CSS GRID LAYOUT

¡HOLA!

Soy **Diana Aceves**

Frontend Developer

Twitter: **@diana_aceves_**

EJEMPLOS DEL CURSO

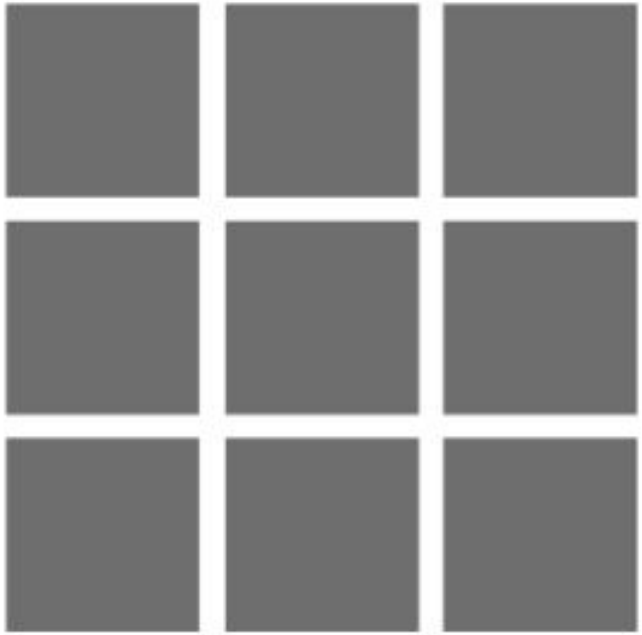
Colección Codepen:

“Escuela IT 2017:
Taller CSS GRID LAYOUT”

https://codepen.io/collection/712dfdc773b2532beaff9157a2ffe194/

¿QUÉ ES GRID LAYOUT ?

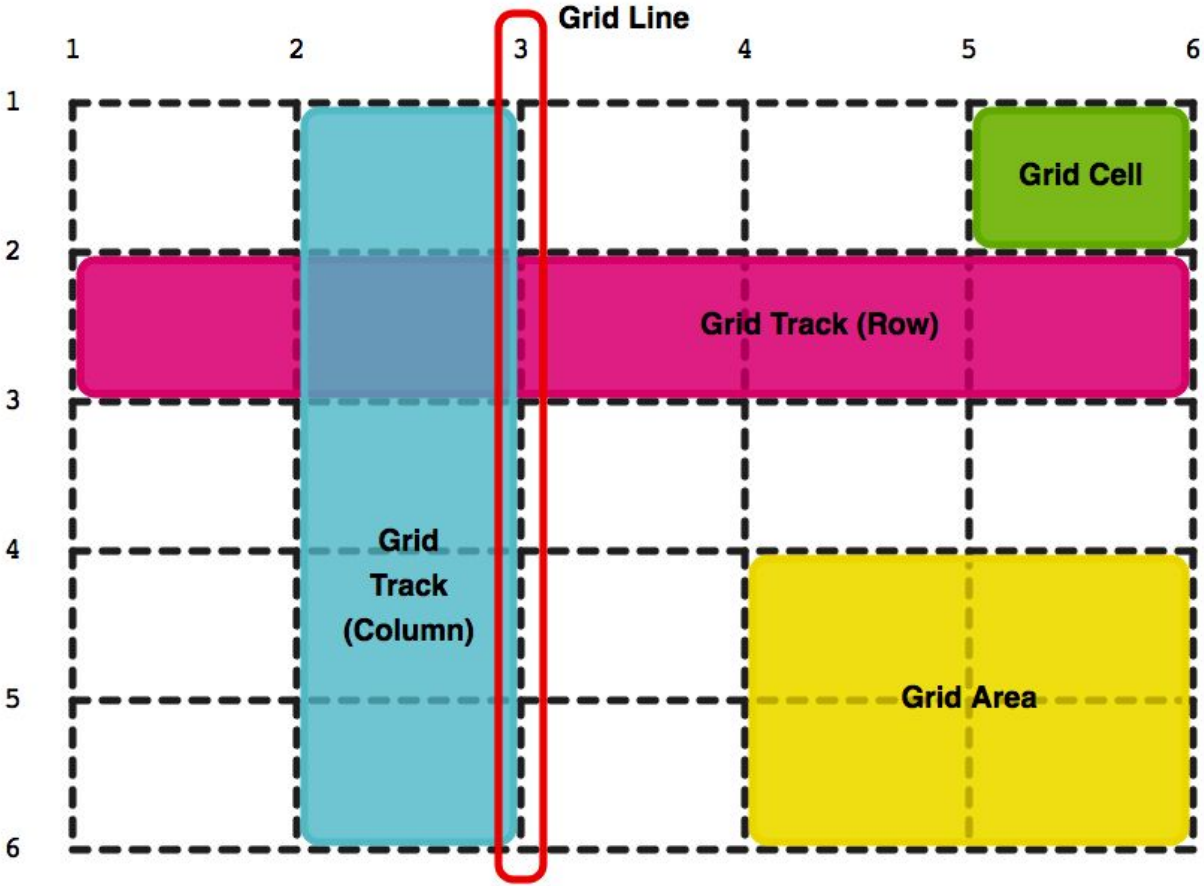
- Sistema de rejilla en
  2 DIMENSIONES



¿QUÉ TIENE DIFERENTE ?

- Voy a poder colocar los items **DONDE QUIERA…**
- **…PERO** habrá items que **SE COLOQUEN SOLOS** (AUTO-PLACEMENT)
- Puedo **HACER LO MISMO DE MUCHAS FORMAS…**
- **…PERO NO TODAS HACEN LO MISMO.**
- Controlo las 2 dimensiones, **NO ES FLEXBOX.**
- La colocación de los items es muy libre, **NO ES UNA TABLA.**
- Tiene una **EXTENSA SINTAXIS.**
- Nos va a volver un poco locos, pero **VA A CAMBIAR EL CSS PARA SIEMPRE.**

CONCEPTOS BÁSICOS



EMPECEMOS: DISPLAY GRID

En cuanto declaro

DISPLAY: GRID o

DISPLAY: INLINE-GRID

los hijos directos de ese elemento pasan a ser **GRID ITEMS.**

Vamos a verlo:
EJ.1 - DISPLAY:GRID

DEFINIENDO TRACKS EN EL GRID

- Creamos el grid **EXPLÍCITAMENTE.**
- Es decir, al **contenedor GRID**, le indico cómo deben ser las filas y las columnas:
- **EXISTEN MÚLTIPLES SINTAXIS Y FORMAS DE HACERLO.**

EJ.2 - GRID-TEMPLATE-COLUMNS/ROWS

Funciones y keywords MUY ÚTILES

- Tengo que saber cada track? Cuánto mide? Cuántos hay? Escribirlos todos? **MUERO!!!**

NO

- repeat()
- minmax()
- auto-fill / auto-fit

EJ.3.1 - REPEAT / MINMAX
EJ.3.2 - AUTO-FILL / AUTO-FIT

Visto hasta AHORA

- **DISPLAY: GRID** genera una serie de tracks donde, si no indico más, **los items se colocan automáticamente** mediante el algoritmo de **AUTO-PLACEMENT.**
- Puedo concretar más la rejilla generada con:
  - **grid-template-columns/rows**
  - **grid-auto-columns/rows**
  - Y todas las **funciones/propiedades/keywords** que hemos visto.

PERO AÚN NO HEMOS TOCADO LOS **GRID-ITEMS**

Posicionando GRID-ITEMS : NÚMEROS DE LÍNEA

La enorme potencia de GRID LAYOUT en parte viene dada porque una vez generada la rejilla, **puedo COLOCAR LOS ITEMS DONDE QUIERA** dentro de esta.

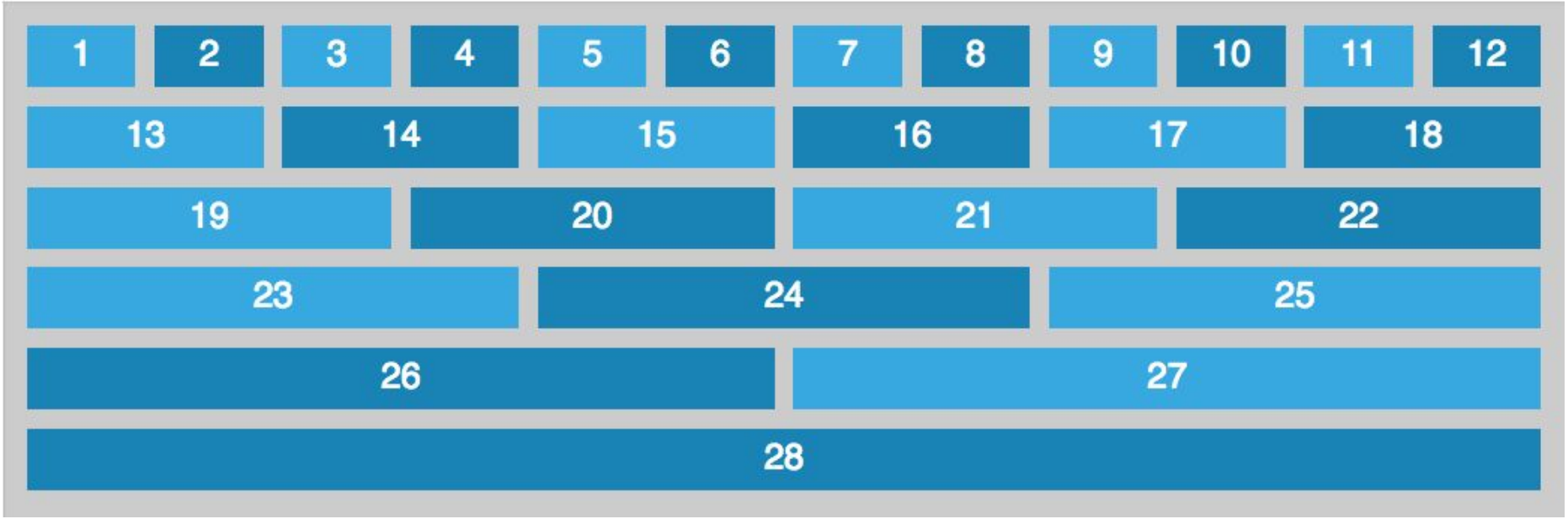
DEFINO UN GRID DONDE DESPUÉS PUEDO POSICIONAR LOS ITEMS.

EJ.4 - COLOCACIÓN + Nº DE LÍNEA

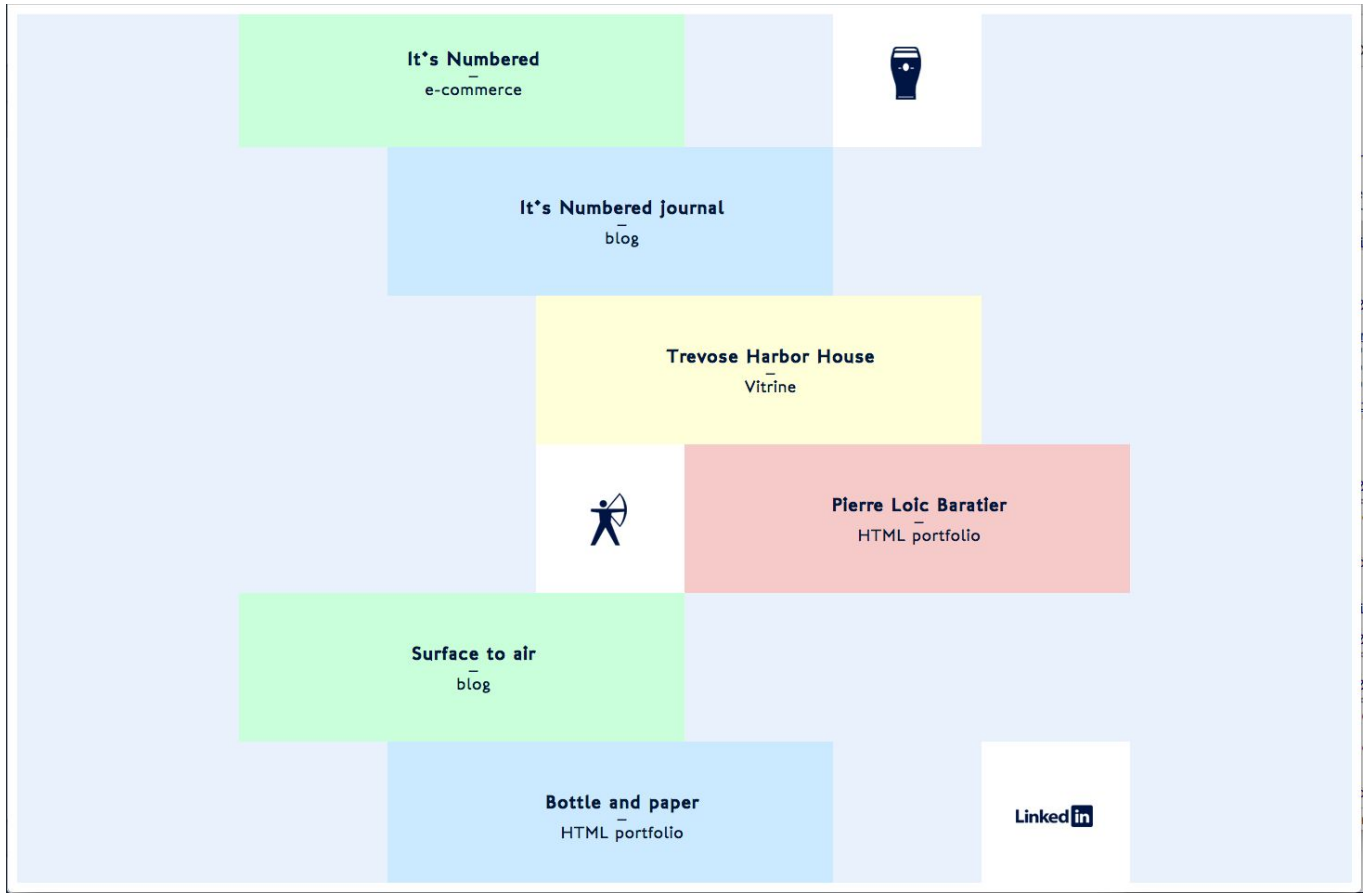
EJ.APLICACIÓN. 1 - GRID 12 COLUMNAS

EJ.APLICACIÓN. 2 - LAYOUT REAL thomasrobin

EJ.APLICACIÓN. 1 - GRID 12 COLUMNAS



EJ.APLICACIÓN. 2 - LAYOUT REAL thomasrobin



Posicionando GRID-ITEMS : LÍNEAS NOMBRADAS

- Me ayudan a recordar cómo van los tracks en layouts complejos.
- En **RESPONSIVE** me evitan sobreescribir la colocación de algunos items.
- Si cambia el número de tracks **NO TENGO QUE REPOSICIONAR TODOS LOS ELEMENTOS.**

EJ.5.1 - LÍNEAS NOMBRADAS

EJ.5.2 - LINEAS NOMBRADAS (nombres múltiples)

EJ.5.3 - LÍNEAS NOMBRADAS (variación número de tracks)

EJ.5.4 - LÍNEAS NOMBRADAS (nombres repetidos)

Posicionando GRID-ITEMS : ÁREAS NOMBRADAS

Dos nuevas propiedades :

- CONTENEDOR -> **GRID-TEMPLATE-AREAS**
- ITEMS -> **GRID-AREA**

EJ.6 - ÁREAS NOMBRADAS

EJ.APLICACIÓN. 3 - HOLY GRAIL LAYOUT