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 ---

Links de Referencia

<https://code.visualstudio.com/api/references/theme-color>

SECCIÓN 1 - INTRODUCCIÓN

TEMA 1

<!—

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 -->

TEMA 2

<!—

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-->

TEMA 3

<!—

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

-->

SECCIÓN 2 - HTML BÁSICO

TEMA 4

<!—

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

-->

TEMA 5

<!--

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>

TEMA 6

<!--

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

-->

TEMA 7

<!--

  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

-->

TEMA 8

<!--

  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"

  -->

TEMA 9

<!--

  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

  -->

TEMA 10

<!--

 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

 -->

TEMA 11

<!--

    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

-->

SECCIÓN 3 - CSS BÁSICO

TEMA 12

/\*

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

\*/

TEMA 13

/\*

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 -->

TEMA 14

/\*

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

\*/

TEMA 15

/\*

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

\*/

TEMA 16

/\*

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

\*/

TEMA 17

/\*

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 -->

TEMA 18

/\*

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" -->

TEMA 19

/\*

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

\*/

TEMA 20

/\*

  BOX SIZING

  Es una propiedad de CSS que normalmente utilizamos para afectar como cada componente de la caja interactúa entre sí. El modelo de caja es cómo los navegadores calculan el total del tamaño de un elemento.

  Porpiedad de css, afecta como funciona el modelo de caja predeterminado.

  El modelo de caja determina como los navegadores calculen el tamaño de un elemento

  content-box: tamaño que aplica sólo para el contenido contenido, si se agrega padding, border y margin la caja se agranda. Es como añadir capas a la caja original.

  Ejemplo

  div{

    width: 100px

    heigh: 100px

  }

  Esos 100px son sólo para el contenido, no incluyen padding ni border

  Al definir un alto y ancho se lo estoy aplicando sólo al contenido, el padding y el border están fuera de esa medida y si se los agrego se agranda la caja. Es la opción por defecto

  border-box: no nos posicionamos ni en el padding, ni en el contenido sino en el borde, le dice al navegador el tamaño que te doy inclucye el padding, el contenido y el borde.  el contenido se ajusta a la caja, es de tamaño fijo que incluye al padding, border, etc

  Mantiene el alto y ancho especificado y ajusta o achica el contenido cuando agrego padding o border

  Ejemplo

  div{

    box-sizing: border-box

    width: 100px

    heigh: 100px

  }

  En este caso los 100px van a ser el tamaño final de la caja incluyendo padding y border además del contenido. El contenido va a chicarse para ajustarse a las medidas del padding y del border de manera que entren en los 100px. El navegador hace los cálculos automáticamente

\*/

TEMA 21

/\*

  COLORES

  Las siguiente son las formas más comunes de dar color.

  RGB y RGBA: básicamente significa red, green, blue. alfa Los tres colores primarios de la luz. Tiene 256 combinaciones posibles para cada uno de los tres colores, es decir 16777216 colores posibles. Alfa habla de la transparencia, crea otro canal que detremina la opacidad del color.

  RGB; para usarlo debemos definir el valor de cada color que podemos elegir entre 0 y 255, la mezcla de los tres valores va a dar el color final. 255 es el máximo del color, 0 es el mínimos del color. Si ponemos todos los valores de cada color iguales voy a tener diferentes grises

  hexadecimal y hexadecimala: tiene la misma cantidad de colores que el rgb, tiene 16 posibilidades, sistema de base 16. Para definirlo uso #rrggbb

  0, 1, 3, 4, 5, 6, 7, 8, 9, a, b, c, d, e, f, g

  Hay 16 posibilidades para cada color que está dado por cada para de número.

  Ejemplo

  #RRGGBB

  #550033

  En el formato completo tenemos

  #EEO9FF

  EE es el rojo valor 1 permite elegir entre los 16 tonos posibles de rojo y valor 2 permite elegir entre otros 16 tonos posibles por cada uno del valor 1

  09 es el verde, valor 1 y valor 2, sigue la misma lógica de tonos que el rojo para el verde

  FF es el azul, valor 1 y valor 2, sigue la misma lógica de tonos que el rojo para el azul

  Hay una forma abreviada de definir los colores en hexadecimal que es usar sólo tres valores, cuando no nos interesa la especificidad del segúndo valor de cada color, en este caso estaríamos escogiendo sólo entre los 16 tonos principales de cada color

  Siguiendo el ejemplo anterior tendríamos

  #rgb

  #503

  Cantidad de colores que hay en hexadecimal

  4294967296

  También tiene un canal alfa

  https://rgbacolorpicker.com/

  HSL y HSLA: tiene menos colores que los otros dos. También tiene.

  El hue es básicamente una rueda que nos permite seleccionar el matiz

  hue es la color

  saturación o luminancia

  iluminación: los colores al 50% son los más fuertes

  Los colores pastel se caracterizan por tener poca saturación y mucha iluminación

  Los colores que voy a usar de fondo siempre tienen que contrastar, si uso un fondo oscuro tengo que usar un texto claro

  https://thumbs.dreamstime.com/z/la-rueda-de-colores-del-color-nombra-los-grados-rgb-78027630.jpg

  Selector de colores en los tres sistemas

  https://htmlcolorcodes.com/

  Cuando uno trabaja va a responder a la paleta colores de la empresa, pero es importante saber la teoría de colores para aprender y trabajar nuestros proyectos

hsl(319, 50%, 86%) es lo mismo que escribir red, la capa del rojo esta al máximo

\*/

TEMA 22

/\*

UNIDADES DE MEDIDA

No es correcto utilizar siempre pixeles como unidad de medida para todo al hacer un diseño

MEDIDAS ABSOLUTAS

Son ideales cuando queremos tener medidas precisas, para diseños muy específicos o imágenes que tienen tamaños fijos, pero no son la mejor opción para diseños responsive

pixeles: unidad absoluta. Son pequeñitos área de luz en nuestra pantalla. El área mínima de luz que tenemos en nuestra pantalla es el pixel

Medidas físicas y absolutas

Se usan por ejemplo para dar alto y ancho a las imágenes, para el tamaño de los textos para los márgenes y los padding

Son aquellas que se van a aplicar a la vida real, en el mundo físico. Necesitamos control, funciona para responsive

puntos (pt) y picas (pc): provienen del mundo de la impresion, cuando queremos crear un diseño para imprimir. También son medidas absolutas. No vamos a profundizar en estas unidades porque no se usan

milimetros (cm), centímetros (mm) y pulgadas (in): se usan tembíen para impresión. estas sí se usan un poquito más, en la pantalla puede cambiar, pero si lo imprimimos va a tener el tamaño exacto. Son formas de medir para definir cosas que se van a aplicar de forma físicas al mundo real, si lo voy a imprimir sí lo vamos a usar estas unidades, para banners, para maquetar interfaces táctiles, carteles

UNIDADES RELATIVAS

El valor de la unidad depende de otro valor, va a depender o otro elemento, el tamaño del contenedor, el tamaño de la fuente, etc. Estas unidades se usan en responsive design. Son ideales para crear diseños responsive

Porcentaje: es la más común, siempre es relativo al contenedor, responde al objeto contenedor. Definen algo respecto al objeto contenedor. En texto el porentaje funciona sobre el tamaño que tenga la fuente de la caja contenedora. En el margin usa como referencia al tamaño del padre, No se suele usar para margin, padding ni font-size. Se suele usar para dar a una imagen un porcentaje del contenedor

em: es la mejor forma de definir muchas cosas. Vale lo mismo que el font-size del elemento padre y este valor se multiplica por el numero delante de la unidad.

Ejemplo: padre 14px, 2em = 28px

rem: en lugar de busar el elemento padre va a busca el elemento raíz, que es <html>, luego funciona igual que el em

Si el elmento padre no tiene un font-size definido sigue hacia arriba buscando cual lo tiene y usa ese.

Si no lo encuentra en ningún lado usa el font-size por defecto del navegador que es 16px

MEDIDAS DEL VIEWPORT

Viewport es el espacio disponible que tengo en la ventana

vh: mide un porcentaje del tamaño viewport horizontal

vw: mide un porcentaje del tamaño viewport vertical

Estas unidades pueden usarse para hacer una ventana modal

vmax: compara el valor de vh y vw y toma el más grande

vmin: compara el valor de vh y vw y toma el más chico

Las siguientes unidades se aplican a tipografía

ex: Usualmente se lo usamos para el line-height, Toma el alto de la x y eso es un ex

ch: Toma el ancho del 0 y eso es 1ch

Estas dos unidades se utilizan para cosas muy específicas, nosotros no la vamos a trabajar

box-sizing: border-box hace que no aparezcan las barras de desplazamiento, y que el elemento entre en el espacio que tengo

USOS

Para todo lo relacionado a tipografía y espaciado em y rem son ideales

Para layout se usa cualquiera de las unidades vistas

Si queremos algo muy preciso usamos pixeles

Entrar a páginas conocidas y ver cuando usan cada udidad

\*/

TEMA 23

/\*

  FONDOS

  Background: es un shorthand engloba todas las posibilidades que tiene background, reune varias propiedades de background

  Ejemplo

  background: imagen posición / size reapeat attachement color

  Entre el position y el size sí o sí hay que poner la barra, de otra manera no funciona

  Al fondo le puedo dar una transparencia, con el cuarto valor o quinto y sexto valores del hexadecimal podemos dar transparencia al color elegido o la opacidad. Es normal aplicar transparencias

  background-color: define el color del fondo

    background-image: url("dirección de la imagen), permite usar una imagen como fondo

    background-size: puedo trabajar con porcentajes, 100%, ajusta la imagen al ancho del contenedor, con 100px la imagen se repite,

    - con contain se ajusta lo más posible para que la imagen entre, es decir obliga a la imagen a adaptarse. Fuerza al navegador a mostrar la imagen entera aunque sea una vez

    - con cover hace que no importa la resolución o tamaño la imagen entre entera en la pantalla

  background-position: con center, left, right, top o bottom, acomoda la imagen en el contenedor. También se pueden dar valores específicos en pixeles, da coordenadas específicos

    background-repeat:

    - no-repeat, no repite la imagen aunque sobre espacio

    - con repeat-x se repite a la derech

    - con repeat-y se repite hacia abajo

    background-attachment:

    - fixed permite dejar una imagen u objeto como fondo detrás de otro objeto al hacer scroll se ve como a través de una ventana

      - scroll: es la propiedad por defecto, para que no se salga de la página

\*/

TEMA 24

/\*

  GRADIENTES

  Son una transición de colores, es una magnitud física en función de la distancia, tenemos dos valores en los extremos y en el medio hay una transición que muestra como los valores van cambiando

  Se puede aplicar gradiente tanto con background como con background-image

    linear-gradient(color1, color2): es una función

  Ejemplo: linear-gradient(red, green)

  Va a convertir el fondo en un gradiente lineal, es interesante para hacer gradientes transparentes

    Para darle inclinación a los gradientes podemos usar to top, to bottom, to left, to right o un ángulo específico

  Se puede indicar desde donde queremos que inicie cada color, para eso usamos un porcentaje al lado del color

    radial-gradient(igual lineal), pero desde el centro

    conic-gradient(igual lineal): formato de cono

  Si como primer parámetro en el lineal usamos un angulo (20deg) el degradado se logra en diagonal

  El uso más común en diseño para el gradiente es usarlo para separar un texto claro de un fondo en el que no contresta

  Los gradientes conic y radial no se usan en diseño web

\*/

TEMA 25

/\*

  SOMBRAS

  Las sombras son muy importantes en el diseño web, porque lo hacen ver más realista y hasta tridimensional. Después crear como una jerarquía visual porque te ayudan a destacar un elemento, además te ayudan a la legibilidad a veces cuando hay que estar separando contenido del fondo

  Hay tres tipos de sombras

  1. box-shadow: la sombra que le damos a una caja

  2. text-shadow: la sombra que le damos a un texto

  3. drop-shadow: es la sombre que le damos a un componente x

  box-shadow: cuanto se corre la sombre hacia  derecha, cuanto se corre la sombre hacia abajo, difuminado o desenfoque, expansión de la sombre y finalmente color.

  El primer parámetro es el movimiento horizontal, también se le puede dar valores negativos, el segundo parámetro es el moviemiento vertical, el tercer parámetro es la difuminación, el cuarto parámetro es la extensión de la sombra que se va a difuminar, el último parámetro es el color, que se suele usar con transparencia

  text-shadow: misma estructura que box-shadow sin la expansión

  filter: drop-shadow(0px 0px 3px #000) por ejemplo, sólo sirve para imagenes png sin fondo o con un canal alfa

  No exagerar con sombres muy marcadas, probar en varios navegadores para verificar compatibilidad

\*/

SECCIÓN 4 - HTML INTERMEDIO/AVANZADO

TEMA 26

<!--

  METATAGS, COMENTARIIOS E ÍCONOS

  COMENTARIOS

  Es un código que escribimos que solamente es legible para el programador, el navegador también lo puede leer pero no lo va a hacer nada no lo va a interpretar

  se escribe con signo menor, exclamación, dos guines medios para la apertura y dos guines medios y el simbolo de mayor para el cierre

  FAVICONS

  Es el iconito o logo que aparece en la pestaña o parte superior de navegador junto al title del head.

  Imagen de 32 x 32 o 64 x 64 pixeles porque se ve muy chiquita

  Se recomienda el atributo type="image/png" para que el navegador la cargue más rápido que sirve, suma y ayuda

  En el atributo rel ponemos icons, rel viene de relationship e indica al navegador la relación del objeto que estamos incorporando

  Hay quienes usan la extensión .ico para tener más compatibilidad con windows, pwro hay sistemas que no lo muestran, entonces recomiendo usar .png con fondo transparene

  METATAGS

  Son etiquetas que tienen como única finalidad brindar información.

  Es un tag que proporcionar información de la página. Van en el head, son importante para el SEO, como se muestra en las redes sociales, y nadie va a ver nuestros metatags. Es importante también cuando un enlace la llame en una red social y también a veces para la accesibilidad

  Todas las metatags tienen dos atributos

  - name: define que tipo de información estas declarando

  - content: indica el contenido

  ATRIBUTOS DE LOS METATAGS

    - meta charset es la etiqueta META permite especificar los caracteres que se utilizarán en la página, UTF-8, le dice al navegador que codificación va a utilizar la página, es importante para usuaris con navegadores antiguos, permiten todos tipo de caracteres

  Ejemplo

  <meta charset="UTF-8">

    - lang de la etiqueta HTML: le indica al navegador en que idioma va a estar la página y le ahorra el tiempo de tener que entender en que lenguaje está la página

  Ejemplo

  <html lang="es">

      - meta viewport: con qué estamos trabajando

  Valor width=device-width: ajusta nuestra página a la pantalla de cualquier dispositivo

  initial-scale=1.0: se carga la página siempre en zoom 1

  Clave al trabajar con responsive design

  Ejemplo

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

  width=device-width: dice que el ancho total de la página va a ser igual al ancho del dispositivo

  initial-scale=1.0: hace que al abrir la página se cargue sin zoom o con zoom normal, haciendo que el contenido se adapte al dispositivo

    - meta description: una descripción de por lo menos 100 caracteres que descibe de qué trata la página

  - meta keyword: indica todas las palabras clave de la página. Si en la búsqueda de alguien figura una de tus palabras clave tu página va a tener más posibilidades de aparecer porque se trata de ese tema. El SEO da más prioridad en las búsquedaº

    - meta author: indica quien es el autor o creador de la página

  - meta robots: si content dice

      nofollow le dice a google no indixes me página en la busqueda

      nosnippet no permite que los buscadores extraigan una sección de la página y la muestre

      noarchive no permite a google pueda guardar nuestra página en caché, está bueno cuando nuestra página depende los cambios, y se mantiene en constante actualización, en este caso no tiene sentido guardarla en caché

      noimageindex es para que no se indexen las imágenes, no permite que las guarde o aparezcan en google fotos

    - meta title: guarda el título de la página

  - meta og: indica que mostrar en la búsqueda, title, image, description, url esta última muestra la dirección a donde se debe redirigir a la persona

  metatags.io

  Para utilizar esta página ya debemos tener subida la página a internet

  En esta página se puede crear los metatags necesarios para nuestra página y ayuda también a mejorar el SEO

-->

TEMA 27

<!--

  TEXTAREA

  Es básicamente como un input que permite excribir texto más largo, texto en varias líneas

  div: es sólo un bloque, no tiene ninguna función especial o semántica, se pueden usar todos los que se quiera, pero no se pueden reemplazar las etiquetas específicas aunque funcionen igual

  readonly en el textarea evita que se pueda modificar el contenido de la misma. El contenido se envia con el formulario y puedo hacer focus en el objeto, también se puede leer pero no modificar

  Hay atributos que no necesitan valor, con ponerlos es suficientes, se llaman atributos booleanos. tienen sólo dos valores, si está puesto es true y si no es false, sólo asigna esos dos valores

  disabled desactiva el campo de texto, la diferencia es que no se puede modificar y el contenido no es enviado con el formulario y no puedo hacer foco en él

  max-length determina la cantidad de caracteres máximos que se pueden escribir en el cuadro. Lo que se suele hacer en ese caso es que cuando alcance el límite de caracteres le aparezca un mensajitos diciendo que no puede escribir más de (la cantidad de caracteres límite)

  Para hacer que el borde no se aplique a todos los lados, lo definimos primero en 0 y luego definimos el borde del lado que queremos estilizar

  Para que el placeholder dentro del textarea se vea no hay que dejar espacio ni línea entre la etiqueta de apertura y la de cierre

  wrap: controla como se controlan los saltos de línea

  - soft: no incluye los saltos de línea en el texto

  - hard: sí incluye los saltos de línea en el texto

  Con el que viene por defecto es sufieciente porque en la mayoría de los casos lo usamos así, si lo vamos a usar es porque vamos a darle alguna funcionalidad puntual que dependa de esos espacios de línea, para esto tenemos que saber más de programación

  col: es la cantidad de columnas que puede tener

  rows: es la cantidad de filas o líneas que puede tener, esto lo usamos para no tener que calcular específicamente el alto del textarea

-->

/\*

  TEXTAREA

  min-width El textarea viene configurado para que si le ponemos un width del 100% no se pone al 100%, esto es por sus propiedades iniciales, las que trae configuradas por defecto, si quieren que se aplique ese porcentaje tenemos que usar min-width: 100%, esta propiedad evita que se amplie hacia los costados, pero solo hacia adentro, porque hacia afuera se puede cambiar el tamaño

  min-heigth: como mínimo puede tener este alto

  max-height: como máximo puede tener este alto

  Al cargar la página lo hace con el alto y/o ancho mínimo

  resize define si puede cambiarse el tamaño del textarea

  - both permite cambiar el tamaño en el eje x y en el eje y

  - vertical se agranda solo en el eje vertical

  - horizontal se agranda solo en el eje horizontal

  - none el tamaño no cambia, no se expande en ninguna dirección, sólo agrega una scroll bar

  La tipografía no va a ser la que nosotros le pongamos en body, si la queremos cambiar debemos hacerlo en el propio textarea, no va a tomar lo que pongamos en body

  form-sizing permite que el textarea crezca con nuestro texto, esta

  - content: el contenido se adapta

  propiedad todavía no está disponible

  Lo que hace Dalto es dejar el resize en none y darle un min-width y dejarlo así

\*/

TEMA 28

<!--

  LABEL

  El label acompaña a los imputs, permitiendo hacer foco en ellos cuando se hace click en su label

  Todos los imputs tienen que estar etiquetados por un label. Mejora la accesibilidad

  Para enlazarla a los inputs podemos

  usar for en el label y id en el input con el mismo valor

  envolver el input con el label. Otra forma de hacer esto es envolver el input con las etiquetas de label, esta forma no necesitamos agregar ni el for ni el id

  Esta última forma es buena para dar click a términos y condiciones, ya que es difícil clickear el checkbox

  Lo único malo de esta forma de asociar el label al input es que si los queremos por separado no los vamos a poder tener, ya que siempre van a ir de la mano

  Los textos deben ser cortos, concisos y claros

    Sirve también con los checkbox para no tener que darle click al cuadradito que a veces se hace dificil, clickeamos en el label y ya está

  No es bueno que el placeholder de un input diga lo mismo que el label de ese input, lo que hacemos es poner un ejemplo

  nombre completo: John Doe

  email: ejemploe@gmail.com

  con esto le estamos dando una idea al usuario de como debe completar los campos

-->

/\*

  LABELS

  Los labels son elementos en lñínea y no podemos darles margen así nomás. Lo que se puede hacer es darle un line-height para separar las líneas de texto

\*/

TEMA 29

<!--

  SELECT, DATALIST Y OPTION

  SELECT (desplegable): es como un campo de entrada que nos permite seleccionar una opción de entre varias opciones.

  La primera vez aparece solo la flechita, no hay opciones, tenemos que agregárselas

  El select debe tener un name

  value: es importante porque es el valor que el servidor va a recibir una vez enviado el form en lugar de la opción seleccionada en la lista.

  No tenemos que hacer que el servidor reciba acentos o espacios, etc. todo en una frase, todo en minúsculas

  Lo que el usuario ve son los options, pero el servidor recibe el value de la opción que eligió el usuario

  DATALIST: es un input en el que escribimos algo y se completa automáticamente con opciones que nosotros predefinimos

    OPTIONS: son los items de los select y los datalist. Los options tienen un value, esto es algo interno, no lovemos nosotros, es para cuando trabajamos con javascript saber leer este dato

  En el caso del datalist si no le pongo un value al option al enviar el formulario se le autoasigna el contenido del option

  También el contenido se puede poner como value y en el contenido usar un texto como descripción, este último no se va a enviar al servidor, sólo se envía el value, por eso esta es una mejor opción

  El datalist tiene que tener un input que lleva un atributo list y el datalist lleva un atributo id igual al list del input

  No es necesario cerrar la etiqueta option, cuando esta no se cierra, html le dice a los navegadores que la pueden cerrar ellos automáticamente. Si no lo cerramos los navegadores antiguos no lo van a ver. Es importante hacer las cosas para que los navegadores antiguos las puedan ver

-->

TEMA 30

/\*

  SELECT

  Para cambiar el color de fondo de las options debemos ponerle un background color del mismo color que el formulario

  Los navegadores están programados para no retocar mucho las options por lo que no podemos usar padding, margin, etc. Para esto debemos utilizar una librería de javascript

\*/

TEMA 31

<!--

  FIELDSET Y LEGEND

  Sirven para agrupar y etiquetar los elementos de un formulario

    Fieldset: delimita los datos

  Legend: da un título a la sección

input type number: permite escribir solamente números y la letra e, también agrega las flechitas arriba y abajo

-->

TEMA 32

<!--

  DETAILS AND SUMMARY

  Crean una funcionalidad nativa para crear estos contenidos  desplegables. Se puede poner el contenido que queramos

    Se utilizan juntos y permiten crear contenido desplegable en la página, también se puede usar para mostrar información adiciónal

  Summary: es lo que se va a ver

  Datails: es el contenido oculto que se muestra al hacer click en el summary

  Sirven para preguntas frecuentes o información adiciónal entre otras cosas

 -->

TEMA 33

/\*

  Para dar estilo a un detail sólo cuando está abierto usamos

  details[open] {}

  Se puede dar estilo a cada estado

\*/

TEMA 34

<!--

  ENLACES AVANZADOS

  Un link permite navegar a través de la web, dentro y fuera de nuestro sitio web

  Link a un archivo local: se usa el elemento <a href="archivoalquevoy.extensión">

  Link a un sección específica de una página: se coloca un id en el lugar de la página al que queremos ir y ponemos en el elemento <a href="#iddellugardondevamos", El id del elemento al que queremos ir con un numeral. Es un enlace que nos manda a otro elemento en lugar de a otra página

  El atributo download en el elemento <a> permite descargar un archivo local

  Target="\_blank" abre la posibilidad a ser redirigido a sitios maliciosos u otro sitio que no es al que quiere ir a través de windows opener. Al abrir la publicidad, esa página me puede redirigir a otra página, no va a poder hacer cambios

  rel="noopener" evita ese posible redireccionamiento. También puede mejorar el rendimiento porque la página se abre en separado y podemos evitar ataques de phishing

  rel="noreferrer" además de no permitir el acceso al objeto windows.opener, tampoco permite ver los encabezados, por lo tanto la página no va a saber de donde vino el link, también evita el robo de datos.

  rel="nofollow" dice a los motores de búsqueda esta página no la sigas, no la indexes. La web es una red de enlaces relacionados. Permite no darle identidad a enlaces que no conocemos que pueden compartir usuarios de nuestra página. Si una página de una empresa grande enlaza una página, google dice esa página tiene que ser importante, pero después ya no va a tomar encuenta las recomendaciones de esa empresa, de esta manera no se pierde posicionamiento

  href="mailto:correo@gmail.com" permite direccionar al editor de correo que usamos para enviar directamente un mensaje desde el link

  href="tel:nrodeteléfono" permite llamar a ese número a partir del enlace

-->

TEMA 35

<!--

  TABLAS

  Son una forma de organizar datos en filas y columnas

  Haces años las tablas se usaban para todo, estructurar páginas, diseño, etc. Hoy en dia la única razón para usar tablas es principalmente cuando queremos mostrar datos tabulares y tenemos la necesidad de hacer algo, que tengan relación entre sí.

  No se usa para listar datos de usuario, no para organizar elementos. Se usa para desarrollar correos. También se puede armar su firma

   <table></table>: define la tabla

  <tr></tr>: arma una fila

  <td></td>: define celdas para esa fila. Son las celdas de la tabla

  Fila: se presentan en forma horizontal

  columna: se muestra en forma vertical

  No se pueden crear columnas, estas se crean a partir de las filas

  <th></th>: determina los encabezados de columna. Posiciona la página mejor, porque asocia el encabezado a su contenido, van dentro de un tr. El navegador nos ayuda a tener mejor el SEO si utilizamos th para los encabezados en lugar de usar td, cuando google ve la página dice ah estos son los encabezados. También lo usamos para crear subtotales y totales

  - colspan: permite permite unir celdas de forma horizontal

  - rowspan: permite unir celdad de foma vertical

  <thead></thead>: indica una sección con encabezados y sin datos

  <tbody></tbody>: es donde van a estar todos los datos

  <tfoot></tfoot>:

  Estas tres etiquetas determinan toda una sección

 -->

/\*

Para visualizar la tabla le damos estilo de borde

\*/

TEMA 36

<!--

  AUDIO Y VIDEO

  Hace cinco años todavía se oía hablar de Flash, hoy eso ya prácticamente no existe, es completamente inseguro y ya no va más. En ese momento HTML5 dijo no puedo depender de flash para la carga del contenido multimedia

  Después de que flash dejó de ser soportado por los navegadores html5 creó dos nuevas etiquetas

  Para reproducir videos se usa la etiqueta video y el atributo src con el nombre del archivo de video o el enlace al video.

  El video no se reproduce al cargar la página

  <audio> permite reproducir audio

  <video> permite reproducir video

  - autoplay: reproduce el video automáticamente, pero al recargar la página ya no lo hace, el video no se puede reproducir a menos que el usuario haga click o interactúe de alguna manera con la página

  - controls: agrega controles de reproducción al video, permite que el video se reproduzca automáticamente

  - muted: mutea el video pero al recargar la página lo reproduce automáticamente, si no lo ponemos no se reproduce, y el usuario tiene que subir el volumen

  Autoplay sólo funciona si está con muted también, hace unos años se modificó para que el usuario tenga que interactuar sí o sí con la página aunque sea una vez

  El texto entre la etiqueta de apertura y la de cierre de video sólo aparece si el navegador no soporta al video y no lo puede cargar

  El atributo poster="imagen.png" permite mostrar una pantalla de preloading o una miniatura

  Si en lugar de video usamos audio con un archivo de video el navegador carga todo el video completo y luego reproduce sólo el audio, no es recomendable hacer esto porque tarda más en cargar el audio, primero se debe convertir a audio y luego usar este nuevo archivo con la etiqueta audio

  Antes el formato de youtube era 16:9, tenía un formato fijo, y si la resolución del video no era la correcta le ponía un fondo negro, que se veía como barras, a los costados o arriba y abajo

  - loop: permite reproducir el video en loop

  La etiqueta <track> permite agregar subtítulos al video, es autocerrable

  src: permite agregar un archivo con el subtítulo, idnicando el nombre del archivo

  - default: permite que aparezca por defecto, de lo contrario tenemos que ponerlos nosotros

  - kind: permite indicar si son subtítulos o caption

  - srclang="lenguaje": indica el idioma de los subtítulos

-->

/\*

  Los archivos web vtt para subtítulos tienen un formato propio que no se puede modificar.

  Si queremos darle formato al texto tenemos que formatear el archivo o utilizar javascript para hacerlo

\*/

TEMA 37

<!--

  CARGA DIFERIDA O LAZY LOADING

  Es un concepto que dice que el contenido de una página se tiene que ir cargando a medida que el usuario va llegando a esas secciones. Va cargando lo que el usaurio va viendo, si hay imágenes más abajo fuera de la visión no las va a cargar, primero carga lo que el usuario va viendo para que la página no se haga lenta.

  Es la prioridad que le estamos dando a la imagen en cuanto a tiempo de carga

  loading="lazy" permite cargar la imagen al último y cuando llego a ella, sin esto carga primero las imágenes y no carga el resto de la página. Damos prioridad de carga, le decimos al navegador cargá la imagen cuando puedas y le da prioridad a todo lo demás. Recomiendo no aplicarlo por ahora,  hasta que no sea necesario, porque puede crear problemas y no cargar algunos elementos si se usa de manera peligrosa

  <br> son atributos que ya no se usan, que sirve para dejar línea en blanco

-->

<!--

  HTML OBSOLETO

  Son etiquetas que fueron discontinuadas y no se recomienda usarla en HTML en ninguna circunstancia

  Por qué una etiqueta se vuelve obsoleta?

  Primero porque los estándares avanzan y son innecesarias o se reeemplazan por alternativas mejores. Después  por su enfoque en css, hay etiquetas cuya funcionalidad se puede hacer con css, como la funcionalidad de HTML no es dar estilo ni diseño a la página, aquellas cuya función era dar estilo se descontinuaron. Otra razón es por compatibilidad y rendimiento algunas etiquetas pueden trear problemas de compatibilidad entre navegadores y perjudicar el rendimiento de la página. Después accesibilidad web

  Las etiquetas que aparecen en blanco no van, son obsoletas

  font: para dar formato al texto, permitía dar tamaño y color a la fuente

  center: para centrar texto

  blink: para que el texto titile

  big: para agrandar el texto

  strike: para tachar texto, ahora se usa la etiqueta del que se utiliza para indicar que algo no es más de esa manera o s para indicar que algo ya no va

  tt: texto monospace para código, se reemplazó por la etiqueta code que muestra el texto con el formato de un editor de texto

  u: subraya texto, no es obsoleta pero no se recomienda

  Los atributos que suelen dar estilo a un elemento desde el atributo ya no se usan

  b: negrita, no es obsoleta pero no se recomienda, se recomienda usar,porque esto se hace desde CSS, si se quiere resaltar algo se usa strong porque tiene un sentido semántico para el seo que dice esto es importante. strong se usa para algo más serio, más importante

  em: pone en negrita y cursiva el texto, da énfasis a ese texto

-->

TEMA 38

<!--

  HTML SEMÁNTICO

  Significa utilizar etiquetas que tienen un significado para dar un significado a la sección que estamos creando, es darle significado a lo que estamos haciendo, porque la otra forma es divs y spans para todo

  div: etiqueta general de bloque

  span: etiqueta general de línea

  No usar sólo divs y spans da mejor accesibilidad a la página para los lectores de pantalla y una mejor comprensión a los motores de búsqueda para el SEO o que aumenta nuestro ranking de posicionamientos, no ubica más aarriba en las búsquedas y mantenibilidad, hace que sea más fácil de mantener y que otros desarrolladores lo entiendan cuando lo trabajen

  header: es el encabezado de la página, contiene logotipos, barra de navegación...

  nav: le indica al navegador que esa sección es para navegar

  Puede haber varios headers en una página

  section: hace referencia a una sección del contenido dividiéndolo en secciones o temas

  main: es el contenido principal de nuestra página o apartado más importante

  aside: es una sección secundaria pero que en cierta forma tiene que ver con el contenido principal

  Recomendación: no usen etiquetas sólo por usarlas, búsquenle un sentido a las etiquetas, que estoy queriendo hacer?, hay alguna etiqueta parecida?

  footer: es el pie de página, allí van los links, términos y condiciones, copyright

-->

TEMA 39

<!--

  ACCESIBILIDAD WEB

  Es hacer una página  web que sea accesible para todos

  Una página web con mejor accesibilidad es una página web con mejor posicionamiento en los buscadores

  Es una página que todos sin importar sus discapacidades o capacidades pueden utilizar

  Entender el DOM

  Decimos este codigo html lo lee el navegador y nos da un resultado lindo, esto no es así, el navegador crea un dom, un árbol que contiene todas las propiedades o elementos de html

  Código html pasa al navegador que lo convierte en el dom y eso es lo que nosotros vemos al final la página web hermosa

  Para las personas con discapacidad el navegador crea una copia del dom que pasa por tecnologías de accesibilidad que después el usuario que utiliza las tecnología accesible puede acceder al contenido, este es el flujo real del contenido. Esta copia está hecha especialmente para las tecnologías de accesibilidad

  Intención: crear un HTML que pueda ser entendido por las personas con discapacidad a través de sus herramientas de accesibilidad

  widows+ctrl+enter

  Extensión chrome box especial para chrome, herramienta de accesibilidad

  Para esto necesito usar texto alternativo, con el atributo alt de las imágenes, en los a el texto entre las etiquetas de apertura y cierre es lo que va a leer el lector de pantalla

  Toda página web debería poder ser navegable con la tecla tab y el enter para hacer todo lo que se hace con el mouse, este es el principio más básico de todos

  aria-label: es el alt para los lectores de pantalla en cualquier elemento

  role del id: indica el rol que va a cumplir una etiqueta

  En los formularios

  TAREA:

  1. Ver video: Cómo hacer PÁGINAS WEB INCLUSIVAS (Accesibilidad)

  Analizar->lighthouse sirve para analizar la página por accesibilidad

  Todos los elementos tienen que ser focuseables, y tienen que seguir un orden para usar tab

-->

SECCIÓN 5 - CSS INTERMEDIO

TEMA 40

/\*

SELECTORES AVANZADOS

CSS básicamente funciona tomando un elemento o seleccionando algo y cambiando o sobreescribiendo sus propiedades

Dos cosas que son las más importantes para hacer cualquier cosa con CSS

1. Saber cómo darle estilo a un elemento, conocer sus propiedades. Las propiedades nos permiten modificar las características de un elemento, las características que sobre todo son visibles

2. Saber cómo seleccionar eso que queremos modificar. Pero tenemos muchas formas en las que podemos tener problemas si no sabemos seleccionar correctamente

Podemos utilizar condiciones para seleccionar elementos que cumplen esas condiciones, tambien podemos seleccionar por estados.

Selector de atributo: vamos a dar estilo a un lemento que tenga un atributo específico.

Se usa

elemento[atributo="valor"]{}.

Si un elemento tienen un atributo definido o ese atributo tienen un valor específico. Si uso el signo $ después del atributo le digo que puede arrancar con algo pero tiene que terminar con lo que figura como valor. Significa termina con...

Se usa

img[src$="severus"]{}

El símbolo ^ significa arranaca o comienza con...

Los tres puntos que aparecen cuando me posiciono sobre el selector significa hay algo más

Se usa

img[src^="severus"]{}

Sin los símbolos anteriores ($ o ^) tiene que ser el valor exacto

Selectores descendientes:

Se usa

elemento1 elemento\_dentro\_elemento1{}

Selecciona de forma descendente

Selector de hijo directo

elemento\_padre > elemento\_hijo{}: indica que elemento interno tiene que ser hijo directo de elemento externo

Selector Hermano Adyascente: hermano1 + hermano2 {} Permite seleccionar al siguiente elemento, que sigue inmediatamente después del indicado

Selector de Hermanos Generales: da estilo a todos los elementos que vengan después del indicado en su mismo nivel

hermano1 ~ hermano2 {}

Los selectores de hijos y hermanos se pueden usar con más de un elemento siguiendo a otro

Selector de múltiples elementos: elemento1, elemento2{} Dan estilo a varios tipos de elementos al mismo tiempo, con las mismas propiedades

\*/

TEMA 41

/\*

HERENCIA

Dice que algunas propiedades pueden heredarse y otras no

Por ejemplo a veces aplico un estilo que se termina aplicando al hijo pero no se aplica al padre

Si le pongo un color al body, todo el texto dentro de él toma ese color. esto se debe a que body tiene una popiedad que dice que si le doy un estilo los hijos lo heredan, la propiedades que le doy a body se van heredando hasta que la cambio en un elemento especificamente y ahí ya no se aplica el estilo dado en body para ese elemento. Esto se aplica a todos los padres e hijos en el documento

color: inherit es el color por defecto que el elemento hereda de su padre o del elemento siguiente hacia arriba.

inherit intenta agarrarse de un objeto que da la propiedad buscada más arriba

La propiedad width no es heredable por defecto, no tiene inherit por defecto, se lo tengo que dar manualmente

Los elementos en bloque por defecto tienen un width del 100% y para los elementos en línea es auto

initial: indica que la propiedad que tiene ese valor no hereda los valores de su padre, toma el valor que tiene el navegador por defecto.

unset: restablece al valor inicial y actúa como inherit, revierte al valor del navegador. El estilo del usuario es por ejemplo el que se le da a través de un plugin o modifia el navegador.

revert: intenta primero encontrar el estilo del usuario, si el usuario definió un estilo lo revierte a eso y si no lo revierte a inherit, al valor del navegador

inherit se puede usar para los textareas que no toman el formato a heredar, ya que los inputs y los textarea no tienen esta propiedad pof defecto

Se define el font-family una vez y se hereda en los inputs de los formularios si usamos inherit

El a también podemos hacer que herede los colores con esta propiedad

En este caso todos los elementos toman estos estilos hasta que los cambie independientemente

\*/

TEMA 42

/\*

CASCADA Y ESPECIFICIDAD

CSS como ya sabemos significa Cascading Stylesheets u Hojas de EStilo en Cascada. Es una de las cosas que componen el CSS

Cuando están trabajando en un sitio ustedes observan que aplican un estilo a un elemento y ese elemento no está tomando ese estilo, esto se debe a que ese estilo depende de una puntuación

La especificidad y la cascada se encargan de resolver los conflictos que ocurren cuando dos estilos chocan sobre un elemento o se sobreescriben.

Ciertas formas de seleccionar tienen mayor puntuación y se le va a dar prioridad a ella

La especificidad: es como la puntuación de un estilo. Si tienen la misma especificidad sigue el orden en que aparecen las reglas. Si pongo dos selectores o aplico los métods de selección combinados se adicionan las especificidades

Ejemplo: body .titulo{} es más específico que .título

Esto puede aplicarse como fallback, cuando no se puede aplicar una regla, se pasa a la siguiente

La cascada nos díce: el orden en el que damos los estilos sí importa

Si dos reglas tienen el mismo valor de especificidad se va a decidir cual aplicar por el orden y va a quedar la última regla que apliquemos

Qué pasa si un elemento tiene más especificidad? Esto significa que ese elemento es más específico. Por ejemplo una clase es más específica que un elemento como selector

Ejemplo

.class (0, 1, 0)

h1 (0, 0, 1)

En este caso la cascada ya no aplica porque la clase tiene mayor especifididad, la importancia está dada de derecha a izquierda

#id (1, 0, 0) este tiene la mayor especificidad

En estos casos el orden no es importante

La especificidad es la forma de resolver confilctos

La prioridad la da la especificidad, la cual es un concepto de cascada

Hay algo aún más importante al aplicar estilos y es el estilo en línea, cuando se aplica directamente a un elemento en el archivo HTML

Acá ya no funciona la especificidad, lo menos importante es lo que indica el navegador

Niveles de importancia

De mayor a menor los niveles son los siguiente:

- !IMPORTANT: tiene más importancia que todo lo demás en los estilos

- Atributo Style

- Etiqueta Style - Archivo CSS externo: tienen el mismo nivel de importancia

- Cambios hechos por el usuario (extensiones, configuración, etc)

- Navegador

fallback: es un estilo que se usa en caso que no funcione lo que aplicamos, son opciones de respaldo

\*/

TEMA 43

/\*

  PSEUDO-CLASES

  una forma de seleccionar ciertos estados o momentos putuales de un elemento

  Son palabras clave que nos permiten seleccionar un estado específico de un elemento que seleccionamos. Cuando la agrego voy a poder modificar el elemento cuando esté en este estado

  :hover selecciona el elemento cuando paso el mouse por encima. Este evento es mouse over, cuando el mouse está por encima del elemento se activa la regla. Es la pseudoclase más utilizada

  :active define el color de un elemento cuando está activo, mientras se esté clickeando el elemento. Se aplica con los dos clicks izquierdo y derecho

  :first-child aplica la regla sólo al primer elemento que encuentra del tipo indicado

  :last-child aplica la regla sólo al último elemento que encuentre de ese tipo

  :nth-child(número elemento) selecciona al elemento indicado entre los paréntesis. Si entre paréntesis ponemos un número acompañado por n, va a comenzar en el elemento indicado y va a saltar de número en numero. Por ejemplo

  :nth-child(3n) a partir del tercer elemento va a saltar de tres en tres. Esta pseudo-clase se usa para dar estilo a tablas. Tiene en cuenta a todos los hijos

  :nth-of-type() hace lo mismo pero sólo con elementos del mismo tipo. Tiene sólo en cuenta a los elementos del mismo tipo

  :not() no aplica la regla al elemento, clase o id indicado entre paréntesis

  :empty selecciona sólo aquellos elementos del tipo indicado que estén vacíos. Se usa, por ejemplo. para en caso de tener un contenido mostrar un estilo y en caso de estar vacío que no se muestre nada

  :root selecciona al elemento raíz. Va siempre arriba de todo en el css y no acompaña a ningún elemento, va solo. Es el elemento base, y todos los demas lo heredan

  :checked selecciona un checkbox sólo si está chequeado o tildado

  :link y :visited se usan con el elemento a, el primero lo selecciona si el link no fue visitado aún, y el segundo se selecciona si el link ya fue visitado

  :invalid se usa con input y se selecciona si lo que se ingresó al input no es válido

  :valid se selecciona si lo que se introdujo al input es válido

  Material de referencia

  mdn pseudo-clases, muestra toda la lista de pseudo-clases de css.

\*/

TEMA 44

/\*

  PSEUDO-ELEMENTOS

  Son palabras clave que trabajan con partes de un elemento, no es un elemento como tal sino una parte de un elemento

  Los pseudoelementos empiezan con dos puntos ::seudoelemento

  Los pseudoelementos tienen sus dificultades, hay propidades o estilos que no funcionan en ellos

  ::first-letter toma la primera letra de un texto

  ::first-line toma la primera línea de un texto, a medida que se agranda la línea va tomando el estilo indicado para la primera línea

  ::selection permite cambiar el estilo de selección del contenido de la página o del elemento indicado

  ::placeholder permite cambiar el estilo del texto en el placeholder de los inputs. El padding se aplica lateralmente

  ::marker cambia el color de las viñetas de los li

  ::before permite poner un texto antes del elemento indicado utilizando la propiedad content. Este texto no va a aparecer si inspeccionamos la página y no puede seleccionarse

  ::after hace lo mismo pero coloca el texto al después de un elemento

\*/

TEMA 45

/\*

METODOLOGÍA BEM

BEM: Block Element Modifier. Se compone de Bloques, elementos y modificadores

Metodología: es una forma específica de trabajar.

En CSS se utiliza una metodología porque:

1. Nos permite reutilizar código que es el propósito principal de esta metodología, que es optimizar código

Por ejemplo: si tenemos dos botones diferentes lo que hacemos es darle un estilo a un botón y otro estilo a otro botón, entonces tenemos dos botones diferentes pero hay partes del código que se repiten.

Esta metodología es una clase de estilo predefinido nos permite crear una base que luego con los modificadores podemos ir modificando para no tener que estar escribiendo el código una y otra y otra vez

2. Se resuelven problemas de especificidad. Si van a generadores de especificidad o páginas que revisan la especificidad de su código normalmente es complicada la forma con que manejan la especificidad los proyectos que no trabajan con una metodología. BEM nos permite crear clases para con sólo verlas saber a que se refieren.

3. Tienen código más legible. Si voy a un elemento sé que hace, donde está y qué quiero modificar

Bloque: componente independiente que no depende de nadie para existir. Ejemplo: una barra de navegación, una tarjeta es un bloque, un formulario es un bloque

Elemento: hace refeencia a una parte de un bloque, un pedazo del bloque que por sí solo no tiene sentido pero dentro del bloque sí

Modificadores; son una forma de modificar un bloque o un elemento

ESTILO SIN METODOLOGÍA BEM

.lista {

  border: 2px solid #000;

}

.ingrediente {

  color: grey;

  margin: 10px 0;

  list-style: square;

}

.ingrediente-importante {

  color: blue;

  margin: 10px 0;

  list-style: square;

}

.ingrediente-danino {

  color: red;

  margin: 10px 0;

  list-style: square;

}

Con la metodología BEM vamos a no repetir código

Ejemplo

El bloque principal es lista

Los elementos de la lista se esriben lista\_\_item, se usan dos guiones bajos para no confundir con un nombre u otro valor

La clase modificadora se agrega luego de dejar un espacio después de la clase de cada elemento, y se escribe con el nombre del elemento, en este caso item, dos guienes medios y el nombre de la modificación que vamos a hacer, en este caso danino e importante, entonces ambos elementos quedarían;

item--danino e item--importante

Si quisiéramos modificar un bloque en lugar de poner el nombre del elemento ponemos el nombre del bloque, en este caso sería lista lista--vieja

La propiedad opacity cambia la opacidad o transparencia de un elemento

\*/

TEMA 46

/\*

  DISPLAY

  display block: convierte el elemento en un bloque, lo primero que hacemos es decirle que ocupe todo el ancho disponible, lo segundo es que comienza una nueva línea. Permiten un margin, permiten un padding, pero si le saco el block, no acepta que margin y padding que funcionen correctamente, en este caso no afectan el tamaño de la caja

  La propiedad por defecto es inline

  inline: hace que un bloque se convierta en un elemento en línea algunos elemento son inline,están en la misma línea. No aceptan margin ni padding un poco se rompe. Sólo acepta margin y padding de forma horizontal, de forma vertical no afectan el flujo del diseño, no aceptan cambios con width o height, el tamaño se ajusta al contenido

  inline-block: es una mezcal entre inline y block, no ocupan todo el ancho, no empiezan una nueva línea y se pueden apilar. Acepta width y height

  none: el elemento sigue estando pero no se muestra y no ocupa espacio en la página y los elementos adyascentes actúan como si no existiera, no forma el flujo de la página

  Hay varios tipos de display

\*/

TEMA 47

/\*

  POSICIÓN RELATIVA

  Es una forma de posicionar elementos. Estas dos formas son absolute y relative, vamos a ver qué significan estas dos formas de posicionar elementos

  Contexto de Apilamiento: permite apilar u ordenar los elementos en capas en el eje z, es como tridimensionalizar el html. Podemos hacer que una capa aparezca detrás de otra o adelante de otra

  Separa los elementos por capa

  relative: le dice al navegador, elemento mantené tu posición original pero yo si quiero ahora puedo moverte arriba o abajo o hacia los costados y además vas a ser un punto de referencia para los elementos que estén adentro tuyo. Mantengo el lugar original de la caja pero puedo moverme visualmente. El siguiente elemento parte al final de la posición original del elemento anterior aunque a este lo hayamos movido, es como si el lugar de la caja lo estuviéramos reservando aunque lo hayamos movido

  No podemos considerar un elemento posicionado si tiene position static, si es estático

  Tenemos cuatro propiedades que se aplican a la posición

  top, bottom, left y right: estas indican cuanto se va a desplazar desde arriba, desde abajo, desde la iquierda o desde la derecha, y el desplazamiento es en sentido contrario a su nombre. Ejemplo si uso top, desde arriba se mueva una x distancia hacia abajo, y así con todas

  top y left tienen prioridad sobre bottom y right y se van a aplicar no importa si uso las últimas

  position funciona como capas y cada una nueva se ubica por encima de la anterior

  La propiedad top tiene más prioridad que bottom

  La propiedad left tiene más prioridad sobre right. Las dos propiedades más importante son top y left

  z-index: permite mover un elemento en el eje <, adelante y atrás, usa sólo un número, no lleva ninguna unidad. El elemento que se quiere ubicar por encima o delante de otro debe tener un valor de z-index mayor que el que va a quedar por detrás. Una buena técnica es darle valores de 10 en adelante y luego de diez en diez. Sólo funciona si las cajas están posicionadas, si no, no

  static: no se puede considerar que un elemento está posicionado cuando está static

\*/

TEMA 48

/\*

  POSICIÓN ABSOLUTA

  Las cajas que están posicionadas crear un contexto de apilamiento y se posicionan más adelante, el elemento no desaparece, sigue atrás

  El elemento ya no ocupa espacio en la página, y se coloca encima del elemento que le sigue. Le dice al objeto que no ocupe lugar en el flujo de la página pero su posición sigue siendo absoluta, no mantiene el lugar

  El desplazamiento va a ser en función del html, del margen superior iquierdo y no relativo a su posición original como en el caso de relative

  Pero si queremos que el punto de referencia sea un contenedor, si a este le asignamos position relative el punto de referencia o de partida del elemento con posición absoluta va a ser el contenedor

  Para top, bottom, left y right el elemento con position absolute va a tomar como referencia los bordes del contenedor que tenga posicion relativ. Si ponemos estas cuatro propiedades en cero y damos margin auto al elemento, este se centra.

  Pierde el espacio en la página.

  Con relative los objetos toman como punto de referencia su propia posición original, pero con absolute toman como pumto de referencia al elemento contenedor

  Si al objeto le damos top, botton, left, right en 0 y a margin auto el elemento se centra en el contenedor

  Los objetos con posicion absoluta se mueven de forma diferente

  Son relativos a su contenedor si ningún contenedor tiene posición relativa toma como refeerencia al html

\*/

TEMA 49

<!--

  VENTANA MODAL

  ETIQUETA DIÁLOG

  Se usa para crear ventanas modal, se coloca todo el contenido entre la etiqueta de apertura y la de cierre.

  Cualquier botón dentro de un form con el method dialog y que esté entre las etiquetas dialog cierran la ventana

  Ejemplo:

  <dialog open>

    <div class="modal\_\_content">

        <h2 class="modal\_\_title">VENTANA MODAL</h2>

        <p class="modal\_\_text">

          Lorem ipsum dolor, sit amet consectetur adipisicing elit. Ab ea, perferendis modi voluptatem doloremque odit pariatur  emporibus consequatur, dolores minima commodi aspernatur beatae. Similique laborum, vitae aspernatur numquam rem distinctio?

        </p>

        <form method="dialog">

          <button class="modal\_\_close-button"></button>

        </form>

      </div>

  </dialog>

  open hace que la ventana se muestro si no está no se muestra

  Dialog viene configurado para que si tiene el atributo open se muestra y si no no, este atributo se cierra solo y tampoco tenemos que poner el position absolute porque se posiciona solo

--> /\*

 ELEMENTO DIALOG

  ::backdrop: es un pseudo elemento que permite estilizar un elemento dialog

  Viene con la propiedad position absolute por defecto

\*/

TEMA 50

/\*

  POSITION FIXED Y STICKY

  Fixed: al principio parece que está absolute. Para arrancar, se va del flujo natural de la página, el espacio reservado se elimina. Siempre se va a posicionar respecto a la ventana. La característica principal es que si hacemos scroll se va a quedar siempre en el lugar

  Puede servir para crear chats que se mantengan siempre visibles, para ocultar el contenido hasta realizar cierta acción o en las barras de navegación

    STICKY: actúa como relative hasta que llega al lugar en que se le indicó que se quede fija. Al principio se desplaza con la página y cuando llega al valor que se le indicó en top, bottom y desde ahí se queda fijo

  Se usaa para encabezados

\*/

TEMA 51

/\*

  TRANSICIONES

  Son el intervalo de cambio de una propiedad entre un valor y otro

  Puedo hacer que al cambiar de color en un hover por ejemplo se vea el paso por todos los valores de color entre el primer color y el segundo

  Para que la tansición se aplique ya tiene que estar en el elemento original, no en el cambio en sí

    transition-property: cuál es la propiedad que se quiere transicionar.

  Usar el valor allí para esta propiedad es una mala práctica porque hace que el navegador trabaje mucho, se recomienda indicar cada propiedad en la que se quiere aplicar la transición separadas por comas

  Cuando cargamos la caja pasa del estado nativo del navegador al que nosotros le pedimos

  transition-duration: cuánto queremos que dure la transición

  transition-delay: cuánto tiempo va a tardar en comenzar la transición

  Hay que tener cuidado cuando se transiciona entre valores que no son matemáticamente compatibles, como auto a 20px por ejemplo

  Curvas de Bezier: es una ecuación matemática que permite definir una curva paramétrica, tiene cuatrao parámetros que controlan la dirección y la intensidad de la animación

  Material de referencia:

  cubic-bezier.com

  o en el inspector del navegador

  transition-timing-function:cambiar la aceleración

  ease: rápido a lento

  ease-in: arranca lento y termina abrupto

  ease-in-out: arranca lento y termina lento

  ease-out: muy lento, casi lenea y termina abrupto

  transition solo es un shorthand que reune las cuatro propiedades.

  En transiciones si quiero poner dos propiedades distintas con distinto valor se separan con comas

\*/

TEMA 52

/\*

  DESBORDAMIENTO U OVERFLOW

Determina cómo se maneja el contenido que se sale de su contenedor

Si no le doy una altura fija a un elemento al agregar texto se va ajustando automáticamente, pero si le doy un alto fijo el texto se sale del mismo (en este caso estamos hablando de un div), esto se maneja con la propiedad overflow

visible: es el valor por defecto y dice que lo que sobresalga del contenedor se vea, sea visible

hidden: oculta todo lo que se salga del contenedor. El navegador guarda el espacio de las barras de desplazamiento aún cuando no se vean

clip: no muestra las barras pero el navegador tampoco guarda el espacio para ellas. El contenido se recorta

scroll: muestra una barra de desplazamiento vertical que permite ver el texto o contenido oculto, pero también muestra una barra de desplazamiento horizontal que se ve horrible

auto: la barra de desplazamiento sólo aparece cuando hay desbordamiento, si no, no lo hace, el contenido se ajusta y sólo muestra el scroll si es necesario, de lo contrario no lo hace

overflow-y: con hidden oculta la barra vertical

overflow-x: con hidden oculta la barra horizontal

overflow-scroll: muestra la barra de desplazamiento vertical, el problema de esta opción es que si el contenido no desborda la barra se sigue viendo igual

Esto se aplica a todos tipo de elemento, imagenes, video, texto, etc

\*/

TEMA 53

/\*

  CONTROL DEL FLUJO DE TEXTO

  Control del texto

  white-space: define cómo se comportan los espacios en blanco

  - normal: es el valor por defecto

  - nowrap: indica que no hay saltos de línea, deja todo el texto en una sola línea. Es algo que no deberíamos usar

  - pre: detecta los saltos de línea tal cual nosotros los ponemos. Pero si una palabra se desborda no hace nada y deja que los haga

  pero el texto se desborda igual

  - pre-wrap: respeta los saltos de líneas puestos por el programador pero si una palabra se desborda la pasa abajo. Es una mezcla entre normal y pre

  text-overflow

  - clip: es la opción por defecto recorta el texto

  - ellipsis: junto con white-space: nowrap o pre y overflow: hidden se ve en el contenedor solo lo que entra y tres puntitos al final del texto. Sólo aparecen los tres puntitos cuando se desborde

  word-break

  - break-all: corta todas las palabras junto con white-space:normal

  - keep-all: evita la ruptura de palabras en algunos idiomas

  - auto-phrase: todavía es experimental, no salió, pero lo que hace es cuando una palabra se está por cortas busca crear una frase inteligente

  word-wrap:

  - break-word: separa en sílabas y si la palabra se puede romper lo hace y tira para abajo el resto

  - anywhere: las palabras largas las rompe si se desborda del contenedor

  - balance: balancea la distribución de palabras para que quede algo estético. Aplica a títulos, balancea el texto para que no queden palabras solas. Balance palabras

  text-wrap

  - pretty: evita las viudas y los huérfanos en los textos. Balancea párrafos

\*/

TEMA 54

/\*

  OBJECT FIT Y OBJECT POSITION

  Si cambiamos el tamñano de un elemento, este sigue teniendo un contenido

  Object Fit: determina como se comporta la imagen o video si le damos un alto y un ancho.

  - fill: propiedad por defecto, rellena el contenido y se adapta, no mantiene relación de aspecto.

  - contain: hace lo mismo que background-contain, hace que la imagen entre por lo menos una vez en el contenedor y que una de las dimensiones de la imagen entre al 100%

  - cover: lo único que hace es adaptarse llenando todo el espacio contenedor

  - none: mantiene la resolución original de la imagen. no toma en cuenta el alto y el ancho

  - scale-down: toma el valor de propiedad  más chica

  object-position: permite ubicar la imagen dentro del contenedor con top, bottom, left, right, pixeles.

  Esto no se aplica al fondo sino a las imágenes y los videos

\*/

TEMA 55

/\*

  OUTLINE

  Sabemos que el borde afecta al tamaño del elemento.

  Es una propiedad que se encuentra entre el borde y el margen Tiene sólo una utilidad visual, no ocupa espacio. Es sólo una guía para el usuario, si le cambio el grosor no cambia nada en la caja o el elemento

  Para que se vea tengo que definir las tres propiedades anteriores

  outline-width: define el ancho

  outline-color: define el color

  outline-style: define el estilo

  outline-offset: define la distancia de sí mismo hasta el borde, no está incluido en el shorhand, por lo que debe darse aparte

  outline: shorthand de esta propiedad, es igual que el borde: grosor estilo color

  Nunca se debe dar un outline invisible a los elementos que tienen que estar focuseados para evitar problemas de accesibilidad

  Para poder ocultar el outline tenemos que usar primero

  input:focus-visible{

    outline: none;

  }

  de esta manera no se muestra en pantalla, pero las herramientas de accesibilidad igual lo detectan

  No afecta al flujo de la página

  En transiciones si quiero poner dos propiedades distintas con distinto valor se separan con comas

\*/

TEMA 56

<!--

EMMET

Es un plugin que básicamente sirve para facilitarnos hacer ciertas cosas con HTML y CSS. Es específico para html y css

Funciona por abreviaturas, y lo que hace es convertir una abreviatura en todo un código

Cualquier elemento de HTML puede ser creado con emmet, es parecido a un snippet

SNIPPET

Es una forma de guardarun código específico para llamarlo desde el editor y poder usarlo sin escribirlo todo desde cero

Usos

nombre de la etiqueta enter crear al etiqueta de apertura y cierre para el elemento indicado

Si pongo lorem se pone un texto de relleno falso llamado lorem ipsum

elementopadre>elementohijo crear las dos etiqutas una dentro de la otra

elementopadre>elementohijo>elementonieto crea las tres etiquetas anidadas

elemento\*cantidad crea esa cantidad de equiquetas iguales. Esto se puede aplicar a los dos últimos items anteriores

elemento.clase agrega una clase al elemento indicado

Se pueden combinar las opciones

Ejemplo

li\*4>a

crea

<li><a href=""></a></li>

<li><a href=""></a></li>

<li><a href=""></a></li>

<li><a href=""></a></li>

Repaso:

El punto es para las clases

El > indica que dentro del elemento de la izquierda va el elemento de la derecha del símbolo

Elemento\*n indica que debe crear n cantidad de elementos

{texto $} indica qué texto es contenido y el $ crea un autoincrementable del 1 a la cantidad dada por n

elemento:atributo crea el elemento indicado con el atributo indicado

-->

<!-- ul>li -->

<!-- ul>li>a -->

<!-- ul>li\*4>a -->

<!-- ul>(li>a)\*4 -->

<!-- ul>li>a.link -->

<!-- ul>li>a#link -->

<!-- ul>(li.list-item>a#link)\*4 -->

<!-- ul.header-ul>li.list-item\*4>a.link -->

<!-- header>nav>ul.header-ul>li.list-item\*4>a.link -->

<!-- header>nav>ul.header-ul>li.list-item\*4>a.link -->

<!-- header>nav>ul.header-ul>li.list-item\*4>a.link{link} -->

<!-- header>nav>ul.header-ul>li.list-item\*4>a.link{link $} -->

/\*

  EMMET

  numero px

  numerop %

  nuemroe em

  m margin

  mnúmero indica margin 10px

  mnúmero-numero indica margin 10px 10px

  p padding

  bgc background-color

  c color

  w width

  c# da el color en hexadecimal

  Se pueden combinar

\*/

SECCIÓN 6 – CSS FLEXBOX

TEMA 57

/\*

  FLEXBOX

  Es un estilo de box model. Hay varios tipos de box model: en línea, en bloque, flexbox

  Es un módulo de CSS, qué hacemos cuando los contenidos de las cajas o los tamaños sean dinámicos, no los conozcamos o vayan cambiando? Necesitamos cajas con cosas dentro que se adapten al tamaño, necesitamos que las cajas sen flexibles y el contenido también

  A medida que cambia la resolución, el contenido va cambiando y se adapta, ubicándose de distintas maneras

  A una caja (caja contenedora) le vamos a dar la propiedad display flex y esa propiedad hace que todas las cajas que están adentro tengan toda la magia. Esta caja va a tener un comportamiento de bloque, pero lo que está dentro va a tener un comportamiento flexible, a ellos es a quienes se le van a aplicar las propiedades

  inline-flex permite tener dos cajas con los elementos de adentro flexibles

  Las cajas contenedoras son prácticamente un bloque, pero cuando le decimos flex, el contenido de adentro pasa a ser flexible, entonces no es a la caja grande a quien se le van a aplicar estas propiedades sino que es a las cajas que están adentro pero para hacer esto debemos darle propiedades a la caja y propiedades a su contenido

  Tenemos dos cosas:

  1. La caja contenedora: es el flexbos, la caja flexible

  2. Los flex items: son los elementos que están dentro de la caja contendora y es donde pasa la magie

  Una característica de flexbos es que es unidireccional, apunta en una sola dirección, y unidimensional porque tiene una sola dimensión, lo movemos en el eje x o en el eje y. Esa es la diferencia con grid que es bidireccional y bidimensional porque se mueve en el eje x y en el eje y

    Row: alinea los elementos en fila, horizontal, para nosotros de izquierda a derecha

    Row-reverse: invierte la dirección, sigue estando horizontal, en fila, pero va de derecha a izquierda, en dirección contraria

    Column: alinea los items de arriba hacia abajo, en forma vertical

  Column-reverse: invierte la dirección a de abajo hacia arriba, sigue estando en forma vertical

  Tenemos dos ejex:

  Main Axis:

  Con Row en nuestro caso va de izquierda a derecha, horizontalmente.

  Con Column va de arriba hacia abajo, verticalmente.

  Con Row reserse sigue horizontal pero cambia la dirección, pasa a ir de derecha a izquierda.

  Con column reverse sigue vertical, pero pasa a ir de abajo hacia arriba

    Cross Axis: sirve para alinear los elementos, para definir sus posiciones

  Con Row va de arriba hacia abajo, verticalmente. Cruza al eje principal.

  Con Column va de izquierda a derecha, horizontalmente.

  Con Row reserse sigue vertical y no cambia la dirección, queda de arriba hacia abajo

  Con column reverse sigue horizontal y no cambia la dirección, sigue de izquierda a derecha

  Dentro de la caja flexible voy a poder ubicar los elementos donde quiera

  gap: separa los elementos entre sí, se usa en lugar de margin

  Para que flexbox exista tenemos que tener una caja principal a la que le vamos a dar display flex y otras cajas que van a estar dentro de la principal que son los flex items. Algunas propiedades se le dan a la caja padre o flexbox y otras a las cajas hijas o flex items, estos últimos tienen que ser hijos directos

\*/

TEMA 58

/\*

  FLEX DIRECTION

  A la caja padre le damos un display flex y los elementos dentro de ella se alinean

  Los flex-ítems se pueden enconger pero no pueden crecer más alla del tamaño que se les dió en el width o en el height

  Todo lo que pasa con los elementos dentro de la caja se puede definir

  Si definimos a los elementos por su padding ya la caja deja de ser flexible

  Por defecto el alto del cross axis se ocupa por completo y si no le doy un width toma el 100% del ancho de la pantalla, alo alto van a ocupar todo el cross axis. Si usamos column van a ocupar todo el ancho. Para esto no hay que definir el width, si lo hacemos podemos hacer que los elementos se centren

  Propiedades;

  display: flex va en la caja padre la hace flexible para su contenido

  flex-direction: row es la propiedad por defecto del flexbox, los elementos se ordena en fila en la dirección del texto

  flex-direction:row-reverse va en la caja padre, e invierte la dirección de los ítems, van en contra de la dirección del texto

  flex-direction: column los elementos se acomodan de arriba hacia abajo y mantienen su width

  flex-direction: column-reverse los elementos se acomodan de abajo hacia arriba y mantienen su width

  direction: con row define la dirección del texto en la página, con column cambia la posición de los items en el cross axis solamente

  Nosotros lo vamos a ver left to right, pero los otros lenguajes no

  direction: ltr (left to right) o rtl (right to left) para escribir en la dirección del texto del idioma utilizado

\*/

TEMA 59

/\*

  FLEX WRAP y FLEX FLOW

  Flex wrap permite controlar el comportamiento de los flex items cuando el espacio del contenedor es insuficienta. Se le da a la caja contenedora, no a los hijos

  - nowrap valor por defecto, no se pueden bajar los flex items

  - wrap permite a los flex items pasar a la siguiente línea. Todos los elementos están alineados de una forma extraña

  - wrap reverse es lo mismo pero los elementos se ordenan al revés

  Flex flow permite definir dirección y wrap. No es muy usado

  flex grow indica cuanto puede crecer

    flex basis indica lo máximo que se puede recortar

\*/

TEMA 60

/\*

  ALINEACIÓN EN LOS EJES

  Estamos aplicando propiedades a la caja padre o caja flexible, no a las cajas hijo o flex items

  Propiedades:

  justify-content es una propiedad que habla del eje principal, lo que vamos a hacer es alinear el contenido en el eje principal, que va a depender si trabajamos row o column. Con esta propiedad lo único que hacemos e mover los flex items en el eje principal, pero mantiene la separación y otras propiedades

  - start: antes se usaba flex-start, pero ahora ya no es necesario usar flex delante del valor, este es el valor por defecto. Significa que va en dirección del texto

  - end: alinea al final del eje

  - center: centra los flex items

  Con esos tres valores lo único que hacemos e mover los flex items en el eje principal, pero mantiene la separación y otras propiedades. Pero hay tres más

  - space-between: pone el primer item al principio del eje, el último al final y los del medio van a  distribuir su espacio para quedar a la misma distancia uno del otro, los únicos que cambian si agrego más elementos son los del medio

  - space-around: hace que cada elemento tenga una separación igual, cada elemento tiene un margan izquierdo igual al margen derecho, los márgenes no se superponen y las distancias no son todas iguales. Los elementos de los extremos tiene una distancia al borde distinta de la que hay entre los elementos centrales

  - space-evenly: dice que va a haber la misma distencia entre todos los elementos, y el primero y el segundo van a tener la misma distancia que hay entre los elementos centrales

  Estos valores funcionan también con column sólo que se alinea verticalmente.

  Los reverse sólo cambian el orden de los elementos pero no la posición de ordenamiento

    align-items: alinea sobre el eje cruzado. Lo usamos cuando tenemos sólo una línea

  - stretch: estirate para llenar el corss axis completo, este es el valor por defecto. Los elementos van a crecer siempre en relación al ítem más alto

  - start: mantiene el height de los elementos, y los posiciona al comienzo

  - end: posiciona los elementos al final

  - center: centra los elementos

  - baseline: los ítems se alinea en la línea base del texto, se pone a la misma altura para que quede el texto relativamente alineado

  align-content: se alinea en el cross axis tambien. Sin esta propiedad es como que divide al contenedir en dos filas y el contenido se alinea en función de esas dos filas. La usamos cuando tenemos varias líneas aplican los mismos valores de justify-content

  Una técnica muy común y muy usada es usar el flexbox sólo para centrar un elemento vertical y horizontalmente con margin auto, esto centra el elemento en el centro del contenedor

  margin: auto solamente para centrar elementos en ambos ejes, poniendo una caja dentro de otro.

  También podemos usar

  align-content: center

  align-item: center

  así obtenemos lo mismo, el objeto centrado tanto en vertical como en horizontal

\*/

TEMA 61

/\*

  ORDENAMIENTO

  Organiza el orden visual, no modifica la posición en el DOM.

  El ordenamiento de los elementos se basa en su posición en el HTML o a través de su propiedad order

  Todos los elementos vienen por defecto con orden 0, si no lo definimos se queda en 0

  Acepta también números negtivos

  Se usa

  order: número de orden

  Esta propiedad sólo cambia el orden visual, no el orden real

  Esto sirve para por ejemplo en un movil se puede cambiar el orden en que se ven los elementos

  Podemos dar números negativos, ejemplo

  order: -1

  posiciona el objeto delante de los demás

  Las tecnologías accesibles reconocen el orden real y no gener problemas

\*/

TEMA 62

/\*

  FLEX BASIS, SHRINK Y GROW

  Flex items son los elementos hijos de una caja flexible

  Estas propiedades se aplican a los hijos no al contenedor o padre

  Flex items: hijos de un contenedor flex

  El problema que tenemos es que el tamaño de los hijos no se agranda

  Propiedades:

  flex-grow: por defecto viene en 0, dice que del espacio disponible puede ocupar algo. Como las cajas van a cambiar, ya no las afecta la alineación. Si se les da un max-width no va a funcionar el grow o va a funcionar hasta ese tamaño. Todo el espacio disponible lo va a ocupar. Es como si agarraramos el espacio disponible y lo rompiéramos en muchas partes, que van a depender de qué valor le demos a la propiedad grow

  El espacio disponible es el espacio que falta para que la caja padre se llene

  flex-shrink: determina la capacidad para encogerse. Por defecto su valor es 1. Determinamos cuánto se van a enconger los elementos

  Si le doy 0 los elementos no se van a encoger

  flex-basis: por defecto es 0, no necesitamos usar el width, con column es como min-height y en rów es como si usara min-width. Se usa flex-basis en lugar de width porque es el tamaño base, de donde parten los objetos. Da el tamaño mínimo que van a tener los hijos

  Si el tamaño base es más chico se van encoger si es más grande se van a estirar

  Estamos definiendo en la dirección que apunta el elemento el tamaño base que tiene este elemento

  La diferencia entre min-width y flex-basis es que uno establece el tamaño mínimo del objeto y el otro el tamaño base

  Lo recomendado es usar un flex-basis: 0 y también un min-width: (de algo)

  flex: define las tres propiedades juntas. flex:0 1 auto por defecto.

  Uso: flex: 1 1 100px. flex: grow shrink basis

  Recomiendo que usen flex y flex-flow

  gap: en el padre da la separación entre elementos

\*/

TEMA 63

/\*

  ALIGN SELF

  Esta propiedad se le da al hijo, porque es para agarrar un elemento puntual y llevarlo donde quiera

  Permite alinear a un hijo de forma diferente de la que se alineó

  Sólo funciona cuando usamos la propiedad align items. Es igual que align item pero sólo para un elemento.

  - center: es el valor por defecto

  Usa todos los valores de las otras propiedades de flexbox

  La propiedad por defecto es auto, que lo determine el navegador

  La única forma de que se estire es escribir stretch

  align-items aplica a todos los hijos, mientras que align-selt aplica sólo a uno de ellos

\*/

TEMA 64

/\*

  LAYOUT CON FLEXBOX

  Layout es básicamente la forma que tenemos nosotros de distribuir los elementos, el contenido para que entre dentro de nuestra página

  Si trabajamos con flexbox sin media query no se pueden cambiar ciertas características al cambiar de dispositivo

  Ser lo más específicos posibles, por ejemplo usar background-color en lugar de sólo background, para evitar conflictos

  Dar un height fijo al os elementos no se hace está mal y no es recomendado, es correcto usar min-height y no height solo

  Técnica del checkbox, es una especie de trampa, cuando está cliqueado el checkbox se abre el menú y cuando no se oculta

  Al h1 se le puede dar estilo poniendo el nombre de la etiqueta en lugar de asignar una clase porque va a se él único que tengamos

  Con flex solo no podemos cambiar ciertos estilo cuando el dispositivo se achica. Hay muchas limitaciones con este diseño

  No es recomendable usar height, ni width fijos, se usa min-width o min-height

\*/

SECCIÓN 7 – RESPONSIVE DESIGN

TEMA 65

/\*

BLOQUES FLEXIBLES

Vamos a ver como crear elementos que se vayan adaptando fluidamente o de forma flexible a las diferentes resoluciones de pantalla

Problemas a resolver y resolución

- El bloque no se adapta al tamaño de la pantalla, para solucionar esto usamos:

min-width: indica el menor ancho que el objeto puede alcanzar el objeto

max-width: indica el mayor ancho que puede alcanzar el objeto

Por defecto tiene 100% de la pantalla

De esta manera ya encontramos una forma de hacer que el objeto se adapte a la resolución del navegador o dispositivo. Cuando le damos un valor a cada uno le estamos diciendo que el 100% tiene que ser mayor que el min-width y menor que el max-width

- El texto no se adapta al bloque, no lo llena por completo o se deborda

Si no se llena y puedo omitir el height lo hago, no es correcto o ideal utilizar un height

Si se desborda la sugerencia salvo que sea muy estricto el uso del alto es utilizar

min-height: le da un alto mínimo al bloque y va a cumplir con el alto pedido, pero al agregar texto se va a ir extendiendo para contenerlo

Con estas dos soluciones tenemos contenido que se ajusta vertical y horizontalmente sin necesidad de utilizar flexbox

- Si queremos ponerlos en una línea usamos

display: inline-block

Pero el problema de esta solución es que aun cuando les doy min-widht y un width de 100% los bloque no se adaptan. Para solucionar esto se creo flexbox

En realidad permiten que se adapten moviendo abajo los bloque que no entren pero lo que quiero es que crezcan. Para esto usamos

En la caja contenedora

display: flex

flex-wrap: wrap

se adaptan pero seguimos teniendo el mismo problema de que no cambian su tamaño

Entonces usamos en el elemento hijo

flex-grow: 1

flex-basis: 0

Si no uso las dos propiedades no funciona

Esta es la forma correcta

También puedo usar

flex: 1

así aplica también al shrink

\*/

TEMA 66

/\*

MULTIMEDIA FLEXIBLE

Pensamos que trabajar con imágenes y con videos es igual a trabajar con otro tipo de contenido, pero presente otros problemas

Imagenes Flexibles

Son imágenes que se cambian para adaptarse al tamaño de la pantalla o a la resolución en la que se encuentran, también puede ser al contenedor

La forma más típica o clásica de hacerlo es

Cuando trabajamos con imágenes no está bien poner

width: 100%

porque si la imagen es pequeña se va a pixelar y ver mal, lo que hacemos es poner

max-width: 100%

con esto la imagen se achica, pero se agranda sólo hasta el valor indicado

Para que el height no nos juegue en contra ponemos

height: auto

y con estas dos propiedades tenemos todo lo que queríamos difinir.

La imagen se ajusta hasta el 100% del dispositivo que la está mostrando. Siempre se va a estar adaptando a una pantalla que sea mayor a él, pero cuando llega a la resolución de la imagen ya no crece más y no se pixelea

Videos flexibles

Si queremos hacer lo mismo con los videos que hicimos con las imágenes vamos a tener un probleam y hay una razón para ello y es por cómo los navegadores manejan el iframe y demás. Una imagen tiene un tamaño intrínseco, es decir le permiten a los navegadores redimensionarlos y que pueden mantener su relación de aspecto cuando cambiamos su tamaño. En los videos las cosas funcionan diferente y si bien podemos darle un width del 100% no se ajustan para mantener una relación de aspecto interesante, esta relación no la hacen, esto hace que el video se salga del contenedor o se deforme, lo que no está bueno

Por lo tanto esto que es lo que hicimos con la imagen no funciona

video {

  max-width: 100%;

}

Una forma de hacer es usando padding top hacking

Antes los que se hacía es tomar el contenedor del video y poner

.container {

  position: relative;

  max-width: 100%;

  padding-top: 56.25%;

  height: 0;

}

video {

  position: absolute;

  top: 0;

  left: 0;

  max-width: 100%;

}

La forma moderna es usar una relación de aspecto, de esta manera la relación se mantiene, porque lo que hicimos fue adaptar el contenedor

\*/

TEMA 67

<!--

  ATRIBUTOS PICTURE, SOURCE Y MEDIA

  Esto no va a permitir entender un poco más como trabajar el tema de decirle al navegador mirá te doy estas opciones vos elegí la más óptima, la más adecuada

  PICTURE

  Sirve para especificar múltiples fuentes para una imagen, acá dentro vas tener una imagen, te voy a definir una especie de contexto que va a servir de fuente para que vos elijas y determines cual buscar, vas a mostrar una nada más. Dentro dela etiqueta Picture siempre va la etiqueta Source, esta etiqueta se puede usar en más situaciones pero esa vez la vamos a usar con picture.

  Tiene significado semántico más acertado, porque le decimos al navegador acá te doy varias fuente y vos hacé lo que quieras.

  Va a mostrar una de las posibilidades

  SOURCE

  Usamos una etiqueta source por cada imagen alternativa que queramos utilizar. Pero también se necesita la etiqueta img

  Siempre va a cargar la imagen que se indique en img

  Esta etiqueta se usa dentro de las etiquetas picture, audio y video principalmente y lo que hacemos es definir estos diferentes medios para distintas situacionees

  Le tenemos que poner también media="(la resolución en px)" con las imagenes alternativas ordenadas de menor a mayor

  MEDIA

  Nos permite dar la resolución al navegador para que cambie el tamaño de una imagen o seleccione entre varias cual va a utilizar.

  TYPE

  Permite indicar el tipo de imagen que se va a estar usando. Carga el recurso según su tipo

  Las imagenes webp tienen una características que hacen que las imágenes carguen mejor, es una propuesta de google

  Picture lo usamos cuando necesitamos cargar una imagen muy grande, vamos dando opciones para optimizar las opciones de carga

  Sugerencia leer la documentación

-->

TEMA 68

/\*

  MEDIA QUERIES

  Es una herramienta poderosísima que nos permite adapatar nuestra página web a cualquier resolución. Paa esto lo que hacemos es definir estilos distintos para cada resolución

  Le permiten al desarrollador usar la lógica de si... entonces... determinado estilo se aplica solamente si el dispositivo de visualización cumple ciertas condiciones

  Sintaxis

  @media not|only mediatype and (expression) | CSS-Code;

  @media es la regla que se aplica

  not evita que se aplique el estilo en los dispositivos que cumplen la condición

  only es para evitar que navegadores antiguos que no soportan el estilo apliquen la regla

  mediatype aplica al tipo de medio al que se le aplica la consulta

  and combinar varias expresiiones en la consulta

  expresiones determita las propiedades que quiero usar

  SE USA

  Abriendo un bloque para el media query y dentro definimos los estilos

  Ejemplo

  @media type {

    selector {

      propiedades: valores

    }

  }

  print se aplica sólo a los medios de impresión

  pt es una unidad de impresión

  screen se aplica solamente a  pantallas

  Esto es lo que hace posible el 99% de los layouts de hoy en día

\*/

TEMA 69

<!--

  EJERCICIO HOLY GRAIL CON FLEXBOX

  wrapper es un contenedor con la finalidad de darle un estilo específico

  Estructura

  div main-wrapper para deleimitar el parte central del diseño

  Dentro van un aside, un main y otro aside

  No es correcto poner los aside dentro del maim si no forman parte del contenido principal

-->/\*

  EJERCICIO "HOLY GRAIL" CON FLEXBOX

  Patrón de diseño, histórico.

  Vamos a empezar a trabajar con el layout, para ener preparado todo para el contenido

  El Saint Grail Layout se puede componer de tres etapas

  1. La etapa principal que es el header con un menú de navegación

  2. Después tenemos la parte principal, el main content y dos columnas

  3. En la parte de abajo el footer

  Estas cinco secciones componen el Saint Grail

  Son tres secciones donde la de arriba y la de abajo están enteras y la del medio está dividida

  En el main section el contenido va a estar siempre dependiendo del contenido del otro

  Cuando el contenido es muy grande lo recomendado es que se achique

  para que sirve dar al main wrapper un max-width de 1024 px?

  Es por un tema de legibilidad, hay que darle a usuario una sección donde leer sin estorbos

\*/

TEMA 70

/\*

  MOBILE FIRST

  Es un modo de pensar el desarrollo web donde primero desarrollamos una página adaptada a dispositivos móbilesy de dispositivos móviles la vamos adaptando a dispositivos más gandes. En el media query usamos min-width

  El otro es Desktop first creamos primero la página para pc y luego la llevamos a dispositivos mobiles. En el media query usamos max-width

  Esto depende del consumo, si vamos a crear una página para consumo general, no de nicho, si no por ejemplo una tienda virtual o una red social, lo hacemos mobile firts porque la gente principalmente va a entrar desde el celular.

    Aquí tenemos el desafío de mostrar la mayor cantidad de infomación en la menor cantidad de contenido posible y en el menor tiempo posibie, en una pantalla chiquitita, lo que hace que nuestras páginas sean más minimalistas, más simples y también ver como gestionamos el contenido para resoluciones chiquititas.

  Si bien mobile first es el principal enfoque hoy en día tenemos que tomar en cuenta al usuario, si es para una plataforma para médicos por ejemplo, vamos a usar desktop first, ya que ellos no van a trabajar con celulares sino con computadoras en su escritorio

  Tenemos que ver a quien va dirigido y el próposito de la página

\*/

TEMA 71

/\*

  FEATURE QUERIES

  Selos conoce también como Media Queries de Nivel 4 porque son de CSS4 que si bien no salió, sí CSS4 va a salir.

  Básicamente permite a los desarrolladores a escribir código en caso de que los navegadores soporten una funcionalidad

  Para qué sirve para hacer cosas en caso de que toda una propiedad no se soporta

  Ejemplo si no soporta grid que aplique flexbox y si lo soporta que aplique grid

  En cada @support trabajo los estilos que quiero que se apliquen

\*/

TEMA 72

/\*

  CONTAINER QUERIES

  Representan un avance significativo en el diseño

  Se puede agarrar un contenedor y según las medidas o propiedades de ese contenedor dar estilo a lo de adentro, es realmente un avance

  container-type define el tipo de característica que se va a tomar en cuenta

  - size tiene en cuenta las dimensiones en bloque y en línea, las dos

  - inline-size indica que según el ancho del contenedor se aplicará el estilo, sólo tiene en cuenta las dimensiones en línea

  - normal hace que el elemento no pueda considerarse un query contaier

  SE USA

  Primero se define el contenedor

  .container {

  container-type: inline-size;

  }

  Se usa esa definición como media query

  @container (max-width: 500px){

  p{

    font-size: 20px;

  }

  }

  De esta manera se va a aplicar a todos los contenedores

  Para aplicarlo a uno específico creo un nombre con container-name

  Así creamos un container context y damos un nombre a ese container

  Permite modularizar y más control, se pueden reutilizar componente

  No es compatible en todos los navegadores, en proyectos grandes se puede hacer más complejo el código

\*/

SECCIÓN 8 - CSSGRID

TEMA 73

/\*

  INTRODUCCIÓN A GRID

  Es un sistema de diseño bidmensional que lo que hace es permitirnos a nosotros los desarrolladores crear layouts enteros complejos de forma fácil y con cosas flexibles

  En una grilla vamos a trabajar no en una sola dirección si no en dos, nos va a permitir trabajar con filas y columnas, no vamos a necesitar elegir con cual queremos trabajar

  Grid se creo para superar limitaciones anteriores de cosas que no se podían hacer y ahora sí se puede hacer, viene a superar esa barrera de trabajar en una sola dirección y trabajar en dos direcciónes

  CUÁNDO TRABAJO CON FLEXBOX Y CUANDO TRABAJO CON GRID?

  GRID cuando vamos atrabajar con contenido que va a cambiar o en dos direcciones

  FLEXBOX cuando necesito trabajar diseños en filas o en columnas, en una sola dirección

  CONCEPTOS BÁSICO

  - Grid Container: básicamente e somo un contenedor padre, el contenedor cuadrícula, se define con

  display: grid

  - Grid Items: son los elementos de la cuadrícula, que son los hijos directos del contenedor de la cuadrícula, cada uno de estos elementos pueden ocupar uno o varios lugares de la cuadrícula. Es el elemento puntual html que se coloca en la cuadrícula

  - Líneas de la Cuadrícula: son los bordes de la cuadrícula y separan las filas de las columnas, estás líneas se pueden referencias para crear los elementos de la cuadrícula

  - Grid Tracks: cada línea de la cuadrícula es un track, puede ser una fila o una columna

  - Grid Cell: son los espacios entre cuatro líneas adyascentes. Es la unidad básica donde se pueden ubicar los elementos. Es el espacio real que grid reserva para colocar el elemento puntual. Nosotros podemos definir una grid y no poner ningún elemento y aún así están las grid cells

  - Grid Area: son espacios de grid delimitados por cuatro line grids, pueden tener uno o más celdas y se usan para ubicar de manera más eficiente los elementos en la página. Podemos decirle a un elemento que ocupe un área completa

\*/

TEMA 74

/\*

  CREANDO UNA GRID

Lo primero que hacemos es crear un contexto en el que podemos trabajar con filas y columnas. Creamos un entorno en el que le decimos al navegador que vamos a trabajar con filas y columnas

  Si definimos una grid y no ponemos nada cada hijo se va a convertir en una fila implícita

  Cuando usamos grid los no hay colpaso de márgenes y sí se van a sumar los dos márgenes de las dos filas o columnas adyascentes

  Si aplico display grid al contenedor los elementos siguen ubicandose uno al lado del otro sin importar que a cada item le puse un display inline-bloc, que deberí formar dos columnas al cambiarles también el ancho.

  Cuando asignamos display grid se crea una grilla con una sola columna y cada hijo formando una fila que ocupa una celda

  diferente

  Tanto flexbox como grid manejan sus ejes principales en el sentido de la escitura

  PROPIEDADES

  - grid-template-columns: me pide que indique la medida de la columna, sólo me pide medidas. Ejemplo si le digo 200px, esto es lo que va a medir cada columna, si no le ponemos nada le estamos diciendo que la columna va a ocupar todo lo que ocupe la grilla, para indicar cuantas columnas tengo que poner una medida al lado de la otra. También podemos usar porcentajes, esto va a hacer que esa celda o celdas crezca, haciéndola dinámica. Podemos usar todas las unidades que vimos. Define el width de la columnas

  - grid-template-rows: permite definir el tamaño y cantidad de filas. Si uso porcentajes es para el alto de la celda o del alto de la celda. Si definimos sólo una fila, la otra se adapta al contenido.   Aunque algunos espacios no estén utilizados, el espacio se guarda si defino una fila. Define el height de la filas

\*/

TEMA 75

/\*

  UNIDADES AUTO Y FR

  - Auto: la celda se estira para ocupar todo el espacio. Una cosa es el tamaño de la columna y otra muy distinta es el tamaño del hijo, el tamaño reservado es el de la celda, el tamaño del item puede no ocupar toda esa celda. El esapcio disponible se calcula por el cntenido de las cajas. Si le pongo auto a más de una columna, el espacio disponible es el que queda luego de quitale el espacio que ocupa el contenido de hijo y ese espacio lo reparten entre las columnas que tengan auto. Se calcula el min-content de las cajas y el espacio de la grid que sobre se distribuye entre las columnas a las que le dimos auto en grid-template-columns. Primer se asegura que el contenido de todos las cajas entre y lo que sobra es lo que reparte entre las que tienen auto

  En rows funciona igual que en columns

  - fr: si le pongo 1fr, ocupa todo el ancho disponible, si se lo pongo a todas las columnas, ya no le importa más el contenido. No se aplica restando el min-content, sino que se aplica tomando en cuenta el width total. Si el contenido ocupa todo el ancho de la celda entonces está se estira, porque el valor dado se toma cómo el mínimo. Representa una fracción de espacio de la grid y actúa como flex-grow

  Mientras auto quita el contenido mínimo, 1fr no lo hace.

  fr se usa mucho más que auto porque divide a todas por igual y nos da una visión más certera de las proporciones que está tomando.

\*/

TEMA 76

/\*

  REPEAT() Y MINMAX()

  REPEAT() nos sirve para evitar repetir siempre la misma expresión o las mismas expresiones

  repeat(cantidadveces, expresión)

  Repite sólo lo que se encuenra entre los paréntesis, puedo poner otro valor delante o detrás del repeat

  También se pueden repetir patrones

  repeat(3, 1fr 2fr)

  Esta es una forma interesante de ir cambiando varios valores de una sola vez

  MINMAX() es para darle un mínimo o un máximo a una columna. Va a decir toda la columna va a tener un mínimo de... y un máximo de...

  SE USA

  minmax(valormínimo, valormaximo)

  Es como usar min-width y max-width pero en lugar de ponerselo a un elemento se lo ponemos a toda la columna

  Se pueden combinar estos dos valores

  \*/

TEMA 77

/\*

  GRID IMPLÍCITO Y EXPLÍCITO

  GRID EXPLÍCITO es aquello que nosotros ya definimos, ya le dimos los tamaños y lo definimos explícitamente

  GRID IMPLÍCITO grid trata a todos los elementos como parte de su grilla, si nosotros estamos creando elementos que son parte de la grilla, grilla tienen que hacer algo con esos elementos, lo que va a hacer es agregar filas, las columnas ya las definimos pero no dimos un alto a esas filas, lo que hace es que se adapte al total del contenido y a la grilla

  - grid-auto-rows le dice a la grilla el alto que deben tener todas las filas que se creen automáticamente

  Lo único que se crea automáticamente son las filas, las columnas no, si definimos al cantidad, se va a mantener

  Se puede cambiar la dirección de los ejes para que en lugar de que se creen filas, se creen columnas

  - grid-auto-flow: columns va a permitir que se creen columnas automáticamente en lugar de fila. ESta propiedad dice ahora vas a seguir el flujo de columans

  Por defecto viene en row

  - dense sirve para rellenar la cuadrícula, es muy usado,por ejemplo para crear galería de imagenes, ya lo vamos a ver

  - grid-auto-columns va a modificar el ancho de esas columnas que se crean automáticamente

  Ahora cada nuevo elemento que agregamos en lugar de ponerse en fila se pone en columna

\*/

TEMA 78

/\*

  GRID GAP

  Es la separación entre elementos

  - gap esta es la forma correcta de crear separaciones entre elementos porque solo nos separa los elementos entre sí, no con el borde del contenedor ni nada así. Este es un shorthand para las dos siguientes

  SE USA

  gap: row column

  - column-gap separa las columnas

  - row-gap separa las filas

  Para que también se separe del borde del contenedor se le da al padre un padding igual al gap

  No es común utilizar el padding, es raro, pero si tiene sentido lo pueden utilizar, el padding sólo lo utilizaríamos si queremos que resalte un borde, por un tema estético

  Cuantas menos media queries, cuantas menos capas de complejidad  tengan que usar es mejor

\*/

TEMA 79

/\*

  GRID DINÁMICO (Y RESPONSIVE)

  GRID ESTÁTICA lo que vimos hasta ahora es un grid estático porque no importa lo que digamos o hagamos no se va a ajustar a distintas medidas y tamaños. Mantienen siempre la misma cantidad de columnas, la misma cantidad de filar

  GRID DINÁMICA sin impotar el tamaño se va modificando

  Con el repeat tenemos dos valores muy importantes

  - auto-fit ajusta os elementos. Con grid-auto-flow en row hace una especie de wrap y va subiendo o bajando los elementos dentro de la grid, se adapta, y hasta acá no hay diferencia con auto-fill, en los dos casos si le damos un solo valor está bien, la forma de funcionar es igual. Cuando usamos fr en el minmax es donde se crean y descrean columnas cuando el espacio alcanza para hacerlo.

  - auto-fill literalmente rellena. Cuando hay muchos elementos no hay diferencia entre los dos auto-fit y auto-fil. La diferencia varía cuando hay pocos elementos. Esta opción crea columnas para ir rellenando el espacio aunque no haya elementos, caso en el que deja el espacio reservado en blanco. Mientras auto-fill rellena, auto-fit adapta los elementos

  Usar auto-fit puede servir para centrar contenido de elementos en la página

  Usar auto-fill para trabajar con imágenes o resultados de búsquedas

\*/

TEMA 80

/\*

  GRID COLUMN Y GRID ROW

  Las líneas (grid lines) que se crean es las que se le indican más una porque, igual con las columnas, si tenemos cuatro hay cinco

  En inspeccionar -> elementos -> grid -> activamos la opción grid en un circulito y muestra el número de línea

  Esto determina las posiciones,

  - grid-column-start: número

  - grid-row-start: número

  el número indica donde comienza el objeto o celda, o a cual me estoy refiriendo, tanto en fila como en columna

  - grid-column-end:

  número o span número

  - grid-row-end:

  número o span número

  En ambos caso número indica donde termina el elemento, en qué línea termina el elemento o celda

  También lo pueden resolver con object-fit: cover

  height: 100%

  Tenemos problemas de adaptación sólo con las imágenes, si usamos un div eso no pasa, entoences no debemos poner una imágen directamente sino dentro de un contenedor div y ajustar la imagen a él.

  Con el div el resto de la grilla no se ve afectada

  La imagen en el contenedor debe ponerse como background image

  Crear un contenedo y poner la imagen dentro o como fondo

  span número indica cuántos espacios se va a expandir el elemento, tanto en filas como en columnas. Se usa para no tener que cambiar todos los valores, cambiando sólo uno o dos es suficiente para desplazar el elemento dentro de la grilla

  MEJOR FORMA DE HACERLO

  Usar sólo

  grid-column: start/end

  grid-row: start/end

\*/

TEMA 81

/\*

  GRID FLOW DENSE

  Al usar span para grid column o grid row podemos tener problemas de que que rompa el layout al achicar o agrandar el dispositivo

  Cuando usamos

  grid-flow: dense

  Se rellenan los espacios en blanco

  No respeta el orden de los elementos, dice estos huecos vacíos hay que rellenarlos de alguna forma

  Esta es la mejor solución, en realidad es la única solución nativa para resolver el problema de que se rompa la página

\*/

TEMA 82

/\*

  GRID AREA

  Nos permite crear un layout teórico donde le decimos al navegador en cuentas filas queremos dividir cada fila o darle un nombre que después le podemos reemplazar por un grid. Podemos asignar un elemento a un área específica que le dimos en grid

  USO

  grid-area rowstart/rowend/columnstart/columnend

  Ejemplo

  grid-area: span 2 / span 3

  esto reemplaza a

  grid-column: span 3;

  grid-row: span 2;

  Ejemplo

    grid-template-areas: "header header header" "nav main aside" "footer footer footer" ;

  Tiene que haber una coincidencia, tiene que haber tres de cada uno

  Delimita las columnas que van a ser usadas y la estructura de la grid

  grid-area: header

  por ejemplo indica el elemento que voy a posicionar

  grid-template-areas define las áreas o secciones del layout

  grid-area asigna cada área o sección a los elementos que van a estar allí

  min-content es una forma de hacer que los elementos sean tan pequeños como su contenido lo permita, sin causar desbordamiento, y es especialmente útil en diseños responsivos y cuadrículas.

\*/

TEMA 83

/\*

 ALINEACIÓN CON GRID

  Con grid no tenemos que preocuparnos por el main axis y esas cosas porque podmeos decirle la dirección como eje horizontal o eje vertical

  FORMAS DE CENTRAR ELEMENTOS

  1. Usando flex en los grid-item

  display: flex

  justify-content: center

  align-items: center

  Esta es una forma válida de centrar elementos pero vamos a ver como centrarlos con grid porque es algo muy poderoso y sencillo de hacer

    2. Con grid

  Propiedades

  justify-items es la más compleja de todas. Es como align-itmes de flexbox, pero en lugar de actuar sobre el eje vertical, lo hace sobre el eje horizontal. En grid no se puede cambiar el flujo tan fácilmente.

  - start

  - self-start

  - left

  Aparentemente las tres hacen lo mismo, pero no. Si cambiamos grid-flow no se afectan los valores

  - center

  - right

  - end

  - self-end

  Las últimas tres también parecen hacer lo mismo pero no

  - baseline dice vamos a ponerlo sobre la línea de texto

  - first-baseline lo pone sobre la primera linea de texto

  - last-baseline lo pone sobre la última linea de texto

    center, start, end, significa en la dirección del texto,

  si usamos direction rtl cambia la distribución de los elementos

  vas a ir como vaya el texto. Esto es como en flexbox, no hay nada nuevo

  La principal diferencia es que en grid start tiene en cuenta la dirección del contenedor y self-start la dirección del hijo, dirección del texto.

  right siempre alinea a la derecha

  left siempre alinea a la izquierda.

  Siempre alinean el contenido dentro de su propia celda

  stretch se estira completamente y es el valor por defecto

  los self toman en cuenta la dirección del hijo, son para mover el elemento hijo dentro del elemento padre de forma independiente del resto de los hijos o todos juntos. Es start solo tiene en cuenta la dirección del contenedor

    justify-content tenemos las mismas propiedades que con flex. Movemos las celdas de forma horizontal dentro del contenedor, justificamos la celda dentro del su contenedor

  - start valor por defecto, funciona con ltr o rtl en función de la dirección del texto

  - end

  - center

  - space-around

  - space-between

  - space-evenly

  Por defecto trae start

  - right

  - left

  En estos dos casos no importa la dirección del texto siempre van a estar a la derecha o a la izquierda y no importa el direction que tengan

  align-items hace lo mismo que justify items pero en el eje vertical. Usa los mismos valores que justify-content. Funciona muy parecido a flexbox pero con bidimensonalidad

  justify-self self se refiere a sí mismo, entonces vamos a autojustificarnos a nosotros mismos (el hijo)

  Los elementos hijos están dentro de una celda con justify-items justifica los elementos en el eje horizontal dentro de su propia celda

  align-self alinea los hijos dentro de la celda individualmente, actúa como justify-self pero en vertical

  align-content alinea las celdas dentro de su contenedor verticalmente

\*/

TEMA 84

/\*

  SUBGRID

  Sirve para hacer que una grid dentro de un grid-item forme parte también de la grid principal

  Supongamos que un element se tiene que posicionar en el medio y que tiene que tener otro grid

  El ejercico de ejemplo se usa mucho

  Al usar subgrid en el hijo que la contiene como filas y columnas, esta se adapta a las celdas que debería ocupar en la grid principal. Puedo cambiar su posición sin desarmar o romper la grid principal

  Puedo cambiar el orden de los subitems, y su tamaño, y va a afectar a los elementos de la grid principal

Para crear una subgrid usamos como padre y definimos el display grid en el elemento hijo que al que le vamos a asignar la subgrid, y le asignamos las mismas configuraciones que al elemento que funciona como grid principal

\*/

TEMA 85

/\*

  CREANDO UNA PÁGINA WEB

  Este cuenta como proyecto final

  Tenemos la complejidad por la cantidad de código y por el código en sí, que vamos a estar trabajando

  Es importante al emprender un proyecto grande como corregir errores sin afectar al resto del código y que tampoco sea tan complicado encontrarlos

  Sistema a usar Mobile First

    PASOS A SEGUIR EN UN PROYECTO REAL

  1. Crear el wireframe

  son como bocetos de los que queremos hacer, hay uno para como se debería ver en dispositivos móviles y otro para como se debería ver en computadora. Vamos a partir de dispositivos móviles y agrandarlo para computadores. Esto está bueno porque requiere menos código, tenemos que concentrarnos en el contenido más relevante

  Carpeta con

  index.html

  styles.css

  logo

  imagen

  2. Elegir la paleta de colores

  #4488ee

  #e2eaff fondo body

  #18233f

  #4988e9 logo

  Cargamos la imagen para ver que está bien

  3. Preparar los archivos a utilizar

  El color de fondo no puede ser blanco puro ni negro puto ni colores muy intensos o brillantes porque hace que la página parezca trucha

  4. Trabajamos el header con el navbar y el menú

  Vamos a usar container queries

  Hay que tener cuidado con las especificidades a la hora de  asignar las propiedades y ver donde comviene más por recursos y por especificidad

  5. Damos estilo a la profile section

  place-content: center, sirve para centrar el contenido en una grid

  Hay cosas que van a ir aprendiendo que van a tener que ir modificando, ya no se hace más overlay si no que usamos

  background-image: gradient

  Recuerden que background-image es para usar imágenes o gradientes

  Se pone primero el gradiente y luego la imagen para que la segunda no tape al gradiente

  scroll-behavior: smooth hace que el desplazamiento sea más lento

  Tarea dar a la caja del - form un estilo acorde al profile

  - agregar un sticky a la navbar

  6. Trabajamos el portfolio

  7. Trabajamos el contact form

  8. Trabajamos el footer

  \*/ <!--

  CREANDO UNA PÁGINA WEB

  1. Creamos el header

  2. Creamos el main con un div clase wrapper, el wrapper se usa como un contenedor que nos deja delimitar el espacio y tamaño del contenido

  Normalmente se usa un id en lugar de clases para armar bloques completos

  3. Creamos un Section para el Portfolio

  4. Creamos otro section para el form de contacto

  5. Creamos el footer

-->

SECCIÓN 9 – ANIMACIONES

TEMA 86

/\*

  TRANSICIONES REPASO

  Con las dos líneas de código siguientes vamos a poder hacer que en un estado determinada una transición tenga un tiempo de duración distenta a cuando ese estado deja de existir

  En el elemento: transition ponemos width 2s linear;

  y en el estado ponemos transition-duration: 1s;

  steps(cantidad de pasos) permite que la animación sea segmentada en la cantidad de pasos indicada

  PARA PERSONAS EPILÉPTICA O CON PROBLEMAS NEUROLÓGICOS

  Google Chrome tiene una configuración que permite reducir las animaciones

  @media (prefers-reduced-motion:reduce){

    aquí podemos hacer que las animaciones o transiciones sean más lentas o directamente que se desactiven

  }

  Hay que tener cuidado para que los movimientos no sean tan bruscos

\*/

TEMA 87

/\*

  ANIMACIONES

  Proceso que quiere generar una ilusión de movimiento, tomar un objeto inanimado y hacerlo animado. Es como una transición entre un estado inicial y un estado final y en el medio pueden pasar más cosas. Tomar un elemento y hacerlo pasar de un estado a otro a través del tiempo.

  Se usa para mejorar la interrelación con el usuario, la gente de UX lo usa para mejorar la experiencia y también en cargas que tardan para mejorar la espera

  Transiciones ocurre sólo cuando hay un cambio de estado, necesitan un evento disperador que diga que están en un estado y ahora pasaron a otro estado. La transición sólo se repite una vez y es de un estado a otro

  ESTRUCTURA

  @keyframes nombre{

    from{

      propiedad: valor

    }

    to{

    propiedad: valor

    }

  }

  En lugar de from y to se pueden usar porcentajes, también se pueden agregar porcentajes o puntos intermedios de animación

  Las animaciones las podemos ejecutar cuando queramos, incluso pueden ser automáticas, podemos cambiar más veces las propiedades, controlar los cambios. También se pueden repetir todas las veces que queramos, incluso infinitamente

  PROPIEDADES

  - animation-name: nombre dado a la animación

  - animation-duration: tiempo que va a durar la animación

  Estas dos propiedades tienen que estar sí o sí en el elemento al que se le aplica la animación

  - animation-delay es el tiempo que tarda en comenzar a ejecutarse la animación. Se puede dar números negativos, al hacer esto es como si la animación hubiera arrancado hace el tiempo indicado en negativo. El dilay por defecto es 0, arranca en 0 y termina en el valor de la duración-1. Se aplica siempre sólo la primera vez que carga o se ejecuta la la animación

  - animation-fill-mode dice cómo va a quedar la animación una vez termine de ejecutarse

    backwards se queda como estaba antes de la animación

    none es como backwards se queda como antes de la animación

    both se queda como está al final de la animación y arranca como le indique la propiedad from. Arranca con el from dela nimación duante el delay incluido

    backwards espera el dilay como estaba y arranca con las propiedades de from

  - animation-timing-function indica el flujo de la animación

    linear toda la animación es pareja

    por defecto arranca rápido y termina más lento

  - animation-iteration-count determina cuantas veces se va a repetir la animación

    infinite repite la animación siempre

    por defecto viene 1

    puedo dar la cantidad específica que quiero

  - animation-direction indica hacia donde se dirige la animación

    reverse del final al principio o de atrás para adelante

    normal es la opción por defecto

    alternate va primero de adelante para atrás y luego de atrás para adelante.

    alternate-reverse va de atras hacia adelante y de adelante hacia atrás

  - animation-play-state permite que se pueda manipular la animación

    paused detiene la animación mientras haya un estado específico

    running es la opción por defecto que mantiene la animación ejecutándose

\*/

TEMA 88

/\*

  BOTONES ANIMADOS

  ::After y ::before son muy usados para hacer animaciones y es muy poderoso

  ::before tiene que tener la propiedad content sí o sí aunque esté vacía para que funcione

Desafío

1. Etiquetarlo en una historia de Instagram con el mejor botón que hayan logrado crear
2. Limpiar el código para que sea menor
3. Hacer los botones más estéticos
4. Tener la capacidad de ver los botones y decir que puedo cambiar

\*/

TEMA 89

/\*

  EFECTO TYPEWRITER

  Desafío

1. Crear una forma de subir el contenedor para que desaparezca cuando termina la animación
2. Crear un efecto para carga de página

\*/

TEMA 90

/\*

  ANIMACIONES BASADAS EN SCROLL

  Hasta ahora lo único que tenemos es una línea de tiempo basada en la duración. El timeline va a estar basada en la duración

  En las animaciones basadas en scroll tenemos dos tipos de timeline

  - Scroll: Uno basado en scroll que básicamente sólo funciona por el scroll del elmento. La animación va a progresar según progrese la barra de scroll del objeto padre

  - View: Hay otra opción que es por el view, cuando el elemento empieza a aparecer en el viewport comienza la animación y cuando el elemento se sale del viewport termina la animación

  animation-iteration-count, sólo sirve cuando estamos trabajando animaciones basadas en la duración

  direction y delay sí se usan pero lo vamos a sacar por ahora

  ANIMACIÓN POR SCROLL

  Hay dos forma de hacer la animación por scroll

  1. Por desplazamiento del scroll: La más básica es darle a la caja que tiene la animación la propiedad animation-timeline con la función scroll() como valor

  En body agregamos la propiedad scroll-timeline-name y le damos un nombre como valor con el formato --nombre, con esto le damos un nombre a las barras de scrill

  scroll-timeline-axis para indicar cual de las dos barras se toma en cuenta

  - inline en el eje horizontal

  - block en el eje vertical

  scroll-timeline es el shorhand de las dos propiedades anteriores

  scroll-timeline: --name axis

  En el body no podemos usar estar propiedad, esta debe ser aplicada a cualquier elemento que no sea el body

  Para que la animación funcione con el scroll del root debemos usar la función scroll() como valor de la propiedade animation-timeline. Esta opción es la más utilizada

  Si queremos que no se base en el root, debemos crear otro contenedor al que le vamos a poner un nombre para la propiedad animation-timeline-name

\*/ /\*

  ANIMACIÓN POR EL VIEW

  No depende del scroll, depende de si el elemento es o no visible

  view-timeline dice al navegador cual es el elemento que debe ser observaro, es decir se pone en el elemento que se va a animar y lleva el valor --nombre

  Le damos un nombre cuando queremos que la animación dependa de si otro elemento se está viendo

  En la animación por scroll el timeline se definía en el contenedor porque dependía del scroll del contenedor

\*/

TEMA 91

Scroll-driven-animations.style/tools/view-timeline/ranges/#range-start-name=contain&range-sta

https://scroll-driven-animations.style/tools/view-timeline/ranges/#range-start-name=contain&range-start-percentage=0&range-end-name=cover&range-end-percentage=100&view-timeline-axis=block&view-timeline-inset=0&subject-size=smaller&subject-animation=reveal&interactivity=clicktodrag&show-areas=yes&show-fromto=yes&show-labels=yes

BÚSQUEDAS EN COPILOT

Para vincular un botón con un formulario en HTML, puedes usar el atributo form del elemento <button>. Este atributo permite asociar el botón a un formulario específico mediante el ID del formulario, incluso si el botón no está dentro del formulario. Aquí tienes un ejemplo:

<!DOCTYPE html>

<html lang="es">

<head>

<meta charset="UTF-8">

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

<title>Vincular Botón con Formulario</title>

</head>

<body>

<!-- Formulario -->

<form id="miFormulario" action="/enviar" method="post">

<label for="nombre">Nombre:</label>

<input type="text" id="nombre" name="nombre" required>

</form>

<!-- Botón fuera del formulario -->

<button type="submit" form="miFormulario">Enviar</button>

</body>

</html>

[En este ejemplo, el formulario tiene el ID miFormulario, y el botón utiliza el atributo form="miFormulario" para vincularse a ese formulario1](https://lineadecodigo.com/html5/asociar-un-boton-a-cualquier-formulario-con-html5/). Al hacer clic en el botón, se enviarán los datos del formulario.