popularly acclaimed `ember-cli`.

The tool is under continuous development, with a dedicated Google team working on it and making it better and better. It is now the recommended and *de facto* standard way of creating and building Angular apps. So let's give it a try, and discover the ton of cool stuff packed in it!

PRACTICE	If you want, you can follow our online exercise Getting Started 🐂! It's free and part of our Pro Pack, where you'll learn how to build a complete application step by step. The first exercise is about getting everything up and running with Angular CLI, and goes further than what we see in the chapter.

First let's install Angular CLI, and generate a new application with the `ng new` command. If you want to use exactly the same CLI version than we are (`7.3.0`), you can use `npm install -g @angular/cli@7.3.0` instead.

```
npm install -g @angular/cli
ng new ponyracer --prefix ns --no-interactive
```

This will create a project skeleton in a new directory called `ponyracer`. From this directory, you can start your app with:

```
ng serve
```

This will start a small HTTP server locally, with a hot reload configuration. It means that every time you modify and save a file, the server will rebuild the app, and the browser will reload it immediately.

Tada! You have your first application up and running! 🎉

8.3. Application structure

Let's dive for a few seconds into the generated code.

Open the project in your preferred IDE. You can use pretty much anything you want, but you should activate the TypeScript support for maximum comfort. Pick your favorite: Webstorm, Visual Studio Code, Atom… All of them have great support for TypeScript.

NOTE	If your IDE supports it, code completion should work as the Angular dependencies have their own `d.ts` files in the `node_modules` directory, and TypeScript is able to detect them. You can even navigate to the type definitions if you want to. TypeScript will bring its type-checking to the table, so you'll see what mistakes you make as you type. As we are using source maps, you can see the TS code directly from your browser, and even debug your app by setting breakpoints in the TypeScript code.

You should see a bunch of configuration files in the root directory: welcome to Modern JavaScript!

The first one you may recognize is the `package.json` file: that's where the dependencies of the