Autor: Juan José Ruiz a.k.a GOGODEV
YouTube: https://youtube.com/c/gogodev
Twitter: https://twitter.com/JJRuizEmpresa (@JJRuizEmpresa)

IMPORTANTE

El contenido de este pdf lo tienes también en vídeo en el canal de YouTube https://youtube.com/c/gogodev , dentro de la lista de reproducción: Curso de HTML.

Todo el contenido formativo es gratuito, no olvides pasarte y suscribirte para no perderte nada, y así nos apoyas :)

<CAPÍTULO .01>

HOLA MUNDO

CONCEPTOS GENERALES Y PRIMEROS PASOS

</CAPÍTULO .01>

HOLA MUNDO

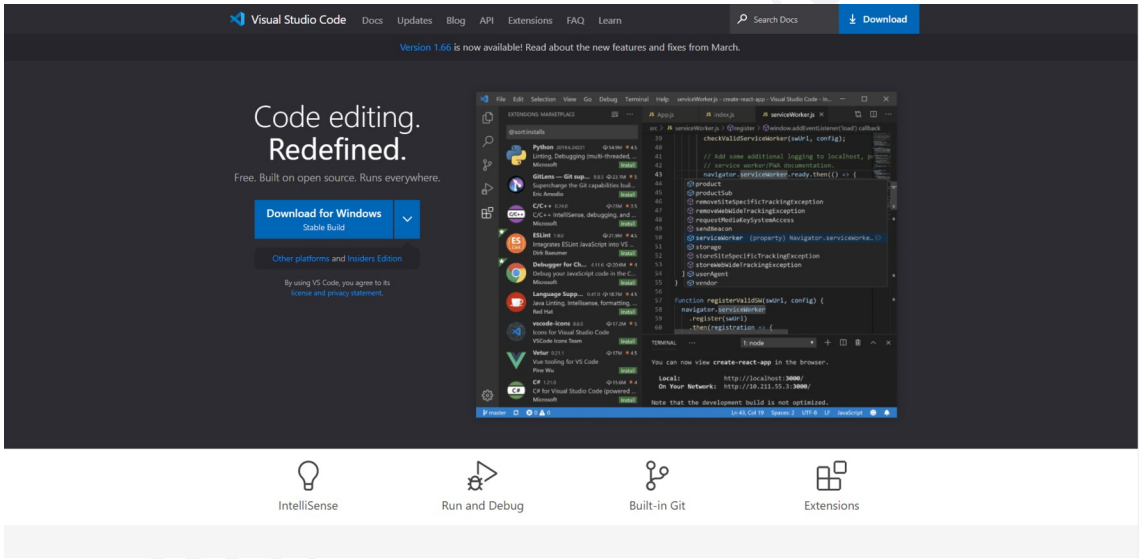
<CONFIGURANDO_EL_ENTORNO_DE_TRABAJO>

Antes de nada, lo que debemos hacer es configurar nuestro entorno de trabajo. A lo largo de este curso vamos a usar Visual Studio Code como nuestro editor de código, no obstante, puedes usar cualquier otro si así lo deseas (sublime text, web storm, etc.)

Me he decantado por Visual Studio Code porque es un editor muy completo y robusto desarrollado por Microsoft. Es totalmente gratuito y es de los más usados por la comunidad, ofreciéndonos todo lo que necesitamos para desarrollar nuestra actividad.

Puedes descargar Visual Studio Code desde su página web oficial[3], y está disponible para todos los sistemas operativos: Windows, Mac y Linux.

Accede a su página web, descarga e instala Visual Studio Code si no lo tienes aún y continuamos. Visual Studio Code será el IDE que estaremos utilizando.



</CONFIGURANDO_EL_ENTORNO_DE_TRABAJO>

3 https://code.visualstudio.com/

<QUÉ_ES_UN_IDE>

En el apartado anterior hemos mencionado que Visual Studio Code va a ser nuestro IDE pero, ¿qué es un IDE?

Un IDE (Entorno de Desarrollo Integrado) es una herramienta de software que proporciona un entorno de programación completo para programadores. Este conjunto de herramientas es utilizado para realizar todas las características de nuestro desarrollo a través de un entorno único y centralizado.

En esencia, va a ser ese editor de texto que va a incorporar todas las herramientas necesarias para que podamos trabajar.

</QUÉ_ES_UN_IDE>

<PRIMEROS_PASOS_CON_VISUAL_STUDIO_CODE>

Ya tenemos descargado e instalado Visual Studio Code, nuestro IDE. Es hora de echarle un vistazo y comenzar a familiarizarnos con él. Lo primero que vamos a hacer es abrir Visual Studio Code, y vamos a observar el menú vertical situado a la izquierda.

Explorador de archivos: Al pulsar sobre este botón se nos mostrará la estructura de archivos de nuestro proyecto. Es donde vamos a trabajar.

Sección de búsqueda: Para buscar cierto código o texto en nuestros archivos.

Control de código fuente: Si estamos usando Git o algún tipo de repositorio, desde aquí podremos controlarlo todo.
Ejecución y depuración: Para compilar lenguajes compilados y buscar errores en nuestro código.
Extensiones: Para buscar y añadir nuevas funcionalidades a nuestro Visual Studio Code.

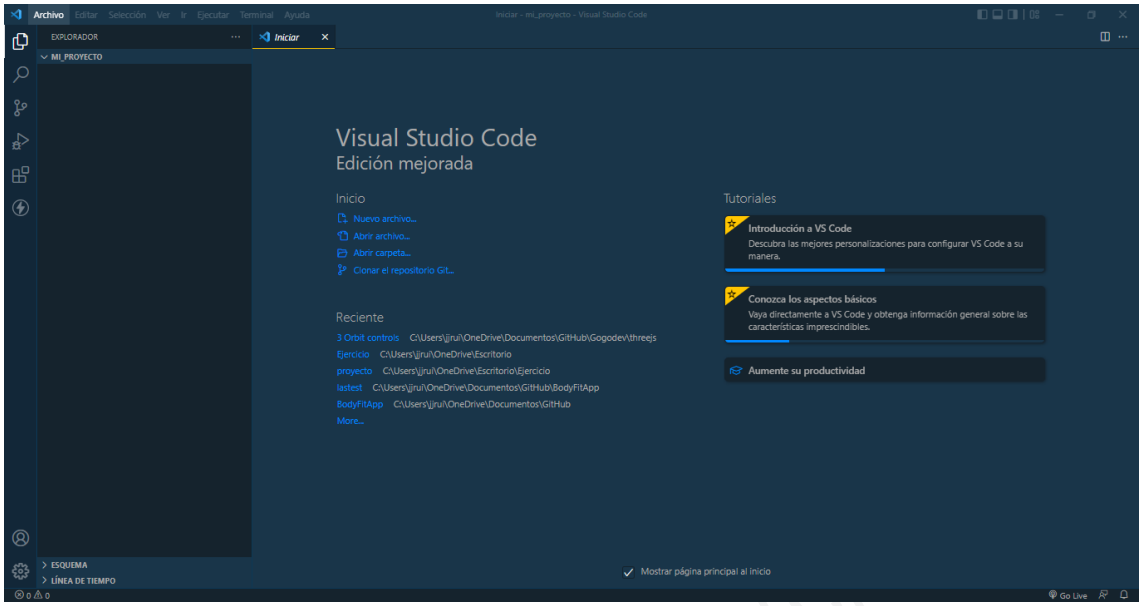
Si no entiendes la explicación de lo que hace cada elemento no te preocupes, no te dejes abrumar por las definiciones. A base de ir usando el IDE irás familiarizándote con cada uno de ellos. En este momento, y a esta altura de la formación, es suficiente con que te quedes con lo siguiente:

- En la sección del primer icono (Explorador de archivos) es donde vamos a trabajar.
- En la sección del último icono (Extensiones) es donde vamos a personalizar nuestro IDE por si queremos añadirle nuevas funcionalidades.

Para entender bien todo esto, vamos a crear nuestro primer proyecto. Acude a tu escritorio o a la ubicación que desees de tu ordenador, y crea una nueva carpeta. (Click derecho > Nueva Carpeta) Dale el nombre que prefieras.

Ahora, arrastra y suelta la carpeta en la zona principal de la ventana de tu Visual Studio Code. Así de sencillo, ya estamos trabajando nuestro proyecto en dicha carpeta. Si lo prefieres, también puedes abrir tu proyecto pulsando sobre Archivo > Abrir Carpeta.

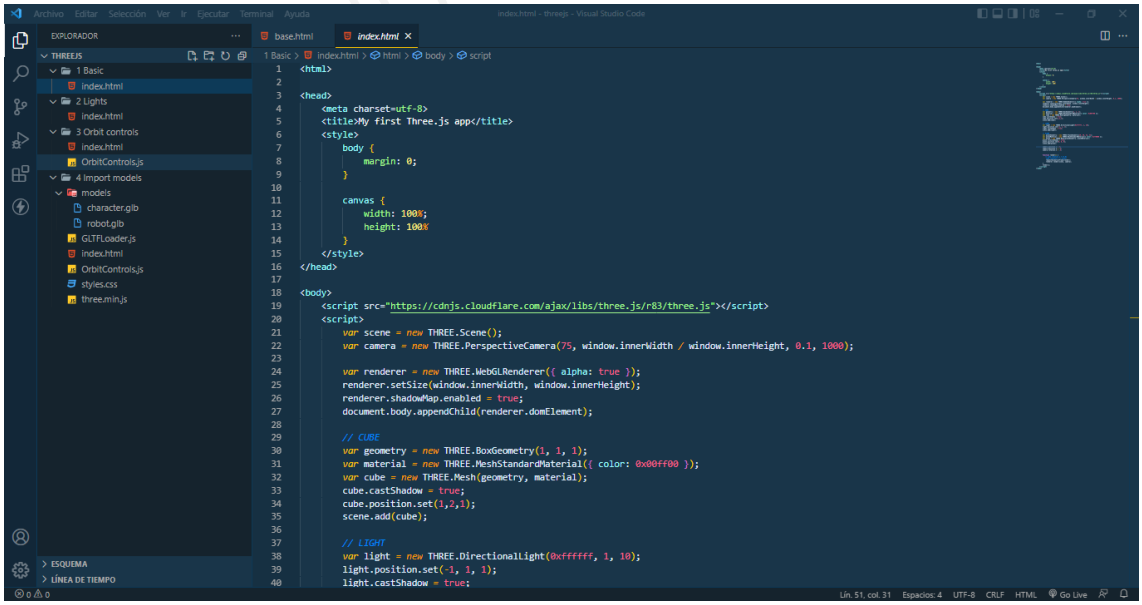
Al haber realizado esta acción, podrás comprobar que en tu Visual Studio Code te aparece una división:



- En la zona de la izquierda nos aparece la estructura de archivos de nuestro proyecto (que actualmente está vacía ya que la carpeta la acabamos de crear)

- En la zona principal y más grande de la derecha, es donde vamos a editar y escribir el código de los archivos que vayamos seleccionando en la zona de la izquierda.

Así se verá en el futuro cuando estemos trabajando un proyecto en el que ya tengamos contenido:



Es hora de crear nuestro primer archivo para podamos comenzar a aprender. En la zona del explorador de archivos (a la izquierda) vamos a hacer click derecho, y seleccionar la opción “Nuevo archivo”, y vamos a darle el nombre index.html

Es importante que su extensión sea .html Más adelante veremos el por qué. De momento, ya hemos creado nuestro primer archivo de nuestro primer proyecto. Más adelante volveremos a él.

</PRIMEROS_PASOS_CON_VISUAL_STUDIO_CODE>

<EXTENSIONES_Y_TEMAS>

Antes he hablado del último botón de la sección de la izquierda de Visual Studio Code, la sección de extensiones. Hablemos un poco de esta sección antes de comenzar ya manos a la obra.

Como podemos deducir de su nombre, en esta sección Visual Studio Code nos permite añadir plugins que amplían las funcionalidades del sistema. Para poder seguir las enseñanzas de este curso no es necesario instalar ninguna extensión adicional. Sin embargo, déjame recomendarte instalar algunas que, aunque no son necesarias, pueden resultarte de tu interés ya sea ahora o en un futuro próximo:

- **Live Server**: Esta extensión una vez instalada nos permitirá abrir un archivo simulando su ejecución en un servidor local, haciendo click derecho sobre dicho archivo > abrir en live server.
- **Thunder client**: Nos permite hacer pruebas de peticiones a una API. Si conoces Postman es igual, pero dentro del IDE de Visual Studio Code.

Además, desde esta sección no solo podemos instalar nuevas funcionalidades, si no también temas que nos permiten modificar el aspecto visual de nuestro entorno de desarrollo, usando nuevos esquemas de colores, por ejemplo. Puedes buscar un tema que te guste para sentirte lo más cómodo posible a la hora de trabajar. Si sientes curiosidad por ello, a mi me gusta usar el tema: Cobalt 2 o Synthwaver’84.

Y ahora sí, ya estamos listos para comenzar. Volvamos al proyecto que creamos en la sección anterior y comencemos.

</EXTENSIONES_Y_TEMAS>

<QUÉ_ES_HTML>

Empecemos por el principio del desarrollo web, y esto significa comenzar por conocer HTML, pero… ¿qué es exactamente HTML?

HTML es un lenguaje de marcado. Dentro de nuestras construcciones web, va a ser el encargado de definir la estructura de cada página web. Con los lenguajes de marcado, como HTML, usamos <etiquetas> para ir definiendo toda la estructura de cada una de nuestras páginas. Y es que a la hora de crear el frontal de un sitio web (frontend) tenemos que tener en cuenta que vamos a usar tres tecnologías principalmente:

- **HTML**: Para crear la estructura de nuestra página web. Con el vamos a definir todo el contenido, que se conoce formalmente por el nombre de semántica. Los archivos HTML tienen la extensión (acaban en) .html
- **CSS**: Este es el lenguaje que usaremos para definir el estilo visual de nuestra web. Dicho de otro modo: se encarga del diseño, de cómo se van a ver los contenidos que hemos escrito en el HTML (su color, su fondo, su tipografía, su posición en la pantalla) Los archivos CSS tienen la extensión .css
- **JavaScript**: Por último, tenemos JavaScript, que es el lenguaje de programación que nos ayuda a añadir lógica y comportamientos a nuestra experiencia de usuario. Los archivos JavaScript tienen la extensión .js.

Así pues y, en resumen, a la hora de crear un sitio web vamos en primer lugar a escribir su estructura (semántica) con HTML, después vamos darle diseño con CSS y posteriormente, y en caso de ser necesario, le añadiremos funcionalidades con JavaScript.
Aclarado este punto es hora de ponernos manos a la obra y escribir nuestro primer código en HTML.

</QUÉ_ES_HTML>

<ETIQUETAS_HTML>

Como hemos dicho anteriormente, HTML es un lenguaje de marcado, y esto significa que nos va a permitir definir su estructura a través de etiquetas.
Para escribir una etiqueta en HTML lo hacemos de la siguiente manera:

```
<etiqueta>
  Contenido de la etiqueta
</etiqueta>
```

Como puedes observar se trata de encerrar entre una apertura <nombre de la etiqueta> y un cierre </nombre de la etiqueta> un contenido.

*NOTA: También es posible encontrarnos con la expresión **<nombre_de_la_etiqueta />** Esta es una abreviatura que abre y cierra la etiqueta directamente, para todas aquellas etiquetas que no van a tener contenido dentro.*

Y por supuesto, en HTML vamos a tener una gran cantidad de etiquetas, y cada una de ellas declara una estructura diferente. Por ejemplo, la etiqueta <p> nos sirve para escribir un párrafo en nuestra página web. Así pues, si escribimos el siguiente código en nuestro archivo HTML:

```
<p>Hola Mundo.</p>
```

Habremos escrito un párrafo en nuestra web que dice: *"Hola Mundo."*
Por último, demos un pasito más en la definición de etiquetas antes de continuar.

Y es que, a la hora de escribir nuestras etiquetas, podemos encontrarnos, y de hecho nos encontraremos constantemente, con que muchas de ellas necesitan de más información que su propio nombre para funcionar.

Tomemos por ejemplo la etiqueta <a>. Esta etiqueta nos servirá para hacer un enlace (también llamado link, anchor, ancla o hipervínculo), es decir, nos sirve para hacer que, en nuestra página, cuando se pulse sobre este enlace, se cargue otra página distinta (como ocurre por ejemplo en un menú de navegación)

En este ejemplo, no nos valdría únicamente con escribir:

```
<a>Pulsa aquí para ver el artículo</a>
```

La información estaría incompleta puesto que no le estamos diciendo al enlace a qué nuevo archivo HTML debe dirigir al usuario cuando pulse sobre el enlace. Pues para esto usamos los atributos de las etiquetas. Podemos usar tantos atributos como sean necesarios en una etiqueta, y estos lo que van a hacer es completar su significado. Y por supuesto, y al igual que ocurre con las etiquetas, vamos a contar con una gran cantidad de atributos, y cada uno de ellos definirá un aspecto diferente. Los atributos se escriben de la siguiente forma:

```
<etiqueta atributo1=”valor_1” atributo2=”valor_2” … >
        Contenido de la etiqueta
</etiqueta>
```

Siguiendo el ejemplo anterior, el atributo href indica hacia dónde debe dirigirse un enlace, es decir, al documento al que referencia. Por tanto, para terminar de completar el ejemplo anterior, deberíamos escribir:

```
<a href="mi_articulo.html">Pulsa aquí para ver el artículo</a>
```

Y eso es todo. Así de sencillo. Sobre estos pilares construiremos nuestra iglesia.
Como siempre, no te preocupes por la píldora teórica. A medida que vayamos avanzando irás interiorizando estos conceptos. Y hablando de avanzar, ahora que ya sabes qué es una etiqueta y cómo se escriben en HTML, es hora de continuar dando pasos.

</ETIQUETAS_HTML>

<ESTRUCTURA_GENERAL_DE_UN_DOCUMENTO_HTML>

Como hemos visto, HTML es el lenguaje con el que vamos a definir la estructura de nuestros sitios web. Y este, es un lenguaje que se basa en etiquetas. A través de ir escribiendo dichas etiquetas vamos a ir construyendo nuestro sitio web.

Ahora que ya sabemos esta, vamos a ver cuál es la estructura básica de cualquier documento HTML. Todos nuestros documentos respetarán siempre esta estructura:

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

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

Analicemos las etiquetas de nuestra estructura base:

```
<!DOCTYPE html>
```

Esta es una etiqueta especial con la que siempre vamos a iniciar nuestros documentos. Lo que hace esta etiqueta es indicar al navegador que el presente documento que estamos escribiendo es de tipo HTML.

Posteriormente, declaramos la etiqueta <html>, y dentro de esta escribimos todo el código de nuestra página. Como puedes observar, podemos escribir etiquetas dentro de otras etiquetas. A esto se le denomina anidamiento. A la etiqueta que contiene otras etiquetas la llamamos etiqueta padre, y a las etiquetas que se encuentran en su interior, etiquetas hijo.

Así pues, dentro de la etiqueta principal de todo documento, la etiqueta <html> vamos a encontrarnos siempre con dos etiquetas hijo: la etiqueta <head> y la etiqueta <body>.

- **<head>** Dentro del head escribiremos todo el contenido que es necesario para nuestra página web, pero que no se muestra en el navegador. Es decir, aquí escribimos los metadatos de nuestro documento. Por ejemplo: El título del documento, qué tipos de caracteres vamos a usar, quién es el autor del código, cuál es el archivo de diseño que va a estar usando este html… en resumen: Toda la información de utilidad para la página que no se muestra.

- **<body>** Aquí es donde escribimos el contenido de nuestra página web. Dentro del body escribiremos todo lo que se va a ver en el navegador cuando se visite la página (los títulos, los textos, las imágenes, los enlaces, los vídeos, los párrafos…)

Continuemos ahora ampliando un poco más la estructura básica de nuestro documento:

Comencemos por la etiqueta <html>. Existe un atributo muy interesante para esta etiqueta que deberíamos especificar en todos nuestros documentos html, y este no es otro que Lang, quien especifica en qué idioma está la página web que estamos viendo. Esto es muy útil para que el navegador, al visitar nuestro sitio web, sepa si está mostrando una página web con su contenido en español, en inglés, en francés… Podemos por tanto ampliar la información que la etiqueta html ofrece añadiéndole este atributo:

```
<html lang="en">
```

En este ejemplo, estamos indicando que nuestra página contiene textos en inglés (en). Si por el contrario queremos especificar que los textos son en español, bastará con cambiar el código internacional del idioma de '*en*' (english) a '*es*' (español):

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

¡Muy bien! Ya hemos usado nuestro primer atributo para ampliar el significado de una etiqueta. Continuemos viendo más etiquetas que deberíamos incluir en todos nuestros documentos html. Veamos ahora el <head>.

Como hemos dicho anteriormente, en el <head> escribimos toda la información de interés que no aparece en la página web, y cuando hablamos de HTML5 (la versión más moderna de HTML hasta la fecha) existen dos etiquetas que siempre va a ser interesante especificar:

- **<title>** Define el título de la página web. Este no se ve en la página web en sí, si no que se muestra arriba del todo, en la pestaña del navegador. Lo veremos más adelante cuando escribamos nuestro primer código.

- **<meta>** Agrega diferentes metadatos a nuestra página web.

La etiqueta <title> es simple y sencilla, ya que tan solo tiene una única función: escribir el título. La etiqueta <meta>, sin embargo, contiene una gran cantidad de atributos que puede definir. Por convención (consenso en la comunidad), cuando estamos escribiendo un documento de html moderno (html5), vamos a especificar siempre los siguientes metas, tal y como se muestran a continuación:

```
<meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
```

Veamos qué hace cada uno de ellos:

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

Este atributo indica que la codificación de caracteres va a ser UTF-8. ¿Y para qué sirve esto? Muy sencillo: Esta codificación de caracteres es la que incluye caracteres especiales de los idiomas anglosajones y latinos, propios del alfabeto occidental extendido, tales como son las tildes, la letra ñ, la letra ç, etcétera. De esta forma, nos aseguramos que el navegador escriba correctamente cada letra. Si por el contrario, estuviésemos escribiendo en una lengua que usa otro tipo de caracteres (japonés, coreano, cirílico) la codificación[5] a utilizar sería diferente.

5 Listado con los tipos de *charsets* según el idioma: https://docs.oracle.com/cd/E26921_01/html/E27143/glset.html

```
<meta http-equiv="X-UA-Compatible" content="IE=edge">
```

Puede parecer algo confusa, pero es ciertamente interesante. Los navegadores (Edge, Chrome, Firefox…) son los encargados de interpretar el código HTML que nosotros escribimos para que el usuario pueda visualizar correctamente la página web. En algunos casos, como es el caso de Microsoft, han realizado nuevas y diferentes versiones de su navegador, hasta el punto de haberlo cambiado completamente de Internet Explorer (navegador antiguo) a Edge (navegador nuevo) Con esta instrucción meta nos aseguramos una correcta compatibilidad para que el usuario de este tipo de navegadores pueda ver nuestra página web tal y como nosotros la hemos escrito.

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

Por último, tenemos esta instrucción meta. A lo largo de la última década, la explosión de la tecnología móvil ha aumentado considerablemente el número de diferentes pantallas a través de las cuáles se accede a una página web. Ahora, no solo consumimos internet a través de un ordenador, sino que también podemos hacerlo a través de una Tablet o un dispositivo móvil.

Con esta instrucción, nos aseguramos de establecer el ancho de la pantalla como la escala inicial del contenido (ya que en este tipo de dispositivos podemos hacer, entre otras cosas, zoom)
Por ende, nos estamos asegurando que el dispositivo interpreta bien cuáles son las dimensiones de la escala inicial del contenido web.

Y ahora sí, ya tenemos todo el contenido básico inicial que todo documento web html escrito por nosotros debería tener. Cada vez que creemos un archivo html, por tanto, nos aseguraremos de escribir siempre esta estructura. Pongámosla completa para que podamos verla mejor:

```
<!DOCTYPE html>
<html lang="es">
<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>Mi página web</title>
</head>
<body>

</body>
</html>
```

</ ESTRUCTURA_GENERAL_DE_UN_DOCUMENTO_HTML>

<SNIPPETS_EMMET_Y_MENU_CONTEXTUAL>

Ahora que sabemos cuál debe ser la estructura mínima que debemos colocar en cada documento HTML, nos puede surgir una pregunta asociada a la productividad bastante pertinente: *"Si debo colocar siempre este código mínimo en cada documento, y mi proyecto web dispone de una gran cantidad de documentos HTML, ¿voy a tener que estar escribiendo siempre una y otra vez lo mismo?"*

La respuesta es no. Visual Studio Code nos va a ofrecer diferentes opciones para facilitarnos la escritura de código: Los snippets, las abreviaciones emmet y el menú contextual.
Empecemos por el primero de ellos:

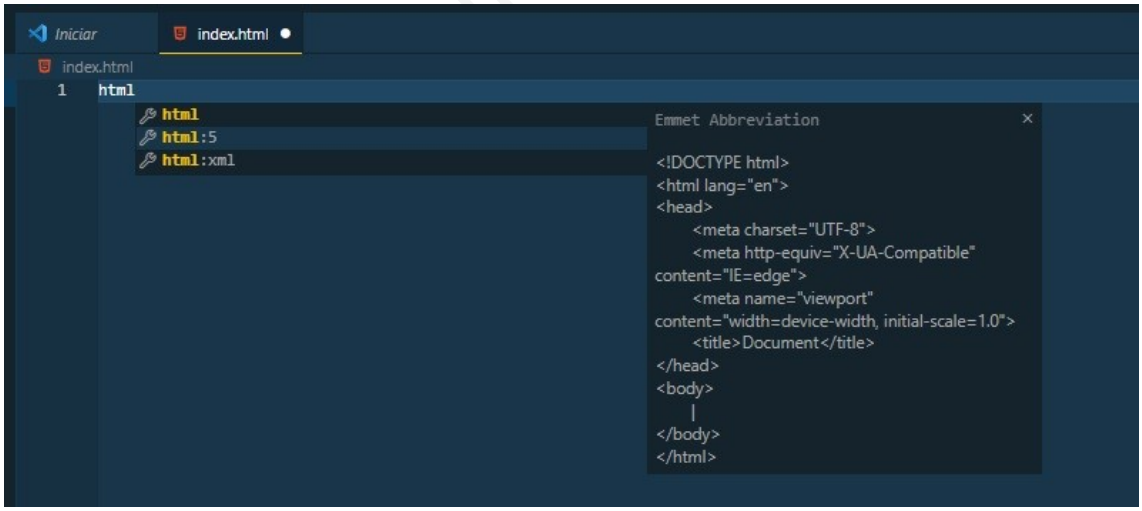
Snippets

Los snippets son trozos de código ya encapsulados que podemos asociar a una palabra para que, cuando escribamos dicha palabra, el editor de código nos autocomplete colocando todo ese trozo de código por nosotros. Si bien podemos crear nuestros propios snippets, de momento nos bastará con saber Visual Studio Code ya nos incorpora un gran repertorio de estos de base.

Vamos a probarlo: Crea un nuevo documento .html (o usa el index.html que ya has creado anteriormente) Selecciónalo y comienza a escribir en él la palabra html.

Al ir haciéndolo, podrás observar que bajo tu línea de texto te aparece un menú contextual con diferentes opciones. Selecciona la que dice html:5, ya sea pulsando sobre ella o navegando con las teclas de dirección en el menú y pulsando *enter* para seleccionar.

¿Has podido observar qué ha ocurrido? Visual Studio Code te ha escrito por ti toda la estructura base de un documento HTML5 tal y como vimos en el punto anterior.



Abreviaciones EMMET

Las abreviaciones EMMET están pensadas para cuando tienes que escribir múltiples trozos de código que se van a repetir. Por ejemplo, supongamos que vamos a escribir cinco párrafos. Como vimos anteriormente, un párrafo lo escribimos usando la etiqueta <p>

Si a la hora escribir nuestro código escribimos:

p*5

Y pulsamos *enter*, podremos ver como Visual Studio Code entiende la abreviación y nos escribe cinco párrafos diferentes.

Existen numerosas abreviaciones, y en este curso iremos destacando las más comunes conforme vaya siendo interesante destacarlas.

Menú contextual
El menú contextual te aparecerá cada vez que estés escribiendo código, sugiriéndote un autocompletar, además de los diferentes EMMET y Snippets para lo que estás escribiendo. Si alguna vez lo desactivas por alguna razón, puedes forzar su uso pulsando control + barra espaciadora.

</SNIPPETS_EMMET_Y_MENU_CONTEXTUAL>

<COMENTARIOS>

Veamos ahora un último punto antes de decirle hola al mundo como aprendices del desarrollo web: Los comentarios.

Cuando estamos escribiendo código, es habitual que en ciertas ocasiones nos interese insertar un comentario en el mismo. Pero no queremos que este comentario aparezca en el resultado final de nuestra página. Es tan solo una indicación en el código para nosotros (o para un compañero si estamos trabajando en un equipo) Puede ser interesante para recordar qué está haciendo una instrucción en concreto, para organizarnos mejor, para delimitar zonas… Pues bien, como hemos dicho, para esto usaremos los comentarios. En HTML, podremos escribirlos de la siguiente manera:

```
<!--
    Hola, soy un comentario.
    Puedo contener toda la extensión que desee,
    y ocupar tantas líneas como sea necesario.
-->
```

Como puedes observar, nuestro editor de código entenderá que estamos escribiendo un comentario, y nos lo marcará de un color diferente. Siempre deben empezar por <!-- y acabar por →

</COMENTARIOS>

<HOLA_MUNDO>

Conocemos las herramientas necesarias para comenzar. Ha llegado la hora de enfrentarnos a nuestro primer código. Digámosle hola al mundo.

Crea un nuevo proyecto (o usa el que creamos anteriormente) y crea un primer archivo index.html (o con el nombre que prefieras) tal y como hemos visto hacer en este capítulo.

Ahora, selecciónalo para comenzar a escribir el código. Hora de colocar su estructura base. Puedes escribirla a mano o usar el snippet html:5 que has aprendido en la sección de *snippets, emmet y menú contextual.*

Modifiquemos ahora la estructura base. Vamos a colocar el idioma en español *lang="es"* y el título va a ser

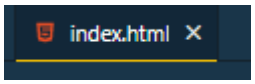
<title>Mi primera página web</title>

Lo tenemos todo preparado, ahora vamos a la parte del <body>, donde recordamos vamos a escribir toda la estructura de nuestra página, y vamos a colocar un párrafo que diga Hola Mundo. Es decir:

Hola Mundo

Nuestro código está completo. Visual Studio Code nos recuerda que no hemos guardado aún los cambios en el código del archivo con un círculo al lado del nombre del archivo que estamos trabajando, en la pestaña.

Guardemos el archivo pulsando en el menú superior *archivo > guardar* o usando el atajo de teclado *control + S* Dicho círculo debe ahora haber desaparecido.



Este es un recordatorio útil para saber cuándo no hemos guardado cambios.

Tu código debe ser similar al siguiente:

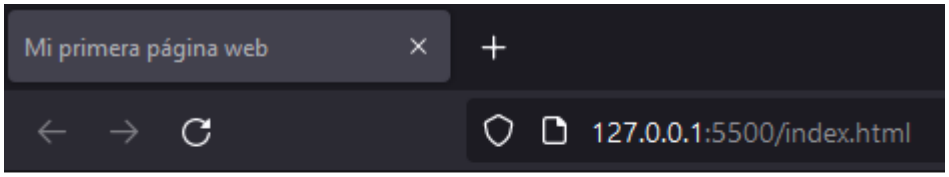
```
<!DOCTYPE html>
<html lang="es">
<head>
   <meta charset="UTF-8">
   <meta http-equiv="X-UA-Compatible" content="IE=edge">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Mi primera página web</title>
</head>
<body>
   <p>Hola Mundo</p>
</body>
</html>
```

Es hora de comprobar si lo hemos hecho bien. Para ello, como aún estamos trabajando páginas web estáticas tenemos dos opciones para ver el resultado rápidamente:

1. Si has instalado la extensión Live Server de Visual Studio Code, en el explorador (la sección de la izquierda de Visual Studio Code) haz click derecho sobre el archivo y selecciona la primera opción *"Abrir en Live Server"* *"Open with Live Server"* en inglés.

2. Si no has instalado la extensión, basta con que en tu ordenador (fuera de Live Server) te dirijas a la carpeta de tu proyecto, y hagas doble click sobre el archivo .html que has creado.

Et voilà ! Ya tenemos nuestra primera página web.



Hola Mundo

NOTA: Si alguna parte del proceso no te ha salido correctamente, recuerda que puedes ver un vídeo con el resumen de este capítulo en nuestro canal GOGODEV.

Quizás no sea aún muy impresionante, pero no te preocupes por eso, aprendiz.

Todo llega a aquel que saber esperar.

</HOLA_MUNDO>

Autor: Juan José Ruiz a.k.a GOGODEV
YouTube: https://youtube.com/c/gogodev
Twitter: https://twitter.com/JJRuizEmpresa (@JJRuizEmpresa)

IMPORTANTE

El contenido de este pdf lo tienes también en vídeo en el canal de YouTube https://youtube.com/c/gogodev , dentro de la lista de reproducción: Curso de HTML.

Todo el contenido formativo es gratuito, no olvides pasarte y suscribirte para no perderte nada, y así nos apoyas :)

<CAPÍTULO .02>

EL DOMINIO DE LA PALABRA

ETIQUETAS RELACIONADAS CON LA ESCRITURA

</CAPÍTULO .02>

EL DOMINIO DE LA PALABRA

<TÍTULOS_Y_PÁRRAFOS>

Es hora de ampliar nuestras herramientas a la hora de escribir texto. Y es que, sea cual sea el proyecto que vayamos a realizar, en todos vamos a necesitarlos.

Comencemos por los títulos:

<h1> <h2> <h3> <h4> <h5> y <h6> Son las etiquetas para escribir títulos. Van desde el título más importante, y por tanto de mayor tamaño <h1> al menos importante y por tanto más pequeño <h6> Ejemplo:

```
<h1>Título h1</h1>
<h2>Título h2</h2>
<h3>Título h3</h3>
<h4>Título h4</h4>
<h5>Título h5</h5>
<h6>Título h6</h6>
```

Título h1

Título h2

Título h3

Título h4

Título h5

Título h6

Decimos que las etiquetas de título son etiquetas de bloque. Esto quiere decir que, por defecto, al escribirlas, cuando una de estas termina el contenido salta automáticamente a la siguiente línea. Por defecto, no tienen contenido a sus laterales.

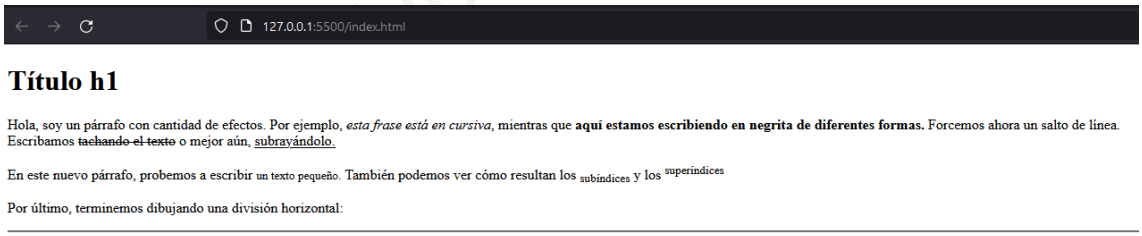
Vayamos ahora a por una ración de etiquetas de texto:

<p> Para escribir párrafos.
<i> Escribe en cursiva.
<b> y **** Escriben en negrita.
<del> Escribe un texto tachado.
<u> Para subrayar un texto.
<sub> y **<sup>** Para subíndices y superíndices.
<small> Escribe textos pequeños.
**
** Fuerza un salto de línea.
<hr /> Dibuja una línea de división horizontal.

Ejemplo en código de las etiquetas anteriores:

```
<h1>Título h1</h1>
  <p>Hola, soy un párrafo con cantidad de efectos.
    Por ejemplo, <i>esta frase está en cursiva</i>, mientras que
    <b>aquí estamos escribiendo en negrita</b> <strong>de diferentes formas.</strong>
    Forcemos ahora un salto de línea. <br />Escribamos <del>tachando el texto</del> o mejor aún,
    <u>subrayándolo.</u>
  </p>
  <p>En este nuevo párrafo, probemos a escribir <small>un texto pequeño.</small> También podemos ver cómo resultan
    los <sub>subíndices</sub> y los <sup>superíndices</sup></p>
  <p>Por último, terminemos dibujando una división horizontal:</p>
  <hr />
```

Resultado:



PRÁCTICA: La práctica hace al maestro. Así pues, y antes de continuar, prueba a realizar un documento html que muestre el siguiente texto, incluyendo su formato:

Hora de practicar

En este primer párrafo vamos a mostrar que sabemos colocar **negritas**.

Es hora de *probar otros textos* en este nuevo párrafo que muestren **todas las opciones** que ~~vamos a aprender~~ hemos aprendido.

Terminamos escribiendo un superíndice y también un subíndice.

Detente a practicar hasta que vayas familiarizándote con las etiquetas.
Fácil, ¿verdad? Así pues, ¡hora de continuar!

</TÍTULOS_Y_PÁRRAFOS>

<LISTAS>

Continuamos con las listas dentro de HTML. Tenemos dos principales tipos de ellas: Las listas ordenadas y las no ordenadas.

Para crear una lista ordenada usamos la etiqueta **<ol>**, mientras que para una lista no ordenada usamos **<ul>**.

Una vez creado la etiqueta, dentro definimos cada uno de los elementos de la lista con la etiqueta **<li>**. **<li>** es común tanto para las listas ordenadas como no ordenadas.

Por ejemplo:

```
<h3>Lista no ordenada</h3>
<ul>
   <li>Elemento 1</li>
   <li>Elemento 2</li>
   <li>Elemento 3</li>
</ul>

<h3>Lista ordenada</h3>
<ol>
   <li>Elemento 1</li>
   <li>Elemento 2</li>
   <li>Elemento 3</li>
</ol>
```

Resultado:

Lista no ordenada

- Elemento 1
- Elemento 2
- Elemento 3

Lista ordenada

1. Elemento 1
2. Elemento 2
3. Elemento 3

Como puedes observar, la principal diferencia entre ambas listas, a parte de su semántica, se basa en el aspecto de sus viñetas.

EMMET PARA LAS LISTAS

Si deseas escribir una lista muy larga, la tarea de escritura puede hacerse un tanto tediosa. Para hacer la tarea más sencilla, dispones de una abreviación EMMET en Visual Studio Code para escribirlas más rápido:

Tipo_de_lista>li*numero_de_elementos

Por ejemplo:

*ul>li*5*

Generará:

```
<ul>
  <li></li>
  <li></li>
  <li></li>
  <li></li>
  <li></li>
</ul>
```

Si observamos la abreviatura, podemos empezar a discernir cierta lógica en su formulación:

- El símbolo **>** indica 'dentro de este' en EMMET.
- El símbolo ***** indica multiplicación en EMMET.

Así pues, **ul>li*5** podríamos leerlo como: Crear una etiqueta **ul**, dentro de esta una etiqueta **li** multiplicado por cinco.

CONSEJO: Si además de usar los EMMET que vamos viendo durante el curso dedicas algo de tiempo a entender su lógica, podrás avanzar mucho más, siendo capaz de ir escribiendo los tuyos propios cuando los necesites. Préstales atención a estos cuando se usen en el curso.

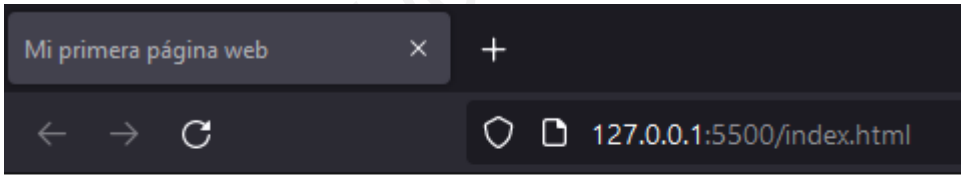
</LISTAS>

<TABLAS>

Es hora de organizar nuestro contenido en elementos un poco más complejos: veamos las tablas. Para crear una tabla en HTML usaremos la etiqueta **<table>**. Dentro de esta definiremos cada fila dentro de una etiqueta **<tr>**, y dentro de cada fila definiremos las celdas con la etiqueta **<td>**

Veámoslo en un ejemplo:

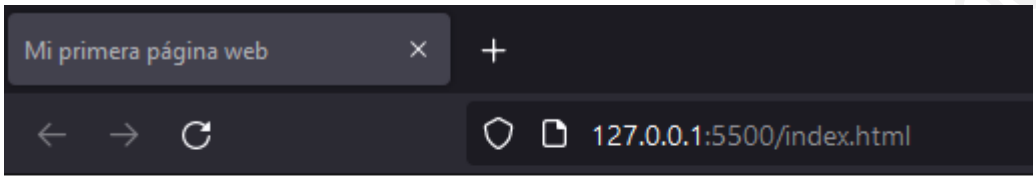
```
<table>
 <tr>
  <td>Celda 1</td>
  <td>Celda 2</td>
  <td>Celda 3</td>

 </tr>

 <tr>
  <td>Celda 4</td>
  <td>Celda 5</td>
  <td>Celda 6</td>
 </tr>
</table>
```



Celda 1 Celda 2 Celda 3
Celda 4 Celda 5 Celda 6

Ahora que disponemos de la estructura básica de una tabla, vamos a ampliarla.
Podemos convertir una celda normal **<td>** en una celda de encabezado, una celda destacada, cambiando la etiqueta **<td>** por **<th>**. Veamos ejemplo:

```
<table>
 <tr>
  <th>Encabezado 1</th>
  <th>Encabezado 2</th>
  <th>Encabezado 3</th>
 </tr>

 <tr>
  <td>Celda 1</td>
  <td>Celda 2</td>
  <td>Celda 3</td>
 </tr>
</table>
```

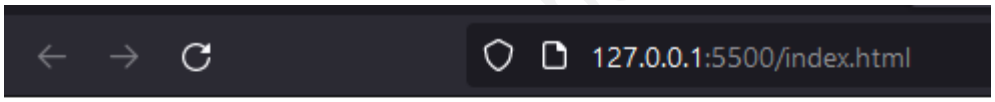


Encabezado 1	Encabezado 2	Encabezado 3
Celda 4	Celda 5	Celda 6

Como podemos observar, las celdas de encabezado ahora destacan más sobre las celdas normales. No te preocupes por el estilo ahora mismo, cuando entremos a trabajar CSS veremos cómo hacer nuestro contenido más sexy. De momento, continuemos manteniendo el foco en la semántica, y en ella las tablas todavía tienen más contenido por explotar.

Uno de los añadidos que podemos aplicar a una tabla es agregarle un título. Esto podemos hacerlo con la etiqueta **<caption>**:

```
<table>
 <caption>Mi primera tabla</caption>
 <tr>
    <th>Encabezado 1</th>
    <th>Encabezado 2</th>
   <th>Encabezado 3</th>
 </tr>

 <tr>
  <td>Celda 1</td>
  <td>Celda 2</td>
  <td>Celda 3</td>
 </tr>
 <tr>
  <td>Celda 4</td>
  <td>Celda 5</td>
  <td>Celda 6</td>
 </tr>
 <tr>
  <td>Celda 7</td>
  <td>Celda 8</td>
  <td>Celda 9</td>
 </tr>
</table>
```

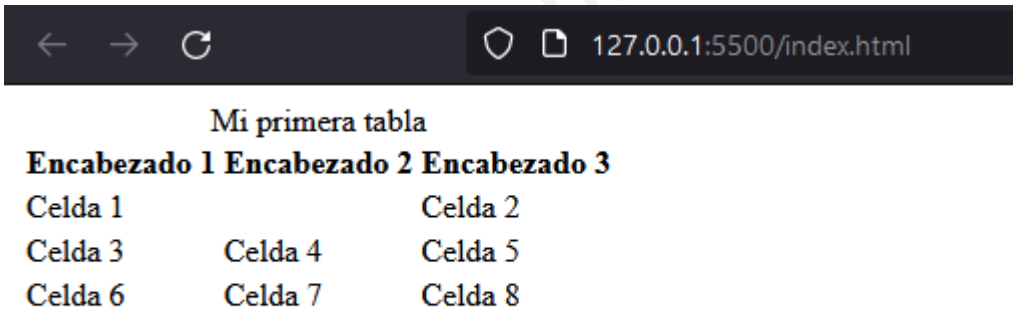


Mi primera tabla

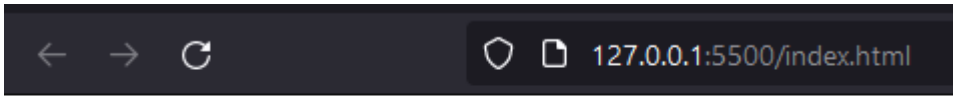
Encabezado 1	Encabezado 2	Encabezado 3
Celda 1	Celda 2	Celda 3
Celda 4	Celda 5	Celda 6
Celda 7	Celda 8	Celda 9

Veamos ahora algunos atributos de las celdas. Y es que cuando estamos estructurando una tabla, suele ser común hacer que una celda de dicha tabla ocupe más de una celda, ya sea vertical y horizontalmente. Lo que comúnmente llamaríamos *"unir celdas"*. Para esto podemos usar los atributos **colspan** y **rowspan** de la celda, indicándoles como valor cuántas celdas desean expandirse, realizando **colspan** una expansión horizontal y **rowspan** una expansión vertical. Veámoslo en un ejemplo:

```
<table>
 <caption>Mi primera tabla</caption>
 <tr>
    <th>Encabezado 1</th>
    <th>Encabezado 2</th>
   <th>Encabezado 3</th>
 </tr>

 <tr>
  <td colspan="2">Celda 1</td>
  <td>Celda 2</td>
 </tr>
 <tr>
  <td>Celda 3</td>
  <td>Celda 4</td>
  <td>Celda 5</td>
 </tr>
 <tr>
  <td>Celda 6</td>
  <td>Celda 7</td>
  <td>Celda 8</td>
 </tr>
</table>
```

Excepcionalmente, voy a permitirme añadir un estilo de borde para que podamos apreciar mejor la expansión de la celda número 1:



Mi primera tabla

Encabezado 1	Encabezado 2	Encabezado 3
Celda 1		Celda 2
Celda 3	Celda 4	Celda 5
Celda 6	Celda 7	Celda 8

Como podemos observar, aprovechando los atributos de **colspan** y **rowspan** podemos modificar el espacio que ocupa cada celda en la tabla.

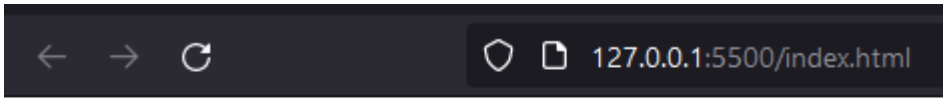
Supongamos ahora que queremos complicar todavía más la estructura de nuestra tabla. A veces, suele ser interesante definir diferentes áreas dentro de nuestra tabla, distinguiendo una primera parte de la tabla o cabecera, un cuerpo central y un pie de tabla. A estas alturas del contenido, donde aún no hemos visto herramientas de diseño, quizás esta división pueda parecer algo forzada, pero cuando entremos a trabajar CSS y el estilo de nuestra página web, poder realizar este tipo de divisiones nos resultará realmente interesante.

Para declarar dichas definiciones podemos agrupar filas dentro de las etiquetas **<thead> <tbody>** y **<tfoot>**, que corresponden a la cabecera de la tabla, su cuerpo, y el pie de tabla respectivamente. Veámoslo una vez más en un ejemplo:

```
<table>

  <thead>
   <tr>
    <th></th>
    <th colspan="2">Juan</th>
    <th colspan="2">Raúl</th>
   </tr>
   <tr>
    <th>Fecha</th>
    <th>Ingresos</th>
    <th>Gastos</th>
    <th>Ingresos</th>
    <th>Gastos</th>
   </tr>
  </thead>

  <tbody>
   <tr>
    <th>15/07/22</th>
    <td>€120,00</td>
```

```
    <td>€50,00</td>
    <td>€10,50</td>
    <td>€5,00</td>
   </tr>
   <tr>
    <th>15/07/22</th>
    <td>€120,00</td>
    <td>€50,00</td>
    <td>€10,50</td>
    <td>€5,00</td>
   </tr>
   <tr>
    <th>15/07/22</th>
    <td>€120,00</td>
    <td>€50,00</td>
    <td>€10,50</td>
    <td>€5,00</td>
   </tr>
   <tr>
    <th>15/07/22</th>
    <td>€120,00</td>
    <td>€50,00</td>
    <td>€10,50</td>
    <td>€5,00</td>
   </tr>
   <tr>
    <th>15/07/22</th>
    <td>€120,00</td>
    <td>€50,00</td>
    <td>€10,50</td>
    <td>€5,00</td>
   </tr>
  </tbody>

  <tfoot>
   <tr>
    <th>Total</th>
    <td colspan="2">€70,00</td>
    <td colspan="2">€5,50</td>
   </tr>
  </tfoot>

</table>
```

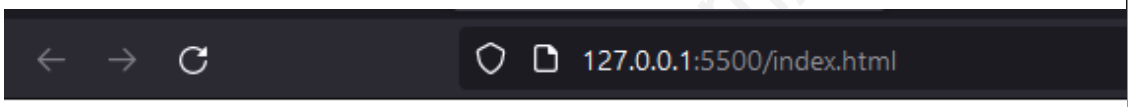

Resultado añadiendo bordes para apreciarlo mejor:



	Juan		Raúl	
Fecha	**Ingresos**	**Gastos**	**Ingresos**	**Gastos**
15/07/22	€120,00	€50,00	€10,50	€5,00
15/07/22	€120,00	€50,00	€10,50	€5,00
15/07/22	€120,00	€50,00	€10,50	€5,00
15/07/22	€120,00	€50,00	€10,50	€5,00
15/07/22	€120,00	€50,00	€10,50	€5,00
Total	€70,00		€5,50	

Hora de practicar

Prueba a realizar la siguiente tabla (sin bordes):



Registro de horas

	Juan		Raúl		Sonia		Pedro	
Día	**Entrada**	**Salida**	**Entrada**	**Salida**	**Entrada**	**Salida**	**Entrada**	**Salida**
Lunes	9:00	16:00	8:30	15:30	9:00	16:00	8:30	15:30
Martes	9:00	16:00	8:30	15:30	9:00	16:00	8:30	15:30
Miércoles	9:00	16:00	8:30	15:30	9:00	16:00	8:30	15:30
Jueves	9:00	16:00	8:30	15:30	9:00	16:00	8:30	15:30
Viernes	9:00	16:00	8:30	15:30	9:00	16:00	8:30	15:30
Fecha	18/04/22							

RECUERDA: Puedes usar EMMET para tus construcciones. Por ejemplo, y siguiendo la lógica descrita en el capítulo anterior, si escribimos:

```
table>tr*5>td*4
```

Estaremos creando una tabla, dentro de la cual habrá 5 filas, dentro de las cuáles habrá 4 celdas. Es decir:

```
<table>
 <tr>
  <td></td>
  <td></td>
  <td></td>
  <td></td>
 </tr>
 <tr>
  <td></td>
  <td></td>
  <td></td>
  <td></td>
 </tr>
 <tr>
  <td></td>
  <td></td>
  <td></td>
  <td></td>
 </tr>
 <tr>
  <td></td>
  <td></td>
  <td></td>
  <td></td>
 </tr>
 <tr>
  <td></td>
  <td></td>
  <td></td>
  <td></td>
 </tr>
</table>
```


¡Y eso es todo dentro de este bloque! Recuerda practicar todo lo posible hasta familiarizarte con lo visto antes de continuar. Puede que aún no luzca nada espectacular, pero ya llegaremos a ello.

Si quieres repasar en vídeo todo este bloque, o te has quedado atascado, puedes ver todo el proceso en el canal GOGODEV de YouTube.

Más adelante nos preocuparemos de que nuestra web sea guapa. Ahora mismo nos estamos centrando en que tengan buen fondo.

</TABLAS>

Autor: Juan José Ruiz a.k.a GOGODEV
YouTube: https://youtube.com/c/gogodev
Twitter: https://twitter.com/JJRuizEmpresa (@JJRuizEmpresa)

IMPORTANTE

El contenido de este pdf lo tienes también en vídeo en el canal de YouTube https://youtube.com/c/gogodev , dentro de la lista de reproducción: Curso de HTML.

Todo el contenido formativo es gratuito, no olvides pasarte y suscribirte para no perderte nada, y así nos apoyas :)

<CAPÍTULO .03>

LA MAGIA DEL MULTIMEDIA

ETIQUETAS RELACIONADAS CON LA ESCRITURA

</CAPÍTULO .03>

LA MAGIA DEL MULTIMEDIA

<CÓMO_CONSTRUIR_RUTAS_RELATIVAS>

A lo largo de este capítulo vamos a comenzar a añadir contenido multimedia a nuestras construcciones. Esto es: imágenes, vídeos, audios… Por ende, vamos a necesitar indicar en dichas etiquetas dónde se encuentran dichos recursos. Es decir, vamos a tener que mostrarle a nuestro código la ruta para que el navegador pueda cargar correctamente la información.

Y por ello, vamos a comenzar este capítulo hablando sobre cómo se construye una ruta, los tipos de rutas existentes y cómo usarlas de forma apropiada para que, cuando entremos en materia, todo vaya como la seda.

Tipos de rutas

Existes dos tipos de rutas principalmente:

- **Rutas absolutas:** Estas son las más fáciles de escribir, ya que contienen el camino completo hacia el recurso. No importa dónde se encuentre el archivo que hace la llamada al recurso, la ruta absoluta siempre se escribe igual. Por ejemplo, si queremos hacer uso de una imagen que se encuentra en https://picsum.photos/200/300 , su ruta absoluta sería exactamente esa: https://picsum.photos/200/300 , expresando la dirección completa. Usaremos este tipo de rutas cuando llamemos a recursos externos que se encontrarán fuera de nuestro sitio web (una foto alojada en Google, por ejemplo) Para el resto de ocasiones (la gran mayoría) usaremos siempre las rutas relativas, ya que las absolutas pueden conducirnos a error.

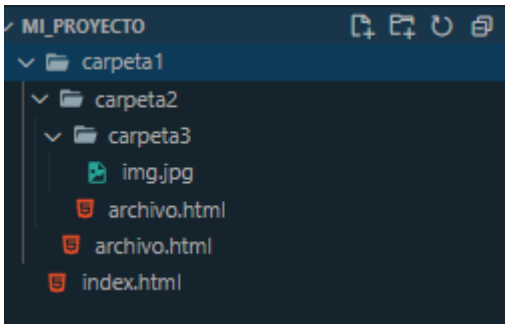
- **Rutas relativas:** Son las rutas en las cuales no especificamos el camino completo para llegar a ellas, si no que expresamos cómo llegar al recurso desde la posición en la que se encuentra el archivo de código que estamos escribiendo. Un ejemplo de ruta relativa sería: *img/equipo/miembro1.jpg* En esta ruta estamos indicando que el archivo se encuentra, desde la ubicación en la que se encuentra el documento que estamos escribiendo, accediendo a una carpeta llamada *img*. Una vez dentro accediendo a una carpeta llamada *equipo* y, dentro de esta, el archivo es la imagen que se llama *miembro1.jpg*.

Cómo se construye una ruta

Profundicemos en cómo escribir una ruta relativa:

Para adentrarnos en una carpeta colocamos el nombre de la carpeta, y posteriormente usamos el símbolo /

Por ejemplo, en la siguiente estructura, donde la carpeta 3 está dentro de la carpeta 2, y la carpeta 2 dentro de la 1…



Si queremos usar la imagen img.jpg que se encuentra en la carpeta3 en el archivo index.html que se encuentra en el directorio raíz del proyecto, la ruta será:

carpeta1/carpeta2/carpeta3/img.jpg

Sin embargo, si queremos usar ese mismo recurso desde el documento archivo.html que se encuentra en la carpeta2, la ruta será:

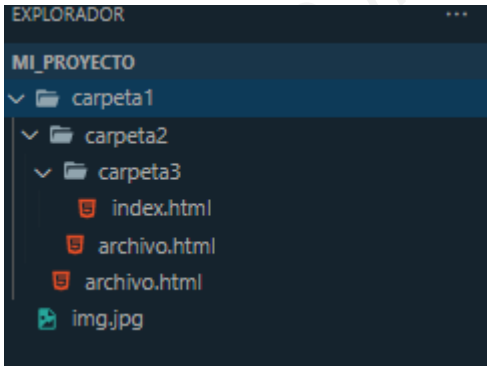
carpeta3/img.jpg

Ya que siempre escribimos la ruta *saliendo* desde la posición del archivo en el que estamos escribiendo la ruta.

Muy bien, ya sabemos entrar en carpetas. Ahora la duda que podría surgir sería ¿y cómo salimos de ellas?

Muy sencillo. Para salir de una carpeta, usamos la expresión ../

Por ejemplo, supongamos la siguiente estructura de archivos:



En esta ocasión, es el index.html el que se encuentra dentro de la carpeta3, mientras que el recurso (img.jpg) se encuentra en el directorio raíz. Por tanto, para construir la ruta correctamente deberíamos indicar:

"Sal de la carpeta3, sal de la carpeta2, sal de la carpeta1, el archivo se llama img.jpg"

Entonces, si para salir de una carpeta al nivel superior usamos ../ la solución sería:

../../../img.jpg

Y así, combinando el acceso y la salida de los diferentes directorios, podemos localizar de forma relativa cualquier recurso de nuestro proyecto.

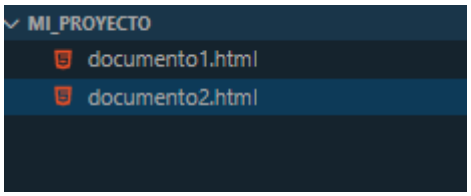
RECOMENDACIÓN: Crea un proyecto nuevo y practica escribiendo cómo serían las diferentes rutas relativas antes de continuar.

</CÓMO_CONTRUIR_RUTAS_RELATIVAS>

<ENLACES>

Ahora que ya sabemos escribir rutas, es hora de afrontar nuestra primera herramienta de navegación: los enlaces.

Los enlaces, o hipervínculos, o links, o anclas, son la base de toda estructura web, ya que nos permiten navegar entre los diferentes documentos. Cuando pulsamos sobre un enlace, el navegador saldrá del documento actual, y cargará el documento al que referencia dicho enlace.
Creemos un proyecto con dos documentos html:



Cada uno de estos documentos, contendrá la estructura básica de un documento HTML, tal y como vimos en el capítulo 01.

Ahora, vamos a crear en el cuerpo del documento1.html un enlace que, al ser pulsado, nos lleve al documento2.html.

En el documento2.html escribiremos en su cuerpo un título <h1> (por ejemplo) que diga: “Hola, soy el documento2”. Así, podremos comprobar que nuestro enlace funciona de una forma rápida y visual.
Es hora de realizar el enlace. Para ello, usamos la etiqueta <a>. Además, completaremos dicha etiqueta usando el atributo **href**, donde indicaremos la ruta relativa al documento que debe abrir. Así pues, y siguiendo el ejemplo, en el documento1.html escribiremos:

```
<a href="documento2.html">Pulsa aquí para ver el documento 2</a>
```

Guarda ambos documentos y comprueba. ¡Felicidades! Has creado tu primer enlace.
Por último, vamos a ampliar un poco más el concepto de enlace con un segundo atributo, el atributo **target**. Este atributo nos permite indicar dónde queremos que se abra el documento. Si no especificamos nada, como hemos podido comprobar, el enlace se abre en el lugar donde se encontraba el documento anterior. Sin embargo, podemos especificar a través de este atributo que éste se abra, por ejemplo, en una pestaña nueva del navegador, sin cerrar la pestaña actual:

- **target="_blank"** Abre el documento en una nueva pestaña.
- **target="_self"** El valor por defecto. Abre el enlace en el marco actual.
- **target="_parent"** Abre el enlace en el marco padre donde ocurre la acción.
- **target="_top"** Abre el enlace en el cuerpo completo de la página.

En la mayoría de ocasiones, se usará su valor por defecto, o el valor "_blank". Las otras opciones se encuentran en desuso y no son aconsejables.

```
<a href="documento2.html" target="_blank">Abrir en una nueva ventana.</a>
```

***IMPORTANTE**: El href indica la ruta relativa. En este ejemplo la ruta relativa es directamente el nombre del documento ya que ambos se encuentran en el mismo lugar, pero si este se encuentra en otra ubicación, debemos expresar correctamente la ruta.*

</ENLACES>

<IMÁGENES>

Continuamos ahora con las imágenes. Una vez que conocemos las rutas relativas, incluir imágenes te va a resultar muy sencillo. Para incluir una imagen, usamos la etiqueta **<img>** y dentro usamos el atributo **src** para indicar la ruta donde se encuentra dicha imagen. Como la etiqueta no contendrá nada en su interior (su contenido será el propio recurso cargado, podemos cerrar la etiqueta en la misma línea:

```
<img src="carpeta1/imagen.jpg" />
```

A dicha etiqueta podemos agregarle también un atributo alt que explique qué contiene la imagen. Esto es muy útil para los robots de búsqueda y el posicionamiento SEO de una página:

```
<img src="carpeta1/imagen.jpg" alt="balón de fútbol" />
```

Además, y si lo deseamos, podemos especificar cual será la anchura **width** y altura **height** de esta con sendos atributos, si bien cuando entremos en las secciones de diseño ahondaremos más en profundidad en este aspecto:

```
<img src="carpeta1/imagen.jpg" alt="balón de fútbol" width="600" height="400" />
```

Hora de practicar

Crea un nuevo proyecto y coloca diferentes imágenes en rutas distintas.
Prueba a crear un documento HTML que las muestre todas.

</IMÁGENES>

<VÍDEOS>

Una vez vistas las imágenes, es hora de dar cabida a la inclusión de vídeos en nuestras construcciones web.

La etiqueta de vídeo, así como la etiqueta de audio, forman parte de la nueva ampliación del estándar de HTML, y permanecen como etiquetas válidas desde la versión 5 (HTML5), por lo que son una de las novedades más interesantes incorporadas en la versión que permanece vigente a fecha de hoy. Al igual que con las imágenes, necesitaremos valernos de un atributo **src** mediante el cuál indicaremos la ubicación donde se encuentra el archivo.

```
<video src="mivideo.mp4"> Vídeo no soportado... </video>
```

Como puedes ver, la etiqueta vídeo funciona de forma similar a la etiqueta de imagen. Además, los atributos que podemos aplicar a una imagen, tales como la altura **height** o anchura **width** también nos serán válidos para la etiqueta de vídeo.

Si el navegador que utiliza el usuario es antiguo y no soporta vídeo, podemos colocar un mensaje que se mostrará para él dentro de la propia etiqueta, tal y como podemos ver en el ejemplo.

Sigamos profundizando en su uso. Uno de los puntos a tener en cuenta cuando colocamos vídeos en nuestra web, son los formatos de vídeo. Si bien el formato .mp4 es el más extendido y usado, existen otro tipo de formatos como el **.ogg** y el **.webm** que también son útiles. Si disponemos de nuestro vídeo en diferentes formatos, podemos especificar en la etiqueta de vídeo la ruta a todos ellos para que el navegador utilice la versión más favorable. Esto lo hacemos de la siguiente manera:

```
<video>
  <source src="mivideo.mp4"  type="video/mp4">
  <source src="mivideo.ogg"  type="video/ogg">
  <source src="mivideo.webm"  type="video/webm">
</video>
```

Una vez hemos insertado el vídeo, es hora de analizar algunos atributos de interés. El primero de ellos es añadir un reproductor, es decir, añadir controles al vídeo para que el usuario pueda pausarlo, reanudarlo, ajustar el volumen, etcétera. Para añadir el reproductor bastará con hacer uso del atributo

controls de la etiqueta. Si lo indicamos, el vídeo aparecerá dentro de un reproductor de vídeo predeterminado según el navegador:

```
<video controls>
  <source src="mivideo.mp4"  type="video/mp4">
  <source src="mivideo.ogg"  type="video/ogg">
  <source src="mivideo.webm"  type="video/webm">
</video>
```

Otra de las características que podemos indicar es si queremos que el vídeo inicie su reproducción automáticamente. Para ello, lo marcamos con el atributo de **autoplay**:

```
<video controls autoplay>
  <source src="mivideo.mp4"  type="video/mp4">
  <source src="mivideo.ogg"  type="video/ogg">
  <source src="mivideo.webm"  type="video/webm">
</video>
```

Por último, vamos a ver el atributo **loop**. Con él, indicamos que el vídeo debe reproducirse en bucle, es decir, que cuando finalice debe volver a comenzar de forma automática:

```
<video controls autoplay loop>
  <source src="mivideo.mp4"  type="video/mp4">
  <source src="mivideo.ogg"  type="video/ogg">
  <source src="mivideo.webm"  type="video/webm">
</video>
```

</VÍDEOS>

<AUDIO>

Veamos ahora la etiqueta de audio. Al igual que ocurre con la etiqueta de vídeo, **<audio>** fue introducida en la ampliación del estándar en su versión 5, y su funcionamiento es muy similar al de la etiqueta vídeo:

```
<audio src="miaudio.mp3">
  Tu navegador no soporta la reproducción de audio.
</audio>
```

Al igual que ocurre con el vídeo, también podemos incluir el audio en diferentes formatos, siendo los más usados **.mp3**, **.ogg**, **.wav**, **.aac** y **.opus**:

```
<audio>
  <source src="miaudio.mp3"  type="audio/mp3">
  <source src="miaudio.wav"  type="audio/wav">
</audio>
```

Por último, también podemos añadirle los atributos de **autoplay**, **loop** y **controls**:

```
<audio controls  autoplay  loop>
   <source src="miaudio.mp3"  type="audio/mp3">
</audio>
```

</AUDIO>

<IFRAME>

En última instancia, y dentro de las etiquetas multimedia, vamos a ver el **<iframe>**. El **iframe** es una etiqueta que debe usarse con cuidado, y que empieza a quedar en desuso. Sin embargo, aún sigue siendo útil en ciertos escenarios. El **iframe** nos permite incrustar un html dentro de otro. Por ejemplo, incrustar otra página web dentro de la nuestra en una sección determinada.

Esto puede acarrear problemas de seguridad, por lo que sólo usaremos **iframes** con sitios confiables, por ejemplo: *YouTube*, *iVoox* o *Spotify*.

A día de hoy, el principal uso de los **iframe** es incluir vídeos de *YouTube* o *Vimeo*, o audios de *Spotify* o *iVoox* dentro de tu página web. Y es que habrás observado que, cuando accedemos a YouTube por ejemplo, dentro de las opciones de compartir cuando vemos un vídeo, una de ellas nos ofrece un enlace, y otra un código **iframe** para que podamos pegar el contenido en nuestra página.

Al igual que ocurre con las etiquetas de **audio** y **video**, podemos aplicar una anchura **width** y altura **height** a nuestro **iframe**.

Ejemplo de iframe de YouTube:

```
<iframe width="560" height="315" src="https://www.youtube.com/embed/GLIbmirxIa0" title="YouTube video player" frameborder="0" allow="accelerometer; autoplay; clipboard-write; encrypted-media; gyroscope; picture-in-picture" allowfullscreen></iframe>
```

En este ejemplo, podemos observar que el **iframe** tiene unas dimensiones de 560 pixeles de ancho y 315 de alto. Nos aparece también la ruta donde se encuentra el vídeo y el título del mismo.

Por último, especifica que el **iframe** no tendrá borde y que se permitirá ampliar a pantalla completa. El resto de atributos son propios de la plataforma *YouTube*, especificados dentro de **allow**.

¡Y eso es todo dentro de este bloque! Recuerda practicar todo lo posible hasta familiarizarte con lo visto antes de continuar. Como habrás podido observar, y aún a falta de diseño, nuestra web ahora comienza a *"hacer cosas"*. Sigamos dando pasos hacia delante.

Si quieres repasar en vídeo todo este bloque, o te has quedado atascado, puedes ver todo el proceso en el canal YouTube de GOGODEV.

Lo importante no es el final. Es disfrutar del camino, pues como desarrollador web, siempre estarás caminando.

</IFRAME>

Autor: Juan José Ruiz a.k.a GOGODEV
YouTube: https://youtube.com/c/gogodev
Twitter: https://twitter.com/JJRuizEmpresa (@JJRuizEmpresa)

IMPORTANTE

El contenido de este pdf lo tienes también en vídeo en el canal de YouTube https://youtube.com/c/gogodev , dentro de la lista de reproducción: Curso de HTML.

Todo el contenido formativo es gratuito, no olvides pasarte y suscribirte para no perderte nada, y así nos apoyas :)

<CAPÍTULO .04>

DIME LO QUE PIENSAS

CONSTRUCCIÓN DE FORMULARIOS PARA INTERACTUAR CON EL USUARIO.

</CAPÍTULO .04>

DIME LO QUE PIENSAS

<FORMULARIOS>

Hasta ahora, todas las etiquetas que hemos ido aprendiendo dentro de HTML estaban destinadas a mostrar información al usuario, manteniendo una comunicación unidireccional. Es hora de incorporar a nuestras construcciones todo el conjunto de herramientas asociadas a los formularios, que permitirán al usuario introducir datos y, por tanto, comunicarse con su entorno web.

Con los formularios no solo podemos hacer secciones de contacto, si no cajas de comentarios, editores de contenido, subir imágenes y mucho más.

A lo largo de este capítulo, vamos a desglosar su conjunto de etiquetas más importantes. Pongámonos manos a la obra:

Cada vez que vayamos a definir una zona de interacción donde vamos a recoger datos que el usuario va a tener que introducir, sean estos del tipo que sean, deberemos crear un formulario valiéndonos de la etiqueta **<form>**. Posteriormente, dentro de esta etiqueta, anidaremos a su vez las etiquetas de todos aquellos elementos que conformarán dicho formulario: Un campo de introducción de texto, una caja de comentarios, unas cajas de selección…

```
<form>
   <!-- Contenido del formulario -->
</form>
```

Una vez tenemos colocada la etiqueta, existen dos atributos esenciales que deben acompañar siempre a la etiqueta: El **method** y el **action**.

Comencemos por el action:

Este, es un atributo que indica la ruta donde se encuentra el archivo que va a procesar los datos. Y es que, cuando el usuario cumplimenta y envía unos datos, nuestro sistema debe ser capaz de recepcionar y procesar los datos recibidos, además de actuar en consecuencia.

El procesado y tratamiento de datos lo veremos más adelantes en cursos futuros del canal, en los que nos dedicaremos a trabajar el *backend* de nuestras aplicaciones web, que es la parte de la programación dedicada a tales efectos, y que estará escrito en un lenguaje de programación concreto. De momento, nos basta con saber que el **action** contiene la ruta donde se encuentra el archivo que procesará los datos recibidos.

El segundo de estos atributos es el **method**, que contiene un verbo que indica cómo viajarán los datos al fichero/función que los procesará. Existen diferentes verbos con particularidades distintas, que forman parte del temario de *backend* de la colección: GET, POST, PUT, DELETE.

De momento vamos a quedarnos con los dos principales:

- **GET**: Usamos este método cuando estamos enviando datos que NO son sensibles. Es decir, cuando la información que está escribiéndonos el usuario no es privada. Por ejemplo: una barra de búsqueda (una caja de buscar). Esto es debido a que cuando usamos GET la información viajará hacia el archivo de procesado dentro de la url, y por lo tanto es susceptible de ser leída por terceros.

- **POST**: Al contrario que ocurre con GET, POST encriptará la información en la petición, y no será visible por terceros. Usaremos esta opción cuando estamos enviando datos sensibles. Por ejemplo: un formulario de registro o acceso donde se está escribiendo la contraseña de acceso.

Así pues, la etiqueta **form** nos quedaría de la siguiente forma:

```
<form method="POST" action="miarchivo.php">
   <!-- Contenido del formulario -->
</form>
```

Ya conocemos cómo declarar una zona de formulario. Es hora de comenzar a cumplimentarla con los diferentes elementos de los que disponemos.

</FORMULARIOS>

<INPUTS>

Comencemos por uno de los elementos más comunes en los formularios: las cajas de texto. Seguro que, a lo largo de tu vida, en internet, has cumplimentado numerosas de ellas.

Para crear una caja de texto, usamos la etiqueta <input> y a dicha etiqueta, debemos darle un atributo *type*, para indicar qué tipo de caja de texto será.

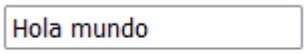
Además, a todas las etiquetas que vayan dentro del formulario, tanto a las cajas de texto como las demás, deberemos darle una propiedad *name*, otorgándoles un nombre identificativo a cada una (consistente en una única palabra que no contenga tildes ni caracteres especiales. Podrá ser el que nosotros prefiramos) Esta no afecta al formulario, pero la parte de la web que se encarga de tratar los datos lo necesitará más adelante.

Puede parecer complicado, pero ya verás que es muy fácil. Veamos las diferentes opciones:

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

Vacía

El usuario puede escribir texto y números en ella




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

Vacía

El usuario puede escribir sólo números en ella


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

Vacía

Lo que escribe el usuario aparece cifrado para que no pueda leerse

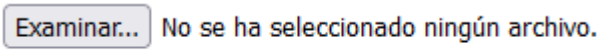

```
<input type="date" name="caja_para_seleccionar_fechas">
<input type="date" name="caja_para_seleccionar_horas">
```

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

Vacía

El usuario solo puede introducir direcciones de email válidas en ella


```
<input type="file" name="mi_caja_para_subir_un_archivo">
```



Si usamos este tipo de input, destinado a subir archivos, al formulario deberemos indicárselo añadiendo a la etiqueta ***form*** el atributo ***enctype*** con el valor ***"multipart/form-data"*** :

```
<form method="miarchivo.php" action="POST" enctype="multipart/form-data">
```

Esto lo hacemos para que, al enviar la información, el formulario codifique correctamente la información del archivo.

¿Ves qué fácil? Ahora, deberíamos indicarle al usuario qué debe introducir en cada una de ellas. Para esto, tanto en las cajas <input> como en el resto de elementos de un formulario, podemos usar dos herramientas diferentes:

- La etiqueta ***<label>,*** para colocar un texto explicativo.
- El atributo ***placeholder*** en la propia etiqueta, para colocar un texto de ayuda dentro del input.

Veámoslo una vez más con un ejemplo:

```
<label>Email:</label>
<input type="email" name="email" placeholder="Ejemplo: john@doe.com">
```

Resultado:

</INPUTS>

<SELECTS>

Ya sabemos colocar en nuestros formularios cajas de textos para que el usuario introduzca información. Ahora vamos a conocer la etiqueta ***select***. Esta, nos permitirá crear una caja con un menú desplegable, donde el usuario tendrá que seleccionar entre una de las opciones disponibles.

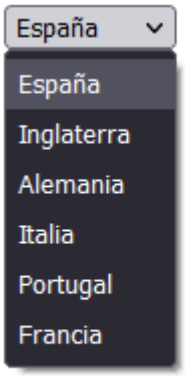
Para establecer un área de selección, marcamos la etiqueta ***select***, a la cual le aplicaremos, al igual que los inputs y el resto de formularios, un atributo ***name*** con un nombre escogido por nosotros, para que posteriormente la información pueda ser tratada. Una vez hecho esto, anidaremos dentro del ***select*** diferentes etiquetas ***option***, cada una de ellas con el texto de cada una de las opciones. Así mismo, las etiquetas ***option*** contarán con un atributo ***value***, que será el valor que el formulario enviará al sistema como seleccionado. Esto hace que el valor que enviamos y el texto que se muestra al usuario puedan ser diferentes.

Veamos un ejemplo:

```
<select name="pais">
    <option value="es">España</option>
    <option value="en">Inglaterra</option>
    <option value="ge">Alemania</option>
    <option value="it">Italia</option>
    <option value="pt">Portugal</option>
    <option value="fr">Francia</option>
</select>
```

Resultado visual:

Y cuando pulsamos sobre el elemento:



En este ejemplo, al pulsar sobre Italia, por ejemplo, el formulario enviará al sistema que el usuario en el campo ***"pais"*** ha seleccionado ***"it"*.**

Al campo select, además, podemos añadirle el atributo ***multiple***. De esta forma, el usuario en lugar de poder seleccionar una única opción de entre las disponibles, podrá seleccionar múltiples de ellas.

```
<select name="pais" multiple>
   <option value="es">España</option>
   <option value="en">Inglaterra</option>
   <option value="ge">Alemania</option>
   <option value="it">Italia</option>
   <option value="pt">Portugal</option>
   <option value="fr">Francia</option>
</select>
```

Select con ***multiple*** en nuestra página web:

En este ejemplo, el usuario ha seleccionado Inglaterra y Alemania.

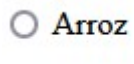
</SELECTS>

<RADIOS>

Continuamos avanzando con los diferentes elementos que podemos añadir a un formulario. En esta ocasión, vamos a ver los botones de radio. Los ***'radio'*** son botones circulares que podemos marcar o no. Normalmente, se usan para darnos a escoger una de entre diferentes opciones. Para ello, usamos la instrucción:

```
<input type="radio" id="arroz" name="comida_fav" value="Arroz">
<label for="arroz">Arroz</label>
```

Esto, nos daría el siguiente resultado:



Donde la casilla circular puede ser o no marcada.

Detengámonos a analizar la instrucción:

```
input type="radio"
```

Con esta parte, indicamos que vamos a pintar un botón de radio.

```
id="arroz"
```

En HTML, a todas las etiquetas podemos darle un ***id*** con el valor que deseemos. En este caso, lo usamos para indicarle posteriormente a la etiqueta a qué botón de radio pertenece, a través de su atributo ***for***.

```
name="comida_fav"
```

En los botones de radio, el ***name***, además de servir al sistema para identificar el contenido que se le está enviando, también sirve para identificar los grupos de opciones. Por ejemplo, si tenemos cuatro botones de radio, los cuatro con el ***name*** de "***comida_fav***" el formulario los identificará como un grupo de opciones, y solo dejará marcar uno de ellos. Si el usuario trata de seleccionar un segundo, el primero se desmarcará.

```
value="Arroz"
```

Por último, tenemos el ***value***. Al igual que ocurre con el resto de etiquetas de formulario, éste será el valor que se enviará al sistema si el usuario escoge dicha opción.

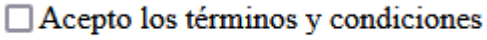
</RADIOS>

<CHECKBOX>

Pasamos ahora, de los botones de radio a su versión cuadrada, el ***checkbox***. Estoy seguro que, navegando por internet te has encontrado con él en numerosas ocasiones, y es que el ***checkbox*** es esa cajita que marcamos cada vez que pulsamos sobre ese mensaje de "acepto los términos y condiciones". La forma de construirlo es muy similar a los botones de radio. Veamos cómo hacerlo:

```
<input type="checkbox" name="terminos" value="yes"><label>Acepto los términos y condiciones</label>
```

Resultado:

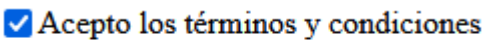


Como puedes observar, los atributos y fórmulas para construir los elementos de un formulario son siempre similares. En esta ocasión, indicamos que el input es de tipo ***checkbox***, le damos un nombre y el valor que se enviará si lo marca.

Como aspecto especial, al ***checkbox*** podemos incluirle el atributo ***checked***. Si lo hacemos, la casilla aparecerá marcada por defecto:

```
<input type="checkbox" name="terminos" value="yes" checked><label>Acepto los términos y condiciones</label>
```

Resultado:



</CHECKBOX>

<RANGOS>

Es hora de ver los rangos. Con ellos, podemos pintar una barra deslizadora para que el usuario la ajuste al valor numérico deseado. Para ello, expresaremos los atributos ***min*** y ***max***, de cara a marcar el valor mínimo y máximo del rango respectivamente.

```
<input type="range" name="mi_porcentaje" min="0" max="100" />
```

Resultado:

</RANGOS>

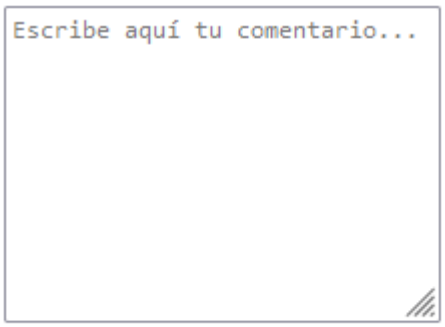
<TEXTAREA>

A la hora de escribir textos, es posible que una única línea como nos ofrece el ***input text*** no sea suficiente. Para esas ocasiones en las que estamos esperando que el usuario introduzca una mayor cantidad de texto, como puede ser por ejemplo un comentario en una página de reseñas, usamos el área de texto. Para darle las dimensiones apropiadas a la caja, podemos usar los atributos ***cols*** y ***rows***, columnas y filas respectivamente. Y al igual que ocurre con los ***input text***, podemos aplicarle un ***placeholder***. Por último, si deseamos que el texto contenga un mínimo de caracteres o un máximo, podemos usar los atributos de ***minlength*** (longitud mínima) y ***maxlength*** (longitud máxima)

Veámoslo con un ejemplo:

```
<textarea name="comentario" cols="30" rows="10" minlength="100" maxlength="500" placeholder="Escribe aquí tu comentario..."></textarea>
```

El resultado:



</TEXTAREA>

<VALIDACIÓN>

Por último, y antes de continuar con el envío, debemos saber cuáles de los campos son obligatorios que el usuario cumplimente, ya que en un formulario algunos de ellos pueden ser opcionales. Para ello, bastará con marcar con el atributo ***required*** a aquellos campos que vayan ser requeridos de forma obligatoria. El resto, quedarán automáticamente como opcionales por defecto. De esta forma, el navegador no dejará enviar el contenido si los campos marcados como requeridos no están debidamente rellenos, avisando al usuario de cuales de los no cumplimentados deberían estarlo, con el mensaje “debes rellenar este campo”.

</VALIDACIÓN>

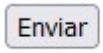
<ENVÍOS>

Ahora ya disponemos de un repertorio suficiente para poder construir nuestros formularios. Nos resta ahora, colocar el botón de enviar. Para ello, usaremos la etiqueta input ***submit***, colocando el texto deseado del botón en el atributo ***value***.

Veamos un ejemplo:

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

Resultado:



Y ahora sí, ya sabemos construir las estructuras HTML necesarias para que los usuarios puedan interactuar con nuestra página.

CONSEJO: Llegados a este punto, te puedes preguntar: ¿Son estas todas las etiquetas disponibles en HTML para formularios? La respuesta es un claro y rotundo no. Estas, son las más habituales, las que, en tu trabajo como desarrollador web, vas a estar escribiendo prácticamente a diario. Sin embargo, existen muchas otras. Por ello, es importante destacar, que NO debes aprenderte todas las etiquetas de memoria. Ningún desarrollador lo hace. A fuerza de usarlas, acabarás interiorizando las más comunes, pero no es posible ni recomendable tratar de conocerlas todas de memoria. El buen desarrollador siempre se preocupará de saber escribir buenas estructuras, de mantener el código limpio y de saber solucionar problemas. Para todas aquellas etiquetas más especiales que te puedan resultar de utilidad en momentos más puntuales, o para comportamientos que quieras aplicar que se salgan de la norma, recuerda consultar siempre la documentación oficial de la tecnología con la que estás trabajando. Esto, forma también parte del día a día de la profesión.

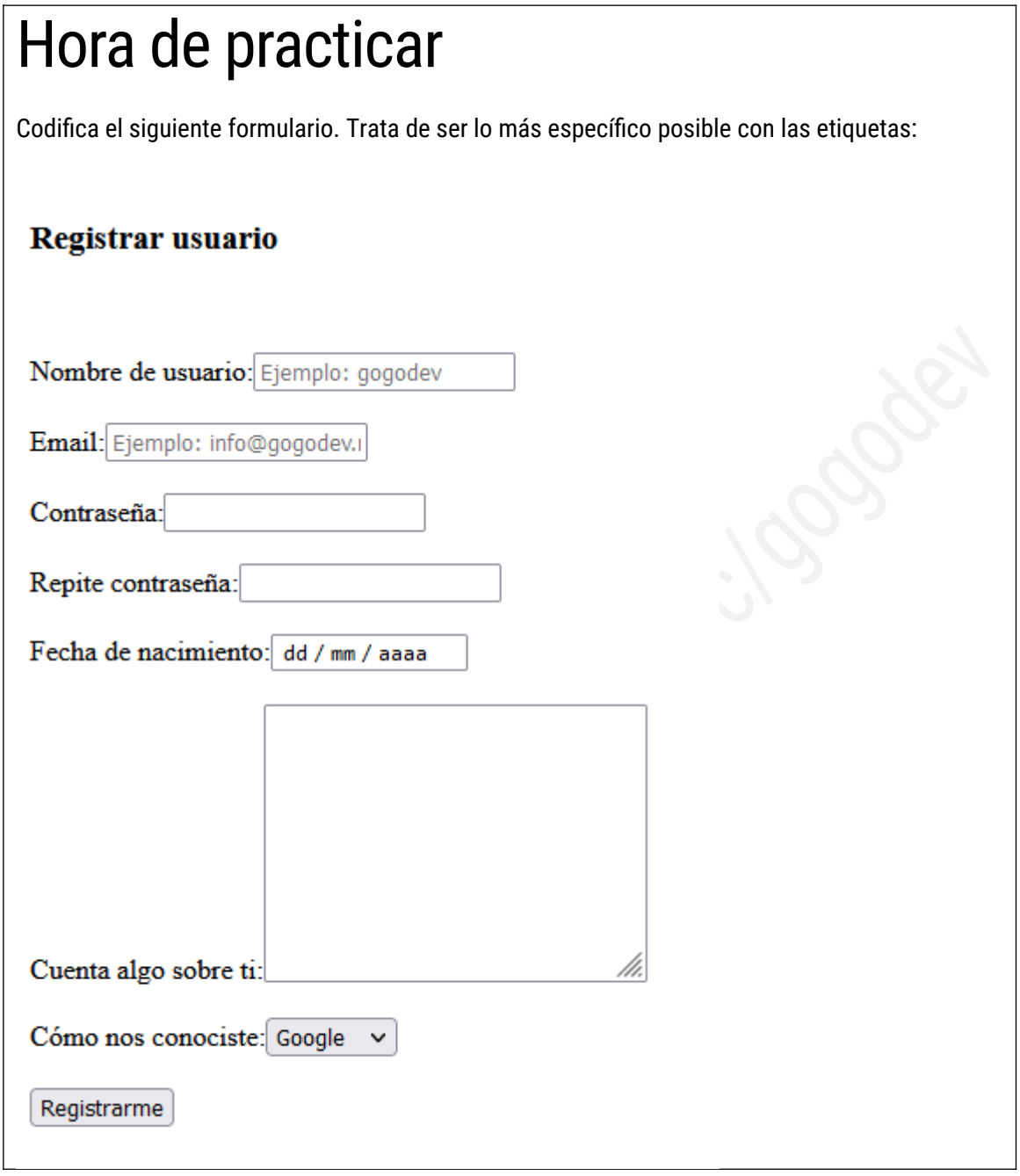
Para consultar etiquetas HTML, las páginas de documentación más usadas son:

- Mozilla: https://developer.mozilla.org/
- W3C: https://www.w3schools.com/

No obstante, no te preocupes por eso ahora. Debemos comenzar por el principio, por los cimientos, y ahora tienes un conjunto de herramientas suficientes para resolver casi cualquier necesidad en lo que a formularios se refiere. Es importante ponerlas en práctica y que comiences a familiarizarte con ellas.

Hora de practicar

Codifica el siguiente formulario. Trata de ser lo más específico posible con las etiquetas:



Si quieres repasar en vídeo todo este bloque, o te has quedado atascado, puedes ver todo el proceso en el canal de YouTube GOGODEV.

Acabar de adquirir la capacidad de establecer comunicaciones bidireccionales en internet, y si bien ha sido tan solo el primer paso, será la piedra angular sobre la que cimentarás tu conocimiento. Tiempo al tiempo.

</ENVÍOS>

Autor: Juan José Ruiz a.k.a GOGODEV
YouTube: https://youtube.com/c/gogodev
Twitter: https://twitter.com/JJRuizEmpresa (@JJRuizEmpresa)

IMPORTANTE

El contenido de este pdf lo tienes también en vídeo en el canal de YouTube https://youtube.com/c/gogodev , dentro de la lista de reproducción: Curso de HTML.

Todo el contenido formativo es gratuito, no olvides pasarte y suscribirte para no perderte nada, y así nos apoyas :)

<CAPÍTULO .05>

LA ESTRUCTURA DE LA SEMÁNTICA

SOBRE ETIQUETAS SEMÁNTICAS DE ESTRUCTURACIÓN DE CÓDIGO.

</CAPÍTULO .05>

LA ESTRUCTURA DE LA SEMÁNTICA

<SOBRE_ESTRUCTURAS>

A lo largo de los capítulos anteriores hemos aprendido a trabajar con numerosas etiquetas para trabajar el texto, los elementos multimedia o los formularios. Ahora ha llegado el momento, antes de comenzar con la estética, de poner orden en nuestras construcciones.

La estructura es un aspecto fundamental en el desarrollo web, no solo porque nos ayuda a mantener un código limpio y mejor estructurado, si no también por la importancia de la semántica.

Y es que las etiquetas de estructura nos ayudarán a agrupar parte de nuestro contenido, pero también serán de gran utilidad para los robots que quieran leer nuestra web y quieran saber qué contiene esta. ¿Y para qué quiero yo que un robot lea mi web? Te preguntarás. Sencillo. Hay múltiples de ellos que lo hacen a diario, y juzgan tu contenido, y a ellos las etiquetas de estructura les ayudan a entender qué contiene tu página.

Ejemplo de robot: Google. El robot de búsqueda de Google lee el contenido de tu página para saber qué contiene esta, y así poder mostrarla en su listado de opciones cuando alguien esté buscando alguna información de interés que contiene tu sitio. Y por supuesto, si deseas que tu sitio obtenga visibilidad (y en el caso de que trabajes como programador tu jefe o cliente seguro que lo querrá) debes ayudar al robot de Google con estructuras bien definidas.

Otro ejemplo de robot: El lector de contenido para personas discapacitadas. Muchas son las personas que navegan por internet y, sin embargo, padecen alguna discapacidad en la visión. Para poder navegar, usan robots como el lector del navegador, que les ayuda a entender qué se está mostrando en el monitor, y les lee o reproduce la información que solicitan.

Ya sea por estos, o el mero motivo de querer (y debes quererlo) ser un buen programador, que ejecuta un código limpio y bien estructurado, es importante agrupar las diferentes partes de nuestra página web en este tipo de etiquetas. ¡Vamos allá!

</SOBRE_ESTRUCTURAS>

<CABECERA_PRINCIPAL_Y_PIE_DE_PÁGINA>

Hasta ahora, la única estructura que hemos aplicado a nuestro código HTML ha sido la diferenciar el contenido de los metadatos dentro de la etiqueta ***<head>*** del contenido que mostramos, que hemos ido colocando dentro de la etiqueta **<body>**, y hasta ahora esto ha sido sencillo puesto que estamos obligados a hacerlo así. Sin embargo, dentro del body hemos ido aplicando todas y cada una de las etiquetas que hemos ido conociendo sin control.

Vamos a comenzar a darles un poco de orden, y vamos comenzar por las etiquetas de ***<header>*** ***<footer>*** y ***<main>***

Cuando estemos construyendo nuestra página web, dentro del ***body***, es un buen ejercicio pararse a pensar lo siguiente:

¿Cuáles de mis contenidos van a colocarse en la parte superior de mi página como elementos destacados, como puede ser el menú de navegación? Todos estos elementos los colocaremos dentro de la etiqueta ***<header>***, es decir, conformarán nuestra cabecera.

¿Cuáles de mis contenidos van a colocarse en el pie de página? Aquí, abajo del todo, se suelen colocar enlaces de refuerzo, los avisos legales, el *copyright* y otros elementos que no son el contenido principal de la página. Todo esto, que irá al final de cada página, lo agruparemos dentro de una etiqueta ***<footer>***.

Y todo lo demás, aquello que no pertenece ni a la cabecera ***<header>*** ni al pie de página ***<footer>*** lo ubicaremos dentro de la etiqueta ***<main>***, conteniendo la información principal.

Así pues, a partir de ahora, nuestras construcciones lucirán con la siguiente estructura básica:

```
<!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>
   <head>
      <!-- Aquí irá el contenido de la cabecera, como el menú de
          navegación o elementos introductorios como un banner general
          de mi sitio -->
   </head>
   <main>
      <!-- Aquí colocamos el contenido principal de nuestra página -->
   </main>
   <footer>
      <!-- Aquí colocamos el contenido que irá dentro del pie de página
           nuestro sitio, como enlaces de refuerzo o el aviso legal de
          nuestro sitio -->
   </footer>
</body>
</html>
```


Es importante destacar que, a la hora de estructurar un sitio web, no existe una única solución posible para cada problema, y podemos encontrar numerosas soluciones a una misma estructuración. Sin embargo, este es un buen punto de comienzo con el que seguro no te equivocarás.

</ CABECERA_PRINCIPAL_Y_PIE_DE_PÁGINA >

<NAVEGACIÓN>

Hablemos ahora de la navegación. En nuestro sitio web, deberemos colocar en diferentes lugares menús a través de los cuáles el usuario pueda navegar a través de las diferentes páginas que conforman nuestro sitio web. Pues bien, a nivel semántico, y sin importar dónde ubiquemos nuestro menú, este lo encerraremos siempre dentro de una etiqueta ***<nav>***

Además, y por convención de uso, siempre que desarrollemos un menú, lo haremos a través de una lista no ordenada, donde cada elemento de la lista contiene un enlace.

Veámoslo con un ejemplo:

```
<nav>
        <ul>
          <li><a href="index.html">Inicio</a></li>
          <li><a href="nosotros.html">Nosotros</a></li>
          <li><a href="servicios.html">Servicios</a></li>
          <li><a href="portfolio.html">Portfolio</a></li>
          <li><a href="contacto.html">Contacto</a></li>
        </ul>
</nav>
```

En este ejemplo, tenemos un menú de navegación, y por eso, lo hemos colocado dentro de una etiqueta ***<nav>***. Posteriormente, para desarrollar el menú, hemos usado una lista no ordenada ***<ul>***, donde cada elemento ***<li>***: inicio, nosotros, servicios, portfolio y contracto, contiene un enlace que te lleva a dicha página.

Como bien hemos dicho anteriormente, no existe una única forma de desarrollar un menú, pero si usas la fórmula estándar no vas a equivocarte.

Por último, indicar que los menús pueden ir dentro de cualquier lugar de nuestra página web. Puedes tener un menú dentro del ***<header>***, un menú de refuerzo en el ***<footer>*** y también, por ejemplo, en uno de los laterales del contenido principal. Ubícalos allá donde los necesites según la página web que estés construyendo.

</NAVEGACIÓN>

<SECCIONES_ARTÍCULOS_Y_SECUNDARIOS>

Gracias a las etiquetas vistas en los dos puntos anteriores, podemos segmentar mucho más nuestro contenido y darle algo más de orden. Sin embargo, y a priori, es presumible entender que el contenido que vayamos a ubicar dentro del ***<main>*** aún puede quedarnos algo extenso, y necesitamos darle orden. Para ello, vamos a conocer tres nuevas etiquetas de estructura que nos van a ayudar a continuar atomizando nuestro contenido: Las etiquetas de sección, las de artículo, y las de contenido secundario o relacionado: ***<section>***, ***<article>*** y ***<aside>***:

- **Etiqueta de sección *<section>*:** Con ella, agruparemos las diferentes secciones en las cuales se divide nuestro contenido principal. Imaginemos por ejemplo que tenemos una página que en su ***<main>*** va a contener información sobre la compañía, después va a tener una galería de imágenes de nuestros trabajos y por último una sección con diferentes testimonios de nuestros clientes. Podríamos agrupar cada una de ellas dentro de una etiqueta ***<section>*** para hacer entender que cada una de estas trata un tema diferente. Como puedes observar, a medida que vamos bajando en la estructuración, el uso de cada una de las etiquetas comienza a ser más subjetivo, pudiendo encontrar un mayor o menor número de ellas en función del criterio del constructor de la página. Y es que, visualmente, el uso de las etiquetas de estructura no tiene efecto. Este, es tan solo a nivel semántico.

- **Etiqueta de artículo *<article>*:** Esta tiene quizás un formato más sencillo de entender que la anterior. Y es que, dentro de ***<article>*** encerraremos, como su propio nombre indica, el contenido de un artículo dentro de una web. Si por ejemplo estamos construyendo un blog, encerraremos dentro de esta etiqueta todo el contenido del artículo, tales como su título, su imagen de cabecera, su descripción y/o su contenido. De igual forma, podríamos encerrar en un ***article*** cada uno de los productos en una tienda online.

- **Etiqueta de contenido secundario *<aside>***: Dentro de un ***aside*** colocaremos toda esa información que está relacionada con la sección o el artículo, pero cuyo material no forma parte de lo importante del mismo. Podemos entenderla como contenido auxiliar. Un ejemplo de ***aside*** sería, dentro de un blog, una barra lateral a la derecha en la que mostramos un listado de artículos relacionados con el post que se está leyendo actualmente, o un banner con un anuncio de interés para el contenido que se está visitando.

</SECCIONES_ARTÍCULOS_Y_SECUNDARIOS>

<DIVISIONES_EN_BLOQUE_Y_EN_LÍNEA>

Llegamos al final de las etiquetas semánticas con las divisiones *** *** y las divisiones en línea ******. Y es que, a pesar de todas las etiquetas vistas anteriormente, puede llegar a ser necesario en algún momento realizar otras divisiones, que no atienden a ninguno de los casos anteriores. Esto va a ser algo muy común a lo largo del desarrollo del siguiente curso (Curso CSS), cuando comencemos a trabajar la parte estética y visual de nuestro contenido y es que, a veces, puede resultar interesante realizar una nueva división, no porque sea una sección o un artículo diferente, si no simplemente porque queremos pintar de colores distintos cada una de ellas.

Por eso, y para realizar cualquier otro tipo de división en nuestra estructura que necesitemos, podemos acudir a las etiquetas de *** *** y ******.

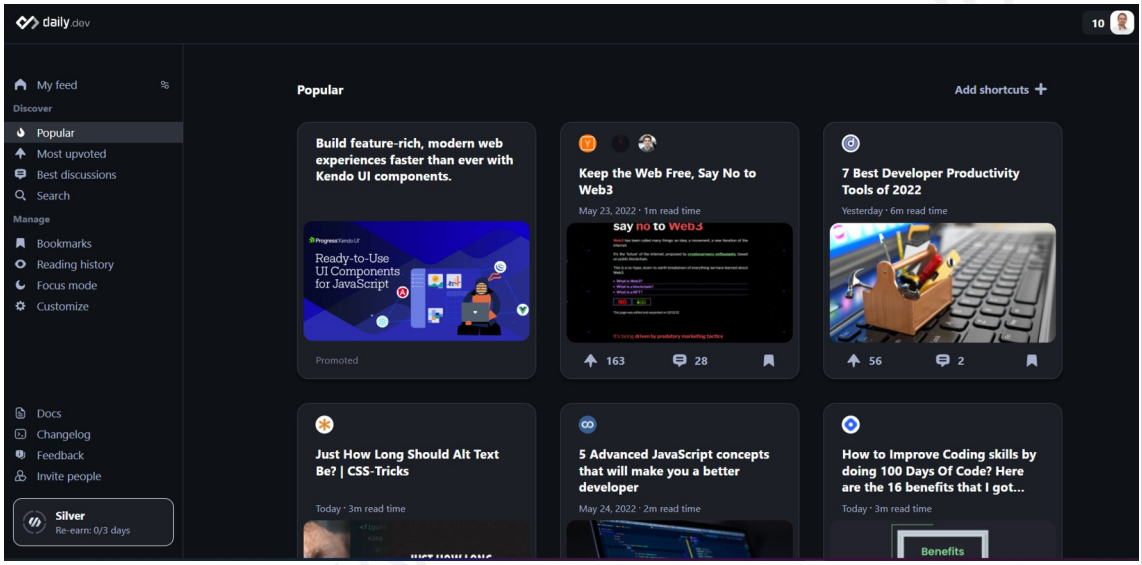
La principal diferencia entre ellas es que *** *** es una división de bloque, mientras que ****** es una división en línea. Estos, son conceptos que trataremos más adelante cuando nos adentremos en el uso de CSS, pero de momento puedes quedarte con esto:

- *** *** será interesante cuando hagamos divisiones en bloques completos, "que conformen más de una línea de contenido".

- ****** por el contrario será utilizado cuando queramos dividir un pequeño contenido dentro de una misma línea. Por ejemplo, para resaltar alguna frase en otro color dentro de un texto, como indicábamos en el ejemplo anterior.

Hora de practicar

¿Cómo estructurarías la siguiente página?



Y con esto no solo llegamos al final del capítulo, si no también al final del curso de HTML donde hemos estado tratando la estructura, el contenido. En el siguiente curso (Curso CSS) arrancamos una nueva senda para, ahora sí, poder comenzar a aplicar estética y diseño a nuestro contenido.

Si quieres repasar en vídeo todo este bloque, o te has quedado atascado, puedes ver todo el proceso en el canal de YouTube GOGODEV.

Ya tenemos el contenido. Es hora de vestirlo bien, hacerlo lucir, y presumir de ello.

</DIVISIONES_EN_BLOQUE_Y_EN_LÍNEA>