Angular 8/9 Unit Testing using Jasmine & Karma

Unit testing plays a very important role when new features are being added in a working functional application, here it reduces the chances of bugs to the rest of the functionality due to the latest change.

What is Jasmine?

Jasmine is a Behavior Driven Development (**BDD**), open-source testing framework for JavaScript. It aims to run on any JavaScript-enabled platform, to not intrude on the application nor the IDE, and to have easy-to-read syntax. It is heavily influenced by other unit testing frameworks, such as ScrewUnit, JSSpec, JSpec, and RSpec.

Jasmine provides several valuable functions to write tests. Here are the main Jasmine methods:

- **it():** Declaration of a particular test
- **describe():** It's a suite of tests
- **expect():** Expect some value in true form

What is Karma?

Karma is a simple tool that allows you to execute JavaScript code in multiple real browsers. The main purpose of **Karma** is to make your test-driven development easy, fast, and fun.

If we talk about the type of functional testing, then there are mainly 3 types of testing.

A. Unit Testing

B. Integration Testing

C. End-to-End Testing

I assume that Node, NPM, and Angular CLI configured properly in your system.

Note: Karma currently works on Node.js 6.x, 8.x, and 10.x.

Latest version:

Jasmine: 3.6.1 [24-07-2020]

Karma: 5.1.1 [28-07-2020]

Let's Get Started

1) Create a new Angular application.
2) Create a component using the below command:

ng generate component post-list

Edit **post-list.component.ts**:

```typescript
import { Component, OnInit } from '@angular/core';

@Component({
selector: 'app-post-list',
templateUrl: './post-list.component.html',
styleUrls: ['./post-list.component.css']
})
export class PostListComponent implements OnInit {

constructor() { }
userName:String = "Pranam Bhat";
ngOnInit(): void {
}

}
```

Edit **post-list.component.spec.ts:**

```typescript
import { async, ComponentFixture, TestBed } from '@angular/core/testing';

import { PostListComponent } from './post-list.component';

describe('PostListComponent', () => {
let component: PostListComponent;
let fixture: ComponentFixture<PostListComponent>;

beforeEach(async(() => {
TestBed.configureTestingModule({
declarations: [ PostListComponent ]
})
.compileComponents();
}));

beforeEach(() => {
fixture = TestBed.createComponent(PostListComponent);
component = fixture.componentInstance;
fixture.detectChanges();
});
```

```
it('should create', () => {
expect(component).toBeTruthy();
});
});


it(`should have a userName 'Pranam Bhat`, async(() => {
  fixture = TestBed.createComponent(PostListComponent);
  component = fixture.debugElement.componentInstance;
  expect(component.userName).toEqual('Pranam Bhat');
}));
```

{If you want to download sample code from **Git Repository**}

1) Download Angular sample code from **GitHub**

Git clone https://github.com/PranamBhat/Angular-UnitTesting.git

2) **cd Angular-UnitTesting**
3) **npm install**
4) **ng test**

Execute these commands:

1) **npm uninstall -g @angular/cli**

2) **npm cache clean**

3) **npm install -g @angular/cli@9.1.0**

4) **npm install --save-dev @angular/cli@latest**

5) **npm install -g @angular/cli@latest**

6) **ng update**

7) **ng update @angular/cli @angular/core**

8) **ng update @angular/cli @angular/core --allow-dirty --force**

9) **ng test**

ng test command built the app in watch mode and launches the **Karma**.

http://localhost:9876/?id=46094964

Test Angular Service

We will perform unit test for a service which manages the **HTTP** requests in **Angular** application. Create a service with below **Angular CLI** command:

ng generate service _services/post

This command will create two files:

1) src/app/_services/post.service.spec.ts
2) src/app/_services/post.service.ts

We will use here **JSON placeholder** dummy **API** of post listing. A method **postList()** is created to fetch the post listing from the third party server.

post.service.ts:

```typescript
import { Injectable } from '@angular/core';
import { Observable } from 'rxjs';
import { HttpClient } from '@angular/common/http';

export interface Post {
userId: number;
id: number;
title: string;
body: string;
}
@Injectable({
providedIn: 'root'
})
export class PostService {
API_URL: String = 'https://jsonplaceholder.typicode.com/posts';

constructor(
private http: HttpClient
) { }

postList(): Observable<Post[]> {
return this.http.get<Post[]>(`${this.API_URL}`)
}

}
```

Angular's new **HttpClient** has a testing module, **HttpClientTestingModule**, which makes it easy to unit test **HTTP** requests.

We are going to write a unit test for the post method crated using the **HTTP GET** request by taking the help of the **HttpTestingController** service.

post.service.spec.ts:

```typescript
import { TestBed, async, inject } from '@angular/core/testing';
import { HttpClientTestingModule, HttpTestingController } from '@angular/common/http/testing';
import { PostService } from './post.service';

describe('PostService', () => {
let service: PostService;
let httpMock: HttpTestingController;
```

```
beforeEach(() => {
TestBed.configureTestingModule({
imports: [
HttpClientTestingModule,
],
providers: [
PostService
]
});
service = TestBed.inject(PostService);
httpMock = TestBed.get(HttpTestingController);
});

it('should be created', () => {
expect(service).toBeTruthy();
});
it(`should fetch posts as an Observable`, async(inject([HttpTestingController, PostService],
(httpClient: HttpTestingController, postService: PostService) => {

const postItem = [
{
"userId": 1,
"id": 1,
"title": "sunt aut facere repellat provident occaecati excepturi optio reprehenderit",
"body": "quia et suscipit suscipit recusandae consequuntur expedita et cum reprehenderit
molestiae ut ut quas totam nostrum rerum est autem sunt rem eveniet architecto"
},
{
"userId": 1,
"id": 2,
"title": "qui est esse",
"body": "est rerum tempore vitae sequi sint nihil reprehenderit dolor beatae ea dolores
neque fugiat blanditiis voluptate porro vel nihil molestiae ut reiciendis qui aperiam non
debitis possimus qui neque nisi nulla"
},
{
"userId": 1,
"id": 3,
"title": "ea molestias quasi exercitationem repellat qui ipsa sit aut",
"body": "et iusto sed quo iure voluptatem occaecati omnis eligendi aut ad voluptatem
doloribus vel accusantium quis pariatur molestiae porro eius odio et labore et velit aut"
}
];

postService.postList()
.subscribe((posts: any) => {
expect(posts.length).toBe(3);
});
```

```
let req = httpMock.expectOne('https://jsonplaceholder.typicode.com/posts');
expect(req.request.method).toBe("GET");

req.flush(postItem);
httpMock.verify();

})));
});
```

Let's run **ng test** command to run the test cases created for **Angular** service.

Conclusion

In this article, we have learned about how to start writing test cases for **Angular Application**.

Let me know your thoughts over email **pranam707@gmail.com**. I would love to hear them and if you like this article, share it with your friends.