ANGULAR – UNIT TESTING

Jasmine & Karma

INTRODUCTION

Jasmine
(JavaScript Testing Framework)

Karma:
(Test runner)

JASMINE

- Open source testing framework for JavaScript
- Behavior-driven testing framework
- Easy to read, easy to understand
- Supports asynchronous testing

KARMA

- Open source test runner for JavaScript created by Angular Team
- Well integrated and easy to execute test using CLI
- Can be configured with various testing frameworks like Jasmine, Mocha, Qunit etc
- Useful for executing tetst for browser and other devices.

ANGULAR TESTING CONFIGURATION

Jasmin and karma

GETTING STARTED WITH TEST…

Test String using Matchers

- toBe
- toEqual
- toContain
- toMatch

Test Array using Matchers

- toEqual
- toContain

Excluding (x) test cases from execution

SETUP AND TEAR-DOWN

Common Methods

- beforeEach
- beforeAll
- afterEach
- afterAll

ARRANGE, ACT & ASSERT (AAA)

Arrange – means arrange everything to setup unit test cases

Act – means execute necessary functionality/methods that needs to be tested

Assert – verifies the functionality that unit test giving the result as per expectation

TESTBED & COMPONENT FIXTURE

- One of the main utilities for unit testing components, directive, services in Angular & it is Angular specific
- The TestBed creates a dynamically-constructured Angular test module that emulates an Angular @NgModule
- It also provides methods for creating components and services for unit testing cases

JASMINE : SPYON

- Jasmine spies help us to mock the execution of method / function
- It's easy way to check a method was called or not, without leaving Subject Under Test (SUT)
- We can chain the spyOn method to get dummy return value using .and.returnValue()
- spyOn can call the original function using .and.callThrough()

WORKING WITH CHANGE DETECTION

Fixture.detectChanges()

DEBUG ELEMENT & DOM EVENTS

- debugElement is the interface created to work safely across different platforms
- Create debugElement tree that wraps the native elements for platforms, instead of HTML element tree.
- It is associated with the root element of the component.
- Comes with methods and properties which are useful for testing.
- nativeElement property returns us platform specific object.

ASYNCHRONOUS TESTING

done()

async…whenStable()

fakeAsync…tick()

JASMINE DONE()

Jasmine has built-in way to handle async code, passed to beforeEach, afterEach and it() statements and is called done() callback.

done() callback are responsible for chaining promises, handling errors and calling done() at the appropriate moments.

The test specs are completed after the invocation of done() callbac and this is its primary feature.

ASYNC…WHEN.STABLE()

Wraps a test function in a async test zone.

whenStable() helps us to test promises by allowing us to wait until all the promises get completed.

fixture.whenStable() return a promise that resolves when the JavaScript engine task queue becomes empty.

FAKEASYNC…TICK()

fakeAsync() executes the test code in special fakeAsync Test Zone.

Tick utility blocks the execution of all the microtasks and simulate passage of time until all the async tasks are executed.

fakeAsync and tick – executes the async code in sync way

HTTP CLIENT TESTING MODULE

HttpClientTestingModule is used for testing HTTP methods using HttpClient

It also inject HttpTestingController that allow us to mock and flush the requests

HTTP TESTING CONTROLLER

It's a controller injected into test, that allow for mocking and flush the requests

Methods:	
	expectOne() – expects single request made matching the given URL and returns mock data
	expectNone() – expects no requests has been made for the given URL
	match() – without any expectation, match the request given as parameter
	verify() – verifies that no unmatched requests are outstanding

TEST REQUEST

A mock request that was received and ready to be answered

It has following methods -

- flushed() – resolves request by returning the body
- error – resolves the request by returning errorEvent.
- event() – returns an arbitrary HttpEvent

It has following properties -

- cancelled – weather the request is cancelled or not
- request – type of HttpRequest Object

CODE COVERAGE

- Code coverage is a way of determining the part of code which is left out of testing or not tested properly
- Karma uses Istanbul reporter tool to create code coverage report
- With Karma, we can produce the code coverage report using CLI Command –
  - `ng test --code-coverage`
- One the CLI command is executed, it creates index.html in /coverage folder

REFERENCES

https://angular.io/guide/testing

https://www.youtube.com/

https://www.testim.io/blog/angular-testing-tutorial/

https://blog.logrocket.com/angular-unit-testing-tutorial-examples/

https://medium.com/swlh/angular-unit-testing-jasmine-karma-step-by-step-e3376d110ab4