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

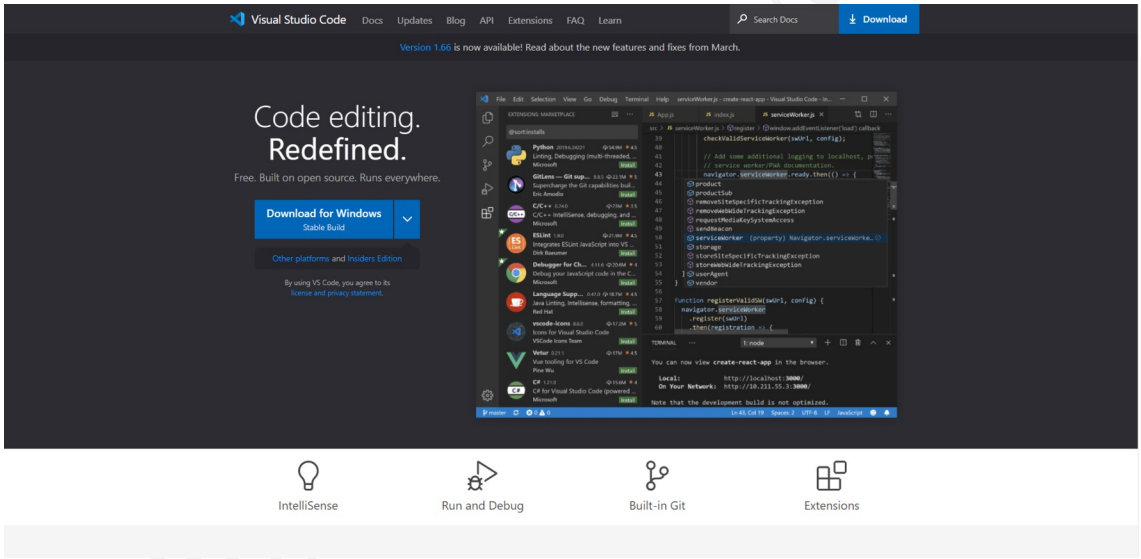
<CONFIGURANDO_EL_ENTORNO_DE_TRABAJO>

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.

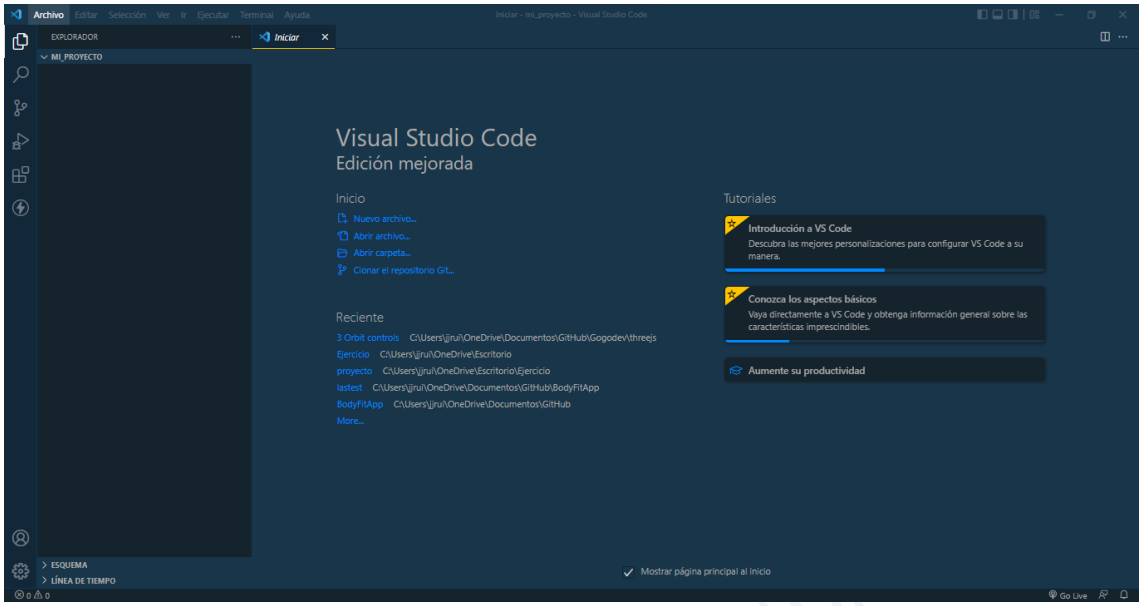
Si no entiendes la explicación de lo que hace cada elemento no te preocupes, no te dejes abrumar por las definiciones. A base de ir usando el IDE irás familiarizándote con cada uno de ellos. En este momento, y a esta altura de la formación, es suficiente con que te quedes con lo siguiente:

- 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.

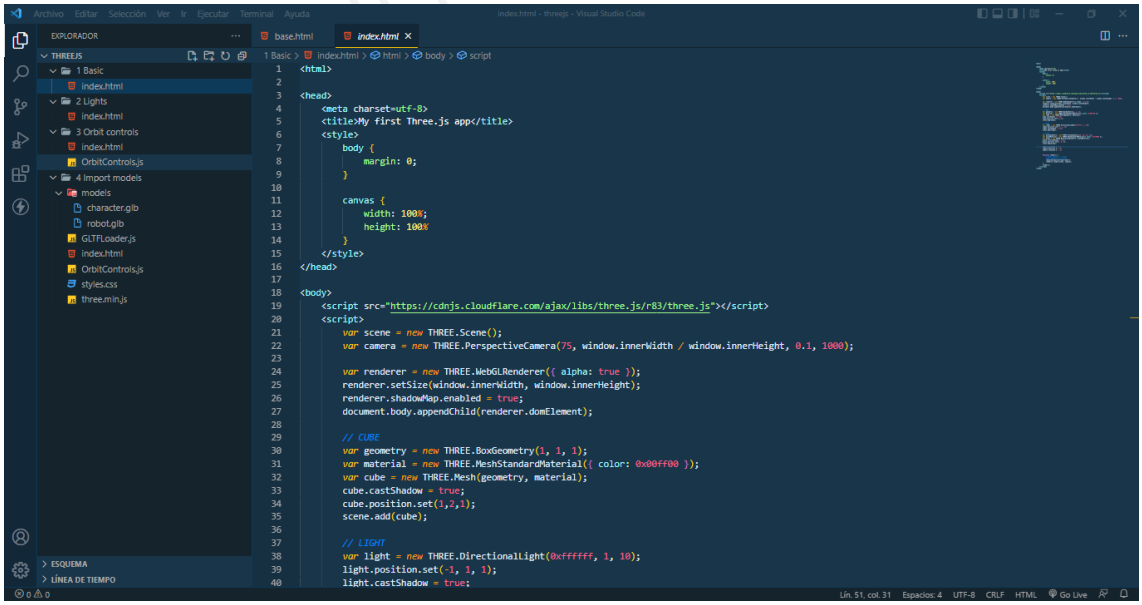
Al haber realizado esta acción, podrás comprobar que en tu Visual Studio Code te aparece una división:



- 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:

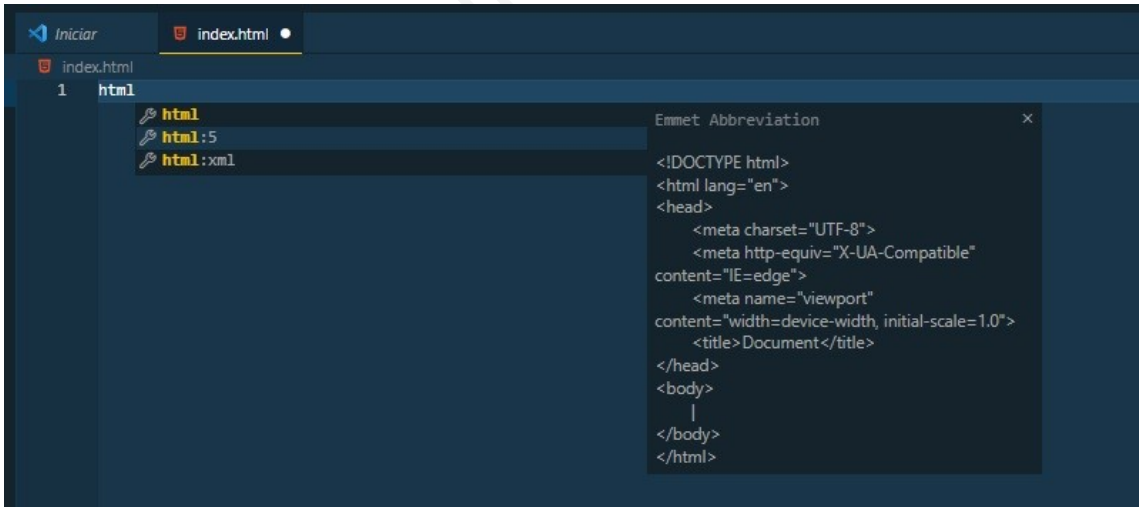
Snippets

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

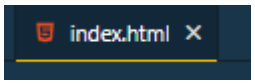
<title>Mi primera página web</title>

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:

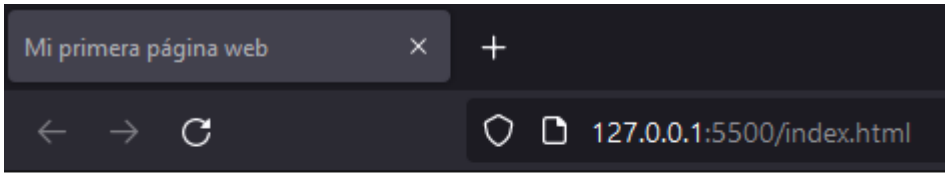
```
<!DOCTYPE html>
<html lang="es">
<head>
   <meta charset="UTF-8">
   <meta http-equiv="X-UA-Compatible" content="IE=edge">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Mi primera página web</title>
</head>
<body>
   <p>Hola Mundo</p>
</body>
</html>
```

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>