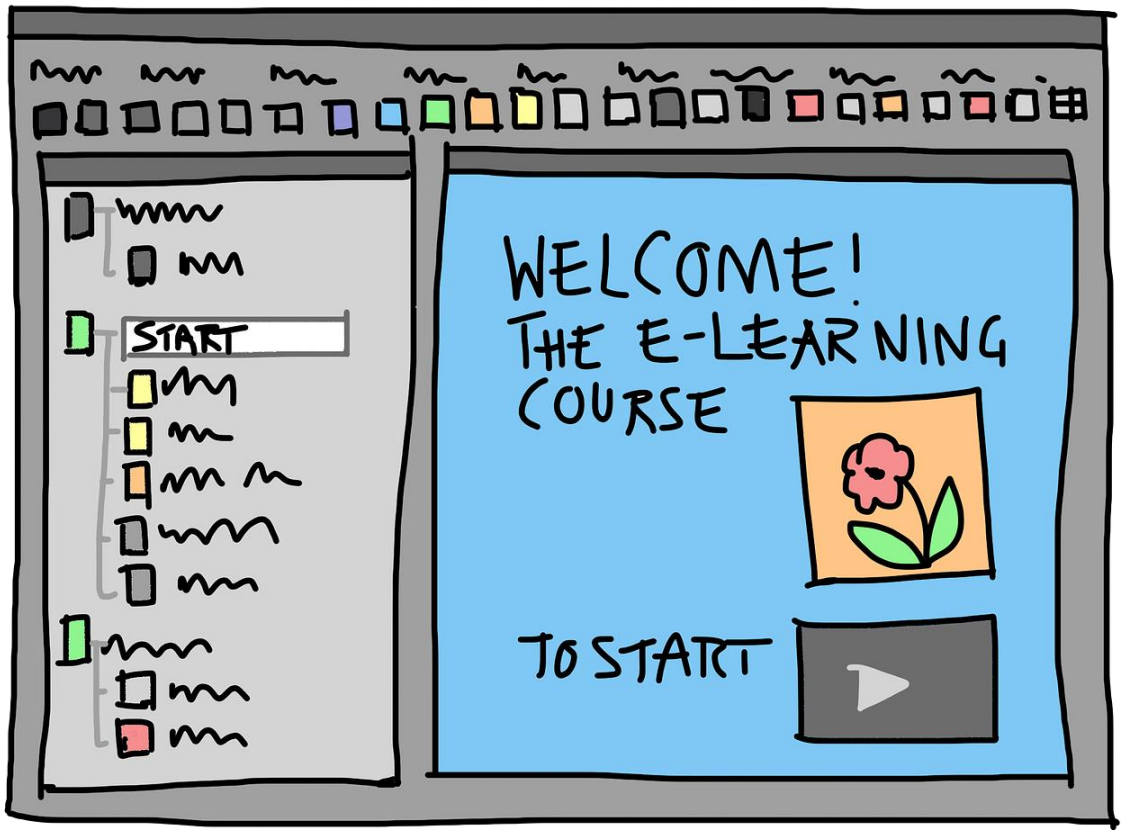


# ADVANCED JAVAFX AND GUI PROGRAMMING

OBJECT ORIENTED PROGRAMMING I

Sercan Külcü | Object Oriented Programming I | 10.01.2023

# Contents

# Introduction

JavaFX is a Java library used for creating desktop and mobile applications with a graphical user interface (GUI). It is built on top of Java's Swing toolkit and provides a rich set of graphical components, animations, and visual effects. In this chapter, we will explore advanced JavaFX concepts and best practices for GUI programming.

# JavaFX Architecture

JavaFX uses a scene graph to represent the visual elements of an application. The scene graph is a tree-like structure that contains nodes, each representing a visual component, such as buttons, labels, and text fields. The root node of the scene graph is the scene, which represents the top-level container of the application. The scene graph is built using JavaFX's application thread, which is responsible for rendering the GUI and handling user input events.

# Layouts and Controls

JavaFX provides a variety of layouts and controls to help you design your GUI. Layouts determine how the visual components are positioned and sized within a container. JavaFX supports several layout types, including BorderPane, FlowPane, GridPane, HBox, VBox, and StackPane. Controls are GUI components that are pre-built for specific tasks, such as buttons, text fields, sliders, and tables.

# Custom Controls

In addition to the pre-built controls, you can create custom controls to meet the specific needs of your application. Custom controls can be created by subclassing the Control or Region class and implementing the necessary methods. You can also use CSS to style your custom controls.

# Event Handling

JavaFX uses an event-driven architecture for handling user input events, such as button clicks and mouse movements. Event handling in JavaFX is done using the EventHandler interface, which defines a single method called handle(). To register an event handler for a GUI component, you can call the setOn<Action>() method, where <Action> is the name of the event you want to handle.

# Animation and Effects

JavaFX provides support for animation and visual effects, which can be used to enhance the user experience of your GUI. Animation in JavaFX is done using the Timeline class, which allows you to create an animation by specifying the duration, target node, and animation properties. Visual effects in JavaFX are done using the Effect class, which provides several built-in effects, such as DropShadow, InnerShadow, and Reflection.

## Best Practices

When designing a JavaFX application, it's important to follow best practices to ensure that your code is maintainable, scalable, and performant. Here are some best practices to keep in mind:

Use FXML to separate the GUI design from the application logic.

Keep your code modular and reusable by using a model-view-controller (MVC) architecture.

Use CSS to style your GUI components and maintain consistency across the application.

Optimize performance by minimizing the number of nodes in the scene graph and avoiding unnecessary bindings and listeners.

Test your GUI using automated testing frameworks, such as TestFX.

# Conclusion

JavaFX provides a powerful set of tools for GUI programming, with support for layout, controls, event handling, animation, and visual effects. By following best practices, you can create maintainable, scalable, and performant applications that meet the needs of your users.