SECTION 01 - COURSE KICKOFF

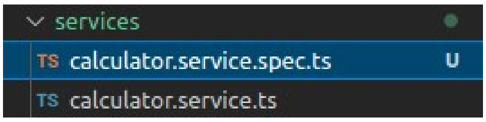
Index

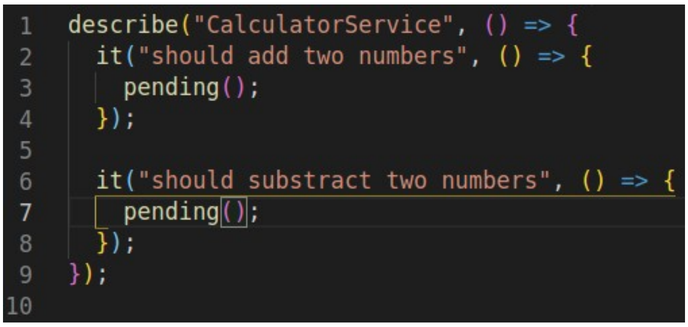
- Initial commands

git clone git@github.com:angular-university/angular-testing-course.git
git checkout 1-start
(terminal tab 1) npm install
(terminal tab 2) npm run server -> start backend app
(terminal tab 3) npm start -> start frontend server
(from the browser) localhost:4200

- Quick introduction to Jasmine

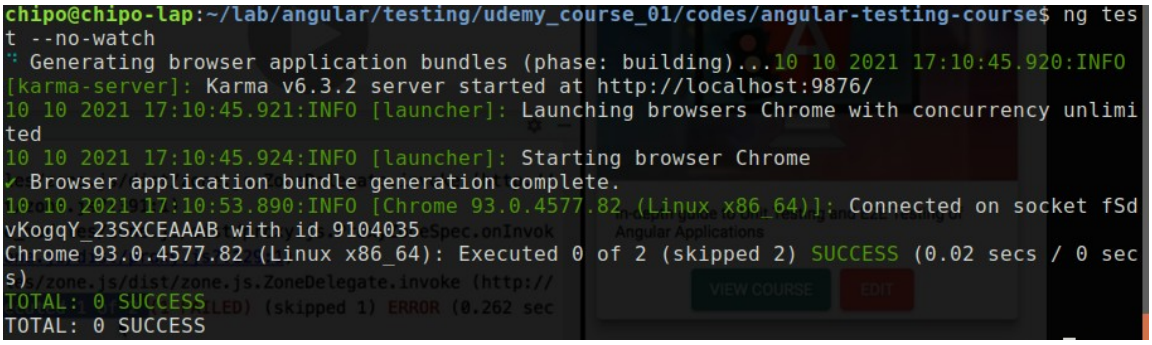
- **Jasmine** is a Behavior Driven JavaScript testing framework.
- In Jasmine, our tests are known as "specifications" or "specs", and they are grouped in test suites.
- A specification is simply a functional test. So a specification is meant to test a particular functional feature of our application.
- Karma is used internally by the Angular CLI.
- **ng test** launches an instance of the Chrome browser where the tests are going to be executed.
- In order to run our tests without hot reload mode, we run "ng test –no-watch".





```
describe("CalculatorService", () => {
  it("should add two numbers", () => {
    pending();
  });

  it("should substract two numbers", () => {
    pending();
  });
});
```



```
chipo@chipo-lap:~/lab/angular/testing/udemy_course_01/codes/angular-testing-course$ ng tes
t --no-watch
⠙ Generating browser application bundles (phase: building)...10 10 2021 17:10:45.920:INFO
[karma-server]: Karma v6.3.2 server started at http://localhost:9876/
10 10 2021 17:10:45.921:INFO [launcher]: Launching browsers Chrome with concurrency unlimi
ted
10 10 2021 17:10:45.924:INFO [launcher]: Starting browser Chrome
✔ Browser application bundle generation complete.
10 10 2021 17:10:53.890:INFO [Chrome 93.0.4577.82 (Linux x86_64)]: Connected on socket fSd
vKogqY_23SXCEAAAB with id 9104035
Chrome 93.0.4577.82 (Linux x86_64): Executed 0 of 2 (skipped 2) SUCCESS (0.02 secs / 0 sec
s)
TOTAL: 0 SUCCESS
TOTAL: 0 SUCCESS
```

- Create a mock

- Crate a mock: jasmine.createSpyObj("LoggerService", ["log"]);

  Where LoggerService is the actual class to mock and the second parameter is the list of methods that are called from the service.

- And based on the mock we can assert something like:
  expect(logger.log).**toHaveBeenCalledTimes**(1);

- TestBed

- **TestBed** provides the dependencies to our services by using dependency injection instead of calling constructor's explicitly.

  ```
  loggerSpy = jasmine.createSpyObj("LoggerService", ["log"]);

  TestBed.configureTestingModule({
   providers: [
     CalculatorService,
     { provide: LoggerService, useValue: loggerSpy },
   ],
  });

  calculator = TestBed.inject<CalculatorService>(CalculatorService);
  // calculator = new CalculatorService(loggerSpy)
  ```

- Disable the test suit

- Add an “x” before *describe*.
  xdescribe("CalculatorService", () => {

- The same applies to disable a specific test case:
  xit("should add two numbers", () => {

- Focusing

- Add an “f” before *describe*. So jasmin will focus on a certain test suit and it will disable all the rest.
  fdescribe("CalculatorService", () => {

- The same applies to focus on a specific test case:
  fit("should add two numbers", () => {