Grid y flex con frameworks y CSS puro

Conceptos Básicos

Grid y Flexbox son propiedades dentro de CSS que nos ayudan a ubicar nuestros elementos dentro del DOM. Mientras Grid basa su sistema en un layout de dos dimensiones (rows y columns) basado en cuadrículas, Flexbox se enfoca en trabajar en una sola dimensión (o rows o columns) y te ayuda a distribuir los elementos dentro del padre.

Aquí les dejo la documentación de Flexbox y Grid.

La mejor forma de trabajar con ellos es combinarlos para tener lo mejor de los dos mundos.

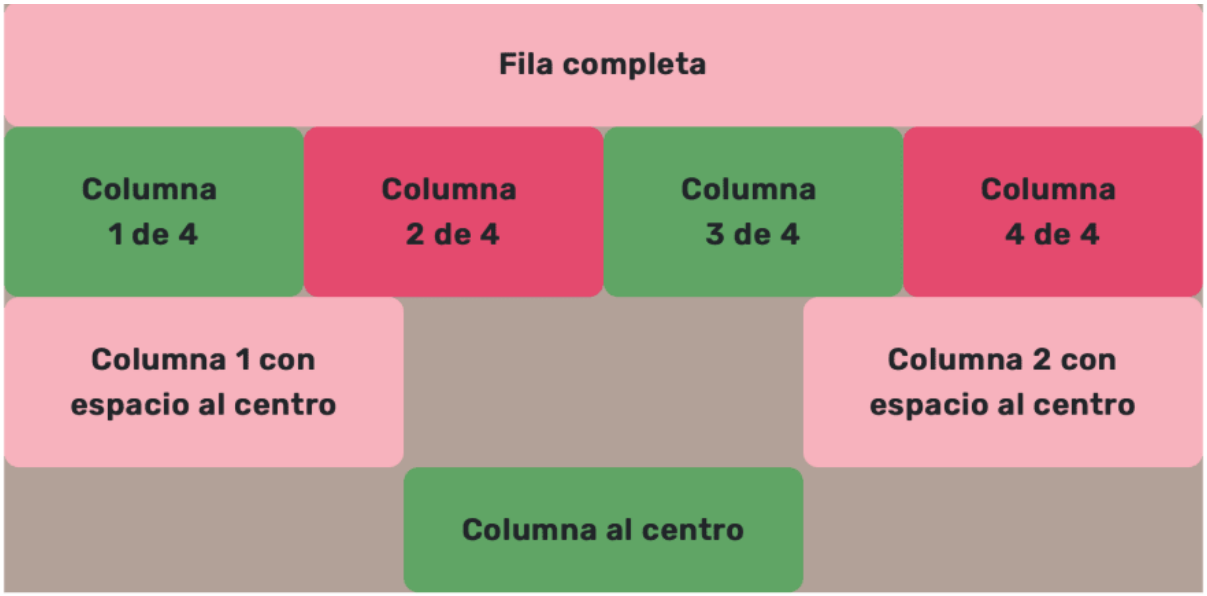
Ahora, dos de los Frameworks más populares actualmente son Bootstrap en su versión 5 y Tailwind

Ambos frameworks cuentan con clases que ya implementan tanto Grid como Flexbox, algunas son iguales y otras varían en cosas muy pequeñas.

Veamos nuestro primer ejemplo:

Grid

Para implementar grid de diferentes formas, con *n* cantidad de columnas y alineaciones, creé este ejemplo que engloba las posibles formas en que lo podemos usar:



Para implementar este layout veamos cómo se hace primero con CSS puro:

Nuestro HTML:

```
<div class="container">
    <div class="grid grid-4">
        <div class="span-4">
            Fila completa
        </div>
        <div class="">
            Columna 1 de 4
        </div>
        <div class="">
            Columna 2 de 4
        </div>
        <div class="">
            Columna 3 de 4
        </div>
        <div class="">
            Columna 4 de 4
        </div>
    </div>
    <div class="grid grid-template-3">
        <div class="col1-1"> columna 1 con espacio al centro</div>
        <div class="col1-3">columna 2 con espacio al centro</div>
        <div class="col2-2">columna al centro</div>
    </div>
</div>

Como no usamos Frameworks, necesitamos nuestro CSS:
/* Inicializamos el grid*/
.grid {
    display: grid;
}

/* En cuántas columnas se dividirá nuestro grid*/
.grid-4{
    grid-template-columns: repeat(4,minmax(0,1fr));
}
/* Cuantos espacios (en columnas )tomará nuestro div en el grid */
.span-4 {
    grid-column: span 4;
}

/* Otra forma de definir grid templates es especificando el nombre del div */
.grid-template-3 {
     grid-template-areas:
        "col1-1 . col1-3"
        ". col2-2 .";
}
/* Das el nombre a los divs */
.col1-1 {
    grid-area: col1-1;
}
.col2-2 {
    grid-area: col2-2;
}
```

```
.col1-3 {
    grid-area: col1-3;
}
```

Cómo ven usamos simplemente Grid.

Ahora, si queremos tener exactamente el mismo resultado tanto en Bootstrap como en Tailwind, afortunadamente ya cuentan con clases para eso y no tenemos que tocar el css para nada.

¡Dios bendiga a los frameworks 🙏!

Grid en Bootstrap

```
<div class="container mt-5">
    <div class="row">
        <div class="col">
            Fila completa
        </div>
    </div>

    <div class="row">
        <div class="col">
            Columna 1 de 4
        </div>
        <div class="col">
            Columna 2 de 4
        </div>
        <div class="col">
            Columna 3 de 4
        </div>
        <div class="col">
            Columna 4 de 4
        </div>
    </div>

    <div class="row">
        <div class="col-md-4"> Columna 1 con espacio al centro</div>
        <div class="col-md-4 offset-md-4">Columna 2 con espacio al centro</div>
    </div>
    <div class="row">
        <div class="col-md-4 offset-md-4">Columna al centro</div>
    </div>
</div>
```

Grid en Tailwind

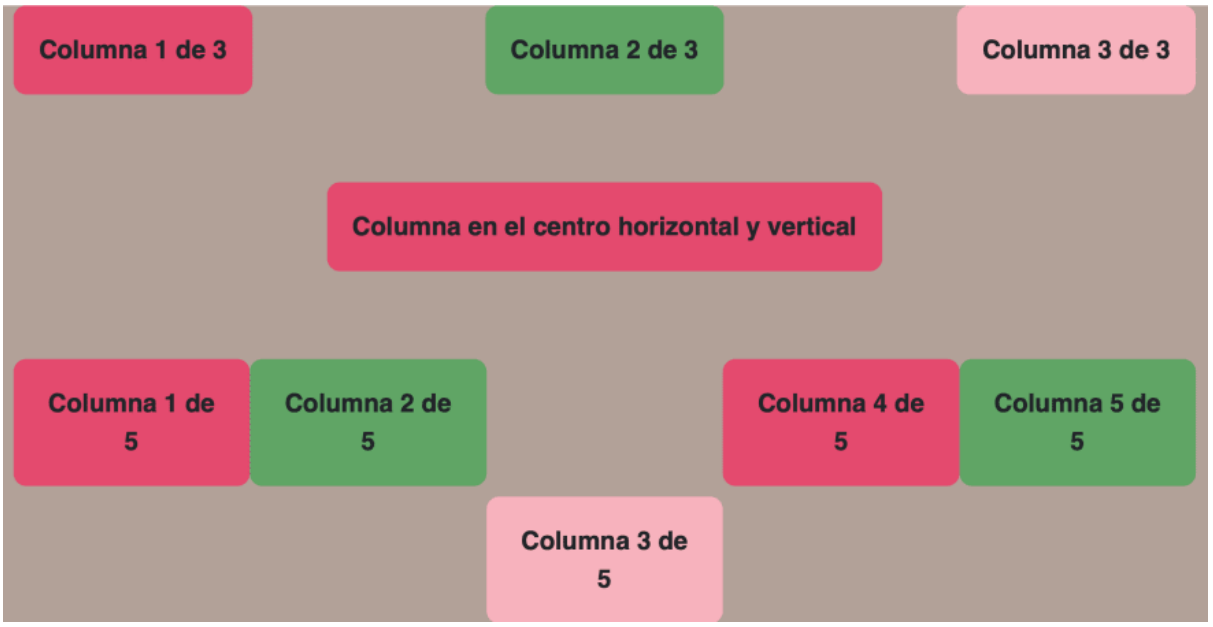
```
<div class="container mt-7 mx-auto">
    <div class="grid grid-cols-4">
```

```
        <div class="col-span-4">
            Fila completa
        </div>
        <div class="">
            Columna 1 de 4
        </div>
        <div class="">
            Columna 2 de 4
        </div>
        <div class="">
            Columna 3 de 4
        </div>
        <div class="">
            Columna 4 de 4
        </div>
    </div>
    <div class="grid grid-cols-3">
        <div class="col-start-1 col-end-2"> columna 1 con espacio al centro</div>
        <div class="col-end-4">columna 2 con espacio al centro</div>
        <div class="col-start-2 col-end-3">columna al centro</div>
    </div>
</div>
```

Como ven no hay gran diferencia en la implementación salvo que Tailwind cuenta con clases más específicas para grid que Bootstrap, pero con ambos se puede tener el mismo resultado.

Flexbox

Siguiendo con Flexbox, como sólo podemos trabajar en una dirección, nos enfocamos en **columns** creando este ejemplo:



Para implementar este layout veamos cómo se hace primero con CSS puro:

Nuestro HTML

```
<div class="container">
    <div class="flex flex-justify">
        <div class="">Columna 1 de 3</div>
        <div class="">Columna 2 de 3</div>
        <div class="">Columna 3 de 3</div>
    </div>
<!-- recuerda que para que funcione el centrado vertical el contenedor debe tener
un tamaño estatico -->
    <div class="flex flex-center box-height-static">
            <div  class="item-height-static">Columna  en  el  centro  horizontal  y
vertical</div>
    </div>
    <div class="flex box-3 box-height-static">
        <div class=" item-height-static">Columna 1 de 5</div>
        <div class=" item-height-static">Columna 2 de 5</div>
        <div class=" item-height-static item-bottom">Columna 3 de 5</div>
        <div class=" item-height-static">Columna 4 de 5</div>
        <div class=" item-height-static">Columna 5 de 5</div>
    </div>
</div>

Como no usamos Frameworks, necesitamos nuestro CSS
/* Agregamos flex */
.flex{
    display: flex;
}

.flex-justify{
    justify-content: space-between;
}

.flex-center{
    justify-content: center;
    align-items: center;
}

.box-3 {
    justify-content: space-between;
    align-items: flex-start;
}

.item-bottom {
    align-self: flex-end;
}

.item-height-static{
    height: 2rem;
}

.box-height-static {
    height: 250px;
}
```

No necesitamos otra cosa aparte de flex.

De igual manera, si queremos tener el mismo resultado en Bootstrap y Tailwind, estos frameworks ya cuentan con clases sin necesidad de editar nuestro CSS.

Flexbox en Bootstrap

```
<div class="container mt-5">
    <div class=" d-flex justify-content-between">
        <div class="">Columna 1 de 3</div>
        <div class="">Columna 2 de 3</div>
        <div class="">Columna 3 de 3</div>
    </div>
<!-- Al igual que con el ejemplo de CSS puro, se necesita un contenedor con tamaño
estático para que algunas de las clases funcionen -->
              <div    class="d-flex    justify-content-center    align-items-center
box-height-static">
        <div class="">Columna en el centro horizontal y vertical</div>
    </div>
            <div    class="d-flex    justify-content-around    align-items-baseline
box-height-static gray">
        <div class="">Columna 1 de 5</div>
        <div class="">Columna 2 de 5</div>
        <div class="align-self-end">Columna 3 de 5</div>
        <div class="">Columna 4 de 5</div>
        <div class="">Columna 5 de 5</div>
    </div>
</div>
```

Flexbox en Tailwind

```
<div class="container mt-7 mx-auto">
    <div class="flex justify-between">
        <div class="">Columna 1 de 3</div>
        <div class="">Columna 2 de 3</div>
        <div class="">Columna 3 de 3</div>
    </div>
    <div class="flex justify-center items-center box-height-static">
         <div class="">Columna en el centro horizontal y vertical</div>
    </div>
    <div class="flex justify-around items-baseline h-48">
        <div class="">Columna 1 de 5</div>
        <div class="">Columna 2 de 5</div>
        <div class="self-end">Columna 3 de 5</div>
        <div class="">Columna 4 de 5</div>
        <div class="">Columna 5 de 5</div>
    </div>
</div>
```

La implementación de flexbox en ambos frameworks es casi la misma (a excepción de algunos nombres de clases), pero la idea se mantiene.