UTN.BA

UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Programador Web Avanzado

Clase N° 11: Pipes – directivas - Módulos

Profesor: Ing. Leandro Rodolfo Gil Carrano
Email: leangilutn@gmail.com

UTN.BA

UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Pipes

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Pipes

Los pipes me sirven para mostrar con un formato determinado los datos en pantalla.

Pueden encontrar la documentación de los Pipes en el manual de Angular en

https://angular.io/guide/pipes

Los pipes disponibles son:

- currency
- date
- uppercase
- json
- limitTo
- lowercase
- async
- decimal
- percent

Pipes

Uppercase

```
{{ heroe.nombre | uppercase}}
```

Date (solo se visualizara el año de la fecha)

```
{{ heroe.aparicion | date: 'y' }}
```

Combinación de ambos pipes

```
{{  fecha | date:'fullDate' | uppercase}}
```

El resultado será el siguiente

```
MONDAY, SEPTEMBER 17, 2012
```

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Pipes

Slice

El pipe slice nos permite cortar la longitud de una variable. Por ejemplo en este caso si indicamos slice:3 cortara las tres primeras letras del nombre.

```
<tr>
      <td>{{ nombre }}</td>
      <td> slice:3 </td>
      <td>{{ nombre | slice:3}}</td>
</tr>
```

Si queremos dejar solo las primeras 3 letras debemos hacer

```
<tr>
      <td>{{ nombre }}</td>
      <td> slice:0:3 </td>
      <td>{{ nombre | slice:0:3}}</td>
</tr>
```

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Pipes

Decimal

El pipe decimal me sirve para mostrar un número en un formato en partícular. Si bien el pipe se llama decimal en realidad se utiliza con la palabra number.

```
<tr>
  <td>{{ PI }}</td>
  <td> number </td>
  <td>{{ PI | number }}</td>
</tr>
<tr>
  <td>{{ PI }}</td>
  <td> number:'3.1-5' </td>
  <td>{{ PI | number:'3.1-5' }}</td>
</tr>
<tr>
  <td>{{ PI }}</td>
  <td> number:'3.0-0' </td>
  <td>{{ PI | number:'3.0-0' }}</td>
</tr>
<tr>
  <td>{{ PI }}</td>
  <td> number:'1.0-2' </td>
  <td>{{ PI | number:'1.0-2' }}</td>
</tr>
```

3.141592653589793	number	3,142
3.141592653589793	number:'3.1-5'	003,14159
3.141592653589793	number:'3.0-0'	003
3.141592653589793	number:'1.0-2'	3,14

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Pipes

Currency

Es utilizado para mostrar formatos de moneda.

```
<tr>
  <td>{{ salario }}</td>
  <td> currency </td>
  <td>{{ salario | currency }}</td>
</tr>
<tr>
  <td>{{ salario }}</td>
  <td> currency:'EUR' </td>
  <td>{{ salario | currency:'EUR' }}</td>
</tr>
<tr>
  <td>{{ salario }}</td>
  <td> currency:'EUR':true:'4.0-0' </td>
  <td>{{ salario | currency:'EUR':true:'4.0-0' }}</td>
</tr>
```

2553.5	currency	2.553,50 $
2553.5	currency:'EUR'	2.553,50 €
2553.5	currency:'EUR':true:'4.0-0'	2.554 €

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Pipes

Currency

Sirve para trabajar con promesas. Muestra el valor en forma asincrónica una vez que recibe la información.

Controlador

```
valorPromesa = new Promise ((resolve,reject)=>{
  setTimeout( ()=>resolve('Llego la data'),3500 );
});
```

Vista

```
<tr>
  <td>{{ valorPromesa }}</td>
  <td> async </td>
  <td>{{ valorPromesa | async }}</td>
</tr>
```

Resultado

[object Promise]	async	Llego la data

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Pipes

Generar un nuevo pipe

```
PS C:\sites\angular\aplicacion> ng generate pipe
? What name would you like to use for the pipe? capitalize
```

Se crear un nuevo archivo .pipe dentro del directorio app

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Pipes

Capitalize.pipe.ts

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

@Pipe({
  name: 'capitalize'
})
export class CapitalizePipe implements PipeTransform {

  transform(value: string, todas:boolean=true ): string {
    value = value.toLowerCase();

    let nombres = value.split(" ");

    if (todas){
      for ( let i in nombres ) {
        nombres [i] = nombres[i][0].toUpperCase() + nombres[i].substr(1);
      }
    }else{
      nombres[0] = nombres[0][0].toUpperCase() + nombres[0].substr(1);
    }


    return nombres.join(" ");
  }

}
```

Se debe definir:

- Parámetros de entrada
  Con tipo de dato
- Tipo de dato de retorno
- Valor retornado

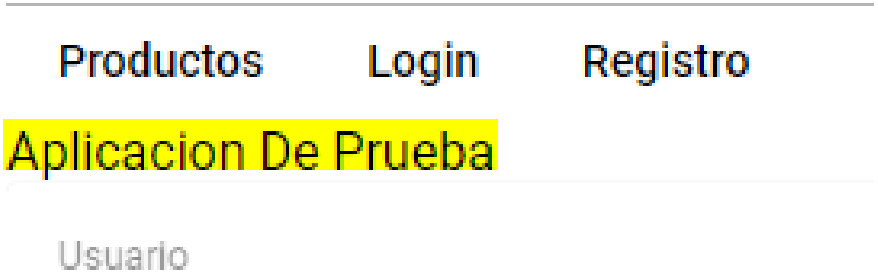
El pipe el llamado de acuerdo al campo “name” de su decorator

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Pipes

Llamado desde la vista

```
<div>{{title | capitalize}}</div>
<router-outlet></router-outlet>
<div>
    footer
</div>
```



UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Módulos

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Módulos

Un módulo es una reunión de componentes y otros artefactos como directivas o pipes.

Los módulos nos sirven principalmente para organizar el código de las aplicaciones Angular y por tanto debemos aprender a utilizarlos bien.

Un módulo es uno de los elementos principales con los que podemos organizar el código de las aplicaciones en Angular.

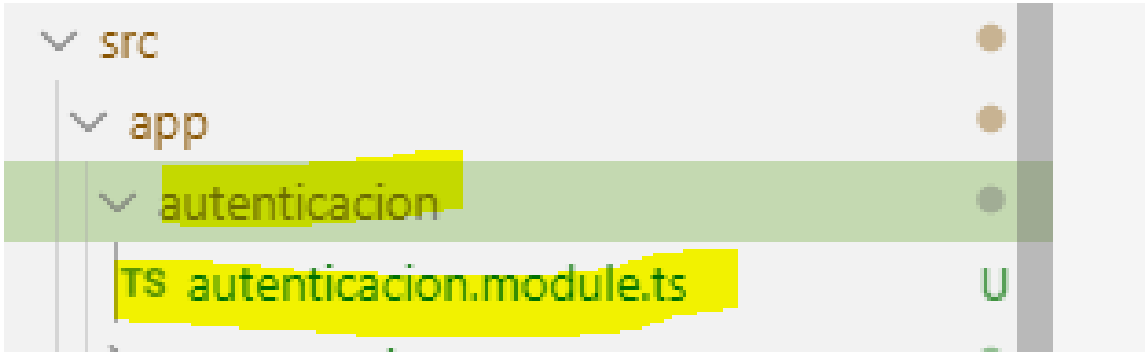
En lugar de colocar el código de todos los componentes, directivas o pipes en el mismo módulo principal, lo adecuado es desarrollar diferentes módulos y agrupar distintos elementos en unos u otros. El orden se realizará de una manera lógica, atendiendo a nuestras propias preferencias, el modelo de negocio o las preferencias del equipo de desarrollo

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Generar nuevo modulo

```
PS C:\sites\angular\aplicacion> ng generate module
? What name would you like to use for the NgModule? autenticacion
```

Generara un nuevo directorio con el nombre del modulo generado.
Dentro se ubicara el archivo **.module.ts** correspondiente



Generar nuevo modulo

Si abrimos el código del módulo generado "nombre.module.ts", encontraremos cómo se define un módulo en Angular. La parte más importante es, como ya viene siendo habitual en Angular, un decorador.

```
import { NgModule } from '@angular/core';
import { CommonModule } from '@angular/common';

@NgModule({
  declarations: [],
  imports: [
    CommonModule
  ]
})
export class AutenticacionModule { }
```

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Generar nuevo modulo

En el decorador, tienes de momento un par de arrays definidos:

- imports: con los imports que este módulo necesita
- declarations: con los componentes, u otros artefactos que este module construye.

Podemos especificar otro módulo donde crearlos, mediante el comando:

```
PS C:\sites\angular\aplicacion> ng generate component autenticacion/testComponent
```

Generar nuevo modulo

Esto nos generará una carpeta dentro del módulo indicado, en la que colocará todos los archivos del componente recién creado. Es decir el css, html, ts y spec.ts

Pero además, el comando del CLI también modificará el código del módulo, agregando automáticamente el import del componente y su referencia en el array "declarations".

Ahora el código del módulo “autenticacion.module.ts” tendrá una forma como esta:

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Generar nuevo modulo

El componente se genera dentro del directorio del modulo

En el modulo se importa el componente en declarations

```
import { NgModule } from '@angular/core';
import { CommonModule } from '@angular/common';
import { TestComponentComponent } from './test-component/test-component.component';

@NgModule({
  declarations: [TestComponentComponent],
  imports: [
    CommonModule
  ]
})
export class AutenticacionModule { }
```

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Generar nuevo modulo

Exportar el modulo

Más adelante, si queremos que este módulo exponga cosas hacia afuera, que se puedan llegar a utilizar desde otros módulos, tendremos que agregar una nueva información al decorador del módulo: el array de exports.

```
@NgModule({
  declarations: [TestComponentComponent],
  imports: [
    CommonModule
  ],
  exports: [
    TestComponentComponent
    ]

})
```

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Generar nuevo modulo

Importar el modulo

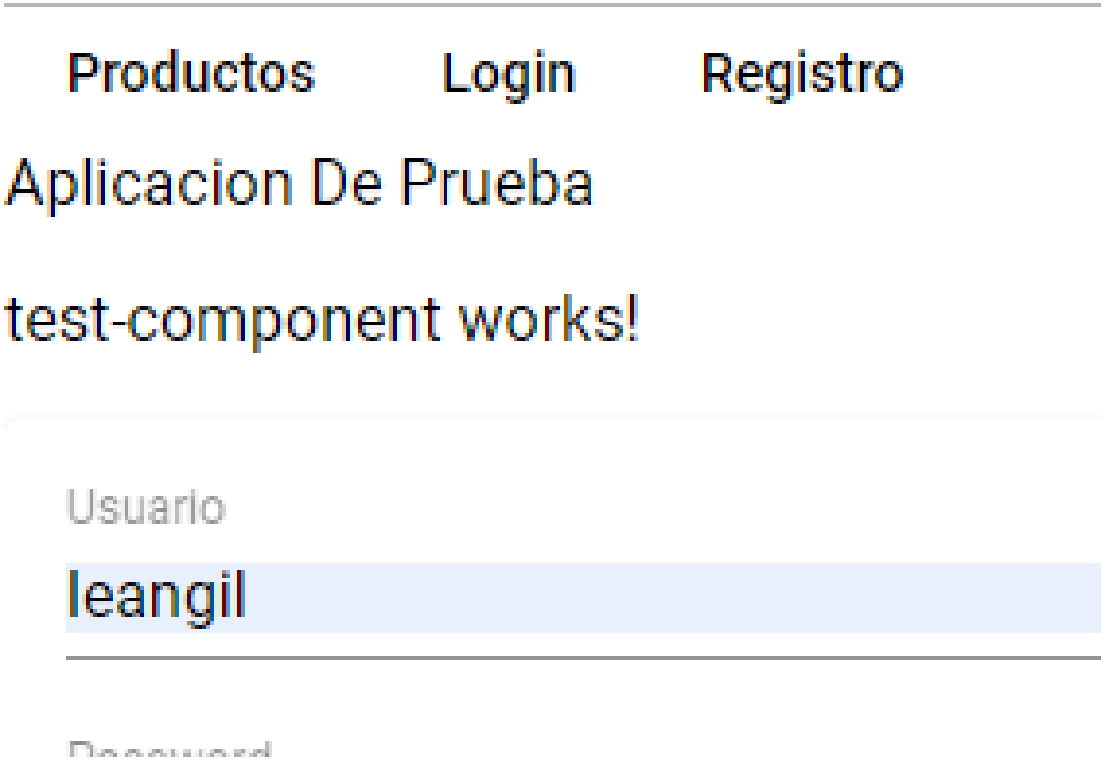
Para utilizar este modulo debemos importarlo.

Ejemplo desde **app.module.ts**

```
import { AutenticacionModule } from './autenticacion/autenticacion.module';
@NgModule({
  declarations: [
    AppComponent,
    RegistroComponent,
    LoginComponent,
    ProductosComponent,
    ProductosDetalleComponent,
    ProductoComponent,
    CapitalizePipe
  ],
  imports: [
    BrowserModule,
    AppRoutingModule,
    FormsModule,
    ReactiveFormsModule,
    HttpClientModule,
    BrowserAnimationsModule,
    MatButtonModule, MatCheckboxModule, MatMenuModule, MatInputModule,MatCardModule,MatSnackBarModule,
    AutenticacionModule
```

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Generar nuevo modulo



UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Directivas

ngSwitch

ngSwitch Directiva estructural

```
<div [ngSwitch]="alerta">
  <div *ngSwitchCase="'success'">success</div>
  <div *ngSwitchCase="'info'">info</div>
  <div *ngSwitchCase="'warning'">warning</div>
  <div *ngSwitchDefault>default</div>
</div>
```

En base al valor de la variable alerta se renderizara el elemento Correspondiente

Al igual que *ngIf o *ngFor modifica la estructura del DOM

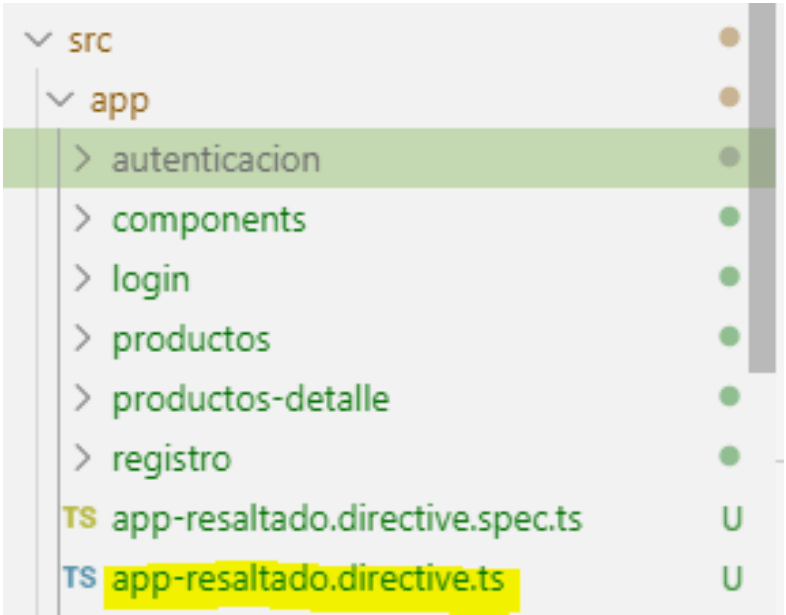
UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Generar nueva directiva

Ejecutar por consola

```
PS C:\sites\angular\aplicacion> ng generate directive
? What name would you like to use for the directive? appResaltado
```

Se genera el archivo **.directive.ts** en la raíz del directorio src



UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

```
import { Directive, ElementRef, HostListener, Input } from '@angular/core';

@Directive({
  selector: '[appResaltado]'
})
export class AppResaltadoDirective {

  constructor( private el:ElementRef ) {
    //console.log("Directiva llamada");
    //el.nativeElement.style.backgroundColor = "yellow";
  }

  @Input("appResaltado") nuevoColor:string;

  @HostListener('mouseenter') mouseEntro(){
    //console.log(this.nuevoColor);
    this.resaltar( this.nuevoColor || 'yellow' );
    //si eliminamos la funcion resaltar comentamos su llamado y descomentamos
    //this.el.nativeElement.style.backgroundColor = "yellow";
  }
  @HostListener('mouseleave') mouseSalio(){
    this.resaltar( null );
    //si eliminamos la funcion resaltar comentamos su llamado y descomentamos
    //this.el.nativeElement.style.backgroundColor = null;
  }

  private resaltar( color:string ){
    this.el.nativeElement.style.backgroundColor = color;
  }

}
```

Llamado desde la vista

```
<p [appResaltado]="'orange'">
  Hola Mundo
</p>
```

Hola Mundo