

**UNIVERSIDAD CATÓLICA DE SANTA MARÍA**  
**PROGRAMA PROFESIONAL DE INGENIERÍA DE SISTEMAS**

## SESIÓN N° 01:

### HTML 5

#### I

#### OBJETIVOS

- ❖ Evaluar la eficiencia y la accesibilidad de diferentes estructuras HTML5 para presentar información en la web, justificando las elecciones de diseño basadas en los principios de programación modernos.
- ❖ Crear páginas web interactivas y dinámicas utilizando HTML5, CSS y JavaScript, integrando formularios, validaciones y elementos multimedia para una experiencia de usuario óptima.
- ❖ Aplicar las mejores prácticas de SEO en la construcción de páginas web, utilizando metadatos, Open Graph y Twitter Cards para mejorar la visibilidad y el posicionamiento en los motores de búsqueda.
- ❖ Utilizar herramientas de control de versiones como Git y plataformas de alojamiento como GitHub Pages para gestionar y publicar proyectos web de manera colaborativa y eficiente.
- ❖ Demostrar iniciativa y autonomía en la resolución de problemas técnicos, buscando y aplicando soluciones creativas e innovadoras en el desarrollo de aplicaciones web.
- ❖ Valorar la importancia de la accesibilidad y la usabilidad en el diseño web, asegurando que el contenido sea inclusivo y esté disponible para todos los usuarios, independientemente de sus capacidades.

#### II

#### TEMAS A TRATAR

- ❖ Introducción.
- ❖ ¿Qué es HTML?
- ❖ Estructura básica HTML
- ❖ Estructura de una etiqueta HTML
- ❖ Etiquetas semánticas y no semánticas
- ❖ Categorización de etiquetas
- ❖ Github Pages
- ❖ Etiquetas de formulario y validación
- ❖ Tipos de Input
- ❖ Atributos de validaciones
- ❖ Patrones regulares (regex)
- ❖ ¿Qué es una tabla?
- ❖ ¿Cuándo usar tablas
- ❖ Atributos
- ❖ ¿Qué es SEO?
- ❖ Metadata
- ❖ Open Graph
- ❖ Twitter Card
- ❖ Resumen

#### III

#### MARCO TEORICO

### 1. INTRODUCCIÓN

La presentación de información ha evolucionado a lo largo de los años, pasando de medios impresos a formatos electrónicos y, finalmente, a la vasta red de internet. El desarrollo de la web, impulsado en parte por el deseo de compartir conocimiento científico de manera más

accesible, ha transformado la forma en que interactuamos con la información. La inclusión de hipervínculos en documentos electrónicos dio origen al hipertexto, permitiendo la navegación fluida entre diferentes fuentes de información y sentando las bases para la creación de la web tal como la conocemos hoy en día.

El lenguaje HTML (Hyper Text Markup Language), en su versión 5, se ha consolidado como la columna vertebral de la web. A través de etiquetas y atributos, HTML proporciona la estructura y organización necesarias para presentar contenido de manera efectiva en navegadores web. La evolución de HTML ha permitido la integración de elementos multimedia, la creación de páginas dinámicas e interactivas, y la optimización para motores de búsqueda, mejorando la accesibilidad y visibilidad de la información en línea.

En esta guía de práctica, exploraremos los fundamentos de HTML5 y su aplicación en el desarrollo web moderno. Aprenderemos a crear la estructura básica de una página web, utilizar etiquetas semánticas para dar significado al contenido, diseñar formularios interactivos, aplicar estilos CSS para mejorar la presentación visual y optimizar nuestras páginas para un mejor posicionamiento en buscadores.

Acompáñanos en este recorrido por el fascinante mundo de HTML5, donde descubrirás cómo este lenguaje de marcado da vida a la web y cómo puedes utilizarlo para crear experiencias digitales atractivas e informativas.

## 2. ¿QUÉ ES HTML?

Es un lenguaje de mercado que permite agregar características estructurales de organización y distribución de la información en la construcción de un documento que presenta y compone información.

El html viene de sus siglas en inglés hyper text mark language, Está basado en agregar etiquetas que identifican a un elemento dentro de un documento, dichas etiquetas son invisibles en la presentación, pero permiten agregar cierto tipo de características organizacionales al documento. (Rubiales Gomez, 2018)

Un documento web es visible a través de un visor, el visor más común es el navegador web de los cuáles hay en varias versiones y fabricantes.

Para transportar a través de la internet documentos web se utiliza el protocolo http el cual se encarga de gestionar la comunicación entre un servidor web y el cliente que posee al navegador que invoca la página web que se desea consumir. (Rubiales Gomez, 2018)

## 3. ESTRUCTURA BÁSICA HTML

De forma estructural un documento web está conformado por 2 componentes esenciales y estructuralmente principales:

cabecera o head:

contiene toda la información necesaria para presentar al documento web bajo diferentes contextos como su título, la información que habla sobre las características de la página o vinculaciones que puede tener a otros archivos. (Mardan, 2018)

cuerpo o body:

contiene todos los elementos de presentación de información que serán visibles a través del navegador web cuando será presentado al cliente que solicitó una página web.

## 4. ESTRUCTURA DE UNA ETIQUETA HTML

Las partes principales de nuestro elemento son las siguientes:

La etiqueta de apertura: Consiste en el nombre del elemento (en este caso, p), envuelto entre paréntesis angulares de apertura y cierre. Esto indica dónde comienza el elemento o comienza a surtir efecto; en este caso, donde comienza el párrafo. (Mozilla, 2022)

La etiqueta de cierre: es lo mismo que la etiqueta de apertura, excepto que incluye una barra

inclinada antes del nombre del elemento. Esto indica dónde termina el elemento; en este caso, donde termina el párrafo. No agregar una etiqueta de cierre es uno de los errores estándar de los principiantes y puede generar resultados extraños. (Matarazzo, 2021)

El contenido: este es el contenido del elemento, que en este caso es solo texto.

El elemento: la etiqueta de apertura, la etiqueta de cierre y el contenido forman el elemento.

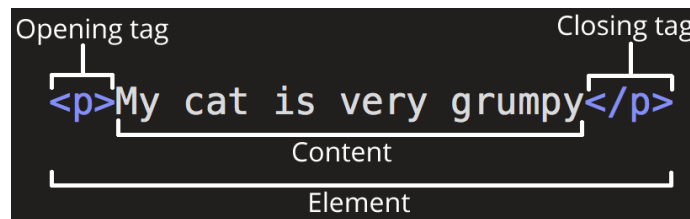


Figura N° 1: Escritura de la etiqueta o tag.

Los elementos también pueden tener atributos similares a los siguientes:

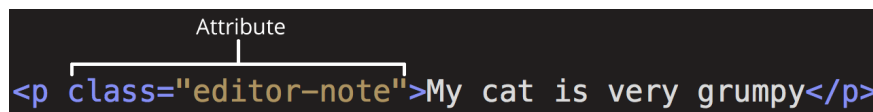


Figura N° 2: Atributo de una etiqueta.

Los atributos contienen información adicional sobre el elemento que no desea que aparezca en el contenido real. Aquí, class es el nombre del atributo y editor-note es el valor del atributo. El class atributo le permite dar al elemento un identificador no único que se puede usar para orientarlo (y cualquier otro elemento con el mismo class valor) con información de estilo y otras cosas. (Matarazzo, 2021)

Un atributo siempre debe tener lo siguiente:

- Un espacio entre éste y el nombre del elemento (o el atributo anterior, si el elemento ya tiene uno o más atributos).
- El nombre del atributo seguido de un signo igual.
- El valor del atributo envuelto por comillas de apertura y cierre.

## 5. ETIQUETAS SEMÁNTICAS Y NO SEMÁNTICAS

Semántica: Perteneciente o relativo a la significación de las palabras. Este tipo de etiquetas del lenguaje HTML son aquellas que dan un significado a las partes del documento.

Las etiquetas semánticas indican qué es el contenido que contienen, en lugar de como se debe formatear al mostrar el documento HTML. Estas etiquetas, tienen importancia en el marco del HTML y de la composición de un documento web para ayudar a los motores de búsqueda como Google a indexar más correctamente los contenidos de un sitio. (Mardan, 2018)

Dentro del etiquetado semántico también tenemos varias funciones, pero las principales y elementales son las que sirven para definir el esquema principal del documento, como HEADER, ARTICLE, FOOTER, etc. Generalmente, en cualquier documento tenemos una cabecera, un cuerpo y un pie de página, elementos que definen la estructura. Todas esas etiquetas semánticas nos indican qué es el contenido que engloban y cuál es su relación con el conjunto de elementos del documento HTML.

<header>: la cabecera de una web, ojo, no confundir con la etiqueta head.

<nav>: establece un apartado con enlaces de la página, por ejemplo, un menú.

<main>: expresa el contenido principal de un documento.

<section>; define una sección de un documento.

<article>: indica que el contenido encerrado dentro de las etiquetas es un artículo.

<aside>: el contenido lateral de una página.

<footer>: hace referencia al pie de una página o sección.

Las etiquetas semánticas no tienen un estilo predeterminado que el navegador nos vaya a

asignar. Es decir, porque HEADER signifique que es una cabecera, el navegador no va en ningún caso a posicionar el elemento en la parte de arriba del documento. Lo mismo con FOOTER, que no lo colocará en la parte de abajo, o ASIDE al lateral.

Nosotros mediante CSS, debemos asignar los estilos que queramos se aplique a cada uno de esos elementos del documento HTML.

**Elementos no semánticos:** las etiquetas como div, span se incluyen en las categorías no semánticas, ya que sus nombres no dicen nada sobre el tipo de contenido que contienen.

div Es un elemento a nivel de bloque o división de una sección. Se utiliza como recipiente.

```
<!DOCTYPE html>
<html>
  <head>
    <title>div Tag</title>
    <style>
      .GFG {
        color:#006400;
      }
    </style>
  </head>
  <body>
    <h1>div Tag</h1>
    <div class="GFG">
      <h1>GeeksforGeeks</h1>

    <p>GeeksforGeeks is a Computer Science portal</p>

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

## 6. CATEGORIZACIÓN DE ETIQUETAS

Cada elemento HTML es miembro de una o más categorías de contenido — estas categorías agrupan elementos que comparten características comunes. Esta es una agrupación de detalles flexible (en realidad no crea una relación entre los elementos de estas categorías), pero ayuda a definir y describir el comportamiento compartido de las categorías y sus reglas asociadas, especialmente cuando te encuentras con sus intrincados. También es posible que los elementos no sean miembros de ninguna de estas categorías. (Rubiales Gomez, 2018)

Hay tres tipos de categorías de contenido:

- Categoría de contenido principal — que describe las reglas comunes compartidas por muchos elementos.
- Categorías de contenido relacionado con formularios — que describen reglas comunes a los elementos relacionados con formularios.
- Categorías de contenido específico — que describen categorías raras compartidas solo por unos pocos elementos, a veces, solo en un contexto específico.

## 7. GITHUB PAGES

GitHub tiene un servicio de hosting gratis llamado GitHub Pages. Con él, puedes tener un repositorio alojado en GitHub y hacer que el contenido se muestre en la web en tiempo real.

Este es un sitio para nuestros proyectos donde lo único que tenemos que hacer es tener un repositorio alojado. En la página, podemos seguir las instrucciones para crear este repositorio

Pasos para subir un repositorio a GitHub Pages

- Debemos tomar la llave SSH y hacer un git clone #SSHexample en mi computador local (Home).
- Luego, accederemos a la carpeta nueva que aparece en nuestra máquina local.

- Creamos un nuevo archivo que se llame index.html
- Guardamos los cambios, hacemos un git pull y seguido de esto un git push a master.
- Vamos a las opciones de settings de este repositorio y, en la parte de abajo, en la columna Github Pages, configuramos el source o fuente para que traiga la rama master
- Guardamos los cambios.

Después de esto, podremos ver nuestro trabajo en la web como si tuviéramos nuestro propio servidor.

## 8. ETIQUETAS DE FORMULARIO Y VALIDACIÓN

Los formularios web son uno de los principales puntos de interacción entre un usuario y un sitio web o aplicación. Los formularios permiten a los usuarios la introducción de datos, que generalmente se envían a un servidor web para su procesamiento y almacenamiento (consulta Enviar los datos de un formulario más adelante en el módulo), o se usan en el lado del cliente para provocar de alguna manera una actualización inmediata de la interfaz (por ejemplo, se añade otro elemento a una lista, o se muestra u oculta una función de interfaz de usuario).

El HTML de un formulario web está compuesto por uno o más controles de formulario (a veces llamados widgets), además de algunos elementos adicionales que ayudan a estructurar el formulario general; a menudo se los conoce como formularios HTML. Los controles pueden ser campos de texto de una o varias líneas, cajas desplegadas, botones, casillas de verificación o botones de opción, y se crean principalmente con el elemento `<input>`, aunque hay algunos otros elementos que también hay que conocer. (Data, 2022)

Los controles de formulario también se pueden programar para forzar la introducción de formatos o valores específicos (validación de formulario), y se combinan con etiquetas de texto que describen su propósito para los usuarios con y sin discapacidad visual. (Mardan, 2018)

### A. DISEÑAR TU FORMULARIO

Antes de comenzar a escribir código, siempre es mejor dar un paso atrás y tomarte el tiempo necesario para pensar en tu formulario. Diseñar una maqueta rápida te ayudará a definir el conjunto de datos adecuado que deseas pedirle al usuario que introduzca. Desde el punto de vista de la experiencia del usuario (UX), es importante recordar que cuanto más grande es tu formulario, más te arriesgas a frustrar a las personas y perder usuarios. Tiene que ser simple y conciso: solicita solo los datos que necesitas.

Diseñar formularios es un paso importante cuando creas un sitio web o una aplicación. Va más allá del alcance de este artículo exponer la experiencia de usuario de los formularios, pero si deseas profundizar en ese tema, puedes leer los artículos siguientes:

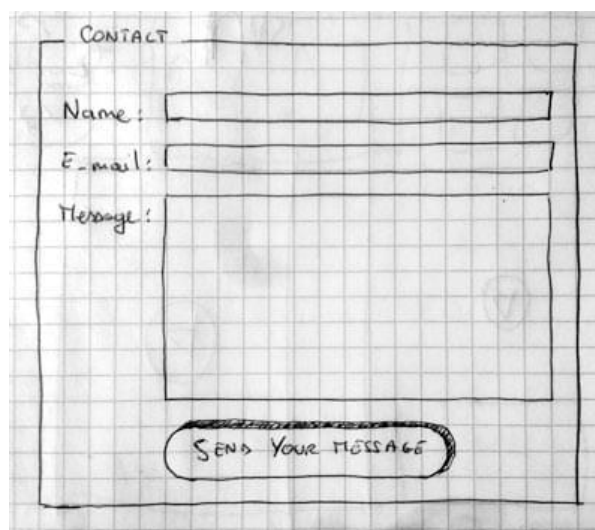


Figura N° 3: Esquema de Formulario Web.

Smashing Magazine tiene algunos artículos muy buenos sobre formularios UX, incluido el artículo —antiguo pero relevante— [An Extensive Guide To Web Form Usability](#) [Amplia guía de usabilidad para formularios web].

UXMatters también es un recurso que da buenos consejos, desde buenas prácticas básicas hasta cuestiones complejas como los formularios de varias páginas.

En este artículo, vamos a crear un formulario de contacto sencillo. Hagamos un esbozo.

Esbozo aproximado del formulario que vamos a construir

Nuestro formulario va a tener tres campos de texto y un botón. Le pedimos al usuario su nombre, su correo electrónico y el mensaje que desea enviar. Al pulsar el botón sus datos se enviarán a un servidor web.

## 9. TIPOS DE INPUT

Ahora veremos en detalle la funcionalidad de los controles de formulario más recientes, incluyendo algunos tipos de input nuevos, los cuales fueron añadidos en HTML5 para permitir la recolección de tipos de datos específicos. (Mardan, 2018)

Debido a que la apariencia de un control de formulario puede ser algo distinta con respecto a unas especificaciones del diseñador, los desarrolladores web a veces construyen sus propios controles de formulario personalizados. Cubrimos este aspecto en un tutorial avanzado: Cómo construir widgets de formulario personalizados.

### A. CAMPO DE DIRECCIÓN DE CORREO ELECTRÓNICO

Este tipo de campo se define utilizando el valor email en el atributo type del elemento <input>:

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

Cuando se utiliza este valor type, se le obliga al usuario a escribir dentro del campo una dirección de correo electrónico válida. Cualquier otro contenido ocasiona que el navegador muestre un mensaje de error cuando se envía el formulario. Puedes verlo en acción en la siguiente captura de pantalla

```
An invalid email input showing the message "Please enter an email address."
```

Puedes utilizar también el atributo multiple en combinación con el tipo inputemail para permitir que sean introducidas varias direcciones de correo electrónico separadas por comas en el mismo input:

```
<input type="email" id="email" name="email" multiple>
```

### B. VALIDACIÓN DEL LADO CLIENTE

Como puedes haber visto anteriormente, email, junto con otros tipos de input más recientes, proporciona la validación de errores en el lado cliente de forma predeterminada, realizados por el navegador antes de que los datos obtenidos se envíen al servidor. Es una ayuda útil guiar a los usuarios a rellenar un formulario de forma precisa y puede ahorrar tiempo: es útil saber de inmediato que tu dato no es correcto, en vez de tener que esperar el viaje de ida y vuelta al servidor.

Pero no debería ser considerado una medida de seguridad exhaustiva. Tus aplicaciones siempre deben realizar comprobaciones de seguridad en cada dato, tanto en el lado servidor como en el lado cliente debido a que la validación en el lado cliente es muy fácil desactivarla, por lo que usuarios malintencionados pueden enviar fácilmente datos incorrectos al servidor. Lee Seguridad en el sitio web para tener una idea de lo que podría ocurrir; Implementar validación en el lado servidor está más allá del alcance de este módulo-guía, pero debería tenerlo en cuenta. (Matarazzo, 2021)

Ten en cuenta que a@b es una dirección de correo electrónico válida de acuerdo a las

restricciones proporcionadas por defecto. Esto es debido a que el tipo de input email, permite por defecto direcciones de correo electrónico de una intranet. Para implementar un comportamiento diferente de validación puedes utilizar el atributo pattern, y también puedes utilizar mensajes de error personalizados; Hablaremos de cómo utilizar estas características en Validación de formularios en el lado cliente en un artículo posterior.

## C. CAMPO DE BÚSQUEDA

Los campos de búsqueda están destinados a ser utilizados para crear cajas de búsqueda en páginas y aplicaciones. Este tipo de campo se define utilizando el valor search en su atributo type:

```
<input type="search" id="search" name="search">
```

La diferencia principal entre un campo text y un campo search, es la forma en que el navegador aplica estilo a su apariencia. A menudo los campos search se muestran con bordes redondeados; y a veces también muestran una "X", el cual despeja el campo de cualquier valor cuando se pulsa sobre él. Adicionalmente, en dispositivos con teclado dinámico, la tecla enter del teclado puede leer "search" o mostrar un icono de lupa. (Mozilla, 2022)

Campo número de teléfono

Se puede crear un campo especial para introducir números de teléfono utilizando tel como valor del atributo type:

```
<input type="tel" id="tel" name="tel">
```

Cuando se accede desde un dispositivo táctil con teclados dinámicos, muchos de ellos mostrarán un teclado numérico cuando se encuentren con type="tel", lo que significa que este tipo es útil no sólo para ser utilizado para números de teléfono, sino también cuando sea útil un teclado numérico. (Rubiales Gomez, 2018)

Campo URL

Se puede crear un tipo especial de campo para introducir URLs utilizando el valor url para el atributo type:

```
<input type="url" id="url" name="url">
```

Este tipo añade restricciones de validación en el campo. El navegador informará de un error si no se introdujo el protocolo (como http:), o si de algún modo el URL está mal formado. En dispositivos con teclados dinámicos a menudo mostrará por defecto algunas o todas las teclas como los dos puntos, el punto, y la barra inclinada.

Campo numérico

Se pueden crear controles para introducir números con el type number de <input>. Este control se parece a un campo de texto, pero solo permite números de punto flotante, y normalmente proporciona botones deslizadores para incrementar o reducir el valor del control. En dispositivos con teclados dinámicos generalmente se muestra el teclado numérico.

Miremos algunos ejemplos. El primero de los siguientes crea un control numérico cuyo valor está restringido a cualquier valor entre 1 y 10, y sus botones cambian su valor en incrementos o decrementos de 2.

```
<input type="number" name="age" id="age" min="1" max="10" step="2">
```

El segundo crea un control numérico cuyo valor está restringido a cualquier valor entre el 0 y 1 ambos inclusive, y sus botones cambian su valor en incrementos o decrementos de 0.01.

```
<input type="number" name="change" id="pennies" min="0" max="1" step="0.01">
```

El tipo de input number tiene sentido cuando esté limitado el rango de valores válidos, por ejemplo, la edad de una persona o su altura. Si el rango es demasiado grande para que los cambios de incremento tengan sentido (por ejemplo, los códigos postales de USA, cuyo rango va de 00001 a 99999), entonces sería una mejor opción utilizar el tipo tel: proporciona el teclado numérico mientras que omite el componente de interfaz de usuario de los deslizadores de número. (Data, 2022)

## 10. INCLUSIÓN DE CARACTERES ESPECIALES EN HTML

En HTML, los caracteres <, >, ", ' y & son caracteres especiales. Forman parte de la sintaxis HTML. Entonces, ¿cómo incluir uno de estos caracteres especiales en tu texto? Por ejemplo, si desea utilizar un signo comercial o menor que, y no hacer que se interprete como código.

Haces esto con referencias de caracteres. Estos son códigos especiales que representan caracteres, para ser usados en estas circunstancias exactas. Cada referencia de carácter comienza con un signo de ampersand (&) y finaliza con un punto y coma (;).

Carácter literal	Equivalente de referencia de caracteres
<	<
>	>
"	"
'	'
&	&erio;

El equivalente de referencia de carácter podría recordarse fácilmente porque el texto que utiliza se puede ver como menor que para '<', cita para '"' y de manera similar para otros.

## 11. PATRONES REGULARES (REGEX)

Otra característica útil de validación es el atributo pattern, que espera una expresión regular como valor. Una expresión regular ( regex ) es un patrón que se puede usar para establecer combinaciones de caracteres en cadenas de texto, por lo que las expresiones regulares son ideales para la validación de formularios y sirven para una gran variedad de otros usos en JavaScript.

Las expresiones regulares son bastante complejas y no vamos a exponerlas exhaustivamente en este artículo. A continuación, hay algunos ejemplos para que te hagas una idea de cómo funcionan.

- a: coincide con un carácter que es a (ni b, ni aa, etc.).
- abc: coincide con a, seguido de b, seguido de c.
- ab?c: coincide con a, seguido opcionalmente de una sola b, seguido de c (aco abc).
- ab\*c: coincidir con a, seguido opcionalmente de cualquier número de b, seguido de c. ( ac, abc, abbbbbc, etc)
- a|b: coincide con un carácter que es a o b.
- abc|xyz: coincidir exactamente con abco xyz (pero no con abcxyz a o y, y así sucesivamente).

Hay muchas más posibilidades que no exponemos aquí. Para obtener una lista completa y muchos ejemplos, consulte nuestro documento de expresiones regulares.

Implementemos un ejemplo. Actualiza tu HTML para añadir un atributo pattern como este:

```
<form>
```



```
<label for="choose">¿Prefieres un plátano o una cereza?</label>
<input id="choose" name="i_like" required pattern="[Pp]látano|[Cc]ereza ">
<button>Enviar</button>
</form>
```

## 12. ¿QUÉ ES UNA TABLA?

Es un conjunto estructurado de datos formado por filas y columnas (datos tabulares). Una tabla le permite buscar rápida y fácilmente valores que indican algún tipo de conexión entre diferentes tipos de datos, por ejemplo, una persona y su edad, o un día de la semana, o el horario de una piscina local. (Rubiales Gomez, 2018)

Una tabla permite buscar con rapidez y facilidad valores entre diferentes tipos de datos que indiquen algún tipo de conexión. Por ejemplo, una persona y su edad, o un día de la semana o el horario de una piscina municipal.

Una tabla de datos de ejemplo que muestra los nombres y las edades de algunas personas: Chris 38, Dennis 45, Sarah 29, Karen 47.

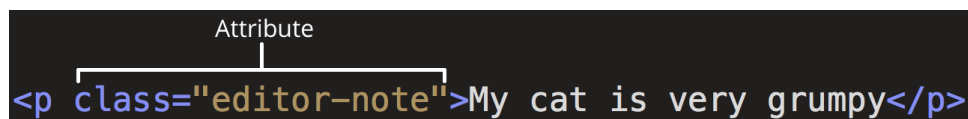
Por lo tanto, no es de extrañar que los creadores de HTML proporcionen un medio con el que estructurar y presentar datos en tablas en la web.

## 13. ¿CUÁNDO USAR TABLAS?

Es necesario hacer uso de tablas cuando la información tiene que organizarse de forma estructural mostrando sí en una organización de filas y columnas, debemos tener en cuenta que la información que se coloca en una columna debe ser homogénea refiriéndose al mismo valor informativo, en cambio la información que ven una fila puede ser heterogénea pues combina diferentes valores informativos que explican un contexto completo sobre el ítem o individuo al cual hacemos referencia al principio de la fila.

## 14. ATRIBUTOS

Los elementos también pueden tener atributos. Los atributos tienen este aspecto:



```
<p class="editor-note">My cat is very grumpy</p>
```

Figura N° 4: Ejemplo de “atributo” en un declaración HTML.

Los atributos contienen información extra sobre el elemento que no se muestra en el contenido. En este caso, el atributo class asigna al elemento un identificador que se puede utilizar para dotarlo de información de estilo.

Un atributo debería tener:

- Un espacio entre este y el nombre del elemento. (Para un elemento con más de un atributo, los atributos también deben estar separados por espacios).
- El nombre del atributo, seguido por un signo igual.
- Un valor del atributo, rodeado de comillas de apertura y cierre.

## 15. ¿QUÉ ES SEO?

Es la abreviación de Search Engine Optimization (optimización en motores de búsqueda). Es el conjunto de técnicas y estrategias centradas en optimizar el posicionamiento orgánico en buscadores de internet. Algunos ejemplos son Google, Bing, Baidu, Yahoo!, Yandex, DuckDuckGo o YouTube. (Matarazzo, 2021)

Contar con una página web para tu marca o empresa es la base de cualquier presencia digital, pero el esfuerzo de crearla no sirve de nada si no te aseguras de que tu audiencia la encuentre. Por eso, es imprescindible hacer que Google funcione a tu favor.

## 16. METADATA

La etiqueta <meta> define metadatos sobre un documento HTML. Los metadatos son datos (información) sobre datos.

Las etiquetas <meta> siempre van dentro del elemento <head> y normalmente se utilizan para especificar el juego de caracteres, la descripción de la página, las palabras clave, el autor del documento y la configuración de la ventana gráfica.

Los metadatos no se mostrarán en la página, pero son analizables por máquina.

Los navegadores utilizan metadatos (cómo mostrar contenido o recargar una página), motores de búsqueda (palabras clave) y otros servicios web.

Existe un método para permitir que los diseñadores web tomen el control de la ventana gráfica (el área visible del usuario de una página web), a través de la <meta>etiqueta (consulte el ejemplo "Configuración de la ventana gráfica" a continuación).

```
<head>
  <meta charset="UTF-8">
  <meta name="description" content="Free Web tutorials">
  <meta name="keywords" content="HTML, CSS, JavaScript">
  <meta name="author" content="John Doe">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
</head>
```

## 17. OPEN GRAPH

Es un protocolo de Internet creado originalmente por Facebook para estandarizar el uso de metadatos dentro de una página web para representar el contenido de una página.

Dentro de él, puede proporcionar detalles tan simples como el título de una página o tan específicos como la duración de un video. Todas estas piezas encajan para formar una representación de cada página individual de Internet.

Si bien debería ser bastante sencillo, aquí hay un desglose de lo que significa cada una de las etiquetas:

og:title: El título de su página. Por lo general, es la misma que la <title>etiqueta de su página web, a menos que desee presentarla de manera diferente.

og:type: El "tipo" de sitio web que tienes. Explicaré más en la siguiente sección, aunque un "tipo" genérico es "sitio web".

og:image: Debe ser un enlace a una imagen que le gustaría representar su contenido. Debe ser una imagen de alta resolución que las redes sociales usarán en sus feeds.

og:url: Esta debería ser la URL de la página actual.

Al colocar una etiqueta en su sitio web, debe colocarla <head>junto con cualquier otro metadato. La etiqueta utilizada será una <meta>etiqueta y debería tener este patrón:

```
<meta property="[NAME]" content="[VALUE]" />
```

## 18. TWITTER CARD

Son un formato enriquecido es el de agrega a un documento web para que puedan destacar o sobresalir en el navegador Cuando se vincula Twitter.

twitter:card: esta etiqueta especifica el tipo de Twitter Card que se debe mostrar . El tipo summary\_large\_image muestra un breve resumen con una vista previa de imagen grande.

twitter:sitio: su nombre de usuario de Twitter, o el nombre de usuario de su sitio o empresa

¿Cómo agrego una Twitter Card a mi sitio web?

- Elija un tipo de tarjeta para implementar.
- Agregue las etiquetas meta correctas a la página.
- Ejecute la URL a través de la herramienta de validación para probar.
- Después de probar en el validador o aprobación de su tarjeta de jugador, twittee la URL y vea que la tarjeta aparece debajo de su tweet en la vista de detalles.

## 19. RESUMEN

El documento explora el lenguaje HTML5, fundamental en la creación de páginas web. HTML, o Lenguaje de Marcado de Hipertexto, permite estructurar y organizar la información en un documento web, el cual es visualizado a través de navegadores. La estructura básica de un documento HTML consta de dos partes principales: la cabecera (<head>) que contiene metadatos como el título y la información sobre la página, y el cuerpo (<body>) donde se presenta el contenido visible al usuario.

También detalla las etiquetas HTML, que son elementos clave para definir la estructura y el significado del contenido. Las etiquetas semánticas, como <header>, <nav> y <article>, describen el propósito del contenido que encierran, mejorando la accesibilidad y la comprensión tanto para los usuarios como para los motores de búsqueda. Por otro lado, las etiquetas no semánticas, como <div> y <span>, se utilizan principalmente para fines de diseño y estructuración visual.

Además, aborda temas como la validación de formularios, el uso de tablas para organizar datos, la optimización para motores de búsqueda (SEO) y la inclusión de metadatos para mejorar la presentación de las páginas en redes sociales. Se destaca la importancia de utilizar etiquetas y atributos de manera adecuada para crear páginas web bien estructuradas, accesibles y optimizadas para una mejor experiencia de usuario y visibilidad en línea.

En resumen, proporciona una visión general de los conceptos esenciales de HTML5, desde la estructura básica de un documento hasta técnicas avanzadas para mejorar la presentación y el rendimiento de las páginas web. El enfoque en la semántica, la accesibilidad y la optimización para motores de búsqueda refleja las mejores prácticas actuales en el desarrollo web, preparando a los estudiantes para crear contenido efectivo y relevante en el entorno digital.

## IV

### (La práctica tiene una duración de 4 horas) ACTIVIDADES

#### 1. EXPERIENCIA DE PRÁCTICA N° 01:

Conforme un grupo tal como lo indica el docente.

Crear una cuenta de GitHub donde creará y registrará todos sus proyectos:

##### a) CREANDO TU PRIMER ELEMENTO HTML

1. Edita la siguiente línea en el área Entrada envolviéndola con las etiquetas <em>y </em>. Para abrir el elemento, coloca la etiqueta de apertura <em>al principio de la línea. Para cerrar el elemento, coloca la etiqueta de cierre </em>al final de la línea.

##### b) AÑADIR ATRIBUTOS A UN ELEMENTO

1. Otro ejemplo de un elemento es <a>. Esto significa ancla. Un ancla puede convertir el texto que encierra en un hipervínculo. Las anclas pueden tener varios atributos, pero varios son como sigue:
  - a) href: El valor de este atributo indica la dirección web a la que se quiere que apunte el enlace, que será hacia donde nos lleve el navegador cuando se haga clic sobre el elemento. Por ejemplo, href="https://www.mozilla.org/".
  - b) title: El atributo title añade información adicional sobre el enlace, como puede ser el título

de la página que vinculas. Por ejemplo, title="La página de inicio de Mozilla". Esta información aparecerá cuando se le pase el ratón por encima.

- c) target: El atributo target especifica el contexto de navegación que va a usar para mostrar el enlace. Por ejemplo, target="\_blank" abrirá el enlace en una nueva pestaña. Si quieres mostrar el enlace en la pestaña activa, simplemente omite este atributo.
2. Edita la línea de abajo en el área de Entrada para convertirlo en un enlace a tu sitio web favorito.
  3. Añade el elemento <a>.
  4. Añade el atributo href y el atributo title.
  5. Especifica el atributo target para abrir el enlace en una nueva pestaña.
  6. Los cambios se actualizarán inmediatamente en la zona de Salida. Deberías ver un enlace que mostrará el contenido del atributo title cuando pases el ratón encima, y que te llevará a la dirección web indicada por el atributo href cuando hagas clic. Recuerda que debes incluir un espacio entre el nombre del elemento y cada atributo.

### c) AÑADIR ALGUNAS CARACTERÍSTICAS A UN DOCUMENTO HTML

Si quieres escribir algo de HTML en tu ordenador local para experimentar, puedes:

1. Copiar el ejemplo de la página HTML del punto anterior.
2. Crear un archivo nuevo en un editor de texto.
3. Pegar el código en el nuevo archivo de texto.
4. Guardar el archivo como index.html.

## 2. EXPERIENCIA DE PRÁCTICA N° 02: FORMULARIO WEB

### a) IMPLEMENTANDO NUESTRO FORMULARIO HTML

1. La implementación de nuestro formulario HTML
2. De acuerdo, intentemos crear el HTML para nuestro formulario. Vamos a utilizar los elementos HTML siguientes: <form>, <label>, <input>, <textarea> y <button>.
3. Antes de continuar, haz una copia local de nuestra plantilla HTML simple: introduce aquí tu formulario HTML.

### b) EL ELEMENTO <FORM>

1. Todos los formularios comienzan con un elemento <form>, como este:

```
<form action="/my-handling-form-page" method="post">

</form>
```

2. Este elemento define formalmente un formulario. Es un elemento contenedor, como un elemento <section> o <footer>, pero específico para contener formularios; también admite algunos atributos específicos para la configuración de la forma en que se comporta el formulario. Todos sus atributos son opcionales, pero es una práctica estándar establecer siempre al menos los atributos action y method:
3. El atributo action define la ubicación (URL) donde se envían los datos que el formulario ha recopilado cuando se validan.
4. El atributo method define con qué método HTTP se envían los datos (generalmente get o post).
5. Nota: Veremos cómo funcionan esos atributos en nuestro artículo Enviar los datos de un formulario que encontrarás más adelante.
6. Por ahora, añade el elemento <form> anterior a tu elemento HTML <body>.

### c) LOS ELEMENTOS <LABEL>, <INPUT> Y <TEXTAREA>

1. Nuestro formulario de contacto no es complejo: la parte para la entrada de datos contiene tres campos de texto, cada uno con su elemento <label> correspondiente:
2. El campo de entrada para el nombre es un campo de texto de una sola línea.
3. El campo de entrada para el correo electrónico es una entrada de datos de tipo correo electrónico: un campo de texto de una sola línea que acepta solo direcciones de correo

electrónico.

4. El campo de entrada para el mensaje es <textarea>; un campo de texto multilínea.
5. En términos de código HTML, para implementar estos controles de formulario necesitamos algo como lo siguiente:

```
<form action="/my-handling-form-page" method="post">
  <ul>
    <li>
      <label for="name">Nombre:</label>
      <input type="text" id="name" name="user_name">
    </li>
    <li>
      <label for="mail">Correo electrónico:</label>
      <input type="email" id="mail" name="user_mail">
    </li>
    <li>
      <label for="msg">Mensaje:</label>
      <textarea id="msg" name="user_message"></textarea>
    </li>
  </ul>
</form>
```

6. Actualiza el código de tu formulario para que se vea como el anterior.
7. Los elementos <li> están ahí para estructurar nuestro código convenientemente y facilitar la aplicación de estilo (ver más adelante en el artículo). Por motivos de usabilidad y accesibilidad incluimos una etiqueta explícita para cada control de formulario. Ten en cuenta el uso del atributo for en todos los elementos <label>, que toma como valor el id del control de formulario con el que está asociado; así es como asocias un formulario con su etiqueta.
8. Hacer esto presenta muchas ventajas porque la etiqueta está asociada al control del formulario y permite que los usuarios con ratón, panel táctil y dispositivos táctiles hagan clic en la etiqueta para activar el control correspondiente, y también proporciona accesibilidad con un nombre que los lectores de pantalla leen a sus usuarios.
9. En el elemento <input>, el atributo más importante es type. Este atributo es muy importante porque define la forma en que el elemento <input> aparece y se comporta.
10. En nuestro ejemplo sencillo, usamos el valor <input/text> para la primera entrada, el valor predeterminado para este atributo. Representa un campo de texto básico de una sola línea que acepta cualquier tipo de entrada de texto.
11. Para la segunda entrada, usamos el valor <input/email>, que define un campo de texto de una sola línea que solo acepta una dirección de correo electrónico. Esto convierte un campo de texto básico en una especie de campo «inteligente» que efectúa algunas comprobaciones de validación de los datos que el usuario escribe. También hace que aparezca un diseño de teclado más apropiado para introducir direcciones de correo electrónico (por ejemplo, con un símbolo @ por defecto) en dispositivos con teclados dinámicos, como teléfonos inteligentes. Encontrarás más información sobre la validación de formularios en el artículo de Validación de formularios por parte del cliente más adelante.
12. Por último, pero no por ello menos importante, ten en cuenta la sintaxis de <input> en contraposición con la de <textarea></textarea>. Esta es una de las rarezas del HTML. La etiqueta <input> es un elemento vacío, lo que significa que no necesita una etiqueta de cierre. El elemento <textarea> no es un elemento vacío, lo que significa que debe cerrarse con la etiqueta de cierre adecuada. Esto tiene un impacto en una característica específica de los formularios: el modo en que defines el valor predeterminado. Para definir el valor predeterminado de un elemento <input>, debes usar el atributo value de esta manera:

```
<input type="text" value="por defecto este elemento se llena con este texto">
```

13. Por otro lado, si deseas definir un valor predeterminado para un elemento <textarea>, lo colocas entre las etiquetas de apertura y cierre del elemento <textarea>, así:

```
<textarea>
Por defecto, este elemento contiene este texto
</textarea>
```

**d) EL ELEMENTO <BUTTON>**

1. El marcado de nuestro formulario está casi completo; solo necesitamos añadir un botón para permitir que el usuario envíe sus datos una vez que haya completado el formulario. Esto se hace con el elemento <button>; añade lo siguiente justo encima de la etiqueta de cierre </form>:

```
<li class="button">
  <button type="submit">Envíe su mensaje</button>
</li>
```

2. El elemento <button> también acepta un atributo de type, que a su vez acepta uno de estos tres valores: submit, reset o button.
3. Un clic en un botón submit (el valor predeterminado) envía los datos del formulario a la página web definida por el atributo action del elemento <form>.
4. Un clic en un botón reset restablece de inmediato todos los controles de formulario a su valor predeterminado. Desde el punto de vista de UX, esto se considera una mala práctica, por lo que debes evitar usar este tipo de botones a menos que realmente tengas una buena razón para incluirlos.
5. Un clic en un botón button no hace... ¡nada! Eso suena tonto, pero es muy útil para crear botones personalizados: puedes definir su función con JavaScript.  
Nota: También puedes usar el elemento <input> con el atributo type correspondiente para generar un botón, por ejemplo <input type="submit">. La ventaja principal del elemento <button> es que el elemento <input> solo permite texto sin formato en su etiqueta, mientras que el elemento <button> permite contenido HTML completo, lo que permite generar botones creativos más complejos.

**e) APLICAR ESTILO BÁSICO A UN FORMULARIO**

1. Ahora que has terminado de escribir el código HTML de tu formulario, guárdalo y observa lo que ocurre en un navegador. Por ahora, se verá bastante feo.  
Nota: Si crees que no has escrito bien el código HTML, compáralo con nuestro ejemplo final: véase first-form.html (ver en vivo).
2. Resulta notablemente difícil aplicar estilo a los formularios. Está más allá del alcance de este artículo enseñarte cómo aplicar estilo a los formularios en detalle, por lo que por el momento solo vamos a exponer cómo añadir un poco de CSS para que se vea un poco bien.
3. En primer lugar, añade un elemento <style> a tu página, dentro de la cabecera del HTML. Debe quedar así:

```
<style>

</style>
```

4. Dentro de las etiquetas style, añade el código CSS siguiente:

```
form {
  /* Centrar el formulario en la página */
  margin: 0 auto;
  width: 400px;
  /* Esquema del formulario */
  padding: 1em;
  border: 1px solid #CCC;
  border-radius: 1em;
}

ul {
  list-style: none;
  padding: 0;
  margin: 0;
}

form li + li {
```

```
margin-top: 1em;
}

label {
  /* Tamaño y alineación uniforme */
  display: inline-block;
  width: 90px;
  text-align: right;
}

input,
textarea {
  /* Para asegurarse de que todos los campos de texto tienen la misma configuración
  de letra
  Por defecto, las áreas de texto tienen un tipo de letra monoespaciada */
  font: 1em sans-serif;

  /* Tamaño uniforme del campo de texto */
  width: 300px;
  box-sizing: border-box;

  /* Hacer coincidir los bordes del campo del formulario */
  border: 1px solid #999;
}

input:focus,
textarea:focus {
  /* Destacado adicional para elementos que tienen el cursor */
  border-color: #000;
}

textarea {
  /* Alinear los campos de texto multilínea con sus etiquetas */
  vertical-align: top;

  /* Proporcionar espacio para escribir texto */
  height: 5em;
}

.button {
  /* Alinear los botones con los campos de texto */
  padding-left: 90px; /* mismo tamaño que los elementos de la etiqueta */
}

button {
  /* Este margen adicional representa aproximadamente el mismo espacio que el
  espacio
  entre las etiquetas y sus campos de texto */
  margin-left: .5em;
}
```

5. Guarda y vuelve a cargar, y observa que tu formulario presenta un aspecto mucho menos feo. Nota: Puedes encontrar nuestra versión en GitHub en [first-form-styled.html](#) (ver en vivo).

#### f) ENVÍO DE DATOS DE FORMULARIO A SU SERVIDOR WEB

1. La última parte, y quizás la más complicada, es manejar los datos del formulario en el lado del servidor. El `<form>` elemento define dónde y cómo enviar los datos gracias a los atributos `action` y `method`
2. Proporcionamos un `name` atributo para cada control de formulario. Los nombres son importantes tanto en el lado del cliente como en el del servidor; le dicen al navegador qué nombre dar a cada dato y, en el lado del servidor, dejan que el servidor maneje cada dato por nombre. Los datos del formulario se envían al servidor como pares de nombre/valor.
3. Para nombrar los datos en un formulario, debe usar el `name` atributo en cada widget de

formulario que recopilará un dato específico. Veamos de nuevo algunos de nuestros códigos de formulario:

```
<form action="/my-handling-form-page" method="post">
  <ul>
    <li>
      <label for="name">Name:</label>
      <input type="text" id="name" name="user_name" />
    </li>
    <li>
      <label for="mail">E-mail:</label>
      <input type="email" id="mail" name="user_email" />
    </li>
    <li>
      <label for="msg">Message:</label>
      <textarea id="msg" name="user_message"></textarea>
    </li>
  </ul>
  ...
</form>
```

4. En nuestro ejemplo, el formulario enviará 3 datos llamados " user\_name", " user\_email" y " user\_message". Esos datos serán enviados a la URL " /my-handling-form-page" utilizando el método HTTP .POST
  5. Del lado del servidor, el script en la URL " /my-handling-form-page" recibirá los datos como una lista de 3 elementos clave/valor contenidos en la solicitud HTTP. La forma en que este script manejará esos datos depende de usted. Cada lenguaje del lado del servidor (PHP, Python, Ruby, Java, C#, etc.) tiene su propio mecanismo de manejo de datos de formulario. Está más allá del alcance de esta guía profundizar en ese tema, pero si desea saber más, proporcionamos algunos ejemplos en nuestro artículo Envío de datos de formulario más adelante.
  6. Publique y visualice en el navegador la página web que ha construido.
- Usemos colspan y rowspan para mejorar esta tabla.
7. Primero, haz una copia local de nuestros archivos animals-table.html y minimal-table.css en un directorio nuevo de tu ordenador. El HTML contiene el mismo ejemplo sobre perros que viste arriba.
  8. Luego, usa colspan para extender las celdas «Animales», «Hipopótamo» y «Cocodrilo» dos columnas más allá.
  9. Por último, usa rowspan para extender las celdas de «Caballo» y «Pollo» dos filas más abajo.
  10. Guarda tu código y ábrelo en un navegador para ver la mejora.
  11. Nota: Puedes encontrar nuestro ejemplo terminado en animals-table-fixed.html en GitHub (o consultarlo en vivo).

### 3. EXPERIENCIA DE PRÁCTICA N° 03: TABLAS

#### a) CREA TU PRIMERA TABLA

Ya hemos hablado bastante sobre la teoría de las tablas, así que veamos un ejemplo práctico y construyamos una tabla simple.

1. En primer lugar, haz una copia local de blank-template.html y minimal-table.css en un directorio nuevo de tu ordenador.
2. El contenido de cada tabla está delimitado entre estas dos etiquetas: <table></table>. Añádelas al cuerpo de tu código HTML.
3. El contenedor más pequeño dentro de una tabla es una celda, que se crea con un elemento <td> ('td' significa 'table data', datos de tabla). Añade lo siguiente dentro de tus etiquetas de tabla:

```
<td>Hola, soy tu primera celda.</td>
```

4. Si quieres una fila de cuatro celdas, tienes que copiar estas etiquetas tres veces. Actualiza el contenido de la tabla para que se vea así:



```
<td>Hola, soy tu primera celda.</td>
<td>Soy tu segunda celda.</td>
<td>Soy tu tercera celda.</td>
<td>Soy tu cuarta celda.</td>
```

- Como verás, las celdas no se colocan una debajo de la otra, sino que se alinean automáticamente entre sí en la misma fila. Cada elemento `<td>` crea una sola celda, y juntas forman la primera fila. Cada celda que agregamos hace crecer la fila.
- Para detener el crecimiento de esta fila y comenzar a colocar las celdas posteriores en una segunda fila, necesitamos usar el elemento `<tr>` ('tr' significa 'table row', fila de tabla). Vamos a verlo en detalle.
- Coloca las cuatro celdas que has creado dentro de las etiquetas `<tr>`, de esta forma:

```
<tr>
  <td>Hola, soy tu primera celda.</td>
  <td>Soy tu segunda celda.</td>
  <td>Soy tu tercera celda.</td>
  <td>Soy tu cuarta celda.</td>
</tr>
```

- Ahora que has hecho una fila, intenta hacer una o dos más: cada fila debe estar delimitada por un elemento `<tr>` adicional, con cada celda contenida en un `<td>`.
- Esto debería dar como resultado una tabla similar a la siguiente:

Hola, soy tu primera celda.	Soy tu segunda celda.	Soy tu tercera celda.	Soy tu cuarta celda.
Segunda fila, primera celda.	Celda 2.	Celda 3.	Celda 4.

**Tabla N° 1: Tabla de datos a programar.**

## b) APRENDIZAJE ACTIVO: ENCABEZADOS DE TABLA

Intentemos mejorar esta tabla.

- Primero, haz una copia local de nuestros archivos `dogs-table.html` y `minimal-table.css` en un directorio nuevo de tu ordenador. El HTML contiene el mismo ejemplo sobre perros que viste arriba.
- Para reconocer los encabezados de la tabla como encabezados, tanto visual como semánticamente, puedes usar el elemento `<th>` ('th' significa 'table header', encabezado de tabla). Funciona exactamente igual que un `<td>`, excepto que denota un encabezado, no una celda normal. Entra en el código HTML, y cambiar todos los elementos `<td>` que delimitan los encabezados de tabla por elementos `<th>`.
- Guarda tu HTML y cárgalo en un navegador. Los encabezados deberían verse como tal.

## c) PROPORCIONAR UN ESTILO COMÚN A LAS COLUMNAS.

Hay una última característica de la que queremos hablar en este artículo antes de continuar. El HTML tiene un método para definir información de estilo para una columna completa de datos en un solo lugar: los elementos `<col>` y `<colgroup>`. Estos atributos existen porque especificar el estilo de las columnas puede resultar enojoso e ineficiente; en general hay que especificar la información de estilo en cada `<td>` o `<th>` de la columna, o utilizar un selector complejo como `:nth-child()` (en-US).

Tomemos el ejemplo sencillo siguiente:

```
<table>
  <tr>

  <th>Dato 1</th>
    <th style="background-color: yellow">Dato 2</th>
  </tr>
</tr>
```

```

    <td>Calcuta</td>
    <td style="background-color: yellow">Pizza</td>
  </tr>
  <tr>
    <td>Robots</td>
    <td style="background-color: yellow">Jazz</td>
  </tr>
</table>

```

4. Esto nos da el resultado siguiente:

```

Dato 1  Dato 2
Calcuta Naranja
Robots  Jazz

```

Esto no es ideal, porque hay que repetir la información de estilo en las tres celdas de la columna (en un proyecto real probablemente habría definida una clase `class` en las tres celdas y el estilo se especificaría en una hoja de estilo por separado). En vez de hacer esto, podemos especificar la información una sola vez, con un elemento `<col>`. Los elementos `<col>` se especifican dentro de un contenedor `<colgroup>` justo debajo de la etiqueta de apertura `<table>`. Podríamos crear el mismo efecto que vemos arriba especificando nuestra tabla de la manera siguiente:

```

<table>
  <colgroup>
    <col>
    <col style="background-color: yellow">
  </colgroup>
  <tr>

<th>Dato 1</th>

<th>Dato 2</th>
  </tr>
  <tr>
    <td>Calcuta</td>
    <td>Pizza</td>
  </tr>
  <tr>
    <td>Robots</td>
    <td>Jazz</td>
  </tr>
</table>

```

5. En efecto, definimos dos tipos de «columnas de estilo», una que especifica la información para la aplicación de estilo en cada columna. No aplicamos estilo a la primera columna, sino que aún tenemos que incluir un elemento `<col>` en blanco; de lo contrario, el estilo también se aplicaría a la primera columna.
6. Si quisiéramos aplicar la información de estilo a ambas columnas, podríamos incluir un elemento `<col>` con un atributo `span`, como este:

```

<colgroup>
  <col style="background-color: yellow" span="2">
</colgroup>

```

Al igual que `colspan` y `rowspan`, `span` toma un valor numérico sin unidades que especifica el número de columnas a las que se desea aplicar el estilo.

#### d) APRENDIZAJE ACTIVO: COLGROUP Y COL

A continuación, puedes ver el horario de una profesora de idiomas. El viernes tiene que enseñar holandés todo el día, pero también enseña alemán durante unas horas los martes y los jueves,

y quiere resaltar las columnas que contienen los días que da clase.

## School timetable

	Mon	Tues	Wed	Thurs	Fri	Sat	Sun
1st period	English			German	Dutch		
2nd period	English	English		German	Dutch		
3rd period		German		German	Dutch		
4th period		English		English	Dutch		

Tabla N° 2: Tabla de horario de cursos de idiomas.

1. Recrea la tabla a partir de los pasos siguientes.
2. Primero, haz una copia local de nuestro archivo timetable.html en un directorio nuevo de tu ordenador. El HTML contiene la misma tabla que viste arriba, menos la información de estilo de las columnas.
3. Añade un elemento `<colgroup>` en la parte superior de la tabla, justo debajo de la etiqueta `<table>`, en la que puedes añadir tus elementos `<col>` (consulta los pasos restantes a continuación).
4. Las dos primeras columnas deben dejarse sin estilo.
5. Añade un color de fondo a la tercera columna. El valor para tu atributo de style es `background-color:#97DB9A`;
6. Establece un ancho distinto para la cuarta columna. El valor de tu atributo de style es `width: 42px`;
7. Añade un color de fondo a la quinta columna. El valor para tu atributo de style es `background-color:#97DB9A`;
8. Añade un color de fondo diferente más un borde a la sexta columna, para indicar que este es un día especial porque da clases de un idioma diferente. Los valores para tu atributo de style son `background-color:#DCC48E; border:4px solid #C1437A`;
9. Los últimos dos días los tiene libres, así que no establezcas ningún color de fondo, pero sí un valor para el ancho; el valor para el atributo de style es `width: 42px`;

## 2. EXPERIENCIA DE PRÁCTICA N° 03: METADATA

### a) ANÁLISIS DE UN EJEMPLO SENCILLO

1. Para comenzar este aprendizaje activo, te proponemos ir a nuestro repositorio de GitHub y descargues una copia de nuestra página `title-example.html` (<https://github.com/mdn/learning-area/blob/main/html/introduction-to-html/the-html-head/title-example.html>). Lo puedes hacer de las siguientes maneras:
  - a) Copia y pega el código de la página en un archivo de texto nuevo en tu editor de código, luego guárdalo en un lugar conveniente.
  - b) Presiona el botón "Raw" en la página de GitHub, lo cual hace que aparezca el código sin procesar (posiblemente en una nueva pestaña del navegador). A continuación, en el menú de tu navegador elige Archivo → Guardar página como... y selecciona un lugar adecuado para guardar el archivo.
2. Ahora abre el archivo en tu navegador. Deberías ver algo como esto:



Figura N° 5: Elemento HTML.

3. Una sencilla página web con el título configurado a <title> element, y el <h1> configurado a <h1> element. Ahora debería quedar claro dónde aparece el contenido de <h1> y dónde aparece el contenido de <title>.
4. También podrías probar a abrir el código en tu editor de código, editar el contenido de estos elementos y luego actualizar la página en tu navegador. Juega un poco con ello.
5. El contenido del elemento <title> también se usa de otras formas. Por ejemplo, si intentas marcar la página como favorita (Marcadores → Marcar esta página, o el icono en forma de estrella de Firefox), verás que el nombre que se sugiere para el marcado es el contenido del elemento <title>.

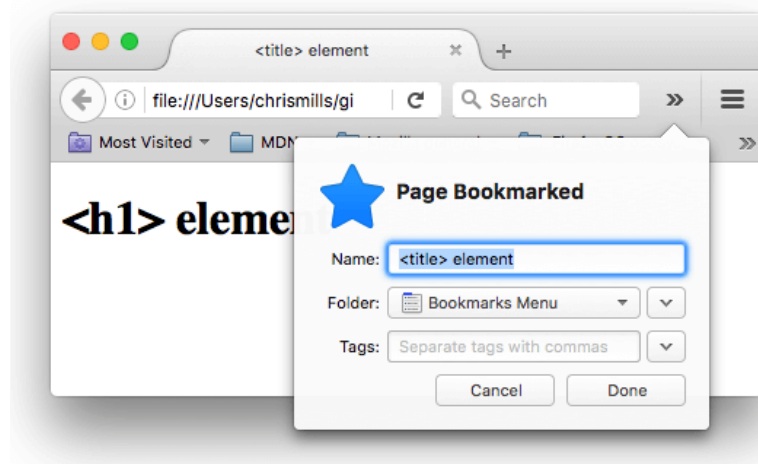


Figura N° 6: Título de la Página Web.

## b) EXPERIMENTA CON LA CODIFICACIÓN DE CARACTERES

1. Para probar esto, vuelve a visitar la plantilla HTML simple que obtuviste en la sección anterior sobre <title> (la página title-example.html) e intenta cambiar el valor de la propiedad meta charset por ISO-8859-1 y añade el japonés a tu página. Este es el código que usamos:

```
<p>Ejemplo en japonés: ご飯が熱い。</p>
```

## c) EL USO DE LA DESCRIPCIÓN EN LOS MOTORES DE BÚSQUEDA

La descripción también se usa en las páginas de resultados del motor de búsqueda. Repasemos un ejercicio para explorar esto:

1. Ve a la página de inicio de Mozilla Developer Network.
2. Observa el código fuente de la página (Ctrl+clic o clic con el botón derecho en la página y selecciona la opción del menú Ver código fuente de la página).
3. Encuentra la etiqueta del metadato description. Se verá más o menos así (aunque puede

cambiar con el tiempo):

```
<meta name="description" content="The Mozilla Developer Network (MDN) proporciona información sobre tecnologías de código abierto que incluyen HTML, CSS y APIs tanto para sitios web como para aplicaciones HTML5. También documenta productos Mozilla como el sistema operativo Firefox.">
```

4. Ahora busca "Mozilla Developer Network" en tu motor de búsqueda favorito (nosotros hemos utilizado Google). Observarás que efectivamente merece la pena que tener el contenido de la descripción <meta> y el elemento <title> que se utiliza en la búsqueda.

Resultado de búsqueda en Yahoo para "Mozilla Developer Network"

#### d) EL USO DE OPEN GRAPH

Para convertir sus páginas web en objetos gráficos, debe agregar metadatos básicos a su página. Hemos basado la versión inicial del protocolo en RDFa, lo que significa que colocará <meta>etiquetas adicionales en la parte superior <head>de su página web. Las cuatro propiedades requeridas para cada página son:

og:title.

og:type.

og:image.

og:url.

#### e) EL USO DE TWITTER CARDS

1. Elija un tipo de tarjeta para implementar.
2. Agregue las etiquetas meta correctas a la página.
3. Ejecute la URL a través de la herramienta de validación para probar.
4. Después de probar en el validador o aprobación de su tarjeta de jugador, twittee la URL y vea que la tarjeta aparece debajo de su tweet en la vista de detalles.

## V

### EJERCICIOS PROPUESTOS

1. Diseño de un portafolio personal: Crea una página web que sirva como tu portafolio profesional, utilizando etiquetas semánticas para estructurar el contenido (por ejemplo, <header>, <nav>, <main>, <section>, <article> y <footer>). Incluye información sobre tus habilidades, experiencia laboral y proyectos destacados. Aplica estilos CSS para lograr una presentación visual atractiva y asegúrate de que la página sea accesible y esté optimizada para motores de búsqueda.
2. Formulario de registro con validación: Diseña un formulario de registro para un sitio web o aplicación, utilizando diferentes tipos de campos de entrada (texto, correo electrónico, contraseña, etc.). Implementa validaciones en el lado del cliente utilizando atributos HTML5 como required, pattern, minlength y maxlength. Proporciona mensajes de error claros y concisos para guiar al usuario en caso de errores de validación.
3. Creación de una página de producto con metadatos enriquecidos: Elabora una página web que presente un producto o servicio, incluyendo imágenes, descripciones detalladas y especificaciones técnicas. Utiliza tablas para organizar la información de manera clara y estructurada. Agrega metadatos enriquecidos, como Open Graph y Twitter Cards, para controlar cómo se muestra la página al ser compartida en redes sociales, incluyendo título, descripción, imagen destacada y URL.

## VI

### CUESTIONARIO

1. ¿Cuáles son las ventajas y desventajas de utilizar etiquetas semánticas en HTML5 en comparación con etiquetas no semánticas?

2. ¿Cómo la elección de una estructura HTML5 adecuada puede impactar la accesibilidad de una página web para usuarios con discapacidades?
3. Analiza dos sitios web populares y evalúa la efectividad de su estructura HTML5 en términos de organización de contenido y facilidad de navegación.
4. ¿Qué criterios utilizarías para decidir si implementar un formulario complejo en una sola página o dividirlo en varias páginas?
5. ¿Cómo puedes asegurarte de que el contenido multimedia (imágenes, videos, audio) incluido en una página web sea accesible para todos los usuarios?
6. Escribe el código HTML5 para crear un formulario de contacto simple con campos para nombre, correo electrónico y mensaje, incluyendo las etiquetas semánticas adecuadas.
7. Implementa validaciones en el lado del cliente en un formulario HTML5 para asegurar que los usuarios ingresen datos en el formato correcto (por ejemplo, una dirección de correo electrónico válida).
8. Crea una tabla HTML5 para mostrar el horario de clases de una semana, utilizando etiquetas como `<table>`, `<tr>`, `<th>` y `<td>`.
9. Optimiza una página web existente para mejorar su posicionamiento en motores de búsqueda, aplicando técnicas de SEO on-page como el uso de metadatos relevantes y palabras clave.
10. Utiliza Git para crear un repositorio, realizar cambios en un archivo HTML, confirmar los cambios y subirlos a GitHub Pages.
11. ¿Por qué es importante considerar la accesibilidad al diseñar y desarrollar aplicaciones web?
12. ¿Cómo puedes asegurarte de que tus elecciones de diseño en HTML5 no excluyan a ningún grupo de usuarios?
13. Describe un problema técnico que hayas encontrado al desarrollar una página web y cómo lo resolviste.
14. ¿Qué estrategias utilizarías para mantenerte actualizado sobre las nuevas tecnologías y tendencias en el desarrollo web?
15. ¿Cómo el diseño web puede influir en la percepción de una marca o empresa por parte de los usuarios?
16. ¿Cuál es la diferencia entre los métodos HTTP GET y POST al enviar datos de un formulario?
17. ¿Qué son las expresiones regulares (regex) y cómo se pueden utilizar para validar datos en formularios HTML5?
18. ¿Cuáles son las ventajas de utilizar tablas HTML5 en lugar de otros elementos para presentar datos tabulares?
19. ¿Qué son los metadatos Open Graph y cómo pueden mejorar la forma en que se comparte una página web en redes sociales?
20. ¿Cómo las Twitter Cards pueden aumentar la visibilidad y el engagement de una página web en Twitter?
21. ¿Qué herramientas de desarrollo web utilizas para crear y depurar tus proyectos HTML5?
22. ¿Cuáles son las últimas tendencias en diseño y desarrollo web que consideras importantes tener en cuenta?
23. ¿Cómo el conocimiento de HTML5 puede complementar otras habilidades en el campo de la tecnología?
24. ¿Qué recursos en línea o libros recomendarías para seguir aprendiendo sobre HTML5 y desarrollo web en general?
25. ¿Qué proyectos personales o profesionales te gustaría desarrollar utilizando HTML5 en el futuro?

## VII

## REFERENCIAS

Data, R. (15 de 06 de 2022). *W3Schools*. Obtenido de <https://www.w3schools.com/>

Mardan, A. (2018). *Full Stack JavaScript*. Nueva York: Apress.

Matarazzo, D. (2021). *Aprenda los lenguajes HTML5, CSS3 y JavaScript para crear su primer sitio web*. Barcelona: Editorial ENI.

Mozilla. (24 de 06 de 2022). *MDN*. Obtenido de <https://developer.mozilla.org/en-US/docs/Learn/HTML>

Rubiales Gomez, M. (2018). *Curso de Desarrollo Web HTML, CSS Y JavaScript*. Madrid: ANAYA MULTIMEDIA.