**73. Using Local References in Templates:**

* -: Back in the application we built throughout the first lectures of this course with our cockpit anti server element component, with using output and input to pass data around our own custom property and event binding.
* Now, in the cockpit, right now I'm using two-way data binding to get the server name and content.
* Now there's nothing wrong with it, but maybe we don't need to use two-way data binding.
* *Since I only want to save or use the data at the point of time I click the button the ad server or the ad server blueprint button, it would be enough to get the value of the input at this point of time, and there is a nice way we can achieve just that.*
* So I'll duplicate this and comment out the old solution, and now I'll structure it out over a couple of lines and remove the two-way binding.
* *There and the other option we would have is to place a local reference on that element.*

**Local Reference:**

* A local reference can be placed on any HTML element, so not only on a input element, on anything you see here in the template, and you add with a hashtag and then a name of your choice, like for example, server name.
* Since this is, well, what does this reference will hold? A reference to this element.
* So maybe server name input.
* Now, important.
* *This reference, as I just said, will hold a reference to this element.*
* *So not to devalue, we entered there to the whole HTML element with all its properties.*
* We can see this, if we pass this now as an argument once we click at server on Add Server.
* *Here I can pass server name input because the other important thing you need to know about references, besides how to create them, is that you can use them everywhere in your template but, important, only there, not inside your TypeScript code****, only in the template****.*

Text

Description automatically generated

* But we do call the method here inside the template, of course.
* So here we can use server name input, and that is a way how we can pass it to the types of code.
* Because in on add server, we now know that we will receive the name input.
* You could also name it server name input.

Text

Description automatically generated

* And if we log this, we can analyze what this actually means, what we actually got here.

Text

Description automatically generated

* So let's save this and let our app reload.
* And, now if I test something and enter something here, and I click add server, of course, this was broken here, this is broken because we're not fetching the data correctly right now, you see we actually got the input element itself.
* So this SD element we got here that is what the local reference gives us, the element with all its properties.

Text

Description automatically generated

* So here we could also output value, the value of this input, since we know that a input element has a value.
* -: And of course this will depend on which kind of element you placed your local reference on.
* Not all elements have the same properties.

Text

Description automatically generated

* So if this, if we try this again, you'd see we now print tests because we access the value of this input to which we got access through the local reference.
* So, local references, a very nice feature to get access to some elements in your template and then use that either directly in the template, you could also output server name input value here, or you can pass it on like we do to use it in a type script code, a nice and handy feature.
* Now with that being passed, we can use it here to not only log our data to the console, but instead, here, once we define or once we create the server, we could access name, input, value.
* And we should also be explicit about the type.
* We know that this will be a HTML input element, so we know that this value property will be there.
* Same can of course be done on the blueprint, but we can also replace this new server name with name input and then value.
* And with that we can get rid of new server name.
* We don't need it anymore.
* I will comment it out, so that we still have it there for reference, but the new approach now only uses this local reference.
* Now for this to work, we, of course, also need to pass this reference to our onAddBlueprint method.
* So these are local references.