**Testing NodeJS app using Mocha and Chai**



* [Mocha](https://mochajs.org/) is a feature-rich JavaScript test framework running on Node.js and in the browser.
* [Chai](http://chaijs.com/) is an assertion library for [node](http://nodejs.org/).

# Mocha

* Mocha is a JavaScript Test Framework.
* Runs on Node.js and Browser
* Installation: (Run the below commands in terminal or cmd)

npm install --global mocha

Note: To run Mocha, we need Node.js v4 or newer.

—- global helps to install the Mocha on computer at global level which helps to run mocha test through command line.

—- save-dev helps to add the mocha as dependency in package.json file for that particular project.

## Mocha Basic Spec

var assert = require('assert');

describe('Basic Mocha String Test', function () {  
 it('should return number of charachters in a string', function () {  
 assert.equal("Hello".length, 4);  
 });

it('should return first charachter of the string', function () {  
 assert.equal("Hello".charAt(0), 'H');  
 });  
});

In the above test snippet,

* ***assert*** helps to determine the status of the test, it determines failure of the test.
* ***describe***is a function which holds the collection of tests. It takes two parameters, first one is the meaningful name to functionality under test and second one is the function which contains one or multiple tests. We can have nested *describe* as well.
* ***it*** is a function again which is actually a test itself and takes two parameters, first parameter is name to the test and second parameter is function which holds the body of the test.

Chai

* Chai is BDD/TDD assertion library.
* Can be paired with any javascript testing framework.
* Assertion with Chai provides natural language assertions, expressive and readable style.
* Installation: (Run the below commands in terminal or cmd)

npm install --save-dev chai

## Assertion interfaces and styles

* There are two popular way of assertion in Chai, **expect** and **should**
* The **expect** interface provides function for assertion.
* The **should** interface extends each object with a should property for assertion.
* **should** property gets added to the Object.Prototype, so that all object can access it through prototype chain.

Below is the usage of expect and should instead of Mocha assert

var assert = require('assert');  
var expect = require('chai').expect;  
var should = require('chai').should();it('should return true if valid user id', function(){  
 var isValid = loginController.isValidUserId('abc123')  
 //assert.equal(isValid, true);  
 **expect(isValid).to.be.true;**  
});it('should return false if invalid user id', function(){  
 var isValid = loginController.isValidUserId('abc1234')  
 //assert.equal(isValid, false);  
 **isValid.should.equal(false);**  
});