P5. CSS IN JAVAFX

Interfaces Persona Computador

Depto. Sistemas Informáticos y Computación

UPV

Summary

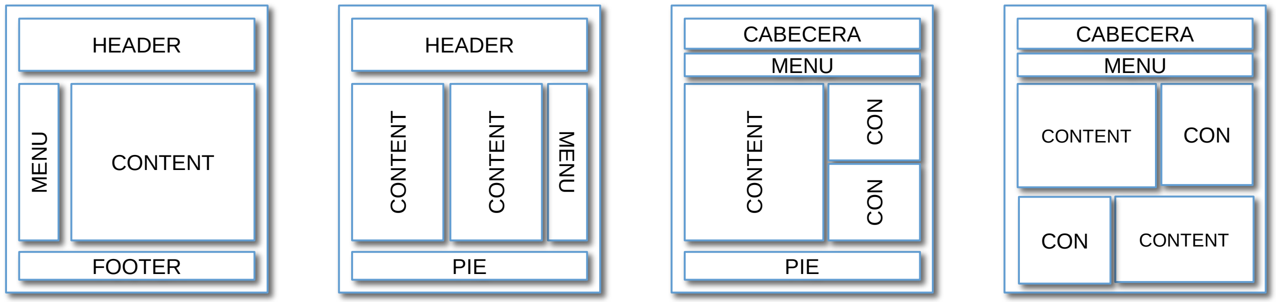
- CSS, What is it and what is it for?
- CSS in JavaFX
- Using a CSS external file
- Changing the theme
- CSS
- CSS in Scene Builder

CSS, What is it and what is it for?

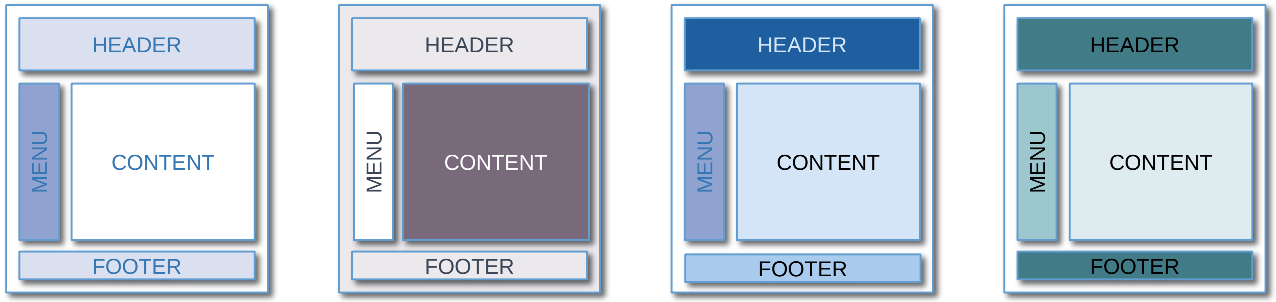
- CSS: Cascading Style Sheets
  - Mainly used in web design:
    - Format: color, size, font styles ...
    - Layout of the elements ...
  - Mechanism to separate the aspect from the content of a website
  - Many web pages can share the styles defined in them
- CSS in JAVAFX
  - It is possible to use a almost the same style language
  - IMPORTANT: the properties in JAVAFX are not the same than in the CSS employed in HTML

CSS, What is it and what is it for?

- The Style Sheets allow defining:
  - How the content of the page is organized



  - What that content should look like



JavaFX and CSS

- The default source for JavaFX 8 styles is located in a file called **modena.css**
  - The file is located in the Java folder in: **/jdk1.8.x/jre/lib/ext/jfxrt.jar**
  - The file **modena.css** can be accessed in the folder **com/sun/javafx/scene/control/skin/modena/** inside the JAR file
- Any JavaFX 8 application uses the style sheet in that file
- The styles in modena.css can be overridden with our own styles

- *Exercise: open the **jfxrt.jar** file as if it were a zip file. Extract and review the contents of **modena.css** to see the styles it contains*

Using an External CSS File

- Two options:
  - Define the style for all the nodes of the application (usually done to change the theme of the application):

```
Application.setUserAgentStylesheet(
              getClass().getResource("theme.css").toExternalForm());
```

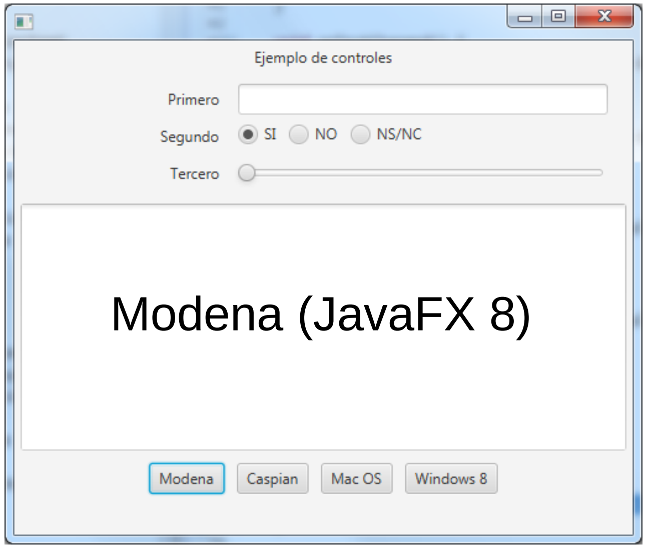
  - Add style sheets to the scene:

```
scene.getStyleSheets().add(
              getClass().getResource("skin.css").toExternalForm();
```

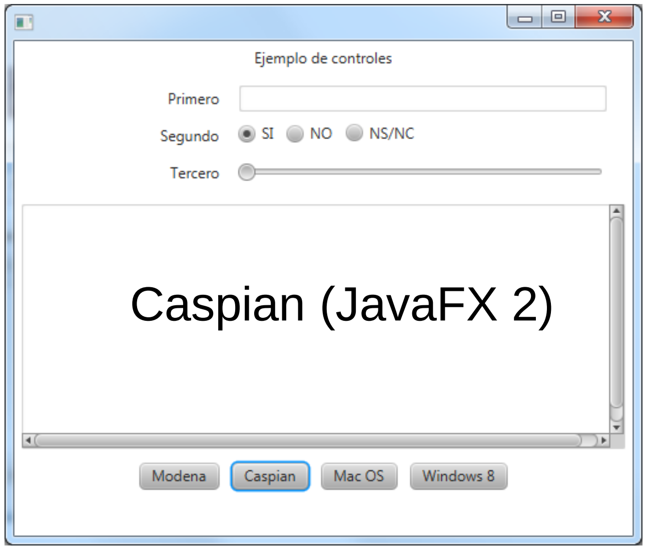
  - Note: the predefined style of a scene can also be overridden with:

```
scene.setUserAgentStylesheet(
    getClass().getResource("theme.css").toExternalForm());
```

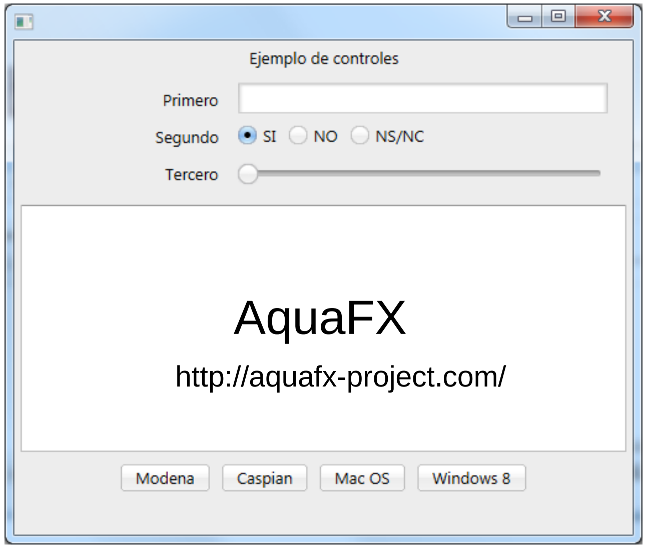
Changing the Theme …



Modena (JavaFX 8)

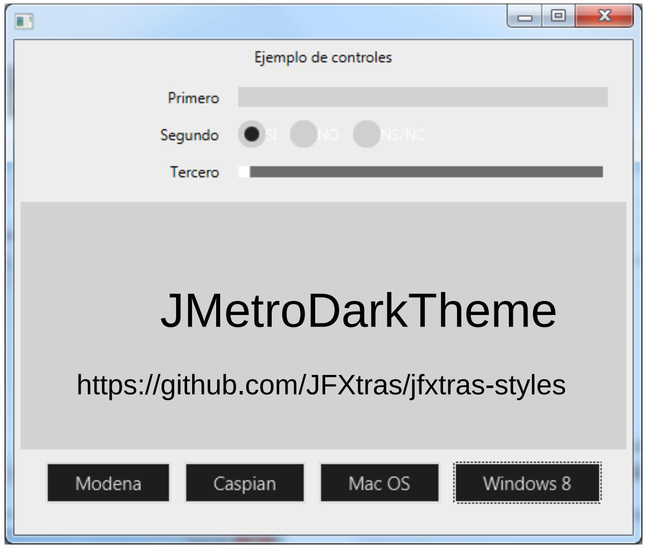


Caspian (JavaFX 2)



AquaFX

http://aquafx-project.com/



JMetroDarkTheme

https://github.com/JFXtras/jfxtras-styles

Changing the Theme …

- Default themes:
  - Modena (JavaFX 8)
    - `Application.setUserAgentStylesheet(Application.STYLESHEET_MODENA);`
    - `Application.setUserAgentStylesheet(null); // in JavaFX 8`
  - Caspian (JavaFX 2)
    - `Application.setUserAgentStylesheet(Application.STYLESHEET_CASPIAN);`

CSS Styles

- The styles in a CSS file are specified using blocks with the following syntax

```
<selector> {
    <property>: <value>;  // rules
    <property>: <value>;
     …
}
```

Selectors

- Selectors
  - Are used to select one or more nodes of a JavaFX’s scene graph to apply to them a given style
  - There are three types of identifiers that can be used to build selectors:
    - `class`: a style class can be assigned to several nodes and a node can have several style classes
    - `type selector`: coincides with the simple name of a class (`Label`, `HBox`, `CheckBox`…)
    - `id`: node identifier (different from `fx:id`)

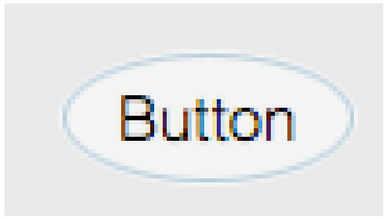
Selection of Nodes by Class

- A node can assigned classes using the method:

```
getStyleClass().add("num-button")
```

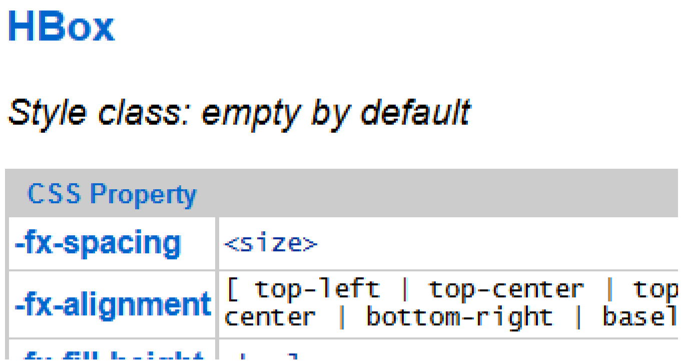
- To give a common style to several components, add to them the same class
- Then, you can add a block in the CSS file to establish the design of the node

```
.num-button {
-fx-background-color: white, rgb(189,218,230), white;
-fx-background-radius: 50%;
-fx-background-insets: 0, 1, 2;
-fx-font-family: "Helvetica";
-fx-text-fill: black;
-fx-font-size: 20px;
}
```



Predefined Classes

- JavaFX’s controls define their own classes
  - `button`, `check-box`, `combo-box`, `label`, `list-view`…
  - See: http://docs.oracle.com/javase/8/javafx/api/javafx/scene/doc-files/cssref.html
  - Watch out! Containers do not define style classes:



HBox

Style class: empty by default

CSS Property	
-fx-spacing	`<size>`
-fx-alignment	`[ top-left | top-center | top` `center | bottom-right | basel`

  - but they define a type selector that coincides with its class (`HBox`, `VBox`, `GridPane`…)

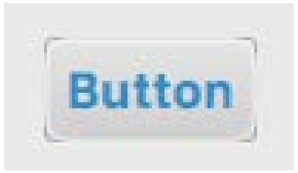
Node Selection by Id

- You can assign an id to a node using the method:
  - `void setId(String value)`

```
@FXML
Button modena;
…
modena.setId(“button-modena”);
```

- Then, you can add a block to the CSS file to establish the design of the node:

```
#button-modena {
-fx-text-fill: rgba(17, 145, 213);
-fx-border-color: rgba(255, 255, 255, .80);
-fx-border-radius: 8;
-fx-padding: 6 6 6 6;
-fx-font: bold italic 12pt "LucidaBrightDemiBold";
}
```



Other Selection Methods

- Descendant (for example any label inside a checkbox):

```
.check-box .label { -fx-text-fill: black; }
```

- Apply the style to all the buttons that are direct children of an HBox

```
HBox > .button { -fx-text-fill: black; }
```

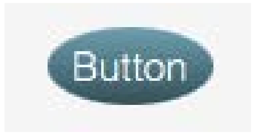
- Apply the same style to different classes

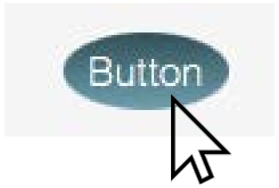
```
.label, .text { -fx-font-size: 20px; }
```

Pseudoclasses

- They are used to set the style of nodes that have different states
  - For example, a button has styles like: `hover`, `selected`, `focused`, etc.

```
.num-button {
-fx-background-color: linear-gradient(#61a2b1, #2A5058);
-fx-background-radius: 50%;
-fx-background-insets: 0, 1, 2;
-fx-font-family: "Helvetica";
-fx-text-fill: white;
}
.num-button:hover {
-fx-background-color: linear-gradient(#2A5058, #61a2b1);
}
```





Assigning a Style to a Node Using Code

- The `setStyle` method allows you to assign a style using code:

```
@FXML
Button win8;

win8.setOnMouseEntered(ae -> win8.setStyle("-fx-font-size: 15px"));
win8.setOnMouseExited(ae -> win8.setStyle(""));
```

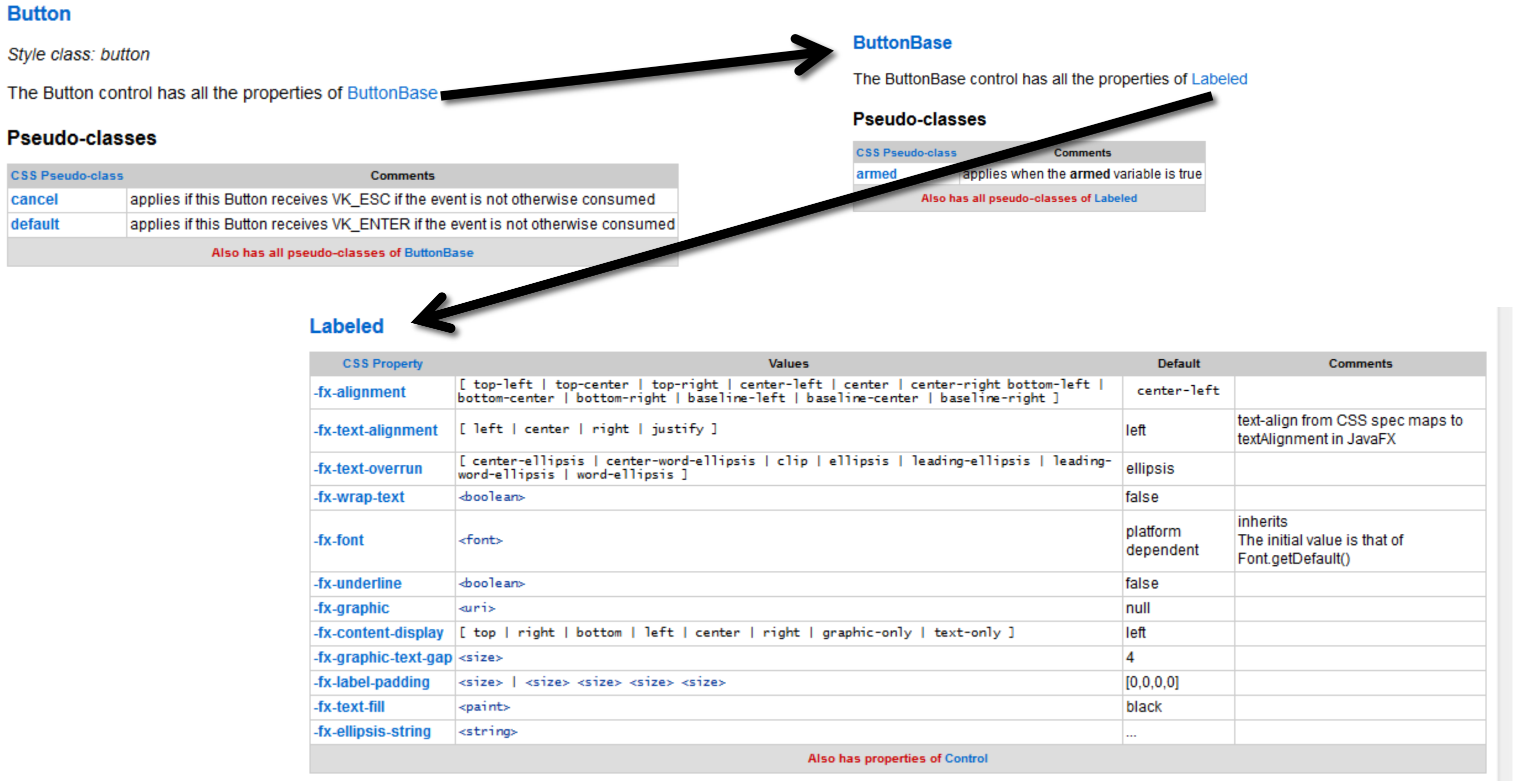
Style Inheritance

- The `.root` class is the root class of all the classes
  - Properties defined for `.root` will be applied to all the elements of the scene, unless the property is redefined for an element

```
.root {
-fx-font-size: 12px;
}
.button {
-fx-font-size: 20px;
}
```

Properties

- Which properties can I change for each node?
  - Check out the CSS Reference Guide
    - http://docs.oracle.com/javase/8/javafx/api/javafx/scene/doc-files/cssref.html



Button

Style class: button

The Button control has all the properties of ButtonBase

Pseudo-classes

CSS Pseudo-class	Comments
cancel	applies if this Button receives VK_ESC if the event is not otherwise consumed
default	applies if this Button receives VK_ENTER if the event is not otherwise consumed
Also has all pseudo-classes of ButtonBase	

ButtonBase

The ButtonBase control has all the properties of Labeled

Pseudo-classes

CSS Pseudo-class	Comments
armed	applies when the **armed** variable is true
Also has all pseudo-classes of Labeled	

Labeled

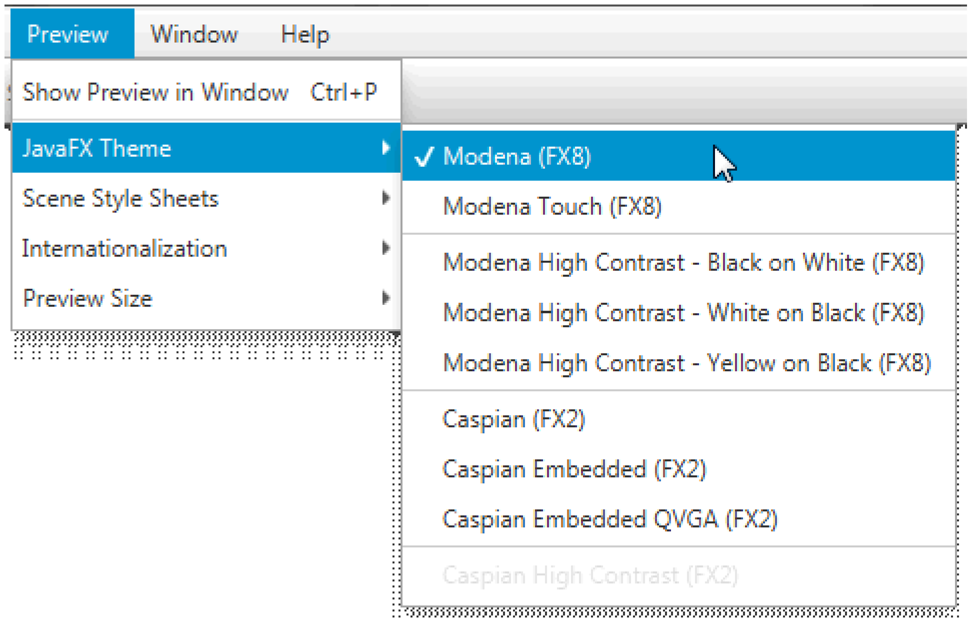
CSS Property	Values	Default	Comments
-fx-alignment	[top-left | top-center | top-right | center-left | center | center-right bottom-left | bottom-center | bottom-right | baseline-left | baseline-center | baseline-right]	center-left	
-fx-text-alignment	[left | center | right | justify]	left	text-align from CSS spec maps to textAlignment in JavaFX
-fx-text-overrun	[center-ellipsis | center-word-ellipsis | clip | ellipsis | leading-ellipsis | leading-word-ellipsis | word-ellipsis]	ellipsis	
-fx-wrap-text	<boolean>	false	
-fx-font	<font>	platform dependent	inherits The initial value is that of Font.getDefault()
-fx-underline	<boolean>	false	
-fx-graphic	<uri>	null	
-fx-content-display	[top | right | bottom | left | center | right | graphic-only | text-only]	left	
-fx-graphic-text-gap	<size>	4	
-fx-label-padding	<size> | <size> <size> <size> <size>	[0,0,0,0]	
-fx-text-fill	<paint>	black	
-fx-ellipsis-string	<string>	...	
Also has properties of Control			

CSS in SceneBuilder (I)

- Scene Builder allows any node in the scenegraph to select its predefined style
- It is necessary to determine the CSS file where that style resides, and the same of the selected style
- As discussed before, by default, JavaFX 8 projects uses the `modena.css`.
- Scene Builder will use the predefined style whenever a new control is dragged from the *Library* to the *Content* or the *Hierarchy* panels

CSS in SceneBuilder (II)

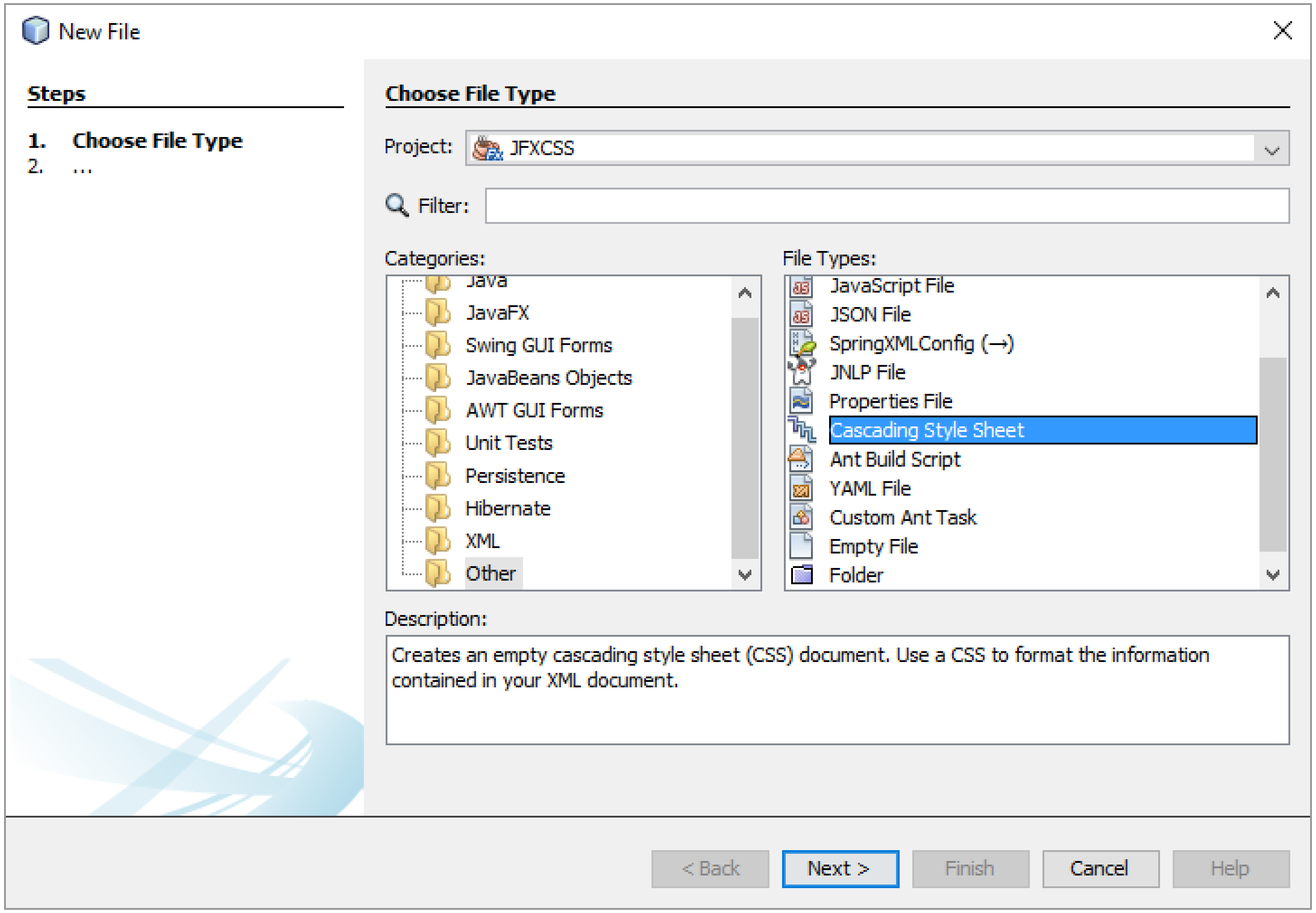
- It is possible to change the theme selecting an option in the menu Preview\JavaFX Theme
- There are specific themes based on *Modena* and *Caspian*


Preview
Window
Help
Show Preview in Window Ctrl+P
JavaFX Theme
Scene Style Sheets
Internationalization
Preview Size
✓ Modena (FX8)
Modena Touch (FX8)
Modena High Contrast - Black on White (FX8)
Modena High Contrast - White on Black (FX8)
Modena High Contrast - Yellow on Black (FX8)
Caspian (FX2)
Caspian Embedded (FX2)
Caspian Embedded QVGA (FX2)
Caspian High Contrast (FX2)

CSS in SceneBuilder (III)

- The CSS style sheets can be added to the whole scene, to a container or to any node in the scene
- The style can be customized changing the properties of a component using the Properties panel of the Inspector, or defining rules in a CSS file
- Scene Builder does not generate CSS files. It has to be created and edited with an external editor
- Scene Builder will update the interface as soon the CSS file is modified

CSS in SceneBuilder (IV)



New File

Steps

1. **Choose File Type**
2. ...

Choose File Type

Project: JFXCSS

Filter:

Categories:
- Java
- JavaFX
- Swing GUI Forms
- JavaBeans Objects
- AWT GUI Forms
- Unit Tests
- Persistence
- Hibernate
- XML
- Other

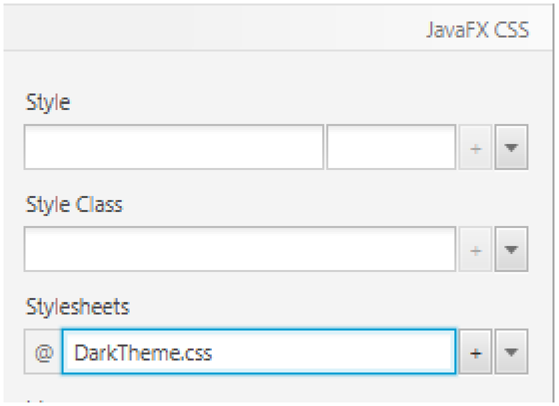
File Types:
- JavaScript File
- JSON File
- SpringXMLConfig (→)
- JNLP File
- Properties File
- Cascading Style Sheet
- Ant Build Script
- YAML File
- Custom Ant Task
- Empty File
- Folder

Description:

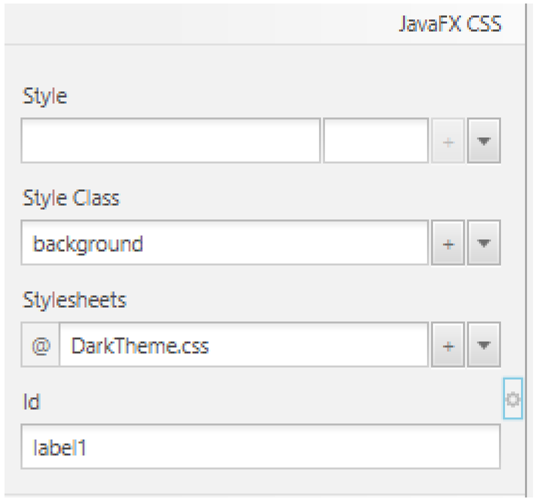
Creates an empty cascading style sheet (CSS) document. Use a CSS to format the information contained in your XML document.

< Back | Next > | Finish | Cancel | Help

CSS in SceneBuilder (V)



Stylesheets: @ DarkTheme.css



Style Class: background

Stylesheets: @ DarkTheme.css

Id: label1

Open an *fxml* file in Scene Builder

Select the root node of the *Hierarchy*

In the *Properties* panel, add the css file in the Stylesheets list (in the JavaFX CSS section)

Add classes or set the id to any node

The CSS styles defined in a parent element affect both the parent and its children

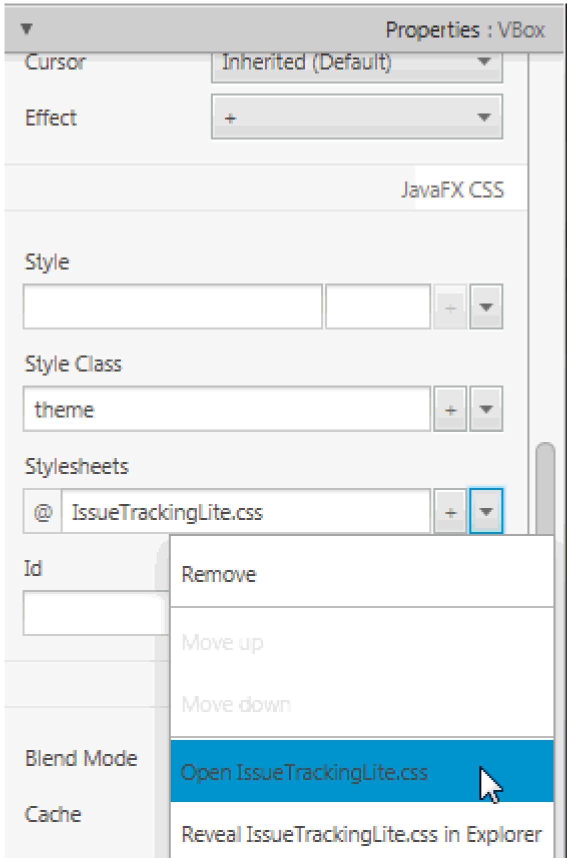
CSS in SceneBuilder (VI)

It is possible to edit a CSS file with any text editor:

1. In the *Properties* section in the *Inspector panel*, clic on the down arrow at the far right of the stylesheet
2. Select the comand Open *<file>.CSS*

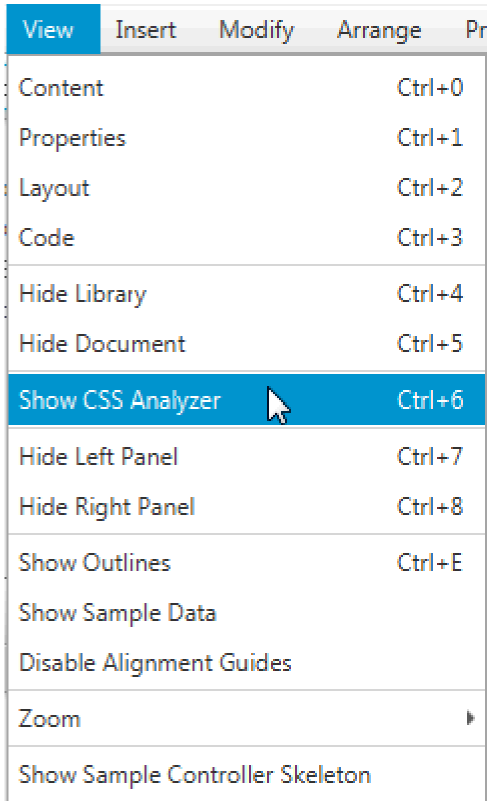
Reveal <file>.css in Explorer opens a file explorer showing the folder with the location of the CSS file

It is also possible to navigate to the CSS file using the CSS Analizer tool

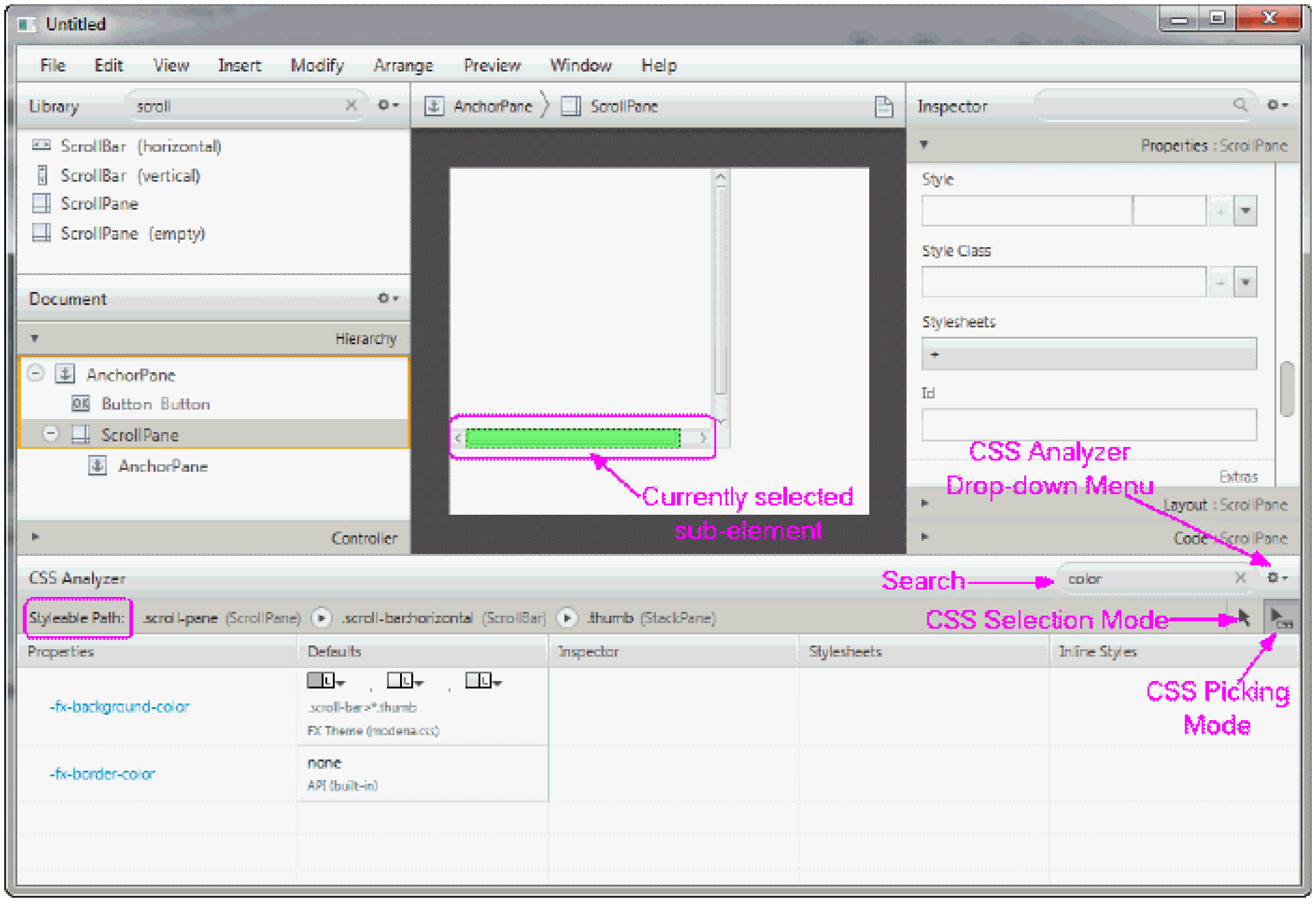


CSS Analyzer (I)

- It helps to interpret how several CSS rules affect visually to an element of the GUI
- It shows all the possible sources for the value of a property
- Any given property of a node can take its value from a number of CSS rules
- The sources are enumerated by priority, thus we figure out easily where a value comes from
- It allows to navigate to the source of the value of a CSS property
- Select the option `View\Show CSS Analyzer` to open its panel



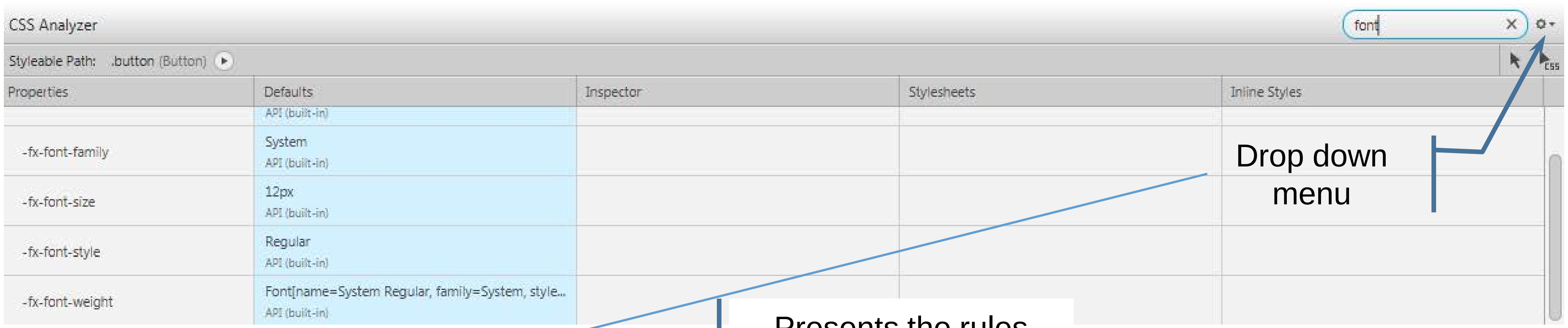
CSS Analyzer (II)



Properties	Defaults	Inspector	Stylesheets	Inline Styles
-fx-background-color	.scroll-bar>*.thumb FX Theme (modena.css)			
-fx-border-color	none API (built-in)			

CSS Analyzer (III)

Filtering properties by name



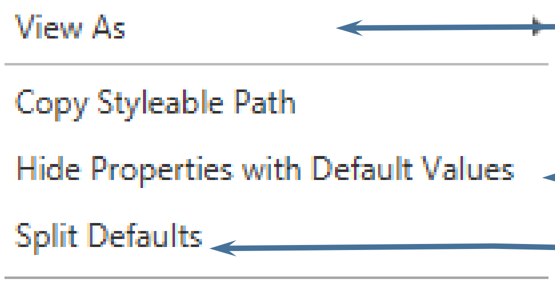
CSS Analyzer font

Styleable Path: .button (Button)

Properties	Defaults	Inspector	Stylesheets	Inline Styles
	API (built-in)			
-fx-font-family	System API (built-in)			
-fx-font-size	12px API (built-in)			
-fx-font-style	Regular API (built-in)			
-fx-font-weight	Font[name=System Regular, family=System, style... API (built-in)			

Drop down menu

Presents the rules using different formats: **Tables**, **Rules** and **Text**

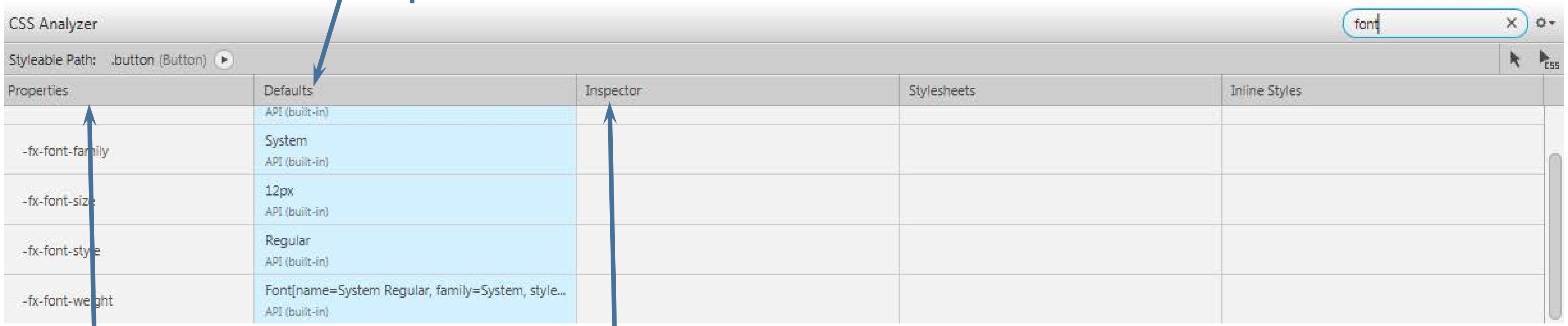


It shows only those properties with a customized value

Separates in two columns the values predefined by the API and FX themes
Join Defaults joins both columns again

CSS Analyzer (IV)

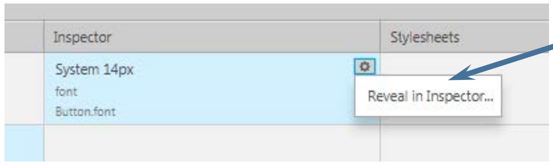
Default values (API and JavaFX theme)



Properties	Defaults	Inspector	Stylesheets	Inline Styles
	API (built-in)			
-fx-font-family	System API (built-in)			
-fx-font-size	12px API (built-in)			
-fx-font-style	Regular API (built-in)			
-fx-font-weight	Font[name=System Regular, family=System, style... API (built-in)			

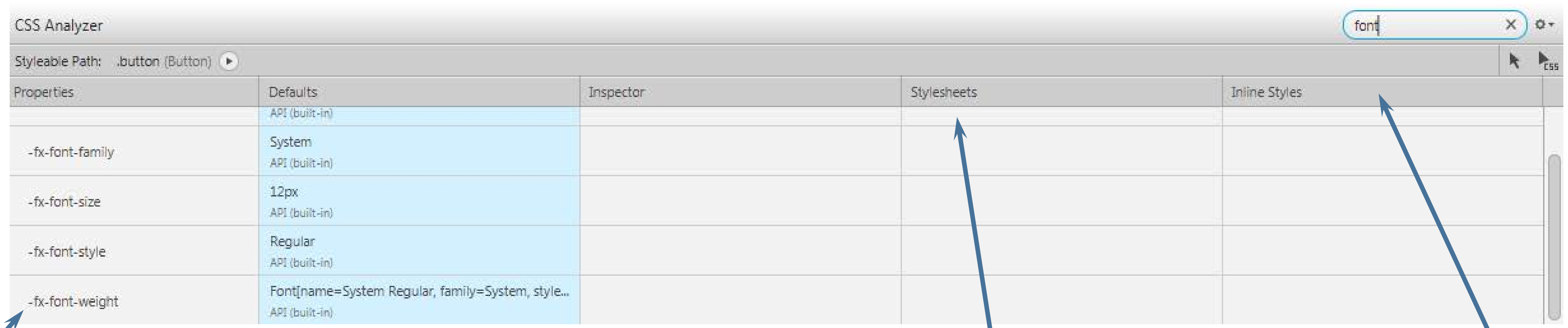
Available styling properties for the currently selected node

Value of the property set using the Inspector panel



Highlights the property in blue in the Inspector panel

CSS Analyzer (IV)



CSS Analyzer — font

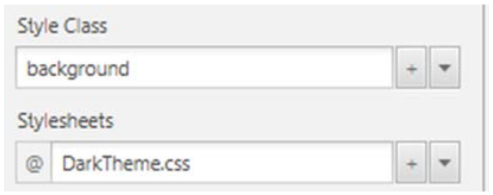
Styleable Path: .button (Button)

Properties	Defaults	Inspector	Stylesheets	Inline Styles
	API (built-in)			
-fx-font-family	System API (built-in)			
-fx-font-size	12px API (built-in)			
-fx-font-style	Regular API (built-in)			
-fx-font-weight	Font[name=System Regular, family=System, style... API (built-in)			

Clicking on the name of the property opens the online documentation in the JavaFX CSS Reference Guide

Property established in a stylesheet in the node or in one of its ancestors

Values established in the Style section of the Properties section in the Inspector

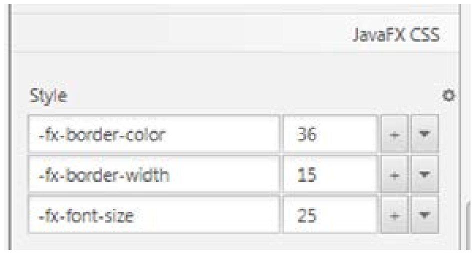


Style Class

background

Stylesheets

@ DarkTheme.css

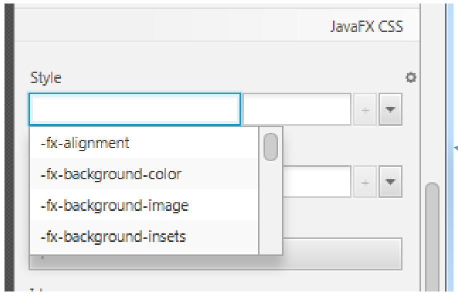


JavaFX CSS

Style

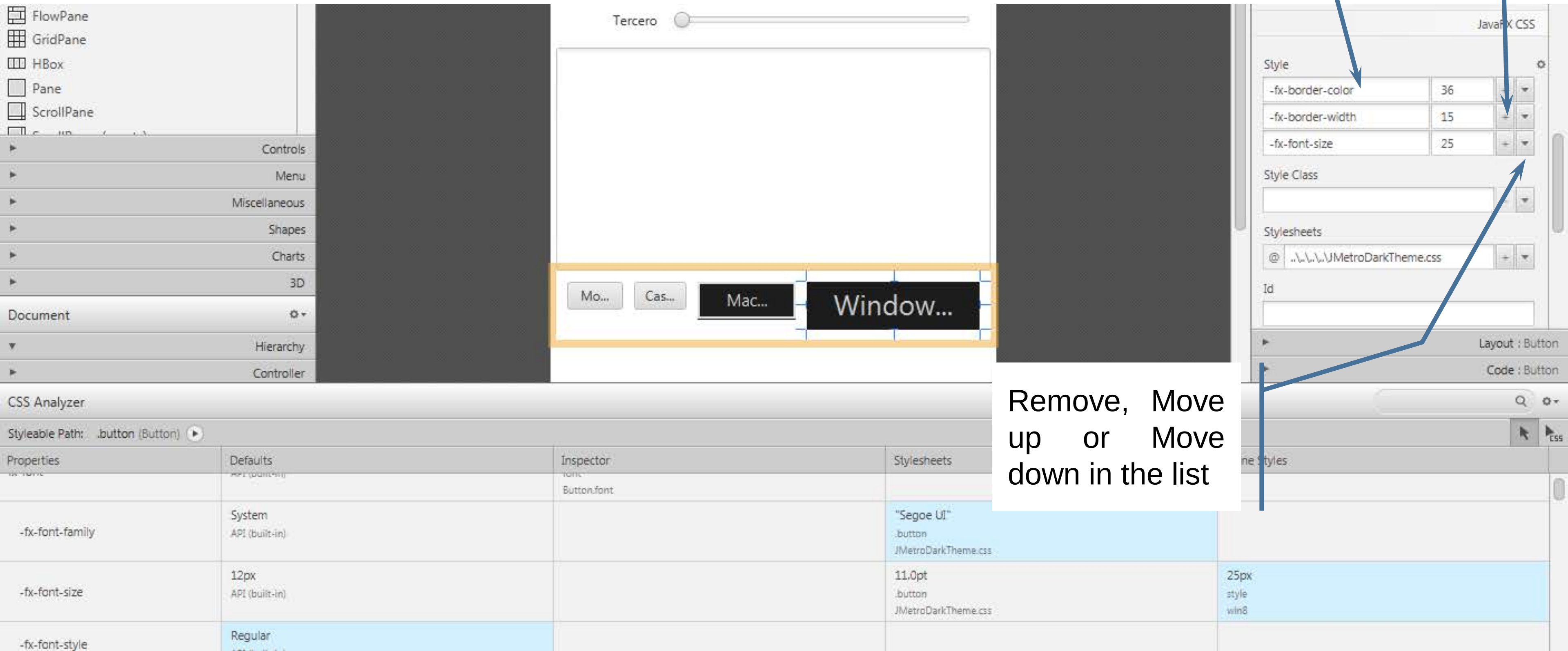
-fx-border-color	36
-fx-border-width	15
-fx-font-size	25

Panel analizador de CSS (V)



When clicking on the text field, a list with all the available properties pops up

The + button adds new properties



Style	
-fx-border-color	36
-fx-border-width	15
-fx-font-size	25

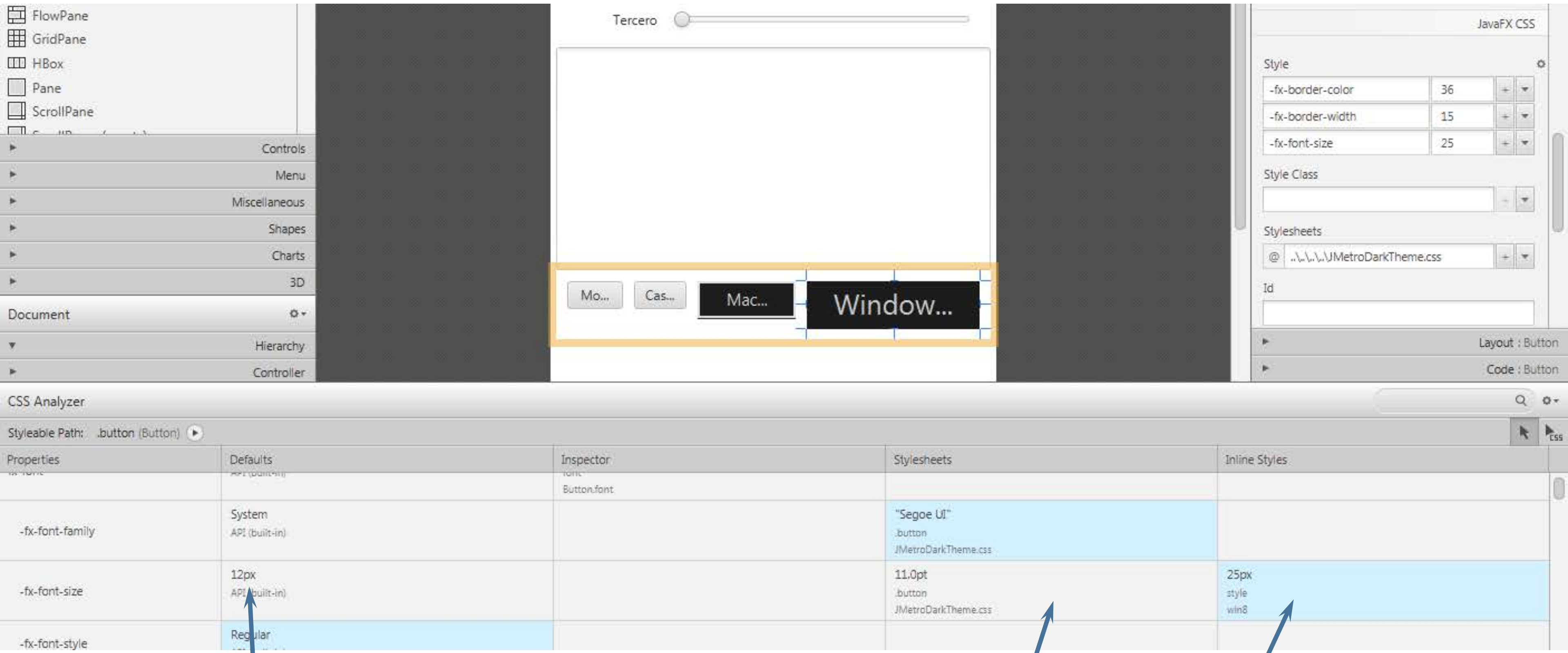
Stylesheets: @ ..\..\..\..\JMetroDarkTheme.css

Styleable Path: .button (Button)

Properties	Defaults	Inspector	Stylesheets	...ne Styles
-fx-font-family	System API (built-in)		"Segoe UI" .button JMetroDarkTheme.css	
-fx-font-size	12px API (built-in)		11.0pt .button JMetroDarkTheme.css	25px style win8
-fx-font-style	Regular			

Remove, Move up or Move down in the list

Panel analizador de CSS (V)



Style

-fx-border-color	36
-fx-border-width	15
-fx-font-size	25

Style Class

Stylesheets: @ ..\..\..\..\JMetroDarkTheme.css

Id

Styleable Path: .button (Button)

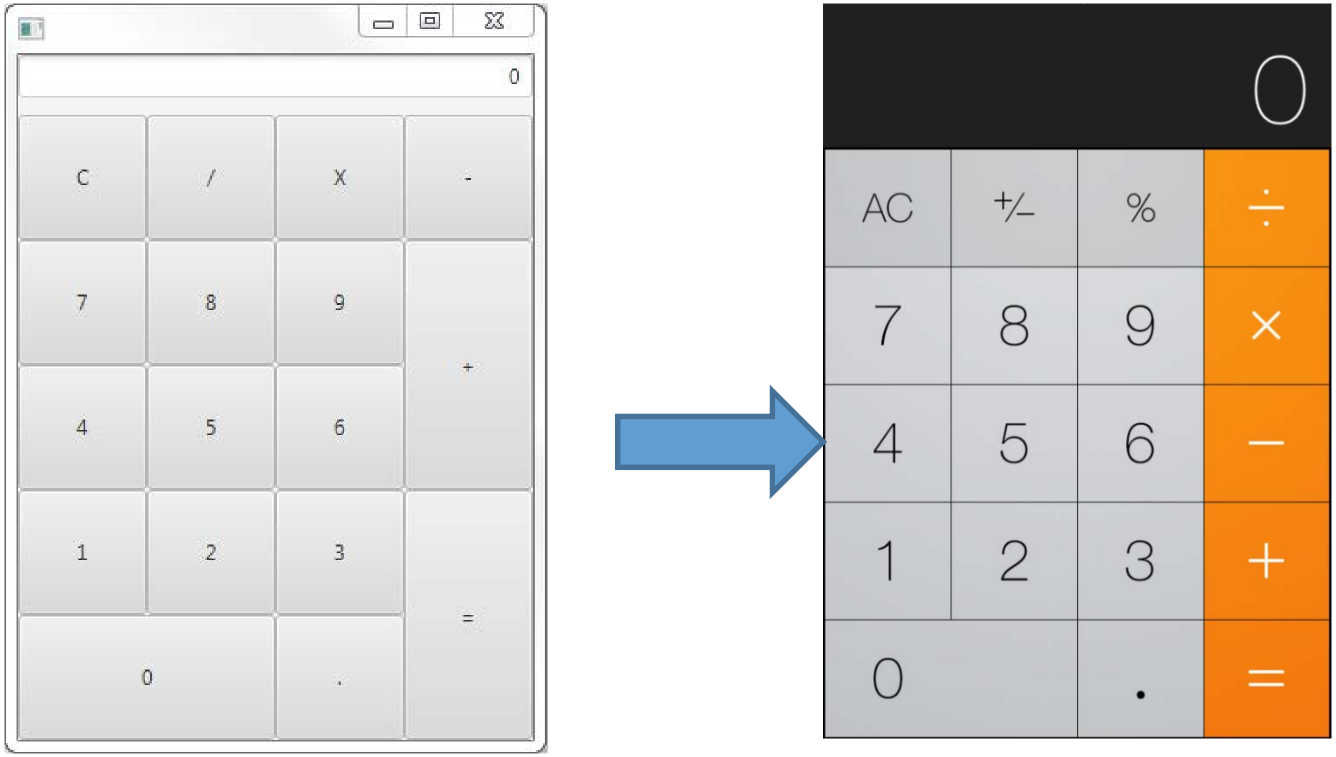
Properties	Defaults	Inspector	Stylesheets	Inline Styles
-fx-font	API (built-in)	font Button.font		
-fx-font-family	System API (built-in)		"Segoe UI" .button JMetroDarkTheme.css	
-fx-font-size	12px API (built-in)		11.0pt .button JMetroDarkTheme.css	25px style win8
-fx-font-style	Regular			

If no value is provided, the properties takes the default value

Then, the properties of the stylesheet

The explicit styles have maximum priority

Exercise



- Modify the calculator built in the Lab 2 for making it similar to the iOS calculator using CSS
- Make the buttons to change their color when pressed

References

- C. Dea et al. JavaFX 8. Introduction by Example. Apress. 2014
  - Chapter 6
- Oracle.
  - Skin Applications with CSS
    - http://www.oracle.com/pls/topic/lookup?ctx=javase80&id=JFXUI733
  - CSS Reference Guide
    - http://docs.oracle.com/javase/8/javafx/api/javafx/scene/doc-files/cssref.html
  - JavaFX Scene Builder: User Guide
    - http://docs.oracle.com/javase/8/scene-builder-2/user-guide/index.html