**Angular Docs:**

**Introduction:**

**What angular**

1.Angular is a framework which is use to build client side application

2.Great for single page application without reloading the entire page so it is a great choice to if my application is based on JavaScript

**Why angular:**

1.by design angular promotes modular approach and hence the application we build have clear structure.

2.by making use component so this component can be reusable.

3.Angular have inbuilt capability for validation and routing hence development quiker and easier.

4.unit testable

5.google team+micrsoft(Type script)

**Angular history:**

2010 angular js

2016 – angular version 2

2016 -4

2017 -5

2018 – april 6

2018 – oct 7

**Perquisites**

HTML,css,js

Basic TypeScript

**Environment Ready**

Tesxt editor -VS code

Install node js latest one

Node –v

Npm –v(it is making compatible to use Angular library)

Install Angulat CLI:

Visual studio Code as aEditor

Angular CLI:

Command line interface :

The [Angular CLI](https://angular.io/cli) is a command-line tool for managing the Angular development cycle .it allows us to generate basic building block for our angular application.

Installation command is there

<https://cli.angular.io/>



**@component:**

A class with the @[Component](https://angular.io/api/core/Component)() [decorator](https://angular.io/guide/glossary#decorator) that associates it with a companion [template](https://angular.io/guide/glossary#template). Together, the component and template define a [view](https://angular.io/guide/glossary#view). A component is a special type of [directive](https://angular.io/guide/glossary#directive). The @[Component](https://angular.io/api/core/Component)() decorator extends the @[Directive](https://angular.io/api/core/Directive)() decorator with template-oriented features.

An Angular component class is responsible for exposing data and handling most of the view's display and user-interaction logic through [data binding](https://angular.io/guide/glossary#data-binding).

Read more about components, templates, and views.

1.first open the folder in VS code where we wanted to create angular project

In vs code terminal

If ng command is not recognized(

Solution:

1. Right Clicked on My Computer (windows)
2. Selected Advanced System Settings
3. Clicked "Environment Variables"
4. Under "Path" variable, made the FIRST value listed %AppData%\npm

Once I did that I was able to close powershell and reopen and all worked.

Or else use

Use node command line and go specific folder

1.**ng new helloworld**

Go to project i.e Helloworld

**2.ng serve** (for running application) get succefully compiled

**Hit --🡪 localhost:4200**



**Architecture:**

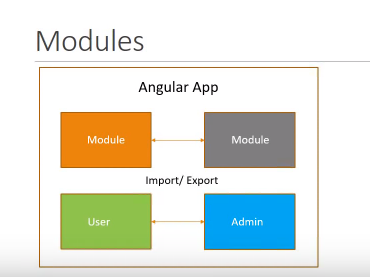
**Modules:**

Angular apps are modular and Angular has its own modularity system called *NgModules*. NgModules are containers for a cohesive block of code dedicated to an application domain, a workflow, or a closely related set of capabilities. They can contain components, service providers, and other code files whose scope is defined by the containing NgModule. They can import functionality that is exported from other NgModules, and export selected functionality for use by other NgModules.

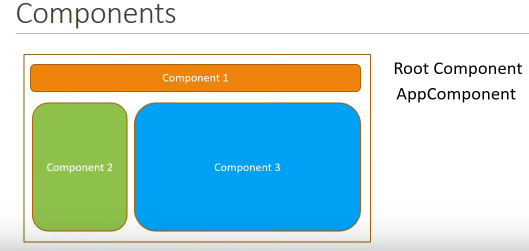
Every Angular app has at least one NgModule class, [the *root module*](https://angular.io/guide/bootstrapping), which is conventionally named AppModule and resides in a file named app.module.ts. You launch your app by *bootstrapping* the root NgModule.

While a small application might have only one NgModule, most apps have many more *feature modules*. The *root*NgModule for an app is so named because it can include child NgModules in a hierarchy of any depth.

An NgModule is defined by a class decorated with @[NgModule](https://angular.io/api/core/NgModule)(). The @[NgModule](https://angular.io/api/core/NgModule)() decorator is a function that takes a single metadata object, whose properties describe the module.



**Components:**

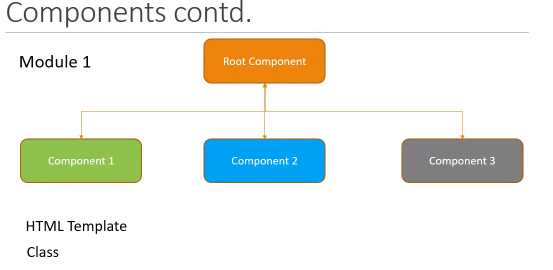


A component controls a patch of screen called a view. For example, individual components define and control each of the following views from the [Tutorial](https://angular.io/tutorial):

* The app root with the navigation links.
* The list of heroes.
* The hero editor.

Each module made up of component and services control a portion of view

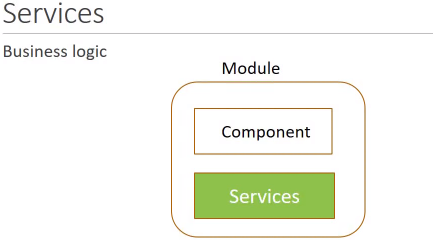
For ex:

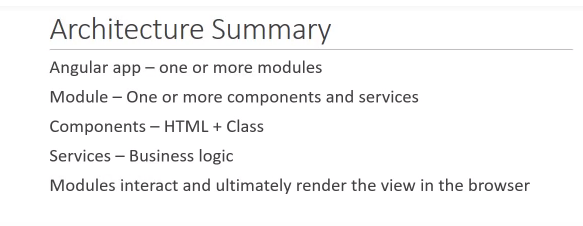


Component 1 :header, Component 2. side bar

**Services:**

**It is a class which contain business logic of our application**





**IN our project**

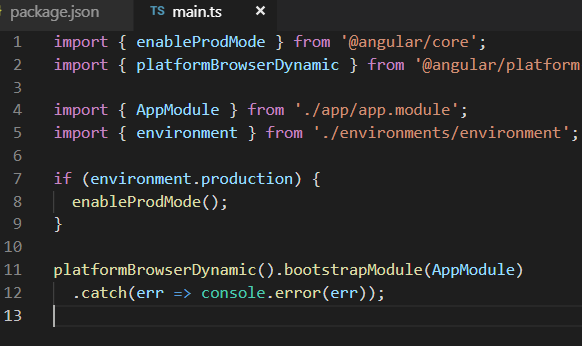
1.Package.json:

Contain all dependency information and script command like ng server which we have use or using

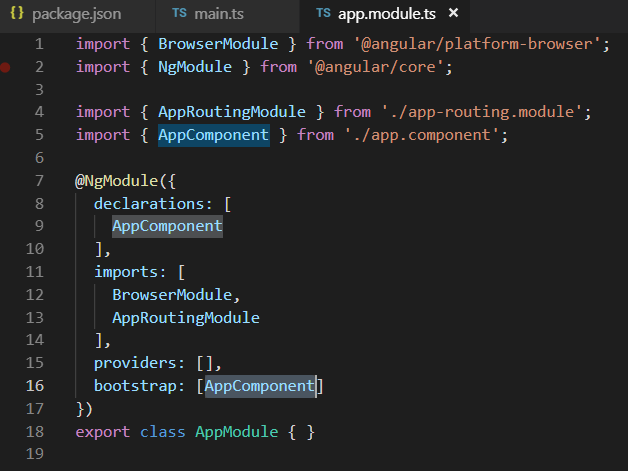
2.Src/app

Main.ts file : it is the entry point of our angular application.

When we run ng server main.ts file get execute and from this file it is bootstrapping the root module i.e AppModule



App.module.ts : it is the root module of our application from this file we are kickstart or bootstrap the AppComponent.

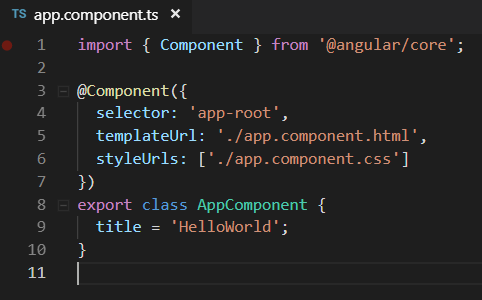


App.component.ts : it is the root component of our application

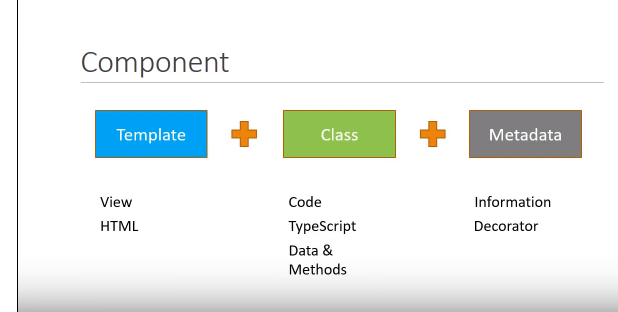
It contain two file

App.component.css,app.component.html (which control the view logic)

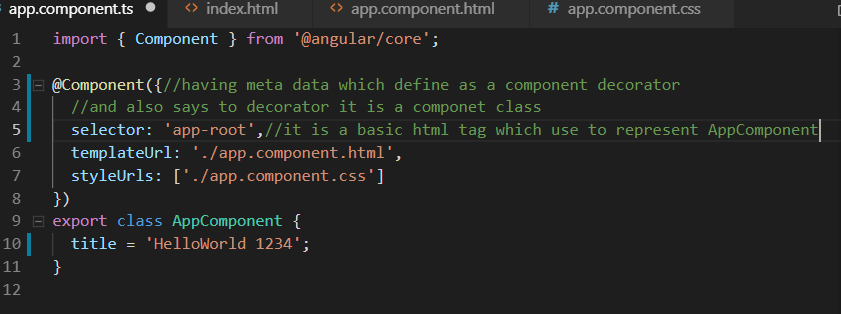
App.compontent.ts (type script contain class which contain logic of view data)



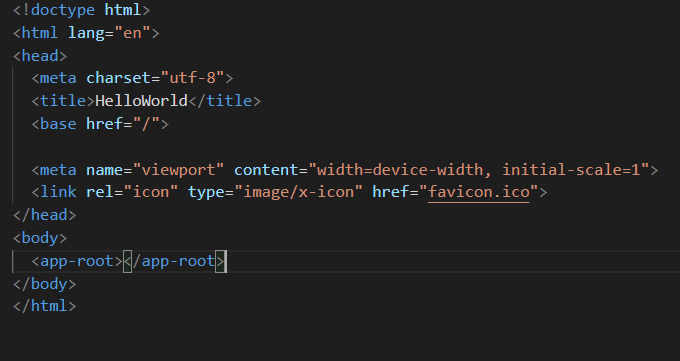
Lecture: 4



.

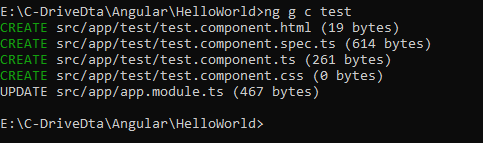


In Index.html we have to give <app-root></app-root> to render root html i.e. app.componenet.html.

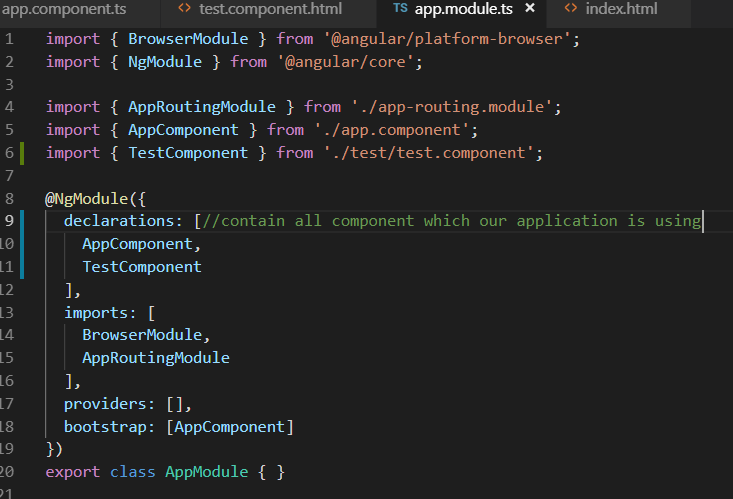


**Create a new component**

**ng g c test**



**Updated to app.module.ts**



App.componet.html- it is a root component html all html will render from here It self here we have to give that particular tag which is define in any component selector

<app-test></app-test>(inside App.componet.html file)

**Selector also we can define like**

**2.Selector : 🡪 .app-test**

**In html we have to include like that way:**

**<div class=”app-test”>**

**</div>**

**3.selector:[app-test]**

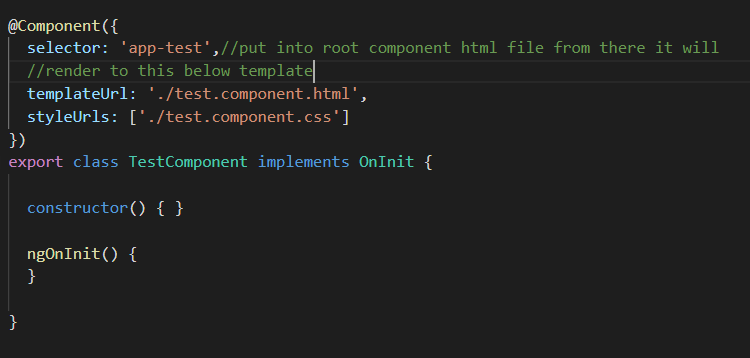
**<div app-test>**

**</div>**

**Template define in following way:**

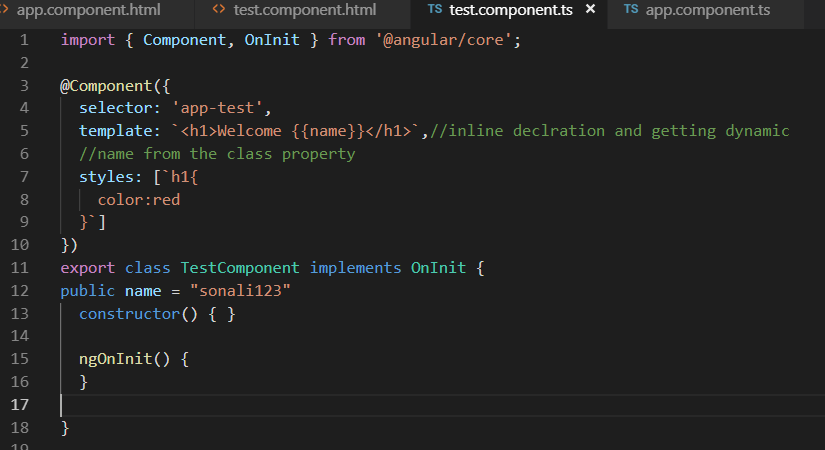
Template: `<div>whwtever we want show ehis is inline template</div>,<div></div>`

Styles:[`div{ color:red}

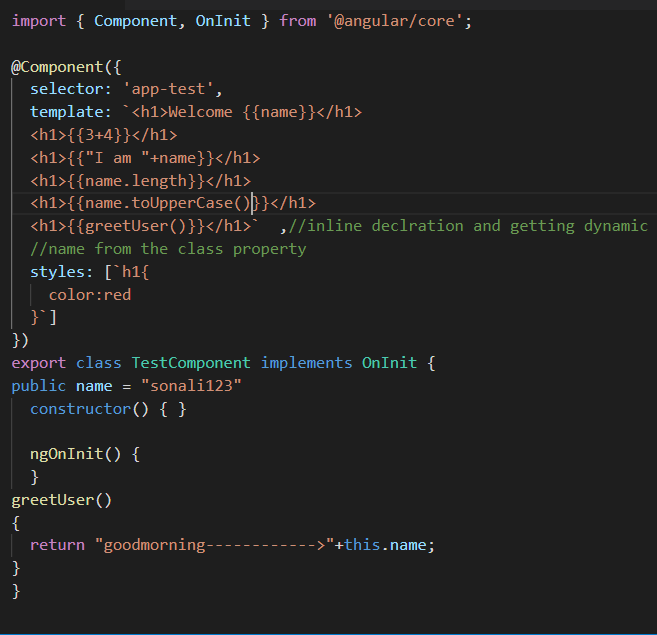


**Lecture -5 : Interpolation**

**Interpolation is binding the class variable to the template**



In interpolation in the {{}} rendering the class property so this is call interpolation so it is binding the data also bind JavaScript methods, and custom method as well which is define below.



Limitation of binding notation i.e. {{}}

1.direct assignment is not possible In {{}}

2.we can’t access global access variable i.e window,console,screen in {{}} beacause it is not aware about this variable

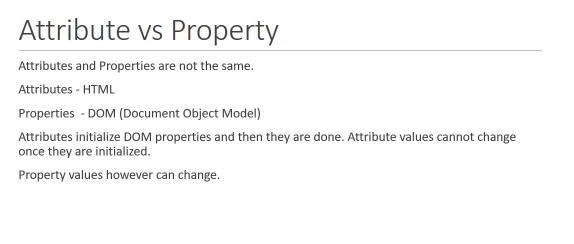
For that we have to perform in the class and bind them to that particular variable

**Lecture:6**

**Property Binding in angular:**

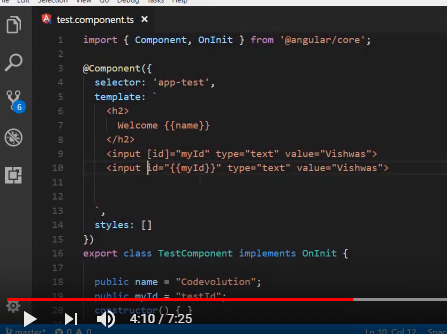
**Property binding Is required because interpolation is not support to bind Boolean value**

**Difference between Html attribute and DOM property:**

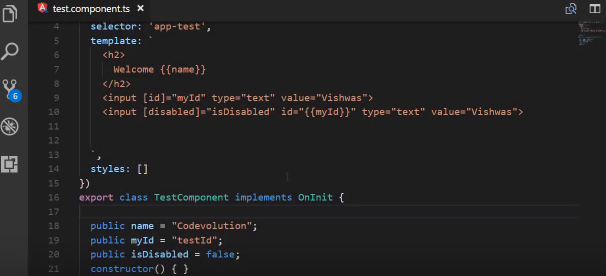


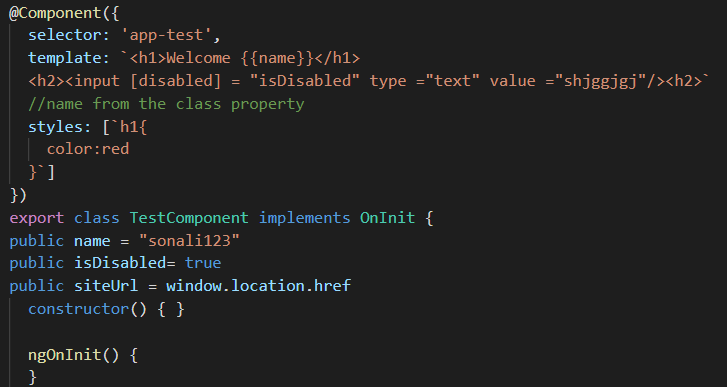
**$0.getAtrribute(“value”)**

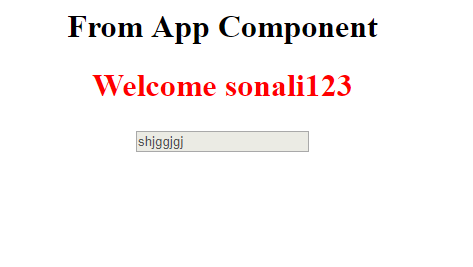
**$0.value = gghg**



**Here we are binding myId property to input html tag**

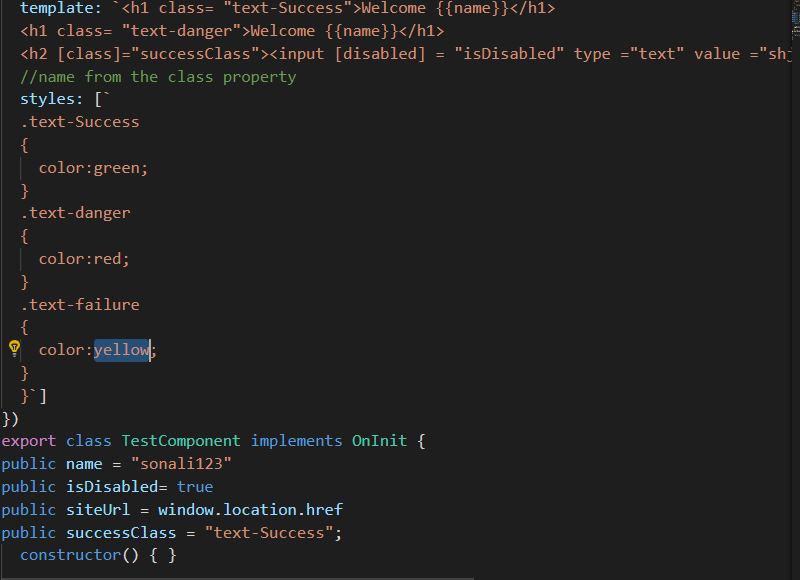






**Lecture 7: Class Binding**

**Apply css by class**



**If we wanted to provide multiple style on a single tag than we have to define attribute like abov is**

**i** public hasError = true;

public messageClass=

{

"text-Success":this.hasError,

"text-special" :this.hasError

}

**To use this In our html tag we have to use [ngClass]=”messageClass”**



**Lecture 8 : Style Binding**

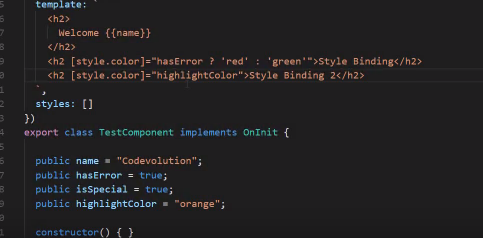
Inline style Binding:



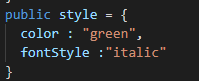
Conditionally apply style:



Binding property define style in html tag:

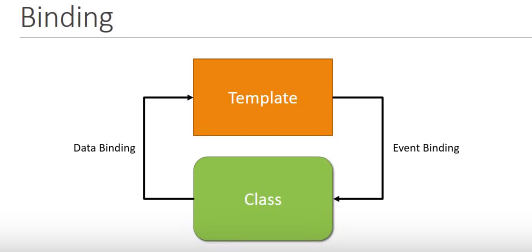


By using ngStyle tag:



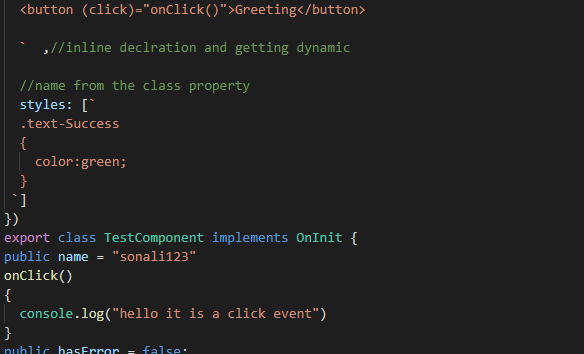


**Lecture 9: Event Binding**

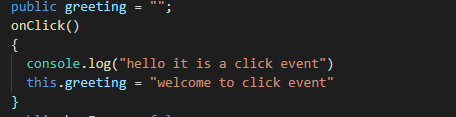


**1.On keyboard event Dom element view is changing**

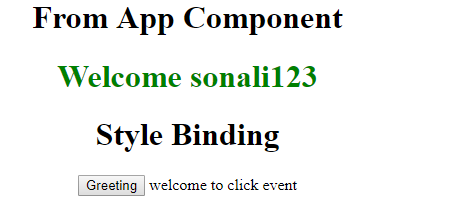
**Click is a event function**



**2.set property on mouse click event**

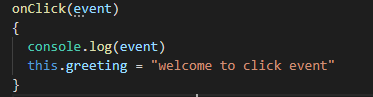


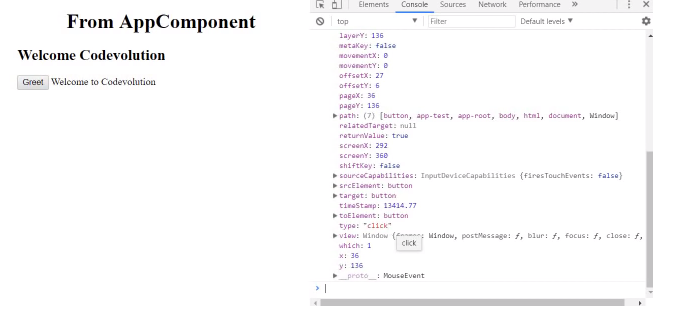




**Some times we required event information with event so have to pass like this way this $ event give us DOM information**





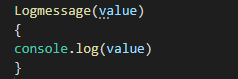


**Lecture - 10 - Template Reference Variables**

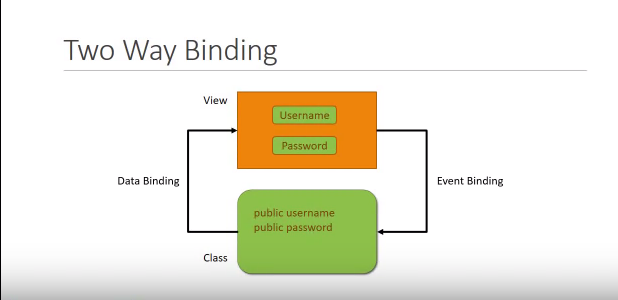
Some times we require to change in our data from view i.e from view to the class by user interaction

For example: some time we want input tag value to the console by clicking on any button here #myinput is define like an id and through myinput.value we are passing the reference variable value



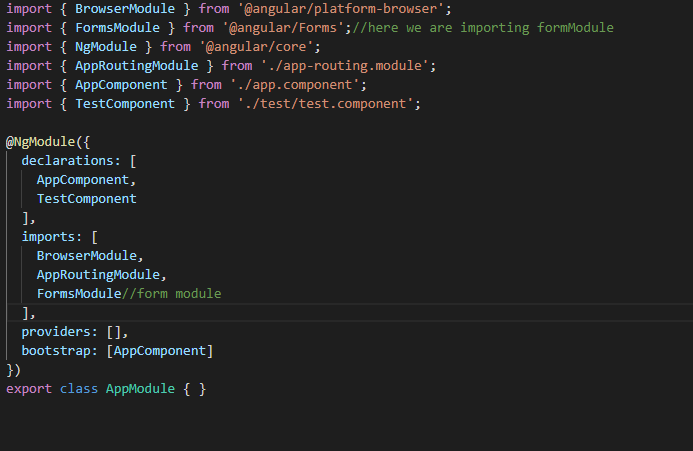


**Lecture - 11 – Two way binding**

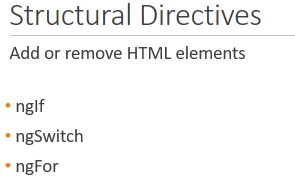


For performing two way binding angular had provide us ngModel directive which is define like this [()] through this directive as we are inserting inside the input box property get update in the model class and through thr interpolation i.e {{}} it is update on view means in the template as well.

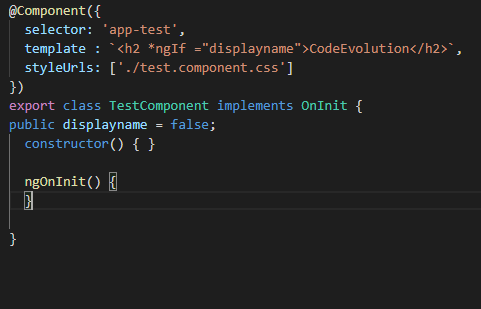
Ngmodel is not directly recognized to the class because it is belong from other model i.e. from FormModel have to import it in app.module.ts

.

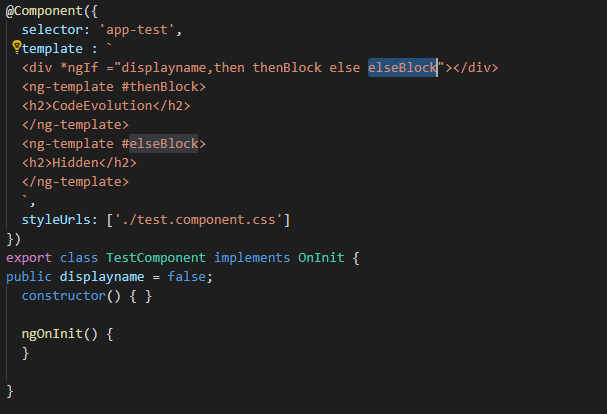
**Lecture - 12 – Ng if Directive**



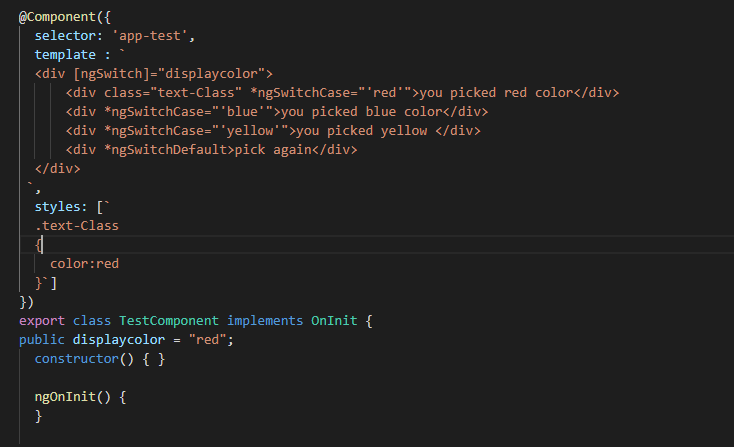
**ngIf : it is a conditional directive**



**ngIf directive for then and else block for performing conditional operation**

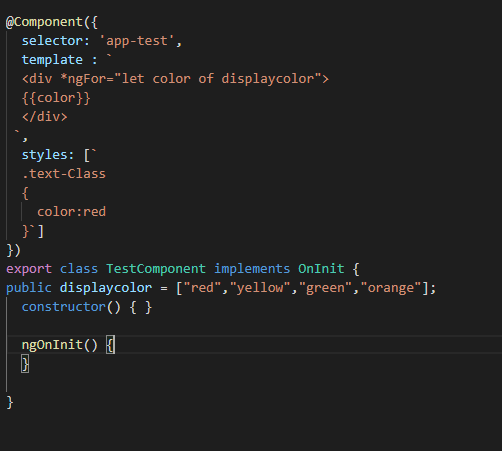


**Lecture - 13 – Ng Swich**



# Lecture - 14 - ngFor Directive

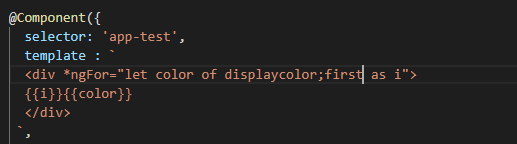
# ngFor Directive we are using for iterating the array elememt



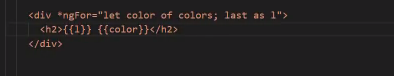
**To Provide indexing we can write like this way**



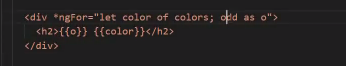
**For checking is this elements is first or not put the condition like this**



**Similarly we can check last**



**For odd or even order**



# Lecture - 15 Component Interaction

# How Component are interact with each other by using @output() and @input() decorator

# 

# How we can use parent component i.e app component property to child component i.e in test component

# 1.App.componenet.ts take one property

# 

# 2.in the test component selector i.e in app.template.html we waill bind parent component property

# 

# 3.we have to receive this perent data property in test component by @input() to inform that it is not a normal property

# 

# For changing property name in child provide aliases as follows

# 

# For sending data from child to parent we have to use events

# Because child component don’t have parent component selector so for that we have to use eventEmitter and @output() as follows.

# Test.componenet.ts

# 

# App.template.html

# 

# Here $event gives us the access of child component event message

# App.component.ts

# Getting access and display in parent componenet

# 

# Lecture - 16 : Pipes in angular

# Pipes perform some inbulid functionality on html element for string property type

# Pipe transform the data only for the view it doesn’t change data for class property

# CODE:

# 

# OutPut:

# 

# Json Conversion:

# 

# Number Pipes

# 

# 

# Lecture - 17 : Services in angular

# 

# 

# 1.Shring data in multiple component class.

# 2.specific operation logic

# 3.database interaction logic

# Lecture - 18 : Dependency injection

# 

# 

# 

# 

# In above code with DI we are passing dependent class as a constructor

# So in case if any thing is changes in dependent class like passing parameter there is no need to change in parent class.

# 

# 

# 

# 

# 

# 1.Create Employee Service

# Ng g s employee

# 

# And have to two componenet

# Ng g c employee

# Ng g c employeeDetails

# 2.register the service in app module so scope will be over the application whatever componenet is register in appModule.ts

# 

# App.module.js

# 

# employeeDetailsComponent.js:

# 

# @injectable : when we require to inject one service to another service then we required @injectable and service class don’t have any decorator as componenet class have @component so that’s y we are keeping @injectable decorator with service class

# Lecture 20 : Http and Observable

# 

# Http : make a request to database or a web server and api to get receve data .

# Observable: it is a reposne from the web server or an api.

# 

# 

# Lecture 21: Http and Observable(practical Implementation fetching data sing Http)

# For make a http request first we have to introduce HttpClient in app.module.ts

# 

# 1.employee.service.ts

# 

# Make a assets/data folder in src folder and make a json file to get employee details.

# 

# Employee.json file

# 

# 2.Http.get method return observable response. So cast the response from observable to employee arraylist

# (a)create a interface

# employee.ts in app folder

# 

# after creating interface employee.service.ts

# 

# EmployeeDetail.componenet.ts

# 

this.\_empoyeeServeice.getEmployee().subscribe(data =>this.employees = data);

# in subscribe method th left hend side data is the argument of the function and right hand side data is the body of the function

# Lecture 22 : Http Error Handling

# For error handling first we have to install

# npm install rxjs-compat –save

# in the service class call catch method as follows:

# employeeService.ts

# 

# In employeeDetailComponent.ts

# 

# If I had given a wrong file name from where we are fetching the data then we find the error as follows.

# 

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# Lecture 23 : Routing and navigation :

# 

# 

# 1.ng g routing-demo –routing

# (a).include base tag in index.html

# 

# (b).app.routing.module.ts

# Contain routing model for our application and configure routing modules

# (c).import approunting module in app.module.ts

# 2.generate departmentList and employeeList componenet

# ng g c departmentList –it –is (for inline template and inline templates in component

# 3.Configure Routes

# We will do that in app.routingmodule.ts

# 

# Module.ts file

# 

# Now where this routing componenet is to display the ans is in router-Outlet directive

# App.componenet.html

# 

# Lecture:24 - Wildcard Route and Redirecting Routes

# Wild card route is usinf if any thing we are passing which routing componenet is not existing then we use wildcard route

# (a)create page-not found component

# 

# Add this component to app-routingModule

# Note : put always at the last in const route :Routes[]

# 

# For empty path url we can make default redirect for any of the component

# As follows:

# But this is not correct way

# 

# This is always go to deparment on any of click

# 

# This will work perfectly:

# 