

1. Introducción. Estándares web. Versiones

En 1997, el W3C comienza la normalización para conseguir la accesibilidad web (WAI, Web Accessibility Initiative). Este fue el punto de partida para la aparición de nuevos estándares que consiguieron que los contenidos web se puedan visualizar de la misma forma en diferentes navegadores. Además, con la ayuda de los estándares se consiguió que la web fuera un sistema abierto, gratuito y accesible para cualquier usuario.



Fig. 2.1. Estándares web del W3C.

¿Qué se entiende por estándares web?

Los estándares web se pueden definir como un conjunto de tecnologías específicas que se utilizan para crear sitios web.

Y ¿qué es un estándar en general?

Un estándar es un documento que contiene las especificaciones técnicas de la tecnología referida y que tienen que ser implementadas por los desarrolladores.

Por ejemplo, para implementar código HTML5, el estándar a seguir está disponible en el siguiente enlace de [WHATWG](#).

Además del W3C existen otros organismos dedicados a estandarización. Entre ellos está WHATWG (<https://whatwg.org/>, Web Hypertext Application Technology Working Group, agrupación de empresas y personas implicadas en el desarrollo del HTML entre las cuales están Apple, Google, Mozilla, etcétera), el [ECMA](#) (que publica el estándar que cumple JavaScript), entre otros.

Entre las ventajas más significativas del uso de estándares están las de:

- Mantener la web gratis y abierta para todos.
- Ayudar a simplificar el código fuente.
- Reducir del tiempo de desarrollo y mantenimiento de los contenidos web.
- Permitir la compatibilidad entre navegadores.
- Ayudar en la optimización de motores de búsqueda.

Entre los estándares web más conocidos están:

- HTML, *HyperText Markup Language*, lenguaje de marcas de hipertexto.
- XML, *eXtensible Markup Language*, lenguaje de marcas extensible.
- CSS, *Cascading Style Sheets*, hojas de estilo en cascada.
- JavaScript.

Respecto a las versiones de los diferentes estándares web, en la actualidad están vigentes:

- HTML. La versión 5. En el apartado siguiente se repasa su evolución brevemente.
- XML. La versión actual es la XML 1.0 5.^a edición (2008).
- CSS. La versión actual es la CSS3, publicada en 2011.
- Javascript. La versión actual estable es ES12 o ECMAScript 2021.

A. Evolución del HTML

HTML (*HyperText Markup Language*, lenguaje de marcas de hipertexto) es un lenguaje de marcas estándar y utilizado en la creación de páginas web. El código generado por este lenguaje de marcas establece la estructura del contenido de la página web como son el texto, las imágenes, los vídeos, el sonido, etcétera.

HTML se caracteriza por ser reconocido por todos los navegadores y muestra prácticamente el mismo aspecto de la página web, con independencia del sistema operativo de base.

Este lenguaje ha ido evolucionando desde las primeras versiones hasta el HTML5. Con el detalle de que, hasta la versión anterior del HTML5, las recomendaciones eran creadas por el W3C. Pero a partir de los años noventa, W3C deja el desarrollo del HTML y comienza con unas recomendaciones nuevas que siguen el estándar XML y que llamó XHTML. La versión HTML5 fue desarrollada en 2004 por el grupo WHATWG.

A partir de esta fecha se producen una serie de acercamientos y alejamientos entre el W3C y WHATWG hasta 2011, cuando separan sus trabajos definitivamente. Ambas organizaciones siguen sus desarrollos del HTML5 dando lugar a diferentes versiones (HTML 5.1 en 2016, HTML 5.2 en 2017) hasta 2018, fecha en la que el W3C abandona definitivamente el HTML 5.3 en favor de WHATWG con el compromiso de publicar una versión de HTML anualmente.

En la actualidad, HTML5 es el lenguaje de marcas para el web más utilizado y se tiene como estándar, aunque sigue evolucionando y adaptándose a las nuevas necesidades que plantean las nuevas tecnologías que van apareciendo. Es, por lo tanto, un lenguaje dinámico y vivo. Además, HTML5 funciona correctamente en dispositivos móviles.

Los desarrollos de esta unidad se basan en HTML5.

La siguiente tabla muestra la evolución de las diferentes versiones del HTML:

HTML	Creado por Tim Berners-Lee al comienzo de los años noventa. Fue el primer sistema de hipertexto para compartir documentos.
HTML 2.0	En 1995, el organismo IETF, Internet Engineering Task Force, organiza un grupo de trabajo de HTML y publica el primer estándar oficial HTML 2.0.

HTML 3.2	En 1997 se publica la versión 3.2, y es la primera recomendación de HTML publicada por el W3C. Esta versión incorpora la utilización de los applets de Java.
HTML 4.0	En 1998, W3C publica la versión 4.0, que incorpora la utilización de las hojas de estilo CSS, añade mejoras en la accesibilidad, la opción de incluir guiones o scripts en las páginas web, optimiza las tablas y los formularios, etcétera.
HTML 4.01	En 1999, el W3C publica la última especificación oficial de HTML, la 4.01, que es una actualización de la versión HTML 4.0 y no incluye novedades significativas. En este punto, W3C paraliza el proceso de estandarización del HTML.
HTML 5	En 2014, WHATWG publica la versión 5 de HTML. En dicha versión se añaden varias etiquetas semánticas nuevas que indican el significado del contenido, como <article>, <header> y <footer>, entre otras.

Tabla 2.1. Evolución del HTML.

Respecto a XHTML, la primera versión (1.0) se publicó en el año 2000 y es, básicamente, una combinación de la sintaxis de la versión HTML 4.01 y XML añadiendo algunas características específicas de XML. Este lenguaje de marcas XHTML resuelve el problema de la incompatibilidad del HTML 4.01 con las herramientas basadas en XML.

La versión actual es XHTML 2.0 y presenta varios problemas, entre ellos: no se añadió en el estándar nuevas funcionalidades demandadas por los programadores, no ofrece compatibilidad hacia atrás con las páginas web ya existentes, su sintaxis es muy estricta y presenta altas exigencias a los navegadores.

Todo ello, unido a que ya había aparecido el estándar HTML5, hizo que el W3C abandonase el proyecto. Esto no afectó mucho en general, ya que HTML5 acepta todo lo que es admitido en XHTML, aunque no al contrario.

2. Estructura de un documento HTML5

Una página web básica se compone de tres tipos de código:

- HTML5, que aporta los elementos estructurales.
- CSS, que aplica estilo a estos elementos.
- JavaScript, que posibilita la interacción dinámica de esos elementos.

Página web = Estructura HTML5 + Estilo CSS + Interacción JS

En esta unidad se estudia HTML5, así como las hojas de estilo CSS3. JavaScript y la integración en documentos HTML se ven en una unidad posterior.

Toda página HTML5 debe tener un prólogo y un ejemplar, y de esa forma es compatible con cualquier documento XML.

2.1. Prólogo

Un documento HTML5 debe tener una declaración de tipo de documento, en la que se indica que es de tipo HTML. Con esta declaración se activa el modo estándar del navegador web. El navegador sabe el tipo de documento que va a mostrar y las incorporaciones de HTML5 son interpretadas siempre que sea posible, o ignoradas si no las soporta.

```
<!DOCTYPE html>
```

2.2. Ejemplar

El ejemplar es el propio documento HTML5 y está delimitado por las marcas `<html>` y `</html>`.

El elemento raíz o nodo raíz es siempre `<html>`. Hay que añadir el atributo `lang`, que define el idioma del contenido del documento que se está creando, en este caso 'es', por español. Para HTML5 no es necesario.

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

El ejemplar consta de dos secciones: la cabecera y el cuerpo.

CABECERA

El elemento `head` (cabecera) contiene los metadatos del documento que incluyen información sobre la página, como son el título, la descripción, el autor, etcétera. Se pueden incluir enlaces a otros contenidos para completar la presentación del contenido del documento, como una hoja de estilo asociada o algunos scripts.

El contenido de la cabecera no se muestra en la página.

Por ejemplo:

```
<head>
    <meta charset="utf-8" />
    <title>TEXTO</title>
    <link rel="stylesheet" type="text/css" href="style-
original.css" />
</head>
```

En la cabecera se pueden incluir los siguientes elementos:

- <meta> permite incluir metadatos y posicionar la web en los buscadores. Un ejemplo de utilización de la etiqueta <meta> es para indicar la codificación, y de esa forma las palabras con acentos no mostrarán símbolos extraños en el navegador.

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

- <style> para la definición de estilos en línea. No es necesaria si se va a utilizar una hoja de estilo externa mediante la etiqueta <link>.

```
<style type="text/css">
    .....
</style>
```

- <title> descripción que identifica el documento y que se muestra en la pestaña de título del navegador.

```
<title>Mi página de prueba</title>
```

- <link> vincula las hojas de estilo como recurso externo. También permite incorporar código Javascript, imágenes o iconos desde archivos externos.

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

En este ejemplo, el atributo `rel` tiene el valor `stylesheet`, que le dice al navegador que el archivo `estilos.css` es un archivo CSS con los estilos necesarios para presentar la página.

- <script> para incluir un script en la página web. Hay dos opciones para indicar el código del script:

a) código en un archivo:

```
<script src="archivo.js" type="text/javascript"></script>
```

b) código insertado aquí:

```
<script type="text/javascript">Código de un script integrado en la
página</script>
```

El atributo `type` indica el tipo MIME, y que no es necesario en los elementos `style` y `script` en HTML5.



IMPORTANTE

El tipo MIME, *Multipurpose Internet Mail Extensions* o extensiones multipropósito de correo de Internet, especifica el tipo de dato del contenido del archivo. Por ejemplo, texto, imagen, audio, etcétera. Cada tipo de dato tiene un sufijo concreto. En el ejemplo anterior para JavaScript es `type="text/javascript"`.

CUERPO

Contiene la información de la página que se verá en el navegador. Está delimitado entre las etiquetas `<body>` y `</body>`.

En el apartado siguiente se estudian los diferentes elementos a incluir en el cuerpo del documento HTML.



IMPORTANTE

Para realizar los ejemplos, casos prácticos y, en general, toda la parte práctica del módulo se recomienda instalar al menos otro navegador web diferente del utilizado normalmente.

3. Identificación de etiquetas y atributos de HTML5

Las etiquetas siguen las mismas reglas que en XML, es decir, HTML tiene etiquetas de apertura (`<etiqueta>`) y de cierre (`</etiqueta>`), pero a diferencia de XML, que puede crear tantas etiquetas como se quiera o necesite, en HTML las etiquetas están predefinidas por el lenguaje.

Por este motivo, para aquellas páginas que, por la complejidad de alguno de sus componentes (elementos multimedia, enlaces, etcétera), necesitan añadir información para su correcta visualización, es preciso complementar las etiquetas predefinidas con atributos.



Fig. 2.2. Esquema de un elemento HTML.

Los atributos son pares **nombre=valor** escritos dentro de una etiqueta de apertura y detrás del nombre de dicha etiqueta. El *valor* puede estar entre “comillas dobles” o ‘simples’.

Cuando el valor de un atributo no es válido, el navegador lo ignora.

El formato o sintaxis general de una línea de código HTML es:

```
<etiqueta atributo1="valor1" atributo2="valor2">contenido</etiqueta>
```

Para un caso concreto, por ejemplo un enlace:

```
<a href="http://www.lmsgi.com" target="_blank">Ejemplo de enlace</a>
```

Donde:

- `<a>` es la etiqueta de apertura y `` la etiqueta de cierre. En este caso, el elemento `a` define un enlace a una página web, pero podría ser un archivo o una dirección de correo.
- `href` y `target` son los atributos. `href` indica la URL.
- `http://www.lmsgi.com` y `_blank` son los valores asignados a los atributos. En este caso concreto se está diciendo con `target="_blank"` que la página de destino hacia la que apunta se abra en otra ventana del navegador.
- El texto `Ejemplo de enlace` es el contenido.



IMPORTANTE

En HTML, las etiquetas y los atributos están definidos previamente, mientras que en XML los define el programador.

A. Clasificación de los atributos comunes según su funcionalidad

Cada etiqueta HTML5 puede tener sus propios atributos, pero hay algunos que son comunes a varias etiquetas. En general, los atributos se pueden agrupar según su funcionalidad en:

- Atributos básicos.
- Atributos de internacionalización.
- Atributos de eventos.
- Atributos de foco.

A.1 Atributos básicos

Los atributos básicos se pueden utilizar en la mayoría de etiquetas HTML.

Atributo	Descripción
title = "texto"	Establece el título del elemento HTML. El título es mostrado por los navegadores cuando el usuario pasa el ratón por encima del elemento.
id = "texto"	Establece un indicador único a cada elemento.
style = "texto"	Aplica al elemento HTML el estilo "texto" directamente.
class = "texto"	Establece la clase CSS que se aplica a los estilos del elemento.

Tabla 2.2. Atributos básicos de HTML.

A.2 Atributos de internacionalización o *i18n*

Estos atributos se utilizan en páginas que muestran sus contenidos en varios idiomas o en las que se quiera indicar de forma explícita el idioma de sus contenidos.

Atributo	Descripción
lang="codigo"	Indica el idioma del elemento. Ejemplo: lang="es".

Tabla 2.3. Atributos de internacionalización.

La recomendación RFC 1766 se refiere a códigos de países para Internet. El documento original está disponible en este [enlace a la web del IETF](#).

A.3 Atributos de eventos y de foco

Los atributos de eventos se utilizan en páginas web que incluyen código JavaScript para realizar acciones dinámicas sobre los elementos de la página.

Se llama foco (*focus*) cuando un control o elemento del documento ha sido seleccionado.



IMPORTANTE

La tabla completa de atributos de eventos y de foco la puedes encontrar en el archivo **UD2_Atributos_eventos_HTML.pdf** disponible en la sección **Material de trabajo > Descarga de documentos**, situada al final de la unidad.

B. Elementos HTML5

En HTML5 (al igual que XML), además de etiquetas y atributos, se utiliza el término elemento para referirse a las partes que componen un documento HTML.

En general, un elemento HTML consta de:

1. Una etiqueta de apertura.
2. Cero o más atributos.
3. Texto encerrado por la etiqueta. Es variable, ya que no todas las etiquetas pueden encerrar texto.
4. Una etiqueta de cierre. No siempre es necesaria.

En HTML existen dos tipos de elementos contenedores: los elementos en línea y los elementos en bloque.

- Elementos en línea o *Inline elements*. Ocupan el espacio necesario para mostrar sus contenidos. Cada elemento en línea se colocará a continuación del elemento anterior, sin introducir ningún salto de línea. Su contenido puede ser texto u otros elementos en línea. Por ejemplo, los enlaces, el salto de línea y el texto resaltado son elementos en línea. En este [enlace](#) tienes disponible información adicional sobre ellos.
- Elementos en bloque o *Block-level elements*. Definen bloques de contenido coherente, ya sea texto o imágenes. Un elemento en bloque incorpora un salto de línea antes y después del mismo. Por ejemplo, encabezados, párrafos, listas, divisiones o imágenes son elementos en bloque. En este [enlace](#) tienes más información sobre ellos.

Hay elementos que se pueden comportar como elementos en línea o en bloque, en función de las circunstancias.

En general se utiliza:

- `` para elementos en línea.
- `<div>` para elementos en bloque.

Por ejemplo, `<div>`, al poder contener otros elementos HTML, se suele utilizar para diseñar la página web, y para añadir estilo al bloque se utilizan las hojas de estilo.

En el caso de `` se suele utilizar para aplicar estilo al texto o agrupar elementos en línea.

En este [enlace a la web para desarrolladores de Mozilla](#) está disponible la lista de todos los elementos HTML.

B.1 Elementos de la estructura básica del documento

La estructura básica de un documento HTML contiene las siguientes etiquetas:

Elemento	Descripción
html	Las etiquetas <html> y </html> indican la apertura y el cierre de un documento HTML, y no debe añadirse nada antes o después de ellas excepto el DOCTYPE.
head	Las etiquetas <head> y </head> indican la apertura y el cierre de la cabecera del documento y la información que contiene no se muestra en el navegador, a excepción del contenido de la etiqueta <title>.
body	Las etiquetas <body> y </body> delimitan el cuerpo del documento HTML, y su contenido se muestra en el navegador.

Tabla 2.4. Estructura básica documento HTML.

EJEMPLO 1

En la sección **Ejemplos para trabajar y analizar**, situada al principio de la unidad, puedes encontrar el archivo **UN02_AP03_ejemplo_1.html**, con el código correspondiente a algunos de estos elementos básicos de todo documento HTML.

B.2 Elementos que dan formato al texto

La siguiente tabla incluye los principales elementos para dar formato al texto.

Elemento	Descripción
p	Delimita párrafos.
h <i>i</i>	Indica el nivel del encabezado. “ <i>i</i> ” es un valor entre 1 y 6. El tamaño de la letra del encabezado será mayor cuanto menor sea el valor de “ <i>i</i> ”. Solo se utilizan en títulos de párrafos.
br	Salto de línea.
b	Texto en negrita.
i / em	Texto en itálica o cursiva. Mejor utilizar .
u	Texto subrayado.

sup	Superíndice.
sub	Subíndice.
center	Texto centrado.
strong	Texto resaltado. Normalmente será en negrita, pero es posible que algún navegador lo muestre en cursiva y naranja.
blockquote	Es una cita textual y se representa como un párrafo indexado con respecto al margen.
hr	Es una línea horizontal. Se escribe </hr>.
comentario	El texto del comentario no es visible en el navegador. Se escribe <!-- comentario -->.

Tabla 2.5. Elementos que dan formato al texto.

EJEMPLO 2

En el archivo UN02_AP03_ejemplo_2.html puedes encontrar el código correspondiente a algunos de estos elementos de formato del texto. En el navegador Firefox se vería así:

Ejemplo 2: elementos de formato

Muestra un texto en **negrita**, *cursiva* y subrayado.

Un ejemplo de nota, al cual se le añade un sangrado o margen por la izquierda.

Otro encabezado

Ejemplo con superíndices ($E=mc^2$) y subíndices (CO_2).

Fig. 2.2. Ejemplo_2.html, formato de un texto.

HTML5 incluye la etiqueta `<hgroup>`, que se utiliza para agrupar un conjunto de uno o más elementos del tipo (h1-h6).

```

<header>
  <hgroup>
    <h2>Título de la página web</h2>
    <h2>Subtítulo</h2>
  </hgroup>
</header>

```

Observa que ahora se utiliza el elemento `header` y no `head`. La diferencia entre ellos es que `head`

contiene información no visible, tal como metadatos o links, y `header` puede contener logotipos, encabezados `h1`, menús de navegación `nav`, secciones, artículos..., y se incluye en el `<body>`.

B.3 Elementos de listas

En HTML5 se pueden utilizar tres tipos de listas:

- a) Ordenadas.
- b) Desordenadas.
- c) Listas de descripción.

La siguiente tabla describe los elementos asociados a cada una de ellas:

Elemento	Descripción
<code>ul</code>	Delimita los elementos que forman una lista desordenada. Anida elementos "li".
<code>ol</code>	Delimita los elementos que forman una lista numérica u ordenada. Anida elementos "li".
<code>li</code>	Indica cada uno de los elementos de una lista.
<code>dl</code>	Delimita los elementos que forman una lista de descripción.
<code>dt</code>	Cada uno de los términos que se definen de una lista de descripción.
<code>dd</code>	Cada una de las definiciones de una lista de descripción.

Tabla 2.6. Elementos de listas.

La lista de descripción sirve para crear un conjunto de elementos con pares concepto-descripción. En ella se especifican varios términos por su nombre y se escribirá una descripción o valor para cada uno de ellos.

En las listas ordenadas, por defecto, se utiliza el orden numérico, pero con el atributo `type` se puede indicar otro estilo de numeración.

Con el atributo `start` se indica el primer número de la lista. Por defecto es 1.

EJEMPLO 3

En el archivo **UN02_AP03_ejemplo_3.html** puedes encontrar el código correspondiente a los elementos de listas. En el navegador Firefox se vería así:

Ejemplo de lista desordenada: Tareas pendientes

- Hacer la compra
- Ordenar papeles
- Contestar correos

Ejemplo de lista ordenada: Prioridades

1. Contestar correos
2. Hacer la compra
3. Ordenar papeles

Ejemplo de lista de descripción: Detalles

- Contestar correos
 - Importante el correo pendiente al tutor
- Hacer la compra
 - No queda leche, huevos y pan
- Ordenar papeles
 - Los apuntes de Lenguaje de marcas

Fig. 2.3. Elementos de listas.

B.4. Elementos de tablas

Los elementos para definir una tabla en HTML5 son los siguientes:

Elemento	Descripción
table	Delimita el contenido de una tabla.
tr	Delimita cada una de las líneas de la tabla. Debe estar dentro de las etiquetas de la tabla.
td	Delimita el contenido de cada celda de la tabla. Debe estar dentro de las etiquetas de una fila.
colgroup	Permite agrupar columnas.
tbody	Permite agrupar líneas de la tabla.
thead	Define la línea cabecera de la tabla.
th	Delimita cada una de las celdas de la cabecera.
tfoot	Define la fila pie de la tabla.

Tabla 2.7. Elementos de tablas.

En el siguiente [enlace puedes consultar más información](#) sobre los atributos de tablas.

EJEMPLO 4

En el archivo **UN02_AP03_ejemplo_4.html** encontrarás el código correspondiente a los elementos de tablas. En el navegador Firefox se vería así:

Menú del comedor

Plato	Opciones
Primero	1. Ensalada 2. Lentejas 3. Macarrones
Segundo	1. Pescado 2. Pollo 3. Arroz
Postre	1. Yogur 2. Fruta

Fig. 2.4. Elementos de tablas.

B.5. Elementos de formularios

El formulario es un mecanismo de petición de información al usuario. Consta de un conjunto de campos, de tipos diferentes, que el usuario rellena desde una página web y envía a la URL indicada. Los elementos asociados a la definición y la gestión de formularios son los siguientes:

Elemento	Descripción
form	Encierra o delimita la definición de todos los campos del formulario.
input	Permite añadir en el formulario cajas de texto para textos cortos.
textarea	Caja de texto para texto largo.
select	Menú desplegable que permite elegir de una lista de opciones que se muestra. Para campos de selección.
option	Delimita las opciones de un menú desplegable que contiene este elemento.
button	Define un botón. Se diferencia de los botones hechos con input en que este elemento permite introducir en el botón otros elementos de HTML, como imágenes.
label	Etiqueta de un campo del formulario.
progress	Representa un avance o progreso en la ejecución de una tarea. Por ejemplo, en la descarga de un archivo.
meter	Es similar a <progress> y se utiliza para representar escalas de medidas conocidas, como la longitud, la masa, el peso, el uso de disco, etc.

Tabla 2.8. Elementos de formularios.

En la etiqueta de apertura <form> se han de indicar los atributos básicos:

- **action=""** Se indica la acción a realizar al enviar el formulario. En general, se indica el nombre de un archivo alojado en el servidor que se encargará de procesar la información. **method=""** (post o get) determina el método de transferencia de las variables. El método "post" envía los datos de forma no visible. El método "get" adjunta los datos a la URL a la que se redirige.
- **enctype=""** Indica el tipo de codificación de la información enviada. En el método "get" no es necesario utilizar este atributo, ya que no se realiza codificación, solo se cambian caracteres especiales como el espacio en blanco. En el método "post" se pueden utilizar los valores:
 - **application/x-www-form-urlencoded**: valor predeterminado y codifica todos los caracteres antes de enviarlos.
 - **multipart/form-data**: requerido al enviar archivos mediante un formulario y no

codifica la información.

- `text/plain`: no codifica la información y cambia los espacios por el "+".

siguiente código muestra un ejemplo de cabecera de un formulario:

```
<form action="http://www.lmsgi.com/UD2/formulario.php"
method="POST">
<!-- Utiliza codificación por defecto -->
<!-- Campo1 del formulario -->
<!-- Campo2 del formulario -->
<!-- Campo3 del formulario -->
</form>
```

B.6. Otras etiquetas para formularios

Etiqueta	Atributos	Descripción
<input>	<code>name=""</code>	Nombre que se le da al campo. Este nombre no es visible en el navegador, solo se utiliza para diferenciar los campos al enviar la información al servidor. Hace la función de nombre de variable a la que se asigna el valor del campo.
	<code>type=""</code>	Tipo de campo a utilizar. Puede ser de tipo text, password, checkbox, radio, file, hidden, submit, reset.
<label>	<code>for=""</code>	<p>Las etiquetas se utilizan para poner un texto o descripción de los campos de un formulario.</p> <p>Se utiliza la etiqueta <code><label></code> con el atributo <code>for</code> para indicar el nombre (atributo <code>name</code>) del campo asociado. Por ejemplo:</p> <pre><label for="campo1">Etiqueta</label> <input name="campo1" type="text"/></pre>
<select>		<p>Se utiliza para seleccionar una o varias opciones de una lista. Las opciones de la lista se indican con la etiqueta <code><option></option></code>. El nombre que se verá debe indicarse entre esas etiquetas.</p> <p>En cada <code>option</code>, el atributo <code>value=""</code> permite indicar el valor que se envía con el formulario, y el atributo <code>selected</code> permite indicar la opción seleccionada por defecto. Si no se indica, aparece como seleccionado el primer elemento de la lista.</p>
	<code>name=""</code>	Nombre del campo.
	<code>size=""</code>	Especifica el número de opciones visibles a la vez. Si no se indica nada o se le da valor uno, se presentará como un menú

		desplegable. Para valores mayores de uno aparece como una lista con barra de desplazamiento.
	multiple	Permite seleccionar más de una opción a la vez para el campo.

Tabla 2.9. Otras etiquetas para formularios.

Puedes consultar la lista completa de atributos de <input> en el archivo **UD2_Atributos_input_HTML.pdf** que puedes encontrar en la sección **Material de trabajo > Descarga de documentos**.

B.7 El elemento indicador de progreso

El elemento `progress` tiene los atributos siguientes relativos a valores que no se identifican con ningún tipo de unidad:

- `max`: define el trabajo total a realizar por la tarea, la duración de un vídeo, etcétera. El valor por defecto es 1.0.
- `value`: estado del progreso. El valor debe ser ≥ 0.0 y ≤ 1.0 , o el valor especificado en `max`.
- `position`: atributo de solo lectura que representa la posición actual del elemento `progress`. El valor de `position` es igual a `value/max`, o -1 si no se puede determinar la posición.

Las unidades son arbitrarias, y no se especifican.

```
<progress value="5" max="20">5</progress>
```

EJEMPLO 5

En el archivo **UN02_AP03_ejemplo_5.html** puedes encontrar el código correspondiente a los elementos para la utilización de formularios. En el navegador Firefox se vería así:

Ejemplo de utilización de formulario sencillo

Introducción de contraseña

Nombre:

Contraseña:

Fig. 2.5. Utilización de formularios.

En `action` no se ha indicado el programa que procesará la información enviada desde el formulario. Se verá más adelante.



EJEMPLO 6

En el archivo **UN02_AP03_ejemplo_6.html** encontrarás el código correspondiente a un ejemplo de uso de diferentes atributos type para input. Es un ejemplo muy sencillo y que no es procesado. Simplemente es una muestra de código validado.

EJEMPLO 7

En el archivo **UN02_AP03_ejemplo_7.html** puedes encontrar el código correspondiente a un ejemplo de utilización de formularios. En este ejemplo se ha utilizado una tabla para organizar los elementos dentro del formulario.

Formulario con diferentes tipos de campos

Nombre:	Apellido:	e-mail:
<input type="text"/>	<input type="text"/>	<input type="text"/>
contraseña:	<input type="password"/>	
Sexo	Profesión	Aficiones:
<input type="radio"/> Hombre	<input checked="" type="radio"/> Profesor	<input type="checkbox"/> Ciencia
<input type="radio"/> Mujer	<input type="radio"/> Programador	<input type="checkbox"/> Libros
	<input type="radio"/> Diseñador gráfico	<input type="checkbox"/> Deportes
Elija la herramienta de diseño web que prefiera:		
<input type="button" value="Herramienta web: -"/>		
Indique sus comentarios:		
<input type="text" value="Comentario:"/>		
<input type="button" value="Enviar"/> <input type="button" value="Limpiar todo"/>		

Fig. 2.6. Utilización de formularios con tablas.

En este [enlace del W3C](#) puedes encontrar más ejemplos de formularios.

B.8 Otros elementos

Otros elementos de HTML 5 que podemos utilizar son:

Elemento	Descripción
img	Inserta una imagen en una página web. Es obligatorio utilizar el atributo <code>src</code> para especificar el path o ubicación del archivo de imagen que se quiere insertar. Se puede escribir un texto alternativo con <code>alt</code> y dar dimensiones con los atributos <code>height</code> y <code>width</code> . <code></code>
blockquote	Contiene un bloque de texto con sangrado.
q	Contiene una cita y el navegador añade las marcas de citación.
br	Inserta una línea en blanco. No tiene etiqueta de cierre.
div	Permite agrupar varios elementos de bloque (párrafos, encabezados, listas, tablas, otras divisiones, etcétera). Si no se le da estilo a la división, el navegador no muestra nada. No hace falta definir divisiones cuando el grupo es de un solo elemento.
video	Permite incrustar vídeos en la página web. Puede reproducirse con autoplay, en bucle con loop, controlando la reproducción con controls, etc. <code><video src="video.mp4" width="640" height="480" controls></video></code>
audio	Permite insertar archivos de audio en diferentes formatos, como mp3 y ogg. También dispone de una interfaz de control sobre la reproducción de este con una API en JavaScript sin necesidad de plugins. <code><audio id="player" src="archivo.ogg" controls></audio></code>
canvas	Es un nuevo elemento de HTML5 y proporciona una API para dibujar formas, imágenes, textos, crear gradientes, aplicar transformaciones, etcétera. El llamado lienzo (canvas) es también como un rectángulo en la página web en la que, con scripts de JavaScript, por ejemplo, se puede dibujar.
mark	Permite representar un texto resaltado porque tiene una importancia destacada. <code><p>Algunos elementos de HTML5 como <mark>aside</mark> tienen una importancia especial por la funcionalidad que introducen.</p></code>
embed	Facilita un punto de integración en la página para una aplicación externa o un contenido interactivo, como un plug-in.

Tabla 2.10. Otros elementos de HTML 5.



IMPORTANTE

Puedes comprobar la compatibilidad HTML de tu navegador visitando el sitio web <http://www.html5test.com/>. La herramienta devuelve, con todo detalle, el grado de compatibilidad de tu navegador con el estándar HTML5.

En este enlace tienes disponible [una lista más completa de símbolos](#).

4. Etiquetas semánticas

En HTML5 se introduce el concepto de esquema (outline). Dicho esquema consiste en estructurar el contenido de la página en secciones y bloques de contenido.

Hasta HTML4, el contenido se jerarquizaba utilizando los títulos “hi”, y se siguen utilizando en HTML5, pero con algunas de las nuevas etiquetas semánticas se definen de forma explícita nuevas secciones y los títulos “hi” son relativos a cada una de esas secciones. De esta forma desaparece el posible problema de no respetar la jerarquía de títulos “hi”. En concreto, las etiquetas `<article>`, `<section>`, `<nav>` y `<aside>` definen secciones y reinician los títulos “hi” que contienen.

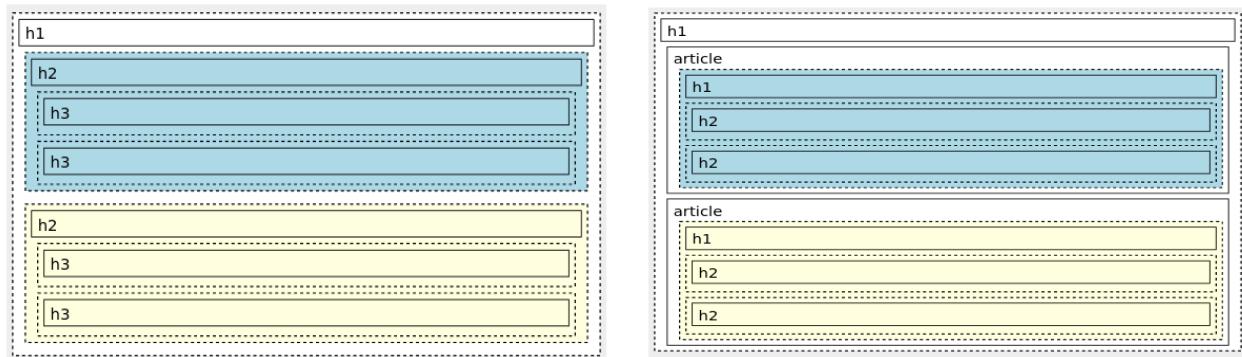


Tabla 2.11. Diferencias jerarquización entre HMTL4 y HTML5.

En general, las etiquetas semánticas de HTML5 se utilizan para estructurar los documentos HTML y evitar la necesidad de usar etiquetas `<div>` (versiones anteriores de HTML) para identificar cada bloque de la página. Utilizando etiquetas semánticas, el navegador al leer la página web, conoce cuál es la estructura de la página y la procesa correctamente.

Estas etiquetas estructurales son: **header**, **section**, **article**, **aside**, **nav**, **footer** y **main**.

Un ejemplo de estructura de una página web en HTML5 utilizando las etiquetas semánticas sería:

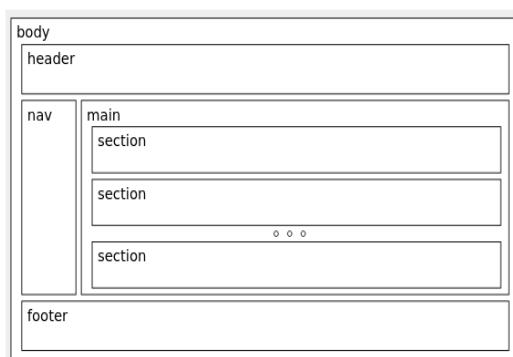


Fig. 2.9. Estructura de página web en HTML5.

La figura siguiente muestra la comparativa entre la estructura de una página en versiones anteriores a HTML5 y con HTML5:

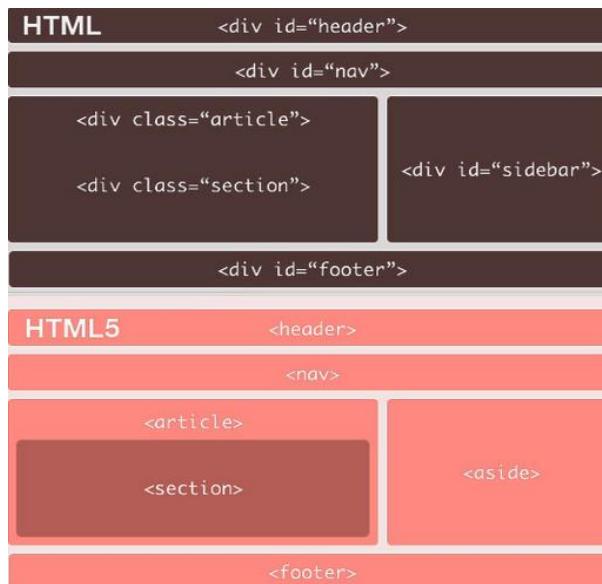


Fig. 2.10. Diferencias de estructura web entre HTML5 y versiones previas.

A continuación, se detalla cada una de estas nuevas etiquetas semánticas.

<header></header>	Contiene un grupo de elementos introductorios (encabezados, logos, menús globales) o de navegación específicos de una sección de la página web o de su totalidad. Por lo tanto, el <code>header</code> no tiene por qué ser único en un documento, se puede incluir un <code>header</code> por cada sección. No hay que confundir con el elemento <code>head</code> , que proporciona los metadatos para el documento. Puede ir dentro de <code>body</code> si es encabezado de toda la página y en <code>section</code> , <code>article</code> , <code>nav</code> o <code>aside</code> si el encabezado es solo para esos elementos.
<section></section>	Identifica las partes o zonas temáticas de los contenidos dentro de un documento o página web. Es decir, <code>section</code> , aunque es genérico, tiene significado semántico y la información que contiene debe estar relacionada de alguna forma. Una sección puede tener subsecciones, y utilizando elementos "hi" se sigue estructurando la página para jerarquizar el contenido, aunque la recomendación de HTML5 es utilizar el esquema (<code>outline</code>). Hasta HTML4 se utilizaba para esta función el elemento <code>div</code> , pero existe una util dif erencia entre <code>section</code> y <code>div</code> . Cuando se crean bloques cuyo contenido está relacionado se usa <code><section></code> , y cuando estos bloques responden a diseño o necesidades de estructura se usa <code><div></code> .
<article></article>	Define un contenido de un tema específico dentro de una página web. Su característica principal es que puede distribuirse de forma independiente. Se utiliza en mensajes de foro, comentarios enviados por los usuarios, artículos de periódicos, revistas y blogs. Es decir,

	contenidos que pueden publicarse o imprimirse de forma independiente. Una página web puede contener uno o varios artículos de temas relacionados o no. También se pueden anidar. Un elemento <code>article</code> puede contener secciones, y viceversa.
<code><aside></aside></code>	Agrupa contenido separado del contenido que acompaña. Suele aparecer al lado del contenido. Ejemplo típico son los anuncios que aparecen en los laterales, contenidos en barras laterales, citas, enlaces externos, etcétera. Puede estar dentro de <code>body</code> , en cuyo caso es contenido secundario en relación con la página web completa. Si va dentro de <code>section</code> o <code>article</code> es contenido secundario en relación con el elemento en el que esté incluido. Es el contenido propiamente, y su relación con otros elementos, lo que lo define.
<code><nav></nav></code>	Agrupa enlaces de navegación a las diferentes secciones de la página web o a otras páginas. Una página web puede contener varios <code>nav</code> , y su significado dependerá del elemento <code>header</code> , <code>body</code> o <code>aside</code> que lo contenga. Sus usos suelen ser: en una barra de navegación principal de la página web, para definir un conjunto de enlaces que apuntan a diferentes partes de la página actual o para una zona de búsqueda utilizada como mecanismo de navegación dentro de la página. Si aparecen enlaces a pie de página web no se utilizará el elemento <code>nav</code> .
<code><footer></footer></code>	Agrupa enlaces hacia otros sitios, secciones de la página o información general de la página, como son el autor, las licencias y el copyright, entre otras. Puede ir dentro de <code>body</code> si es pie de toda la página web y en <code>section</code> , <code>article</code> , <code>nav</code> o <code>aside</code> si es pie solo para esos elementos.
<code><main></main></code>	Establece el contenido principal del documento y que es único. Todo aquello que se repite en todas las páginas del sitio web (menús, pie de página, barras laterales, etcétera) no va en <code>main</code> . No tiene que haber más de un <code>main</code> por página y no puede ir dentro de <code>header</code> , <code>article</code> , <code>aside</code> , <code>footer</code> y <code>nav</code> .

Tabla 2.12. Etiquetas semánticas.

Es importante tener en cuenta que, utilizando las etiquetas semánticas, cada sección tiene su propio esquema de documento y no se necesitan los `div`, aunque se pueden seguir utilizando. Por otra parte, la jerarquía de títulos es relativa a la sección o artículo. De esa forma, además, se puede mover una sección completa a un documento diferente manteniendo el mismo esquema.

En el siguiente enlace tienes [información referente a las etiquetas semánticas](#).



EJEMPLO 1

En el archivo UN02_AP04_ejemplo_1.html encontrarás el código correspondiente a un ejemplo de utilización de etiquetas semánticas.

Ejemplo 8 Uso de etiquetas semánticas

Este es el header de la sección

Este es un título h1 dentro del header del article 1

Esto es un texto en un párrafo 1

Esto es el pie del article 1

Este es un título h1 dentro del header del article 2

Esto es un texto en un párrafo 2

Esto es el pie del article 2

Fig. 2.11. Uso de etiquetas semánticas.



IMPORTANTE

Las etiquetas semánticas no definen un estilo predeterminado.

Las etiquetas semánticas no tienen un estilo predeterminado que el navegador vaya a asignar directamente. Esto quiere decir que, aunque header significa 'cabecera' no por eso el navegador va a colocar el elemento en la parte superior del documento. O en el caso de footer, no lo colocará en la parte inferior.

Es el desarrollador quien tiene que diseñar o establecer el estilo que se quiera aplicar a los elementos del documento HTML. Y esto lo hace mediante el uso de CSS.

5. Hojas de estilo CSS

CSS puede ser utilizado tanto en HTML en general (HTML5 en particular) y en XML.

El W3C (*World Wide Web Consortium*) es el organismo encargado de formular las recomendaciones para las hojas de estilo y que son un estándar para los navegadores.

En resumen, las ventajas de utilizar CSS son:

- Control centralizado del aspecto de todas las páginas del sitio web al mismo tiempo. Facilita la actualización.
- Los usuarios pueden configurar su propia hoja de estilo, ya que el navegador lo permite y así se mejora la accesibilidad.
- Una misma página puede vincularse a hojas de estilo diferentes que puede elegir el usuario en función de sus necesidades.
- El uso de hojas de estilo simplifica y reduce el tamaño del código HTML.
- Una misma hoja de estilo puede ser utilizada en diferentes sitios web.

La versión actual de CSS es la 3, CSS3 que, entre otras opciones, incluye soporte para selectores adicionales, sombras, esquinas redondeadas, múltiples fondos, animaciones y transparencias.

A. Soporte de CSS en los navegadores

No todos los navegadores soportan por igual el uso de hojas de estilo CSS. Por este motivo es muy importante para el diseñador de la página conocer si el navegador que utiliza el cliente soporta dichas hojas de estilo. Cada navegador utiliza su motor para interpretar el código HTML y CSS y mostrar las páginas correctamente. La versión de este motor es la que le dirá al diseñador si su diseño se verá como se espera o no.

La siguiente tabla muestra el soporte de CSS3 de los cinco navegadores más populares:

Navegador	Motor	CSS3
Google Chrome	WebKit	Todos los selectores, pseudoclases y muchas propiedades.
Microsoft Edge	WebView2	Todos los selectores, pseudoclases y muchas propiedades.
Firefox	Gecko	Todos los selectores, pseudoclases y muchas propiedades.
Safari	WebKit	Todos los selectores, pseudoclases y muchas propiedades.
Opera	Presto	Todos los selectores, pseudoclases y muchas propiedades.

Tabla 2.13. Soporte de CSS en los navegadores.

Para conocer el soporte del navegador a cada uno de los selectores de CSS se puede utilizar la herramienta online llamada [caniuse](#). Este software es de código abierto.

Tiene una herramienta de búsqueda que, al introducir una propiedad, parámetro o característica dice exactamente qué navegadores la soportan y qué versiones. Además, esa compatibilidad la expresa también en porcentaje, ya que podría ser parcialmente compatible.

B. Cómo incluir CSS en un documento HTML5

Las hojas de estilo se pueden añadir al código HTML de tres formas:

1. In line (estilo en línea). Utilizando el atributo `style` en los diferentes elementos HTML.
2. Estilo interno. Utilizando la etiqueta `<style>` en la sección de cabecera `<head>`.
3. Estilo externo. Utilizando un archivo CSS.

ESTILO EN LÍNEA

El estilo en línea ya se ha utilizado en apartados anteriores y solo permite aplicar un estilo a un solo elemento HTML. Ejemplo:

```
<p style="color:blue">Esta es mi página</p>
```

Mostrará el texto “Esta es mi página” en azul. En este caso, style se define como atributo del párrafo. Esta forma de aplicar estilo se utiliza cuando hay que definir pocos estilos o se necesita incluir estilos concretos que complementen el estilo global de la pagina.



EJEMPLO 1

En el archivo **UN02_AP05_ejemplo_1.html** puedes ver un ejemplo de utilización del estilo en línea.

ESTILO CSS INTERNO

El estilo interno permite aplicar un solo estilo a toda la página web. Se define en la etiqueta `<head>` de la página web utilizando la etiqueta `<style>`.



EJEMPLO 2

En el archivo **UN02_AP05_ejemplo_2.html** puedes ver un ejemplo de utilización del estilo CSS interno.

ESTILO CSS EXTERNO

En este caso, el archivo de estilo creado puede ser utilizado en diferentes páginas HTML. El archivo puede crearse con cualquier editor de texto y debe tener extensión .css.

Se puede decir que ahora sí que existe una separación real entre el código HTML y el estilo CSS.

La vinculación a una hoja de estilo externa se incluye en la sección <head>. Por ejemplo:

```
<head>
    <meta charset="utf-8" />
    <title>Mi pagina de prueba</title>
    <link rel="stylesheet" href="estilos.css" />
</head>
```

Para el ejemplo anterior, el contenido del archivo de `estilos.css` sería simplemente:

```
p {color: blue;}
```

Los atributos de link son:

- `rel` Indica el tipo de relación existente entre el archivo que se referencia con la etiqueta `link` y el documento HTML donde se carga. En el ejemplo se indica `stylesheet` porque la relación es de hoja de estilos, es decir, el archivo que se referencia es una hoja de estilos y se aplica al documento HTML.
- `href` Indica la URL absoluta o relativa del archivo externo referenciado.
- `type` indica el tipo MIME del archivo referenciado. En este caso, y al ser una hoja de estilos, será `type="text/css"`, pero HTML5 no lo necesita ya que, por defecto, una hoja de estilo es `text/css`.

C. Sintaxis CSS

Una hoja de estilo consta de uno o más estilos, y cada uno de ellos se llama regla.

Existen dos tipos de reglas:

- reglas @
- reglas, propiamente

Las reglas @ siempre comienzan con el símbolo de arroba @ junto a un identificador y acaban con el símbolo punto y coma (;). El esquema es el siguiente:



Fig. 2.12. Regla CSS arroba @.

En este ejemplo, el identificador incluido import obliga a que esta regla esté siempre al comienzo del archivo CSS, y permite cargar uno o varios archivos CSS dentro de otro. Se pueden utilizar otros identificadores como name o media, entre otros.

El archivo de estilos puede contener muchas reglas @, pero hay que tener en cuenta que cada vez que se encuentra una regla @ se ejecuta una petición al servidor y se puede ralentizar el proceso de carga de la página, así como de trabajo al servidor.

Cada regla normal puede contener varios selectores y declaraciones, y cada declaración puede contener otras declaraciones. El archivo de estilo puede contener muchas reglas de este tipo.

Resumiendo, cada regla consta de:

- **Selector**, para indicar a qué elemento/s HTML se aplica dicha regla.
- Una **llave de apertura {**.
- **Declaración**, para establecer los estilos concretos que se aplican a los elementos y que consta de:
 - **Propiedad**, para modificar el aspecto o la característica de un atributo del elemento.
 - **Valor**, para establecer el nuevo valor del atributo modificado o propiedad.
- Una **llave de cierre }**.

En el ejemplo anterior:

- `p {color: blue;}` p es el selector
- `color: blue` es la declaración; en concreto, `color` es la propiedad y `blue` es el valor.

El esquema es el siguiente:

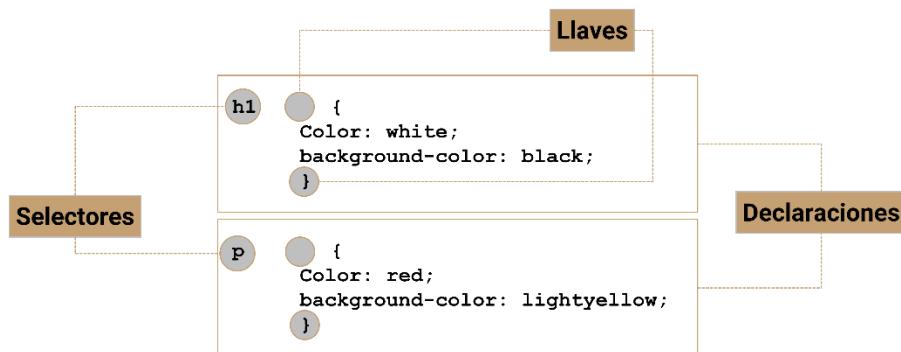


Fig. 2.13. Esquema de una regla CSS.

No se deben repetir los nombres de selectores, para evitar confusión.

Una declaración puede contener varias propiedades, que van separadas entre sí por un punto y coma (,).

Entre la propiedad y su valor se deben poner dos puntos (:).

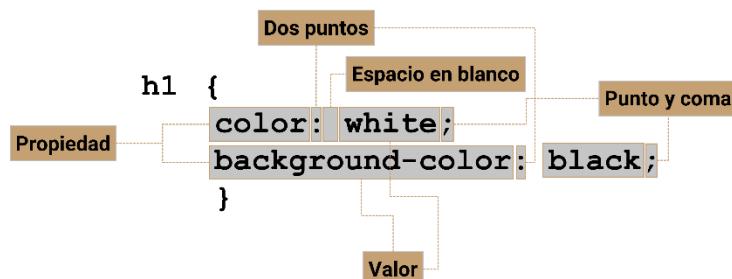


Fig. 2.24. Esquema de la declaración.



IMPORTANTE

Los signos de llaves {}, dos puntos () y el punto y coma () pertenecen a la sintaxis CSS, y se deben escribir.

Las unidades de medida que se pueden utilizar en CSS son de dos tipos:

Absolutas	<ul style="list-style-type: none"> milímetros (mm). centímetros (cm), un cm son 10 mm. puntos (pt), un punto son 1/72 in pulgadas (in), una pulgada son 2,54 cm.
Relativas	<ul style="list-style-type: none"> element (em) hace referencia al tamaño en puntos de la letra que se está utilizando. Si se utiliza una tipografía de 12 puntos, 1 em equivale a 12 puntos. pixels (px) es relativo a la resolución del monitor.

Tabla 2.14. Unidades de medida.

D. Propiedades de CSS

Se clasifican en los siguientes grupos:

D.1 Propiedades de color y fondo

La siguiente tabla enumera las principales propiedades de color y fondo usados en CSS.

Propiedad	Descripción y valores
color	Define el color del texto. Lo admiten casi todas las etiquetas de HTML. Valor: es un nombre de color o su valor RGB.
background-color	Define el color de fondo de un elemento. Valor: es un nombre de color o su valor RGB, o la palabra "transparent".
background-image	Coloca una o más imágenes de fondo para un elemento. El valor es un nombre y se referencia con path absoluto o relativo.
background-repeat	Define si ha de repetirse el fondo del documento, y si es así, cómo lo ha de hacer, en horizontal o en vertical. Valores posibles: repeat-x, repeat-y, no-repeat. Si aparecen dos valores, el primero se aplica al eje X (repetición horizontal) y el segundo al eje Y (repetición vertical).
background-position	Define la posición inicial de la imagen de fondo especificada. Puede venir en porcentaje, una medida o valores preestablecidos. Valores posibles: porcentaje, tamaño, o [top, center, bottom] [left, center, right].
background-attachment	Indica si la posición de la imagen de fondo será fija dentro de la pantalla o se desplazará con su bloque contenedor. Valores posibles: scroll o fixed.
background	Define los valores individuales del fondo en una única regla CSS. Valores posibles: background-color, background-image, background-repeat, background-attachment, background-position.

Tabla 2.15. Propiedades de color y fondo.



IMPORTANTE

Aunque es más sencillo utilizar el nombre del color, dado que no todos los nombres están estandarizados, se aconseja utilizar el valor RGB. En el siguiente enlace puedes consultar [información adicional sobre RGB y los valores de los colores más utilizados](#).

D.2 Propiedades de fuente



EJEMPLO 3

En el archivo **UN02_AP05_ejemplo_3.html** puedes ver un ejemplo de utilización de las propiedades CSS de color y fondo usando una hoja de estilos interna.

La siguiente tabla enumera las principales propiedades de formato de fuente en CSS.

Propiedad	Descripción y valores
font-size	Indica el tamaño del tipo de letra (fuente) del elemento. El valor puede ser el tamaño absoluto, relativo, distancia o porcentaje.
font-family	Indica el tipo de letra o fuente del elemento. El valor es el nombre de la familia fuente.
font-weight	Define el grosor del trazo. Valores posibles: normal, bold, bolder, lighter y valores numéricos 100, 200, 300, 400, 500, 600, 700, 800 o 900, que van del más fino al más grueso.
font-style	Permite elegir la inclinación de la fuente. Valores posibles: normal, italic, oblique. La diferencia entre italic y oblique es que la letra oblique no cambia de forma, solo se inclina, y en la letra italic algunas de ellas cambian de forma y se inclinan.
font-variant	Indica si la fuente es normal o mayúsculas pequeñas. Valores posibles: normal o small-caps.
line-height	Establece el espaciado entre líneas de un bloque de texto. Valores posibles: en unidad de longitud, porcentaje o em.
font	Permite definir simultáneamente las propiedades relacionadas con el tipo de

letra que figuran en la tabla.

El orden es importante, aunque no es necesario que aparezcan todas ellas: font-style, font-variant, font-weight, font-size/line-height, font family. Deben ir separadas por espacios. Obligatorias son font-size y font family.

Tabla 2.16. Propiedades de fuente.



EJEMPLO 4

En el archivo [UN02_AP05_ejemplo_4.html](#) puedes ver un ejemplo de utilización de las propiedades de fuente con CSS interno.



D.3 Propiedades de texto

La tabla siguiente enumera las principales propiedades que se pueden aplicar a un texto.

Propiedad	Descripción y valores
<code>text-decoration</code>	Indica si el texto es normal, subrayado, sobrerrayado o tachado. Valores posibles: none, underline, overline y line-through.
<code>text-align</code>	Establece alineación horizontal en el texto. Valores posibles: left, right, center y justify (justificado a derecha e izquierda).
<code>vertical-align</code>	Establece alineación vertical en el texto. Valores posibles: baseline, sub, super, top, text-top, middle, bottom, text-bottom o un porcentaje.
<code>text-indent</code>	Establece la tabulación o sangría del texto. Valores posibles: una longitud o porcentaje.
<code>text-transform</code>	Permite cambiar el texto a mayúsculas o minúsculas. Valores posibles: none (ninguno), capitalize (la primera letra de cada palabra en mayúsculas), uppercase (todas las letras en mayúsculas) y lowercase (todas las letras en minúsculas).
<code>word-spacing</code>	Permite establecer un espaciado entre las palabras de un texto. Valores posibles: tamaño positivo o negativo, se establecen como una longitud y se añaden al espaciado normal.
<code>letter-spacing</code>	Permite establecer un espaciado entre las letras de un texto. Valores posibles: tamaño positivo o negativo, se establecen como una longitud y se añaden al espaciado normal.

Tabla 2.17. Propiedades de texto.



EJEMPLO 4-BIS

Utilizaremos el mismo archivo del ejemplo anterior **UN02_AP05_ejemplo_4.html**, modificándolo para utilizar propiedades de texto con CSS externo.

Si en lugar de definir el estilo en el propio HTML lo hacemos utilizando un archivo externo CSS, dicho archivo (al que podríamos nombrar como **estilo_4.css**) contendría el siguiente código:

```
h1 {  
color:blue;  
text-decoration:underline; }  
p{  
color:black;  
font-size:12px;  
text-align:center;  
text-transform:uppercase;  
word-spacing:20px; }  
ul{  
text-align:left;  
text-indent:20px;  
font-weight:bold;  
font-style:italic;  
font-size:10px;  
color:red; }
```

Y en el archivo HTML del ejemplo 4 tendremos que eliminar esta parte del código e incluir la siguiente línea:

```
<link rel="stylesheet" href="estilo_4.css"/>
```

Tanto el archivo HTML como la hoja de estilo deberán estar en el mismo directorio. Si no es así hay que indicar el path o camino hasta su ubicación.

D.4 Propiedades de caja

El diseño de las hojas CSS se basan fundamentalmente en el modelo de caja. Cada caja ocupa un espacio en la página web, y todo elemento de la página web está contenido en una caja. La figura siguiente muestra la estructura de la caja:

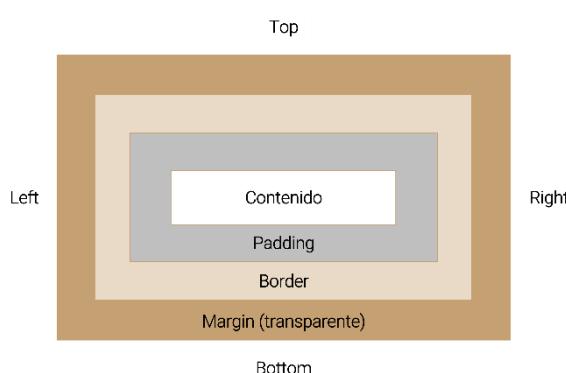


Fig. 2.15. Estructura de una caja.

El detalle de todas las propiedades de las cajas y sus valores está disponible en el archivo **UD2_Propiedades_cajas_CSS.pdf**, en la sección **Material de trabajo>Descarga de documentos**.

En el siguiente enlace se explican en detalle [los márgenes de la caja de texto](#).



IMPORTANTE

El margen exterior (margin) y el margen interior (padding) son transparentes por defecto.



EJEMPLO 5

En el archivo **UN02_AP05_ejemplo_5.html** puedes ver un ejemplo de creación de una caja con las siguientes características: ancho de 450 px, altura de 180 px, color de fondo rojo, borde de 8 px con puntos, color verde, margen de 50 px, relleno de 40 px y que contenga una frase en medio que diga: "Mi primera CAJA" con el tipo de letra sans serif, tamaño 28 px, color azul y en mayúsculas.

La hoja de estilo está en el archivo **UN02_AP05_ejemplo_5.css**.

D.5 Propiedades de clasificación

Las propiedades de clasificación influyen en la forma de representar elementos de clasificación HTML, como son las listas.

La siguiente tabla describe las principales propiedades de clasificación y sus valores posibles.

Propiedad	Descripción y valores
display	Indica si el elemento es de bloque, línea, lista o ninguno de ellos. Valores posibles: block, inline, list-item o none.
white-space	Especifica el modo en que se tratarán los espacios en blanco. Valores posibles: pre (se representan todos los espacios), nowrap (la línea no se cortará al llegar al extremo de la página.), normal (si hay varios espacios seguidos, se representarán como uno solo).
list-style-type	Selecciona la marca o viñeta que acompaña a un elemento de lista. Se aplica a los elementos que tengan definido su atributo display como list-item. Valores posibles: disc, circle, square, decimal, lower-roman, upper-roman, lower-alpha, upper-alpha, none.

list-style-image	<p>Selecciona la imagen que se usará como marca de elemento de lista.</p> <p>Se aplica a los elementos que tengan definido su atributo display como list-item.</p> <p>El valor que toma es el path del archivo imagen .</p>
list-style-position	<p>Indica si la marca de lista aparece dentro del cuadro al que pertenece o aparece fuera del cuadro.</p> <p>Se aplica a los elementos que tengan definido su atributo display como list-item.</p> <p>Valores posibles: outside o inside.</p>
list-style	<p>Establece de una única vez todas las características de una lista. Hay que seguir el orden siguiente: list-style-type, list-style-position y list-style-image.</p>

Tabla 18. Propiedades de clasificación.



EJEMPLO 6

En el archivo **UN02_AP05_ejemplo_6.zip** encontrarás un ejemplo de propiedades de clasificación utilizando CSS externo. La hoja de estilo está en el archivo **UN02_AP05_ejemplo_6.css**

E. CSS de posicionamiento

Los navegadores crean una caja para representar cada elemento de la página HTML y la posicionan de forma automática en ella, pero desde CSS se puede modificar dicha posición.

Existen tres formas de posicionar una caja:

- Posicionamiento normal o estático. Es el utilizado por defecto en los navegadores. Los elementos se muestran en la pantalla en el mismo orden en que se encuentran en el código fuente. Hay que conocer si el elemento es de bloque o en línea, sus propiedades `width`, `height` y su contenido.
- Posicionamiento flotante. Los elementos se muestran en la pantalla en el mismo orden en que se encuentran en el código fuente, pero pueden estar desplazados a derecha o izquierda de la posición en la que inicialmente estaba. Utiliza la propiedad `float`.
- Posicionamiento absoluto. Se utiliza para establecer de forma absoluta o exacta la posición en la que se muestra la caja de un elemento. Los elementos se pueden mostrar en pantalla en orden diferente al del código fuente. Utiliza la propiedad `position`.

E.1 Posicionamiento absoluto

El posicionamiento absoluto de un elemento en la caja se controla con la propiedad `position`, que puede tomar los valores:

- **relative**: posicionamiento absoluto relativo. Es como el posicionamiento normal, pero luego la caja se desplaza y se controla con las propiedades `top`, `right`, `bottom` y `left`.
- **fixed**: el posicionamiento absoluto es fijo cuando la caja es posicionada de forma fija, es decir, inamovible dentro de la ventana del navegador. En ese caso, las cajas no modifican su posición, aunque el usuario mueva la página que muestra el navegador.
- **absolute**: el posicionamiento absoluto es absoluto cuando la caja se sitúa en la posición exacta que se le da. En ese caso, las cajas sí que pueden modificar su posición si el usuario mueve la página que muestra el navegador.

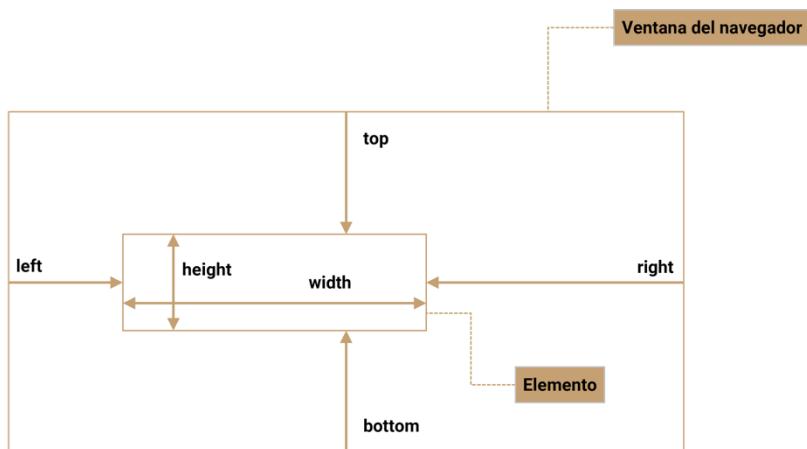


Fig. 2.36. Análisis de una caja.

En el posicionamiento absoluto, en cualquiera de las tres modalidades, las cajas se pueden superponer. Con la propiedad `z-index` se puede controlar qué caja va encima o debajo.

El valor de la propiedad `z-index` es un número entero ($>0, <0, 0$). Valor cero equivale a no tener `z-index`. Cajas con valores mayores de `z-index` se ponen encima de las que tengan valores menores.

E.2 Otras propiedades

La siguiente tabla describe un conjunto de propiedades de las cajas.

Propiedad	Descripción y valores
height	Permite establecer la altura de un elemento. Los valores que puede tomar son: auto o un tamaño o porcentaje.
width	Permite establecer la anchura de un elemento. Los valores que puede tomar son: auto o un tamaño o porcentaje.
visibility	Indica si el elemento sobre el que actúa será visible o no. Valores posibles: <ul style="list-style-type: none"> hidden (esconde un elemento, pero deja vacío el espacio donde debería aparecer) collapse (esconde filas o columnas de una tabla).
left	Indica la posición del lado izquierdo del elemento. Los valores que puede tomar son: auto o un tamaño o porcentaje.
top	Indica la posición del lado superior del elemento. Los valores que puede tomar son: auto o un tamaño o porcentaje.
overflow	Indica si el elemento será visible o no en caso de superar los límites del contenedor. Los valores que pueden tomar son: <ul style="list-style-type: none"> visible (se verá aunque sobrepase). hidden (se recorta si no cabe). scroll (se recorta pero muestra barras de desplazamiento). auto (depende del navegador que aparezcan barras o no).

Tabla 2.19. Otras propiedades de las cajas.



EJEMPLO 7

En el archivo **UN02_AP05_ejemplo_7.html** encontrarás un ejemplo de propiedades de posicionamiento utilizando CSS interno.

Ejemplo de propiedades CSS de posicionamiento

La imagen tiene un valor del atributo z-index menor que el del encabezado de primer nivel y que el del párrafo por ello la Imagen aparece por detrás del texto.

Fig. 2.17. Ejecución del ejemplo 7.

F. Definición y uso de clases

Cuando se quiere aplicar estilos a un solo elemento de la página HTML se puede utilizar el atributo `class` de HTML sobre ese elemento y así enlazar directamente con la regla CSS que se le debe aplicar.

La clase define un estilo que, en principio, no está asociado a ninguna etiqueta HTML, pero que es susceptible de asociar a etiquetas concretas.

El atributo `class` se define mediante una lista de elementos separados por espacio, y uno de esos elementos debe coincidir exactamente con el nombre de clase dado en el selector.

En la declaración del selector CSS se incluye el nombre de clase precedido de un punto (.). Así, el selector de identificador aplica el estilo al elemento HTML cuyo identificador sea ese nombre.

Recordar que los selectores se utilizan para seleccionar los elementos a los que se van a aplicar las propiedades declaradas en el código CSS.

Por ejemplo, se define así una clase para el color rojo:

```
.clase_rojo{color:red}
```

Si se quiere que el título “h1” se muestre en rojo, se asocia en el archivo HTML el elemento `h1` a dicha clase de la forma:

```
<h1 class="clase_rojo">Cabecera h1 en rojo</h1>
```

Si solo se quiere utilizar esta clase con el elemento HTML de párrafo, en la declaración de la clase hay que añadir el selector `p`, y queda:

```
p.clase_rojo{color:red}
```



IMPORTANTE

Las clases se pueden utilizar varias veces dentro del mismo documento. Si solo se utiliza una vez se puede emplear el atributo `id`.



EJEMPLO 8

En el archivo `UN02_AP05_ejemplo_8.html` encontrarás un ejemplo del uso de clases utilizando CSS interno.

G. Frameworks para CSS

Los frameworks CSS son herramientas de diseño web que facilitan la tarea de crear y editar hojas de estilo en cascada. Facilitan el trabajo del desarrollador/diseñador y le proporcionan una colección de hojas de estilo que se pueden adaptar a sus necesidades.

De ellas, las más importantes o conocidas son las siguientes:

- [Bootstrap](#).
- [Bulma](#).
- [Foundation](#).
- [Materialize CSS](#).

Muchas de ellas utilizan un diseño responsive, es decir, adaptan el diseño de la página al dispositivo que la visualiza.

6. Validación de código HTML y CSS. Herramientas

Existe software descargable o servicios online que facilitan la tarea de validación haciéndola más eficiente, rápida y fiable.

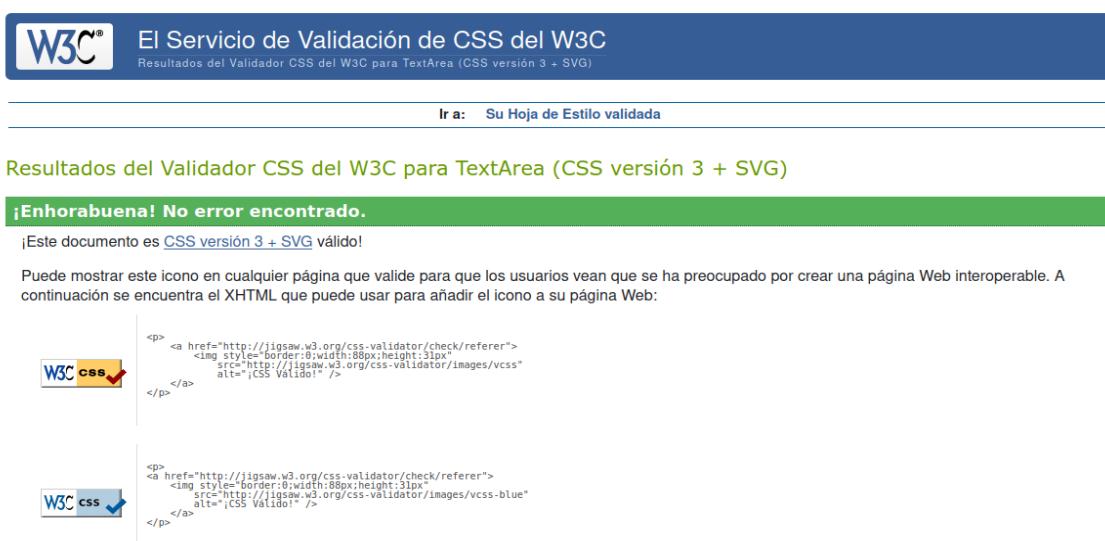
Un servicio de validación online para HTML es el que proporciona el [W3C](#). La página dispone de tres pestañas en función de si se da la URL de la página a validar, se sube el archivo o, si la página aun no es pública, se copia y se pega el código.

El resultado puede ser de dos tipos: anotaciones (verde) y advertencias (amarillo) o errores propiamente (rojo).

Una vez se han realizado los ajustes necesarios hay que comprobar de nuevo.

Para validar hojas de estilo, estando en el [sitio de W3C](#) hay que seleccionar la opción para hojas CSS.

Otra opción en castellano es el enlace de [W3C siguiente](#). El aspecto de la página es el siguiente:



The screenshot shows the W3C CSS Validator interface. At the top, it says "El Servicio de Validación de CSS del W3C" and "Resultados del Validator CSS del W3C para TextArea (CSS versión 3 + SVG)". Below that, a green bar says "¡Enhorabuena! No error encontrado." A note below it says "Este documento es CSS versión 3 + SVG válido!". The main content area displays two examples of valid CSS code snippets, each preceded by a "W3C CSS" logo with a checkmark. The code snippets are:

```
<p> <a href="http://jigsaw.w3.org/css-validator/check/referer">
    
</p>
```



```
<p> <a href="http://jigsaw.w3.org/css-validator/check/referer">
    
</p>
```

Fig. 2.18. Validación de CSS del W3C.

Desde Visual Studio Code también se puede realizar la validación mediante la extensión HTMLHint. Desde la pestaña “Extensions” de Visual Studio Code hay que buscar esta extensión e instalarla.

Una vez instalada, para ver los posibles errores detectados ir al menú: Ver → Problemas. De cualquier forma, la validación que hace Visual Studio Code no es completa, pero sí que puede ayudar al desarrollador a detectar algunos problemas.

Existen muchos otros validadores que de forma gratuita y online permiten validar el código HTML y/o CSS, como por ejemplo:

- <https://www.freeformatter.com/html-validator.html>
- <https://html5.validator.nu/>
- <https://www.htmlhelp.com/tools/csscheck/>

Otros tipos de validadores

- Validador HTML del W3C para móviles: <http://validator.w3.org/mobile/>

- Existe también un validador o comprobador de enlaces del W3C. Esta herramienta comprueba todos los enlaces que contiene la/s página/s, indicando aquellos que no funcionan. El enlace es: <http://validator.w3.org/checklink/>.
- Por último, existe también un validador de feeds del W3C que permite validar archivos de fuentes RSS. Disponible en <http://validator.w3.org/feed/>. Se verá en otra unidad.

Herramientas

Existe una serie de herramientas que facilitan al desarrollador la tarea de generar y optimizar código HTML y estilo CSS. Algunas de estas herramientas vienen incluidas en el propio navegador y ayudan a depurar el código. Es el caso de las devtools, que para el navegador Firefox están disponibles desde el menú de la aplicación



Fig. 2.21

Desplegado este menú de la aplicación, ir a Desarrolladores web → Herramientas para desarrolladores. En la parte inferior de la ventana de Firefox se muestra un menú con una serie de opciones, como el inspector de estilos y DOM, una consola, un depurador, un editor de estilos, medidor de rendimientos, ocupación de memoria, etcétera. También se puede acceder con el atajo de teclado CTRL + May + I.

En otros navegadores, el procedimiento de acceso a estas herramientas puede ser otro. La imagen siguiente muestra su aspecto para el código de la página visitada:



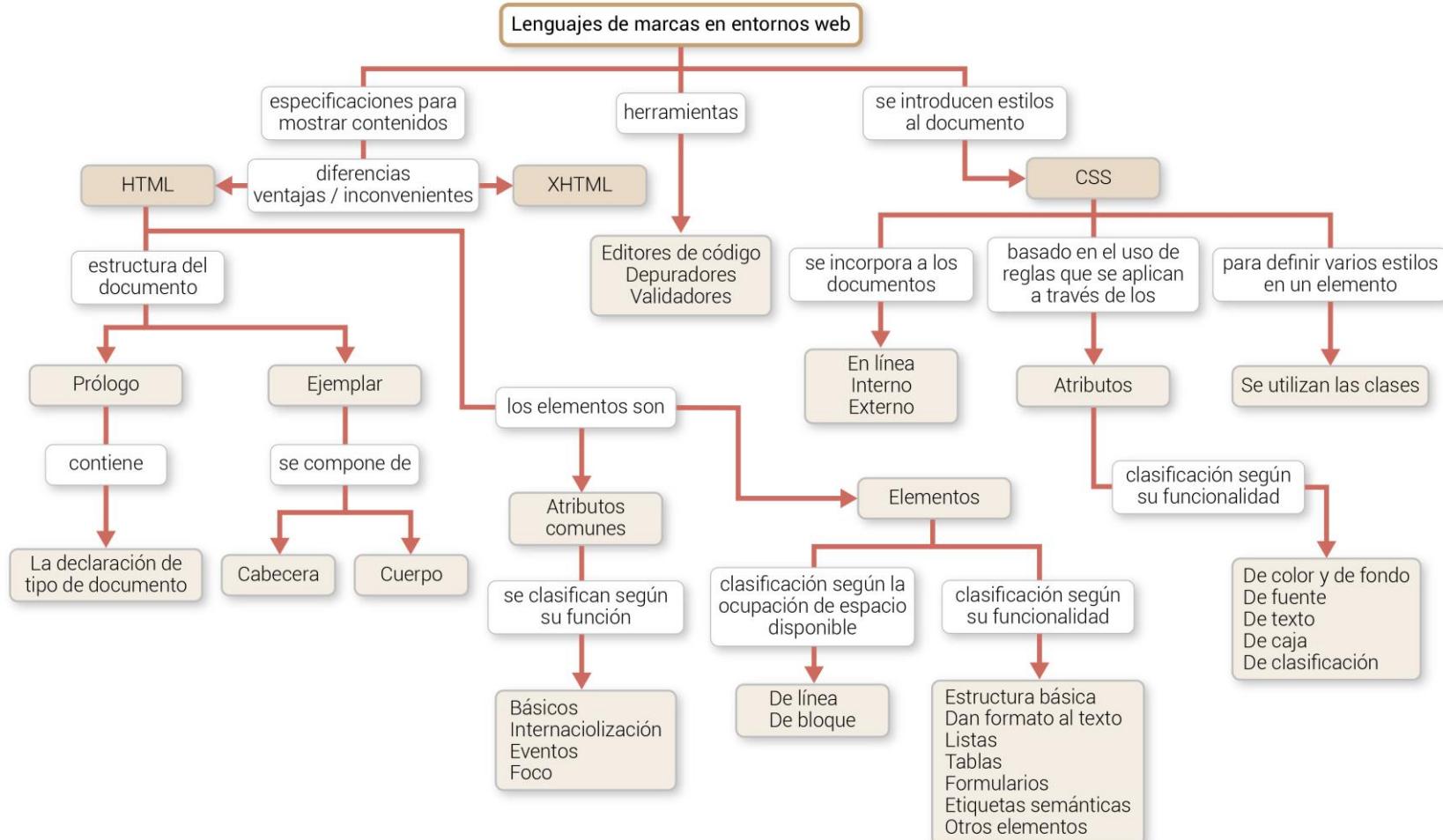
Fig. 2.19. Consola del navegador Firefox.

Existe otro tipo de herramientas más especializadas para ejecutar el código y comprobar si funciona como se espera en diferentes navegadores. Más información sobre estas herramientas en el siguiente [enlace](#).

Este punto de la unidad 2 es también válido para la unidad en la que se describe JavaScript.



UTILIZACIÓN DE LENGUAJES DE MARCAS EN ENTORNOS WEB



2

UNIDAD



Atributos de cajas CSS

La siguiente tabla recoge las principales propiedades de las cajas en CSS:

Atributo	Descripción
margin	Establece todos los márgenes de una vez. Hay que seguir el orden: superior, derecho, inferior e izquierdo.
margin-left	Indica el tamaño del margen izquierdo. Valores posibles: una longitud en unidades CSS o un porcentaje.
margin-right	Indica el tamaño del margen derecho. Valores posibles: una longitud en unidades CSS o un porcentaje.
margin-top	Indica el tamaño del margen superior. Valores posibles: una longitud en unidades CSS o un porcentaje.
margin-bottom	Indica el tamaño del margen inferior. Valores posibles: una longitud en unidades CSS o un porcentaje.
padding	Establece el espacio entre los bordes y el contenido de una sola vez. Hay que seguir el orden: superior, derecho, inferior e izquierdo.
padding-left	Indica el espacio izquierdo entre el borde y el contenido. Valores posibles: una longitud en unidades CSS o un porcentaje.
padding-right	Indica el espacio derecho entre el borde y el contenido. Valores posibles: una longitud en unidades CSS o un porcentaje.
padding-top	Indica el espacio superior entre el borde y el contenido. Valores posibles: una longitud en unidades CSS o un porcentaje.
padding-bottom	Indica el espacio inferior entre el borde y el contenido. Valores posibles: una longitud en unidades CSS o un porcentaje.
border-left-color	Establece el color del borde izquierdo del elemento. Valores posibles: un color RGB o el nombre del color.
border-right-color	Establece el color del borde derecho del elemento. Valores posibles: un color RGB o el nombre del color.
border-top-color	Establece el color del borde superior del elemento. Valores posibles: un color RGB o el nombre del color.

border-bottom-color	Establece el color del borde inferior del elemento. Valores posibles: un color RGB o el nombre del color.
border-color	Establece el color de los bordes del elemento de una sola vez. Hay que seguir el orden: superior, derecho, inferior e izquierdo. Valores posibles: un color RGB o el nombre del color.
border-style	Establece el estilo del borde. Valores posibles: <code>none</code> (ningún borde), <code>dotted</code> (punteado, pero no funciona siempre), <code>solid</code> (sólido), <code>double</code> (doble borde). Los valores <code>groove</code> , <code>ridge</code> , <code>inset</code> y <code>outsets</code> son bordes con varios efectos 3D.
border-left-width	Grosor del borde izquierdo. Valores posibles: <code>thin</code> , <code>medium</code> , <code>thick</code> o un tamaño.
border-right-width	Grosor del borde derecho. Valores posibles: <code>thin</code> , <code>medium</code> , <code>thick</code> o un tamaño.
border-top-width	Grosor del borde superior. Valores posibles: <code>thin</code> , <code>medium</code> , <code>thick</code> o un tamaño.
border-bottom-width	Grosor del borde inferior. Valores posibles: <code>thin</code> , <code>medium</code> , <code>thick</code> o un tamaño.
border-width	Establece el tamaño de los bordes del elemento de una sola vez. Hay que seguir el orden: superior, derecho, inferior, izquierdo.
width	Establece el ancho del contenido del elemento. Valores posibles: un porcentaje o un tamaño.
height	Establece la altura del contenido del elemento. Valores posibles: un porcentaje o un tamaño.
float	Sirve para alinear un elemento a la izquierda o la derecha haciendo que el texto se agrupe alrededor de dicho elemento. Valores posibles: <code>none</code> , <code>left</code> o <code>right</code> .
clear	Establece si un elemento tiene a su altura imágenes u otros elementos alineados a la derecha o la izquierda. Valores posibles: <code>none</code> , <code>left</code> , <code>right</code> o <code>both</code> .



Atributos de <input>

La etiqueta <input> permite añadir campos al formulario y debe tener siempre definidos sus atributos:

Atributo	Descripción
name=""	Nombre que se le da al campo. Este nombre no es visible en el navegador, solo se utiliza para diferenciar los campos al enviar la información al servidor. Hace la función de nombre de variable a la que se asigna el valor del campo.
type=""	Tipo de campo a utilizar. Puede ser de tipo: text, password, checkbox, radio, file, hidden, submit, reset.
pattern	Delimita mediante expresiones regulares el contenido insertado por el usuario en los inputs del formulario. <pre><input type="text" name="nombre" pattern="[A-Za-z]{3-10}></pre>
list	List, combinado con elemento <datalist>, permite al usuario introducir contenido en un campo de texto. Las opciones dadas en <datalist> aparecen como lista desplegable. La lista debe estar dentro de un elemento <datalist> y su identificador debe ser igual al del atributo list. <pre><input id="direccion" type="text" list="tipo"> <datalist id="tipo"> <option label="Calle" value="C/"> <option label="Avenida" value="Av/"> <option label="Plaza" value="Plz/"> </datalist></pre>
autofocus	Si aparece el atributo autofocus en una etiqueta input, el cursor se posiciona automáticamente en este campo cuando la página se carga. Se puede indicar el valor del atributo como autofocus o no poner nada.
required	Si aparece este atributo, el navegador no permite el envío del formulario si el campo en concreto está vacío. Puede ser utilizado en <textarea> y en muchos elementos <input>, excepto en los tipos hidden, image o botones como submit. <pre><label for="nombre">Nombre</label> <input id="nombre" name="nombre" type="text" required/></pre>
autocomplete	Admite dos valores: on y off. Cuando está en on está activado el servicio de autocompletar algunos campos del formulario con valores que se han introducido en otro momento. Si está en off, dicha funcionalidad se encuentra desactivada. Hay que tener en cuenta que no siempre es útil y/o conveniente tenerla activada. <pre><input type="text" name="campoprivado" autocomplete="off" /></pre>

placeholder	<p>Se utiliza a modo de ayuda al usuario para que sepa qué tiene que introducir en este campo del formulario. Cuando el usuario pone el foco en este campo, el texto del placeholder desaparece. Si el usuario no introduce nada y el foco se va a otro campo el texto del placeholder, vuelve a aparecer.</p> <pre><label for="nombre">Nombre</label> <input id="nombre" name="nombre" type="text" placeholder="Escribe tu nombre y apellidos"/></pre>
-------------	---

En el siguiente enlace están disponibles todos los atributos del elemento `form`:

https://developer.mozilla.org/es/docs/orphaned/Learn/HTML/Forms/HTML5_updates

Los campos `<input>`, según su valor `type`, pueden ser:

- `type="text"` campo de tipo texto de una línea. Sus atributos son:
 - `maxlength=""` delimita el número máximo de caracteres.
 - `size=""` delimita el número de caracteres a mostrar en pantalla. No limita la longitud del texto que se puede introducir, sino el tamaño visible del campo.
 - `value=""` indica el valor inicial del campo.

Ejemplo:

```
<input name="usuario" type="text" maxlength="24"/>
```

- `type="password"`, campo igual al de tipo `"text"`, pero que oculta el texto introducido cambiando las letras por asteriscos (*) o puntos (.). Los atributos son los mismos que para `"text"`.
- `type="checkbox"`, campo que muestra una casilla cuadrada que permite marcar varias opciones a la vez de una lista. Si varias casillas pertenecen al mismo grupo deben tener el mismo nombre en el atributo `"name"`. El texto que se quiera que aparezca a continuación de la casilla del `"checkbox"` hay que escribirlo después de cerrar la etiqueta `input`. Sus atributos son:
 - `value=""`, define el valor que se envía si la casilla está marcada.

Ejemplos:

```
<input type="checkbox" name="opcion1" value="yogur"/>Yogur<br/>
<input type="checkbox" name="opcion1" value="fruta"
checked/>Fruta<br/>
<input type="checkbox" name="opcion1" value="flan"/>Flan<br/>
```

- `type="radio"`, el campo se elige al marcar una casilla circular de entre varias opciones. El resto de casillas de ese grupo de desmarcan automáticamente. Si varias casillas pertenecen al mismo grupo deben tener el mismo nombre en `"name"`. Para el `type="radio"` además hay que indicar:
 - `value=""`, define el valor que se envía si la casilla está marcada.

- `checked`, atributo opcional, y la casilla aparece marcada por defecto. Solo se podrá usar para una casilla. No necesita darle ningún valor.

Ejemplo (paralelo al anterior, solo cambia el `type`):

```
<input type="radio" name="grupo1" value="yogur"/>Yogur<br/>
<input type="radio" name="grupo1" value="fruta" checked/>Fruta<br/>
<input type="radio" name="grupo1" value="flan"/>Flan<br/>
```

- `type="file"`, permite al usuario subir archivos. Se necesita un programa que gestione esos archivos en el servidor mediante un lenguaje diferente al HTML. El único atributo opcional a utilizar es `size=""`, que indica la anchura visual del campo.

Ejemplo:

```
<input type="file" name="archivo" size="40"/>
```

- `type="hidden"`, valor que permanece oculto y no puede modificarse. Se utiliza para enviar algún dato adicional necesario para procesar el formulario. Para indicar el valor de este campo se utiliza el atributo `value="valor"`.
- `type="submit"`, es el botón de "Enviar". Al pulsar este botón, la información de todos los campos se envía realizando la acción indicada en `<form>`. Para decir el texto que aparece en el botón se utiliza el atributo `value="texto"`.
- `type="reset"`, si se pulsa este botón se borra el contenido de todos los campos del formulario. Para decir el texto que aparece en el botón se utiliza el atributo `value="texto"`.
- `type="email"`, comprueba que la dirección de correo dada tiene formato válido antes del envío del formulario. Si se quiere especificar una lista de correos válidos se utiliza el atributo múltiple. Los saltos de línea se eliminan automáticamente.
- `type="url"`, comprueba que la URL dada es válida (puede no ser una dirección web) antes del envío del formulario.
- `type="date"`, el navegador muestra la interfaz de usuario para la fecha y los datos enviados al servidor cumplen la norma ISO-8601, independientemente del formato mostrado.
- `type="time"`, el navegador muestra la interfaz de usuario para la hora y permite utilizar el formato de 24 h.
- `type="datetime"`, es una combinación de los tipos `date` y `time`, y valida tanto la fecha como la hora introducida.
- `type="month"`, permite elegir un mes concreto. La representación interna del mes es un valor entre 1 y 12, y es el navegador el que muestra al usuario el nombre del mes.
- `type="week"`, permite elegir una semana concreta del año. La representación interna del mes es un valor entre 1 y 53, y es el navegador el que define la manera de mostrarlo al usuario. La representación interna de la semana 7, por ejemplo, es la siguiente: 2021-W07.
- `type="number"`, para aceptar datos de tipo numérico. Se puede combinar con los atributos `min`, `max` y `step`.

- `type="range"`, para mostrar un control deslizante en el navegador. Se puede combinar con los atributos `min`, `max` y `step`.
- `type="tel"`, acepta un número de teléfono. No hay validación de ningún tipo. Se pueden utilizar los atributos `pattern` y `maxlength` para restringir los valores introducidos en el campo.
- `type="search"`, acepta un término de búsqueda. Es similar a un campo de texto normal, pero proporciona un histórico de términos introducidos.
- `type="color"`, para seleccionar un color de una paleta de colores mostrada por el navegador.

La lista completa de atributos disponibles para `<input>` se puede encontrar en <https://developer.mozilla.org/es/docs/Web/HTML/Element/input#attr-type>.

2

UNIDAD



Atributos de eventos

1. Atributos de eventos

Los atributos de eventos se agrupan en función de su ámbito de utilización:

A. Pueden ser utilizados por todos los elementos

Atributos	Descripción
onclick	Ejecuta la acción cuando se realiza un clic sobre el elemento.
ondblclick	Ejecuta la acción cuando se realiza un doble clic sobre el elemento.
onmousedown	Ejecuta la acción cuando se detecta el botón pulsado del ratón.
onmouseup	Ejecuta la acción cuando se detecta que se ha soltado el botón del ratón.
onmousemove	Ejecuta la acción cuando se detecta el movimiento del ratón sobre el elemento.
onmouseout	Ejecuta la acción cuando el ratón abandona el elemento.
onmouseover	Ejecuta la acción cuando se detecta que el ratón se sitúa sobre el elemento.

B . Pueden ser utilizados en el elemento <body>

Atributos	Descripción
onload	Ejecuta la acción cuando se carga el documento.

onunload	Ejecuta la acción cuando se abandona el documento.
onresize	Ejecuta la acción cuando se ha modificado el tamaño de la ventana del navegador.

C . Pueden ser utilizados por elementos de formulario y en <body>

Atributos	Descripción
onkeydown	Ejecuta la acción cuando se detecta que la tecla está pulsada.
onkeyup	Ejecuta la acción cuando se detecta que se ha soltado la tecla pulsada.
onkeypress	Ejecuta la acción cuando se pulsa una tecla.

D. Pueden ser utilizados por varios

Atributos	Descripción	Elementos que pueden usarlo
onblur	Ejecuta la acción cuando el elemento pierde el foco, bien sea a través del ratón o por navegación tabulada.	<button>,<input>,<label>,<select>,<textarea>,<body>
onfocus	Ejecuta la acción cuando el elemento obtiene el foco, bien sea a través del ratón o por navegación tabulada.	<button>,<input>,<label>,<select>,<textarea>,<body>
onchange	Ejecuta la acción cuando el valor de un control ha sido modificado.	<input>,<select>,<textarea>

onreset	Ejecuta la acción cuando el formulario es restablecido a sus valores por defecto.	<form>
onselect	Ejecuta la acción cuando un usuario selecciona texto en un campo de texto.	<input>,<textarea>
onsubmit	Ejecuta la acción cuando el formulario es enviado	<form>

2. Atributos de foco

Se llama **foco** (focus) cuando un control o elemento del documento ha sido seleccionado.

Cuando ese elemento deja de estar seleccionado, " pierde el foco" y es el nuevo elemento seleccionado el que se dice que tiene "el foco".

Atributo	Descripción
accesskey="letra"	Establece una tecla de acceso rápido a un elemento HTML.
tabindex="numero"	Establece la posición del elemento en el orden de tabulación de la página (valor entre 0 y 32.767).
onfocus,onblur	Controlan los eventos JavaScript que se ejecutan cuando el elemento obtiene o pierde el foco.