

# Display Grid

# Grid

Sistema de diseño basado en cuadrícula, enfocado en facilitar la construcción de páginas web.

Al igual que las tablas(<table>), Grid layout permite distribuir y alinear elementos en columnas y filas, pero de una forma mucho más simple e intuitiva.

# Grid

Para comenzar a utilizar Grid, es necesario seleccionar la etiqueta que hará de cuadrícula.

Para este ejemplo se utilizará un `<div>`.

Observar que para utilizar una cuadrícula, es necesario indicar un **`display: grid;`** en la etiqueta que funcionará de contenedor.

```
.modo-grid {  
  display: grid;  
}
```

```
<div class="modo-grid"></div>
```

# Filas y Columnas

Para definir la distribución es necesario indicar cuántas filas y columnas tendrá.

Propiedades a utilizar:

1. `grid-template-columns`
2. `grid-template-rows`

```
.modo-grid {  
  display: grid;  
  grid-template-columns: 200px 100px 150px;  
  grid-template-rows: 100px 150px 200px;  
}
```

200px para la primera columna.

100px para la segunda columna.

150px para la tercera columna.

```
.modo-grid {  
  display: grid;  
  grid-template-columns: 200px 100px 150px;  
  grid-template-rows: 100px 150px 200px;  
}
```

100px para la primera fila.

150px para la segunda fila.

200px para la tercera fila.

# Escriba la siguiente estructura

```
.modo-grid {  
  display: grid;  
  grid-template-columns: 100px 100px 100px;  
  grid-template-rows: 100px 100px;  
}  
  
.modo-grid > div {  
  border: 1px solid white;  
  background-color: tomato;  
}
```

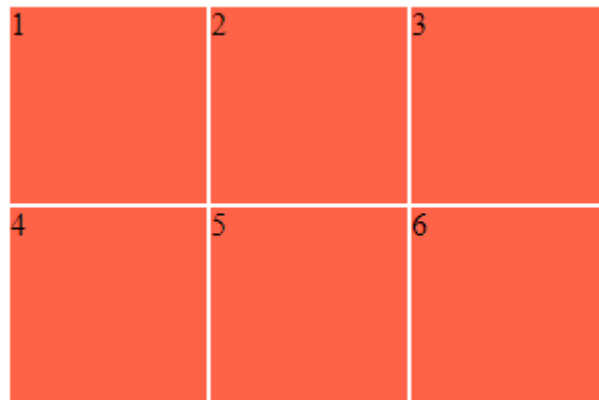
```
<div class="modo-grid">  
  <div>1</div>  
  <div>2</div>  
  <div>3</div>  
  <div>4</div>  
  <div>5</div>  
  <div>6</div>  
</div>
```

¿Qué resultado obtuvo?

¿Qué sucede si quita o agrega más <div> en el interior del contenedor?

## Respuesta

1. Se crea una cuadrícula de 3x2.
2. Cada elemento se posiciona en una celda automáticamente.
3. Si se agrega un elemento excediendo la cantidad de celdas disponibles, las nuevas filas utilizarán el valor de altura respecto al tamaño del contenido.



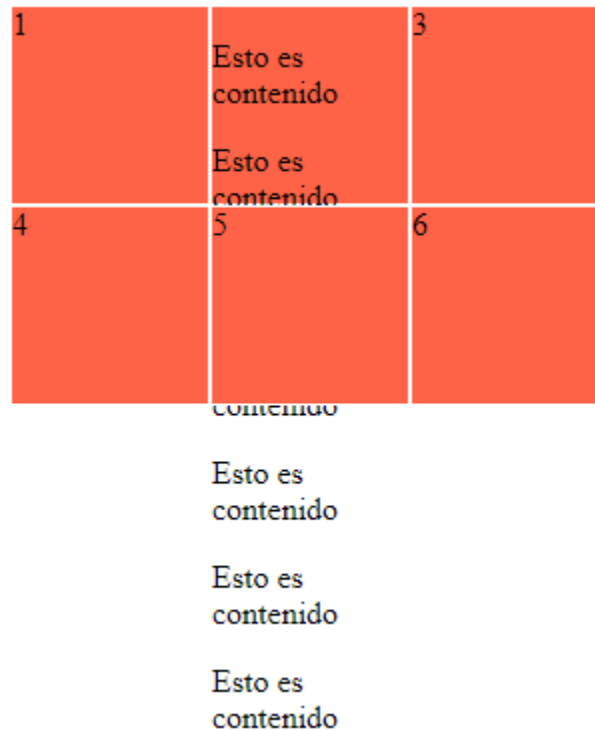
¿Qué sucedería si agrega más contenido a la capacidad de una celda de ancho y alto fijo?

```
<div class="modo-grid">  
  <div>1</div>  
  <div>  
    <p>Esto es contenido</p>  
    <p>Esto es contenido</p>  
    <p>Esto es contenido</p>  
    <p>Esto es contenido</p>  
    <p>Esto es contenido</p>  
    <p>Esto es contenido</p>  
    <p>Esto es contenido</p>  
  </div>  
  <div>3</div>  
  <div>4</div>  
  <div>5</div>  
  <div>6</div>  
</div>
```



# Respuesta

El contenido se desplazará hacia abajo, pero por detrás de las filas inferiores.



# Flexibilizando columnas y filas

Las propiedades `grid-template-columns` y `grid-template-rows` también aceptan otros tipos de valores como porcentajes.

Ejemplo:

`grid-template-columns: 20% 30% 50%;`

Utilizando la estructura de ejemplo, cambie los valores a porcentajes indicados anteriormente.

```
<div class="modo-grid">
  <div>1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
  <div>5</div>
  <div>6</div>
</div>
```

Resultado:

1	2	3
4	5	6

# Ejercicio

Diseñe una tabla con las siguientes propiedades:

1. La primera columna tiene un ancho del 10%.
2. La segunda columna tiene un ancho de 200px.
3. La tercera columna tiene un ancho de 20%.
4. La cuarta columna tiene un ancho de 30%.
5. La primera fila tiene un alto de 50px.
6. La segunda fila tiene un alto de 100px.
7. La tercera fila tiene un alto de 200px.
8. Llene cada celda con un cuadro de color y borde blanco.

# Resultado

```
.modo-grid {  
  display: grid;  
  grid-template-columns: 10% 200px 20% 30%;  
  grid-template-rows: 50px 100px 200px;  
}  
  
.modo-grid > div {  
  border: 1px solid white;  
  background-color: tomato;  
}
```

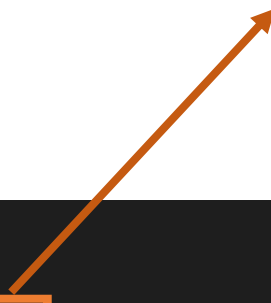


# Utilizando el ejercicio anterior

Cambie el 10% de la primera columna por un 50%.

¿Qué conclusión puede obtener respecto al espacio calculado en porcentaje?

**Cambiar por 50%**



```
.modo-grid {  
  display: grid;  
  grid-template-columns: 10% 200px 20% 30%;  
  grid-template-rows: 50px 100px 200px;  
}  
  
.modo-grid > div {  
  border: 1px solid white;  
  background-color: tomato;  
}
```

# Respuesta

Los porcentajes se calculan respecto a la dimensión de su contenedor padre, **NO** respecto al ancho/alto **que sobre**.

Por lo tanto, si el ancho del contenedor padre es de 1000px.

¿Cuál sería el ancho de cada columna?

¿Cuánto sería el ancho total?

```
.modo-grid {  
  display: grid;  
  grid-template-columns: 50% 200px 20% 30%;  
  grid-template-rows: 50px 100px 200px;  
}
```

# Respuesta

Columna 1 :  $1000 * (50/100) = 500\text{px}$ .

Columna 2 :  $200\text{px}$ .

Columna 3 :  $1000 * (20/100) = 200\text{px}$ .

Columna 4:  $1000 * (30/100) = 300\text{px}$ .

Ancho total:  $1200\text{px}$ .

Suponiendo que el contenedor padre ocupa todo el ancho de la pantalla, al excederse  $200\text{px}$  aparecerá el temido scroll horizontal.

**¿Cómo se podrá calcular los porcentajes respecto al espacio disponible para evitar este problema?**



## Unidades de tipo fr (fracción)

Las unidades fraccionadas trabajan respecto al espacio que aún no ha sido ocupado.

Observemos el siguiente código:

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

Esto quiere decir, que hay 5 columnas y cada columna pesa lo mismo.

Por lo que 1fr es igual a  $1/5 = 20\%$ .

Sería lo mismo que indicar:

```
grid-template-columns: 20% 20% 20% 20% 20%;
```

# Unidad fr

¿Cuál sería la proporción?

```
grid-template-columns: 1fr 2fr 1fr 3fr 1fr;
```

Primero se suman todos los fr.

Total: 8fr.

Esto quiere decir que el espacio disponible se divide en 8 unidades.

$$1\text{fr} = 1\text{f}/8\text{fr} = 12.5\%$$

Por lo tanto si esto se escribiera en porcentaje sería:

```
grid-template-columns: 12.5% 25% 12.5% 37.5% 12.5%;
```

# Ejercicio

Utilizando el ejercicio anterior, corrija el problema de desborde (Los porcentajes deben calcularse por el espacio disponible).

Diseñe una tabla con las siguientes propiedades:

1. La primera columna tiene un ancho del **50%**.
2. La segunda columna tiene un ancho de 200px.
3. La tercera columna tiene un ancho de **20%**.
4. La cuarta columna tiene un ancho de **30%**.
5. La primera fila tiene un alto de 50px.
6. La segunda fila tiene un alto de 100px.
7. La tercera fila tiene un alto de 200px.
8. Llene cada celda con un cuadro de color y borde blanco.

# Solución

Como tenemos 3 columnas porcentuales.

50% 20% 30% = 100%.

Buscamos un divisor común y simplificamos:

50/10 20/10 30/10.

5fr 2fr 3fr

5\*10 = 50%

2\*10 = 20%

3\*10 = 30%

Por lo tanto:

```
.modo-grid {  
  display: grid;  
  grid-template-columns: 5fr 200px 2fr 3fr;  
  grid-template-rows: 50px 100px 200px;  
}
```

# Ejercicio

Corrija el problema de desborde (Los porcentajes deben calcularse por el espacio disponible).

Diseñe una tabla con las siguientes propiedades:

1. La primera columna tiene un ancho del **40%**.
2. La segunda columna tiene un ancho de 200px.
3. La tercera columna tiene un ancho de **25%**.
4. La cuarta columna tiene un ancho de 50px.
5. La quinta columna tiene un ancho de **15%**.
6. La sexta columna tiene un ancho de **20%**.
7. La primera y segunda fila tiene un alto de 50px.
8. Llene cada celda con un cuadro de color y borde blanco.

# Solución

40% 25% 15% 20%

Buscamos un divisor común: 5.

40/5 25/5 15/5 20/5

8fr 5fr 3fr 4fr

Por lo tanto:

```
.modo-grid {  
  display: grid;  
  grid-template-columns: 8fr 200px 5fr 50px 3fr 4fr;  
  grid-template-rows: 50px 50px;  
}
```

## Valor **auto**

Ocupa el espacio necesario correspondiente a su contenido.

Los “fr” incluirán en el calculo las columnas “auto”, para poder obtener el espacio disponible.

Por lo tanto, si una columna auto ocupa una gran cantidad de espacio, las columnas con “fr” serán mucho más pequeñas.

Pd: El valor auto se puede ocupar tanto en filas y columnas.

```
.modo-grid {  
  display: grid;  
  grid-template-columns: 8fr auto 5fr auto 3fr 4fr;  
  grid-template-rows: 50px 50px;  
}
```

# Función - repeat(cant. veces, valor)

La función repeat puede ser utilizada cuando se repite un valor continuamente en una fila o columna.

Ejemplo:

```
.modo-grid {  
  display: grid;  
  grid-template-columns: 25% 25% 25% 25%;  
  grid-template-rows: 100px 100px;  
}
```

```
.modo-grid {  
  display: grid;  
  grid-template-columns: repeat(4, 25%);  
  grid-template-rows: repeat(2, 100px);  
}
```



# Ejercicio

Utilice la función repeat para reducir el número de valores.

```
.modo-grid {  
  display: grid;  
  grid-template-columns: 10% 10% 10% 10% 10% 1fr 1fr 1fr 1fr 1fr 1fr 1fr 1fr;  
  grid-template-rows: 100px 100px 100px 100px 100px 250px;  
}
```

Forma: repeat(Cant. Veces, valor)

# Solución

```
.modo-grid {  
  display: grid;  
  grid-template-columns: repeat(5, 10%) repeat(8, 1fr);  
  grid-template-rows: repeat(5, 100px) 250px;  
}
```

# Función minmax(min. valor, max. valor)

La función minmax() en CSS define un rango de tamaño mayor o igual que min y menor o igual que max.

Se emplea para indicar los tamaños posibles que puede tomar cada columna o fila.

Ejemplos de valores posibles:

minmax(200px, 1fr)

minmax(30%, 300px)

minmax(400px, 50%)

minmax(200px, auto)

```
.modo-grid {  
  display: grid;  
  grid-template-columns: minmax(100px, 500px) minmax(100px, 80%);  
  grid-template-rows: minmax(50px, 100px) minmax(200px, auto);  
}
```

# Ejercicio

Construya un Grid de 3x2

1. Columna 1 tiene un ancho de 100px.
2. Columna 2 tiene un ancho del 80% del espacio sobrante.
  - a) La columna máximo se puede encoger hasta 300px.
3. Columna 3 tiene un ancho del 20% del espacio sobrante.
  - a) La columna máximo se puede encoger hasta 100px.
4. Fila 1.
  - a) La altura mínima es de 150px
  - b) La altura máxima es de 250px.
5. Fila2.
  - a) La altura mínima es de 400px.
  - b) La altura máximo es relativa al contenido.

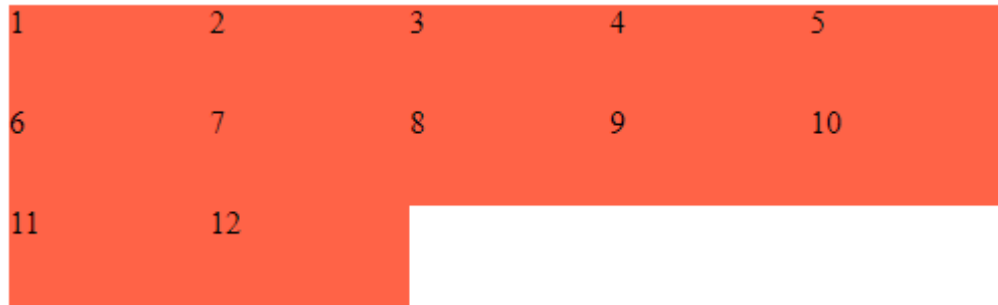
# Solución

```
.modo-grid {  
  display: grid;  
  grid-template-columns: 100px minmax(300px, 4fr) minmax(100px, 1fr);  
  grid-template-rows: minmax(150px, 250px) minmax(400px, auto);  
}
```

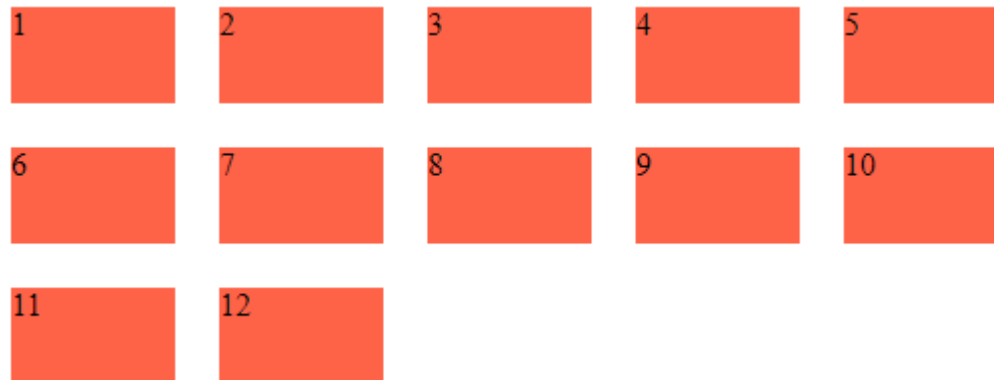
# Propiedad gap.

Separa las celdas entre ellas por un margen.

La propiedad gap se debe asignar al contenedor.

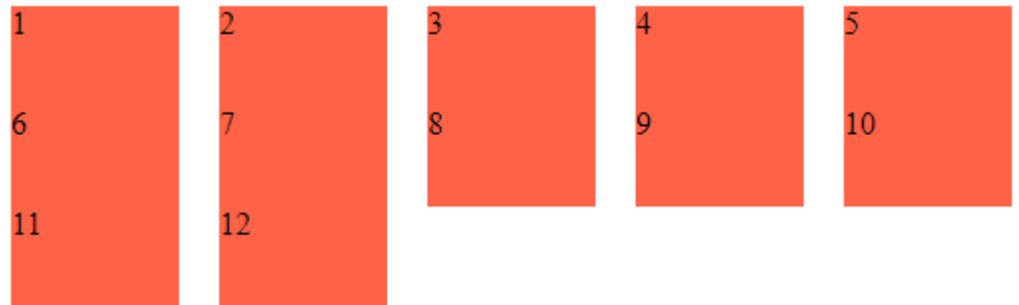


**gap: 20px;**

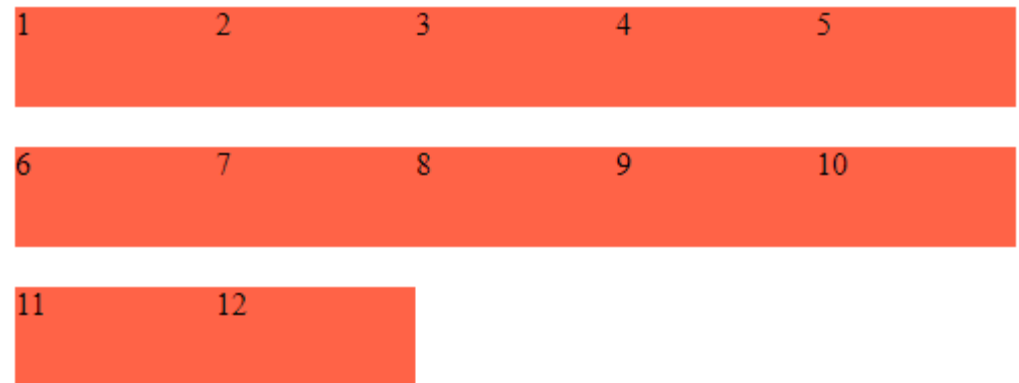


# Propiedad gap Fila/Columna

**column-gap: 20px;**

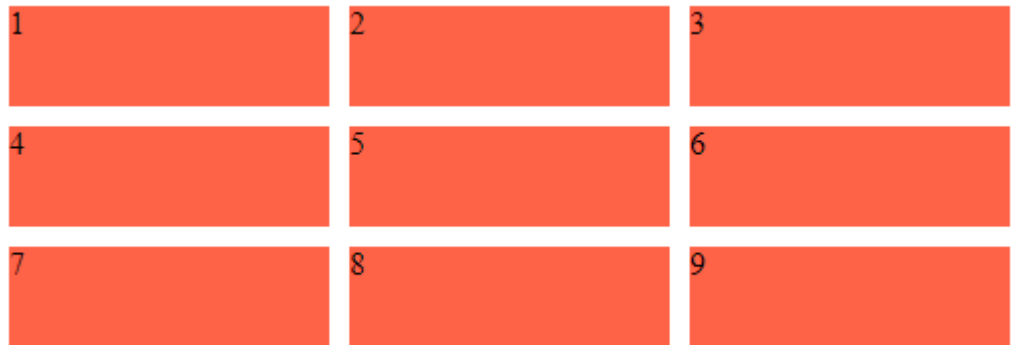


**row-gap: 20px;**



# Ejercicio

Construya el siguiente Grid.



```
.modo-grid {  
  display: grid;  
  width: 500px;  
  grid-template-columns: repeat(3, 1fr);  
  grid-template-rows: repeat(3, 50px);  
  gap: 10px;  
}  
  
.modo-grid > div {  
  background-color: tomato;  
}
```

```
<div class="modo-grid">  
  <div>1</div>  
  <div>2</div>  
  <div>3</div>  
  <div>4</div>  
  <div>5</div>  
  <div>6</div>  
  <div>7</div>  
  <div>8</div>  
  <div>9</div>  
</div>
```



## Posicionando Elementos –

Propiedades grid-column/grid-row

Gracias a estas dos propiedades es posible indicar qué celdas serán ocupadas.

Para entender como funciona ,es importante saber que una fila y columna inicia en la posición 1.

## Seleccionando celdas

Propiedades	Descripción
grid-column-start	Indica donde inicia una columna
grid-column-end	Indica donde termina una columna
grid-row-start	Indica donde inicia una fila
grid-row-end	Indica donde termina una fila

**Observación:** Estas propiedades deben ser aplicadas en una celda.

# Estructura valores

	1	2	3	4
1	1	2	3	
2	4	5	6	
3	7	8	9	
4				

Si deseo ocupar la celda 4 y 5 a la vez (Como única celda).  
¿Qué valores tendrá que tener las propiedades de grid?

# Buscando valores para columnas

	1	2	3	4
1	1	2	3	
2	4	5	6	
3	7	8	9	
4				

Para ocupar la celda 4 y 5 los valores necesarios en columna serán:

**grid-column-start: 1;**

**grid-column-end: 3;**

## Buscando valores para filas

	1	2	3	4
1	1	2	3	
2	4	5	6	
3	7	8	9	
4				

Para ocupar la celda 4 y 5 los valores necesarios en fila serán:

**grid-row-start: 2;**

**grid-row-end: 3;**

Por último, se agrega las propiedades a una clase y se asocia a una etiqueta.

**Observación:** Recordar borrar los divs  
excedentes (Ya que se está ocupando una  
celda extra)

```
#seccion_verde{
    grid-column-start: 1;
    grid-column-end: 3;
    grid-row-start: 2;
    grid-row-end: 3;

    background-color: #greenyellow;
}
```


[illegible]

# Resultado



# Reduciendo líneas

Para indicar los rangos de posiciones se requiere de “start” y “end”, pero también se puede escribir de forma abreviada.

Observar que se indican los valores separados por un “/” y las palabras “start” y “end” ya no se escriben.

```
#seccion_verde {  
    grid-column-start: 1;  
    grid-column-end: 3;  
    grid-row-start: 2;  
    grid-row-end: 3;  
  
    background-color:  greenyellow;  
}
```

## Abreviación

```
#seccion_verde {  
    grid-column: 1/3;  
    grid-row: 2/3;  
  
    background-color:  greenyellow;  
}
```



# Diseñe un Grid de 3x3.

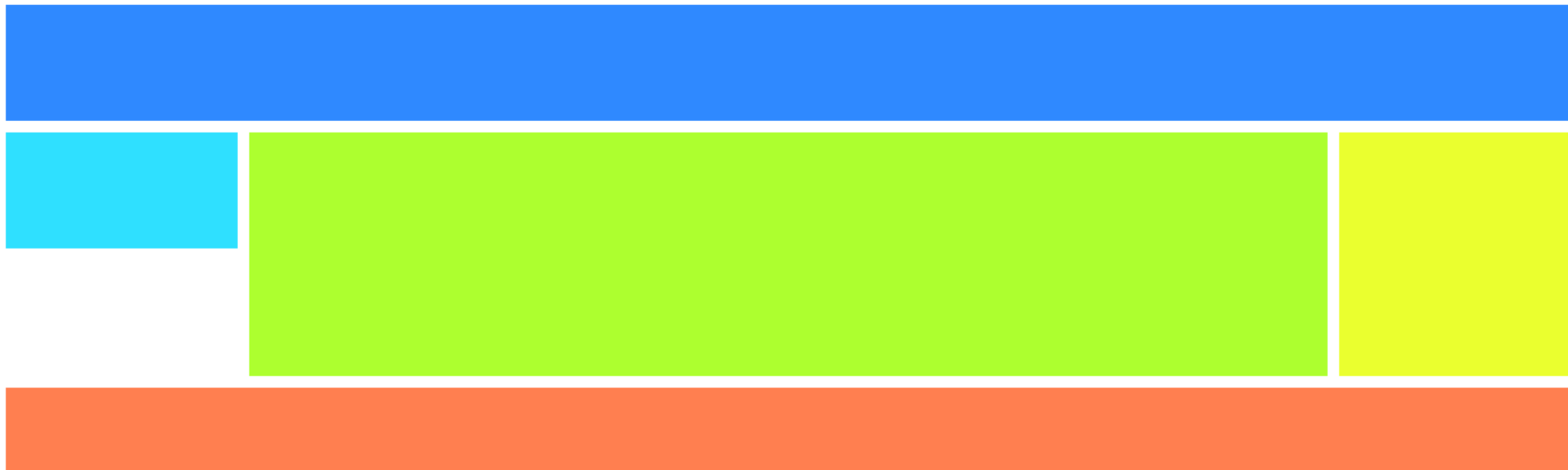
Utilizando la base del código anterior, implemente el siguiente diseño.



# Diseñe un Grid que represente una estructura base de página web.


1. Nav tiene un ancho de 200px
2. Aside tiene un ancho de 200px
3. Section tiene un ancho del 100% respecto al espacio sobrante.
4. La altura mínima de cada etiqueta es de 100px.
5. La separación entre filas y columnas es de 10px.


```
<div class="modo-grid">  
  <header></header>  
  <nav></nav>  
  <section></section>  
  <aside></aside>  
  <footer></footer>  
</div>
```





# Solución


```
.modo-grid {  
  display: grid;  
  grid-template-columns: 200px 1fr 200px;  
  grid-template-rows: repeat(4, minmax(100px, auto));  
  gap: 10px;  
}
```

```
header {  
  grid-column: 1/4;  
  grid-row: 1/2;  
  background-color:  rgb(47, 137, 255);  
}
```

```
nav {  
  grid-column: 1/2;  
  grid-row: 2/3;  
  background-color:  rgb(47, 224, 255);  
}
```

```
section {  
  grid-column: 2/3;  
  grid-row: 2/4;  
  background-color:  greenyellow;  
}
```

```
aside {  
  grid-column: 3/4;  
  grid-row: 2/4;  
  background-color:  rgb(234, 255, 47);  
}
```

```
footer {  
  grid-column: 1/4;  
  grid-row: 4/5;  
  background-color:  coral;  
}
```