03

Angular Architecture

- Angular CLI
- Folder Structure
- Process of Angular
- Realtime Architecture of project

Angular CLI

0.0 - Command Line Interface
-> In the terminal - you can run commands and get most of the work done easily
-> It provides some commands and schematics which helps us with faster code generation

0.1 To get the terminal in Visual Studio Code
-> The short cut is
"ctrl+backtick (upper top left side)"

1. check for node version by running the command
-> node -v

2. Check for the npm version
-> npm -v \

3. To install Angular CLI
npm i -g @angular/cli

4. To create new project
ng new <project_name>

- it will generate all the required files, folders, test scripts to run a application

5. We will deep dive and explore each and every folder/file to understand the Angular project in detail
- In the next episode
->

6. cd project01

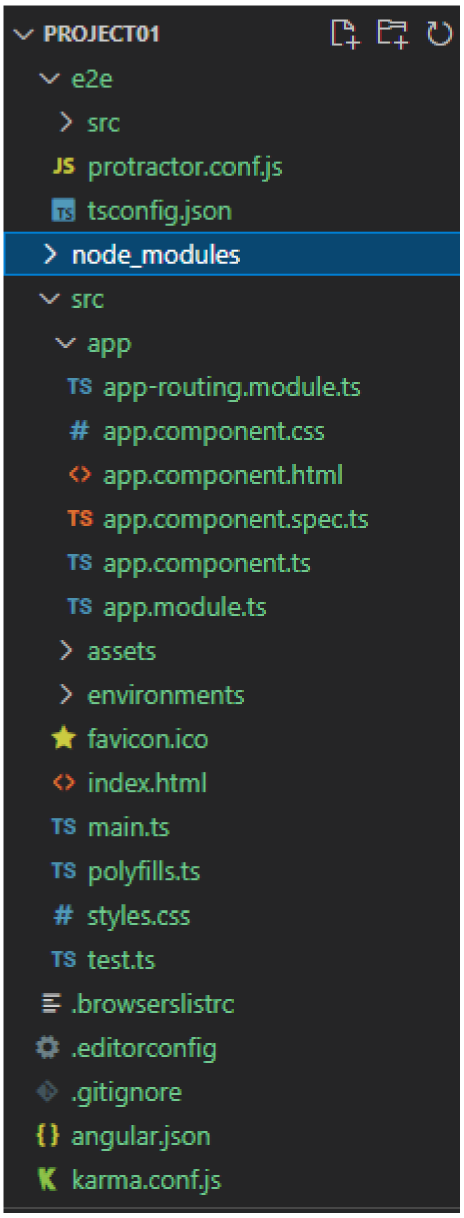
7. ng serve
-> The application will be compiled here
-> default port of Angular is 4200

-> http://localhost:4200

8. Application is up and running

Folder Structure

Below is folder structure



```
PROJECT01
  e2e
    src
    protractor.conf.js
    tsconfig.json
  node_modules
  src
    app
      app-routing.module.ts
      app.component.css
      app.component.html
      app.component.spec.ts
      app.component.ts
      app.module.ts
    assets
    environments
    favicon.ico
    index.html
    main.ts
    polyfills.ts
    styles.css
    test.ts
  .browserslistrc
  .editorconfig
  .gitignore
  angular.json
  karma.conf.js
```

1.project folder <project01>
  ng new <project name>

2.e2e
- -protractor framework is used to end to end testing by default
- -end to end test scripts
- -will end with. e2e-spec.ts extension

```
src
  app.e2e-spec.ts
protractor.conf.js
        ->configuration settings for runnig the end to end test cases
tsconfig.json
        ->basic typescript settings
```

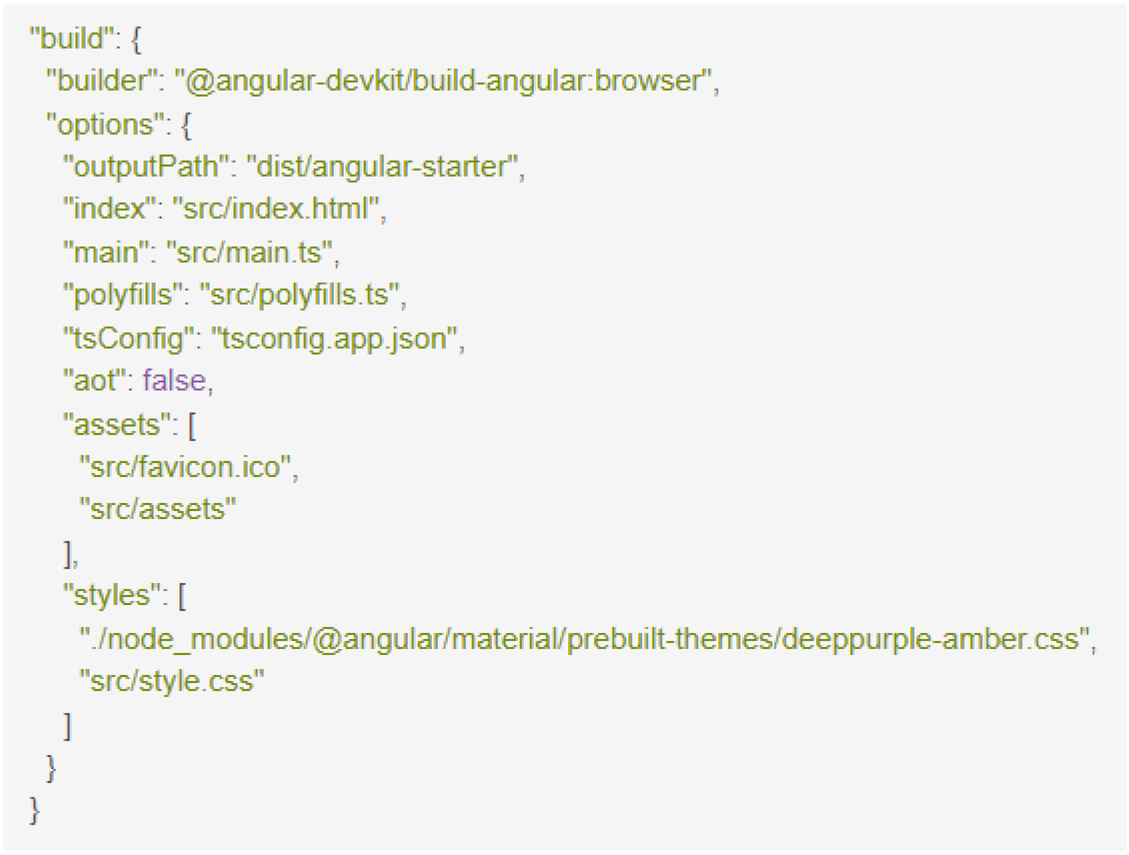
3.src

```
  -entire source code of your application is inside src
  -app module
    app components
            app.component.html ->view/ui/html presentation logic
                 app.component.css  -> stylesheet
                 app.component.spec.ts -> unit test cases script
                 app.component.ts      -> class file where write presentation
        logic
        -assets
                -images
                - mock data
                - pngs
                -apis
        -enviroments
          -local
          -dev
          -test
          -stage
          -prod
```

-favicon.ico
-index.html
Single Page Application
-main.ts
bootstrapModule- it is responsible to load appModule
-Starting point of your application
- any other module can boostrap
- pollyfils.ts
supporting older browsers
-style.css
it is gloable stylesheet for gloable styles
-test.ts
-test script for main.ts file
-testing main.ts file code
-angular.json
this is a backbone of application it has
all
->ports
->settings
->configuration
->scripts
->schemantics
->angular cli
->build
->test
->serve
-karma.conf.js
->spec.ts
->test script runner
->runnig or executing
//next part
-packge.json & package lock json
-tsconfige
typescript setting
-tslint
coding standard

Process of Angular

1. Every Angular app consists of a file named **angular.json**. This file will contain all the configurations of the app. While building the app, the builder looks at this file to find the entry point of the application. Following is an image of the angular.json file:



```
"build": {
  "builder": "@angular-devkit/build-angular:browser",
  "options": {
    "outputPath": "dist/angular-starter",
    "index": "src/index.html",
    "main": "src/main.ts",
    "polyfills": "src/polyfills.ts",
    "tsConfig": "tsconfig.app.json",
    "aot": false,
    "assets": [
      "src/favicon.ico",
      "src/assets"
    ],
    "styles": [
      "./node_modules/@angular/material/prebuilt-themes/deeppurple-amber.css",
      "src/style.css"
    ]
  }
}
```

2. Inside the build section, the main property of the options object defines the entry point of the application which in this case is **main.ts**.
   The main.ts file creates a browser environment for the application to run, and, along with this, it also calls a function called **bootstrapModule**, which bootstraps the application. These two steps are performed in the following order inside the main.ts file:

   ```
   import { platformBrowserDynamic } from '@angular/platform-browser-dynamic';


   platformBrowserDynamic().bootstrapModule(AppModule)
   ```

3. In the above line of code, **AppModule** is getting bootstrapped.
   The AppModule is declared in the app.module.ts file. This module contains declarations of all the components.
   Below is an example of app.module.ts file:

   ```
   import { BrowserModule } from '@angular/platform-browser';
   import { NgModule } from '@angular/core';
   import { AppComponent } from './app.component';

   @NgModule({
     declarations: [
       AppComponent
     ],
     imports: [
       BrowserModule
     ],
     providers: [],
     entryComponents: [],
     bootstrap: [AppComponent]
   })
   export class AppModule { }
   ```

4.As one can see in the above file, **AppComponent** is getting bootstrapped. This component is defined in **app.component.ts** file. This file interacts with the webpage and serves data to it.
Below is an example of app.component.ts file:

```
import { Component } from '@angular/core';

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
  title = 'angular';
}
```

Each component is declared with three properties:
1. **Selector** - used for accessing the component
2. **Template/TemplateURL** - contains HTML of the component
3. **StylesURL** - contains component-specific stylesheets

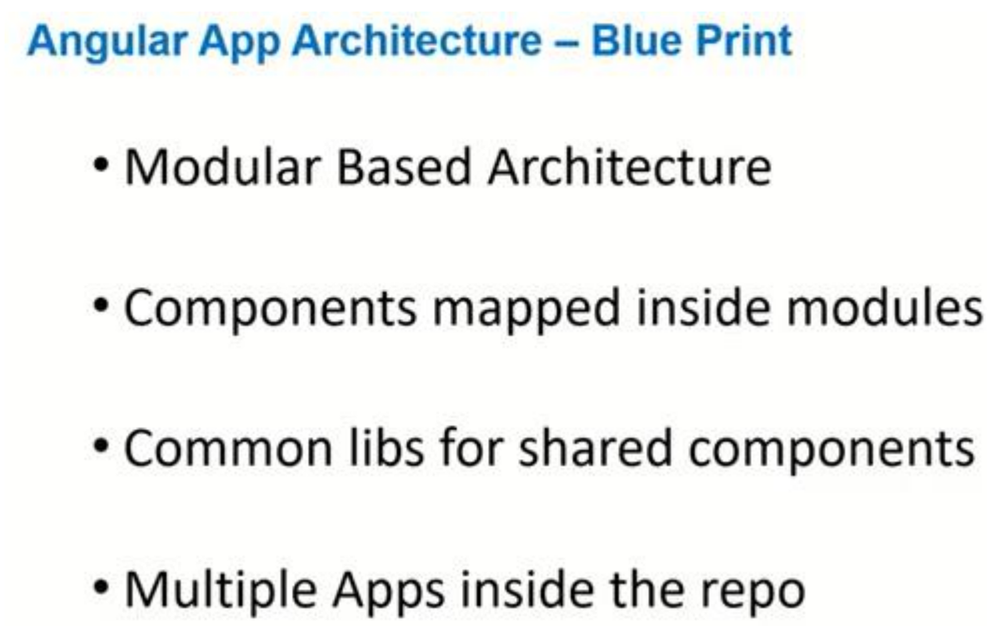
After this, Angular calls the **index.html** file. This file consequently calls the root component that is **app-root**. The root component is defined in **app.component.ts**.
This is how the index.html file looks:

```
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>Angular</title>
  <base href="/">
  <meta name="viewport" content="width=device-width, initial-scale=1">
</head>
<body>
  <app-root></app-root>
</body>
</html>
```

The HTML template of the root component is displayed inside the `<app-root>` tags.

This is how every angular application works.

Realtime Architecture of project



Angular App Architecture – Blue Print

- Modular Based Architecture
- Components mapped inside modules
- Common libs for shared components
- Multiple Apps inside the repo

Enterprises Application structure

```
ProjectApplication

  src
    contacts -> Module
      add-contact -> components inside module
      edit-contact -> components inside module
      delete-contact -> components inside module
      view-contact -> components inside module
    users
      add-user -> components inside module
    orders
    leads
    reports
    settings
    profile

    shared
      auth
      tokens

    services -> HTTP/ Resuable code
      contact-service.ts
      user-service.ts

    assets
      images
      mock-data

    pipes
      highlight-pipe
```

Thank You!!

Beginning Next Session