

Desarrollo Web

Estructura básica en HTML



Índice

Introducción	4
Comentarios	6
Estructura de una página web	6
Organización del head	6
Organización del body	8
Formatos de texto en HTML	8
Párrafos.	11
La etiqueta <pre>	13
Hipervínculos	13
Hipervínculos o enlaces internos. Etiqueta <a>	13
Hipervínculos externos. Atributo href.	14
Imágenes como enlaces	16
Listas	16
Listas no ordenadas	16
Listas ordenadas	17
Listas de definiciones o de descripciones	17
Tablas	19
Unificación de celdas	19
Celdas de encabezado	20
El atributo scope	22
Imágenes	23
Videos	26
Audio	27
Formularios	27
Formas de envío de los datos de un formulario html.	
Métodos get y post	29
Ingreso de datos en formularios	29
Text	29
Label	32
Select	34
Textarea	35
Checkbox	36
Radiobutton	37
Botón submit	37
Botón de reset	38
Botón de imagen	38
<fieldset> y <legend>	38



Ficheros adjuntos	39
Campos ocultos	39
iframes	39
Rutas en HTML	40
Rutas de archivo absolutas	40
Rutas de archivo relativas	40
Apéndice Etiquetas HTML	41

Introducción

HTML (Lenguaje de Marcas de Hipertexto, del inglés HyperText Markup Language) es un “lenguaje” que se compone de **etiquetas** con las que iremos diciéndole al programa que nos muestre texto, imágenes, videos, etc.

El origen de HTML se remonta a 1980, cuando el físico Tim Berners-Lee, trabajador del CERN (Organización Europea para la Investigación Nuclear) propuso un nuevo sistema de "hipertexto" para compartir documentos.

HTML no es un lenguaje de programación propiamente dicho, aunque a veces se lo denomina lenguaje de programación coloquialmente, es simplemente una serie de etiquetas que el navegador interpretará de una u otra forma para mostrar distintos contenidos por pantalla.

HTML es el lenguaje que se emplea para el desarrollo de páginas de internet. Está compuesto por una serie de etiquetas que el navegador interpreta y da forma en la pantalla. HTML dispone de etiquetas para imágenes, hipervínculos que nos permiten dirigirnos a otras páginas, saltos de línea, listas, tablas, etc.

Clásicamente se dice que los lenguajes de programación incluyen tres capacidades básicas de generar flujos de procesos: la secuencial (secuencias de instrucciones), la condicional (capacidad para tomar decisiones o ejecutar un proceso u otro en función del valor de uno o varios parámetros) y la de repetición (capacidad para repetir un proceso un cierto número de veces). Los lenguajes clásicos como C, C++, Java, C#, Visual Basic, Fortran, etc. cuentan con estas capacidades. HTML no cuenta con ellas, no porque sea mejor ni peor, sino porque es una cosa distinta.

En resumen, podríamos decir que HTML no es un lenguaje de programación, es un lenguaje de maquetación web o lenguaje de etiquetas destinado a crear estructuras de documentos HTML.

Podríamos decir que HTML sirve para crear páginas web, darles estructura y contenido.

El estándar actual de HTML5 consta de tres “lenguajes” en uno, HTML, CSS y Javascript:

- Con HTML hacemos el “esqueleto” de nuestra página web. La disposición inicial de títulos, párrafos, imágenes, etc.
- Con CSS3, le damos la apariencia a nuestra página.
- Con Javascript aportamos interactividad y animación.

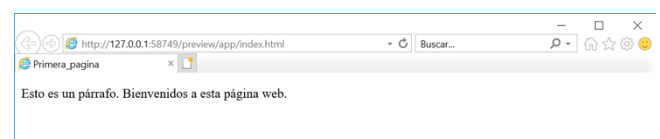
Un ejemplo sencillo de código **HTML** podría ser:

```
<html>
  <body>
    <p>Esto es un párrafo. Bienvenidos a
    esta página web.</p>
  </body>
</html>
```

Este ejemplo está formado por 3 etiquetas **HTML**. Como podemos observar cada una de las etiquetas debe acabar con su homóloga de cierre. En este caso la etiqueta `<html>` debe cerrarse con `</html>`, la etiqueta `<body>` con `</body>` y la etiqueta `<p>` con `</p>`.

Hay muchas más etiquetas que veremos más adelante, pero nos debe quedar claro que, en general, por cada etiqueta que abramos, deberemos incluir la correspondiente etiqueta de cierre (hay excepciones, las veremos más adelante). Así conseguiremos un código **HTML** bien formado y que los navegadores puedan interpretar sin ambigüedad.

Este sencillo ejemplo mostraría por pantalla lo siguiente.



¿Qué ocurriría si una etiqueta que abramos no tiene su correspondiente cierre? Digamos que se trataría de un código **HTML** mal construido, y los navegadores, esto lo pueden interpretar de distintas maneras. Quizás nos muestren la página tal y como esperábamos sin aparente error. Quizás nos muestren una página de error o se quede en blanco el navegador. Nuestro objetivo ha de ser siempre construir páginas **HTML** bien estructuradas y sin ambigüedades, es decir, hacer un correcto uso del lenguaje para que los navegadores puedan saber exactamente qué es lo que pretendemos mostrar.

Las etiquetas que no tienen par de cierre se cierran con una sintaxis similar a esta:

```
<meta/>
```

Actualmente **HTML** se encuentra en la versión 5. Programaremos en esta versión porque, en principio, todos los navegadores están optimizados para dicha versión.

¿Qué navegador debemos usar para probar nuestras páginas web? Muy sencillo, **todos**. Debemos asegurarnos de que nuestra página se pueda ver correctamente en cualquier dispositivo. Un programador web debe probar sus páginas en todos los dispositivos y navegadores posibles, ya que puede darse el caso de que los navegadores interpreten la página de diferentes formas y, en un navegador nuestra página se vea perfecta y en otro se vea descuadrada. Si usamos las reglas **HTML** de forma correcta esta situación no debería darse.

Comentarios

La sintaxis para usar comentarios en **HTML5** es:

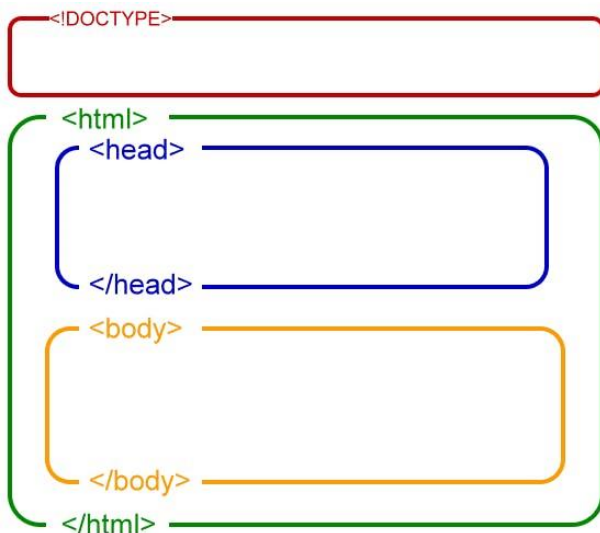
```
<!--Aquí va el comentario-->
```

Los comentarios HTML son visibles para cualquiera que vea el código fuente de la página, pero no se representan cuando el navegador procesa el documento HTML.

Como buena práctica se debemos comentar todo lo posible nuestro código. Facilitará el seguimiento por parte de otro programador o incluso para nosotros mismos en caso de que volvamos a retomar el código transcurrido un tiempo.

Estructura de una página web

Una página web **HTML** consta en principio de dos partes bien diferenciadas, el **head** y el **body**, y en ellas se distribuyen las etiquetas:



```
<!DOCTYPE html>
<html>
  <head>
    <title>Título de la página
web</title>
    ...
  </head>
  <body>
    Cuerpo de la página web
  </body>
</html>
```

Por convenio se suele poner como nombre **index.html** a la página central de cualquier proyecto WEB, pero no es obligatorio.

Organización del head

En el **head** (o cabecera de la página) se encuentran los metadatos (datos que el usuario no ve pero que al navegador le resultan útiles), el **title** de la página y los posibles **links** a páginas **CSS** o **Javascript**:

La siguiente tabla muestra un resumen de elementos que pueden ir dentro de la etiqueta head.

Etiqueta	Función	¿Obligatoria?
<title>	Da un título al documento HTML	Sí
<base>	Define ruta de acceso	No
<link>	Define archivos vinculados	No
<meta>	Define metadatos como descripción y palabras clave	No
<script>	Delimita scripts incluidos	No
<style>	Delimita definición de estilos	No

No es necesario que nuestro **head** contenga todas estas etiquetas. La única etiqueta obligatoria es **title**.

Las etiquetas meta no tienen par de cierre.

La etiqueta **Description** es fundamental para los buscadores, da una pista sobre la temática de la página y les sirve para categorizarla.

La etiqueta **Keywords** ya no tiene tanta importancia para los buscadores debido a los abusos que se ha hecho de ella. Hacer abuso de palabras clave en esta etiqueta tiene penalización por parte de los buscadores. En ella van palabras clave que a los buscadores les servirán para categorizar nuestra página y saber cuál es la temática.

<title></title>: Define el título del documento y nos cambia el título que aparece en la pestaña del navegador y además tiene mucha importancia de cara a los buscadores. Solamente puede contener texto y cualquier otra etiqueta contenida no será interpretada. La etiqueta **<title>** es obligatoria.

<title>Primera página</title>

El orden de las etiquetas es indiferente excepto la etiqueta **link**, que, en caso de existir, lo ideal es ponerla justo a continuación de la etiqueta **title**.

Etiqueta <base>. Sirve para indicar la **URL** base en caso de especificarse **URLs** relativas dentro de la página web. Por ejemplo:

```
<base
href="http://www.páginaprincipal.com/imag
es/" target="_blank">
```

Haría que, si escribimos como ruta de una imagen "logo.png", dicha ruta sea en realidad:

<http://www.páginaprincipal.com/images/logo.png>

Etiquetas <link>. Sirven para indicar que el documento **html** está relacionado con otro archivo o recurso externo. Enlaza nuestra página web con otras páginas externas, de tipo **CSS**, **Javascript**, etc. No tiene etiqueta de cierre.

Por ejemplo:

```
<link rel="stylesheet" type="text/css"
href="estilos.css"/>
```

En este caso indicamos que el documento **HTML** está vinculado a un archivo de estilos **css**, **javascript**, etc, lo veremos más adelante.

Etiqueta <style>. Sirve para incluir estilos **CSS** que permiten dotar de colores, bordes, imágenes de fondo, etc. a los elementos de la página web.

Etiquetas <meta>. Sirven para incluir información que no se muestra como parte de la página web, pero sirve para informar a los navegadores de características de la página web, como su descripción breve y sus palabras clave.

Ejemplo:

```
<meta name="description"
content="Programación HTML, CSS y
Javascript">
<meta name="keywords" content="curso,
web, programación, aprender">
```

En este caso las etiquetas les indican a los buscadores el contenido de nuestras páginas (**description**) y algunas palabras clave (**keywords**) para su localización. Esto es muy útil para que nuestra página aparezca en los buscadores (google, bing, yahoo, etc.).

Etiquetas <script>. Sirven para incluir código en lenguajes de script, como es el caso de **javascript**. También lo veremos más adelante.

Dentro de la cabecera en muchas páginas se incluye código en **JavaScript**, que es un lenguaje de programación que los navegadores son capaces de reconocer e interpretar. El código **JavaScript** se reconoce porque va comprendido entre las etiquetas `<script></script>`:

```
<script>
  Aquí iría el código
</script>
```

El siguiente ejemplo sería una implementación de `head` perfecta para una página con contenido **HTML** y **CSS**:

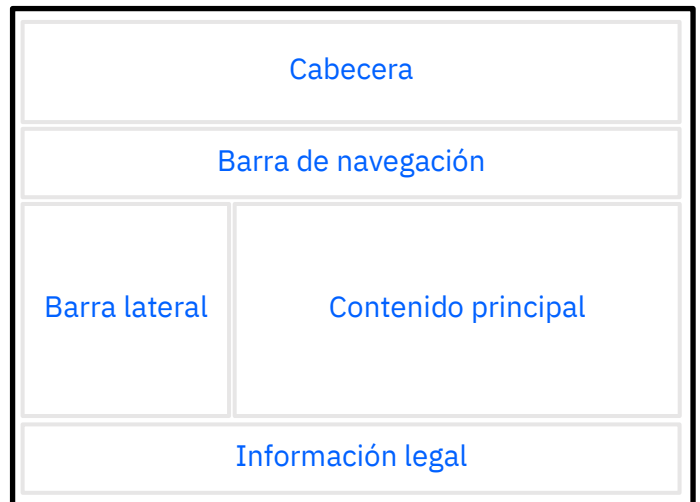
```
<head>
  <meta charset="utf-8">
  <meta name="Description"
content="Página de prueba"/>
  <meta name="Keywords" content=" HTML,
CSS, Javascript"/>
  <title> Curso de HTML, CSS,
Javascript </title>
  <link rel="stylesheet"
href="Hojadeestilos.css"/>
</head>
```

Organización del body

A continuación de la etiqueta de cierre del `head` iría la de apertura del `body`.

En el `body` (o cuerpo de la página) introducimos la parte visible de la página.

Antiguamente se dividía la página en celdas con la etiqueta `<table></table>`, posteriormente paso a usarse la etiqueta `<div></div>`, lo que se denominó modelo caja-contenedor, hoy en día se usa el siguiente esquema:



Esta distribución es orientativa, se puede prescindir de cualquier elemento o cambiarlo de sitio según nuestro diseño. Solo el **HEAD** y el **BODY** son obligatorios, pero incluir todos estos elementos mejora el posicionamiento **SEO**.

`header` y `footer` se pueden aplicar no solo al documento, también a cualquier parte de él, es decir, podemos poner, por ejemplo, un `header` y un `footer` dentro de etiquetas `section`.

Formatos de texto en HTML

A la hora de trabajar con textos en HTML tenemos una serie de etiquetas que escribimos envolviendo la palabra o el texto y que transforman ese texto en el formato que nosotros le hayamos querido dar. Algunas de estas etiquetas están no recomendadas (deprecated) por lo que no debemos emplearlas. Debido a su amplia difusión en el pasado conviene conocer los que fueron usos tradicionales de estas etiquetas, pero hoy en día han sido sustituidas por estilos CSS. No obstante, siguen estando disponibles.

Etiqueta	Uso	Observaciones
...	Poner texto en negrita	Puede ser sustituido por CSS.
...	Poner texto en negrita	Puede ser sustituido por CSS.
<i>...</i>	Poner texto en cursiva	Puede ser sustituido por CSS.
...	Poner texto en cursiva	Puede ser sustituido por CSS.
<u>...</u>	Poner texto subrayado	Deprecated. Sustituir por CSS.
<small>...</small>	Poner texto más pequeño	Puede ser sustituido por CSS.
<big>...</big>	Poner texto más grande	Puede ser sustituido por CSS.
<sub>...</sub>	Poner texto subíndice	Puede ser sustituido por CSS.
<sup>...</sup>	Poner texto superíndice	Puede ser sustituido por CSS.
<strike>...</strike>	Poner texto como tachado	Deprecated. Sustituir por CSS.
<s>...</s>	Poner texto como tachado	Deprecated. Sustituir por CSS.
...	Poner texto como tachado	Puede ser sustituido por CSS.

Etiquetas y su uso para adaptar en el formato en la programación HTML

Nuestro lenguaje se mostraría de la siguiente forma:

```
<!DOCTYPE html>
<html lang="es">

<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible"
content="IE=edge">
  <meta name="viewport"
content="width=device-width, initial-
scale=1.0">
  <title>Document</title>
</head>

<body>
  <b>Poner texto en negrita</b><br>
  <strong>Poner texto en
negrita</strong><br>
  <i>Poner texto en cursiva</i><br>
  <em>Poner texto en cursiva</em><br>
  <u>Poner texto subrayado</u><br>
  <small>Poner texto más
pequeño</small><br>
  <big>Poner texto más grande</big><br>
  <sub>Poner texto subíndice</sub><br>
  <sup>Poner texto
superíndice</sup><br>
  <strike>Poner texto como
tachado</strike><br>
  <s>Poner texto como tachado</s><br>
  <del>Poner texto como
tachado</del><br>
</body>

</html>
```

Mientras que la salida en el navegador sería la siguiente:

Poner texto en negrita

Poner texto en negrita

Poner texto en cursiva

Poner texto en cursiva

Poner texto subrayado

Poner texto más pequeño

Poner texto más grande

Poner texto subíndice

Poner texto superíndice

~~Poner texto como tachado~~

~~Poner texto como tachado~~

~~Poner texto como tachado~~

Encabezados

Los encabezados son lo que en cualquier noticia de prensa entenderíamos como los títulos y los subtítulos.

Para poner los títulos se usan las etiquetas `<h></h>`, que van desde `<h1></h1>` hasta `<h6></h6>`, siendo `<h1></h1>` la más relevante y `<h6></h6>` la menos importante. No solo nos cambian el tamaño del texto, los buscadores también les dan importancia a estas etiquetas.

Los encabezados generan por sí solos un salto de línea.

Es recomendable que haya solo una etiqueta `<h1></h1>` en cada página web ya que a nivel de **SEO** el buscador siempre va a buscar el título de la WEB.

Cuando tenemos varios encabezados H, los agrupamos con la etiqueta `<hgroup></hgroup>`

Por ejemplo, esta sería una relación de todas las posibles etiquetas `H` que tenemos a nuestra disposición:

```
<body>
  <hgroup>
    <h1>Esto es un H1</h1>
    <h2>Esto es un H2</h2>
    <h3>Esto es un H3</h3>
    <h4>Esto es un H4</h4>
    <h5>Esto es un H5</h5>
    <h6>Esto es un H6</h6>
  </hgroup>
</body>
```

El resultado en nuestro navegador sería:

Esto es un H1

Esto es un H2

Esto es un H3

Esto es un H4

Esto es un H5

Esto es un H6

Párrafos

Para indicarle al navegador que queremos poner un texto en un párrafo, debemos escribirlo entre las etiquetas `<p>` y su cierre `</p>`, quedando el texto separado por un margen en blanco por encima y por abajo.

Cuando tenemos varios párrafos que están relacionados se suelen agrupar en una etiqueta `<article></article>`.

Un atributo muy usado en el pasado (hoy deprecado y sustituido por `text-align` desde CSS) fue `align`, que servía para establecer la alineación del texto dentro del párrafo.

```
text-align:
  center
  end
  inherit
  justify
```

Para separar un texto de otro o un párrafo de otro podemos utilizar una línea horizontal de un tamaño o un grosor determinado por nosotros. Este separador lo escribimos con la etiqueta `<hr>` y su grosor, color, etc. veremos cómo se cambia más adelante mediante CSS:

```
<p>Esto es un párrafo</p>
<hr>
<p>Esto es otro párrafo bajo una
línea de separación hecha con HR</p>
```

Resultado en el navegador:

Esto es un párrafo

Esto es otro párrafo bajo una línea de separación hecha con HR

```
<!doctype html>

<html lang="ES-es"> <!--Inicio de la página-->

<head>                                <!--Inicio del HEAD-->
  <meta charset="utf-8"/>

                                <!--Descripcion de la pagina-->
  <meta name="Description" content="Descripcion de la pagina"/>

                                <!--Palabras clave-->
  <meta name="Keywords" content="Palabras clave, separadas por comas" />

                                <!--Titulo de la pagina-->
  <title>Titulo de mi pagina</title>

                                <!--Vinculo a una hoja de estilos CSS externa-->
  <link rel="stylesheet" href="MiHojaDeEstilos.css"/>

</head>                                <!--Fin del HEAD-->

<body>                                <!--Comienzo del BODY-->

                                <!--Parrafo-->
  <p>Esto es un párrafo de texto. Para crear automáticamente un texto de prueba,
escribimos LoremXX, donde XX será el número de palabras que queremos crear. Y a
continuación le damos a la tecla TAB.
  </p>

  <br>                                <!--Salto de linea-->

  <pre>
Etiqueta de texto preformateado. El navegador interpreta el texto escrito tal y como
está, respetando los saltos de línea, espacios, etc. indicados.
Por ejemplo, ahora vamos a poner dos saltos de línea

...y luego seguiremos escribiendo.</pre>

</body>                                <!--Fin del BODY-->
</html>                                <!--Fin de la pagina WEB-->
```

Ejemplo de trabajo con párrafos.

La etiqueta <pre>

La etiqueta <pre> se denomina de “texto pre-formateado” y sirve para que el navegador interprete el texto escrito tal y como está, respetando los saltos de línea, espacios, etc. indicados.

En ausencia de la etiqueta <pre> el navegador no toma en consideración que en el código fuente existan saltos de línea, espacios, etc. en el texto. En cambio, utilizando la etiqueta <pre> y cerrándola con su correspondiente etiqueta, el navegador sí respetará los saltos de línea, espacios, tabuladores, etc.

```
<body>
  <p>Esto es un párrafo hecho
    sin usar texto preformateado.
    A pesar de estar escrito en
diferentes líneas
    el navegador lo mostrará todo
seguido
  </p>
  <br>
  <pre>Esto es un párrafo hecho
    usando texto preformateado.
    El navegador lo mostrará
    tal y como está escrito
    en diferentes líneas
  </pre>
</body>
```

Hipervínculos

Los enlaces o hipervínculos, también llamados hipertextos, son los textos o los objetos sobre los que podemos hacer clic para que nos lleven a otra parte del documento, a otra página web en el mismo sitio o a otra página de Internet, entre otras funciones.

Hipervínculos o enlaces internos.

Etiqueta <a>

Los enlaces internos o marcadores son enlaces dentro de la misma página. Es decir, al hacer clic en uno de ellos nos llevará a una posición distinta dentro de la misma página que estamos visualizando.

Para crear este tipo de enlaces hay que hacer dos operaciones:

- Establecer marcadores (anclas) a lo largo de la página (son los lugares a los que queremos saltar con los enlaces).
- Poner enlaces que salten a los marcadores.

El código de los marcadores se crea con los atributos **name** (no recomendado) o **id** (recomendado):

```
<a name="nombre_del_marcador">Texto
asociado al marcador</a> (No recomendado:
no es aceptado por las nuevas versiones
de HTML, aunque se uso bastante en el
pasado).
```

```
<a id="nombre_del_marcador">Texto
asociado al marcador</a>
```

El **name** o **id** de una etiqueta debe ser único, es decir, no puede haber dos etiquetas cuyo **name** o **id** sea el mismo dentro de un documento **HTML**.

Por ejemplo, imaginemos que al principio del código de nuestra página web tenemos la siguiente línea:

```
<a id="marcadorDeportes">Resumen
sección deportes</a>
```

Podemos poner un enlace interno al final de la página para que el usuario lo pueda pulsar y que el navegador le lleve al principio. Es decir, un enlace para que salte a un marcador:

```
Pulsa para volver al <a  
href="#marcadorDeportes">Inicio</a>
```

Hipervínculos externos. Atributo href

Un hipervínculo externo es un vínculo a otro sitio **web** en Internet (sitio externo). Es un vínculo a cualquier otro lugar fuera del sitio actual. Cuando ponemos un vínculo externo, escribimos la dirección completa de la página incluido <http://www....> Estas rutas que incluyen <http://...> las denominamos rutas absolutas.

En el hipervínculo distinguimos las siguientes partes:

- Las etiquetas de apertura y cierre del hipervínculo `<a>` y ``
- El atributo `href`, con un valor que se indica con el símbolo igual y la **URL** a la que dirige el hipervínculo dentro de las comillas.
- Un texto que es el que ve el usuario como **link**.

Por ejemplo:

```
<a href="http://www.miPágina.es"  
title="Titulo del enlace"  
target="_blank">Ir a miPágina.es</a>
```

Destino del hipervínculo. Atributo target

Cuando creamos un vínculo, por defecto el navegador abrirá la página web destino en la misma ventana, pero podemos pedirle al navegador que la abra “aparte”, es decir, en otra ventana.

Esto es útil por ejemplo si queremos abrir una página externa a nuestro sitio, pero sin que el visitante pierda la nuestra. Para ello utilizaremos el atributo `target` con alguna de las siguientes opciones.

Valores de `target` más habituales:

`_blank`: Abre el documento vinculado en una ventana nueva del navegador.

`_self`: Es la opción predeterminada o por defecto. Abre el documento vinculado en el mismo marco o ventana en el que se ha pulsado el **link**.

Ejemplo:

```
<a href="http://www.miPágina.es"  
title="Titulo del enlace"  
target="_blank">Ir a miPágina.es</a>
```

Como última cuestión, debemos tener en cuenta que es muy aconsejable poner un atributo extra cada vez que pongamos un hipervínculo en nuestras páginas. Este atributo es `'title'` y con él pondremos título a nuestro hipervínculo.

Así conseguiremos en la mayoría de los navegadores un efecto de ‘tooltip’ que consiste en que cuando ponemos el cursor encima del hipervínculo nos aparezca una información adicional que es la que hayamos puesto en el atributo `'title'`.

Ejemplo de página web con enlaces internos y externos:

```
<!DOCTYPE html>
<html lang="es">

<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible"
content="IE=edge">
  <meta name="viewport"
content="width=device-width, initial-
scale=1.0">
  <title>Document</title>
</head>

<body>
  <h1 id="titulo">Título de mi
página</h1>

  <p id="parrafo1">Lorem ipsum dolor
sit amet consectetur adipisicing elit.
Eveniet eos aliquid fugiat maiores iure
voluptate? Iusto hic placeat
beatae libero consequuntur, sapiente in
quae, velit fuga facere iure consequatur
quas, suscipit repellendus
assumenda vero. Architecto repellendus
delectus itaque neque? Consequatur eos
ea,

  cumque, cupiditate soluta
doloremque eveniet veritatis et sed
inventore fugit architecto tempore nihil?
Porro

  excepturi fuga expedita
consequatur, rerum adipisci
necessitatibus facilis animi tenetur
corporis! Sed, rem vero

  cum obcaecati recusandae amet
deleniti cupiditate assumenda error
consectetur modi ex dolore explicabo
temporibus suscipit animi
laudantium at nobis fugit ipsa aspernatur
ab nam! Provident recusandae ipsam maxime
rem ducimus.</p>

  <p id="parrafo2">Lorem ipsum, dolor
sit amet consectetur adipisicing elit.
Illo ab expedita facere natus corporis
```

```
consequuntur nobis eveniet ad
laborum aspernatur, quam similique libero
accusamus tempore distinctio deleniti
dicta vero? Ab possimus ipsam sit
neque aspernatur, rem sed dolores
blanditiis ullam aperiam nemo odio
praesentium ex, maiores
consequatur expedita? Cum enim unde
quaerat ex hic voluptatibus aliquam
eaque! Ut odio,

  animi ad soluta libero dolorum
recusandae! Eos nesciunt rem itaque
consectetur. Esse dolores explicabo enim
quo

  voluptates at numquam excepturi,
quia fugiat. Eligendi dolorem
necessitatibus illum explicabo veniam
perferendis

  eius officia tempora doloribus!
Excepturi minima dolore deserunt possimus
error dolores! Fugiat, porro eum

  minima ratione numquam tenetur
minus, voluptatem quidem molestiae ipsa
iste eveniet nobis asperiores alias ipsam
iure id quibusdam quaerat? Error
tempora officiis quisquam animi tenetur
magni libero dolor aut. Esse maxime

  odio dolores quas expedita
incidunt id consequatur! Deserunt
consequuntur, molestiae officiis
distinctio tempore

  libero dignissimos qui vitae odit
quisquam dolore odio delectus! Vero,
omnis debitis saepe atque cumque

  consequuntur quidem nisi,
quisquam maxime obcaecati, numquam iusto
necessitatibus nobis magnam dicta?
Maxime architecto quibusdam, modi
accusamus alias amet reprehenderit
adipisci enim.

  </p>
  <a id="enlace"
href="http://www.santillana.es"
title="www.santillana.es"
target="_blank">Ir a santillana.es</a>
  <br>
  <a href="#titulo">Inicio</a>
  <br>
  <br>
```

```
<a href="#parrafo3">tercer
parrafo</a>

</body>

</html>
```

Imágenes como enlaces

Para poner una imagen como enlace basta con crear un enlace y añadir en su interior en lugar de texto, una imagen. Veamos un ejemplo:

```
<a href="http://www.miPágina.es">
  
</a>
```

La sintaxis para poner una imagen se verá más adelante. De momento el alumno solo debe entender que las imágenes también pueden funcionar como enlaces.

Listas

Las listas en **HTML** nos permiten crear conjuntos de elementos en forma de lista dentro de una página, todos los cuales irán precedidos, generalmente, por un guion o número.

Los tipos de listas en **HTML** son los siguientes:

- Listas no ordenadas
- Listas ordenadas
- Listas de definiciones

Listas no ordenadas

Las listas no ordenadas son aquellas que se encuentran entre las etiquetas ```` (ul indica unordered list), etiqueta de apertura y cierre respectivamente. Si queremos añadir un nuevo punto, lo tendremos que hacer dentro de las etiquetas ````.

Si no le indicamos nada a la etiqueta ``, la viñeta o marca que indica que se trata de un elemento de una lista se generará de forma automática. Por defecto las listas desordenadas están compuestas por puntos. Pero si queremos definir nosotros mismo el símbolo del punto o marca a emplear, debemos indicarlo específicamente. En el pasado se usaba el atributo `"type"`, hoy en día no recomendado (deprecated). Con este atributo se podía hacer que la lista estuviera definida por puntos negros (`li type="disc"`), por puntos con el fondo blanco (`li type="circle"`) o por cuadrados (`li type="square"`).

La sintaxis recomendada para indicar la apariencia se basa en usar CSS como indicamos a continuación. Veremos con detenimiento cómo se debe hacer de forma correcta cuando se vean los estilos CSS para listas:

```
<ul style="list-style-type:disc">
<ul style="list-style-type:circle">
<ul style="list-style-type:square">
```

Ejemplo de lista no ordenadas:

```
<ul>
  <li>Primer elemento de la lista
sin ordenar</li>
  <li>Segundo elemento de la lista
sin ordenar</li>
  <li>Tercer elemento de la lista
sin ordenar</li>
</ul>
```


Listas ordenadas

Si lo que pretendemos es definir una lista ordenada, lo tendremos que hacer entre las etiquetas `` `` (ol indica ordered list). Además, cada elemento de la lista se escribirá con la misma etiqueta que para las listas no ordenadas: ``. Pero al ser listas ordenadas los símbolos por defecto serán números y éstos se irán generando automáticamente por orden, conforme escribamos nuevos elementos de la lista.

```
<ol>
  <li>Primer elemento de la lista
ordenada</li>
  <li>Segundo elemento de la lista
ordenada</li>
  <li>Tercer elemento de la lista
ordenada</li>
</ol>
```

Mediante CSS se pueden cambiar los tipos de números que nos mostrará nuestra lista ordenada. Veremos que mediante CSS podemos poner números romanos, etc.

Listas de definiciones o de descripciones

Este tipo de listas no es de uso frecuente, por lo que vamos a citarlas solo por si encontramos este tipo de código en alguna página **web**, poder interpretar su significado. Las listas de definiciones se usan cuando queremos hacer una enumeración tipo “diccionario”, donde tenemos varios términos y su definición o descripción.

Por ejemplo, esto serían varios términos y sus definiciones:

Término	Definición o descripción
FTP	Protocolo para transmisión de ficheros entre ordenadores
HTML	Lenguaje de etiquetas empleado para generar páginas web
PHP	Lenguaje interpretado en servidor que permite generar páginas web dinámicas

Estos términos y sus definiciones o descripciones podríamos ponerlos de varias maneras dentro de una página **web** (como texto sin más, como lista ordenada, no ordenada...) y una de estas maneras es ponerlo como lista de definiciones.

Para crear una lista de definiciones debemos usar las etiquetas `<dl>`, `<dt>` y `<dd>`. Vamos a explicarlas por partes:

La etiqueta `<dl>` indica que dentro de ella va a ir una lista de definiciones o lista de descripciones.

La etiqueta `<dt>` indica que dentro de ella va un término que vamos a definir.

La etiqueta `<dd>` nos dice que dentro de ella se encuentra una definición o descripción asociada a un término. Un término podría tener varias descripciones. Por ejemplo, el término Autor podría tener como descripciones: Mateo Renzi, Olivo Pascua, Jorge Guillén.

Los listados de definición se separarán automáticamente si escribimos varios de ellos.

Si lo deseamos, podemos combinar unos tipos de listas con otros. Por ejemplo, tener listas ordenadas dentro de cada elemento de una lista desordenada.

```
<dl>
  <dt>Café</dt>
  <dd>Bebida caliente</dd>
  <dt>Refresco</dt>
  <dd>Bebida fría</dd>
</dl>
```

Todas las listas que hemos visto se pueden mezclar según nuestras necesidades. Es decir, podemos meter listas dentro de otras, crear listas ordenadas dentro de listas no ordenadas, etc. Cualquier combinación que se nos ocurra es posible.

Por ejemplo:

```
<!doctype html>

<html lang="ES-es"> <!--Inicio de la
página-->

  <head>                                <!--Inicio del
HEAD-->
    <meta charset="utf-8"/>

                                <!--Descripción
de la página-->
    <meta
name="Description" content="Descripción
de la página"/>

                                <!--Palabras
clave-->
    <meta name="Keywords"
content="Palabras clave, separadas por
comas" />

                                <!--Título de la
página-->
    <title>Título de mi
página</title>
```

```
                                <!--Vínculo a una
hoja de estilos CSS externa-->
    <link rel="stylesheet"
href="MiHojaDeEstilos.css"/>

  </head>                                <!--Fin del HEAD--
-->

  <body>                                <!--Comienzo del
BODY-->

                                <!--Lista
ordenada-->
    <ol>
      <li>Primer elemento de la
lista ordenada</li>
      <li>Segundo elemento de la
lista ordenada</li>
      <li>Tercer elemento de la
lista ordenada</li>
    </ol>

    <br>
    <br>

                                <!--Lista no
ordenada-->
    <ul>
      <li>Primer elemento de la
lista no ordenada</li>
      <li>Segundo elemento de la
lista no ordenada</li>
      <li>Tercer elemento de la
lista no ordenada</li>
    </ul>

    <br>
    <br>

                                <!--Lista no
ordenada dentro de otra ordenada-->
    <ol>
      <li>Primer elemento de la
lista ordenada</li>
      <li>
        <ul>
          <li>Segundo de la
ordenada y primero de la lista no
ordenada</li>
```

```

                <li>Segundo de la
ordenada y segundo de la lista no
ordenada</li>
                <li>Segundo de la
ordenada y tercero de la lista no
ordenada</li>
            </ul>
        </li>
        <li>Tercer elemento de la
lista ordenada</li>
    </ol>

    <br />
    <br />
    <p>Lista por definición</p>
    <br />
    <dl>
        <dt>Coffee</dt>
        <dd>Black hot drink</dd>
        <dt>Milk</dt>
        <dd>White cold drink</dd>
    </dl>

</body>                <!--Fin del BODY-
->
</html>                <!--Fin de la pagina
WEB-->

```

Esto hará que tu página se visualice correctamente, ya que algunos navegadores tienen problemas representando celdas vacías. Ejemplo:

```
<td>&nbsp;</td>
```

Ejemplo de tabla sencilla:

```

<table border="1px" >
    <caption>Tabla simple de
3x3</caption>
    <tr>
        <td>Celda A</td>
        <td>Celda B</td>
        <td>Celda B</td>
    </tr>
    <tr>
        <td>Celda C</td>
        <td>Celda D</td>
        <td>Celda E</td>
    </tr>
    <tr>
        <td>Celda F</td>
        <td>Celda G</td>
        <td>Celda H</td>
    </tr>
</table>

```

Tablas

Las tablas pueden ser consideradas como un grupo de filas donde cada una de ellas contiene un grupo de celdas (columnas). Una tabla puede ser insertada en un documento **HTML** usando tres elementos básicos: el elemento **TABLE** (estructura contenedora principal), el elemento **TR** (contenedor de fila) y el elemento **TD** (celda).

Cuando el contenido de una celda debe ser vacío, deberías usar un espacio en blanco (que en **HTML** se escribe como ` `) como su contenido.

Por defecto las tablas en HTML no tienen borde. Si queremos que nuestra tabla disponga de bordes, debemos especificarlo en la declaración de la misma con el atributo `border` seguido del grosor del `border`. En nuestro ejemplo hemos creado un borde para nuestra página con un grosor de 1 pixel:

```
<table border="1px" >
```

Unificación de celdas

En algunas ocasiones, puede ser necesario unificar dos o más celdas en una sola que pasará a ocupar el lugar de aquellas afectadas.

Estas unificaciones pueden ser llevadas a cabo con los atributos “[rowspan](#)” (para unificación vertical) y “[colspan](#)” (para unificación horizontal).

```
<table border="1">
  <caption>Tabla con celdas unidas
  mediante colspan</caption>
  <tr>
    <th>NOMBRE</th>
    <th>Día 1</th>
    <th>Día 2</th>
    <th>Día 3</th>
    <th>Día 4</th>
  </tr>
  <tr>
    <th>Mike (lastimado)</th>
    <td colspan="3">0 km</td>
    <td>4 km</td>
  </tr>
  <tr>
    <th>Susan</th>
    <td>23 km</td>
    <td>18 km</td>
    <td>19 km</td>
    <td>15 km</td>
  </tr>
</table>
```

El siguiente ejemplo ilustra la unificación vertical de celdas en una tabla:

```
<!--Tabla con
celdas unidas por rowspan-->
<table border="1">
  <caption>Tabla con celdas unidas
  mediante rowspan</caption>
  <tr>
    <th>&nbsp;</th> <!--Celda vacía-->
  </tr>
  <tr>
    <th>Básico</th>
    <th>Completo</th>
  </tr>
  <tr>
    <th>Instalación</th>
    <td rowspan="2">Gratis!</td>
    <td>$49.99</td>
  </tr>
  <tr>
    <th>Primer año</th>
    <td>$19.99</td>
  </tr>
  <tr>
    <th>Segundo año</th>
    <td>$9.99</td>
    <td>$29.99</td>
  </tr>
</table>
```

Cuando se utilizan conjuntamente en una misma tabla, los atributos [colspan](#) y [rowspan](#) deben ser cuidadosamente declarados de forma que no produzcan celdas superpuestas.

Celdas de encabezado

Hay dos tipos de celdas que pueden ser definidas en una tabla **HTML**. Una de ellas es la celda simple (elemento **TD**), ya definido anteriormente, y la otra es un tipo especial de celda (elemento **TH** o table header, cabecera de tabla) que contiene información de encabezado para un conjunto de celdas específicas.

Los navegadores representan normalmente el contenido de las celdas especiales como texto centrado y en negrita, atributos que también pueden ser visualmente logrados con la utilización de celdas normales (elemento **TD**). Entonces, ¿para qué son útiles estos encabezados? Cuando utilizamos **th** la celda queda definida como encabezado, no solo tiene el aspecto de un encabezado. Por poner un símil, no es lo mismo vestirse de policía sin serlo, que ser policía. No es lo mismo una celda que parece un encabezado sin estar definida como tal, que una celda definida realmente como encabezado. Los navegadores para personas invidentes identifican este tipo de encabezados y le dan un tratamiento especial. Además, algunos motores de búsqueda (bing, google, yahoo) dan un tratamiento distinto a este tipo de celdas, y algunos navegadores crean efectos especiales para este tipo de celdas.

Mediante el elemento **caption**, podemos definir el título de una tabla. Este título debería describir de una manera breve y precisa el contenido y la naturaleza de la tabla y es habitualmente representado en algún lugar cercano a la tabla (su posición puede ser establecida usando CSS). El elemento **caption** solo está permitido si va justo después de la apertura de la tabla.

```
<h1>Su pedido</h1>
<table border="1px">
  <tr>
    <th scope="col">Nombre
producto</th>
    <th scope="col">Precio
unitario</th>
    <th scope="col">Unidades</th>
    <th scope="col">Subtotal</th>
  </tr>
  <tr>
    <td>Reproductor MP3 (80
GB)</td>
    <td>192.02</td>
    <td>1</td>
    <td>192.02</td>
  </tr>
</table>
```

```
<td>Fundas de colores</td>
<td>2.50</td>
<td>5</td>
<td>12.50 </td>
</tr>
<tr>
  <td>Reproductor de radio</td>
  <td>12.99</td>
  <td>1</td>
  <td>12.99</td>
</tr>
<tr>
  <th scope="row">TOTAL</th>
  <td>-</td>
  <td>7</td>
  <td><strong>207.51</strong></td>
</tr>
</table>
```

En el siguiente ejemplo, construiremos una tabla para mostrar información acerca del clima en los próximos días. Aquí, las celdas de encabezado, representadas por el elemento **th**, son ubicadas en la primera fila de la tabla, encima de las celdas comunes.

```
<table border="1px">
  <tr>
    <th>Hoy</th>
    <th>Mañana</th>
    <th>Jueves</th>
  </tr>
  <tr>
    <td>Soleado</td>
    <td>Mayormente soleado</td>
    <td>Parcialmente nublado</td>
  </tr>
```

Es fácil ver aquí cómo cada celda de encabezado en la tabla provee información para el resto de las celdas en la columna a la que pertenece.

Algunos agentes, como los navegadores de voz, hacen el mismo análisis cuando deben informar al usuario qué celda de encabezado está asociada a una celda en particular. Pero en algunos casos, las ambigüedades necesitan ser evitadas y es por este motivo que HTML provee algunos atributos como `scope`.

El atributo `scope`

El atributo `scope` provee un mecanismo para indicar explícitamente a qué celdas afecta una celda de encabezado. Este atributo solo puede ser declarado en una celda de encabezado y tomar los valores "`col`", "`row`", "`colgroup`" y "`rowgroup`". Los valores "`col`" y "`row`" indican que la celda de encabezado provee información para el resto de las celdas en la columna o fila (respectivamente) en que está presente. Los otros dos valores tendrán sentido más adelante en este tutorial.

En el siguiente ejemplo, la tabla presentada anteriormente ha sido mejorada con más celdas de encabezado, con el fin de aumentar la legibilidad. Aquí, la celda en la esquina superior izquierda de la tabla proveería información ambigua si el atributo `scope` no estuviera presente. En otras palabras, afectaría a las celdas en su columna, así como a las celdas en su fila.

La presencia del atributo `scope` ha dejado en claro que las celdas afectadas por este encabezado son aquellas en la misma fila.

```
<table border="1px">
  <tr>
    <th scope="row">Día</th>
    <th>Hoy</th>
    <th>Mañana</th>
    <th>Jueves</th>
  </tr>
  <tr>
    <th>Condición</th>
    <td>Soleado</td>
    <td>Mayormente soleado</td>
    <td>Parcialmente nublado</td>
  </tr>
  <tr>
    <th>Temperatura</th>
    <td>19°C</td>
    <td>17°C</td>
    <td>12°C</td>
  </tr>
  <tr>
    <th>Vientos</th>
    <td>E 13 km/h</td>
    <td>E 11 km/h</td>
    <td>S 16 km/h</td>
  </tr>
</table>
```

Imágenes

Las imágenes dentro de una página web se incluyen utilizando la etiqueta ``, que **no tiene una etiqueta correspondiente de cierre**.

Al usar la etiqueta ``, dentro de la misma hay que especificar la ruta donde se encuentra dicha imagen, ya sea una ruta a un directorio de nuestro disco duro o a una dirección de internet. Eso se hace con el atributo `src`.

Otro atributo, no obligatorio, pero sí muy recomendable, es el atributo `alt`, que nos permite mostrar un mensaje en caso de que la imagen no pueda ser encontrada.

El atributo `title` nos permite poner un título a nuestra imagen, muy recomendable de cara al posicionamiento **SEO**.

```

```

En este ejemplo hemos mostrado en nuestra página web una imagen especificando su dirección de internet mediante el atributo `src`, le hemos puesto un texto alternativo mediante el atributo `alt`, que se mostrará en caso de que no se encuentre la imagen y como título le hemos puesto “título de la foto”.

Los formatos de imagen más usados son **JPG**, **GIF** y **PNG**. Lo ideal es usar siempre que sea posible el formato .jpg, ya que nos brinda una muy buena calidad de imágenes sin que las mismas sean muy pesadas.

En la siguiente tabla se resumen los atributos que podemos encontrar trabajando con imágenes:

Atributo	Observaciones	Uso
src	Este atributo es obligatorio e indica el nombre del archivo de imagen (entre comillas) o la URL desde la que se va a obtener la imagen.	Obligatorio. Si no se incluye no se mostrará imagen alguna.
align	Permite controlar la alineación de una imagen con respecto a una línea de texto adyacente o a otras imágenes en esa línea. Los valores posibles son los ya conocidos left, right, middle, top, bottom.	Atributo obsoleto (deprecated). Sustituir por CSS.
alt	Entre comillas podremos escribir un texto que se mostrará si la imagen no llega a cargar, mientras se carga o, cuando visualizando ya la imagen, pasamos el ratón por encima.	Atributo requerido, se recomienda incluirlo, aunque si no se hace la imagen se mostrará.
width	Este atributo es opcional, pero podemos ponerlo para especificar al navegador que muestre la imagen con un tamaño específico. Significa “ancho de la imagen” que vamos a representar. Si no se escribe, se carga la imagen con el tamaño original.	Opcional. Indicar valor en pixeles. También se puede indicar con CSS.
height	Al igual que el atributo width, es opcional. Este atributo indica el alto de la imagen.	Opcional. Indicar valor en pixeles. También se puede indicar con CSS.
border	Con border especificamos el ancho del borde que rodea la imagen. Si se indica 0 equivale a “sin borde”.	Atributo obsoleto (deprecated). Sustituir por CSS.

Tabla resumen de los atributos que podemos encontrar trabajando con imágenes.

NOTA: Cuando usamos **width** y **height** puede que nuestra imagen se deforme. Para evitar eso, usaremos solo uno de los dos. De esta forma las proporciones de nuestra imagen se mantendrán.

Debemos tener en cuenta que, al especificar la dirección de la imagen, si esta imagen está en nuestro propio servidor podemos utilizar una dirección relativa, es decir, estas dos expresiones serían igualmente válidas:

```
  
  

```

También podremos usar como origen de la imagen otro servidor, lo que significa que el navegador irá a buscar la imagen en la ruta que le indiquemos. Pero en este caso, deberemos especificar la ruta completa, no será válido una ruta relativa. Ejemplo:

```

```

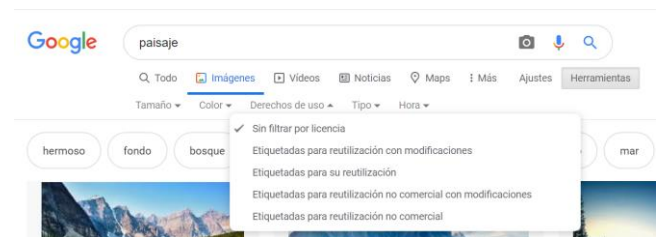
HTML nos ofrece también la opción de poner un pie de imagen para describir el contenido de nuestras imágenes.

Para poner un pie de imagen usaremos la etiqueta: `<figcaption></figcaption>`

Para mejorar el posicionamiento **SEO** conviene meter la etiqueta `` dentro de un `<figure></figure>`. Esta etiqueta no solo sirve para insertar imágenes, también cualquier elemento multimedia, videos, sonidos, animaciones flash, etc.

```
<figure>  
    
  <figcaption>Descripción de la  
imagen</figcaption>  
  <!--Debe ir dentro de figure-->  
</figure>
```

Nota: Hay que tener mucho cuidado con el tema de los derechos de autor en las imágenes. Google dispone de una opción para buscar imágenes libres de derechos:



El atributo `title` es el encargado de proporcionar un elemento de texto en forma de descripción emergente (tooltip) cuando posicionamos el puntero del ratón sobre la imagen.

Su misión es proporcionar información adicional al usuario sobre dicho elemento.

```

```



Vídeos

Trabajar con videos es muy parecido a hacerlo con imágenes.

¿Qué formatos soportan los navegadores actualmente?

Navegador	MP4	WebM	Ogg
Internet Explorer	SÍ	NO	NO
Chrome	SÍ	SÍ	SÍ
Firefox	SÍ	SÍ	SÍ
Safari	SÍ	NO	NO
Opera	SÍ	SÍ	SÍ
	(desde Opera 25)		

La etiqueta que nos permite utilizar videos es la etiqueta `<video></video>`.

Posee los siguientes atributos:

- **Src:** Permite especificar dónde se encuentra el video.
- **Controls:** Añade los controles visuales de video.
- **Autoplay:** El video se ejecuta automáticamente al cargarse la página.
- **Loop:** El video se reproduce en bucle.
- **Poster:** Permite definir una imagen .jpg que aparecerá antes de que el video se reproduzca.
- **Preload:** Permite especificar algunas características del video antes de que se cargue. Por ejemplo, cuántos fotogramas tiene el video, su duración, etc.

```
<video id="video" width="720px"
controls loop autoplay>
  <source
src="file://C:/Users/video.mp4">
  <source
src="file://C:/Users/video.ogg">
</video>
```

Para mejorar aspectos sobre accesibilidad se permiten añadir subtítulos en formato `vtt`. Para ello se usa la etiqueta `track`, que habilitará una nueva opción en las opciones del reproductor.

```
<track label="Español" kind="subtitle
s" srclang="es" src="subtitulos.vtt"
default>
```

Si queremos incrustar en nuestra página HTML procedente de Youtube deberemos hacerlo con la etiqueta `<iframe>`.

Para ello, en la página del video de Youtube, con el botón derecho se nos abrirá un menú emergente y deberemos pulsar la opción de copiar “código de inserción”:



Ahora ya tenemos copiado en nuestro portapapeles el código necesario para incrustar el video en una página HTML. Lo incluimos en nuestra página web y listo:

```
<iframe width="640" height="360"
  src="https://www.Youtube.com/embed/f0W8Y09GVek"
  title="Paisajes Hermosos del
Mundo 4K"
  frameborder="0"
allow="accelerometer; autoplay;
  clipboard-write; encrypted-media;
gyroscope;
  picture-in-picture"
allowfullscreen>
</iframe>
```

A veces podemos encontrar con que el vídeo de Youtube aparece en negro. Ello es debido a que el vídeo posee derechos de autor y no se permite su uso en otras páginas web.

Audio

Actualmente no se recomienda el uso de audio en las páginas web. Es algo incómodo para el usuario, sobre todo cuando abre varias páginas y cada una hace sonar una canción, todas a la vez. No obstante, es interesante conocer cómo se usa esta herramienta.

Los formatos soportados por **HTML** son .mp3 y .ogg.

La etiqueta que nos permite utilizar audio es `<audio></audio>`

Tiene los siguientes atributos:

- **Src**: Necesario para especificar la ruta del archivo de audio.
- **Controls**: Muestra el panel de control del audio, con botones como play, stop, volumen+, volumen-, etc.
- **Autoplay**: Especifica que el audio se reproduzca automáticamente al cargar la página.
- **Loop**: Especifica que el audio se reproduzca en bucle.

Formularios

Los formularios en **HTML** sirven al propósito de recolectar información proporcionada por los visitantes del sitio, la cual es luego enviada nuevamente al servidor para ser tratada.

Para su correcto funcionamiento es importante que el formulario provisto en **HTML** sea acompañado de un código del lado servidor, al que denominaremos "agente procesador", que se encargará de recibir y procesar la información como el autor vea conveniente. Este procesamiento puede consistir en, por ejemplo, el almacenamiento de la información o su envío mediante correo electrónico.

Un formulario, identificado en HTML por las etiquetas `<form>``</form>`, es básicamente un contenedor para controles. Cada control, en un formulario, está pensado para recolectar información ingresada por los usuarios, en formas que pueden ir desde líneas de texto, a subida de archivos, pasando por opciones, fechas, contraseñas y mucho más. Una vez que los usuarios han rellenado el formulario con los datos, pueden enviarlo de regreso al servidor para que el agente procesador administre la información recolectada.

Los usuarios interaccionan con los formularios a través de los llamados controles. Un control vamos a definirlo, de forma simplificada, como un objeto que aparece en la pantalla y que es modificable por el usuario. Por ejemplo, un botón, un cuadro de texto, un menú desplegable, etc.

Los formularios suelen llevar una etiqueta **action** que hace referencia a la página a la cual va a ir la información del formulario:

```
<form action = "pagina.php"></form>
```

Si no se va a mandar a ninguna otra página se deja vacío para mandar la información a la misma página en la que estamos.

```
<form action = ""></form>
```

Después se suele poner el atributo **method**, que puede ser **post** o **get**.

La diferencia entre los métodos **get** y **post** radica en la forma de enviar los datos a la página cuando se pulsa el botón "Enviar". Mientras que el método **get** envía los datos usando la URL, el método **post** los envía de forma que no podemos verlos (en un segundo plano u "ocultos" al usuario).

En el siguiente ejemplo podemos ver el resultado de un formulario hecho con el método **get**:

Se puede apreciar como los datos introducidos en el formulario aparecen en la URL de la página cuando se manda al servidor. Este método es muy poco fiable ya que es fácil interceptar los datos si se sabe cómo. Por ello siempre que se envíen formularios, lo ideal es hacerlo mediante el método **post** con el que los datos introducidos no son visibles en la URL.

Enctype, permite subir archivos al servidor. Puede ser:

```
<form enctype="application/x-www-form-urlencoded"></form>
```

O también:

```
<form enctype="multipart/form-data"></form>
```

Este último significa que permite enviar archivos desde el formulario al servidor, subir ficheros, etc.

Con **accept** le indicamos que tipo de ficheros nos permite subir:

```
<form enctype="multipart/form-data" accept-charset="UTF-8"></form>
```

Formas de envío de los datos de un formulario html. Métodos get y post

Cuando un usuario rellena un formulario en una página web los datos hay que enviarlos de alguna manera. Vamos a considerar las dos formas de envío de datos posibles: usando el método **POST** o usando el método **GET**.

Por ejemplo:

```
<form action="http://www.curso de
HTML.com/prog/newuser" method="get">
```

En el ejemplo anterior la acción que se ejecutará cuando el usuario pulse el botón “Enviar” (**submit**) será el envío de los datos a la **url** especificada usando el método **get**.

Como ya hemos mencionado, la diferencia entre los métodos **get** y **post** radica en la forma de enviar los datos a la página cuando se pulsa el botón “Enviar”. Mientras que el método **get** envía los datos usando la **URL**, el método **post** los envía de forma que no podemos verlos (en un segundo plano u “ocultos” al usuario).

Un resultado usando el método **get**, a modo de ejemplo, podría ser el siguiente:

```
http://www.CursodeHTML.com/newuser.php?nombr
e=Pepe&apellido=Flores&email=h52turam%40uco.e
s&sexo=Mujer
```

En esta **URL** podemos distinguir varias partes:

```
http://www.CursodeHTML.com/newuser.php
```

es la dirección web en sí.

El símbolo **?** indica dónde empiezan los parámetros que se reciben desde el formulario que ha enviado los datos a la página.

Después del símbolo **?** aparecen parejas de datos con su nombre y valor separadas por el símbolo **&**.

Las parejas dato1=valor1, dato2=valor2, dato3=valor3... reflejan el nombre y el valor de los campos enviados por el formulario.

Por ejemplo: nombre=Pepe, apellidos=Flores, etc. nos dice que el campo del formulario que se denomina nombre llega con valor “Pepe” mientras que el campo del formulario que se denomina apellidos llega con valor “Flores”. Estos valores son recibidos en la página web de destino del formulario.

Tened en cuenta que para separar la primera pareja de la dirección web en sí se usa el símbolo **‘?’** y para separar las restantes parejas entre sí se usa el símbolo **‘&’**.

Ingreso de datos en formularios

Los controles de entrada de datos en formularios suelen ser controles visuales y permiten la introducción de datos o selección de opciones al usuario. Su uso depende del tipo de control y también del tipo de información que pueden obtener.

Los elementos de entrada de un formulario pueden ser definidos mediante el uso de estos elementos:

HTML input, HTML textarea, HTML select y otros elementos **HTML**.

El resto de etiquetas ya van dentro del formulario entre otras tenemos:

Text

```
<input type="text" name="miFormulario"
id="etiquetaText" value="Angel" >
```

Crea una casilla para que el usuario pueda introducir datos y se puedan enviar. Con **textarea** hacemos lo mismo, pero creamos una ventana mayor. Para enviar dichos datos:

`<input type="submit">` // Con esto creo el botón de enviar.

El **name** es el nombre que el que nos referiremos a la hora de mandarlo al servidor, es decir, si le ponemos como nombre `miFormulario` luego lo recuperaremos con `$_post("miFormulario")` o con `$_get("miFormulario")`

Con **value** cambiamos el texto que aparece en el botón.

`<input type="submit" value="Aceptar">`

Por ejemplo:

```
<form>
  <input type="text" name
="miFormulario" id="etiquetaText"
value="Angel">
  <input type="submit" value="Aceptar">
</form>
```

Nos crearía un pequeño formulario con una etiqueta de texto (cuyo valor por defecto es "Angel") y un botón de enviar (**submit**) al que le hemos puesto el valor por defecto de "Aceptar"

Angel Aceptar

Los elementos de tipo **text** suelen tener por defecto un comportamiento de autocompletar, es decir, se suelen quedar en memoria las entradas anteriores, por ejemplo, en este caso hemos rellenado el formulario dos veces antes con los valores "Manolo" y "Manuel".

Al comenzar a rellenar el formulario por tercera vez, **HTML** nos sugiere las entradas anteriores:

Para cambiar esto tenemos dos etiquetas disponibles:

- **Autocomplete(on/off)**.
- **Novalidate(boolea)**. Si lo ponemos le decimos que no evalúe el formulario.

`<form name="formulario" id="formulario" method="get" autocomplete="off">`

De esta forma, ya no guardará en memoria entradas anteriores.

La etiqueta **type** en **HTML5** ya viene preparada para tomar y validar los siguientes valores:

- **Text**: Etiqueta de tipo texto genérico.
- **Email**: Etiqueta de tipo dirección de correo electrónico.
- **Search**: Etiqueta de tipo búsqueda.
- **URL**: Etiqueta de tipo dirección de página WEB.
- **Tel**: Etiqueta de tipo número de teléfono.
- **Number**: Etiqueta de tipo número genérico. Junto con MIN, MAX y STEP.
- **Range**: Etiqueta de tipo rango de valores. Junto con MIN, MAX y STEP.
- **Date**: Etiqueta de tipo fecha.
 - **Week**.
 - **Month**.
 - **Time**.
 - **Datetime**.

En el siguiente ejemplo hemos implementado un formulario tipo en el que se incluyen las etiquetas anteriormente mencionadas:

```
<form name="formulario" id="formulario"
method="get" autocomplete="off">
  Email: <input type="email"
name="mail" id="mail">
  <input type="submit"
value="Aceptar"><br><br>
  Edad;
  <input type="number" name="edad"
id="edad" min="18" max="100">
  <input type="submit"
value="Aceptar"><br><br>
  Buscar:
  <input type="search" name="buscar"
id="buscar"><br><br>
  URL:
  <input type="url" name="URL"
id="URL"><br><br>
  Edad:
  <input type="number" name="edad"
id="edad" min="0" max="100" step="5"
placeholder="Introduce edad"><br><br>
  Rango: <input type="range" min="0"
max="100" step="5"><br><br>
  Fecha: <input type="date"
name="fecha" id="fecha"><br><br>
  Hora: <input type="time" name="hora"
id="hora" required><br><br>
  Código postal: <input pattern="(0-
9){5}" placeholder="5 dígitos" br><br>
  <input type="submit"
value="Enviar"><br><br>
</form>
```

Email:

Edad:

Buscar:

URL:

Edad:

Rango:

Fecha:

Hora:

Código postal:

Otras etiquetas:

Con [placeholder](#) añadido un texto que le de pistas al usuario sobre cómo rellenar un campo. Desaparecerá cuando el usuario empiece a escribir.

Con el atributo [required](#) definimos un campo como obligatorio.

Con el atributo [multiple](#) podemos enviar más de un dato dentro de un campo. Los datos se separan con comas.

[Autofocus](#) lleva el “foco” al elemento que elijamos.

[Pattern](#) personaliza el campo definiendo rangos personalizados. Es decir, nos permite incluir [expresiones regulares](#) dentro de los **input**.

[Form](#) construye un elemento de formulario fuera del propio formulario. Hay que añadir el nombre de mi formulario:

Email2: `<input type="email" name="mail2" id="mail2" form="formulario">`

Datalist: Crea una lista de elementos en la que el usuario tendrá que elegir uno.

```
<form name="formulario" method="get">
  <datalist id="informacion">
    <option value="952773366"
label="    Telefono 1"></option>
    <option value="952734366"
label="    Telefono 2"></option>
    <option value="952773236"
label="    Telefono 3"></option>
  </datalist>
  Telefono:
  <input type="tel" name="telefono"
id="telefono" list="informacion">
</form>
```

Telefono:

952773366
Telefono 1

952734366
Telefono 2

952773236
Telefono 3

Label

Hasta este momento siempre que queríamos disponer un mensaje antes o después de un control de formulario lo escribíamos sin más.

Existe en **HTML** un elemento que permite asociar un texto con un control de formulario. Esto será muy útil si se accede desde un navegador no gráfico o una persona ciega que utiliza un programa que lee en voz alta el contenido de la página.

Veamos como lo hacíamos hasta ahora:

Ingrese su nombre:

```
<input type="text" name="nombre"
size="20">
```

Utilizando el elemento **label** podemos hacer una referencia entre el texto y el control de entrada de datos:

```
<label for="nombre">Ingrese su
nombre:</label>
<input type="text" name="nombre"
size="20" id="nombre">
```

Veamos que hemos agregado:

Hemos definido un **id** a la marca **input**. Es decir, le hemos dado un “alias” con el que poder referirnos a él.

El elemento **label** tiene su etiqueta de apertura y de cierre, entre medio se dispone el texto a mostrar.

Para vincular esta **label** con el elemento **input** debemos inicializar la propiedad **for** con el nombre asignado a la propiedad **id** del elemento **input**.

Confeccionemos un ejemplo completo:

```
<form action="registrardatos.php"
method="post">
  <fieldset>
    <legend>Formulario de
comentarios.</legend>
    <label for="nombre">Ingrese su
nombre:</label>
    <input type="text" name="nombre"
size="30" id="nombre"><br>
    <label for="mail">Ingrese su
mail:</label>
    <input type="text" name="mail"
size="50" id="mail"><br>
    <label
for="comentarios">Comentarios:</label><br
>
    <textarea name="comentarios"
rows="5" cols="60" id="comentarios">
    </textarea>
    <br>
    <input type="submit"
value="Enviar">
  </fieldset>
</form>
```

El resultado en el navegador es:

Como podemos ver asociamos cada etiqueta con el correspondiente control de entrada de datos:

```
<label for="nombre">Ingrese su
nombre:</label>
<input type="text" name="nombre"
size="30" id="nombre"><br>
```

```
<label for="mail">Ingrese su
mail:</label>
<input type="text" name="mail"
size="50" id="mail"><br>
```

```
<label
for="comentarios">Comentarios:</label><br
>
<textarea name="comentarios" rows="5"
cols="60" id="comentarios">
</textarea>
```

Normalmente las propiedades `id` y `name` de los controles de entrada de datos (`input`, `textarea`, etc.) se les asigna el mismo nombre, aunque no es obligatorio.

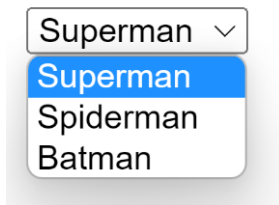
La propiedad `for` de la etiqueta `label` hace referencia al `id` del control y no al `name`, esto es importante si inicializamos con valores distintos el `id` y `name` de los controles.

Select

El elemento `<select>` define una lista desplegable en la que el usuario elegirá una de las opciones que le brindemos:

```
<form action="/action_page.php"
method="get" enctype="multipart/form-
data" accept-charset="UTF-8">
  <select name="peliculas">
    <option
value="Superman">Superman</option>
    <option
value="Spiderman">Spiderman</option>
    <option
value="Batman">Batman</option>
  </select>
</form>
```

Darí­a como salida:



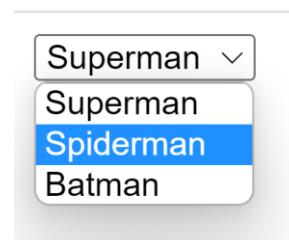
La etiqueta `value` es para que luego en el **backend** se reciba el valor y se suele poner el mismo valor que en el `<option>`

Los elementos `<option>` definen una opción que se puede seleccionar.

De forma predeterminada, se selecciona el primer elemento de la lista desplegable.

Para definir una opción preseleccionada, agregue el atributo `selected` a la opción:

```
<form action="/action_page.php"
method="get" enctype="multipart/form-
data" accept-charset="UTF-8">
  <select name="peliculas">
    <option
value="Superman">Superman</option>
    <option value=" Spiderman "
selected> Spiderman</option>
    <option
value="Batman">Batman</option>
  </select>
</form>
```



Utilizaremos el atributo `size` para especificar el número de valores visibles:

```
<form action="/action_page.php"
method="get" enctype="multipart/form-
data" accept-charset="UTF-8">
  <label for="cars">Elija un modelo de
coche:</label><br>
  <select id="cars" name="cars"
size="3">
    <option
value="volvo">Volvo</option>
    <option value="saab">Saab</option>
    <option value="fiat">Fiat</option>
    <option value="audi">Audi</option>
  </select>
</form>
```

Elija un modelo de coche:



Utilice el atributo `multiple` para permitir que el usuario seleccione más de un valor:

```
<form action="/action_page.php"
method="get" enctype="multipart/form-
data" accept-charset="UTF-8">
  <label for="cars">Choose a
car:</label>
  <select id="cars" name="cars"
size="4" multiple>
    <option
value="volvo">Volvo</option>
    <option value="saab">Saab</option>
    <option value="fiat">Fiat</option>
    <option value="audi">Audi</option>
  </select>
</form>
```

Textarea

Con el elemento `text` definíamos un área de entrada de texto de una sola línea.

El elemento `<textarea>` define un campo de entrada de varias líneas (un área de texto):


```
<form action="" method="get">
  <textarea name="message" rows="10"
cols="30">
    Aquí el usuario puede escribir un
texto más extenso que si usáramos el
atributo text.
  </textarea>
</form>
```

Aquí el usuario puede escribir un texto más extenso que si usáramos el atributo `text`.

El atributo `rows` especifica el número visible de líneas en un área de texto.

El atributo `cols` especifica el ancho visible de un área de texto.

Así es como se mostrará el código HTML anterior en un navegador:



También puede definir el tamaño del área de texto usando CSS:

```
<form action="" method="get">
  <textarea name="message"
style="width:200px; height:600px;">
    Aquí el usuario puede escribir un
    texto más extenso que si usáramos el
    atributo text.
  </textarea>
</form>
```

Aquí el usuario puede escribir un texto más extenso que si usáramos el atributo text.

Checkbox

Los **checkbox** o “casillas de verificación” son controles de formulario que permiten al usuario seleccionar y deseleccionar opciones individualmente. Aunque en ocasiones se muestran varios **checkbox** juntos, cada uno de ellos es completamente independiente del resto. Por este motivo, se utilizan cuando el usuario puede activar y desactivar varias opciones relacionadas, pero no excluyentes.

```
<form action="" method="get">
  Puestos de trabajo buscados <br/>
  <input name="puesto_directivo"
type="checkbox" value="direccion"/>
Dirección<br>
  <input name="puesto_tecnico"
type="checkbox" value="tecnico"/>
Técnico<br>
  <input name="puesto_empleado"
type="checkbox" value="empleado"/>
Empleado<br>
</form>
```

Puestos de trabajo buscados

- ☐ Dirección
- ☐ Técnico
- ☐ Empleado

El valor del atributo **type** para estos controles de formulario es **checkbox**. Como se muestra en el ejemplo anterior, el texto que se encuentra al lado de cada **checkbox** no se puede establecer mediante ningún atributo, por lo que es necesario añadirlo manualmente fuera del control del formulario. Si no se añade un texto al lado de la etiqueta **<input />** del **checkbox**, el usuario solo ve un pequeño cuadrado sin ninguna información relativa a la finalidad de ese **checkbox**.

El valor del atributo **value**, junto con el valor del atributo **name**, es la información que llega al servidor cuando el usuario envía el formulario.

Si se quiere mostrar un **checkbox** seleccionado por defecto, se utiliza el atributo **checked**. Si el valor del atributo es **checked**, el **checkbox** se muestra seleccionado. En cualquier otro caso, el **checkbox** permanece sin seleccionar. Aunque resulta redundante que el nombre y el valor del atributo sean idénticos, es obligatorio indicarlo de esta forma porque los atributos en **XHTML** no pueden tener valores vacíos:

```
<input type="checkbox"
checked="checked" ... /> Checkbox
seleccionado por defecto
```

☑ Checkbox seleccionado por defecto

Ejemplo:

```
Selecciona tus intereses:<br/>
<input name="Películas" type="checkbox"
/>Películas<br/>
<input name="Libros" type="checkbox"
checked="checked"/>Libros<br/>
<input name="Deportes" type="checkbox"
/>Internet<br/>
```

Selecciona tus intereses:

- ☐ Películas
☒ Libros
☐ Deportes

Radiobutton

Los controles de tipo `radiobutton` son similares a los controles de tipo `checkbox`, pero presentan una diferencia muy importante: son mutuamente excluyentes. Los `radiobutton` se utilizan cuando el usuario solamente puede escoger una opción entre las distintas opciones relacionadas que se le presentan. Cada vez que se selecciona una opción, automáticamente se deselectiona la otra opción que estaba seleccionaba.

```
Sexo: <br/>
<input type="radio" name="sexo"
value="hombre"/>Hombre<br>
<input type="radio" name="sexo"
value="mujer" />Mujer
```

Sexo:

- ☒ Hombre
☐ Mujer

El valor del atributo `type` para estos controles de formulario es `radio`. El atributo `name` se emplea para indicar los `radiobutton` que están relacionados. Por lo tanto, cuando varios `radiobutton` tienen el mismo valor en su atributo `name`, el navegador sabe que están relacionados y puede deselectionar una opción del grupo de `radiobutton` cuando se seleccione otra opción.

Botón submit

La mayoría de formularios dispone de un botón para enviar al servidor los datos introducidos por el usuario:

```
<input type="submit" name="buscar"
value="Buscar" />
```

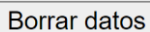
Buscar

El valor del atributo `type` para este control de formulario es `submit`. El navegador se encarga de enviar automáticamente los datos cuando el usuario pincha sobre este tipo de botón. El valor del atributo `value` es el texto que muestra el botón. Si no se establece el atributo `value`, el navegador muestra el texto predefinido `Enviar`.

Botón de reset

Aunque su uso era muy popular hace unos años, la mayoría de formularios modernos ya no utilizan este tipo de botón. Se trata de un botón especial que borra todos los datos introducidos por el usuario y devuelve el formulario a su estado original:

```
<input type="reset" name="limpiar"
value="Borrar datos" />
```



El valor del atributo `type` para este control de formulario es `reset`. Cuando el usuario pulsa este botón, el navegador borra toda la información introducida y muestra el formulario en su estado original. Si el formulario no contenía originalmente ningún valor, el botón de `reset` lo vuelve a mostrar vacío. Si el formulario contenía información, el botón `reset` vuelve a mostrar la misma información original.

Como es habitual en los botones de formulario, el atributo `value` permite establecer el texto que muestra el botón. Si no se utiliza este atributo, el navegador muestra el texto predefinido del botón, que en este caso es `Reset`.

Botón de imagen

El aspecto de los botones de formulario se puede personalizar por completo, ya que incluso es posible utilizar una imagen como botón:

```
<input type="image" name="enviar"
src="accept.png" />
```



El valor del atributo `type` para este control de formulario es `image`. El atributo `src` indica la **URL** de la imagen que debe mostrar el navegador en lugar del botón normal.

Su principal ventaja es que permite personalizar por completo la estética de los botones y mostrarlos con un aspecto homogéneo en todos los navegadores. El principal inconveniente es que ralentiza la carga del formulario y que, si se quiere modificar su aspecto, es necesario crear una nueva imagen.

<fieldset> y <legend>

El elemento `<fieldset>` se utiliza para agrupar datos relacionados en un formulario.

El elemento `<legend>` define un título para el elemento `<fieldset>`.

Ejemplo:

```
<form action="/action_page.php">
  <fieldset>
    <legend>Datos de contacto:</legend>
    <label
for="name">Nombre:</label><br>
    <input type="text" id="name"
name="name" value="Andrés"><br>
    <label
for="apellido">Apellido:</label><br>
    <input type="text" id="apellido"
name="apellido" value="García">
  <br><br>
    <input type="submit"
value="Submit">
  </fieldset>
</form>
```

Así es como se mostrará el código HTML anterior en un navegador:



Ficheros adjuntos

Los formularios también permiten adjuntar archivos para subirlos al servidor. Aunque desde el punto de vista de **HTML** y del navegador no existe ninguna limitación sobre el número, tipo o tamaño total de los archivos que se pueden adjuntar, todos los servidores añaden restricciones por motivos de seguridad.

Fichero adjunto

 Elegir...

Fichero adjunto

```
<input type="file" name="adjunto" />
```

Fichero adjunto

El valor del atributo **type** para este control de formulario es **file**. El navegador se encarga de mostrar un cuadro de texto donde aparece el nombre del archivo seleccionado y un botón que permite navegar por los directorios y archivos del ordenador del usuario.

Si se incluye un control para adjuntar archivos, es obligatorio añadir el atributo **enctype** en la etiqueta **<form>** del formulario. El valor del atributo **enctype** debe ser **multipart/form-data**, por lo que la etiqueta **<form>** de los formularios que permiten adjuntar archivos siempre es:

```
<form action="..." method="post"
enctype="multipart/form-data">
...
</form>
```

Campos ocultos

Los campos ocultos se emplean para añadir información oculta en el formulario:

```
<input type="hidden" name="url_previa"
value="/articulo/primer.html"/>
```

El valor del atributo **type** para este control de formulario es **hidden**. Los campos ocultos no se muestran por pantalla, de forma que el usuario desconoce que el formulario los incluye. Normalmente los campos ocultos se utilizan para incluir información que necesita el servidor pero que no es necesario o no es posible que la establezca el usuario.

Iframes

Un **iframe** o marco flotante **HTML** se utiliza para mostrar una página web dentro de una página web. Por ejemplo, para mostrar un video de Youtube dentro de nuestra página web, como vimos en la sección de videos.

Sintaxis:

```
<iframe src="url"
title="description"></iframe>
```

Sugerencia: es una buena práctica incluir siempre un atributo **title** para el archivo **<iframe>**. Los lectores de pantalla lo utilizan para leer cuál es el contenido del **iframe**.

Utilizaremos los atributos **height** y **width** para especificar el ancho y alto del **iframe** o en su defecto, lo haremos con CSS.

La altura y el ancho se especifican en píxeles de forma predeterminada:

Ejemplo:

```
<iframe src="demo_iframe.htm"
height="200" width="300"
title="Ej_Iframe"> </iframe>
```

La mayoría de los videos que importemos de Youtube tendrán este formato [Iframe](#).

Rutas en HTML

Una ruta de archivo describe la ubicación de un archivo en la estructura de carpetas de un sitio web.

- ****: El archivo "imagen.jpg" se encuentra en la misma carpeta que la página actual
- ****: El archivo "imagen.jpg" se encuentra en la carpeta de imágenes en la carpeta actual
- ****: El archivo "imagen.jpg" se encuentra en la carpeta de imágenes en la raíz de la web actual
- ****: El archivo "imagen.jpg" se encuentra en la carpeta un nivel por encima de la carpeta actual

Las rutas de archivo se utilizan cuando se vincula a archivos externos, como:

- Páginas web
- Imágenes, videos o audios.
- Hojas de estilo
- JavaScripts

Rutas de archivo absolutas

Una ruta de archivo absoluta es la URL completa de un archivo:

```

```

Rutas de archivo relativas

Una ruta de archivo relativa apunta a un archivo relativo a la página actual.

En el siguiente ejemplo, la ruta del archivo apunta a un archivo en la carpeta de imágenes ubicada en la raíz de la web actual:

```

```

En esta ocasión deberíamos tener la imagen llamada picture.jpg en el directorio que hemos especificado "images" de nuestro disco duro o servidor.

En el siguiente ejemplo, la ruta del archivo apunta a un archivo en la carpeta de imágenes ubicada en la carpeta un nivel por encima de la carpeta actual:

```

```

Se recomienda utilizar rutas de archivo relativas (si es posible).

Al usar rutas de archivo relativas, sus páginas web no estarán vinculadas a su URL base actual. Todos los enlaces funcionarán en su propio ordenador (localhost), así como en su dominio público actual y en sus futuros dominios públicos.

Apéndice Etiquetas HTML

Etiqueta	Función
<!--...-->	Define un comentario
<!DOCTYPE>	Define el tipo de documento
<a>	Define un hipervínculo
<abbr>	Define una abreviación
<address>	Define la información de contacto del autor / propietario del documento. Debe ir dentro del footer.
<area>	Define un área dentro de un mapa de imagen
<article>	Define un artículo. Importante para SEO.
<aside>	Define el contenido lateral del contenedor de una página. Importante para SEO.
<audio>	Define contenido de sonido
	Define texto en negrita
<base>	Especifica la base donde se abrirán todas las URL del documento
<bdi>	Aísla una parte del texto que puede tener un formato diferente del texto externo
<bdo>	Sobreescribe la dirección del texto
<blockquote>	Define una sección que tiene otra fuente. Se suele usar para citas largas.
<body>	Define el cuerpo del documento
 	Define un salto de línea
<button>	Define un botón clickeable
<canvas>	Se usa para dibujar gráficos en pantalla

<caption>	Define el título de una tabla
<cite>	Define el título de un trabajo. Libros, películas...
<dl>	Define una lista de definición
<dt>	Define un término (un ítem) en una lista de definición
	Define énfasis en un texto y lo pone en cursiva. Sustituye a la antigua etiqueta <i></i>
<embed>	Define el contenedor de una aplicación externa (no html)
<fieldset>	Grupo de elementos relacionados en un formulario
<figcaption>	Define el título para una figura <figure> Importante para SEO.
<figure>	Especifica auto-contenido Importante para SEO.
<footer>	Define el pie de página de un documento Importante para SEO.
<form>	Define un formulario html
<h1> a <h6>	Define encabezados o títulos Importante para SEO.
<head>	Define información acerca del documento
<header>	Define la sección de encabezado del documento Importante para SEO.
<hgroup>	Grupo de encabezado (<h1> a <h6>) Importante para SEO.
<hr>	Define un cambio de temática a partir de una línea dibujada
<html>	Define la raíz del documento
<i>	Define una parte del texto de modo alternativo
<iframe>	Define un frame en línea
	Define una imagen
<input>	Define un control de entrada de texto
<ins>	Define texto que ha sido insertado en un documento
<kbd>	Define entrada del teclado

<keygen>	Define un campo generador de claves para formularios
<label>	Define el rótulo para un elemento <input>
<legend>	Define un título para los elementos <fieldset>, <figure>, <details>
	Define un ítem de una lista
<link>	Define la relación entre un documento y un recurso externo (generalmente con hojas de estilo)
<map>	Define un mapa de imagen del cliente
<mark>	Define texto resaltado o marcado. Efecto “marcador fluorescente”.
<menu>	Define la lista de un menú
<meta>	Define un metadato de un documento
<meter>	Define una medida escalar en un rango conocido
<nav>	Define un link de navegación Importante para SEO.
<noscript>	Define un contenido alternativo para los usuarios que no soportan scripts del cliente
<object>	Define un objeto embebido
	Define una lista ordenada
<optgroup>	Define un grupo de opciones relacionadas en una lista desplegable
<option>	Define una opción en una lista desplegable
<output>	Define el resultado de un cálculo
<p>	Define un párrafo
<param>	Define un parámetro para un objeto
<pre>	Define texto pre-formateado
<progress>	Representa el progreso de una tarea en una barra
<q>	Define una cita corta
<rp>	Define que debe mostrar en navegadores que no soportan scripts de ruby

<rt>	Define una pronunciación de caracteres
<ruby>	Define una notación de ruby
<s>	Define texto que no es correcto
<samp>	Define un ejemplo de salida de un programa
<script>	Define un script del lado cliente
<section>	Define una sección de un documento Importante para SEO.
<select>	Define un drop-down list
<small>	Define textos legales.
<source>	Define los recursos para elementos multimedia
	Define una pequeña sección de un documento
	Define un texto en negrita y lo trata como importante. Importante para SEO
<style>	Define un estilo para la información de un documento
<sub>	Define un texto que es subíndice
<summary>	Define un encabezado visible para el elemento <details>
<sup>	Define un texto que es superíndice
<table>	Define una tabla
<tbody>	Define el cuerpo de una tabla
<td>	Define una celda en una tabla
<textarea>	Define un control de entrada de múltiples líneas
<tfoot>	Agrupar los footer contenidos en una tabla
<th>	Define una celda de encabezado en una tabla
<thead>	Agrupar los encabezados de una tabla
<time>	Define fecha / hora. Ej: <time datetime="2018/01/31" pubdate> Noticia publicada el día...</time>
<title>	Define un título para el documento. Importante para SEO.

<tr>	Define una fila en una tabla
<track>	Define texto de la pista para elementos multimedia (vídeo y audio)
	Define una lista desordenada
<var>	Define una variable
<video>	Define un vídeo o película
<wbr>	Define un posible salto de línea