JavaFX Scene Builder

User Guide

Release 2.0

E51279-01

April 2014

This user guide introduces you to and describes how to use the JavaFX Scene Builder features and graphical user interface (GUI).

JavaFX Scene Builder User Guide Release 2.0

E51279-01

Primary Author: Cindy Castillo

Contributor: Yves Joan

Contents

Preface

This preface gives an overview about this tutorial and also describes the document accessibility features and conventions used in this tutorial - *JavaFX Scene Builder User Guide*.

About This Tutorial

This guide introduces you to the graphical user interface (GUI) and describes the features of the JavaFX Scene Builder, a design tool for the JavaFX platform.

Audience

This document is intended for JavaFX developers.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc`.

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info` or visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs` if you are hearing impaired.

Related Documents

For more information, see the following documents in the JavaFX Scene Builder and JavaFX documentation sets:

- *JavaFX Scene Builder Installation Guide*
- *JavaFX Scene Builder Release Notes*
- *Getting Started with JavaFX Scene Builder*
- *Using JavaFX Scene Builder with Java IDEs*
- *Mastering FXML*

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1 Introduction

This chapter discusses what is required before using this JavaFX Scene Builder User Guide and gives a brief introduction on what is covered in the document.

It is assumed that you have already installed Java SE Development Kit 8 (JDK 8) and the JavaFX Scene Builder product on your system. You also should have downloaded the Scene Builder samples bundle file that is provided at the same download site. See the JavaFX Scene Builder Installation Guide for more detailed information.

The following topics describe the different JavaFX Scene Builder features.

- Starting Up Scene Builder
- Getting to Know the Main Window
- Using the Menu Bar Commands
- Using the Path, Selection and Message Bar
- Working with the Content Panel
- Designing GUI with the Library Panel
- Using the Document Panel
- Defining Properties in the Inspector Panel
- Styling with CSS and the CSS Analyzer
- Internationalizing Your FXML Layout
- Next Step

2

Starting Up Scene Builder

This chapter describes how to start JavaFX Scene Builder and how to open an existing FXML file, such as the IssueTrackingLite sample provided with the downloadable Scene Builder samples.

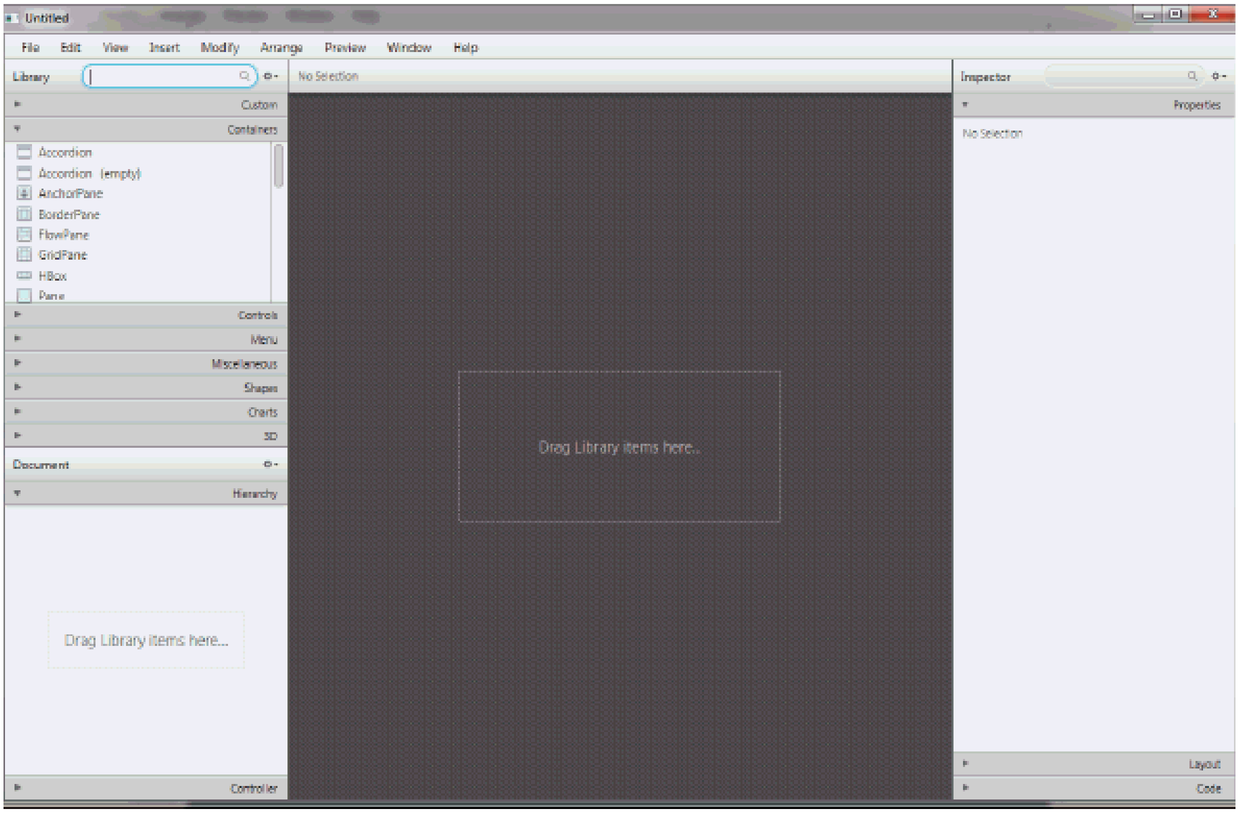
You start a standalone version of JavaFX Scene Builder by using one of the following methods:

- On Windows 7 or earlier platforms, do one of the following steps:
  - Double-click the **JavaFX Scene Builder 2.0 desktop icon.**
  - Select **Start**, then **All Program**, then **JavaFX Scene Builder**, and finally, **JavaFX Scene Builder 2.0.**
- On Windows 8 platform, use the Search field and type any part of `JavaFX Scene Builder 2.0` to locate it. Alternatively, use the `Apps View` to locate JavaFX Scene Builder from the list of installed applications on your system.
- On the Mac OS X platform, double-click the **JavaFX Scene Builder 2.0** shortcut in the Applications folder.
- On the Ubuntu platform, Scene Builder appears in the Dash desktop search utility after it is installed. Alternatively, you can launch the Scene Builder executable from **/opt/JavaFXSceneBuilder2.0/JavaFXSceneBuilder2.0**.

The initial window that you see is the main window, as shown in Figure 2–1. It displays an empty and untitled FXML file.

An element is a principal visual object that you can add to the Content panel and manipulate when editing an application's GUI layout in JavaFX Scene Builder. An element can be dragged from the Library panel of predefined GUI elements onto the Content or Hierarchy panel. You can also group elements so that you can assign property and layout values on multiple elements at the same time.

Figure 2–1 JavaFX Scene Builder Startup Window



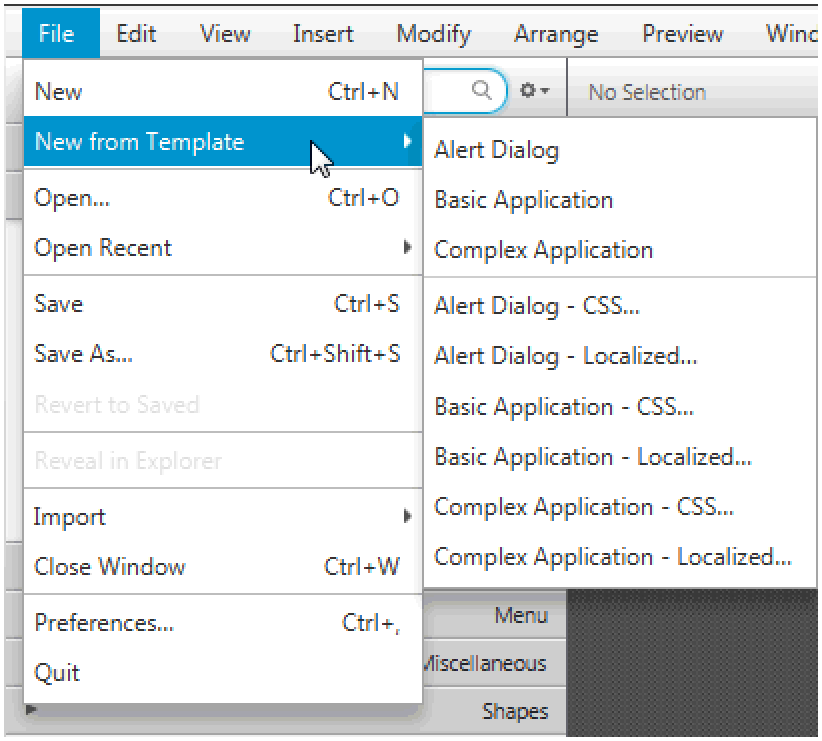
From within Scene Builder, you can also open an existing FXML document from the Menu bar using one of the following steps:

- From the **File** menu, select **Open.**
- From the **File** menu, select **Open Recent** and select the file from the list of recently opened FXML files.

You can also open a new FXML file from an opened Scene Builder window using the following steps:

- From the **File** menu, select **New.**
- From the **File** menu, select **New from Template** and select from the list of available FXML templates, as shown in Figure 2–2.

Figure 2–2 New from Template File Command



For illustration purposes, use the `IssueTrackingLite.fxml` sample file that is included with the JavaFX Scene Builder samples bundle.

1. Go to where you extracted the `javafx_scenebuilder_samples-2_0.zip` file.
2. Expand the **IssueTrackingLite**, **src,** and **issuetrackinglite** folders.
3. Open the `IssueTrackingLite.fxml` file.
   The file is opened in Scene Builder, if you have Scene Builder properly installed.

You can also use Scene Builder with one of the Java IDEs, such as NetBeans IDE, Eclipse, or IntelliJ IDEA, to create a new FXML file or open an existing one. Use the information in Using JavaFX Scene Builder with Java IDEs to learn more.

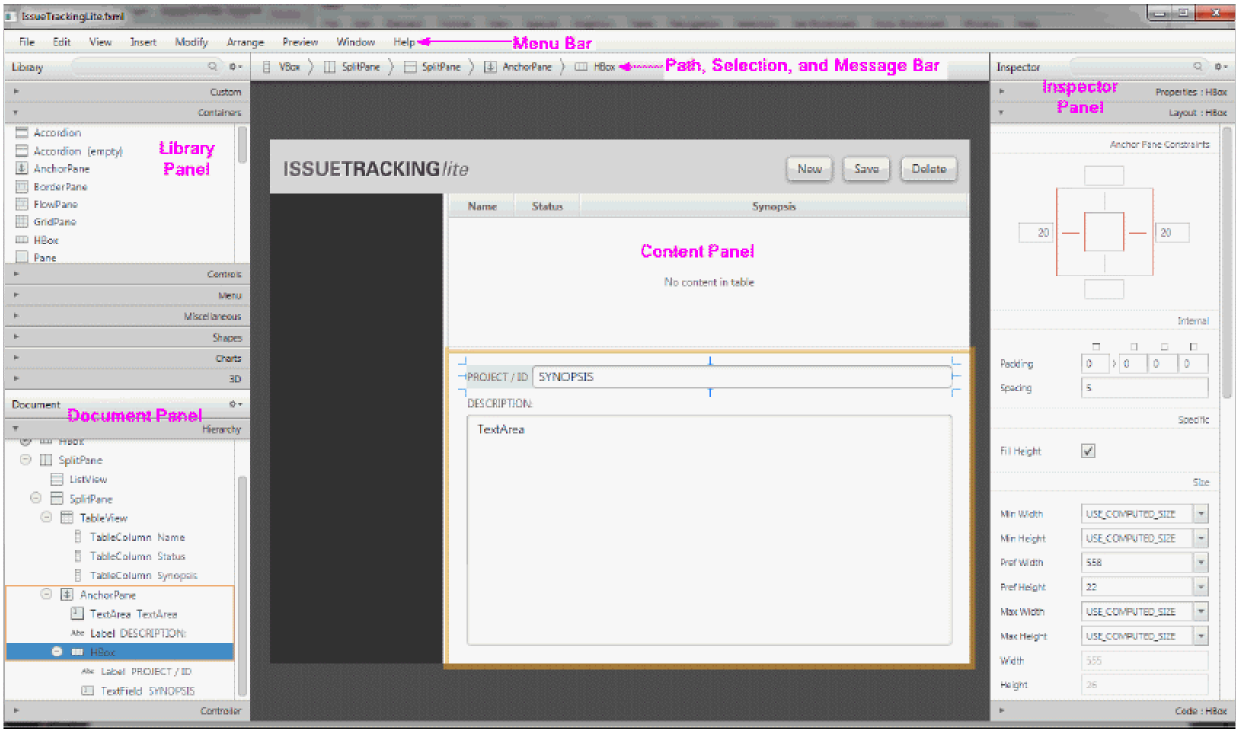
3

Getting to Know the Main Window

This chapter discusses the different major parts of the main window that appears after Scene Builder is started up.

After you open the FXML file to work with, it's contents are displayed in the Content panel of the main window, similar to what is shown in Figure 3–1.

Figure 3–1 Main Window of JavaFX Scene Builder



By default, the main window of JavaFX Scene Builder includes the following sections, which are labeled in Figure 3–1.

- **Menu Bar**: Provides access to the menu of commands available in JavaFX Scene Builder.
- **Path, Selection and Message Bar**: Displays the path to a selected element and allows you to select an element to put into focus. It also displays any error or status messages.

- **Content Panel**: The scene container for the GUI elements that make up your FXML layout. By default, a new empty FXML file is opened in JavaFX Scene Builder.
- **Library Panel**: Lists the available JavaFX GUI elements or controls, including custom controls, that you can use to build your FXML layout. You select the GUI elements from this panel and add them to the Content panel or the Hierarchy panel.
- **Document Panel**: Contains the Hierarchy and Controller sections. The Hierarchy section displays a tree view representation of the FXML layout that you are building in the Content panel. Elements that are not visible in the Content panel can be placed into focus by selecting it in the Hierarchy panel. The Controller section enables you to manage the controller source information and gives information about assigned fx:id values.
- **Inspector Panel**: Contains the Properties, Layout, and Code sections. The Properties and Layout sections help you manage the properties of the currently selected GUI element in the Content panel or in the Hierarchy panel. The Code section enables you to manage the event handling actions to use for the selected GUI element. The Inspector panel also contains a Search text field that enables you to isolate specific properties that you want to modify.

The following panel is also displayed in the main window when you select **View** from the Main menu and then **Show CSS Analyzer.**

- **CSS Analyzer Panel**: Allows you to explore all the CSS properties available for a JavaFX component on your FXML layout and helps you to build the CSS rules

The next chapters describe the parts listed above in more detail.

4

Using the Menu Bar Commands

This chapter describes the commands that are available from the Scene Builder main Menu bar.

The Menu bar displays the menus of commands that you can perform with an FXML layout file. The following sections describe the list of subcommands available when you select a command from the Menu bar. Table 4–1 describes the subcommands for the File command.

> **Note:** The shortcut key shown in the tables below are for Windows platform. Replace the Ctrl key with the Cmd key to use the equivalent shortcut key for the Mac OS X platform.

Table 4–1 List of File Subcommands

Command (Shortcut Key)	Description
New (Ctrl+N)	Create a new FXML file and opens it in a separate Scene Builder window.
New from Template	Create a new FXML layout using one of the available templates: Alert Dialog, Basic Application, Complex Application, Alert Dialog - CSS, Alert Dialog - Localized, Basic Application - CSS, Basic Application - Localized, Complex Application - CSS, and Complex Application - Localized layouts.
Open (Ctrl+O)	From an Open FXML dialog box, select the existing FXML file to open.
Open Recent	Select the file to open from a list of recently opened FXML files.
Save (Ctrl+S)	Save the changes made since the last Save action was performed. If it is a new FXML document, you are prompted to provide the new file name. If it is an existing FXML file, the current file name is retained.
Save As (Ctrl+Shift+S)	Save the current FXML file you are working on, with the option to save the changes to a different file.
Revert to Saved	Undo the changes made since the last Save action was performed.
Reveal in Explorer (Windows), Reveal on Desktop (Linux), Reveal in Finder (Mac OS X)	Open a native file browser, which displays the folder containing the current FXML file and gives you the opportunity to open the file.

Table 4–1 (Cont.) List of File Subcommands

Command (Shortcut Key)	Description
Import	Import an existing image, FXML, or media file. The Import > FXML command inserts the content of the imported file into the current FXML file you are working on. An image or media file should be imported into Scene Builder from inside your project sources directory. If the image or media file you are importing is located somewhere other than your project sources directory, you need to move or copy the file. Scene Builder does not copy the imported image or media file for you. It makes a link to the imported media or image file using a path relative to the location of the FXML file currently being worked on. Consequently, if the relative location of the imported image or media file is different at runtime, then your FXML file may not load properly. You need to ensure all imported files are packaged properly by your IDE and that their relative location at runtime and edit time are the same. The simplest way to achieve this is to only import image or media files which are already located under your project sources directory and to instruct your IDE to include these files in the packaging, which is usually done by default.
Include	Insert FXML, Edit the Included File, or Reveal the Included File. Insert FXML command inserts an fx:include construct into your FXML document. Edit the Included File command allows you to edit any imported or included FXML file that is referenced by an fx:include statement defined for the selected element. When you edit an included FXML file, a new JavaFX Scene Builder window is opened. Changes you save in the included FXML file are reflected in the FXML layout file from which you opened the included FXML file. Reveal Included File command opens a native file browser to show the folder that contains the included file.
Close Window (Ctrl+W)	Close the current window. If you have any unsaved changes, you are prompted to save the file first before the window is closed. All other active windows remain open until they are closed or you exit the tool.
Preferences (Ctrl+,)	Set the preferred values to use for certain properties in the tool. You can change the default values for the root container's width and height, the image to use for the tool's background, the colors to use for the alignment guides and drop/parent rings, the theme to use for the Scene Builder tool, the default display mode to use for the Hierarchy and Library panels, the order in which the CSS Analyzer columns are displayed, and the number of files to list in the Open > Recent Items command. You can reset to the default values by clicking **Reset to Built-in Default Values**.
Quit	Exit the tool. All opened Scene Builder windows are closed. You are prompted to save any unsaved changes.

Selecting the **Edit** command menu displays the subcommands that enable you to select the **Undo** or **Redo** command on the last action that you made in the current FXML layout. In addition, you can select the **Cut**, **Copy**, **Paste**, **Paste Into, Duplicate**, and **Delete** commands that you can apply to the selected elements in the Content or Hierarchy panel. The Cut, Copy, and Paste commands use the native editor and clipboard, unless a text field element is selected. This means that when you copy or cut an element and paste it on a text editor, what is actually copied and pasted is the

FXML code that defines a Group element that wraps the selected element and all its children elements. If the focus is on a text field element then the select, copy, and paste commands are performed from and to the text field element. With the Paste Into command, you need to first select where you want to paste an item into before selecting the Paste Into command.

Select the **Select All** command to select all of the elements in your FXML layout, while the **Select None** command deselects any selected elements. The **Select Parent** command selects the parent container of the selected element. **Select Next** and **Select Previous** commands allow you to select next and previous element in the hierarchy, respectively.

Select the **Trim Document to Selection** command to delete all of the elements up to the selected row in the Hierarchy panel. When you select the Trim Document to Selection command and the selected element is for a container element, that selected container element becomes the new root node container of your FXML document.

Table 4–2 describes the subcommands available when you select the **View** command from the Menu bar.

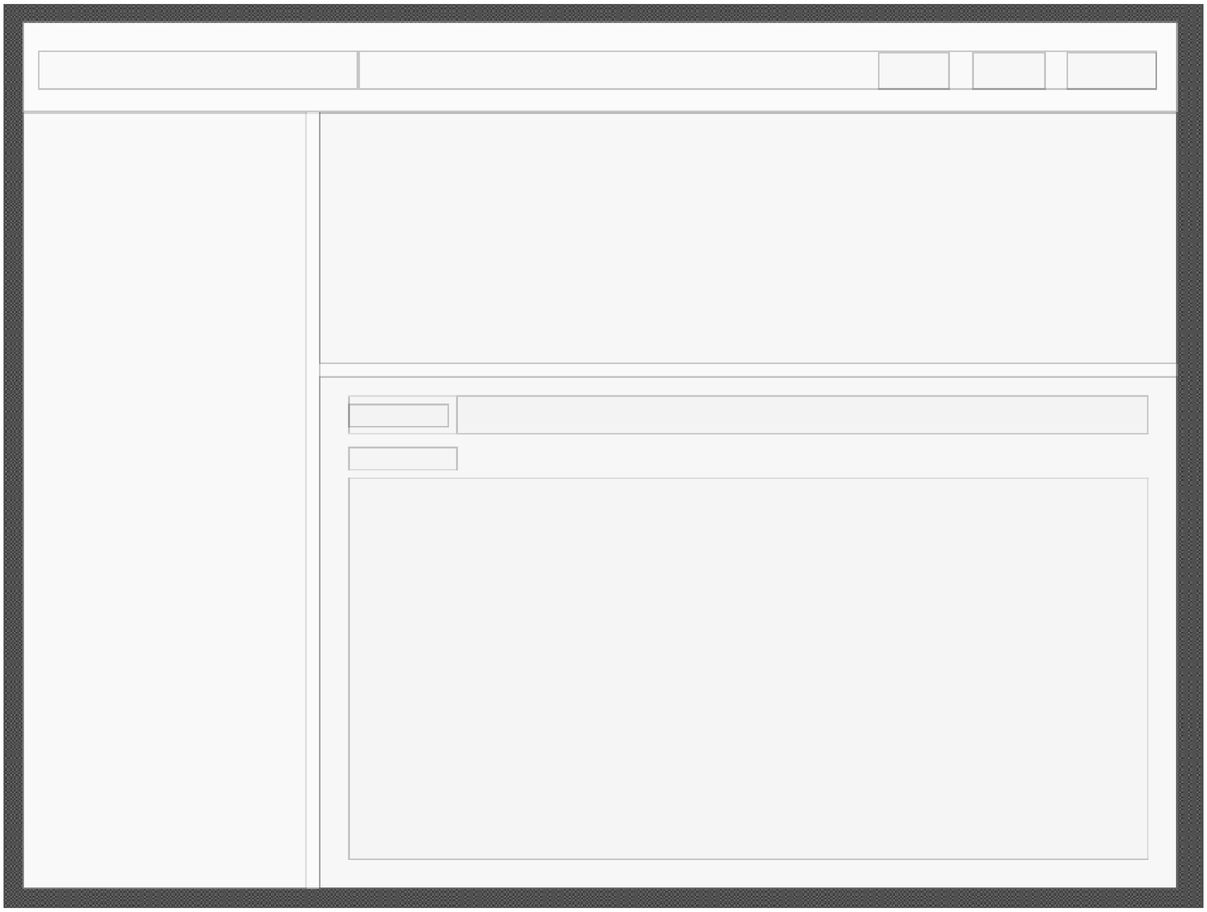
Table 4–2 List of View Subcommands

Command (Shortcut key)	Description
Content (Ctrl+0)	Place focus on the Content panel.
Properties (Ctrl+1)	Open the Properties section of the Inspector panel and place the current focus on it.
Layout (Ctrl+2)	Open the Layout section of the Inspector panel and place focus on it.
Code (Ctrl+3)	Open the Code section of the Inspector panel and place focus on it.
Hide/Show Library (Ctrl+4)	Hide or show the Library panel on the left side of the current window. If the Document panel is still visible, it automatically adjusts to occupy the entire left side of the window. If the Document panel is not visible, the Content panel adjusts to occupy the rest of the window's left side.
Hide/Show Document (Ctrl+5)	Hide or show the Document panel. If the Library panel is still visible, it is automatically adjusted to occupy the entire left side of the window. If the Library panel is not visible, the Content panel is automatically adjusted to occupy the rest of the window's left side.
Show/Hide CSS Analyzer (Ctrl+6)	Show or hide the CSS Analyzer panel. The panel appears at the bottom of the window.
Hide/Show Left Panel (Ctrl+7)	Hide or show the panels on the left side of the window. Both the Library and Document panels are displayed with the Show Left command, regardless if the Document or Library panel was the only left panel active when the Hide Left Panel command was last executed.
Hide/Show Right Panel (Ctrl+8)	Hide or show the Inspector Panel on the right side of the window.
Show/Hide Outlines (Ctrl+E)	Show or hide the outline of each element in your entire layout, as shown in Figure 4–1. The outline shows the LayoutBounds property of each element. See the API documentation for more information about the `LayoutBounds` property of the `Node` class in the `javafx.scene` package.

Table 4–2 (Cont.) List of View Subcommands

Command (Shortcut key)	Description
Show/Hide Sample Data	Show or hide demonstration data for those elements, such as a tree view, that have some sample data associated with them. The sample data is not saved in your FXML layout file.
Disable/Enable Alignment Guides	Disable or enable the alignment guidelines that help you align the elements with one another.
Zoom	Provides a drop-down list of percentage values into which you can zoom the current layout.
Show Sample Controller Skeleton	Open a window that displays sample skeleton code that you can use to create a controller class for the FXML layout you are building

Figure 4–1 Show Outlines



Select the **Insert** command from the Menu bar to enable you to select GUI elements, such as containers, controls, menu content, shapes, charts, custom components, 3D, and miscellaneous elements, that you can add to the current FXML layout. These are the same GUI elements available from the Library panel. If none of the existing GUI elements in the layout is selected, the insertion point for the new element is the center of the Content panel. If one of the elements is selected before selecting the Insert command, the new element being added is placed slightly off to the side, but over the selected element. The Insert submenu displays only the valid GUI controls that can be inserted to the selected element or container.

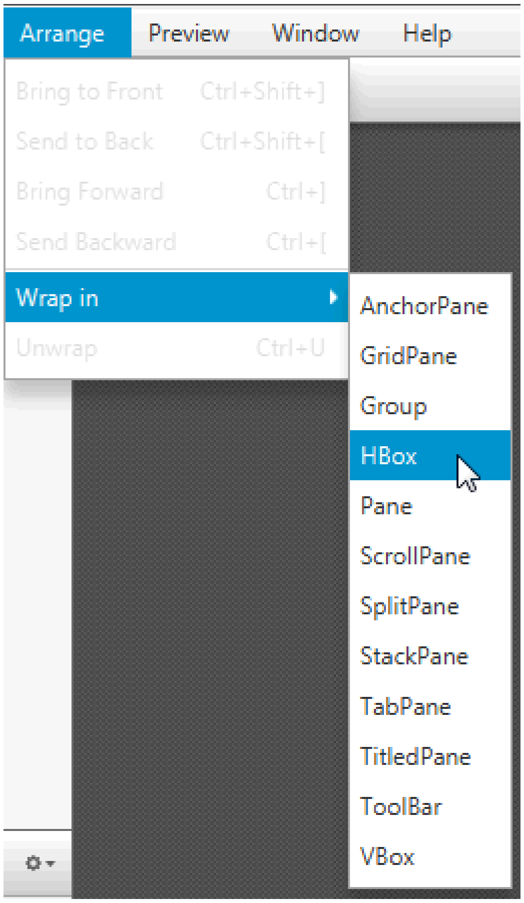
The subcommands displayed when you select **Modify** from the Menu bar are described in Table 4–3.

Table 4–3 List of Modify Subcommands

Command (Shortcut Key)	Description
Fit to Parent (Ctrl+K)	Resize a selected element to fill the area that its parent container occupies. The element is also anchored to each of the sides of the parent. This feature is only available when the parent container is an AnchorPane element.
Use Computed Sizes (Ctrl+Shift+K)	Reset the selected element's layout property values to the default values. The USE_COMPUTED_SIZE value is used for the Min Width, Min Height, Pref Width, Pref Height, Max Width, or Max Height layout properties. (See the API documentation for the `Region` class in the `javafx.scene.layout` package for more information.)
Grid Pane	Display all the subcommands available for working with the selected Grid Pane component in your layout. You can delete, move, add, or resize rows or columns in the selected Grid Pane.
Add Effect	Apply effects, such as a drop shadow or a reflection, to the selected element or group of elements in the layout.The Effect editor allows you to set chain of effects to an object.
Add Popup Control	Add a Context Menu or a Tooltip element to the selected element in your layout. Note that the added Popup control is only selectable in the Hierarchy panel.
Scene Size	Set the screen size by selecting from the list of pre-defined dimensions.

Use the **Arrange** menu, which is shown in Figure 4–2, to bring the selected element to the front, send it back, or move it forward or backward. Selecting the **Wrap In** subcommand places the selected elements into one of the available containers, such as an Anchor Pane, HBox, or a Grid Pane container. The **Unwrap** command removes the currently selected container, but leaves the GUI elements it contains untouched.

Figure 4–2 Menu Bar with Arrange Menu Opened



The subcommands displayed when you select the **Preview** command from the Menu Bar are described in Table 4–4.

Table 4–4 List of Preview Subcommands

Command (Shortcut Key)	Description
Show Preview in Window (Ctrl+P)	Display the window that lets you preview the user interface design rendered by your current FXML code when it is deployed.
JavaFX Theme	Set the JavaFX Theme to use on the FXML layout being previewed. The drop-down list shows the available selection.
Scene Style Sheets	Open a submenu that displays commands that enable you to add a new style sheet that you can apply to your current layout, remove a style sheet from the list of style sheets being used, or open an existing style sheet in a separate editor window specified for the style sheet file type. The layout is immediately updated.
Internationalization	Enables you to set, remove, or reveal a resource file in the native file browser.
Preview Size	Allows you to set the size of the Preview window to one of the predefined scene sizes as an alternative to the preferred size defined in the FXML document.

Selecting the **Window** menu gives you a list of the names of the FXML file s that are currently opened in JavaFX Scene Builder windows. When you select one of the file names in the list, the name is marked with a check mark, and its corresponding Scene Builder window becomes active.

Selecting **Help** and then **Scene Builder Help** opens your default web browser and displays the web site that contains the available JavaFX Scene Builder documentation. Select the **About Scene Builder** command to open a window that displays the release information for the JavaFX Scene Builder you are using.

5

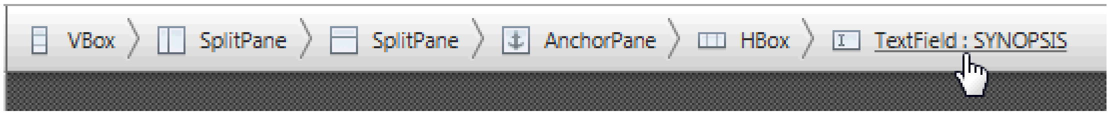
Using the Path, Selection and Message Bar

This chapter describes the Selection and Message bar that is located just above the Content panel of the Scene Builder main window.

The Path, Selection and Message bar is shown in Figure 5–1 and Figure 5–2. It serves the following purposes:

- It displays the containment path of the selected GUI element. It enables you to easily go to and place focus on a specific element by clicking that element's name in the containment path. When you hover the cursor over an element's name, it is underlined, as shown in Figure 5–1.

Figure 5–1 Selection Bar



- The same area that is used to display the containment path is also used to display informational and warning messages. After a warning message is displayed for a couple of seconds, the warning icon remains on the right end of the Message bar. Click the triangular warning icon, circled in Figure 5–2, to view the list of warning messages. The number shown in the icon indicates the number of warning messages that are available for viewing. Informational messages do not persist after being displayed and are not viewable in the Message Panel.

  The appearance of the document-looking icon, shown to the right of the warning icon in Figure 5–2, indicates that changes have been made to the document since the last save action was performed.

Figure 5–2 Message Bar Displaying a Warning Message

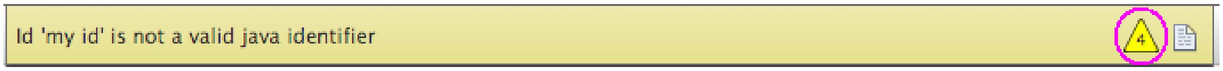
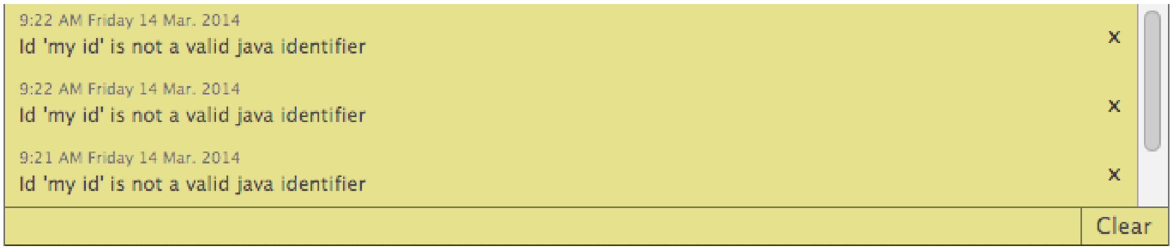

Id 'my id' is not a valid java identifier

Figure 5–3 shows a sample of the Message Panel with several messages listed. You can clear one or all of the messages in the list view.

Figure 5–3 Message Panel Displaying a List of Warning Messages



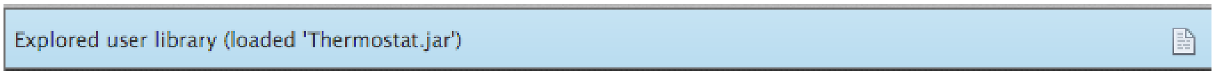
9:22 AM Friday 14 Mar. 2014
Id 'my id' is not a valid java identifier

9:22 AM Friday 14 Mar. 2014
Id 'my id' is not a valid java identifier

9:21 AM Friday 14 Mar. 2014
Id 'my id' is not a valid java identifier

Figure 5–4 illustrates an example of the Message Bar displaying an informational message.

Figure 5–4 Message Bar Displaying an Informational Message



Explored user library (loaded 'Thermostat.jar')

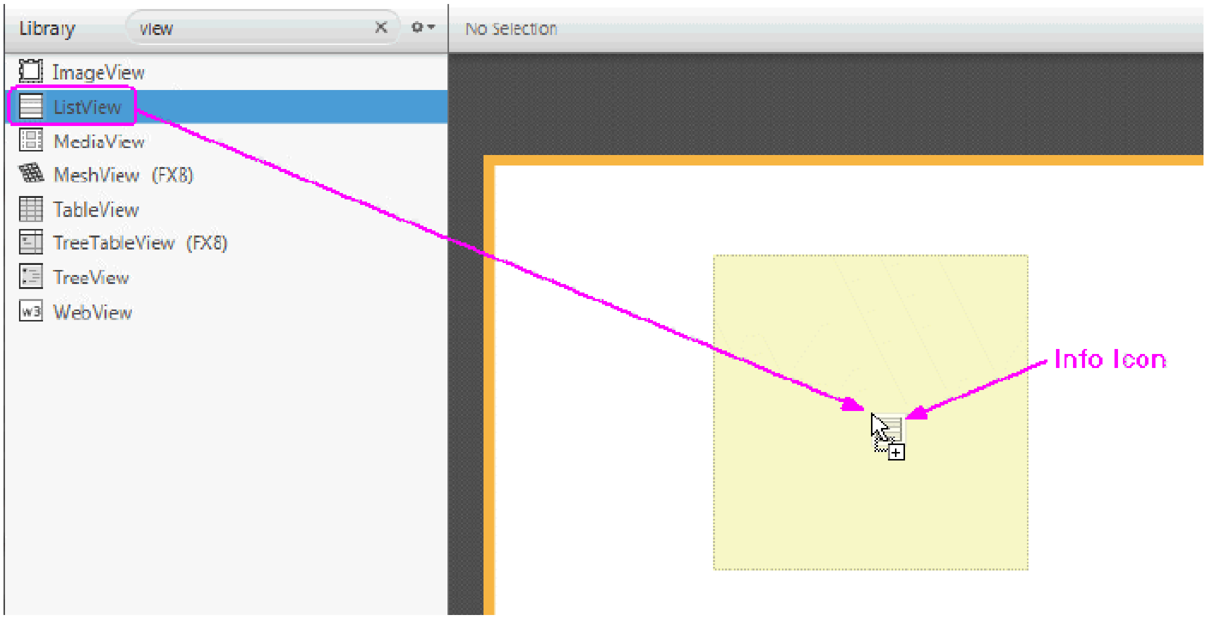
6

Working with the Content Panel

This chapter describes the Content panel of the JavaFX Scene Builder.

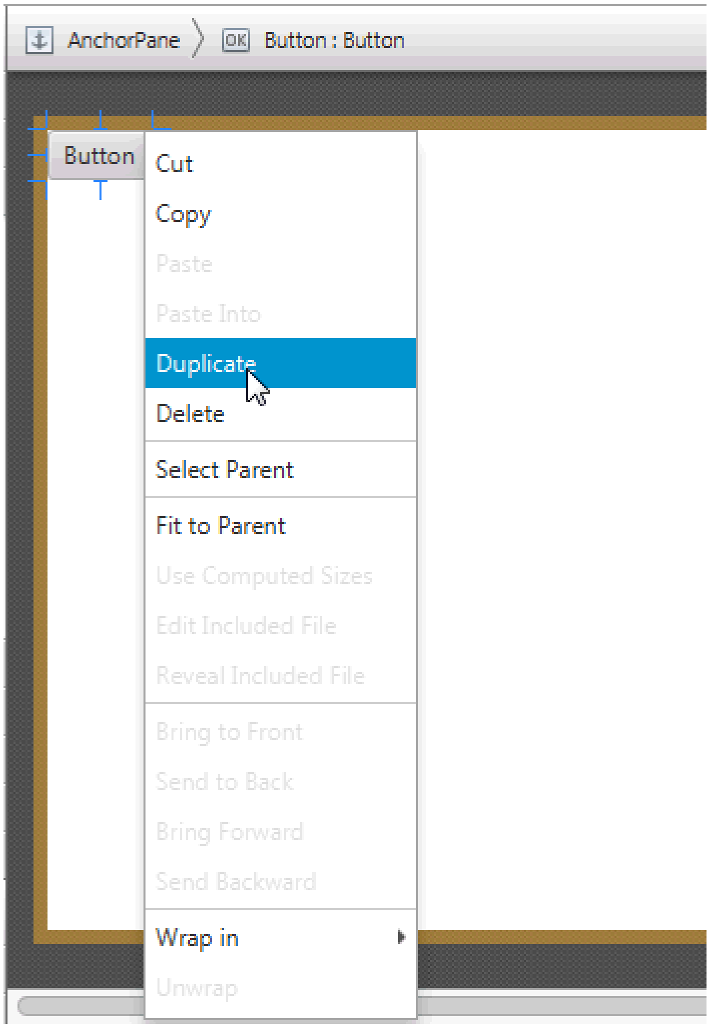
The Content panel is the rectangular area that occupies the center of the JavaFX Scene Builder window. It is empty by default when you create a new FXML file. It is your design canvas and gives you a view of what you are designing. It enables you to directly manipulate the graphical elements used in your FXML layout. One way to add a GUI element to your layout is to drag it from the Library panel and place it in a chosen location in the Content panel. As you drag an object onto the Content panel, the drop target area is highlighted, as shown in Figure 6–1. An info icon is also displayed to identify the GUI element being dragged.

Figure 6–1 Dropping a ListView Element into the Content Panel



The selected elements have special handles that enable you to scale or resize the elements. You move the selected element by using the mouse to drag it to a different location. Right-clicking anywhere in the Content panel displays a contextual menu of commands that can be used on the currently selected element, as shown in Figure 6–2.

Figure 6–2 Contextual Menu in Content Panel



Double-clicking on a GUI element in the Content panel places that element in an inline editing mode. If a GUI element with a Text property is selected, pressing Return also places the element in an inline editing mode.

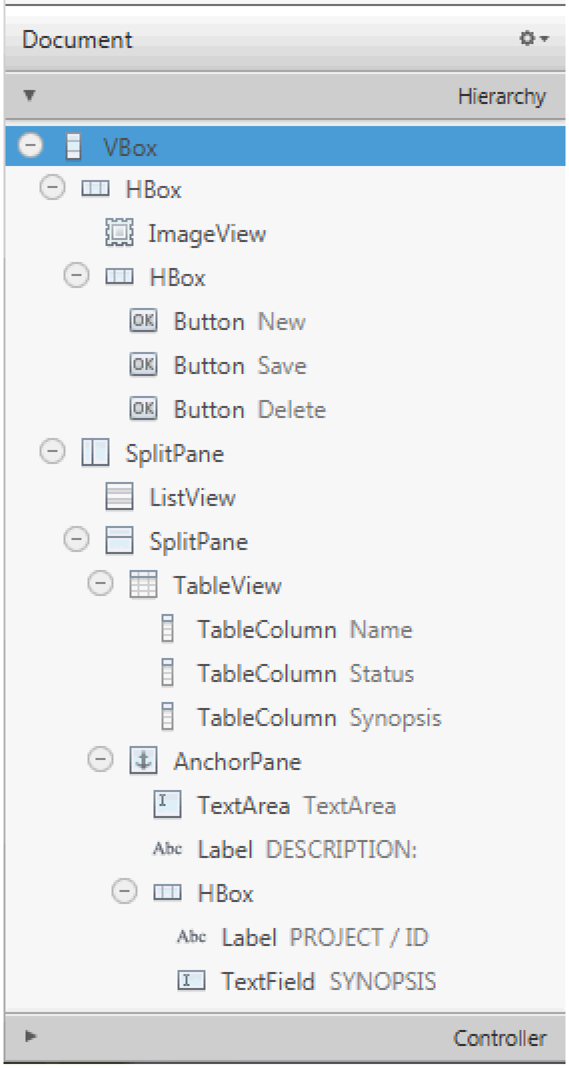
GUI elements in a FlowPane, TextFlow, TilePane, ToolBar, HBox, and VBox containers can be re-ordered. A grey line guides you with the placement or order location of the selected object.

7 Using the Document Panel

This chapter describes the Document panel of the JavaFX Scene Builder tool.

The Document panel, shown in Figure 7–1, is located at the lower left side of the tool's window. It is comprised of the Hierarchy and Controller sections. You can hide the Document panel by selecting **View** from the Main menu and then selecting **Hide Document**, or **Hide Left Side Panel** to hide it along with the Library panel.

Figure 7–1 Document Panel



Hierarchy Section

The Hierarchy section shows all the GUI elements that comprise your FXML layout, including those that are not visible in the Content panel. You can use the Hierarchy panel to focus on one specific GUI element, whether it is a parent node or a leaf node. After the element is isolated, you can use the Content panel or the Inspector panel to focus your work on that particular element and its properties. The path from the root of the FXML layout to a particular GUI element is displayed in the Path and Selection bar when a GUI element is isolated in the Hierarchy panel. The selected GUI object is highlighted in the Hierarchy panel and in the Content panel.

You can also add a new GUI element to your FXML layout by dragging it from the Library panel to the desired location in the Hierarchy panel. This is especially helpful when adding a Popup control, such as a Tooltip or ContextMenu. For example, to add a Tooltip to a Button element, drag the Tooltip control from the Library panel to the target Button element in the Hierarchy panel. The Tooltip becomes a child of that Button element.

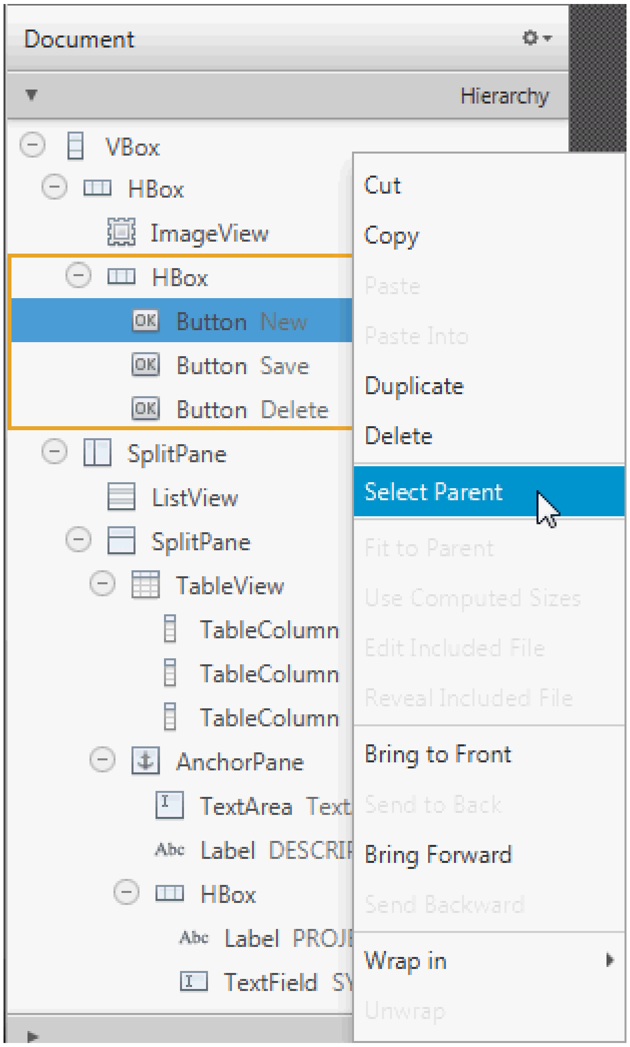
You can rearrange the position of a GUI element in your layout by dragging it in and out of containers within the Hierarchy panel. You can select multiple elements at once in the Hierarchy panel and drag and drop them into a single location.

The Hierarchy panel is also useful for adding a graphic element to a GUI element that has the graphic property, per its API. For example, you can add a graphic element to a Button element, but not to a GridPane element. Use the following steps to add a Circle graphic element to a Button element using the Hierarchy panel:

1. From the Library panel, drag a Circle element to the target Button layer in the Hierarchy panel.
2. Without releasing the mouse cursor, hover over the Button layer for a second and you will notice a new sub-layer is created for the Circle.
3. Release your mouse cursor and the Circle appears in the Content panel, with its default radius of size 100.
4. In the Layout section of the Inspector panel, reduce the circle's radius size to 6 so that it fits more appropriately in the Button.
   You can only have one graphic element added to a Button element at any one time. To add a new graphic, you must first delete the existing one. If the element is a container, then it can contain any number of graphical elements.

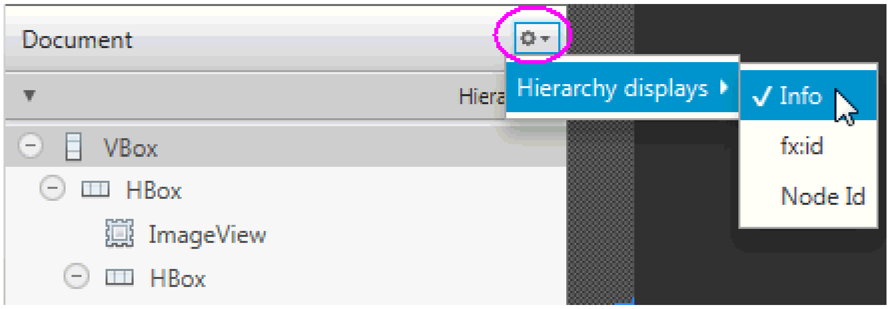
Right-clicking an element in the Content panel or a specific element layer in the Hierarchy panel displays a contextual menu of commands that you can use for the selected element, as shown in Figure 7–2. These commands are a subset of the commands that are available from the Menu Bar. You can click on the right side of the row for the GUI element listed in the Hierarchy panel to activate the inline property editor.

Figure 7–2 Hierarchy Panel with Drop-down List of Commands



When you click the drop-down arrow, which is circled on the top right corner in Figure 7–3, and select **Hierarchy displays**, you can select **Info**, **fx:id,** or **Node Id** to display next to each of the elements listed in the Hierarchy panel. The default display mode is **Info**, which displays any related information for the element, or the text property value if the element has a text property. The **fx:id** display mode shows the value of each element's fx:id property, which is used by the application controller class. When you select **Node Id**, the Hierarchy panel shows the same node names used in the Selection bar for each element.

Figure 7–3 Hierarchy Display Menu in Document Panel

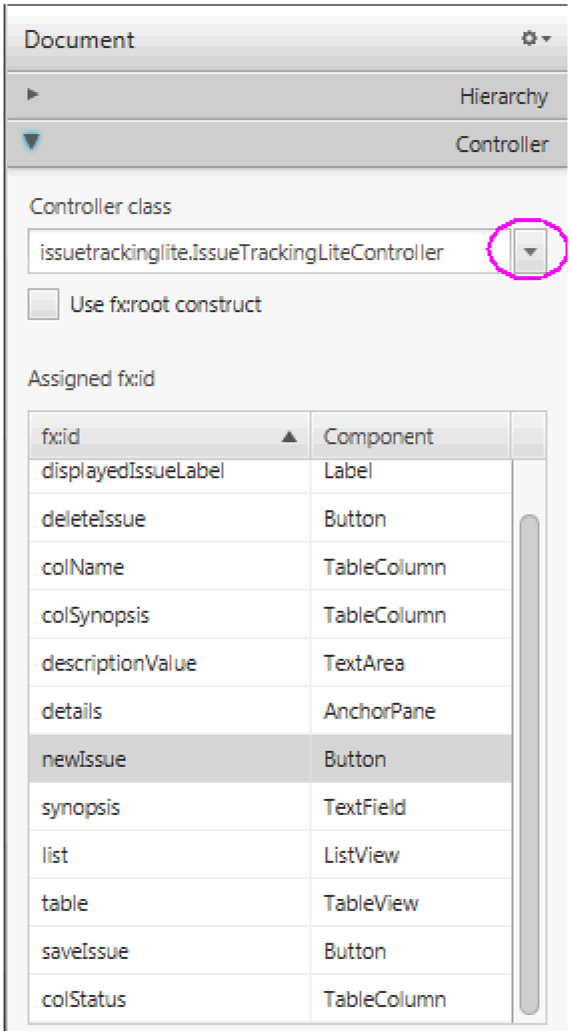


Controller Section

The Controller section, shown in Figure 7–4, enables you to manage the controller class you want to use with your FXML document. The controller class provides the logic to handle the behavior of the objects in your FXML layout. In the Controller class text field, you set the fully qualified Java class name of the controller class file you want to use. This file must be in the same folder, a sub-folder, or a parent folder in which the FXML file is also located. If the controller class file already exists in the same folder as the FXML file you are working with, you can select the controller class from the drop-down list by clicking the down arrow that is circled in Figure 7–4. If you are using NetBeans IDE with JavaFX Scene Builder and your FXML file is located in a NetBeans project, then the Controller class java file can be located in any java package inside your NetBeans project's `src` folder.

The list of all the fx:id values used in the FXML document is listed in a table view. Also included in the Controller section is the **fx:root construct** checkbox that creates a reference to a previously defined root element. To learn more about the `<fx:root>` element, refer to the JavaFX API document.

Figure 7–4 Controller Section of the Document Panel



Controller class

issuetrackinglite.IssueTrackingLiteController

□ Use fx:root construct

Assigned fx:id

fx:id ▲	Component
displayedIssueLabel	Label
deleteIssue	Button
colName	TableColumn
colSynopsis	TableColumn
descriptionValue	TextArea
details	AnchorPane
newIssue	Button
synopsis	TextField
list	ListView
table	TableView
saveIssue	Button
colStatus	TableColumn

8 Designing GUI with the Library Panel

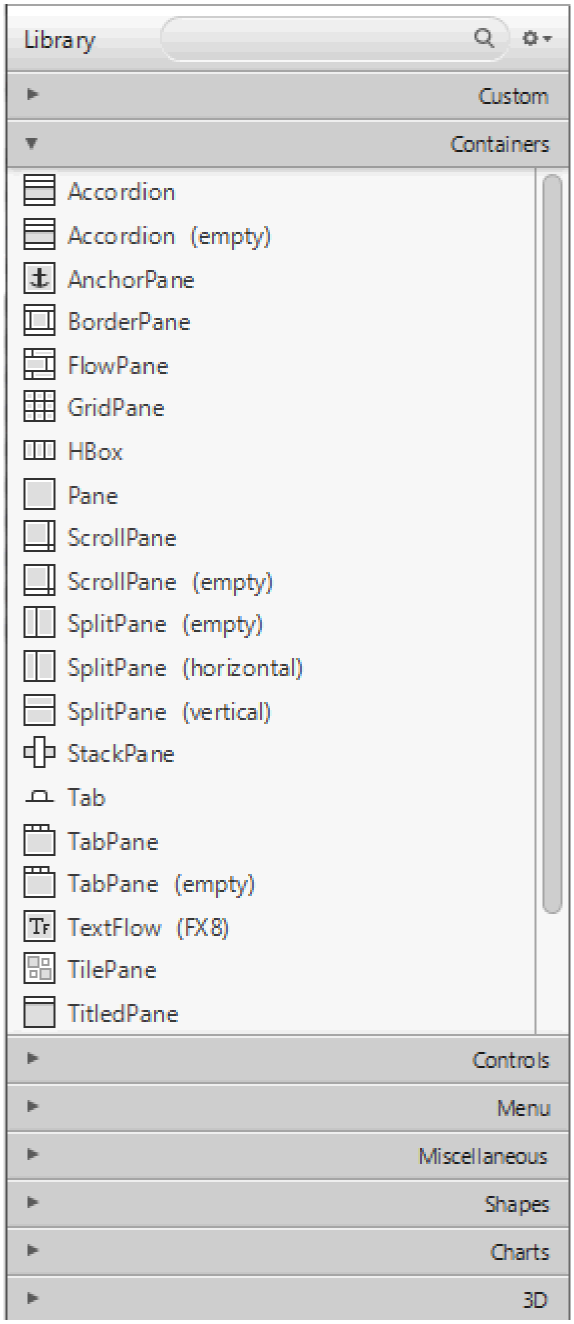
This chapter describes JavaFX Scene Builder's Library panel and the JavaFX GUI objects that are accessible from it. Information is provided about working with some of the container elements available. Also discussed in this chapter is how to import custom GUI objects into JavaFX Scene Builder.

The Library panel, shown in Figure 8–1, is located on the left side of the Main window. You can hide the Library panel by selecting **View** from the Main menu and then selecting **Hide Library** or **Hide Left Side Panel** to hide it along with the Document panel. The Library panel is comprised of collapsible sections for Containers, Controls, Menus, Shapes, Charts, 3D, and Miscellaneous elements that are available for your use to build your FXML layout. These GUI elements are the same JavaFX GUI elements that are delivered with the Java SE release. To locate the GUI components, such as TextFlow, SwingNode, and 3D objects, that were introduced with the JDK 8 release, type `FX8` in the Library panel's Search text field.

There is also a Custom section, which lists the custom GUI components that have been imported into the Library. Custom type elements are those GUI components that were not delivered with the JavaFX API libraries in JDK 8. See Managing Custom GUI Elements for more information on how to import your own or third-party custom GUI components.

Support for 3D GUI elements is limited. You can open FXML files that contain 3D objects, but you can only edit some of their properties via the Inspector panel and you can not directly manipulate the objects in the Content panel. Pre-defined 3D shapes (Box, Sphere, and Cylinder) that are available in the JavaFX API are provided in the Shapes section of the Library panel. In the 3D section are objects related to lights and cameras. These are available in the Library so that you can edit their properties and add their fx:id so that they can be referenced from your controller source code.

Figure 8–1 Library Panel



When you open a new FXML layout in Scene Builder, it is an empty file. You start building your layout by selecting the top root container from the Containers section and adding it to the Content or Hierarchy panel. You can then add into that root container as many containers or GUI elements that are needed for your FXML layout.

To add one of the GUI elements from the Library panel, do one of the following:

- Select an item from the list on the Library panel and drag it onto the Content panel or the Hierarchy panel.
- Select an item from the Library panel and press the Enter key.

- Double-click a selected item in the Library panel and it gets added to the Content panel.
- Click the **Insert** command on the Menu bar and select one of the available GUI elements from the drop-down list. If the layout is currently empty, the element is added as the root element. If a root element already exists and nothing is currently selected in the Hierarchy or Content panel, the element is added inside the root element. Otherwise, it is added next to the currently selected element.

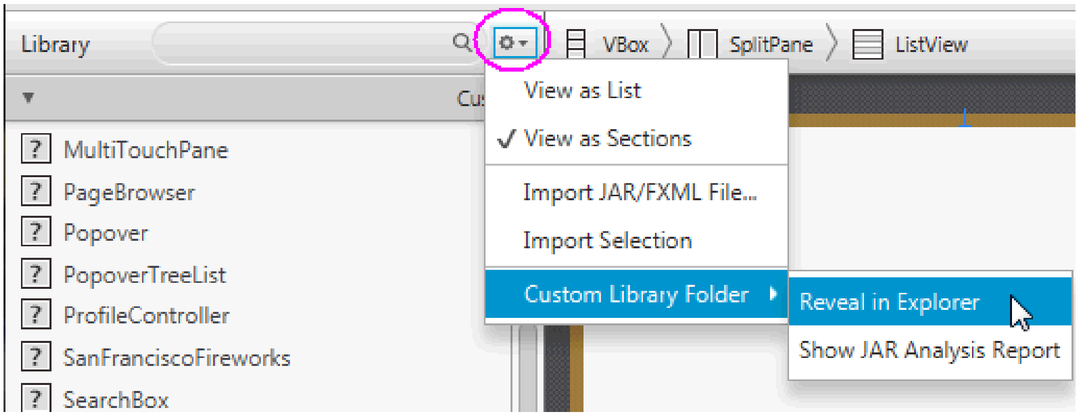
Use the Search text field, located at the top of the panel, to filter the list of GUI elements available in the Library. For example, typing `button` in the Search field displays all the GUI elements with the button in its name.

To the right of the Library panel's Search text field is a menu button which when clicked displays a menu of commands, as shown in Figure 8–2. The commands help you manage how the Library contents are to be displayed: as a long list, or as groups of elements listed in separate sections, which is the default. Also available are commands that allow you to import custom GUI components into the Library. See Managing Custom GUI Elements for more information.

Selecting the **Custom Library Folder** menu item displays a sub-menu that includes Reveal in *native file manager* and the Show JAR Analysis Report commands. Selecting the **Reveal in *native file manager*** command (where *native file manager* is **Finder** for the Mac OS platform, **Explorer** for the Windows platform, and **Desktop** for the Linux platform) opens your system's file manager at the location of the folder containing the custom component JAR and/or FXML files. The command is disabled if the Custom Library folder is empty.

Selecting the **Show JAR Analysis Report** command allows you to view an analysis report for the JAR file(s) located in the Custom Library Folder. This report will list all of the Java classes which were inspected and determined as custom components that are unsuitable to be added to the Library. Also provided is an indication of the failed criteria, such as the fact that the class is not a JavaFX Node, or Scene Builder failed to instantiate the class. In the latter case, the Java exception is also listed.

Figure 8–2 Library Panel's Sub-menu of Commands



The following sections discuss some of the container elements that are available in the Library panel and how to maximize their usage in the JavaFX Scene Builder tool. Also described is how to manage custom types. To learn more about all the other GUI elements offered with the JavaFX SDK platform, see the JavaFX API documentation.

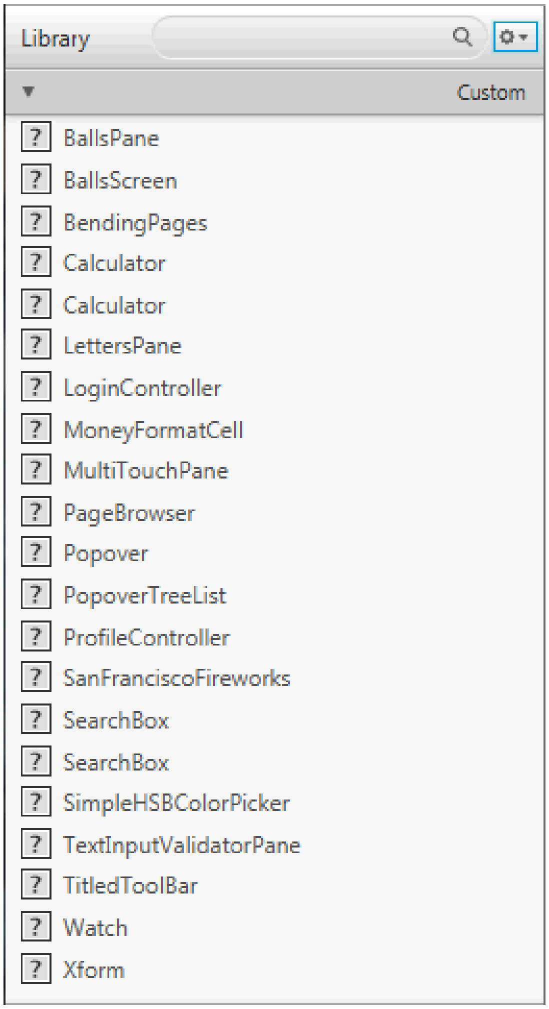
- Managing Custom GUI Elements

- Using a Grid Pane Container
- Working with a Border Pane Container
- Using a Tab Pane Container
- Working with Menu Controls

Managing Custom GUI Elements

The first section displayed in the Library panel is the Custom section, as shown in Figure 8–3,. It lists all the custom GUI elements that are not part of the JavaFX 8 API libraries, but are available for your use in your GUI layout. A custom GUI element is a Java class that has been created by the user or by a third party, and added to the Library of available components you can use in your FXML layout.

Figure 8–3 Custom Section of the Library Panel



Adding Custom Components to the Library

You can import customized GUI components by using the Library panel's Import command. You can add any simple or complex graphical object from the Hierarchy or Content panel into the Library, and make it a reusable component available from the

Library panel. You can also import a graphical component from an FXML document, or import a JAR file and select a subset of the custom components it contains, to make them re-usable components permanently available in the Library.

- **To import custom GUI components from a JAR or FXML file:**

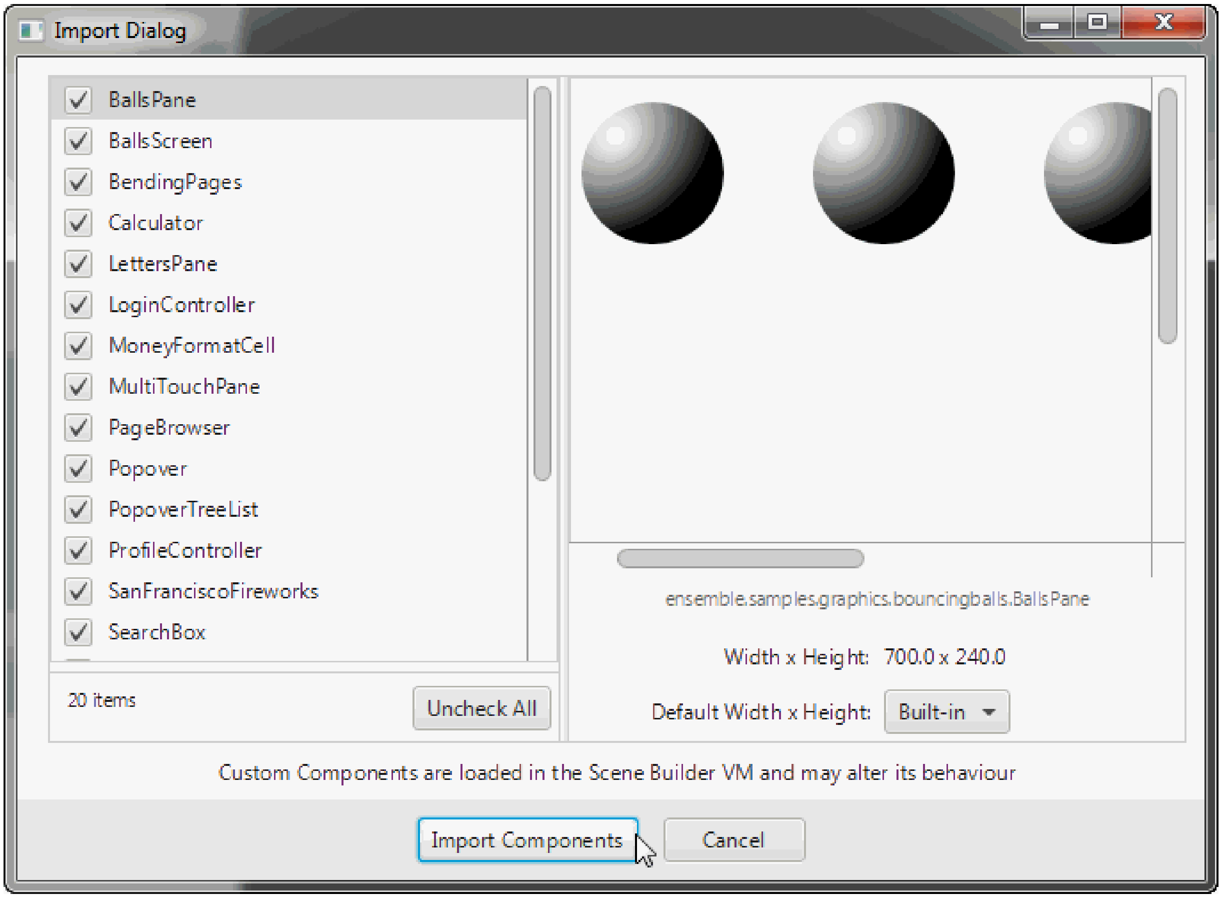
  1. Select **Import JAR/FXML file** command from the Library panel's menu, or drag the JAR or FXML file directly from your system's native file manager (Explorer or Finder) and drop it into the Library panel

  2. In the Open dialog window, navigate to the location of the JAR or FXML file that you want to import.
     The Import Dialog, similar to what is shown in Figure 8–4, is displayed. The JAR file's contents are inspected and all the Java classes that are determined as being suitable custom components are displayed in the dialog window. The FXML file's contents are parsed to make sure that the component being added is valid and self-contained.

  3. From the Import dialog window, select or unselect items from the list of items that you are able to import.

  4. Click **Import Components**.
     Imported items are added to the Custom section of the Library panel. They can be used immediately and they persist in the Library even after Scene Builder is restarted

Figure 8–4 Dialog Window for Importing JAR/FXML File



☑ BallsPane
☑ BallsScreen
☑ BendingPages
☑ Calculator
☑ LettersPane
☑ LoginController
☑ MoneyFormatCell
☑ MultiTouchPane
☑ PageBrowser
☑ Popover
☑ PopoverTreeList
☑ ProfileController
☑ SanFranciscoFireworks
☑ SearchBox

20 items

ensemble.samples.graphics.bouncingballs.BallsPane

Width x Height: 700.0 x 240.0

Default Width x Height: Built-in

Custom Components are loaded in the Scene Builder VM and may alter its behaviour

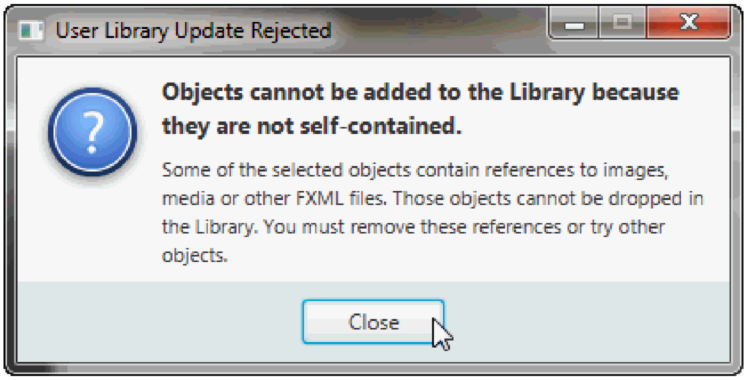
- **To add custom GUI objects from your current layout to the Library panel:**

1. Using the Hierarchy or Content panel, select the self-contained component in your current layout.
2. From the Library panel's menu, select the **Import Selection** command. Alternatively, drag the selected component from the Content or Hierarchy panel and drop it in the Library panel.

   The selected item is inspected and if there are issues encountered, an error dialog similar to Figure 8–5 is displayed. If no errors are encountered, the custom component is added to the Library and a message is displayed in the Message bar. The name of the FXML file created for the imported selection is based on the name of the root element. For example, if in your selection the root element is of the `Button` class, then the file created when you import the element for the first time is `Button_1.fxml`.

   You can also import multiple selections made in the Content or Hierarchy panel. For example, you can select a `Button` and a `CheckBox` element in the Content panel, and drag and drop both elements into the Library. Two rows for the imported elements appear in the Custom section of the Library panel: one named `Button_1` and the other named `CheckBox_1`. Both are stored into corresponding FXML files, `Button_1.fxml` and `CheckBox_1.fxml`, respectively and are created in the Custom Library folder.

Figure 8–5 Sample Error Message When Importing GUI Objects into the Library



User Library Update Rejected

Objects cannot be added to the Library because they are not self-contained.

Some of the selected objects contain references to images, media or other FXML files. Those objects cannot be dropped in the Library. You must remove these references or try other objects.

When you import the same JAR or FXML file at another time, the newly imported version of that file replaces the previously imported version. If you import the same self-contained GUI element from the Content or Hierarchy panel at another time, then the FXML file created is named with a version number. For example, if a selection with a Button element as its root has been previously imported, then when it is imported a second time, the new FXML file created for the custom element is named `Button_2.fxml`.

If you decide you no longer need the custom GUI components, you can directly remove the JAR and FXML files from your Custom Library folder, or delete the whole folder. This can be safely done even as the Scene Builder tool is running. The Scene Builder tool periodically checks the content of the Custom Library folder and will update the content of the Library accordingly.

To work with the Custom Library folder, open the Library panel's menu, click **Custom Library Folder**, and select **Reveal in *native file manager***, where *native file manager* is **Finder** for the Mac OS platform, **Explorer** for the Windows platform and **Desktop** for the Linux platform.

Using a Grid Pane Container

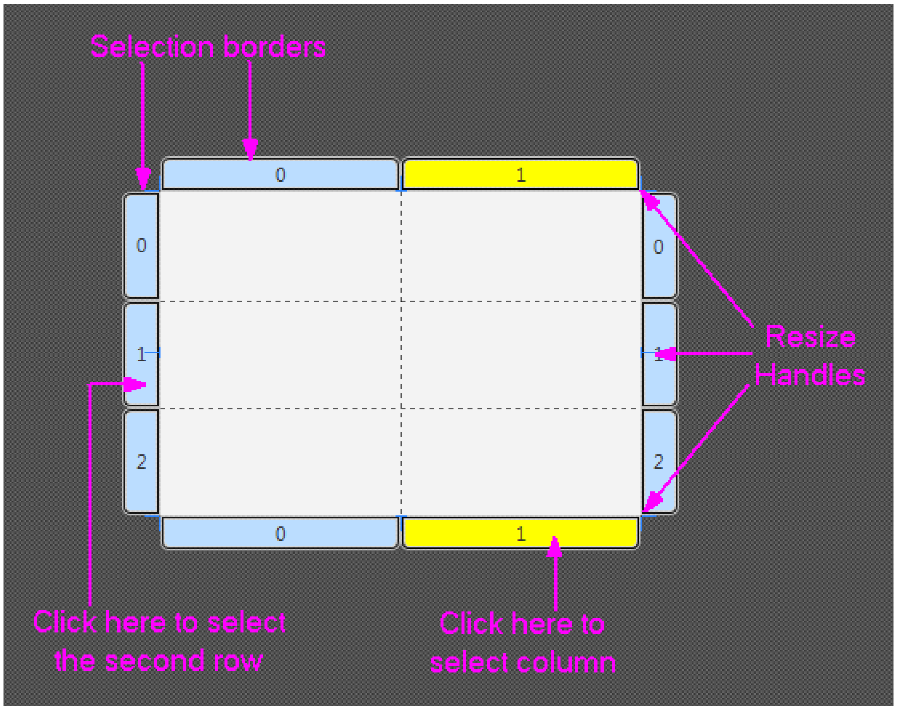
The Grid Pane container enables you to create a flexible layout of GUI elements that are organized in rows and columns. Use the Grid Pane in your layout by selecting **Grid Pane** from the Container section of the Library panel and dragging it to the Content or Hierarchy panel. You can then add other GUI elements within the Grid Pane container, and elements are automatically arranged based on the padding, gap, and other properties.

You can also add a Grid Pane container element so that it contains the GUI elements that you have already added to your layout. Use the following steps:

1. Select **Edit** and then **Select All** from the Main menu to select all of the elements in your layout.
2. Add the Grid Pane container by selecting **Arrange** from the Main menu, **Wrap in**, and then **Grid Pane** from the list of containers.

An empty and unselected grid pane is not visible in the Content panel. To make its borders visible, click the corresponding layer in the Hierarchy panel for that Grid Pane element. To work with a specific grid pane, click directly within its perimeter, but not on any of the elements that it contains. When selected, a Grid Pane is surrounded by a light blue colored border tabs and each corresponding index number is displayed, as shown in Figure 8–6. To select either a single column or row, click within the selection border tab. The tabs of a selected column or row acquires a yellow background, as shown by column 1 in Figure 8–6. You can also select multiple rows or multiple columns simultaneously so that you can edit the values for those selected items at once. You can not, however, select a combination of rows or columns at the same time.

Figure 8–6 Grid Pane in Selected Mode



To see the submenu of actions that you can take on a Grid Pane, do one of the following:

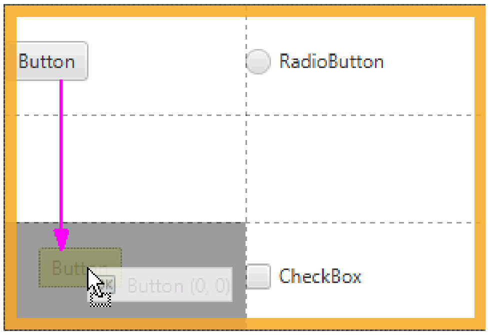
- Right-click anywhere over the grid pane, select a command from the contextual menu.
- From the Main menu, select **Modify** and then **Grid Pane**.

You can move a row up or down, and move a column left or right. You can also delete or add a row or column, and increase or decrease the span for a row or column.

The Layout section of the Inspector panel also enables you to modify the currently selected Grid Pane. The properties related to the selected row or column are displayed and can be managed in the GridPane Row or GridPane Column sections.

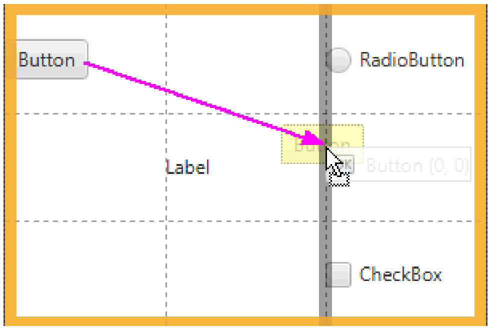
You can move an element from one grid cell to another existing grid cell. In the Content panel, select the element that you want to relocate, and with the mouse button pressed, drag the element to the target grid cell, as shown in Figure 8–7. Notice that the target grid cell changes to have a gray background as soon as you hover your cursor over it. When you release the mouse button, the element is moved to the target grid cell location in the Grid Pane.

Figure 8–7 Moving an Element to Another Grid Cell



To move an element to a new row or column that has not yet been created, select the element in the Content panel and drag it to the divider line that is between the rows or columns where you want the new row or column to be inserted. As you hover the mouse over the cell divider, the cell divider line becomes thicker and gray. When you release the cursor, the new row or column is created, and the element is dropped in the target grid cell. Figure 8–8 shows a Button element being dragged to a new grid cell on the second row of a new column that is about to be inserted.

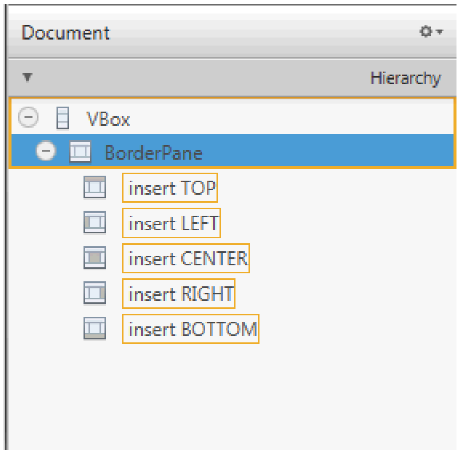
Figure 8–8 Moving an Element to a New Row or Column



Working with a Border Pane Container

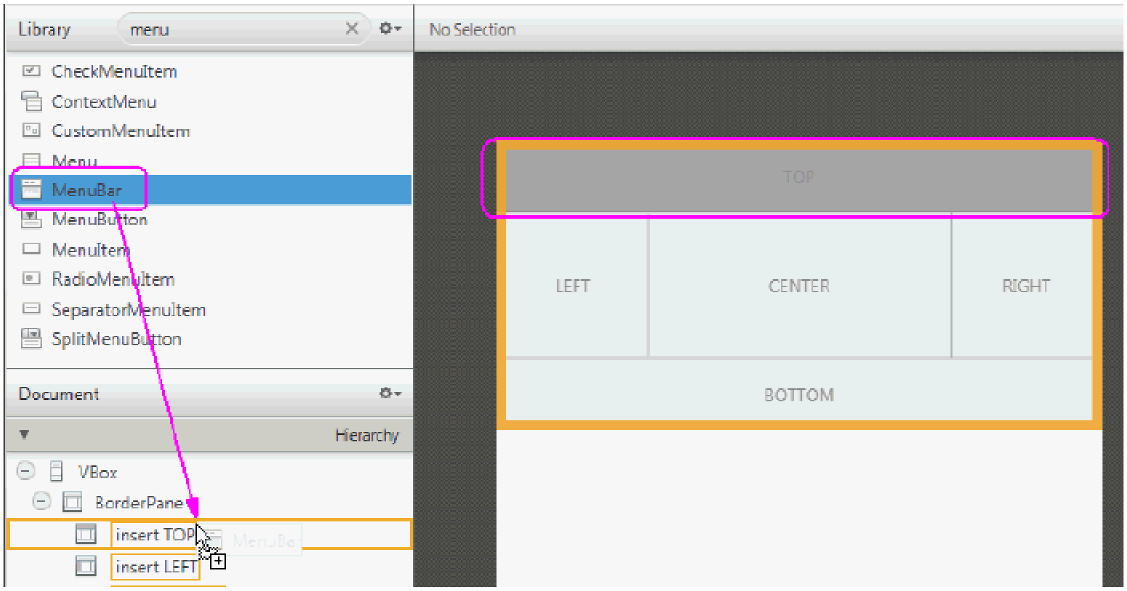
The Border Pane container enables you to layout GUI elements in the top, right, left, bottom, and center positions of the container. Use the Border Pane in your layout by selecting the **BorderPane** element from the Container section of the Library panel and dragging it to the Content panel. Figure 8–9 shows the Hierarchy panel after the BorderPane element is added.

Figure 8–9 Hierarchy Panel After BorderPane Container Is Added



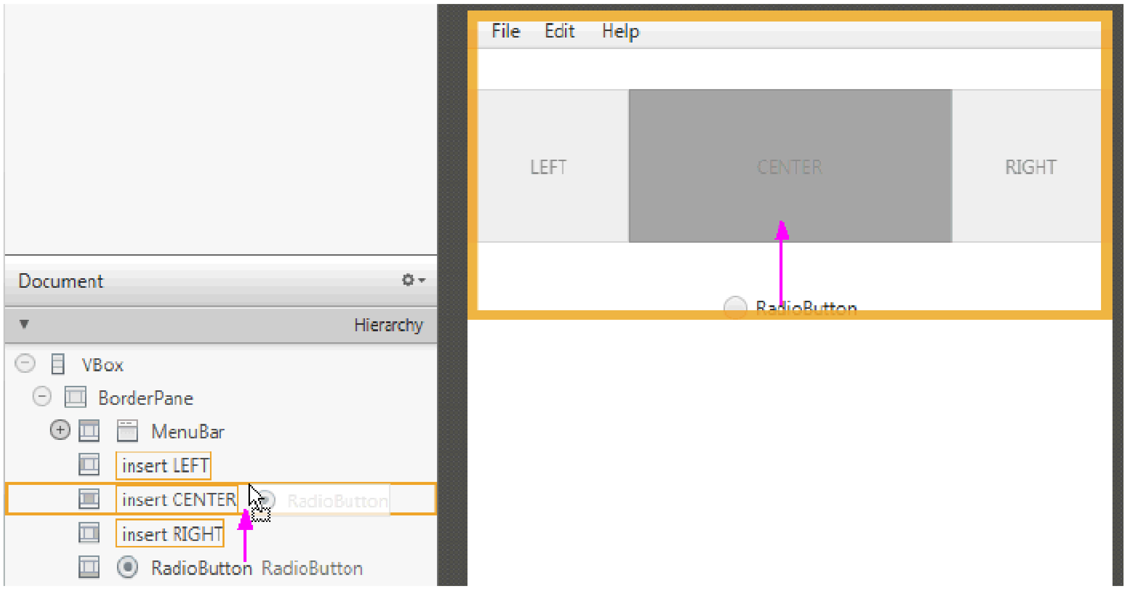
You can add a GUI element to the Border Pane container by dragging the element from the Library panel onto the specific position in the Border Pane. You can also drag an element from the Library panel to the Hierarchy panel, as shown in Figure 8–10. Notice that when the element is dragged over a target layer in the Hierarchy panel, the layer’s corresponding position in the Border Pane changes to a darker shade of color, as shown on the right side of Figure 8–10.

Figure 8–10 Add Menu Bar Element to the BorderPane



You can move an element from one position to another in the Border Pane. In the Content panel or the Hierarchy panel, select the element that you want to relocate and drag the element to the target position, as shown in Figure 8–11. Notice that the background color of the target position gets darker as soon as you hover your cursor over it. When you release the mouse button, the element is moved to the target position in the Border Pane.

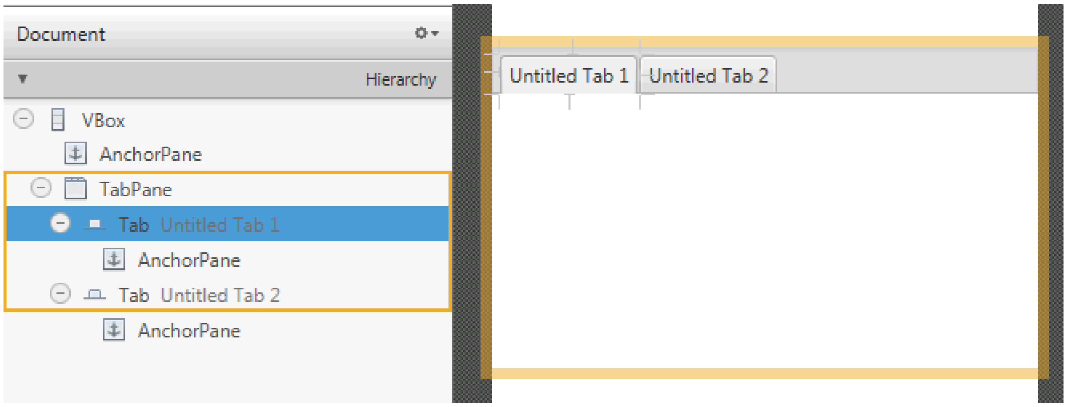
Figure 8–11 Moving an Element to Another Position



Using a Tab Pane Container

By default, the Tab Pane comes with two Tab elements, each of which contain an Anchor Pane element, as shown in Figure 8–12.

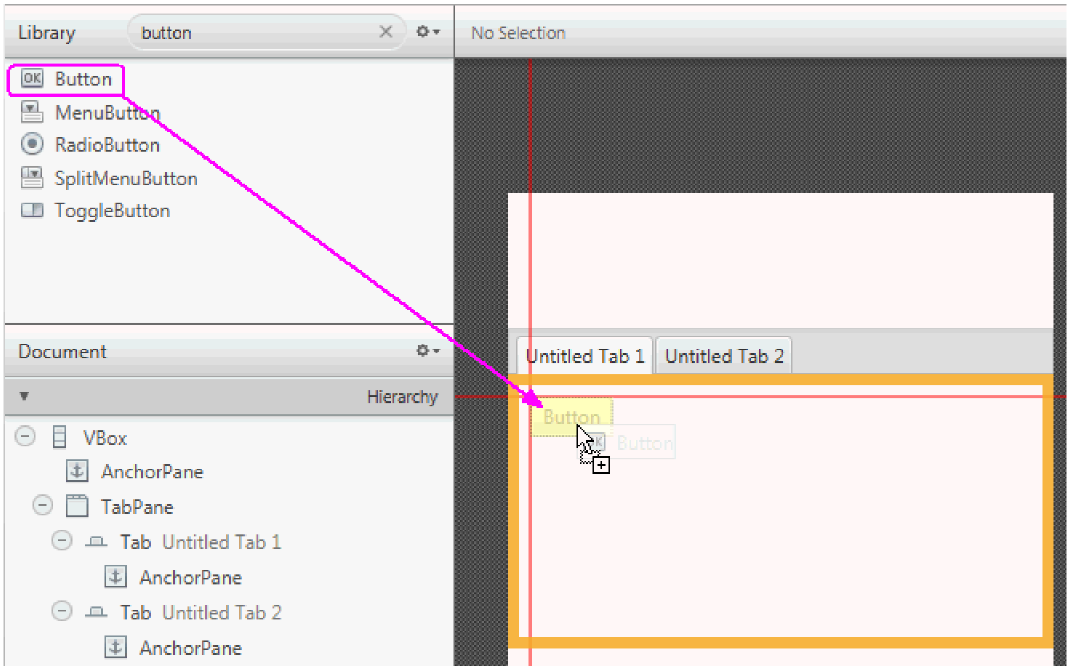
Figure 8–12 Tab Pane Element Added in the Hierarchy Panel and Content Panel



You can change a Tab element's default title by double-clicking the tab and directly editing its Text property in the Content panel. To add another Tab element to a Tab Pane container, drag and drop a Tab element from the Library panel to the Tab Pane in the Content panel or Hierarchy panel. The new Tab element is added to the right of any existing Tab elements. To change the order of the Tab elements within the Tab Pane, use the Hierarchy panel. Drag and drop the Tab element's row in the Hierarchy panel to the target location within the Tab Pane container. You can also use the Content panel to drag its tab to the target location within the Tab Pane container.

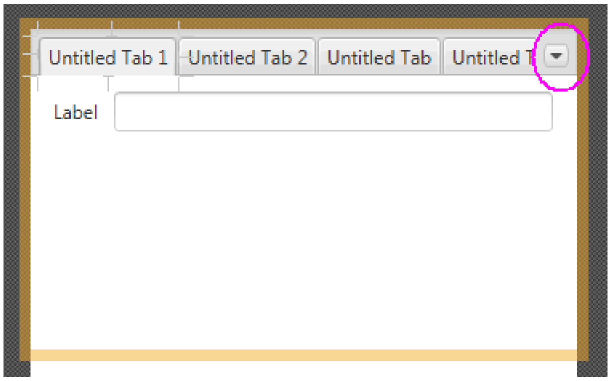
To add GUI elements to a Tab element's Anchor Pane, click the title of the target Tab element to select it. Drag and drop the GUI element that you want to add from the Library panel to the Tab element, as shown in Figure 8–13.

Figure 8–13 Add GUI Elements from Library Panel to the Tab Element in Content Panel



Depending on the length of each Tab element's title and the number of Tab elements in your Tab Pane container, there comes a point when some Tabs become hidden. When this occurs, a drop-down menu is automatically added to the Tab Pane container's menu bar, as shown in Figure 8–14. This drop-down menu enables the tab of choice to be selected. This menu is not active in the Content panel, but is functional when you view the layout in the Preview window. If the Tab Pane container is resized so that all the Tab elements are fully visible, the drop-down menu is automatically removed from the Tab Pane's menu bar.

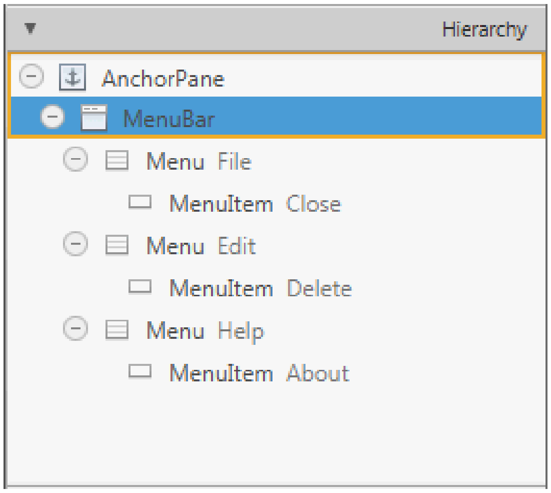
Figure 8–14 Tab Pane with Multiple Tab Elements and a Drop-Down Menu



Working with Menu Controls

The Controls section of the Library panel contains three menu-related control elements: MenuBar, Menu Button, and Split Menu Button. When you drag and drop a MenuBar from the Library panel onto the Content panel, it comes populated with Menu elements that are labeled File, Edit, and Help. By default, each of these Menu elements contains one Menu Item element, as shown in Figure 8–15.

Figure 8–15 Menu Bar and Its Contents Displayed in the Hierarchy Panel



The Menu and Menu Item elements do not have graphical representations that are visible in the Content panel, so you can only use the Hierarchy panel to select and manage these elements. You can change the order of how the Menu elements are displayed in the Menu bar by selecting the Menu element in the Hierarchy panel and dragging it to a new location within the Menu where you want it to appear. You can directly modify the text property of the Menu or Menu Item element by selecting it and clicking on the text area, which puts you into edit mode. You can also use the Properties section of the Inspector panel to modify the Text field.

You can add other menu-related items, such as another Menu Item or a Separator Menu Item, by dragging the element from the Library panel to the desired Menu location in the Hierarchy panel. To create a second-level menu, drop a Menu element within an already existing Menu element in the Hierarchy panel.

Use the Preview menu to see your Menu bar in action. From the Preview window, you can click a Menu item and any of its submenus.

9

Defining Properties in the Inspector Panel

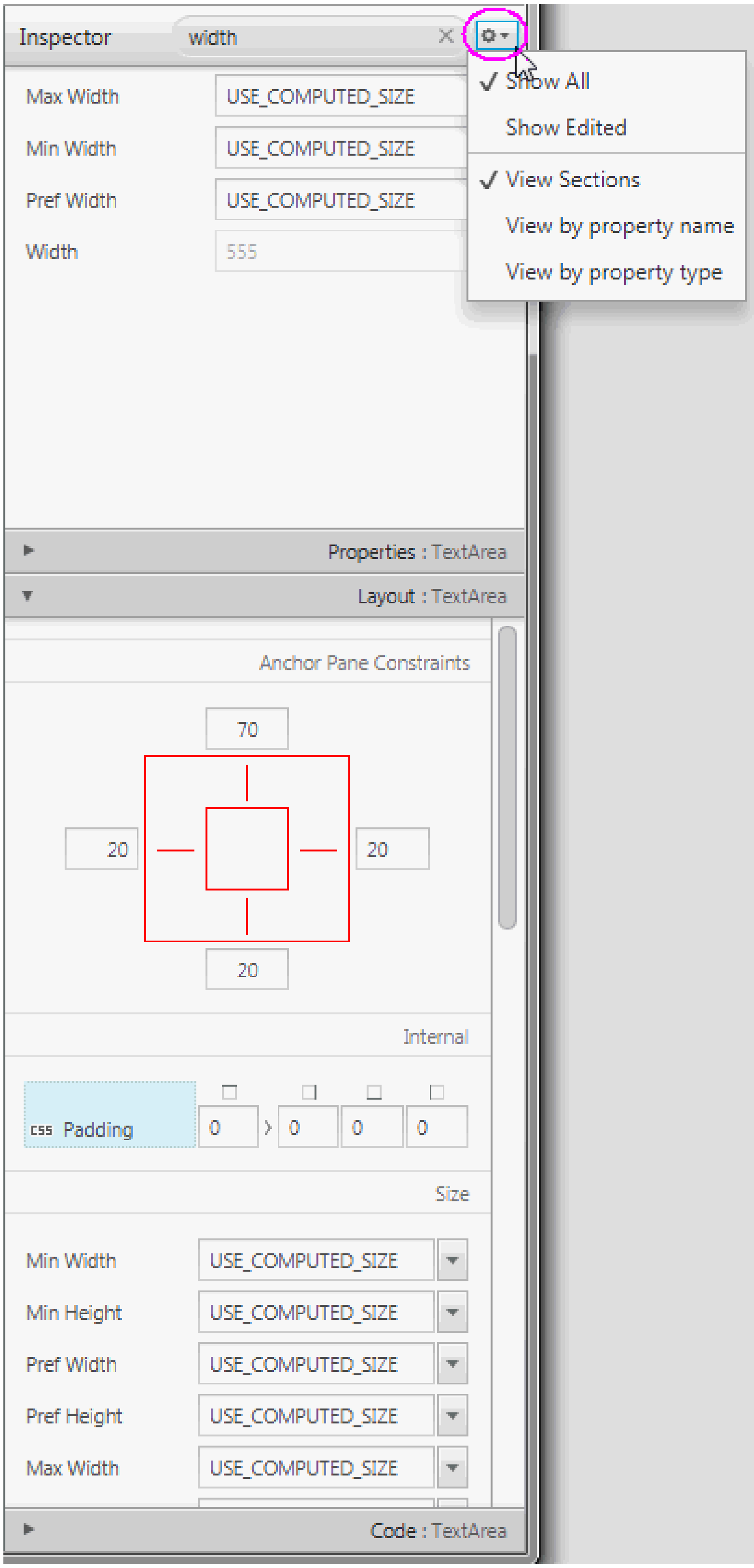
This chapter describes the JavaFX Scene Builder tool's Inspector panel, including its Properties, Layout, and Code sections.

On the right side of the Content panel is the Inspector panel, as shown in Figure 9–1 with the Layout section expanded and the Inspector menu displayed. The Inspector panel is also referred to as the Right Side panel. It contains an Accordion container with multiple sections. The top section contains a Search text field, which you can use to quickly locate the properties you would like to edit. The search results are displayed in the top section, as shown in Figure 9–1. You can directly modify the properties by using the Search Results section. You can display the Inspector menu by clicking the drop-down arrow at the top right corner, which is circled in Figure 9–1. By default, the Show All and View Sections modes are chosen. selecting the **Show Edited** mode displays only the properties that appears explicitly in the FXML file. For example, only the Text, LayoutX, and LayoutY properties are displayed in the Inspector panel when a Button element is dropped from the Library panel. selecting the **View by property name** or the **View by property type** modes display all the properties in a single tabbed section, ordered either by name or type.

The Properties, Layout, and Code sections give you access to properties for the selected GUI element. Each displayed property name shown in the sections is a representation of the corresponding given property name in the JavaFX API. For example, minWidth property in the JavaFX API is displayed in the Inspector panel as Min Width. When you click on the property name, a browser window is opened and displays the corresponding JavaFX API name for the property and its description.

You can hide the Inspector panel by selecting **View** from the Main menu and then selecting **Hide Right Side Panel**. This feature enables you to have more room for the Content panel. Select **View** and then **Show Right Side Panel** to make the panel visible again.

Figure 9–1 Inspector Panel with the Search Results Displayed and Layout Section Opened



Inspector width

Max Width USE_COMPUTED_SIZE

Min Width USE_COMPUTED_SIZE

Pref Width USE_COMPUTED_SIZE

Width 555

Anchor Pane Constraints

70

20 20

20

Internal

CSS Padding 0 0 0 0

Size

Min Width USE_COMPUTED_SIZE

Min Height USE_COMPUTED_SIZE

Pref Width USE_COMPUTED_SIZE

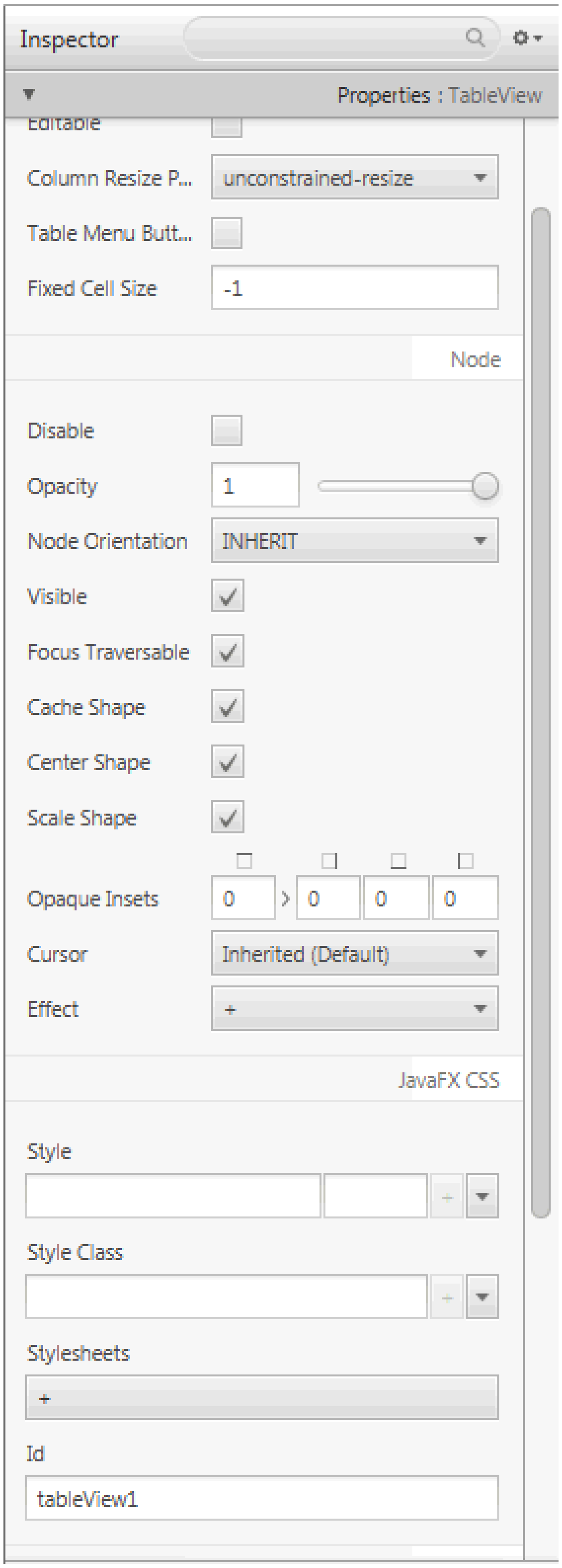
Pref Height USE_COMPUTED_SIZE

Max Width USE_COMPUTED_SIZE

Properties Section

The Properties section of the Inspector panel, shown in Figure 9–2, enables you to define the style of the selected element in the Hierarchy panel or the Content panel.

Figure 9–2 Properties Section of the Inspector Panel



Inspector

Properties : TableView

Editable □

Column Resize P... unconstrained-resize

Table Menu Butt... □

Fixed Cell Size -1

Node

Disable □

Opacity 1

Node Orientation INHERIT

Visible ☑

Focus Traversable ☑

Cache Shape ☑

Center Shape ☑

Scale Shape ☑

Opaque Insets 0 0 0 0

Cursor Inherited (Default)

Effect +

JavaFX CSS

Style

Style Class

Stylesheets +

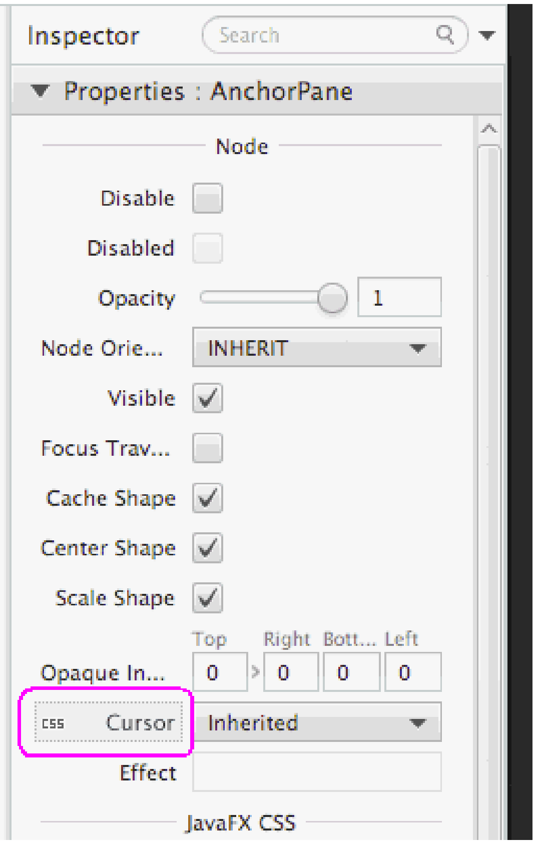
Id tableView1

When the fx:id value is assigned for the selected GUI element via the Code Section, that same value is also assigned to the Id field in the CSS subsection of the Properties section only if that Id field is `null`.

The Style, Style Class, and Stylesheets text areas in the JavaFX CSS subsection enable you to specify the style rule and style sheet to use for a particular GUI component and all of its children components. You can add, delete, and put in order the style sheets that will be applied. See the Styling with CSS and the CSS Analyzer chapter in this document for additional information.

A property value can be overridden by a CSS styling defined in a CSS file or in the `Style` property. A CSS icon is added to the left side of the affected property whose property value is overridden by a CSS file definition, as shown in Figure 9–3. You can navigate to the CSS file or Style property by clicking the property field.

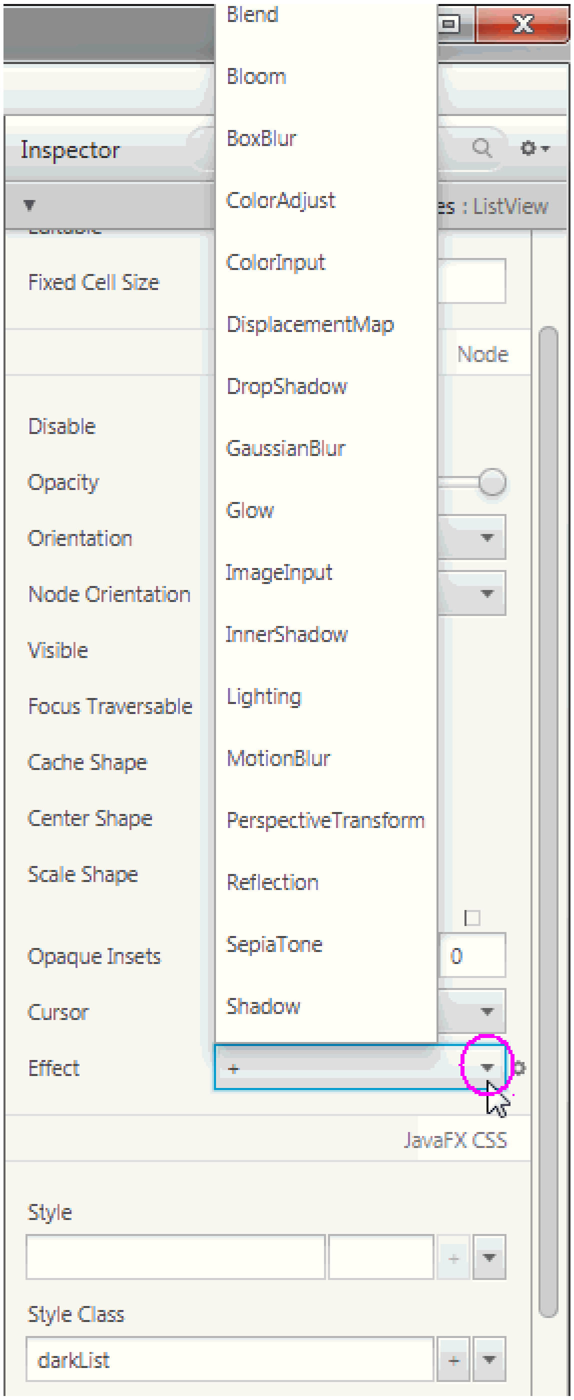
Figure 9–3 CSS Icon for Overridden Property Value



Disable □
Disabled □
Opacity 1
Node Orie... INHERIT
Visible ☑
Focus Trav... □
Cache Shape ☑
Center Shape ☑
Scale Shape ☑
Opaque In... Top 0 Right 0 Bott... 0 Left 0
CSS Cursor Inherited
Effect

You can go from one property to another by pressing the Tab key. A new value for a property is saved when you press **Enter** or when you move to another property field. You can manipulate the look of the selected GUI element by directly changing the value of its properties that are available and editable. For example, you can define the effects of a selected list view element by clicking the **down arrow** button to display the menu of available effects, as shown in Figure 9–4. From the list of effects available, you can select one or more effects to apply to the selected GUI element.

Figure 9–4 Effects Menu in the Properties Section of the Inspector Panel



The remaining properties displayed on the Properties section of the Inspector panel depend on the type of GUI element that is selected in either the Content panel or in the Hierarchy panel. For example, if you select a radio button, toggle button, or radio menu item in the Content panel, the Properties section shows the Toggle Group property. You can use this property to assign the same `ToggleGroup id` for each toggle control that you want to add to a toggle group.

When an invalid value is entered for a property, a dialog window appears and prompts you to select one of these commands: **Fix the Value** or **Revert to Previous Value**. The focus returns to the text field with the invalid property value if you select the **Fix the Value** command.

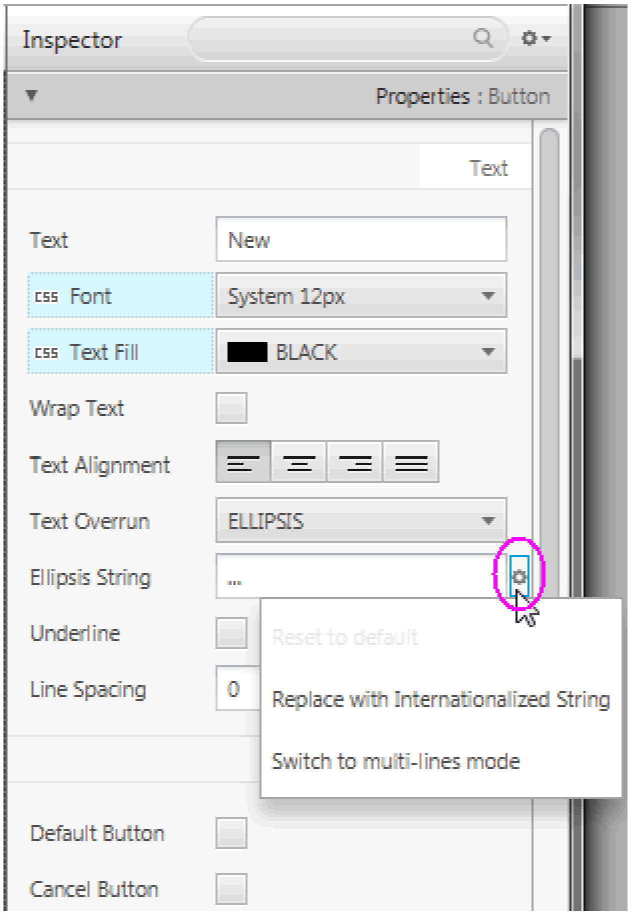
If a property is allowed to have a value list assigned to it, a single empty item is initially shown by default. To add an item, click the + button and enter or select the value. Click the down arrow to display the menu that allows you to remove, move up or move down an item for the value list.

When multiple objects of the same type (for example, two button elements) are selected in the Content panel or the Hierarchy panel, and if any of the properties have a different value, the minus sign (-) is displayed in the text field for that property. If a new value is entered, then the new value is set for all the selected objects.

If multiple objects of different types are selected in the Content or Hierarchy panel, the properties which are common to all objects are displayed.

There are some properties that have a cog menu that appears on the right side of the property field when the mouse hovers over the property value editor, as shown in Figure 9–5. The cog menu provides specific actions that can be used for that editor. For example, you can reset the current property value to the default value in the JavaFX API by selecting **Reset to default** command. Figure 9–5 shows actions available for an Ellipsis String.

Figure 9–5 Cog Menu for Property Value Editors


Inspector
Properties : Button
Text
Reset to default
Replace with Internationalized String
Switch to multi-lines mode

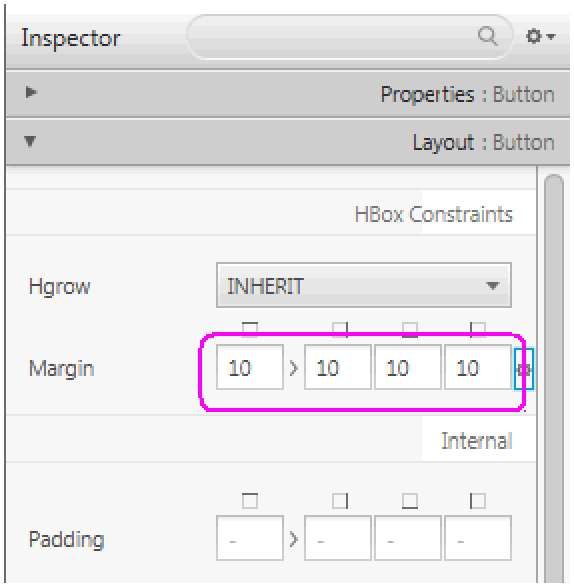
Text: New
CSS Font: System 12px
CSS Text Fill: BLACK
Wrap Text: □
Text Alignment
Text Overrun: ELLIPSIS
Ellipsis String: ...
Underline: □
Line Spacing: 0
Default Button: □
Cancel Button: □

Any numeric value can be increased or decreased using the Up or Down arrow keys, respectively. Using the Shift-Up and Shift-Down keys increase and decrease the number value by 10, respectively.

The editor for the `Insets`, `Margins`, and `Padding` properties allows you to enter the first value (Top) and set the same value to the remaining three text fields (Right, Bottom, Left) using the following steps:

1. Enter the value in the first property text field.
2. Press the **Tab** key and the focus is placed on the link button.
3. Press the **Enter** key to apply the property value you just entered to the remaining three property text fields.
   Figure 9–6 illustrates the uniform values assigned to the Margin property for the Buttons in an HBox.

Figure 9–6 Margin Property Editor



Auto Suggest editors are available for all properties that have predefined values. When you click in the property's text field, a list of suggested values are displayed. The list is filtered as you start typing in the text field.

Some property values require paths to external resource files, such as image files or CSS files. You can specify the file paths using one of the formats listed in Table 9–1.

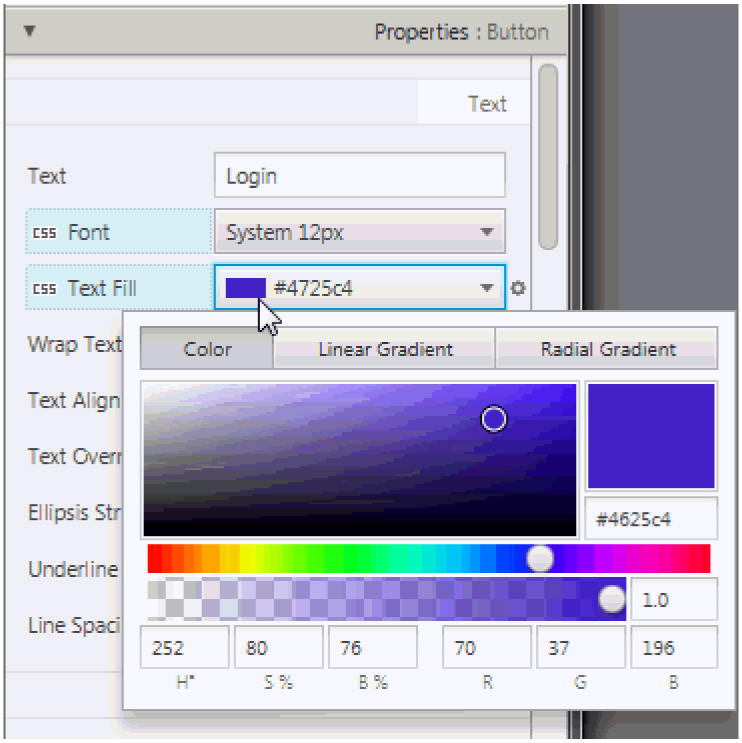
Table 9–1 File Path Formats for External Resource Files

File Format Types	Example
Absolute path	`C:/johndoe/css/myCss.css`
Path relative to the current FXML document's location	`@../css/myCss.css`
Path relative to the class or resource file location	`@/com/oracle/myProject/resources/myCss.css`

A Paint Editor is available for properties that can have color for its value. When you click on the property value field, the Paint Editor window appears. In Figure 9–7 the

Paint Editor window is displayed for the Text Fill color property of the selected Button element. You can change the color value for the GUI element by moving the cursor along the color palette.

Figure 9–7 Paint Editor for Color



Text: Login
CSS Font: System 12px
CSS Text Fill: #4725c4

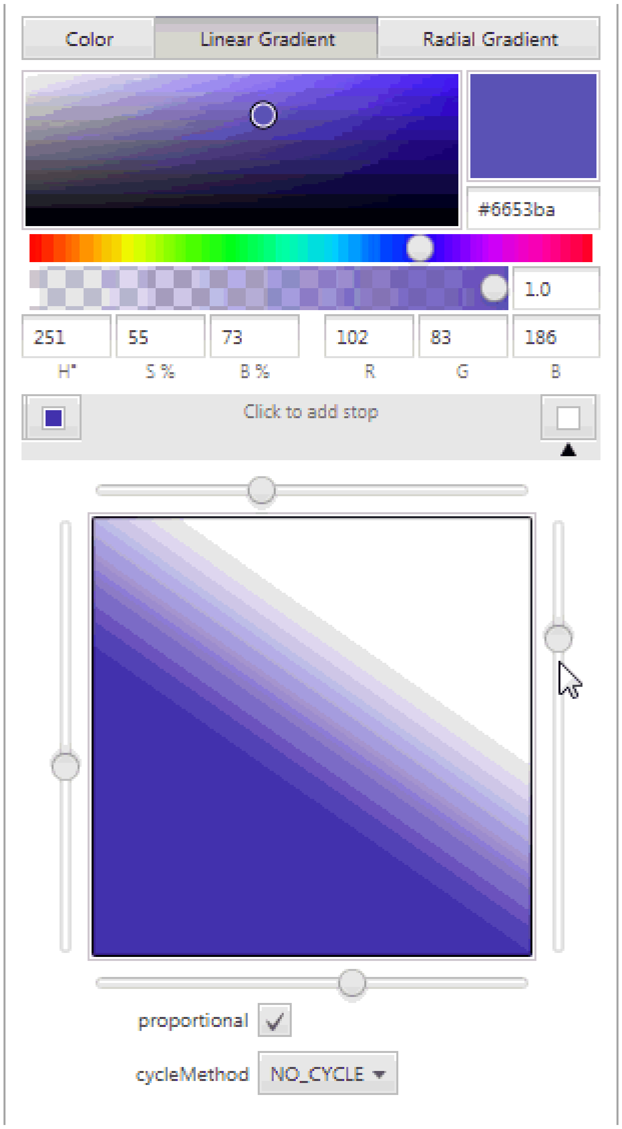
#4625c4
1.0

H°	S %	B %	R	G	B
252	80	76	70	37	196

When you click on the Linear Gradient or Radial Gradient tab, the Text Fill property value is changed accordingly to `Linear Gradient` or `Radiant Gradient`. The Linear or Radial editor section of the Paint Editor is displayed. Figure 9–8 shows the Linear section for the Text Fill property. From that section, you can adjust the gradient values by moving the available sliders on sides until you see the desired color hues you are trying to achieve. Values for additional properties are displayed and can be edited from the editor as well.

As you adjust the color, linear gradient, or radiant gradient for the particular GUI element, the adjustments are also simultaneously reflected on the GUI element that is displayed in the Content panel.

Figure 9–8 Linear Gradient Editor



Layout Section

The Layout section of the Inspector panel, shown in Figure 9–9, helps you to specify the runtime behavior of the layout when the application's window is resized. It also enables you to change the size (such as, Pref Width and Pref Height) and position (such as, Layout X and Layout Y) of the selected element. Some of the information displayed in the Layout section pertains to the selected element itself (such as Min Width, Max Height, or Pref Width), and some of the information depends on the type of the container element. If the selected element is contained in an AnchorPane container, then an AnchorPane section is present in the Layout section. If the selected element is contained in a Grid Pane container, then there is a Grid Pane subsection in the Layout section of the Inspector panel. Some containers, however, do not have any associated contextual information, and so no contextual container subsection appears in the Layout section.

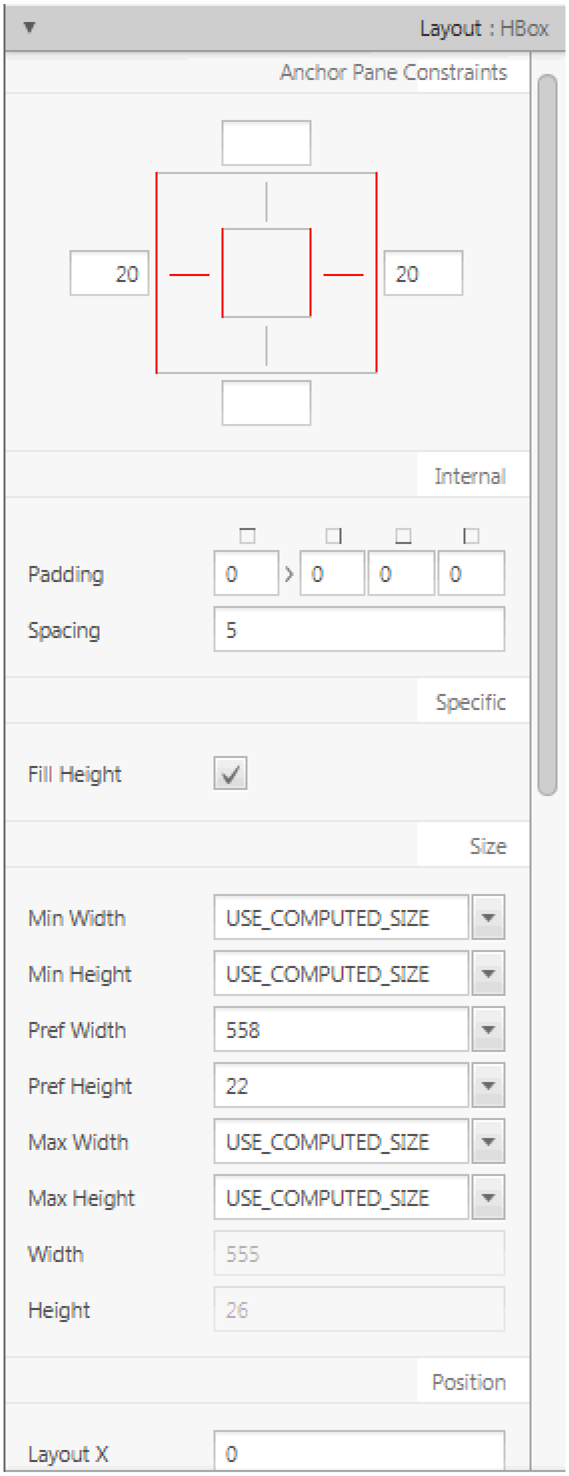
The AnchorPane Constraints subsection only appears when a selected element has an AnchorPane as its most immediate container. For example, in Figure 9–9, the selected

HBox element's container is an AnchorPane. You can use either the diagram or the text fields in the AnchorPane Constraint subsection to manage the anchor pane's anchors. The anchors help you manage the behavior of the selected GUI element when you resize the window. You can click the anchor lines to specify whether the object's size changes as the application's window is adjusted. The lines change from solid black lines to solid red lines when you click them. You can also specify the numerical values in the fields around the diagram. A grayed out numerical value means the anchor is disabled. You can use the Tab key to navigate between the anchor fields.

If the HBox element's most immediate container is a StackPane element, then the top subsection of the Layout section will be labeled as StackPane Constraints and different editable properties will be displayed.

The Transforms subsection enable you to define settings for different types of transformations, such as rotation, scaling, and translation.

Figure 9–9 Layout Section of the Inspector Panel



Anchor Pane Constraints: 20 (left), 20 (right)

Padding: 0 0 0 0

Spacing: 5

Fill Height: ☑

Min Width: USE_COMPUTED_SIZE

Min Height: USE_COMPUTED_SIZE

Pref Width: 558

Pref Height: 22

Max Width: USE_COMPUTED_SIZE

Max Height: USE_COMPUTED_SIZE

Width: 555

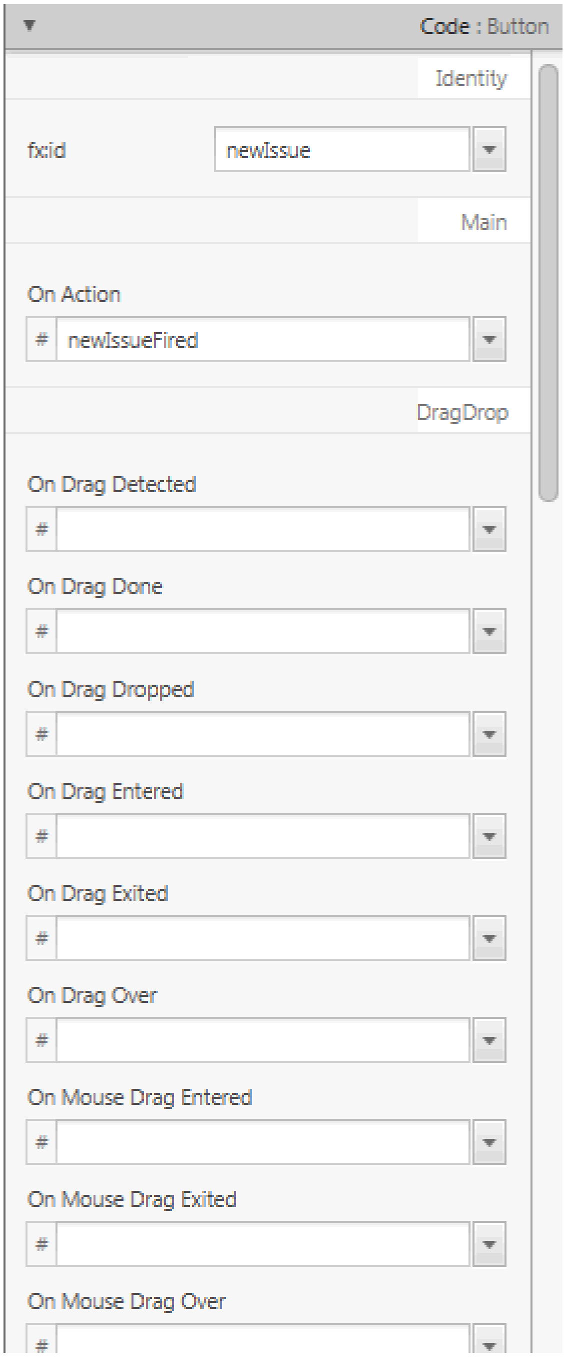
Height: 26

Layout X: 0

Code Section

Figure 9–10 displays the opened Code section when the New button element is selected in the Hierarchy panel. In the fx:id field, you can assign a unique value for the selected GUI element. You can select from the drop-down list of available variables if you already assigned the controller source class to use with your FXML layout.

Figure 9–10 Code Section of the Inspector Panel When the New Button is Selected



fx:id newIssue

On Action # newIssueFired

The Code section enables you to specify the action that should be taken when a drag-and-drop, keyboard, mouse, or any other event occurs on the selected element. From the drop-down list on the event text field, select the corresponding command that should be executed when a specific event or events occur for that GUI element. The given command is a call to the controller method event handler. It is specified by a leading number sign (#), which is already added by default, followed by the name of the handler method. The method is expected to conform to the signature of a standard event handler, which means that it takes a single argument of a type that extends the `javafx.event.Event` class and should return void. For example, Figure 9–10 shows that when the OnAction event occurs on the button, the `newIssueFired()` controller method is executed.

When you click **View** from the Menu bar and then select **Show Sample Controller Skeleton**, a text editor window appears and displays the source file example of what the method should look like in your controller source code. You can copy the contents of the skeleton source file to customize as you please.

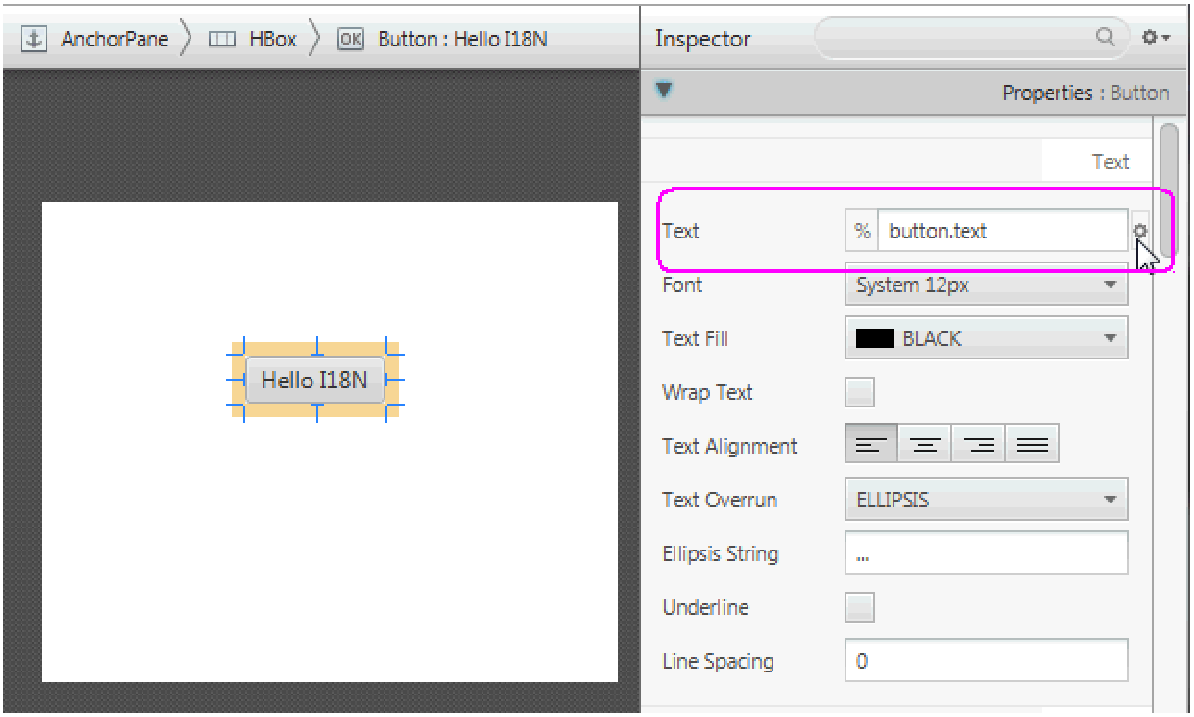
10

Internationalizing Your FXML Layout

This chapter discusses the internationalization support that is provided with JavaFX Scene Builder.

The JavaFX Scene Builder tool enables you to adapt your FXML layout to various languages and regions through the use of resource bundle files. When you add a new element to the Content panel, the element's visible property values are set to default values. For example, a button element has a default Text property value of Button. This default value can be changed with an internationalized string by hovering the cursor over the right of the Text property text field, clicking the round cog icon that appears, and then selecting **Replace with Internationalized String**, as shown in Figure 10–1. Notice that the default value is then replaced by %key.unspecified, which you need to replace with the appropriate resource key name that is defined in your resource file. The percent sign (%) prefix is a visible indicator that the Text property value following it is a resource key name. You can still use the percent sign as a first character for the Text property value. If you have not selected the Replace with Internationalized String setting, then in the FXML file, the percent sign will appear as \%. This indicates that the rest of the property value has no link with internationalization and that it is not a key name.

Figure 10–1 Internationalizing a Button Element



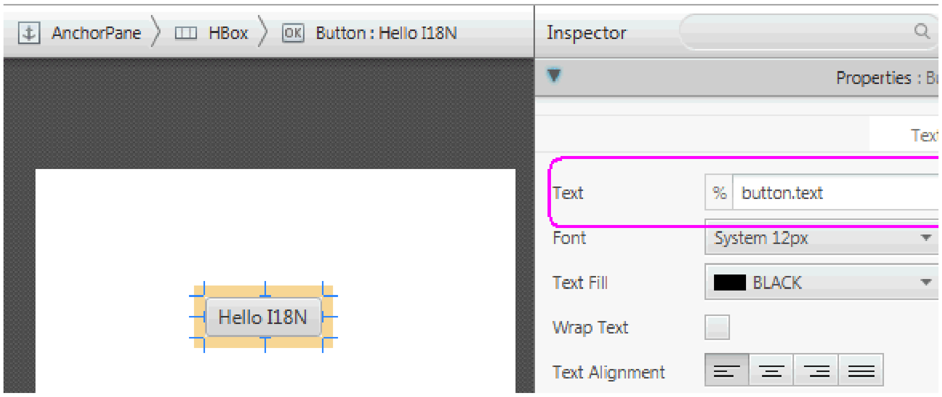
Text: % button.text
Font: System 12px
Text Fill: BLACK
Wrap Text: □
Text Alignment
Text Overrun: ELLIPSIS
Ellipsis String: ...
Underline: □
Line Spacing: 0

The HelloI18N sample can help show the internationalization support. The sample is included with the JavaFX Scene Builder samples bundle. Its `Main` class shows how to load the resource bundle. The sample includes the Bundle.properties and Bundle_fr_FR.properties files. When you open the <*javafx-scenebuilder-samples-2_0-install-dir*>/HelloI18N/src/helloi18n/HelloI18N.fxml file, the FXML file is opened in its own window and you see a button that is labeled %button.text. To localize this label, select **Preview** from the Menu bar, select **Internationalization**, **Set Resource**, and then open the <*javafx-scenebuilder-samples-2_0-install-dir*>/HelloI18N/src/helloi18n Bundle.properties file.

Figure 10–2 shows that the button element's Text property value was set to use the button.text resource key, which is defined in the Bundle.properties file as having the value of HelloI18N. The percent sign in front of the Text property's text field is a visible indicator that the value in the text field is a resource name. If you replace the value assigned to button.text in the Bundles.properties file and save the change, the new value is immediately reflected in the button displayed on the Content panel. You can define additional resource keys in your resource bundle files and use them in the FXML layout you are building.

Select the **Preview** from the Main menu, then **Internationalization** command, and then **Set Resource**. In the select Internationalization Resource File dialog box, select the `Bundle_fr_FR.properties` file to see how the HelloI18N sample application appears when using the French resource file. Notice that the text displayed on the button is now Bonjour. Changes you make and save to the resource file are reflected in the FXML layout you are working on in the JavaFX Scene Builder tool.

Figure 10–2 Using Resource Key Value to Internationalize the Text Property Value



Text % button.text

Font System 12px

Text Fill BLACK

Wrap Text □

Text Alignment

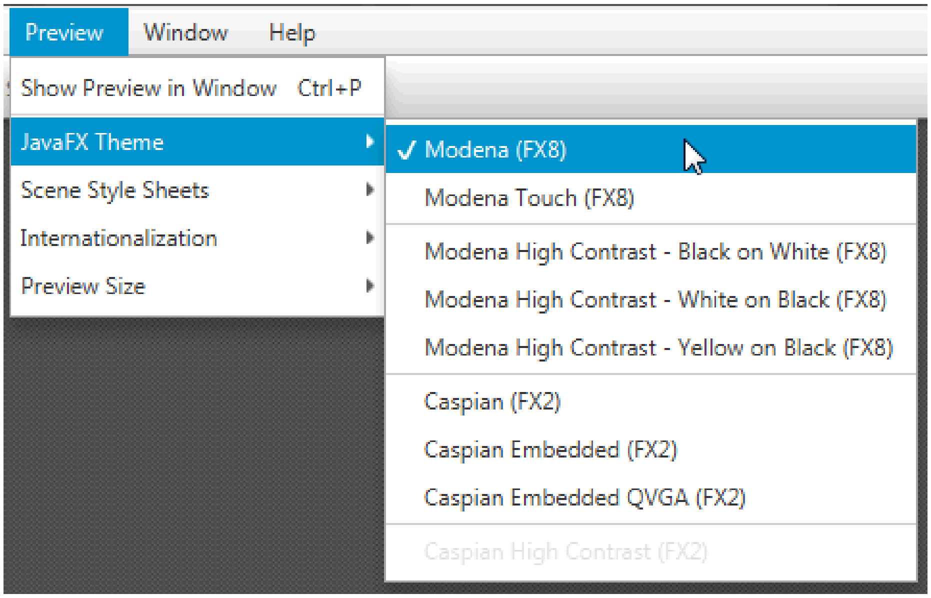
11

Styling with CSS and the CSS Analyzer

This chapter describes the Cascading Style Sheet (CSS) support and describes the CSS Analyzer feature that JavaFX Scene Builder provides.

The JavaFX GUI controls used by Scene Builder are pre-styled with the default JavaFX Modena look and feel. Scene Builder immediately renders this predefined JavaFX style when you drag the GUI control from the Library panel to the Content or Hierarchy panels. You can change the JavaFX theme used in your layout by selecting **Preview** from the Menu bar and selecting one of the JavaFX themes, as shown in Figure 11–1. From the drop-down list you can select a Modena-based theme specific for your target platform. You can also select Caspian-based theme, which was the default theme used in the JavaFX 2.x releases. You can also customize the style used in your application by changing the component's properties via the Properties section of the Inspector panel or by defining your own styling rules in a CSS file.

Figure 11–1 JavaFX Theme Menu

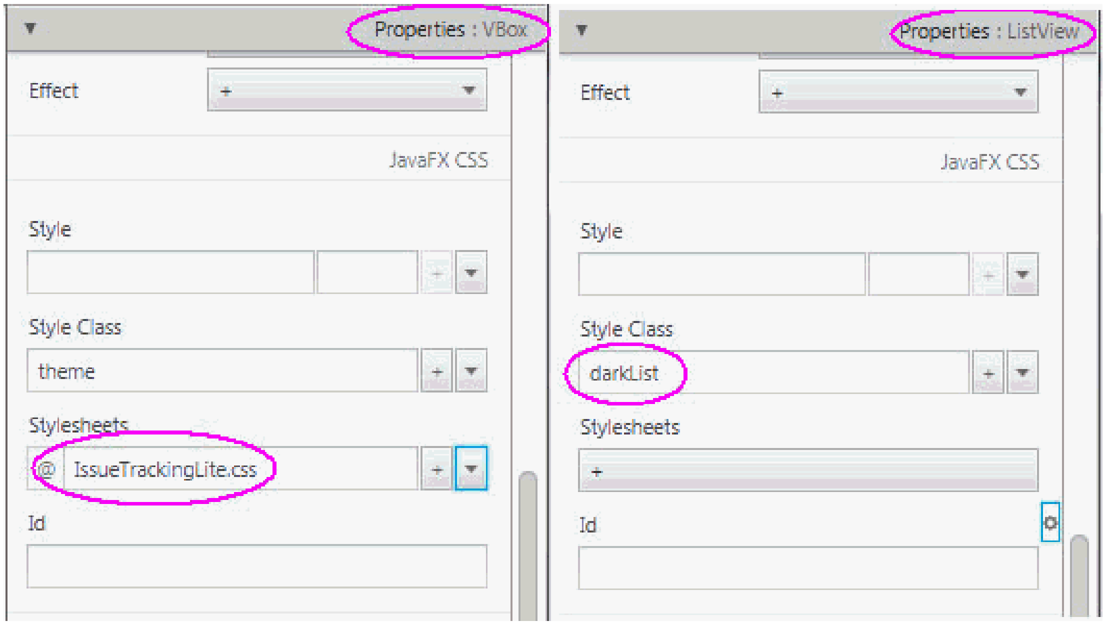

Preview
Window
Help
Show Preview in Window Ctrl+P
JavaFX Theme
Scene Style Sheets
Internationalization
Preview Size
✓ Modena (FX8)
Modena Touch (FX8)
Modena High Contrast - Black on White (FX8)
Modena High Contrast - White on Black (FX8)
Modena High Contrast - Yellow on Black (FX8)
Caspian (FX2)
Caspian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

Scene Builder currently does not generate CSS files, but enables you to use your local CSS editor to create and modify your CSS file. The changes you save in the CSS file that is used by the current FXML layout displayed in the Content panel is immediately rendered by Scene Builder.

You can add the CSS rules at the Scene level, within a given container, or by an inline styling at the Node level. By using the Stylesheets list view in the JavaFX CSS subsection of the Properties section of the Inspector panel, you can assign a CSS file to use on a selected GUI element in your FXML layout if that GUI element is either a container or a GUI control. You can attach a style sheet to any part of your FXML layout, from the topmost parent GUI element to the lowest.

The left side of Figure 11–2 shows the Properties section of the Inspector panel with the IssueTracking.css file assigned to the topmost VBox container used in the layout. The right side of the figure shows the darkList style class assigned to the ListView element. The darkList style class is defined in the IssueTracking.css style sheet and inherited by the ListView element from its container parent, the topmost VBox container.

Figure 11–2 Properties Section with Style Class and Style Sheet List Views Displayed



Style Class: theme

Stylesheets: @ IssueTrackingLite.css

Style Class: darkList

CSS rules that are defined on a parent element can be used to style the parent GUI element itself and all of its children elements. You can define the specific style class to use with a GUI element by adding it to the Style Class property in the Properties section of the Inspector panel. The CSS files referenced from the Stylesheets properties are referenced from within the FXML file and so they are deployed with the FXML file.

In Scene Builder, you can simulate the attachment of a style sheet to an application Scene by selecting **Preview**, then **Scene Style Sheets**, and finally selecting **Add a Style Sheet** or **Open a Style Sheet** option. This Preview command is useful when the “root” style class is defined in the style sheet. In this case, once the style sheet is attached to the Scene, the styles defined in the “root” class are applied to the layout in the Content panel.

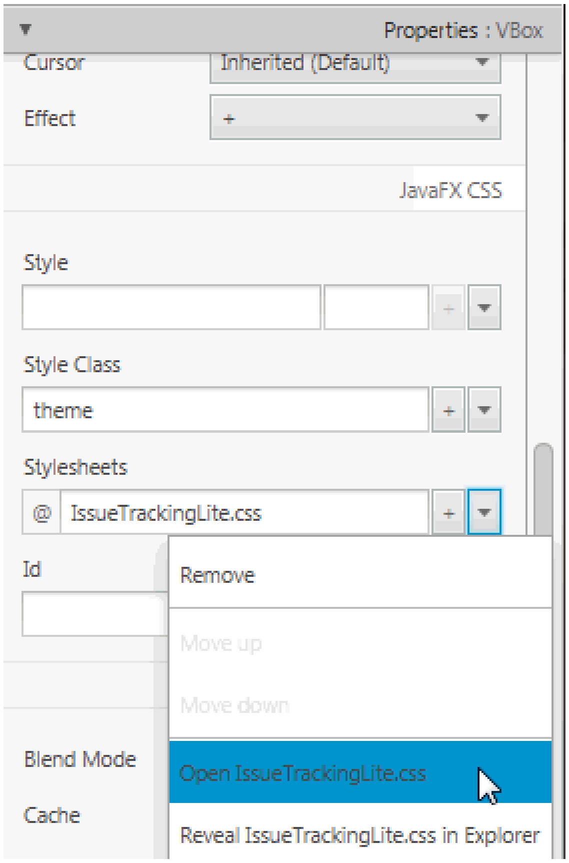
You can edit an existing CSS file with your system editor by using the following steps:

1. In the Properties section of the Inspector panel, locate the Stylesheets property and click on the CSS file you want to edit.
2. Click the drop-down arrow on the bottom right of the list view, as shown in Figure 11–3.

3. Select the **Open** command for the CSS file you want to edit.

   You can reveal the CSS file’s location in your system using the Reveal command that is also available in the drop-down menu. You can also navigate to the CSS file via the CSS Analyzer panel, as explained later in Using the CSS Analyzer Panel.

Figure 11–3 Open CSS File From the Properties Section of the Inspector Panel

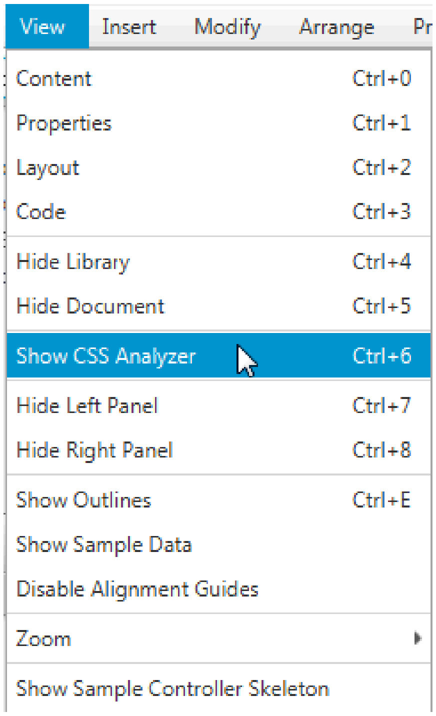


Using the CSS Analyzer Panel

The CSS Analyzer Panel enables you to understand how various possible CSS rules can affect aspects of a currently selected GUI element. It displays a synoptic view of all the possible sources for the property values. Each CSS property value assigned to a certain aspect of the selected GUI element may originate from either the API or predefined CSS rules. The sources are listed in prioritized order, which enables you to understand why a given source takes precedence over another. The panel also provides a means for you to navigate to the source of CSS property value to help you develop and troubleshoot CSS stylesheets.

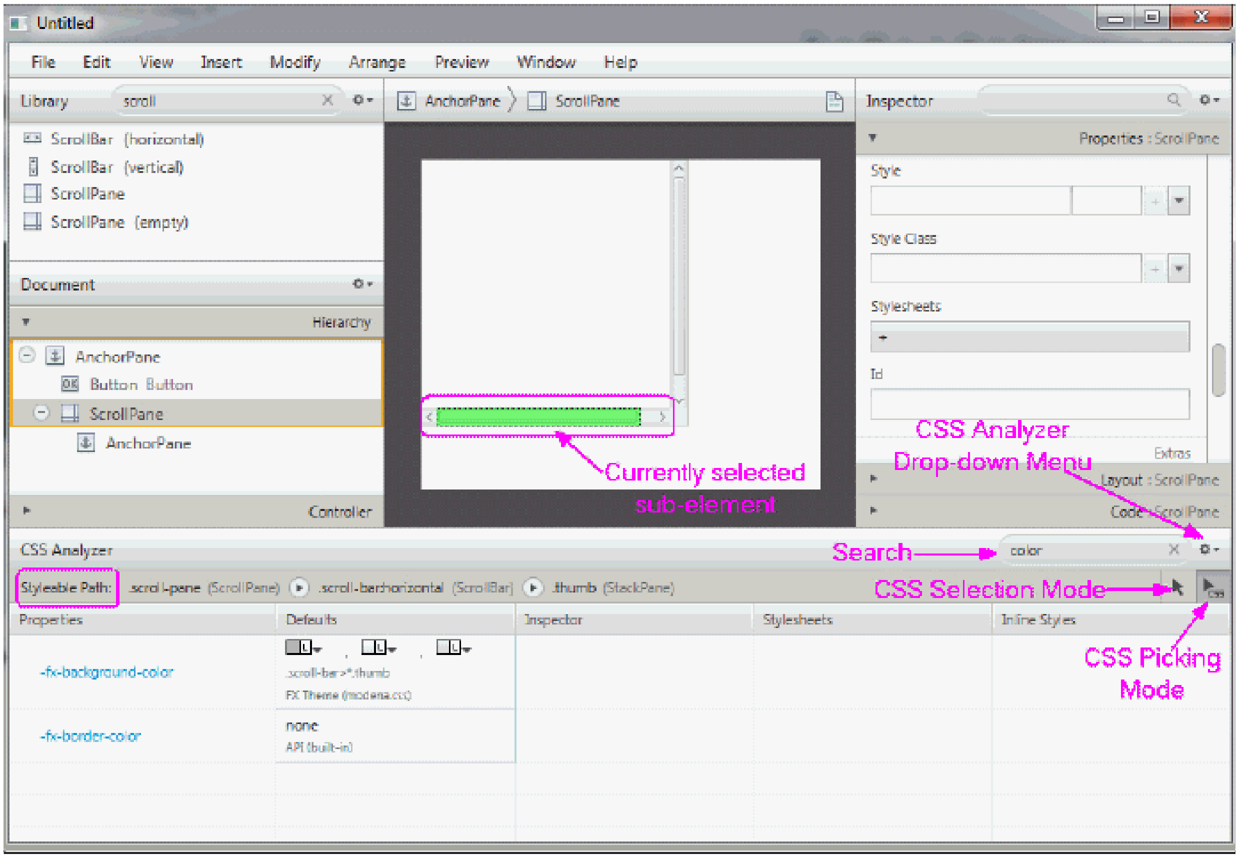
The CSS Analyzer panel is not displayed by default. To display the panel, select **View** from the Main menu and then **Show CSS Analyzer**, as shown in Figure 11–4.

Figure 11–4 Show CSS Analyzer



The CSS Analyzer panel appears at the bottom of the Main window, similar to what is shown in Figure 11–5.

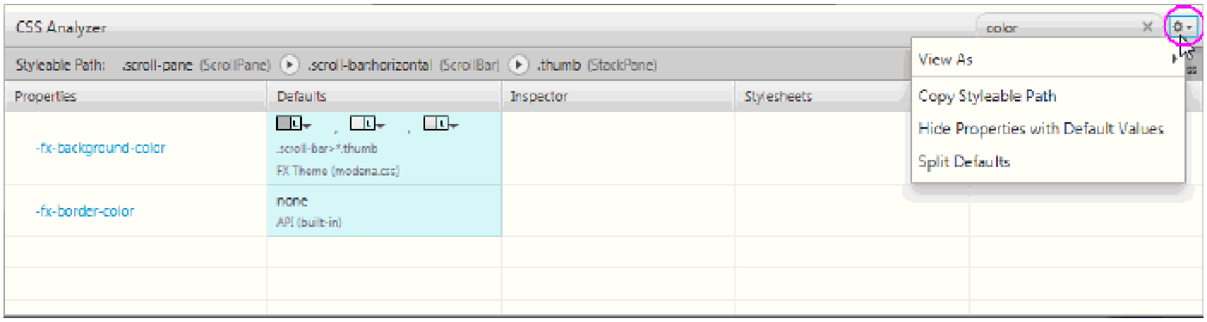
Figure 11–5 CSS Analyzer Panel Displayed



The panel includes the following sections, most of which are highlighted in Figure 11–5.

- **Search** text field: Located at the top right corner of the panel. It enables you to isolate the specific properties that you want to view.
- **CSS Analyzer Menu:** Located to the right of the Search text field. Click on the drop-down arrow to see the menu of available commands, as shown in Figure 11–6.
  - **View As:** Enables you to select the display format for the style properties. The default is Table view. The Rules view mode displays the properties in CSS rules formatting. The Text view mode displays the properties in text styling.
  - **Copy Styleable Path:** Enables you to copy the current value in the Styleable Path text field and you can paste it into your CSS file to modify the style of the currently selected component or its sub-element.
  - **Hide Properties with Default Values:** Removes from view all the properties that only have default values assigned to them. Properties that have non-default values, such as Stylesheets and Inline Styles, continue to be displayed. The **Show Properties with Default Values** shows all the properties for the component.
  - **Split Defaults:** Refreshes the view of the properties to include two columns for the default values of the style properties: API Defaults and the FX Theme Defaults. The Join Defaults command displays the combined default values into a single column, which is the default.

Figure 11–6 CSS Analyzer Menu



CSS Analyzer

color

Styleable Path: .scroll-pane (ScrollPane) ▸ .scroll-bar:horizontal (ScrollBar) ▸ .thumb (StackPane)

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

- **Styleable Path text field:** Located on the top left corner of the panel. It enables you to discover the sub-elements when you click on the arrows in the path. You can copy that path using the CSS Analyzer menu and paste the path in your CSS file to assign a new style value.

- **CSS Picking Mode button:** Located below the Search text field. This is the default mode when CSS Analyzer is opened. It allows you to select a sub-element of the currently selected component. In Figure 11–5, the CSS Picking Mode button is selected. This allowed you to select the bottom horizontal scrollbar of the ScrollPane component in the Content panel. The scrollbar is highlighted in yellow to indicate that it is the currently selected sub-element of the ScrollPane. Correspondingly, the Styleable Path shows the complete path to the currently selected element. This feature shows how skins can be styled.

- **CSS Selection Mode button:** Located next to the CSS Picking Mode button. It is the standard selection mode and allows you to select a component.

- **Properties column:** First column in the table. Displays all of the available style properties that is available for the currently selected element.

- **Defaults column:** Displays the default values, both from the API and the JavaFX theme, that are delivered for the style property.

- **Inspector column:** Displays the property value that has been set using the Inspector panel. Some properties appear in both the Properties section of the Inspector panel and the CSS Analyzer panel. To edit these properties, hover the mouse cursor in the corresponding cell of the CSS Analyzer panel and click the cog icon in the top right corner to select **Reveal in Inspector**. The corresponding property is highlighted with a blue ring in the Properties section of the Inspector panel.

- **Stylesheets column:** Displays the value that is defined for the property via the CSS file set in the Stylesheets text view in the Properties section of the Inspector panel. The name of the CSS file the value is derived from is included in the column. You can also click on the cog icon on the upper right corner of the cell and select the **Open** command to open your CSS file in your default CSS editor.

- **Inline Styles column:** Displays the inline style value that is defined for the property via the Style text view in the JavaFX CSS subsection of the Properties section of the Inspector panel. You can select the **Reveal in Inspector** command using the cell's cog icon on the upper right corner of the table cell to display the Style text view in the Inspector panel.

You can modify the properties values from your CSS file. Alternatively, you can use the JavaFX CSS subsection of the Properties section of the Inspector panel to either edit a CSS property or use the Style property to override any CSS property.

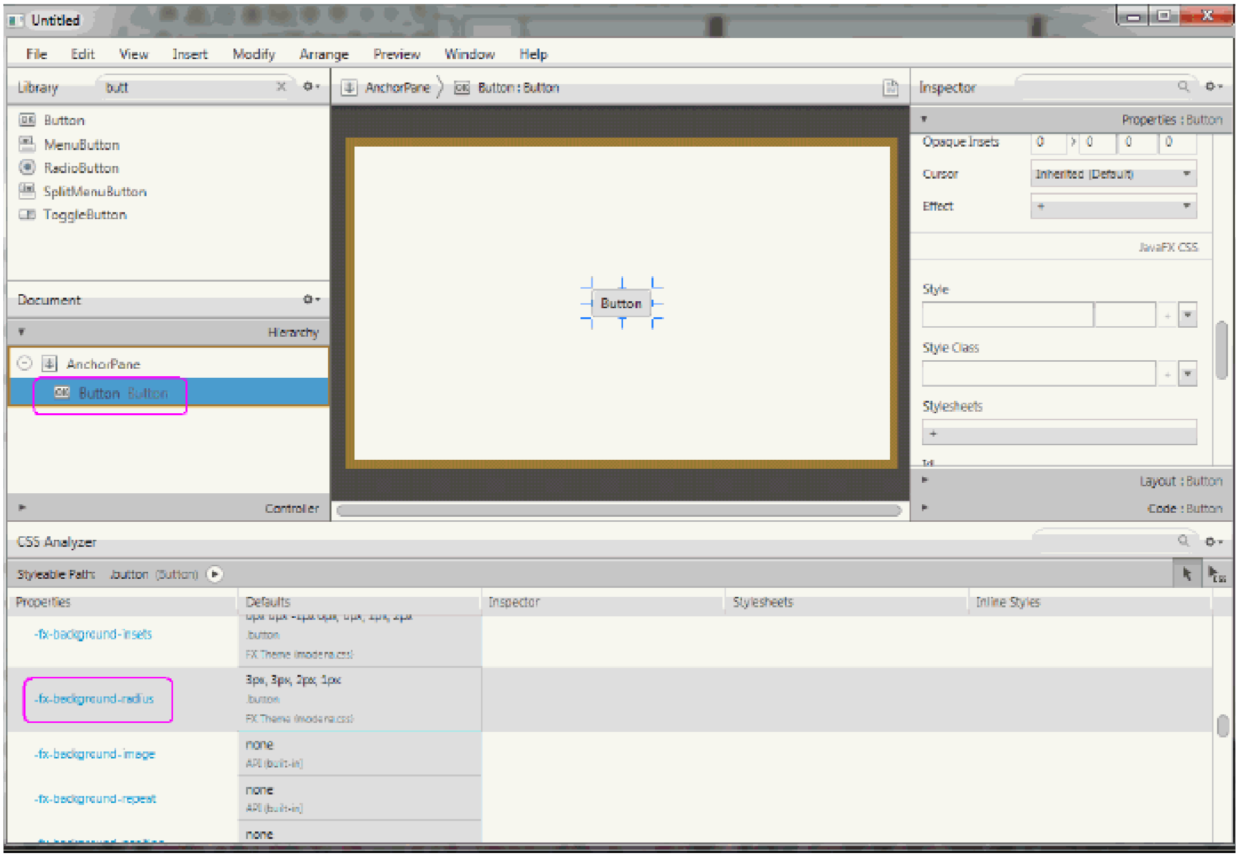
When a CSS property value is assigned to a currently selected GUI component, the new style is immediately rendered in the Content panel.

To learn more about the CSS Analyzer's functionalities, use the following steps to customize the Button component to have rounded edges and use that style for all buttons that are subsequently added to your FXML layout.

1. Drag a Button from the Library to the Content panel.

   You will see the CSS Analyzer displays the value of -fx -background-radius, shown in Figure 11–7, as one of the CSS properties for the button.

Figure 11–7 A Button's Properties Displayed in CSS Analyzer



Properties	Defaults	Inspector	Stylesheets	Inline Styles
-fx-background-insets	0px 0px -1px 0px, 0px, 1px, 2px .button FX Theme (modena.css)			
-fx-background-radius	3px, 3px, 2px, 1px .button FX Theme (modena.css)			
-fx-background-image	none API (built-in)			
-fx-background-repeat	none API (built-in)			
-fx-background-position	none			

2. Click the **-fx -background-radius** property name in the Properties column and you are taken to the corresponding online API documentation in the `JavaFX CSS Reference Guide`, which provides the syntax you can use with each property's value.

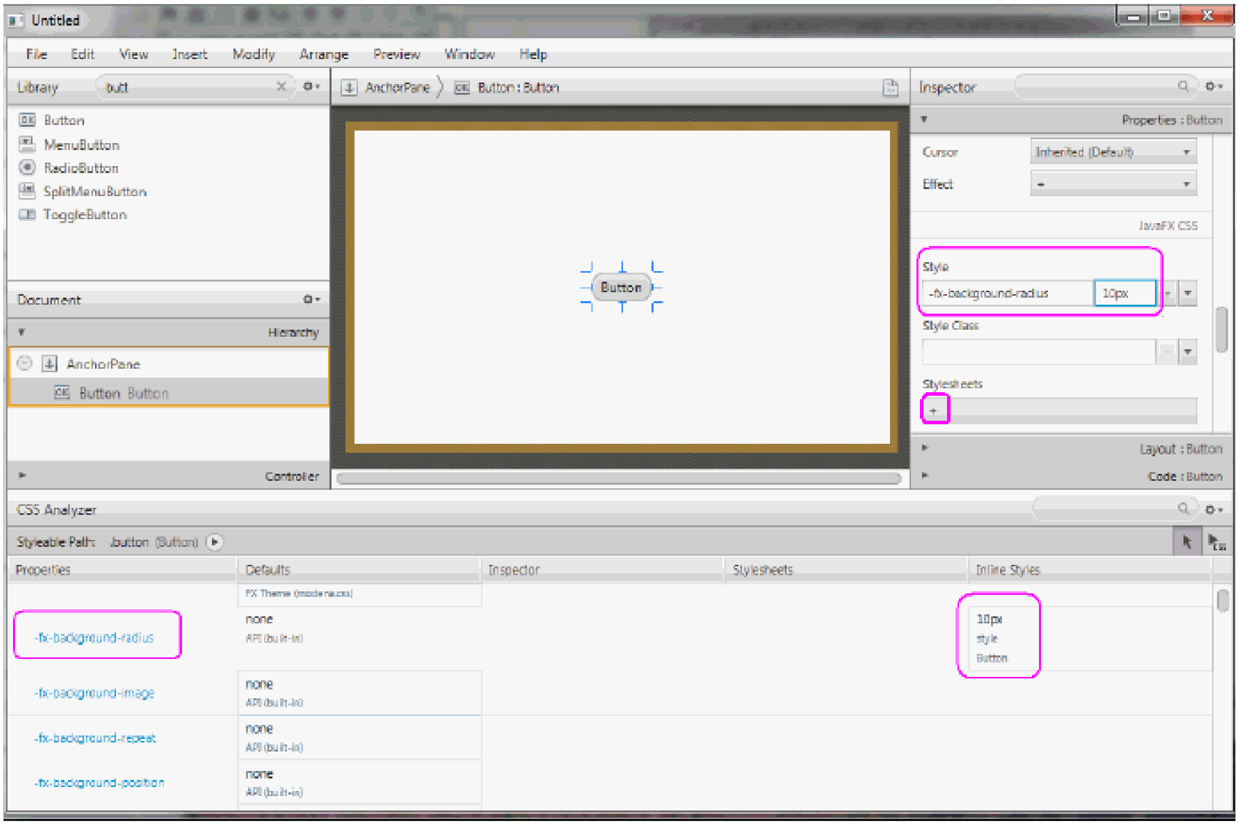
3. Modify the -fx-background-radius property value.

   a. In the Scene Builder window, click on the Properties section of the Inspector panel.

   b. Click in the Style text field located in the JavaFX CSS sub-section and select **-fx-background-radius** from the drop-down list.

c. Enter `10px` in the Style text field of the JavaFX CSS subsection, as shown in Figure 11–8.
Notice that the Button component in the Content panel is now rounded around the corners. Also, in the CSS Analyzer panel, you'll see that the row for the -fx-background-radius property now has two values: the default and the inline style value of 10px that you just entered in the Style property, as shown in Figure 11–8. That cell's background color also changed to blue, which indicates that the value is now the current value being rendered on the currently selected component

Figure 11–8 Making a Button Rounded



4. Create a CSS rule so that you can apply the new button style to all of the buttons you add to your current FXML layout.

a. Using your system's command, create am empty CSS file, e.g. `test.css`, in the same directory that contains your current FXML layout.

b. In the Properties section of the Inspector panel, click the **+** button in the Stylesheets property, as indicated in Figure 11–8.

c. In the Add Style Sheet dialog, navigate to the `test.css` file location, select it, and click **Open** to add it to the Stylesheets property.
You'll see the `test.css` file added in the Stylesheets property.

d. Edit the `test.css` file by clicking on the down arrow in Stylesheets property and select **Open test.css**.

e. Add the following CSS rule to `test.css` and save the file.

Example 11–1 Add CSS Rule for a Rounded Button

```
.button {
-fx-background-radius: 10px;
}
```

f. Select the button again in the Content panel. Notice that in the CSS Analyzer panel, the button is now inheriting the CSS rule from the `test.css` file because the button is a child of the AnchorPane component. Since it still has the inline style value assigned to it, that value has precedence and is the style rendered in the Content panel. If you do not have a need to change the style for this particular button, you can remove that style value from the Properties section of the Inspector panel.

12

Next Step

Now that you are familiar with the JavaFX Scene Builder user interface, use Getting Started with JavaFX Scene Builder to create a simple issue-tracking application.