So, in the last video, we introduced constructor's well, in this one, we need to fix the error that introduced in the constructor to our class as brought about.

And we're going to introduce self-referencing.

Here we go, because we've named our constructor parameters the same name as our attributes in our class, whatever they are, they are that they are, then we created an ambiguity.

The system doesn't know what we're referring to on the left and right side of this equal sign. So, we must qualify this, and we can do this using the self-reference. It's normally the arrowhead that I call it.

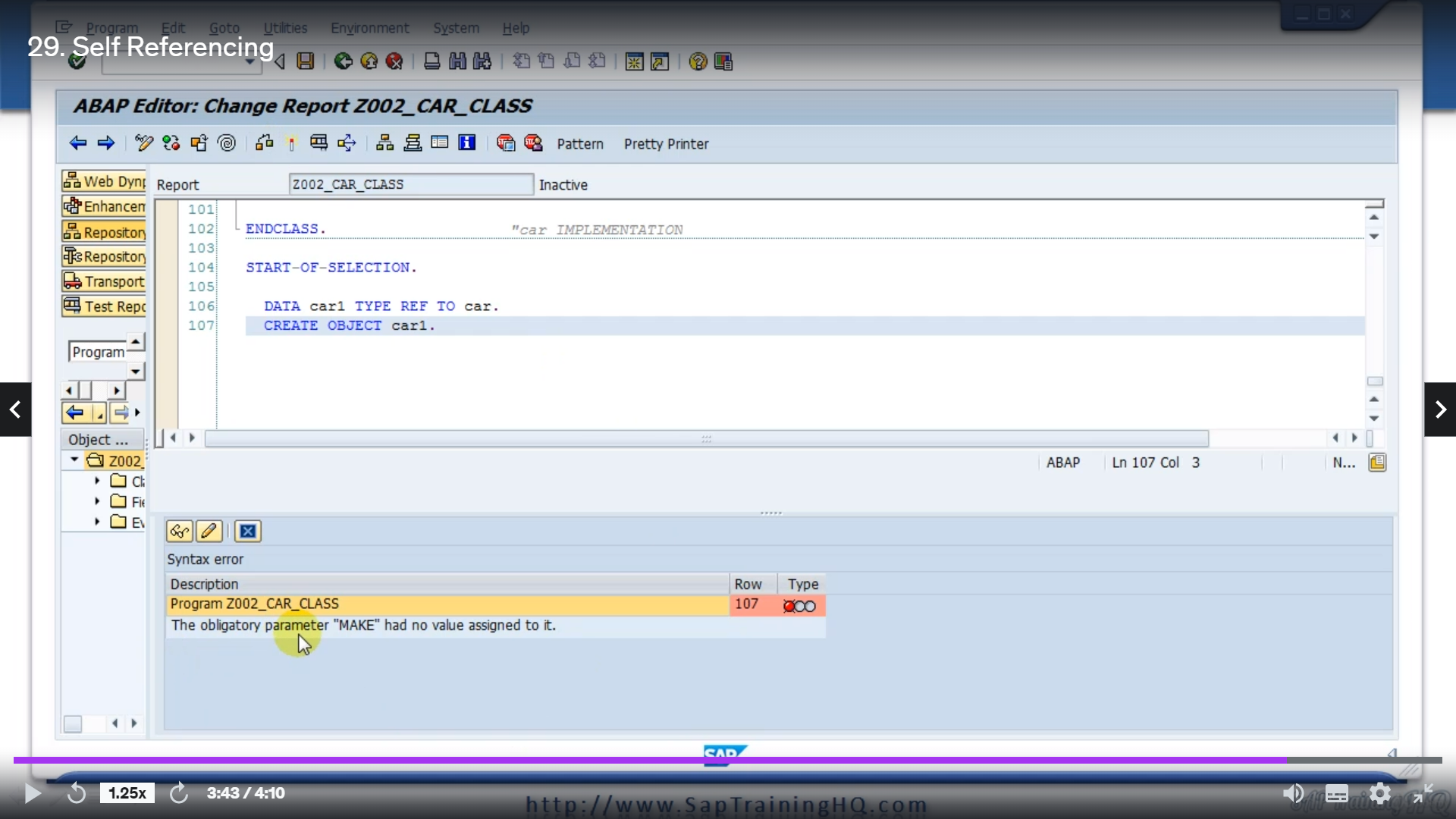
Now, normally, I would say first thing, don't call your constructors or any other method parameters the same name as attributes within your class because it does cause ambiguity. It causes errors like this to happen.

But we're doing it in this case so that we can see how the self-reference works.

Well, let's flip back to the slide and to use the self-reference in syntax.

What we do, we use it to refer to an object attribute.

So, in our case here, what we need to do is add the ME, then the arrowhead and then refer to the attribute name.



So, we got me Arrowhead then make and that refers to the attribute of the object.

So now the system will know that this make refers to the perimeter of the constructor.

Everything should work, the constructor will run, and this code will run so that they may contribute.

Gets updated with what the value that was imported in our constructor for parameter make and then we got model and so on.

But the thing is, it's still not going to work now.

I'll save the code if I do.

A syntax check was still going to get errors.

The obligatory parameter make had no value assigned to it.

And this is because now when we're trying to create our object, the system now knows that we must pass in those four parameters at the time when we're creating the object.

So, I'm going to pause this video and we'll have a separate one where we go back, and we extend the create object statement.