JavaFX

Prof. Michele Loreti

Programmazione Avanzata

Corso di Laurea in Informatica (L31)

Scuola di Scienze e Tecnologie

JavaFX

https://openjfx.io

JavaFX is a GUI toolkit for Java (GUI is short for Graphical User Interface).

JavaFX

https://openjfx.io

JavaFX is a GUI toolkit for Java (GUI is short for Graphical User Interface).

JavaFX makes it easier to create desktop applications and games in Java.

JavaFX

https://openjfx.io

JavaFX is a GUI toolkit for Java (GUI is short for Graphical User Interface).

JavaFX makes it easier to create desktop applications and games in Java.

JavaFX comes with a large set of built-in GUI components, like buttons, text fields, tables, trees, menus, charts and much more.

JavaFX

https://openjfx.io

JavaFX is a GUI toolkit for Java (GUI is short for Graphical User Interface).

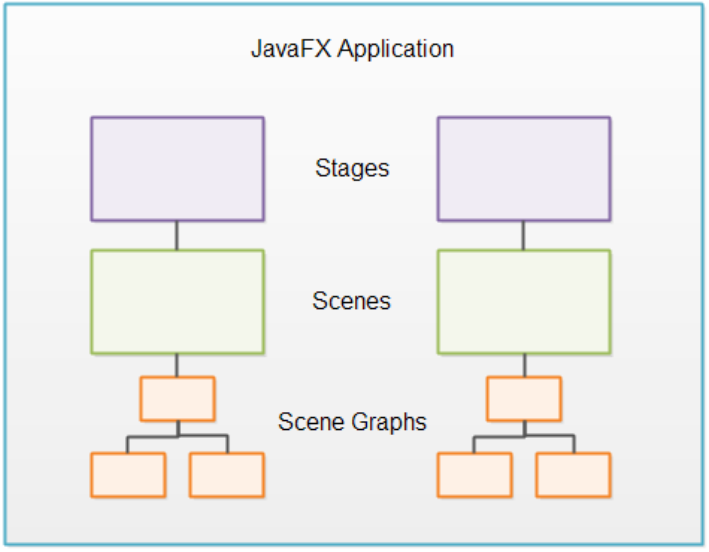
JavaFX makes it easier to create desktop applications and games in Java.

JavaFX comes with a large set of built-in GUI components, like buttons, text fields, tables, trees, menus, charts and much more.

JavaFX components can be styled using CSS, and we can use FXML to compose a GUI instead of doing it in Java code.

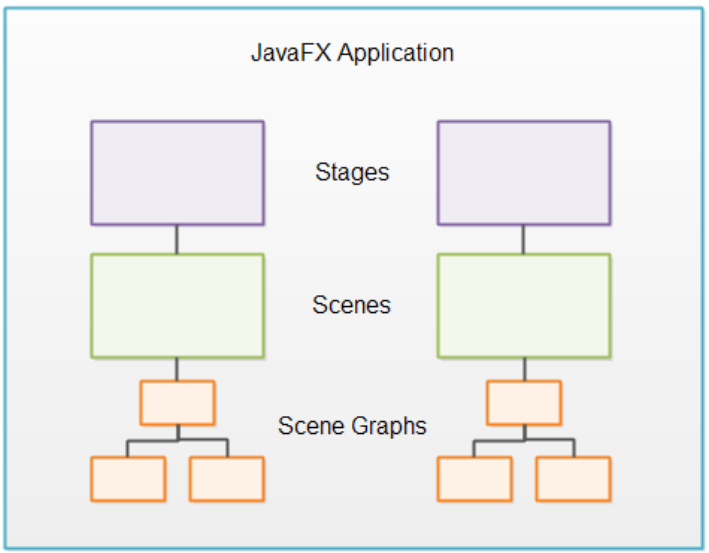
JavaFX Overview

- A JavaFX application contains one or more stages which corresponds to windows.



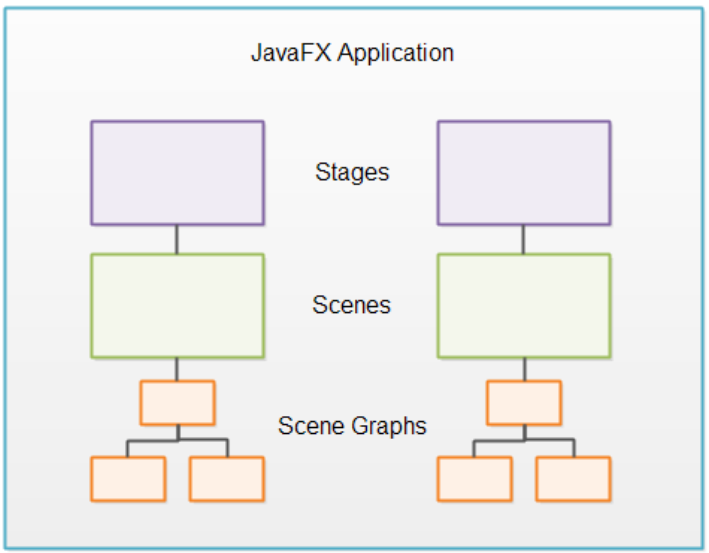
JavaFX Overview

- A JavaFX application contains one or more stages which corresponds to windows.
- Each stage has a scene attached to it.



JavaFX Overview

- A JavaFX application contains one or more stages which corresponds to windows.
- Each stage has a scene attached to it.
- Each scene can have an object graph of controls, layouts etc. attached to it, called the scene graph.



JavaFX: Stage

The stage is the outer frame for a JavaFX application. The stage typically corresponds to a window.

JavaFX: Stage

The stage is the outer frame for a JavaFX application. The stage typically corresponds to a window.

When used in a desktop environment, a JavaFX application can have multiple windows open. Each window has its own stage.

JavaFX: Stage

The stage is the outer frame for a JavaFX application. The stage typically corresponds to a window.

When used in a desktop environment, a JavaFX application can have multiple windows open. Each window has its own stage.

Each stage is represented by a Stage object inside a JavaFX application. A JavaFX application has a primary Stage object which is created for you by the JavaFX runtime.

JavaFX: Stage

The stage is the outer frame for a JavaFX application. The stage typically corresponds to a window.

When used in a desktop environment, a JavaFX application can have multiple windows open. Each window has its own stage.

Each stage is represented by a Stage object inside a JavaFX application. A JavaFX application has a primary Stage object which is created for you by the JavaFX runtime.

A JavaFX application can create additional Stage objects if it needs additional windows open. For instance, for dialogs, wizards etc.

JavaFX: Scene

To display anything on a stage in a JavaFX application, you need a scene.

JavaFX: Scene

To display anything on a stage in a JavaFX application, you need a scene.

A stage can only show one scene at a time, but it is possible to exchange the scene at runtime.

JavaFX: Scene

To display anything on a stage in a JavaFX application, you need a scene.

A stage can only show one scene at a time, but it is possible to exchange the scene at runtime.

It is like in a theatre where stage can be rearranged to show multiple scenes during a play. Similarly, stage object in JavaFX can show multiple scenes (one at a time) during the life time of a JavaFX application.

JavaFX: Scene

To display anything on a stage in a JavaFX application, you need a scene.

A stage can only show one scene at a time, but it is possible to exchange the scene at runtime.

It is like in a theatre where stage can be rearranged to show multiple scenes during a play. Similarly, stage object in JavaFX can show multiple scenes (one at a time) during the life time of a JavaFX application.

A scene is represented by a Scene object inside a JavaFX application. A JavaFX application must create all Scene objects it needs.

JavaFX: Scene Graph and Nodes

All visual components (controls, layouts etc.) must be attached to a scene to be displayed, and that scene must be attached to a stage for the whole scene to be visible.

JavaFX: Scene Graph and Nodes

All visual components (controls, layouts etc.) must be attached to a scene to be displayed, and that scene must be attached to a stage for the whole scene to be visible.

The total object graph of all the controls, layouts etc. attached to a scene is called the scene graph.

JavaFX: Scene Graph and Nodes

All visual components (controls, layouts etc.) must be attached to a scene to be displayed, and that scene must be attached to a stage for the whole scene to be visible.

The total object graph of all the controls, layouts etc. attached to a scene is called the scene graph.

All components attached to the scene graph are called nodes. All nodes are subclasses of a JavaFX class called `javafx.scene.Node`.

JavaFX: Scene Graph and Nodes

All visual components (controls, layouts etc.) must be attached to a scene to be displayed, and that scene must be attached to a stage for the whole scene to be visible.

The total object graph of all the controls, layouts etc. attached to a scene is called the scene graph.

All components attached to the scene graph are called nodes. All nodes are subclasses of a JavaFX class called `javafx.scene.Node`.

There are two types of nodes:

- Branch nodes: are nodes that can contain other nodes (child nodes). They are also referred to as parent nodes because they can contain child nodes.
- Leaf node: is a node which cannot contain other nodes.

JavaFX: Controls

JavaFX controls are JavaFX components which provide some kind of control functionality inside a JavaFX application. For instance, a button, radio button, table, tree etc.

JavaFX: Controls

JavaFX controls are JavaFX components which provide some kind of control functionality inside a JavaFX application. For instance, a button, radio button, table, tree etc.

For a control to be visible it must be attached to the scene graph of some Scene object.

JavaFX: Controls

JavaFX controls are JavaFX components which provide some kind of control functionality inside a JavaFX application. For instance, a button, radio button, table, tree etc.

For a control to be visible it must be attached to the scene graph of some Scene object.

Controls are usually nested inside some JavaFX layout component that manages the layout of controls relative to each other.

JavaFX: Layouts

JavaFX layouts are components which contains other components inside them. The layout component manages the layout of the components nested inside it.

JavaFX: Layouts

JavaFX layouts are components which contains other components inside them. The layout component manages the layout of the components nested inside it.

JavaFX layout components are also sometimes called parent components because they contain child components, and because layout components are subclasses of the JavaFX class `javafx.scene.Parent`.

JavaFX: Layouts

JavaFX layouts are components which contains other components inside them. The layout component manages the layout of the components nested inside it.

JavaFX layout components are also sometimes called parent components because they contain child components, and because layout components are subclasses of the JavaFX class `javafx.scene.Parent`.

A layout component must be attached to the scene graph of some `Scene` object to be visible.

JavaFX: Layouts

JavaFX layouts are components which contains other components inside them. The layout component manages the layout of the components nested inside it.

JavaFX layout components are also sometimes called parent components because they contain child components, and because layout components are subclasses of the JavaFX class `javafx.scene.Parent`.

A layout component must be attached to the scene graph of some `Scene` object to be visible.

It is possible to nest layout components inside other layout components.

JavaFX: Setting up Gradle

```
plugins {
    ...
    id 'org.openjfx.javafxplugin' version '0.0.7'
}

javafx {
    version = "12.0.1"
    modules = [ 'javafx.controls' ]
}
```

JavaFX: Application

A JavaFX application needs a primary launch class. This class has to extend the `javafx.application.Application`.

JavaFX: Application

A JavaFX application needs a primary launch class. This class has to extend the `javafx.application.Application`.

All subclasses of the `Application` class must implement the abstract `start()` method of the `Application` class.

JavaFX: Application

A JavaFX application needs a primary launch class. This class has to extend the `javafx.application.Application`.

All subclasses of the `Application` class must implement the abstract `start()` method of the `Application` class.

The `start()` method is called when the JavaFX application is started:

```
@Override
public void start(Stage primaryStage) throws Exception {
    primaryStage.setTitle("My First JavaFX App");

    primaryStage.show();
}
```

JavaFX: Application

JavaFX provides an execution environment where our classes run.

JavaFX: Application

JavaFX provides an execution environment where our classes run.

In the `main` we have only to call the `Application.launch(args)` method to pass the program argument to the environment.

JavaFX: Application

JavaFX provides an execution environment where our classes run.

In the main we have only to call the Application.launch(args) method to pass the program argument to the environment.

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

JavaFX: Application

JavaFX provides an execution environment where our classes run.

In the main we have only to call the Application.launch(args) method to pass the program argument to the environment.

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

The launch() method will detect from which class it is called.

JavaFX: Adding a Scene

To display elements we have to attach them a Scene.

JavaFX: Adding a Scene

To display elements we have to attach them a Scene.

```
Label label = new Label("Hello World, JavaFX !");
Scene scene = new Scene(label, 400, 200);
primaryStage.setScene(scene);
```

JavaFX: Adding a Scene

To display elements we have to attach them a Scene.

```
Label label = new Label("Hello World, JavaFX !");
Scene scene = new Scene(label, 400, 200);
primaryStage.setScene(scene);
```

The first parameter of the Scene constructor is the root element of the scene graph.

JavaFX: Adding a Scene

To display elements we have to attach them a Scene.

```
Label label = new Label("Hello World, JavaFX !");
Scene scene = new Scene(label, 400, 200);
primaryStage.setScene(scene);
```

The first parameter of the Scene constructor is the root element of the scene graph.

The scene graph is a graph like object structure containing all the visual components to be displayed in the JavaFX application.

JavaFX: Stage

A JavaFX Stage, `javafx.stage.Stage`, represents a window in a JavaFX desktop application.

JavaFX: Stage

A JavaFX Stage, javafx.stage.Stage, represents a window in a JavaFX desktop application.

Inside a JavaFX Stage you can insert a JavaFX Scene which represents the content displayed inside a window - inside a Stage.

JavaFX: Stage

A JavaFX Stage, javafx.stage.Stage, represents a window in a JavaFX desktop application.

Inside a JavaFX Stage you can insert a JavaFX Scene which represents the content displayed inside a window - inside a Stage.

Creating a Stage:

```
Stage stage = new Stage();
```

JavaFX: Stage

A JavaFX Stage, javafx.stage.Stage, represents a window in a JavaFX desktop application.

Inside a JavaFX Stage you can insert a JavaFX Scene which represents the content displayed inside a window - inside a Stage.

Creating a Stage:

```
Stage stage = new Stage();
```

Showing a Stage:

- show()
- showAndWait()

JavaFX: Stage

Set a Scene: To display elements in a stage, you have to equip it with a scene...

```
VBox vBox = new VBox(new Label("A JavaFX Label"));
Scene scene = new Scene(vBox);

Stage stage = new Stage();
stage.setScene(scene);
```

JavaFX: Stage

Set a Scene: To display elements in a stage, you have to equip it with a scene...

```
VBox vBox = new VBox(new Label("A JavaFX Label"));
Scene scene = new Scene(vBox);

Stage stage = new Stage();
stage.setScene(scene);
```

Set title:

```
stage.setTitle("JavaFX Stage Window Title");
```

JavaFX: Stage

Set a Scene: To display elements in a stage, you have to equip it with a scene...

```
VBox vBox = new VBox(new Label("A JavaFX Label"));
Scene scene = new Scene(vBox);

Stage stage = new Stage();
stage.setScene(scene);
```

Set title:

```
stage.setTitle("JavaFX Stage Window Title");
```

Set position:

```
stage.setX(50);
stage.setY(50)
```

JavaFX: Stage

Set size:

```
stage.setWidth(600);
stage.setHeight(300);
```

JavaFX: Stage

Set size:

```
stage.setWidth(600);
stage.setHeight(300);
```

Modality: The Stage modality determines if the window representing the Stage will block other windows opened by the same JavaFX application.

- stage.initModality(Modality.APPLICATION_MODAL)
- stage.initModality(Modality.WINDOW_MODAL)
- stage.initModality(Modality.NONE)

JavaFX: Stage

Set size:

```
stage.setWidth(600);
stage.setHeight(300);
```

Modality: The Stage modality determines if the window representing the Stage will block other windows opened by the same JavaFX application.

- stage.initModality(Modality.APPLICATION_MODAL)
- stage.initModality(Modality.WINDOW_MODAL)
- stage.initModality(Modality.NONE)

Owner: A JavaFX Stage can be owned by another Stage.

```
stage.initOwner(primaryStage);
```

JavaFX: Stage

Stage Style: You can set the style of a JavaFX Stage via its initStyle() method. There are a set of different styles you can choose from:

- DECORATED
- UNDECORATED
- TRANSPARENT
- UNIFIED
- UTILITY

You can set the style by using the `initStyle` method.

JavaFX: Scene

You create a JavaFX Scene object via its constructor.

JavaFX: Scene

You create a JavaFX Scene object via its constructor.

As parameter you must pass the root JavaFX GUI component that is to act as the root view to be displayed inside the Scene:

```
VBox  vBox  = new VBox();
Scene scene = new Scene(vBox);
```

JavaFX: Scene

You create a JavaFX Scene object via its constructor.

As parameter you must pass the root JavaFX GUI component that is to act as the root view to be displayed inside the Scene:

```
VBox  vBox  = new VBox();
Scene scene = new Scene(vBox);
```

In order to make a JavaFX Scene visible, it must be set on a JavaFX Stage:

```
VBox vBox = new VBox(new Label("A JavaFX Label"));
Scene scene = new Scene(vBox);

Stage stage = new Stage();
stage.setScene(scene);
```

JavaFX: Scene

You create a JavaFX Scene object via its constructor.

As parameter you must pass the root JavaFX GUI component that is to act as the root view to be displayed inside the Scene:

```
VBox  vBox  = new VBox();
Scene scene = new Scene(vBox);
```

In order to make a JavaFX Scene visible, it must be set on a JavaFX Stage:

```
VBox vBox = new VBox(new Label("A JavaFX Label"));
Scene scene = new Scene(vBox);

Stage stage = new Stage();
stage.setScene(scene);
```

A JavaFX Scene can be attached to only a single Stage at a time, and Stage can also only display one Scene at a time.

JavaFX: FXML

JavaFX FXML is an XML format that enables you to compose JavaFX GUIs in a fashion similar to how you compose web GUIs in HTML.

JavaFX: FXML

JavaFX FXML is an XML format that enables you to compose JavaFX GUIs in a fashion similar to how you compose web GUIs in HTML.

FXML thus enables you to separate your JavaFX layout code from the rest of your application code. This cleans up both the layout code and the rest of the application code.

JavaFX: FXML

JavaFX FXML is an XML format that enables you to compose JavaFX GUIs in a fashion similar to how you compose web GUIs in HTML.

FXML thus enables you to separate your JavaFX layout code from the rest of your application code. This cleans up both the layout code and the rest of the application code.

FXML can be used both to compose the layout of a whole application GUI, or just part of an application GUI, e.g. the layout of one part of a form, tab, dialog etc.

JavaFX: FXML

JavaFX FXML is an XML format that enables you to compose JavaFX GUIs in a fashion similar to how you compose web GUIs in HTML.

FXML thus enables you to separate your JavaFX layout code from the rest of your application code. This cleans up both the layout code and the rest of the application code.

FXML can be used both to compose the layout of a whole application GUI, or just part of an application GUI, e.g. the layout of one part of a form, tab, dialog etc.

FXML file can be generated by using tools supporting design of GUI (see SceneBuilder).

JavaFX: FXML
Example

```
<?xml version="1.0" encoding="UTF-8"?>
<?import javafx.scene.layout.VBox?>
<?import javafx.scene.control.Label?>

<VBox>
    <children>
        <Label text="Hello world FXML"/>
    </children>
</VBox>
```

JavaFX: FXML

```
FXMLLoader loader = new FXMLLoader();
loader.setLocation(anUrl);
VBox vbox = loader.<VBox>load();

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

To be continued...