<codoacodo/>

Desarrollo Fullstack

BA

Les damos la bienvenida

Vamos a comenzar a grabar la clase

CSS

Estructura en nuestros estilos

Hoy presentamos...

Agencia de Aprendizaje a lo largo de la vida

Buenos Aires Ciudad
Educación

FLEXBOX

Nuestro nuevo mejor amigo

¿Qué es Flexbox?

Como mencionamos anteriormente, en el **pasado** para maquetar la estructura de *un sitio web*, era necesario **recurrir** a las tablas o a la propiedad **float** y sus raros comportamientos.

Flexbox es un **sistema** de elementos que nos invita a olvidar estos mecanismos y utilizar una mecánica más potente, limpia y flexible, en la que los elementos HTML *se adaptan y fluyen automáticamente*.

*De este modo **lograremos** que nuestros diseños sean adaptables a diferentes tamaños de pantallas con menos código.*

Flex Container - Flex Item

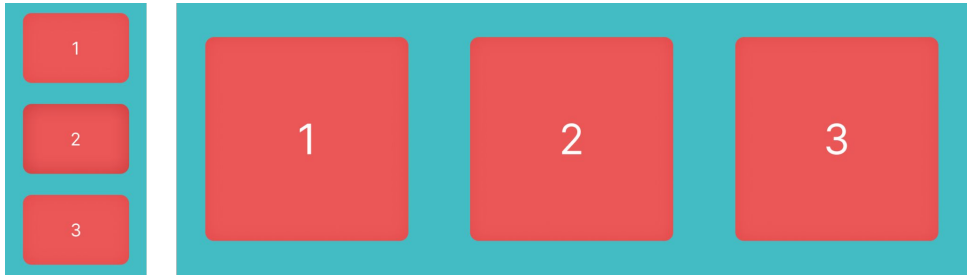
En Flexbox para **posicionar** nuestros elementos debemos asignarle el comportamiento que buscamos al contenedor padre de estos, a diferencia de otras alternativas como ***display inline, inline-block*** o con la propiedad **float** donde la propiedad **se aplicaba** a cada elemento de forma individual.

Esto nos trae un nuevo enfoque de ***"Flex Container"*** que es el elemento padre que tendrá en su interior cada uno de los ítems flexibles y ***"Flex Item"*** que son cada uno de los hijos que tendrá el contenedor en su interior.

Ejes

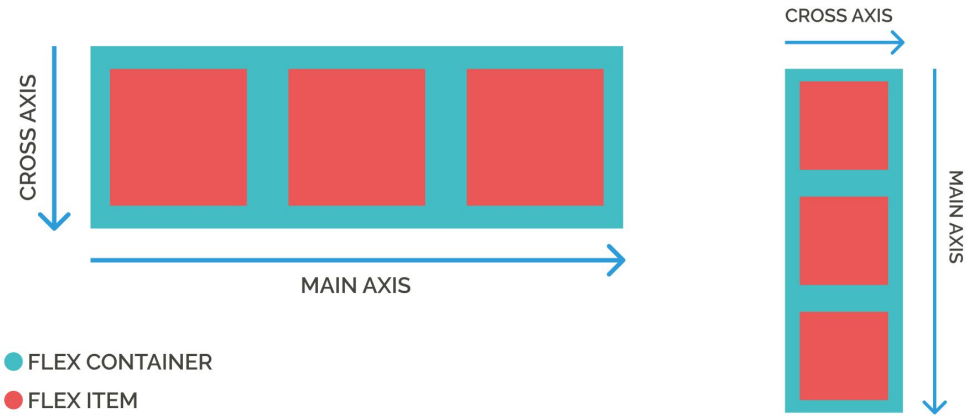
Flexbox es un **conjunto de propiedades CSS** que nos permiten posicionar nuestros elementos HTML con mayor facilidad bajo la premisa de *establecer una dirección vertical u horizontal* para nuestros elementos hijos desde un elemento padre.

Esto nos obliga a **trabajar nuestros contenedores** sobre **un** solo eje al mismo tiempo, ya sea en eje **Y** (vertical) o el eje **X** (horizontal) o como comúnmente se los conoce, **columna** o **fila**.



Main Axis - Cross Axis

Al trabajar de esta manera, nos encontramos con que tenemos un ***Eje Principal*** (main axis) y un ***Eje secundario*** (cross axis).



Propiedades

Flexbox no depende de una única propiedad si no que se encuentra compuesto por un **conjunto** de ellas que nos permiten configurar la disposición de **nuestros elementos** en contenedores flexibles.

En su mayoría, estas propiedades son aplicadas a los elementos padres (flex container) aunque **algunas** se usan en los ***elementos hijos*** (flex item).

Declaración de Contenedor Flexible

Lo primero que debemos hacer **para comenzar a usar flex** es declarar a nuestro *contenedor padre*, que **sus hijos** serán posicionados de forma **flexible**.

Eso lo hacemos con la propiedad `display` que ya conocemos, pero ahora con el valor: `flex;`

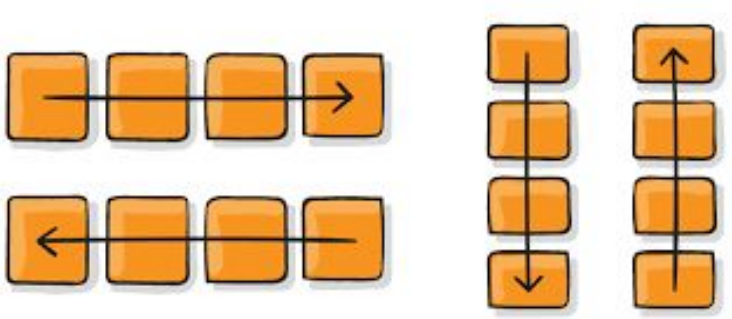
```html
<section class="flex-container">
    <article class="flex-item">
        <h2>Cetaceos</h2>
        <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit.
        Quam iure error libero.</p>
    </article>
    <article class="flex-item">
        <h2>Felinos</h2>
        <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit.
        Quam iure error libero.</p>
    </article>
</section>
```

```css
.flex-container {
    display: flex;
}
```

Dirección - Multilínea

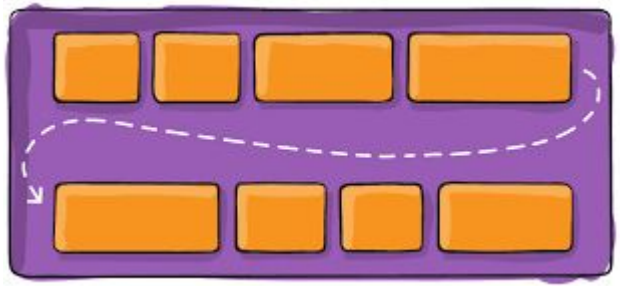
Para manejar la **dirección** de nuestros items vamos a utilizar la propiedad `flex-direction` que puede tomar los valores:

`row* | row-reverse | column |column-reverse`



Si queremos un comportamiento **multilínea** entonces nuestra propiedad es `flex-wrap` que puede tomar los valores:

`wrap | wrap-reverse | nowrap*`



`*` valores por defecto

`flex-direction` y `flex-wrap` se pueden declarar juntos mediante el shorthand `flex-flow`.

```
.flex-container {
    display: flex;
    flex-flow: row nowrap;
}
```

Alineaciones

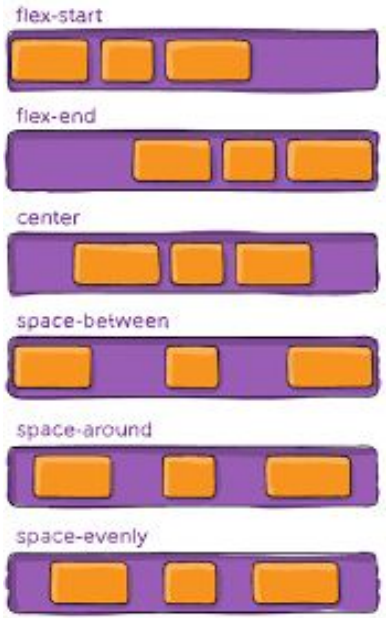
Son la magia de flexbox, con ellos podemos **distribuir** *uniforme y automáticamente* nuestros **elementos hijos** dentro de un contenedor flexible.

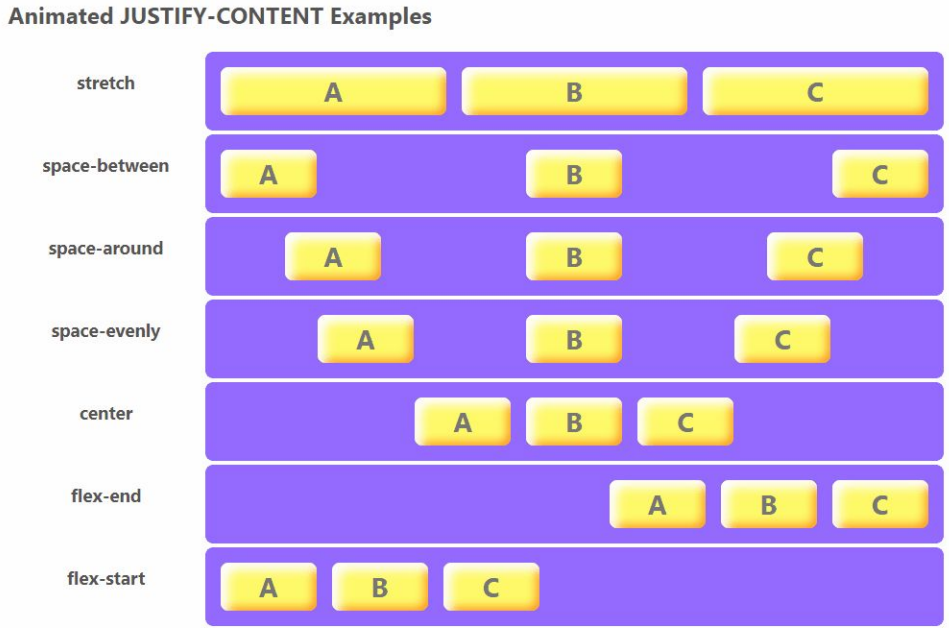
Existen múltiples opciones y según cual usemos **actúan** **sobre el main axis** o el **cross axis** dependiendo el flex direction que hayamos aplicado.

justify-content

`flex-start | flex-end | center | space-between | space-around | space-evenly`

Se utiliza para alinear los ítems del **eje principal** o **main axis**.

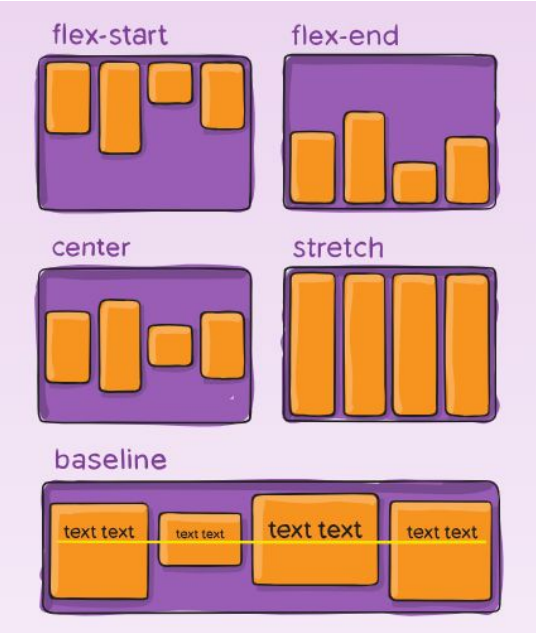




align-items

`flex-start | flex-end | center | stretch | baseline`

Se utiliza para alinear los ítems del **eje secundario** o **cross axis**.

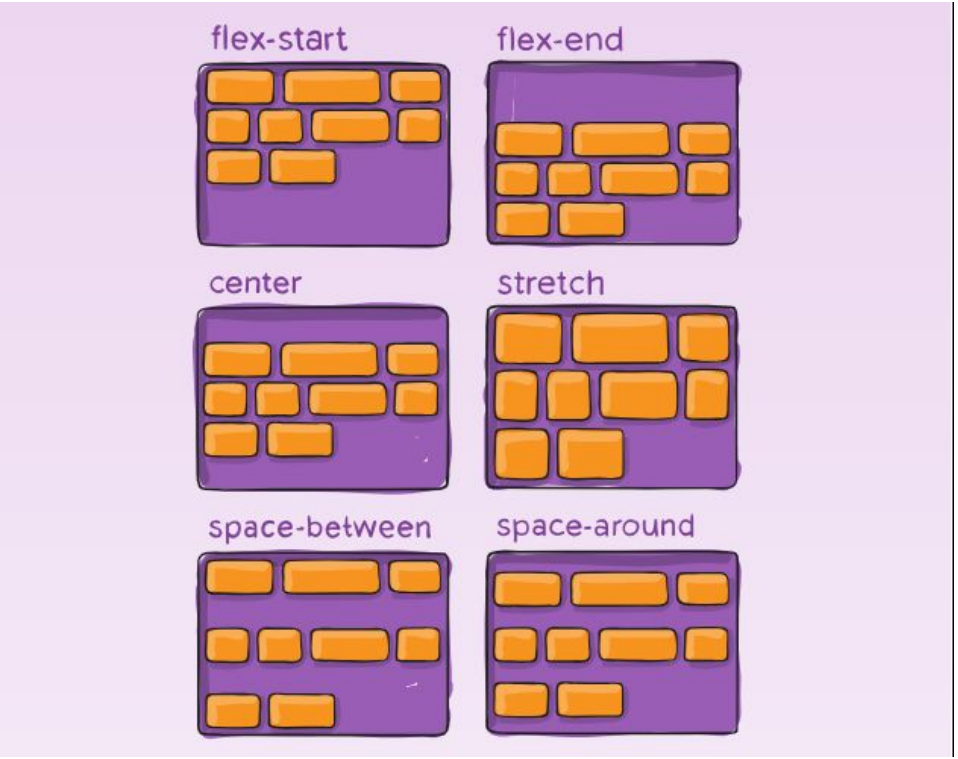


align-items: flex-start;

align-content

`flex-start | flex-end | center | space-between | space-around | space-evenly | stretch`

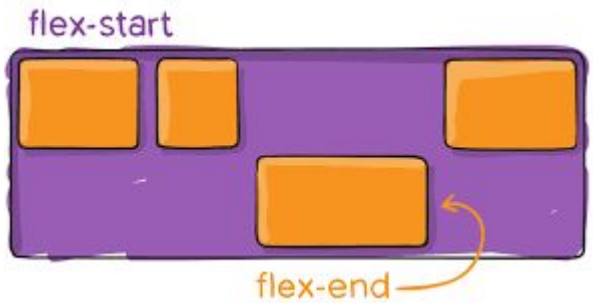
Servirá para **alinear** cada una de las **líneas** del contenedor multilínea.



align-self

`auto | flex-start | flex-end | center | stretch | baseline`

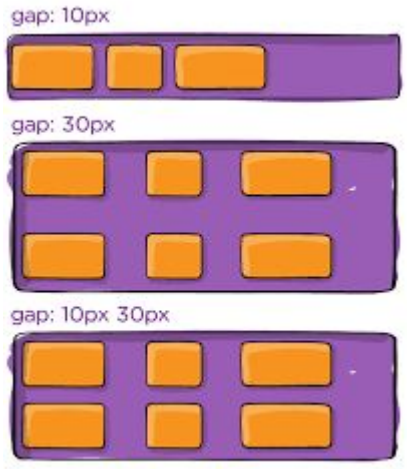
Nos permite asignar un **align-item** diferente a un solo elemento hijo.



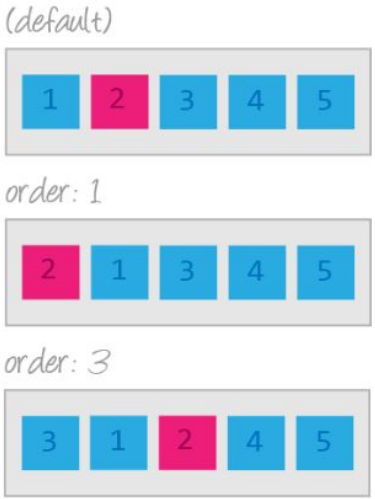

```
.flex-container {
    align-items: flex-start;
}

.flex-item {
    align-self: flex-end;
}
```

Espaciado - Orden

Para manejar el espaciado entre nuestros items vamos a utilizar la propiedad `gap` que toma cualquier unidad de medida conocida.



Podemos cambiar el orden de nuestros hijos con `order`, esta propiedad se aplica sobre el item hijo con el valor de posición que debe tomar.




```
.flex-item {
    order: 3;
}
```

Flex Grow

Nos ayuda a **distribuir** el espacio sobrante del contenedor padre entre los items hijos. Se declara en estos últimos a través de la propiedad `flex-grow` y su valor es un **número entero** que indica cuánto le corresponde de ese sobrante.

```
.flex-item-1 {
    flex-grow: 2;
}

.flex-item-1 {
    flex-grow: 4;
}

.flex-item-3 {
    flex-grow: 2;
}
```

`200px*3 = 600px | 400px/8 = 50xp`

Flex Shrink

`flex-shrink` funciona igual que `flex-grow` solo que en este caso va a **determinar** qué sucede cuando *al elemento padre* **le falta** espacio para ubicar a sus hijos, es decir, de quien toma el espacio faltante.

El valor y como se calcula es igual que el anterior.

```
.flex-item-1 {
    flex-shrink: 1;
}

.flex-item-1 {
    flex-shrink: 3;
}

.flex-item-3 {
    flex-shrink: 1;
}
```

Flex Basis

`flex-basis` nos permite determinar el **ancho por defecto** de un item flexible como si de la propiedad **width** se tratara.

```
.flex-item {
    flex-basis: 250px;
}
```

Flex

La propiedad `flex` es el shorthand property de las 3 anteriores.

```
.flex-item {
    /* grow | shrink | basis */
    flex:1 1 250px;
}
```

No te olvides de dar el presente

Recordá:

- Revisar la Cartelera de Novedades.
- Hacer tus consultas en el Foro.

Todo en el Aula Virtual.

Gracias