Introducción

Algo de Historia…..

Cuando Tim Berners-Lee publicó la primera página web a finales de 1990 en el CERN, la web era muy distinta a como la conocemos en la actualidad. Las páginas web sólo tenían texto. En los 31 años de historia de la web, el lenguaje de marcado o etiquetado que se emplea para crear las páginas web ha evolucionado poco a poco y se han ido desarrollando sucesivas versiones. A la versión inicial del lenguaje se añadieron nuevas características como las imágenes, las tablas o los marcos que permitían dividir las páginas web en varias partes. Las páginas web fueron evolucionando y cada vez contenían más imágenes. Los diseñadores gráficos se incorporaron al desarrollo de las páginas web y se desarrolló una nueva disciplina: el diseño web. Además, los navegadores web, cada vez eran más potentes y las conexiones Internet más rápidas, así que, las páginas web cada vez mostraban más información.

Durante los primeros cinco años la web sólo servía para leer, para consumir contenidos, no existía mucha interacción con las páginas web. Sin embargo, cuando se añadieron los formularios la web comenzó a cambiar. Los formularios permitían un mayor grado de interacción entre el usuario y las páginas web. Además, el lenguaje incorporó la posibilidad de añadir nuevos tipos de contenidos a las páginas web como audio, vídeo o animaciones y, los navegadores web se volvieron más rápidos y más potentes. Todo ello ayudó a que las páginas web se transformarán en aplicaciones web, en inglés, Web Apps, que permiten realizar a través de una página web las mismas tareas que tradicionalmente se realizaban mediante un software instalado en un ordenador a partir de un CD-ROM.

Hoy en día, a través de una página web podemos enviar correos electrónicos, podemos jugar, podemos editar fotografías, podemos ver vídeos o incluso, podemos editar los vídeos, todo ello a través de una página web. En este curso te vamos a enseñar a crear páginas web. Las páginas web son el ejemplo más conocido de **hipertexto** e **hipermedia** pero existen otros sistemas que también se basan en estos dos conceptos.

Introducción

Fundamentos técnicos de una página web…

Pero, ¿qué es el **hipertexto**? Según el diccionario de la lengua de la Real Academia Española hipertexto es: conjunto estructurado de textos, gráficos etcétera, unidos entre sí por enlaces y conexiones lógicas. Un texto normal como por ejemplo, un libro, normalmente está limitado a una organización lineal o secuencial. Sin embargo, el hipertexto permite saltar de un punto a otro, en un mismo texto, o a otro texto a través de referencias. De este modo, en lugar de leer el texto de forma continua, en el hipertexto ciertos términos están relacionados y el texto se puede leer siguiendo diferentes caminos. Las relaciones en el hipertexto se establecen entre lo que se suele llamar como referencias, enlaces, vínculos o hipervínculos.

Y, ¿qué es la **hipermedia**? El término hipermedia no figura en el diccionario de la lengua española de la RAE pero, podemos buscar un término relacionado con hipermedia, multimedia. Según el diccionario de la Lengua Española, multimedia es un adjetivo que significa: que utiliza conjunta y simultáneamente diversos medios como imágenes, sonidos y texto, en la transmisión de una información.

Por tanto, un sistema multimedia es un sistema de comunicación en el que se emplean dos o más medios de comunicación distintos de forma concurrente. Un sistema multimedia puede integrar texto, voz, audio, fotografías, gráficos interactivos, vídeos, realidad virtual y otros.

Un sistema multimedia proporciona una gran riqueza y una mayor flexibilidad a la hora de comunicar la información. La calidad multimedia no está restringida al mundo de los ordenadores así, por ejemplo, un libro acompañado de un CD de música ya es una obra multimedia. Para algunos autores, hipermedia es un término que nace de la unión del hipertexto más la multimedia, por tanto, **si juntamos las definiciones de hipertexto y multimedia, podemos obtener la siguiente definición de hipermedia**: conjunto estructurado de diversos medios como textos, gráficos, imágenes y sonidos unidos entre sí por enlaces y conexiones lógicas para la transmisión de una información.

Introducción

Fundamentos técnicos de una página web…

Si la multimedia proporciona una gran riqueza a la información, el hipertexto aporta una estructura que permite que la información pueda presentarse y explorarse siguiendo distintas secuencias de acuerdo a las necesidades y preferencias del usuario. Existen muchos sistemas que se basan en el hipertexto y la hipermedia pero, la web es el sistema más conocido y por eso la web se ha convertido en sinónimo de hipertexto e hipermedia.

¿Qué son los lenguajes de **marcado**? Los lenguajes de **etiquetas**, también conocidos como lenguajes de marcado o de marcas, son los que nos permiten estructurar un documento mediante el uso de etiquetas. Los lenguajes de etiquetas no se identifican con los de programación; esto ocurre principalmente porque los lenguajes de etiquetas no definen algunos aspectos básicos presentes en los lenguajes de programación, como es el caso de funciones aritméticas o el uso de variables, por citar algunos ejemplos.

Cuando queremos crear o dar formato a una página web, se nos presentan múltiples herramientas, pero todas tienen algo en común: el lenguaje de marcas. Algunas características fundamentales de este lenguaje son:

-Texto Plano

Puede ser editado por un usuario con un sencillo editor de textos, sin perjuicio de que se puedan utilizar programas más sofisticados que faciliten el trabajo. Al tratarse solamente de texto, los documentos son independientes de la plataforma, sistema operativo o programa con el que fueron creados.

-Compacidad

Las instrucciones de marcado se entremezclan con el propio contenido en un único archivo o flujo de datos.

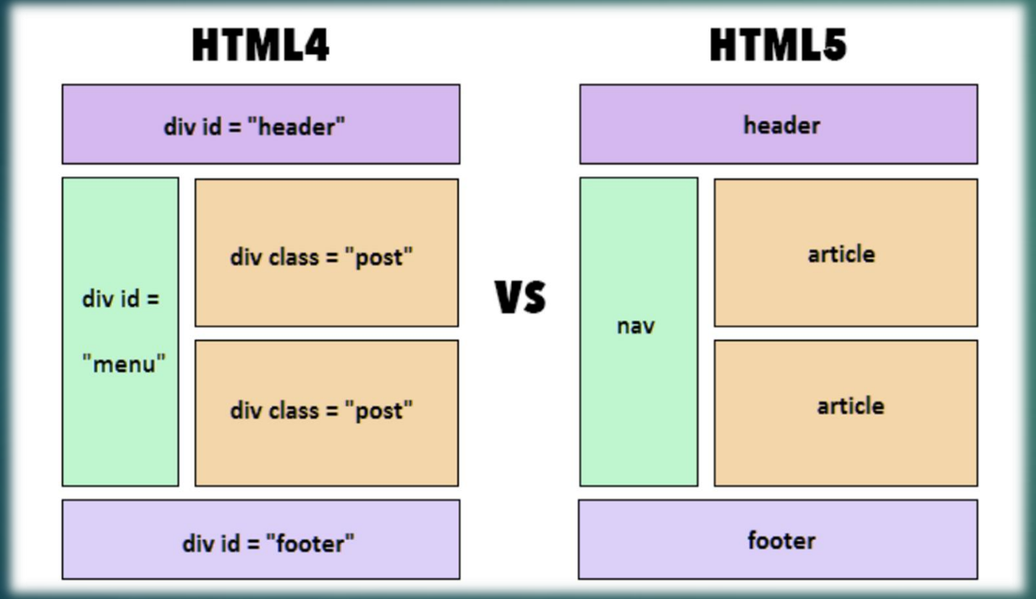
-Facilidad de procesamiento

Facilitan el interoperatividad o el intercambio de información entre distintos programas, flexibilidad de poder ser adaptados a unas necesidades concretas..

Introducción

Fundamentos técnicos de una página web…

HTML ha tenido muchas actualizaciones a lo largo del tiempo, y la última versión de HTML es HTML5. En HTML5 se destacan sus características semánticas, las posibilidades multimedia que incorpora, las nuevas funciones para formulario y las características que se definen para poder integrarse con tecnologías que permitirán abrir una nueva etapa en Internet, en lo que se refiere a la arquitectura de las aplicaciones.



Ventajas de HTML5

- Nueva estructura de etiquetas mejorada, esta nueva estructura permite definir por separado el encabezado, la barra de navegación, las secciones de la página web, los textos del sitio, los diálogos y el pie de página de los sitios web.
- Inclusión de las etiquetas video y audio, dicha etiqueta soporta de manera eficiente y estable cualquier opción de ejecución de video y audio, sin generar errores o incluir código flash en nuestro sitio web.

Introducción

Tecnologías de software para el desarrollo de páginas web…

Para poder aprovechar al máximo el potencial de HTML5 es fundamental también comprender el rol de las tecnologías que interactúan con este lenguaje de etiquetas y de qué manera deben integrarse.

A continuación, nos centraremos en conocer las características principales de CSS, JavaScript y AJAX.

- Las hojas de estilo en cascada, tal es su traducción del inglés Cascading Style Sheets (**CSS**), tienen como función establecer reglas de representación de un documento en un medio o dispositivo. Mediante estas reglas podremos establecer medidas, colores o cualquier otra característica de representación de una página web, para que se vea reflejada en una pantalla de monitor, de un dispositivo móvil, una tablet, una impresora, un dispositivo braille o un televisor. La función principal de CSS es, por lo tanto, la de permitir separar el contenido y la estructura que se define en un documento HTML, de la representación, que queda a cargo de las hojas de estilos.
- **JavaScript** es un lenguaje multiparadigma que requiere de un intérprete para ser ejecutado, permite la creación de páginas dinámicas, con código que puede ejecutarse desde el lado cliente, alivianando la tarea del servidor y disminuyendo la cantidad de peticiones que se le hagan. Por sus características, resulta útil para validación de formularios, mostrar y aplicar efectos, y exhibir avisos en pantalla.
- El término **AJAX** es un acrónimo que proviene de Asynchronous JavaScript And XML, que, al castellano, podría traducirse como JavaScript asíncrono y XML. Justamente este es el punto fuerte de AJAX: poder trabajar con datos de manera asincrónica, valiéndose de JavaScript como lenguaje del lado cliente para manejar datos que le llegan desde el servidor. De esta manera, el motor de AJAX trabaja como un intermediario entre el cliente y el servidor, pero, en lugar de demorar procesos, los administra de tal manera que es posible, por ejemplo, la recarga de solo algunas partes de una página web. Esta posibilidad cambia el paradigma de la necesidad de una recarga completa de la página y permite construir aplicaciones web más potentes, emulando incluso a muchas de las soluciones que se veían posibles solo en software de escritorio.

Introducción

Funcionamiento del navegador…

Es importante que todo el mundo tenga acceso a la web, pero también es fundamental que todos comprendamos las herramientas que utilizamos para acceder a ella. Usamos navegadores web como Mozilla Firefox, Google Chrome, Microsoft Edge y Apple Safari todos los días, pero ¿entendemos qué son y cómo funcionan?

- Un navegador web te lleva a cualquier lugar de Internet. Recupera información de otras partes de la web y la muestra en tu escritorio o dispositivo móvil. La información se transfiere mediante el Protocolo de Transferencia de Hipertexto (HTTP), que define cómo se transmiten el texto, las imágenes y el video en la web. Esta información debe compartirse y mostrarse en un formato consistente para que las personas que utilizan cualquier navegador, en cualquier parte del mundo, puedan ver la información.
- Lamentablemente, no todos los fabricantes de navegadores eligen interpretar el formato de la misma manera. Para los usuarios, esto significa que un sitio web puede funcionar y verse diferente. Crear una experiencia consistente entre navegadores, para que los usuarios puedan disfrutar de internet, sin importar el navegador que elijan, se llama estándares web.
- Cuando el navegador web obtiene datos de un servidor conectado a Internet, utiliza un software llamado **motor** de **renderizado** para traducir esos datos en texto e imágenes (**Representación Interactiva**). Estos datos están escritos en "lenguaje de marcas de hipertexto" (HTML) y los navegadores web leen este código para construir lo que vemos, escuchamos y experimentamos en Internet.
- Los hipervínculos permiten a los usuarios seguir una ruta a otras páginas o sitios en la web. Cada página web, imagen y video tiene su propio Localizador Uniforme de Recursos (URL), que también se conoce como dirección web. Cuando un navegador visita un servidor en busca de datos, la dirección web le dice al navegador dónde buscar cada elemento que se describe en el html, que luego le dice al navegador dónde situarlo en la página web.

Un editor de texto es un programa informático que permite crear, abrir, ver y modificar el contenido de un documento. Los editores de texto se proporcionan con los sistemas operativos y paquetes de desarrollo de programación. El ejemplo de editor de texto más simple es Bloc de Notas, que está integrado en el software de Windows. Puedes crear, editar y ver archivos de texto simples con el bloc de notas. Un editor de texto simple tiene las siguientes características:

- Cortar, copiar y pegar.
- Dar formato a un texto.
- Buscar y reemplazar.
- Deshacer y Rehacer.

Los editores de texto son utilizados por una amplia variedad de propósitos y por una gran variedad de personas. Cualquiera que necesite escribir, editar o leer texto puede simplemente usar editores de texto como el bloc de notas. Los programadores de software, los desarrolladores web y de aplicaciones utilizan editores de texto para leer, escribir y editar el código fuente de muchos lenguajes de programación y marcado. El uso del editor de texto para el código fuente es el propósito principal de los editores de texto y hay muchas otras características del software de edición de texto creadas para ayudar a estos usuarios a leer y escribir código. Como vamos a aprender el lenguaje HTML también necesitamos un editor de texto.

Un editor de texto en línea son editores de textos pero al cual puedes acceder desde cualquier navegador web. Algunos de estos editores de textos te permiten crear cuentas personales de tal manera que no necesitas guardar en el ordenador en el que trabajas ningún documento, todo se guarda en tu cuenta personal y puedes tener accesos en cualquier momento a tu proyecto o trabajo desde cualquier ordenador y desde cualquier lugar donde esté. Una de las grandes ventajas que tiene con respecto a los editores de texto que instalas en el ordenador es que de alguna manera puedes hacer a un lado las restricciones de hardware o software del ordenador para editar, modificar, crear algún archivo o documento

Estos son algunos ejemplos de editores de texto:

- Bloc de notas, también conocido como Notepad (en inglés).
- Texto sublime
- Visual Studio Code (o VScode)
- Atom,
- Notepad ++
- VIM
- Ultraedit
- CodePen
- jsFiddle
- Glitch
- Adobe Dreamweaver CC
- TextEdit
- UltraEdit
- Brackets
- Komodo Edit

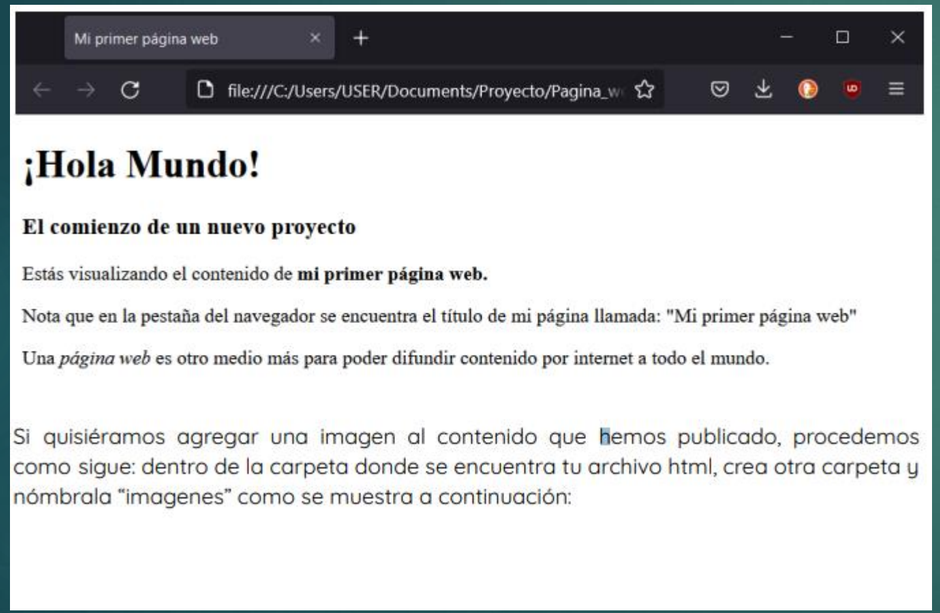
Abre el editor de textos predeterminado del sistema. En este caso nosotros usaremos el Bloc de notas de Windows. Y coloca el siguiente texto dentro del editor.

```
<!DOCTYPE html>
<html>
    <head>
        <title>Mi primera página web</title>
    </head>
    <body>
        <h1>¡Hola Mundo!</h1>
        <h3>El comienzo de un nuevo proyecto</h3>
        <p>Estás visualizando el contenido de <strong> mi primera página web.</strong></p>
        <p>Nota que en la pestaña del navegador se encuentra el título de mi página llamada:"Mi primera página web"</p>
        <p>Una <i>página web</i> es otro medio más para poder difundir contenido por internet a todo el mundo.</p>
    </body>
</html>
```

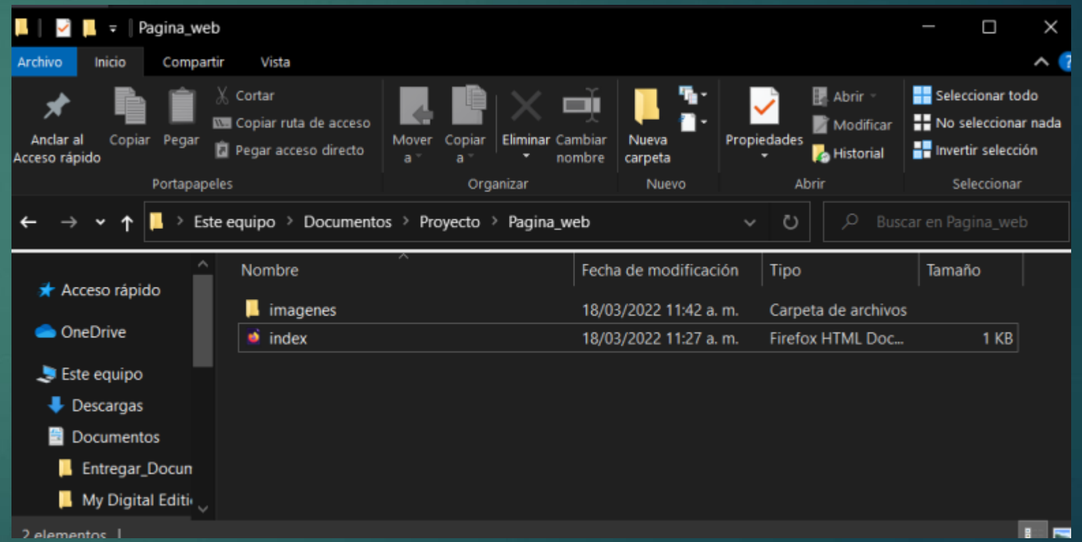
Guarda el archivo en una carpeta nueva, agregando al final la extensión .html, en este ejemplo nosotros pusimos de nombre al archivo “index.html”

Al guardar en archivo en la carpeta nueva que creaste, visualizará el archivo el cual podrás ejecutar dando doble click o abriendo desde tu navegador predefinido o favorito.

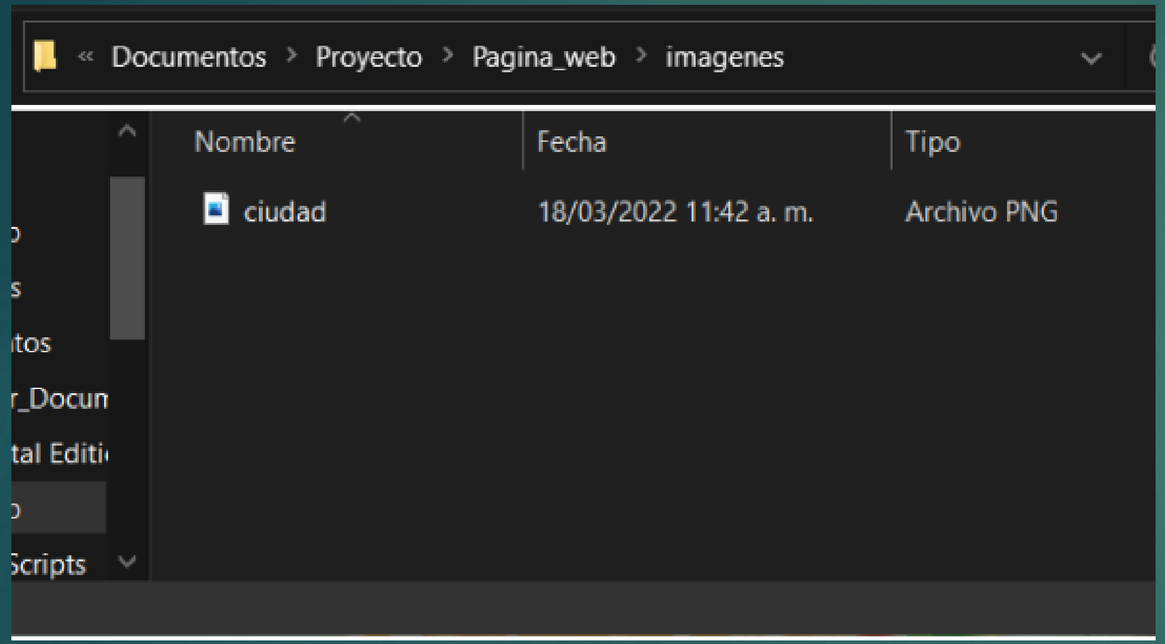
El resultado final al abrir el archivo html sería como lo siguiente:



Si quisiéramos agregar una imagen al contenido que hemos publicado, procedemos como sigue: dentro de la carpeta donde se encuentra tu archivo html, crea otra carpeta y nómbrala "imagenes" como se muestra a continuación:



Descarga o ubica en los archivos de tu computadora la imagen que deseas agregar. Copia la imagen y pégala en la carpeta de "imagenes" como se muestra; en nuestro caso usamos la imagen llamada "ciudad" que tiene extensión .PNG:



Estos son algunos formatos de imagen que puedes incorporar dentro del código html5:

- JPG
- PNG
- GIF
- SVG
- WEBP

Modifica el código html agregando las líneas de código html que se muestran a continuación:

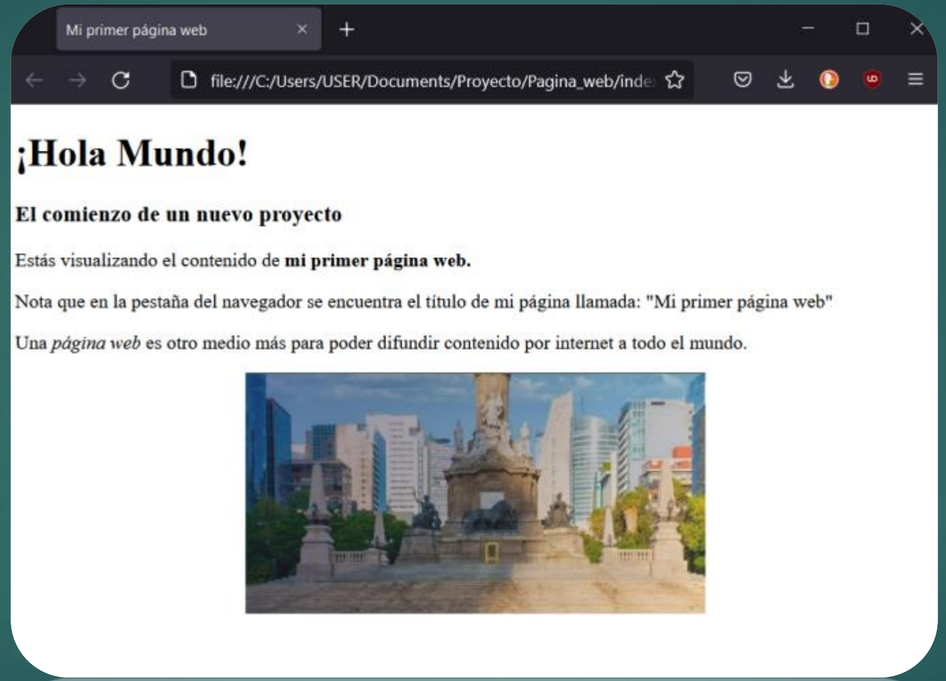
```
<!DOCTYPE html>
<html>
	<head>
		<title>Mi primera página web</title>
		<style>
			img {
			display: block;
			margin: auto;
			width: 50%;
			}
		</style>
	</head>
	<body>
		<h1>¡Hola Mundo!</h1>
		<h3>El comienzo de un nuevo proyecto</h3>
		<p>Estás visualizando el contenido de <strong> mi primera página web.</strong></p>
		<p>Nota que en la pestaña del navegador se encuentra el título de mi página llamada:"Mi primera página web"</p>
		<p>Una <i>página web</i> es otro medio más para poder difundir contenido por internet a todo el mundo.</p>
	<img src="imagenes/ciudad.PNG">
	</body>
</html>
```

Uno de los buenos hábitos para desarrollar nuestra página web es tener en distintas carpetas distintos tipos de archivo, por ejemplo, los archivos relacionados a imágenes en una carpeta, en otra los relacionados a multimedia, en otra a lo referente a los estilos (CSS), ejemplo…

```
img {
	display: block;
	margin: auto;
	width: 50%;
	}
```

Este fragmento de código css puede colocarse en un archivo externo con extensión css, como por ejemplo **style.css**

Guarda las modificaciones del archivo que contiene el código html. Refresca o cierra y abre la ventana del explorador que muestra el contenido de tu página web. Deberás visualizar algo como lo que sigue:



Estructura básica de un documento HTML

Antes que nada, debes saber que el código HTML se compone de etiquetas o marcas. Las etiquetas en HTML son palabras clave que se escriben entre los signos <> y que el navegador entiende. Normalmente las etiquetas se componen de una etiqueta de apertura (entre los signos <>), una etiqueta de cierre (entre los signos) y un contenido. El contenido puede ser texto u otras etiquetas. Aunque existen algunas etiquetas que no tienen ni contenido ni etiqueta de cierre, son una excepción. La sintaxis sería la siguiente:

```
<etiqueta>
    contenido etiqueta
</etiqueta>
```

- **<a>:** Define y da forma a un ancla.
- **<abbr>:** Introduce una abreviatura.
- **<address>:** Especifica un elemento de dirección.
- **<article>:** Define una parte independiente del contenido de un documento, como una entrada de blog o un artículo de periódico.
- **<aside>:** Define el contenido aparte del contenido principal. Representado principalmente como barra lateral.
- **<audio>:** permite introducir un archivo de audio.
- **
:** Inserta un solo salto de línea.
- **<botón>:** Especifica un botón pulsador.
- **<canvas>:** Se utiliza para representar gráficos de mapa de bits dinámicos sobre la marcha, como gráficos o juegos.
- ** :** Define una sección en un documento HTML 5
- **<form>:** Para introducir formularios.
- **<footer>:** Representa el pie de una sección, con información acerca de la página/sección que poco tiene que ver con el contenido de la página, como el autor, el copyright o el año.
- **<header>:** representa un grupo de artículos introductorios o de navegación. Está destinado a contener por lo general la cabecera de la sección (un elemento h1-h6 o un elemento hgroup), pero no es necesario.
- **<hgroup>:** Sirve para crear el encabezado de una sección.
- **<h1> a <h6>:** Especifica el encabezado 1 al encabezado 6.
- **<img>:** Introduce una sección para una imagen.
- **<map>:** Una etiqueta destinada a elaborar mapas de imagen.
- **<nav>:** Define una sección del documento destinada a la navegación.
- **<p>:** Etiqueta usada para escribir párrafos de texto.
- **<script>:** Sirve para introducir un script.
- **<section>:** Define una sección genérica para un documento.

Existen otros tipos de etiquetas para crear una página web en HTML5 que nos permiten enlazar o encuadrar otros elementos dentro de las páginas web. Algunas de las más usadas son:

Estructura básica de un documento HTML

HTML es un lenguaje que se compone por elementos que permiten definir la estructura del documento. Estos elementos son los que nos posibilitan determinar cómo estará armada la página y sus secciones. Las etiquetas nos brindan la oportunidad de definir los elementos en el código.

```
<elemento1>
    contenido del elemento 1
</elemento1>

<elemento2>
    contenido del elemento 2
</elemento2>
```

Todos los elementos se dividen en dos categorías:

Elementos en bloque.

Estos son los elementos que estructuran la parte principal de la página web, dividiendo una página en bloques coherentes. Un elemento a nivel de bloque siempre comienza con una nueva línea y ocupa todo el ancho de la página web, de izquierda a derecha.

Los siguientes son algunos de los elementos en bloque en HTML:

<address>, <article>, <aside>, <blockquote>, <canvas>, <dd>, , <dl>, <dt>, <fieldset>, <figcaption>, <figure>, <footer>, <form>, <h1> hasta <h6>, <header>, <hr>, <li>, <main>, <nav>, <noscript>, <ol>, <output>, <p>, <pre>, <section>, <table>, <tfoot>, <ul> y <video>.

```
<p>
    Este párrafo es un elemento en bloque.
</p>
```

Elementos en línea.

Los elementos en línea son aquellos elementos que diferencian la parte de un texto dado y le proporcionan una función particular. Estos elementos no comienzan con una nueva línea y toman el ancho según el requisito. Los elementos en línea se utilizan principalmente con otros elementos.

Los siguientes son algunos de los elementos en linea en HTML:

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

```
<p>
    Texto normal y... <b>Texto en negrita</b>
</p>
```

Estructura básica de un documento HTML

Un atributo en HTML son palabras especiales utilizadas dentro de la etiqueta de apertura, para controlar el comportamiento del elemento. Los atributos HTML brindan información adicional sobre el elemento. Cada atributo HTML tiene su nombre y valor:

```
<elemento atributo="valor">contenido del elemento</elemento>
```

- **id =" "** : Para identificar un elemento único.
- **class =" "** : Identificador múltiple.
- **align =" "** : Alineación de contenido.
- **border =" "** : Para darle borde al contenido.
- **style =" "** : Para darle un estilo al contenido.
- **background-color =" "** : Para color de fondo.
- **href =" "** : Para enlaces html.
- **height =" "** : Para determinar altura.
- **width =" "** : Para determinar ancho.
- **src=" "** : Para imágenes.

Algunos elementos no tienen etiqueta final ni contenido, estos elementos se denominan elementos vacíos o elementos de cierre automático. Veamos algunos atributos y cuál es su función.

Ejemplo. Usamos el *elemento* **<a>** con el *atributo* **href** y *valor* **https://pilares.cdmx.gob.mx/inicio**

```
<a href="https://pilares.cdmx.gob.mx/inicio">Esto es un link</a>
```

Ejemplo. Usamos el *elemento* **<img>** con el *atributo* **scr** y el *valor* **pilares.jpg**

```
<img scr="pilares.jpg" height="400" width="600">
```

El ejemplo anterior también tiene *atributos* de height y width, que definen el alto y el ancho de la imagen en la página web.

En HTML5, también puede omitir el uso de comillas alrededor de los valores de los atributos.

```
<a href=https://pilares.cdmx.gob.mx/inicio>Esto es un link</a>
```

```
<img scr=pilares.jpg height=400 width=600>
```

Los atributos globales en HTML son aquellos atributos que son comunes para todos los elementos de HTML. Los atributos globales se pueden usar con todos los elementos, aunque es posible que no tengan ningún efecto en algunos elementos. A continuación se muestra la lista completa de atributos globales con su descripción:

Atributos	Valor	Descripción
accesskey	character	Se utiliza para generar atajos de teclado para el elemento actual.
class	classname	Se utiliza para proporcionar el nombre de clase para el elemento actual. Se utiliza principalmente con la hoja de estilo.
Contenteditable	true false	Determina si el contenido dentro de un elemento es editable o no.
contextmenu	menu_id	Define el id para el elemento <menu> que se utiliza como menú contextual (aparece un menú al hacer clic con el botón derecho) para un elemento.
data-*	somevalue	Se utiliza para almacenar datos privados específicos de elementos a los que se puede acceder mediante Javascript.
dir	rtl ltr auto	Especifica la dirección del contenido dentro del elemento actual.

Atributos	Valor	Descripción
draggable	True false auto	Especifica si el contenido dentro de un elemento se puede mover o no mediante la API de arrastrar y soltar.
dropzone	copy move link	Especifica la acción que se realiza en el elemento arrastrado cuando se suelta, por ejemplo, si se copia, se mueve o se vincula.
hidden		Se utiliza para ocultar el elemento de la vista.
id	id	Especifica una identificación única para el elemento. Se puede utilizar con CSS y JavaScript.
lang	language_code	Especifica el idioma principal para el contenido de un elemento.
style	style	Se utiliza para aplicar CSS en línea al elemento actual.
spellcheck	true false	Especifica si el contenido debe revisarse en busca de errores ortográficos o no.
tabindex	number	Determina el orden de tabulación de un elemento.
title	text	Se utiliza para proporcionar el título, el nombre o alguna información adicional sobre el elemento.

Semántica…
En 2004, Ian Hickson, el autor de la especificación de HTML5, analizó 1.000.000.000 de páginas web utilizando el motor de Google, intentando identificar la manera en la que la web real estaba construida. Uno de los resultados de este análisis, fue la publicación de una lista con los nombres de clases más utilizados. Este estudio revela que los desarrolladores utilizan clases o IDs comunes para estructurar los documentos. Esto llevó a considerar que quizás fuese una buena idea crear etiquetas concretas para reflejar estas estructuras: **la web semántica.**

En cualquier idioma, es esencial comprender el significado de las palabras durante la comunicación. Y si se trata de una comunicación informática, se vuelve aún más crítica. HTML5 proporciona elementos semánticos que facilitan la comprensión del código.

En este sentido, **la semántica** define el significado de palabras y frases, es decir, tener elementos semánticos es equivalente a tener **elementos con significado**. Los elementos semánticos tienen un significado simple y claro tanto para el navegador como para el desarrollador exceptuando elementos como por ejemplo: .

¿Por qué usar elementos semánticos? En primer lugar, porque es mucho más fácil de leer. Esta es probablemente la primera cosa que notará al mirar el primer bloque de código usando elementos semánticos. Como programador podría estar leyendo cientos o miles de líneas de código. Cuanto más fácil sea leer y comprender ese código, más fácil será su trabajo

Tiene mayor accesibilidad. No serás el único que encuentre los elementos semánticos más fáciles de entender. Los motores de búsqueda y las tecnologías de asistencia (como lectores de pantalla para usuarios con discapacidad visual) también podrán comprender mejor el contexto y el contenido de tu sitio web, lo que significa una mejor experiencia para los usuarios.

Estructura básica de un documento HTML

En general, los elementos semánticos también conducen a un código más consistente.

HTML5 introduce elementos específicos para poder definir secciones del documento y también características que pretenden hacer de la semántica una capacidad importante para el lenguaje.

En lo que se refiere a la estructura del documento, en HTML4 estábamos acostumbrados a definir las partes del cuerpo mediante el uso de la etiqueta . El problema se planteaba en la imposibilidad de asignarles la semántica correspondiente a las diferentes partes. Por ejemplo, si bien podíamos aplicar una id con el valor nav o footer (barra de navegación y pie), esto no era más que el valor de un atributo y no le daba un significado semántico diferente al elemento.

A partir de HTML5 se definen etiquetas que nos permiten estructurar el cuerpo de una página con una semántica específica para cada elemento: **<header> <hgroup> <nav> <section> <article> <aside> <footer> <figure> <figcaption> <time> <details> <summary> <mark>** etiquetas que nos permiten definir una estructura semántica en nuestros documentos HTML y a la vez un código mucho más claro de leer, tanto para los desarrolladores como también para los agentes informáticos (robots de los buscadores, dispositivos electrónicos para facilitar las características de accesibilidad).

Entre los atributos que soportan los elementos **<header>, <hgroup>, <nav>, <section>, <article>, <footer>, <figure> y <figcaption>**, encontramos algunos que se heredan de la versión 4 de HTML: **accesskey, class, dir, id, lang, style, tabindex y title.**

Estructura básica de un documento HTML

El atributo rel no sirve para agregarles información a los enlaces. Con <link>, le brindamos más información al navegador, con <a>, le estaremos incluyendo datos que podrán ser útiles para los buscadores. El atributo rel nos permite establecer la relación entre el documento actual y el que se está vinculando. La relación se da a través de los valores que adquiere el atributo rel: alternate, archives, author, bookmark, external, first, help, icon, index, last, licence, next, nofollow, noreferrer, pingback, prefetch, prev, search, sidebar, stylesheet, tag, up. Los valores bookmark, external, nofollow y noreferrer no son soportados por <link>. icon, pingback, prefetch y stylesheet no pueden utilizarse con <a> ni con <area>.

A continuación comparamos dos códigos HTML, uno usando una versión anterior a HTML5 y el otro usando HTML5.

```
<div id="header"> <!--Inicio header -->
        <h1>Header</h1>
</div><!--Fin header -->
<div id="nav"> <!--Inicio Nav -->
        <p>Nav</p>
</div> <!--Fin nav -->
<div id="section"> <!--Inicio Section -->
        <p>section</p>
</div> <!--Fin section -->
<div id="aside"> <!--Inicio aside -->
        <p>aside</p>
</div><!--Fin aside -->
<div id="footer"> <!--Inicio footer -->
        <p>footer</p>
</div> <!--Fin footer -->
```

Ahora usando HTML5:

```
<header>
        <h1>Header</h1>
</header>
<nav>
        <p>Nav</p>
</nav>
<section>
        <p>section</p>
</section>
<aside>
        <p>aside</p>
</aside>
<footer>
        <p>footer</p>
</footer>
```

Estructura básica de un documento HTML

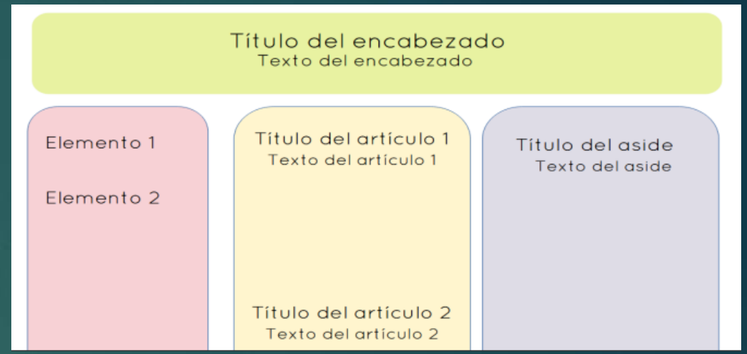
La semántica de HTML5 aporta mucha claridad

Vamos ahora a estructurar una página en HTML5 desde cero. En este punto ya debimos de haber realizado maquetas del sitio, analizado la necesidades, así como haber verificado aspectos de usabilidad, accesibilidad y navegabilidad. Teniendo esto, partimos a andar en el camino de estructurarlo con HTML5. Una recomendación que se hace en estos casos es realizar primero la estructura HTML de las secciones más importantes del cuerpo del documento (cabecera, contenido principal, barra de navegación, pie, etcétera) y, luego, ir agregando los elementos que se incorporan en ellas.

A continuación, veremos el ejemplo de un código básico de estructura semántica de HTML5. En este caso, trabajaremos el código del cuerpo del documento con un encabezado <header>, una barra de navegación <nav>, seguido de un bloque principal denominado <section> que contiene dos artículos <article>, una barra con información adicional <aside> y también un pie <footer>:

```
<header>
  <h1>Título del encabezado</h1>
   <p><strong>Texto del encabezado</strong></p>
</header>

<nav>
  <ul>
   <li>Elemento 1</li>
   <li>Elemento 2</li>
  </ul>
</nav>


<section>
    <article>
      <h2>Título del artículo 1</h2>
      <p>Texto del artículo 2</p>
    </article>

    <article>
       <h2>Título del artículo 2</h2>
       <p>Texto del artículo 2</p>
    </article>
</section>

<aside>
  <h3>Título del Aside</h3>
  <p>Contenido del Aside</p>
</aside>

<footer>
  <p>Texto de pie de página</p>
</footer>
```

Visualmente, con la ayuda de CSS, tendría una forma como se muestra a continuación:



Estructura básica de un documento HTML

Antes de continuar, vale aclarar que esta es simplemente la estructura HTML, y que todo lo relacionado con la apariencia y modo de representación de los elementos en pantalla, lo manejaremos a través de CSS, cuyas reglas veremos en detalle en el capítulo siguiente. Volviendo a nuestro ejemplo, vamos a comenzar con el <header>. En este caso, aplicamos la etiqueta <h1> al título del sitio o blog, que aparece en su página principal, ya que es el elemento de texto más importante. Debajo, utilizamos la etiqueta <p> y la destacamos con para el texto de descripción que suelen tener los sitios.Algunos sitios utilizan imágenes en el encabezado; en esos casos, por una cuestión de accesibilidad es recomendable utilizar en la etiqueta <img> los atributos alt y title para describir el texto relativo a dicha imagen. Otra alternativa para resolver el encabezado, dependiendo de la necesidad de nuestra estructura, consiste en el uso de la siguiente semántica:

```
<header>
  <hgroup>
    <h1>Título del encabezado</h1>
    <h2>Subtítulo del encabezado</h2>
  </hgroup>
</header>
```

Ahora pasamos a la barra de navegación , que bien podría estar a la izquierda de la sección de contenido o por encima de ella. En general, si creamos un menú, podremos estructurarlo con una lista, como en este caso, en la que los ítems están constituidos por la enumeración de los elementos. Luego, por medio de CSS, le daremos el estilo que deseemos.

En este ejemplo, <section> contiene el resumen de los artículos. En esta estructura simplificada, vemos dos <article>; cada uno de ellos cuenta con un título <h2> y un texto en un párrafo <p> para la breve descripción de la noticia. En este caso, el <aside> ha sido simplificado en un título <h3> y un texto en un <p>. Tengamos en cuenta que el pie de página, el cual se encuentra definido con el <footer>, puede tener un texto sobre el autor, copyright, enlaces internos o hacia sitios amigos, etcétera. La estructura que hemos visto aquí puede adaptarse, fácilmente, a cualquier tipo de necesidad, ya sea un e-commerce, una red social, un foro o cualquier otra opción que podamos imaginar. Luego será solo cuestión de incorporar los elementos para personalizar nuestro proyecto y, con ellos, terminar de darle la estructura deseada a cada página.

Estructura básica de un documento HTML

En ésta imagen muestra el contenido de la pagina web. Colocando elementos semánticos de HTML5 dentro de cada uno de los recursos según corresponda.

Estás son algunos elementos y atributos que trabajan en conjunto con las etiquetas mostradas en la imagen.

figcaption: Define el titulo para un elemento <figure>.
header: Define el contenido que debe considerarse la información introductoria de una página o sección.
time: Define una fecha/hora.
section: Usar esta etiqueta es una forma de agrupar contenido cercano de un tema similar.
mark: Define el texto marcado/resaltado.
figure: Especifica contenido independiente, como ilustraciones, diagramas, fotos, listas de códigos, etc.
details: Define detalles adicionales que el usuario puede ver u ocultar.
nav: Los enlaces del menú de navegación principal se colocarían todos dentro de esta etiqueta.
aside: Define el contenido que es menos importante. A menudo se usa para barras laterales, áreas que agregan información completamente per no vital.
article: Define el contenido autónomo que podría ser independiente de la pagina o el sitio en el que se encuentra. Por ejemplo, una entrada de blog.
main: El cuerpo de una pagina debe ir dentro de esta etiqueta, no en las barras laterales ni en la navegación principal, Debe haber solo una por pagina.
footer: Define un piel de página para un documento o sección.

Lenguaje HTML

Cuando empezamos a escribir código HTML en el editor, primero tenemos que dar el Tipo de Documento. Este tipo de documento le dice al navegador qué tipo de código está escrito en el archivo. Este tipo de documento se denomina como doctype en HTML. Todo documento HTML debe comenzar con doctype. La sintaxis para escribir doctype es la siguiente: `<!doctype>`

Tenemos que especificar aún más el nombre y la versión de HTML en la etiqueta anterior. Ya que usaremos HTML 5 en estas notas, la sintaxis de HTML5 doctype es la siguiente: `<!doctype html>`

No tiene ni contenido ni etiqueta de cierre, simplemente indica al navegador que el documento es de tipo HTML5

Justo después de esta irá la etiqueta <html>. La etiqueta <html> sirve para indicar que el documento es un documento HTML y el navegador lo interpretara como tal. Esta segunda etiqueta sí que tiene etiqueta de cierre (</html>) y englobará todo el contenido de la página.

```
<!doctype html>
<html>
  ...contenido...
</html>
```

```
<!doctype html>
<html>
    <head>
        ...elementos de la cabecera...
    </head>
</html>
```

Lenguaje HTML

Dentro de las etiquetas <head> y </head>, colocaremos las etiquetas <title> y </title> de la siguiente manera:

```
<!doctype html>
<html>
    <head>
        <title> Mi primer titulo </title>
    </head>
</html>
```

<title> contiene el título de la página y esta se muestra en la pestaña de los navegadores, más no se ve en el cuerpo de la página.

```
<!doctype html>
<html>
    <head>
        <title> Mi primer titulo </title>
    </head>
    <body>
        ...cuerpo de la página...
    </body>
</html>
```

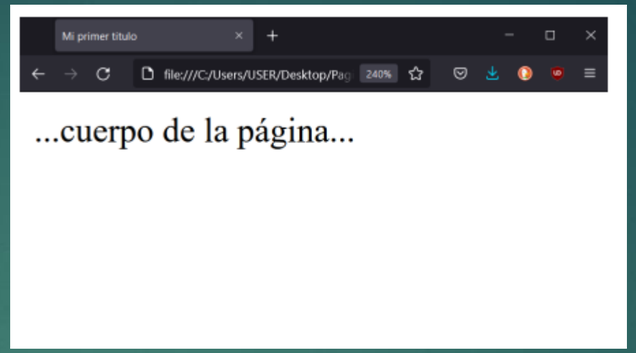
Después de la cabecera sigue el cuerpo de la página y se indica mediante la etiqueta <body>, el cuerpo o body contiene prácticamente los elementos visuales de nuestra página, aquí irán los textos de encabezado o títulos, subtítulos, párrafos, listas, tablas, formularios, multimedia, etc. Es decir el contenido de la página, todo lo que queremos que se vea en el navegador.

Lenguaje HTML

Finalmente la estructura básica HTML quedará de la siguiente manera:

```
<!doctype html>
<html>
    <head>
        <title> Mi primer titulo </title>
    </head>
    <body>
        ...cuerpo de la página...
    </body>
</html>
```

Para poder visualizar el resultado de lo anterior es necesario que abras un editor de texto y escribas el código de arriba, al finalizar, guarda el documento asociándole un nombre y al final agrega .html y guárdalo en alguna carpeta. Abre la carpeta donde guardaste el documento HTML y da doble click sobre el archivo. Automáticamente te abrirá el navegador predeterminado (Edge, Mozilla FireFox, u otro) y visualizarás algo como lo siguiente:



Reto. Escribe un programa en html para imprimir tu nombre completo con tu número de teléfono agregando como título de página "Información de contacto".

Más elementos en HTML5

Secciones

La etiqueta HTML <section> se utiliza para definir secciones en un documento. Cuando colocas tu contenido en una página web, esta puede contener muchos capítulos, encabezados, pies de página u otras secciones en la misma página web, para esto se usa la etiqueta HTML <section>.

HTML <section> es una nueva etiqueta introducida en HTML5.

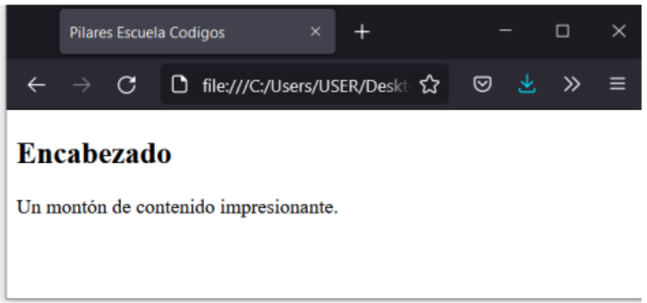
Ejemplo.

```
<section>
  <h1>Encabezado</h1>
  <p>Un montón de contenido impresionante.</p>
</section>
```

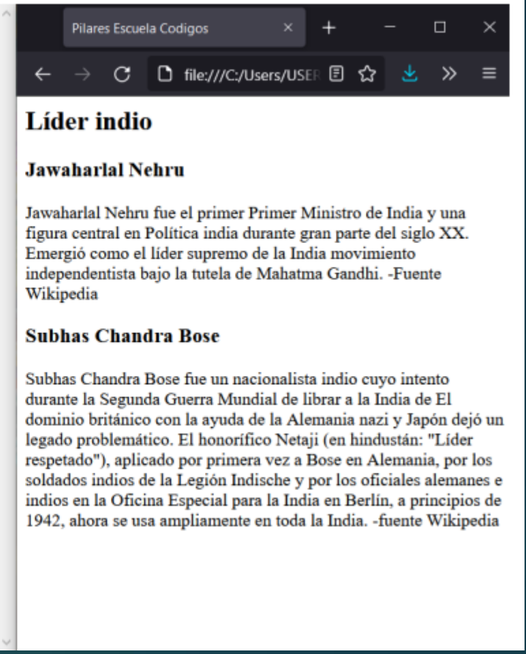
Si guardamos el archivo añadiendo extensión .html y abrimos en el navegador mostrará algo semejante a lo siguiente:

Lenguaje HTML

```
<!DOCTYPE>
<html>
<head>
<title>Pilares Escuela Codigos</title>
</head>
<body>
<section>
  <h1>Encabezado</h1>
  <p>Un montón de contenido impresionante.</p>
</section>
</body>
</html>
```




```
<!DOCTYPE>
<html>
<head>
<title>Pilares Escuela Codigos</title>
</head>
<body>
<h2> Líder indio</h2>
     <section>
     <h3> Jawaharlal Nehru </h3>
     <p> Jawaharlal Nehru fue el primer Primer Ministro de India y una
figura central en
     Política india durante gran parte del siglo XX. Emergió como el
líder supremo de la India
     movimiento independentista bajo la tutela de Mahatma Gandhi. -
Fuente Wikipedia</p>
     </section>

     <section>
     <h3>Subhas Chandra Bose</h3>
     <p>Subhas Chandra Bose fue un nacionalista indio cuyo intento
durante la Segunda Guerra Mundial de librar a la India de
     El dominio británico con la ayuda de la Alemania nazi y Japón
dejó un legado problemático.
     El honorífico Netaji (en hindustán: "Líder respetado"), aplicado
por primera vez a Bose en Alemania,
      por los soldados indios de la Legión Indische y por los
oficiales alemanes e indios
       en la Oficina Especial para la India en Berlín, a principios de
1942, ahora se usa ampliamente en toda la India. -fuente Wikipedia</p>
     </section>
</body>
</html>
```



Lenguaje HTML

Listas

Las **listas** HTML se utilizan para especificar listas de información. Todas las listas pueden contener uno o más elementos de lista. Hay tres tipos diferentes de listas HTML:

- Lista Ordenada o Lista Numerada (**ol**)
- Lista desordenada o lista con viñetas (**ul**)
- Lista de descripción o Lista de definición (**dl**)

Note que podemos crear una lista dentro de otra lista, que se denominará *Lista anidada*.

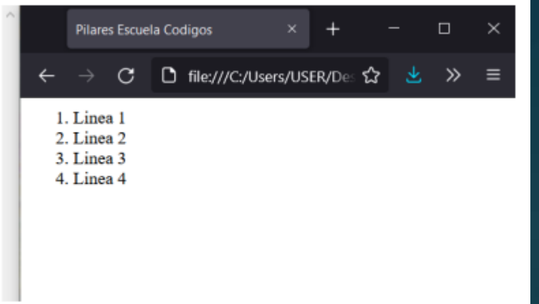
Lista Ordenada o Lista Numerada (lo). En las listas HTML ordenadas, todos los elementos de la lista están marcados con números de forma predeterminada. También se conoce como lista numerada. La lista ordenada comienza con la etiqueta <ol> y los elementos de la lista comienzan con la etiqueta <li>.

Ejemplo.

```
<ol>
  <li>Linea 1</li>
  <li>Linea 2</li>
  <li>Linea 3</li>
  <li>Linea 4</li>
</ol>
```

Si escribimos en nuestro editor de texto el código anterior y guardamos y ejecutamos, se visualizaría algo parecido a lo siguiente:

```
<!DOCTYPE>
<html>
<head>
<title>Pilares Escuela Codigos</title>
</head>
<body>
    <ol>
        <li>Linea 1</li>
        <li>Linea 2</li>
        <li>Linea 3</li>
        <li>Linea 4</li>
    </ol>
</body>
</html>
```



Lenguaje HTML

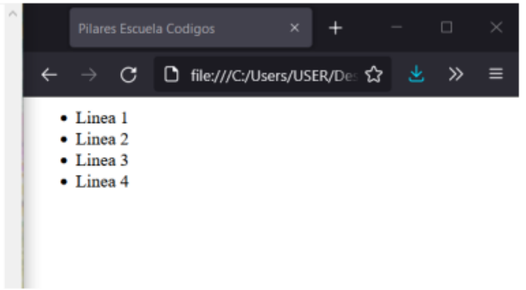
Lista desordenada en HTML o lista con viñetas (ul). En la lista desordenada de HTML, todos los elementos de la lista están marcados con viñetas. También se conoce como lista con viñetas. La lista desordenada comienza con la etiqueta <ul> y los elementos de la lista comienzan con la etiqueta <li>.

Ejemplo.

```
<ul>
  <li>Linea 1</li>
  <li>Linea 2</li>
  <li>Linea 3</li>
  <li>Linea 4</li>
</ul>
```

Si escribimos en nuestro editor de texto el código anterior y guardamos y ejecutamos, se visualizaría algo parecido a lo siguiente:

```
<!DOCTYPE>
<html>
<head>
<title>Pilares Escuela Codigos</title>
</head>
<body>
     <ul>
       <li>Linea 1</li>
       <li>Linea 2</li>
       <li>Linea 3</li>
       <li>Linea 4</li>
    </ul>
</body>
</html>
```



Lenguaje HTML

Lista de descripción HTML o Lista de definiciones (dl). La lista de descripción se se conoce también como lista de definiciones donde las entradas se enumeran como un diccionario o una enciclopedia.

La lista de definiciones es muy apropiada cuando se desea presentar un glosario, una lista de términos u otra lista de nombres y valores.

La lista de definiciones en HTML contiene las siguientes tres etiquetas:

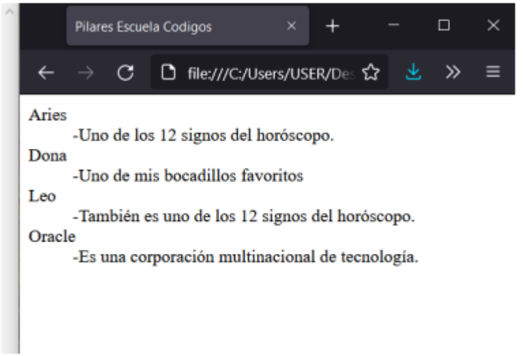
- La etiqueta **<dl>** : define el inicio de la lista.
- La etiqueta **<dt>** : define un término.
- La etiqueta **<dd>** : define la definición del término (descripción).

Ejemplo.

```
<dl>
    <dt>Aries</dt>
      <dd>-Uno de los 12 signos del horóscopo.</dd>
    <dt>Dona</dt>
      <dd>-Uno de mis bocadillos favoritos</dd>
   <dt>Leo</dt>
     <dd>-También es uno de los 12 signos del horóscopo.</dd>
    <dt>Oracle</dt>
     <dd>-Es una corporación multinacional de tecnología.
     </dd>
</dl>
```

Si escribimos en nuestro editor de texto el código anterior y guardamos y ejecutamos, se visualizaría algo parecido a lo siguiente:

```
<!DOCTYPE>
<html>
<head>
<title>Pilares Escuela Codigos</title>
</head>
<body>
<dl>
       <dt>Aries</dt>
          <dd>-Uno de los 12 signos del horóscopo.</dd>
       <dt>Dona</dt>
          <dd>-Uno de mis bocadillos favoritos</dd>
      <dt>Leo</dt>
         <dd>-También es uno de los 12 signos del horóscopo.</dd>
       <dt>Oracle</dt>
         <dd>-Es una corporación multinacional de tecnología.</dd>
</dl>
</body>
</html>
```



Lenguaje HTML

Tablas

La etiqueta de **table** en HTML se utiliza para mostrar datos en forma tabular (fila * columna). Puede haber muchas columnas en una fila.

Podemos crear una tabla para mostrar datos en forma tabular, usando el elemento **<table>**, con la ayuda de los elementos **<tr>**, **<td>** y **<th>**.

En cada tabla, la *fila de la tabla* está definida por la etiqueta **<tr>**, el *encabezado de la tabla* está definido por **<th>** y los *datos de la tabla* están definidos por las etiquetas **<td>**.

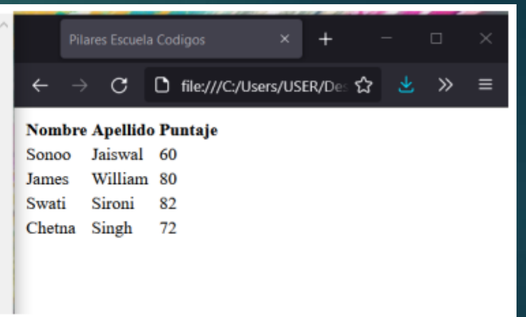
Las tablas en HTML se utilizan para gestionar el diseño de la página, por ejemplo, sección de encabezado, barra de navegación, contenido del cuerpo, sección de pie de página, etc. Pero se recomienda usar la etiqueta *div* sobre *table* para administrar el diseño de la página.

Ejemplo.

```
<table>
  <tr><th>Nombre</th><th>Apellido</th><th>Puntaje</th></tr>
  <tr><td>Sonoo</td><td>Jaiswal</td><td>60</td></tr>
  <tr><td>James</td><td>William</td><td>80</td></tr>
  <tr><td>Swati</td><td>Sironi</td><td>82</td></tr>
  <tr><td>Chetna</td><td>Singh</td><td>72</td></tr>
</table>
```

Si escribimos en nuestro editor de texto el código anterior y guardamos y ejecutamos, se visualizará algo parecido a lo siguiente:

```
<!DOCTYPE>
<html>
<head>
<title>Pilares Escuela Codigos</title>
</head>
<body>
    <table>
    <tr><th>Nombre</th><th>Apellido</th><th>Puntaje</th></tr>
    <tr><td>Sonoo</td><td>Jaiswal</td><td>60</td></tr>
    <tr><td>James</td><td>William</td><td>80</td></tr>
    <tr><td>Swati</td><td>Sironi</td><td>82</td></tr>
    <tr><td>Chetna</td><td>Singh</td><td>72</td></tr>
    </table>
</body>
</html>
```



Lenguaje HTML

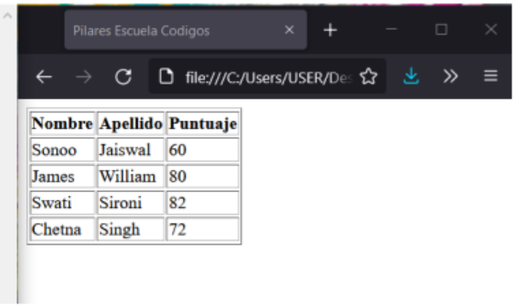
Atributo de borde en HTML. Puedes usar el atributo **border** de la etiqueta **table** en HTML para especificar el borde de una tabla.

Ejemplo.

```
<table border=1>
   <tr><th>Nombre</th><th>Apellido</th><th>Puntuaje</th></tr>
   <tr><td>Sonoo</td><td>Jaiswal</td><td>60</td></tr>
   <tr><td>James</td><td>William</td><td>80</td></tr>
   <tr><td>Swati</td><td>Sironi</td><td>82</td></tr>
   <tr><td>Chetna</td><td>Singh</td><td>72</td></tr>
</table>
```

Si escribimos en nuestro editor de texto el código anterior y guardamos y ejecutamos, se visualizará algo parecido a lo siguiente:

```
<!DOCTYPE>
<html>
<head>
<title>Pilares Escuela Codigos</title>
</head>
<body>
  <table border=1>
    <tr><th>Nombre</th><th>Apellido</th><th>Puntuaje</th></tr>
    <tr><td>Sonoo</td><td>Jaiswal</td><td>60</td></tr>
    <tr><td>James</td><td>William</td><td>80</td></tr>
    <tr><td>Swati</td><td>Sironi</td><td>82</td></tr>
    <tr><td>Chetna</td><td>Singh</td><td>72</td></tr>
  </table>
</body>
</html>
```



Nombre	Apellido	Puntuaje
Sonoo	Jaiswal	60
James	William	80
Swati	Sironi	82
Chetna	Singh	72

Lenguaje HTML

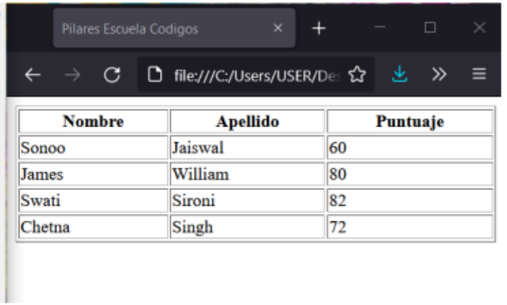
Ancho de la tabla en HTML. Podemos ajustar el ancho de nuestra tabla según nuestras necesidades. El atributo **style** y **width** son usados en la etiqueta **table** para especificar el ancho de la tabla, este puede especificado en píxeles o porcentaje.

Ejemplo.

```
<table style=width:100% border=1>
  <tr><th>Nombre</th><th>Apellido</th><th>Puntuaje</th></tr>
  <tr><td>Sonoo</td><td>Jaiswal</td><td>60</td></tr>
  <tr><td>James</td><td>William</td><td>80</td></tr>
  <tr><td>Swati</td><td>Sironi</td><td>82</td></tr>
  <tr><td>Chetna</td><td>Singh</td><td>72</td></tr>
</table>
```

Si escribimos en nuestro editor de texto el código anterior y guardamos y ejecutamos, se visualizará algo parecido a lo siguiente:

```
<!DOCTYPE>
<html>
<head>
<title>Pilares Escuela Codigos</title>
</head>
<body>
  <table style=width:100% border=1>
    <tr><th>Nombre</th><th>Apellido</th><th>Puntuaje</th></tr>
    <tr><td>Sonoo</td><td>Jaiswal</td><td>60</td></tr>
    <tr><td>James</td><td>William</td><td>80</td></tr>
    <tr><td>Swati</td><td>Sironi</td><td>82</td></tr>
    <tr><td>Chetna</td><td>Singh</td><td>72</td></tr>
  </table>
</body>
</html>
```



Lenguaje HTML

Tabla en HTML con colspan. Si deseas que una celda abarque más de una columna, puedes usar el atributo **colspan** sobre la etiqueta <th>. El atributo dividirá una celda/fila en varias columnas, y el número de columnas dependerá del valor del atributo *colspan*.

Ejemplo.

```
<table style=width:100% border=1>
    <tr>
    <th>Nombre</th>
    <th colspan=2>No. Movil</th>
    </tr>
    <tr>
    <td>Omar Montoya</td>
    <td>55123456</td>
    <td>55654321</td>
    </tr>
</table>
```

Si escribimos en nuestro editor de texto el código anterior y guardamos y ejecutamos, se visualizará algo parecido a lo siguiente:

```
<!DOCTYPE>
<html>
<head>
<title>Pilares Escuela Codigos</title>
</head>
<body>
    <table style=width:100% border=1>
      <tr>
      <th>Nombre</th>
      <th colspan=2>No. Movil</th>
      </tr>
      <tr>
      <td>Omar Montoya</td>
      <td>55123456</td>
      <td>55654321</td>
      </tr>
    </table>
</body>
</html>
```

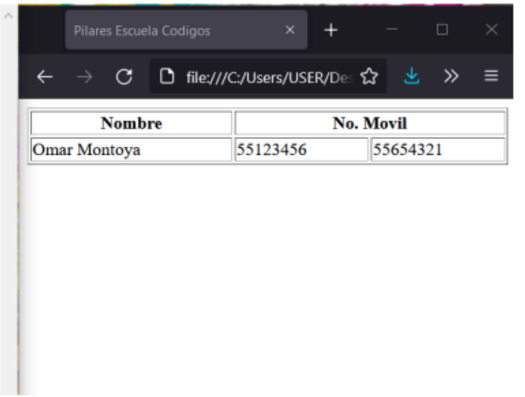
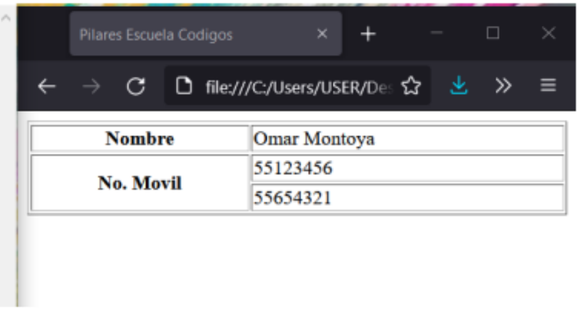


Tabla en HTML con rowspan. Si deseas que una celda abarque más de una fila, puedes usar el atributo rowspan sobre la etiqueta <th>. Este dividirá una celda en varias filas. El número de filas divididas dependerá de los valores de intervalo de filas.

Ejemplo.

```
<table style=width:100% border=1>
  <tr><th>Nombre</th><td>Omar Montoya</td></tr>
  <tr><th rowspan=2>No. Movil</th><td>55123456</td></tr>
  <tr><td>55654321</td></tr>
</table>
```

```
<!DOCTYPE>
<html>
<head>
<title>Pilares Escuela Codigos</title>
</head>
<body>
    <table style=width:100% border=1>
    <tr><th>Nombre</th><td>Omar Montoya</td></tr>
    <tr><th rowspan=2>No. Movil</th><td>55123456</td></tr>
    <tr><td>55654321</td></tr>
    </table>
</body>
</html>
```



Lenguaje HTML

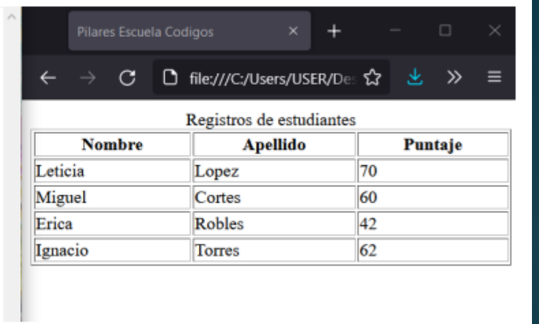
Tabla en HTML con caption. El título de una tabla en HTML se muestra encima de la tabla. Para colocarlo debe usarse la etiqueta *caption* solo después de la etiqueta *table*.

Ejemplo.

```
<table border=1 style=width:100%>
    <caption>Registros de estudiantes</caption>
      <tr><th>Nombre</th><th>Apellido</th><th>Puntaje</th></tr>
      <tr><td>Leticia</td><td>Lopez</td><td>70</td></tr>
      <tr><td>Miguel</td><td>Cortes</td><td>60</td></tr>
      <tr><td>Erica</td><td>Robles</td><td>42</td></tr>
      <tr><td>Ignacio</td><td>Torres</td><td>62</td></tr>
</table>
```

Si escribimos en nuestro editor de texto el código anterior y guardamos y ejecutamos, se visualizaría algo parecido a lo siguiente:

```
<!DOCTYPE>
<html>
<head>
<title>Pilares Escuela Codigos</title>
</head>
<body>
    <table border=1 style=width:100%>
      <caption>Registros de estudiantes</caption>
        <tr><th>Nombre</th><th>Apellido</th><th>Puntaje</th></tr>
        <tr><td>Leticia</td><td>Lopez</td><td>70</td></tr>
        <tr><td>Miguel</td><td>Cortes</td><td>60</td></tr>
        <tr><td>Erica</td><td>Robles</td><td>42</td></tr>
        <tr><td>Ignacio</td><td>Torres</td><td>62</td></tr>
    </table>
</body>
</html>
```



Lenguaje HTML

Hipervínculos

La etiqueta anchor en HTML. La etiqueta **anchor <a>** en HTML define un hipervínculo que vincula una página a otra página. Puede crear un hipervínculo a otra página web, así como a archivos, ubicación o cualquier URL. El atributo **href** es el atributo más importante de la etiqueta *<a>* en HTML y la que vincula a una determinada página o URL.

Ejemplo.

```
<a href=segundapagina.html>Click para ir a mi otra pagina web</a>
```

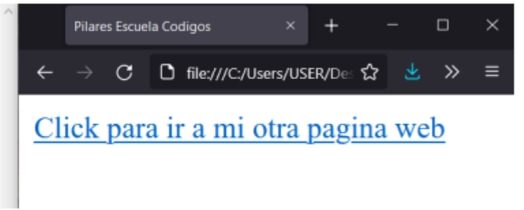
Si escribimos en nuestro editor de texto el código anterior y guardamos y ejecutamos, se visualizaría algo parecido a lo siguiente:

Si queremos abrir ese enlace en otra página, podemos usar el atributo **target** de destino de la etiqueta **<a>**. Con la ayuda de este atributo la segunda página se abrirá en otra ventana del navegador.

Ejemplo.

```
<p>Da click en <a href=https://pilares.cdmx.gob.mx/inicio target=_blank>este link
</a>para ir a la página de PILARES</p>
```

```
<!DOCTYPE>
<html>
<head>
<title>Pilares Escuela Codigos</title>
</head>
<body>
    <a href="segundapagina.html">Click para ir a mi otra pagina
    web</a>
</body>
</html>
```



Lenguaje HTML

Imágenes

La etiqueta **img** en HTML se utiliza para mostrar una imagen en una página web. La etiqueta **img** es una etiqueta vacía que solo contiene *atributos*, las etiquetas de cierre no se utilizan en el elemento **img**.

Ejemplo.

```
<h2>Ejemplo de imagen en HTML</h2>
  <img src=ADIP.jpg />
```

Para visualizar la siguiente imagen tomamos un archivo imagen ADIP.jpg y lo guardamos en la misma carpeta donde está guardado nuestro archivo con extensión .html. Luego, escribimos en nuestro editor de texto el código anterior y guardamos y ejecutamos, el resultado es el siguiente:

```
<!DOCTYPE>
<html>
<head>
<title>Pilares Escuela Codigos</title>
</head>
<body>
    <h2>Ejemplo de imagen en HTML</h2>
    <img src=ADIP.jpg />
</body>
</html>
```

Atributos de la etiqueta img en HTML. El *src* y *alt* son atributos importantes de la etiqueta **img**. Todos los atributos de la etiqueta de imagen HTML se dan a continuación.

- Atributo **scr**. Es un atributo necesario que describe la fuente o ruta de la imagen. Le indica al navegador dónde buscar la imagen en el servidor. La ubicación de la imagen puede estar en el mismo directorio o en otro servidor.

  Ejemplo.

  ```
  <img src=ADIP.jpg />
  ```

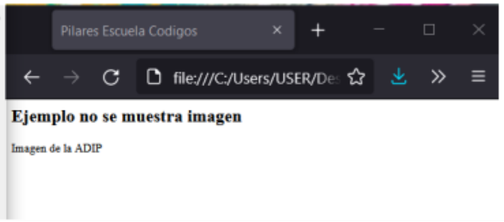
- Atributo **alt**. El atributo **alt** define un texto alternativo para la imagen, si no se puede mostrar. El valor del atributo **alt** describe la imagen en palabras.

  Ejemplo.

  ```
  <img src=ADIP.jpg alt=Imagen de la ADIP />
  ```

Si escribimos en nuestro editor de texto el código anterior y guardamos y ejecutamos, se visualizaría algo parecido a lo siguiente:

```
<!DOCTYPE>
<html>
<head>
<title>Pilares Escuela Codigos</title>
</head>
<body>
    <h2>Ejemplo no se muestra imagen</h2>
    <img src=ADI.jpg alt="Imagen de la ADIP" />
</body>
</html>
```



Videos

HTML5 también admite la etiqueta **<video>**. La etiqueta de video en HTML se usa para transmitir archivos de video, como clips de películas, clips de canciones, en la página web.

Actualmente, hay tres formatos de video compatibles con la *etiqueta de video* HTML:

- mp4
- webM
- ogg

Ejemplo.

```
<video controls>
    <source src=ADIP.mp4 type=video/mp4>
    tu navegador no admite la etiqueta de video html.
</video>
```

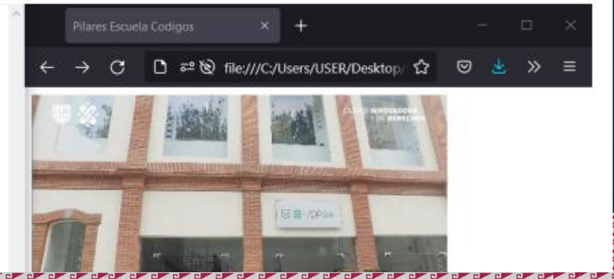
Nota que incluimos una ruta a los medios que queremos mostrar usando el atributo **src**; podemos incluir otros atributos para especificar información como el ancho (**width**) y la altura (**heigh**) del video; si queremos que se reproduzca automáticamente (**autoplay**) o se reproduzca en bucle (**loop**), también está el atributo **controls** si queremos mostrar los controles de video predeterminados del navegador, **muted** se utiliza para silenciar la salida de vídeo,etc.

Ejemplo.

```
<video width=426 height=240 controls autoplay loop muted>
    <source src=ADIP.mp4 type=video/mp4>
    Tu navegador no admite la etiqueta de video html.
</video>
```

Para visualizar el video necesitamos colocar el archivo de video (en este caso ADIP.mp4) en la misma carpeta donde guardamos nuestro archivo con extensión .html, de esta manera, si escribimos en nuestro editor de texto el código anterior y guardamos y ejecutamos, se visualizaría algo parecido a lo siguiente:

```
<!DOCTYPE>
<html>
<head>
<title>Pilares Escuela Codigos</title>
</head>
<body>
    <video width=426 height=240 controls autoplay loop muted>
      <source src=ADIP.mp4 type=video/mp4>
      Tu navegador no admite la etiqueta de video html.
    </video>
</body>
</html>
```



Formularios

Un formulario en HTML es una *sección de un documento* que contiene controles como campos de texto, campos de contraseña, casillas de verificación, botones de radio, botón de envío, menús, etc.

Un formulario en HTML facilita al usuario ingresar datos que se enviarán al servidor para su procesamiento, como nombre, dirección de correo electrónico, contraseña, número de teléfono, etc.

Por ejemplo: si un usuario desea comprar algunos artículos en Internet, debe completar un formulario, como la dirección de envío y los detalles de la tarjeta de crédito/débito, para que el artículo pueda enviarse a la dirección indicada.

La sintaxis de un formulario en HTML es como sigue:

```
<form action="server url" method="get|post">
    // controles de entrada, campo de texto, área de texto, botes, etc.
</form>
```

A continuación tenemos una lista de etiquetas útiles para la realización de formularios:

Etiqueta	Descripción
<form>	Define un formulario HTML para ingresar datos para ser usados en otro lado.
<input>	Define un control de entrada.
<textarea>	Define un control de entrada multilínea.
<label>	Define una etiqueta para un elemento de entrada.
<fieldset>	Agrupa el elemento relacionado en un formulario.
<legend>	Define un título para un elemento <fieldset>.
<select>	Define una lista desplegable.
<optgroup>	Define un grupo de opciones relacionadas en una lista desplegable.
<option>	Define una opción en una lista desplegable.
<button>	Define un botón en el que se puede hacer clic.
<datalist>	Especifica una lista de opciones predefinidas para el control de entrada.
<keygen>	Define un campo generador de pares de claves para formularios.
<output>	Define el resultado de un cálculo.

Lenguaje HTML

El elemento <form> no crea un formulario en sí mismo, pero es un contenedor para colocar todos los *elementos* de un formulario necesarios, como <input>, <label>, etc.

La etiqueta **<form>** dispone de varios atributos para utilizar:

Atributo	Valor	Descripción
action	*URL*	Dirección URL del back-end donde se enviará la información del formulario.
method	**get | post**	Método HTTP de envío. GET a través de URL, POST para envío extenso.
name	*nombre*	Nombre del formulario. Útil para procesar posteriormente.
target	*destino*	Nombre del lugar donde se abrirá el formulario. _blank para nueva pestaña.
enctype	*tipo*	Codificación para el envío del formulario. Importante para envío de archivos.
accept-charset	*codificación*	Fuerza a utilizar una codificación en los parámetros de texto del formulario.
autocomplete	**on | off**	Activa o desactiva el autocompletado para todos los campos del formulario.
novalidate		Con este atributo presente, el formulario obvia la validación HTML5.
type	tipo de campo	Indica el tipo de campo del que se trata.

El elemento **<input>** en HTML es un elemento fundamental de la etiqueta **form**. Se utiliza para crear campos de formulario, para recibir información del usuario. El siguiente es un ejemplo para mostrar la entrada de texto simple.

```
<form>
    Introduce tu nombre  <br>
    <input type="text" name="username">
</form>
```

Si escribimos en nuestro editor de texto el código anterior y guardamos y ejecutamos, se visualizaría algo parecido a lo siguiente:

```
<!DOCTYPE>
<html>
<head>
<title>PILARES Escuela Codigos</title>
</head>
<body>
    <form>
     Introduce tu nombre  <br>
     <input type="text" name="username">
    </form>
</body>
</html>
```

El atributo **type="text"** de la etiqueta de entrada crea un control de campo de texto, también conocido como *control de campo de texto de una sola línea*. El atributo **name** es opcional, pero se requiere para el componente del lado del servidor.

La etiqueta <textarea> en form. La etiqueta **<textarea>** en HTML se usa para insertar texto de varias líneas en un formulario. El tamaño de <textarea> se puede especificar usando el atributo **rows** o **cols.**

Ejemplo.

```
<form>
     Ingresa tu dirección:<br>
    <textarea rows="2" cols="20"></textarea>
</form>
```

Si escribimos en nuestro editor de texto el código anterior y guardamos y ejecutamos, se visualizaría algo parecido a lo siguiente:

```
<!DOCTYPE>
<html>
<head>
<title>PILARES Escuela de Codigos</title>
</head>
<body>
   <form>
       Ingresa tu dirección:<br>
      <textarea rows="2" cols="20"></textarea>
   </form>
</body>
</html>
```

La etiqueta <label> en form. Se considera usar **label** en **form** ya que hace que el código sea amigable para el analizador/navegador/usuario.

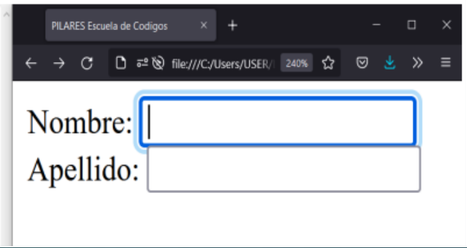
Si haces clic en la etiqueta **label**, se centrará en el *control de texto* (en la caja donde se introduce el texto), esto es tiene mayor funcionalidad en pantallas táctiles. Para hacerlo, debes tener el atributo **for** en la etiqueta **label** cuyo valor debe ser el mismo que el valor del atributo **id** de la etiqueta **input**.

Ejemplo.

```
<form>
      <label for="firstname">Nombre: </label>
            <input type="text" id="firstname" name="firstname"/><br/>
     <label for="lastname">Apellido: </label>
            <input type="text" id="lastname" name="lastname"/><br/>
</form>
```

Si escribimos en nuestro editor de texto el código anterior y guardamos y ejecutamos, se visualizaría algo parecido a lo siguiente:

```
<!DOCTYPE>
<html>
<head>
<title>PILARES Escuela de Codigos</title>
</head>
<body>
   <form>
        <label for="firstname">Nombre: </label>
                <input type="text" id="firstname" name="firstname"/><br/>
       <label for="lastname">Apellido: </label>
                <input type="text" id="lastname" name="lastname"/><br/>
   </form>
</body>
</html>
```



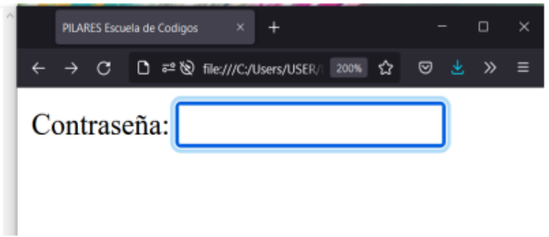
Control de campo de password en form. La contraseña no es visible para el usuario en el control de campo de **password** que se pone como valor en el atributo **type** de la etiqueta **input**.

Ejemplo.

```
<form>
    <label for="password">Contraseña: </label>
      <input type="password" id="password" name="password"/> <br/>
</form>
```

Si escribimos en nuestro editor de texto el código anterior y guardamos y ejecutamos, se visualizaría algo parecido a lo siguiente:

```
<!DOCTYPE>
<html>
<head>
<title>PILARES Escuela de Codigos</title>
</head>
<body>
    <form>
       <label for="password">Contraseña: </label>
          <input type="password" id="password" name="password"/> <br/>
    </form>
</body>
</html>
```



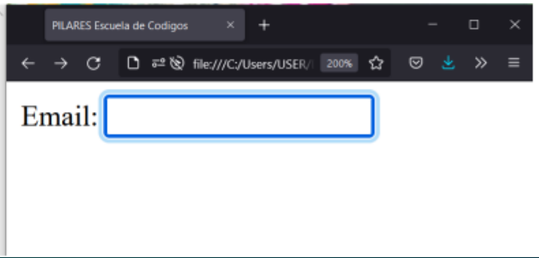
Control de campo email en form. El campo **email** es nuevo en HTML 5. Valida el texto para introducir la dirección de correo electrónico correctamente. Debes utilizar @ (arroba) y . (punto) en este campo.

Ejemplo.

```
<form>
    <label for="email">Email: </label>
      <input type="email" id="email" name="email"/> <br/>
</form>
```

Si escribimos en nuestro editor de texto el código anterior y guardamos y ejecutamos, se visualizaría algo parecido a lo siguiente:

```
<!DOCTYPE>
<html>
<head>
<title>PILARES Escuela de Codigos</title>
</head>
<body>
    <form>
       <label for="email">Email: </label>
          <input type="email" id="email" name="email"/> <br/>
    </form>
</body>
</html>
```



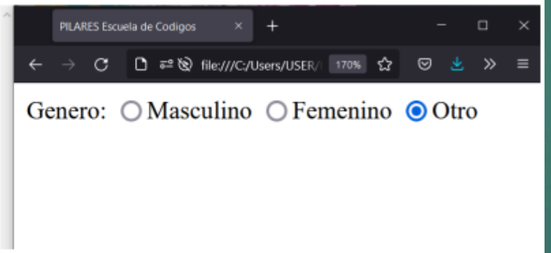
Lenguaje HTML

Control de botón de radio en form. El **botón de radio** se utiliza para seleccionar una opción entre múltiples opciones. Se utiliza para la selección de género, preguntas de prueba, etc. Si usas un nombre para multiples *botones de radio*, solo se podrá seleccionar un botón de radio a la vez.

Ejemplo.

```
<form>
    <label for="gender">Genero: </label>
      <input type="radio" id="gender" name="gender" value="male"/>Masculino
      <input type="radio" id="gender" name="gender" value="female"/>Femenino
      <input type="radio" id="gender" name="gender" value="other"/>Otro <br/>
</form>
```

```
<!DOCTYPE>
<html>
<head>
<title>PILARES Escuela de Codigos</title>
</head>
<body>
    <form>
      <label for="gender">Genero: </label>
        <input type="radio" id="gender" name="gender" value="male"/>Masculino
        <input type="radio" id="gender" name="gender" value="female"/>Femenino
        <input type="radio" id="gender" name="gender" value="other"/>Otro <br/>
    </form>
</body>
</html>
```



Control de checkbox en form. El control **checkbox** se utiliza para marcar múltiples opciones de casillas de verificación dadas.

Ejemplo.

```
<form>
    Pasatiempo:<br>
     <input type="checkbox" id="sing" name="sing" value="sing"/>
     <label for="sing">Cantar</label><br>
     <input type="checkbox" id="football" name="football" value="football"/>
     <label for="football">Futbol</label><br>
     <input type="checkbox" id="reading" name="reading" value="reading"/>
     <label for="reading">Leer</label>
</form>
```

Si escribimos en nuestro editor de texto el código anterior y guardamos y ejecutamos, se visualizaría algo parecido a lo siguiente:

```
<!DOCTYPE>
<html>
<head>
<title>PILARES Escuela de Codigos</title>
</head>
<body>
    <form>
      Pasatiempo:<br>
       <input type="checkbox" id="sing" name="sing" value="sing"/>
       <label for="sing">Cantar</label><br>
       <input type="checkbox" id="football" name="football" value="football"/>
       <label for="football">Futbol</label><br>
       <input type="checkbox" id="reading" name="reading" value="reading"/>
       <label for="reading">Leer</label>
    </form>
</body>
</html>
```

Control de botón submit en form. En HTML **<input type="submit">** se utiliza para agregar un botón de ***Enviar*** en la página web. Cuando el usuario hace clic en el botón ***Enviar***, el formulario se envía al servidor. La sintaxis es la siguiente:

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

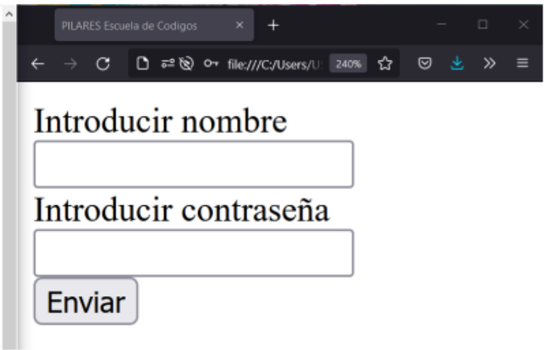
Aquí **type=submit** está especificando que es un botón de ***Enviar***. El atributo **value** puede ser cualquier cosa que escribamos en el botón de la página web. El atributo **name** puede ser omitido aquí.

Ejemplo.

```
<form>
   <label for="name">Introducir nombre</label><br>
     <input type="text" id="name" name="name"><br>
   <label for="pass">Introducir contraseña</label><br>
     <input type="Password" id="pass" name="pass"><br>
   <input type="submit" value="Enviar">
</form>
```

Si escribimos en nuestro editor de texto el código anterior y guardamos y ejecutamos, se visualizaría algo parecido a lo siguiente:

```
<!DOCTYPE>
<html>
<head>
<title>PILARES Escuela de Codigos</title>
</head>
<body>
   <form>
    <label for="name">Introducir nombre</label><br>
    <input type="text" id="name" name="name"><br>
    <label for="pass">Introducir contraseña</label><br>
    <input type="Password" id="pass" name="pass"><br>
    <input type="submit" value="Enviar">
   </form>
</body>
</html>
```

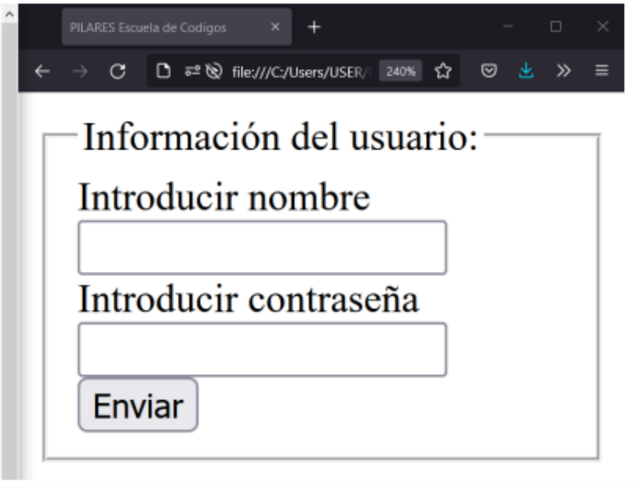


Lenguaje HTML

El elemento <fieldset> en HTML. El elemento <fieldset> en HTML se usa para agrupar la información relacionada de un formulario. Este elemento se usa con el elemento **<legend>** que proporciona un título para los elementos agrupados.

```
<form>
  <fieldset>
    <legend>Información del usuario:</legend>
     <label for="name">Introducir nombre</label><br>
     <input type="text" id="name" name="name"><br>
     <label for="pass">Introducir contraseña</label><br>
     <input type="Password" id="pass" name="pass"><br>
     <input type="submit" value="Enviar">
  </fieldset>
</form>
```

```
<!DOCTYPE>
<html>
<head>
<title>PILARES Escuela de Codigos</title>
</head>
<body>
   <form>
    <fieldset>
      <legend>Información del usuario:</legend>
      <label for="name">Introducir nombre</label><br>
      <input type="text" id="name" name="name"><br>
      <label for="pass">Introducir contraseña</label><br>
      <input type="Password" id="pass" name="pass"><br>
      <input type="submit" value="Enviar">
    </fieldset>
   </form>
</body>
</html>
```



Lenguaje HTML

El siguiente es el ejemplo de un formulario simple de registro.

```
<!DOCTYPE>
<html>
<head>
<title>PILARES Escuela de Codigos</title>
</head>
<body>
   <h2>Formato de registro</h2>
    <form>
     <fieldset>
        <legend>Información personal del usuario</legend>
        <label>Introduce tu nombre completo</label><br>
        <input type="text" name="name"><br>
         <label>Introduce tu email</label><br>
         <input type="email" name="email"><br>
         <label>Introduce tu contraseña</label><br>
         <input type="password" name="pass"><br>
         <label>Confirma tu contraseña</label><br>
         <input type="password" name="pass"><br>
         <br><label>Género</label><br>
         <input type="radio" id="gender" name="gender"
value="male"/>Masculino <br>
         <input type="radio" id="gender" name="gender"
value="female"/>Femenino<br/>
         <input type="radio" id="gender" name="gender"
value="others"/>Otro <br/>
          <br>Introduce tu dirección:<br>
         <textarea></textarea><br>
         <input type="submit" value="Registrarse">
     </fieldset>
  </form>
</body>
</html>
```

Lenguaje HTML

Contenedores

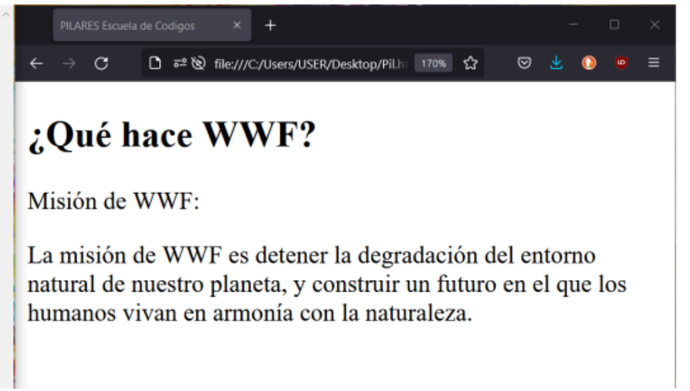
El elemento <header> como contenedor. El elemento **<header>** representa un contenedor de contenido introductorio o un conjunto de enlaces de navegación.

Un elemento **<header>** normalmente contiene:

- uno o más elementos de encabezado (**<h1> — <h6>**)
- logotipo o icono,
- información de autoría.

Nota: Puedes tener varios elementos **<header>** en un documento HTML. Sin embargo, **<header>** no se puede colocar dentro de un **<footer>**, **<address>** u otro elemento **<header>**.

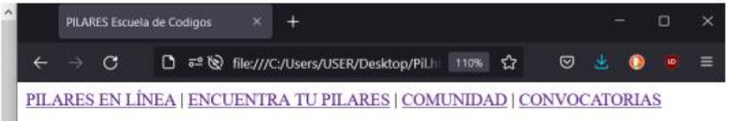
```
<!DOCTYPE>
<html>
<head>
<title>PILARES Escuela de Codigos</title>
</head>
<body>
 <article>
  <header>
    <h1>¿Qué hace WWF?</h1>
    <p>Misión de WWF:</p>
  </header>
  <p>La misión de WWF es detener la degradación del
entorno natural de nuestro planeta,
   y construir un futuro en el que los humanos vivan en
armonía con la naturaleza.</p>
</article>
</body>
</html>
```



Lenguaje HTML

El elemento <nav> como contenedor. El elemento **<nav>** define un conjunto de enlaces de navegación.

```
<!DOCTYPE>
<html>
<head>
<title>PILARES Escuela de Codigos</title>
</head>
<body>
  <nav>
     <a href="https://pilares.cdmx.gob.mx/pilares-en-
linea">PILARES EN LÍNEA</a> |
     <a
href="https://pilares.cdmx.gob.mx/inicio#mapa">ENCUENTRA
TU PILARES</a> |
     <a
href="https://pilares.cdmx.gob.mx/comunidad">COMUNIDAD</a
> |
     <a
href="https://pilares.cdmx.gob.mx/convocatorias">CONVOCAT
ORIAS</a>
  </nav>
</body>
</html>
```



Ten en cuenta que NO todos los enlaces de un documento deben estar dentro de un elemento <nav>. El elemento <nav> está diseñado solo para el bloque principal de enlaces de navegación.

El elemento <section> como contenedor. El elemento **<section>** define una sección en un documento.

De acuerdo con la documentación HTML de W3C: "Una sección es una agrupación temática de contenido, generalmente con un encabezado".

Ejemplos de dónde se puede usar un elemento **<section>**:

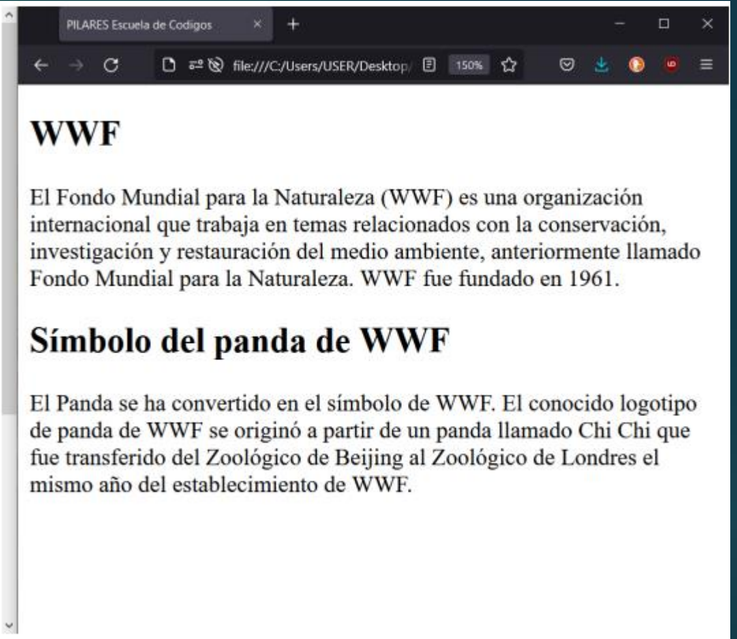
- Capítulos
- Introducción
- Noticias
- Información del contacto

Normalmente, una página web se puede dividir en secciones para la introducción, el contenido y la información de contacto.

Ejemplo. Dos secciones en un documento:

```
<section>
   <h1>WWF</h1>
    <p>El Fondo Mundial para la Naturaleza (WWF) es una organización internacional que trabaja en temas relacionados con la conservación, investigación y restauración del medio ambiente, anteriormente llamado Fondo Mundial para la Naturaleza. WWF fue fundado en 1961.</p>
</section>

<section>
   <h1>Símbolo del panda de WWF</h1>
    <p>El Panda se ha convertido en el símbolo de WWF. El conocido logotipo de panda de WWF se originó a partir de un panda llamado Chi Chi que fue transferido del Zoológico de Beijing al Zoológico de Londres el mismo año del establecimiento de WWF.</p>
</section>
```

```
<!DOCTYPE>
<html>
<head>
<title>PILARES Escuela de Codigos</title>
</head>
<body>
   <section>
    <h1>WWF</h1>
      <p>El Fondo Mundial para la Naturaleza (WWF) es una
organización internacional que trabaja en temas
relacionados con la conservación, investigación y
restauración del medio ambiente, anteriormente llamado
Fondo Mundial para la Naturaleza. WWF fue fundado en
1961.</p>
   </section>

   <section>
    <h1>Símbolo del panda de WWF</h1>
      <p>El Panda se ha convertido en el símbolo de WWF.
El conocido logotipo de panda de WWF se originó a partir
de un panda llamado Chi Chi que fue transferido del
Zoológico de Beijing al Zoológico de Londres el mismo año
del establecimiento de WWF.</p>
   </section>
</body>
</html>
```



El elemento <article> como contenedor. El elemento **<article>** especifica un contenedor independiente. Un artículo debe tener sentido por sí mismo y debe ser posible distribuirlo independientemente del resto del sitio web.

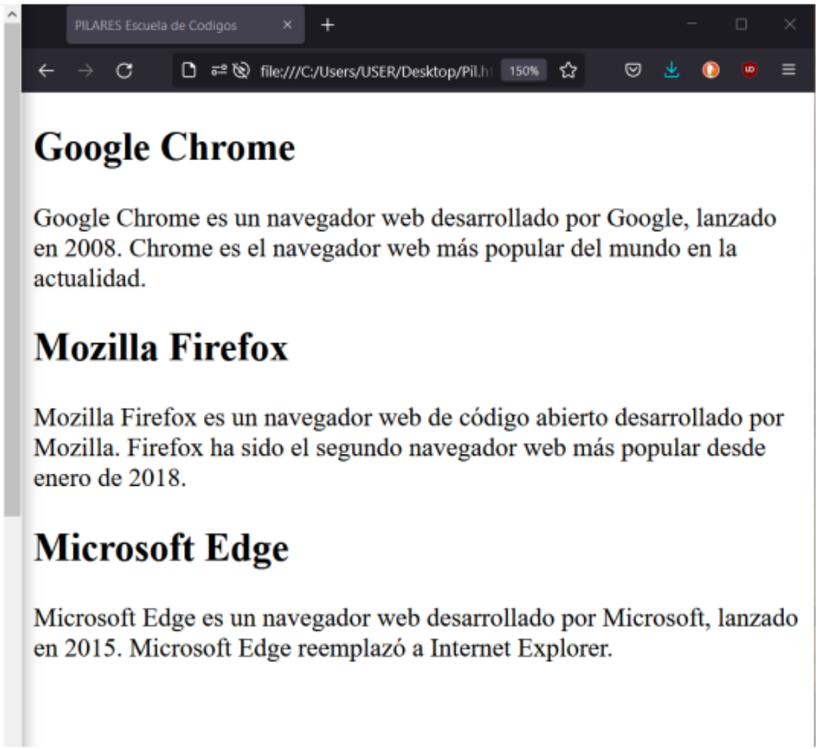
Ejemplos de dónde se puede usar el elemento **<article>**:

- Mensajes del foro
- Publicaciones de blog
- Comentarios del usuario
- Tarjetas de productos
- Artículos del periódico

```
<article>
    <h2>Google Chrome</h2>
     <p>Google Chrome es un navegador web desarrollado por Google, lanzado en 2008. Chrome es el navegador web más popular del mundo en la actualidad.</p>
</article>

<article>
    <h2>Mozilla Firefox</h2>
     <p>Mozilla Firefox es un navegador web de código abierto desarrollado por Mozilla. Firefox ha sido el segundo navegador web más popular desde enero de 2018.</p>
</article>

<article>
    <h2>Microsoft Edge</h2>
     <p>Microsoft Edge es un navegador web desarrollado por Microsoft, lanzado en 2015. Microsoft Edge reemplazó a Internet Explorer.</p>
</article>
```

Si escribimos en nuestro editor de texto el código anterior y guardamos y ejecutamos, se visualizaría algo parecido a lo siguiente:

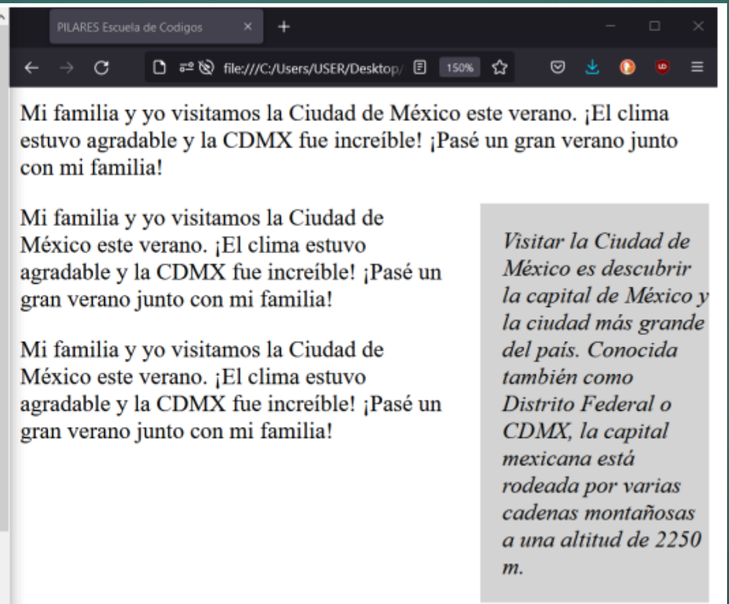
```
<!DOCTYPE>
<html>
<head>
<title>PILARES Escuela de Codigos</title>
</head>
<body>
    <article>
     <h2>Google Chrome</h2>
       <p>Google Chrome es un navegador web desarrollado
por Google, lanzado en 2008. Chrome es el navegador web
más popular del mundo en la actualidad.</p>
    </article>

    <article>
     <h2>Mozilla Firefox</h2>
       <p>Mozilla Firefox es un navegador web de código
abierto desarrollado por Mozilla. Firefox ha sido el
segundo navegador web más popular desde enero de
2018.</p>
    </article>

    <article>
     <h2>Microsoft Edge</h2>
       <p>Microsoft Edge es un navegador web desarrollado
por Microsoft, lanzado en 2015. Microsoft Edge reemplazó
a Internet Explorer.</p>
    </article>
</body>
</html>
```



El elemento <article> como contenedor. El elemento **<aside>** define algún contenido además del contenido en el que se coloca (como una barra lateral).

El contenido **<aside>** debe estar indirectamente relacionado con el contenido circundante.

```
<html>
<head>
<title>PILARES Escuela de Codigos</title>
</head>
<body>

<p>Mi familia y yo visitamos la Ciudad de México este verano.
¡El clima estuvo agradable y la CDMX fue increíble! ¡Pasé un
gran verano junto con mi familia!</p>

<aside>
<p>Visitar la Ciudad de México es descubrir la capital de
México y la ciudad más grande del país. Conocida también como
Distrito Federal o CDMX, la capital mexicana está rodeada por
varias cadenas montañosas a una altitud de 2250 m.</p>
</aside>

<p>Mi familia y yo visitamos la Ciudad de México este verano.
¡El clima estuvo agradable y la CDMX fue increíble! ¡Pasé un
gran verano junto con mi familia!</p>
<p>Mi familia y yo visitamos la Ciudad de México este verano.
¡El clima estuvo agradable y la CDMX fue increíble! ¡Pasé un
gran verano junto con mi familia!</p>

</body>
</html>
```



El elemento <footer> como contenedor. El elemento **<footer>** define un pie de página para un documento o sección.

Un elemento **<footer>** normalmente contiene:

- Información de autoría
- Información derechos de autor
- Información del contacto
- Mapa del sitio
- Volver a los enlaces superiores
- Documentos relacionados

Puedees tener varios elementos <footer> en un documento.

```
<footer>
  <p>Author: Hege Refsnes</p>
  <p><a href="mailto:hege@example.com">hege@example.com</a></p>
</footer>
```

Etiqueta Meta

La etiqueta **<meta>** en HTML se utiliza para representar los metadatos sobre el documento HTML. Especifica la descripción de la página, las palabras clave, los derechos de autor, el idioma, el autor de los documentos, etc.

Los metadatos no se muestran en la página web, pero los utilizan los motores de búsqueda, los navegadores y otros servicios web que escanean el sitio o la página web para conocerla.

Con la ayuda de la etiqueta **meta**, puedes experimentar y obtener una vista previa de cómo se mostrará tu página web en el navegador.

La etiqueta **<meta>** se coloca dentro de la etiqueta **<head>** y se puede usar más de una vez en un documento.

A continuación una lista de **atributos** para la etiqueta **meta**:

Atributo	Valor	Descripción
charset	character_set	Especifica la codificación de caracteres de una página HTML.
content	text	Contiene el valor del atributo **name** y **http-equiv** en un documento HTML, dependiendo del contexto.
http-equiv	• Content-type • default-style • refresh	Especifica la información proporcionada por el servidor web sobre cómo se debe servir la página web.
name	• application-name • author • description • generator • keywords	Especifica el nombre de los metadatos a nivel de documento.

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta name="keywords" content="Curso, Codigos, PILARES, Escuela">
    <meta name="description" content="Escuela de Codigos en PILARES">
    <meta name="author" content="ADIP">
    <meta http-equiv="refresh" content="5; url=https://pilares.cdmx.gob.mx/inicio">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
  </head>
  <body>
    <h2>Ejemplo de la etiqueta meta</h2>
     <p>Este ejemplo muestra el uso de la etiqueta meta dentro de un documento
HTML</p>
  </body>
  </html>
```

A continuación se presentan algunas sintaxis específicas de la etiqueta **meta** que muestran sus diferentes usos.

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

Define la codificación de caracteres. El valor de **charset** es "utf-8", lo que significa que admitirá mostrar en cualquier idioma.

```
<meta name="keywords" content="Curso, Codigos, PILARES, Escuela">
```

-

Especifica la lista de *palabras clave* que utilizan los motores de búsqueda.

```
<meta name="description" content="Escuela de Codigos en PILARES">
```

Define la descripción del sitio web que es útil para proporcionar una búsqueda relevante realizada por los motores de búsqueda.

```
<meta name="author" content="ADIP">
```

Especifica el autor de la página. Es útil extraer la información del autor mediante el sistema de gestión de contenido automáticamente.

```
<meta name="refresh" content="50">
```

Especifica proporcionar instrucciones al navegador para actualizar automáticamente el contenido después de cada 50 segundos (o en cualquier momento dado).

```
<meta http-equiv="refresh" content="5; url=https://pilares.cdmx.gob.mx/inicio">
```

En el ejemplo anterior, hemos establecido una URL con contenido para que redirija automáticamente a la página dada después del tiempo proporcionado.

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

Especifica la ventana gráfica para controlar la dimensión de la página y la escala para que nuestro sitio web se vea bien en todos los dispositivos. Si esta etiqueta está presente, indica que esta página es compatible con dispositivos móviles.

Lenguaje CSS

Las hojas de estilo en cascada, tal es su traducción del inglés Cascading Style Sheets (CSS), tienen como función establecer reglas de representación de un documento en un medio o dispositivo. Mediante estas reglas podremos establecer medidas, colores o cualquier otra característica de representación de una página web, para que se vea reflejada en una pantalla de monitor, de un dispositivo móvil, una tablet, una impresora, un dispositivo braille o un televisor.

La función principal de CSS es, por lo tanto, la de permitir separar el contenido y la estructura que se define en un documento HTML, de la representación, que queda a cargo de las hojas de estilos.

Esta separación es importante para un proyecto web ya que, además de permitir la definición de criterios que se deben respetar en el sitio, ofrece la posibilidad de que se definan clases para evitar la necesidad de rescribir código y, además, se pueden crear reglas para que el sitio se represente de una manera correcta en diferentes dispositivos.

Tres de los principales beneficios para el uso de CSS son:

- Resuelve un gran problema. Antes de CSS, las etiquetas como fuente, color, estilo de fondo, alineación de elementos, borde y tamaño tenían que repetirse en cada página web. Este era un proceso muy largo. Por ejemplo: si estabas desarrollando un sitio web grande donde se agregaban fuentes e información de color en cada página, estoy se convertía en un proceso largo y costoso. CSS fue creado para resolver este problema.
- Ahorra mucho tiempo. Las definiciones de estilo CSS se guardan en archivos CSS externos, por lo que es posible cambiar todo el sitio web cambiando solo un archivo.
- Proporcionar más atributos. CSS proporciona atributos más detallados que usar simplemente HTML para definir la apariencia del sitio web.

Lenguaje CSS

Antes de comenzar a trabajar con CSS hay que conocer las diferentes formas para incluir estilos en nuestros documentos HTML, ya que hay varias, cada una con sus particularidades y diferencias. En principio, tenemos tres formas diferentes de hacerlo, siendo la primera la más común y la última la menos habitual:

Nombre	Método	Descripción
CSS Externo	Etiqueta **<link>**	El código se escribe en un archivo .css a parte. **Método más habitual**.
CSS Interno	Etiqueta **<style>**	El código se escribe en una etiqueta **<style>** en el documento HTML.
Estilos en línea	Atributo **style**="..."	El código se escribe en un atributo HTML de una etiqueta.

Veamos cada una de ellas detalladamente:
Enlace a CSS externo (link). En la cabecera de nuestro documento HTML, más concretamente en el bloque <head></head>, podemos incluir una etiqueta <link> con la que establecemos una relación entre el documento actual y el archivo CSS que indicamos en el atributo href:

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

El atributo type="text/css" no es necesario en HTML5. Muchas veces se indica para mantener retrocompatibilidad con navegadores muy antiguos, pero en la actualidad se puede omitir.

Nótese que, además de acceder a indicar la ruta en el href, también se procede a indicar el tipo de medio para el cual ha sido preparada la hoja de estilo; en este caso se trata de screen. Tengamos en cuenta que esto quiere decir que podemos contar con diferentes hojas de estilo para trabajar en el sitio y también para adaptar nuestro proyecto a la representación en distintas clases de dispositivos.

Lenguaje CSS

Incluir CSS en el HTML (style). Otra de las formas habituales que existen para incluir estilos CSS en nuestra página es la de añadirlos directamente en el documento HTML, a través de una etiqueta

```
<!DOCTYPE html>
<html>
 <head>
  <title>Título de la página</title>
  <style>
   div {
    background: hotpink;
    color: white;
   }
  </style>
 </head>
```

Nota: Aunque no es obligatorio, es muy común que las etiquetas <style> se encuentren en la cabecera <head> del documento HTML, ya que antiguamente era la única forma de hacerlo.

Este sistema puede servirnos en ciertos casos particulares, pero hay que darle prioridad al método anterior (CSS externo), ya que incluyendo el código CSS en el interior del archivo HTML arruinamos la posibilidad de tener el código CSS en un documento a parte, pudiendo reutilizarlo y enlazarlo desde otros documentos HTML mediante la etiqueta <link>.

Estilos en línea (atributo style). Por último, la tercera forma de aplicar estilos en un documento HTML es hacerlo directamente, a través del atributo style de la propia etiqueta donde queramos aplicar el estilo, colocando ahí las propiedades CSS:

```
<p>¡Hola <span style="color:red">amigo lector</span>!</p>
```

Lenguaje CSS

De la misma forma que en el método anterior, con la etiqueta <style>, se recomienda no utilizar este método salvo en casos muy específicos y justificados, ya que los estilos se asocian a la etiqueta HTML en cuestión y no pueden reutilizarse. Es una opción que puede venir bien en ciertos casos, pero se considera una mala práctica por muchos diseñadores cuando la sobre utilizas (sin una razón de peso) pudiendo utilizar el primer método. Al igual que los documentos HTML, los documentos CSS son archivos de texto donde se escribe una serie de órdenes y el cliente (navegador) las interpreta y aplica a los documentos HTML asociados. Sintaxis básica. La estructura CSS se basa en reglas que tienen el siguiente formato: Debemos escribir el selector, abrir llaves, indicar la propiedad y posteriormente asignarle el valor correspondiente. Cerramos la línea con punto y coma, y seguimos agregando pares de propiedad/valor hasta que completemos la declaración (cada una debe estar finalizada con punto y coma). Cuando terminamos, cerramos la llave. La declaración sería entonces:

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

- **Selector**: El selector es el elemento HTML que vamos a seleccionar del documento para aplicarle un estilo concreto, este podría ser el nombre un elemento, su clase o su identificador. Por ejemplo, con p seleccionaríamos todas las etiquetas <p> del HTML.
- **Propiedad**: La propiedad es una de las diferentes características que brinda el lenguaje CSS y que aplicaremos al selector para darle estilo.
- **Valor**: Cada propiedad CSS tiene una serie de valores concretos a que se le pueden asignar, con los que tendrá uno u otro comportamiento.

Lenguaje CSS

Con todo esto le iremos indicando al navegador que, para cada etiqueta (selector especificado) debe aplicar las reglas (propiedad y valor) indicadas.
Un ejemplo muy sencillo de lo anterior es el siguiente:

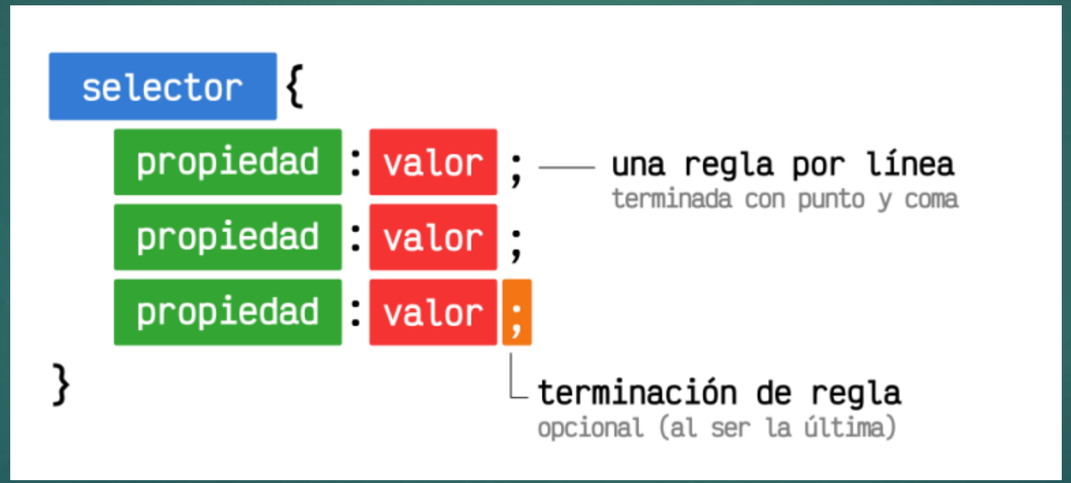
```
<!DOCTYPE html>
<html>
 <head>
  <title>PILARES Escuela de Códigos</title>
  <link rel="stylesheet" href="index.css" />
 </head>
 <body>
  <div id="first">
   <p>Texto en párrafo color rojo aplicado con CSS</p>
  </div>
  <div id="second">
   <span>Texto sin la aplicación de color con CSS</span>
  </div>
 </body>
```

```
p {
 color: red; /* Color de texto */
}
```

Si la primer parte del código se escribe y guarda en el editor de texto agregando al final la extensión .html, y la segunda parte del código para CSS se escribe y guarda en el editor de texto como index.css, lo que se obtiene al final al ejecutar simplemente el archivo con extensión .html es algo parecido a lo siguente:

En este caso, estamos seleccionando todas las etiquetas <p> del documento HTML (en este ejemplo es una sola, pero si existieran más se aplicaría a todas), y les aplicaremos el estilo indicado: color de texto rojo.

Nota: Se pueden incluir comentarios entre los caracteres /* y */, los cuales serán ignorados por el navegador. Estos suelen servir para añadir notas o aclaraciones dirigidas a humanos.
Sin embargo, esto es sólo un ejemplo muy sencillo. Se pueden aplicar muchas más reglas (no sólo una, como el color del ejemplo), consiguiendo así un conjunto de estilos para la etiqueta indicada en el selector.
Cada una de estas reglas se terminará con el carácter punto y coma (;), seguido de la siguiente regla. El último punto y coma es opcional y se puede omitir si se desea:



Lenguaje CSS

Para poder seleccionar elementos que se encuentran dentro de otros, se emplea lo que se conoce como selector descendiente.

```
p strong {color: red;}
```

Tendremos el texto del párrafo en color negro y lo que se envuelva dentro del párrafo con la etiqueta de color rojo. Si deseamos aplicar una misma regla a diversos elementos, podemos declararlos antes de abrir la llave, y separarlos con comas; por ejemplo, vamos a aplicarles color gris a todos los títulos de una página.

```
h1, h2, h3, h4, h5, h6 {color:grey;}
```

El uso de clases en CSS permite abstraerse del elemento en particular y construir reglas que se apliquen a toda aquella etiqueta HTML que cuente con el atributo class correspondiente. Aquí es importante resaltar que las clases pueden aplicarse a diversos elementos. Cuando construimos reglas para clases, debemos incluir el . (punto) antes del nombre.

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

Si creamos reglas para una determinada id, debemos anteponerle el símbolo #.

```
#nombre_id {propiedad:valor;}
```

Lenguaje CSS

Herencia en CSS

En primer lugar, debemos saber que algunas propiedades CSS se heredan desde los elementos padres a los elementos hijos, modificando el valor que tienen por defecto:

```
body {
  color: green;    /* Color de texto */
}
```

En el ejemplo anterior, aplicamos a la etiqueta HTML <body> el color de texto verde. En principio, esta propiedad aplicará dicho color a los textos que estén dentro de dicha etiqueta <body>.

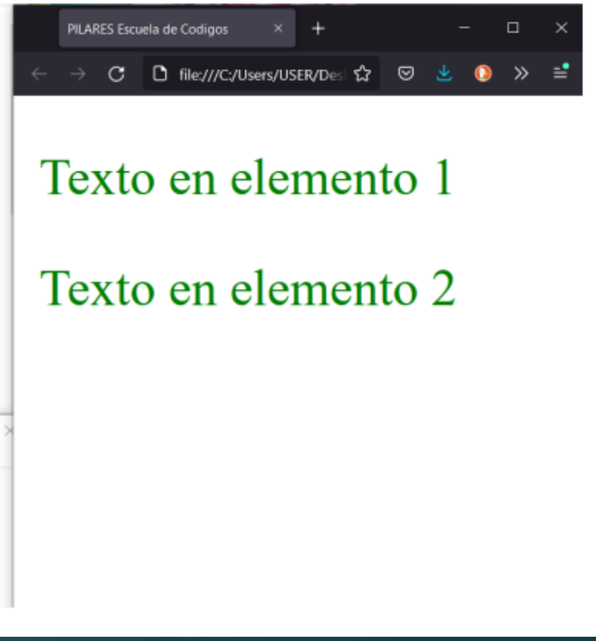
Sin embargo, si tenemos más etiquetas dentro, como por ejemplo una etiqueta con texto en su interior, si no tenemos aplicada una propiedad color a dicho elemento, veremos que también aparece en color verde. Esto ocurre porque la propiedad color es una de las propiedades CSS que, en el caso de no tener valor específico, hereda el valor de su elemento padre.

```
<!DOCTYPE html>
<html>
  <head>
    <title>PILARES Escuela de Codigos</title>
    <link rel="stylesheet" href="index.css" />
  </head>
  <body>
    <div id="first">
      <p>Texto en elemento 1</p>
    </div>
    <div id="second">
      <span>Texto en elemento 2</span>
    </div>
  </body>
</html>
```

index: Bloc de notas

Archivo Edición Formato Ver Ayuda

```
body {
  color: green;    /* Color de texto */
}
```



Lenguaje CSS

Hay que tener en cuenta que esto no ocurre si lo hacemos con otras propiedades CSS, como por ejemplo, con los bordes de un elemento HTML:

```
body {
  border-width: 2px;
  border-style: solid;
  border-color: red;
}
```

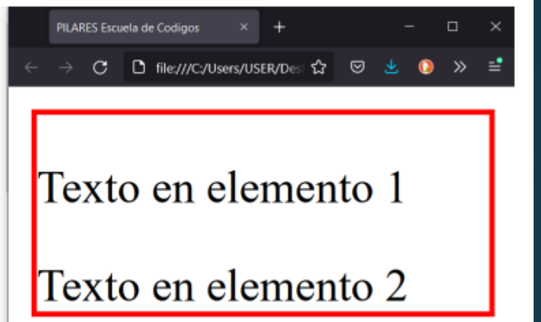
Si esta propiedad aplicara herencia, todos los elementos HTML situados en el interior de <body> tendrían un borde rojo, comportamiento que no suele ser el deseado. Por esa razón, la herencia no ocurre con todas las propiedades CSS, sino sólo con algunas propiedades como color o font, donde si suele ser deseable.

```
<!DOCTYPE html>
<html>
  <head>
    <title>PILARES Escuela de Codigos</title>
    <link rel="stylesheet" href="index.css" />
  </head>
  <body>
    <div id="first">
      <p>Texto en elemento 1</p>
    </div>
    <div id="second">
      <span>Texto en elemento 2</span>
    </div>
  </body>
</html>
```

index: Bloc de notas

Archivo Edición Formato Ver Ayuda

```
body {
  border-width: 2px;
  border-style: solid;
  border-color: red;
}
```



Lenguaje CSS

Valores especiales de herencia. Además de los valores habituales de cada propiedad CSS, también podemos aplicar ciertos valores especiales que son comunes a todas las propiedades existentes. Con estos valores modificamos el comportamiento de la herencia en dicha propiedad:

Valor	Significado
inherit	Hereda el valor que tiene la misma propiedad CSS en su elemento padre.
initial	Establece el **valor inicial** que tenía la propiedad CSS inicialmente.
unset	Combinación de las dos anteriores: Hereda el valor del padre, y en caso de no existir, su valor inicial.

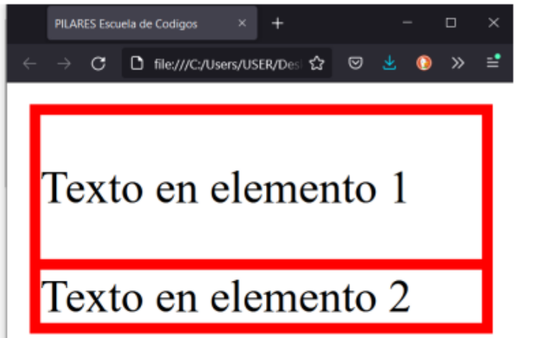
```
body {
  border-width: 2px;
  border-style: solid;
  border-color: red;
}

div {
  border: inherit;
}
```

```
<!DOCTYPE html>
<html>
  <head>
    <title>PILARES Escuela de Codigos</title>
    <link rel="stylesheet" href="index.css" />
  </head>
  <body>
    <div id="first">
      <p>Texto en elemento 1</p>
    </div>
    <div id="second">
      <span>Texto en elemento 2</span>
    </div>
  </body>
</html>
```

index: Bloc de notas

Archivo Edición Formato Ver Ayuda

```
body {
  border-width: 2px;
  border-style: solid;
  border-color: red;
}

div {
  border: inherit;
}
```



Cascada en CSS

Uno de los conceptos principales más importantes de CSS es el concepto denominado cascada. De hecho, la cascada es la que le da sentido a la C inicial en el nombre de CSS. Supongamos que aplicamos unos estilos CSS a exactamente el mismo selector (div) y donde coincide la propiedad CSS color con diferente valor en cada bloque:

```
<div>Texto del elemento</div>

<style>
div {
  color: red;
  padding: 8px
}

div {
  color: blue;
  background-color: grey
}
</style>
```

En este caso, ¿cuál de las dos reglas prevalece, si tenemos en cuenta que se refieren al mismo elemento y están al mismo nivel? La respuesta es muy fácil: Prevalece siempre la última regla definida, la cuál mezcla y sobre escribe las propiedades anteriores.

En el caso anterior, el resultado final (valor computado) sería el siguiente:

```
div {
  color: blue; /* Se sobreescribe la última */
  padding: 8px;
  background-color: grey;
}
```

Sin embargo, puede ocurrir que en determinados casos no esté tan claro cuál es el estilo que debería sobre escribir a los anteriores. Ahí es cuando entra en juego el concepto de cascada en CSS, que es el que se encarga de eliminar la ambigüedad y determinar el que tiene prioridad.
Supongamos el siguiente caso, donde tenemos un mismo elemento con un id y una clase:

```
<div id="nombre" class="clase">Texto del elemento</div>
<style>
div { color: red; }
#nombre { color: blue; }
.clase { color: green; }
</style>
```

Tenemos un elemento HTML que encaja con los tres bloques del ejemplo anterior. ¿Cómo sabe CSS que estilo aplicar? ¿Cuál tiene prioridad sobre los demás? Aquí es donde entra en acción el concepto de cascada en CSS.

El navegador, para saber que bloque de estilos tiene prioridad sobre los demás, analiza (por orden) tres conceptos clave del código CSS: su importancia, su especificidad y su orden. Veamos en que se basa cada uno de ellos.:

- La importancia de un código CSS se determina dependiendo de las hojas de estilo donde está colocado. Generalmente, no necesitaremos preocuparnos de este factor, pero siempre es una buena idea conocer como funciona. Existen varios tipos de hojas de estilo, de menor a mayor importancia

Tipo de CSS	Descripción	Definido por
Agente de usuario	Son los estilos CSS que aplica el navegador por defecto.	Navegador
CSS de usuario	Son los estilos CSS que añade el usuario, por razones de personalización.	Usuario
CSS de autor	Son los estilos CSS que coloca el autor	Desarrollador

```
<div>Texto del elemento</div>
<style>
div {
color: red !important;
padding: 8px
}
div {
color: blue;
background-color: grey
}
</style>
```

Aunque no es recomendable utilizarlo frecuentemente (puede convertirse en una mala práctica), se puede añadir al final de cada regla el texto !important, consiguiendo que la regla en cuestión tenga prioridad sobre las demás, independientemente del nivel o la altura a la que estén:

Nota que en el caso de que una misma propiedad del CSS de usuario y una propiedad del CSS de autor tuvieran !important, como caso excepcional tendría prioridad la del CSS de usuario sobre la del CSS de autor.

Lenguaje CSS

● Si la importancia no elimina la ambigüedad, se pasa a determinar la especificidad de los selectores CSS, que es uno de los criterios más importantes de la cascada de CSS. Para determinar la especificidad de un selector, se sigue un cálculo matemático que podríamos pensarlo como en un sistema en el cual se suman y comparan los puntos acumulados: los elementos valen 1 punto; las clases (class), 10 puntos; los identificadores (id), 100 puntos, y la declaración de estilo inline, 1000 puntos.
-Ganará la regla más específica.
-Si resulta que dos o más reglas tienen el mismo puntaje, triunfará la que se encuentre más abajo en la cascada.
-En todos los casos, una declaración inline será la más específica, salvo que otra tenga !important.

```
<button class=hi> Hola </button>

<style>
    .hi{background-color: green;}
    button{background-color: red;}
</style>
```

En este ejemplo hay dos reglas que afectan al mismo elemento. Aunque la regla de elemento botón está al final de la cascada, la regla de clase es más especifica y por tanto es la que se aplicara.

```
<button class=hi> Hola </button>

<style>
  hi {background-color: green;}
  button {background-color: red;}
  button .hi {background-color: yellow;}
</style>
```

Si nuestra regla incluye varios selectores el puntaje de estos se sumará, dándole mayor especificidadad.

Lenguaje CSS

En CSS, es posible crear múltiples reglas CSS para definir un mismo concepto. En este caso, la que prevalece ante todas las demás depende de ciertos factores, como es la «altura» a la que está colocada la regla y que da un orden: El CSS en linea en un elemento HTML es el que tiene mayor precedencia, por lo que siempre será el que tenga prioridad sobre otras reglas CSS. Recuerda que éstos son los estilos incluidos en una etiqueta HTML a través del atributo style. En segundo lugar, el CSS interno definido a través de bloques <style> en el propio documento HTML será el siguiente a tener en cuenta en orden de prioridad. Por último, los documentos CSS externos son la tercera opción de prioridad a la hora de tomar en cuenta las reglas CSS. Son aquellos que se relacionan en un documento HTML a través de la etiqueta <link>.
Teniendo esto en cuenta, hay que recordar que las propiedades que prevalecerán serán las que estén en último lugar, siempre respetando la prioridad de la lista anterior.

Lenguaje CSS

Selectores en CSS
Los selectores de CSS se utilizan para seleccionar el contenido que se desea diseñar. Los selectores son parte del conjunto de reglas CSS.
Hay varios tipos diferentes de selectores en CSS.

- Selector de elementos CSS
- Selector de ID de CSS
- Selector de clase CSS
- Selector universal CSS
- Selector de grupo CSS

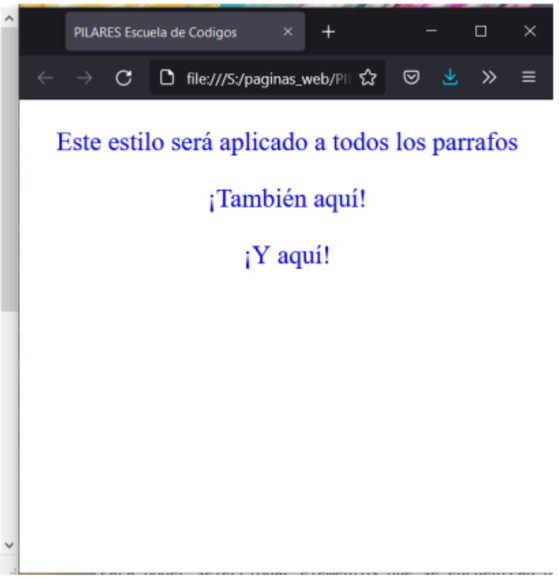
Selector de elementos CSS. El selector de elementos selecciona el elemento HTML por su nombre.
Ejemplo. Vamos a decirle al navegador que todas las etiquetas que encuentre en la página, le ponga color de fondo rojo.

```
div {
  background-color: red;
}
```

Ejemplo

```
<!DOCTYPE html>
<html>
  <head>
    <title>PILARES Escuela de Codigos</title>
  </head>
  <body>
        <p>Este estilo será aplicado a todos
los parrafos</p>
        <p id="para1">¡También aquí!</p>
        <p>¡Y aquí!</p>
<style>
        p{
                text-align: center;
                color: blue;
        }
</style>
  </body>
</html>
```

Línea 1, columna 1 | 150% | Windows (CRLF) | UTF-8



Lenguaje CSS

Selector de ID de CSS. El selector id selecciona el atributo id de un elemento HTML para seleccionar un elemento específico. Un id siempre es único dentro de la página, por lo que se elige para seleccionar un único elemento.
Se escribe con el carácter hash (#), seguido del id del elemento.
Ejemplo.

```
<p id="para1">Hola a PILARES</p>
<p>Este párrafo no se afectará.</p>
<style>
#para1 {
   text-align: center;
   color: blue;
}
</style>
```

En la práctica, los id no suelen utilizarse para dar estilo, ya que en la mayoría de los casos utilizar una clase es perfectamente válido y mucho más mantenible a la larga. La situación más recomendable para usar id es cuando queremos designar una zona del documento como una zona única que sabemos que no se va a repetir.

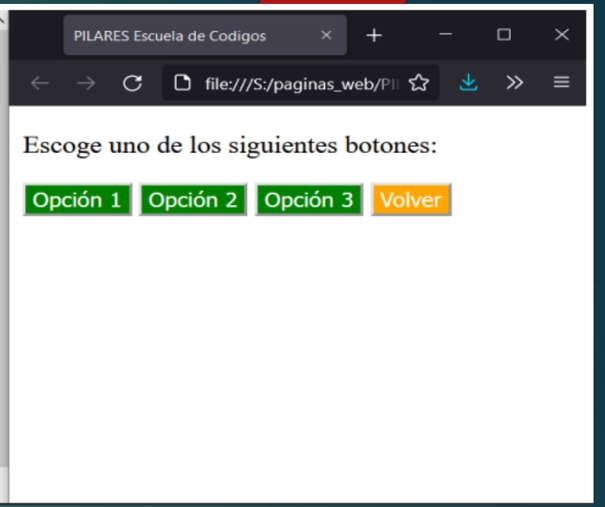
Lenguaje CSS

Selector de clase CSS. El selector class selecciona elementos HTML con un atributo de class específico. Se utiliza con un carácter de punto . (símbolo de punto) seguido del nombre de la clase. La diferencia principal respecto a los IDs es que las clases no se requiere que sean únicas, sino que pueden repetirse a lo largo del documento HTML. Nota que el nombre de una clase no debe comenzar con un número.
Ejemplo.

```
<h1 class="center">Este encabezado es azul y está alineado al centro.</h1>
<p class="center">Este párrafo es azul y está alineado al centro.</p>
<style>
.center {
   text-align: center;
   color: blue;
         }
</style>
```

Ejemplo.

```
<!DOCTYPE html>
<html>
  <head>
    <title>PILARES Escuela de Codigos</title>
  </head>
  <body>
    <div id="main">
    <p>Escoge uno de los siguientes botones:</p>

    <button class="classic">Opción 1</button>
    <button class="classic">Opción 2</button>
    <button class="classic">Opción 3</button>
    <button class="back">Volver</button>
  </div>

   <style>
     .classic {
  background-color: green;
  color: white;
}
```



Selecciones mixtas. Anteriormente en otro capítulo, vimos que es posible utilizar varias clases en un mismo elemento HTML, simplemente separando por espacios dentro del atributo class.
Ejemplo.

```
<button class="classic primary large">Botón</button>
```

De esta forma, a dicho elemento se le aplicarán los estilos de cada una de las clases indicadas, las cuales suelen tener un grupo de características relacionadas con su nombre, lo cuál puede ser muy interesante y práctico en algunos casos, dándonos mucha soltura a la hora de crear clases y reutilizarlas. Mencionar que también tenemos la posibilidad de ser más específicos y aplicar estilo sólo a los elementos HTML que contengan todas las clases indicadas, colocando cada clase de forma consecutiva en el CSS, de la siguiente forma:

```
button.classic.primary.large {
  /* Estilos CSS */
}
```

En este ejemplo, sólo se aplicarán los estilos cuando el elemento HTML <button> tenga una clase classic, una clase primary y una clase large. Si falta alguna de ellas, no se aplicará el estilo.

```
h1 {
   text-align: center;
   color: blue;
}
h2 {
   text-align: center;
   color: blue;
}
p {
   text-align: center;
   color: blue;
}
```

Selector de grupo CSS. El selector de agrupación se utiliza para seleccionar todos los elementos con las mismas definiciones de estilo. El selector de agrupación se utiliza para reducir el código. Las comas se utilizan para separar cada selector en la agrupación. Veamos el código CSS sin selector de grupo.

```
<h1>Hola</h1>
<h2>Bienvenidos</h2>
<p>Este es un párrafo</p>

<style>
h1,h2,p {
      text-align: center;
      color: blue;
         }
</style>
```

Como puedes ver, debes definir las propiedades CSS para todos los elementos. Si agrupamos quedaría de la siguiente manera:

Selector universal CSS. El selector universal (*) se utiliza como carácter comodín. Selecciona todos los elementos de las páginas.

```
/* Elimina márgenes y rellenos de todos los elementos de un documento HTML */
* {
  margin: 0;
  padding: 0;
}
```

Otros tipos de selectores:

Nombre	Símbolo	Ejemplo	Significado
Selector descendiente		#page div { }	Se aplican estilos a elementos dentro de otros.
Selector hijo	>	#page > div { }	Se aplican estilos a elementos hijos directos.
Selector hermano adyacente	+	div + div { }	Se aplican estilos a elementos que siguen a otros.
Selector hermano general	~	div ~ div { }	Se aplican estilos a elementos al mismo nivel.

Pseudoclases y Pseudoelementos

Pseudoclases en CSS. Una pseudoclase se puede definir como una palabra clave que se combina con un selector que define el estado especial de los elementos seleccionados. Se agrega al selector para agregar un efecto a los elementos existentes en función de sus estados. Los nombres de la pseudoclase no distinguen entre mayúsculas y minúsculas. Volvamos a recordar el esquema general de sintaxis de CSS:

```
selector #id .clase :pseudoclase ::pseudoelemento [atributo] {
    propiedad : valor ;
    propiedad : valor
}
```

Lenguaje CSS

Las pseudoclases se definen añadiendo dos puntos (:) antes de la pseudoclase concreta. En el caso de existir selectores de etiqueta, id o clases, estas se escribirían a su izquierda. Pseudoclases de enlaces. Existen algunas pseudoclases orientadas a los enlaces o hipervínculos. En este caso, permiten cambiar los estilos dependiendo del comportamiento del enlace:

pseudoclase	Descripción
:link	Aplica estilos cuando el enlace no ha sido visitado todavía.
:visited	Aplica estilos cuando el enlace ha sido visitado anteriormente.

```
<h1>Hola Mundo</h1>
  <h2>La pseudoclase :active </h2>
  <h3>Da Click en el enlace para ver el efecto</h3>
  <a href="#">Click aquí</a>

 <style>
   a:link {
 color: green;
 font-weight: bold
}
       a:visited {
 color: orange;
 font-weight: bold
}
  </style>
```

En donde se usa Font-weight: bold para hacer enmarcado en negritas

Lenguaje CSS

Pseudoclases de ratón. Originalmente, las siguientes pseudoclases se utilizaban solamente en enlaces (Internet Explorer no los soportaba en otros elementos). Sin embargo, actualmente pueden ser utilizadas con seguridad en cualquier otro elemento,
sin necesidad de ser <a>.

pseudoclase	Descripción
:hover	Aplica estilos cuando pasamos el ratón sobre un elemento.
:active	Aplica estilos cuando estamos pulsando sobre el elemento.

Observese que podemos realizar acciones un poco más específicas, como el segundo ejemplo anterior, donde al movernos sobre un elemento div (div:hover), aplicaremos los estilos a los enlaces (a) que están dentro del mencionado div. Por otro lado, la segunda pseudoclase, :active, permite resaltar los elementos que se encuentran activos, donde el usuario está pulsando de forma activa con el ratón:

```
<h1>Hola Mundo</h1>
  <h2>La pseudoclase :hover </h2>
  <h3>Pasa el ratón encima de los enlaces</h3>
  <a href="#5">Enlace1</a>

    <div>Aquí va texto <a href="#1">Enlace2</a> y sigue acá.</div>

  <style>
  /* Usuario mueve el ratón sobre un enlace */
   a:hover {
     background-color: cyan;
     padding: 2px
      }

  /* Usuario mueve el ratón sobre un div y resalta todos los enlaces que contiene */
   div:hover a{
     background-color: steelblue;
     color: white;
}
  </style>
```

Por otro lado, la segunda pseudoclase, :active, permite resaltar los elementos que se encuentran activos, donde el usuario está pulsando de forma activa con el ratón:

```
<h1>Pseudoclases</h1>
  <h2>La pseudoclase :active </h2>
  <h3>Da click en el enlace y manten presionado</h3>
  <a href="#5">Enlace1</a>

  <style>
  a:active {
    border: 2px solid #FF0000;
    padding: 2px
}
  </style>
```

Aunque las pseudoclases anteriores se inventaron para interactuar con un ratón en un sistema de escritorio, pueden funcionar en dispositivos táctiles. Aún así, ten en cuenta que, por ejemplo, el :hover no tiene mucho sentido en dispositivos móviles, ya que, aunque podría hacerlo, un usuario no navega por móvil arrastrando el dedo por la pantalla. Pseudoclases de interacción. Existen pseudoclases orientadas principalmente a los campos de formulario de páginas webs y la interacción del usuario con ellos, veamos otro par interesante:

pseudoclase	Descripción
:focus	Aplica estilos cuando el elemento tiene el foco.
:checked	Aplica estilos cuando la casilla está seleccionada.

```
<form>
<h1>Nombre: <input type="text" value="Introduce tu
nombre"></h1>
</form>
<style>
form{
        text-align:center;
}
input:focus{
        border:5px solid lightblue;
        box-shadow:10px 10px 10px black;
        color: blue;
        width:300px;
}
</style>
```

Aunque estas pseudoclases suelen utilizarse con elementos de formularios como <input>, también pueden utilizarse con otros elementos, como por ejemplo los enlaces <a>. Esta es una excelente oportunidad para personalizar el estilo de los campos de texto de un formulario (<input> y <textarea>) mientras el usuario escribe y se mueve por ellos.

Cuando estamos escribiendo en un campo de texto de un formulario de una página web, generalmente pulsamos TAB para cambiar al siguiente campo y SHIFT+TAB para volver al anterior. Cuando estamos posicionados en un campo se dice que ese campo tiene el foco, mientras que al pulsar TAB y saltar al siguiente, decimos que pierde el foco.
El comportamiento de «ganar el foco» puede gestionarse mediante la pseudoclase :focus:

Lenguaje CSS

Por otro lado, la pseudoclase :checked permite aplicar el estilo especificado a los elementos <input> (casillas de verificación o botones de radio) u <option> (la opción seleccionada de un <select>).
Ejemplo.

```
<form>
      <br><label>Género</label><br>
            <input type="radio" id="gender" name="gender"
value="male"/><span>Masculino</span> <br>
            <input type="radio" id="gender" name="gender"
value="female"/>Femenino<br/>
            <input type="radio" id="gender" name="gender"
value="others"/>Otro <br/>
   </form>

<style>
   input:checked + span {
      color: green;
      }
    input[type="radio"]:checked {
        box-shadow: 0 0 0 3px orange;
         }
</style>
```

Este ejemplo añade el selector hermano + para darle formato al que contiene el texto y se encuentra colocado a continuación de la casilla <input>. De esta forma, los textos que hayan sido seleccionados, se mostrarán en verde. También da color a los botones de radio que aparecen. Pseudoclases de activación. Por norma general, los elementos de un formulario HTML están siempre activados, aunque se pueden desactivar añadiendo el atributo disabled (es un atributo booleano, no lleva valor) al elemento HTML en cuestión. Esto es una práctica muy utilizada para impedir al usuario escribir en cierta parte de un formulario porque, por ejemplo, no es aplicable. Existen varias pseudoclases para detectar si un campo de un formulario está activado o desactivado:

Pseudoclase	Descripción
:enabled	Aplica estilos cuando el campo del formulario está activado.
:disabled	Aplica estilos cuando el campo del formulario está desactivado.
:read-only	Aplica estilos cuando el campo es de sólo lectura.
:read-write	Aplica estilos cuando el campo es editable por el usuario.

Lenguaje CSS

Utilizando las dos primeras pseudoclases, bastante autoexplicativas por si solas, podemos seleccionar elementos que se encuentren activados (comportamiento por defecto) o desactivados:

```
<form action="url_of_form">
<label for="FirstField">Campo 1 (abilitado):</label>
<input type="text" id="FirstField" value="Escribe aquí"><br>
<label for="SecondField">Campo 2 (desabilitado):</label>
<input type="text" id="SecondField" value="No está
abilitado" disabled="disabled"><br>
<input type="button" value="Enviar">
</form>
<style>
input:enabled {
color: cyan;
background-color: orange;
}
input:disabled {
color: #aaa;
background-color: blue;
}
</style>
```

Por otro lado, las pseudoclases read-only y read-write nos permiten seleccionar y diferenciar elementos que se encuentran en modo de solo lectura (tienen especificado el atributo readonly en el HTML) o no:

```
  <h1>Demostración de la pseudoclase :read-only</h1>
 <p>La pseudoclase :read-only elije elementos de formulario
con un atributo "readonly":</p>
   <p>Un elemento de entrada de solo lectura:<br><input
readonly value="Hola"></p>

<style>
input:read-only {
 background-color: yellow;
}
</style>
```

En el ejemplo anterior, la pseudoclase :read-only le da estilo a aquellos campos <input> de un formulario que están marcados con el atributo de sólo lectura readonly. La diferencia entre un campo con atributo disabled y un campo con atributo readonly es que la información del campo con readonly se enviará a través del formulario, mientras que la del campo con disabled no se enviará. Aún así, ambas no permiten modificar el valor.

Por otro lado, la pseudoclase :read-write es muy útil para dar estilos a todos aquellos elementos que son editables por el usuario, sean campos de texto <input> o <textarea>.

```
    <h1>Demostración de la pseudoclase :read-write</h1>
<p>La pseudoclase :read-write elije elementos de formulario sin el atributo "readonly":</p>

<p>Un elemento normal de entrada:<br><input value="Aquí puedes escribir"></p>

<style>
input:read-only {
  background-color: yellow;
}
</style>
```

La pseudoclase read-write da estilo también a elementos HTML que contengan el atributo contenteditable, como por ejemplo un párrafo editable por el usuario con dicho atributo.

```
        <p contenteditable>Este párrafo es editable; es read-write.</p>

<style>
p:read-write {
  background-color: yellow;
   }
</style>
```

Pseudoclases de validación. En HTML5 es posible dotar de capacidades de validación a los campos de un formulario, pudiendo interactuar desde Javascript o incluso desde CSS. Con estas validaciones podemos asegurarnos de que el usuario escribe en un campo de un formulario el valor esperado que debería. Existen algunas pseudoclases útiles para las validaciones, como por ejemplo las siguientes:

Pseudoclase	¿Cuándo aplica estilos?
:required	Cuando el campo es obligatorio, o sea, tiene el atributo required.
:optional	Cuando el campo es opcional (por defecto, todos los campos).
:invalid	Cuando los campos no cumplen la validación HTML5.
:valid	Cuando los campos cumplen la validación HTML5.
:out-of-range	Cuando los campos numéricos están fuera del rango.
:in-range	Cuando los campos numéricos están dentro del rango.

Pseudoclases de negación. Existe una pseudoclase muy útil, denominada pseudoclase de negación. Permite seleccionar todos los elementos que no cumplan los selectores indicados entre paréntesis. Veamos un ejemplo:

```
p:not(.general) {
  border: 1px solid #DDD;
  padding: 8px;
  background: #FFF;
}
```

Este pequeño fragmento de código nos indica que todos los párrafos (elementos <p>) que no pertenezcan a la clase general, se les aplique el estilo especificado. Las reglas de negación pueden ser complejas, ineficientes y poco escalables. Intenta utilizarlas sólo en los casos que sea absolutamente necesario.

Lenguaje CSS

Otras pseudoclases.

Pseudoclase	Significado
:lang(es)	Aplica estilo a los elementos con el atributo lang="es".
:target	Aplica estilo al elemento que coincida con el ancla de la URL.
:root	Aplica estilo al elemento raíz (padre) del documento.
:default	Experimental. Aplica estilo al elemento por defecto. Útil en formularios.
:dir(A)	Experimental. Aplica estilo al elemento que coincide con la dirección ltr o rtl.
:indeterminate	Experimental. Aplica estilo a la casilla checkbox o al elemento <progress> sin definir.
:fullscreen	Experimental. Aplica estilo si la página está en el modo de pantalla completa.
:scope	Experimental. Aplica estilo a los elementos en el ámbito indicado.
:any(A)	Experimental. Aplica estilo si coincide con algún elemento indicado en A.

Lenguaje CSS

Pseudoelementos en CSS. Una pseudoclase se puede definir como una palabra clave que se combina con un selector que define el estado especial de los elementos seleccionados. A diferencia de las pseudoclases, los pseudoelementos se utilizan para diseñar la parte específica de un elemento, mientras que las pseudoclases se utilizan para diseñar el elemento. Al igual que las pseudoclases, los pseudoelementos son otra de las características de CSS que permiten hacer referencias a «comportamientos virtuales no tangibles», o lo que es lo mismo, se le puede dar estilo a elementos que no existen realmente en el HTML, y que se pueden generar desde CSS. Recordemos la sintaxis de los pseudoelementos, que está precedida de dos puntos dobles (::) para diferenciarlos de las pseudoclases, las cuales sólo tienen dos puntos (:). No obstante, este cambio surgió posteriormente, por lo que aún hoy en día es frecuente ver fragmentos de código con pseudoelementos con la sintaxis de pseudoclase con un
solo par de puntos.

```
selector #id .clase :pseudoclase ::pseudoelemento [atributo] {
    propiedad : valor ;
    propiedad : valor
}
```

Dentro de la categoría de los pseudoelementos CSS, como punto central, se encuentra la propiedad content. Esta propiedad se utiliza en selectores que incluyen los pseudoelementos ::before o ::after, para indicar que vamos a crear contenido antes o después del elemento en cuestión:

Lenguaje CSS

Propiedad/Pseudoelemento	Descripción
content	Propiedad para generar contenido. Sólo usable en ::before o ::after.
::before	Aplica los estilos **antes** del elemento indicado.
::after	Aplica los estilos **después** del elemento indicado.

La propiedad **content** admite parámetros de diverso tipo, incluso concatenando información mediante espacios. Podemos utilizar tres tipos de contenido:

Valor	Descripción	Ejemplo
"string"	Añade el contenido de texto indicado.	content: "Contenido:";
attr(*atributo***)**	Añade el valor del atributo HTML indicado.	content: attr(href);
url(*URL***)**	Añade la imagen indicada en la URL.	content: url(icon.png);

Por otro lado, los pseudoelementos ::before y ::after permiten hacer referencia a «justo antes del elemento» y «justo después del elemento», respectivamente. Así, se podría generar información (usualmente con fines decorativos) que no existe en el HTML, pero que por circunstancias de diseño sería apropiado colocar:

```
<p>El objetivo de WWF es:
<q>Construir un futuro donde las personas vivan en armonía con la naturaleza.</q>
Esperamos que tengan éxito.</p>

<style>
 q::before {
   content: "«";
   color: #888;
}
 q::after {
   content: "»";
   color: #888;
}
</style>
```

Los ejemplos anteriores insertan el carácter « antes de las citas indicadas con el elemento HTML <q> y el carácter » al finalizar la misma, ambas de color gris.

Lenguaje CSS

Atributos HTML. Es interesante recalcar la utilidad de la expresión attr(), que en lugar de generar el contenido textual que le indiquemos, permite recuperar esa información del valor del atributo HTML especificado. Veamos un ejemplo para clarificarlo, concatenándolo con texto:

```
<a href="https://adip.cdmx.gob.mx/">Visita ADIP!</a>

<style>
 a::after {
 content: " ( " attr(href) " )";
}
</style>
```

Este pequeño ejemplo muestra a continuación de todos los enlaces la URL literalmente, dentro de dos paréntesis. Esto puede ser realmente útil en una página de estilos que se aplica a una página en el momento de imprimir, en los cuales se pierde la información del enlace al no ser un medio interactivo.

Primera letra y primera línea. También existen pseudoelementos con los que podemos hacer referencia a la primera letra de un texto. Para ello utilizamos el pseudoelemento ::first-letter, así como el pseudoelemento ::first-line si queremos hacer referencia a la primera línea de un texto. De esta forma, podemos dar estilo a esas secciones concretas del texto:

Pseudoelemento	Descripción
::first-letter	Aplica los estilos en la primera letra del texto.
::first-line	Aplica los estilos en la primera línea del texto.

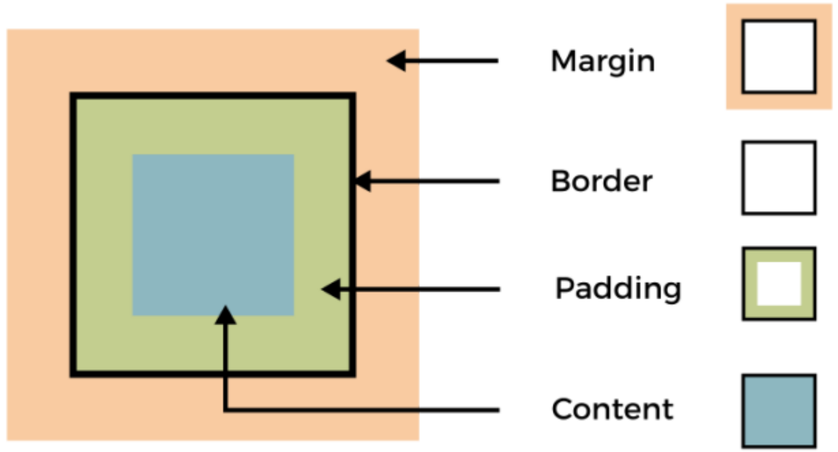
```
<p> CSS está diseñado principalmente para marcar la separación del contenido del documento y la forma de presentación de este, características tales como las capas o layouts, los colores y las fuentes.</p>
<style>
p {
color: green;
font-family: Verdana, sans-serif;
font-size: 16px;
}
p::first-letter {
color: red;
font-family: 'Times New Roman', serif;
font-size: 42px;
}
p::first-line {
color: orange;
}
</style>
```

En donde **font-size: 42px;** se usa para controlar el tamaño del texto Y el **font-family** para cambiar la fuente.

Modelo de caja CSS

Cuando hablamos del modelo de cajas en CSS, estamos haciendo referencia a un sistema que tiene el navegador de interpretar las diferentes partes de lo que solemos denominar «caja»: un elemento HTML con unas ciertas dimensiones.

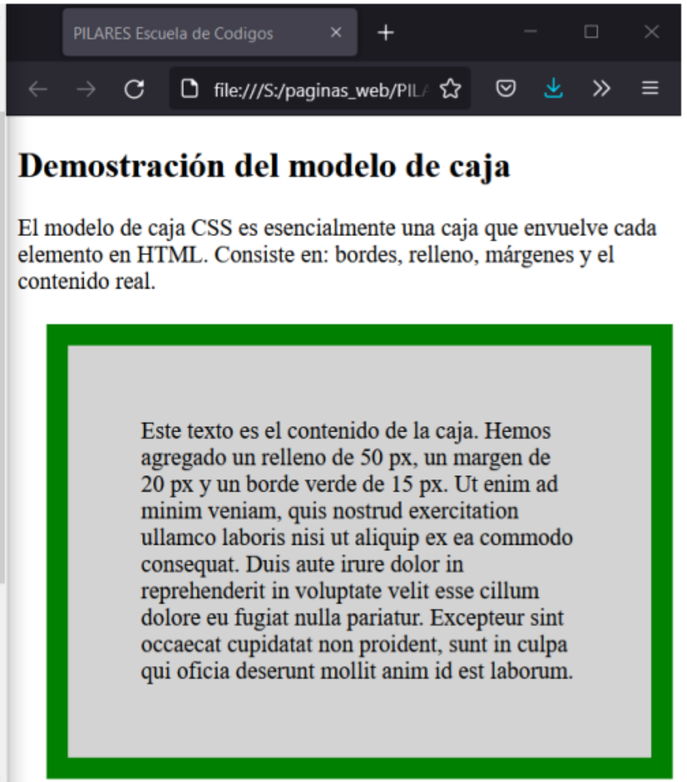
La representación básica del modelo de cajas se basa en varios conceptos importantes, como veremos a continuación:



El borde (border), en negro, es el límite que separa el interior del exterior del elemento.

- El márgen (margin), en naranja, es la parte exterior del elemento, por fuera del borde.
- El relleno (padding), en verde, es la parte interior del elemento, entre el contenido y el borde (no se permiten valores negativos).
- El contenido, en azul, es la parte interior del elemento, excluyendo el relleno.

```
<h2>Demostración del modelo de caja</h2>

<p>El modelo de caja CSS es esencialmente una caja que
envuelve cada elemento en HTML. Consiste en: bordes, relleno,
márgenes y el contenido real.</p>

<div>Este texto es el contenido de la caja. Hemos agregado un
relleno de 50 px, un margen de 20 px y un borde verde de 15
px. Ut enim ad minim veniam, quis nostrud exercitation
ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis
aute irure dolor in reprehenderit in voluptate velit esse
cillum dolore eu fugiat nulla pariatur. Excepteur sint
occaecat cupidatat non proident, sunt in culpa qui oficia
deserunt mollit anim id est laborum.</div>

<style>
  div {
  background-color: lightgrey;
  width: 300px;
  border: 15px solid green;
  padding: 50px;
  margin: 20px;
}
</style>


  </body>
</html>
```



El border también trabaja de forma:

border-width: 10px 1px 5 px 20 px;

Y se interpreta:
Borde superior =10px
Borde inferior= 5px
Borde izquierdo= 20 px
Borde derecho= 1px

(Horario)

Es importante que cuando se establecen las propiedades de ancho y alto de un elemento con CSS, se establecen de igual manera el ancho y el alto del área de contenido. Para calcular el tamaño completo de un elemento, también debes agregar el tamaño del relleno, bordes y márgenes.
Ejemplo. Este elemento que se muestra a continuación tendrá un ancho total de 350px:

```
<h2>Calcular el ancho total:</h2>
<img src="ADIP.jpg" width="350" height="263" alt="Agencia Digital de Innovación Pública">
<div>La imagen de arriba tiene 350px de ancho. El ancho total de este elemento también es 350px.</div>

<style>
 div {
  width: 320px;
  padding: 10px;
  border: 5px solid gray;
  margin: 0;
}
</style>
```

Aquí está el cálculo:

320px (ancho)
+ 20px (relleno izquierdo + derecho)
+ 10px (borde izquierdo + derecho)
+ 0px (margen izquierdo + derecho)
= **350 píxeles**

El **ancho total** de un elemento debe calcularse así:

Ancho total del elemento = ancho + relleno izquierdo + relleno derecho + borde izquierdo + borde derecho + margen izquierdo + margen derecho

La **altura total** de un elemento debe calcularse así:

Altura total del elemento = altura + relleno superior + relleno inferior + borde superior + borde inferior + margen superior + margen inferior

Estilos en Textos

Nos encargaremos ahora muy brevemente de analizar aquellas propiedades que se relacionan con las características de texto.

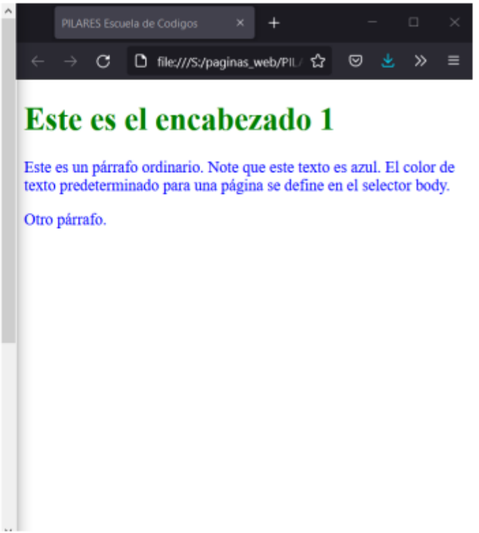
Color de texto. La propiedad de color se utiliza para establecer el color del texto. El color se especifica por:

- Un nombre de color - como "red",
- un valor HEX - como "#ff0000",
- un valor RGB - como "rgb(255,0,0)".

Ejemplo.

```
<h1>Este es el encabezado 1</h1>
<p>Este es un párrafo ordinario. Note que este texto es azul. El color de texto predeterminado para una página se define en el selector body.</p>
<p>Otro párrafo.</p>

<style>
 body {
   color: blue;
    }

 h1 {
   color: green;
   }
</style>
```

```
<!DOCTYPE html>
<html>
  <head>
    <title>PILARES Escuela de Codigos</title>
  </head>
  <body>

<h1>Este es el encabezado 1</h1>
<p>Este es un párrafo ordinario. Note que este texto es azul. El color de texto predeterminado para una página se define en el selector body.</p>
<p>Otro párrafo.</p>

<style>
  body {
    color: blue;
       }

  h1 {
    color: green;
     }
</style>

  </body>
</html>
```



Lenguaje CSS

Espaciado entre palabras. La propiedad word-spacing se utiliza para especificar el espacio entre las palabras de un texto. Puede recibir el valor normal (por defecto), una unidad de longitud o inherit.
El siguiente ejemplo demuestra cómo aumentar o disminuir el espacio entre palabras:

```
<h1>Uso de espacios entre palabras</h1>
<p>Este es un párrafo con espaciado entre palabras normal.</p>
<p class="one">Este es un párrafo con espacio entre palabras más grande.</p>
<p class="two">Este es un párrafo con espacio entre palabras más pequeño.</p>

<style>
  p.one {
  word-spacing: 10px;
}
p.two {
  word-spacing: -2px;
}
</style>
```

```
<!DOCTYPE html>
<html>
  <head>
    <title>PILARES Escuela de Codigos</title>
  </head>
  <body>

<h1>Uso de espacios entre palabras</h1>

<p>Este es un párrafo con espaciado entre palabras
normal.</p>

<p class="one">Este es un párrafo con espacio entre
palabras más grande.</p>

<p class="two">Este es un párrafo con espacio entre
palabras más pequeño.</p>


<style>
  p.one {
  word-spacing: 10px;
}

p.two {
  word-spacing: -2px;
}
</style>

  </body>
</html>
```

Uso de espacios entre palabras

Este es un párrafo con espaciado entre palabras normal.

Este es un párrafo con espacio entre palabras más grande.

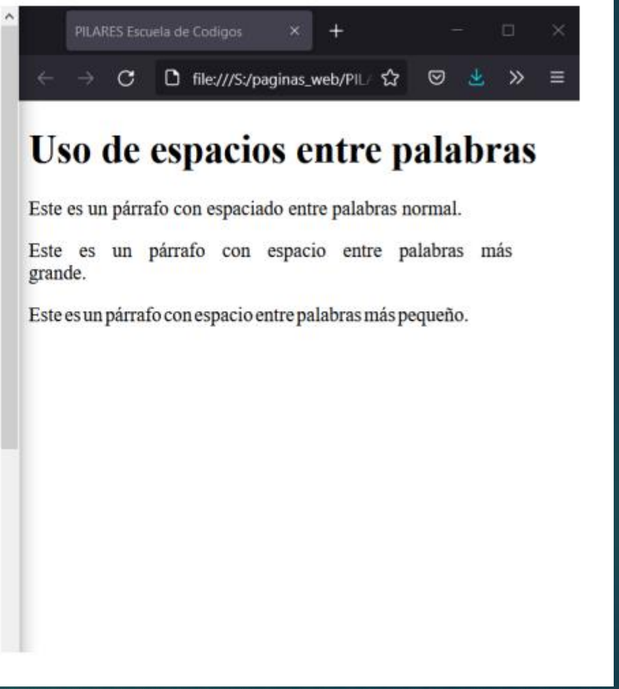
Este es un párrafo con espacio entre palabras más pequeño.

Lenguaje CSS

Espaciado de letras. La propiedad letter-spacing se utiliza para especificar el espacio entre los caracteres de un texto. Puede recibir el valor normal (por defecto), una unidad de longitud o inherit.
El siguiente ejemplo demuestra cómo aumentar o disminuir el espacio entre caracteres:

```
<h1>Uso de espaciado entre letras</h1>
<h2>Este es el encabezado 1</h2>
<h3>Este es el encabezado 2</h3>

<style>
 h2 {
   letter-spacing: 5px;
}

 h3 {
   letter-spacing: -2px;
}
</style>
```

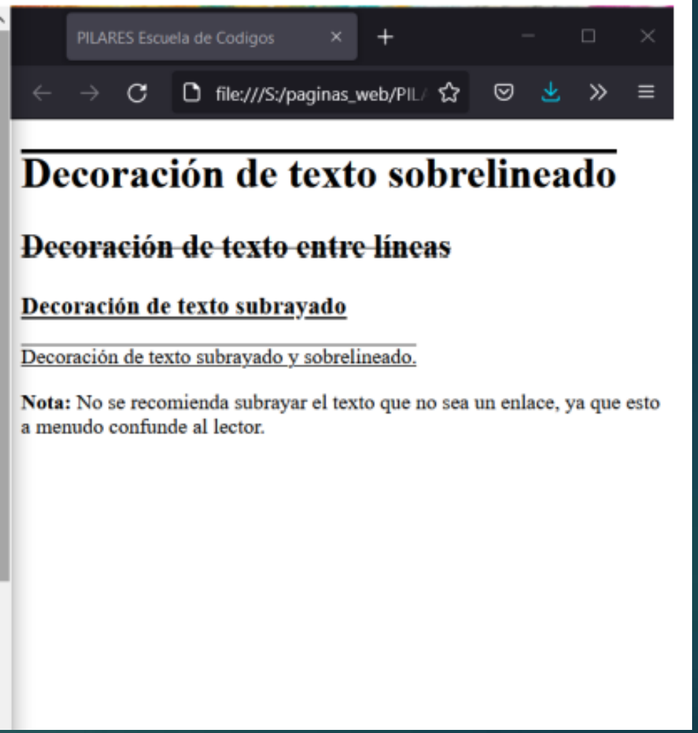
```
<!DOCTYPE html>
<html>
  <head>
    <title>PILARES Escuela de Codigos</title>
  </head>
  <body>

<h1>Uso de espacios entre palabras</h1>

<p>Este es un párrafo con espaciado entre palabras
normal.</p>

<p class="one">Este es un párrafo con espacio entre
palabras más grande.</p>

<p class="two">Este es un párrafo con espacio entre
palabras más pequeño.</p>


<style>
  p.one {
  word-spacing: 10px;
}

p.two {
  word-spacing: -2px;
}
</style>

  </body>
</html>
```



Línea de Decoración de texto CSS . La propiedad text-decoration-line se usa para agregar una línea de decoración al texto. Se puede combinar más de un valor, como tachado y subrayado, para mostrar líneas tanto encima como debajo de un texto.

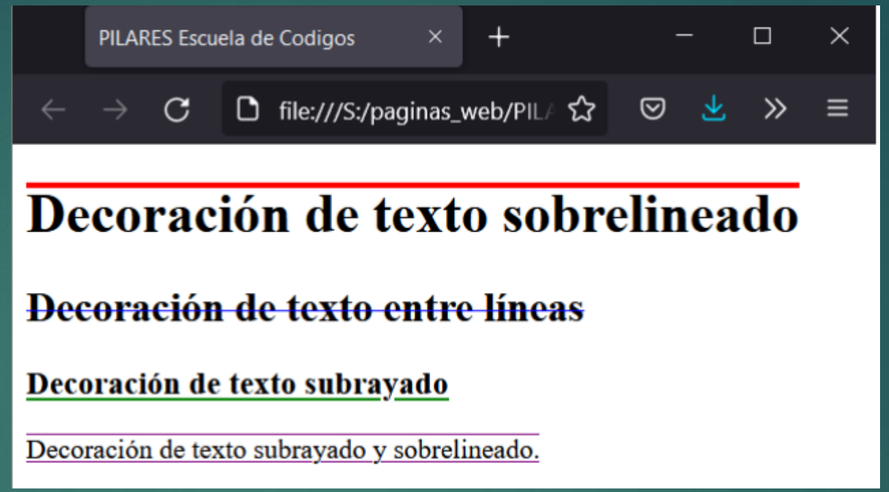
```
<h1>Decoración de texto sobrelineado</h1>
<h2>Decoración de texto entre líneas</h2>
<h3>Decoración de texto subrayado</h3>
<p class="ex">Decoración de texto subrayado y sobrelineado.</p>

<p><strong>Nota:</strong> No se recomienda subrayar el texto que no sea un enlace, ya que esto a menudo confunde al lector.</p>

<style>
 h1 {
 text-decoration: overline;
}

h2 {
 text-decoration: line-through;
}

h3 {
 text-decoration: underline;
}

p.ex {
 text-decoration: overline underline;
}
</style>
```

```
<!DOCTYPE html>
<html>
  <head>
    <title>PILARES Escuela de Codigos</title>
  </head>
  <body>

<h1>Decoración de texto sobrelineado</h1>
<h2>Decoración de texto entre líneas</h2>
<h3>Decoración de texto subrayado</h3>
<p class="ex">Decoración de texto subrayado y sobrelineado.</p>

<p><strong>Nota:</strong> No se recomienda subrayar el texto que no sea un enlace, ya que esto a menudo confunde al lector.</p>

<style>
  h1 {
  text-decoration: overline;
}

h2 {
  text-decoration: line-through;
}

h3 {
  text-decoration: underline;
}

p.ex {
  text-decoration: overline underline;
}
</style>

  </body>
</html>
```



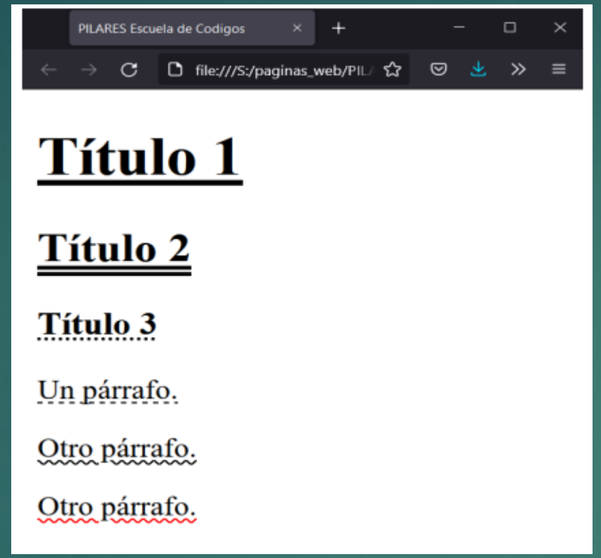
Especifique un color para la línea de decoración. La propiedad text-decoration-color se utiliza para establecer el color de la línea de decoración.

```
<h1>Decoración de texto sobrelineado</h1>
<h2>Decoración de texto entre líneas</h2>
<h3>Decoración de texto subrayado</h3>
<p>Decoración de texto subrayado y sobrelineado.</p>

<style>
 h1 {
 text-decoration-line: overline;
 text-decoration-color: red;
}

h2 {
 text-decoration-line: line-through;
 text-decoration-color: blue;
}

h3 {
 text-decoration-line: underline;
 text-decoration-color: green;
}

p {
 text-decoration-line: overline underline;
 text-decoration-color: purple;
}
</style>
```



Especifique un estilo para la línea de decoración. La propiedad text-decoration-style se utiliza para establecer el estilo de la línea de decoración.

```
<h1>Título 1</h1>
<h2>Título 2</h2>
<h3>Título 3</h3>
<p class="ex1">Un párrafo.</p>
<p class="ex2">Otro párrafo.</p>
<p class="ex3">Otro párrafo.</p>

<style>
  h1 {
  text-decoration-line: underline;
  text-decoration-style: solid; /* this is default */
}

h2 {
  text-decoration-line: underline;
  text-decoration-style: double;
}

h3 {
  text-decoration-line: underline;
  text-decoration-style: dotted;
}

p.ex1 {
  text-decoration-line: underline;
  text-decoration-style: dashed;
}

p.ex2 {
  text-decoration-line: underline;
  text-decoration-style: wavy;
}

p.ex3 {
  text-decoration-line: underline;
  text-decoration-color: red;
  text-decoration-style: wavy;
}
</style>
```

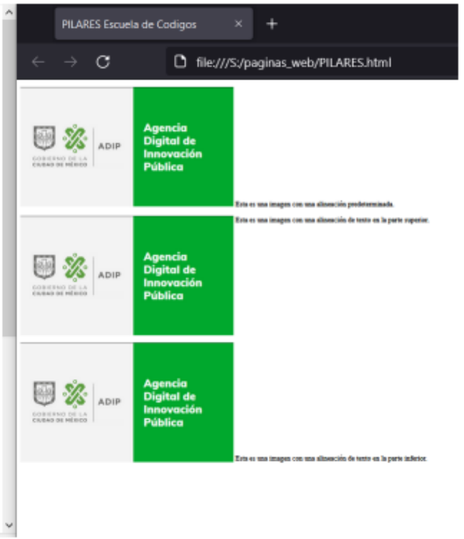


Lenguaje CSS

Alineamiento vertical. La propiedad de vertical-align establece la alineación vertical de un elemento.

```
<p><img src="ADIP_.png" alt="Innovación Pública"/> Esta es una imagen con una alineación predeterminada.</p>
<p><img src="ADIP_.png" class="top" alt="Innovación Pública"/> Esta es una imagen con una alineación de texto en la parte superior.</p>
<p><img src="ADIP_.png" class="bottom" alt="Innovación Pública"/> Esta es una imagen con una alineación de texto en la parte inferior.</p>

<style>
 img.top {
   vertical-align: text-top;
}
img.bottom {
   vertical-align: text-bottom;
}
</style>
```

```
<!DOCTYPE html>
<html>
  <head>
    <title>PILARES Escuela de Codigos</title>
  </head>
  <body>

<p><img src="ADIP_.png" alt="Innovación Pública"/>
Esta es una imagen con una alineación
predeterminada.</p>
<p><img src="ADIP_.png" class="top" alt="Innovación
Pública"/> Esta es una imagen con una alineación de
texto en la parte superior.</p>
<p><img src="ADIP_.png" class="bottom"
alt="Innovación Pública"/> Esta es una imagen con
una alineación de texto en la parte inferior.</p>

<style>
  img.top {
    vertical-align: text-top;
}
img.bottom {
    vertical-align: text-bottom;
}
</style>

  </body>
</html>
```

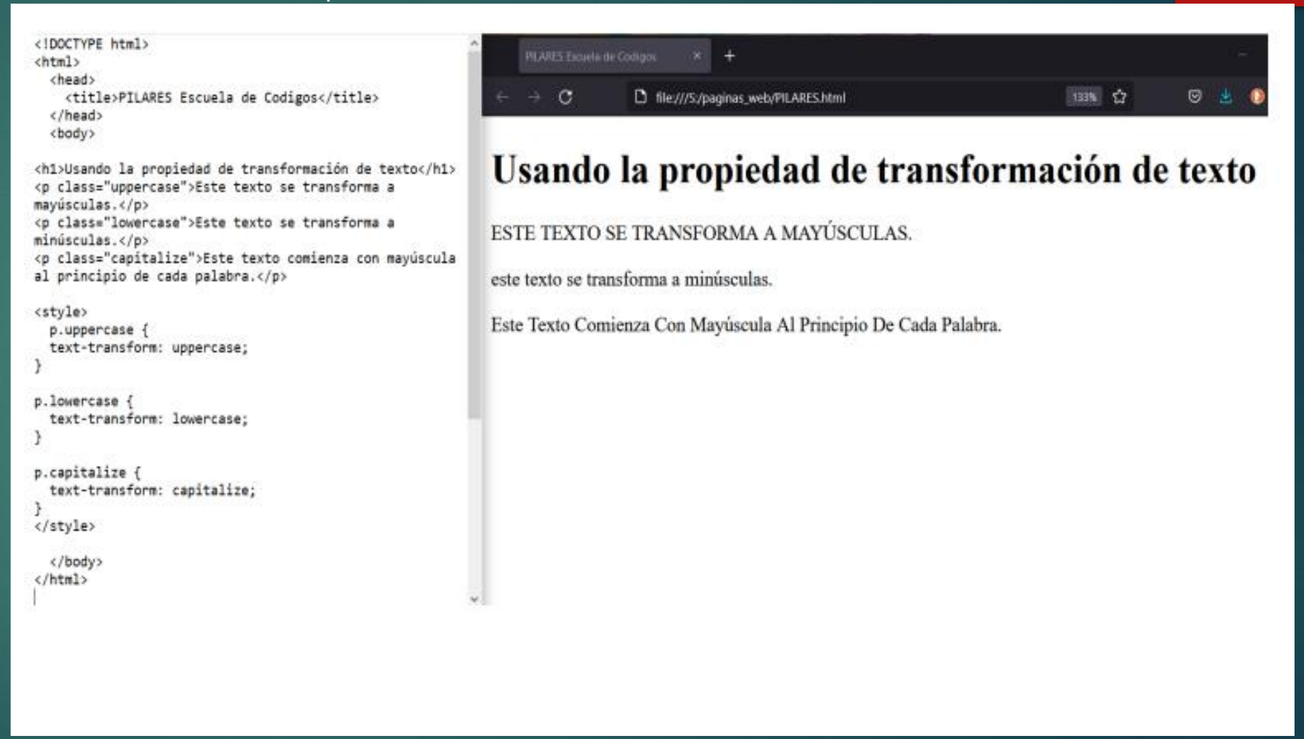


Lenguaje CSS

Transformación de texto. La propiedad text-transform se utiliza para especificar letras mayúsculas y minúsculas en un texto. Se puede usar para convertir todo en letras mayúsculas o minúsculas, o poner en mayúscula la primera letra de cada palabra:

```
<h1>Usando la propiedad de transformación de texto</h1>
<p class="uppercase">Este texto se transforma a mayúsculas.</p>
<p class="lowercase">Este texto se transforma a minúsculas.</p>
<p class="capitalize">Este texto comienza con mayúscula al principio de cada palabra.</p>

<style>
 p.uppercase {
 text-transform: uppercase;
}

p.lowercase {
 text-transform: lowercase;
}

p.capitalize {
 text-transform: capitalize;
}
</style>
```

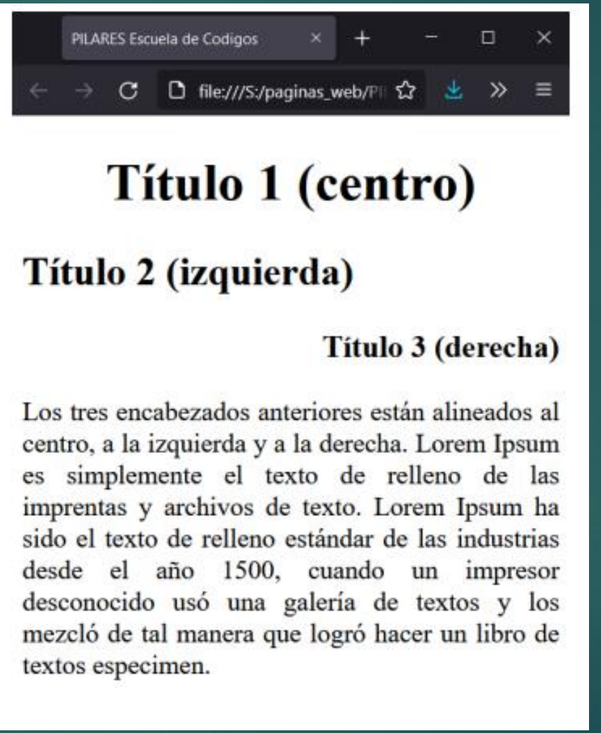
```
<!DOCTYPE html>
<html>
  <head>
    <title>PILARES Escuela de Codigos</title>
  </head>
  <body>

<h1>Usando la propiedad de transformación de texto</h1>
<p class="uppercase">Este texto se transforma a
mayúsculas.</p>
<p class="lowercase">Este texto se transforma a
minúsculas.</p>
<p class="capitalize">Este texto comienza con mayúscula
al principio de cada palabra.</p>

<style>
  p.uppercase {
  text-transform: uppercase;
}

p.lowercase {
  text-transform: lowercase;
}

p.capitalize {
  text-transform: capitalize;
}
</style>

  </body>
</html>
```



Lenguaje CSS

Alineación del texto. La propiedad text-align se utiliza para establecer la alineación horizontal de un texto. Un texto puede estar alineado a la izquierda o a la derecha, centrado o justificado.

```
<h1>Título 1 (centro)</h1>
<h2>Título 2 (izquierda)</h2>
<h3>Título 3 (derecha)</h3>
<p>Los tres encabezados anteriores están alineados al centro, a la izquierda y a la derecha. Lorem Ipsum es simplemente el texto de relleno de las imprentas y archivos de texto. Lorem Ipsum ha sido el texto de relleno estándar de las industrias desde el año 1500, cuando un impresor desconocido usó una galería de textos y los mezcló de tal manera que logró hacer un libro de textos especimen. </p>

<style>
  h1 {
  text-align: center;
}
h2 {
  text-align: left;
}
h3 {
  text-align: right;
}
p {
  text-align: justify;
}
</style>
```



Lenguaje CSS

Sangría de texto en CSS. La text-indent propiedad en CSS establece la sangría de la primera línea en un bloque de texto. Especifica la cantidad de espacio horizontal que se pone antes de las líneas de texto.

Permite los valores negativos, y si se define algún valor negativo, entonces la sangría de la primera línea será hacia la izquierda.

```
<h1>Uso de sangría de texto</h1>

<p>En mis años más jóvenes y vulnerables, mi padre me dio un consejo que he estado dando vueltas en mi mente desde entonces. "Siempre que tengas ganas de criticar a alguien", me dijo, "recuerda que todas las personas en este mundo no han tenido las ventajas que tú has tenido".</p>

<style>
 p {
 text-indent: 50px;
}
</style>
```

Uso de Float

La propiedad float de CSS es una propiedad de posicionamiento. Se utiliza para empujar un elemento hacia la izquierda o hacia la derecha, permitiendo que otro elemento lo rodee. Generalmente se usa con imágenes y diseños. La propiedad float puede tener uno de los siguientes valores:

- left - El elemento flota a la izquierda de su contenedor
- right - El elemento flota a la derecha de su contenedor
- none: el elemento no flota (se mostrará justo donde aparece en el texto). Esto es por defecto
- inherit: el elemento hereda el valor flotante de su padre

El siguiente ejemplo especifica que una imagen debe flotar hacia la derecha en un texto:

```
<h2>Flotar a la derecha</h2>

<p>En este ejemplo, la imagen flotará a la derecha del párrafo y el texto del párrafo envolverá la imagen.</p>

<p><img src="ADIP_.png" alt="ADIP" style="width:270px;height:170px;margin-left:15px;">
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Phasellus imperdiet, nulla et dictum interdum, nisi lorem egestas odio, vitae scelerisque enim ligula venenatis dolor. Maecenas nisl est, ultrices nec congue eget, auctor vitae massa. Fusce luctus vestibulum augue ut aliquet. Mauris ante ligula, facilisis sed ornare eu, lobortis in odio. Praesent convallis urna a lacus interdum ut hendrerit risus congue. Nunc sagittis dictum nisi, sed ullamcorper ipsum dignissim ac. In at libero sed nunc venenatis imperdiet sed ornare turpis. Donec vitae dui eget tellus gravida venenatis. Integer fringilla congue eros non fermentum. Sed dapibus pulvinar nibh tempor porta. Cras ac leo purus. Mauris quis diam velit.</p>
<style>
 img {
 float: right;
}
</style>
```

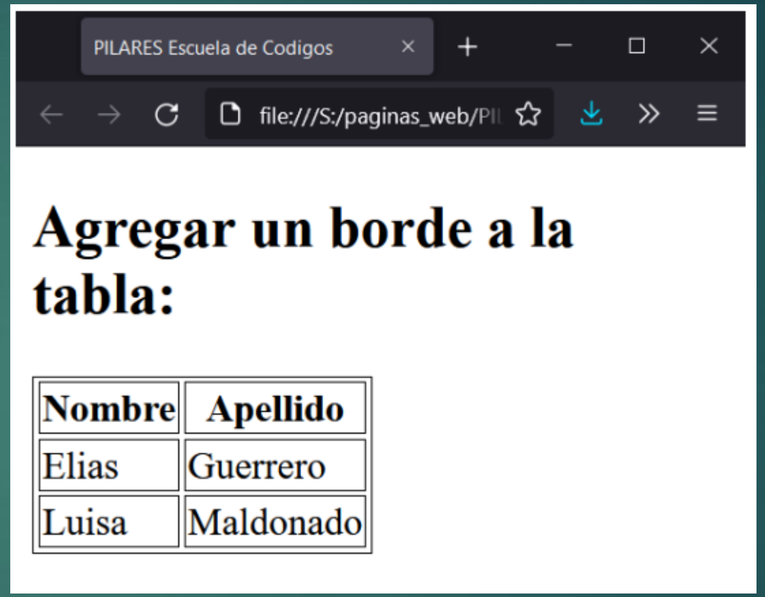
Lenguaje CSS

Estilos en Tablas

Podemos aplicar estilo en tablas HTML para una mejor apariencia. Hay algunas propiedades de CSS que se usan ampliamente en el diseño de tablas usando CSS:

- border
- border-collapse
- padding
- width
- height
- text-align
- color
- background-color

```
<h2>Agregar un borde a la tabla:</h2>

<table>
 <tr>
  <th>Nombre</th>
  <th>Apellido</th>
 </tr>
 <tr>
  <td>Elias</td>
  <td>Guerrero</td>
 </tr>
 <tr>
  <td>Luisa</td>
  <td>Maldonado</td>
 </tr>
</table>

<style>
table, th, td {
  border: 1px solid;
}
</style>
```



Lenguaje CSS

Estilos en Listas

Hay varias propiedades CSS que se pueden usar para controlar las listas. Las listas se pueden clasificar en listas ordenadas y listas desordenadas. En las listas ordenadas, el marcado de los elementos de la lista se realiza con letras y números, mientras que en las listas desordenadas, los elementos de la lista se marcan con viñetas.

Las propiedades CSS para dar estilo a las listas son las siguientes:

- list-style-type: Esta propiedad se encarga de controlar la apariencia y la forma del marcador.
- list-style-image: Establece una imagen para el marcador en lugar del número o una viñeta.
- list-style-position: Especifica la posición del marcador.
- list-style: Es la propiedad abreviada de las propiedades anteriores.

La propiedad de list-style-type. Nos permite cambiar el tipo de marcador de lista predeterminado a cualquier otro tipo, como cuadrado, círculo, números romanos, letras latinas y muchos más. De forma predeterminada, los elementos de la lista ordenada se numeran con números arábigos (1, 2, 3, etc.) y los elementos de una lista desordenada se
marcan con viñetas redondas (•). Si establecemos su valor en none, eliminará los marcadores/viñetas.

La lista también incluye el relleno (padding) y el margen (margin) predeterminados. Para eliminar esto, necesitamos agregar padding:0 y margin:0 a <ol> y <ul>.

```
<ul clase="a">
  <li>Miguel Hidalgo</li>
  <li>Tláhuac</li>
  <li>Coyoacan</li>
</ul>

<ul clase="b">
  <li>Miguel Hidalgo</li>
  <li>Tláhuac</li>
  <li>Coyoacan</li>
</ul>

<p>Ejemplo de listas ordenadas:</p>

<ol clase="c">
  <li>Miguel Hidalgo</li>
  <li>Tláhuac</li>
  <li>Coyoacan</li>
</ol>
<ol clase="d">
  <li>Miguel Hidalgo</li>
  <li>Tláhuac</li>
  <li>Coyoacan</li>
</ol>

<style>
ul.a {list-style-type: circle;}
ul.b {list-style-type: square;}
ol.c {list-style-type: upper-roman;}
ol.d {list-style-type: lower-alpha;}
</style>
```

Lenguaje CSS

La propiedad list-style-image. Especifica una imagen como marcador. Usando esta propiedad, podemos configurar viñetas de imágenes. Su sintaxis es similar a la propiedad background-image. Si no encuentra la imagen correspondiente, se utilizarán las viñetas predeterminadas.

```
<h1>La Propiedad list-style-image</h1>

<p>La propiedad list-style-image remplaza el marcador de elementos de lista por una imagen:</p>

<ul>
  <li>Miguel Hidalgo</li>
  <li>Tláhuac</li>
  <li>Coyoacan</li>
</ul>

<style>
ul {
  list-style-image: url('sqpurple.gif');
}
</style>
```

La Propiedad list-style-image

La propiedad list-style-image remplaza el marcador de elementos de lista por una imagen:

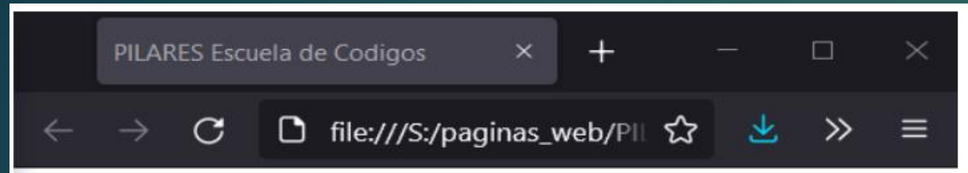
- Miguel Hidalgo
- Tláhuac
- Coyoacan

La propiedad list-style-position. Representa si la aparición del marcador está dentro o fuera del cuadro que contiene las viñetas. Incluye dos valores.

- inside: significa que las viñetas estarán en el elemento de la lista. En esto, si el texto va en la segunda línea, el texto se ajustará debajo del marcador.
- outside: Representa que las viñetas estarán fuera del elemento de la lista. Es el valor predeterminado. El siguiente ejemplo lo explica más claramente.

```
<h1>
Bienvenido a PILARES
</h1>
<h2>
Listas ordenadas
</h2>
<ol class="num">
<li>INSIDE Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod
tempor incididunt ut labore et dolore magna aliqua.</li>
<li>DOS Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod
tempor incididunt ut labore et dolore magna aliqua.</li>
<li>TRES Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod
tempor incididunt ut labore et dolore magna aliqua.</li>
</ol>
<ol class="roman">
<li>OUTSIDE Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod
tempor incididunt ut labore et dolore magna aliqua.</li>
<li>DOS Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod
tempor incididunt ut labore et dolore magna aliqua.</li>
<li>Tres Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod
tempor incididunt ut labore et dolore magna aliqua.</li>
</ol>
<h2>
Listas desordenada
</h2>
<ul class="disc">
<li>INSIDE Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod
tempor incididunt ut labore et dolore magna aliqua.</li>
<li>DOS Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod
tempor incididunt ut labore et dolore magna aliqua.</li>
<li>TRES Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod
tempor incididunt ut labore et dolore magna aliqua.</li>
</ul>
```

```
<ul class="circle">
<li>INSIDE Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna
aliqua.</li><li>DOS Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore
magna aliqua.</li> <li>TRES Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod
tempor incididunt ut labore et dolore magna aliqua.</li></ul>
<ul class="square">
<li>DEFAULT Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed doeiusmod tempor incididunt ut labore et dolore magna
aliqua.</li> <li>DOS Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod
tempor incididunt ut labore et dolore magna aliqua.</li> <li>TRES Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do
eiusmod tempor incididunt ut labore et dolore magna aliqua.</li> </ul>
<style>
.num{
list-style-type:decimal;
list-style-position:inside;
}
.roman{
list-style-type:lower-roman;
list-style-position:outside;
}
.circle{
list-style-type:circle;
list-style-position:inside;
}
.square{
list-style-type:square;
}
.disc{
list-style-type:disc;
list-style-position:inside;
}
</style>
```



Bienvenido a PILARES

Listas ordenadas

1. INSIDE Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.
2. DOS Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.
3. TRES Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

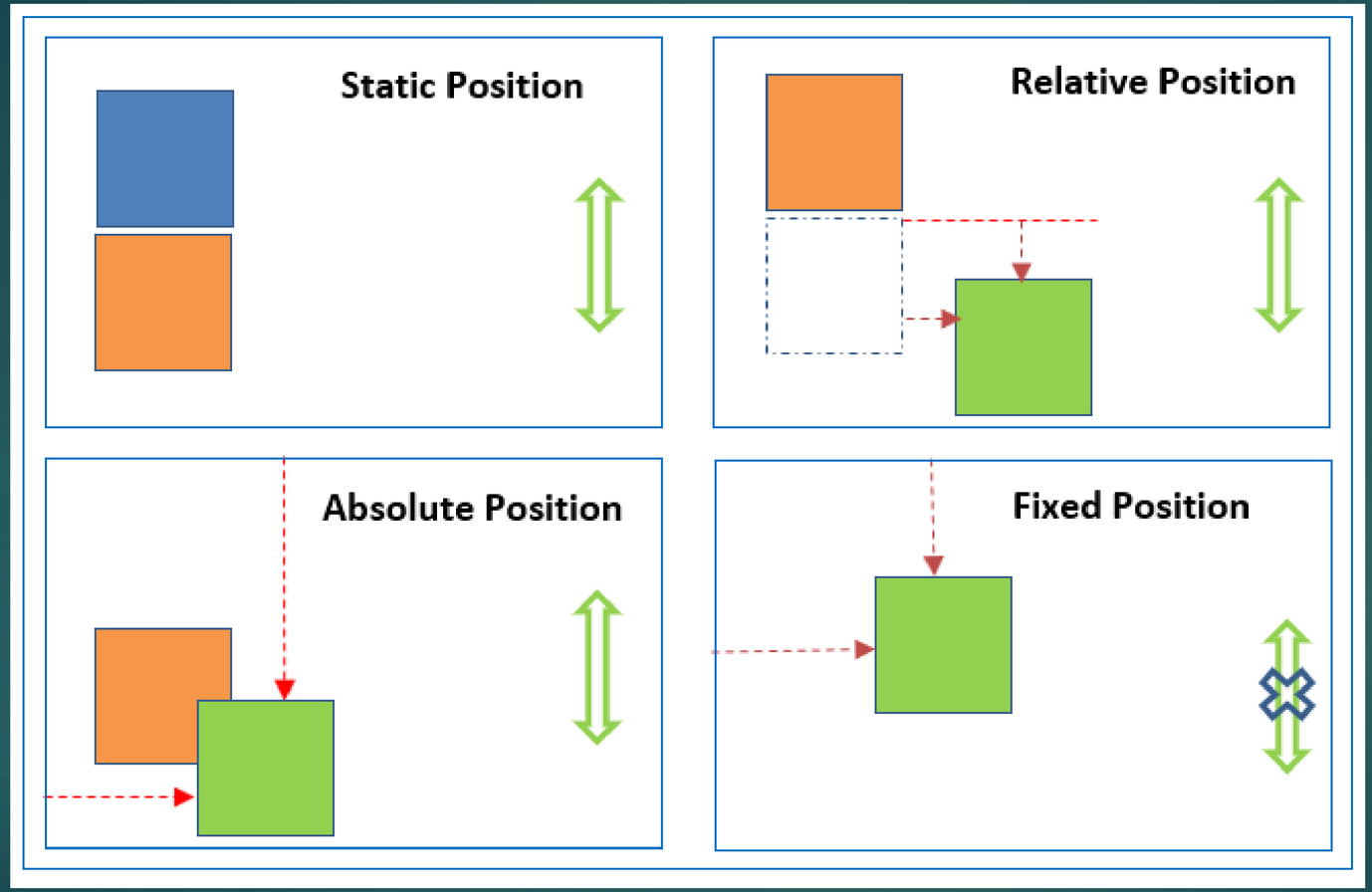
i. OUTSIDE Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.
ii. DOS Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.
iii. Tres Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Listas desordenada

- INSIDE Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.
- DOS Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.
- TRES Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

- INSIDE Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.
- DOS Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.
- TRES Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

- DEFAULT Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.
- DOS Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.
- TRES Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.



En donde **static** es el valor predefinido por **position**.