

CS 213 – Software Methodology

Spring 2016

Lecture 4 – Jan 28

Graphical User Interface

Fahrenheit-Celsius Converter

Version 2

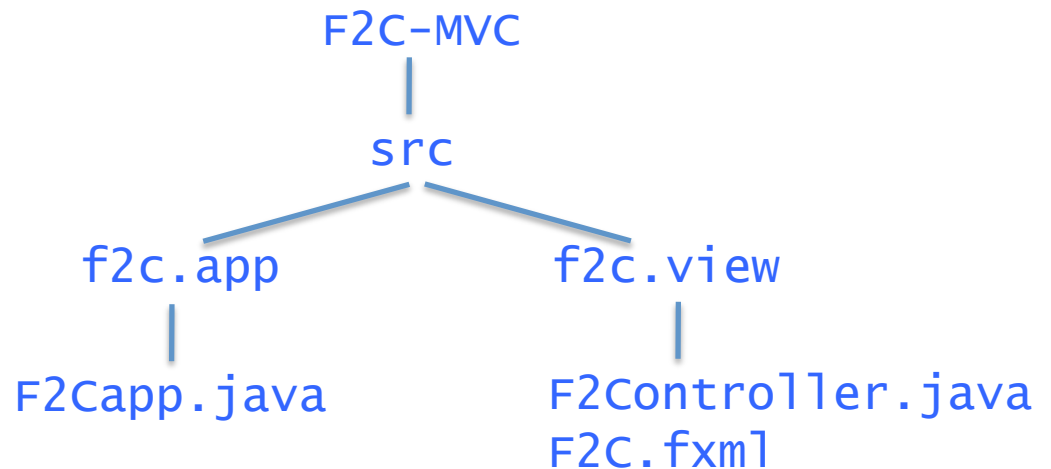
Separating View (UI in fxml) from
Controller (Java code)

The MVC Code Architecture (Model-View-Controller)

Model is the set of classes that stores and manages the data

View is the set of Java classes and non-Java design artifacts (e.g. xml, css, etc.) that implements the user interface

Controller is the set of classes that brokers between Model and View



(There is not always a separate Model, and each of M, V, and C need not always be in its own separate package)

View: Layout using fxml

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<?import javafx.scene.layout.*?>
<?import javafx.scene.control.*?>
<?import javafx.scene.text.*?>
<?import javafx.geometry.*?>
```

← Don't forget imports!! (Editor won't flag errors for unresolved tags.)

```
<GridPane
```

```
  xmlns="http://javafx.com/javafx/8.0.40"
  xmlns:fx="http://javafx.com/fxml/1"
  fx:controller="f2c.view.F2CController"
  vgap="10" hgap="10">
```

← Name space for Java FX tags (e.g. Text)
← Name space for FXML tags
(e.g. fx:controller)

Row and column indexes default to 0

```
<Text text="Fahrenheit" GridPane.valignment="BOTTOM"/>
<Button text="&gt;&gt;&gt;" GridPane.columnIndex="1" />
<Text text="Celsius" GridPane.columnIndex="2" GridPane.valignment="BOTTOM"/>
<TextField prefColumnCount="10" promptText="-40.0" GridPane.rowIndex="1" />
<Button text="&lt;&lt;&lt;" GridPane.rowIndex="1" GridPane.columnIndex="1" />
<TextField prefColumnCount="10" promptText="-40.0"
  GridPane.rowIndex="1" GridPane.columnIndex="2" />
<padding>
  <Insets top="10" right="10" bottom="10" left="10"/>
</padding>
</GridPane>
```

View: Set up SceneBuilder

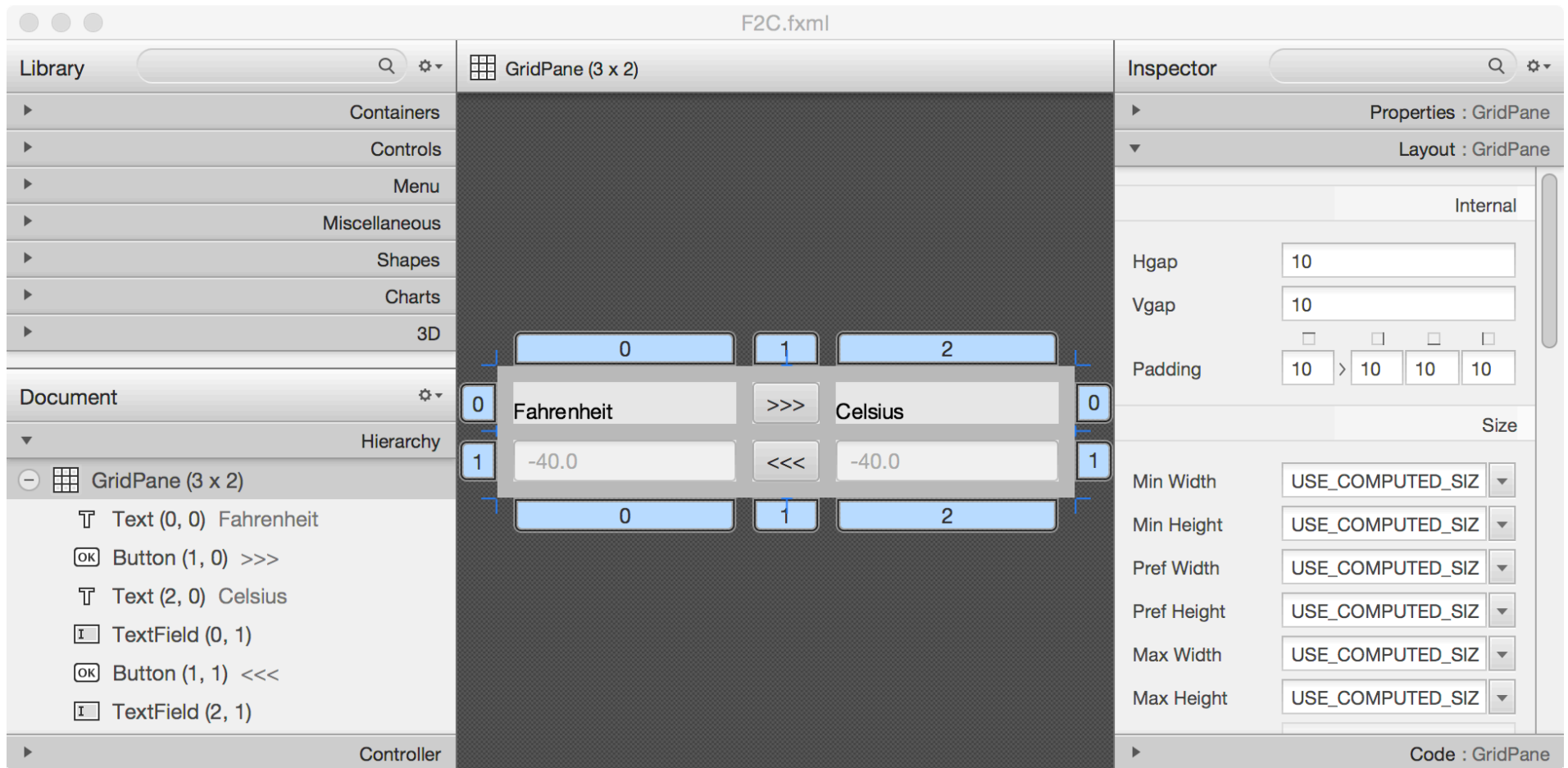
- Get SceneBuilder 8.0 at Gluon:

<http://gluonhq.com/open-source/scene-builder/>

(This will allow SceneBuilder to be opened from within Eclipse)

- You can construct UIs exclusively using SceneBuilder interface, or you can write up the UI in an editor and optionally verify/polish using SceneBuilder
- To open SceneBuilder from Eclipse on an fxml file:
Right click on fxml file -> open with -> other -> external programs (radio button) -> SceneBuilder

Verify fxml Layout with SceneBuilder



(In SceneBuilder, do [Preview -> Show Preview in Window](#) to simulate behavior)

FXML Layout – Id'ing widgets/event handler

...

```
<Text text="Fahrenheit" GridPane.valignment="BOTTOM"/>
```

```
<Button fx:id="f2c" text=">>>" GridPane.columnIndex="1"  
        onAction="#convert" />
```

```
<Text text="Celsius" GridPane.columnIndex="2" GridPane.valignment="BOTTOM"/>
```

```
<TextField fx:id="f" prefColumnCount="10" promptText="-40.0"  
            GridPane.rowIndex="1" />
```

```
<Button fx:id="c2f" text="<<<" GridPane.rowIndex="1"  
        GridPane.columnIndex="1" onAction="#convert" />
```

```
<TextField fx:id="c" prefColumnCount="10" promptText="-40.0"  
            GridPane.rowIndex="1" GridPane.columnIndex="2" />
```

```
<padding>
```

```
    <Insets top="10" right="10" bottom="10" left="10"/>
```

```
</padding>
```

Controller – Java Code

```
package f2c.view;
```

```
import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.scene.control.Button;
import javafx.scene.control.TextField;
```

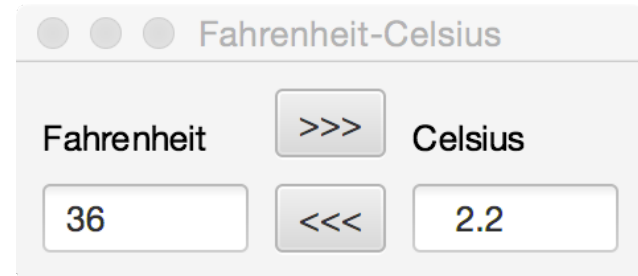
```
public class F2CController {
```

```
    @FXML Button f2c;
    @FXML Button c2f;
    @FXML TextField f;
    @FXML TextField c;
```

```
    public void convert(ActionEvent e) {
        Button b = (Button)e.getSource();
        if (b == f2c) {
            float fval = Float.valueOf(f.getText());
            float cval = (fval-32)*5/9;
            c.setText(String.format("%5.1f", cval));
        } else {
            float cval = Float.valueOf(c.getText());
            float fval = cval*9/5+32;
            f.setText(String.format("%5.1f", fval));
        }
    }
}
```

← @FXML directive links widget to fxml element:
var name in code = id in layout

← Name of method = name assigned
in # directive in fxml file for onAction
attribute



Main App for View/Controller

```
package f2c.app;
```

```
import javafx.application.Application;  
import javafx.fxml.FXMLLoader;
```

```
...
```

```
public class F2CApp extends Application {
```

```
    @Override
```

```
    public void start(Stage primaryStage) throws Exception {
```

```
        FXMLLoader loader = new FXMLLoader();  
        loader.setLocation(getClass().getResource("/f2c/view/F2C.fxml"));
```

```
        GridPane root = (GridPane)loader.load();
```

```
        Scene scene = new Scene(root);
```

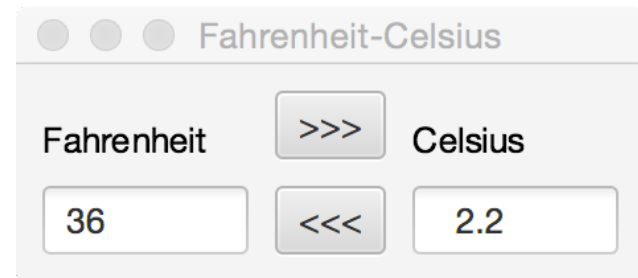
```
        ...
```

```
    }
```

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

```
        launch(args);
```

```
    }
```



Loading means creating
objects for various widgets
and layouts in the fxml file