47 - Directiva de atributo [ngClass] y directivas para clases individuales - Sintaxis completa

En el concepto anterior vimos las diferentes herramientas que nos provee Angular para manipular los estilos en línea. Ahora veremos que Angular nos permite manipular las clases que se asocian a un elemento HTML.

Podemos crear un proyecto para ir probando estas funcionalidades:

```
ng new proyecto032
```

La directiva de atributo [ngClass] permite establecer una o más clases a un elemento HTML.

Veamos distintas formas de utilizar la directiva [ngClass]

Si tenemos definidas las siguientes clases en el archivo 'app.component.css':

```
.clase1 {
    color:red;
    background-color:blue;
}

.clase2 {
    font-size:20px;
    font-family: 'Courier New', Courier, monospace;
}
```

Luego en la vista 'app.component.html':

```
<p [ngClass]="'clase1 clase2'">Prueba de directiva ngClass</p>
```

Se le asignan las clases 'clase1' y 'clase2' al párrafo. Son importante las comillas simples para que funcione. Normalmente no haríamos esto ya que es más sencillo fijar el atributo 'class' de HTML:

```
<p class="clase1 clase2">Prueba de class</p>
```

Una alternativa más útil es definir un objeto literal cuyas claves son nombres de clases y sus valores un valor boolean que indica si se debe activar o no la clase al elemento HTML.

Si tenemos los atributos:

```
  estado1=true;
  estado2=false;
```

Luego en la vista definimos la directiva de atributo [ngClass]:

```
<p [ngClass]="{'clase1':estado1,'clase2':estado2}">Prueba de directiva ngClass</p>
```

Luego solo se activa la 'clase1' ya que 'estado1' almacena true, debido a que 'estado2' almacena false significa que la 'clase2' no se aplica a la etiqueta.

Como vemos podemos en forma dinámica cambiar variables definidas en el modelo y se aplican los cambios gracias a la directiva [ngClass]

Disponer una condición en la expresión

Por ejemplo si necesitamos que una tabla que muestra los números del 1 al 5, tengan las celdas de colores alternos, lo podemos resolver con la sintaxis:

```
<table>
  @for(elemento of [1,2,3,4,5];track $index) {
  <tr>
    <td [ngClass]="{'color1':elemento%2==0,'color2':elemento%2==1}">{{elemento}}</td>
  </tr>
  }
</table>
```

Suponiendo que las dos clases están definidas en el archivo 'app.component.css':

```
.color1 {
    color: white;
    background-color: blue;
}

.color2 {
    color: white;
    background-color: green;
}
```

Cada vuelta del for solo se aplica una de las dos clases.

Vamos a agregar a la aplicación otra funcionalidad que permita mediante dos botones fijar o eliminar dos estilos que se aplican a un elemento HTML.

app.component.css

```
.clase1 {
  color:red;
  background-color:blue;
}

.clase2 {
  font-size:20px;
  font-family: 'Courier New', Courier, monospace;
}

.color1 {
  color: white;
  background-color: blue;
}

.color2 {
  color: white;
  background-color: green;
}

.clase3 {
  color:red;
  background-color:blue;
}

.clase4 {
  font-size:20px;
  font-family: 'Courier New', Courier, monospace;
}
```

app.component.html

```
<p [ngClass]="'clase1 clase2'">Prueba de directiva ngClass</p>
<p [ngClass]="{'clase1':estado1,'clase2':estado2}">Prueba de directiv
a ngClass</p>

<table>
  @for(elemento of [1,2,3,4,5];track $index) {
  <tr>
    <td [ngClass]="{'color1':elemento%2==0,'color2':elemento%2==1}">{
{elemento}}</td>
  </tr>
  }
</table>

<p [ngClass]="forma">Prueba de directiva</p>
<button (click)="fijar()">Fijar clases</button>
<button (click)="eliminar()">Eliminar clases</button>
```

app.component.ts

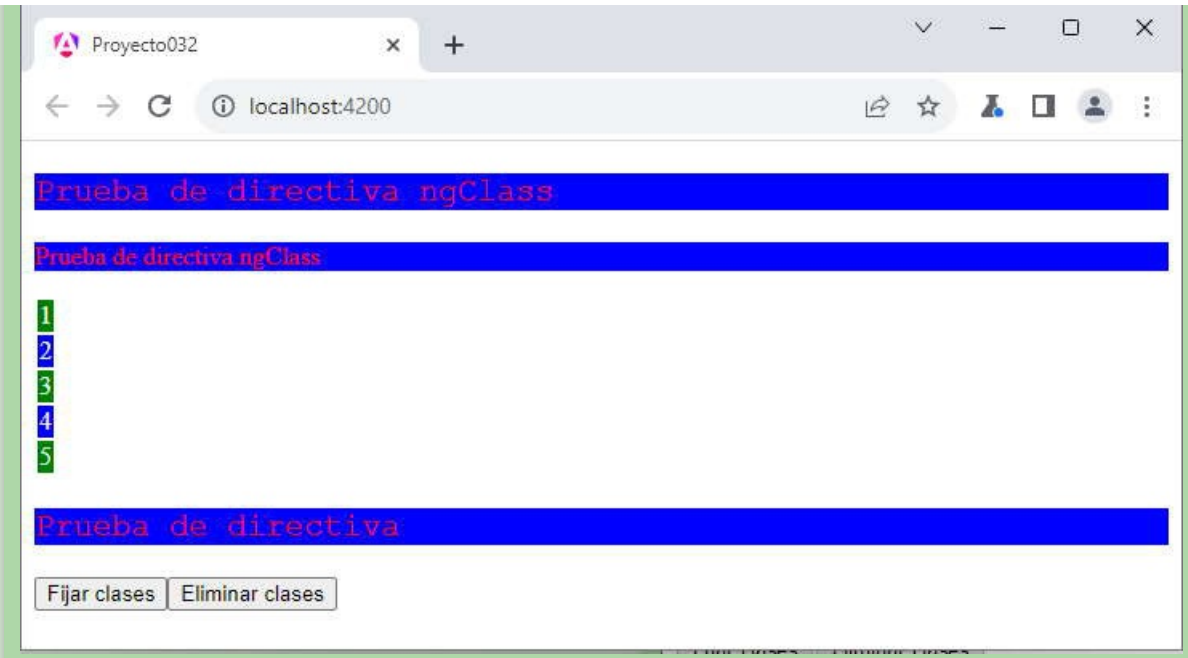
```
import { Component } from '@angular/core';
import { RouterOutlet } from '@angular/router';
import { NgClass } from '@angular/common';

@Component({
  selector: 'app-root',
  imports: [RouterOutlet, NgClass],
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
  estado1 = true;
  estado2 = false;

  forma = {
    'clase3 clase4': true
  };
  fijar() {
    this.forma['clase3 clase4'] = true;
  }

  eliminar() {
    this.forma['clase3 clase4'] = false;
  }
}
```

Se asocia la variable 'forma' a la directiva 'ngClass', luego significa que cuando comienza la ejecución de la aplicación las dos clases se asocian a dicha etiqueta:



Luego si se presiona el botón de 'eliminar', modificamos el objeto literal asignando el valor false, esto hace que las dos clases se eliminen de la etiqueta.

Enlace a clases individuales.

En lugar de utilizar la directiva 'ngClass' podemos utilizar una sintaxis alternativa para agregar o eliminar clases en particular para una etiqueta HTML:

```
<p [class.clase1]="true">Prueba de class</p>
```

Se añade las 'clase1' al parrafo. En lugar de disponer un true podemos definir una condición o llamar a una función que retorne true o false:

```
<table>
  @for(elemento of [1,2,3,4,5];track $index) {
  <tr>
    <td [class.clase1]="elemento%2==0">{{elemento}}</td>
  </tr>
  }
</table>
```

Con el código anterior solo se incorpora la 'clase1' a las filas pares de la tabla. Si la etiqueta ya dispone de otras clases no se borran las mismas

Podemos probar esta aplicación en la web aquí.

Retornar