Unit Testing in Javascript

Outline

- Introduction to Unit Testing
- Jasmine / Karma
- Code Examples
- Best Practices

About Me

Hi All, Myself Mohit Khandelwal. I have 1 year of experience working as Software developer at Infosys, Pune.

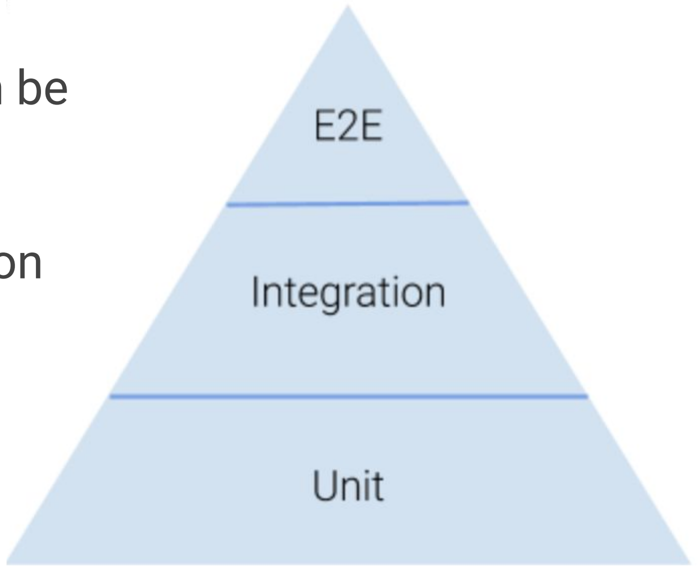
linkedin.com/in/mohitkh7

github.com/mohitkh7

What is Unit Test ?

Test the smallest individual part of your code that can be logically isolated in system.

Written by developers to ensure a section of application works as intended.



Advantages

- Lightweight and Fast
- Catches bugs in early stage of development.
- Helps to refactor code with confidence.
- Provides a sort of living documentation for piece of code.

Drawbacks

- Unit test will not catch every error of program.
- It takes time and effort. (But it's worth it.)

Tools for unit testing

- **Testing Framework -** help you arrange your tests in a readable and scalable way (Mocha, Jasmine, Jest, Cucumber, TestCafe, Cypress)
- **Assertion Library -** to check if a test returns what you expect (Chai, Jasmine, Jest, Unexpected, TestCafe, Cypress)
- **Test Runner** - to run all are test automated way and generate summarised result. (Karma, Jasmine, Jest, TestCafe, Cypress, webdriverio)
- **Coverage Report** - to generate reports on what all part of code is covered by tests. (Istanbul, Jest, Blanket)

Jasmine

Jasmine is a behavior-driven development framework for testing JavaScript code. It provides test structure, assertion functions, mocks/spies and everything you need to test your code.

Why prefer Jasmine ?

- It has a clean and obvious syntax.
- It has a huge community support.
- It is a recommend testing framework for Angular Applications.

Test Structure

In Jasmine test are structured as suites and specs -

- ***Suite*** - It is a group of related test cases (specs). This begins with a call to the Jasmine global function `describe`
- ***Spec*** - It represents a test case inside a suite. Usually spec contains one or more expectations that test the state of the code. Specs are defined by calling the global Jasmine function `it`

Jasmine Matcher

Matchers are used to make assertions by comparing the actual and expected outputs of any jasmine test. Jasmine not keyword can be used with every matcher's criteria for inverting the result

- toBe()
- toEqual()
- toBeTruthy()
- toBeFalsy()
- toContain()
- toBeDefined()
- toBeNull()
- toBeLessThan()
- toBeGreaterThan()
- toThrow()

Karma

Karma is a test runner tool which automates whole process from running test cases to display results in browser. It spawns a web server that executes source code against test code for each of the browsers connected.

Key Features

- It can watch source file for changes, and automatically triggers test run.
- It can execute test on multiple browser and device simultaneously.
- Integration with CI tools such as Jenkins, Travis etc.

Ignored / Focused Test

- Adding a character `x` in front of suite/spec will make it excluded so that all suite/spec will run except that one.

  Ex - `xdescribe`, `xit`

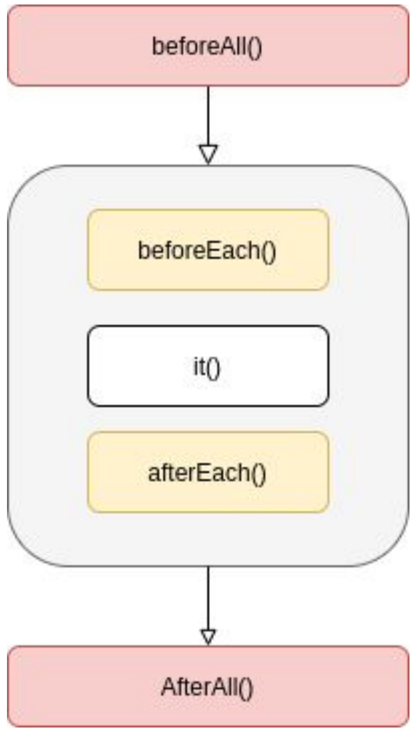
- Adding a character `f` in front of suite/spec will make it focused so that they are the only suite/spec that run.

  Ex - `fdescribe`, `fit`

Setup / Teardown

- beforeAll() - runs once before all specs in suite.
- beforeEach() - runs before every specs.
- afterEach() - runs after every specs.
- afterAll() - runs once after all spec in suite are executed.

This methods are used for common initialization, setup and finalization code so that we don’t have to repeat in every spec.



Best Practices

- Write unit test along with code.
- Have a clear description of your test case.
- Each spec should test only specific single unit.
- Avoid making too many assertion in a test spec.
- Mock / Spy all the external dependencies.

Source Code - https://github.com/mohitkh7/presentation-unit-testing-js

For any query/comment/feedback you can mail me at *mohitkh7@gmail.com*

Thank You !