

Grid y Media Queries

Grid

¿Qué es?

Grid es una herramienta muy potente de css que le permite a un elemento padre modificar el layout (su ubicación en la página) de los elementos hijos en 2 ejes, vertical y horizontal (a diferencia de flex que solo puede organizar en 1 eje).

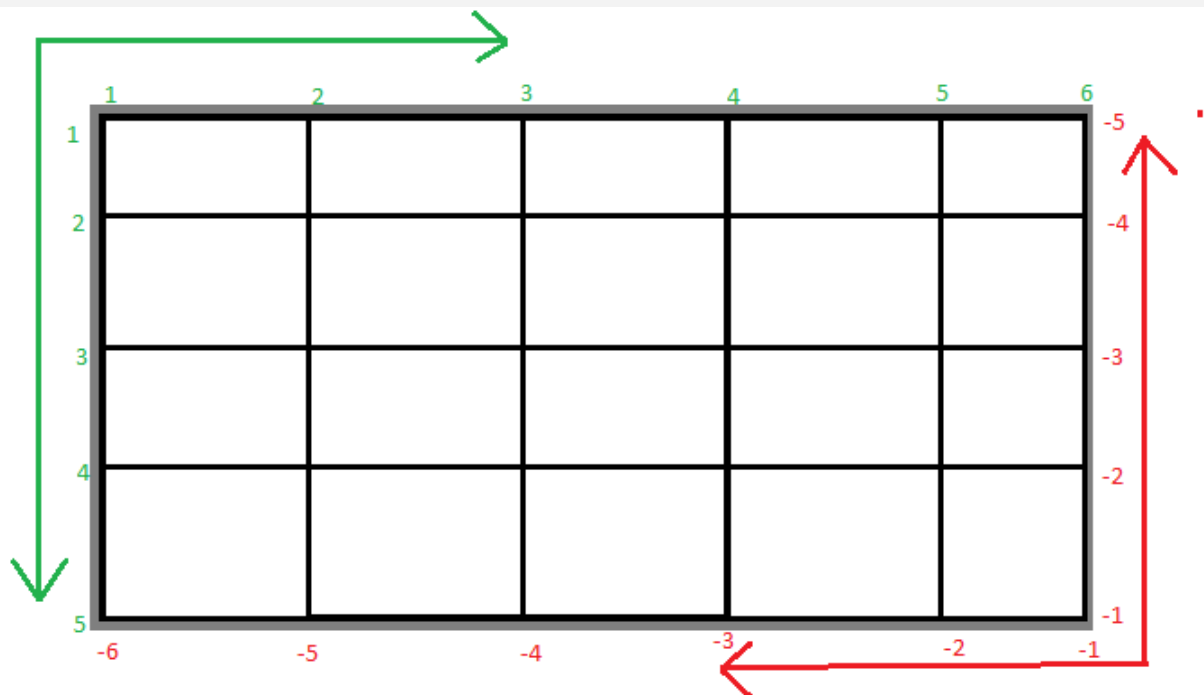
Herencia

Grid no se hereda, por lo tanto, no es capaz de modificar a los “nietos” del elemento padre. Sin embargo, uno puede crear otro grid a nivel del hijo o utilizar flex donde sea necesario para modificar a los “nietos”. Hay solo una excepción a esto que es cuando se activa subgrids en los elementos hijos pero por ahora solo funciona en firefox así que no va a ser explicado.

Bases

En grid se pueden clasificar 6 cosas distintas (después de nombrarlos no voy a estar mencionando constantemente de grid ya que todos estamos en contexto)


1. *Contenedor*: el elemento que contiene el display: grid;
2. *Línea de grid*: son las líneas divisorias de la grilla. Estas pueden ser verticales u horizontales y son numeradas desde 1. No se olviden que se cuentan las líneas y no las filas/columnas, osea que se incluyen los bordes del contenedor y las líneas que separan filas y columnas. Cada línea se numera 2 dos maneras distintas de la siguiente manera:
 - a. Desde la esquina superior izquierda (verde) se toman valores positivos de 1 a infinito positivo
 - b. Desde la esquina inferior derecha (rojo) de valores negativos desde el -1 a infinito negativo.



3. *Ítem de grid*: hijo del contenedor
4. *Celda de grid*: el espacio entre 2 columnas y filas adyacentes de grid.
5. *Banda de grid*: el espacio entre 2 líneas paralelas adyacentes
6. *Área de grid*: la región delimitada entre 2 líneas de filas y columnas. Se considera área cuando ocupa un tamaño mayor a una celda y siempre es rectangular (un cuadrado es un rectángulo)

- Celda 
- Banda 
- Área 
- Línea 

Unidades de medición


En grid se van a trabajar mucho con unidades de medición y especiales. Estas unidades se dividen en absolutas y relativas, si poseen el símbolo  es que son exclusivas de grid.

Absolutas

- px (pixel)
- cm (centímetros)
- mm (milímetros)
- in (pulgadas)
- pt (puntos)
- pc (picas)

Recomiendo utilizar pixeles para grids




Relativas

- fr (fracción) : el espacio restante no tomado por unidades absolutas se divide en X partes iguales donde X es la cantidad de fracciones asignadas (algo puede medir más de 1 fracción, como 3 fr)
- % (porcentaje)
- em
- rem
- vw
- vh
- vmin
- vmax
- ex

- ch

Recomiendo utilizar % y fr para grids

Especiales

- minmax(X, Y) : permite que el tamaño se encuentre entre un valor mínimo X y un valor máximo Y, pueden ser unidades absolutas o relativas.
- min(X) : indica el tamaño mínimo (absoluto o relativo), puede aumentar lo que quiera.
- max(X) : indica el tamaño máximo (absoluto o relativo), puede disminuir lo que quiera.
- min-content: el tamaño mínimo para contener todo (en un párrafo sería el tamaño de la palabra más larga)
- max-content: el tamaño máximo para contener todo (intentaría poner un párrafo en una sola línea)
- auto: funciona como fr pero tiene menor prioridad
- fit-content: usa el espacio disponible pero nunca más del min content ni más del max content (es un minmax con min-content y max-content de parámetros)

Recomiendo todos para grids

Propiedades del Contenedor

display

Toma valores grid para un grid tamaño bloque o inline-grid para tamaño en línea.

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

grid-template-columns | grid-template-rows

Se definen las filas y las columnas. Ambos reciben la misma sintaxis. Por cada columna/fila a crear se define un tamaño, esto se puede repetir por cada columna/fila a crear separado con un espacio.

A su vez, otro uso mucho menos utilizado es asignar nombre a las líneas intercalando el nombre en [] con el tamaño de columnas/filas.

```
.contenedor{
  display: grid;
  grid-template-columns: 1fr 50px [mirame] 30%;
  grid-template-rows: [inicio] minmax(50px, max-content) 4fr;
  /* genera un grid de 3 columnas y 2 filas, la linea de fila de valor 1
  se llama inicio y la linea columna de valor 3 se llama mirame*/
}
```

A su vez se puede utilizar la función repeat(x, y) , que repite x veces lo escrito en el parámetro y. Ejemplo:

```
.contenedor{
  display: grid;
  grid-template-columns: 100px repeat(5, 1fr);
  grid-template-rows: repeat(3, minmax(50px, 1fr));
  /* genera un grid de 6 columnas y 3 filas*/
}
```

grid-template-areas

Llama a los grid area names de los ítems para posicionarlos por nombre en el grid. No puede llamar clase, etiqueta o id, tiene que declararse el nombre en los ítems con grid-area (veanlo dentro de ítems). A su vez, se puede reemplazar el nombre por "." si uno quiere una celda vacía o por "none" si uno desea que no se defina un área.

Cada fila debe de tener un " en principio y fin y los nombres se separan con espacio .

Finalmente, si se repite el nombrado de un nombre varias veces de manera adyacente en la que respete rectángulos se puede hacer que un ítem ocupe varias casillas:

grid-template

Es un shorthand (propiedad de css que combina a varias propiedades). Este combina grid-template-rows, grid-template-columns y grid-template-areas.

Además de los valores normales se le puede poner none para dejar a las 3 propiedades en sus valores default.

gap y familia

column-gap (el viejo era grid-column-gap), row-gap (el viejo era grid-row-gap) y gap (el viejo era grid-gap) son las propiedades que definen el espacio entre filas y columnas (también se puede llamar el tamaño de las líneas).

A column-gap y a row-gap se le asigna un valor de tamaño único.

Por otro lado, gap es un shorthand de column-gap y row-gap y puede recibir 1 o 2 valores:

- Si se le asignan 2 valores, el primero se le asigna a row-gap y el segundo a column-gap
- Si se asigna solo 1 valor se le aplica a row-gap y column-gap por igual.

justify-items | align-items | place-items

Estas propiedades solo van a funcionar si el tamaño del ítem es MENOR a la celda que los contiene, de no haber espacio entre la celda y el ítem no va a haber ningún cambio visual.

justify-items

Alinea en el eje de las filas. Los valores posibles son:

- start: alinea al inicio de la celda.
- end: alinea al final de la celda.
- center: alinea al centro de la celda.
- stretch(default): hace que llene el contenido de la celda.

Se puede modificar a nivel de ítem con la propiedad de item justify-self.

align-items

Alinea en el eje de las columnas. Los valores posibles son:

- start: alinea al inicio de la celda.
- end: alinea al final de la celda.
- center: alinea al centro de la celda.
- stretch(default): hace que llene el contenido de la celda.
- baseline: valor más complejo. Alínea con respecto al texto, este puede ser "first baseline" o "last baseline" si se quiere que sea con respecto a la primera o la última línea de texto.

place-items

Es un shorthand para justify-items y align-items. Recibe primero el valor de align items y luego de justify items, al igual que gap, si se declara 1 solo valor se aplica a ambos por igual.

justify-content | align-content | place-content

Estas son propiedades que se pueden utilizar solo si queda espacio libre entre las celdas y el contenedor, los gap NO cuentan como espacio libre. Funcionan como un flex interno del grid

Al igual que las propiedades anteriores, justify es para el eje de las filas y align es para el eje de las columnas. Los valores que pueden recibir son:

- start: alinea al inicio de la celda.
- end: alinea al final de la celda.
- center: alinea al centro de la celda.
- stretch: hace que llene el contenido de la celda.
- space-around: hace que el espacio libre se reparta por igual a las puntas.
- space-between: hace que el espacio se reparta por igual entre los elementos
- space-evenly: hace que el espacio se reparta por igual a los lados de cada elemento.

place-content es un shorthand de ambas propiedades, recibe primero align-content y segundo justify-content, de recibir 1 solo valor lo aplica a ambos por igual.

grid-auto-columns | grid-auto-rows

Define el tamaño de las columnas y las filas que se generan cuando algún ítem no entra en las filas/columnas declaradas en los templates.

El único argumento que recibe es un valor de tamaño, por defecto tienen el valor auto.

grid-auto-flow

Define como se agregan a la grid los ítems que no fueron declarado/no entraron en los templates. Los valores que puede adquirir son:

- row(default): al tratar de colocar el ítem, trata de llenar una fila y si no hay filas suficientes, genera una fila nueva.
- column: lo mismo que row pero en base a columnas.
- dense: trata de llenar todos los espacios vacíos con un algoritmo, esto es puramente visual, eso causa que en lectores de accesibilidad no señala correctamente el elemento que se ve (porque su posición en el layout es distinta), normalmente no se recomienda.

grid

Es un shorthand para grid-template-rows/columns, grid-template-areas, grid-auto-rows/columns y grid-auto-flow.

Es muy complejo como para explicarlo bien en un resumen, vean esto:

<https://developer.mozilla.org/es/docs/Web/CSS/grid>

Propiedades del item

grid-column-start/end | grid-row-start/end

Estas propiedades determinan la posición del ítem en el grid. Los start indican donde empieza en la grid y los end indican dónde terminan. Estas propiedades reciben un único valor, que puede ser:

- Valor o nombre de la línea
- span X: hace que el item ocupe hasta el valor X. La X puede ser:
 - Número de línea
 - Nombre de línea
- auto: el valor por defecto (suele ser 1)

grid-column | grid-row

Son shorthand de grid-X-start y grid-X-end donde x puede ser row o column. Recibe el valor inicial, el final y/o span con un valor en ese orden.

grid-area

Permite designar un nombre de área para grid-template-area e incluye dentro de el un shorthand de las propiedades grid-row-start/end y grid-column-start/end en ese orden. El nombre debe de ser una palabra única (no contener espacios).

justify-self | align-self | place-self

align/justify-self

Es lo mismo que el align/justify-items del contenedor. Pueden adquirir el valor:

- start: alinea al inicio de la celda.
- end: alinea al final de la celda.
- center: alinea al centro de la celda.
- stretch(default): hace que llene el contenido de la celda.

Place-self

Es un shorthand de las propiedades align-self y justify-self, puede recibir 1 o 2 valores que se puedan designar a align-self y justify-self:

- Con 2 valores se le aplica a align-self el primer valor y justify-self el segundo
- Con 1 valor se le aplica la misma propiedad a ambos.

Bibliografía:

- https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Grid_Layout

- <https://css-tricks.com/snippets/css/complete-guide-grid/>
-

Media Queries

¿Que son?

Las Media Queries son una herramienta de css que permite que un código de css se aplique si se cumple la condición utilizada. Los que tienen conocimientos de otros lenguajes de programación los encontraran similares a condicionales.

Mayormente se utilizan para hacer que una página web sea responsiva (responsive) aprovechándose de la propiedad cascada de css.

Sintaxis

@regla tipo-de-media (rasgo-de-media)

Regla

Es indispensable para el funcionamiento y le indica al dispositivo que es un media query, su valor es: @media

Tipo de Media

Se selecciona el tipo de media para el query, lo normal es usar "screen" pero hay varios tipos posibles:

- all: se fija en todos los dispositivos.
- print: se activa cuando se intenta imprimir la página.
- Screen: se activa en dispositivos con pantalla.
- speech (el nuevo aural): es activa en dispositivos diseñados como lector de pantallas.

Rasgos de Media

Acá se define tanto el rasgo como un valor. Hay distintas clasificaciones:

- Viewport
- Display Quality (calidad de presentación)
- Color
- Interaction (interacción)
- Video Prefixed (prefijos de video)
- Scripting (si permite scripts)
- User Preference (preferencia de usuario)

Para poder utilizar los de viewport, es necesario que en el head de nuestro html tenga este código para que pueda detectar cambios en el:

```
<meta name="viewport" content="width=device-width, initial-scale=1" />
```

Los rasgos que nos interesan entre todas las posibles son:

- width: para un valor específico como 1080px es igual a la pantalla (usen px preferiblemente).
- min-width: cuándo un valor de largo es mayor o igual a la pantalla (usen px preferiblemente).
- max-width: cuándo un valor de largo es mayor o igual a la pantalla (usen px preferiblemente).
- orientation: adquiere los valores "portrait" (portrato/vertical) o "landscape" (horizontal)

Siempre están entre (), normalmente se ven algo similar a esto:

```
@media only screen and (max-width: 768px) {
```



```
/*Codigo de CSS*/  
}
```

Operadores

También se pueden aplicar operadores de lógica en un query:

- and: un "y" lógico, se tienen que cumplir ambas condiciones
- or (también se puede usar ","): un "ó" lógico, se tiene que cumplir una de las condiciones.
- not: para cuando no se cumple la condición (le van a dar MUY poco uso).

```
@media screen (min-width: 320px) and (max-width: 768px) {  
  /* estilado */  
}
```

Anidación

Se pueden anidar media queries.

Responsividad

La página web tiene que ser responsiva, pero acá hay unos valores notables:

- 320px (celular en vertical con pantalla MUY chica)
- 360px (celular en vertical)
- 640px (celular en horizontal)
- 768px (tablet en vertical)
- 1024px (tablet en horizontal/laptop chica)
- 1366px (laptop/desktop chica)
- 1920px (desktop horizontal)

Mobile First

Es una escuela de diseño web que se centra en programar inicialmente para un dispositivo móvil y utilizar media queries de min-width de manera creciente para que se adapte a otras resoluciones.

Desktop First

Es otra escuela de programación de diseño web en la que se diseña inicialmente para una desktop y se utilizan media queries de max-width en orden decreciente para asegurarse la responsividad de la página.

Bibliografía

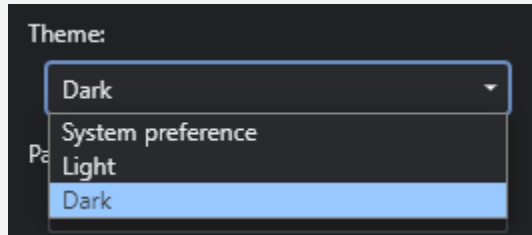
- <https://css-tricks.com/a-complete-guide-to-css-media-queries/>
- https://developer.mozilla.org/es/docs/Web/CSS/Media_Queries/Using_media_queries
- https://www.w3schools.com/css/css_rwd_mediaqueries.asp

Consola del ordenador

Una herramienta muy útil para cualquier diseñador web. Esta se puede abrir con f12 o clickeando la página con click derecho y seleccionar inspect (o inspeccionar).

Por defecto les aparecerá a la derecha del ordenador pero recomiendo que hagan click al menú de tres puntos (⋮) y lo seleccionen para que se ubique abajo (📄) o aparte (📄) ya que solemos utilizar media queries con width como referencia.

También si quieren, hagan click a setting (⚙️) para poder elegir el tema de la consola si blanco les es molesto:



Para verlo mas a profundo vean esta guía:

<https://docs.google.com/document/d/1AmGpUM12YeZpPiPMViXl6yXHTow6Kx-8nwicWfPkHpA/edit?usp=sharing>