Flex in a Week, Flex 4.5

Video 5.08: Creating custom skin properties

You now know how to create component skins and draw graphics within them using MXML.

In this video, you will learn how to create a contract between a component skin and a custom component.

You will also learn how to use the value set for a custom property in the skin definition.

This is the Employee Portal application that you completed in the last exercise.

You created an application skin and you used graphic code generated by Adobe Illustrator CS5 to apply this blue background to the header area.

You also applied a skin to three of the Panel instances with orange highlights on a gray header bar.

In this video, I will show you how to extend the Panel container to create a custom property that will accept different color values for implementation in the skin class.

This is the main application file for the Employee Portal site.

Currently there are four custom components—the Employee of the Month, the Employee Directory, the Cafeteria Special and the Monthly Events components.

I am highlighting all four custom components and commenting them out and saving the file.

Within the Package Explorer view, I am right-clicking the components package and selecting New > MXML Component.

In the New MXML Component dialog I am naming the component EmployeePortalPanel and then clicking the Browse button next to the Based on field.

Within the Based on dialog, I am typing Panel and selecting the Spark Panel container component class.

I'm setting the width to 250 and the height to 475.

I am clicking the Finish button to see that the new MXML component file opens in the Editor view.

Now, I want to be able to add a custom color property to each of the Panel instances.

I'm cleaning up the code a little bit, and then adding a Script block.

Inside the Script block, I’m declaring a public variable named panelTitleColor and data typing it a uint,

It will be accepting a color value.

I am saving the file and returning to the main application file.

After the commented components tags, I am creating an instance of the EmployeePortalPanel custom component I just created.

To the opening tag, I am adding the title property with a value of EMPLOYEE OF THE MONTH.

I am pressing CTRL+Space to use content assist to add the panelTitleColor property and giving the property a value of #64BC48.

Between the tags, I am adding an instance of the EmployeeOfTheMonth component.

I am saving the file and running the application.

There are two panels for the Employee of the Month.

The one in the back is the EmployeePortalPanel instance because it has a title property.

The one in the front is the EmployeeOfTheMonth instance.

This doubling occurs because each component is based on the Panel container

I am returning to Flash Builder and CTRL + clicking to open up the EmployeeOfTheMonth component.

I am changing component's Panel container to a Group container.

I’m also changing the width and height values to 100%.

I’m copying this so that I can also apply it to the other two panels.

I am saving the file and returning to the Package Explorer to open up the Cafeteria Specials component and making the same change there.

I’m saving that file and then selecting the Monthly Events panel to do the same thing.

repeating these steps for the Cafeteria.mxml and MonthlyEvents.mxml files.

Remember that we’re not applying any skin to the Employee Directory.

Back in the main application file, I am copying the x, y and skinClass properties from the commented EmployeeOfTheMonth component and pasting them into the opening EmployeePortalPanel component instance.

I’m also adding the height property with a value of 295.

When I save the file and run the application, you can see that only the Employee of the Month panel is displayed and the panel's box is still the same color.

In the skin class, I will first need to modify the HostComponent Metadata directive to make the skin contract with the custom component rather than the Flex framework component.

Now, I can access the custom component property, in this case, the panelTitleColor, by referencing the hostComponent instance.

Notice the lowercase letter h in hostComponent.

I’m locating the PanelContainerSkin class in the skins package and then locating the Metadata tag block and changing the value of the HostComponent directive to reference the EmployeePortalPanel component.

Now, in this skin, I will create a function to handle the panelTitleColor property, which I declared in the custom component

I am locating the nested fill property within the second Rect tag block, which draws the colored box in the panel’s header.

From the nested SolidColor tag, I am removing the color property value and assigning the id property with a value of headerSkinColor.

Now, under the Script comment, I am adding a Script block.

Within the Script block, I am creating a private function named initTitleSkin that returns a void data type.

In the function, I am assigning the color property of the headerSkinColor that I just created to the hostComponent.panelTitleColor property that I defined in this custom component.

To the opening SparkSkin tag, I am assigning the creationComplete property value to call the initTitleSkin() function.

I'm saving the file and running the application.

The color block in the Employee of the Month panel is now green.

Back in the main application file I have updated all my panels, except the Employee Directory panel, to use the EmployeePortalPanel custom component and the PanelContainerSkin.

Each defines a different panelTitleColor property.

Employee of the Month is green.

Cafeteria Special is orange and Monthly Events is blue.

I'm saving the file and running the application.

You can see that the three upper panels display their defined colors and the Employee Directory panel remains skinless.

For your next step, work through the exercise titled “Creating custom skin properties”.