Flex in a Week, Flex 4.5

Video 5.03: Defining styles in CSS

In the last exercises, you added Sparktext controls to the Employee Portal application.

In this video, I will show you how to enhance the application further by first using Design mode to define some global application styles.

Then you will learn how to define CSS styles for the Flex framework components and for your own custom components.

Let me take a moment to remind you that, while MX components have many built-in styles, Spark components have fewer of them.

As you learned throughout this series, the Spark components have some basic styling abilities, but if you are going to change the component display more drastically, you would actually implement a skin for the component.

You will learn how to do that in the next few videos.

This means that, for Spark, CSS is largely used for three main purposes.

To define some overall properties for an application, to wire up skins to components, and to apply advanced CSS selectors.

You will learn how to perform these three tasks with CSS in this last Day of training.

This image shows the enhancements that I will make to the application.

I will use the Appearance view in Design mode to make a global change to all the text controls.

Then I will style the Flex framework's Panel and Button controls.

Lastly, you will learn how to create a selector to target the styles in the Monthly Events custom component.

This is the starter version of my application for this video.

You can see that the application appears with white panel container backgrounds and uses an Arial font throughout the application.

I will update all the panels to have gray backgrounds and use Verdana for the font.

In Flash Builder, note that there is only one file in the default package for this project – the starter MXML application file.

I am switching to Design mode.

To the right of the Properties view, I am selecting the Appearance view.

From here I can change the global application font from Arial to Verdana, and the font size to 11.

When I use the Appearance view, Flash Builder automatically creates a CSS file for me in the default package of the project.

The Flex application points to that CSS file using this Style tag that was inserted into the main application file.

I’m going to move that Style tag right here to follow my coding convention.

I am double-clicking on the CSS file to open it.

Starting with Flex 4, each CSS file defines namespaces for the Spark and MX libraries at the top of the file.

You can also see the global selector that defines the settings that you selected in the Appearance tab.

The global selector sets styles that will apply to the entire application.

I am saving the files and running the application.

You can see that the Verdana font at a size of 11 appears smaller than Arial.

You can apply styles to all instances of a Flex component in your application by using the Flex components as selectors in CSS.

You use the familiar component name, for instance, in this case, Button, and prefix it with the appropriate namespace.

Between the namespace and component name place a pipe character.

The rest of the CSS syntax is the same.

You define the styles and their values between the curly braces for the selector.

Back in Design mode, in the main application file, I am selecting the Employee of the Month panel and changing its background color to #E8E8E8 using the Properties view.

When I switch to Source mode and locate the same container, you can see that it has a backgroundColor property set to the value I defined.

I'm saving the file and running the application.

The Employee of the Month panel now has a light-gray background color but the other panels do not.

I am returning to Flash Builder and opening the CSS file.

Below the global selector, I am referencing the Panel container in the Spark namespace and setting its backgroundColor to a light gray.

When I save the file and run the application, you see that all of the Panel containers have a gray background.

I'm returning to the main application file and removing the backgroundColor property from the Employee of the Month Panel container.

When I save the file and run the application you can see that there is no visual change to the application.

This means that the component selector is working for the Panel containers.

Flex also supports the creation of multiple selectors with the same name and different styles.

In this example, you can see that the Button control, in the Spark namespace, is being defined three times.

The first two instances have different defined styles, but the last instance is defining the same style, color, as the first.

If styles in more than one selector overlap, than the last one is applied.

Otherwise, all are applied.

Within the CSS file, below the Panel container selector, I am creating a Spark Button selector.

I am setting the color property to blue, #0074AA.

When I save the file and run the application, you can see that all of the text values in all of the Button instances are blue.

I am creating a second Spark selector, for the Button control, with a font-family property set to Arial.

I’m creating a third selector and changing the color value to black.

Let’s see whether the black color or the blue color is applied.

When I save the file and run the application, you can see that all the Button control labels are Arial and black.

The last color property defined is used.

Just as in your Flex application, you can create custom namespaces in your CSS files.

You create a custom namespace using the @namespace directive.

Then you add the arbitrary namespace alias, which is cx here, and the package it applies to within double quotes.

Follow that statement with a semicolon.

You apply the custom component selector by referencing the namepace and custom component name with a pipe character in between.

All of the properties that you define between the curly braces are applied to the custom component instances.

In the CSS file, below the Spark and MX namespaces, I am using the @namespace syntax to create the namespace cx, that references the custom components in the components folder.

At the end of the file, I am using the cx namespace to reference the MonthlyEvents custom component and setting its color property to blue.

When I save the file and run the application, you can see that the component’s content is blue text.

I am returning the main application file and locating the Monthly Events component.

To the component I am adding a color property and setting its value to #000000, black.

I'm saving the file and running the application.

The color of the text within the Monthly Events panel is now black.

This happens because the property on the Panel container overrides the value set with CSS.

For your next step, work through the exercise titled “Defining selector styles”.