**ELEMENTOS DE BLOQUE**

Ocupa 100% del ancho de la pagina (width)

hace salto de linea

Ejemplo: *<div>*ocupa todo el ancho, tira abajo etiqueta siguiente *</div>*

**ELEMENTOS DE LÍNEA**

<span> se suelen usar</span> <span> para contener porciones de texto </span>

determinan un estilo independiente al bloque de código al que pertenecen.

**DISPLAY**

Cambia la posición del elemento seleccionado

Los valores que recibe son block, inline, inline-block y none.

*inline:*

elemento comportamiento en línea,

no recibe algunas propiedades del modelo de caja

*Inline-block*

semi bloque

recibe propiedades de caja y en linea

*block:*

lo mismo con comportamiento en caja

recibe todas las propiedades de modelo de caja

*none:*

oculta un nivel

desaparece de la vista

**LISTAS ORDENADAS**

numera ítems de manera consecutiva

<Ol>

<li> Nombre completo:</li> //1.Nombre completo:

<li> Año:</li> //2.Año:

</Ol>

**atributos:**

type: cambia el tipo de la viñeta de la lista.

pueden ser:

“1”- numérico

“A”-alfabético

“I”-numérica romana

<ol type="1"> ... </ol>

start= con qué número empieza

puede ser positivo o negativo

<ol start="20"> ... </ol>

**LISTAS DESORDENADAS**

<ul>

<li> Nombre completo:</li>

<li> Año:</li>

</ul>

**atributos:**

type= tipo de viñeta

3 posibles valores.

* Disc
  + Circle
    - Square, cuadrado

<ul type="disc"> ... </ul>

**LISTAS ANIDADAS**

Crea varios niveles de jerarquía y organización

Las listas se pueden anidar agregando un <ol> o <ul> dentro de un <li>.

<ul>

<li> recordar para viaje:

<ol>

<li>Pasajes</li>

<li>DNI:</li>

</ol>

</li>

</ul>

**RUTAS**

Dirección o camino (también conocido como path)

permite al navegador encontrar un recurso como página web, una imagen, un video o cualquier otro tipo de archivo.

En el caso de los enlaces, la ruta indica la dirección a la que tiene que llevarnos el navegador cuando pulsamos sobre él.

**ruta absoluta:**

[www.google.com.ar](http://www.google.com.ar)

**ruta relativa:**

../img/wallpaper.png

Si una imagen se encuentra en la misma carpeta que el archivo original, la ruta debe hacer referencia al nombre del otro archivo ya que ambos archivos comparten el mismo lugar.

Si una imagen que se encuentra en una carpeta inferior a la del archivo original agregar la barra /antes del nombre de la carpeta y el archivo

**HIPERVÍNCULOS**

<a>

**atributo:**

href=” “ destino que va el enlace

**tipo externo:**

<a href="www.google.com.ar">Vamos a Google</a>

**tipo local:**

rutas están dentro del sitio.

Se recomienda que sean relativas siempre que sea posible.

**<a href="contacto.html">Contacto</a>**

**correo:**

se abrirá el programa de correo que tengamos como predeterminado para enviar un correo a esa dirección de e-mail

<a href="mailto:nadianahir3264@gmail.com">Mandame un mail</a>

**llamadas:**

inicia llamada o copia el numero para marcar

<a href="tel:2664261366">Llámame</a>

**IMAGEN:**

<img src="/img/logo batman.png" alt="logo batman">

**atributo:**

**src=** destino donde esta la imagen

**alt=** descripción corta de la imagen, hasta 125 caracteres

**ancho:**

en píxeles(sólo va el número) o porcentajes

<img widht="100” src="/img/logo batman.png" alt="logo batman">

**largo:**

en píxeles(solo va el número) o porcentajes

<img height="60” src="/img/logo batman.png" alt="logo batman">

**CSS**

vincular archivo en html en el head

rel=relación entre los documentos a enlazar, siempre es la misma

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

**REGLAS:**

Instrucción a elemento determinado para indicarle un estilo

img --selector:elemento que aplica la regla--{

height--propiedad del elemento a modificar--: 30rem;--valor:comportamiento de la propiedad--

}

**TIPOS DE SELECTORES**

**id**

atrapa elemento HTML que tenga asignado el atributo id con el valor correspondiente.

únicos, en css se invoca con # seguido del nombre del id

**class**

Posible asignar muchas clases a un mismo elemento.

Para llamarlo desde el CSS se usa el . seguido del nombre de la clase.

**etiqueta:**

valido para todo el HTML que la contenga

/\* todas etiquetas p \*/

p{

color: blue;

}

<p id=”principal”>Hola, hola</p>

#principal{

color: yellow

}

/\* ej clase="rechazado" es '.' en css,

varias etiquetas esas propiedades \*/

ul li.rechazado{

color:red;

}

/\*id="negro" pone en ese color

el elemento de esa lista \*/

ol#negro li{

color:red

}

/\* combinación de selectores y especificidad \*/

ol#negro li.primero {

color:blanchedalmond;

}

**SELECTORES COMBINADOS**

combinación de todos los anteriores, también podemos usar varios selectores del mismo tipo.

**SELECTORES DESCENDIENTES**

Sirven para agregar especificidad

Para llamarlos desde el CSS se escriben los selectores separados por un espacio (el de la derecha siempre será el que está dentro del de la izquierda).

**FUENTES**

Como valor recibe el nombre de la tipografía que queramos usar.

**atributos:**

font-size en valores: px, em y rem.

font-style: italic, normal y oblique.

font-weight:bold, lighter, normal,

Puede recibir un valor numérico que se irá incrementando de 100 en 100.

text-align: center, left, right, inherit y justify. El valor por defecto es left.

formato y valor es el mismo no varia si es fuente externa

text-decoration: valores line-through, underline, overline y none.

line-height: el alto de cada línea de textos o interlineado.

Recibe un valor numérico acompañado de la unidad de medida con el tamaño de la letra

sans-serif: sin detalles

body{

font-family: Arial, Helvetica, sans-serif;

}

fuentes de google descargar links y poner en index, copiar propiedad y pegar en el css

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

<link href="https://fonts.googleapis.com/css2?family=Fredoka:wght@300;700&display=swap"

rel="stylesheet">

h{

font-weight: 300;

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

}

**FUENTES PROPIAS:**

descargar fuentes hacer una carpeta con el nombre de "fonts"

linkear en css

Alojar fuentes localmente significa que tenemos los archivos de fuentes en propio servidor. Para ello vamos a hacer uso de la directiva css @font-face.

La regla @font-face permite cargar fuentes personalizadas en una página web.

Una vez que se agrega a una hoja de estilo, la regla indica al navegador que descargue la fuente desde donde está alojada y luego la muestre como se especifica en el CSS.

@font-face {

font-family: Roboto, sans-serif;

src: url('../fonts/Robot/Robot-Black.ttf');

}

**COLOR**

Digitamos intensidad de luz: rojo, verde o azul.

4 tipos de colores

no usar \_ problema con algunos navegadores

li {

color: red;

color: #ff0000; hexadecimal 2 valors rojo, 2 verde, 2 azul

color: rgb(255, 0, 0);rojo, verde y azul

color: rgba(255, 0, 0, 0.5); especifica color con valor entre 0 a 255, especifica su transparencia

}

buena practica, ordenar alfabeticamente el codigo

body{

font-family: Arial, Helvetica, sans-serif;

font-size: 24px;

font-style: oblique;

font-weight:bold;

line-height: 36px;

text-align: center;

text-decoration: underline;

text-transform: capitalize;

}

**FONDOS**

**Background**

Atributos:

* background-color: cualquier formato( rgb, rgba, hexadecimal
* background-image: imagen de fondo al elemento, definiendo la ruta a través de la URL.
* background-repeat: si se va a repetir la imagen. Valores: repeat, no repeat, repeat-x, repeat-y, round y space.
* background-position: mover la imagen dentro del elemento y decidir dónde colocarla.

Valores en pixeles y porcentajes, así como también right, bottom, left, etcétera. Puede tener uno o dos valores. El primero para especificar la posición en el eje x y el segundo, la posición en el eje y

* background-attachment: si la imagen de fondo se va a mover junto con la página al hacer scroll o si se va a quedar fija. Recibe como valor: fixed, scroll, inherit e initial.
* background-size: establece el tamaño de la imagen de fondo. Valores: contain, cover, inherit; también tamaños en pixeles y porcentajes, indicando con el primer valor el ancho y con el segundo, el alto.

**ICONOS**

Disponibles en: Font Awesome, Iconic y Material Design.

Incluir el enlace a nuestros iconos dentro del <head> del HTML:

<link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.15.2/css/all.css">

Insertar el icono seleccionado en el lugar del <body> que más nos guste:

<i class="fas fa-exclamation-triangle"></i>

**MODELADO DE CAJAS**

Cada elemento en HTML es una caja, y esas cajas se componen de márgenes (margin), bordes (border), relleno (padding) y finalmente el contenido (content).

4 lados: superior(1) lateral derecho(2) inferior(3)e izquierdo(4)

cualquier elemento de HTML, es una caja de 2 dimensiones: ancho y alto

**box model**

Comportamiento de un elemento determinado según las propiedades y si es de línea o de bloque

propiedades: *display:*

* block:elemento de línea en bloque propiedades de bloque margen superior e inferior, todo el ancho
* inline-block: elemento de bloque en línea, no agrega salto de linea después del elemento
* inline: elemento de bloque en línea. no recomendado, sin margen de ambos lados, ocupa ancho necesario

**width(ancho)**

Si un elemento de bloque no tiene declarada la propiedad width, el ancho será igual al 100% de su padre contenedor.

pixeles, porcentaje o u otra

Para la altura de los elementos no se recomienda usar porcentaje.

**height(alto)**

lo mismo que el otro, elemento si o si de bloque

**padding(relleno):**

separación entre contenido y costado que lo rellenan. área de reserva

generalmente usado valor numérico en pixeles

puede tener hasta 4 valores

si tiene 2 valores: el primero se aplica arriba y abajo y el segundo a la izquierda y a la derecha

3 valores: primero arriba, segundo a los costados y tercero abajo.

4 valores: primero arriba,segundo a la derecha, 3ro abajo y el último a la izquierda

sentido las agujas del reloj.

padding-top: costado superior;

padding-right: lateral derecho

padding-bottom: costado inferior o abajo

padding-left: lateral izquierdo.

**Border:**

**valores border y padding se suman al ancho y alto del elemento**

línea decorativa en 4 costados que rodea un elemento

Se ubica entre el contenido y el margen.

border-width: ancho del borde

border-style:tipo de línea del borde. Dolid, dashed,dotted o double

border-color: color

**Margin:**

no contempla en box-sizing, márgenes exteriores de un elemento, distanciamiento separación de un elemento y los demás que lo rodean

no suma tamaño a la caja, pero si dentro de los elemento que lo contienen

margin-top,

margin-right,

margin-bottom

margin-left

margin: auto, elemento alineado horizontalmente al centro si es de bloque

**box-sizing: border-box;**

Por defecto su valor será content-box, y el comportamiento será el de aplicarle el ancho y alto que definamos al contenido del elemento.

Valor border-box: el ancho y alto tomará en cuenta no solo el contenido del elemento, sino también el **padding y el borde**, dejando solo el margen por fuera.

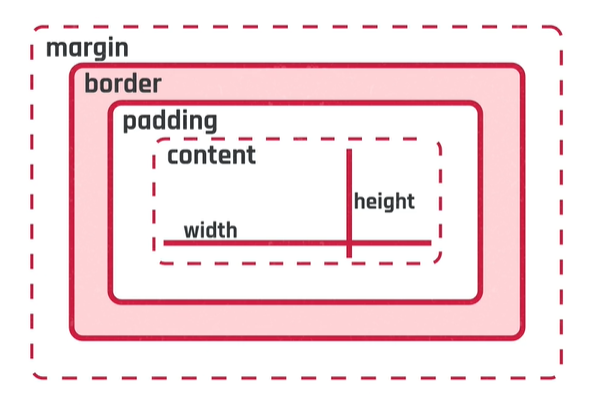
**evita que elemento se amplíe más de los estipulado por width y height**

selector universal para usar únicamente con box-sizing

\*{

box-sizing: border-box;

}



**POSITION RELATIVE- POSICIONAMIENTO RELATIVO**

Traslada un elemento de una posición original a una nueva

**superpone** elementos

Desplazar un elemento al costado, si el número es positivo se mueve a la derecha, si es negativo a la izquierda

Al posicionar la caja 1 de manera relativa, el espacio que ocupaba originalmente seguirá ocupado aunque se desplace.

la caja 2 no se ve modificada.

uso recurrente para **desplazar un elemento sin modificar el flujo original** de los demás que están a su lado.

**POSITION ABSOLUTE**

traslada un elemento tomando como referencia los **costados del body**

Cuando se mueve una caja de manera absoluta, el espacio que ocupaba quedará vacío y otros elementos podrán ocuparlo.

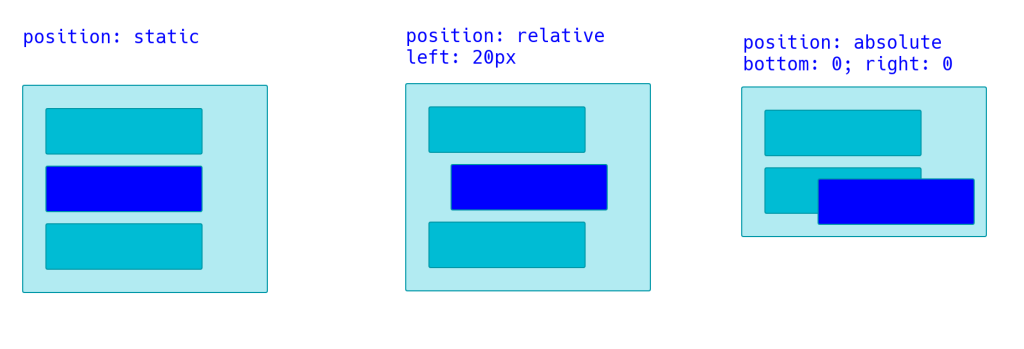
*Si las cajas (hijas) están dentro de otra caja (padre), el punto de referencia seguirá siendo el body a menos que hagamos relativa la posición de su padre*.

Uso para sacar un elemento del flujo normal y posicionarlo en un punto fijo con respecto a su contenedor o el body.

**POSITION FIXED O FIJO**

Puntos de referencia los costados de la ventana del navegador

Uso cuando se quiere que un elemento siga al usuario a medida que navega nuestro sitio.



**POSITION STICKY**

híbrido entre relativo y fijo

es posicionado relativo hasta que cruza un umbral especificado; es fijo hasta que alcanza el límite de su padre

Ejemplo de uso en menú sticky para que al hacer scroll quede fijo en la parte superior, generando así un efecto agradable y elegante.

.caja1{

position:sticky;

left:100px;



**POSICIONAMIENTO Z-index**

permite **cambiar el orden de las “capas”** dentro de un documento HTML

**Lleva hacia atrás o trae hacia adelante** un elemento determinado.

Movimiento en el **eje z, el de la profundidad.**

valor numérico, negativo o positivo

elemento a superponer al de abajo, debe tener un valor mayor del otro

**mayor valor = mas arriba se coloca el elemento**

si 2 cajas tienen el mismo z-index, se coloca adelante el último elemento del HTML

si o si debe tener posicionamiento fixed, relative o absolute para usar z-index

Cualquier elemento con position: **relative, absolute o fixed** se mostrará por encima del resto y tendrá un valor de z-index de 0

.caja1{

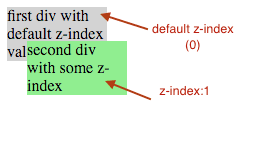
position:relative;

left:100px;

top:50px;

}

Si a la segunda caja se le agrega la propiedad z-index con el valor 10, se moverá delante del resto ya que ahora tiene un valor mayor.

****

**FLEX BOX**

organiza elementos del HTML, maqueta sitio utilizando estructuras de filas y columnas

1994 origen de css

2008 maquetación en cajas flexibles

2011 flexbox

flujo único, libertad para distribuir, redimensionar y reordenar elementos

**EJES**

Dirección que se desplaza el elemento, define orientación a partir de la cual se desplazan los elementos internos

2 ejes no definidos ni por plano vertical ni horizontal:

en función del eje principal se distribuyen en filas horizontales o columnas verticales

**main axis**

horizontal

main axis start(lateral izquierda)

main axis end(lateral derecha)

**cross axis**

vertical

cross axis start superior

cross axis end inferior

posible cambiar orientación de los ejes,

1. **flex-direction (4):**

main axis (eje principal) puede ser vertical como horizontal orientación de los ejes,

**1.flex-direction:row(fila)** por defecto eje principal horizontal y los hijos(flex items) van de izquierda a derecha.

desde flex start a flex end; uno al lado del otro

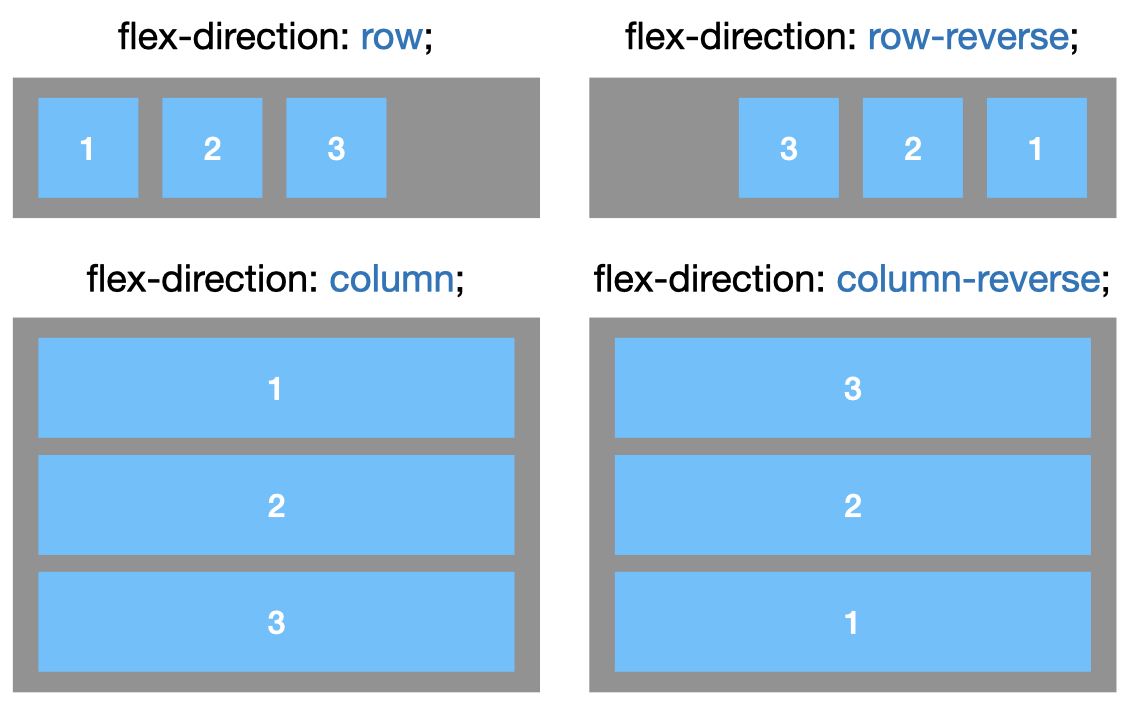
**2.flex-direction:column** cambia el eje principal al vertical, poniendo de arriba a abajo 1, 2, y 3.

eje y es el eje horizontal

**3.flex-direction:row-reverse** eje principal x, pero comienza de izquierda a derecha

**4.flex-direction:column-reverse** eje principal x en vertical, de abajo a arriba

y actúa como eje x, horizontal



**alinear elementos:**

1. **justify-content(6):**

se distribuyen a lo largo del **main axis eje x**

**1.justify-content: flex start:** se alinean sobre el inicio del main, sobre flex start es el valor por **defecto**

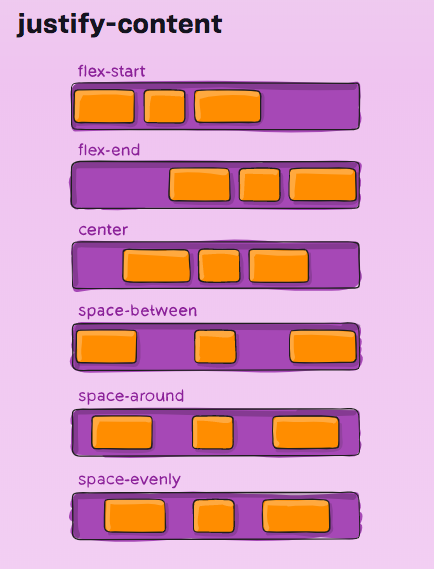
**2.justify-content: flex-end:** alineación respecto al final del main axis, empieza en flex end

**3.justify-content: center** alinea en el centro del main axis los elementos pegados

**4.justify-content: space-between:** distribución uniforme, 1ra caja al inicio del main axis y el último al final. espacio libre que los separa

**5.justify-content: space-around**: espacio libre distribuido entre todos los elementos, mitad a la derecha, mitad a la izquierda.

**6.justify-content: space-evenly**:



1. **align-items:**

alinean a lo largo del cross axis, osea **eje Y**

valor por defecto **stretch** toma todo el espacio disponible

si es vertical cross axis (eje y) se ajustan en función de columna

si es horizontal, se ponen en fila

**align-items: flex-start:**

alinean al inicio del eje y a la izquierda

**align-items: flex-end**

alinean al fondo del eje y a la izquierda

**align-items: center**

se alinean al centro del eje y a la izquierda



1. **align-content:**

contenedores **multilínea**

a lo largo del cross axis, cuando los contenedores incluyen varias líneas y **flex flow es wrap p wrap-reverse**

similares a align-content

**align-content: flex-start**

**eje y como principal** y toma todo el ancho del contenedor, ubica elementos juntos arriba a la izquierda

**align-content: flex-end**

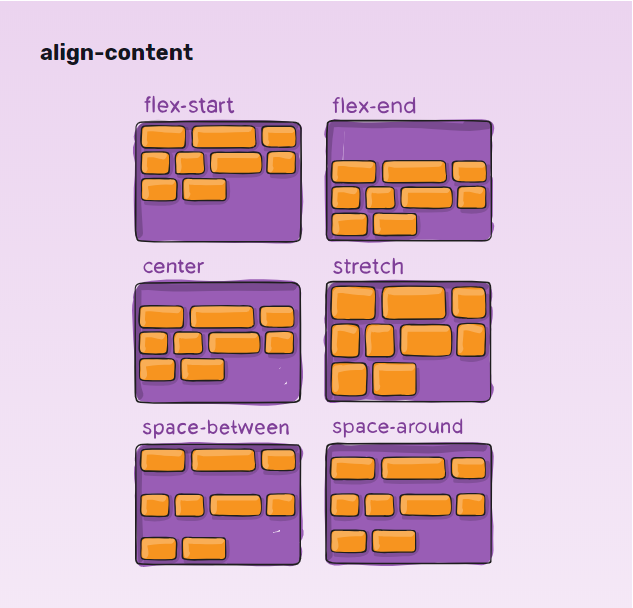
eje y como principal y toma todo el ancho del contenedor, ubica elementos juntos abajo a la izquierda

**align-content: stretch(estirar)**

ocupa a lo largo del eje y y sin dejar espacio libre

**align-content: space-around**

deja espacio a lo largo del eje y,dejando espacio al medio entre 2 filas de elementos



**FLEX-FLOW:** Combinación del flex direction y flex-wrap

**gap**: espacio entre los hijos, valor en pixeles

**ESTRUCTURA**

Uso de flex container y flex-items

Primero: definir display:flex en el padre contenedor

flex-container:

elemento HTML que tiene adentro uno o mas elementos

en el se pone el display en flex para trabajar con los hijos

flex-items:

**flex-wrap (4)**

elementos anidados dentro del container o flex padre.

pueden cambiar a flex-container. asignar display flex así sus hijos se vuelvan flex-items.

**flex-wrap(envoltura flexible)**

**flex-wrap:nowrap** por defecto todos elementos hijos se van a acomodar dentro de una línea

**flex-wrap:wrap** aclara al contenedor padre que debe respetar ancho de sus hijos asi tienen libertad de ocupar mas de una linea

**flex-wrap:wrap-reverse** líneas ordenadas, pero comienza en la línea de abajo para arriba



**PROPIEDADES APLICADAS DIRECTAMENTE A LOS HIJOS**

1. **Order: numero**

número entero positivo o negativo , todos tienen valor 0, si a la primera se asigna 1; pasa al último lugar de la fila al tener un número alto

valor -1, trae a esa caja al primer lugar

secuencia de números negativos a positivos

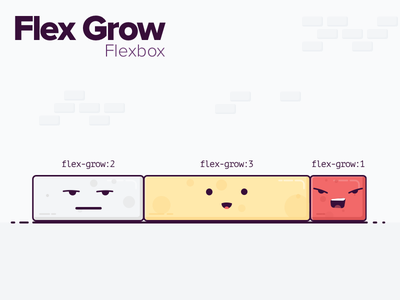
1. **flex-grow:número**

cuanto puede crecer un ítem si tiene espacio libre en el contenedor

1 = 100% del espacio disponible

si 2 tienen la propiedad con valor 1, van a crecer y ocupar partes iguales

si lo tiene solo uno, ese ocupa el espacio libre que le de el contenedor



1. **align-self:**

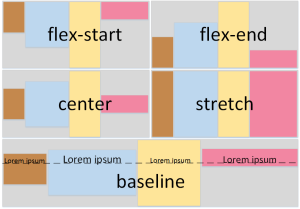
alinear sobre el eje Y solo al ítem que se asignó, mas allá de lo que tengo align-items

**align-self: center** se alinea al centro del eje y

**align-self: flex-start** al centro al inicio del eje y

**align-self: stretch** es ajusta a lo largo del eje y, viene por defecto, siempre que no tenga altura definida

**align-self: flex-end** al centro al abajo del eje y



**flesh-shrink(encojer)** defecto 1, valor mayor a 1, si contenedor se agranda hijo se contrae segun valor que adquiere, mas grande, mayor se contrae

**flex-basis:** crece lo que mas puede en el contenedor, sin desbordarse valor en pixeles

view\_height: vh (altura)

view\_width: vw (ancho)

responsively (app para ver el diseño en dif dispositivos) https://responsively.app/