# Formularios

## Validando formulario

Antes de nada. Si queremos validar un formulario con PHP no tiene sentido que tengamos la validación activada con JavaScript. Se supone que si validamos todo con JavaScript no haría falta validar nuestro código con PHP.

De todas maneras, conviene tener las dos opciones activas por si el navegador no admite JavaScript.

Otra consideración es pensar que el propio navegador valida los formularios con JavaScript cuando ponemos un atributo como required en nuestros controles. Resulta un poco fastidiado tener que borrarlos y ponerlos a continuación.

## Un poco de JavaScript

Pon el siguiente script en un archivo js

//capturamos el formulario

miFormulario = document.querySelector("#form\_contact");

//quitamos la opción automática de validado del formulario. Las opciones dejan de ser operativas

miFormulario.noValidate = true;

¡¡¡Listo!!!

Si queremos activar todo de nuevo lo borramos o ponemos la propiedad noValidate = false.

## Validando ahora sí el formulario

Lo primero es comprobar si el formulario fue enviado:

if(isset($\_POST['submit'])){

}

Elementos a comprobar

* Si el campo está vacío  
  Para ello tenemos la función empty()
* Limpiar los espacios en blanco en los laterales  
  trim()
* Evitar la inyección de código HTML con
  + htmlspecialchars() Convierte caracteres de las etiquetas html en entities.
  + stripslashes() Evita caracteres como \, que también puede dar lugar a inyección de código
* Aplicar filtros de saneamiento con la función filter\_var()  
  https://www.php.net/manual/es/filter.filters.sanitize.php

Veamos un ejemplo validando el campo nombre del formulario. Lo primero que haremos es comprobar que no se encuentre vacío:

if(!empty($nombre)):

//tu código

else:

//mensaje de error

endif;

A continuación, procederemos a realizar un filtrado del campo. Una manera de hacerlo es:

if(!empty($nombre)):

  $nombre=trim($nombre); //Limpiamos lo espacios en blanco

$nombre=htmlspecialchars($nombre); //convertimos las etiquetas html en entidades

  $nombre=stripslashes($nombre); //limpiamos otros caracteres como / o \

endif;

Otra forma de hacerlo es con filtros preparados para “sanear” las cadenas de texto de nuestras variables. Para ello utilizaremos la función que contiene dos parámetros: la variable que vamos a limpiar/sanear y un constante que indica el tipo de filtro. Para saber todos los filtros de saneamiento ver la siguiente dirección:

* https://www.php.net/manual/es/filter.filters.sanitize.php

if(!empty($nombre)):

     $nombre=trim($nombre);

     $nombre=filter\_var($nombre,FILTER\_SANITIZE\_STRING); //filtro

endif;

Ejercicio

A continuación, sanear (SANITIZE) el correo electrónico

Una vez “saneado” el correo el electrónico hay que validarlo. Sanearlo intenta almacenar caracteres válidos para un correo, pero no implica que sea correcto.

Os recomiendo instalar el plugin **PHP IntelliSense** en VS Code que entre otras cosas os muestra las constantes (hay que escribirlas en mayúsculas) de los filtros.

Eso lo veremos en el próximo Word.