Development Environment Setup

* Install NodeJS
* Install NPM
* Install Angular CLI globally

ng new my-app

* Create a new application

ng new my-app

* Go to the app

Cd my-app

Ng serve --open

The ng serve command launches the server, watches your files, and rebuilds the app as you make changes to those files.

Note you are doing all your working in Angular CLI(Power Shell)

Angular Components and Files

1. Components

Root component => app-root

Each component will have css file, html file, component file, spec file, module file.

1. Readme.md

Commands for ng

1. Src folder

App lives in src folder. Src folder has your angular component, templates,styles,images

1. App component

app/app.componet.css – css

app/app.component.html – html template

app/app.component.ts – template file

app/app.component.spec.ts – unit test file

app/app.module.ts – module tells how to assemble the application

1. Assets/\* - images
2. Environmens/\* - has one file for each of your environment each has simple configuration variable to use in your application. The files are replaced on the fly when you build the application
3. Favi con.ico – bookmark bar image
4. Index.html – the main html page that serves when someone visits your site Most of the time you'll never need to edit it. The CLI automatically adds all js and css files when building your app so you never need to add any <script> or <link> tags here manually.
5. Main.ts – main entry point for your application. Compiles the application with the JIT compiler and bootstraps the application root module(app module) to run in browser.
6. Polyfills.ts - Different browsers have different levels of support of the web standards. Polyfills help normalize those differences.
7. Style.css – global styles go here this is central styles
8. Test.ts – main entry point for unit tests.
9. Tsconfig.app.json – typescript compiler configuration for the angular app
10. Tsconfig.spec.json – typescript compiler configuration for the unit tests
11. Root folder
12. E2e/ - end to end tests
13. Node-modules/ - node.js creates this folder and put all third party modules listed in package.json
14. Angular-cli.json - Configuration for Angular CLI. In this file you can set several defaults and also configure what files are included when your project is built.
15. Editorconfig - Simple configuration for your editor to make sure everyone that uses your project has the same basic configuration
16. .gitignore-ignore these files
17. Karma.conf.js – unit test configuration for the karma test runner used when running ng test
18. Package.json – third party packages that u used In the project. It is just the list.
19. Protractor.conf.js – end to end test configuration for protractor used when running ng e2e
20. Tsconfig.json – typescript compiler configuration for your IDE to pick up and give helpful tooling.
21. Tslint.json   
    Linting configuration for [TSLint](https://palantir.github.io/tslint/) together with [Codelyzer](http://codelyzer.com/), used when running ng lint. Linting helps keep your code style consistent.

Generating new component

Ng generate component heroes

ng generate component heroes

The CLI creates a new folder, src/app/heroes/ and generates the three files of the HeroesComponent.

What is Angular

Used to build single page applications(SPA)

Front end client side javascript framework

Why Angular

Quick development

Code Generation(CLI)

Code organization and productivity

Dynamic content (if variable, etc)

Cross platform (windows,mac,chrome, firefox)

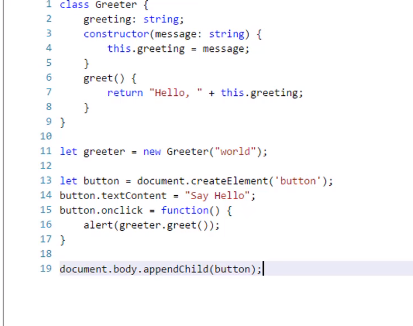
Unit testing ready

TypeScript

Superset of javacript with added features

Optical static typing

Class based OOP



Component

Sections of UI can be broken into encapsulated components

An angular application is tree of angular components

Decroators allow us to mark a class as angular component & provide metadata that determines the component should be processed,instantiated and used at runtime.



ngOnInit() in OnInit => life cycle hook event this will run when component initializes

Services

Classes that send data and functionality across components

Don’t repeat , ideal place for ajax calls



1. Create a service

Ng generate service services/data

My service name is dataservice

1. Import Add to app.module inside the provider
2. Use the service in the component that you are going to use

Import services

Call the dependency injection

Constructor(private dataservice:DataService) {

}

1. Go to the service ts file

Write service methods

(use http module, react module map)

Method() return {This.http.get(‘https://’);)

1. Call the service method in component

Dataservice.dataposts.subscribe(posts)

Map to interface posts

1. Html page -> map the output let posts

**Routing**

1. Import the router module and routes from app.module
2. Create our route variable array

Eg: homepage, postspage

1. Import routmodule and add them routes contstant

So routing defined http/ocalhost/about or http:/localhost/post

1. Use them <route-applet> in app.componet.html