**Phase 2 : Day 1**

**12-06-2021**

**Babel : babel is a type script external JavaScript module or library which help to convert ES6 to ES5 code.**

**React JS : JSX (JavaScript and XML) to convert JSX to Plain JavaScript and HTML we use babel.**

**JSX is one of features we have writing html code in JavaScript.**

**webpack : webpack is a static module bundler.**

**Module is a collection of variables, function, classes, interfaces part of different files.**

**Webpack treats all files and assets as module.**

**Web pack create dependency graph which help to describe how module are related or connected to each other using require(import) and export state between two file or modules.**

**If we want to use webpack we have to install few dependencies.**

**Please install these two module without or with g**

**npm install –g webpack**

**npm install –g wepack-cli**

**Phase 2**

**Node JS**

**Typescript**

**Angular 8 , 10 etc**

**Json-server to create fake rest service.**

**Angular Framework : Angular is a Open Source Framework which help to create SPA (Single Page Application).**

**One.html welcome.html**

**Hyperlink**

**Submit button**

**Normal button**

**Using JavaScript**

**When we move from one page to another the whole get loaded in browser once again.**

**According SPA rather than loading whole page we are loading only part of the page base upon the path.**

**Angular, React and Vue JS : which help to create SPA.**

**Angular JS 1.0,1.1 1.x**

**Angular JS is base upon ES5 features.**

**HTML/HTML5, CSS/CSS3 and JavaScript using ES5.**

**Angular Framework 2 to 12 version**

**HTML/HTML5, CSS/CSS3, JavaSCript using ES5 and TypeScript with ES6 features.**

**Difference between Angular JS Vs Angular Framework**

**TypeScript**

**Features**

**let,const and var**

**data types (Javascript using ES5 and ES6 doesn’t data types).**

**types of loop : in loop and of loop**

**function types :**

**optional parameter, default parameter, return type of function, spread and rest operator.**

**OOPs concept using ES6 with typescript**

**Class, constructor, interface**

**Module**

**export and import**

**welcome.ts**

**then convert this ts file to js using command**

**tsc**

**npm install –g tsc**

**babel : it is a type of transpiler**

**to convert ES6 to ES5**

**JSX to JavaScript (React JS)**

**typescript : it is a type of transpiler**

**tsc**

**convert ts to js**

**npm install typescript –g**

**browser doesn’t support ts file. So we have to covert ts to js using transpiler.**

**Using tsc we are converting ts to js and run the js using node or using html page.**

**Data Types**

**Typescript support data types**

**Syntax**

**let/var variableName:datatype = value;**

**Array with Data types**

**var num = [100,200,300,400,500]; in ES5**

**var info = [100,”Raj”,true] in ES5**

**so we can create array using 2 ways**

**literal style**

**let num:datatype[] = [v1,v2,v3];**

**object creation style**

**let num:Array<type>= new Array();**

**function types**

**In JavaScript using ES5 or ES6 only function name must match doesn’t matter number of parameter as well as type of parameter.**

**Optional parameter must be start from right side. So in one function we can use on optional parameter or more than one but no gap between two optional parameter.**

**In ES5 and ES6 JavaScript function can return any value or may not return.**

**Spread and rest**

**De structure**

**OOPs Concept using Typescript**

**Constructor with parameter**

**Constructor short cut initialization**

**We can use public or private keyword with constructor parameter variable in typescript to make those variable is a type of instance variable.**

**Phase 2 : Day 2**

**13-06-2021**

**Inheritance**

**Inheritance is use to inherits the properties and behaviour of old class to new class.**

**class Employee { //generics**

**}**

**class Manager extends Employee{ // specific**

**}**

**TypeScript support interface.**

**Interface is a to provide the specification.**

**It is also one of type reference data type like a class.**

**But all function incomplete(abstract function/methods) and we can declare normal variable like class.**

**class extends another class (only one)**

**class implements interface (more than one)**

**interface extends another interface (more than one)**

**interface extends/implements not possible for class**

**(interface can’t extends or impelements to class)**

**Class extends class (only one)**

**Class implements interface (more than one)**

**Interface extends interface (more than one)**

**Interface extends/implements class (Wrong)**

1. **Interface contains variable and function (not advisable)**
2. **It contains only function**
3. **It contains only variable**

**While creating project with only typescript we have to create typescript configuration file.**

**Syntax to create the typescript file is**

**tsc --init**

**tsconfig.json**

**Decorator : A decorator is a special kind of declaration (meta-data) that can be attached with class, property, function as well as parameter.**

**It is like a Annotation in Java.**

**All decorator start with @ followed by decorator name.**

**Using Typescript we can create own or custom decorator with help of function do add extra behaviour to our class, property or parameter variables.**

**Angular Decorator**

**@NgModel**

**@Component**

**@Input**

**@Output**

**@ViewChild**

**@Injectable**

**@Pipe**

**Etc**

**Angular Framework which help to create SPA (Single Page Application). Using Angular Framework we can create desktop application, web application as well as mobile application (Ionic Framework).**

**React JS -🡪 Native React (Mobile application).**

**Angular Framework 2 to 12**

**TypeScript**

**Google company They provided angular-cli (command line interface) which help to create sample angular projects.**

**ng (next generation) for HTML or DOM.**

**Angular**

**npm install –g @angular/cli**

**or**

**npm install @angualr/cli**

**or**

**npm install –g @angualr/cli@versionNumber**

**after installation successfully**

**ng –-version**

**syntax to create angular project**

**ng new project-name**

**ng new welcome-app**

**routing option (Yes/No)----🡪No**

**stylesheet ----------🡪Css : Enter**

**Move inside a project directory**

**cd project-name**

**cd welcome-app**

**then open the project in VS code.**

**To run the project we have to run the command as**

**ng serve (this command must be run inside a project folder).**

**If it ask policies option (yes/no) : No/Yes**

**After 100% compiled the project**

**Then open the browser**

[**http://localhost:4200**](http://localhost:4200)

**src 🡪**

**app 🡪**

**app.component.html (open this file)**

**template**

**app.component.ts (component)**

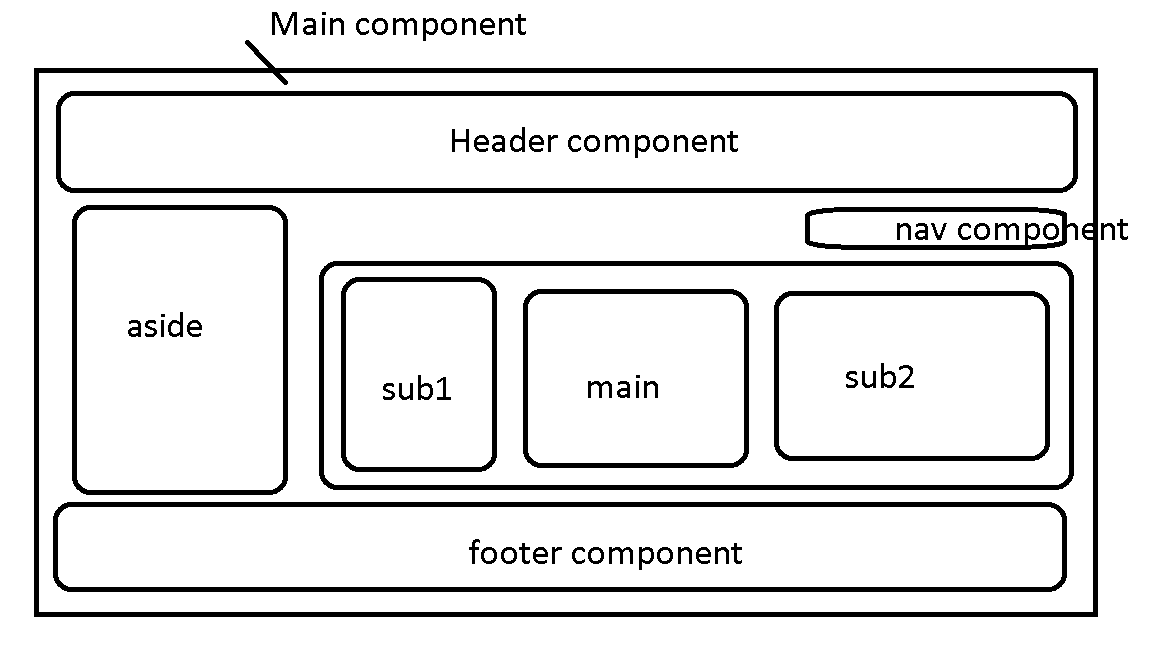
**app.component.css (css styling file)**

**Component : Component is use to control the view or part of the View. Using component we can created custom UI Component or User-defined tags.**

**Using Component we are creating user-defined tags**

**<b>Hi</b>**

**<font1 color=”red”></font1>**

****

**@Component : It is a type of decorator provided by angular to make normal typescript class is a type of special class.**

**@Component decorator is a type class decorator.**

**@Component decorator which contains lot of pre-defined properties**

**Selector : this property is use to create user-defined <app-root></app-root>**

**templateUrl : This property is to connect external html page.**

**stylesUrl : This property is use to connect external CSS page.**

**Where ever we use <app-root></app-root> The code which written in html page get replace that tag.**

**app.module.ts**

**@NgModule : It is type of decorator which provide by angular which help to make the class is a module class.**

**Module a collection of more than component, services, pipe etc.**

**Module is a collection of variable, function, class and interface in typescript.**

**@NgModule contains more than one property**

**declaration : In this section we have to declare all component declaration.**

**In Angular if you want to create one single html page it must be part of component and that component must part of module.**

**As a good practise every file contains only one components.**

**import : This section we have to provide pre-defined as well as user-defined module declaration.**

**BrowserModule : This module is use to display the data in browser page.**

**provider : This property help us to provide the details about angular service class.**

**bootstrap : This property help us to bootstrap or load those component at the beginning of the projects.**

**main.ts**

**This file provide the details about main module to load first ie bootstrap.**

**OrderModule CustomerModule**

**DisplayOrderComponent ViewCustomerComponent UpdateOrderComponent DeleteCustomerComponent**

**AppModule**

**AppComponent**

**EmployeeComponent**

**index.html**

**Please create component ts file as**

**child.component.ts**

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

@Component({

    selector:"app-child",

    templateUrl:"./child.component.html",

    styleUrls:["./child.component.css"]

})

export class ChildComponent {

    msg:string="This is child component";

}

**Then create html page as**

**Child.componen.html**

<div>

  <h2>Welcome to My Angular Project Akash</h2>

  <p>Id is {{id}}, name is {{fname}}, age is {{age}}

    Result is {{result}}

  </p>

  <p>

    {{dis()}}

  </p>

  <app-child></app-child>

  <app-child></app-child>

  <app-child></app-child>

  <app-child></app-child>

</div>

**Then create css page**

h2{color: darkmagenta;}

p{color: darkorange;}

**Then write this component declaration in app.module.ts file**

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

import { BrowserModule } from '@angular/platform-browser';

import { AppComponent } from './app.component';

import {ChildComponent} from './child.component';

@NgModule({

  declarations: [

    AppComponent,ChildComponent // all component declaration provide

  ],

  imports: [

    BrowserModule

  ],

  providers: [],

  bootstrap: [AppComponent]

})

export class AppModule { }

**Phase 2 : Day 3**

**19-06-2021**

**Using this command after compiled angular project it automatically in default browser with url as**

[**http://localhost:4200**](http://localhost:4200)

**ng serve –o**

**Angular Data Binding**

**Data binding is use to share the data between component to template and vice-versa.**

**Data binding provide the bridge between component (ts) and template (view/html page).**

**ng serve –o**

**create the component using angular cli command**

**ng generate component component-name**

**or**

**ng g c component-name**

**2 types**

1. **One – way data binding**
   1. **String interpolation : component --🡪view**

**{{}}**

**{{variableName}}**

**{{10+20+30}}**

**{{functionName()}}**

**ng g c string-interpolation**

* 1. **Property binding : component -🡪 view**

**[]**

**Html**

**<input type=”text” value=”hello”/>**

**<input type=”text” [value]=”variableName”/>**

**<div [innerText]=”variableName”></div>**

**<p></p>**

**<span></span>**

* 1. **Event binding : view --🡪 component**

**()**

**Angular using same dom events but all event pre-fix on removed and event name wrap with ().**

**DOM element Angular Event**

**onClick (click)**

**onDblClick (dblclick)**

**onMouseOver (mouseover)**

**onSubmit (ngSubmit)**

**Template Reference : template reference is use to create the object or reference of DOM elements. With the help of that reference we can get the value of any dom element in components.**

**Syntax**

**<input type=”text” #referecenceName/>**

**Html**

**<input type=”text” id=”obj”/>**

**<input type=”button” value=”click” onClick=”fun()”/>**

**JavaScript**

**Pull the value from html in JavaScript function**

**function fun() {**

**var obj = document.getElementById(“obj”).value;**

**}**

**Two way data binding = component to view and view – component.**

**Event Binding + String interpolation = two way data binding.**

**Event binding -🡪 View -🡪Component**

**String interpolation 🡪 Component -🡪View**

**View 🡨------🡪Component**

1. **Two way data binding : 2 data binding using ngModel attribute**