**Análisis y Desarrollo de Software (ADSO)**



**Presentado a**

David Eduardo Lozada

**Presentado por**

Jesús Adrián Molina Velasco

**Ficha**

**2502319**

**15 de febrero de 2023**

1. **Html 5**

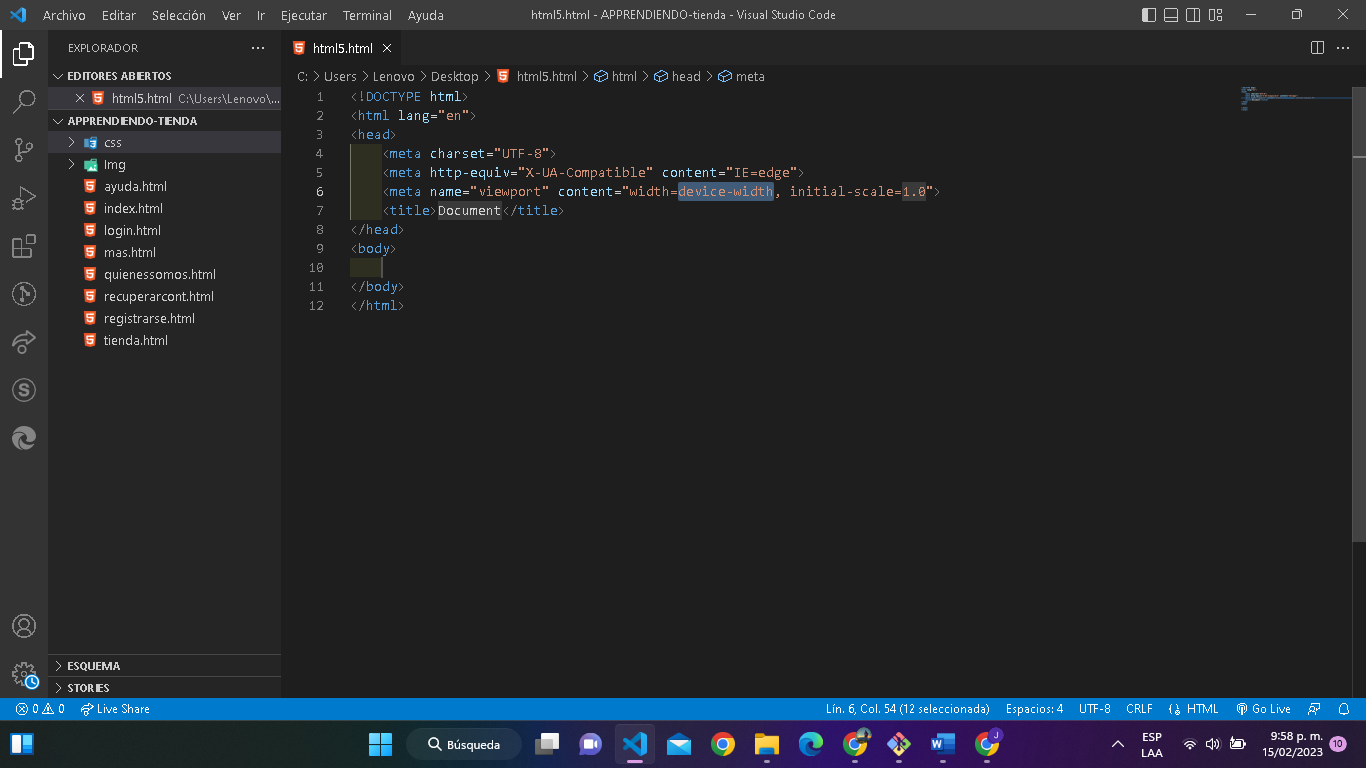
Hyper Text Markup Language, lo que significa Hyper Text, hipertexto que es un texto que enlaza otros contenidos, Markup que significa etiqueta o marca, pues todas las páginas web están construidas en base a etiquetaas y Languaje que significa lenguaje ya que HTML es un lenguaje, algo para resaltar es que HTMl es un lenguaje, pero no quiere decir que es un lenguaje de programación. Continuando con lo anterior, HTML5 es un estándar que sirve para definir la estructura y el contenido de las páginas web.

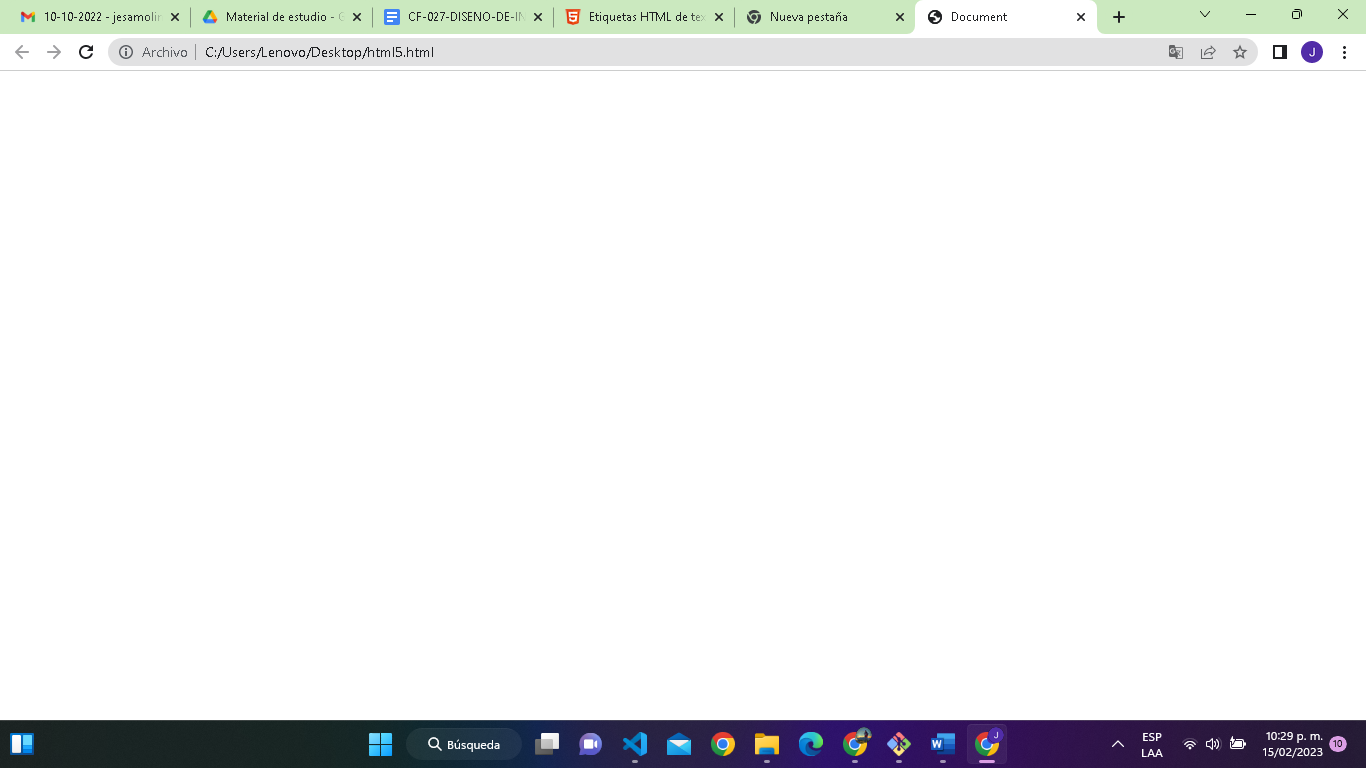
* 1. **Tags**

Los tags HTML corresponden a elementos de código HTML en una página web. Forman parte integral de la composición de la página, ya que permiten que se estructure y dé forma al contenido.

**Ejemplo:**

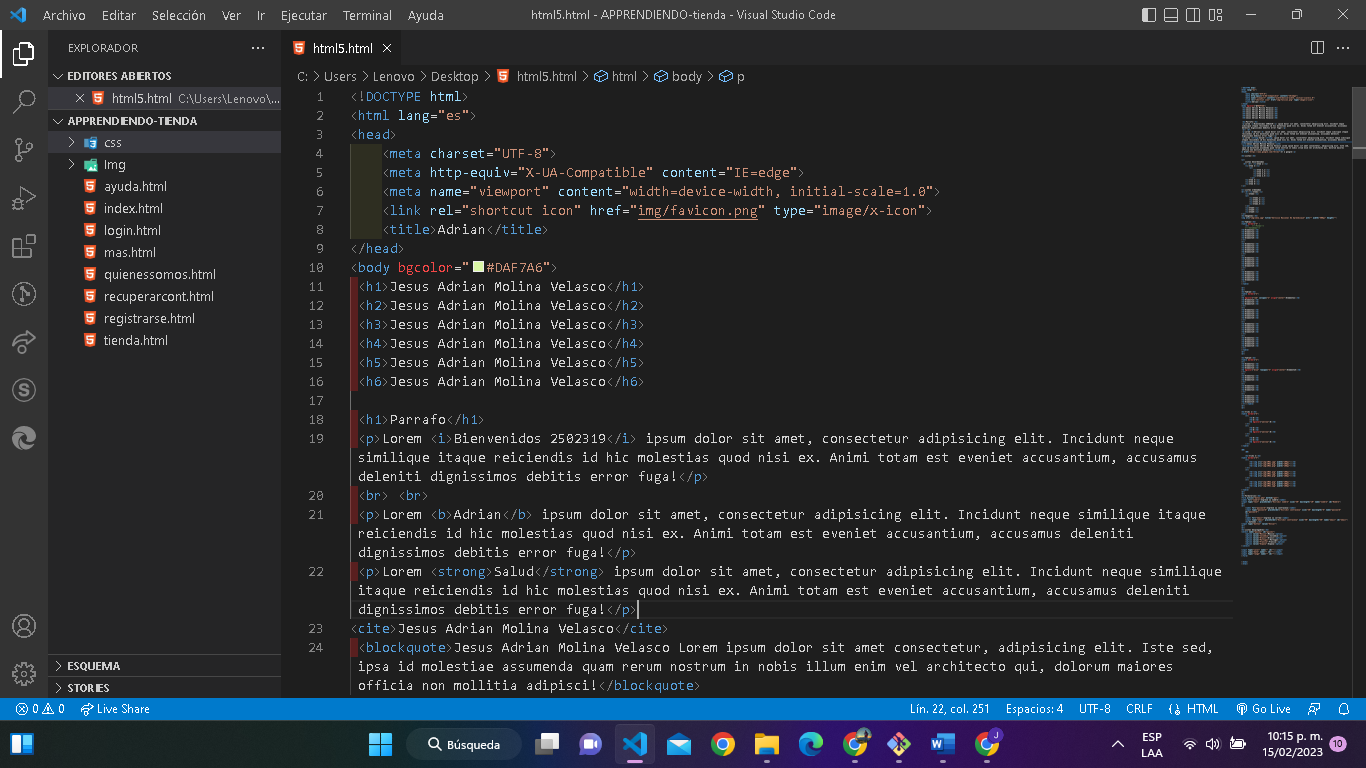
* <body> para el contenido
* <head> para información sobre el documento
* <div> división dentro del contenido
* <a> para enlaces
* <strong> para poner el texto en negrita
* <br> para saltos de línea
* <H1>…<H6> para títulos dentro del contenido
* <img> para añadir imágenes al documento
* <ol> para listas ordenadas, <ul> para listas desordenadas, <li> para elementos dentro de la lista
* <p> para parágrafos
* <span> para estilos de una parte del texto

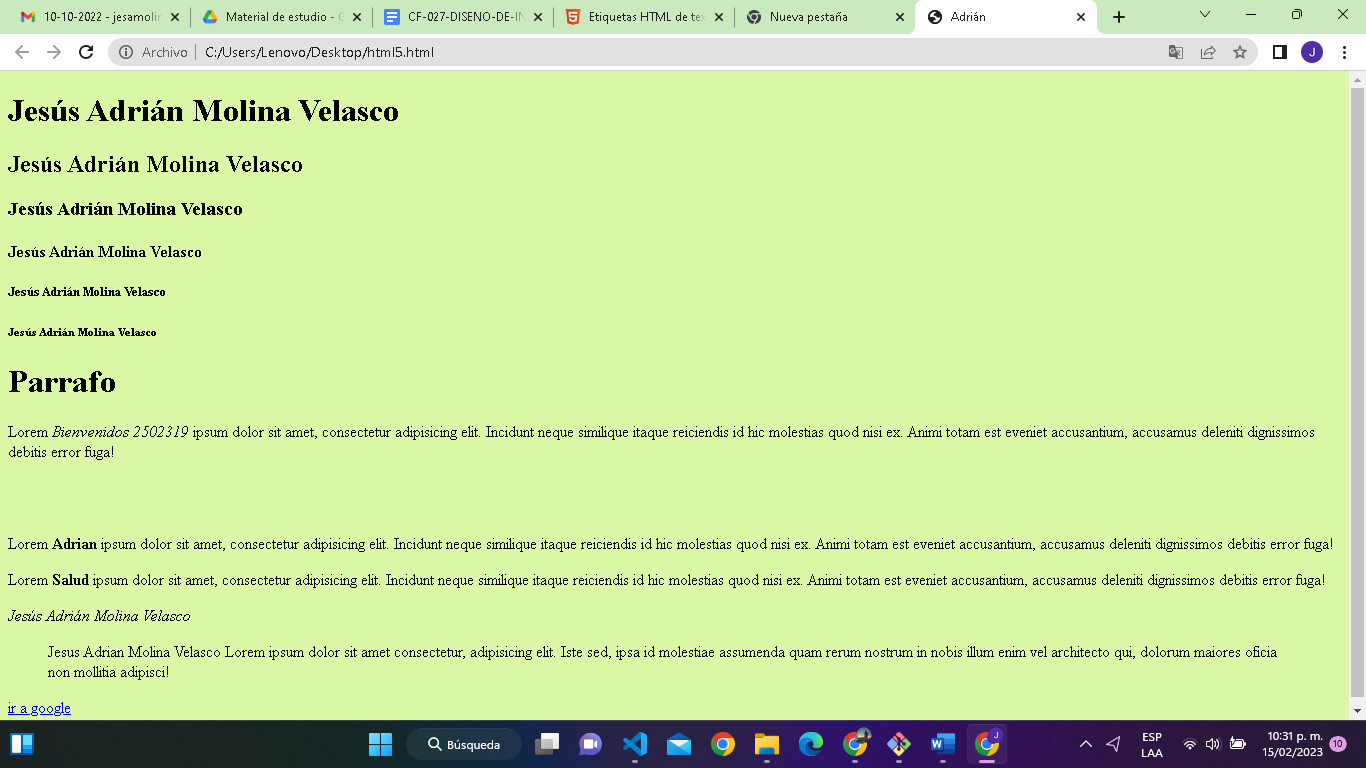




* 1. **Textos**

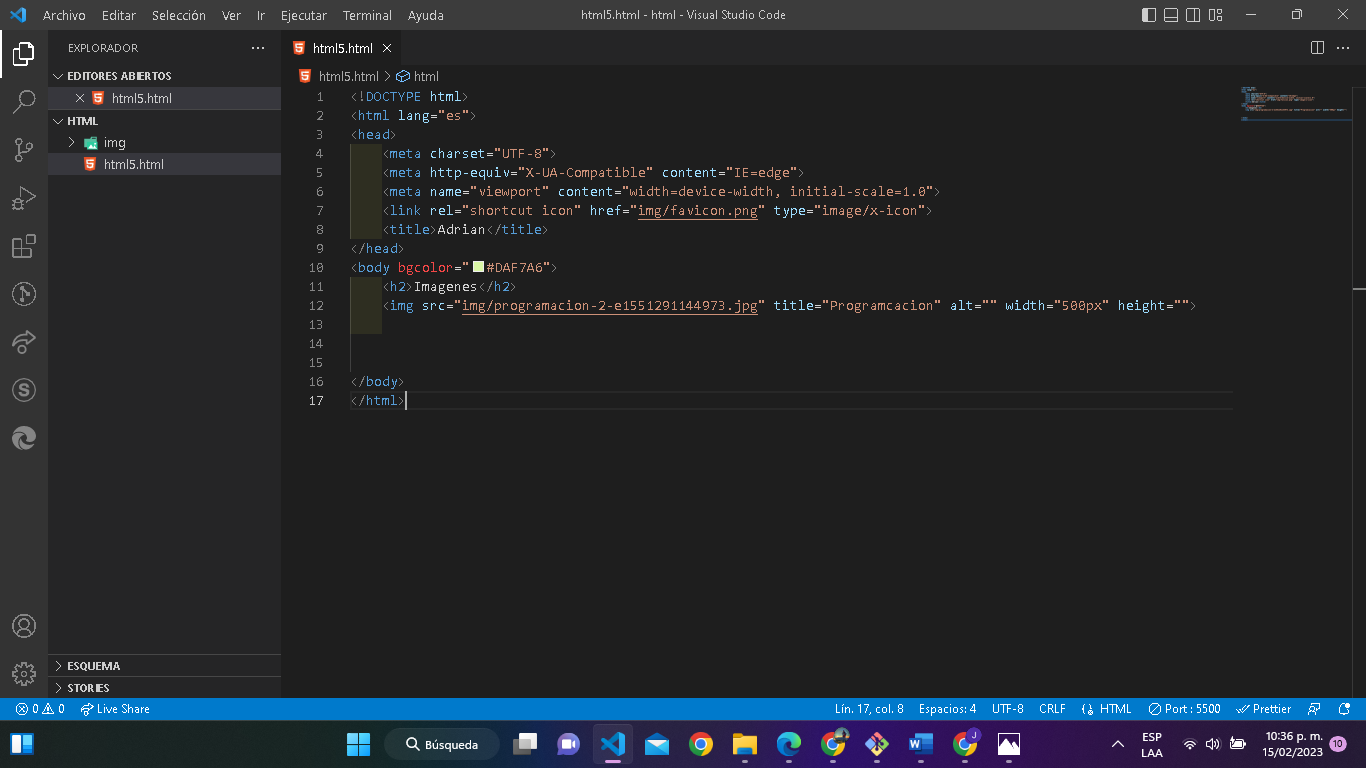
En otras palabras, el texto en una página web está «marcado» con estos códigos para dar instrucciones al navegador web sobre cómo mostrar el texto. Estas etiquetas de marcado son las propias etiquetas HTML. Cuando escribes código en HTML, estás escribiendo etiquetas HTML.





* 1. **Imágenes**

Para poner una imagen simple en una página web, utilizamos el elemento <img> . Se trata de un elemento vacío (lo que significa que no contiene texto o etiqueta de cierre) que requiere de por lo menos de un atributo para ser utilizado: src (a veces denominado por su nombre completo, source).



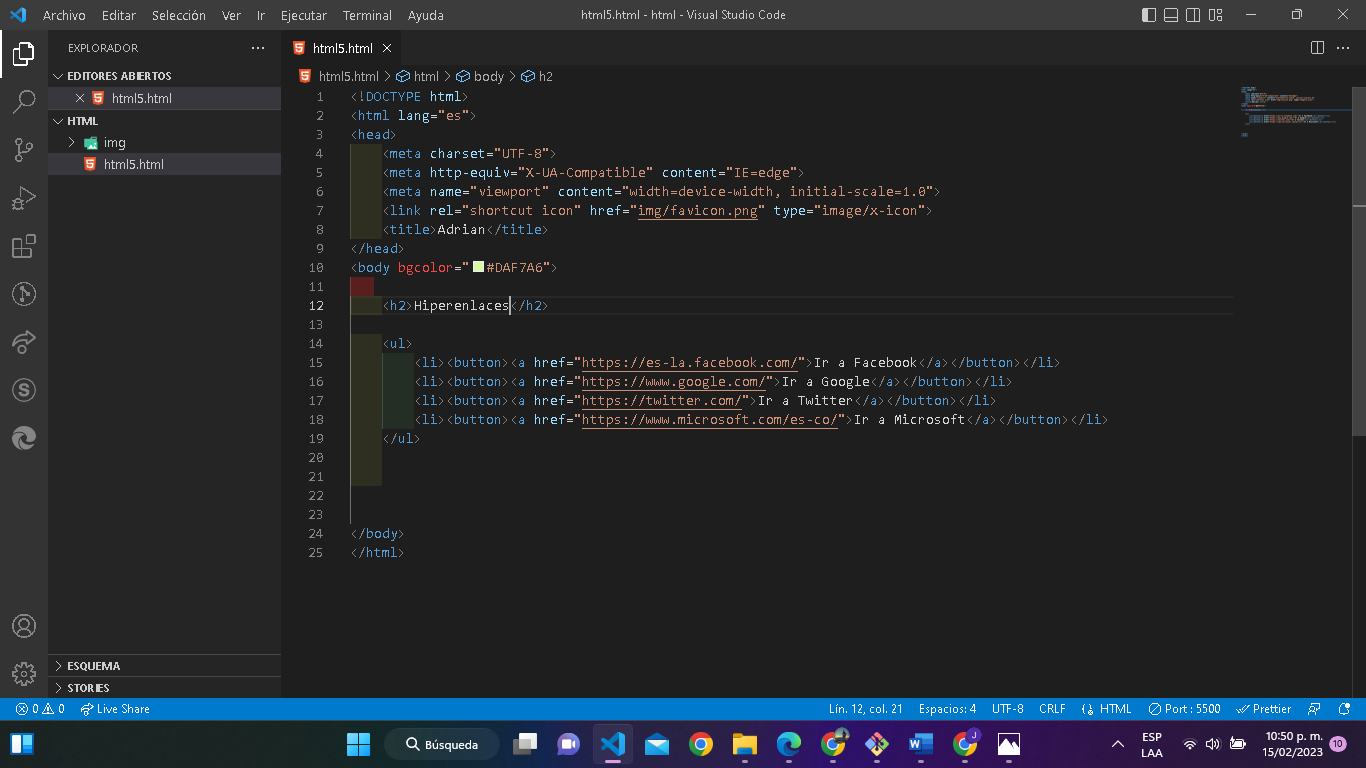
Captura de pantalla de computadora

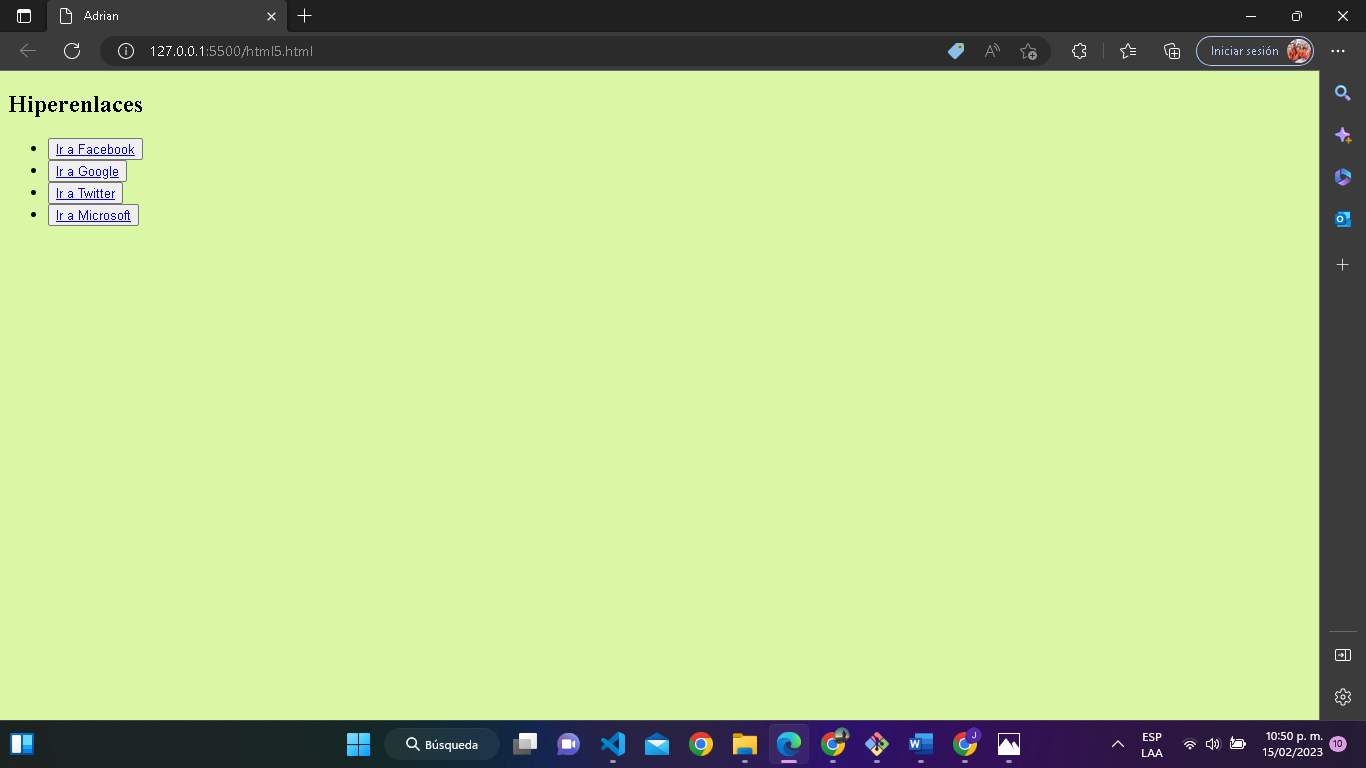
Descripción generada automáticamente

**1.4 Hiperenlaces**

La característica principal de una página Web es que podemos incluir Hiperenlaces. Un Hiperenlace es un elemento de la página que hace que el navegador acceda a otro recurso, otra página Web, un archivo, etc...

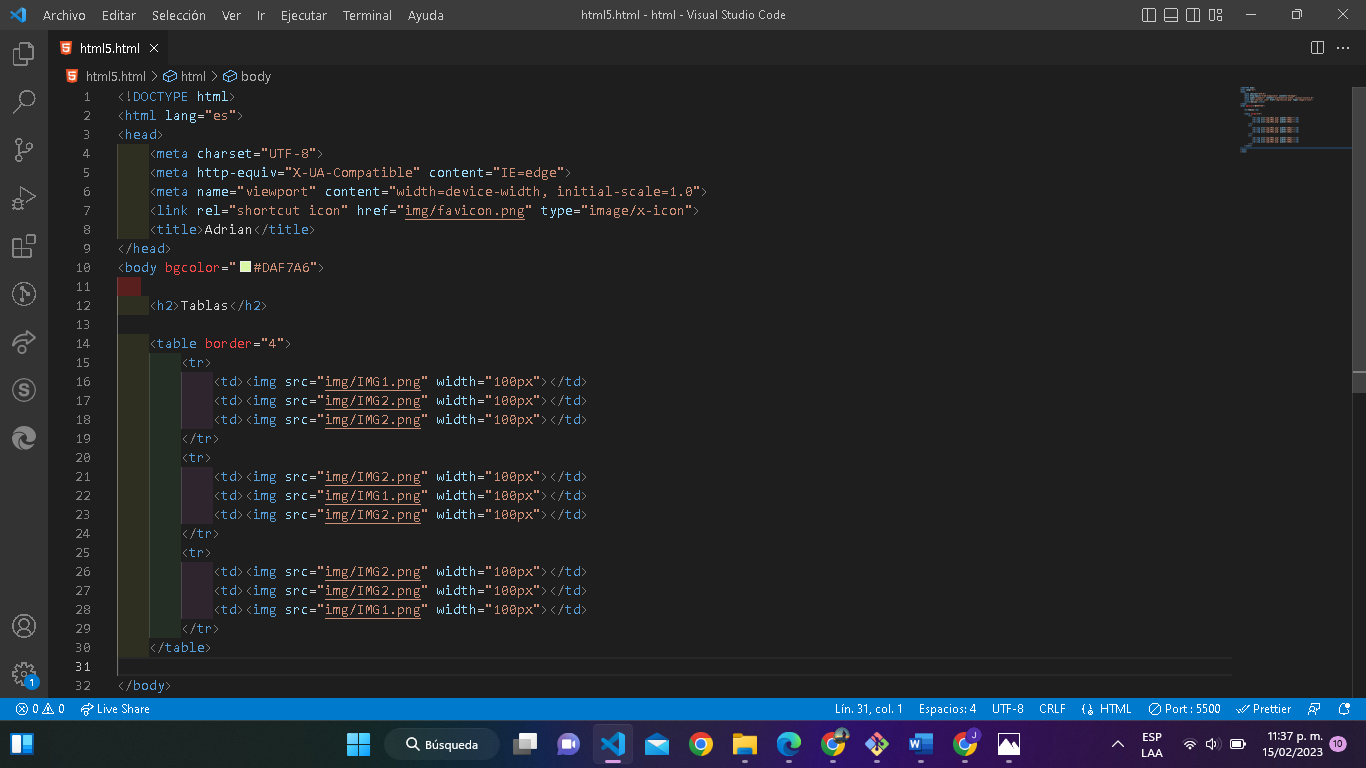
Para incluir un Hiperenlace se utiliza la directiva <A></A>. El texto o imagen que se encuentre dentro de los límites de esta directiva será sensible, esto quiere decir que si pulsamos con el raton sobre el, se realzará la función de hiperenlace indicada por la directiva <A></A>. Si el Hiperenlace esta indicado por un texto, este aparecerá subrayado y en distinto color, si se trata de una imagen, esta aparecerá con un borde rodeándola. Esta directiva tiene el parámetro href que indica el lugar a donde nos llevará el Hiperenlace si lo pulsamos.





**1.5 Tablas**

Las tablas HTML son elementos dentro del diseño web que permiten a los desarrolladores web organizar datos como texto, imágenes y enlaces en filas y columnas. Dentro del lenguaje de diseño web, las tablas HTML se crean usando las etiquetas <table> y </table>.

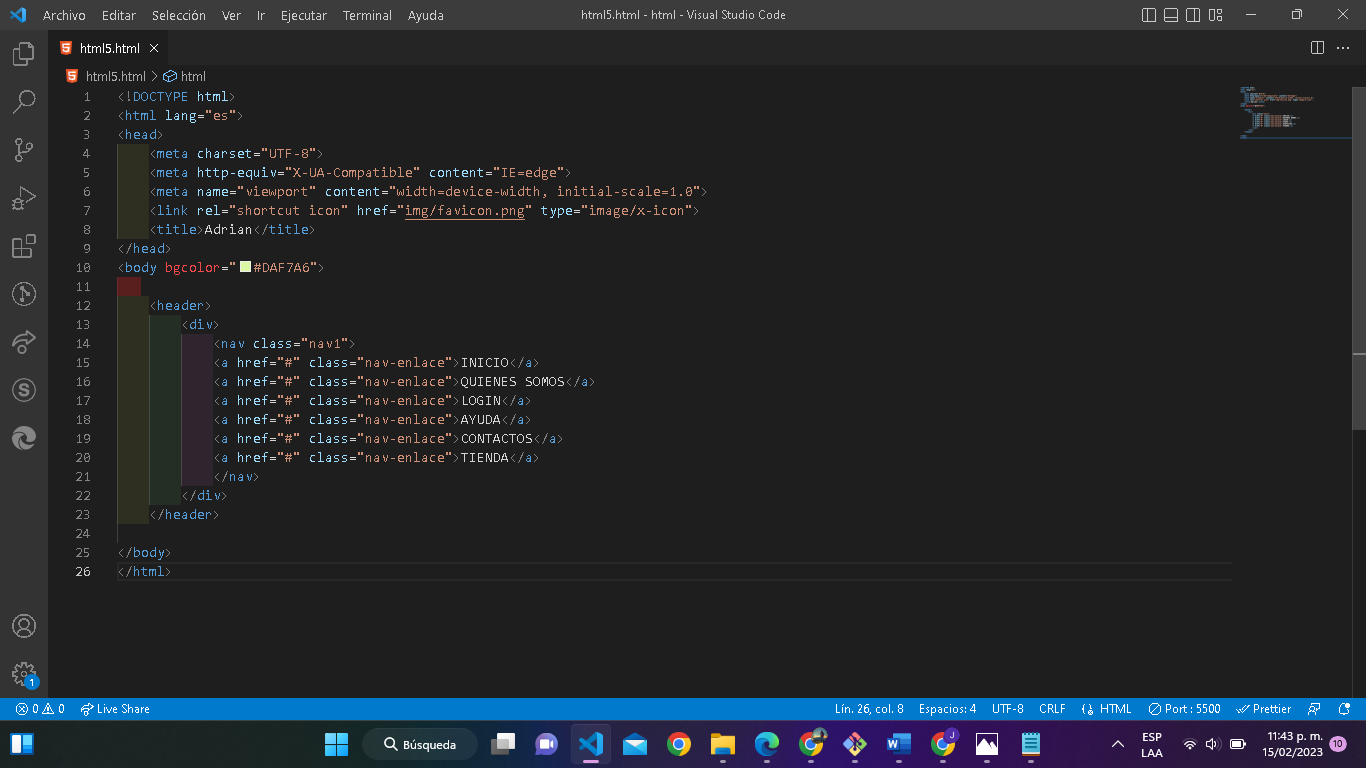


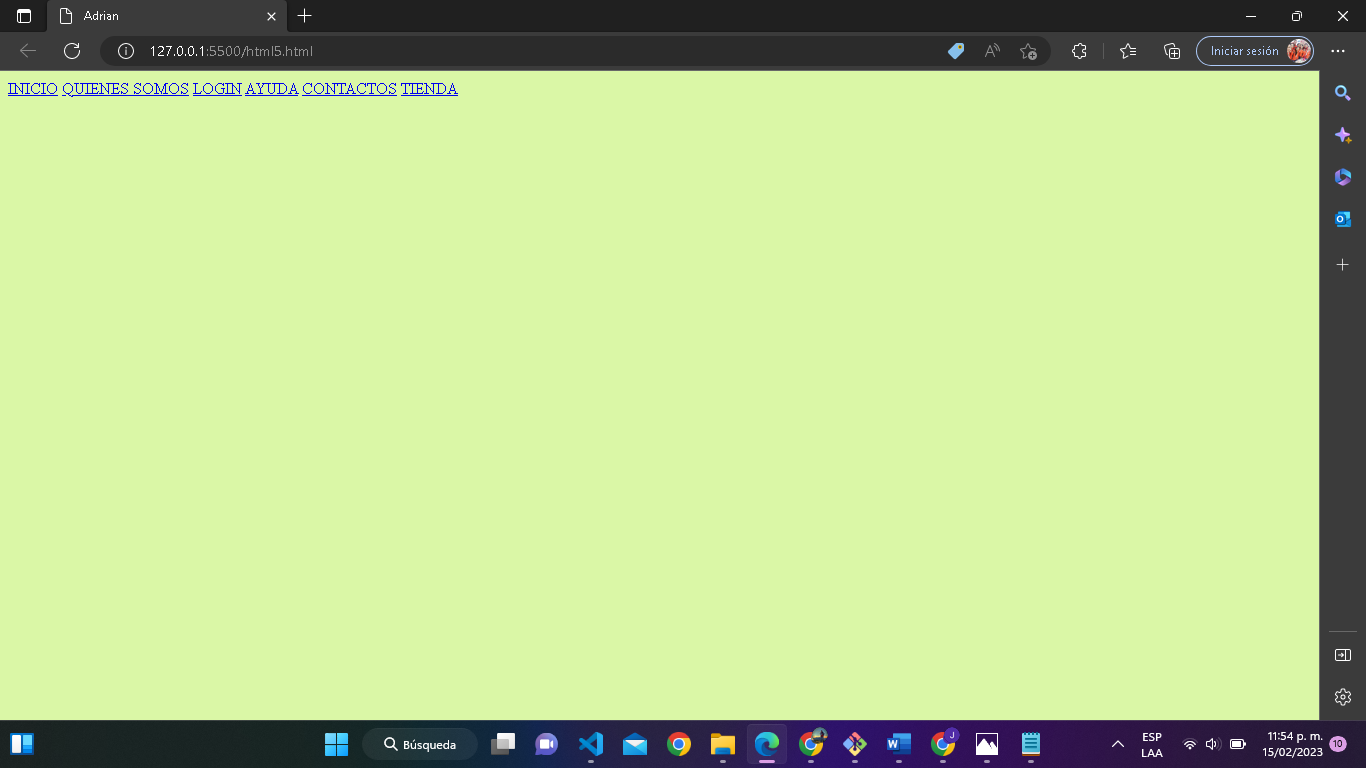
Captura de pantalla de un celular

Descripción generada automáticamente

**1.5.1 Divs**

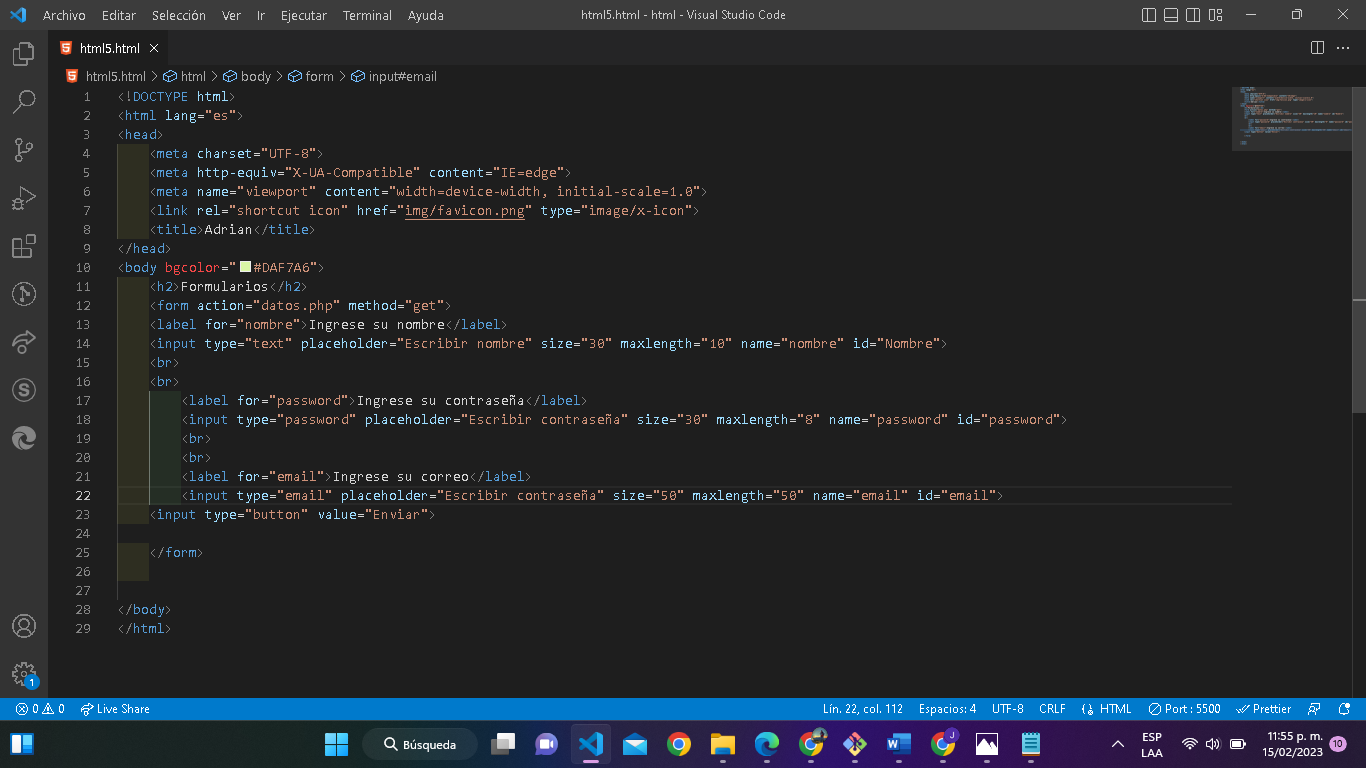
La etiqueta <div> se emplea para definir un bloque de contenido o sección de la página, para poder aplicarle diferentes estilos e incluso para realizar operaciones sobre ese bloque específico.





**1.6 Formularios**

Un formulario HTML es una caja de texto que puede contener casillas de verificación, botones y campos alfanuméricos. Suelen encontrarse en los sitios web para recolectar los datos personales de los usuarios interesados en la oferta del sitio. Estos datos posteriormente se envían a una base de datos para ser procesados.



Interfaz de usuario gráfica

Descripción generada automáticamente

1. **Hojas de Estilo**

Una **hoja de estilo** es un archivo de extensión \*.CSS (CSS, Cascading Style Sheets = Hojas de estilo) que contempla definiciones de formato (tipo de fuente, tamaño, color de la fuente, color de fondo, párrafos, etc) de las distintas etiquetas que forman una página \*.HTML. Su principal ventaja es definir un mismo aspecto para todas las páginas de un sitio web. Se crea una hoja de estilo y se vinculan todas las páginas del sitio web a este archivo. Cualquier cambio efectuado en la hoja de estilo afecta instantáneamente al formato de todas las páginas vinculadas a la misma.

**2.1 Conceptos**

Las hojas de estilo son una adición al HTML que aporta a los desarrolladores una manera más sofisticada de estructurar páginas web. Esto lo consigue separando el contenido de la página web (el texto), de la visualización (colores, estilos, posicionamiento, etc).

* Control más preciso que el existente sobre la presentación, fuentes, colores, fondos y otros efectos tipográficos.
* Un modo de mejorar la apariencia y formatear un gran número de páginas con tan sólo modificar un documento.
* Compatibilidad frente a los navegadores y las plataformas.
* Menos y más claro código, páginas más pequeñas y descargas más rápidas.

**2.2 Introducción CSS3**

CSS permite separar los contenidos de la página y la información sobre su aspecto. En el ejemplo anterior, dentro de la propia página HTML se crea una zona especial en la que se incluye toda la información relacionada con los estilos de la página. Utilizando CSS, se pueden establecer los mismos estilos con menos esfuerzo y sin ensuciar el código HTML de los contenidos con etiquetas <font>. Como se verá más adelante, la etiqueta <style> crea una zona especial donde se incluyen todas las reglas CSS que se aplican en la página.

**2.3 Estructura**

* **Regla:** cada uno de los estilos que componen una hoja de estilos CSS. Cada regla está compuesta de una parte de "selectores", un símbolo de "llave de apertura" ({), otra parte denominada "declaración" y por último, un símbolo de "llave de cierre" (}).
* **Selector:** indica el elemento o elementos HTML a los que se aplica la regla CSS.
* **Declaración:** especifica los estilos que se aplican a los elementos. Está compuesta por una o más propiedades CSS.
* **Propiedad:** característica que se modifica en el elemento seleccionado, como por ejemplo su tamaño de letra, su color de fondo, etc.
* **Valor:** establece el nuevo valor de la característica modificada en el elemento.

**2.4 Formularios CSS3**

1. **JavaScript**

JavaScript es un lenguaje de programación o de secuencias de comandos que te permite implementar funciones complejas en páginas web, cada vez que una página web hace algo más que sentarse allí y mostrar información estática para que la veas, muestra oportunas actualizaciones de contenido, mapas interactivos, animación de Gráficos 2D/3D, desplazamiento de máquinas reproductoras de vídeo, etc., puedes apostar que probablemente JavaScript está involucrado. Es la tercera capa del pastel de las tecnologías web estándar, dos de las cuales (HTML y CSS) hemos cubierto con mucho más detalle en otras partes del Área de aprendizaje.

Es un lenguaje de secuencias de comandos que te permite crear contenido de actualización dinámica, controlar multimedia, animar imágenes y prácticamente todo lo demás. (Está bien, no todo, pero es sorprendente lo que puedes lograr con unas pocas líneas de código JavaScript).

**3.1 Introducción**

Por ejemplo, si tuvieses tu propio archivo personalizado de JavaScript llamado ‘script.js’ y quisieras agregar su funcionalidad a tu página HTML, lo agregarías de la siguiente manera:

<!DOCTYPE html>

<html lang="en">

<head>

<title>Ejemplo de atributo de script Src</title>

</head>

<body>

<script src="./script.js"></script>

</body>

</html>

Esto apuntaría a un archivo llamado ‘script.js’ que está en el mismo directorio del archivo.html. También puedes enlazar otros directorios usando ’..’ en la ruta del archivo.

<script src="../public/js/script.js"></script>

Esto salta un nivel arriba del directorio, luego a un directorio 'público', luego a un directorio llamado ‘js’ y luego al archivo ‘script.js’.

También puedes usar el atributo ‘src’ para enlazar archivos ".js" externos alojados por un tercero. Esto se usa si no quieres descargar una copia local del archivo. Solo ten en cuenta que, si el enlace cambia o si el acceso de red está caído, el archivo externo que estás enlazando no funcionará.

Este ejemplo enlaza un archivo jQuery.

<script src="https://code.jquery.com/jquery-3.2.1.slim.min.js"></script>

**3.2 Versiones**

ECMAScript suele venir acompañado de un número que indica la versión o revisión de la que hablamos (algo similar a las versiones de un programa). En cada nueva versión de ECMAScript, se modifican detalles sobre Javascript y/o se añaden nuevas funcionalidades, manteniendo Javascript vivo y con novedades que lo hacen un lenguaje de programación moderno y cada vez mejor preparado para utilizar en el día a día.

**Ed. Fecha Nombre formal / informal Cambios significativos**

1 JUN/1997 ECMAScript 1997 (ES1) Primera edición

2 JUN/1998 ECMAScript 1998 (ES2) Cambios leves

3 DIC/1999 ECMAScript 1999 (ES3) RegExp, try/catch, etc...

4 AGO/2008 ECMAScript 2008 (ES4) Versión abandonada.

5 DIC/2009 ECMAScript 2009 (ES5) Strict mode, JSON, etc...

5.1 DIC/2011 ECMAScript 2011 (ES5.1) Cambios leves

**Ed. Fecha ECMAScript Cambios significativos**

6 JUN/2015 ES2015 Clases, módulos, hashmaps, sets, for of, proxies.

7 JUN/2016 ES2015 Array includes(), Exponenciación \*\*

8 JUN/2017 ES2015 Async/await

9 JUN/2018 ES2015 Rest/Spread operator, Promise.finally()...

10 JUN/2019 ES2015 flat, flatMap, trimStart(), optional error catch...

11 JUN/2020 ES2015 Dynamic imports, BigInt, Promise.allSettled

**3.3 Sintaxis**

**Espacios en blanco**

No se tienen en cuenta los espacios en blanco y las nuevas líneas: como sucede con XHTML, el intérprete de JavaScript ignora cualquier espacio en blanco sobrante, por lo que el código se puede ordenar de forma adecuada para entenderlo mejor (tabulando las líneas, añadiendo espacios, creando nuevas líneas, etc.) Sin embargo, en ocasiones estos espacios en blanco son totalmente necesarios, por ejemplo, para reparar nombres de variables o palabras reservadas. Por ejemplo:

var that = this;

Aquí el espacio en blanco entre var y that no puede ser eliminado, pero el resto sí.

**Comentarios**

JavaScript ofrece dos tipos de comentarios, de bloque gracias a los caracteres /\* \*/ y de línea comenzando con //. El formato /\* \*/de comentarios puede causar problemas en ciertas condiciones, como en las expresiones regulares, por lo que hay que tener cuidado al utilizarlo. Por ejemplo:

/\* var rm\_a = /a\*/.match(s);\*/

Provoca un error de sintaxis. Por lo tanto, suele ser recomendable utilizar únicamente los comentarios de línea, para evitar este tipo de problemas.

**Variables**

Las variables en JavaScript se crean mediante la palabra reservada var. De esta forma, podemos declarar variables de la siguiente manera:

var numero\_1 = 3; var numero\_2 = 1; var resultado = numero\_1 + numero\_2;

La palabra reservada var solamente se debe indicar al declarar por primera vez la variable. Cuando se utilizan las variables en el resto de las instrucciones del script, solamente es necesario indicar su nombre. En otras palabras, en el ejemplo anterior sería un error indicar lo siguiente:

var numero\_1 = 3; var numero\_2 = 1; var resultado = var numero\_1 + var numero\_2;

En JavaScript no es obligatorio inicializar las variables, ya que se pueden declarar por una parte y asignarles un valor posteriormente. Por tanto, el ejemplo anterior se puede rehacer de la siguiente manera:

var numero\_1; var numero\_2;

numero\_1 = 3; numero\_2 = 1;

var resultado = numero\_1 + numero\_2;

Una de las características más sorprendentes de JavaSript para los programadores habituados a otros lenguajes de programación es que tampoco es necesario declarar las variables. En otras palabras, se pueden utilizar variables que no se han definido anteriormente mediante la palabra reservada var. El ejemplo anterior también es correcto en JavaScript de la siguiente forma:

var numero\_1 = 3; var numero\_2 = 1; resultado = numero\_1 + numero\_2;

La variable resultado no está declarada, por lo que JavaScript crea una variable global (más adelante se verán las diferencias entre variables locales y globales) y le asigna el valor correspondiente. De la misma forma, también sería correcto el siguiente código:

numero\_1 = 3; numero\_2 = 1; resultado = numero\_1 + numero\_2;

En cualquier caso, se recomienda declarar todas las variables que se vayan a utilizar. Diagrama de declaración de variables

Nombres de variables

El nombre de una variable también se conoce como identificador y debe cumplir las siguientes normas:

Sólo puede estar formado por letras, números y los símbolos $ (dólar) y \_ (guión bajo).

El primer carácter no puede ser un número.

Por tanto, las siguientes variables tienen nombres correctos:

var $numero1; var \_$letra; var $$$otroNumero; var $\_a\_\_$4;

Sin embargo, las siguientes variables tienen identificadores incorrectos:

var 1numero; // Empieza por un número var numero;1\_123; // Contiene un carácter ";"

Diagrama de sintaxis de nombres

A continuación se indica el listado de palabras reservadas en JavaScript, y que no podremos utilizar para nombrar nuestras variables, parámetros, funciones, operadores o etiquetas:

Abstract boolean break byte case catch char class const continue debugger default delete do double else enum export extends false final finally float for function goto if implements import in instanceof int interface long native new null package private protected public return short static super switch synchronized this throw throws transient true try typeof var volatile void while with

Tipos de variables

JavaScript divide los distintos tipos de variables en dos grupos: tipos primitivos y tipos de referencia o clases.

Tipos primitivos

JavaScript define cinco tipos primitivos: undefined, null, boolean, number y string. Además de estos tipos, JavaScript define el operador typeof para averiguar el tipo de una variable.

**3.4 Tipo Datos**

El último estándar ECMAScript define nueve tipos:

Seis tipos de datos primitivos, controlados por el operador typeof Undefined: typeof instance === "undefined" Boolean: typeof instance === "boolean" Number: typeof instance === "number" String: typeof instance === "string" BigInt: typeof instance === "bigint" Symbol: typeof instance === "symbol" Null: typeof instance === "object". Tipo primitivo especial que tiene un uso adicional para su valor: si el objeto no se hereda, se muestra null; Object: typeof instance === "object". Tipo estructural especial que no es de datos pero para cualquier instancia de objeto construido que también se utiliza como estructuras de datos: new Object, new Array, new Map (en-US), new Set, new WeakMap, new WeakSet, new Date y casi todo lo hecho con la palabra clave new; Function: una estructura sin datos, aunque también responde al operador typeof: typeof instance === "function". Esta simplemente es una forma abreviada para funciones, aunque cada constructor de funciones se deriva del constructor Object.

**3.5 Estructuras de control**

Al hacer un programa necesitaremos establecer condiciones o decisiones, donde buscamos que el navegador realice una acción A si se cumple una condición o una acción B si no se cumple. Este es el primer tipo de estructuras de control que encontraremos. Para ello existen varias estructuras de control:

**Estructura de control Descripción**

If Condición simple: Si ocurre algo, haz lo siguiente...

If/else Condición con alternativa: Si ocurre algo, haz esto, sino, haz lo esto otro...

?: Operador ternario: Equivalente a If/else, método abreviado.

Switch Estructura para casos específicos: Similar a varios If/else anidados.

**Bibliografía**

<https://openwebinars.net/blog/que-es-html5/>

<https://www.atinternet.com/es/glosario/tag-html/> <https://www.iebschool.com/blog/que-es-etiqueta-html-analitica-usabilidad/#:~:text=En%20otras%20palabras%2C%20el%20texto,HTML%2C%20est%C3%A1s%20escribiendo%20etiquetas%20HTML>

<https://developer.mozilla.org/es/docs/Learn/HTML/Multimedia_and_embedding/Images_in_HTML>

<http://www.lcc.uma.es/~eat/services/html-js/manual8.html>

<https://www.juntadeandalucia.es/averroes/centros-tic/11000289/helvia/aula/archivos/repositorio/0/222/html/la_etiqueta_div.html#:~:text=La%20etiqueta%20se%20emplea,operaciones%20sobre%20ese%20bloque%20espec%C3%ADfico>.

<https://blog.hubspot.es/website/crear-formulario-html#:~:text=Un%20formulario%20HTML%20es%20una,de%20datos%20para%20ser%20procesados>.

<https://www.uv.es/fragar/html/html0701.html>

<http://www.ugr.es/~gallardo/cursoxhtml/2-ventajas.pdf>

<https://developer.mozilla.org/es/docs/Learn/JavaScript/First_steps/What_is_JavaScript>

<https://lenguajejs.com/javascript/introduccion/ecmascript/>

<https://www.arkaitzgarro.com/javascript/capitulo-3.html>

<https://lenguajejs.com/javascript/introduccion/estructuras-de-control/>