

CURSO DE PROGRAMACIÓN FULL STACK

HTML & CSS





Objetivos de la Guía

En esta guía aprenderemos a:

- Utilizar un nuevo IDE.
- Comprender qué es HTML
- Conocer las etiquetas básicas de HTML y cómo implementarlas
- Crear una página web.
- Saber dar estilos CSS desde un archivo externo
- Saber usar e implementar Bootstrap en nuestras páginas web

INTRODUCCIÓN

El *World Wide Web* (WWW) es un sistema que contiene una cantidad de información casi infinita. Pero esta información debe estar ordenada de alguna forma de manera que sea posible encontrar lo que se busca. La unidad básica donde está almacenada esta información son las páginas Web. Estas páginas se caracterizan por contener texto, imágenes, animaciones... e incluso sonido y video.

Una de las características más importantes de las páginas Web es que son hipertexto. Esto quiere decir que las páginas no son elementos aislados, sino que están unidas a otras mediante los links o enlaces hipertexto. Gracias a estos enlaces el navegador de internet puede pulsar sobre un texto de una página para navegar hasta otra página. Será cuestión del programador de la página inicial decidir que palabras o frases serán activas y a dónde nos conducirá pulsar sobre ellas.

¿QUÉ ES HTML?

Entendiendo que las páginas web son hipertexto, aquí es donde entra HTML. El **Lenguaje de Marcado de Hipertexto** o *Hyper Text Markup Language* (HTML) es el lenguaje que se utiliza para estructurar y desplegar una página web y sus contenidos respectivos. HTML es el lenguaje con el que se escribe el contenido de las páginas web. Las páginas web pueden ser vistas por el usuario mediante un tipo de aplicación llamada cliente web o más comúnmente "navegador". Podemos decir por lo tanto que el HTML es el lenguaje usado para especificar el contenido que los navegadores deben representar a la hora de mostrar una página web.

Este lenguaje nos permite aglutinar textos, imágenes, enlaces... y combinarlos a nuestro gusto. La ventaja del HTML a la hora de representar el contenido en un navegador, con respecto a otros formatos físicos como libros o revistas, es justamente la posibilidad de colocar referencias a otras páginas, por medio de los enlaces hipertexto

Cuando nos referimos al contenido queremos indicar párrafos, imágenes, listas, tablas y todo aquello que forma parte de "el qué". Nunca debemos usar HTML para definir cómo se debe de ver un contenido, si el texto debe tener color rojo, con una fuente mayor, o si se debe alinear a la derecha. Para especificar el aspecto que debe tener una web se usa un lenguaje complementario, llamado CSS.

HTML LENGUAJE DE MARCADO

HTML no es un lenguaje de programación; es un lenguaje de marcado que define la estructura de tu contenido. Basa su sintaxis en un elemento base al que llamamos marca, tag o simplemente etiqueta. A través de las etiquetas vamos definiendo los elementos del documento, como enlaces, párrafos, imágenes, etc. Así pues, un documento HTML estará constituido por texto y un conjunto de etiquetas para definir la función que juega cada contenido dentro de la página. Todo eso le servirá al navegador para saber cómo se tendrá que presentar el texto y otros elementos en la página.

Existen etiquetas para crear negritas, párrafos, imágenes, tablas, listas, enlaces, etc. Así pues, aprender HTML es básicamente aprenderse una serie de etiquetas, sus funciones, sus usos y saber un poco sobre cómo debe de construirse un documento básico.

Es una tarea muy sencilla de afrontar, al alcance de cualquier persona, puesto que el lenguaje es muy entendible para los seres humanos.

Por ejemplo, toma la siguiente línea de texto:

Mi gato es muy gruñón

Si quieras especificar que se trata de un párrafo, podrías encerrar el texto con la etiqueta de párrafo (<p>):

<p> Mi gato es muy gruñón </p>

¿CÓMO ES LA ANATOMÍA DE UNA ETIQUETA HTML?



Las partes principales de un elemento conformado por una son:

1. **La etiqueta de apertura:** consiste en el nombre de la etiqueta (en este caso, p), encerrado por paréntesis angulares (<>) de apertura y cierre. Establece dónde comienza o empieza a tener efecto la etiqueta, en este caso, dónde es el comienzo del párrafo.
2. **La etiqueta de cierre:** es igual que la etiqueta de apertura, excepto que incluye una barra de cierre (/) antes del nombre de la etiqueta. Establece dónde termina la etiqueta, en este caso dónde termina el párrafo.
3. **El contenido:** este es el contenido de la etiqueta, que en este caso es sólo texto.
4. **El elemento:** la etiqueta de apertura, más la etiqueta de cierre, más el contenido equivale al elemento.

ANIDAMIENTO DE ETIQUETAS

Puedes también colocar etiquetas dentro de otras etiquetas. Esto se llama **anidamiento**. Si, por ejemplo, quieras resaltar una palabra del texto (en el ejemplo la palabra «muy»), podemos encerrarla en una etiqueta ``, que significa que dicha palabra se debe enfatizar:

```
<p> Mi gato es <strong> muy </strong> gruñón </p>
```

Debes asegurarte de que las etiquetas estén correctamente anidadas: en el ejemplo, creaste la etiqueta de apertura del elemento `<p>` primero, luego la del elemento ``, por lo tanto, debes cerrar esta etiqueta primero, y luego la de `<p>`.

Las etiquetas deben abrirse y cerrarse ordenadamente, de forma tal que se encuentren claramente dentro o fuera el uno del otro. Si estos se encuentran solapados, el navegador web tratará de adivinar lo que intentas decirle, pero puede que obtengas resultados inesperados.

ELEMENTOS EN BLOQUE Y EN LÍNEA

El lenguaje HTML clasifica a todos los elementos en dos grupos: **elementos en línea** o *inline elements* y **elementos en bloque** o *block elements*. La diferencia entre ambos viene dada por el modelo de contenido, por el formato y la dirección.

Los elementos en bloque siempre empiezan en una nueva línea y ocupan todo el espacio disponible hasta el final de la línea, mientras que los elementos en línea sólo ocupan el espacio necesario para mostrar sus contenidos sin realizar saltos de línea.

ANATOMÍA DE UN DOCUMENTO HTML

Hasta ahora has visto lo básico de elementos HTML individuales, pero estos no son muy útiles por sí solos. Ahora verás cómo los elementos individuales son combinados para formar una **página HTML entera**.

Los documentos HTML van a ser archivos de texto con la extensión .html y tienen la siguiente anatomía:

```
<!DOCTYPE html>

<html>

<head>
  <meta charset="utf-8">
  <title>Mi página de prueba</title>
</head>

<body>
  <p> Cooperación Humana </p>
</body>

</html>
```

Tienes:

- **<!DOCTYPE html>**: el tipo de documento. Es un preámbulo requerido. Anteriormente, cuando HTML era joven (cerca de 1991/1992), los tipos de documento actuaban como vínculos a un conjunto de reglas que el código HTML de la página debía seguir para ser considerado bueno, lo que podía significar la verificación automática de errores y algunas otras cosas de utilidad. Sin embargo, hoy día es simplemente un artefacto antiguo que a nadie le importa, pero que debe ser incluido para que todo funcione correctamente. Por ahora, eso es todo lo que necesitas saber.
- **<html></html>**: la etiqueta <html>. Esta etiqueta encierra todo el contenido de la página entera y, a veces, se le conoce como la etiqueta raíz (*root element*).
- **<head></head>**: la etiqueta <head>. Esta etiqueta actúa como un contenedor de todo aquello que quieras incluir en la página HTML que no es contenido visible por los visitantes de la página. Incluye cosas como palabras clave (*keywords*), una descripción de la página que quieras que aparezca en resultados de búsquedas, código CSS para dar estilo al contenido, declaraciones del juego de caracteres, etc.
- **<meta charset="utf-8">**: esta etiqueta establece el juego de caracteres que tu documento usará en utf-8, que incluye casi todos los caracteres de todos los idiomas humanos. Básicamente, puede manejar cualquier contenido de texto que puedas incluir. No hay razón para no incluirlo, y puede evitar problemas en el futuro.
- **<title></title>**: la etiqueta <title> establece el título de tu página, que es el título que aparece en la pestaña o en la barra de título del navegador cuando la página es cargada, y se usa para describir la página cuando es añadida a los marcadores o como favorita.
- **<body></body>**: la etiqueta <body>. Encierra todo el contenido que deseas mostrar a los usuarios web que visiten tu página, ya sea texto, imágenes, videos, juegos, pistas de audio reproducibles, y demás. Estos, delimitados a su vez por otras etiquetas como las que hemos visto.

¿QUÉ SON LOS ATRIBUTOS DE LAS ETIQUETAS?

Las etiquetas son la estructura básica del HTML. Estas etiquetas se componen y contienen otras propiedades, como son los atributos y el contenido.

En la actualidad, HTML define un total de 142 etiquetas. Sin embargo, una etiqueta por sí sola a veces no contiene la suficiente información para estar completamente definida. Para ello contamos con los **atributos**. Estos están compuestos de un par nombre-valor que se encuentran separados por "=" y escritos en la etiqueta inicial de un elemento después del nombre del elemento. El valor puede estar encerrado entre "comillas dobles" o 'simples'. Existen, también, algunos atributos que afectan al elemento por su presencia en la etiqueta de inicio.

Esta sería la estructura general de una línea de código en lenguaje HTML:

```
<etiqueta atributo1="valor1" atributo2="valor2">contenido</etiqueta>  
<a href="http://www.enlace.com" target="_blank">Ejemplo de enlace</a>
```

Donde:

- <a> es la etiqueta inicial y la etiqueta de cierre.
- href y target son los atributos.
- http://www.enlace.com y _blank son las variables o valores de los atributos.
- Ejemplo de enlace es el contenido.

Nota: las etiquetas y <a> las veremos en mayor profundidad más adelante.

TIPOS DE ATRIBUTOS

Aunque cada una de las etiquetas HTML define sus propios atributos, encontramos algunos comunes a muchas o casi todas las etiquetas, que se dividen en cuatro grupos según su funcionalidad:

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

En esta guía solo vamos a ver los básicos. Ya que el resto de los atributos son para el uso de otro lenguaje.

Atributos Básicos

Los atributos básicos se utilizan en la mayoría de las etiquetas HTML y XHTML, aunque adquieren mayor sentido cuando se utilizan hojas de estilo en cascada (CSS):

Atributo	Descripción
id="texto"	Establece un indicador único a cada elemento
class="texto"	Establece la clase CSS que se aplica a los estilos del elemento
style="texto"	Aplica de forma directa los estilos CSS de un elemento
title="texto"	Establece el título del elemento (Mejora la accesibilidad)

Nota: los atributos de id, class y style los veremos en mayor profundidad en la parte de CSS.

SINTAXIS HTML

En html existen ciertas reglas de sintaxis que aprenderemos a utilizar y diferenciar, para sacarle mayor provecho al lenguaje y tener más herramientas como desarrolladores.

LAS MAYÚSCULAS O MINÚSCULAS SON INDIFERENTES AL ESCRIBIR ETIQUETAS

En HTML las mayúsculas y minúsculas son indiferentes. Quiere decir que las etiquetas pueden ser escritas con cualquier tipo de combinación de mayúsculas y minúsculas. Resulta sin embargo aconsejable acostumbrarse a escribirlas en minúscula ya que otras tecnologías que pueden convivir con nuestro HTML (XML por ejemplo) no son tan permisivas y nunca viene mal hacernos a las buenas costumbres desde el principio, para evitar fallos triviales en un futuro.

¿CÓMO HACER COMENTARIOS EN HTML?

En un documento HTML, los comentarios se escriben entre los caracteres "`<!--`" y "`-->`". Por ejemplo:

```
<!--Esto es un comentario en HTML-->
```

¿CÓMO HACER UN SALTO DE LÍNEA EN HTML?

Otra de las cosas importantes de conocer sobre la sintaxis básica del HTML es que los saltos de línea no importan a la hora de interpretar una página. Un salto de línea será simplemente interpretado como un separador de palabras, un espacio en blanco. Es por ello que para separar líneas necesitamos usar la etiqueta de párrafo, o la etiqueta BR que significa un salto de línea simple.

Esto es una línea

```
<br>
```

Esto es otra línea

Esto en una página se vería así

Esto es una línea

Esto es otra línea

Nota: La etiqueta BR no tiene su correspondiente cierre. Es un detalle que quizás te haya llamado la atención.

FORMATO DE PARRAFOS HTML

Previamente en nuestra guía habíamos visto la etiqueta `` que nos permitía darle formato a nuestro texto, más concreto ponerlo en negrita. Ahora veremos con más detalle las más ampliamente utilizadas y ejemplificaremos algunas de ellas posteriormente.

Formatear un texto pasa por tareas tan evidentes como definir los párrafos, justificarlos, introducir viñetas, numeraciones o bien poner en negrita, itálica, etc.

Hemos visto que para definir los párrafos nos servimos de la etiqueta P que introduce un salto y deja una línea en blanco antes de continuar con el resto del documento.

Podemos también usar la etiqueta
, de la cual no existe su cierre correspondiente, para realizar un simple salto de línea con lo que no dejamos una línea en blanco, sino que solo cambiamos de línea. Cabe destacar que la etiqueta
, no es la única etiqueta sin cierre.

Podes comprobar que cambiar de línea en nuestro documento HTML sin introducir alguna de estas u otras etiquetas no implica en absoluto un cambio de línea en la página visualizada. En realidad, el navegador introducirá el texto y no cambiará de línea a no ser que esta llegue a su fin o bien lo especifiquemos con la etiqueta correspondiente.

¿CÓMO ALINEAR UN TEXTO?

Los párrafos delimitados por etiquetas P pueden ser fácilmente justificados a la izquierda, centro o derecha especificando dicha justificación en el interior de la etiqueta por medio de un atributo "**align**". Recordemos que los atributos no son más que un parámetro incluido en el interior de la etiqueta que ayudan a definir el funcionamiento de la etiqueta de una forma más personalizada.

Es importante tener muy en cuenta lo siguiente, que ya hemos comentado anteriormente. El HTML se usa para definir el contenido. Por lo tanto, los atributos align que vamos a conocer a continuación se estarán metiendo en una terreno que no le corresponde al HTML, porque están definiendo la forma en la que un párrafo debe de representarse, su estilo, y no el contenido. Es importante señalarlo para aprender que estas cosas se deben hacer mediante el lenguaje CSS, que sirve para definir el estilo, la forma. Usamos este ejemplo también para reforzar el uso de los atributos de una manera más práctica.

Así, si deseásemos introducir un texto alineado a la izquierda escribiríamos:

```
<p align="left">Texto alineado a la izquierda</p>
```

Para una justificación al centro:

```
<p align="center">Texto alineado al centro</p>
```

Para alinear a la derecha:

```
<p align="right">Texto alineado a la derecha</p>
```

Esto en una página se vería así:

Texto alineado a la izquierda

Texto alineado al centro

Texto alineado a la derecha

Como veis, en cada caso el atributo align toma determinados valores que son escritos entre comillas. En algunas ocasiones necesitamos especificar algunos atributos para el correcto funcionamiento de la etiqueta. En otros casos, el propio navegador toma un valor definido por defecto. Para el caso de align, el valor por defecto es left.

FORMATO DE LETRA

Además de todo lo relativo a la organización de los párrafos, uno de los aspectos primordiales del formato de un texto es el de la propia letra. Resulta muy común y práctico presentar texto resaltado en negrita, itálica y otros. Todo esto y mucho más es posible por medio del HTML a partir de multitud de etiquetas entre las cuales vamos a destacar algunas.

Pero antes de comenzar cabe hacer una reflexión sobre por qué son interesantes estas etiquetas y se siguen usando, a pesar de que están entrando prácticamente en el terreno de CSS, ya que en la práctica están directamente formateando el aspecto de las fuentes. Son importantes porque las etiquetas en si no están para definir un estilo en concreto, sino una función de ciertas palabras dentro de un contenido.

NEGRITA

Podemos escribir texto en negrita incluyéndolo dentro de las etiquetas **strong** y su cierre. Recordemos que ya la habíamos visto previamente.

 Texto en negrita y texto normal</p>

Esto en una página se vería así:

Texto en negrita y texto normal

ITÁLICA

En este caso existen dos posibilidades, una corta: *i* y su cierre (*italic*) y otra un poco más larga: **EM** y su cierre. En esta guía vamos a usar, y en la mayoría de las páginas que veréis por ahí, os encontraréis con la primera forma sin duda más sencilla a escribir y a acordarse.

<i> Texto en itálica </i> y texto normal</p>

Esto en una página se vería así:

Texto en italica y texto normal

SUBRAYADO

El HTML nos propone también para el subrayado la etiqueta: **U** (underlined). Sin embargo, el uso de subrayados ha de ser aplicado con mucha precaución dado que los enlaces hipertexto van, a no ser que se indique lo contrario, subrayados con lo que podemos confundir al lector y apartarlo del verdadero interés de nuestro texto.

Además, cabe decir que la etiqueta U se ha quedado obsoleta, debido a que es algo que realmente se debe hacer del lado del CSS, al ser básicamente un estilo.

<u> Texto subrayado</u> y texto normal</p>

Esto en una página se vería así:

Texto subrayado y texto normal

ENCABEZADOS

Existen otras etiquetas para definir párrafos especiales, que funcionaran como títulos de nuestra página. Son los encabezados o headings en inglés. Como decimos, son etiquetas que formatean el texto como un título, pero el hecho de que cambien el formato no es lo que nos tiene que preocupar, sino el significado en sí de la etiqueta. Es cierto que los navegadores asignan un tamaño mayor de letra y colocan el texto en negrita, pero lo importante es que sirven para definir la estructura del contenido de un documento HTML. Así los navegadores para ciegos podrán informar a los invidentes que esta es una división nueva de contenido y que su título es este o aquél. También los motores de búsqueda sabrán interpretar mejor el contenido de una página en función de los títulos y subtítulos.

Hay varios tipos de encabezados, que se diferencian visualmente en el tamaño de la letra que utilizan. La etiqueta en concreto es la H1, para los encabezados más grandes, H2 para los de segundo nivel y así hasta H6 que es el encabezado más pequeño. Pero lo importante, insistimos es la estructura que denotan. Una página tendrá generalmente un encabezado de nivel 1 y dentro varios de nivel 2.

Luego, dentro de los H2 encontraremos si acaso H3, etc. Nunca debemos usar los encabezados porque nos formatee el texto de una manera dada, sino porque nuestro documento lo requiera según su estructura.

Los encabezados se verán de esta manera en la página:

Encabezado de nivel 1

Encabezado de nivel 2

Encabezado de nivel 3

Encabezado de nivel 4

Encabezado de nivel 5

Encabezado de nivel 6

Los encabezados implican también una separación en párrafos, así que todo lo que escribamos dentro de H1 y su cierre (o cualquier otro encabezado) se colocará en un párrafo independiente.

Podemos ver cómo se presentan algunos encabezados a continuación.

<h1> Encabezado de nivel 1 </h1>

Los encabezados, como otras etiquetas de HTML, soportan el atributo align. Veremos un ejemplo de encabezado de nivel 2 alineado al centro, aunque repetimos que esto debería hacerse en CSS.

<h2 align="center"> Encabezado de nivel 2 </h2>

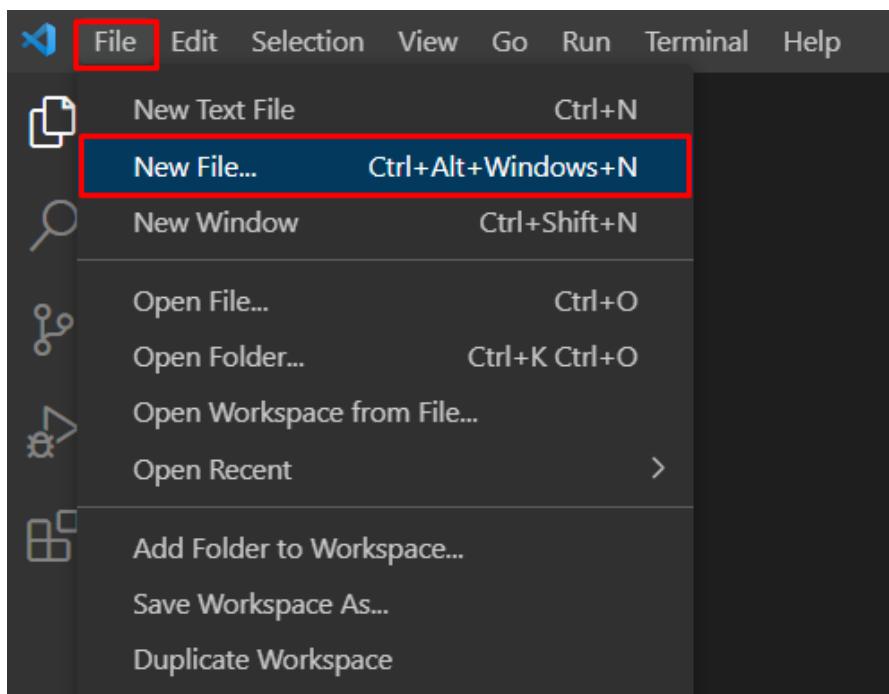


MANOS A LA OBRA!

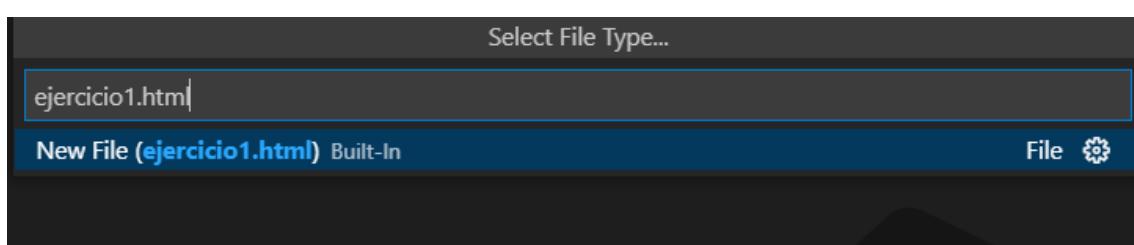
Actividad: Presentación Personal – Parte1

Ya conocemos qué son las etiquetas y cómo se conforma un documento HTML. También conocimos las etiquetas para realizar párrafos y encabezados. Pongamos esto en práctica con la siguiente ejercitación:

- 1) **Crear un archivo HTML.** Para realizar esto, abriremos el editor Visual Studio Code (el IDE que utilizaremos en este módulo) y localizaremos la barra de navegación que se haya en la parte superior del programa. Iremos al apartado “File” y dentro de las opciones seleccionaremos “New File”. Como se puede observar en la siguiente imagen:

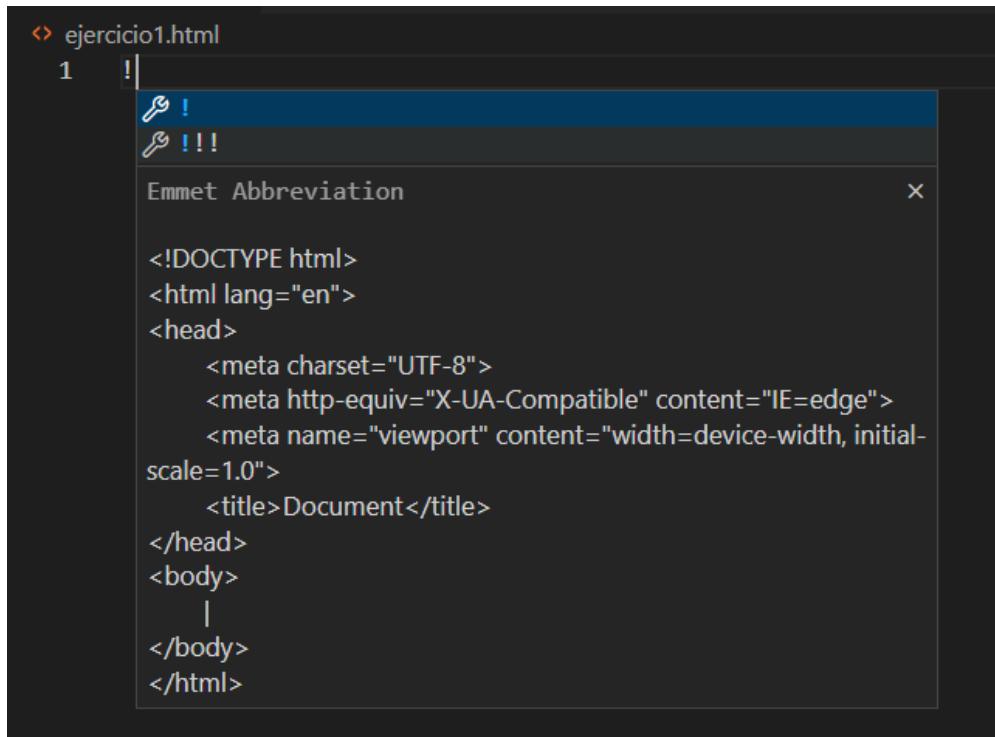


A continuación, se nos desplegará una ventana donde podremos escribir el nombre de nuestro archivo para especificar el tipo del mismo y posteriormente, guardarlo en nuestro ordenador.



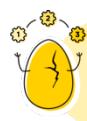
Nota: Hay muchas formas de crear archivos HTML, esta es una. Lo importante para crear un documento HTML en Visual Studio Code es nombrar el mismo con la extensión “.html” al final. De esta manera, el editor reconoce que se trata de un documento html.

- 2) **Crear estructura principal de un documento HTML.** Utilizando un shortcut (atajo de teclado) vamos a crear la estructura principal de nuestro documento. El shortcut es un signo de exclamación “!” y seleccionaremos la primera opción como se puede ver en la siguiente imagen.



The screenshot shows a code editor window with the file name "ejercicio1.html". In the top left, there is a placeholder "1" followed by a cursor. A dropdown menu titled "Emmet Abbreviation" is open, showing two options: "!" and "!!!". Below the menu, the HTML code for a basic document structure is displayed:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Document</title>
</head>
<body>
    |
</body>
</html>
```



Estos pasos son los que deberás hacer para la realización de cada ejercicio, ya que cada uno, se corresponde con un nuevo archivo HTML. Asegúrate de practicarlo y comprenderlo. Si tienes dudas: ¡consulta con tus compañeros de mesa!

- 3) **Agregar etiquetas de texto.** Ahora agrega al documento (dentro de la etiqueta <body>) los siguientes elementos:
- Un encabezado <h1> con tu nombre.
 - Un párrafo con una breve presentación.
 - Resalta con negrita lo que consideres más importante del párrafo.
 - Utiliza saltos de línea si los consideras necesarios.
- 4) **Renderizar el documento HTML.** Renderizar significa desplegar nuestro documento en la web. Para desplegar el documento que acabamos de crear haremos lo siguiente:
- a) **Guardar el documento.** Puedes usar el shortcut **ctrl + s** o buscar la opción “Save” en las opciones de “File” (primera opción de la barra de tareas).
 - b) **Buscar el documento en tu ordenador.** Busca el documento y ábrelo. Se abrirá en tu navegador web predeterminado.



Hasta el momento has aprendido

- Crear un archivo HTML.
- Crear la estructura principal de un documento HTML.
- Utilizar etiquetas.
- Renderizar un documento HTML.

LISTAS EN HTML

Las posibilidades que nos ofrece el HTML en cuestión de tratamiento de texto son realmente notables. No se limitan a lo visto hasta ahora, sino que van más lejos todavía. Varios ejemplos de ello son las listas, que sirven para enumerar y definir elementos.

Las listas originalmente están pensadas para citar, enumerar y definir cosas a través de características, o al menos así lo hacemos en la escritura de textos. Sin embargo, las listas finalmente se utilizan para mucho más que enumerar una serie de puntos, en realidad son un recurso muy interesante para poder maquetar elementos diversos, como barras de navegación, pestañas etc.

Por ahora, trataremos las listas desde el punto de vista de su construcción y veremos los diferentes tipos que existen, y que podemos utilizar para resolver nuestras distintas necesidades a la hora de escribir textos en HTML.

Podemos distinguir dos tipos de listas HTML:

- Listas desordenadas
- Listas ordenadas

LISTAS DESORDENADAS

Son delimitadas por las etiquetas **UL** y su cierre (*unordered list*). Cada uno de los elementos de la lista es citado por medio de una etiqueta **LI** (*list item*).

Nota: La etiqueta LI tiene su respectiva etiqueta de cierre, aunque si no lo colocas, el navegador al ver el siguiente LI interpretará que estás cerrando el anterior.

```
<p>Países del mundo</p>
```

```
<ul>
```

```
  <li>Argentina</li>
  <li>Perú</li>
  <li>Chile</li>
```

```
</ul>
```

Esto renderizado (desplegado en la web) se vería así:

Países del mundo

- Argentina
- Perú
- Chile

Tipos de viñetas

Podemos definir el tipo de viñeta empleada para cada elemento. Para ello debemos especificarlo por medio del **atributo type**. Si queremos que el estilo sea válido para toda la lista lo incluiremos dentro de la etiqueta de apertura UL, mientras que si queremos hacerlo específico a un solo elemento lo incluiremos dentro de la etiqueta LI.

La sintaxis es del siguiente tipo:

```
<ul type="tipo de viñeta">
```

Donde tipo de viñeta puede ser uno de los siguientes:

- circle
- disc
- square



```
<ul type="square">  
  <li>Elemento 1</li>  
  <li>Elemento 2 </li>  
  <li>Elemento 3 </li>  
  <li type="circle">Elemento 4 </li>  
</ul>
```

Esto en una página web se vería así:

- Elemento 1
- Elemento 2
- Elemento 3
- Elemento 4

En este ejemplo nos encontramos con una lista desordenada donde uno de los elementos tiene una viñeta cuadrada en lugar de un redondel como se especifica para el resto de los elementos en la etiqueta .

LISTAS ORDENADAS

Las listas ordenadas sirven también para presentar información en diversos elementos o items, con la particularidad que éstos estarán precedidos de un número o una letra para enumerarlos manteniendo un orden.

Para implementar listas ordenadas usaremos la etiqueta OL (*ordered list*) y su cierre. Cada elemento será igualmente indicado por la etiqueta LI(*list item*), que ya vimos en las listas desordenadas.

```
<p> Reglas de convivencia</p>
```

```
<ol>
```

```
 <li> Respeta a tu compañero y escucha su opinión </li>
```

```
 <li> Trata amablemente a tu entorno </li>
```

```
</ol>
```

Esto en una página se vería así:

Reglas de comportamiento en el trabajo

1. El jefe siempre tiene la razón
2. En caso de duda aplicar regla 1

Tipo de numeración.

Del mismo modo que para las listas desordenadas, las listas ordenadas ofrecen la posibilidad de modificar el estilo. En concreto nos es posible especificar el tipo de numeración empleado eligiendo entre números (1, 2, 3, ...), letras (a, b, c, ...) y sus mayúsculas (A, B, C, ...) y números romanos en sus versiones mayúsculas (I, II, III, ...) y minúsculas (i, ii, iii, ...).

Para realizar dicha selección hemos de utilizar, como para el caso precedente, el **atributo type**, el cual será situado dentro de la etiqueta OL.

Los valores que puede tomar el atributo en este caso son:

- 1 Para ordenar por números
- a Por letras del alfabeto
- A Por letras mayúsculas del alfabeto
- i Ordenación por números romanos en minúsculas
- I Ordenación por números romanos en mayúsculas

Puede que en algún caso deseemos comenzar nuestra enumeración por un número o letra que no tiene por qué ser necesariamente el primero de todos. Para solventar esta situación, podemos utilizar un segundo atributo, **start**, que tendrá como valor un número. Este número, que por defecto es 1, corresponde al valor a partir del cual comenzamos a definir nuestra lista. Para el caso de las letras o los números romanos, el navegador se encarga de hacer la traducción del número a la letra correspondiente.



¿NECESITAS UN EJEMPLO?

<p> Ordenamos por números </p>

<ol type="1">

 Elemento 1

 Elemento 2

<p> Ordenamos por letras </p>

<ol type="a">

 Elemento a

 Elemento b

<p> Ordenamos por números romanos empezando por el 10</p>

<ol type="i" start="10">

 Elemento x

 Elemento xi

Esto en una página se vería así:

Ordenamos por números

1. Elemento 1
2. Elemento 2

Ordenamos por letras

- a. Elemento a
- b. Elemento b

Ordenamos por números romanos empezando por el 10

- x. Elemento x
- xi. Elemento xi

ANIDANDO LISTAS

Nada nos impide utilizar todas estas etiquetas de forma anidada como hemos visto en otros casos. De esta forma, podemos conseguir listas mixtas.

```
<p>Ciudades del mundo</p>
```

```
<ul>
```

```
    <li>Argentina </li>
```

```
    <ol>
```

```
        <li>Buenos Aires </li>
```

```
        <li>Bariloche </li>
```

```
    </ol>
```

```
    <li>Uruguay </li>
```

```
    <ol>
```

```
        <li>Montevideo </li>
```

```
        <li>Punta del Este </li>
```

```
    </ol>
```

```
</ul>
```

Esto en una página se vería así:

Ciudades del mundo

- Argentina
 - 1. Buenos Aires
 - 2. Bariloche
- Uruguay
 - 1. Montevideo
 - 2. Punta del Este



MANOS A LA OBRA!

Actividad: Presentación Personal – Parte 2

¡Pongamos en práctica lo aprendido sobre listas!

Para ello emplearemos la actividad realizada anteriormente llamada “Presentación Personal – Parte 1”.

- 1) Agrega un encabezado de menor relevancia que el usado en la actividad anterior (Presentación Personal – Parte 1) y nómbralo “Aptitudes Personales”.
- 2) **Implementar listas desordenadas.** Luego crea una lista y agrega tus principales aptitudes (por ejemplo: asertividad, trabajo en equipo, escucha activa, entre otros).
- 3) Debajo agrega un nuevo encabezado (igual relevancia que en el punto 1) nombrado “Tecnologías”
- 4) **Implementar listas ordenadas.** Debajo del encabezado enumera las tecnologías que conoces y has aprendido en este curso. (Java, MySQL, HTML entre otras).



Hasta el momento has aprendido

- Implementar etiquetas anidadas
- Utilizar listas ordenadas.
- Utilizar listas desordenadas.

ENLACES EN HTML

Hasta aquí, hemos podido ver que una página web es un archivo HTML en el que podemos incluir, entre otras cosas, textos formateados a nuestro gusto e imágenes (las veremos con detalle enseguida). Del mismo modo, un sitio web podrá ser considerado como el conjunto de archivos, principalmente páginas HTML e imágenes, que constituyen el contenido al que el navegante tiene acceso.

Sin embargo, no podríamos hablar de navegación si estos archivos HTML no estuviesen debidamente conectados entre ellos y con el exterior de nuestro sitio por medio de enlaces hipertexto. En efecto, el atractivo original del HTML radica en la posible puesta en relación de los contenidos de los archivos introduciendo referencias bajo forma de enlaces que permitan un acceso rápido a la información deseada. De poco serviría en la red tener páginas aisladas a las que la gente no puede acceder y desde las que la gente no puede saltar a otras.

Un enlace puede ser fácilmente detectado por el usuario en una página. Basta con deslizar el puntero del ratón sobre las imágenes o el texto y ver cómo cambia su forma original transformándose por regla general en una mano con un dedo señalador.

Adicionalmente, estos enlaces suelen ir, en el caso de los textos, coloreados y subrayados para que el usuario no tenga dificultad en reconocerlos.

SINTAXIS DE UN ENLACE

Para colocar un enlace, nos serviremos de las etiquetas `<a>` y su cierre. Dentro de la etiqueta de apertura deberemos especificar asimismo el destino del enlace. Este destino será introducido bajo forma de **atributo**, el cual lleva por nombre "`href`".

La sintaxis general de un enlace es por tanto de la forma:

```
<a href="destino">contenido</a>
```

Siendo el "contenido" un texto o una imagen. Es la parte de la página que se colocará activa y donde deberemos pulsar para acceder al enlace. Por su parte, "destino" será una página, un correo electrónico o un archivo.



Veamos un ejemplo con un enlace al home de EggEducación:

```
<a href="https://eggeducacion.com/es-AR/">Home de EggEducación</a>
```

Renderizado se vería de la siguiente manera:

Home de EggEducación

Enlaces con imágenes

Ahora, si queremos que el contenido del enlace sea una imagen y no un texto, deberemos colocar la correspondiente etiqueta `` dentro de la etiqueta `a`.

```
<a href="https://eggeducacion.com/es-AR/"></a>
```

Nota: veremos la etiqueta de imágenes más adelante.

EL ASPECTO DE LOS ENLACES

Utilizando HTML, y las hojas de estilo CSS, podremos definir el aspecto que tendrán los enlaces de una página. Sin embargo, ya de manera predeterminada el navegador los destaca para que los podamos distinguir. Generalmente encontraremos a los enlaces subrayados y coloreados en azul, aunque esta regla depende del navegador del usuario y de sus estilos definidos como predeterminados.

TIPOS DE ENLACES

Para estudiar en profundidad los enlaces tenemos que clasificarlos por su tipo, ya que, dependiendo del mismo, algunas características pueden cambiar.

En función del destino los enlaces son clásicamente agrupados del siguiente modo:

- **Enlaces locales:** los que se dirigen a otras páginas del mismo sitio web.
- **Enlaces remotos:** los dirigidos hacia páginas de otros sitios web. Estos son los que vimos en el ejemplo anterior.

ENLACES LOCALES

Como hemos dicho, un sitio web está constituido de páginas interconectadas, que se relacionan mediante enlaces de hipertexto. Para cumplir con esto es que vamos a utilizar los enlaces locales.

Los enlaces locales se tratan de un tipo de enlace mucho más común en el día a día del desarrollo. De hecho, es el tipo de enlace que más se produce en general. Estos enlaces locales nos permiten relacionar distintos documentos HTML que componen un sitio web. Gracias a los enlaces locales podremos convertir varias páginas sueltas en un sitio web completo, compuesto de varios documentos.

Para crear este tipo de enlaces, hemos de usar la misma etiqueta `<a>` que ya conocemos, de la siguiente forma:

```
<a href="archivo.html">contenido</a>
```

RUTAS DE LOS ENLACES

Hacer un enlace en si no es para nada complejo. No requiere muchas explicaciones, con lo que ya hemos visto en la guía alcanza. Sin embargo, hay que abordar con detalle un tema importante: las **rutas de los enlaces**.

Como rutas nos referimos al destino del enlace, o sea, lo que ponemos en el atributo "href" y es importante que nos paremos aquí porque nos puede dar algunos problemas al desarrollar. Principalmente para aquellas personas que tengan menos experiencia en el trabajo con el ordenador.

Por regla general, para una mejor organización, los sitios suelen estar ordenados por directorios. Estos directorios suelen contener diferentes secciones de la página como imágenes, scripts, estilos, etc. Es por ello que en muchos casos no nos valdrá con especificar el nombre del archivo, sino que tendremos que especificar además el directorio en el que nuestro archivo.html está alojado.

Para aquellos que no saben cómo mostrar el camino de un archivo, aquí van una serie de indicaciones que los ayudaran a comprender la forma de expresarlos.

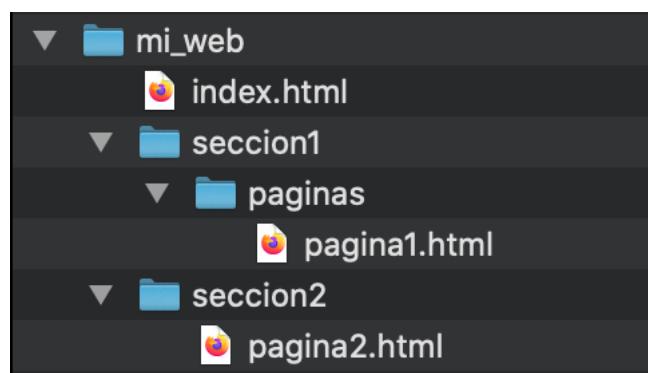
- 1) Hay que situarse mentalmente en el directorio en el que se encuentra la página donde vamos a crear el enlace.
- 2) Si la página destino está en el **mismo directorio** que el archivo desde donde vamos a enlazar podemos colocar simplemente el **nombre del archivo de destino**, ya que no hay necesidad de cambiar de directorio.
- 3) Si la página de destino está en una carpeta o subdirectorío **interior al directorio** donde está el archivo de origen, hemos de marcar la ruta **enumerando cada uno de los directorios** por los que pasamos hasta llegar al archivo de destino. Separándolos por el símbolo barra "/". Al final obviamente, escribimos el **nombre del archivo destino**.

- 4) Si la página destino se encuentra en un **directorio padre** (superior al de la página del enlace), hemos de escribir dos puntos y una barra "../" tantas veces como niveles subamos en la arborescencia hasta dar con el directorio donde está emplazado el archivo destino. Finalmente, también agregaremos el **nombre del archivo destino**
- 5) Si la página se encuentra en otro directorio no incluido ni incluyente del archivo origen, tendremos que subir como en la regla 3 por medio de ".." hasta encontrar un directorio que englobe el directorio que contiene a la página destino. A continuación, haremos como en la regla 2. Escribiremos todos los directorios por los que pasamos hasta llegar al archivo.

Nota: puede parecer complicado en la lectura. Pero te aseguramos que cuando empieces a practicarlo vas a comprenderlo mejor. Recuerda que siempre que tengas alguna duda puedes volver a este material y releer la información.



Imagina que tienes la siguiente estructura de carpetas y archivos. La que aparece en la siguiente imagen.



- 1) Para hacer un enlace a index.html desde index.html

```
<a href="index.html">Ir a index.html</a>
```

- 2) Para hacer un enlace desde index.html hacia pagina1.html:

```
<a href="seccion1/paginas/pagina1.html">Ir a pagina1.html</a>
```

- 3) Para hacer un enlace desde pagina2.html hacia pagina1.html:

```
<a href="../seccion1/paginas/pagina1.html">Ir (también) a pagina1.html</a>
```

- 4) Para hacer un enlace desde pagina1.html hacia pagina2.html:

```
<a href=".../seccion2/pagina2.html">Ir ahora a pagina2.html</a>
```



MANOS A LA OBRA!

Actividad: Presentación Personal - Enlaces

¡Hemos incorporado un nuevo conocimiento de HTML! ¿Lo practicamos?

Seguimos con la actividad Presentación Personal. En este caso agregaremos 2 enlaces a nuestra página.

- 1) **Agregar un enlace externo sobre un texto.** Agrega un enlace externo con dirección a tu LinkedIn personal (si no tienes puede ser al home page de LinkedIn) bajo el texto “LinkedIn Personal”.

IMÁGENES EN HTML

Sin duda uno de los aspectos más vistosos y atractivos de las páginas web es el **grafismo**. La incorporación de imágenes en nuestros textos puede ayudarnos a explicar más fácilmente los objetivos y organizar la información al tiempo que se le da un aire más estético a nuestra web. No obstante, su uso en exceso puede conducirnos a una sobrecarga que se traduce en una distracción para el navegante, quien tendrá más dificultad en encontrar la información necesaria.

El uso de imágenes también tiene que ser realizado con cuidado porque aumentan el tiempo de carga de la página, lo que puede ser un efecto nefasto si nuestro visitante no tiene una buena conexión o si es un poco impaciente. Por ello es recomendable siempre optimizar las imágenes para Internet, haciendo que su tamaño en bytes sea lo mínimo posible, para facilitar la descarga, pero sin que ello comprometa mucho su calidad.

En esta guía no explicaremos como crear ni tratar las imágenes, únicamente diremos que para ello se utilizan aplicaciones como Paint Shop Pro, Photoshop o Gimp. Tampoco explicaremos las particularidades de cada tipo de archivo: GIF, JPG o PNG y la forma de optimizar nuestras imágenes.

La etiqueta que utilizaremos para insertar una imagen es **** (*image*). Esta etiqueta no posee su cierre correspondiente y en ella hemos de especificar obligatoriamente el paradero de nuestro archivo gráfico mediante el atributo **src** (*source*).

La sintaxis queda entonces de la siguiente forma:

```

```

Para expresar el camino, lo haremos de la misma forma que vimos para los enlaces. Las reglas siguen siendo las mismas, lo único que cambia es que, en lugar de una página siendo el destino, el destino es un archivo gráfico. En el código anterior estamos enlazando un archivo con extensión .jpg, pero podrá ser otro tipo de archivo como .gif o .png.

A parte de este atributo, indispensable para la visualización de la imagen, la etiqueta IMG nos propone otra serie de atributos de mayor o menor utilidad, que listamos a continuación:

ATRIBUTO ALT

Dentro de las comillas de este atributo colocaremos una breve descripción de la imagen. Esta etiqueta no es indispensable, pero presenta varias utilidades. La sintaxis quedaría de esta manera:

```

```

Primeramente, sirve para el **posicionamiento de la página en buscadores**. De los atributos alt el buscador puede extraer palabras clave y le ayuda a entender qué función o contenido tiene la imagen, y por lo tanto la página en sí.

Otra utilidad importante la encontramos en determinadas aplicaciones, usadas por personas con discapacidad. Navegadores para gente con discapacidad visual, por ejemplo, no muestran las imágenes y por tanto los alt ofrecen la posibilidad de **leerlas**. Nunca está de más pensar en crear páginas **accesibles**.

Por último, durante el proceso de carga de la página y cuando la imagen no ha sido todavía cargada, el navegador podría mostrar esta descripción, con lo que el navegante se puede hacer una idea de lo que va en ese lugar. Si hubo problemas de conexión y no se pudo mostrar la imagen, también podría usarse ese alt para mostrar al menos su descripción.

En general podemos considerar como aconsejable el uso de este atributo, salvo para imágenes de poca importancia. Si la imagen es usada como cuerpo de un enlace todavía se hace más indispensable.

IMÁGENES QUE SON ENLACES Y EL ATRIBUTO BORDER

Si un texto puede servir de enlace, una imagen puede cumplir la misma función:

```
<a href="archivo.html"></a>
```

El problema de hacer esto en ciertos navegadores es que se crea un borde en la imagen, del mismo color que el color configurado para los enlaces, lo que suele ser un efecto poco deseado.

Sin embargo, en HTML podemos indicar que una imagen tenga borde. Mediante el atributo "border" se define el tamaño en píxeles del cuadro que rodea la imagen. De esta forma podemos recuadrar nuestra imagen si lo deseamos. No es algo que se use mucho, pero resulta particularmente útil cuando deseamos eliminar el borde que aparece cuando la imagen sirve de enlace. En dicho caso tendremos que especificar border="0".



MANOS A LA OBRA!

Actividad Presentación Personal - Imágenes.

Ahora que sabemos utilizar la etiqueta `` y sus atributos básicos, lo incluyamos en nuestra página.

- 1) **Agregar una imagen.** Debajo del encabezado con nuestro nombre, pondremos una imagen nuestra que tengamos en nuestro ordenador.
- 2) **Agregar un enlace externo sobre una imagen.** Al final de la página agrega una imagen de un personaje de películas o series que te guste con un enlace a un recurso que lo represente puede ser: un video de YouTube, alguna web, algún recurso de Netflix ¡lo que prefieras!

Nota: Recuerda que la intención de estas actividades es reforzar lo aprendido: no deberíamos dedicar tanto tiempo eligiendo un personaje o un recurso en específico.



Revisemos lo aprendido hasta aquí

- Crear enlaces internos y externos.
- Crear enlaces sobre textos.
- Crear enlaces sobre imágenes.
- Agregar imágenes a un documento HTML.

TABLAS EN HTML

Una tabla es un conjunto de celdas organizadas dentro de las cuales podemos alojar distintos contenidos. HTML dispone de una gran variedad de etiquetas para crear tablas, con sus atributos.

Como veremos a continuación, existen diversas etiquetas que se deben utilizar en una forma determinada para la creación de tablas. Por ello, puede que en un principio nos resulte un poco complicado trabajar con estas estructuras, pero con un poco de práctica podremos crear tablas con absoluta soltura.

Si deseamos mostrar datos de una manera sencilla de leer, dispuestos en filas y columnas, tarde o temprano observaremos que las tablas son la mejor solución y apreciaremos las posibilidades nos ofrecen.

ETIQUETAS BÁSICAS PARA TABLAS EN HTML

Para empezar, las tablas son definidas por las etiquetas `<table>` y su cierre.

Dentro de estas dos etiquetas colocaremos todas las otras etiquetas de las tablas, hasta llegar a las celdas. Dentro de las celdas ya es permitido colocar textos e imágenes que darán el contenido a la tabla.

Las tablas son descritas por líneas de arriba a abajo (y luego por filas de izquierda a derecha). Cada una de estas líneas, llamadas columnas, es definida por otra etiqueta y su cierre: `<tr>` (*table row: columna de tabla*).

Asimismo, dentro de cada línea, habrá diferentes celdas. Cada una de estas celdas será definida por otra etiqueta: `<td>` (*table data*). Dentro de ésta y su cierre será donde coloquemos nuestro contenido, el contenido de cada celda.



¿NECESITAS UN EJEMPLO?

```
<table>
<tr>
  <td> Celda 1, línea 1 </td>
  <td> Celda 2, línea 1 </td>
</tr>
<tr>
  <td> Celda 1, línea 2 </td>
  <td> Celda 2, línea 2 </td>
</tr>
</table>
```

Esto en una página se vería así:

Celda 1, linea 1 Celda 2, linea 1
Celda 1, linea 2 Celda 2, linea 2

También existe la etiqueta **<th>** (*table header*), que sirve para crear una celda cuyo contenido esté formateado como un título o cabecera de la tabla. En la práctica, lo que hace es poner en negrita y centrado el contenido de esa celda.



¿NECESITAS UN EJEMPLO?

```
<table>
<tr>
  <th>Titulo Celda 1</th>
  <th> Titulo Celda 2</th>
</tr>
<tr>
  <td>Celda 1, línea 1</td>
  <td> Celda 2, línea 1</td>
</tr>
<tr>
  <td> Celda 1, línea 2</td>
  <td> Celda 2, línea 2</td>
</tr>
</table>
```

Esto en una página se vería así:

Titulo Celda 1 Titulo Celda 2

Celda 1, linea 1 Celda 2, linea 1

Celda 1, linea 2 Celda 2, linea 2



Revisemos lo aprendido hasta aquí

- Crear tablas en HTML

FORMULARIOS HTML

Hasta ahora hemos visto la forma en la que el HTML gestiona y muestra la información, esencialmente mediante texto, imágenes y enlaces. Nos queda por ver de qué forma podemos intercambiar información con nuestro visitante. Desde luego, este nuevo aspecto resulta primordial para gran cantidad de acciones que se pueden llevar a cabo mediante la Web: comprar un artículo, llenar una encuesta, enviar un comentario al autor, registrar un usuario, etc.

Los formularios son esas famosas cajas de texto y botones que podemos encontrar en muchas páginas web. Son muy utilizados para realizar búsquedas o bien para introducir datos personales por ejemplo en sitios de comercio electrónico. Los datos que el usuario introduce en estos campos son enviados al correo electrónico del administrador del formulario o bien a un programa que se encarga de procesarlo automáticamente. Nosotros en esta guía no vamos a mostrar como enviar la información al mail, ya que nos interesa, más adelante poder manejar esa información.

¿QUÉ SE PUEDE HACER CON UN FORMULARIO?

Usando HTML podemos únicamente enviar el contenido del formulario a un correo electrónico, es decir, construir un formulario con diversos campos y, a la hora pulsar el botón de enviar, generar un mensaje de que se ha registrado con éxito la información.

Pero para todo lo que sea manejar esa información y guardarla, por ejemplo, en una base de datos vamos a tener que utilizar Java. Como lo veremos más adelante en el curso.

Así pues, en resumen, con HTML podremos construir los formularios, con diversos tipos de campos, como cajas de texto, botones de radio, cajas de selección, menús desplegables, etc. Sin embargo, debe quedar claro que desde HTML no se puede manejar esta información para guardarla o enviarla a algún correo, etc. Eso será trabajo de Java.

¿CÓMO HACER UN FORMULARIO EN HTML?

Los formularios son definidos por medio de las etiquetas `<form>` y su cierre. Entre estas dos etiquetas colocaremos todos los campos y botones que componen el formulario. Dentro de esta etiqueta `<form>` debemos especificar algunos atributos:

action: define el tipo de acción a llevar a cabo con el formulario. Como ya hemos dicho, existen dos posibilidades:

- El formulario es enviado a una dirección de correo electrónico. Para esto hay que poner el mail en el action.
- El formulario es enviado a un programa o script que procesa su contenido. Esta es la posibilidad que más nos interesa y abordaremos en otro módulo.

`<form action="ruta del método que va a manejar la información"></form>`

method: Este atributo se encarga de especificar la forma en la que el formulario es enviado. Los dos valores posibles que puede tomar este atributo son **POST** y **GET**. A efectos prácticos y, salvo que se diga lo contrario, daremos siempre el valor POST.

enctype: Se utiliza para indicar la forma en la que viajará la información que se mande por el formulario. En el caso más corriente, enviar el formulario por correo electrónico, el valor de este atributo debe de ser "text/plain". Así conseguimos que se envíe el contenido del formulario como texto plano dentro del email. Si fuéramos a enviar una imagen dentro del formulario, este atributo debería ser "multipart/form-data". También todos estos conceptos vamos a detallarlos más adelante.

Nota: Este último atributo no es indispensable. Sin embargo, si quisieramos guardar la información de nuestro formulario en Java, requeriremos indispensablemente de los primeros dos atributos. Estos conceptos los abordaremos y profundizaremos en otro módulo.



¿NECESITAS UN EJEMPLO?

Entonces con todo lo anterior ya explicado, la etiqueta `<form>` con los atributos nos quedaría así:

```
<form action="ruta del método que va a manejar la información" method="POST"  
enctype="multipart/form-data">
```

```
    <!-- contenido del formulario -->
```

```
</form>
```

Entre esta etiqueta y su cierre colocaremos el resto de las etiquetas que darán forma a nuestro formulario.

CAMPOS DE TEXTO

El lenguaje HTML nos propone una gran diversidad de alternativas a la hora de crear nuestros formularios, es decir, una gran variedad de elementos para diferentes propósitos. Estas van desde la clásica caja de texto, hasta la lista de opciones en un menú desplegable, pasando por las cajas de validación, etc.

Las etiquetas que tenemos que utilizar para crear campos de texto, pueden ser de dos tipos. Veamos en qué consiste cada una de estas modalidades y cómo podemos implementarlas en nuestro formulario.

ETIQUETA INPUT

Las cajas de texto son colocadas por medio de la etiqueta `<input>`. Dentro de esta etiqueta hemos de especificar el valor de dos atributos: **type** y **name**.

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

Como todos sabrán un input se ve así:

De este modo expresamos nuestro deseo de crear una caja de texto cuyo contenido será llamado "nombre" (por ejemplo, en el caso de la etiqueta anterior, pero podemos poner distintos nombres a cada uno de los campos de texto que habrá en los formularios).

ATRIBUTO TYPE

Como hemos visto el atributo type nos sirve para especificar el tipo de dato que se va a ingresar en nuestro input, en el ejemplo anterior lo habíamos puesto como tipo text, para que sea una caja de texto y poder ingresar texto. Pero existen otros tipos de valores para el atributo type

NUMBER

Este tipo permite al usuario ingresar números. Los navegadores vienen con validaciones para evitar que el usuario ingrese algo que no sea números. Además, en los navegadores modernos, los campos numéricos suelen venir con controles que permiten a los usuarios cambiar su valor de forma gráfica.

```
<input type="number">
```

Se vería así:

▼

DATE

Este le permite al usuario ingresar una fecha, ya sea mediante una caja de texto o una interfaz gráfica con selector de fecha.

```
<input type="date">
```

Se vería así:



EMAIL

Este tipo permite al usuario ingresar un mail. Los navegadores vienen con validaciones para validar que se esté ingresando con el formato correcto de un mail. Este input se va a ver como un input de texto común y corriente.

```
<input type="email">
```

TEXTO OCULTO

Hay determinados casos en los que podemos desear esconder el texto escrito en el campo input, por medio de círculos negros, de manera que aporte una cierta confidencialidad. Para esto vamos a usar el type **password**.

```
<input type="password">
```

Se vería así:



Más adelante veremos otros valores para el atributo type con otras utilidades.

ATRIBUTO NAME

Veamos de nuevo el ejemplo del principio:

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

En este ejemplo creamos una caja de texto cuyo contenido será llamado "nombre", elegimos nombre, pero podemos darle el valor que queramos.

El nombre del elemento del formulario es de gran importancia para poder identificarlo en nuestro programa de procesamiento (Java).

ATRIBUTOS PRESCINDIBLES.

Además de estos dos atributos, esenciales para el correcto funcionamiento de nuestra etiqueta, existen otra serie de atributos que pueden resultarnos de utilidad pero que no son imprescindibles:

size: define el tamaño de la caja de texto, en número de caracteres visibles. Si al escribir el usuario llega al final de la caja, el texto que escriba a continuación también cabrá dentro del campo, pero irá corriendo a medida que se escribe, haciendo desaparecer la parte de texto que queda a la izquierda.

maxlength: indica el tamaño máximo del texto, en número de caracteres, que puede ser escrito en el campo. En caso de que el campo de texto tenga definido el atributo maxlength, el navegador no permitirá escribir más caracteres en ese campo que los que hayamos indicado.

value: en algunos casos puede resultarnos interesante asignar un valor predefinido al campo en cuestión. Esto puede ayudar al usuario a llenar más rápidamente el formulario o darle alguna idea sobre la naturaleza de datos que se requieren. Este valor inicial del campo puede ser expresado mediante el atributo value.



¿NECESITAS UN EJEMPLO?

```
<input type="text" name="instituto" value="Egg Educación">
```

Con la línea de especificada anteriormente se genera un campo de texto como el siguiente:

Egg Educación

placeholder: este atributo especifica una pequeña pista que describe el valor esperado para el campo (input).

La pequeña sugerencia se muestra en el campo de entrada antes de que el usuario ingrese un valor. Una vez que escriba, ese valor va a desaparecer.



¿NECESITAS UN EJEMPLO?

```
<input type="text" name="nombre" placeholder="Nombre del usuario">
```

Genera un campo de este estilo:

Nombre del usuario

Nota: recordemos que todos estos ejemplos de input deben ir entre las etiquetas de apertura y de cierre form.

```
<form>
```

```
    <input type="text" name="instituto" value="Egg Educación">
```

```
</form>
```

ETIQUETA TEXTAREA PARA TEXTO LARGO

Si deseamos poner a disposición del usuario un campo de texto donde pueda escribir cómodamente sobre un espacio compuesto de varias líneas, hemos de invocar una nueva etiqueta: <textarea> y su cierre correspondiente.

Este tipo de campos son prácticos cuando el contenido a enviar no es un nombre, teléfono, edad o cualquier otro dato breve, sino más bien, un comentario, opinión, etc. En los que existe la posibilidad de que el usuario desee llenar varias líneas.

Dentro de la etiqueta textarea deberemos indicar, como para el caso visto anteriormente, el **atributo name** para asociar el contenido a un nombre que será asemejado a una variable en un lenguaje de programación. Además, podemos definir las dimensiones del campo a partir de los atributos siguientes:

- **rows:** define el número de líneas del campo de texto.
- **cols:** define el número de columnas del campo de texto.

La etiqueta queda por tanto de esta forma:

```
<textarea name="comentario" rows="10" cols="40"></textarea>
```

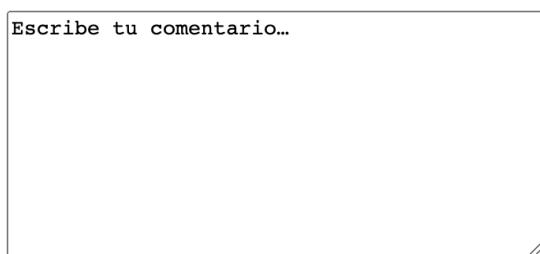
El resultado es el siguiente:



Asimismo, es posible predefinir el contenido del campo. Para ello, no usaremos el atributo value, sino que escribiremos dentro de la etiqueta el contenido que deseamos atribuirle.



```
<textarea name="comentario" rows="10" cols="40">Escribe tu comentario...</textarea>
```



Esta etiqueta al igual que el input debe ir dentro de la etiqueta form.

ETIQUETA LABEL

El elemento **<label>** y su etiqueta de cierre, provee una descripción corta que acompaña al campo de texto. Su función es indicarle al usuario que información debería ingresar en el campo o input.

También podemos asociar una etiqueta label a un campo para que el usuario pueda acceder al campo de texto con solo clickear el label.



La etiqueta podría verse de esta forma:

```
<label>Nombre del Usuario</label>
```

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

Esto en una página se vería así:

Nombre del Usuario

También podríamos poner un salto de línea para que el label quede arriba del input, si lo quisieramos.

Nombre del Usuario

Atributo for

La etiqueta label solo consta del atributo **for**. Mediante la utilización del atributo for podemos asociar el label con el input. Para lograr esto vamos a tener que utilizar también el atributo **id**, este atributo lo explicamos previamente y lo vamos a ver más en detalle en la parte de CSS.

La manera que asociamos un label a un input es la siguiente:

- En la etiqueta label agregaremos un **atributo for** con un valor, este va a representar el dato que se va a ingresar en el input.
- En el input vamos a poner el mismo valor, pero en el **atributo ID**.



```
<label for="nombre">Nombre del Usuario</label>
```

```
<input type="text" id="nombre" name="nombre">
```

El label y el input se verán igual pero ahora cuando el usuario cliquee el label, se va a activar el campo de texto del input para poder ingresar el valor que el usuario necesite. Despues vamos a ver un ejemplo más útil con las cajas de opciones.

OTROS ELEMENTOS DE FORMULARIOS

Seguramente hayan notado que los inputs son una manera muy práctica de hacernos llegar la información del navegante. No obstante, en muchos casos, permitir al usuario que escriba cualquier texto permite demasiada libertad y puede que la información que éste escriba no sea la que nosotros estamos necesitando.

Por ejemplo, pensemos que queremos que el usuario indique su país de residencia. En ese caso podríamos ofrecer una lista de países para que seleccione el que sea. Este mismo caso se puede aplicar a gran variedad de informaciones, como el tipo de tarjeta de crédito para un pago, la puntuación que da a un recurso, si quiere recibir o no un boletín de novedades, etc...

Este tipo de opciones predefinidas por nosotros pueden ser expresadas por medio de diferentes campos de formulario. Veamos a continuación cuáles son:

LISTAS DE OPCIONES

Las listas de opciones son ese tipo de menús desplegables que nos permiten elegir una (o varias) de las múltiples opciones que nos proponen. Para construirlas emplearemos una etiqueta **SELECT**, con su respectivo cierre.

Como para los casos ya vistos, dentro de esta etiqueta definiremos su nombre por medio del atributo name. Cada opción será incluida en una línea precedida de la etiqueta **OPTION**.

Podemos ver, a partir de estas explicaciones, la forma más típica y sencilla de esta etiqueta:

```
<select name="estacion">  
    <option>Primavera</option>  
    <option>Verano</option>  
    <option>Otoño</option>  
    <option>Invierno</option>  
</select>
```

Esto en una página se vería así:

A screenshot of a dropdown menu. The word "Primavera" is displayed in a white font inside a dark grey rounded rectangle, with a small downward arrow icon to its right.

Y cuando el usuario clickea en el select muestra las opciones así:

A screenshot of a dropdown menu. It shows four options: "Primavera", "Verano", "Otoño", and "Invierno". The first option, "Primavera", has a checkmark to its left and is highlighted with a darker grey background. The other three options are in a lighter grey background.

Atributos para la etiqueta select

Esta estructura puede verse modificada principalmente a partir de otros dos atributos:

size: indica el número de valores mostrados a la vez en la lista. Lo típico es que no se incluya ningún valor en el atributo size, en ese caso tendremos un campo de opciones desplegable, pero si indicamos un valor para el atributo size aparecerá un campo donde veremos las opciones definidas por size y el resto podrán ser vistos por medio de la barra lateral de desplazamiento.

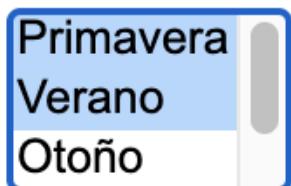
multiple: permite la selección de más elementos de la lista. Este atributo se expresa sin valor alguno, es decir, no se utiliza con el igual, simplemente se pone para conseguir el efecto, o no se pone si queremos una lista desplegable común.



¿NECESITAS UN EJEMPLO?

```
<select name="estacion" size="3" multiple>  
    <option>Primavera</option>  
    <option>Verano</option>  
    <option>Otoño</option>  
    <option>Invierno</option>  
</select>
```

Esto renderizado se vería así:



Vemos que se pueden seleccionar múltiples opciones (en este caso, "Primavera" y "Verano") y que además sólo son visibles 3 opciones.

Atributo para la etiqueta option

La etiqueta **<option>** puede asimismo ser modificada por medio de otro atributo.

selected: del mismo modo que multiple, este atributo no toma ningún valor, sino que simplemente indica que la opción que lo presenta está elegida por defecto.

```
<option selected>Otoño</option>
```

BOTONES DE RADIO

Existe otra alternativa para plantear una elección, en este caso, obligamos al usuario a elegir únicamente una de las opciones que se le proponen.

La etiqueta empleada en este caso es <**input**> en la cual usaremos el atributo type con el valor **radio**. Este atributo colocara una casilla pinchable al lado del valor del input.



¿NECESITAS UN EJEMPLO?

```
<input type="radio" name="estacion" value="1">Primavera  
<br>  
<input type="radio" name="estacion" value="2">Verano  
<br>  
<input type="radio" name="estacion" value="3">Otoño  
<br>  
<input type="radio" name="estacion" value="4">Invierno
```

Esto en una página web se vería así:

- Primavera
- Verano
- Otoño
- Invierno

En este tipo de input para elegir una opción debemos tocar en la casilla clickeable, pero habíamos explicado previamente en la etiqueta label, que podíamos hacer que la etiqueta label al clickearla se active la caja de texto del input. Ahora, podemos usar eso para que cuando el usuario cliquee la palabra primavera se seleccione esa opción. Esto se vería así:

```
<input type="radio" id="primavera" name="estacion" value="1">  
<label for="primavera">Primavera</label>  
<br>  
<input type="radio" id="verano" name="estacion" value="2">  
<label for="verano">Verano</label>  
<br>  
<input type="radio" id="otonio" name="estacion" value="3">  
<label for="otonio">Otoño</label>  
<br>  
<input type="radio" id="invierno" name="estacion" value="4">  
<label for="invierno">Invierno</label>
```

Esto en la página se verá igual que el anterior:

- Primavera
- Verano
- Otoño
- Invierno

La única diferencia va a ser que el usuario va a poder clickear el nombre de la estación que quiere para seleccionar esa opción, además de poder clickear la casilla.

CAJAS DE VALIDACIÓN

Este tipo de elementos pueden ser activados o desactivados por el visitante por un simple click sobre la caja en cuestión. Para esto vamos a usar la etiqueta INPUT con el valor **checkbox** en el atributo type.

```
<input type="checkbox" name="estacion" value="1">Primavera
```

Esto se verá así:

- Primavera
- Verano
- Otoño
- Invierno

BOTONES

Ha llegado el momento de explicar cómo podemos hacer un botón para provocar el envío del formulario, entre otras cosas.

Como podremos imaginarnos, en formularios no solamente habrá elementos o campos donde solicitar información del usuario, sino también habrá que implementar otra serie de funciones. Concretamente, han de permitirnos su envío mediante un botón. También puede resultar práctico poder proponer un botón de borrado o bien acompañar el formulario de datos ocultos que puedan ayudarnos en su procesamiento.

BOTÓN DE ENVÍO DE FORMULARIO (BOTÓN SUBMIT)

Para dar por finalizado el proceso de relleno del formulario y hacerlo llegar a su gestor, el usuario ha de enviarlo por medio de un botón previsto a tal efecto. Para esto vamos a utilizar la etiqueta **<button>** y su respectivo cierre. Dentro el elemento button se puede poner texto (y etiquetas como **<i>**, ****, ****, **
, **, etc.). Se vería así:

```
<button type="submit">Enviar</button>
```

Esto en la página se verá así:

Enviar

Como puede verse, tan solo hemos de especificar que se trata de un botón de envío (**type="submit"**) y hemos de definir el mensaje que queremos que aparezca escrito en el botón.

BOTÓN DE BORRADO (BOTÓN DE RESET)

Este botón nos permitirá borrar el formulario por completo, en el caso de que el usuario desee rehacerlo desde el principio. Su estructura sintáctica es parecida a la anterior:

```
<button type="reset">Borrar</button>
```

A diferencia del botón de envío, indispensable en cualquier formulario, el botón de borrado resulta meramente optativo y no es utilizado frecuentemente. Hay que tener cuidado de no ponerlo muy cerca del botón de envío y de distinguir claramente el uno del otro, para que ningún usuario borre el contenido del formulario que acaba de llenar por error.

BOTONES NORMALES

Dentro de los formularios también podemos colocar botones normales, pulsables como cualquier otro botón. Estos botones por si solos no tienen mucha utilidad, pero podremos necesitarlos para realizar acciones en el futuro. Su sintaxis es la siguiente:

```
<button type="button">Borrar</button>
```

DATOS OCULTOS (CAMPOS HIDDEN)

En algunos casos, aparte de los propios datos rellenados por el usuario, puede resultar práctico enviar datos definidos por nosotros mismos que ayuden al programa en su procesamiento del formulario. Este tipo de datos, que no se muestran en la página, pero si pueden ser detectados solicitando el código fuente, no son frecuentemente utilizados por páginas construidas en HTML, son más bien usados por páginas que emplean tecnologías de servidor. No se asusten, veremos más adelante qué quiere decir esto. Tan solo queremos dar constancia de su existencia y de su modo creación.

He aquí un ejemplo:

```
<input type="hidden" name="instituto" value="Egg Educación">
```

Esta etiqueta, incluida dentro de nuestro formulario, enviará un dato adicional al programa encargado de la gestión del formulario.



¿NECESITAS UN EJEMPLO?

Ejemplo completo de formulario

Ya hemos visto todo lo respectivo a formularios. Pasemos ahora a exemplificar todo lo aprendido a partir de la creación de un formulario.

Si tienes alguna duda, te recomendamos complementar el aprendizaje con los videos respectivos.

```
<form action="ruta del método que va a manejar la información" method="POST"
      enctype="multipart/form-data"></form>

<label>Nombre del usuario</label> <br>
<input type="text" name="nombre"> <br>

<label>Edad del usuario</label> <br>
<input type="number" name="edad "> <br>

<label>Fecha de nacimiento del usuario</label> <br>
<input type="date" name="fechanac"> <br>

<label>Sexo del usuario</label> <br>
<input type="radio" name="sexo" value="Hombre"> Hombre <br>
<input type="radio" name="sexo" value="Mujer"> Mujer <br>

<label>País nacimiento del usuario</label> <br>
<select name="pais">
    <option>Argentina</option>
    <option>Brasil</option>
    <option>Chile</option>
    <option>Uruguay</option>
</select>
<br>

<button type="submit">Enviar</button>
<button type="reset">Borrar</button>
```

Este formulario se verá así:

Nombre del usuario

Edad del usuario

Fecha de nacimiento del usuario
 dd / mm / aaaa

Sexo del usuario
 Hombre
 Mujer

Pais nacimiento del usuario



Hasta el momento has aprendido

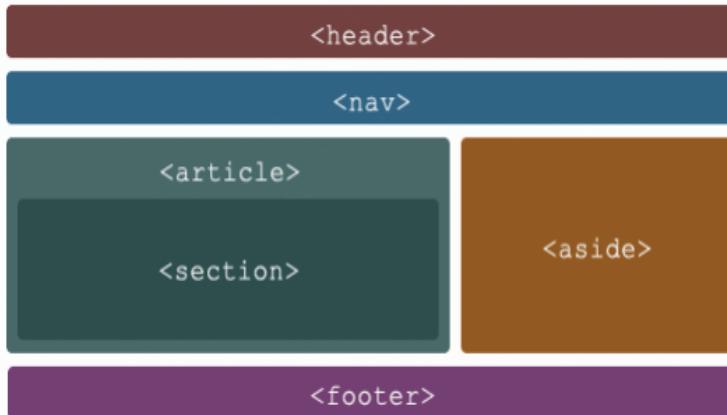
- Crear un campo de texto (input) adecuado al tipo de dato que necesitamos que se complete.
- Vincular campos de textos (input) con las etiquetas label mejorando la navegación del usuario.
- Realizar listas desplegables con opciones.
- Realizar botones de distintos tipos.
- Utilizar todos estos recursos para realizar un formulario competo.

SECCIONES EN HTML

Las páginas web se trabajan con lo que se conoce como un **esquema**. El esquema (*outline*) de una página web es un índice de los apartados de una página web que muestra la relación de jerarquía entre los diferentes apartados y subapartados. El concepto de esquema se formalizó en HTML 5 con más precisión que en HTML 4 / XHTML 1 y explica algunas características y formas de utilización de las etiquetas de secciones y bloques de contenido.

En relación con esto se pensó que las páginas de HTML se pueden dividir en secciones y en HTML 5 se introdujo una serie de etiquetas que nos van a ayudar con la división de nuestra página en secciones. Dentro de cada sección van a haber más etiquetas, esto es simplemente para que podemos tener un índice de los apartados de la página web.

La página dividida en secciones con sus respectivas etiquetas se ve así:



```
<body>
  <header>
    <a href="/"><img src=logo.png alt="home"></a>
    <hgroup>
      <h1>Title</h1>
      <h2 class="tagline">
        A lot of effort went into making this effortless.
      </h2>
    </hgroup>
  </header>
  <nav>
    <ul>
      <li><a href="#">home</a></li>
      <li><a href="#">blog</a></li>
      <li><a href="#">gallery</a></li>
      <li><a href="#">about</a></li>
    </ul>
  </nav>
  <section class="articles">
    <article>
      <time datetime="2009-10-22">October 22, 2009</time>
      <h2>
        <a href="#" title="link to this post">Travel day</a>
      </h2>
      <div class="content">
        Content goes here...
      </div>
      <section class="comments">
        <p><a href="#">3 comments</a></p>
      </section>
    </article>
  </section>
  <aside>
    <div class="related"></div>
  </aside>
```

<section>: se utiliza para representar una sección "general" dentro de un documento o aplicación, como un capítulo de un libro. Puede contener subsecciones y si lo acompañamos de h1-h6 podemos estructurar mejor toda la página creando jerarquías del contenido, algo muy favorable para el buen posicionamiento web.

<article>: representa un componente de una página que consiste en una composición autónoma en un documento, página, aplicación, o sitio web con la intención de que pueda ser reutilizado y repetido.

<aside>: representa una sección de la página que abarca un contenido relacionado con el contenido que lo rodea, por lo que se le puede considerar un contenido independiente. Este elemento puede utilizarse para efectos tipográficos, barras laterales, elementos publicitarios u otro contenido que se considere separado del contenido principal de la página.

<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).

<nav>: representa una sección de una página que enlaza a otras páginas o a otras partes dentro de la página. No todos los grupos de enlaces en una página necesita estar en un elemento nav, sólo las secciones que constan de bloques de navegación principales son apropiadas para el elemento de navegación.

<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.

<hgroup>: representa el encabezado de una sección. El elemento se utiliza para agrupar un conjunto de elementos h1-h6 cuando el título tiene varios niveles, tales como subtítulos o títulos alternativos.

ETIQUETAS EXTRAS

En este apartado que va a ser el último de nuestra parte de HTML vamos a ver unas etiquetas que no hemos visto todavía y que son importantes.

ETIQUETA DIV

La etiqueta div se conoce como etiqueta de división. Esta etiqueta se usa en HTML para hacer divisiones de contenido en la página web como (texto, imágenes, encabezado, pie de página, barra de navegación, etc.). La etiqueta div tiene etiquetas de apertura (`<div>`) y de cierre (`</div>`) y es obligatorio cerrar la etiqueta. Div es la etiqueta más útil en el desarrollo web porque nos ayuda a separar datos en la página web y podemos crear una sección particular para datos o funciones particulares en las páginas web. Cabe aclarar que la etiqueta div genera un salto de línea.

- La etiqueta Div es una etiqueta de nivel de bloque
- Es una etiqueta de contenedor genérica
- Se utiliza para agrupar varias etiquetas de HTML para que se puedan crear secciones y aplicarles estilo en conjunto.



¿NECESITAS UN EJEMPLO?

Supongamos que tenemos 3 párrafos que queremos alinear a la izquierda, esto recordemos lo haríamos con el **atributo align**. Crearíamos algo así:

```
<p align="left">Párrafo 1</p>
<p align="left">Párrafo 2</p>
<p align="left">Párrafo 3</p>
```

Una forma de simplificar nuestro código anterior y de evitar introducir continuamente el atributo align sobre cada una de nuestras etiquetas es utilizando la **etiqueta div**. Por lo tanto, usaremos un div para generar una sección de todos los párrafos y le agregaremos el atributo align a ese div.

Esto se vería así:

```
<div align="left">  
<p>Párrafo 1</p>  
<p>Párrafo 2</p>  
<p>Párrafo 3</p>  
</div>
```

Como hemos visto, la etiqueta DIV marca divisiones en las que definimos un bloque de contenido, y a los que podríamos aplicar estilo de manera global, aunque lo correcto sería aplicar ese estilo del lado del CSS.

ETIQUETA SPAN

El elemento span HTML es un contenedor en línea genérico para elementos y contenido en línea. Se usa para agrupar elementos con fines de compartir un determinado estilo (mediante el uso de los atributos de clase o id).

La mejor manera de usarlo es cuando no hay ningún otro elemento semántico disponible. Esta etiqueta es muy similar a la etiqueta div, pero se diferencian en que div es una etiqueta a nivel de bloque y span es una etiqueta en línea.

La etiqueta span es una etiqueta emparejada, lo que significa que tiene una etiqueta de apertura (<) y de cierre (>), y es obligatorio cerrar la etiqueta.

La etiqueta span se utiliza para agrupar elementos en línea y no realiza ningún cambio visual por sí misma.



¿NECESITAS UN EJEMPLO?

En el siguiente caso vemos que tenemos un párrafo donde necesitamos agregar estilos para una palabra. Por lo que agregaremos una etiqueta span para poder implementar ese estilo deseado.

```
<p>Yo estudio en <span style="color: yellow">Egg</span> Programación FullStack </p>
```

Esto en una página se vería así:

Yo estudio en Egg Programación FullStack

La etiqueta span no crea un salto de línea, sino que permite al usuario elegir cuando separar un elemento de otros dentro de la misma línea.

Nota: en el apartado de CSS vamos a ver mejor el atributo style y el atributo color. Ahora los estamos usando sólo para exemplificar.

CSS

En este apartado veremos qué es CSS, sus fundamentos y generalidades y cómo implementarlo en nuestros documentos HTML.

¿QUÉ ES CSS?

CSS es el acrónimo de **Cascading Style Sheets**, o lo que sería en español Hojas de Estilo en Cascada. Es un lenguaje que sirve para especificar el estilo o aspecto de las páginas web. CSS se define en base a un estándar publicado por una organización llamada W3C, que también se encarga de estandarizar el propio lenguaje HTML.

¿POR QUÉ EXISTE CSS?

El lenguaje HTML está limitado a la hora de aplicar forma a un documento. Sirve de manera excelente para especificar el contenido que debe tener una página web, pero no permite definir cómo ese documento se debe presentar al usuario.

Otro motivo que ha hecho necesaria la creación de CSS ha sido la separación del contenido de la presentación. Al inicio las páginas web tenían mezclado en su código HTML el contenido con las etiquetas necesarias para darle forma. Esto tiene sus inconvenientes, ya que la lectura del código HTML se hace pesada y difícil a la hora de buscar errores o depurar las páginas. Además, desde el punto de vista de la riqueza de la información y la utilidad de las páginas a la hora de almacenar su contenido, es un gran problema que los textos están mezclados con etiquetas incrustadas para dar forma a éstos, pues se degrada su utilidad.

CSS SOLVENTA ESTOS PROBLEMAS

Como hemos visto, para facilitar un correcto mantenimiento de las páginas web y para permitir que los diseñadores pudieran trabajar como sería deseable, había que introducir un nuevo elemento en los estándares y éste fue el lenguaje CSS.

CSS se ideó para aplicar el formato en las páginas, de una manera mucho más detallada, con nuevas posibilidades que no estaban al alcance de HTML. Al mismo tiempo, gracias a la posibilidad de aplicar el estilo de manera externa al propio documento HTML, se consiguió que el mantenimiento de las páginas fuese mucho más sencillo.

¿CUÁLES SON LAS VENTAJAS Y CARACTERÍSTICAS DE CSS?

El modo de funcionamiento de CSS consiste en definir, mediante una sintaxis especial, la forma de presentación que le aplicaremos a los elementos de la página.

Podemos aplicar CSS a muchos niveles, desde un sitio web entero hasta una pequeña etiqueta. Estos son los principales bloques de acción.

- **Una web entera:** de modo que se puede definir en un único lugar el estilo de toda una web, de una sola vez.
- **Un documento HTML o página en particular:** se puede definir la forma de cada uno de los bloques de contenido de una página, en una declaración que afectará a un solo documento de un sitio web.
- **Una porción del documento:** aplicando estilos visibles en un trozo de la página, como podría ser la cabecera.
- **Una etiqueta en concreto:** llegando incluso a poder definir varios estilos diferentes para una sola etiqueta. Esto es muy importante ya que ofrece potencia en nuestra programación.

La potencia de la tecnología salta a la vista. Pero no solo se queda ahí, ya que además esta sintaxis CSS permite aplicar al documento formato de modo mucho más exacto. Si antes el HTML se nos quedaba corto para maquetar las páginas y teníamos que utilizar trucos para conseguir nuestros efectos, ahora tenemos muchas más herramientas que nos permiten definir esta forma:

- Podemos definir la distancia entre líneas del documento.
- Se puede aplicar identado (sangrado) a las primeras líneas del párrafo.
- Podemos colocar elementos en la página con mayor precisión, y sin lugar a errores.
- Y mucho más, como definir la visibilidad de los elementos, márgenes, subrayados, tachados, etc.

Otra ventaja importante de CSS es la capacidad de especificar las medidas con diversas unidades. Si con HTML tan sólo podíamos definir atributos en las páginas con píxeles y porcentajes, ahora podemos definir utilizando muchas más unidades como:

- Píxeles (px) y porcentaje (%), como antes.
- Pulgadas (in).
- Puntos (pt).
- Centímetros (cm).
- Y otras que podrán explorar en el Anexo CSS disponible en el aula virtual

SINTAXIS HTML

La meta básica del lenguaje Cascading Style Sheet (CSS) es permitir al motor del navegador pintar elementos de la página con características específicas, como colores, posición o decoración. La sintaxis CSS refleja estas metas y estos son los bloques básicos de construcción.

- La propiedad que es un identificador, un nombre leíble por humanos, que define qué característica es considerada.
- El valor que describe como las características deben ser manejadas por el motor. Cada propiedad tiene un conjunto de valores válidos, definido por una gramática formal, así como un significado semántico, implementados por el motor del navegador.

DECLARACIONES DE CSS

La función principal de CSS es configurar determinadas propiedades con valores específicos. Este par (propiedad y valor) es llamado una declaración.

Ambos propiedades y valores son sensibles a mayúsculas y minúsculas en CSS. El par se separa por dos puntos, “：“, y espacios en blanco antes, entre ellos y después.

Declaración CSS:



Hay más de 100 propiedades diferentes en CSS y cerca de un número infinito de diferentes valores. No todos los pares de propiedades y valores son permitidos, cada propiedad define qué valores son válidos. Cuando un valor no es válido para una propiedad específica, la declaración es considerada inválida y es completamente ignorada por el motor del CSS.

BLOQUES DE DECLARACIONES EN CSS

Las declaraciones son agrupadas en bloques, que es una estructura delimitada por una llave de apertura, ‘{’, y una de cierre, ‘}’. Los bloques en ocasiones pueden anidarse, por lo que las llaves de apertura y cierre deben de coincidir.

Bloque css:

{

Aquí escribiremos las declaraciones css

}

Las llaves delimitan el inicio y el final del bloque.

Esos bloques son naturalmente llamados bloques de declaraciones y las declaraciones dentro de ellos están separadas por un punto y coma, “；”. Un bloque de declaración puede estar vacío, que es contener una declaración nula. Los espacios en blanco alrededor de las declaraciones son ignorados. En cuanto a la última declaración de un bloque, esta no necesita terminar en un punto y coma, aunque es usualmente considerado una buena práctica porque previene el olvidar agregarlo cuando se extienda el bloque con otra declaración.



¿NECESITAS UN EJEMPLO?

Bloque Css con declaraciones:

{

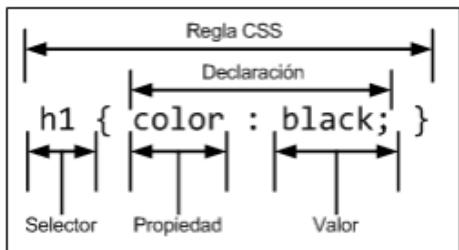
background-color : red;

background-style: none;

}

¿QUÉ PARTES CONFORMAN A UNA DECLARACIÓN CSS?

CSS define una serie de términos que permiten describir cada una de las partes que componen los estilos CSS. El siguiente esquema muestra las partes que forman un estilo CSS muy básico:



Los diferentes términos se definen a continuación:

Regla: cada uno de los estilos que componen una hoja de estilos CSS.

Selector: indica el elemento o elementos HTML a los que se aplica la regla CSS.

Declaración: especifica los estilos que se aplican a los elementos. Está compuesta por una o más propiedades CSS.

Propiedad: permite modificar el aspecto de una característica del elemento.

Valor: indica el nuevo valor de la característica modificada en el elemento.

INCLUIR CSS A NUESTRO HTML

CSS sirve para definir el aspecto de las páginas web, eso ya debe haber quedado claro. No obstante, hay diferentes niveles a los que podemos aplicar los estilos. Vamos a ir por orden, describiendo los puntos desde el más específico al más general, de manera que también iremos aumentando la dificultad e importancia de los distintos usos.

PEQUEÑAS PARTES DE LA PÁGINA

Para definir estilos en secciones reducidas de una página se puede utilizar el atributo **style** en la etiqueta sobre la que queremos aplicar estilos. Como valor de ese atributo indicamos en sintaxis CSS las características de estilos. Lo vemos con un ejemplo, pondremos un párrafo en el que determinadas palabras las vamos a visualizar en color verde.

```
<p> Yo estudié en <span style="color: yellow"> Egg Education </span> programación </p>
```

ESTILO DEFINIDO PARA UNA ETIQUETA

De este modo podemos hacer que toda una etiqueta muestre un estilo determinado. Por ejemplo, podemos definir un párrafo entero en color rojo y otro en color azul. Para ello utilizamos el atributo **style**, que es admitido por todas las etiquetas del HTML.

```
<p style="color: #990000">
```

Esto es un párrafo de color rojo.

```
</p>
```

ESTILO DEFINIDO EN UNA PARTE DE LA PÁGINA

Con la etiqueta <div> podemos definir secciones de una página y aplicarle estilos con el atributo **style**, es decir, podemos definir estilos de una vez a todo un bloque de la página.

```
<div style="color: #000099; font-weight: bold">  
    <h3>Estas etiquetas van en <strong>azul y negrita</strong></h3>  
    <p>  
        Seguimos dentro del DIV, por lo tanto permanecen los estilos  
    </p>  
</div>
```

ESTILO DEFINIDO PARA TODA UNA PÁGINA

Podemos definir, en la cabecera del documento, estilos para que sean aplicados a toda la página. Es una manera muy cómoda de darle forma al documento y muy potente, ya que estos estilos serán seguidos en toda la página y nos ahorraremos así "ensuciar" las etiquetas HTML colocando el atributo style.

Además, es común que los estilos declarados se quieran aplicar a distintas etiquetas dentro del mismo documento. Gracias a la aplicación de estilos para toda la página, podemos escribir los estilos una vez y usarlos para un número indefinido de etiquetas. Por ejemplo, podremos definir el estilo a todos los párrafos una vez y que se aplique igualmente, sea cual sea el número de párrafos del documento. Por último, también tendremos la ventaja que, si más adelante deseamos cambiar los estilos de todas las etiquetas, lo haremos de una sola vez, ya que el estilo fue definido una única vez de manera global.

A grandes rasgos, entre <style> y </style>, se coloca el nombre de la etiqueta (o selector) para la que queremos definir los estilos y entre llaves -{}- colocamos en sintaxis CSS las características de estilos. El concepto de selectores lo veremos más adelante.

```
<html>  
<head>  
    <title>Ejemplo de estilos para toda una página</title>  
    <style>  
        h1 { text-decoration: underline; text-align: center }  
        p { font-family: arial,verdana; color: white; background-color: black }  
        body { color: black; background-color: #cccccc; text-indent: 1cm }  
    </style>  
</head>  
<body>  
    <h1>Pagina con estilos</h1>  
    <p>Pagina con estilos de ejemplo</p>  
</body>  
</html>
```

Como se puede apreciar en el código, hemos definido que la etiqueta <h1> se presentará

- Subrayado
- Centrada

También, por ejemplo, hemos definido que el cuerpo entero de la página (etiqueta <body>) se le apliquen los estilos siguientes:

- Color del texto negro
- Color del fondo grisáceo
- Margen lateral de 1 centímetro

Cabe destacar que muchos de los estilos aplicados a la etiqueta <body> son heredados por el resto de las etiquetas del documento, como el color del texto o su tamaño. Esto es así, siempre y cuando no se vuelvan a definir esos estilos en las etiquetas hijas, en cuyo caso el estilo de la etiqueta más concreta será el que mande. Puede verse este detalle en la etiqueta <p>, que tiene definidos estilos que ya fueron definidos para <body>. Los estilos que se tienen en cuenta son los de la etiqueta <p>, que es más concreta.

ESTILO DEFINIDO PARA TODO UN SITIO WEB

Una de las características más potentes del desarrollo con hojas de estilos es la posibilidad de **definir los estilos de todo un sitio web en una única declaración**.

Esto se consigue creando un archivo de extensión **.css** donde tan sólo colocamos las declaraciones de estilos de la página y enlazando todas las páginas del sitio con ese archivo. De este modo, todas las páginas comparten una misma declaración de estilos, reutilizando el código CSS de una manera mucho más potente.

Este es el modelo más ventajoso de aplicar estilos al documento HTML y por lo tanto el más recomendable. De hecho, cualquier otro modo de definir estilos no es considerado una buena práctica y lo tenemos que evitar siempre que se pueda.

Algunas de las ventajas de este modelo de definición de estilos son las siguientes:

- Se ahorra en líneas de código HTML, ya que no tenemos que escribir el CSS en la propia página (lo que reduce el peso del documento y mejora la velocidad de descarga).
- Se mantiene separado correctamente lo que es el contenido (HTML) de la presentación (CSS), que es uno de los objetivos de las hojas de estilo y una de las máximas de todo desarrollador: cada cosa en su sitio.
- Se evita la molestia de definir una y otra vez los estilos con el HTML y lo que es más importante, si cambiamos la declaración de estilos, cambiarán automáticamente todas las páginas del sitio web. Esto es una característica muy deseable, porque aumenta considerablemente la facilidad de mantenimiento del sitio web.

¿CÓMO INCORPORAR ESTILOS DESDE UN ARCHIVO EXTERNO?

Veamos ahora cómo el proceso para incluir estilos con un fichero externo.

1- CREAMOS EL FICHERO CON LA DECLARACIÓN DE ESTILOS

Es un fichero de texto normal con la extensión **.css** para aclararnos qué tipo de archivo es. El texto que debemos incluir debe ser escrito exclusivamente en sintaxis CSS, es decir, sería erróneo incluir código HTML en él: etiquetas y demás. Podemos ver un ejemplo a continuación.

El nombre de este archivo va a ser **estilos.css**.

```
p {  
    font-size: 12cm;  
    font-family: arial, helvetica;  
    font-weight: normal;  
}  
  
h1 {  
    font-size: 36cm;  
    font-family: verdana, arial;  
    text-decoration: underline;  
    text-align: center;  
    background-color: Teal;  
}  
  
body {  
    background-color: #006600;  
    font-family: arial;  
    color: White;  
}
```

2- ENLAZAMOS LA PÁGINA WEB CON LA HOJA DE ESTILOS

Para ello, vamos a colocar la etiqueta **<link>** dentro de la etiqueta **<head></head>** con los atributos siguientes:

- **rel:** indica el tipo de relación que tiene el recurso enlazado y la página HTML. Para los archivos CSS, siempre se utiliza el valor **stylesheet**.
- **href:** indica la URL del archivo CSS que contiene los estilos. La URL indicada puede ser relativa o absoluta y puede apuntar a un recurso interno o externo al sitio web.

Veamos una página web entera que enlaza con la declaración de estilos anterior:

```
<html>
<head>
    <link rel="stylesheet" href="estilos.css">
    <title>Ejemplo de página que lee estilos </title>
</head>
<body>
    <h1>Pagina con estilos</h1>
    <p>Pagina con estilos de ejemplo</p>
</body>
</html>
```

SELECTORES CSS

Teniendo en cuenta que ya podemos asignarle estilos a todo un sitio web, mediante un archivo css que usa selectores para elegir las etiquetas a las que asignarles los estilos, también tenemos que entender que existen varios tipos de selectores

SELECTOR UNIVERSAL

Se utiliza para seleccionar todos los elementos de la página. El siguiente ejemplo elimina el margen y el relleno de todos los elementos HTML (por ahora no es importante fijarse en la parte de la declaración de la regla CSS):

```
* {
    margin: 0;
    padding: 0;
}
```

SELECTOR DE ETIQUETA

Selecciona todos los elementos de la página cuya etiqueta HTML coincide con el valor del selector. El siguiente ejemplo selecciona todos los párrafos de la página:

```
p {
    text-align: justify;
    font-family: Verdana;
}
```

El siguiente ejemplo selecciona todas las tablas y div de la página:

```
table, div {
    border: 1px solid red;
}
```

SELECTOR DESCENDENTE

Selecciona los elementos que se encuentran dentro de otros elementos. Un elemento es descendiente de otro cuando se encuentra entre las etiquetas de apertura y de cierre del otro elemento.

El selector del siguiente ejemplo selecciona todos los elementos `` de la página que se encuentren dentro de un elemento `<p>`.

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

SELECTOR DE CLASE

¿Como hago para aplicarle estilos solo al primer párrafo?

Una de las soluciones más sencillas para aplicar estilos a un solo elemento de la página consiste en utilizar el atributo class de HTML sobre ese elemento para indicar directamente la regla CSS que se le debe aplicar. Ejemplo:

HTML:

```
<body>  
<p class="destacado"> Párrafo 1</p>  
<p class="error"> Párrafo 2</p>  
<p> Párrafo 3</p>  
</body>
```

CSS:

```
.destacado {  
    font-size: 15px;  
}  
.error {  
    color: red;  
}
```

En nuestro archivo CSS para especificar una clase, vamos a poner punto ('.') y el nombre de la clase que queremos que coincida con valor que pongamos en nuestro atributo class en el html.

Entonces, en el ejemplo podemos ver como el primer párrafo tiene el valor **destacado** y el segundo párrafo el valor **error** para el atributo class y en nuestro archivo CSS, hemos definido un estilo para esas clases.

El beneficio del atributo class, además de dejarnos asignar estilos a un solo elemento, es que después podemos reutilizar esa class para asignarle ese estilo a otros párrafos concretos o a otras etiquetas, solo deberemos ponerle el valor de un estilo que ya existe en el atributo class.

SELECTOR DE ID

En un documento HTML, los selectores de ID de CSS buscan un elemento basado en el contenido del atributo id. El atributo ID del elemento seleccionado debe coincidir exactamente con el valor dado en el selector. Este tipo de selectores sólo seleccionan un elemento de la página porque el valor del atributo id no se puede repetir en dos elementos diferentes de una misma página.

HTML:

```
<div id="identificador"> ¡Este div tiene un ID especial! </div>
```

```
<div> Este solo es un div regular. </div>
```

CSS:

```
#identificador{
```

```
background-color: blue;
```

```
}
```

En nuestro archivo CSS para especificar un ID, vamos a poner el numeral ('#') y el nombre del ID que queremos que coincida que con valor que pongamos en nuestro atributo ID en el html.

Como podemos ver el ID, es muy parecido al atributo class, pero la diferencia es que el ID se puede usar para identificar un solo elemento, mientras que una clase se puede usar para agrupar más de uno.



MANOS A LA OBRA!

Actividad: Presentación Personal – Estilos CSS

En esta actividad incorporaremos algunos estilos a la página que hemos creado, con el motivo de practicar lo aprendido sobre CSS.

En el Anexo CSS (material disponible en el aula virtual) encontrarás diferentes formas de utilizar las propiedades que utilizaremos en la actividad para así, sacarle más provecho.

- 1) Abre tu IDE y crea un nuevo archivo (en la actividad 1 se explica cómo). Pero esta vez, con la extensión “css” en lugar de “html”. Puedes nombrarlo como prefieras. Te recomendamos algunos nombres: “estilos.css”, “style.css”.
- 2) Agrega una declaración CSS con selector universal (estas aplican a todo el documento). Dentro de esta declaración especificaremos el valor de las siguientes propiedades:
 - a. background-color: (elige el color que prefieras).
 - b. font-family: (elige la fuente que prefieras).
- 3) Agregaremos una declaración CSS con selector de etiqueta, para todas las etiquetas H1 (o el encabezado de mayor relevancia que posea tu documento HTML). En esta declaración te pediremos que especifiques los valores para las siguientes propiedades:
 - a. font-size: (elige el tamaño que prefieras. Recomendamos usar px).
 - b. color: (elige el color que prefieras).

PRIORIDAD EN APLICACIÓN DE ESTILO

Ahora que entendemos los selectores en CSS, tenemos que entender como priorizar los estilos en CSS, para saber qué estilo hace efecto y cuál no sobre determinado elemento.

HERENCIA

Los hijos heredan los estilos de sus elementos padres, no es necesario declarar sus estilos si estos se mantienen igual.

```
body{  
    color: yellow;  
}  
h2{  
    color: yellow; /*No es necesario*/  
}
```

CASCADA

Todo estilo sobrescribe a uno anterior.

```
h2{  
    color: yellow;  
}  
h2{  
    color: red;  
}
```

ESPECIFICIDAD

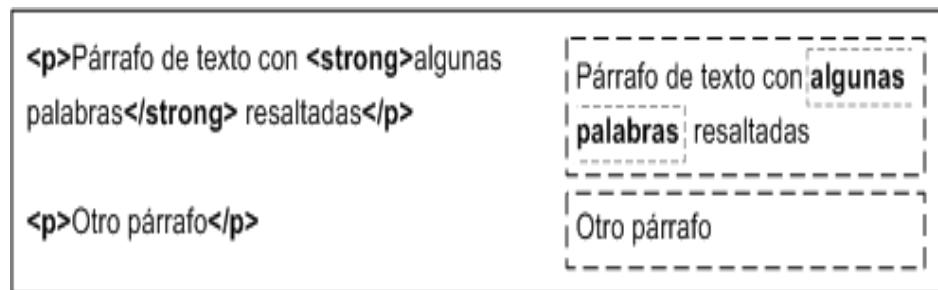
Cuando hay conflictos de estilos el navegador aplica sólo el de mayor especificidad.

```
h2{  
    color: red  
}  
h2.subtitle{  
    color: purple;  
}
```

¿QUÉ ES EL MODELO DE CAJA?

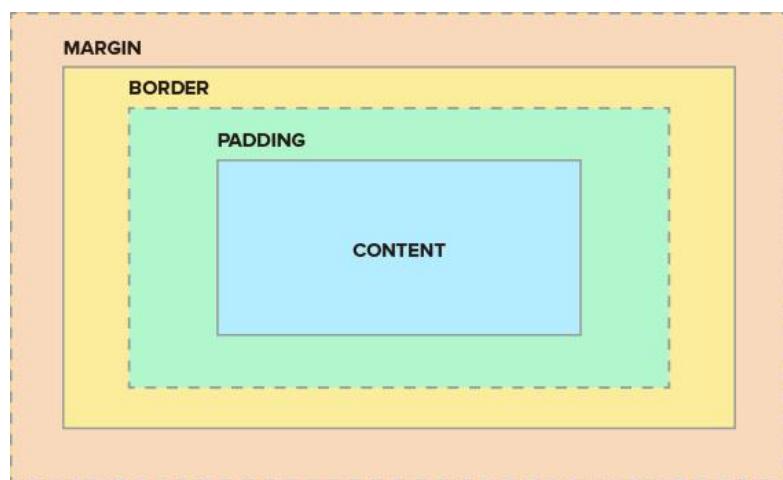
El **modelo de cajas** o "box model" es seguramente la característica más importante del lenguaje de hojas de estilos CSS, ya que condiciona el diseño de todas las páginas web. El modelo de cajas es el comportamiento de CSS que hace que todos los elementos de las páginas se representen mediante cajas rectangulares.

Las cajas de una página se crean automáticamente. Cada vez que se inserta una etiqueta HTML, se crea una nueva caja rectangular que encierra los contenidos de ese elemento. La siguiente imagen muestra las tres cajas rectangulares que crean las tres etiquetas HTML que incluye la página:



Las cajas de las páginas no son visibles a simple vista porque inicialmente no muestran ningún color de fondo ni ningún borde.

Los navegadores crean y colocan las cajas de forma automática, pero CSS permite modificar todas sus características. Cada una de las cajas está formada por cuatro partes, tal y como muestra la siguiente imagen:



- **Contenido** (content): se trata del contenido HTML del elemento (las palabras de un párrafo, una imagen, el texto de una lista de elementos, etc.)
- **Relleno** (padding): espacio libre opcional existente entre el contenido y el borde.
- **Borde** (border): línea que encierra completamente el contenido y su relleno.
- **Margen** (margin): separación opcional existente entre la caja y el resto de las cajas adyacentes.

Existen otras dos partes de una caja que son:

- **Imagen de fondo** (background image): imagen que se muestra por detrás del contenido y el espacio de relleno.
- **Color de fondo** (background color): color que se muestra por detrás del contenido y el espacio de relleno.

El relleno y el margen son transparentes, por lo que en el espacio ocupado por el relleno se muestra el color o imagen de fondo (si están definidos) y en el espacio ocupado por el margen se muestra el color o imagen de fondo de su elemento padre (si están definidos). Si ningún elemento padre tiene definido un color o imagen de fondo, se muestra el color o imagen de fondo de la propia página (si están definidos).

Si una caja define tanto un color como una imagen de fondo, la imagen tiene más prioridad y es la que se visualiza. No obstante, si la imagen de fondo no cubre totalmente la caja del elemento o si la imagen tiene zonas transparentes, también se visualiza el color de fondo. Combinando imágenes transparentes y colores de fondo se pueden lograr efectos gráficos muy interesantes.

PSEUDO-CLASES

CSS también permite aplicar diferentes estilos a un mismo enlace en función de su estado. De esta forma, es posible cambiar el aspecto de un enlace cuando por ejemplo el usuario pasa el ratón por encima o cuando el usuario pincha sobre ese enlace.

Como con los atributos id o class no es posible aplicar diferentes estilos a un mismo elemento en función de su estado, CSS introduce un nuevo concepto llamado **pseudo-clases**. En concreto, CSS define las siguientes cuatro pseudo-clases:

- **:link**, aplica estilos a los enlaces que apuntan a páginas o recursos que aún no han sido visitados por el usuario.
- **:visited**, aplica estilos a los enlaces que apuntan a recursos que han sido visitados anteriormente por el usuario. El historial de enlaces visitados se borra automáticamente cada cierto tiempo y el usuario también puede borrarlo manualmente.
- **:hover**, aplica estilos al enlace sobre el que el usuario ha posicionado el puntero del ratón.
- **:active**, aplica estilos al enlace que está clickeado el usuario.

Esto se vería así:

```
/* link sin visitar */  
a:link {  
    color: red;  
}
```

```
/* link visitado */  
a:visited {  
    color: green;  
}
```

```
/* mouse sobre el link */  
a:hover {  
    color: pink;  
}
```

```
/* link clickeado */  
a:active {  
    color: blue;  
}
```

BOOTSTRAP

Es un framework de interfaz de usuario, de código abierto, creado para un desarrollo web más rápido y sencillo. Mark Otto y Jacob Thornton fueron los creadores iniciales. El framework combina CSS y JavaScript para estilizar los elementos de una página HTML.

Contiene todo tipo de plantillas de diseño basadas en HTML y CSS para diversas funciones y componentes, como navegación, sistema de cuadrícula, carruseles de imágenes y botones.

Si bien Bootstrap ahorra tiempo al desarrollador de tener que administrar las plantillas repetidamente, su objetivo principal es crear sitios responsive. Permite que la interfaz de usuario de un sitio web funcione de manera óptima en todos los tamaños de pantalla, ya sea en teléfonos de pantalla pequeña o en dispositivos de escritorio de pantalla grande.

Por lo tanto, los desarrolladores no necesitan crear sitios específicos para dispositivos y limitar su rango de audiencia.

ARCHIVOS PRIMARIOS DE BOOTSRAP

Ya sabemos qué es Bootstrap; consiste en una colección de sintaxis que realizan funciones específicas. Debido a esto, tiene sentido que el marco tenga solo tres diferentes tipos de archivos. A continuación, detallamos los tres archivos principales que administran esta interfaz de usuario y la funcionalidad de un sitio web.

BOOTSTRAP.CSS

Esta es la hoja de estilos de bootstrap, gracias a esta podremos implementar estilos ya definidos y así estilizar nuestra página de una manera sencilla. Además, las plantillas que contiene bootstrap, usan esta hoja de estilos.

BOOTSTRAP.JS

Este archivo es la parte principal de Bootstrap. Consiste en archivos JavaScript que son responsables de la interactividad del sitio web.

CÓMO USAR BOOTSTRAP

Para utilizar bootstrap lo único que vamos a tener que hacer es ir a estas dos páginas:

<https://getbootstrap.com/docs/4.5/getting-started/introduction/#css>

<https://getbootstrap.com/docs/4.5/getting-started/introduction/#js>

Dentro de estas dos páginas vamos a encontrar una etiqueta link para el CSS de Bootstrap y unas etiquetas script para el JavaScript de Bootstrap.

Para poder usar Bootstrap lo que haremos es pegar el link con la hoja de estilos de Bootstrap en la etiqueta <head> de nuestro html y las etiquetas script antes de la etiqueta de cierre </body>.

Esto se vería así:

```
<!DOCTYPE html>

<html>
  <head>
    <meta charset="utf-8">

    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@4.5.3/dist/css/bootstrap.min.css" integrity="sha384-TX8t27EcRE3e/ihU7zmQxVncDAy5ulKz4rEkgIXeMed4M0jlIDPvg6uqKI2xXr2" crossorigin="anonymous">

    <title>Pagina con bootstrap</title>

  </head>
  <body>

    <script src="https://code.jquery.com/jquery-3.5.1.slim.min.js" integrity="sha384-DfXd2htPH0lsSSs5nCTpuj/zy4C+OGpamoFVy38MVBnE+IbbVYUew+OrCXaRkfj" crossorigin="anonymous"></script>

    <script src="https://cdn.jsdelivr.net/npm/bootstrap@4.5.3/dist/js/bootstrap.bundle.min.js" integrity="sha384-ho+j7yWK8fNQe+A12Hb8AhRq26LrZ/JpcUGGOn+Y7RsweNrtN/tE3MoK7ZeZDyx" crossorigin="anonymous"></script>

  </body>
</html>
```

Una vez que hemos hecho esto, ya podemos usar Bootstrap y sus plantillas.

PLANTILLAS

Dado que es uno de los framework más utilizados, podemos encontrar un amplio abanico de marcos de trabajo pensados y diseñados a partir de los componentes y estilos que presenta Bootstrap, de modo que existen variables y ejemplos listos para utilizar en proyectos específicos.

<https://themes.getbootstrap.com/official-themes/>

EJERCICIOS DE APRENDIZAJE

Para la realización de los ejercicios que se describen a continuación, es necesario tener instalado Visual Studio Code o alguna aplicación parecida. El material para realizar la instalación se encuentra en el aula virtual. En este apartado tendrás 13 ejercicios para realizar.



VIDEOS: Te sugerimos ver los videos relacionados con este tema, antes de empezar los ejercicios, los podrás encontrar en tu aula virtual o en nuestro canal de YouTube.

Recomendamos también ir al apartado de bibliografía al final de la guía por si necesitamos reforzar o queremos saber más sobre los temas vistos.

1. Crear un archivo HTML, que contenga un encabezado `<h1>` seguido de un párrafo y un encabezado `<h2>` seguido de otro párrafo. Después, hacer que un párrafo se muestre en negrita y el otro en itálica. Utilizar saltos de línea si los consideran necesarios.
2. Ahora vamos a tener que centrar nuestros encabezados y nuestros párrafos alinearlos a la izquierda. A continuación, después del último párrafo vamos a tener que crear una lista ordenada de lo que queramos. Tendremos que mostrar la lista donde la enumeración sean letras del alfabeto. **Sin usar CSS**
3. Ahora crearemos otro archivo HTML en que crearemos una lista anidada de enlaces, deberá verse así:

- **Buscadores**
 - **Google**
 - **Bing**
- **Redes sociales**
 - **Instagram**
 - **Twitter**

- a. Cada buscador y red social que sale en la lista deben ser links a las respectivas páginas.
- b. Recordemos que no debemos utilizar CSS para lograr ninguna de estas tareas, la idea es practicar HTML por su cuenta y después sumaremos CSS.

4. Crear un nuevo archivo HTML en el que explicaremos la receta para hacer papas fritas. La página debería verse así:

Papas fritas

Receta de papas fritas caseras



Ingredientes

- 3 o 4 papas (300gr)
- Aceite
- Sal

Elaboración (Pasos)

- Pelar las papas
- Cortalas en baston
- Calentar aceite en una sartén
- Cocinar hasta que estén doradas
- Removerlas del aceite y salar al gusto

5. Ahora vamos a crear una página web de nuestra banda favorita, vamos a mostrar un ejemplo con los Beatles:

Los Beatles

Es una banda de rock formada en el año 1960 en Liverpool.



Ingrantes

- Paul McCartney
- John Lennon
- Ringo Star
- George Harrison

Año	Disco
1965	Help!
1968	The Beatles
1969	Abbey Road

"Abbey Road fue su ultimo disco".

La tabla tendrá bordes que se lo debemos agregar sin css. Investigar atributo border.

6. Por último, vamos a crear un formulario para registrar un usuario que se vea de la siguiente manera:

Registrar un usuario

Nombre del usuario

Contraseña del usuario

Edad del usuario

Fecha de nacimiento del usuario

 dd/mm/aaaa

Sexo del usuario

- Hombre
 Mujer
 Prefiero no decir

Pais nacimiento del usuario

 Argentina

Es importante que en las casillas de sexo del usuario se puede clickear la/s palabra/s hombre, mujer o prefiero no decir para seleccionar la opción. Recordemos que eso lo podemos hacer con la etiqueta label.

7. Crear un archivo HTML y un archivo CSS, vamos a linkear el archivo CSS al archivo HTML y vamos a hacer lo mismo que el primer ejercicio, pero ahora la página va a tener un color de fondo a elección, el encabezado H1 tiene que tener una fuente a elección y estar centrado, lo mismo para el encabezado H2, y por ultimo los párrafos deben tener una fuente a elección, deben tener un color a elección y deben estar centrados a la izquierda.

8. Definir las reglas CSS que permiten mostrar los enlaces con los siguientes estilos:

- En su estado normal, los enlaces se muestran de color rojo #CC0000.
- Cuando el usuario pasa su ratón sobre el enlace, se muestra con un color de fondo rojo #CC0000 y la letra de color blanco #FFF.
- Los enlaces visitados se muestran en color gris claro #CCC.

9. A partir del siguiente código HTML proporcionado, añadir las reglas CSS necesarias para que la página resultante tenga el mismo aspecto que el de la siguiente imagen:

Lorem ipsum dolor sit amet

Nulla pretium. Sed tempus nunc vitae neque. **Suspendisse gravida**, metus a scelerisque sollicitudin, lacus velit ultricies nisl, nonummy tempus neque diam quis felis. **Etiam sagittis tortor** sed arcu sagittis tristique.

Aliquam tincidunt, sem eget volutpat porta

Vivamus velit dui, placerat vel, feugiat in, ornare et, urna. [Aenean turpis metus, aliquam non, tristique in](#), pretium varius, sapien. Proin vitae nisi. Suspendisse porttitor purus ac elit. Suspendisse eleifend odio at dui. In elit sed metus pretium elementum.

	Título columna 1	Título columna 2
Título fila 1	Donec purus ipsum	Curabitur <i>blandit</i>
Título fila 2	Donec purus ipsum	Curabitur blandit
	Título columna 1	Título columna 2

Donec purus ipsum, posuere id, venenatis at, placerat ac, lorem. Curabitur blandit, eros sed gravida aliquet, risus justo porta lorem, ut mollis lectus tortor in orci. Pellentesque nec augue.

Fusee nec felis eu diam pretium adipiscing. Nunc elit elit, vehicula vulputate, venenatis in, posuere id, lorem. Etiam sagittis, tellus in ultrices accumsan, diam nisi feugiat ante, eu congue magna mi non nisl.

Vivamus ultrices aliquet augue. Donec arcu pede, pretium vitae, rutrum aliquet, tincidunt blandit, pede. Aliquam in nisi. Suspendisse volutpat. Nulla facilisi. Ut ullamcorper nisi quis mi.

Nota: el código para este ejercicio se encuentra en GitHub o Moodle.

10. Ahora vamos a utilizar Bootstrap. Deberemos crear una página que tenga un encabezado H1 con un párrafo y un encabezado H2 con un párrafo. Tenemos que lograr que se vean con los estilos de Bootstrap.

11. Usar Bootstrap para mostrar una tabla de productos así:

Nombre	Precio
Producto1	10000
Producto2	10000
Producto3	10000

12. Una vez que tenemos esa tabla vamos a sumarle una columna más que se vea así:

Nombre	Precio	Detalle
Producto1	10000	<button>Ver Producto</button>
Producto2	10000	<button>Ver Producto</button>
Producto3	10000	<button>Ver Producto</button>

Para esto investigar la clase **button** de Bootstrap para las etiquetas <a> e investigar los colores de Bootstrap y como sumarlos.

13. Un formulario para que el usuario se pueda registrar en nuestra página usando Bootstrap.

Formulario Registro

Email:

Contraseña:

Recuerdame

EJERCICIOS DE APRENDIZAJE EXTRAS

Estos van a ser ejercicios para reforzar los conocimientos previamente vistos. Estos pueden realizarse cuando hayas terminado la guía y tengas una buena base sobre lo que venimos trabajando. Además, si ya terminaste la guía y te queda tiempo libre en las mesas, podes continuar con estos ejercicios extra, recordando siempre que no es necesario que los termines para continuar con el tema siguiente. Por último, recuerda que la prioridad es ayudar a los compañeros de la mesa y que cuando tengas que ayudar, lo más valioso es que puedas explicar el ejercicio con la intención de que tu compañero lo comprenda, y no sólo mostrarlo. ¡Muchas gracias!

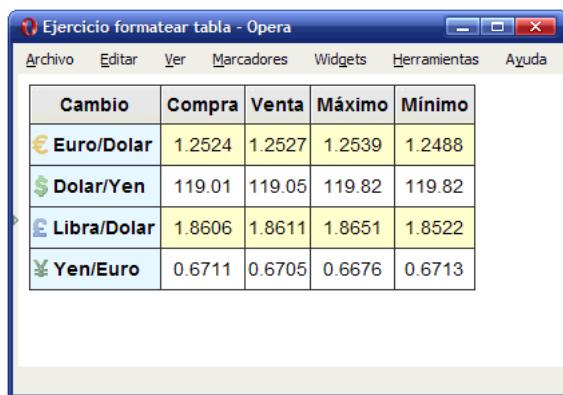
1. Para este apartado vamos a crear 2 listas:
 - a. Lista con viñetas con aspectos personalizados a elección.
 - b. Lista vertical con links a páginas a elección.
2. Determinar las reglas CSS necesarias para mostrar la siguiente tabla con el aspecto final mostrado en la imagen (modificar el código HTML que se considere necesario añadiendo los atributos class oportunos).

Tabla original:



Cambio	Compra	Venta	Máximo	Mínimo
Euro/Dolar	1.2524	1.2527	1.2539	1.2488
Dolar/Yen	119.01	119.05	119.82	119.82
Libra/Dolar	1.8606	1.8611	1.8651	1.8522
Euro/Yen	149.09	149.13	149.79	148.96

Tabla final:



Cambio	Compra	Venta	Máximo	Mínimo
€ Euro/Dolar	1.2524	1.2527	1.2539	1.2488
\$ Dolar/Yen	119.01	119.05	119.82	119.82
£ Libra/Dolar	1.8606	1.8611	1.8651	1.8522
¥ Yen/Euro	0.6711	0.6705	0.6676	0.6713

Pasos a realizar:

1. Alinear el texto de las celdas, cabeceras y título. Definir los bordes de la tabla, celdas y cabeceras (color gris oscuro #333).

Ejercicio formatear tabla - Opera

Cambio	Compra	Venta	Máximo	Mínimo
Euro/Dolar	1.2524	1.2527	1.2539	1.2488
Dolar/Yen	119.01	119.05	119.82	119.82
Libra/Dolar	1.8606	1.8611	1.8651	1.8522
Yen/Euro	0.6711	0.6705	0.6676	0.6713

- Formatear las cabeceras de fila y columna con la imagen de fondo correspondiente en cada caso (fondo_gris.gif, euro.png, dolar.png, yen.png, libra.png). Modificar el tipo de letra de la tabla y utilizar Arial. El color azul claro es #E6F3FF.

Ejercicio formatear tabla - Opera

Cambio	Compra	Venta	Máximo	Mínimo
€ Euro/Dolar	1.2524	1.2527	1.2539	1.2488
\$ Dolar/Yen	119.01	119.05	119.82	119.82
£ Libra/Dolar	1.8606	1.8611	1.8651	1.8522
¥ Yen/Euro	0.6711	0.6705	0.6676	0.6713

- Mostrar un color alterno en las filas de datos (color amarillo claro #FFFFCC).

Ejercicio formatear tabla - Opera

Cambio	Compra	Venta	Máximo	Mínimo
€ Euro/Dolar	1.2524	1.2527	1.2539	1.2488
\$ Dolar/Yen	119.01	119.05	119.82	119.82
£ Libra/Dolar	1.8606	1.8611	1.8651	1.8522
¥ Yen/Euro	0.6711	0.6705	0.6676	0.6713

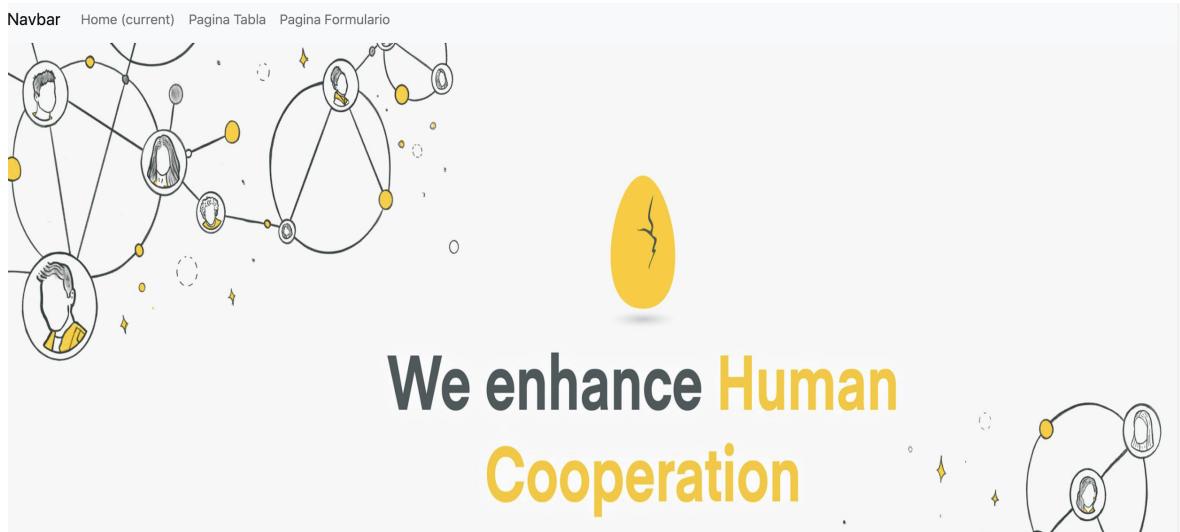
Nota: el código para este ejercicio se encuentra en GitHub o Moodle.

3. Siguiendo el ejercicio de Bootstrap, haremos una landing page que contenga un navbar. Investigar el navbar en Bootstrap, con botones a todas las páginas previamente mencionadas.

Navbar Home (current) Pagina Tabla Pagina Formulario

Pueden también hacer que ese navbar salga en el resto de las páginas.

4. Además, le sumaremos una imagen a la landing page usando las clases para la etiqueta que nos provee Bootstrap. Ejemplo:



5. Ahora deberemos hacer una página para el detalle del producto. Una página que muestre el nombre, el precio del producto y un botón para comprar y otro para agregar al carrito. A esta página se accederá con el botón en a la tabla de ver detalle.



6. Por último, sumarle a la landing cartas que muestren productos de esta manera:

Ejemplo Producto Descripción del producto Ver Producto	Ejemplo Producto Descripción del producto Ver Producto
Ejemplo Producto Descripción del producto Ver Producto	Ejemplo Producto Descripción del producto Ver Producto
Ejemplo Producto Descripción del producto Ver Producto	Ejemplo Producto Descripción del producto Ver Producto

BIBLIOGRAFÍA

Información sacada de las páginas:

HTML:

- <https://desarrolloweb.com/manuales/manual-html.html>
- https://developer.mozilla.org/es/docs/Learn/Getting_started_with_the_web/HTML_basics
- <https://www.arkaitzgarro.com/xhtml/capitulo-2.html#que-es-html>
- <https://www.geeksforgeeks.org/span-tag-html/>
- <https://www.geeksforgeeks.org/div-tag-html/>

CSS:

- <https://desarrolloweb.com/manuales/manual-css-hojas-de-estilo.html>
- <https://uniwebsidad.com/libros/css/capitulo-1>
- <https://lenguajecss.com/css/modelo-de-cajas/unidades-css/>
- https://developer.mozilla.org/es/docs/Learn/CSS/First_steps
- <https://web.dev/learn/css/>