Angular 2 Component Testing

In this exercise, we will implement the component testing in Angular 2 application.

Step 1: Modify the package.json as shown in the following code (Highlighted)

```
{
  "name": "component-test-app",
  "version": "0.0.0",
  "license": "MIT",
  "angular-cli": {},
  "scripts": {
    "start": "concurrently \"tsc -w\" \"node server.js\"",
    "tsc": "tsc",
    "tsc:w": "tsc -w",
    "lite": "lite-server"
  },
  "private": true,
  "dependencies": {
    "@angular/common": "2.0.0",
    "@angular/compiler": "2.0.0",
    "@angular/core": "2.0.0",
    "@angular/forms": "2.0.0",
    "@angular/http": "2.0.0",
    "@angular/platform-browser": "2.0.0",
    "@angular/platform-browser-dynamic": "2.0.0",
    "@angular/router": "3.0.0",
    "core-js": "^2.4.1",
    "rxjs": "5.0.0-beta.12",
    "ts-helpers": "^1.1.1",
    "es6-shim": "^0.35.0",
    "zone.js": "^0.6.23",
    "koa": "^1.2.0",
    "koa-static": "^2.0.0",
    "livereload": "^0.4.1",
    "systemjs": "0.19.27",
    "reflect-metadata": "^0.1.3",
    "bootstrap": "*"
  },
  "devDependencies": {
    "@types/jasmine": "^2.2.30",
    "angular-cli": "1.0.0-beta.16",
    "@types/es6-shim": "^0.31.32",
    "codelyzer": "~0.0.26",
    "jasmine-core": "2.4.1",
    "jasmine-spec-reporter": "2.5.0",
    "karma": "1.2.0",
    "karma-chrome-launcher": "^2.0.0",
    "karma-cli": "^1.0.1",
```

```
    "karma-jasmine": "^1.0.2",
    "karma-remap-istanbul": "^0.2.1",
    "protractor": "4.0.9",
    "ts-node": "1.2.1",
    "tslint": "3.13.0",
    "typescript": "2.0.2"
  }
}
```

We will be using Jasmine for browser testing

Step 2: Modify the systemjs.config.js for the testing library mapping as shown in the following code (Highlighted)

```
var map = {
        "rxjs": "node_modules/rxjs",
        "@angular/common": "node_modules/@angular/common",
        "@angular/forms": "node_modules/@angular/forms",
        "@angular/compiler": "node_modules/@angular/compiler",
        "@angular/compiler/testing": "node_modules/@angular/compiler/bundles",
        "@angular/core": "node_modules/@angular/core",
        "@angular/core/testing":"node_modules/@angular/core/bundles",
        "@angular/platform-browser":"node_modules/@angular/platform-browser",
        "@angular/platform-browser/testing":"node_modules/@angular/platform-
browser/bundles",
        "@angular/platform-browser-dynamic":"node_modules/@angular/platform-browser-
dynamic",
        "@angular/platform-browser-dynamic/testing":"node_modules/@angular/platform-browser-
dynamic/bundles"
};
var packages = {
        "rxjs": { "defaultExtension": "js" },
        "@angular/common": { "main": "bundles/common.umd.js", "defaultExtension": "js" },
        "@angular/forms": { "main": "bundles/forms.umd.js", "defaultExtension": "js" },
        "@angular/compiler": { "main": "bundles/compiler.umd.js", "defaultExtension": "js" },
        "@angular/compiler/testing": { "main": "compiler-testing.umd.js", "defaultExtension": "js" },
        "@angular/core": { "main": "bundles/core.umd.js", "defaultExtension": "js" },
        '@angular/core/testing':{"main":"core-testing.umd.js", "defaultExtension": "js"},
        "@angular/platform-browser":{"main":"bundles/platform-
browser.umd.js","defaultExtension": "js"},
        '@angular/platform-browser/testing':{"main":"platform-browser-
testing.umd.js","defaultExtension":"js"},
        "@angular/platform-browser-dynamic":{"main":"bundles/platform-browser-
dynamic.umd.js","defaultExtension":"js"},
        "@angular/platform-browser-dynamic/testing":{"main":"platform-browser-dynamic-
testing.umd.js","defaultExtension":"js"},
        "app": {
                format: 'register',
```

```
                    defaultExtension: 'js'
            }
};

var config = {
            map: map,
            packages: packages
};

System.config(config);
```

Step 3: Add a simple component in the project of name TestComponent with the following code

```
import { Component, OnInit } from '@angular/core';

@Component({
   template: '<h1>{{testData}}</h1>'
})
export class TestComponent implements OnInit {
    public a1:number;
    public b1:number;
    public c1:number;
   constructor() {
      this.a1 = 10;
      this.b1 = 10;
    }

   ngOnInit() {

     this.c1 = (this.a1 * this.a1) + 2 *this.a1 * this.b1+ (this.b1 * this.b1);
   }
   addNumbers(x:number,y:number):number{
      return x+y;
   }
}
```

Step 4: Add a new test specification file in the project of name app.testcomponent.spec.ts with the following code

```
import { inject,TestBed,ComponentFixture,async } from '@angular/core/testing';
import { BrowserDynamicTestingModule, platformBrowserDynamicTesting } from
'@angular/platform-browser-dynamic/testing';
import { By } from '@angular/platform-browser';
import { FormsModule } from '@angular/forms';
import { TestComponent } from './app.component';
TestBed.initTestEnvironment(
   BrowserDynamicTestingModule, platformBrowserDynamicTesting());
```

```
describe(TestComponent, () => {
  let employee;
  let app;
  let fixture: ComponentFixture< TestComponent >;

beforeEach(() => {
     TestBed.configureTestingModule({
     declarations: [TestComponent],
   });
   fixture = TestBed.createComponent(TestComponent);
   app    = fixture.componentInstance;
   fixture.detectChanges();
  });
it('ngOnInitTest', () => {
   app.c1 = 0;
   let res = 400;
   app.ngOnInit();
   expect(app.c1).toEqual(res);
  });
it('addNumberTest', () => {
   let x = 10;
   let y = 20;
   let res = 30;
   let actRes = app.add(x,y);
   expect(actRes).toEqual(res);
  });
});
```

Step 5: Add the following code in boot.ts

```
import { platformBrowserDynamic } from '@angular/platform-browser-dynamic';
import { NgModule } from '@angular/core';
import { FormsModule } from '@angular/forms';
import { BrowserModule } from '@angular/platform-browser';
import{TestComponent} from './app.component';
 @NgModule({
   imports: [BrowserModule,FormsModule],
  declarations: [TestComponent],
  bootstrap: [TestComponent]
})
export class AppModule { }

platformBrowserDynamic().bootstrapModule(AppModule);
```

Step 6: Add a TestComponent.html file with the following markup in it

```
<!DOCTYPE html>
<html>
<head>
  <meta http-equiv="content-type" content="text/html;charset=utf-8">
  <title>Ng App Unit Tests</title>
  <link rel="stylesheet" href="./node_modules/jasmine-core/lib/jasmine-core/jasmine.css">

  <script src="./node_modules/jasmine-core/lib/jasmine-core/jasmine.js"></script>
  <script src="./node_modules/jasmine-core/lib/jasmine-core/jasmine-html.js"></script>
  <script src="./node_modules/jasmine-core/lib/jasmine-core/boot.js"></script>
  <script src="node_modules/es6-shim/es6-shim.min.js"></script>
  <script src="node_modules/reflect-metadata/Reflect.js"></script>
  <script src="node_modules/zone.js/dist/zone.js"></script>
  <script src="node_modules/zone.js/dist/long-stack-trace-zone.js"></script>
  <script src="node_modules/zone.js/dist/async-test.js"></script>
  <script src="node_modules/zone.js/dist/fake-async-test.js"></script>
  <script src="node_modules/zone.js/dist/sync-test.js"></script>
  <script src="node_modules/zone.js/dist/proxy.js"></script>
  <script src="node_modules/zone.js/dist/jasmine-patch.js"></script>
  <script src="node_modules/systemjs/dist/system.src.js"></script>

</head>
<body>
  <script src="systemjs.config.js"></script>
  <script>
    Promise.all([
  System.import('@angular/core/testing'),
  System.import('./app/app.component.spec')
])
      .then(window.onload)
       .catch(console.error.bind(console));
  </script>
</body>
</html>
```

Run the server using the Command prompt (npm command) and browse to the TestComponent.html page the following result will be displayed