Validaciones HTML5

- Instructor Jose Luis Sarta Alvarez

Caso 1.

- En este primer caso, el formulario no tiene validación de ningún tipo. El usuario puede escribir la información y el sistema no comprobará los datos, ni realizará ningún tipo de validación. Es el peor escenario posible, puesto que el usuario podría enviar desde información incorrecta, hasta datos malintencionados que podrían comprometer la seguridad de la página.

Caso 2.

- Otro caso podría ser que el formulario tiene validación sólo en el front-end (cliente). De esta forma, los datos son verificados en el navegador del usuario antes de enviarse, pero carecen de validación en el back-end, por lo que un usuario malintencionado podría eliminar la validación del front-end y saltársela, enviando datos malintencionados que comprometan la seguridad de la página.

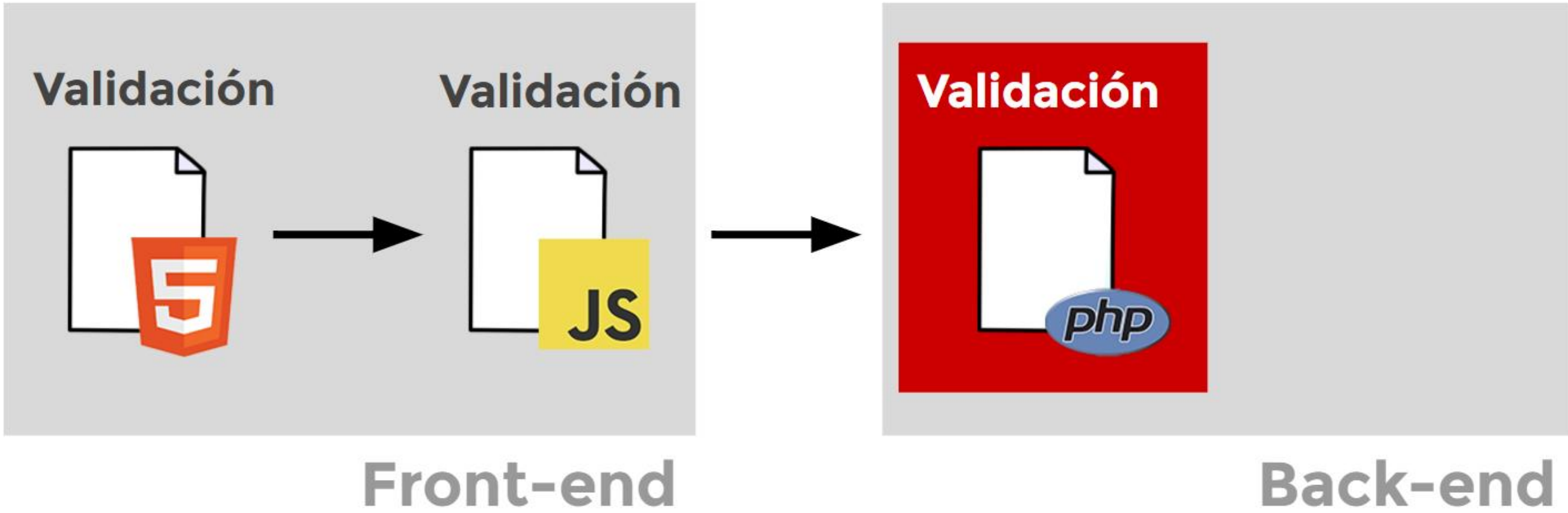
Caso 3.

- El tercer caso posible es uno donde el formulario tiene validación sólo en el back-end. De esta forma, garantizamos que un usuario malintencionado no podrá eliminar el proceso de validación, y los datos siempre se comprobarán. Sin embargo, la desventaja de este método es que el usuario puede rellenar un formulario y es necesario que lo envíe (con la tardanza que eso puede acarrear), se procese en el back-end y al devolver un error, el usuario tenga que retroceder al formulario y en algunos casos, incluso tener que volver a rellenar todos los campos de nuevo.

Caso 4.

- Por último, tendríamos el caso ideal, donde el formulario tiene validación en el front-end y en el back-end, también denominado doble validación. En este caso, el formulario es sometido a un proceso de validación en la parte del front-end, y si lo supera, vuelve a pasar otro proceso de validación en el back-end. La desventaja de este método es que conlleva más trabajo de validación, pero es el sistema recomendado, puesto que es más estricto y sobre todo, más seguro.

- Tradicionalmente, la validación de un formulario se ha hecho siempre en Javascript, sin embargo, HTML5 introduce unos nuevos atributos para formularios que permiten realizar la validación del formulario directamente en HTML5, sin necesidad de recurrir a Javascript (aunque es posible hacerlo si se desea)


Validación
Validación
JS
Validación
php
Front-end
Back-end

Atributos básicos de validación

Atributo	Valor	Puede actuar sobre...	Descripción
minlength	*número*	Campos de texto	Establece la longitud mínima del texto requerida.
maxlength	*número*	Campos de texto	No permite escribir textos superiores a *número* carácteres.
min	*número*	Campos numéricos	Establece el número mínimo permitido.
	fecha	Campos de fecha	Establece la fecha mínima permitida.
	hora	Campos de hora	Establece la hora mínima permitida.
max	*número*	Campos numéricos	Establece el número máximo permitido.
	fecha	Campos de fecha	Establece la fecha máxima permitida.
	hora	Campos de hora	Establece la hora máxima permitida.
step	*número*	Campos numéricos	Establece el salto de números permitido. Por defecto, 1.
	fecha	Campos de fecha	Establece el salto de días permitido. Por defecto, 1.
	hora	Campos de hora	Establece el salto de segundos permitido. Por defecto, 1.
required		Campos en general	Campo obligatorio. Se debe rellenar para enviar formulario.
disabled		Campos en general	Campo desactivado. No se puede modificar. No se envía.
readonly		Campos en general	Campo de sólo lectura. No se puede modificar. Se envía.

Atributos básicos de validación

- Además, utilizando las pseudoclases CSS de validación :valid e :invalid podemos aplicar estilos a los campos <input> y <textarea> teniendo en cuenta su validación.

```css
input:valid, textarea:valid {
  background:green;
}
input:invalid, textarea:invalid {
  background:red;
}
```

Patrones de validación HTML5

- No obstante, aunque los atributos de validación básicos son muy interesantes y pueden facilitarnos la tarea de validación, en muchos casos son insuficientes. Para ello tenemos los patrones de validación HTML5, mucho más potentes y flexibles, que nos permitirán ser mucho más específicos utilizando expresiones regulares para validar datos.
- Una expresión regular es una cadena de texto que representa un posible patrón de coincidencias, que aplicaremos mediante el atributo pattern en los campos que queramos validar.

Patrones de validación HTML5

Expresión regular	Carácter especial	Denominación	Descripción
`.`	Punto	Comodín	Cualquier carácter (o texto de tamaño 1)
`A|B`	*Pipe*	Opciones lógicas	Opciones alternativas (o `A` o `B`)
`C(A|B)`	Paréntesis	Agrupaciones	Agrupaciones alternativas (o `CA` o `CB`)
`[0-9]`	Corchetes	Rangos de carácteres	Un dígito (del `0` al `9`)
`[A-Z]`			Una letra mayúscula de la `A` a la `Z`
`[^A-Z]`	`^` en corchetes	Rango de exclusión	Una letra que no sea mayúscula de la `A` a la `Z`
`[0-9]*`	Asterisco	Cierre o clausura	Un dígito repetido 0 ó más veces (vacío incluido)
`[0-9]+`	Signo más	Cierre positivo	Un dígito repetido 1 ó más veces
`[0-9]{3}`	Llaves	Coincidencia exacta	Cifra de 3 dígitos (dígito repetido `3` veces)
`[0-9]{2,4}`		Coincidencia (rango)	Cifra de 2 a 4 dígitos (rep. de `2` a `4` veces)
`b?`	Interrogación	Carácter opcional	El carácter `b` puede aparecer o puede que no
`\.`	Barra invertida	Escape	El carácter `.` literalmente (no como comodín)

Validación nativa de formularios,

Instructor Jose Luis Sarta Alvarez

- La validación de formularios del lado del cliente ha sido un problema que siempre se ha solucionado con la utilización de **Javascript**, ya sea con código propio o de terceros. Esto es debido a que en el pasado era impensable tener una solución con **HTML** puro, pero con la llegada del conjunto de especificaciones y tecnologías que componen el actual **HTML5**, ahora esto es posible.

- Debemos tener en cuenta la clara desventaja de que todavía no es soportado por la mayoría de los navegadores y quienes lo soportan, ya que cada uno tienen su forma particular de ejecutarse, sin embargo es importante tenerlo presente, ya que en un futuro será de una utilidad invaluable.

El atributo «*required*»

- Al incluir el atributo required dentro de un elemento <input>, automáticamente se hace obligatorio su llenado y al ser un atributo tipo booleano, solo se requiere su presencia sin más.

```
1 | <input type="text" name="nombre" required/>
```

El atributo «*required*»

- En el momento en que se encuentra un error de validación se ejecuta la acción por defecto del navegador, normalmente aparece un pequeño globo emergente (tooltip) conteniendo un texto de advertencia. Añadiendo el atributo title en la etiqueta <input>, se puede extender la información mostrada.

```
1 <input title="Se necesita un nombre" type="text" name="nombre" required/>
```

El atributo type

- Con el atributo type indicamos al navegador el tipo de control que debe ejercer sobre el elemento <input>. Ya en HTML4 nos acostumbramos a <input type="text" /> pero a partir de HTML5 podemos utilizar una amplia gama:

El atributo type

- search: Para cajas de búsqueda
- number: Para añadir o restar números mediante botones.
- range: Para seleccionar valores dentor de un rango.
- color: Para seleccionar un color.
- tel: Para números de teléfono.
- url: Para direcciones web.
- email: Para direcciones de correo electrónico.
- month: Para meses.
- date: Para fechas.
- week: Para semanas.
- time: Para horas.
- datetime: Para una fecha exacta, absoluta y con hora
- datetime-local: Para fechas locales y frecuencia

Ejemplos de patrones HTML5

Ejemplo 1

- Tipo de campo: Nombre de usuario

Campo obligatorio: required.

Entre 5-40 carácteres: minlength="5" maxlength="40"

Sólo se permiten letras (mayúsculas y minúsculas) y números: pattern="[A-Za-z0-9]+"

Ejemplo 1

```HTML
<form name="formulario" method="post" action="http://pagina.com/send.php">
  <!-- Nombre de usuario.
       Campo obligatorio, entre 5-40 carácteres
       Sólo se permiten letras y números -->
  <input type="text" name="nombre" placeholder="Su nombre de usuario"
         minlength="5" maxlength="40" required pattern="[A-Za-z0-9]+">
</form>
```

- Nótese que de no incluir los atributos minlength y maxlength el usuario no tendría limitación en cuanto al tamaño.

Ejemplo 1

- Esto también puede incorporarse en la propia expresión regular, y prescindir de dichos atributos:

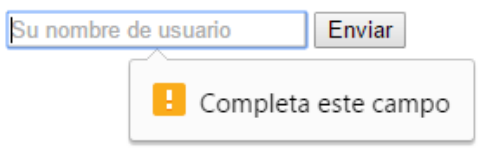
HTML

```html
<form name="formulario" method="post" action="http://pagina.com/send.php">
  <!-- Nombre de usuario.
       Campo obligatorio, entre 5-40 carácteres
       Sólo se permiten letras y números -->
  <input type="text" name="nombre" placeholder="Su nombre de usuario"
         required pattern="[A-Za-z0-9]{5,40}"
         title="Letras y números. Tamaño mínimo: 5. Tamaño máximo: 40">
</form>
```

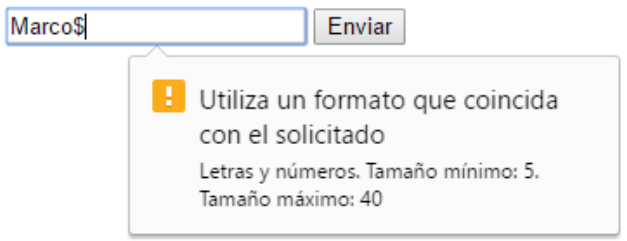
Ejemplo 1

- Sin embargo, en este caso, no se limitará al usuario a la hora de escribir, como hace maxlength, sino que permitirá al usuario escribir la información que desee y en caso de no pasar la validación, mostrará un mensaje de advertencia y no lo dejará continuar hasta que termine. Podemos ampliar el mensaje de advertencia incluyendo el texto en el atributo title.

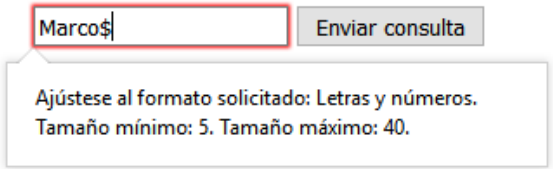
Ejemplo 1



required (Chrome, Windows 10)



pattern (Chrome, Windows 10)



pattern (Firefox, Windows 10)

Ejemplo 2

- En el siguiente caso, se pide al usuario que indique el modelo de coche que posee, en un posible formulario de servicio técnico. Los modelos posibles son A1, A3, A4 y A15. En lugar de mostrar una lista de selección, podemos mostrar un campo de texto y colocar una validación como la siguiente:

- Tipo de campo: Modelo de coche
- Campo obligatorio: required.
- Sólo se permiten las opciones: A1, A3, A4 y A15

Ejemplo 2

```HTML
<form name="formulario" method="post" action="http://pagina.com/send.php">
  <!-- Modelo de coche
       Campo obligatorio, opciones posibles: A1, A3, A4 y A15
       Sólo se permiten letras y números -->
  <input type="text" name="coche" placeholder="Su modelo de coche"
         required pattern="A|a(1|3|4|15)"
         title="Modelos posibles: A1, A3, A4 y A15">
</form>
```

- http://rubular.com/
- https://regex101.com/

Código Ejemplo.

```
<form action="accion.php" method="post">
        <fieldset>
            <legend>EDITOR</legend>
<label for="email">Email *</label>
 <input type="text" name="email" id="email" pattern="^[\w._%-]+@[\w.-]+\.[a-zA-Z]{2,4}$"  placeholder="Introduce un email" required />
<label for="alias">Alias *</label>
<input type="text" name="alias" id="alias" readonly required /> <input type="button" id="edit-button" value="editar" />
<label for="archivo">Archivo (PDF) *</label> <span class="upload"> <input type="file" name="archivo" id="archivo" accept=".pdf" required /> </span>
<label for="imagen">Imagen del manual *</label> <span class="upload"> <input type="file" name="imagen" id="imagen" accept="image/*" required /> </span>
        </fieldset>
<input type="submit" value="Crear" name="boton" />


</form>
```

Entregable

- Problema 1.

Diseñar un formulario que nos solicite la carga obligatoria de un número binario de ocho dígitos.

- Problema 2.

Diseñar un formulario que nos solicite la carga de un número entero decimal que contenga 5 dígitos..

- Problema 3.

Diseñar un formulario que nos solicite la carga de un número entero decimal que contenga entre 1 y 5 dígitos.

- Problema 4.

Diseñar un formulario que nos solicite la carga de una patente de un auto. Toda patente cuenta con tres caracteres seguido de tres números.