JavaFX Projects

- In Eclipse, New > Project > JavaFX > **JavaFX Project**
- A default class and CSS is created:

```
public class Main extends Application {
      @Override
      public void start(Stage primaryStage) {
            try {
                  BorderPane root = new BorderPane();
                  Scene scene = new Scene(root,400,400);
                  scene.getStylesheets().add(getClass().getResource("application.css").toExternalForm());
                  primaryStage.setScene(scene);
                  primaryStage.show();
            } catch(Exception e) {
                  e.printStackTrace();
            }
      }

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

```
public class Main extends Application {

	@Override
	public void start(Stage primaryStage) {
		try {
			BorderPane root = new BorderPane();

			Scene scene = new Scene(root,400,400);
			scene.getStylesheets().add(getClass().getResource("application.css").toExternalForm());

			primaryStage.setScene(scene);
			primaryStage.show();

		} catch(Exception e) {
			e.printStackTrace();
		}
	}


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

javafx.application.Application

BorderPane layout

Set up the scene & style sheet

Set the scene to the stage & show the application

Runs your application

A Brief History of Java GUIs

- Pronounced *“GOO-ee”*
- JDK 1.0: **AWT** (Abstract Window Toolkit)
- **SWT** (Standard Widget Toolkit) - developed by IBM, maintained by the Eclipse community.
  - *Not included as a default in Java.*
- **Swing** - developed by Oracle, as part of AWT.
  - *In maintenance mode only.*
- **JavaFX** - Java’s GUI, graphics, and multimedia API of the future. (-*Oracle*)

Swing versus JavaFX

- Both libraries can be leveraged to follow the MVC design pattern in your application.

- Therefore, both types of applications will have:
    - Separate packages for model, view, controller
    - Separate classes for models, views, controllers

- Conceptually building some components will be the same..

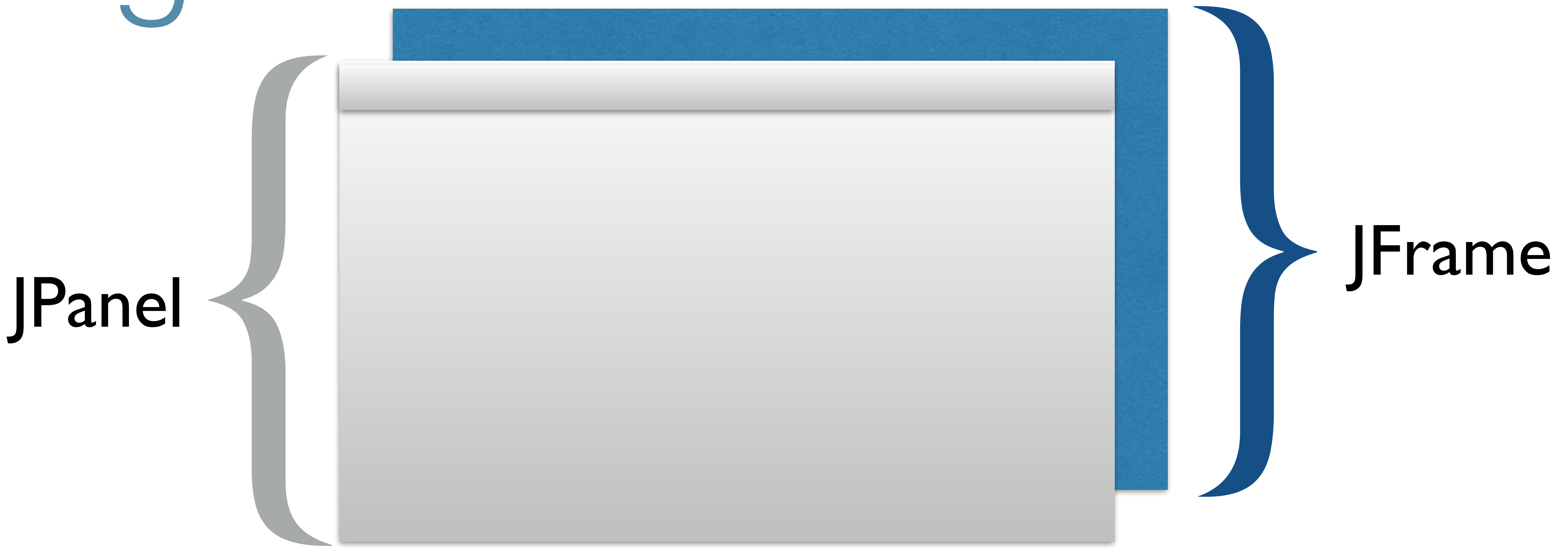
Swing versus JavaFX

- Models in a Swing project and a JavaFX project are exactly the same.
    - *These are classes meant to represent data, which is <u>not</u> dependent on the library.*

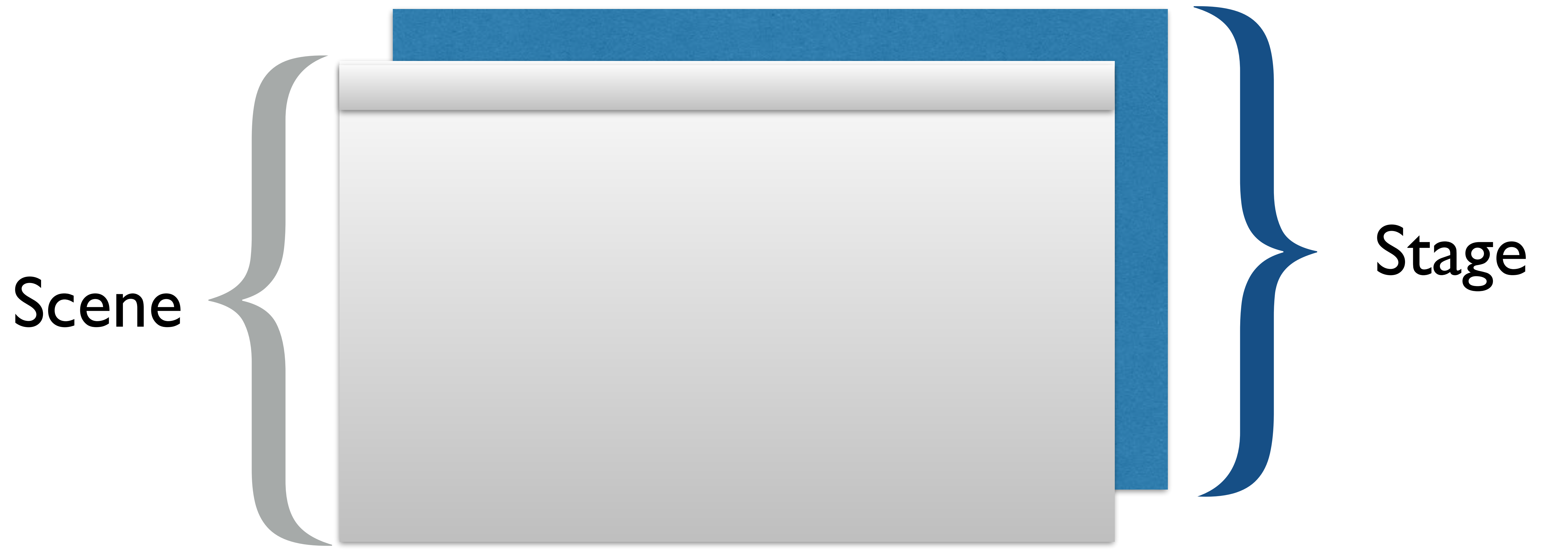
- However, views and controllers will be implemented differently in these two libraries.

View

Swing

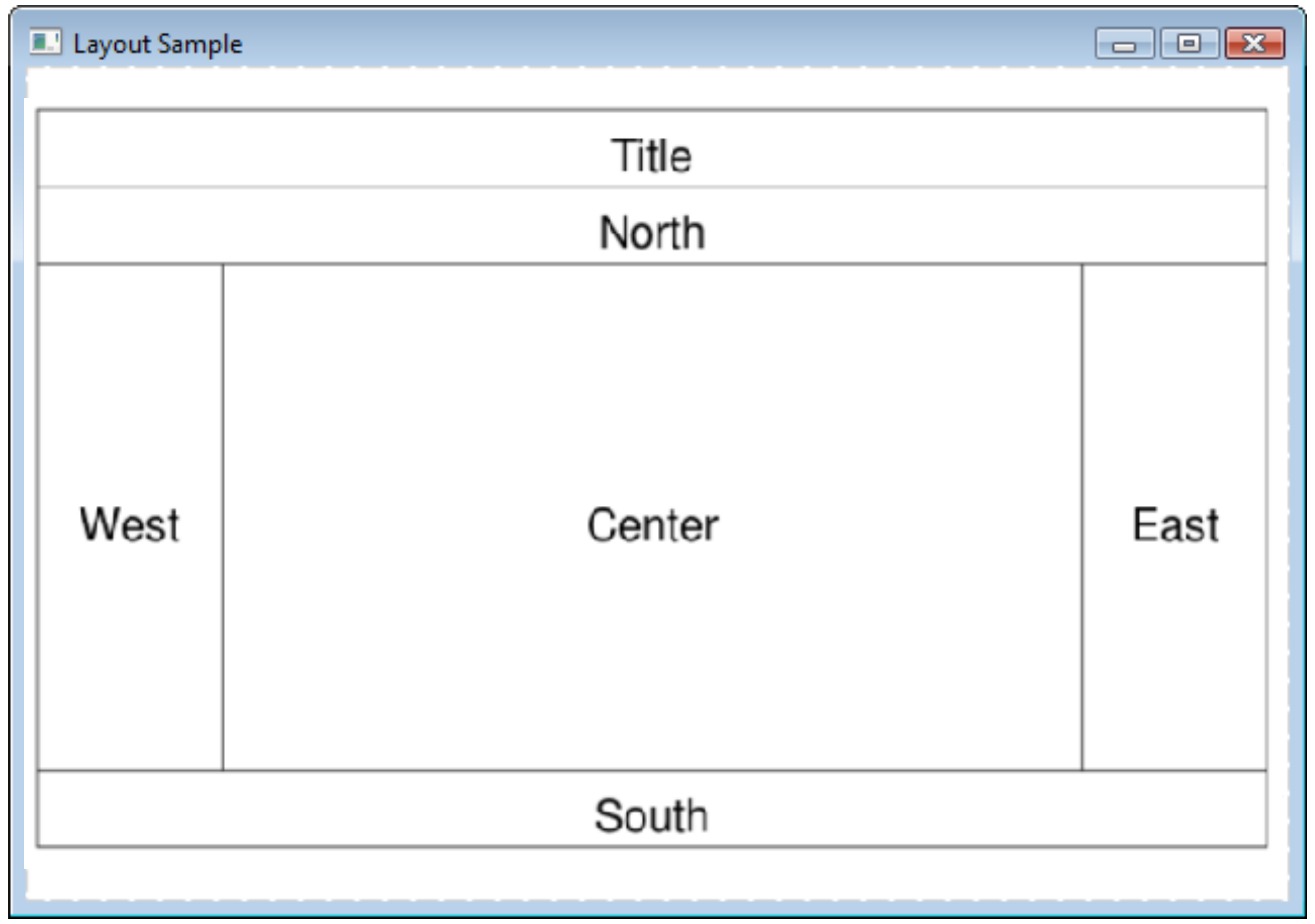


JavaFX



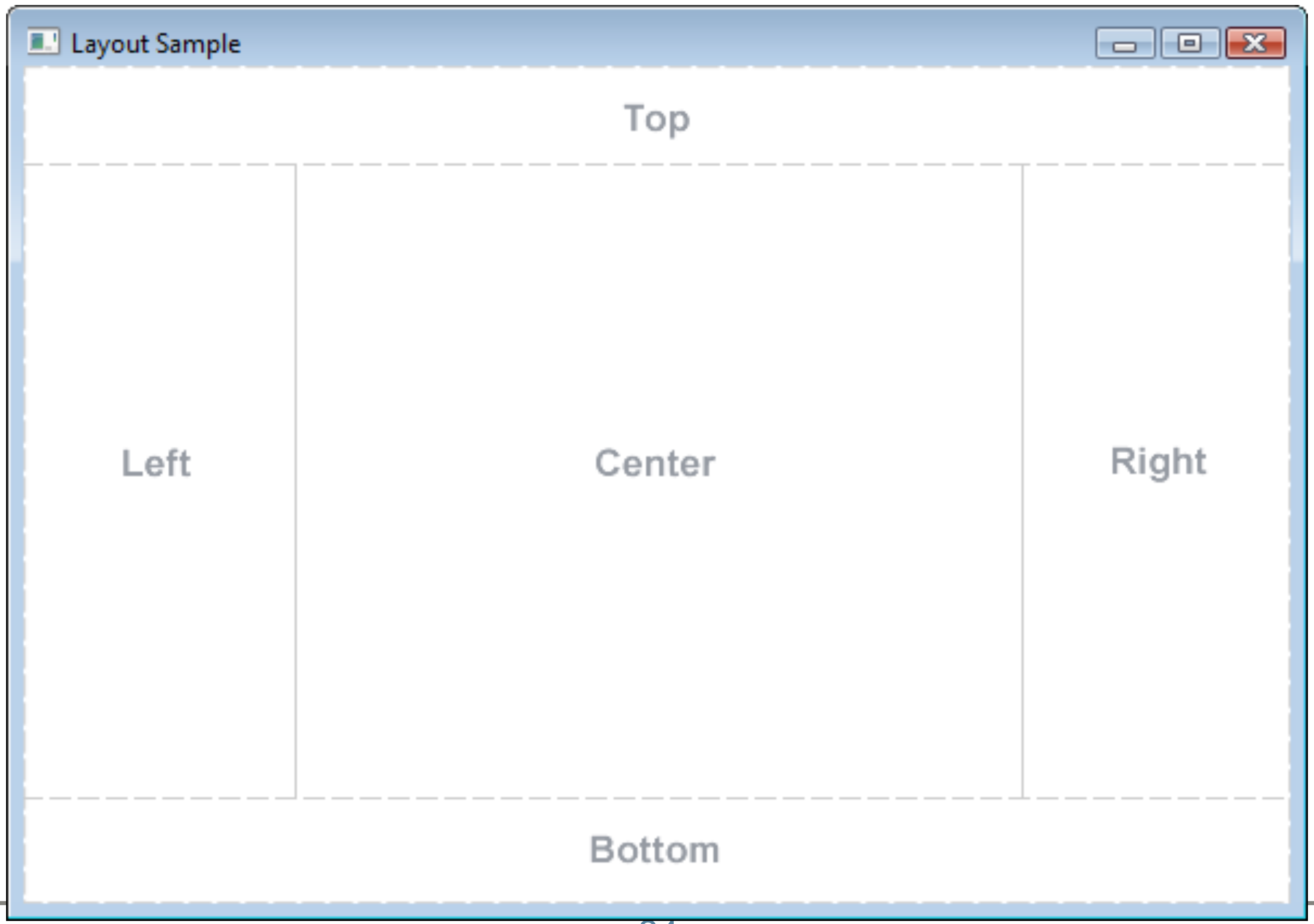
View

Swing



BorderLayout

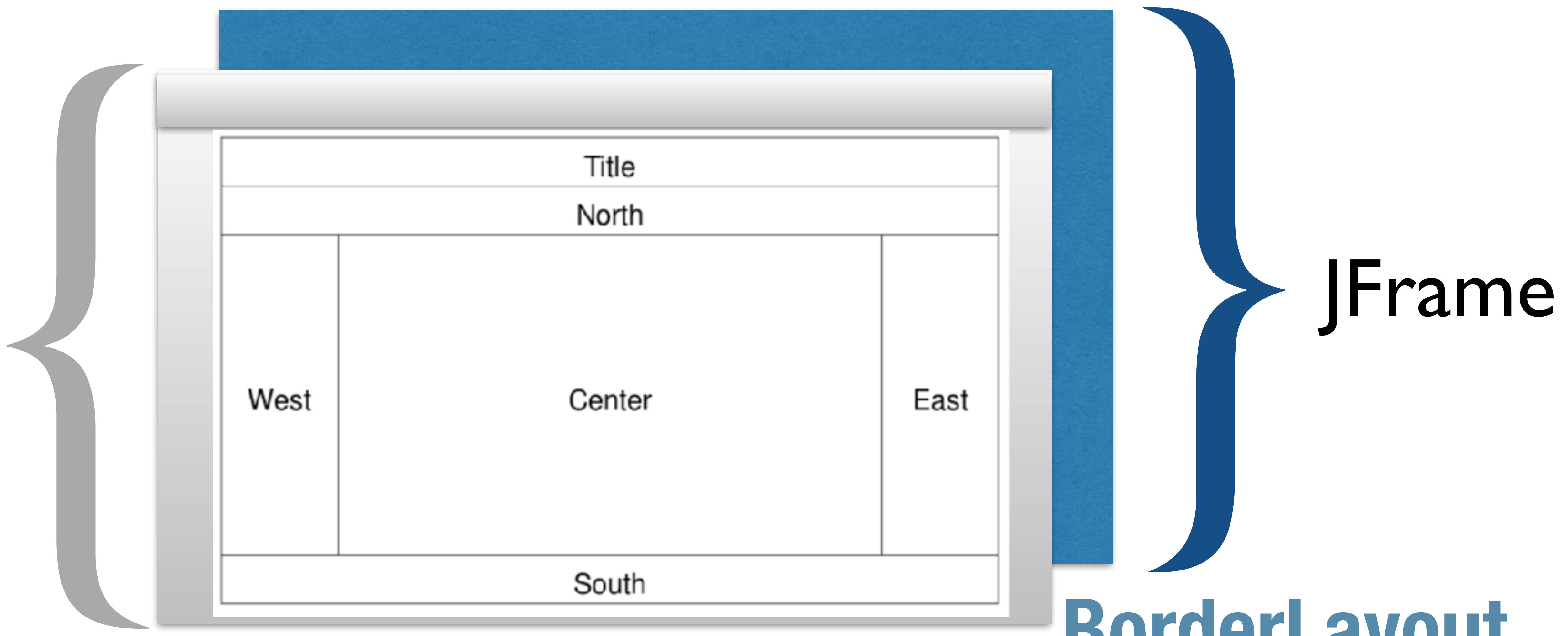
JavaFX



BorderPane

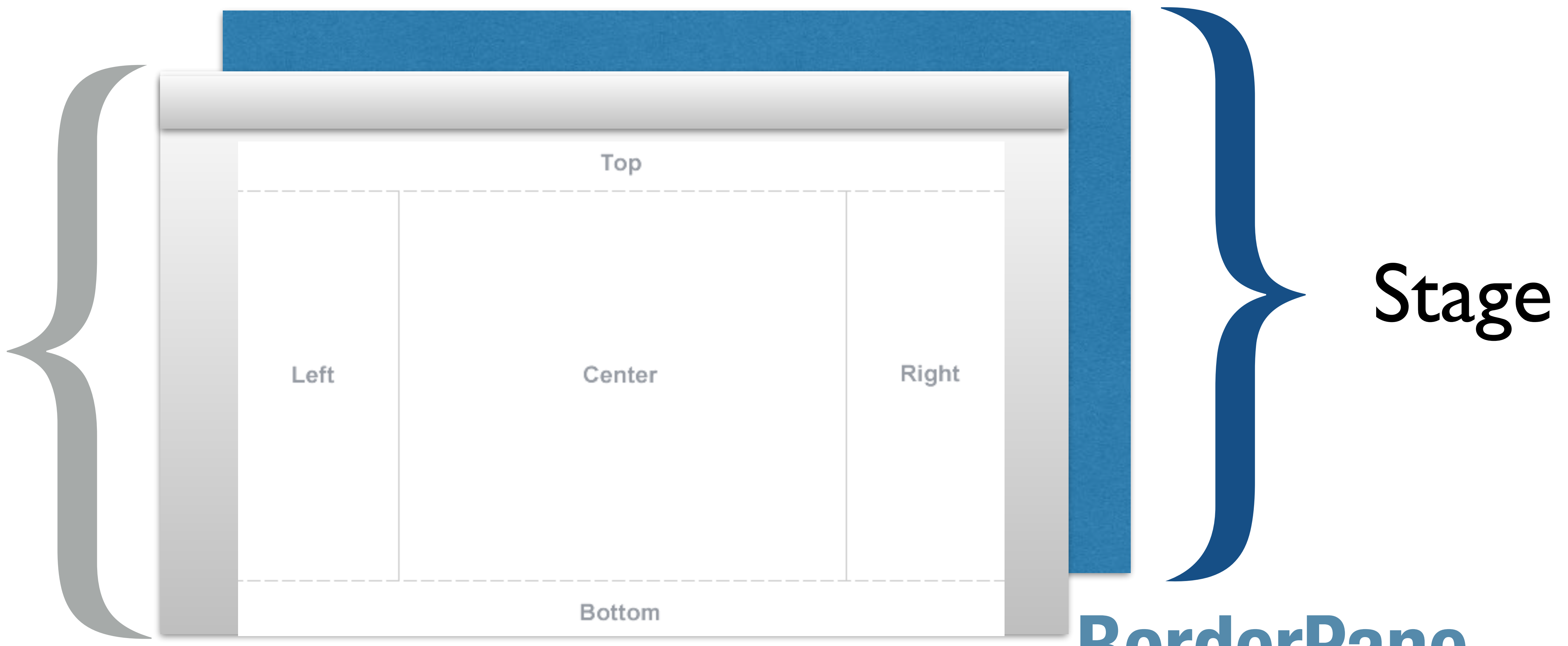
View

Swing



BorderLayout

JavaFX



BorderPane

GUI Components

Swing

JLabel

JTextField

☐ JCheckBox

JButton

JPasswordField

JavaFX

Label

TextField

☐ CheckBox

Button

PasswordField

Swing

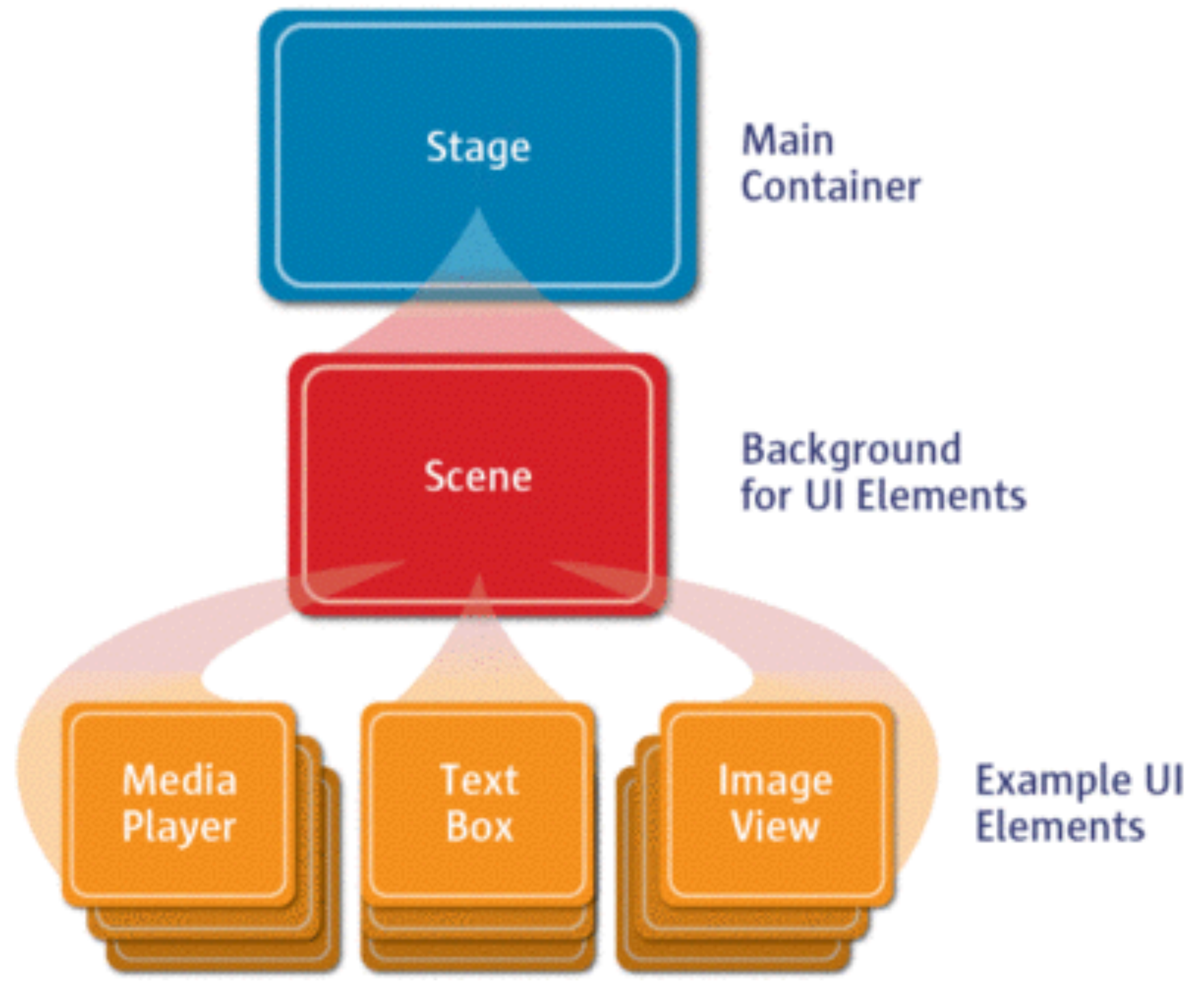
Listener Name	Listens To…
ActionListener	JButton, JTextField, JPasswordField
ItemListener	JCheckBox
ListSelectionListener	JList
MouseListener	(mouse clicks)
MouseMotionListener	(mouse motion)

JavaFX

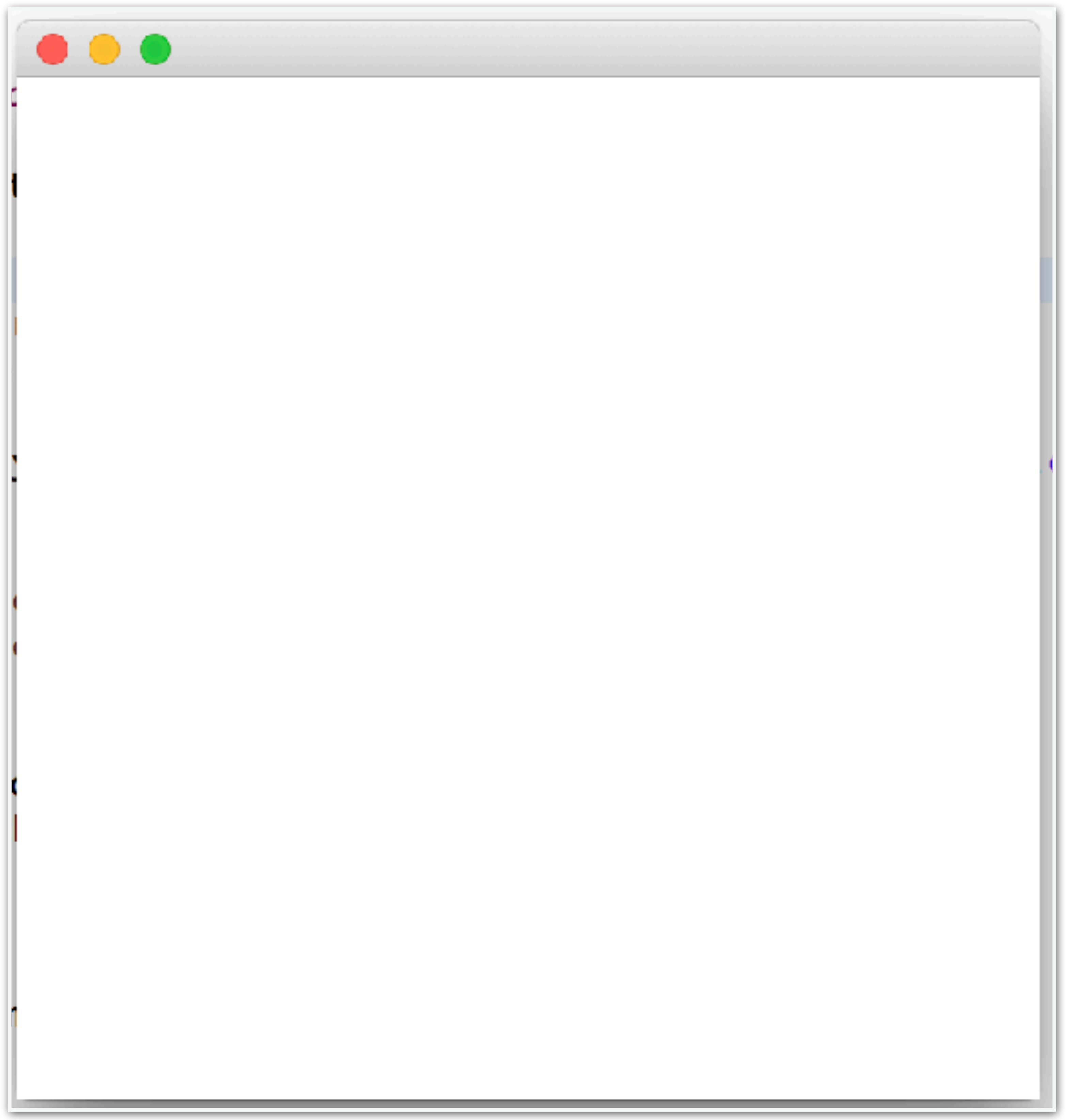
Listener Name	Listens To…
ActionEvent	Button, TextField, PasswordField
ActionEvent	CheckBox
ChangeListener	ListView
MouseEvent	(mouse clicks)
MouseEvent	(mouse motion)

JavaFX Projects

- Next step: follow MVC design pattern!
    - Set up packages the src folder:
        - application.model
        - application.controller



Default JavaFX code



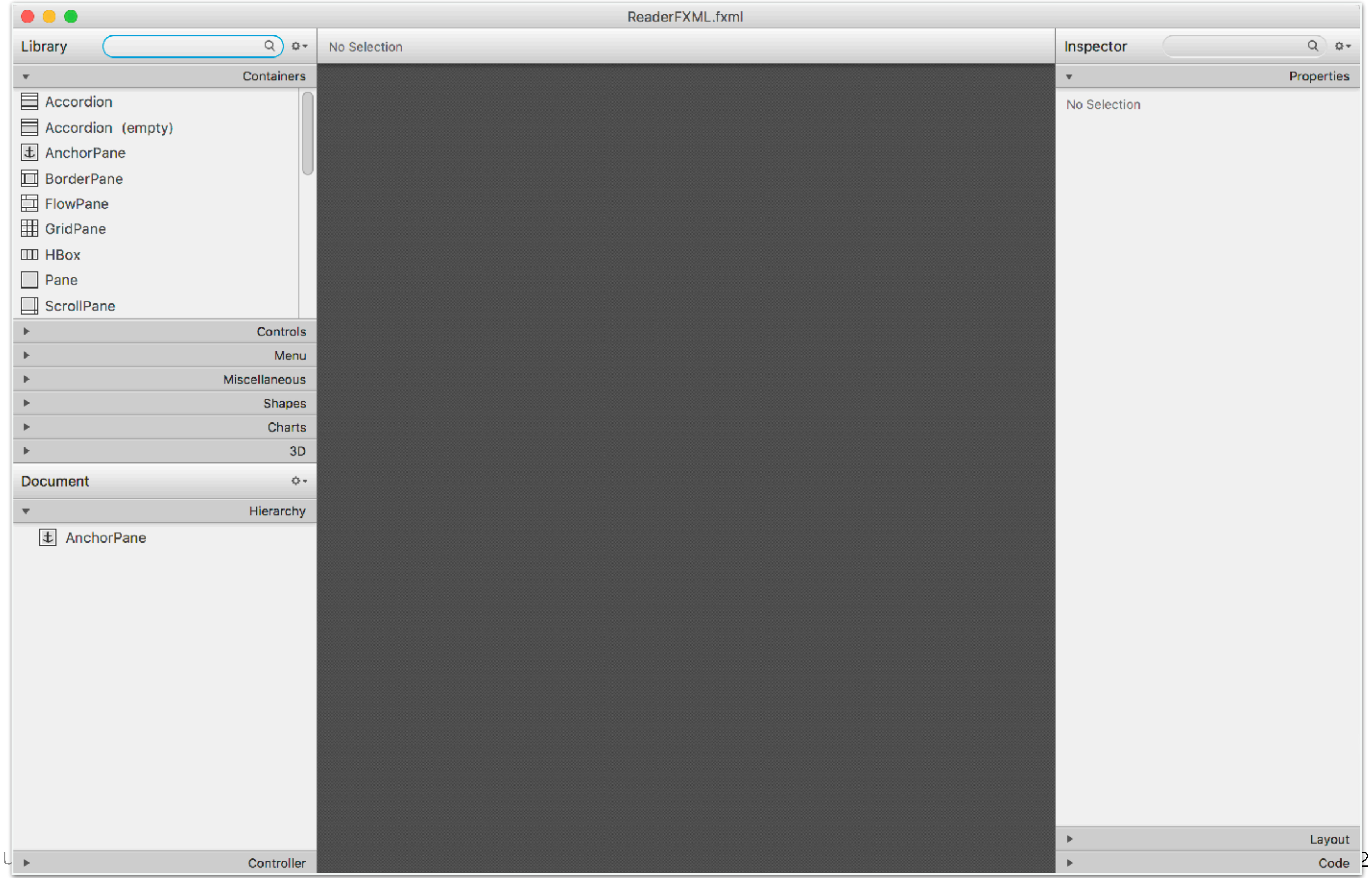
…let's add something to the application..

User Interface

- There are 2 ways to create a GUI for your application:
  - Code everything in Java
  - Create an FXML file:
    - New > Other > New FXML Document
    - Open that document with SceneBuilder

Only useful for small views (i.e.pop up error windows)

Scene Builder



FXML

- Let’s set up the view..
    - Select “**AnchorPane**” on the left.
    - On the right, under “**Layout**: AnchorPane”
        - Update the **Pref Width** & **Height** to the size of your application.
        - Add a **Split Pane** to the AnchorPane - fit to parent.

Displaying the FXML

- Connect the Main class to the FXML document by *replacing* the auto-generated code in start():

```
AnchorPane root = new AnchorPane();
    FXMLLoader loader = new FXMLLoader();
    loader.setLocation( Main.class.getResource("view/Main.fxml") );
    root = (AnchorPane) loader.load();

    Scene scene = new Scene( root );
    primaryStage.setScene( scene );
    primaryStage.show();
```