

# Tema 2: Desarrollo web en cliente: HTML.

(parte 2)



Bloque 1: Desarrollo web en cliente: HTML y CSS.

Módulo: Lenguajes de marcas y sistemas de gestión de la información.

IES San Vicente  
Curso 2021-2022

# Índice de contenidos

1 Formularios.....	3
1.1 El elemento input.....	4
1.2 Texto alfanumérico libre (texto corto).....	5
1.2.1 Otros textos cortos.....	6
1.2.2 Campos para contraseñas.....	6
1.3 Texto alfanumérico libre (largo).....	7
1.4 Campos numéricos.....	7
1.5 Campos fecha/hora.....	8
1.6 Controles/opciones.....	8
1.7 Listas de selección.....	9
1.8 Campo color.....	10
1.9 Campo de archivo(s).....	11
1.10 Botones.....	11
1.11 Organización de campos.....	12
1.12 Validaciones HTML5.....	12
2 Multimedia.....	14
2.1 Nuevas etiquetas de imágenes.....	14
2.2 Etiquetas de audio.....	16
2.3 Etiqueta de vídeo.....	17
2.4 Etiquetas de contenido externo.....	19

# 1 Formularios

Los formularios es la forma más sencilla para que el usuario interactúe con la web de forma sencilla e intuitiva. Todo el contenido de un formulario debe estar entre la etiqueta **form**, esta etiqueta tiene entre otros los siguientes atributos:

- **method**: El método HTTP que el navegador usa para enviar el formulario. Los valores posibles son:
  - **post**: Corresponde al método [POST HTTP](#), los datos del formulario son incluidos en el cuerpo del formulario y son enviados al servidor.
  - **get**: Corresponde al método [GET HTTP](#), los datos del formulario son adjuntados a la URI del atributo **action**, con un '?' como separador y la URI resultante es enviada al servidor. Use este método cuando el formulario no tiene efectos secundarios y contiene solo caracteres ASCII.
- **action**: la URI de un programa que procesa la información enviada por medio del formulario
- **name**: El nombre del formulario Debe ser único entre los formularios en un documento.
- **autocomplete**: Indica cuales de los controles en este formulario puede tener sus valores automáticamente completados por el navegador, los valores que puede tomar son **on|off**
- **novalidate**: Este atributo previene que el formulario sea validado antes del envío, los valores que puede tener son **true|false**.

```
<form name="..." method="..." action="...">
  <!-- Contenido del formulario -->
</form>
```

Para que un usuario final pueda interactuar, se requiere una serie de campos que le permitan introducir información. Para ello, debemos saber a priori qué tipo de dato le vamos a pedir al usuario (texto, número, fecha, etc.) y así saber qué tipo de campo de entrada es más conveniente poner.

Tipo de información a obtener	Ejemplos	Etiqueta a utilizar
Texto	Nombre, dirección, opinión...	<code>&lt;input type="text"&gt;</code> <code>&lt;textarea&gt;</code>
Texto	Email	<code>&lt;input type="email"&gt;</code>
Texto	Búsqueda en Google	<code>&lt;input type="search"&gt;</code>
Texto	Teléfono	<code>&lt;input type="tel"&gt;</code>
Texto	URL	<code>&lt;input type="url"&gt;</code>
Número	Edad	<code>&lt;input type="number"&gt;</code>
Fechas u horas	Fecha de nacimiento, hora de	<code>&lt;input type="date   datetime  </code>

	comienzo, ...	<code>&lt;datetime-local&gt;</code>
Verdadero / falso	Sí/No, Opción A/B, ON/OFF	<code>&lt;input type="checkbox"&gt;</code>
Opción única	Elige una opción	<code>&lt;input type="radio"&gt;</code> <code>&lt;select&gt;</code>
Varias opciones	Elige la(s) opciones correctas	<code>&lt;input type="checkbox"&gt;</code> <code>&lt;select&gt;</code>
Opción única abierta	Elige una opción o añade una si consideras que no está ahí	<code>&lt;datalist&gt;</code>
Color	Selecciona un color	<code>&lt;input type="color"&gt;</code>
Archivo	Selecciona el archivo a adjuntar	<code>&lt;input type="file"&gt;</code>

A continuación, vamos a ver cada uno de estos campos con algunas de sus peculiaridades. Debemos tener en cuenta que no todos los navegadores soportan por completo las peculiaridades de estos campo de entrada.

## 1.1 El elemento input

La etiqueta más utilizada en los formularios es el input:

```
<form name="formularioEjemplo" method="post" action="#">
  <input type="email" name="email" placeholder="Introduce tu correo" value="">
  <input type="password" name="password" placeholder="Introduce tu pass" value="">
  <input type="submit" value="Enviar">
</form>
```

En este ejemplo tenemos un formulario donde se le pide al usuario que introduzca el email al usuario con la etiqueta `<input type="email">`, la contraseña con la etiqueta `<input type="password">` y hemos insertado un botón con la etiqueta `<input type="submit">` para enviar los datos al servidor. Como vemos, estamos utilizando la misma etiqueta pero dependiendo del tipo de dato a recoger tendrá un tipo (type) u otro.

Los atributos que podemos utilizar de forma general para cualquier campo de entrada `input` son:

Atributo	Valor	Descripción
<b>type</b>	Tipo de campo	Indica el tipo de campo que se trata
<b>name</b>	Nombre del campo	Indica el nombre del campo
<b>value</b>	Valor por defecto	Indica el valor inicial que tendrá ese campo
<b>form</b>	Nombre del formulario	Asocia el campo al nombre del formulario que indica
<b>placeholder</b>	sugerencia	Indica una sugerencia al usuario antes de escribir
<b>size</b>	número	Tamaño visual (número de caracteres) del

		campo de datos
<b>autocomplete</b>	on   off	Activa o desactiva el autocompletado de este campo
<b>autofocus</b>	-	Establece el foco (cursor) en este campo al cargar la página

Por tanto, dependiendo de qué queremos que el usuario introduzca tenemos:

Tipo	Etiqueta & atributo	Cómo se ve
Texto alfanumérico libre (texto corto)	<code>&lt;input type="text"&gt;</code>	<input type="text" value="Texto corto"/>
Texto para búsquedas	<code>&lt;input type="search"&gt;</code>	<input type="text" value="Texto a buscar"/>
Número de teléfono	<code>&lt;input type="tel"&gt;</code>	<input type="text" value="666777888"/>
Dirección URL	<code>&lt;input type="url"&gt;</code>	<input type="text" value="https://www.google.es"/>
Email	<code>&lt;input type="email"&gt;</code>	<input type="text" value="rmedina@iessanvicente.com"/>
Contraseña	<code>&lt;input type="password"&gt;</code>	<input type="password" value="....."/>
Campo oculto (no se muestra al usuario)	<code>&lt;input type="hidden"&gt;</code>	
Texto alfanumérico libre (largo)	<code>&lt;textarea&gt;&lt;/textarea&gt;</code>	<input type="text" value="Texto extenso"/>

Para las etiquetas `input` como `textarea` podemos utilizar un atributo (`spellcheck`) que nos va a permitir revisar la ortografía del texto escrito.

## 1.2 Texto alfanumérico libre (texto corto)

Lo utilizaremos para guardar información de texto libremente, es la opción más habitual, sin ir más lejos cuando no especificamos el atributo `type` en la etiqueta `<input>` se considera que es `type="text"`. Se utiliza para introducir nombre, apellidos, direcciones, etc.

```
<form name="inputText" method="post" action="#">
  <input type="text" name="nombre" autocomplete="off" placeholder="Introduce tu nombre">
</form>
```

Hemos utilizado los siguientes atributos:

- **autocomplete**: autocompletado desactivado (`autocomplete="off"`), es decir, si antes habíamos escrito en ese input no aparecerá.
- **placeholder**: es el texto que aparece en gris para indicarle al usuario qué debe escribir. Ese texto desaparece cuando empezamos a escribir algo y aparecerá en el caso de borrar lo escrito.

**Nota:** Es importante que no pongamos sugerencias en el atributo **value**. El atributo **value** se utiliza para dar un valor por defecto a ese input.

### 1.2.1 Otros textos cortos

Como hemos visto antes, hay otros inputs para introducir textos cortos como **search**, **tel**, **email** o **url**. Aunque la idea es la misma que si ponemos que el campo es de tipo texto porque no notaremos ningún cambio, estos cambios facilitarán la introducción de texto por parte del usuario. Por ejemplo, si accedemos desde el móvil y debemos introducir un teléfono que hemos puesto de tipo **tel** se despliega el teclado numérico en lugar del habitual.

```
<form name="inputText" method="post" action="#" novalidate="true">
  <label>Teléfono</label>
  <input type="tel" name="telefono" placeholder="+XX XXX XXXXXX" >
  <label>Email</label>
  <input type="email" name="correo" placeholder="nombre@dominio.com" >
  <label>Página web</label>
  <input type="url" name="web" placeholder="https://pagina.com/" >
</form>
```

### 1.2.2 Campos para contraseñas

La etiqueta input como hemos visto tiene un tipo de dato que es **password**, este campo nos permite introducir una contraseña sin que el texto sea visible.

```
<form name="inputText" method="post" action="#" novalidate="true">
  <label>Nombre de usuario</label>
  <input type="text" name="nombre" placeholder="Por ejemplo, Juan" >
  <!-- Campo de entrada de password -->
  <label>Contraseña</label>
  <input type="password" name="pass" placeholder="Contraseña" >
  <input type="hidden" name="informacion" value="72625" >
</form>
```

En el formulario anterior, hemos añadido un input de tipo **hidden**, este tipo de dato podemos utilizarlo cuando queremos enviar información en el formulario pero no queremos que el navegador lo muestre en el formulario, por ejemplo un identificador.

### 1.3 Texto alfanumérico libre (largo)

Cuando queremos que el usuario nos introduzca o nos pueda introducir un texto más largo, lo correcto es utilizar la etiqueta `<textarea>` que se utiliza por ejemplo para poner observaciones/comentarios en muchas páginas. Esta etiqueta tiene los siguientes atributos:



- **cols**: número de caracteres que caben en horizontal (columnas)
- **rows**: número de caracteres que caben en vertical (filas)

Un ejemplo básico de esta etiqueta sería:

```
<form name="camposEntrada" method="post" action="#" novalidate="true">
  <textarea name="textoLargo" cols="80" rows="10" placeholder="Introduce la opinión del curso">
    Este es el valor por defecto, no se usa value
  </textarea>
</form>
```

### 1.4 Campos numéricos

Si queremos recoger un número podemos utilizar un campo de tipo de texto, pero lo correcto sería utilizar las etiquetas propias de HTML5 que nos permiten recoger números. Dentro de este apartado nos encontramos con dos opciones, un `input type="number"` o `input type="range"`, donde en lugar de introducir un número con el teclado usaremos una barra de desplazamiento para indicar el número.

Tipo de información	Etiqueta	Cómo se ve
Número o cantidad numérica	<code>&lt;input type="number"&gt;</code>	
Rango numérico	<code>&lt;input type="range"&gt;</code>	

Tanto en una etiqueta como en la otra, podemos establecer un valor mínimo, máximo y/o step (incremento/decremento). Con esto el usuario podrá introducir un número menor al especificado en el valor min, pero no se enviarán los datos al servidor hasta que no introduzca el campo de forma correcta.

```
<form name="formulario" method="post" action="#">
  <!-- Número entre 5 y 50, de 5 en 5. Valor por defecto: 25 -->
  <input type="number" name="numero" value="25" min="5" max="50" step="5" />
  <!-- Su misma versión, utilizando range -->
  <input type="range" name="numrango" value="25" min="5" max="50" step="5" />
  <input type="submit" value="Enviar">
</form>
```

## 1.5 Campos fecha/hora

Si queremos que el usuario nos introduzca una fecha, lo ideal sería utilizar un control llamado datepicker que te permita de un calendario seleccionar la fecha que deseamos (día, mes y año). Si en lugar de una fecha lo que queremos es seleccionar una fecha y hora, usaremos un timepicker.

Vamos a ver los distintos tipos de inputs que HTML5 nos ofrece para insertar una fecha/hora/mes/semana.

Tipo de información	Etiqueta	Cómo se ve
Fecha	<code>&lt;input type="date"&gt;</code>	<input type="text" value="dd/mm/2018"/>
Hora	<code>&lt;input type="time"&gt;</code>	<input type="text" value="--:00"/>
Fecha y hora local	<code>&lt;input type="datetime-local"&gt;</code>	<input type="text" value="dd/mm/aaaa --:--"/>
Mes	<code>&lt;input type="month"&gt;</code>	<input type="text" value="----- de ----"/>
Semana	<code>&lt;input type="week"&gt;</code>	<input type="text" value="Semana --, ----"/>

Ejemplo:

```
<form name="formulario" method="post" action="/send.php">
  <label>Selecciona una fecha</label>
  <input type="date" name="fecha" min="2021-10-25" max="2022-08-25" step="2">
  <label>Selecciona una hora</label>
  <input type="time" name="hora" min="18:00" max="21:00" step="3600">
  <label>Selecciona una fecha y hora</label>
  <input type="datetime-local">
  <input type="submit" value="Enviar">
</form>
```

Como vemos, aquí también podemos indicar los atributos min y max para indicarle un límite de fecha permitida, así como el atributo step.

## 1.6 Controles/opciones

Si queremos que el usuario elija podemos utilizar o casillas de verificación o botones de opción (radio). Elegiremos una opción u otra dependiendo de si le damos la posibilidad de elegir más de una opción o no.

Tipo de información	Etiqueta	Cómo se ve
Botón radio (opción única)	<code>&lt;input type="radio"&gt;</code>	Hombre <input type="radio"/> Mujer <input type="radio"/>
Casilla verificación (0..N opciones)	<code>&lt;input type="checkbox"&gt;</code>	LM <input type="checkbox"/> Programación <input type="checkbox"/>

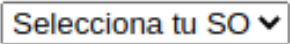


Tanto en la casilla de verificación(**checkbox**), como en el botón radio(**radio**) se puede poner el atributo **checked** para indicar que por defecto aparezca marcada. Por otro lado, en los **input type="radio"** si queremos que sólo seleccione una opción de las que estamos dando, cada input debe tener el atributo **name** con el mismo valor. Ejemplo:

```
<form name="formulario" method="post" action="/send.php">
  <label>Hombre</label>
  <input type="radio" name="sexo" value="hombre">
  <label>Mujer</label>
  <input type="radio" name="sexo" value="mujer">
</form>
```

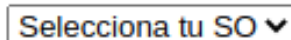
## 1.7 Listas de selección

Además de los controles/opciones vistos en el punto anterior, tenemos otra forma de darle a elegir al usuario entre varios valores mediante listas:

Tipo de información	Etiqueta	Cómo se ve
Lista (cerrada) de opciones	<pre>&lt;select&gt;   &lt;option&gt;...&lt;/option&gt; &lt;/select&gt;</pre>	
Lista (abierta) de opciones	<pre>&lt;datalist&gt;</pre>	

La lista de selección más habitual es **<select>**. En su interior debemos poner todas las opciones posibles utilizando la etiqueta **<option>**, y el usuario elegirá de entre todas ellas una. Si queremos que por defecto haya una seleccionada, deberemos ponerle a ese **<option>** el atributo de **selected**.

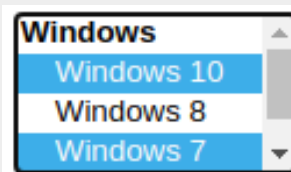
```
<form name="formulario" method="post" action="/send.php">
  <select name="sistema" >
    <option value="ninguno">Selecciona tu SO</option>
    <option value="windows">Windows</option>
    <option value="linux">Linux</option>
    <option value="osx">MacOS</option>
    <option value="unix">Unix</option>
  </select>
</form>
```



Si por ejemplo de la lista cerrada que hemos creado anteriormente queremos que el usuario elija varias opciones, a la etiqueta **<select>** le podemos poner el atributo **multiple** y con eso le permitimos al usuario que elija uno o más valores. En el ejemplo anterior, puede ser que tenga instalado en su ordenador el sistema operativo Linux y Windows. Para poder seleccionar más de una opción deberá pulsar la tecla CTRL y marcar las opciones de la lista que quiera.

Hay otra "variante" de esta etiqueta, y es utilizada para cuando queremos tener agrupadas varias etiquetas de forma que la lista de opciones estén más organizadas. Para ello cada uno de los grupos de **<option>** que queramos hacer los agruparemos dentro de una etiqueta **<optgroup>**.

```
<form name="formulario" method="post" action="/send.php">
  <select name="sistema" multiple>
    <optgroup label="Windows">
      <option value="windows10" label="Windows 10"></option>
      <option value="windows8" label="Windows 8"></option>
      <option value="windows7" label="Windows 7"></option>
      <option value="windowsVista" label="Windows Vista"></option>
      <option value="windowsXP" label="Windows XP"></option>
    </optgroup>
  </select>
</form>
```



Si nos fijamos en lugar de poner entre la etiqueta `<option>` el texto a mostrar en el navegador lo hemos puesto dentro del atributo `label`, y es que es un formato alternativo, podemos usar ambas formas que serán correctas.

El último ejemplo de listas que tenemos son las listas seleccionables abiertas, se caracterizan porque el usuario puede seleccionar opciones sugeridas mediante un `<datalist>` o bien indicar la suya propia escribiéndola. La diferencia que tiene con los `<select>` es que visualmente no muestra nada.

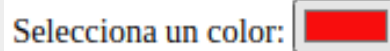
```
<form name="formulario" method="post" action="/send.php">
  <label>¿Qué navegador usas?:</label>
  <input list="browsers" name="myBrowser" >
  <datalist id="browsers">
    <option value="Chrome"></option>
    <option value="Firefox"></option>
    <option value="Internet Explorer"></option>
    <option value="Opera"></option>
    <option value="Safari"></option>
    <option value="Microsoft Edge"></option>
  </datalist>
</form>
```



## 1.8 Campo color

HTML5 incluye un nuevo campo llamado colorpicker, el cual nos permite seleccionar un color específico especificando sus colores (RGB).

```
<form name="formulario" method="post" action="/send.php">
  <label>Selecciona un color: </label>
  <input type="color" name="color" value="#F4F4F4">
</form>
```



Podemos utilizar el atributo `value` para darle un valor por defecto.

## 1.9 Campo de archivo(s)

Cuando queremos que el usuario nos adjunte un documento, debemos utilizar la etiqueta `<input type="file">`. Antes de nada, deberemos tener en cuenta que a la etiqueta `<form>` debemos indicarle el atributo `enctype="multipart/form-data"` como señal de que se van a adjuntar archivos en nuestro formulario. Si queremos restringir que el usuario sólo nos pueda adjuntar un cierto tipo de documentos, utilizaremos el atributo `accept` con la lista de extensiones permitidas separadas por comas.

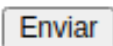

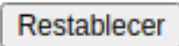
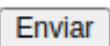
```
<form name="formulario" method="post" action="/send.php" enctype="multipart/form-data">
  <input type="file" name="adjunto" accept=".pdf,.jpg,.png" multiple />
</form>
```

Elegir archivos Ningún archivo seleccionado

**Nota:** Si ponemos el atributo `multiple` en la etiqueta `<input>` estaremos dando la posibilidad de adjuntar varios archivos

## 1.10 Botones

Si un formulario no tiene un botón para poder enviar los datos del formulario sólo podría enviarlos si pulsa ENTER en el último campo. Lo aconsejable aun así es siempre introducir un botón para que el usuario envíe el formulario. Los tipos de botones que tenemos son:



Tipo de botón	Etiqueta & atributo	Cómo se ve
Botón de envío	<code>&lt;input type="submit"&gt;</code>	
Botón de envío con imagen	<code>&lt;input type="image" src="enviar.png" width="80" height="28"&gt;</code>	
Botón para borrar el formulario	<code>&lt;input type="reset"&gt;</code>	
Botón sin funcionalidad	<code>&lt;input type="button" value="Enviar"&gt;</code> <code>&lt;button&gt;Enviar&lt;/button&gt;</code>	

El botón que se usa de forma habitual es `<input type="submit">`, sirve para enviar el formulario una vez que el usuario ha rellenado los campos y pulsado el botón. Por defecto el texto del botón es Enviar, pero se puede modificar mediante el atributo `value`. El botón de tipo imagen (`<input type="image">`) tiene la misma funcionalidad que el anterior pero con los atributos `src`, `alt`, `width` y `height` como la etiqueta `img` para indicarle la imagen que hará la función de botón. El botón reset (`<input type="reset">`) nos ofrece la posibilidad de limpiar los campos de un formulario dejándolos a sus valores por defecto.

Por último, tenemos el botón (`<input type="button">`) o la etiqueta (`<button>`) que en ambos casos añade un botón sin ninguna funcionalidad por defecto. Será a posteriori con JS cuando veamos cómo darle una funcionalidad.

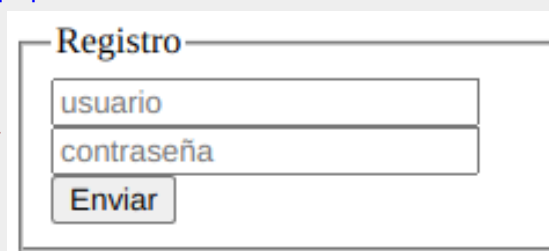
## 1.11 Organización de campos

Hay una serie de etiquetas que aunque no se utilizan, sirven para organizar mejor los elementos de un formulario ya sea por categorías/temáticas, etc. Estas etiquetas son:

Tipo	Etiqueta	Cómo se ve
Agrupación visual o temática de campos	<code>&lt;fieldset&gt; &lt;/fieldset&gt;</code>	
Leyenda para la etiqueta <code>fieldset</code>	<code>&lt;legend&gt;Leyenda&lt;/legend&gt;</code>	
Relación de campo y texto	<code>&lt;label&gt;texto&lt;/label&gt;</code>	

La primera etiqueta es `fieldset`, se utiliza como etiqueta contenedora para agrupar mediante un recuadro todos los campos que están relacionados de un formulario. La etiqueta `legend` se suele incluir en su interior para mostrar un título sobre lo que agrupamos.

```
<form name="formulario" method="post" action="/send.php">
  <fieldset>
    <legend>Registro</legend>
    <input type="text" placeholder="usuario">
    <input type="password" placeholder="contraseña">
    <input type="submit">
  </fieldset>
</form>
```



La etiqueta `<label>` nos permite establecer una relación semántica de un texto con un campo de entrada de datos. Esta etiqueta suele tener el atributo `for` con un nombre específico para establecer la relación de una etiqueta HTML con el id con el mismo nombre.

## 1.12 Validaciones HTML5

Cuando creamos un formulario debemos tener en cuenta que los usuarios se pueden equivocar a la hora de rellenar nos datos del mismo, y por tanto, debemos anticiparnos a estos e intentar que los datos lleguen correctamente al servidor sin ningún tipo de error.

Para evitar que los datos lleguen con errores debe haber un proceso de validación, esta validación se debe realizar tanto en la parte front-end (cliente, JS) como en la parte back-end(servidor, PHP). Tradicionalmente la validación de un formulario siempre la ha hecho JavaScript, pero HTML5 introduce nuevos atributos que nos permiten validar los campos de un formulario. Estos atributos que nos permiten validar un formulario son:

Atributo	Valor	Se aplica a...	Descripción
<code>minlength</code>	Número	Campos de texto	Longitud mínima del texto
<code>maxlength</code>	Número	Campos de texto	Longitud máxima del

			texto
min	número/fecha/hora	Campos numéricos/fecha/hora	Número/fecha/hora mínimo permitido
max	número/fecha/hora	Campos numéricos/fecha/hora	Número/fecha/hora máximo permitido
step	número/fecha/hora	Campos numéricos/fecha/hora	Salto de números/fechas/horas permitidas. Por defecto el valor es 1
required		Campos en general	Campo obligatorio. Se debe rellenar para enviar formulario
disabled			Campo desactivado. No se puede modificar ni se envía.
readonly			Campo de solo lectura. No se puede modificar, si se envía

Además de estos atributos, tenemos una serie de patrones para validar los campos en HTML utilizando expresiones regulares para validar los datos. Una expresión regular no es más que una cadena que representa un posible patrón de coincidencias. Si queremos utilizar un patrón, deberemos utilizar el atributo `pattern` y como valor la expresión regular que deba cumplir.

Expresión	Carácter especial	Significado	Descripción
.	Punto	Comodín	Cualquier carácter (longitud 1)
A B	Barra vertical (pipe)	Opciones lógicas	Opciones alternativas (A o B)
C(A B)	Paréntesis	Agrupaciones	Agrupaciones alternativas (o CA o CB)
[0-9]	Corchetes	Rangos de caracteres	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	0 o más dígitos
[0-9]+	Signo más	Cierre positivo	1 o más dígitos

[0-9]{3}	Llaves	Coincidencia	Cifra de 3 dígitos
[0-9]{2,4}			Cifra de 2 a 4 dígitos
B?	Interrogación		El carácter B puede aparecer o no
\.	Barra invertida		El carácter . Literal (no como comodín como en el primer ejemplo)

Ejemplo: Nombre de usuario requerido con una longitud entre 5 y 40 caracteres. Sólo se permiten letras y número

Opción A:

```
<form name="formulario" method="post" action="/send.php">
  <input type="text" name="nombre" placeholder="Nombre de usuario" minlength="5"
maxlength="40" required pattern="[A-Za-z0-9]+">
</form>
```

Opción B:

```
<form name="formulario" method="post" action="/send.php">
  <input type="text" name="nombre" placeholder="Nombre de usuario" required pattern="[A-Za-z0-9]{5,40}" title="Válido sólo letras y números. (mín: 5, max: 40)">
</form>
```

## 2 Multimedia

### 2.1 Nuevas etiquetas de imágenes

HTML5.1 incorpora nuevas etiquetas de imágenes además de la etiqueta que había antes (`<img>`):

Etiqueta	Atributos	Descripción
<code>&lt;picture&gt;</code>		Agrupar una serie de imágenes. Es una etiqueta contenedora
<code>&lt;source&gt;</code>	<code>srcset</code> , <code>sizes</code> , <code>media</code> , <code>type</code>	Muestra la imagen que cumpla una serie de criterios opcionales

Como podemos ver, la etiqueta `<source>` tiene una serie de atributos disponibles para utilizar:

- `srcset`: varas imágenes separadas por comas. (atributo obligatorio)
- `sizes`: Tamaño con el que queremos que se visualice la imagen
- `media`: Condición (media queries, CSS) a cumplir para que la imagen se muestre.
- `type`: Tipo de formato de imagen (Atributo opcional)

Una de las ventajas que tiene estas etiquetas es que podemos utilizar diferentes formatos de la imagen dependiendo si el navegador soporta ese formato o no.

```
<picture>
  <source srcset="imagen.webp"><!-- Formato WebP -->
  <source srcset="imagen.jxr"><!-- Formato JPEG XR -->
  
</picture>
```

En el ejemplo anterior, lo que estamos diciendo es que utilice primero la imagen en formato WebP, si no soporta este formato pasará a mostrar el formato JPEG XR, pero si tampoco lo soporta mostrará entonces la imagen JPEG que es soportada por todos los navegadores a excepción de aquellos que no muestran imágenes como Lynx que mostrará el texto alternativo (**alt**).

Utilizando el atributo **media**, podemos crear imágenes responsive que cambien dependiendo del min-width/max-width de la pantalla.

```
<picture>
  <source media="(min-width: 600px)" srcset="imagen-xl.png">
  <source media="(min-width: 300px) and (max-width: 600px)" srcset="imagen-large.png">
  <source media="(max-width: 50px)" srcset="imagen-small.png">
  
</picture>
```

De esta forma, estamos haciendo que dependiendo de la resolución de la pantalla (del ancho en concreto) se muestre una imagen u otra:

- Dispositivos con una resolución de pantalla mayor a 600px: [imagen-xl.png](#)
- Dispositivos con una resolución entre 300 y 600px: [imagen-large.png](#)
- Dispositivos con una resolución menor a 50px: [imagen-small.png](#)
- Dispositivos que no cumple las condiciones anteriores o no soporta HTML5.1: [imagen-medium.png](#)

Por último, esta etiqueta nos permite indicar diferentes imágenes dependiendo la densidad de la pantalla (alto). Para ello, tenemos que usar un descriptor tras el nombre de la imagen con el atributo **srcset** (el valor por defecto es 1x)

```
<picture>
  <source media="(min-width: 600px)" srcset="imagen-xl.png, imagen-xl-hd.png 2x, imagen-xl-fhd.png 3x" />
  <source media="(min-width: 300px) and (max-width: 600px)" srcset="imagen-large.png, imagen-large-hd.png 2x, imagen-large-fhd.png 3x" />
  <source media="(max-width: 50px)" srcset="imagen-small.png, imagen-small-hd.png 2x, imagen-small-fhd.png 3x" />
  <img srcset="imagen-medium.png, imagen-medium-hd.png 2x, imagen-medium-fhd.png 3x" alt="HTML5 logo" />
</picture>
```

## 2.2 Etiquetas de audio

Antiguamente si queríamos añadir a nuestra web música/sonidos se utilizaba la etiqueta obsoleta `<bgsound>`. Hoy en día se utiliza la etiqueta `<audio>`

Los atributos que podemos utilizar con esta etiqueta son:

Atributo	Valor	Descripción
<code>src</code>	URL	Audio a reproducir. Obligatoria si está como etiqueta contenedora.
<code>preload</code>	auto   metadata   none	Indica cómo realizar la precarga del audio
<code>mediagroup</code>	nombre	Establece el nombre para un grupo de contenidos multimedia
<code>autoplay</code>	boolean	Comienza a reproducir el audio automáticamente o no
<code>loop</code>	boolean	Se reproduce en modo bucle
<code>muted</code>	boolean	Pone el audio en silencio
<code>controls</code>	boolean	Muestra los controles de reproducción. Por defecto no se muestran.

```
<audio src="sonido.mp3"></audio>
```

En el ejemplo anterior no se va a ver nada visualmente, si se reproducirá nada. Al no poner los controles(`controls`) el usuario no pulsará play, ni tampoco hemos puesto el `autoplay` y por tanto no se reproducirá automáticamente.

```
<audio src="sonido.mp3" preload="none" controls></audio>
<audio src="sonido.ogg" autoplay loop></audio>
```

En el primer ejemplo anterior, con el atributo(`preload="none"`) estamos diciendo que no precargue nada, de modo que el audio se descargue cuando el usuario pulse los controles de reproducción evitando por tanto el consumo de ancho de banda en aquellos audios que el usuario probablemente no escuche. El segundo ejemplo, está cargando un audio en formato OGG, y lo reproduce automáticamente y en bucle, es decir, que una vez que finaliza el audio vuelve a empezar.

A continuación, vamos a ver qué formatos de audio son más o menos recomendados utilizar en una web:

Formato	Codec	Características	¿Recomendado?
<code>MP3</code>	MP3 Layer-3	Buena calidad	Sí
<code>AAC</code>	Advanced Audio Coding	Mejora el MP3, usado como audio en MP4	Sí
<code>OGG</code>	Ogg Vorbis	Buena calidad, alternativa libre a MP3	Sí



Opus	Opus	Buena calidad, alternativa libre también a MP3	Sí
FLAC	FLAC Audio Lossless	Compresión sin pérdidas. Gran tamaño	Sí
WAV	Wave sound	Formato de Microsoft.	No, muy pesado

Como hemos dicho antes, la etiqueta `<audio>` puede utilizarse como etiqueta contenedora e incluir varias etiquetas HTML para una mayor compatibilidad o capacidades adicionales:

Etiqueta	Atributos	Descripción
<code>&lt;audio&gt;</code>	<code>src, type</code>	Establece un archivo de audio o lo añade como alternativa
<code>&lt;track&gt;</code>	<code>src, srclang, label, kind, default</code>	Establece un archivo de subtítulos o lo añade como alternativa

De esta forma estamos pasando a usar la etiqueta `<audio>` un poco más avanzada. Veamos un ejemplo:

```
<audio>
  <source src="audio.opus" >
  <source src="audio.ogg" >
  <source src="audio.mp3" >
</audio>
```

En este ejemplo el navegador intenta reproducir el archivo Opus, en el caso de no ser soportado por el navegador intentará reproducir OGG y en el caso de no ser soportado pasa a reproducir el MP3

## 2.3 Etiqueta de vídeo

Con HTML5 tenemos la posibilidad de mostrar/reproducir vídeo directamente en nuestro navegador utilizando las etiquetas `<video>` y `<source>`. Debemos tener en cuenta que estos vídeos no son vídeos de Youtube/Vimeo/otro servicio, para ello se utiliza la etiqueta `iframe` como veremos en el siguiente punto (etiquetas de contenido externo)

Los atributos que podemos utilizar en la etiqueta `<video>` son:

Atributo	Valor	Descripción
<code>src</code>	URL	Vídeo a reproducir. Obligatoria si está como etiqueta contenedora.
<code>poster</code>	URL	Muestra una imagen a modo presentación
<code>preload</code>	auto   metadata   none	Indica cómo realiza la precarga del vídeo
<code>mediagroup</code>	nombre	Establece un nombre para un grupo de contenidos multimedia
<code>autoplay</code>	boolean	Comienza a reproducir el vídeo automáticamente

loop	boolean	Se reproduce el vídeo en modo bucle
muted	boolean	Silenciamos el vídeo
controls	boolean	Muestra los controles de reproducción. Por defecto no se muestran
width	tamaño	Indica el tamaño de ancho del vídeo
height	tamaño	Indica el tamaño del alto del vídeo

```
<video src="video.mp4" width="640" height="480"></video>
```

Con el elemento anterior, estamos insertando un vídeo en nuestra web de tamaño 640x480, pero se verá como una imagen al no haber puesto los controles(**controls**) del vídeo ni tener la auto-reproducción(**autoplay**) de forma automática.

```
<video src="video.webm" poster="intro.jpg" controls></video>
<video src="video.mp4" autoplay muted loop></video>
```

En el primer ejemplo se mostrará una imagen (intro.jpg) hasta que el usuario pulse a reproducir el vídeo. En el segundo ejemplo, el vídeo se reproduce automáticamente al cargar la página, en silencio y en bucle.

**Nota:** Desde 2017, Chrome, Firefox y otros navegadores establecieron un cambio de políticas con el atributo de reproducción automática **autoplay**, por lo que no funcionará salvo que el usuario haya interactuado antes en la página.

Los formatos/codecs de vídeo más conocidos y utilizados son:MP4(x264, DivX H264) te permite una alta calidad de vídeo y tienen buen soporte. Algo parecido pasa con WebM (VP8, VP9), es una alternativa libre a MP4 de Google.

Al igual que el audio, podemos poner la etiqueta de vídeo como etiqueta contenedora

```
<video width="640" height="480">
  <source src="video.mp4" type="video/mp4">
  <source src="video.webm" type="video/webm">
  <source src="video.ogv" type="video/ogg">
  
</video>
```

## 2.4 Etiquetas de contenido externo

Para introducir contenido externo en nuestra página como puede ser servicios de Youtube, Vimeo, SoundCloud, etc. necesitaremos una de las siguientes etiquetas:

Etiqueta	Atributos	Descripción
<code>&lt;iframe&gt;</code>	<code>src, srcdoc, name, width, height</code>	Permite incrustar contenido externo en "vivo"
<code>&lt;embed&gt;</code>	<code>src, type, width,height</code>	Permite incrustar contenido interactivo
<code>&lt;object&gt;</code>	<code>data, type, name, form, width, height</code>	Permite incrustar contenido externo con fallbacks
<code>&lt;param&gt;</code>	<code>name, value</code>	Define parámetros de un elemento <code>&lt;object&gt;</code>

Por ejemplo, si queremos insertar un vídeo de YouTube (también válido para SoundCloud) usaremos la etiqueta `<iframe>`:

```
<iframe width="560" height="315" src="https://www.youtube.com/embed/CM_A4f9qqzo"
title="YouTube video player" ></iframe>
```



**Nota:** Este iframe lo podemos conseguir si desde YouTube pulsamos en compartir → Insertar

Otro de los ejemplos más típicos es el de insertar un iframe para poner una ubicación.

```
<iframe src="https://www.google.com/maps/embed?pb=!1m18!1m12!1m3!1d12506.093962966048!
2d-0.5318828639471987!3d38.40626542438854!2m3!1f0!2f0!3f0!3m2!1i1024!2i768!4f13.1!3m3!1m2!
1s0xd6233fea3991575%3A0xf28fcef8c48c1513!2sIES%20San%20Vicente!5e0!3m2!1ses!2ses!
4v1635452091546!5m2!1ses!2ses" width="600" height="450" style="border:0;" allowfullscreen=""
loading="lazy"></iframe>
```



**Nota:** Del mismo modo que con el vídeo de YouTube, el mapa lo conseguimos pulsando desde Google Maps en Compartir → Insertar mapa.

Como ejemplo de la etiqueta object:

```
<object type="application/pdf" data="tema2.pdf" width="250" height="200"></object>
```

