

*Universidad Tecnológica de la*

*Riviera Maya*

**Desarrollo de Aplicaciones III**

**Profesor:**

*Lic. Julián Villegas*

**Autor(es):**

*Domínguez Juárez Juan Antonio*

*Solidaridad, Quintana Roo, 26 de Marzo del 2018*

**Carrera:**

*TSU en Tecnologías de la Información y Comunicación Área Sistemas Informáticos (TI51)*

**Tabla de contenido**

[**Definición** 2](#_Toc509874627)

[**Ejemplo1** 3](#_Toc509874628)

[**Ejemplo2** 4](#_Toc509874629)

[**Fuentes de información** 5](#_Toc509874630)

## **Definición**

Tan pronto como importe FormsModule, esta directiva se activará por defecto en todas las etiquetas <form>. No necesita agregar un selector especial.

Puede exportar la directiva a una variable de plantilla local utilizando ngForm como clave (por ejemplo: # myForm = "ngForm"). Esto es opcional, pero útil. Muchas propiedades de la instancia subyacente de FormGroup están duplicadas en la directiva misma, por lo que una referencia a ella le dará acceso al valor agregado y al estado de validez del formulario, así como a las propiedades de interacción del usuario como dirty y tocado.

Para registrar controles secundarios con el formulario, querrá usar NgModel con un atributo de nombre. También puede usar NgModelGroup si desea crear subgrupos dentro del formulario.

Puede escuchar el evento ngSubmit de la directiva para recibir una notificación cuando el usuario haya activado un envío de formulario. El evento ngSubmit se emitirá con el evento de envío de formulario original.

En los formularios dirigidos por plantilla, todas las etiquetas <form> se etiquetan automáticamente como NgForm. Si desea importar el FormsModule pero omita su uso en algunos formularios, por ejemplo, para usar la validación HTML5 nativa, puede agregar ngNoForm y las etiquetas <form> no crearán una directiva NgForm. En formas reactivas, el uso de ngNoForm es innecesario porque las etiquetas <form> son inertes. En ese caso, se abstendría de usar la directiva formGroup.

## **Ejemplo1**

import {[Component](https://angular.io/api/core/Component)} from '@angular/core';

import {[NgForm](https://angular.io/api/forms/NgForm)} from '@angular/forms';

@[Component](https://angular.io/api/core/Component)({

selector: 'example-app',

template: `

<form #f="[ngForm](https://angular.io/api/forms/NgForm)" (ngSubmit)="onSubmit(f)" novalidate>

<input name="first" [ngModel](https://angular.io/api/forms/NgModel) required #first="[ngModel](https://angular.io/api/forms/NgModel)">

<input name="last" [ngModel](https://angular.io/api/forms/NgModel)>

<button>Submit</button>

</form>

<p>First name value: {{ first.value }}</p>

<p>First name valid: {{ first.valid }}</p>

<p>[Form](https://angular.io/api/forms/Form) value: {{ f.value | [json](https://angular.io/api/common/JsonPipe) }}</p>

<p>[Form](https://angular.io/api/forms/Form) valid: {{ f.valid }}</p>

`,

})

export class SimpleFormComp {

onSubmit(f: [NgForm](https://angular.io/api/forms/NgForm)) {

console.log(f.value); // { first: '', last: '' }

console.log(f.valid); // false

}

}

## **Ejemplo2**

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">

<head>

<title></title>

<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.4.2/angular.min.js"></script>

</head>

<body ng-app>

<div>

<ng-form ng-hide="isDetail">

First Name:

<input type="text" ng-model="fName">

<br />

Last Name:

<input type="text" ng-model="lName">

<br />

User Name:

<input type="text" ng-model="uName">

<br />

Password:

<input type="password" ng-model="password">

<br />

</ng-form>

<input type="button" ng-click="isDetail=true" value="Submit" />

<div ng-show="isDetail">

First Name:<b>{{fName}}</b><br />

Last Name:<b>{{lName}}</b><br />

User Name:<b>{{uName}}</b><br />

Password:<b>{{password}}</b>

</div>

</div>

</body>

</html>

## **Fuentes de información**

https://angular.io/api/forms/NgForm

https://www.learnkode.com/Examples/Angular/Ng-Form