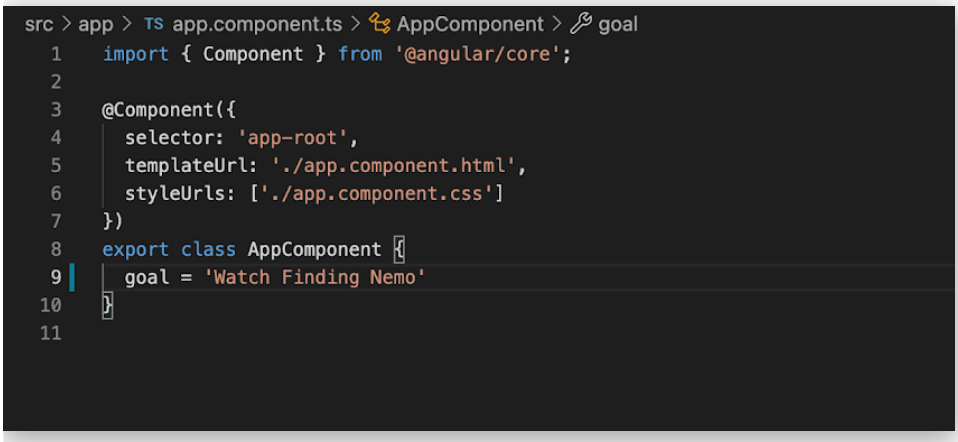
Monday: Updating App Component

Now that we have an idea of what's happening behind the scenes in our Angular App. Let's start building our Goals App properly.

So far all the content that's rendered on the browser came from Angular. We'll start by replacing that with our own custom code.

Let's update our app.component.ts

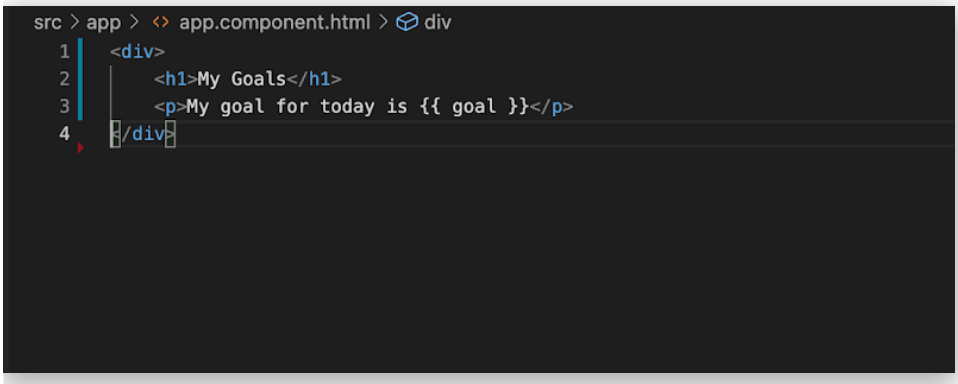
app/app.component.ts



Here, we've created a new variable, `goal`, and gave it a value of `'Watch Finding Nemo'`

Next, let's also change the HTML code in the app.component.html.

app/app.component.html



We have deleted the initial content in the file and created a div and inside it, created a <h1> tag and a <p> tag. Inside the <p> tag, we've used string interpolation to access the goal variable we created in the app.component.ts file.

If your server is running, switch over to the tab in the browser to see the changes we've just made.

Displaying an Array

Typically, we all have more than one goal to achieve. Let's create more goals in an array and then display them in our Goals app.

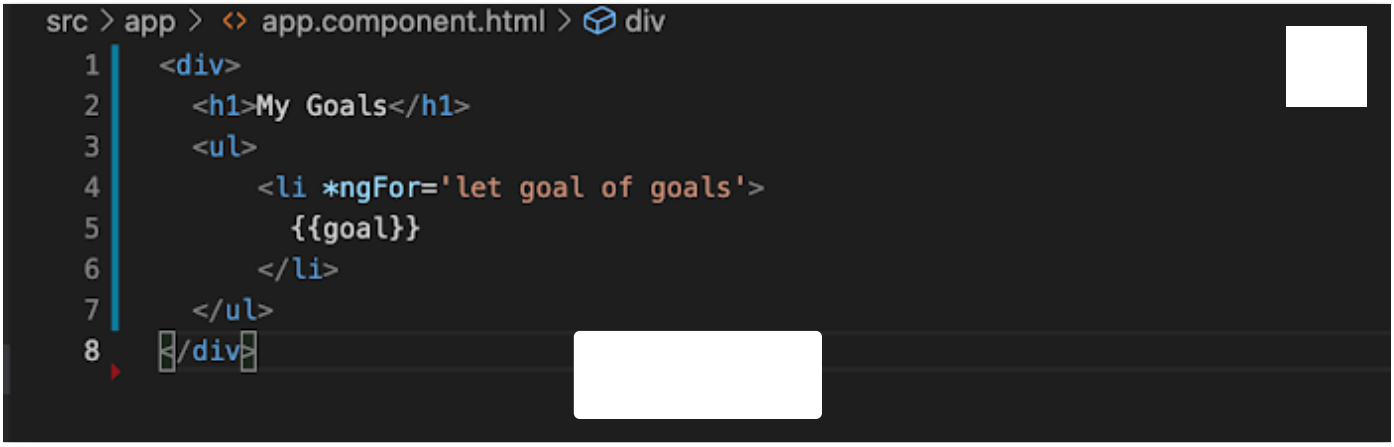
We have created a property, `goals`, and attributed it to an empty String array. We have then created a **constructor** function and given the empty goals array 3 string values so it is no longer empty.
A **constructor** function defines the logic that should be executed once the class is instantiated. In this

case, when the *AppComponent* is initiated, it creates an instance of the goals array with the 3 values that we have assigned it.

ngFor Directive

Let us now display these goals.

app/app.component.html



We have created an unordered list with a list tag that has some logic, the *ngFor directive. The *ngFor is an angular **repeater directive** that loops through the host element goals which is a list. Don't forget the asterisk sign *before ng, it's part of the syntax. The directive goes through the goals array and assigns each item in the array the variable goal. We have then displayed the goal variable inside the list tag using Angular's interpolation binding syntax.

This directive works the same as the for loop in Javascript that we learned in Moringa Prep. An excerpt of the loop syntax used we used in Javascript is:

var arr = [3, 5, 7,11,13];

for (var i in arr){

console.log(i);}

The directive gives a similar output as this loop, only that it now has its own syntax that's different.