**HTML**

* **<from>**
  + **<label>**
    - **for** - indicarle el ID del input que está relacionado, al hacer clic en el label, el foco se posicionará en el input relacionado.
  + **<input>**
    - **type -** text, email, radio, checkbox, textarea, submit, etc.
    - **placeholder** - En un tipo text, se agrega un texto de ayuda.
    - **required**
    - **value** - Valor que se enviará al servidor.
    - **name** - El nombre de lo que enviaremos al servidor. En el caso de los radios se pone el mismo name para crear un radiogroup.
    - **checked**
* **<table>**
  + **<thead>**
    - **<tr> - Fila**
    - **<th> - Celdas para cabeceras**
  + **<tbody>**
    - **<tr> - Fila**
    - **<td> - Celdas para cabeceras**
* **<span>** - Se puede utilizar para encerrar texto dentro del HTML y poderle poner estilo.

**HTML5**

* **<header>** Este elemento presenta información introductoria y puede ser aplicado en diferentes secciones del documento. Tiene el propósito de contener la cabecera de una sección pero también puede ser utilizado para agrupar índices, formularios de búsqueda, logos, etc…
* **<nav>** Este elemento indica una sección de enlaces con propósitos de navegación, como menús o índices. No todos los enlaces dentro de una página web tienen que estar dentro de un elemento <nav>, solo aquellos que forman partes de bloques de navegación.
* **<section>** Este elemento representa una sección general del documento. Es usualmente utilizado para construir varios bloques de contenido (por ejemplo, columnas) con el propósito de ordenar el contenido que comparte una característica específica, como capítulos o páginas de un libro, grupo de noticias, artículos, etc…
* **<aside>** Este elemento representa contenido que está relacionado con el contenido principal pero no es parte del mismo. Ejemplos pueden ser citas, información en barras laterales, publicidad, etc…
* **<footer>** Este elemento representa información adicional sobre su elemento padre. Por ejemplo, un elemento insertado al final del cuerpo proveerá información adicional sobre el cuerpo del documento, como el pie normal de una página web. Puede ser usado no solo para el cuerpo sino también para diferentes secciones dentro del cuerpo, otorgando información adicional sobre estas secciones específicas.
* **<article>** Este elemento representa una porción independiente de información relevante (por ejemplo, cada artículo de un periódico o cada entrada de un blog). El elemento <article> puede ser anidado y usado para mostrar una lista dentro de otra lista de ítems relacionados, como comentarios de usuarios en entradas de blogs, por ejemplo.
* **<hgroup>** Este elemento es usado para agrupar elementos H cuando la cabecera tiene múltiples niveles (por ejemplo, una cabecera con título ysubtítulo).
* **<figure>** Este elemento representa una porción independiente de contenido (por ejemplo, imágenes, diagramas o videos) que son referenciadas desde el contenido principal. Esta es información que puede ser removida sin afectar el fluido del resto del contenido.
* **<figcaption>** Este elemento es utilizado para mostrar una leyenda o pequeño texto relacionado con el contenido de un elemento <figure> , como la descripción de una imagen.
* **<mark>** Este elemento resalta un texto que tiene relevancia en una situación en particular o que ha sido mostrado en respuesta de la actividad del usuario.
* **<small>** Este elemento representa contenido al margen, como letra pequeña (por ejemplo, descargos, restricciones legales, declaración de derechos, etc…).
* **<cite>** Este elemento es usado para mostrar el título de un trabajo (libro, película, poema, etc…).
* **<address>** Este elemento encierra información de contacto para un elemento <article> o el documento completo. Es recomendable que sea insertado dentro de un elemento <footer>.
* **<time>** Este elemento se utiliza para mostrar fecha y hora en formatos comprensibles por los usuarios y el navegador. El valor para los usuarios es ubicado entre las etiquetas mientras que el específico para programas y navegadores es incluido como el valor del atributo datetime. Un segundo atributo optativo llamado pubdate es usado para indicar que el valor de datetime es la fecha de publicación.
* **<video>** Para insertar videos, se utiliza el atributo src para indicarle el video que se quiere mostrar. Para que el video pueda ajustarse al tamaño de la ventana, basta con ponerle el width en 100%.

**LASY LOADING (Para cargar elementos (imágenes) solo cuando se necesitan)**

1. Remplazar en las etiquetas “img” que querremos que se carguen dinámicamente la propiedad “src” por “data-src”.
2. Agregar una etiqueta “script” para descargar el blasy.
   * <script type="text/javascript" src="https://cdnjs.cloudflare.com/ajax/libs/blazy/1.8.2/blazy.min.js"></script>
3. Agregar en el javascript de nuestro proyecto el selector para las imágenes.
   * var bLazy = new Blazy({

selector: 'img'

});

**CSS**

* Formas de agregar estilos a un HTML:
  1. En línea por medio del atributo style de una etiqueta HTML.
  2. Etiqueta style dentro de la etiqueta head.
  3. Archivo externo de estilos agregado con la etiqueta link en el head.
* En un archivo CSS se pueden agregar estilos de 4 maneras:
  1. Selector universal (se aplicará a todo el HTML)

\* {}

* 1. Selector de etiqueta

section {} (se aplicará solo a la etiqueta section)

article h3 {} (se aplicará a todos los h3 que estén dentro de todos los article)

* 1. Selector de clase (se aplicará a todas las etiquetas que tengan ese valor en el atributo class)

.clase {}

* 1. Selector de id (se aplicará a la etiqueta que tenga ese valor en el atributo id)

#id {}

* 1. Selectores múltiples (se aplicará a todos los selectores indicados)

.clase1, .clase2 {}

* 1. Selector de atributo

input[type=”text”] {}

* 1. Selectores de pseudoelementos (No todos pueden tener pseudoelementos)

.clase li::before | after | first-line | first-letter {

content: ‘☼’;

font-size: 50px;

line-height: 20px;

}

* 1. Selectores de pseudoclases

**Primer elemento**

a:first-child {

border-left: none;

}

**Liga sin visitar**

a:link {

color: blue;

}

**Liga visitada**

a:visited {

color: red;

}

**Cuando el mouse esta sobre el elemento**

a:hover {

box-shadow: inset 0px 5px 30px red; - Sombra

}

**Cuando un elemento está entre el press y el release del mouse**

a:active {

color: yelow;

}

**Cuando el elemento tiene el foco**

a:focus {

color: gray;

}

**Animaciones**

@keyframes nombre-de-animacion {

0% {

transform: traslateY(5px) rotate(2deg);

}

50% {

transform: traslateY(5px);

}

100% {

transform: traslateY(0px);

}

}

* Propiedades
  + display
    - block - El objeto se posiciona uno sobre otro hacia abajo (valor por default en la mayoría de los objetos).
    - inline-block - Los objetos son posicionados uno al lado del otro respetando el modelo de caja.
    - inline - Los objetos son posicionados uno al lado del otro pero sin respetar el modelo de caja, por lo que si agregamos un elemento con display:block se posicionará dónde estaría antes de aplicar estilos a los objetos inline.
    - flex - Afecta a todos los hijos poniéndolos uno al lado del otro y redimensiona cuando el usuario cambia el tamaño del navegador. Solo a sus hijos, no a sus nietos, etc.
  + (**Propiedades que se pueden utilizar en un contenedor con display: flex**)
    - flex-direction - **Solo sirve después de display: flex.**
      * row - Valor por default, los objetos se ponen de forma horizontal de izquierda a derecha.
      * row-reverse - Los objetos se ponen de forma horizontal de derecha a izquierda.
      * column - Los objetos se ponen de forma vertical de arriba hacia abajo.
      * column-reverse - Los objetos se ponen de forma vertical de abajo hacia arriba.
    - flex-wrap - Como se comportará cuando el usuario modifique el tamaño del navegador. **Solo sirve después de display: flex.**
      * nowrap - Valor por default, los elementos se distribuyen cuando cambia el tamaño del navegador y si el ancho del navegador es más pequeño redimensiona el tamaño de los elementos.
      * wrap - Los elementos no cambian de tamaño cuando el ancho del navegador es menor al de los objetos, sino que los mueve los últimos hacia abajo respetando su tamaño.
      * wrap-reverse - Los elementos no cambian de tamaño cuando el ancho del navegador es menor al de los objetos, sino que los mueve los primeros hacia abajo respetando su tamaño.
    - justify-content - Alineación horizontal de los elementos que contiene. **Solo sirve después de display: flex.**
      * flex-start - Valor por defecto. Los hijos de alinean a la izquierda.
      * flex-end - Los hijos de alinean a la derecha.
      * center - Los hijos se centran.
      * space-around - Los hijos se distribuyen dentro de su contenedor y se ponen automáticamente los márgenes necesarios entre ellos tanto de izquierda como de derecha.
      * space-between - Igual que el space-around pero elimina el margen izquierdo del primer elemento y el margen derecho del último elemento.
    - align-items - La alineación vertical de los elementos que contiene. **Solo sirve después de display: flex.**
      * flex-start - Arriba.
      * flex-end - Abajo.
      * center - Centro.
      * stretch - Los alarga en todo lo alto y/o ancho.
      * baseline - Alinea los elementos al primer texto que contenga sin importar padding o margin.
    - align-content - Cuando hay múltiples filas (flex-wrap:wrap) se establece la alineación vertical de los elementos que contiene entre cada fila. **Solo sirve después de display: flex.**
      * flex-start - Arriba.
      * flex-end - Abajo.
      * center - Centro.
      * stretch - Los alarga en todo lo alto.
      * space-around - Los hijos se distribuyen dentro de su contenedor y se ponen automáticamente los márgenes necesarios entre ellos tanto arriba como abajo.
      * space-between - Igual que el space-around pero elimina el margen superior del primer elemento y el margen inferior del último elemento.
  + (**Propiedades que se pueden utilizar en los elementos hijos de un contenedor con display: flex**)
    - order - En qué orden aparecerán los elementos. **Solo sirve en los elementos hijos de un contenedor con display: flex.**
      * Valor entero.
    - flex-grow - El tamaño horizontal que tendrán los elementos y como crecerán con relación a sus hermanos y el espacio disponible. **Solo sirve en los elementos hijos de un contenedor con display: flex.**
      * Valor entero, 0 si no se modificará su tamaño aunque se agrande el tamaño de la ventana.
    - flex-shrink - El tamaño vertical que tendrán los elementos y como crecerán con relación a sus hermanos y el espacio disponible. **Solo sirve en los elementos hijos de un contenedor con display: flex.**
      * Valor entero, 0 si no se modificará su tamaño aunque se achique el tamaño de la ventana.
    - align-self - La alineación vertical de ese solo elemento.
      * flex-start - Arriba.
      * flex-end - Abajo.
      * center - Centro.
      * stretch - Los alarga en todo lo alto.
      * baseline - Alinea los elementos al primer texto que contenga sin importar padding o margin.
  + position
    - static - Posición por default. Los elementos se muestran en el orden del flujo del documento.
    - relative - El elemento se posiciona relativo a su posición normal. Si se utilizan propiedades como top, bottom, left y right.
    - absolute - El elemento se posiciona relativo a su primer ancestro no static.
    - fixed - El elemento se sitúa relativo a la ventana del explorador.
    - inherit - Toma el valor de su padre.
  + z-index - Posición de los elementos en plano de profundidad cuando tienen position: absolute.
    - Valor entero.
  + float
    - none - Respeta la posición original del flujo del documento.
    - left - Envía el objeto a la izquierda y los demás objetos hermanos se situarán a su derecha.
    - right - Envía el objeto a la derecha y los demás objetos hermanos se situarán a su izquierda.
    - inherit - Toma el valor de su padre.
  + background
    - #ffffff;
    - linear-gradient(to left, <primer color>, <segundo color>);
  + background-image - Imagen de fondo.
    - none;
    - background-image: url(‘ruta’);
  + background-size - El tamaño que se aplicará al
    - auto - Tamaño original de la imagen de fondo.
    - <x>px
    - <x>%
    - cover - Automáticamente se dimensiona para abarcar todo el espacio del contenedor.
    - contein - Tamaño más grande sin deformar la imagen que quepa en su contenedor.
  + background-position - Donde se colocará el fondo.
    - ejeX ejeY.

Ejemplo: background-position: left top;

* + background-repeat
    - repeat;
    - repeat-x;
    - repeat-y
    - no-repeat;
  + opacity - Transparencia que tendrá el elemento sobre el fondo o su contenedor.
    - Valor decimal 0.0 (completamente transparente) - 1.0 (completamente sólido)
  + margin - Margen alrededor del objeto con respecto a otros objetos.
  + padding - Margen al interior del objeto con respecto a los hijos del objeto.
  + border-radius - Permite establecer bordes redondeados a una caja. Se establece en pixeles. Si las 4 esquinas tendrán el mismo valor solo se escribe el valor como por ejemplo 8px. Si queremos que los bordes tengan diferente tamaño se deben escribir los 4. El orden es superior izquierdo, inferior izquierdo, inferior derecho y superior derecho.
    - border-radius: 8px;
    - border-radius: 100%; - Círculo
  + text-decoration - Tipo de subrayado en texto.
    - underline - Subrayado
    - overline - Subrayado arriba.
    - line-through - Tachado.
    - none - Nada.
  + font-weight - Grosor del texto.
    - normal ;
    - bold; - Negrita.
  + font-style - Estilo del texto.
    - normal;
    - italic; - Muy delgada y cursiva.
    - oblique; - Normal y cursiva.
  + list-style - Estilo en las listas numeradas y no numeradas.
    - none; - Le quita el puntito o número de la izquierda.
  + cursor
    - none - Desaparece cursor.
    - pointer - Dedo indice apuntando.
    - progress - Flecha con ruedita moviéndose de espera.
    - wait - Ruedita moviéndose de espera.
    - url();
    - zoom-in - Lupita con +;
    - zoom-out - Lupita con -;
  + box-shadow - Pone sombra al elemento.
    - inset ejeX ejeY difuminado color;
    - inset 0 -5px 0 rgba(0, 0, 0, .3);
  + transform
    - scale(.9); - Cambia el tamaño del elemento.
    - translateY(<x>px); - El elemento se mueve en el eje Y los pixeles que se indican.
    - translateX(<x>px); - El elemento se mueve en el eje X los pixeles que se indican.
    - rotate(<x>deg); - El elemento rota los grados indicados;
    - skew(<x> deg); - El elemento se tuerce los grados indicados;
  + transition
    - .2s; - El cambio de tamaño lo realizará en .2 segundos.
  + animation
    - tiempo-en-segundos nombre-de-animacion tipo; - Se crea una animación para utilizarla con @keyframes en css. Ejemplo: (1s anima infinite)
  + mask-image - Poner una imagen de máscara sobre el elemento.
    - url(‘<imagen>’);
  + height, min-height, max-height, width, etc.
    - <x>px;
    - <x>%;
    - 0-100<vh> - viewport height que es el alto de la ventana del explorador.
    - 0-100<vw> - viewport width que es el ancho de la ventana del explorador.
  + overflow - Que pasará con el área que se desborda de un contenedor.
    - hidden - No mostrará los elementos o pedazos de los elementos fuera del contenedor.
    - auto - Habrá scroll para que se pueda visualizar el contenido que se desborda.
  + zoom - Hace más grande el elemento al que se le aplica.
    - 1-∞
  + box-sizing - Para indicar el tamaño del objeto si contemplará el borde y el padding o no.
    - border-box - Toma en cuenta el borde y el padding.
    - content-box - No toma en cuenta el borde ni el padding (opción por default).
* Videos
  + Para insertar videos que tenemos descargados en alguna ruta, basta con agregarlos en una etiqueta <video> con src de la ruta de nuestro video. Debemos ponerle un width de 100% para que pueda tener todo el ancho de su contenedor cuando el usuario modifique el tamaño de la pantalla o lo vea en móviles.
  + Para insertar videos de youtube tenemos que copiar la etiqueta <iframe> que nos proporciona youtube. Esta etiqueta debemos tenerla en un contenedor como puede ser un div. Al div se le debe de poner un estilo como este:

position: relative;

padding-top: 56.25%; (Sale de la proporción del video, 315px es un 56.25% de 560px)

y al iframe un estilo como este:

position: absolute;

left: 0;

right: 0;

top: 0;

bottom: 0;

width: 100%;

height: 100%;

* Herramientas
  + ColorPick Eyedropper (Para instalar en Chrome y poder ver el color de cierto elemento).
  + http://caniuse.com (Manual super actualizado y verificar compatibilidad de cierto elemento en navegadores).
  + [www.w3schools.com](http://www.w3schools.com) (Manual).
  + <https://icomoon.io> (Iconos gratis)
    - Se copia la carpeta “fonts” al proyecto a la altura de css.
    - Se copia el contenido del archivo “style.css” al inicio de nuestro archivo css del proyecto donde utilizaremos los iconos y verificamos que las rutas de los url estén correctas.
    - Agregamos una etiqueta span con la clase del estilo en nuestro HTML.
  + <http://dinbror.dk/blazy/> (Documentación de blasy para lazy loading de imagenes)
  + [www.cdnjs.com](http://www.cdnjs.com) (Descargar librerías js como blasy para lazy loading)
  + <https://developers.google.com/speed/pagespeed/> (Analizar el performance en producción)
  + <https://tinypng.com/> (Comprimir las imágenes para mejorar performance)
  + <https://jonassebastianohlsson.com/criticalpathcssgenerator/> (Genera css para ponerlo en etiqueta style dentro de head en lugar de cargar archivo css y optimizar página).
  + <http://csscompressor.com/> (Comprimir o minificar css en una sola línea para que este en la etiqueta de stile dentro del head en una sola linea).

**OPTIMIZAR SITIO WEB**

* Analizar performance con pagespeed.
* Comprimir imágenes con tinypng.
* Cambiar etiqueta <link> que tenemos en el <head> con nuestro css al final de la etiqueta <body> después de nuestro javascript.
* Agregar un estilo default para lo primero que se carga cuando el usuario entra a nuestro sitio en una etiqueta <style> dentro de nuestro <head> haciendo critical generator y después minimizando el css resultante en una sola línea con css compressor.
* Comprimir nuestro código css y pegarlo en un nuevo archivo .css.
* Modificar llamada a nuestro css para que cargue el archivo con css comprimido.