17 December 2025 09:03

Revision -

Typescript

1. Using interface ,
2. type keyword - unions of types, intersections , declare function prototype
   declare objects similar to interface
   to declare LITERALS
   type gender = "male" | "female" | "other"

3. enums
   enum weekdays{ Monday , Tuesday , wednsday }
4. Generics in typescript
   Formal type parameters
   Actual type parameters

```
class Stack<T>
{
}

let s : Stack<string> = new Stack<string>();
```

Angular 21

Angular command line interface = ng
using ng = ng new project
using ng = we created a new component
ng g c componentname

How the flow takes place

VS code - TS =====> BUILD ===>Deployed on the web server
Browser accesses the webserver === http://localhost:4200
Index.html + main.js loaded from server to client
DOM is created and modified using main.js and RENDERED by the browser

A angular component

TS , VIEW fragment, CSS

Every component is DECORATED by a **@Component** Decorator

selector = specify the TAG name of the Angular CUSTOM tag
Tie the html with the TS class using **templateURL**
Tie the CSS using the **styleURL**
Tie components to each other using **imports :[ ]**

Communicate between the Model and the View

1. Model to View = Interpolation {{ propertyname OR functioncall OR expression, ANY SINGLE VALUE }}
2. Event Binding = handler to handle the event and passed data and also $event
   $event = implicit object - it has all the properties that are related to the current event fired
   e.target - accessed the attributes of the TAG
   pass text values from the VIEW to MODEL
   pass selected value from DROPDOWN VEIW to MODEL

3. Property Binding and Style Binding

```
<TAG    [attribute] = "classproperty OR expression to be evaluated"  />
        [disabled] ="textdisable"
```

[style.color] ="perc < 35 ?'red':'green'"

Template reference variable = the variable that can be used in the VIEW/HTML
We attach the variable to a TAG /HTML_ELEMENT

```
<input  type="text" #inp1  />

{{ inp1.value }}  // Use the variable on the template itself

<button  (click) = "handler(inp1) ">OK </button>
```

EX1 - add a textfiled bound to a template reference variable inp1
Whatever we type in the tf is displayed in the p tag

```
<input type="text"  #inp1  />
<p> YOU ENTERED : {{inp1.value}}</p>
```

PROBLEM : The component is NOT RERENDERED
Because the change detection is not happening
a. No event binding
b. No property change in the component class

As the component is not rerendered we don't see the latest value

---

Angular Framework = MANAGE The COMPONENTS
a. Create the object of the COMPONENT
b. Call the constructor
c. Inject the dependecies
d. Invoke life cycle callbacks
e. Invoke event callbacks
f. Resolve the custom tags

ONLY Component classes are MANAGED by the Angular Framework
HOW does the Angular Framework come to know that a class is a Component !!!
Programmers communicate with the Framework using DECORATORS !!!!
FRAMEWORK inspects the decorated classes and as per the decorators
it will manage the classes , properties , parameters , functions

Decorators are in the TS file
Directives are in the HTML file

Two Way Data Binding = use
Angular Directive = we tell the Framework to perform certain things

event binding = pass from view to model
(click) = ""
property binding = pass from model to view
[disabled] =""

ngModel for two way binding is always written as a banana in the box syntax

```
<input  type="text"  [(ngModel)] ="propertyofclass"   />
```

The current input tag is TIED to the property
if the value of tag changes the property changes
if the property changes then the value of tag changes

We need to include the **FormsModule** in the component that uses ngModel

EX2 - give a textfield which is using ngModel two way databinding
With a property name
Show the name in the p tag
Show a button that will change the name to uppercase

ROUTING in SPA - we always STAY on the index.html
NAVIGATE between VIEWS without RELOADING page

PROVIDER = Library

1. We define ROUTES = we map a URL fragment to the component
2. We will give a MENU = routerLink

---

1. App.routes.ts = define routes
2. Add the Menu in the app.html
3. app.ts }} import RounterLink , RouterOutlet components

Structural Directives = *ngIf , *ngFor , *ngSwitch

They modify the structure of the view !!!

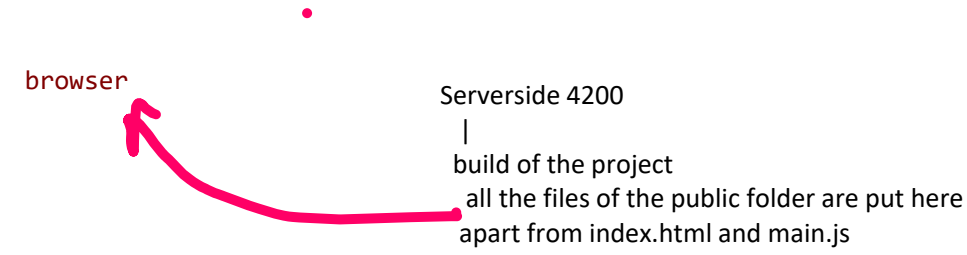
Either you create a component ---
ng g c nameofcomponent

OR

We just write a few small components in a file and use them

Ex3 - show a checkbox , a description with image is present in the <p> tag

When the checkbox is checked show the description
Otherwise hide the p tag



Ex4 - array of student names in the ts
Display the array in the dropdown list
Using *ngFor or @for

Use the for in the HTML

```
<div>
    <!-- (change)="1"  is to prompt RERENDER-->
 <p><input type="checkbox" #c1 (change)="1" />
```

```
    The value of c1 is {{c1.checked}} </p>
 <p *ngIf="c1.checked">
    Twinkle Twinkle Little Star
 </p>

</div>
```

Page = html = VIEW

TS = File = Model File

We create many separate components

A component is a class DECORATED by @Component

Directive - WHERE do we use the directive - TEMPLATE - Tag

1. Attribute Directive = ngModel
2. Structural Directive = *ngIf , *ngFor

Ex - Modify = add a textfield and when I enter a color in the textfield and press enter add it to the list

Use ngModel .

We can use only one structural directive for one tag !!!

If we may want two structural directives for one tag - we can use div tag or <ng-container> tag

<ng-container> is like an invisible tag that does not get added to the DOM

so instead of div we can use <ng-container>

HW -

Write a component to show student details

Create a JSON array =

[{fname: "" , lname:"" , dob:"" , standard:""} , {} ,{} , {}]

Show the array in a html table

Show a serial number as first column it should start with 101

Show headings over each column

Modify to add 4 textfields that will accept 4 values of fname lname dob and standard

On click of addtotable button add the json object to the array .

TRY THE SWITCH CASE SYNTAX In a new component NgSwitchCaseEx

<p [style.color]="paracolor" /> paracolor is a variable whose value should be used

<p [style.color] =" 'red' " /> 'red' is a value to be used

```
<input type="number" #num (change)="1"/>
<div [ngSwitch]="num.value">
    <div *ngSwitchCase="'1'">One</div>
    <div *ngSwitchCase="'2'">Two</div>
    <div *ngSwitchCase="'3'">Three</div>
    <div *ngSwitchDefault>This is Default</div>
</div>
<div [ngSwitch]="num.value">
    <div *ngSwitchCase="x">One</div>
    <div *ngSwitchCase="y">Two</div>
    <div *ngSwitchCase="z">Three</div>
    <div *ngSwitchDefault>This is Default</div>
</div>
```

ADD the following in the Model :

```
x:string="1"
   y:string="2"
   z:string="3"
```

COMMUNICATION between COMPONENTS !!!!

Important to develop REUSABLE COMPONENTS !!!
REUSE ?
Code/UI can be used many times . YES
how to use one component in another component
Add the selector of one component in the VIEW of the other component
+
Import the other component

Parent component REUSES the child component
Can the parent component
send data or receive data from the Reusable /child component?

EX 5 - Create a component Title
We want to pass the text-heading of the title as an attribute

<input type="text" class="s1" />

type is the attribute of the input tag!!
class is the attribute of the input tag !!

<app-counter initial="10" step="5"> </app-counter> }} user defined tag
Initial
Step
Are attributes of the custom tag <app-counter>

MODIFY index.html to have the bootstrap links !!!!

Use attributes on the reusable components
1. Displaymessage
2. Infomessage

collect the attributes as inputs in the reusable components
use the @Input()

For example there would be many Reusable components available to you -
EYButton
EYTextfield
EYDropdown
Normally for UNIFORMITY across teams

You might get LIBRARY of reusable
npm install that_library

How much configuration should be given ?
Only that much that will customize and not change the entire
component structure !!!!

Parent

Reusable component

EX 6 - In the Title component -
add a dropdown of user-choice ----> white , pink , blue
When the user selects a color in the dropdown
---- a message of user's choice of color should be
Passed to the PARENT
Using the message
the PARENT may change the bgcolor of the component

Classes that are Components !!

Classes that are services !!

Services and Components both are managed classes
Managed by the Angular Framework .

---

1. 1. write a service MathsUtil
2. 2. **Inject** the service in the MathComponent
This is a reusable code at TS level

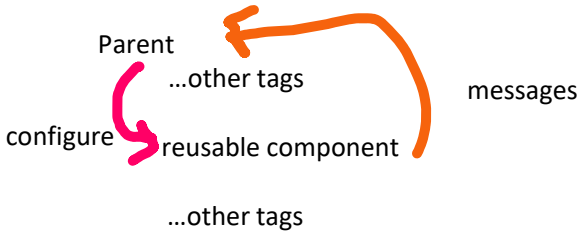
3. WHAT is dependency injection ?
dependency ? THE **PROPERTY OF THE CLASS** that is of type another class

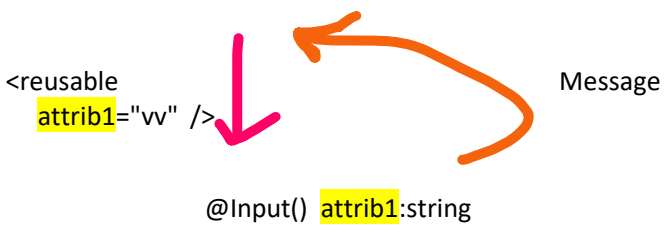
```
class Person
{
    name : string,
    dob : MyDate
    address:  Address

}
```

DI = dependency injection = the dependency will be fulfilled by the
Angular Framework

The Angular Framework will create the objects of dependencies
And set them in the object
This can be DONE IF ONLY IF this condition is fullfilled ???????
The classes of the dependencies are MANAGED by the FRAMEWORK


Parent
…other tags
configure
reusable component
messages
…other tags


<reusable
attrib1="vv" />
Message
@Input() attrib1:string