Cómo funciona CSS grid

DigitalHouse >
Coding School

Certified Tech Developer
The Ultimate Degree

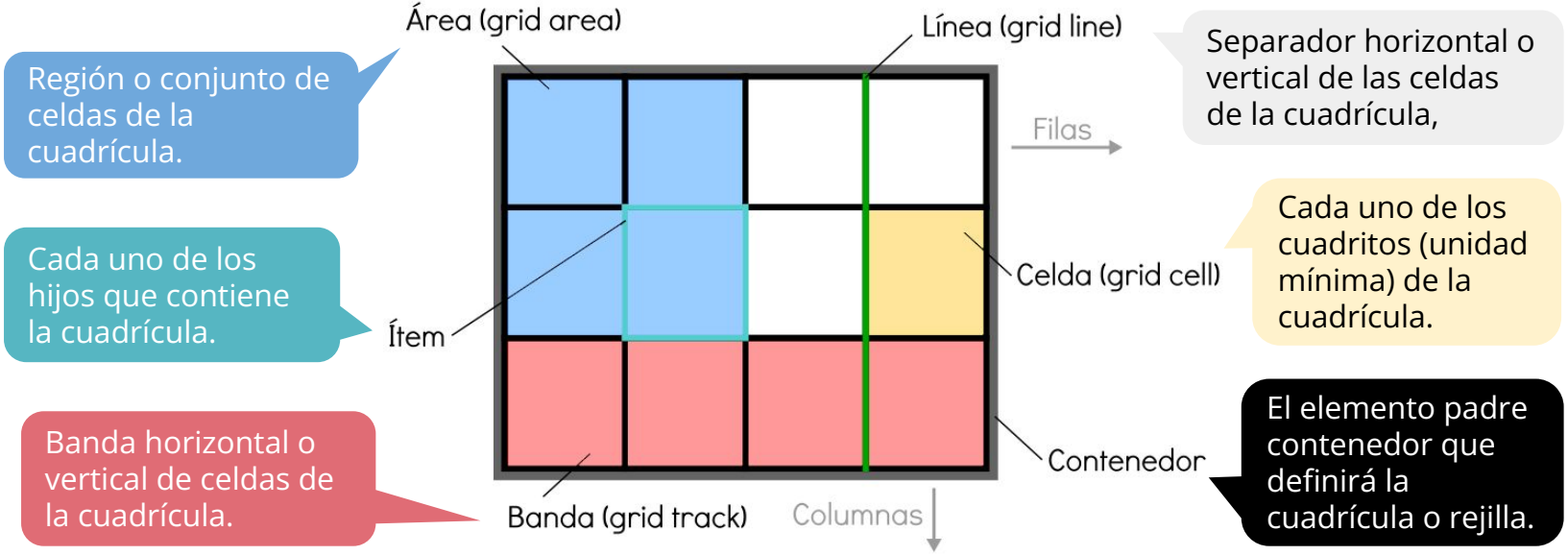
Índice

1 ¿Qué es CSS grid?

CSS grid es un modelo de maquetado basado en un sistema de rejillas incorporado en CSS3 para gestionar de manera sencilla y customizable la disposición de nuestros elementos en nuestra página web.

Elementos de CSS grid

Para utilizar **CSS grid** necesitaremos tener en cuenta una serie de conceptos que usaremos a partir de ahora y que definiremos a continuación:



2 | ¿Cómo funciona?

Activar grid con display: grid | inline-grid

Para activar la cuadrícula grid hay que utilizar sobre el elemento contenedor la propiedad **display** y especificar el valor `grid` o `inline-grid`.

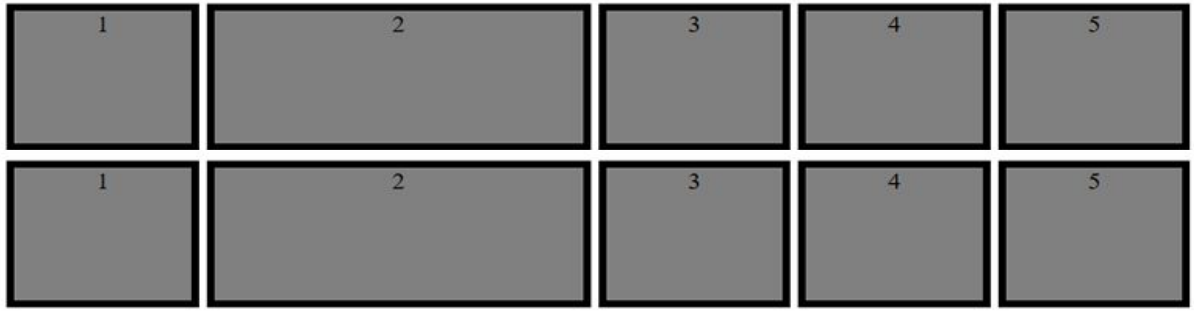
```css
.grid { display: grid; }
```

Este valor influye en cómo se comportará la cuadrícula con el contenido exterior.

- **inline-grid:** Establece una cuadrícula con ítems en línea, de forma equivalente a inline-block.
- **grid:** Establece una cuadrícula con ítems en bloque, de forma equivalente a block.

Armar la rejilla con columnas, filas y espacios

Es posible crear cuadrículas con un tamaño explícito usando las propiedades CSS `grid-template-columns` y `grid-template-rows`, que sirven para indicar las dimensiones de cada celda de la cuadrícula.



Con el siguiente código tendremos una cuadrícula como la anterior:

grid-template-rows

Establece el tamaño de cada fila.

grid-template-columns

Establece el tamaño de cada columna.

```css
.grid {
  display: grid;
  grid-template-columns:  1fr 2fr 1fr 1fr 1fr;
  grid-template-rows:100px 100px;
  grid-gap: 5px;
}
```

Trabajar con **tamaños relativos**, en lugar de especificaciones estáticas, permite que la cuadrícula pueda adaptarse automáticamente al tamaño de la pantalla. La segunda columna (en nuestro ejemplo) siempre será dos veces más grande que las otras. **1fr** es una medida sobre el espacio disponible (free space).

grid-gap

Crea un espacio vacío.

Otras maneras de escribir lo mismo

En algunos casos, en las propiedades `grid-template-columns` y `grid-template-rows` podemos necesitar indicar las mismas cantidades un número alto de veces.

```css
grid-template-columns: 100px 50px 50px 200px;
grid-template-rows: 50px 100px 50px 100px;
```

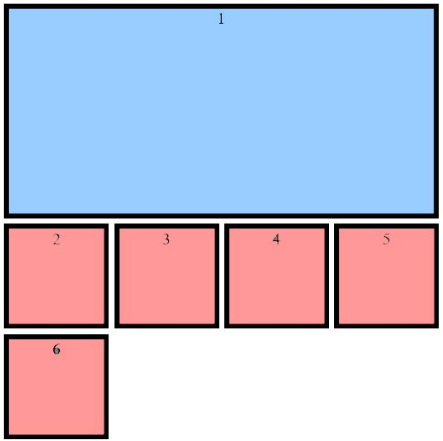
Atajo repeat()

La expresión resultante sería: **repeat([núm de veces], [valor o valores])**

```css
grid-template-columns: 100px repeat(2, 50px) 200px;
grid-template-rows: repeat(2, 50px 100px);
```

Ubicar los elementos

Después de definir la rejilla, se colocan en ella los elementos.



Para esto, debemos crear los **elementos** y especificar los **valores** de inicio y fin. No obstante, no todos los elementos tienen que ocupar necesariamente una sola celda dentro de la cuadrícula.

HTML

```html
<div class="grid-container">
  <div class="grid-item1">1</div>
  <div class="grid-item2">2</div>
  <div class="grid-item2">3</div>
  <div class="grid-item2">4</div>
  <div class="grid-item2">5</div>
  <div class="grid-item2">6</div>
</div>
```

CSS

```css
.grid-container {
  display: grid;
  grid-template-rows: 100px 100px 100px 100px;
  grid-template-columns: 100px 100px 100px 100px;
  grid-gap: 5px;
}
.grid-item1 {  background: #99CCFF;
  text-align: center; border: black 5px solid;
  grid-column-start: 1;
  grid-column-end: 5;
  grid-row-start: 1;
  grid-row-end: 3;
}
.grid-item2 { background: #FF9999; text-align:
center; border: black 5px solid; }
```

Otras maneras de escribir lo mismo

Podemos simplificar las siguiente líneas de código a formas resumidas de las siguiente maneras:.

```css
grid-column-start: 1;
grid-column-end: 5;
grid-row-start: 1;
grid-row-end: 3;
```

Atajos `grid-column` y `grid-row`

Indicamos las línea [comienzo] / [fin].

```css
grid-column: 1 / 5;
grid-row: 1 / 3;
```

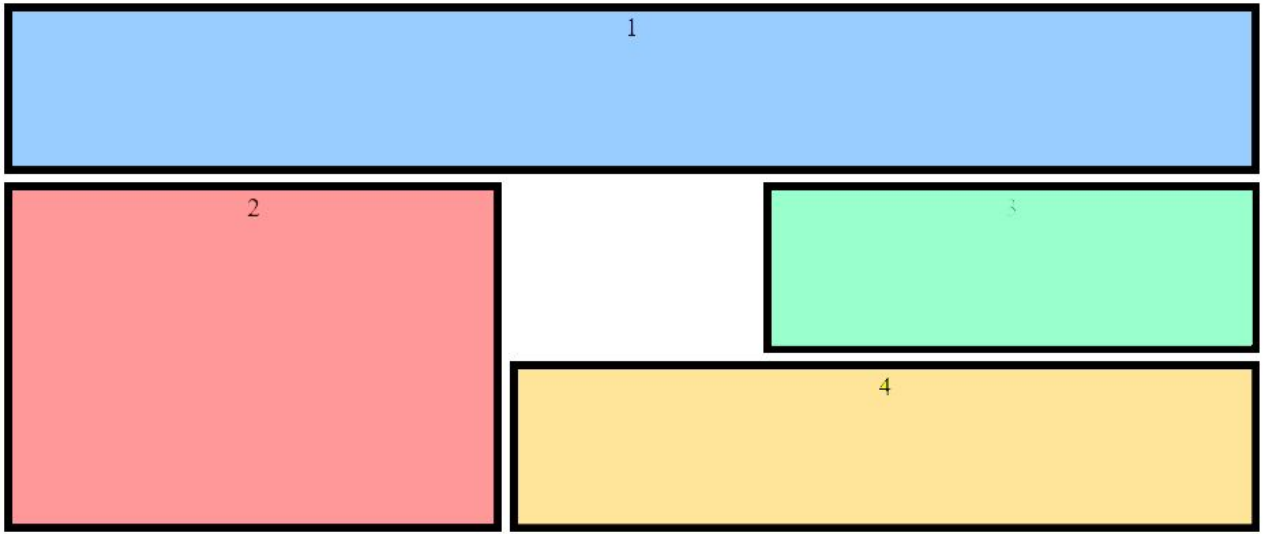
Atajo `span`

Indicamos cuánto se expande desde la posición actual.

```css
grid-column: span 4;
grid-row: span 2;
```

Asignar áreas

CSS grid layout permite combinar celdas en áreas y nombrarlas. Esto facilita la tarea de dividir los elementos en la rejilla. Los ajustes para esto se hacen en el contenedor con `grid-template-areas`.



En este ejemplo, usamos el comando `grid-template-areas` y escribimos los nombres de las áreas deseadas en las celdas línea por línea.

Si no queremos asignar una celda y dejarla en blanco, insertamos un punto . en esta ubicación. Cada fila queda entrecomillada.

```css
.grid-container {
  display: grid;
  grid-template-rows: 100px 100px 100px;
  grid-template-columns: 1fr 1fr 1fr 1fr 1fr;
  grid-gap: 5px;
  grid-template-areas:
  "area1 area1 area1 area1 area1"
  "area2 area2 . area3 area3"
  "area2 area2 area4 area4 area4";
}
```

Hemos definido 4 áreas diferentes. Una celda se ha dejado en blanco.

Para definir los elementos, ya no será necesario especificar los valores desde-hasta. Basta con referenciar el área correspondiente.

```css
.grid-item1 {
  background: blue;
  text-align: center;
  border: black 5px solid;
  grid-area: area1;
}
.grid-item2 {
  background: red;
  text-align: center;
  border: black 5px solid;
  grid-area: area2;
}
.grid-item3 {
  background: green;
  text-align: center;
  border: black 5px solid;
  grid-area: area3;
}
.grid-item4 {
  background: yellow;
  text-align: center;
  border: black 5px solid;
  grid-area: area4;
}
```


DigitalHouse>
Coding School