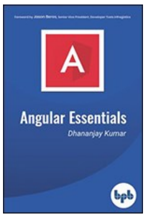


Angular Essentials

Dhananjay Kumar

Angular Essentials: The Essential Guide to Learn Angular

by Dhananjay Kumar

BPB Publications. (c) 2019. Copying Prohibited.

Chapter 1: Introduction

In this chapter, you will learn about what is Angular, how it is used, and setting up the development environment. Following topics will be covered in this chapter:

- What is Angular
- How is Angular different from AngularJS
- Angular's basic architecture
- Angular CLI
- Creating your First App

What is Angular

Angular is a widely used web application platform and framework created and maintained by Google. It serves as a total rewrite to AngularJS, and the *Angular* name is meant to include all versions of the framework starting from 2 and up.

TypeScript is the core of Angular, being the language upon which Angular is written. As such, Angular implements major and core functionalities as TypeScript libraries while building client applications with additional HTML.

For a variety of reasons, Angular has grown in popularity with developers. It lends itself to maintenance ease with its component and class-based system, modular building, hierarchical structure, and simple, declarative templates. Furthermore, its cross-platform capabilities are advantageous to enterprise and SMB developers, including its speed with server-side rendering.

Angular is a structural framework, which makes it easy to build dynamic web apps. It is a platform that gives developers the power to build applications that reside on web, mobile, or the desktop. In 2009, Angular started as a side-project by Misko Hevery and Adam Abrons to help developers build applications using simple HTML tags. It was named Angular due to the fact that HTML tags are surrounded by these angle brackets **<>**. The team of developers at Google then kept on releasing the updated versions of Angular and at the time of writing this, current version being used is 7.0.

This essential book will go over the essential pieces of Angular and the main concepts behind working with the ever-growing platform for web-based applications.

How is Angular Different from AngularJS?

In the past, you might have worked with or learned about AngularJS. There are a few main differences between the two that you need to know about:

- **Modularity:** More of Angular's core functionalities have moved to modules.
- **Hierarchy:** Angular has an architecture built around a hierarchy of components.
- **Syntax:** Angular has a different expression syntax for event and property binding.
- **Dynamic loading:** Angular will load libraries into memory at runtime, retrieve and execute functions, and then unload the library from memory.
- **Iterative callbacks:** Using RxJS, Angular makes it easier to compose asynchronous or callback-based code.
- **Asynchronous template compilation:** Angular, without controllers and the concept of scope, makes it easier to pause template rendering and compile templates to generate the defined code.
- **TypeScript:** Angular includes ES6 and its superset, TypeScript.

Angular's Basic Architecture

Here's a brief overview of the architecture involved and the building blocks that I'll cover in this piece:

- **NgModules:** Declares a compilation context for a set of components that is dedicated to an application domain, a workflow, or a related set of capabilities.
- **Components:** Defines a class that contains application data and logic and works with an HTML template that defines a view.
- **Template:** Combines HTML with Angular markup that can modify HTML elements before they're displayed.
- **Directive:** Attaches custom behavior to elements in the DOM.
- **Two-way data binding:** Coordinates the parts of a template with the parts of a component.
- **Services:** Typically, a class used to increase modularity and reusability with a narrow and well-defined purpose.
- **Dependency injection:** Provides components with needed services and gives access to a service class.
- **Routing:** Defines a navigation path among the different application states lets you view application hierarchies.

Angular CLI

Setting up an Angular project requires many steps, such that:

- Setting up TypeScript compiler
- Setting up Webpack
- Setting up local web development server
- Configuring Unit Test environment

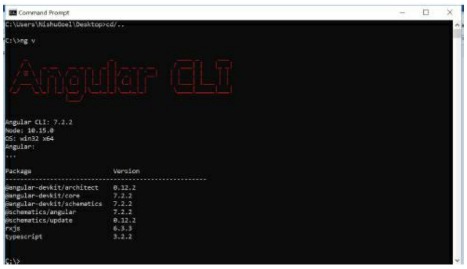
All these tasks are taken care by Angular CLI. Angular Command Line Interface is a command line tool for creating Angular apps. It is recommended to use angular CLI for creating angular apps as you don't need to spend time installing and configuring all the required dependencies and wiring everything together. It provides you with boilerplates and saves your time. It uses Webpack to include all the packaging, importing, BrowserLink etc. all the Webpack configuration is done completely by CLI and the developer needs not worry about it then. It also configures Jasmine and Karma for unit tests and TypeScript complier to transpile TypeScript file to JavaScript. You can install Angular CLI globally on local development machine using npm. To work with npm, make sure to install NodeJS from here : https://nodejs.org/en/. Once NodeJS is installed, you can use npm to install Angular CLI:

```
npm install  -g @angular/cli
```

Here, npm install command is to use npm to install angular CLI, -g is used to install CLI globally in your machine. After the process completes, you can check if it is successfully installed and also the version of the CLI is installed. To check that, we use the following command:

```
    ng v or ng version
```

This gives a preview like in *figure 1.1*.



```
C:\Users\NishuGoel\Desktop>cd/..

C:\>ng v

Angular CLI

Angular CLI: 7.2.2
Node: 10.15.0
OS: win32 x64
Angular:
...

Package                      Version
------------------------------------------------------
@angular-devkit/architect    0.12.2
@angular-devkit/core         7.2.2
@angular-devkit/schematics   7.2.2
@schematics/angular          7.2.2
@schematics/update           0.12.2
rxjs                         6.3.3
typescript                   3.2.2

C:\>
```

Figure 1.1

In here, we can then check the version of Angular CLI installed. We can also see the node version here with the versions of different Angular packages like core, rxjs, typescript, to name a few.

We can also update the CLI using the following steps:

```
npm uninstall -g @angular/cli
npm cache clean
npm install -g @angular/cli
```

CLI comes to great use when generating components, services, directives, modules etc. There are so many commands that can be used with the help of CLI to generate various Angular features:

Add Component

The command used to *add/generate* a component using CLI is:

`ng generate component <component-name>`

or

`ng g c <component-name>`

Add Module

The command used to add a module to our Angular application is:

`ng generate module <modu.le-name>`

or

`ng g m <module-name>`

Add Directive

The command which is used to add directives to an angular application is as follows:

`ng generate directive <directive-name>`

or

`ng g d <directive-name>`

Add Service

The command to add service using CLI is:

`ng generate service <service-name>`

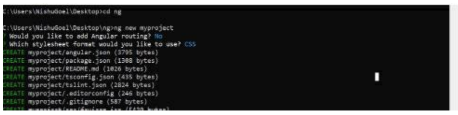
or

ng g s *<service-name>*

There are many commands Angular CLI offers, which help you to speed up Angular application development. You can learn about them in details by following the given link: https://cli.angular.io/

Create First Angular App

Now that we have understood the importance of CLI in working with Angular applications, let us create our first angular application using angular CLI.



```
C:\Users\NishuGoel\Desktop>cd ng

C:\Users\NishuGoel\Desktop\ng>ng new myproject
? Would you like to add Angular routing? No
? Which stylesheet format would you like to use? CSS
CREATE myproject/angular.json (3795 bytes)
CREATE myproject/package.json (1308 bytes)
CREATE myproject/README.md (1026 bytes)
CREATE myproject/tsconfig.json (435 bytes)
CREATE myproject/tslint.json (2824 bytes)
CREATE myproject/.editorconfig (246 bytes)
CREATE myproject/.gitignore (587 bytes)
```

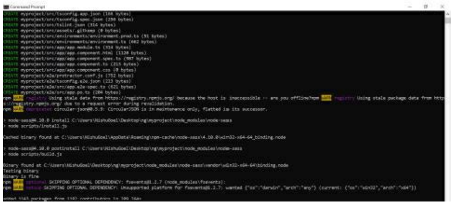
Figure 1.2

To create a new angular app, we use the command

```
ng new <app-name>
```

ng new allows you to generate a new angular project with all the boilerplate files already generated for you. On the latest version, it asks before proceeding like *figure 1.2*.

Once started, the CLI gives us results like in *Figure 1.3:*



```
CREATE myproject/src/tsconfig.app.json (166 bytes)
CREATE myproject/src/tsconfig.spec.json (256 bytes)
CREATE myproject/src/tslint.json (314 bytes)
CREATE myproject/src/assets/.gitkeep (0 bytes)
CREATE myproject/src/environments/environment.prod.ts (51 bytes)
CREATE myproject/src/environments/environment.ts (662 bytes)
CREATE myproject/src/app/app.module.ts (314 bytes)
CREATE myproject/src/app/app.component.html (1120 bytes)
CREATE myproject/src/app/app.component.spec.ts (987 bytes)
CREATE myproject/src/app/app.component.ts (215 bytes)
CREATE myproject/src/app/app.component.css (0 bytes)
CREATE myproject/e2e/protractor.conf.js (752 bytes)
CREATE myproject/e2e/tsconfig.e2e.json (213 bytes)
CREATE myproject/e2e/src/app.e2e-spec.ts (621 bytes)
CREATE myproject/e2e/src/app.po.ts (204 bytes)
npm WARN registry Using stale data from https://registry.npmjs.org/ because the host is inaccessible -- are you offline?npm WARN registry Using stale package data from http
s://registry.npmjs.org/ due to a request error during revalidation.
npm WARN deprecated circular-json@0.5.9: CircularJSON is in maintenance only, flatted is its successor.

> node-sass@4.10.0 install C:\Users\NishuGoel\Desktop\ng\myproject\node_modules\node-sass
> node scripts/install.js

Cached binary found at C:\Users\NishuGoel\AppData\Roaming\npm-cache\node-sass\4.10.0\win32-x64-64_binding.node

> node-sass@4.10.0 postinstall C:\Users\NishuGoel\Desktop\ng\myproject\node_modules\node-sass
> node scripts/build.js

Binary found at C:\Users\NishuGoel\Desktop\ng\myproject\node_modules\node-sass\vendor\win32-x64-64\binding.node
Testing binary
Binary is fine
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@1.2.7 (node_modules\fsevents):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@1.2.7: wanted {"os":"darwin","arch":"any"} (current: {"os":"win32","arch":"x64"})
```

Figure 1.3

To open this newly created app inside Visual Studio code, we use **code followed by a dot.**

Inside VS Code, we can see the structure of the newly created project like in *figure 1.4*.

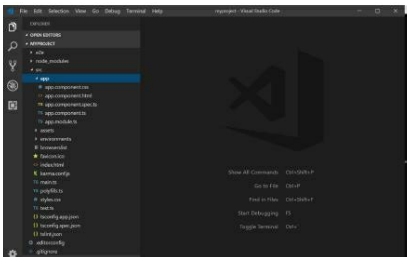


Figure 1.4

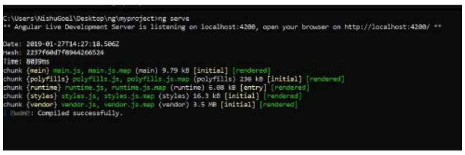
Now to compile and run our application, we use the command:

```
ng serve
```

We can also change the port no., in case one is already in use. To do that, we use the command:

```
ng serve -- port 4300
```

This gives us results like in *figure 1.5*.

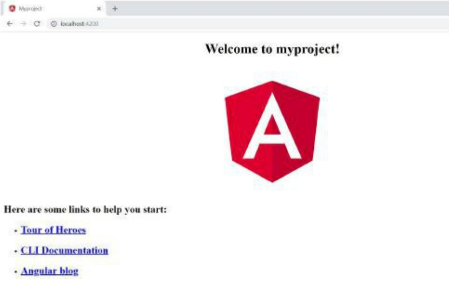


```
C:\Users\NishuGoel\Desktop\ng\myproject>ng serve
** Angular Live Development Server is listening on localhost:4200, open your browser on http://localhost:4200/ **

Date: 2019-01-27T14:27:18.506Z
Hash: 2237f60d7f8944266524
Time: 8039ms
chunk {main} main.js, main.js.map (main) 9.79 kB [initial] [rendered]
chunk {polyfills} polyfills.js, polyfills.js.map (polyfills) 236 kB [initial] [rendered]
chunk {runtime} runtime.js, runtime.js.map (runtime) 6.08 kB [entry] [rendered]
chunk {styles} styles.js, styles.js.map (styles) 16.3 kB [initial] [rendered]
chunk {vendor} vendor.js, vendor.js.map (vendor) 3.5 MB [initial] [rendered]
i ｢wdm｣: Compiled successfully.
```

Figure 1.5

The URL is mentioned in the results. In my case, it is **localhost:4200.** And now can go to the browser with the URL and check the CLI-created Angular app in *figure 1.6*.



Welcome to myproject!

Here are some links to help you start:

- Tour of Heroes
- CLI Documentation
- Angular blog

Figure 1.6

Summary

In this chapter, we learnt about theoretical concepts of Angular. A solid understanding of theory of Angular helps you to write Angular application faster and in better way. In this chapter, you learnt about the following topics:

- What is Angular
- How is Angular different from AngularJS
- Angular's basic architecture
- Angular CLI
- Creating your First App