

=====  
Angular  
=====

- => Angular is a client side framework
- => Angular framework developed by Google company
- => Angular developed using TypeScript
- => Angular is mainly used for SPA (single page app)
- => Angular supports multiple browsers
- => Angular is free & open source

Note: Angular JS & Angular framework both are not same.

- => Angular JS developed using Java Script. (Angular 1.x)
- => Google identified some performance issues in Angular JS 1.x version then they re-developed angular is Typescript which is called as Angular framework

Note: From 2.x version onwards it is called as Angular Framework.

=====  
Angular Building Blocks  
=====

- 1) Components
- 2) Metadata
- 3) Template
- 4) Data Binding
- 5) Modules
- 6) Services
- 7) Dependency Injection
- 8) Directives
- 9) Pipes

- 
- => Template is a view page (html file)
  - => Component represents small portion in web page

Ex: header component  
    menu component  
    body component  
    footer component etc...

Note: Every component will have its own template

- => Component & template relation will be represented using Metadata
- => Data binding is the process of sending data from component to template and vice versa

Note: Angular supports two way data binding

component <-----> template

- => Service is a typescript class which contains business logic.
- => Directives are used to manipulate DOM elements in template.

Ex: if - else, loops etc...

=> Pipes are used to transform the data in template.

ex: lower case to upper case & INR to USD etc...

=> Dependency Injection means injecting one class obj into another class obj

Ex: Inject service obj into component

=> Modules represents collection of components + services + directives .....

Note: Modules are used for logical grouping

=====  
Angular Setup  
=====

Step-1 : Download and Install Node

URL : <https://nodejs.org/en/>

Note: After installation, verify node version

\$ node -v

Step-2 : Install Type Script

\$ npm install -g typescript

\$ tsc -v

Step-3 : Install Angular CLI

\$ npm install @angular/cli -g  
\$ ng v

Step-4 : Download and install VS Code IDE

Step-5: Create Angular Application

\$ ng new app1  
\$ cd app1  
\$ ng serve --open

=====  
=> In angular application by default "app-component" will be created. It is called as Parent Component.

Note: App-Component is the entry point for angular application.

=> Every component will have a selector, which is used to invoke the component.

### app-component selector name is 'app-root'

=> app-component will be accessed using its selector in 'index.html' page

Ex: <app-root></app-roo>

=> index.html page is called as welcome page in angular application

=> When we run angular application, index.html page will be loaded and it will invoke app-component hence we will get response from app.component.html page.

Note: To apply styles for app-component template we have app-component.css file.

=====

=> We can create component using below command

```
$ ng generate component <component-name>
```

or

```
$ ng g c <component-name>
```

=> Every component will have its own

- 1) Component class (Ts file)
- 2) Template (html file)
- 3) CSS file

=====

## Data Bindings

=====

=> It is used to establish relation between "component" and template

=> In Angular we can perform data binding in 4 ways

- 1) Interpolation
- 2) Property Binding
- 3) Event Binding
- 4) Two Way Data Binding (Property binding + Event Binding)

=====

## Interpolation

=====

=> It is used to access component variable/property in template

Syntax : {{propertyName}}

```
// java variable
String msg = "Welcome To Ashok IT";
```

```
// type script variable
msg:string = "Welcome To Ashok IT";
```

=====

## Event binding

=====

=> It is used to pass notifications from template to component

Ex: Button click

-----

```
export class AppComponent {
  title = 'app3';
```

```
  msg:string = "Welcome to Ashok IT..!!";
```

```

displayMsg1(){
    this.msg = "Welcome to Angular..!!";
}

displayMsg2(){
    this.msg = "Welcome to Fullstack Zone..!!";
}
}

```

```

-----
<h1>{{msg}}</h1>

```

```

<input type="button" value="Msg-1"
      (click)="displayMsg1()"/>

```

```

<input type="button" value="Msg-2"
      (click)="displayMsg2()" />

```

```

=====
Two way data bining
=====

```

=> It is combination of both property binding and event binding

=> When we change the value of the property in component then automatically it will be updated in template

=> When we change the value in template then automatically it will be updated in component property

=> To work with two-way-data-binding we will use "ngModel" directive

=> Two way data binding is applicable only for <input/> and <select/> tags

Note: "FormsModule" should be imported to work with two way data binding.

```

Enter Name : <input type="text" [(ngModel)]="fname"/> <br/> <br/>

```

```

Good Morning, {{fname}}

```

```

-----app-component-----
import { Component } from '@angular/core';
import { CommonModule } from '@angular/common';
import { RouterOutlet } from '@angular/router';
import { FormsModule } from '@angular/forms';

@Component({
  selector: 'app-root',
  standalone: true,
  imports: [CommonModule, RouterOutlet, FormsModule],
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
  title = 'app3';

  msg:string = "Welcome to Ashok IT..!!";
  fname:string = "Ashok";
  myColor:string="red";

  displayMsg1(){
    this.msg = "Welcome to Angular..!!";
  }
}

```

```

displayMsg2(){
  this.msg = "Welcome to Fullstack Zone..!!";
}
}

```

-----app.component.html-----

```

<h1>{{msg}}</h1>

<input type="button" value="Msg-1"
  (click)="displayMsg1()"/>

<input type="button" value="Msg-2"
  (click)="displayMsg2()" />

<hr/>

<div [style.color]="myColor">
  <p>This is my text</p>
</div>

<hr/>
Enter Name : <input type="text" [(ngModel)]="fname"/> <br/> <br/>

Good Morning, {{fname}}

```

-----

```

=====
Customer Application Development
=====

```

=> Develop angular application to manager customers details

a) Save Customer Data

b) Display All Customers in Table format

```

=====
Services
=====

```

=> Services are used to write business logic

=> We can create a service class using below command

```
$ ng generate service customer
```

=> Service object we can inject into Component using Dependency Injection.

=> Once we inject service obj into component obj then component can access service class functions.

-----

1) What is Angular

2) Angular Architecture

3) Angular Building Blocks

- 4) Angular Project Creation
- 5) Angular Project Execution
- 6) Execution Flow Of Angular App
- 7) Components in Angular
- 8) Templates in Angular
- 9) Data binding in Angular
- 10) Services
- 11) Dependency Injection

=====  
Angular + Spring Boot Integration  
=====

- 1) Create Spring Boot REST API Project
- 2) Create RestController with required methods
- 3) Run springboot rest api and test backend functionality
- 4) Create Angular Application
- 5) Run angular application
- 6) Declare 'msg' variable in component and access in template using interpolation
- 7) Import HttpClientModule in app.component.ts file
- 8) Inject HttpClient in app.component.ts file
- 9) Write functions to make backend calls using httpClient
- 10) Write Presentation logic in template.

=====

**\*Requirement:\***

Develop Bookstore application with fullstack architecture.

- a) Save Book
- b) Get Books

Backend : SpringBoot REST API

Frontend : Angular

Database : H2

```
=====
Angular Routing
=====
```

=> It is used to establish navigation for multiple components

-----app.routes.ts file-----

```
import { Routes } from '@angular/router';
import { HomeComponent } from './home/home.component';
import { ServicesComponent } from './services/services.component';
import { ContactusComponent } from './contactus/contactus.component';
import { SchedulesComponent } from './schedules/schedules.component';

export const routes: Routes = [

  {path: 'home', component: HomeComponent},

  {path: 'services', component:ServicesComponent},

  {path: 'schedules', component:SchedulesComponent},

  {path: 'contact', component:ContactusComponent},

  {path: '', redirectTo:'/home', pathMatch:'full'}

];
```

-----app.component.html file-----

```
<div class="container">
  <h1>{{msg}}</h1>
<hr/>

<a href="home" class="btn btn-primary">Home</a> &nbsp;
<a href="services" class="btn btn-primary">Services</a> &nbsp;
<a href="schedules" class="btn btn-primary" > Training Schedules</a> &nbsp;
<a href="contact" class="btn btn-primary">Contact Us</a> &nbsp;

<hr/>

<router-outlet></router-outlet>
</div>
```

-----