**81. Getting Access to ng-content with @ContentChild**

* -: In the last lecture, we learned that we have ViewChild and how we can access it in our lifecycle hooks.
* Now, there's one other nice little addition.
* In app component where we add our content which we then project into our server element.
* Let's say we also want to place a local reference on this paragraph, so the contentParagraph maybe.
* But now we want to use that in our server element component which is where this content will end up in the end.
* Now, in the app component, which is where we currently placed this in the template, we could use at ViewChild because it's in the template of the app component even though it will be kind of passed on to the server element component later.
* Since we know that it will be there it would be nice if we could also access it from the server element component, but at the ViewChild won't work because it's not part of the view, it's part of the content which is why we also have separate hooks here ContentInIt and ViewInIt.
* The great thing, is we don't only have at ViewChild we also have at ContentChild, which also needs to be imported from at Angler Core.
* This now also allows us to pass a selector, so the reference name contentParagraph would be the selector, which shows here in the app component HTML file.
* And then just like with at ViewChild, we get we can store this in some property, which will be of type ElementRef, like this, and now we can use it.
* And just like with ViewRef, we of course can't access the value or anything before we reached ContentInIt.
* So that is something I just wanted to add at ContentChild to get access to content, which is stored in another component, but then passed on via NG content.
* Now let me log this, and I will log it here in OnInIt to demonstrate that it is not there.
* Here, I will simply output Text Content of paragraph, oops, paragraph, and access this paragraph nativeElement textContent.
* And then I will copy this to AfterContentInIt.
* Now, once I save this and reload the application, you see that the first time we call it it's empty, hereafter OnInIt but thereafter an NG AfterContentInIt or after this has been called, once this hook has been called we see just the test.
* So now it was initialized.
* That is the missing piece, understanding how we can get access to elements in our DOM and how we can work with these lifecycle hooks.