**97. Using HostBinding to Bind to Host Properties**

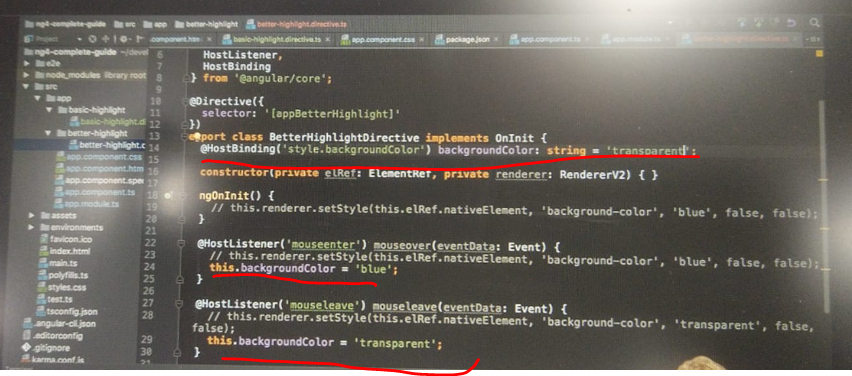
* -: So we learned about host listener and this setup here is absolutely fine for what we're doing but we got another decorator we can use which then allows us to not use the renderer.
* There is nothing wrong with using the renderer but we got an even easier way of simply changing the background color if that is all we want to do in the directive.
* Again, using the renderer is not wrong though.

**Easier way of changing the background colour:@HostBinding**

* The decorator I'm referring to is called at host binding which also needs to be imported from at Angular core.
* And since we get a lot of imports I'll quickly restructure this to cover multiple lines.
* So host binding, that's the new decorator and I will come back to what we pass here in a second.
* First of all, we need to bind this to some property which value will become important.
* So that could be a background color property a new property I create here, which is of type string.

**@HostBinding Syntax:**

* Now in host binding, we can pass a string defining to which property of the hosting element we want to bind.
* Now, properties of the hosting element that is simply what we also access here in the basic highlight directive.
* Style would be such a property And there then the background color property to be precise.
* Therefore, we can simply say style dot.
* And now camel case background color and that's all.
* Camel case is important here because we're accessing the Dom property which doesn't know dashes.



* So with this, what we're telling Angular is on the element this directive sits, please access the style property which pretty much any input has on our directives which access something like the value.
* You of course have to make sure that they are only added to inputs which have this property.
* But here again the style property should be available on any element.

Graphical user interface, application

Description automatically generated

* So here we access the style property and then there a sub property, the background color.
* And we set this equal to whatever background color is set to here.
* Well, and we can simply change background color here when we mouse over it.
* We can set background color to blue in this case and set it transparent in this case here.
* Transparent and I will comment out the render code here.
* Again, not wrong to use it, but no longer need it in this example.
* Of course, we also have to set some initial color so that we don't get an error before we mouse over it the first time.
* So here I will set this to transparent.
* Initially, give this initial value.
* With this it's transparent, no error looks good.
* And if we hover over it, it works the same way as before, but now solely driven through host listener and host binding.
* A great way for working with the element inside of a directive.
* And of course, here on host binding, you can bind to any property of the element you're sitting on.