Angular LAB and Homwork (Day 20)

Forms Testing Animation Material and PrimeNG libraries

Forms

In the ongoing project, create a component for entering new people. Use the template-based approach for wiring up a person model object.

To simulate saving simply either display an alert when the form is submitted or console.log the values.

Create another editing component, but use the reactive model for building the form.

Create an class that has several properties, then create an editing component for that class. Use the FormBuilder to shorten the creation of the form.

create a button that is not part of the form. Code the button so that it will change the data in the model when pressed. With the form open, press the button to ensure that the data is refreshed on the form.

Testing

Create a new project. From the command line run the "ng test" command. If there are errors concerning a type.d file it is most likely a version issue between TypeScript and Jasmine. In the package.json file make sure the TypeScript reference is set to "2.9.0". You may have to do an "npm install" to get the version to take, and also possibly a cache clean and then an "npm install".

Once the tests are running successfully, first change the title of the page. Notice how the tests run immediately and should be broken. Change the test so that it passes.

Stop the testing by pressing Ctrl-C in the command window where the tests are running.

create a new service component. Review the .spec file that is generated. Run the test framework again to make sure that the service tests are seen and run.

Add some logic to the service and then write a test to validate the logic.

TDD is the process of writing tests BEFORE writing the code. So now, write a test for some functionality that doesn't yet exist in the service. It could be something like get person's full name. Obviously since the function doesn't exist yet the test will not even compile, let alone pass. Now code just enough functionality in the service to make the test pass. For example, if the expectation is for the method to return the name "Tom", simply return the hardcoded string "Tom". Once it passes, refactor it to actually get the data from an actual source (fake or otherwise). Refactor/Run Tests until everything passes again.

Congratulations - you just did a very small case of TDD, aka "Red-Green-Refactor."

Create a new service component WITHOUT the automatically generated .spec file. Run "ng generate -help" to get the proper syntax for the option.

Create tests for the new service by manually creating the spec file.

Animation

In the AppModule, import the BrowserAnimationsModule.

Then, create a new component simply to test the process to create animations.

In the new component import the animation functions you will need (trigger, state, style, animate, transition).

Add the animations property in the @Component declaration, create an animation to transition between to different states on a small area of the screen, such as a small div. Control the state via a simple button.

Add logic to create more states via an enum - at least 3. Change the logic to switch states; create separate buttons for each state such that clicking the button will set the state property.

Material and PrimeNG

Create two new projects - one called Mat-Test and the other called Prime-Test. Either use the example page by removing all of its content, or create a new test-component. Using the documentation on each site, create instances of the following in each project:

- Button
- Input
- Select
- Menu with Menu Items
- Tabbed page
- Data Table

In each case, experiment with the different styles and options. For instance, create a normal button, a rounded button, a button with an icon, etc.

What controls are unique to a particular library?

Are there any that are much easier in one library?