70% 80% practical + 20% theory

Day 1 : 12-04-2021..

Chrome :

Node JS Software 14.x /15.x

VSCode

REST API or JSON Server

Html, css, JS ES5

Angular 2 to 11 version

Node JS (npm)

HTML,CSS,JS, typescript

Angular 11 Version

SPA

TypeScript : 1, 2

Day 3 : Angular, SPA, Angular architecture, creating angular project using angular CLI and Data binding

Day 4 : Angular forms template driven form and model driven form with validation

Day 5 Angular Service : rxjs Calling REST API

Day 6 : Angular component life cycle, Angular component communication

Day 7 : angular material /bootstrap , Pipe and creating custom pipe

Day 8 : Routing, deploy the application in backend technologies, Testing and Other features.

<https://www.google.com>

req(http/https)---🡪

Client Server

🡨----------Res(http/htps) HTML/HTML5

CSS/CSS3

JS (JavaScript)

JavaScript was object based interpreter scripting language upto ES5.

ES6 (ECMA) Script

ES is concept and JavaScript as well as typescript are implementation of ES6.

jQuery Library

jQuery provide lot of pre-defined function which internally connected to each other to read , write and update dom very easily.

MVC : Model View Controller.

View : HTML/JSP/JSF

Framework : framework provided lot of pre-defined API which internally connected to each other to perform a specific task. The implementation of all design pattern is taken are by framework.

70 to 80 task taken care by framework. But framework is a final product it is template or protocol.

Angular JS : HTML/CSS/JavaScript using ES5.

1.x

Angular Framework. HTML/CSS/JS/TypeScript.

DOM :document object model.

Html tags. Read dom dynamically, write, and update.

document.formname.componentname.value

document.getElementById(“idname”).value;

Node JS : Node is not a library or framework. It is a run time environment for JavaScript program or library or framework.

Like JRE for Java. Node JS for JavaScript.

Before Node JS Javascript is known as Client Side scripting language but after node js JavaScript is use to write Server side programming language.

Node JS provided lot or pre-defined modules (like a package in java) which help to create file handling programming, server side programming, rest api, connecting data base using JavaScript.

JEE

Spring boot

Asp.net

Php

Python

Node JS

TypeScript : TypeScript is a super set of JavaScript. It support all features provided by ES6.

Upto ES5 to declare the variable in javaScript we are using var keyword.

But in ES6 or TypesScript we can use var, let and const.

var Vs let

var is use to declare global scope where let is use to declare local scope.

Using var we can re-declare the variable but using let we can’t re-declare same variable once again with same value or different value.

We have to convert ts to js file

Tsc (typescript transpiler)

Typescript : it is type of transpiler which help to covert ts to js.

Npm (node package manager). Which help to download external modules.

**npm install –g moduleName**

**npm install –g typescript**

converting ts to js file

**tsc filename.ts**

by default tsc convert js file (ES5) ES5 not let and const keyword.

JavaScript ES6

typescript

**dataTypes :**

ES5 as well as ES6 JavaScript.

var a=10;

a=”Ravi”;

a=true;

a = new Date();

syntax

var/let variableName:dataType;

let a:number;

let b:number;

let name:string;

let msg:any;

let a:number=100;

let a =100;

let a:number;

let result:boolean = true;

let result = false;

let result = “true”;

a=300;

array using Typescript

ES5

var names=[“Ravi”,”Ramesh”,”Raju”,100,true,”Ajay”];

Typescript

let n:number[]=[100,200,300,400,500];

let names:string[]=[“Ravi”,”Ramesh”,”Rajesh”];

let data:any[]=[100,”Ramesh”,true,200];

Generics style

let n:Array<number>=[100,200,300,400,500];

let names:Array<string>=[“Ramesh”,”Ajay”,”Vijay”];

type of functions

Rest and spread operator or parameter

Rest parameter is use to receive 0, 1 and may parameter

And Spread parameter is use to pass the value to rest parameter if variable is a type of array.

We can use only one rest operator or parameter in function and it must be last parameter in functions.

Callback function, anonymous function and array function.

Day 2 : 14-04-2021

Arrow function :

Arrow function is a one of the means features of ES6.

It is use to create a function a cleaner manner was compare to normal function. It is like a lambda expression in Java.

OOPs Concept Using TypeScript

object : any real world entity

property (state) -🡪 have 🡪 variables, fields,

Person

Behavior -🡪do/does -🡪 function / methods

Bank

Car

Animal

Customer

class : blue print of object or template of object.

Up ES5 JavaScript

ES6 style

1. class
2. constructor

In Typescript we can’t write more than one constructor.

By default (default constructor ie empty) or we can write parameterized constructor but only one.

Constructor short cut initialization.

Inheritance :

Interface :

Module : module is a collection of variable, function, classes, interfaces. Like a package in java or namespace in C#.

Using export we can export function, variable, classes and interface to another module. Another module can import those function, variable, classes and interfaces.

**Decorator** : Decorator is a type of special kind of declaration that can be attached to the classes, properties and function. Like a annotation in java. It is also known as meta-data(data about data).

Decorator can be evaluated into a function that be called at the runtime.

All decorator are start with pre-fix @ followed by decorator name.

Decorator

@Component

@Input

@NgModule

@Injectable

Modules

@angular/core

@angular/forms

@angular/common/http

Angular JS 1.x Version

Angular JS depends upon the html/css/javaScript ES5 style.

This framework is use to develop only web application.

Angular JS base upon the MVC architecture framework.

Angular Framework 2 to 11 Version

Angular Framework depends upon the HTML/CSS/JavaScript and

Typescript. Angular framework open source belong google company.

Angular Framework 2 to 11 version is use to create desktop, web application as well as mobile application.

Angular framework base upon the component base architecture.

Angular mainly use to create SPA(Single Page Application).

Multi page application.

index.html welcome.html

hyperlink

button

submit button

using JavaScript

AJAX

XMLHttpRequest or ActiveXObject

Whole DOM loaded once again from a scratch.

SPA

Component : it control the view or part of view. Every component work independently.



Angular CLI (Angular Command Line Interface).

Angular CLI provide set of command which help to create the project, test, build project we can add dependencies.

ng command (next generation for DOM or HTML).

Adding extra behavior to DOM or html page.

Angular

We have to install angular module with the help of npm command.

npm install –g @angular/cli (globally any location we can create the project)

npm install –g @angular/cli@version

or

create folder or directory AngularProjects

npm install @angular/cli (project must be create in current or local folder).

ng –version

creating angular projects

ng new project-name

ng new welcome-app

Setting help option yes/no : yes or no

Angular routing : no

Stylesheet option : css

cd welcome-app move inside a project folder.

To run the project we have to run the command as

ng serve

details for setting option (yes/no);

after compiled 100% successfully.

Then open the browser

<http://localhost:4200>

New project created

**ng new angular-data-binding**

Day 3 : 15-04-2021

app.component.css

app.component.html

app.component.ts

app.modue.ts

app.component.spec.ts

@Component is a pre-defined decorator

Three properties

selector : name for decorator behave like a user-defined tags.

Using selector property we are creating user-defined tags.

<h1><//h1>

<form></form>

templateUrl: This property is use to connect the html page.

styleUrl : This property is use to connect css like link tag in html and css.

<link stylesheet =”rel” href=”style.css”/>

app.module.ts

@NgModule

Module : module is a collection of more than one component.

Property

declarations : we have to give all component declaration.

**imports:** this property is use import pre-defined or user-defined modules.

**provider :** This property hold angular service class details.

boostrap : So we have to provide main component which you want to load at the beginning of the application.

**main.ts**

platformBrowserDynamic().bootstrapModule(AppModule)

  .catch(err => console.error(err));

From this file angular get the details about main module to load using bootstrap functions.

index.html

Creating user-defined components.

Header component

**Angular data binding**

Using Data binding we can share the data between component (ts) to template (html or view).

Data binding provide the bridge between component to view and vice-versa.

2 types

1. one way data binding
   1. string interpolation : component ---🡪 view

{{variableName}}

{{functionCall()}}

{{expression }}

Result is string.

ng g c string-interpolation

* 1. property binding : component 🡪 view

[attributeName]=”variableName”

ng g c property-binding

String interpolation always output string in template

Property binding output depending upon the data types.

* 1. Event binding : template 🡪 view

Angular event like a same type of event provided by JavaScript

JavaScript Angular Event

onClick (click)

onDbClick (dblclick)

onSubmit (ngSubmit)

onFocus (change)

(input)

(mouserover)

ng g c event-binding

2 way binding = Event binding + string interpolation or property binding

Passing the vale of textfield, passwordfield, radiobutton, dom element from view (template) to component

Using template reference.

<input type=”text” #nameRef/>

1. two way data binding

two way data binding from component to view and vice-versa.

View(Template) <-----------------------🡪Component

ngModel : ngModel is a pre-defined attribute part of FormsModules. So we have to import FormsModule in app.module.ts file.

Syntax

<input type=”text” [(ngModel)]=”variableName”/>

ng new angular-forms ( forms and validation)

ng new angular-directives

We can create the component using ng command.

ng generate component componentname

or

ng g c componentname

AppModule

AppComponent

N component Child to app component

CustomerModule AccountModule LoginModule

DisplayComponent AccountCreate

CustomerCreate ViewAccount

DeleteCustomer

Day 4 : 16-04-2021

Using template reference we can pass the value from template to component.

Using 2 way data binding we can assign the value to component’s variables.

Angular Forms : Using angular forms we can pass the group value values from template to component

2 types of forms

1. Template Driven Form
   1. Flow of the application template or view ---🡪 component
   2. Easy to create this type of form.
   3. Good for simple type of forms.
   4. More code on html or template side.
   5. Submit event is required to process the form.
2. Model Driven Form or Reactive Form
   1. Flow of the application component to view (template)
   2. Good for complex type of forms.
   3. More code on ts file (component side)
   4. Any other type of event to process the forms.

**ng serve –o** (automatically open the application on default browser)

ng g c tdf-login-page

ng g c mdf-login-page

In Template Driven Form we have to create the reference of forms

ngForm is a pre-defined attribute help to create the FormReference.

ngForm attribute is a part of FormsModule So we have to import this FormsModule in app.module.ts file.

<form #loginRef=”ngForm”>

</form>

Model Driven Form

According to Model Driven form Every textfield, passwordfield, radiobuton, checkbox, textarea is known as FormControl.

FormController wrap by FromGroup.

<input type=”text” name=”user”/>

Login Page --🡪FormGroup

UserName 🡪 FormControl

Password 🡪FormControl

FormGroup

City 🡪FormControl

State 🡪 FormControl

PinCode🡪FormControl

FormGroup is a collection of more than FormControl as well as FormGroup.

While writing FormGroup and FormControll in template we will get the error. To remove this error we have to import ReactiveFormsModule in app.module.ts file.

ng new angular-directive

Angular Form Validation :

Angular provided 6 pre-defined property which help to do validation.

ng-valid, ng-invalid base upon the rules

ng-touched(not visited) , ng-untouched(visited)

ng-dirty(not change) , ng-pristine(change)

HTML5 Validation

HTML5 we can do validation using required, minlength, maxlength and pattern.

Angular Directive

Angular Directive is use to add extra behavior to DOM or html page.

3 types

1. Component directive : Component is a type of directive which help to created user-defined tags.

@Component({

Selector:”my-tag”

templateUrl:”./filename.html”

})

export class Demo {

fields;

functions;

}

1. Structure directive : it is use to add and remove dom elements dynamically.

\*ngIf

\*ngFor

1. Attribute directive : attributed directive is use to add styling for web page dynamically.

ngStyle style

ngClass class

Create two component

ng g c structure-directive

ng g c attribute-directive

**ng new component-communication**

**ng new angular-service**

**bootstrap / material**

**spring boot : deploy**

**build angular project :**

**6 methods**

**Create login with post method**

**Check login with get method**

**Get, post, put and delete for customer**

**REST API : Login**

**Customer : CRUD Operation**

**emp.json (using node module make json file as server).**

**Day 5 : RxJS and Angular Service calling REST API.**

**19-04-2021**

**Attribute directive : ngStyle and ngClass**

**Angular Service**

**ng new angular-service**

Angular Service :

Using Angular service we can provide common business logic for more than one components.

View -🡪 template or html

Controller / Component -🡪 ts file

Service -🡪 Angular Service

2 types

1. User-defined service
   1. Creating object using new keyword
   2. Creating object using DI.
2. Pre-defined service to call Rest API

Open the Angular service project

And create two component

ng g c first

ng g c second

IOC : Inversion of control : It is a concept. Design pattern. In place of creating object or resource explicitly allow to maintain by container. Pull from container whenever require.

DI : Dependency Injection : DI is a implementation of IOC.

Constructor base

Setter base

But in Angular we can achieve DI using constructor base.

If we want to DI for angular service class we have to create use-defined class with decorator @Injectable.

We have to register this class details in module level or component level with the help of provider attribute.

Angular pre-defined service

HttpClient is a pre-defined API provided by angular to call REST API.

Using HttpClient we can call get(), post(), put() and delete() REST API methods.

HttpClient take the help of rxjs Observable API to subscribe the data in angular side.

From UI or Frontend technologies we can call REST API using two ways

1. Using promise
2. Using Observable

Promise is a pre-defined object provided by JavaScript which help to load asynchronous data or event data from REST API.

Creating user-defined promise.

So promise can resolve (successfully load the data) or rejected (error generated).

let prom = new Promise((resolve,reject)=> {

    resolve("successfully done")

})

prom.then().catch()

If promise resolve then() function get call to load the data.

If promise rejected catch() function get call to get error message.

Reactive Programming : It is use to do events(data) as a stream of values.

Reactive Programming base upon the Observable design pattern.

Observable allow you to handle different asynchronous event or data.

Consumer can then subscribe to observable to listen all event or data.

Rxjs library for JavaScript or typescript.

Install rxjs module using npm module

npm install rxjs

subscribe function takes 3 parameter

next : This function help load the data one by one

error :This parameter get call if any error generated at beginning or middle or at last

completed : after loaded all data successfully third parameter get call.

In JavaScript we can’t use import concept.

To run this program we have to install esm modue (Embedded System Module) which help run import concept in JavaScript.

npm install esm

node –r esm demo1.js

REST API to do CRUD Operation

First install

Json-sever module help to make server as a static json file.

npm install –g json-server

json-server employee.json

In angular-service project create four component

ng g c create-employee

ng g c display-employee

ng g c update-employee

ng g c delete-employee

HttpClient API help to call REST API method. So we have to do DI for HttpClient in user-defined service.

HttpClient API is part of HttpClientModule so we have to import HttpClientModule in app.module.ts file.

Get , Put Operation using Angular Service

Sync call same code

Component communication 🡪

ng new angular-component-communication

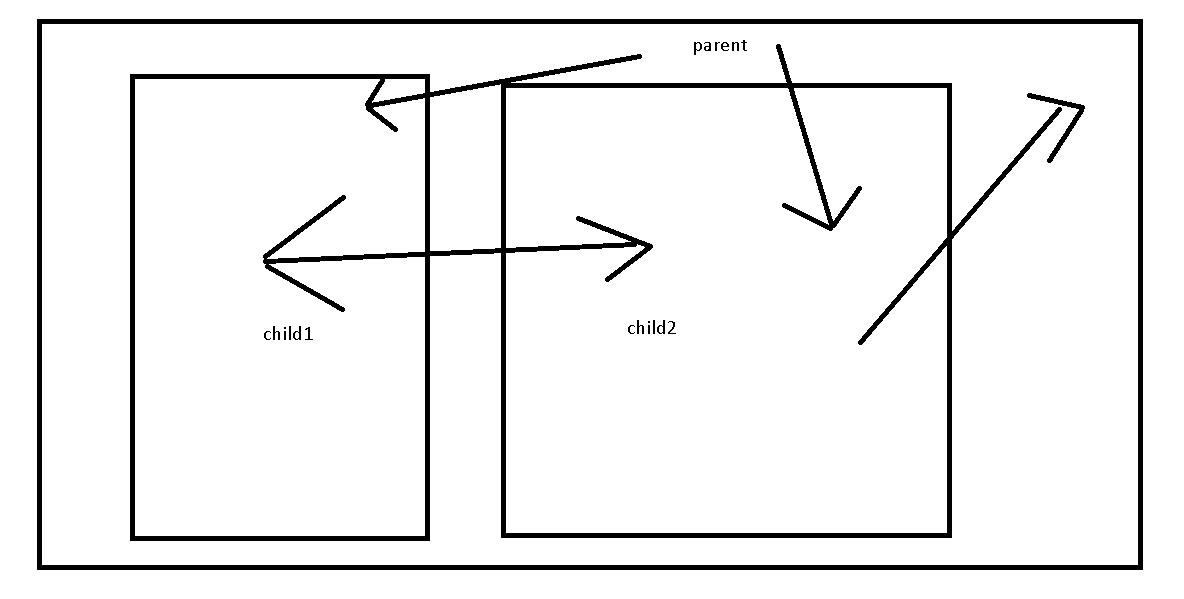
Angular life cycle

Bootstrap

Angular routing

**20-04-2021**

Component communication



1. Parent – Child : @Input decorator. This decorator help to pass the value from parent to child component.
2. Child – Parent :
   1. @Output: This decorator help to pass the value from child to parent with EventEmitter API provided by Angular.
   2. @ViewChild This decorator is use to access child component properties and behavior in parent component.
3. Sibling :
   1. sessionStorag() and localStorage() : Using html5 and JavaScript.

sessionStorage.setItem(“key”,tokenData);

logout button

sessionStoager.removeItem(“key”)

sessionStorage.getItem(“key”);

* 1. Using Shared Service with module level register using provider attribute.
  2. Using rxjs BehaviourSubject API.

ng g c child3 : create component

ng g c child4 : create component

ng g s shared : create service

share the data using BehaviourSubject

ng g c child5

ng g c child6

ng g c child7

ng g s common

Life cycle methods or hook methods

Angular life cycle or hook methods get called in 3 different phase.

1. Create
2. Changes (state or properties)
3. Destroy

Create only one time at staring

OnInit (interface) -🡪 ngOnInit

AfterContentInit(interface)-🡪 ngAfterContentInit

AfterViewInit(interface)-🡪ngAfterViewInit

Changes : again and again when we do change on property

DoCheck (interface)🡪 ngDoCheck

onChanges (interface)🡪 ngOnChanges : @Input decorator

AfterContentChecked(interface)🡪 ngAfterContentChecked

AfterViewCheched(interface)-🡪 ngAfterViewChecked

Destroy : at last (hide component or close the component).

onDestroy (interface)--🡪ngOnDestroy