**TRABAJAR CON FORMULARIOS ANGULAR**

* Vamos a visualizar cómo podemos trabajar de forma nativa con formularios en Angular.
* Para trabajar con Formularios, será necesario declarar el módulo de formularios dentro de la aplicación para que funcionen los binding.



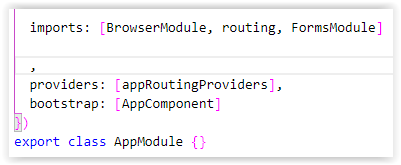
Lo primero que vamos a realizar es la declaración del módulo de formularios en la app.

Abrimos el archivo **app.module.ts**

**APP.MODULE.TS**



Indicamos que utilizaremos el módulo en la app dentro de la zona de **imports**:



A continuación, nos creamos un nuevo component llamado **Formulario**

**FORMULARIO.COMPONENT.TS**

import { Component, OnInit } from "@angular/core";

@Component({

  selector: "app-formulario",

  templateUrl: "./formulario.component.html",

  styleUrls: ["./formulario.component.css"]

})

export class FormularioComponent implements OnInit {

  public user: any;

  public mensaje: string;

  constructor() {

    this.user = {

      nombre: "",

      apellidos: "",

      edad: 0

    };

    this.mensaje = "";

  }

  ngOnInit(): void {}

  onSubmit() {

    this.mensaje = "Datos recibidos correctamente";

    console.log("Enviado");

    console.log(this.user);

  }

}

**FORMULARIO.COMPONENT.HTML**

<div style="float:left">

  <h1>Formularios</h1>

  <h2 style="color:blue">{{ mensaje }}</h2>

  <form #userForm="ngForm" (ngSubmit)="onSubmit()">

    <label for="nombre">Nombre: </label>

    <input

      type="text"

      name="nombre"

      #nombre="ngModel"

      [(ngModel)]="user.nombre"

    />

    <br />

    <label for="apellidos">Apellidos: </label>

    <input

      type="text"

      name="apellidos"

      #apellidos="ngModel"

      [(ngModel)]="user.apellidos"

    />

    <br />

    <label for="edad">Edad: </label>

    <input type="number" name="edad" #edad="ngModel" [(ngModel)]="user.edad" />

    <br />

    <button>Enviar datos</button>

  </form>

</div>

<div style="float:left" \*ngIf="user.nombre">

  <h2>Datos del formulario</h2>

  <ul>

    <li>Nombre: {{ this.user.nombre }}</li>

    <li>Apellidos: {{ this.user.apellidos }}</li>

    <li>Edad: {{ this.user.edad }}</li>

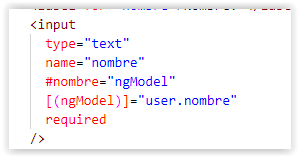
  </ul>

</div>

Ahora vamos a visualizar cómo podríamos validar los campos del formulario con Angular.

Vamos a incluir la validación para un campo obligatorio en **nombre**.

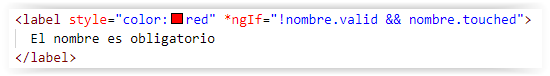
Para ello, incluimos la etiqueta HTML5 de required sobre la caja nombre:



Para validar con angular, tenemos un atributo **valid** sobre los controles **form** que nos indicará si son válidos.

**objetoForm.valid**

Entonces preguntamos si el nombre es válido y además, si ha sido tocado.



Podemos incluir etiquetas de HTML5 de todo tipo, como por ejemplo, **pattern**

    <input

      type="text"

      name="nombre"

      #nombre="ngModel"

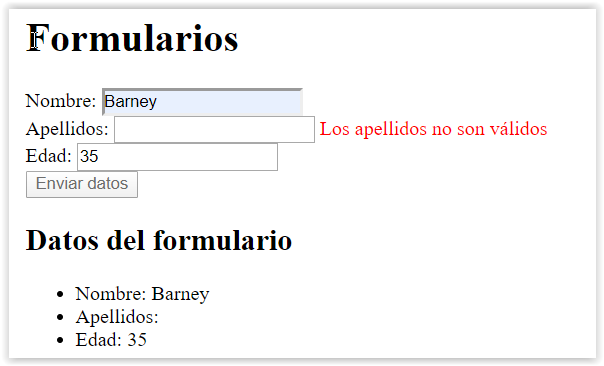
      [(ngModel)]="user.nombre"

      pattern="[A-Z]{1}[a-z]{2,}"

      required

    />

Y visualizaremos el resultado



**FORMULARIO.COMPONENT.HTML**

<div style="float:left">

  <h1>Formularios</h1>

  <h2 style="color:blue">{{ mensaje }}</h2>

  <form #userForm="ngForm" (ngSubmit)="onSubmit()">

    <label for="nombre">Nombre: </label>

    <input

      type="text"

      name="nombre"

      #nombre="ngModel"

      [(ngModel)]="user.nombre"

      pattern="[A-Z]{1}[a-z]{2,}"

      required

    />

    <label style="color:red" \*ngIf="!nombre.valid && nombre.touched">

      El nombre no es válido

    </label>

    <br />

    <label for="apellidos">Apellidos: </label>

    <input

      type="text"

      name="apellidos"

      #apellidos="ngModel"

      [(ngModel)]="user.apellidos"

      required

    />

    <label style="color:red" \*ngIf="!apellidos.valid && apellidos.touched">

      Los apellidos no son válidos

    </label>

    <br />

    <label for="edad">Edad: </label>

    <input type="number" name="edad" #edad="ngModel" [(ngModel)]="user.edad" />

    <label style="color:red" \*ngIf="!edad.valid && edad.touched">

      La edad no es válida (18-99)

    </label>

    <br />

    <button [disabled]="userForm.invalid">Enviar datos</button>

  </form>

</div>

<div style="float:left" \*ngIf="user.nombre">

  <h2>Datos del formulario</h2>

  <ul>

    <li>Nombre: {{ this.user.nombre }}</li>

    <li>Apellidos: {{ this.user.apellidos }}</li>

    <li>Edad: {{ this.user.edad }}</li>

  </ul>

</div>



**SUMARNUMEROS.COMPONENT.HTML**

<h1>Formularios con referencia</h1>

<!-- DEBEMOS UTILIZAR ngForm EN EL ID

DEL FORMULARIO DE ANGULAR -->

<form #sumarForm="ngForm">

  <div>

    <label for="cajanumero1">Número 1: </label>

    <!-- AL CONTROL HTML SE LE INDICA UN ID DE ANGULAR

        Y NADA MAS

        EL ENLACE DEL OBJETO SE REALIZA EN EL CONTROLLER -->

    <input type="number" #cajanumero1 name="cajanumero1" />

  </div>

  <div>

    <label for="cajanumero2">Número 2: </label>

    <input type="number" name="cajanumero2" #cajanumero2 />

  </div>

  <br />

  <button (click)="sumarNumeros()">Sumar números</button>

</form>

<h1 style="color:blue" \*ngIf="suma">

    La suma de los números es: {{ suma }}

</h1>

**SUMARNUMEROS.COMPONENT.TS**

import { Component, OnInit, ViewChild, ElementRef } from "@angular/core";

//ELEMENTREF: PARA INDICAR UNA REFERENCIA

//EQUIVALENTE A createRef

//VIEWCHILD: INDICAR EL ID DE ANGULAR DE LA CAJA

//HAREMOS UNA COMBINACION PARA ASOCIAR EL ID DE LA CAJA

//DE LA VISTA CON UNA REFERENCIA

@Component({

  selector: "app-sumar-numeros",

  templateUrl: "./sumar-numeros.component.html",

  styleUrls: ["./sumar-numeros.component.css"]

})

export class SumarNumerosComponent implements OnInit {

  //VARIABLES PARA TRABAJAR

  public suma: number;

  //NECESITAMOS LA CAJA COMO REFERENCIA

  @ViewChild("cajanumero1") cajanumero1: ElementRef;

  @ViewChild("cajanumero2") cajanumero2: ElementRef;

  sumarNumeros() {

    //CAPTURAR LOS VALORES DE LAS CAJAS DE REFERENCIA

    //objeto.nativeElement.value

    //value SIEMPRE ES UN STRING AUNQUE VENGA COMO NUMBER

    var num1 = parseInt(this.cajanumero1.nativeElement.value);

    var num2 = parseInt(this.cajanumero2.nativeElement.value);

    this.suma = num1 + num2;

    console.log(num1 + num2);

  }

  constructor() {}

  ngOnInit(): void {}

}