**Validación de entradas de información en un aplicativo Laravel**

Contenido

[Los formularios en Laravel 2](#_Toc128582551)

[Archivo validation.php 2](#_Toc128582552)

[Elementos de validación en el componente Livewire 4](#_Toc128582553)

[El arreglo $messages 4](#_Toc128582554)

[La función rules() 5](#_Toc128582555)

[Llamada a método validate() 5](#_Toc128582556)

[Creación de reglas propias de validación en Laravel 6](#_Toc128582557)

[Anexo de reglas de validación comunes de Laravel 7](#_Toc128582558)

## Los formularios en Laravel

En el proceso de validación de datos de entradas, típicamente el usuario capturará información que la aplicación le solicitará a través de un formulario web.

Al tener nuestro aplicativo desarrollado en el framework Laravel-Livewire, nuestro formulario estará dentro de un archivo .blade.php

Consideremos el siguiente formulario, en dónde se solicita al usuario ingresar su nombre.

<label class="block text-sm font-medium text-gray-700 leading-5"

                for="etiqueta\_nombre">Nombre:

                <span class="text-sm text-red-600">\* </span>

                <textarea required class="w-full" maxlength="100" rows="4" wire:model="nombre" id="nombre"

                    name="nombre">

                </textarea>

                <span class="block error h-2 mb-3">

                    @error('nombre')

                        {{ $message }}

                    @enderror

                </span>

            </label>

Una directiva importante del motor de interfaces blade es @error y @enderror, en dónde se colocará el mensaje de error por haber roto las reglas que validan al campo nombre en el componente Livewire. Por lo que en el ejemplo, $message es la variable que contendrá el texto con el mensaje que indicará al usuario que el campo está validado y le deberá brindar la ayuda para ingresar correctamente la información.

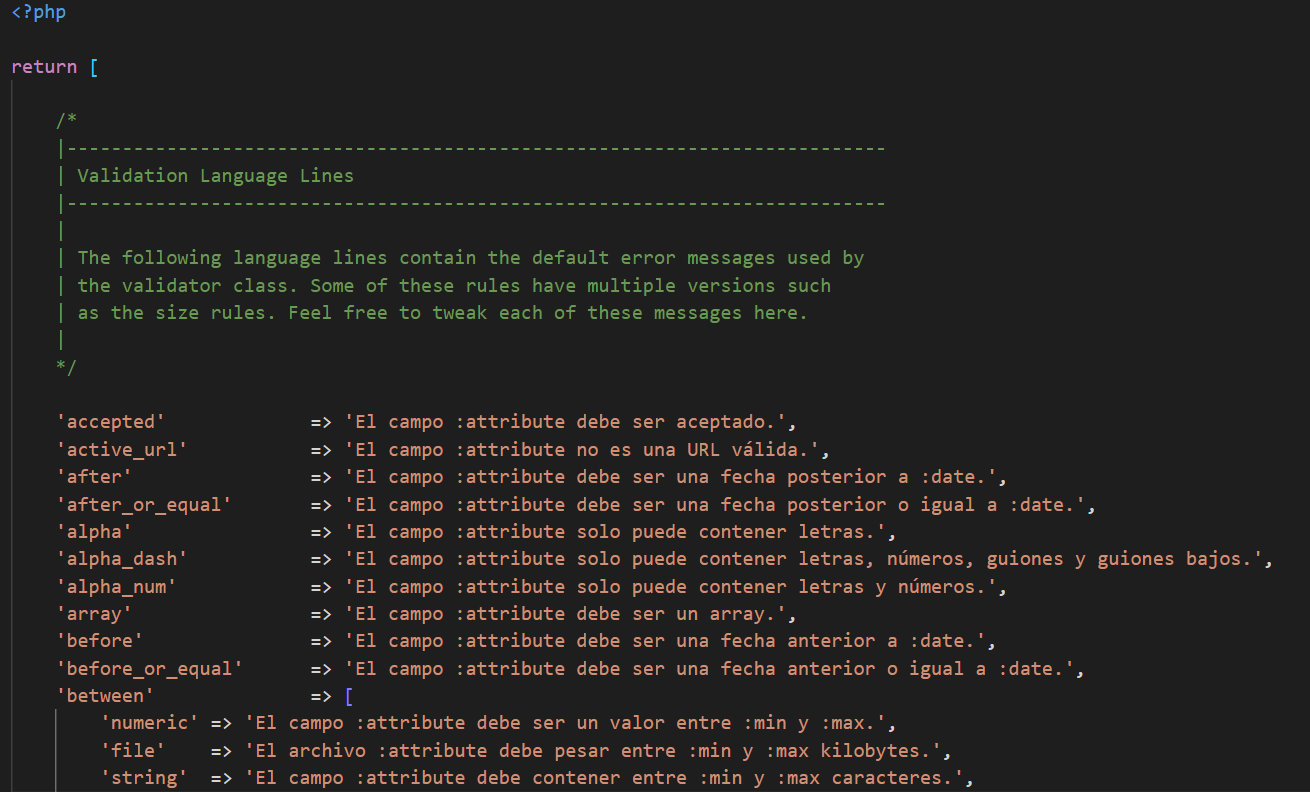
## Archivo validation.php

Otro elemento importante dentro del proceso de validación de Laravel es el archivo validation.php que se ubica en el directorio lang.

Si hemos configurado el idioma español, entonces deberemos tener este archivo en el directorio lang\es\validation.php

Si no lo tenemos, hay que buscarlo en la red, descargarlo y colocarlo en el directorio mencionado, puesto que ya otros desarrolladores han traducido todos los mensajes de error de las reglas de validación pre construidas en Laravel.

En la siguiente imagen se muestra una parte del contenido del archivo validation.php



Como se puede apreciar, el contenido del archivo sólo son los tipos de error de las reglas de validación y el mensaje que se mostrará al usuario en la posición dónde se ubique @error {{ $message }}@enderror dentro del blade.

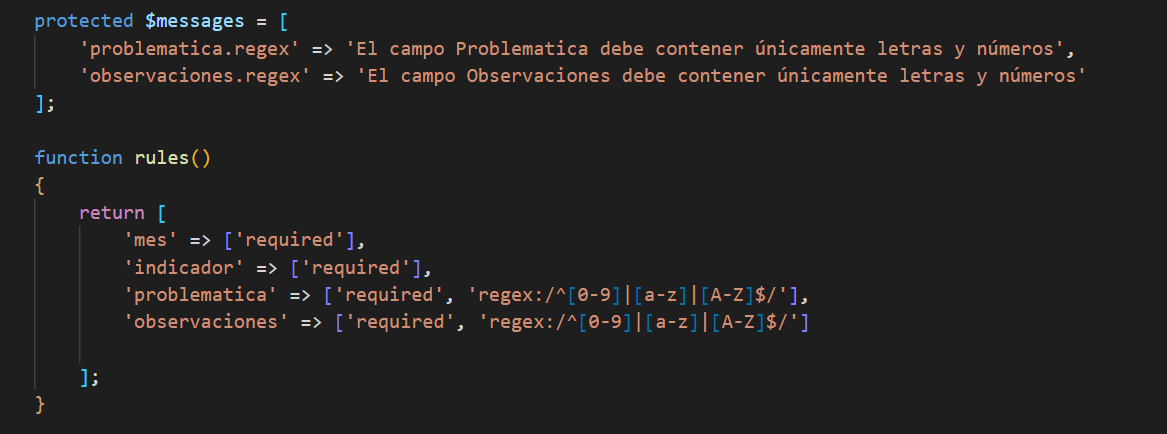
Nosotros podemos modificar el texto del mensaje para tropicalizarlo al aplicativo que estemos desarrollando y también podemos especificar cómo llamará Laravel a cada campo del formulario al notificar el mensaje de error hacia el usuario.

En la imagen que veremos a continuación se muestra cómo nombrar los campos del formulario para cada variable de tipo input de nuestro formulario. Por ejemplo, para el campo observaciones, Laravel dirá que ha ocurrido un error en Observaciones, porque así se le ha especificado en este archivo que debe llamar al campo observaciones. Debe desplegar el nombre con “O” mayúscula.



## Elementos de validación en el componente Livewire

Ahora bien, los elementos de validación que lleva el componente de Livewire son los siguientes.



### El arreglo $messages

Un arreglo $messages de tipo protected, para asociar los mensajes de error para una regla en particular.

En el ejemplo, observemos cómo para el campo problemática, si se rompiera la regla de validación regex, entonces el sistema mostraría el mensaje “El campo Problemática debe contener únicamente letras y números”.

### La función rules()

Y ahora observemos la función rules(). Esta función debe contener las reglas de validación para cada campo del formulario que deseemos validar. Dichas reglas de validación deben definirse en un arreglo.

Para el campo problemática se observa que se deben cumplir dos reglas:

1. Una es que el campo es requerido, es decir, que el usuario debe capturar algo en el campo problemática. No lo debe dejar vacío.
2. La segunda regla es que lo que capture el usuario, debe satisfacer la expresión regular que ahí se señala y que como vemos, puede contener dígitos del 0 al 9, letras minúsculas desde la letra a hasta la letra z y también letras mayúsculas desde la letra A hasta la letra Z. Vemos que no debe contener caracteres especiales.

Algo importante a remarcar sobre las reglas de validación es que ya existen muchas previamente construidas por el framework Laravel, así que es necesario leer la documentación y utilizar las reglas que ya están definidas por el framework, tal como required, unique, entre otras muy comúnmente utilizadas.

### Llamada a método validate()

Finalmente, es necesario invocar al método validate desde un lugar adecuado dentro de la lógica de nuestra aplicación, para que Laravel realice las validaciones correspondientes que se definieron en el componente.

 $this->validate();

## Creación de reglas propias de validación en Laravel

También Laravel nos permite definir nuevas reglas de validación para casos muy específicos de nuestra aplicación.

Para crear la regla de validación deberemos ejecutar el comando:

php artisan make:rule Nombre\_Nueva\_Regla

Con esta sentencia, Laravel creará un archivo para que definamos el código de la validación en el directorio app\Rules.

El archivo para especificar nuestra nueva regla será parecido a lo siguiente:

namespace App\Rules;

use Illuminate\Contracts\Validation\Rule;

class Nombre\_Nueva\_Regla implements Rule

{

/\*\*

\* Determine if the validation rule passes.

\*

\* @param string $attribute

\* @param mixed $value

\* @return bool

\*/

public function passes($attribute, $value)

{

//

}

/\*\*

\* Get the validation error message.

\*

\* @return string

\*/

public function message()

{

return 'The validation error message.';

}

}

La función passes debe devolver un valor booleano de true o un false, que le indicará a Laravel si el atributo que recibió, cumplió con la regla definida dentro de esta misma función.

Finalmente en la función message, hay que retornar el mensaje de error si la información capturada por el usuario no cumpliera con nuestra regla de validación.

## Anexo de reglas de validación comunes de Laravel

A continuación se muestran algunas de las reglas de validación más comúnmente utilizadas ya pre-construidas por Laravel.

