

Display

Grid

Propiedad grid-area

La propiedad grid-area especifica el tamaño y la ubicación de un elemento en la cuadrícula.

Columnas y filas indicadas en una línea.

grid-column: 1 / 7;

grid-row : 5 / 8

Resumido en:

grid-area: 5 / 1 / 8 / 7;

grid-area: grid-row-start / grid-column-start / grid-row-end /
grid-column-end

Propiedad grid-template-areas

Uno de los puntos fuertes de **grid-template-areas** es indicar posiciones de acuerdo a nombres.

Podemos nombrar las celdas y de acuerdo a eso enlazar una estructura.

De esta forma ya no será necesario calcular por números las posiciones de columna/fila.


Ejemplo


A cada celda se le asignará un nombre, si un nombre se repite en otra celda, se entenderá que son la misma, por lo tanto se fusionarán.






Por último enlazamos cada bloque con su nombre.

Observar que el nombre se escribe **sin comillas**.

```
.verde{
  grid-area: Verde;
  background-color:  #ADFF2F;
}

.azul{
  grid-area: Azul;
  background-color:  #2F89FF;
}

.amarillo{
  grid-area: Amarillo;
  background-color:  #E AFF2F;
}

.coral{
  grid-area:  Coral;
  background-color:  #FF6347;
}
```

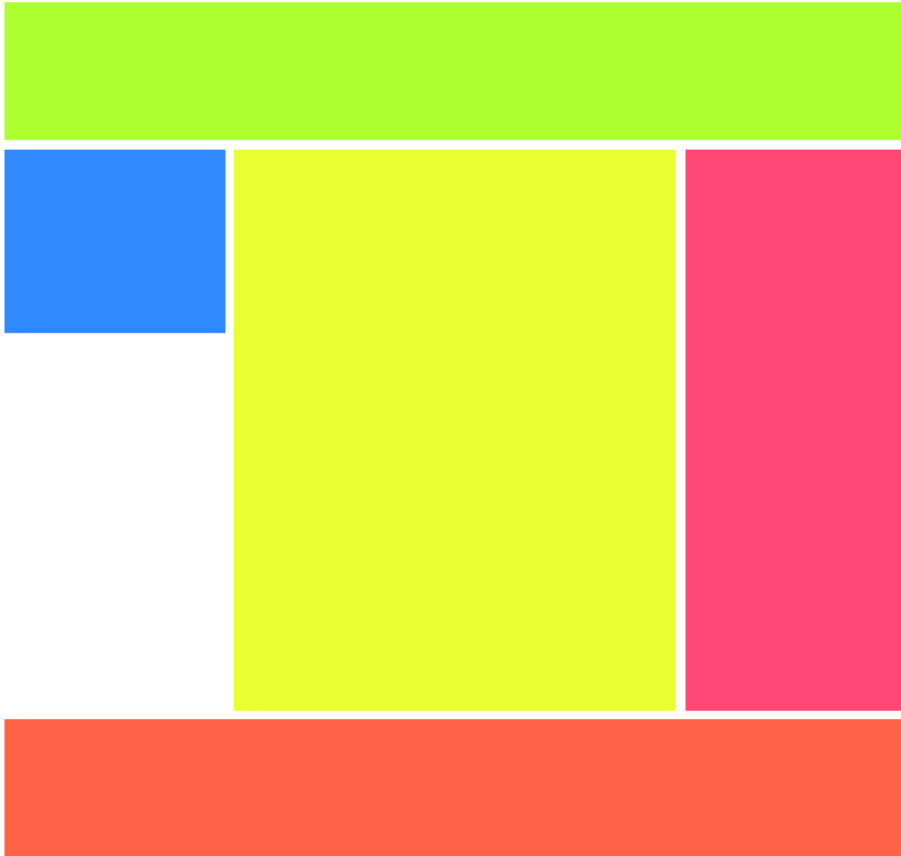
```
<div class="modo-grid">
  <div class="coral"></div>
  <div class="amarillo"></div>
  <div class="azul"></div>
  <div class="verde"></div>
</div>
```

Ejercicio

Utilizando la distribución por nombre (**grid-template-areas**), diseñe la siguiente página.

Reglas de construcción:

1. La página tiene un ancho de 980px.
2. La página se encuentra centrada.
3. La sección ocupa el doble de ancho que el aside y nav.
4. La altura mínima del Header es de 150px.
5. La altura mínima del Nav es de 200px.
6. La altura mínima del Section es de 600px.
7. La altura mínima del Nav es de 600px.
8. La altura mínima del Footer es de 150px.
9. La separación entre columnas/filas es de 10px.



Solución

```
.modo-grid {
  display: grid;
  grid-template-columns: 1fr 2fr 1fr;
  grid-template-rows:
    minmax(150px, auto) minmax(200px, auto)
    minmax(400px, auto) minmax(150px, auto);

  grid-template-areas:
    "Header      Header      Header"
    "Nav         Section     Aside"
    "Nada        Section     Aside"
    "Footer      Footer      Footer";

  gap: 10px;
  margin: 0 auto;
  width: 980px;
}
```

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

```
header {
  grid-area: Header;
  background-color: #adff2f;
}

nav {
  grid-area: Nav;
  background-color: #2f89ff;
}

section {
  grid-area: Section;
  background-color: #eaff2f;
}

aside {
  grid-area: Aside;
  background-color: #ff4775;
}

footer {
  grid-area: Footer;
  background-color: #ff6347;
}
```


Alineación en Horizontal - General

Para Grid, existe una propiedad general para indicar como se tiene que alinear las celdas de forma horizontal. Dicha propiedad debe ser incluida en el contenedor principal.

Propiedad:

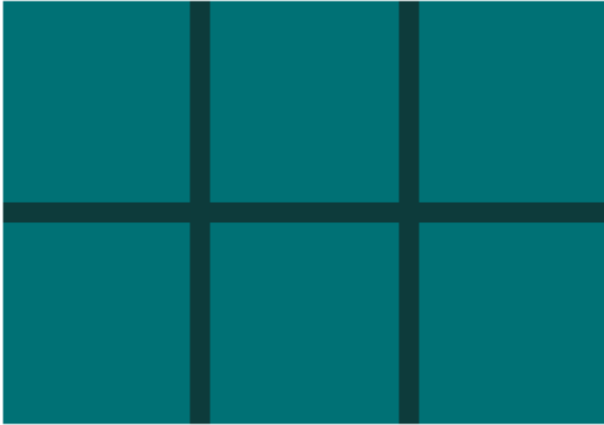
justify-items

Utilice los siguientes valores para observa el comportamiento:

1. **stretch** – Valor por defecto, estira el contenido (width) en un 100%.
2. **start/left** – Inicia al comienzo de la celda (Lado Izquierdo).
3. **end/right** – Inicial al final de la celda (Lado Derecho).
4. **center** – Centra el contenido horizontalmente en la celda.

Soluciones

stretch – Por defecto.



start/left



end/right



center



Alineación en Vertical - General

Para Grid, existe una propiedad general para indicar como se tiene que alinear las celdas en forma vertical. Dicha propiedad debe ser incluida en el contenedor principal.

Propiedad:

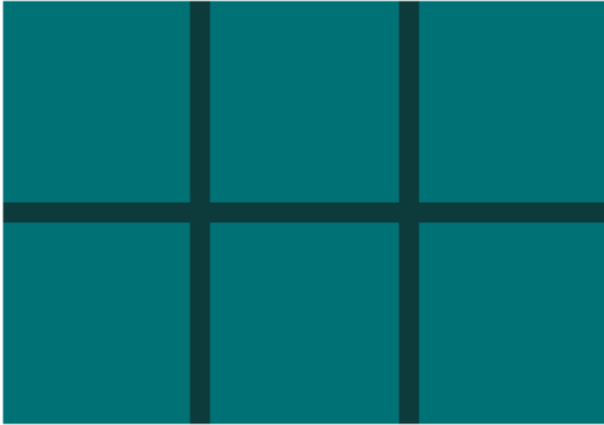
align-items

Utilice los siguientes valores para observa el comportamiento:

1. **stretch** – Valor por defecto, estira el contenido (Height) en un 100%.
2. **baseline/start** – Inicia en la parte superior de la celda.
3. **end** – Inicial en la parte inferior de la celda.
4. **center** – Centra el contenido verticalmente en la celda.

Soluciones

stretch – Por defecto.



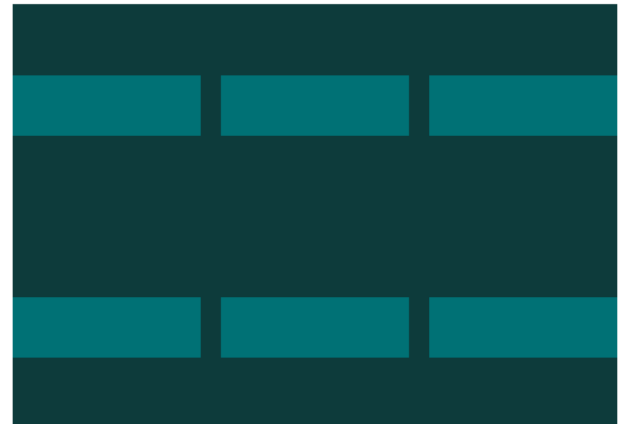
end



start/baseline



center



Recordar que align-items y justify-items se pueden combinar para obtener un mejor resultado.

Por ejemplo, centrar en ambas direcciones (Horizontal/Vertical).

```
.modo-grid {  
  display: grid;  
  grid-template-columns: 1fr 1fr 1fr;  
  grid-template-rows: repeat(2, 100px);  
  background-color: #0d3737;  
  gap: 10px;  
  width: 300px;  
  
  align-items: center;  
  justify-items: center;  
}
```



Observación: align-items y justify-items **solamente alinean la celda, no el contenido interno** que puede almacenar.

Alineación de celdas - Específico

Hasta el momento hemos alineado de forma general, pero también es posible indicar por celda como debe alinear. En el caso que hubiera una alineación general, esta será anulada solamente para la celda que aplique alguna de estas propiedades.

Propiedades:

justify-self

align-self

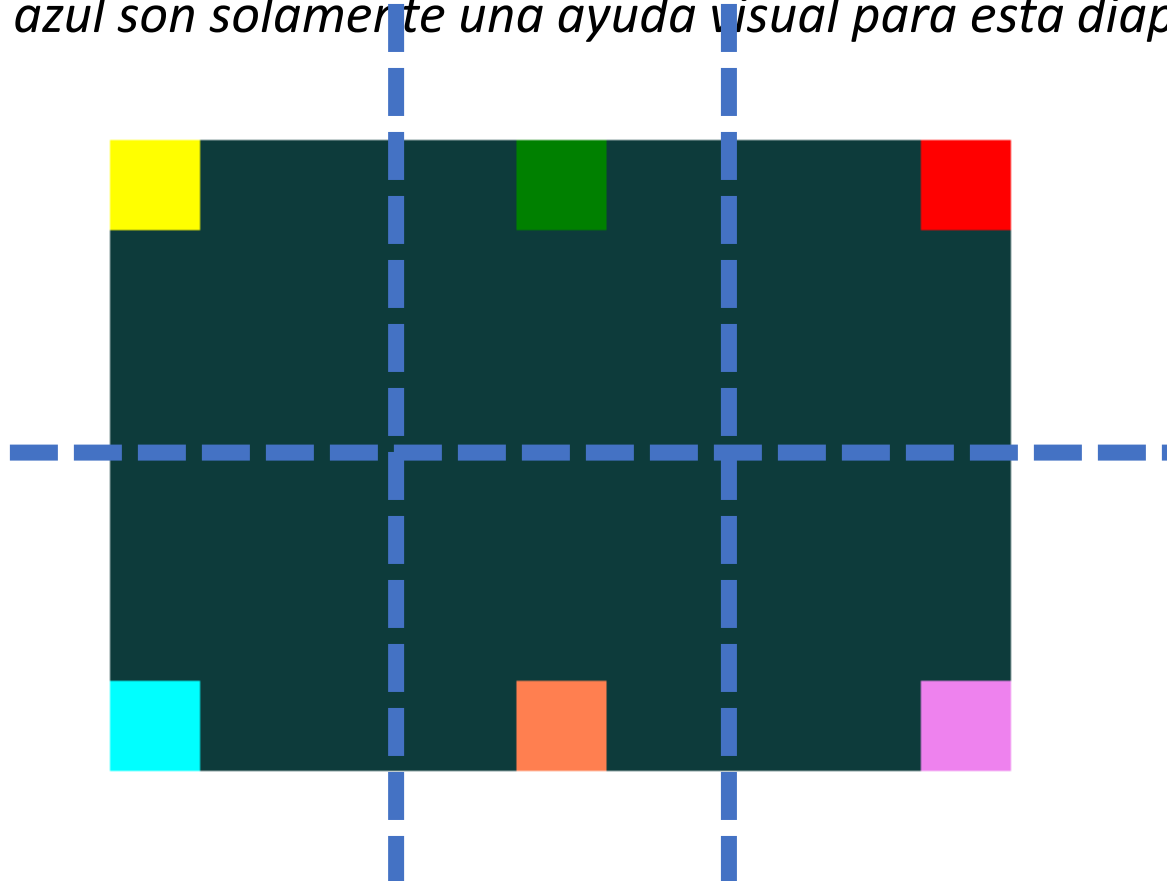
Ambas propiedades funcionan iguales que su versión general.

¡Solamente recuerda incluir estas propiedades en alguna clase que esté asociada a una celda!

Ejercicio





Utilizando el código base ejercicio anterior, intente diseñar la siguiente tabla.



Las líneas azul son solamente una ayuda visual para esta diapositiva.



■		■		■
■		■		■

Solución

```
.yellow {  
  background-color:  yellow;  
  align-self: start;  
  justify-self: start;  
}  
  
.green {  
  background-color:  green;  
  align-self: start;  
  justify-self: center;  
}  
  
.red {  
  background-color:  red;  
  align-self: start;  
  justify-self: end;  
}  
  
.cyan {  
  background-color:  cyan;  
  align-self: end;  
  justify-self: start;  
}
```

```
.coral {  
  background-color:  coral;  
  align-self: end;  
  justify-self: center;  
}  
  
.violet {  
  background-color:  violet;  
  align-self: end;  
  justify-self: end;  
}
```

```
<section class="modo-grid">  
  <div class="yellow"></div>  
  <div class="green"></div>  
  <div class="red"></div>  
  <div class="cyan"></div>  
  <div class="coral"></div>  
  <div class="violet"></div>  
</section>
```

Bonus

Cuando un valor se repita en ambas direcciones (Horizontal/Vertical) se puede utilizar las propiedades:

- `place-items` - Asigna el mismo valor a `justify-items` y `align-items`.
- `place-self` – Asigna el mismo valor a `justify-self` y `align-self`.

Ejemplo:

```
place-items: center;
```

El mismo que escribir

```
justify-items: center;
```

```
align-items: center;
```

Observación: Todos los navegadores modernos aceptan `place-items/place-self`, exceptuando a Edge.

Recordar siempre revisar compatibilidad en

<https://caniuse.com/>

Recordar

align-items, justify-items,
align-self y justify-self
**solamente alinean la
celda, no el contenido
interno.**

Superposición de elementos

En Grid, es posible superponer elementos, solamente indicando un rango que ya se encuentre ocupado por otros elementos, para que se produzca una superposición.

Escriba el siguiente código y observe el comportamiento.

```
<section class="modo-grid">
  <div class="seccion-1">
    1
  </div>
  <div class="section-2">
    2
  </div>
</section>
```

```
.modo-grid {
  display: grid;
  grid-template-columns: 1fr 1fr 1fr;
  grid-template-rows: 100px;
  width: 300px;
}

div {
  text-align: center;
  line-height: 95px;
  font-weight: bolder;
}

.seccion-1 {
  background-color: yellow;
  grid-column: 1 / 3;
  grid-row: 1/2;
}

.section-2 {
  background-color: lightgreen;
  grid-column: 2 / 4;
  grid-row: 1/2;
}
```

Resultado

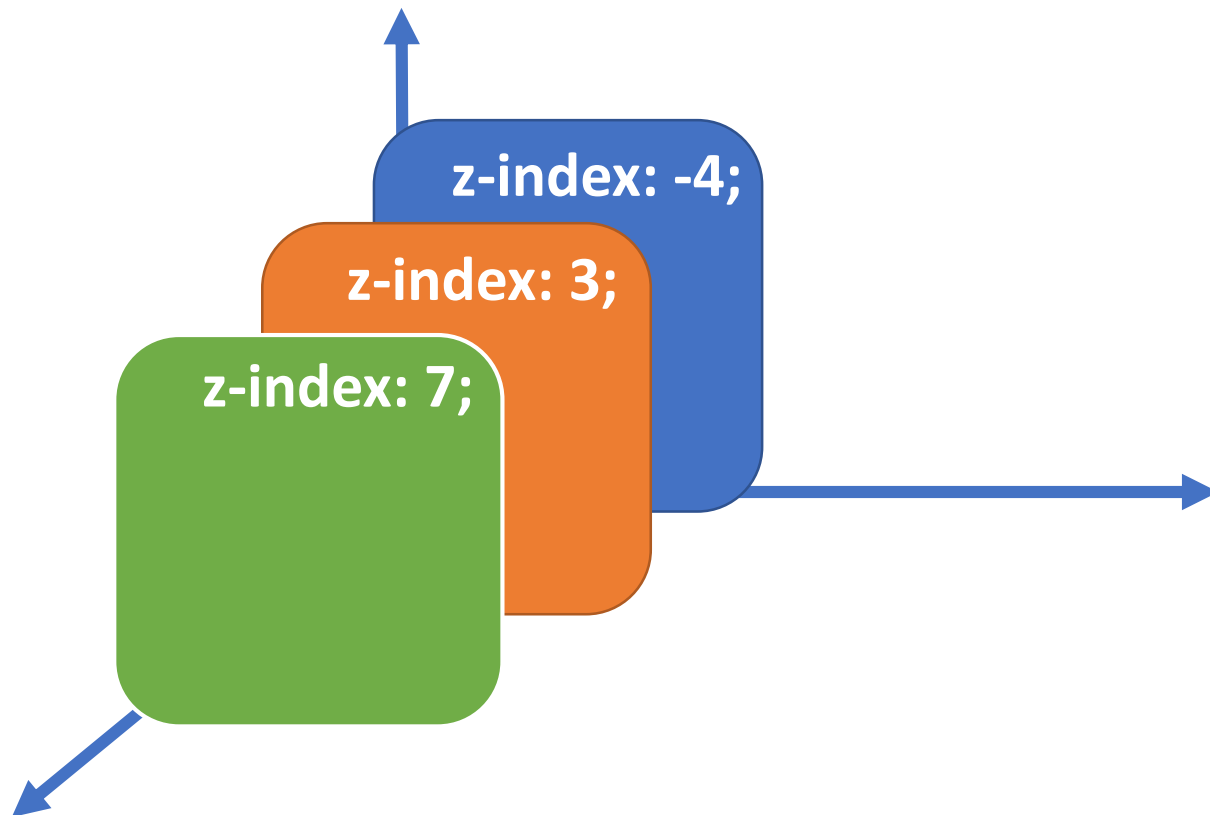
Como se puede apreciar, la celda 2 se está superponiendo con la celda 1.



Propiedad z-index

Cuando varios elementos se superponen, z-index determina cuales van sobre otros (Imaginar una pila de hojas). Por lo general un elemento con mayor z-index se posicionará por delante de uno menor.

Por defecto todos los elementos inician con un z-index igual a cero.



Resultado

Utilizando el ejercicio anterior.

¿Cómo la celda uno podría ir sobre la celda dos?




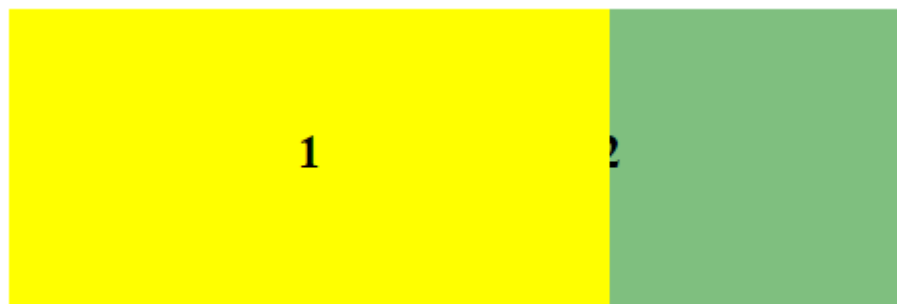
Observación: z-index debe ir en el bloque de CSS correspondiente a la celda.

El valor para z-index es solamente un número sin unidad de medida.

Solución

Agregar la propiedad z-index en el bloque CSS de la celda 1.

```
.seccion-1 {  
  background-color:  rgb(255, 255, 0);  
  grid-column: 1 / 3;  
  grid-row: 1/2;  
  z-index: 1;  
}
```



Ejercicio

Desarrollar el ejercicio de Grid encontrado en la página del curso

Ejercicio - Grid - Layer

Password: **grid++**