

Tema 1: HTML y CSS

Parte 1: HTML

HTML



BLOQUE 1: Desarrollo web en cliente - HTML y CSS

Módulo: Desarrollo de Interfaces Web

Ciclo: Desarrollo de Aplicaciones Web

Curso: 2023 – 2024

Profesora: Rosa Medina

Índice

1. Estructura de etiqueta HTML.....	4
1.1. Anidamiento de elementos.....	5
1.2. Elementos vacíos.....	5
2. Estructura básica de un documento HTML.....	5
2.1. DOCTYPE.....	6
2.2. Elementos de la cabecera.....	6
2.2.1. Las metaetiquetas, <meta>.....	6
charset.....	7
description.....	7
keywords.....	7
author.....	7
viewport.....	7
2.2.2. Elemento <title>.....	8
2.2.3. Elemento <script>.....	8
3. Ejemplo de página web.....	8
4. Comentarios en HTML.....	9
5. Etiquetas de secciones.....	9
5.1. Elementos de diseño HTML.....	10
5.2. Encabezado <header>.....	10
5.3. Menú <nav>.....	11
5.4. Contenido principal <main>.....	11
5.5. Elemento <article>.....	12
5.6. Elemento <section>.....	12
5.7. Barra lateral <aside>.....	13
5.8. Pie de página <footer>.....	14
5.9. Búsqueda de contenido <search>.....	15
5.10. Información contacto/autor <address>.....	15
5.11. Los encabezados de título <h1>.....	16
6. Etiquetas de texto.....	16
6.1. Párrafos.....	17
6.2. Resaltado de texto: Negrita y cursiva	17
6.3. La etiqueta <s>.....	18
6.4. La etiqueta	18
6.5. Citas: etiquetas <cite>, <blockquote> y <q>.....	18
6.6. La etiqueta <u>.....	19
6.7. La etiqueta <sup> y <sub>.....	19
6.8. La etiqueta <abbr> y <dfn>.....	20
6.9. Texto preformateado: etiquetas <pre> y <code>.....	20
6.10. Marcado de texto: etiqueta <mark>.....	21
6.11. Borrados e inserciones: etiquetas e <ins>.....	21
6.12. La etiqueta <time> y <data>.....	21
6.13. La etiqueta <small>.....	22
6.14. La etiqueta , <wbr> y <hr>.....	22
6.15. La etiqueta <kdb> y <samp>.....	23
7. Enlaces <a>.....	23
7.1. Ruta absoluta y ruta relativa.....	24
8. Etiquetas multimedia.....	24

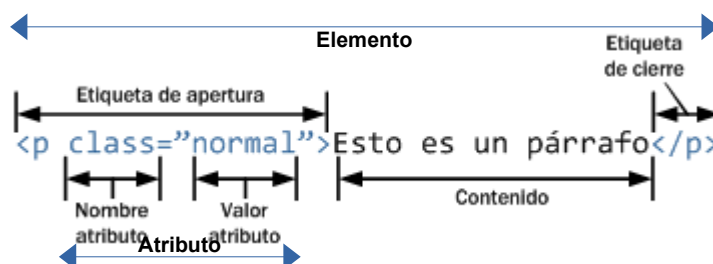
8.1. Imágenes	24
8.1.1. Imágenes y enlaces.....	25
8.1.2. Favicon (icono de pestaña).....	25
8.1.3. Elemento <figure>.....	26
8.1.4. Nuevas etiquetas de imágenes.....	26
8.2. Etiquetas de audio.....	27
8.3. Etiqueta de vídeo.....	29
8.4. Etiqueta <track>.....	30
8.5. Etiquetas de contenido externo.....	32
9. Tablas, <table>.....	34
10. Agrupación de etiquetas.....	37
10.1.1. Elemento <div>.....	37
10.2. Listas.....	37
10.2.1. Listas no ordenadas	37
10.2.2. Listas ordenadas	38
10.2.3. Listas anidadas.....	38
10.2.4. Listas de descripción: etiquetas <dl>, <dt> y <dd>.....	39
11. Formularios.....	39
11.1. El elemento input.....	41
11.2. Texto alfanumérico libre (texto corto).....	42
11.2.1. Otros textos cortos.....	42
11.2.2. Campos para contraseñas.....	43
11.3. Texto alfanumérico libre (largo).....	43
11.4. Campos numéricos.....	44
11.5. Campos fecha/hora.....	44
11.6. Controles/opciones.....	45
11.7. Listas de selección.....	45
11.8. Campo color.....	47
11.9. Campo de archivo(s).....	47
11.10. Barra de progreso <progress>.....	47
11.11. Etiqueta para medidores <meter>.....	48
11.12. Botones.....	48
11.13. Organización de campos.....	49
12. Validación de formularios.....	49

Una página web es un archivo de texto con una serie de etiquetas de marcas que contienen la información a mostrar en el navegador.

El lenguaje HTML (“Hypertext markup language”, en castellano “Lenguaje de marcas de hipertexto”) consiste en un conjunto de etiquetas predefinidas que el navegador web interpreta a la hora de construir una página para su visualización. Es el lenguaje utilizado para estructurar una página web.

1. Estructura de etiqueta HTML

Cuando trabajamos con HTML lo haremos con elementos, etiquetas y atributos. Ejemplo:



Un **elemento** es un componente del documento que tiene un significado semántico (un párrafo, una cabecera, un listado, etc.). Las partes principales por las que están formados los elementos son:

- **La etiqueta de apertura:** consiste en el nombre del elemento (en este caso es la p de párrafo), encerrado por paréntesis angulares (< >) de apertura y cierre. Esta etiqueta indica dónde comienza el elemento, en este caso el párrafo.
- **La etiqueta de cierre:** es igual que la etiqueta de apertura, excepto que incluye una barra de cierre (/) antes del nombre de la etiqueta. Establece dónde termina el elemento, en este caso el párrafo.
- **El contenido:** este es el contenido del elemento, que en este caso es sólo texto (“Esto es un párrafo”).

Por tanto, el elemento está formado por la etiqueta de apertura, el contenido y la etiqueta de cierre.

```
<p class="saludo">Hola mundo!</p>
```

El diagrama muestra la etiqueta de apertura `<p class="saludo">` y la etiqueta de cierre `</p>`. El atributo `class="saludo"` se identifica como "Un atributo y su valor", y el texto `Hola mundo!` entre las etiquetas se identifica como "Contenido de texto encerrado".

Los **atributos** contienen información adicional acerca del elemento, en el ejemplo anterior *class* es el nombre del atributo y *saludo* el valor del atributo.

Un atributo siempre debe tener:

1. Un espacio entre este y el nombre del elemento (o del atributo previo si hay más).
2. El nombre del atributo seguido del signo =

3. Comillas de apertura y de cierre, encerrando el valor del atributo.

Los atributos siempre se encuentran en la **etiqueta de apertura**, **nunca** en la de cierre.

1.1. Anidamiento de elementos

Podemos también tener elementos dentro de otros elementos (anidamiento). En el siguiente ejemplo vamos a poner en negrita las palabras *se llama* del párrafo.

```
<p>Mi perra <strong>se llama</strong> Lola</p>
```

Debemos asegurarnos que los elementos están correctamente anidados, es decir, el último elemento que abrimos deberá ser el primero en cerrar, y así sucesivamente. En el siguiente ejemplo donde la negrita se le pone a la palabra *Lola* **es incorrecto** al haber cerrado primero el párrafo y luego la negrita.

```
<p>Mi perra se llama <strong>Lola</p></strong>
```

Los elementos deben abrirse y cerrarse ordenadamente. Si estos se encuentran solapados el navegador web intentará adivinar lo que intentas hacer y puede que el resultado no sea el deseado.

1.2. Elementos vacíos

Algunos elementos no poseen contenidos, y son llamados elementos vacíos. Por ejemplo, el elemento imagen.

```

```

El ejemplo imagen anterior tiene dos atributos, pero no hay etiqueta de cierre `` ni contenido encerrado. Esto es porque es un elemento de imagen, no encierra contenido.

2. Estructura básica de un documento HTML

En general, cualquier página web en HTML debe tener la extensión `.html`. El contenido de una página por un navegador, encargado de mostrar el contenido de la misma.

Un documento HTML tiene una estructura básica similar a la siguiente:

```
<!DOCTYPE ...>
<html ...>
  <head>
    ...
  </head>
  <body>
    ...
  </body>
</html>
```

En esta estructura nos encontramos:

- `<!DOCTYPE html>`: Nos indica el tipo de documento que es, es obligatorio.
- `<html>...</html>`: Este elemento encierra todo el contenido de la página entera, se le conoce como el elemento raíz. Es recomendable incluir dentro de la etiqueta `html` el atributo `lang` seguido del idioma de la página. En el siguiente ejemplo veremos cómo indicar que el documento está en español (`lang="es"`), pero este atributo no es obligatorio, hoy en día desde la versión HTML5 este atributo se puede prescindir y el documento validaría y funcionaría correctamente.

```
<html lang="es">
....
</html>
```

- `<head>...</head>`: Actúa como un contenedor de todo aquello que queremos incluir en la web pero que no es contenido visible (metadatos). Dentro de este elemento se incluyen las *keywords*, las hojas de estilo CSS, la declaración de caracteres, etc.
- `<body>...</body>`: Encierra todo el contenido que desees mostrar, ya sea texto, imágenes, etc.

2.1. DOCTYPE

Como hemos visto en el punto anterior es lo primero que aparece en un documento html y sirve para indicar qué versión del lenguaje se está utilizando.

```
<!DOCTYPE html>
```

2.2. Elementos de la cabecera

La siguiente parte del documento es la sección `<head>`, donde deberemos incluir:

- `<title>`: El título de la página web
- `<meta>`: Las metaetiquetas que describen el documento, la codificación y la información a la que pueden acceder los motores de búsqueda (*keywords*)
- Las referencias a archivos externos que necesita la web como hojas de estilos(`<link>`) o scripts(`<script>`), estos elementos los veremos en las siguientes unidades.

2.2.1. Las metaetiquetas, `<meta>`

Las metaetiquetas se utilizan para indicar el juego de caracteres, la descripción de la página, las palabras clave, el autor del documento y la configuración del área visible e una página (`viewport`), que será diferente dependiendo del dispositivo.

Las metaetiquetas utilizan dos atributos: `name` para indicar el nombre de la metaetiqueta y `content`, para indicar su contenido, a excepción de la metaetiqueta de la codificación de caracteres que sólo utiliza el atributo `charset`.

```
<meta charset="UTF-8">
<meta name="description" content="Mi primera página web en html5">
<meta name="keywords" content="web html5">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

charset

Define la codificación de caracteres que se utiliza en el documento. Este elemento se ha simplificado en HTML5, y se reduce a:

```
<meta charset="utf-8">
```

La codificación predeterminada en HTML5 es UTF-8, que cubre casi todos los caracteres, signos de puntuación y símbolos del mundo.

Es importante que la declaración de caracteres se encuentre dentro de los 512 primeros caracteres del documento. También es importante que esté antes de cualquier elemento de contenido (como el `<title>`). Una buena práctica sería poner la codificación como la primera etiqueta dentro del bloque `<head>`.

description

Se utiliza para describir brevemente el contenido de una página web. La descripción aparece debajo del título y URL de la página en los resultados del motor de búsqueda. Para permanecer visible dentro de Google debe tener entre 140-160 caracteres. Se recomienda que para tener un buen posicionamiento tenga una descripción de forma significativa y que incluya alguna de sus palabras claves.

```
<meta name="description" content="Mi primera página web en html5">
```

keywords

Define una serie de palabras relacionadas con el contenido de la página web que pueden resultar útiles para los motores de búsqueda, aunque su uso cada vez tiene menos influencia en el posicionamiento web, por ejemplo, Google lo ignora.

```
<meta name="keywords" content="web html5">
```

author

Indica el nombre del autor de la página web.

```
<meta name="author" content="Rosa Medina">
```

viewport

Configura el área visible de la página web para que se visualice bien en todos los dispositivos.

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

Con el anterior código estamos indicándole al navegador:

- `width=device-width`: Ajusta el ancho de la página al ancho de la pantalla del dispositivo.
- `initial-scale=1.0`: Establece el nivel de zoom inicial cuando el navegador carga la web por primera vez. Al poner 1 no se aplica zoom, si por ejemplo no queremos que se pueda hacer zoom se pondría: `maximun-scale=1.0`

2.2.2.Elemento <title>

El elemento `<title>` es el título de la página web, se escribe entre las etiquetas `<title>` y `</title>` y se muestra en la barra de título del navegador.

```
<title>Mi primera página web</title>
```

2.2.3.Elemento <script>

Para utilizar JavaScript en nuestra página, debemos indicarle a nuestro HTML que vamos a cargar un script, para ello, utilizaremos la etiqueta `<script>`. Esta etiqueta admite dos atributos:

- `src`: Dirección URL del script externo a cargar
- `type`: Este atributo puede ser:
 - (sin atributo type): Es el script clásico, se carga en modo tradicional
 - `module`: Carga JavaScript como módulo (permite import y export)

De momento, en el caso de tener que usar algún script nos centraremos en el script clásico.

3. Ejemplo de página web

Un ejemplo de una página web sencilla sería:

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <meta name="description" content="Mi primera página web en html5">
  <meta name="keywords" content="web html5">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Mi primera página web</title>
</head>
<body>
  <p>Mi perra se llama <strong>Lola</strong></p>
</body>
</html>
```

Para aquellos que habéis estado trabajando con vscode, sabréis que si queremos crear un archivo HTML con las etiquetas básicas bastará con usar la combinación de teclas: **! + Enter**.

4. Comentarios en HTML

HTML permite insertar comentarios en el código, estos comentarios no se visualizarán en el navegador. Podemos utilizar los comentarios para explicar nuestro código, o para delimitar/marcar una franja de nuestro código.

Un comentario en HTML comienza con el símbolo `<!--` y termina con el símbolo `-->`. Lo que hay en medio de los dos símbolos será el comentario que tendremos visible sólo en el

editor de texto que usemos para crear/modificar la web, pero no se mostrará en nuestro navegador. Por ejemplo:

```
<!DOCTYPE html>
<!-- El idioma de la web es opcional, en este caso indicamos que está en español -->
<html lang="es">
<head>
  <!-- Primero ponemos el juego de caracteres utilizado -->
  <meta charset="UTF-8">
  <!-- Insertamos el resto de metadatos -->
  <meta name="description" content="Mi primera página web en html5">
  <meta name="keywords" content="web html5">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <!-- Ponemos el título de la web -->
  <title>Mi primera página web</title>
</head>
<body>
  <!-- En el cuerpo de nuestra web ponemos lo que nos mostrará el navegador -->
  <p>Mi perra se llama <strong>Lola</strong></p>
</body>
</html>
```

5. Etiquetas de secciones

En versiones anteriores a HTML5, era habitual crear una página web utilizando las etiquetas `<div>` para agrupar secciones, hoy en día, se sigue haciendo. Con HTML5, se introducen una serie de etiquetas semánticas de agrupación que funcionan como los `divs` empleados en versiones previas pero añadiendo un significado dada la información que contienen. Estas etiquetas son:

- **encabezado visual de la página (Logo, título):** `<header>`
- **menú de navegación:** `<nav>`
- **contenido principal:** `<main>`, con varias subsecciones (además de la barra lateral) representadas por los elementos `<article>`, `<section>`, y `<div>`
- **barra lateral:** `<aside>`; a menudo colocada dentro de `<main>`.
- **pie de página/sección:** `<footer>`
- **Búsqueda de contenido:** `<search>`
- **Información de contacto/autor:** `<address>`
- **Los encabezados de título:** del `h1` al `h6`, de mayor a menor importancia `<h1>`.

En ocasiones, cuando queremos ocultar un elemento empleamos el `display: none` de CSS, sin embargo, lo correcto sería darle sentido semánticamente, de tal forma que podemos utilizar el atributo `hidden` para indicar que esa información que tiene ese contenedor no la queremos mostrar.

5.1. Elementos de diseño HTML

El típico diseño utilizando elementos HTML5 de una página web es el siguiente:

<header></header>	
<nav></nav>	
<main>	<aside>
<article></article>	
<article></article>	
<article></article>	
<article></article>	
</main>	
<footer></footer>	

Esta tabla muestra el típico ejemplo de una página con los correspondientes elementos HTML5 incluyendo sus etiquetas de apertura y cierre. Si dentro del contenido principal tuviéramos artículos agrupados por diferente temática, cada grupo estaría dentro de un elemento llamado `<section>`

5.2. Encabezado <header>

Esta sección representa un grupo de contenido introductorio. Si este es “hijo” directo del `<body>` se convertirá en el encabezado principal de nuestra web, pero si es hijo de un elemento `<article>` o de un elemento `<section>`, entonces simplemente será el encabezado particular de cada sección. Es importante que no lo confundamos con títulos o encabezados y mucho menos con el `<head>`. Ejemplo:

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Página con secciones html5</title>
</head>
<body>
  <header>
    
    <h1>Titulo principal de nuestra web</h1>
  </header>
  ...
</body>
</html>
```

5.3. Menú <nav>

El elemento `<nav>` contiene la funcionalidad de navegación principal de la página. Los enlaces secundarios, etc. no entrarán en la navegación.

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Página con secciones html5</title>
```

```

</head>
<body>
  <header>
    
    <h1>Titulo principal de nuestra web</h1>
  </header>
  <nav>
    <a href="#">Home</a>
    <a href="ventas.html">Ventas</a>
    <a href="sobrenosotros.html">About us</a>
  </nav>
  ...
</body>
</html>

```

5.4. contenido principal <main>

Este elemento encierra el contenido particular a esta página. Utilizaremos una sola vez la etiqueta `<main>` para cada página, y lo pondremos directamente dentro del `<body>`. Es mejor que no lo anidemos en otros elementos.

Este elemento es muy importante, ya que en el caso de utilizar navegadores de voz, el navegador puede ir directamente al contenido principal, saltando toda la información previa.

```

<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Página con secciones html5</title>
</head>
<body>
  <header>
    
    <h1>Titulo principal de nuestra web</h1>
  </header>
  <nav>
    <a href="#">Home</a>
    <a href="ventas.html">Ventas</a>
    <a href="sobrenosotros.html">About us</a>
  </nav>
  <main>
    ...
  </main>
  ...
</body>
</html>

```

5.5. Elemento <article>

Este elemento encuadra un bloque de contenido que tiene sentido por sí mismo aparte del resto de la página (por ejemplo la entrada en un blog). Cada una de las entradas del blog estarían dentro de la etiqueta `<article>`

5.6. Elemento <section>

Es parecido al elemento `<article>` pero nos permite agrupar secciones genéricas, es decir, si vamos a crear una agrupación temática pero no es tan específica como para ponerla con los `<article>`, pasamos a usar `<section>`. Se considera una buena práctica comenzar cada una de estas secciones con un título de encabezado (`<h1>`, `<h2>`, ...), pudiendo dividir los artículos (`<article>`) en distintas secciones (`<section>`) o también secciones en distintos artículos dependiendo del contexto.

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Página con secciones html5</title>
</head>
<body>
  <header>
    
    <h1>Titulo principal de nuestra web</h1>
  </header>
  <nav>
    <a href="#">Home</a>
    <a href="ventas.html">Ventas</a>
    <a href="sobrenosotros.html">About us</a>
  </nav>
  <main>
    <section>
      <h2>Venta de bicis</h2>
      <article>
        <!-- Ventas de bicis -->
      </article>
    </section>
    <section>
      <h2>Venta de componentes</h2>
      <article>
        <!-- Ventas de componentes -->
      </article>
    </section>
  </main>
  ...
</body>
</html>
```

5.7. Barra lateral <aside>

El elemento `<aside>` incluye contenido que no está directamente relacionado con el contenido principal, pero que puede aportar información adicional relacionada indirectamente con él (resúmenes, biografías del autor, enlaces relacionados, etc.)

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Página con secciones html5</title>
```

```

</head>
<body>
  <header>
    
    <h1>Titulo principal de nuestra web</h1>
  </header>
  <nav>
    <a href="#">Home</a>
    <a href="ventas.html">Ventas</a>
    <a href="sobrenosotros.html">About us</a>
  </nav>
  <main>
    <section>
      <h2>Venta de bicis</h2>
      <article>
        <!-- Ventas de bicis -->
      </article>
    </section>
    <section>
      <h2>Venta de componentes</h2>
      <article>
        <!-- Ventas de componentes -->
      </article>
    </section>
  </main>
  <aside>
    <!-- Anuncios a patrocinadores -->
  </aside>
  ...
</body>
</html>

```

5.8. Pie de página <footer>

Este elemento representa un grupo de contenido al final de una página, como el autor, dirección de contacto, licencia, condiciones de uso, etc. Este elemento puede estar incluido también en `<article>`, `<section>`, `<nav>` y `<aside>` entonces se entiende que es el pie de página del elemento que lo contiene.

```

<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Página con secciones html5</title>
</head>
<body>
  <header>
    
    <h1>Titulo principal de nuestra web</h1>
  </header>
  <nav>
    <a href="#">Home</a>
    <a href="ventas.html">Ventas</a>
    <a href="sobrenosotros.html">About us</a>

```

```

</nav>
<main>
  <section>
    <h2>Venta de bicis</h2>
    <article>
      <!-- Ventas de bicis -->
    </article>
  </section>
  <section>
    <h2>Venta de componentes</h2>
    <article>
      <!-- Ventas de componentes -->
    </article>
  </section>
</main>
<aside>

  <!-- Anuncios a patrocinadores -->

</aside>
<footer>

  <p>Creado por: <a href="mailto:rmedina@iessanvicente.com">Rosa Medina</a></p>

</footer>
</body>
</html>

```

5.9. Búsqueda de contenido <search>

Esta etiqueta representa una agrupación de elementos destinados a realizar búsquedas o filtros. Por ejemplo, un formulario de búsqueda global de internet.

```

<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Página con secciones html5</title>
</head>
<body>
  <main>
    <search>
      <form method="get" action="https://www.google.com/search">
        <label>Dime</label><input type="search" name="q" autocomplete="off">
        <input type="submit" value="Buscar">
      </form>
    </search>
  </main>
</body>
</html>

```

5.10. Información contacto/autor <address>

Este elemento encierra la información de contacto de una persona, como puede ser el creador de un artículo o comentario en nuestro blog

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Página con secciones html5</title>
</head>
<body>
  <main>
    ...
    <footer>
      <address>
        Contacta con Rosa escribiendo a: <span>rmedina@iessanvicente.com</span>
      </address>
    </footer>
  </main>
</body>
</html>
```

5.11. Los encabezados de título <h1>

La mayor parte del texto estructurado está compuesto por encabezados y párrafos, independientemente de si lees una historia, un periódico, un libro de texto, una revista, etc. El encabezado nos va a permitir destacar un titular del resto del texto.

Cada sección tiene que estar delimitada por un elemento encabezado:

```
<h1>La historia de mi perra Lola</h1>
```

Hay seis elementos de encabezado: <h1>, <h2>, <h3>, <h4>, <h5> y <h6>. Cada elemento representa un nivel de contenido diferente en el documento, <h1> representa el título principal, <h2> el subtítulo, <h3> representa el subtítulo del subtítulo y así sucesivamente. Es decir, <h1> es el encabezado de mayor importancia y <h6> es el de menor.

Por ejemplo, en esta historia <h1> representa el título de la historia principal, <h2> representa el título de cada capítulo, y <h3> representa las diferentes secciones del capítulo y así sucesivamente.

```
<h1>La historia de mi perra Lola</h1>
<h2>Capítulo 1: Buscamos un perro</h2>
<p>Era una noche oscura, se había ido la luz tras la
fuerte tormenta...</p>
<h2>Capítulo 2: Contactamos con los dueños</h2>
<p>Paqui tenía puesto en FB una manada de perros.</p>
<h3>La hija de Paqui</h3>
<p>La hija de Paqui me facilitó el teléfono....</p>
```

La historia de mi perra Lola

Por Rosa Medina

Capítulo 1: Buscamos un perro

Era una noche oscura, se había ido la luz tras la fuerte tormenta...

Capítulo 2: Contactamos con los dueños

Paqui tenía puesto en FB una manada de perros...

La hija de Paqui

La hija de Paqui me facilitó el teléfono....

Como podemos observar el `<h1>` es el más grande y por tanto más importante y a medida que hemos incrementado, los títulos han sido más pequeños.

6. Etiquetas de texto

En el siguiente apartado, vamos a ver las etiquetas que contienen fragmentos de textos. Uno de los atributos que surgió con HTML5 en las etiquetas de texto fue el atributo `contenteditable` sobre una etiqueta HTML, de tal forma que nos da como usuarios la posibilidad de editar su contenido de texto, pudiendo modificar el texto que aparece. Ejemplo:

```
<h3 contenteditable>Título del artículo editable</h3>
<div contenteditable>
  <p>Párrafo editable al ser todo su div editable.</p>
  <p>Y este que forma parte del contenedor también.</p>
</div>
```

6.1. Párrafos

En HTML, cada párrafo tiene que estar delimitado por un elemento `<p>`, por ejemplo:

```
<p>Había una vez, una perrica llamada Lola</p>
```

Dentro de un párrafo podemos generar un salto de línea con la etiqueta `
`. Sólo se usará para hacer un salto de línea simple, nunca para separar párrafos. Esta etiqueta es útil para escribir un poema o una dirección, donde la división de las líneas es significativa.

```
<p>Verso 1, <br> Verso 2, <br> Verso 3.</p>
```

No debemos usar `
` para incrementar el espacio entre líneas de texto, para ello usaremos la propiedad de CSS `margin` por ejemplo o el elemento `<p>`.

6.2. Resaltado de texto: Negrita `` y cursiva ``

En HTML existen dos tipos principales de resaltado o enfatizado de texto: **negrita** y *cursiva*.

Hace unos años, se utilizaba la etiqueta `` para poner el texto en negrita, sin embargo esta etiqueta no aporta ningún significado especial por lo que su uso está desaconsejado. En su lugar, se recomienda utilizar la etiqueta ``, que también muestra el texto en negrita pero que además nos indica que ese texto es muy importante. Al haber elegido la etiqueta `` estamos haciendo que los navegadores de voz lean ese texto dándole el énfasis correspondiente.

```
<!DOCTYPE html>
<html lang="es">
<head>...</head>
<body>
  <p>Cuento contigo. <strong>¡No llegues tarde!</strong></p>
</body>
</html>
```


Del mismo modo, elemento `<i>` pone el texto en *cursiva*, pero no aporta ningún significado especial esta etiqueta y se desaconseja. En su lugar, se utiliza la etiqueta `` que indica a los lectores de pantalla que el texto debe tener un mayor énfasis. Por ejemplo:

```
<!DOCTYPE html>
<html lang="es">
<head>...</head>
<body>
  <p>Me alegro de que no llegues tarde</p>
  <p>Me <em>alegro</em> de que no llegues <em>tarde</em></p>
</body>
</html>
```

La primera frase suena aliviada porque la persona no llega tarde. Por lo contrario, la segunda suena sarcástica y un tanto pasivo-agresiva, expresa molestia porque la persona ha llegado algo tarde.

El navegador, de manera predeterminada, aplica el estilo de letra itálica, pero no debes utilizar esta etiqueta solamente para establecer el estilo de letra itálica. Para usar ese estilo, debes utilizar únicamente la etiqueta del elemento `` y algo de CSS.

6.3. La etiqueta `<s>`

Si queremos mostrar un texto que ha sido modificado y queremos mostrar el texto anterior usamos la etiqueta `<s>` para mostrar el texto tachado una línea horizontal.

```
<p> <del>Don Quijote</del> dijo: <s>"En un lugar de la mancha..."</s> </p>
```

Debemos tener en cuenta que esta etiqueta no sustituye a la etiqueta `` (para marcar un texto como eliminado)

6.4. La etiqueta ``

El elemento `` es un elemento que no tiene semántica, se utiliza para delimitar contenido cuando se le quiere aplicar CSS (o tratarlo con JS) sin proporcionarle ningún significado extra. Por ejemplo:

```
<!DOCTYPE html>
<html lang="es">
<head>...</head>
<body>
  <p>El boli <span style="color: red;">rojo</span> no funciona</p>
</body>
</html>
```

Nota: En la siguiente unidad se estudiará CSS y por tanto será ahí cuando estudiemos qué es el atributo `style` que le hemos puesto a la etiqueta ``.

6.5. Citas: etiquetas <cite>, <blockquote> y <q>

La etiqueta `<cite>` sirve para marcar una referencia a una fuente o el autor de un texto citado.

```
<p> <cite>Don Quijote</cite> dijo: "En un lugar de la mancha..." </p>
```

En navegadores basados en Mozilla (y en otros) el contenido de `cite` es mostrado con un estilo de texto en cursiva. Es importante saber que este elemento no se debe utilizar para marcar el texto de las citas, es para marcar el autor.

El elemento HTML `<q>` indica que el texto adjunto es una cita corta en línea. La mayoría de los navegadores muestran el texto entre comillas. Este elemento está destinado a citas breves que no requieren saltos de párrafo, para citas de bloque independiente, se debe utilizar el elemento `<blockquote>`.

```
<p>Conforme al sitio web de Mozilla,  
  <q cite="https://www.mozilla.org/en-US/about/history/details/">Firefox 1.0  
fue lanzado en 2004 y se convirtió en un gran éxito.</q></p>
```

El elemento `<blockquote>` se utiliza para crear citas en bloque, marca las citas a otros autores o documentos.

```
<blockquote cite='http://html.conclase.net/w3c/html401...def-BLOCKQUOTE'>  
  <p>  
    <strong>Nota.</strong> Recomendamos que las implementaciones de hojas de estilo porporcionen un  
    mecanismo para insertar signos de puntuación de citas antes y después de una cita delimitada por un  
    BLOCKQUOTE de un modo apropiado según el contexto  
    del idioma actual y el grado de anidamiento de las citas.  
  </p>  
</blockquote>
```

Tanto para el elemento `<q>` como para `<blockquote>`, se utiliza el atributo `cite` para proporcionar un enlace al documento original o fuente.

6.6. La etiqueta <u>

Esta etiqueta nos permitía en versiones anteriores a HTML5 subrayar el texto. A día de hoy esta etiqueta se utiliza para destacar textos mal escritos o en un contexto que no es el actual. A día de hoy esta etiqueta ha sido desplazada por la etiqueta `<mark>`

6.7. La etiqueta <sup> y <sub>

El elemento `<sub>` se utiliza para mostrar un texto más pequeño que el principal, es decir, un subíndice. Del mismo modo, tenemos el elemento `<sup>` que se utiliza para lo opuesto, es decir, para mostrar un texto más alto, un superíndice.

6.8. La etiqueta <abbr> y <dfn>

El elemento `<abbr>` se utiliza para representar una abreviación o acrónimo. Si usamos el atributo `title` podemos indicarle el significado del texto expandido, de forma que demos más información.

```
<p>El término <abbr title="Centímetros"> cm</abbr> es una unidad de longitud</p>
```

Del mismo modo, la etiqueta `<dfn>` se utiliza para indicar que a continuación se va a dar una definición.

```
<p>El <dfn>HTML</dfn> es un lenguaje de marcado para hipertextos.</p>
```

6.9. Texto preformateado: etiquetas <pre> y <code>

La etiqueta `<pre>` se utiliza para mostrar texto respetando el formato con el que está escrito, es decir, respetando los espacios en blanco, saltos de línea, etc. Esta etiqueta es muy habitual utilizarla junto con la etiqueta `<code>`.

```
<!-- Un poco de código CSS -->
<pre> body{
  color:  red;}
a  {
  color:green;
}
</pre>
```

La etiqueta `<code>` es apropiada para marcar el código de un programa. Por ejemplo:

```
<code>
public class HolaMundo {
    public static void main (String[] args) {
        System.out.println("Hola mundo");
    }
}
</code>
```

```
public class HolaMundo { public static void main (String[] args) { System.out.println("Hola mundo"); } }
```

Si el ejemplo anterior lo hacemos utilizando las dos etiquetas (`<pre>` y `<code>`) el resultado a mostrar en el navegador sería muy diferente:

```
<pre>
  <code>
    public class HolaMundo {
      public static void main (String[] args) {
        System.out.println("Hola mundo");
      }
    }
  </code>
</pre>
```

```
public class HolaMundo {
    public static void main (String[] args) {
        System.out.println("Hola mundo");
    }
}
```

6.10. Marcado de texto: etiqueta <mark>

El elemento `<mark>` representa un texto marcado o resaltado como referencia o anotación, debido a su relevancia o importancia en un contexto particular. Es un fragmento de texto que parece estar subrayado con un rotulador amarillo. Ejemplo:

```
<blockquote>It is a period of civil war. Rebel spaceships, striking from a hidden base, have won their first victory against the evil Galactic Empire. During the battle, <mark>Rebel spies managed to steal secret plans</mark> to the Empire's ultimate weapon, the DEATH STAR, an armored space station with enough power to destroy an entire planet.
</blockquote>
```

It is a period of civil war. Rebel spaceships, striking from a hidden base, have won their first victory against the evil Galactic Empire. During the battle, Rebel spies managed to steal secret plans to the Empire's ultimate weapon, the DEATH STAR, an armored space station with enough power to destroy an entire planet.

6.11. Borrados e inserciones: etiquetas e <ins>

El elemento `` marca las partes de un texto o documento que han sido suprimidas o sustituidas. Nos muestra esa parte como tachada con una línea. Por otro lado, el elemento `<ins>` marca las partes de un texto que han sido añadidas a un documento. Veamos un ejemplo:

```
<p>El agua es insípida <del>y húmeda.</del> <ins>inodora e incolora.</ins></p>
```

El agua es insípida y húmeda. inodora e incolora.

6.12. La etiqueta <time> y <data>

Las fechas y horarios son partes fundamentales de los contenidos de las páginas web. Los motores de búsqueda son capaces de filtrar los resultados basándose en el tiempo y, en algunos casos, un resultado de búsqueda específico puede recibir más o menos peso en función de cuándo fue publicado por primera vez. El elemento `<time>` representa un período específico en el tiempo. Puede incluir el atributo `datetime` para convertir las fechas en un formato interno legible por un ordenador.

```
<p>El concierto fue el <time datetime="2001-05-15T19:00">15 de Mayo</time>.</p>
```

6.13. La etiqueta <small>

El elemento `<small>` hace el tamaño del texto una “talla” más pequeña que el tamaño mínimo de fuente del navegador. En HTML5, este elemento es reutilizado para representar comentarios laterales y letra pequeña incluyendo derechos de autor y texto legal, independientemente de su estilo de presentación.

```
<p>MDN Web Docs is a learning platform for Web technologies and the software that powers the Web.</p>
<hr>
<p><small>The content is licensed under a Creative Commons Attribution-ShareAlike 2.5 Generic
License.</small></p>
```

MDN Web Docs is a learning platform for Web technologies and the software that powers the Web.

The content is licensed under a Creative Commons Attribution-ShareAlike 2.5 Generic License.

6.14. La etiqueta
, <wbr> y <hr>

El elemento `
` produce un salto del línea en el texto (retorno de carro) Es útil para escribir un poema o dirección donde la división de las líneas es significativa.

```
<p>IES San Vicente<br> Calle Lillo Juan<br> San Vicente del Raspeig</p>
```

IES San Vicente
Calle Lillo Juan
San Vicente del Raspeig

La etiqueta `<wbr>` funciona de forma similar a la anterior etiqueta, pero a diferencia de esta, se utiliza para evitar que se desborde el contenido, cortando la palabra en el caso de poder hacerlo, de lo contrario permanecerá junta. De tal forma que si el contenedor que tiene la el siguiente ejemplo es muy pequeño la palabra *supernova* sea cortada.

```
<p>super<wbr>nova</p>
```

Por otro lado, la etiqueta `<hr>` representa un cambio de tema entre párrafos. En versiones previas de HTML se representaba una línea horizontal, todavía puede representarse como una línea en los navegadores visuales, pero ahora es definida en términos semánticos y no tanto en términos representativos, por tanto para representar una línea en horizontal deberíamos utilizar CSS.

```
<p>Este es un párrafo únicamente visual para ver el uso de la etiqueta hr</p>
<hr>
<p>Este es el segundo párrafo, separado del primero por la etiqueta hr, que puede representarse mediante una línea
```

```
horizontal.</p>
```

Este es un párrafo únicamente visual para ver el uso de la etiqueta `hr`

Este es el segundo párrafo, separado del primero por la etiqueta `hr`, que puede representarse mediante una línea horizontal.

6.15. La etiqueta `<kdb>` y `<samp>`

La etiqueta `<kdb>` se utiliza para representar la entrada de datos del usuario, por ejemplo cuando decimos que debe utilizar una combinación de teclas para realizar cierta acción.

La etiqueta `<samp>` corresponde a la opuesta, es decir, representa la salida de datos del ordenador hacia el usuario. Podemos interpretarla como la salida de un programa.

7. Enlaces `<a>`

Una parte fundamental de la web es que los documentos estén enlazados entre sí, es decir, que podamos ir de una página a otra con un simple clic. La etiqueta utilizada para crear enlaces usaremos la etiqueta `<a>`. El texto o imagen introducido dentro de esta etiqueta, reaccionará ante un clic de ratón llevándonos al destino especificado. Esta etiqueta utiliza el atributo `href` para indicar la url de destino del enlace.

```
<a href="https://iesanvicente.com/">Ir a la web del IES</a>
```

Otro de los atributos que admite la etiqueta `<a>` es `title`, este atributo nos permite especificar información extra sobre el elemento. Esta información se muestra cuando ponemos el ratón sobre este elemento. En el ejemplo anterior, podría ser algo así: `title="URL del instituto"`.

Si queremos crear un enlace que en lugar de ir a url determinada vaya a un punto determinado de nuestro documento actual, haremos:

1. Marcar el elemento a dónde queremos ir, para ello usaremos el atributo `id` y le asignamos un valor sin espacios.
2. En el enlace ponemos el símbolo `#` seguido del `id` al que queremos que vaya.

```
<h1 id="sobreNosotros">Sobre nosotros</h1>
<p>Somos una empresa...</p>
<p>...</p>
<a href="#sobreNosotros">Ir a Sobre nosotros</a>
```

Nota: El `id` debe ser un valor único en la página, es un identificador. Si queremos que vaya a una sección de otra página sería: `href="pagina2.html#sobreNosotros"`

Tenemos la posibilidad de abrir un enlace en una ventana o pestaña nueva, para ello utilizaremos el atributo `target` dándole el valor de `_blank`. Más opciones [aquí](#).

Para enlazar una dirección de correo electrónico, se indica con `mailto`: seguido de la dirección de correo.

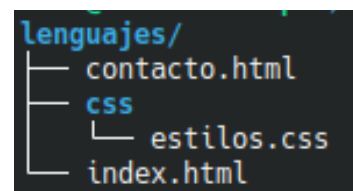
```
<p><a href="mailto:rmedina@iessanvicente.com" title="Correo profe">Envía un correo</a></p>
```

7.1. Ruta absoluta y ruta relativa

Si estamos desarrollando nuestra aplicación en un directorio llamado lenguajes que tiene el siguiente contenido:

Dentro del atributo `href` debemos indicar la ruta a la web que queremos acceder al hacer clic. Esta ruta la podemos indicar de dos modos:

1. Como una ruta absoluta a la raíz de la aplicación. Como `contacto.html` está en la carpeta raíz de la aplicación `lenguajes`, el enlace quedaría así:



```
<a href="/contacto.html" title="Página de contacto">Contacta con nosotros</a>
```

2. Como una ruta relativa desde la carpeta en donde nos situamos. Por ejemplo si nos encontramos en la misma carpeta el enlace sería:

```
<a href="contacto.html" title="Página de contacto">Contacta con nosotros</a>
```

- a) En el caso de estar dentro de una subcarpeta y debemos subir de nivel (acceder a la carpeta padre) debemos utilizar `..` pudiendo enlazarlos para subir de nivel en varias carpetas, por ejemplo `../../` para salir de tres carpetas.

```
<a href="../../contacto.html" title="Página de contacto">Contacta con nosotros</a>
```

8. Etiquetas multimedia

8.1. Imágenes

Para insertar imágenes en nuestra web utilizamos la etiqueta `` con su atributo `src` para indicar en dónde se encuentra la imagen. La etiqueta `` es un elemento vacío, por lo que no es necesario que introduzcamos la etiqueta de cierre. Podemos utilizar 3 formatos de imágenes para nuestra página: JPEG, GIF y PNG.

- Con **JPEG** conseguimos el menor tamaño, aunque no la mejor calidad. Está recomendado para imágenes medianas o grandes con muchos colores y variedad. No soporta transparencias.
- El formato **GIF** soporta transparencias, pero está muy limitado en colores (256). También nos permite animaciones sencillas.
- El formato **PNG** es el que mejor calidad nos da, además de soportar transparencias. Todo esto a un coste un poco mayor en cuanto tamaño.
- Formato **SVG** es el formato vectorial, este formato es el ideal para imágenes escalables

Se debe poner el atributo `alt` a todas las imágenes, establece un texto alternativo que describa la imagen a mostrar.

```

```

Además de estos dos atributos, la etiqueta `` admite los siguientes atributos:

- **width:** ancho de la imagen (en píxeles o porcentaje). Si no indicamos altura, mantiene proporciones de la imagen. Normalmente se establece por CSS
- **height:** alto de la imagen.

8.1.1. Imágenes y enlaces

Para que una imagen nos sirva como enlace, simplemente hay que meterla entre la etiqueta `<a>` como si fuera texto (de hecho, en HTML una imagen es tratada a nivel de texto en muchos aspectos, no es un bloque independiente).

```
<a href="https://iessanvicente.com/">
  
</a>
```

Si cargamos una imagen grande (Ej: 200KB) y hacemos que se vea mucho más pequeña, no conseguimos nada a nivel de ahorro en ancho de banda. El navegador cargará la imagen entera (los 200KB) y luego la escalará.

Para los llamados thumbnails, o previsualizaciones pequeñas, lo mejor es crearse una imagen pequeña con algún programa de edición, a partir de la original que ocupe mucho menos espacio.

8.1.2. Favicon (icono de pestaña)

Cuando tenemos varias pestañas en nuestro navegador, para una búsqueda más fácil e intuitiva entre las pestañas que tenemos abiertas, el navegador suele mostrarnos unos iconos a la izquierda del título de cada pestaña, estos iconos se conocen como favicons (iconos de favoritos).

La etiqueta que se utiliza para poner este icono es la etiqueta `<link>` la cual va dentro de la etiqueta `<head>` de nuestro HTML (**no en el body**). El problema que presenta favicon es que cada navegador (y casi cada sistema operativo) la implementa como quiere. Hay navegadores que sólo soportan favicon en formato png, otros que solo soportan resoluciones específicas, etc. Abajo aparece el uso de favicon en 3 plataformas diferentes.

```
<link rel="icon" sizes="64x64" href="iconos/favicon.png" /> <!-- HTML5 -->
<link rel="apple-touch-icon" sizes="180x180" href="iconos/apple-touch-icon-180x180.png" /> <!-- iPhone/iPad -->
<meta name="msapplication-TileImage" content="iconos/mstile-144x144.png" /> <!-- Windows Phone -->
```

Si queremos crearnos un favicon de forma cómoda podemos utilizar la herramienta [Real Favicon Generator](#), donde le pasamos un logo a alta resolución y nos genera un .zip con el código y diferentes versiones y resoluciones de los iconos.

8.1.3. Elemento `<figure>`

En ocasiones puede que nos interese añadir un pie de foto a una imagen. Para poderlo tratar como un único elemento (imagen + pie) podemos utilizar la etiqueta `<figure>`, así como la etiqueta `<figcaption>` para definir dicho pie de foto.

```
<figure>
```



```

<figcaption>Imagen reciente de la Sagrada Familia</figcaption>
</figure>
```

8.1.4. Nuevas etiquetas de imágenes

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

Etiqueta	Atributos	Descripción
<code><picture></code>		Agrupar una serie de imágenes. Es una etiqueta contenedora
<code><source></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>
```

8.2. 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, ni 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?
MP3	MP3 Layer-3	Buena calidad	Sí
AAC	Advanced Audio Coding	Mejora el MP3, usado como audio en MP4	Sí
OGG	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
<audio>	src, type	Establece un archivo de audio o lo añade como alternativa
<track>	src, srclang, label, kind, default	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

8.3. 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
<code>loop</code>	boolean	Se reproduce el vídeo en modo bucle
<code>muted</code>	boolean	Silenciamos el vídeo
<code>controls</code>	boolean	Muestra los controles de reproducción. Por defecto no se muestran
<code>width</code>	tamaño	Indica el tamaño de ancho del vídeo
<code>height</code>	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>
```

8.4. Etiqueta <track>

Con HTML5 se añade el **estándar webVTT, destinado a la creación de subtítulos en archivos de vídeo o audio**. Este formato se puede utilizar tanto dentro de la etiqueta `<video>` como de `<audio>`. Los atributos que puede contener son: `src`, `srclang` (codigo del idioma), `label` (título que verá al pulsar el botón CC), `default` (booleano que marca como los subtítulos principales por defecto).

```
<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">
  
  <track src="sub-es.vtt" srclang="es" label="Español" default>
  <track src="sub-en.vtt" srclang="en" label="English" default>
  <track src="sub-fr.vtt" srclang="fr" label="Français" default>
</video>
```

Como web de refuerzo tenéis <https://htmldesdezero.es/etiquetas/track/>

8.5. 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><iframe></code>	<code>src</code> , <code>srcdoc</code> , <code>name</code> , <code>width</code> , <code>height</code>	Permite incrustar contenido externo en “vivo”
<code><embed></code>	<code>src</code> , <code>type</code> , <code>width</code> , <code>height</code>	Permite incrustar contenido interactivo
<code><object></code>	<code>data</code> , <code>type</code> , <code>name</code> , <code>form</code> , <code>width</code> , <code>height</code>	Permite incrustar contenido externo con fallbacks (contenidos alternativos)
<code><param></code>	<code>name</code> , <code>value</code>	Define parámetros de un elemento <code><object></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_A4fgqqzo" 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.

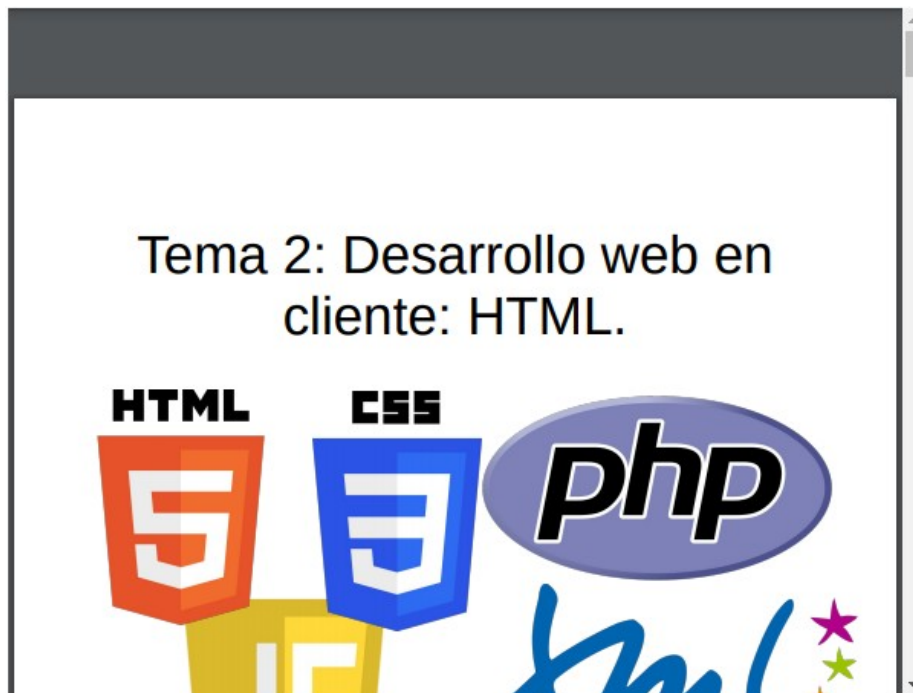
```
<iframe src="https://www.google.com/maps/embed?pb=!1m18!1m12!1m3!1d12506.093962966048!2d-0.5318828639471987!3d38.40626542438854!2m3!1fo!2fo!3fo!3m2!1i1024!2i768!4f13.1!3m3!1m2!1soxd6233fea3991575%3Aoxf28fcef8c48c1513!2sIES%20San%20Vicente!5eo!3m2!1ses!2ses!4v1635452091546!5m2!1ses!2ses" width="600" 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>
```



9. Tablas, <table>

Las tablas se definen en HTML con la etiqueta `<table>`. Una tabla está dividida en filas con la etiqueta `<tr>` y una fila se compone de celdas de datos, mediante la etiqueta `<td>`, aunque también se puede dividir en celdas de cabecera con la etiqueta `<th>`.

Los elementos `<td>` son los contenedores de los datos en la tabla y pueden contener todo tipo de elementos HTML como texto, imágenes, listas, otras tablas, etc.

```

<table>
  <tr>
    <th>Nombre</th>
    <th>Apellidos</th>
    <th>Correo</th>
  </tr>
  <tr>
    <td>Rosa</td>
    <td>Medina</td>
    <td>rmedina@iessanvicente.com</td>
  </tr>
  <tr>
    <td>Carlos</td>
    <td>Bernal</td>
    <td>carlos@iessanvicente.com</td>
  </tr>
</table>
```

Nombre	Apellidos	Correo
Rosa	Medina	rmedina@iessanvicente.com
Carlos	Bernal	carlos@iessanvicente.com

Nota importante: las tablas están en desuso y no se deben de usar. Solo se imparten por si algún día os toca mantener código ya creado

Nota: Con solo el código anterior sólo conseguimos la estructura de la tabla, si queremos que nuestra tabla tenga el mismo aspecto que en la imagen tenemos que aplicar estilos CSS.

Los algunos de los atributos que pueden aparecer en la etiqueta `<table>` son:

- **summary:** es parecido al atributo `alt` de las imágenes. Proporciona na descripción de la tabla para navegadores en modo texto, o invidentes.
- **border:** el valor es numérico y establece el grosor del borde de la tabla (0 para quitarlo)
- **cellpadding:** espacio que debe quedar entre las propias celdas (entre los bordes)
- **width:** ancho de la tabla (suele usarse con CSS)
- Otros atributos como `align`, para alinear la tabla o `bgcolor` para establecer el color de fondo de la tabla están desaconsejados, mejor establecerlos con CSS.

Por otro lado, los atributos que admiten los elementos `<td>` (aunque es mejor aplicarlos con CSS) son:

- **align:** alinea el texto de la celda en horizontal. Puede tener los valores `left`, `right` o `center`.
- **valign:** alinea el texto de la celda en vertical. Puede tener los valores `top`, `bottom` o `middle`.
- **rowspan:** expande la celda x filas hacia abajo.
- **colspan:** expande la celda x filas a la derecha

- **width:** indica la anchura específica de esa columna, su valor viene dado en px ó %
- **height:** altura de la celda, en px ó %
- **bgcolor:** color de fondo de la celda (o de la tabla si se aplica en `<table>`)

```

<table border="1">
  <tr>
    <td rowspan="2" valign="middle" align="right" width="50%">Lenguajes de marcas</td>

    <td>Entornos</td>
    <td>Programación</td>
    <td>FOL</td>
  </tr>
  <tr>
    <td>lunes</td>
    <td colspan="2" align="right">martes</td>
  </tr>
  <tr>
    <td colspan="4" align="center">Primera evaluación</td>
  </tr>
</table>

```

Lenguajes de marcas	Entornos	Programación	FOL
	lunes	martes	
Primera evaluación			

Como podemos observar con el atributo `colspan` podemos hacer que una celda ocupe el espacio equivalente a 2 o más celdas. El valor será un número con las columnas que queramos que ocupe. Ejemplo:

```

<table summary="Precio de venta" border="1">
  <tr>
    <td colspan="2">Ventas mayo 2021</td>
  </tr>
  <tr>
    <td>Fruta</td>
    <td>€/Kg</td>
  </tr>
</table>

```

Ventas mayo 2021	
Fruta	€/Kg

También podemos hacer que una celda ocupe varias filas (en ese caso también tendremos que poner una celda menos en la fila ocupada) para ello usaremos el atributo `rowspan`

```

<table summary="Precio de venta" border="1">
  <tr>
    <td colspan="3">Ventas del 2021</td>
  </tr>
  <tr>
    <td rowspan="2">Mayo</td>
    <td>Manzanas</td>
    <td>280€</td>
  </tr>
  <tr>
    <td>Peras</td>
    <td>385€</td>
  </tr>
</table>

```

Ventas del 2021		
Mayo	Manzanas	280€
	Peras	385€

```

    <td>Peras</td>
    <td>385€</td>
  </tr>
</table>

```

Además de estas etiquetas vistas para tablas, tenemos:

<code><caption></code>	Permite asignar una breve descripción a la tabla. Es un atributo de <code><table></code>
<code><thead></code>	Define el encabezado de una tabla
<code><tbody></code>	Define el cuerpo de una tabla
<code><tfoot></code>	Define el pie de la tabla

```

<table border="1">
  <caption>Ahorros del 2021</caption>
  <thead>
    <tr>
      <th>Month</th>
      <th>Savings</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>January</td>
      <td>$100</td>
    </tr>
    <tr>
      <td>February</td>
      <td>$80</td>
    </tr>
  </tbody>
  <tfoot>
    <tr>
      <td align="right">Sum</td>
      <td>$180</td>
    </tr>
  </tfoot>
</table>

```

Ahorros del 2021

Month	Savings
January	\$100
February	\$80
Sum	\$180

10. Agrupación de etiquetas

En HTML disponemos de dos tipos de elementos: elementos en bloque y elementos en línea. Un elemento en bloque es aquel que incorpora un salto de línea antes y después del mismo, mientras que un elemento en línea se colocará a continuación del elemento anterior, sin introducir ningún salto de línea.

Ejemplo de elementos en línea: ``, `<a>`, ``, `<code>`, etc. Por otro lado, algunos de los elementos en bloque son: `<div>`, `<p>`, `<h1>`, ``, etc.

10.1.1. Elemento <div>

El elemento `<div>` se puede utilizar como un contenedor para agrupar otros elementos HTML. Este elemento no tienen ningún significado especial, salvo que, por tratarse de un elemento a nivel de bloque, el navegador mostrará un salto de línea antes y después de él, pero semánticamente no aporta nada al documento.

Cuando se utiliza con CSS, el elemento `<div>` se puede utilizar para establecer atributos de estilo para grandes bloques de contenido. Otro uso común del elemento `<div>` es para el diseño del documento. Sustituyendo al anterior método, que utilizaban tablas para definir el diseño de la página.

```
<div class="container">
  <h1>Título de mi web</h1>
  <p>Elementos de una página web</p>
  <p>...</p>
</div>
```

El elemento `` puede utilizarse como un contenedor para texto. Al igual que `<div>`, no tiene ningún significado especial. Cuando se utiliza junto con CSS, se puede utilizar para establecer los atributos de estilo a partes del texto.

10.2. Listas

Las listas están en todos los aspectos de nuestra vida, desde lista de la compra, lista de tareas, etc. Debemos distinguir entre tres tipos de listas:

10.2.1. Listas no ordenadas

Las listas no ordenadas se usan para marcar listas de artículos cuyo orden no es importante. Por ejemplo, la lista de una compra. Cada lista desordenada comienza con un elemento `` (unordered list) que delimita todos los elementos de la lista, y cada uno de los artículos de la lista está delimitado con un elemento `` (list item).

```
<ul>
  <li>leche</li>
  <li>huevos</li>
  <li>pan</li>
  <li>chocolate</li>
</ul>
```

Lista no ordenada

- leche
- huevos
- pan
- chocolate

10.2.2. Listas ordenadas

Las listas ordenadas son aquellas en las que el orden de los elementos sí importa. Por ejemplo, las instrucciones para seguir la ruta para llegar al instituto.

La estructura del marcado es la misma que para las listas ordenadas, excepto que debes delimitar los elementos de la lista con una etiqueta `` (ordered list), en lugar de ``.

```
<ol>
  <li>Conduce hasta el final de la calle</li>
  <li>Gira a la derecha</li>
```

Lista ordenada

1. Conduce hasta el final de la calle
2. Gira a la derecha
3. Sigue derecho por las dos primeras glorieta
4. Gira a la izquierda en la tercer glorieta
5. El instituto está a tu derecha, 300 metros más adelante

```
<li>Sigue derecho por las dos primeras glorieta</li>
<li>Gira a la izquierda en la tercer glorieta</li>
<li>El instituto está a tu derecha, 300 metros más adelante</li>
</ol>
```

Podemos personalizar la lista con los siguientes atributos:

- **type**: indica el tipo de viñeta o marcador que se utiliza en la lista: **1** (enteros), **A** (letras mayúsculas), **a** (letras minúsculas), **I** (números romanos en mayúsculas), **i** (números romanos en minúscula)
- **start**: se utiliza para determinar el valor inicial, debe ser un número entero válido.
- **reversed**: es un atributo booleano (verdadero o falso). Indica que la lista es descendente (... , c, b, a) mientras que si se omite indica que la lista es ascendente (a, b, c, ...)

Si no indicamos ninguno de estos tres atributos se crea una lista ordenada de números enteros empezando desde el 1.

```
<ol reversed start="6" type="A">
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ol>
```

Lista ordenada invertida

F. Coffee
E. Tea
D. Milk

Además de estos atributos que le podemos dar a la lista, podemos modificar el valor de un determinado ítem de la lista (el valor del número de la lista), para ello utilizamos el atributo **value**.

10.2.3. Listas anidadas

Las listas se pueden anidar, es decir, podemos incluir una lista dentro de otra y cada una tendrá su propia numeración o viñeta. Para anidarlas, crearemos otra lista (con `` o ``) dentro de una etiqueta ``, y cerraremos `` después de cerrar previamente la definición de la sublista. Ejemplo:

```
<ol>
  <li> Opción 1 </li>
  <li> Opción 2 <!-- La etiqueta de cierre no va aquí -->
    <ul>
      <li> Opción 2A </li>
      <li> Opción 2B </li>
      <li> Opción 2C </li>
    </ul>
  </li> <!-- Aquí está la etiqueta de cierre -->
  <li> Opción 3 </li>
  <li> Opción 4 </li>
</ol>
```

Listas anidadas

1. Opción 1
2. Opción 2
 ◦ Opción 2A
 ◦ Opción 2B
 ◦ Opción 2C
3. Opción 3
4. Opción 4

10.2.4. Listas de descripción: etiquetas <dl>, <dt> y <dd>

Son listas donde se asocian términos con descripciones o valores, por ejemplo, glosarios, listas de vocabulario, listas de metadatos o listas de preguntas y respuestas, no es adecuado para representar diálogos.

Para representar o definir estas listas utilizamos el elemento `<dl>`, donde cada término está envuelto en un elemento (**data list**) `<dt>` (término de descripción, **data description**) y cada descripción está envuelta en un elemento `<dd>` (definición de descripción, **data definition**). Se puede asociar un término con múltiples descripciones y viceversa. El navegador muestra el contenido del texto `<dd>` con una sangría.

```
<dl>
  <dt>Coffee</dt>
  <dd>Black hot drink</dd>
  <dt>Milk</dt>
  <dd>White cold drink</dd>
</dl>
```

Lista de descripción

```
Coffee
  Black hot drink
Milk
  White cold drink
```

11. 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><input type="text"></code> <code><textarea></code>
Texto	Email	<code><input type="email"></code>
Texto	Búsqueda en Google	<code><input type="search"></code>
Texto	Teléfono	<code><input type="tel"></code>
Texto	URL	<code><input type="url"></code>
Número	Edad	<code><input type="number"></code>
Fechas u horas	Fecha de nacimiento, hora de comienzo, ...	<code><input type="date datetime datetime-local"></code>
Verdadero / falso	Sí/No, Opción A/B, ON/OFF	<code><input type="checkbox"></code>
Opción única	Elige una opción	<code><input type="radio"></code> <code><select></code>
Varias opciones	Elige la(s) opciones correctas	<code><input type="checkbox"></code> <code><select></code>
Opción única abierta	Elige una opción o añade una si consideras que no está ahí	<code><datalist></code>
Color	Selecciona un color	<code><input type="color"></code>
Archivo	Selecciona el archivo a adjuntar	<code><input type="file"></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.

11.1.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
type	Tipo de campo	Indica el tipo de campo que se trata
name	Nombre del campo	Indica el nombre del campo
value	Valor por defecto	Indica el valor inicial que tendrá ese campo
form	Nombre del formulario	Asocia el campo al nombre del formulario que indica
placeholder	sugerencia	Indica una sugerencia al usuario antes de escribir
size	número	Tamaño visual (número de caracteres) del campo de datos
autocomplete	on off	Activa o desactiva el autocompletado de este campo
autofocus	-	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><input type="text"></code>	<input type="text" value="Texto corto"/>
Texto para búsquedas	<code><input type="search"></code>	<input type="text" value="Texto a buscar"/>
Número de teléfono	<code><input type="tel"></code>	<input type="text" value="666777888"/>
Dirección URL	<code><input type="url"></code>	<input type="text" value="https://www.google.es"/>
Email	<code><input type="email"></code>	<input type="text" value="rmedina@iessanvicente.com"/>
Contraseña	<code><input type="password"></code>	<input type="password" value="....."/>
Campo oculto (no se muestra al usuario)	<code><input type="hidden"></code>	

Texto alfanumérico libre (largo)	<code><textarea></textarea></code>	
-------------------------------------	--	---

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

11.2. 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.

11.2.1. 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>
```


11.2.2. 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.

11.3. 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>
```

11.4. 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><input type="number"></code>	
Rango numérico	<code><input type="range"></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>
```

11.5. 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><input type="date"></code>	<input type="text" value="dd/mm/2018"/>
Hora	<code><input type="time"></code>	<input type="text" value="--:00"/>
Fecha y hora local	<code><input type="datetime-local"></code>	<input type="text" value="dd/mm/aaaa --:--"/>
Mes	<code><input type="month"></code>	<input type="text" value="----- de -----"/>
Semana	<code><input type="week"></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`.

11.6. 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><input type="radio"></code>	Hombre <input type="radio"/> Mujer <input type="radio"/>
Casilla verificación (0..N opciones)	<code><input type="checkbox"></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>
```

11.7. 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	<code><select></code> <code><option>...</option></code> <code></select></code>	<input type="text" value="Selecciona tu SO ▼"/>
Lista (abierta) de opciones	<code><datalist></code>	

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>
```

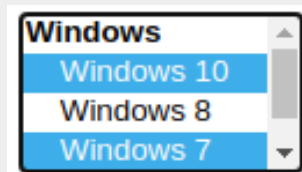
Selecciona tu SO ▼

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.

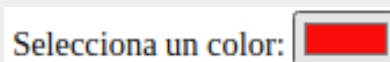
```
<form name="formulario" method="post" action="/send.php">
  <label>¿Qué navegador usas?:</label>
  <input list="browsers" name="myBrowser" >
  <datalist id="browsers">
    <option value="Chrome"></option>
    <option value="Firefox"></option>
    <option value="Internet Explorer"></option>
    <option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



11.8. 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.

11.9. 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.

11.10. Barra de progreso <progress>

El control progress se puede utilizar para mostrar información de forma visual, pero debemos tener en cuenta que al igual que con la etiqueta `meter`, debemos usar JavaScript, ya que en caso contrario su funcionalidad sería muy básica.

La etiqueta `progress` tendrá dos atributos: `max` y `value`, y se recomienda poner dentro de la etiqueta el valor que debería verse en esa barra de progreso para que en el caso de no ser soportada la etiqueta por el navegador se conozca su valor. Otros ejemplos podéis encontrarlos en <https://www.htmlquick.com/es/reference/tags/progress.html>

```
<form name="formulario" method="post" action="/send.php">
  <progress value="50" max="100">50%</progress>
</form>
```

```
const progress = document.querySelector("progress");

function increment() {
  progress.value += 10;
  if (progress.value < 100) {
    setTimeout(increment, 1000);
  }
}

setTimeout(increment, 1000);
```

11.11. Etiqueta para medidores <meter>

Mediante esta etiqueta podemos crear medidores que nos permita mostrar el nivel o escala de un cierto valor. Los atributos que podemos encontrar con esta etiqueta son: `value`, `min`, `max`, `low`, `high` y `optimum`. Siendo estos tres últimos atributos opcionales.

```
<form name="formulario" method="post" action="/send.php">
  <meter min="0" max="100" value="91" low="50" high="90">
</form>
```

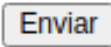


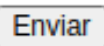
Ejemplos rápidos los podéis encontrar en:

<https://www.htmlquick.com/es/reference/tags/meter.html>

Nota: El valor `low` siempre será menor que el `high`. Además, ambos deberán estar entre `min` y `max`.

11.12. 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:


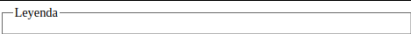
Tipo de botón	Etiqueta & atributo	Cómo se ve
Botón de envío	<code><input type="submit"></code>	
Botón de envío con imagen	<code><input type="image" src="enviar.png" width="80" height="28"></code>	
Botón para borrar el formulario	<code><input type="reset"></code>	
Botón sin funcionalidad	<code><input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

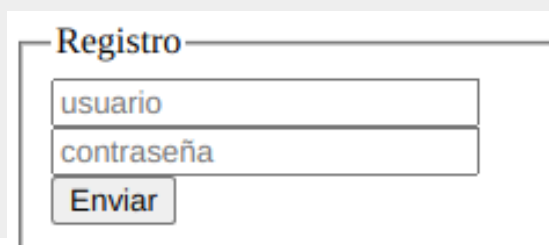
11.13. Organización de campos

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><fieldset> </fieldset></code>	
Leyenda para la etiqueta <code>fieldset</code>	<code><legend>Leyenda</legend></code>	
Relación de campo y texto	<code><label>texto</label></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.

12. Validación de formularios

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
<code>min</code>	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>
```