

Lenguaje de Marcas y Sistemas de Gestión

UA 2.18 - Introducción CSS: Flexbox



**Universidad
Europea**

LAUREATE INTERNATIONAL UNIVERSITIES

Raúl Rodríguez Mercado

raul.rodriguez@universidadeuropea.es / @raulrodriguezue

Dpto. Ciencias y Tecnología de la Informática y Comunicación

UA 2.18: Introducción CSS: Flexbox



Objetivos

- ☐ Definir qué es CSS
- ☐ Conocer la estructura de las hojas de estilo y como se aplican a los documentos HTML



UA 2.18: Introducción CSS: Flexbox



Contenidos

- Flexbox



UA 2.18: Introducción CSS: Flexbox



Flexbox en CSS

- ❑ Hasta ahora en CSS, se ha estado utilizando el posicionamiento (static, relative, absolute...), los elementos en línea o en bloque (y derivados) o los **float**, para agrupar elementos.
- ❑ Este sistema, no dejaba a los diseñadores aclimatarnos a las necesidades actuales de aplicaciones de escritorio, dispositivos móviles, etc.
- ❑ **Flexbox** es un sistema de **elementos flexibles** que llega con la idea de olvidar estos mecanismos y acostumbrarnos a una mecánica más potente, limpia y personalizable, en la que los elementos HTML se adaptan y colocan automáticamente y es más fácil personalizar los diseños.



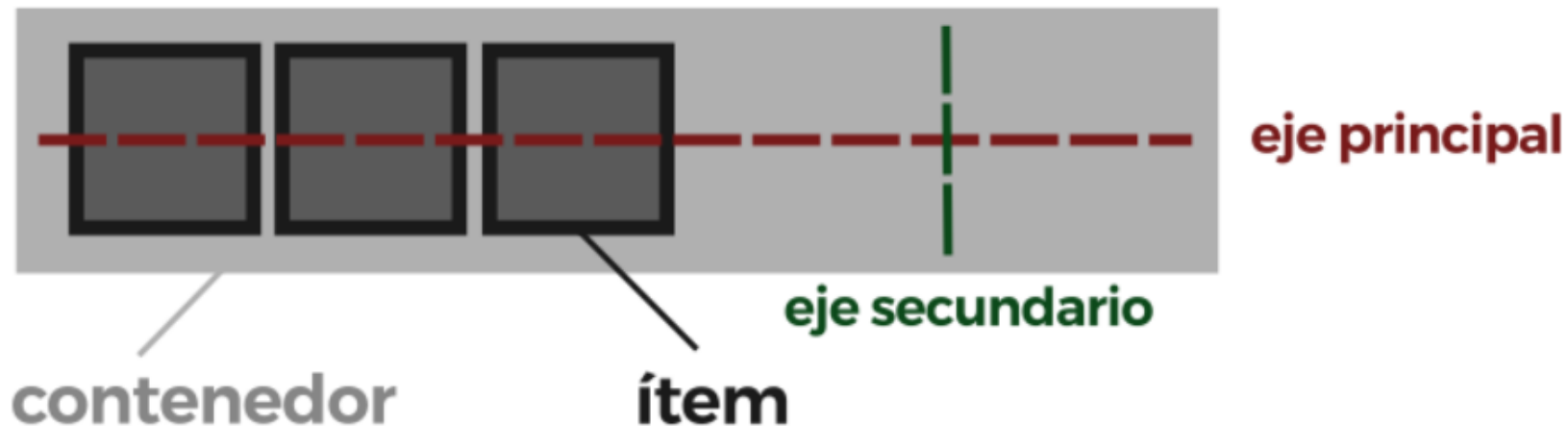
Flexbox

UA 2.18: Introducción CSS: Flexbox



Conceptos de Flexbox

- ❑ **Contenedor:** Existe un elemento padre que es el contenedor que tendrá en su interior cada uno de los ítems flexibles y adaptables.
- ❑ **Ítem:** Cada uno de los hijos flexibles que tendrá el contenedor en su interior.
- ❑ **Eje principal:** Los contenedores flexibles tendrán una orientación principal específica. Por defecto, es en horizontal (fila).
- ❑ **Eje secundario:** De la misma forma, los contenedores flexibles tendrán una orientación secundaria, perpendicular a la principal. Si la principal es en horizontal, la secundaria será en vertical, y viceversa.



UA 2.18: Introducción CSS: Flexbox



Conceptos de Flexbox

- ❑ Para activar el modo **flexbox** hay que utilizar sobre el elemento contenedor la propiedad ***display***, y especificar el valor **flex** o **inline-flex** dependiendo de como queramos que se comporte el contenedor: si como un elemento en línea, o como un elemento en bloque.

Tipo de elemento	Descripción
inline-flex	Establece un contenedor de ítems flexible en línea, de forma equivalente a inline-block.
flex	Establece un contenedor de ítems flexible en bloque, de forma equivalente a block.

- ❑ Por defecto, y sólo con esto, observaremos que los elementos se disponen todos sobre una misma línea. Esto ocurre porque estamos utilizando el modo **flexbox** y estaremos trabajando con ítems flexibles básicos, garantizando que no se desbordarán ni mostrarán los problemas que tienen los porcentajes sobre elementos que no utilizan flexbox.

UA 2.18: Introducción CSS: Flexbox



Conceptos de Flexbox

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Flex</title>
  <meta name="viewport" content="width=device-width, user-scalable=no, initial-scale=1.0,
maximum-scale=1.0, minimum-scale=1.0">
  <link rel="stylesheet" href="flexbox1.css">
</head>
<body>
  <div class="contenedor">
    <div class="hijo">A</div>
    <div class="hijo">B</div>
    <div class="hijo">C</div>
  </div>
</body>
</html>
```

```
*{
  margin: 0;
  padding: 0;
  -webkit-box-sizing: border-box;
  -moz-box-sizing: border-box;
  box-sizing: border-box;
}
.contenedor{
  width: 80%;
  height: 500px;
  background: #FF5C29;
  margin: 50px auto;
  display: flex;
}
.hijo{
  background: #00B285;
  width: 200px;
  height: 100px;
  color: #fff;
  margin: 30px;
  padding: 20px;
  font-size: 40px;
  text-align: center;
}
```

UA 2.18: Introducción CSS: Flexbox



Conceptos de Flexbox



Con `display: flex;` habilitamos los elementos contenedor. La misma función que obteníamos con `float: left`, ahora lo hacemos con flex, posicionando los elementos uno detrás del otro

```
*{
  margin: 0;
  padding: 0;
  -webkit-box-sizing: border-box;
  -moz-box-sizing: border-box;
  box-sizing: border-box;
}
.contenedor{
  width: 80%;
  height: 500px;
  background: #FF5C29;
  margin: 50px auto;
  display: flex;
}
.hijo{
  background: #00B285;
  width: 200px;
  height: 100px;
  color: #fff;
  margin: 30px;
  padding: 20px;
  font-size: 40px;
  text-align: center;
}
```


UA 2.18: Introducción CSS: Flexbox



Conceptos de Flexbox: Dirección de los Ejes

- ❑ Para manipular la dirección y comportamiento de los ítems a lo largo del eje principal del contenedor, podemos utilizar las siguientes propiedades:

Propiedad	Valor	Significado
flex-direction:	row row-reverse column column-reverse	Cambia la orientación del eje principal.
flex-wrap:	nowrap wrap wrap-reverse	Evita o permite el desbordamiento (multilinea).

- ❑ La propiedad ***flex-direction*** permite modificar la dirección del eje principal del contenedor para que se oriente en horizontal o vertical, y con el sufijo ***-reverse***, indicar el orden de los ítems de manera inversa.

Valor	Descripción
row	Establece la dirección del eje principal en horizontal.
row-reverse	Establece la dirección del eje principal en horizontal (invertido).
column	Establece la dirección del eje principal en vertical.
column-reverse	Establece la dirección del eje principal en vertical (invertido).

UA 2.18: Introducción CSS: Flexbox

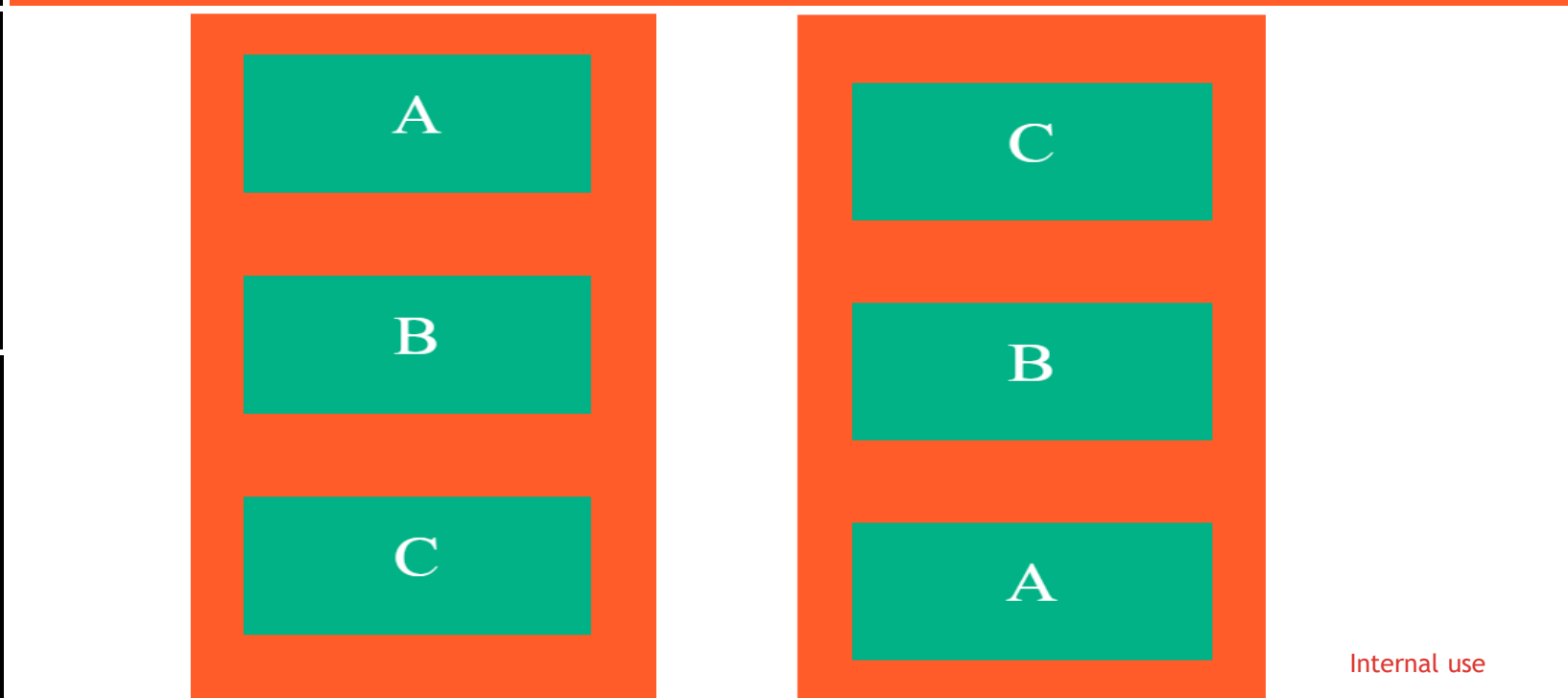


Conceptos de Flexbox: Dirección de los Ejes

```
.contenedor{  
  width: 80%;  
  height: 500px;  
  background: #FF5C29;  
  margin: 50px auto;  
  display: flex;  
  flex-direction: row-reverse;  
}
```



```
.contenedor{  
  width: 80%;  
  height: 500px;  
  background: #FF5C29;  
  margin: 50px auto;  
  display: flex;  
  flex-direction: column;  
}
```



```
.contenedor{  
  width: 80%;  
  height: 500px;  
  background: #FF5C29;  
  margin: 50px auto;  
  display: flex;  
  flex-direction: column-reverse;  
}
```

Conceptos de Flexbox: Dirección de los Ejes

- ❑ Por otro lado, existe otra propiedad llamada ***flex-wrap*** con la que podemos especificar el comportamiento del contenedor respecto a evitar que se desborde (nowrap, valor por defecto) o permitir que lo haga, en cuyo caso, estaríamos hablando de un **contenedor flexbox multilinea**.

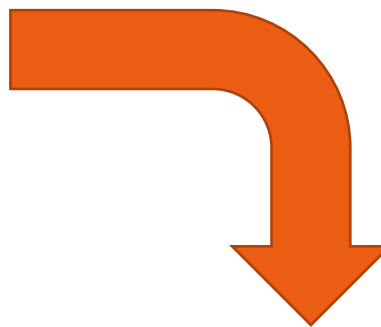
Valor	Descripción
nowrap	Establece los ítems en una sola línea (no permite que se desborde el contenedor).
wrap	Establece los ítems en modo multilínea (permite que se desborde el contenedor).
wrap-reverse	Establece los ítems en modo multilínea, pero en dirección inversa.

UA 2.18: Introducción CSS: Flexbox



Conceptos de Flexbox: Dirección de los Ejes

```
.contenedor{  
  width: 80%;  
  height: 500px;  
  background: #FF5C29;  
  margin: 50px auto;  
  display: flex;  
  flex-direction: row;  
  flex-wrap: nowrap; /*No permite que  
    el desbordamiento y se ajustan las  
    cajas al contenedor padre*/  
}
```



Con *nowrap*, no se permite el desbordamiento de los elementos y estos se hacen más pequeños según el tamaño del contenedor padre.

A

B

C

C

C

C

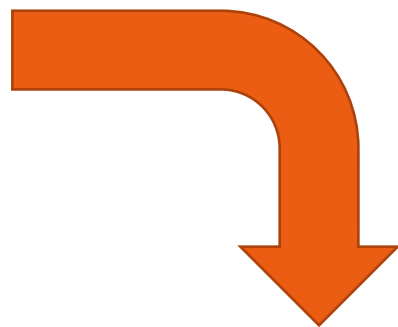
C

UA 2.18: Introducción CSS: Flexbox

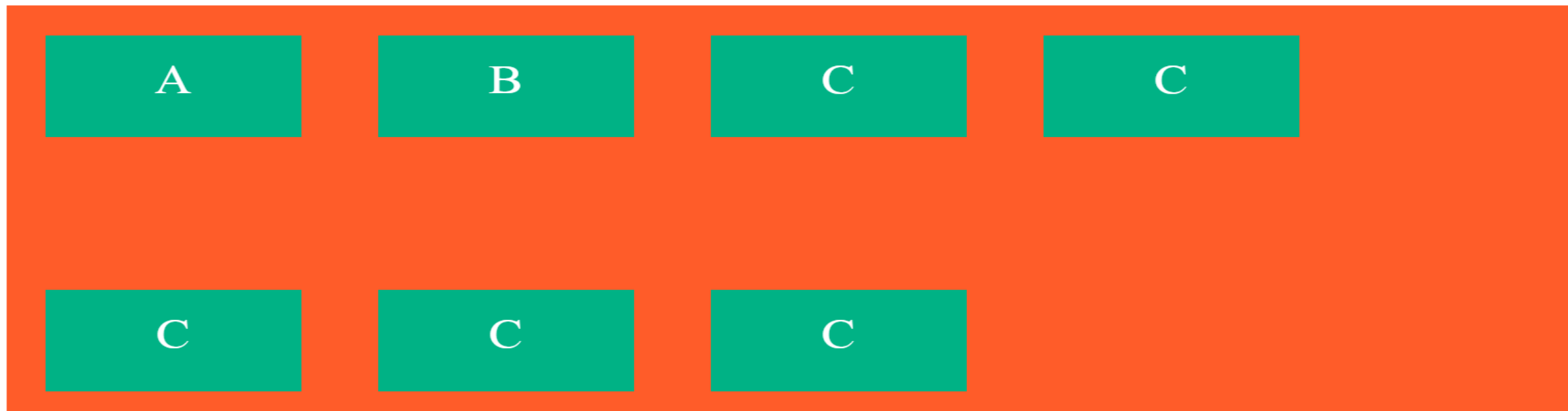


Conceptos de Flexbox: Dirección de los Ejes

```
.contenedor{  
  width: 80%;  
  height: 500px;  
  background: #FF5C29;  
  margin: 50px auto;  
  display: flex;  
  flex-direction: row;  
  flex-wrap: wrap; /* Permite  
    el desbordamiento y se respeta el  
    tamaño de las cajas y se crean  
    nuevas líneas */  
}
```



Con **wrap**, se permite el desbordamiento de los elementos y estos se hacen que se creen una línea debajo respetando el tamaño de las cajas.



UA 2.18: Introducción CSS: Flexbox



Conceptos de Flexbox: Alineación de Ítems

Propiedad	Valor	Actúa sobre
justify-content:	flex-start flex-end center space-between space-around	Eje principal
align-content:	flex-start flex-end center space-between space-around stretch	Eje principal
align-items:	flex-start flex-end center stretch baseline	Eje secundario
align-self:	auto flex-start flex-end center stretch baseline	Eje secundario

- ❑ De esta pequeña lista, nos centraremos en la primera y la tercera propiedad, que son las más importantes. Las otras dos son casos más particulares y que también hay que tener en cuenta:
 - ✓ **justify-content:** Se utiliza para alinear los ítems del **eje principal** (por defecto, el horizontal).
 - ✓ **align-items:** Usada para alinear los ítems del **eje secundario** (por defecto, el vertical).

UA 2.18: Introducción CSS: Flexbox



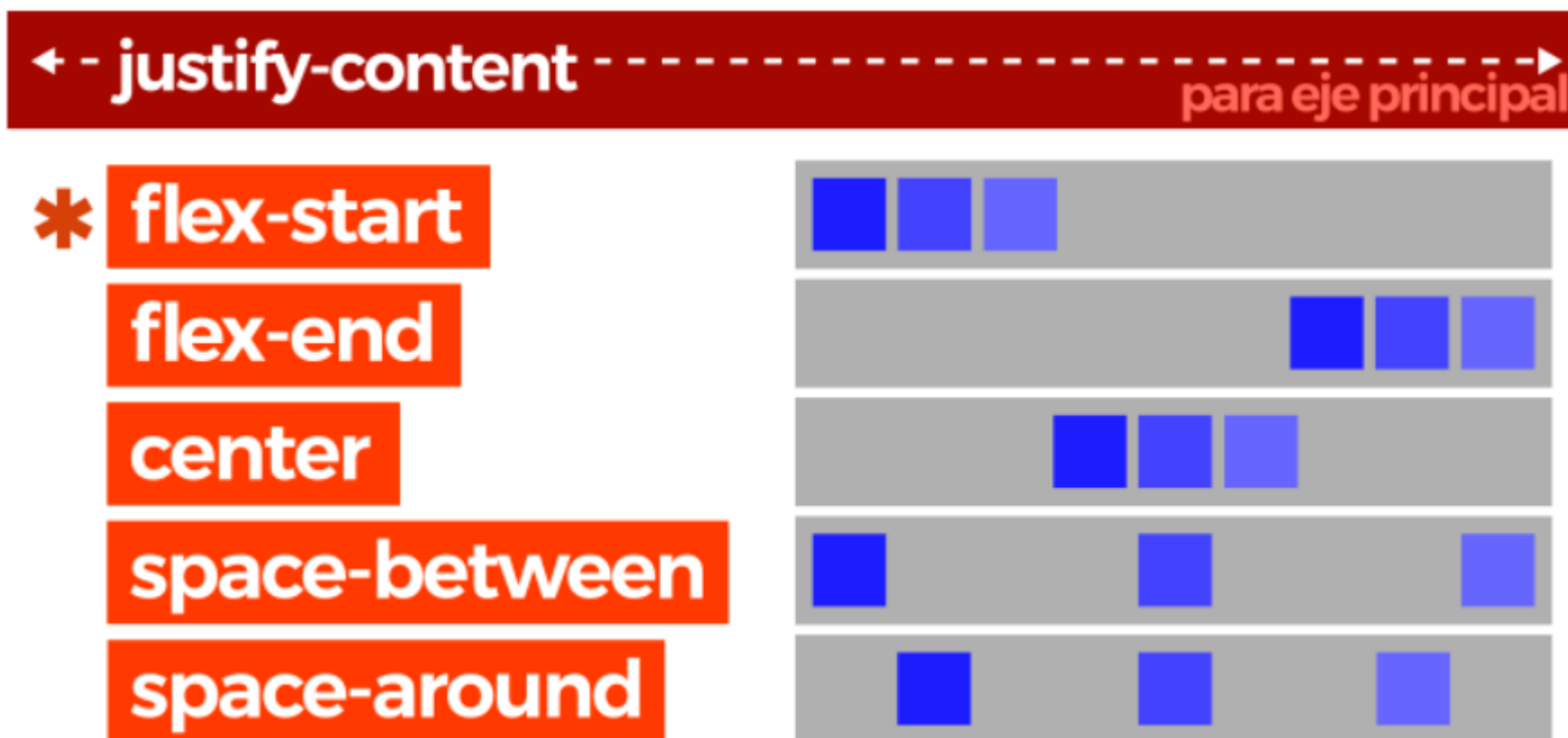
Alineación de Ítems sobre el Eje Principal: *justify-content*

- ❑ La primera propiedad, ***justify-content***, sirve para colocar los ítems de un contenedor mediante una disposición concreta a lo largo del **eje principal**:

Valor	Descripción
flex-start	Agrupar los ítems al principio del eje principal.
flex-end	Agrupar los ítems al final del eje principal.
center	Agrupar los ítems al centro del eje principal.
space-between	Distribuye los ítems dejando (el mismo) espacio entre ellos.
space-around	Distribuye los ítems dejando (el mismo) espacio a ambos lados de cada uno de ellos.

Alineación de Ítems sobre el Eje Principal: *justify-content*

- ❑ Con cada uno de estos valores, modificaremos la disposición de los ítems del contenedor donde se aplica, pasando a colocarse como se ve en la imagen siguiente (fijaros en las diferentes tonalidades azules para indicar las posiciones de cada ítem):



UA 2.18: Introducción CSS: Flexbox



Alineación de Ítems sobre el Eje Principal: justify-content

A

B

C

```
.contenedor{  
  width: 80%;  
  height: 500px;  
  background: #FF5C29;  
  margin: 50px auto;  
  display: flex;  
  flex-direction: row;  
  justify-content: flex-start;  
}
```

```
.contenedor{  
  width: 80%;  
  height: 500px;  
  background: #FF5C29;  
  margin: 50px auto;  
  display: flex;  
  flex-direction: row;  
  justify-content: flex-end;  
}
```

A

B

C

A

B

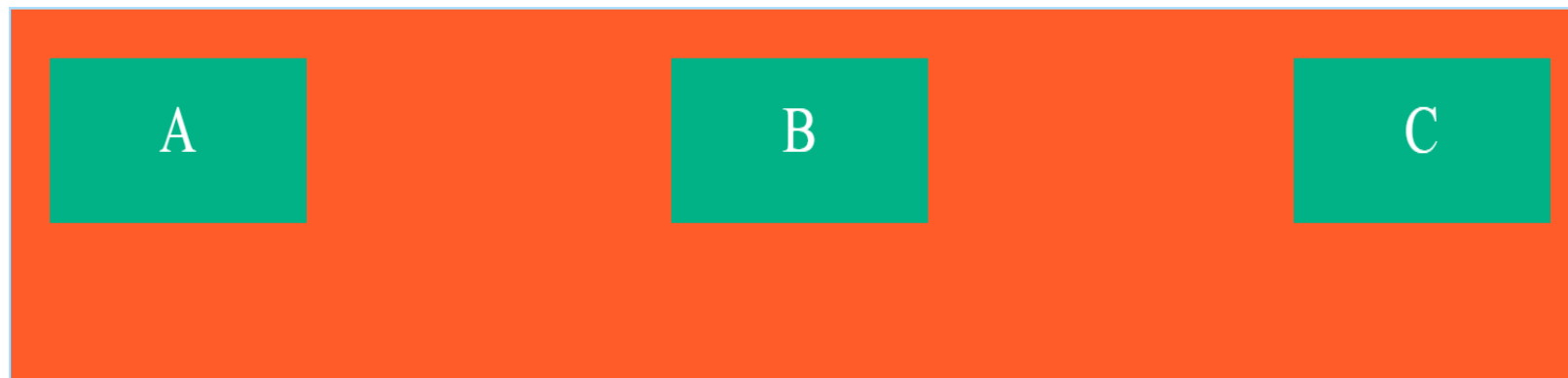
C

```
.contenedor{  
  width: 80%;  
  height: 500px;  
  background: #FF5C29;  
  margin: 50px auto;  
  display: flex;  
  flex-direction: row;  
  justify-content: center;  
}
```

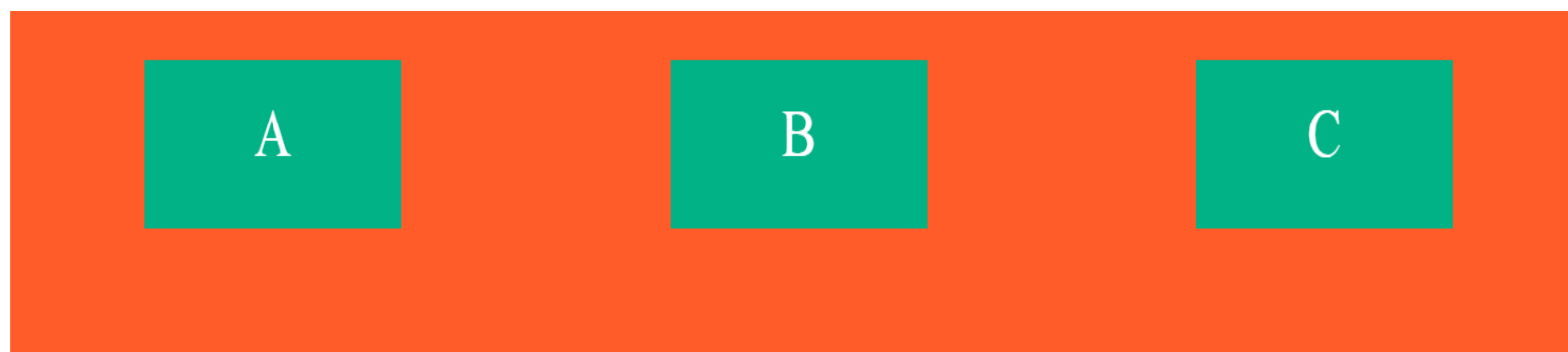
UA 2.18: Introducción CSS: Flexbox



Alineación de Ítems sobre el Eje Principal: justify-content



```
.container{  
  width: 80%;  
  height: 500px;  
  background: #FF5C29;  
  margin: 50px auto;  
  display: flex;  
  flex-direction: row;  
  justify-content: space-between;  
}
```



```
.container{  
  width: 80%;  
  height: 500px;  
  background: #FF5C29;  
  margin: 50px auto;  
  display: flex;  
  flex-direction: row;  
  justify-content: space-around;  
}
```

UA 2.18: Introducción CSS: Flexbox



Alineación de Ítems sobre el Eje Secundario: *align-items*

- ❑ La otra propiedad importante de este apartado es ***align-items***, que se encarga de alinear los ítems en el eje secundario del contenedor.
- ❑ Tener cuidado de no confundir ***align-content*** con ***align-items***, puesto que el primero actúa sobre cada una de las líneas de un contenedor multilinea (no tiene efecto sobre contenedores de una sola línea), mientras que ***align-items*** lo hace sobre la línea actual.

Valor	Descripción
flex-start	Alinea los ítems al principio del eje secundario.
flex-end	Alinea los ítems al final del eje secundario.
center	Alinea los ítems al centro del eje secundario.
stretch	Alinea los ítems estirándolos de modo que cubran desde el inicio hasta el final del contenedor.
baseline	Alinea los ítems en el contenedor según la base del contenido de los ítems del contenedor.

UA 2.18: Introducción CSS: Flexbox



Alineación de Ítems sobre el Eje Secundario: align-items



UA 2.18: Introducción CSS: Flexbox



Alineación de Ítems sobre el Eje Secundario: align-items

```
.contenedor{  
  width: 80%;  
  height: 500px;  
  background: #FF5C29;  
  margin: 50px auto;  
  display: flex;  
  flex-direction: row;  
  justify-content: space-around;  
  align-items: flex-end;  
}
```

A

B

C

A

B

C

```
.contenedor{  
  width: 80%;  
  height: 500px;  
  background: #FF5C29;  
  margin: 50px auto;  
  display: flex;  
  flex-direction: row;  
  justify-content: space-around;  
  align-items: flex-start;  
}
```

A

B

C

```
.contenedor{  
  width: 80%;  
  height: 500px;  
  background: #FF5C29;  
  margin: 50px auto;  
  display: flex;  
  flex-direction: row;  
  justify-content: space-around;  
  align-items: center;  
}
```

A

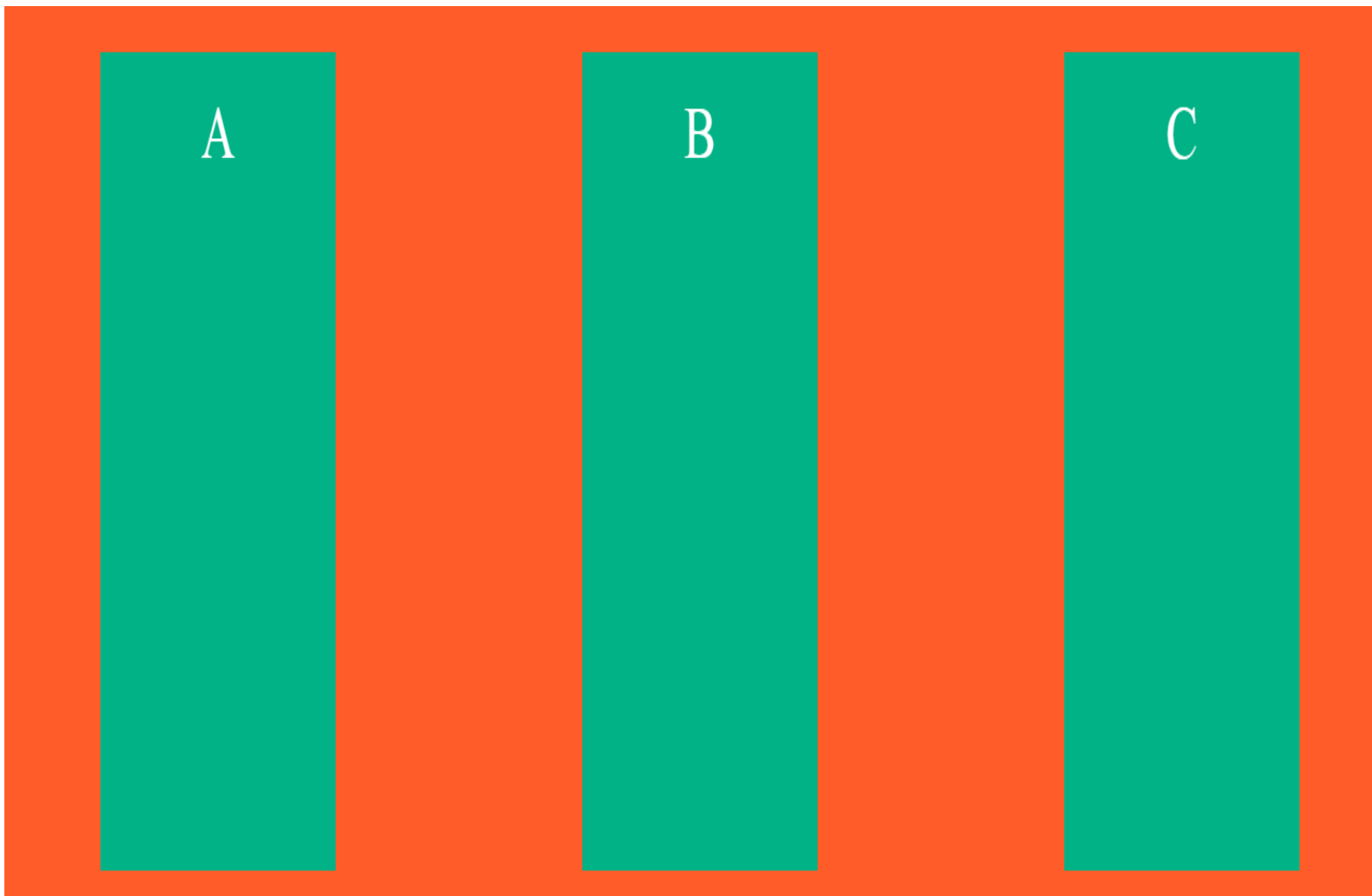
B

C

UA 2.18: Introducción CSS: Flexbox



Alineación de Ítems sobre el Eje Secundario: align-items



```
.contenedor{
  width: 80%;
  height: 500px;
  background: #FF5C29;
  margin: 50px auto;
  display: flex;
  flex-direction: row;
  justify-content: space-around;
  align-items: stretch;
  /*Stretch funciona cuando le quitamos
  la propiedad de altura a las cajas
  hijos*/
}

.hijo{
  background: #00B285;
  width: 200px;
  /*height: 100px;*/
  color: #fff;
  margin: 30px;
  padding: 20px;
  font-size: 40px;
  text-align: center;
}
```

UA 2.18: Introducción CSS: Flexbox



Alineación de Ítems sobre el Eje Secundario: align-items

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Flex</title>
  <meta name="viewport" content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">
  <link rel="stylesheet" href="flexbox1.css">
</head>
<body>
  <div class="contenedor">
    <div class="hijo">A</div>
    <div class="hijo">B</div>
    <div class="hijo otro">C</div>
  </div>
</body>
</html>
```

```
.contenedor{
  width: 80%;
  height: 500px;
  background: #ff5c29;
  margin: 50px auto;
  display: flex;
  flex-direction: row;
  justify-content: space-around;
  align-items: baseline;
  /*baseline alinea todo en base a
  la tipografía que tenemos*/
}

.hijo{
  background: #00b285;
  width: 200px;
  height: 100px;
  color: #fff;
  margin: 30px;
  padding: 20px;
  font-size: 40px;
  text-align: center;
}

.otro{
  font-size: 60px;
  font-family: cursive;
}
```

A

B

C

Alineación de Ítems sobre el Eje Principal: *align-content*

- ❑ La propiedad ***align-content***, es un caso particular del anterior. Nos servirá cuando estemos tratando con un contenedor flex multilinea, que es un contenedor en el que los ítems no caben en el ancho disponible, y por lo tanto, el eje principal se divide en múltiples líneas.
- ❑ De esta forma, ***align-content*** servirá para alinear cada una de las líneas del contenedor multilinea. Los valores que puede tomar son los siguientes:

Valor	Descripción
flex-start	Agrupar los ítems al principio del eje principal.
flex-end	Agrupar los ítems al final del eje principal.
center	Agrupar los ítems al centro del eje principal.
space-between	Distribuye los ítems desde el inicio hasta el final.
space-around	Distribuye los ítems dejando el mismo espacio a los lados de cada uno.
stretch	Estira los ítems para ocupar de forma equitativa todo el espacio.

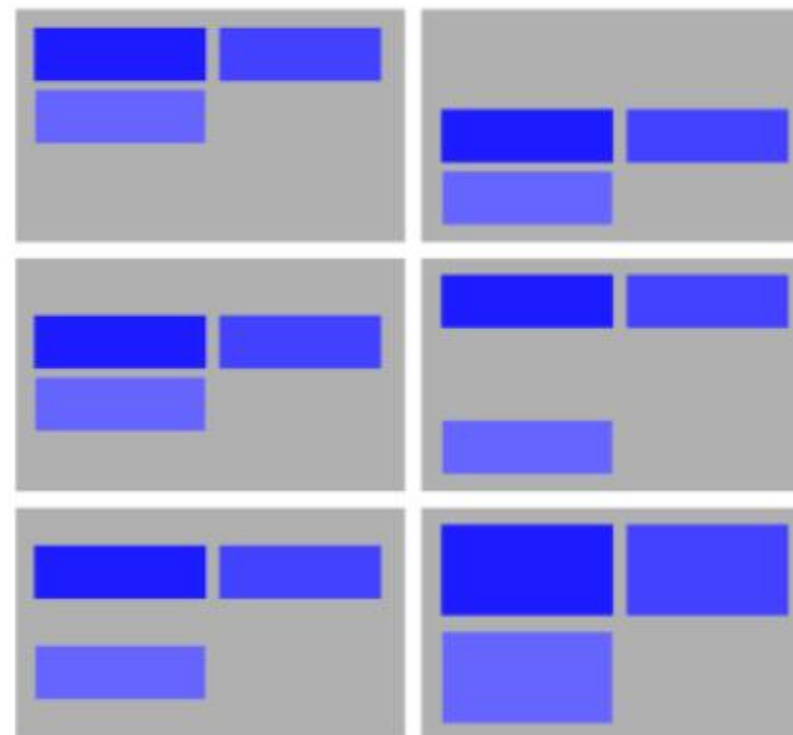
UA 2.18: Introducción CSS: Flexbox



Alineación de Ítems sobre el Eje Principal: align-content



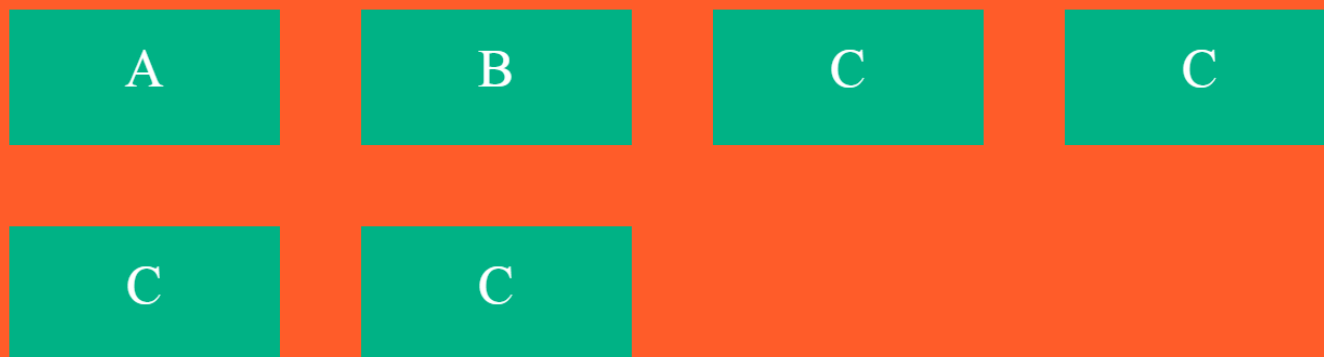
- flex-start**
- flex-end**
- center**
- space-between**
- space-around**
- * stretch**



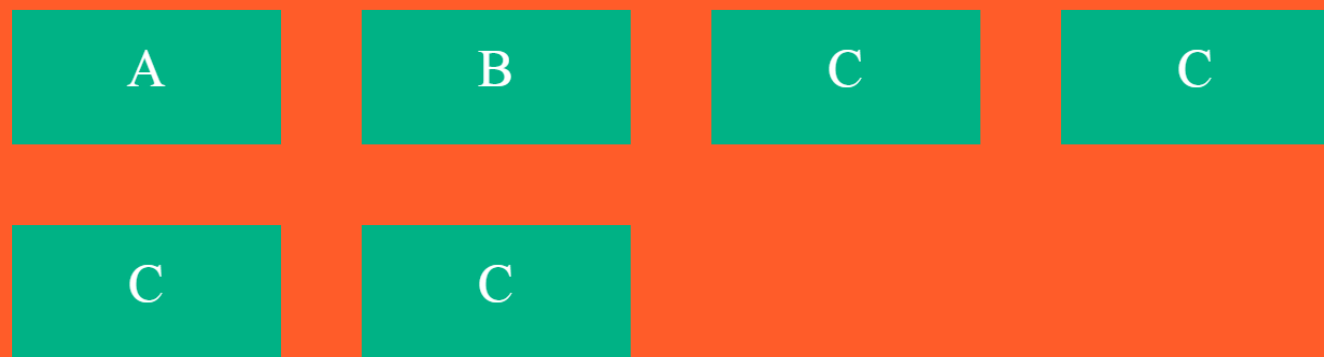
UA 2.18: Introducción CSS: Flexbox



Alineación de Ítems sobre el Eje Principal: align-content



```
.contenedor{  
  width: 80%;  
  height: 500px;  
  background: #FF5C29;  
  margin: 50px auto;  
  display: flex;  
  flex-direction: row;  
  flex-wrap: wrap;  
  align-content: flex-end;  
}
```

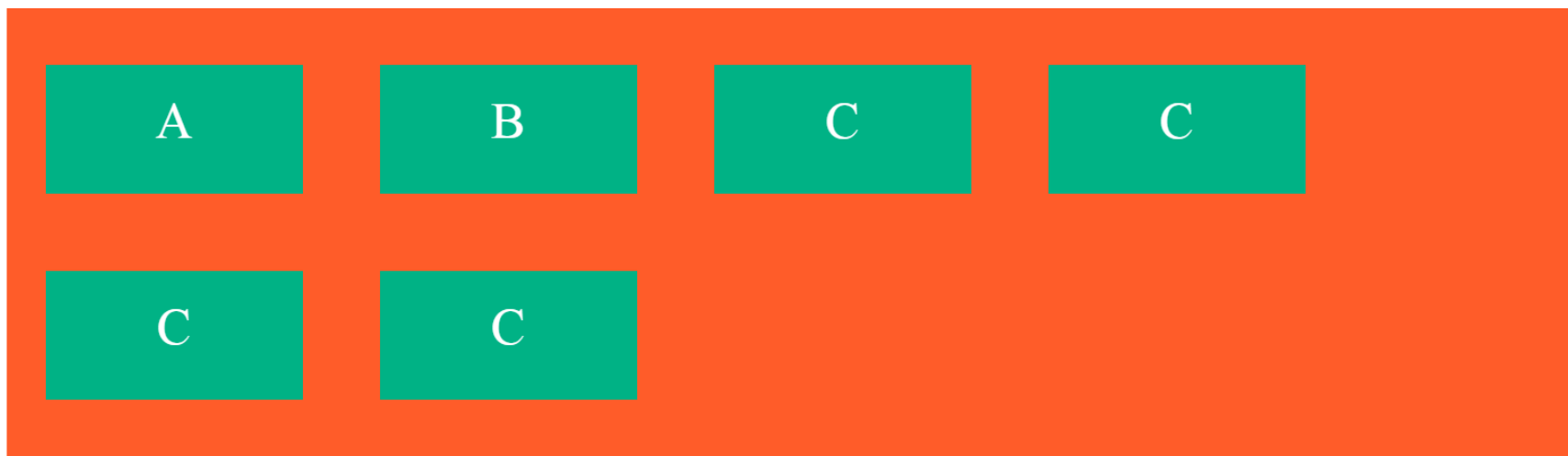


```
.contenedor{  
  width: 80%;  
  height: 500px;  
  background: #FF5C29;  
  margin: 50px auto;  
  display: flex;  
  flex-direction: row;  
  flex-wrap: wrap;  
  align-content: flex-start;  
}
```

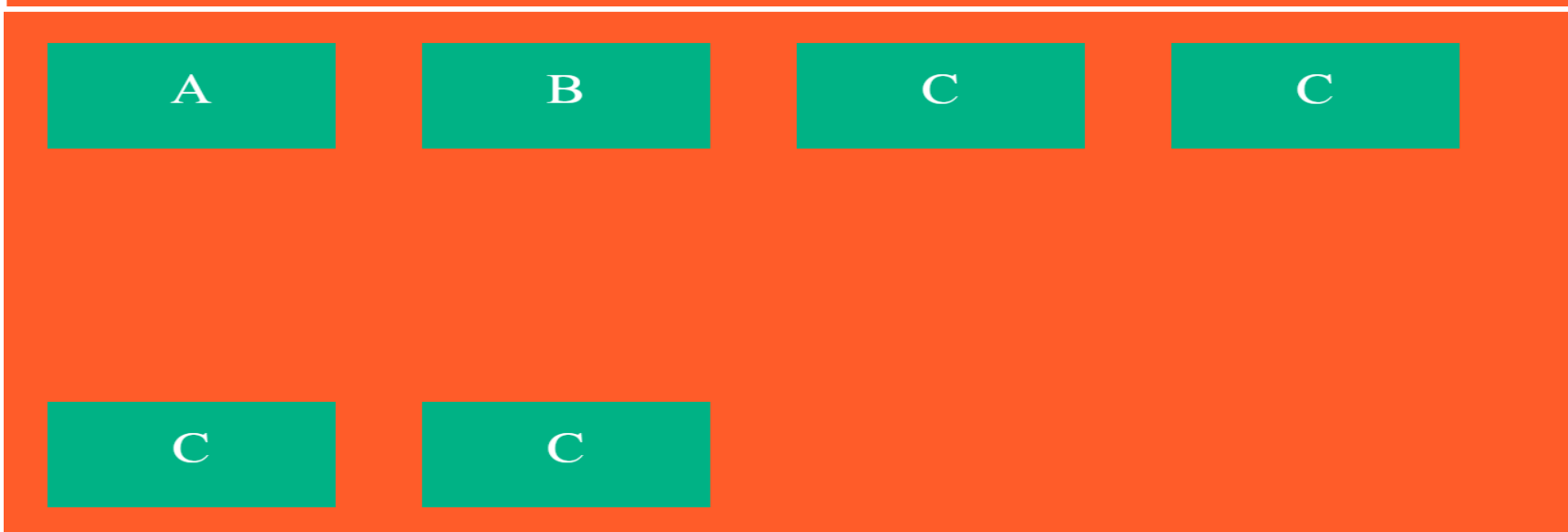
UA 2.18: Introducción CSS: Flexbox



Alineación de Ítems sobre el Eje Principal: align-content



```
.contenedor{  
  width: 80%;  
  height: 500px;  
  background: #FF5C29;  
  margin: 50px auto;  
  display: flex;  
  flex-direction: row;  
  flex-wrap: wrap;  
  align-content: center;  
}
```

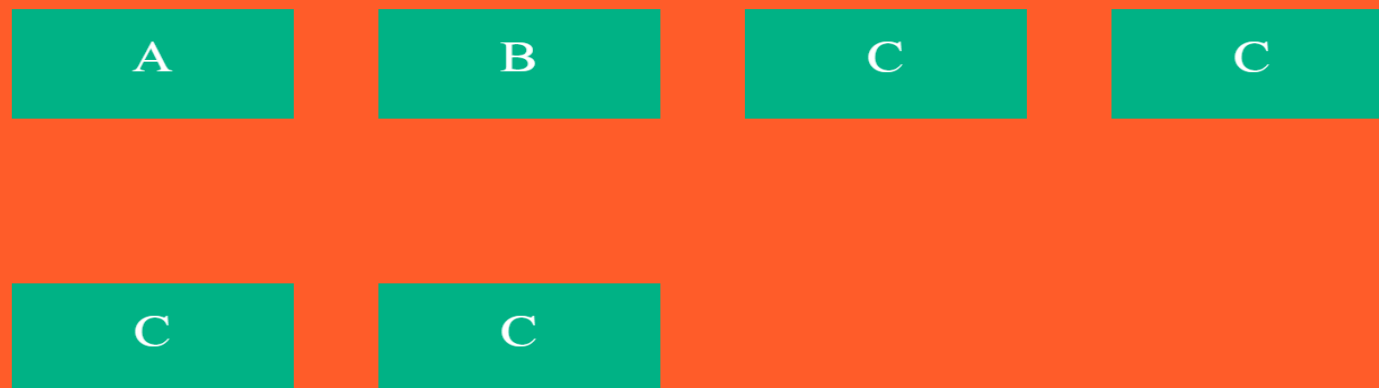


```
.contenedor{  
  width: 80%;  
  height: 500px;  
  background: #FF5C29;  
  margin: 50px auto;  
  display: flex;  
  flex-direction: row;  
  flex-wrap: wrap;  
  align-content: space-between;  
}
```

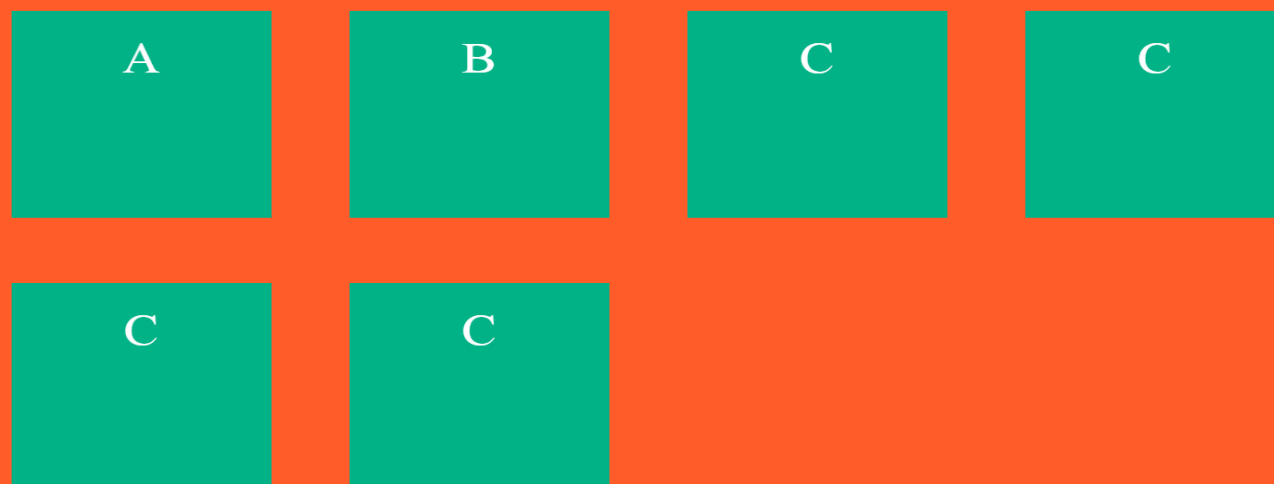
UA 2.18: Introducción CSS: Flexbox



Alineación de Ítems sobre el Eje Principal: align-content



```
.contenedor{  
  width: 80%;  
  height: 500px;  
  background: #FF5C29;  
  margin: 50px auto;  
  display: flex;  
  flex-direction: row;  
  flex-wrap: wrap;  
  align-content: space-around;  
}
```



```
.contenedor{  
  width: 80%;  
  height: 500px;  
  background: #FF5C29;  
  margin: 50px auto;  
  display: flex;  
  flex-direction: row;  
  flex-wrap: wrap;  
  align-content: stretch;  
}  
  
.hijo{  
  background: #00B285;  
  width: 200px;  
  /*height: 100px;*/  
  color: #fff;  
  margin: 30px;  
  padding: 20px;  
  font-size: 40px;  
  text-align: center;  
}
```

UA 2.18: Introducción CSS: Flexbox



Propiedades de ítems hijos

- ❑ A excepción de la propiedad ***align-self*** que veremos a continuación, todas las propiedades que hemos visto hasta ahora se aplican sobre el elemento **contenedor**. Las siguientes propiedades se aplican sobre los ítems hijos:

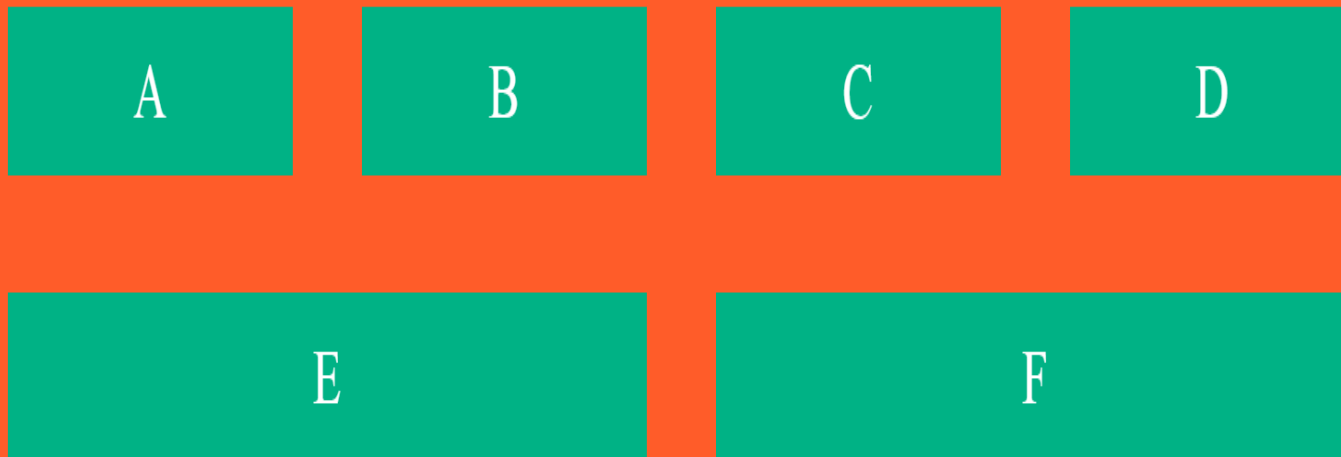
Propiedad	Valor	Descripción
flex-grow:	0 <u>[factor de crecimiento]</u>	Número que indica el crecimiento del ítem respecto al resto.
flex-shrink:	1 <u>[factor de decrecimiento]</u>	Número que indica el decrecimiento del ítem respecto al resto.
flex-basis:	<u>[tamaño base]</u> content	Tamaño base de los ítems antes de aplicar variación.
order:	<u>[número]</u>	Número que indica el orden de aparición de los ítems.

UA 2.18: Introducción CSS: Flexbox



Propiedades de ítems hijos: flex-grow

- ❑ La propiedad **flex-grow** indica el factor de crecimiento de los ítems en el caso de que no tengan un ancho específico.
- ❑ Por ejemplo, si con flex-grow indicamos un valor de 1 a todos sus ítems, tendrían el mismo tamaño cada uno de ellos. Pero si colocamos un valor de 1 a todos los elementos, salvo a uno de ellos, que le indicamos 2, ese ítem será más grande que los anteriores.
- ❑ Los ítems a los que no se le especifique ningún valor, tendrán por defecto valor de 0.



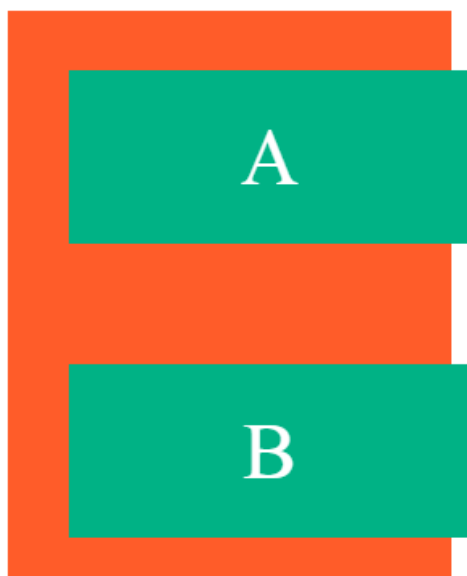
```
.hijo{
  background: #00B285;
  width: 200px;
  /*height: 100px;*/
  color:#fff;
  margin: 30px;
  padding: 20px;
  font-size: 40px;
  text-align: center;
  flex-grow: 1;
  /*Indicamos que ocupe el 100% del
  contenedor*/
}
```

UA 2.18: Introducción CSS: Flexbox

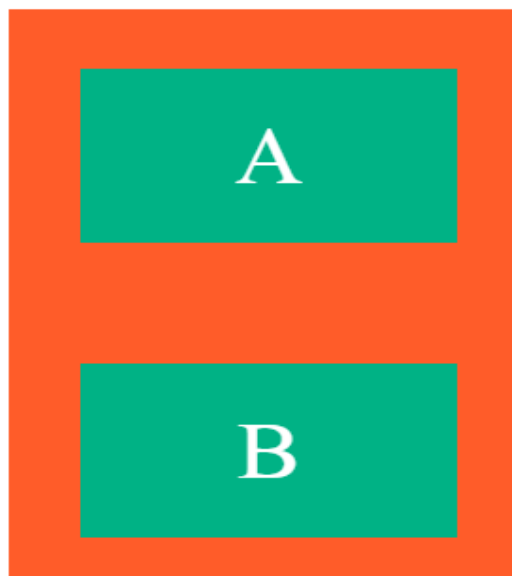


Propiedades de ítems hijos: *flex-shrink*

- ❑ La propiedad *flex-shrink* es complementaria a *flex-grow*.
- ❑ Mientras que la anterior indica un factor de crecimiento, *flex-shrink* hace justo lo contrario: aplica un factor de decrecimiento.
- ❑ De esta forma, los ítems que tengan un valor numérico más grande, serán más pequeños, mientras que los que tengan un valor numérico más pequeño serán más grandes, justo al contrario de como funciona la propiedad *flex-grow*.



```
.hijo{
  background: #00B285;
  width: 200px;
  /*height: 100px;*/
  color:#fff;
  margin: 30px;
  padding: 20px;
  font-size: 40px;
  text-align: center;
  flex-grow: 1;
  flex-shrink: 0;
}
```



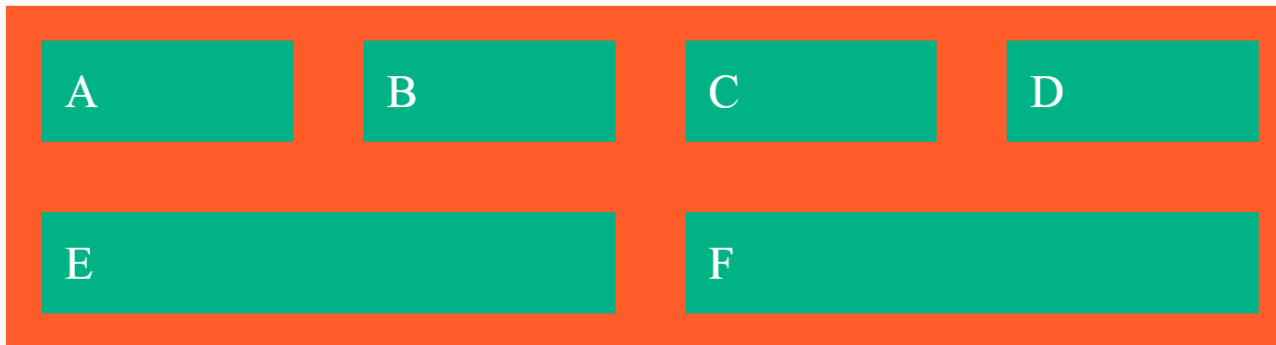
```
.hijo{
  background: #00B285;
  width: 200px;
  /*height: 100px;*/
  color:#fff;
  margin: 30px;
  padding: 20px;
  font-size: 40px;
  text-align: center;
  flex-grow: 1;
  flex-shrink: 1;
}
```

UA 2.18: Introducción CSS: Flexbox



Propiedades de ítems hijos: *flex-basis*

- ❑ Por último, la propiedad ***flex-basis*** define el tamaño por defecto (de base) que tendrán los ítems antes de aplicarle la distribución de espacio.
- ❑ Generalmente, se aplica un tamaño (unidades, porcentajes, etc...), pero también se puede aplicar la palabra clave **content** que ajusta automáticamente el tamaño al contenido del ítem, que es su valor por defecto.



```
.hijo{
  background: #00B285;
  width: 200px;
  /*height: 100px;*/
  color:#fff;
  margin: 30px;
  padding: 20px;
  font-size: 40px;
  flex-grow: 1;
  flex-shrink: 0;
  flex-basis: content;
}
```



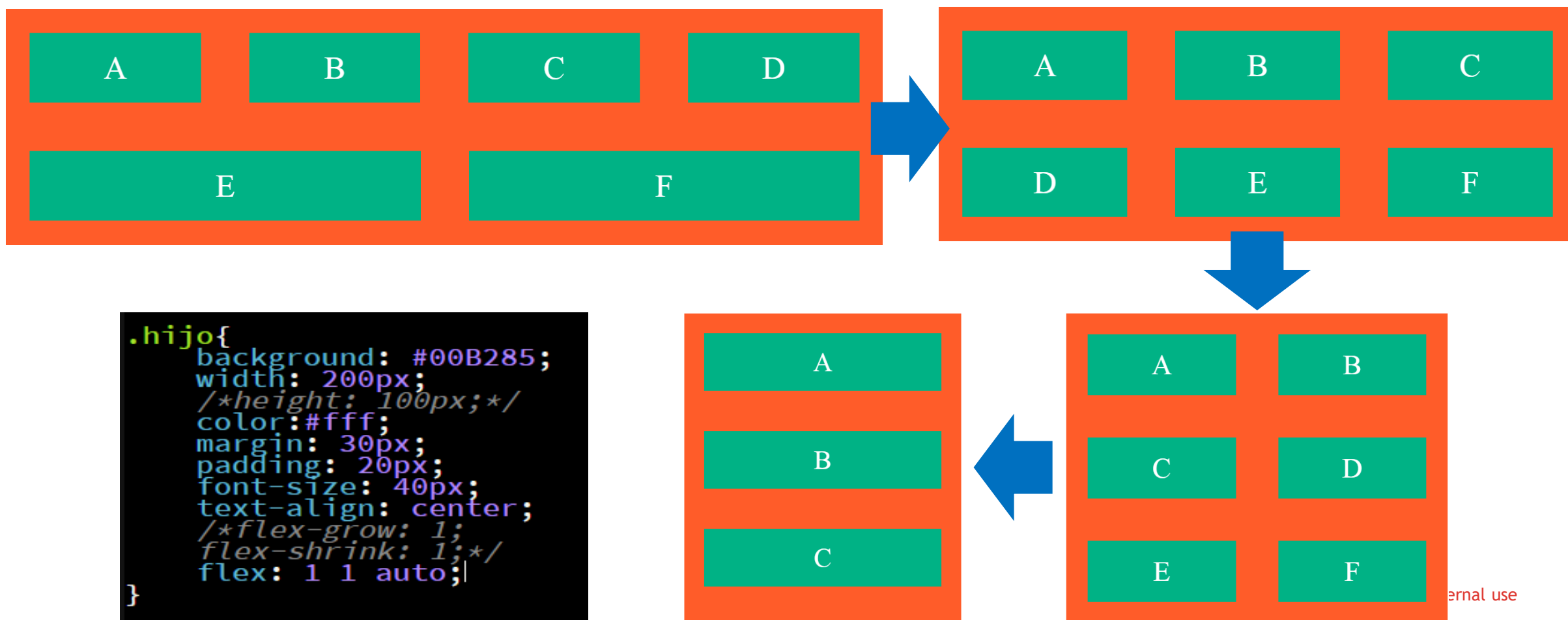
```
.hijo{
  background: #00B285;
  width: 200px;
  /*height: 100px;*/
  color:#fff;
  margin: 30px;
  padding: 20px;
  font-size: 40px;
  flex-grow: 1;
  flex-shrink: 0;
  flex-basis: 50%;
}
```


UA 2.18: Introducción CSS: Flexbox



Propiedades de ítems hijos: flex

- ❑ Con ***flex: 1 1 auto***, conseguimos que todos los elementos crezcan por igual y mantengan un tamaño similar en todo momento.

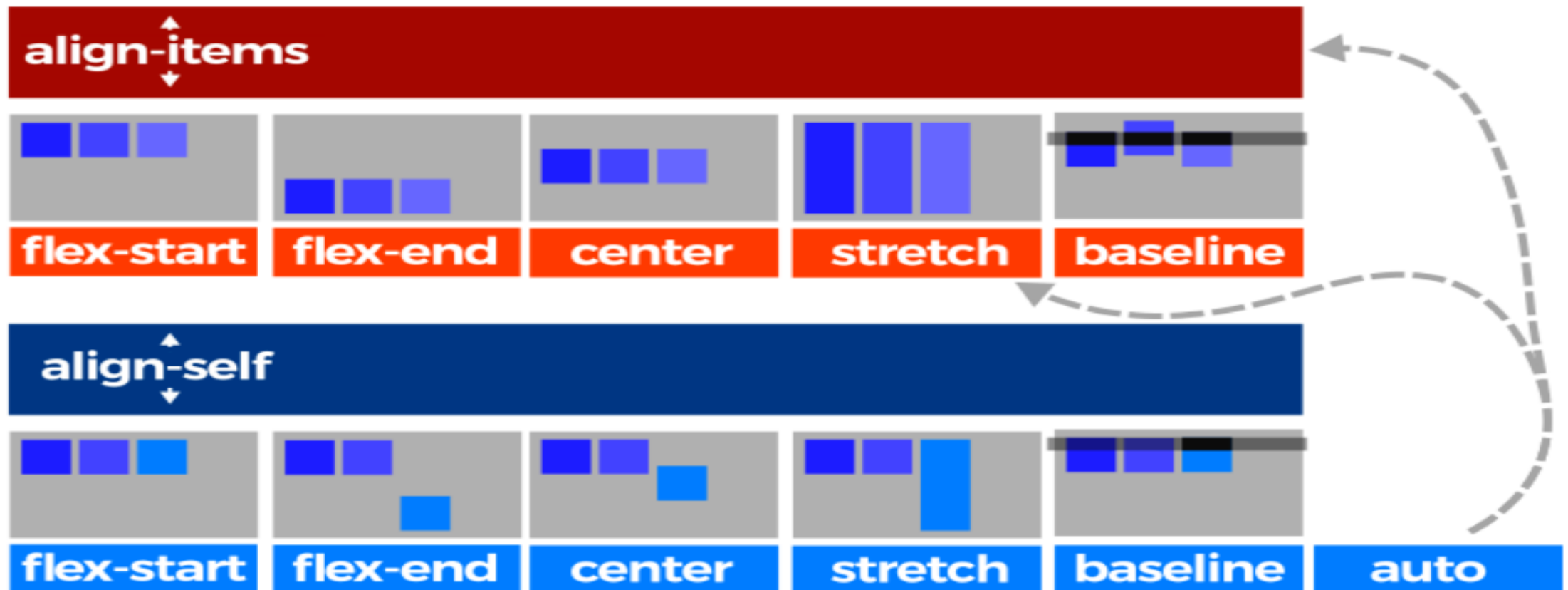


UA 2.18: Introducción CSS: Flexbox



Propiedades de ítems hijos: sobre eje secundario → align-self

- ❑ La propiedad **align-self** actúa exactamente igual que **align-items**, sin embargo esta se utiliza sobre un ítem hijo específico y no sobre el elemento contenedor.
- ❑ Salvo por este detalle, funciona exactamente igual que **align-items**.



UA 2.18: Introducción CSS: Flexbox



Propiedades de ítems hijos: sobre eje secundario → align-self

- ❑ Gracias a ese detalle, **align-self** nos permite cambiar el comportamiento de **align-items** y sobrescribirlo con comportamientos específicos para ítems concretos que no queremos que se comporten igual que el resto.
- ❑ Las propiedades que puede tener son las siguientes:

Valor	Descripción
flex-start	Alinea los ítems al principio del contenedor.
flex-end	Alinea los ítems al final del contenedor.
center	Alinea los ítems al centro del contenedor.
stretch	Alinea los ítems estirándolos al tamaño del contenedor.
baseline	Alinea los ítems en el contenedor según la base de los ítems.
auto	Hereda el valor de align-items del padre (o si no lo tiene, stretch).

- ❑ Si se especifica el valor **auto** a la propiedad **align-self**, el navegador le asigna el valor de la propiedad **align-items** del contenedor padre, y en caso de no existir, el valor por defecto: **stretch**.

UA 2.18: Introducción CSS: Flexbox



Orden de los Items

- ❑ Con **order** establecemos el orden de los ítems según una secuencia numérica.
- ❑ Por defecto, todos los ítems flex tienen un **order: 0** implícito, aunque no se especifique. Si indicamos un **order** con un valor numérico, irá recolocando los ítems según su número, colocando antes los ítems con número más pequeño (incluso valores negativos) y después los ítems con números más altos.
- ❑ De esta forma podemos recolocar fácilmente los ítems incluso utilizando media queries o responsive design.

```
.hijo{
  background: #00B285;
  width: 200px;
  height: 100px;
  color: #fff;
  margin: 30px;
  padding: 20px;
  font-size: 40px;
  text-align: center;
  /*flex-grow: 1;
  flex-shrink: 0;*/
  flex: 1 1 auto;
  order: 2;
}
.otro{
  order: 1;
}
```



UA 2.18: Introducción CSS: Flexbox



Propiedades de ítems hijos: sobre eje secundario → align-self

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <title>Flex</title>
  <meta name="viewport" content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">
  <link rel="stylesheet" href="flexboxhijos.css">
</head>
<body>
  <div class="contenedor">
    <div class="hijo">A</div>
    <div class="hijo">B</div>
    <div class="hijo">C</div>
    <div class="hijo otro">D</div>
  </div>
</body>
</html>

.contenedor{
  width: 80%;
  height: 500px;
  background: #FF5C29;
  margin: 50px auto;

  display: flex;

  flex-direction: row;
  flex-wrap: wrap;
  /*align-content: space-around;*/
  /*Se lo quitamos para poder probar
  el align-self*/
}
.hijo{
  background: #00B285;
  width: 200px;
  height: 100px;
  color: #fff;
  margin: 30px;
  padding: 20px;
  font-size: 40px;
  text-align: center;
  /*flex-grow: 1;
  flex-shrink: 0;*/
  flex: 1 1 auto;
}
.otro{
  align-self: flex-start;
}
```



UA 2.18: Introducción CSS: Flexbox



Propiedades de ítems hijos: sobre eje secundario → align-self

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <title>Flex</title>
  <meta name="viewport" content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">
  <link rel="stylesheet" href="flexboxhijos.css">
</head>
<body>
  <div class="contenedor">
    <div class="hijo">A</div>
    <div class="hijo">B</div>
    <div class="hijo">C</div>
    <div class="hijo otro">D</div>
  </div>
</body>
</html>
```

```
.hijo{
  background: #00B285;
  width: 200px;
  height: 100px;
  color: #fff;
  margin: 30px;
  padding: 20px;
  font-size: 40px;
  text-align: center;
  /*flex-grow: 1;
  flex-shrink: 0;*/
  flex: 1 1 auto;
}
.otro{
  align-self: flex-end;
}
```



UA 2.18: Introducción CSS: Flexbox



Propiedades de ítems hijos: sobre eje secundario → align-self

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <title>Flex</title>
  <meta name="viewport" content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">
  <link rel="stylesheet" href="flexboxhijos.css">
</head>
<body>
  <div class="contenedor">
    <div class="hijo">A</div>
    <div class="hijo">B</div>
    <div class="hijo">C</div>
    <div class="hijo otro">D</div>
  </div>
</body>
</html>
```

```
.hijo{
  background: #00B285;
  width: 200px;
  height: 100px;
  color: #fff;
  margin: 30px;
  padding: 20px;
  font-size: 40px;
  text-align: center;
  /*flex-grow: 1;
  flex-shrink: 0;*/
  flex: 1 1 auto;
}
.otro{
  align-self: center;
}
```



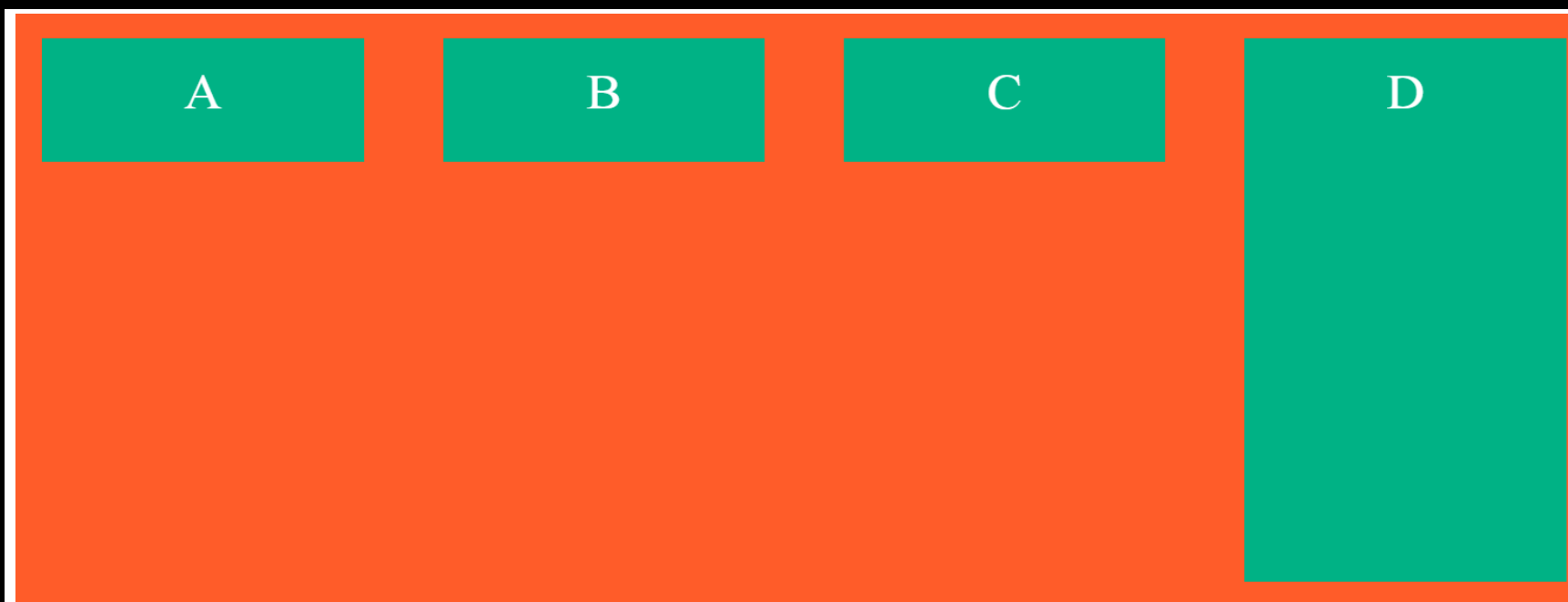
UA 2.18: Introducción CSS: Flexbox



Propiedades de ítems hijos: sobre eje secundario → align-self

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <title>Flex</title>
  <meta name="viewport" content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">
  <link rel="stylesheet" href="flexboxhijos.css">
</head>
<body>
  <div class="contenedor">
    <div class="hijo">A</div>
    <div class="hijo">B</div>
    <div class="hijo">C</div>
    <div class="hijo otro">D</div>
  </div>
</body>
</html>
```

```
.hijo{
  background: #00B285;
  width: 200px;
  height: 100px;
  color: #fff;
  margin: 30px;
  padding: 20px;
  font-size: 40px;
  text-align: center;
  /*flex-grow: 1;
  flex-shrink: 0;*/
  flex: 1 1 auto;
}
.otro{
  align-self: stretch;
  height: auto;
}
```





**Universidad
Europea**

LAUREATE INTERNATIONAL UNIVERSITIES

Madrid

Valencia

Canarias