UT 4 – CSS Avanzado.
Flexbox Layout.

Contenido

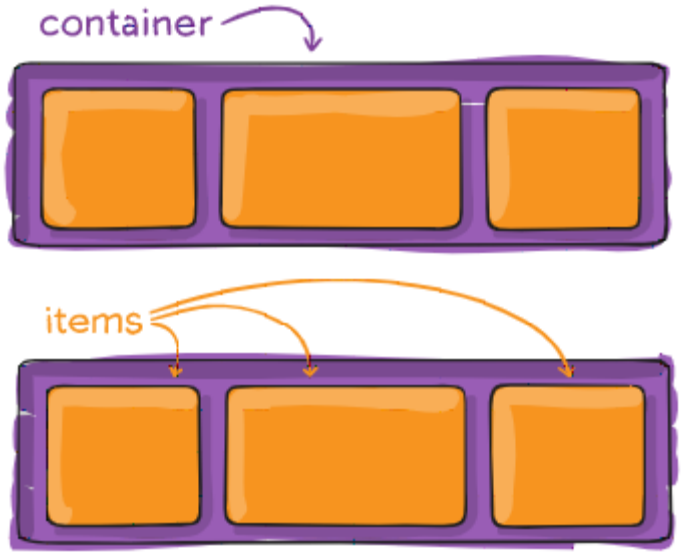
1. ¿Qué es flexbox?
2. Funcionamiento.
3. Propiedades contenedor.
4. Propiedades flex ítems.
5. Ejemplos prácticos.

1. ¿Qué es flexbox?

- El Módulo de Caja Flexible, comúnmente llamado flexbox, fue diseñado como un modelo unidimensional de *layout* (filas o columnas).
- Ayuda a distribuir el espacio disponible entre los ítems de una interfaz.
- Mejora las capacidades de alineación.

2. Funcionamiento

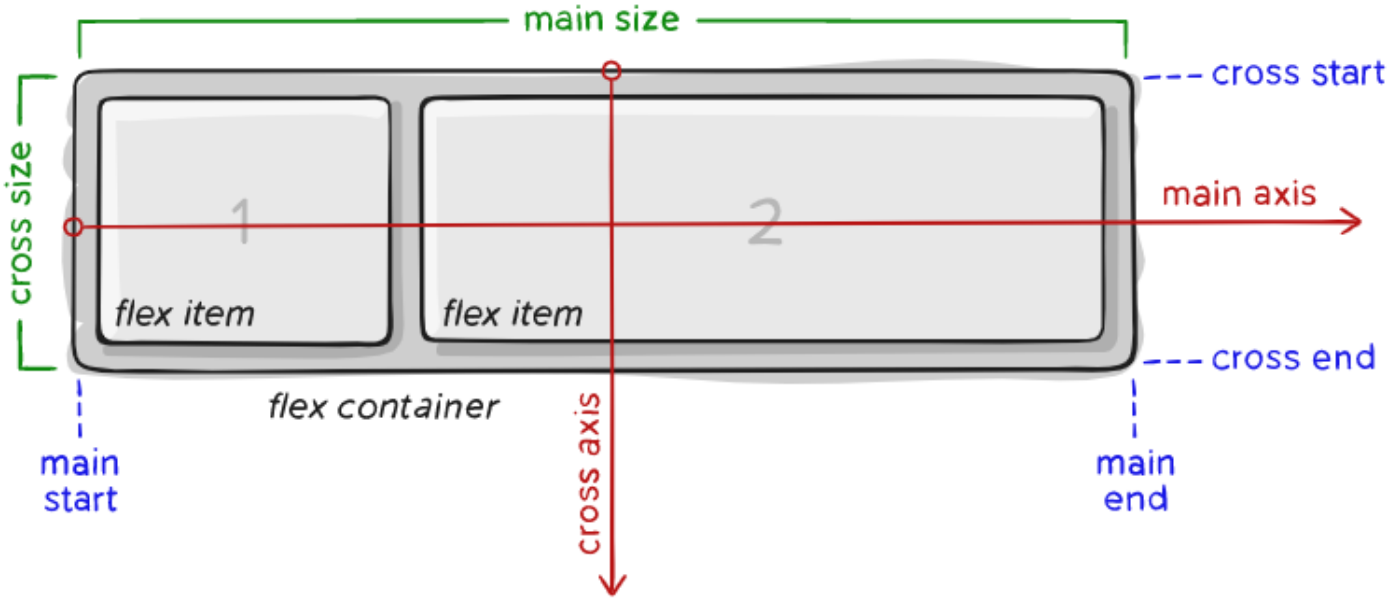
- **Idea principal**: Hay un contenedor de tipo flex (elemento padre) dentro del cual hay ítems (elementos hijos).



- Algunas propiedades es necesario aplicarlas al contenedor y otras a los ítems.

2. Funcionamiento

- Hay un eje principal (*main*) y uno secundario (*cross*), el cual es perpendicular al principal.
- Los elementos se disponen a lo largo del eje principal, desde el principio (*main start*) hasta el final (*main end)*.



3. Propiedades contenedor

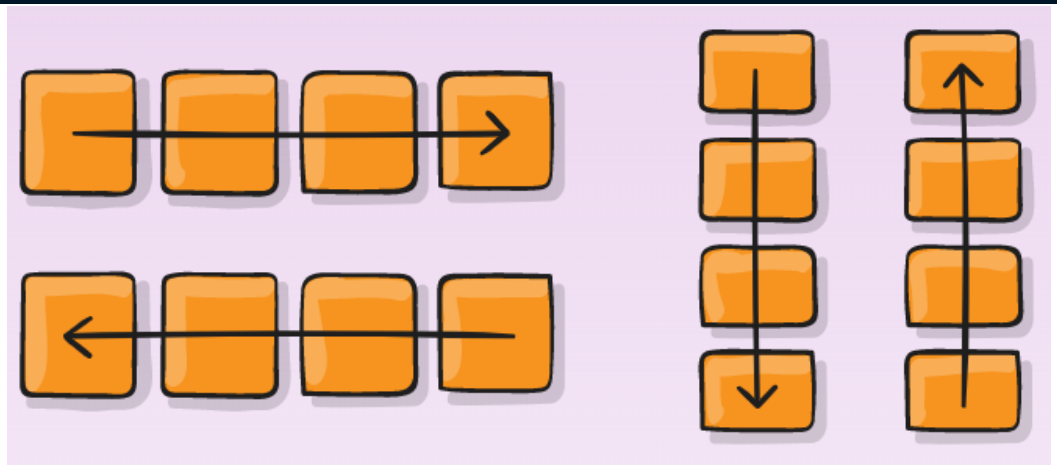
- Para definir un elemento como contenedor flexible (*flex container*) se utiliza la propiedad *display*.
- Se utilizará:
  - **display:flex**; Para que se comporte como un elemento de bloque.
  - **display:inline-flex**; Para que se comporte como un elemento en línea.

```css
.container {
  display: flex; /* or inline-flex */
}
```

3. Propiedades contenedor

- El eje principal puede ser tanto horizontal (valor por defecto) como vertical.
- Del mismo modo los valores de start y end pueden estar a la derecha o a la izquierda.
- Este comportamiento se configura con la propiedad **flex-direction**.

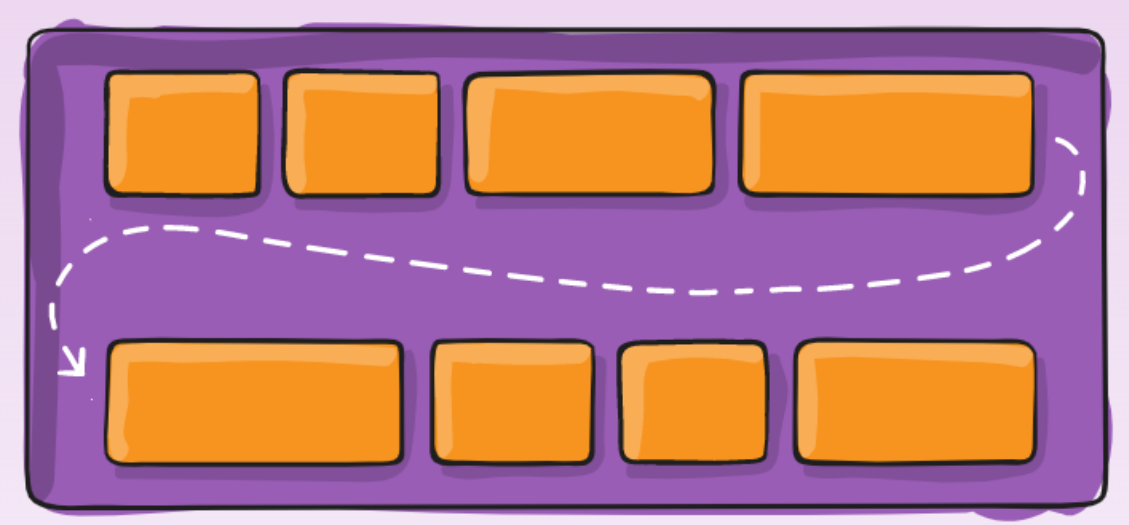
```css
.container {
  flex-direction: row | row-reverse | column | column-reverse;
}
```



3. Propiedades contenedor

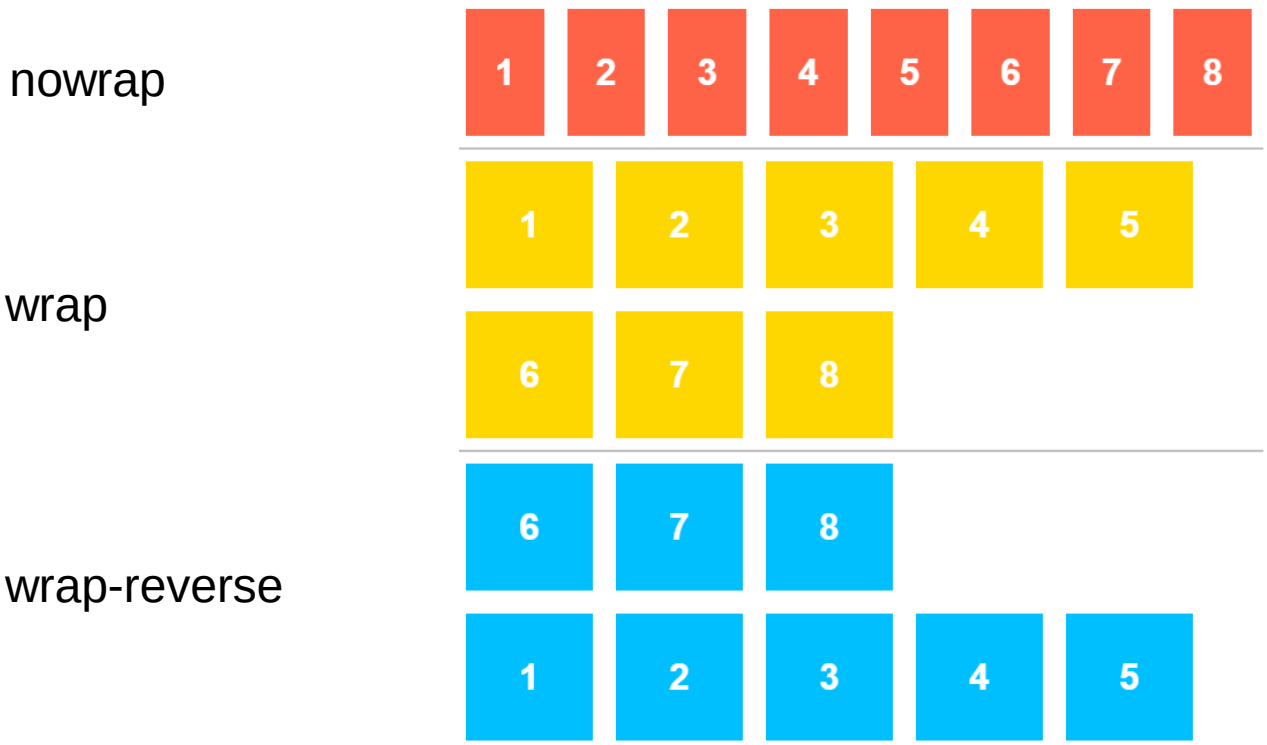
- Por defecto, se intentan ubicar los ítems en una sola línea (nowrap).
- Pero se puede modificar este comportamiento para permitir que salten de línea con la propiedad **flex-wrap**.

```css
.container{
  flex-wrap: nowrap | wrap | wrap-reverse;
}
```



3. Propiedades contenedor

- Ejemplo: Valores de flex-wrap (con flex-direction:row).



3. Propiedades contenedor

- Hay una propiedad *shorthand* llamada **flex-flow** que permite definir las propiedades flex-direction y flex-wrap a la vez.

```css
flex-flow: <‘flex-direction’> || <‘flex-wrap’>
```

3. Propiedades contenedor

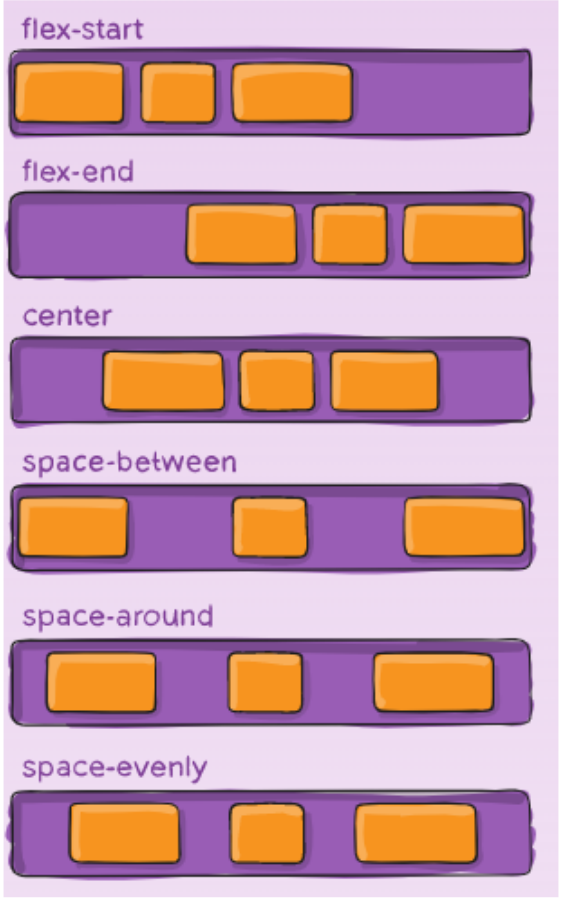
- La propiedad **justify-content** define la alineación respecto al eje principal.

```css
.container {
  justify-content: flex-start | flex-end | center | space-between | space-around | space-evenly | start | end | left | right ... + safe | unsafe;
}
```

- Ayuda a distribuir el espacio sobrante cuando todos los elementos de una línea son inflexibles o son flexibles pero han alcanzado su tamaño máximo.
- También ejerce cierto control sobre la alineación de los elementos cuando desbordan la línea.
  - Definiéndolo como safe no se puede empujar un elemento de modo que se muestre fuera de pantalla (solo Firefox).

3. Propiedades contenedor

- Valores de propiedad **justify-content:**
  - **flex-start** (valor por defecto): Alineación al comienzo del eje principal.
  - **flex-end:** Al final del eje principal.
  - **center:** Centrado en el eje principal.
  - **space-between:** Se reparte el espacio uniformemente entre los ítems, el primer ítem comienza en *main start* y el último finaliza en *main end*.
  - **space-around:** Similar al anterior, pero además se reserva un espacio libre al principio y al final. El espacio entre elementos es el doble que el que hay al principio y al final.
  - **space-evenly:** Se reparte el espacio uniformemente entre los ítems y se reserva espacio al principio y al final (el espacio al principio y al final es el mismo que el que hay entre los elementos).



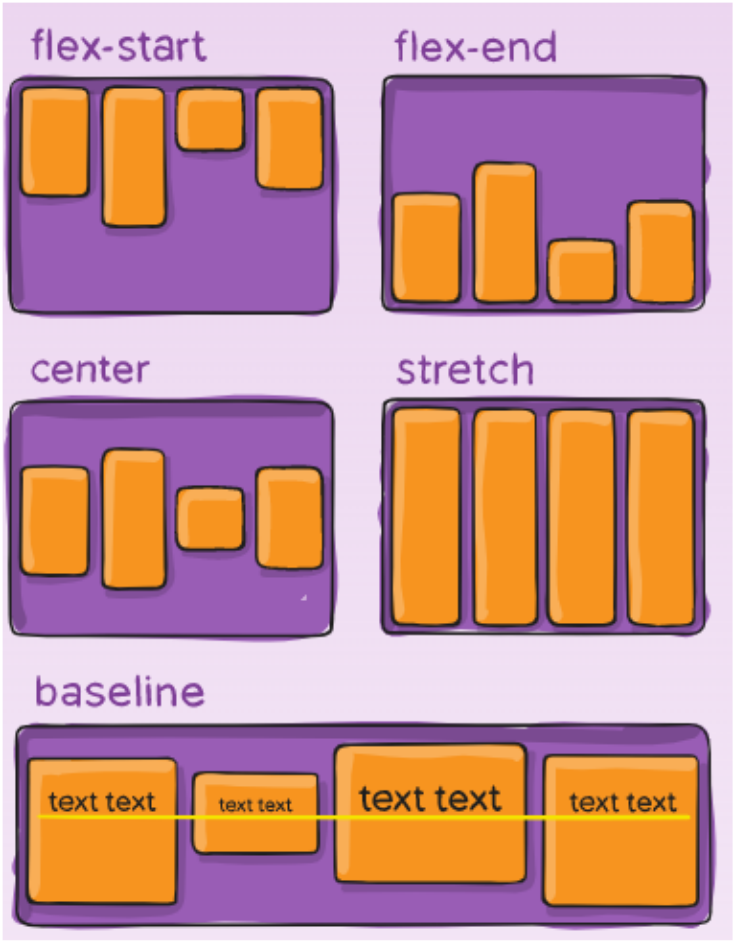
3. Propiedades contenedor

- La propiedad **align-items** define la alineación respecto al eje secundario.

```css
.container {
  align-items: stretch | flex-start | flex-end | center | baseline | first baseline | last baseline | start | end | self-start | self-end + ... safe | unsafe;
}
```

3. Propiedades contenedor

- Valores propiedad **align-ítems:**
  - **stretch** (valor por defecto): Se estira hasta rellenar todo el contenedor (aún respeta las propiedades min-width y max-width).
  - **flex-start / start / self-start**: Se alinean respecto *cross start*.
  - **flex-end / end / self-end**: Se alinean respecto *cross end*.
  - **center**: Se centran en el eje secundario.
  - **baseline**: Los ítems se alinean en la dirección secundaria según la primera línea de texto.



3. Propiedades contenedor

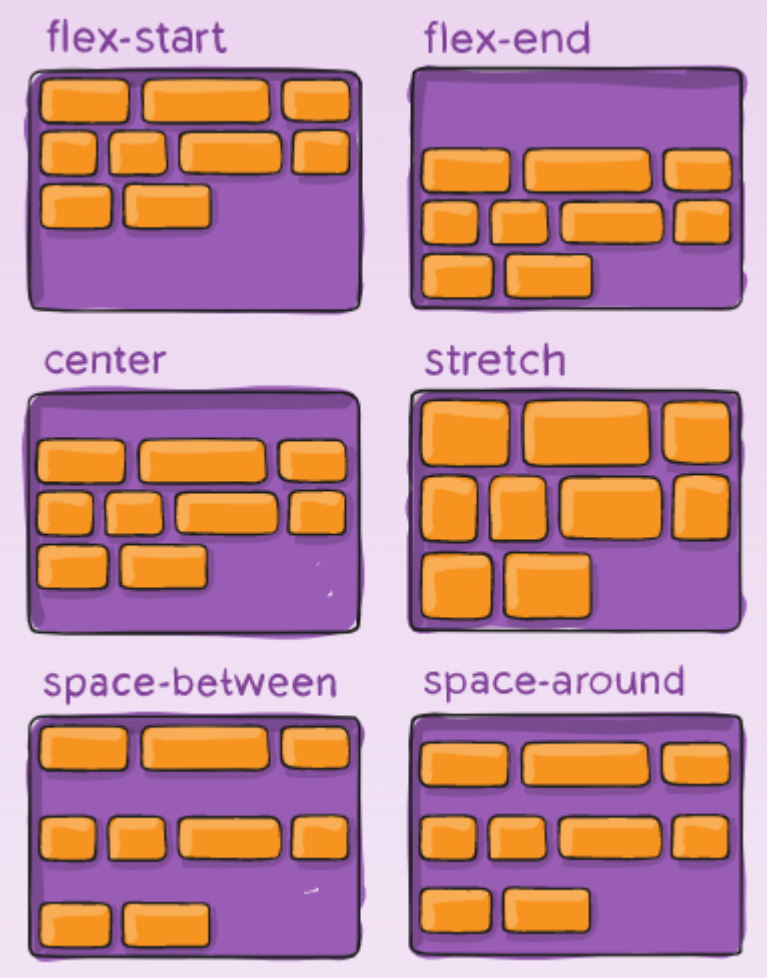
- La propiedad **align-content** define la alineación respecto al eje secundario cuando hay más de una línea.

```css
.container {
  align-content: flex-start | flex-end | center | space-between | space-around | space-evenly | stretch | start | end | baseline | first baseline | last baseline + ... safe | unsafe;
}
```

- Su comportamiento es similar a justify-content pero para el eje secundario.
- No tendrá efecto si solo hay una línea de ítems.

3. Propiedades contenedor

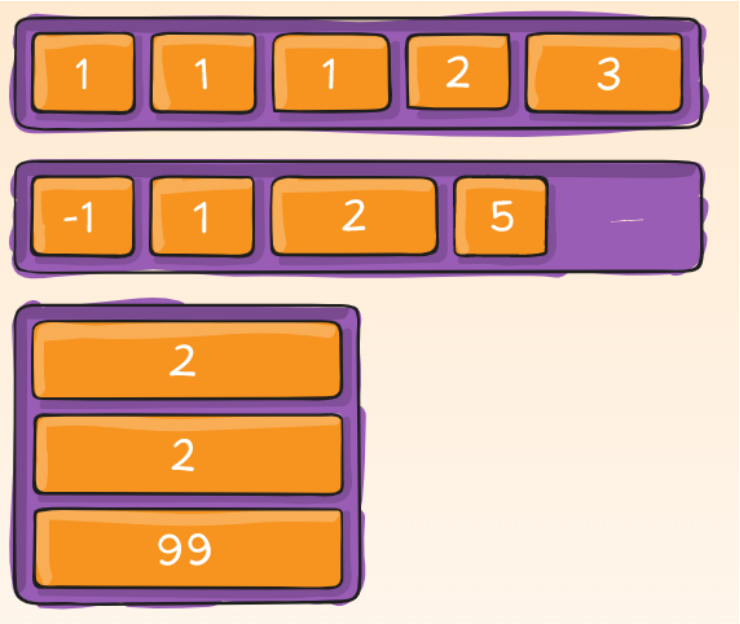
- Valores propiedad **align-content:**



4. Propiedades flex ítems

- Por defecto los ítems se muestran en el orden que aparecen en el código, pero se puede modificar con la propiedad **order**.
- El valor para la propiedad order es un número entero, que indica el orden.
- El valor por defecto para order es 0.

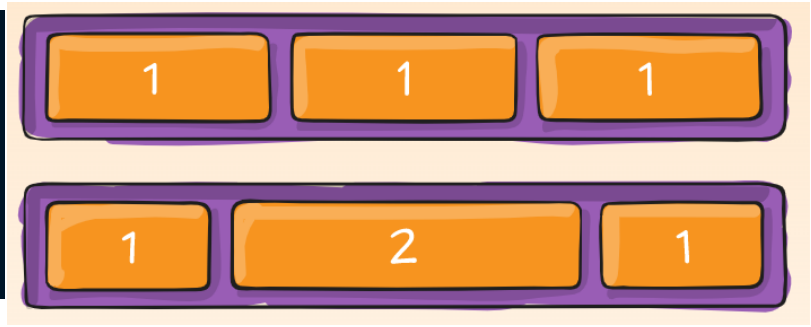
```css
.item {
  order: <integer>; /* default is 0 */
}
```



4. Propiedades flex ítems

- Para controlar la habilidad de un flex ítem de “crecer” cuando es necesario se controla con la propiedad **flex-grow**.
- Le indica al elemento cuanto espacio libre le corresponde.
  - Ejemplo: Si uno de los elementos tiene valor 2 y los demás 1, al elemento que tiene valor 2 le corresponderá el doble de espacio que a los demás.
- flex-grow es 0 (valor por defecto) o un número natural (no acepta valores negativos).

```css
.item {
  flex-grow: <number>; /* default 0 */
}
```



- **flex-shrink:** Es similar, pero indica la proporción en que se estrechan los elementos cuando su contenido no cabe en el espacio disponible.

4. Propiedades flex ítems

- **flex-grow**:

Uno	Dos	Tres (clase "mayor")	Cuatro

- **flex-shrink:**

Uno, con bastante texto, así que los elementos no van a caber en una sola línea	Dos, con bastante texto, así que los elementos no van a caber en una sola línea	Tres (clase "menor"), con una cantidad de texto similar al resto	Cuatro, con bastante texto, así que los elementos no van a caber en una sola línea

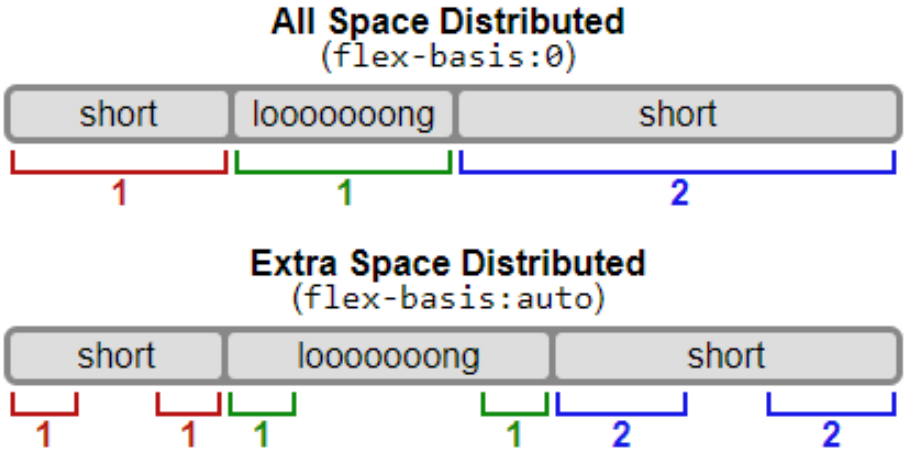
4. Propiedades flex ítems

- La propiedad **flex-basis** define el tamaño de un elemento antes de que el espacio restante sea distribuido (ej. 20%, 5em, etc.).
- Con el valor **auto** (valor por defecto), se tomaría el valor de width o height.
- El valor **content** indica que su valor está basado en tamaño del contenido del elemento (solo Firefox).

```css
.item {
  flex-basis: <length> | auto; /* default auto */
}
```

4. Propiedades flex ítems

- Su valor influye en el comportamiento de la propiedad **flex-grow** del siguiente modo:



4. Propiedades flex ítems

- Hay una propiedad shorthand llamada **flex** que permite definir flex-grow, flex-shrink y flex-basis a la vez:

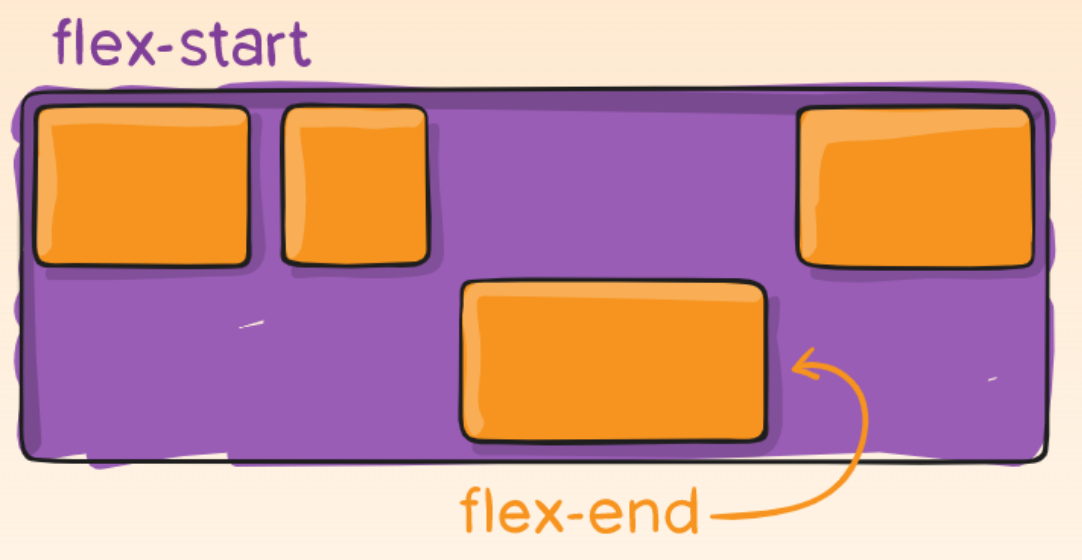
```css
.item {
  flex: none | [ <'flex-grow'> <'flex-shrink'>? || <'flex-basis'> ]
}
```

- Sus valores por defecto son 0, 1 y auto.

4. Propiedades flex ítems

- La propiedad **align-self** permite definir una alineación de manera similar a align-items pero para un ítem en concreto:

```css
.item {
  align-self: auto | flex-start | flex-end | center | baseline | stretch;
}
```



5. Ejemplos prácticos

- flexboxfroggy.com/#es

5. Ejemplos prácticos

- http://www.flexboxdefense.com/

Referencias

- CSS-TRICKS. *A complete Guide to Flexbox*.
https://css-tricks.com/snippets/css/a-guide-to-flexbox/

- Chris Coyer. Flex-wrap. CSS-TRICKS.
https://css-tricks.com/almanac/properties/f/flex-wrap/

- Bartolomé Sintes Marco, Mclibre.
http://www.mclibre.org/consultar/htmlcss/css/css-flexbox.html

- MDN, Conceptos Básicos de Flexbox.
https://developer.mozilla.org/es/docs/Web/CSS/CSS_Flexible_Box_Layout/Conceptos_Basicos_de_Flexbox

- Casos de uso típicos de Flexbox.
https://developer.mozilla.org/es/docs/Web/CSS/CSS_Flexible_Box_Layout/Casos_de_uso_tipicos_de_Flexbox.

- W3C. CSS Flexible Box Layout Module.
https://www.w3.org/TR/css-flexbox-1/#propdef-flex-grow