**Angular ng commands:-**

1. ng generate class models/Student
2. ng generate service services/api --skipTests=true
3. ng generate interceptor helper/interceptor
4. ng generate module moduleName
5. ng generate component componentName

**BUILDING BLOCKS OF ANGULAR:-**

1. Module
2. Component
3. Templates
4. Metadata
5. Data Binding
6. Directives
7. Service
8. Dependency Injection
9. Decorators
10. Pipes
11. Routing

**DataTypes:-**

1. String
2. Number
3. Boolean
4. Void
5. Null
6. Undefined
7. Any

**Module-**

app.module.ts is the file which contains all components, directives, pipe, services related to that module.

All above mentioned files should needs to be imported in app.module.ts

-**declarations**: The components in this module.

**-imports**: The modules that are required by the current module. It enables an Angular module to use functionality that was defined in another Angular module.

**-exports**:- It enables an Angular module to expose some of its components/directives/pipes to the other modules in the applications. Without it, the components/directives/pipes defined in a module could only be used in that module.

-**providers**:- The service providers, if any.

**-bootstrap**: The root component that Angular creates and inserts into the index.html host web page.

**Rotuing-Module:-**

App-routing.module.ts is to declare list of routings required and their respective components for those routings.

**Eager Loading:-**

* all the feature modules will be loaded before the application starts. Hence the subsequent request to the application will be faster.
* Better useful for small size application

**Lazy Loading:-**

* module will be loaded on demand and hence application start will be faster.
* For large size application
* It load it using l**oadChildren** property in route configuration

**Component Life Cycle:-**

1. Constructor
2. ngOnChanges

It is executed whenever the component's input properties change.

1. ngOnInit

It is executed after the component's constructor method and is a good place to perform initial setup for the component.

1. ngDoCheck

It is executed during every change detection cycle and is a good place to perform custom change detection.

1. ngAfterContentInit

It is executed after the component's content has been initialized and is a good place to perform additional setup for the component's content.

1. ngAfterContentChecked

It is executed after the component's content has been checked and is a good place to perform additional operations based on the component's content.

1. ngAfterViewInit

It is executed after the component's view has been initialized and is a good place to perform additional setup for the component's view.

1. ngAfterViewChecked

It is executed after the component's view has been checked and is a good place to perform additional operations based on the component's view.

1. ngOnDestroy

It is executed just before the component is destroyed and is a good place to perform cleanup operations for the component.

Angular Forms:-

1. Reactive Form

Ref [Angular Basics: What are Reactive Forms and When to Use Them? (telerik.com)](https://www.telerik.com/blogs/angular-basics-what-reactive-forms-when-use-them)

1. Template Driven Form

**Pipes(Filter):-**

Format the data in the template It is also called filter

1. Currency Pipe
2. Date Pipe
3. Uppercase Pipe
4. Lowercase Pipe
5. Decimal Pipe
6. JSON Pipe

**Template Reference Variable**:-

We can use template reference variable to pass the value instead of $event.

For example,

<h2>Template Reference Variable</h2>

<input #el (input)="handleInput1(el)">

<p>You have entered {{val}}</p>

val="";

handleInput1(element) {

this.val=element.value;

}

**Decorators:-**

There are 4 types of decorators

1. Class Decorators

@NgModule, @Component, @Injectable, @Directive, @Pipe

1. Property decorators

@Input, @Output, @HostBinding, @ContentChild, @ContentChildren, @ViewChild, @ViewChildren

1. Method decorators

@HostListener

1. Parameter decorators

@Inject, @Self, @SkipSelf, @Optional, @Host

**Change Detection:-**

Angular change detection is a built-in framework feature that ensures the automatic synchronization between the data of a component and its HTML template view

**ViewChild and ViewChildren:-**

Ref: <https://www.youtube.com/watch?v=NJFIEp2RDBM&ab_channel=NishaSingla>

**Authentication and Authorization:-**

Authentication is such as login. Authorization is nothing but which resources can I use/access that.

**Angular Boot Strapping:-**

1. Executes main.ts file

2. Main.ts file booting Appmodule file using the code of

platformBrowserDynamic().bootstrapModule(AppModule)

3. Appmodule bootstrapping the AppComponent file using the code of

bootstrap: [AppComponent]

4. Now the index.html file will be triggered through AppComponent file.AppComponent.ts having the selector tag of app-root, this root tag is defined in index.html file.

<app-root></app-root>

Ref:- <https://www.tektutorialshub.com/angular/angular-bootstrapping-application/>

**Services:-**

1. create an export class and you need to decorate that class with @Injectable decorator
2. import the Injectable decorator from the angular core library.
3. services are used to share a single piece of code across multiple components.
4. These services are used to hold business logic

import { Injectable } from '@angular/core';

@Injectable()

export class DebugService {

constructor() { }

// Method and Properties

}

**Another Way**

@Injectable({

providedIn: 'root'

})

**Angular Components Communication:-**

1. Parent to Child Communication:-
2. Child to Parent Communication:-
3. Communication through service

Ref <https://reactgo.com/angular-component-communication/>

https://www.knowledgehut.com/tutorials/angular/services-in-angular

Ref: https://www.thirdrocktechkno.com/blog/angular-component-communication/

**How to delete Angular Components:-**

1.Remove the import line reference from Angular app.module.ts file.

2.Remove the component declaration from @NgModule declaration array in app.module.ts file

3.And then manually delete the component folder from Angular project.

4.Finally Delete all the references of component manually from the Angular project.

**Async Pipe:-**

It subscribes to the observable and emits the last value emitted.

When a new value is emitted, it marks the component to be checked for the changes.

The async pipe automatically unsubscribes when the component is destroyed to avoid potential memory

**STATE MANAGEMENT WITH NGRX:-**

The NgRx state management process is a combination of five key elements:

**Store:** The application state is maintained in the store. It is immutable.

**Selectors:** Angular components can subscribe to the store and get updates through selectors.

**Reducers:** Reducers are responsible for handling the transition from one state to another.

**Actions:** Actions modify the state of the store by using reducers.

**Effects:** Effects are the results of actions. Also, they can be used to listen for particular action types and run when the action happens.

Eg:- https://www.syncfusion.com/blogs/post/angular-state-management-ngrx

JD for an MNC:-

Strong technical understanding of Angular framework with regards to subscriptions, promises, pipes, directives, observables, b

npm config set legacy-peer-deps true