Setting up the Project

Reggie Dawson
WEB DEVELOPER

@reggiewrites

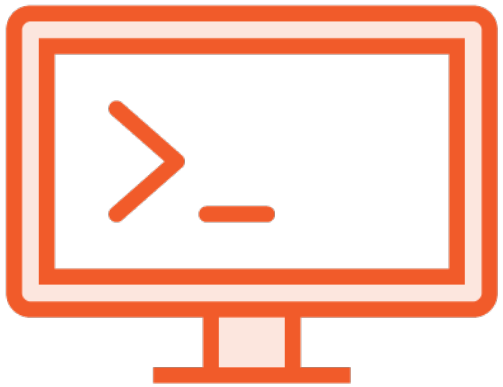
Introduction

Configure Angular 2

Webpack and TypeScript

Routing

Setting up the Project



Angular CLI

Allows you to generate Angular 2 Components, Services, and Routes

Starter Project

Webpack and Foundation based project, manual install

Project Files

karma.conf.js

```
module.exports = require('./config/karma.conf.js');
```

Project Files

package.json

```
{
   "name": "angular2-webpack-starter",
   "version": "1.0.0",
   "description": "A webpack starter for Angular",
   "scripts": {
      "start": "webpack-dev-server --inline –progress
         --port 8080",
      "test": "karma start",
      "build": "rimraf dist && webpack --config
         config/webpack.prod.js --progress --profile --bail",
 },
```

Project Files

package.json(continued)

```
"dependencies": {
    "@angular/common": "~2.1.0",
    "@angular/compiler": "~2.1.0",
    "@angular/core": "~2.1.0",
    "@angular/forms": "~2.1.0",
    "@angular/http": "~2.1.0",
    "@angular/platform-browser": "~2.1.0",
    "@angular/platform-browser-dynamic": "~2.1.0",
    "@angular/router": "~3.1.0",
    "core-js": "^2.4.1",
    "rxjs": "5.0.0-beta.12",
    "zone.js": "^0.6.25",
    "foundation-sites": "^6.2.4",
    "jquery": "^3.1.1"},
```

Project Files

package.json(continued)

```
"devDependencies": {
    "angular2-template-loader": "^0.4.0",
    "awesome-typescript-loader": "^2.2.4",
    "css-loader": "^0.23.1",
    "extract-text-webpack-plugin": "^1.0.1",
    "file-loader": "^0.8.5",
    "html-loader": "^0.4.3",
    "html-webpack-plugin": "^2.15.0",
    "jasmine-core": "^2.4.1",
    "karma": "^1.2.0",
    "karma-jasmine": "^1.0.2",
    "karma-phantomjs-launcher": "^1.0.2",
    "karma-sourcemap-loader": "^0.3.7",
    "karma-webpack": "^1.8.0",
```

Project Files

package.json(continued)

```
"null-loader": "^0.1.1",
"phantomjs-prebuilt": "^2.1.7",
"raw-loader": "^0.5.1",
"rimraf": "^2.5.2",
"style-loader": "^0.13.1",
"typescript": "^2.0.2",
"webpack": "^1.13.0",
"webpack-dev-server": "^1.14.1",
"webpack-merge": "^0.14.0",
"node-sass": "^3.10.1",
"sass-loader": "^4.0.2",
"script-loader": "^0.7.0“,
```

Project Files

package.json(continued)

```
    "@types/core-js": "^0.9.34",
    "@types/jasmine": "^2.5.38",
    "@types/node": "^6.0.51"
    }
}
```

Project Files

tsconfig.json

```
{
    "compilerOptions": {
        "target": "es5",
        "module": "commonjs",
        "moduleResolution": "node",
        "sourceMap": true,
        "emitDecoratorMetadata": true,
        "experimentalDecorators": true,
        "removeComments": false,
        "noImplicitAny": true,
        "suppressImplicitAnyIndexErrors": true
    }
}
```

Project Files

webpack.config.js

```
module.exports = require('./config/webpack.dev.js');
```

Project Files

helpers.js

```
var path = require('path');

var _root = path.resolve(__dirname, '..');

function root(args) {
  args = Array.prototype.slice.call(arguments, 0);
  return path.join.apply(path, [_root].concat(args));
}

exports.root = root
```

Project Files

karma-test-shim.js

```
Error.stackTraceLimit = Infinity;
require('core-js/es6');
require('core-js/es7/reflect');
require('zone.js/dist/zone');
```

karma.conf.js

```
var webpackConfig = require('./webpack.test');

module.exports = function (config) {
  var _config = {
    basePath: '',
```

Project Files

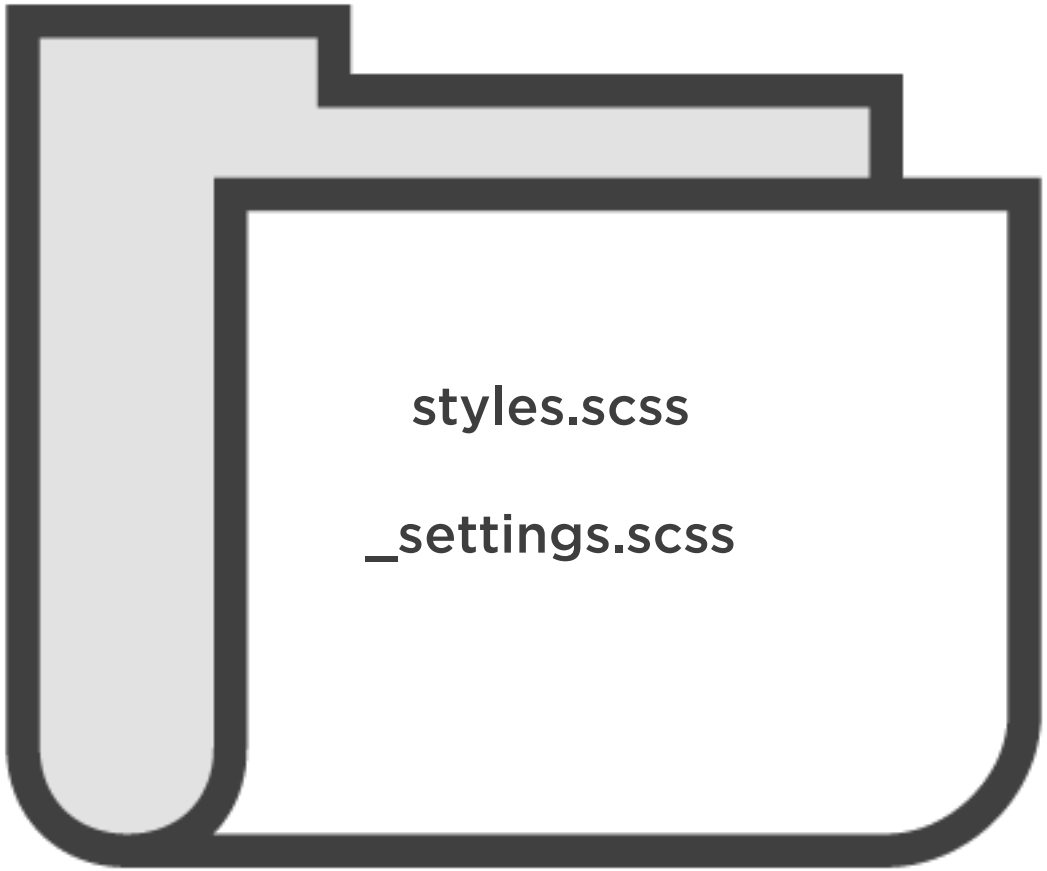
webpack.common.js

webpack.dev.js

webpack.prod.js

webpack.test.js

Project Files



sass folder

Project Files

index.html

```
<!DOCTYPE html>
<html>
  <head>
    <base href="/">
    <title>Angular With Webpack</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1">
  </head>
  <body>
    <my-app>Loading...</my-app>
  </body>
</html>
```

Project Files

main.ts

```
if (process.env.ENV === 'production') {enableProdMode();}
platformBrowserDynamic().bootstrapModule(AppModule);
```

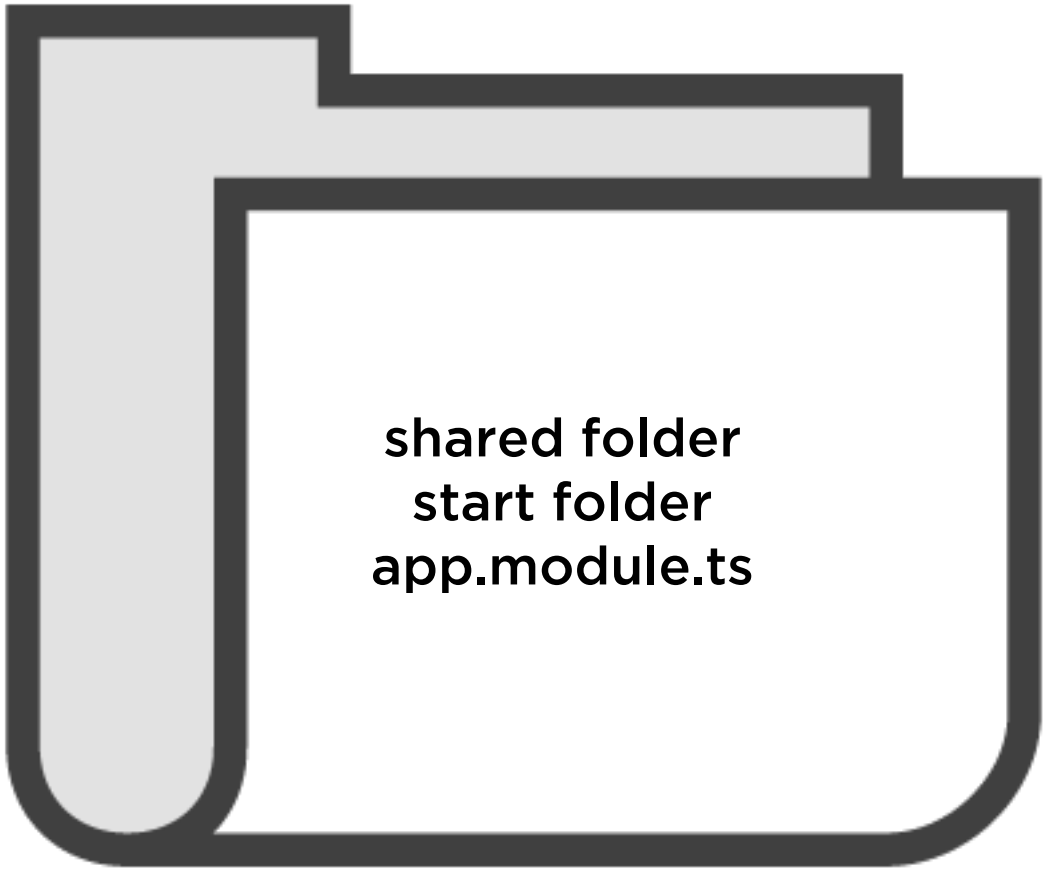
polyfills.ts

```
import 'core-js/es6';
import 'core-js/es7/reflect';
require('zone.js/dist/zone');
```

vendor.ts

```
import '@angular/platform-browser';
import '@angular/platform-browser-dynamic';
import '@angular/core';
import '@angular/common';
import '@angular/http';
import '@angular/router';
```

Project Files



app folder

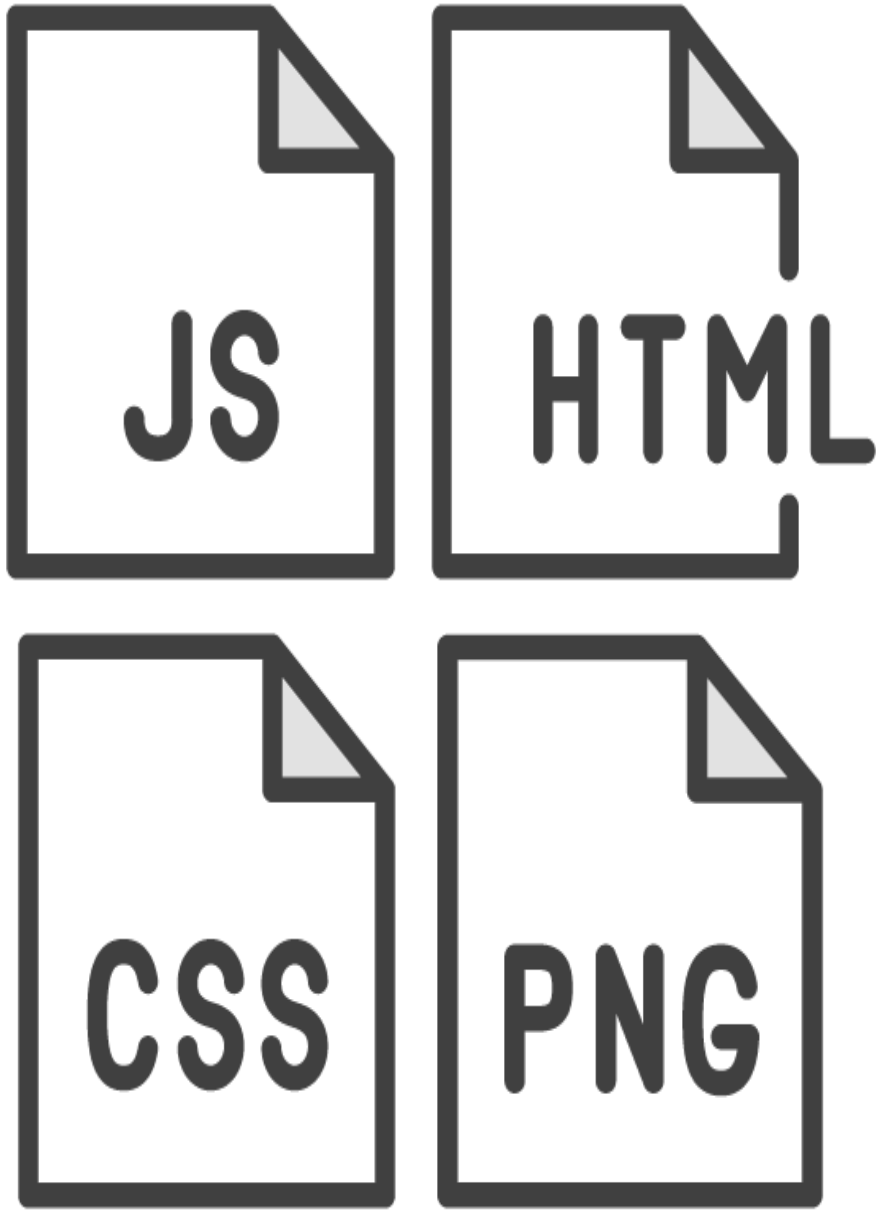
Webpack

Module Bundler

Alternate to system.js

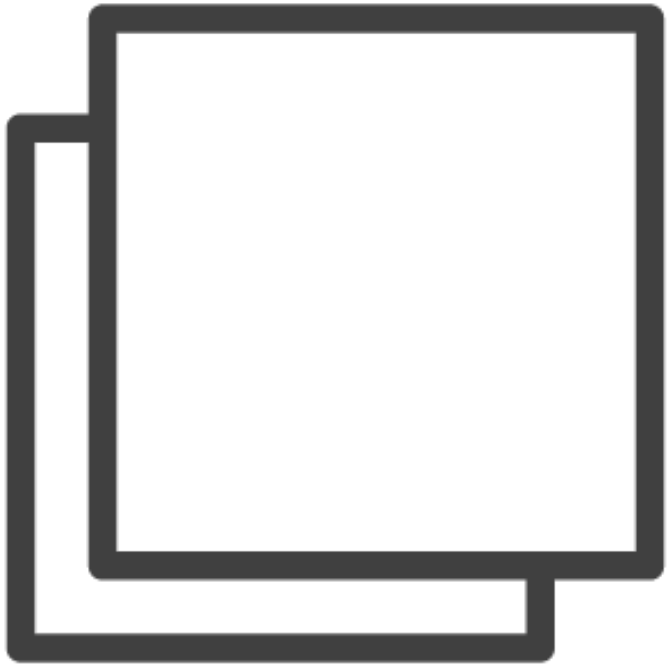
Dynamically add scripts

JavaScript and CSS Files



Webpack can handle various file types

Loaders will handle compilation



Compile modules into a bundle

Imports vendor libraries such as Angular 2, Foundation, and jQuery

Webpack Configuration

webpack.common.js

```
var webpack = require('webpack');
var HtmlWebpackPlugin = require('html-webpack-plugin');
var ExtractTextPlugin = require('extract-text-webpack-plugin');
var helpers = require('./helpers');
```

Webpack Configuration

webpack.common.js (continued)

```
module.exports = {
  entry: {
    'polyfills': './src/polyfills.ts',
    'vendor': './src/vendor.ts',
    'app': './src/main.ts'
  },
```

Webpack Configuration

polyfills.ts

```
import 'core-js/es6';
import 'core-js/es7/reflect';
require('zone.js/dist/zone');

if (process.env.ENV === 'production') {
  // Production
} else {
  // Development
  Error['stackTraceLimit'] = Infinity;
  require('zone.js/dist/long-stack-trace-zone');
}
```

Webpack Configuration

vendor.ts

```
// Angular
import '@angular/platform-browser';
import '@angular/platform-browser-dynamic';
import '@angular/core';
import '@angular/common';
import '@angular/http';
import '@angular/router';
// RxJS
import 'rxjs';
// Other vendors for example jQuery, Lodash or Bootstrap
// You can import js, ts, css, sass, ...
import 'script!jquery‘;
import 'foundation-sites/dist/foundation.js';
import '../public/sass/styles.scss';
```

Webpack Configuration

styles.scss

```
@import 'settings';
@import '~foundation-sites/assets/foundation-flex';
```

Webpack Configuration

webpack.common.js (continued)

```
module.exports = {
  entry: {
    'polyfills': './src/polyfills.ts',
    'vendor': './src/vendor.ts',
    'app': './src/main.ts'
  },
```

Webpack Configuration

webpack.common.js (continued)

```
resolve: {
    modulesDirectories: ['node_modules'],
    extensions: ['', '.js', '.ts', '.scss']
},
```

Webpack Configuration

webpack.common.js (continued)

```
module: {
    loaders: [
      {
        test: /\.ts$/,
        loaders: ['awesome-typescript-loader', 'angular2-template-
loader']
      },
      {
        test: /\.html$/,
        loader: 'html'
      },
```

Webpack Configuration

webpack.common.js (continued)

```
{
    test: /\.(png|jpe?g|gif|svg|woff|woff2|ttf|eot|ico)$/,
    loader: 'file?name=assets/[name].[hash].[ext]'
},
{
    test: /\.scss$/,
    exclude: helpers.root('src', 'app'),
    loader: ExtractTextPlugin.extract('style', 'css!sass')
},
{
    test: /\.css$/,
    include: helpers.root('src', 'app'),
    loader: 'raw'
}
```

Webpack Configuration

webpack.common.js (continued)

```
    plugins: [
        new webpack.optimize.CommonsChunkPlugin({
            name: ['app', 'vendor', 'polyfills']
        }),

        new HtmlWebpackPlugin({
            template: 'src/index.html'
        })
    ]
};
```

Webpack Configuration

webpack.dev.js

```
var webpackMerge = require('webpack-merge');
var ExtractTextPlugin = require('extract-text-webpack-plugin');
var commonConfig = require('./webpack.common.js');
var helpers = require('./helpers');
```

Webpack Configuration

webpack.dev.js (continued)

```
module.exports = webpackMerge(commonConfig, {
    devtool: 'cheap-module-eval-source-map',

    output: {
        path: helpers.root('dist'),
        publicPath: 'http://localhost:8080/',
        filename: '[name].js',
        chunkFilename: '[id].chunk.js'
    },
```

Webpack Configuration

webpack.dev.js (continued)

```
    plugins: [
        new ExtractTextPlugin('[name].css')
    ],

    devServer: {
        historyApiFallback: true,
        stats: 'minimal'
    }
});
```

Webpack

Webpack Production Configuration

Webpack Test Configuration

Angular 2 Development Languages

Dart

JavaScript

TypeScript

Typescript

Compiles to standard JavaScript

Uses ES6 syntax

Static Typing

Easy to learn if you know JavaScript

```
let isDone: boolean = false;
```

◀ **boolean**

```
let decimal: number = 6;
let hex: number = 0xf00d;
let binary: number = 0b1010;
let octal: number = 0o744;
```

◀ **number**

```
let color: string = 'blue';

let sentence: string = `My
favorite color is ${ color }.`
```

◀ string

```
let list: number[] = [1, 2,
3];

let list: Array<number> = [1,
2, 3];
```

◀ array

```
let theValue: any = 5;
```

◀ any

```
function a(){
    let first = 1;
    if(first) {
        let second = 2;
    }
    return second;
}
```

- scoped to if statement
- error, not available

```
const appID = 123456;
```

◀ cannot be changed

```
@Component({
    selector: 'my-app',
    templateUrl:
'./app.component.html',
    styleUrls:
['./app.component.css']
})
```

◀ Decorator

```
class Greeter {
    greeting: string;
    constructor(message: string){
        this.greeting = message;
    }
    greet() {
    return `Hello,this.greeting`;
     }
}
```

◀ class

```
interface LabelledValue {
    label: string;
}

function printLabel(labelledObj:
LabelledValue) {
   console.log(labelledObj.label);
}
let myObj = {size: 10,
    label:"Size 10 Object"};

printLabel(myObj);
```

- interface

- Size is optional, label required

```
export class someClass {

}
```

◀ export

```
import { someClass } from
"./someClass";
```

◀ import

TypeScript Compiler Configuration

tsconfig.json

```
{
    "compilerOptions": {
        "target": "es5",
        "module": "commonjs",
        "moduleResolution": "node",
        "sourceMap": true,
        "emitDecoratorMetadata": true,
        "experimentalDecorators": true,
        "removeComments": false,
        "noImplicitAny": true,
        "suppressImplicitAnyIndexErrors": true
    }
}
```

The Root Module

app.module.ts

Root Module

Multiple modules

Must have Root Module!

The Root Module

app.module.ts

```
import { NgModule } from '@angular/core';
import { BrowserModule }  from '@angular/platform-browser';
import { AppComponent } from './start/app.component';
```

The Root Module

app.module.ts (continued)

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

Angular 2 Bootstrap

main.ts
Runs the bootstrap method against app.module.ts

app.module.ts
Launches app.component.ts

app.component.ts
Initial component displayed by app

The Root Module

app.module.ts

```
@NgModule({
    imports: [
        BrowserModule
    ],
    declarations: [
        AppComponent
    ],
    bootstrap: [ AppComponent ],
    exports: [ AppComponent ],
    providers: [ MyService ]


})
```

The Root Module

app.module.ts (continued)

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

AppComponent Files

app.component.ts

app.component.css

app.component.html

app.component.spec.ts

app.component.ts

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


@Component({

  selector: 'my-app',

  templateUrl:
'./app.component.html',

  styleUrls:
['./app.component.css']

})

export class AppComponent { }
```

◀ import

◀ component decorator

◀ selector

◀ templateUrl

◀ styleUrls

◀ exported class

app.component.html

```
<h1>Hello from Angular App with
Webpack</h1>
```

◀ Displays message when app runs

AppComponent Files

app.component.css

Component level styles

app.component.spec.ts

Karma test file

main.ts

Entry point to app bundle

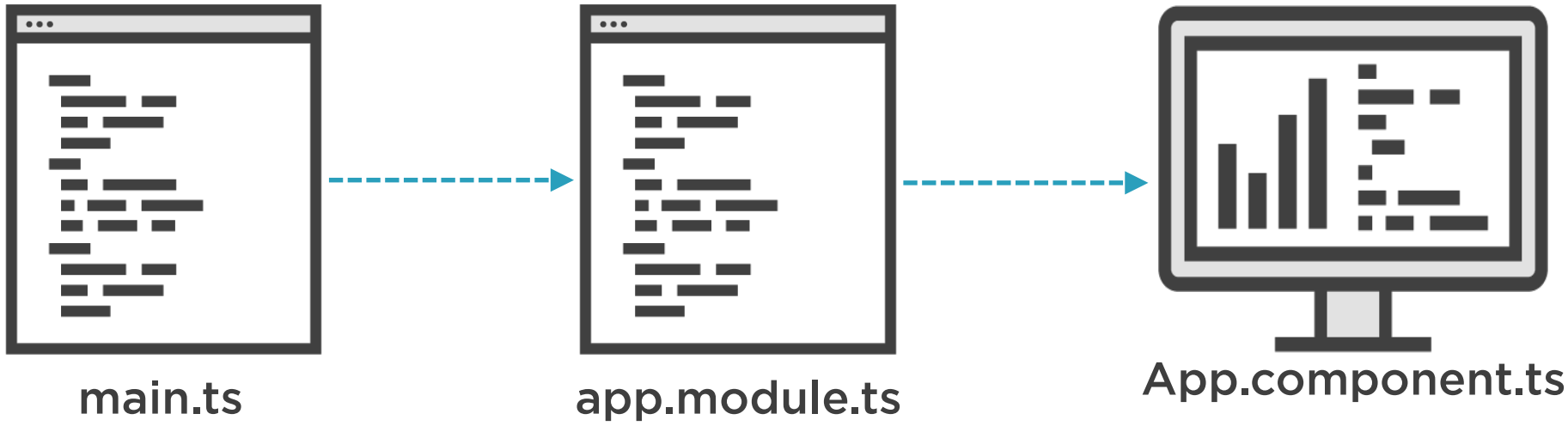
Bootstrap

Bootstrap the App

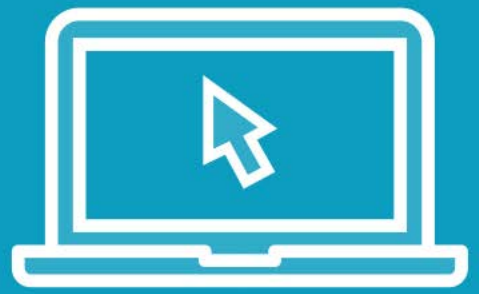
main.ts

```
import { platformBrowserDynamic } from '@angular/platform-
browser-dynamic';

import { enableProdMode } from '@angular/core';

import { AppModule } from './app/app.module';

if (process.env.ENV === 'production') {

  enableProdMode();

}

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

Angular 2 App



Demo



Preview app

Routing configuration

Check for errors

Summary

Working app

Angular 2 syntax

Routing configured

Next up: Adding Authentication