



Angular 19 Online Training



01

By: Chandan Kumar





Angular Environment Setup

In this class we will learn how to configure an Angular 2 + application development environment.

To Set up Development Environment for Angular 2+, we require the following-

- IDE for writing your code (Editor)
- Nodejs
- Npm
- Angular CLI
- :



IDE for writing your code (Editor)

There are many editors that can be used for Angular 8 development such as Visual Studio code and WebStorm. In this course, we will use the Visual Studio code, which is free from Microsoft.

Installation of Visual Studio Code

- Some features of Visual Studio Code are following–
- Light editor as compared to the actual version of Visual Studio.
- It can be used for coding languages such as Clojure, Java, Objective-C, and many other languages.
- It supports built-in Git extension so that you can work with source control without leaving the editor.
- It includes built-in support for IntelliSense code completion, rich semantic code understanding and navigation, and code refactoring.
- It includes an interactive debugger, so you can step through source code, inspect variables, view call stacks, etc.
- Many more extensions for development.

Note: The link of official site for Visual Studio code is <https://code.visualstudio.com/>



IDE for writing your code (Editor)

<https://code.visualstudio.com/>

The screenshot shows the Visual Studio Code website with the headline "Code editing. Redefined." and a "Download for Windows" button. Overlaid on the website is a screenshot of the Visual Studio Code editor interface. The editor shows a file named "app.ts" with the following code:

```
1 import app from './app';
2 import debugModule = require('debug');
3 import http = require('http');
4
5 const debug = debugModule('node-express-typescript:server');
6
7 // Get port from environment and store in Express.
8 const port = normalizePort(process.env.PORT || '3000');
9 app.set('port', port);
10
11 // create
12 const server = export
13 server.listen
14 server.on
15 server.on
16
17 /**
18  * Normal
19  */
20 function normalizePort(val: any): number|string|boolean {
21   let port = parseInt(val, 10);
22 }
```

The editor interface also shows a sidebar with extensions like C#, Python, Debugger for Chrome, C/C++, Go, and ESLint.

01



Nodejs

Download Nodejs from <https://nodejs.org/en/download/> , and click on the windows installer. you can show here both the LTS and the current version of the node from where the you can download the recommended version or the current version.

The screenshot shows the Node.js website's 'Downloads' section. At the top is the Node.js logo and a navigation bar with links: HOME, ABOUT, DOWNLOADS, DOCS, GET INVOLVED, SECURITY, NEWS, and FOUNDATION. The 'Downloads' section title is followed by the text 'Latest LTS Version: 10.16.0 (includes npm 6.9.0)'. Below this is a prompt to 'Download the Node.js source code or a pre-built installer for your platform, and start developing today.' The page is divided into two main columns: 'LTS Recommended For Most Users' and 'Current Latest Features'. Under the 'LTS' column, there are three options: 'Windows Installer' (with a Windows logo and file name 'node-v10.16.0-x64.msi'), 'macOS Installer' (with an Apple logo and file name 'node-v10.16.0.pkg'), and 'Source Code' (with a cube icon and file name 'node-v10.16.0.tar.gz').

01



Nodejs

After installing the node, open the Visual Studio integral terminal and type node -v to verify the installed version of the node. This will help you to see the version of nodejs currently installed on your system.

EXAMPLE

```
C:\>node -v
```

```
v19.12.0
```




Angular CLI

Angular CLI is very important in the setting of Angular, visit the homepage <https://cli.angular.io/> of angular to get the reference of the command.



01



Angular CLI

To install angular cli globally on your system type `npm install -g @angular/cli`. It installs Angular CLI globally where g is referred to globally.

EXAMPLE

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

.

If you want to make sure you have correctly installed the angular CLI, open the Visual Studio integrated terminal and type `ng -v`. If you can see the cli version as shown below, then installation is complete.



Angular CLI

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Angular CLI
Angular CLI: 6.2.3
Node: 10.13.0
OS: win32 x64
Angular: 6.1.8
... animations, common, compiler, compiler-cli, core, forms
... http, language-service, platform-browser
... platform-browser-dynamic, router

Package                                  Version
-----
@angular-devkit/architect               0.7.5
@angular-devkit/build-angular           0.7.5
@angular-devkit/build-optimizer         0.7.5
@angular-devkit/build-webpack           0.7.5
@angular-devkit/core                    0.7.5
@angular-devkit/schematics              0.8.3
@angular/cli                            6.2.3
@ngtools/webpack                        6.1.5
@schematics/angular                     0.8.3
@schematics/update                      0.8.3
rxjs                                    6.3.2
typescript                              2.7.2
webpack                                4.9.2
```



Summary:

- Verify if npm is installed or not: `npm -v`
- verify the installed version of the node: `npm -v`
- Verify if node is installed or not: `node -v`
- verify the installed version of the node: `node -v`
- To install angular cli globally on your system type: `npm install -g @angular/cli`
- To check correctly installed the angular CLI: `ng v`

01



What is Node.js?

Node.js is a server-side platform built on Google Chrome's JavaScript Engine (V8 Engine). Node.js was developed by Ryan Dahl in 2009 and its latest version is v0.10.36.

Node.js is an open source, cross-platform runtime environment for developing server-side and networking applications. Node.js applications are written in JavaScript, and can be run within the Node.js runtime on OS X, Microsoft Windows, and Linux.

Node.js also provides a rich library of various JavaScript modules which simplifies the development of web applications using Node.js to a great extent.

- Node.js is an open source server environment
- Node.js is free
- Node.js runs on various platforms (Windows, Linux, Unix, Mac OS X, etc.)
- Node.js uses JavaScript on the server



What is Node.js?

Node.js = Runtime Environment + JavaScript Library

Where to Use Node.js?

Following are the areas where Node.js is proving itself as a perfect technology partner.

- I/O bound Applications
- Data Streaming Applications
- Data Intensive Real-time Applications (DIRT)
- JSON APIs based Applications
- Single Page Applications

01



Why does Angular 2 and above need Node.js?

Angular does not need node.js directly. Node js is used for all the build and development tools.

It is not mandatory to use node.js for developing angular application. You can very well go ahead without node.js for developing angular application but it would not be wise to do so.

You do not need to use Node anywhere in production server to use front-end JavaScript frameworks like Angular or react etc.

Use Node and NPM not as production server but as tooling and building Angular apps.

Let me explain you some of the reasons how node.js makes angular app development process easier for us:

01



Why does Angular 2 and above need Node.js?

- Node allows you to spin up a **lightweight web server** to host your application locally in your system.
- **NPM** (Node Package Manager) comes with node.js by default. NPM allows you to manage your dependencies. So, you don't have to worry for operations like adding a dependency, removing some, updating your package.json.
- Third and the most important, npm gives you **angular cli** or **ng cli**(angular command line interface) . Angular CLI is a great tool for scaffolding your application. So, you don't need to write boilerplates manually.
- Angular recommends the use of TypeScript. Now, your browser does not understand TypeScript. It needs to be transpiled to JavaScript. Also, you need to bundle your js files and stylesheets together with the html doc so as to get the web app CLI which is ready to be hosted. Angular CLI helps you to do all these behind the scene. By default, ng cli uses **webpack** for bundling your application and is very helpful for beginners who have just jumped into web development with angular as it abstracts such complexities.



The World's Largest Software Registry (Library)

- **npm** is the world's largest **Software Registry**.
- The registry contains over 800,000 **code packages**.
- **Open-source** developers use **npm** to **share** software.
- Many organizations also use npm to manage private development.



What is Angular CLI?

Angular CLI stands for Angular Command Line Interface. As the name implies, it is a command line tool for creating angular apps.

The Angular CLI is a command-line interface tool that you use to initialize, develop, scaffold, and maintain Angular applications. You can use the tool directly in a command shell, or indirectly through an interactive UI such as [Angular Console](#).

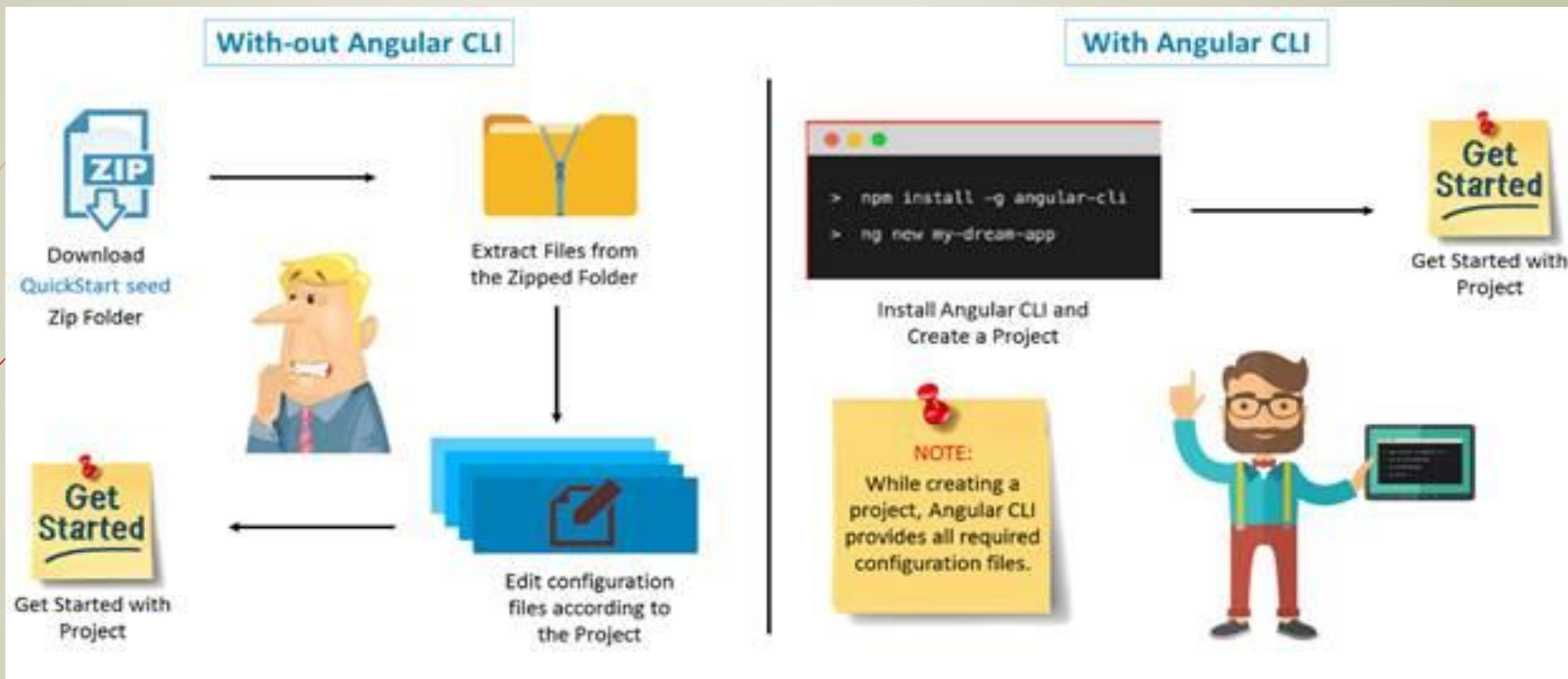
It is recommended to use angular cli for creating angular apps as you don't need to spend time installing

and configuring all the required dependencies and wiring everything together.

The official site for Angular CLI is <https://cli.angular.io/>



What is Angular CLI?



01



What is Angular CLI?

Now, for installing angular cli, follows the steps given below:

Install node.js first if not already install (which I think you probably would have downloaded)

Open the node.js command prompt and issue the command:

Install the CLI using the npm package manager:

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

Note: The -g flag in the above command signifies the fact that the ng-cli is being installed in a global scope.

01

If you want to check out the latest version of angular cli, modify the above stated command as:

Install the latest CLI using the nm package manager:

```
>> npm install @angular/cli@8.0.0
```



Creating a New Project

Now let's create our first project in Angular 8. To create a project in Angular 8, we will use the following command –

To Set up Development Environment for Angular 8, we require the following –

ng new projectname

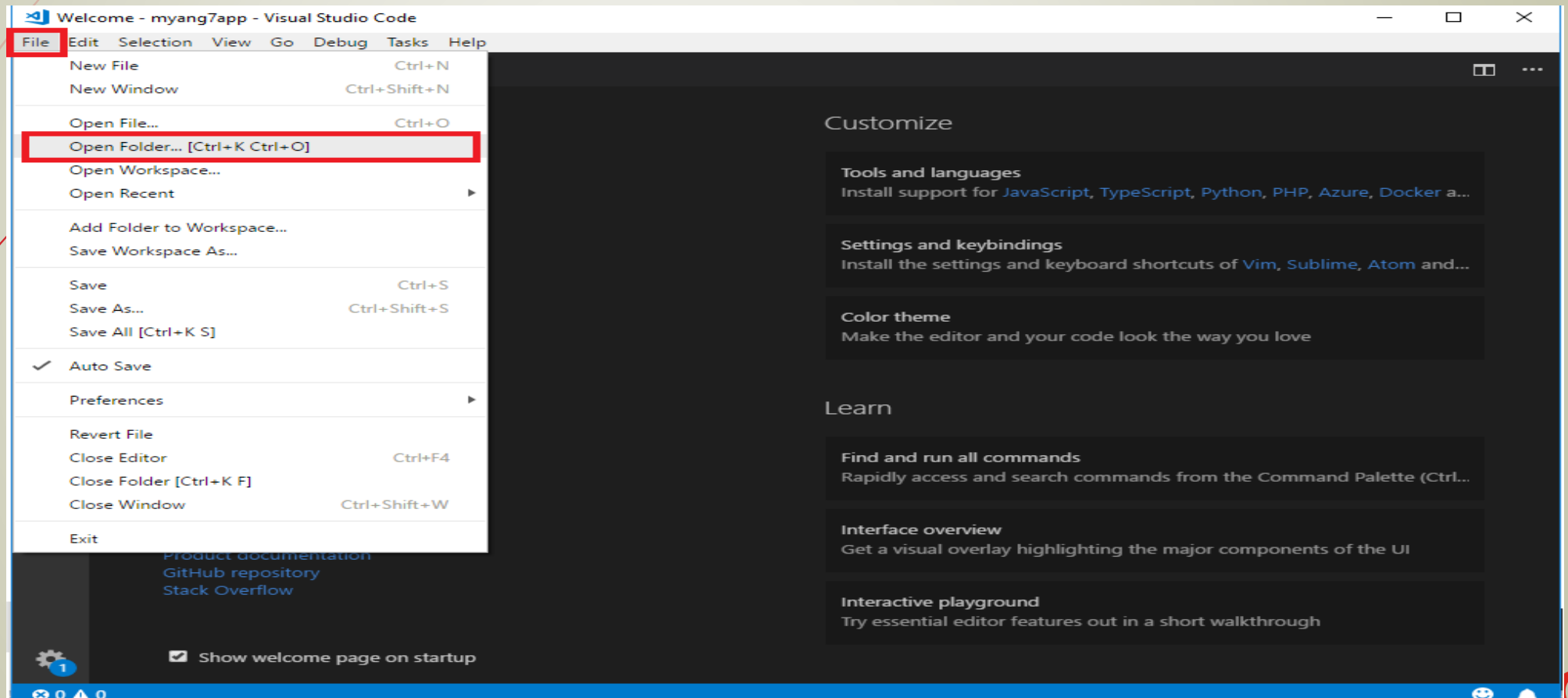
We will name the project as ng new angular12app.



Creating a New Project

Setp 1:

First, create a folder name as Angular, on a desktop or wherever you want. Open Visual code, click on File, go to "Open Folder (ctrl+O)" option and then click on it.

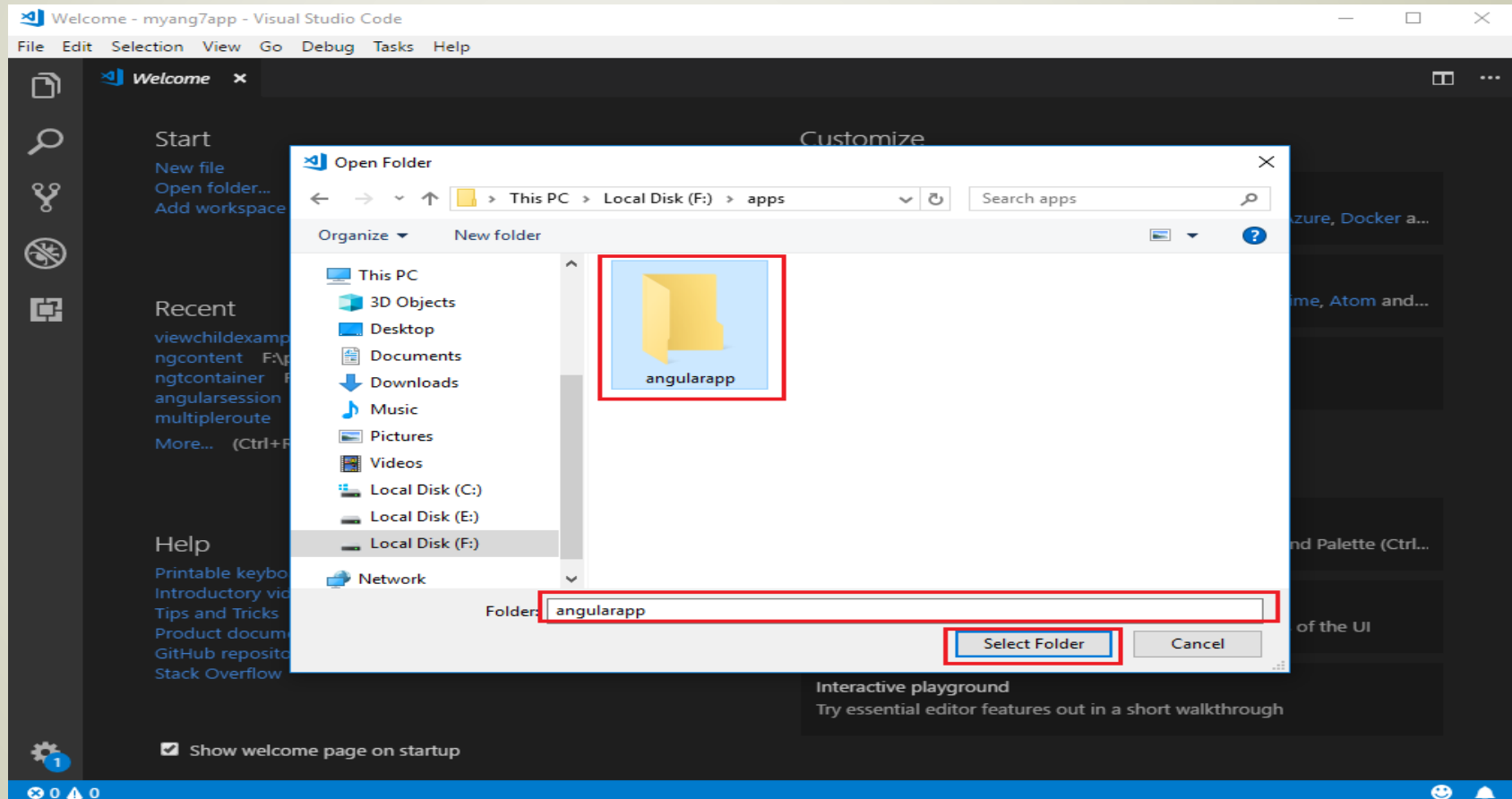




Creating a New Project

Setp 2:

After that, a window will appear. Select the created folder and click on Select Folder box as shown in the below image

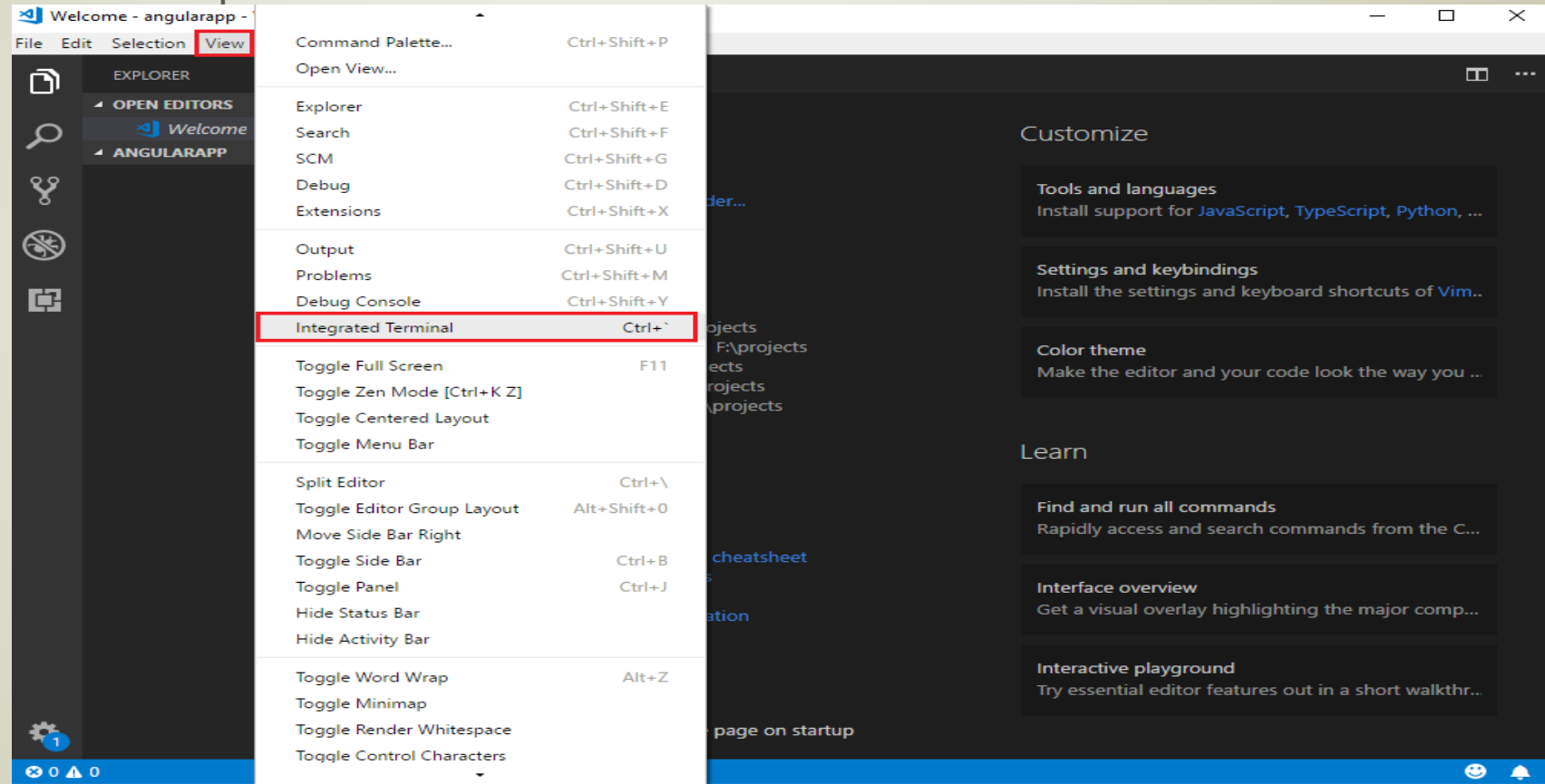




Creating a New Project

Setp 3:

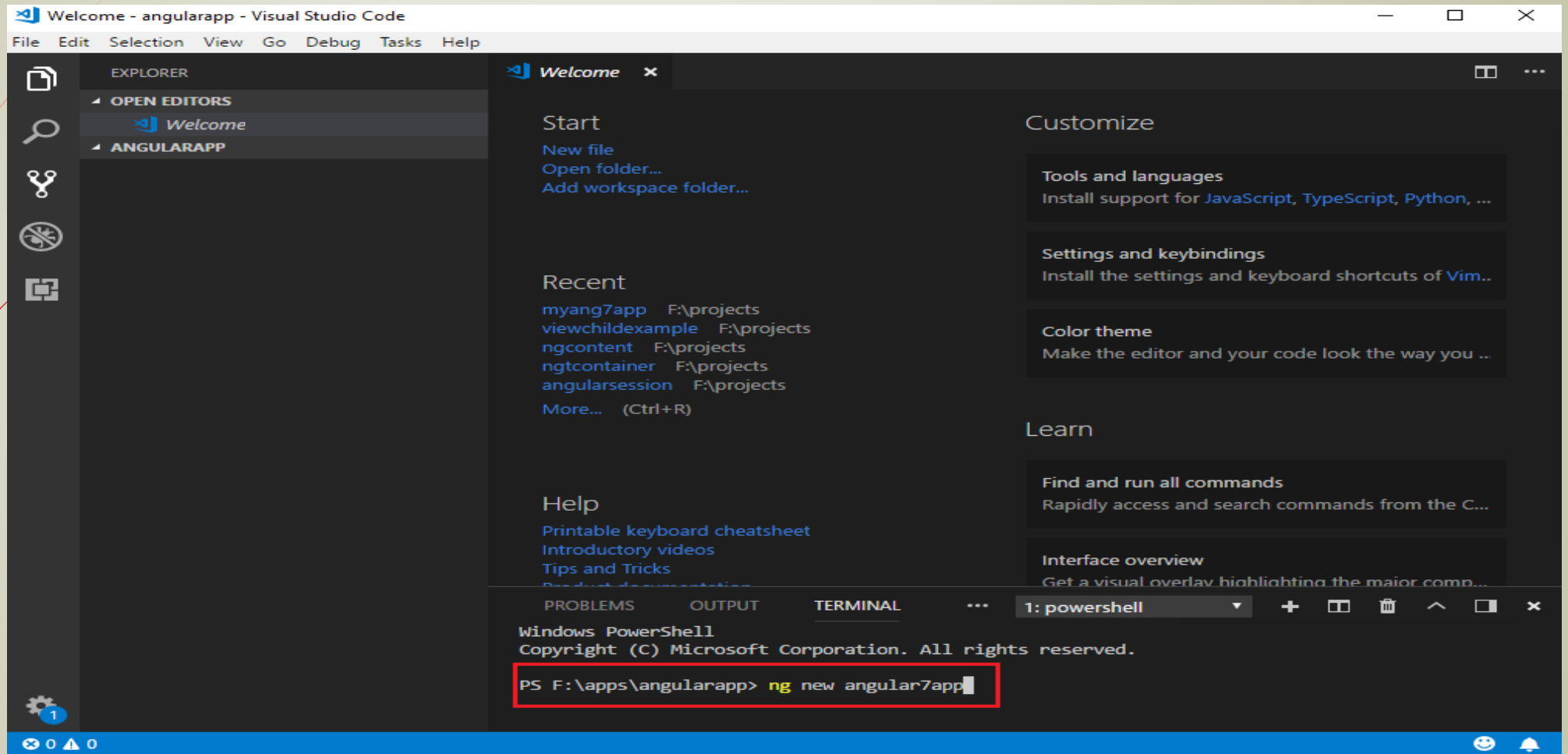
To create the application, click on the view, in the list select Integrated terminal and click on it. Visual Code Console will open.





Creating a New Project

Now run the `ng new angular7app` command in the command line.



01



Building the Application

The build is the process of compiling all source code for creating executables which are ready to be deployed. Building an application also optimizes it, which matters a lot for the production environment. `ng build` command is used to build the application in angular 8.

On the local system, it may not be required to build an angular application i.e. we may simply run through `ng serve`. But to deploy the application we should build it, as it reduces the size of many files. We may choose the profile to be selected for a build.

The profile is normally one of the following

1. development (dev)
2. production (prod)



Building the Application

Angular apps run in development mode by default. When we switch to production mode it runs faster by disabling development time checks such as the dual change detection cycles.

When production builds are enabled via `--prod` command line flag, the runtime production mode is enabled as well.

As a result of the build process, a `dist` folder is created in the root of the application. the contents of this `dist` folder can be uploaded to any http server.

```
ng build <project>
```

```
ng b <project>
```

```
ng build <project> --prod
```

```
ng b <project>--prod
```

By using `ng build` we saw that **five files** are generated for us.

You'll see the five files are:

- 1 main.js
- 2 polyfills.js
- 3 styles.js
- 4 runtime.js
- 5 vendor.js

01



Running the Application

According to the above, npm start will run which you have defined for the start command of the scripts object so in the above case execute an automatic ng-serve command.

It's the reason that, when we execute npm start command, the ng-serve command will execute.

Make sure you're in the project's root directory and then run:

```
ng serve (s)
```

or

```
ng serve (s) --open
```

01



Running/build the Application

By using ng serve we saw that five files are generated for us.

You'll see the five files are:

1. **main.js**: The bulk of our Angular application
2. **polyfills.js**: The things needed to let Angular work in older browsers
3. **runtime.js**: Contains all files (Component, module, services, pipes, etc.) convert into js and transform into minimized version.
4. **styles.js/ styles.css**: The styles!
5. **vendor.js/3rdpartylicenses**: The Angular specific libraries



Project Structure

We know that lots of files and folders are generated whenever we create a new project in Angular.

When we open the Angular 19 project in the editor, we find three main folders e2e, node_modules, src, and different configuration files.



node_modules

When you run npm install, all 3rd party libraries are installed into this folder on which the application may depend. Node.js creates this folder and puts all third-party modules listed in package.json. These libraries are bundled to our application and are purely for development.

What is important to know is that you shouldn't include this folder during deploying your application to production or committing to the git repository. While moving your project to a new location you should skip this folder and run npm install in a new location.



This folder contains the authentic source code for developers. It contains

```
src
├── app
├── assets
├── environments
├── browserslist
├── favicon.ico
├── index.html
├── karma.conf.js
├── main.ts
├── polyfills.ts
├── styles.css
├── test.ts
├── tsconfig.app.json
├── tsconfig.spec.json
└── tslint.json
```

01



app

It contains all the modules and components of your application where every application has at least one module and one component.

Asset

In this folder, you can put images and whatever else which needed to be copied extensively while building your application. In other words, this is the place where you can store static assets of your application for example images, icons etc.

Environments

It contains 2 files, each for different environments. You can use this file to store environment specific configuration like database credentials or server addresses. These files are-
environment.prod.ts file for the production
environment. Environment.ts file for the development environment.

01



src

browserslist

This file is currently used by autoprefixer to adjust CSS to support the specified browsers. For additional information regarding the format and rule options go to

favicon.ico:

It is an icon file which displays on the browser.

index.html:

This is a simple HTML file. It contains HTML code with the head and body section. It is the starting point of your application or you can say that this is where our angular app bootstraps. If you open it you will find that there are no references to any stylesheet (CSS) nor JS files this is because all dependencies are injected during the build process.

01

karma.conf.js:

It is used to store the setting of Karma i.e. test cases. It has a configuration for writing unit tests. karma is the test runner and it uses jasmine as a framework. Both tester and framework are developed by the angular team for writing unit tests.



src

main.ts:

This is the starting point for our app. If you ever coded in languages like Java or C you can compare it with the main() method. If you have didn't just remember that our application starts to execute from this point. This is where we are bootstrapping our first and only module i.e.AppModule.

polyfills.ts:

Polyfills files are used by the compiler to compile our Typescript to specific JavaScript methods which can be parsed by different browsers.

It is basically imported script required for running Angular because angular framework uses the features of javascript which are not available in the current version of javascript, supported by the most browser. So, basically, it fills the gap to provide the features of JavaScript that are needed by Angular and the feature supported by the latest browsers. It is mainly used for backward compatibility. Polyfills files can be divided into two parts-

- Browser Polyfills these are applied before loading zone.js and are sorted by the browser.
- Application imports files imported after zone.js file, they should be loaded before your main file.

01



styles.css:

Global stylesheet file by default means it is where we can add global styles for our applications, including our project. **Note that** each component has its own style component which applies styles only within the scope of the given component.

test.ts:

This is a configuration file of Karma which is used for setting the test environment. In this file, the unit test cases for testing the project will be handled.

tsconfig.app.json:

It is used during compilation and contains the configuration about how your application should compile.

tsconfig.spec.json:

tsconfig.spec.json is used for testing which runs in the node.js environment. It also helps in maintaining the details for testing.



editorconfig

It is the config file for the editor which contains the setting of your editor. It has a parameter like style, size of the character, line length..

Gitignore:

01

This file instructs git which files should be ignored when working with a git repository in order to share the ignore rules with any other users that clone the repository.



angular.json:

Since Angular CLI v6-RC2, the angular-cli.json file has been replaced by angular.json

It contains all the configuration of Angular 7 Project. It has the project name, root directory as source folder (src) name which contains all the components, services, directives, pipes, the starting point of our application (index.html file), the starting point of typescript file (main.ts), style files (style.css). It is also used by @angular/clitool which is used to automate the angular workflow by automating different operations related to the development and testing of angular apps.



package-lock.json:

package-lock.json is automatically generated for those operations where npm modifies either the node_modules tree or package.json. In other words, the package-lock.json is generated after an npm install.

It allows future devs and automated systems to download the same dependencies as the project. It also allows you to go back to past versions of the dependency tree without actually committing the node_modules folder.

01

package-lock.json records the same version of each installed package which allows to re-install them. Future installs will be capable of building an identical dependency tree



package.json:

This file is mandatory for every npm project. It contains basic information regarding the project (name, description, license etc), commands which can be used, dependencies - these are packages required by the application to work correctly, and devDependencies - again the list of packages which are required for application however only during the development phase. i.e. we need Angular CLI only during development to build a final package however for production we don't use it anymore.



README.md:

This file contains the description of the project. It contains information which we would like to provide to the users before they start using the app. It contains basic documentation for your project, pre-filled with CLI command information. Always make sure to enhance it with project documentation so that anyone checking out the repository can build your application.



tsconfig.json:

ts stands for typescripts. Since Angular 2 came out, typescripts are used for developing angular applications. This file contains the configurations for typescripts. If there is a tsconfig file in a directory, that means that directory is a root directory of a typescript project, moreover, it is also used to define different typescript compilation related options.