CURSO DE HTML Y CSS DESDE CERO – SOY DALTO

TEMARIO:

00:00 - Introducción

02:35 - Entendiendo la WEB

13:38 - Entendiendo HTML y CSS

26:10 - Editor de código

--- HTML BÁSICO ---

00:35:02 - Estructura de una etiqueta

00:45:48 - Estructura de una página WEB

00:51:58 - Parrafos y encabezados

01:00:09 - Listas

01:10:45 - Enlaces (Básico)

01:26:09 - Imagenes y Rutas

01:41:40 - Formularios

--- CSS BÁSICO ---

01:56:45 - Introducción a CSS

02:08:24 - Selectores (Básico)

02:18:04 - Propiedades de texto y fuente

02:44:56 - Tipografías externas

02:55:25 - Modelo de caja (box model)

03:07:04 - Relleno y Margen (margin y padding)

03:37:57 - Bordes

03:53:43 - Tamaño de caja (box sizing)

04:02:09 - Colores

04:25:16 - Unidades

04:50:29 - Fondos, Gradientes y Sombras

05:31:15 - Ejercicio práctico

--- HTML AVANZADO ---

05:38:02 - Metatags, Comentarios e Iconos

05:57:18 - Textarea y Labels

06:24:18 - Select, Datalist y Option

06:33:02 - Fieldset y Legend

06:44:35 - Details y Summary

06:51:34 - Enlaces (Avanzado)

07:04:12 - Tablas

07:22:25 - Audio y Video

07:34:10 - Lazy Loading

07:39:37 - HTML Obsoleto

07:49:10 - HTML Semántico

07:57:42 - Accesibilidad WEB

--- CSS INTERMEDIO ---

08:19:58 - Selectores (Avanzado)

08:36:14 - Herencia, Cascada y Especificidad

09:04:51 - Pseudoclases

09:30:38 - Pseudoelementos

09:37:51 - Metodología BEM

09:49:33 - Display

10:03:21 - Posición Relativa y Absoluta

10:26:36 - Ventanas Modal

10:35:20 - Posicion Fixed y Sticky

10:43:28 - Transiciones

10:56:53 - Desbordamiento (overflow)

11:01:49 - Control de flujo del texto

11:11:54 - Object fit y Object position

11:16:34 - Contorno (outline)

11:25:13 - Emmet (material re-usado)

--- CSS FLEXBOX ---

11:43:11 - Introducción a Flexbox

11:56:29 - Flex Direction, Flex Wrap y Flex Flow

12:17:33 - Alineación en los ejes

12:34:59 - Order

12:41:37 - Flex Básis, Shrink y Grow

12:59:10 - Align Self

13:02:42 - Layout con Flexbox

--- RESPONSIVE DESIGN ---

13:35:25 - Bloques y Multimedia flexible

13:56:07 - Atributos SRCSET y SIZES

14:10:17 - Picture, Source y Media

14:21:18 - Media Queries

14:29:57 - Ejercicio "Holy Grail" con Flexbox

14:48:36 - Mobile First

14:55:08 - Feature Queries

14:59:01 - Container Queries

--- CSS GRID ---

15:05:11 - Introducción a Grid

15:12:50 - Creando un Grid

15:24:54 - Unidades "auto" y "fr"

15:34:24 - Repeat y Minmax

15:42:18 - Grid implícito y explícito

15:53:34 - Grid dinámico (y responsive)

16:04:54 - Grid column y Grid row

16:14:52 - Grid flow: Dense

16:19:34 - Grid areas

16:35:06 - Alineacion con grid

16:53:28 - Subgrid

17:02:14 - Creando una página WEB

--- ANIMACIONES ---

18:02:13 - Transiciones (Repaso)

18:13:13 - Animaciones

18:29:19 - Botones Animados y Efecto Typewriter

18:52:42 - Animaciones basadas en Scroll

19:07:35 - Rango de animaciones

19:17:33 - Integrando animaciones en una WEB

19:29:14 - Animaciones, cumpliendo la promesa

--- WEB HOSTING ---

19:31:07 - Introducción al WEB HOSTING

19:57:59 - Subir tu WEB a INTERNET

---- SECCIÓN AVANZADA ----

20:09:16 - Filter y Backdrop Filter

20:29:30 - Transform

20:40:54 - Min, Max y Clamp

20:54:26 - Variables (custom properties)

21:05:50 - Función Calc

21:14:58 - Propiedades del Scroll

21:22:31 - Initial Letter

21:27:28 - Unidades del Viewport (Large, Small y Dynámic)

21:32:45 - Min-content, Max-content y Fit-content

21:39:19 - Función Color Mix

---- PROYECTOS FINALES ----

21:49:41 - Sidebar Menu & Chatbox (proyecto 1 y 2)

22:43:47 - Flip Card & One-Page View (proyecto 3 y 4)

23:21:42 - Menú Acordeón (proyecto 5)

---- COMO SEGUIR ---

23:42:39 - Como seguir aprendiendo

--- FIN ---

INTRODUCCIÓN

<!-- ENTENDIENDO LA WEB

Cuando trabajo con un archivo y lo ejecuto este se ejecuta sólo en nuestra pc

La www surge para lograr que la ejecución sea para cualquiera que la quiera ver

Este acceso se produce a través de un enlace que se puede compartir con otras personas

Todos los accesos se producen por una petición y una respuesta -

Los navegadores son los que convíerten el códio en una página web -->

<!— ENTENDIENDO HTML Y CSS

Html Lenguaje de Marcado de Hipertexto.

Lenguaje de marcado, presentan texto de forma diferente que necesitan un programa para que lo interprete. Es un lenguaje interpretado

html da estructura a la página

Bloque o algo que da más información al bloque

Cada uno es un elemento, cada osa que ponemos es un elemento

Los elementos se llaman tags o etiquetas

markdown

\*texto\* negrita

#texto encabezado grande

html

<h1>texto</h1> texto grande

Interprete, programa que va leyendo el código linea por línea y lo va interpretado en tiempo real

En html etiquetamos elementos

Propiedades, son sus características, estas son declaradas por el navegador, que trae valores predefinidos por el mismo, como no pueden quedar vacios pone valor por defecto

tamaño base 16, color negro

CSS - Hoja de Estilo en cascas

Le dice a todos los navegadores que propiedades standard tienen que tener

Establece un standard que todos los navegadores tienen que cumplir para decirle que propiedades quiero que tengan

CSS nos permite modificar las propiedades por defecto

CSS se define por reglas de estilo

identificador {propiedades: valor} se usa punto y coma para separar declaraciones, o propiedades-->

<!-- EDITOR DE CÓDIGO

Visual Studio Code

Instalar

Crear Acceso Directo en Escritorio

Tema dark mode

Abrir Folder Cursito HTML

Configuración

Hide las pestañas del navegador

Cerrar todos los menúes

Extensiones

Prettier

Indent Rainbow

image review - visualizar imagenes

Live Server

Tema - palenight theme dark One

-->

HTML BÁSICO

<!--

ESTRUCTURA DE UNA ETIQUETA

Sugerencia: tener activada la opción extensiones de archivo en el navegador de windows

en HTML cada elemento está definido por etiquetas

Se llama index porque cuando la mayoría de los servidores va a abrir una página web el archivo que va a llama es el index.html, es un índice y el html porque ese es el formato

Todo funciona por etiquetas, cada etiqueta

ETIQUETAS

<tag>todo esto forma parte de esa etiqueta</tag>

<tag>CONTENIDO</tag>

Todas las etiquetas tienen una función, hay muchas que no se recomienda utilizar, y otras que están obsoletas

No se da estilo en html porque es difícil de modificar luego con css

No es sólo para que se vea estética sino para que google la posicione más arriba

Escribir html con etiquetas semánticas que cumplen una una función específica

Puede haber etiquetas que son exactamente iguales pero para el buscador no, para SEO no, si le damos el significado correspondiente google los va a posicionar más arriba, para esto debemos saber usar las etiquetas

Las etiquetas ayudan al SEO, al posicionamiento en las búsquedad de Google

-->

<!--

ESTRUCTURA DE UNA PÁGINA WEB

Configurar el navegador predeterminado para Windows

En búsqueda escribimos navegador -> seleccionar aplicaciones predeterminadas -> navegador -> Chrome

Todas las páginas web tienen una estructura

html: 5 escribe un modelo básico de página web

-->

<!DOCTYPE html> <!-- determina el tipo de documento que estamos creando. Le dice al navegador que tipo de html estamos usando-->

<html> <!-- define que todo lo que está acá dentro va a ser parte de la página web.. Le habla al navegador diciéndole que todo lo que va acá va a ser parte de la página web -->

<!-- La estructura de una página web está formada básicamente por dos cosa

-->

<head><!-- Le da al navegador un montón de información que no podemos ver -->

<title>Aprendiendo HTML</title><!-- Define el título de la pestaña. El título siempre tiene que ser una descripción de la página -->

</head>

<body><!-- Acá va a estar la parte visual de la página. Dentro del body vamos a tener todo lo visible dee la web --></body>

</html>

<!--

PÁRRAFOS Y ENCABEZADOS

Para los párrafos se usa la etiqueta <p></p> que sirve para definir párrafos

usar esta etiqueta hace que haya un poco de espacio entre párrafo y párrafo, si no lo uso el texto queda peqgado

Cada párrafo tiene que tratarse de un tema, tiene que describir algo, tiene que abarcar la misma idea

ENCABEZADOS

Son van de <h1> a <h6> cuanto más grande el número más pequeña la letra

Por cuestiones de SEO sólo se usa un <h1> por página web, si se usan más google nos penaliza y ese título tiene que ser lo más descriptivo posible, es importante también por cuestión de accsibilidad

EJEMPLO

TÍTULO (h1)

SUBTÍTULO (h2) subtítulo(h2)

Punto 1 (h3)

Punto 2 (h3)

Punto 3 (h3)

<h1> lo usamos para poner el título principal de la página

<h2> lo usamos para las secciones

zh3> para subsecciones

-->

<!--

  LISTAS

  LISTAS ORDENADAS

  Se escribe <ol></ol>, dentro de estas etiquetas van los items que se ecriben <li></li>

  Por dejecto el navegador muestra los items de la lista como

  1.

  2.

  3.

  Podemos usar esto para crear una lista de pasos a seguir, por ejemplo, el orden sí impotat

  LISTA DESORDENADAS

  El orden no importa, por ejemplo una barra de navegación

  Se escibe <ul></ul> y para los ítems se usa <li></li> igual que en la lista ordenada

  En este caso no importa cómo se ve la págima, lo que mi importa es la estructura, que sea semanticamente correcto

  Estructura de Caja

  html

    head

    body

      li

      li

      li

  Esturctura de arbol

            html

      head         body

      title         ul

                 li li li

  Una pagina web es un archivo html

  Un sitio web son un onjunto de páginas web

-->

<!--

  ENLACES (Básico)

  También se pueden llamar hipervínculos.

  Se usa la etiqueta za></a> o anchor o ancla pra crear los enlaces

  Si pongo texto dentro de estas dos etiquetas lo único que voy a ver es ese texto, para que se un enlace y al hacer click me lleve a otro lado debo usar un atributo

  Los atributos son características de los elementos, similar a las propiedades en css, sólo que van directamente en html

  ATRIBUTOS

  href (Referencia de Enlace): Crea un link

  Los atributos que utiliamos en html son aquellos que agregan funcionalidad a la etiqueta, no aquellos que le dan estilo

  Si queremos agregar una dirección externa para nuestros enlaces tenemos que sí o sí poner https:// delante de la dirección, si no lo hacemos va a buscar una ubicación local, en nuestra computadora

  Si bien en html moderno no es necesario poner https, con poner las dos barras es suficiente, se recomienda ponerlo igual por distintas razones como seguridad, posicionamiento y otras

  Para crear un enlace o otra página en nuestro sitio, en el href escribimow el nombre del archivo al que queremos movernos

  Para que un enlace se abra en una nueva pestaña escribimos el atributo

  target="\_blank"

  target="\_self" abre la página en la misma pestaña, es la opción por defecto

  El atributo title agrega una leyenda al link o cualquier elemento html, se muestra como un recuadro con el texto. Normalmente se usa en los enlaces, en las imágenes, campos de formularios y algunos botones. Sirven para los lectores de pantalla

  Buena práctica: que lo elementos se reconozcan por lo que son, que se diferencien

  Enlaces dentro de la misma página: se usa href="#id del elemento"

  -->

<!--

  IMAGENES

  Hay dos forma de clasificar las etiquetas en lo que a código se refiere

  1. Las que necesitan un contenido, por lo que requieren una etiqueta de cierre que envuelva a su contenido

  2. La que no lo necesitan, porlo tanto no es necesario cerrarlas, ya que no hay un contenido que encerrar

    IMÁGENES EN LA MISMA CARPETA DEL ARCHVIO INDEX

  Para agregar una imagen se usa la etiqueta autocerrable image con el atributo source

  <img src="nombre.estensión de la imagen">

  El atributo alternative o alt sirve para los lectores dew pantalla y al usuaario para saber de qué se trata una imagen si por alguna razón esta no se cargó en la página, si la imagen carga este texto no aparece

  Sirve también para el SEO, ya que describe la imagen y permite buscarlas en el buscador de imágenes

  Atributo title muestra un texto al pasar el mouse por encima de la imagen

    IMAGEN DENTRO DE OTRA CARPETA

  Al programar trabajamos con muchísimos archvios que si los pusieramos todos juntos crearía problemas, debido a esto es que organizamos los archivos separados en carpetas

  En este caso creamos una carpeta images y dentro de ella colocamos las imágenes que vamos a necesitar

  -->

<!--

 RUTAS

 Existen dos tipos:

 ABSOLUTAS: es un recurso que está ahí, siempre, sin importar desde que dispositivo lo llame, siempre va a ir almismo lugar siempre. No importa donde tengamos la págian, en qué carpeta donde esté mi archivo

 RELATIVAS: sólo pueden acceder a archivos que estén en la misma carpeta que el index, si no lo tiene que hacer de otra manera dependiendo donde se encuente mi página

 No es buena práctica acceder con rutas absolutas a archivos locales, siempre se hace con rutas relativas,

 PARA REGRESA UNA CARPETA

 ../carpeta/archivo

 PARA CARPETA MÁS ADELANTE

 carpetaquequierocceder/archivo

 -->

<!--

    FORMULARIOS

    Todo es en base a petición-respuesta

    Forma que tenemos nosotros para pedir datos al usuario.

    Aunque todo es texto en formularios, las cosas se formatean los tipos de datos

    Para crear un formulario usamos la etiqueta de bloque

    <form></form>.

    Hasta ahora parece que no pasa nada, esto se debe a que nosotros somos quienes tenemos que definir los campos, los tipos de datos que queremos pedir

    <input> forma en que decimos dame un dato

    Atributos de input

    type: que tipo de dato o input queremos recibir

        - text: es el más comun de todos, permite tipear texto

        - color permite seleccionar un color

        - submit: crea un botón para enviar el formulario

    Value: indica el texto que aparece en

        - email: es igual a texto pero el formulario no se va a poder enviar hasta que no ese texto no tenga formato de email. texto@texto.extensión

        - password: oculta con asteriscos el texto del input. Para ver la contraseña, vamos a inspeccionar y cambiamos el type password por texto y se muestra la contraseña oculta

        - checkbox: permite marcar con un tilde una o varias opciones

        - file: permite seleccionar y enviar un archivo

    Cuando presionamos el  botón enviar el formulario va a recopilar todos los datos y los datos de los inputs y los va a enviar a un servidor para que los procese y haga algo, porque si estamos recogiendo los datos del usuario es para hacer algo. Con esto lo que hacemos es defini los datos del usuario para luego poder hacer algo

        - fecha: muestra un calendaro para seleccionar la fecha solicitada

    Hay cuatro atributos además del type de los input es muy importante saber\_

        - Required: lo que hace es obligar al usuario a escribir algo. No deja enviar el formulario si no sellenó el campo que tiene este atributo

        - name: es como un identificador de los inputs, definimos como se va a llamar este campo, cuando lo envío el servidor recibe el campo nombre, los names son únicos. Si queremos acceder a dato enviado por el usuario le tenemos que poner un nombre

        - placeholder: escribe un texto que desaparece cuando el usuario empieza a escribir, se usa para dar un ejemplo de lo que se epera en ese campo

        - value: es lo que va a recibir el servidor, si queremos recuperar un dato enviado por el usuario lo hace a través del value

        - min-length: no lo recomiendo, no es algo que usuaría siempre, pero limita el mínimo de caracteres a ingresar en el input

-->

CSS BÁSICO

/\*

INTRODUCCIÓN A CSS

CSS (Cascading Style Sheet): significa hoja de estilo en cascada, porque al dar estilo a los elementos vamos a seleccionar varios estilos, dependiendo el orden en que damos los estilos, el código se sobreescribe.

Se usa para darle un estilo a cada componente de un html, para que el conjunto de estilos que le damos a cada elemento nos de como resultado final una página estática, funcional, navegable, etc. Lo importante es entender como funciona cada propiedade y cómo funcionan sus combinaciones, a nosotros nos oimporta como se relacionan las cajas entre sí

CSS no existe sin HTML, HTML es la base sobre la que vamos a trabajar CSS

Las propiedades CSS contienen a los elementos HTML pero no son atributos como tal. Al tener una sintaxis diferente la forma de escribir CSS es muy distinta de la forma en que escribimos HTML

Hay tres formas de escribir CSS de manera que podamos afectar a los elementos

- Estilos en línea: se llama así porque normalmente se escribe en una sola línea de código

Se usa poniendo en la etiqueta HTML el atributo style y a continuación las propiedades CSS que queremos utilizar entre comillas y separadas por punto y coma

Ejemplo: <p style="color:red">

Al usar esta opción hay dos malas prácticas, porque tenemos que usar cada lenguaje por separado, no podemos mezclar código HTML, con CSS con JS. Cada lenguaje debe estar en una sección específica.

- Style como etiqueta: permite dar estilos

Se usa colocando la etiqueta style justo encima de la etiqueta de cierre de body y dentro se coloca la declaración que implica el nombre del elemento y entre llaves las propiedades CSS

Ejemplo:

p{

color: red

}

Si bien esta opción es mejor que la anterior, tampoco se recomienda utilizarla

Lo que se recomienda es tener todo en archivos separados porque vamos a tener mucho código en una página real, lo que complica el manejo del mismo tenerlo en una de las opciones anteriores

- Estilo en un archivo separado: impleca crear un archivo que va a contener todo el código CSS y para conectarlo con el HTML utilizamos la etiqueta link

Se usa colocando esta etiqueta en el head del archivo HTML

Atributos de link

rel: viene de relationship e indica que el archivo que se va a vincular

href: es la ruta a la hoja de estilo

Ejemplo:

<link rel="stylesheet" href="styles.css" />

Repasando

Lo primero que tenemos que hacer es elegir un selector, hay muchas formas de selecciónar elementos, la forma más básica es seleccionar un elemento utilizando su nombre

Lo primero que hacemos para cambiar las propiedades de un elemento es utilizar un selector, tenermos que seleccionar de alguna forma el elemento que queremos modificar, luego se habre un bloque de declaraciones, esto se hace con llaves, dentro de ellas se pone la propiedad a modificar, uego se coloca el valor de esa propiedad separado con dos puntos

\*/

/\*

SELECTORES (Básico)

Selectores: hacen referencia a la forma de seleccionar uno o varios elementos dentro de un documento de html en CSS para poder modificarlo

TRES FORMAS DE SELECCIONAR ELEMENTOS HTML

Selector Element (Selección por Elemento): busca el nombre del elemento html indicado para modificarlo.

Ejemplo HTML

<p>texto a estilizar</p>

Ejemplo CSS

p{

  color: blue

}

La peor práctica que hay es crear nuestros propios elementos, no se hace

Si queremos seleccionar un solo elemento de la lista, por ejemplo, podemos utilizar el elemento span, pero esto deja el puntito o viñeta en color negro y además no está tan bien hacer esto. Estaría bien si queremos resaltar una palabra o frase específica

Selector por Clase: permite asignar al elemento un nombre especial y separar los elemento por clases o tema, permite modificar varios elementos al mismo tiempo con las mismas características. Se pueden aplicar a varios elementos. No son elementos únicos.

Se usa poniendo un punto delante del nombre de la clase

Ejemplo HTML

<li class="faltante">item</li>

Ejemplo CSS

.faltante{

  color: red

}

Cada vez que agreguemos a un elemento el nombre de la clase se le va a aplicar la característica indicada para ella en CSS. De esta meanera podemos aplicarle a varios elementos la misma clase y las mismas propiedades. Está bien utilizar clases para muchos elementos

Hay estrategias para poner el nombre a una clase, si viene otro desarrollador tiene que entender por qué se usa una determinada clase, que significa esa clase

Selector por id: es un selector que se le puede dar a un sólo elemento en toda la página, no puede haber dos elementos con el mismo id. Son elementos únicos

Se usa colocando un hashtag o numeral delante del nombre de id

Ejemplo HTML

<li id="ingrediente-secreto">item</li>

Ejemplo CSS

#ingrediente-secreto{

  color: blue

}

En la práctica se puede usar más de un id en la misma página, pero es una mala práctica, no se recomienda hacerlo, y luego en javascript va a causar problemas. El id le dice al navegador este va a ser un identificador único para cada elemento

\*/

<!-- La etiqueta <span></span> se utiliza para seleccionar algo muy puntual -->

/\*

PROPIEDADES DE TEXTO Y FUENTE

Son propiedades que le vamos a dar al texto para que se modifique

El punto y coma separa las propiedades, la última no necesita llevarlo, si no lo pongo las propiedades no se aplican o no se aplican correctamente. Aunque no es necesario es una buena prácricta poner punto y coma luego de la última propiedad, ya que no sabemos si vamos a necesitar agregar más propiedades

color: determina el color del texto al que hace referencia, se podría poner font-color pero ya no se usa, por lo que dejamos sólo color

Colores relativos: es cuando usamos el nombre del color en inglés, pero el problema es que no todos los navegadores toman el mismo tono del color indicado y pueden verse distintos

Colores absolutos: son en los que se usa RGB; hexadecimal, etc., que van a ser el mismo en todos los navegadores. esto da un color fijo y absoluto

font-family: es la tipografía utilizada

Se usa comillas en la tipografía cuando su nombre tiene más de una palabra, si el nombre de la tipografía es de una sola palabra o las palabras están separadas con un guión se puede poner sin comillas.

Se pueden poner varias tipografía separadas por comas en caso de que la primera falle por alguna razón, el sistema va a buscar una por una de izquierda a derecha hasta encontrar una que funcione

font-size: define o cambia el tamaño de la fuente, por defecto es 16px

font-weight: es poner la tipografía más o menos gruesa, básicamente define el grosor de la tipografía.

Con <b> se pone la tipografía en negrita aunque ya no se usa en html, en su lugar lo podemos hacer con un span y modificarlo o se puede hacer con <strong> ya que google le va a dar más importancia a esta etiqueta en el seo y no a <b>

Los creadores de fuentes crear distitnos grosores para la misma tipografía, a más chiquito el número más fina la letra

- bold: equivale a 700 de grosor

- bolder: equivale al siguiente más grueso

- 100-900: rangos de grosor que va a depender de cada tipografía

font-style: es el estilo de la fuenta,

- normal: la letra es derecha

- ilalic: pone la letra en cursiva, la letra es inclinada (entre 8 y 12 grados aprox)

- oblique: menos inclinada que italic

- oblique valordeg: permite establecer que tan inclinada queremos la letra en grados, es una propiedad aún experimental

El estilo más usado es italic

Text-align: ajusta el texto a su contenedor, sólo funciona para elementos de bloque, no para los de línea

- left: valor por defecto, arranca a la izquiera

- right: arranca de la derecha

- center: la ubica en el centro del contenedor

- justified: aujuta el texto para que todas las líneas comiencen al inicio de la línea y termine al final, que se muestren ambos lados parejos

elemento de bloque: ocupal todo el espacio disponible

elemento de línea: el contenedor tiene el mismo tamaño que el elemento

start: toma en consideración de que lado se escribe normalmente, comienza donde comienza la escritura

end: toma también en consideración de que lado se escribe, y el texto termina donde termina la escritura

text-decoration:

- underline: subrayado inferior, por debajo del texto

- overline: subrayado superior, por encima del texto

- linethough: tacha el texto, lo cruza por el medio

- blink: el texto titila o parpadea, no se recomienda, no se usa

- none: es el más utilizado, no pone ninguno de los efectos anteriores

Se pueden mezclar las propiedades, puede tener más de una propiedad

line-height: define el espaciado de la línea, por defecto es 1, es el mismo tamaño de la letra

letter-spacing: es el espacio entre letra y letra, valorpx

text-transform

- none: valor por defecto, no hace nada

- appercase: texto en mayúsculas, solo para la vista

- lowercase: texto en minúscula

- capitalize: primer letra de cada palabra lo convierte en mayúscula

- full-width: da al texto el ancho completo

\*/

/\*

TIPOGRAFÍAS EXTERNAS

Son tipografías que no se encuentran en nuestro sitema.

Hay dos formas de integrar fuentes externas a nuestra página

1. Ingresar a google fonts, donde vamos a encontrar una gran cantidad de fuentes. Buscamos la que queremos

Por ejemplo: montserrat, esta fuente trae diferentes grosores o weights.

Seleccionamos la fuente que deseamos usar, hacemos click en el botón get font arriba a la derecha de la página, luego nos da dos opciones

embeded: nos da un link para llamar a la fuente desde el repositorio donde se encuentra

download: descarga las fuentes a nuestra computadora

Tener en cuenta que el sitio de google fonts cambia periódicamnete y con ello la forma de utilizar o descargar las tipografías (esto no lo dijo Dalto, el muestra otro método, de una versión anterior del sitio).

Si bien esta forma no está mal, recomienda hacer la segunda forma

2. Con un archivo css que guarde las tipografías: nos permite definir un nuevo valor para una nueva propiedad.

Para usar esta forma debemos primero descargar las tipografías a nuestro equipo, en la carpeta fonts, con los archivos de nuestra página, esta carpeta va a contener todas las tipografías descargadas

Se usa escribiendo font-face y crea la estructura.

Puedo ponerle el nombre que quiera, pero lo correcto es usar el nombre de la tipografía, en un archivo llamado fonts.css.

Ejemplo:

@font-face {

  font-family: "Montserrat";

  src: url("fonts/Montserrat-ExtraLight.ttf") format("truetype");

  font-weight: 100;

}

De esta manera creamos una variable que contiene la tipografía que queremos utilizar de las disponibles en la que elegimos en google fonts u otro sitio de fonts

Dentro de la variable debemos agregar el formato, ya que el navegador carga más rápido si sabe que tipo de archivo es e incluso navegadores que no soportan la tipografía la pueden mostrar también

font-family: es el nombre que le estoy dando a la fuente que estoy definiendo

src: url(); define la dirección o ruta donde se encuentra el archivo de la fuente a utilizar

format: especifica el tipo de archvio de fuente a usar. ttf = truetype, es un formato de fuente

font-weight: indica el grosor de la fuente asignada a esa variable. Esta propiedad funciona por cercanía, si no lo pongo el weight no me lo carga

De esta forma configuramos nuestra tipografía

\*/

/\*

BOX MODEL O MODELO DE CAJA

Es un concepto, no una propiedad, y este concepto lo que hace es describir como se diseñan y como se deberían poner todos elementos HTML en una página web

Un elemento en una página, se considera una caja rectangular, en HTML todo es es un bloque, como si fuera una cajita y este modelo lo que hace es describir como esas cajas se dimensionan, cambian su tamaño, cambian sus propiedades y también como interactúan entre sí

Hay cuatro propiedades principales de las cajas

- Contenido o content: el contenido es el núcleo del box model, básicamente es el área donde se muestra el texto, también pueden ser imágenes o cualquier otro contenido del elemento

- Padding o Relleno: todos le dicen padding, que es el espacio entre el contendio y el borde del elemento, esta propiedad aumenta el área alrededor del contendio pero debajo del borde, lo que hace es separar más el contendio del borde

- Borde o border: es una línea que rodea tanto al contendio como al padding, se puede controlar el estilo, el color y el grosor del borde

- Margen o margin: es el espacio alrededor de la caja, está fuera del borde, este espacio separa al elemento de otros elementos en la págian, a diferencia de todos los anteriores el margin no forma parte de la caja. Cuando el margen aumenta el contenido se achica

Estas propiedades pueden aplicarse hacia todos los lados o puede aplicarse a cada lado por separado

- Border Radius: redondea los bordes

- Box-sizing: define como se afectan todas las propiedades

\*/

/\*

WIDTH Y HEIGHT

Al comenzar no veo el color del div, porque no tenemos ni width ni height

Width y height: define el ancho y alto del elemento

Si el valor de la caja interna esta en porcenaje, este va a ser en función de la caja que lo contiene. El contenido se ajusta al tamaño de la caja que lo contiene

Lo que se hace para inputs en un celular podemos usar porcentajes

\*/ <!-- <div>Elemento de Bloque: que no tiene ninguna funcionalidad, es como crear una cajita vacía sin nada, solo es un divisor -->

/\*

PADDING Y MARGIN

Ahora nos toca hablar de dos propiedades que son muy importantes a la hora de trabajar con CSS. Son dos propiedades muy parecidas sólo que afectan a cosas distintas de la caja. El contendio simpre va a medir lo mismo, no cambia, lo que cambia es el relleno. Es necesario para muchas cosas

PADDING

Es básicamente la distancia que hay entre el contenido y el borde de una caja. Lo que hace es aumentar el tamaño del espacio alrededor del contenido pero dentro de los bordes de la caja.

El padding básicamente está cambiando el tamaño de la caja

Shorhand: es una manera de abreviar propiedades, es cuando resumimos en una propiedad varias juntas. por ejemplo: padding reune:

padding-top: separa el contenido del borde superior

padding-bottom: separa el contendio del borde inferior

padding-left: separa el contendio del borde izquierdo

padding-right: separa el contenido del borde derecho

Ejemplo: en lugar de poner el mismo valor a cada una de las propiedades anteriores, usams

padding: 20px

esto aplica 20px de paddigan a cada uno de los lados del contenido

El orden en que se aplica el padding con el shorthand es siguiendo las agujas del reloj: top right bottom left

Ejemplo

padding: 10px 0 20px 40px

Cuando usamos 0 no necesitamos poner la unidad que usamos, ya que en todos los casos daría 0

También puedo poner valores a las propiedades de a pares: (top bottom) (right left)

Ejemplo padding: 10px 20px

padding cambia el tamaño de la caja pero no el tamaño del contenido

Un valor: igual para todos los lados

Dos valores: vertical y horizontal, arriba y abajo un valor, a izquierza y derecha otro valor

Cuatro valores: un valor distinto a cada lado del contenido

Afecta como se ve la caja pero no cambia el tamaño del contenido

background y background-color es lo mismo

MARGIN

Es una propiedad que permite separar los objetos. Si agrego margin, lo que hago es que las cajas se separen entre sí.

Ejemplo

margin: 10px

Agrega un margen de 10px a cada lado de la caja

margin colapse: deberíamos tener el mismo margen a cada lado de la caja, pero arriba y abajo los margenes se superponen el margen más grande es el que se queda cuando dos cajas tienen su propio margin. Los márgenes siempre colapsan, no se suman, sino que se superponen. La separación total no va a ser la suma del margen de arriba y el margen de abajo de dos cajas, sino el margen más grande

El body tiene un margin por defeto, para sacarlo debemos poner para body margin: 0

Elementos hermanos son los que estan consecutivo, es decir en el mismo rango jerárquico, están adentro de la misma caja contenedora o padre. En este caso se aplica el margin colapse. Esto se puede evitar utilizando flexbox, pero a su vez trae otros problemas porque se verían afectados por otras propiedades de flexbox

Antes si usabas un border o un padding o un overflow, no colapsaban, pero ahora esto no funciona y colapsan igual

El contenido, el padding y el borde forman parte de la caja, el margen no. Por lo que los tres primeros cambian el tamaño de la caja, pero el margen no. sólo cambia el espacio que ocupa la caja

\*/

<!-- El form es un elemento en bloque, porlo que ocupa el ancho de su contenedor

Si quiero tener dos clases en un mismo elemento las pongo dentro de las mismas comilla separada por un espacio

Ejemplo: class="formulario-input formulario-submit" -->

/\*

MINI EJERCICIO - TARJETA PRESENTACIÓN 1

margin: auto permite centrar todo el contenido en su contenedor

\*/

/\*

BORDES

Esta propiedad nos permite modificar tres aspectos fundamentales de los bordes

1. Tamaño: todos los bordes tienen un tamaño un grosor

2. Estilo: tipo o diseño del borde del elemento

3. Color: color con que se mostrará el borde del objeto

4. Redondeo: permite redondear las esquinas de un elemento

border-width: permite definir el grosor del borde

border-style: permite definir el estilo del borde liso, punteado, doble, línea, etc

- hidden: el borde se oculta y deja de afectar al flujo del elemento, se oculta hasta que lo llamemos, es como que no existe

border-color: permite definir el color del borde

- transparent: el borde no se ve pero existe, no afecta a lo visual del elemento pero está ahí

Para que el borde se vea debemos haber definidos las tres propiedades anteriores, de lo contrario no se va a mostrar

border-radius: permite redondear las esquinas o vertices del borde, especifica cuanto quiero que se redondeen las esquinas

Si usamos pixeles se redondea sólo las esquinas, tomando solo el centro vertical, si usamos porcentaje se redondea desde el medio horizonal y vertical.

border: es un shorthand que reemplaza a las tres primeras propiedades

Ejemplo

border: tamaño estilo color

border: 3px solid black

Si queremos modificar sólo una de las propiedades puedo agregarla aparte

Si queremos dar estilo sólo a uno de los lados del borde usamos una de las siguientes propiedades

border-top:

border-bottom:

border-left:

border-right

Se pueden usar con width, style y color como border o simplificarlo para cada lado igual que con border, actuando como shorthands

border-radius: tambien se puede aplicar a cada esquina por separado, también se puede utilizar como shorthand

border-radius: arriba a la izquierda arriba al a derecha abajo a la derecha abajo a la izquierda, sigue las agujas del reloj

Hay que entender que muchas de las cosas que vamos a ver en CSS existen pero no se usan

border-style: se usa solid, los otros estilos no se usan

No se hace de poner dos selectores iguales en el mismo archivo, en este caso se hizo para separar el tema de la clase

\*/