CitiusTech

Behaviour-Driven Testing with MOCHA & CHAI – Session 1

November 2015

Agenda

- **What is Mocha?**
- Installing Mocha
- Using Mocha from Command Line
- Mocha Test Case & Test Suite
- Using Mocha Inside a Browser
- Assertion Libraries

What is Mocha?

- A JavaScript testing framework
- Open Source
  - Supports both **TDD & BDD** style testing
  - Supports both **client & server** side testing
  - Can be run on both, **browser & command-line**
    - The command-line would require **Node JS** to be installed
  - Supports any assertion library
  - Supports **asynchronous testing**
- Primarily developed for **Node JS**, but has gained wide-spread adoption for browser testing

Agenda

- What is Mocha?
- **Installing Mocha**
- Using Mocha from Command Line
- Mocha Test Case & Test Suite
- Using Mocha Inside a Browser
- Assertion Libraries

Installing Mocha

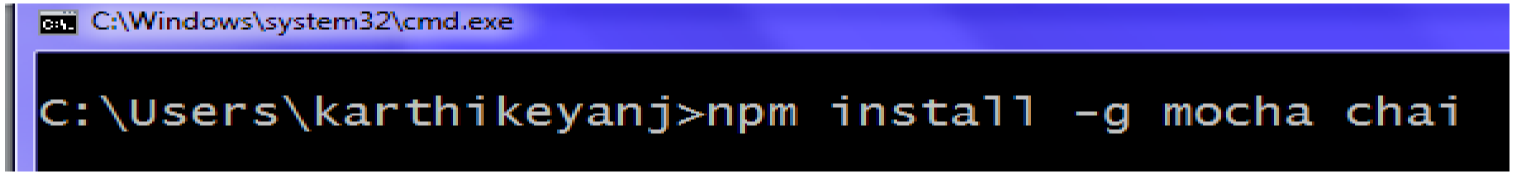
- Mocha can be installed in two ways:
  - Get the source code from: https://github.com/visionmedia/mocha
    - This is useful when you want to use Mocha inside a browser
  - Install it using **npm (Node Package Manager)** from the command-line
    - **npm install –g mocha**
  - **npm** is an online repository for the publishing of open-source Node.js projects. It is also a command-line utility for interacting with a repository that aids in package installation, version management, & dependency management
  - Any **Node JS** library can be easily installed from npm using a single command
- In case you want to use both, i.e., command-line & browser, then:
  - Install mocha using **npm** as shown above
  - Go to the installation directory & copy the relevant files into your web application

Agenda

- What is Mocha?
- Installing Mocha
- **Using Mocha from Command Line**
- Mocha Test Case & Test Suite
- Using Mocha Inside a Browser
- Assertion Libraries

Using Mocha from Command-Line

- Install **Node JS**
- Perform a **global installation** of **Mocha & Chai** using **npm** from the command-line as shown below:



C:\Windows\system32\cmd.exe

```
C:\Users\karthikeyanj>npm install -g mocha chai
```

- This installation creates a folder named **node_modules** in the currently logged in user's directory at the following location:
 C:\Users\<<currently logged in user>>\AppData\Roaming\npm

Agenda

- What is Mocha?
- Installing Mocha
- Using Mocha from Command Line
- **Mocha Test Case & Test Suite**
- Using Mocha Inside a Browser
- Assertion Libraries

Mocha Test Case & Test Suite

- Test case
  - A test case is a scenario, which has a set of test data, preconditions, expected results and post conditions, developed for a particular test scenario in order to verify compliance against a specific requirement
- Test suite
  - A container that has a set of tests which helps testers in executing and reporting the test execution status
- The **describe()** function in Mocha is used for creating a test suite

```
describe(“test suite name”,function())
```

- The **it()** function in Mocha is used for creating a test case

```
it(“test case description” , function())
```

Create Tests from Command-Line

- Create a file named **mytests.js** inside the **npm** folder
- Write the following code inside **mytests.js**:

```
var chai = require('chai');             //load the chai module from Node
var expect = chai.expect;               //specify that we are using ‘expect’ syntax

//create a test suite
describe("A Sample Test Suite", function ()
{
   it("This is test1", function ()
   {
      expect(true).to.equal(true);   //this is an ASSERTION
   });

   it("This is test2", function ()
   {
      expect('hello').to.equal('hello');              //this is an ASSERTION
   });
});
```

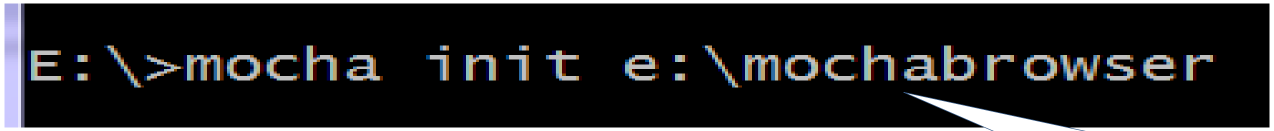
CitiusTech

Agenda

- What is Mocha?
- Installing Mocha
- Using Mocha from Command Line
- Mocha Test Case & Test Suite
- **Using Mocha Inside a Browser**
- Assertion Libraries
- Chai JS

Using Mocha Inside a Browser (1/4)

- Run the following command from the command prompt so that Mocha creates a boilerplate HTML file with the necessary set up to run Mocha from a browser



```
E:\>mocha init e:\mochabrowser
```

This path can be any path on your system. In this example, the files will be created inside a pre-created folder named **mochabrowser**

- Copy **chai.js** from the **chai** folder under **npm\node_modules** folder in the Node installation path, into **mochabrowser** folder
  - This is required since we will be using assertions from the **Chai** library
- Open **index.html** file from the **mochabrowser** folder for editing

Using Mocha Inside a Browser (2/4)

- Modify code inside the HTML file to add reference to chai.js as shown in red

```
<html>
  <head>
    <title>Mocha</title>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="mocha.css" />
    <script src="chai.js"></script>
  </head>
  <body>
    <div id="mocha"></div>
    <script src="mocha.js"></script>
    <script>mocha.setup('bdd');</script>
    <script src="tests.js"></script>
    <script>
      mocha.run();
    </script>
  </body>
</html>
```

1. There must be a **container** element with the **id** as **mocha.** This is used by mocha to display the test results on the page.
2. Reference the **mocha library**
3. This tells mocha to use the **BDD** approach for writing unit tests. You can use the **TDD** approach as well
4. Reference your test files here
5. Initiate the **Mocha Test Runner**

CitiusTech

Using Mocha Inside a Browser (3/4)

Add Tests

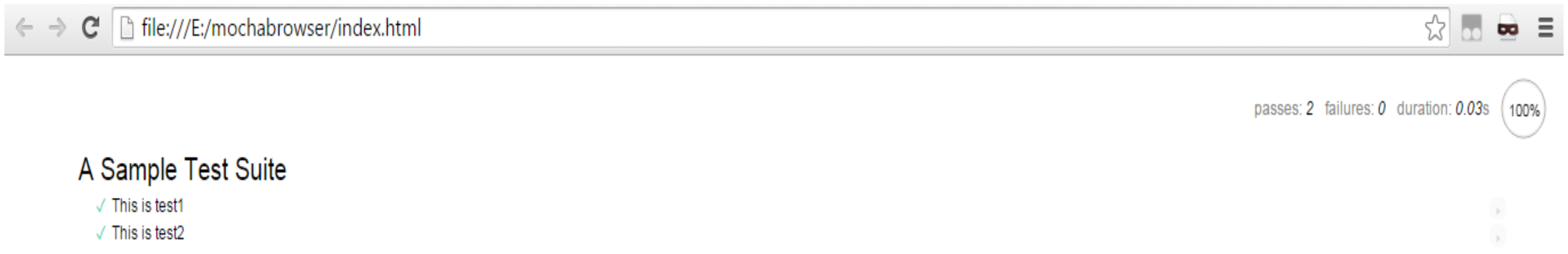
- Open **tests.js** file for editing
- Add the following tests:

```
var expect = chai.expect;

//create a test suite
describe("A Sample Test Suite", function ()
{
    it("This is test1", function ()
    {
        expect(true).to.equal(true);
    });

    it("This is test2", function ()
    {
        expect('hello').to.equal('hello');
    });
});
```

In this case, there is no need to explicitly load the **chai** module since a reference to **chai.js** has been added in the HTML file

Using Mocha Inside a Browser (4/4)

- Launch the **Index.html** page inside a browser
  - Mocha will run the tests inside **tests.js** file & display the output as shown below:



```
file:///E:/mochabrowser/index.html

passes: 2  failures: 0  duration: 0.03s  100%

A Sample Test Suite
  ✓ This is test1
  ✓ This is test2
```

Agenda

- What is Mocha?
- Installing Mocha
- Using Mocha from Command Line
- Mocha Test Case & Test Suite
- Using Mocha Inside a Browser
- **Assertion Libraries**

Assertion Libraries

- Are JavaScript libraries that allow you to choose your own assertion syntax
- They are add-on libraries that you can use with any unit-testing Framework
- Mocha does not come with its own assertion library, so you are forced to choose one
- Other popular assertion libraries include:
  - YUI Port
  - expect.js
  - should.js
  - jshould.js
  - assert.js
- **chai.js** is the most common & versatile assertion library that can be used with Mocha

THANK YOU