**Unit testing in angular**

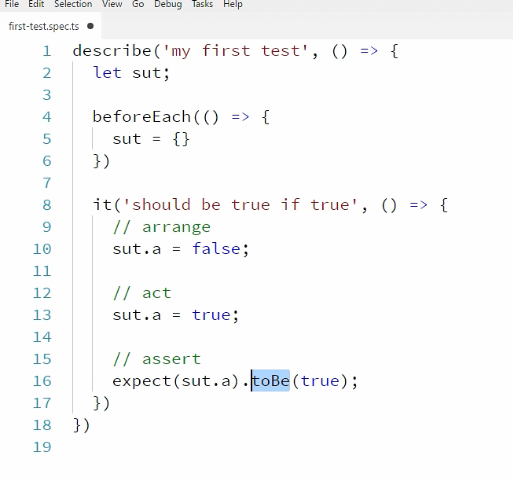
**Course Introduction:**

Download the project from below

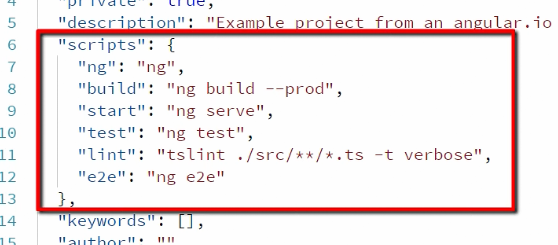


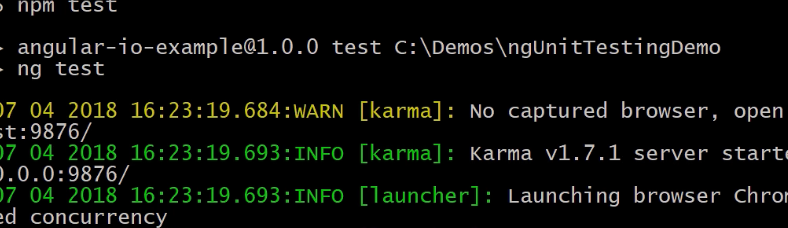
**First Unit Test :**

Create any spec.ts file in your project

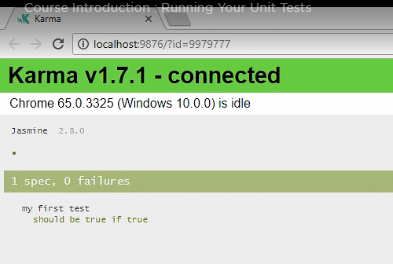


How karma picks .spec files?

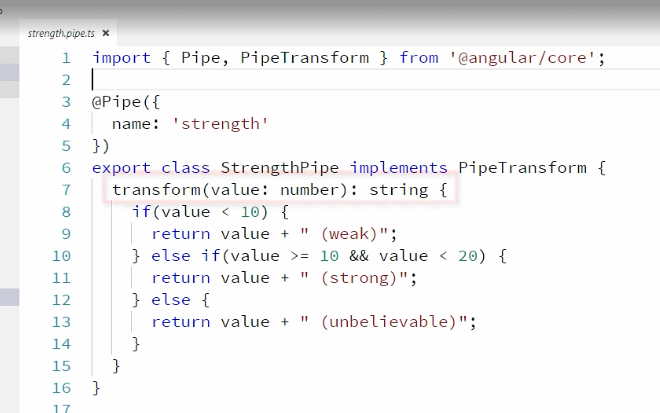


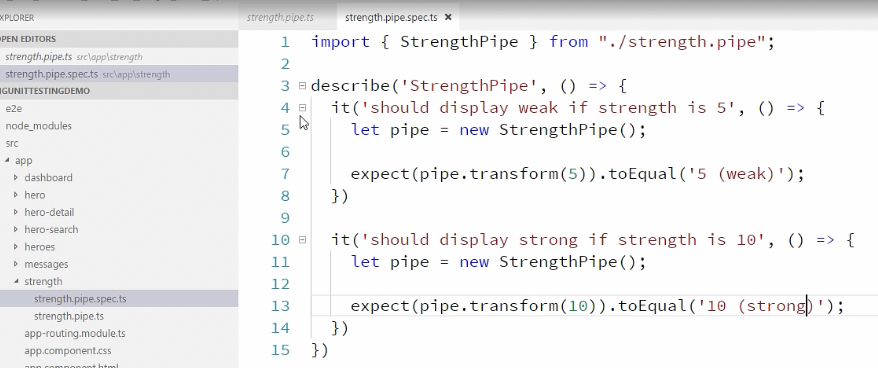


Output

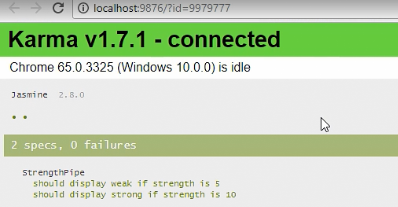


**Testing a pipe**

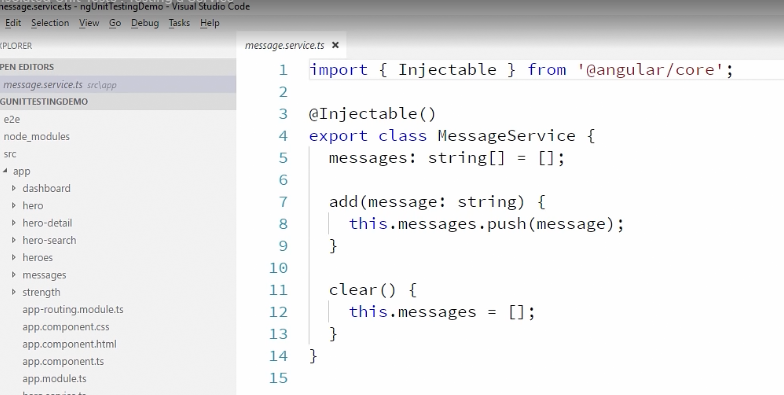


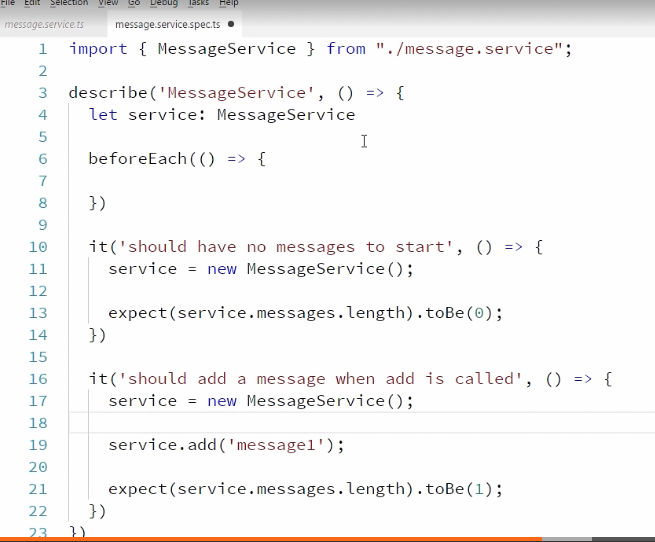


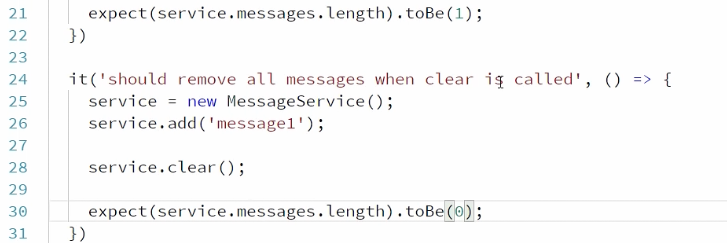
Run ng test



Testing a simple service:

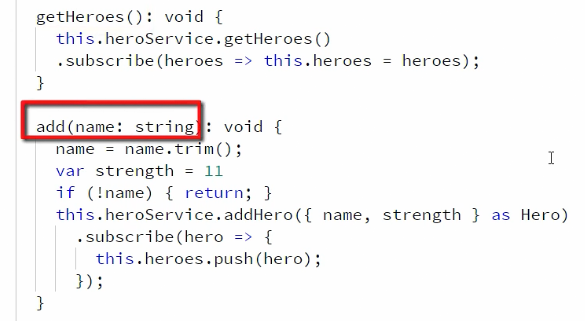


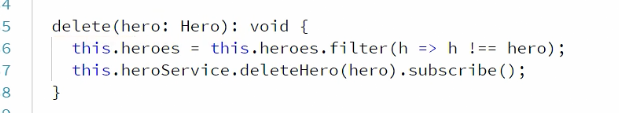




Testing a component :

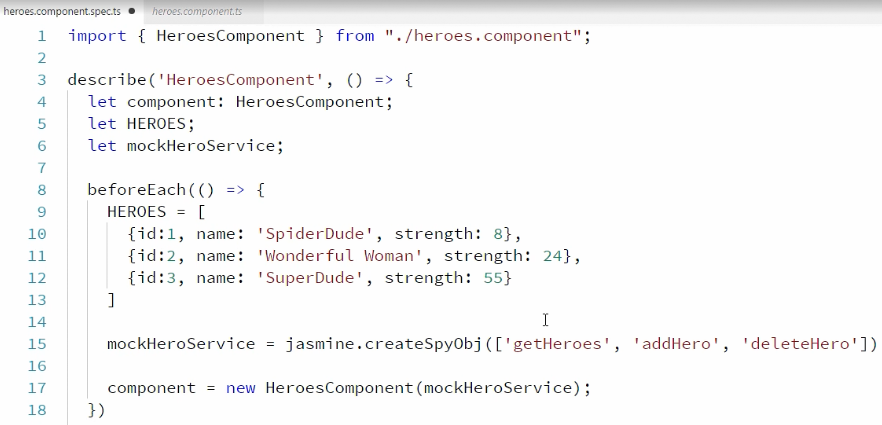


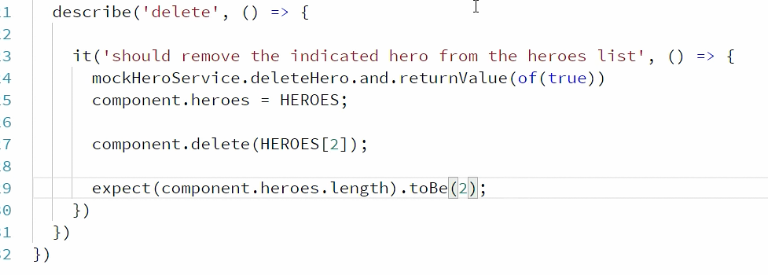




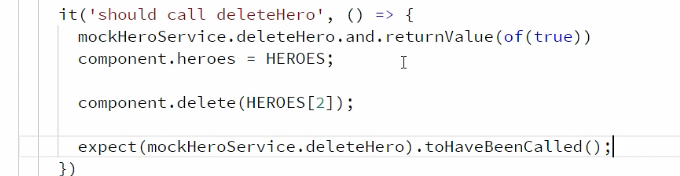
And now the test file

Import { of } from ‘rxjs/observable’ ;



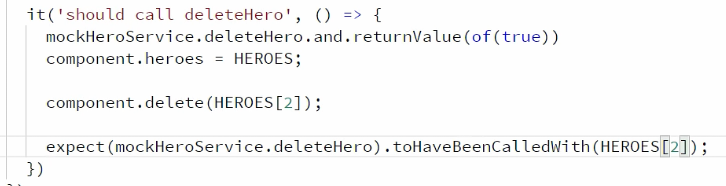


Another scenario is that if we want to check if heroService.deleteHero method is called in the component’s delete method. Below is the test case



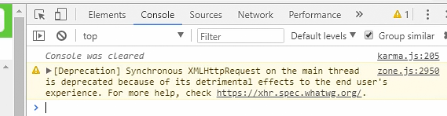
But in the component , delete Service method is called with parameter of type hero

So in this case,



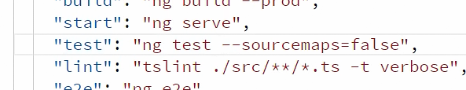
**Shallow integration tests:**

In console, we see following sourcemaps in console often

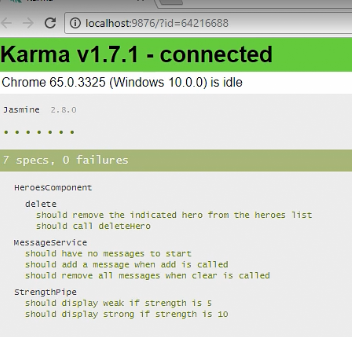


To remove those

Go to package.json and add



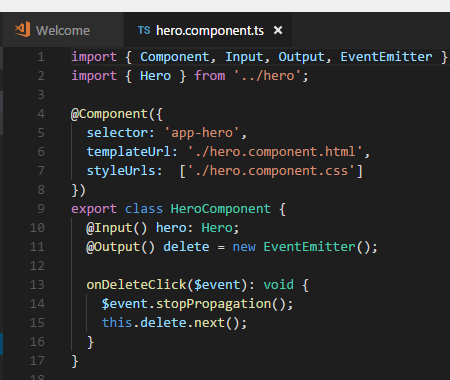
Output of ng test



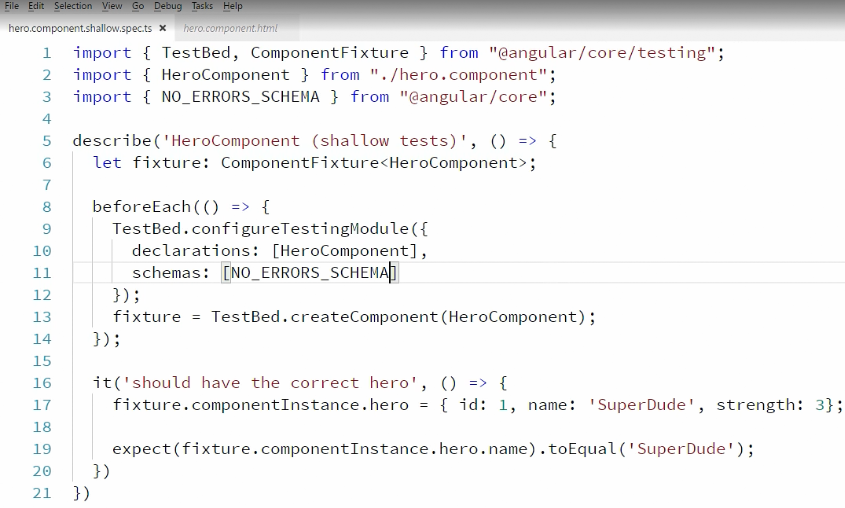
**For shallow integration tests, we use TestBed**

It allows us test the component along with its template.

Hero component



It has input hero which we need to set



You can notice we have added schemas : NO\_ERROR\_SCHEMA (from @angular/core) to ignore hero.html file.

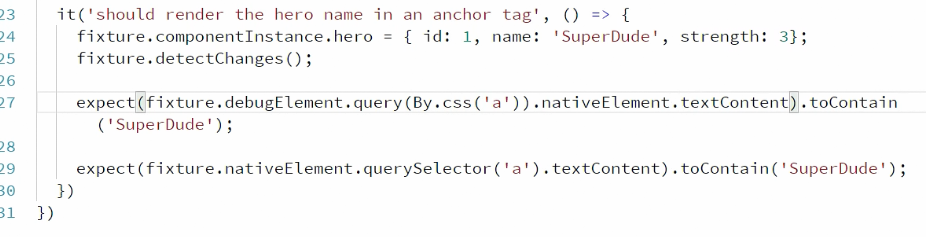
But it is not a good practice. Ignore html code scan is incorrect.

For example: User can type <za instead of anchor tag <a

To check a particular template, we can use the below 2 way

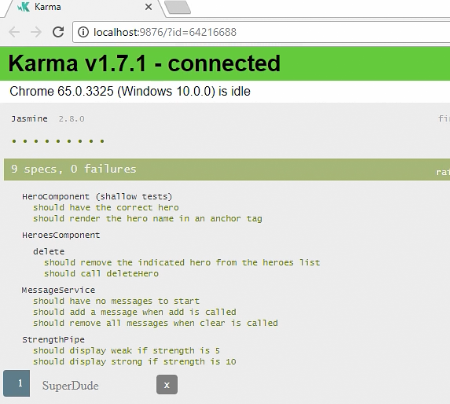
1: native element

2: debug element



‘By’ is from angular/ platform-browser

Karma status:



For hero component , there was no provider in the td constructor . Html didn’t have any child component

But in case of Heros.component

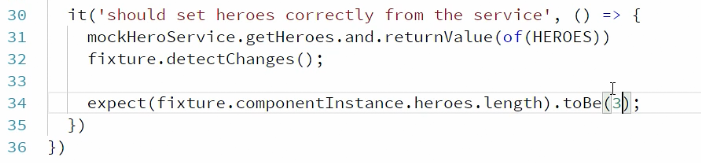
We have hero service registered in the constructor

And in heroes.html we have



Lets do the test file:





In above example , we have used NO\_ERRORS\_SCHEMA and avoided html scan.

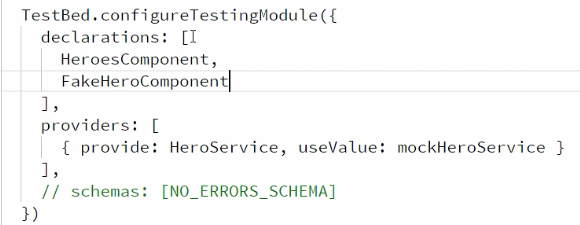
But inorder to scan, we need to comment this line.

Now in this case , heroes html has app-hero selector tag. That means it has child component .

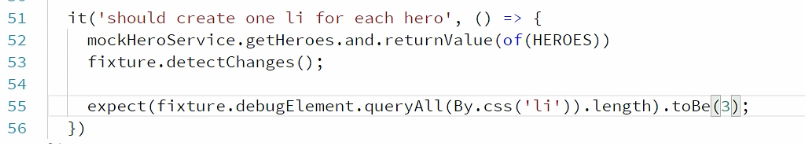
So we need to mock this component in our case.



And register in



A test case



Counts number of li tags in html

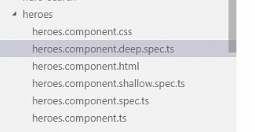
**Deep Integration testing:**

Shallow integration tests allow us test ts + DOM

It also allows to provide dependent but fake components just to complete the shallow case. But in deep integration , we are also checking the interactions between components.

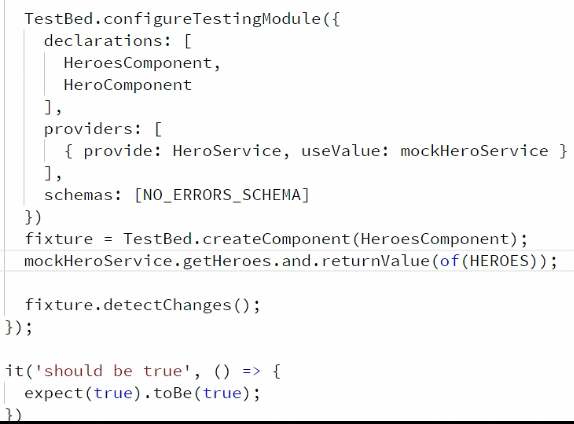
In our case, its heroes component + Dom and hero component

Create a file





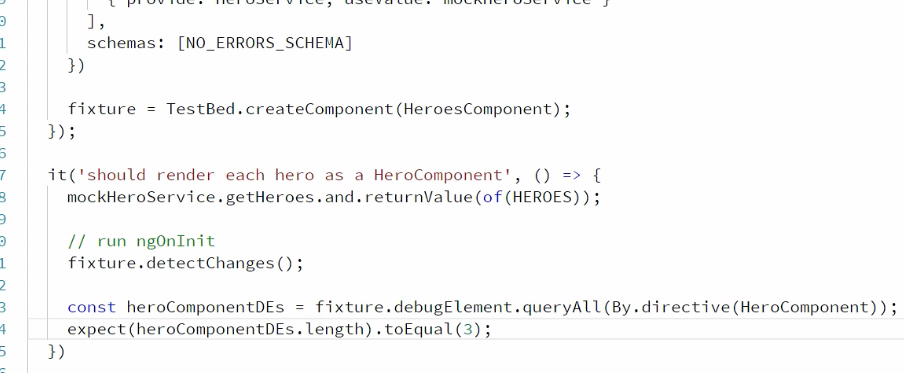




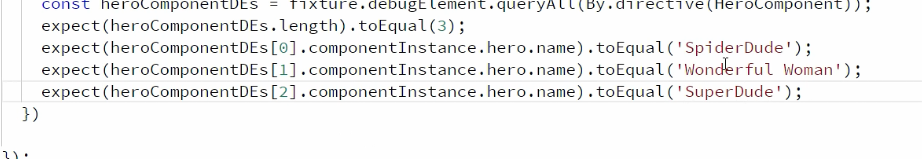


**Finding Elements directive:**

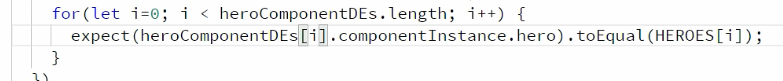
Make some changes in the above file:



You can have multiple expects

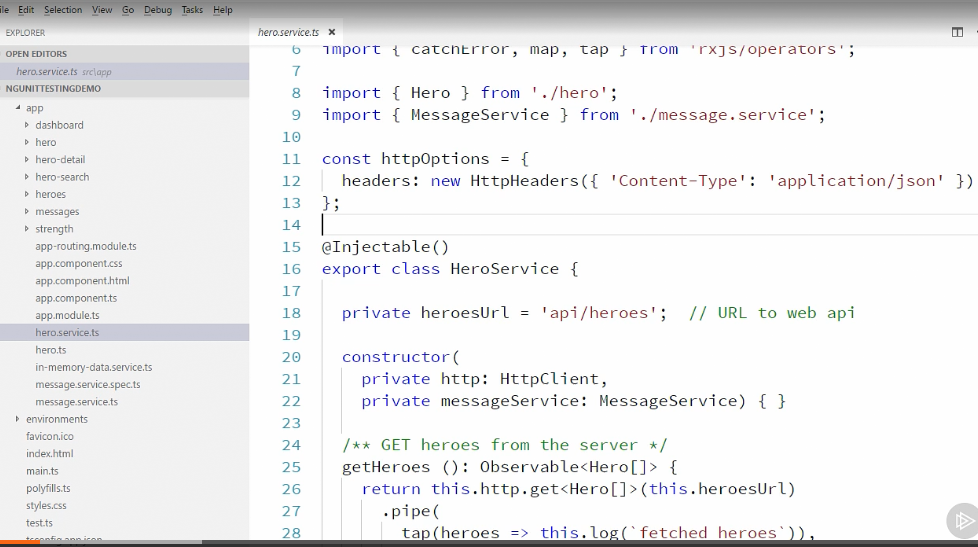


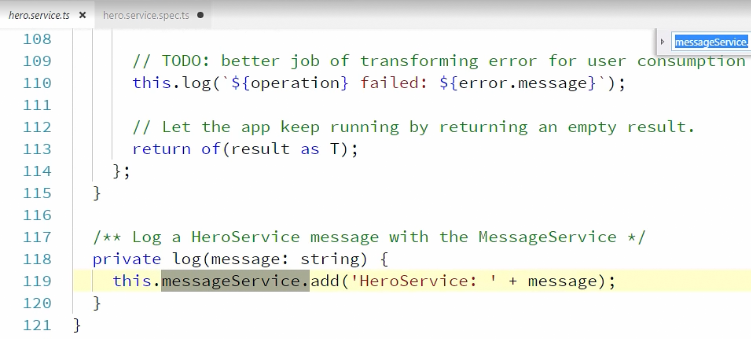
Using loop



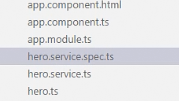
**Integration Testing of services:**

Consider below service

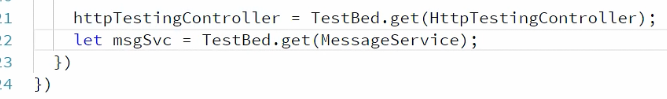




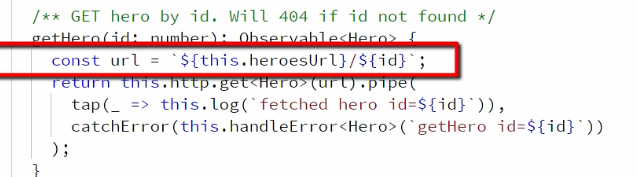
Create below file in app/ folder





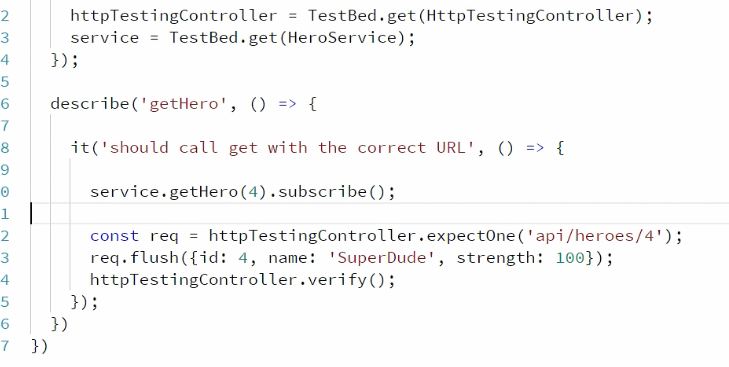


Lets test below method of hero service:

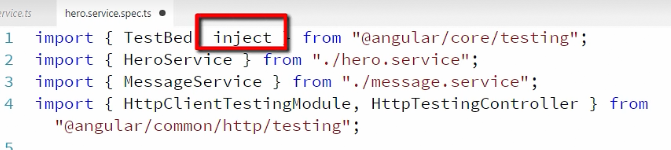


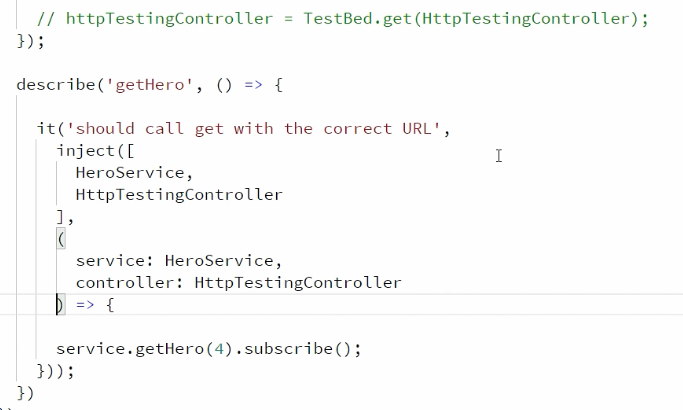
There are 2 ways to test:

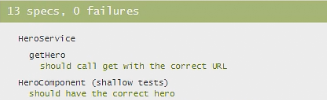
1. By using testbed instances:



1. By using inject function

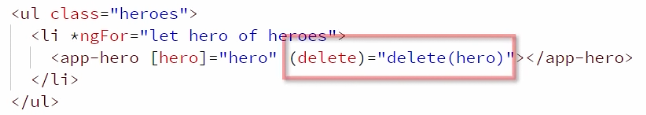




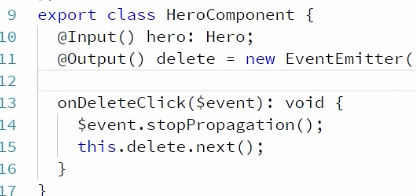


Testing DOM Interaction and Routing Components

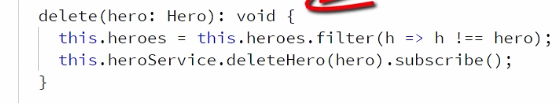
In heroes component html



In hero component , we have event propagation



In heroes component we have delete(hero) method



In Heroes spec deep test file, we write our case of delete as below

