

**Objetivo:** Poner en práctica los conceptos aprendidos en clase sobre los formularios en IONIC framework para el desarrollo de aplicaciones para móviles.

Abra una terminal de comandos de NodeJs (Con privilegios de administrador).

```
Administrador: Node.js command prompt
Your environment has been set up for using Node.js 13.13.0 (x64) and npm.

C:\Windows\System32>cd /

C:\>cd proyectosIonic

C:\proyectosIonic>
```

Desplácese hasta la carpeta donde ha estado creando los proyectos de IONIC y cree una nueva App con la plantilla en blanco:

```
Administrador: Node.js command prompt
Your environment has been set up for using Node.js 13.13.0 (x64) and npm.

C:\Windows\System32>cd /

C:\>cd proyectosIonic

C:\proyectosIonic>ionic start operaciones2numeros blank --type=angular
```

Una vez creada la App, abra su editor de texto predilecto y abra la carpeta del proyecto que acaba de crear.

Realice las modificaciones siguientes al archivo **home.module.ts**

```
TS home.module.ts
src > app > home > TS home.module.ts > ...

1 import { NgModule } from '@angular/core';
2 import { CommonModule } from '@angular/common';
3 import { IonicModule } from '@ionic/angular';
4 import { FormsModule, ReactiveFormsModule } from '@angular/forms';
5 import { HomePage } from './home.page';
6
7 import { HomePageRoutingModule } from './home-routing.module';
8
9
10 @NgModule({
11   imports: [
12     CommonModule,
13     ReactiveFormsModule,
14     FormsModule,
15     IonicModule,
16     HomePageRoutingModule
17   ],
18   declarations: [HomePage]
19 })
20 export class HomePageModule {}
```

1

2

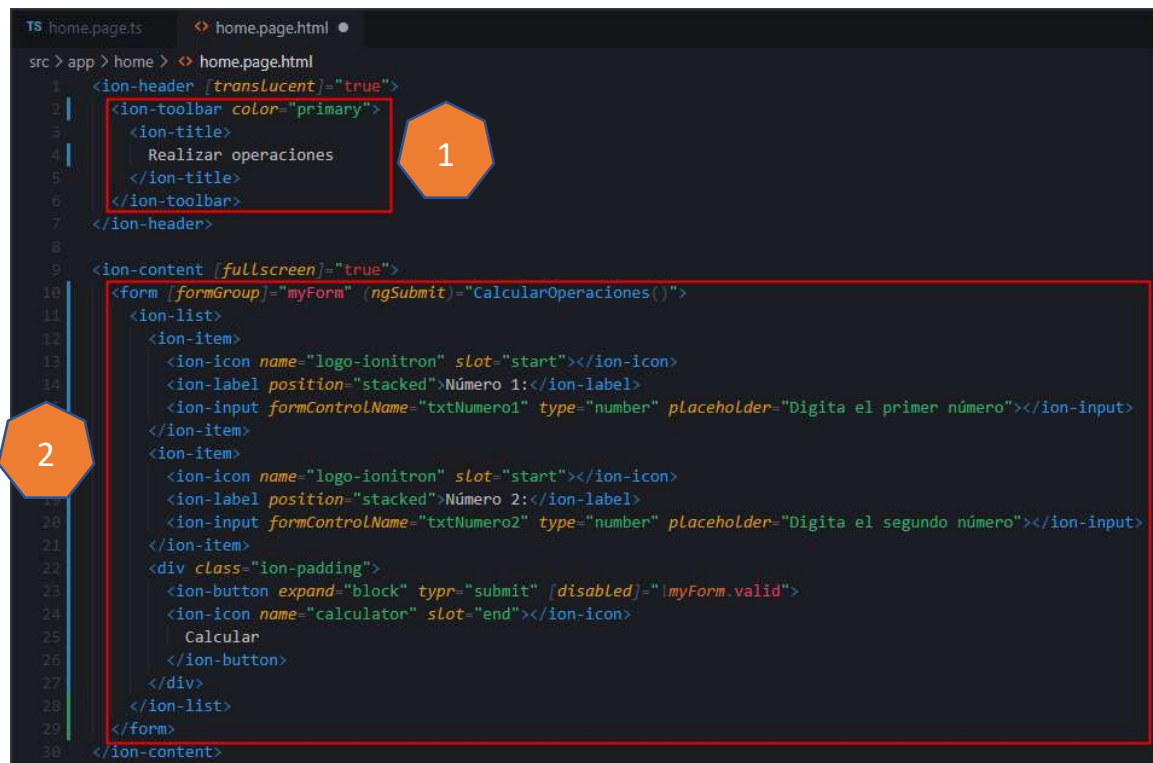
- 1- Importe el modulo «**ReactiveFormsModule**» del paquete @angular/forms.
- 2- Agregue a las importaciones del NgModule el módulo «**ReactiveFormsModule**».

Luego realice las modificaciones siguientes en el archivo **home.page.ts**

```
TS home.page.ts X
src > app > home > TS home.page.ts > ...
1  import { Component } from "@angular/core";
2  import { FormBuilder, FormGroup, Validators } from "@angular/forms";
3
4  @Component({
5    selector: "app-home",
6    templateUrl: "home.page.html",
7    styleUrls: ["home.page.scss"],
8  })
9  export class HomePage {
10    myForm: FormGroup;
11
12    constructor(public formBuilder: FormBuilder) {
13      this.myForm = this.createMyForm();
14    }
15
16    private createMyForm() {
17      return this.formBuilder.group({
18        txtNumero1: ["", Validators.required],
19        txtNumero2: ["", Validators.required]
20      });
21    }
22
23    CalcularOperaciones()
24    {
25      console.log(this.myForm.value);
26    }
27  }
```

1. Debemos hacer el Import de los módulos **FormBuilder**, **FormGroup**, **Validators** que vienen con el core de angular.
2. Creamos una variable de tipo **FormGroup** que nos va a servir para poder estructurar el formulario.
3. Debemos inyectar a **FormBuilder** como dependencia a nuestro constructor.
4. Cuando se cargue el método constructor se debe realizar la llamada al método **createMyForm()**
5. Ahora vamos a crear el método privado **createMyForm** para crear el formulario, haciendo uso de nuestra variable **this.formBuilder** podremos crear un controlador para cada uno de nuestros campos en el formulario, aquí podremos enviar validaciones tan sencillas o complejas como queramos.
6. Creamos un método **CalcularOperaciones()** que por el momento solo debe imprimir un mensaje en la Consola del DOM del navegador.

Ahora vamos a trabajar en la vista del formulario para ello realice los siguientes cambios en el archivo **home.page.html**:



```
1 <ion-header [translucent]="true">
2   <ion-toolbar color="primary">
3     <ion-title>
4       Realizar operaciones
5     </ion-title>
6   </ion-toolbar>
7 </ion-header>
8
9 <ion-content [fullscreen]="true">
10  <form [formGroup]="myForm" (ngSubmit)="CalcularOperaciones()">
11    <ion-list>
12      <ion-item>
13        <ion-icon name="logo-ionitron" slot="start"></ion-icon>
14        <ion-label position="stacked">Número 1:</ion-label>
15        <ion-input formControlName="txtNumero1" type="number" placeholder="Digita el primer número"></ion-input>
16      </ion-item>
17      <ion-item>
18        <ion-icon name="logo-ionitron" slot="start"></ion-icon>
19        <ion-label position="stacked">Número 2:</ion-label>
20        <ion-input formControlName="txtNumero2" type="number" placeholder="Digita el segundo número"></ion-input>
21      </ion-item>
22      <div class="ion-padding">
23        <ion-button expand="block" type="submit" [disabled]="!myForm.valid">
24          <ion-icon name="calculator" slot="end"></ion-icon>
25          Calcular
26        </ion-button>
27      </div>
28    </ion-list>
29  </form>
30 </ion-content>
```

1. Agregamos en la barra de navegación un color, no necesariamente el "primary" (azul).
2. Agregamos el formulario dentro de las etiquetas <ion-content>

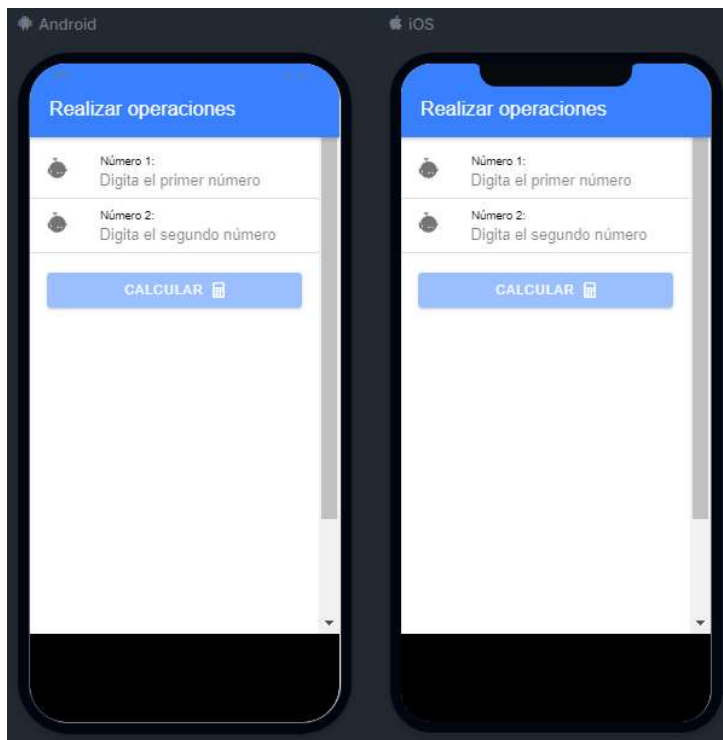
Ahora para agregar el controlador de nuestro formulario, debemos asignar dentro del atributo **formGroup** la instancia de **myForm**, además debemos asignar un método para recibir la información **(ngSubmit)="CalcularOperaciones()"**.

Por último, por cada campo debemos usar **formControlName** para asignar el controlador en cada campo.

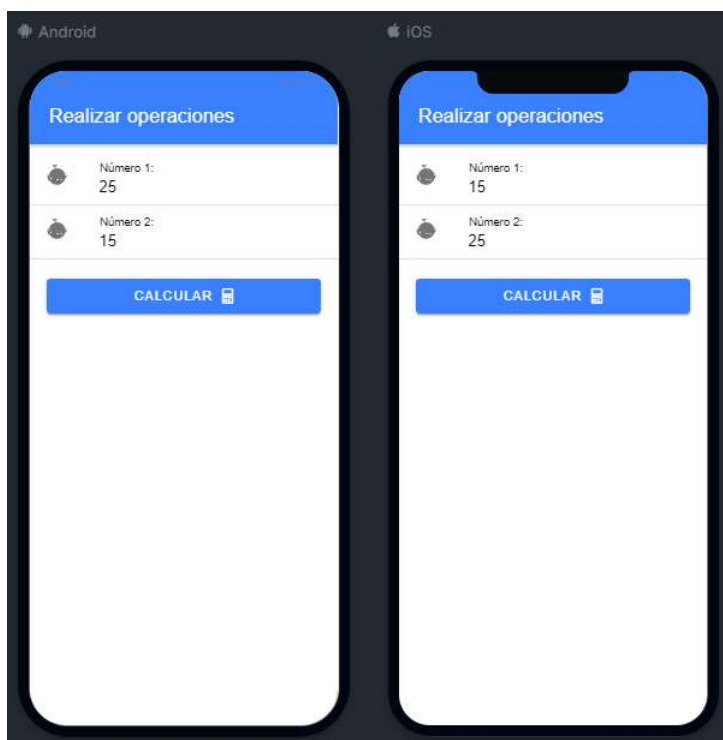
Ahora es necesario verificar el funcionamiento de nuestra App, ejecuta en la terminal de VSCode:

➤ **ionic serve -l**

Debemos llegar a un resultado similar al siguiente:



Al digitar los valores se debe activar el botón Calcular.



Realice las siguientes modificaciones al archivo **home.page.ts** para poder realizar la operación de suma de ambos números y presentarlos al usuario por medio de un mensaje de tipo TOAST.

```
import { Component } from "@angular/core";
import { FormBuilder, FormGroup, Validators } from "@angular/forms";
import { ToastController } from '@ionic/angular';

@Component({
  selector: "app-home",
  templateUrl: "home.page.html",
  styleUrls: ["home.page.scss"],
})
export class HomePage {
  myForm: FormGroup;
  formValues: any[];
  suma: number=0;
  mensaje: string;

  constructor(public formBuilder: FormBuilder, public toastCtrl: ToastController) {
    this.myForm = this.createMyForm();
  }

  private createMyForm() {
    return this.formBuilder.group({
      txtNumero1: ["", Validators.required],
      txtNumero2: ["", Validators.required]
    });
  }

  CalcularOperaciones()
  {
    console.log(this.myForm.value);

    this.formValues = this.myForm.value;

    console.log(parseInt(this.formValues['txtNumero1']) + parseInt(this.formValues['txtNumero2']));

    this.suma = parseInt(this.formValues['txtNumero1']) + parseInt(this.formValues['txtNumero2']);

    this.mensaje = 'La suma de los numeros '+this.formValues['txtNumero1'] +' y '+ this.formValues[
'txtNumero2'] + ' es: ' + this.suma;

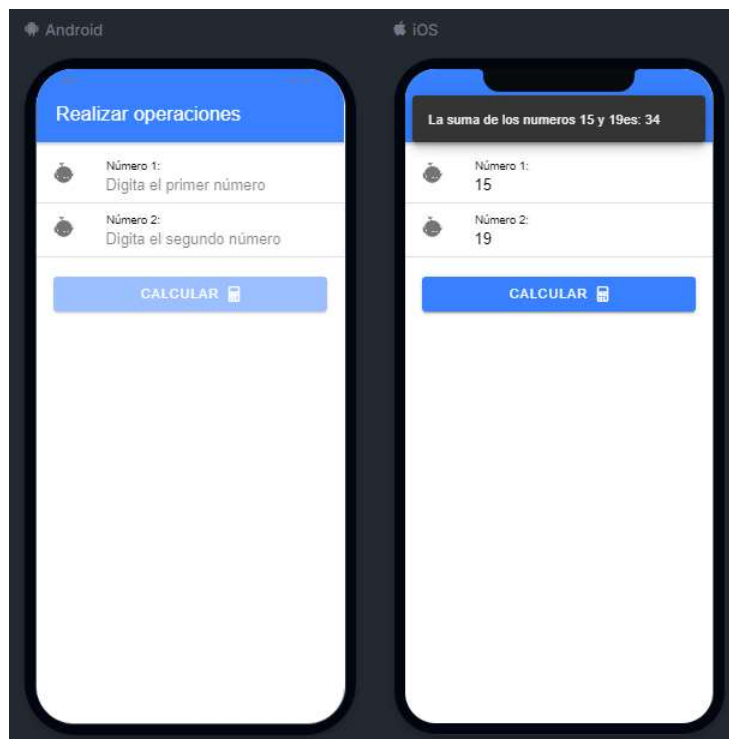
    this.mostrarToast(this.mensaje);
  }
}
```

```

async mostrarToast(msj: string)
{
    const toast = await this.toastCtrl.create({
        message: msj,
        duration: 3000,
        position: 'top'
    });
    toast.present();
}
}

```

El resultado debería mostrarle al usuario un mensaje similar a este en pantalla:



Agregue el código necesario para que en el mismo método se calculen las otras tres operaciones básicas con ambos números (**resta**, **multiplicación**, **división**).

Agregue al mensaje **toast** el resultado de cada operación.

Comprima el proyecto y súbalo a la plataforma virtual en el enlace asignado.