

[Twitter](#) [Facebook](#) [RSS Feed](#)

- [Skip to navigation](#)
- [Skip to main content](#)
- [Skip to secondary content](#)
- [Skip to footer](#)

Programando a medianoche

El blog de Scientia® Soluciones Informáticas

- [Gustavo Cantero \(The Wolf\)](#)
- [Dario Krapp](#)

Guía completa para aprender a utilizar CSS Grid Layout



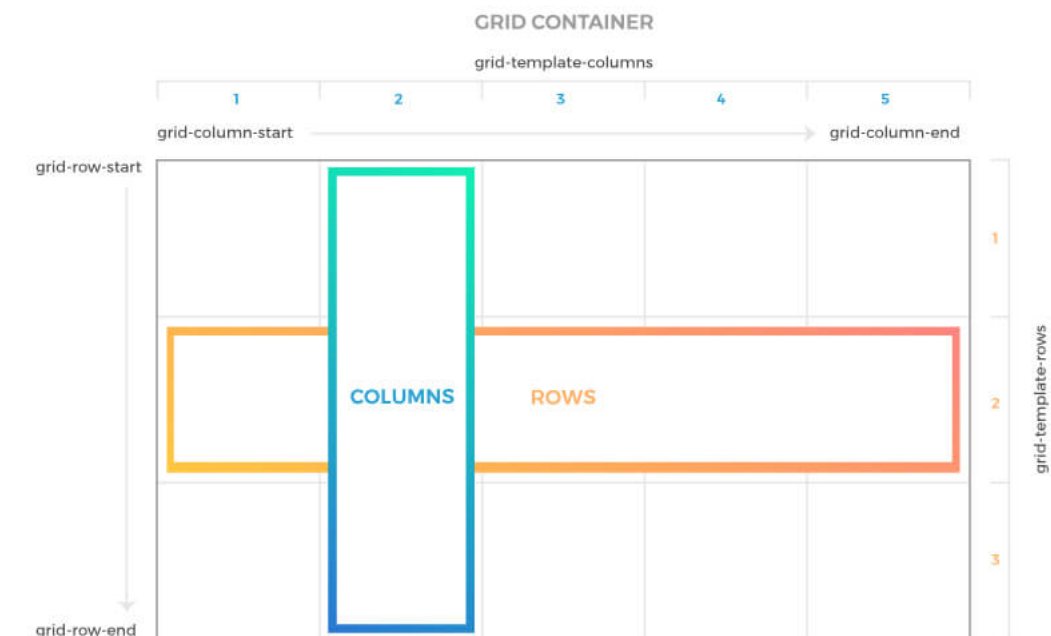
[jueves, 02 de mayo de 2019 a las 14:53hs](#) por Valentina Echezuria



Empezando a utilizar Grid

A diferencia de Flexbox que sirve para posicionamientos en una sola dimensión, el posicionamiento en Grid permite la disposición de elementos en 2 dimensiones, logrando valores o espacios que ocuparan los elementos dentro de una composición.





Contenedor con grid

Para activar la cuadrícula grid hay que determinar sobre el elemento contenedor la propiedad display y especificar el valor grid o inline-grid. Este valor determina como cuadrícula con el contenido exterior. El primero permite que la cuadrícula aparezca encima/debajo del contenido exterior (en bloque) y el segundo de ellos permite que muestre de izquierda/derecha (en línea) del contenido exterior.

```

1 <div class="contenedor">
2   <div class="item">ITEM 1</div>
3   <div class="item">ITEM 2</div>
4   <div class="item">ITEM 3</div>
5   <div class="item">ITEM 4</div>
6   <div class="item">ITEM 5</div>
7   <div class="item">ITEM 6</div>
8 </div>

1 .contenedor{
2   display: grid;
3 }
4
5 .item{
6   background-color: #b4b8b9;
7   border: 1px solid #ffffff;
8   padding: 1rem;
9   font-family: arial;
10  color: white;
11  text-align: center;
12 }

```

Resultado:

ITEM 1
ITEM 2
ITEM 3
ITEM 4
ITEM 5
ITEM 6

Establecer los valores

Podemos observar que la disposición de los elementos no ha cambiado porque no hemos definido el numero de columnas que deberá ocupar nuestra grilla, esto lo pode utilizando la propiedad grid-template-columns en distintas maneras:

Utilizando la unidad de medida fr, que permite establecer la cantidad de espacio que ocupara un elemento dentro del área disponible , es decir , 1 fr es igual a una parte

1fr	1fr	1fr
-----	-----	-----

```
1 grid-template-columns: 2fr 1fr 1fr;
```

2fr	1fr	1fr
-----	-----	-----

En este caso la primera columna ocupara 2 veces mas que el resto de las columnas.
Esta medida puede ser utilizada en unión con unidades de medidas fijas como px:

```
1 grid-template-columns: 200px 2fr 1fr;
```

200px	2fr	1fr
-------	-----	-----

Las medidas de la primera columna serán absolutas mientras que las que estén establecidas en fr serán flexibles.

```
1 .contenedor{
2   display: grid;
3   grid-template-columns: 200px 2fr 1fr;
4   height: 500px;
5 }
6
7 .item{
8   background-color: #b4b8b9;
9   border: 1px solid #ffffff;
10  padding: 1rem;
11  font-family: arial;
12  color: white;
13  text-align: center;
14 }
```

Resultado:

ITEM 1	ITEM 2	ITEM 3
ITEM 4	ITEM 5	ITEM 6

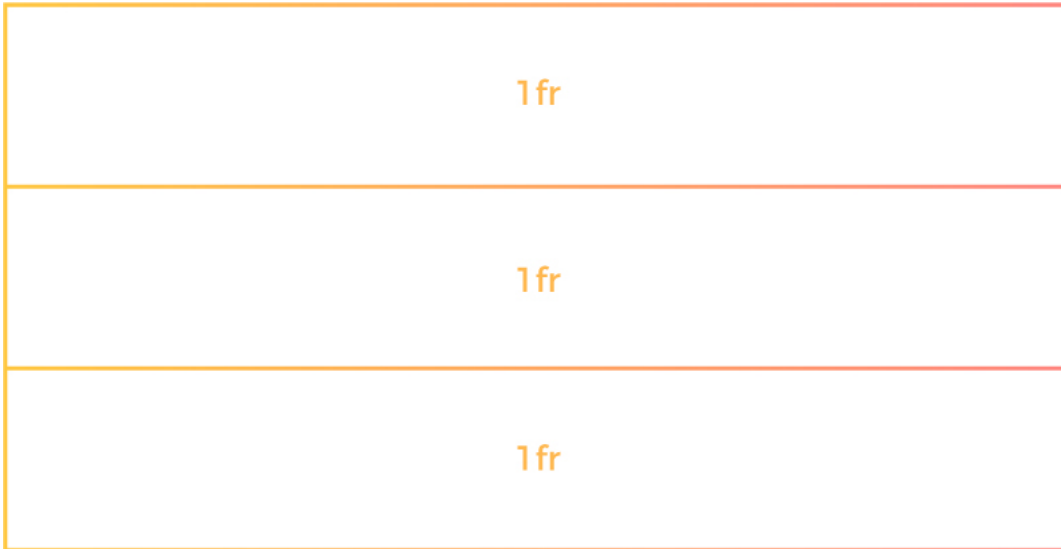
Vista de la grilla:

ITEM 1	ITEM 2	ITEM 3
ITEM 4	ITEM 5	ITEM 6

Definiendo el numero de filas que tendrá nuestra grilla:

Determinaremos el numero de filas utilizando la propiedad grid-template-rows que puede ser utilizado igual que grid-template-columns, definiendolo en fr o medidas c o vw.

```
1 grid-template-rows: 1fr 1fr 1fr;
```



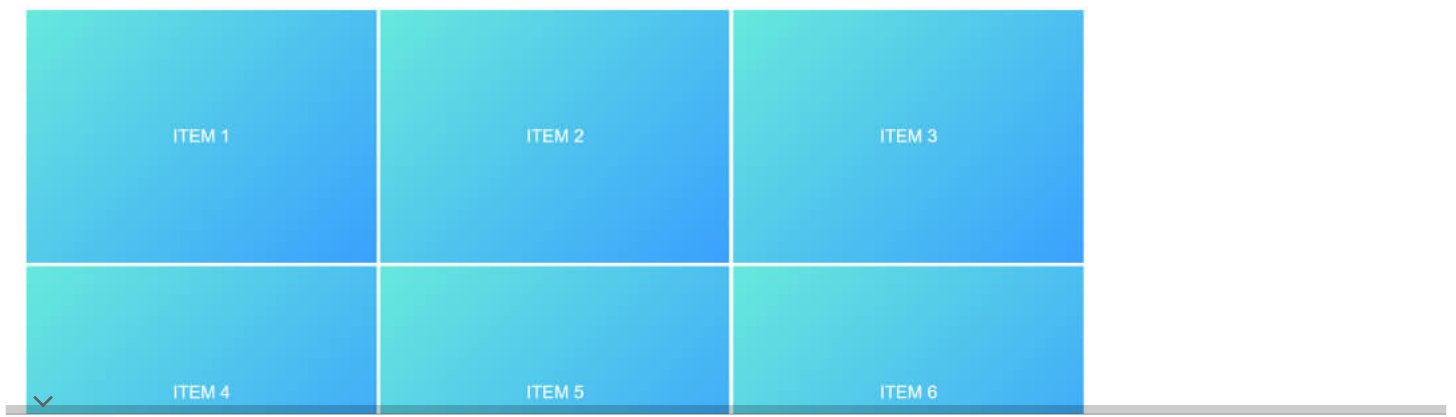
```
1 grid-template-rows: 1fr 2fr 1fr;
```

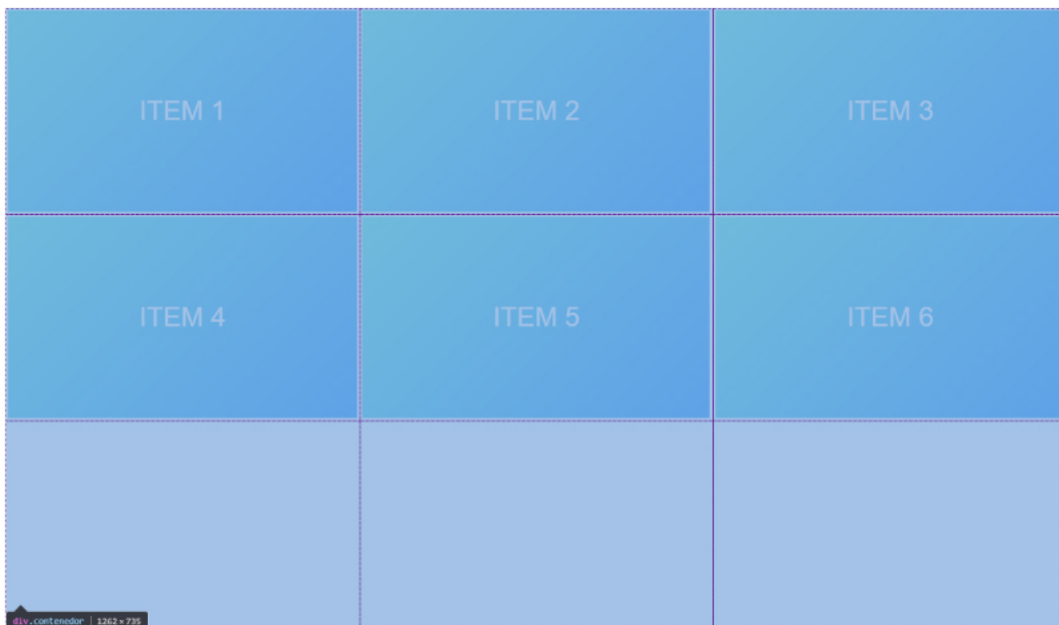
Ejemplo:

```
1 <div class="item">ITEM 1</div>
2 <div class="item">ITEM 2</div>
3 <div class="item">ITEM 3</div>
4 <div class="item">ITEM 4</div>
5 <div class="item">ITEM 5</div>
6 <div class="item">ITEM 6</div>
```

```
1 .contenedor{
2   display: grid;
3   height: 100vh; /*ocupa el alto de la pantalla*/
4   grid-template-columns: repeat(3, 1fr); /*los elementos ocuparan 3 columnas*/
5   grid-template-rows: repeat(2, 1fr); /*los elementos ocuparan 2 filas*/
6 }
7
8 .item{
9   border: 3px solid #ffffff;
10  padding: 1rem;
11  font-family: arial;
12  color: white;
13  text-align: center;
14  background: linear-gradient(135deg, rgb(97, 234, 222) 0%, rgb(30, 168, 255) 100%);
15  display: flex;
16  justify-content: center;
17  align-items: center;
18  font-size: 2rem;
19 }
```

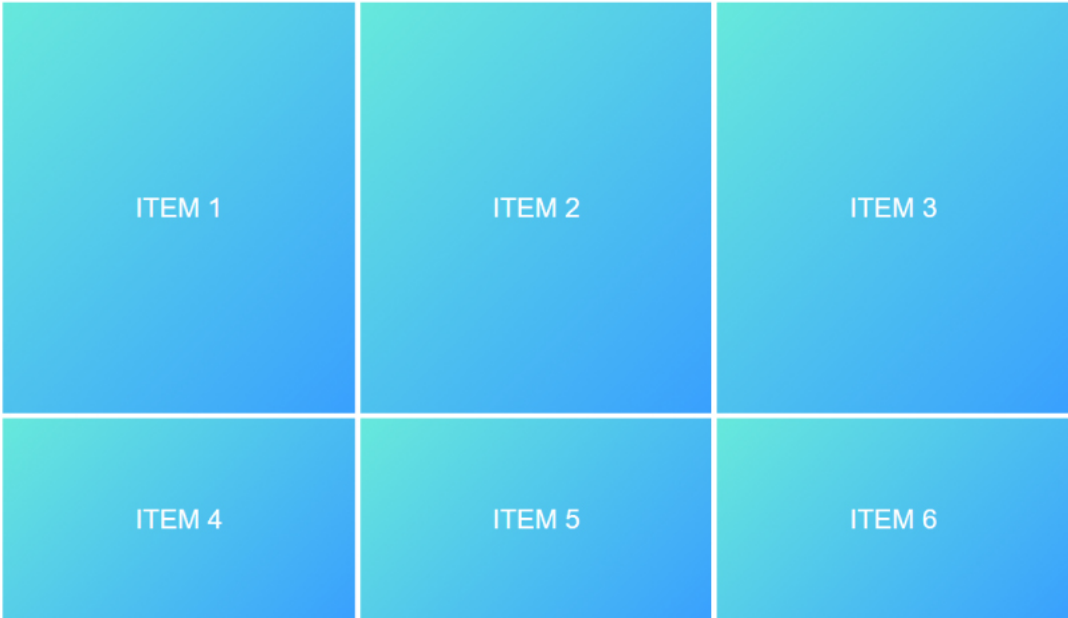
Resultado:





También podremos establecer que algunas filas ocupen mayor espacio que otras

```
1 | grid-template-rows: repeat(1, 2fr) repeat(1, 1fr);
```



A parte de poder establecer el numero de columnas y filas y del espacio que ocuparan cada una de ellas, también podremos establecer estos valores en cada celda indivi superponiendo celdas con otras, esto lo podemos lograr con las propiedades:

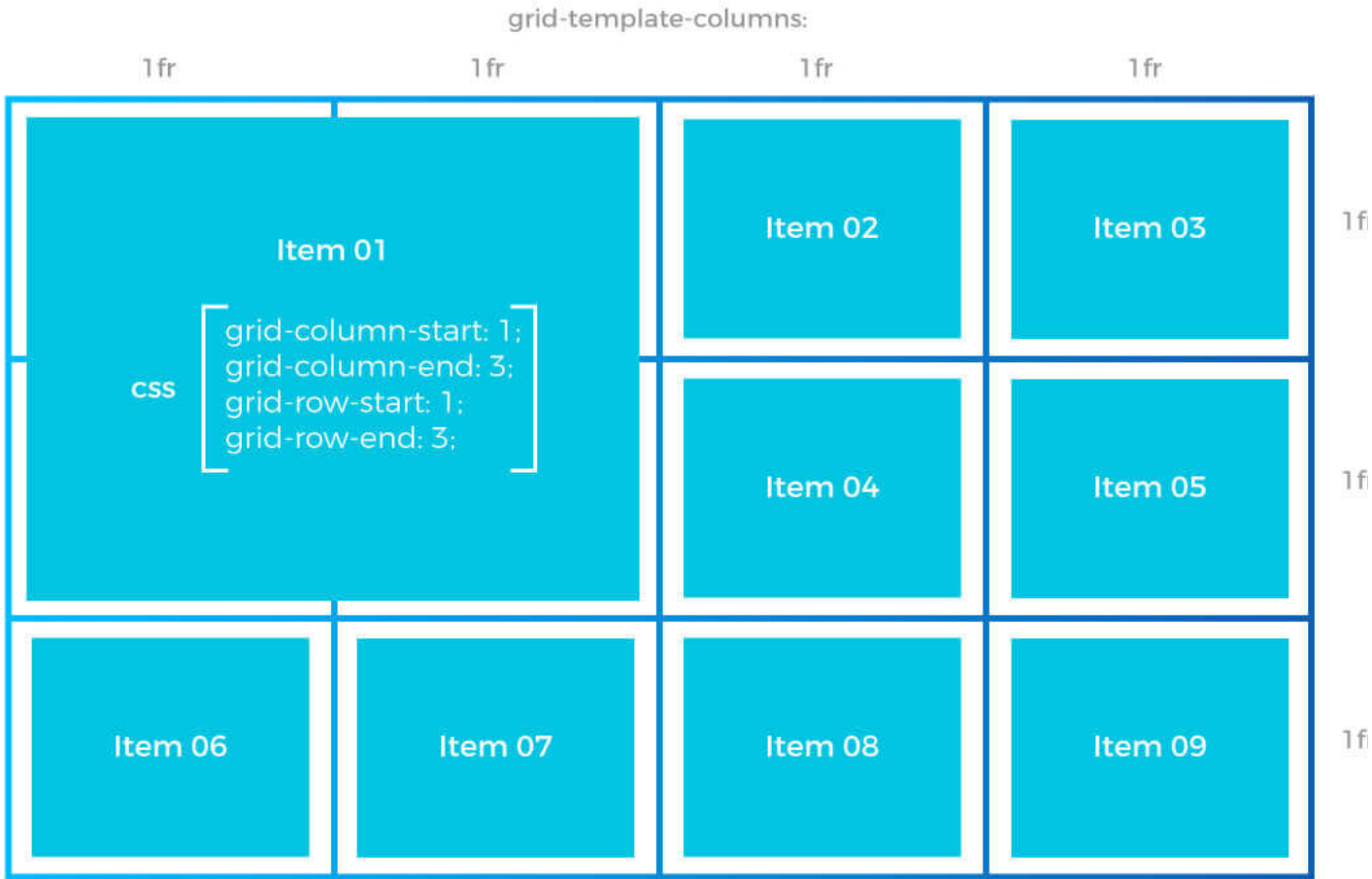
- grid-column-start: el lugar de inicio entre las columnas que ocupara esta celda o item.
- grid-column-end el lugar final entre las columnas que ocupara esta celda o item.
- grid-row-start: el lugar de inicio entre las filas que ocupara esta celda o item.
- grid-row-end el lugar final entre las filas que ocupara esta celda o item.

```
3      <div class="item two">ITEM 2</div>
4      <div class="item">ITEM 3</div>
5      <div class="item">ITEM 4</div>
6      <div class="item">ITEM 5</div>
7      <div class="item">ITEM 6</div>
8      <div class="item">ITEM 7</div>
9      <div class="item">ITEM 8</div>
10     <div class="item">ITEM 9</div>
11 </div>
```

```

1 .contenedor{
2     display: grid;
3     grid-template-columns: 1fr 1fr 1fr 1fr;
4     grid-template-rows: 1fr 1fr 1fr;
5 }
6 .one {
7     grid-column-start: 1;
8     grid-column-end: 3;
9     grid-row-start: 1;
10    grid-row-end: 3;
11 }

```



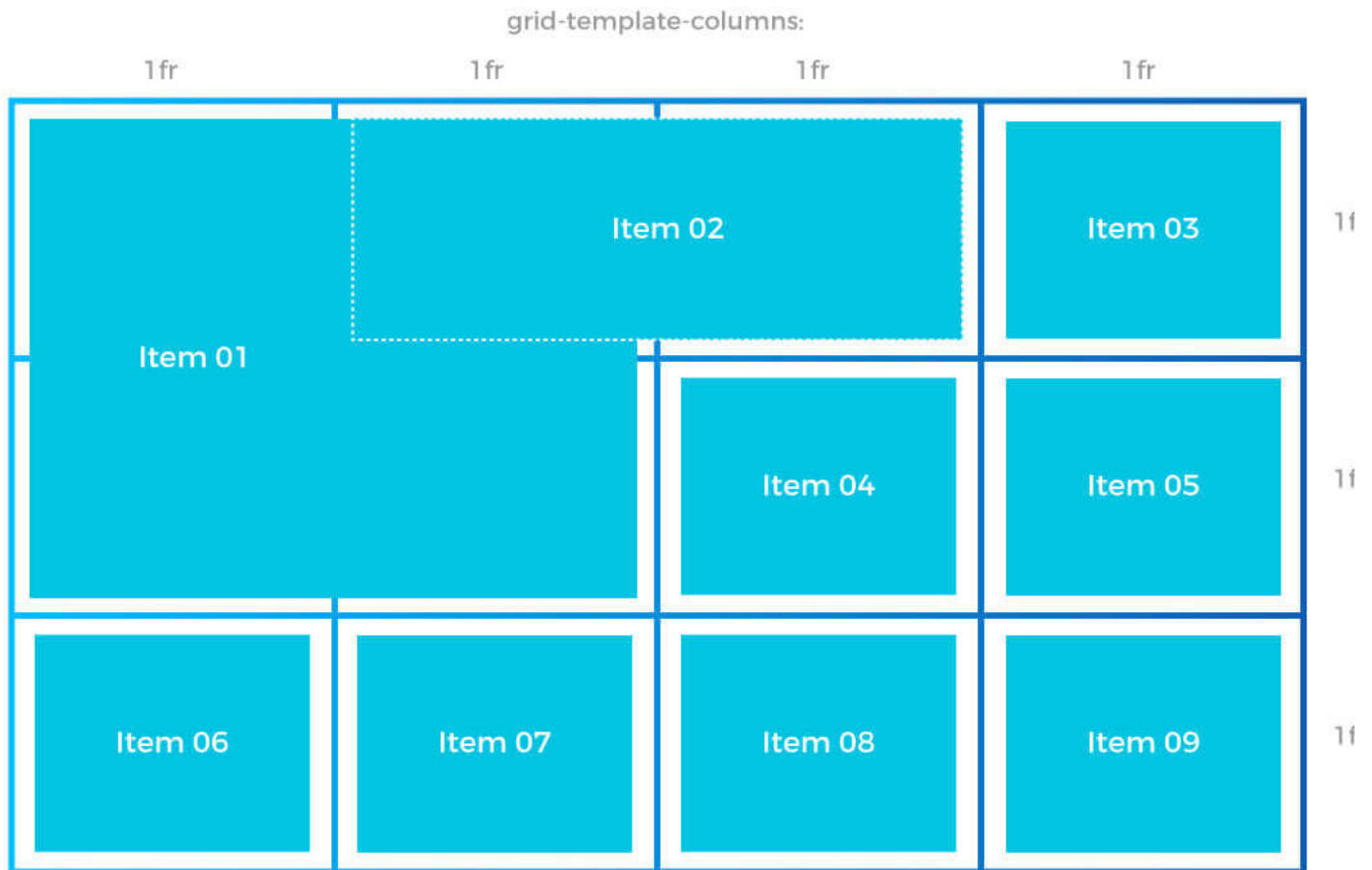
Superposición de celdas:

Con este método también podemos superponer celdas determinando la relevancia de cada uno mediante la propiedad z-index:

```
1 .one {
2     grid-column-start: 1;
3     grid-column-end: 3;
4     grid-row-start: 1;
5     grid-row-end: 3;
6 }
7 .two {
8     grid-column-start: 2;
9     grid-column-end: 4;
10    grid-row-start: 1;
11    grid-row-end: 2;
12 }
```

que sería lo mismo que:

Resultado:

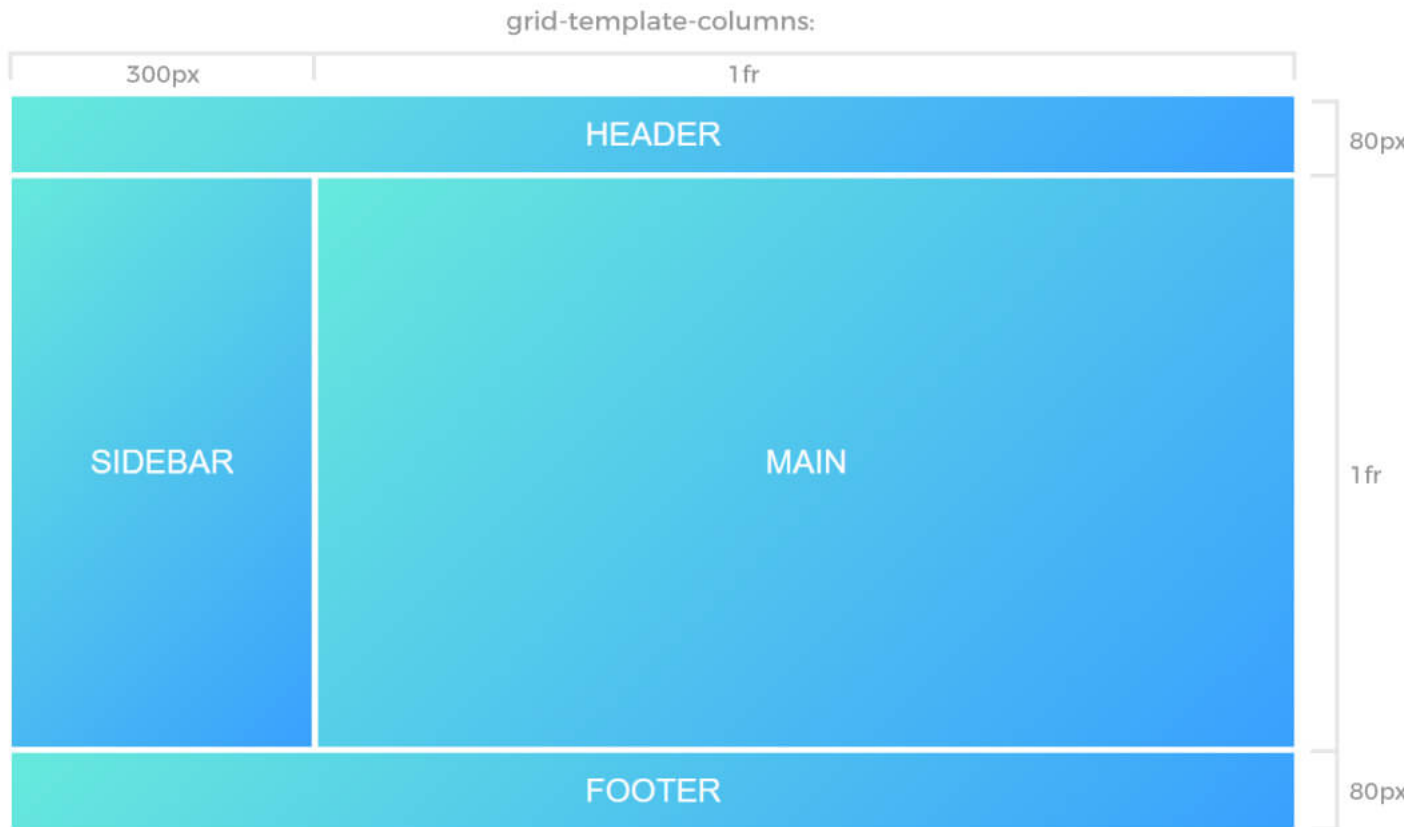


Denominando áreas de grilla

La propiedad `grid-template-areas` permite ponerle nombres a cada una de las celdas, determinado el espacio que ocuparan y la ubicación de estas. Podemos emplearlas de la siguiente manera:

```
1 <div class="contenedor">
2   <div class="header">HEADER</div>
3   <div class="sidebar">SIDEBAR</div>
4   <div class="main">MAIN</div>
5   <div class="footer">FOOTER</div>
6 </div>
```

```
1 .contenedor{
2   display: grid;
3   height: 98vh;
4   grid-template-areas: "header header"
5                       "sidebar main"
6                       "footer footer";
7   grid-template-rows: 80px 1fr 80px;
8   grid-template-columns: 300px 1fr;
9 }
10
11 .header{
12   grid-area: header;
13 }
14
15 .sidebar{
16   grid-area: sidebar;
17 }
18
19 .main{
20   grid-area: main;
21 }
22
23 .footer{
24   grid-area: footer;
25 }
```



Repetir columnas y filas utilizando la propiedad repeat()

La propiedad `repeat()` representa un fragmento repetido de la lista de pistas, lo que permite escribir un gran número de columnas o filas que exhiben un patrón recurrente de manera más compacta.

Por ejemplo:

```
1 grid-template-columns: 1fr 1fr 1fr;
```

sería igual a:

```
1 grid-template-columns: repeat(3, 1fr);
```

y:

```
1 grid-template-columns: 2fr 2fr 2fr 1fr 1fr 1fr;
```

sería igual a:

```
1 grid-template-columns: repeat(3, 2fr) 1fr;
```

Propiedad minmax

Podemos establecer mediante esta propiedad el tamaño mínimo y máximo de las filas y columnas, previendo los cambios que tendrá el objeto al reducirse o ampliarse en la pantalla.

Por ejemplo

```
1 .contenedor{
2   display: grid;
3   grid-template-columns:minmax(100px, 200px) 1fr 1fr;
4 }
```

También podemos establecer valores en otras unidades de medidas flexibles:

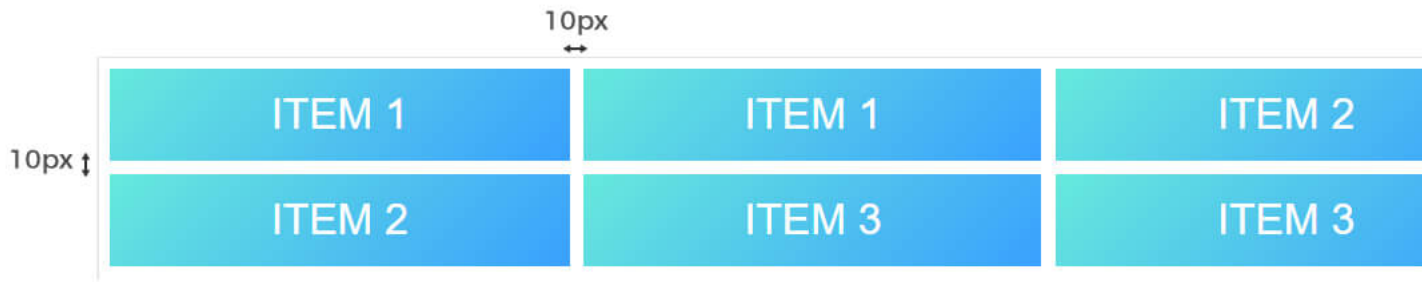
```
1 grid-template-columns:minmax(20%, 40%) 1fr 1fr;
```



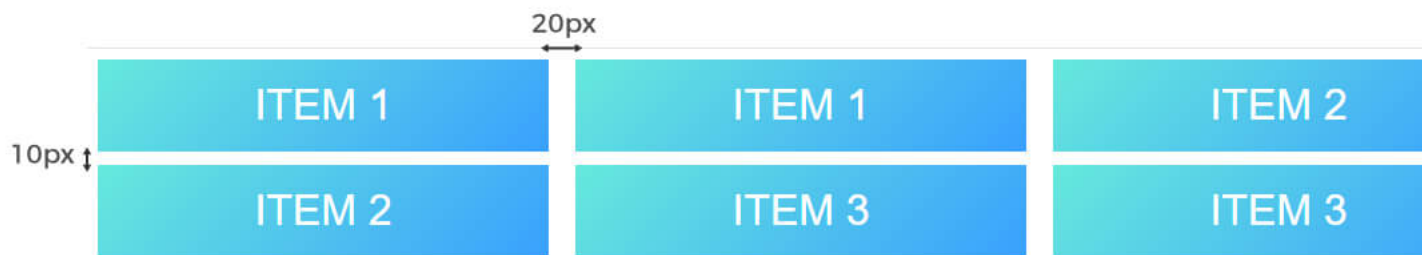
Con esta propiedad definiremos el espaciado que habrá entre las filas y columnas, pudiendo ser especificadas en pixeles o porcentaje.

Si se quiere que las medidas asignadas apliquen tanto a filas como columnas se puede usar la propiedad grid-gap:

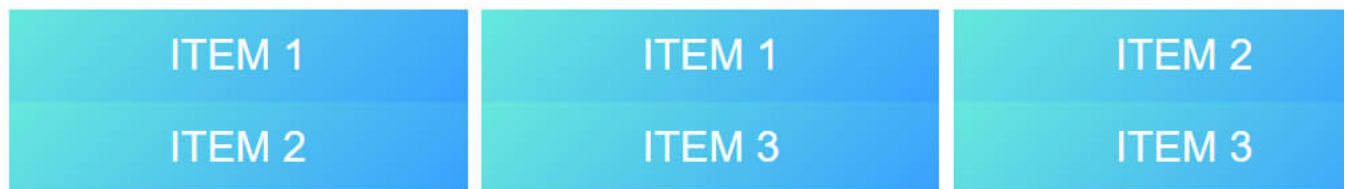
```
1 .contenedor {
2   display: grid;
3   grid-template-columns: 1fr 1fr 1fr;
4   grid-gap: 10px;
5 }
```



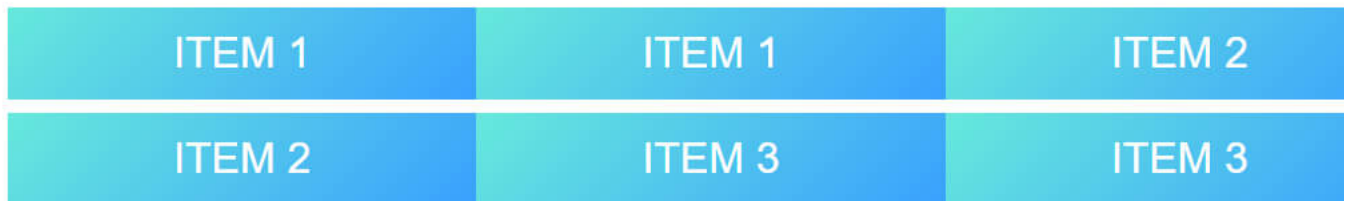
└─ Espaciado entre filas
grid-gap: 10px 20px;
└─ Espaciado entre columnas



grid-column-gap: 10px;



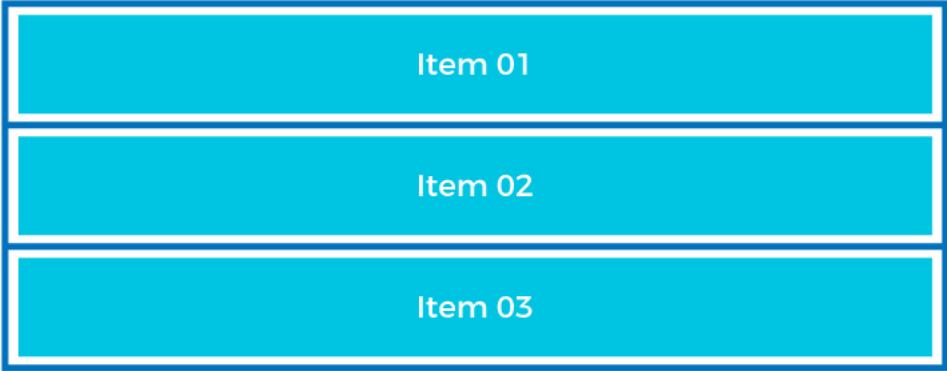
grid-row-gap: 10px;



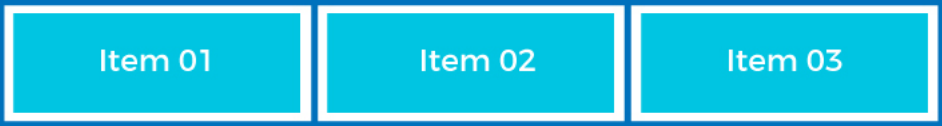
grid-auto-flow: row / column

Tal como la propiedad que tiene flex de dirigir la disposición que tendrán los items (ya sea en filas o en columnas), grid tiene la propiedad grid-auto-flow:

```
1 grid-auto-flow: row
```



```
1 | grid-auto-flow: column
```



Justificado de columnas con justify-content

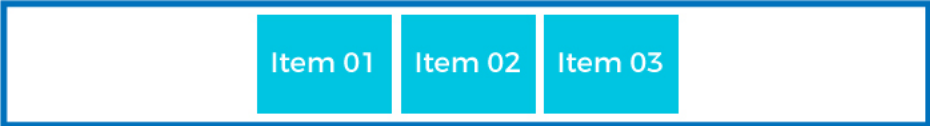
justify-content: left;



justify-content: right



justify-content: center



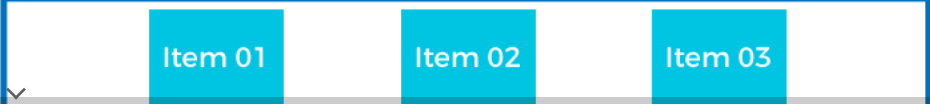
justify-content: space-between

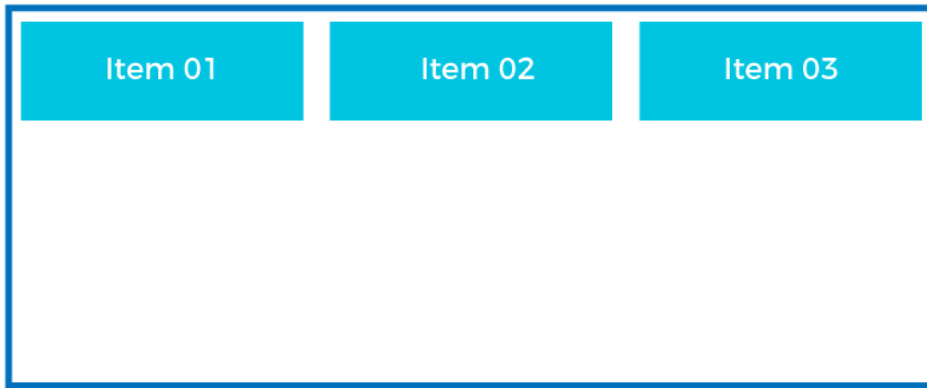


justify-content: space-around

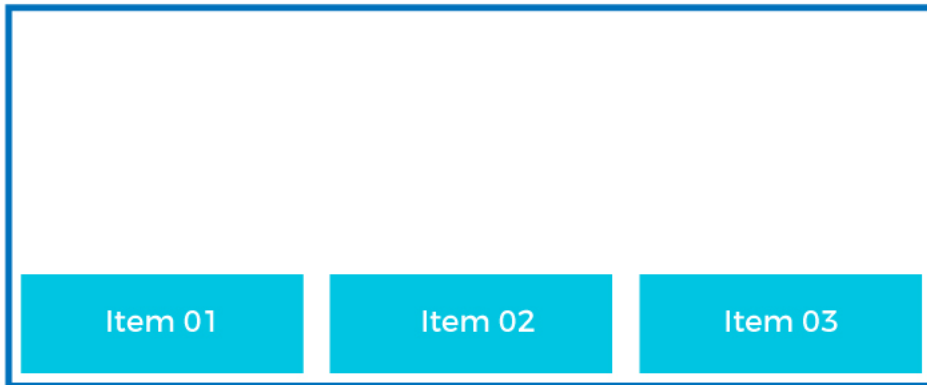


justify-content: space-evenly

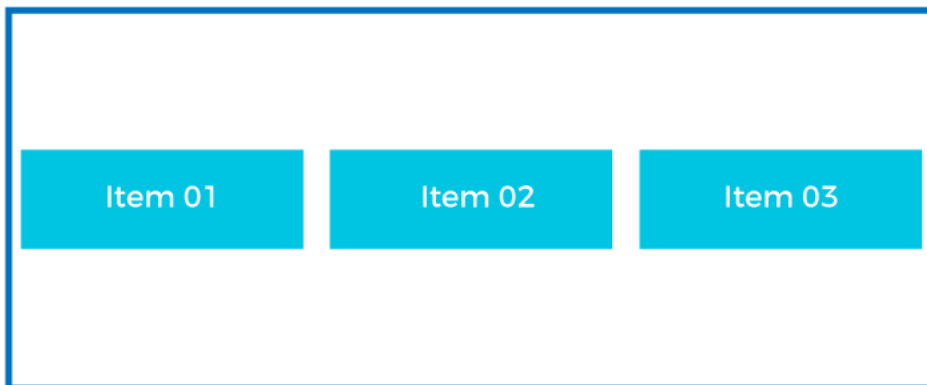




align-items: end



align-items: center



align-items: stretch



Soporte de grid en navegadores

En el siguiente [link](#) se puede consultar el soporte actual que tiene el grid layout en los distintos navegadores.

IE	Edge	Firefox	Chrome	Safari	Opera	iOS Safari	Opera Mini	Android Browser	Blackberry Browser	Opera Mobile	Chrome for Android	Firefox for Android	IE Mobile	UC Browser for Android	Samsung Internet
		2-39	4-28												
		40-51	29-56		10-27										
6-9	12-15	52-53	57	3.1-10	28-43	3.2-10.2									4-5.4
10	16-17	54-65	58-72	10.1-11	44-57	10.3-11		2.1-4.4	7	12-12.1			10		6.2-7.4
11	18	66	73	12	58	12.1	all	67	10	46	71	64	11	11.8	8.2
		67-68	74-76	12.1-TP		12.2									

Categoría [CSS3](#) | Etiquetas: [Error 405 en WebAPI al hacer PUT y DELETE](#)
[Color Pantone del año 2019 Living Coral 16-1546](#)


4 comentarios »

-  [Mike Salinas](#) dice:
[27 abril 2022 a las 14:37](#)

aqui fue donde aprendi a utilizar grid

Cargando...


[Responder](#)

-  [Gustavo Cantero \(The Wolf\)](#) dice:
[27 abril 2022 a las 14:39](#)

Hola Mike.
Me alegra mucho que el artículo de Valentina te haya servido.
Saludos!

Cargando...


[Responder](#)

-  [Lacqua09](#) dice:
[15 agosto 2022 a las 04:54](#)

Es claro el artículo. He leído un par,más detallados, interactivos y más extensos, pero en ESTE encontré CLARIDAD.
Lo que no se comprende, eso sí, es la diferencia entre inline-grid y grid. Una gráfica favorecería la comprensión.
Súper agradecida de haberlos leído.

Cargando...

[Responder](#)

-  [Josué SV](#) dice:
[20 mayo 2023 a las 21:35](#)

Muy buen contenido, esta wenardo, es muy directo al grano.

Cargando...

[Responder](#)

Deja un comentario

Este sitio usa Akismet para reducir el spam. [Aprende cómo se procesan los datos de tus comentarios](#).



Programando a medianoche de [Scientia@ Soluciones Informáticas](#) está licenciada bajo una [Licencia Creative Commons Atribución 2.5 Argentina](#).

