

## Lesson 11 Demo 09

### Implementing Reporters in Mocha

**Objective:** To test Node.js with the Mocha module and generate different output using the reporter concept for improved test result presentation and analysis

**Tools Required:** Visual Studio

**Prerequisites:** Knowledge of JavaScript, and Node.js

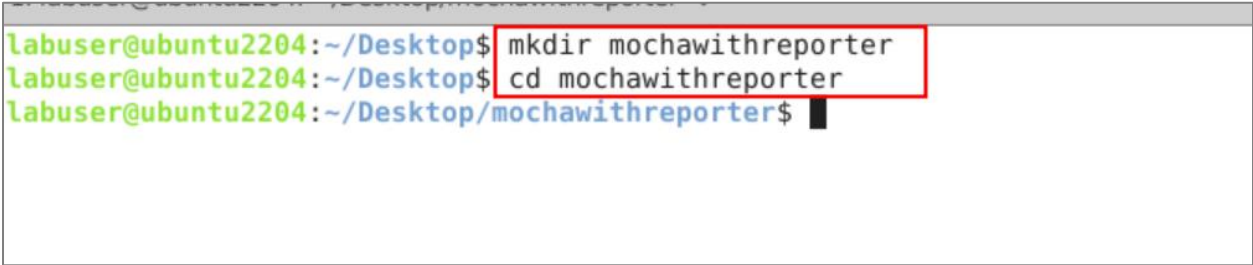
Steps to be followed:

1. Create the project structure, package.json file and install the mocha module
2. Create JavaScript and testing the files

#### Step 1: Create the project structure, package.json file and install mocha module

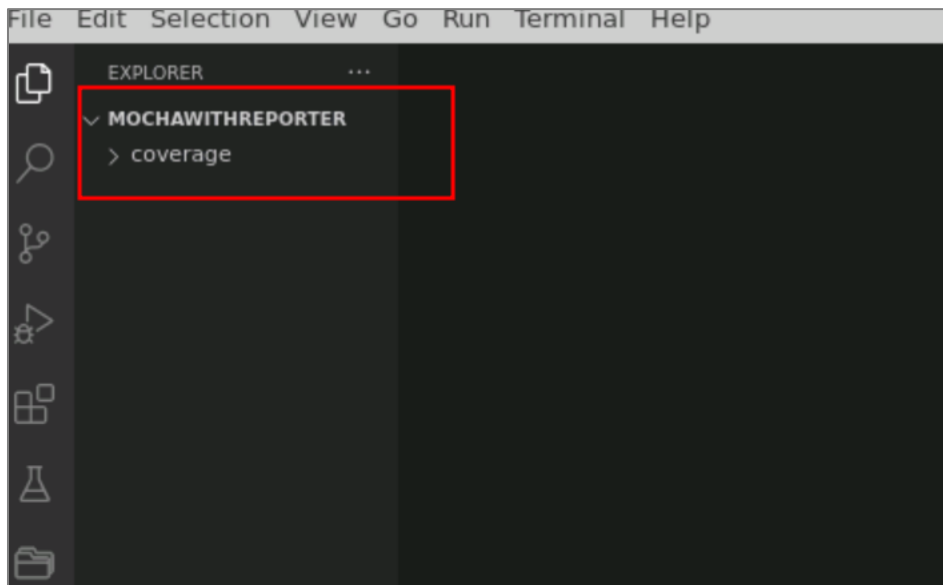
- 1.1 Run the following code to create a directory to hold the application and make it the working directory:

```
mkdir mochawithreporter  
cd mochawithreporter
```



```
labuser@ubuntu2204:~/Desktop$ mkdir mochawithreporter  
labuser@ubuntu2204:~/Desktop$ cd mochawithreporter  
labuser@ubuntu2204:~/Desktop/mochawithreporter$
```

## 1.2 Open this folder in the VSCode IDE



## 1.3 Open the terminal and run the following command to create a **package.json** file for the application

**npm init -y**

```
labuser@ubuntu2204:~/Desktop$ cd mochawithreporter
labuser@ubuntu2204:~/Desktop/mochawithreporter$ npm init -y
Wrote to /home/labuser/Desktop/mochawithreporter/package.json:

{
  "name": "mochawithreporter",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC"
}

labuser@ubuntu2204:~/Desktop/mochawithreporter$
```

It creates a package.json file with default option

Now, you can view the package.json file

```

() package.json X
() package.json > ...
1  {
2    "name": "nodejstestingmocha",
3    "version": "1.0.0",
4    "description": "",
5    "main": "index.js",
6    > Debug
7    "scripts": {
8      "test": "echo \"Error: no test specified\" && exit 1"
9    },
10   "author": "",
11   "license": "ISC"
12 }
  
```

1.4 Install Chai by using the **npm install mocha chai -D** command

```

labuser@ubuntu2204:~/Desktop/mochawithreporter$ npm install mocha chai -D
added 8 packages, and audited 86 packages in 1s

20 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
labuser@ubuntu2204:~/Desktop/mochawithreporter$
  
```

## Step 2: Create JavaScript and test the files

- 2.1 Create a src folder which contains all the source code; Inside the src folder, create an app.js file.
- 2.2 Create a test folder which contains all the source code; Inside the test folder, create an appTest.js file.
- 2.3 Add the code in the app.js file as shown in the screenshot below:

```

package.json  JS app.js  X  JS appTest.js
src > JS app.js > [0] <unknown>
1  function add(a,b){
2      return a+b;
3  }
4  function sub(a,b){
5      return a-b;
6  }
7  function mul(a,b){
8      return a*b;
9  }
10 function div(a,b){
11     return a/b;
12 }
13
14 module.exports={add,sub,mul,div}

```

2.4 Add the code in the appTest.js file as shown in the screenshot below:

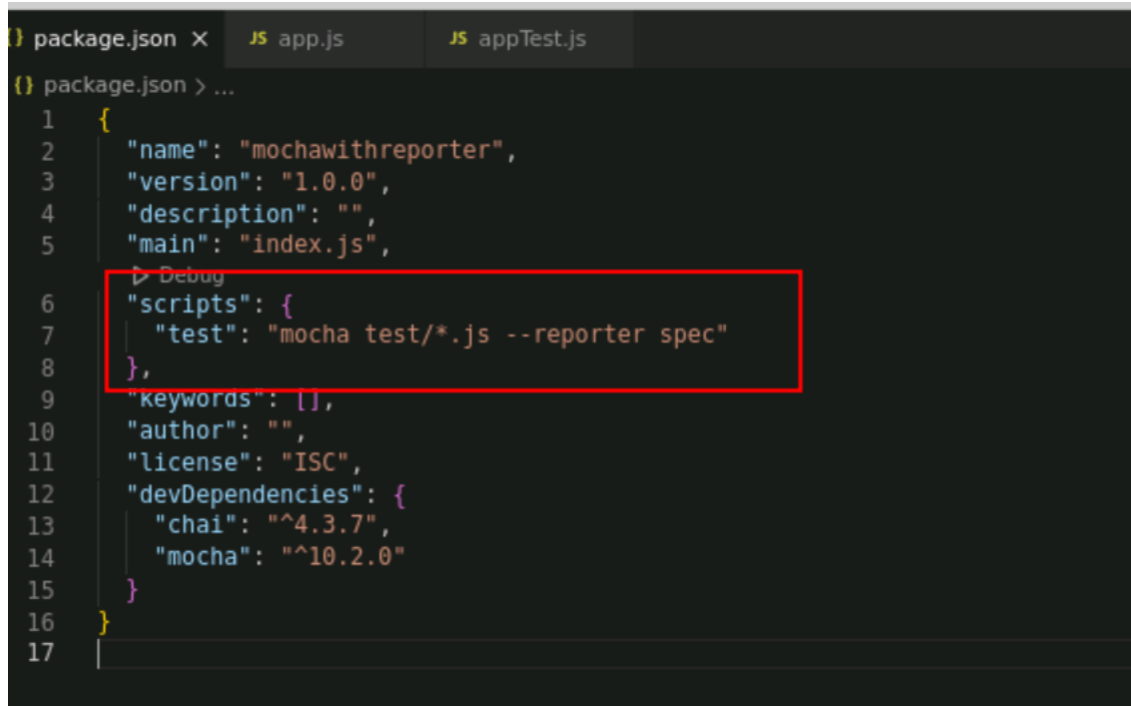
```

package.json  JS app.js  JS appTest.js X
test > JS appTest.js > describe("Operation ") callback > it("div testing ") callback
1  const assert = require("chai").assert;
2  const appRef = require("../src/app");
3  describe("Operation ", () => {
4      it("add testing ", () => {
5          let actualResult = appRef.add(10,20);
6          let expectedRsult = 30;
7          assert.equal(actualResult,expectedRsult);
8      })
9      it("sub testing ", () => {
10         let actualResult = appRef.sub(100,50);
11         let expectedRsult = 50;
12         assert.equal(actualResult,expectedRsult);
13     })
14     it("mul testing ", () => {
15         let actualResult = appRef.mul(5,4);
16         let expectedRsult = 20;
17         assert.equal(actualResult,expectedRsult);
18     })
19     it("div testing ", () => {
20         let actualResult = appRef.div(100,20);
21         let expectedRsult = 5;
22         assert.equal(actualResult,expectedRsult);
23     })
24 })

```

## 2.5 In the package.json file write the testing file details

Make reporter option as spec

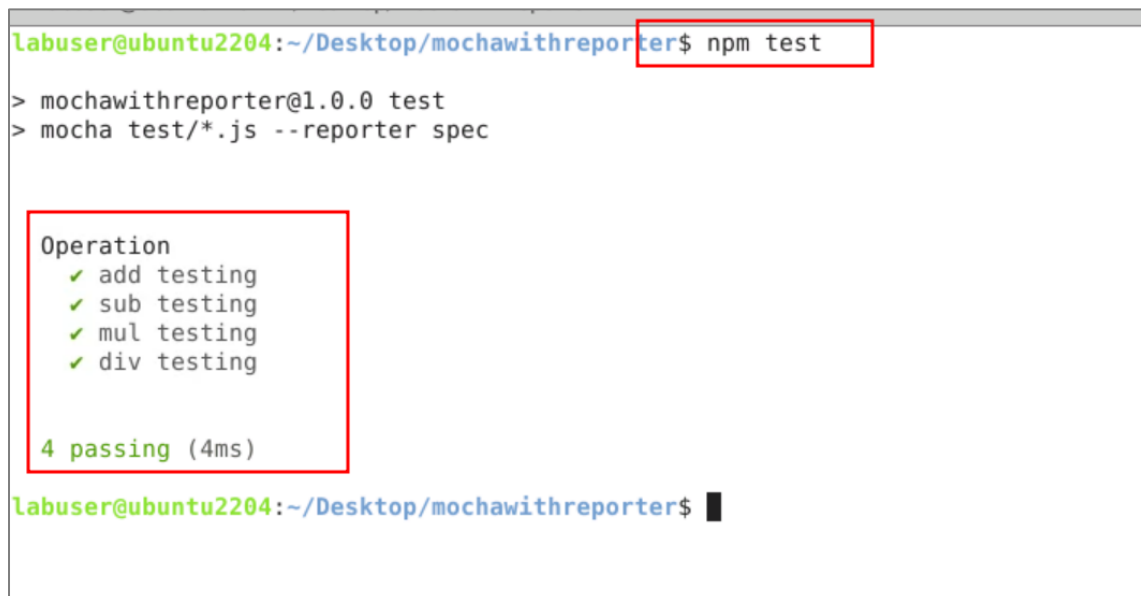


```

1  {
2    "name": "mochawithreporter",
3    "version": "1.0.0",
4    "description": "",
5    "main": "index.js",
6    "scripts": {
7      "test": "mocha test/*.js --reporter spec"
8    },
9    "keywords": [],
10   "author": "",
11   "license": "ISC",
12   "devDependencies": {
13     "chai": "^4.3.7",
14     "mocha": "^10.2.0"
15   }
16 }
17

```

## 2.6 Run the application using command as **npm test**



```

labuser@ubuntu2204:~/Desktop/mochawithreporter$ npm test

> mochawithreporter@1.0.0 test
> mocha test/*.js --reporter spec

Operation
  ✓ add testing
  ✓ sub testing
  ✓ mul testing
  ✓ div testing

4 passing (4ms)

labuser@ubuntu2204:~/Desktop/mochawithreporter$

```

```
labuser@ubuntu2204: ~/Desktop/mochawithreporter$ npm test
```

```
> mochawithreporter@1.0.0 test
> mocha test/*.js --reporter nyan
```

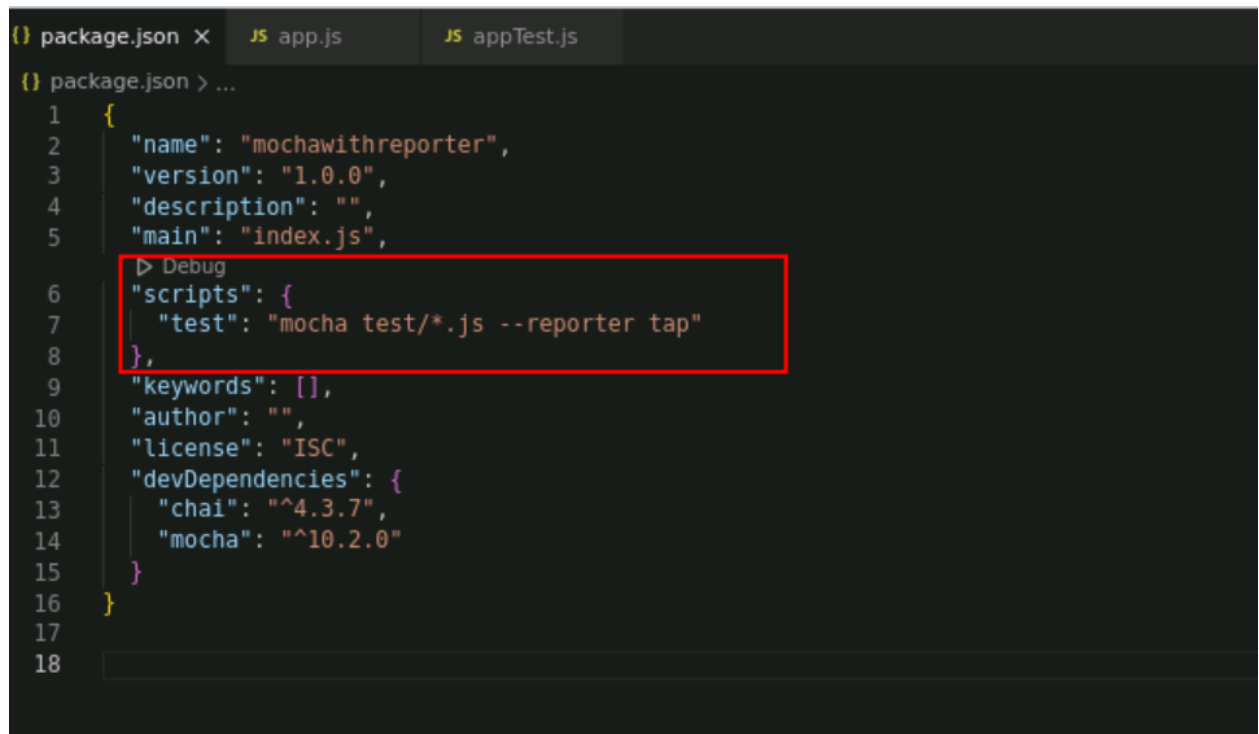
```

  4
  0
  0
    - - - - -
    |         \   /
    |        ^ _ ^
    |       ___( ^ .^ )___
    |      " "     " "
    - - - - -

  4 passing (6ms)
```

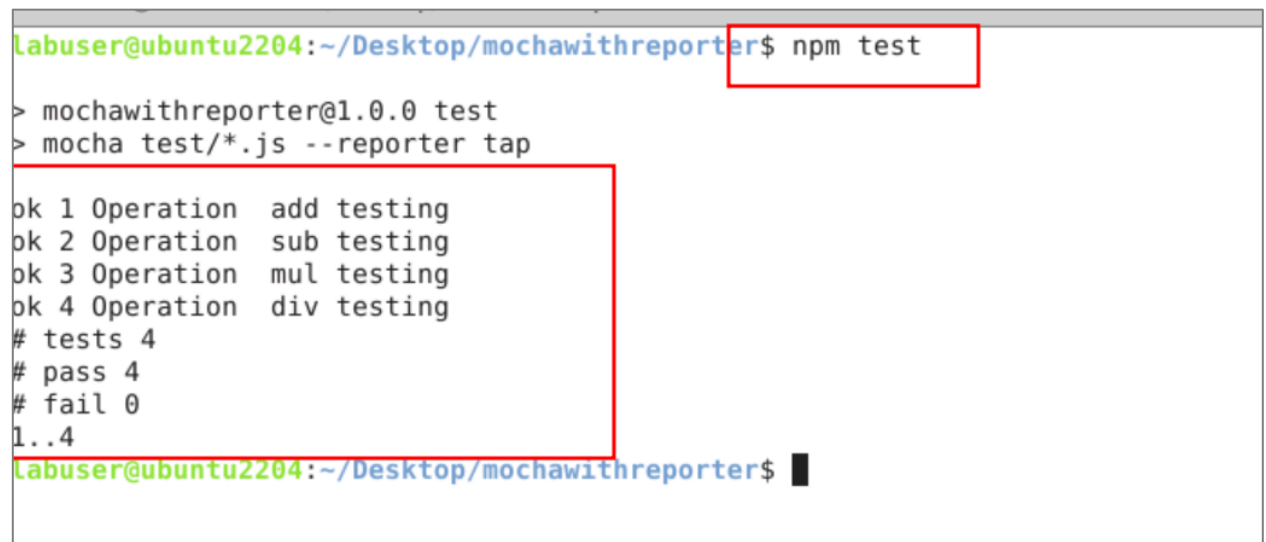
```
labuser@ubuntu2204:~/Desktop/mochawithreporter$
```

Make reporter option as tap



```
{ package.json x JS app.js JS appTest.js
{} package.json > ...
1 {
2   "name": "mochawithreporter",
3   "version": "1.0.0",
4   "description": "",
5   "main": "index.js",
6   "scripts": {
7     "test": "mocha test/*.js --reporter tap"
8 },
9   "keywords": [],
10  "author": "",
11  "license": "ISC",
12  "devDependencies": {
13    "chai": "^4.3.7",
14    "mocha": "^10.2.0"
15  }
16 }
17
18
```

Now run the test



```
Labuser@ubuntu2204:~/Desktop/mochawithreporter$ npm test

> mochawithreporter@1.0.0 test
> mocha test/*.js --reporter tap

ok 1 Operation    add testing
ok 2 Operation    sub testing
ok 3 Operation    mul testing
ok 4 Operation    div testing
# tests 4
# pass 4
# fail 0
1..4
Labuser@ubuntu2204:~/Desktop/mochawithreporter$
```

By following these steps, you have successfully tested Node.js with the Mocha module, utilizing various reporter concepts to generate diverse outputs for enhanced test result presentation and analysis.