Angular – 18.X.X

Started updating as on July 18th, 2024

Introduction

Angular is an application design framework and development platform for creating efficient and sophisticated single-page apps.

Angular is use to build CLIENT SIDE APPLICACTION.

Core feature for a SPA is Partial Rendering.

Angular promotes Modular approach.

Rapid and easy development of Responsive Web Design.

Re-usable and Unit Testable application development.

What is Angular?

Angular now is a TypeScript-based open-source web application framework led by the Angular Team at Google and by a community of individuals and corporations.

AngularJS was their first JavaScript-based open-source front-end web framework mainly maintained by Google.

Current Angular is a complete rewrite from the same team that built AngularJS.

It aims to simplify both development and testing of such applications by providing a framework for client-side model–view–controller (MVC) and model–view–view–model (MVVM) architectures, along with components commonly used in rich Internet applications.

Angular Features at a Glance

DEVELOP ACROSS ALL PLATFORMS:

- One framework - For web, mobile web, native mobile and native desktop.

SPEED & PERFORMANCE:

- Achieve the maximum speed possible on the Web Platform via Web Workers and server-side rendering.
- Control over scalability
- Meet huge data requirements by building data models on RxJS.

INCREDIBLE TOOLING

- Build features quickly with simple, declarative templates.
- IDE and editor's support let you focus on building apps rather than trying to make the code work.

LOVED BY MILLIONS

What is and why it is called framework?

Angular is a development platform, built on TypeScript. With Angular, you're taking advantage of a platform that can scale from single-developer projects to enterprise-level applications.

Angular by Ranjan

As a platform, Angular includes:

- A component-based framework for building scalable web applications
- A collection of well-integrated libraries that cover a wide variety of features, including routing, forms management, client-server communication, and more
- A suite of developer tools to help you develop, build, test, and update your code

A framework is like a platform for developing software applications.

A framework can have pre-defined classes and functions that can be re-used to add several functionalities, which otherwise we would have to write from scratch by our own.

A framework is a collection of pre-defined classes and methods which provides APIs for performing different operations when used in an application

What is a SPA?

Single Page Applications are web applications that load a single HTML page and only a part of the page instead of the entire page gets updated with every click of the mouse. The page does not reload or transfer control to another page during the process. This ensures high performance and loading pages faster.

Angular is a full featured SPA framework. So, when you load the application for the first time, not all the pages from the server will be rendered... It's only index.html that loads when you load the application. Since only a single page is loaded it is called SPA.

Single page application (SPA) is a web application that fits on a single page. All your code (JavaScript, HTML, and CSS) is recovered with a single page stack. Furthermore, route between pages performed without invigorating the entire page.

Advantages of SPA:

- No page flickers. Native application feels.
- Client-side routing and data rendering on the client side.
- Data from server is in JSON format.

Limitation of JavaScript / jQuery

Vanilla JavaScript or jQuery code becomes hard to maintain and we will need a way to properly structure our application.

A lot of applications built using vanilla JavaScript / jQuery is hard to test.

There are some functionalities which we will have to write from scratch when using JavaScript / jQuery

Advantages of using Angular

- Angular gives our application a clean and loosely coupled structure that is easy to understand & maintain.
- It brings a lot of utility code which can be re-used in lot of applications. Especially, when dealing with user navigation & browser history.
- Applications built with Angular are more testable.

Angular Version History

1. Angular JS was not designed with the need of todays applications.
2. Angular 2 was completely re-written from ground up.
3. It fixed all the issues which Angular 1 had.
4. There had not been major changes after the release of Angular 2.

Latest Angular version is Angular 18.2 (Minor Release) released on July 14th, 2024. Angular 18 (Major Release) arrived on May 22nd, 2024.

Prerequisites

Like most modern frameworks, Angular expects you to be familiar with HTML, CSS and JavaScript.

In addition, it's recommended to have familiarity with JavaScript Classes, TypeScript fundamentals and TypeScript Decorators.

Setup local development environment

- Install NODE JS on development machine.
- Install angular CLI globally.
- Create an Angular project.
- Compile & run Angular project.

First Identify the version of node.js that Angular requires

Angular requires an active LTS or maintenance LTS version of Node. Let's confirm your version of node.js. For information about specific version requirements, see the engines property in the package.json file.

From a Terminal window run following command:

node --version

Confirm that the version number displayed meets the requirements. If you do not have a version of node.js installed, Install the correct version of node.js for Angular.

Install the latest version of Angular

Angular CLI is a command line interface which we use to create new angular project or generate some boiler plate code as well as create deployable packages.

With node.js and npm installed, the next step is to install the Angular CLI which provides tooling for effective Angular development.

From a Terminal window run following command:

npm install -g @angular/cli@latest

Build your first Angular app

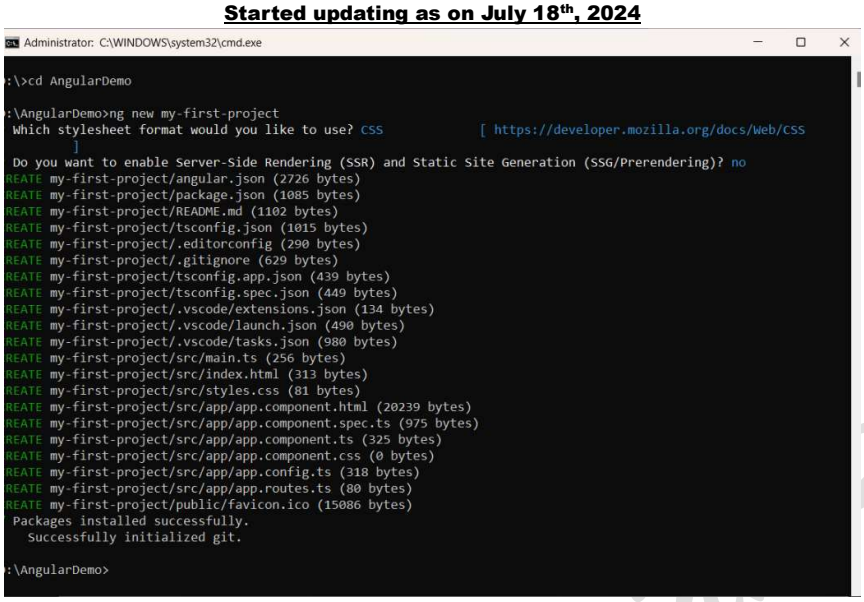
To create a new angular project, move to the folder where you wand to create the project using command prompt / terminal and type following command.

ng new my-first-project

You develop applications in the context of an Angular workspace. A workspace contains the files for one or more projects. A project is the set of files that comprise an application or a shareable library.

When you run this command, the CLI installs the necessary Angular npm packages and other dependencies in a new workspace, with a root-level application named my-first-project.

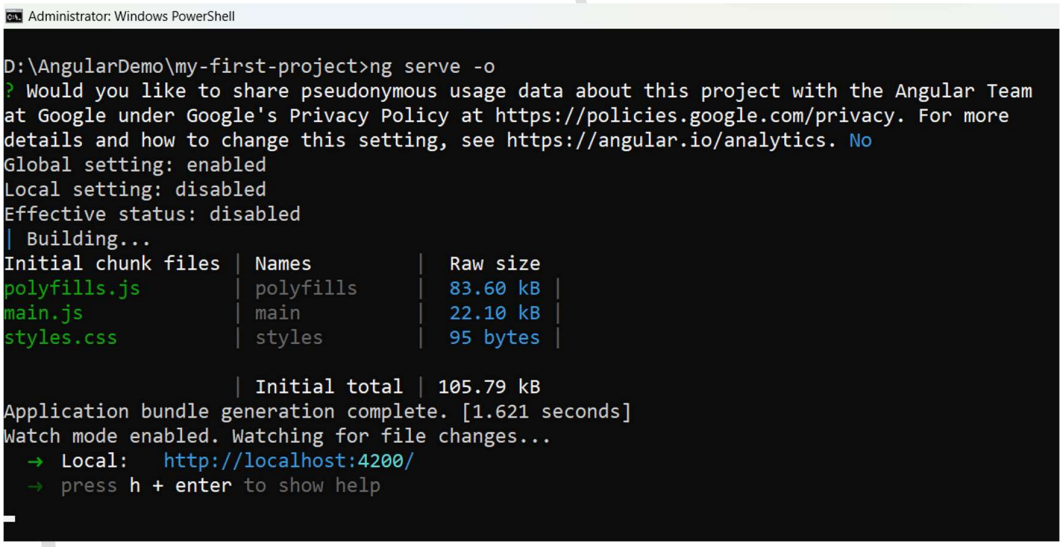
By default, ng new creates an initial skeleton application at the root level of the workspace, along with its end-to-end tests. The skeleton is for a simple welcome application that is ready to run and easy to modify.



cd my-first-project

To run an Angular project, move to the project folder using command prompt / terminal and type following command.

ng serve



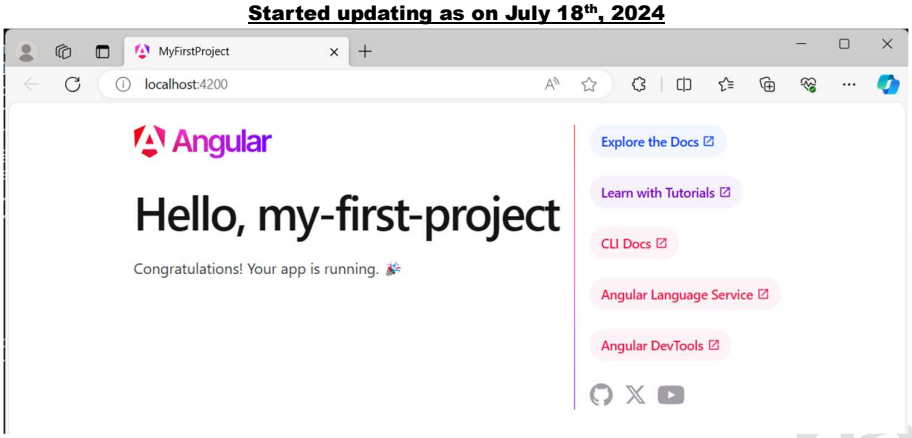
In your browser, open http://localhost:4200/ to see the new application run.

Or use command

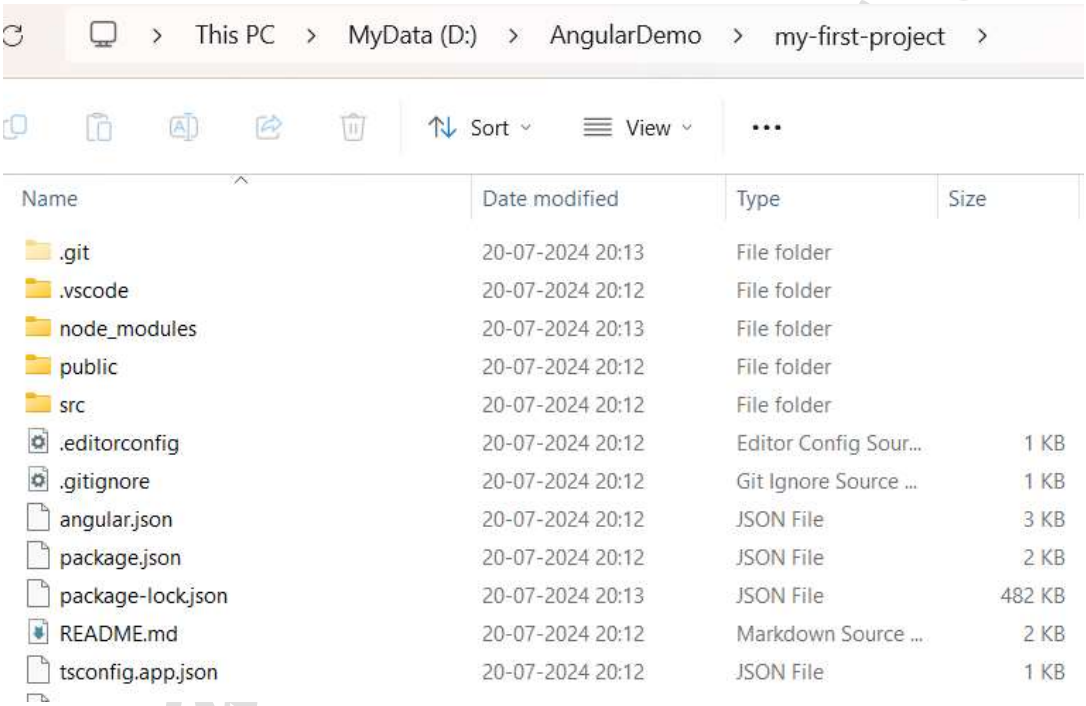
ng serve - o

will open http://localhost:4200/ automatically in browser.

Set -executionpolicy remotesigned



Project's Folder Structure



Review the files in the project. (Structure may change in various major or minor version)

1. **Workspace configuration files**
   a. **.editorconfig** Configuration for code editors. See EditorConfig.
   b. **.gitignore** Specifies intentionally untracked files that Git should ignore.
   c. **README.md** Documentation for the workspace.
   d. **angular.json** CLI configuration for all projects in the workspace, including configuration options for how to build, serve, and test each project. For details, see Angular Workspace Configuration.
   e. **package.json** Configures npm package dependencies that are available to all projects in the workspace. See npm documentation for the specific format and contents of this file.
   f. **package-lock.json** Provides version information for all packages installed into node_modules by the npm client. See npm documentation for details.

g. **src/** Source files for the root-level application project.
h. **public/** Contains image and other asset files to be served as static files by the dev server and copied as-is when you build your application.
i. **node_modules/** Installed npm packages for the entire workspace. Workspace-wide node_modules dependencies are visible to all projects.
j. **tsconfig.json** The base TypeScript configuration for projects in the workspace. All other configuration files inherit from this base file. For more information, see the relevant TypeScript documentation.

2. **Application project files**
   a. **Application source files**
      i. **app/** Contains the component files in which your application logic and data are defined. See details below.
      ii. **favicon.ico** An icon to use for this application in the bookmark bar.
      iii. **index.html** The main HTML page that is served when someone visits your site. The CLI automatically adds all JavaScript and CSS files when building your app, so you typically don't need to add any <script> or<link> tags here manually.
      iv. **main.ts** The main entry point for your application.
      v. **styles.css** Global CSS styles applied to the entire application.
   b. **app/ component files**
      i. **app.config.ts** Defines the application configuration that tells Angular how to assemble the application. As you add more providers to the app, they should be declared here.

         Only generated when using the --standalone option.

      ii. **app.component.ts** Defines the application's root component, named AppComponent. The view associated with this root component becomes the root of the view hierarchy as you add components and services to your application.
      iii. **app.component.html** Defines the HTML template associated with AppComponent.
      iv. **app.component.css** Defines the CSS stylesheet for AppComponent.
      v. **app.component.spec.ts** Defines a unit test for AppComponent.
      vi. **app.module.ts** Defines the root module, named AppModule, that tells Angular how to assemble the application. Initially declares only the AppComponent. As you add more components to the app, they must be declared here.

         Only generated when using the --standalone false option.

      vii. **app.routes.ts** Defines the application's routing configuration.
   c. **Application configuration files**
      i. **tsconfig.app.json** Application-specific TypeScript configuration, including Angular compiler options.
      ii. **tsconfig.spec.json** TypeScript configuration for application tests.

Angular by Ranjan B.

NOTE: Above example of **my-first-project** is created with default angular command and with this new version it has created bit complex first application. As a beginner let's try with some simple application having less complex structure.

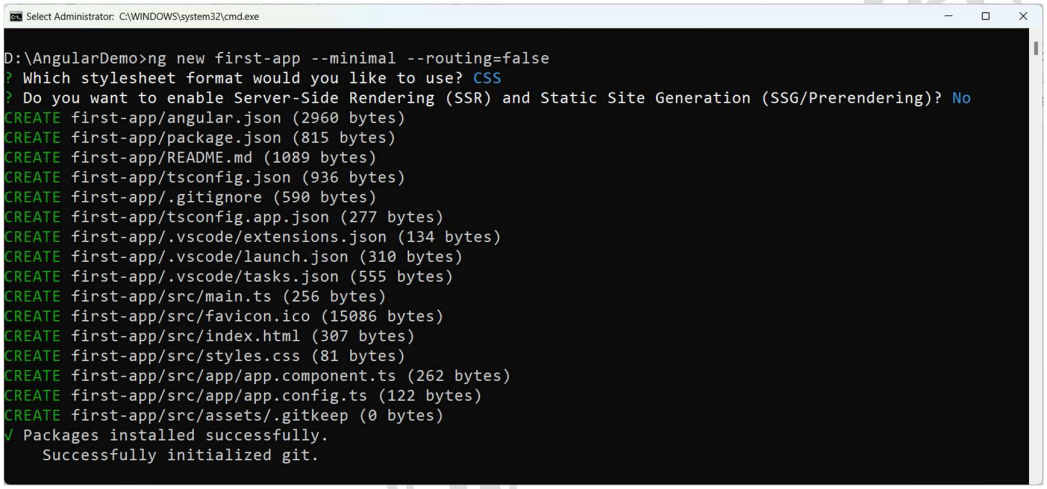
Create another simple application using following command:

ng new first-app --minimal

And if you don't want routing enabled then

ng new first-app --minimal --routing=false

The command above will generate bare minimum set of files in project and good for learning purpose.



Multiple projects

A multi-project workspace is suitable for an organization that uses a single repository and global configuration for multiple Angular projects (the "monorepo" model). A multi-project workspace also supports library development.

The following command creates a workspace with all of the workspace-wide configuration files, but no root-level application.

ng new my-workspace --no-create-application

You can then generate applications and libraries with names that are unique within the workspace.

cd my-workspace

ng generate application my-app

ng generate library my-lib

Following is a typical multiple project folder structure.

```
my-workspace/
  ├── …                  (workspace-wide configuration files)
  └── projects/          (applications and libraries)
      ├── my-app/        (an explicitly generated application)
      │   └── …          (application-specific code and configuration)
      └── my-lib/        (a generated library)
          └── …          (library-specific code and configuration)
```

Install integrated development environment (IDE)

Now for day-to-day work and building projects using command line is really difficult. So, here onwards we will be using IDE for working with angular projects.

You can use

1. Visual Studio Code [Free IDE with great Angular support] Download from
    a. https://code.visualstudio.com/
2. Another popular IDEs
    a. Web Storm - https://www.jetbrains.com/webstorm/
    b. Atom - https://atom.io/
    c. Sublime - http://www.sublimetext.com/

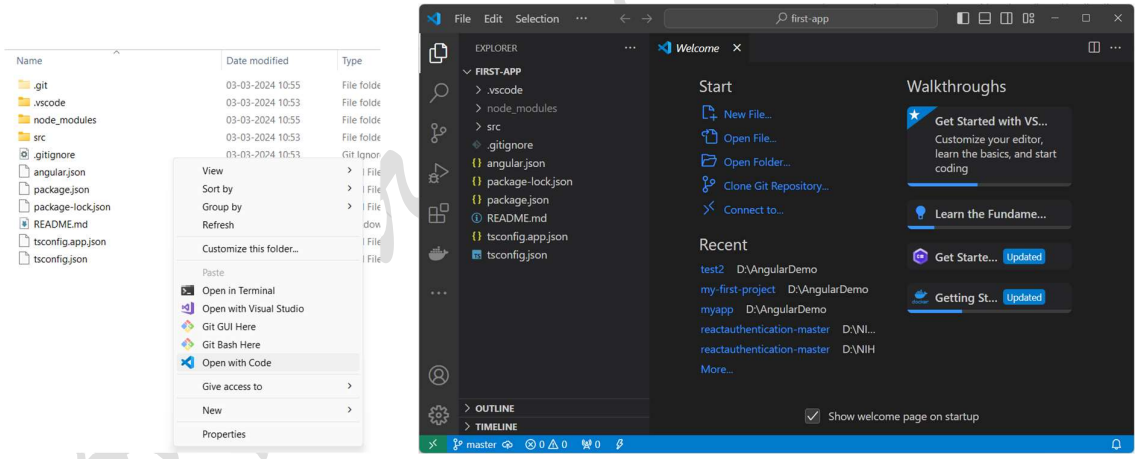
You are free to use any tool you prefer to build apps with Angular. Angular recommend the following:

1. Visual Studio Code
2. As an optional, but recommended step you can further improve your developer experience by installing the Angular Language Service.

Using Visual Studio Code

We will be using Visual Studio Code. Download and Install Visual Studio Code from its website.

Once installed you can open your project Folder using Visual Studio Code. You can also use context menu option in Windows Explorer as shown below:

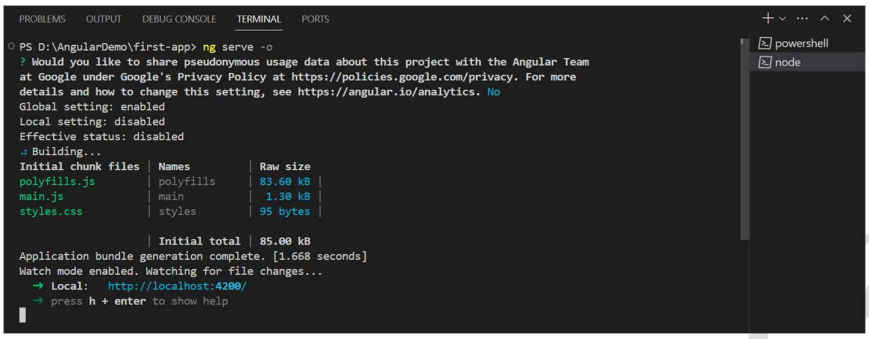


It will open project folder in Visual Studio Code. Now you can edit any of the source code in Visual Studio Code. Explore the project by opening files.

Make a note that you still need ng serve from the CLI to run the application during development. Visual Studio Code is equipped with Windows PowerShell.

You can use Windows PowerShell terminal for your CLI working. Press Ctrl + ` to activate/deactivate Windows PowerShell terminal.

You can fire any of the CLI command from this terminal. For an angular application execution, we need 'ng serve -o' command. And also, we need to keep it running for previewing changes. To do so we will be opening another terminal. Use + on the right-hand side and create one more terminal window and use ng serve in that terminal.

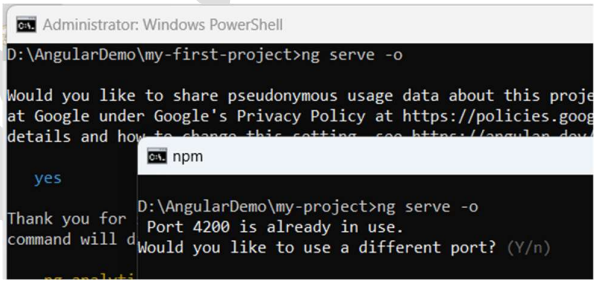


And http://localhost:4200/ is running in browser:

Now in IDE you have two terminal windows, we can switch to either one using right panel.

Run an Angular application on different port

By default when you use ng serve command it run application on http://localhost:4200/. If you want to execute one more application angular automatically provide option to run on another port.



If you want to run your angular app on a different port then there are two ways to change port in the angular app.

1. You can define direct port with ng server command, let's run below command:

   ng serve --port 3000

2. Another way is, you can configure default post in angular.json file as:

```
"projects": {
  "my-project-name": {
    ... extra configurations
    "architect": {
      "serve": {
        "options": {
          "port": 3000
        }
      }
    }
  }
}
```

☸✠End of Chapter✠☸