Learn Angular

BY MAKING A QUIZ APP FROM SCRATCH

HUNGRYTURTLECODE.COM

Part 9

Check Out The Whole Course Index

Mocking An API Request With Angular Factories

We have already covered how to build a basic factory when we built the quizMetrics factory in a previous part. In this part, we will build another factory; this time to mock data coming from an API. Let's continue using Angular factories to further separate concerns.

Earlier, we mentioned that we won't be using an actual API in this application but will instead copy and paste the JSON into the scripts. This is what we did for the list data in the list controller.

The only problem with that is that now we have the JSON copied into the list controller and now if we followed that same idea we would copy the quiz JSON into the quiz controller. Things are starting to get messy.

To solve this problem we will create a factory and copy all of the data into that instead (including refactoring the data out of the list controller and into the factory). That way our controllers have no knowledge of where the data is coming from, just that it is getting the data.

This way, we can copy the data in for now, but at a later date we can actually make API requests and put all the logic into the factory and the controller still receives the data in the exact same way. We won't have to touch the controllers to make this change. Can you see how this allows our application to grow over time?

We don't have to completely refactor the entire codebase to

allow API calls, which is something we would have to do if we copied the data into each controller separately. We now put all that data into a factory and separate all of our concerns. It's all coming together nicely.

If you want to see the app for yourself, check it out here.

The git repo can be found here.

Get On With It!

Ok, enough chat, let's write some code.

Click the image to go to the video on youtube or go here https://www.youtube.com/watch?v=TlR3bI7Azvk&list=PLqr0oBkln1FBmOjK24_B4y_VAA8736wPq&index=9

The next part can be found here

We will create another script in the factories directory and call it dataservice.js. We will start this in the exact same way we started the quizMetrics factory.

```
(function(){

    angular
        .module("turtleFacts")
        .factory("DataService", DataService);


    function DataService(){

        var dataObj = {
            // We will add properties to this object soon
        };

        return dataObj;
    }

})();
```

Now we will add the object we will return and start adding in some of the data that we need. The turtlesData variable that contained the JSON from our list controller will be copied over into this factory and we will create a new variable to hold the JSON for the quiz questions.

```
function DataService(){

    var dataObj = {
        turtlesData: turtlesData,
        quizQuestions: quizQuestions
    };

    return dataObj;
}
```

We can see that we have created some entries in the dataObj which reference variables that contain JSON that we have copied into the script at the bottom. This is then returned from the factory so that our controllers can use this data.

The turtlesData is the exact same JSON as we used before. The quizQuestions is new JSON which can be seen here:

```
var quizQuestions  = [
        {
            type: "text",
            text: "How much can a loggerhead weigh?",
            possibilities: [
                {
                    answer: "Up to 20kg"
                },
                {
                    answer: "Up to 115kg"
                },
                {
                    answer: "Up to 220kg"
                },
                {
                    answer: "Up to 500kg"
                }
            ],
            selected: null,
            correct: null
        },
        {
            type: "text",
            text: "What is the typical lifespan of a Green Sea Turtle?",
            possibilities: [
                {
                    answer: "150 years"
                },
```

Taking a closer look at the JSON for each quiz question we see that it has a type (text, or image) which allows us to have different styles of questions, the text of the question itself, four possible answers and two flags called selected and correct which are initialised to null. We will discuss all of this more later.

Angular Factories leads to Dependency Injection

We have seen it all before. So we will just go back to our controllers and add the “dataservice” factory to the array of dependencies. Also adding it as an argument to the controller functions.

Here is the quiz controller:

```
(function(){

    angular
        .module("turtleFacts")
        .controller("quizCtrl", QuizController);

    QuizController.$inject = ['quizMetrics', 'DataService'];

    function QuizController(quizMetrics, DataService){
        var vm = this;

        vm.quizMetrics = quizMetrics;
        vm.dataService = DataService;

    }

})();
```

We also do the same in the list controller. While we are in there we can delete the turtlesData variable that we now have in the factory (if you haven’t already removed it).

```
(function(){

    angular
        .module("turtleFacts")
        .controller("listCtrl", ListController);

    ListController.$inject = ['quizMetrics', 'DataService'];

    function ListController(quizMetrics, DataService){

        var vm = this;

        vm.quizMetrics = quizMetrics;
        vm.data = DataService.turtlesData;
        vm.activeTurtle = {};
        vm.changeActiveTurtle = changeActiveTurtle;
        vm.activateQuiz = activateQuiz;
        vm.search = "";

        function changeActiveTurtle(index){
            vm.activeTurtle = index;
        }

        function activateQuiz(){
            quizMetrics.changeState(true);
        }
    }

})();
```

Notice that we previously had a line **vm.data = turtlesData** but of course, the turtlesData variable doesn't exist in the controller, it is now inside the dataservice factory. So we change that line to **vm.data = DataService.turtlesData** .

Before we forget, we need to add the new dataservice factory script to the tags at the bottom of our HTML. The scripts area of the HTML now looks like this:

```
<script src="js/app.js"></script>
<script src="js/controllers/list.js"></script>
<script src="js/controllers/quiz.js"></script>
<script src="js/factories/quizMetrics.js"></script>
<script src="js/factories/dataservice.js"></script>
```

Onwards And Upwards

In part 10 we will use the quiz JSON we now have access to and start creating the bootstrap markup for the quiz.

See you over there.

Adrian