HTML no es un lenguaje de programación, es un lenguaje de marcado que define la estructura de tu contenido y se basa en etiquetas o elementos o también conocidos como tagname.



También podemos agregar atributos, ejemplo



Acá estamos agregando el atributo “class” y dentro del atributo le ponemos el nombre que tendrá la etiqueta.

Los archivos html tienen que tener un nombre y la extensión (prueba.html, ejemplo.html, hola\_a\_todos.html), los nombre no deben tener espacios ya que al momento de subirlos a un servidor nos traerá problemas, para reemplazar el espacio utilizamos guion o guion bajo, evitar mayúsculas, tildes y caracteres especiales ejemplo (mi\_sitio\_web.html, hola\_mundo.html, pagina-prueba-cliente.html, pagina-cliente.html, pagina\_usuario\_proyecto.html).

ESTRUCTURA

<!DOCTYPE *html*>

<html *lang*="es">

<head>

    <meta *charset*="UTF-8">

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

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

    <title>Mi sitio web 2</title>

</head>

<body>

    <p>Este es un párrafo</p>

</body>

</html>

La etiqueta <!DOCTYPE *html*> indica que estamos usando la versión 5 de html

La etiqueta <html *lang*="es"> es conocido de indicar el contenido y el cierre de la etiqueta debe ir después de la etiqueta de cierre del body, el atributo “lang” indicará el lenguaje con el cuál se manejará el sitio.

La etiqueta head es un encabezado lo cual proporciona información al navegador, la etiqueta title que está dentro del head indica el título que se le pondrá a la página en la parte de la pestaña, también podremos colocar las palabras especiales, descripción del sitio web y los estilos de css.

La etiqueta body mostrará todo lo que podrá ver el cliente.

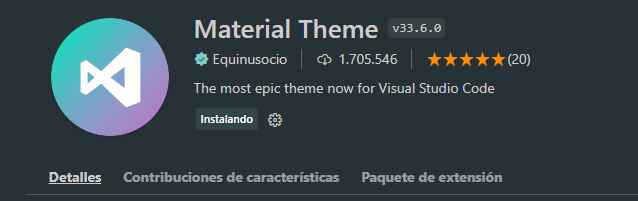
La etiqueta meta charset=”utf-8” establece el juego de caracteres que el documento utilizará, esto incluye todos los idiomas y símbolos especiales.

La etiqueta <meta http-equiv="X-UA-Compatible" content="IE=edge"> es la compatibilidad de internet explorer con edge, actualmente internet explorer está en desuso, así que está en discusiones si utilizar o no este meta.

La etiqueta <meta name="viewport" content="width=device-width, initial-scale=1.0"> la propiedad width controla el tamaño de la ventana gráfica. Se puede establecer un número específico de pixeles como width=600 o el valor especial device-width, que es el ancho de la pantalla en pixeles CSS a una escala del 100%, la propiedad initial-scale controla el nivel de zoom cuando se carga la página por primera vez, el maximum-scale, minimum-scale y user-scalable propiedades que controlan que usuarios están autorizados para agrandar la página adentro o hacia afuera.

EXTENSIONES PARA VISUAL STUDIO CODE QUE NOS AYUDARA AL ESCRIBIR CÓDIGO

Material theme es para cambiar el tema de visual studio code, al instalar esta extensión también nos instala material theme icons



Para que las etiquetas se ordenen al guardar debemos ir a archivos, preferencias, editor de texto y formato y tickear estas 3 opciones, esta configuración aplicara a html, css, js, etc



ENCABEZADOS

Los encabezados llevan etiqueta de apertura y de cierre y van desde el h1 hasta el h6

<!DOCTYPE html>

<html lang="es">

  <head>

    <meta charset="UTF-8" />

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

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

    <title>Document</title>

  </head>

  <body>

    <h1>Este es un título</h1>

    <h2>Este es otro subtítulo</h2>

    <h3>Este es otro subtítulo</h3>

    <h4>Este es otro subtítulo</h4>

    <h5>Este es otro subtítulo</h5>

    <h6>Este es otro subtítulo</h6>

  </body>

</html>

FORMATO DE TEXTOS

La etiqueta P es para párrafos.

La etiqueta B es para texto en negrita.

La etiqueta I es para cursivas.

También se pueden poner etiquetas dentro de etiquetas como muestra el código.

Si dentro de una etiqueta escribimos lorem se nos generará un texto que podemos usar para rellenar.

La etiqueta S es para tachar.

La etiqueta STRONG es para remarcar algo importante, es similar a la etiqueta B.

La etiqueta MARK indica texto marcado/resaltado.

La etiqueta DEL indica texto eliminado, similar a la etiqueta S.

La etiqueta INS indica texto subrayado.

La etiqueta SUB indica texto subíndice.

La etiqueta SUP indica superíndice.

La etiqueta SMALL indica texto peuqeño

<!DOCTYPE html>

<html lang="es">

  <head>

    <meta charset="UTF-8" />

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

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

    <title>Document</title>

  </head>

  <body>

    <p>Este es un ejemplo de párrafo</p>

    <b>Es un texto en negrita</b>

    <i>Cursivas</i>

    <p>

      Este es otro párrafo

      <b><i>Este es un texto en negrita y cursiva</i></b>

    </p>

<p>Lorem ipsum dolor sit amet consectetur, adipisicing elit. Illum, veritatis aperiam labore illo nostrum voluptatem error assumenda temporibus qui repudiandae fugit maiores dolorem, inventore cumque consequatur, quidem dolore praesentium aliquid.</p>

<p>

        <s>Texto tachado</s>

    </p>

    <strong>Esto es importante</strong>

<mark>Esto indica texto destacado</mark>

  </body>

</html>

LISTAS O INDICES

La etiqueta UL indica listas desordenadas.

La etiqueta OL indica listas desordenadas.

La etiqueta LI son los ítems de las listas que van dentro de la etiqueta OL o UL.

<!DOCTYPE html>

<html lang="es">

<head>

    <meta charset="UTF-8">

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

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

    <title>Document</title>

</head>

<body>

    <ul>

        <li>Elemento #1</li>

        <li>Elemento #2</li>

        <li>Elemento #3</li>

        <li>Elemento #4</li>

        <li>Elemento #5</li>

    </ul>

    <ol>

        <li>Elemento #1</li>

        <li>Elemento #2</li>

        <li>Elemento #3</li>

        <li>Elemento #4</li>

        <li>Elemento #5</li>

    </ol>

</body>

</html>

COMENTARIOS Y ANCLAS O VINCULOS

Los comentarios son para textos que no se ven en la página, se utiliza para anotar parte de código o alguna idea por si estamos trabajando con un equipo, también podemos comentar código, hay varias formar de comentarios en css, html, php, javascript, etc.

La etiqueta A es para poner enlaces en su atributo href y en la etiqueta colocamos el texto a donde lleva el enlace, por ejemplo aca tenemos 4 enlaces, el primero lleva a google, el segundo a youtube, el tercero a facebbok y el cuarto a github.

*<!-- Este link lleva a google -->*

    <a href="https://www.google.com/">Google</a>

*<!-- Este link lleva a youtube -->*

    <a href="https://www.youtube.com/">Youtube</a>

*<!-- Este link lleva a facebook -->*

    <a href="https://www.facebook.com/">Facebook</a>

*<!-- Este link lleva a github -->*

    <a href="https://github.com/">Github</a>

IMÁGENES

Con la etiqueta IMG podemos colocar imágenes en nuestra página web mediante su atibuto src lo cual indica la ruta de la imagen y el atributo alt es un texto que podemos colocar el cual reemplazara si la imagen es borrada o no se encuentra, podemos hacerlo de 2 maneras la primera es copiando el enlace de la imagen lo cual no estaríamos sobrecargando nuestra página pero necesitaríamos conexión a internet y si la imagen es borrada del servidor externo donde se encuentra también desaparecerá de nuestra página.

También podemos hacerlo de manera local, descargando la imagen en nuestra carpeta colocando la ruta de donde se encuentra nuestra imagen.

Formatos de imagen soportado:

* Jpeg
* Gif
* Png
* Apng
* Svg
* Bmp
* Bmp ico
* Png ico

<img src="https://www.google.com/url?sa=i&url=https%3A%2F%2Fwww.youtube.com%2Fwatch%3Fv%3D7TezZ2JbvZs&psig=AOvVaw37ltysb0yErG-dcv5HEp3g&ust=1672168690319000&source=images&cd=vfe&ved=0CBAQjRxqFwoTCOjPuf3\_l\_wCFQAAAAAdAAAAABAE" alt="imagen de lenguajes">

<img src="Blog210219\_2.jpg" alt="imagen de programación">

Como estamos trabajando con imágenes dentro de la misma carpeta no necesitamos especificar otro subdirectorio o carpeta, en este ejemplo convertimos una imagen en enlace, ya que al darle click en la imagen nos lleva a otra página:

<a href="https://bluuweb.github.io/desarrollo-web-bluuweb/01-html/#%C2%BFque-es-html">

    <img src="frontend-backend.png" alt="imagen de desarrollo frontend y backend">

</a>

EJERCICIO PRÁCTICO

En el siguiente ejercicio creamos 3 archivos diferentes cada uno con su titulo en H1 y su etiqueta A para enlaces y en en el atributo href de cada etiqueta A colocamos el nombre del archivo donde queremos dirigirnos, estos archivos están dentro del directorio de manera local en la etiqueta A de la pagina 1 nos lleva a la página 2 y su estiqueta A de la página 2 nos lleva a la página 3 y su etiqueta de la página 3 nos devuelve a la página 1

Nota: la etiqueta BUTTON es para crear botones y entre medio de su apertura y cierre colocamos el nombre que tendrá el botón

<h1>Página 1</h1>

    <a href="pagina\_2.html">

        <button>Ir a página #2</button>

    </a>

<h1>Página 2</h1>

    <a href="pagina\_3.html">

        <button>Ir a página #3</button>

    </a>

<h1>Página 3</h1>

    <a href="pagina\_1.html">

        <button>Ir a página #1</button>

    </a>

Live server es una extensión que nos simula un servidor para poder ver los cambios en tiempo real de nuestro código siempre y cuando guardemos los cambios del código en nuestro IDE, ya que un servidor está imitando el comportamiento de nuestro sitio web cuando lo subimos a dominio real.

TABLAS

La etiqueta TABLE es para realizar tablas, dentro de esta etiqueta tenemos las etiquetas TR para filas y TD para columnas, el atributo border en la etiqueta TABLE es para colocar bordes a la tabla

<h1>Tabla 1</h1>

    <table border="1">

*<!-- tr para fila -->*

        <tr>

*<!-- td para columna -->*

            <td>primera fila</td>

            <td>segunda columna</td>

        </tr>

        <tr>

            <td>segunda fila</td>

            <td>segunda columna</td>

        </tr>

    </table>

TABLAS: ENCABEZADOS O HEADERS Y TITULOS CON CAPTION

Colocamos Th dentro de TR, ya que Th resalta el encabezado para mostrar que es un título de una columna, con la etiqueta CAPTION le damos un título a la tabla

<h1>Tabla 1</h1>

    <table border="1">

<caption>Lista de mascotas</caption>

        <tr>

*<!-- th es para encabezados -->*

            <th>Mascota</th>

            <th>Edad</th>

            <th>Color</th>

        </tr>

*<!-- tr para fila -->*

        <tr>

*<!-- td para columna -->*

            <td>Gato</td>

            <td>3 años</td>

            <td>Negro</td>

        </tr>

        <tr>

            <td>Perro</td>

            <td>1 año</td>

            <td>Cáfe</td>

        </tr>

        <tr>

            <td>Conejo</td>

            <td>1 mes/td>

            <td>Blanco</td>

        </tr>

    </table>



TABLAS: THEAD, TBODY Y TFOOT

Con estas etiquetas podemos ordenar nuestras tablas, THEAD es para los encabezados, TBODY para los elementos de la tabla y TFOOT para el pie de la tabla

<h1>Tabla 1</h1>

    <table border="1">

        <caption>Lista de mascotas</caption>

        <thead>

            <tr>

                <th>Mascota</th>

                <th>Edad</th>

                <th>Color</th>

            </tr>

        </thead>

        <tbody>

            <tr>

                <td>Gato</td>

                <td>3 años</td>

                <td>Negro</td>

            </tr>

            <tr>

                <td>Perro</td>

                <td>1 año</td>

                <td>Cáfe</td>

            </tr>

            <tr>

                <td>Conejo</td>

                <td>1 mes/td>

                <td>Blanco</td>

            </tr>

        </tbody>

        <tfoot>

            <tr>

                <th>Mascota</th>

                <th>Edad</th>

                <th>Color</th>

            </tr>

        </tfoot>

    </table>

COMBINAR TABLAS O/Y COLUMNAS

Con el atributo colspan podemos combinar columnas dependiendo de la cantidad que le pongamos en su valor y con rowspan podemos combinar filas dependiendo de la cantidad que le pongamos de valor.

<h1>Tabla 2</h1>

    <table border="1">

        <thead>

            <th>Nombre</th>

            <th>Edad</th>

            <th>Nacionalidad</th>

        </thead>

        <tbody>

            <tr>

                <td>Juan</td>

                <td colspan="2">Sin información</td>

            </tr>

            <tr>

                <td>Mauricio</td>

                <td>23 años</td>

                <td>Bolivia</td>

            </tr>

        </tbody>

    </table>

    <br>

    <table border="1">

        <thead>

            <th>Nombre</th>

            <th>Edad</th>

            <th>Nacionalidad</th>

        </thead>

        <tbody>

            <tr>

                <td>Juan</td>

                <td>20 años</td>

                <td rowspan="2">Sin información</td>

            </tr>

            <tr>

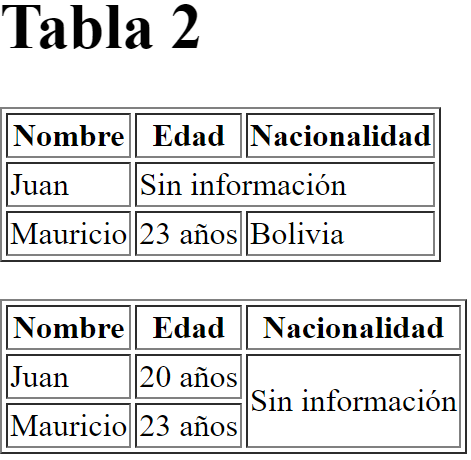
                <td>Mauricio</td>

                <td>23 años</td>

            </tr>

        </tbody>

    </table>



FORMULARIOS

Los formularios son uno de los principales puntos de interacción entre usuario y sitio web o aplicación, ya que permiten ingresar datos que generalmente se envían a un servidor web para su procesamiento y almacenamiento.

Mediante la etiqueta FORM creamos nuestro formulario

*<!-- el atributo action nos sirve para procesar nuestro formulario, osea a donde se iran los datos que seran escritos en el formulario, existen 2 tipos de method: get envia los datos en una cadena visible en el link que se utiliza para enviar pocos datos y post que envia los datos en forma invisible que no se muestra en el link que se utiliza para enviar una gran cantidad de datos -->*

<form action="enviando.html" method="GET">

*<!-- nos ayuda a relacionar o colocar un texto de ayuda para nuestros inputs el atributo for nos ayuda para relacionar el label con el input mediante el identificador que le coloquemos al input -->*

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

*<!-- input es la caja donde el usuario escribira los datos y el atributo type nos indica que tipo de datos puede colocar el usuario estableciendo su valor y el atributo placeholder nos ayuda como una guia para el cliente de los datos que pide el input, al colocarle el atributo name relacionamos el input con el label y tambien para poder trabajar con un archivo javascript-->*

<input name="nombre" id="nombre" type="text" placeholder="Ingrese su nombre">

<br>

<label for="apellido">Apellido: </label>

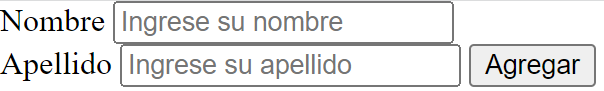
<input name="apellido" type="text" id="apellido" placeholder="Ingrese su apellido">

<br>

*<!-- al poner el atributo type y valor submit indica que el boton procesara los datos del formulario -->*

<button type="submit">Agregar</button>

</form>



INPUTS PARA FORMULARIOS

<form action="">

<input type="email" required>

<br>

<input type="password">

<br>

<textarea name="" id="" cols="30" rows="10"></textarea>

<br>

<input type="radio" name="mascota" value="gato" id="gato">

<label for="gato">Gato</label>

<input type="radio" name="mascota" value="perro" id="perro">

<label for="perro">Perro</label>

<br>

<input type="color">

<br>

<input type="date">

<br>

<input type="file">

<br>

<input type="month">

<br>

<input type="range">

<br>

<input type="time">

<br>

<input type="week">

<br>

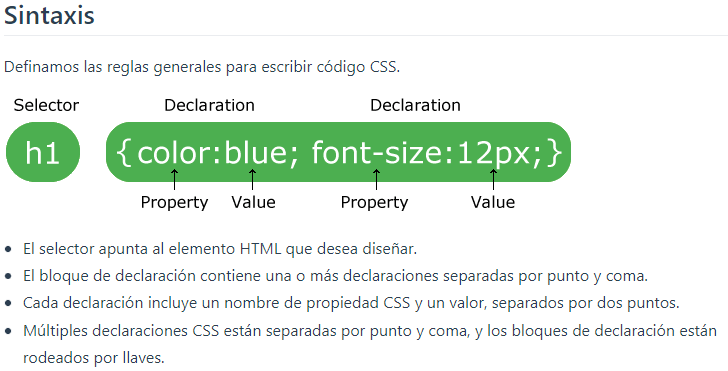
<button type="submit">Enviar</button>

<input type="button" value="Enviar">

</form>

* el input tipo email solo permite correos electrónicos.
* el tipo password oculta los datos que se estan escribiendo en el input.
* textarea es un campo donde se puede mandar comentarios o mensajes que el usuario desee esta etiqueta tiene atributos como el name, el id, col que es para el tamaño de columnas y rows para tamaño de filas.
* el input radio es para opciones el cual elegimos usando un label el cual relacionamos mediante su atributo for y el atributo value de la etiqueta radio, el input radio también cuenta con el atributo name que es para adjuntar varios inputs radio y asi solo elegir una opción.
* el input color nos sirve para elegir colores.
* el input date nos sirve para colocar fechas ya sea con el icono del calendario o manualmente.
* el input file nos permite trabajar con archivos.
* el input range nos permite trabajar con rangos.
* el input time nos permite trabajar con horas.
* el input week nos permite trabajar con semanas.

CSS

Dentro del head podemos invocar nuestro archivo css el cual nos permitirá darle estilo a nuestro sitio web con la siguiente etiqueta <link rel="stylesheet" href="estilos.css"> el atributo href nos permite colocar la ruta de nuestro archivo y el atributo rel nos indica que estamos poniendo una hoja de estilo.

Alternativas para declarar CSS

En línea cuando le damos estilos mediante las etiquetas en html, luego tenemos cuando utilizamos la etiqueta STYLE en el head y luego cuando creamos un archivo aparte con la extensión css (lo mas recomendable es hacerlo en otro archivo con extensión css).

<head>

    <meta charset="UTF-8">

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

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

    <title>3 alternativas para css</title>

    <style>

        h3{

            color: red;

        }

        h4{

            color: orange;

            font-size: 30px;

        }

    </style>

</head>

<body>

    <p style="color: blue; font-size: 50px;">Lorem ipsum dolor</p>

    <h2 style="color: peru;">Lorem ipsum dolor</h2>

    <h3>Lorem ipsum dolor</h3>

    <h4>Lorem ipsum dolor</h4>

</body>

Práctica

Contenido del HTML

<!DOCTYPE html>

<html lang="es">

<head>

<meta charset="UTF-8">

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

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

<title>Sitio web de práctica</title>

<link rel="stylesheet" href="estilos.css">

</head>

<body>

*<!-- header se utiliza para la parte principal también podemos utilizar div (ya que todo lo imaginamos como cajas), en esta parte colocamos los encabezados de la página -->*

<header class="bg-primary">

<hr>

<h1 class="text-light text-center">Mi primer sitio web con CSS</h1>

<hr>

<p class="text-secondary text-center">Lorem ipsum dolor sit amet consectetur.</p>

<hr>

</header>

*<!-- nav para realizar botones o también se puede utilizar div, en esta parte colocamos una botonera -->*

<nav class="text-center">

<button class="text-light bg-secondary width-100">Inicio</button>

<button class="text-light bg-secondary width-100">Nosotros</button>

<button class="text-light bg-secondary width-100">Servicios</button>

<button class="text-light bg-secondary width-100">Blog</button>

<button class="text-light bg-secondary width-100">Contacto</button>

</nav>

*<!-- se utiliza main para el contenido del sitio web, acá colocamos las tarjetas con encabezado, contenido y pie de página -->*

<main>

<table>

<tr>

<td>

<div class="bg-info text-center">

<hr>

<h2 class="text-primary">Lorem, ipsum dolor.</h2>

<hr>

<p>Lorem ipsum dolor sit amet, consectetur adipisicing elit. Minima hic exercitationem ad maiores velit nesciunt saepe accusantium officiis eveniet voluptatum ut at in, quisquam aspernatur cum adipisci nemo dicta. Perferendis.</p>

<hr>

<button class="bg-primary text-light">Más información</button>

<hr>

</div>

</td>

<td>

<div class="bg-info text-center">

<hr>

<h2 class="text-primary">Lorem, ipsum dolor.</h2>

<hr>

<p>Lorem ipsum dolor sit amet, consectetur adipisicing elit. Minima hic exercitationem ad maiores velit nesciunt saepe accusantium officiis eveniet voluptatum ut at in, quisquam aspernatur cum adipisci nemo dicta. Perferendis.</p>

<hr>

<button class="bg-primary text-light">Más información</button>

<hr>

</div>

</td>

<td>

<div class="bg-info text-center">

<hr>

<h2 class="text-primary">Lorem, ipsum dolor.</h2>

<hr>

<p>Lorem ipsum dolor sit amet, consectetur adipisicing elit. Minima hic exercitationem ad maiores velit nesciunt saepe accusantium officiis eveniet voluptatum ut at in, quisquam aspernatur cum adipisci nemo dicta. Perferendis.</p>

<hr>

<button class="bg-primary text-light">Más información</button>

<hr>

</div>

</td>

</tr>

</table>

</main>

*<!-- para colocar contenido en el pie de página, también se puede utilizar div, en esta parte colocamos el contenido del pie de página -->*

<footer class="bg-secondary text-center">

<hr>

<p>Lorem ipsum dolor sit.</p>

<hr>

</footer>

</body>

</html>

Contenido del CSS

*/\* background-color es para colocar color de fondo \*/*

.bg-primary{

    background-color: #46518c;

}

.bg-secondary{

    background-color: #f69e7b;

}

.bg-info{

    background-color: #d4b5b0;

}

*/\* color es para colocar color a las letras \*/*

.text-primary{

    color: #383e56;

}

.text-light{

    color: #eedad1;

}

.text-secondary{

    color: #f69e7b;

}

*/\* text-align es para alinear un texto ya sea center, right o left \*/*

.text-center{

    text-align: center;

}

.width-100{

    width: 100px;

}

CSS intermedio

Bordes

Código html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

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

<title>Document</title>

<link rel="stylesheet" href="estilos.css">

</head>

<body>

<p class="border-solid display-inline">Lorem ipsum dolor sit amet consectetur.</p>

<p class="border-dashed display-inline">Lorem ipsum dolor sit amet consectetur.</p>

<a href="#" class="border-solid display-block">Ancla</a>

</body>

</html>

Código css

*/\* border-style es para definir el estilo de un borde  \*/*

.border-solid{

    border-style: solid;

}

.border-dashed{

    border-style: dashed;

}

*/\* display es para transformar un elemento en linea o en bloque, los elementos en linea ocupan el ancho del espacio del contenido pero los elementos en bloque ocupan todo el ancho disponible*

*inline para convertir el elemento en linea.*

*block para convertir el elemento en bloque.*

*\*/*

.display-inline{

    display: inline;

}

.display-block{

    display: block;

}

Padding y Margin

Código CSS

.border{

    border-style: solid;

}

*/\* padding es el espacio entre el contenido y el contenedor.*

*padding-top espacio para arriba.*

*padding-bottom espacio para abajo.*

*padding-right espacio desde la derecha.*

*padding-left espacio desde la izquierda.*

*al colocar la propiedad padding sin indicar la direccion entonces el valor se tomara para los 4 lados, los valores se añaden de acuerdo a las manecillas de un reloj: arriba, derecha, abajo e izquierda*

*\*/*

.pt-50{

    padding-top: 50px;

}

.pb-50{

    padding-bottom: 50px;

}

.pr{

    padding-right: 50px;

}

.pl-50{

    padding-left: 50px;

}

.p-total{

    padding-top: 50px;

    padding-bottom: 50px;

    padding-right: 50px;

    padding-left: 50px;

}

.p-50{

    padding: 50px;

}

.px-100{

    padding: 0 100px;

}

.py-100{

    padding: 100px 0;

}

.p-completo{

    padding: 50px 20px 30px 15px;

}

*/\* margin es el espacio entre contenedores.*

*margin-top espacio para arriba.*

*margin-bottom espacio para abajo.*

*margin-right espacio desde la derecha.*

*margin-left espacio desde la izquierda.*

*al colocar la propiedad margin sin indicar la direccion entonces el valor se tomara para los 4 lados, los valores se añaden de acuerdo a las manecillas de un reloj: arriba, derecha, abajo e izquierda*

*\*/*

.mt-100{

    margin-top: 100px;

}

.m-50{

    margin: 50px;

}

.mx-100{

    margin: 0 100px;

}

Código HTML

<p class="border pt-50 pb-50 pl-50 mt-100">Lorem ipsum dolor sit.</p>

<p class="border p-total m-50">Maiores autem ipsam consequatur?</p>

<p class="border p-50 mx-100">Repellat voluptate fugit accusamus.</p>

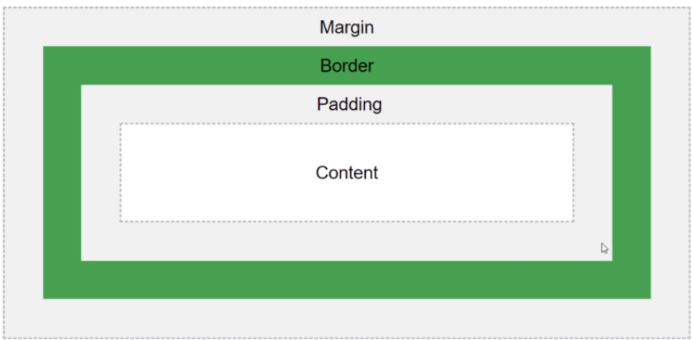
<p class="border px-100">Lorem ipsum dolor sit amet consectetur adipisicing elit.</p>

<p class="border py-100">Lorem ipsum dolor sit amet consectetur adipisicing elit.</p>

<p class="border p-completo">Lorem ipsum dolor sit amet consectetur adipisicing elit.</p>

Modelo de caja

Para adivinar las medidas de nuestra caja debemos calcular el padding, border y el margin.



Ancho y alto

Código html

<p class="border w-100 h-100 p-50 m-50">Lorem ipsum dolor sit</p>

<p class="border w-300 h-100 p-50 border-box m-50">Lorem ipsum dolor sit amet</p>

<a href="" class="border w-100 h-100 d-inline-block">Lorem ipsum dolor sit</a>

<a href="" class="border w-100 h-100 d-inline-block">Lorem ipsum dolor sit</a>

Código css

.border{

    border-style: solid;

}

*/\* width = indica el ancho para un elemento.*

*height = indica el alto para un elemento.*

*si le ponemos alto y ancho a un elemento en linea no nos obedecera\*/*

.w-100{

    width: 100px;

}

.w-300{

    width: 300px;

}

.h-100{

    height: 100px;

}

.p-50{

    padding: 50px;

}

.m-50{

    margin: 50px;

}

*/\* display : inline-block permite colocar ancho y alto a los elementos en linea y no agrega saltos de linea, osea que los agrupara uno al lado de otro \*/*

.d-inline-block{

    display: inline-block;

}

*/\* box-sizing: border-box \*/*

.border-box{

    box-sizing: border-box;

}

Normalize

Normalize.css hace que los navegadores procesen todos los elementos de manera más consistente y en línea con los estándares modernos. Precisamente se dirige solo a los estilos que necesitan normalizarse, apenas es un fichero que pesa 8kb, cuyo principal objetivo es mantener los estilos similares en los navegadores, ya que cada uno agrega sus propios estilos por defecto, por ejemplo, te suelen agregar distintos paddings, margins, font-sizes, etc.

Unidades relativas y absolutas

En css se pueden utilizar diferentes unidades de medida y no existe “la más recomendable”. Es necesario conocerlas y saber cómo funcionan en el contexto para evaluar cuál es la más conveniente en cada caso.

Medidas absolutas

Su valor se encuentra definido en términos concretos y de manera medible. Esto quiere decir que no depende de otro valor de referencia, ni del contexto.

* mm: milímetros.
* cm: centímetros.
* in: pulgada ("inches", en inglés). Una pulgada equivale a 2.54 centímetros.
* pt: puntos. Un punto equivale a 1 /72 de pulgada, es decir, unos 0.35 milímetros.
* pc: picas. Una pica equivale a 12 puntos, o aproximadamente a 4.23 milímetros.
* px: pixel. Es la unidad mínima de resolución de la pantalla. En realidad, suele considerársela una unidad.

Medidas relativas

Las unidades relativas no son valores exactos, sino que se calculan a partir de otro valor de referencia. A pesar de parecer más difíciles de calcular son las más utilizadas en el diseño de sitios web responsive por su adaptabilidad a los diferentes dispositivos.

* em: tamaño relativo al tamaño de su contenedor.
* rem: funciona igual que el em, con la diferencia que es relativo al valor de la fuente del elemento html, osea que toma como base el tamaño del body (tiene como tamaño predeterminado 16px).

Práctica

Código html

<!DOCTYPE html>

<html lang="es">

<head>

<meta charset="UTF-8">

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

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

<title>Práctica</title>

*<!-- normalize debe ir primero que los estilos propios -->*

<link rel="stylesheet" href="normalize.css">

<link rel="preconnect" href="https://fonts.googleapis.com">

<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>

<link href="https://fonts.googleapis.com/css2?family=Roboto:wght@100;400;900&display=swap" rel="stylesheet">

<link rel="stylesheet" href="estilos.css">

</head>

<body>

<header class="container mt-2 mt-2">

<a href="#" class="logo">Logo Empresa</a>

</header>

<nav class="container mt-2 mt-2">

<a href="#" class="btn btn-primary">Primary</a>

<a href="#" class="btn btn-secondary">Secondary</a>

<a href="#" class="btn btn-danger">Danger</a>

<a href="#" class="btn btn-info">Info</a>

</nav>

*<!-- la idea de una section es que contenga contenido similar a la seccion -->*

<section class="container mt-2">

*<!-- article habla de un tema en particular -->*

<article class="card">

<img src="assets/img/800x500.jpg" alt="" class="card-img">

<div class="card-body">

<h5>Lorem, ipsum dolor.</h5>

<p>Lorem ipsum dolor, sit amet consectetur adipisicing elit. Sunt assumenda minima esse quisquam iusto porro voluptate quod corporis molestiae culpa dolores, veniam eos, impedit odio in sint fuga dolorem. Aperiam.</p>

<a href="#" class="btn btn-info">Info</a>

</div>

</article>

</section>

</body>

</html>

Código css

*/\* esta seria la mejor configuracion para que los elementos contengan en valor de border-box\*/*

html{

    box-sizing: border-box;

}

\*,

\*:before,

\*:after{

    box-sizing: inherit;

}

*/\* en los valores de font-family indica que tipo de letra se usara para trabajar, tiene más de 2 valores por si un tipo de letra no lo encuentra entonces utilizara el valor que le sigue \*/*

body{

    font-family: 'Roboto', sans-serif;

}

*/\* con max-width definimos un ancho maximo que tendra los contenedores, también definimos un ancho de 95%, centramos los contenedores con margin left y right asignandoles un valor automatico y un fondo aqua*

*\*/*

.container{

    width: 95%;

    max-width: 600px;

    margin-left: auto;

    margin-right: auto;

}

*/\* aca el margin sera de 32 pixeles porque rem trabaja de acuedo a la medida base del body que serian 16 pixeles \*/*

.mt-2{

    margin-top: 2rem;

}

*/\* con text-decoration podemos controlar el subrayado que tiene la etiqueta a, tambien le agregamos display block para convertirlo en elemento de bloque ya que los elementos en linea no se pueden centrar, le pusimos font-size 3rem(48px ya que la medida del body es de 16px y rem trabaja en base a esa medida), el padding de 1em(em trabaja de acuerdo a la medida de su contenedor padre y aca esta trabajando con 48 pixeles ya que esta trabajando en base al fontsize que es de 48 pixeles, colo blanco, fondo aqua pero con codigo hexadecimal y lo alineamos al centro con text-align center) \*/*

.logo{

    text-decoration: none;

    display: block;

    font-size: 3rem;

    padding: 1em;

    color: white;

    background-color: #264653;

    text-align: center;

}

*/\* display inline-block es un elemento en linea pero con los beneficios de un elemento en bloque \*/*

.btn{

    display: inline-block;

    text-decoration: none;

    font-size: 1rem;

    padding: 0.5em;

    margin: 0 0.1em;

    border-radius: 0.5em;

}

.btn-primary{

    background-color: #264653;

    color: white;

}

.btn-secondary{

    background-color: #2a9d8f;

    color: white;

}

.btn-danger{

    background-color: #e76f51;

    color: white;

}

.btn-info{

    background-color: #e9c46a;

    color:white;

}

*/\* ponemos un ancho de 20rem, un fondo, overflow hidden recorta el sobrante del contenido, rem aca toma como medida la del html ya que article no esta cnfigurado, tampoco section ni el body y por eso toma como medida el tamaño del html \*/*

.card{

    width: 20rem;

    background-color: #264653;

    overflow: hidden;

    color: white;

    border-radius: 0.5rem;

}

*/\* ponemos un width de 100% para que la imagen no sobresalga del contenedor y ocupe el espacio disponilbe completo del contenedor \*/*

.card-img{

    width: 100%;

}

.card-body{

    padding: 1em;

}

*/\* aca estamos llamando al card-boy y al/los h5 que esten adentro del contenedor con la clase card-body luego ponemos un margin y un font-weight que esto indica el ancho de la letra(como hemos asignado 400 y 900 entonces 900 está disponible ya que es una tipografía de un sitio web) \*/*

.card-body h5{

    font-size: 1.5rem;

    margin: 0.5em 0;

    font-weight: 900;

}

.card-body p{

    font-weight: 400;

}

Flexbox

El módulo de caja flexible, comúnmente llamado flexbox, fue diseñado como un modelo unidimensional de layout y como un método que pueda ayudar a distribuir el espacio entre los ítems de una interfaz y mejorar las capacidades de alineación.

Código html

<!DOCTYPE html>

<html lang="es">

<head>

<meta charset="UTF-8">

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

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

<title>Flexbox</title>

<link rel="stylesheet" href="estilos.css">

</head>

<body class="container">

*<!-- al contenedor div con clase border es el contenedor padre ya que tiene contenedores adentros -->*

<div class="border flex-container">

<div class="item flex-grow-1 flex-basis-1">item 1</div>

<div class="item flex-grow-1 flex-shrink-0">item 2</div>

<div class="item flex-grow-1">item 3</div>

</div>

<div class="border flex-container">

<div class="item flex-2">item 1</div>

<div class="item flex-2">item 2</div>

<div class="item flex-2">item 3</div>

</div>

<div class="border flex-container">

<div class="item h-300">item 1</div>

<div class="item align-self-start">item 2</div>

<div class="item">item 3</div>

</div>

</body>

</html>

Código css

*/\* le pusimos una clase container al body con un width de 90%, y lo centramos con margin left y right automático y un espacio desde arriba con margin-top de 16px \*/*

.container{

    width: 90%;

    margin-left: auto;

    margin-right: auto;

    margin-top: 16px;

}

*/\* una clase a un div con un borde de 2px color negro y sólido \*/*

.border{

    border: 2px #000 solid;

}

.h-700{

    height: 700px;

}

.h-300{

    height: 300px;

}

*/\* a cada div hijo le pusimos una clase item que están dentro de un div padre llamado border, con fondo peru, texto centrado, borde de 2px negro y sólido, con borde radius de 10px, un margin de 10px y un padding de 10px \*/*

.item{

    background-color: peru;

    text-align: center;

    border: 2px #000 solid;

    border-radius: 10px;

    margin: 10px;

    padding: 10px;

*/\* width: 200px; \*/*

}

*/\* si le ponemos display flex al contenedor padre entonces sus elementos hijos se haran flexibles ya que se posicionan uno al lado del otro, por defecto estaran en row que se distribuyen de izquierda a derecha, si ponemos flex-direction: row-reverse se distribuiran de derecha a izquierda, con column se distribuyen de arriba hacia abajo.*

*con flex-wrap: wrap, podemos hacer que las cajas que no tengan el espacio suficiente se acomoden para abajo y con reverse se acomoden de abajo para arriba.*

*con justify-content podemos distribuir los elementos en el contenedor, flex-start los distribuye desde el principio, flex-end desde el final, center los distribuye en el centro, space-between los distribuye uniformemente en la linea primer elemento está en el inicio y el último en el final, space-around los distribuye casi parecido al centro pero con un espacio entre elementos, space-evenly hace lo mismo solo que con un espacio más pequeño de separación.*

*con align-items primero debemos establecer una altura al contenedor padre, el valor por defecto es stretch estira los elementos ocupado la altura disponible de su contenedor, flex-start los elementos se colocan al principio, flex-end los elementos se posicionan abajo del contenedor, center se centran, baseline se alinean según el texto.*

*con align-content alinemos los elementos verticalmente, flex-star es el valor por defecto, flex-end los alinea para abajo, center los centra, stretch los estira según la altura que tengan disponible, space-between los alinea colocandolos al borde del contenedor junto con los elementos para el centro, space-around los alinea dejando una separación entre los bordes del contenedor padre.*

*\*/*

.flex-container{

    display: flex;

*/\* flex-direction: row-reverse; \*/*

    flex-wrap: wrap;

*/\* justify-content: center; \*/*

    align-items: flex-end;

*/\* align-content: space-between; \*/*

}

*/\* los elementos hijos tambien tienen propiedades*

*con order podemos establecer el orden de los elementos hijos dependiendo del valor que le asignemos.*

*\*/*

.order-1{

    order: 0;

}

.order-2{

    order: 2;

}

*/\* con flex-grow podemos distribuir el tamaño de los elementos de acuerdo al valor que le indiquemos para asi no estar calculando el ancho necesitamos asignar \*/*

.flex-grow-1{

    flex-grow: 1;

}

*/\* debemos primero establecer un ancho ya que flex-shrink indica cual sera su ancho minimo, aca le pondremos 300pixeles y ese será su ancho mínimo \*/*

.flex-shrink-0{

    width: 300px;

    flex-shrink: 0;

}

*/\* flex-basis obliga a un item a tener una proporción determinada, aca por ejemplo le asignamos que ocupe el espacio del 50% disponible\*/*

.flex-basis-1{

    flex-basis: 50%;

}

*/\* aca combinaremos flex-grow, shrink y basis en un nuevo contenedor que hemos creado.*

*el primer valor correponde a flex-grow.*

*el segundo valor corresponde a flex-shrink.*

*el tercer valor corresponde a flex-basis.*

*\*/*

.flex-1{

    flex: 1 0;

}

.flex-2{

    flex: 1 0 300px;

}

*/\* align-self le decimos a un elemento hijo que se posicione donde queramos \*/*

.align-self-start{

    align-self: flex-start;

}

Práctica flexbox

Código html

<!DOCTYPE html>

<html lang="es">

<head>

<meta charset="UTF-8">

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

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

<title>Altos Game</title>

<link rel="stylesheet" href="normalize.css">

<link rel="preconnect" href="https://fonts.googleapis.com">

<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>

<link href="https://fonts.googleapis.com/css2?family=Raleway:wght@100;200;300;400;600&display=swap" rel="stylesheet">

<link rel="stylesheet" href="estilos.css">

</head>

<body>

*<!-- utilizaremos la técnica mobile first, primero maquetaremos la versión móvil -->*

*<!-- esta es la cabecera -->*

<header class="bg-header">

<div class="bg-navbar">

<div class="container">

<div class="navbar">

<a href="#" class="navbar-brand">

    <img src="assets/icon-game.svg" alt="juego ícono">

    Alto's Game

</a>

<nav class="navbar-nav">

    <a href="#" class="nav-link">Inicio</a>

    <a href="#" class="nav-link">Tienda</a>

    <a href="#" class="nav-link">Nosotros</a>

    <a href="#" class="nav-link">Contacto</a>

</nav>

</div>

</div>

</div>

</header>

<main class="container main">

<h1 class="main-title">Nuestros productos</h1>

<section class="tienda">

<article class="card">

<img src="assets/card-1-min.png" class="card-img" alt="">

<div class="card-body">

<h5>Lorem, ipsum dolor.</h5>

<p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Inventore aperiam pariatur expedita voluptate, error quidem, laudantium molestiae corporis officiis libero rerum totam sunt nulla officia repellat natus. Minima, eveniet natus.</p>

<a href="#" class="btn btn-dark">Google</a>

</div>

</article>

<article class="card">

<img src="assets/card-2-min.jpg" class="card-img" alt="">

<div class="card-body">

<h5>Lorem, ipsum dolor.</h5>

<p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Inventore aperiam pariatur expedita voluptate, error quidem, laudantium molestiae corporis officiis libero rerum totam sunt nulla officia repellat natus. Minima, eveniet natus.</p>

<a href="#" class="btn btn-dark">Google</a>

</div>

</article>

<article class="card">

<img src="assets/card-3-min.jpg" class="card-img" alt="">

<div class="card-body">

<h5>Lorem, ipsum dolor.</h5>

<p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Inventore aperiam pariatur expedita voluptate, error quidem, laudantium molestiae corporis officiis libero rerum totam sunt nulla officia repellat natus. Minima, eveniet natus.</p>

<a href="#" class="btn btn-dark">Google</a>

</div>

</article>

</section>

<section class="nosotros">

<div class="nosotros-col-img"></div>

<div class="nosotros-col-text">

<div class="nosotros-body">

<h2>Nosotros</h2>

<p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Sequi harum saepe facere tempora fugit earum iste! Corporis cum, obcaecati fuga facere commodi quasi, nemo sed dicta fugiat, accusamus eos ullam?</p>

<a href="#" class="btn btn-danger">Más información</a>

</div>

</div>

</section>

</main>

<footer class="bg-footer">

<div class="container">

<p>Lorem, ipsum dolor.</p>

</div>

</footer>

</body>

</html>

Código css con mediaquerie

html{

    box-sizing: border-box;

}

\*,

\*:after,

\*:before{

    box-sizing: inherit;

}

body{

    font-family: 'Raleway', sans-serif;

}

*/\* clase utilitaria, porque la voy a usar más de una vez.*

*estilos para el container*

*width de 80%.*

*un max-width de 1280px para que no sobrepase de ese tamaño.*

*margin right y left automáticos para que el contenedor se centre.*

*\*/*

.container{

    width: 80%;

    max-width: 1280px;

    margin-right: auto;

    margin-left: auto;

}

*/\* header*

*con background-image elegimos la imagen de fondo.*

*con background-repeat: no-repeat la imagen no se repite si es muy pequeña.*

*con background-position: center centramos la imagen.*

*con background-size: cover la imagen se adaptara al espacio disponible de su contenedor.*

*con height le pusimo 18.75em que serian 300px ya que esta tomando el tamaño del html como referencia.*

*\*/*

.bg-header{

    background-image: url('assets/bg-header-min.jpg');

    background-repeat: no-repeat;

    background-position: center;

    background-size: cover;

    height: 18.75em;

}

*/\* un fondo oscuro con opacidad.*

*padding de 1.5em arriba y abajo y 0 para los lados. \*/*

.bg-navbar{

    background-color: rgba(0, 0, 0, 0.288);

    padding: 1.5em 0;

}

*/\* le aplicamos display: flex para que sean elementos flexibles.*

*flex-direction-column para distribuirlos de arriba hacia abajo.*

*y align items porque estamos trabajando de manera vertical.*

*media querie en 768px navbar*

*posicionamos los elementos en fila.*

*distribuimos los elementos dejando un espacio a los lados\*/*

.navbar{

    display: flex;

    flex-direction: column;

    align-items: center;

}

*@media* (min-width:768px){

    .navbar{

        flex-direction: row;

        justify-content: space-evenly;

    }

}

*/\* display: flex para que sean elementos flexibles.*

*flex-direction: column para distribuirlos de arriba hacia abajo.*

*color blanco al texto.*

*align-items center para centrar los elementos.*

*font-size 2rem.*

*text-decoration none para quitarle el subrayado.*

*font-weight 600 seria el grosor que solicite a la fuente.*

*y un margin-bottom 1em seria 32 pixeles ya que toma como base la fuente del texto.*

*media querie navbar-brand en 768px.*

*posicionamos los elementos en fila.*

*espacio externo hacia abajo de 0px\*/*

.navbar-brand{

    display: flex;

    flex-direction: column;

    color: white;

    align-items: center;

    font-size: 2rem;

    text-decoration: none;

    font-weight: 600;

    margin-bottom: 1em;

}

*@media* (min-width:768px){

    .navbar-brand{

        flex-direction: row;

        margin-bottom: 0;

    }

}

*/\* le damos un ancho de 2em a la imagen que se encuentra dentro del contenedor con la clase navbar-brand \*/*

.navbar-brand img{

    width: 2em;

}

*/\* un fondo color negro con transparencia.*

*un padding de 1em.*

*border-radius: 0.3em. \*/*

.navbar-nav{

    background-color: rgba(0, 0, 0, 0.884);

    padding: 1em;

    border-radius: 0.3em;

}

*/\* display: inline-block permite agregar ancho y alto a elementos en linea.*

*padding de 0.3em.*

*text-decoration: none.*

*color: white \*/*

.nav-link{

    display: inline-block;

    padding: 0.3em;

    text-decoration: none;

    color: white;

}

*/\* con la pseudoclase hover le añadimos un estilo que se activara al pasar el elemento con el mouse.*

*fondo color negro.*

*texto color blanco.*

*bordes redondeados.\*/*

.nav-link:hover{

    background-color: black;

    color: white;

    border-radius: 0.3em;

}

*/\* main*

*fondo blanco.*

*espacio interno de 0.3em*

*espacio externo negativo de -2em.*

*borde redondeado de 0.3em.*

*main-title*

*alineamos el texto al centro.*

*tranformamos el texto a mayúscula con la propiedad text-transform.*

*\*/*

.main{

    background-color: white;

    padding: 0.3em;

    margin-top: -2em;

    border-radius: 0.3em;

}

.main-title{

    text-align: center;

    text-transform: uppercase;

}

*/\* ponemos un ancho de 20rem, un fondo, overflow hidden recorta el sobrante del contenido, rem aca toma como medida la del html ya que article no esta cnfigurado, tampoco section ni el body y por eso toma como medida el tamaño del html \*/*

.card{

    background-color: #A6444C;

    color: white;

    overflow: hidden;

    border-radius: 0.5rem;

    margin-bottom: 1em;

}

*/\* ponemos un width de 100% para que la imagen no sobresalga del contenedor y ocupe el espacio disponilbe completo del contenedor \*/*

.card-img{

    width: 100%;

}

.card-body{

    padding: 1em;

}

*/\* aca estamos llamando al card-boy y al/los h5 que esten adentro del contenedor con la clase card-body luego ponemos un margin y un font-weight que esto indica el ancho de la letra(como hemos asignado 400 y 900 entonces 900 está disponible ya que es una tipografía de un sitio web) \*/*

.card-body h5{

    font-size: 1.5rem;

    margin: 0.5em 0;

    font-weight: 500;

}

.card-body p{

    font-weight: 400;

}

*/\* display inline-block es un elemento en linea pero con los beneficios de un elemento en bloque \*/*

.btn{

    display: inline-block;

    text-decoration: none;

    font-size: 1rem;

    padding: 0.5em;

    margin: 0 0.1em;

    border-radius: 0.5em;

}

.btn-danger{

    background-color: #A6444C;

    color:white;

}

.btn-primary{

    background-color: #0F2620;

    color:white;

}

.btn-secondary{

    background-color: #7BA69A;

    color:white;

}

.btn-light{

    background-color: #DDF2CE;

    color:white;

}

.btn-dark{

    background-color: #302C2D;

    color:white;

}

*/\* nosotros*

*convertimos los elementos en cajas flexibles.*

*los direccionamos verticalmente.*

*los alineamos al centro.*

*le damos espacio externo abajo de 1em*

*fondo color café.*

*.nosostros-col-img*

*ponemos la imágen de fondo.*

*ponemos la imágen al centro.*

*le decimos que no se repita si la imágen es pequeña.*

*que se adapte al espacio disponible.*

*altura de 21.875em.*

*ancho del 100% (sin esta propiedad la imágen no se verá ya que utilizamos flex en el contenedor).*

*ordén de 2.*

*nosotros-col-text*

*ordén de 1.*

*nosotros-body*

*espacio interno de 1em.*

*alineamos el texto al centro.*

*color blanco del texto.*

*nosotros-body h2 = entramos al h2 donde un elemento tenga la clase nosotros-body*

*transformamos el texto en mayúscula.*

*le ponemos un grosor a la fuente de 600.*

*nosotros-body p = entramos al p donde un elemento tenga la clase nosotros-body*

*le ponemos un grosor a la fuente de 200.*

*\*/*

.nosotros{

    display: flex;

    flex-direction: column;

    align-items: center;

    margin-bottom: 1em;

    background-color: #302C2D;

}

.nosotros-col-img{

    background-image: url("assets/nosotros-min.png");

    background-position: center;

    background-repeat: no-repeat;

    background-size: cover;

    height: 21.875em;

    width: 100%;

    order: 2;

}

.nosotros-col-text{

    order: 1;

}

*@media* (min-width:576px){

    .nosotros-col-img{

        height: 31.25em;

    }

}

*@media* (min-width:768px){

    .nosotros{

        flex-direction: row;

    }

    .nosotros-col-img{

        order: 1;

        flex: 1 1 0;

    }

    .nosotros-col-text{

        order: 2;

        flex: 1 1 0;

    }

}

.nosotros-body{

    padding: 1em;

    text-align: center;

    color: white;

}

.nosotros-body h2{

    text-transform: uppercase;

    font-weight: 600;

}

.nosotros-body p{

    font-weight: 200;

}

*/\* footer*

*.bg-footer*

*le ponemos fondo café oscuro.*

*texto blanco.*

*bg-footer p = tomamos la etiqueta p del elemento que tenga la clase bg-footer.*

*un espacio externo de 0 a todos los lados.*

*espacio interno de arriba y abajo de 1 em y a los lados de 0.*

*centramos el texto.*

*\*/*

.bg-footer{

    background-color: #302C2D;

    color: white;

}

.bg-footer p{

    margin: 0;

    padding: 1em 0;

    text-align: center;

}

*/\* media querie en 768px de la clase tienda*

*le ponemos flex para que los elementos se hagan flexibles.*

*gap otorga un espacio entre los elementos que esten dentro del contenedor, el espacio depende del valor que nosotros le asignemos. \*/*

*@media* (min-width:768px){

    .tienda{

        display: flex;

        gap: 0.5em;

    }

}

Mediaqueries

Se utiliza los mediaqueries para controlar nuestros elementos y poder realizar diseños adaptables para diferentes dispositivos.

Código html (ejemplo)

<!DOCTYPE html>

<html lang="es">

<head>

    <meta charset="UTF-8">

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

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

    <title>Media Queries</title>

    <style>

        main{

            background-color: black;

            color: white;

        }

*/\* cuando la pantalla tenga como mínimo, desde un punto hacia adelante \*/*

*@media* (min-width: 576px){

            main{

                background-color: peru;

                display: flex;

            }

            article{

                font-size: 2rem;

            }

        }

*@media* (min-width: 768px){

            main{

                background-color: violet;

            }

            article{

                font-size: 3rem;

            }

        }

    </style>

</head>

<body>

    <main>

        <article>

            <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Distinctio totam nobis unde in fuga deserunt mollitia. Fuga id deleniti nostrum earum sint sed maxime, soluta maiores impedit alias voluptates doloremque.</p>

        </article>

        <article>

            <p>Quam corporis omnis deserunt eligendi necessitatibus fugit in a molestias, dolore ut consequuntur, sit quas aspernatur! Autem veniam suscipit perferendis sit commodi illo fuga, unde veritatis facilis consequuntur eveniet odio.</p>

        </article>

    </main>

</body>

</html>

Terminal

La consola CMD es la ventana de comandos con la que se puede dar órdenes simples a un ordenador sin usar la interfaz gráfica de Windows, con ella se puede navegar en equipos, unidades, carpetas y archivos, visualizar datos técnicos y ejecutar tareas básicas.

Help = nos ayuda para mostrarnos una lista de todos los comandos y su definición.

Help exit = si lo ponemos con un comando este nos mostrará su definición.

Exit = cierra la terminal.

Cd = muestra el nombre del directorio actual o cambia de directorio. Ejemplo: si queremos viajar a esta ruta:” C:\Users\bluuweb\Desktop\jugando con la terminal” debemos poner lo siguiente:

Cd desktop.

Cd jugando con la terminal o Cd “jugando con la terminal”.

Cd.. = Para retroceder un directorio.

Dir = el comando lista el contenido del directorio o carpeta donde te encuentras, mostrando todas las subcarpetas o archivos que tiene. Con este comando podrás saber si el archivo que buscas está ahí o a que subcarpeta navegar.

Cls = limpia la consola.

Mkdir o md = sirve para crear una carpeta.

Move = podemos mover archivos a otras carpetas. Ejemplo: move app.js assets (en lo ultimo definimos a que carpeta queremos mover).

En caso de mover a otro directorio utilizar comillas dobles: move index.html “C:\users\bluuweb\desktop”.

Ctrl + c = termina con una ejecución, este comando es muy útil.

A diferencia de powershell necesitamos las comillas si la carpeta tiene espacios, ejemplo:

Cd “jugando con la terminal”.

Podemos utilizar mv o move para mover carpetas.

Podemos ver el path o ruta actual con pwd.

Para crear archivos: ni index.html.

GIT (para trabajar desde la computadora)

Es un software de control de versiones, su propósito es llevar registro de los cambios en archivos de computadora y coordinar el trabajo que varias personas realizan sobre los archivos compartidos (también se puede trabajar solo), existe la posibilidad de trabajar de forma remota y una opción es github, git nos permite:

* Permite regresar a versiones anteriores de forma sencilla y muy rápida.
* Facilita el trabajo colaborativo.
* Permite respaldar los proyectos en la nube (ej con github).
* Reduce considerablemente los tiempos de deploy.
* Las “branches” o ramas, permiten trabajar con una base de código paralela al proyecto en si.

GITHUB (para trabajar remoto)

Es una plataforma de desarrollo colaborativo para alojar proyectos (en la nube) utilizando el sistema de control de versiones GIT, además cuenta con una herramienta muy útil que es GitHub Pages donde podemos publicar nuestros proyectos estáticos (html, css y js) gratis.

Comandos de git

git versión = podemos ver la versión de git que tenemos instalado.

git help = nos muestra ayuda sobre los comandos.

git init = iniciamos un nuevo repositorio, crea la carpeta oculta “.git” y solo se ejecuta una vez por proyecto.

git status –s = podemos ver que archivos no han sido registrados.

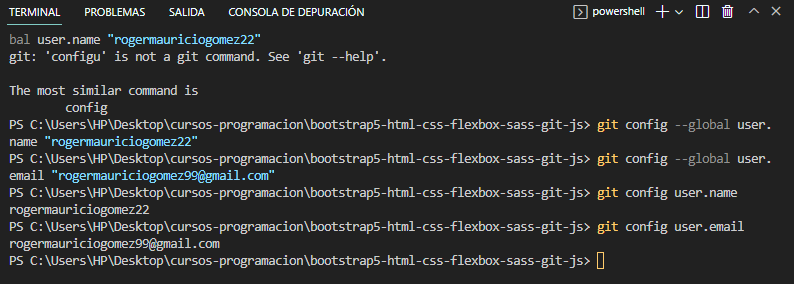
git add . = agrega todos los archivos para que esté pendiente de los cambios.

git commit –m “mensaje que le ponemos a nuestro repositorio” = crear commit (fotografía del proyecto en ese momento).

git log –oneline = muestra la lista de commit del más reciente al más antiguo.

git remote –v = nos muestra en que repositorio estamos enlazados remotamente.

Vim es el editor de código en la línea de comandos, para salir del modo edición Vim debemos escribir “:q”, en caso de realizar algún cambio sin guardar escribir “:qa”, también sirve para salir sin guardar “:q!”.

Cuenta de git

Para futuros cambios y subir los registros a github ejecutar:

* Git add .
* Git commit –m “nuevos cambios”.
* Git push.

Para registrarse de manera local en la computadora

donde trabajaremos abrimos la terminal y ponemos: git config --global user.name “rogermauriciogomez22” y al darle enter no les aparece un mensaje entonces se registro correctamente, ahora para el correo ponemos: git config --global user.email “rogermauriciogomez99@gmail.com” ahora para saber si esta todo correcto debemos poner: git config user.name (para usuario) y git config user.email (para correo) y la terminal debe mostrarnos el correo y el usuario que registramos.

Para subir nuestro repositorio

Ahora para subir nuestro repositorio debemos hacer lo siguiente: ir al ícono + y new repository, luego nos saldrá un cuadro de texto al lado de nuestro nombre de usuario en el cuál pondremos el título de nuestro repositorio (sin espacios), seleccionamos public o privado y le damos en crear repositorio, si en la computadora esta una cuenta de git que no es nuestra entonces debemos eliminar los certificados yendo al panel de control, cuentas de usario, administrar credenciales, credenciales de Windows, buscamos la cuenta de git que está registrada, le damos en quitar y confirmamos.

Para trabajar de manera remota haremos lo siguiente:

git remote add origin https://github.com/rogermauriciogomez22/eliminar-por-favor.git.

git log --oneline (para saber si estamos trabajando con la rama master o main).

git branch -M master o main (depende de la rama que nos muestre).

git push -u origin master o main (depende con que rama estemos trabajando).

Luego nos sale una ventana para poder ingresar nuestra cuenta de github (si es no tenemos cuenta registrada en la computadora).

Github Pages

Nos permite subir páginas estáticas, para poder subir necesitamos hacer lo siguiente: vamos a settings, luego a pages, en branch seleccionamos la rama con la que estamos trabajando ya sea main o master, elegimos a root y le damos en save, denemos esperar unos minutos para que nos entregue el link de nuestra página.

El readme lo utilizamos para documentar, le damos en crear readme y ponemos los comentarios del trabajo que realizamos y le damos en new commit file.

Clonar nuestro repositorio

Para clonar nuestro repositorio debemos abrir el cmd y ubicarnos en la carpeta donde queremos clonar y ponemos el siguiente comando: git clone https://github.com/rogermauriciogomez22/eliminar-por-favor.git (aca sería el link del repositorio que queremos clonar, el link lo obtenemos cuando damos click en el ícono code y copiamos el link que nos muestra).

Ignorando archivos

Para no hacer seguimiento de carpetas o archivos, debemos crear el siguiente archivo: .gitignore.

Su estructura de ejemplo sería así: dentro del archivo ponemos el nombre del archivo que queramos ignorar

archivo.js //ignora el archivo en cuestión.

\*.js //ignora todos los archivos con extensión js.

node\_modules/ //ignora toda la carpeta.

Para ver los commits realizados

Con git log --oneline podemos ver los commits realizados, a lo que nos muestra un identificador (cada confirmación tiene un hash SHA-1 único).

Si queremos visitar un commit en específico utilizamos: git checkout a00beae (este es un identificador de prueba, acá debemos poner el identificador que asigna los commits y los podemos ver con git log --oneline).

Durante el curso normal del desarrollo. HEAD apunta por lo general a la rama main u otra rama local (HEAD -> master) pero cuando extraes una configuración anterior, HEAD ya no apunta a una rama, sino que apunta directamente a una confirmación, este estado recibe el nombre de “HEAD desasociado” (detached HEAD).

Comprobar un commit específico pondrá el repositorio en un estado “HEAD desasociado”, esto significa que ya no estás trabajando en ninguna rama, para eso debemos volver al master con el siguiente comando:

git checkout master (aca volvemos a la rama master).