Welcome to Ashok IT........

My name is Ashok (having 8+ yrs experience as a java developer, currently i am working as Technical Lead in Usa based banking company (product based).

I am having 5+ years of experience in teaching.....

-------------------------------------------------------------------------

Today : First Session for Angular -11 with TypeScript

-------------------------------------------------------------------------

Pre-Requesities

---------------

HTML, CSS and Basics of Java Script

Course Content

--------------

Part-1

---------

What is a web application

Multi Page web application

Single Page Web Application (SPA)

Angular Introduction

Angular JS vs Angular Framework

Angular Framework Version History

Advantages of Angular

Angular Architecture

Part-2

-------

TypeScript Introduction

Advantages of TypeScript

Java Script vs TypeScript

Environment Setup to work TypeScript

Variables

DataTypes

OOPS

Class

Object

Constructor

Inheritence

Access Modifiers

Interfaces (Contracts)

Enumerations

Modules

Examples Using TypeScript

Part-3

------

Revise Angular Introduction

Revise Angular Architecture

Environment Setup (NPM, Angular CLI, VS Code IDE)

First Application Creation Using Angular

Understanding Angular Application Folder Structures

Live Server to run Angular apps

Components in Angular

Modules

Data Bindings

Login App using Angular

Registration App using Angular

Directives

Pipes

Form Based Applications using Angular

- Template driven forms

- Reactive Forms

Form validations

Routings

Part-4

------

Http Client

2 or 3 REST APIS using Spring Boot

Consuming Backend Rest Apis using Angular

Spring Boot with Angular Integration

Interview Questions

Duration : 30 classes

Timings: 11:30 AM IST - 1:00 PM IST

Course FEE : FREE

Trainer: Mr. Ashok

---------------------------------------------------------------------------------

Lot of demand is there for Full Stack Developers

-------------------------------------------------------------------------------

Backend Development : Spring Boot & Microservices

Frontend Development : Angular or React

---------------------------------------------------------------------------------

Course Name : Angular 11 with TypeScript

Start Date : 19-Apr-2021

Class Timings : 11:30 AM - 1:00 PM IST (Mon-Sat)

Duration : 30 classes

Course FEE : 0 INR (FREE COURSE)

Trainer : Mr. Ashok (8+ yrs experience, working professional)

-----------------------------------------------------------------------

Pre-Requisites : HTML, CSS and Java Script (Basics)

-----------------------------------------------------------------------

19-Apr-2021 : Class Recording: https://www.youtube.com/watch?v=VgWl5QzJwHo

-----------------------------------------------------------------------------

Today's session : Full Stack Development, MPA and SPA

-------------------------------------------------------------------------------

-> Now a days in the industry companies are following Microservices architecture to develop the applications.

-> Microservices is an Architectural Design Pattern which is used to develop our applications with loosely coupling and easy maintenence.

-> As part of this Microservices architecture we will develop business logics as Rest Apis.

-> REST APIS are for B 2 B communication (Business to Business Communication).

---------------------------------------------------------------------------------

Fullstack Development :Backend Development (REST apis) + Frontend Development (UI)

---------------------------------------------------------------------------------

-> As a fullstack developer we are going to develop web applications.

What is a web application?

---------------------------

-> The application which runs in server is called as web application.

-> Multiple users can access web applications at a time by using internet.

Ex: Facebook, Gmail, IRCTC etc.....

-> We can see 2 types of web applications

1) Multi page web applications

2) Single page web applications

-> In Multi page web applications for every request new web page will be loaded at client side. Here everytime entire page will be reloaded.

-> In Single page web application web page will be loaded only for first request. From second request onwards only content will be updated without reloading the entire page.

19-Apr (Session-1) - Pre Requisites for Angular-11 and Course Content

20-Apr (Session-2) - Multi Page Web App vs Single Page Web app

Last two sessions videos available in our YouTube Channel

------------------------------------------------------------------------

YouTube Channel URL : http://www.youtube.com/c/AshokIT

------------------------------------------------------------------------

What is Fullstack Development?

--------------------------------

-> As part of fullstack development we should develop backend Rest Apis and Front End user Interface.

-> To develop backend rest apis we will use "Spring Boot & Microservices"

-> To develop frontend user interface we will use below technologies

HTML

CSS

Java Script

BootStrap

Angular or React JS

-> The resouce who can develop both fronend and backend logics is called as Fullstack developer.

-> Fullstack developers are having lot of demand in the market and companies are providing high packages for full stack developers.

What is Web Application?

--------------------------

-> The application which runs in the server is called as Web Application.

-> Multiple users can access web applications at a time.

Ex : Gmail, Facebook, IRCTC, Net Banking etc...

-> Now a days web applications are divided into 2 types

1) Multi Page Web applications (MPA)

2) Single Page Web applications (SPA)

-> In Multi Page Web Application for every request new web page will be loaded at the client side.

-> In Single Page web application only one web page will be available for every request just content will be updated without reloading the entire page.

Ex : Gmail, angular.io etc....

Advantages of SPA

------------------

-> Development will be easy

-> Maintenence will be simple

-> Good Performance

Note: In single page application common content will be loaded only one time.

---------------------------------------------------------------------------------

Angular Introduction

---------------------------------------------------------------------------------

-> Angular is a client side framework

-> Angular is used to create Web Applications

-> Angular is mainly used for Single Page Applications Development.

-> Angular supports all platforms (Desktop and Mobile)

-> Angular is free & open source

-> Angular is having cross-browser compatability

-> Angular developed by Google.

---------------------------------------------------------------------------------

Angular JS and Angular Framework

---------------------------------------------------------------------------------

-> The intial version of Angular is called as Angular JS (Angular 1.x verions) which is developed by using Java Script.

-> After few days Google realized the problems with Angular JS then they started developing one new framework to develop web applications thats where Angular Famework came into Market. This Angular framework is developed by using Type Script.

-> From Version 2.0 onwards it is called as Angular Framework

-> Angular 2.0 is not enhancement for Angular 1.0 because in 2.0 version Google Developers completley re-written Angular framework by using TypeScript.

-> TypeScript developed by Microsoft Company.

-> Angular 2+ versions are extensions for Angular 2.0 version

-> Angular 1.x versions are called as Angular JS

-> Angular 2.0 and later versions are called as Angular Framework versions

-> The current version of Angular is 11 (Latest version released in Nov-2020)

Angular Architecture

--------------------

1) Components

2) Meta Data

3) Services

4) Templates

5) Directives

6) Pipes

Today : 04-Session on Angular - 11

-----------------------------------------------------------------------

-> In last session we discussed about Angular Introduction

-----------------------------------------------------------------------------------------

-> Angular is a client side framework

-> Angular is used to create Web Applications

-> Angular is mainley used to develop Single Page Applications

-> Angular framework is free and open source

-> Angular framework developed by Google.

-> There are 2 flavours of Angular

1) Angular JS (Angular 1.x)

2) Angular (Angular 2+ versions)

-> Angular JS is developed using Java Script

-> Angular framework is developed by using Type Script

Note: Angular framework is completley re-written by google in Angular 2.0 version

-> Angular is not extension for Angular JS (Angular is completley different from Angular JS)

Angular Architecture

-----------------------

1) Component

2) Meta Data

3) Service

4) Data Bindings

5) Template

6) Directives

7) Pipes

-> Component is a TypeScript class which acts as Request Handler

-> Meta Data Describes Component and Template mapping

-> Service is a TypeScript Class which is used to write business logics

-> Data Bindings are used for tranferring the data from component to template and from template to component.

-> Template is a HTML file which contains presentation logic

-> Directives are used for DOM manipulations

-> Pipes are used to transforming the data from one format to another format.

Last session : Angular Architecture

------------------------------------------------------------------

1) Component

2) Service

3) Meta Data

4) Data Bindings

5) Template

6) Directives

7) Pipes

-------------------------------------------------------------------------

-> Angular is a client side framework

-> Angular is used to develop single page applications

Ex : gmail, angular.io

-> Angular is free & open source

-> Angular is Developed By Google

-> The current version of Angular is Angular-11

-> Angular framework is developed by using TypeScript

-> Angular JS is developed by using Java Script

-> Angular framework is not extension for Angular JS

-------------------------------------------------------------------------

Today's session : TypeScript

-------------------------------------------------------------------------

-> TypeScript is a programming language which is developed by using Java Script.

-> TypeScript is superset of java scirpt which adds data types + classes + Interfaces etc...

-> TypeScript can be used for both client side programming and server side programming.

-> TypeScript developed by Microsoft company in 2012.

------------------------------------------------------------------------------

Features of TypeScript

------------------------------------------------------------------------------

-> TypeScript is general purpose programming language

-> TypeScript built on top of java script

-> TypeScript supports all features of java script and it is providing some additional features (data types, classes, objects, interfaces, modules etcc).

Note: Java Script can be executed in browser directley where as Browser can't execute TypeScript Directley.

-> Typescript files should be converted into java script.

-> The process of converting TypeScript file into JavaScript is called as "Transpilation".

file.ts -----> tsc ----> file.js ---> browser ---> output

TypeScript Versions

-------------------

TypeScript 0.8 : 2012 (Intial Version)

.....

TypeScript 4.x : 2020

Static Typing & Dynamic Typing

------------------------------

-> Whenver we can fix a data type for the variable while declaration of the variable and we can't change its datatype throughout the program then it is called as "Static Typing"

int age = 20; //valid

age = "20 Years" ; //in-valid

Ex : C, C++, Java, C#.Net are the examples for Static Typing languages

-> If we can't fix data type for the variable while declaration and the data type will be automatically taken by the runtime engine at the time of program execution then it is called as "Dynamic Typing".

var age;

age = 20 ; //valid

age = "20 Years" ; //valid

Ex : JavaScript and Python are examples for Dynamic Typing Languages

-> TypeScript supports "Optional Static Typing". It supports both 'Static Typing' and 'Dynamic Typing'.

-> TypeScript maintains Type Safety. If we specify data type while declaring the variable and if we assign wrong type of value into variable then compiler shows error.

TypeScript Installation

-----------------------

1) Install Node JS (nodejs.org)

2) Install TypeScript (in command prompt -> npm install -g typescript)

TypeScript First Example

-------------------------

-> Create TypeScript file with .ts extension and add below code

var s:string = "Hello World"; // String s = "Hello World";

console.log(s); //System.out.println(s);

-> Open Cmd and compile typescript file

tsc <filename>.ts

Note: Tsc compiler will covert ts file into js file (Transpilation)

-> Run js file using node with below command

node <filename>.js

-> The above program should print Hello World in console

Last session (26-Apr-21) : First Example Using TypeScript

-----------------------------------------------------------------------------------

-> TypeScript is a general purpose programming language.

-> TypeScript is superset of java script

-> We can use TypeScript at client side development and at server side development also

-> Java Script doesn't support for data types where as TypeScript support for Data types.

-> We can't execute Typescript file directly in browser.

-> TypeScript file should be converted into Java Script file then we can execute in browser directley.

-> The process of converting TypeScript file into java script file is called as Transpilation.

-> TypeScript provided compiler (TSC) it is used to perform "Transpilation".

-> TypeScript supports for OOPS also.

TypeScript Installation

-----------------------

1) Download and Install Node Js

2) Install TypeScript

3) Download VS Code IDE (https://code.visualstudio.com/download)

Last Session : First Application Using TypeScript

-------------------------------------------------------------------------

-> TypeScript is general purpose programming language.

-> TypeScript is superset of java script.

-> TypeScript programs can't be executed in browser directley

-> TypeScript program should be converted into java script using TypeScript Compilor.

-> The process of converting TS file into JS file is called as Transpilation.

----------------------------------------------------------------------------------

-> JS file we can execute in command prompt using node

-> JS file we can execute using HTML also

----------------------------------------------------------------------------------

-> In Realtime we will develop our front end applications using VS Code IDE

-> Visual Studio Code IDE given by Microsoft (Free of cost)

-----------------------------------------------------------------------------------

Variables In TypeScript

----------------------------------------------------------------------------------

-> Variables are used to store the value

-> Variable is a named memory (Every variable will have a name and will occupy some memory)

Syntax : var variableName:dataType = value;

Ex : var age:int = 20; //variable in typescript

int age = 20; //variable in java

DataTypes

----------

-> Data Type specifies what type of value that can be stored into variable.

TypeScript DataTypes

----------------------

1) number

2) string

3) boolean

4) any

number : For all types of numbers

-> Integer and Floating numbers can be represented by using number datatype.

Ex : 100, 150.56

string : Collection of characters in double quotes or in single quotes

Ex : "hello" & 'hi'

boolean : true or false

any : Any type of value we can store

--------------------------------------------------------------------------------

var pname:string = "Ashok";

var age:number = 26;

var gender:string = "Male";

var isEmployeed:boolean = true;

console.log(`Name = ${pname}, Age = ${age}, Gender = ${gender}, IsEmployee = ${isEmployeed} `);

----------------------------------------------------------------------------------------------

Last session: variables & data types in TypeScript

------------------------------------------------------------------------

-> DataTypes are used to specify type of data that we can store into variable.

-> In TypeScript we have below 4 data types

1) number

2) string

3) boolean

4) any

-> Variable is used to store the value

Syantax : var variableName:dataType = value;

var age:number = 20;

var name:string = "ram";

var isEmp:boolean = false;

What is var keyword?

--------------------

-> var keyword is used to declare the variables.

-> var keyword is functional scope

--------------------------------vardemo.ts------------------------------

var index = 0;

for(var index = 0; index <=5; index++){

Console.log("Index Value Inside For Loop : " + index);//12345

}

console.log("Index Value Outside Loop :: "+index); //Expected Output : 0

-----------------------------------------------------------------------

-> In the example index value will be printed as 0 , 1, 2, 3, 4 and 5

-> Outside the loop index value will be printed as 6 (It is modified)

-> "var" keyword is functional scope

-----------------------------------------vardemo.ts-------------------------------------------

var petName:string = "dog";

function setPetName(){

var petName:string = "cat";

console.log("Inside Function :: "+ petName);

}

setPetName();

console.log("Outside Function :: "+petName);

var index = 0;

for(var index = 0; index <=5 ; index++){

console.log("Index Value Inside For Loop:: " + index);

}

console.log("Inde value Outside For Loop :: " + index);

----------------------------------------------------------------------------------------------

-> To overcome drawback of var keyword they introduced 'let' keyword

-> let keyword is block scope

------------------------------------------------letdemo.ts-----------------------------------

var index = 0;

for(let index = 0; index<=5; index++){

console.log("Index Value Inside For Loop:: "+ index);

}

console.log("Index Value Outside For Loop :: "+ index); // It reamins 0

console.log("=======================================");

function display(){

let msg:string = "Welcome To Ashok IT...";

{

let msg:string = "Welcome To Angular";

console.log("Inside Block Msg Value :: " + msg);

}

console.log("Outside Block Msg Value :: " + msg); // It remains Welcome To Ashok IT

}

display();

---------------------------------------------------------------------------------------------

-> 'var' keyword is function scope

-> 'let' keyword is block scope

----------------------------------------------------------------------------------------------

What is 'const' keyword?

---------------------------------------------------------------------------------------------

-> 'const' keyword is used to declare constant variables in Java Script.

-> For Constant variables initialization is mandatory.

-> We can't modify the value for 'const' variables in java script..

--------------------------------constdemo.ts--------------------------------------------------

const pname:string = "Ashok IT";

console.log(pname);

----------------------------------------------------------------------------------------------

Last session : var, let and const keywords in Java Script

-------------------------------------------------------------------------

-> var, let , const keywords are used to declare variables in JS and TS.

-> var keyword is functional scope

-> let keword is block scope

-> const keyword is used to declare constant variables. Intialization is mandatory for constant varibles and we can't modify the value once they are intialized.

-------------------------------------------------------------------------

-> What is Variable ?

-> What is Data Type ?

-> How many Data Types Available in TypeScript ?

-> How to declare variable in TypeScript ?

-> var vs let vs const ?

var variableName:dataType = value;

let variableName:dataType = value;

const variableName:dataType = value;

------------------------------------------------------------------------

Arrays in Type Script

-------------------------------------------------------------------------

-> Arrays are used to store group of values

-> In TypeScript Arrays size is not fixed. We can store as many values as we want. (No limitation on size)

-> In TypeScript arrays can store hetereogenious values also

-------------------------------------------------------------------------

let fruits:string[];

fruits = ['mango', 'apple', 'banana', 'orange'];

console.log(fruits);

---------------------arraysdemo.ts--------------------------------------

let fruits:string[];

fruits = ['mango', 'apple', 'banana', 'orange'];

console.log(fruits);

let animals : Array<string> ;

animals = ['Tiger', 'Lion', "Rabbit"];

console.log(animals);

let persons: Array<string | number>;

persons = ['Ram', 45, 'Anil', 50];

console.log(persons);

let genricArray : Array<any> ;

genricArray = ['Ram', 20, true, 450.56];

console.log(genricArray);

------------------------------------------------------------------------

Last session : Arrays in Typescript

-------------------------------------------------------------------------

-> Arrays are used to store group of values

-> In TypeScript, Arrays don't have any limit on size

-> In TypeScript, Arrays can store hetereogenious values also

------------------------------------------------------------------------

Today's session : Functions

------------------------------------------------------------------------

-> Functions are primary building blocks of any program.

-> In Java Script, Functions are very important bcz java script itself is a functional programming language.

-> Functions ensure that the program is maintainable and resusable

-> TypeScript supports for both functions and OOPS.

-> In TypeScript, functions can be of two types

1) Named functions (function which contains a name)

2) Anonymous functions (function without name)

--------------------------Named function-------------------------------

function welcome(){

console.log("Welcome to Ashok IT");

}

welcome( );

-> Function can have a parameter and can have a return type also

function add(x:number, y:number) : number {

return x + y;

}

let result = add(10,20);

console.log("Sum ::"+result);

----------------------------------------------------------------------------------

-> The function which doesn't contain any name is called as an Anonymous function.

-> Anonymous function is the one which is declared as an expression

let result = function (x:number, y:number) : number {

return x+y;

}

result(10,20);

----------------------------Working Example-------------------------------------

function doWork(x:number, y:number): number {

return x + y;

}

let result = doWork(10,20);

console.log("Result Of N.F :: "+ result);

let sum = function (x:number, y:number) : number {

return x + y;

}

let value = sum(30,40);

console.log("Result Of A.F :: "+ value);

-----------------------------------------------------------------------------------

Function with Parameters

----------------------------------------------------------------------------------

-> In Typescript function can take parameters

-> Parameters are the values or arguments which we will supply to function as input

function fullName(fname:string, lname:string) : string {

return fname+" "+lname;

}

fullName("Ashok"); // Invalid - Function expecting 2 args but passed 1

fullName("Ashok", "IT", "School") ; // Invalid - Expected 2 args but passed 3

fullName("Ashok", "IT") ;; // valid

Optional Parameters

-------------------

-> In TypeScript we have optional parameter functionality

-> Optional Parameter should be the last argument

-> Optional Parameter will be represented using ?

Rest Parameters

----------------

-> Rest Parameters are similar to varargs in java

-> If we don't know how many parameters we need to take for a function then we can simply use Rest Parameters

function(msg:string, ...names:string[]){

//logic

}

Note: Rest Parameter should be the last parameter of the function.

----------------------------------------------------------------------------------

function greeting(msg:string, name?:string) : string {

return name + " "+ msg;

}

console.log(greeting("Good Afternoon", "Ashok"));

console.log(greeting("Good Afternoon"));

function wish(name:string, msg:string="Hi") : string {

return name + msg;

}

console.log(wish("Ashok", "Hello"));

console.log(wish("Ashok"));

function greet(msg:string, ...names:string[]){

console.log(msg + "--"+names.join(","));

}

greet("Good Afternoong", "Ashok", "Ram", "Raj");

greet("Good Afternoong", "Ashok", "Ram", "Raj", "Bharat", "Chanti", "Suresh");

----------------------------------------------------------------------------------

Last session : Functions

------------------------------------------------------------------------

-> Function with Parameters

-> Function with Return Type

-> Function with Parameters & Return Type

-> Function with optional parameters

-> Function with default parameters

-> Function with Rest Parameters (var-args)

------------------------------------------------------------------------

Today's session : OOPS in TypeScript

-------------------------------------------------------------------------

-> OOPS stands for object oriented programming system

-> In OOPS programming languages "Classes" are fundamental entities which are used to create re-usable components.

-> "Class" is a plan or model which is used to create the object.

-> "Object" is a physical entity which is used to store the data.

Class contains following things

-------------------------------

1) Fields / Properties : Variables declared in class

2) Methods : To perform action

3) Nested Class : A class can contain another class

4) Constructor : It is used to initialize the object

5) Object : Physical item or collection of properties

Syntax to declare class in TypeScript

-------------------------------------

class <class-name> {

//fields

//methods

}

Example for class

------------------

class Student {

let studentName : string ;

let studentRank : number ;

let studentMarks : number ;

getStudentGrade ( ) : string {

//logic

return grade;

}

}

-------------------------------------------Student.ts----------------------------------------

class Student {

studentName: string;

studentRank: number;

studentMarks: number;

getStudentGrade() : string {

if(this.studentMarks >= 75 ){

return "A";

}else if(this.studentMarks >=65 && this.studentMarks < 60){

return "B";

}else{

return "C";

}

}

}

let s1 = new Student(); //obj creation

s1.studentName = "John";

s1.studentMarks = 50;

s1.studentRank = 3;

console.log("Grade :: "+ s1.getStudentGrade());

let s2 = new Student() ; //obj creation

s2.studentName = "Smith";

s2.studentMarks = 90;

s2.studentRank = 2;

console.log("Grade :: " + s2.getStudentGrade());

----------------------------------------------------------------------------------------------

Constructors in TypeScript

----------------------------------------------------------------------------------------------

-> Constructor is a special function which is part of the class.

-> Constructor is used to initialize the object.

-> Constructor name will be same as Class Name.

-> In constructor we can access class members using 'this' keyword

-> When we create object for a class constructor will be called automatically.

-> Constructor will take arguments but it can't return anything.

\*\*\*\* -> In TypeScript, Constructor doesn't support for overloading \*\*\*\*

\*\*\*\* -> In TypeScript, we will use 'constructor' keyword to define constructor \*\*\*\*\*

-------------------------------User.ts-------------------------------------------------------

class User{

userId : number;

userName : string;

constructor(userId : number, userName:string){

this.userId = userId;

this.userName = userName;

}

}

let user = new User(101, "Ashok");

console.log("user id :: "+ user.userId);

console.log("username :: "+user.userName);

----------------------------------------------------------------------------------------------

Last session : Classes, Objects & Constructor

-------------------------------------------------------------------------

-> Class is a plan or model which is used to create the object

-> Object is a physical entity which is used to store the data

-> Constructor is a special function which is used to initialize the object

-------------------------------------------------------------------------

Today's session : Access Modifiers

-------------------------------------------------------------------------

-> Access modifiers specify weather the members of the class can be accessible or not.

-> Access modifiers are used to achieve security in OOPS.

-> TypeScript supports 3 access modifiers

1) public

2) private

3) protected

-> public members of the class are accessible anywhere in the program. public members can be accessible inside the class and outside the class also.

-> private members of the class are accssible only with in the same class. If we try to access private members outside of the class then we will get compiletime error.

-> protected members are accessible with in the same class and in corresponding child classes also.

Syntax for creating class members with Access Modifiers

---------------------------------------------------------

class <class-name> {

accessmodifier propertyName : dataType ;

accessmodifier functionName : returnType {

//body

}

}

-------------------------------------------------------------------------

Interfaces

-------------------------------------------------------------------------

-> Interface is the one which contains only abstract methods and properties

-> The method which doesn't contain body is called as abstract method

-> In Interface we will write only method declarations. Interface doesn't contain method implementations.

-> To implement an interface, class will use "implements" keyword.

-> When a class is implementing an interface, its mandatory that all methods of interface should be implemented in that class.

-> Interface methods are by default public.

-> One interface can be implemented by multiple classes.

-> One class can implement multiple interfaces also

\*\*\*\*\*\*\*\*\*\*\*\* Interfaces are called as contracts \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Syntax For Interface Creation

-----------------------------

interface InterfaceName{

property : dataType ;

method(args) : returnType ;

}

Implementation class for Interface

-----------------------------------

class ClassName implements InterfaceName {

property : dataType ;

method(args) : returnType {

//method implementation here

}

}

TypeScript - Modules

---------------------

-> In Large applications we will have multiple classes

-> It is highly recommended to write each class in a seperate file.

-> To access the class of one file in another file we will use Modules concept in TypeScript.

-> Module is a file (ts file) which can export one or more classes to other files.

-> To export a class we will use 'export' keyword (source file)

-> To import a class we will use 'import' keyword (desitnation file)

------------------------------- Student.ts----------------------------------------

export class Student {

studentId : number;

studentName : string;

constructor(id:number, name:string){

this.studentId = id;

this.studentName = name;

}

}

-------------------------------School.ts------------------------------------------

import {Student} from "./Student" ;

class School{

students : Student[] = [

new Student(101, "John"),

new Student(102, "Smith"),

new Student(103, "Nick")

];

display() : void {

for(var i in this.students){

console.log(this.students[i]);

}

}

}

let school:School = new School();

school.display();

--------------------------------------------------------------------------------------------------------------

Last session : Modules in TypeScript

------------------------------------------------------------------------

-> If we want to use class/interface/enum of file in another file then we will use Modules concept.

-> To work with modules we will use export and import options.

-------------------------------------------------------------------------

-> TypeScript is a programming language which is developed based on java script.

-> TypeScript is superset of java script.

-> TypeScript built on top of Java Script

-> TypeScript can't execute directley in browser.

-> TypeScript file should be converted into java script file using TypeScript compilor that is called as 'Transpilation'.

-----------------------------------------------------------------------------------------------

Angular

----------------------------------------------------------------------------------------------

-> Angular is a client side framework which is used to create web applications.

-> Angular is developed by using TypeScript

-> Angular developed by Google company

-> Angular is free & open source

-> Angular is cross platform (It works in all operating systems)

-> Angular is cross browser compatible (It works in all browsers)

-> Angular is mainley used for Single Page Applications Development (SPA)

Multi Page Application vs Single Page Application

-------------------------------------------------

-> In Multi page application for every request web page will be reloaded

-> In signle page application for every request only content will be re-loaded.

----------------------------------------------------------------------------------------------

Building Blocks of Angular

----------------------------------------------------------------------------------------------

1) Component : Application state + Application logic

2) Metadata : Details about component / Module

3) Template : View File (Presentation logic) - HTML

4) Databinding : Connection between HTML elements and Component properties

5) Module : Group of Components, Directives and Pipes

6) Service : Re-usable business logic

7) Dependency Injection : Injecting service objects into components

8) Directives : To manipulate DOM elements

9) Pipes : Transforming values before displaying

Last session : Angular Framework Building Blocks

-----------------------------------------------------------------------------------

-> Angular is a framework which is used to develop web applications

-> Angular is given by Google

-> Angular is open source and free

-> Angular supports cross platform

-> Angular support cross browsers

-> Angular developed using TypeScript

----------------------------------------------------------------------------------

1) Component

2) Metadata

3) Template

4) Databinding

5) Module

6) Service

7) Dependency Injection

8) Directives

9) Pipes

10) Routings

----------------------------------------------------------------------------------

-> Component is a typescript class which is responsible to handle requests coming from template.

-> Template is a html file which contains presentation logic

-> Databinding is used to bind the data between component and template. Component properties will be binded to HTML elements for data binding.

-> Service is a Typescript class which contains re-usable business logic.

-> Dependency Injection is used for Injecting Service class obj into Component class obj.

-> Data about the data is called as Meta Data. Relation between component and template will be maintained in meta data.

-> Directives are used to manipulate DOM elements

-> Pipes are used to transform the data before displaying in template.

-> Collection of components is called as Module

-> Routings are used to navigate from one page to another page.

----------------------------------------------------------------------------------

Angular Environment Setup

--------------------------------------------------------------------------------

1) Install Node JS

2) Install Typescript

3) Install Angular CLI

4) Install Visual Studio Code

-----------------------------------------------------------------------------------

Last session : Angular framework building blocks & Env setup

-------------------------------------------------------------------------

Today's session : First Angular Application Development

-------------------------------------------------------------------------

-> To create angular application execute below command in command prompt

ng new <app-name>

-> Once application got created, nagivate into application folder and execute below command to run angular application.

ng serve

-> Angular applications will be deployed into live server which runs on port number 4200. Once angular application deployed into live server then we can access using below URL

http://localhost:4200/

-------------------------------------------------------------------------

-> When we create angular application by default 'app-component' will be created.

-> 'app-component' will be called as Parent component in the angular application.

-> Angular application execution will begin from 'app-component' only

app.module.ts

app.component.ts ----> logic to handle request

app.component.spec.ts --> Unit testing logic

app.component.html --> template( presentation logic )

app.component.css --> styles will write here