Learning and Knowledge Management

Angular Pipes

Ground Rules

For a successful class, please:

- Arrive on time.
- Turn off cell phones.
- Assist your colleagues; show respect to all individuals regardless of their skill and knowledge level.
- Do not use class time to surf the net, check e-mail, or use instant messaging.
- Adhere to attendance policy as directed by your local training coordinator.
- Make the most of face-to-face training; ask questions and share your own insights and expertise.
- Leave the room only during break times or in the event of an emergency.
- Manage expectations for your other responsibilities.

Module Objectives

At the end of this module, you should be able to:

- Understand what are Pipes
- Understand how to parameterize a Pipe
- Understand on built-in pipes
- Understand on Pipes chaining
- Understand on how to create Custom Pipes

Topic List

#No	Module Topics
1	Pipes in Angular
2	Built in Pipes
3	Chaining Pipes
4	Custom Pipes

Topic List

#No	Module Topics
1	Pipes in Angular
2	Built in Pipes
3	Chaining Pipes
4	Custom Pipes

Pipes in Angular

What are Pipes?

Pipes helps us to format the data before they are displayed.

- Represented within the two curly braces while displaying the ‘data’ in the html document
- Accepts parameters . Parameters are passed after a colon (:)

Syntax

Colon

data | Pipe name : Pipe Parameters

Pipe Operator

- Angular provides several built-in pipes like
  - Date pipe
  - Slice pipe
  - Json pipe
  - Uppercase pipe etc…
- Also supports in creating custom pipes

Pipes in Angular

Parameterizing Pipes

Pipes can accept any number of optional parameters

- Helps in fine- tuning the output
- To add parameters to a pipe, the pipe name is followed with a colon (:) and then the parameter value
- In case of multiple parameters , separate the values with multiple colons

Example:

currency:'INR'

Currency pipe with single parameter

slice:1:5

Slice pipe with multiple parameters

Topic List

#No	Module Topics
1	Pipes in Angular
2	Built in Pipes
3	Chaining Pipes
4	Custom Pipes

Built-in Pipes

Built-in pipes in Angular

Built- in Pipes

Following are few of the built-in pipes provided by Angular

- Lowercase Pipe
- Uppercase Pipe
- Titlecase Pipe
- Decimal Pipe
- Date Pipe
- Currency Pipe
- Percent Pipe
- Slice Pipe
- Json Pipe

Built-in pipes in Angular

Uppercase , Lowercase and Titlecase Pipes

- Lowercase Pipe
  - Transforms text to lowercase

usage :

```
{{ data | lowercase }}
```

- Uppercase Pipe
  - Transforms text to uppercase

usage :

```
{{ data | uppercase }}
```

- Titlecase Pipe
  - Transforms text to titlecase

usage :

```
{{ data | titlecase }}
```

Built-in pipes in Angular

Decimal Pipe

- Formats a number as text with requires numbers of digits before and after the decimal point
- Specifies the minimum & maximum number of digits after the decimal point.

Syntax

```
        number_expression | number[:digitInfo]
```

where number_expression is a number and

digitInfo is a string which has a following format:

```
'{minIntegerDigits} . {minFractionDigits} - {maxFractionDigits}'
```

<u>Note: enclose the digitinfo in single quotes</u>

minIntegerDigits is the minimum number of integer digits to use. Defaults to 1.

minFractionDigits is the minimum number of digits after fraction. Defaults to 0.

maxFractionDigits is the maximum number of digits after fraction. Defaults to 3.

Built-in pipes in Angular

Currency Pipe

- Formats a number as currency

Syntax

number_expression |currency[:currencyCode[:symbolDisplay[:digitInfo]]]

where

currencyCode

is the ISO 4217 currency code, such as USD for the US dollar , EUR for the euro and INR for Rupees.

symbolDisplay

is a boolean indicating whether to use the currency symbol or code.

true: uses symbol (e.g. $ for USD , ₹ for INR).

false(default) : uses code (e.g. USD , INR).

digitInfo is same as digitinfo in DecimalPipe

Built-in pipes in Angular

Percent and Json Pipes

- Percent Pipe
  - Formats a number as percentage
  - Syntax/rules for “digitInfo” is same as “digitinfo” in DecimalPipe

usage :

number_expression |percent[:digitInfo]

- Json Pipe
  - Converts value into string using JSON.stringify
  - Useful for debugging

usage :

{{ expression | json }}

Built-in pipes in Angular

Date Pipe

- Transforms the given date to the required form
- The date format can be predefined or custom

usage :

date_expression |date[:format]

where **date_expression** is a date object or a number (milliseconds since UTC epoch) or an ISO string

format indicates which date/time components to include

- Predefined date formats

Short form	Format	Example
'medium'	'yMMMdjms'	Sep 3, 2010, 12:05:08 PM
'short'	'yMdjm'	9/3/2010, 12:05 PM
'fullDate'	'yMMMMEEEEd'	Friday, September 3, 2010
'longDate'	'yMMMMd'	September 3, 2010
'mediumDate'	'yMMMd'	Sep 3, 2010
'shortDate'	'yMd'	9/3/2010
'mediumTime'	'jms'	12:05:08 PM

Built-in pipes in Angular

Date Pipe – Custom date formats

COMPONENT	SYMBOL	NARROW	SHORT FORM	LONG FORM	NUMERIC	2-DIGIT
era	G	G (A)	GGG (AD)	GGGG (Anno Domini)	-	-
year	y	-	-	-	y (2015)	yy (15)
month	M	L (S)	MMM (Sep)	MMMM (September)	M (9)	MM (09)
day	d	-	-	-	d (3)	dd (03)
weekday	E	E (S)	EEE (Sun)	EEEE (Sunday)	-	-
hour	j	-	-	-	j (1 PM)	jj (1 PM)
hour12	h	-	-	-	h (1)	hh (01)
hour24	H	-	-	-	H (13)	HH (13)
minute	m	-	-	-	m (5)	mm (05)
second	s	-	-	-	s (9)	ss (09)
timezone	z	-	-	z (Pacific Standard Time)	-	-
timezone	Z	-	Z (GMT-8:00)	-	-	-
timezone	a	-	a (PM)	-	-	-

Built-in pipes in Angular

Slice Pipe

- Slices a given array or string into subset Formats a number as percentage

usage :

array_or_string_expression | slice :start [:end]

- Start : The starting index of the subset to return.
  - a positive integer: return the item at start index and all items after in the list or string expression.
  - a negative integer: return the item at start index from the end and all items after in the list or string.
  - if positive and greater than the size of the expression: return an empty list or string.
  - if negative and greater than the size of the expression: return entire list or string
- End : The ending index of the subset to return.
  - If omitted : return all items until the end.
  - if positive: return all items before end index of the list or string.
  - if negative: return all items before end index from the end of the list or string

Built-in pipes in Angular

Pipes DEMO

product.component.html

```
<h1 align="center"><u>Product Details</u></h1>
<div >
        <table align="center" border=2px>
                <thead>
                        <tr>
                                <td><b>Product Property</b></td>
                                <td><b>Details</b></td>
                        </tr>
                </thead>
        <tbody>
                <tr>
                        <td >Product Id</td>
                        <td >{{"product.productId }}</td>
                </tr>
                <tr>
                        <td>Product Name</td>
                        <td>{{product.productName |
                        uppercase}}</td>
                </tr>
                <tr>
                        <td>ProductPrice</td>
                        <td>{{product.ProductPrice | currency :'INR':true:'5.2'}}</td>
                </tr>
                <tr>
                        <td >Date Of Manufacture</td>
                        <td >{{product.DoM | date:'medium'}}</td>
                </tr>
                <tr>
                        <td>Manufacturer Name</td>
                        <td>{{product.ManufacturerName | titlecase}}</td>
                </tr>
                <tr>
                        <td>Description</td>
                        <td >{{product.Description | lowercase}}</td>
                </tr>
        </tbody>
        </table>
</div>
```

Topic List

#No	Module Topics
1	Pipes in Angular
2	Built in Pipes
3	Chaining Pipes
4	Custom Pipes

Chaining Pipes

Chaining Pipes

What is Data Binding?

- Angular allows us to chain pipes
- Pipe operator “|” is used to apply more than one pipe to an expression

Example :

- To display the birthday in uppercase, the birthday is chained to the DatePipe and on to the UpperCasePipe

```
{{ birthday | date | uppercase}}
```

- The birthday displays as Jun2 , 1934.

Topic List

#No	Module Topics
1	Pipes in Angular
2	Built in Pipes
3	Chaining Pipes
4	Custom Pipes

Custom Pipes

Custom Pipes

- Angular allows us to create custom pipes
- A pipe is a class decorated with pipe metadata
- Pipe class implements the **PipeTransform** interface
- The transform() method has to be overridden from the interface
- It accepts an input value followed by optional parameters and returns the transformed value
- **@Pipe** decorator tells Angular that it is a Pipe, which you import from the core Angular library
- **@Pipe** decorator allows the developer to define the pipe name that will be used within template expressions
- To create a custom pipe, execute the command
  - **ng generate pipe <PipeName>**

Example : ng g pipe MyPipe

- Two files **my-pipe.pipe.ts** and **my-pipe.pipe.spec.ts** would be created

Custom Pipes

Custom Pipe - DEMO

The following pipe transforms a given name into a name with title "Mr." or "Ms".

Example: “Watson” | myPipe: 'male'

app.component.ts

```
import { Component } from '@angular/core';
@Component({
        selector: 'app-root',
        template: `<div *ngFor="let emp of
        employees">
        Name : {{emp.name | myPipe: emp.gender}}
        </div>`
})
export class AppComponent {
        PageHeader:String = 'Employee Details
        Page';
        employees =
        [{name:"Watson",gender:"male"},

        {name:"Diana",gender:"female"}];
}
```

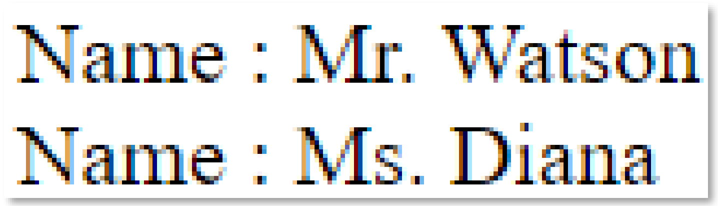
Custom Pipes

Custom Pipe – Demo & Output

my-pipe.pipe.ts

```
import { Pipe, PipeTransform } from
'@angular/core';
@Pipe({
      name: 'myPipe'
})
export class MyPipePipe implements
PipeTransform {
transform(name: string, gender:string): string {
      if (gender=="Male" || gender == "male")
            return "Mr. "+name;
      else
            return "Ms. "+name;
      }
}
```

Output



```
Name : Mr. Watson
Name : Ms. Diana
```

Module Summary

Now, you should be able to:

- Understand what are Pipes
- Understand how to parameterize a Pipe
- Understand on built-in pipes
- Understand on Pipes chaining
- Understand on how to create Custom Pipes

Reference

Heading	Description
Pipes	Link

Thank You