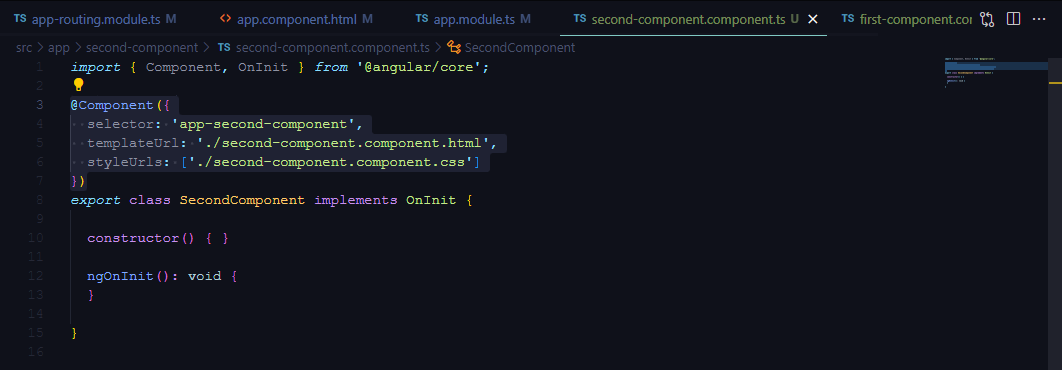
Contents

* Directives
* Pipes



**Directives**

* **What are directives in angular?**
  1. Directives are defined as classes that can add **new behavior** to the elements in the template or modify existing behavior.
  2. Manage the lists, styles, element visible behavior, reusable elements, Class, and Forms.
* **Type of directives?**
  1. **Component Directive**
     + This is a special directive called a component.
     + For this type of directive has a **selector** or **templateUrl**, **styleUrls**.
     + And on output & effect of this directive is that shows something in dom.



* 1. **Structural Directive**
     + Structural directive changes the structure of **DOM** by **adding** and **removing** elements.
     + It is denoted by a \* sign with three pre-defined directives **NgIf, ngIfElse, NgFor**, and **NgSwitch.**
     + **NgIf Directive**
       - **NgIf** directive is used to display or hide data in your application based on the condition becomes true or false.
       - We can add this to any tag in your template.

**<div class=”my-test” \*ngIf=”true”>Display data</div>**

* + - **ngIfElse Directive**
      * **ngIfElse** is similar to **ngIf** except, it provides an option to render content during failure scenario as well.

**<div class=”my-test” \*ngIf=”true else linkToTemplate”>Display data</div>**

**<ng-template #linkToTemplate>Display another data<ng-template>**

* + - **ngFor Directive**
      * **ngFor** is used to repeat a portion of elements from the list of items.
      * Add the list in .ts file.

**public lists: any[] = [1,2,3,4,5];**

* + - * Add the **ngFor** in .html file.

**<ul><li \*ngFor=”let item of lists”>**

**{{item}}**

**</li></ul>**

* + - **ngSwitch Directive**
      * **ngSwitch** is used to check **multiple conditions** and keep the DOM Structure simple and easy to understand.
      * Add the list in the .ts file.

**public userRole: string = ‘Admin’;**

* + - * Add the **ngSwitch** in .html file.

**<ul [ngSwitch]=”userRole”>**

**<li \*ngSwitchCase=”User”>**

**<p> Display Data 1</p>**

**</li>**

**<li \*ngSwitchCase=”Admin”>**

**<p> Display Data 2</p>**

**</li>**

**</ul>**

* 1. **Attribute directive**
     + Attribute directives perform the appearance or behavior of DOM elements or components.
     + Examples of directives are ngStyle, ngClass, and ngModel.
     + **ngStyle Directive**
       - **ngStyle** directive is used to add dynamic style.

**<ul>**

**<li [ngStyle]=”{ ‘color’: role == ‘User’ ? ‘red’ : ‘blue’ }”>**

**<p> Display Data 1</p>**

**</li>**

**</ul>**

* + - **ngClass Directive**
      * **ngClass** is used to add and remove CSS classes in HTML elements.

**<ul>**

**<li [ngClass]=”{ ‘redClass’: role == ‘User’ }”>**

**<p> Display Data 1</p><li></ul>**

* 1. **Custom Directive**
     + When we want to target an element and change its DOM behavior via custom.

**ng g directive custom-style**

* + - Add changes in directive file.ts

**import {Directive, ElementRef} from '@angular/core';**

**@Directive({ selector: '[appCustomstyle]' })**

**export class CustomstyleDirective {**

* + - **In Constructor**

**constructor(el: ElementRef) {**

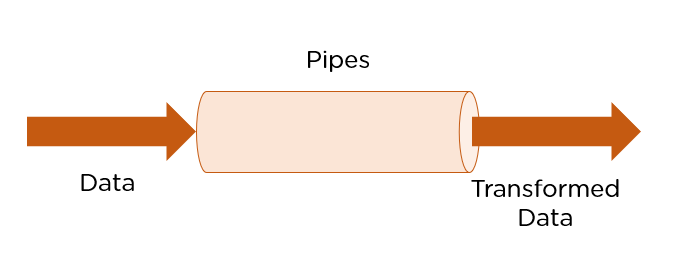
**el.nativeElement.style.fontSize = '24px';**

**}**

* 1. **Component** **Directive**
     + **Component directive** provides **@Input** and **@Output** decorator to **send** and **receive** information between parent and child components.
     + When we want to send the data parent to child on that we use **@Input** decorator.
     + When we want to send any data or emit to the parent component on that time we use **@Output** decorator.

**Pipes**

* **What are pipes in angular?**



* 1. Pipe is used via **@pipe** decorator.
  2. Pipe can take Data and transform the value with the specified format or business logic.
  3. Pipes are defined using the pipe “I” symbol.
  4. Pipes can be provided with acceptable arguments by using the colon (:) sign.
* **Types of Pipes?**
  1. Pure Pipes
     + These pipes use pure functions.
     + Angular calls the pipe only when it detects a change in the parameters being passed.
     + A single instance of the used throughout all components.
     + By Passing the **pure: true**
  2. Examples of pipes.
  3. Impure Pipes
     + These pipes use impure functions.
     + Angular’s every change detection, an impure pipe is called regardless of the change in input fields.
     + By Passing the **pure: false**
* **Build-in Pipes**
  1. Angular provides build-in pipes.
     + **DatePipe**
       - Formats a date value according to locale rules.
     + UpperCasePipe
       - Transforms text to all upper case.
     + LowerCasePipe
       - Transforms text to all lowercase.
     + CurrencyPipe
       - Transforms a number to a currency string, formatted according to locale rules.
     + DecimalPipe
       - Transforms a number into a string with a decimal point, formatted according to locale rules.
     + PercentPipe
       - Transforms a number to a percentage string, formatted according to locale rules.