AngularJS 2.0 Hello World

Plunker link: http://bit.ly/2bQc3fV

In this exercise we are going to get a simple hello world style web app.

The main file we will be using will be app/app.component.ts. First open the plunker link. You will see that we have an Angular app all set up and ready to go.

Open the app/app.component.ts file. As we see we have a simple angular component that just shows the text “Hello World” in an h1 tag. We want to achieve having a name that was entered in an input box show in the h1 tag.

Remember that in angular, the view and the class are tied together through data binding. That is anything we define on the class will be available in the template. Let’s start by adding a name variable on our AppComponent class like so

```
...
export class AppComponent {
  name: string 'Joe';
}
...
```

This will set up a variable called ‘name’ with the value ‘Joe’ that we can access from the template. Now we need to add this variable to our template. The problem is how do we tell angular that we want it to use the variable and not just some text. Luckily angular has the expression evaluator syntax {{}}. Anything between these double curly brackets will be evaluated by Angular. So if we want Angular to display our ‘name’ variable we need to put it between the expression evaluators. Add `{{name}}` to our h1 content in our template (replacing the text “World”) so that we end up with this:

```
@Component({
  selector: 'my-app',
  template: `
  <h1>Hello {{name}}</h1>
  `
})
...
```

And now we have our JavaScript variables binding to our view!

While this is great we could do with a bit more functionality. Let’s add an input box so that users can enter their name and get a personalised hello message! Add the following to your template

```
<input type="text" class=form-control" placeholder="Enter
 your name"/>
```

This is just a basic html input element. We now need to tie this to our greeting message. Currently we are showing the name variable. So if we want dynamic text in our greeting message we need to change the value of that variable. To do this Angular has the ngModel concept. When we use ngModel we bind the element that it is defined on to the object we pass it. Let's add it to our app to get a better understanding. Add `[(ngModel)]="name"` to our input box. You should end up with something like:

```
<input [(ngModel)]="name" type="text" class=form-control"
 placeholder="Enter your name"/>
```

You should now see the value 'Joe' in the input box. Go ahead and enter in something else and see what happens. You should see the greeting message change! This is because the ngModel directive binds the object we passed to it (the 'name' object) to the element we defined it on (the input we added) and angular will automatically watch for changes and update both when anything changes.

Challange

It looks a bit funny when there is no name and it is just saying hello. See if you can hide the hello message until there is a name in the input box.

Hint: We seen a very helpful directive we could use here from presentation.

Resources

Plunker link: http://bit.ly/2bQc3fV

Plunker tutorial complete link: http://bit.ly/2bwl5wn