001 Angular2 - getting started

TypeScript Kata List – Blog Page

[TypeScript Kata List on GitHub](https://github.com/robertdunaway/katas-typescript)

# Duration

30 minutes

# Brief

This is a very labor intensive tutorial for setting up a project for Angular2. You can take the time to work through this or skip to the next kata where all this work has been done for you.

### For More Information

BING/GOOGLE: “Angular2 ”

https://www.ng-book.com/2/

# Instructions

Get tutorial folder or the entire katas-angular2 repo.

Open the [before/\*.sln] file and execute the kata.

Feel free to execute this kata multiple times because repetition creates motor memory.

# Github

* Before (start kata with this)
  + https://github.com/robertdunaway/katas-angular2/tree/master/001%20Angular2%20-%20getting%20started/before
* After
  + https://github.com/robertdunaway/katas-angular2/tree/master/001%20Angular2%20-%20getting%20started/after

# Kata

This isn’t exactly a Kata as much as a tutorial on getting started. We already have a base project with TypeScript compilation by Gulp.

Now let’s add a few Javascript libraries that we will be using.

Add the following libraries to the package.json file. These are recommended by Google but some can be swapped out for something you prefer. IE: SystemJS is a module loader. Other module loaders like WebPack are preferred by some and can be used instead.

"angular2": "2.0.0-beta.0",

"systemjs": "0.19.6",

"es6-promise": "^3.0.2",

"es6-shim": "^0.33.3",

"reflect-metadata": "0.1.2",

"rxjs": "5.0.0-beta.0",

"zone.js": "0.5.10"

After adding these our package.json will look like this. It’s getting lengthy but does a lot for us. A few of these libraries might go away if we find we never use them.

{

"version": "1.0.0",

"name": "ASP.NET",

"private": true,

"devDependencies": {

"gulp": "^3.9.0",

"gulp-clean": "^0.3.1",

"gulp-minify-html": "^1.0.5",

"gulp-newer": "^1.1.0",

"gulp-plumber": "^1.0.1",

"gulp-rename": "^1.2.2",

"gulp-sourcemaps": "^1.6.0",

"gulp-tsd": "0.0.4",

"gulp-tslint": "^4.3.0",

"gulp-tslint-stylish": "^1.1.1",

"gulp-typescript": "^2.10.0",

"gulp-uglify": "^1.5.1",

"gulp-watch": "^4.3.5",

"merge": "^1.2.0",

"reporters": "0.0.4",

"run-sequence": "^1.1.5",

"typescript": "^1.7.5",

"angular2": "2.0.0-beta.0",

"systemjs": "0.19.6",

"es6-promise": "^3.0.2",

"es6-shim": "^0.33.3",

"reflect-metadata": "0.1.2",

"rxjs": "5.0.0-beta.0",

"zone.js": "0.5.10"

}

}

Now that our libraries are installed we need to give access to the application at run time. To accomplish this we will copy the libraries to the wwwroot folder with Gulp.

Add the following to your Gulp file so they are copied to wwwroot.

// for angular2

, 'node\_modules/\*\*//es6-shim/es6-shim.min.js'

, 'node\_modules/\*\*//angular2/bundles/angular2-polyfills.min.js'

, 'node\_modules/\*\*//systemjs/dist/system.src.js'

, 'node\_modules/\*\*//rxjs/bundles/rx.min.js'

, 'node\_modules/\*\*//angular2/bundles/angular2.min.js'

The lib task in the Gulp file should look like this:

gulp.task('libs', function () {

return gulp.src(['bower\_components/\*\*//normalize-css/normalize.css'

, 'bower\_components/\*\*//font-awesome/css/font-awesome.min.css'

, 'bower\_components/\*\*/font-awesome/fonts/\*.\*'

, 'bower\_components/\*\*//jquery/dist/jquery.min.js'

, 'bower\_components/\*\*//lodash/lodash.min.js'

// for angular2

, 'node\_modules/\*\*//es6-shim/es6-shim.min.js'

, 'node\_modules/\*\*//angular2/bundles/angular2-polyfills.min.js'

, 'node\_modules/\*\*//systemjs/dist/system.src.js'

, 'node\_modules/\*\*//rxjs/bundles/rx.min.js'

, 'node\_modules/\*\*//angular2/bundles/angular2.min.js'

])

.pipe(plumber({

errorHandler: onError

}))

.pipe(gulp.dest('wwwroot/lib/./'));

});

Add references to the new libraries in the index.html file.

<script src="/lib/es6-shim/es6-shim.min.js"></script>

<script src="/lib/angular2/bundles/angular2-polyfills.min.js"></script>

<script src="/lib/systemjs/dist/system.src.js"></script>

<script src="/lib/rxjs/bundles/rx.min.js"></script>

<script src="/lib/angular2/bundles/angular2.min.js"></script>

Install the TSD for Angular2

tsd query angular2 --action install –save

tsd query systemjs --action install --save

Let’s go ahead and make sure we have a working application.

Don’t be concerned about what you do and do not understand at this point. Let’s just get things working.

1. Open the app.ts file.
2. Enter the following code.

import { bootstrap } from 'angular2/platform/browser';

import { Component } from 'angular2/core';

@Component({

selector: 'hello-world',

template: `

<div>

Hello world

</div>

`

})

class HelloWorld { }

bootstrap(HelloWorld);

# Next

Take a few minutes and imagine more examples.