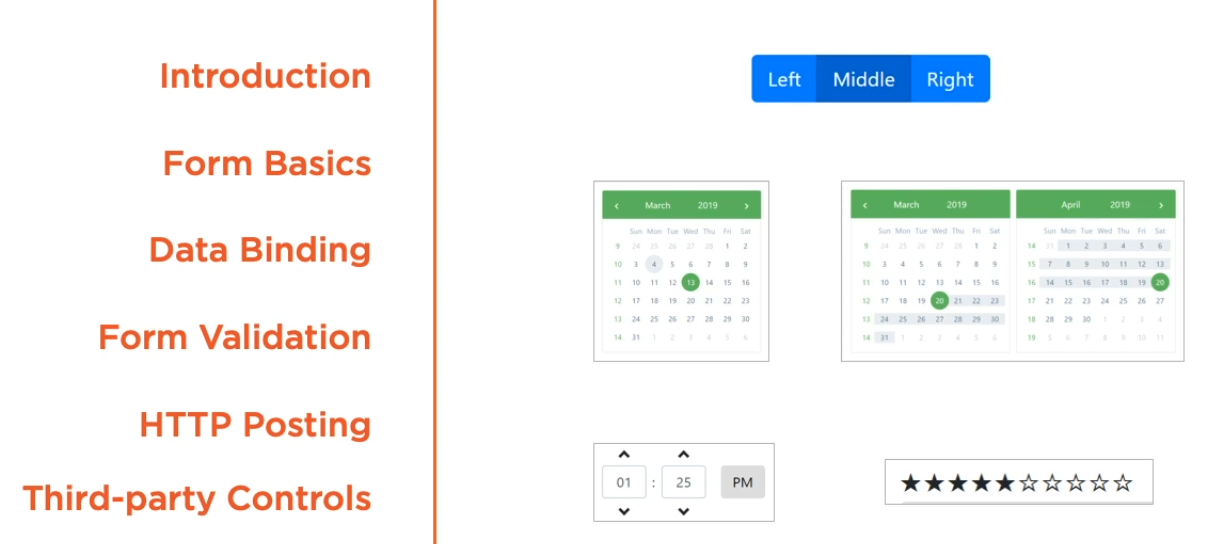
**Angular Forms**

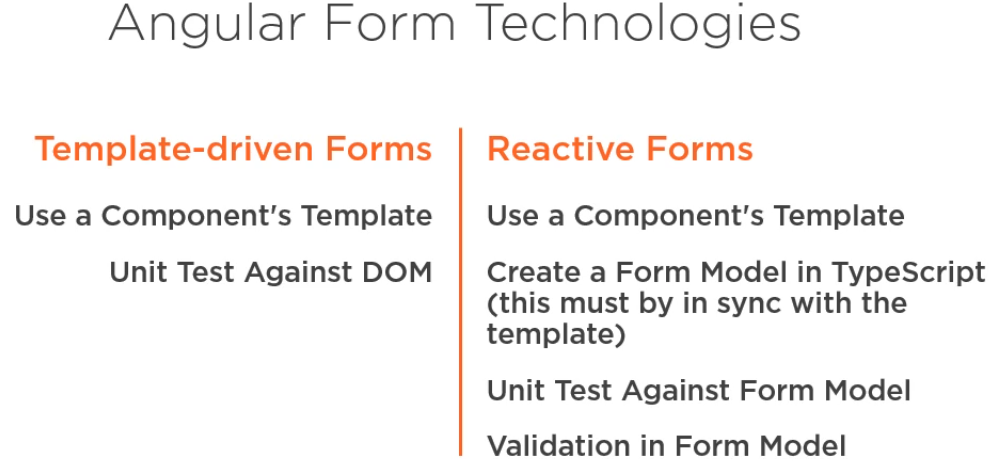
**Introduction:**



**Angular Form Technologies**

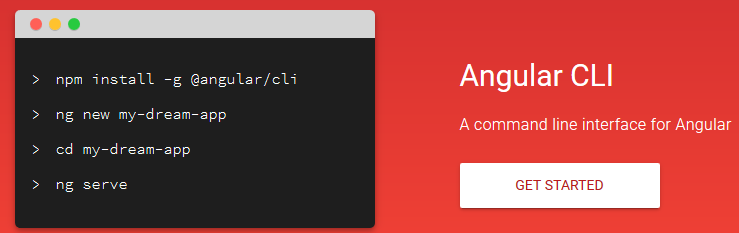
Existen dos tecnologías de Angular Forms, **Formularios Basados en Plantillas** y **Formularios Reactivas (Reactive Forms).**

* Cuando creamos un Componente en Angular tiene una vista.
* La vista se llama Plantilla.
* Po lo general, se coloca en el archivo HTML.



**Creating the Demo Application**

Se procedió a crear una aplicación mediante la CLI de Angular.. y a Instalar el CLI….



**Form Basics in Angular:**

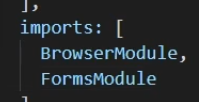
Estos son los puntos que vamos a ver en este 3er modulo.



**Angular’s FormsModule**

Antes de utilizar cualquier tipo de formulario en Angular, debemos de Importar el **FORMSMODULE** en el archivo **app.module.ts.**





**Creating a Form Component**

**Module 4 - Data Binding in Angular Forms**

Angular Posee una directiva llamada **ngForm** que se agrega de manera automática a cada formulario del Proyecto. Esta funcionalidad la obtuvimos al momento que importamos el  en el proyecto.

Vamos a ver también **NgModel**, esto se usa en los campos del formulario y ayuda a construir nuestro modelo, el objeto utilizado para mantener nuestros datos.



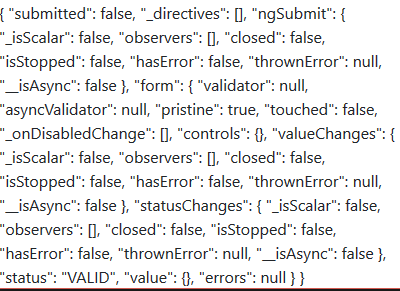
**1.- Using NgForm**

Primero debemos crear una variable en el formulario en el archivo HTML…..

Esta es la sintaxis: #form = “ngForm” .

Ya para ver la data, utilizamos la interpolación y le pasamos la variable:

{{ form | json }} … La data la podemos ver como un Json

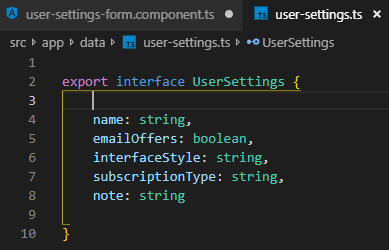


**2.- NgModel**

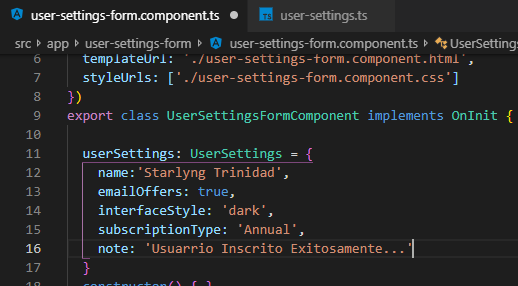
Al crear un formulario, la directiva **NgModel** se debe de colocar en cada campo que queremos que Angular Conozca.

**3.- Creating a Data Model**

Se procedió a crear una carpeta llamada DATA… Dentro de la misma creamos un archivo llamado **user-settings.ts** y en el mismo creamos una interfaz para utilizarla de Nuestro Modelo de Datos.



Ya más adelante utilizamos esa interfaz en el componente del usuario y pasamos nuestra data:

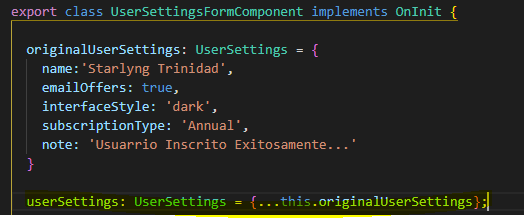


**4.- Two-way Data Binding**

Para establecer un enlace de dos vías, solo debemos de pasar al formulario en nuestro archivo HTML y en la directiva NgModel, la colocamos entre paréntesis y corchetes  a esta sintaxis se le conoce como la sintaxis del Banano, luego de esto la vinculamos a nuestra fuente de datos, en este caso es UserSettings.Name o nombre de la propiedad correspondiente

**5.- Copying Form Data**

Hay que tener en cuenta este escenario cuando vamos a trabajar con Formularios en Angular, ¿Qué sucede si un usuario comienza a llenar un formulario y luego presiona le botón Cancelar o el botón Atrás? Deseamos que los cambios que hicieron se cancelen. Entonces, una manera rápida de hacerlo es asegurarnos que el usuario siempre este trabajando con una copia de los datos del formulario original. Y esto se consigue con esta manera sencilla:



Aquí lo que hicimos es, crear dos instancias de la interfaz donde se encuentran nuestros datos… una le colocamos el nombre de **originalUserSettings** estos son los datos originales y la otra es **userSettings** que es la copia, como podemos observar a **userSettings** la igualamos a **originalUserSettings**, ya con esto estamos protegiendo nuestra información.

**Module 5 – Form Validation**

Este módulo trata de las validaciones de los formularios. Entre otros puntos:



**1.- HTML5 Field Validation**

En HTML5 las validaciones se realizan principalmente con atributos en cada campo.



Estos son algunos de los atributos utilizados en HTML5.

En la etiqueta de Form en el archivo HTML, debemos de colocar ngNativeValidate:



Esto para poder comenzar con las validaciones.

Ya luego podemos colocar las restricciones en las etiquetas:



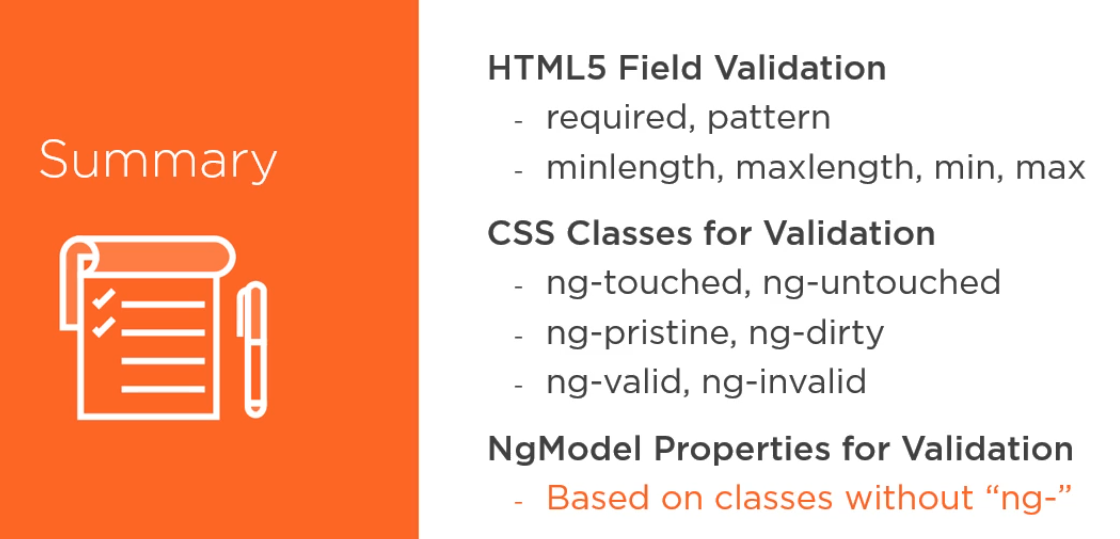
**2.- CSS Classes for Validation**

**3.- NgModel Properties for Validation**

**4.- Styling Forms with Validation Errors**

**5.- Submitting Forms**

Acá simplemente se crea un evento y lo pasamos al form.





**Module 6 – HTTP Form Posting and Data Access**

1 - En este módulo primero vamos a crear un Servicio de Datos.

2 – Veremos cómo publicar nuestro formulario usando Observables.

Los observables se implementan en la biblioteca RxJS, y se utilizan exactamente para la programación asíncrona.

3 – utilizaremos el HTTPClient.



1 – Creando un Data Service

Procedimos a crear el data Service utilizando el CLI de angular.

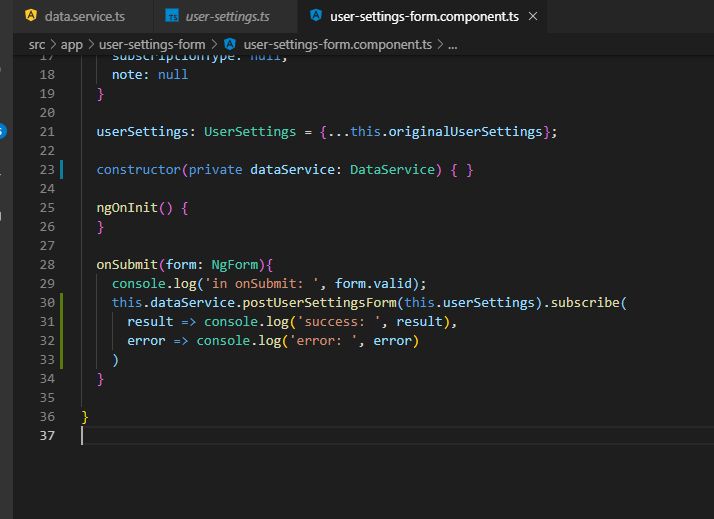
Ng g s data

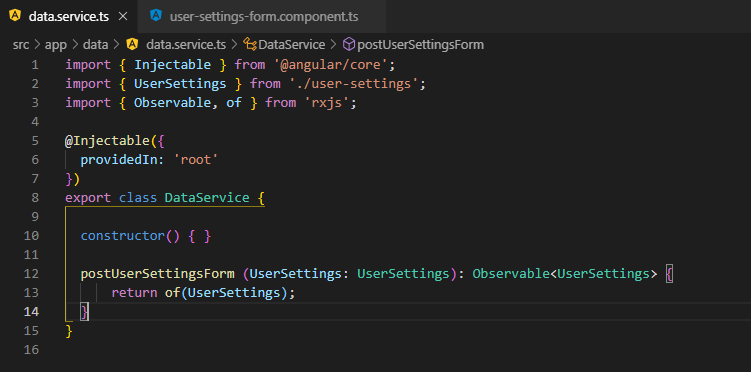
Y Luego procedimos a crear el método



2 – Form Posting Using Observables

Cuando se publica un formulario, enviamos los datos del formulario a un servidor, y esperamos una respuesta, y eso es una operación asíncrona.





3 – HTTP Access Using HttpClient

Exportamos en el módulo principal app.module.ts en cliente HTTP:

 y 

4 – Posting a Form

Acá utilizamos una página web llamada <https://putsreq.com> para probar los Request de nuestro formulario.