Exercises

In these assignments weare going to makean Auto Applicatie. Save optional exercises for when there's time left. Difficulty of an optional assignment is indicated by a rating of 1 to 5 stars.

1. Introduction

a. Create your first project using the Angular CLI

- You install the CLI globally in a command prompt by means of:

  npm install @angular/cli -g

- Navigate to the directory where you want to create your project directory.
- Give the command ng new<projectname>

b. Customize the app.component's HTML to:

- Contains a header (H1)
- Contains a disordered list (UL)

c. Try to understand how the app works,

see app.component.ts, app.module.ts, main.ts and index.html

d. Optional (*):

Are you ready in time? Then read a piece about TypeScript (data types and classes) at:

https://www.typescriptlang.org/docs/home.html

2. Databinding

a. Consider at least 3 characteristics of a car and process these properties in a model.

b. In the app.component.ts, create a variable 'myAuto'.

Also create a constructor and initialize the variable in it

by creating a new instance of the car model.

c. Show one or more features of the car in the HTML

using Simple Data Binding.

d. Add an image to the model as a string property

and enter it as a URL for your existing car. Show this image

by attribute data binding of the src attribute.

e. Now create an array of cars, instead of a single 'myAuto'.

Also, fill the array with at least three cars. Show this array of cars

in the HTML in the list (UL of LI's) created in a previous assignment;

use the *ngFor directive for this.

f. Create a detail div (or other container element), in which the details of a selected car can (later) be shown.

g. Make sure by means of event binding that after clicking on a car from the list,

the details of this car are shown in the detail block. The block serves

only to be shown if a car is selected (use *ngIf).

Hint: use a variable that keeps track of the current car.

h. Optional (****):

Make sure that you can also select a current car with the keyboard.

Think of high-lightening a selected car using the arrow keys,

but also being able to confirm by means of a test.

3. Services

a. Create an autoService and use the providedIn notation or choose to name in the app.module.ts under 'providers: []'

b. Inject the service into the component that the service will use.

c. For now, create the collection of cars in the service, instead of in the component

4. Observables and RXJS

a. Create a JSON file containing the collection of cars you created in the service

b. Create a new method in the service that reads the data from the JSON file and returns an Observable.

c. Fill in the collection of cars in the service, by subsribing to the Observable in the constructor of the service.

d. (optional): Map the data in a pipe() function, so that you give the properties different names, for example. Do this in the method of the service that yields an Observable.

5. Multiple components

6. Routing

7. Forms