# ASP.NET MVC 5 – Using Angular 4 with TypeScript in Visual Studio

BY MITHUN PATTANKAR · PUBLISHED JULY 28, 2016 · UPDATED AUGUST 23, 2017

ASP.NET MVC 5 is widely used web development framework, it's stable, matured and most importantly its is used in production on large scale. Many folks had requested me to write how to wire up Angular 2 in MVC 5.

- **Update 23/8/17 – Github repo updated with Angular 4.3.5, TypeScript 2.4**
- Update 26/5/17 – Github repo updated with Angular 4.1, TypeScript 2.3
- Update 31/3/17 – Github repo updated with **Angular 4**, TypeScript 2.1. Angular 4 is backward compatible with Angular 2 with much-reduced bundle sizes.
- Update 26/9 – Github repo updated with Angular 2 Final release version. Install TypeScript 2.0 RC

These steps can be used for new or existing MVC 5 application.

In this post, I will summarize the steps needed to getting started with Angular 2 in MVC 5.

> *ASP.NET MVC 5 is full .NET framework web development framework, it's different from ASP.NET Core 1.0*

What will we learn?

1. Adding package.json to MVC 5
2. Configure to transpile TypeScript files
3. Using gulpfile.js to move files.

4. Add TypeScript files for bootstrapping
5. Include systemjs.config.js to load Angular 2 modules
6. Change HTML to load and render Angular

## Step 1: Adding ***package.json*** to ASP.NET MVC 5

Assuming that you already have existing or created new ASP.NET MVC 5. Let's add ***NPM configuration file*** known as *package.json*. It contains Angular 4 (works for Angular 2) & related package name to installed using NPM (Node). This is similar to package.config of NuGet.

> *Latest NodeJS & NPM needs to be installed.*

```json
{
  "version": "1.0.0",
  "name": "aspnet",
  "private": true,
  "scripts": {},
  "dependencies": {
    "@angular/animations": "4.3.5",
    "@angular/common": "4.3.5",
    "@angular/compiler": "4.3.5",
    "@angular/compiler-cli": "4.3.5",
    "@angular/core": "4.3.5",
    "@angular/forms": "4.3.5",
    "@angular/http": "4.3.5",
    "@angular/platform-browser": "4.3.5",
    "@angular/platform-browser-dynamic": "4.3.5",
    "@angular/platform-server": "4.3.5",
    "@angular/router": "4.3.5",
    "@angular/upgrade": "4.3.5",
    "angular-in-memory-web-api": "0.3.2",
    "bootstrap": "3.3.7",
    "core-js": "2.5.0",
    "ie-shim": "0.1.0",
    "rxjs": "5.4.3",
    "zone.js": "0.8.16",
    "systemjs": "^0.20.18"
  },
  "devDependencies": {
    "gulp": "^3.9.1",
    "gulp-clean": "^0.3.2",
    "gulp-concat": "^2.6.1",
    "gulp-tsc": "~1.3.2",
    "gulp-typescript": "^3.2.2",
    "path": "^0.12.7",
    "typescript": "^2.4.2"
  }
}
```

**package.**json contains Angular 2 (using version 4) along with, system.js, RxJs and also some dev dependencies.

Open Command Prompt & navigate to package.json location, then run ***npm install*** this will install packages related to Angular 2 (using version 4) under *node_modules* folder in your folder structure.

They won't be showing in project solution explorer, don't worry they need not show.

## Step 2: Configure to transpile TypeScript files

TypeScript(TS) would be new for most of the developers, maybe these will give get started on TypeScript

> *In short – It's superset of JavaScript, means everything you know about JS will be in use.*

All TS files need to be transpiled or compiled to JS files so that we can run them on browser. To accomplish this we need to add "TypeScript Configuration File" called as *tsconfig.json*

Create a folder called "**tsScripts**" which contains all TS files and also configuration file. Create above tsconfig.json in this folder.

```
{
  "compilerOptions": {
    "emitDecoratorMetadata": true,
    "experimentalDecorators": true,
    "module": "commonjs",
    "noEmitOnError": true,
    "noImplicitAny": false,
    "outDir": "../Scripts/",
    "removeComments": false,
    "sourceMap": true,
    "target": "es5",
    "moduleResolution": "node",
    "typeRoots": [
      "./node_modules/@types",
      "./node_modules"
    ],
    "types": [
      "node"
]
  },
  "exclude": [
    "node_modules"
  ]
}
```

It’s fairly simple config “All TS files present in ***tsScripts*** folder will be transpiled using ***commonjs*** module to ***outDir*** (Output Directory) by keeping comments, sourceMap intact”

One of the important step is to create ***typings.json*** file, this file will create typings to ensure that TypeScript understands all Angular 2 (using version 4) modules in respect to ES5 standard.

Create JSON file with name “***typings.json***” & add below code then run command **typings install** from CMD

```
{
  "globalDependencies": {
    "core-js": "registry:dt/core-js#0.0.0+20160725163759",
    "jasmine": "registry:dt/jasmine#2.2.0+20160621224255",
    "node": "registry:dt/node#6.0.0+20160909174046"
  }
}
```

## Step 3: Using gulpfile.js to move files

From Step 1 you got to know that all Angular 2, other packages are downloaded into node_modules in your solution folder. Now we need to move required files only like JS, sourcemaps (debugging on chrome) into our MVC 5 apps Scripts folder.

Step 2 also requires to move TS files to JavaScript file, so we use create GULP tasks which does this transpile. Also it contains CSS files movement also (which is not relevant at this point)

Create gulpfile.js in your project and copy this code

```
var ts = require('gulp-typescript');
var gulp = require('gulp');
var clean = require('gulp-clean');

var destPath = './libs/';

// Delete the dist directory
gulp.task('clean', function () {
    return gulp.src(destPath)
        .pipe(clean());
});

gulp.task("scriptsNStyles", function() {
    gulp.src([
            'core-js/client/*.js',
            'systemjs/dist/*.js',
            'reflect-metadata/*.js',
            'rxjs/**',
            'zone.js/dist/*.js',
            '@angular/**/bundles/*.js',
            'bootstrap/dist/js/*.js'
    ], {
        cwd: "node_modules/**"
    })
        .pipe(gulp.dest("./libs"));
});

var tsProject = ts.createProject('tsScripts/tsconfig.json', {
    typescript: require('typescript')
});
gulp.task('ts', function (done) {
    //var tsResult = tsProject.src()
    var tsResult = gulp.src([
            "tsScripts/*.ts"
    ])
        .pipe(tsProject(), undefined, ts.reporter.fullReporter());
    return tsResult.js.pipe(gulp.dest('./Scripts'));
});

gulp.task('watch', ['watch.ts']);

gulp.task('watch.ts', ['ts'], function () {
    return gulp.watch('tsScripts/*.ts', ['ts']);
});

gulp.task('default', ['scriptsNStyles', 'watch']);
```

## Step 4: Add TypeScript files for bootstrapping

As we are using Angular 2 with TypeScript, we need to add TS file to bootstrap the Angular app into MVC 5 app. So let's create boot.ts file in "TsScripts" folder and copy this code.

We using ngModule to browser specifics, AppComponent.

```
///<reference path="./../typings/globals/core-js/index.d.ts"/>
import { NgModule } from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';
import { AppComponent } from './app';

@NgModule({
    imports: [BrowserModule ],
    declarations: [AppComponent],
    bootstrap: [AppComponent]
})
export class AppModule { }
```

AppComponent is starter component which we have to create it now (app.ts). Add this code below

```
import { Component } from '@angular/core';
@Component({
    selector: 'my-app',
    template: `
    <h2>My favorite skill is: {{myskills}}</h2>
    <p>Skill:</p>
    <ul>
      <li *ngFor="let skl of skills">
        {{ skl }}
      </li>
    </ul>
  `
})
export class AppComponent {
    title = 'ASP.NET MVC 5 with Angular 2';
    skills = ['MVC 5', 'Angular 2', 'TypeScript', 'Visual Studio 2015'];
    myskills = this.skills[1];
}
```

This is template based app component which displays list.

Now create **tsScripts/main.ts**, this entry point where Angular 2 loads the components.

```
import { platformBrowserDynamic } from '@angular/platform-browser-dynamic';
import { AppModule } from './boot';
const platform = platformBrowserDynamic();
platform.bootstrapModule(AppModule);
```

## Step 5: Include *systemjs.config.js* to load modules

This is most important part of Angular 2 (using version 4) which loads it into the browser. There are different ways to load it but am using SystemJS here.

In the existing "***Scripts***" folder, create "***systemjs.config.js***" and copy below code

```
/**
 * System configuration for Angular samples
 * Adjust as necessary for your application needs.
 */
(function (global) {
    System.config({
        paths: {
            // paths serve as alias
            'npm:': '/libs/'
        },
        // map tells the System loader where to look for things
        map: {
            // our app is within the app folder
            app: '/Scripts',
            // angular bundles
            '@angular/core': 'npm:@angular/core/bundles/core.umd.js',
            '@angular/common': 'npm:@angular/common/bundles/common.umd.js',
            '@angular/compiler': 'npm:@angular/compiler/bundles/compiler.umd.js',
            '@angular/platform-browser': 'npm:@angular/platform-browser/bundles/platform-browser.umd.js',
            '@angular/platform-browser-dynamic': 'npm:@angular/platform-browser-dynamic/bundles/platform-browser-dynamic.umd.js
            '@angular/http': 'npm:@angular/http/bundles/http.umd.js',
            '@angular/router': 'npm:@angular/router/bundles/router.umd.js',
            '@angular/forms': 'npm:@angular/forms/bundles/forms.umd.js',
            // other libraries
            'rxjs': 'npm:rxjs',
            'angular-in-memory-web-api': 'npm:angular-in-memory-web-api/bundles/in-memory-web-api.umd.js',
        },
        // packages tells the System loader how to load when no filename and/or no extension
        packages: {
            app: {
                main: './main.js',
                defaultExtension: 'js',
            },
            rxjs: {
                defaultExtension: 'js'
            }
        }
    });
})(this);
```

## Step 6: Change csHTML to load and render Angular 4

To load Angular 2 in ASP.NET MVC 5, we need to include the script references in ***_Layout*** file and index.cshtml page.

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>@ViewBag.Title - My ASP.NET Application</title>

    <!-- 1. Load libraries -->
    <!-- Polyfill(s) for older browsers -->

    <script src="~/libs/core-js/client/shim.min.js"></script>
    <script src="~/libs/zone.js/dist/zone.js"></script>
    <script src="~/libs/systemjs/dist/system.src.js"></script>

    <!-- 2. Configure SystemJS -->
    <script src="~/Scripts/systemjs.config.js"></script>
    <script>
        System.import('../Scripts/main').catch(function (err)
        {
            console.error(err);
        });
    </script>

    @Styles.Render("~/Content/css")
    @Scripts.Render("~/bundles/modernizr")
</head>
<body>
    <div class="navbar navbar-inverse navbar-fixed-top">
        <div class="container">
            <div class="navbar-header">
                <button type="button" class="navbar-toggle" data-toggle="collapse" data-target=".navbar-collapse">
                    <span class="icon-bar"></span>
                    <span class="icon-bar"></span>
                    <span class="icon-bar"></span>
                </button>
                @Html.ActionLink("Application name", "Index", "Home", new { area = "" }, new { @class = "navbar-brand" })
            </div>
            <div class="navbar-collapse collapse">
                <ul class="nav navbar-nav">
                    <li>@Html.ActionLink("Home", "Index", "Home")</li>
                    <li>@Html.ActionLink("About", "About", "Home")</li>
                    <li>@Html.ActionLink("Contact", "Contact", "Home")</li>
                </ul>
            </div>
        </div>
    </div>
    <div class="container body-content">
```

```
        @RenderBody()
        <hr />
        <footer>
            <p>&copy; @DateTime.Now.Year - My ASP.NET Application</p>
        </footer>
    </div>

    @Scripts.Render("~/bundles/jquery")
    @Scripts.Render("~/bundles/bootstrap")
    @RenderSection("scripts", required: false)
</body>
</html>
```

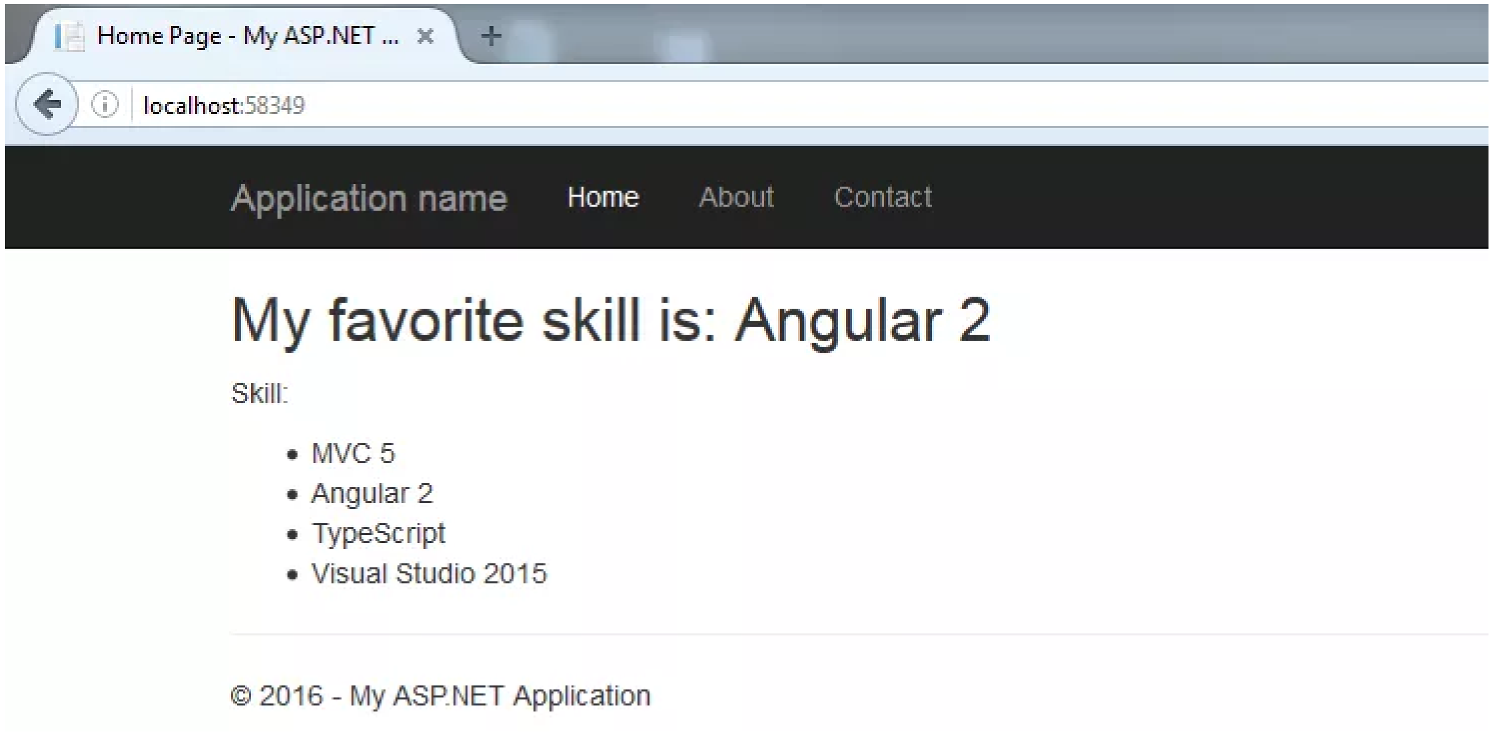
In the Views/Home/index.cshtml you need to add “**my-app**” component we defined in app.TS file. This is the starting point of Angular 2 application to render into the browser.

```
@{
    ViewBag.Title = "Home Page";
}

<my-app>Loading...</my-app>
```

Now that we are almost done, we need to run GULP tasks so that Angular 2 files, TS files are moved to an appropriate folder. Open ***Task Runner Explorer*** in Visual Studio 2015 and run **default** task shown.

Its better to do show ALL Files in solution Explorer to see “*libs*“, “*typings*“, “*app.js, boot.js, *.map*” files. Includes these files and run the application to load Angular 2 in ASP.NET MVC 5.



My favorite skill is: Angular 2

Skill:

- MVC 5
- Angular 2
- TypeScript
- Visual Studio 2015

© 2016 - My ASP.NET Application

*Running Angular 2 in ASP.NET MVC 5*

The entire source code is on my Github repo, clone or fork or download it and follow instructions to run it.

Let me know in comments if you face any issues running applications.