Flex in a Week, Flex 4.5
Video 5.11: Skinning the SkinnableDataContainer

At the beginning of Day 4, you used the DataGroup container to display employee information in a repeating item renderer.

Now that you know how to create and apply Spark skins, I will introduce you to the SkinnableDataContainer as the skinnable equivalent to the DataGroup container.

You will learn how to create a skin for the SkinnableDataContainer and how to further modify the associated item renderer.

This is the Employee Portal: Employee Directory display that you created in Day 4 with the DataGroup container and an item renderer.

Each employee has a discrete background display.

In this video, I will show you how to convert the DataGroup container into a SkinnableDataContainer and then apply this light gray background around the entire list of employees.

I will also show you how to modify the item renderer to create a subtle hovered state for the each employee.

In earlier videos, you learned that the DataGroup container itself is not a skinnable container.

However, it does have a skinnable equivalent: the SkinnableDataContainer class.

The only difference between the two is that the SkinnableDataContainer can have visual elements applied to it.

You apply a skin class to it using the skinClass property.

This is the main application file for my example.

I am locating the DataGroup container and changing the opening and closing tags to the SkinnableDataContainer.

When I save the file and run the application, you can see that the application looks the same as it did before you applied the new container.

I am returning to the main application file and highlighting the opening SkinnableDataContainer tag and switching to Design mode.

In the Properties view for the SkinnableDataContainer, I am clicking the icon to the right of the Skin field and selecting the Create Skin option, which is outside your field of view.

In the New MXML Skin dialog box, I'm typing skins for the Package and BackgroundSkin for the Name.

I'm clicking Finish and the new skin file opens in the Editor view.

This file is configured with a contract between the host component and the SkinnableDataContainer class.

It also contains a states block with the normal and disabled states defined.

Below the states block, I am adding a Rect primitive block and adding the height and width properties and setting their values to 100% and 120, respectively.

I am also adding the radiusX and radiusY properties and setting both their values to 4.

Within the Rect block, I am adding a fill property block.

Inside the fill block, I am adding an instance of the SolidColor class and assigning it the alpha property with a value of .4 and the color property with a value of 0x000000.

This Rect block draws a gray rectangle, that is lightened with an alpha property and has rounded corners.

You learned about all of these elements earlier in this Day of training.

I'm saving this file and then returning to the main application file.

Notice the skinClass property has been added to the SkinnableDataContainer class and it is referencing the skins.BackgroundSkin class.

When I save the file and run the application, you can see that the skin has been applied to the SkinnableDataGroup container, but the content is not centered within the skin.

Remember from the previous videos that the Application container has one skin part named contentGroup that represents all of the content in the container.

The SkinnableDataContainer, likewise, has one skin part, but it is named dataGroup and is an instance of the DataGroup container.

It represents all of the content in the container.

In my example that would be all the employees displayed via the item renderer.

At the end of the BackgroundSkin.mxml file, I am locating the DataGroup container with the id property value of dataGroup.

I am assigning the DataGroup container the horizontalCenter property with a value of 0.

I am removing the left, right and top properties and reassigning the bottom property a value of 5.

When I save the file and run the application you can see that the data is now centered within the skin, but notice that when I mouse over an employee there is a small rectangle that indicates I am hovering over the item.

This occurs because the top property of the BorderContainer of the EmployeeItemRenderer class has been set previously.
Remember that each item renderer display is defined in the itemRenderer property class.

To create the skin for each of the item renderer instances, you simply add the visual elements to the renderer class file.

Back in the main application file, I am opening the EmployeeItemRenderer.mxml file by CTRL + clicking on it.

This BorderContainer block draws the background for each item renderer and also displays the employee data.

I am cutting the top property from the opening BorderContainer tag and saving the file.

I'm returning to the BackgroundSkin.mxml file and pasting the top property into the opening tag for the DataGroup container.

I’m saving the file and running the application.

When I mouse over an employee now, the rectangle above the employee item is now gone, but I also want to add a hover effect.

Back in the EmployeeItemRenderer.mxml file, I am adding another State instance to the states block and setting the name property to hovered.

Below the BorderContainer instance I am creating a Rect instance with a height value of 100% and a width value of 100%.

This will create a rectangle that will completely overlay the item renderer.

Next, I am creating a fill property tag set with a nested SolidColor instance that has an alpha value of .2.

This makes the highlight transparent.

Lastly, I am adding a color property with a value of #FFFFFF to create a white highlight when a user hovers over the item with the mouse.

This code would display the new rectangle all the time.

I am adding the includeIn property to the Rect instance to define that it only appears in the hovered state.

I am adding the .hovered state to the color property as well.

When I save the file and run the application, you can see the hovered effect when I mouse over an employee.

For your next step, work through the exercise titled “Skinning the SkinnableDataContainer container”.