5 - Sintaxis de Template para estructuras condicionales y repetitivas: @if / @else - @for - @switch/@case/@default

Angular usa @if para expresar visualizaciones condicionales en plantillas. La plantilla o template @if nos permiten condicionar si se deben agregar o no bloques de código.

La plantilla @for nos permite generar muchos elementos HTML repetidos a partir del recorrido de un arreglo de datos.

Para analizar con un ejemplo estas plantillas procederemos nuevamente a modificar el proyecto001.

En el archivo 'app.component.ts' procedemos a codificar la clase AppComponent con la definición de:

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

@Component({
  selector: 'app-root',
  imports: [RouterOutlet],
  templateUrl: './app.component.html',
  styleUrl: './app.component.css'
})
export class AppComponent {
  nombre = 'Rodriguez Pablo';

  edad = 40;

  articulos = [{
    codigo: 1,
    descripcion: 'naranjas',
    precio: 540
  }, {
    codigo: 2,
    descripcion: 'manzanas',
    precio: 900
  }, {
    codigo: 3,
    descripcion: 'peras',
    precio: 490
  }];

  generarNumero() {
    return Math.floor(Math.random() * 3) + 1;
  }
}
```

Hemos definido las propiedades nombre, edad y articulos:

```
nombre = 'Rodriguez Pablo';

edad = 40;

articulos = [{
  codigo: 1,
  descripcion: 'naranjas',
  precio: 540
},{
  codigo: 2,
  descripcion: 'manzanas',
  precio: 900
},{
  codigo: 3,
  descripcion: 'peras',
  precio: 490
}];
```

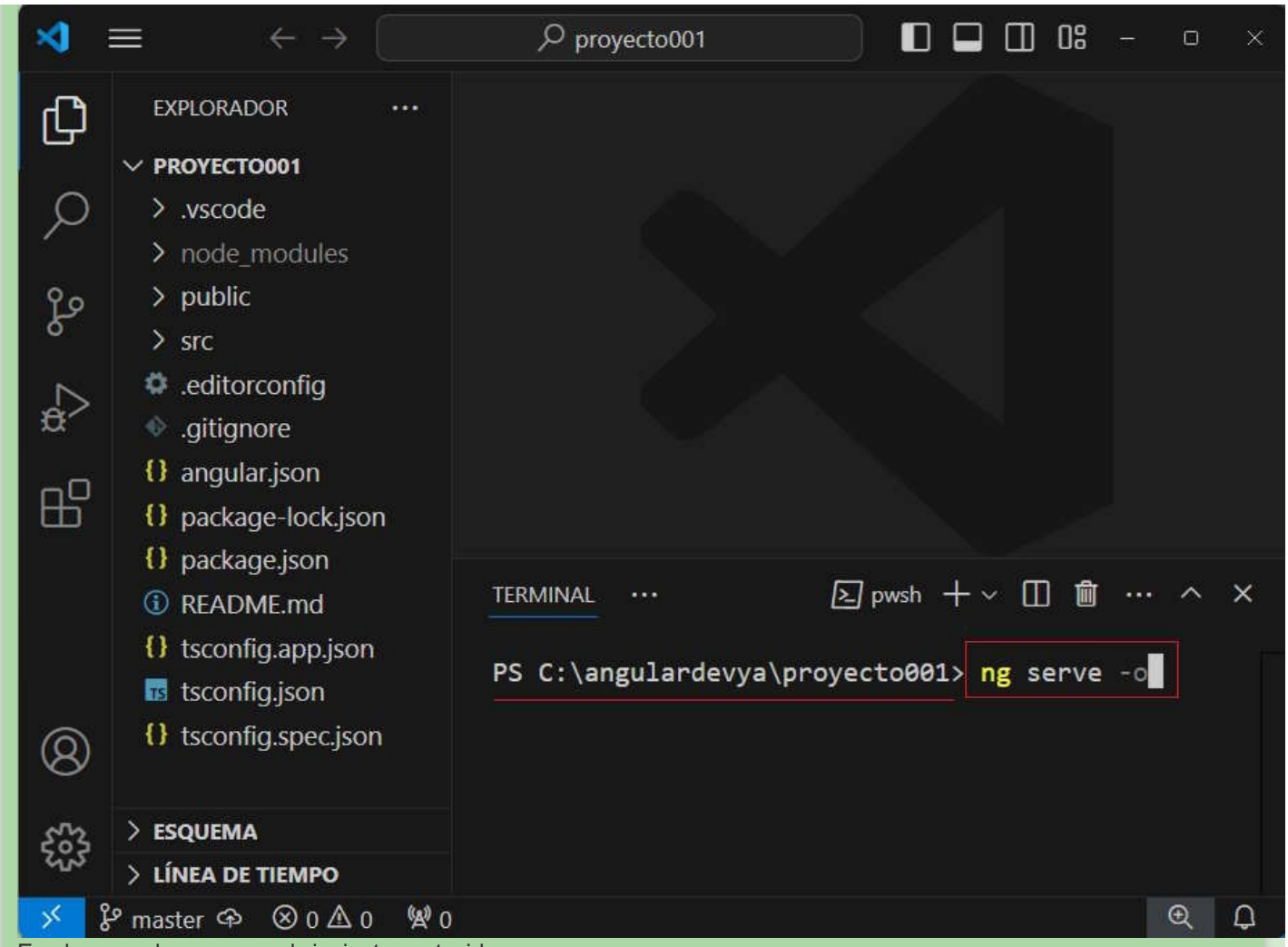
Por otro lado un método que retorna un valor aleatorio comprendido entre 1 y 3:

```
generarNumero() {
  return Math.floor(Math.random() * 3) + 1;
}
```

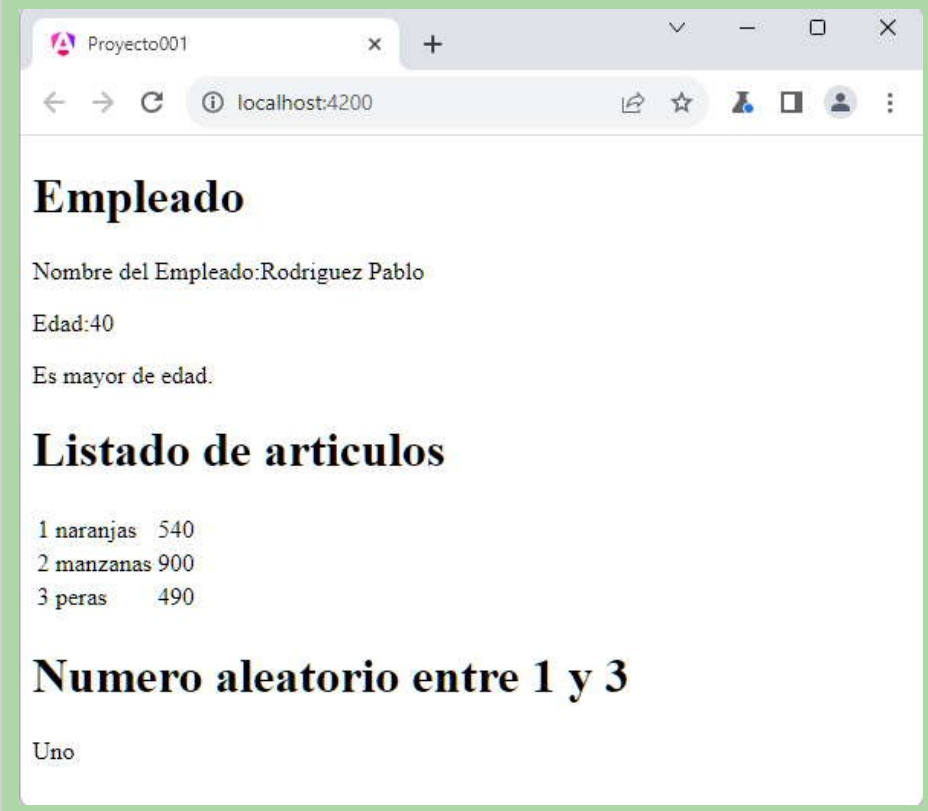
Ahora procedemos a modificar el archivo app.component.html:

```
<div>
  <h1>Empleado</h1>
  <p>Nombre del Empleado:{{nombre}}</p>
  <p>Edad:{{edad}}</p>
  @if (edad>=18) {
  <p>Es mayor de edad.</p>
  } @else {
  <p>Es menor de edad.</p>
  }
  <h1>Listado de articulos</h1>
  <table>
    @for(articulo of articulos; track articulo.codigo) {
    <tr>
      <td>{{articulo.codigo}}</td>
      <td>{{articulo.descripcion}}</td>
      <td>{{articulo.precio}}</td>
    </tr>
    }
  </table>
  <h1>Numero aleatorio entre 1 y 3</h1>
  @switch (generarNumero()) {
    @case (1) {
      <p>Uno</p>
    }
    @case (2) {
      <p>Dos</p>
    }
    @case (3) {
      <p>Tres</p>
    }
  }
</div>
<router-outlet />
```

Ejecutemos nuestra aplicación desde la línea de comandos de Node.js:



En el navegador aparece el siguiente contenido:



La instrucción @if verifica la condición, en el caso de verificarse verdadero se inserta el bloque contenido entre las llaves:

```
@if (edad>=18) {
<p>Es mayor de edad.</p>
} @else {
<p>Es menor de edad.</p>
}
```

Luego si la condición se verifica falsa se ejecuta el bloque definido en el @else. Podemos probar cambiando la edad por un valor menor a 18.

La instrucción @for nos genera posiblemente muchos elementos HTML repetidos, en este ejemplo una serie de filas de una tabla HTML:

```
  @for(articulo of articulos; track articulo.codigo) {
  <tr>
    <td>{{articulo.codigo}}</td>
    <td>{{articulo.descripcion}}</td>
    <td>{{articulo.precio}}</td>
  </tr>
  }
```

En cada repetición en la variable 'articulo' se almacena un objeto del arreglo 'articulos'. De esta forma podemos mostrar los datos del objeto respectivo.

Es importante agregar la sentencia track al @for indicando un valor único en cada vuelta del @for (es común utilizar un id o clave)

Por último también disponemos la instrucción @switch donde llamamos al método 'generarNumero' y según el valor retornado entrará en el @case respectivo:

```
@switch (generarNumero()) {
  @case (1) {
    <p>Uno</p>
  }
  @case (2) {
    <p>Dos</p>
  }
  @case (3) {
    <p>Tres</p>
  }
}
```

Retornar