application are defined. You can have a look inside, it should now contain the following dependencies:

- the different `@angular` packages.
- `rxjs`, a really cool library for reactive programming. We have a dedicated chapter on this topic and about RxJS in particular.
- `zone.js`, doing the heavy lifting for detecting the changes (we'll dive into this later also).
- some dependencies for developing the application, like the CLI, TypeScript, some test librairies, some typings…

TypeScript itself has a configuration file `tsconfig.json` (and another one in `src` called `tsconfig.app.json`), which stores the compilation options. As we saw in the previous chapters, we are using TypeScript with decorators (hence the two options about decorators), and we want our code to transpile to ECMAScript 5, allowing it to run in every browser. The `sourceMap` option allows generating source maps, i.e. files that contain a mapping between the generated ES5 code and the original TypeScript code. Those source maps are used by the browser to let you debug the ES5 code it executes by stepping through the original TypeScript code that you have written.

TypeScript projects often also use TSLint, a linter used to check your code against the best practices. TSLint has its own options, stored in `tslint.json`, where you add/remove some of its rules.

Angular CLI itself has a configuration file `angular.json` if you want to override some of its defaults.

The last one, `karma.conf.js` is used to configure the testing tool (more about that when we'll talk about testing).

> **NOTE**
>
> The ebook is using Angular version `7.2.3` for the examples. Angular CLI will probably install the most recent version, which might not be exactly the same. If you want to use the same version as we are, replace the version in the `package.json` by `7.2.3` for each Angular package. That might save you a few headaches! Or, even better, follow our free online exercise Getting Started 🐴! which is always up-to-date and battle-tested!

Now that we have been over the configuration, let's see the application code.

8.4. Our first component

As we saw in the previous section, a component is a combination of a view (the template) and some logic (our TS class). The CLI has already created one for us: `src/app/app.component.ts`. Let's check it out: