UTN AVELLANEDA

aprendé programando

GYEE

Desarrollo Web I

Clase 7: Grilla responsiva

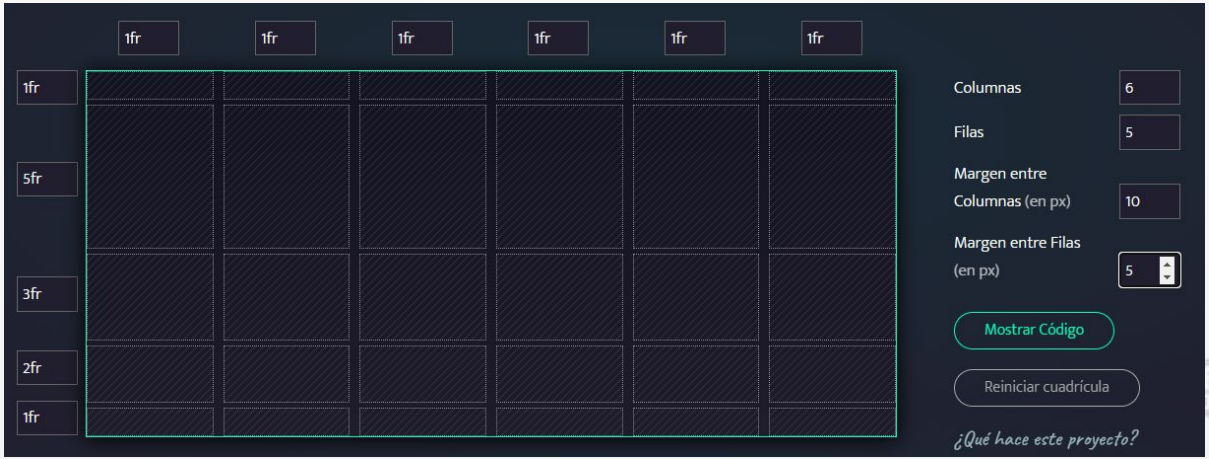
grid-generator

Este generador de grillas nos permitirá generar un layout en cuestión de minutos. Ingresá a https://cssgrid-generator.netlify.app/ y conozcamos su pantalla.

PASO A PASO

1) El primer paso es crear la grilla con la cantidad de filas y columnas que se requiera.
2) Luego, haciendo clic en cada input fr, cambiar la medida que se desee.
3) Determina el gap (recuerda que es en pixeles).
4) Esto se configura con los inputs del lado derecho de la pantalla.

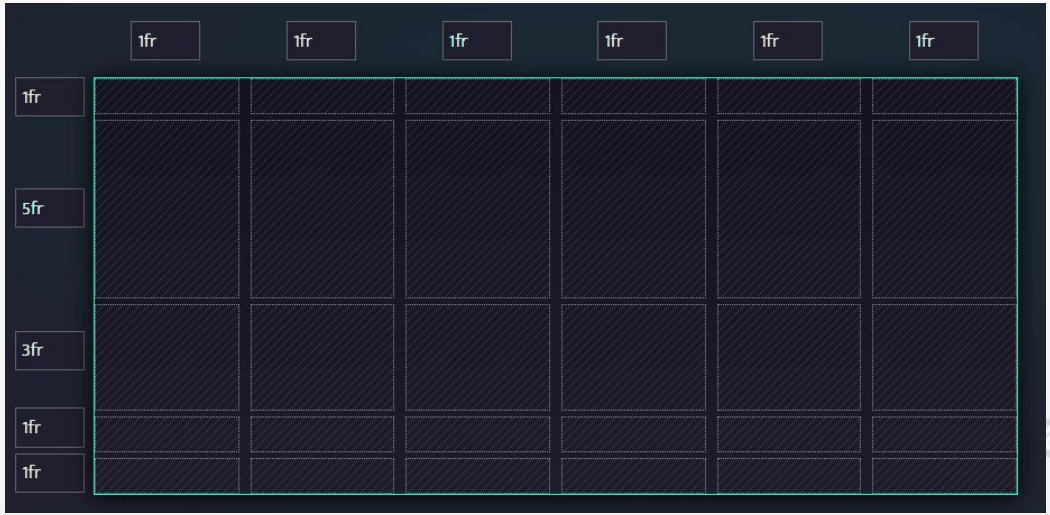
¡Inventemos una grilla!



PASO A PASO

Una vez creada la grilla, ya se pueden introducir los div o contenedores. Para eso se hace un clic en cada cuadrante. Esto depositara un div en cada espacio.

Para que un div ocupe más de un cuadrante se debe hacer clic sostenido, y sin soltar, recorrer todos los cuadrantes que se quieren ocupar.



PASO A PASO

Haciendo clic en “Mostrar código”, podremos ver el HTML y CSS de la grilla creada.

```html
<div class="parent">
  <div class="div1"> </div>
  <div class="div2"> </div>
  <div class="div3"> </div>
  <div class="div4"> </div>
  <div class="div5"> </div>
  <div class="div6"> </div>
  <div class="div7"> </div>
  <div class="div8"> </div>
</div>
```

```css
.parent {
  display: grid;
  grid-template-columns: repeat(6, 1fr);
  grid-template-rows: 1fr 5fr 3fr repeat(2, 1fr);
  grid-column-gap: 10px;
  grid-row-gap: 5px;
}
.div1 { grid-area: 1 / 1 / 2 / 7; }
.div2 { grid-area: 2 / 1 / 3 / 3; }
.div3 { grid-area: 2 / 3 / 3 / 5; }
.div4 { grid-area: 2 / 5 / 3 / 7; }
.div5 { grid-area: 3 / 1 / 4 / 4; }
.div6 { grid-area: 3 / 4 / 4 / 7; }
.div7 { grid-area: 4 / 1 / 5 / 7; }
.div8 { grid-area: 5 / 1 / 6 / 7; }
```

COSAS NUEVAS

Vamos a ver cosas nuevas en este código. Analicemos en detalle.

- ¿Qué significa **repeat(6, 1fr)**?

Significa que se crearán 6 columnas iguales de una fracción cada una.

Repetir 6 columnas de una fracción"

- ¿Qué significa **.div1 { grid-area: 1 / 1 / 2 / 7; }**?

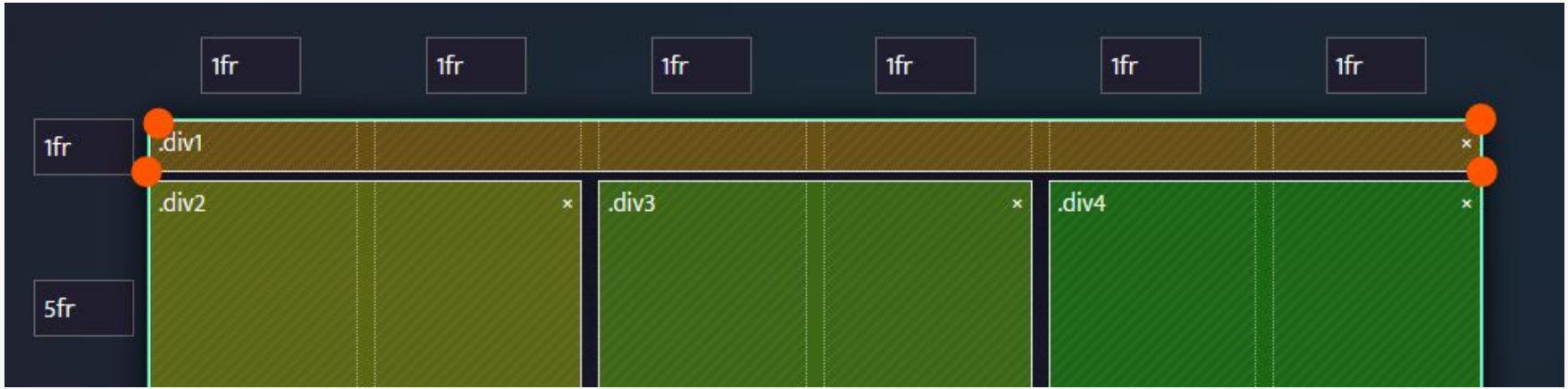
La propiedad grid-area se utiliza para especificar la ubicación y el tamaño del elemento dentro de una grilla. En este caso, la declaración 1 / 1 / 2 / 7 define los siguientes valores:

- El primer valor 1 indica que el elemento comienza en la primera fila de la grilla.
- El segundo valor 1 indica que el elemento comienza en la primera columna de la grilla.
- El tercer valor 2 indica que el elemento termina en la segunda fila de la grilla.
- El cuarto valor 7 indica que el elemento termina en la séptima columna de la grilla.

Es otra manera de ubicar elementos dentro de una grilla.

grid-area

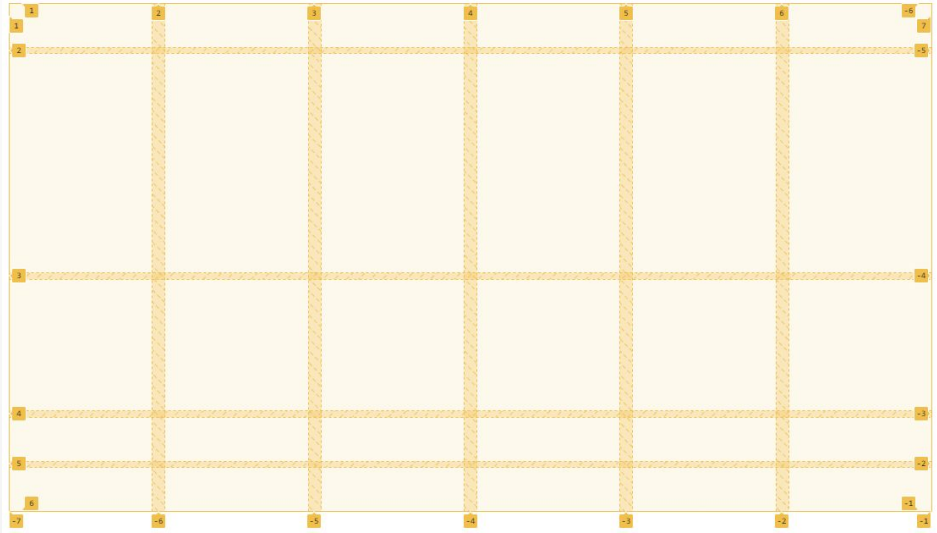
.div1 { grid-area: 1 / 1 / 2 / 7; }



GRILLA

Llevemos estos códigos al proyecto.

Te recomendamos darle una height al contenedor para poder ver los resultados. Recordá que la grilla y los div que contiene están vacíos.



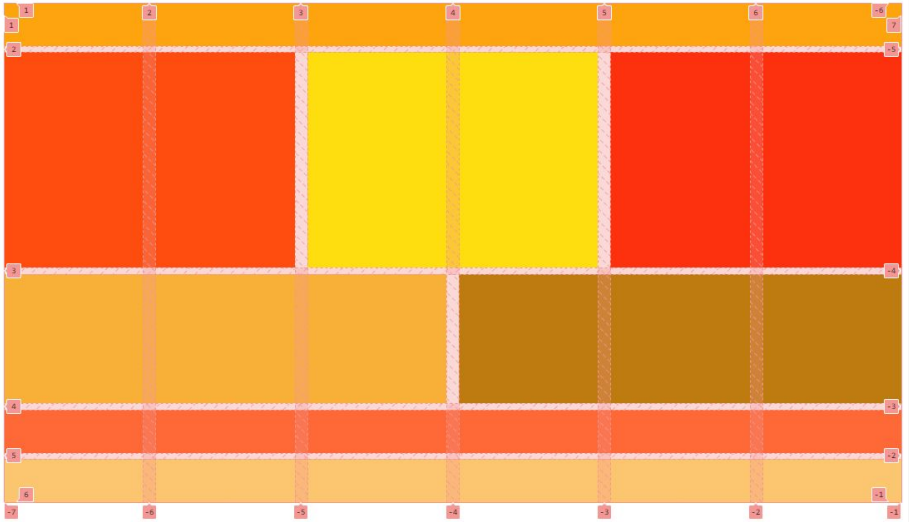
GRILLA

Para poder visualizar cada div, te proponemos colorearlos.

¿Cómo?

Colocando un background-color a cada uno de ellos. Ejemplo:

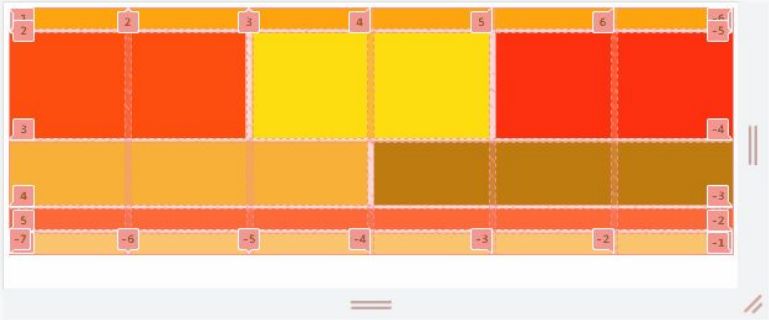
```
.div1 {
 grid-area: 1 / 1 / 2 / 7;
 background-color: orange;
}
```



GRILLA RESPONSIVA

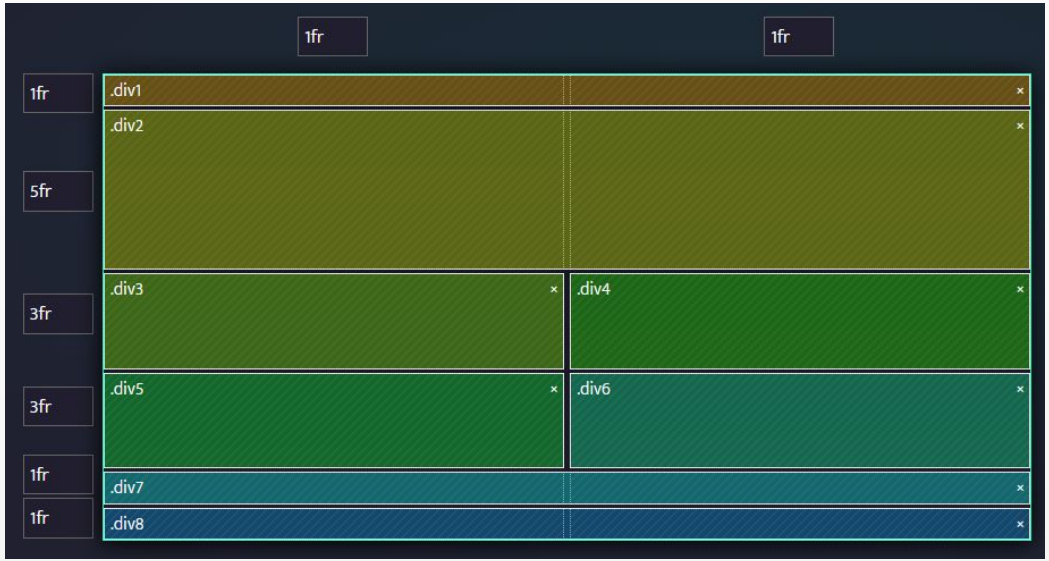
Ya creamos una grilla completa y le colocamos div vacíos para ver la disposición de los mismos. Para lograr que esta grilla cambie la disposición de sus filas y columnas, y además, cambié la distribución de los div, debemos seguir estos pasos:

1) Crear una nueva grilla para un tamaño de pantalla más pequeño
2) Crear una nueva distribución de TODOS los div en la nueva grilla. Si en el ejemplo anterior hay ocho divs, la misma cantidad debe haber en la grilla más pequeña.
3) Integrar estas nuevas propiedades del contenedor dentro de una media query.



GRILLA RESPONSIVA

Dibujamos una nueva grilla con una cantidad de filas y columnas diferente, ajustamos sus medidas (los fr), y colocamos en ella los ocho div anteriores.



GRILLA RESPONSIVA

Copiamos los nuevos códigos CSS que nos ofrece **dentro de una media query.**

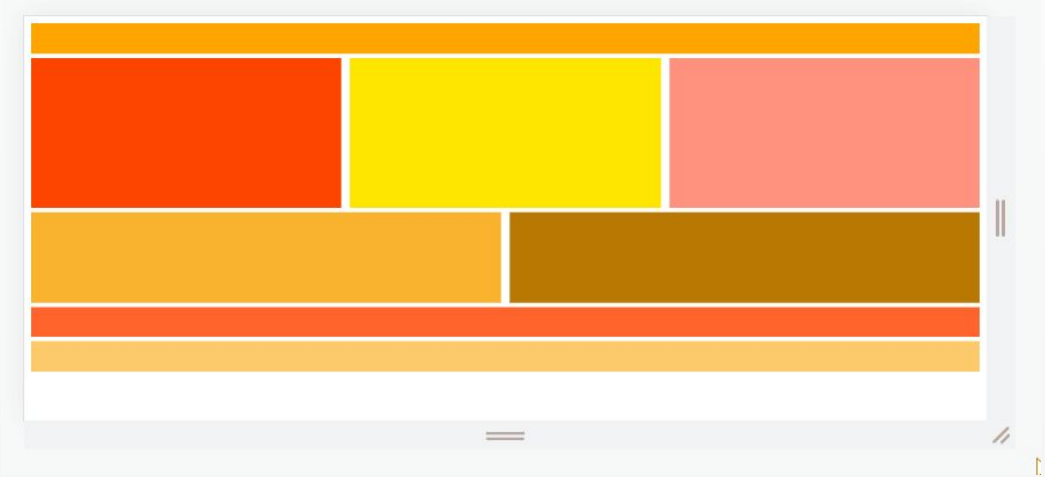
Ajustamos lo que creamos necesario, ya sea color, una nueva distribución, etc.

```css
@media (max-width: 922px) {
  .parent {
    display: grid;
    grid-template-columns: repeat(2, 1fr);
    grid-template-rows: 1fr 5fr repeat(2, 3fr) repeat(2, 1fr);
    grid-column-gap: 5px;
    grid-row-gap: 3px;
  }

  .div1 {
    grid-area: 1 / 1 / 2 / 3;
```

RESULTADO

Probamos la grilla con el inspector, achicando la pantalla para ver si cambia la grilla.



¡BRAVO!

Aprendiste a hacer una grilla responsiva.

Flexbox y grids son herramientas son de suma importancia en el diseño y desarrollo de sitios web responsivos.

De acá en adelante repensá el diseño de tu proyecto en base a una pantalla más pequeña como por ejemplo, celular.

¿Qué otros dispositivos podrían mostrar páginas web?

¿Qué otros tamaños de pantalla existen?

Con los nuevos celulares que se pliegan, ¿qué tamaño corresponde?

Esta y muchas otras preguntas te las dejamos para investigar y descubras un mundo de posibilidades con la cantidad de tamaños de pantallas que existen hoy.

¡MUCHAS GRACIAS!

UTN AVELLANEDA

aprendé programando

GYEE