**Angular Pipes**

Pipes in Angular are a powerful feature that allows you to transform data in your templates. They are easy to use and can significantly simplify your code. In this chapter, we'll cover the basics of pipes in Angular and provide some examples.

**What are Pipes?**

Pipes are simple functions that you can use in template expressions to accept an input value and return a transformed value. Pipes are useful because they define transformations that can be easily reused across your application.

**Built-in Pipes**

Angular comes with several built-in pipes. Let's look at a few examples:

**1. Uppercase Pipe**

import { Component } from '@angular/core';

@Component({

selector: 'app-root',

template: `

<h2>{{ 'hello world' | uppercase }}</h2>

`

})

export class AppComponent { }

Output:

HELLO WORLD

**2. Date Pipe**

import { Component } from '@angular/core';

@Component({

selector: 'app-root',

template: `

<p>Today is {{ today | date:'fullDate' }}</p>

`

})

export class AppComponent {

today = new Date();

}

Output (assuming today is September 1, 2024):

Today is Sunday, September 1, 2024

**3. Currency Pipe**

import { Component } from '@angular/core';

@Component({

selector: 'app-root',

template: `

<p>The product costs {{ price | currency:'USD' }}</p>

`

})

export class AppComponent {

price = 29.99;

}

Output:

The product costs $29.99

**Custom Pipes**

You can also create your own custom pipes. Let's create a simple pipe that reverses a string:

import { Pipe, PipeTransform } from '@angular/core';

@Pipe({

name: 'reverse',

standalone: true

})

export class ReversePipe implements PipeTransform {

transform(value: string): string {

return value.split('').reverse().join('');

}

}

Now you can use this pipe in your component:

import { Component } from '@angular/core';

import { ReversePipe } from './reverse.pipe';

@Component({

selector: 'app-root',

template: `

<p>{{ 'Hello Angular' | reverse }}</p>

`,

imports: [ReversePipe],

standalone: true

})

export class AppComponent { }

Output:

ralugnA olleH

**Pipe Parameters**

Pipes can also accept parameters. Let's modify our reverse pipe to optionally capitalize the result:

import { Pipe, PipeTransform } from '@angular/core';

@Pipe({

name: 'reverse',

standalone: true

})

export class ReversePipe implements PipeTransform {

transform(value: string, capitalize: boolean = false): string {

let result = value.split('').reverse().join('');

return capitalize ? result.toUpperCase() : result;

}

}

Now we can use it like this:

import { Component } from '@angular/core';

import { ReversePipe } from './reverse.pipe';

@Component({

selector: 'app-root',

template: `

<p>{{ 'Hello Angular' | reverse }}</p>

<p>{{ 'Hello Angular' | reverse:true }}</p>

`,

imports: [ReversePipe],

standalone: true

})

export class AppComponent { }

Output:

ralugnA olleH

RALUGNA OLLEH

**Conclusion**

Pipes in Angular are a powerful and flexible way to transform data in your templates. Whether you're using built-in pipes or creating your own custom pipes, they can help make your code more readable and maintainable.