**Angular 7 Tutorial**

1. Angular 7 Tutorial

Auglar က java script framework ၿဖစ္ၿပီးေတာ့ **Single Page Applications** (SPAs) မ်ားကို ဖန္တီးဖို႔ရာ အသံုးၿပဳႏိုင္တယ္။ leading front-end development framework ၿဖစ္ၿပီးေတာ့ သူ႕ကို Google ရဲ႕ angular team က ပံုမွန္ updated လုပ္ေပးေနပါတယ္။ angular 7 က components မ်ားအေပၚကို လံုး၀အေၿခခံထားတဲ့ language တစ္ခုၿဖစ္တယ္။ သူ႕မွာက parent child components (tree structure)လိုမ်ိဳး

Components မ်ားစြာ ပါ၀င္ဖြဲ႕စည္းထားတယ္။

2.Angular 7 Introduction

Angular 7 ဆိုတာဘာလဲ

Augular 7 က javascript (တကယ္ကေတာ့ TypeScript ၿဖစ္ၿပီးေတာ့ open-source full-stack web application ကို အေၿခခံထားပါတယ္)။သူ႕ကို Single Page Applications (SPAs) မ်ား ေရးသားရာတြင္ အသံုးၿပဳပါတယ္။ angular 7 က components မ်ားအေပၚကို လံုး၀အေၿခခံထားတဲ့ language တစ္ခုၿဖစ္တယ္။ သူ႕မွာက parent child components (tree structure)လိုမ်ိဳး

Components မ်ားစြာ ပါ၀င္ဖြဲ႕စည္းထားတယ္။ 2+ ေနာက္ပိုင္း version မ်ားကို Angular လို႔ပဲ လူေတြသိၾကပါတယ္။ ပထမဆံုး Angular 1.0 ကို AngularJS လို႔ လူသိမ်ားပါတယ္။

တကယ္တန္းက်ေတာ့ Angular ဆိုတာက Angular js ကို ေရးသားခဲ့တဲ့ အဖြဲ႕က အတူတူပဲၿဖစ္တယ္။

Single Page Application (SPA) ဆိုတာဘာလဲ

web application တစ္ခု သို႔မဟုတ္ website တစ္ခုၿဖစ္တယ္။ သူက ဘာနဲ႔တူလည္း ဆိုေတာ့ destop application တစ္ခုနဲ႔ အလားတူပါတယ္။ သူ႕မွာ menu ေတြ button ေတြ block ေတြက single page တိုင္းမွာ ပါ၀င္ၿပီးေတာ့ user က တစ္ခုခုကို click ႏွိပ္လိုက္ခ်ိန္မွာ server ဆီကေန page အသစ္တစ္ခုကို loading ဆြဲတင္တာထက္ current page ကို dynamically rewrite ၿပန္လုပ္ေပးပါတယ္။ အဲ့တာက သူ႕ဘာလို႔ၿမန္ဆန္ေနတာလဲဆိုတဲ့ အေၾကာင္းၿပခ်က္ၿဖစ္တယ္။

AngularJS နဲ႔ Angular ဘာကြာသလဲ

|  |  |
| --- | --- |
| **AngularJS** | **Angular** |
| AngularJS သည္ common ၿဖစ္ၿပီး လူသိမ်ားေသာ Angular 1.0 ရဲ႕ နာမည္ၿဖစ္တယ္။ | Angular သည္ common ၿဖစ္ၿပီး လူသိမ်ားေသာ Angular version 2 ေနာက္ပိုင္း ရဲ႕ နာမည္ၿဖစ္တယ္။ |
| JavaScript-based open-source front-end web framework | TypeScript-based open-source full-stack web application framework |
| AngularJS က controller သို႔ scope ရဲ႕ concept ကို အသံုးၿပဳတယ္။ | Augular က controller နဲ႔ scope ေတြအစား သူ႕ရဲ႕ primary architectural characteristic အၿဖစ္ components မ်ားရဲ႕ hierarchy ကို အသံုးၿပဳပါတယ္။ |
| AngularJS က ရိုးရိုးရွင္းရွင္း syntax နဲ႔ေရးၿပီး HTML pages မ်ားတြင္ source location နဲ႔အတူ အသံုးၿပဳပါတယ္။ | Angular က မတူညီတဲ့ expression syntax ကို အသံုးၿပဳပါတယ္။ property binding အတြက္“[]” event binding အတြက္ "( )" ကို အသံုးၿပဳပါတယ္။ |
| AngularJS က ရိုးရွင္းတဲ့ javascript file တစ္ခုၿဖစ္ၿပီးေတာ့ သူ႕ကို html page ေတြမွာ အသံုးၿပဳတယ္။ server-side programming language တစ္ခုရဲ႕ feature မ်ားကို support မလုပ္ႏိုင္ပါ။ | Angular က Microsoft's TypeScript language ကို အသံုးၿပဳတယ္။ သူက Class-based Object Oriented Programming, Static Typing, Generics အစရွိတာေတြကို လုပ္ေဆာင္ေပးပါတယ္။ အဲ့တာေတြက server-side programming language တစ္ခုရဲ႕ feature မ်ားၿဖစ္ၾကပါတယ္။ |
| AngularJS က page ရဲ႕ dynamic loading ကို support မလုပ္ႏိုင္ပါ။ | Angular က page ရဲ႕ dynamic loading ကို support လုပ္ႏိုင္ပါတယ္။ |

3. Angular Features

Angular က platform မ်ားစြာကို support လုပ္ႏိုင္တယ္

cross platform language တစ္ခုၿဖစ္တယ္။ platform မ်ားစြာကို support လုပ္ႏိုင္တယ္။ Angular ကိုသံုးၿပီးေတာ့ မတူညီတဲ့ application မ်ားစြာကို ေရးသားႏိုင္တယ္။

**Desktop applications:** Windows, Mac or Linux ဘယ္ os မွာမဆို Angular methods ေတြအလုပ္လုပ္ပံုက တူညီၾကတယ္။

**Native applications:** Cordova, Ionic, or NativeScript မွ strategies နဲ႔ angular ကို သံုးၿပီးေတာ့ native apps မ်ားကို ေရးသားလို႔ ရႏိုင္ပါတယ္။

**Progressive web applications:** Progressive web applications မ်ားသည္ Angular နဲ႔ တည္ေဆာက္ထားတဲ့ အသံုးအမ်ားဆံုးေသာ apps မ်ားၿဖစ္ၾကပါတယ္။ Angular က modern web platform capabilities မ်ားကို ေပးစြမ္းပါတယ္။

High Speed, Ultimate Performance

Angular က ေအာက္က reason သံုးခုေၾကာင့္ ၿမန္ဆန္စြာ လုပ္ေဆာင္ေနႏိုင္တာၿဖစ္ပါတယ္။

Universal support: Angular ကို Node.js, .Net, PHP, Java Struts and Spring နဲ႔ အၿခား server မ်ားၿဖစ္ၾကေသာ HTML နဲ႔ CSS ကဲ့သို႔ေသာ near-instant rendering အတြက္ front-end web development tool တစ္ခုအၿဖစ္ အသံုးၿပဳႏိုင္ပါတယ္။ ပိုမိုေကာင္းမြန္ေသာ SEO အတြက္ website ကိုလည္း ေကာင္းမြန္ေအာင္ ၿပဳလုပ္ေပးပါတယ္။

Code splitting: ၿမန္ဆန္တယ္ load quickly ၿဖစ္တယ္ ဘာလို႔လဲဆိုေတာ့ Component Router မ်ားေၾကာင့္ၿဖစ္တယ္။ automatic code-splitting ပါ၀င္ေသာေၾကာင့္ user မ်ားက သူတို႔ request လုပ္တဲ့ view ကိုပဲ render လုပ္ရန္ လိုအပ္ေသာ code ကို load လုပ္ဖို႔ရန္ လိုအပ္ပါတယ္။

Code generation: Angular က ကၽြန္ေတာ္တို႔ရဲ႕ template မ်ားကို ယေန႔အခ်ိန္မွာ highly optimized code နဲ႔ ၿပဳလုပ္သလား။

JavaScript virtual machines က hand-written code ရဲ႕ အက်ိဳးေက်းဇူးေတြကို ေပးပါတယ္။

Productivity

Powerful templates: UI view ကို ၿမန္ဆန္စြာ create လုပ္ႏုိင္ရန္ ရိုးရွင္းၿပီး powerful ၿဖစ္တဲ့ template မ်ားကို ေပးပါတယ္။

IDEs: popular ၿဖစ္တဲ့ editors မ်ားနဲ႔ IDEs မ်ားတြင္ intelligent code completion, instant errors, and other feedback မ်ားကို Angular က ေပးထားပါတယ္။

Angular CLI: Angular က command line tools start building fast, add components and tests, and then instantly deploy အစရွိသည္တို႔ကိုလည္း ေပးထားပါတယ္။

Full Stack Development

Testing: Angular က unit testing အတြက္ Karma နဲ႔ Jasmine ကို support လုပ္ပါတယ္။ သူ႕ကို အသံုးၿပဳၿခင္းၿဖင့္ ကၽြန္ေတာ္တို႔ရဲ႕ broken things မ်ားကို ကၽြန္ေတာ္တို႔က save တိုင္းမွာ check လုပ္ႏိုင္တယ္။ Karma သည္ Angular team မွ create လုပ္ေသာ JavaScript test runner tool တစ္ခုၿဖစ္ပါတယ္။ Jasmine က Angular apps မ်ားထဲမွာရွိတဲ့ testing framework form unit testing ၿဖစ္တယ္။ Karma က ကၽြန္ေတာ္တို႔ code ေရးေနစဥ္မွာ ကၽြန္ေတာ္တို႔ရဲ႕ Jasmine test မ်ားကို call ရန္ ပိုမိုလြယ္ကူေစပါတယ္။

Animation Support: Angular က high-performance, complex choreographies and animation timelines မ်ားကို Angular's intuitive API ကို ၿဖတ္သန္းၿပီးေတာ့ code အနည္းငယ္နဲ႔ create လုပ္ႏိုင္ရန္ ကူညီပါတယ္။

Accessibility: Angular မွာ ARIA-enabled components, developer guides, and built-in a11y test infrastructure နဲ႔ accessible applications မ်ားကို create လုပ္ႏိုင္ပါတယ္။

4. Angular 7 installation

Angular 7 Environment Setup

Visual Studio Code IDE သို႔ JetBrains WebStorm ကို install လုပ္မယ္။

သင္၏ Angular 7 အက္ပလီေကးရွင္းကိုအသုံးျပဳရန္ Visu Studio Code IDE သို႔မဟုတ္ JetBrains WebStorm ကဲ့သို႔ IDE တစ္ခုရွိရမည္။

VS Code သည္ေပါ့ပါးၿပီး setup လုပ္ရန္လြယ္ကူသည္။ ၎တြင္ built-in code editing, format ႏွင့္ refactoring features မ်ားစြာရွိသည္။ ၎သည္အခမဲ့အသုံးျပဳႏိုင္သည္။ ၎သည္သင္၏productivity ကိုသိသိသာသာတိုးျမႇင့္ေစမည့္ မ်ားၿပားလွတဲ့ extensions မ်ားကိုလည္းေပးသည္။

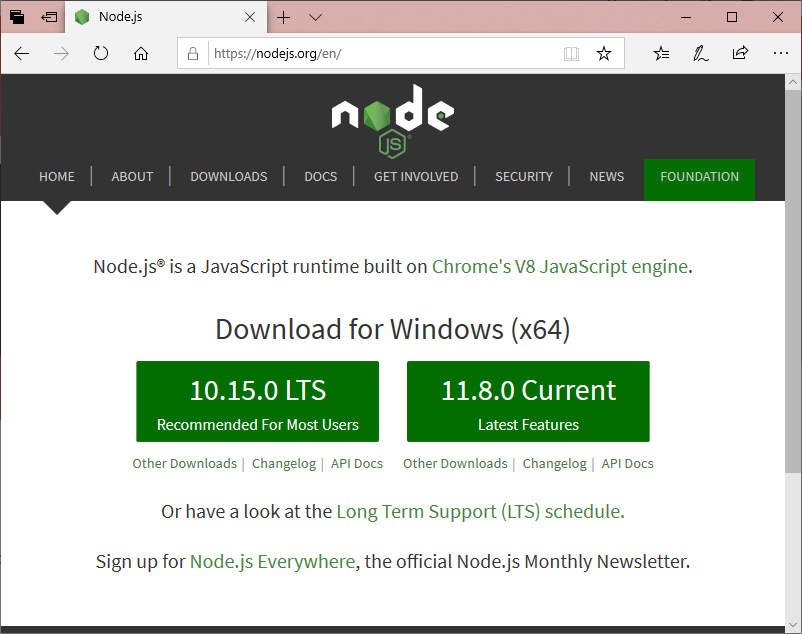
VS Code ကိုဤေနရာမွကူးယူႏိုင္သည္။ https://code.visualstudio.com

JetBrains WebStorm သည္ Angular 7 အက္ပလီေကးရွင္းမ်ားကိုတီထြင္ရန္အလြန္ေကာင္းမြန္ေသာ IDE တစ္ခုျဖစ္သည္။ ၎သည္ျမန္ဆန္။ ဆြဲေဆာင္မႈရွိၿပီးအသုံးျပဳရန္အလြန္လြယ္ကူေသာ္လည္း၎သည္အသုံးျပဳရန္အခမဲ့မဟုတ္ပါ။ ေနာက္က်မွ ၀ ယ္ယူရမွာပါ။ စမ္းသပ္မႈကာလကိုရက္ ၃၀ သာအခမဲ့ျဖစ္သည္။

VS Code ကိုဤေနရာမွရယူႏိုင္ပါသည္။ https://www.jetbrains.com/webstorm/download/#section=windows

ဤသင္ခန္းစာတြင္ JetBrains WebStorm ကိုအသုံးျပဳသည္။

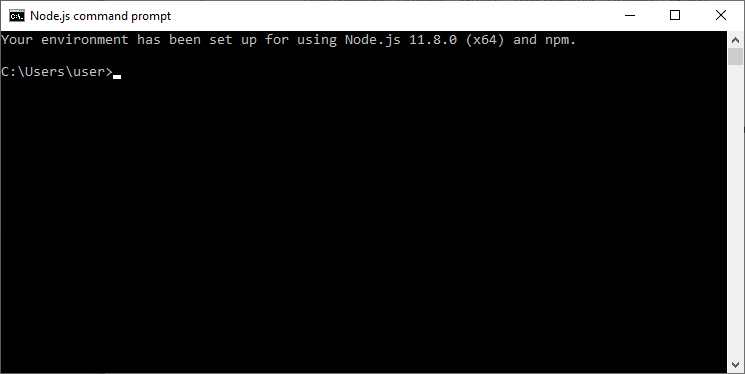
Install Node.js

Angular 7 app ကို run ရန္ node.js ကို install လုပ္သင့္တယ္။ ၎က npm dependencies အေနျဖင့္အခ်ိဳ႕ေသာ page မ်ားကို load လုပ္ေသာအခါ browser အခ်ိဳ႕ကို support လုပ္ပါတယ္။ angular project ကို run ရန္ လိုအပ္ေသာ libraries မ်ားကို ေပးပါတယ္။ Node.js က ကၽြန္ေတာ္တို႔ရဲ႕ run-time environment ကို localhost အၿဖစ္ လုပ္ေဆာင္ပါတယ္။

Just go to node.js official website [**https://nodejs.org/en/**](https://nodejs.org/en/)

Download and install latest version of node.js. In my case, it is 11.8.0

အဲ့တာၿပီးရင္ common prompt တစ္ခုတက္လာလိမ့္မယ္။ အဲ့တာကိုပိတ္လိုက္။ window key ကို ႏွိပ္ၿပီး node.js commonprompt လို႔ရိုက္လိုက္။ အဲ့တာဆိုရင္ node.js ရဲ႕ တကယ္ common prompt တစ္ခုက်လာလိမ့္မယ္။



 Angular CLI ကို install လုပ္ရန္ npm ကို သံုးၿခင္း

npm install -g @angular/cli

ေနာက္တစ္နည္းကေတာ့ Angular CLI official website [**https://cli.angular.io/**](https://cli.angular.io/) ကို သြားမယ္။

Angular app တစ္ခုကို create လုပ္ရန္  cli command တစ္ခုလံုးကို ၿမင္ေနရလိမ့္မယ္။ Angular CLI ကို install လုပ္ရန္ first command ကို run ရန္ လိုအပ္လိမ့္မယ္။

ng new projectname

cd projectname

ng serve

7. Angular 7 App files explanation

Angular 7 App folder အတြင္းရွိ အသံုးၿပဳေသာ file မ်ားအေၾကာင္း

**src folder:** ဒီေနရာမွာက ကၽြန္ေတာ္တို႔ရဲ႕ angular application နဲ႔ သက္ဆိုင္တဲ့ အဓိက code ေတြပါ၀င္တဲ့ folder တစ္ခုၿဖစ္ပါတယ္။

**app folder**: app folder ထဲမွာ app components မ်ားအတြက္ ကၽြန္ေတာ္တို႔ create လုပ္ထားေသာ files မ်ား ပါ၀င္ပါတယ္။

**app.component.css:** ဒီ file မွာက ကၽြန္ေတာ္တို႔ရဲ႕ app component အတြက္ cascading style sheets code မ်ား ပါ၀င္ပါတယ္။

**app.component.html:** ဒီ file မွာက app component နဲ႔ သက္ဆိုင္တဲ့ html မ်ားပါ၀င္ပါတယ္။ ဒီ file က data binding လုပ္ရန္ angular မွသံုးေသာ template file ၿဖစ္ပါတယ္။

**app.component.spec.ts:** ဒီ file က app component နဲ႔ သက္ဆိုင္တဲ့ unit testing file ၿဖစ္ပါတယ္။ ဒီ file ကို အၿခား unit test မ်ားအတူ အသံုးၿပဳပါတယ္။ သူ႕ကို common ng test နဲ႔ Angular CLI မွာ run ပါတယ္။

**app.component.ts:** သူကေတာ့ အေရးအပါဆံုး typescript file တစ္ခုၿဖစ္ၿပီး သူ႕မွာ component ေနာက္ကြယ္မွာ ရွိတဲ့ view logic ေတြပါ၀င္ပါတယ္။

**app.module.ts:** သူက website အတြက္ dependencies အားလံုးပါ၀င္တဲ့ typescript file တစ္ခုၿဖစ္ပါတယ္။ ဒီ file က import လုပ္ရန္ လိုအပ္သည့္ modules မ်ား declare လုပ္ရန္ component မ်ား နဲ႔ bootstrapped အတြက္ main component ကို သတ္မွတ္ရန္ အသံုးၿပဳပါတယ္။

အၿခားအေရးၾကီးေသာ file မ်ား

**package.json:** npm configuration file ၿဖစ္ပါတယ္။ သူ႕မွာ website ရဲ႕ package dependencies နဲ႔ ကၽြန္ေတာ္တို႔ရဲ႕ own website က package တစ္ခုၿဖစ္သည့္ detail မ်ားပါ၀င္ပါတယ္။

**package-lock.json :** သူက auto-generated လုပ္ၿပီး modified လုပ္ထားေသာ file ၿဖစ္သည္။ node\_modules သို႔ package.json နဲ႔ သက္ဆိုင္တဲ့ operation တစ္ခုကို npm လုပ္ေဆာင္တဲ့အခါတိုင္း update လုပ္ပါတယ္။

**angular.json:** ဒါက ကၽြန္ေတာ္တို႔ရဲ႕ angular application နဲ႔ သက္ဆိုင္ေသာ အလြန္အေရးၾကီးတဲ့ configuration file တစ္ခုၿဖစ္ပါတယ္။ သူက ကၽြန္ေတာ္တို႔ရဲ႕ app structure ကို သတ္မွတ္ၿပီး app နဲ႔ ဆက္ႏြယ္တဲ့ မည္သည့္ setting ကိုမဆိုပါ၀င္ပတ္သက္ပါတယ္။ ဒီေနရာမွာ file ရဲ႕ environment (development, production) ကို ကၽြန္ေတာ္တို႔က သတ္မွတ္ႏိုင္ပါတယ္။ ဒါက Angular 7 အလုပ္လုပ္ဖို႔ရန္ Bootstrap file ကို ထည့္ေပးလိုက္တဲ့ file ၿဖစ္ပါတယ္။

**.gitignore:** ဒီ file က source control git နဲ႔ သက္ဆိုင္တဲ့ file တစ္ခုၿဖစ္ပါတယ္။

**.editorconfig:** ဒီ file ကေတာ့ ရိုးရွင္းတဲ့ file တစ္ခုၿဖစ္ပါတယ္။ သူက code editors မ်ားမွာ indentation နဲ႔ whitespaces ကဲ့သို႔ေသာ အေၿခခံစည္းမ်ဥ္းစည္းကမ္းမ်ားကို ထိန္းသိမ္းရန္ အသံုုးၿပဳပါတယ္။

**assets folder:** ဒီ folder က images, locales, translations စတဲ့ application တြင္ အသံုးၿပဳေသာ resource files မ်ားအတြက္ placeholder ၿဖစ္ပါတယ္။

**environments folder:** အဆိုပါ angular application ကို တည္ေဆာက္ရာမွာ အေထာက္အကူၿဖစ္ေစေသာ environment configuration constants မ်ားကို hold လုပ္ထားရန္ environments folder ကို အသံုးၿပဳပါတယ္။ constants မ်ားကို separate .ts files (environment.ts and environment.prod.ts) တြင္ သတ္မွတ္ပါတယ္။ ဒီ constants မ်ားကို Angular CLI မွ angular.json file အတြင္းမွာ အသံုးၿပဳပါတယ္။ ဥပမာအားၿဖင့္ ကၽြန္ေတာ္တို႔က ng build command ကို run လုိက္မယ္ဆိုရင္ သူက development environment settings ကို အသံုးၿပဳၿပီးေတာ့ application ကို build လုပ္မွာၿဖစ္တယ္။ prod က production environment settings မ်ားနဲ႔ project ကို build လုပ္လိမ့္မယ္။

**browserlist:** ဒီ file ကို သတ္မွတ္ထားေသာ browser မ်ားကို support လုပ္ရန္ CSS ကို adjust လုပ္ရန္ autoprefixer က အသံုးၿပဳတယ္။

**favicon.ico:** website ရဲ႕ browser ရဲ႕ tab ေဘးမွာ ေပၚလာမယ့္ icon ကို သတ္မွတ္ေပးတယ္။

**index.html:** angular application အတြက္ high level container ကို hold လုပ္တဲ့ entry file ၿဖစ္တယ္။

**karma.config.js:** Karma Test Runner အတြက္ config file ကို သတ္မွတ္ေပးတယ္။ Karma ကို AngularJS and Angular 2+ ႏွစ္ခုစလံုးအတြက္ test မ်ားကို run ႏိုင္ရန္ AngularJS team က developed လုပ္ေပးထားတာ ၿဖစ္ပါတယ္။

**main.ts:** angular.json file မွာ သတ္မွတ္ထားတဲ့အတိုင္း သူက ပထမဆံုးအလုပ္လုပ္မယ့္ ts file ၿဖစ္တယ္။ ဒီ file က app.module.ts က AppModule ကို စၿပီး အလုပ္လုပ္ေစတယ္။ သူ႕ကို global configurations မ်ား သတ္မွတ္ရန္ အသံုးၿပဳပါတယ္။

**polyfills.ts:** old browser မ်ားအတြက္ compatibility support လုပ္ေပးႏိုင္တဲ့ set of code ၿဖစ္တယ္။ Angular 7 code ကို ES6+ language specifications ထဲမွာ အဓိက ေရးထားပါတယ္။ သူက front-end development ပိုင္းမွာ ပိုၿပီး အသံုး၀င္ပါတယ္။ ဒါေၾကာင့္ browsers အားလံုးက ES6+ language specifications မ်ားကို အၿပည့္အ၀မပံ့ပိုးေပးႏိုင္တာေၾကာင့္ pollyfills က ေပးထားေသာ browser မွ missing ၿဖစ္ေနေသာ မည္သည့္ feature ကိုမဆို cover လုပ္ႏိုင္တယ္။

**styles.css:** angular application မွ အသံုးၿပဳေနေသာ global css file တစ္ခုၿဖစ္ပါတယ္။

**tests.ts:** Angular CLI command ng test က application အတြင္းမွာ unit tests အားလံုးကို ၿဖတ္ၿပီး run ဖို႔ရန္အတြက္ tests.ts file ကို အသံုးၿပဳပါတယ္။

**tsconfig.json:** typecript compiler configuration file တစ္ခုၿဖစ္ပါတယ္။

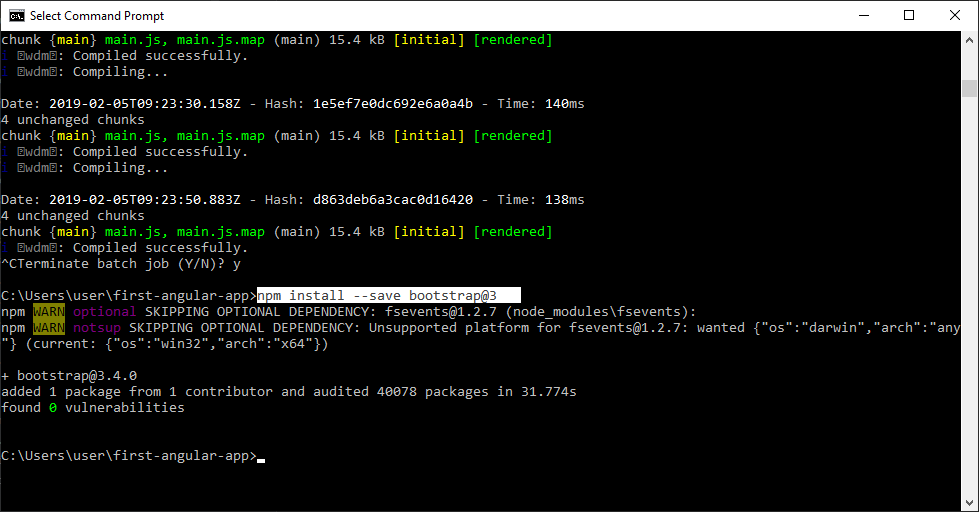
**tsconfig.app.json:** ဒီ app ကို specific configurations မ်ားႏွင့္အတူ tsconfig.json file အား override လုပ္ရန္ အသံုးၿပဳပါတယ္။

**tsconfig.spec.json:** ဒီ json က app specific unit test configurations နဲ႔အတူ tsconfig.json ကို overrides လုပ္ပါတယ္။

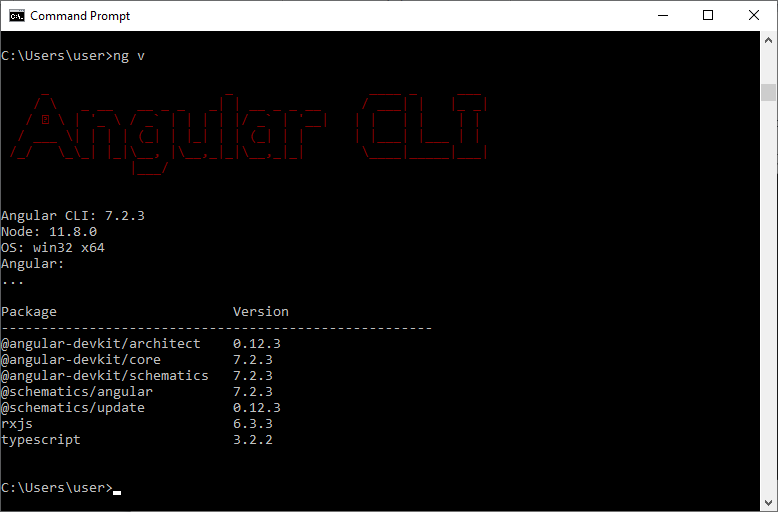
8. Angular 7 with Bootstrap

Angular project အတြက္ Bootstrap  ကို ဘယ္လို install လုပ္ရမလဲ

npm install --save bootstrap@3



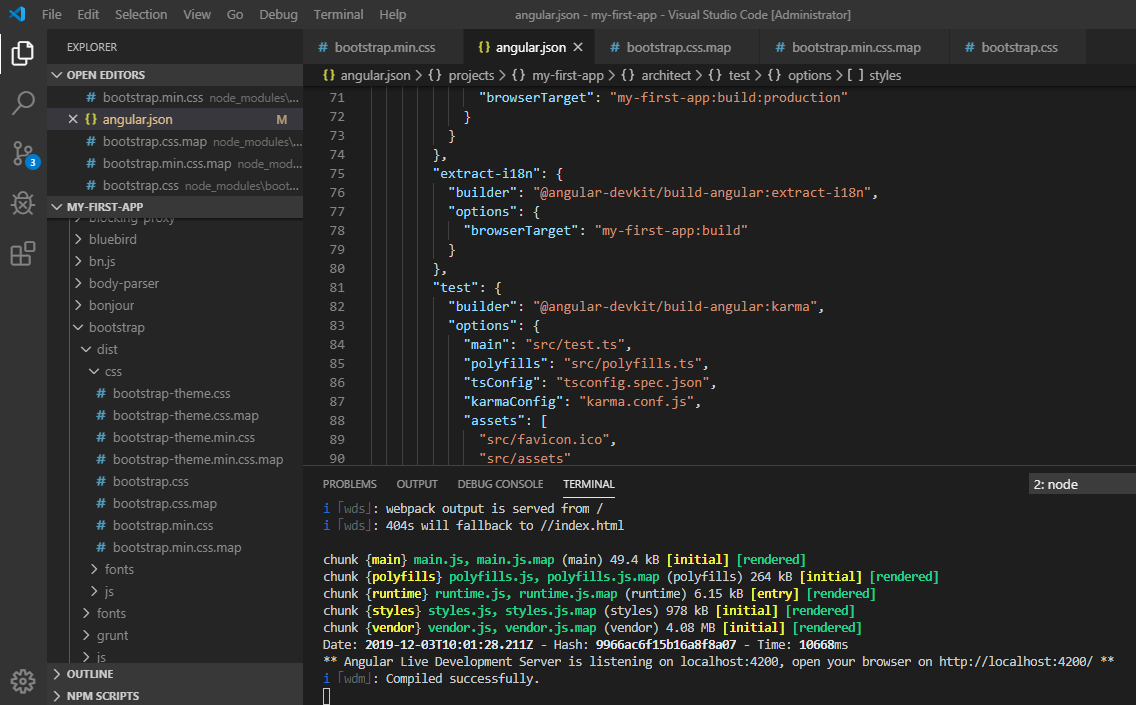
ng v

ဒါ့အၿပင္ ကၽြန္ေတာ္တို႔က Angular CLI 6+ ၿဖင့္ create လုပ္ထားေသာ project ကို အသံုးၿပဳတဲ့အခါ (ng v မွတစ္ဆင့္ စစ္ေဆး) .angular-cli.json file အစား angular.json file ရွိလိမ့္မယ္။ အဲ့ file မွာ Bootstrap ကို styles[] array ထဲကို ထပ္ထည့္ဖို႔ လိုအပ္ေသာ္လည္း path က node\_modules/bootstrap/dist/css/bootstrap.min.css ၿဖစ္ေနတယ္။ ../node\_modules/bootstrap/dist/css/bootstrap.min.css မဟုတ္ဘူး။ ../ က ပါမွာ မဟုတ္ဘူး။

ကၽြန္ေတာ္လက္ရွိ project မွာေတာ့ Angular 8.3.20 ကို သံုးထားပါတယ္။

bootstrap.css file ကို project ထဲသို႔ ဘယ္လိုထည့္မလဲ

Node Module ကို expand လုပ္လိုက္ပါ (သူက library root folder ၿဖစ္တယ္)



အခု ၀ိုင္းၿပထားတဲ့ css file ကို angular.json file ထဲကို ထည့္ေပးရမယ္။

"styles": [

  "node\_modules/bootstrap/dist/css/bootstrap.min.css",

  "src/styles.css"

],

အခုဆိုရင္ bootstrap ကို angular 7 ထဲမွာ ထည့္ၿပီးသြားၿပီ ၿဖစ္တယ္။ Run လုိ႔ရပါၿပီ။

အသံုး၀င္မယ့္ Shorcut မ်ားကေတာ့

1. file ေတြကို ရွာခ်င္ရင္ Ctr+p
2. run ေနတာကို ရပ္ခ်င္ရင္ Ctr+C

7. Angular 7 Files Explanation

Angular 7 Project Setup (Create first app)

ပထမဆုံး Angular app ကိုဖန္တီးရန္ Angular CLI command မ်ားသည္ေအာက္ပါအတိုင္းျဖစ္သည္။

npm install -g @angular/cli

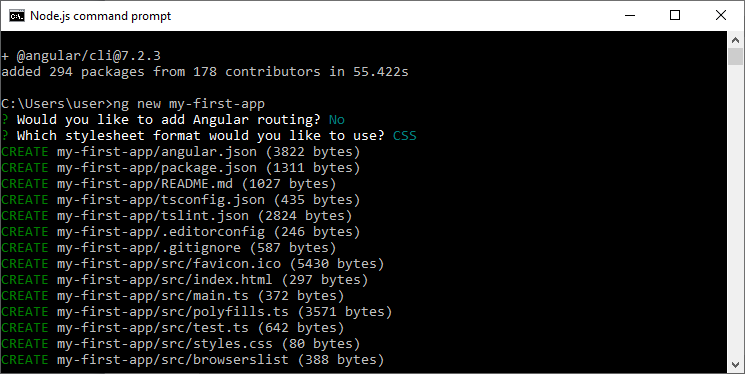
ng new my-dream-app

cd my-dream-app

ng serve

သင္၏ပထမဆုံး Angular application ကိုဖန္တီးရန္ေအာက္ပါ command ကို run ပါ။

ng new my-first-app

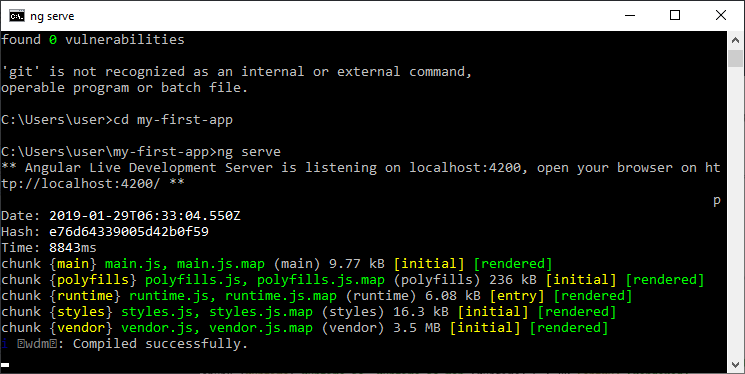


သင္၏ပထမဆုံး app သို႔သြားပါ။

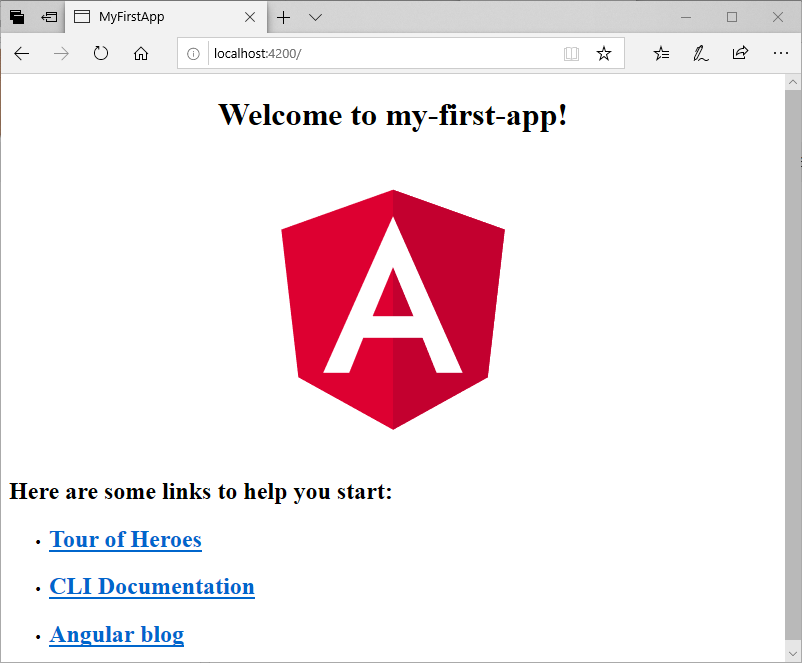
cd my-first-app

App ကို run ရန္သင္၏ server ကို start လုပ္ပါ။

ng serve

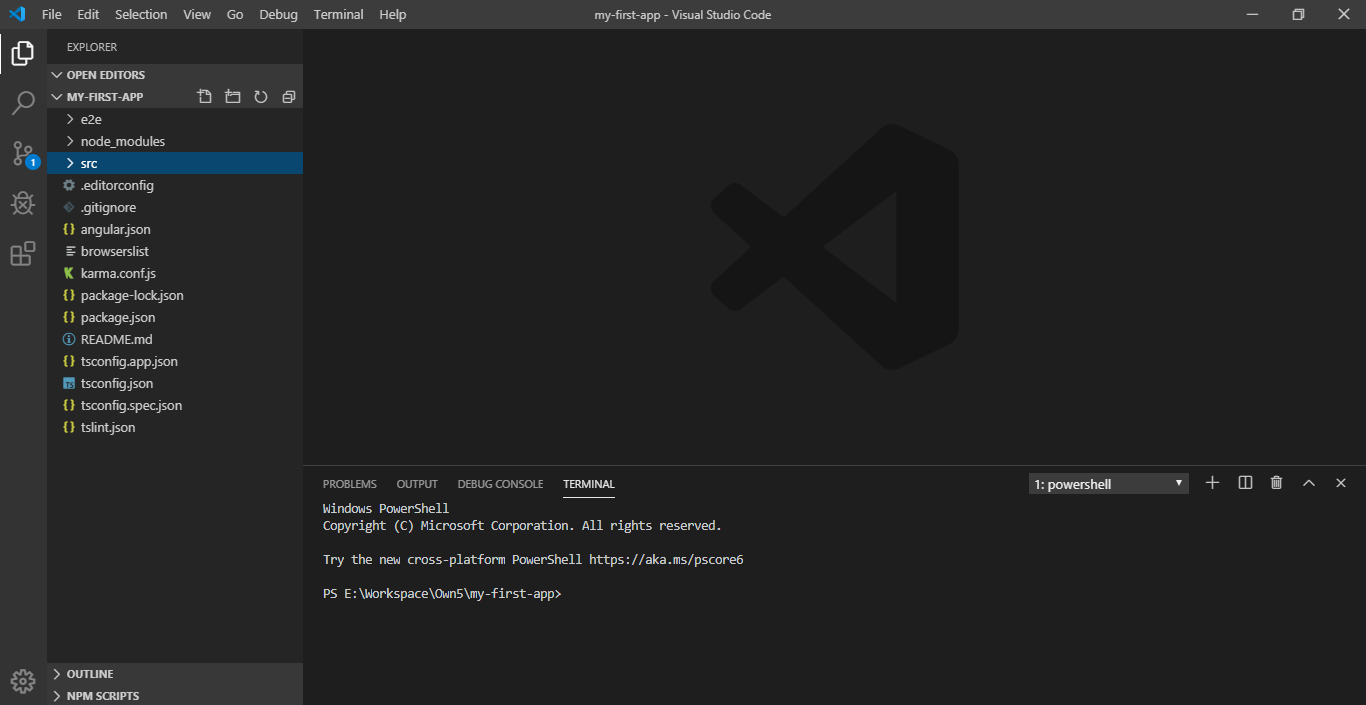


သင္၏ server သည္ localhost:4200 တြင္ run ေနသည္။ Browser ကိုဖြင့္လိုက္ပါ။

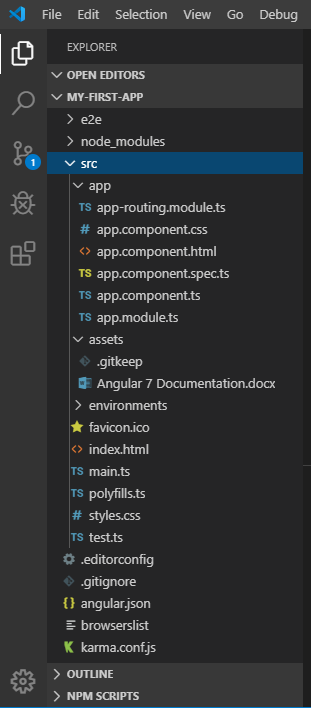


သင္၏ app ကုဒ္ကိုတည္းျဖတ္ရန္ႏွင့္ run ရန္ IDE လိုအပ္သည္။ ဒီမွာ Visual Studio Code ကိုသုံးေနတယ္။

Visual Studio Code ကိုဖြင့ ္၍ IDE တြင္သင္၏ "my-first-app" အက္ပ္ကိုဖြင့္ပါ။ ဒီဟာကေအာက္ပါအတိုင္းျဖစ္သည္ -



ဤေနရာတြင္ src folder သို႔သြားပါ။ ထိုေနရာတြင္ app folder ကိုေတြ႕လိမ့္မည္။ app folder ကို expand လုပ္ပါ။



အဲ့မွာ component 5 ခု ေတြ႕ရလိမ့္မယ္။

1. app.component.css
2. app.component.html
3. app.component.spec.ts
4. app.component.ts
5. app.module.ts

**app.component.css**

**ဒီေနရာမွာကေတာ့ empty ၿဖစ္ေနလိမ့္မယ္ ဘာလို႔လဲဆိုေတာ့ကၽြန္ေတာ္တို႔က ဘယ္ css ကိုမွ specify မလုပ္ရေသးလို႔ပါ။**

**app.component.html**

သူကေတာ့ အေရးၾကီးတဲ့အစိတ္အပိုင္း ၿဖစ္ပါတယ္။ ဘာလို႔လဲဆိုေတာ့ ကၽြန္ေတာ္တို႔ app ရဲ႕ front page ၿဖစ္ေနလို႔ပါ။ ဒီေနရာမွာ ကၽြန္ေတာ္တို႔ app name မတိုင္ခင္ အသံုးၿပဳေနေသာ salutation ကို ေၿပာင္းလဲႏိုင္ပါတယ္။ ၿပီးေတာ့ ကၽြန္ေတာ္တို႔အေနနဲ႔ front page ေပၚမွာ ရွိတဲ့ content တို႔ သူတို႔နဲ႔ သက္ဆိုင္တဲ့ links မ်ားကိုလည္း ေၿပာင္းလဲႏိုင္ပါတယ္။

**app.component.spec.ts:**

သူ႕ကို အသံုးၿပဳရတဲ့ ရည္ရြယ္ခ်က္ေတာ့ testing only အတြက္ပဲ ၿဖစ္ပါတယ္။

**app.component.ts**

**ကၽြန္ေတာ္တို႔ရဲ႕ app name နဲ႔ title ကို ေၿပာင္းလဲႏိုင္ပါတယ္။**

**9. All CLI Commands**

Angular CLI က command line interface tool တစ္ခုျဖစ္ၿပီးေတာ့ သူ႕ကို Angular applications မ်ား initialize လုပ္ၿခင္း develop လုပ္ၿခင္း scaffold လုပ္ၿခင္း နဲ႔ maintain လုပ္ၿခင္းမ်ား လုပ္ေဆာင္ပါတယ္။ ကၽြန္ေတာ္တို႔အေနနဲ႔ တိုက္ရိုက္နည္းလမ္းၿဖစ္တဲ့ command prompt မွ command ရိုက္ၿပီးေတာ့ေသာ္လည္းေကာင္း Angular Console ကဲ့သို႔ေသာ interactive UI မ်ားကိုအသံုးၿပဳၿပီးေသာ္လည္းေကာင္း Angular CLI ကို အသံုးၿပဳႏိုင္ပါတယ္။

|  |  |  |
| --- | --- | --- |
| Command | Alias | Description |
| add |  | ကၽြန္ေတာ္တို႔ရဲ႕ project မွာ external library အတြက္ support လုပ္ရန္အသံုးၿပဳပါတယ္။ |
| build | b | ေပးထားေသာ output path မွာရွိတဲ့ dist/ လို႔အမည္ရွိတဲ့ output directory ထဲကို Angular app ကို compile လုပ္ေပးပါတယ္။ workspace directory ထဲမွာ executed လုပ္ေပးရမွာ ၿဖစ္ပါတယ္။ |
| config |  | သူက workspace အတြက္ angular.json file တြင္ Angular configuration values မ်ားကို retrieves or sets တစ္ခုခုကို လုပ္ေဆာင္ေပးပါတယ္။ |
| doc | d | သူက browser တစ္ခုအတြင္းမွာ official Angular documentation (angular.io) ကို open လုပ္ေပးၿပီးေတာ့ ေပးထားတဲ့ keyword တစ္ခုအတြက္ search လုပ္ေပးပါတယ္။ |
| e2e | e | သူက Angular app ကို builds and serves လုပ္ပါတယ္။ အဲ့ေနာက္ Protractor ကို အသံုးၿပဳၿပီးေတာ့ end-to-end tests ကို run ပါတယ္။ |
| generate | g | schematic တစ္ခုအေပၚကို အေၿခခံၿပီးေတာ့ files မ်ားကို generate လုပ္မယ္ modifies လုပ္ပါမယ္။ |
| help |  | available commands မ်ားရဲ႕ list ကို ေပးပါတယ္ ၿပီးေတာ့ သူတို႔ရဲ႕ short descriptions မ်ားကိုလည္း ေပးထားပါေသးတယ္။ |
| lint | l | သူကေတာ့ given project folder ထဲမွာ Angular app code ေပၚမွာ linting tools မ်ားကို run ရန္ အသံုးၿပဳပါတယ္။ |
| new | n | workspace အသစ္တစ္ခုနဲ႔ initial Angular app တစ္ခုကို create လုပ္ေပးပါတယ္။ |
| run |  | ကၽြန္ေတာ္တို႔ရဲ႕ project မွာ သတ္မွတ္ထားေသာ optional custom builder configuration နဲ႔ Architect target ကို run ပါတယ္။ |
| serve | s | ကၽြန္ေတာ္တို႔ရဲ႕ app ကို build and serve လုပ္ေပးသလို file ေတြကို change ခဲ့မယ္ဆိုရင္လည္း rebuilding လုပ္ေပးပါတယ္။ |
| test | t | project တစ္ခုအတြင္းမွာရွိတဲ့ unit tests မ်ားကို run ေပးပါတယ္။ |
| update |  | ကၽြန္ေတာ္တို႔ရဲ႕ application နဲ႔ သူ႕ရဲ႕ dependencies မ်ားကို update လုပ္ေပးပါတယ္။ |
| version | v | Angular CLI version ကို update လုပ္ေပးပါတယ္။ |
| xi18n |  | source code ကေနၿပီး i18n messages မ်ားကို extracts လုပ္ပါတယ္။ |

ng add Command

ေရးထားတာေတာ့ အမ်ားၾကီးပဲ but ဆိုလိုတာေတာ့ library ကို add လုပ္လိုက္တဲ့သေဘာပဲၿဖစ္တယ္။

Syntax:

ng add **<collection>** [options]

Parameter Explanation:

<collection>: ကၽြန္ေတာ္တို႔ ထည့္သြင္းခ်င္ေသာ package ကို ေရးေပးရမယ္။

Options

**--defaults=true|false: When true, it disables interactive input prompts for options with a default.**

--help=true|false|json|JSON: console မွာ help message ကို ၿပဖို႔အတြက္ အသံုးၿပဳတယ္။ default က false.

--interactive=true|false: false ၿဖစ္တဲ့အခါက်ရင္ interactive input prompts မ်ားကို disables လုပ္ပါတယ္။

ng build Command

Angular app ကို ေပးထားေသာ output directory မွ dist/ လို႔ေခၚတဲ့ output directory ထဲသို႔ compile လုပ္ရန္ ng build command ကို အသံုးၿပဳပါတယ္။ သူက workspace directory အတြင္းမွ execute လုပ္ေဆာင္ပါတယ္။

Syntax:

ng build <project> [options]

ng b <project> [options]

Parameter Explanation:

**<project>:** build လုပ္မယ့္ project ရဲ႕ name ကို သတ္မွတ္ေပးပါတယ္။

Options

--aot=true|false: Time compilation ရဲ႕ ahead ကို အသံုးၿပဳၿပီး build လုပ္ေပးတယ္။ default: false

--baseHref=baseHref: build လုပ္ေနေသာ application အတြက္ base url ကို သတ္မွတ္ေပးတယ္။

--buildEventLog=buildEventLog: Build Event Protocol events မ်ားအတြက္ output file path ၿဖစ္ပါတယ္။(experimental)

--buildOptimizer=true|false: 'aot' option ကို အသံုးၿပဳေသာအခါ သူက '@angular-devkit/build-optimizer' ကို enable ၿဖစ္ေစတယ္။ default: false

--commonChunk=true|false: across multiple bundles မ်ားကို အသံုးၿပဳထားတဲ့ separate bundle containing code ကို အသံုးၿပဳပါတယ္။ default: true

--configuration=configuration: A named build target, as specified in the "configurations" section of angular.json. အဲ့ဒီ target အတြက္ option defaults ရဲ႕ configuration တစ္ခုက named target တစ္ခုခ်င္းစီကို လိုက္ပါေစပါတယ္။

Aliases: -c

--deleteOutputPath=true|false: building မလုပ္ခင္မွာ output path ကို delete လုပ္ဖို႔ အသံုးၿပဳပါတယ္။Default: true

--deployUrl=deployUrl: file မ်ားကို ဘယ္ေနရာမွာ deployed လုပ္မယ္ဆိုတဲ့ URL

--es5BrowserSupport=true|false: Enables conditionally loaded ES2015 polyfills. Default: false

--extractCss=true|false: သူ႕ကို js အစား global styles မ်ားမွ extract လုပ္ရန္ အသံုးၿပဳပါတယ္။ Default: false

--extractLicenses=true|false: separate file တစ္ခုမွာ licenses မ်ားအားလံုးကို extract လုပ္ရန္ အသံုးၿပဳပါတယ္။ Default: false

-forkTypeChecker=true|false: အဲ့တာကို TypeScript type checker ကို forked process တစ္ခုမွာ run ဖို႔ရန္ အသံုးၿပဳပါတယ္။ Default: true

--help=true|false|json|JSON: ဒီ command အတြက္ console မွာ help message ကို ၿပသရန္ အသံုးၿပဳပါတယ္။ Default: false

--i18nFile=i18nFile: i18n အတြက္ အသံုးၿပဳရန္ Localization file

--i18nFormat=i18nFormat: --i18n-file နဲ႔ သတ္မွတ္ထားတဲ့ localization file ရဲ႕ format ၿဖစ္တယ္။

--i18nLocale=i18nLocale: i18n အတြက္ အသံုးၿပဳရန္ Locale

--i18nMissingTranslation=i18nMissingTranslation: i18n အတြက္ missing translations မ်ားကို ဘယ္လို handle လုပ္မလဲ။

--index=index: index HTML file ရဲ႕ name

--lazyModules: additional NgModule files မ်ားရဲ႕ list ၿဖစ္တယ္။ သူတို႔က lazy loaded. Lazy router modules မ်ားကို အလိုအေလ်ာက္ ရွာေဖြေတြ႕ရွိလိမ့္မယ္။

--main=main: current workspace နဲ႔ ႏွိဳင္းယွဥ္လွ်င္ app သို႔ main entry point အတြက္ full path ၿဖစ္တယ္။

--namedChunks=true|false: lazy loaded chunk အတြက္ file name ကို အသံုးၿပဳပါ။ Default: true

--ngswConfigPath=ngswConfigPath: ngsw-config.json အတြက္ path

--optimization=true|false: build output ရဲ႕ optimization ကို enable လုပ္ေပးတယ္။

--outputHashing= none|all|media|bundles: output filename cache-busting hashing mode ကို define လုပ္တယ္။ Default: none

--outputPath=outputPath: current workspace နဲ႔ ဆက္စပ္ေနေသာ new output directory အတြက္ full path ၿဖစ္တယ္။

By default, current project အတြင္းရွိ folder named dist/ သို႔ output ကို write ေပးတယ္္

--poll: milliseconds အတြင္း စစ္တမ္းေကာက္ယူသည့္အခ်ိန္ကာလကို ေစာင့္ၾကည့္ေသာ file ကို ဖြင့္ၿပီး define လုပ္တယ္ enable လုပ္တယ္။

--polyfills=polyfills: current workspace နဲ႔ ဆက္စပ္ၿပီး polyfills file အတြက္ full path ၿဖစ္တယ္။

--preserveSymlinks=true|false: module မ်ားကို resolve လုပ္ေသာအခါ real path ကို မသံုးပါဘူး။ Default: false

--prod=true|false: true ၿဖစ္တဲ့အခါ build configuration ကို set လုပ္ဖို႔ အတြက္ သတ္မွတ္ထားတယ္။ builds အားလံုးက bundling and limited tree-shaking ရဲ႕ အသံုးၿပဳၿခင္းကို အသံုးၿပဳတယ္။ production build တစ္ခုက limited dead code elimination ကို run ေပးပါတယ္။

--profile=true|false: Chrome profiler အတြက္ Output profile events မ်ား။Default: false

--progress=true|false: building လုပ္ေနစဥ္မွာ log process ကို console မွာ လုပ္ေဆာင္။

--resourcesOutputPath= resourcesOutputPath: outputPath နဲ႔ ဆက္စပ္ၿပီး style resources ကို ေနရာခ်မည့္ path။

--serviceWorker=true|false: production builds ေတြအတြက္ service worker config ကို generate လုပ္ေပးပါတယ္။Default: false

--showCircularDependencies=true|false: circular dependency warnings on builds မ်ားကို ၿပသေပးတယ္။ Default: true

--sourceMap=true|false: Output sourcemaps မ်ားကို ၿပသဖို႔ အသံုးၿပဳပါတယ္။Default: true

--statsJson=true|false: သူက 'webpack-bundle-analyzer' သို႔မဟုတ္ <https://webpack.github.io/analyse> ကဲ့သို႔ေသာ tools မ်ားကို သံုးၿပီးေတာ့ ခြဲၿခမ္းစိတ္ၿဖာႏိုင္ေသာ 'stats.json' ဆိုတဲ့ file တစ္ file ကို generates လုပ္ေပးပါတယ္။Default: false

--subresourceIntegrity=true|false: သူ႕ကို subresource integrity validation အသံုးၿပဳၿခင္းအတြက္ enables လုပ္ပါတယ္။ Default: false

--tsConfig=tsConfig: current workspace နဲ႔ ဆက္စပ္ၿပီး TypeScript configuration file အတြက္ full path။

--vendorChunk=true|false: သူက vendor libraries တည္းသာလွ်င္ ပါ၀င္သည့္ separate bundle ကို အသံုးၿပဳပါတယ္။Default: true

--verbose=true|false: သူက output logging ဆီသို႔ more details မ်ား ေပါင္းထည့္ပါတယ္။Default: false

--watch=true|false: file မ်ား change တဲ့အခါ သူက build ကို run တယ္။Default: false

ng config Command

Angular configuration values မ်ားကို workspace အတြက္ angular.json file တြင္ retrieve လုပ္ရန္ သို႔မဟုတ္ set လုပ္ရန္ ng config command အား အသံုးၿပဳပါတယ္။

Syntax:

ng config **<jsonPath>** **<value>** [options]

Parameter Explanation:

<jsonPath>: JSON path format ထဲမွာ set or query လုပ္ရန္ configuration key ၿဖစ္တယ္။ for example: "a[3].foo.bar[2]"

new value ကို မေပးပါက ဒီ key ရဲ႕ current value ကို return ၿပန္ေပးပါတယ္။

<value>: ေပးထားခဲ့ရင္ configuration key အတြက္ new value တစ္ခုေပးတယ္။

Options

**--global=true|false: true ၿဖစ္ခဲ့လွ်င္ caller's home directory ထဲမွာ သူက global configuration ကို access လုပ္ပါတယ္။** **Default: false**

Aliases: -g

**--help= true|false|json|JSON: ဒီ command အတြက္ console မွာ help message ကိုၿပရန္ အသံုးၿပဳတယ္။** **Default: false**

ng doc Command

ng doc command ကို browser တြင္ official Angular documentation (angular.io) အား ဖြင့္ရန္ႏွင့္ ေပးထားေသာ keyword ကို ရွာေဖြရန္ အသံုးၿပဳပါတယ္။

Syntax:

ng doc **<keyword>** [options]

ng d **<keyword>** [options]

**Parameter Explanation:**

**<keyword>: angular.io ရွိတဲ့ search bar တြင္ ေဖာ္ၿပထားသကဲ့သို႔ သူ႕ကို ရွာေဖြရန္ keyword သတ္မွတ္ရန္ အသံုးၿပဳတယ္။**

**Options**

**--help=true|false|json|JSON: ဒီ command အတြက္ console မွာ help message ကို ၿပရန္ အသံုးၿပဳတယ္။** Default: false

**--search=true|false: true ၿဖစ္ခဲ့ရင္ angular.io ရဲ႕အားလံုးကို သူက ရွာေဖြေပးတယ္။ မဟုတ္ဘူးဆိုရင္ API reference documentation သာ ရွာေဖြေပးတယ္။ Default: false**

Aliases: -s

ng e2e Command

**သူ႕ကို Angular app တစ္ခုကို build လုပ္ရန္ ႏွင့္ serve လုပ္ရန္ အတြက္ အသံုးၿပဳတယ္။ ထို႔ေနာက္ Protractor ကို အသံုးၿပဳၿပီးေတာ့ end-to-end tests ကို run ပါတယ္။**

**Syntax:**

**ng e2e <project> [options]**

**ng e <project> [options]**

**သူက workspace directory အတြင္းမွ executed လုပ္ရမွာ ၿဖစ္တယ္။ ကၽြန္ေတာ္တို႔က project name ကို specify မလုပ္တဲ့အခါ သူက projects အားလံုးအတြက္ executes လုပ္ပါတယ္။**

**Parameter Explanation:**

**<project>: သူက ကၽြန္ေတာ္တို႔ build လုပ္လိုေသာ project ရဲ႕ name ကို သတ္မွတ္ေပးပါတယ္။ သူက library တစ္ခု သို႔မဟုတ္ app တစ္ခု ၿဖစ္ႏိုင္ပါတယ္။**

**Options**

**--baseUrl=baseUrl: သူက protractor ကို connect လုပ္ရန္အတြက္ base URL ကို specifies လုပ္ေပးပါတယ္။**

**--configuration=configuration: angular.json ရဲ႕ "configurations" section တြင္ သတ္မွတ္ထားတဲ့အတိုင္း သူ႕က build target ကို သတ္မွတ္ရန္ အသံုးၿပဳတယ္။ named target တစ္ခုခ်င္းစီကို အဲ့ target အတြက္ option defaults မ်ားရဲ႕ configuration တစ္ခုက လိုက္ပါၿခင္း ခံရတာၿဖစ္တယ္။**

**Aliases: -c**

**--devServerTarget=devServerTarget: စမ္းသပ္မွဳမ်ားၿပဳလုပ္ရန္ dev server target ကို သတ္မွတ္တယ္။**

**--elementExplorer=true|false: သူက debugging အတြက္ Protractor's Element Explorer ကို start လုပ္ပါတယ္။** **Default: false**

**--help= true|false|json|JSON: ဒီ command အတြက္ console ထဲက help message ကို ၿပတယ္။** **Default: false**

**--host=host: နားေထာင္ရန္ Host ၿဖစ္တယ္။ Default: localhost**

**--port: သူက application ကို serve လုပ္ရန္ port ကို သတ္မွတ္ေပးတယ္။**

**--prod=true|false: true ၿဖစ္တဲ့အခါ သူက production target ဆီသို႔ build configuration ကို ထည့္သြင္းတယ္။ builds အားလံုးက bundling and limited tree-shaking ရဲ႕ အသံုးၿပဳၿခင္းကို ၿပဳလုပ္တယ္။ production build တစ္ခုက limited dead code elimination ကိုလည္းပဲ run ေပးပါတယ္။**

**--protractorConfig= protractorConfig: Protractor configuration file ရဲ႕ နာမည္ကို specifies လုပ္ပါတယ္။**

**--specs: သူက protractor config ထဲမွာ specs ကို overrides လုပ္ပါတယ္။**

**--suite=suite: သူက protractor config ထဲမွာ suite ကို overrides လုပ္ပါတယ္။**

**--webdriverUpdate=true|false: သူက webdriver ကို update လုပ္ရန္ အသံုးၿပဳတယ္။** **Default: true**

ng generate Command

schematic ကို အေၿခခံၿပီးေတာ့ file ကို generate လုပ္ရန္ သို႔ modify လုပ္ရန္ ng generate command ကို အသံုးၿပဳပါတယ္။

ng generate **<schematic>** [options]

ng g **<schematic>** [options]

**Parameter Explanation:**

<schematic >: ဒါက ကၽြန္ေတာ္တို႔ generate လုပ္ခ်င္ေသာ schematic သို႔မဟုတ္ collection ကို သတ္မွတ္ေပးပါတယ္။ သူက **ေအာက္ပါ sub-command မ်ားထဲမွ တစ္ခုကို ယူႏိုင္တယ္။**

1. appShell
2. application
3. class
4. component
5. directive
6. enum
7. guard
8. interface
9. library
10. module
11. pipe
12. service
13. serviceWorker
14. universal

appShell: သူ႕ကို application တစ္ခုရဲ႕ server side version အတြက္ အသံုးၿပဳရန္ app shell တစ္ခုထုတ္လုပ္ရန္ အသံုးၿပဳပါတယ္။

Syntax:

ng generate appShell [options]

ng g appShell [options]

**application**

**သူ႕ကို workspace ရဲ႕ "projects" subfolder တြင္ အေၿခခံ app အသစ္တစ္ခုကို create လုပ္ရန္ အသံုးၿပဳပါတယ္။**

**Syntax:**

**ng generate application <name> [options]**

**ng g application <name> [options]**

**class**

**default သို႔ given project တြင္ generic class definition အသစ္တစ္ခုုကို create လုပ္ရန္ အသံုးၿပဳပါတယ္။**

ng generate class **<name>** [options]

ng g class **<name>** [options]

component

**default သို႔ given project တြင္ generic component definition အသစ္တစ္ခုုကို create လုပ္ရန္ အသံုးၿပဳပါတယ္။**

ng generate component **<name>** [options]

ng g component **<name**

directive

**default သို႔ given project တြင္ generic** directive **definition အသစ္တစ္ခုုကို create လုပ္ရန္ အသံုးၿပဳပါတယ္။**

ng generate directive **<name>** [options]

ng g directive **<name>** [options]

enum

**default သို႔ given project တြင္ generic** enum **definition အသစ္တစ္ခုုကို create လုပ္ရန္ အသံုးၿပဳပါတယ္။**

ng generate enum **<name>** [options]

ng g enum **<name>** [options]

guard

**default သို႔ given project တြင္ generic route guard definition အသစ္တစ္ခုုကို create လုပ္ရန္ အသံုးၿပဳပါတယ္။**

ng generate enum **<name>** [options]

ng g enum **<name>** [options]

interface

**default သို႔ given project တြင္ generic interface definition အသစ္တစ္ခုုကို create လုပ္ရန္ အသံုးၿပဳပါတယ္။**

ng generate interface **<name>** **<type>** [options]

ng g interface **<name>** **<type>** [options]

library

**default သို႔ given project တြင္ generic** library **definition အသစ္တစ္ခုုကို create လုပ္ရန္ အသံုးၿပဳပါတယ္။**

ng generate library **<name>** [options]

ng g library **<name>** [options]

module

**default သို႔ given project တြင္ generic** NgModule **definition အသစ္တစ္ခုုကို create လုပ္ရန္ အသံုးၿပဳပါတယ္။**

ng generate module **<name>** [options]

ng g module **<name>** [options]

pipe

**default သို႔ given project တြင္ generic** pipe **definition အသစ္တစ္ခုုကို create လုပ္ရန္ အသံုးၿပဳပါတယ္။**

ng generate pipe **<name>** [options]

ng g pipe **<name>** [options]

service

**default သို႔ given project တြင္ generic** service **definition အသစ္တစ္ခုုကို create လုပ္ရန္ အသံုးၿပဳပါတယ္။**

ng generate service **<name>** [options]

ng g service **<name>** [options]

serviceWorker

ဒီ schematic ကို service worker တစ္ခု create လုပ္ရန္ "run" command ကို pass လုပ္ရန္ အသံုးၿပဳပါတယ္။

ng generate serviceWorker [options]

ng g serviceWorker [options]

universal

ဒီ command ကို app ရဲ႕ server-side rendering ကို set up လုပ္ရန္ "run" command ကို ဒီ schematic ကို pass လုပ္ရန္ အသံုးၿပဳပါတယ္။

ng generate universal [options]

ng g universal [options]

Options

--defaults=true|false: မွန္ရင္၊ ပုံမွန္အားျဖင့္ေ႐ြးခ်ယ္စရာမ်ားအတြက္အျပန္အလွန္ဆက္သြယ္မႈဆိုင္ရာအခ်က္ျပခ်က္မ်ားကိုပိတ္ထားသည္။

--dryRun = true | false: မွန္လွ်င္၎သည္ေျပးသြားၿပီးရလဒ္မ်ားကိုေရးသားျခင္းမျပဳဘဲလုပ္ေဆာင္မႈကိုသတင္းပို႔သည္။ Default: false

Aliases: -d

--force = true | false: မွန္ကန္လွ်င္၎သည္ရွိၿပီးသားဖိုင္မ်ားကို overwrite လုပ္သည္။ Default: false

Aliases: -f

--help = true | false | json | JSON: ဒီ console မွာအကူအညီေပးေရးသတင္းကိုျပရန္အသုံးျပဳသည္။ Default: false

--interactive = true | false: မွားယြင္းပါကအျပန္အလွန္ဆက္သြယ္မႈဆိုင္ရာအခ်က္ျပခ်က္မ်ားကိုပိတ္ထားသည္။

10. Angular Libraries

Angular libraries မ်ားကို general problems မ်ားရဲ႕ solution တစ္ခုအၿဖစ္ build လုပ္ပါတယ္။ ဥပမာ presenting a unified user interface, presenting data, and allowing data entry တို႔ ၿဖစ္ၾကတယ္။ developer မ်ားက မတူညီေသာ apps မ်ားထဲမွာ ၿပန္လည္အသံုးၿပဳရန္ အဆင္ေၿပေစမည့္ particular domains မ်ားအတြက္ general solutions မ်ားကို create လုပ္ႏိုင္တယ္။ ထို solutions မ်ားကို Angular libraries မ်ားအၿဖစ္ build လုပ္ႏိုင္တယ္။ ထို libraries မ်ားကို published လုပ္ၿပီး npm packages မ်ားအၿဖစ္ shared ႏိုင္ပါတယ္။

Angular library ဆိုတာက angular project တစ္ခုၿဖစ္တယ္။ ဒါေပမယ့္ Angular app နဲ႔ မတူတာက သူက ကိုယ္တိုင္ run လို႔ မရလို႔ပဲ။ library မ်ားကို app ထဲမွာ import လုပ္ၿပီး အသံုးၿပဳပါတယ္။

Usage of Angular libraries

1. Angular libraries မ်ားက Angular's base functionalities မ်ားကို extend လုပ္ပါတယ္။ ဥပမာအားၿဖင့္ ကၽြန္ေတာ္တို႔က app တစ္ခုထဲသို႔ reactive forms မ်ားကို add လုပ္ခ်င္ပါက @angular/forms ကို အသံုးၿပဳၿပီးေတာ့ library package ထည့္ပါ။ ထို႔ေနာက္ ကၽြန္ေတာ္တို႔ရဲ႕ application code အတြင္းမွာရွိတဲ့ @angular/forms library မွ ReactiveFormsModule ကို import လုပ္ပါ။
2. Angular Material က large, general-purpose library ရဲ႕ example တစ္ခုၿဖစ္ၿပီးေတာ့ sophisticated, reusable, and adaptable UI components မ်ားကို ထုတ္ေပးပါတယ္။

Installing libraries

Libraries မ်ားကို npm packages မ်ားအၿဖစ္ published လုပ္ၿပီး Angular CLI နဲ႔ integrated လုပ္ထားပါတယ္။ reusable library code ကို application တစ္ခုထဲသို႔ ေပါင္းထည့္ရန္ အတြက္ package ကို install လုပ္ၿပီး ကၽြန္ေတာ္တို႔က သူ႕ကို သံုးမယ့္ ေနရာမွာ import လုပ္ေပးရမယ္။

Syntax

ng add **<lib\_name>**

Library typing

Library packages မ်ားက .d.ts files မ်ားထဲမွာ typings မ်ား ပါ၀င္ေစပါတယ္။ ကၽြန္ေတာ္တို႔ရဲ႕ library's package ထဲမွာ typings မ်ား မပါ၀င္ခဲ့ဘူးဆိုရင္ IDE က error ၿပမွာၿဖစ္ပါတယ္။ အဲ့လိုၿဖစ္ခဲ့မယ္ဆိုရင္ library's associated @types/ package ကို install လုပ္ရမွာ ၿဖစ္ပါတယ္။

ဥပမာ ကၽြန္ေတာ္တို႔မွာ d1 ဆိုတဲ့ library တစ္ခုရွိတယ္ဆိုပါစို႔။

npm install d1 --save

npm install @types/d1 --save-dev

Workspace တစ္ခုတြင္ install လုပ္ထားေသာ library တစ္ခုအတြက္ @types/ package တြင္ သတ္မွတ္ထားေသာ types မ်ားက library ကို သံုးေသာ project အတြက္ typescript ကို automatically added လုပ္ေပးပါတယ္။ TypeScript က default အားၿဖင့္ node\_modules/@types folder အတြင္းမွာရွိတဲ့ type မ်ားကို ရွာေဖြေပးပါတယ္။ ဒါေၾကာင့္ ကၽြန္ေတာ္တို႔က type package တစ္ခုခ်င္းစီကို add လုပ္စရာမလိုပါဘူး။

တကယ္လို႔မ်ား library မွာ @types/ ဆိုတာ မပါခဲ့ဘူးဆိုရင္ manually ရိုက္ထည့္လို႔ရပါတယ္။ ကၽြန္ေတာ္တို႔က ေအာက္ပါအတိုင္း လိုက္ၿပီး လုပ္ေဆာင္ႏိုင္ပါတယ္။

src/ folder ထဲမွာ typings.d.ts file ကို create လုပ္လိုက္ပါ။ ဒီ file က global type definition အၿဖစ္ atomically ပါ၀င္သြားမွာ ၿဖစ္ပါတယ္။

ေအာက္ပါ Code မ်ားကို src/typings.d.ts. မွာ ထည့္လို႔ရပါတယ္။

declare module 'host' {

  export interface Host {

    protocol?: string;

    hostname?: string;

    pathname?: string;

  }

  export function parse(url: string, queryString?: string): Host;

}

Library ကို သံုးတဲ့ component or file အတြင္းမွာရွိတဲ့ code ကို ေပါင္းထည့္လိုက္ပါ။

import \* as host from 'host';

const parsedUrl = host.parse('https://angular.io');

console.log(parsedUrl.hostname);

Updating libraries

ng update command ကို သံုးၿပီးေတာ့ libraries မ်ားကို update လုပ္ႏိုင္ပါတယ္။ သူက individual library versions ကို update လုပ္ေပးပါတယ္။ Angular CLI က latest published release လုပ္ထားတဲ့ library ကို စစ္ေဆးေပးပါတယ္။ တကယ္လို႔မ်ား ကၽြန္ေတာ္တို႔ ထည့္ထားတဲ့ version က latest version ထက္နိမ့္ေနမယ္ဆိုရင္ အဲ့ေနာက္ဆံုး version ကို download လုပ္ၿပီး ကၽြန္ေတာ္တို႔ရဲ႕ package.json ကို latest version နဲ႔ကိုက္ညီေအာင္ update လုပ္ေပးမွာ ၿဖစ္ပါတယ္။

Syntax

ng update **<lib\_name>**

ကၽြန္ေတာ္တို႔က Angular ကို new version တစ္ခုသို႔ update ၿပဳလုပ္တဲ့အခါ ကၽြန္ေတာ္တို႔ အသံုးၿပဳေနေသာ libraries မ်ားက current အသံုးၿပဳေနေၾကာင္း ေသခ်ာရမည္ၿဖစ္တယ္။ libraries မ်ား interdependencies ၿဖစ္ေနခဲ့မယ္ဆိုရင္ ၿဖစ္ေနေသာ libraries မ်ားကိုလည္း update လုပ္ေပးရမွာ ၿဖစ္ပါတယ္။

runtime global scope မွာ library တစ္ခု ဘယ္လိုထည့္မလဲ

app တစ္ခုအတြင္းသို႔ import မလုပ္ရေသးတဲ့ Legacy JavaScript libraries မ်ားကို runtime global scope သို႔ add လုပ္ႏိုင္ၿပီး script tag တစ္ခုအေနနဲ႔ loaded လုပ္ႏိုင္ပါတယ္။

CLI configuration file, angular.json မွာ build target ရဲ႕ "scripts" and "styles" option မ်ာကို အသံုးၿပဳၿပီးေတာ့ build လုပ္ခ်ိန္မွာ CLI ကို configure လုပ္ဖို႔ လိုအပ္ပါတယ္။

ဥပမာ Bootstrap 4 library ကို အသံုးၿပဳရန္္ npm package manager ကို အသံုးၿပဳၿပီး library နဲ႔ သူ႕ရဲ႕ dependencies ကို ဦးစြာ install လုပ္ရမယ္။

npm install bootstrap --save

Bootstrap CSS file ကို "styles" array ထဲကို ထည့္လိုက္ပါ။

"styles": [

  "node\_modules/bootstrap/dist/css/bootstrap.css",

  "src/styles.css"

],

Creating new libraries

Angular လုပ္ေဆာင္ခ်က္မ်ားကို တိုးခ်ဲ႕မယ္ဆိုရင္ ကၽြန္ေတာ္တို႔ကိုယ္တိုင္ new libraries မ်ားကို create လုပ္ၿပီး publish လုပ္လို႔ရပါတယ္။ေယဘုယ်ေၿပာရမယ္ဆိုရင္ တစ္ခုထက္ပိုတဲ့ app မွာ ၿဖစ္တဲ့ တူညီေနတဲ့ problem မ်ားကို ေၿဖရွင္းမယ္ သို႔မဟုတ္ တစ္ၿခား developer မ်ားကို solution share ခ်င္တယ္ဆိုလို႔ရွိရင္ new libraries ကို create လုပ္ၾကပါတယ္။ ကၽြန္ေတာ္တို႔မွာ library တစ္ခုအတြက္ candidate တစ္ခုရွိပါတယ္။

ဥပမာ ကၽြန္ေတာ္တို႔က button တစ္ခုကို create လုပ္လိုက္တယ္။ အဲ့ button ကိုႏွိပ္လိုက္မယ္ဆိုရင္ users ေတြကို company website ကို send လုပ္ေပးတယ္။ အဲ့လိုဟာမ်ိဳးက company builds လုပ္လိုက္တဲ့ apps အားလံုးမွာ ပါ၀င္သင့္တယ္။

Angular CLI ကို ဖြင့္ၿ႔ပီး library တစ္ခုကို create လုပ္ရန္ ေအာက္ပါ syntax ကို သံုးပါ။

ng generate library my-lib

သူက NgModule တစ္ခုအတြင္းမွာ component တစ္ခုနဲ႔ service တစ္ခုပါ၀င္တဲ့ ကၽြန္ေတာ္တို႔ရဲ႕ workspace ထဲမွာ projects/my-lib folder တစ္ခုကို create လုပ္ေပးလိမ့္မယ္။ type 'library' ရဲ႕ workspace configuration file, angular.json တုိ႔က project နဲ႔ update ၿပဳလုပ္ခံရမယ္။

"projects": {

  ...

  "my-lib": {

    "root": "projects/my-lib",

    "sourceRoot": "projects/my-lib/src",

    "projectType": "library",

    "prefix": "lib",

    "architect": {

      "build": {

        "builder": "@angular-devkit/build-ng-packagr:build",

        ...

အခု ကၽြန္ေတာ္တို႔က ေအာက္ပါ command မ်ားကို အသံုးၿပဳၿပီးေတာ့ ကၽြန္ေတာ္တို႔ရဲ႕ project ကို built, test and lint လုပ္ႏိုင္ပါတယ္။

ng build my-lib

ng test my-lib

ng lint my-lib

11. Angular 7 Architecture

Angular 7 ဆိုတာက HTML နဲ႔ TypeScript မွာ client application ကို create လုပ္ရန္အသံုးၿပဳေသာ framework တစ္ခုၿဖစ္ပါတယ္။

Angular 7 ကို TypeScript မွာ ေရးပါတယ္။

ကၽြန္ေတာ္တို႔ app မွာ import လုပ္ထားတဲ့ TypeScript libraries မ်ားအေနၿဖင့္ Angular 7 က core and optional functionality ကို implements လုပ္ပါတယ္။

NgModules မ်ားက Angular 7 application တစ္ခုရဲ႕ အေၿခခံအုတ္ၿမစ္မ်ား ၿဖစ္ၾကပါတယ္။ သူတို႔က all components မ်ားအတြက္ compilation context တစ္ခုကို ေပးပါတယ္။ Angular 7 app တစ္ခုကို NgModules ရဲ႕ set တစ္ခုအေနနဲ႔ သတ္မွတ္ၿပီး NgModules က functional sets မ်ားဆီသို႔ related code မ်ားကို collect လုပ္ေပးပါတယ္။

Angular 7 app တစ္ခုမွာ အနည္းဆံုး root module တစ္ခုရွိၿပီး အၿမဲတမ္း bootstrapping လုပ္ႏိုင္တယ္။ ပံုမွန္အားၿဖင့္ေတာ့ တစ္ၿခား feature modules မ်ားစြာရွိပါတယ္။

Components မ်ားက views မ်ားကို သတ္မွတ္ေပးပါတယ္။ သူတို႔က Angular 7 ကေန ကၽြန္ေတာ္တို႔ရဲ႕ program logic နဲ႔ data အရ ေရြးခ်ယ္ထားတဲ့ ၿပဳၿပင္ထားတဲ့ screen elements မ်ားရဲ႕ sets မ်ား ၿဖစ္ၾကပါတယ္။

Components မ်ားက services မ်ားကို အသံုးၿပဳၾကပါတယ္။ အဲ့ services မ်ားက views မ်ားနဲ႔ တိုက္ရိုက္မသက္ဆိုင္တဲ့ specific functionality ကို ေပးပါတယ္။ Service providers မ်ားကို components မ်ားဆီသို႔ dependencies မ်ားအၿဖစ္ injected ၿပဳလုပ္ပါတယ္။ ကၽြန္ေတာ္တို႔ရဲ႕ code ကို modular, reusable နဲ႔ efficient ၿဖစ္ေစပါတယ္။

Components

Components နဲ႔ services မ်ားက decorators’ မ်ားႏွင့္ simply classes မ်ားၿဖစ္ၾကပါတယ္။ သူတို႔ရဲ႕ အမ်ိဳးအစားကို မွတ္သားၿပီး လုပ္ငန္းမ်ားလုပ္ေဆာင္ရန္ Angular ကို လမ္းညြန္ေပးေသာ metadata ကို provide လုပ္ပါတယ္။

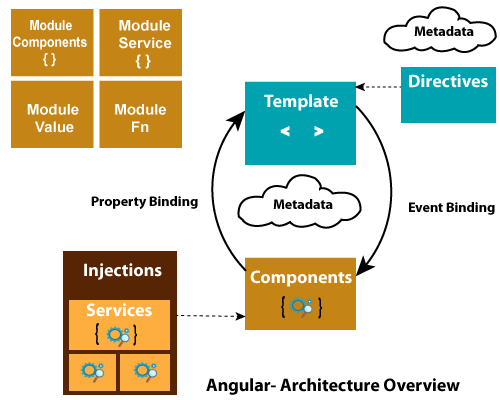
Angular application တိုင္းမွာ page DOM နဲ႔ page အဆင့္ဆင့္ကို ခ်ိတ္ဆက္ေပးတဲ့ root component လို႔ေခၚတဲ့ အနည္းဆံုး component တစ္ခု အၿမဲတမ္းပါရွိတယ္။

component တစ္ခုစီက application data နဲ႔ logic ပါ၀င္တဲ့ class တစ္ခုကို သတ္မွတ္ၿပီး target environment တစ္ခုထဲမွာ ၿပသမယ့္ view ကို သတ္မွတ္ေပးေသာ HTML template နဲ႔ ဆက္စပ္ေနတယ္။

Metadata of Component class:

component class တစ္ခုအတြက္ metadata က သူ႕ကို view တစ္ခုအၿဖစ္ သတ္မွတ္ထားတဲ့ template တစ္ခုနဲ႔ ဆက္စပ္ေနပါတယ္။ ordinary HTML ကို Angular directives နဲ႔ binding markup တို႔နဲ႔ combine လုပ္လိုက္တယ္။ သူက Angular ကို display အတြက္ rendering မလုပ္ခင္ HTML ကို modify လုပ္ဖို႔ ခြင့္ၿပဳထားပါတယ္။

service class တစ္ခုအတြက္ metadata က Angular လိုအပ္ေသာ dependency injection (DI) မွတစ္ဆင့္ components မ်ားကို ရရွိေစရန္ လုပ္ေဆာင္ေပးပါတယ္။



Modules

Angular 7 NgModules က တစ္ၿခား javascript modules မ်ားနဲ႔ မတူပါ။ Angular 7 app တိုင္းမွာ AppModule လို႔ေခၚတဲ့ root module တစ္ခု ရွိတယ္။ အဲ့ appmodule က bootstrap mechanism ကို ေပးတယ္။ အဲ့ bootstrap mechanism က application ကို launches လုပ္တယ္။

ေယဘူယအားၿဖင့္ကေတာ့ Angular 7 app တိုင္းမွာ functional modules မ်ားစြာ ပါ၀င္တယ္။

Anngular 7 Modules ရဲ႕ အခ်ိဳ႕ေသာ အေရးၾကီးတဲ့ features မ်ားကေတာ့……

Angular 7 NgModules က အၿခားေသာ JavaScript modules မ်ားကဲ့သို႔ အၿခား NgModules မ်ားမွ functionalities မ်ားကို import လုပ္တယ္။

NgModules မ်ားက သူတို႔ရဲ႕ own functionality ကို exported လုပ္ၿပီး အၿခားေသာ NgModules မ်ားကိုလည္း အသံုးၿပဳခြင့္ေပးထားပါတယ္။ ဥပမာအေနနဲ႔ ကၽြန္ေတာ္တို႔ရဲ႕ app မွာရွိတဲ့ router service ကို အသံုးၿပဳလိုပါက Router NgModule ကို import လုပ္ေပးရမယ္။

Template, Directives and Data Binding

Angular 7 မွာ template တစ္ခုကို HTML နဲ႔ Angular Markup တို႔ combine လုပ္ရန္ အသံုးၿပဳတယ္။ ၿပီးေတာ့ သူတို႔ကို display မလုပ္ခင္မွာ HTML elements မ်ားကို modify လုပ္ပါတယ္။

Template directives မ်ားက program logic ကို provide လုပ္တယ္။ binding markup က ကၽြန္ေတာ္တို႔ရဲ႕ application data နဲ႔ DOM ကို connect လုပ္ေပးပါတယ္။ binding markup က ကၽြန္ေတာ္တို႔ရဲ႕ application data နဲ႔ DOM ကို ဆက္သြယ္ေပးပါတယ္။

data binding အမ်ိဳးအစား ႏွစ္မ်ိဳးကေတာ့……

**Event Binding:** Event binding ကို အသံုးၿပဳၿပီးေတာ့ event မ်ားကို ကၽြန္ေတာ္တို႔ရဲ႕ app နဲ႔ ခ်ိတ္ဆက္ရန္ ၿပီးေတာ့ ကၽြန္ေတာ္တို႔ရဲ႕ application data ကို update လုပ္ၿခင္းအားၿဖင့္ target environment ထဲမွာရွိတဲ့ user response ကို return ၿပန္တယ္။

**Property Binding:** Property binding က component class မွ data မ်ားကို pass လုပ္ရန္ အသံုးၿပဳၿပီးေတာ့ values မ်ားကို interpolate (ၾကားၿဖတ္ေၿပာဆိုသည္) လုပ္ရန္ ကၽြန္ေတာ္တို႔ကို လြယ္ကူေခ်ာေမြ႕ေစပါတယ္။ HTML မွ application data သို႔ သြားတဲ့ values မ်ားကိုလည္း တြက္ခ်က္ေပးပါတယ္။

Services and Dependency Injection

Angular 7 မွာ developers မ်ားက data သုိ႔မဟုတ္ logic အတြက္ specific view တစ္ခုနဲ႔ မသက္ဆိုင္တဲ့ service class တစ္ခုကို ဖန္တီးၿပီး developers မ်ားက components မ်ားကို share across လုပ္ခ်င္ၾကတယ္။

Dependency Injection (DI) က ကၽြန္ေတာ္တို႔ရဲ႕ component classes မ်ားကို ပိုမိုထိေရာက္ေစရန္ အသံုးၿပဳပါတယ္။ DI က fetch data from the server, validate user input, or log directly to the console စသည္တို႔ကို မလုပ္ေဆာင္ပါ။it simply renders such tasks to services.

Routing

Angular 7 တြင္ router သည္ NgModule တစ္ခုၿဖစ္တယ္။ NgModule က developers မ်ားအား မတူညီေသာ application states အၾကား navigation path တစ္ခုကို သတ္မွတ္ရန္နဲ႔ သူတို႔ရဲ႕ app အတြင္းမွာ view hierarchies လုပ္ေဆာင္ရန္ ေဆာင္ရြက္ေပးပါတယ္။

routing က browser ရဲ႕ navigation နဲ႔ လုပ္ေဆာင္ခ်က္ၿခင္း တူညီပါတယ္။

address bar မွာ url ကို ရိုက္ထည့္လိုက္ပါ။ browser က အဲ့သက္ဆိုင္ရာ page ကို navigate လုပ္လိမ့္မယ္။

browser ရဲ႕ back သို႔ forward button ကို ႏွိပ္လိုက္မယ္ဆိုရင္ browser က သူ႕ရဲ႕ history အတိုင္း backwork သို႔ forward navigate လုပ္ေပးလိမ့္မယ္။