Fundamentos HTML, CSS

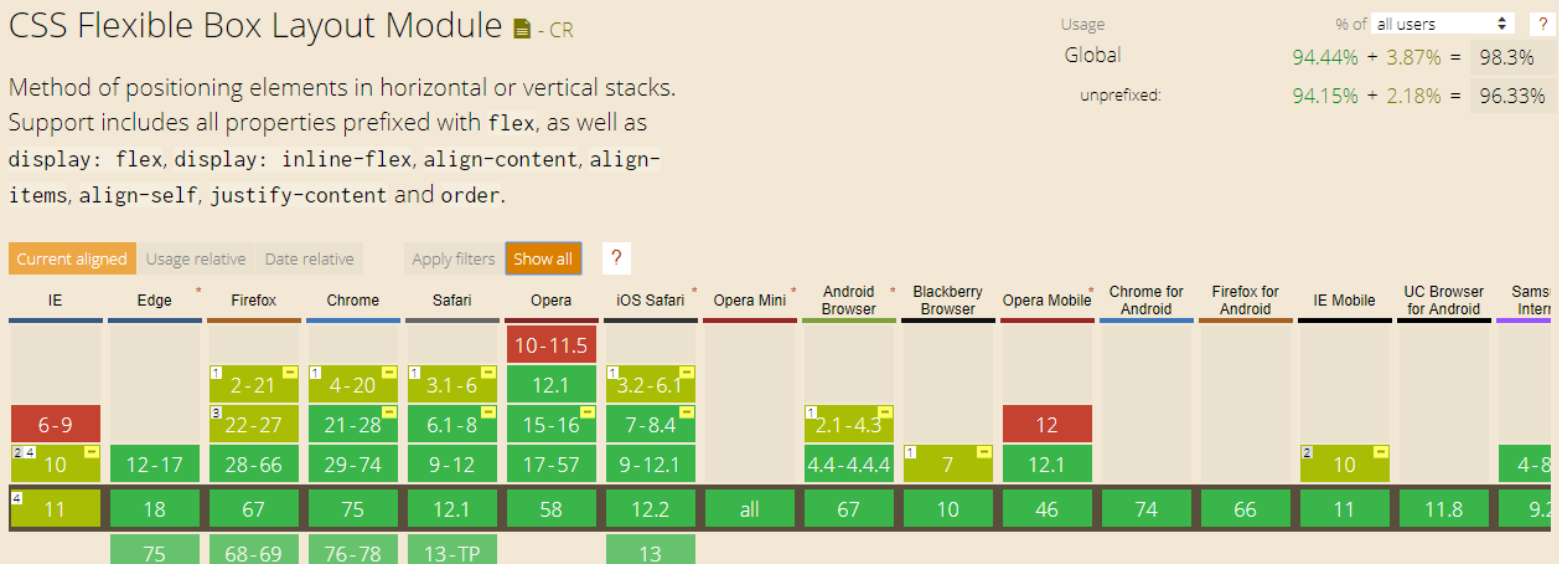
Flexbox y CSS Grid

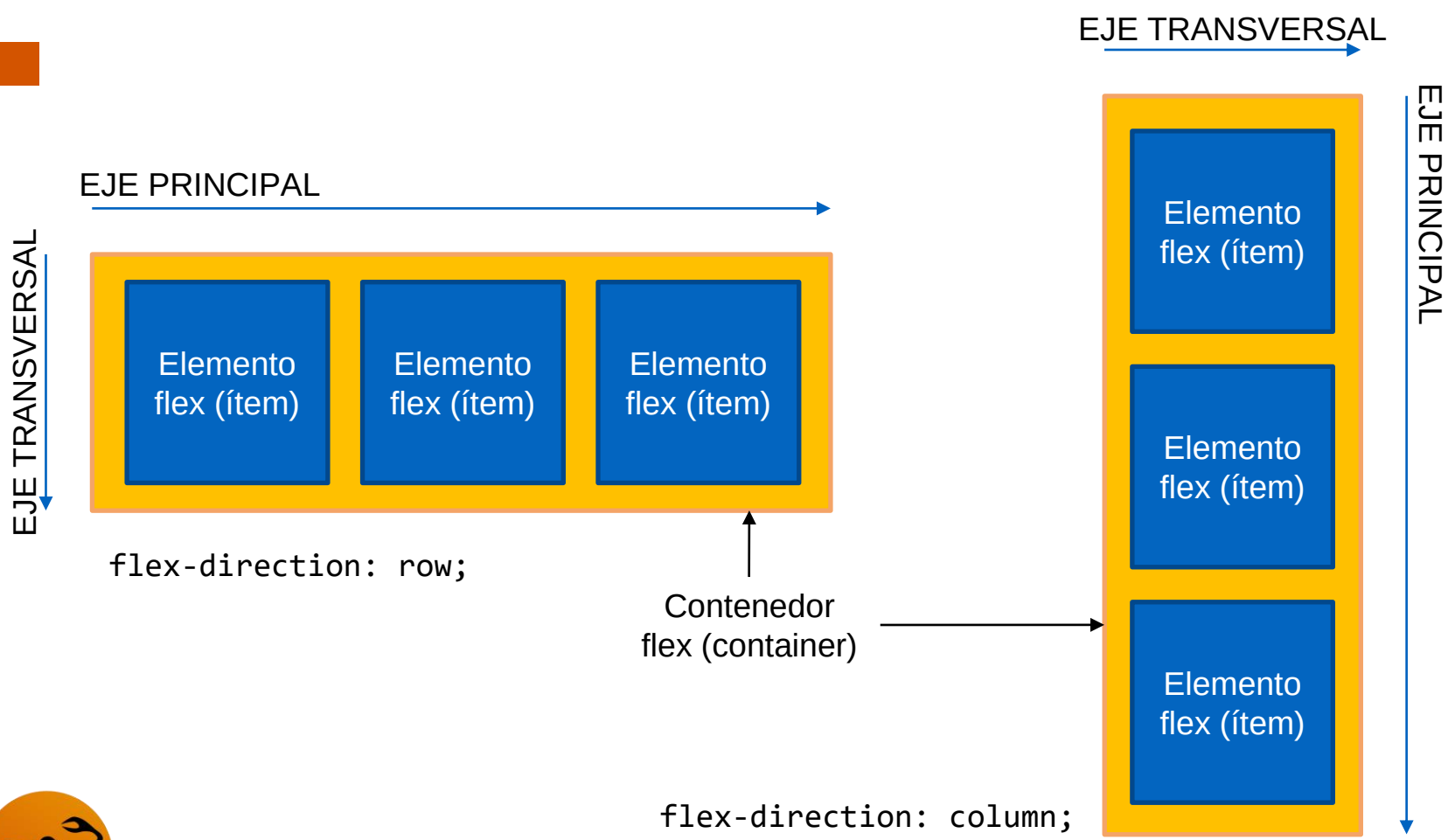
Tony G. Bolaño
@tonybolanyo – tonybolanyo@gmail.com
KeepCoding Full Stack Web Developer
Junio 2020

Flexbox

Flexbox

- Método para posicionar elementos en una sola dimensión





Flex Container

```
display: flex;
display: inline-flex;

flex-flow: row nowrap; (flex-direction flex-wrap)

flex-direction: row;
flex-direction: row-reverse;
flex-direction: column;
flex-direction: column-reverse;

flex-wrap: nowrap;
flex-wrap: wrap;
flex-wrap: wrap-reverse;

justify-content: flex-start;
justify-content: center;
justify-content: flex-end;
justify-content: space-between;
justify-content: space-around;
```

```
align-items: stretch;
align-items: flex-start;
align-items: center;
align-items: flex-end;
align-items: baseline;

align-content: stretch;
align-content: flex-start;
align-content: center;
align-content: flex-end;
align-content: space-between;
align-content: space-around;
```

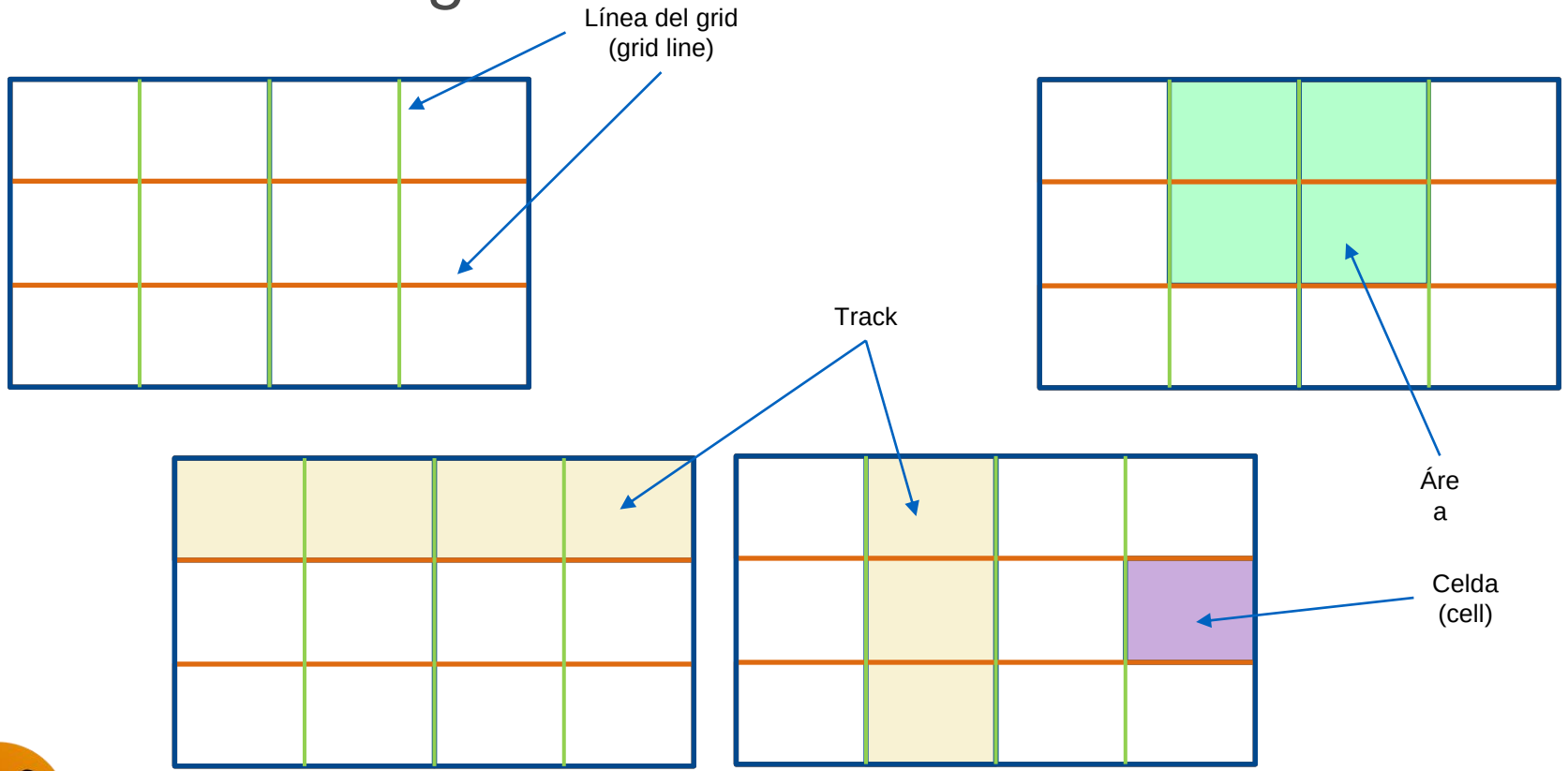
Flex Item

```
order: 0;

flex: 0 1 auto; (flex-grow flex-shrink flex-basis)

flex-grow: 0;
flex-grow: 1;

flex-shrink: 0;
flex-shrink: 1;

flex-basis: auto;

align-self: auto;
align-self: flex-start;
align-self: center;
align-self: flex-end;
align-self: baseline;
align-self: stretch;
```

Fuente: https://darekkay.com/dev/flexbox-cheatsheet.html

CSS Grid

El modelo de grid



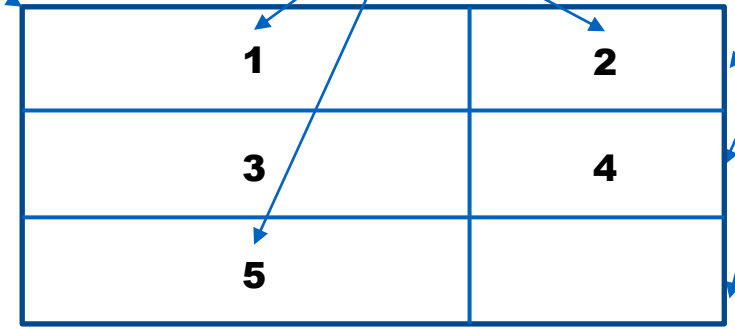
HTML Markup y CSS básico

Contenedor como grid

Elementos del grid

Las filas se crean según sea necesario (implícitas).

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



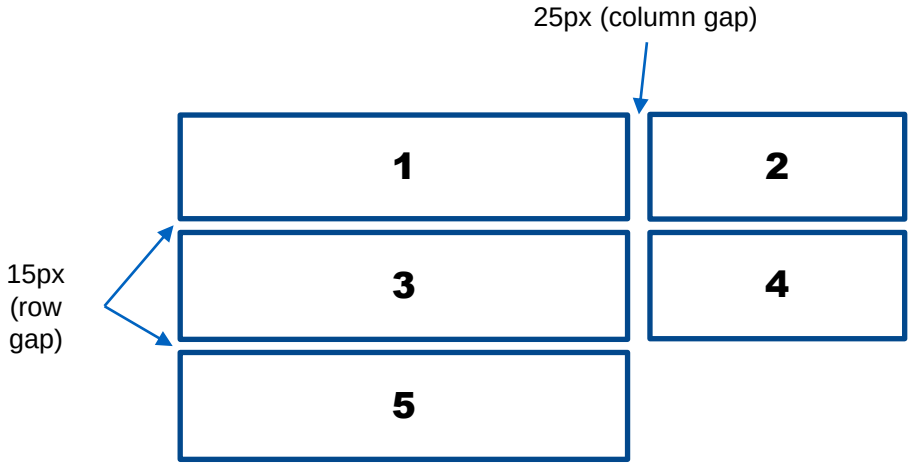

```
.container {
  display: grid;
  grid-template-columns: 400px 200px;
}
```

Dos columnas con ancho específico.
Las columnas están definidas explícitamente

Espaciado entre celdas

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

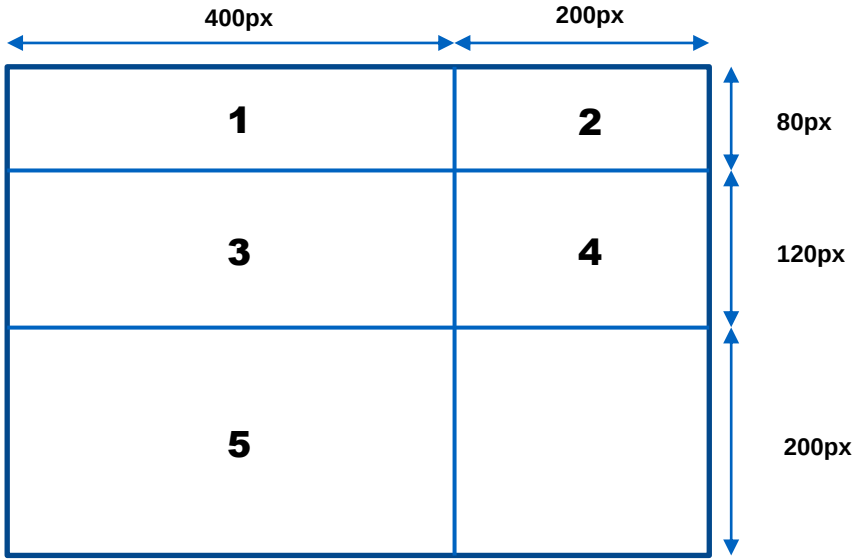
```
.container {
  display: grid;
  grid-template-columns: 400px 200px;
  grid-gap: 15px 25px;
}
```



Definir filas y columnas explícitamente

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

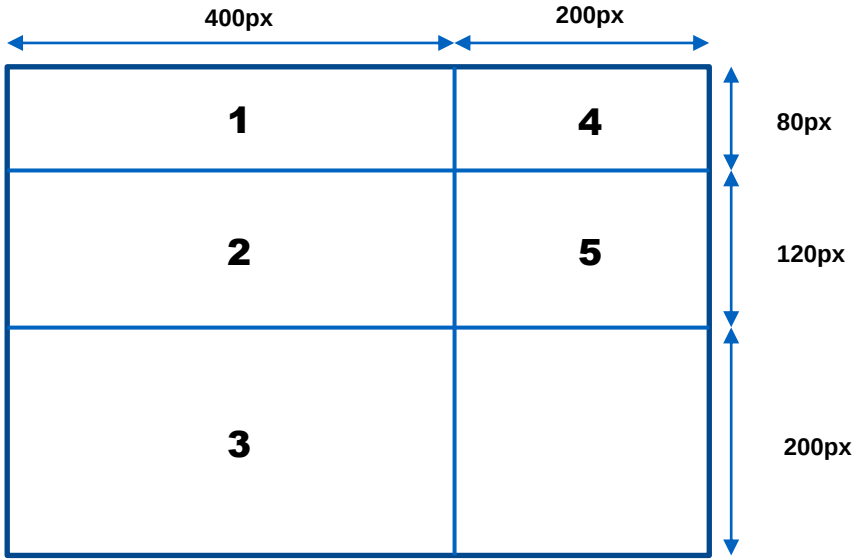
```
.container {
  display: grid;
  grid-template-columns: 400px 200px;
  grid-template-rows: 80px 120px 200px;
}
```



Elementos en columnas

```html
<div class="container">
  <div class="item">1</div>
  <div class="item">2</div>
  <div class="item">3</div>
  <div class="item">4</div>
  <div class="item">5</div>
</div>
```

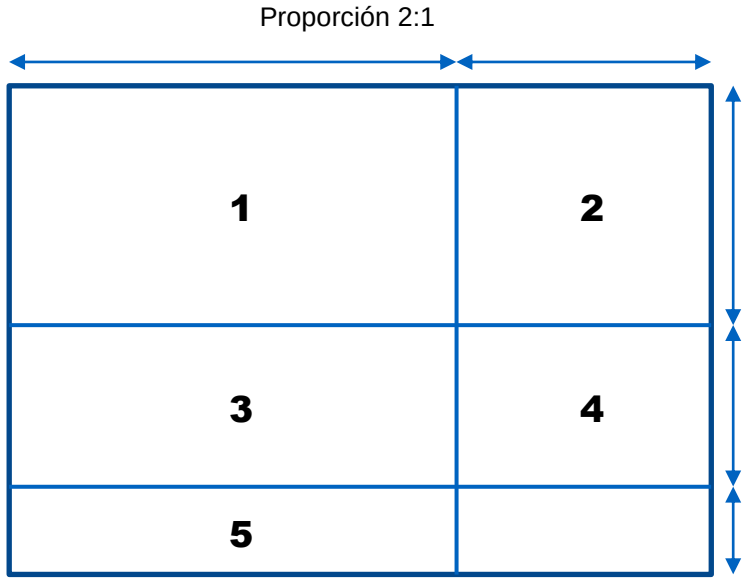
```css
.container {
  display: grid;
  grid-template-columns: 400px 200px;
  grid-template-rows: 80px 120px 200px;
  grid-auto-flow: column;
}
```



Unidades relativas: fr

```html
<div class="container">
  <div class="item">1</div>
  <div class="item">2</div>
  <div class="item">3</div>
  <div class="item">4</div>
  <div class="item">5</div>
</div>
```

```css
.container {
  display: grid;
  grid-template-columns: 2fr 1fr;
  grid-template-rows: 3fr 2fr 1fr;
}
```

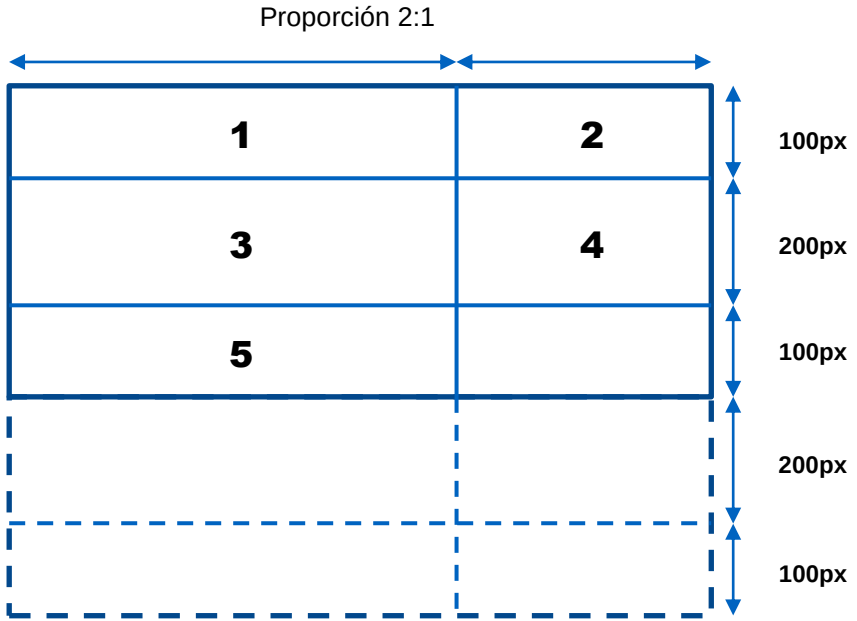


grid-auto-rows

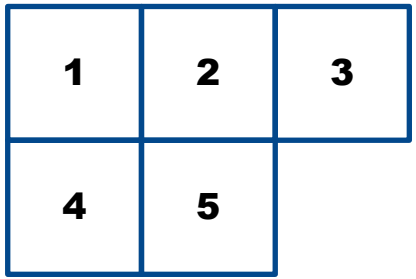
```html
<div class="container">
  <div class="item">1</div>
  <div class="item">2</div>
  <div class="item">3</div>
  <div class="item">4</div>
  <div class="item">5</div>
</div>
```

```css
.container {
  display: grid;
  grid-template-columns: 2fr 1fr;
  grid-auto-rows: 100px 200px;
}
```

Según se necesite, se agregarán filas siguiendo el mismo patrón



Funciones repeat() y minmax()




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

Tres columnas de 200px

```
.container {
  display: grid;
  grid-template-columns: repeat(3,
200px);
  grid-auto-rows: 100px;
}
```

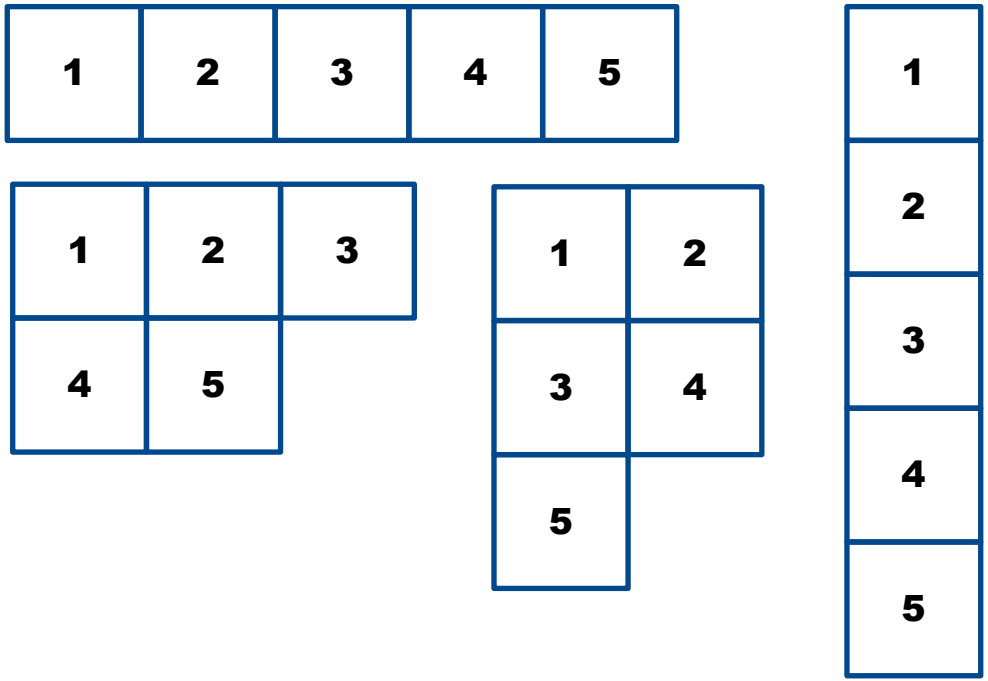
Tres columnas ajustadas al espacio disponible

```
.container {
  display: grid;
  grid-template-columns: repeat(3,
1fr);
  grid-auto-rows: 100px;
}
```

Tres columnas ajustadas al espacio disponible con un mínimo de 200px

```
.container {
  display: grid;
  grid-template-columns: repeat(3, minmax(200px,
1fr));
  grid-auto-rows: 100px;
}
```

auto-fill y auto-fit



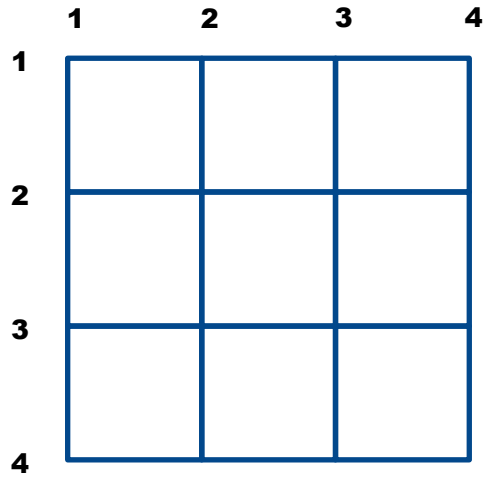
Con `auto-fill` se ajusta cuando no hay espacio suficiente

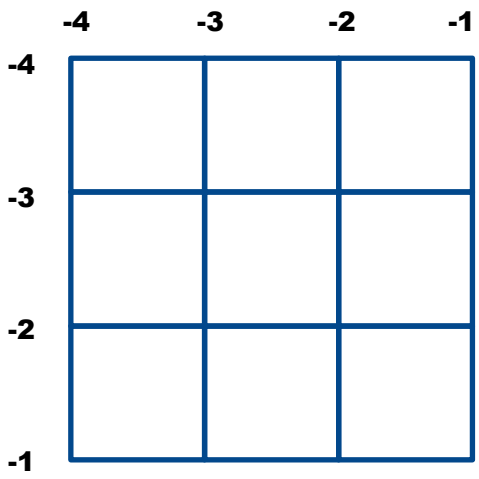
```
.container {
  display: grid;
  grid-template-columns: repeat(auto-fill, minmax(150px, 1fr));
  grid-auto-rows: 150px;
}
```

Con `auto-fit`, además, se ajusta a todo el espacio, incluso cuando es mayor

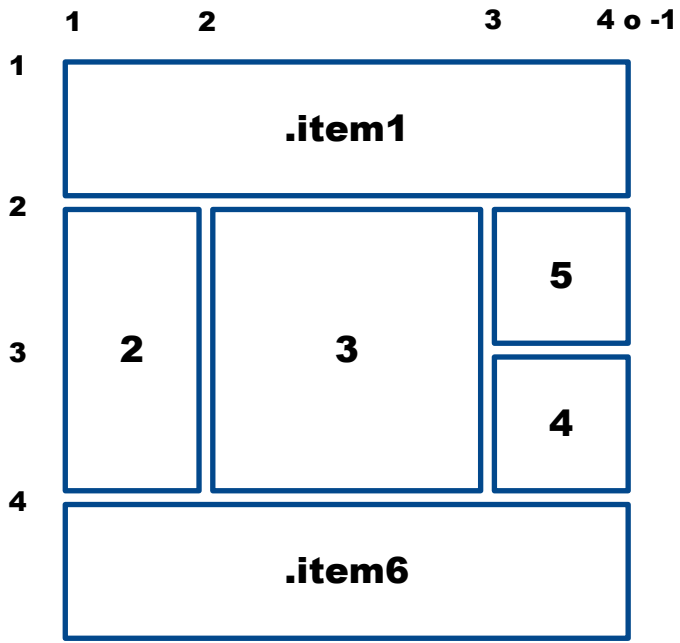
```
.container {
  display: grid;
  grid-template-columns: repeat(auto-fill, minmax(150px, 1fr));
  grid-auto-rows: 150px;
}
```

Cómo se numeran las líneas del grid





Colocar objetos en el grid



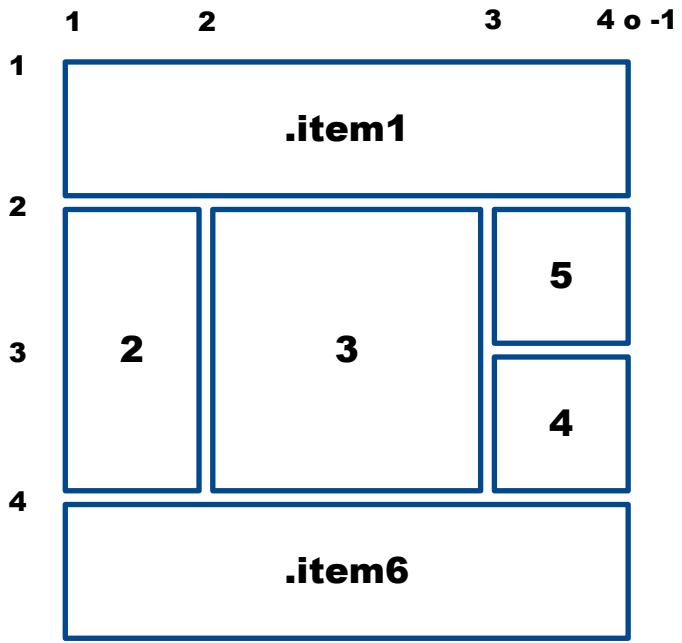

```css
.container {
  display: grid;
  grid-template-columns: 100px minmax(200px, auto) 100px;
  grid-auto-rows: 100px;
  grid-gap: 10px;
}

.item1 {
  grid-column-start: 1;
  grid-column-end: -1;
}

.item6 {
  grid-column: 1 / -1;
}

.item2, .item3 {
  grid-row: span 2;
}

.item5 {
  grid-column: 3;
  grid-row: 2;
}
```

De la primera a la última línea

Ocupa dos filas

Se mueva a la columna 3, fila 2

Nombrar las líneas del grid

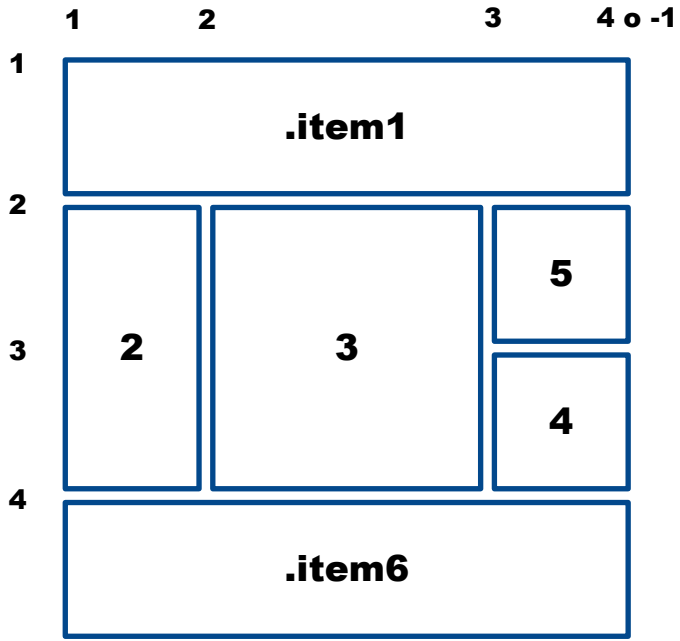



```
.container {
  display: grid;
  grid-template-columns: [grid-start nav-start] 100px
                         [nav-end main-start] minmax(200px, auto)
                         [main-end] 100px
                         [grid-end];
  grid-auto-rows: 100px;
  grid-gap: 10px;
}

.item1 {
  grid-column: grid-start / grid-end;
}

.item6 {
  grid-column: 1 / -1;
}

.item2 {
  grid-column: nav-start / nav-end;
  grid-row: span 2;
}

.item3 {
  grid-column: main-start / main-end;
  grid-row: span 2;
}

.item5 {
  grid-column: 3;
  grid-row: 2;
}
```

Definimos los nombres al definir la plantilla

Los usamos para establecer la posición inicial y final

Definir áreas del grid




```
.container {
  display: grid;
  grid-template-columns: 100px minmax(200px, auto) 100px;
  grid-template-rows: repeat(4, 100px);
  grid-gap: 10px;
  grid-template-areas: "header header header"
                       "nav main side1"
                       "nav main side2"
                       "footer footer footer";
}

.item1 {
  grid-area: header;
}

.item2 {
  grid-area: nav;
}

.item3 {
  grid-area: main;
}

.item4 {
  grid-area: side2;
}

.item5 {
  grid-area: side1;
}

.item6 {
  grid-area: footer;
}
```

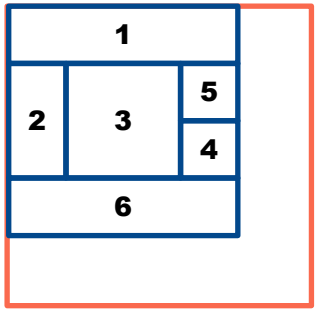
Definimos los nombres de las áreas del grid

Asociamos los elementos a su área usando el nombre

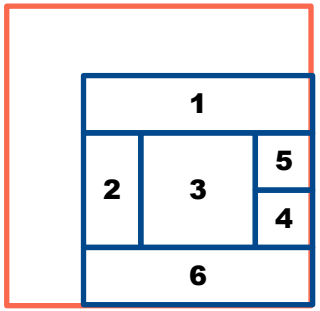
Cómo se calcula la posición de los elementos

1. Se generan los elementos del grid como anónimos
2. Se colocan los elementos con posición explícita
3. Se colocan los elementos con fila explícita, pero sin columna
4. Se calcula el número de columnas implícitas del grid
5. Se coloca el resto de elementos

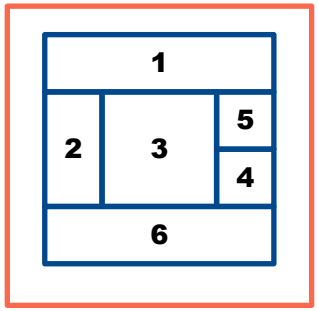
Alineación del grid



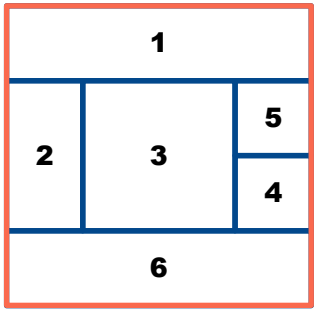

```
justify-content:
start;
align-content: start;
```




```
justify-content: end;
align-content: end;
```



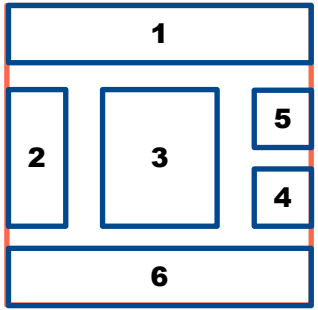

```
justify-content:
center;
align-content: center;
```



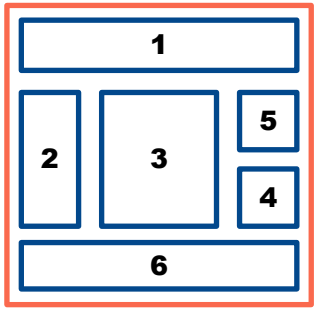

```
justify-content:
stretch;
align-content: stretch;
```

justify-content: ↔
(eje de columnas);

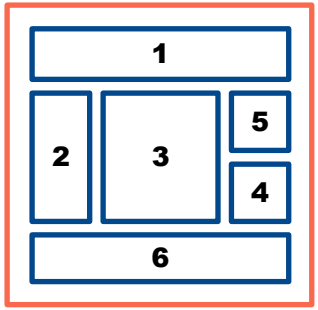
align-content:
(eje de filas); ↕




```
justify-content: space-
between;
align-content: space-between;
```




```
justify-content: space-around;
align-content: space-around;
```




```
justify-content: space-
evenly;
align-content: space-evenly;
```

Alineación de elementos

- En el contenedor, se aplica para todos los elementos:
  - `justify-items`: eje de columna ↔
  - `align-items`: eje de fila ↕
  - Valores: `start` | `end` | `center` | `stretch`

- Para un solo elemento:
  - `justify-self`: eje de columna ↔
  - `align-self`: eje de fila ↕
  - Valores: `start` | `end` | `center` | `stretch`

GRACIAS

www.keepcoding.io