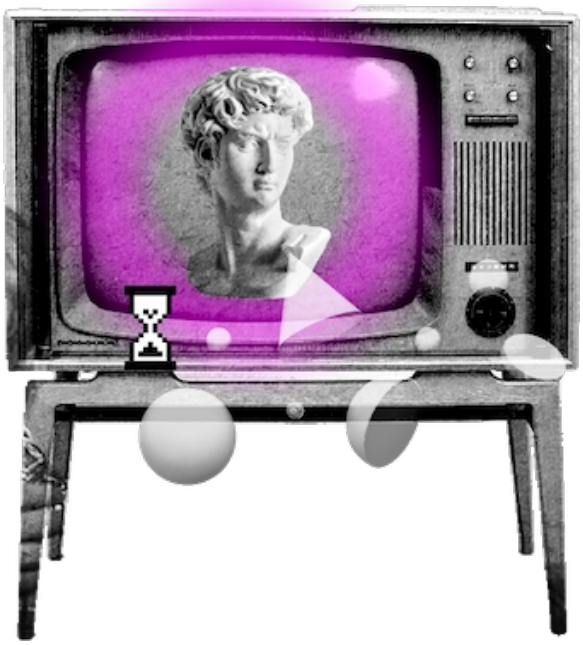
NUCBA

DISEÑO WEB

FLEXBOX



APUNTES

CSS III - FLEXBOX

¿Qué es Flexbox?

Flexbox (también conocido como **Flexible Box Layout**) es un **modelo de diseño** en CSS que se utiliza para **organizar y distribuir elementos HTML dentro de un contenedor**. Permite crear diseños **flexibles y dinámicos** que se adaptan a diferentes tamaños de pantalla y dispositivos.

¿Cómo funciona Flexbox?

El modelo de Flexbox funciona a través de un **contenedor (o elemento padre)** que envuelve a uno o más **elementos secundarios (o hijos)**. El contenedor se define con la propiedad CSS "**display: flex"**, lo que establece que se va a utilizar el modelo de Flexbox para organizar sus hijos.

```html
    <section class="padre">
        <div class="hijo">elemento hijo 1</div>
        <div class="hijo">elemento hijo 2</div>
        <div class="hijo">elemento hijo 3</div>
    </section>
```

```css
.padre {
  display: flex;
}
```

Una vez que se ha establecido **"display: flex"** en el contenedor, se pueden aplicar diferentes propiedades a los elementos secundarios para **controlar su tamaño**, **posición** y **comportamiento**.

Importante: el uso de display flex en el elemento padre y las respectivas propiedades que se utilicen en él se aplicarán **únicamente** a los **hijos directos** del contenedor. Si quisiéramos aplicar **flexbox** a elementos dentro de alguno de los hijos, deberemos colocar "**display:flex"** al contenedor hijo que sea padre de sus propios elementos hijos.

Propiedades de flexbox

Como mencionamos anteriormente, flexbox tiene una serie de propiedades que permiten controlar el tamaño, posición, dirección y comportamiento de los elementos.
Existen propiedades de flexbox que se aplicarán a los elementos padre y otras que se aplicarán a los elementos hijos.

Propiedades de elementos padres

Flex-direction

La propiedad **"flex-direction"** es una propiedad CSS que se utiliza para establecer la **dirección** en la que se colocan los elementos secundarios (o hijos) dentro de un contenedor en **Flexbox**. Esta propiedad define el **eje principal del contenedor** y afecta a la forma en que se distribuyen los elementos secundarios a lo largo de este eje.

```
.padre {
  display: flex;
  flex-direction: /* row | row-reverse | column | column-reverse */
}
```

row:

Es el valor por defecto de **flex-direction**. Implica que **los elementos hijos se colocarán en una fila de izquierda a derecha**.

No suele ser necesario colocar esta propiedad con este valor ya que, como mencionamos antes, es el valor predeterminado que tomará un contenedor con la propiedad **display:flex**. No obstante, existen casos puntuales como cuando trabajemos en la adaptabilidad de nuestra página a todos los dispositivos en los cuales podemos llegar a tener que colocar este valor.

row-reverse:

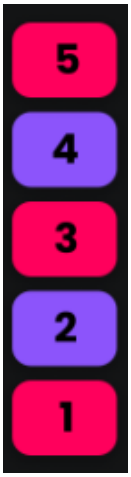
Colocar este valor implica que los elementos hijos se colocarán en una **fila de derecha a izquierda** , a la inversa del valor **row**.

column:

Colocar este valor implica que los **elementos hijos se colocaran en columna, de arriba hacia abajo**.

column-reverse:

Colocar este valor implica que los **elementos hijos se colocaran en columna, de abajo hacia arriba,** a la inversa de **column**.



Flex-wrap

La propiedad **"flex-wrap"** es una propiedad CSS que se utiliza para establecer si los elementos hijos dentro de un contenedor en Flexbox deben **volcarse a una nueva línea** si no caben en el espacio disponible o si **deben reducir su tamaño** para **ajustarse al contenedor.** Los posibles valor de **flex-wrap** son:

```
.padre {
  display: flex;
  flex-wrap: /* nowrap | wrap | wrap-reverse */
}
```

nowrap:

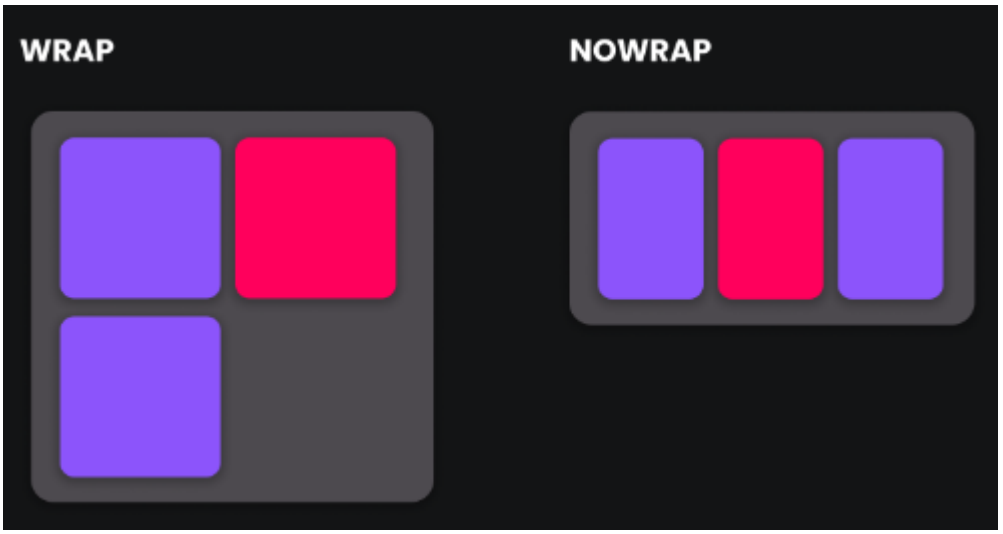
Los elementos hijos se **ajustarán automáticamente para intentar que todos ellos encajen en una sola línea**. Si no hay suficiente espacio en la línea, se **reducirán de tamaño para ajustarse**. Es el valor por defecto de la propiedad **flex-wrap** , por lo que si este es el comportamiento que queremos que tengan por defecto los elementos hijos de nuestro contenedor, no es necesario ponerlo en el código.

wrap:

Los elementos hijos se **distribuirán en líneas**, si es necesario, respetando la dirección establecida con la propiedad **"flex-direction"**. Es decir, si la dirección es **"row"**, los elementos se **distribuirán en filas** y si es **"column"**, **en columnas**. Si alguno de los elementos no cabe en la línea, se volcará a una nueva línea.

wrap-reverse:

Los elementos hijos se **distribuirán en líneas en la dirección opuesta a la establecida con "flex-direction"**. Si algunos de los elementos no caben en la línea, se envolverán a una nueva línea en la dirección opuesta.



Flex-flow

Es una propiedad abreviada de CSS que combina las propiedades **flex-direction** y **flex-wrap**. Podemos utilizar cualquiera de los valores de cada una , separados por un espacio, siendo el primer valor la **dirección** y el segundo el **wrap**.

```
.padre {
  display: flex;
  flex-flow: column wrap;
}
```

Ejes en flexbox

Ahora que ya vimos las propiedades más básicas de flexbox, y antes de avanzar hacia otras más complejas, es buena idea que entendamos que son los **ejes de flexbox.**

En Flexbox, hay **dos ejes** que son importantes para el posicionamiento y alineación de elementos hijos dentro del contenedor: el **eje principal** y el **eje transversal**.

El **eje principal** es el eje a lo largo del cual se establece la dirección de los elementos hijos dentro del contenedor. Esto se define mediante la propiedad **"flex-direction"**.

El **eje transversal** es **perpendicular al eje principal** y se **extiende en la dirección opuesta**. Su dirección es determinada por la dirección establecida por **"flex-direction"**.

Por ejemplo, si establecemos la propiedad **"flex-direction" en "row"**, los elementos hijos se colocarán uno al lado del otro en el **eje X** (horizontal), siendo este el **eje principal** y siendo el **eje Y** (vertical) el **eje transversal**. Si, en cambio, establecemos la propiedad **"flex-direction" en "column"**, los elementos secundarios se colocarán uno debajo del otro en el **eje Y** (vertical), siendo este el **eje principal** y pasando el **eje X** (vertical) a ser el eje transversal.

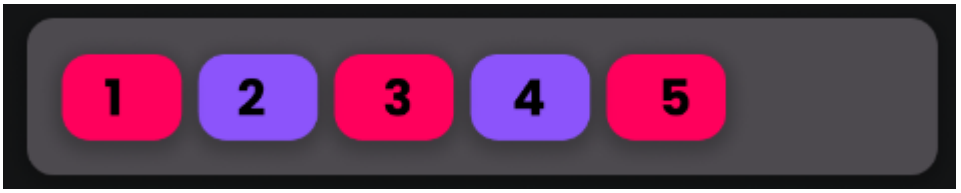
Justify-content

La propiedad **"justify-content"** es una propiedad de CSS que se utiliza para **alinear los elementos hijos** dentro del contenedor en el **eje principal** (ya sea horizontal o vertical, dependiendo de la dirección establecida por **"flex-direction"**). Esta propiedad se aplica sólo a los elementos secundarios que no ocupan todo el espacio disponible en el contenedor. Los posibles valores para esta propiedad son:

```css
.padre {
  display: flex;
  justify-content: flex-start;
  justify-content: center;
  justify-content: flex-end;
  justify-content: space-between;
  justify-content: space-around;
  justify-content: space-evenly;
}
```

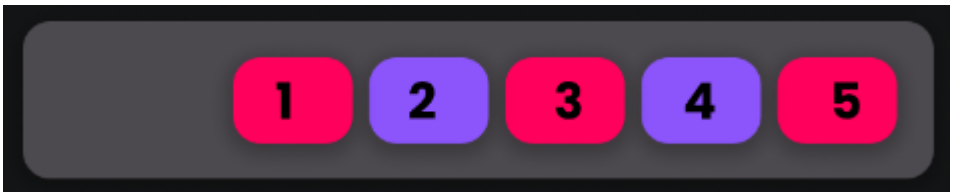
flex-start:

Este valor alinea los elementos hijos al **inicio del contenedor en el eje principal** (izquierda si "flex-direction" es "row" o arriba si "flex-direction" es "column").



flex-end:

Este valor alinea los elementos hijos al **final del eje principal del contenedor**, es decir, en el extremo derecho si el valor de "flex-direction" es "row" o en el extremo inferior si es "column".



center:

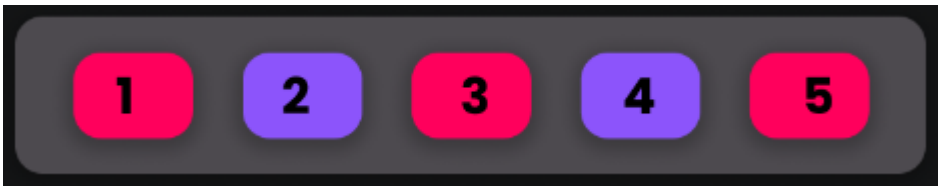
Este valor alinea los elementos hijos en el **centro del eje principal del contenedor**.

space-between:

Este valor **distribuye** los elementos hijos **de manera uniforme a lo largo del eje principal del contenedor**, con el **primer elemento al comienzo** y el **último elemento al final del contenedor.**

space-around:

Este valor también **distribuye los elementos hijos de manera uniforme a lo largo del eje principal del contenedor**, pero con un **espacio entre el primer y último elemento respecto de los extremos del eje principal.**



space-evenly:

Este valor distribuye los elementos hijos de manera uniforme a lo largo del eje principal del contenedor, incluyendo el espacio antes del primer elemento y después del último elemento. La diferencia con **space-around** radica en que será **idéntico el espacio entre el primer y último elemento con los extremos del eje principal y el espacio entre elementos**.

Align-items

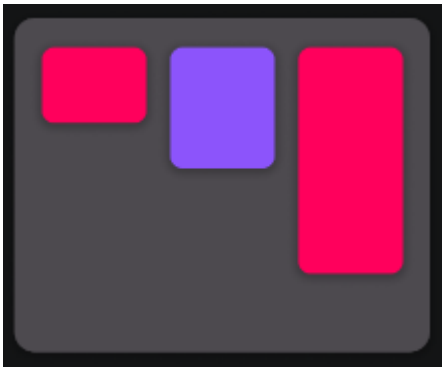
La propiedad **"align-items"** se utiliza para **alinear los elementos hijos a lo largo del eje transversal** del contenedor. Los posibles valores para esta propiedad son:

```
.padre {
  display: flex;
  align-items: flex-start;
  align-items: flex-end;
  align-items: center;
  align-items: baseline;
  align-items: stretch;

}
```

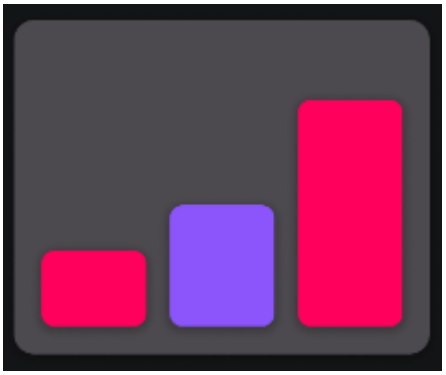
flex-start:

Este valor alinea los elementos hijos al **inicio del contenedor en el eje transversal del contenedor**.



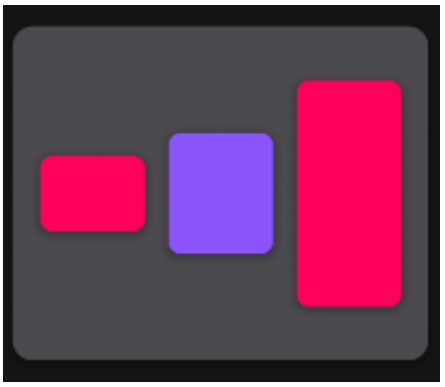
flex-end:

Este valor alinea los elementos hijos al **final del eje transversal del contenedor**.



center:

Este valor alinea los elementos hijos en el **centro del eje transversal del contenedor**.

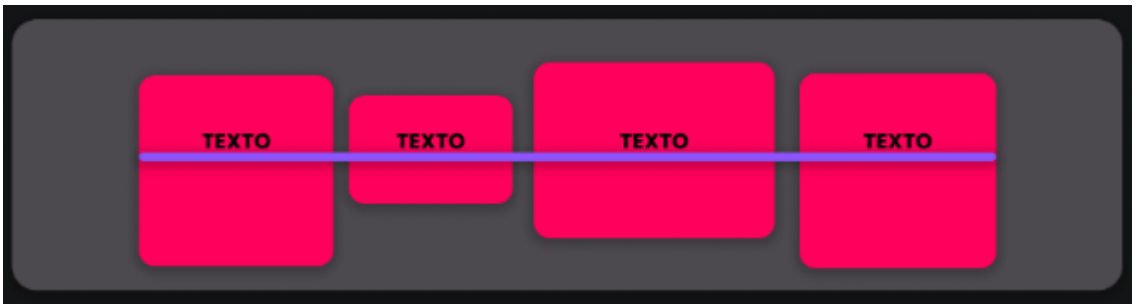


stretch:

Este valor es el valor por defecto de "align-items" y **estira los elementos hijos para que se ajusten al alto del contenedor a lo largo del eje transversal**.

baseline:

Este valor alinea los elementos hijos en su **línea base común**, es decir, en la **línea imaginaria donde descansan las letras en un texto**.



Align-content

La propiedad **"align-content"** en Flexbox controla la **alineación de varias líneas de elementos hijos dentro de un contenedor en el eje transversal**.

Esta propiedad se aplica **solo cuando hay varias líneas de elementos en un contenedor Flexbox**, es decir, cuando la propiedad **"flex-wrap"** está establecida en **"wrap"** o **"wrap-reverse"**. Puede tomar los siguientes valores:

```
.padre {
  display: flex;
  align-content: flex-start;
  align-content: flex-end;
  align-content: center;
  align-content: stretch;
  align-content: space-around;
  align-content: space-between;
  align-content: space-evenly;
}
```

"flex-start": Alinea las líneas de elementos al comienzo del eje transversal del contenedor.
"flex-end": Alinea las líneas de elementos al final del eje transversal del contenedor.
"center": Alinea las líneas de elementos en el centro del eje transversal del contenedor.
"space-between": Distribuye uniformemente el espacio adicional entre las líneas de elementos, dejando un espacio en blanco en los bordes superior e inferior del contenedor.
"space-around": Distribuye uniformemente el espacio adicional alrededor de las líneas de elementos, incluyendo los bordes superior e inferior del contenedor.
"stretch": Estira las líneas de elementos para llenar todo el espacio disponible en el eje transversal del contenedor.
“space-evenly”: Distribuye uniformemente el espacio disponible tanto entre las líneas de elementos como alrededor de los bordes superior e inferior del contenedor en el eje transversal.

Gap

La propiedad **"gap"** es una propiedad de Flexbox que **define el espacio entre los elementos de un contenedor**.

Para determinar el valor de esta propiedad, podemos aplicar las distintas unidades de medida que vimos anteriormente,, que representan la cantidad de espacio que deseamos dejar entre los elementos.

Es importante destacar que, como en el caso del resto de las propiedades de los elementos padres, la propiedad **"gap"** solo se aplica a los elementos que se encuentran en el **contenedor directo de Flexbox (hijos directos)**, no a los elementos anidados dentro de ellos.

```
.padre {
  display: flex;
  justify-content:center;
  align-items: center;
  gap: 100px;
}
```

En este caso de ejemplo, nuestros elementos van a alinearse en el centro de nuestro contenedor, pero tendrán un espacio de 100px entre sí.

Gap suele ser más útil que utilizar **margin** o **padding** para **separar elementos internos** de un contenedor ya que no importa si los elementos están uno a la derecha del otro o uno abajo del otro, ya que el **espacio siempre será entre medio de ellos**.

Propiedades de elementos hijos

IMPORTANTE: Antes de que pasemos a ver algunas propiedades que se colocan en los elementos hijos de flex, creemos que es necesario aclarar que, para que las mismas funcionen, el contenedor padre debe ser un contenedor con **display:flex;**

Order

La propiedad **"order"** es una propiedad de Flexbox que permite **especificar el orden de visualización de los elementos dentro de un contenedor, independientemente del orden en que aparecen en el código HTML**.

El valor predeterminado de la propiedad "order" es 0. Si deseamos cambiar el orden de visualización de un elemento, debemos establecer un valor numérico diferente a 0.

Los elementos se ordenan de **menor a mayor**, es decir, **los elementos con un valor menor aparecen antes que los elementos con un valor mayor**.

```
    <section class="padre">
        <div class="hijo1">elemento hijo 1</div>
        <div class="hijo2">elemento hijo 2</div>
    </section>
```

```
.padre {
  display: flex;
}

.hijo2 {
  order: 1;
}

.hijo1 {
  order: 2;
}
```

En este ejemplo, podemos ver que, según la ubicación en el documento HTML, el elemento con clase **hijo1** se vería primero en la página que el elemento con la clase **hijo2**.

Utilizando la propiedad **order** lo que logramos es cambiar dicho orden, haciendo que el elemento con la clase **hijo2** aparezca primero en nuestra **página**.

Flex-grow

La propiedad **flex-grow** es una propiedad de Flexbox que se utiliza para especificar cómo se **distribuyen los espacios disponibles entre los elementos flexibles** dentro de un contenedor. Esta propiedad **controla la capacidad de un elemento para crecer en relación con otros elementos flexibles dentro del mismo contenedor**.

El valor de **flex-grow** determina **cuánto espacio adicional puede ocupar un elemento flexible dentro del contenedor en relación con otros elementos flexibles**. Si todos los elementos tienen el mismo valor de **flex-grow**, entonces el espacio adicional se distribuye entre ellos de manera uniforme. Si un elemento tiene un valor mayor de flex-grow qué otros elementos, entonces se le asignará más espacio adicional en relación con los demás.

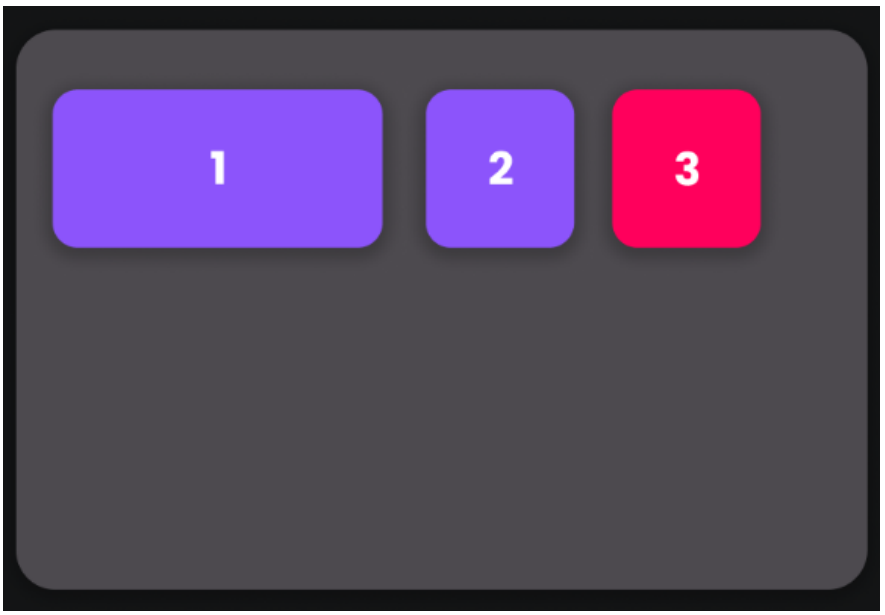
Por ejemplo, si tenemos un contenedor con tres elementos flexibles, y el primer elemento tiene un valor de **flex-grow de 2**, mientras que **los otros dos elementos** tienen un valor de **flex-grow de 0 por defecto**, entonces el **primer elemento tendrá el doble de espacio adicional** en relación con los otros dos elementos.

Es importante aclarar que 0 es el valor por defecto, por lo tanto, no es necesario colocarlo.

```css
.padre {
  display: flex;
}

.hijo1 {
  flex-grow: 2;
}

/*Por defecto, hijo2 e hijo3 tiene un valor de flex-grow: 0;*/
```



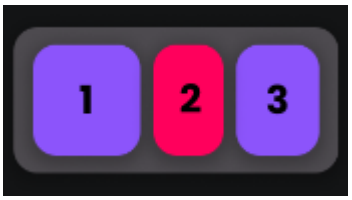
Flex-shrink

La propiedad **flex-shrink** se utiliza en flexbox para definir cómo se **deben encoger los elementos flexibles en un contenedor flexible si no hay suficiente espacio disponible**. La propiedad acepta un valor numérico que representa el **factor de encogimiento** de un elemento. Por ejemplo, si el **factor de encogimiento** de un elemento es 2 y el de otro es 1, el primer elemento se encogerá el doble que el segundo si es necesario.

Por defecto, el valor de flex-shrink es 1, lo que significa que el elemento se **encogerá en proporción a los demás elementos si no hay suficiente espacio disponible**. Si se establece en 0, el elemento no se encogerá y mantendrá su tamaño original, incluso si no hay suficiente espacio disponible.

```css
.padre {
  display: flex;
}

.hijo1 {
  flex-shrink: 0;
}
.hijo2 {
  flex-shrink: 2;
}
/*Por defecto, hijo3 tiene un valor de flex-shrink: 1;*/
```



Flex-basis

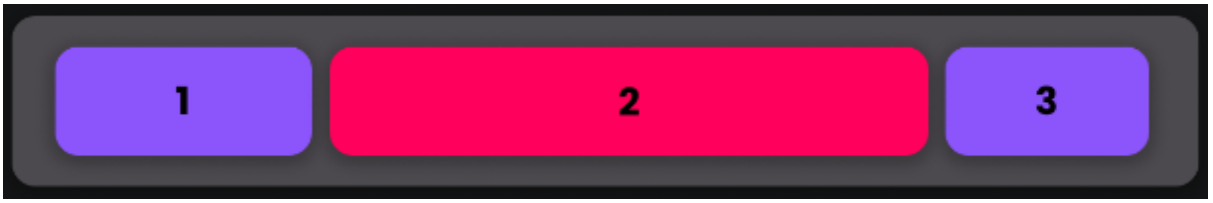
Es una propiedad de Flexbox que **establece el tamaño base de un elemento flexible antes de que se distribuya el espacio sobrante entre los elementos**. Es similar a la propiedad width o height, ya que **define el tamaño inicial del elemento**.

El valor por defecto de flex-basis es **auto**, lo que significa que **el tamaño inicial del elemento se basará en el tamaño contenido en su interior o en su ancho o alto establecido por otras propiedades CSS**. Si se desea establecer un tamaño base específico para el elemento, se debe utilizar un valor diferente a auto.

Veamos un ejemplo de uso de esta propiedad:

```css
.padre {
  display: flex;
}

.hijo1 {
  flex-basis: 200px;
}
.hijo2 {
  flex-basis: 500px;
}
.hijo3 {
  flex-basis: 150px;
}
```



En este ejemplo definimos el tamaño de base para cada uno de los elementos. Luego, podremos trabajar en su tamaño combinándolo con el uso de **flex-grow** y **flex-shrink**.

Es importante aclarar que cada elemento va a ocupar esa base siempre y cuando pueda ocupar esa cantidad de píxeles.

Flex

La propiedad flex es una **propiedad abreviada** de CSS que permite establecer de manera fácil y rápida las tres propiedades de flexbox mencionadas anteriormente: **flex-grow, flex-shrink y flex-basis**. Se colocarán las 3 opciones separadas por un espacio.

```css
.padre {
  display: flex;
}

.hijo1 {
  flex: /* flex-grow | flex-shrink | flex-basis; */
```

Align-self

La propiedad **align-self** se utiliza en Flexbox para **alinear individualmente un elemento hijo a lo largo del eje transversal** (eje perpendicular) del contenedor. Es similar a la propiedad align-items, pero se aplica **solo a un elemento específico** en lugar de a todos los elementos hijos del contenedor.

Puede tomar los mismos valores que tomaba la propiedad **align-items.**

```
.padre {
  display: flex;
  align-items: center; /* Alinea todos los elementos hijos en el centro verticalmente */
  height: 300px;
}

.hijo2 {
  align-self: flex-start; /* Este elemento se alinea en la parte superior del contenedor */
}
```

En este ejemplo, los 3 hijos se colocarían en el centro del contenedor, pero debido a que el elemento con la clase **hijo2** tiene la propiedad **align-self** seteada en **flex-start** , dicho elemento se alineará en la parte superior del contenedor.

Conclusiones finales

Aprender Flexbox es clave para ser un desarrollador web porque permite crear diseños adaptables y flexibles para las páginas web de una manera más fácil y eficiente. Con Flexbox, se pueden crear diseños más complejos con menos código y menos dependencia del uso excesivo de propiedades como **position, margin y padding**, lo que simplifica el mantenimiento y la escalabilidad del código. Además, Flexbox es compatible con la mayoría de los navegadores modernos, lo que lo convierte en una opción viable y segura para la mayoría de los proyectos web.

Por ende, ahora que sabemos **flexbox** , no usaremos las 3 propiedades antes mencionadas para centrar o ubicar elementos dentro de un contenedor. Las usaremos únicamente cuando sean puntualmente necesarias, como cuando necesitemos generar un espaciado con los bordes, separación entre secciones o posicionar , por ejemplo, una barra de navegación en el tope de nuestra página.

Nucba tip: *Sigan investigando sobre los temas aquí expuestos. Practiquen, apoyense en la documentación oficial y sobre todo mantengan la constancia y la curiosidad a medida que vayan aprendiendo cosas nuevas.*

#HappyCoding 🚀