Pipes en Angular

Manejo de Pipes en Angular

1. ¿Qué es un Pipe en Angular?

Un **Pipe** en Angular es una función pura que se utiliza para transformar datos directamente en la plantilla sin necesidad de modificar las clases que contienen la información de nuestra aplicación. Los Pipes ayudan a formatear, calcular o manipular datos de manera sencilla y declarativa en el HTML de un componente.

Por ejemplo, puedes usar un Pipe para convertir texto a mayúsculas, formatear una fecha, o modificar la visualización de números en las plantillas. Angular provee varios Pipes integrados, pero también puedes crear tus propios Pipes personalizados.

2. Sintaxis Básica

La sintaxis de los Pipes en Angular es muy simple. Utilizan el símbolo de barra vertical `|` para aplicar un Pipe sobre una expresión en la plantilla:

```
{{ expression | pipeName }}
```

Puedes aplicar múltiples Pipes en una misma expresión:

```
{{ expression | pipe1 | pipe2 }}
```

Además, algunos Pipes aceptan parámetros, que se pasan después del nombre del Pipe separados por `:`:

```
{{ expression | pipeName:arg1:arg2 }}
```

3. Pipes Integrados en Angular

Angular ofrece una serie de Pipes listos para usar que cubren muchas necesidades comunes:

- **DatePipe**: Formatea fechas.

```
{{ today | date }} <!-- Fecha actual en formato predeterminado -->
{{ today | date:'short' }} <!-- Fecha en formato corto -->
```

- **UpperCasePipe y LowerCasePipe**: Convierte texto a mayúsculas o minúsculas.

```
{{ 'angular' | uppercase }} <!-- Muestra: ANGULAR -->
```

- **CurrencyPipe**: Formatea números como una moneda.

```
{{ 12345 | currency:'USD' }} <!-- Muestra: $12,345.00 -->
```

- **DecimalPipe**: Formatea números decimales.

```
{{ 3.14159 | number:'1.2-2' }} <!-- Muestra: 3.14 -->

Resumen del formato '1.2-2':
• 1 dígito mínimo para la parte entera.
• 2 dígitos mínimos después del punto decimal.
• 2 dígitos máximos después del punto decimal (Angular redondea si hay más de 2 dígitos).
```

- **PercentPipe**: Convierte números a porcentajes.

```
{{ 0.25 | percent }} <!-- Muestra: 25% -->
```

4. Ejemplo Listado de Empleados usando pipes

Vamos a crear un ejemplo para mostrar un listado de empleados, aplicando diferentes Pipes para mejorar la presentación de los datos.

Creamos el componente de ejemplo-pipes:

ng g c ejemplo-pipes --skip-tests

HTML: Archivo ejemplo-pipes.component.html

```
<div class="container my-4">
    <h3 class="text-center text-primary">Lista de Empleados</h3>
    <ul class="list-group w-50 mx-auto">
        @for (empleado of empleados; track empleado) {
        <li class="list-group-item d-flex justify-content-between align-items-center">
            <div>
                <h5 class="mb-1">{{ empleado.nombre | uppercase }}</h5>
                <small class="text-muted">
               Fecha Nacimiento: {{ empleado.fechaNacimiento | date:'longDate' }}</small>
            </div>
            <span class="badge bg-success">
                {{ empleado.sueldo | currency:'USD' }}
            </span>
        </li>
        }
    </ul>
</div>
```

TypeScript: Archivo ejemplo-pipes.component.ts

```
import { CommonModule } from '@angular/common';
import { Component, LOCALE_ID } from '@angular/core';
import { registerLocaleData } from '@angular/common';
import localeEs from '@angular/common/locales/es';

// Registrar los datos de localización para español
registerLocaleData(localeEs, 'es');

@Component({
  selector: 'app-ejemplo-pipes',
  standalone: true,
  imports: [CommonModule],
  providers: [{ provide: LOCALE_ID, useValue: 'es' }], // Configurar idioma español
  templateUrl: './ejemplo-pipes.component.html',
  styleUrl: './ejemplo-pipes.component.css'
})
export class EjemploPipesComponent {
  empleados = [
    { nombre: 'Ricardo Suarez', sueldo: 12000, fechaNacimiento: new Date('2000-08-01') },
    { nombre: 'Laura Mejía', sueldo: 7000, fechaNacimiento: new Date('1988-09-05') },
    { nombre: 'Gilberto Anaya', sueldo: 8000, fechaNacimiento: new Date('1995-07-12') }
  ];
```

```
}
```

En este ejemplo:

- **UpperCasePipe**: Convierte el nombre del empleado a mayúsculas.
- **CurrencyPipe**: Formatea el precio en dólares estadounidenses.
- **DatePipe**: Muestra la fecha de nacimiento del empleado en formato corto.
- **Importante**: Es necesario importar el Módulo CommonModule para que puedan funcionar los pipes sin problemas.
- **Importante**: Además, si queremos mostrar la fecha con textos en español es necesario registrar el proveedor de internacionalización (locale) en español (es) usando la función registerLocaleData.

Resultado Final:

5. Creación de un Pipe Personalizado

Además de los Pipes integrados, Angular permite crear Pipes personalizados para realizar transformaciones específicas que no están cubiertas por los Pipes predeterminados.

Aquí te muestro cómo crear un Pipe personalizado que convierte un texto en la primera letra en mayúscula:

Paso 1: Crear el Pipe

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

@Pipe({
  name: 'capitalize'
})
export class CapitalizePipe implements PipeTransform {
  transform(value: string): string {
```

```
    if (!value) return '';
    return value.charAt(0).toUpperCase() + value.slice(1).toLowerCase();
  }
}
```

Paso 2: Uso del Pipe en la plantilla

Una vez creado, puedes usarlo como cualquier otro Pipe en tu plantilla:

```
<p>{{ 'angular' | capitalize }}</p> <!-- Muestra: Angular -->
```

6. Consideraciones sobre Pipes

- **Pipes Puros**: La mayoría de los Pipes en Angular son "puros", lo que significa que solo se recalculan si la entrada cambia. Esto mejora el rendimiento, ya que no se recalculan en cada cambio de estado del componente.
- **Pipes Impuros**: Los Pipes pueden ser declarados como "impuros" si necesitan recalcularse en cada cambio de estado (por ejemplo, cuando la referencia de un objeto no cambia pero sus propiedades internas sí). Para crear un Pipe impuro, simplemente define `pure: false` en su decorador `@Pipe`:

```
@Pipe({
  name: 'pipeImpuro',
  pure: false
})
export class PipeImpuro implements PipeTransform {
  transform(value: any): any {
    // lógica del pipe
  }
}
```

7. Conclusión

Los Pipes en Angular son una herramienta poderosa que te permite manipular y transformar datos directamente en la vista sin necesidad de modificar el código de tu componente. Con el uso de Pipes integrados o personalizados, puedes crear una presentación de datos mucho más rica y amigable para el usuario.

Saludos!

Ing. Ubaldo Acosta

Fundador de GlobalMentoring.com.mx