Exercise (Instructions): Angular Testing

Objectives and Outcomes

In this exercise, you will learn about unit testing in your Angular applications. You will learn about Angular test environment, Karma, and Jasmine and learn how to use them to carry out unit testing. At the end of this exercise, you should be able to:

- Configure your application to conduct unit tests
- Set up unit tests using Jasmine and carry out the unit test automatically

Set up Project for Running a Single Test

- Open test.ts file in the Project's root folder and update the following line. This will enable you to run a single test on a specific component rather than all the tests.

```
const context = require.context('./', true, /menu\.component\.spec\.ts$/);
```

- Start up the test using the Angular CLI test support by typing the following at the command prompt:

```
ng test
```

- You will notice that Karma will open a browser and run the tests and show a number of errors on the console. You need to update the test file as shown in the next step.

Set up the Test File

- Open menu.component.spec.ts and update it as follows:

```
. . .
import { RouterTestingModule } from '@angular/router/testing';
import {BrowserAnimationsModule} from '@angular/platform-browser/animations';

. . .

import { MaterialModule } from '@angular/material';
import { FlexLayoutModule } from '@angular/flex-layout';
import { Dish } from '../shared/dish';
import { DishService } from '../services/dish.service';
import { DISHES } from '../shared/dishes';
import { baseURL } from '../shared/baseurl';
import { Observable } from 'rxjs/Observable';

. . .

  beforeEach(async(() => {

    let dishServiceStub = {
      getDishes: function(): Observable<Dish[]> {
        return Observable.of(DISHES);
      }
    };

    TestBed.configureTestingModule({
      imports: [ BrowserAnimationsModule,
        MaterialModule,
        FlexLayoutModule,
        RouterTestingModule.withRoutes([{ path: 'menu', component: MenuComponent
          }])
      ],
      declarations: [ MenuComponent ],
      providers: [
        { provide: DishService, useValue: dishServiceStub },
        { provide: 'BaseURL', useValue: baseURL },
      ]
    })
    .compileComponents();

    let dishservice = TestBed.get(DishService);

  }));

. . .
```

- Next, add a new test to the list of tests as follows:

```
  it('dishes items should be 4', () => {
    expect(component.dishes.length).toBe(4);
    expect(component.dishes[1].name).toBe('Zucchipakoda');
    expect(component.dishes[3].featured).toBeFalsy();
  });
```

- Next we will add another test to check if the template is correctly getting the information for rendering the view:

```
. . .
import { By } from '@angular/platform-browser';
import { DebugElement } from '@angular/core';

. . .

  it('should use dishes in the template', () => {
    fixture.detectChanges();

    let de:      DebugElement;
    let el:      HTMLElement;
    de = fixture.debugElement.query(By.css('h1'));
    el = de.nativeElement;

    expect(el.textContent).toContain(DISHES[0].name.toUpperCase());

  });
  . . .
```

- Save all the changes and do a Git commit with the message "Component Test".

Conclusions

In this exercise you have learnt to use Jasmine, Karma, Angular test support and Angular CLI to test a component.

Mark as completed