Module Outline

- A first and simple example: scene, stage

Module Outline

- A first and simple example: scene, stage
- Layout

Module Outline

- A first and simple example: scene, stage
- Layout
- Designing a GUI using the JavaFX API

Module Outline

- A first and simple example: scene, stage
- Layout
- Designing a GUI using the JavaFX API
- Using an FXML file

Module Outline

- A first and simple example: scene, stage
- Layout
- Designing a GUI using the JavaFX API
- Using an FXML file
- Dependency injection in a JavaFX controller

Module Outline

- A first and simple example: scene, stage
- Layout
- Designing a GUI using the JavaFX API
- Using an FXML file
- Dependency injection in a JavaFX controller
- Catching events in callbacks

A Simple Example

- How to launch a JavaFX application?

A Simple Example

- **Create a class that extends Application, and overrides start()**
- **Call the launch() method on that class**

A Simple Example

- Create a class that extends Application, and overrides start()
- Call the launch() method on that class

```
import javafx.application.Application;

public class FirstApplication extends Application {

    public void start(Stage stage) {
        // callback
    }

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

A Simple Example

- Then add some content

```
public void start(Stage stage) {
   // a simple UI
   Label message = new Label("Hello world!");
   message.setFont(new Font(100));

   stage.setScene(new Scene(message));
   stage.setTitle("Hello");
   stage.show();
}
```

A Few Key Concepts

- **Stage: « top-level window »**

A Few Key Concepts

- Stage: « top-level window »
- A Stage can be a top-level window

A Few Key Concepts

- Stage: « top-level window »
- A Stage can be a top-level window
- A Stage can be a rectangular area in the case of an applet

A Few Key Concepts

- Stage: « top-level window »
- A Stage can be a top-level window
- A Stage can be a rectangular area in the case of an applet
- A Stage can be the full screen itself

A Few Key Concepts

- **Stage: « top-level window »**
- **Scene: a stage must hold a scene, a scene must reside in a stage**

A Few Key Concepts

- Stage: « top-level window »
- Scene: a stage must hold a scene, a scene must reside in a stage
- A Scene holds all the graphical components, shapes, etc…

A Few Key Concepts

- On our example:
- The Label is added to a Scene
- The Scene is added to the Stage
- And we call the show() method on the stage

A Layout Example

- Layout: can hold several components

```
public void start(Stage stage) {
   // javacontrol.Label
   Label message1 = new Label("Hello world!");
   message1.setFont(new Font(100));

   Label message2 = new Label("Hello world!");
   message2.setFont(new Font(100));
   // java.scene.layout.VBox
   VBox vbox = new VBox(message1, message2);

   stage.setScene(new Scene(vbox));
   stage.setTitle("Hello");
   stage.show();
}
```

A Login Window Example – XML Version

- Can also be designed in a XML file

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

<?import javafx.geometry.*?>
<?import javafx.scene.control.*?>
<?import javafx.scene.layout.*?>
<?import javafx.scene.paint.*?>

<GridPane hgap="10" vgap="10">

   <!-- content of the grid pane -->

</GridPane>
```

A Login Window Example – XML Version

- Can also be designed in a XML file

```
<GridPane hgap="10" vgap="10">

   <padding>
      <Insets bottom="10.0" top="10.0"
              left="10.0" right="10.0" />
   </padding>

   <children>
      <!-- children components -->
   </children>
</GridPane>
```

A Login Window Example – XML Version

- Can also be designed in a XML file

```xml
<children>
   <Label text="User name:" />
   <TextField />
</children>
```

A Login Window Example – XML Version

- Can also be designed in a XML file

```
<children>
   <Label text="User name:" />
   <TextField id="username"/>
</children>
```

A Login Window Example – XML Version

- Can also be designed in a XML file

```xml
<children>
   <Label text="User name:"
          GridPane.columnIndex="0" GridPane.rowIndex="0" />
   <TextField id="username"
          GridPane.columnIndex="1" GridPane.rowIndex="0" />
</children>
```

A Login Window Example – XML Version

- Can also be designed in a XML file

```xml
<children>
   <Label text="User name:"
          GridPane.columnIndex="0" GridPane.rowIndex="0"
          GridPane.halignment="RIGHT" />
   <TextField id="username"
          GridPane.columnIndex="1" GridPane.rowIndex="0" />
</children>
```

A Login Window Example – XML Version

- Can also be designed in a XML file

```
<children>
   <Label text="User name:"
          GridPane.columnIndex="0" GridPane.rowIndex="0"
          GridPane.halignment="RIGHT" />
   <TextField id="username"
          GridPane.columnIndex="1" GridPane.rowIndex="0" />
   <Label text="Password:"
          GridPane.columnIndex="0" GridPane.rowIndex="0"
          GridPane.halignment="RIGHT" />
   <PasswordField id="password"
          GridPane.columnIndex="1" GridPane.rowIndex="0" />
</children>
```

A Login Window Example – XML Version

- Can also be designed in a XML file

```
<children>
   <!-- labels + textfields -->
   <HBox >
      <children>
         <Button text="Ok" />
         <Button text="Cancel" />
      </children>
   </HBox>
</children>
```

A Login Window Example – XML Version

- Defining the ID attributes

```
<GridPane hgap="10" vgap="10"
          xmlns:fx="http://javafx.com/fxml"
          fx:controller="org.paumard.javafx.MyController">

   <children>
      <Label text="User name:" />
      <TextField fx:id="username"/>
   </children>

   <HBox >
      <children>
         <Button text="Ok" onAction="#okAction"/>
      </children>
   </HBox>
   <!-- rest of the UI -->

</GridPane>
```

A Login Window Example – XML Version

- The controller class

```
public class MyController implements Initializable {

    @FXML
    private TextField username;

    @Override
    public void initialize(URL url, ResourceBundle bundle) {
    }

    public void okAction(ActionEvent event) {

        System.out.println("Clicked ok");
        System.out.println("user name = " +
                                   username.getText());
    }
}
```

A Login Window Example – XML Version

- The Application class

```
public class MyApplication extends Application {
   @Override
   public void start(Stage stage) {
      try {
         FXMLLoader loader = new
            FXMLLoader(getClass().getResource("ihm.fxml"));

         Parent root = loader.load();
         stage.setScene(new Scene(root));
         stage.show();
      } catch (IOException ioe) {
         // ...
      }
   }

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

And There Is More

- Supports CSS for customizing the look and feel of the GUI
- A rich animation API for moving, scaling, rotating etc… components
- Support for touch interfaces
- Works on many types of displays
- Compatible with Swing (to a certain extent)

Summary

- Quick overview of Java FX 8
- How to create basic interfaces
- Building an interface with the API or FXML
- Dependency injection on GUI components
- Callbacks on simple events