HTML

“Hyper Text Markup Language”

- Lenguaje de marcado de hipertexto.
- Compuesto de **ETIQUETAS** y **ATRIBUTOS**, quienes a su vez forman **ELEMENTOS.**

SINTAXIS

Etiqueta de apertura

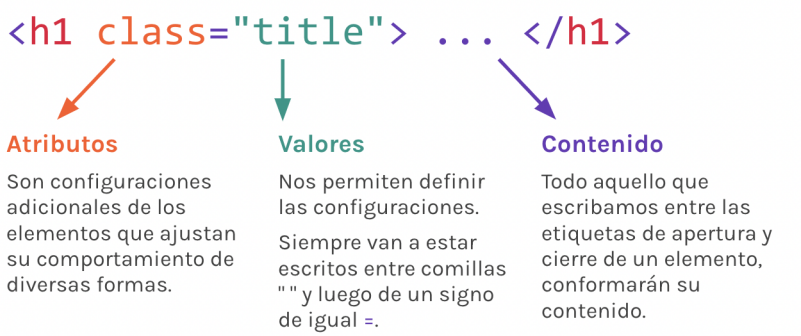
Indica el comienzo de un elemento, siempre debe iniciar con el símbolo de menor < y terminar con el signo de mayor >.

Dentro debe ir el nombre de la etiqueta.

Etiqueta de clausura

Indica el final de un elemento, siempre debe iniciar con el símbolo de menor seguido de la barra diagonal </ y terminar con el signo de mayor >.

Dentro debe ir el nombre de la etiqueta.



¿Cómo se compone un elemento?

1. Una etiqueta de **apertura**
   a. **Opcionalmente** uno o más **atributos**
2. El **contenido**
3. Una etiqueta de **cierre**. Algunos elementos no la llevan.

ENCABEZADOS, PÁRRAFOS Y LISTAS

ENCABEZADOS

Las etiquetas **<h…>** implementan seis niveles de encabezado del documento.
<h1> es el más importante y **<h6>** el menos importante.
→ Un elemento de encabezado describe brevemente el tema de la sección que presenta.

PÁRRAFOS

Las etiquetas **<p>** nos permiten distribuir el texto en párrafos. Podemos usar tantas como necesitemos.

- Éste es un párrafo, puede tener todo el texto que necesitemos.
- El navegador se encargará de agregar espacio vertical entre cada uno de los párrafos que escribamos.

LISTAS

Las listas se conforman de **etiquetas combinadas.**
Nos permiten listar elementos con las tradicionales **viñetas** o de manera ordenada, con algún estilo de **numeración.**

Lista ordenada (ol)

= Nos permiten **enumerar** ítems de manera consecutiva. Por defecto van a empezar en el número 1 y se irán incrementando con cada ítem nuevo.

Atributo → start

Nos permite definir dónde va a empezar nuestra numeración

Atributo → type

Nos permite cambiar el tipo de viñeta de la lista. (Valor: 1 Numérica. A Alfabética, I Numérica romana)

Lista desordenada (ul)

Las **listas desordenadas** nos permiten listar ítems.
Por defecto, van a generar una viñeta tipo “bolita” por cada ítem nuevo que se agregue.

Atributo → type

Nos permite cambiar el tipo de viñeta de la lista.

Listas ANIDADAS

Las listas **anidadas** nos permiten crear varios niveles de jerarquía y organización.
Las podemos anidar como deseemos y generar los niveles que necesitemos.

- Dentro de un **<ol>** o un **<ul>** sólo pueden haber elementos **<li>**.
- Las listas se pueden anidar agregando un **<ul>** u **<ol>** dentro de un **<li>**.

RUTAS, HIPERVÍNCULOS E IMÁGENES

RUTAS

= Es una dirección o camino que permite al navegador encontrar un recurso.

Un recurso puede ser otra página web, una imagen, un video o cualquier otro tipo de archivo.

Tenemos dos tipos de rutas:

RUTA **ABSOLUTA**

Puedo acceder a un recurso sin importar la carpeta en donde me encuentro trabajando.

- Es decir.. puedo acceder a ella sin importar el lugar en el que esté navegando en ese momento.

- La ruta de acceso al recurso es siempre la misma.

RUTA **RELATIVA**

Representa solo una parte de la ruta considerando la carpeta actual donde me encuentro trabajando.

- Es una ruta relativa a mi posición actual.

Cuando indiquemos la ubicación de un recurso ***dentro de nuestros proyectos*** será necesario hacerlo mediante ***rutas relativas***, es decir, teniendo en cuenta la ubicación del archivo desde el que estoy partiendo.

- Para acceder a una imagen que se encuentra en una **carpeta inferior** a la del archivo original deberíamos descender un nivel. Eso lo hacemos con la barra /.

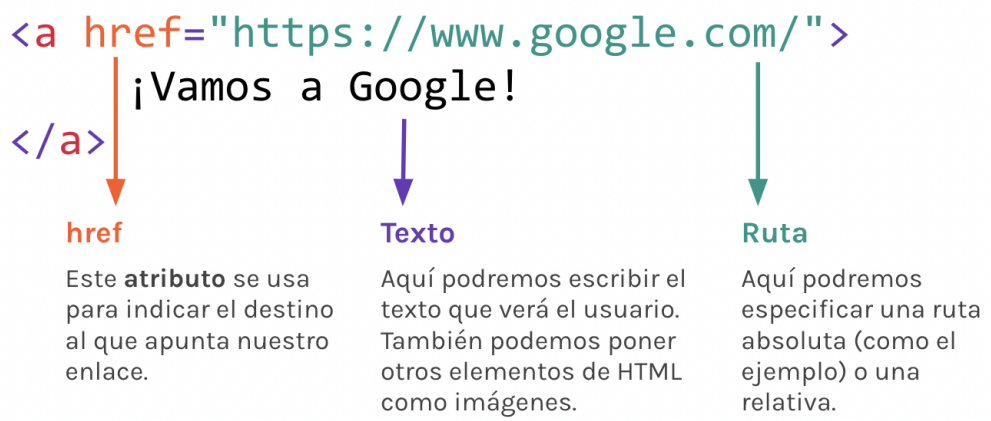
  → La barra / indica que nos movemos **al siguiente nivel descendente** en el árbol de carpetas.

- Para ir un **nivel hacia atrás** en el árbol de carpetas usamos dos puntos seguidos ../.

Etiquetas que usan rutas

ENLACES

A través de la etiqueta **<a>** vamos a poder crear nuestros enlaces. Ésta es una etiqueta de apertura y cierre: **<a>**.



TIPOS DE ENLACES

EXTERNOS

Sus rutas están fuera de nuestro sitio y son siempre rutas absolutas.

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

LOCALES

Sus rutas están dentro de nuestro sitio y deben ser siempre rutas relativas.

```
<a href="./index.html">Inicio</a>
```

CORREO

Al hacer click en ellos abren el programa de correo predeterminado para enviar un correo a la dirección de email.

```
<a href="mailto:user@server.com">Dejanos tu mensaje</a>
```

TELÉFONO

Navegando desde un celular al tocar el enlace se iniciará una llamada a ese número.

```
<a href="tel:1145678900">¡Llamanos!</a>
```

IMAGENES

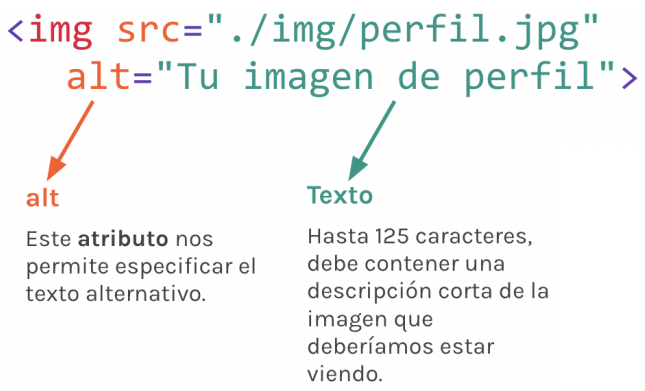
Dentro de nuestro documento html podemos agregar imágenes a través de la etiqueta **<img>.**

Esta etiqueta permite **invocar** las imágenes, es decir, hacer referencia al lugar donde están alojadas para que aparezcan en el navegador.

- Las imágenes no llevan etiqueta de cierre (</img>).

El **texto alternativo** nos permite describir una imagen. Se muestra cuando la imagen no carga por alguna razón o para las personas que usen lectores de pantalla (ej: no videntes).

- el atributo **alt** también sirve para que los buscadores puedan entender nuestras imágenes y mejorar el posicionamiento de nuestro sitio en las búsquedas



ETIQUETAS IFRAME

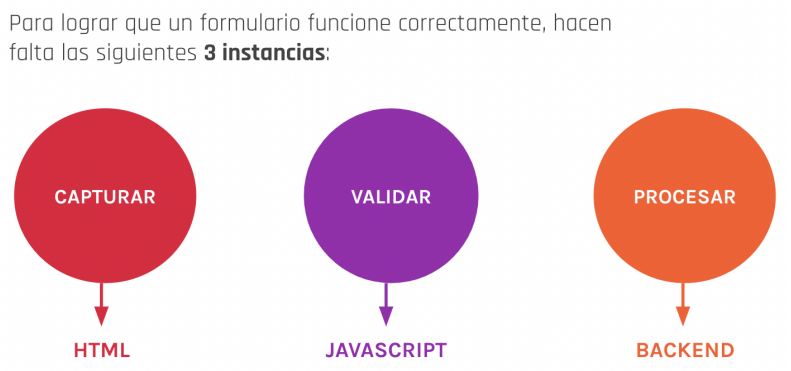
A través de la **etiqueta iframe** podemos insertar contenido de otros sitios web dentro de nuestro proyecto.
La etiqueta iframe creará un marco dentro del cual se verá el contenido literal del sitio web de destino.

- Se utiliza mayormente para colocar los players de contenido multimedia alojado en páginas específicas: youtube, vimeo, deezer, twitter, etc.

FORMULARIOS

Un formulario es un **sistema** de etiquetas combinadas para **obtener datos del usuario.**

Necesita de lenguajes adicionales para su funcionamiento completo.



ETIQUETAS PRINCIPALES

<form>

Es la etiqueta que le permite al formulario funcionar, por eso es la etiqueta **más importante**. **TODAS** las demás etiquetas de un formulario van dentro de *<form> </form>*

- El atributo ***action*** indica a donde redireccionará el formulario luego del envío.

- El atributo ***method*** indica el método http* que se utilizará el formulario para enviar la información.

<input>

La etiqueta *<input>* genera los campos para que el usuario **complete** con información.

- Cambiando el **valor** del atributo ***type*** obtenemos distintos tipos de campos.

- El atributo ***name*** identifica al campo. ***name*** es **fundamental** para procesar la información del mismo. Si lo olvidamos la información de ese campo **no se enviará.**

<label>

La etiqueta ***<label>*** permite colocar una descripción junto al campo. El texto se muestra en el navegador e indica la información que tiene que completar el usuario.

- En ocasiones la etiqueta ***<label>*** puede reemplazarse por el atributo ***placeholder*** colocando la descripción del campo como valor del atributo.

<button>

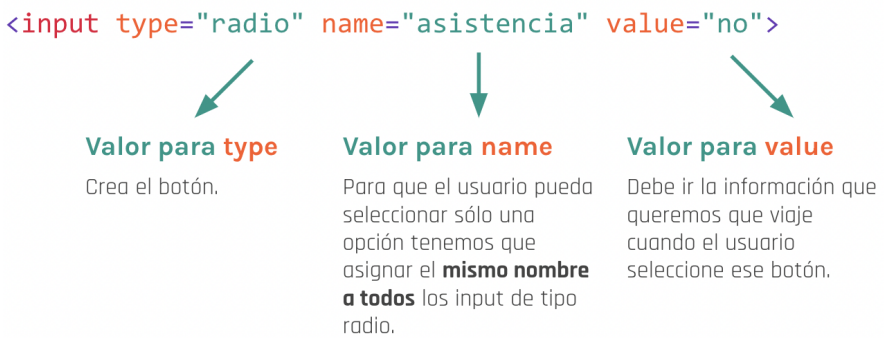
Con la etiqueta **<button>** creamos botones.

Dependiendo del valor que asignamos al atributo ***type*** vamos a poder **enviar**, **borrar** o generar otro tipo de **acción**.

RADIO BUTTONS Y CHECKBOX

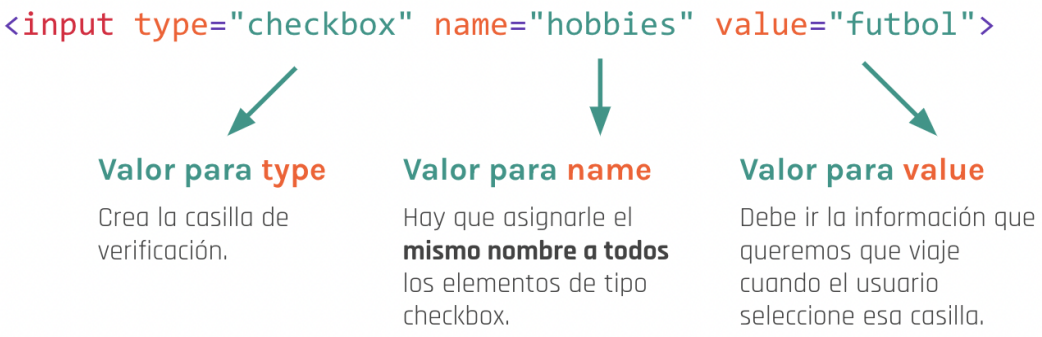
TYPE RADIO

Seteando el atributo ***type*** con el valor ***radio,*** generamos un botón de única opción conocido como **radio-button.**



TYPE CHECKBOX

Seteando el atributo ***type*** con el valor ***checkbox,*** generamos una casilla de verificación para que el usuario seleccione una o más opciones conocidas como **check boxes.**



"El atrib

preseleccionado por defecto.”

TEXT AREAS Y SELECT

TEXT AREAS

La etiqueta **<textarea>** crea un campo para escribir varias líneas de texto.

```
<textarea name="mensaje"></textarea>
```

SELECT

Las etiquetas combinadas ***<select>*** y ***<option>*** crean un **combobox** ó **dropdown** de selección única.

Permite agrupar muchas opciones en un solo control. La etiqueta ***<select>*** será el contenedor de las etiquetas ***<option>*.**

```
<select name="pais">
    <option value="Ar">Argentina</option>
    <option value="Br">Brasil</option>
    <option value="Ve">Venezuela</option>
</select>
```

CSS

= Cascading Style Sheets

- Hojas de **estilo** en cascada. **[Estilo/ Apariencia/ Diseño]**

Compuestas de REGLAS, SELECTORES y DECLARACIONES.

Las hojas de estilo sirven para **estilizar** nuestro contenido **HTML**. Con **CSS** podemos cambiar colores, fondos, tipografías, anchos, altos, etc.

Contamos con **3 métodos** para vincular nuestros archivos CSS con el documento HTML:

- **VINCULACIÓN INTERNA**

  A través de la etiqueta **<style>** dentro del **<head>.**

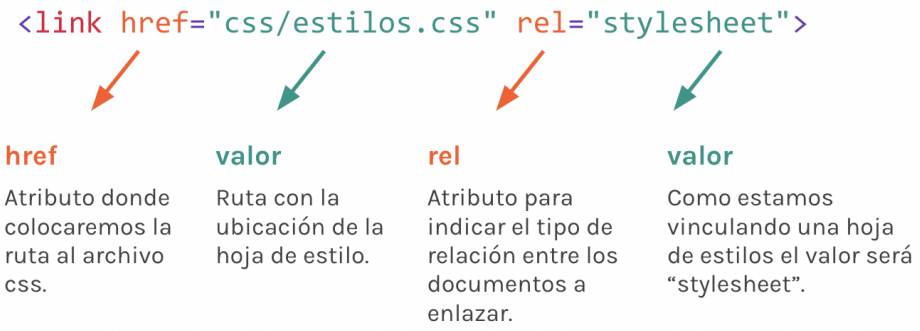
- **VINCULACIÓN EN LÍNEA**

  Usando el atributo **style** en cada elemento de nuestro HTML.

Ambas reemplazadas por..

VINCULACIÓN EXTERNA

Escribiendo todos nuestros estilos **en un archivo CSS** y vinculándolo al archivo **HTML** usando la etiqueta **<link>** dentro de **<head>** de nuestro documento.



REGLAS Y SELECTORES

REGLAS

Las **reglas de CSS** son el bloque de código CSS conformado por **un selector, propiedades** y **valores** que permiten aplicar estilo al HTML

- selector : A **quién** quiero agregar o modificar un estilo o su apariencia.

- Las **llaves** permiten agrupar todas las **declaraciones** que queramos indicar en nuestra regla de estilo**.** → Todas las propiedades y valores que queramos agregar o modificar deben estar contenidas entre llaves **{ }**. Sí, puedo escribir muchas declaraciones dentro de una misma

regla de estilo.

- propiedad: Son los nombres de los estilos que queremos agregar o modificar. Definimos la propiedad **que** queremos agregar o modificar
- valor; Asignamos el valor que queremos o necesitamos en la propiedad seleccionada. Acá ponemos el nuevo **valor** que queremos asignar a la propiedad.

SELECTORES

= Los **selectores** permiten "seleccionar" los **elementos HTML** que queramos modificar y así aplicarles estilo.

SELECTORES **DE ETIQUETA**

El **selector de etiqueta** selecciona todas las etiquetas con su mismo nombre dentro del documento HTML.

Sintaxis css: Para seleccionarlo desde css usamos el **nombre de la ETIQUETA.**
Para distinguir etiquetas que son del mismo tipo podemos usar los atributos de HTML **class** y **id.**

SELECTORES **DE CLASE**

A través del atributo class asignamos una clase al elemento HTML. Los elementos HTML pueden tener **múltiples clases.** - Cada una estará **separada por un espacio**.

Su uso es muy frecuente y recomendado ya que pueden colocarse múltiples nombres de clase y usarse de forma combinada en los selectores de css.

Sintaxis del selector de css: Para seleccionarlo desde css usamos el punto (**.**) seguido del **valor de la clase.**

El **selector de clase** selecciona a todos los elementos HTML que contengan su mismo nombre dentro del atributo **"class"**.

SELECTORES **DE ID**

A través del atributo id asignamos un nombre único al elemento HTML. Los nombres de id no deben repetirse a lo largo del documento HTML.

Su uso es muy específico, puntual y limitado a un sólo término.

Sintaxis del selector css: Para seleccionarlo desde css usamos el **#** seguido del **valor del id.**

El **selector de id** selecciona al elemento HTML que contenga su mismo nombre dentro del atributo **"id"** en el elemento HTML.

SELECTORES **COMBINADOS**

Afectan a los elementos HTML que cumplan con **todas** las condiciones que establezca el selector.

Sintaxis del selector css: Para llamarlos desde css escribimos **un selector al lado del otro (sin espacios)** cada uno con la sintaxis que corresponda.

Ejemplo: h2.subtitulo

SELECTORES **DESCENDENTES**

Sirven para agregar especificidad. El espacio vacío entre cada selector indica descendencia.

Sintaxis del selector css: Para llamarlos desde css escribimos los **selectores separados por un espacio.** El espacio indica la relación descendente.

Ejemplo: .lista li

CASCADA Y ESPECIFICIDAD

Se llaman "hojas de estilo **en cascada**" porque si una regla se repite más abajo en el archivo el lenguaje tomará como válido a las propiedades y valores de la última regla. → El último valor es el que prevalece.

Pero…

CSS **siempre** va a priorizar a los selectores **más específicos** para aplicar estilo. → **Mayor especificidad** en el selector **es más importante** que el **orden de lectura en cascada.**

PROPIEDAD DISPLAY Y MODELO DE CAJA

PROPIEDAD DISPLAY

La **propiedad display** especifica el comportamiento de visualización de un elemento HTML.

Etiquetas de **bloque**

** **
Las etiquetas de bloque ocupan el 100% del ancho del sitio y generan un salto de línea.
El valor por defecto de la propiedad es:
div { display: **block;** }

Etiquetas de **línea**

** **

Las etiquetas de línea no cambian la distribución del sitio y ocupan solo el ancho de su contenido. El valor por defecto de la propiedad es:
span { display: **inline;** }

block

Define un elemento con comportamiento de bloque.

Recibe “fácilmente” propiedades del modelo de caja.

inline

Define un elemento con comportamiento de línea.

No recibe algunas propiedades del modelo de caja.

Tipos de Elementos

display: inline-block;

inline-block

Define un elemento con comportamiento de **semibloque**.

Recibe “fácilmente” propiedades del modelo de caja. Comparte también propiedades de elemento de línea.

display: none;

none

Oculta a un elemento en la vista.

No lo elimina de la estructura de HTML.
Solo es un efecto visual.

MODELO DE CAJA

En HTML todos los elementos se representan mediante **cajas**.

Se llama modelo de caja al conjunto de propiedades de CSS que permiten manipular el alto, ancho, relleno, borde y margen de las cajas que componen un documento HTML

Las propiedades asociadas al modelo de caja sólo aplican a las **etiquetas de bloque.**

Propiedad **WIDTH**

Si un elemento no tiene declarado **“width”,** el mismo será igual al 100% de su contenedor padre, siempre y cuando sea un **elemento de bloque**.

Propiedad **HEIGHT**

Si un elemento no tiene declarado **“height”,** el mismo será igual a la altura que le proporcione su contenido interno, sea un **elemento de bloque ó un elemento de línea.**

Propiedad **PADDING**

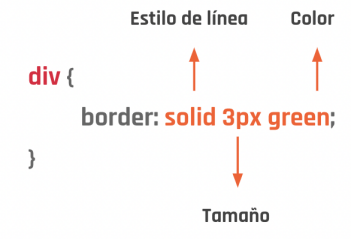
Hace referencia al **margen interior** del elemento. Para asignarle **valor** a esta propiedad lo podemos hacer usando la medida píxeles (px), indicando…

```
div {
```

```
        padding: 10px 20px 30px 40px;
    //padding: top  right  bottom left;
}
```

Propiedad **BORDER**

Hace referencia al borde del elemento. Para asignarle **valor** a esta propiedad, lo hacemos definiendo el **estilo de línea**, su **tamaño** y su **color**. El estilo de línea puede ser **solid, dotted, dashed o double.**



Hace referencia al **margen** del elemento. Sirve para **separar una caja de la otra**. Para asignarle valor a esta propiedad podemos hacerlo de la siguiente manera:

El ancho total de la caja es igual a la suma de su **width**, **padding** y **border**.

Propiedad **BOX-SIZING**

Esta propiedad permite que el modelo de caja sea más fácil de usar porque descuenta automáticamente del ancho y alto lo que agregamos en relleno y borde. El único valor que se sigue sumando es el del **margin**.

Para usar la propiedad **box-sizing: border-box** en todo mi documento llamamos al **selector universal *{ }**

FLEXBOX

Flexbox o “sistema de cajas flexibles” es un set de **propiedades** que nos entrega **CSS** para **construir** un sitio web utilizando una **estructura** de **filas** y **columnas**.

Nos ayudará a que los **elementos de bloque** compartan una misma línea.

Estructura basica

Flexbox propone una estructura basada en la creación de un **contenedor padre (*Flex-container*)** que administra la ubicación de sus **elementos hijos (*Flex-items*).**

Para organizar los elementos dentro del contenedor Flexbox trabaja con dos ejes:

- **Main axis. (row → horizontal, column → vertical)**
- **Cross axis. (row →vertical , column → horizontal)**

La posición de los ejes dependerá de la dirección en la que se ubican los elementos mediante la propiedad ***flex-direction***.

El **primer paso** será definir al contenedor padre como un un **contenedor flexible.**

El **segundo paso** será hacer que el container respete los anchos de sus hijos. En caso de no entrar los elementos hijos crearán más filas. → Si no declaramos **explícitamente** el valor de la propiedad ***flex-wrap*** por defecto será ***nowrap*** y los elementos dentro del contenedor padre se **ubicarán todos** en **una** misma **fila** haciéndose **cada vez más angostos** si el contenedor se achica o aparecen más elementos**.**
Para crear filas o columnas tenemos la propiedad ***flex-direction*** cuyo valor por defecto será ***row.***

La propiedad estará aplicada al contenedor padre aún cuando no la declaremos explícitamente.

Para crear columnas tendremos que declarar explícitamente la propiedad ***flex-direction*** con valor ***column*** *ó* ***column-reverse.***

Para alinear contenido sobre el **main axis** contamos con la propiedad ***justify-content.*** Su valor default implícito será ***flex-start*** pero también recibe otros valores.

Para alinear contenido sobre el **cross axis** contamos con la propiedad ***align-items*** cuyo valor por default es ***stretch*** pero que además puede tomar otros valores,

FLEX-ITEMS

Flexbox complementa sus funcionalidades con un **set de propiedades** que permiten **gestionar** a los ***flex-items*** **directamente.**

Recordemos que siempre será dentro de un ***flex-container.***
La propiedad ***order*** permite modificar el orden natural de los elementos dentro del ***flex-container.***

Por defecto todos los elementos tienen orden en valor 0.

Para que la propiedad funcione correctamente todos los elementos deben estar identificados y tener asignado un orden de manera explícita.

También podemos alinear los elementos individualmente dentro del ***flex-container*** utilizando la propiedad ***align-self.***

RESPONSIVE WEB DESIGN

Responsive web design o diseño web adaptativo consiste en crear páginas web que se vean bien en todos los dispositivos.

Una página web diseñada para adaptarse modificará la ubicación y visibilidad de sus elementos “automáticamente” de acuerdo a los diferentes tamaños de pantalla y/o ventanas gráficas (viewports) en dónde debe mostrarse.

- ¿Qué es el diseño web adaptativo?

El diseño web adaptativo es la combinación de etiquetas HTML y reglas de CSS que permiten cambiar el tamaño, ocultar, reducir o agrandar “automáticamente” un sitio web y/o sus elementos para que

se vea bien en todos los dispositivos (computadoras de escritorio, tabletas y teléfonos).

Viewport: Representa a la sección del navegador en donde se muestran los documentos HTML.