



# CSS GRID



**CSS Grid** nos permite diagramar una página (o partes de ella) como si fuera una **grilla**.

- No es un reemplazo de **Flexbox** (Flex), pero puede lograr muchas de las mismas cosas.
- Con Grid podemos definir **filas**, **columnas** e incluso **áreas**.
- Además de esto, podemos nombrar **regiones** o colocar elementos declarativamente.

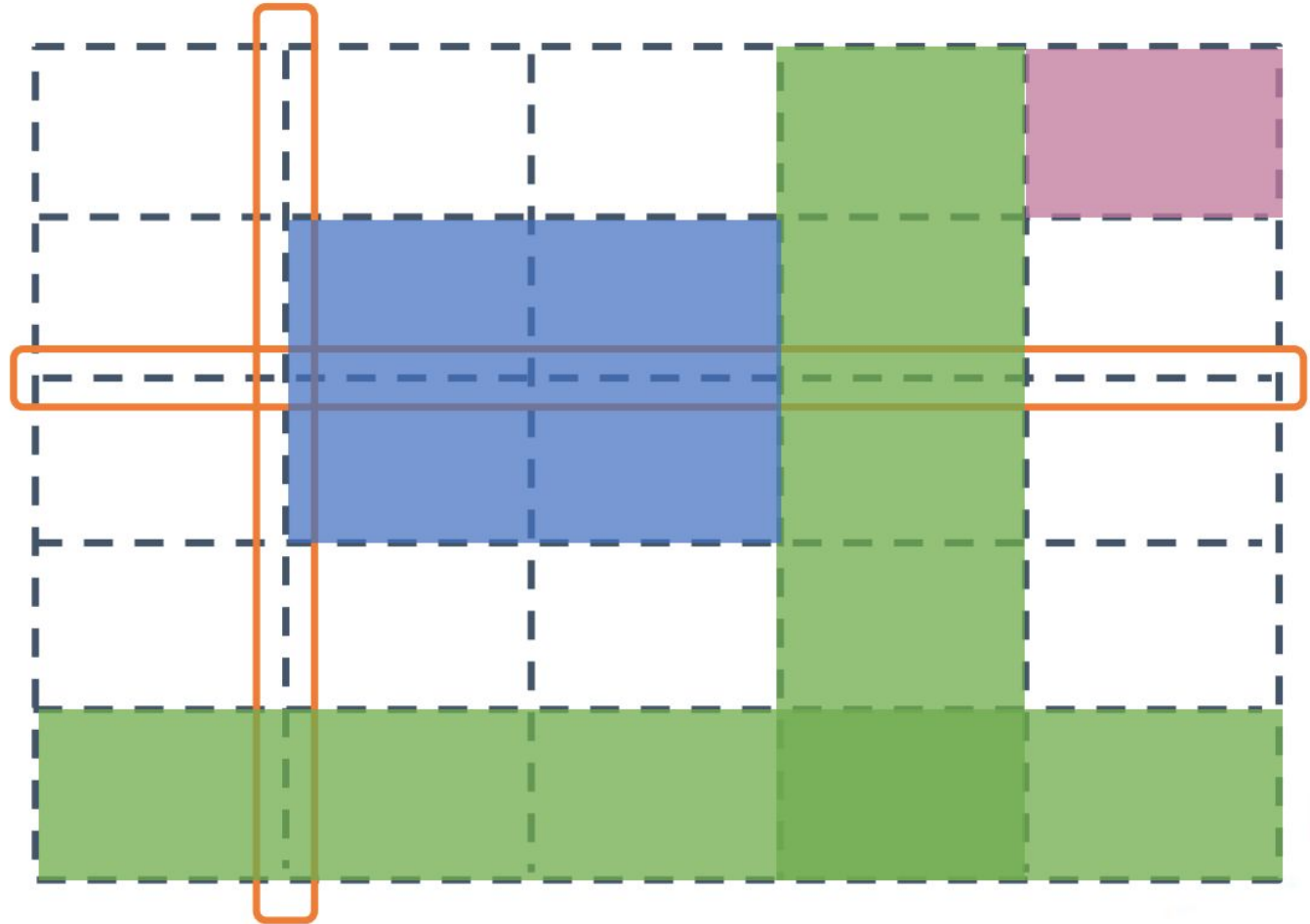
## CSS GRID - Intro

Para empezar a utilizar **Grid**, sólo tenemos que especificarlo de la siguiente manera:

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

## ¿Qué hay en un contenedor Grid?

- **Grid:** la grilla completa.
- **Grid lines:** son las líneas verticales u horizontales que dividen los tracks.
- **Grid Tracks:** filas o columnas completas.
- **Grid Areas:** conjunto rectangular de celdas
- **Grid Cell:** una única celda.



# Grid vs. Flex

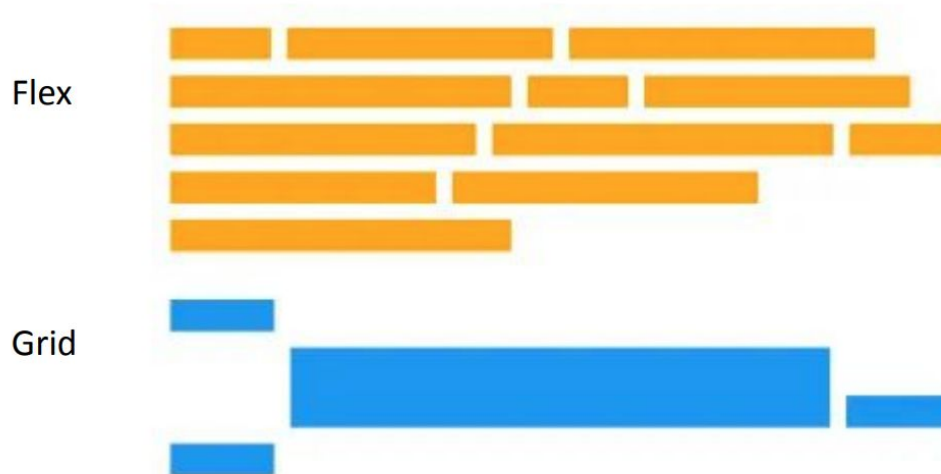
¿Cuál elegir?



## CSS GRID - ¿Cuál elegir?

**Flex es unidimensional.** Si bien tenemos control sobre el “eje secundario”, los elementos se van apilando uno tras otro en el eje principal y no se permite declarar la ubicación granularmente más allá del orden de los elementos.

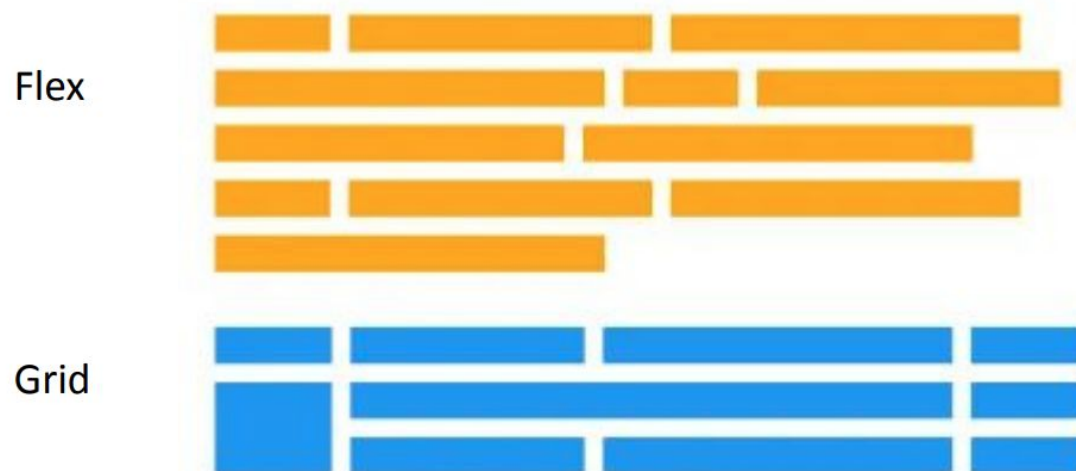
**Grid es bidimensional.** Podemos pensar en Grid como “bidimensional” ya que podemos declarar el tamaño de filas y columnas y ubicar elementos a gusto de manera explícita en ellas.



## CSS GRID - ¿Cuál elegir?

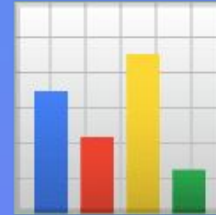
**Flex es muy bueno para listas.** Gracias a la habilidad de hacer “wrap” en un contenedor Flex, podemos crear listas de elementos que sigan cierto flujo sin preocuparnos de la cantidad de elementos.

**Grid es muy bueno para layouts.** Gracias a la versatilidad en cuanto a columnas, filas y áreas, podemos diagramar sitios enteros en un sólo contenedor Grid.





# CSS GRID



## CSS GRID - Filas y columnas

Podemos crear filas y columnas de manera explícita utilizando **grid-template-rows** y **grid-template-columns**.

```
.contenedor {  
  display: grid;  
  /* Dos columnas con valor absoluto */  
  grid-template-columns: 200px 100px;  
  /* Tres filas de valor absoluto */  
  grid-template-rows: 50px 100px 200px;  
}
```

```
.contenedor {  
  display: grid;  
  /* Dos columnas en fracciones */  
  grid-template-columns: 1fr 2fr;  
  /* Tres filas de valor absoluto */  
  grid-template-rows: 50px 100px 200px;  
}
```

## CSS GRID - Separar columnas y filas (gap)

Podemos separar celdas utilizando **gap**, **column-gap** y/o **row-gap**. Si utilizamos **gap**, podemos definir la distancia entre filas y columnas en una única declaración.

```
.contenedor {  
  display: grid;  
  /* Ambas distancias iguales */  
  gap: 10px;  
}
```

```
.contenedor {  
  display: grid;  
  /* Primero fila luego columna */  
  gap: 10px 20px;  
}
```

```
.contenedor {  
  display: grid;  
  /* Gap individual */  
  row-gap: 10px;  
  column-gap: 20px;  
}
```

## CSS GRID - Áreas o secciones

```
.contenedor {
  display: grid;
  width: 100%;
  height: 100vh;
  grid-template-areas: "head head"
                      "nav main"
                      "nav footer";
  grid-template-rows: 200px 1fr 100px;
  grid-template-columns: 200px 1fr;
}

.contenedor header {
  grid-area: head;
  background-color: rosybrown;
}

.contenedor nav {
  grid-area: nav;
  background-color: aquamarine;
}

.contenedor main {
  grid-area: main;
  background-color: blueviolet;
}

.contenedor footer {
  grid-area: footer;
  background-color: yellowgreen;
}
```

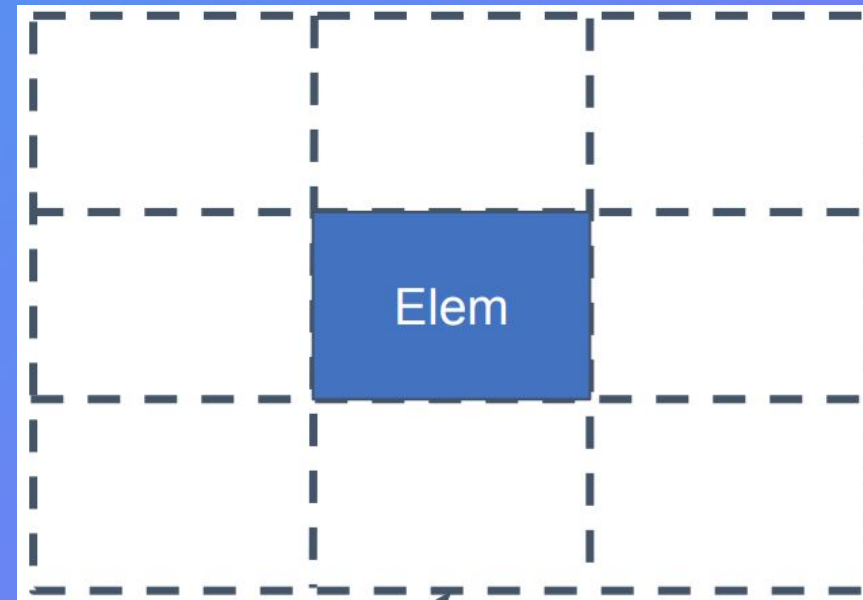
Podemos nombrar filas y columnas utilizando **grid-template-areas** y luego asignar elementos a esas “áreas” utilizando **grid-area**.



## CSS GRID - Posicionar elementos por columna y fila

Además de utilizando “áreas”, podemos posicionar un elemento utilizando **grid-column** y **grid-row**.

```
.contenedor {  
  display: grid;  
  width: 100%;  
  height: 100vh;  
  background-color: yellowgreen;  
  grid-template-rows: 1fr 1fr 1fr;  
  grid-template-columns: 1fr 1fr 1fr;  
}  
  
.contenedor div {  
  background-color: rosybrown;  
  border: 2px solid #000;  
  
  grid-column: 2;  
  grid-row: 2;  
}
```

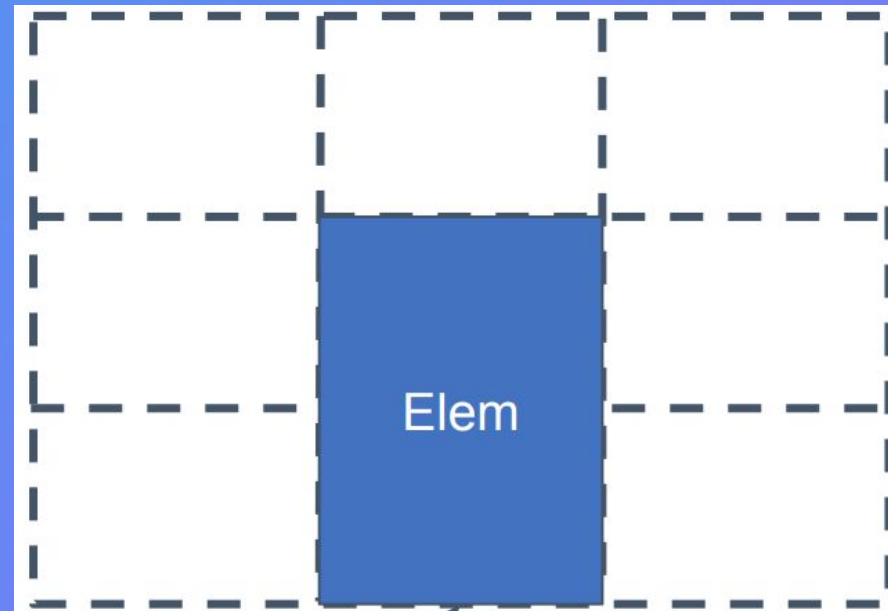


## CSS GRID - Posicionar elementos por columna y fila

Podemos también definir cuántas filas o columnas debe abarcar utilizando la palabra clave **span**.

En el siguiente ejemplo, el elemento está siendo ubicado en la segunda fila y abarca 2 filas.

```
.contenedor {  
  display: grid;  
  width: 100%;  
  height: 100vh;  
  background-color: yellowgreen;  
  grid-template-rows: 1fr 1fr 1fr;  
  grid-template-columns: 1fr 1fr 1fr;  
}  
  
.contenedor div {  
  background-color: rosybrown;  
  border: 2px solid #000;  
  
  grid-column: 2;  
  grid-row: 2 / span 2;  
}
```



## CSS GRID - La función repeat()

La función **repeat()** nos facilita la creación de varias filas o columnas del mismo tamaño.

```
.contenedor {  
  display: grid;  
  grid-template-columns: 1fr 1fr 1fr;  
  grid-template-columns: repeat(1fr, 3);  
}
```

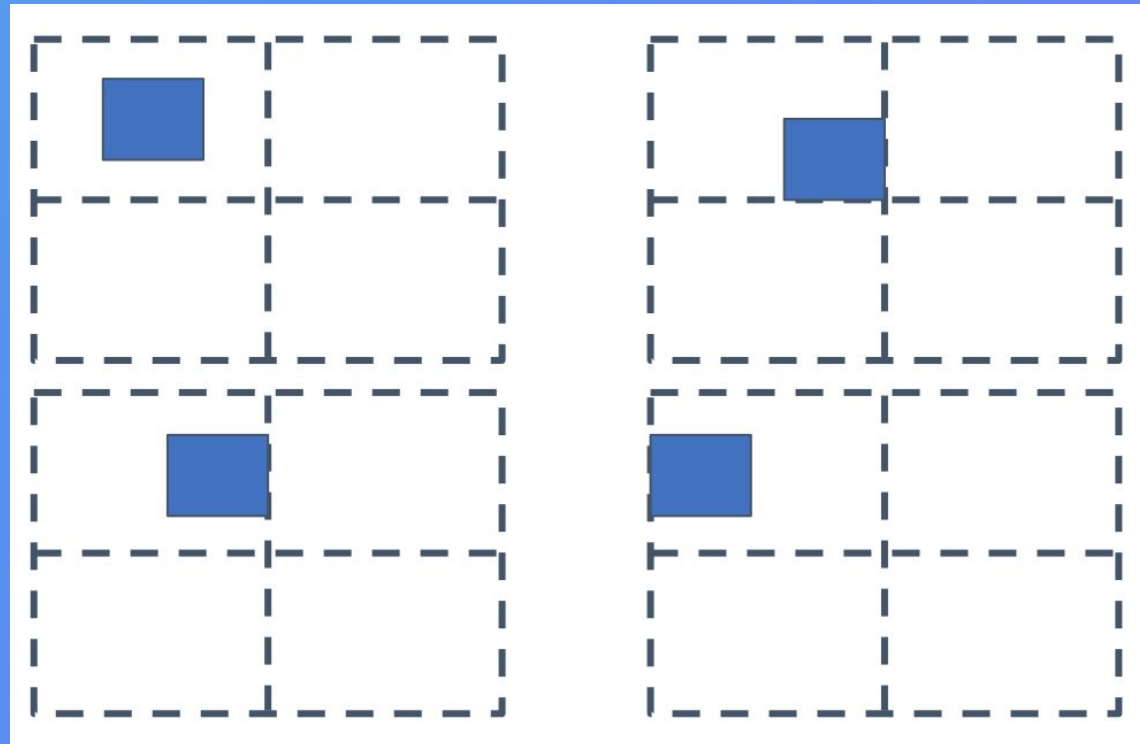




## CSS GRID - Alinear elementos

En Grid podemos utilizar **justify-content**, **justify-items**, **align-content** y **align-items** para cambiar la alineación.

**justify-items** y **align-items** alinean los elementos dentro de sus respectivas celdas.





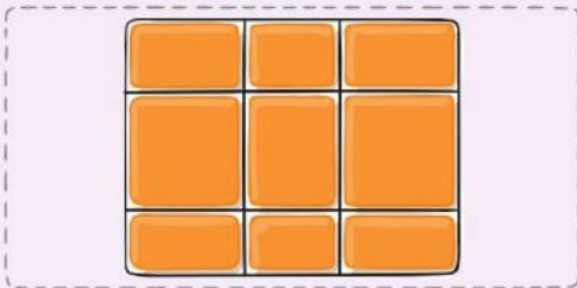
## CSS GRID - Alinear elementos

**justify-content** y **align-content** alinean la grilla dentro del contenedor (sólo si no ocupa el 100%).

```
.container {  
  justify-content: center;  
}
```

CSS

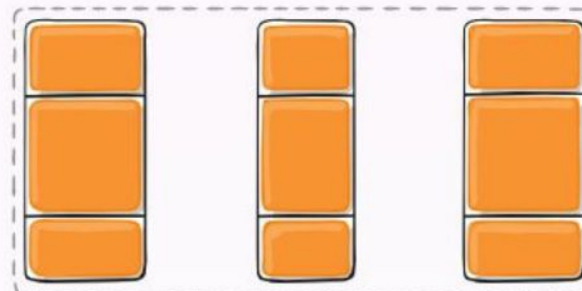
grid container



```
.container {  
  justify-content: space-between;  
}
```

CSS

grid container

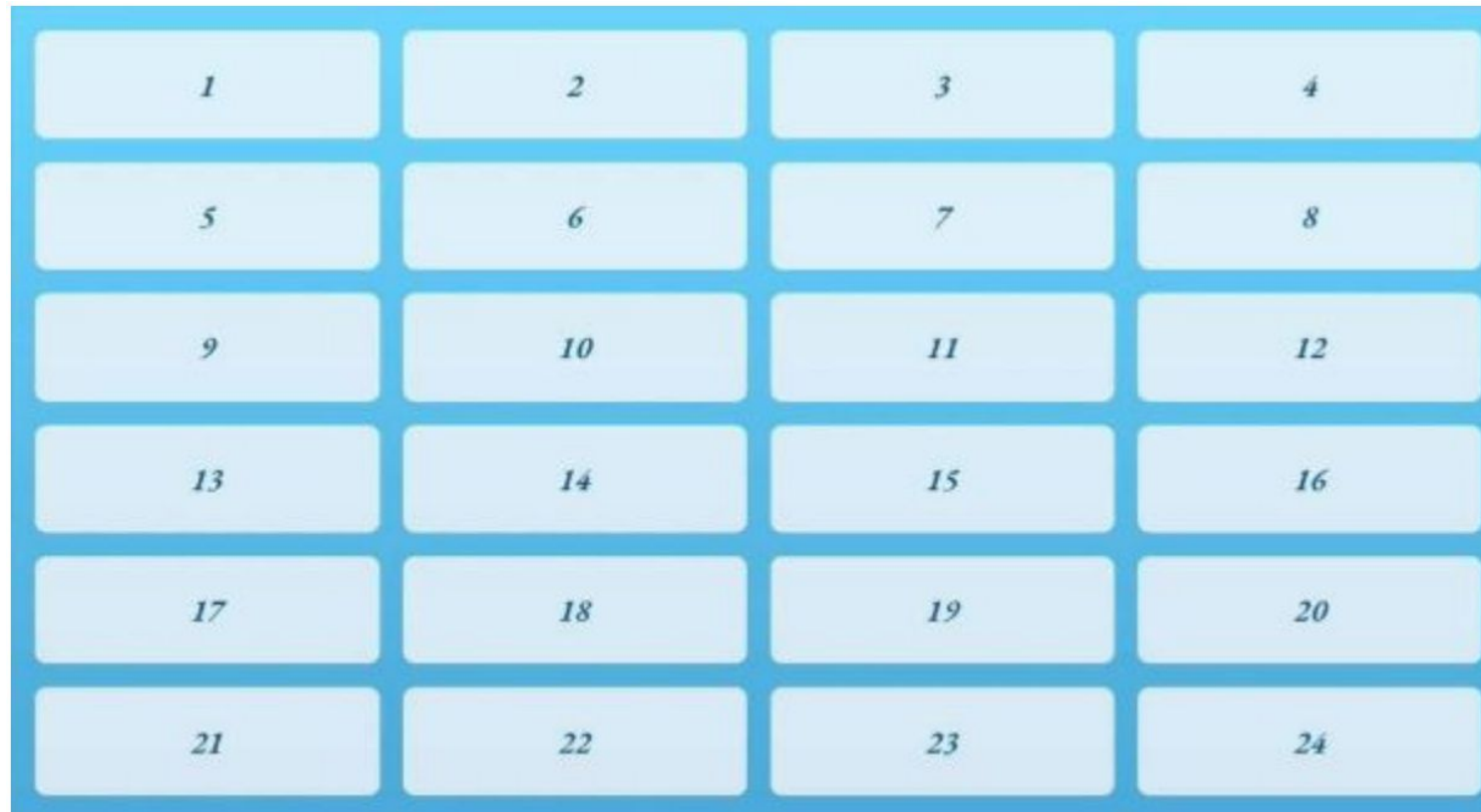


# Ejercicios



## CSS GRID - Ejercicios

Replicar el siguiente layout utilizando **grid** (pista: usar grid-template-columns y grid-gap)



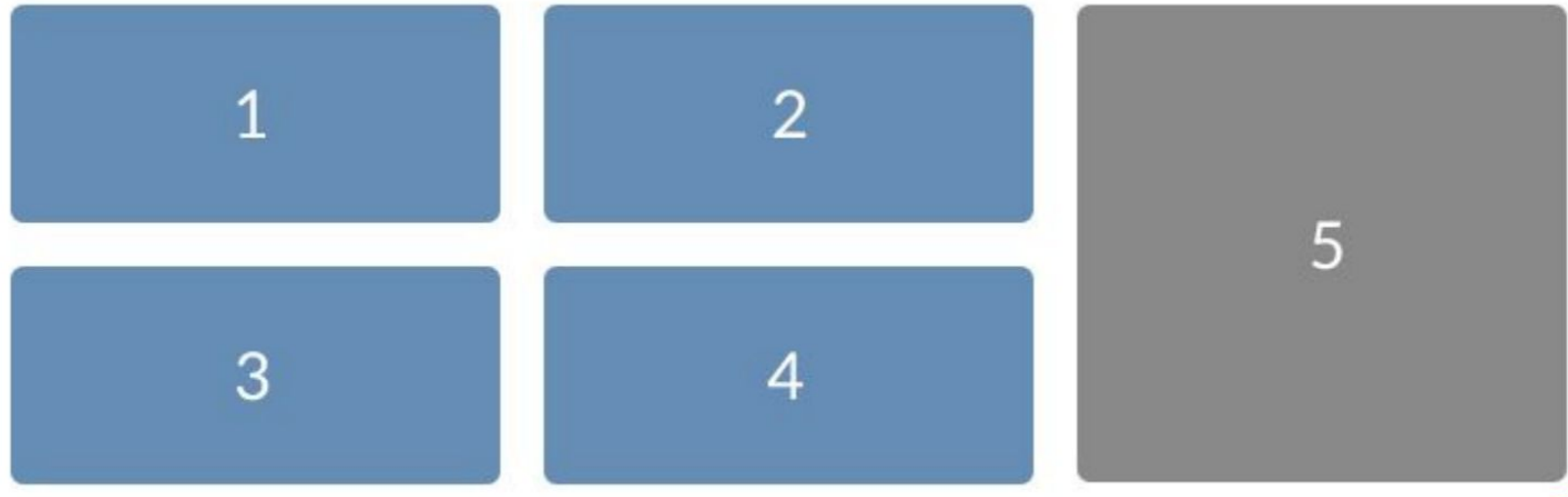
1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16
17	18	19	20
21	22	23	24

## CSS GRID - Ejercicios

Replicar el siguiente layout utilizando grid.

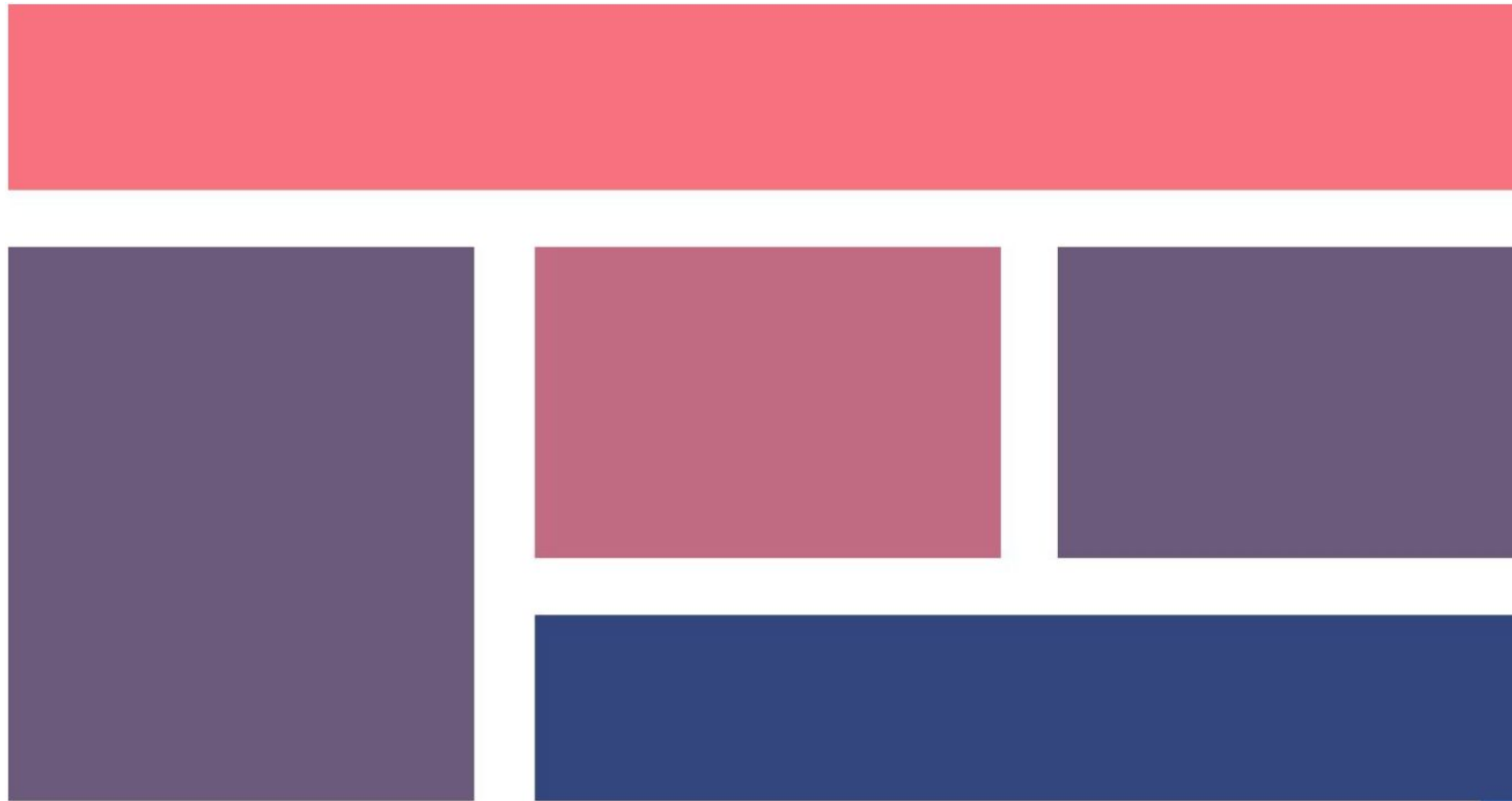
Pistas: **grid-template-columns** en el contenedor, **grid-column** y/o **grid-row** en el elemento a mover.

Nota: pueden utilizar el elemento 3 o 5 para el de la derecha.

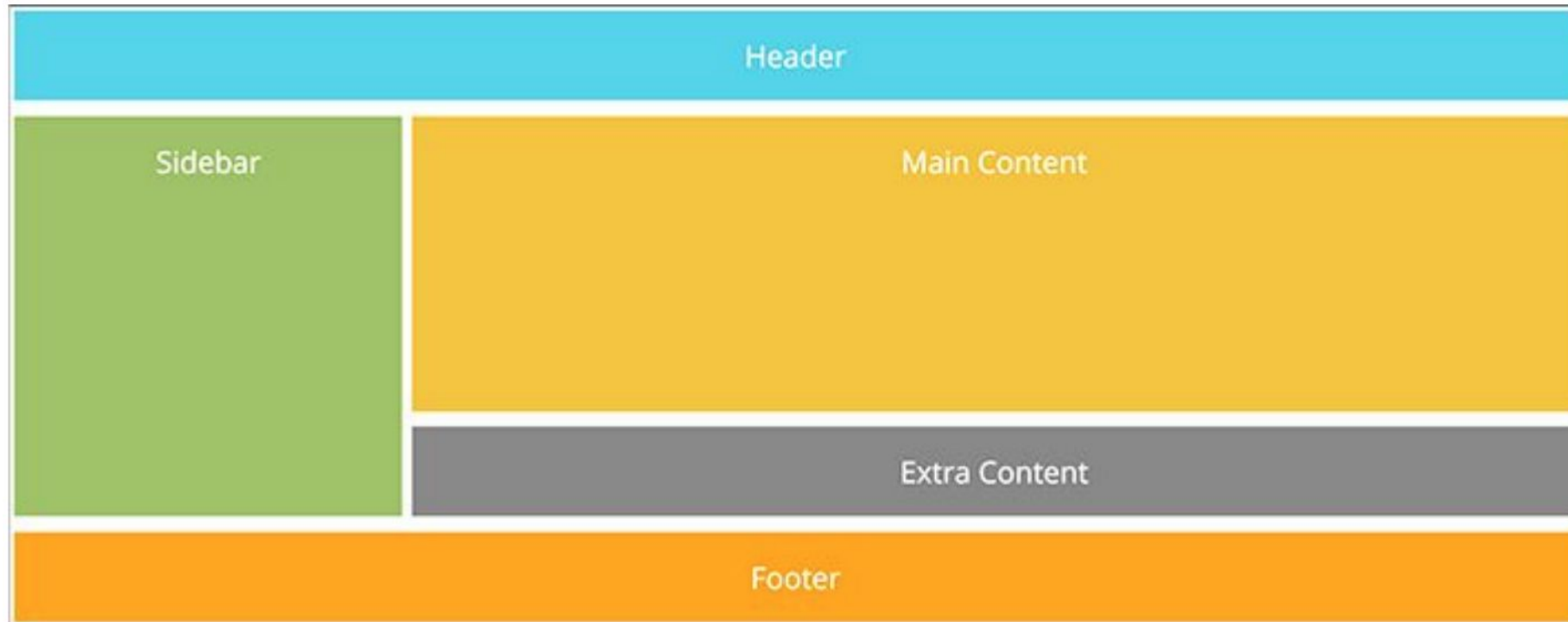


## CSS GRID - Ejercicios

Replicar el siguiente layout utilizando grid (pista: usar **grid-template-areas**)

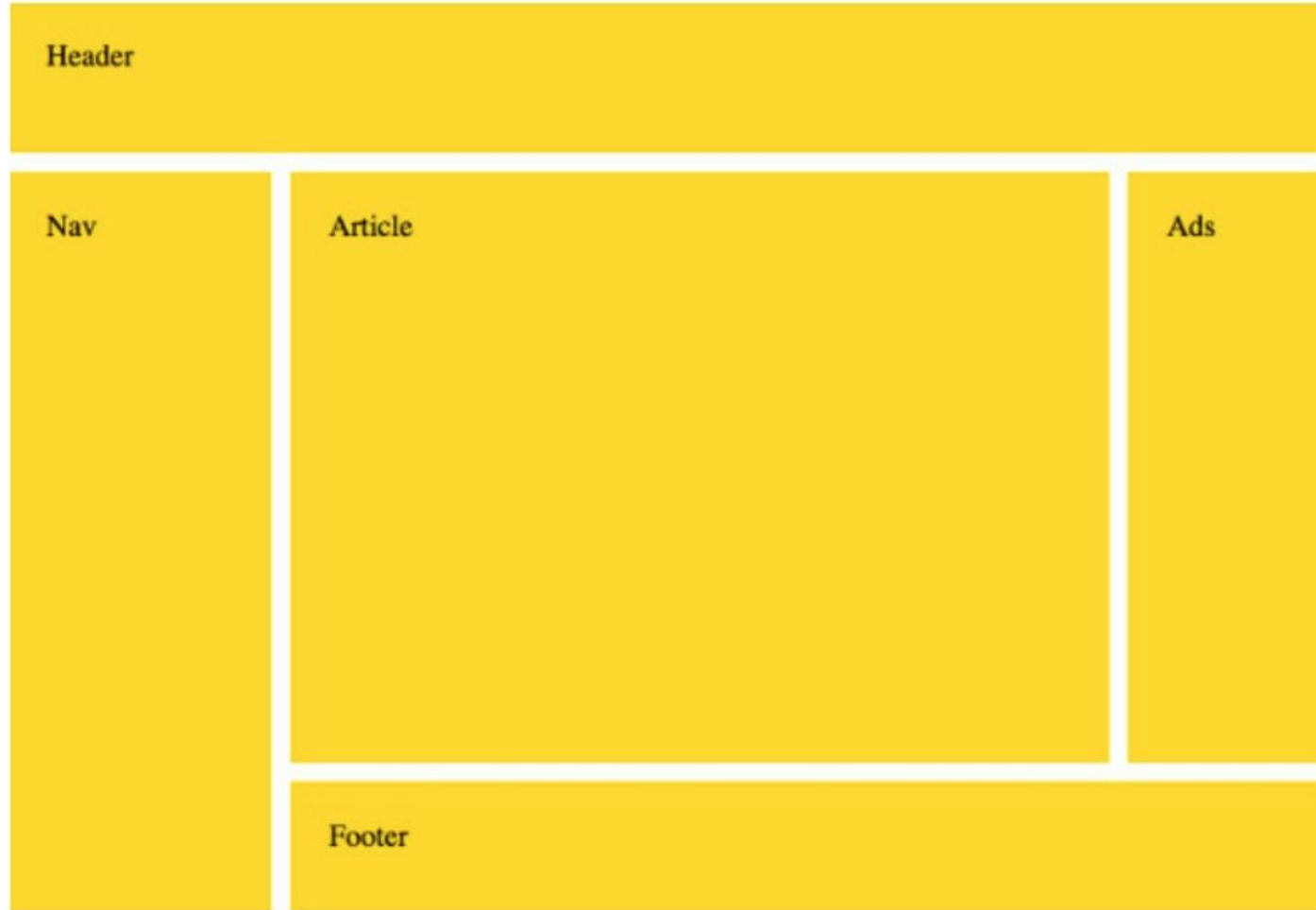


## CSS GRID - Ejercicios



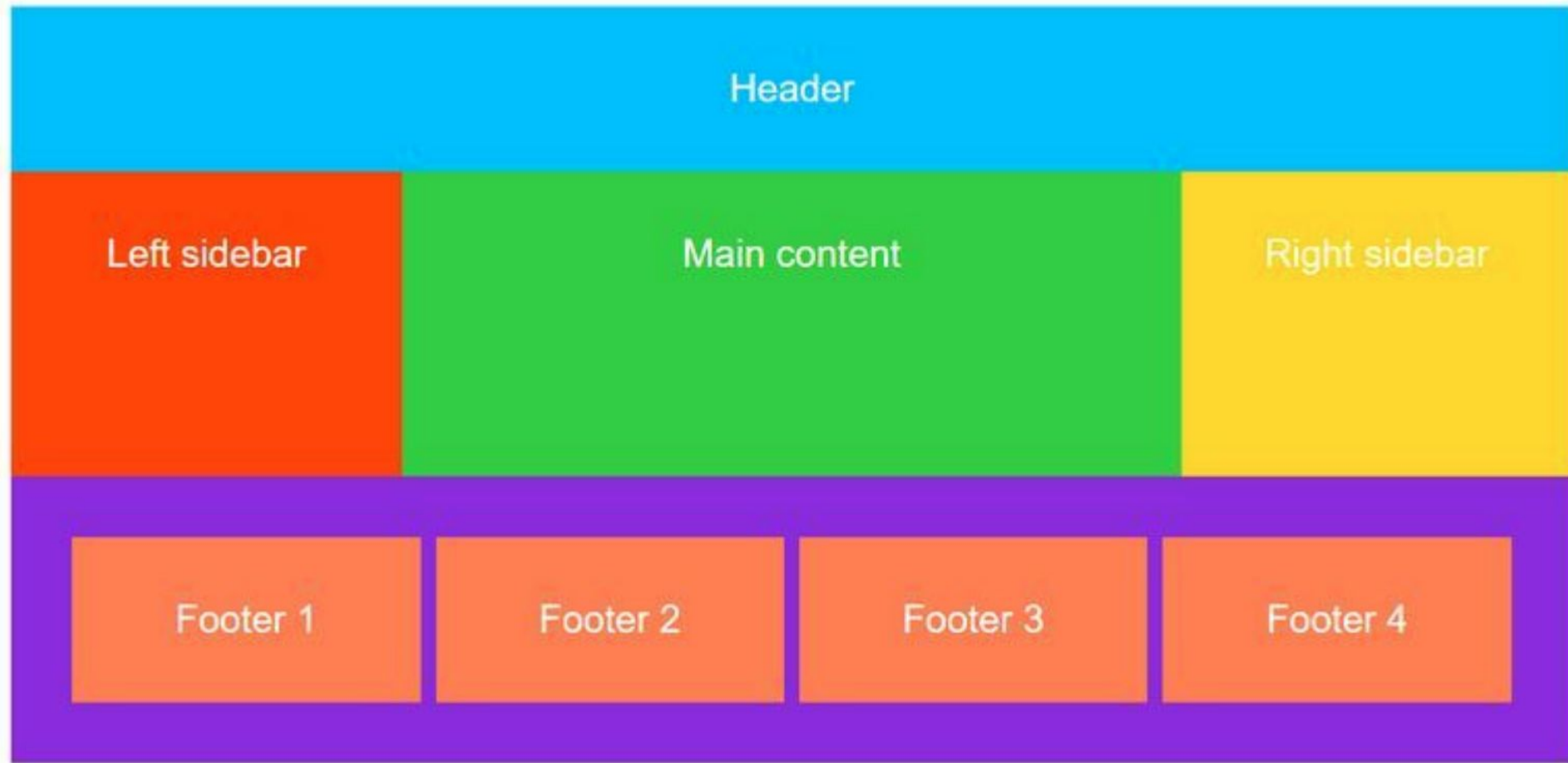
## CSS GRID - Ejercicios

## CSS GRID - Ejercicios

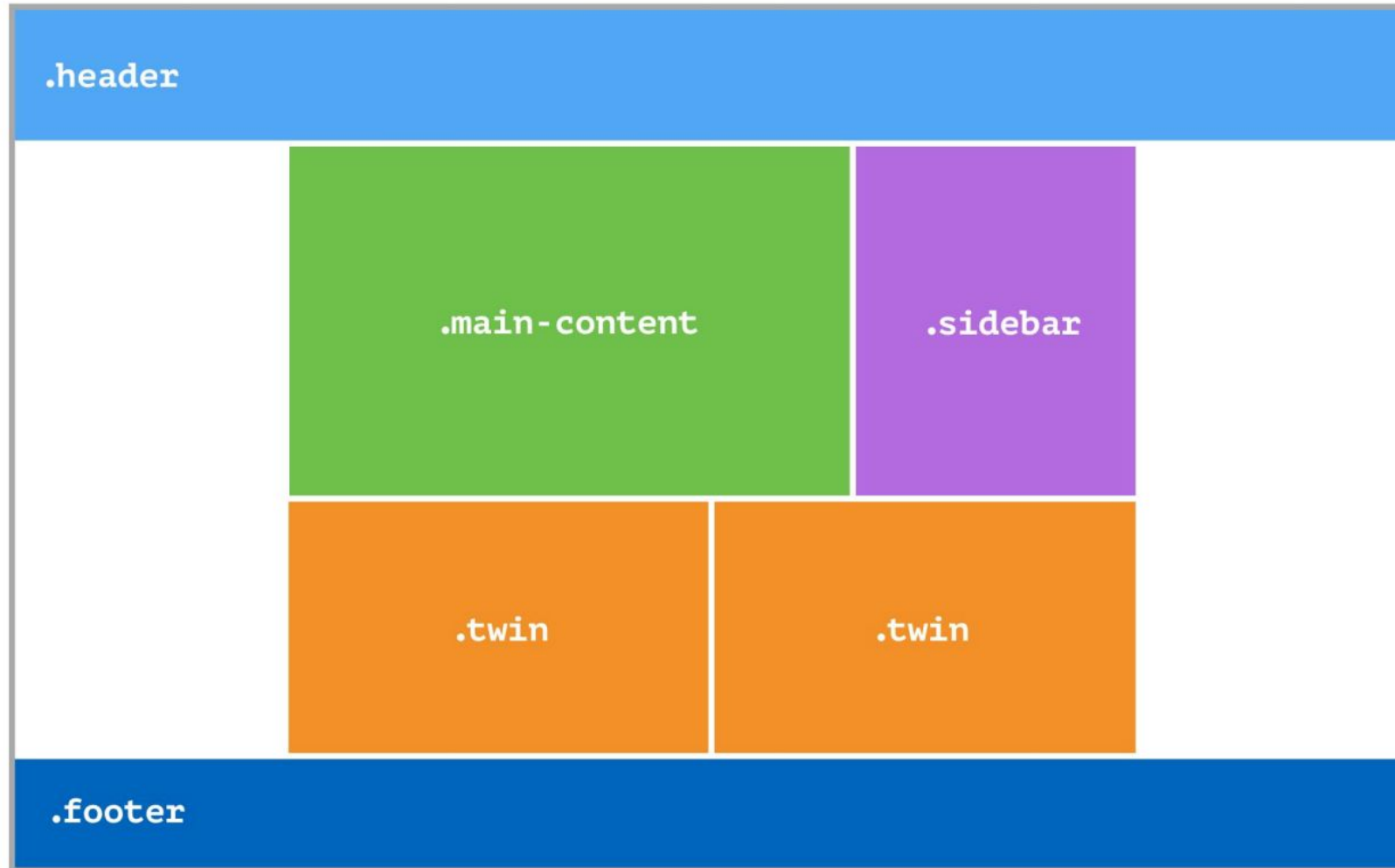




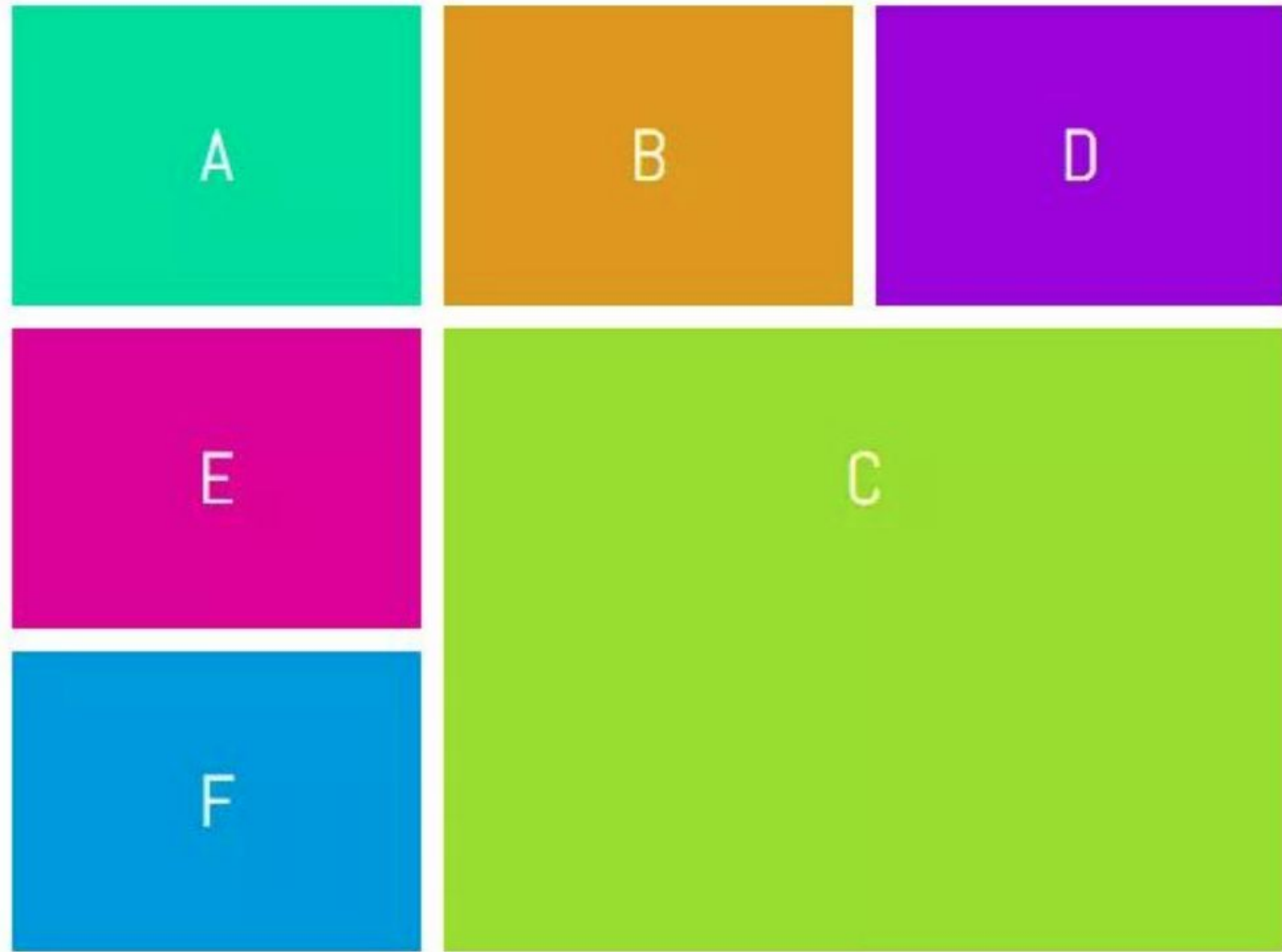
## CSS GRID - Ejercicios



## CSS GRID - Ejercicios



## CSS GRID - Ejercicios



# LINKS

# LINKS

- **CSS Grid**

- <https://grid.malven.co/>
- [https://developer.mozilla.org/es/docs/Web/CSS/CSS\\_Grid\\_Layout](https://developer.mozilla.org/es/docs/Web/CSS/CSS_Grid_Layout)
- <https://lenguajecss.com/css/maquetacion-y-colocacion/grid-css/>
- <https://css-tricks.com/snippets/css/complete-guide-grid/>
- <https://www.youtube.com/watch?v=-nVJwYKx6H8>
- <https://developer.mozilla.org/en-US/docs/Web/CSS/grid-template-columns>
- <https://developer.mozilla.org/en-US/docs/Web/CSS/grid-template-rows>
- <https://css-tricks.com/introduction-fr-css-unit/>
- [https://developer.mozilla.org/es/docs/Web/CSS/repeat\(\)](https://developer.mozilla.org/es/docs/Web/CSS/repeat())



**GUSTAVO RODRIGUEZ**

FULL STACK DEVELOPER  
SOLCRE



[gustavgueez](#)



[gustavgueez](#)



[gustavgueez](#)