TS Fundamentals

1. Show how to compile ts file to js file and then run it using node
2. Show how to declare a variable and difference between var and let
3. Variable types and enums
4. Type assertions
5. Arrow functions

ANGULAR FUNDAMENTALS

COMPONENTS

* Creating components
* Create component, register in module, add tag in html
* Create component old way
* Create component using cli

TEMPLATES

* Interpolation data and method

DIRECTIVES

* \*ngFor

SERVICES

* Why use services
* How to write services old way
* Generate using cli
* Dependency injection for services in components

DATA BINDING EVENT HANDLING

PROPERTY BINDING

1. Bind property using interpolation
2. Property binding [ ]
3. Bind property [ ] textContent

ATTRIBUTE BINDING

1. Create table and show how [colSpan] fails
2. Explain difference between DOM and HTML
3. Usually 1-1 mapping but not for all
4. Draw tree of DOM n HTML
5. How to use attr.colSpan for attribute binding

ADDING BOOTSTRAP

1. Add bootstrap with npm
2. Configure bootstrap
3. Show rise in styles size
4. Create a bootstrap button

CLASS BINDING

1. Show [class.active] = “isActive” for button
2. Change true to false and show behavior

STYLE BINDING

1. Want to add inline styles based on condition
2. [style.backgroundColor] = “isActive ? ‘blue’: ‘green’”
3. DOM style object properties

EVENT BINDING

1. Used to handle events in DOM keystorkes, clicks etc
2. (click) = “” demo
3. Event bubbling demo up the DOM tree from inside div
4. $event.stopPropagation to cancel event bubbling

EVENT FILTERING

1. Handle keyup event (keyup) = “onKeyUp()”
2. Check if $event.keyCode === 13 show message
3. (keyup.enter)

TEMPLATE VARIABLES

1. Event.target.value demo with input
2. Replace with #email and email.value

TWO WAY BINDING

1. This.email [(ngModel)] and [value]
2. Show how [(ngModel)] fails and add forms module to success

PIPES

1. Used to format data
2. upperCase, lowerCase, currency, decimal, percent currency:’AUD’ number: 1.1-1
3. Create custom pipe
4. Class implements pipetransform
5. Register pipe

Import Pipe, PipeTransform from @angular/core

@Pipe({

name: ‘summary’

})

Export class SummaryPipe implements PipeTransform {

transform(value: any, args? : any){

if(!value) return null

return value.substr(0,3)+’...’;

}

}

**REUSABLE COMPONENTS**

INPUT PARAMETERS

1. Create child component and pass data using [ ] syntax
2. Show how aliases work

OUTPUT PARAMETERS

1. Show events and child to parent communication
2. Show aliases
3. Passing data in events

NG\_CONTENT

1. Show ng-content with select = “.header”
2. Show ng-container with
3. Tell them how the difference is between ng-content and ng-container

Tell them what view encapsulation is

**DIRECTIVES**

ngIF

1. Show how \*ngIf works
2. Show how \*ngIf= “ courses.length > 0; else noCourses” using ng-template
3. Show how \*ngIf = “courses.length>0; then yesCourses else noCourses” using ng-template
4. Show how [hidden] works
5. Explain difference between ngIf and hidden
6. When to use what

ngSwitchCase

1. Show how [ngSwitch] works with \*ngSwitchCase=” ’map’ ” and \*ngSwitchDefault

ngFor

1. Display list of courses using ngFor
2. Show how to capture index with index as i
3. Demo trackBy

<trackBy: trackCourse>

trackCourse(index, course){

return course ? course.id : undefined;

}

LEADING ASTERISK

1. Show how leading star works with ng-template

<ng-template [ngIf]=”condition”>

ngClass

1. Show how ngClass works

[ngClass]= “{ blue: true, red : false}”

ngStyle

1. Show how ngStyle works

[ngStyle] = “{ color: expression? ‘’ : ‘’, backgroundColor: expression ? ‘option1’ : ‘option2’ }”

Safe traversal operator

1. {{ course.assignee?.name}}

CUSTOM DIRECTIVES

ng g d mobile-format

Import HostListener

@Input(‘format’) format;

@HostListener(‘blur’) onBlur() {

let value = this.el.nativeElement.value;

if(this.format === ‘uppercase’)

this.el.nativeElement.value = value.toUpperCase();

else

this.el.nativeElement.value = value.toLowerCase()

}

<input onInputFormat [format]=”’uppercase’”>

<input [onInputFormat]=”’uppercase’”>

constructor(private el: ElementRef)