.loadFont(fontIs, 22);
  imgHeader.setFont(f);
}
```

Reusable ClockPane

The ClockPane draws an analog clock with tick marks and three hands. It updates once per second via a Timeline and exposes public properties like hour, minute, second, running, and styleMode.

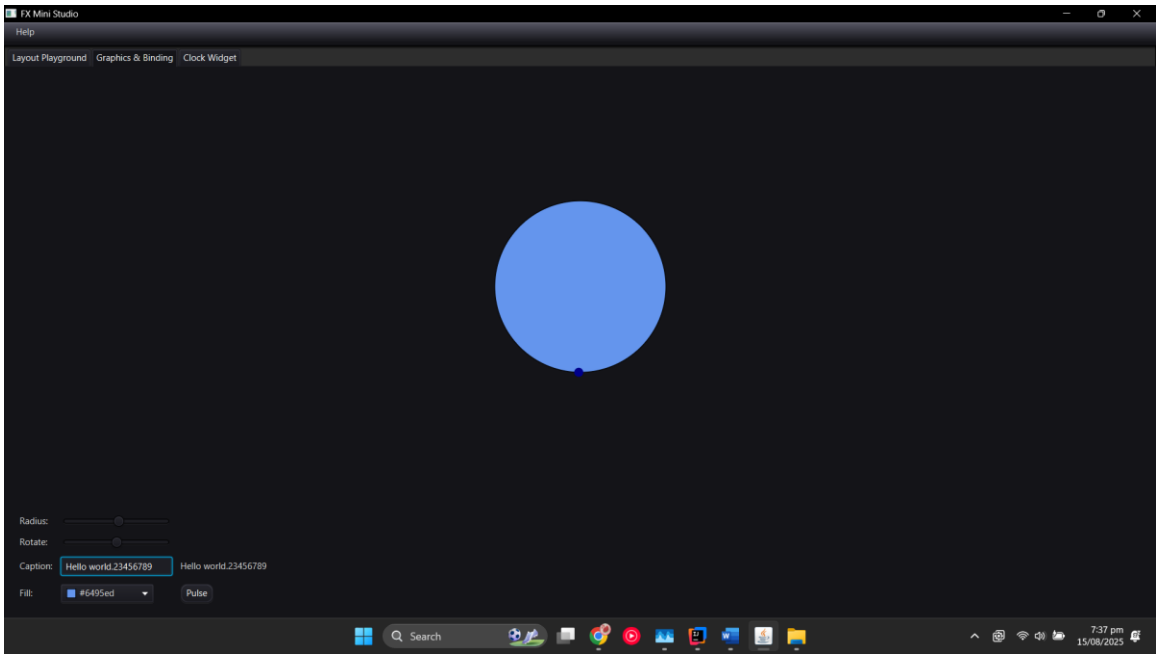
Example of Encapsulation:

```
public void start() { running.set(true); }
public void stop() { running.set(false); }
```

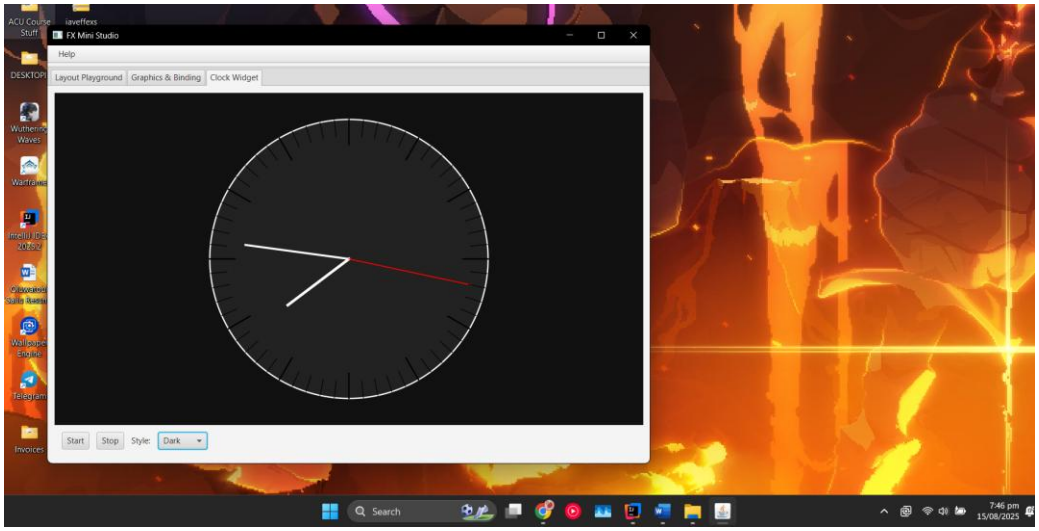
These let other parts of the app control the clock without knowing its internal code.

Testing Evidence

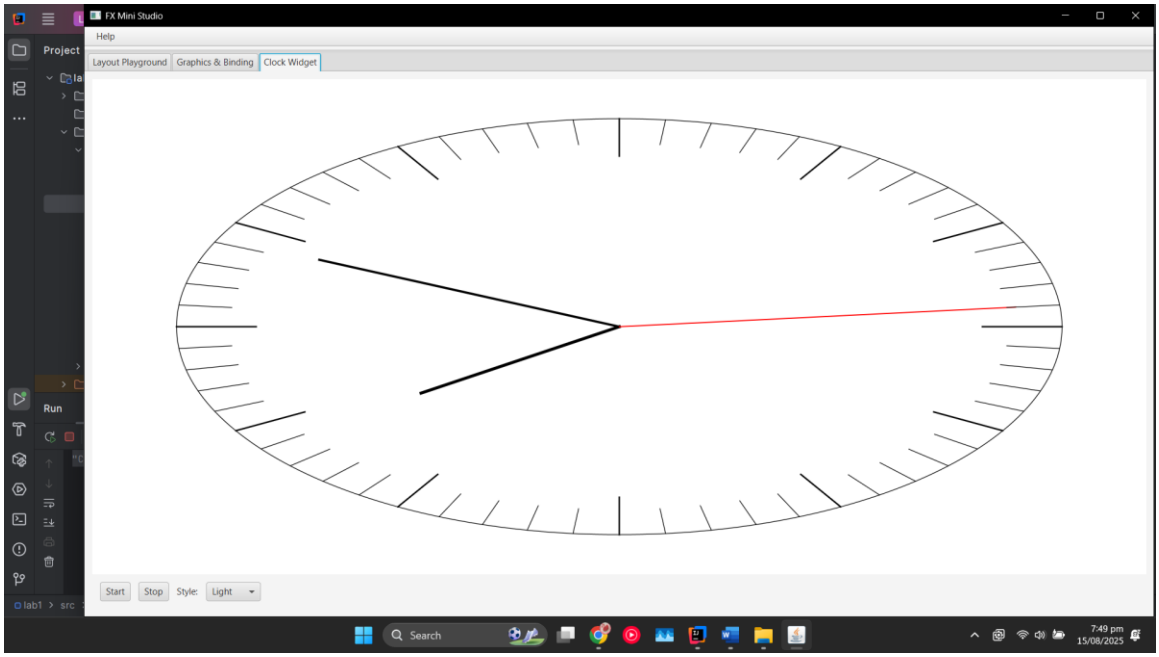
Layout Playground with Dark theme



Graphics and Binding tab showing rotation, Caption and label bidirectionally binded(Bound) in darkmode



Clock running in dark mode



Resized window showing responsive layout

Reflection

I Came away with the following things in the course of this Lab assignment- Learned how to use different layout panes.

- Practiced unidirectional and bidirectional property binding.
- Added basic theme switching with CSS.
- Used external font and handled missing files properly.
- Designed and reused a custom JavaFX control (ClockPane).
- Had to fix imports and dependencies when JavaFX wasn't detected.
- Found that a visible marker made rotation effects clearer.
- Improved understanding of responsive layouts in JavaFX.