Tecnología web

Parte 1: HTML y CSS para páginas docentes interactivas

Pedro Corcuera
1-1-2023

ÍNDICE

LISTA DE FIGURAS

1 INTRODUCCIÓN

Antes de entrar en los detalles del desarrollo del contenido para la Web, describiremos los términos y tecnología que subyace en Internet. La comprensión básica de cómo funciona la web es importante para entender los mensajes que producen los navegadores.

1.1 INTERNET

Internet (Web) es un conjunto de escala mundial de redes de ordenadores interconectados que permite la ejecución de diversos tipos de aplicaciones para el intercambio de información.

1.1.1 Historia

Internet no nació en un momento determinado, sino que evolucionó sobre una serie de avances tecnológicos. El predecesor del Internet moderno fue una red del departamento de defensa de los Estados Unidos llamada ARPANET (Advanced Research Projects Agency Network) que se utilizó a finales de los años 60 y 70 para el envío de correos electrónicos y transferencia de ficheros. ARPANET tenía características únicas que serían importantes en el desarrollo de Internet. Algunas de esas características eran:

- enviar datos entre computadoras en fragmentos de paquete,
- habilidad de las subredes de valerse por sí mismas,
- habilidad de las computadoras de unirse y abandonar dinámicamente la red,
- construido sobre estándares abiertos, de tal forma que cualquiera podía crear un nuevo dispositivo o programa para conectarse,
- carencia de un control centralizado sobre la red.

Un aspecto clave de una red es que los ordenadores puedan enviar y recibir datos en un formato estándar para ser capaces de entenderse unos con otros. Tal estándar para el flujo de información entre ordenadores es llamado *protocolo*. Al respecto el protocolo estándar que soporta las comunicaciones en

Internet es TCP/IP (Transmission Control Protocol/Internet Protocol) desarrollado en la década de los 70. ARPANET continuó creciendo durante la década de los 70 y empezó a usar varios de las tecnologías y protocolos modernos de Internet a mediados de la década de los 80. La apertura de Internet para fines comerciales fue en 1988 cuando se permitió a varias compañías el envío de correo electrónico a través de él.

El crecimiento de la popularidad de Internet vino en 1991 cuando desde el CERN (Conseil Européen pour la Recherche Nucléaire) propuso un nuevo proyecto y un conjunto de estándares llamados de forma colectiva World Wide Web (WWW), tomando como base el desarrollo realizado por Tim Berners-Lee en 1989.

El siguiente enlace (inglés) contiene una historia detallada de Internet.

1.1.2 Organizaciones

Como se ha mencionado, la gestión de Internet no es llevada a cabo por una organización en concreto, siendo esta una de las características más interesantes: la ausencia de un control centralizado. Sin embargo, es deseable algún nivel de estandarización y consistencia a través de Internet, por lo que varios grupos organizacionales y de estandarización se han formado para ayudar en este proceso.

Las organizaciones más importantes son:

- La Internet Engineering Task Force (IETF) que crea especificaciones para los estándares de protocolo, gobernando la forma en que se intercambia la información en Internet. El documento de estandarización IETF se le conoce como un Request for Comments o RFC.
- La Internet Corporation for Assigned Names and Numbers (ICANN) controla los nombres de dominio en la web y las direcciones IP.
- La World Wide Web Consortium (W3C) proporciona recomendaciones para los estándares web para los lenguajes de programación usados en la web y cómo debe mostrarse las páginas web en los navegadores. La W3C está dirigida por Tim Berners-Lee, el creador de la Web.

1.1.3 Tecnologías

La idea clave detrás de Internet es que trabaja en capas. Una aplicación típica de Internet como un navegador we envía datos que van a través de capas de hardware y software tales como:

- una capa física de dispositivos como cables Ethernet, líneas de fibras ópticas o módems.
- una capa de datos que permite que esos dispositivos hablen directamente unos con otros, tales como protocolo Ethernet o protocolos Wireless (wifi).
- una capa de red representado por el protocolo internet (IP)
- una capa de transporte que añade características a la capa de red como TCP
- una capa de aplicación que implementa la comunicación específica para este tipo de programa, como HTTP (web), POP3/IMAP (email), FTP, etc.

Las capas de red y transporte merecen una descripción con más detalle:

Protocolo Internet (IP)

El Protocolo Internet (Internet Protocol) o IP es el sistema de comunicación subyacente de todos los datos enviados a través de Internet. Es un protocolo simple para un ordenador enviar paquetes de datos a otro ordenador.

Un aspecto importante de cualquier protocolo es la cuestión del direccionamiento. Esto es, cómo se especifica a quien se desea enviar el mensaje. En IPv4 se trata este asunto dando a cada ordenador en Internet un único número de identificación compuesto por 32 bits llamado *dirección IP*. Para que sea más manejable, los 32 bits se dividen en 4 grupos de 8 bits, por lo que las cuatro partes de una dirección IP es un número entre 0 y 255, ejemplo 185.73.172.162. Teniendo en cuenta la limitación de direcciones que ofrece IPv4 y por razones técnicas relativas a seguridad, se ha generado IPv6 que utiliza 128 bits y para su especificación se utilizan 8 grupos de 4 dígitos hexadecimales separados por dos puntos, ejemplo: 2001:0db8:85a3:0000:0000:8a2e:0370:7334.

Protocolo de control de transmisión (TCP)

El protocolo de control de transmisión (Transmission Control Protocol) o TCP es otro estándar de comunicación implementado sobre IP. Cuando un ordenador envía un paquete a otro ordenador usando IP, una vez que el paquete llega, el conjunto de reglas software que interpretan y gestionan sus datos se llama TCP. TCP garantiza la entrega de la información de forma fiable, sin corrupción y ordenada. TCP añade la capacidad de que varios programas y servicios compartan la misma conexión física y de Internet, mediante la asociación del programa/servicio con un entero único llamado *puerto*. Se han asignado puertos estándar a determinados servicios comunes de Internet mostrados en la Tabla 1.

Tabla 1 Puertos estándar TCP

Puerto TCP	Servicio
21	Transferencia de ficheros (FTP)
22	Shell seguro (SSH)
25, 110	Email (SMTP, POP3)
80	Web (HTTP)
443	Web seguro (HTTPS)
993, 995	Email seguro

1.2 LA WORLD WIDE WEB (WWW)

La World Wide Web (WWW) es un conjunto mundial de documentos formateados en un lenguaje llamado Lenguaje de Marcas de Hipertexto (Hyper Text Markup Language, HTML). La web está construida sobre TCP/IP y añade varios conceptos, lenguajes, y tecnologías específicas para recuperar y visualizar contenido enriquecido y multimedia. Un aspecto clave de la web es que las páginas se conectan unas con otras a través de referencias de texto llamados hiperenlaces.

1.2.1 Clientes y Servidores

La web también se puede considerar como un sistema de información distribuido masivo basado en la arquitectura cliente/servidor, tal como se muestra en la Figura 1-1.

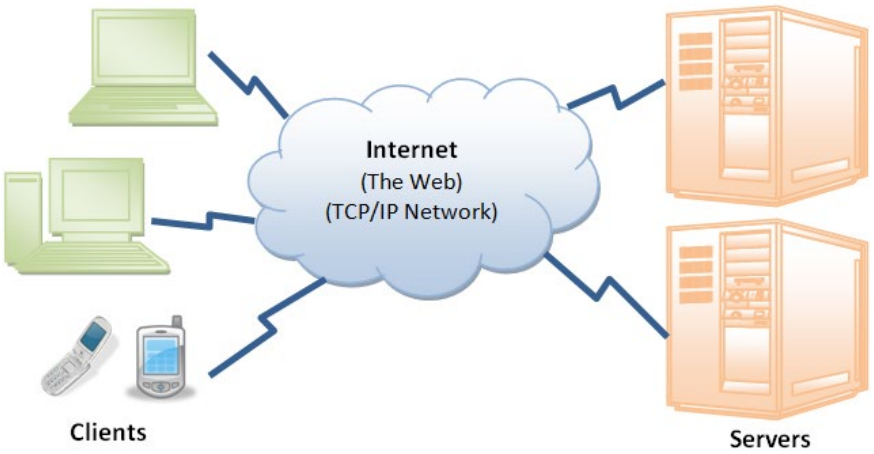


Figura 1-1: Arquitectura Cliente-Servidor

Así, la web consiste de ordenadores llamados *servidores web* que están conectados a Internet y esperando por conexiones, usualmente en el puerto 80 de TCP. Los servidores ejecutan programas especiales que aceptan solicitudes web de los clientes y les responden enviando páginas web a los clientes. Los programas servidores más populares son:

- Nginx
- Apache
- Cloudflare
- LiteSpeed
- Microsoft-IIS
- Node.js

Los usuarios conectados a un servidor web usan programas llamados navegadores web (web browser) que solicita y muestran las páginas web enviadas por los servidores web. Los navegadores más populares son:

- Chrome
- Safari
- Firefox
- Edge
- Opera

1.2.2 Sistema de nombres de dominio (DNS) y Localizador Uniforme de Recursos (URLs)

Para evitar recordar las direcciones IP de los ordenadores en Internet, se creó una capa llamada Sistema de nombres de dominio (Domain Name System, DNS) que permite al usuario escribir o asociar un nombre en texto plano para un servidor web y el sistema buscará la dirección IP para el servidor web y se conectará con él. El nombre en texto para el servidor web se llama *nombre de host*, pudiendo existir varios nombres de host dentro de un *nombre de dominio* más general. Para ello hay servidores DNS en Internet que proporcionan la asociación de nombre/dirección IP.

Para identificar de forma única un recurso en la web se utiliza el localizador uniforme de recursos (Uniform Resource Locator, URL) que tiene la siguiente sintaxis general:

```
protocolo://nombre de host:puerto/ruta-y-nombre-de-archivo
```

Hay 4 partes en una URL:

1. *Protocolo*: corresponde al protocolo de nivel de aplicación utilizado por el cliente y el servidor, por ejemplo, http, https, ftp y mailto.
2. *Nombre de host*: corresponde al nombre de dominio DNS (por ejemplo, www.universidades.gob.es) o la dirección IP (p. ej., 185.73.172.162) del servidor.
3. *Puerto*: el número de puerto TCP que el servidor está escuchando para las solicitudes entrantes de los clientes. Se puede omitir si el servidor web utiliza los puertos estándar del protocolo, 80 para http.
4. *Ruta-y-nombre-de-archivo*: el nombre y la ubicación del recurso solicitado, en el directorio base del documento del servidor.

Por ejemplo, en la URL *http://www.ine.es/prensa/seccion_prensa.htm*, el protocolo de comunicación es http; el nombre de host es *www.ine.es*. El número de puerto no se especificó en la URL y adopta el número predeterminado, que es el puerto TCP 80 para http. La ruta y el nombre de archivo del recurso que se encuentra en el directorio raíz del servidor es "*/prensa/ seccion_prensa.htm*".

1.2.3 Protocolo de transferencia de hipertexto (Hypertext Transfer Protocol, HTTP)

El protocolo de transferencia de hipertexto o HTTP es un conjunto de comandos que un ordenador puede enviar a un servidor para solicitar ficheros alojados en él. Es un servicio de Internet desarrollado como una capa adicional a TCP que constituye la WWW.

Cuando se solicita una página desde un navegador web, el navegador envía un mensaje GET al servidor especificado en la URL. El servidor responde enviando la página y el navegador lo muestra. Una característica importante de HTTP es que es un *protocolo sin estado*, que quiere decir que no hay existe una conexión persistente entre el ordenador cliente y el servidor web y por tanto no se guarda ninguna información sobre conexiones anteriores. Para conseguir la posibilidad de mantener una sesión debe utilizarse otras tecnologías web avanzadas.

Cuando se solicita un documento a un servidor web, este envía el documento en respuesta a esa petición, si existe, junto con un número llamado código de estado, que es una indicación de si la solicitud fue exitosa. Algunos de los códigos de estado más comunes se muestran en la Tabla 2. Los códigos de estado diferentes a 200 se muestran en el navegador con un mensaje de error y existe la posibilidad de elaborar el tipo de mensaje a mostrar.

Tabla 2 Códigos de estado HTTP comunes

Código	Significado
200	OK
301 303	La página se ha movido (temporal o permanente)
403	Acceso prohibido a la página
404	Página no encontrada
500	El servidor experimentó un error interno

Tipos MIME

Los datos que se transmiten por Internet son de varios tipos, como texto, imágenes, audio, video, etc. Los protocolos web clasifican cada tipo de datos usando un identificador de dos partes llamado tipo MIME (Multipurpose Internet

Mail Extension). Cada tipo MIME consiste de un tipo de categoría amplia y un sub tipo separado por una barra (/). Por ejemplo, el tipo MIME image/jpeg representa imágenes en formato jpeg, almacenados usualmente en ficheros .jpg. La Tabla 3 muestra los tipos MIME más comunes.

Tabla 3 Tipos MIME más comunes

Tipo MIME	Extensión fichero	Descripción
application/octet-stream	.exe	Programas ejecutables
application/xml	.xml	Datos en formato xml
application/pdf	.pdf	Ficheros pdf
application/json	.json	Ficheros formato JSON
image/jpeg	.jpg	Imágenes JPEG
image/gif	.gif	Imágenes GIF
video/mp4	.mp4	Video formato mp4
audio/mpeg	.mp3, .mpg	Audio formato mp3
text/html	.html	Páginas web
text/css	.css	Hojas de estilo
text/javascript	.js	Programas JavaScript

1.2.4 Protocolo seguro de transferencia de hipertexto (HTTPS)

El protocolo HTTP fue diseñado inicialmente para intercambiar archivos en un entorno controlado, en un laboratorio. Quiere decir que desde el punto de vista de seguridad HTTP es inseguro y está sujeto a ataques por intervención de la comunicación (man-in-the-middle) y capturas de paquetes de la red para leer el contenido de datos (eavesdropping) que pueden permitir al atacante obtener acceso a bancos y a cuentas de un sitio web e información confidencial.

Para aumentar la transferencia segura de datos se desarrolló el *protocolo seguro de transferencia de hipertexto* (en inglés, Hypertext Transfer Protocol Secure o https). El protocolo https utiliza un cifrado basado en la seguridad de textos SSL/TLS para crear un canal cifrado, entre el servidor y el navegador utilizado por el cliente, que resulta más apropiado para el tráfico de información sensible

que el protocolo HTTP. Así se consigue que la información sensible que un atacante haya podido interceptar en la transferencia de datos durante una conexión, sea un flujo de datos cifrados que le resultará imposible de descifrar. El puerto estándar utilizado por https es el 443.

Para identificar de manera inequívoca que se está usando este protocolo seguro los navegadores lo indican mediante un pequeño candado que se sitúa en la zona izquierda en la barra de direcciones (Figura 2).

Figura 1-2: Indicación de uso de https

1.3 LENGUAJES

Cuando surgió la Web el desarrollo del contenido, solo texto, se realizaba con un lenguaje de marcado llamado HTML (HyperText Markup Language). El propósito inicial del HTML era estructurar el texto para poder compartir documentos entre ordenadores remotos. Con la mejora del hardware y software, obligaron al lenguaje a evolucionar y poder trabajar con otros tipos de medios además de texto, como imágenes y tipos de letras personalizados. Esta extensión significó para los desarrolladores más dificultad para crear y mantener sitios web extensos usando solo HTML. Para resolver ese problema se incorporó un nuevo lenguaje llamado CSS (Cascading Style Sheets), con el que se podía dar el formato o estilo al documento.

La separación entre estructura con HTML y formato con CSS, simplificó el trabajo de los desarrolladores, pero la capacidad de estos lenguajes para responder al usuario o realizar tareas como la reproducción de vídeo o audio era muy limitada. Las empresas dedicadas a dar servicios Web de manera independiente desarrollaron y ofrecían sus propias alternativas. Después de usar varias tecnologías y superar la limitación de la conexión con el contenido del documento se adoptó el lenguaje JavaScript, que ya se encontraba incluido en los navegadores y que, por lo tanto, estaba fuertemente integrado en HTML. De esta manera JavaScript permitía a los desarrolladores acceder al contenido del documento y modificarlo de forma dinámica, solicitar datos adicionales desde el

servidor, procesar información y mostrar los resultados en la pantalla, convirtiendo los sitios web en aplicaciones.

La combinación de las tecnologías HTML, CSS y JavaScript fue la base para expandir la Web. Sin embargo, había un problema que resolver que era que estos lenguajes habían sido desarrollados de forma independiente sin tener en cuenta los cambios presentados entre ellos. La solución surgió con la definición de una nueva especificación llamada HTML5 en 2014. HTML5 unifica todas las tecnologías involucradas en el desarrollo web. A partir de ahora, HTML se encarga de definir la estructura del documento, CSS prepara esa estructura y su contenido para ser mostrado en pantalla, y JavaScript introduce la capacidad de procesamiento necesaria para construir aplicaciones web completamente funcionales.

1.3.1 HTML

HTML (HyperText Markup Language) es un lenguaje compuesto por un grupo de etiquetas o marcas definidas con un nombre rodeado de paréntesis angulares. Los paréntesis angulares delimitan la etiqueta y el nombre define el tipo de contenido que representa. Por ejemplo, la etiqueta `<html>` indica que el contenido es código HTML. Algunas de las etiquetas son declaradas individualmente (por ejemplo, `<br>`) y la mayoría son declaradas en pares, que incluyen una de apertura y otra de cierre, como `<html> </html>` (en la etiqueta de cierre el nombre va precedido por una barra invertida). Las etiquetas individuales y las de apertura pueden incluir atributos para ofrecer información adicional acerca de sus contenidos (por ejemplo, `<html lang="es">`). Las etiquetas individuales y la combinación de etiquetas de apertura y cierre se llaman *elementos*. Los elementos compuestos por una sola etiqueta se usan para modificar el contenido que los rodea o incluir recursos externos, mientras que los elementos que incluyen etiquetas de apertura y cierre se utilizan para delimitar el contenido del documento. Una especificación general de un elemento esta compuesto por una etiqueta inicial que puede contener atributos y una etiqueta de cierre a la que se antepone al nombre de la etiqueta / encerradas ambas con paréntesis angulares. Los documentos HTML se almacenan en ficheros tipo texto con extensión `.html` o `.htm`.

El detalle de HTML se estudiará en el capítulo 2.

1.3.2 CSS

CSS (Cascading Style Sheets) es el lenguaje que se utiliza para definir los estilos de los elementos HTML, como el tamaño, tipos de fuente, el color, el fondo, el borde, etc. Con CSS se puede modificar por completo los estilos por defecto que asignan por defecto los navegadores. Para declarar estilos personalizados, CSS utiliza propiedades y valores y su sintaxis incluye dos puntos después del nombre de la propiedad, y un punto y coma al final para cerrar la línea, por ejemplo `color: #FF0000;`. Las propiedades CSS se pueden agrupar usando llaves. Un grupo de una o más propiedades se llama *regla* y se identifica por un nombre llamado *selector*. El detalle de CSS se estudiará en el capítulo 3.

1.3.3 JavaScript

JavaScript es un lenguaje de programación con el que se puede realizar tareas personalizadas, tales como almacenar valores hasta calcular algoritmos complejos y capacidad de interactuar con los elementos del documento y procesar su contenido dinámicamente. Para insertar código JavaScript dentro de un documento, HTML ofrece el elemento `<script>`. El detalle del estudio de este lenguaje se encuentra en el curso:

Tecnología web aplicada a la creación de materiales interactivos para uso docente. Parte 2: JavaScript aplicado a docencia interactiva: librerías y estrategias.

1.4 HERRAMIENTAS

La creación de un portal web requiere del uso de una serie de herramientas software. El desarrollo de los ficheros con los documentos en HTML, hojas de estilos CSS y los códigos en JavaScript deben escribirse usando un editor de texto. Cuando se tenga la estructura de directorios con los ficheros del portal web será necesario seleccionar y configurar un servidor para alojar las páginas y hacerlas visibles a los usuarios. Para ello existen muchas herramientas que facilitan estas tareas, siendo la mayoría gratuitas.

1.4.1 Editores y entornos de desarrollo

Los archivos con contenido HTML, CSS y JavaScript, son de tipo texto, por lo que se puede usar cualquier editor para crearlos (bloc de notas en Windows o el editor de texto de Apple). Existen editores de texto y entornos de desarrollo especialmente diseñados para el desarrollo web que simplifican el trabajo, por ejemplo, resaltan texto con diferentes colores para ayudar a identificar cada parte del código, indican con marcas errores y advertencias, tienen una edición asistida, listan los archivos de un proyecto en un panel lateral para ayudar a trabajar con múltiples archivos al mismo tiempo, etc. La siguiente es una lista de los editores y de entornos de desarrollo integrado (Integrated Development Environments, IDE) más populares disponibles para ordenadores personales:

- Brackets (brackets.io) es un editor gratuito creado originalmente por Adobe.
- Notepad++ (notepad-plus-plus.org) es un editor gratuito simple de usar y personalizable.
- NetBeans (netbeans.apache.org) es un IDE gratuito multiplataforma con herramientas para administrar archivos y proyectos.
- Visual Studio Code (code.visualstudio.com) es un IDE gratuito multiplataforma desarrollado por Microsoft.

Otras herramientas permiten crear páginas de forma online como Codepen.

Configuración de Visual Studio Code (VS Code)

Debido a las características que ofrece y la popularidad alcanzada, a continuación, se mostrará el uso del IDE Visual Studio Code.

Se puede descargar de forma gratuita desde la URL

https://code.visualstudio.com/download

Después de instalar VS Code se puede añadir extensiones para facilitar y aumentar la rapidez de la codificación. Para instalar las extensiones, se hace clic en el icono Extensions (ver Figura 1-3), se escribe el nombre de la extensión en el cuadro de búsqueda para seleccionar la extensión que se desea y se hace clic en el botón Install.

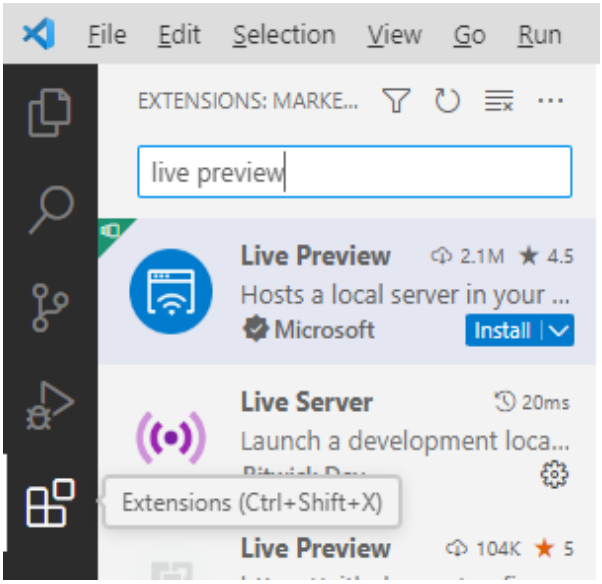


Figura 1-3 Instalación de extensión en VS Code

Conviene añadir las siguientes extensiones (Figura 1-4):

- Prettier – Code formatter: formateador de código.
- Path Intellisense: Complemento que autocompleta nombres de archivos y rutas.
- Auto Rename Tag: cambio automático de nombre de etiquetas HTML/XML pareadas.
- ESLint: encuentra y soluciona problemas en su código JavaScript
- Rainbow Brackets: añade colores para los paréntesis y corchetes.
- Vscode-icons: añade iconos de archivos y carpetas en Visual Studio Code.
- Live Server: ejecuta un servidor local servidor con característica de recarga automática para páginas web estáticas o dinámicas.

Para ejecutar un archivo HTML con Live Server:

1. Abrir el fichero HTML con Visual Studio Code.
2. En el área de edición del fichero hacer un clic en el botón derecho del ratón y seleccionar "Open with Live Server" o hacer click en el botón "Go Live" que se encuentra en la esquina inferior derecha de VS Code.

Después de seguir los pasos anteriores, se visualizará el fichero HTML en el navegador predeterminado usando un servidor disponible en la dirección: `http://127.0.0.1:5500/`

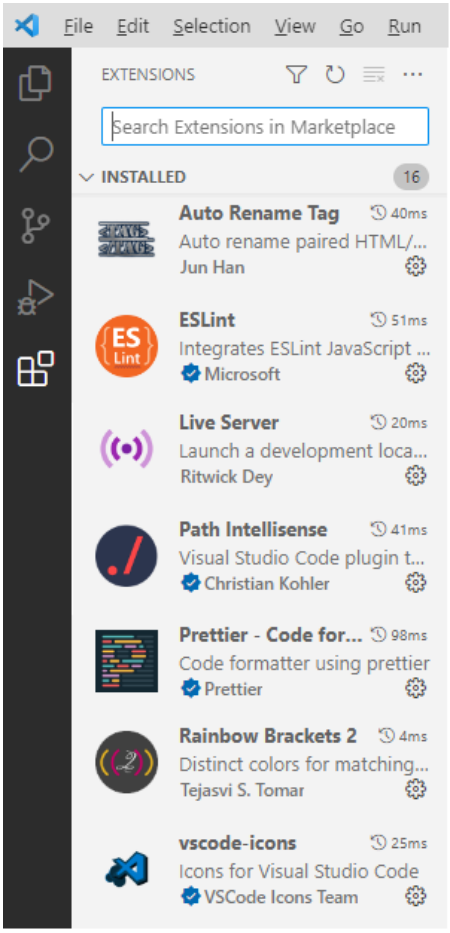
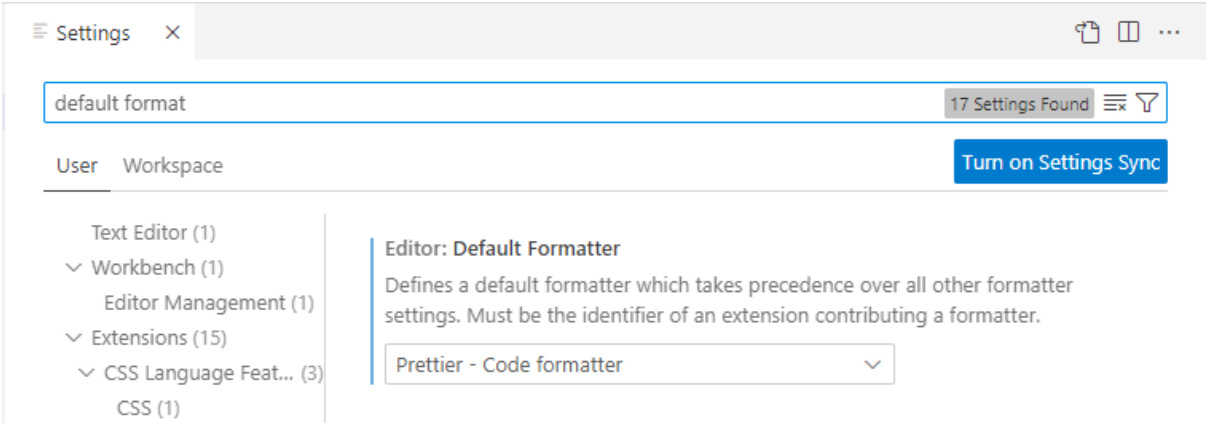


Figura 1-4 Extensiones recomendadas

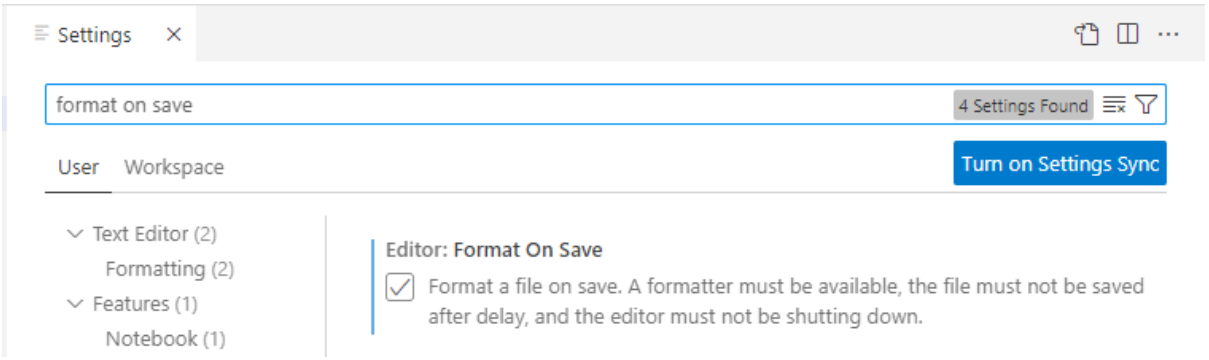
Para configurar el formateador predeterminado para usar la extensión "Prettier – Code formatter" seguir los siguientes pasos:

- Ir al menú y seleccionar: File > Preferences > Settings
- Escribir "default format" en el Search Box
- Seleccionar la opción "Prettier – Code formatter" tal como se muestra en la siguiente figura.



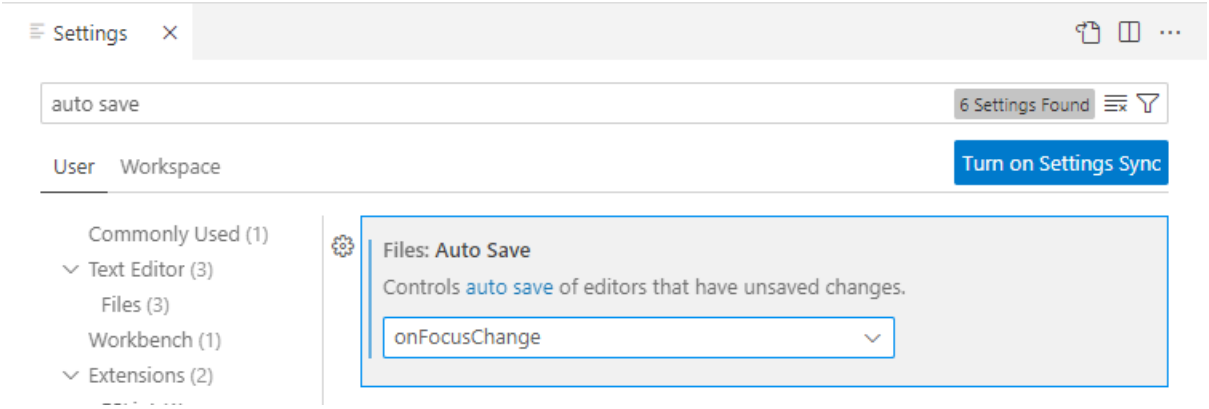
Para permitir dar formato cuando se guarda y tener un código más legible, seguir los siguientes pasos:

- Ir al menú y seleccionar: File > Preferences > Settings
- Escribir "format on save" en el Search Box
- Seleccionar el cajetín "Editor: Format On Save" tal como se muestra en la siguiente figura.



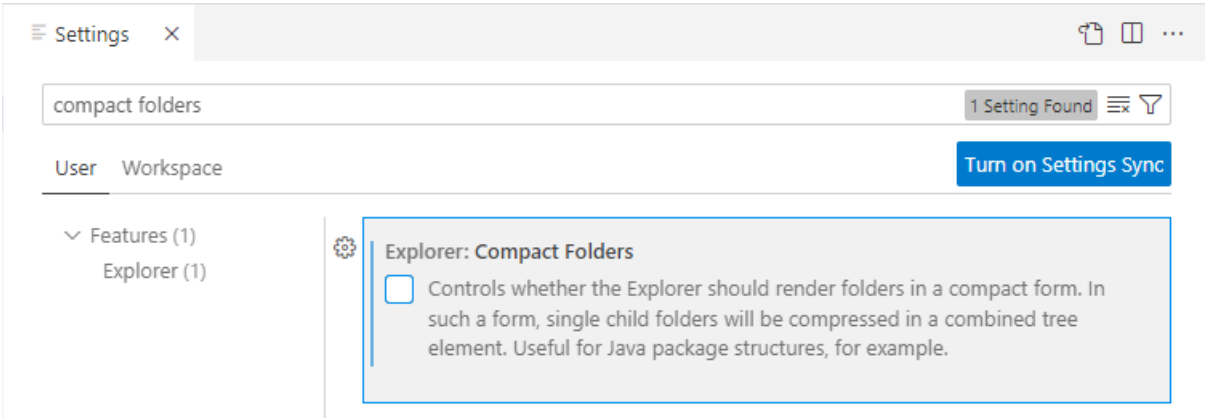
Para permitir guardar archivos automáticamente cuando cambia el texto, seguir los siguientes pasos:

- Ir al menú y seleccionar: File > Preferences > Settings
- Escribir "auto save" en el Search Box
- Seleccionar la opción "OnFocusChange" tal como se muestra en la siguiente figura.



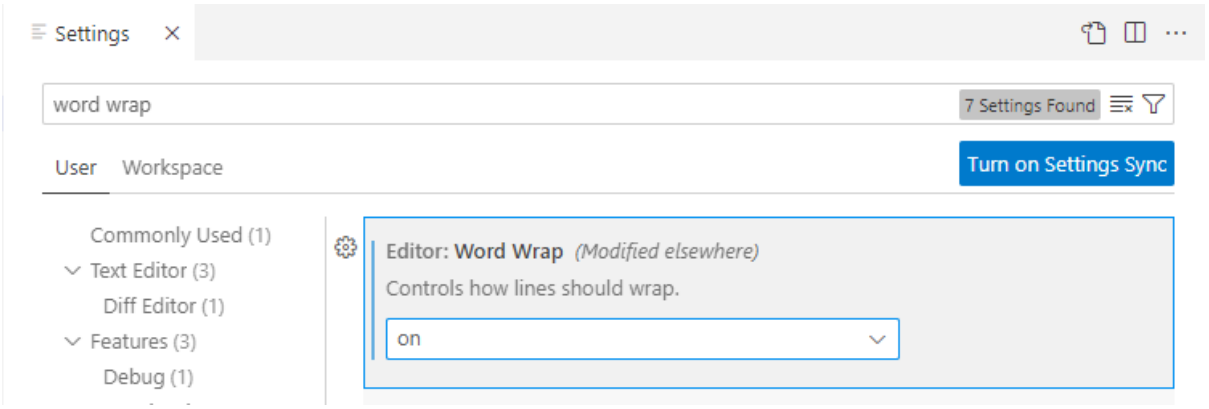
VS Code muestra las carpetas en su explorador en una forma compacta y las subcarpetas serán comprimidas. Para desactivar la opción de "Compact Folders" seguir los siguientes pasos:

- Ir al menú y seleccionar: File > Preferences > Settings
- Escribir "compact folders" en el Search Box
- Deseleccionar la opción "Explorer: Compact Folders" tal como se muestra en la siguiente figura.



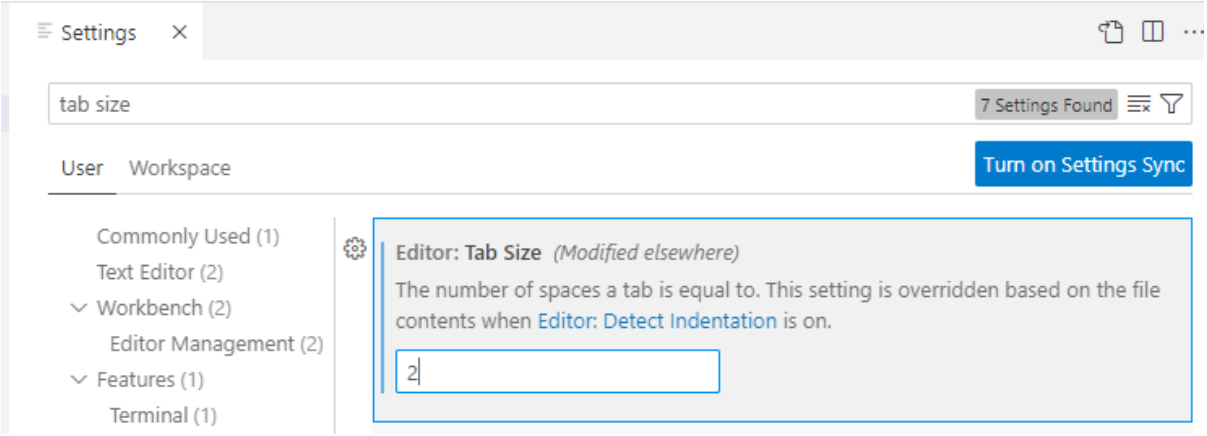
Para desactivar la barra de desplazamiento horizontal se usa la opción "Word Wrap", para ello seguir los siguientes pasos:

- Ir al menú y seleccionar: File > Preferences > Settings
- Escribir "word wrap" en el Search Box
- Seleccionar la opción "on" de "Editor: Word Wrap" tal como se muestra en la siguiente figura.



El número de espacios por defecto del tabulador es 4. Normalmente es necesario dos espacios, así que para cambiarlo seguir los siguientes pasos:

- Ir al menú y seleccionar: File > Preferences > Settings
- Escribir "tab size" en el Search Box
- Escribir el número 2 en "Editor: Tab Size" como se muestra en la siguiente figura.



Configuración del espacio de trabajo

Para crear un espacio de trabajo se sigue los siguientes pasos:

- Crear un directorio, por ejemplo "C:\CursoI\Ejemplo"
- En VS Code seleccionar el menú File > Open Folder…
- Seleccionar el directorio, en el ejemplo "C:\CursoI\Ejemplo"
- Pasar el cursor sobre la carpeta Ejemplo para que aparezcan iconos (Figura 1-5).
- Hacer clic en el icono New File
- Escribir "index.html" y pulsar Enter
- Escribir el siguiente código HTML en el editor

index.html

```
<!DOCTYPE html>
<html>
  <head>
    <title>Prueba de VS Code</title>
  </head>
  <body>
    <h2>Página de prueba</h2>
  </body>
</html>
```

- Seleccionar en el menú File > Save para guardar los cambios en el fichero index.html

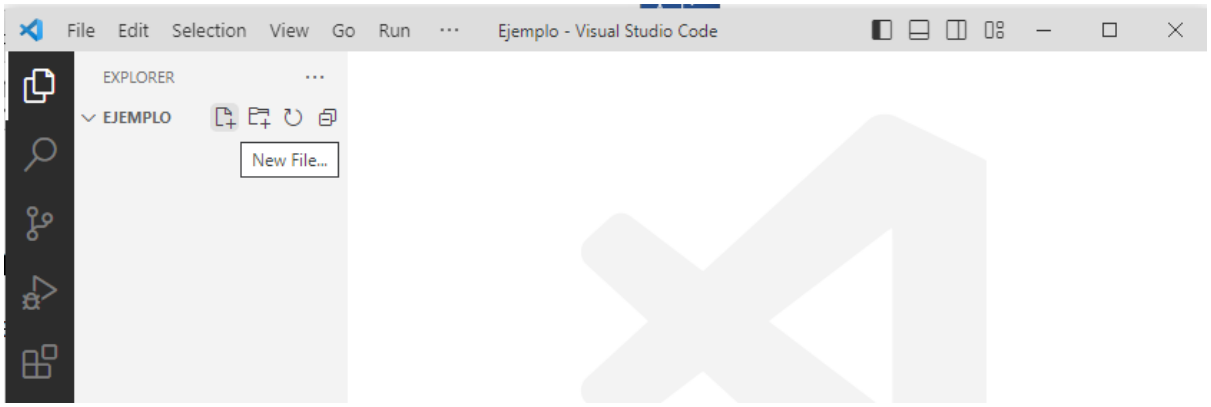


Figura 1-5 Configuración de espacio de trabajo en VS Code

Para visualizar el código en un navegador se hace clic en botón secundario del ratón sobre el fichero index.html de la Explorer Window y clic sobre la opción Open with Live Server (Figura 1-6). Se lanzará un servidor local web con recarga automática del fichero index.html (Figura 1-7).

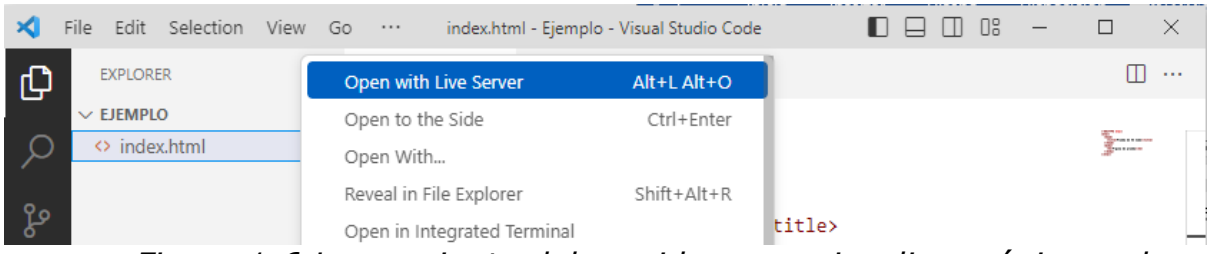


Figura 1-6 Lanzamiento del servidor para visualizar página web

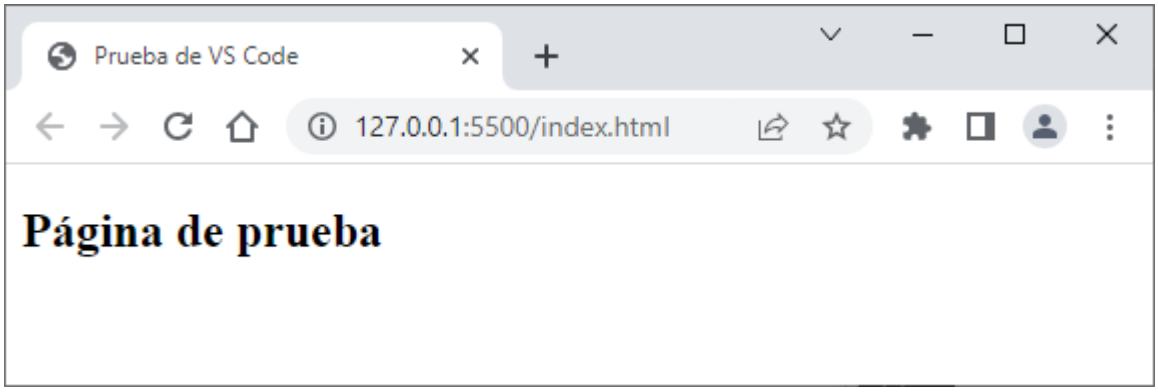


Figura 1-7 Visualización de fichero index.html

1.4.2 Herramientas de los navegadores

La mayoría de navegadores ofrecen herramientas para la visualización de la estructura y modificación de los elementos de las páginas web. En particular disponen de herramientas útiles para ejecutar y depurar código JavaScript. A continuación, se describirá tales herramientas usando el navegador Google Chrome por ser el más popular, en otros navegadores el proceso es similar.

Panel de Elementos

Para usar las herramientas de desarrollo de Chrome se consigue de cualquiera de las siguientes formas:

- Haciendo clic con el botón derecho del ratón en una página y seleccione "Inspeccionar" en el menú.
- En al menú de configuración de Google Chrome (los tres puntos verticales en la esquina superior derecha del panel de control de la ventana del navegador). En el menú desplegable, seleccionar Más herramientas, luego Herramientas para desarrolladores.
- Utilizando la combinación de teclas "Ctrl+Mayus+I" o pulsando la tecla de función F12.

El panel de herramientas para desarrolladores consta de tres partes principales (Figura 1-8):

- Panel Elementos/DOM: contiene el árbol del Modelo de Objetos del Documento (DOM) de la página y da acceso completo al código fuente HTML. Normalmente está situado en la parte superior de la ventana.
- Panel CSS: permite visualizar y modificar las reglas de estilo de una página web cambiando, añadiendo o eliminando propiedades CSS. Este panel se encuentra debajo de Estilos.
- Consola: se utiliza para ejecutar JavaScript. Aparece en la sección inferior de la ventana.

También incluyen otras funciones, como Fuentes, Red, Aplicación y Seguridad entre otras.

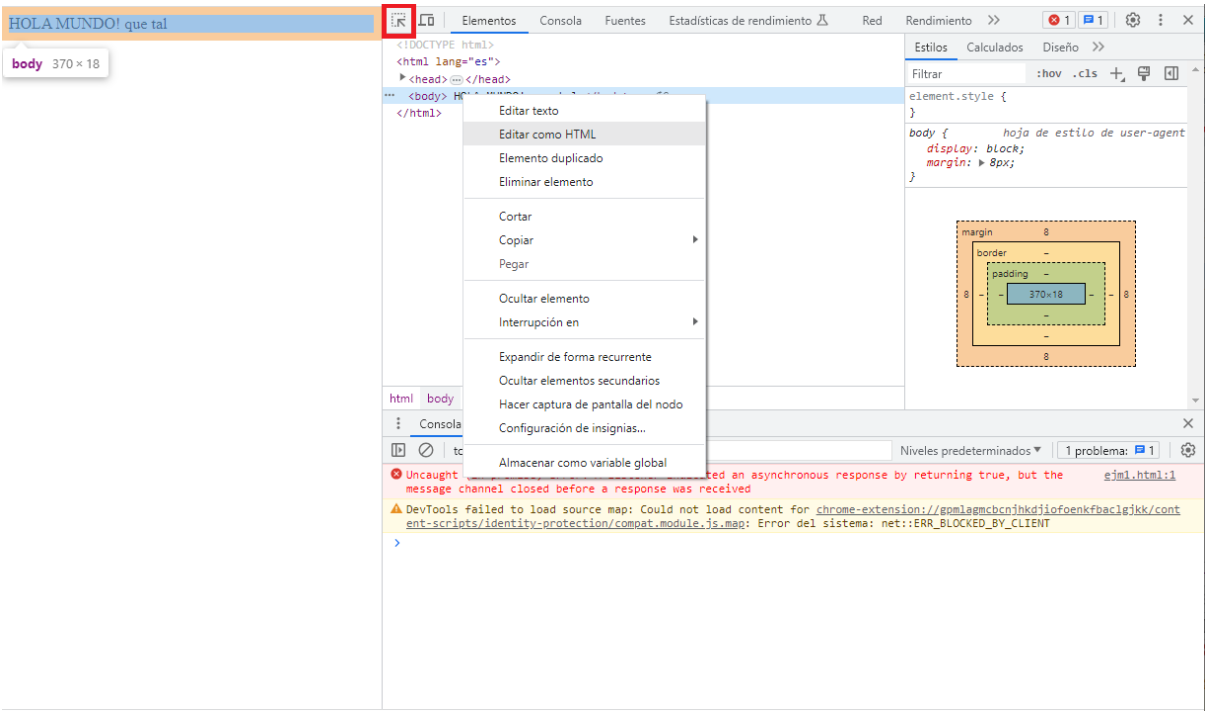


Figura 1-8 Paneles de Inspeccionar

Desde los paneles de Elementos y CSS se puede realizar las siguientes acciones:

- Edición en vivo de CSS: permite ver los cambios en tiempo real mientras se edita en el panel de CSS.
- Prueba de diseño: permite probar diferentes diseños de un sitio web antes de realizar cambios permanentes en el código.
- Diagnóstico de depuración: ayuda a comprobar si la página contiene código no válido.
- Edición temporal: permite ajustar los elementos de la página web para verlos en el navegador.

Por ejemplo, para cambiar un elemento de la página, es necesario modificar el código fuente CSS o HTML. De este modo, se puede editar el texto y los elementos de estilo, como el tamaño y el color de la fuente. El panel DOM permite modificar fácilmente el texto. Tras abrir el cuadro de Elementos, se utiliza la función Inspeccionar (el icono del cursor situado en la parte superior izquierda del panel) para resaltar el elemento cuyo código fuente se desea modificar. A continuación, se hace clic con el botón derecho del ratón en el código resaltado dentro del árbol DOM y se selecciona Editar como HTML (Figura 9). El cuadro del editor se expandirá, permitiendo modificar el texto. Para visualizar

los cambios se desmarca el elemento. Un método más rápido consiste en hacer doble clic en el texto que se quiere modificar en el panel DOM y sustituirlo.

La modificación del estilo de los elementos es similar usando el panel CSS. Después de seleccionar un elemento con Inspeccionar se hace clic en la propiedad `element.style` en la parte superior del panel CSS y se añade las respectivas descripciones de estilo. Por ejemplo, para añadir un color de fondo al body de la página se escribe `background-color:red` (Figura 1-9).

Figura 1-9 Adición de estilo

Para añadir otro estilo, se selecciona de nuevo la propiedad `element.style` y el inspector web añadirá otra línea vacía para rellenar. Al pasar el ratón por encima de las propiedades CSS en el panel CSS, aparecerá una casilla de verificación junto a cada una de ellas. Si se desmarca la casilla se omiten los estilos que no se desean que se muestren. También se puede hacer clic en una propiedad o en un valor para reemplazarlo.

1.4.3 Estructura y Alojamiento de sitios Web

Los sitios web son un conjunto de archivos que los usuarios descargan con sus navegadores desde servidores, que son ordenadores con mayores prestaciones que los normales, conectados permanentemente a la red y ejecutando programas que les permiten responder a las solicitudes de los usuarios. Cuando un usuario accede a un sitio web a través de un navegador indicando la dirección o URL, éste se comunica con el servidor que responde con los archivos, procesa

su contenido y lo muestra en pantalla. Para el desarrollo de un sitio web es necesario estructurarlo y probar con el navegador de forma local cada página. También se puede instalar un programa servidor para el alojamiento del sitio web de forma local o de forma global si se registra el dominio. Otra posibilidad es usar un portal de alojamiento web donde poder descargar el sitio web.

Estructura del sitio Web

Los sitios web están compuestos de múltiples documentos que se pueden descargar desde el navegador. Los documentos que conforman un sitio web se llaman páginas que pueden estar relacionadas a través de hiperenlaces con lo que se navega entre ellas.

El desarrollo de un sitio web conlleva la creación de un archivo por cada página que se requiere incluir. Junto con los archivos, es necesario incluir los archivos con las imágenes, videos y cualquier otro recurso contenido dentro de estas páginas. Quiere decir que un sitio web se puede organizar a través de directorios y archivos. Por ejemplo, la Figura 10 muestra los directorios y archivos de un sitio web que incluye tres directorios: *html*, *imagenes* y *recursos*, y tres archivos: *contacto.html*, *index.html* y *novedades.html*. El directorio *imagenes* contiene las imágenes que queremos mostrar dentro de las páginas web, el directorio *recursos* los archivos que con los códigos en CSS y JavaScript y el directorio *html* contiene los archivos de otras páginas web en lenguaje html. El archivo *index.html* contiene el código y la información correspondiente a la página principal que el usuario ve cuando entra al sitio web, el archivo *contacto.html* contiene el código para presentar un formulario que el usuario puede rellenar para enviar un mensaje y el archivo *novedades.html* contiene el código necesario para mostrar información novedosa para el usuario.

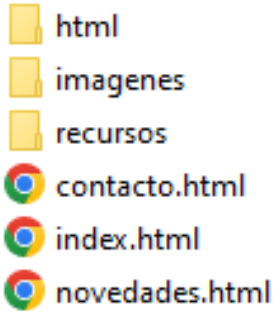


Figura 1-10 Estructura de un sitio web

Para asignar los nombres de los archivos no usar espacios en blanco, en su lugar se usa el guión bajo (_) para separar palabras, evitar caracteres especiales (?, %, #, /), y usar solo letras minúsculas sin acentos y números. La extensión del archivo responde al tipo o lenguaje usado en su contenido, por ejemplo, los archivos .html contienen código HTML. Otro aspecto a tener en cuenta es que la mayoría de servidores designan archivos por defecto en caso de que el usuario no especifique ninguno en la URL. El nombre utilizado con más frecuencia es *index*. Si un usuario accede al servidor sin especificar el nombre del archivo que intenta abrir, el servidor busca un archivo con el nombre *index* y lo envía como respuesta al cliente. Por esta razón, el archivo *index* es el punto de entrada del sitio web y debemos incluirlo. Algunos servidores usan otros nombres como *home* o *default*.

Como se describió en las secciones 1.1.3 y 1.2.2, los servidores se identifican con un valor IP que es único para cada ordenador y sirve como una dirección que permite identificar a un ordenador dentro de una red. Cuando el usuario solicita un documento a través del navegador, primero busca el servidor a través de la dirección IP y luego le pide que le envíe el documento. Puesto que una dirección IP es difícil de recordar se utiliza un sistema (DNS) que identifica a cada servidor con un nombre específico, llamado dominio, por ejemplo *www.google.com*, de tal forma que cuando el usuario pide al navegador que acceda a un dominio, por ejemplo *www.google.com*, primero accede al servidor DNS que contiene una lista de dominios con sus respectivas direcciones IP. Este servidor encuentra la IP 142.250.200.132 asociada al dominio *www.google.com*, la devuelve al navegador para acceder al sitio web de Google por medio de esta IP. Los dominios, a su vez, son identificadores o nombres sencillos con una extensión asociada al propósito del sitio web o país, por ejemplo: *.com* (comercial), *.org* (organización), *.edu* (educación), *.es* (España). Como los sitios web están compuestos por múltiples archivos, se debe agregar el nombre del archivo al dominio para formar el localizador del recurso o URL (Figura 1-11).

Figura 1-11 Ejemplo de URL

Los sitios web normalmente están compuestos por muchas páginas web y para relacionar unas páginas con otras se usan hipervínculos o enlaces que se pueden definir con URL absolutas o relativas. Las URL absolutas incluyen toda la información necesaria para acceder al recurso, mientras que las

relativas solo declaran la parte de la ruta que el navegador tiene que agregar a la URL actual para acceder al recurso. Por ejemplo, si en el documento de la Figura 1-10 *index.html* se tiene un enlace que referencia una imagen dentro del directorio *imagenes*, se puede construir el enlace con la URL absoluta *http://www.ejemplo.com/imagenes/imagen1.jpg*, pero también se puede usar la URL relativa *imagenes/imagen1.jpg* al que el navegador agregará a esta ruta la URL actual.

Las URL relativas no solo pueden determinar una ruta hacia abajo, sino también hacia arriba de la jerarquía. Por ejemplo, si en un documento dentro del directorio *recursos* de la Figura 1-10 se quiere acceder al documento *noticias.html* en el directorio raíz, se puede crear una URL relativa usando los caracteres *../* al comienzo de la ruta para indicar al navegador que el documento se encuentra dentro del directorio padre del actual directorio, por lo que la URL relativa sería *../noticias.html*.

Registro de dominio y alojamiento web

Para hacer disponible un sitio web en Internet, es necesario registrar el dominio que los usuarios deben escribir en la barra de navegación del navegador para acceder a él. En el apartado anterior se mencionó que un dominio está compuesto de cualquier nombre personalizado y una extensión que determina el propósito del sitio web. También es necesario alojar el sitio web en alguna plataforma que reúna las características idóneas (fiabilidad, seguridad y rapidez) para ello.

Existe un mercado de proveedores de estos servicios que ofrecen una serie opciones con sus respectivos precios. Existen portales que ofrecen alojamiento web gratuito. Uno de ellos es *https://neocities.org/* cuyo registro es muy simple.

En principio, para probar el desarrollo de páginas con HTML y CSS solo es necesario un navegador. Sin embargo, algunos programas JavaScript solo trabajan cuando se descargan desde un servidor por lo que requieren ser alojadas en un servidor para funcionar. En ese caso existen dos opciones: obtener una cuenta de alojamiento web o instalar un servidor en el ordenador de desarrollo. En el caso de optar por la última opción, existen varios programas que se pueden descargar gratuitamente, por ejemplo WampServer, XAMPP, MAMP, MEAN.JS para que un ordenador personal se comporte como servidor web local.

1.5 EJERCICIOS

1. Dadas las siguientes URLs

   http://www.ejemplo.es/misitio/index.html
   http://www.ejemplo.es/misitio/imagenes/sonrie.jpg
   la url relativa de la imagen sonrie.jpg para enlazarla desde el fichero index.html es: (seleccionar una opción)

   a) /imagenes/sonrie.jpg

   b) sonrie.jpg

   c) ../imagenes/sonrie.jpg

   d) imagenes/sonrie.jpg <---- Respuesta correcta

2. Dadas las siguientes URLs

   http://www.ejemplo.es/misitio/files/index.html
   http://www.ejemplo.es/misitio/imagenes/juego.jpg
   la url de forma relativa de la imagen juego.jpg para mostrarla en el fichero index.html es: (seleccionar una opción)

   a) ../imagenes/juego.jpg

   b) ../../imagenes/juego.jpg <---- Respuesta correcta

   c) /imagenes/juego.jpg

   d) imagenes/juego.jpg

3. En la siguiente estructura:

/index.html
|---imagenes/
|---CSS/
|---Multimedia/
|---paginas/

se tiene la imagen fondo.jpg en el directorio imagenes y se enlaza como imagen de fondo desde la hoja de estilos estilo.css en el directorio CSS: Seleccionar una opción:

a) ./imagenes/fondo.jpg

b) ../../imagenes/fondo.jpg

c) ../imagenes/fondo.jpg <---- Respuesta correcta

d) imagenes/fondo.jpg

4. En la estructura de la pregunta 3, se tiene una página index.html en el directorio paginas que se quiere enlazar con la pagina index.html en la raíz. Seleccionar una opción:

a) ./paginas/index.html

b) ../CSS/index.html

c) ../index.html <---- Respuesta correcta

d) /index.html

2 HTML

HTML (HyperText Markup Language o lenguaje de marcado de hipertexto) es un lenguaje de marcas que describe el contenido y la estructura de las páginas web y que se puede visualizar con cualquier navegador web.

Un documento HTML o página web es un archivo de texto con extensión `.html` que contiene una serie de *elementos*, que se utilizan para encerrar, delimitar o marcar diferentes partes del contenido a través de *etiquetas*. Las etiquetas pueden escribirse tanto en mayúsculas como en minúsculas, pero se recomienda el uso de minúsculas por cuestiones de legibilidad.

En un documento HTML también se puede incluir estilos definidos en CSS (Cascading Style Sheets o hojas de estilo en cascada) para darle formato y código JavaScript (para programación) para añadirle dinámica.

HTML se encuentra estandarizado en la norma ISO de SGML (Standard Generalized Markup Lenguage) por el W3C (World Wide Web Consortium). La última versión es HTML 5 publicada como recomendación W3C HTML 5.3 con en Enero de 2021.

Etiquetas, atributos y elementos HTML

Una especificación general de los elementos HTML se muestra en la Figura 2-1, que muestra que un elemento está definido por una etiqueta inicial que puede contener atributos y una etiqueta de cierre a la que se antepone al nombre de la etiqueta / encerradas ambas con paréntesis angulares. Los elementos pueden anidarse, es decir que pueden convertirse en un contenedor o ser contenido por otros elementos, a excepción de los elementos estructurales como `<html>`, `<head>` y `<body>` que tienen un lugar específico en un documento HTML.

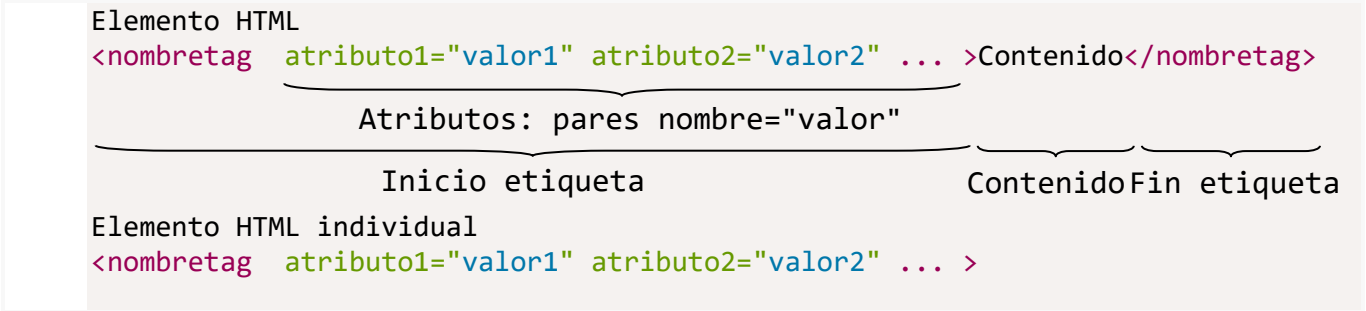


Figura 2-1 Especificación general de un elemento HTML

Etiquetas HTML

Una etiqueta (tag) de apertura HTML está encerrada por un par de corchetes angulares en forma de `<nombretag` por ejemplo `<p>`, `<title>`, que está asociado con una etiqueta de cierre emparejada que tiene una barra diagonal inicial, por ejemplo `</p>`, `</title>`. Los nombres de las etiquetas deben escribirse en minúsculas.

Atributos de una etiqueta

Los atributos, en forma de pares `nombre="valor"`, se pueden incluir en la etiqueta de apertura para proporcionar información adicional del elemento. Ejemplos:

En `<html lang="es">`, el atributo `lang="es"` especifica el idioma natural del documento (español).

En `<meta charset="utf-8">`, el atributo `charset="utf-8"` especifica el esquema de codificación de caracteres.

La etiqueta individual `<img>` (imagen) puede contener estos atributos:

```
<img src="logotipo.gif" alt="logotipo" width="50" height="30" >
```

El atributo `src` especifica la URL de origen de la imagen; el atributo `alt` especifica un texto alternativo, si la imagen no se puede mostrar; los atributos `width` y `height` especifican la anchura y la altura del área de visualización de la imagen.

Algunos de los atributos son obligatorios (por ejemplo, los atributos `src` y `alt` de la etiqueta), mientras que otros son opcionales, por ejemplo, los atributos `width` y `heigth` de la etiqueta `img`, que sirve cambiar el tamaño de la imagen. Si no se especifican el navegador usará el ancho y el alto original de la imagen.

Se pueden especificar múltiples atributos separados por espacios, de la siguiente manera:

```
<nombre_tag atributo1="Value1" atributo2="Value2" ...>...</nombre_tag >
```

Los atributos se escriben en forma de pares nombreatributo="valor". El valor del atributo debe ponerse entre comillas simples o dobles.

Elementos HTML

Un elemento HTML consta de las etiquetas de apertura y cierre, y el contenido intermedio, por ejemplo, `<p>A por almendra</p>`, `<strong>¡Peligro!</strong>`.

Hay dos tipos de elementos:

Elemento contenedor: un elemento contenedor tiene una etiqueta de apertura `<tag-name>` que activa un efecto en su contenido y una etiqueta de cierre correspondiente `</tag-name>` para interrumpir el efecto. En otras palabras, los elementos contenedores aplican formato a sus contenidos. Por ejemplo:

```
<h1> Las etiquetas h1 encierran un encabezado de nivel 1 </h1>

<p> Las etiquetas p se utilizan para <em> marcar </em> un <strong> párrafo
</strong> . </p>
```

Elemento vacío (o elemento independiente): un elemento vacío no encierra ningún contenido, pero se usa para lograr un efecto determinado, por ejemplo, `<hr>` se usa para dibujar una regla horizontal; `<br>` introducir un salto de línea manual; y `<img>` para incrustar una imagen externa.

Anidar elementos y el árbol del documento

Un elemento HTML se puede anidar dentro de otro elemento, por ejemplo, `<p>Esta es mi <em>primera</em> página web!</p>`. Es importante anidar correctamente los elementos, por ejemplo, la secuencia `<p>...<em>...</p>...</em>` no es válida.

Un documento HTML válido exhibe una estructura de árbol llamada DOM (Document Object Model o Modelo de objetos del documento), con el elemento `<html>` como elemento raíz del árbol del documento, con dos elementos secundarios: `<head>` y `<body>`. Por ejemplo, para el siguiente fichero:

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="utf-8">
  <title>Árbol</title>
  <style>
</head>
<body>
<div class="caja">
  <header>
    <h1>Cabecera: header</h1>
```

```
  </header>
  <nav>
    <ul>
      <li><a href="#"> Menú 1</a></li>
      <li><a href="#"> Menú 2</a></li>
      <li><a href="#"> Menú 3</a></li>
    </ul>
  </nav>
  <article>Contenido de la página </article>
  <footer>Pie de página</footer>
</div>
</body>
</html>
```

el árbol DOM correspondiente se muestra en la Figura 2-2.

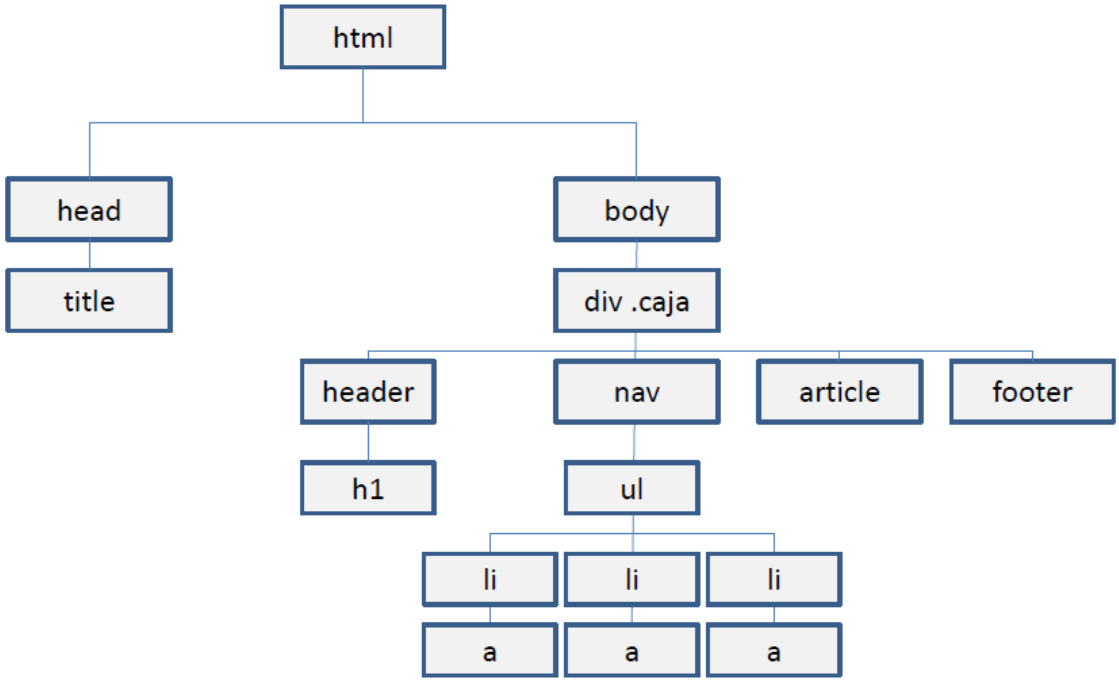


Figura 2-2 Estructura DOM de un documento HTML

2.1 ESTRUCTURA DE UNA PÁGINA

La primera línea que debe insertarse en un documento HTML es:

```
<!DOCTYPE html>
```

para asegurar que el contenido sea interpretado por el navegador como código HTML5. A continuación, se inserta el elemento `<html>`, que es el elemento raíz del documento HTML. Tiene dos hijos `<head>` y `<body>` con los que se describe la estructura general del documento. Por tanto, la estructura de toda página web es:

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

Una descripción de estos elementos es:

`<html></html>` - Elemento que encierra todo el contenido de la página y se le conoce como el elemento raíz. Puede incluir el atributo `lang` para definir el idioma del contenido del documento.

`<head></head>` - Elemento que actúa como un contenedor de todo aquello que se quiere incluir en la página HTML que no es contenido visible, tales como palabras clave (keywords), descripción de la página para búsquedas, código CSS para dar estilo al contenido, declaraciones del juego de caracteres y los archivos externos requeridos por el documento.

`<body></body>` — El elemento <body> encierra todo el contenido visible de la página (texto, imágenes, videos, pistas de audio, etc.).

2.1.1 Elemento <head>

El elemento `<head>` se coloca entre la etiqueta `<html>` y la etiqueta `<body>` y puede contener los siguientes elementos:

- `<title>` que es obligatorio y define el título del documento.
- `<meta>` usado para especificar el conjunto de caracteres, la descripción de la página, las palabras clave, el autor del documento y la configuración de la ventana gráfica.
- `<link>` usado para vincular a recursos externos a la página.
- `<script>` usado para definir código JavaScript del lado cliente.
- `<style>` usado para definir la información de estilo para el documento.
- `<base>` especifica la URL base y/o el destino para todas las URL relativas en una página.

Un ejemplo de algunos elementos y sus atributos, se muestra en el siguiente código:

```
<!DOCTYPE html>
<html lang="es">
  <head>
    <meta charset="utf-8">
    <title>Título de la página</title>
    <!-- Etiquetas opcionales -->
    <meta name="description" content="Contenido">
    <meta name="keywords" content="Palabra clave1, Palabra clave2">
    <meta name="author" content="Nombre">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link href="favicon.ico" rel="shortcut ico" type="image/x-icon">
    <link href="filename.css" rel="stylesheet">
    <script src="filename.js"></script>
  </head>
  <body>
  </body>
</html>
```

La descripción de los elementos del código es:

Para insertar **comentarios** en cualquier parte de la página se encierra el comentario, que puede ser multilínea, entre las etiquetas `<!-- ... -->`. Los comentarios se utilizan para explicar el código y hacerlo más legible.

La etiqueta `<html>` puede incluir el atributo `lang` para declarar el idioma en el que se escribe el contenido de la página, en este caso `lang="es"`.

`<title></title>` - establece el título de la página, que es el título que aparece en la pestaña o en la barra de título del navegador cuando la página es cargada, y se usa para describir la página cuando es añadida a los marcadores o como favorita.

Con el elemento `<meta>` se representa metadatos asociados con el documento a través de atributos. En el código anterior:

`<meta charset="utf-8">` - establece el juego de caracteres que se usará en el documento, `utf-8` (Unicode Transformation Format 8-bit) que representa el código de caracteres Unicode y es compatible con el código ASCII. Permite visualizar los caracteres de todos los idiomas.

`<meta name="description" content="Contenido">` - define una descripción de la página web

`<meta name="keywords" content="Palabra clave1, Palabra clave2">` - define las palabras clave para los buscadores.

`<meta name="author" content="Nombre">` - define el autor de la página

`<meta name="viewport" content="width=device-width, initial-scale=1">` - configura la ventana gráfica para que la página se vea bien en todos los dispositivos. En este caso, indica que en los dispositivos móviles el ancho de la página debe corresponder con el ancho de la pantalla

Otro ejemplo del elemento <meta> es:

`<meta http-equiv="refresh" content="30">` - para refrescar el documento cada 30 segundos.

Con el elemento `<link>` se define la relación entre el documento actual y un recurso externo.

`<link href="favicon.ico" rel="shortcut ico" type="image/x-icon">` - añade una imagen favicon especificada con el atributo `href` a la izquierda del título de la pestaña del navegador. Hay portales web que permiten la creación de estos iconos, por ejemplo https://www.favicon.cc/

`<link href="filename.css" rel="stylesheet">` - enlaza a una hoja de estilo externa.

El elemento <script> se usa para definir programas en código JavaScript que se ejecuta en el navegador, es decir, en el lado cliente.

`<script src="filename.js"></script>` - define un script Javascript externo.

2.1.2 Elemento <body>

El elemento `<body>` encierra todo el contenido de un documento HTML a través de elementos para definir la estructura del documento, encabezados, párrafos, imágenes, hipervínculos, tablas, listas, etc. Un documento HTML solo puede tener un elemento `<body>`.

En general, los elementos HTML se pueden clasificar, desde el punto de vista del espacio que aplican, en dos bloques: elementos "en bloque" o elementos "en línea".

- Un *elemento en bloque* **block** ocupa todo el ancho de la página y fuerzan a una nueva línea antes y después del elemento. Ejemplos de elementos block son:
- Un *elemento en línea* **inline** solo ocupa el ancho necesario y no fuerzan nuevas líneas, aunque puede abarcar varias líneas.

Por otra parte, en HTML5 se agregaron *elementos semánticos* que describen con precisión el propósito del elemento y el tipo de contenido dentro de ellos.

La lista completa de elementos HTML excluyendo los obsoletos o no estándar son:

<a>
<abbr>
<address>
<area>
<article>
<aside>
<audio>
<b>
<base>
<bdi>
<bdo>
<blockquote>
<body>

<button>
<canvas>
<caption>
<cite>
<code>
<col>
<colgroup>
<data>
<datalist>
<dd>
<del>
<details>
<dfn>
<dialog>

<dl>
<dt>

<embed>
<fieldset>
<figcaption>
<figure>
<footer>
<form>
<h1>
<head>
<header>
<hgroup>
<hr>
<html>
<i>
<iframe>
<img>
<input>
<ins>
<kbd>
<label>
<legend>

<link>
<main>
<map>
<mark>
<menu>
<meta>
<meter>
<nav>
<noscript>
<object>

<optgroup>
<option>
<output>
<p>
<picture>
<pre>
<progress>
<q>
<rp>
<rt>
<ruby>
<s>
<samp>
<script>
<section>
<select>
<slot>
<small>
<source>

<style>
<sub>
<summary>
<sup>
<table>
<tbody>
<td>
<template>
<textarea>
<tfoot>
<th>

<thead>	<track>	<video>
<time>	<u>	<wbr>
<title>		
<tr>	<var>	

La lista de los elementos en bloque de HTML con su descripción es:

<address>	Información de contacto.
<article>	Contenido de Articulo.
<aside>	Contenido adicional.
<audio>	Reproductor de audio
<blockquote>	Bloque de "cita".
<canvas>	Dibujo canvas.
<dd>	Descripción de definición.
	División de documento.
<dl>	Lista de definición.
<fieldset>	Etiqueta de conjunto de campos.
<figcaption>	Leyenda de figura.
<figure>	Grupos contenido multimedia con una leyenda.
<footer>	Sección o pie de página.
<form>	Formulario de entrada.
<h1>,<h2>,<h3>, <h4>,<h5>,<h6>	Niveles de cabecera 1-6.
<header>	Sección o cabecera de página.
<hgroup>	Grupos información de encabezado.
<hr>	Regla Horizontal (línea divisoria).
	Elemento de lista.
<main>	Engloba el único contenido central del documento.
<nav>	Contiene enlaces de navegación.
<noscript>	Contenido para ser usado si los scripts no son soportados.
	Lista ordenada.
<output>	Formulario de salida.
<p>	Párrafo.
<pre>	Texto preformateado.
<section>	Sección de una página web.
<table>	Tabla.
<tfoot>	Pie de tabla.
	Lista no ordenada.
<video>	Reproductor de vídeo.

Lista de elementos en línea de HTML:

b, big, i, small, tt
abbr, acronym, cite, code, dfn, em, kbd, strong, samp, time, var
a, bdo, br, img, map, object, q, script, span, sub, sup
button, input, label, select, textarea

2.2 ELEMENTOS DE CONTENIDO

A continuación, se detallan los elementos y atributos más importantes que se usan dentro del elemento <body> para dar contenido al documento.

2.2.1 Elementos para estructurar el contenido

Son elementos con nombres más descriptivos (semánticos) que permiten identificar cada parte del contenido de un documento. Estos elementos informan al navegador sobre el propósito de cada sección. La siguiente lista incluye todos los elementos disponibles para definir la estructura del contenido de la página.

`<main>`- elemento para definir una división que contiene el contenido principal del documento. No debe haber más de un elemento `<main>` en un documento.

`<header>` - elemento para definir la cabecera del cuerpo o de secciones.

`<nav>` - elemento para definir una división con enlaces necesarios para la navegación.

`<article>` - elemento usado para marcar un artículo independiente. Puede contener su propia cabecera, secciones y pie de página.

`<section>` - elemento para definir una sección genérica.

`<aside>` - elemento para definir una división que contiene información relacionada con el contenido principal pero que no es parte del mismo. Asociada como barra lateral al contenido principal.

`<footer>` - elemento para definir el pie del cuerpo o de secciones

`<div>` - elemento para definir una división genérica. Debe usarse cuando no se puede aplicar ningún otro elemento.

La Figura 2-3 muestra un diagrama de una página que utiliza estos elementos.

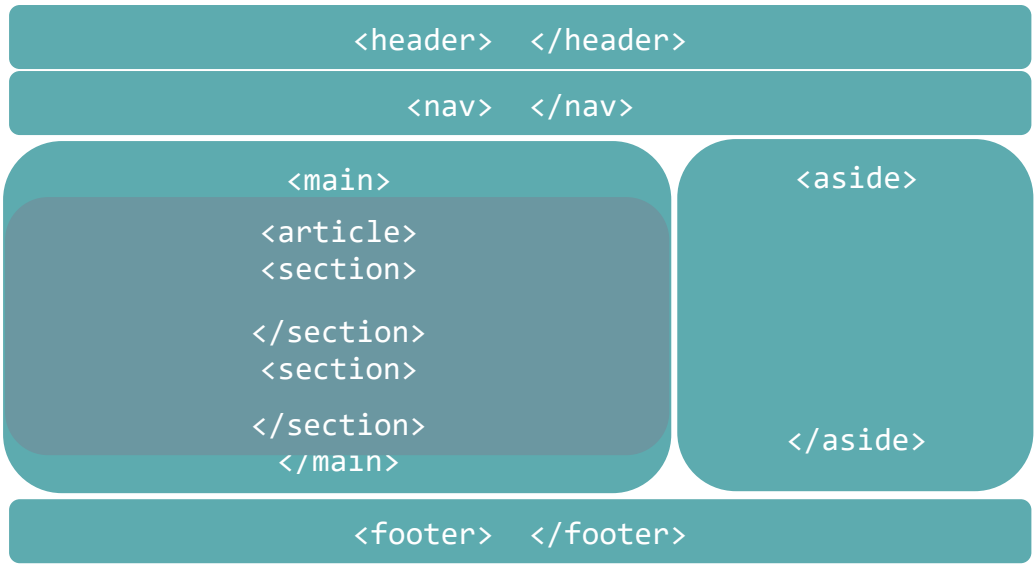


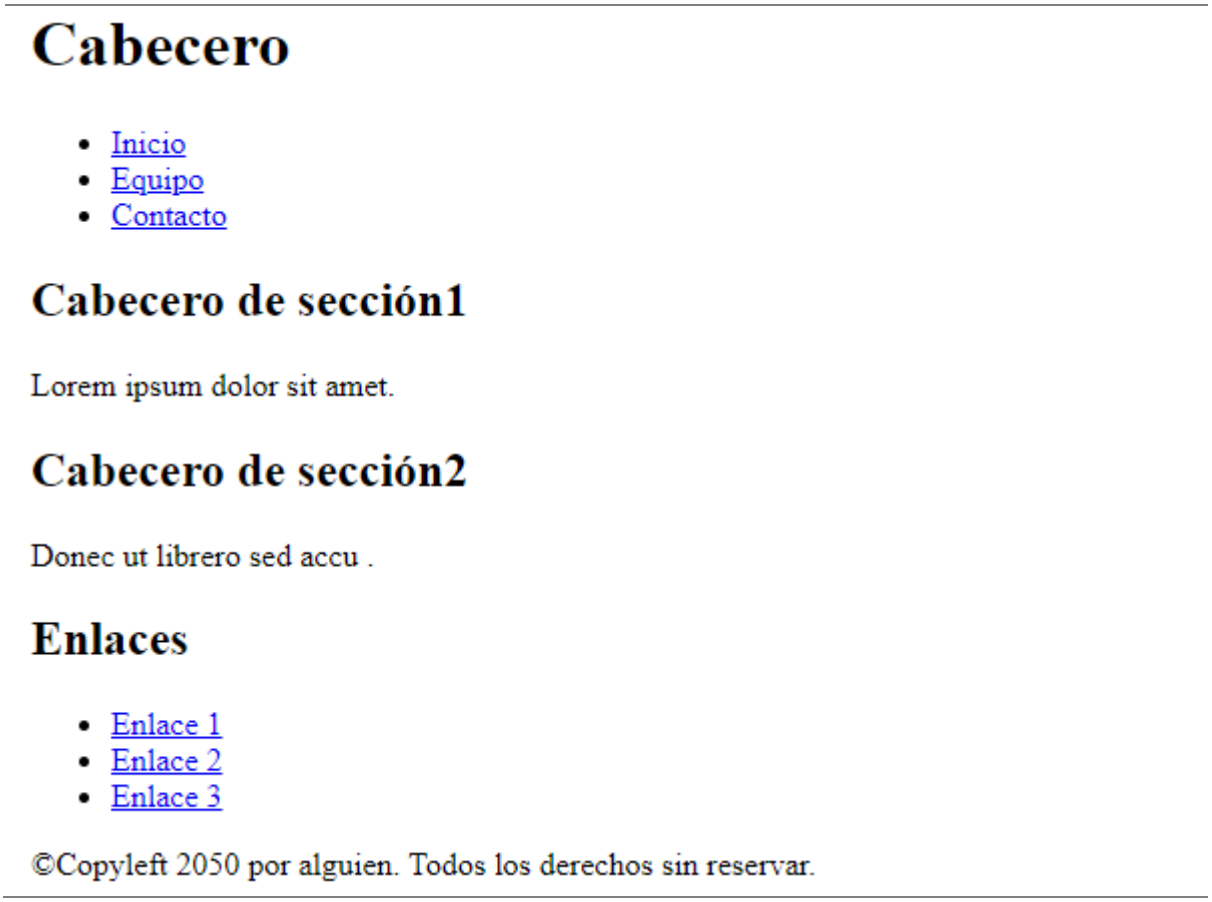
Figura 2-3 Documento con elementos estructurales

Un ejemplo de código correspondiente al diagrama anterior es:

```
<body>
  <header>
    <h1>Cabecero</h1>
  </header>

  <nav>
    <ul>
      <li><a href="#">Inicio</a></li>
      <li><a href="#">Equipo</a></li>
      <li><a href="#">Contacto</a></li>
    </ul>
  </nav>

  <main>
    <article>
      <section>
        <h2>Cabecero de sección1</h2>
        <p>Lorem ipsum dolor sit amet.</p>
      </section>
      <section>
        <h2>Cabecero de sección2</h2>
        <p>Donec ut librero sed accu .</p>
      </section>
    </article>

    <aside>
      <h2>Enlaces</h2>
      <ul>
        <li><a href="#">Enlace 1</a></li>
        <li><a href="#">Enlace 2</a></li>
```

```
        <li><a href="#">Enlace 3</a></li>
      </ul>
    </aside>
  </main>

  <footer>
    <p>&copy;Copyleft 2050 por alguien. Todos los derechos sin reservar.</p>
  </footer>
</body>
```

Los elementos que se encuentren dentro de los elementos de estructura del documento se verán progresivamente en las siguientes secciones. El resultado del código anterior es:



2.2.2 Atributos globales

Los atributos globales son atributos comunes a todos los elementos HTML. La Tabla 4 lista los atributos globales. Los atributos `id` y `class`, que permiten asignar un identificador a los elementos, son usados por algunos elementos para identificar otros elementos y también por reglas CSS y código JavaScript para acceder y modificar elementos específicos en el documento.

Tabla 4 Atributos globales

Atributo	Descripción
accesskey	especifica una tecla de acceso directo para el elemento
autocapitalize	pone el texto en mayúsculas automáticamente a medida que el usuario la introduce
autofocus	atributo booleano que indica que un elemento debe focalizarse en la carga de la página
class	asigna el mismo identificador a un grupo de elementos
contenteditable	indica si el elemento es editable por el usuario
data-*	se usa para almacenar datos personalizados privados para la página o aplicación
dir	atributo enumerado que indica la dirección del texto de los elementos
enterkeyhint	atributo enumerado que define la etiqueta de acción (o icono) a presentar para la tecla Intro en los teclados virtuales
exportparts	permite seleccionar y diseñar elementos existentes en árboles de sombras anidados, exportando sus nombres
hidden	atributo Booleano para ocultar elementos de la página
id	permite definir un identificador único a un elemento el cual no debe repetirse en todo el documento
inert	atributo booleano que le indica al navegador ignorar el elemento
inputmode	atributo enumerado que sugiere el tipo de datos que el usuario puede ingresar mientras edita el elemento o su contenido
is	especifica el comportamiento personalizado de un elemento
itemid	proporciona microdatos en forma de un identificador global único de un elemento
itemprop	proporciona más información al motor de búsqueda acerca de imágenes o datos
itemref	provee una lista de ids de los elementos con propiedades adicionales en otras partes dentro del documento
itemscope	atributo booleano que define el alcance asociado del metadata.
itemtype	especifica la URL del vocabulario que se utilizará para definir itemprop en la estructura de datos
lang	especifica el idioma del contenido del elemento
nonce	atributo de contenido que define un "número utilizado una vez" encriptado que puede ser utilizado por la política de seguridad de contenido
part	contiene una lista separada por espacios de los nombres de las partes del elemento
style	contiene declaraciones de estilo CSS a ser aplicados a un elemento
tabindex	especifica el orden de tabulación de un elemento (uso de tecla Tab)
title	especifica información adicional sobre un elemento

2.2.3 Elementos relativos a texto

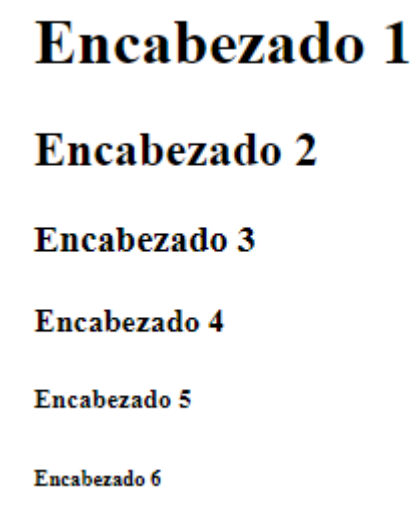
Encabezados

Hay seis niveles de encabezados o títulos: <h1> a <h6>.

La etiqueta <h1> es el encabezado más importante y tiene el tamaño de fuente más grande. Por el contrario, <h6> define el encabezado menos importante y es el más pequeño.

```
<!DOCTYPE html>
<html>
  <head>
    <title>Ejemplo de encabezados</title>
  </head>
  <body>
    <h1>Encabezado 1</h1>
    <h2>Encabezado 2</h2>
    <h3>Encabezado 3</h3>
    <h4>Encabezado 4</h4>
    <h5>Encabezado 5</h5>
    <h6>Encabezado 6</h6>
  </body>
</html>
```

Resultado:



Párrafos

Con el elemento <p> se crean párrafos. Un párrafo siempre comienza en una nueva línea y los navegadores agregan automáticamente un espacio en blanco antes y después de un párrafo.

```
<p>Este es el primer párrafo.</p>
```

Al respecto, se considera que los espacios en blanco (cuando se pulsa la barra espaciadora), los tabuladores y las nuevas líneas se conocen colectivamente como *espacios en blanco*. Cuando un párrafo incluye múltiples espacios en blanco, el elemento <p> automáticamente reduce esos espacios a solo un carácter e ignora el resto. En caso de querer mostrar los espacios en blanco se usa el elemento <pre>. Por ejemplo, el siguiente código produce dos párrafos iguales.

```
<p>
  Este párrafo
  contiene varias lineas
  en el código fuente,
  pero el navegador
  lo ignora.
</p>

<p>
  Este párrafo
  contiene            muchos espacios
  en el código                  fuente,
  pero el         navegador
  los ignora.
</p>
```

Resultado:

Este párrafo contiene varias lineas en el código fuente, pero el navegador lo ignora.

Este párrafo contiene muchos espacios en el código fuente, pero el navegador los ignora.

Texto preformateado

Con el elemento <pre> se muestra el texto en una fuente fija, no proporcional, exactamente como es mostrado en el archivo. Los espacios dentro de este elemento también son mostrados como están escritos. Es útil para mostrar código de programación. Ejemplo:

```
<p>Poema de César Vallejo</p>
<pre>
  Hay golpes en la vida, tan fuertes... ¡Yo no sé!
  Golpes como del odio de Dios; como si ante ellos,
  la resaca de todo lo sufrido
  se empozara en el alma... ¡Yo no sé!
</pre>
```

```
<p>Programa Python cálculo de factorial</p>
<pre>
  def factorial_recursivo(numero):
  if numero &lt;= 1:
      return 1
  return numero * factorial(numero-1)

  valor = 5
  f = factorial(valor)
  print(f"El factorial (calculo recursivo) de {valor} es {f}")
</pre>
```

Resultado:

Poema de César Vallejo

```
Hay golpes en la vida, tan fuertes... ¡Yo no sé!
Golpes como del odio de Dios; como si ante ellos,
la resaca de todo lo sufrido
se empozara en el alma... ¡Yo no sé!
```

Programa Python cálculo de factorial

```
def factorial_recursivo(numero):
if numero <= 1:
    return 1
return numero * factorial(numero-1)

valor = 5
f = factorial(valor)
print(f"El factorial (calculo recursivo) de {valor} es {f}")
```

Caracteres especiales

Para incluir en el texto caracteres que son propias del lenguaje HTML es necesario usar su notación codificada. Una lista completa se encuentra en el siguiente enlace http://dev.w3.org/html5/html-author/charref

Las más comunes son:

	Espacio en blanco	
<	Menor que	&lt;
>	Mayor que	&gt;
&	Ampersand	&amp;
"	Comillas	&quot;
'	Apóstrofo	&apos;
→ ⇒ ↔ ⇔	Flechas	→ ⇒ ↔ ⇔

© ® € ¢ ¥	Símbolos comerciales	© ® € ¢ ¥
× ± ∞ π ∏ σ Σ ω Ω ≥ ≤ ≡ ≈ ⊂ ⊃ ⊆ ⊇ ∈	Símbolos matemáticos	× ± ∞ π Π σ Σ ω Ω ≥ ≤ ≡ ≈ ⊂ ⊃ ⊆ ⊇ ∈
☎ 🔍	Iconos	☎ 🔍
° ~ ✓ ✗		° ˜ ✓ ✗

Elementos de salto

Para forzar o insertar un cambio de línea dentro de elementos de bloque se usa el elemento `<br>`.

Para insertar un salto de cambio temático entre párrafos se usa el elemento `<hr>` que se representa como una línea horizontal.

Ejemplo:

```
<p>Primer párrafo</p>
<hr>
<p>Segundo párrafo</p>

<p>La calidad nunca es un accidente, <br> siempre es el resultado
   de un esfuerzo de la inteligencia</p>
<p>Dos cosas me admiran: la inteligencia de las bestias <br>  y la
        bestialidad de los hombres.</p>
```

Resultados:

Primer párrafo

Segundo párrafo

La calidad nunca es un accidente,
siempre es el resultado de un esfuerzo de la inteligencia

Dos cosas me admiran: la inteligencia de las bestias
y la bestialidad de los hombres.

Elementos para formato de texto

La siguiente lista enumera los elementos en línea, de estilo físico y lógico, para dar formato al texto que contiene.

`<b>` - negrita

<i> - cursiva

<u> - subrayado

<big> - fuente texto grande

<small> - fuente texto pequeño

<sub> - subíndice

<sup> - superíndice

<tt> - teletipo

<mark> - resaltado

 - negrita

 - cursiva

<code> - código (fuente monoespaciada de ancho fijo)

<q> - doble comilla

<ins> - texto insertado

<del> - texto eliminado

<dfn> - definición (negrita o negrita-cursiva)

<cite> - cita (cursiva)

<kbd> - teclado (fuente monoespaciada de ancho fijo)

<samp> - texto muestra (fuente monoespaciada de ancho fijo)

<acronym> - acrónimo (subrayado punteado, con `title` como información)

<abbr> - abreviatura (subrayado punteado, con `title` como información)

<address> - dirección

<var> - variable (ancho fijo o cursiva)

Ejemplo:

```
    <p>Lorem <q>curly quoted</q>,
      sed do <cite>citation</cite> <mark>incididunt</mark> ut quis
      <samp>sample</samp> exercitation <code>code</code> ex <br> ea
      <kbd>keyboard</kbd> consequat. <ins>Insert</ins> occaecat
      <del>delete</del> non <kbd>proident</kbd>,
```

```
    H<sub>2</sub>O polynomio <var>x</var><sup>2</sup>.
  </p>
  <p>Lorem <abbr title="abreviatura">abbr</abbr> dolor <u>sit amet</u>,
    <q>consectetur</q> adipisicing <tt>elit</tt>,
    sed <b>do</b> eiusmod <i>tempor</i>
    <acronym title="Hypertext Markup Language">HTML</acronym> <br>
    ut <big>labore</big>.
    <dfn>HTML</dfn> es <em>realmente</em>,<strong>REALMENTE</strong>
    <small>divertido!</small>.
    <address>mi dirección es un lugar del mundo</address>
  </p>
```

Resultados:

Lorem “curly quoted”, sed do *citation* incididunt ut quis `sample` exercitation `code` ex ea `keyboard` consequat. Insert occaecat ~~delete~~ non `proident`, H_2O polynomio x^2.

Lorem abbr dolor sit amet, “consectetur” adipisicing `elit`, sed **do** eiusmod *tempor* HTML ut labore. *HTML* es *realmente*,**REALMENTE** divertido!.

mi dirección es un lugar del mundo

Listas

Cuando se requiere mostrar la información como una lista de ítems, con viñetas o numeradas, se usan los siguientes elementos:

`<ul>` - elemento para crear una lista de ítems sin orden. Está compuesto por etiquetas de apertura y cierre para agrupar los ítems (`<ul>` y `</ul>`) y trabaja junto con el elemento `<li>` para definir cada uno de los ítems de la lista.

`<ol>`- elemento para crear una lista ordenada de ítems. Está compuesto por etiquetas de apertura y cierre para agrupar los ítems (`<ol>` y `</ol>`) y trabaja junto con el elemento `<li>` para definir los ítems de la lista. Puede incluir los atributos `reversed` para invertir el orden de los indicadores, `start` para determinar el valor desde el cual los indicadores tienen que comenzar a contar y `type` para determinar el tipo de indicador que se desea usar: 1 (números), a (letras minúsculas), A (letras mayúsculas), i (números romanos en minúsculas) e I (números romanos en mayúsculas).

`<dl>` - elemento para crear una lista de términos y descripciones. El elemento trabaja junto con los elementos `<dt>` para definir los términos y `<dd>` para definir las descripciones los ítems de la lista.

Los siguientes elementos se utilizan también para construir listas de ítems.

`<blockquote>` - elemento para representar un bloque de texto mostrado como una cita.

`<details>` - elemento que crea un desplegable que se expande cuando se hace clic en la viñeta para mostrar información adicional. La parte visible se define con el elemento `<summary>` para definir el contenido.

Ejemplos:

```
<ul>
  <li>Sin zapatos</li>
  <li>Sin camiseta</li>
  <li>No hay problema!</li>
</ul>
<!-- Anidamiento de listas -->
<ul>
  <li>Simpsons:
    <ul>
      <li>Homer</li>
      <li>Marge</li>
    </ul>
  </li>
  <li>Familia del individuo:
    <ul>
      <li>Peter</li>
      <li>Lois</li>
    </ul>
  </li>
</ul>

<p>Grados ofertados en la Escuela:</p>
<ol>
  <li>Ingeniería de Tecnologías de Telecomunicación</li>
  <li>Ingeniería en Tecnologías Industriales</li>
  <li>Ingeniería Eléctrica</li>
  <li>Ingeniería Electrónica Industrial y Automática</li>
  <li>Ingeniería Mecánica</li>
  <li>Ingeniería Química</li>
</ol>

<dl>
  <dt>SGML</dt> <dd>Metalenguaje para la definición de otros lenguajes de
marcado</dd>
  <dt>XML</dt> <dd>Lenguaje basado en SGML y que se emplea para describir
datos</dd>
  <dt>XHTML</dt> <dd>Lenguajes derivados de XML para determinadas
aplicaciones</dd>
```

```
  </dl>

<blockquote>
  <p>El poder de la web está en su universalidad.</p>
  <p>El acceso para todos sin importar su discapacidad
     es un aspecto esencial.</p>
</blockquote>

<details>
  <summary>My Books</summary>
    <p>IT</p>
    <p>Carrie</p>
    <p>El Resplandor</p>
    <p>Misery</p>
</details>
```

Resultados:

- Sin zapatos
- Sin camiseta
- No hay problema!

- Simpsons:
  - Homer
  - Marge
- Familia del individuo:
  - Peter
  - Lois

Grados ofertados en la Escuela:

1. Ingeniería de Tecnologías de Telecomunicación
2. Ingeniería en Tecnologías Industriales
3. Ingeniería Eléctrica
4. Ingeniería Electrónica Industrial y Automática
5. Ingeniería Mecánica
6. Ingeniería Química

SGML
: Metalenguaje para la definición de otros lenguajes de marcado

XML
: Lenguaje basado en SGML y que se emplea para describir datos

XHTML
: Lenguajes derivados de XML para determinadas aplicaciones

El poder de la web está en su universalidad.

El acceso para todos sin importar su discapacidad es un aspecto esencial.

▼ My Books

IT

Carrie

El Resplandor

Misery

Enlaces

Los enlaces o hipervínculos son elementos fundamentales del HTML. Un enlace contiene un atributo de URL, que indica la ubicación del recurso y se pueda navegar hasta él.

Para crear un enlace, se usa el elemento `<a>`. El contenido entre las etiquetas de apertura y cierre es el que se muestra en la página web. El elemento incluye el atributo obligatorio `href` para especificar la URL del enlace.

Un enlace permite al usuario:

- navegar a un documento diferente.
- navegar a un "Punto de anclaje" (o marcador) en el documento actual o en otro documento anteponiendo al nombre el carácter #, o
- solicitar otros recursos web (como el correo electrónico).

El elemento `<a>...</a>` también se puede utilizar para configurar un "nombre de punto de anclaje" (o marcador) dentro de un documento, para que sea el objetivo de otros hipervínculos con el atributo `name`. Para esta función es mejor usar el atributo id de los elementos.

Enlaces externos: Los enlaces externos llevarán al usuario a otra página web. Esta página puede pertenecer al mismo sitio web, o puede ser a otro sitio web. El atributo `href` especifica la URL del enlace.

Enlaces internos: Los enlaces internos llevarán al usuario a navegar dentro de una página web en función del atributo `name` del elemento `<a>` o el atributo `id` de los elementos.

Ejemplo de estos enlaces:

```
<p>
  Ejemplo de enlaces externos: <br />
  <a href="index.html">Principal</a> |
  <a href="/">Principal</a> |
  <a href="fotos.html">Fotos</a> |
  <a href="videos.html">Videos</a> |
  <a href="html/contacto.html">Contacto</a> |
  <a href="https://www.google.com/">Google</a>
</p>

<p>
  Ejemplo de enlaces internos: <br>
  <a href="#nombre_marcador1">Titulo 1</a> |
  <a href="#nombre_marcador2">Titulo 2</a> |
  <a href="#nombre_marcador3">Titulo 3</a> |
  <a name="nombre_marcador_arriba"></a>
  <h3>Marcadores en HTML</h3>
  <a name="nombre_marcador1"></a>
  <h4>Titulo 1</h4>
  <p>Lorem ipsum dolor sit amet consectetur. </p>

  <a name="nombre_marcador2"></a>
```

```
   <h4>Titulo 2</h4>
   <p>Sed ut perspiciatis unde omnis iste natus. </p>

   <h4 id="nombre_marcador3">Titulo 3</h4>
   <p>Unde accusantium doloremque laudantium.</p>
   <a href="#nombre_marcador_arriba">Ir arriba</a>
 </p>
```

Resultado:

Ejemplo de enlaces externos:
Principal | Principal | Fotos | Videos | Contacto | Google

Ejemplo de enlaces internos:
Titulo 1 | Titulo 2 | Titulo 3 |

Marcadores en HTML

Titulo 1

Lorem ipsum dolor sit amet consectetur.

Titulo 2

Sed ut perspiciatis unde omnis iste natus.

Titulo 3

Unde accusantium doloremque laudantium.

Ir arriba

La etiqueta `<a>` tiene un atributo `target="targetName"` para configurar la ventana/marco de destino del enlace. Con este tributo, en lugar de mostrar la página de destino a la que apunta `href` en la misma ventana del navegador, se puede valor del atributo para mostrar la nueva página en otra ventana. Los valores posibles son:

- `target="_blank"`: abre el documento vinculado en una nueva pestaña o ventana.

- `target="_self"` (predeterminado): abre el documento vinculado en la misma ventana/marco.
- `target="_parent"`: abre el documento vinculado en el marco principal.
- `target="_top"`: abre el documento vinculado en el cuerpo completo de la ventana.
- `target="frame-name"`: abre el documento vinculado en el marco nombrado.

Otro atributo interesante de la etiqueta `<a>` es `download` que especifica que el archivo de destino (especificado en el atributo " href ") se descargará cuando los usuarios hagan clic en el hipervínculo. Se puede declarar opcionalmente un nuevo nombre para el archivo estableciendo el valor del atributo `download`. Además, si no se especifica la extensión del archivo, el navegador lo detecta automáticamente y lo agrega al archivo.

```
<a href="images/html5.png" download="imagen_html5">Descarga de imagen</a>
```

Resultado:

Tablas

Las tablas son útiles para mostrar grandes conjuntos de datos en formato fila-columna.

Los elementos más utilizados para crear una tabla son los siguientes:

<table> - elemento para definir una tabla. Incluye etiquetas de apertura y cierre para agrupar el resto de los elementos que definen la tabla.

<tr> - elemento para definir una fila de celdas. Incluye etiquetas de apertura y cierre para agrupar las celdas.

<td>- elemento para definir una celda. Incluye etiquetas de apertura y cierre para delimitar el contenido de la celda y puede incluir los atributos colspan y rowspan para indicar cuántas columnas y filas ocupa la celda.

<th>- elemento para definir una celda para la cabecera de la tabla. Incluye etiquetas de apertura y cierre para delimitar el contenido de la celda y puede incluir los atributos colspan y rowspan para indicar cuántas columnas y filas ocupa la celda.

<caption> - elemento para definir un cabecero de la tabla.

Para insertar una tabla en el documento, en primer lugar, se declara el elemento <table> y luego se describen las filas una por una con los elementos <tr> y <td>. Ejemplo:

```
<table>
  <caption>Tabla de ejemplo</caption>
  <tr><th>Columna 1</th><th>Columna 2</th><th>Columna 3</th></tr>
  <tr>
    <td colspan="2">Celdas 1,1-1,2</td>
    <td rowspan="3">Celdas 3-1,3-2,3-3</td>
  </tr>
  <tr><td>Celda 2,1</td><td>Celda 2,2</td></tr>
  <tr><td>Celda 3,1</td><td>Celda 3,2</td></tr>
</table>
```

Resultado:

Tabla de ejemplo

Columna 1	Columna 2	Columna 3
Celdas 1,1-1,2		
Celda 2,1	Celda 2,2	Celdas 3-1,3-2,3-3
Celda 3,1	Celda 3,2	

2.2.4 Elementos multimedia

Imágenes

La capacidad de mostrar imágenes en las páginas web es otro medio importante en presentar la información. Los formatos de archivos de imagen más comunes utilizados en la web se resumen en la Tabla 5.

Tabla 5 Tipos MIME más comunes

Abrev.	Formato fichero	Tipo MIME image/xx	Extensión	Característica
APNG	Animated Portable Network Graphics	apng	.apng	Adecuado para animación sin pérdida
AVIF	AV1 Image File Format	avif	.avif	Bueno para imágenes y animación
GIF	Graphics Interchange Format	gif	.gif	Bueno para imágenes y animación simple
JPEG	Joint Photographic Expert Group image	jpeg	.jpeg, .jpg, jfif, .pjpeg, .pjp	Bueno imágenes fijas comprimidas (más popular)
PNG	Portable Network Graphics	png	.png	Mejor que JPEG y cuando se necesita transparencia.
SVG	Scalable Vector Graphics	svg+xml	.svg	Formato vectorial; ideal para dibujar elementos en diferentes tamaños.
WebP	Web Picture format	webp	.webp	Excelente para imágenes fijas y animadas.

HTML dispone de los siguientes elementos para introducir imágenes en las páginas web:

`<img>` - no tiene etiqueta de cierre por no tener contenido. Requiere del atributo `src` para especificar la URL del archivo con la imagen que se desea mostrar. Es un elemento de tipo inline.

`<figure>` - es un elemento que incluye los elementos `<img>` para las imágenes y `<figcaption>` para introducir un título.

`<picture>` - requiere el uso del elemento `<source>` para ofrecer múltiples imágenes en diferentes resoluciones. Es útil para crear sitios web adaptables.

Para insertar una imagen en una página web de forma simple solo es necesario usar el elemento `<img>` y asignar la URL del archivo al atributo `src`.

```
<p>
  <img src="images/html5css3.jpg">
</p>
```

Resultado:

La imagen se representa en su tamaño original. Para definir un tamaño personalizado y otros parámetros de configuración se pueden usar otros atributos disponibles:

`width` - atributo para asignar el ancho de la imagen.

`height` - atributo para asignar la altura de la imagen.

`alt` - atributo para especificar el texto que se muestra cuando la imagen no se puede cargar. Su uso es obligatorio para accesibilidad.

`title` - atributo global para especificar texto cuando el cursor se encuentra sobre el elemento, en este caso la imagen.

`style` - atributo global para especificar propiedades de estilo a la imagen, por ejemplo, con `style="width:30%;"` se consigue que el ancho de la imagen se adapte al 30% del ancho de la ventana.

Los atributos `width` y `height` asignan las dimensiones de la imagen sin tener en cuenta la relación, es decir, puede deformarse. Se puede especificar solo uno de los atributos y dejar que el navegador calcule el otro.

Con el elemento en bloque `<figure>` se representa contenido autónomo que puede ser un elemento `<img>` al que se añade un título opcional mediante el elemento `<figcaption>`.

Con el elemento `<picture>` se puede seleccionar la imagen a mostrar según condiciones incluídas en uno o más elementos `<source>` para especificar las

posibles fuentes de la imagen y un elemento <img> al final para mostrar la imagen seleccionada en pantalla.

Ejemplo:

```
<figure>
  <img src="images/html5css3.jpg" alt="Prueba de figure" style="width:30%">
  <figcaption>Logos de HTML CSS</figcaption>
</figure>

<picture>
  <source media="(max-width: 1024px)" srcset="html/images/gollum.jpg">
  <img src="html/images/gandalf.jpg">
</picture>
```

Resultados: (imagen izquierda cuando el ancho de la ventana es menor a 1024px, imagen derecha mayor a 1024px)

Logos de HTML CSS

Logos de HTML CSS

Audio

Los formatos de archivos de audio más comunes utilizados en la web se resumen en la Tabla 6.

Tabla 6 Tipos MIME de audio más comunes

Abrev.	Formato fichero	Tipo MIME image/xx	Extensión
AAC	Advanced Audio Coding	audio/aac	.aac
MIDI	Musical Instrument Digital Interface	audio/midi, audio/x-midi	.mid, .midi

MP3	MP3 audio	audio/mpeg	.mp3
OGG	Ogg Vorbis	audio/ogg	.oga
WAV	Waveform Audio Format	audio/wav	.wav

HTML5 dispone de un elemento para reproducir archivos de audio.

`<audio>` - elemento para insertar un fichero audio en el documento.

Tiene los siguientes atributos:

`src` - atributo para especificar la URL del archivo a reproducir.

`controls` - atributo booleano. Si está presente el navegador ofrecerá controles para permitir que el usuario controle la reproducción de audio, incluyendo volumen, búsqueda y pausar/reanudar reproducción.

`autoplay` - atributo booleano. Si está presente, el navegador reproduce el audio automáticamente tan pronto como puede.

`loop` - atributo booleano. Si está presente, el navegador reproduce el audio en bucle.

`preload` - atributo que determina si el navegador debe comenzar a cargar el archivo de audio antes de reproducirse. Acepta tres valores:

`none` indica que el audio no se debería cargar y generalmente se utiliza para minimizar tráfico web.

`metadata` recomienda al navegador que descargue información acerca del recurso, como la duración del audio.

`auto` solicita al navegador que descargue el archivo tan pronto como le sea posible (valor por defecto).

El elemento `<audio>` requiere etiquetas de apertura y cierre, y especificar los atributos `src` y `controls` para mostrar los controles de reproducción. Ejemplo:

```
<audio src="media/horse.mp3" controls>
</audio>
```

Resultado:

En caso de querer reproducir un sonido de fondo sin mostrar nada en la página se usaría los atributos autoplay y loop. Ejemplo:

```
<audio src="media/horse.mp3" autoplay loop>
</audio>
```

Para la reproducción de audio los navegadores usan codificadores (códecs) y puede que no sean compatibles con un determinado formato especificado. En ese caso, es recomendable usar el elemento `<source>` para facilitar al menos dos formatos para que el navegador pueda elegir. Ejemplo:

```
<audio controls>
  <source src="media/gallo.mp3">
  <source src="media/gallo.ogg">
  <source src="media/gallo.wav">
</audio>
```

Video

Los videos son métodos muy eficaces para la comunicación, siendo actualmente el medio más utilizado en la Web por la facilidad de generar videos con los dispositivos móviles.

Los formatos de archivos de video más comunes utilizados en la web se resumen en la Tabla 7.

Tabla 7 Tipos MIME de audio más comunes

Abrev.	Formato fichero	Tipo MIME image/xx	Extensión
AVI	Audio Video Interleave	video/x-msvideo	.avi
MP4	Moving Picture Experts Group	video/mp4	.mp4
OGG	Ogg Vorbis	video/ogg	.ogv
WEBM	WEBM video	video/webm	.webm
QTFF	QuickTime File Format	video/quicktime	.mov

HTML5 dispone de un elemento para reproducir archivos de video.

`<video>` - elemento para insertar un video en el documento.

Tiene los siguientes atributos:

`src` - atributo para especificar la URL del video a reproducir.

`width` - atributo para asignar el ancho del área de visualización del video en pixels.

`height` - atributo para asignar la altura del área de visualización del video en pixels.

`controls` - atributo booleano. Si está presente el navegador ofrecerá controles para permitir que el usuario controle la reproducción de video, incluyendo volumen, búsqueda y pausar/reanudar reproducción.

`autoplay` - atributo booleano. Si está presente, el video comenzará a reproducirse automáticamente tan pronto como sea posible, sin detenerse para terminar de cargar los datos.

`poster` - atributo que especifica la URL de la imagen que se mostrará mientras el navegador espera a que el vídeo se reproduzca.

`loop` - atributo booleano. Si está presente, el navegador reproduce el video en bucle.

`preload` - atributo que determina si el navegador debe comenzar a cargar el archivo de video antes de reproducirse. Acepta tres valores:

`none` indica que el video no se debe almacenar en caché, por lo que se usa generalmente para minimizar tráfico web,

`metadata` recomienda al navegador que descargue información acerca del recurso, como la duración del video.

`auto` solicita al navegador que descargue el video completo tan pronto como le sea posible (valor por defecto).

El elemento `<video>` requiere etiquetas de apertura y cierre, y requiere especificar el atributo `src` y los métodos de reproducción con los atributos `controls` y `autoplay`. Un ejemplo completo con la mayoría de atributos es:

```
<video src="media/windtunnel.mp4" width="720" height="400" preload controls
loop poster="images/gandalf.jpg">
</video>
```

Resultado:

Como ocurre con el elemento audio para reproducir un video los navegadores usan codificadores (códecs) y puede existir algún problema de compatibilidad con un determinado formato especificado. Por ello, es recomendable usar el elemento `<source>` para facilitar al menos dos formatos para que el navegador pueda elegir.

Inclusión de video de portales (YouTube, TED talks)

Se puede insertar videos desde cualquier portal que ofrezca la posibilidad de generar el elemento `iframe` e incorporarlo a la página.

En caso que los servidores no tengan capacidad o no permitan el uso de ficheros multimedia, se puede subir el recurso a YouTube y tomar nota de su identificador o copiar el código de inserción iframe que facilita YouTube. Por ejemplo:

```
<iframe width="424" height="238"
src="https://www.youtube.com/embed/_smFO9k9L2I" title="Universidad de
Cantabria. ¡Mucho más que un título!" frameborder="0"
allow="accelerometer; autoplay; clipboard-write; encrypted-media;
gyroscope; picture-in-picture; web-share" allowfullscreen></iframe>
```

Subtítulos

HTML ofrece la posibilidad de introducir subtítulos mientras se reproduce el audio o vídeo a través del uso del elemento <track>. El elemento tiene que incluirse dentro de un elemento <video> o <audio>. Para especificar la fuente, el tipo y el modo en el que se mostrarán los subtítulos en la pantalla, el elemento `<track>` ofrece los siguientes atributos:

`src` - atributo para declarar la URL del archivo que contiene el texto de los subtítulos. El formato oficial de este archivo es WebVTT.

`srclang` - atributo para declarar el idioma del texto similar al atributo `lang` del elemento `<html>`.

`default` - atributo para declarar la pista (track) que se quiere incluir por defecto. Si solo se facilita un elemento `<track>`, este atributo se puede usar para activar los subtítulos.

`label` - atributo para facilitar un título para la pista. Si se incluyen varios elementos `<track>`, este atributo ayuda al usuario a encontrar el correcto.

`kind` - atributo para declarar el tipo de contenido asignado a una pista. Los valores disponibles son `subtitles` (para subtítulos), `captions` (para subtítulos que representan sonido), `descriptions` (destinado a sintetizado de audio), `chapters` (para navegación entre capítulos) y `metadata` (para información adicional que no se muestra en la pantalla). El valor por defecto es subtitles (subtítulos).

Respecto al formato WebVTT (Web Video Text Tracks), es un formato estándar para subtítulos. Los archivos de este formato son de tipo texto con una estructura especial. La primera línea con el texto "WEBVTT" es obligatoria, así como las líneas vacías entre cada declaración. Las declaraciones se llaman cues (señales), y requieren la sintaxis minutos:segundos.milisegundos para indicar el tiempo de inicio y finalización en el que aparecerán utilizando un formato para el número de dígitos (dos para minutos, dos para segundos y tres para milisegundos).

El formato WebVTT también ofrece la posibilidad de alinear y posicionar cada cue usando los siguientes parámetros y valores.

`align` - para alinear la cue en relación al centro del espacio que cubre el medio. Los valores disponibles son start, middle, y end.

`vertical` - para cambiar la orientación a vertical y ordena la cue de acuerdo a

dos valores: `rl` (derecha a izquierda) o `lr` (izquierda a derecha).

`position` - para declarar la posición de la cue en columnas. El valor se puede expresar como un porcentaje o como un número de 0 a 9. La posición se declara de acuerdo a la orientación.

`line` - para declarar la posición de la cue en filas. El valor se puede expresar en porcentaje o como un número de 0 a 9. En una orientación horizontal, la posición que se declara es vertical, y viceversa. Los números positivos declaran la posición desde un lado y los números negativos desde el otro, dependiendo de la orientación.

`size` - valor para declarar el tamaño de la cue. El valor se puede declarar en porcentaje y se determina a partir del ancho del medio.

Estos parámetros y sus correspondientes valores se declaran al final de la cue separados por dos puntos pudiéndose hacer varias declaraciones para la misma cue.

Un ejemplo de especificación del elemento `<video>` que usa el elemento `<track>` para mostrar subtítulos y el fichero .vtt necesario se muestra a continuación:

```
<video src="media/windtunnel.mp4" controls>
  <track src="media/subtitulo.vtt" srclang="es" default>
</video>
```

```
Subtitulo.vtt
```

```
WEBVTT

00:00.000 --> 00:05.000 align:start position:5%
Bienvenido
al elemento &lt;track&gt;!

00:05.000 --> 00:10.000
Este es un ejemplo simple.

00:10.000 --> 00:15.000
Se pueden usar varias simultaneamente

00:15.000 --> 00:20.000
para ofrecer textos en diferentes lenguajes.

00:20.000 --> 00:30.000
Hasta luego!
```

Elemento Canvas

HTML dispone del elemento <canvas> que permite dibujar sobre una región de la página web utilizando un programa en JavaScript. Para ello requiere del elemento <canvas> con los atributos width y head para definir el tamaño de la región para el dibujo, así como el atributo id para que sea referido por el programa JavaScript.

El siguiente es un ejemplo:

```
<!DOCTYPE html>
<html lang="es">
  <head>
    <meta charset="utf-8" />
    <title>Dibujo de formas primitivas Canvas</title>
    <script>
      window.addEventListener("load", draw, true);
      function draw() {
        // Obtiene el elemento canvas y dimensiona
        let canvas = document.getElementById("canvas");
        canvas.width = 500;
        canvas.height = 300;
        // Obtiene el contexto grafico 2D para canvas
        let context = canvas.getContext("2d");
        // Dibuja un Rectangulo usando la primitiva rect
        context.fillStyle = "rgb(0, 255, 0)";
        context.fillRect(30, 80, 100, 120); // x, y, width, height
        context.strokeStyle = "#0000FF";
        context.strokeRect(30, 80, 100, 120);
        // Dibuja un triangulo usando path
        context.beginPath();
        context.moveTo(150, 50);
        context.lineTo(200, 180);
        context.lineTo(100, 180);
        context.closePath();
        context.fillStyle = "rgba(0, 0, 255, 0.5)"; // transparente
        context.fill();
        context.stroke();
        // Rellena un circulo usando path
        context.beginPath();
        // centerX, centerY, radius, beginAngle, engAngle, antiClockwise
        context.arc(220, 180, 50, 0, Math.PI * 2, true);
        context.fillStyle = "red";
        context.fill(); // closePath() implicito
      }
    </script>
  </head>
  <body>
    <h2>Dibujo de formas primitivas</h2>
    <canvas id="canvas">Canvas no es soportado</canvas>
```

```
  </body>
</html>
```

Resultado:

2.3 FORMULARIOS

Un formulario HTML o formulario web es un grupo de controles de interface de usuario que permiten a un usuario seleccionar o insertar información y enviarla al servidor web para ser procesada. En el servidor web, un programa del lado del servidor recopila estos datos y devuelve una respuesta dinámica basada en las entradas enviadas. También sirven como interfaz de usuario para programas JavaScript que se ejecutan en el lado cliente, es decir en el propio navegador, para crear aplicaciones interactivas.

Sintaxis de formularios

Un formulario queda definido por el elemento `<form>` que incluye etiquetas de apertura y cierre para agrupar al resto de los elementos y requiere de algunos atributos para determinar cómo se envía la información al servidor. Los atributos de `<form>` son:

`name` - atributo que especifica el nombre del formulario, para estar disponible para otros elementos.

`method` - atributo que determina el método a utilizar para enviar la información al servidor. Existen dos valores disponibles: `get` y `post`. El método `get` se usa para enviar una cantidad limitada de información de forma pública (los datos son incluidos en la URL y no puede contener más de 255 caracteres). El método

`post` se utiliza para enviar una cantidad ilimitada de información de forma privada (los datos no son visibles y pueden tener la longitud necesaria).

`action` - atributo para declarar la URL del archivo en el servidor que va a procesar la información enviada por el formulario.

`target` - atributo que determina dónde se mostrará la respuesta recibida desde el servidor. Los valores disponibles son `_blank` (nueva ventana), `_self` (mismo recuadro), `_parent` (recuadro padre), y `_top` (la ventana que contiene el recuadro). El valor `_self` se usa por defecto, por lo que la respuesta recibida desde el servidor se mostrará en la misma ventana.

`enctype` - atributo que declara la codificación aplicada a los datos que envía el formulario. Puede tomar tres valores: `application/x-www-form-urlencoded` (los caracteres son codificados), `multipart/form-data` (los caracteres no son codificados), `text/plain` (solo los espacios son codificados). El primer valor se asigna por defecto.

`accept-charset` - atributo que declara el tipo de codificación aplicada al texto del formulario. Los valores más comunes son UTF-8 e ISO-8859-1. El valor por defecto se asigna al documento con el elemento `<meta>`.

Por ejemplo, un formulario simple para hacer una búsqueda haciendo uso del buscador de Google es:

```
<form action="http://www.google.com/search">
  <p>
    Escribir lo que sea desea buscar en Google:
    <input name="q">
    <input type="submit">
  </p>
</form>
```

Elementos de formularios

Un formulario puede incluir diferentes elementos para permitir al usuario seleccionar o insertar información. HTML incluye varios elementos que se muestran en la Tabla 8. Los más utilizados son:

`<input>` - elemento que crea un campo de entrada. Hay diferentes tipos de entradas dependiendo del valor del atributo `type`.

<textarea> - elemento que crea un campo de entrada para insertar múltiples líneas de texto. El tamaño se puede declarar en números enteros usando los atributos rows y cols.

<select> - elemento que crea una lista de opciones que el usuario puede elegir. Funciona junto con el elemento <option> para definir cada opción y el elemento <optgroup> para organizar las opciones en grupos.

<button> - elemento que crea un botón. El atributo type define el propósito del botón. Los valores disponibles son submit para enviar el formulario (por defecto), reset para reiniciar el formulario, y button para realizar tareas personalizadas.

<output> - elemento que representa un resultado producido por el formulario. Se implementa por medio de código JavaScript para mostrar el resultado de una operación al usuario.

<meter> - elemento que representa una medida o el valor actual de un rango.

<progress> - elemento que representa el progreso de una operación.

<datalist> - elemento que crea un listado de valores disponibles para otros controles. Trabaja junto con el elemento <option> para definir cada valor.

<label> - elemento que crea una etiqueta para identificar un elemento de formulario.

<fieldset> - elemento que agrupa otros elementos de formulario. Se usa para crear secciones dentro de formularios extensos. El elemento puede contener un elemento <legend> para definir el título de la sección.

El elemento <input> genera un campo de entrada en el que el usuario puede seleccionar o insertar información, pero puede adoptar diferentes características y aceptar varios tipos de valores dependiendo del valor de su atributo type. Los valores disponibles para este atributo.

Cada campo <input> suele estar asociado con un <label> para etiquetar el campo de entrada. Se puede vincular un elemento <input> con una etiqueta en cualquiera de las formas:

<label> como envolvente: se coloca el elemento <input> dentro de las etiquetas <label>...</label>.

<label> como referencia: se hace coincidir el atributo for del elemento <label> con el atributo id del elemento input.

La etiqueta no tiene ningún efecto visual. Pero si hace clic en la etiqueta, se seleccionará el campo input asociado.

Un formulario bien diseñado usa el elemento <fieldset> que permite agrupar los campos de entrada en conjuntos, con un elemento <legend> para proporcionar una leyenda descriptiva para el conjunto de campos.

HTML define atributos globales que son exclusivos de elementos de formulario. Los siguientes son los más utilizados.

disabled - atributo booleano que desactiva el elemento. Cuando el atributo está presente, el usuario no puede introducir valores o interactuar con el elemento.

readonly - atributo para indicar que el valor del elemento no se puede modificar.

placeholder - atributo que muestra un texto en el fondo del elemento que indica al usuario el valor que debe introducir.

autocomplete - atributo que activa o desactiva la función de autocompletar. Los valores disponibles son on y off.

novalidate - atributo booleano para el elemento <form> que indica que el formulario no debería ser validado.

formnovalidate - atributo booleano para los elementos <button> e <input> de tipo submit e image que indica que el formulario al que pertenecen no debería ser validado.

required - atributo booleano que indica al navegador que el usuario debe seleccionar o insertar un valor en el elemento para validar el formulario.

multiple - atributo booleano que indica al navegador que se pueden insertar múltiples valores en el campo (se aplica a elementos <input> de tipo email y file).

autofocus - atributo booleano que solicita al navegador que mueva el foco al elemento tan pronto como se carga el documento.

`pattern` - atributo que define una expresión regular que el navegador debe usar para validar el valor insertado en el campo.

`form` - atributo que asocia el elemento con un formulario. Se usa para conectar un elemento con un formulario cuando el elemento no se define entre las etiquetas `<form>`. El valor asignado a este atributo debe ser el mismo asignado al atributo id del elemento `<form>`.

`spellcheck` - atributo que solicita al navegador que compruebe la ortografía y gramática del valor introducido en el campo. Los valores disponibles son true y false.

Tabla 8 Elementos de formularios

Elemento	Control	Descripción
`<input type="text">` (predeterminado)	Entrada de texto de una sola línea	Entrada texto. Tipo predeterminado para <input>
`<input type="password">`	Entrada de contraseña de una sola línea	Contraseña mostrada en asterisco (*)
`<textarea>...</textarea>`	Entrada de texto de varias líneas	Para entrada de texto
`<input type="checkbox">`	Cajetín selección	Puede marcar cero o más casillas
`<input type="radio">`	Botón radio	Puede verificar como máximo 1 botón

`<select>...</select>`
`<optgroup>...</optgroup>`
`<option>...</option>`
	Lista desplegable	

`<input type="submit">`	Botón de enviar	Hacer clic para enviar el formulario al servidor
`<input type="reset">`	Botón de reinicio	Restablecer todos los campos a su valor predeterminado
`<input type="image">`	Botón de imagen	Usa una imagen como botón de envío
`<input type="button">`	Botón genérico	

`<input type="file">`	Selector de ficheros	
`<input type="hidden">`	Campo escondido	
`<button>...</button>`	Botón	Igual que `<input type="button">`
`<input type="email">`	Dirección Email	Se puede validar
`<input type="url">`	URL	Se puede validar
`<input type="number">`	Valor numérico	Puede usar atributos `min|max` para especificar los valores mínimo y máximo; y `step` para especificar el tamaño del paso (predeterminado de 1)
`<input type="range">`	Valor numérico	Puede usar atributos `min|max` para especificar los valores mínimo y máximo. Muchos navegadores usan un *slider* para `range`
`<input type="color">`	Selector de color	El navegador muestra un selector de color

`<input type="datetime_local">`
`<input type="datetime">`
`<input type="date|time|month|week">`
	Fecha y hora	El navegador proporciona un calendario desplegable

`<input type="tel">`	Número de teléfono	Sin validación al variar el número de teléfono según el país
`<input type="search">`	Buscar palabras clave	Cuadro de búsqueda para introducir palabras clave de búsqueda. Sin validación
`<datalist>..</datalist>` `<option>`	Lista sugerida para la entrada	Define una lista de opciones "sugeridas" para `<input type="text">`
`<output>..</ouput>`	Salida generada	Define el resultado de un cálculo (mediante JavaScript)
`<keygen>`		Define campo de par de claves pública-privada para la autenticación del formulario
`<progress>..</progress>`		Muestra la progresión de una tarea
`<meter>...</meter>`		Muestra el valor en curso (en un rango conocido) o porcentaje

El siguiente código muestra el uso de varios tipos de elementos para formularios:

```
<p>Ejemplo de elementos de formulario:</p>
<p><input type="text" name="nombre"></p>
<p><input type="text" name="edad" value="35"></p>
<p><label>Nombre: <input type="text" name="nombre"
maxlength="15"></label></p>
<p><label>Edad: <input type="number" name="edad" min="13"
max="100"></label></p>
<p><label>Edad: <input type="range" name="edad" min="13" max="100"
value="35"></label></p>
<p><label>Correo: <input type="email" name="correo"></label></p>
<p><label>Teléfono: <input type="tel" name="telefono"></label></p>
<p><label>Sitio Web: <input type="url" name="sitioweb"></label></p>
<p><label><input type="radio" name="genero" value="m">Masculino</label></p>
<p><label><input type="radio" name="genero" value="f">Femenino</label></p>
<p><label><input type="radio" name="edad" value="15" checked> 15
Años</label></p>
<p><label><input type="radio" name="edad" value="30"> 30 Años</label></p>
<p><label><input type="radio" name="edad" value="45"> 45 Años</label></p>
<p><label><input type="radio" name="edad" value="60"> 60 Años</label></p>
<!-- Emparejar "for" con "id" -->
<label for="username">Nonmbre</label>
<input type="text" id="username" name="username">
<p><input type="text" size="10" maxlength="8"> NetID <br></p>
<p><input type="password" size="16"> Password</p>
<p><label>Texto: <textarea name="texto" cols="50"
rows="6"></textarea></label></p>
<p><input type="checkbox" name="lechuga"> Lechuga
  <input type="checkbox" name="tomate" checked="checked"> Tomate
  <input type="checkbox" name="pepino" checked="checked"> Pepino
</p>
<p><input type="radio" name="cc" value="visa" checked="checked"> Visa<br>
  <input type="radio" name="cc" value="mastercard"> MasterCard<br>
  <input type="radio" name="cc" value="amex"> American Express<br>
  <label><input type="radio" name="cc" value="visa" checked="checked">
Visa</label><br>
  <label><input type="radio" name="cc" value="mastercard">
MasterCard</label><br>
  <label><input type="radio" name="cc" value="amex"> American
Express</label>
</p>
<select name="favoritecharacter">
  <option>Jerry</option>
  <option>George</option>
  <option selected="selected">Kramer</option>
  <option>Elaine</option>
</select>
<select name="favoritecharacter[]" size="3" multiple="multiple">
  <option>Jerry</option>
  <option>George</option>
  <option>Kramer</option>
  <option>Elaine</option>
```

```
  <option selected="selected">Newman</option>
</select>
<select name="favoritecharacter">
  <optgroup label="Major Characters">
    <option>Jerry</option>
    <option>George</option>
    <option>Kramer</option>
    <option>Elaine</option>
  </optgroup>
  <optgroup label="Minor Characters">
    <option>Newman</option>
    <option>Susan</option>
  </optgroup>
</select>
<datalist id="datos">
  <option value="123123123" label="Teléfono 1">
  <option value="456456456" label="Teléfono 2">
</datalist>
<p><label>Teléfono: <input type="tel" name="telefono"
list="datos"></label></p>
<p>
  Name: <input type="text" name="name"> <br>
  Food: <input type="text" name="meal" value="pizza"> <br>
  <label>Meat? <input type="checkbox" name="meat"></label> <br>
  <input type="reset">
</p>
<fieldset>
  <legend>Tarjetas de crédito:</legend>
  <input type="radio" name="cc" value="visa" checked="checked"> Visa
  <input type="radio" name="cc" value="mastercard"> MasterCard
  <input type="radio" name="cc" value="amex"> American Express
</fieldset>
<p><input type="date" name="fecha" value="2023-01-01"></p>
<p><input type="submit" value="Enviar Datos"></p>
<p><input type="color" name="micolor" value="#99BB00"></p>
<p><input type="image" src="images/botonenviar.png" width="100"></p>
<progress value="30" max="100">0%</progress>
<meter value="60" min="0" max="100" low="40" high="80"
optimum="100">60</meter>
<p><button type="submit">Enviar Formulario</button></p>
```

Resultado:

Ejemplo de elementos de formulario:

35

Nombre:

Edad:

Edad:

Correo:

Teléfono:

Sitio Web:

Masculino

Femenino

15 Años

30 Años

45 Años

60 Años

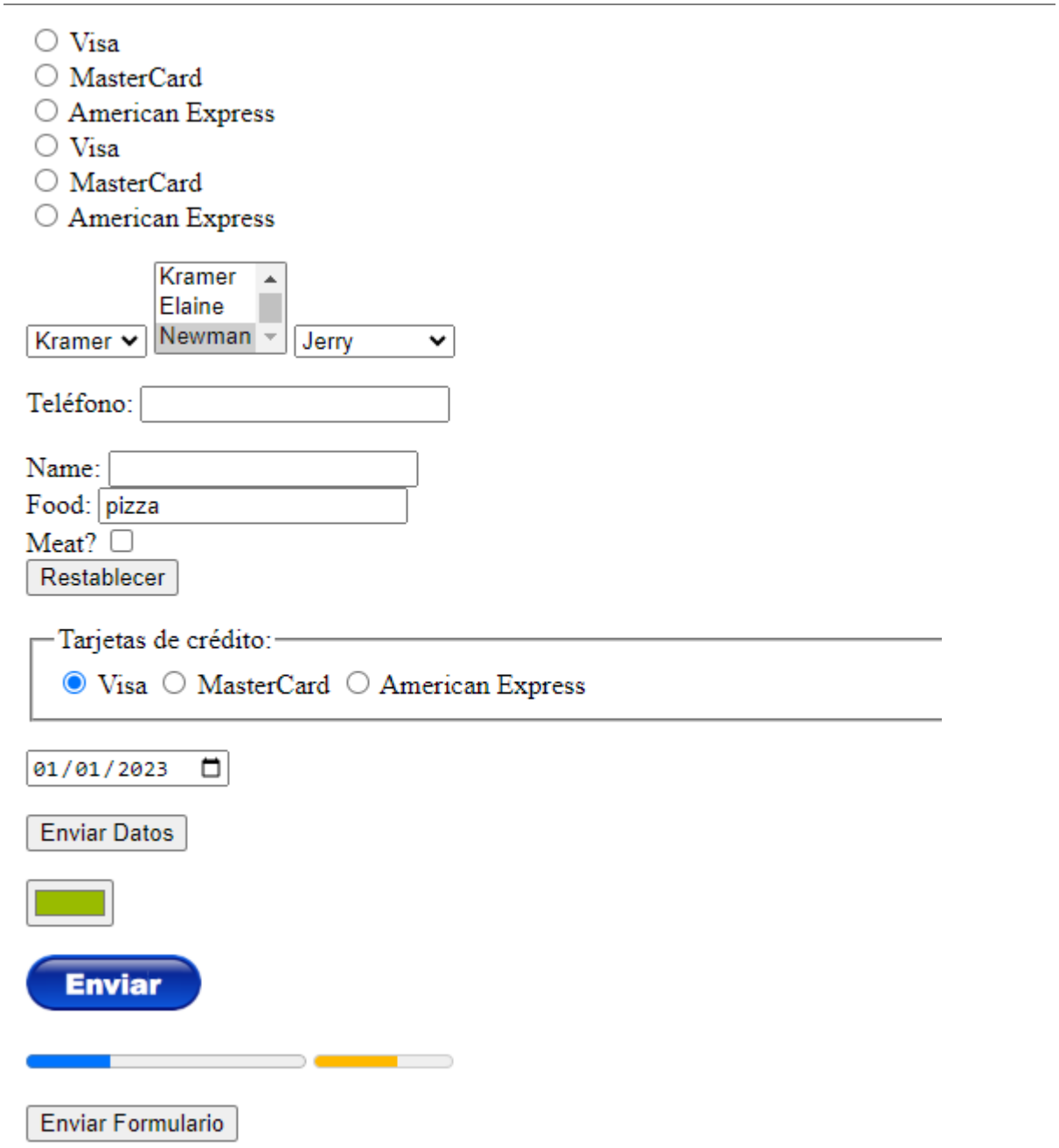
Nonmbre

NetID

Password

Texto:

Lechuga Tomate Pepino



2.4 VALIDACIÓN HTML CSS W3C

Los lenguajes de programación disponen de aplicaciones como compiladores o intérpretes que pueden verificar si la sintaxis de los programas desarrollados es válida. En el caso de HTML los navegadores sirven como intérprete del lenguaje, pero no proporcionan tanta información de si una página o su estilo es válido. En general los navegadores intentan hacer lo mejor para aceptar y corregir código HTML inválido, es decir, que pueden encubrir errores en la página.

Existe un servicio de validación que proporciona la organización W3C para HTML. El servicio se encuentra en la URL https://validator.w3.org/ (Figura 2-4) que es

una web que verifica si el código HTML proporcionado cumple con las especificaciones oficiales del W3C.

El validador permite especificar el código a verificar en tres maneras:

- Proporcionando una URL del fichero HTML en la Web.
- Subir un fichero HTML desde el ordenador.
- Copiando y pegando el código HTML a ser validado.

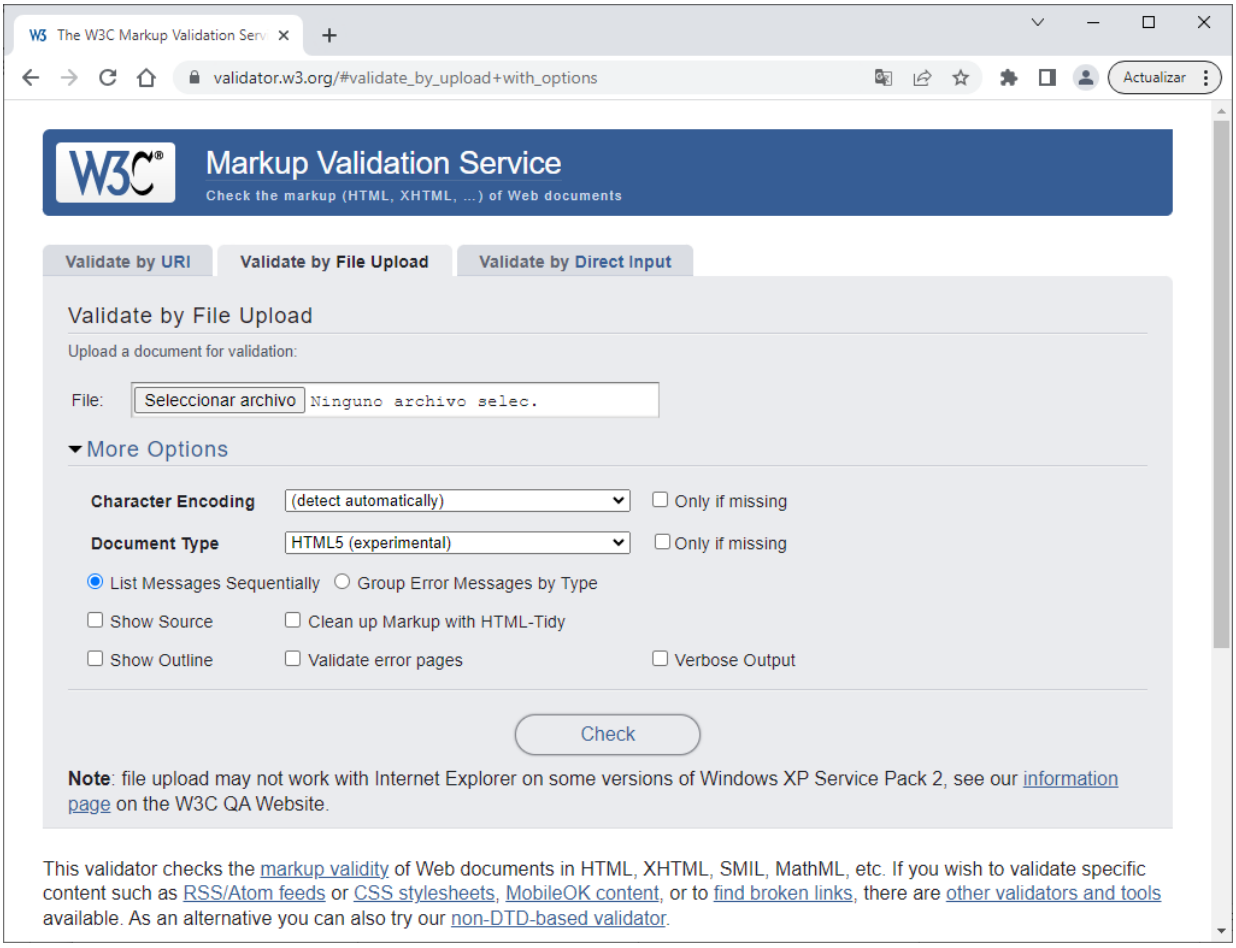


Figura 2-4 Servicio de validación del W3C

2.5 EJERCICIOS

1. La etiqueta para identificar las listas no ordenadas es:

   a)

   b)

   c) <---- Respuesta correcta

   d) <dl>

2. Para escribir el carácter ">" en el texto de una página se usa:

   a) >

   b) ">"

   c) > <---- Respuesta correcta

   d) <

3. Indicar cuál etiqueta **no** es de bloque (block):

   a) <hr>

   b) <---- Respuesta correcta

   c) <p>

   d) <h2>

4. Indicar cuál etiqueta **no** es en línea (inline):

   a) <a>

   b)

   c) <pre> <---- Respuesta correcta

   d)

5. Indicar la forma correcta de escribir un enlace en HTML:

   a) …

   b) …

   c) … <---- Respuesta correcta

   d) <link href=" ">…

6. Indicar en los siguientes formatos cuál no es un formato de imagen:

   a) .jpg

   b) .wav <---- Respuesta correcta

   c) .svg

   d) .png

3 CSS (Cascading Style Sheets – hojas de estilo en cascada)

Se ha visto que HTML se utiliza para definir la estructura y la semántica del contenido. Las Hojas de estilo en cascada (Cascading Style Sheets o CSS) se usan para darle estilo y diseño al contenido de las páginas Web. CSS se puede usar, por ejemplo, para cambiar la fuente, el color, el tamaño y el espaciado del contenido, para formar múltiples columnas y otros elementos decorativos. La especificación CSS le indica al navegador como se deben visualizar los elementos de un documento. Así se consigue separar el aspecto del contenido.

Los navegadores asignan estilos por defecto a los elementos HTML que pueden ser modificados en la configuración del navegador o reemplazados y complementados con estilos personalizados de CSS. Un estilo personalizado se declara con el nombre de la propiedad y su valor separado por dos puntos y terminado con un punto y coma. Por ejemplo, `font-size: 24px;` declara una propiedad que cambia el tamaño de la letra a 24 píxeles.

Se puede definir una hoja de estilo como una colección de *reglas de estilo* que se pueden aplicar a un conjunto seleccionado de elementos HTML. Se utiliza una regla de estilo para controlar la apariencia de los elementos HTML, tales como sus propiedades de fuente (tipo de letra, tamaño y alineamiento), propiedades de color (colores de fondo y de primer plano), alineación, margen, borde, relleno y posicionamiento.

Una característica de CSS es que los estilos se asignan en cascada. Los elementos en los niveles bajos de la jerarquía heredan los estilos asignados a los elementos en los niveles más altos. Los estilos heredados de elementos en niveles superiores se pueden reemplazar por nuevos estilos definidos para los elementos en niveles inferiores de la jerarquía.

El W3C define tres niveles de CSS, siendo el último el nivel 3. CSS3 está dividido en módulos (módulo de selectores, módulos de valores y unidades, módulo de alineación de cajas, etc.) alguno de los cuales se encuentran en desarrollo.

3.1 SINTAXIS DEL CSS

CSS es un lenguaje basado en reglas. Una regla es una sentencia CSS que describe un conjunto de etiquetas y el conjunto de estilos que se aplican a esas etiquetas. Cada regla consiste de uno o más *selectores* que seleccionan el elemento que se va a diseñar seguido por un conjunto de llaves { }. Entre estas habrá una o más declaraciones, que tomarán la forma de pares de *propiedad* y *valor*. Cada par especifica cada una de las propiedades de los elementos seleccionados y el valor que se quiere dar a esa propiedad. Por tanto, la sintaxis general de una regla CSS para un selector es:

```
selector {
  propiedad: valor;
  propiedad: valor;
  ...
  propiedad: valor;
}
```

Existen muchos tipos diferentes de selectores. La Tabla 9 resume los selectores de CSS de uso frecuente.

Tabla 9 Selectores CSS de uso frecuente

Selector	Ejemplos	Descripción
*	* { }	Selector universal: selecciona todos los elementos del documento
tag-name	h1 { }	Selector de etiquetas: selecciona todos los elementos <h1>
#id-name	#header { }	Selector de ID: selecciona el elemento único con id="header"
.class-name	.new { }	Selector de clase: selecciona elementos con class="new"
:pseudo-class	:first-child { } :focus { } a:active { }	Selector de seudo-clase: selecciona un estado especial como focus y hover.
::pseudo-element	::first-line { } ::first-letter { } ::before { } ::after { } ::selection { }	Selector de seudo-elemento: selecciona una parte de un elemento y puede modificar el elemento seleccionado.

La Tabla 10 resume los selectores combinados.

Tabla 10 Selectores combinados

Selector	Ejemplos	Descripción
S1,S2,S3	`h1, h2 { }`	Selector de grupo: aplica el estilo a S1, S2 y S3
S1 S2	`div p { }`	Selector de descendientes (separados por un espacio): selecciona S2 si es descendiente (hijo, nieto, etc.) de S1
S1>S2	`tr > td { }`	Selector de hijo (descendiente directo): solo si S1 es un hijo inmediato de S1
S1:S2	`ul:li { }`	Selector de primer hijo: solo si S2 es el primer hijo de S1
S1+S2	`a + span { }`	Selector de hermanos adyacentes: el (uno) S2 que está inmediatamente después de S1
S1~S2	`a ~ span { }`	Selector general de hermanos: todos los S2 que son hermanos después de S1
T.C	`div.example { }`	Selector de etiqueta cum clase: `<div class="example">`
T#C	`div#header { }`	Selector de etiqueta cum ID: `<div id="header">`
.C1.C2	`.new.example { }`	Selector de clases múltiples (sin espacios intermedios): `<div class="new example">`

La lista de selectores por atributos es:

- T[att]: selecciona elementos que poseen el atributo dado (att), independientemente del valor.
- T[att="valor"]: selecciona elementos que poseen el atributo dado, con el valor dado.
- T[att^="valor"]: selecciona elementos que poseen el atributo dado, comenzando con el valor dado. (donde ^ representa el comienzo en expresiones regulares).
- T[att$="valor"]: selecciona elementos que poseen el atributo dado, terminando con el valor dado. (donde $ representa el final en expresiones regulares).
- T[att*="valor"]: selecciona elementos que poseen el atributo dado, que contiene el valor dado.

En lo que respecta a las propiedades CSS en la Tabla 11 se encuentra la lista de las disponibles hasta el momento. Son cerca de cuatrocientos.

Tabla 11 Tabla de propiedades CSS

accent-color	align-content	align-items
align-self	all	animation
animation-composition	animation-delay	animation-direction
animation-duration	animation-fill-mode	animation-iteration-count
animation-name	animation-play-state	animation-timing-function
appearance	aspect-ratio	backdrop-filter
backface-visibility	background	background-attachment
background-blend-mode	background-clip	background-color
background-image	background-origin	background-position
background-position-x	background-position-y	background-repeat
background-size	block-size	border
border-block	border-block-color	border-block-end
border-block-end-color	border-block-end-style	border-block-end-width
border-block-start	border-block-start-color	border-block-start-style
border-block-start-width	border-block-style	border-block-width
border-bottom	border-bottom-color	border-bottom-left-radius
border-bottom-right-radius	border-bottom-style	border-bottom-width
border-collapse	border-color	border-end-end-radius
border-end-start-radius	border-image	border-image-outset
border-image-repeat	border-image-slice	border-image-source
border-image-width	border-inline	border-inline-color
border-inline-end	border-inline-end-color	border-inline-end-style
border-inline-end-width	border-inline-start	border-inline-start-color
border-inline-start-style	border-inline-start-width	border-inline-style
border-inline-width	border-left	border-left-color
border-left-style	border-left-width	border-radius
border-right	border-right-color	border-right-style
border-right-width	border-spacing	border-start-end-radius
border-start-start-radius	border-style	border-top
border-top-color	border-top-left-radius	border-top-right-radius
border-top-style	border-top-width	border-width
bottom	box-decoration-break	box-shadow
box-sizing	break-after	break-before
break-inside	caption-side	caret-color
clear	clip-path	color
color-scheme	column-count	column-fill
column-gap	column-rule	column-rule-color
column-rule-style	column-rule-width	column-span
column-width	columns	contain
contain-intrinsic-block-size	contain-intrinsic-height	contain-intrinsic-inline-size
contain-intrinsic-size	contain-intrinsic-width	container
container-name	container-type	content
counter-increment	counter-reset	counter-set
cursor	direction	display
empty-cells	filter	flex
flex-basis	flex-direction	flex-flow
flex-grow	flex-shrink	flex-wrap
float	font	font-family
font-feature-settings	font-kerning	font-language-override
font-optical-sizing	font-palette	font-size
font-size-adjust	font-stretch	font-style
font-synthesis	font-synthesis-small-caps	font-synthesis-style
font-synthesis-weight	font-variant	font-variant-alternates
font-variant-caps	font-variant-east-asian	font-variant-emoji
font-variant-ligatures	font-variant-numeric	font-variant-position
font-variation-settings	font-weight	forced-color-adjust

gap	grid	grid-area
grid-auto-columns	grid-auto-flow	grid-auto-rows
grid-column	grid-column-end	grid-column-start
grid-row	grid-row-end	grid-row-start
grid-template	grid-template-areas	grid-template-columns
grid-template-rows	hanging-punctuation	height
hyphenate-character	hyphenate-limit-chars	hyphens
image-orientation	image-rendering	inline-size
inset	inset-block	inset-block-end
inset-block-start	inset-inline	inset-inline-end
inset-inline-start	isolation	justify-content
justify-items	justify-self	left
letter-spacing	line-break	line-height
list-style	list-style-image	list-style-position
list-style-type	margin	margin-block
margin-block-end	margin-block-start	margin-bottom
margin-inline	margin-inline-end	margin-inline-start
margin-left	margin-right	margin-top
mask	mask-border	mask-border-mode
mask-border-outset	mask-border-repeat	mask-border-slice
mask-border-source	mask-border-width	mask-clip
mask-composite	mask-image	mask-mode
mask-origin	mask-position	mask-repeat
mask-size	mask-type	math-style
max-block-size	max-height	max-inline-size
max-width	min-block-size	min-height
min-inline-size	min-width	mix-blend-mode
object-fit	object-position	offset
offset-anchor	offset-distance	offset-path
offset-rotate	opacity	order
orphans	outline	outline-color
outline-offset	outline-style	outline-width
overflow	overflow-anchor	overflow-block
overflow-clip-margin	overflow-inline	overflow-wrap
overflow-x	overflow-y	overscroll-behavior
overscroll-behavior-block	overscroll-behavior-inline	overscroll-behavior-x
overscroll-behavior-y	padding	padding-block
padding-block-end	padding-block-start	padding-bottom
padding-inline	padding-inline-end	padding-inline-start
padding-left	padding-right	padding-top
page	page-break-after	page-break-before
page-break-inside	paint-order	perspective
perspective-origin	place-content	place-items
place-self	pointer-events	position
print-color-adjust	quotes	resize
right	rotate	row-gap
ruby-position	scale	scroll-behavior
scroll-margin	scroll-margin-block	scroll-margin-block-end
scroll-margin-block-start	scroll-margin-bottom	scroll-margin-inline
scroll-margin-inline-end	scroll-margin-inline-start	scroll-margin-left
scroll-margin-right	scroll-margin-top	scroll-padding
scroll-padding-block	scroll-padding-block-end	scroll-padding-block-start
scroll-padding-bottom	scroll-padding-inline	scroll-padding-inline-end
scroll-padding-inline-start	scroll-padding-left	scroll-padding-right
scroll-padding-top	scroll-snap-align	scroll-snap-stop
scroll-snap-type	scrollbar-color	scrollbar-gutter
scrollbar-width	shape-image-threshold	shape-margin
shape-outside	tab-size	table-layout
text-align	text-align-last	text-combine-upright
text-decoration	text-decoration-color	text-decoration-line
text-decoration-skip-ink	text-decoration-style	text-decoration-thickness
text-emphasis	text-emphasis-color	text-emphasis-position
text-emphasis-style	text-indent	text-justify

text-orientation	text-overflow	text-rendering
text-shadow	text-transform	text-underline-offset
text-underline-position	top	touch-action
transform	transform-box	transform-origin
transform-style	transition	transition-delay
transition-duration	transition-property	transition-timing-function
translate	unicode-bidi	user-select
vertical-align	visibility	white-space
widows	width	will-change
word-break	word-spacing	writing-mode
z-index		

A su vez, existen una variedad de valores de las propiedades que se describen en las siguientes secciones.

3.1.1 Selectores

Un selector * selecciona todos los elementos. Por ejemplo, la regla para que el elemento <p> se muestre en color rojo y con un tamaño de 24 píxeles es:

```
p {
  color: #ff0000;
  font-size: 24px;
}
```

Los nombres de las propiedades deben escribirse siempre en minúsculas. Las propiedades multi palabra se separan con guiones, por ejemplo, `font-family` o `list-style-type`. También es posible tener reglas que aplican a varios selectores que se separan por comas, por ejemplo:

```
h1, h2 {
  color: green;
}
```

Para referenciar elementos que se encuentran dentro de un elemento en particular se listan los selectores separados por un espacio. Estos tipos de selectores se llaman *selectores de descendencia* porque afectan a elementos dentro de otros elementos, sin importar el lugar que ocupan en la jerarquía. Por ejemplo, la siguiente regla afecta solo a los elementos `<p>` que se encuentran dentro de un elemento `<main>`, ya sea como contenido directo o insertados en otros elementos.

```
main p {
  font-size: 20px;
}
```

Para referenciar un elemento que es hijo directo de otro elemento se usa el carácter >. El carácter > indica que el elemento afectado es el elemento de la

derecha cuando tiene como padre al elemento de la izquierda. Por ejemplo, la regla para modificar los elementos <p> que son hijos del elemento <section> es

```
section > p {
  font-size: 20px;
}
```

Con el carácter + se crea otro selector que referencia un elemento que está precedido por otro elemento. Ambos deben compartir el mismo elemento padre. Por ejemplo, la siguiente regla afecta a todos los elementos `<p>` que se ubican inmediatamente después de un elemento `<h1>`.

```
h1 + p {
  font-size: 20px;
}
```

Con el carácter ~ se crea un selector que afecta a todos los elementos que se ubican a continuación de otro elemento. Esta regla afecta a todos los elementos encontrados, no solo al primero. Por ejemplo, la regla para afectar a todos los elementos `<p>` que preceden a otro elemento `<p>` es:

```
p ~ p {
  font-size: 20px;
}
```

Atributo id

Para seleccionar un elemento HTML sin considerar su tipo, podemos usar el atributo id. Este atributo es un identificador exclusivo del elemento y se puede usar para encontrar un elemento en particular dentro del documento. Para referenciar un elemento usando su atributo id, el selector debe incluir el valor del atributo precedido por el carácter numeral (#).

En el siguiente código HTML se especifica un elemento `<p>` con el atributo `id` con el valor `mitexto`.

```
<p>Frase 1</p>
<p id="mitexto">Frase 2</p>
<p>Frase 3</p>
```

La regla que aplica al elemento identificado con el atributo `id="mitexto"` es:

```
#mitexto {
  font-size: 20px;
}
```

Se recuerda que el atributo `id` es única en todo el documento. Por tanto, es una forma muy específica de referenciar a un elemento por lo que se usa comúnmente con elementos estructurales, como `<section>` o `<div>`. Debido a su especificidad, el atributo id también se usa frecuentemente para referenciar elementos desde JavaScript

Atributo class

El atributo class se puede asignar a varios elementos dentro del mismo documento. Para referenciar un elemento usando su atributo class, el selector debe incluir el valor del atributo precedido por un punto.

En el siguiente código HTML se especifica varios elementos `<p>` con el atributo `class` con el valor `mitexto`.

```
<p class="mitexto">Frase 1</p>
<p class="mitexto">Frase 2</p>
<p>Frase 3</p>
<p>Frase 4</p>
```

La regla que aplica a los elementos identificados con el atributo `class="mitexto"` es:

```
.mitexto {
  font-size: 20px;
}
```

A un mismo elemento se le pueden asignar varias clases. Para ello se declaran los nombres de las clases separados por un espacio (por ejemplo, `class="texto1 texto2"`). Las clases también se pueden declarar como exclusivas para un tipo específico de elementos declarando el nombre del elemento antes del punto. Por ejemplo, una regla que referencia la clase llamada `mitexto`, pero solo para los elementos `<p>` es:

```
p.mitexto {
  font-size: 20px;
}
```

Otros elementos que contengan el mismo valor en su atributo class no se verán afectados por esta regla.

Otros atributos

Existen situaciones en las que los atributos `id` y `class` no son suficientes para encontrar el elemento exacto que se desea modificar. CSS permite referenciar un elemento por medio de cualquier otro atributo que se necesite. La sintaxis para definir esta clase de selectores incluye el nombre del elemento seguido del nombre del atributo en corchetes. Por ejemplo, La regla que modifica solo los elementos `<p>` que tienen un atributo llamado `name` es:

```
p[name] {
  font-size: 20px;
}
```

Para emular lo explicado con los atributos id y class, también se puede incluir el valor del atributo. La regla para referenciar elementos `<p>` que tienen un atributo name con el valor `mitexto` es:

```
p[name="mitexto"] {
  font-size: 20px;
}
```

CSS permite combinar el carácter = con otros caracteres para realizar una selección más específica. Los caracteres más utilizados son: ~, ^, $ y *. Por ejemplo, las reglas con selectores que incluyen los mismos atributos y valores, pero referencian diferentes elementos según se aplican los caracteres con = es:

```
p[name~="mi"] {
  font-size: 20px;
}
p[name^="mi"] {
  font-size: 20px;
}
p[name$="mi"] {
  font-size: 20px;
}
p[name*="mi"] {
  font-size: 20px;
}
```

Con estas reglas el selector con los caracteres:

~= referencia cualquier elemento `<p>` con un atributo `name` cuyo valor incluye la palabra `"mi"` (por ejemplo, `"mi texto"`, `"mi coche"`).

^= referencia cualquier elemento `<p>` con el atributo `name` cuyo valor comienza en "mi" (por ejemplo, "mitexto", "micoche").

$= referencia cualquier elemento <p> con un atributo name cuyo valor termina en "mi" (por ejemplo, "textomi", "cochemi").

*= referencia cualquier elemento <p> con un atributo name cuyo valor contiene la cadena de caracteres "mi" (en este caso, la cadena de caracteres podría también encontrarse en el medio del texto, como en "textomicoche").

Seudoclases

Las seudoclases permiten referenciar elementos HTML por medio de sus características, como sus posiciones en el código o sus condiciones actuales. Las más utilizadas son:

:nth-child(valor) - selecciona un elemento de una lista de elementos hermanos que se encuentra en la posición especificada por el valor entre paréntesis.

:first-child - selecciona el primer elemento de una lista de elementos hermanos.

:last-child - selecciona el último elemento de una lista de elementos hermanos.

:only-child - selecciona un elemento cuando es el único hijo de otro elemento.

:first-of-type - selecciona el primer elemento de una lista de elementos del mismo tipo.

:not(selector) - selecciona los elementos que no coinciden con el selector entre paréntesis.

Con estos selectores, es posible realizar una selección más dinámica. Por ejemplo, en el siguiente código HTML:

```
<body>
  <main>
    <section>
      <p class="mitexto1">Frase 1</p>
      <p class="mitexto2">Frase 2</p>
      <p class="mitexto3">Frase 3</p>
      <p class="mitexto4">Frase 4</p>
    </section>
  </main>
</body>
```

Los cuatro elementos <p> son hermanos y, por lo tanto, hijos del mismo elemento <section>. Usando seudoclases, se puede aprovechar esta

organización y referenciar elementos sin importar cuánto conocemos acerca de sus atributos o valores. Por ejemplo, con la seudoclase `:nth-child()` se puede modificar todos los segundos elementos `<p>` que se encuentran en el documento con la regla:

```
p:nth-child(2) {
  font-size: 20px;
}
```

La seudoclase `:nth-child()` indica algo como "el hijo en la posición…", por lo que el número entre paréntesis corresponde al número de posición del elemento hijo, o índice. Es posible asignar estilos a cada elemento creando una regla similar para cada uno de ellos. Por ejemplo, las siguientes reglas cambia el color de fondo de los párrafos según el orden de hijo:

```
p:nth-child(1) {
  background-color: #999999;
}
p:nth-child(2) {
  background-color: #cccccc;
}
p:nth-child(3) {
  background-color: #999999;
}
p:nth-child(4) {
  background-color: #cccccc;
}
```

En las reglas anteriores hay una alternativa de usar las palabras clave `odd` y `even` para esta seudoclase. Con `odd` se considera los hijos de otro elemento que tienen un índice impar y palabra clave `even` afecta a aquellos que tienen un índice par. Usando estas palabras clave, las reglas se reducen a:

```
p:nth-child(odd) {
  background-color: #999999;
}
p:nth-child(even) {
  background-color: #cccccc;
}
```

Existen otras seudoclases que están relacionadas y también ayudan a encontrar elementos específicos dentro de la jerarquía, como

`:first-child` para hacer referencia solo el primer elemento hijo.

:last-child referencia solo el último elemento hijo.

:only-child afecta a un elemento cuando es el único elemento hijo.

:not() para seleccionar elementos que no coinciden con un selector.

Otras seudoclases se refieren al estado:

Para normalmente con el elemento <a>

:link para seleccionar un elemento que aún no se ha visitado

:visited para seleccionar enlaces que el usuario ya ha visitado

:focus para seleccionar un elemento (como una entrada de formulario) que ha recibido el foco. Generalmente se activa cuando el usuario hace clic, toca un elemento o lo selecciona con la tecla "Tab" del teclado.

Para otros elementos:

:hover coincide cuando el usuario interactúa con un elemento con un dispositivo señalador, pero no necesariamente lo activa. Generalmente se activa cuando el usuario se desplaza sobre un elemento con el cursor (puntero del mouse).

:active para seleccionar un elemento (como un botón) que el usuario está activando.

:enabled para seleccionar cualquier elemento habilitado

:disabled para seleccionar a cualquier elemento deshabilitado

:checked para seleccionar cualquier radio (<input type="radio">), checkbox (<input type="checkbox">) u option (<option> en un elemento <select>) que está marcado o conmutado a un estado on.

En validación de formularios

:invalid para seleccionar cualquier elemento <input> u otro elemento <form> cuyos contenidos no se puedan validar.

:required para seleccionar cualquier elemento <input>, <select>, o <textarea> que tenga el atributo required establecido en él.

:out-of-range para seleccionar un elemento <input> cuyo valor actual está fuera de los límites de rango especificados por los atributos min y max.

3.1.2 Unidades

El código CSS tiene varias unidades diferentes para expresar una longitud que son utilizadas para asignar los valores de algunas propiedades. La longitud se expresa como un número seguido de una unidad de longitud, por ejemplo, 10px o 2em. No puede aparecer un espacio en blanco entre el número y la unidad. Algunas propiedades pueden tener longitudes negativas.

Hay dos tipos de medidas de longitud: absoluta y relativa. La longitud relativa especifica una longitud relativa a otra propiedad de longitud. Con ellas se puede escalar mejor entre diferentes tipos de pantallas.

Las unidades absolutas son:

`cm` : centimetros

`mm` : milimetros

`in` : inches (1in = 96px = 2.54cm)

`px` : pixels (1px = 1/96th of 1in)

`pt` : points (1 pulgada tiene 72 puntos. 1pt es 1/72 pulg ≈ 0,014 pulg ≈ 0,35 mm.)

Las unidades relativas son:

`rem` : relativo al elemento `font-size` raíz o elemento `<html>`.

`em` : el ancho de la letra 'm' de una fuente referenciada, generalmente, la fuente actual. Por ejemplo, `margin:2em` significa que los márgenes son el doble del tamaño de la fuente actual (referenciado).

`vw` : uno por ciento del ancho de la ventana gráfica.

`vh` : uno por ciento de la altura de la ventana gráfica.

`%` : en términos del porcentaje de una propiedad de un elemento referenciado, generalmente, la misma propiedad del elemento padre. Por ejemplo, `table {width:80%}` establece el ancho de la tabla en el 80% del ancho del padre (puede ser `<div>` o `<body>`).

Comentarios

Se puede insertar comentarios en un archivo `.css` en usando el caracter compuestos `/*` al inicio del comentario y `*/` para terminar. Ejemplo:

```
/* Comentario de una linea */
/* Comentario
   en varias
   lineas
*/
p {
  color: red;
  font-size: 20px;
}
```

Modelo de caja CSS

Se comentó en HTML que se definen dos tipos de elementos: elemento de bloque y elemento en línea. Un elemento de bloque (como `<p>`, `<div>`, `<h1>` a `<h6>`) siempre tiene forma rectangular y presenta el llamado modelo de caja, con cuatro rectángulos virtuales que envuelven su "área de contenido", que representan el área de contenido, el relleno (padding), el borde (border) y el margen (margin), como se muestra en la Figura 3-1 y se describe a continuación:

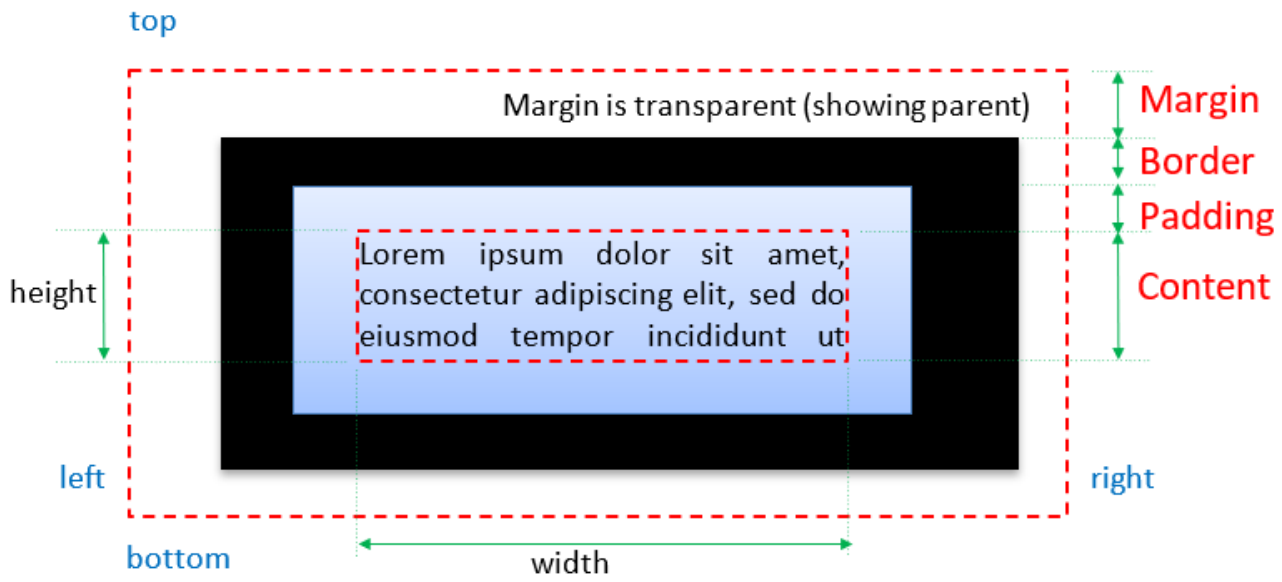


Figura 3-1 Modelo de caja CSS

content area (área de contenido) contiene los textos, la imagen o los elementos descendientes.

padding (relleno) es el espacio entre el área de contenido y el borde. Establece un área fuera del área de contenido. Tiene el mismo fondo que el área de contenido.

border (borde) va entre el relleno y el margen. Puede establecer un color y un estilo (sólido, guión, punteado) en el borde.

margin (margen) es el espacio fuera del borde (a otro elemento). El margen no tiene fondo, y es totalmente transparente.

Este modelo de caja se puede inspeccionar para los elementos de una página utilizando la herramienta para desarrolladores que tienen la mayoría de navegadores, obtenido de manera rápida con la tecla de función F12. Por ejemplo, la Figura 3-2 muestra el modelo de caja en Chrome.

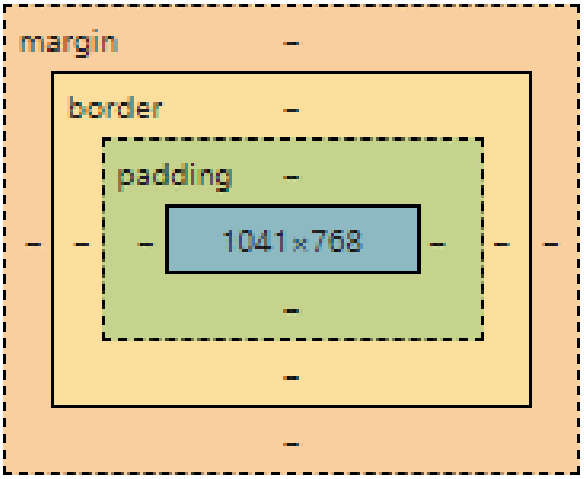


Figura 3-2 Modelo de caja obtenido en Chrome

Este modelo de caja se puede inspeccionar para los elementos de una página utilizando la herramienta para desarrolladores que tienen la mayoría de navegadores, obtenido de manera rápida con la tecla de función F12. Por ejemplo, la Figura 17 muestra el modelo de caja en Chrome.

3.2 ESTILOS EN HTML

Hay varias maneras de aplicar estilos en una página web: estilo en línea, estilos internos o incrustados y hoja de estilo externa. La recomendación general es aplicar el método de hojas de estilo externa por razones de facilidad de mantenimiento y porque se puede aplicar el mismo estilo a un conjunto de páginas.

3.2.1 Estilo en línea

El estilo se coloca en el atributo `style` de los elementos HTML. Los estilos se aplican solo a los elementos que declaran el atributo `style`.

```
<body>
  <p style="color: red; font-size: 26px">Este es el primer párrafo</p>
  <p>Este es el segundo párrafo</p>
</body>
```

El atributo `style` solo afecta al elemento en el que se ha declarado, en este caso el primer párrafo.

3.2.2 Estilo incrustado o interno

Las reglas CSS se insertan en la cabecera (`<head>`) del documento usando el elemento `<style>` y usando los selectores que determinan los elementos que se verán afectados. Ejemplo:

```
<!DOCTYPE html>
<html lang="es">
  <head>
    <title>Este texto es el título del documento</title>
    <style>
      p {
        color: red;
        font-size: 20px;
      }
    </style>
  </head>
  <body>
    <p>Este es el primer párrafo</p>
    <p>Este es el segundo párrafo</p>
  </body>
</html>
```

3.2.3 Estilo externo

Los estilos se colocan en un archivo CSS externo con extensión .css. Para especificar en el archivo HTML el archivo CSS externo se usa el elemento `<link>` dentro del elemento `<head>`. El elemento `<link>` tiene dos

atributos:

`rel`: indica relación entre el documento y el archivo que se vincula, por lo que se debe declarar con el valor `stylesheet`.

`href`: atributo que especifica la URL del recurso vinculado.

Ejemplo:

```
<!DOCTYPE html>
<html lang="es">
  <head>
    <title>Este texto es el título del documento</title>
    <link rel="stylesheet" href="misestilos.css">
  </head>
  <body>
    <p>Mi texto</p>
  </body>
</html>
```

misestilos.css

```
p {
  color: red;
  font-size: 20px;
}
```

En el elemento `link` se puede especificar el atributo `media` para especificar el tipo de medio al que aplica el recurso enlazado. Este atributo es usado para permitir al agente usuario seleccionar la que mejor se adapte al dispositivo sobre el que se ejecuta. Los valores que puede asignarse a este atributo son: `all`, `print`, `screen` siendo `all` el valor por defecto. Ejemplo:

```
<link rel="stylesheet" href="misestilos.css" media="print">
```

También es posible enlazar múltiples hojas de estilo:

```
<head>
  <link rel="stylesheet" href="general.css">
  <link rel="stylesheet" href="especifico.css">
  <link rel="stylesheet" href="misestilos.css">
</head>
```

En ese caso, las reglas de los ficheros finales prevalecen sobre los primeros.

Además del elemento `<link>`, también se puede usar la directiva de CSS `@import` para vincular a una hoja de estilo externa bajo las etiquetas `<style>`, de la siguiente manera:

```
<!-- en fichero HTML -->
<style>
  @import url("cssURL1.css");
  @import url("cssURL2.css");
</style>
```

`@import` es una directiva CSS (parte del lenguaje CSS). También se puede usar en un archivo CSS para incluir reglas de otro archivo CSS, por ejemplo:

```
/* en archivo CSS */
@import '/css/estilos_extras.css';

/* Importar una fuente externa de Google */
@import 'https://fonts.googleapis.com/css?family=Open+Sans';

/* El segundo argumento son tipos de medios del dispositivo y característica */
@import '/css/print-styles.css' print ;
@import '/css/landscape.css' screen and (orientation:landscape) ;
```

3.3 PROPIEDADES

Las propiedades se pueden se pueden clasificar en dos tipos: propiedades de formato y propiedades de diseño. Las propiedades de formato se encargan de dar forma a los elementos y su contenido, mientras que las de diseño están enfocadas a determinar el tamaño y la posición de los elementos en la pantalla. A su vez, las propiedades de formato se pueden clasificar según el tipo de modificación que producen. Por ejemplo, algunas propiedades cambian el tipo de letra que se usa para mostrar el texto, otras generan un borde alrededor del elemento, asignan un color de fondo, etc.

3.3.1 Fuentes

Las propiedades disponibles en CSS para definir el tipo de letra, tamaño, y estilo de un texto son:

`font-family` - propiedad para declarar el tipo de letra que se usa para mostrar el texto. Se pueden declarar varios valores separados por coma para ofrecer al navegador varias alternativas en caso de que algunos tipos de letra no se encuentren disponibles en el ordenador del usuario. Algunos valores estándar son `Georgia`, `"Times New Roman"`, `Arial`, `Helvetica`, `"Arial Black"`, `Gadget`, `Tahoma`, `Geneva`, `Helvetica`, `Verdana`, `Geneva`, `Impact`, y `sans-serif` (los nombres compuestos por más de una palabra se deben declarar entre comillas dobles).

`font-size` - propiedad que determina el tamaño de la letra. El valor puede ser declarado en píxeles (`px`), porcentaje (`%`), o usando cualquiera de las unidades

disponibles en CSS como em, rem, pt, etc. También hay valores nominales como xx-small, x-small, small, medium, large, x-large, xx-large, smaller, larger. El valor por defecto es normalmente 16px.

font-weight - propiedad para determinar si el texto se mostrará en negrita o no. Los valores disponibles son normal y bold, pero también se puede asignar valores entre 100 y 900 en intervalos de 100 (solo disponibles para algunos tipos de letra).

font-style - propiedad para asignar el estilo de la letra. Los valores disponibles son normal, italic, y oblique.

font - propiedad abreviada que permite declarar varios valores al mismo tiempo. Los valores deben declararse separados por un espacio y en un orden preciso. El estilo y el grosor se deben declarar primero seguido del tamaño, y el tipo de letra al final (por ejemplo, font: bold 24px Arial,sans-serif).

3.3.2 Texto

También hay propiedades relativas a otros aspectos del texto como el alineamiento, la sangría, el espacio entre líneas, etc. Algunas de las propiedades disponibles para este propósito son:

text-align - propiedad para alinear el texto dentro de un elemento. Los valores disponibles son left, right, center, y justify.

text-align-last - propiedad para alinear la última línea de un párrafo. Los valores disponibles son left, right, center, y justify.

text-indent - propiedad para definir el tamaño de la sangría de un párrafo (el espacio vacío al comienzo de la línea). El valor se puede declarar en píxeles (px), porcentaje (%), o usando cualquiera de las unidades disponibles en CSS, como em, rem, pt, etc.

text-decoration - propiedad para resaltar el texto con una línea. Los valores disponibles son none, underline, overline, line-through, blink.

text-transform - propiedad para especificar el cambio entre mayúsculas y minúsculas del texto de un elemento. Puede ser usada para que un texto aparezca completamente en mayúsculas (uppercase), en minúsculas

(lowercase), o con la primera letra de cada palabra en mayúscula (capitalize). Valor por defecto none.

letter-spacing - propiedad para definir el espacio entre letras. El valor se debe declarar en píxeles (px), porcentaje (%) o usando cualquiera de las unidades disponibles en CSS, como em, rem, pt, etc.

word-spacing - propiedad para definir el ancho del espacio entre palabras. El valor puede ser declarado en píxeles (px), porcentaje (%) o usando cualquiera de las unidades disponibles en CSS, como em, rem, pt, etc.

line-height - propiedad para definir el espacio entre líneas. El valor se puede declarar en píxeles (px), porcentaje (%) o usando cualquiera de las unidades disponibles en CSS, como em, rem, pt, etc.

vertical-align - propiedad para alinear elementos verticalmente. Se usa frecuentemente para alinear texto con imágenes (la propiedad se aplica a la imagen). Los valores disponibles son baseline, sub, super, text-top, text-bottom, middle, top, y bottom.

white-space - propiedad para determinar cómo se maneja el espacio en blanco dentro de un elemento. Los valores disponibles son normal, pre, nowrap

3.3.3 Incrustado de Fuentes

Los tipos de letra o fuentes (fonts) que se usan para mostrar las páginas web, se cargan desde el ordenador del usuario. Cabe la posibilidad que el usuario no tenga la fuente especificada por lo que las páginas se verán diferentes.

Con el fin de asegurar que todos los usuarios visualizan el documento con la fuente que se desea existe la regla @font-face que permite incluir un archivo con la fuente de letra a usar para mostrar el texto de una página web.

La regla @font-face necesita al menos dos propiedades para declarar la fuente y cargar el archivo. La propiedad font-family especifica el nombre a usar para referenciar este tipo de letra y la propiedad src indica la URL, a través de la función url(), del archivo con las especificaciones de la fuente (formatos .ttf, .otf, .woff). Una vez que se carga la fuente, se puede usar en cualquier elemento

del documento por medio de su nombre. Por ejemplo, el siguiente código carga una fuente desde el archivo font.ttf y le asigna el nombre MiNuevaLetra

```
#titulo1 {
  font: bold 26px MiNuevaLetra, Verdana, sans-serif;
}

@font-face {
  font-family: "MiNuevaLetra";
  src: url("font.ttf");
}
```

Existe también la posibilidad de importar tipos de letras de código abierto desde portales que ofrecen el servicio de repositorio de fuentes. Una de ellas es Google Fonts que ofrece un conjunto enorme de fuentes de alta calidad. Hay varias posibilidades para usar esas fuentes. Por ejemplo, para el tipo de letra "Roboto" se usa la regla `@import` que proporciona la página y se coloca en la hoja de estilo (.css) donde se va a usar el tipo de letra. La regla completa es:

```
@import url('https://fonts.googleapis.com/css2?family=Roboto&display=swap');
```

Después se puede usar la propiedad:

```
font-family: 'Roboto', sans-serif;
```

Hay varios portales alternativos similares, por ejemplo Adobe Fonts o MyFonts.

3.3.4 Color

Existen dos formas para especificar el color en CSS: usar una combinación de tres colores básicos (rojo, verde y azul) llamado RGB, o definir el matiz, la saturación y la luminosidad llamado HSL. El color final se crea considerando los niveles que asignamos a cada componente. Dependiendo del tipo de sistema usado para definir el color, se declaran los niveles usando números hexadecimales (rango 00 - FF), números decimales (rango 0 - 255) o porcentajes.

Por ejemplo, si se usa el formato RGB, los valores del color se declaran en secuencia y precedidos por el carácter numeral, como en #335544 (33 para el nivel de rojo, 55 para el nivel de verde y 44 para el nivel de azul). También se puede especificar el color mediante nombres de color predefinidos, algunos de los cuales se muestran en la Figura 3-3. La lista estándar de los nombres de

colores se puede encontrar en https://www.w3.org/wiki/CSS/Properties/color/keywords

Figura 3-3 Colores predefinidos en CSS

Además, CSS proporciona las siguientes funciones:

`rgb(rojo, verde, azul)` - función para definir un color por medio de los valores especificados por los atributos (de 0 a 255 o en porcentaje entre 0% y 100%). El primer valor representa el nivel de rojo, el segundo valor representa el nivel de verde y el último valor el nivel de azul (por ejemplo, `rgb(123, 152, 48)`).

`rgba(rojo, verde, azul, alfa)` - función similar a la función `rgb()`, pero incluye un componente adicional para definir la opacidad (alfa). El valor se puede declarar entre 0 (totalmente transparente) y 1 (totalmente opaco).

`hsl(matiz, saturación, luminosidad)` - función para definir un color desde los valores especificados por los atributos. El matiz (hue) es el color en la rueda de colores en grados entre 0 y 360. La saturación (saturation) indica la pureza de color y se expresa en porcentaje entre 0% y 100% (color puro). La luminosidad (lightness) indica el brillo o intensidad, se expresa en porcentaje entre 0% (más oscuro) y 100% (más brillante).

`hsla(matiz, saturación, luminosidad, alfa)` - función similar a la función `hsl()`, pero incluye un componente adicional para definir la opacidad (alfa). El valor se puede declarar entre 0 (totalmente transparente) y 1 (totalmente opaco).

A continuación, se muestra ejemplos de especificación de colores en diferentes formatos:

```
/* ejemplos para el mismo color RGB: */
#f03
#F03
#ff0033
#FF0033
rgb(255,0,51)
rgb(100%,0%,20%)
/* ejemplos con rgba() */
rgba(255,0,0,0.1)    /* 10% opaque red */
rgba(255,0,0,0.4)    /* 40% opaco red */
rgba(255,0,0,0.7)    /* 70% opaco red */
rgba(255,0,0,  1)    /* full opaco red */
/* ejemplos hsl(): */
hsl(0,  100%,50%)    /* red */
hsl(120,100%,50%)    /* green */
hsl(240,100%,50%)    /* blue */
hsl(360,100%,50%)    /* red */
hsl(120,100%,25%)    /* dark green */
hsl(120,100%,50%)    /* green */
hsl(120,100%,75%)    /* light green */
hsl(120, 60%,70%)    /* pastel green */
/* ejemplos hsla() */
hsla(240,100%,50%,0.05)   /* 5% opaco blue */
hsla(240,100%,50%, 0.4)   /* 40% opaco blue */
hsla(240,100%,50%, 0.7)   /* 70% opaco blue */
hsla(240,100%,50%,   1)   /* full opaco blue */
```

Los navegadores ayudan a determinar los valores de los componentes de los colores mediante la inspección de la página (tecla de función F12) como se muestra en la Figura 3-4.

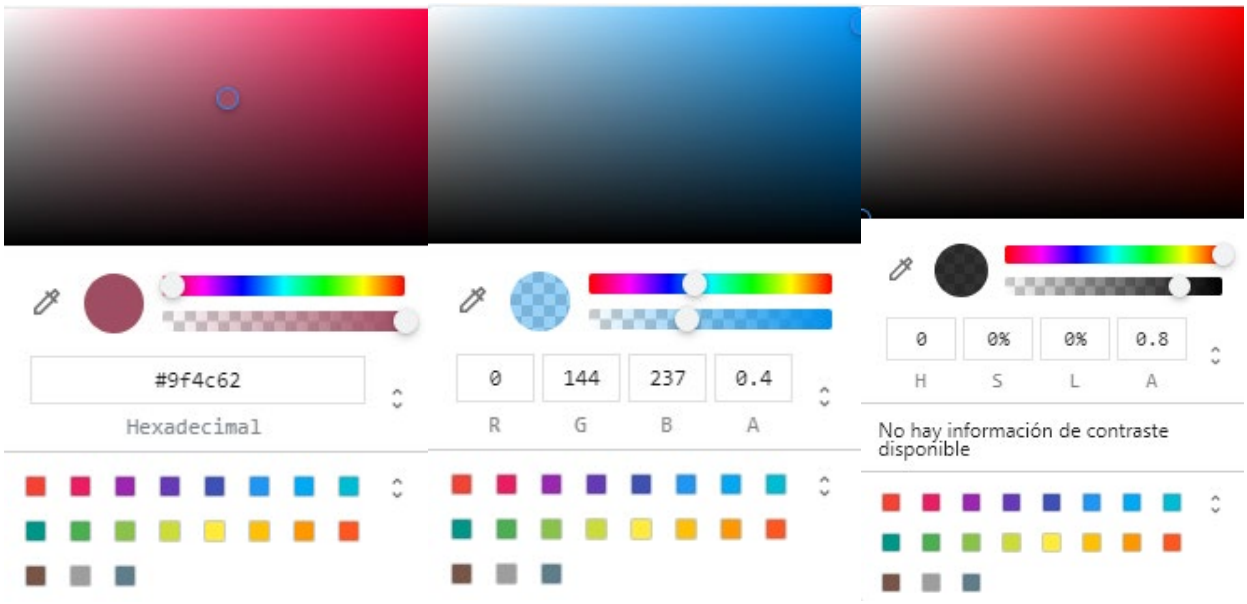


Figura 3-4 Paletas para asignar colores en Chrome

3.3.5 Tamaño

El tamaño de la mayoría de los elementos se determina según el espacio disponible en el contenedor. El ancho de un elemento se define como 100 %, lo que significa que será tan ancho como su contenedor, y tendrá una altura determinada por su contenido. Para declarar un tamaño personalizado hay las siguientes propiedades:

`width` - propiedad para declarar el ancho de un elemento. El valor se puede especificar en píxeles, porcentaje, o con la palabra clave `auto` (por defecto). Cuando el valor se especifica en porcentaje, el ancho se calcula a partir del ancho del contenedor, y con el valor auto, el elemento se expande hasta ocupar todo el espacio horizontal disponible dentro del contenedor.

`height` - propiedad para declarar la altura de un elemento. El valor se puede especificar en píxeles, porcentaje, o con la palabra clave `auto` (por defecto). Cuando el valor se especifica en porcentaje, el navegador calcula la altura a partir de la altura del contenedor, y cuando se declara con el valor auto, el elemento adopta la altura de su contenedor.

Según el modelo de cajas se genera una caja alrededor de cada elemento que determina el área que ocupa en la pantalla. Cuando se declara un tamaño personalizado, la caja se modifica y el contenido del elemento se adapta para

encajar dentro de la nueva área. La Figura 3-5 muestra el tamaño por defecto de un elemento y el tamaño personalizado según la regla:

```
#descripcion1 {
  width: 200px;
  height: 100px;
  border: 1px solid;
}
```

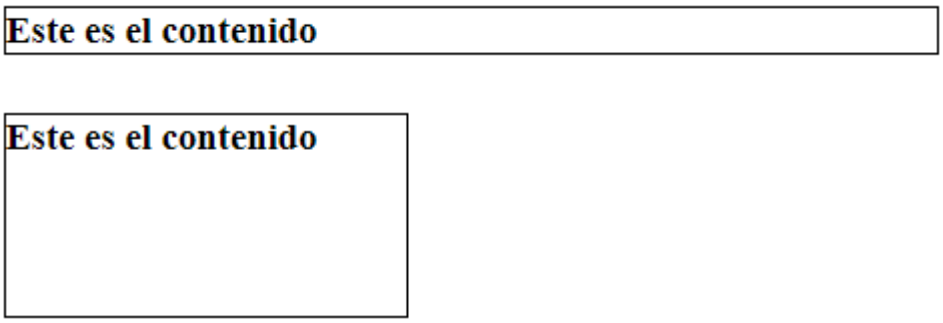


Figura 3-5 Caja de elemento

Puede ocurrir que cuando se define un tamaño para un elemento, el contenido no se pueda mostrar en su totalidad. Por defecto, los navegadores muestran el resto del contenido fuera del área de la caja. En ese caso, parte del contenido de una caja con tamaño personalizado se puede posicionar sobre el contenido del elemento que se encuentra debajo, tal como se ilustra en la Figura 3-6.

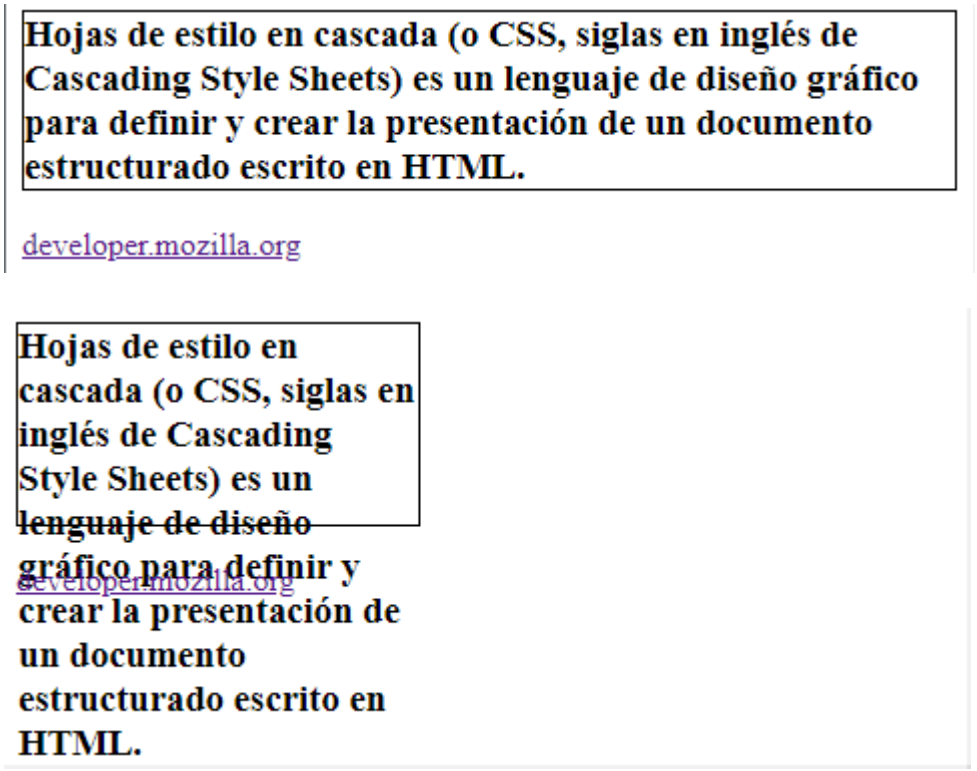


Figura 3-6 Contenido de elemento solapa a otro elemento

Para solucionar esos casos se pueden usar las siguientes propiedades:

`overflow` - propiedad para especificar cómo se mostrará el contenido que desborda el elemento. Los valores disponibles son `visible` (por defecto), `hidden` (esconde el contenido que no entra dentro de la caja), `scroll` (muestra barras laterales para desplazar el contenido), `auto` (deja que el navegador decida qué hacer con el contenido).

`overflow-x` - propiedad para especificar cómo se mostrará el contenido que desborda el elemento horizontalmente. Acepta los mismos valores que la propiedad overflow.

`overflow-y` - propiedad para especificar cómo se mostrará el contenido que desborda el elemento verticalmente. Acepta los mismos valores que la propiedad overflow.

`overflow-wrap` - propiedad para indicar si una palabra debería ser dividida en un punto arbitrario cuando no hay suficiente espacio para mostrarla en la línea. Los valores disponibles son `normal` (la línea será dividida naturalmente) y `break-word` (las palabras se dividirán en puntos arbitrarios para acomodar la línea de texto en el espacio disponible).

El tamaño de un elemento queda determinado además por el relleno (padding) que es el espacio entre el contenido del elemento y los límites de su caja, y los márgenes (margin) que es el espacio que hay alrededor de la caja.

Se puede configurar el espacio alrededor de la caja para separar el elemento de otros elementos a su alrededor (margen), además de incluir espacio entre los límites de la caja y su contenido (relleno). Las siguientes propiedades permiten definir márgenes y rellenos para un elemento.

`margin` - propiedad para configurar el margen de un elemento. Puede recibir cuatro valores que representan el margen superior, derecho, inferior, e izquierdo, en ese orden y separados por un espacio (por ejemplo, `margin: 10px 30px 10px 30px;`). Si solo se declaran uno, dos o tres valores, los otros toman los mismos valores (por ejemplo, `margin: 10px 30px` asigna 10 píxeles al margen superior e inferior y 30 píxeles al margen izquierdo y derecho). Los valores se pueden declarar independientemente usando las propiedades asociadas `margin-`

`top`, `margin-right`, `margin-bottom` y `margin-left` (por ejemplo, `marginleft: 10px;`). La propiedad también acepta el valor auto para obligar al navegador a calcular el margen (usado para centrar un elemento dentro de su contenedor).

`padding` - propiedad para configurar el relleno de un elemento. Los valores se declaran de la misma forma que para la propiedad margin, aunque también se pueden declarar de forma independiente con las propiedades `padding-top`, `padding-right`, `paddingbottom` y `padding-left` (por ejemplo, `padding-top: 10px;`).

La siguiente regla agrega márgenes y relleno a la cabecera de un documento. Debido a que se asigna solo un valor, el mismo valor se usa para definir todos los márgenes y rellenos del elemento (superior, derecho, inferior e izquierdo, en ese orden).

```
header {
  margin: 30px;
  padding: 15px;
}
```

El tamaño final de un elemento se calcula con la fórmula: tamaño + márgenes + relleno + bordes. Por ejemplo, si hay un elemento con un ancho de 200 píxeles y un margen de 10 píxeles a cada lado, el ancho del área ocupada por el elemento será de 220 píxeles. El total de 20 píxeles de margen se agrega a los 200 píxeles del elemento y el valor final se representa en la pantalla.

Es importante notar que los elementos en línea solo pueden ocupar el espacio determinado por sus contenidos. Por ejemplo, el elemento `<span>` es un elemento en línea y no puede tener un tamaño y unos márgenes personalizados.

3.3.6 Fondo

Los elementos incluyen un fondo que se muestra detrás del contenido del elemento y a través del área ocupada por el contenido y el relleno. Las propiedades para configurar el fondo con colores e imágenes son:

`background-color` - propiedad para asignar un fondo de color a un elemento.

`background-image` - propiedad para asignar una o varias imágenes al fondo de un elemento. La URL del archivo se declara con la función `url()` (por ejemplo, `url("ladrillos.jpg")`). Si se requiere más de una imagen, los valores se

separan por una coma.

`background-position` - propiedad para declarar la posición de comienzo de una imagen de fondo. Los valores se pueden especificar en porcentaje, píxeles o usando una combinación de las palabras clave `center`, `left`, `right`, `top`, y `bottom`.

`background-size` - propiedad para declarar el tamaño de la imagen de fondo. Los valores se pueden especificar en porcentaje, píxeles, o usando las palabras clave `cover` y `contain`. La palabra clave `cover` expande la imagen hasta que su ancho o su altura cubren el área del elemento, mientras que `contain` estira la imagen para ocupar toda el área del elemento.

`background-repeat` - propiedad para determinar cómo se distribuye la imagen de fondo usando cuatro palabras clave: `repeat`, `repeat-x`, `repeat-y` y `no-repeat`. La palabra clave `repeat` repite la imagen en el eje vertical y horizontal, mientras que `repeat-x` y `repeat-y` lo hacen solo en el eje horizontal o vertical, respectivamente. Finalmente, `norepeat` muestra la imagen de fondo una sola vez.

`background-origin` - propiedad para determinar si la imagen de fondo se posicionará considerando el borde, el relleno o el contenido del área del elemento. Los valores disponibles son `border-box`, `padding-box`, y `content-box`.

`background-clip` - propiedad para declarar el área a cubrir por el fondo usando los valores `border-box`, `padding-box`, y `content-box`. El primer valor corta la imagen al borde de la caja del elemento, el segundo corta la imagen en el relleno de la caja y el tercero corta la imagen alrededor del contenido de la caja.

`background-attachment` - propiedad para determinar si la imagen es estática o se desplaza con el resto de los elementos usando dos valores: `scroll` (por defecto) y `fixed`. El valor `scroll` hace que la imagen se desplace con la página, y el valor `fixed` fija la imagen de fondo en su lugar original.

`background` - propiedad que permite declarar todos los atributos del fondo al mismo tiempo.

La Figura 3-7 muestra el resultado de aplicar las siguientes reglas que usan propiedades de fondo:

```
#titulo2 {
  margin: 30px;
  padding: 15px;
  text-align: center;
  background-color: #cccccc;
}
#titulo3 {
  margin: 30px;
  padding: 15px;
  text-align: center;
  background-image: url("images/ladrillosclaros.jpg");
}
#titulo4 {
  margin: 30px;
  padding: 15px;
  text-align: center;
  background-image: url("images/ladrillosclaros.jpg");
  background-repeat: repeat-y;
}
```

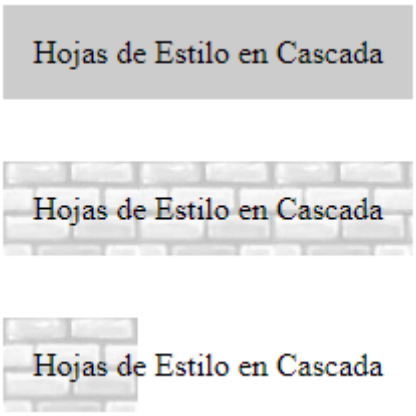


Figura 3-7 Resultado de uso de propiedad background

3.3.7 Bordes

Los elementos tienen un borde en los límites de la caja del elemento. Por defecto, los navegadores no muestran ningún borde, pero se pueden usar las siguientes propiedades para definirlo:

`border-width` - propiedad para definir el ancho del borde. Acepta hasta cuatro valores separados por un espacio para especificar el ancho de cada lado del borde (superior, derecho, inferior, e izquierdo, es ese orden). También se puede declarar el ancho para cada lado de forma independiente con las propiedades `border-top-width`, `borderbottom-width`, `border-left-width`, y `border-right-width`.

`border-style` - propiedad para definir el estilo del borde. Acepta hasta cuatro valores separados por un espacio para especificar los estilos de cada lado del borde (superior, derecho, inferior, e izquierdo, en ese orden). Los valores disponibles son `none`, `hidden`, `dotted`, `dashed`, `solid`, `double`, `groove`, `ridge`, `inset`, y `outset`. El valor por defecto es `none`, lo que significa que el borde no se mostrará a menos que se asigne un valor diferente a esta propiedad. También se puede declarar los estilos de forma independiente con las propiedades `border-top-style`, `border-bottom-style`, `border-leftstyle`, y `border-right-style`.

`border-color` - propiedad para definir el color del borde. Acepta hasta cuatro valores separados por un espacio para especificar el color de cada lado del borde (superior, derecho, inferior, e izquierdo, en ese orden). También se puede declarar los colores de forma independiente con las propiedades `border-top-color`, `border-bottom-color`, `border-left-color`, y `border-right-color`.

`border-radius` - propiedad para definir el radio del círculo virtual que el navegador utilizará para dibujar las esquinas redondeadas. Acepta hasta cuatro valores para definir los radios de cada esquina (superior izquierdo, superior derecho, inferior derecho e inferior izquierdo, en ese orden). También se puede usar las propiedades `border-top-leftradius`, `border-top-right-radius`, `border-bottom-right-radius`, y `borderbottom- left-radius` para definir el radio de cada esquina de forma independiente.

`border` - propiedad que permite declarar todos los atributos del borde al mismo tiempo. También se puede usar las propiedades `border-top`, `border-bottom`, `border-left` y `border-right` para definir los valores de cada borde de forma independiente.

Por ejemplo, la Figura 3-8 muestra el resultado de aplicar la siguiente regla que usa la propiedad border:

```
#titulo5 {
  margin: 30px;
  padding: 15px;
  text-align: center;
  border: 5px dashed #ff1122;
}
```

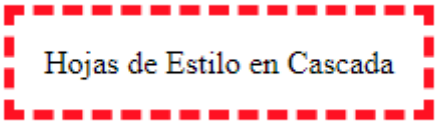


Figura 3-8 Resultado de aplicar la propiedad border

Los bordes creados se dibujan en los límites de la caja del elemento, pero también se puede enmarcar el elemento con un segundo borde que se dibuja alejado de esto límites. El propósito de estos tipos de bordes es resaltar el elemento. Algunos navegadores lo usan para demarcar texto, pero la mayoría dibuja un segundo borde fuera de los límites de la caja. Las siguientes propiedades permiten crear este segundo borde:

`outline-width` - propiedad para definir el ancho del borde. Acepta valores en cualquiera de las unidades disponibles en CSS (`px`, `%`, `em`, etc.) y también las palabras clave thin, medium, y thick.

`outline-style` - propiedad para definir el estilo del borde. Los valores disponibles son `none`, `auto`, `dotted`, `dashed`, `solid`, `double`, `groove`, `ridge`, `inset`, y `outset`.

`outline-color` - propiedad para definir el color del borde.

`outline-offset` - propiedad para definir el desplazamiento (distancia de los límites de la caja a la que se dibujará el segundo borde). Acepta valores en cualquiera de las unidades disponibles en CSS (`px`, `%`, `em`, etc.).

`outline` - propiedad para nos permite especificar el ancho, estilo y color del borde al mismo tiempo (el desplazamiento aún se debe definir con la propiedad `outline-offset`).

Por defecto, el desplazamiento se declara con el valor 0, por lo que el segundo borde se dibujará a continuación del borde de la caja. Si se desea separar los dos bordes, es necesario definir la propiedad `outline-offset`.

Las propiedades `border` y outline se limitan a simples líneas y unas pocas opciones de configuración. Para definir bordes personalizados usando imágenes se puede usar la propiedad:

`border-image` - propiedad para especificar todos los atributos del borde al mismo tiempo. Los atributos en orden son: la imagen que se usará para crear el borde con la función `url`, ancho del borde, uso de la imagen para generar el borde con valores disponibles: `repeat`, `round`, `stretch`, y `space`, valor entero o en porcentaje para definir cómo se va a cortar la imagen para representar las esquinas del borde y el desplazamiento del borde (la distancia a la que se encuentra de la caja del elemento).

3.3.8 Sombras

Con esta propiedad se añaden sombras a un elemento. Las siguientes propiedades generan sombras para la caja de un elemento y también para el texto.

`box-shadow` - propiedad para generar una sombra desde la caja del elemento. Acepta hasta seis valores. Se puede declarar el desplazamiento horizontal y vertical de la sombra, el radio de difuminado, el valor de propagación, el color de la sombra, y también se puede incluir el valor `inset` para indicar que la sombra deberá proyectarse dentro de la caja.

`text-shadow` - propiedad para generar una sombra desde un texto. Acepta hasta cuatro valores. Se puede declarar el desplazamiento horizontal y vertical, el radio de difuminado y el color de la sombra.

La Figura 3-9 muestra el resultado de aplicar la siguiente regla que usa las propiedades shadow, outline, border-radius y border-image.

```
#titulo5 {
  margin: 30px;
  padding: 15px;
  text-align: center;
```

```
    text-shadow: rgb(150, 150, 150) 3px 3px 5px;
    border: 5px dashed #ff1122;
    box-shadow: rgb(150, 150, 150) 10px 10px 20px 10px;
    outline: 2px dashed #000000;
    outline-offset: 15px;
    border-radius: 20px;
    border-image: url("images/diamantes.png") 29 round;
}
```

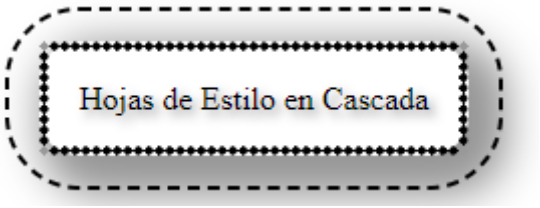


Figura 3-9 Resultado de aplicar la propiedad border

3.3.9 Display

La característica de un elemento de ser del tipo Bloque o En línea lo determina el navegador, pero es posible cambiar esa característica con la siguiente propiedad:

`display` - propiedad para definir el tipo de caja usado para presentar el elemento en pantalla. Los valores más utilizados son

`none` para eliminar el elemento.

`block` para mostrar el elemento en una nueva línea y con un tamaño personalizado.

`inline` para mostrar el elemento en la misma línea.

`inlineblock` para mostrar el elemento en la misma línea y con un tamaño personalizado.

3.3.10 Funciones para gradientes

Un gradiente se forma mediante una serie de colores que varían continuamente con una transición suave de un color a otro. Los gradientes se crean como imágenes y se agregan al fondo del elemento con las propiedades `background-image` o `background`. Para crear la imagen con el gradiente, hay las siguientes funciones:

`linear-gradient(posición, ángulo, colores)` - función para crear un gradiente lineal. El atributo posición determina el lado o la esquina desde la cual comienza el gradiente y se declara con los valores `top`, `bottom`, `left` y `right`; el atributo ángulo define la dirección del gradiente y se puede declarar en las unidades `deg` (grados), `grad` (gradianes), `rad` (radianes), o `turn` (espiras), y el atributo colores es la lista de colores que participan en el gradiente separados por coma. Los valores para el atributo colores pueden incluir un segundo valor en porcentaje separado por un espacio para indicar la posición donde finaliza el color.

`radial-gradient(posición, forma, colores, extensión)` - función para crear un gradiente radial. El atributo posición indica el origen del gradiente y se puede declarar en píxeles, en porcentaje, o por medio de la combinación de los valores `center`, `top`, `bottom`, `left` y `right`, el atributo forma determina la forma del gradiente y se declara con los valores `circle` y `ellipse`, el atributo colores es la lista de los colores que participan en el gradiente separados por coma, y el atributo extensión determina la forma que el gradiente va a adquirir con los valores `closest-side`, `closest-corner`, `farthest-side`, y `farthest-corner`. Los valores para el atributo colores pueden incluir un segundo valor en porcentaje separado por un espacio para indicar la posición donde finaliza el color.

Los gradientes se declaran como imágenes de fondo, por lo que se pueden aplicars a un elemento por medio de las propiedades `background-image` o `background`. Ejemplo:

```
background: -webkit-linear-gradient(30deg, #ffffff, #666666);
```

3.3.11 Filtros

Los filtros agregan efectos a un elemento y su contenido. CSS incluye la propiedad filter para asignar un filtro a un elemento, que pueden ser imágenes u otros elementos en el documento, y las siguientes funciones para crearlo:

`blur(valor)` - función para producir un efecto de difuminado. Acepta valores en píxeles desde 1px a 10px.

`grayscale(value)` - función para convertir los colores de la imagen en una escala de grises. Acepta números decimales entre 0.1 y 1.

`drop-shadow(x, y, tamaño, color)` - función para generar una sombra. Los atributos x e y determinan la distancia entre la sombra y la imagen, el atributo tamaño especifica el tamaño de la sombra, y el atributo color declara su color.

`sepia(valor)` - función que le otorga un tono sepia (ocre) a los colores de la imagen. Acepta números decimales entre 0.1 y 1.

`brightness(valor)` - función para cambiar el brillo de la imagen. Acepta números decimales entre 0.1 y 10.

`contrast(valor)` - función para cambiar el contraste de la imagen. Acepta números decimales entre 0.1 y 10.

`hue-rotate(valor)` - función para aplicar una rotación a los matices de la imagen. Acepta un valor en grados desde `1deg` a `360deg`.

`invert(valor)` - función para invertir los colores de la imagen y produce un negativo. Acepta números decimales entre 0.1 y 1.

`saturate(valor)` - función para saturar los colores de la imagen. Acepta números decimales entre 0.1 y 10.

`opacity(valor)` - función para cambiar la opacidad de la imagen. Acepta números decimales entre 0 y 1 (0 es totalmente transparente y 1 totalmente opaco).

3.3.12 Transformaciones

Es posible modificar la posición, tamaño y apariencia de un elemento mediante la propiedad transform. Esta propiedad realiza cuatro transformaciones básicas a un elemento: escalado, rotación, inclinación y traslación. Las siguientes son las funciones definidas para este propósito.

`scale(x, y)` - función para modificar la escala del elemento. Recibe dos parámetros, el valor x para la escala horizontal y el valor y para la escala vertical. Si solo se declara un valor, ese mismo valor se usa para ambos parámetros. Existen otras dos funciones relacionadas llamadas `scaleX()` y `scaleY()` pare especificar los valores horizontales y verticales independientemente.

`rotate(ángulo)` - función para rotar el elemento. El atributo representa los grados de rotación y se puede declarar en `deg` (grados), `grad` (gradianes), rad (radianes) o turn (espiras).

`skew(ángulo)` - función para inclinar el elemento. El atributo representa los grados de desplazamiento. La función puede incluir dos valores para representar el ángulo horizontal y vertical. Los valores se pueden declarar en `deg` (grados), `grad` (gradianes), `rad` (radianes) o `turn` (espiras).

`translate(x, y)` - función para desplazar al elemento a la posición determinada por los atributos x e y.

También se puede realizar transformaciones en tres dimensiones. Estos tipos de transformaciones se realizan considerando un tercer eje que representa profundidad y se identifica con la letra z. Las siguientes son las funciones disponibles para este propósito:

`scale3d(x, y, z)` - función para asignar una nueva escala al elemento en un espacio 3D. Acepta tres valores en números decimales para establecer la escala en los ejes x, y y z. Al igual que con transformaciones 2D, el valor 1 preserva la escala original.

`rotate3d(x, y, z, ángulo)` - función para rotar el elemento en un ángulo y sobre un eje específicos. Los valores para los ejes se deben especificar en números decimales y el ángulo se puede expresar en `deg` (grados), `grad` (gradianes), `rad` (radianes), o `turn` (espiras). Los valores asignados a los ejes determinan un vector de rotación, por lo que los valores no son importantes, pero sí lo es la relación entre los mismos. Por ejemplo, `rotate3d(5, 2, 6, 30deg)` producirá el mismo efecto que `rotate3d(50, 20, 60, 30deg)`, debido a que el vector resultante es el mismo.

`translate3d(x, y, z)` - función para mover el elemento a una nueva posición en el espacio 3D. Acepta tres valores en píxeles para los ejes x, y y z.

`perspective(valor)` - función para agrega un efecto de profundidad a la escena incrementando el tamaño del lado del elemento cercano al espectador.

Algunas transformaciones 3D se pueden aplicar directamente al elemento, pero otras requieren que primero se declaren algunas propiedades para lograr un efecto más realista:

`perspective` - propiedad para trabajar de forma similar a la función `perspective()`, pero opera en el elemento padre. La propiedad crea un contenedor que aplica el efecto de perspectiva a los elementos en su interior.

`perspective-origin` - propiedad para cambiar las coordenadas x e y del espectador. Acepta dos valores en porcentaje, píxeles, o las palabras clave `center`, `left`, `right`, `top` y `bottom`. Los valores por defecto son `50% 50%`.

`backface-visibility` - propiedad para determinar si el reverso del elemento será visible o no. Acepta dos valores: `visible` o `hidden`, con el valor `visible` configurado por defecto.

3.3.13 Transiciones

Para conseguir una animación real se requiere una transición entre los pasos del proceso. Para ello se pueden usar las siguientes propiedades:

`transition-property` - propiedad para especificar las propiedades que participan en la transición. Además de los nombres de las propiedades, podemos asignar el valor `all` para indicar que todas las propiedades participarán de la transición.

`transition-duration` - propiedad para especificar la duración de la transición en segundos (s).

`transition-timing-function` - propiedad para determinar la función que se usa para calcular los valores para la transición. Los valores disponibles son `ease`, `ease-in`, `easeout`, `ease-in-out`, `linear`, `step-start`, y `step-end`.

`transition-delay` - propiedad para especificar el tiempo que el navegador esperará antes de iniciar la animación.

`transition` - propiedad para declarar todos los valores de la transición al mismo tiempo.

3.3.14 Animaciones

La propiedad transition crea una animación básica que involucra dos estados en el proceso: el estado inicial determinado por los valores actuales de las propiedades y el estado final, determinado por los nuevos valores. Para crear una animación real, se necesita declarar más de dos estados, como los fotogramas de una película. Las siguientes propiedades permiten componer animaciones más complejas:

`animation-name` - propiedad para especificar el nombre usado para identificar los pasos de la animación. Se puede usar para configurar varias animaciones al mismo tiempo declarando los nombres separados por coma.

`animation-duration` - propiedad para determinar la duración de cada ciclo de la animación. El valor se debe especificar en segundos (por ejemplo, 1s).

`animation-timing-function` - propiedad para determinar cómo se llevará a cabo el proceso de animación por medio de una curva Bézier declarada con los valores `ease`, `linear`, `ease-in`, `ease-out` y `ease-in-out`.

`animation-delay` - propiedad para especificar el tiempo que el navegador esperará antes de iniciar la animación.

`animation-iteration-count` - propiedad para declarar la cantidad de veces que se ejecutará la animación. Acepta un número entero o el valor `infinite`, el cual hace que la animación se ejecute por tiempo indefinido. El valor por defecto es `1`.

`animation-direction` - propiedad para declarar la dirección de la animación. Acepta cuatro valores: `normal` (por defecto), `reverse`, `alternate`, y `alternate-reverse`. El valor reverse invierte la dirección de la animación, mostrando los pasos en la dirección opuesta en la que se han declarado. El valor `alternate` mezcla los ciclos de la animación, reproduciendo los que tienen un índice impar en dirección normal y el resto en dirección invertida. Finalmente, el valor `alternate-reverse` hace lo mismo que `alternate`, pero en sentido inverso.

`animation-fill-mode` - propiedad para definir cómo afecta la animación a los estilos del elemento. Acepta los valores `none` (por defecto), `forwards`, backwards, y `both`. El valor `forwards` mantiene al elemento con los estilos definidos por las

propiedades aplicadas en el último paso de la animación, mientras que `backwards` aplica los estilos del primer paso tan pronto como se define la animación (antes de ser ejecutada). Finalmente, el valor both produce ambos efectos.

`animation` - propiedad que permite definir todos los valores de la animación al mismo tiempo.

Estas propiedades configuran la animación, pero los pasos se declaran por medio de la regla `@keyframes`. Esta regla se debe identificar con el nombre usado para configurar la animación, y debe incluir la lista de propiedades que se queiere modificar en cada paso. La posición de cada paso de la animación se determina con un valor en porcentaje, donde 0 % corresponde al primer fotograma o al comienzo de la animación, y 100 % corresponde al final. Ejemplo:

```
#titulo5 {
  margin: 30px;
  padding: 15px;
  text-align: center;
  text-shadow: rgb(150, 150, 150) 3px 3px 5px;
  border: 5px dashed #ff1122;
  box-shadow: rgb(150, 150, 150) 10px 10px 20px 10px;
  outline: 2px dashed #000000;
  outline-offset: 15px;
  border-radius: 20px;
  border-image: url("images/diamantes.png") 29 round;
}
```

3.3.15 Estilo de listas

Se puede cambiar el estilo de numeracióno viñetas de una lista configurando la propiedad `list-style-type` que puede tomar los siguientes valores:

`none` - sin marcador

`disc` - círculo relleno (por defecto)

`circle` - círculo hueco

`square` - cuadrado relleno

`decimal` - 1, 2, 3, etc. (por defecto en `ol`)

`decimal-leading-zero` - 01, 02, 03, etc.

`lower-roman` - i, ii, iii, iv, v, etc.

`upper-roman` - I, II, III, IV, V, etc.

`lower-alpha` - a, b, c, d, e, etc.

`upper-alpha` - A, B, C, D, E, etc.

`lower-greek` - alpha, beta, gamma, etc.

otros: `hebrew`, `armenian`, `georgian`, `cjk-ideographic`, `hiragana`, `katakana`, `hiragana-iroha`, `katakana-iroha...`

Para configurar el estilo del marcador del elemento `<li>` para `<ul>`, `<ol>` respectivamente:

`list-style-position` - define si el marcador del elemento de la lista debe estar dentro o fuera del elemento del item. Valores `inside|outside`

`list-style-image` - usa una imagen como marcador de elementos de la lista. Valores `none|url(URLimagen)`.

`list-style` - notación abreviada, permite definir todos los parámetros en una sola línea.

3.3.16 Tablas

Existe la propiedad caption-side que aplica a las tablas que usan el elemento `<caption>` para determinar la posición vertical de ese elemento dentro de una tabla.

Los valores para esta propiedad son:

`top` para colocar el elemento <caption> encima de la tabla.

`bottom` para colocar el elemento <caption> debajo de la tabla.

3.3.17 Contenido flotante

La disposición de los elementos en una página depende de su tipo: en bloque (Block) o en línea (Inline). Los elementos Block se colocan unos sobre otros y los elementos Inline se posicionan de izquierda a derecha en la misma línea. Para cambiar la posición de un elemento del flujo normal del documento hay las siguientes propiedades:

`float` - propiedad que permite a un elemento flotar hacia un lado u otro, y ocupar el espacio disponible, incluso cuando tiene que compartir la línea con otro elemento. Los valores disponibles son `none` (el elemento no flota, valor por defecto), `left` (el elemento flota hacia la izquierda) y `right` (el elemento flota hacia la derecha).

`clear` - propiedad para restaurar el flujo normal del documento, y no permite que el elemento siga flotando hacia la izquierda, la derecha o ambos lados. Los valores.

A continuación, se presenta un ejemplo de aplicación de float para una sección. La Figura 3-10 muestra la presentación sin estilos y la Figura 3-11 la aplicación del código de estilo especificado abajo.

```
<section>
  <img src="images/miimagen.jpg" width="100">
  <p>CSS, sigla en inglés de Cascading Style Sheets (Hojas de Estilo en
Cascada) es el lenguaje de estilos utilizado para describir la presentación
de documentos HTML o XML . CSS describe como debe ser renderizado el
elemento estructurado en la pantalla, en papel, en el habla o en otros
medios.
  </p>
</section>
```

CSS, sigla en inglés de Cascading Style Sheets (Hojas de Estilo en Cascada) es el lenguaje de estilos utilizado para describir la presentación de documentos HTML o XML . CSS describe como debe ser renderizado el elemento estructurado en la pantalla, en papel, en el habla o en otros medios.

Figura 3-10 Estilos por defecto para una imagen y párrafo

```
section img {
  float: right;
  margin: 0px 10px;
}
```

CSS, sigla en inglés de Cascading Style Sheets (Hojas de Estilo en Cascada) es el lenguaje de estilos utilizado para describir la presentación de documentos HTML o XML . CSS describe como debe ser renderizado el elemento estructurado en la pantalla, en papel, en el habla o en otros medios.

Figura 3-11 Aplicación de float:right

disponibles son `none`, `left`, `right`, y `both`.

Puede ocurrir que exista solapamientos con otros elementos cuando un contenedor no tiene la dimensión adecuada para contener los elementos flotantes. Para evitar esa situación se usa la propiedad `overflow` con el valor `auto`, para forzar al navegador a calcular el tamaño del contenedor considerando todos los elementos en su interior.

3.3.18 Posicionamiento de elementos

La posición de un elemento puede ser relativa o absoluta. Con una posición relativa, que es es el modo de posicionamiento por defecto, los bloques o cajas correspondientes a cada elemento se colocan una después de la otra en el espacio designado por el contenedor. Si el espacio no es suficiente o los elementos no son flotantes, las cajas se colocan en una nueva línea.

En el posicionamiento absoluto se permite especificar las coordenadas exactas en las que se quiere posicionar cada elemento. Las siguientes son las propiedades disponibles para el posicionamiento de los elementos.

`position` - propiedad para definir el tipo de posicionamiento usado para colocar un elemento en la página. Los valores disponibles son `static` (se posiciona de acuerdo con el flujo normal del documento, posición por defecto), `relative` (se posiciona según la posición original del elemento), `absolute` (se posiciona con una posición absoluta relativa al contenedor del elemento), y `fixed` (se posiciona con una posición absoluta relativa a la ventana del navegador).

`top` - propiedad para especificar la distancia entre el margen superior del elemento y el margen superior de su contenedor. Los valores pueden especificarse en `px`, `pt`, `em` o `%`.

`bottom` - propiedad para especificar la distancia entre el margen inferior del elemento y el margen inferior de su contenedor.

`left` - propiedad para especificar la distancia entre el margen izquierdo del elemento y el margen izquierdo de su contenedor.

`right` - propiedad para especificar la distancia entre el margen derecho del elemento y el margen derecho de su contenedor.

Las propiedades `top`, `bottom`, `left`, y `right` se aplican en ambos tipos de posicionamiento, relativo o absoluto, pero trabajan de diferentes maneras. Cuando el elemento se ubica con posicionamiento relativo, el elemento se desplaza, pero el diseño no se modifica. Con posicionamiento absoluto, el elemento se elimina del diseño, por lo que el resto de los elementos también se desplazan para ocupar el nuevo espacio libre.

El orden de los elementos en una página determina la ubicación de las cajas y también qué caja va a estar por encima de las demás cuando se superponen. Con la siguiente propiedad se puede cambiar este comportamiento.

`z-index` - propiedad para definir un índice que determina la posición del elemento en el eje z. El elemento con el índice más alto se dibujará sobre el elemento con el índice más bajo. El valor puede ser negativo.

3.3.19 Columnas

Es usual disponer la información en columnas para facilitar su lectura. Haciendo uso de la propiedad `float` se puede conseguir ese efecto, pero resulta engorroso. Para facilitar la creación de columnas de un elemento, se dispone de las siguientes propiedades:

`column-count` - propiedad para especificar el número de columnas que el navegador tiene que generar para distribuir el contenido del elemento.

`column-width` - propiedad para declarar el ancho de las columnas. El valor se puede declarar como `auto` (por defecto) o en `px`, `pt`, `%`, `em`.

`column-span` - propiedad para aplicar a elementos dentro de la caja para determinar si el elemento se ubicará en una columna o repartido entre varias

columnas. Los valores disponibles son `all` (todas las columnas) y `none` (por defecto).

`column-fill` - propiedad para determinar cómo se repartirá el contenido entre las columnas. Los valores disponibles son `auto` (las columnas son completadas de forma secuencial) y `balance` (el contenido se divide en partes iguales entre todas las columnas, valor por defecto).

`column-gap` - propiedad para definir el tamaño del espacio entre las columnas. Acepta un valor en `px`, `pt`, `%`, `em`.

`columns` - propiedad para permitir declarar los valores de las propiedades `column-count` y `column-width` al mismo tiempo.

`column-rule-style` - propiedad para definir el estilo de la línea usada para representar la división entre columnas. Los valores disponibles son `hidden` (por defecto), `dotted`, `dashed`, `solid`, `double`, `groove`, `ridge`, `inset`, y `outset`.

`column-rule-color` - propiedad para especificar el color de la línea usada para representar la división.

`column-rule-width` - propiedad para especificar el ancho de la línea usada para representar la división.

`column-rule` - propiedad para permitir definir todos los valores de la línea entre columnas al mismo tiempo.

3.3.20 Modelo de caja flexible (flexbox)

El modelo de caja flexible o 'flexbox' facilita el diseño de interfaces de usuario permitiendo alinear y distribuir el espacio entre los elementos de un contenedor de modo que los elementos se comporten de forma predecible y adaptable. Un contenedor flexible expande los elementos para llenar el espacio disponible y los encoge para evitar el desbordamiento.

Un contenedor flexible es un elemento cuya posición se quiere controlar que convierte su contenido en cajas flexibles. En este modelo, cada grupo de cajas debe estar incluido dentro de otra caja que es la encargada de configurar sus características.

Para hacer flexible un elemento se asigna la propiedad `display` con los valores `flex` o `inline-flex`. El valor `flex` define un elemento Block flexible y el valor `inline-flex` define un elemento Inline flexible.

Para que un elemento dentro de un contenedor flexible se vuelva flexible, es necesario declararlo como tal. Las siguientes son las propiedades disponibles para configurar elementos flexibles.

`flex-grow` - propiedad para declarar la proporción en la cual el elemento se va a expandir o encoger. La proporción se determina considerando los valores asignados al resto de los elementos de la caja (los elementos hermanos).

`flex-shrink` - propiedad para declarar la proporción en la que el elemento se va a reducir. La proporción se determina a partir de los valores asignados al resto de los elementos de la caja (los elementos hermanos).

`flex-basis` - propiedad para declarar un tamaño inicial para el elemento.

`flex` - propiedad para nos permite configurar los valores de las propiedades `flex-grow`, `flex-shrink`, y `flex-basis` al mismo tiempo.

Las cajas flexibles se expanden o encogen para ocupar el espacio libre dentro de la caja padre. La distribución del espacio depende de las propiedades del resto de las cajas. Si todas las cajas se configuran como flexibles, el tamaño de cada uno de ellas dependerá del tamaño de la caja padre y del valor de la propiedad flex.

La propiedad flex-basis permite definir un tamaño inicial para influenciar al navegador a la hora de distribuir el espacio disponible entre las cajas, pero las cajas se pueden expandir o reducir más allá de ese valor. Están disponibles las siguientes propiedades para poner limitaciones al tamaño de una caja.

`max-width` - propiedad para especificar el ancho máximo permitido para el elemento. Unidades en `px`, `pt`, `%`, `em`.

`min-width` - propiedad para especificar el ancho mínimo permitido para el elemento. Unidades en `px`, `pt`, `%`, `em`.

`max-height` - propiedad para especificar la altura máxima permitida para el elemento. Unidades en `px`, `pt`, `%`, `em`.

`min-height` - propiedad para especificar la altura mínima permitida para el elemento. Unidades en `px`, `pt`, `%`, `em`.

Se puede cambiar la organización de las cajas mediante el uso de las siguientes propiedades:

`flex-direction` - propiedad para definir el orden y la orientación de las cajas en un contenedor flexible. Los valores disponibles son `row`, `row-reverse`, `column` y `columnreverse`, con el valor row configurado por defecto.

`order` - propiedad para especificar el orden de las cajas. Acepta números enteros que determinan la ubicación de cada caja.

`justify-content` - propiedad para determinar cómo se va a distribuir el espacio libre. Los valores disponibles son `flex-start`, `flex-end`, `center`, `space-between`, y `space-around`.

`align-items` - propiedad para alinear las cajas en el eje transversal. Los valores disponibles son `flex-start`, `flex-end`, `center`, `baseline`, y `stretch`.

`align-self` - propiedad para alinear una caja en el eje transversal. Trabaja como `align-items` pero afecta cajas de forma individual. Los valores disponibles son `auto`, `flexstart`, `flex-end`, `center`, `baseline`, y `stretch`.

`flex-wrap` - propiedad para determinar si se permiten crear múltiples líneas de cajas. Los valores disponibles son `nowrap`, `wrap`, y `wrap-reverse`.

`align-content` - propiedad para alinear las líneas de cajas en el eje vertical. Los valores disponibles son `flex-start`, `flex-end`, `center`, `space-between`, `space-around`, y `stretch`.

En el caso de tener contenedores con múltiples líneas, es posible la necesidad de alinearlas. Existe la propiedad `align-content` para alinear líneas de cajas en un contenedor flexible.

Puede tener seis valores: `flex-start`, `flex-end`, `center`, `space-between`, `space-around` y `stretch`. El valor `stretch` se declara por defecto y lo que hace es expandir las líneas para llenar el espacio disponible a menos que se haya declarado un tamaño fijo para los elementos.

3.3.21 Modelo de caja cuadrículas (grid)

El modelo de caja en cuadrícula o 'grid' es de reciente incorporación en CSS. Facilita aun más el diseño de interfaces de usuario. A diferencia de Flexbox que sirve para posicionamientos en una sola dimensión, el posicionamiento en Grid permite la disposición de elementos en 2 dimensiones, logrando establecer los valores o espacios que ocuparán los elementos dentro de una composición.

Para activar la cuadrícula grid hay que determinar sobre el elemento contenedor la propiedad `display` y especificar el valor `grid` o `inline-grid`. Este valor determina como se comportará la cuadrícula con el contenido exterior. El grid-template-columnsexterior (en bloque) y el segundo de ellos permite que la cuadrícula se muestre de izquierda/derecha (en línea) del contenido exterior. Algunas de las propiedades de grid son:

`grid-template-columns` – propiedad para definir el número de columnas que deberá ocupar la cuadrícula en unidades `px`, `pt`, `%`, `em`, `fr`. La unidad `fr` distribuye el espacio por proporciones.

`grid-column-gap` – para crear espacios entre trazas de columnas.

`grid-row-gap` – para crear espacios entre trazas de filas.

`grid-gap` - para configurar ambos a la vez. Los espacios pueden expresarse en cualquier unidad de longitud o en porcentaje, pero no en trazas.

`grid-auto-rows` y `grid-auto-columns` – para asignar un tamaño a las trazas de las cuadrículas implícitas.

`grid-template-areas` – para asignar un nombre a los diversos elementos del diseño.

3.4 EJEMPLOS DE HTML Y CSS

Los ejemplos en línea correspondientes a HTML y CSS se encuentran en el siguiente enlace: ejemplos HTML5 y CSS. Un fichero de todos los ejemplos está en el siguiente enlace: fichero comprimido de ejemplos.

3.5 EJERCICIOS

1. En la estructura DOM del siguiente documento:

```html
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8" />
    <title>Estructura DOM</title>
  </head>
  <body>
    <ul>
      <li>Elemento 1</li>
      <li>Elemento 2</li>
      <li>Elemento 3</li>
    </ul>
  </body>
</html>
```

la etiqueta ul sería el padre de:

a) html

b) los li <---- Respuesta correcta

c) del propio ul

d) body

2. Según el modelo de caja si el ancho de un div es de 500px, el padding 20 px, el borde 5px y el margen 5px, ¿cuánto mide el ancho total de la caja?:

a) 500px

b) 560px <---- Respuesta correcta

c) 530

d) 540

3. Indicar que valor de la propiedad position no es correcta:

a) absolute

b) variable <---- Respuesta correcta

c) fixed

d) relative

4. Indicar que valor de la propiedad de flex justify-content no es correcta:

a) center

b) space-between

c) space-around

d) left <---- Respuesta correcta

5. Indicar que valor de la propiedad de float no es correcta:

a) none

b) center <---- Respuesta correcta

c) right

d) left

4 REFERENCIAS

[1] C. Hock-Chuan, «HTTP (HyperText Transfer Protocol),» [En línea]. Available: https://www3.ntu.edu.sg/home/ehchua/programming/webprogramming/HTTP_Basics.html. [Último acceso: 9 2021].

[2] J. M. a. V. K. Marty Stepp, Web Programming Step by Step, Lulu, 2013.

[3] J. Duckett, HTML and CSS: Design and Build Websites, Wiley, 2011.

[4] J. N. Robbins, Learning Web Design, O'Reilly, 2018.

[5] N. D. Truman, HTML5 and CSS3: Learn HTML and CSS from Experts, Publicación Independiente, 2022.

[6] J. Gauchat, El gran libro de HTML5, CSS3 y JavaScript, Marcombo, 2017.

[7] Mozilla, «MDN Web Docs,» Mozilla Foundation., [En línea]. Available: https://developer.mozilla.org/.

[8] W. W. W. Consortium., «W3C,» [En línea]. Available: https://www.w3.org/.