JavaFX

Introduction to JavaFX

Dalton State College

T. Gonzalez

Graphical User Interfaces

A graphical user interface (GUI) is a form of user interface that allows users to interact with a device through graphical icons and visual indicators.

Most users expect to interact with computers using a GUI.

GUI programs are event-driven. User actions such as a clicking a mouse button or pressing a key on the keyboard generate events, and the program responds to the events as they occur.

JavaFX

JavaFX is a software platform for creating GUI desktop applications and internet applications.

JavaFX is intended to replace the older Java GUI platform, Swing.

We will be using JavaFX 8 in this class.

First JavaFX Example

See HelloWorldFX.java

Line 1: `import javafx.application.Application;`

The JavaFX classes we will write will be child classes of the `Application` class.

The `Application` class is the entry point for JavaFX applications.

Whenever a JavaFX application is launched, the JavaFX runtime does the following in order:

- Constructs an instance of the specified `Application` class
- Calls the `init()` method.
- Calls the `start()` method.
- Waits for the application to finish.
- Call the `stop()` method.

The `start()` method is abstract and must be overridden. We will use this method to do the setup of our application.

Line 2: `import javafx.stage.Stage;`

A `Stage` object represents a window on the computer screen.

Line 3: `import javafx.scene.Scene;`

A `Scene` object can be filled with GUI components such as buttons, menus, check boxes, and many more.

A `Stage` object is used to display a `Scene` object.

Line 4: `import javafx.scene.layout.Pane;`

`Pane` is an example of a GUI component known as a container. A container can contain other GUI components including other containers.

A **container** is any descendant of the `javafx.scene.Parent` class.

In our applications, `Scene` objects will have at least one container that contains all other components in the `Scene`. This component will be referred to as the **root component**.

There are many different containers that can be used as the root component of a `Scene`. This first example just happens to use `Pane`.

Line 7: `public class HelloWorldFX extends Application`

The `HelloWorldFX` class is a child class of the `Application` class.

Since the `start()` method of the `Application` class is `abstract`, our class must provide an implementation.

Line 10: `public void start(Stage stage)`

We will use the `start()` method to initialize applications.

The `start()` method has one parameter which is a `Stage` object.

The `Stage` object passed to the `start()` method is constructed by the system and represents the main window of the program.

Line 15: `Pane root = new Pane();`

This creates the `Pane` object that be the root component of the `Scene` that we will build.

The first example is very simple and will not contain any other GUI components, but in subsequent examples, we will start placing other GUI components into the root component.

Line 19: `Scene scene = new Scene(root, 400, 250);`

Create a `Scene` object to display on the `Stage`.

The `Scene` object needs a root component.

The numbers 400 and 250 set the width and the the height of the `Scene`, respectively. These arguments are optional.

Line 22: `stage.setScene(scene);`

Tells the `stage` object that it will be displaying the `scene` object.

Line 25: `stage.setTitle("Hello World!");`

Sets the title of the `stage` object.

Line 25: `stage.setTitle("Hello World!");`

Sets the title of the `stage` object.

Line 28: `stage.show();`

Tells the `stage` object to show itself.

Line 41: `Application.launch(args);`

Calls the `static` method `launch()` in the `Application` class to launch the application and passes the value of the array `args` to the application.

In turn, this will eventually trigger the `start()` method.