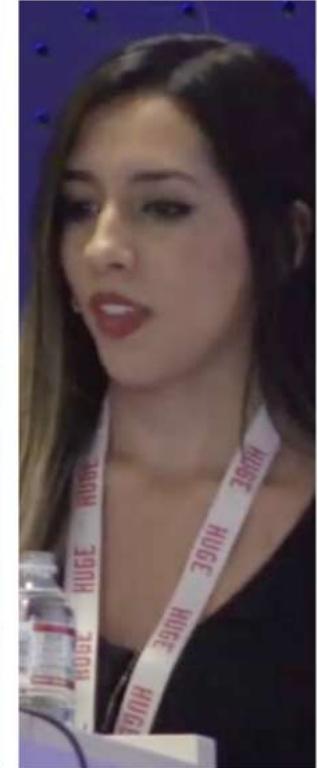
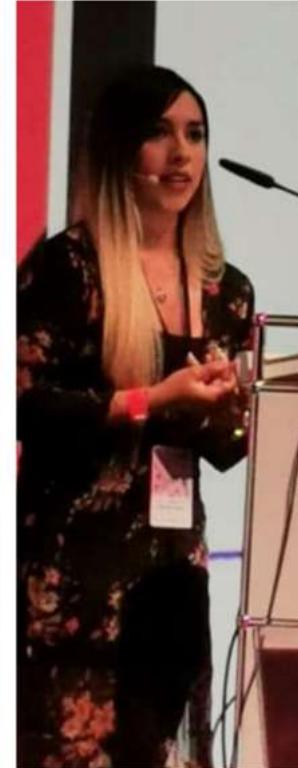


**Tips para que lleves  
tus conocimientos  
de CSS a otro nivel**

# Sobre mí

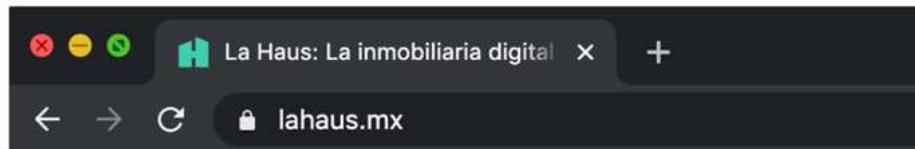


**ESTEFANY AGUILAR**

@teffcode   



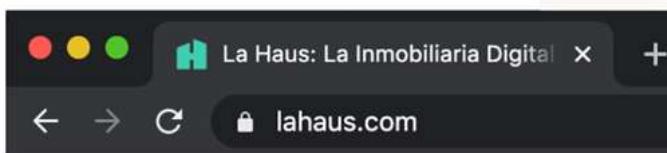
• • • • •  
• • • • •



LaHaus



Comprar casa comien  
una buena búsqueda



LaHaus



Comprar casa comienza con  
una buena búsqueda

• • • • •  
• • • • •



## Curso de Frontend Developer



Instruido por:  Estefany Aguilar

Básico

4 horas de contenido

[Ver la ruta de aprendizaje ▾](#)



[Proyecto del curso](#)

### Desarrollo visual de Platzi Video

Usando HTML Y CSS programarás y darás estilos al front de Platzi Video, una aplicación para ver videos en línea. Diseña el home, la pantalla de registro, el login y la pantalla de error de esta aplicación web.



Platzi **LIVE** PlatziConf 2020 - ...

Clases Blog Foro Agenda TV Ver Dashb

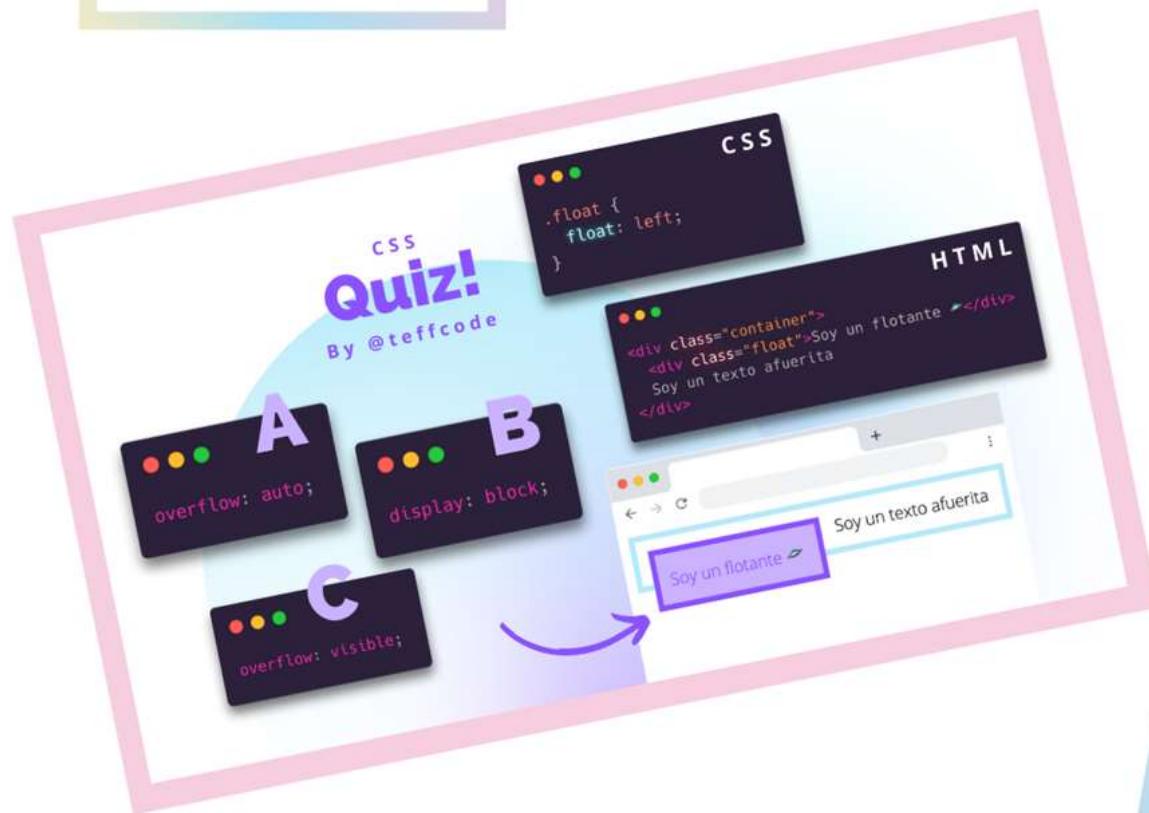
Freddy Vega

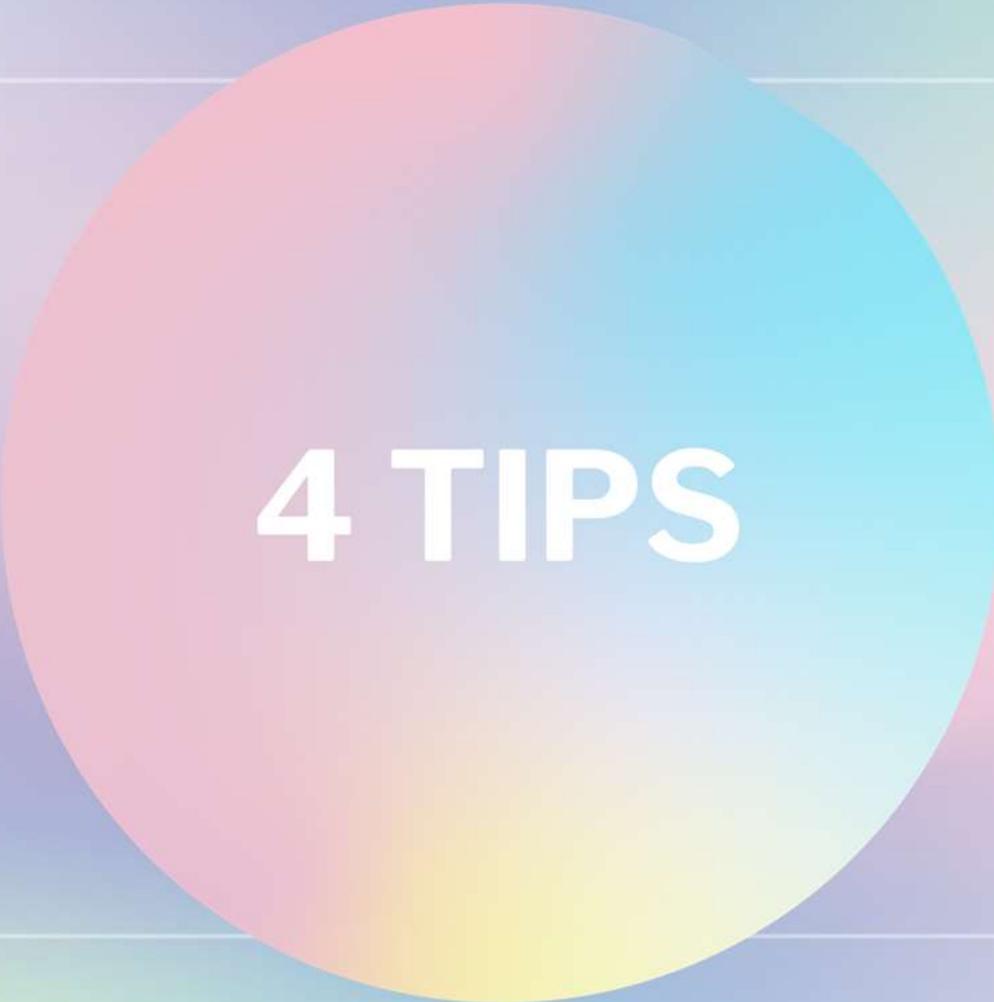
@freddier

Platzi **CONF**  
2020

#platziconf

# Hago Quices :)





**4 TIPS**

# TIP 1

No es necesario  
memorizar todas las  
propiedades y valores :)



**TIP 2**

**Conoce los conceptos  
fundamentales**

- Layout
- Selectores
- Responsive
- Flujo normal
- Modelo de caja
- Herencia y cascada
- Formatting Contexts

## TIP 3

**Usar DevTools para  
encontrar errores y  
hacer pruebas rápidas**

## TIP 4

**Saber qué herramientas  
te sirven para aprender**

# **En este curso**

- Historia
- Conceptos
- Retos
- Quices
- Proyecto  
creativo

# CSS 2020

## By @teffcode

Hola :) En este repositorio encontrarás toda la documentación que utilizamos en los cursos de CSS Grid y Diseño Web con CSS Grid y Flexbox. Adicional, también encontrarás los slides para que puedas estudiar con mucho más detalle (pero, son sólo para ti 😊).

Para comenzar, solo queremos recordarte que este contenido es exclusivo de Platzi 🎉

1. [Curso de CSS Grid](#)
2. [Curso de Diseño Web con CSS Grid y Flexbox](#)

## Curso de CSS Grid ❤️

1. [Tips para que lleves tus conocimientos de CSS a otro nivel + Quices](#)
2. [¿Cómo fue pensado CSS cuando se creó?](#)
3. [Limitaciones de CSS y el problema de los elementos flotantes](#)
4. [Herramientas que nos han facilitado el camino](#)
5. [¿CSS Grid es una idea nueva? La evolución de la especificación](#)
6. [¿Qué significa Grid para CSS?](#)
7. [Técnicas de alineamiento antes de CSS Grid \(Parte 1\)](#)
8. [Técnicas de alineamiento antes de CSS Grid \(Parte 2\)](#)

# Quices de CSS

**Te los dejo en la  
cajita de enlaces**



# 01

## ¿Cómo fue pensado CSS cuando se creó?



x



## Viajemos al pasado

Muchas de las suposiciones actuales sobre el diseño se basan en lo que sucedió antes.

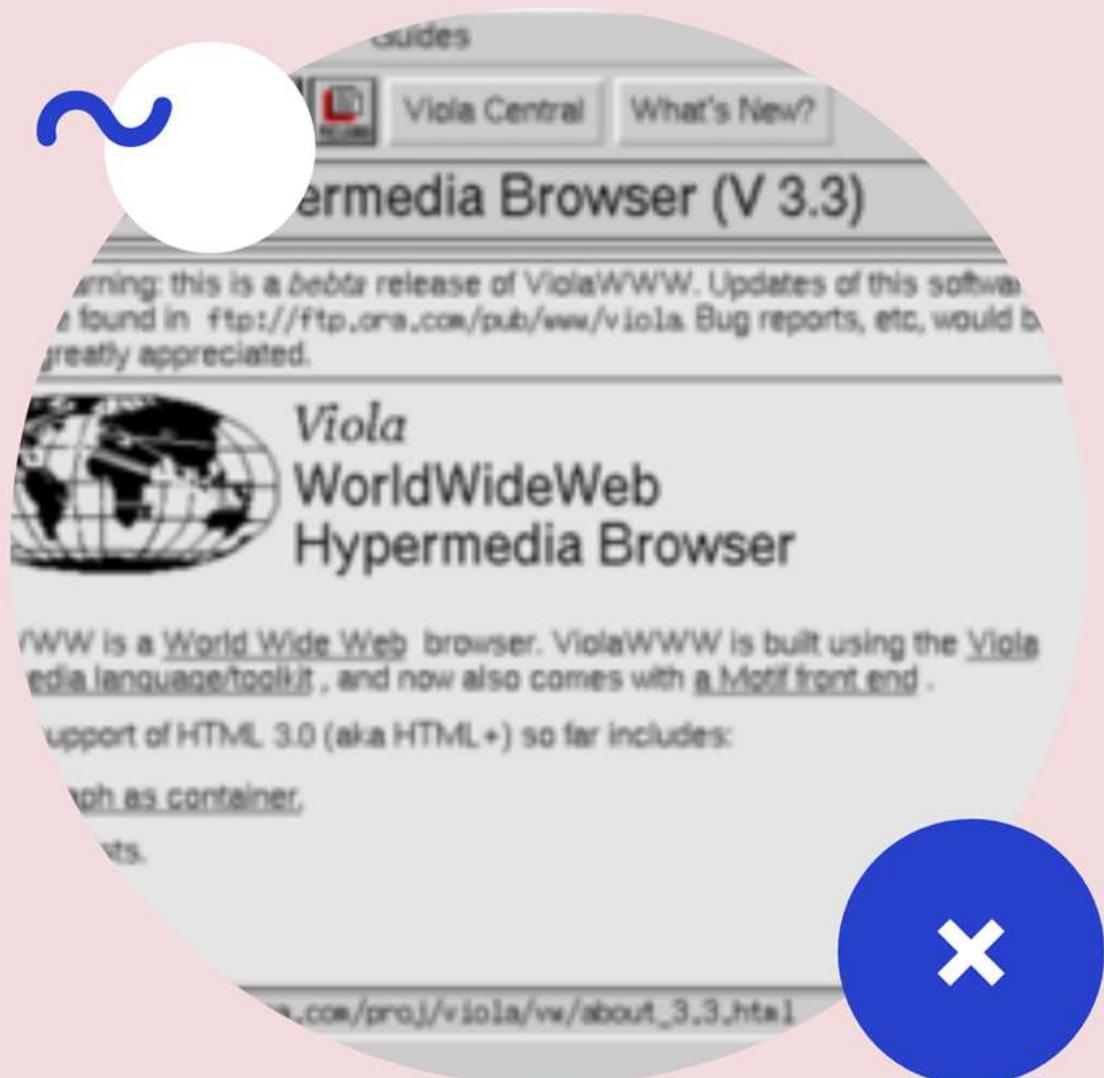
# 04

## Tim Berners-Lee



# Navegador Viola en 1993

- Tenía su propio lenguaje de estilos
- Quería convertir su lenguaje en un estándar para la web



06

1994

Navegador  
NCSA Mosaic  
(Hizo popular  
la web)



Marc Andreessen

07



×



# Håkon Wium Lie

Cascading HTML  
Style Sheets  
(CHSS)

**En 1994 se  
presentó la  
propuesta inicial  
de CSS**

\*



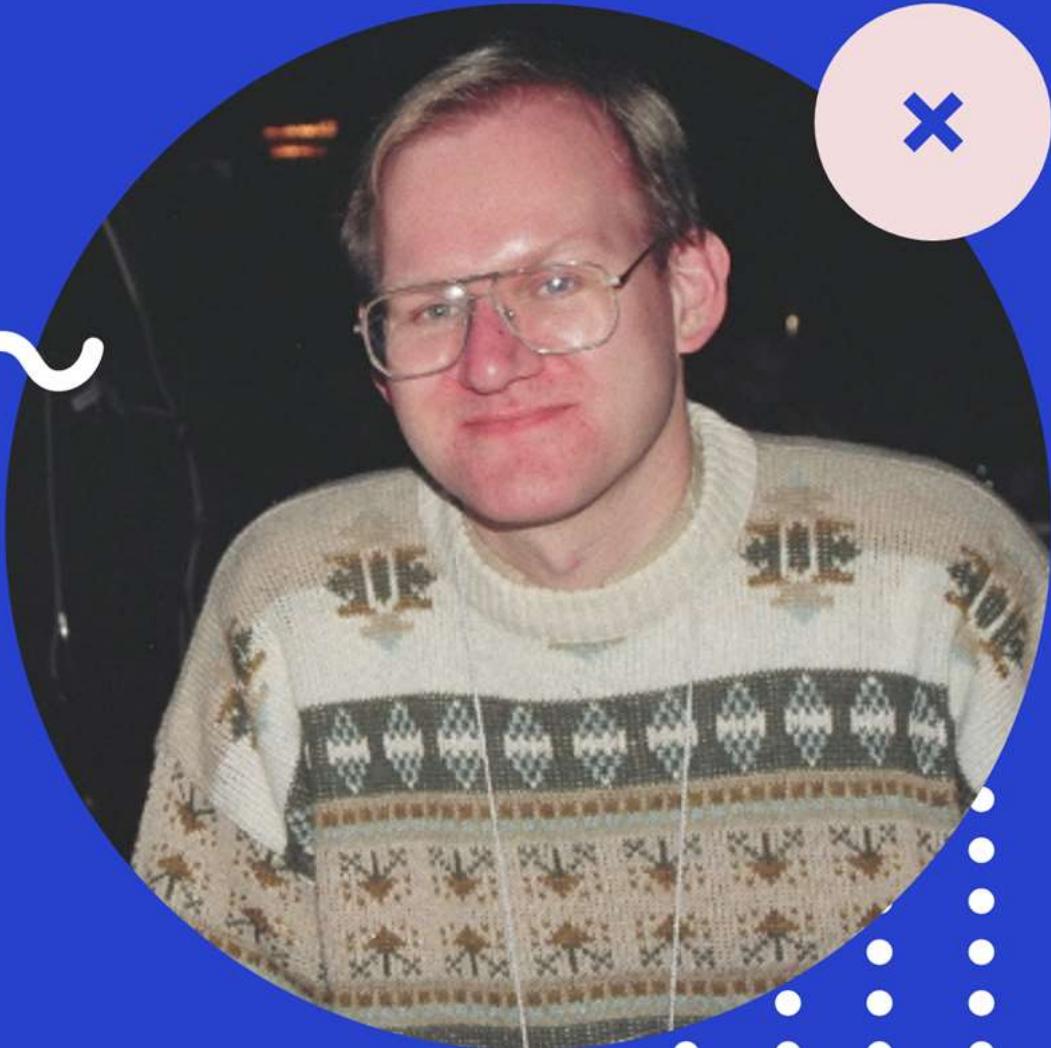
**Eso fue en el día  
del desarrollador y  
provocó mucha  
discusión**

# 09

## Bert Bos

Respondió al primer borrador de CSS y unió fuerzas con Håkon y se presentaron en la conf de WWW en 1995

?



X

1995 • W3C  
HTML ERB



✗

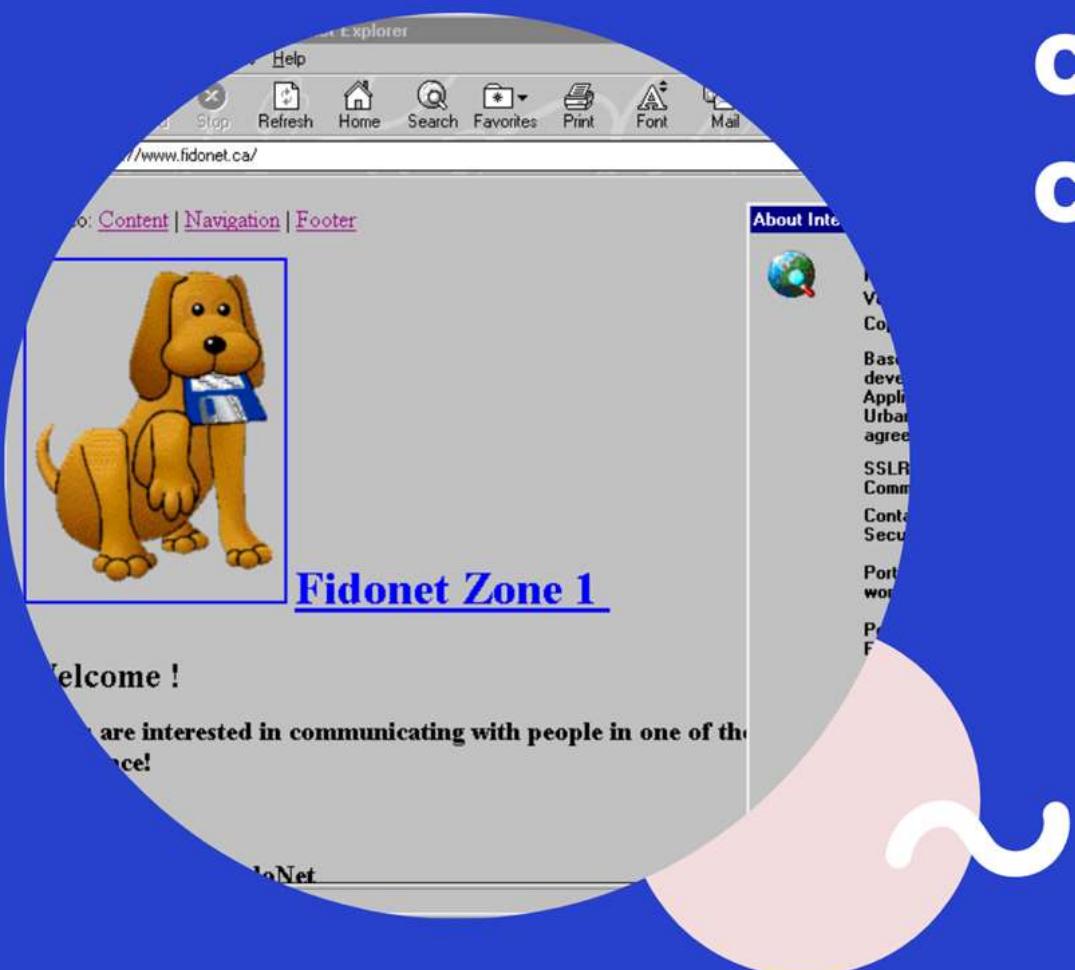
**1996 nace  
CSS como  
una  
recomen-  
dación de  
la W3C**

?



# Primeros navegadores compatibles con CSS:

1. IE3
2. NETSCAPE
3. OPERA



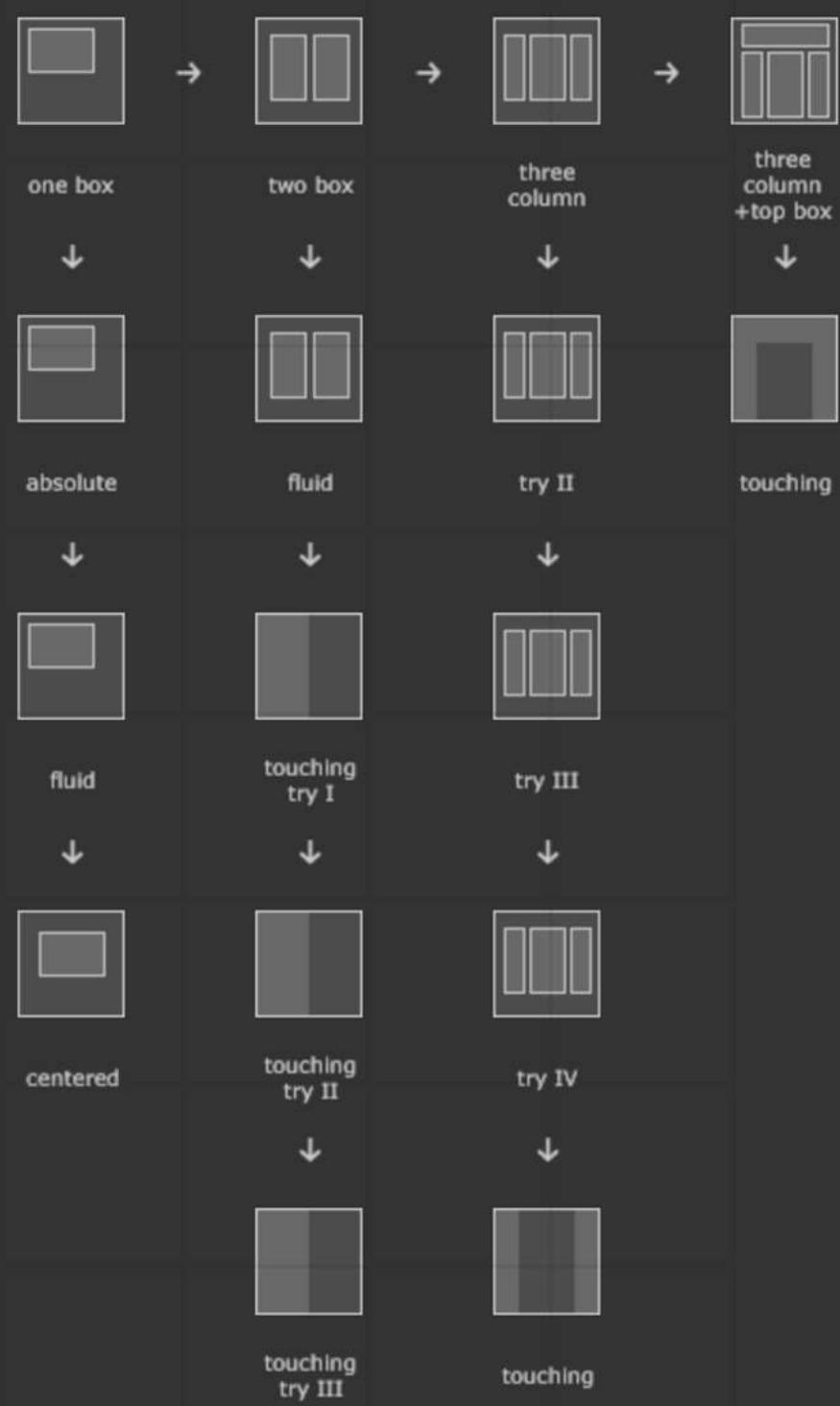
X

# **Limitaciones de CSS y el problema de los elementos flotantes**



**Los primeros diseños de  
CSS eran una mezcla  
entre elementos  
flotantes y posicionados**

# Noodle Incident (Recursos en línea para copiar y pegar)



**Seguían frustrados  
por la falta de  
columnas de altura  
completa**

# Columnas Falsas

Por Dan Cederholm

```
background: #ccc url(..../images/bg_birch_800.gif) repeat-y 50% 0;
```



# Diseño Responsivo

Por Ethan Marcotte

```
@media screen and (max-width: 400px) {  
    .figure,  
    li#f-mycroft {  
        margin-right: 3.317535545023696682%;      /* 21px / 633px */  
        width: 48.341232227488151658%;           /* 306px / 633px */  
    }    li#f-watson,  
    li#f-moriarty {  
        margin-right: 0;  
    }  
}
```



SHERLOCK  
HOLMES

DR JOHN HEMISH  
WATSON



# El problema de los elementos flotantes



No account of the early history of English aeronautics could possibly be complete unless it included a description of the Nassau balloon, which was inflated by coal-gas, from the suggestion of Mr. Charles Green, who was one of Britain's most famous aeronauts. Because of his institution of the modern method of using coal-gas in a balloon, Mr. Green is generally spoken of as the Father of British Aeronautics. During the close of the eighteenth and the opening years of the nineteenth century there had been numerous ascents in Charlier balloons, both in Britain and on the Continent. It had already been discovered that hydrogen gas was highly dangerous and also expensive, and Mr. Green proposed to try the experiment of inflating a balloon with ordinary coal-gas, which had now become fairly common in most large towns, and was much less costly than hydrogen.



# Esto funciona bien si se calcula con precisión el ancho y si el contenido tiene la misma altura

## Card 1

Floated items will always rise to the top. This means that we can use floats to create multiple columns.

## Card 2

Floated items will always rise to the top. This means that we can use floats to create multiple columns.

## Card 3

Floated items will always rise to the top. This means that we can use floats to create multiple columns.

## Card 4

Floated items will always rise to the top. This means that we can use floats to create multiple columns.

## Card 5

Floated items will always rise to the top. This means that we can use floats to create multiple columns.

## Card 6

Floated items will always rise to the top. This means that we can use floats to create multiple columns.

## Card 1

Floated items will always rise to the top. This means that we can use floats to create multiple columns.

## Card 2

Floated items will always rise to the top. This means that we can use floats to create multiple columns.

This card now has some additional content.

## Card 3

Floated items will always rise to the top. This means that we can use floats to create multiple columns.

## Card 4

Floated items will always rise to the top. This means that we can use floats to create multiple columns.

## Card 5

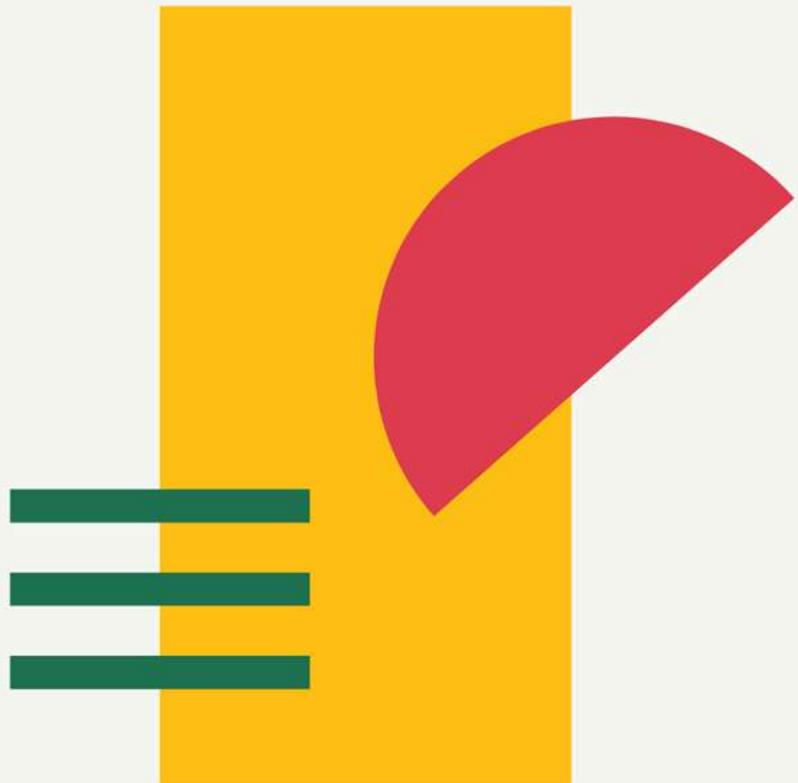
Floated items will always rise to the top. This means that we can use floats to create multiple columns.

## Card 6

Floated items will always rise to the top. This means that we can use floats to create multiple columns.

# **Se trabajó con display: table**

Los valores se pueden ampliar a otros elementos de HTML diferentes a <tr> y <td> como los <div>, <ul>, etc.



**Existen una gran  
cantidad de técnicas  
que son simplemente  
TRUCOS**

**Gracias a eso, el  
diseño con CSS  
parece difícil y frágil:  
No habían  
herramientas de  
diseño**

# Herramientas que nos han facilitado el camino

**La  
comunidad  
desarrolló  
herramientas  
para facilitar  
el camino**



# Objetivo

## EVALUAR EL PANORAMA ACTUAL

Ya que las herramientas tienen un impacto en la forma en la que diseñamos y desarrollamos

# Arquitecturas

# Pre y Post procesadores

# Diseño de componentes

(Como Atomic Design)

# Frameworks

# Performance

# Accesibilidad

# **Evergreen Browsers**



**Aumentemos  
nuestra comprensión  
del por qué suceden  
ciertas cosas**

# ¿CSS Grid es una idea nueva?

La evolución de la especificación



No debería sorprendernos que los diseños basados en cuadrículas hayan sido un objetivo de CSS desde el principio.



"CSS comenzó como algo muy simple", recordó Bos.



"Era solo una forma de crear una vista de un documento en una pantalla pequeña muy simple en ese momento..."

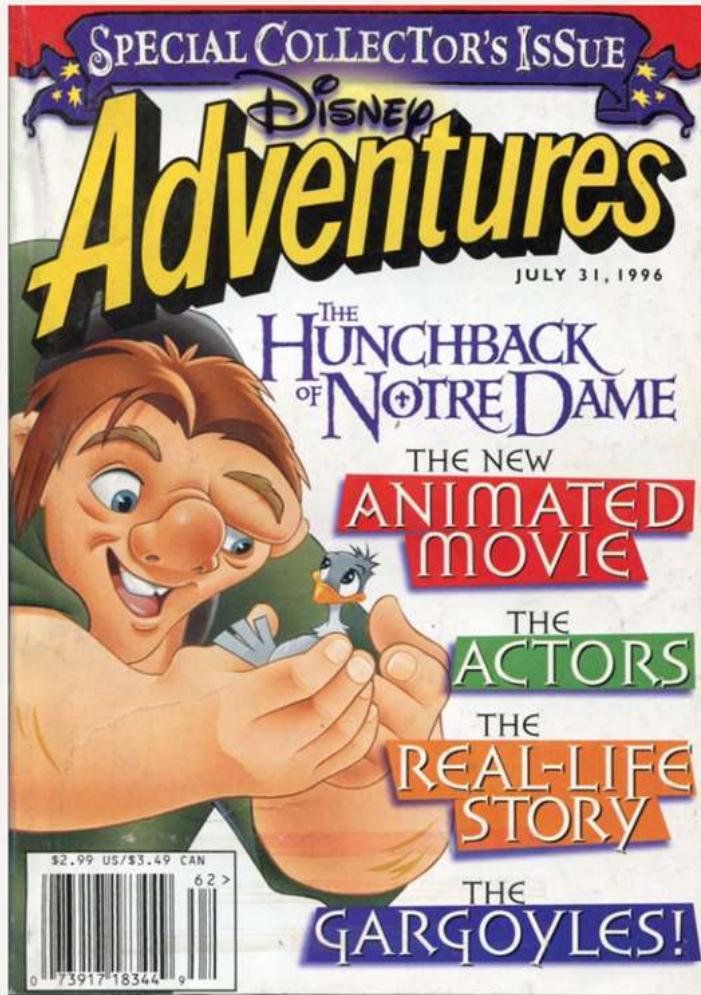


"...Hace veinte  
años, las  
pantallas eran  
muy pequeñas.  
Entonces,  
cuando vimos  
que podíamos  
hacer una hoja  
de estilo para  
documentos,  
pensamos:..."



"...Bueno, ¿qué más podemos hacer ahora que tenemos un sistema para hacer hojas de estilo?"

Observar lo que hacían los libros y las revistas con el diseño fue una gran inspiración para ellos.



## CSS Advanced Layout Module

W3C Working Draft 9 August 2007

This version:

<http://www.w3.org/TR/2007/WD-css3-layout-20070809>

Latest version:

<http://www.w3.org/TR/css3-layout/>

Previous version:

<http://www.w3.org/TR/2005/WD-css3-layout-20050816>

Editors:

Bert Bos (W3C) [bert@w3.org](mailto:bert@w3.org)

▼ collapse

**This version is outdated!**

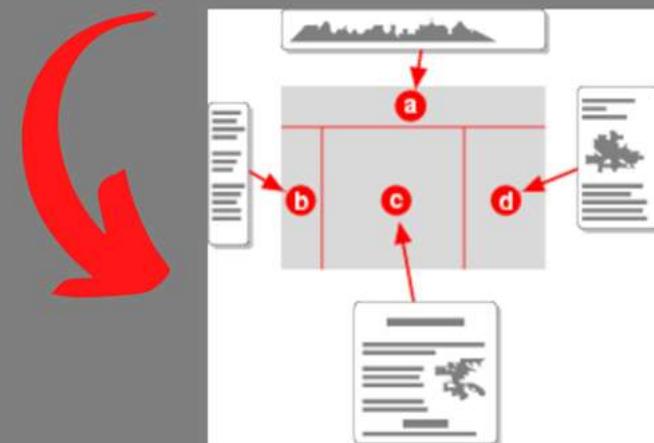
For the latest version, please look at <https://www.w3.org/TR/css-template-3/>.

Copyright © 2007 W3C® (MIT, ERCIM, Keio). All Rights Reserved. W3C [liability](#), [trademark](#) and [document use](#) rules apply.

## Abstract

CSS is a simple, declarative language for creating style sheets that specify the rendering of HTML and other structured documents. This specification is part of *level 3 of CSS* ("CSS3") and contains features to describe layouts at a high level, meant for tasks such as the positioning and alignment of "widgets" in a graphical user interface or the layout grid for a page or a window, in particular when the desired visual order is different from the order of the elements in the source document. Other CSS3 modules contain properties to specify fonts, colors, text alignment, list numbering, tables, etc.

The features in this module are described together for easier reading, but are usually not implemented as a group. CSS3 modules often depend on other modules or contain features for several media types. Implementers should look at the various "profiles" of CSS, which list consistent sets of features for each type of media.



# 2016

## TABLE OF CONTENTS

1. **Introduction**
  - 1.1. Summary and use cases
  - 1.2. Dependencies on other modules
  - 1.3. Values
  - 1.4. A note about accessibility
2. **Stack of cards layout**
3. **Declaring templates**
  - 3.1. Declaring a template: 'grid-template-areas'
  - 3.2. Specifying the widths of columns: 'grid-template-columns'
  - 3.3. Specifying the height of rows: 'grid-template-rows'
  - 3.4. The 'grid' and 'grid-template' shorthand properties
  - 3.5. Default slots
  - 3.6. Interaction of 'grid' and 'columns'
  - 3.7. Calculating the size of the grid
4. **Flowing content into slots: 'flow'**
5. **Comparison with 'display: grid' and 'display: inline-grid'**
6. **Styling slots**
  - 6.1. The '::slot()' pseudo-element
  - 6.2. The '::blank()' pseudo-element
  - 6.3. Slots and the 'content' property
7. **Styling the contents of slots**
8. **Rendering of grid elements**

## CSS Template Layout Module

Editor's Draft 1 March 2016



**This version:**

<http://dev.w3.org/csswg/css-template-3/>

**Latest version:**

<https://www.w3.org/TR/css-template-3/>

**Previous version:**

<https://www.w3.org/TR/2015/NOTE-css-template-3-20150326/>

**Feedback:**

[www-style@w3.org](mailto:www-style@w3.org) with subject line "[css-template-3] ... message topic ..." ([archives](#))

**Editors:**

Bert Bos (W3C) [bert@w3.org](mailto:bert@w3.org)

César Acebal (University of Oviedo)

**Editors' draft:**

<https://drafts.csswg.org/css-template/>

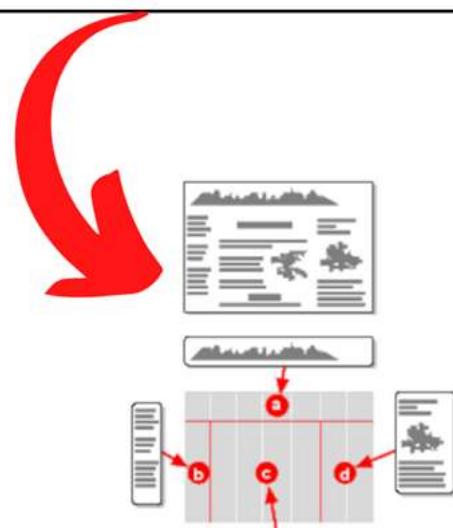
Copyright © 2016 W3C® (MIT, ERCIM, Keio, Beihang). W3C liability, trademark and document use rules apply.

### Abstract

CSS is a simple, declarative language for creating style sheets that specify the rendering of HTML and other structured documents. This Note contains experimental ideas for *layout templates* and alternative layout models in CSS.

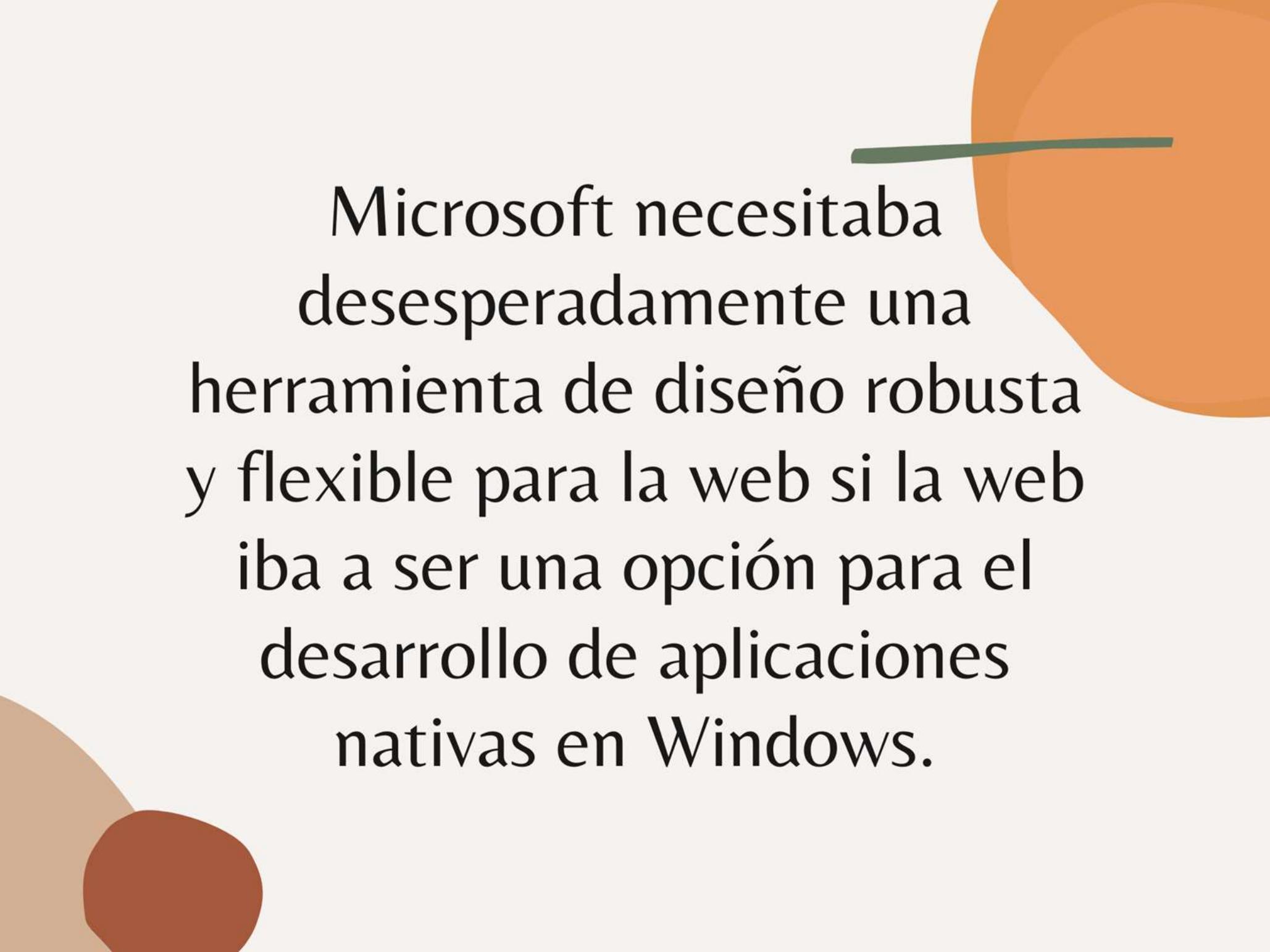
At the moment, it defines a *typographic grid* for CSS. It has features to set up a grid-based template, to style the *slots* of the template and to flow content into them.

A grid template can be seen as a cross between table

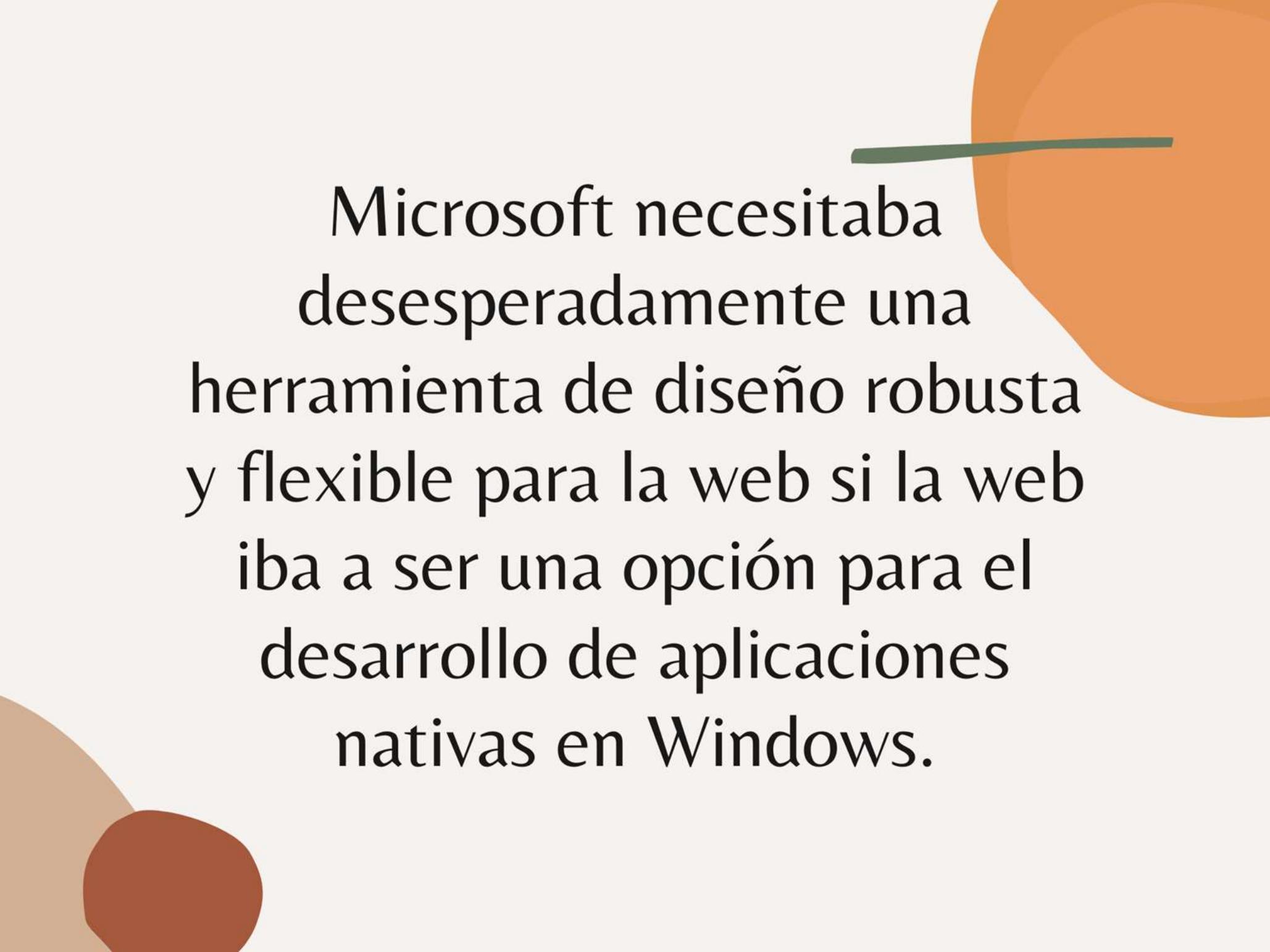


¿Por qué surgió la necesidad de trabajar en esta especificación?





Microsoft necesitaba desesperadamente una herramienta de diseño robusta y flexible para la web si la web iba a ser una opción para el desarrollo de aplicaciones nativas en Windows.



Microsoft necesitaba desesperadamente una herramienta de diseño robusta y flexible para la web si la web iba a ser una opción para el desarrollo de aplicaciones nativas en Windows.



# Rachel Andrew

"Queremos esto  
para la web"



18 December 2012

Published in [Code](#)

7 comments

# Giving Content Priority with CSS3 Grid Layout

Rachel Andrew

Browser support for many of the modules that are part of CSS3 have enabled us to use CSS for many of the things we used to have to use images for. The rise of mobile browsers and the concept of responsive web design has given us a whole new way of looking at design for the web. However, when it comes to layout, we haven't moved very far at all. We have talked for years about separating our content and source order from the presentation of that content, yet most of us have had to make decisions on source order in order to get a certain visual layout.

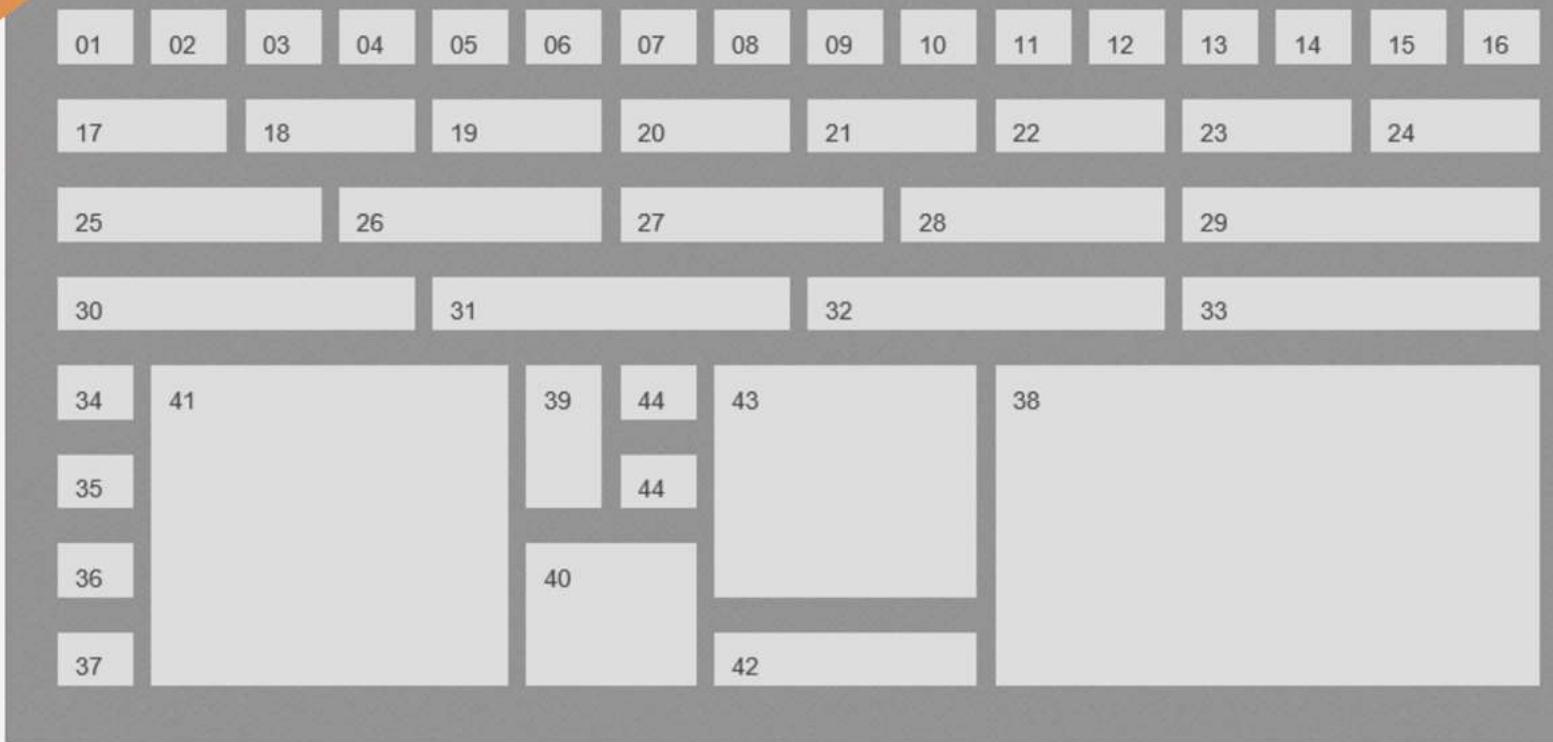
Owing to some interesting specifications making their way through the W3C process at the moment, though, there is hope of change on the horizon. In this article I'm going to look at one CSS module, the [CSS3 grid layout module](#), that enables us to define a

```
.wrapper {  
    display: -ms-grid;  
    -ms-grid-columns: 200px 20px auto 20px 200px;  
    -ms-grid-rows: auto 1fr;  
}
```

```
.content {  
    -ms-grid-column: 3;  
    -ms-grid-row: 2;  
    -ms-grid-row-span: 1;  
}
```

```
.wrapper {  
    width: 90%;  
    margin: 0 auto 0 auto;  
    display: -ms-grid;  
    -ms-grid-columns: 1fr (4.25fr 1fr)[16];  
    -ms-grid-rows: (auto 20px)[24];  
}
```

# Flexible 960 Grid



# Welcome!

If you have a curious obsession with Nativity sets, then you are in the right place.

Our collection contains more crazy looking Nativity sets than you could shake a tinsel covered stick at.

## About Nativity Sets

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed euismod lacinia sodales. Etiam lobortis neque id nunc semper auctor. Nam sagittis, enim sed dignissim faucibus, neque magna fermentum neque, venenatis ullamcorper risus dolor at metus. Curabitur in neque est. Integer fringilla gravida dolor, a consequat erat blandit sed. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Nunc eleifend leo ut lectus malesuada elementum. Donec sed neque nisi.

## History of the Museum

Morbi tincidunt odio in mi mattis faucibus vel ac turpis. Quisque feugiat nibh quis sapien bibendum eleifend. Phasellus at molestie arcu. Nunc at urna condimentum nisi ultricies rhoncus. Aliquam feugiat arcu vel risus bibendum at venenatis nunc mollis. Mauris aliquet dictum leo, sed tristique sapien luctus eu. Praesent ultricies nulla eget turpis tincidunt suscipit.

## The Museum Today

Nullam iaculis, magna a dignissim dictum, quam diam porta eros, sed luctus purus velit in eros. Morbi ullamcorper nunc eu lorem placerat lobortis. Sed condimentum sagittis nunc id imperdiet. Etiam pharetra sodales lorem, et tempus lectus vestibulum elementum. In viverra tempor magna non laoreet. Proin a augue eros, sit amet malesuada odio. Cras tellus tortor, laoreet id laoreet sit amet, volutpat accumsan metus. Nulla dapibus ante ut nisi vulputate ultrices. In hac habitasse platea dictumst. Aenean lectus massa, congue in malesuada eget, mollis vitae urna. Aenean neque nunc, laoreet id convallis a, interdum et arcu. Etiam aliquam euismod sem, et feugiat nunc lobortis id.

## Visit Us

Opening times: 9am - 5pm throughout Advent.

**The Nativity Museum**  
1 Christmas Street, Tinsel  
Town, United Kingdom



## Welcome!

If you have a curious obsession with Nativity sets, then you are in the right place.

Our collection contains more crazy looking Nativity sets than you could shake a tinsel covered stick at.

### Visit Us

Opening times: 9am - 5pm throughout Advent.

**The Nativity Museum**  
1 Christmas Street, Tinsel Town, United Kingdom

### About Nativity Sets

Lorum ipsum dolor sit amet, consectetur adipiscing elit. Sed euismod lacinia sodales. Etiam lobortis neque id nunc semper auctor. Nam sagittis, enim sed dignissim faucibus, neque magna fermentum neque, venenatis ullamcorper risus dolor at metus. Curabitur in neque est. Integer fringilla gravida dolor, a consequat erat blandit sed. Class aptent taciti sociosqua ad litora torquent per conubia nostra, per inceptos himenaeos. Nunc eleifend leo ut lectus malesuada elementum. Donec sed neque nisi.

### History of the Museum

Morbi tincidunt odio in mi mattis faucibus vel ac turpis. Quisque feugiat nibh quis sapien bibendum eleifend. Phasellus at molestie arcu. Nunc at urna condimentum nisi ultricies rhoncus. Aliquam feugiat arcu vel risus bibendum at venenatis nunc mollis. Mauris aliquet dictum leo, sed tristique sapien luctus eu. Praesent ultricies nulla eget turpis tincidunt suscipit.

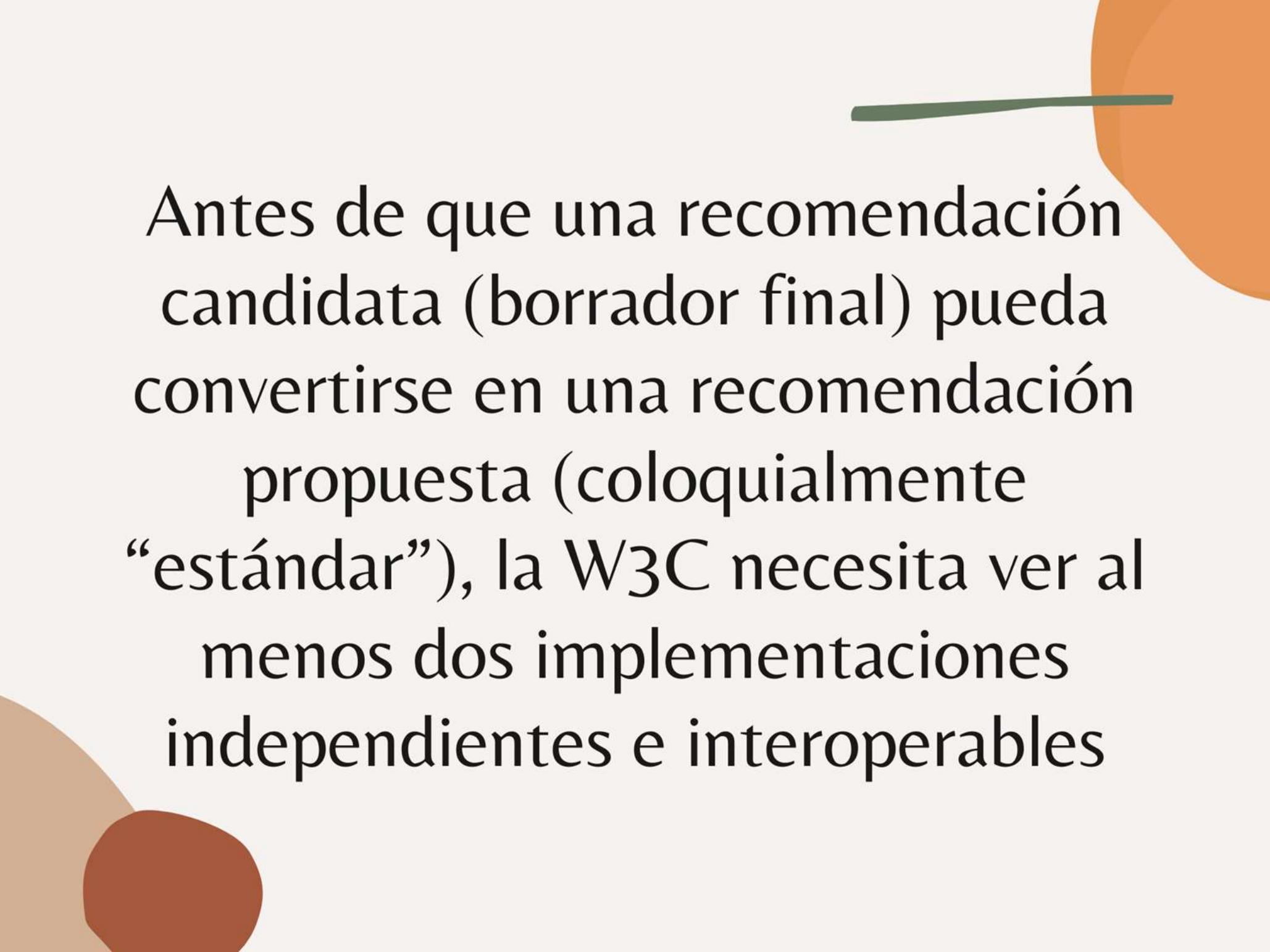
### The Museum Today

Nullam iaculis, magna a dignissim dictum, quam diam porta eros, sed luctus purus velit in eros. Morbi ullamcorper nunc eu lorem placerat lobortis. Sed condimentum sagittis nunc id imperdiet. Etiam pharetra sodales lorem, et tempus lectus vestibulum elementum. In viverra tempor magna non laoreet. Proin a augue eros, sit amet malesuada odio. Cras tellus tortor, laoreet id laoreet sit amet, volutpat accumsan metus. Nulla dapibus ante ut nisi vulputate ultrices. In hac habitasse platea dictumst. Aenean lectus massa, congue in malesuada eget, mollis vitae urna. Aenean neque nunc, laoreet id convallis a, interdum et arcu. Etiam aliquam euismod sem, et feugiat nunc lobortis id.



# Hubo 3 ideas fundamentales

- Idea de Microsoft
- Diseño Avanzado de Bos
- Adición de líneas de cuadrícula de Linss



Antes de que una recomendación candidata (borrador final) pueda convertirse en una recomendación propuesta (coloquialmente “estándar”), la W3C necesita ver al menos dos implementaciones independientes e interoperables

# Establishing Web Standards



Oliver Lindberg

Follow

Mar 31, 2016 · 3 min read

[Twitter](#) [LinkedIn](#) [Facebook](#) [Bookmark](#)

Aaron Gustafson explains how an idea becomes a specification at the World Wide Web Consortium (W3C)

When we're busy building stuff for the web, we don't get much time to ponder how the HTML tags we write and CSS properties we use come to be. I'm here to shed a little light on the way our standards become standards.

First off, the 'standards' created by the W3C aren't really *standards*, but are rather a collection of specifications that instruct browsers on how to implement certain language features, so when we use a certain HTML element, browsers display it in pretty much the same way. The term 'standards' isn't used by the W3C; what we think of as 'standards' are actually 'recommendations'. The term 'web standards' actually came from the Web Standards Project. Surprising, I know.

## The Idea Stage

Every new standard — ahem, recommendation — starts as an idea. That idea can come from the W3C, a browser vendor or other company, or folks like you and me. The W3C's activities are organised into Working Groups, all of which function in their own unique way. Depending on where the idea fits in, it could surface on a mailing list, IRC, in a face-to-face, or possibly even on GitHub. From there, the Group discusses the idea and may opt to formalise it into a specification (also known as a 'spec').

## Working Draft

The Working Draft (WD) is an initial stab at turning the idea into a spec. It



Hubo un cambio  
fundamental con  
CSS Grid

A close-up portrait of Jen Simmons, a woman with long, wavy brown hair and blue eyes, wearing dark blue-rimmed glasses and a wide smile. She is positioned on the left side of the slide, with a large orange abstract shape above her head and a large green abstract shape below her chin.

# Jen Simmons

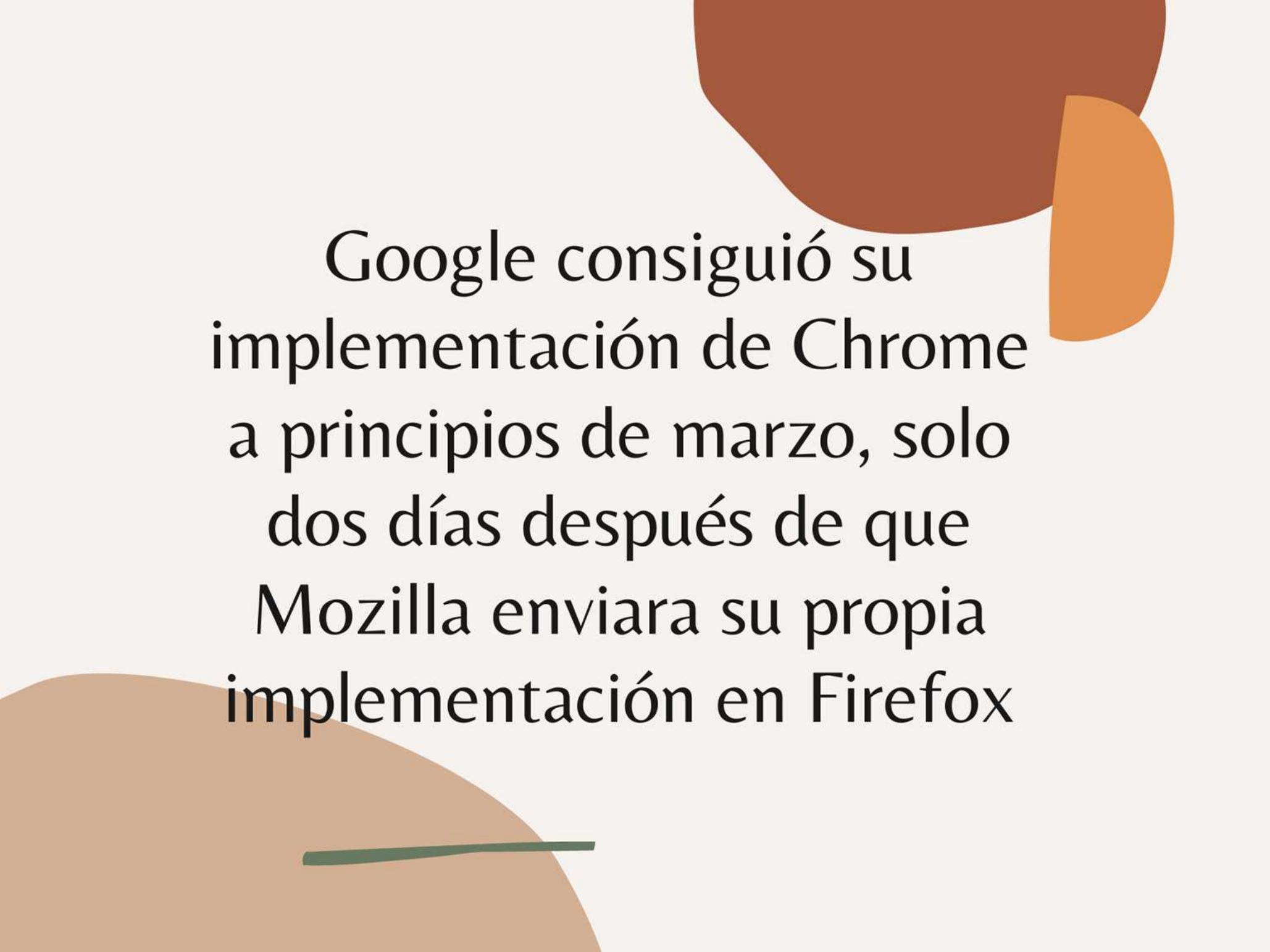
Colocó muchas  
demostraciones  
que creó para CSS  
Grid en la web



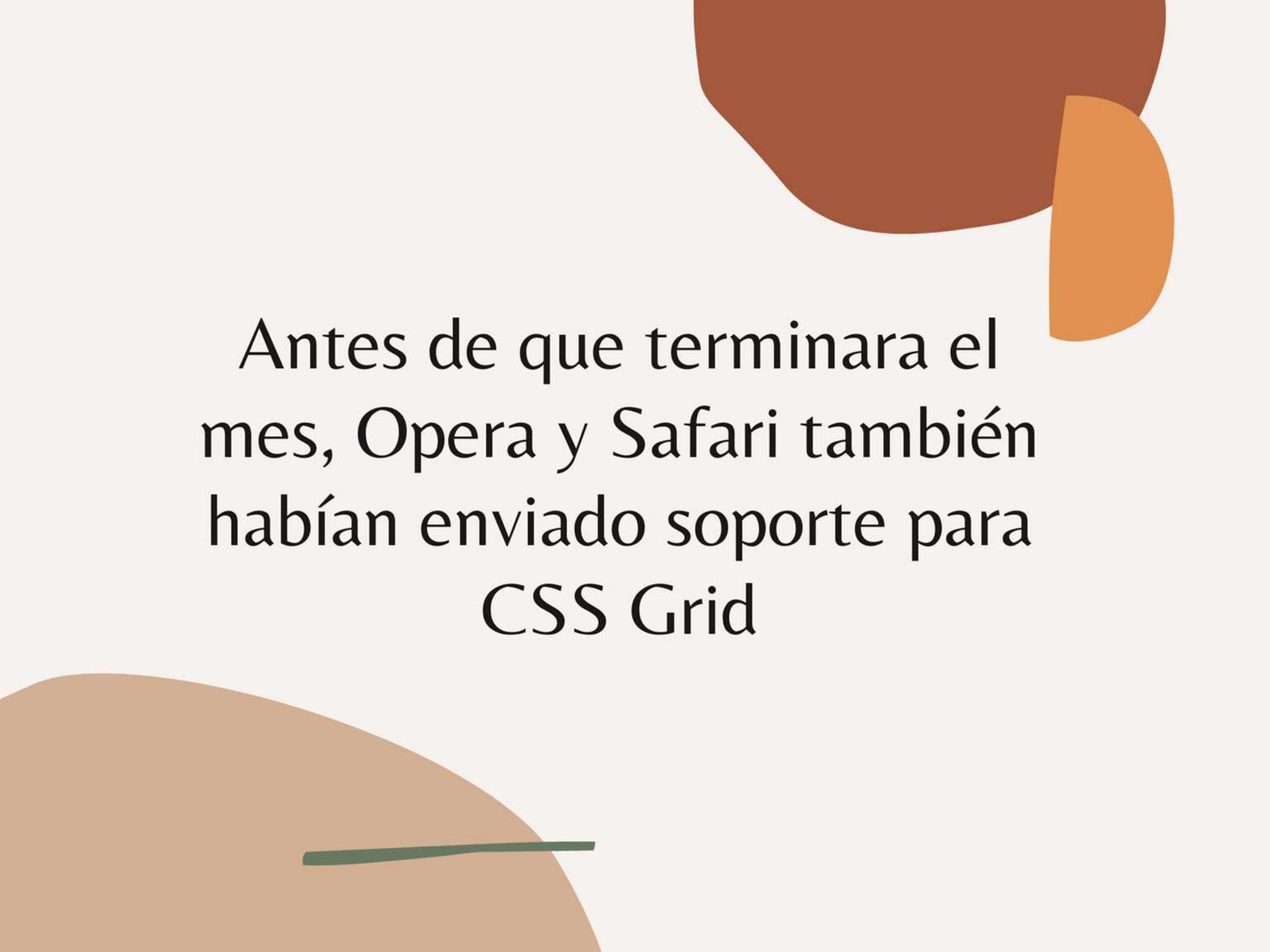
Sin el entusiasmo de los desarrolladores, los proveedores de navegadores son reacios a gastar dinero para ver si la idea gana terreno



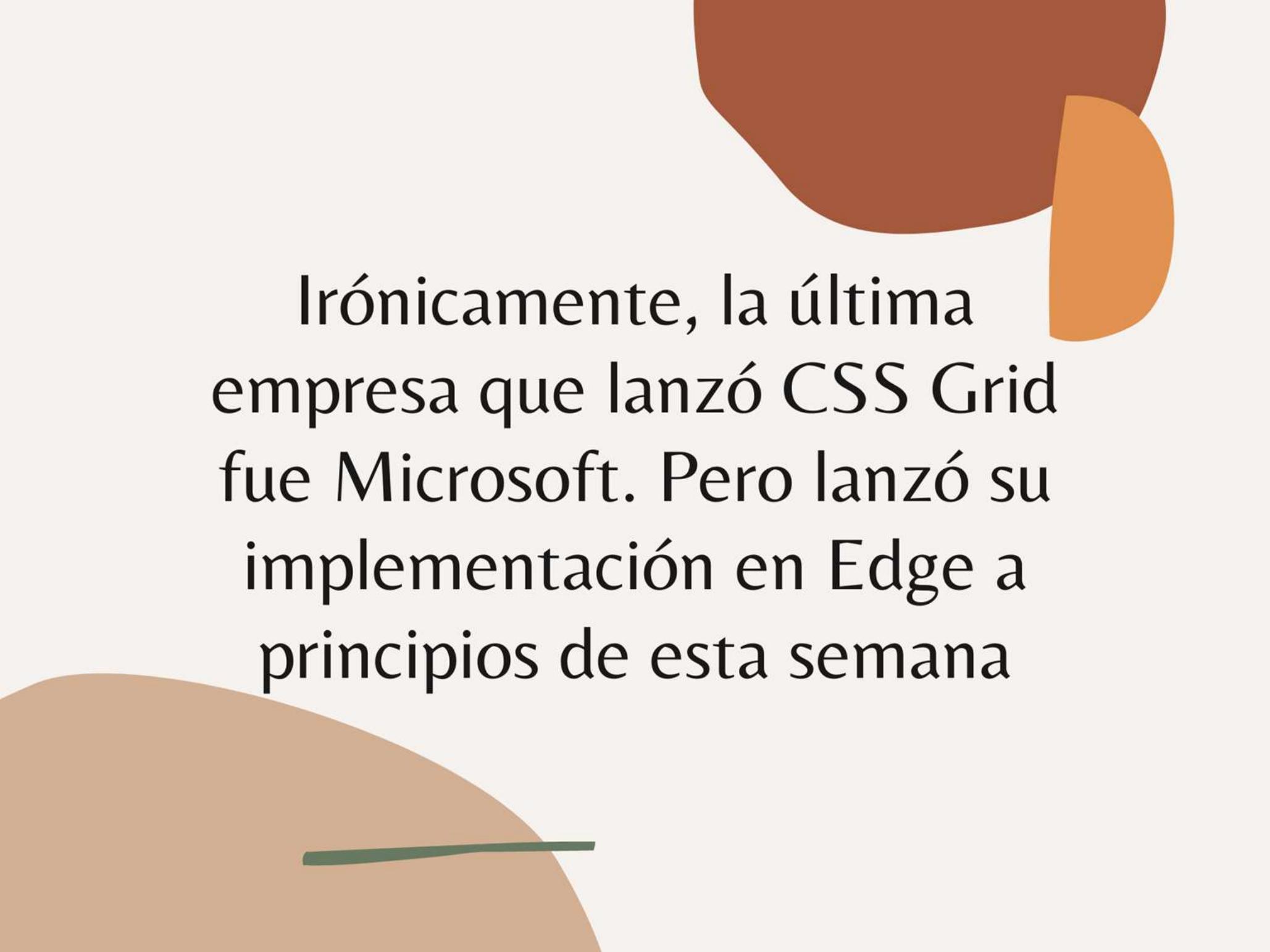
Google consiguió una implementación de CSS Grid en Chromium 56 para Android en enero de 2017



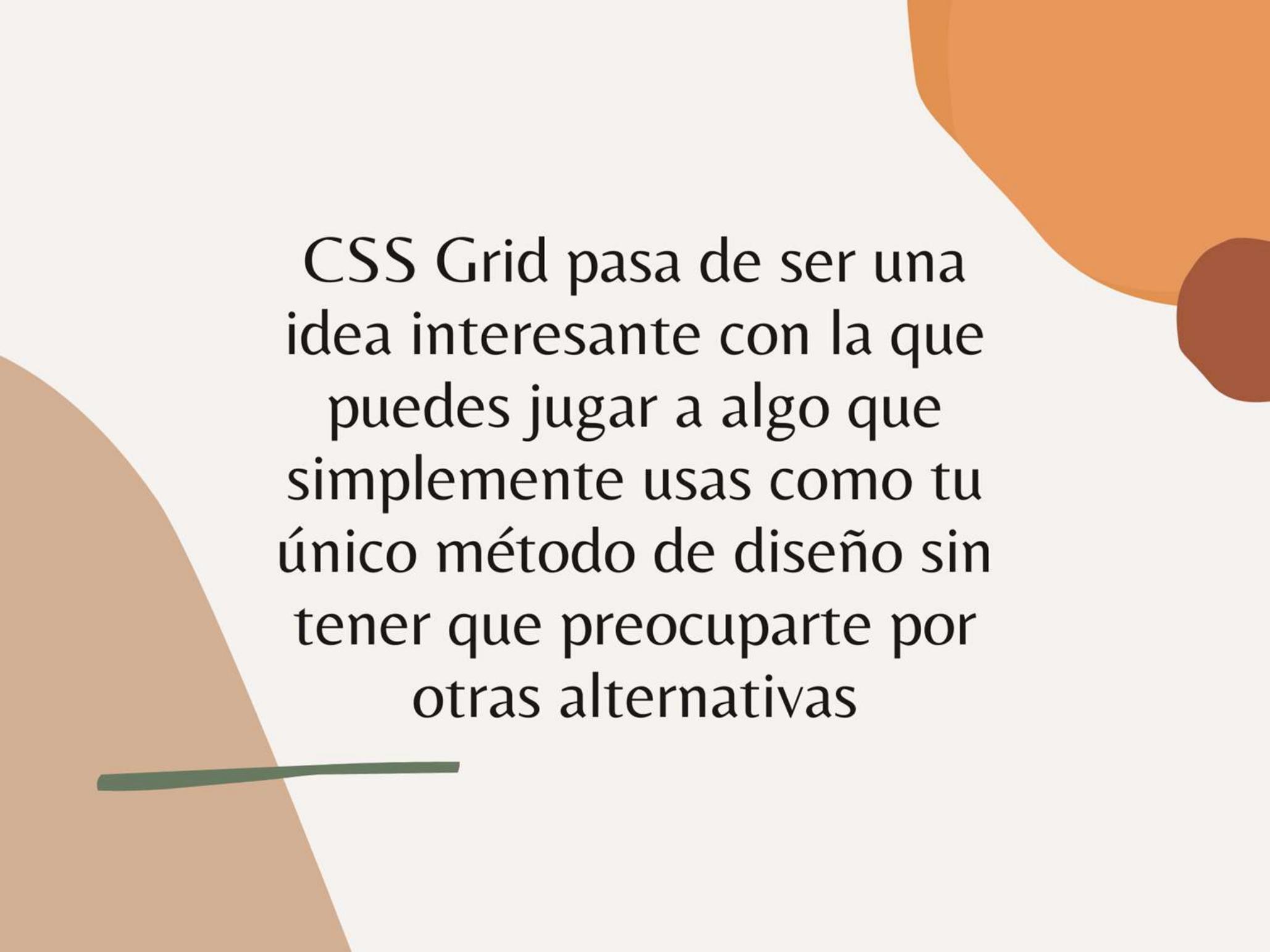
Google consiguió su implementación de Chrome a principios de marzo, solo dos días después de que Mozilla enviara su propia implementación en Firefox



Antes de que terminara el mes, Opera y Safari también habían enviado soporte para CSS Grid



Irónicamente, la última empresa que lanzó CSS Grid fue Microsoft. Pero lanzó su implementación en Edge a principios de esta semana



CSS Grid pasa de ser una idea interesante con la que puedes jugar a algo que simplemente usas como tu único método de diseño sin tener que preocuparte por otras alternativas

---

...

# ¿Qué significa Grid para CSS?

01



**CSS Grid requiere una  
forma completamente  
nueva de pensar sobre  
el diseño en CSS**

# Bert Bos :

**"No se trata simplemente de agregar propiedades a cada elemento individual. Ahora puedes tener un modelo diferente en el que comienzas con tu diseño primero y luego incorporas los diferentes elementos en ese diseño"**

# Atkins :

**"Es la herramienta de  
diseño más poderosa que  
hemos inventado hasta  
ahora para CSS"**

# Etemad :

"CSS Grid toma todas esas cosas complicadas que tuvimos que hacer para lograr diseños básicos y lo hace completamente innecesario"

# Whitworth :

**"Estoy entusiasmado con el futuro del diseño CSS.**

**CSS Grid no es el final; en realidad es solo el comienzo"**

Conozcamos el  
panorama completo de  
la alineación con CSS  
para poder entrar en  
materia con CSS Grid



# Técnicas de alineamiento antes de CSS Grid (Parte 1)



Veremos 4  
técnicas de  
alineamiento

- margin
- line-height
- table-cell
- positions

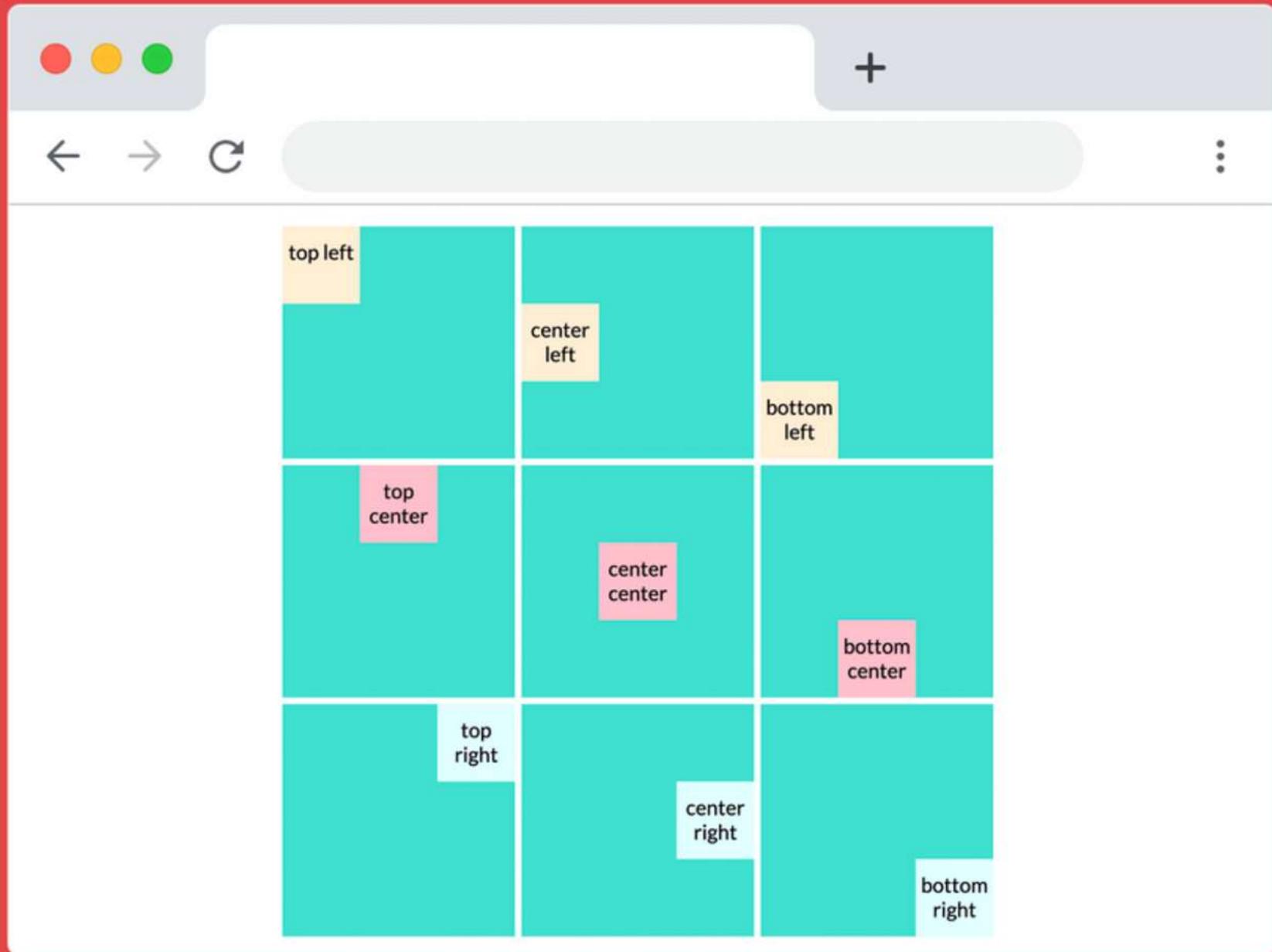


En esta clase  
veremos  
estas dos:

- margin
- line-height
- table-cell
- positions

# IMPORTANTE

Conocer muy bien  
las propiedades que  
necesitamos para  
cada técnica :)



# Técnica margin

# Propiedades que necesitamos



- margin-top
- margin-right
- margin-bottom
- margin-left



# margin



*/\* Individual margin properties \*/*

```
margin-top: 10px;  
margin-right: 15px;  
margin-bottom: 20px;  
margin-left: 25px;
```

*/\* Shorthand property \*/*

```
margin: 10px 15px 20px 25px;
```



**Yo: Vamos al código !  
Mi página web:**



# Técnica line-height

# Propiedades que necesitamos



- text-align
- vertical-align
- line-height



# text-align

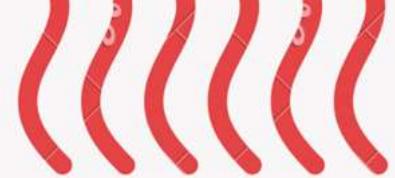


```
text-align: left|right|center|justify|initial|inherit;
```



horizontal



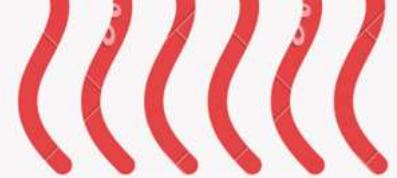


# vertical-align



```
vertical-align: baseline | length | sub | super |  
top | text-top | middle | bottom |  
text-bottom | initial | inherit;
```





# line-height



```
line-height: normal|number|length|initial|inherit;
```



**Yo: Vamos al código !  
Mi página web:**



# Técnicas de alineamiento antes de CSS Grid (Parte 2)



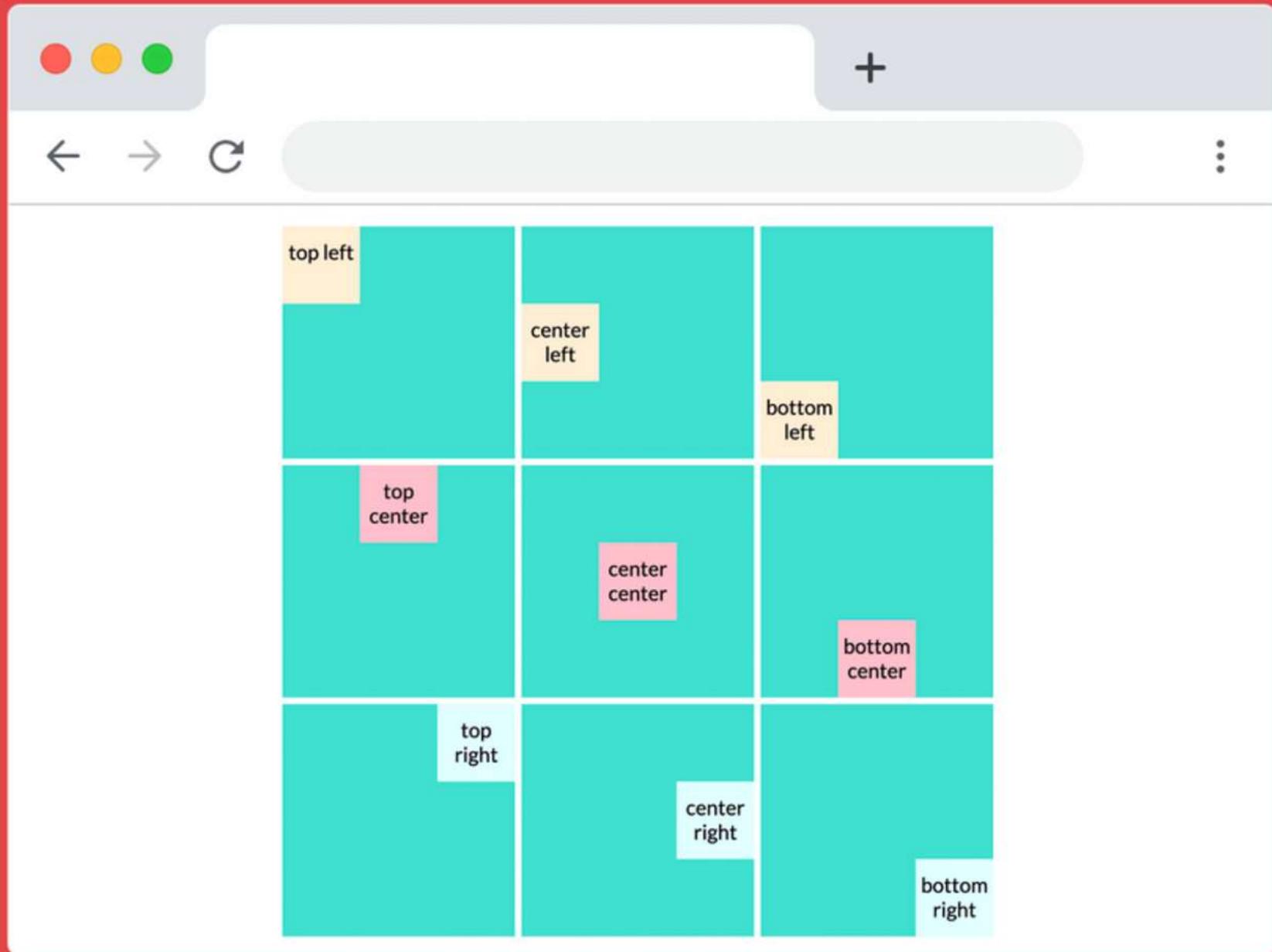
Veremos 4  
técnicas de  
alineamiento

- margin
- line-height
- table-cell
- positions



En esta clase  
veremos  
estas dos:

- margin
- line-height
- table-cell
- positions



# IMPORTANTE

Conocer muy bien  
las propiedades que  
necesitamos para  
cada técnica :)

# Técnica table-cell

# Propiedades que necesitamos



- display: table-cell
- text-align
- vertical-align



# display: table-cell



```
display: table-cell;
```

*/\* Hace que cualquier elemento que deseas se  
comporte como si fuera un elemento de tabla \*/*





# display: table-cell



```
•••  
display: table          /* <table>      */  
display: table-cell    /* <td>        */  
display: table-row      /* <tr>        */  
display: table-column    /* <col>        */  
display: table-column-group/* <colgroup>   */  
display: table-footer-group/* <tfoot>     */  
display: table-header-group/* <thead>     */
```





# HTML

```
<table>
  <thead>
    <tr>
      <th>Estudiante</th>
      <th>ID</th>
      <th>Clase favorita</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>Tomás</td>
      <td>00001</td>
      <td>Diseño responsivo</td>
    </tr>
    <tr>
      <td>Sue</td>
      <td>00002</td>
      <td>Accesibilidad</td>
    </tr>
  </tbody>
</table>
```

A screenshot of a web browser window displaying the rendered HTML table. The browser has a light gray header bar with three colored dots (red, yellow, green) on the left and a '+' button on the right. Below the header is a navigation bar with back, forward, and search/clear buttons. The main content area shows a table with three columns: 'Estudiante', 'ID', and 'Clase favorita'. The data rows correspond to the entries in the code: Tomás (ID 00001) and Sue (ID 00002). The table has a thin gray border and is centered on the page.

Estudiante	ID	Clase favorita
Tomás	00001	Diseño responsivo
Sue	00002	Accesibilidad



**Yo: Vamos al código !  
Mi página web:**



# Técnica positions

# Propiedades que necesitamos



- position:  
relative
- position:  
absolute
- top
- right
- bottom
- left
- transform:  
translate()



# positions

static    relative    absolute    fixed

Posicionado de acuerdo  
al flujo normal



Su posición final la determinan  
top, right, bottom, y left



Crea un nuevo contexto  
de apilamiento

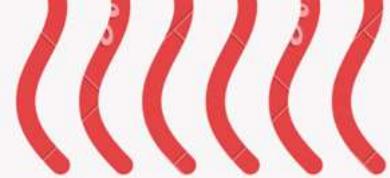


(z-index != auto)

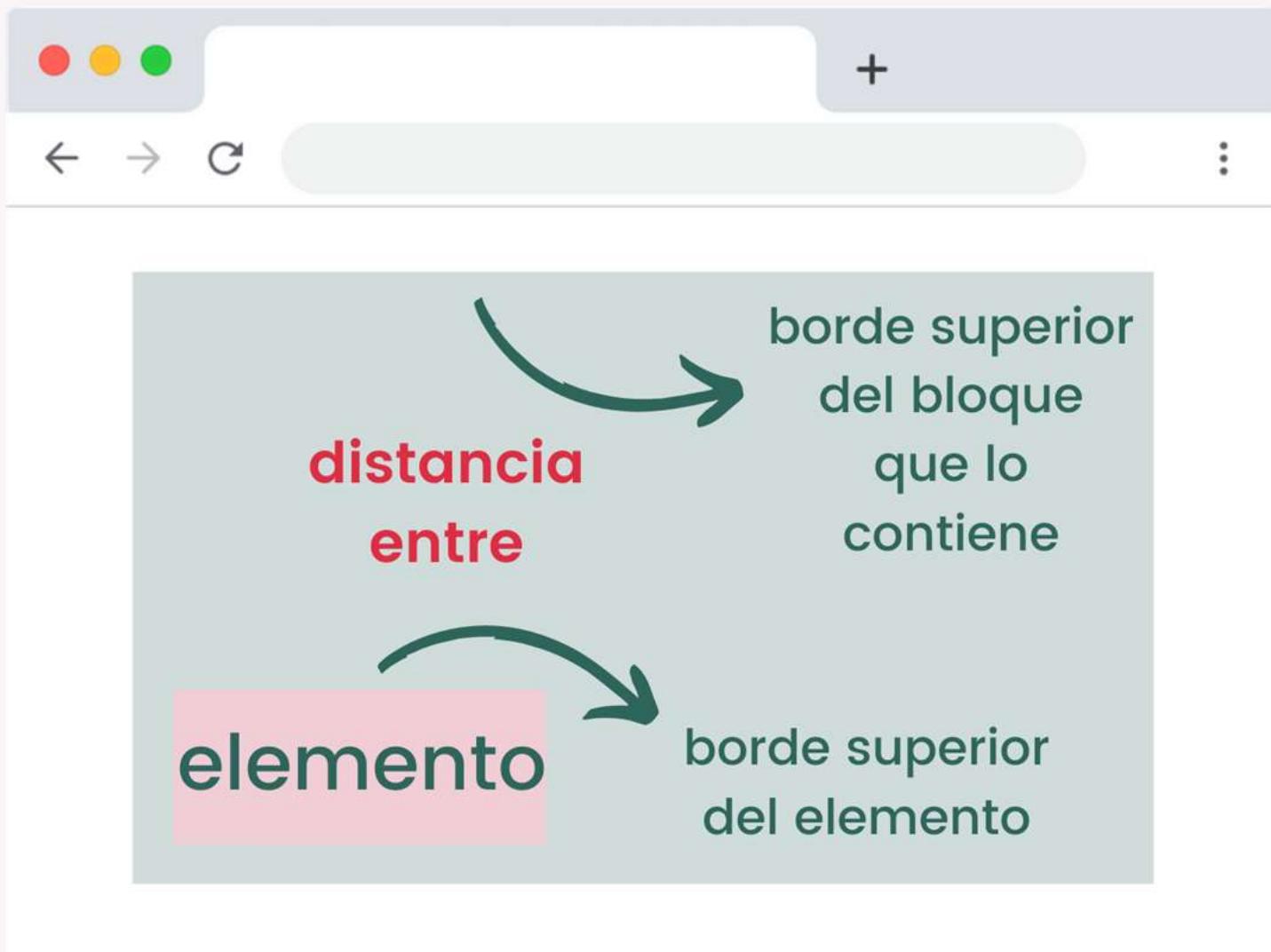
(z-index != auto)



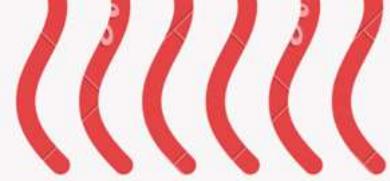
# top



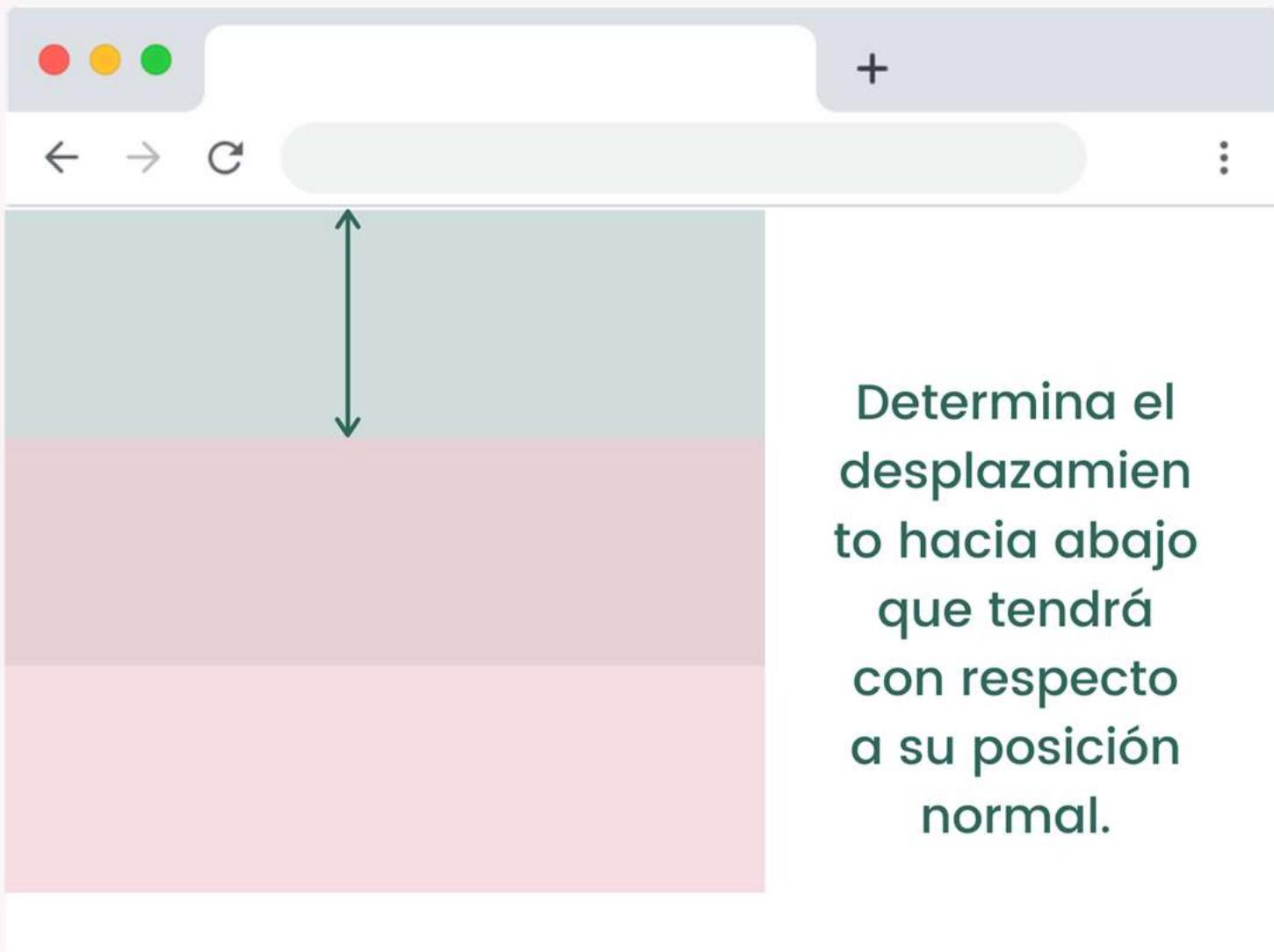
position:  
absolute

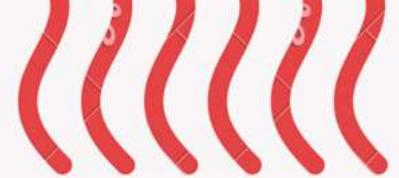


# top



position:  
relative





# top · sintaxis



```
top: <longitud> | <porcentaje> | auto | inherit;
```



px, em...

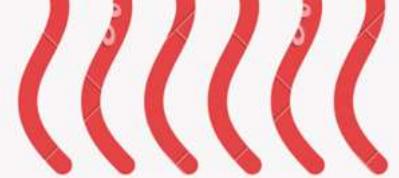


%



valor por hereda  
defecto del  
padre





# transform: translate()



```
transform: none|transform-functions|initial|inherit;
```



```
translate(x,y)
```



**Yo: Vamos al código !  
Mi página web:**



# PROS Y CONTRAS DE LAS TÉCNICAS DE ALINEAMIENTO ANTES DE CSS GRID



“

No hay ninguna  
propiedad  
específica  
para alinear  
**elementos de  
bloque en CSS2**

...

LO QUE HICIMOS  
EN LAS DOS  
CLASES  
ANTERIORES

•••

NO DEJAN DE SER  
TRUCOS PARA  
ALINEAR ELEMENTOS  
DE BLOQUE

GRACIAS A ESTOS  
TRUCOS, MUCHAS  
PERSONAS LE HUYEN  
A TRABAJAR CON CSS  
Y PIENSAN QUE HOY  
EN DÍA SEGUIMOS  
TRABAJANDO CON  
TRUCOS DE MAGIA

...

**Ventaja:** El valor **auto** alinea horizontalmente cualquier elemento con cualquier ancho.

**MARGIN**

**Desventaja:** Alinear verticalmente, ya que, en cada caso, deberán calcularse estos valores.

**Ventaja:** Correcta  
alineación.

LINE-  
HEIGHT

**Desventaja:** Si el texto  
ocupa más de una línea el  
elemento toma un alto más  
grande que lo necesario  
para los cálculos

**Ventaja:** La alineación vertical no está condicionada por fuentes, tamaños de fuentes o alturas de línea.

TABLE-  
CELL

**Desventaja:** vertical-align se aplica sólo a elementos Inline



<https://www.wextensible.com/>



Inicio

Menú

Página Accesos

Pie

07  
ENE  
2016

## Alinear elementos de bloque en CSS3

Alineación vertical con CSS3

### 1. Diversas formas de alinear elementos de bloque en CSS2 y CSS3

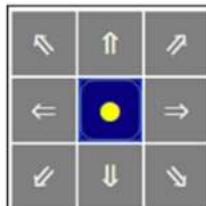


Figura 1. Nueve posiciones de alineado con CSS2 y CSS3.

No hay ninguna propiedad específica para alinear **elementos de bloque** en CSS2. Se han venido usando diversas técnicas como la del valor **auto** para los **márgenes horizontales** que nos permite alinear en esa dirección. Otra forma es dar al elemento de bloque hijo el valor **inline-block** en su propiedad **display**. Así podemos dimensionarlo y obedecerá la alineación de elementos *Inline* que vimos en temas anteriores. Usando **text-align** para el bloque padre y **vertical-align** para el bloque hijo conseguimos, con ciertas limitaciones, alinear en ambas direcciones.

Una tercera forma de alinear con CSS2 es usando el valor **table-cell** para la propiedad **display**. El elemento de bloque se comportará como una celda de tabla, un elemento en CSS2 sobre el que se puede aplicar al mismo tiempo las propiedades **text-align** y **vertical-align**.

Lo anterior no deja de ser otra cosa que trucos para alinear elementos de bloque. Porque CSS3 finalmente deberá incorporar propiedades específicas para este cometido. La especificación en fase de borrador [CSS Box Alignment Module Level 3](#), documento del

HTML-5

CSS-3

JavaScript

PHP

Servidor

Layout

Componentes

Herramientas

Optimización

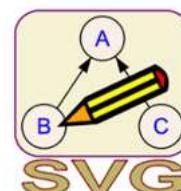
Varios

XHTML+CSS

[Ver en página](#)

Nuevo

Selectores CSS  
nivel 4



# LA MAYOR LIMITANTE

PROPIEDADES  
FÍSICAS:

- margin-top
- padding-bottom
- border-right
- left

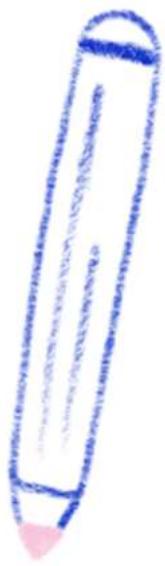
•••

MODOS DE  
ESCRITURA

PROPIEDADES  
LÓGICAS



WRITING  
MODES





LEFT

TOP

RIGHT

BOTTOM

# TOP-LEFT

ORIGIN

PADDING

MARGIN

BORDER



# WRITING MODE

PROPERTY

1 LEFT TO RIGHT

LATIN

2 RIGHT TO LEFT

ARABIC

3 BI-DIRECTIONAL

LATIN + ARABIC

VERTICAL

ASIAN CHARAC.

HORIZONTAL TB

VERTICAL LR

VERTICAL RL

SIDEWAYS LR

SIDEWAYS RL

No

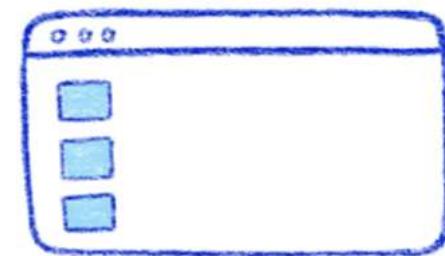
TABLES

- ROW
- COLUMN



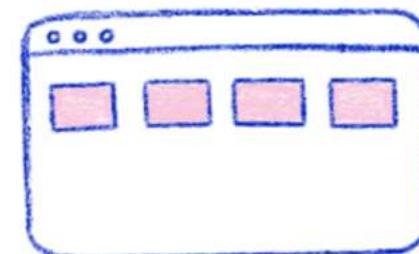
# BLOCK FLOW DIRECTION

DISPLAY: BLOCK



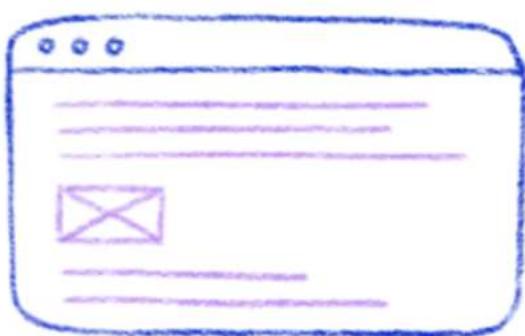
# INLINE BASE DIRECTION

DISPLAY: iNLINE

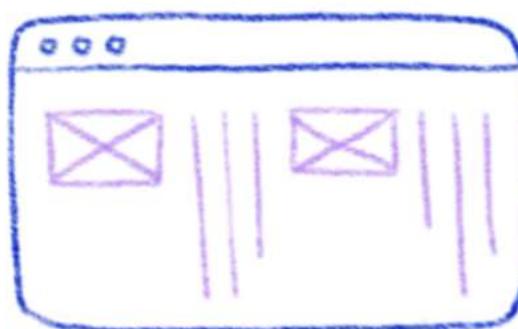


# :VALUES:

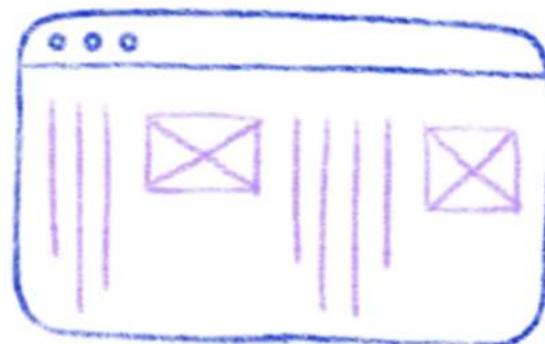
HORIZONTAL TB



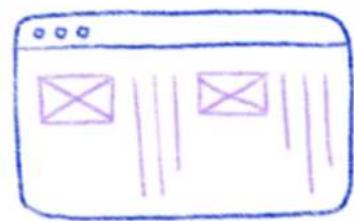
VERTICAL RL



VERTICAL LR

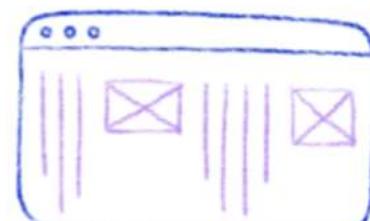


VERTICAL RL

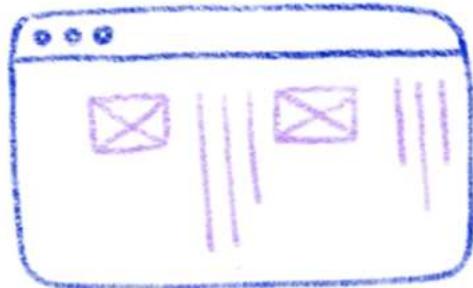


# : VALUES :

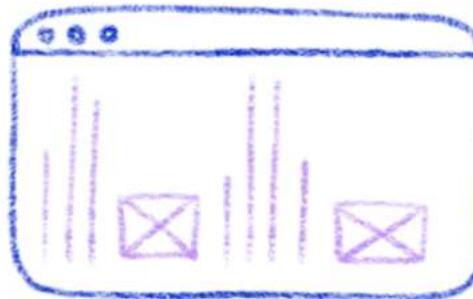
VERTICAL LR



SIDEWAYS RL



SIDEWAYS LR



# CAN I USE

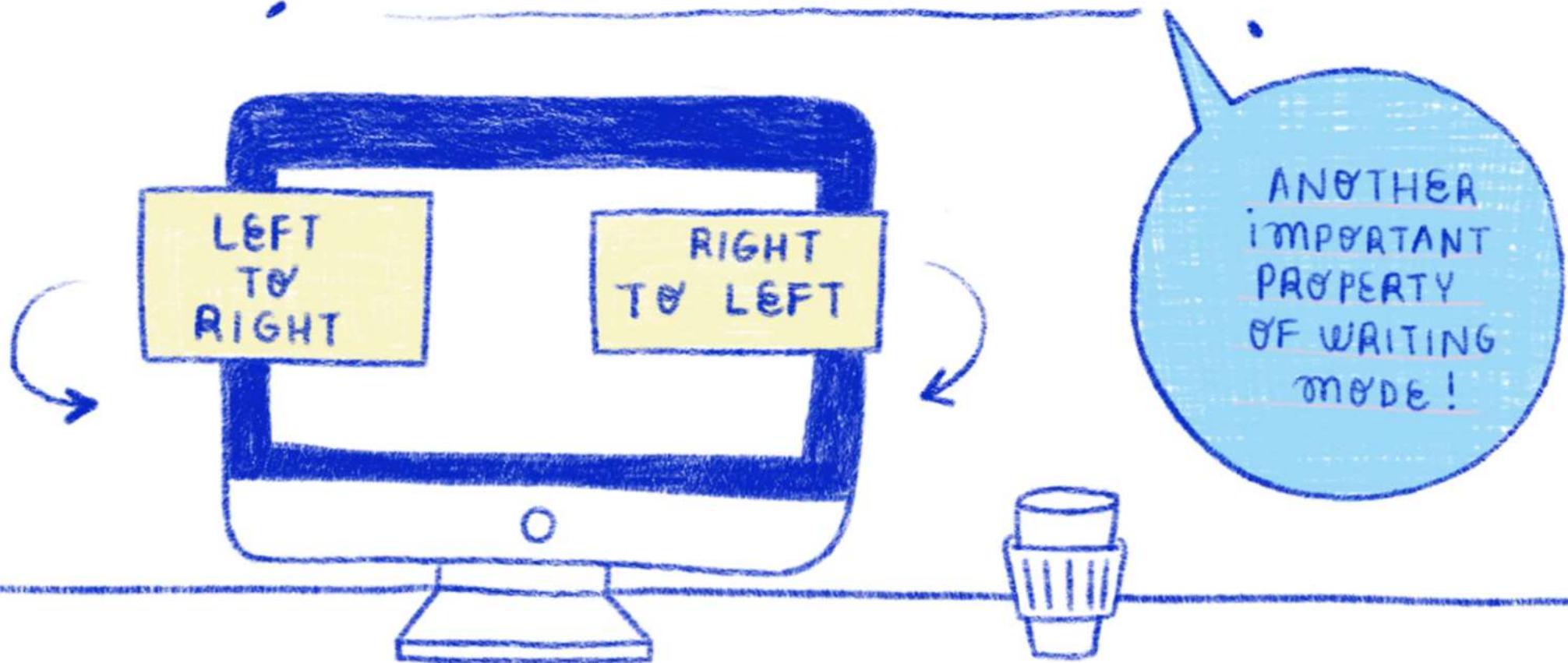
## COMPATIBILITY

CSS writing-mode property - CR

Property to define whether lines of text are laid out horizontally or vertically and the direction in which blocks progress.

Usage	% of all users	?
Global	94.38% + 2.78% = 97.16%	
unprefixed:	90.85% + 2.67% = 93.52%	

# DIRECTION





# FLEXBOX

1 DISPLAY: FLEX

2 FLEX-DIRECTION:

ROW

COLUMN



3 JUSTIFY-CONTENT:

FLEX-START

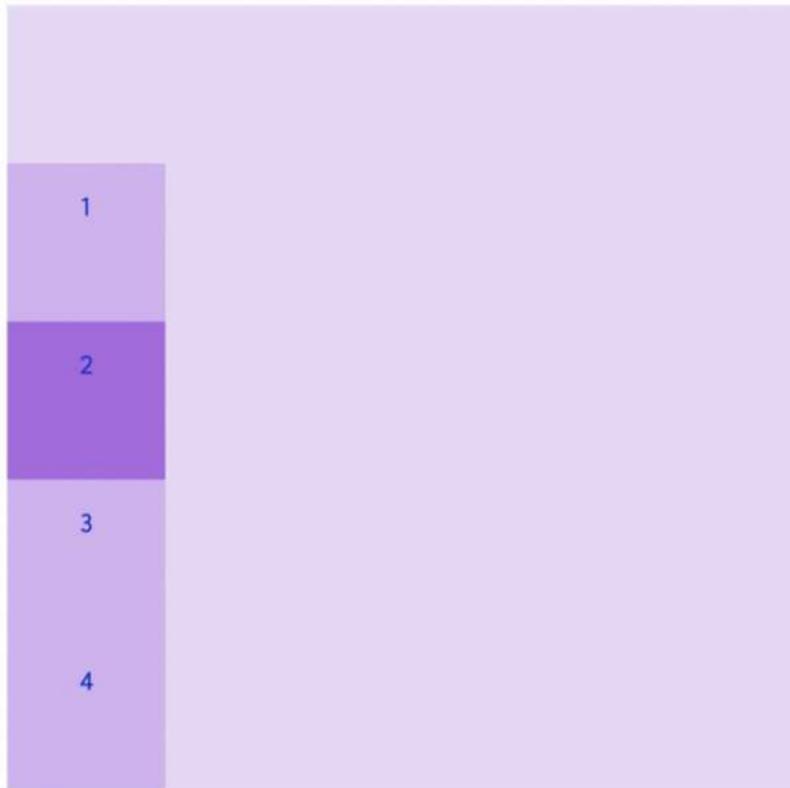
FLEX-END



HORIZONTAL TB

VERTICAL LR

VERTICAL RL



# FLEXBOX

1 DISPLAY: FLEX

2 FLEX-DIRECTION:

ROW

COLUMN



3 JUSTIFY-CONTENT:

FLEX-START

FLEX-END



HORIZONTAL TB

VERTICAL LR

VERTICAL RL

1 2 3 4

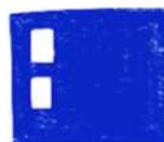
# FLEXBOX

1 DISPLAY: FLEX

2 FLEX-DIRECTION:

ROW

COLUMN



3 JUSTIFY-CONTENT:

FLEX-START

FLEX-END



HORIZONTAL TB

VERTICAL LR

VERTICAL RL

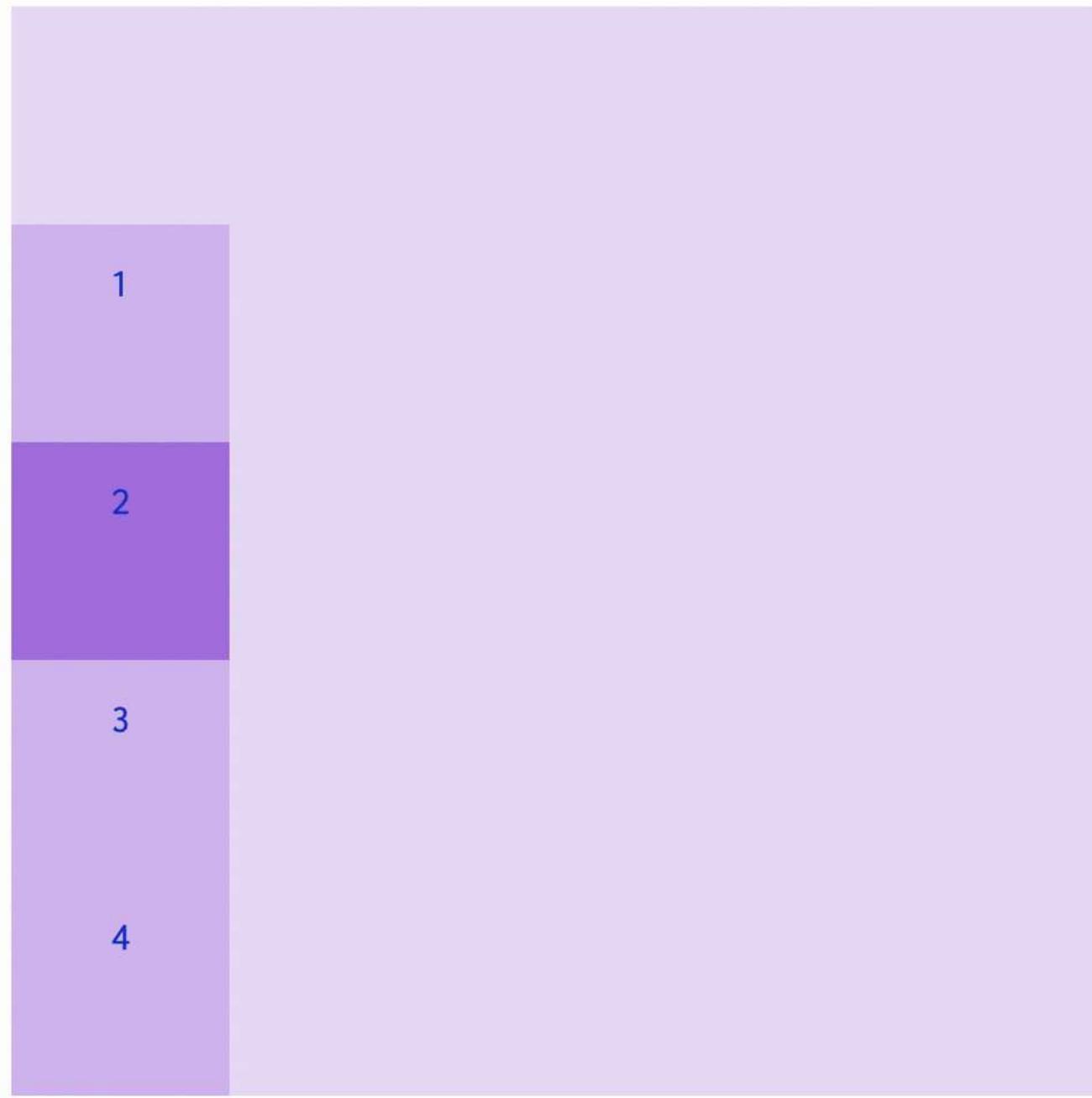


4 3 2 1

HORIZONTAL TB

VERTICAL LR

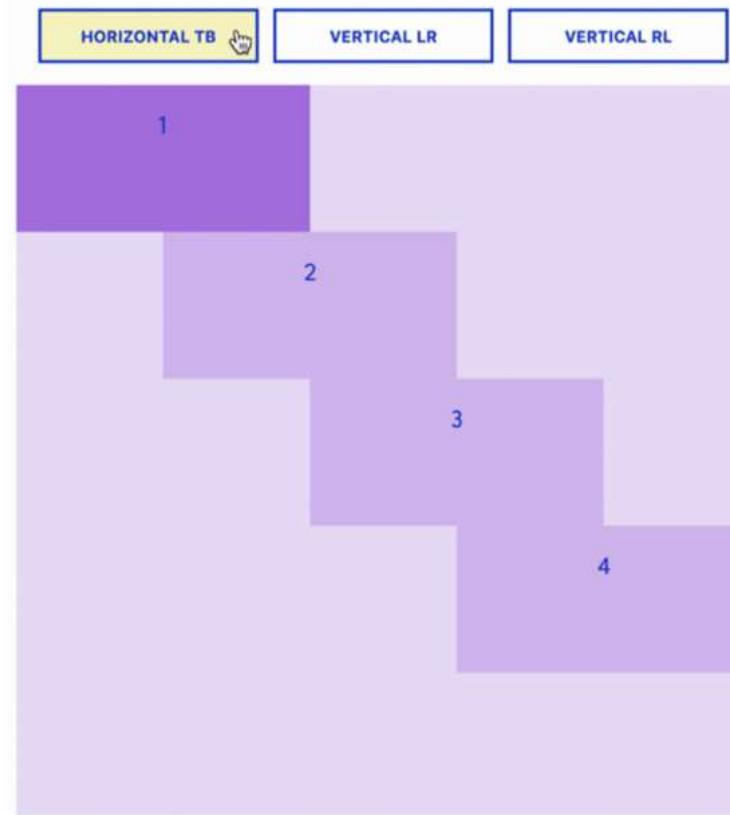
VERTICAL RL



# CSS GRID

GRID-COLUMN-START : 1  
GRID-COLUMN-END : 2  
GRID-COLUMN: 1 / 2

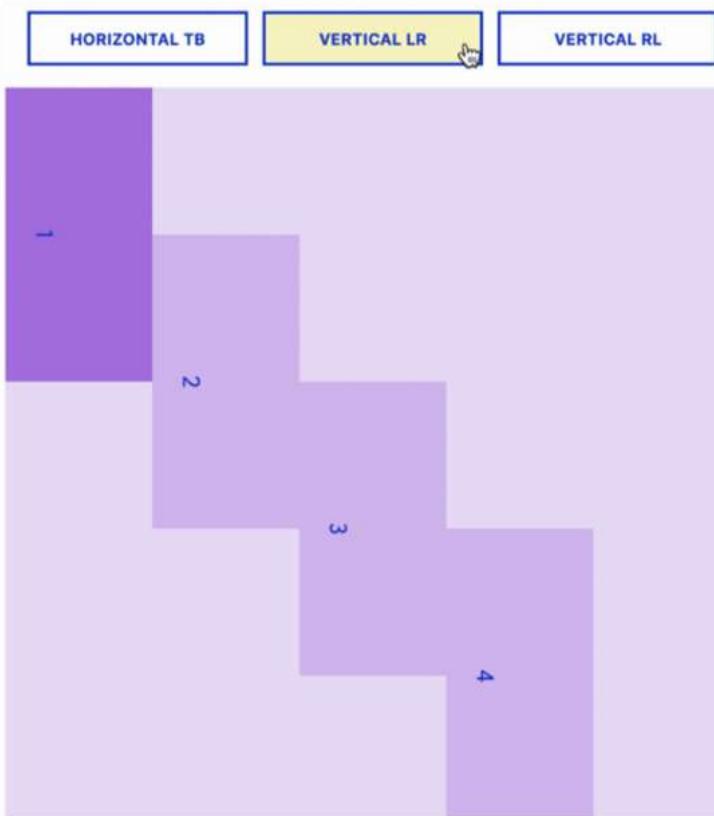
GRID-ROW-START: 1  
GRID-ROW-END: 2  
GRID-ROW: 1 / 2



# CSS GRID

GRID-COLUMN-START: 1  
GRID-COLUMN-END: 2  
GRID-COLUMN: 1 / 2

GRID-ROW-START: 1  
GRID-ROW-END: 2  
GRID-ROW: 1 / 2



# CSS GRID

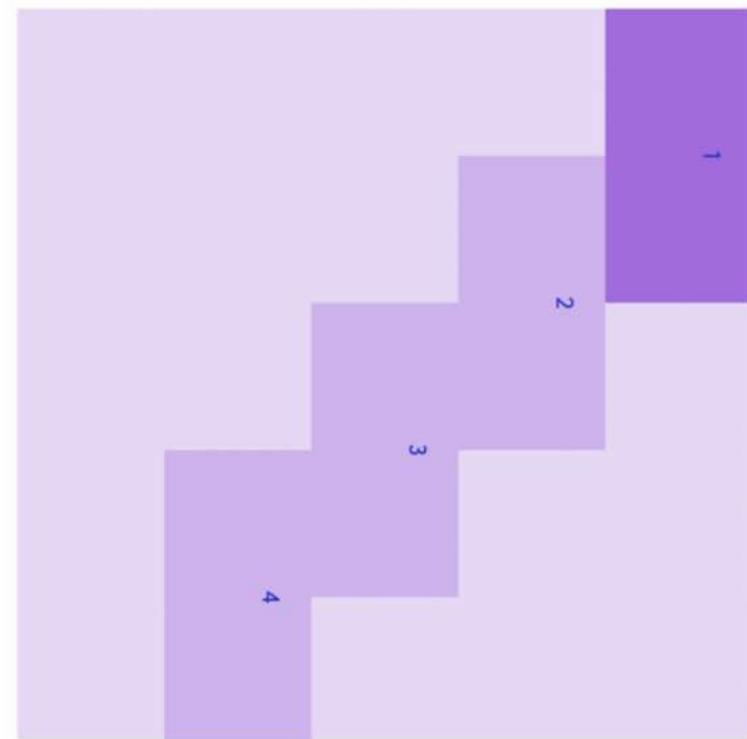
GRID-COLUMN-START : 1  
GRID-COLUMN-END : 2  
GRID-COLUMN: 1 / 2

GRID-ROW-START: 1  
GRID-ROW-END: 2  
GRID-ROW: 1 / 2

HORIZONTAL TB

VERTICAL LR

VERTICAL RL

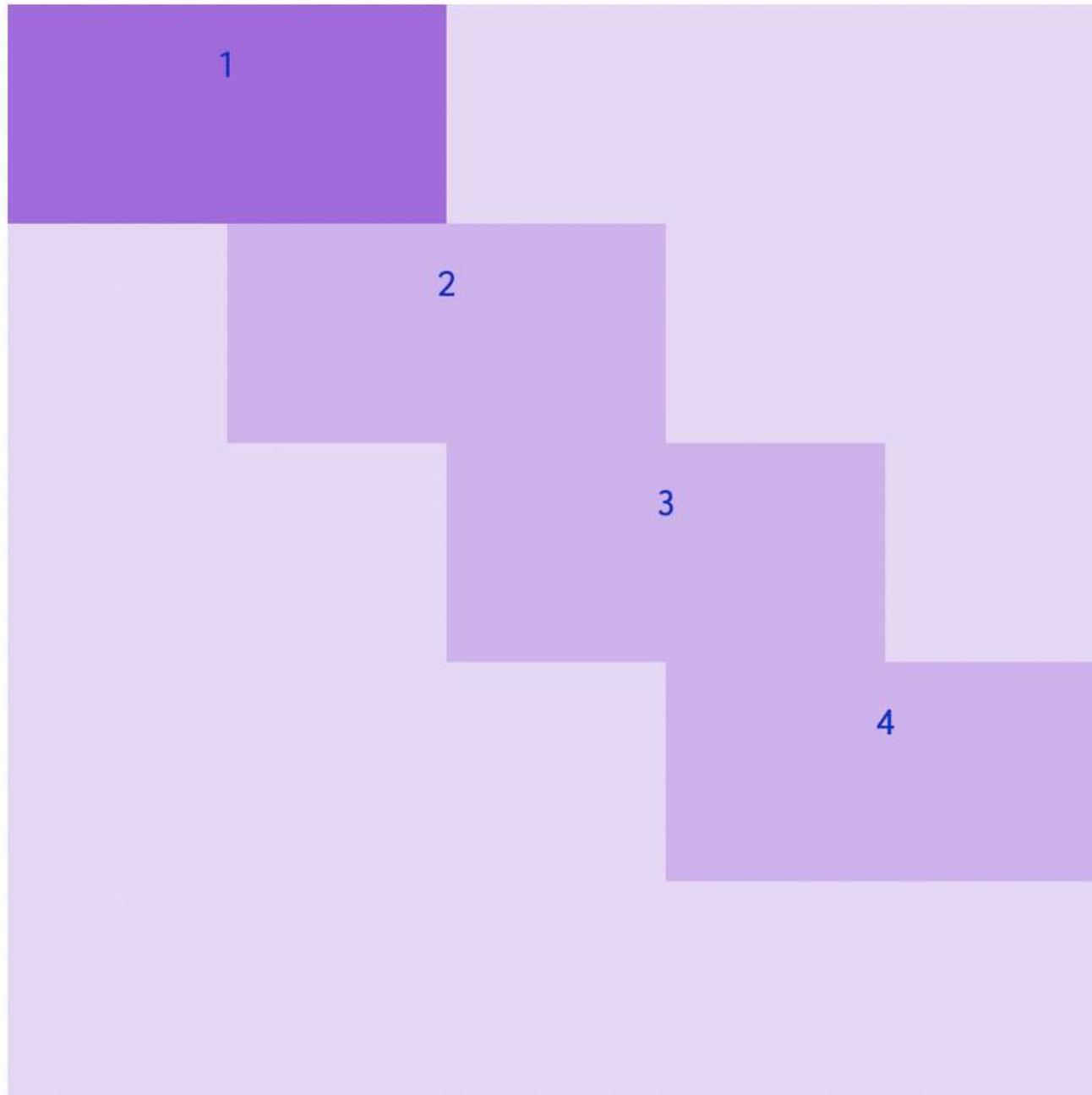


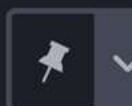
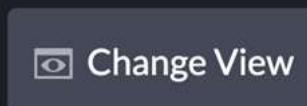
HORIZONTAL TB



VERTICAL LR

VERTICAL RL





## HTML

```
1 <div class="card">
2   <div class="card__header">El curso de CSS Grid</div>
3   <div class="card__main">Ta muy bueno :3</div>
4 </div>
```

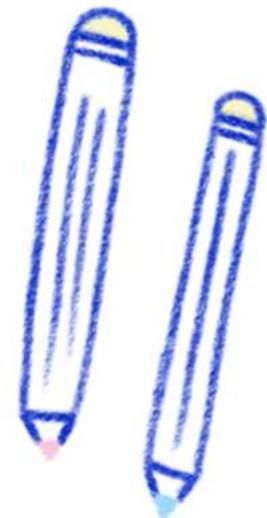
## CSS

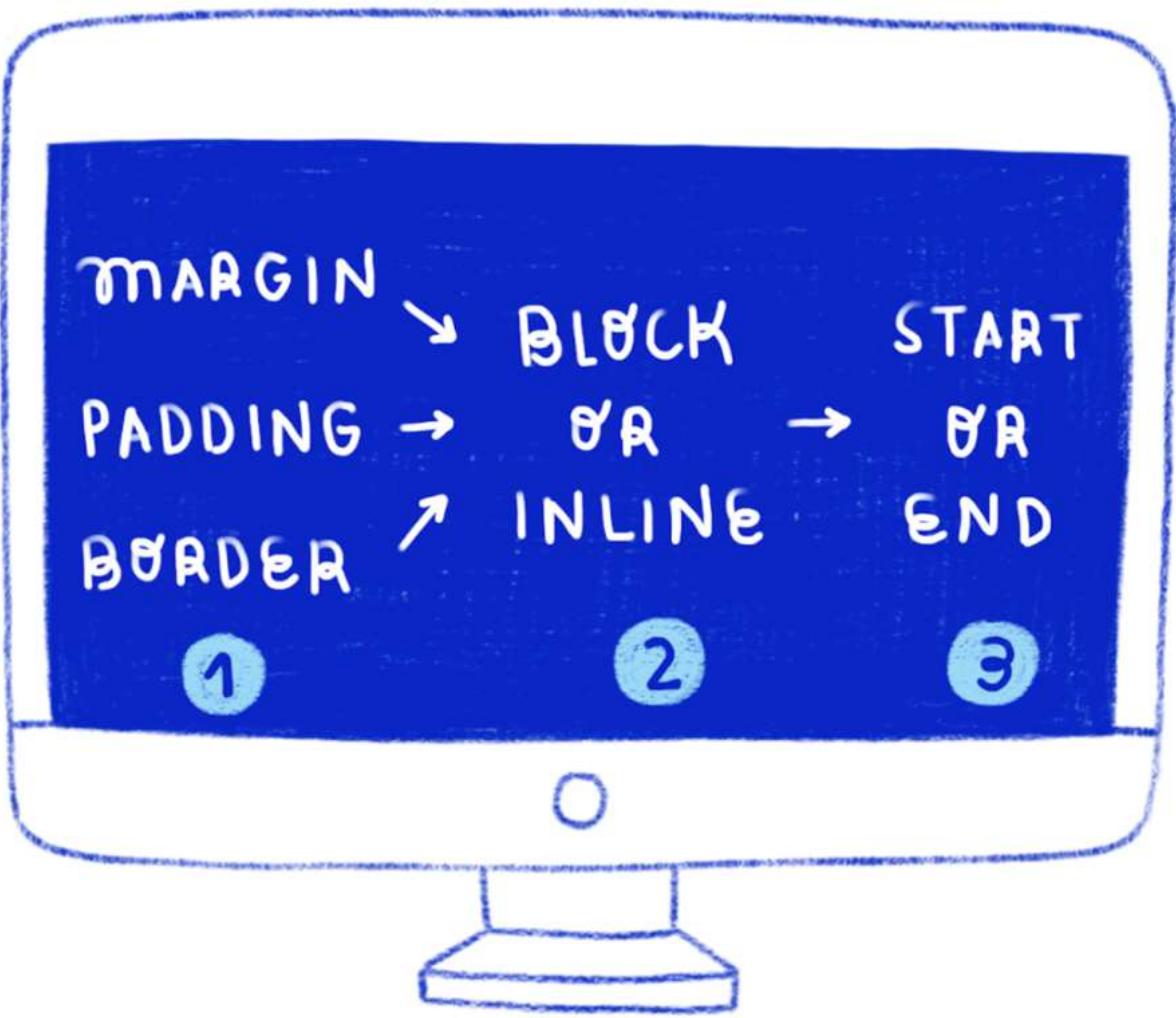
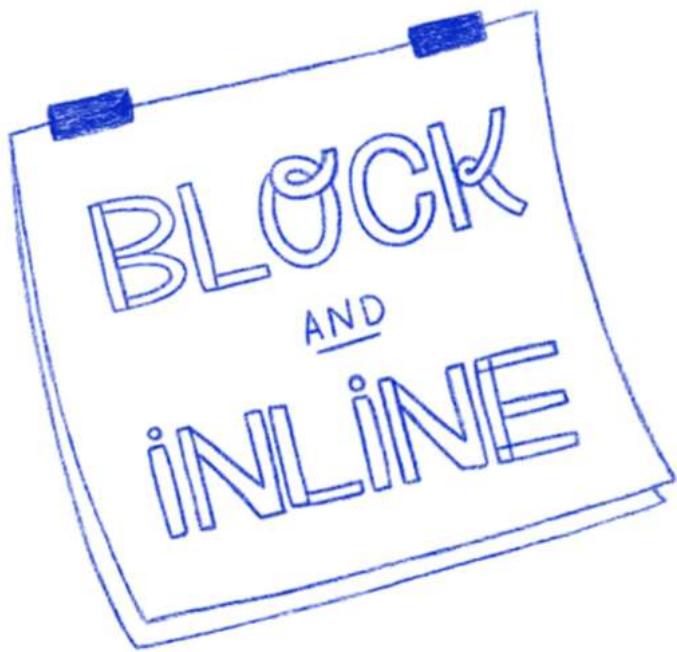
```
1 .card {
2   background: #ffdada;
3   width: 300px;
4   height: 150px;
5   border-radius: 10px;
6   overflow: hidden;
7 /* direction: rtl;
8 writing-mode: vertical-rl; */
9 }
10
```

El curso de CSS Grid

Ta muy bueno :3

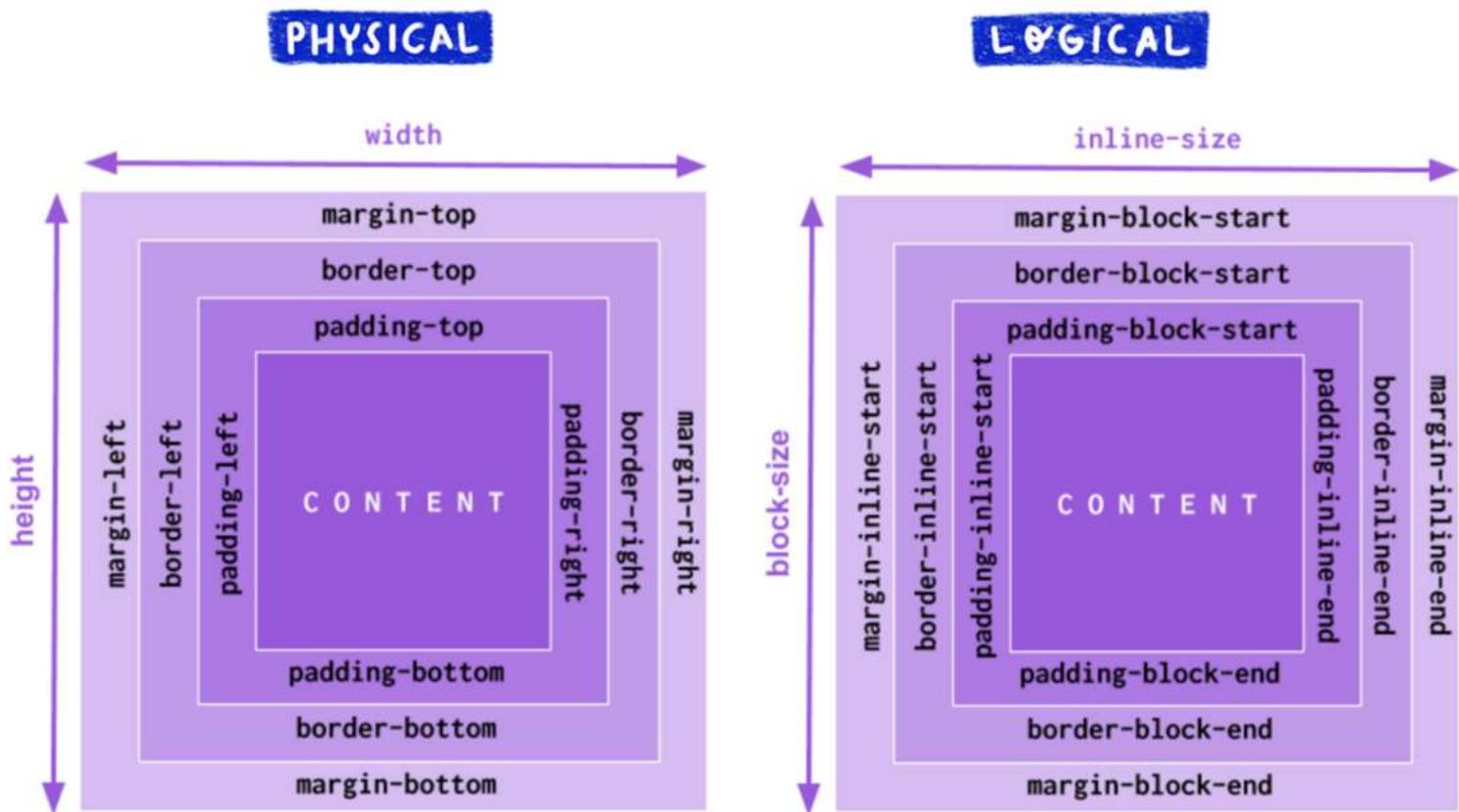
# CSS LOGICAL PROPERTIES



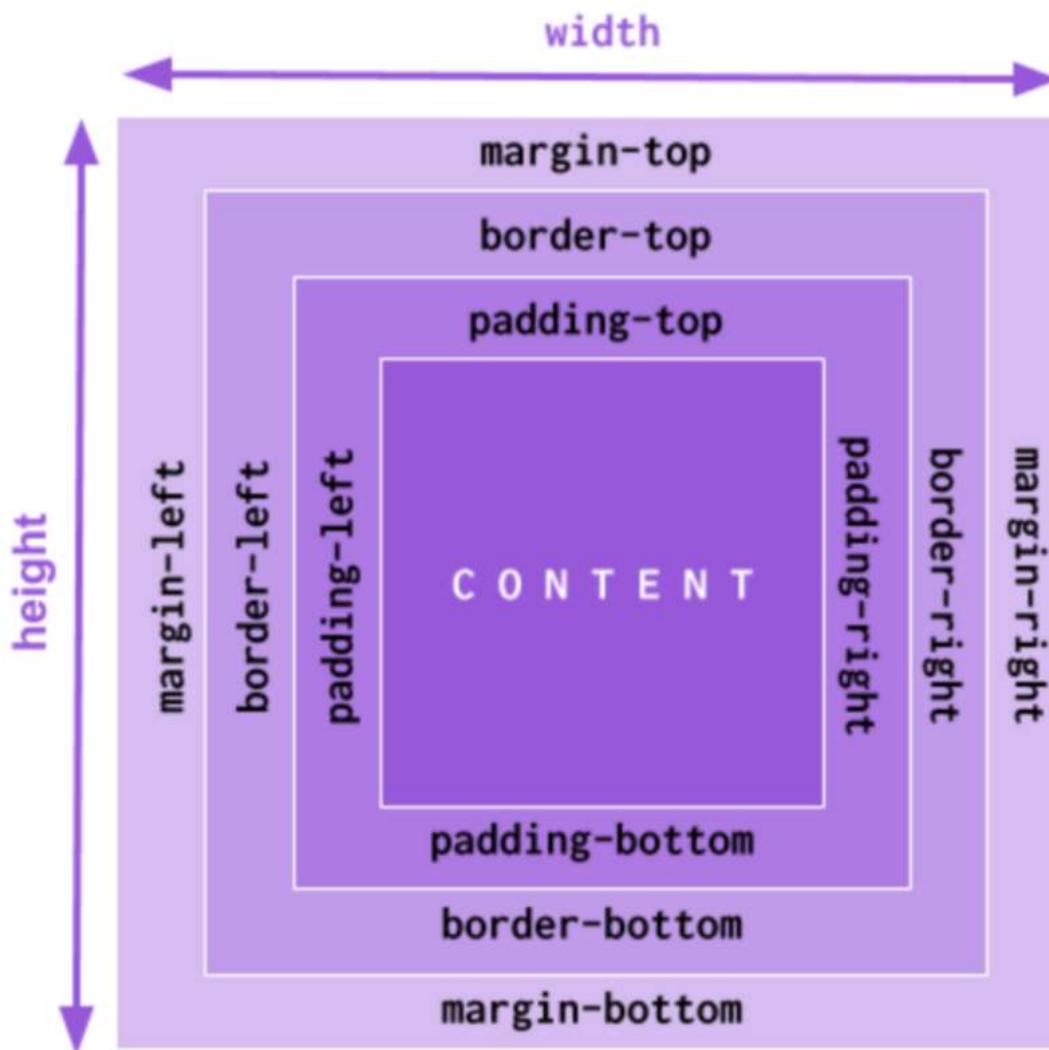


# BOX MODEL

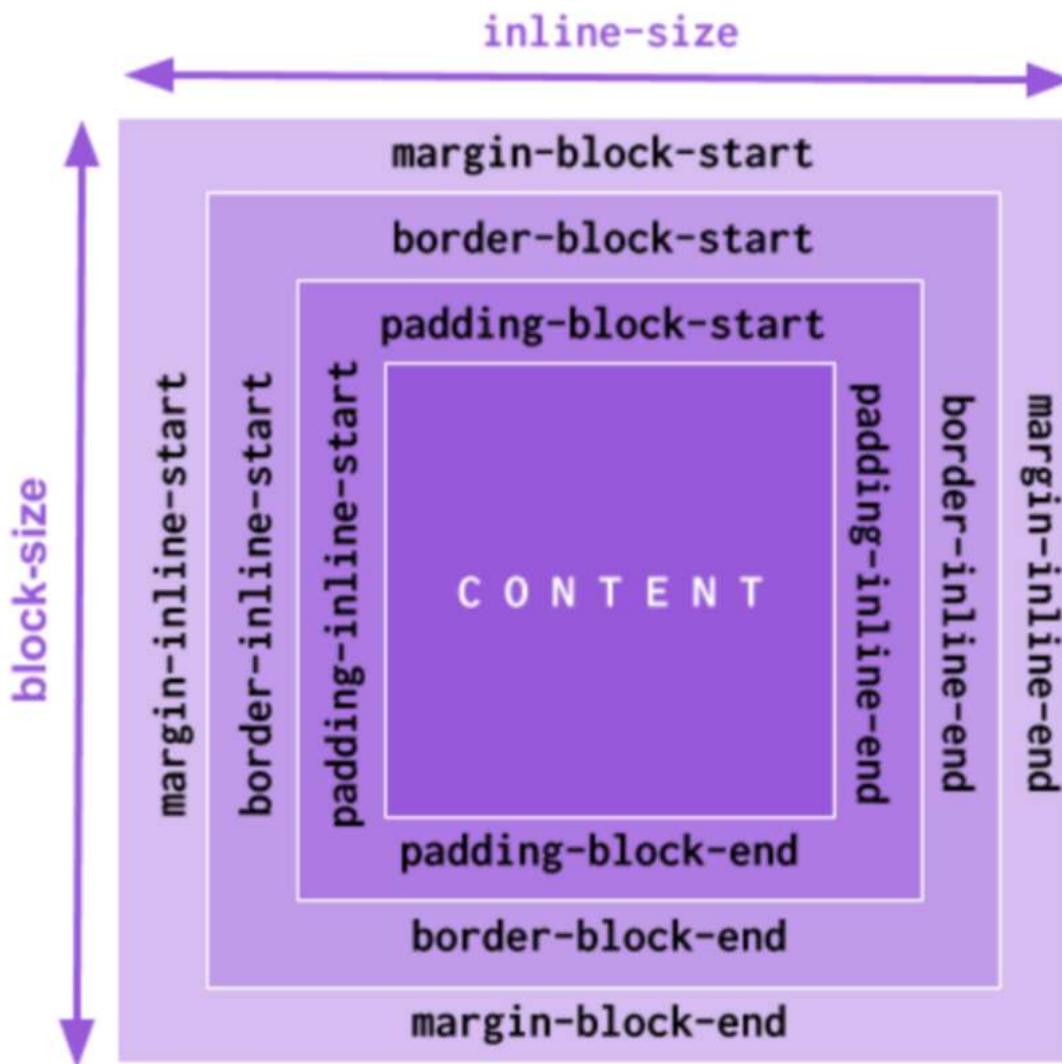
THE WAY OF  
THINKING IN CSS  
**LOGICAL  
PROPERTIES**



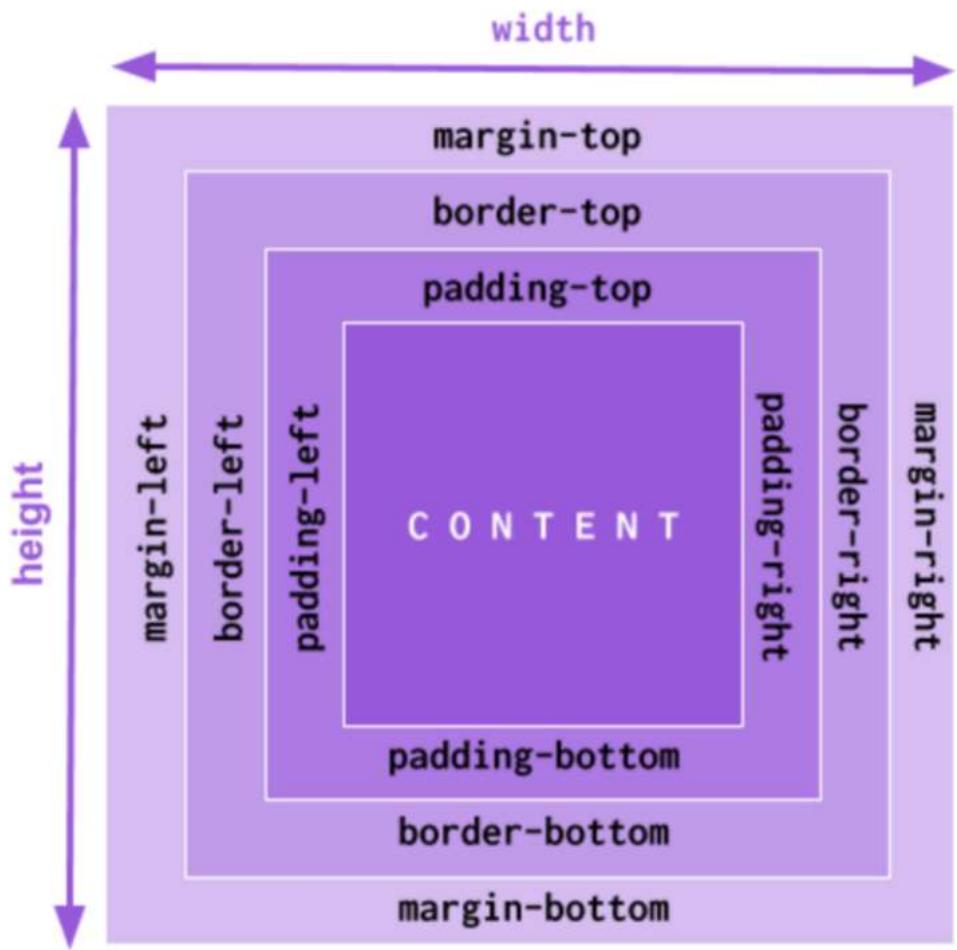
## PHYSICAL



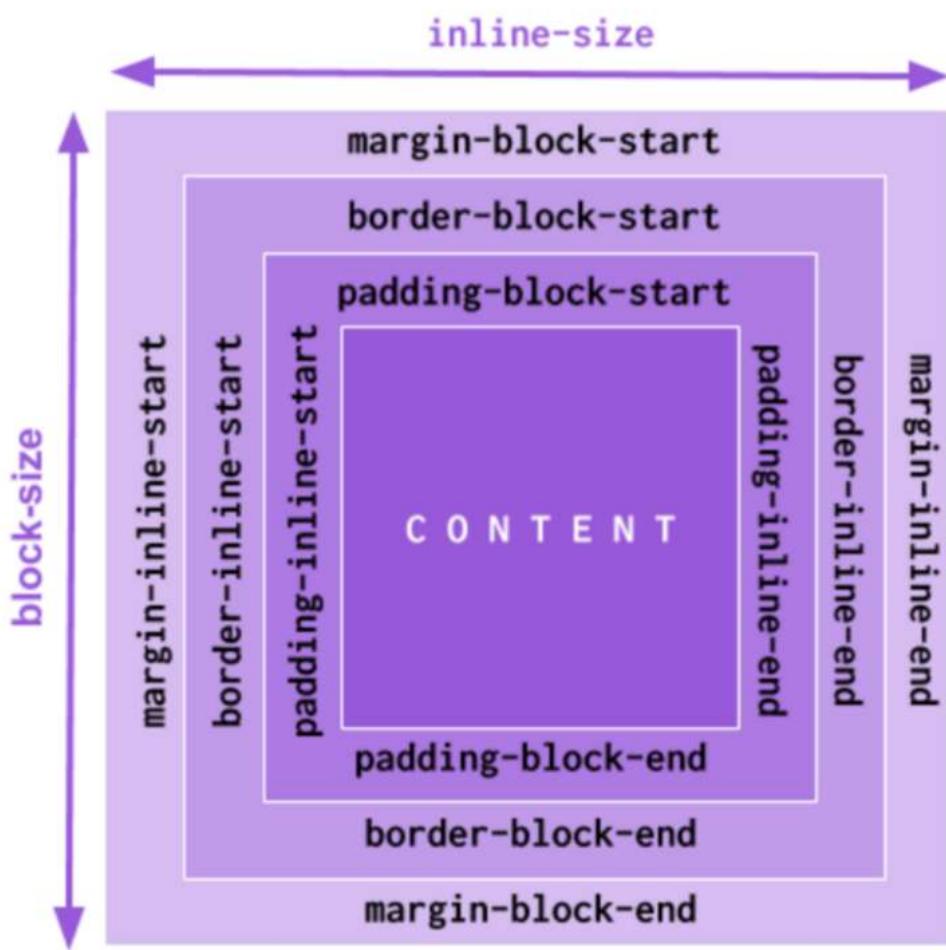
# LOGICAL



## PHYSICAL



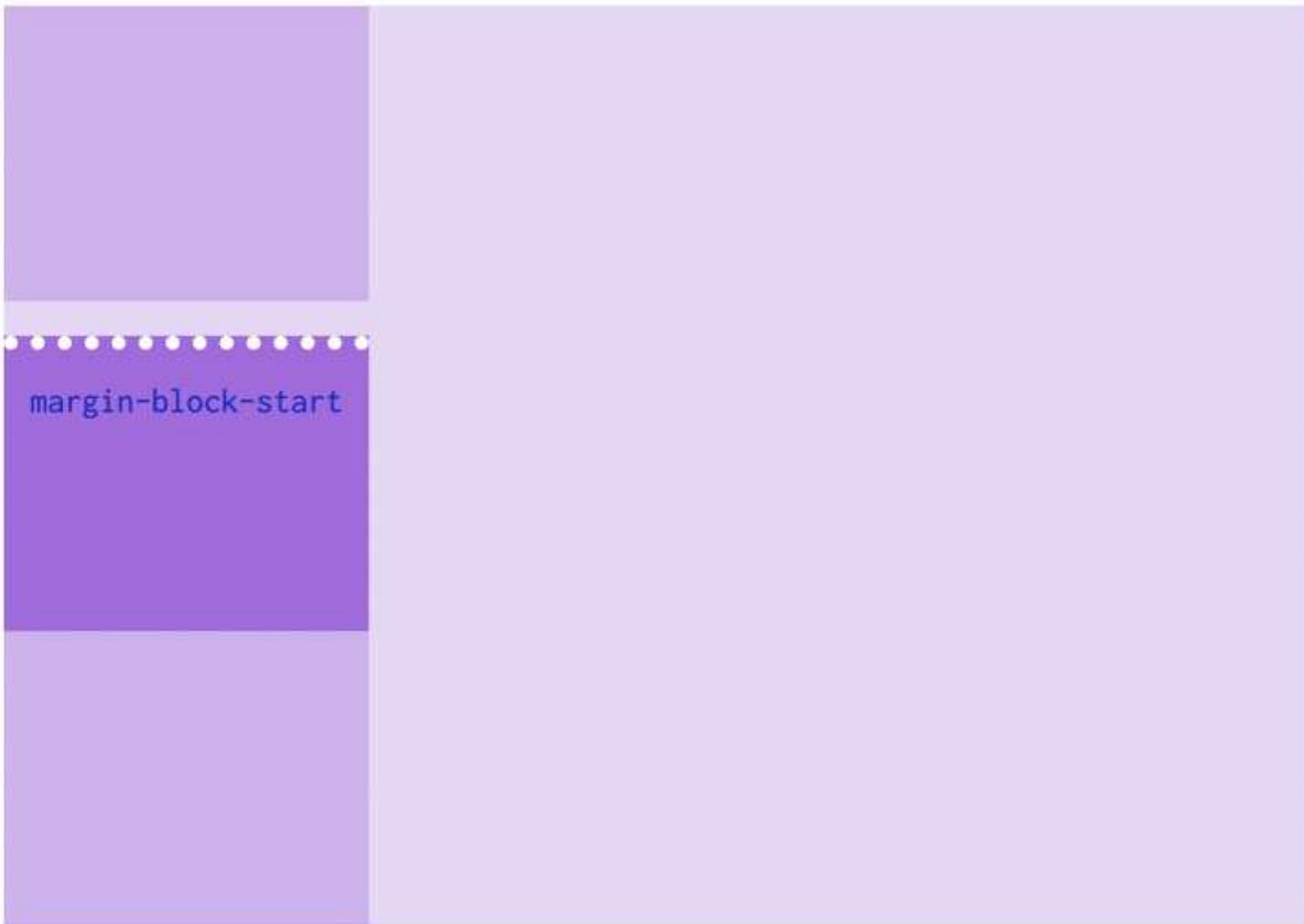
## LOGICAL



# MARGINS

PROPERTY	LOGICAL PROPERTY
MARGIN - TOP	MARGIN - BLOCK - START
MARGIN - LEFT	MARGIN - INLINE - START
MARGIN - RIGHT	MARGIN - INLINE - END
MARGIN - BOTTOM	MARGIN - BLOCK - END

- HORIZONTAL TB
- VERTICAL LR
- VERTICAL RL



HORIZONTAL TB

VERTICAL LR

VERTICAL RL

margin-block-start

HORIZONTAL TB

VERTICAL LR

VERTICAL RL

margin-block-start

A diagram showing a horizontal row of four colored boxes: light blue, light red, light green, and light orange. The light green box contains the text "margin-block-start" vertically. To the right of the light green box is a vertical dotted line consisting of white dots on a black background, representing the margin boundary.

margin-block-start

HORIZONTAL TB

VERTICAL LR

VERTICAL RL

# PADDINGS

PROPERTY	LOGICAL PROPERTY
PADDING-TOP	PADDING-BLOCK-START
PADDING-LEFT	PADDING-INLINE-START
PADDING-RIGHT	PADDING-INLINE-END
PADDING-BOTTOM	PADDING-BLOCK-END

HORIZONTAL TB



VERTICAL LR

VERTICAL RL

padding-inline-end

HORIZONTAL TB

VERTICAL LR

VERTICAL RL

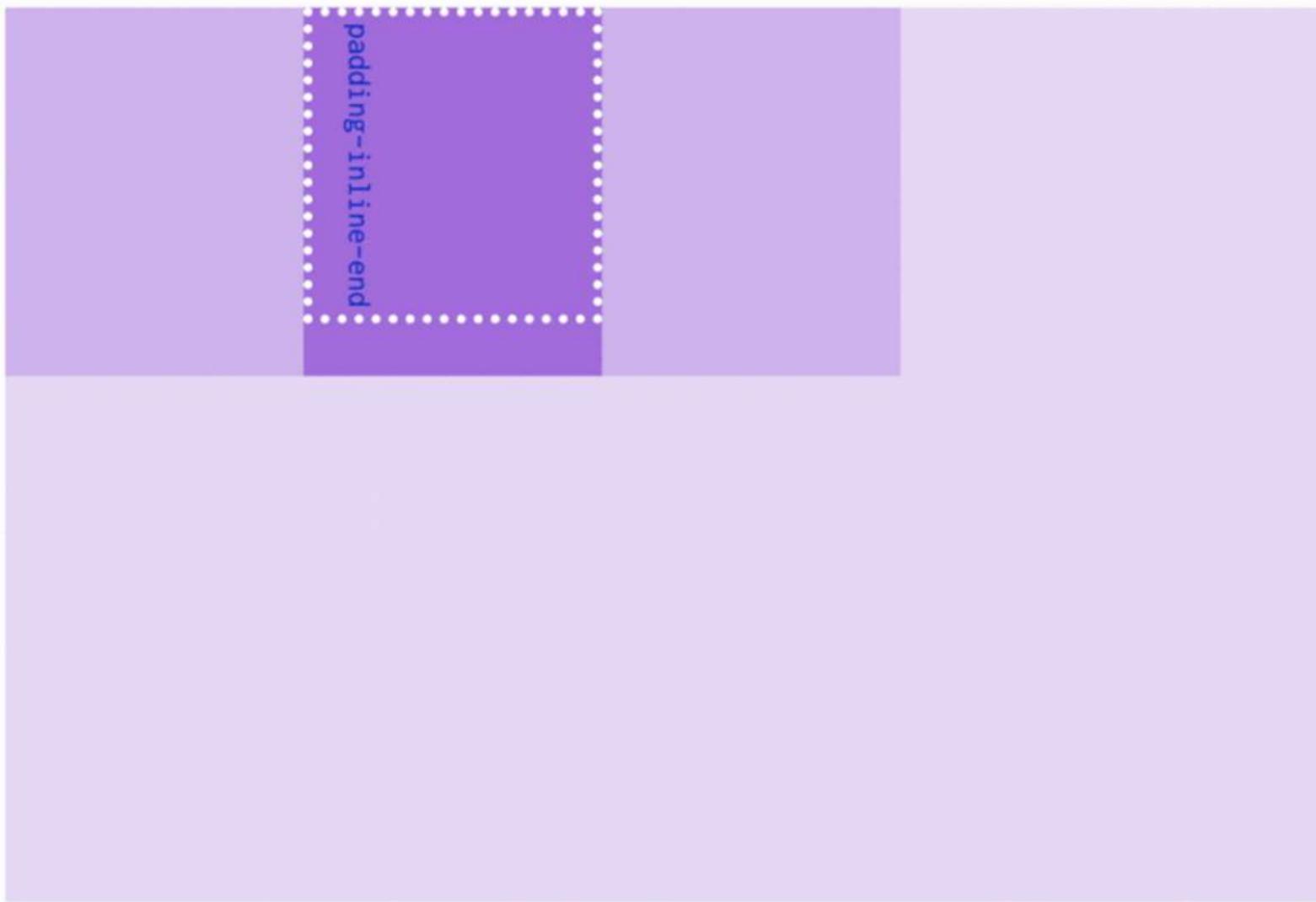
padding-inline-end

HORIZONTAL TB

VERTICAL LR



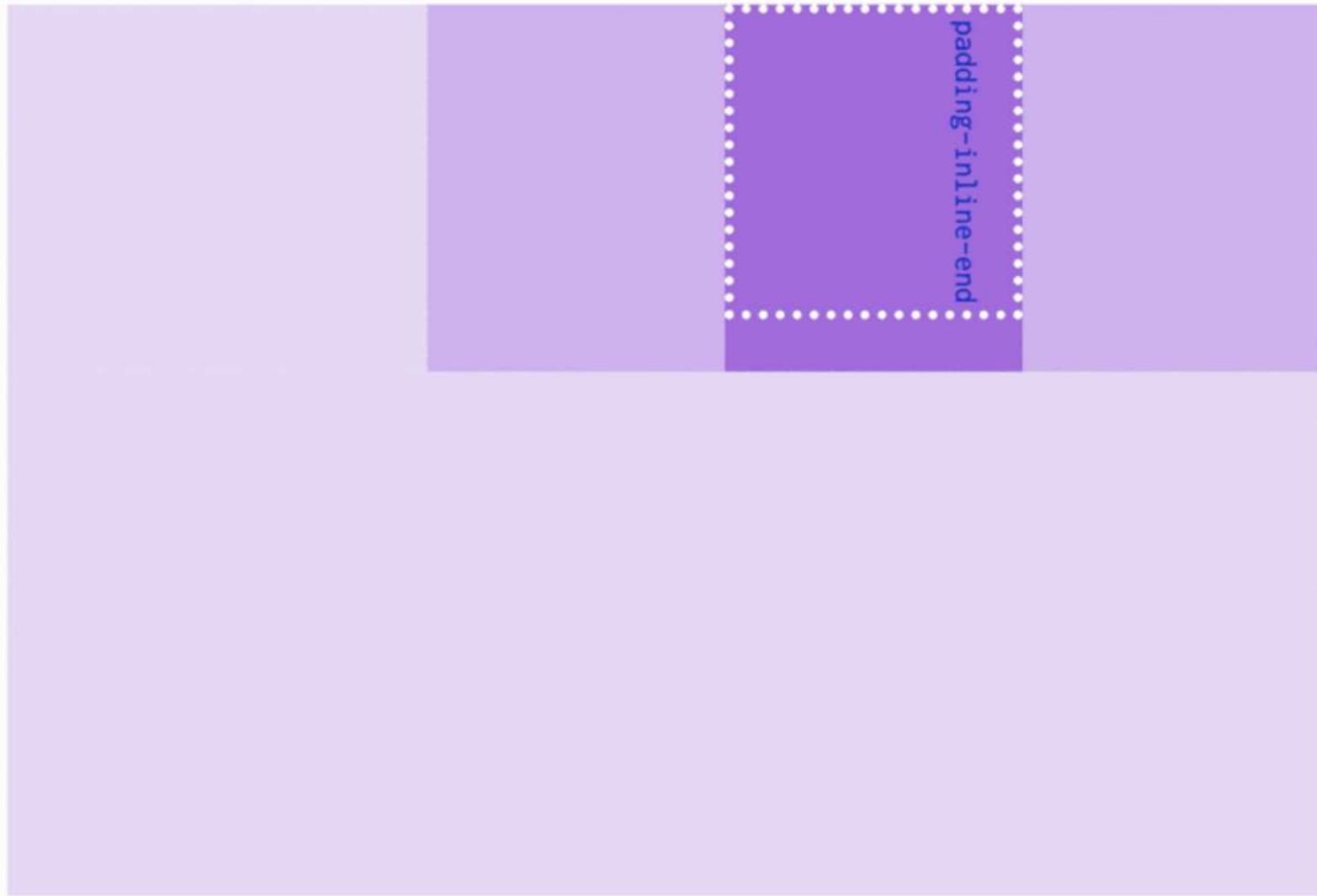
VERTICAL RL



HORIZONTAL TB

VERTICAL LR

VERTICAL RL



# BORDERS

PROPERTY	LOGICAL PROPERTY
BORDER-TOP { -SIZE   STYLE   COLOR }	BORDER-BLOCK-START { -SIZE }
BORDER-LEFT { -SIZE   STYLE   COLOR }	BORDER-INLINE-START { -SIZE }
BORDER-RIGHT { -SIZE   STYLE   COLOR }	BORDER-INLINE-END { -SIZE }
BORDER-BOTTOM { -SIZE   STYLE   COLOR }	BORDER-BLOCK-END { -SIZE }

HORIZONTAL TB

VERTICAL LR

VERTICAL RL

border-inline-start

border-inline-end

border-block-end

- HORIZONTAL TB
- VERTICAL LR
- VERTICAL RL



HORIZONTAL TB

VERTICAL LR

VERTICAL RL

border-inline-start

border-block-end

border-inline-start

border-inline-end

border-block-end



HORIZONTAL TB

VERTICAL LR

VERTICAL RL



# POSITIONS

PROPERTY	LOGICAL PROPERTY
TOP	INSET - BLOCK - START
LEFT	INSET - INLINE - START
RIGHT	INSET - INLINE - END
BOTTOM	INSET - BLOCK - END

# : CAN I USE : COMPATIBILITY

## CSS Logical Properties WD

Logical properties and values provide control of layout through logical, rather than physical, direction and dimension mappings. These properties are **writing-mode** relative equivalents of their corresponding physical properties.

Usage

Global

% of all users

72.26% + 20.22% = 92.49%

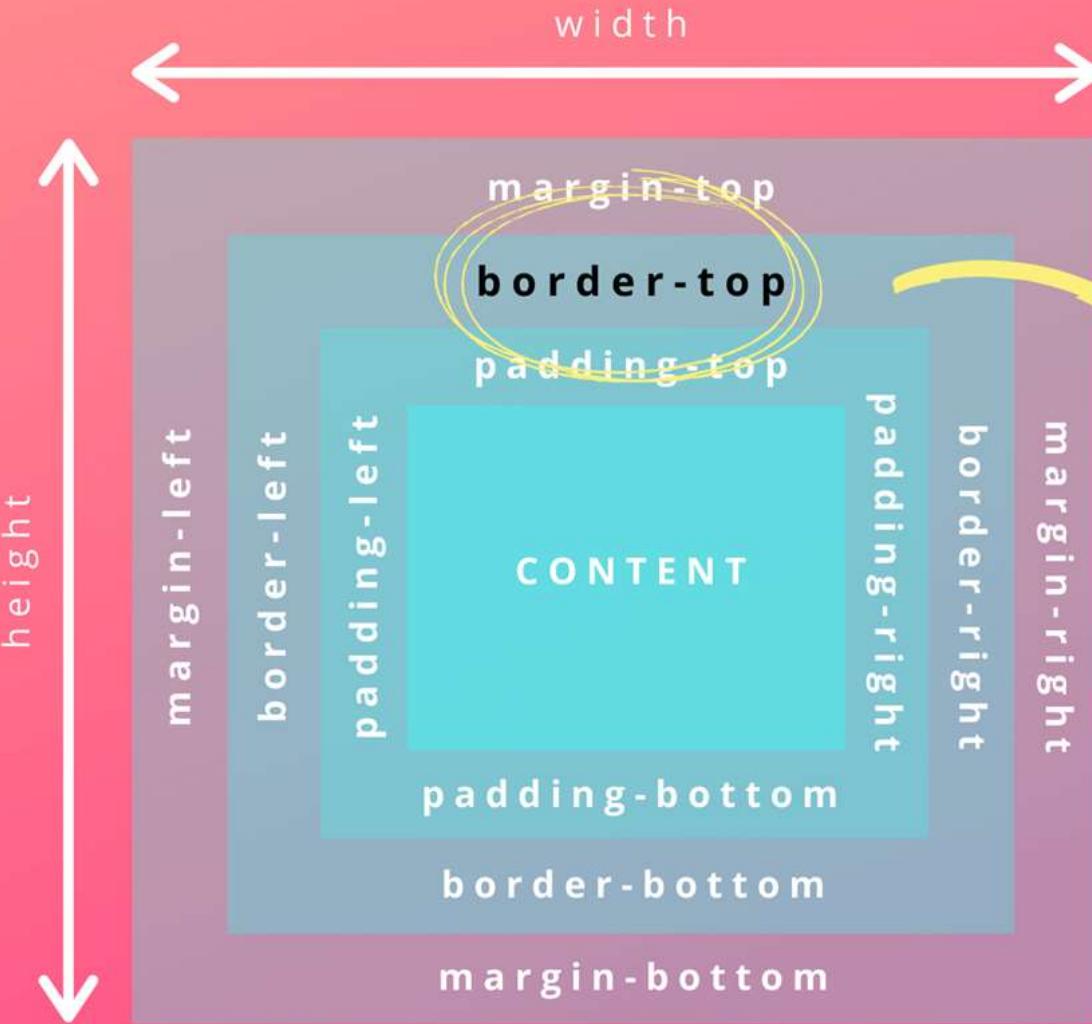
unprefixed:

72.26%





WEB PAGE  
ACCESSIBLE  
TO EVERYONE

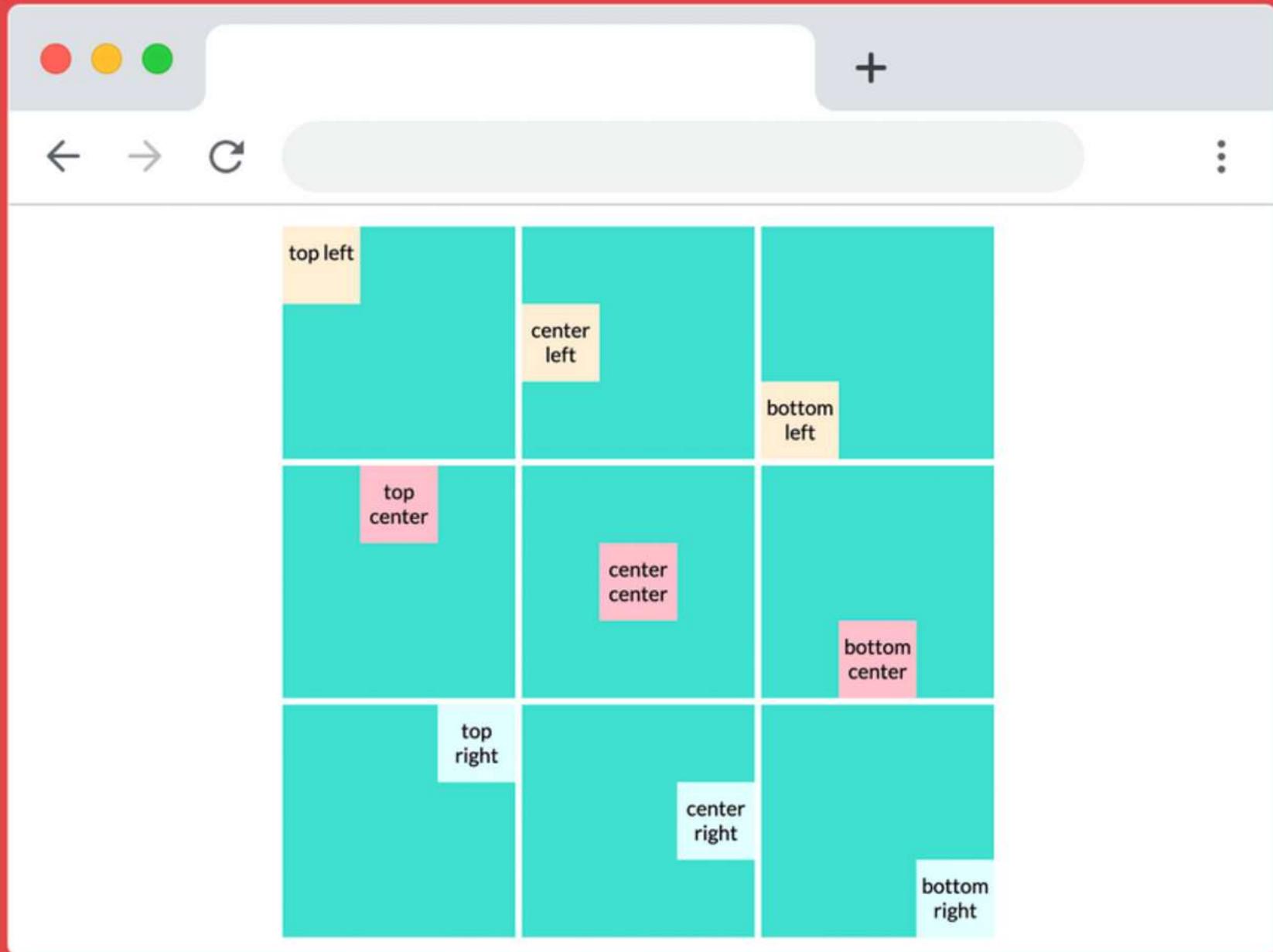


# CSS Quiz!

@teffcode

- A** border-inline-start
- B** border-block-start
- C** border-top-start

# Técnicas de alineamiento con Flexbox



# Propiedades que necesitamos



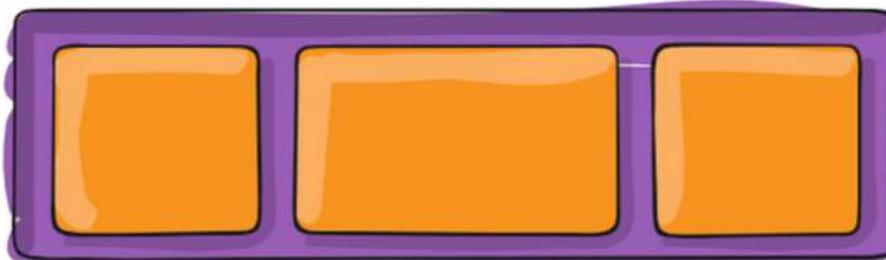
Para padres

- display: flex
- justify-content
- align-items

## Properties for the Parent (flex container)

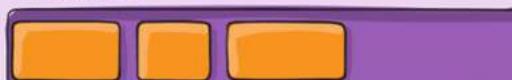
\* CSS-TRICKS

container ↗

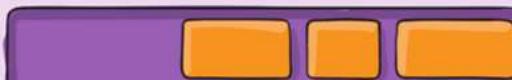


### justify-content

flex-start



flex-end



center



space-between



space-around

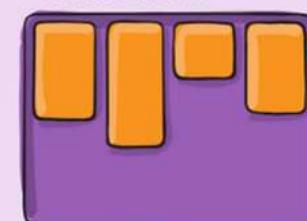


space-evenly

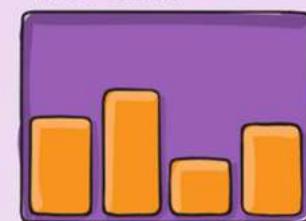


### align-items

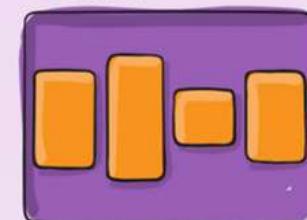
flex-start



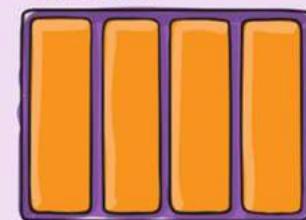
flex-end



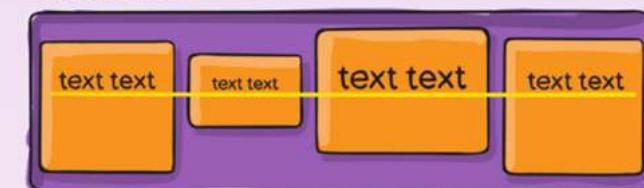
center



stretch



baseline



**Yo: Vamos al código !  
Mi página web:**

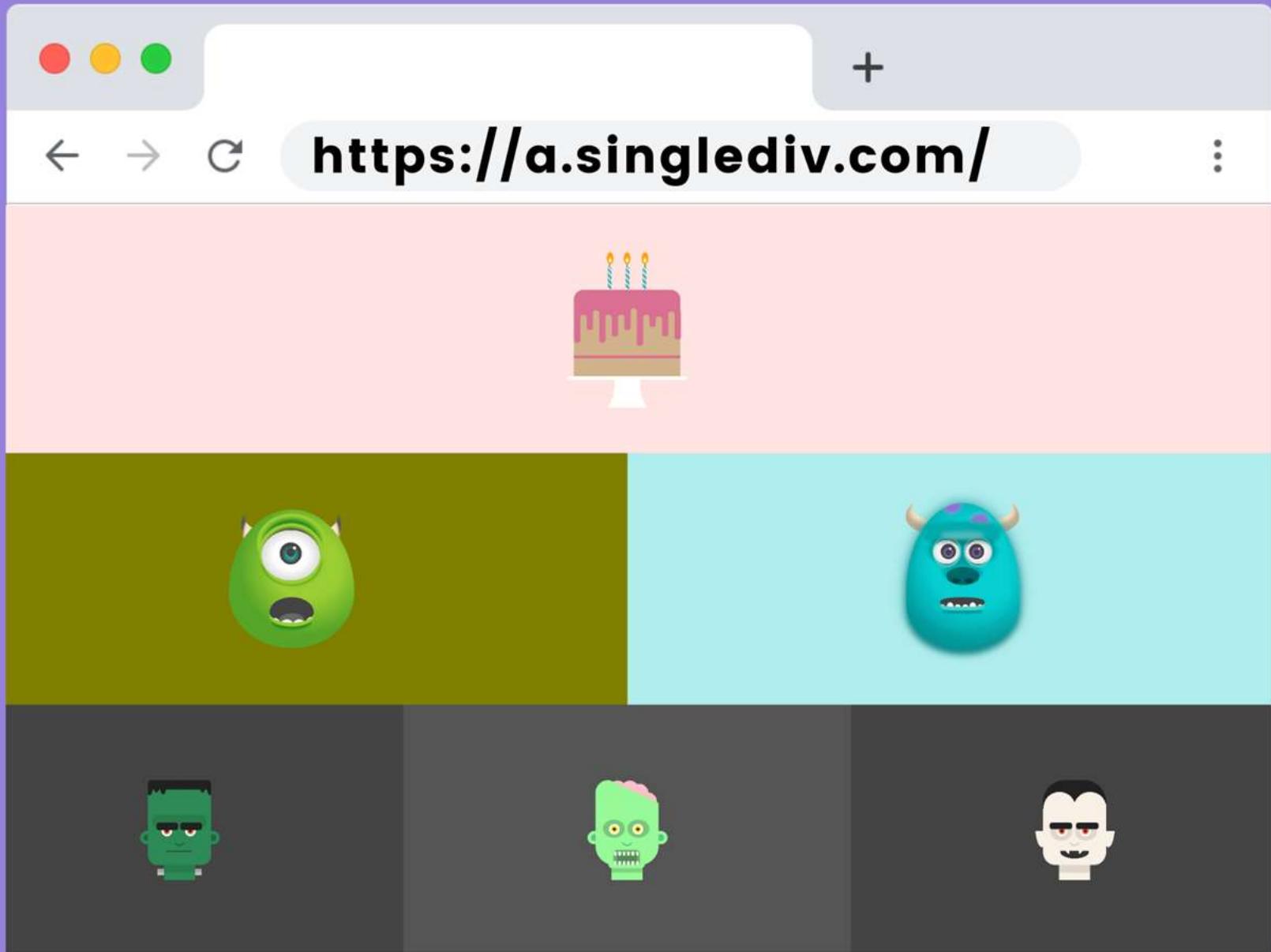


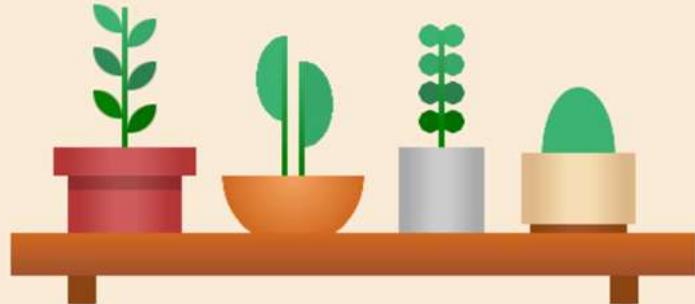
DIBUJEMOS  
CON CSS  
+ RETO



S







The screenshot shows a web browser window with a purple header bar. The address bar contains the URL <https://dev.to/raulmar>. The main content area displays a blog post by Raúl Martín. The title of the post is "🎃 No tengas miedo a dibujar con CSS". Below the title, there are three small blue buttons labeled "#css", "#html", and "#webdev". The author's profile picture is shown next to the name "Raúl Martín". Below the author information, it says "4 de oct. · 5 min read". A large image of a carved jack-o'-lantern is centered on the page. To the left of the main content, there is a sidebar with social sharing icons: a heart icon with "23", a share icon with "6", a bookmark icon with "15", and a more options icon. On the right side of the post, there is a sidebar for the author's profile. It includes a larger profile picture, the name "Raúl Martín", a bio "Interested in web dev and electronics", a "Follow" button, and sections for "WORK" (Web developer), "LOCATION" (Monterrey, MX), "EDUCATION" (UANL & PLATZI), and "JOINED" (28 de sep. de 2020). Below this, there is a section titled "Trending on DEV 🔥" with links to other posts: "My Dev.to Writing Process", "The 25 Best VS Code Extensions", and "20 free developer tools we loved in 2020 😊".



# CHARLA RECOMENDADA

**MAKE CSS YOUR  
SECRET SUPER  
DRAWING TOOL**

Wenting Zhang  
dotCSS 2016

Watch this talk on  
<http://thedotpost.com>



dotCSS 16:52

dotCSS 2016 - Wenting Zhang -  
Make CSS your secret super...  
9.9K views • 3 years ago

 dotconferences

Filmed at <http://2016.dotcss.io> on December 2nd in Paris. More talks on <http://thedotpost.com> As we all



Tú y yo  
dibujaremos  
algo hoy !



**Yo: Vamos al código !  
Mi página web:**



S

# RETO





💻 Mariangélica Usoche ✨

@musartedev Te sigue

⭐ You can call me Mus. Soy desarrolladora Front End  
💻 . Front End Developer at @platzi ❤️

⌚ Bogotá ⌚ musarte.dev ⌚ Fecha de nacimiento:  
📅 Se unió en abril de 2011

330 Siguiendo 3.420 Seguidores

🌐 Desiré 🎉, React Medellin y 30 más de las cuentas que



{Mus:arte}



# @MUSARTEDEV

@GNDX



Oscar Barajas 🇻🇪

@gndx Te sigue

Foundation Layer at @platzi - Lead at Developer Circles from Facebook,  
Speaker/Blogger. I teach React & Svelte in @platzi - 🇧🇷 🇻🇪 #EStreamerCoders  
#Frontend

⌚ Bogotá, D.C., Colombia ⌚ gndx.dev  
⌚ Fecha de nacimiento: 28 de septiembre de 1987  
📅 Se unió en marzo de 2007

949 Siguiendo 17 mil Seguidores

🌐 Juan Jose Gutierrez ★, Front End Chile y 101 más de las cuentas que sigues  
siguen a este usuario

[Code](#)[Issues 2](#)[Pull requests](#)[Actions](#)[Projects](#)[Wiki](#)[Security](#)[Insights](#)[master](#)[1 branch](#)[0 tags](#)[Go to file](#)[Add file](#)[Code](#)

musartedev Merge pull request #20 from cabustillo13/ana...

7048cfb on Sep 16

40 commits

<a href="#">.github/ISSUE_TEMPLATE</a>	Update issue templates	4 months ago
<a href="#">public</a>	Add Data Analysis	3 months ago
<a href="#">src</a>	Add Hugo Zelda art	3 months ago
<a href="#">.editorconfig</a>	add file editorconfig	4 months ago
<a href="#">.gitignore</a>	Add netlify folder to ignore	4 months ago
<a href="#">README.md</a>	Add my name as contributor to README	3 months ago
<a href="#">package.json</a>	Initial commit	4 months ago
<a href="#">yarn.lock</a>	add dark mode	4 months ago

[README.md](#)

## Dibujarte CSS

💡 ¡Llegó el momento de dibujar con CSS! 💡

[Español](#) · [English](#) · [Português Brasileiro](#)[Live](#)[▶ Preview](#)

### 👀 ¿De qué se trata?

Vamos a adentrarnos en el uso de las hojas de estilo. ¿Preparadas/os para aprender y estimular su imaginación? Cualquiera puede participar, la idea es compartir el proceso.

### 🤔 ¿Cómo puedo participar?



### About

[Galería de arte con CSS](#)[dibujartecss.musarte.dev](#)[css](#) [css-art](#)[Readme](#)

### Releases

No releases published

### Packages

No packages published

### Contributors 4



musartedev Mariangélica U...



cabustillo13 Carlos Bustillo

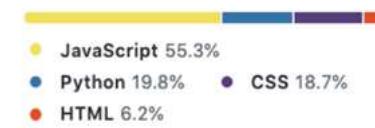


AndresParraGO Andres Par...



aldo-rl

### Languages

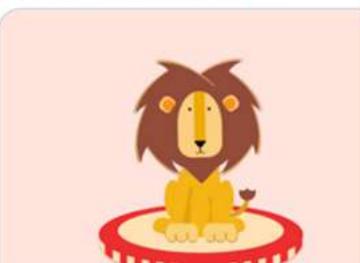




**Camilo** @camilofrontend · 22 ago.

Mi primer dibujo con CSS, el reto que lanzaron @musartedev y @gndx  
#DibujarteCSS

[codepen.io/camilo-aguiler...](https://codepen.io/camilo-aguiler...)

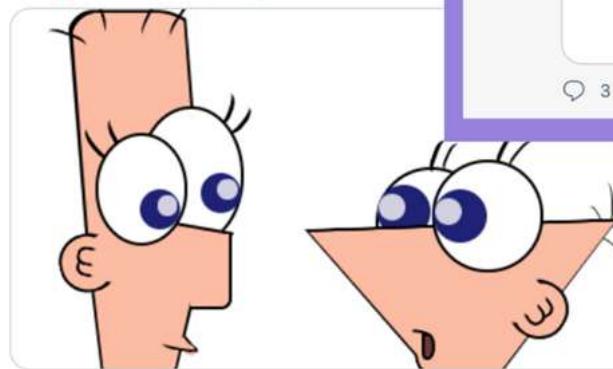


**Tomas Salina** @salinatomass · 10 ago.

#dibujarteCSS

Es simplemente increíble lo que se puede lograr con paciencia.

Muchas gracias @musartedev por tu desafío, aquí Link: [github.com/salinatomas/Ph...](https://github.com/salinatomas/Ph...)



**Diego** @diego\_osunag · 4 ago.

Mi primer #DibujarteCSS, recordando viejos tiempos de niño dibujando con paint, cuando sólo eran 150. Lo prometido es deuda @musartedev, que tal quedó?

[codepen.io/diego\\_osunag/p...](https://codepen.io/diego_osunag/p...)



**Emmanuel García** @EmmalsWorking · 17 ago.

Mi avance del 2do reto de #DibujarteCSS con @musartedev y @gndx. Este viernes no olviden echarnos 🎉 porras por Youtube



**UnaKadaUwU** @UwKada · 17 ago.

Hice este pequeño dibujo C; del imprimable @HeRnyBreak, espero te guste mucho uwu con muzhoLooff <3  
#dibujos #digitalart #dibujartecss

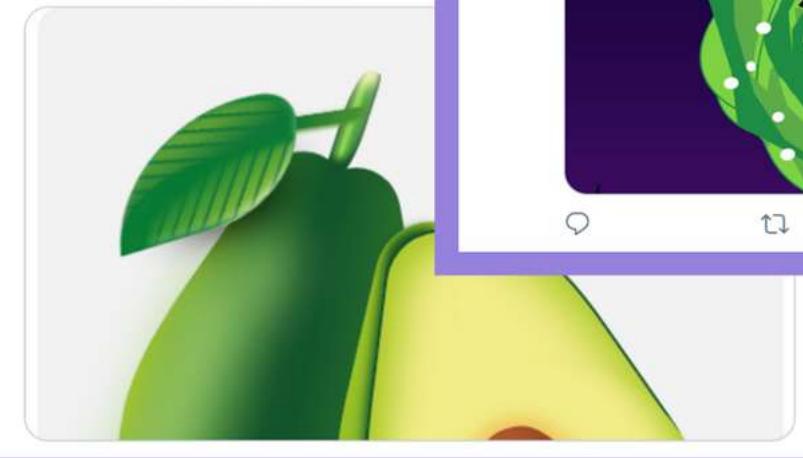


**Ana Hernández** @ianahernandez ·

Hace varios días me uní al reto de @mu primer dibujo en CSS, me encanta el re

Mi #DibujarteCSS se resumen en sombra

👉 [codepen.io/ianahernandez/...](https://codepen.io/ianahernandez/...)



**Daniek Tj** @daniektj · 25 ago.

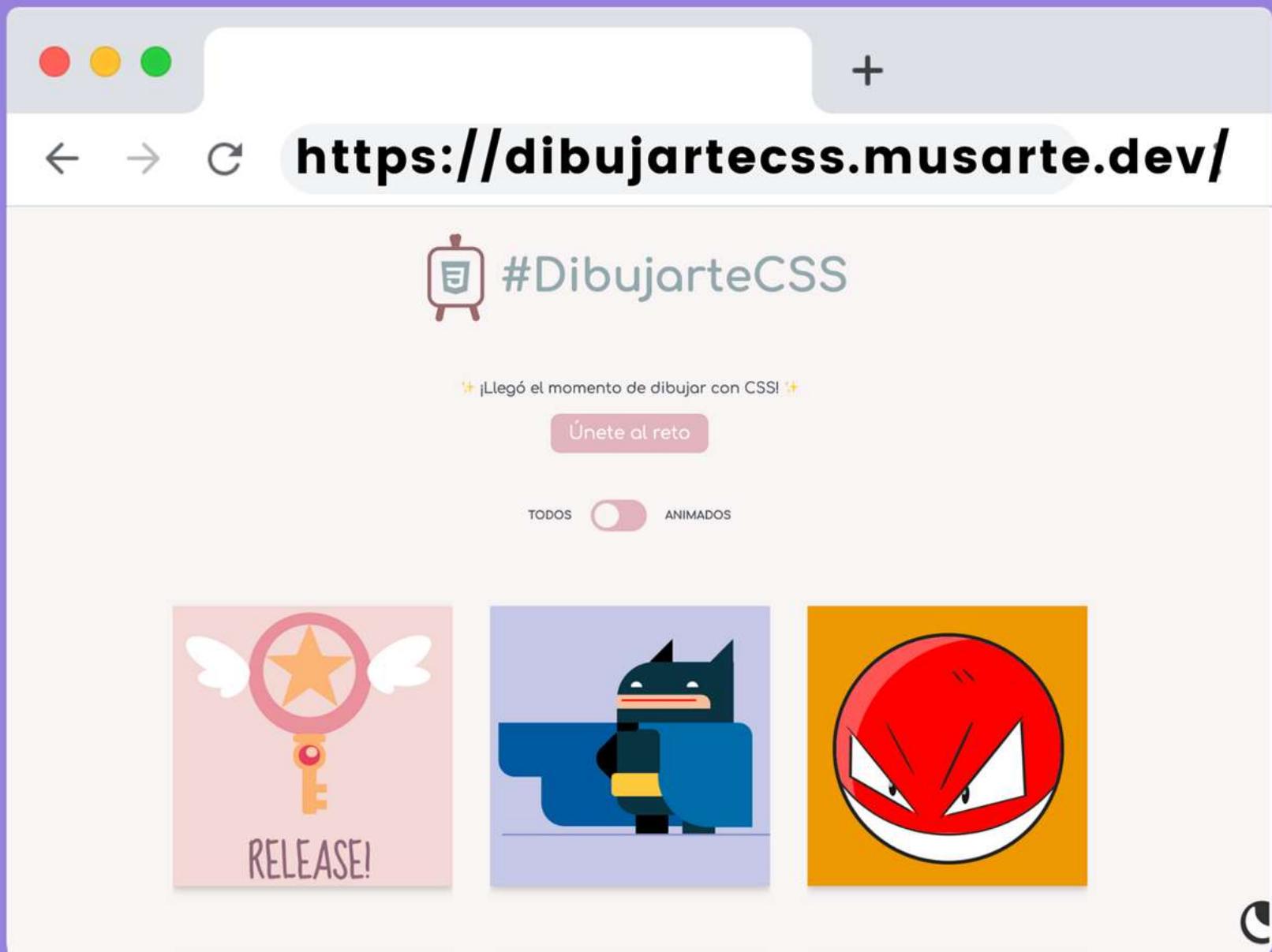
Esta es la versión actualizada con los mejoras sugeridas por @mus y @gndx en #DibujarteCSS

- Parpadeo de ojos en la cara de Rick.
- Un solo botón con Toggle.

[codepen.io/daniektj/pen/e...](https://codepen.io/daniektj/pen/e...)

Gracias por el concurso y por motivarnos, aprendí mucho realizando reto.





# RETO

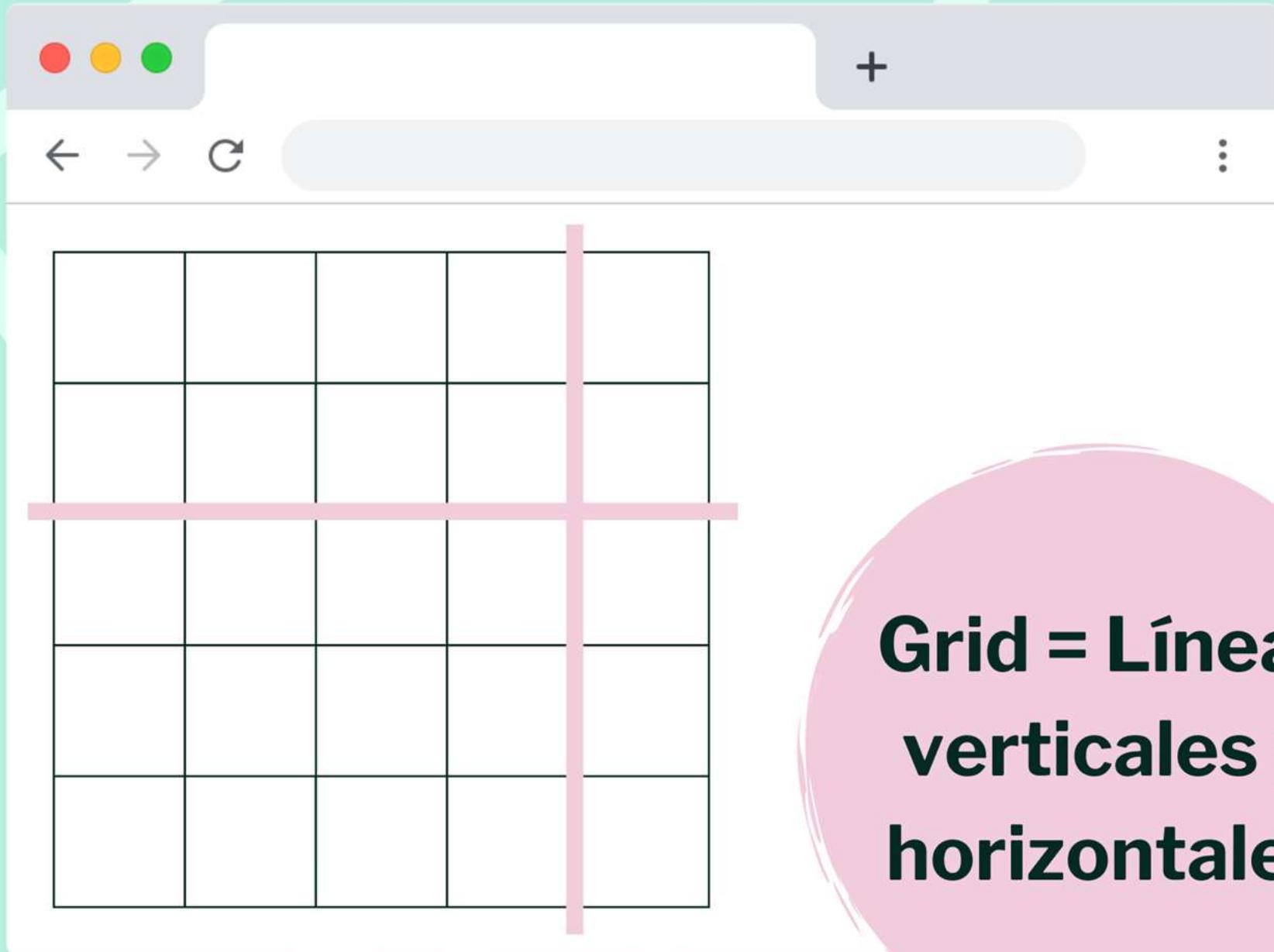
Dibujar con CSS

El Twitter  
usando el  
#dibujartecss

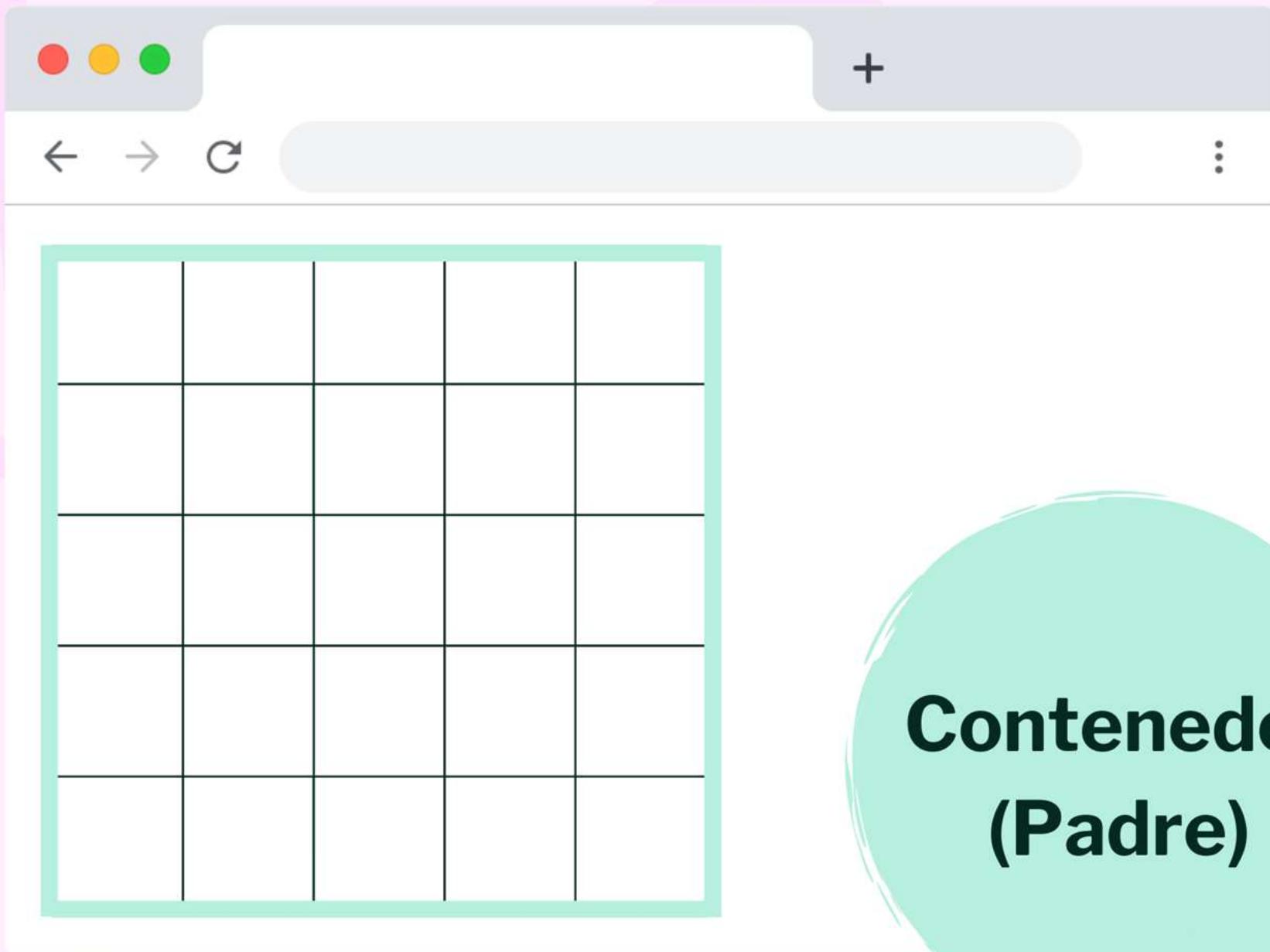


Aquí en los  
comentarios  
del curso :)

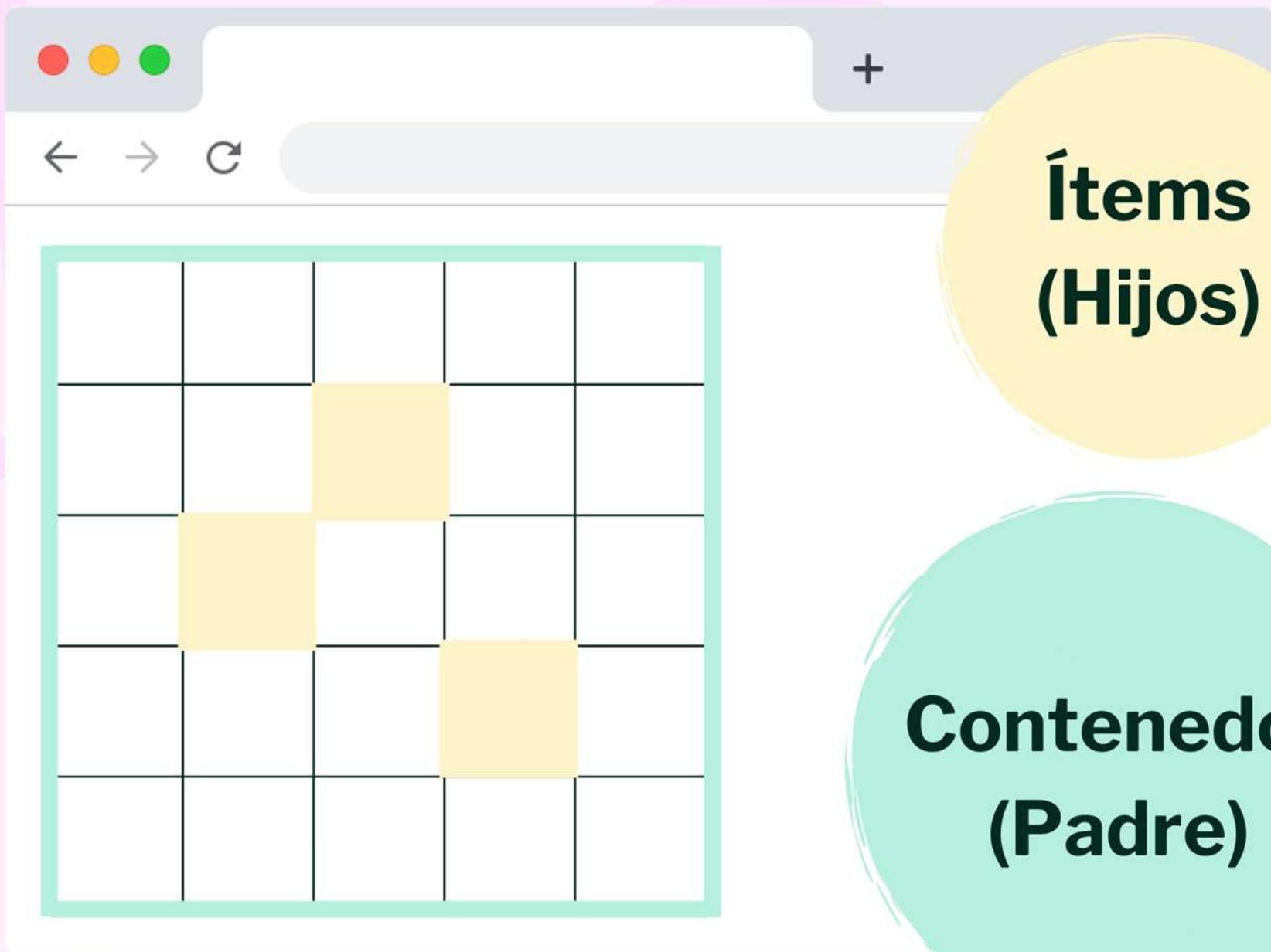
# **Grid y las relaciones padre e hijos inmediatos + Quíz**



**Grid = Líneas  
verticales y  
horizontales**



**Contenedor  
(Padre)**



**Ítems  
(Hijos)**

**Contenedor  
(Padre)**



```
<div class="container">
    <div class="item"></div>
    <div class="item"></div>
    <div class="item"></div>
    ...
</div>
```



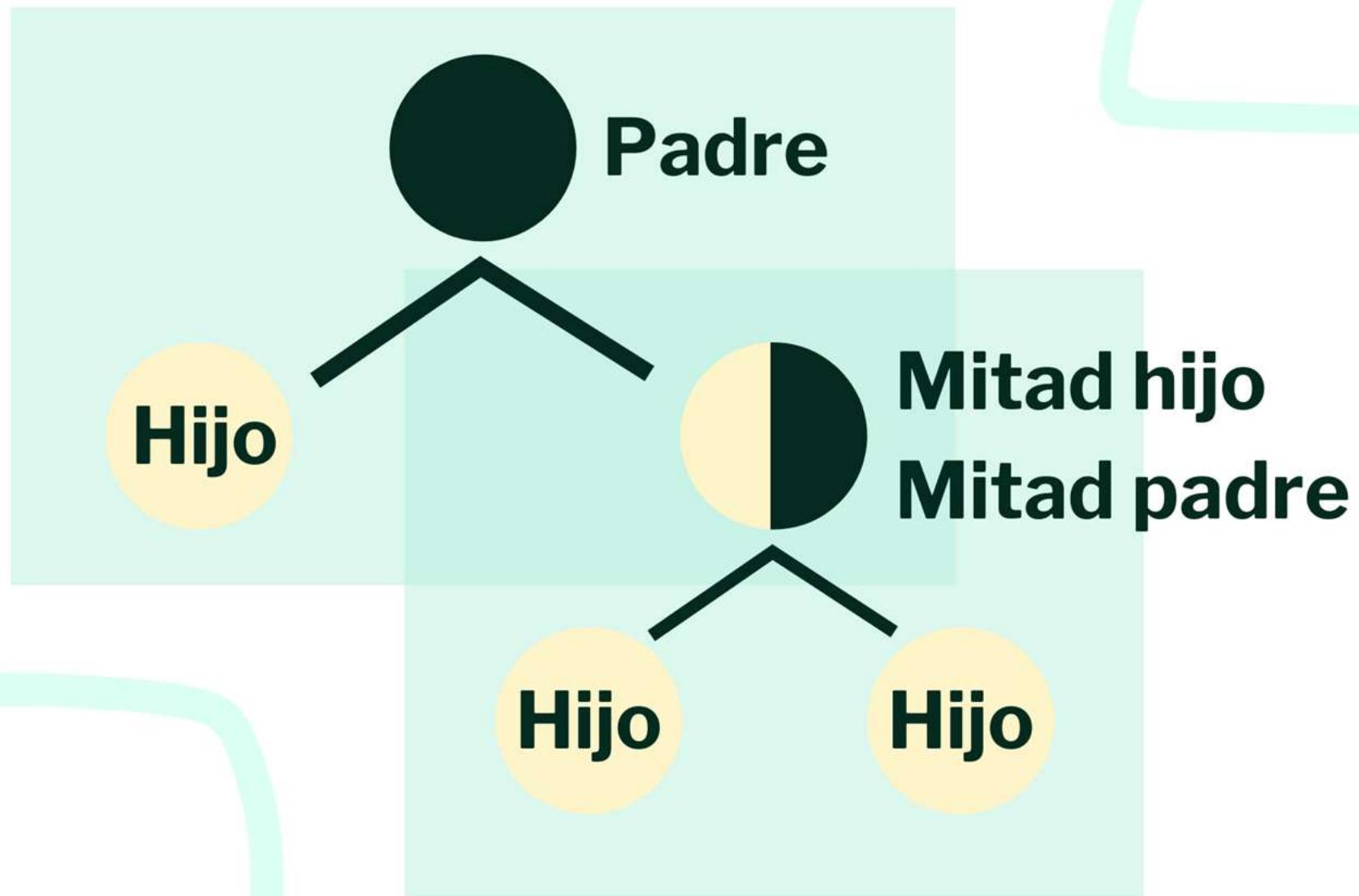
## Padre

```
<div class="container">  
  <div class="item"></div>  
  <div class="item"></div>  
  <div class="item"></div>  
  ...  
</div>
```

## Hijos



**Los hijos también  
pueden ser padres**





```
<div class="container">
  <div class="item">
    <div class="sub-item"></div>
    <div class="sub-item"></div>
  </div>
  <div class="item"></div>
  <div class="item"></div>
</div>
```

```
<div class="container">
  <div class="item">
    <div class="sub-item"></div>
    <div class="sub-item"></div>
  </div>
  <div class="item"></div>
  <div class="item"></div>
</div>
```

**Padre**

**Hijo-Padre**

The diagram illustrates the structure of a Document Object Model (DOM). It features a dark blue rectangular container representing the document. Inside, there are three main levels of nodes:

- Padre (Parent):** The outermost level, represented by a green border around the entire container.
- Hijo-Padre (Child-Parent):** The middle level, represented by a yellow border around the inner content area. It contains several `<div class="item">` nodes.
- Hijos del Padre (Children of the Parent):** The innermost level, represented by a pink border around the individual node elements. It includes `<div class="sub-item">` nodes nested within the `<div class="item">` nodes.

Arrows point from the text labels to their corresponding levels in the DOM structure. The code itself shows a hierarchical structure:

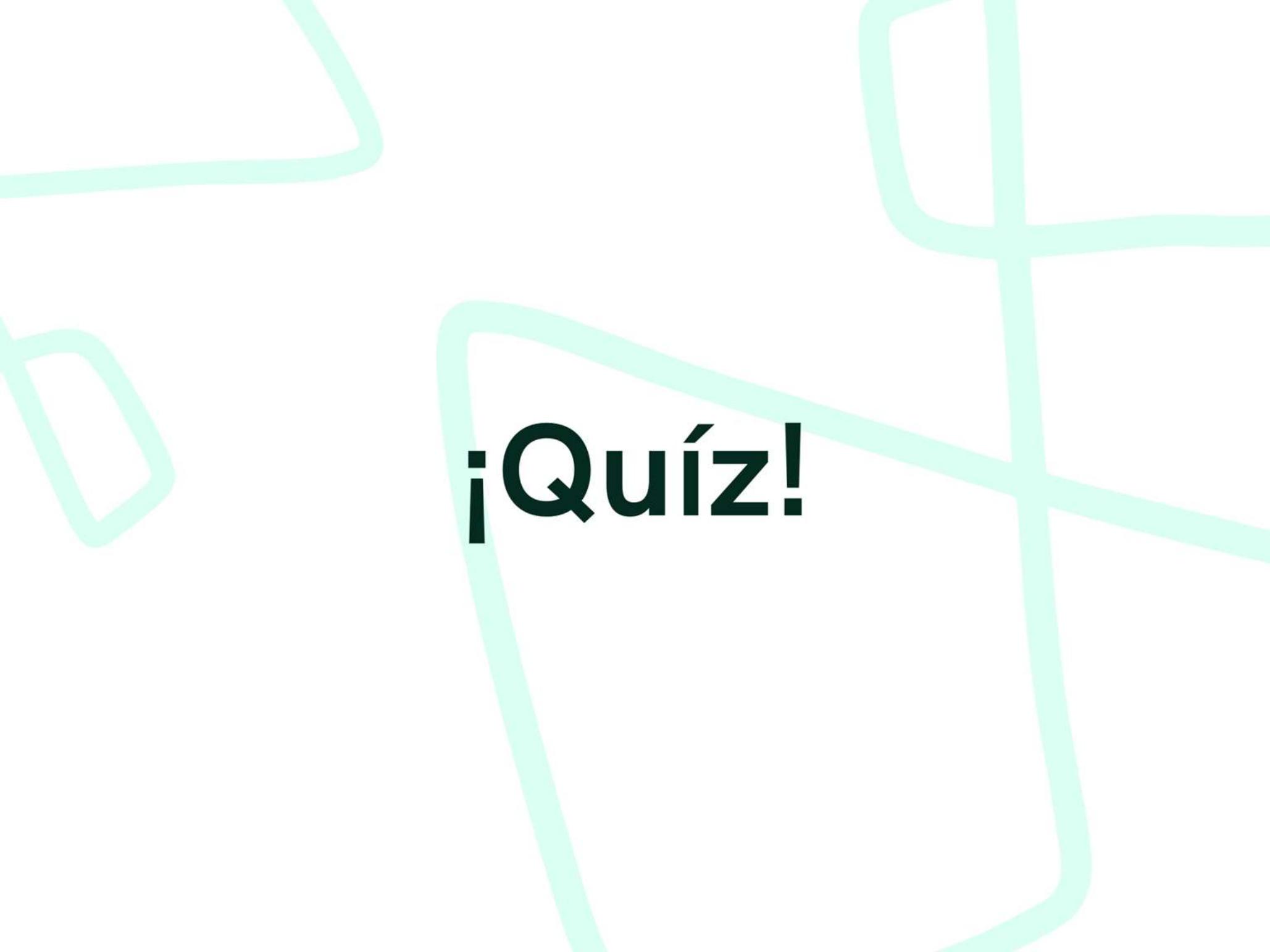
```
<div class="container">
  <div class="item">
    <div class="sub-item"></div>
    <div class="sub-item"></div>
  </div>
  <div class="item"></div>
  <div class="item"></div>
</div>
```

**Hijos del  
Hijo Padre**

**Hijos del  
Padre**

# Para tener en cuenta:

Todos los padres DEBEN tener la propiedad display y el valor grid

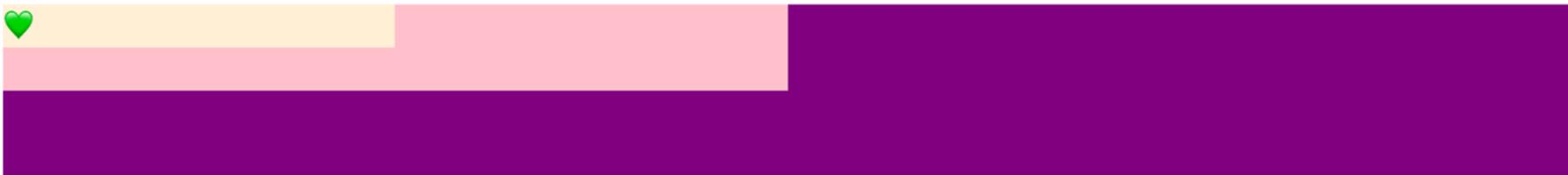
The background features abstract, light blue, organic-shaped lines and curves that intersect and overlap each other.

**¡Quíz!**



← → ⌂

⋮





**Lines, tracks,  
cell, area,  
gutters, grid axis,  
grid row, grid  
column + Reto**

**Importante:  
Relaciones entre  
padre e hijos  
inmediatos**

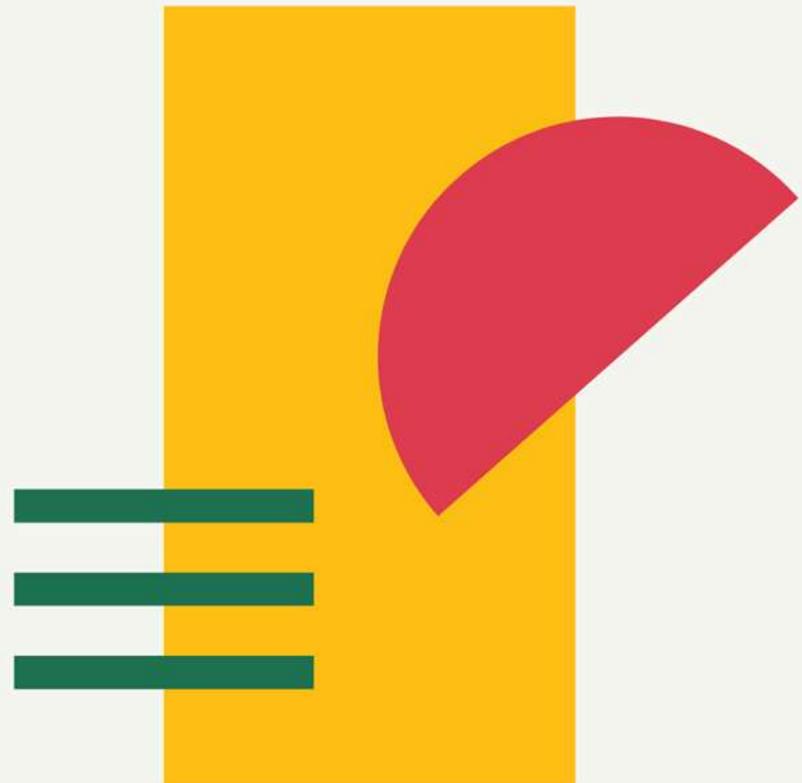
# **Dividiremos la clase de hoy en 2:**

(más nuestro reto al final)

Lines, tracks,  
cell, area

Gutters, grid  
axis, grid row,  
grid column

**Lines,  
track,  
cell,  
area**



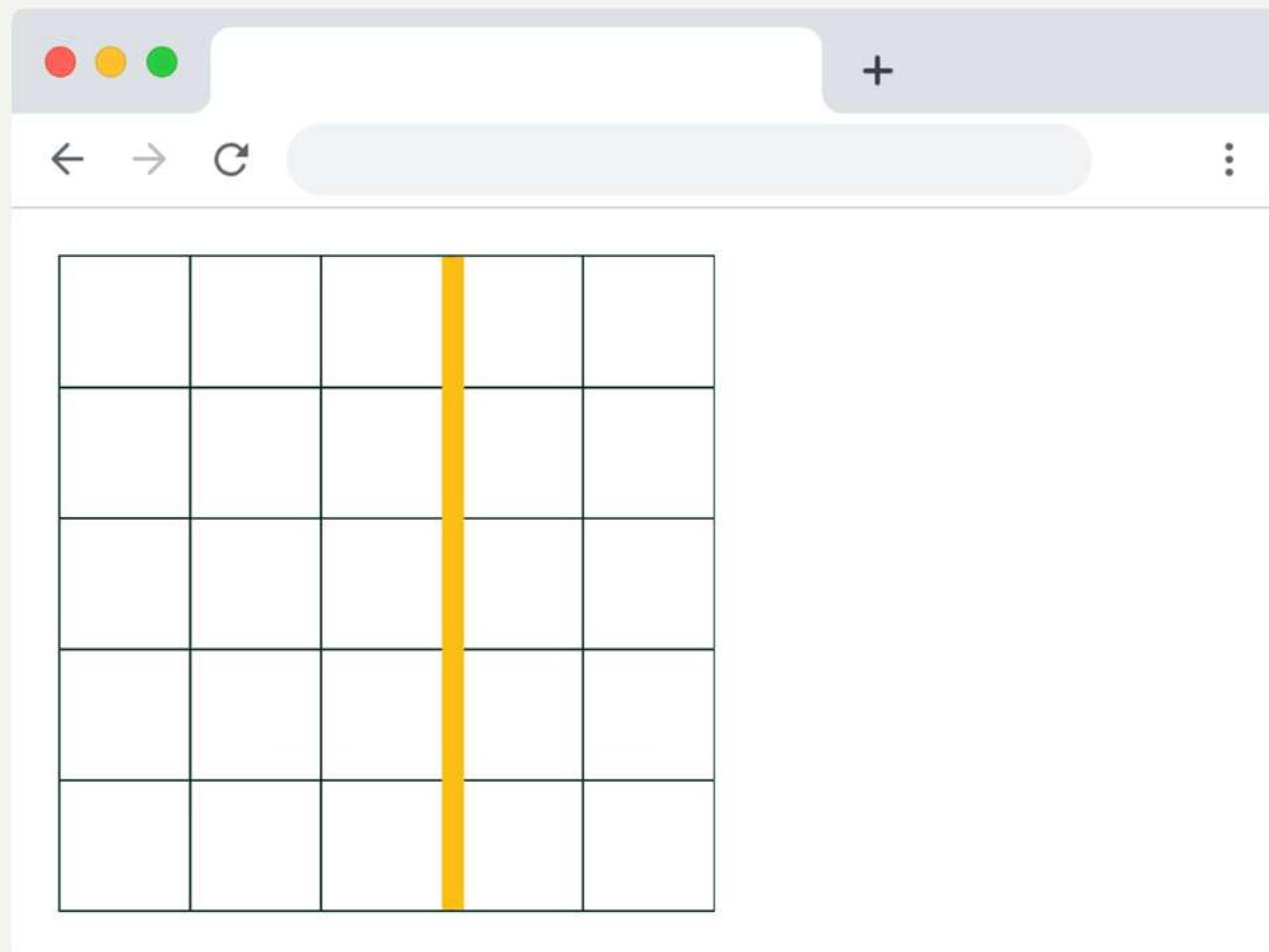
# Padre

# Hijos

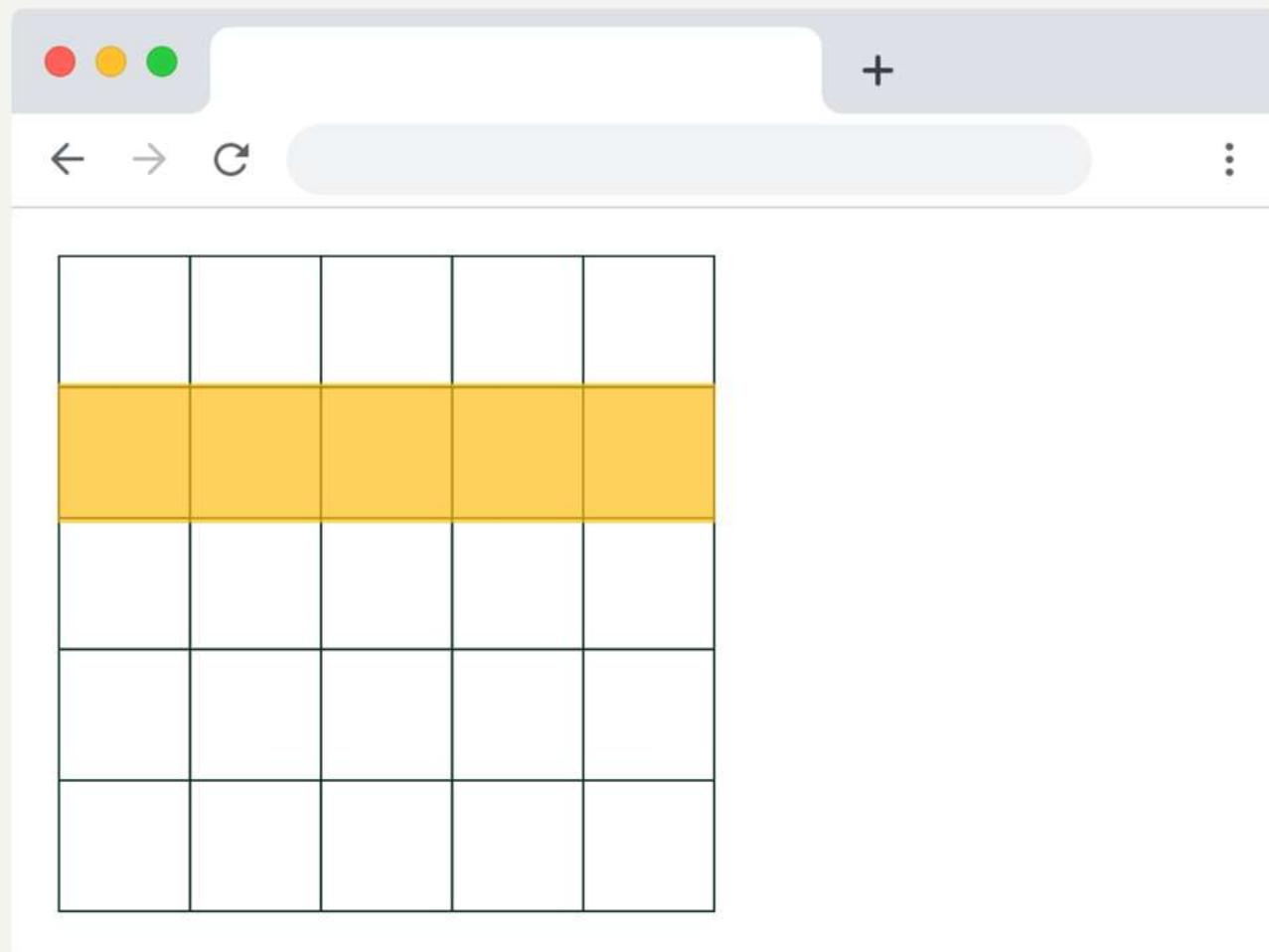
- Lines
- Track
- Area

- Cell

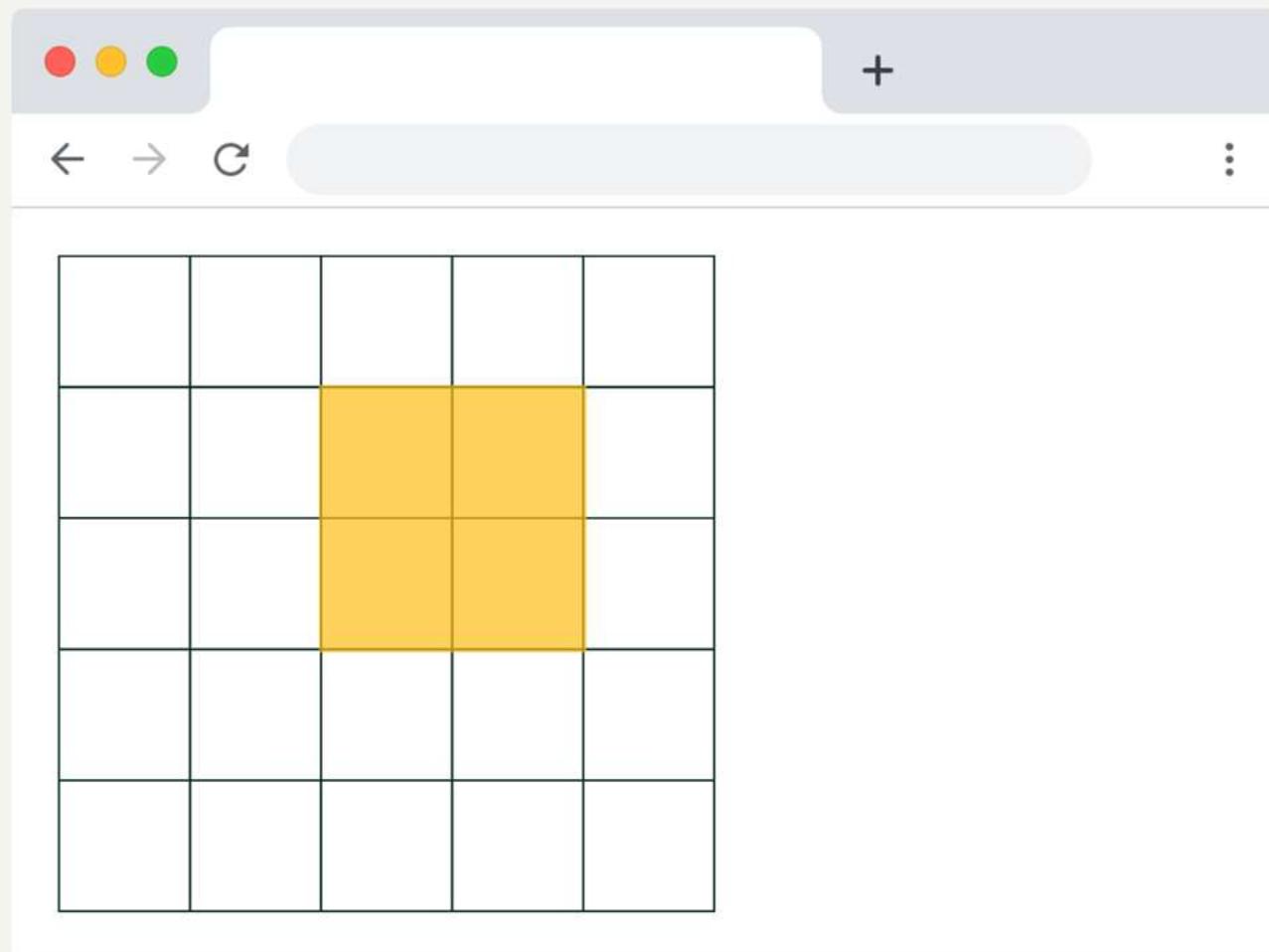
# Lines • Padre



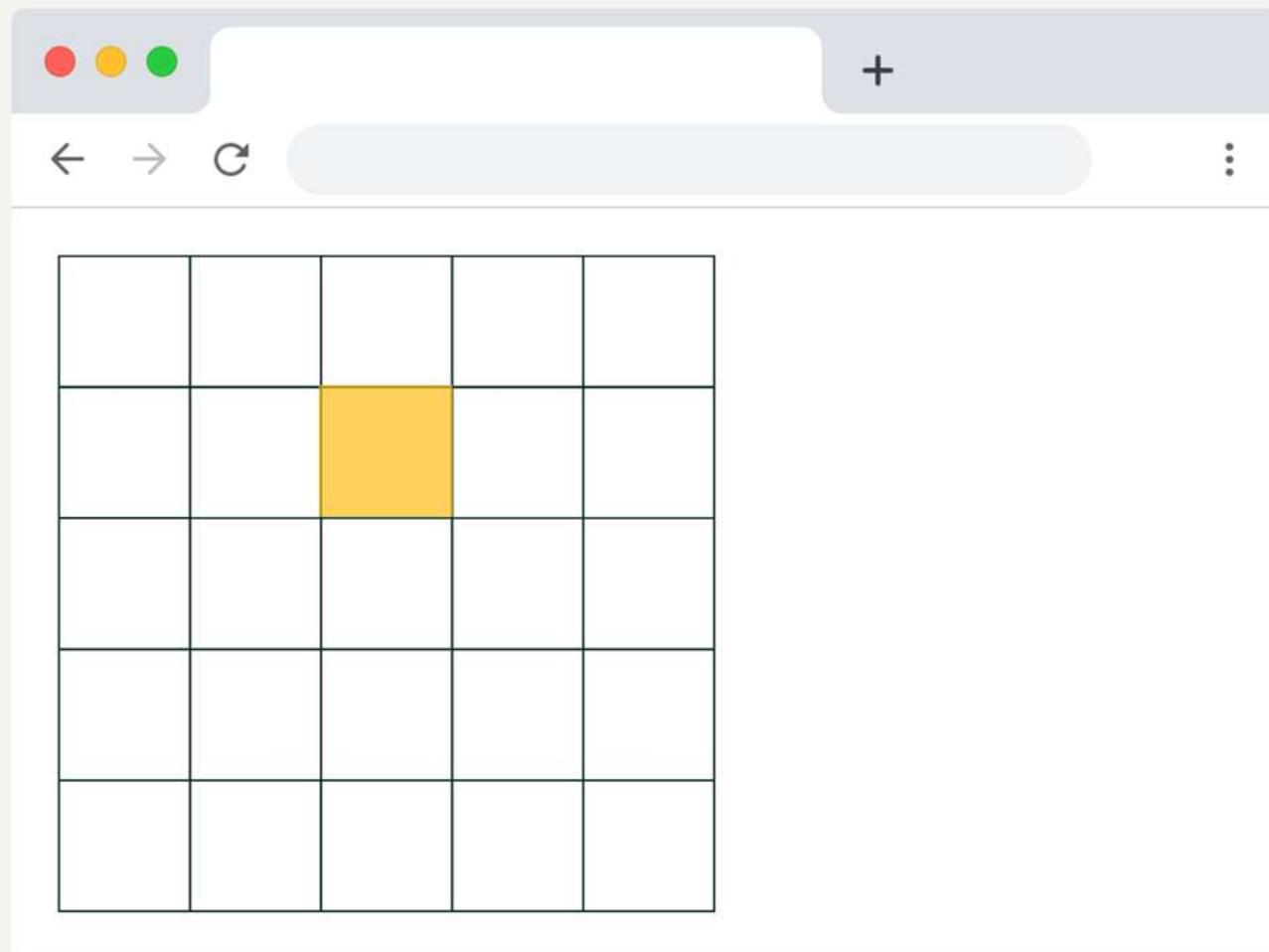
# Track • Padre



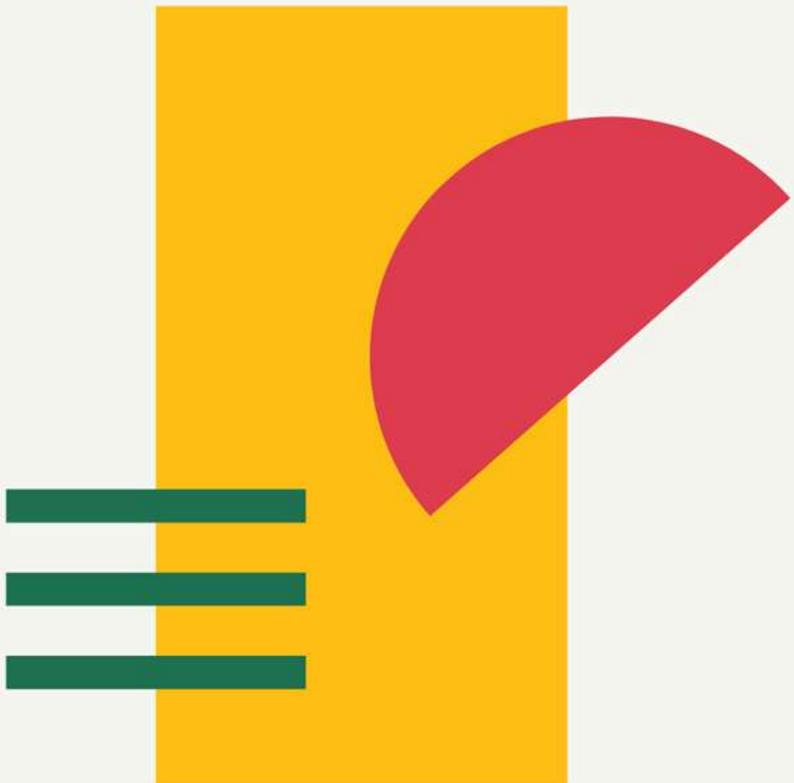
# Area • Padre



# Cell • Hijos



# Gutters, grid axis, grid row, grid column



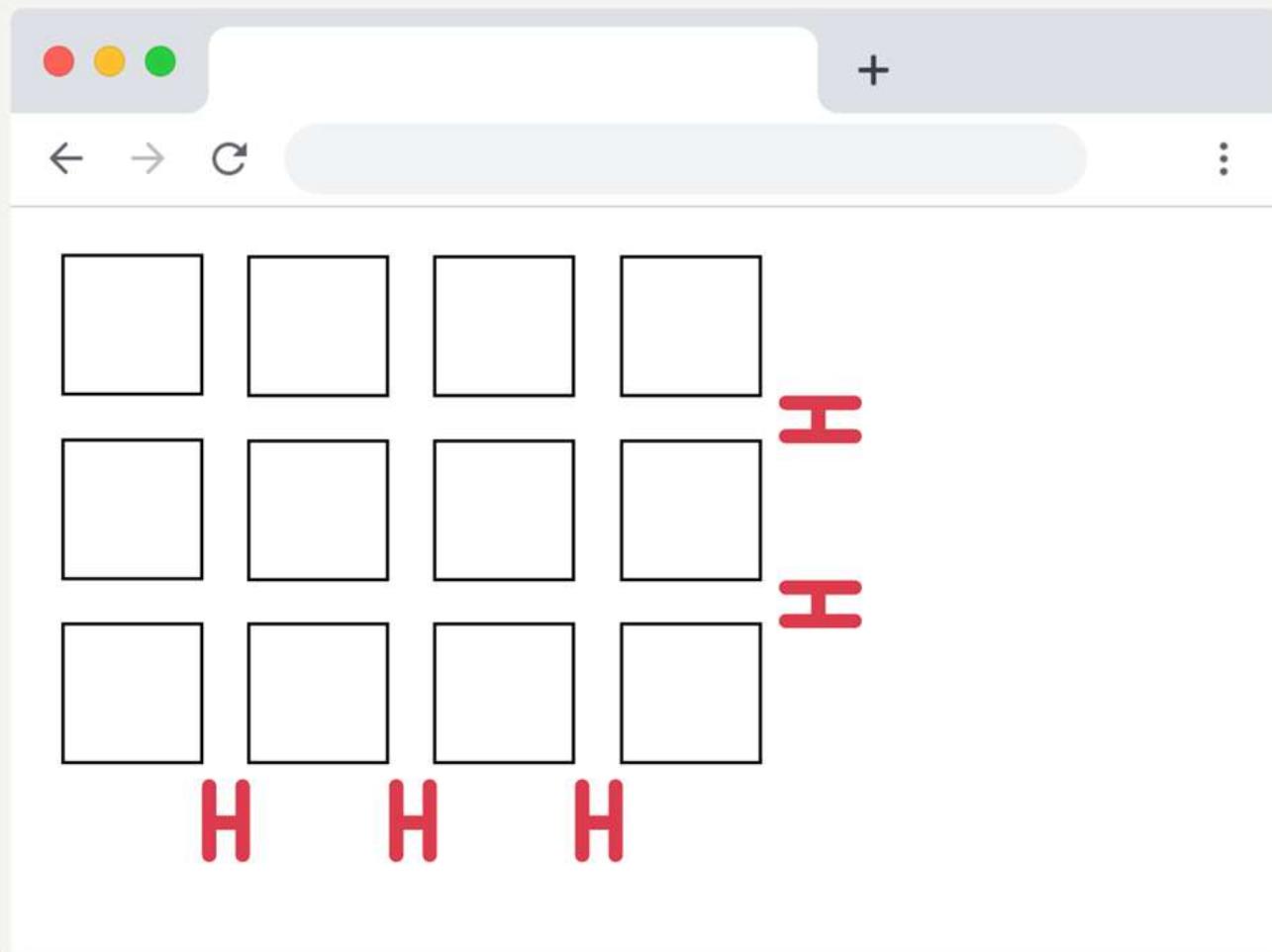
# Padre

- **Gutters**
- **Grid axis**

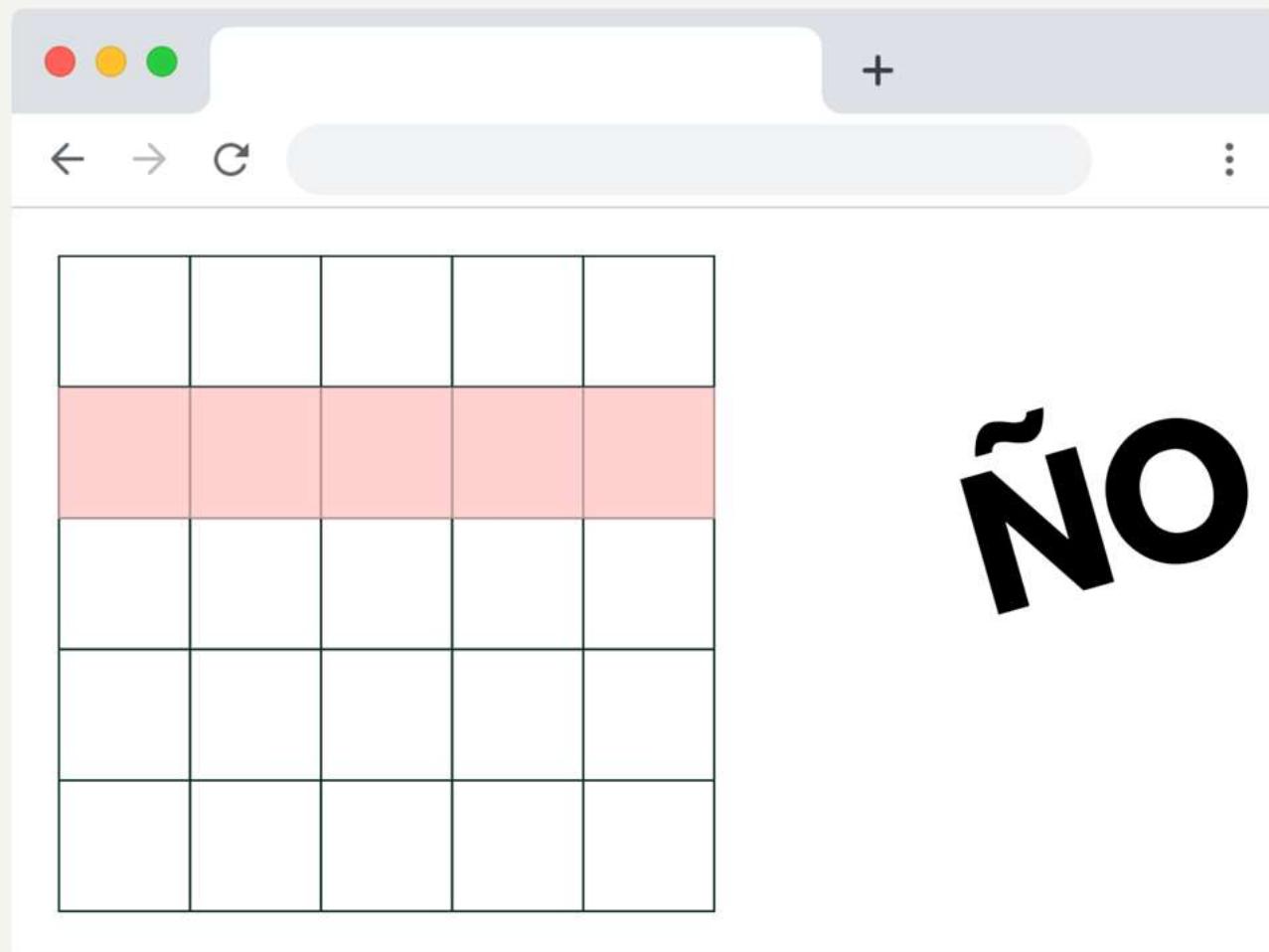
# Hijos

- **Grid row**
- **Grid column**
- **Grid axis**

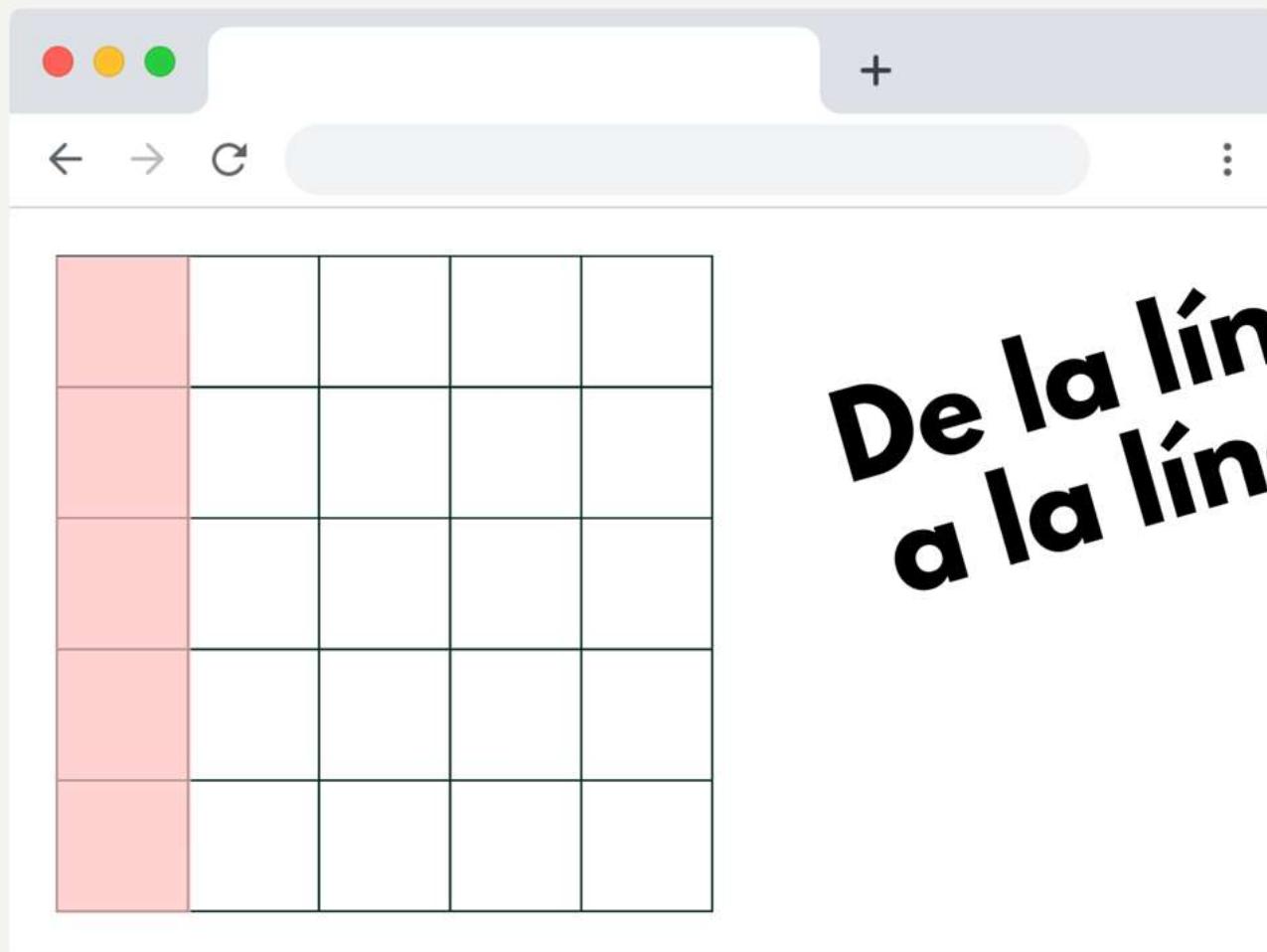
# Gutters• Padre



# Grid row • Hijos

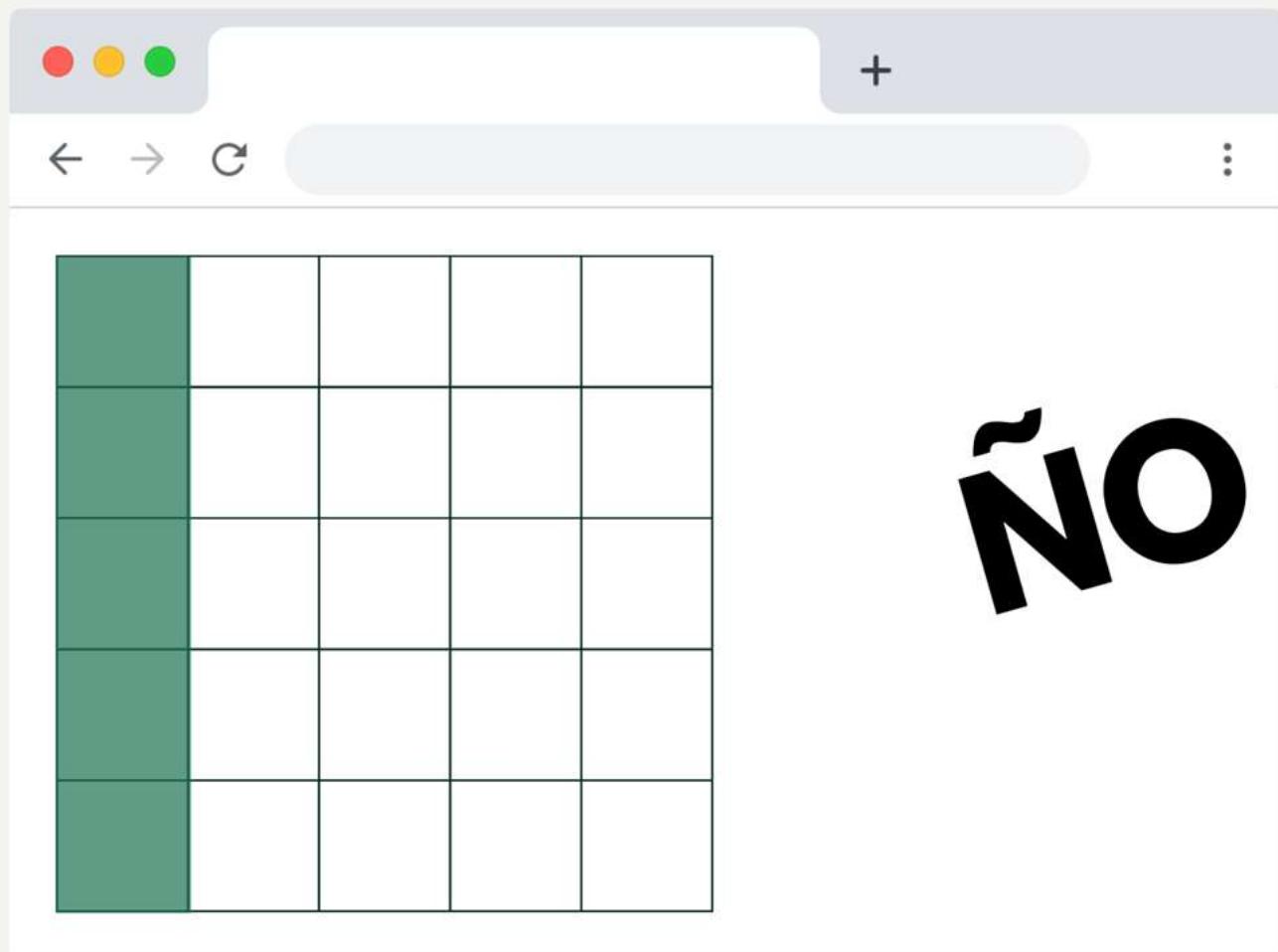


# Grid row • Hijos



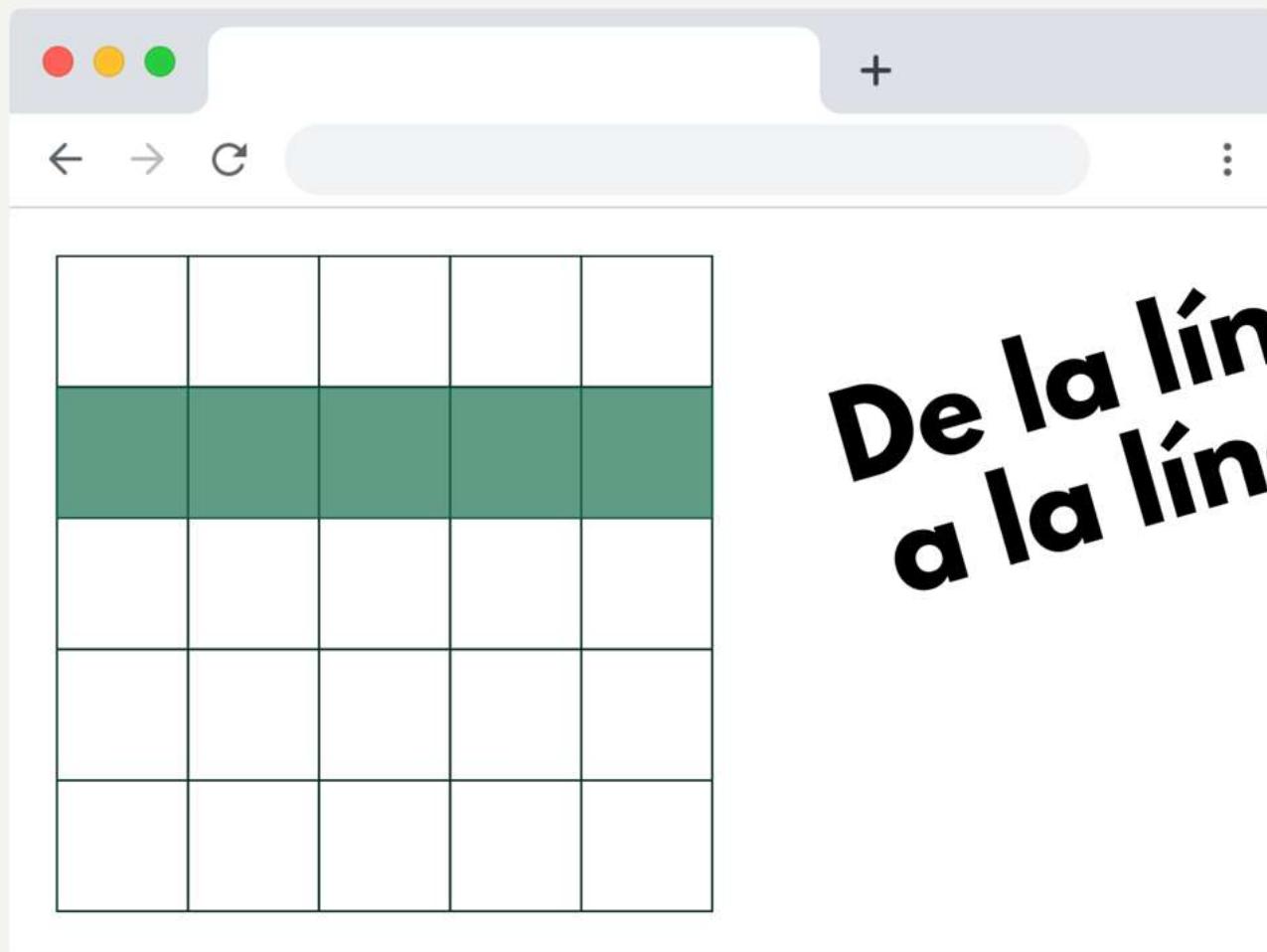
De la línea 1  
a la línea 6

# Grid column • Hijos



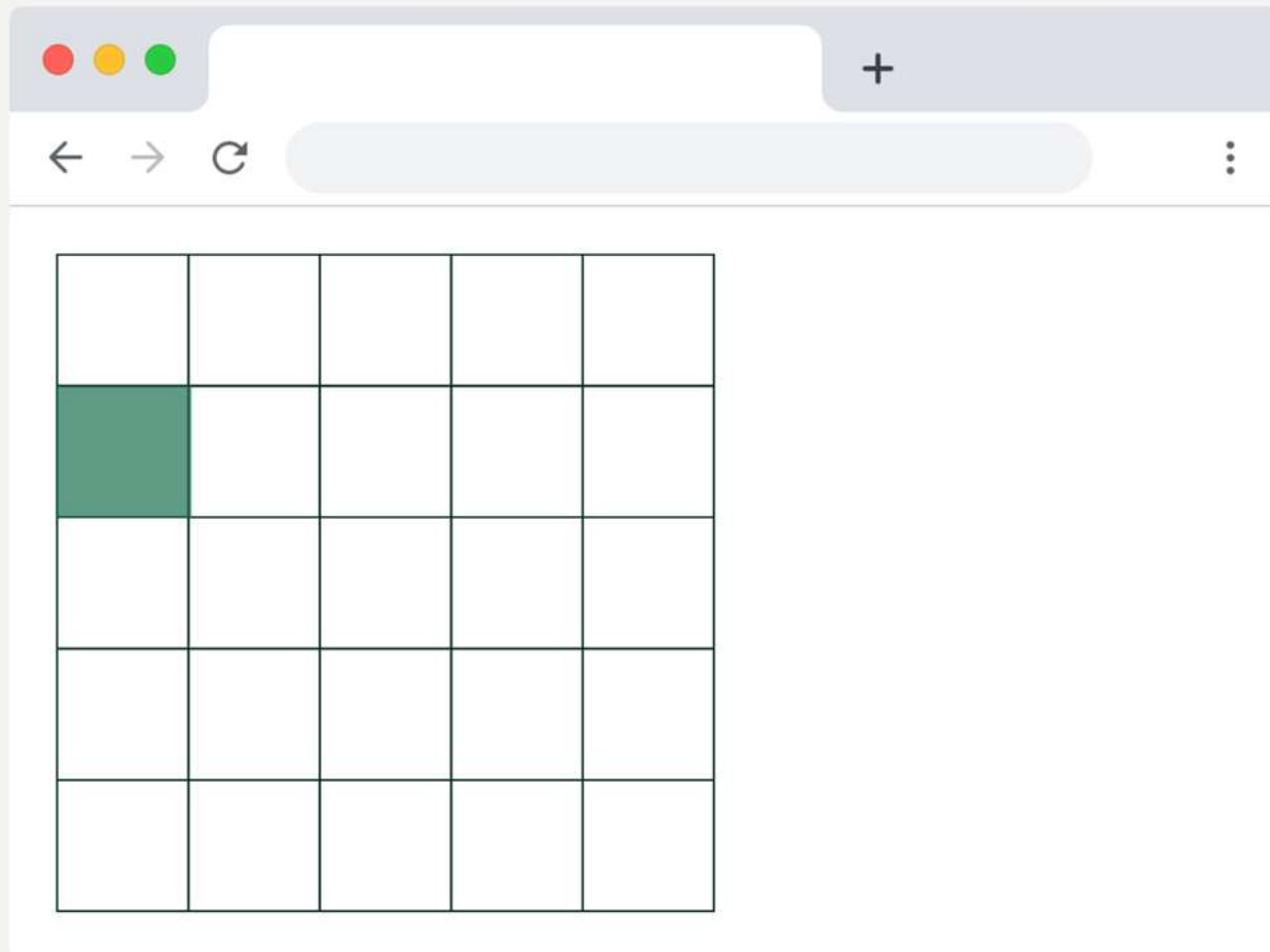
NO!

# Grid column • Hijos



De la línea 1  
a la línea 6

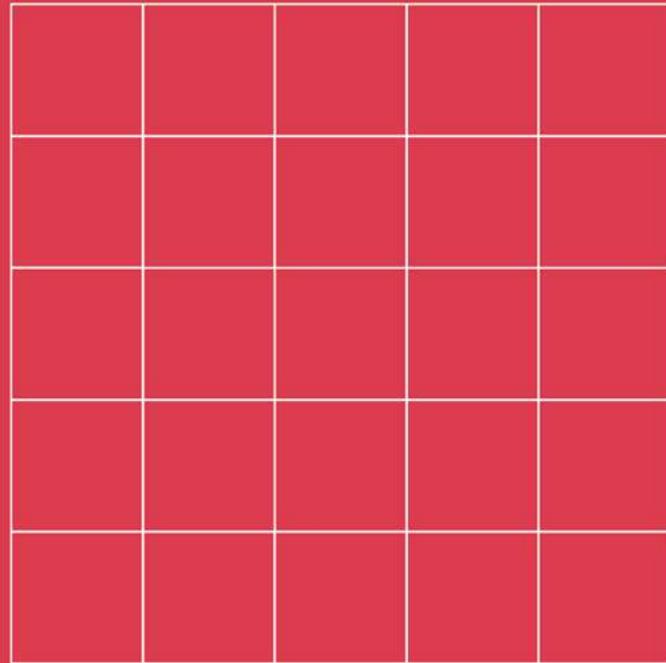
# Grid axis • Padre e hijos



# **Grid axis para temas de alineación**



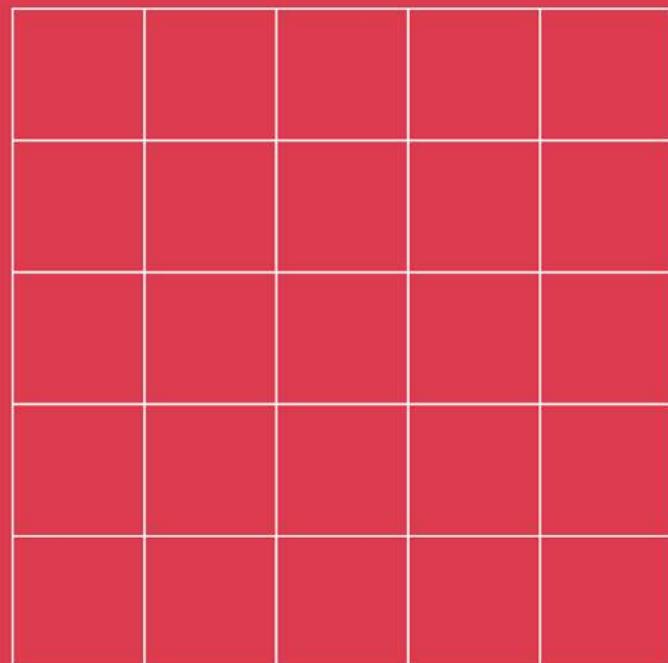
**row axis**





**Grid axis  
para temas  
de alineación**

**column axis**



**Miremos cómo nos  
pueden servir estos  
conceptos a la hora  
de trabajar con Grid  
en la vida real**



<https://www.airbnb.com.co/>



Medellin

4 de ene. de 2021 – 21 de ene. de 2021

2 huéspedes



Hazte anfitrión



Más de 300 alojamientos · 4 ene. - 21 ene. · 2 huéspedes

## Estadías en Medellín

Flexibilidad de cancelación

Tipo de alojamiento

Precio

Más filtros



Loft entero en Medellín

★UNIQUE VIEW★, Near LLERAS PARK



2 huéspedes · 1 habitación · 1 cama · 1 baño

Wifi · Cocina · Estacionamiento gratuito

Excepcional

★ 4.72 (29)

\$23 por noche

Total: \$407



Apartamento entero en Medellín

Apartamento Nuevo acogedor para una o dos pers...



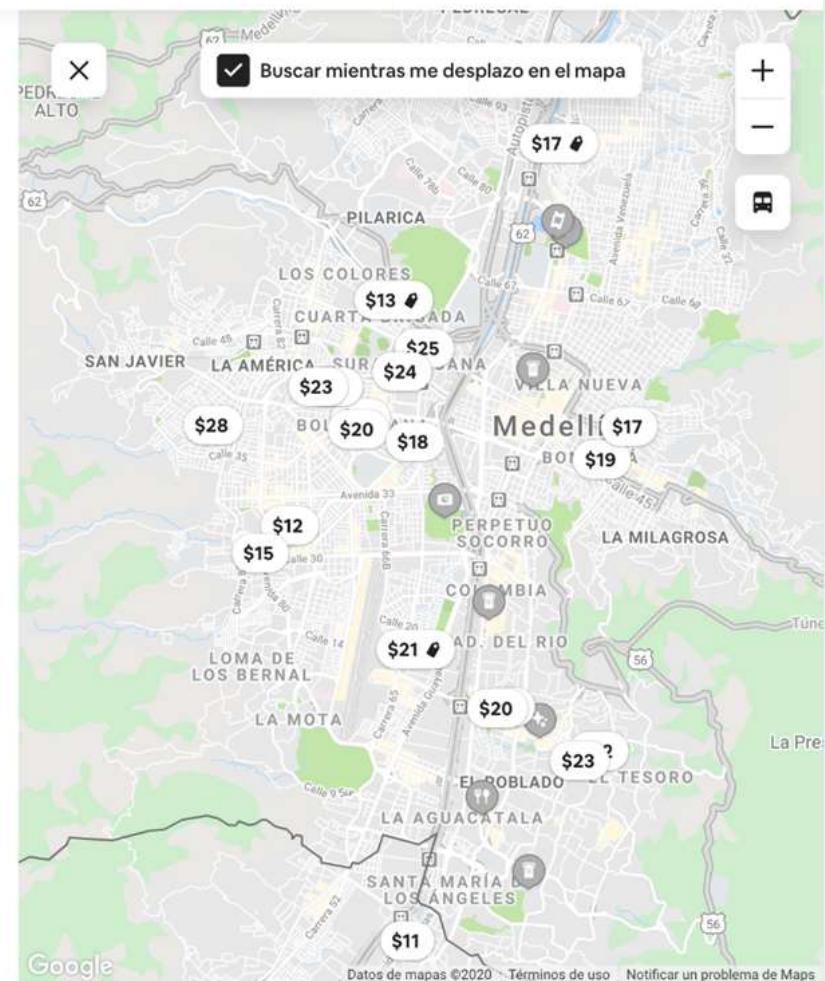
2 huéspedes · 1 habitación · 1 cama · 1 baño

Wifi · Cocina · Lavadora

★ 4.33 (3)

\$15 por noche

Total: \$251



# Imaginemos una grid en ese diseño





← → ⌂

<https://www.airbnb.com.co/>



Medellín

4 de ene. de 2021 – 21 de ene. de 2021

2 huéspedes



Hazte anfitrión



Más de 300 alojamientos · 4 ene. - 21 ene. · 2 huéspedes

## Estadías en Medellín

Flexibilidad de cancelación

Tipo de alojamiento

Precio

Más filtros



Loft entero en Medellín

★UNIQUE VIEW★, Near LLERAS PARK



2 huéspedes · 1 habitación · 1 cama · 1 baño

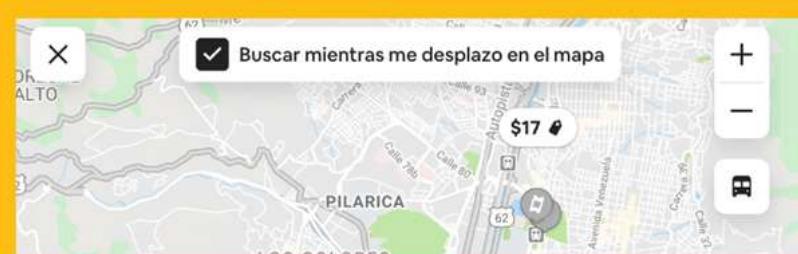
Wifi · Cocina · Estacionamiento gratuito

Excepcional

★ 4.72 (29)

\$23 por noche

Total: \$407



Apartamento entero en Medellín

Apartamento Nuevo acogedor para una o dos pers...



2 huéspedes · 1 habitación · 1 cama · 1 baño

Wifi · Cocina · Lavadora

★ 4.33 (3)

\$15 por noche

Total: \$251



Más de 300 alojamientos · 4 ene. - 21 ene. · 2 huéspedes

## Estadías en Medellín

Flexibilidad de cancelación · Tipo de alojamiento · Precio · Más filtros

Medellin | 4 de ene. de 2021 – 21 de ene. de 2021 | 2 huéspedes

Hazte anfitrión

Más de 300 alojamientos · 4 ene. - 21 ene. · 2 huéspedes

## Estadías en Medellín

Flexibilidad de cancelación · Tipo de alojamiento · Precio · Más filtros

Medellin | 4 de ene. de 2021 – 21 de ene. de 2021 | 2 huéspedes

Hazte anfitrión



<https://www.airbnb.com.co/>



Medellín

4 de ene. de 2021 – 21 de ene. de 2021

2 huéspedes



Hazte anfitrión



Más de 300 alojamientos · 4 ene. - 21 ene. · 2 huéspedes

## Estadías en Medellín

Flexibilidad de cancelación

Tipo de alojamiento

Precio

Más filtros



Loft entero en Medellín

★UNIQUE VIEW★, Near LLERAS PARK



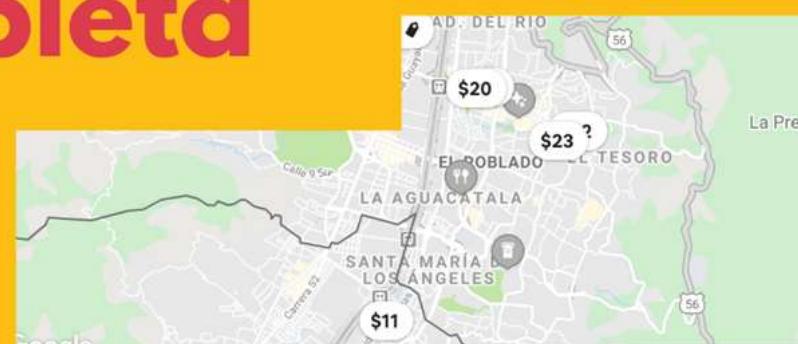
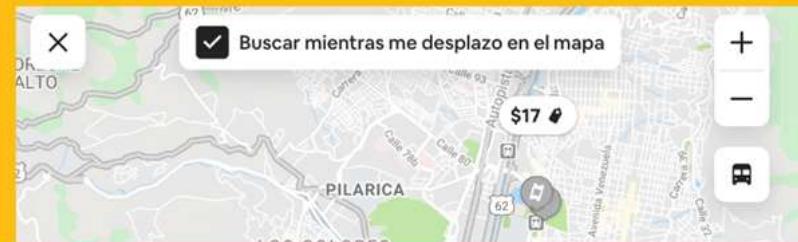
# Debemos hacer la Grid completa

2 huéspedes · 1 habitación · 1 cama · 1 baño  
Wifi · Cocina · Lavadora

★ 4.33 (3)

\$15 por noche

Total: \$251





<https://www.airbnb.com.co/>



Medellín

4 de ene. de 2021 – 21 de ene. de 2021

2 huéspedes



Hazte anfitrión



Más de 300 alojamientos · 4 ene. - 21 ene. · 2 huéspedes

## Estadías en Medellín

Flexibilidad de cancelación

Tipo de alojamiento

Precio

Más filtros



Loft entero en Medellín  
★UNIQUE VIEW★, Near LLERAS PARK



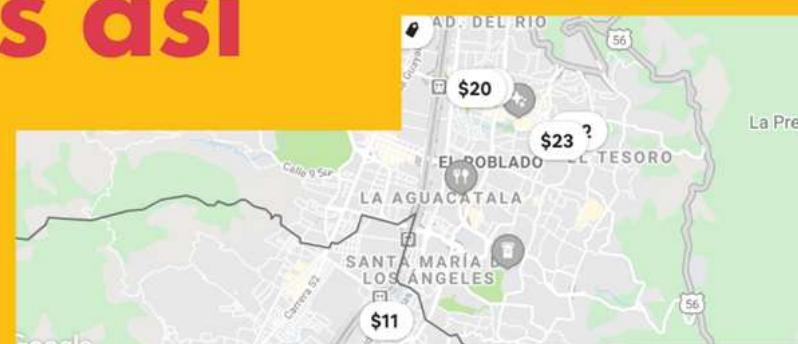
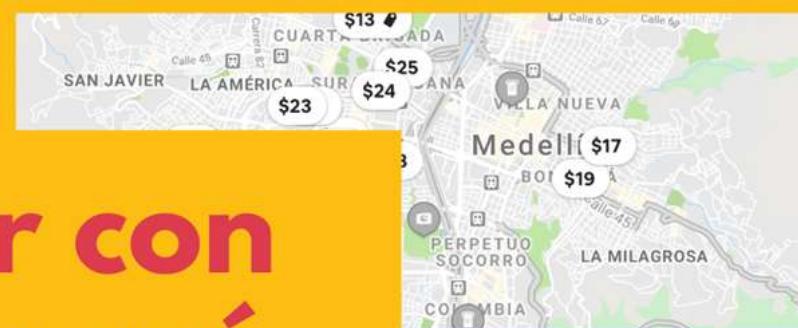
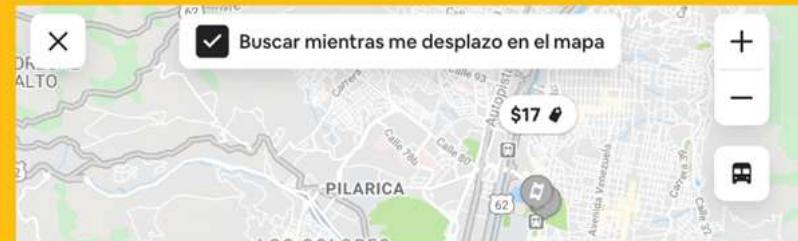
# Y no llegar con resultados así

2 huéspedes · 1 habitación · 1 cama · 1 baño  
Wifi · Cocina · Lavadora

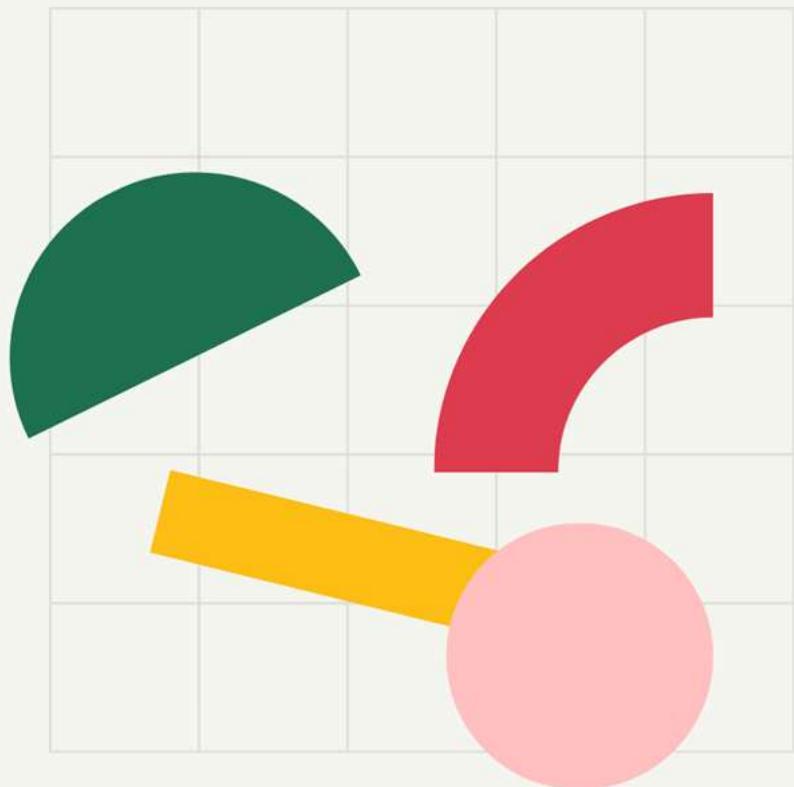
★ 4.33 (3)

\$15 por noche

Total: \$251



**Pero...  
faltan algunos  
componentes  
internos como  
cards, pills,  
search box, etc.**



# Separar cada componente



Apartamento entero en Medellín

Apartamento Nuevo acogedor para una o dos pers..



2 huéspedes · 1 habitación · 1 cama · 1 baño

Wifi · Cocina · Lavadora

★ 4.33 (3)

\$15 por noche

Total \$251



<https://www.airbnb.com.co/>



M **edellin** | 4 de ene. de 2021 – 21 de ene. de 2021 | 2 huéspedes

Hazte anfitrión

Más de 300 alojamientos · 4 ene. – 21 ene. · 2 huéspedes

**Estadías en Medellín**

Flexibilidad de cancelación

**SUPERANFITRIÓN**

Loft entero en Medellín \$17

★UNIQUE VIEW★ Near LAS PARK

2 huéspedes · 1 habitación · 1 cama · baño · Wifi · Cocina · Estacionamiento gratuito

Exceptional! \$23 por noche

★ 4.72 ( )

Apartamento entero en Medellín \$15

Apartamento Nuevo acogedor para una o dos pers.

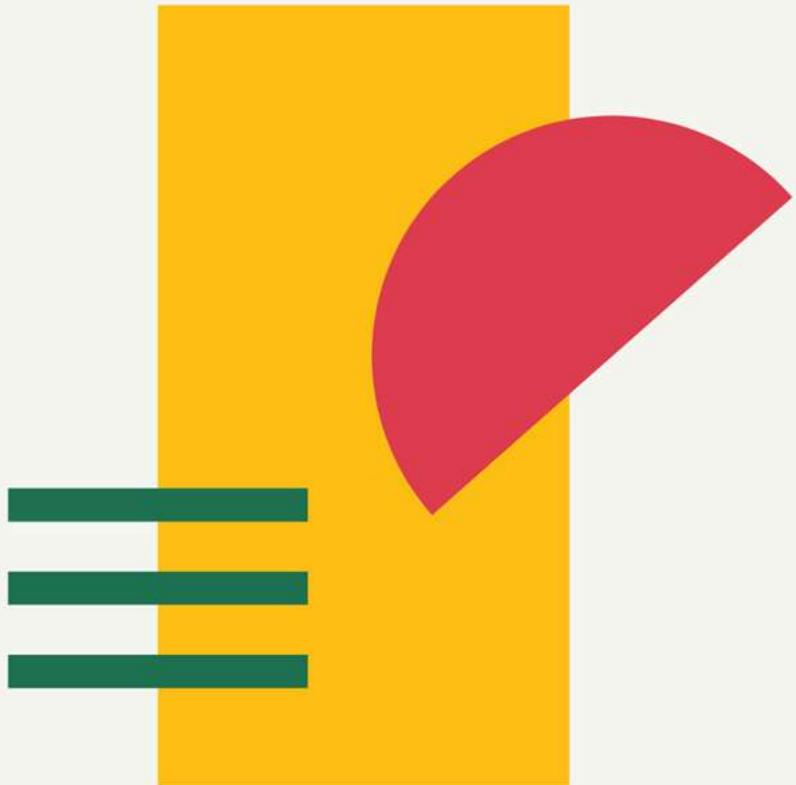
2 huéspedes · 1 habitación · 1 cama · baño · Wifi · Cocina · Lavadora

Exceptional! \$15 por noche

★ 4.33 ( )

**NO!**

**Te reto !**





<https://www.alpina.com/>



amigos con la boca abierta ¡descubrelas aquí! #CocinemosJuntos



Sostenibilidad Nutrición Productos Recetas Tiendas Alpina Empleo



Regístrate Login



Obleas con Arequipe Alpina y Queso Holandés Alpina

⌚ 10 Min.



Bites de Mac & Cheese con Quesos Alpina

⌚ 30 MIN.



Bisques de Mariscos con Crema de Leche Alpina

⌚ 40 Min.



Alitas al Ajo en Airfryer con Queso Parmesano Alpina

⌚ 20 Min.

[Ver más recetas](#)

Opcional !

# *¡Iniciemos nuestro proyecto!*

FASE DE CREATIVIDAD E IDENTIFICACIÓN DE ELEMENTOS

*Vamos a  
construir una  
web súper  
creativa  
inspirándonos  
en revistas*





***¿Dónde busco  
inspiración?***



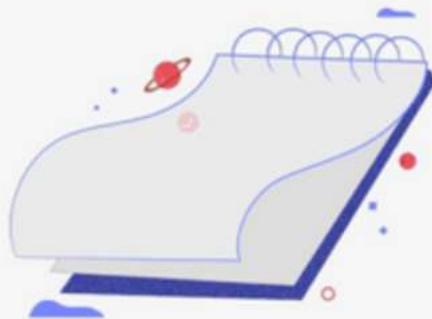


## Curso de Técnicas para Desarrollar tu Creatividad

Instruido por:  Nadia Michelle

 Básico  2 horas de contenido

[Ver la ruta de aprendizaje](#) ▾



[Proyecto del curso](#)

### Ideas a la medida

Escoge uno de los dos briefs propuestos para desarrollar ideas de forma estructurada y utiliza las herramientas del curso para seleccionar la idea ganadora que desarrollarás en el Curso para Probar tus Ideas: Prototipos y Testing.



## Curso de Diseño para Programadores

Instruido por:  Samanta Martínez

 Básico  2 horas de contenido

[Ver la ruta de aprendizaje](#) 



[Proyecto del curso](#)

### ¡Crea una página web!

Crearás la página web de un restaurante con todo el aprendizaje que obtendrás en este curso.

# *Aplicaciones que usaremos*



# Magazine design inspiration



Inicio • Hoy Siguiendo

magazine design inspiration



Todos los Pines



Graphic designers

Creativity

Layout

Fashion

Digital

Cover

Food

Travel

Modern

Typography

Editorial

Templates

Ideas



Templates de magazine gratuits dans InDesign | Creative blog...



About



Transpassar – Poetics of movement



VOLUME TWO



Create Beautiful Drop Caps in InDesign - InDesign Skills



엔디 리먼터 / Andy Reemerter 8p 편집디자인 - 그래픽 디자인, 브랜딩/편집



Ladypreneur eBook template | CANVA



# Asymmetrical grid design

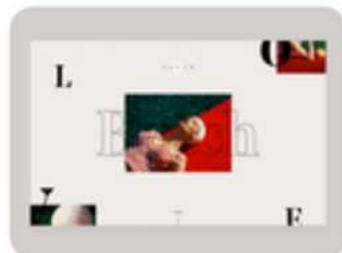
Inicio • Hoy Siguiendo asymmetrical grid design

X

Todos los Pines ▾



...



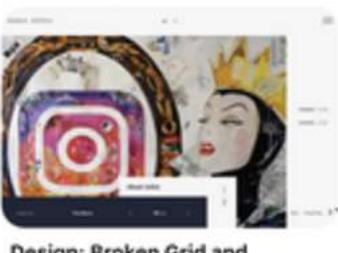
3 ways to experiment with latest UI trend—asymmetrical grids...



propagandism - typo/graphic posters



3 ways to experiment with latest UI trend—asymmetrical grids...



Design: Broken Grid and Overlapping Elements



3 ways to experiment with latest UI trend—asymmetrical grids...



3 ways to experiment with latest UI trend—asymmetrical grids...



3 ways to experiment with latest UI trend—asymmetrical grids...



3 ways to experiment with latest UI trend—asymmetrical grids...



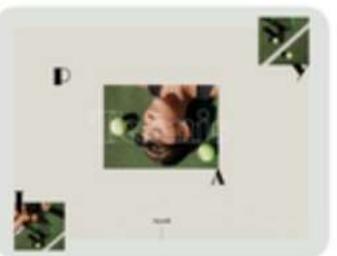
3 ways to experiment with latest UI trend—asymmetrical grids...



6 fantastic editorial designs and what we can learn from...



Grid

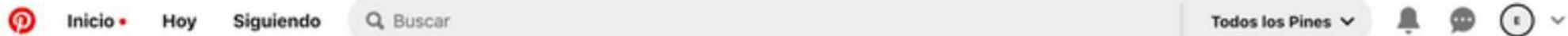


Canada Creators - About Us

+

?

# Organiza tu moodboard



## Magazine · Platzi Project

...

E +

🔒 Tablero secreto



Más ideas



Organizar



Notas



Holos Reviews Section  
Animation



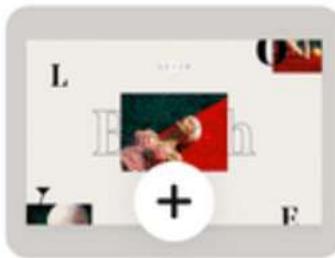
8 UI design trends for 2020



Dada-Data  
Krystle Wright



3 ways to experiment with latest  
UI trend—asymmetrical grids...



?



**veamos  
algunas ideas**



# Idea 1

INSTAGRAM  
@ZEKA\_DESIGN

WWW.ZEKAGRAPHIC.COM



DECONSTRUCTION

ZEKA  
DESIGN

WWW.ZEKAGRAPHIC.COM

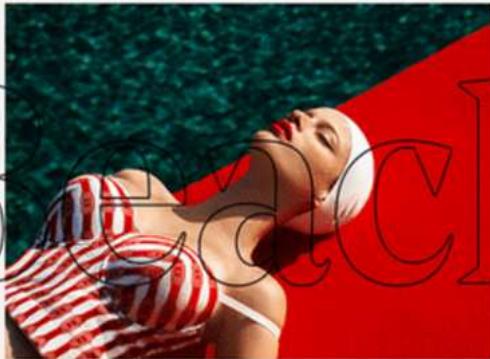
02

L

D R E A M



B e a c h



Scroll

E

Idea 2

# Idea 3

cure

Menus

About & Careers

Contact Us

Book an Appointment

## Our Services



# Idea 4

Z. Work

ABOUT ME SOME CASES

0 1 0 2

PLAY SHOWREEL

IN RED

0 7 0 8 0 9

Workflow

ALL THE PROCESSES

Contact me

FOR ANY COLLABORATIONS

CLICK CLICK

Art direction  
Digital production  
Branding

Dribbble Behance Twitter

IN LIGHT

The image shows a website layout for a creative professional. At the top left is a large, stylized title 'Idea 4'. Below it is a main content area with a large central photograph of a person in a dynamic, bent-over pose, wearing a dark coat and light-colored pants. The background features several large, semi-transparent orange circles of different sizes. To the left of the central image, the name 'Zhenya' is written in a large, white, serif font. To the right, 'Rynzhuk' is written in a similar style. On the far left, there's a navigation section with 'Z. Work' and links to 'ABOUT ME' and 'SOME CASES'. Below this are numbers '0 1' and '0 2'. In the bottom left corner, there's a link 'PLAY SHOWREEL'. On the far right, there's a 'Workflow' section with a link 'ALL THE PROCESSES' and a 'Contact me' section with a link 'FOR ANY COLLABORATIONS'. In the bottom right corner, there's a red starburst icon with the text 'CLICK CLICK' and a list of services: 'Art direction', 'Digital production', and 'Branding'. At the very bottom, there are links to social media platforms: 'Dribbble', 'Behance', and 'Twitter'. The overall aesthetic is clean and modern, with a focus on bold colors and geometric shapes.

# Idea 5



# *Hagamos una lista*

DE ELEMENTOS QUE NECESITAMOS



- ④ Temática
- ④ Figuras principales
- ④ Imágenes
- ④ Tipografía
- ④ Paleta de colores

# Temática

LLUVIA DE IDEAS

Comida

Música

Deportes

Libros

Juegos

# *Figuras principales*

LLUVIA DE IDEAS

Cuadros

Círculos

Líneas

Overlap

Letras

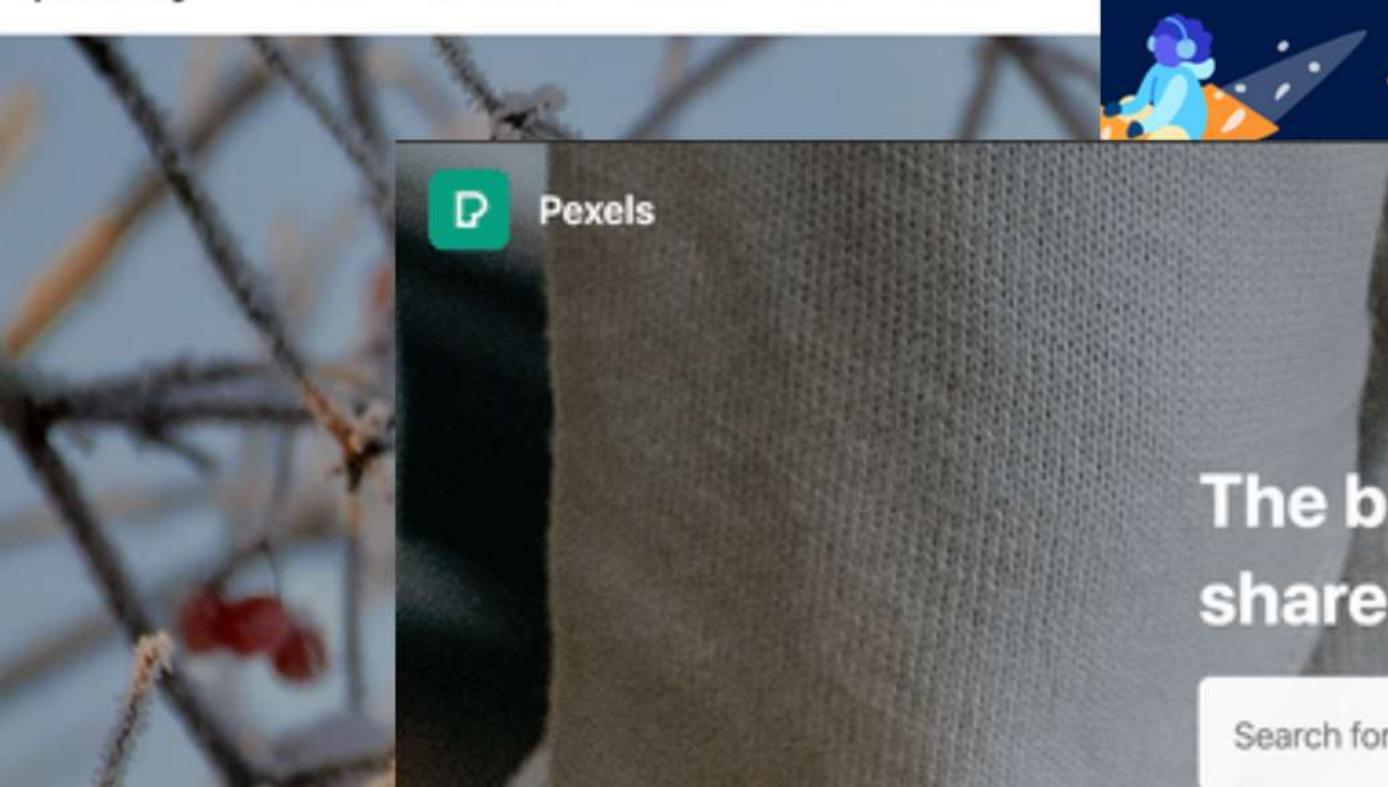
# Imágenes

QUE SEAN DE USO LIBRE



pixabay

Fotos Ilustraciones Vectores Videos Música



# Tipografía

QUE SEA DE USO LIBRE

Google Fonts

Browse fonts

Featured

Articles

About



Search

Sentence ▾ Type something

40px ▾



Categories ▾

Language ▾

Font properties ▾

Show only variable fonts (i)

1023 of 1023 families

Sort by: Trending ▾



Roboto

Christian Robertson

12 styles

Almost before we  
knew it, we had left  
the ground.

Langar

TypeLand, Alessia Mazzarella

1 style

Almost before we  
knew it, we had left  
the ground.

Andika New Basic

SIL International

4 styles

Almost before we  
knew it, we had left  
the ground.

# Paleta de colores

Color Hunt Palettes > New ▾



26

in 1 hour



Picker Convert ▾ Chart Names ▾ Library

COLOR NAMES

Modern browsers support 140 names which are listed below. Use them in your HTML and CSS by name, Hex color code or RGB value.

JUMP TO COLOR ▾

I NEW FEATURE! You can now create a gradient out of 3 colors!

Beta ColorSpace PALETTES GRADIENT 3-COLOR

Generate a CSS Color Gradient

Choose orientation

Enter colors



Temática

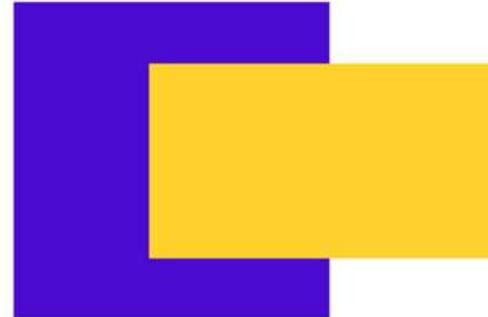


Figuras principales



Imágenes

# CSS Grid



ñ



Tipografía

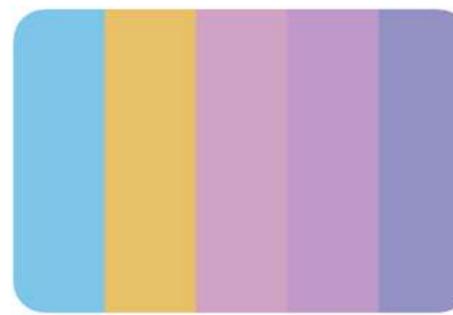
Roboto

Bold 700

Add more styles   Remove all

^

⊖



After Laughter Color Palette



Paleta de colores



# Creando nuestro contenedor: *display: grid o display: inline-grid?*



# **Definición en español de Display**

## **VERB:**

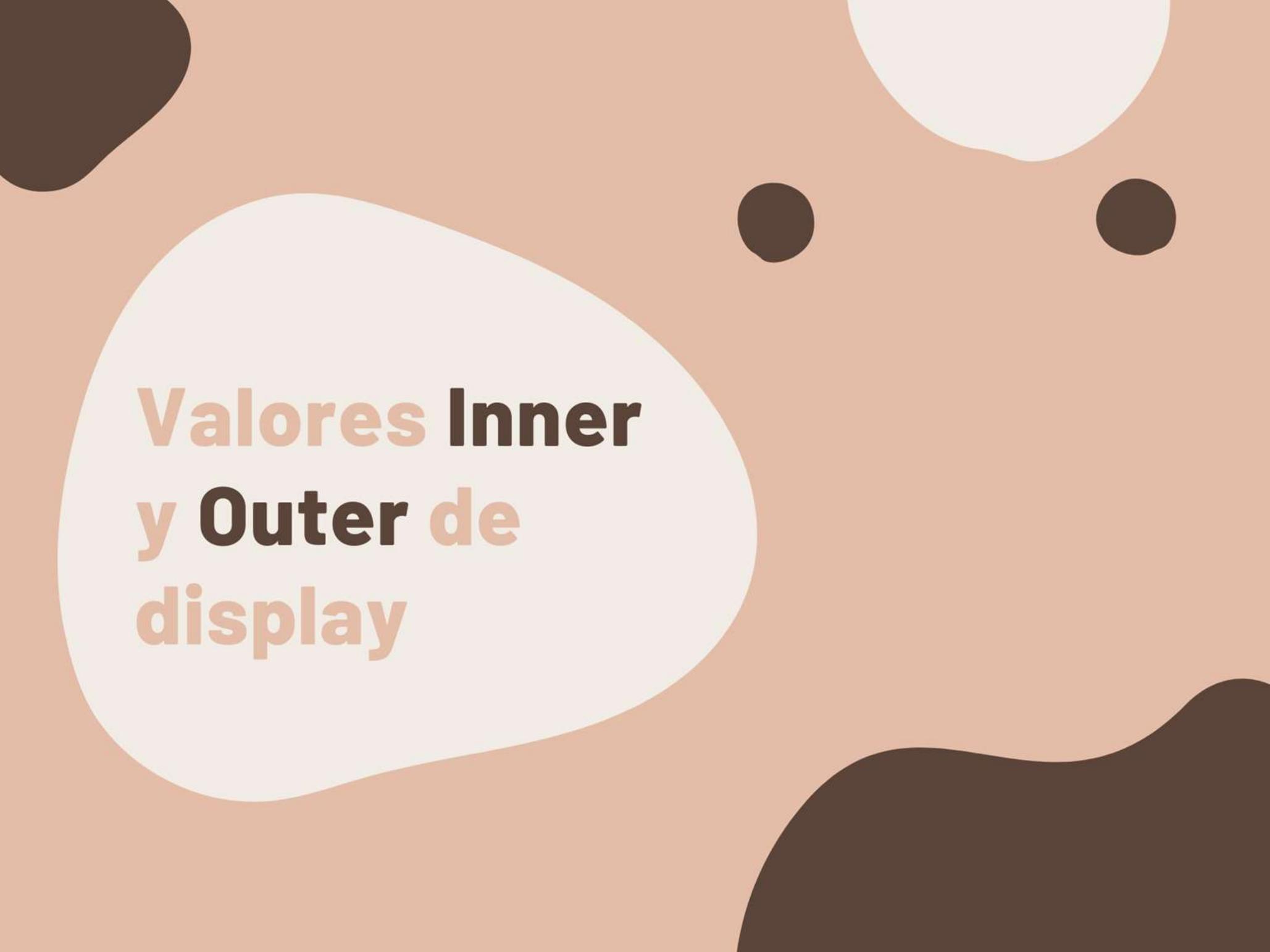
Desplegar, colocar a la vista, exhibir, lucir, mostrar, presentar.



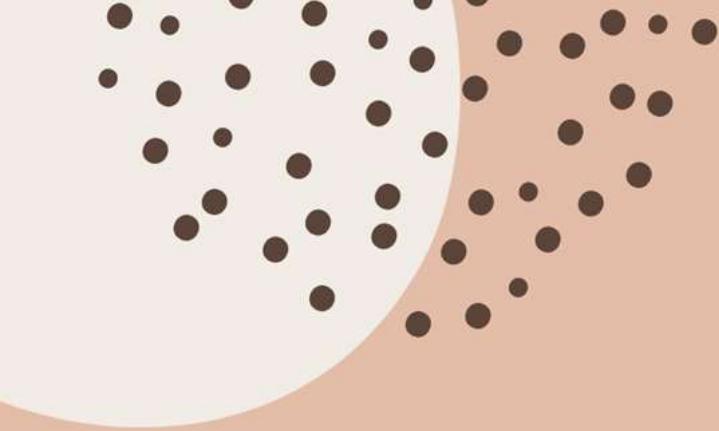


# Display

**Define el tipo  
de visualización  
de un elemento.**

The background features abstract organic shapes in shades of brown, tan, and white, resembling stylized leaves or petals.

**Valores Inner  
y Outer de  
display**



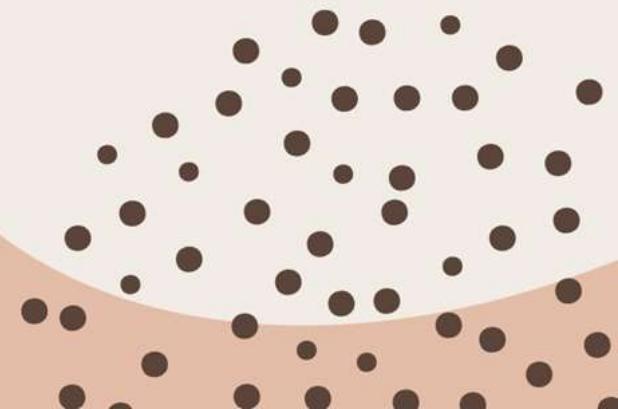
# ¿Cuál es la diferencia entre block y grid?



**Block**

**Grid**

**Los valores  
block e inline  
nos van a  
definir dos  
cosas:**



**Valor  
EXTERNO de  
visualización**

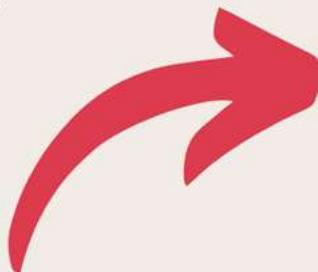
CÓMO SE  
COMPORTA EL  
ELEMENTO EN EL  
DISEÑO JUNTO  
CON OTROS  
ELEMENTOS

Valor  
**INTERNO** de  
visualización

CÓMO SE  
COMPORTAN  
LOS HIJOS  
DIRECTOS DE  
ESE ELEMENTO



Cuando  
dices:



***DISPLAY: GRID;***

lo que realmente  
estás diciendo es:



***DISPLAY: BLOCK GRID;***

(estás solicitando un contenedor  
grid a nivel de bloque)



# Un elemento que tenga todos los atributos de **bloque** puede darle:



margin y  
padding



width



height

“

**SIN EMBARGO,**  
*a los hijos de un  
contenedor **display: grid;**  
se les da un valor **interno**  
de **grid***

**Esta forma de  
pensar el **display**  
es realmente útil !**

**SI VES POR  
EJEMPLO**

**display: inline-flex;**  
**INMEDIATAMENTE  
PENSARÁS QUE SE  
COMPORTA COMO UN  
ELEMENTO EN LÍNEA.**

# Block, Grid y Flex

BLOCK  
GRID  
FLEX

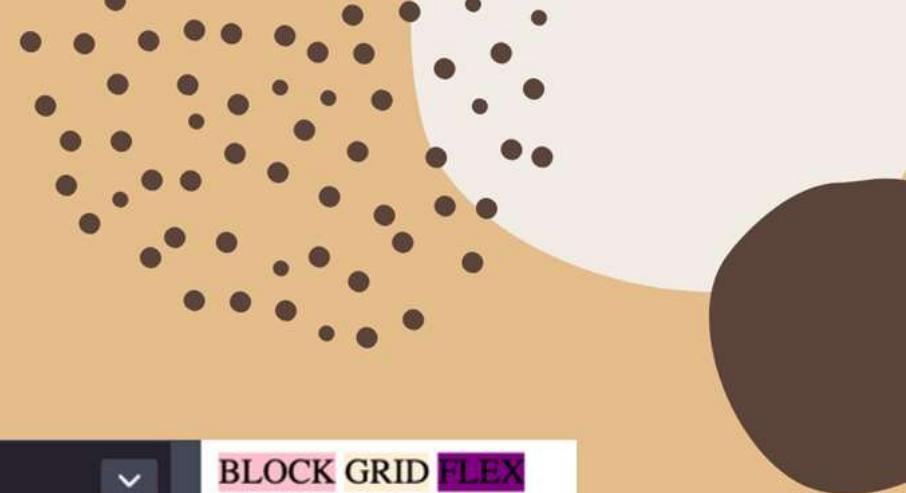
## HTML

```
1 <div class="block">BLOCK</div>
2 <div class="grid">GRID</div>
3 <div class="flex">FLEX</div>
```

## CSS

```
1 .block {
2   background: pink;
3   display: block;
4 }
5
6 .grid {
7   background: papayawhip;
8   display: grid;
9 }
10
11 .flex {
12   background: purple;
13   display: flex;
14 }
```

# inline, inline-grid e inline-flex



The slide features a decorative background on the right side consisting of a large dark brown circle and a cluster of smaller black dots arranged in a roughly triangular shape.

**HTML**

```
1 <div class="block">BLOCK</div>
2 <div class="grid">GRID</div>
3 <div class="flex">FLEX</div>
```

**CSS**

```
1 .block {
2   background: pink;
3   display: inline;
4 }
5
6 .grid {
7   background: papayawhip;
8   display: inline-grid;
9 }
10
11 .flex {
12   background: purple;
13   display: inline-flex;
14 }
```

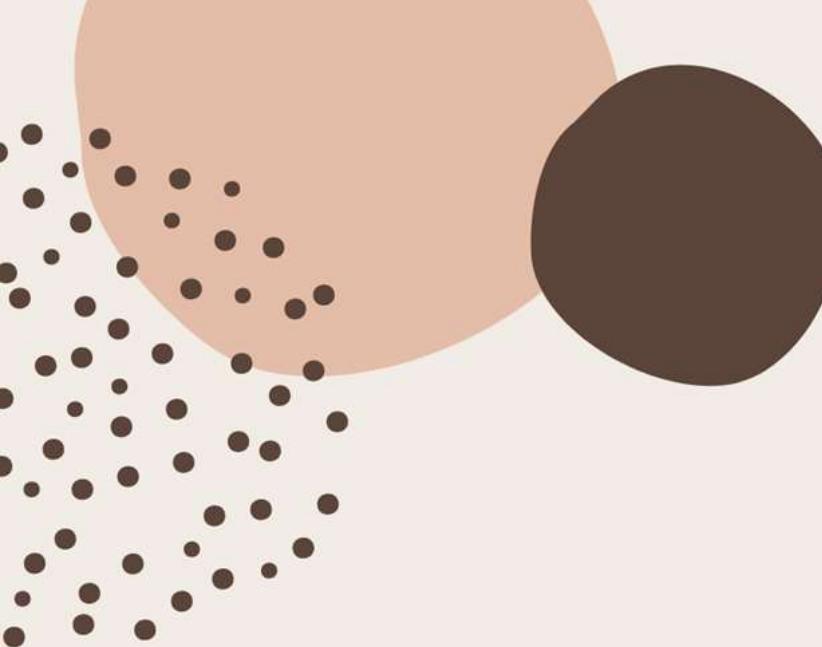
**BLOCK GRID FLEX**

**Siempre volvemos  
al flujo normal del  
documento  
(bloque o en línea)**

# En CSS2

**TENÍAMOS LO SIGUIENTE :**

- inline
- block
- inline-block
- list-item
- none
- table
- inline-table



## **SE AGREGARON:**

- flex
- inline-flex
- grid
- inline-grid

# **Y en CSS3**

“

Como ya veníamos  
desde **CSS2** con una  
sintaxis, no se  
modificó en **CSS3**.

Valor único	Valores de dos palabras clave	Descripción
block	block flow	Caja de bloque con flujo normal interior
flow-root	block flow-root	Cuadro de bloque que define un BFC
inline	inline flow	Caja en línea con flujo normal interior
inline-block	inline flow-root	Cuadro en línea que define un BFC
list-item	block flow list-item	Caja de bloque con flujo normal interior y caja de marcador adicional
flex	block flex	Caja de bloque con diseño flexible interno
inline-flex	inline flex	Caja en línea con diseño flexible interno
grid	block grid	Caja de bloque con diseño de cuadrícula interna
inline-grid	inline grid	Caja en línea con diseño de cuadrícula interna
table	block table	Caja de bloques con diseño de mesa interior
inline-table	inline table	Caja en línea con diseño de mesa interior



**Valor EXTERNO de  
visualización**



**display: block grid;**  
**display: inline-grid;**

**Valor INTERNO de  
visualización**





# Diferencias

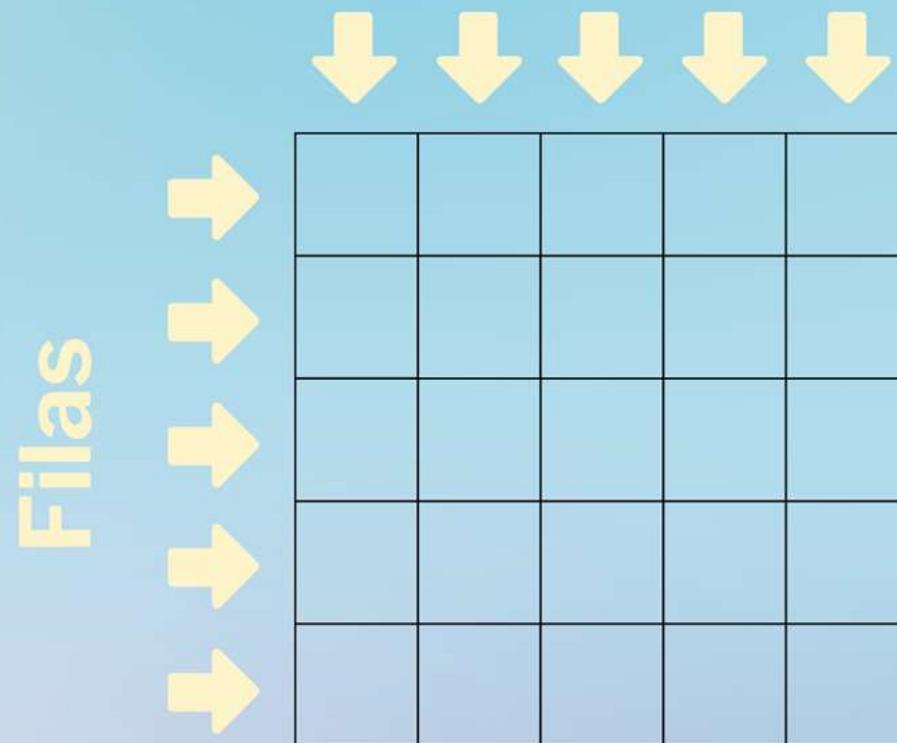
<b>block</b>	<b>inline</b>
?	?



# **Creando filas, columnas y espaciado + Reto**

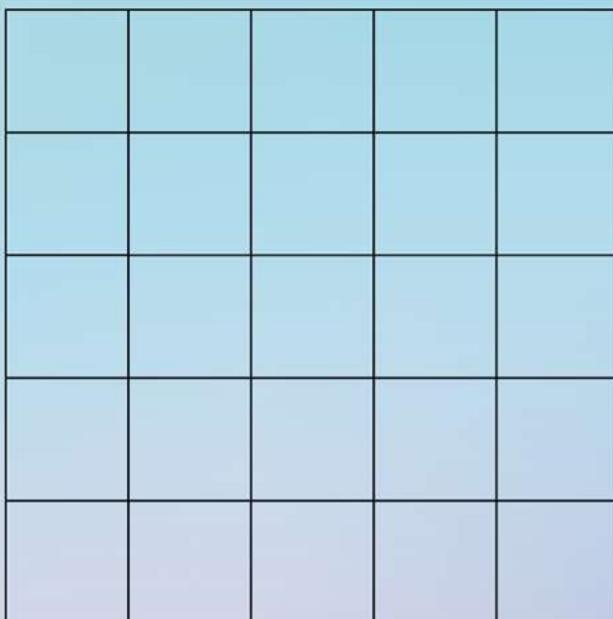
CSS GRID

# Columnas



# CSS Grid

# Columnas



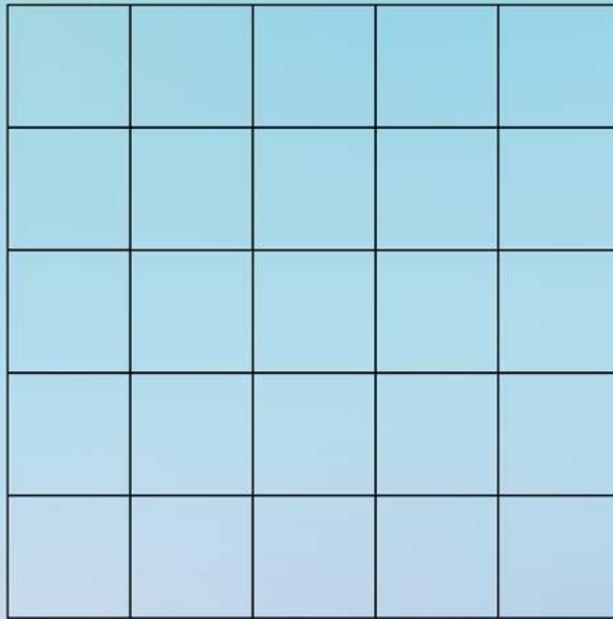
# Propiedad



grid-template-columns

# CSS Grid

**Filas**



**Propiedad**



`grid-template-rows`

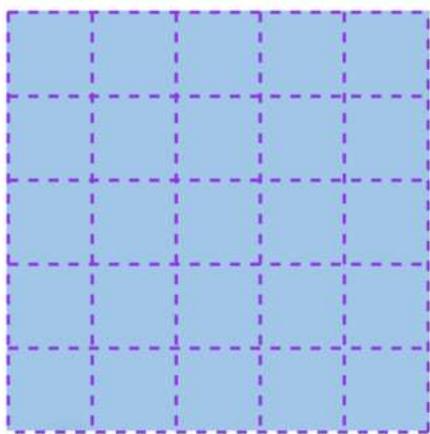
**CSS Grid**

## HTML

```
<div class="container"></div>
```

## CSS

```
.container {  
    display: grid;  
    grid-template-columns: 20px 20px 20px 20px 20px;  
    grid-template-rows: 20px 20px 20px 20px 20px;  
}
```



....

¿Conoces alguna forma en  
la que no tengamos que  
escribir tantas veces 20px?



```
.container {  
    display: grid;  
    grid-template-columns: 20px 20px 20px 20px 20px;  
    grid-template-rows: 20px 20px 20px 20px 20px;  
}
```

**de 20px**

**5 columnas**

```
● ● ●  
.container {  
    display: grid;  
    grid-template-columns: repeat(5, 20px);  
    grid-template-rows: repeat(5, 20px);  
}
```

# y mejor aún !



```
.container {  
    display: grid;  
    grid-template: repeat(5, 20px) / repeat(5, 20px);  
}
```

# y mejor aún!

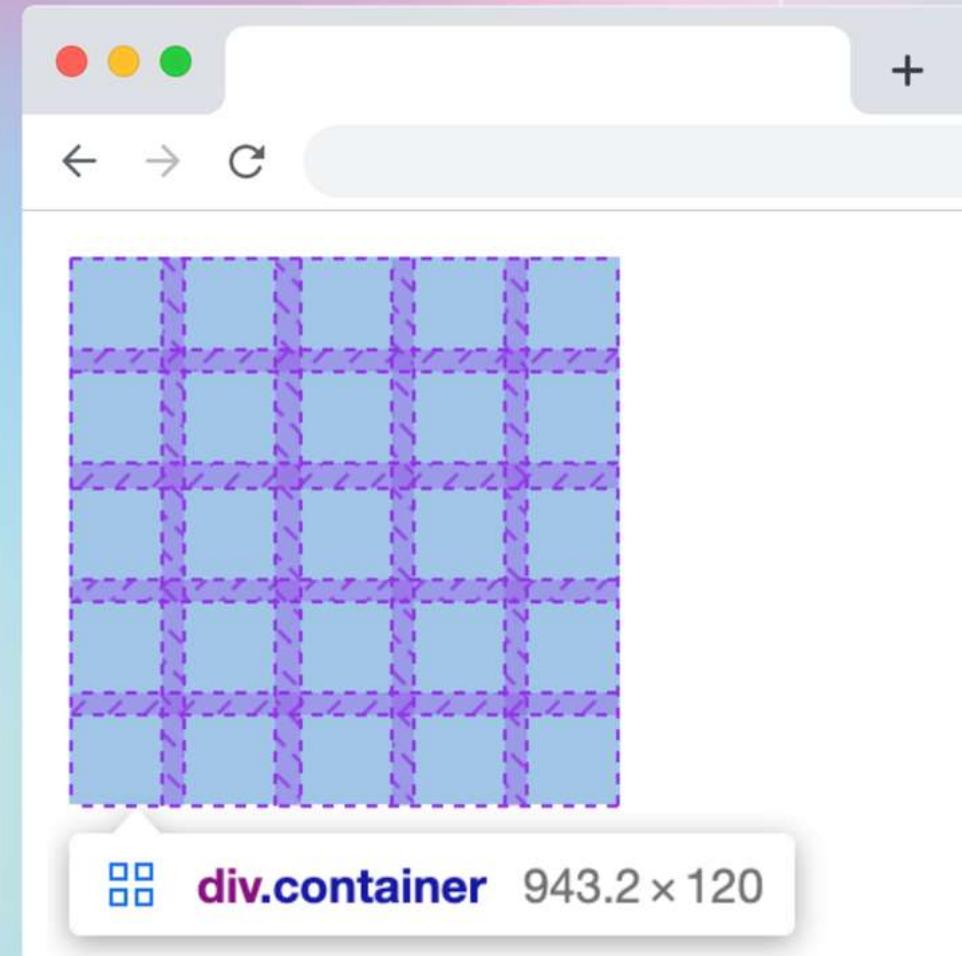


```
.container {  
  display: grid;  
  grid-template: repeat(5, 20px) / repeat(5, 20px);  
}
```



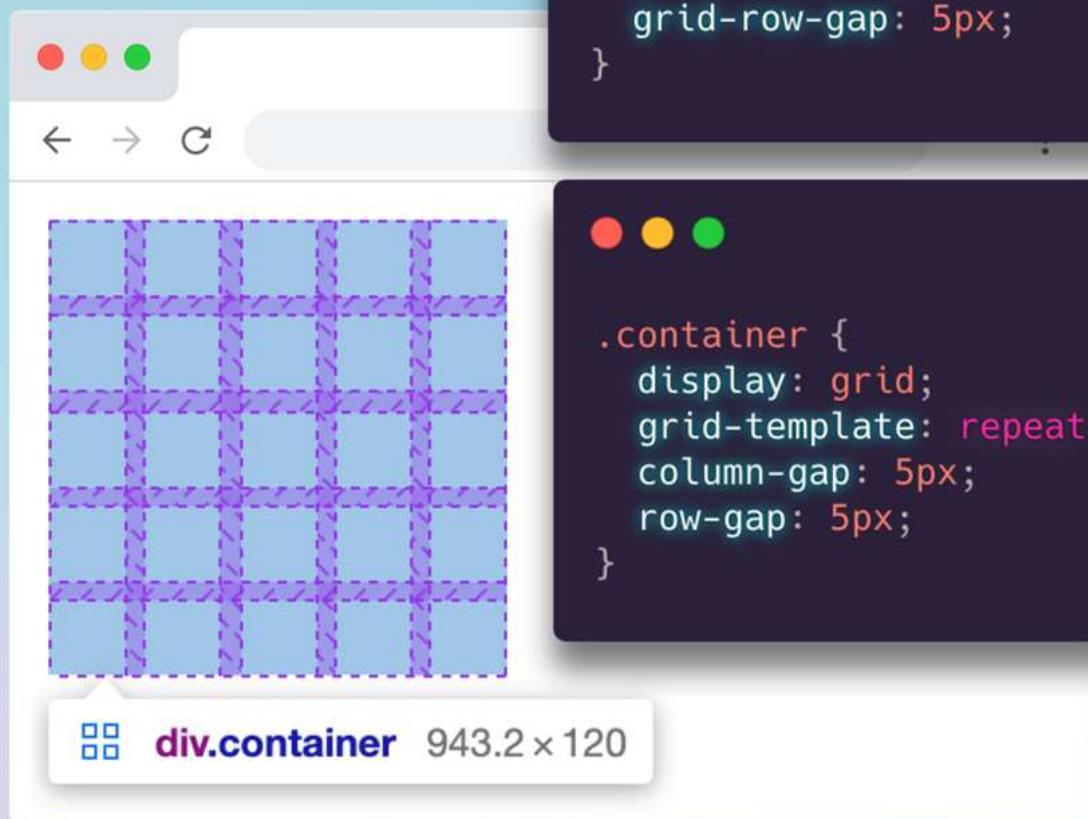
**filas    columnas**

**¿Y si quiero que  
tengan espacios?**



OLD

```
.container {  
  display: grid;  
  grid-template: repeat(5, 20px) / repeat(5, 20px);  
  grid-column-gap: 5px;  
  grid-row-gap: 5px;  
}
```



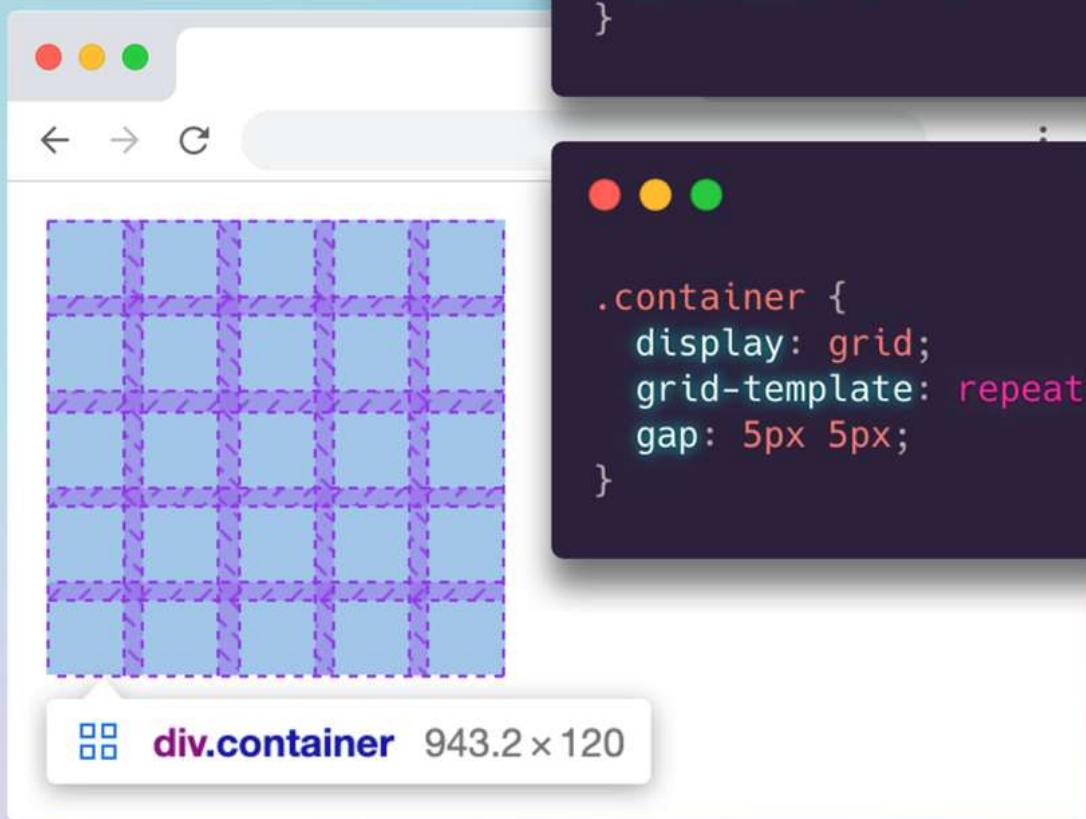
STANDARD

```
.container {  
  display: grid;  
  grid-template: repeat(5, 20px) / repeat(5, 20px);  
  column-gap: 5px;  
  row-gap: 5px;  
}
```

Forma larga

OLD

```
.container {  
  display: grid;  
  grid-template: repeat(5, 20px) / repeat(5, 20px);  
  grid-gap: 5px 5px;  
}
```



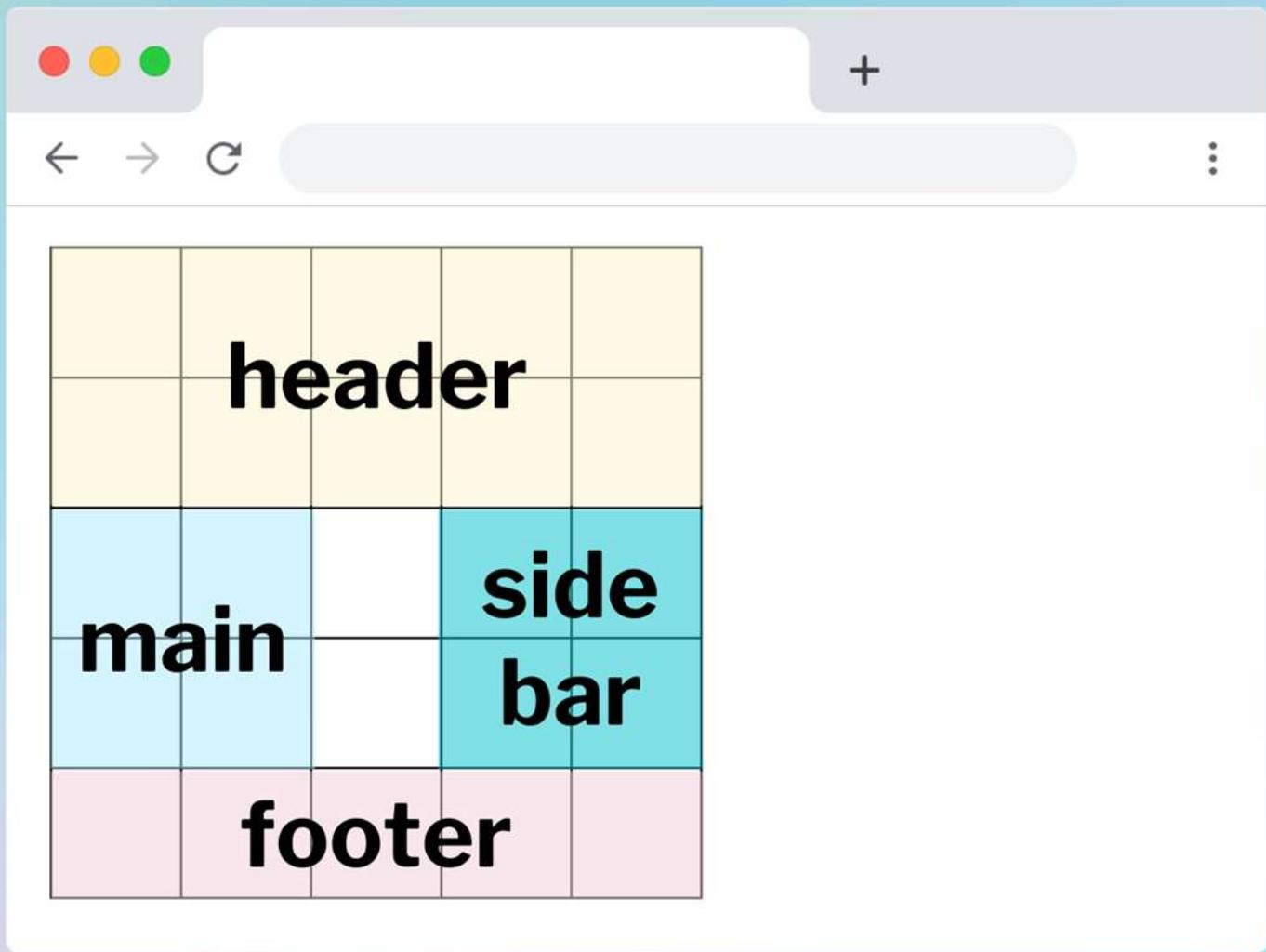
STANDARD

```
.container {  
  display: grid;  
  grid-template: repeat(5, 20px) / repeat(5, 20px);  
  gap: 5px 5px;  
}
```

Forma corta

# ÁREAS

```
.container {  
    display: grid;  
    grid-template: repeat(5, 20px) / repeat(5, 20px);  
    grid-template-areas:  
        "header header header header header"  
        "header header header header header"  
        "main   main   .     sidebar sidebar"  
        "main   main   .     sidebar sidebar"  
        "footer footer footer footer footer";  
}
```





← → ⌘



header	header	header	header	header
header	header	header	header	header
main	main	vacío	sidebar	sidebar
main	main	vacío	sidebar	sidebar
footer	footer	footer	footer	footer

```
.container {  
    display: grid;  
    grid-template: repeat(5, 20px) / repeat(5,  
    grid-template-areas:  
        "header header header header header"  
        "header header header header header"  
        "main   main   .     sidebar sidebar"  
        "main   main   .     sidebar sidebar"  
        "footer footer footer footer footer";  
}
```



**¡Hora del reto!**

# Pasos:

1

Ir al diseño  
de nuestro  
proyecto

3

A cada contenedor,  
identificarle las  
diferentes filas y  
columnas

2

Identificar todos  
los posibles  
contenedores



# ALINEAMIENTO + QUÍZ

# **ALINEAMIENTO**

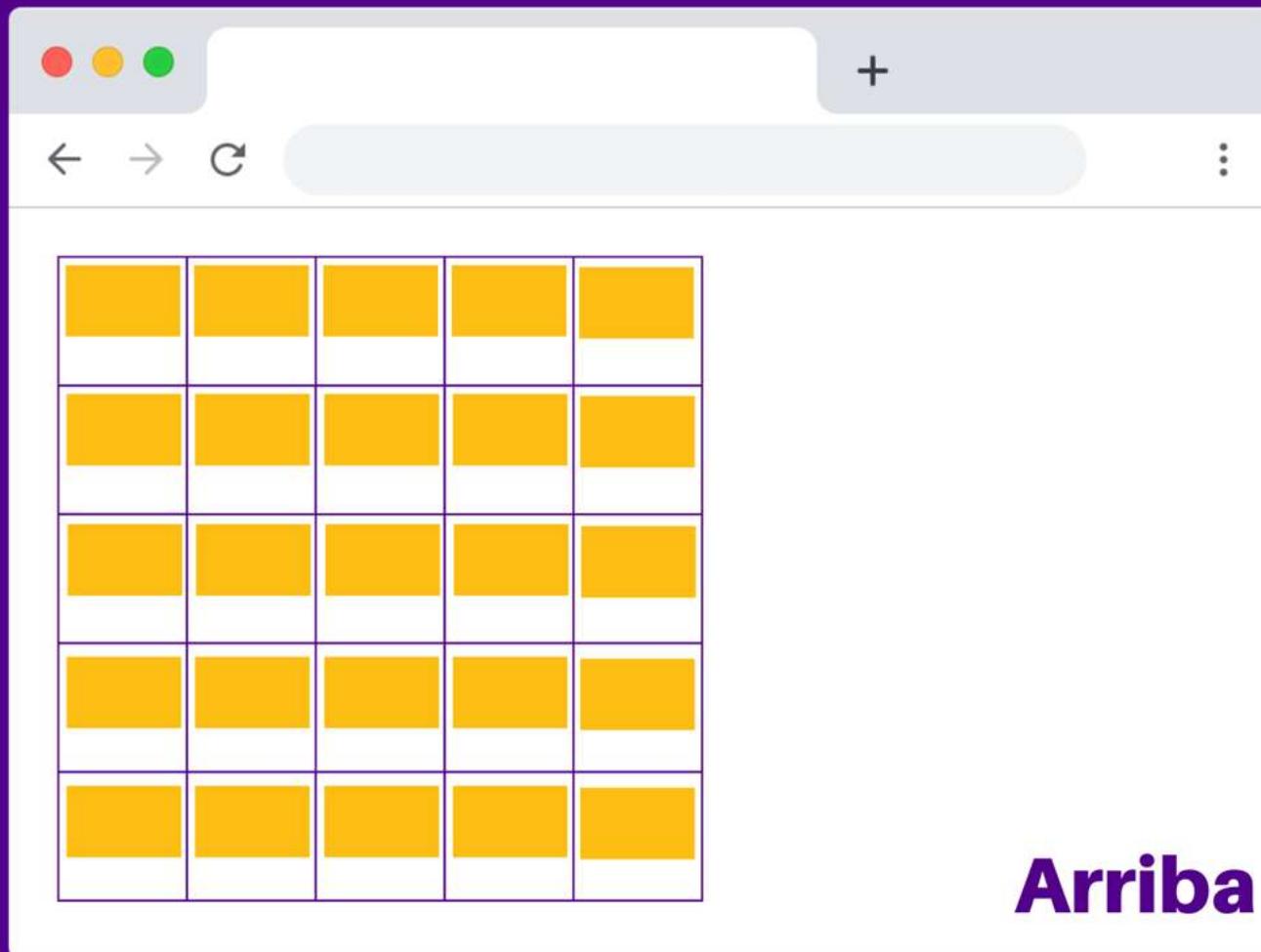
**En dónde quiero que  
estén los elementos  
de mi grid**



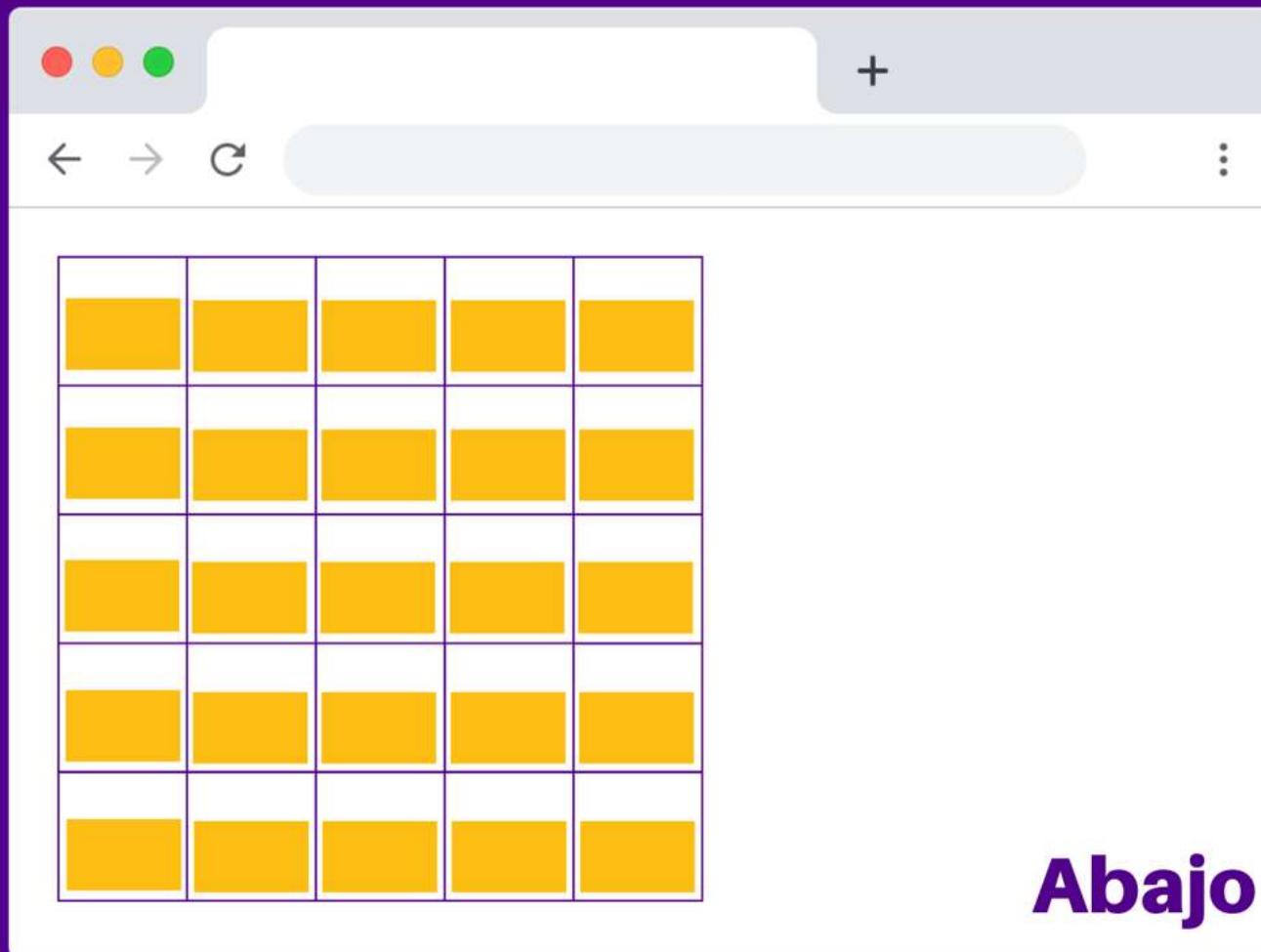
A la derecha



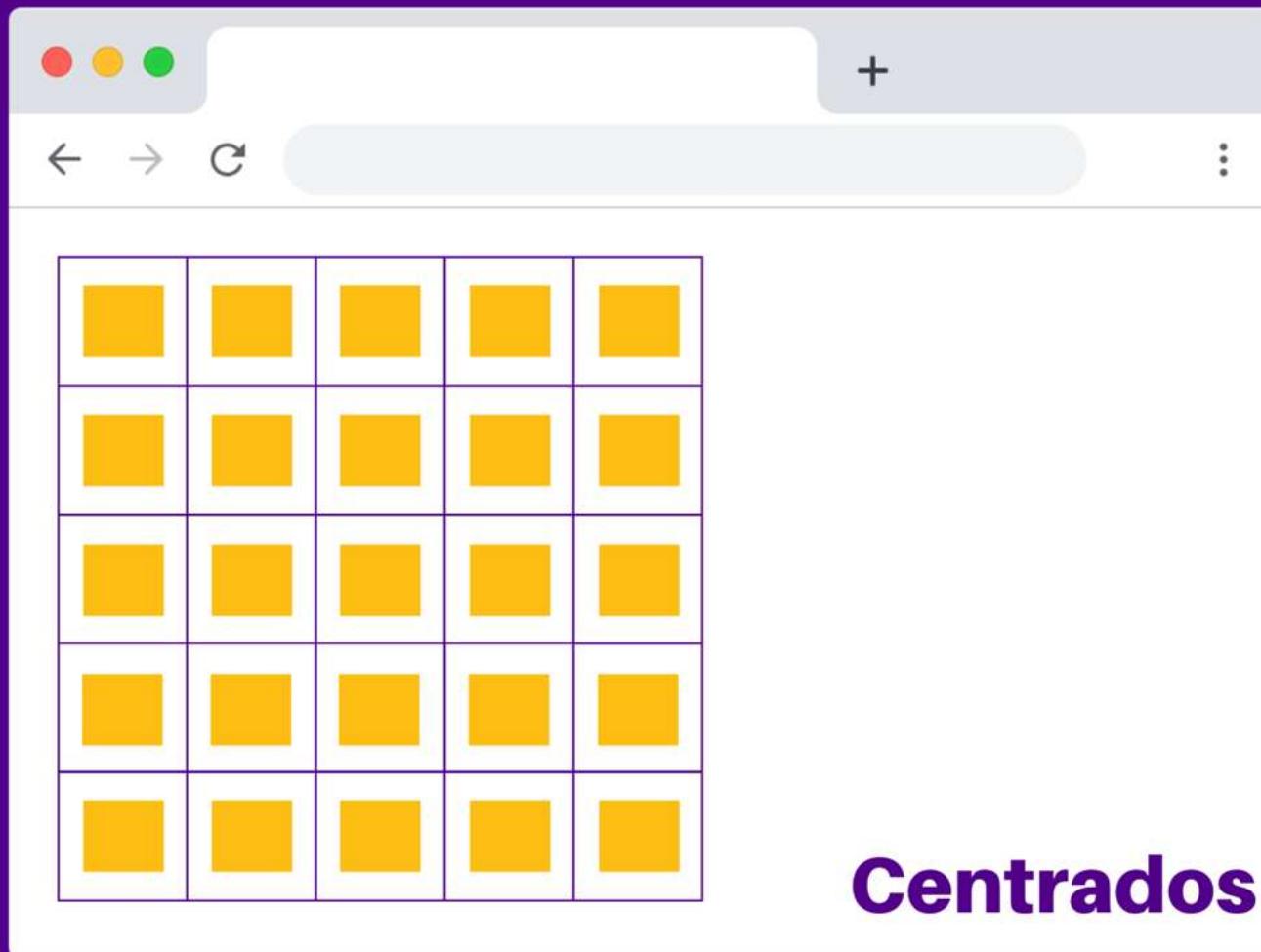
A la izquierda



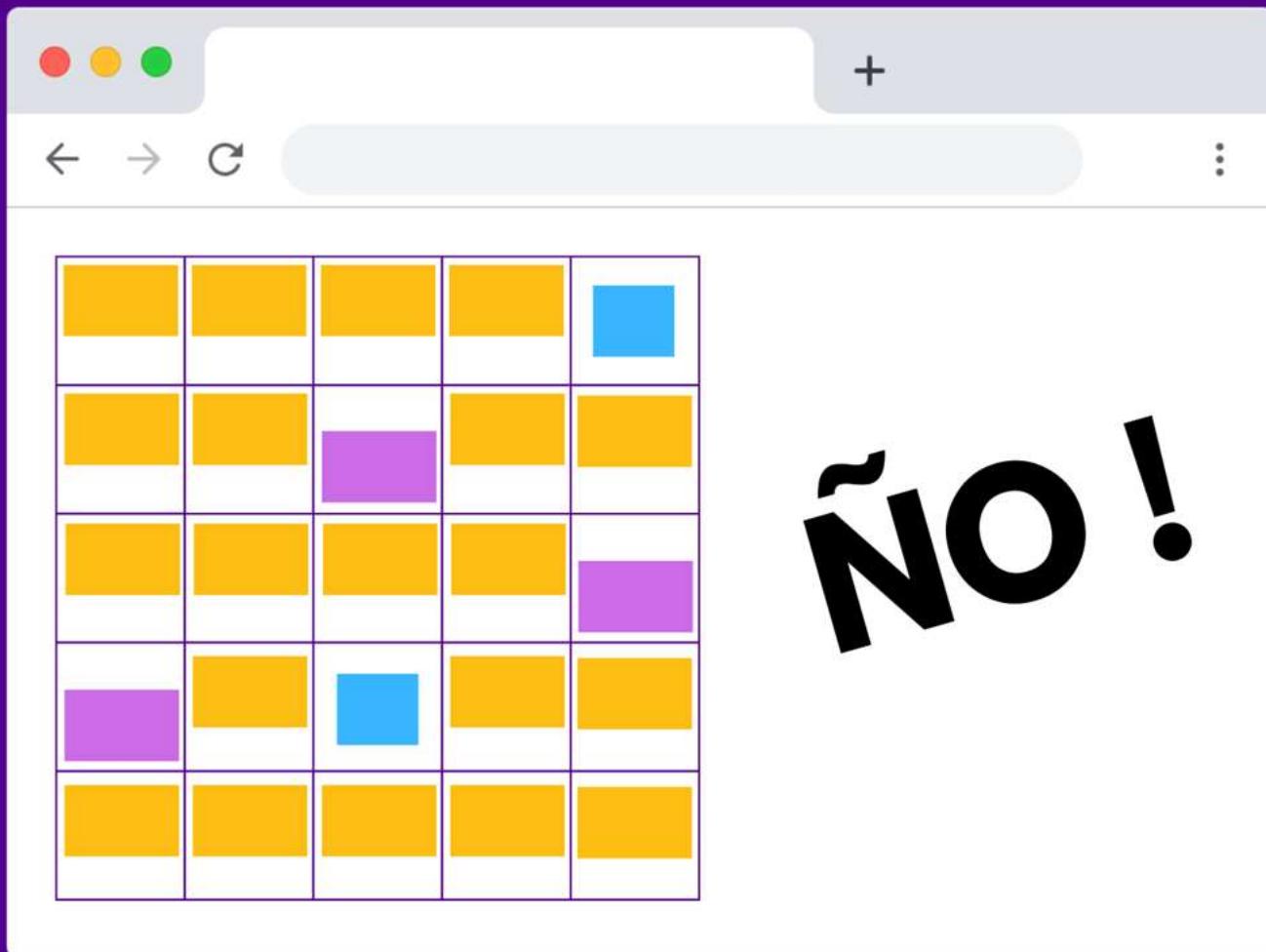
**Arriba**



**Abajo**



**Centrados**





**justify-items**  
**align-items**  
**place-items**



**justify-content**  
**align-content**  
**place-content**



**justify-items**  
**align-items**  
**place-items**



**justify-content**  
**align-content**  
**place-content**

**justify**



**inline axis  
(row axis)**



**block axis  
(column axis)**

**align**

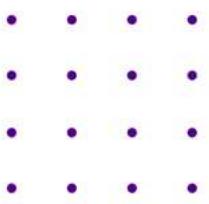
# justify

**start**

**end**



**inline axis  
(row axis)**



# **align**

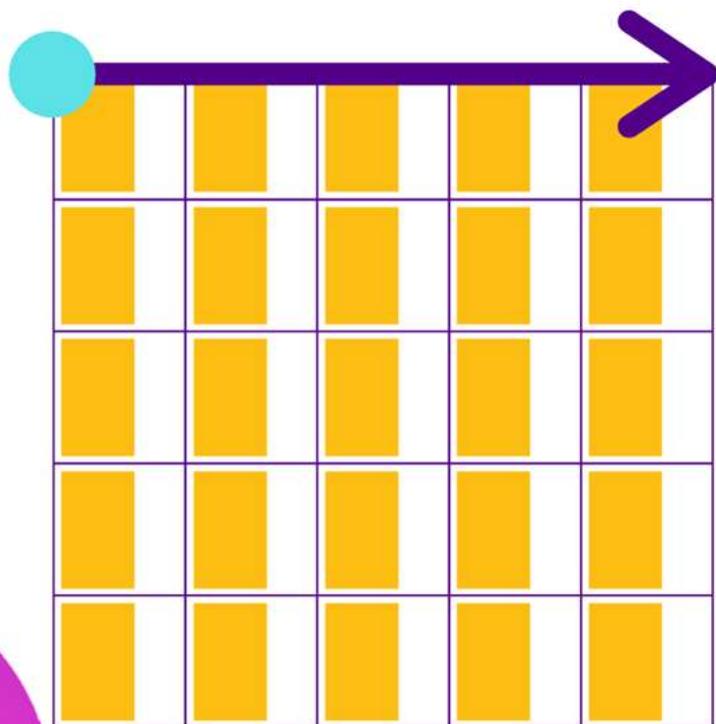
**start**



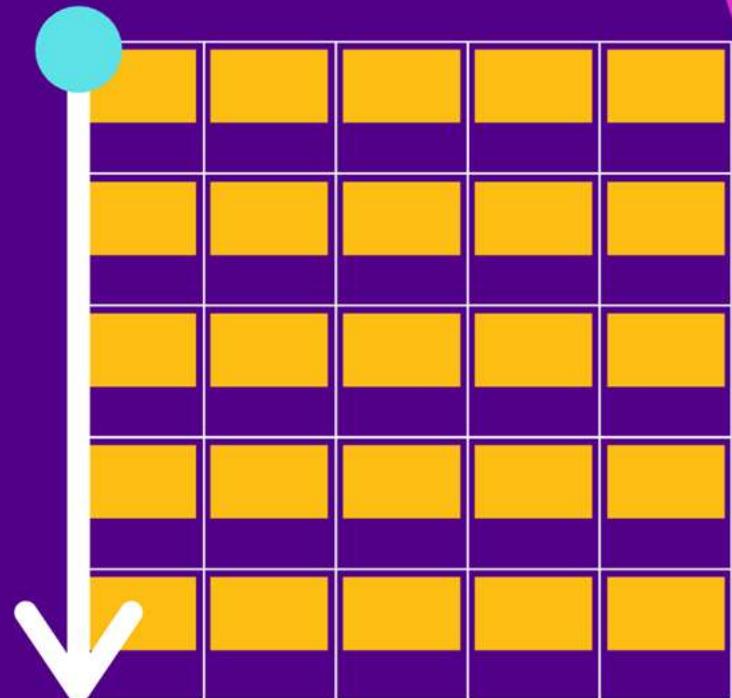
**block axis  
(column axis)**

**end**

# justify



START



# align



**justify-items**  
**align-items**  
**place-items**



**justify-content**  
**align-content**  
**place-content**



**justify-items**  
**align-items**  
**place-items**



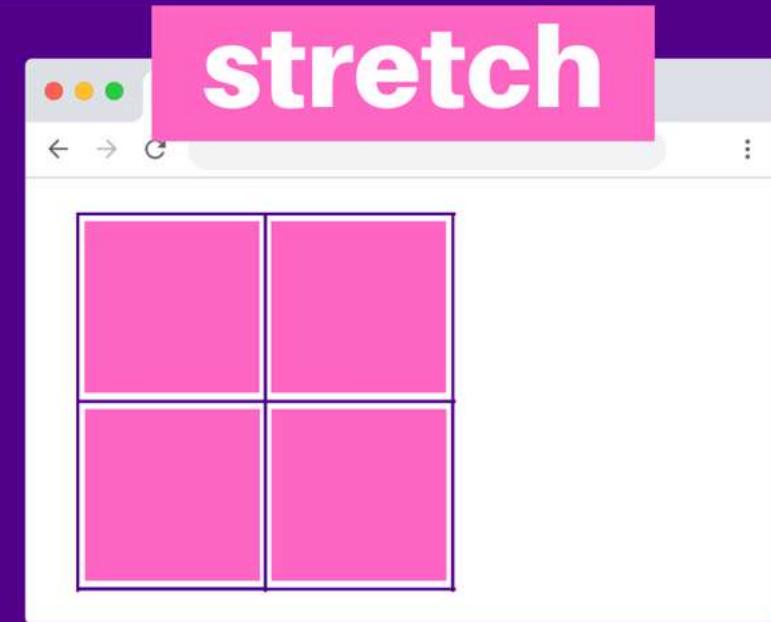
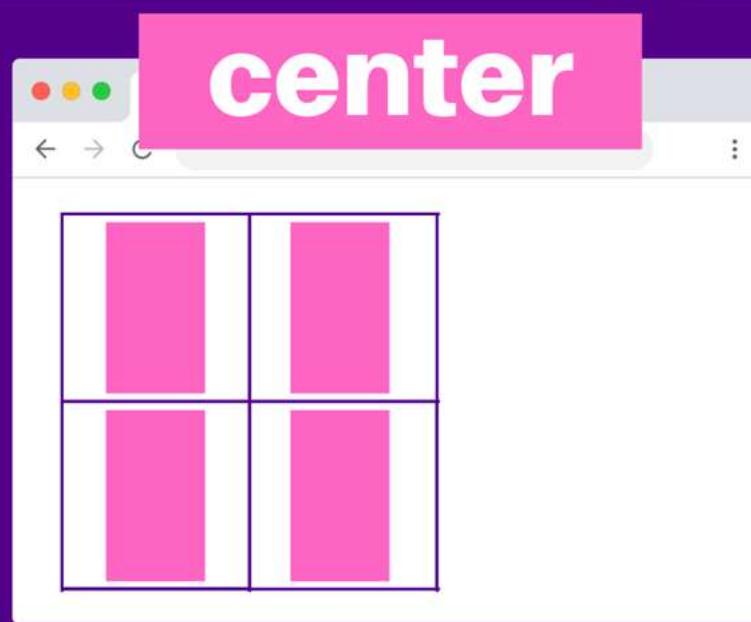
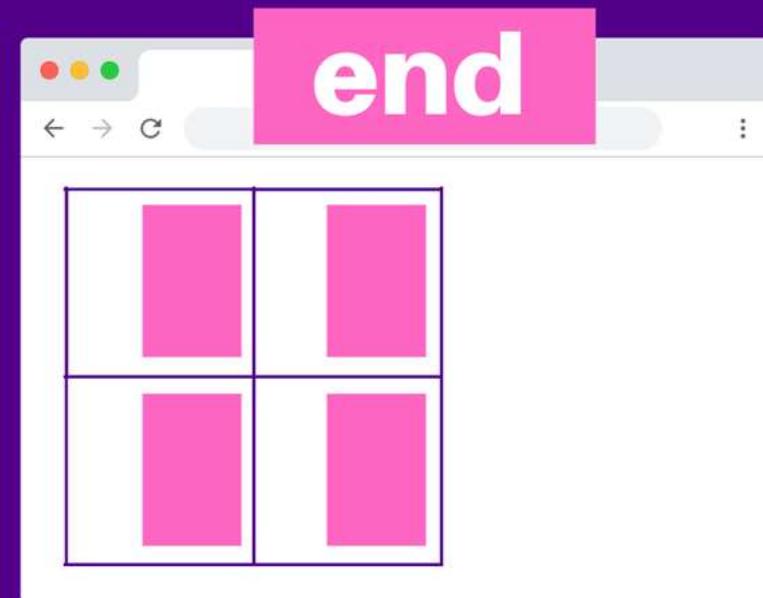
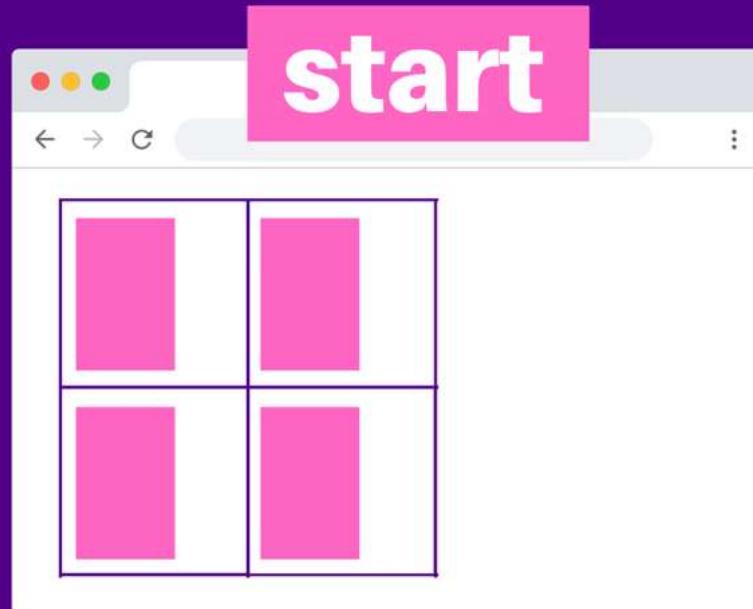
**Alinear lo que  
está por dentro  
de la Grid**

**Veamos los  
diferentes valores  
que pueden tener  
estas propiedades**

# justify-items



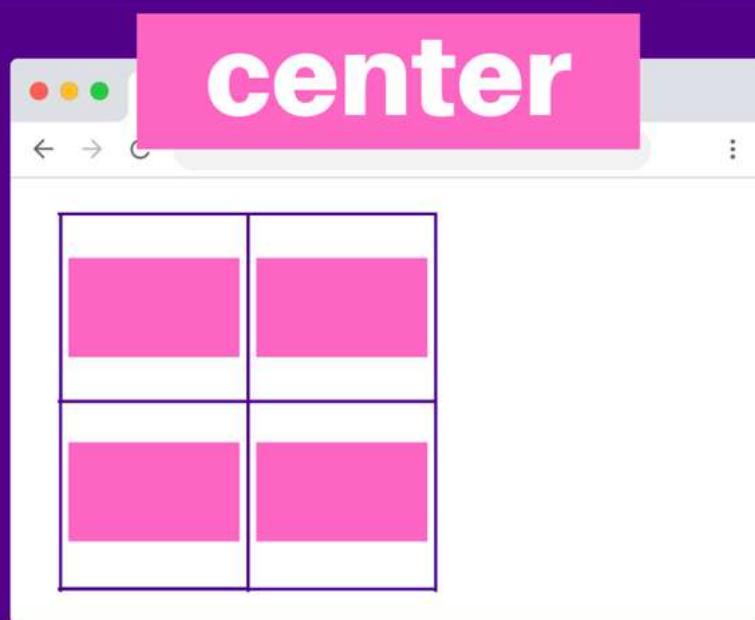
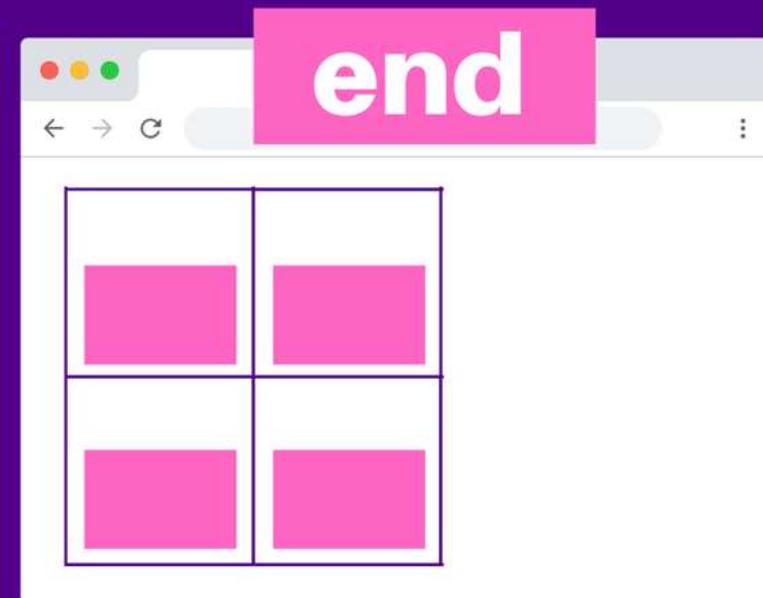
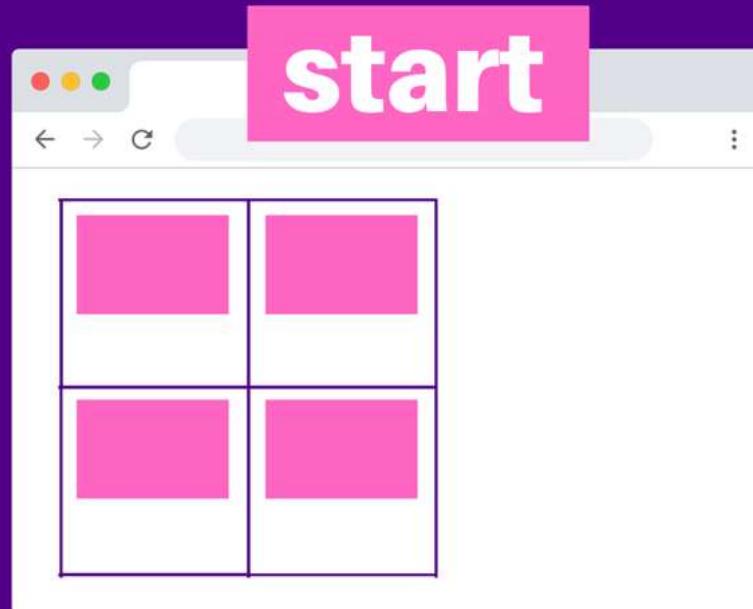
```
.container {  
  justify-items: start | end | center | stretch;  
}
```



# align-items



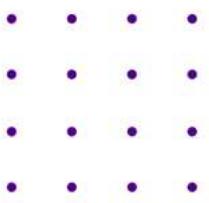
```
.container {  
  align-items: start | end | center | stretch;  
}
```



# place-items



```
.container {  
  place-items: <align-items> / <justify-items>;  
}
```



**Alinear la  
Grid en su  
contenedor**



**justify-content  
align-content  
place-content**

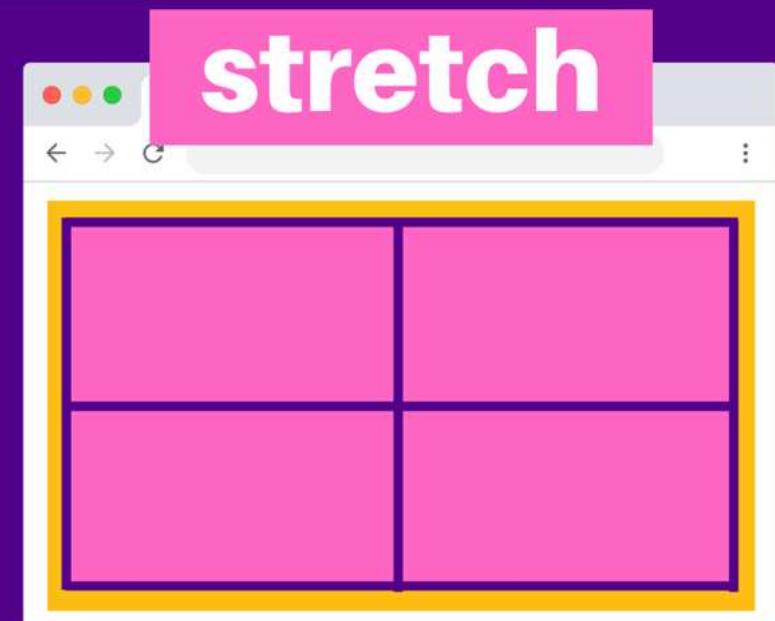
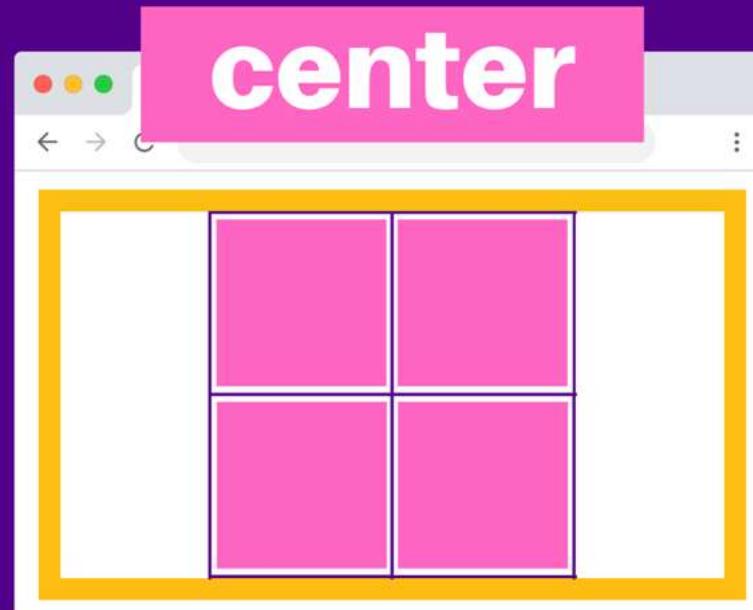
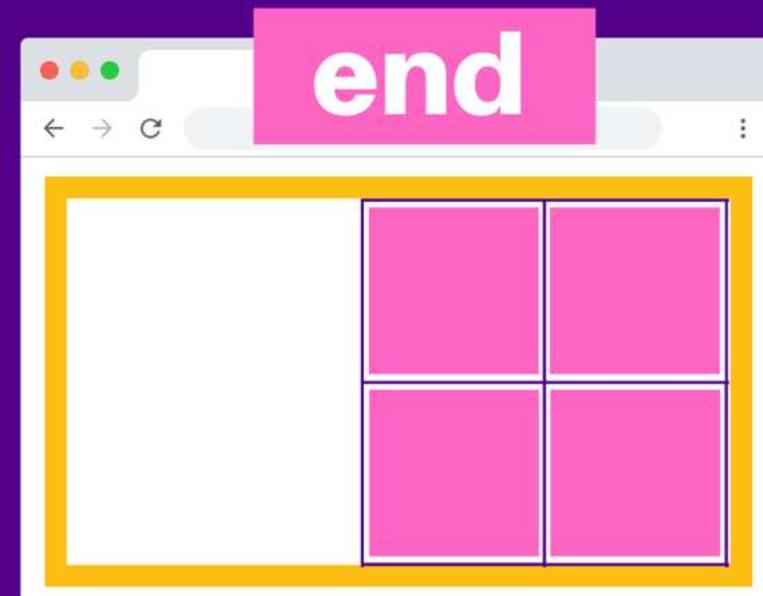
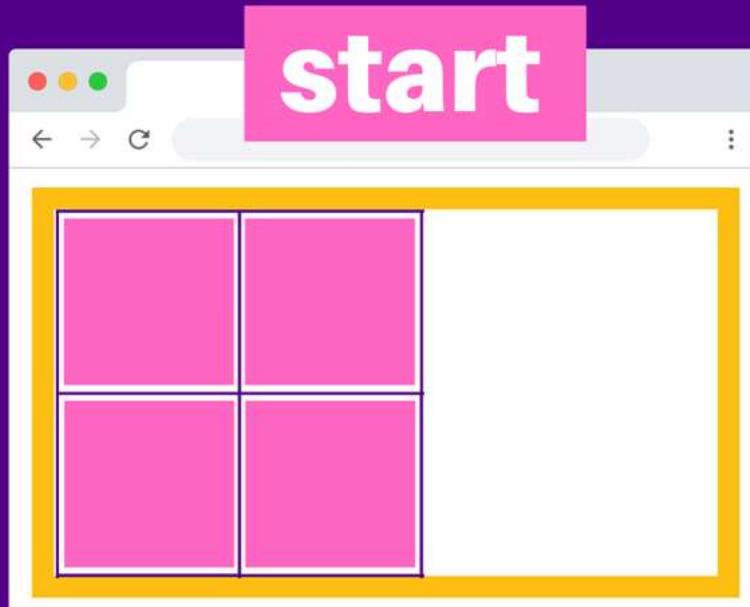


**Veamos los  
diferentes valores  
que pueden tener  
estas propiedades**

# justify-content (Parte 1)



```
.container {  
  justify-content: start | end | center | stretch;  
}
```



# justify-content (Parte 2)



```
.container {  
  justify-content: space-around | space-between | space-evenly;  
}
```

# Tarea # 1

Hacer la demostración para:

**justify-content: space-around;**

**justify-content: space-between;**

**justify-content: space-evenly;**

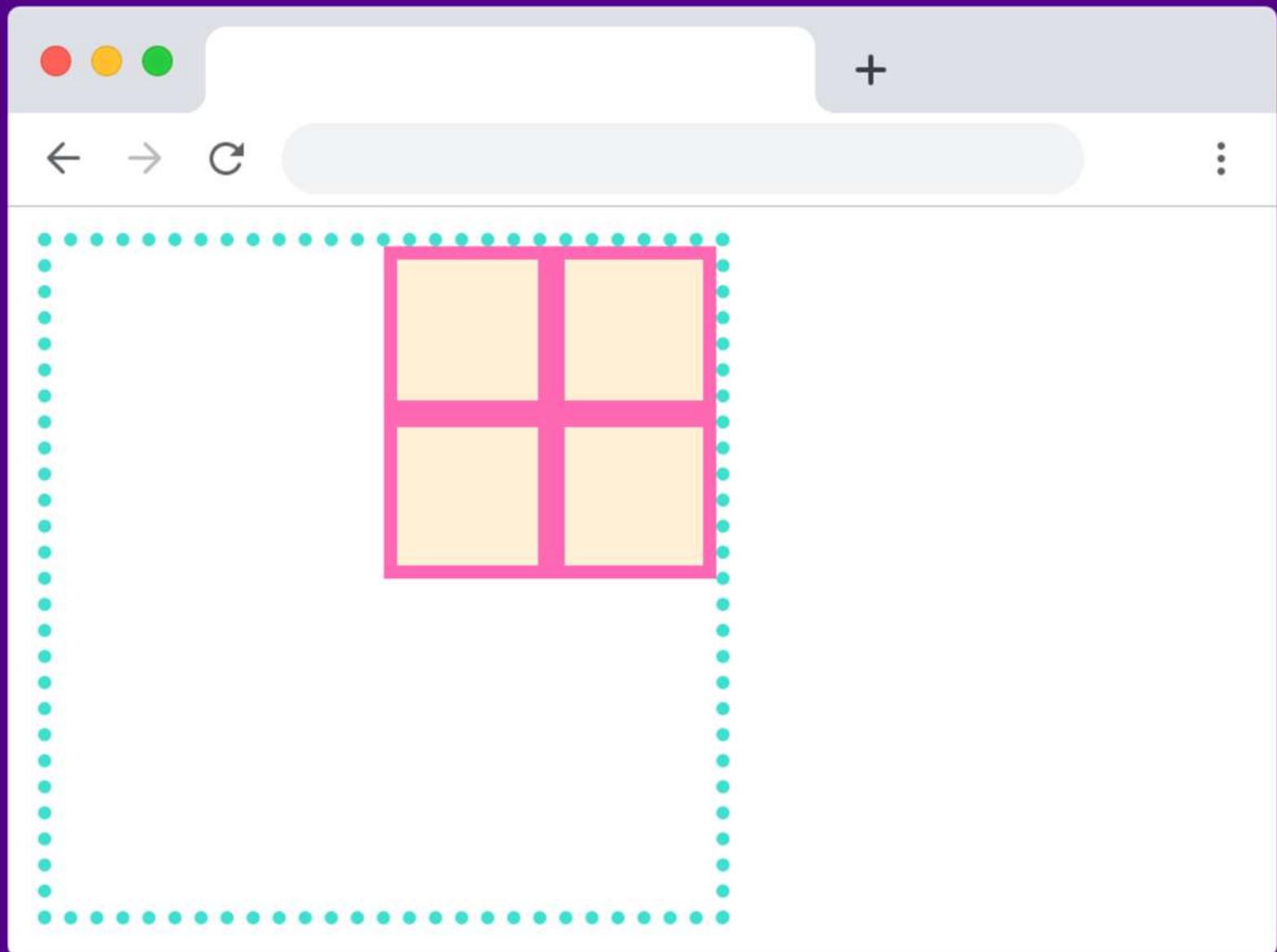


# HTML

```
<div class="grid-container">
  <div class="container">
    <div class="item"></div>
    <div class="item"></div>
    <div class="item"></div>
    <div class="item"></div>
  </div>
</div>
```

# CSS

```
.grid-container {  
    border: 4px dotted turquoise;  
    display: grid;  
    justify-content: end;  
    width: 200px;  
    height: 200px;  
}  
  
.container {  
    background: papayawhip;  
    display: grid;  
    grid-template-columns: repeat(2, 1fr);  
    grid-template-rows: auto;  
    width: 100px;  
    height: 100px;  
}  
  
.item {  
    background: papayawhip;  
    border: 4px solid hotpink;  
}
```

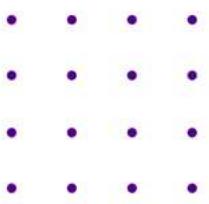


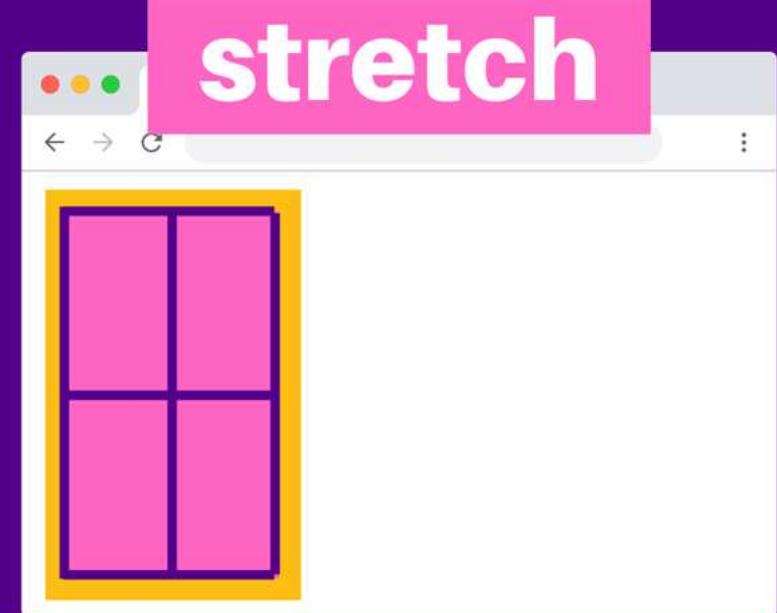
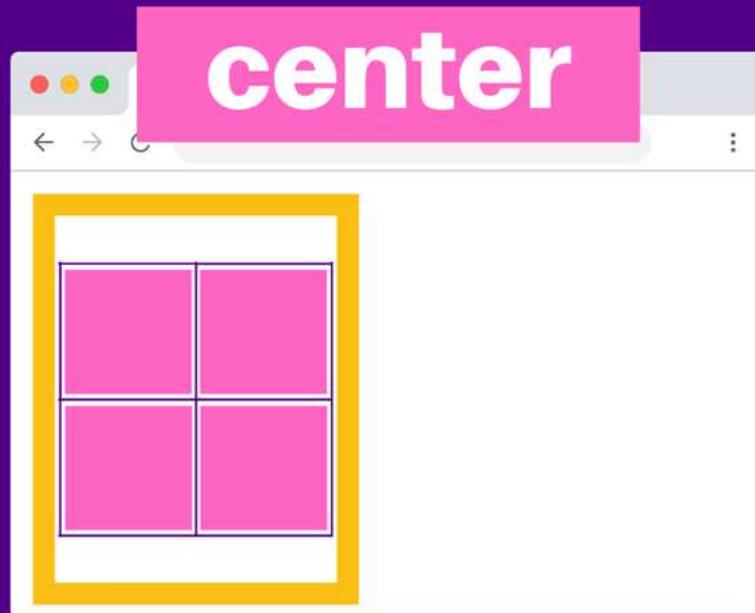
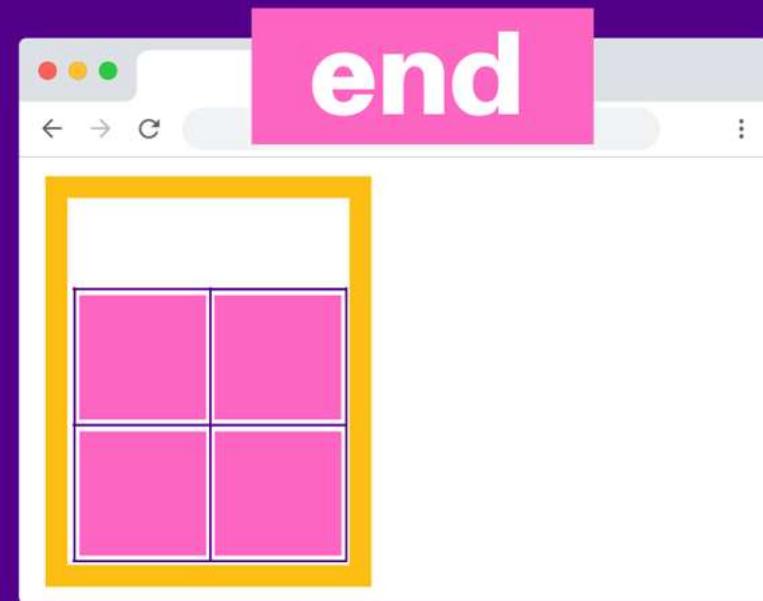
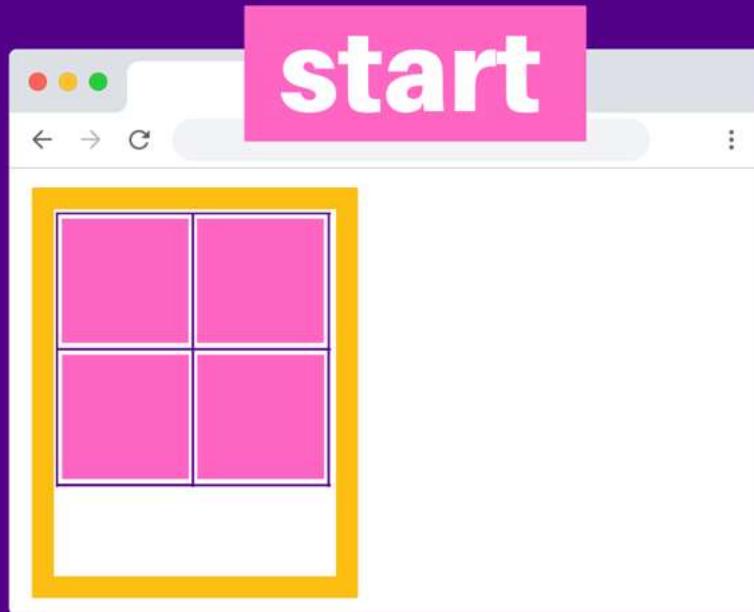
# align-content

## (Parte 1)



```
.container {  
  align-content: start | end | center | stretch;  
}
```



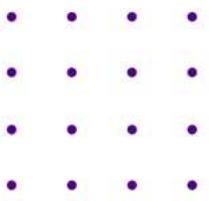


# align-content

## (Parte 2)



```
.container {  
    align-content: space-around | space-between | space-evenly;  
}
```



# Tarea # 2

Hacer la demostración para:

**align-content: space-around;**  
**align-content: space-between;**  
**align-content: space-evenly;**

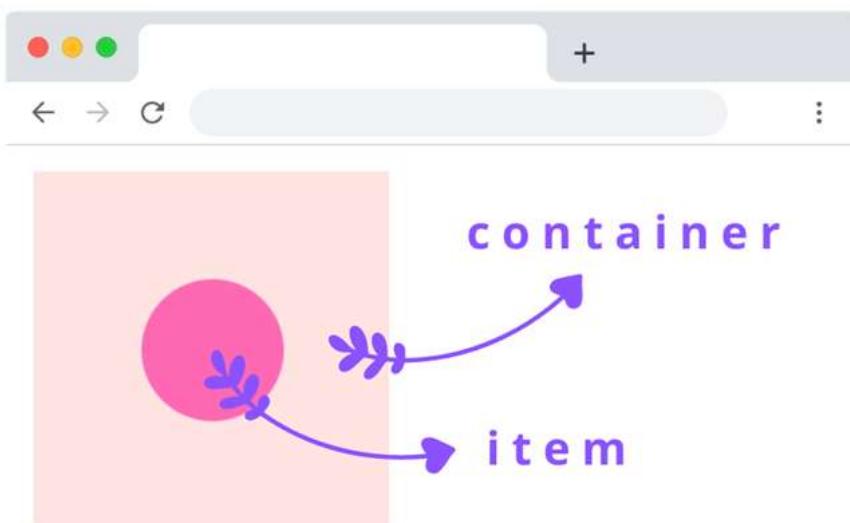
# place-content



```
.container {  
  place-content: <align-items> / <justify-items>;  
}
```

CSS  
**Quiz!**

By @teffcode



A

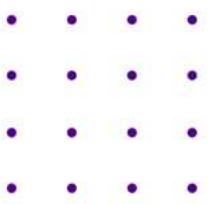
```
display: flex;  
align-items: center;
```

B

```
display: flex;  
justify-content: center;
```

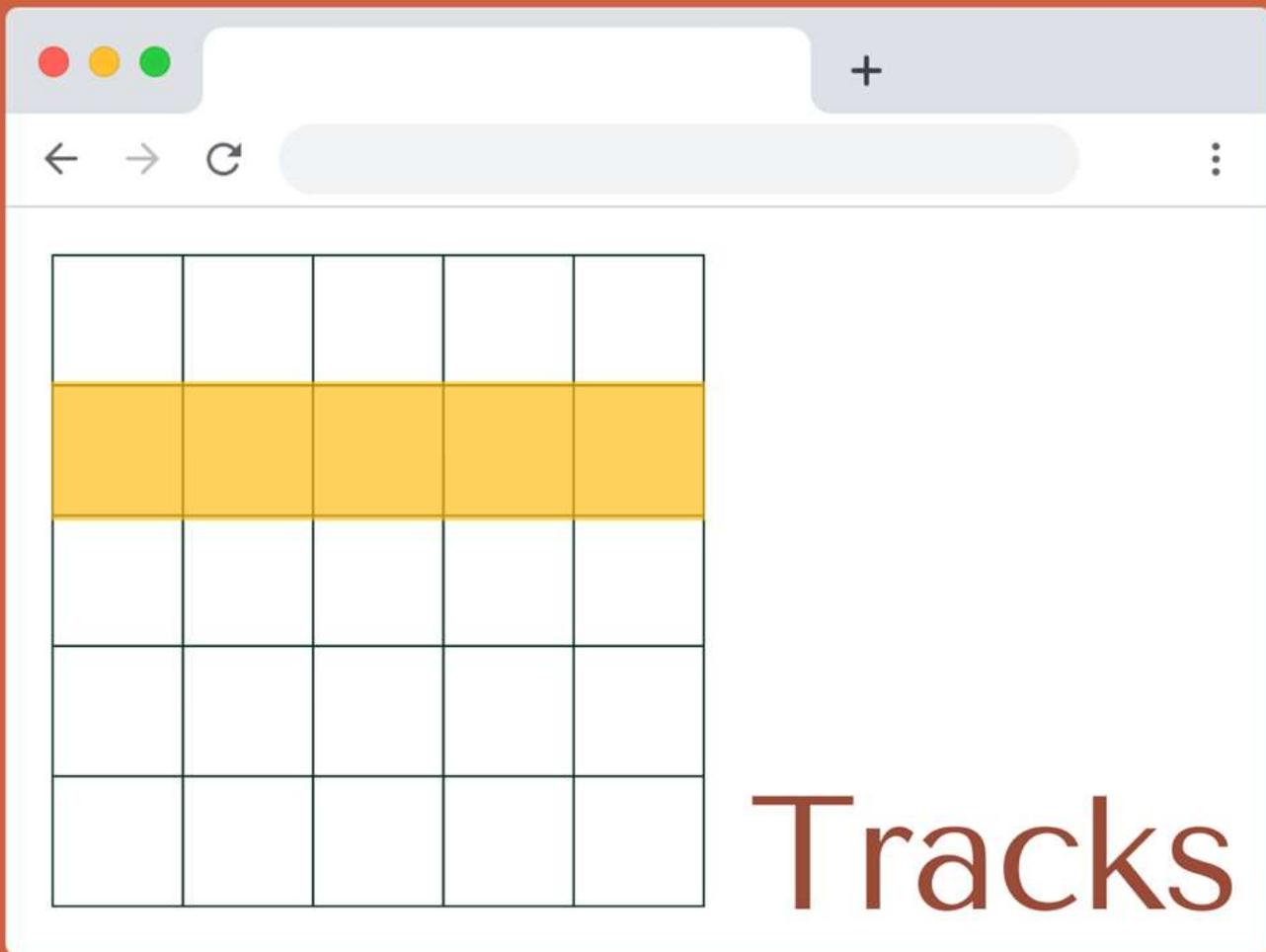
C

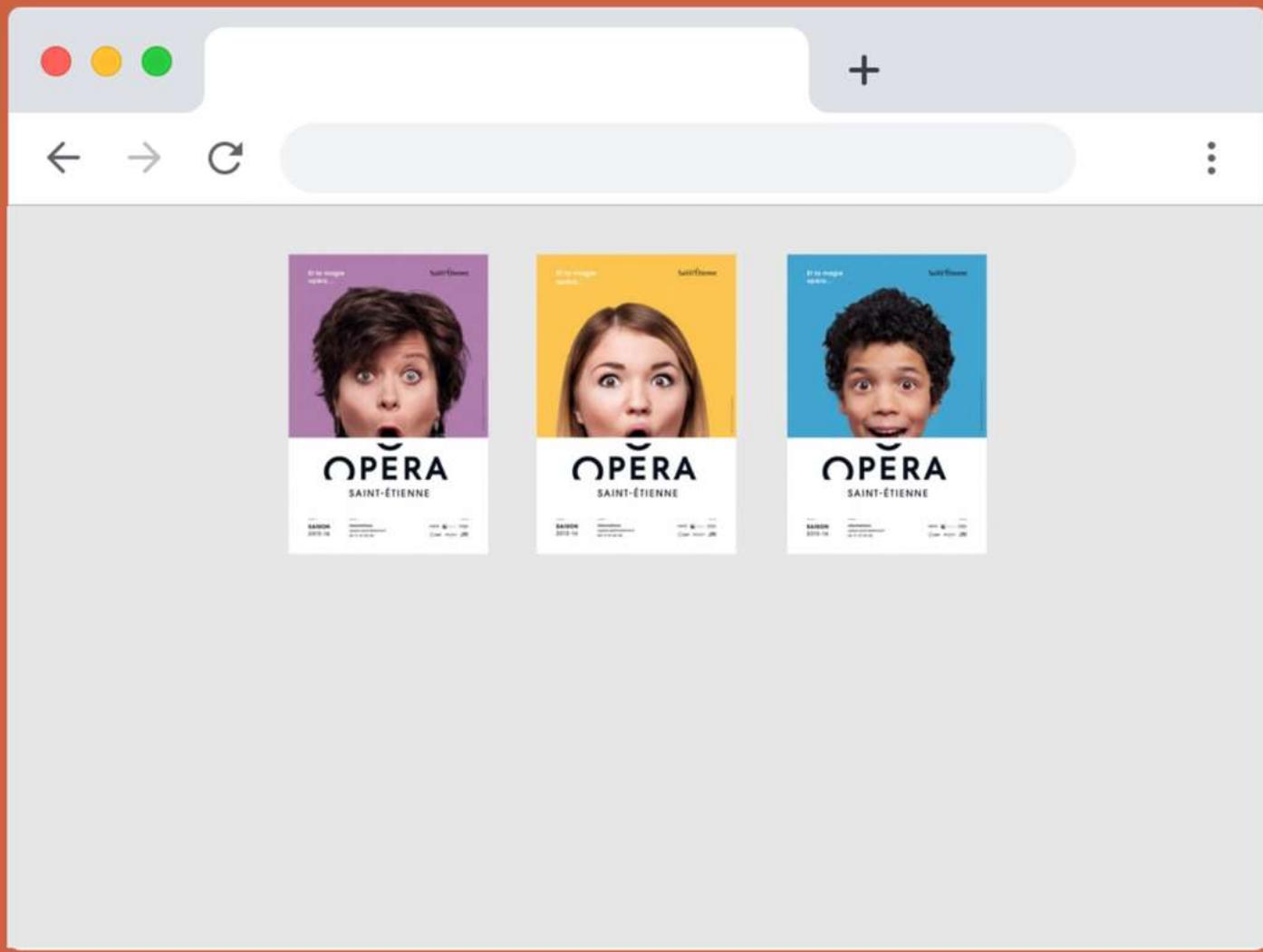
```
display: grid;  
place-items: center;
```

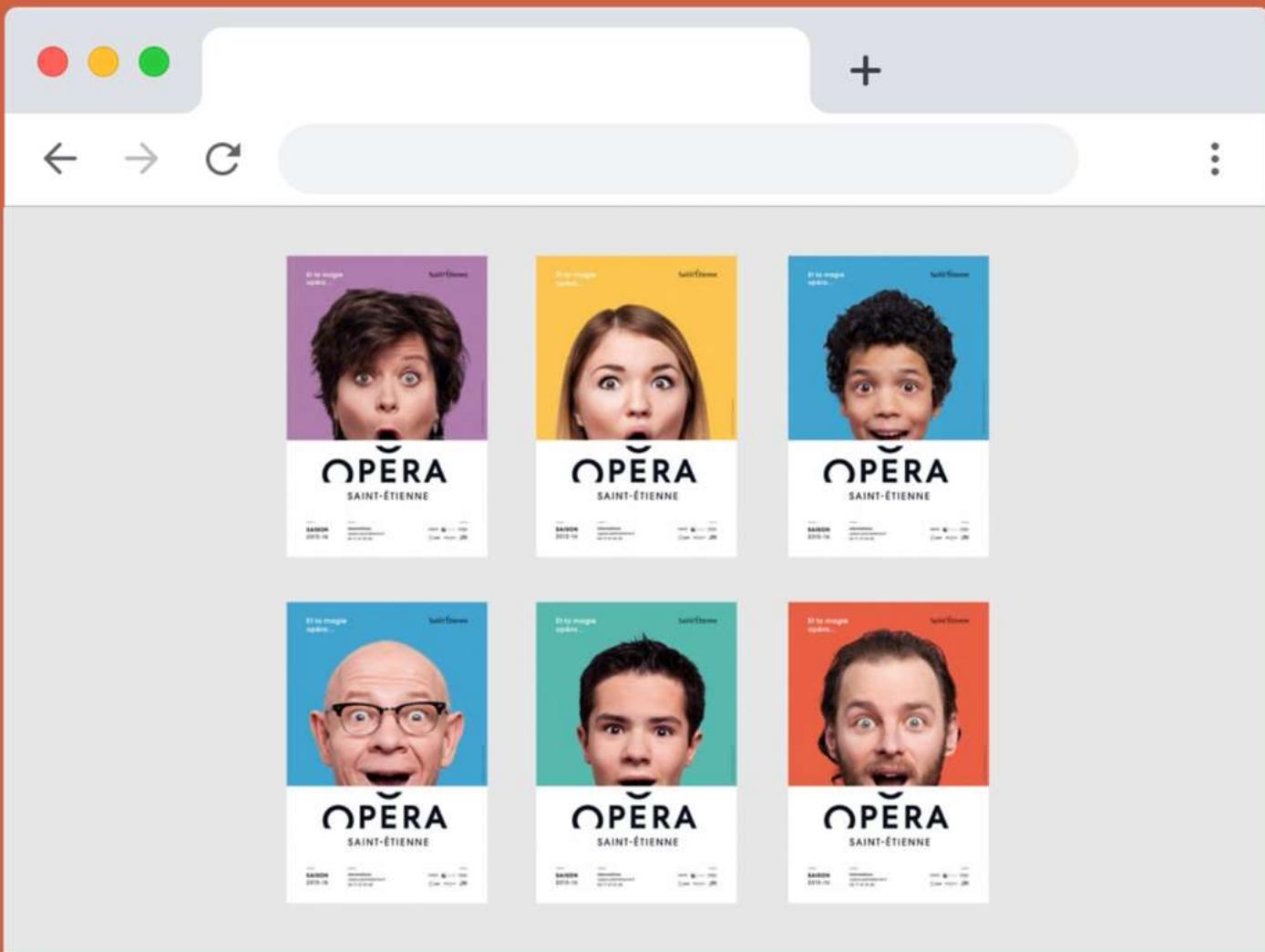


# Generación automática de tracks

+ QUÍZ









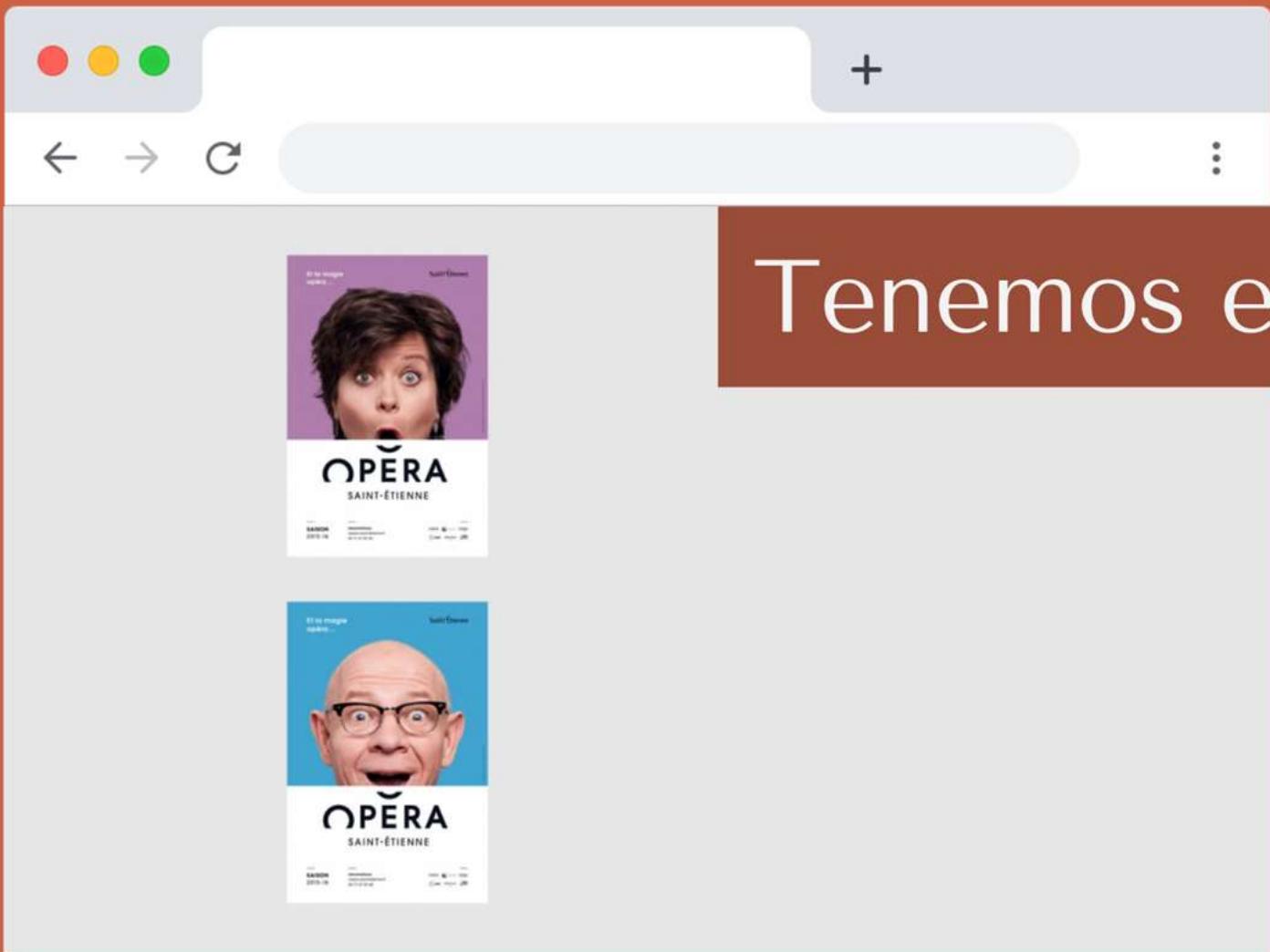
No establecemos  
la cantidad de  
filas desde el  
principio



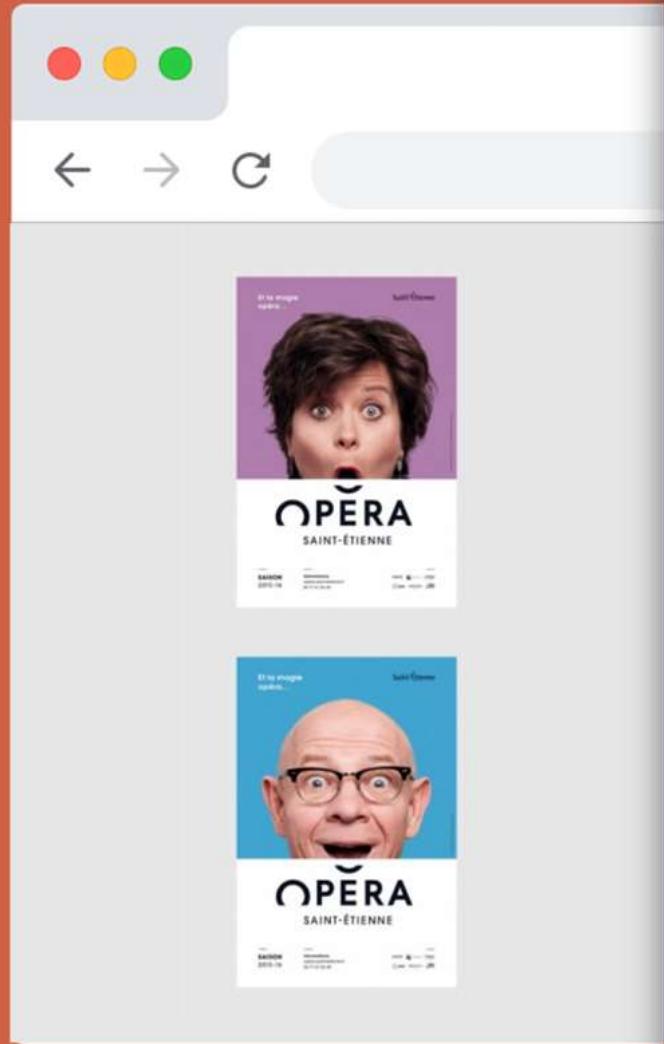
Dejamos que se  
creen solitas

# Grid Implícita

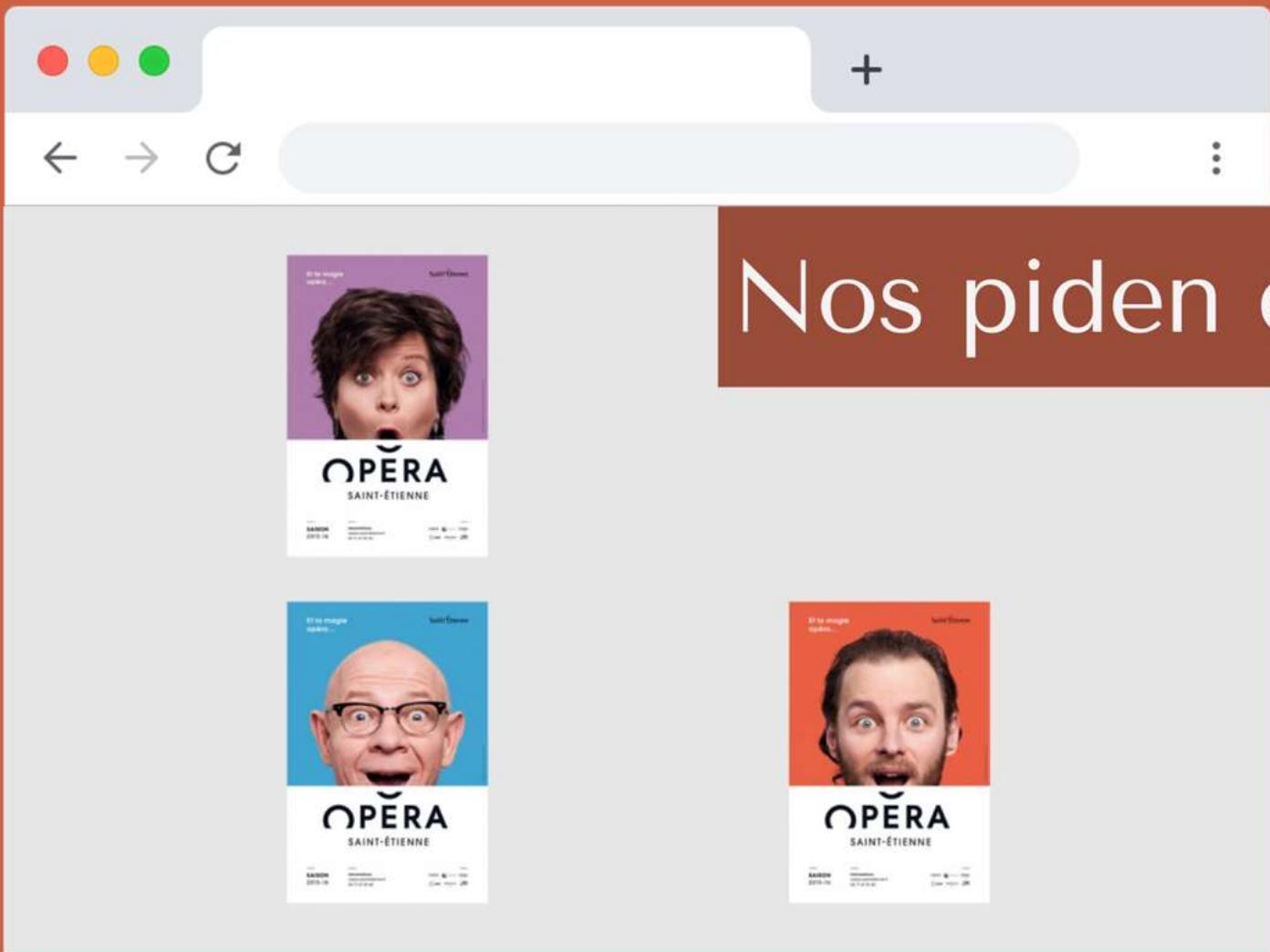
Te crea filas o columnas  
si las necesitas con  
anchos sin tamaño



Tenemos esto



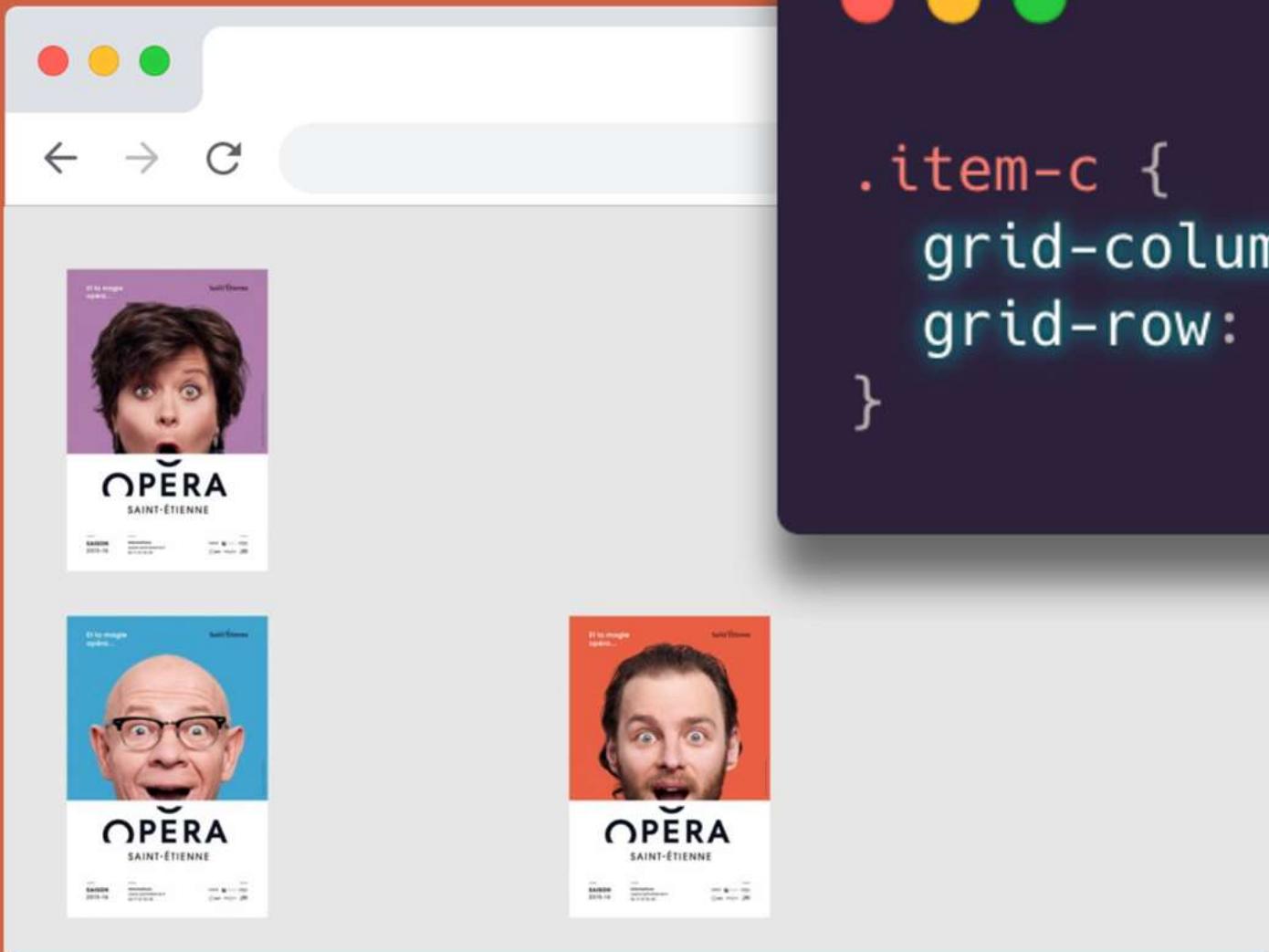
```
.container {  
    grid-template-columns: 60px;  
    grid-template-rows: 90px 90px;  
}  
  
.item-a {  
    grid-column: 1 / 2;  
    grid-row: 1 / 2;  
}  
  
.item-b {  
    grid-column: 1 / 2;  
    grid-row: 2 / 3;  
}
```

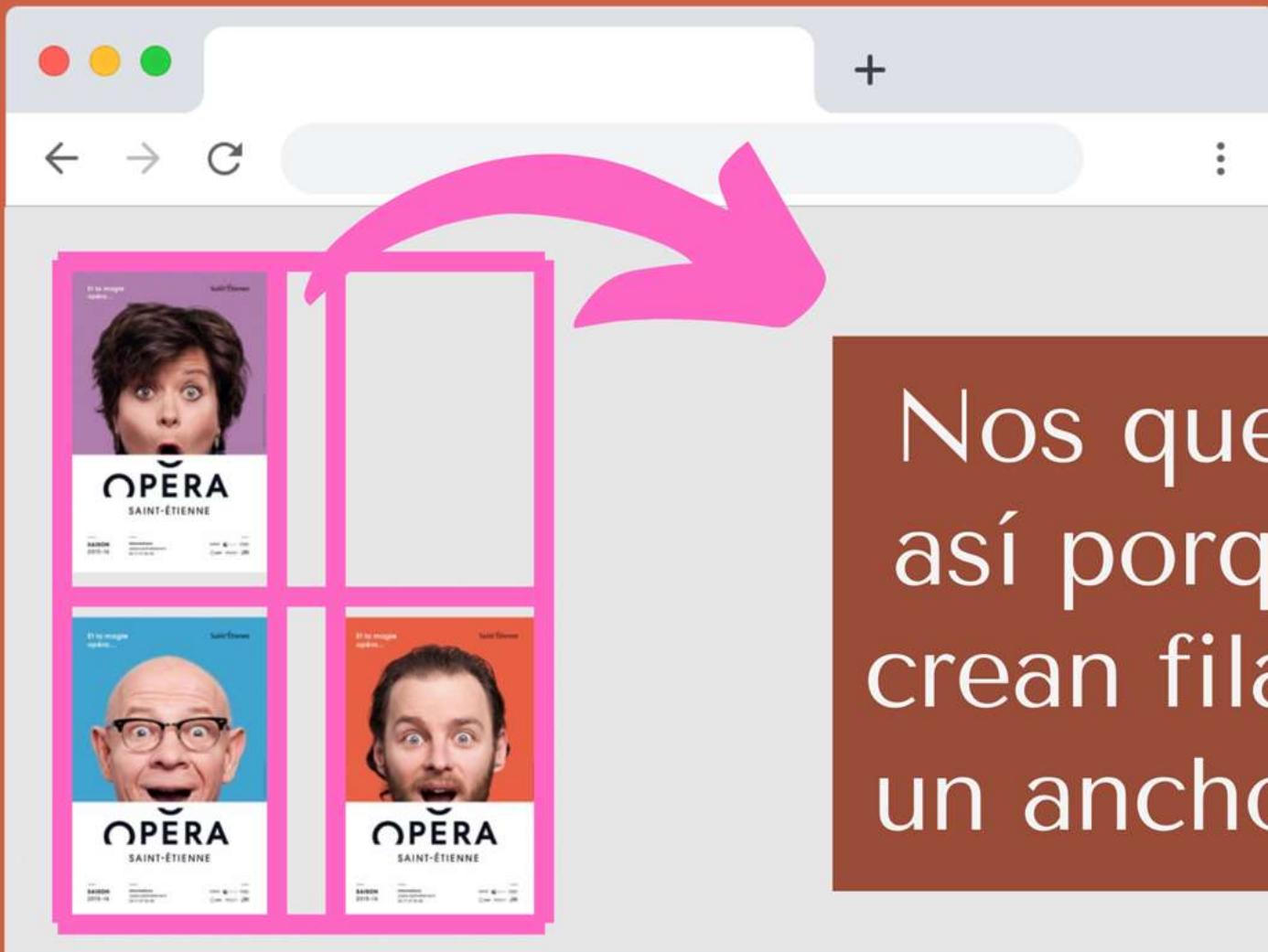


Nos piden esto



```
.item-c {  
    grid-column: 3 / 4;  
    grid-row: 2 / 3;  
}
```





Nos quedaría  
así porque se  
crean filas con  
un ancho de 0



```
.container {  
    grid-auto-columns: 60px;  
}
```

# Nos piden esto



stereo+



jamsolove



Explore

Songs

Artists

Albums

## Blinding Lights

The Weekend

After silence, that which comes nearest to expressing the inexpressible is music.

... e e e



Intention  
Justin Bieber



01

Classical  
music  
collections

196 Songs

02

Pop  
music  
collections

82 Songs

Download



Intention  
Justin Bieber



What A Man Gotta Do  
Jonas Brothers



You Should Be Sad  
Halsey



To Die For  
Sam Smith



Boss Bitch  
Doja Cat



Dancing with a Stranger  
Sam Smith



Underdog  
Alicia Keys



# Nos piden esto

← → ⌂

stereo

Explore

Songs

Artists

Albums

## Blinding Lights

The Weekend

After silence, that which comes nearest to expressing the inexpressible is music.



Intentions  
Justin Bieber



01

Classical  
music  
collections

196 Songs

02

Pop  
music  
collections

82 Songs



jamsolove



### Download

Intentions  
Justin Bieber



What A Man Gotta Do  
Jonas Brothers



You should be Sad  
Halsey



To Die For Sam Smith  
Eminem



Boss Bitch  
Doja Cat



Dancing with a Stranger  
Sam Smith



Underdog  
Alicia Keys





```
.container {  
  grid-auto-flow: row | column | row dense | column dense;  
}
```

dense solo cambia el  
orden visual de sus  
elementos



# ¿Con qué propiedad(es) de CSS Grid podemos darle tamaño a tracks implícitos ?

A



grid-gap

B



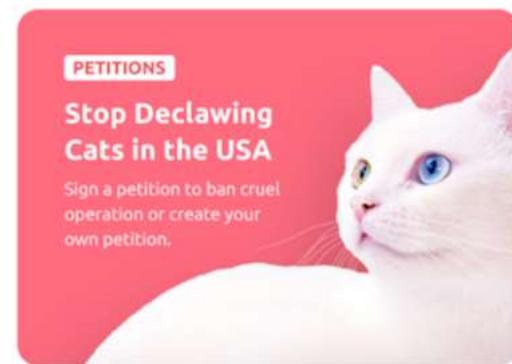
grid-auto-columns

C

grid-auto-rows

# Funciones: `repeat()`, `minmax()` y `fit-content()`

+ Quíz



The screenshot shows a web browser window with a grid layout example. The grid consists of three columns, each with a red border. The first column contains a card for 'VOLUNTEERING' featuring a person in an orange sweatshirt holding a dog. The second column contains a card for 'CHARITY' featuring a colorful parrot. The third column contains a card for 'PETITIONS' featuring a white cat. Below the grid is a dark box containing CSS code for creating the grid.

```
.container {  
  display: grid;  
  grid-template-columns: 1fr 1fr 1fr;  
}
```

The screenshot illustrates a web page layout using a grid system. At the top, there's a browser header with standard controls like back, forward, and search. Below the header is a main content area featuring three cards arranged horizontally. Each card has a colored background (blue, green, red) and contains text and an image.

**Card 1 (Blue):**

- VOLUNTEERING**
- Join our dedicated team**
- Every day we help animals in 23 countries.

**Card 2 (Green):**

- CHARITY**
- African Wildlife Foundation**
- Or choose from 120 other organizations to support according to your choice.

**Card 3 (Red):**

- PETITIONS**
- Stop Declawing Cats in the USA**
- Sign a petition to ban cruel operation or create your own petition.

Below the cards, a dark overlay shows the CSS code for the container element:

```
.container {  
  display: grid;  
  grid-template-columns: repeat(3, 1fr);  
}
```

CHARITY

## African Wildlife Foundation

Or choose from 120 other organizations to support according to your choice.



**Cada card tiene un tamaño para que se vea bien**

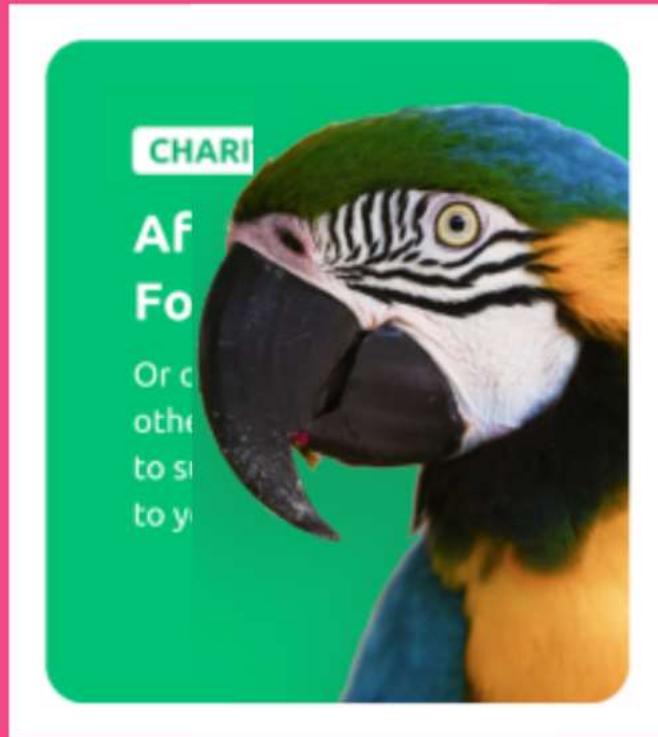
CHARITY

## African Wildlife Foundation

Or choose from 120 other organizations to support according to your choice.



**Si estiramos el navegador,  
empezamos a ver la card así**



**Si contraemos el navegador,  
empezamos a ver la card así**

**VOLUNTEERING**

Join our dedicated team  
Every day we help animals in 23 countries.

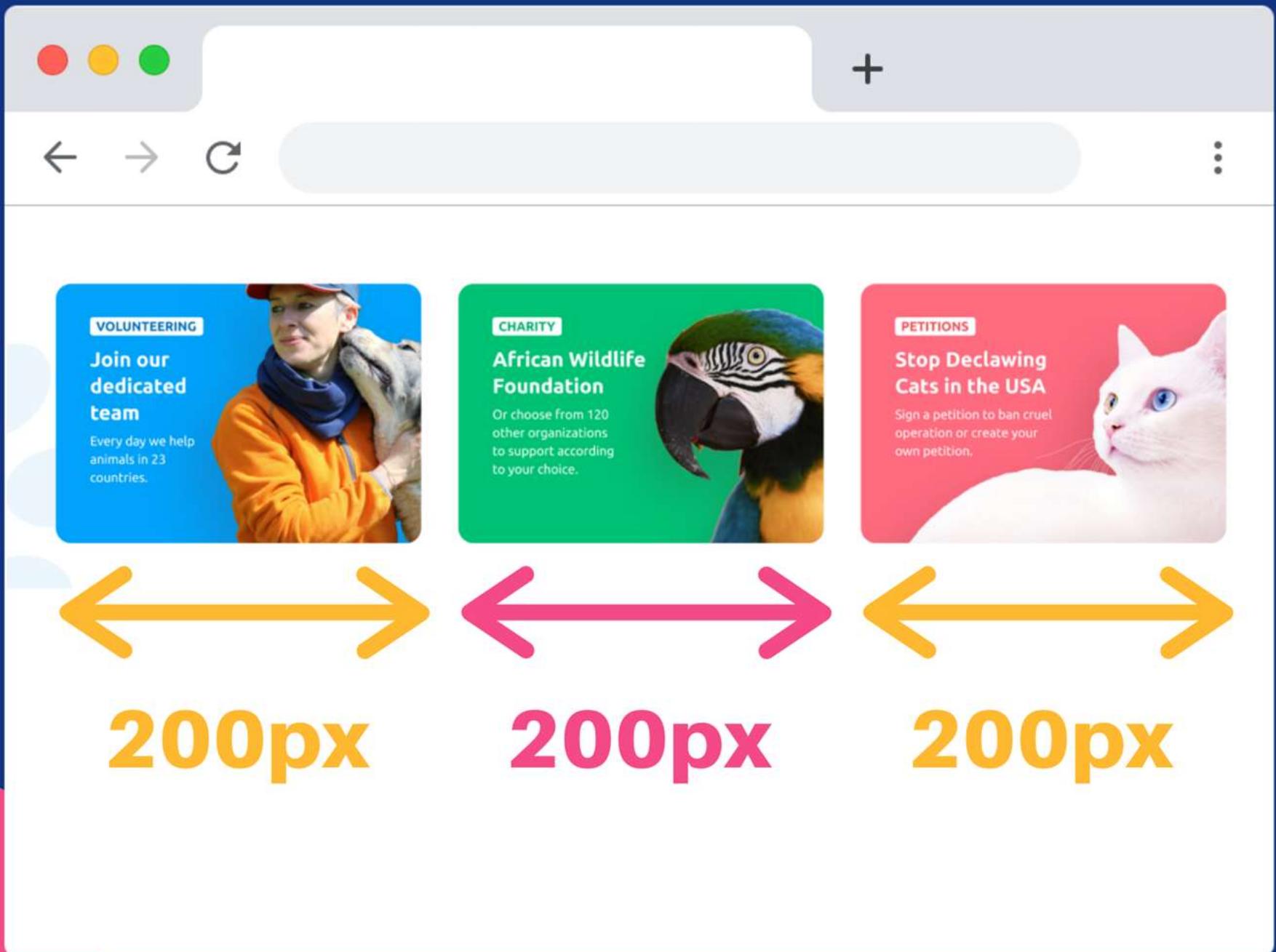
**CHARITY**

African Wildlife Foundation  
Or choose from 120 other organizations to support according to your choice.

**PETITIONS**

Stop Declawing Cats in the USA  
Sign a petition to ban cruel operation or create your own petition.

```
.container {  
  display: grid;  
  grid-template-columns: repeat(3, minmax(200px, 300px));  
}
```



# Grid implícita

The image shows a web browser window with three cards displayed:

- VOLUNTEERING**: A woman in an orange sweatshirt and blue scarf holding a grey animal. Text: "Join our dedicated team. Every day we help animals in 23 countries."
- CHARITY**: An African Macaw parrot. Text: "African Wildlife Foundation. Or choose from 120 other organizations to support according to your choice."
- PETITIONS**: A white cat with one blue eye and one green eye. Text: "Stop Declawing Cats in the USA. Sign a petition to ban cruel operation or create your own petition."

A yellow double-headed arrow is positioned below the Charity and Petitions cards, with the text "300px" written in large yellow font between them, indicating the implicit grid column width.

# auto-fit

The image shows a code editor interface with two tabs: "HTML" and "CSS". The "HTML" tab contains the following code:

```
1 <div class="container">
2   <div class="item"></div>
3   <div class="item"></div>
4   <div class="item"></div>
5   <div class="item"></div>
6   <div class="item"></div>
7 </div>
```

The "CSS" tab contains the following code:

```
1 .container {
2   background: papayawhip;
3   display: grid;
4   grid-template-columns: repeat(auto-fit, minmax(20px, 1fr));
5 }
6
7 .item {
8   background: papayawhip;
9   border: 4px solid hotpink;
10  width: 100%;
11  height: 50px;
12 }
```

Below the code editor, there is a visual representation of the grid layout. It consists of a horizontal row divided into four equal-width columns by thick pink vertical lines. Each column contains a white rectangular box with a thin pink border, representing the "item" elements from the CSS code.

# auto-fill

The image shows a code editor interface with two tabs: "HTML" and "CSS". The "HTML" tab contains the following code:

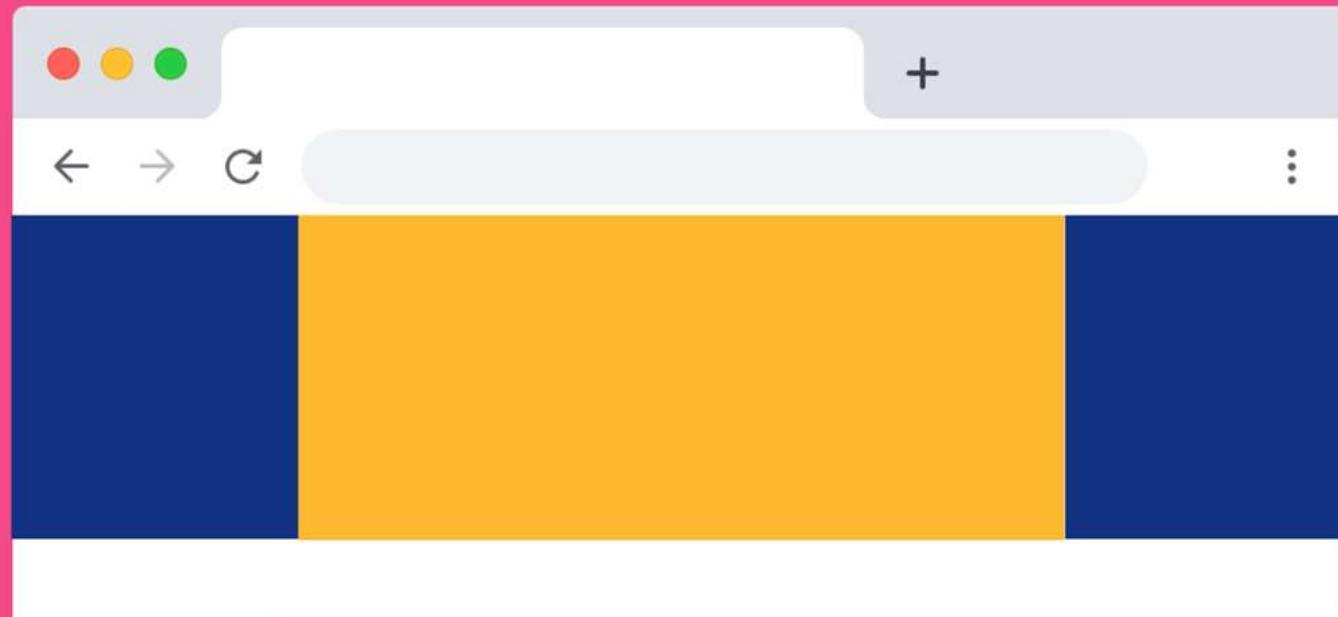
```
1 <div class="container">
2   <div class="item"></div>
3   <div class="item"></div>
4   <div class="item"></div>
5   <div class="item"></div>
6   <div class="item"></div>
7 </div>
```

The "CSS" tab contains the following code:

```
1 .container {
2   background: papayawhip;
3   display: grid;
4   grid-template-columns: repeat(auto-fill, minmax(20px, 1fr));
5 }
6
7 .item {
8   background: papayawhip;
9   border: 4px solid hotpink;
10  width: 100%;
11  height: 50px;
12 }
```

A tooltip is visible at the bottom left, showing the element path: `iframe#iFrameKey-92be4402-2 62b-fd3f-2e4e-29d93b28d3e3.r result-iframe`. A status bar at the bottom also displays the element path: `div.container 871.2 × 58`.

# fit-content()



```
.container{  
  display: grid;  
  grid-template-columns: auto fit-content(800px) auto;  
}
```



**¡MANOS  
AL CÓDIGO!**

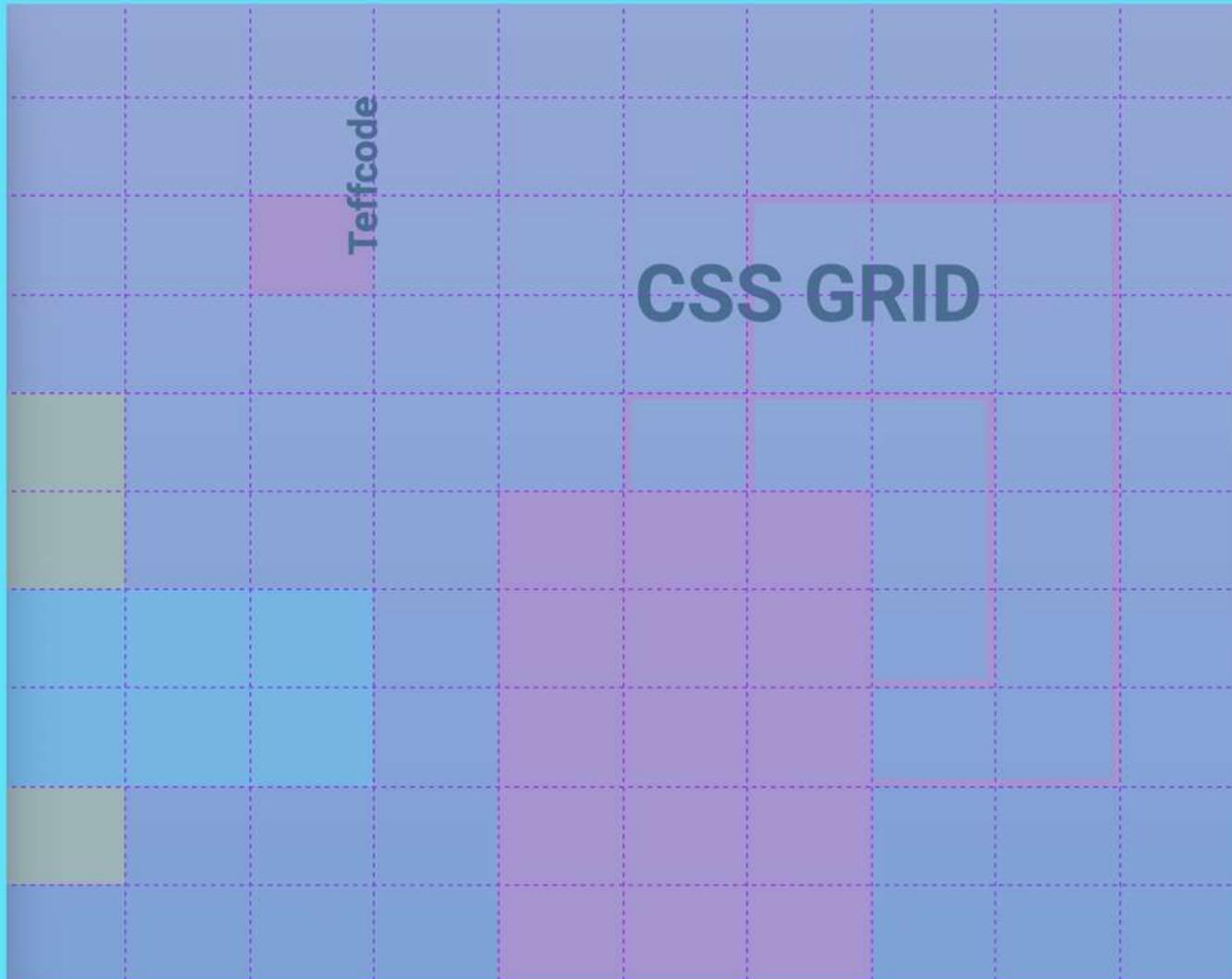
CONSTRUCCIÓN DE  
LA GRID PRINCIPAL

Teffcode

# CSS GRID

# COLUMNAS

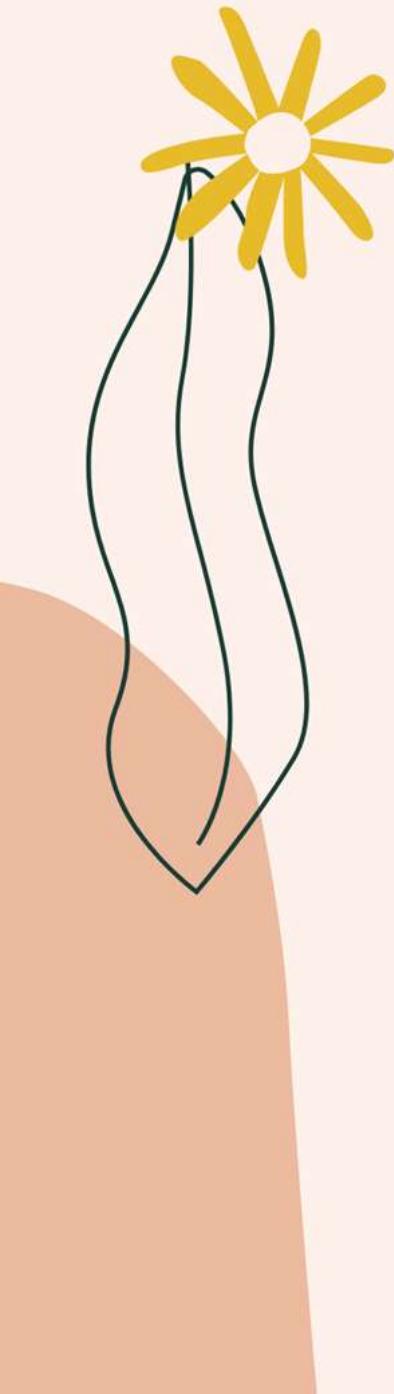
FILAS



**Yo: Vamos al código !  
Mi página web:**



# UBICACIÓN + Reto

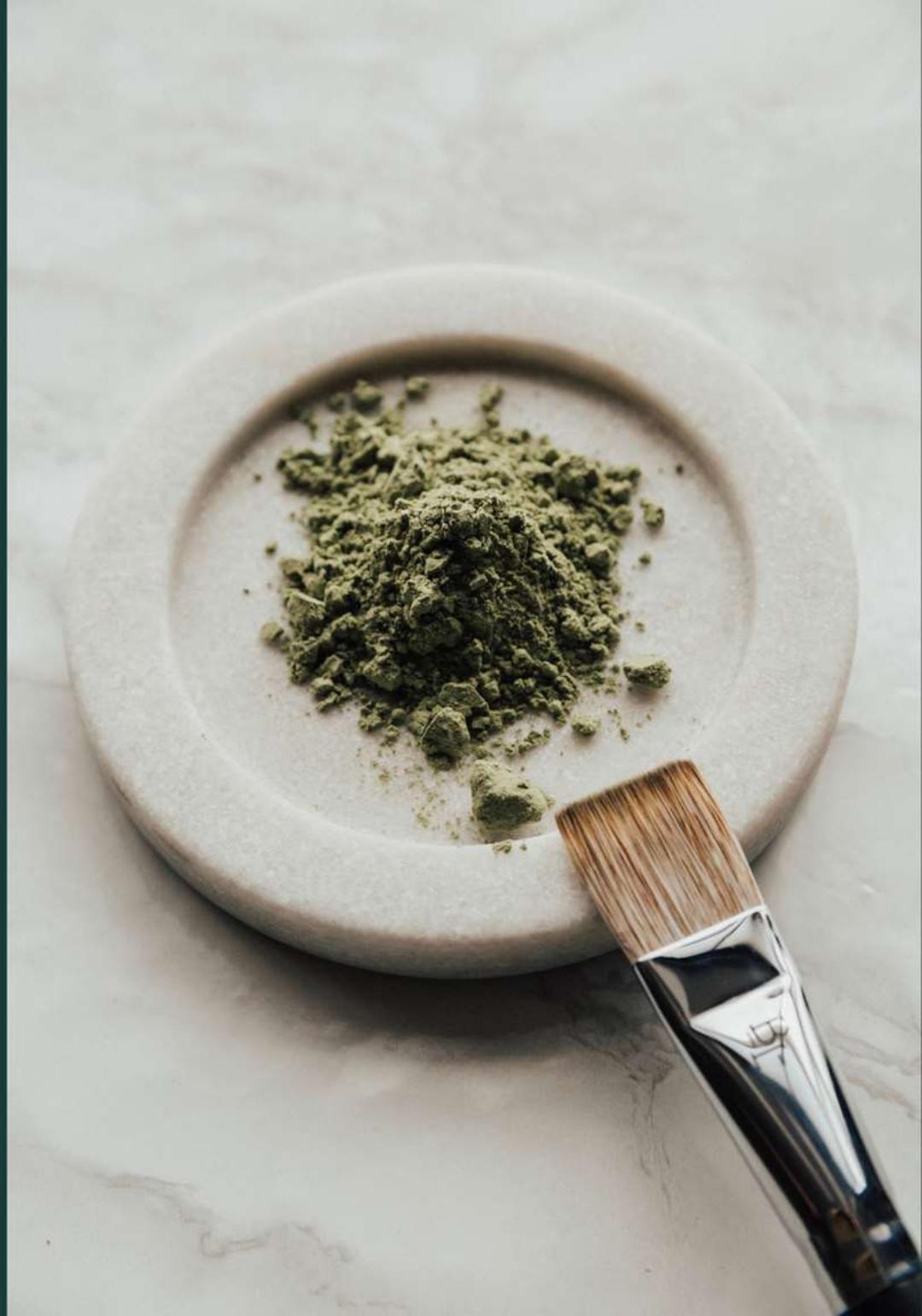


A digital interface window with a light gray header bar featuring three colored dots (red, yellow, green) on the left and a plus sign (+) on the right. Below the header is a toolbar with arrows and a 'C' icon. The main area contains a 5x5 grid of yellow squares with red outlines. To the right of the grid is a text area with the following text:

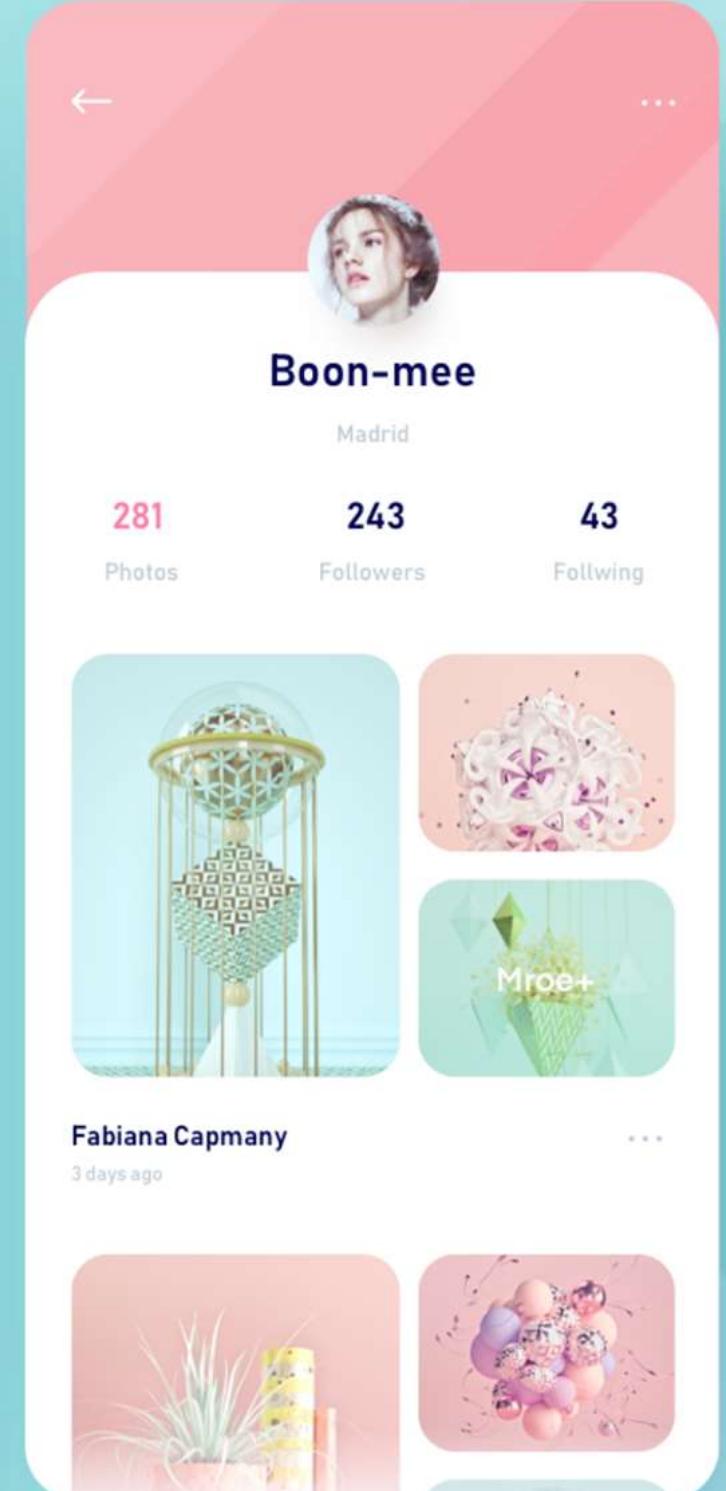
Ya  
hablamos  
del  
contenedor

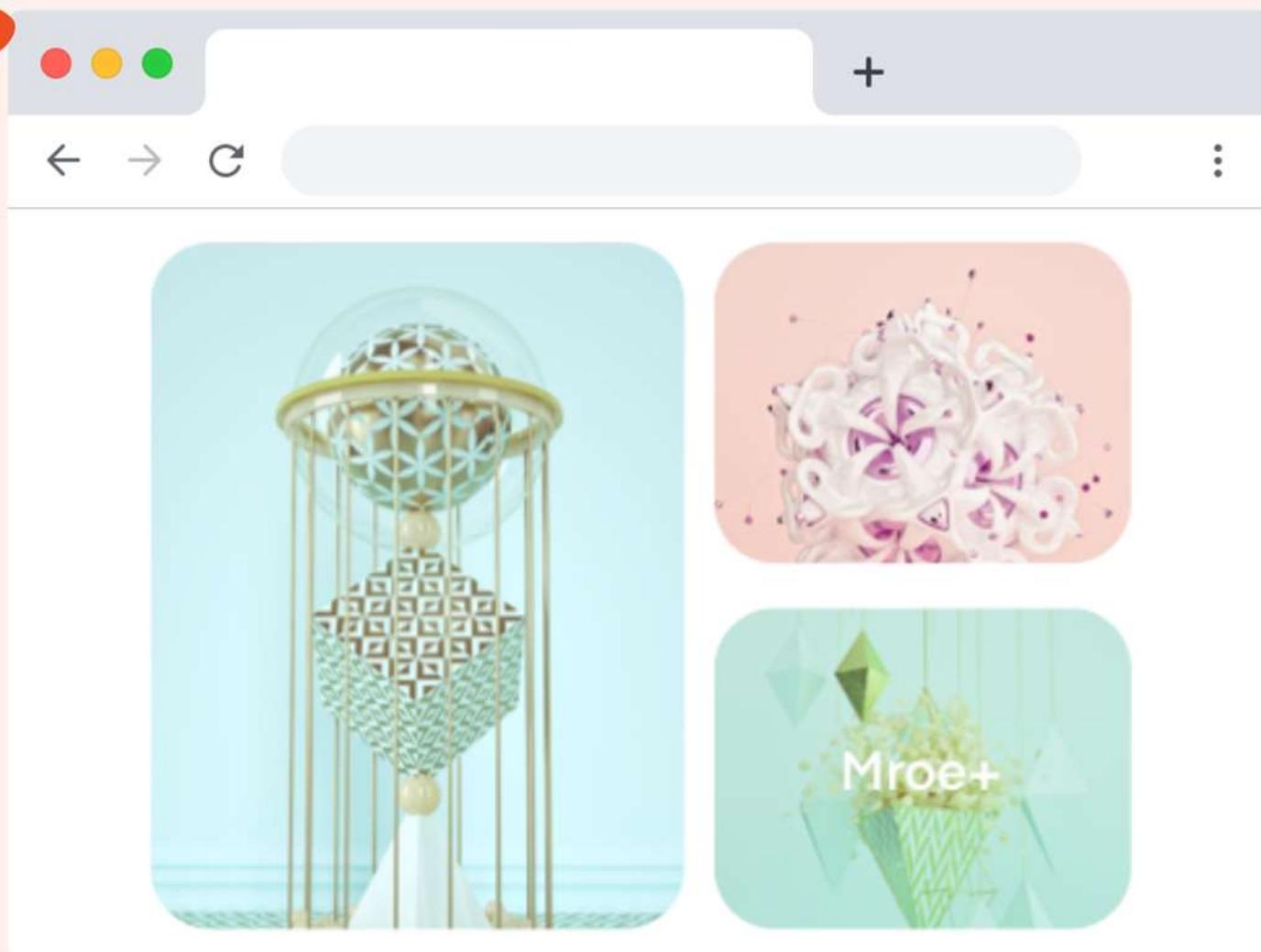


Ahora  
hablaremos de  
los elementos  
hijos

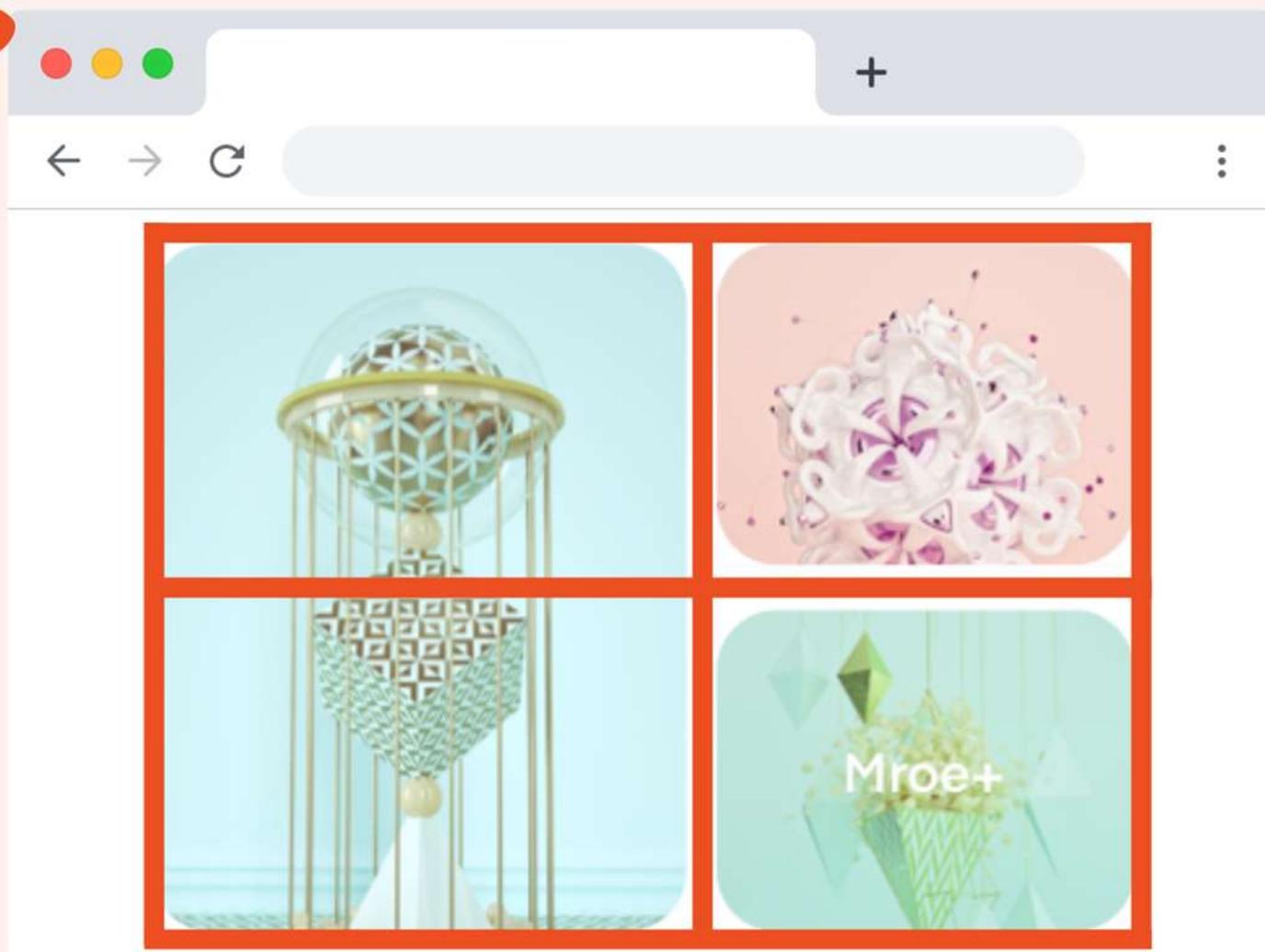


Supongamos  
que tenemos  
que hacer  
esta sección





m



moe

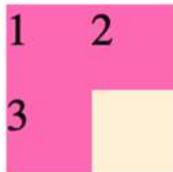


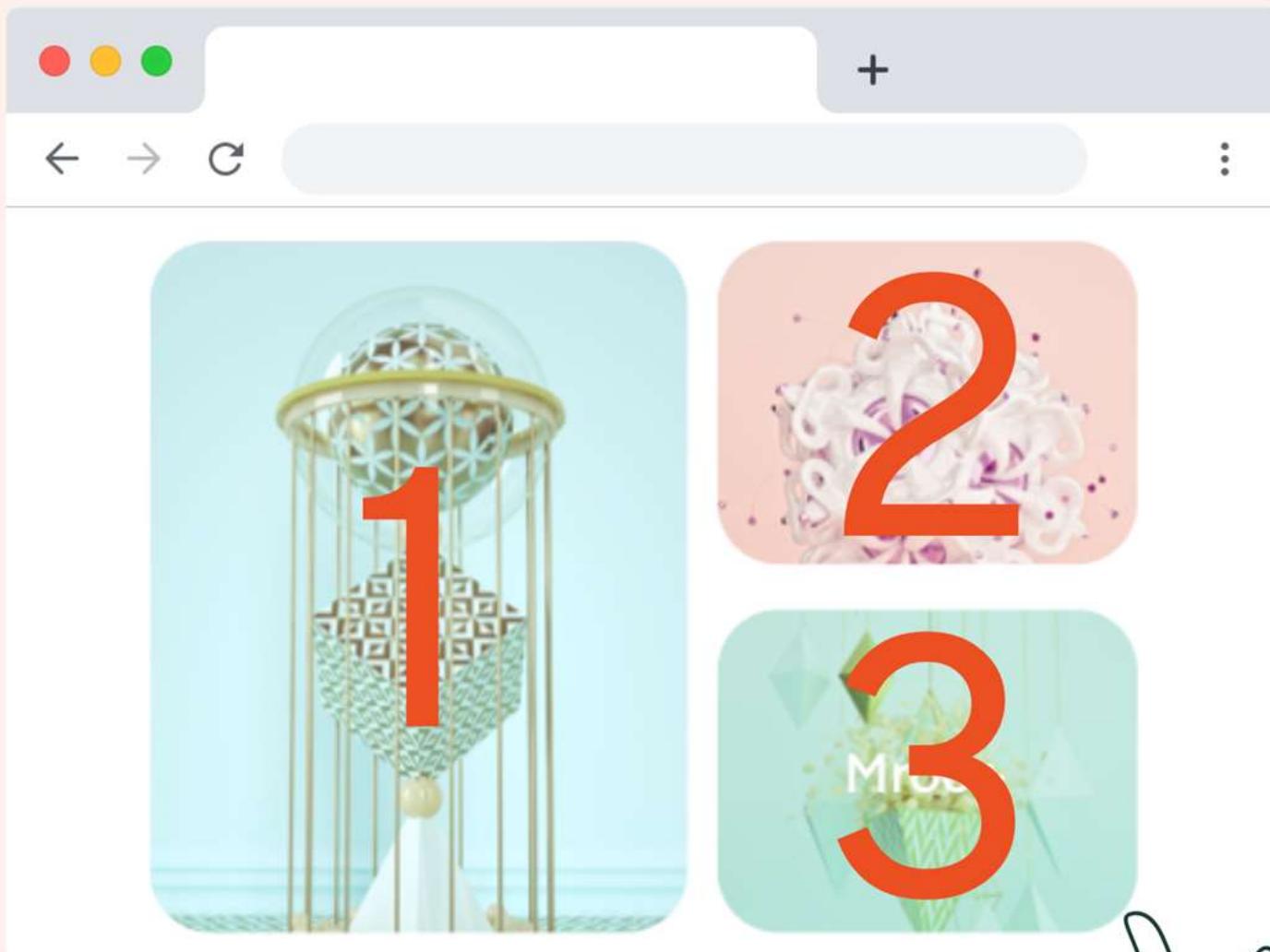
## HTML

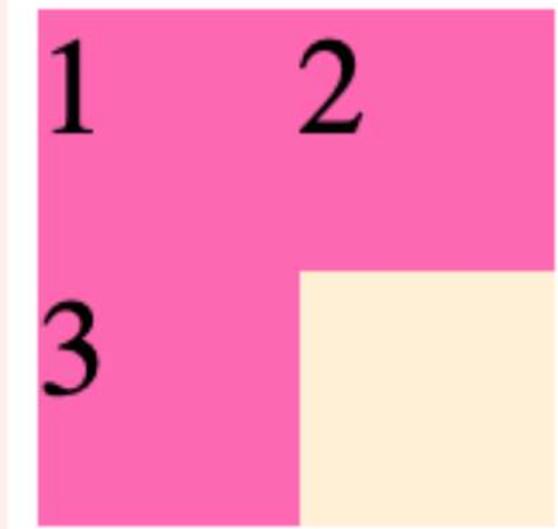
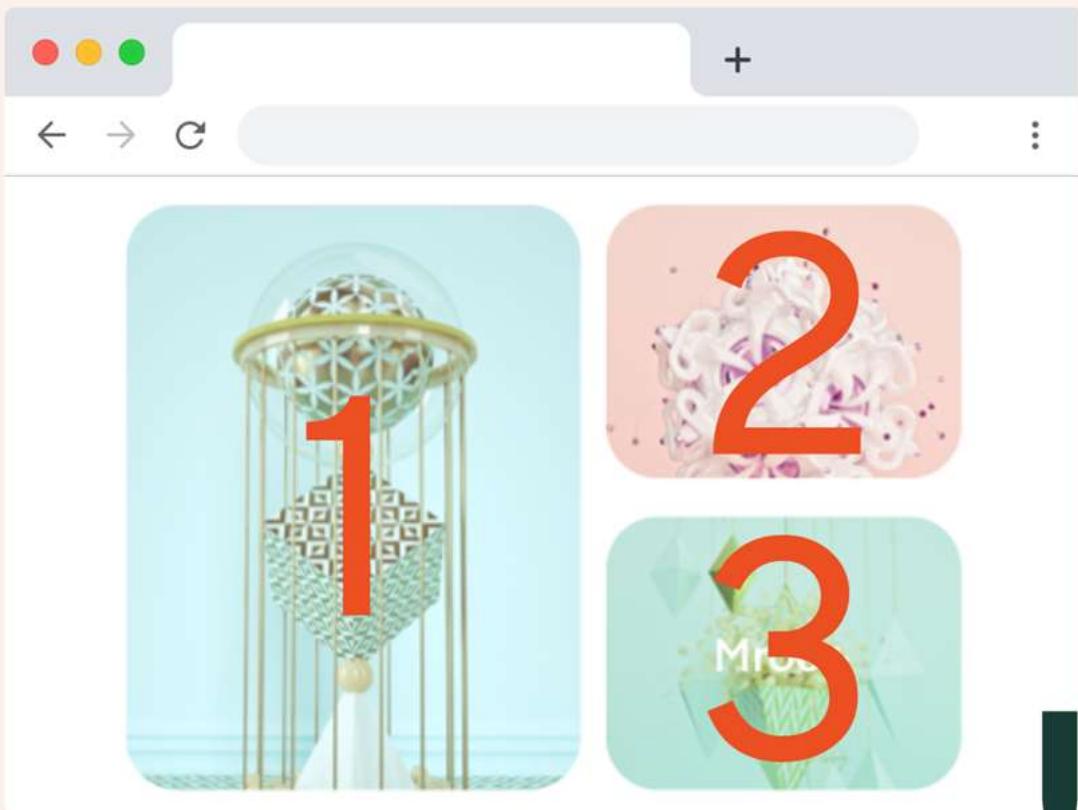
```
1 <div class="container">
2   <div class="item">1</div>
3   <div class="item">2</div>
4   <div class="item">3</div>
5 </div>
```

## CSS

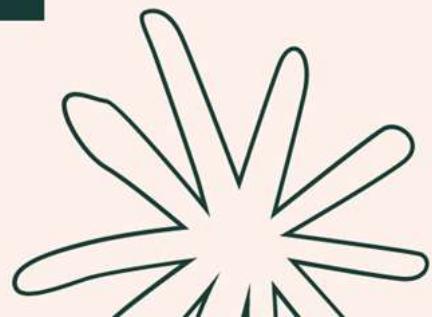
```
1 .container {
2   background: papayawhip;
3   display: grid;
4   grid-template-columns: repeat(2, 1fr);
5   width: 60px;
6 }
7
8 .item {
9   background: hotpink;
10  width: 30px;
11  height: 30px;
12 }
```



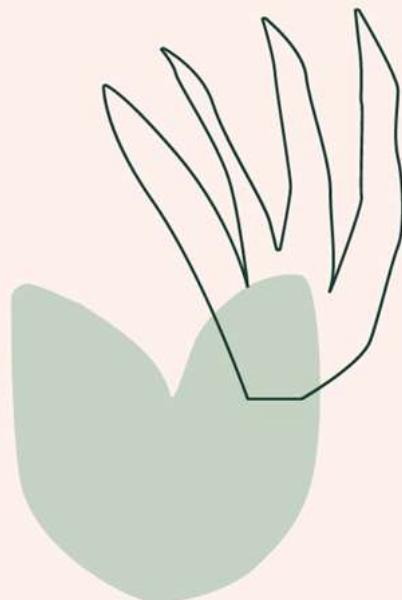
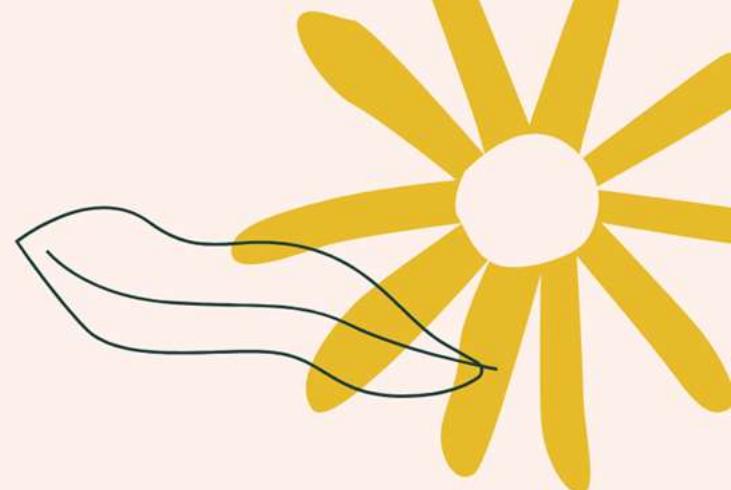




!=



# PROPIEDADES



Las dividiremos en  
tres grupitos



# Grupito 1



grid-column-start



grid-column-end

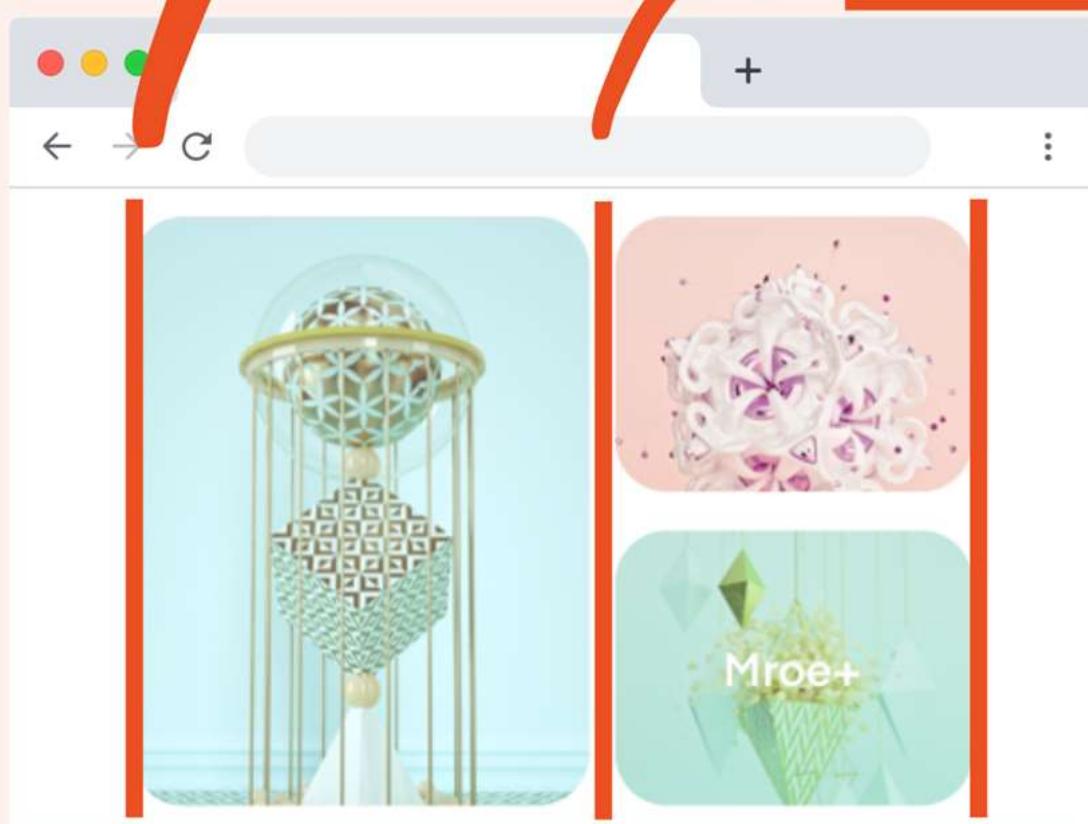


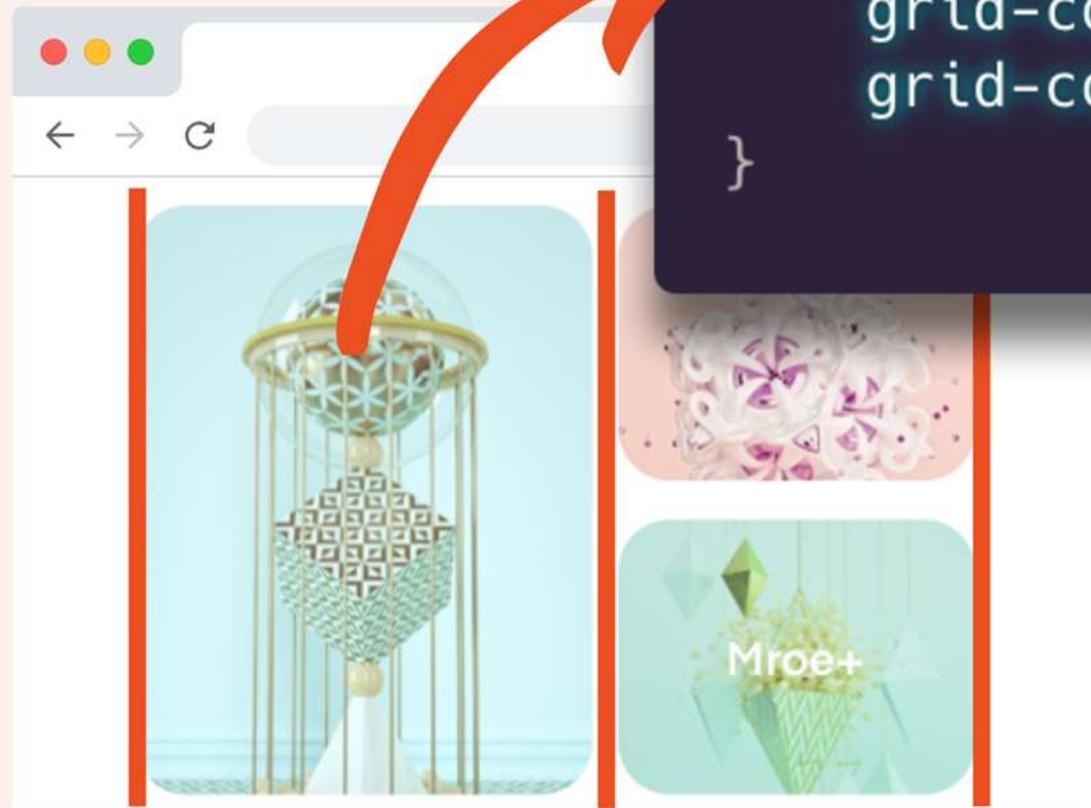
grid-column



grid-column-start

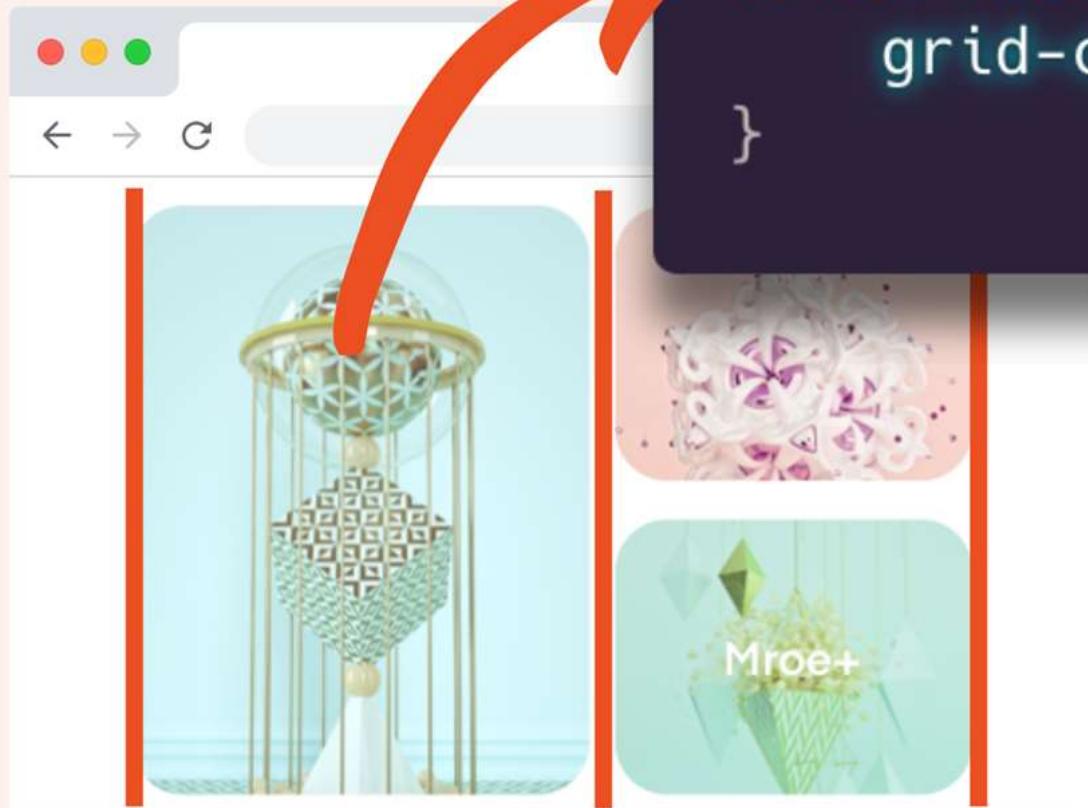
grid-column-end





```
.item-one {  
    grid-column-start: 1;  
    grid-column-end: 2;  
}
```





```
...  
.item-one {  
    grid-column: 1 / 2;  
}
```





# Grupito 2

grid-row-start



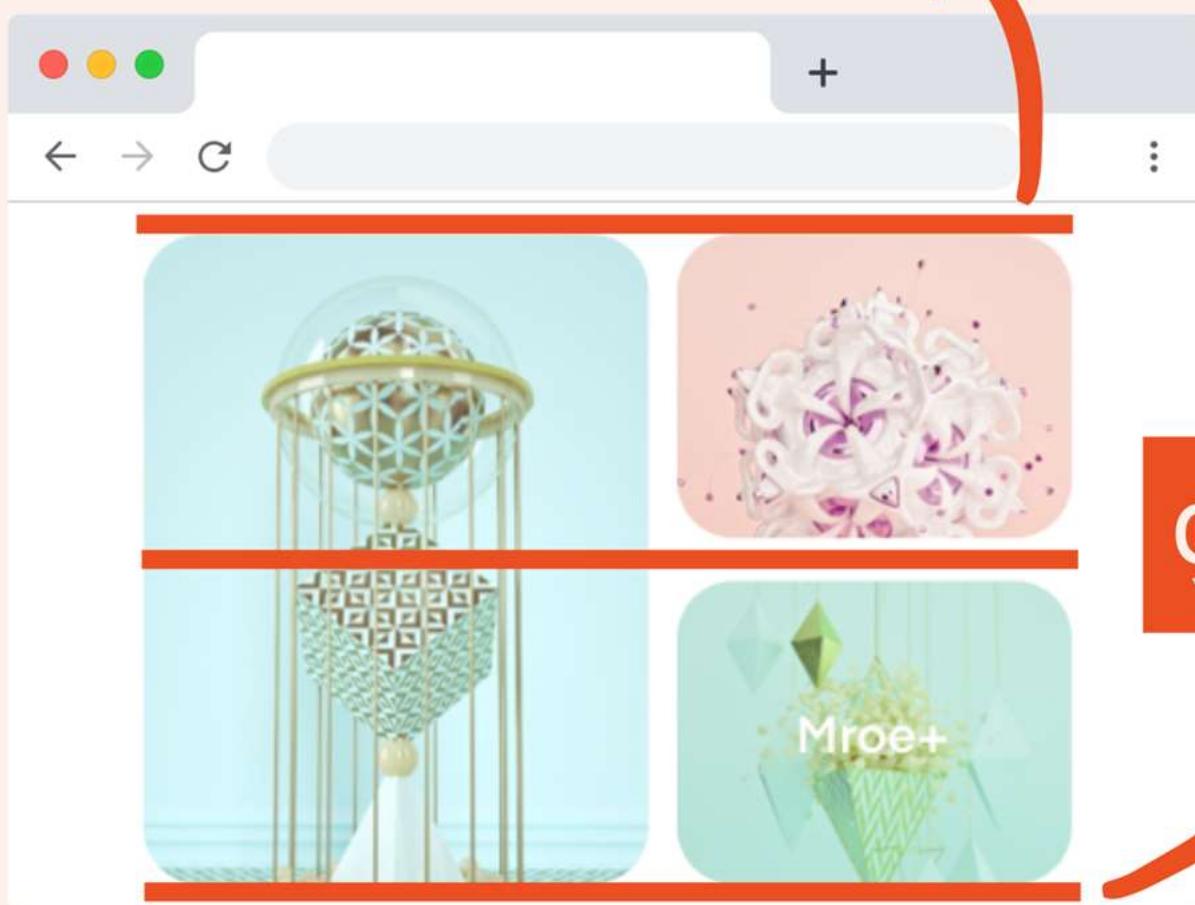
grid-row-end



grid-row

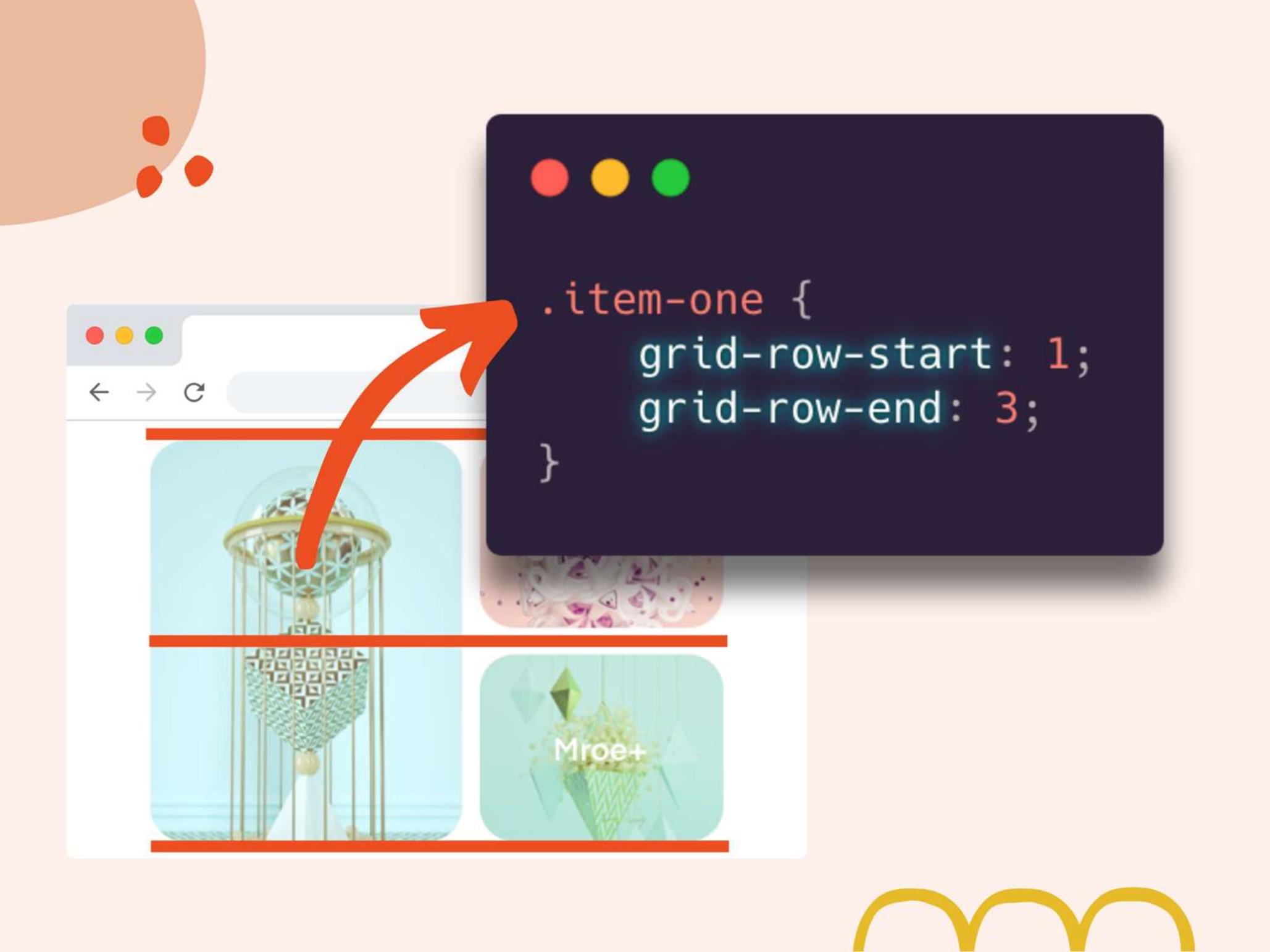
A large, solid orange circle at the bottom center of the slide.

# grid-row-start



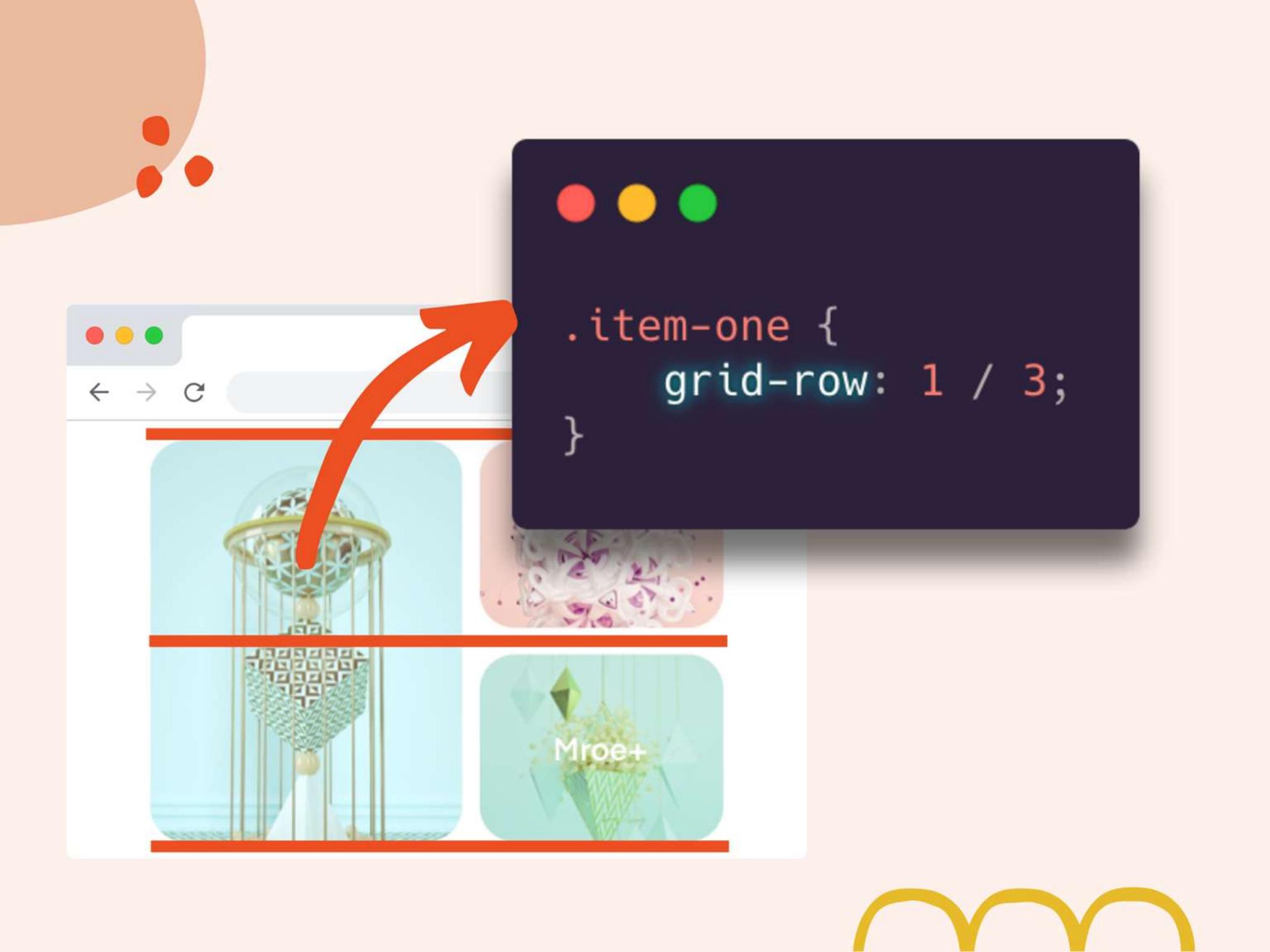
# grid-row-end





```
.item-one {  
    grid-row-start: 1;  
    grid-row-end: 3;  
}
```





```
.item-one {  
    grid-row: 1 / 3;  
}
```



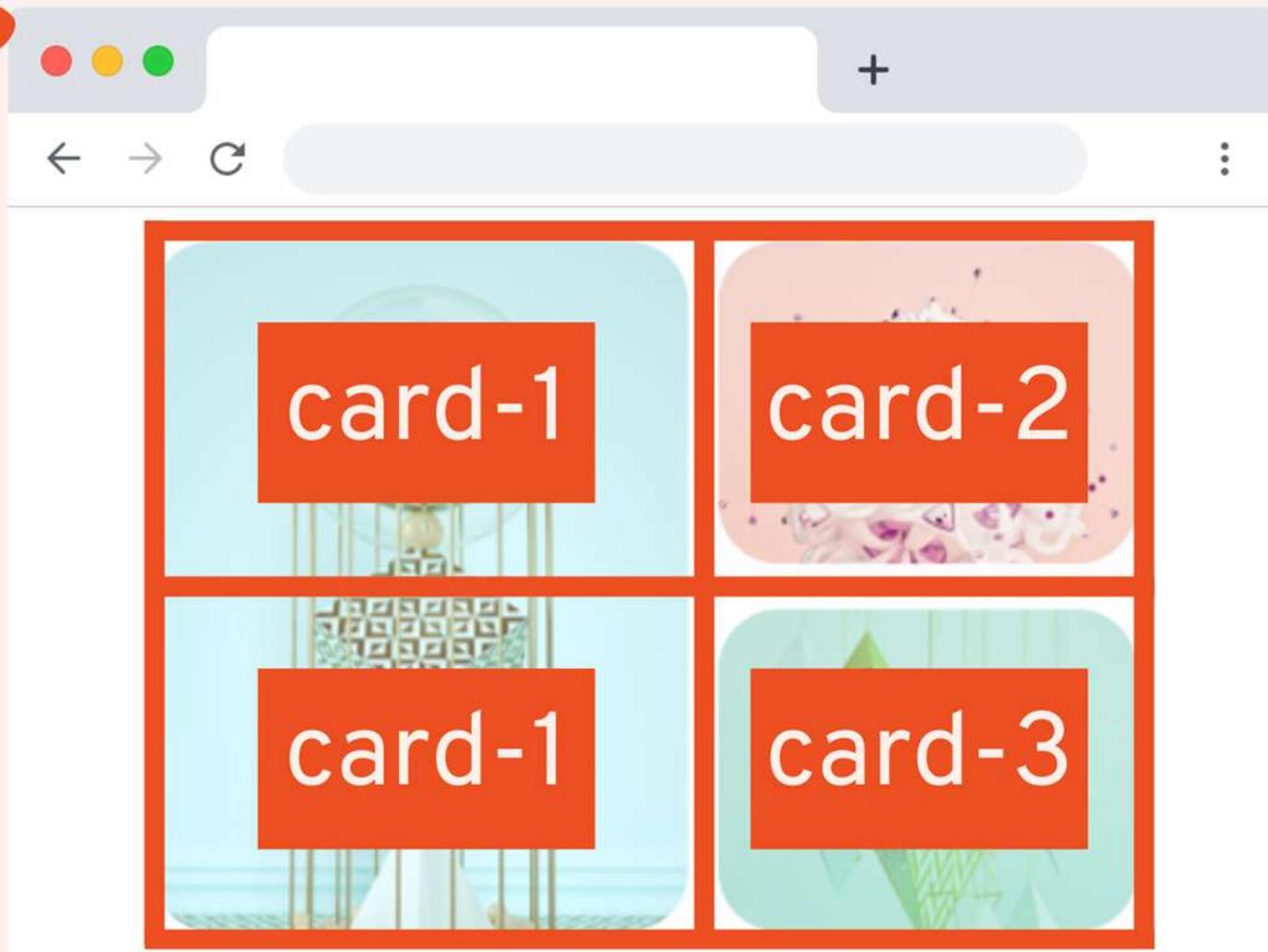


# Grupito 3



grid-area





www



```
.container {  
    display: grid;  
    grid-template-columns: repeat(2, 1fr);  
    grid-template-areas:  
        "card-1 card-2"  
        "card-1 card-3";  
}
```



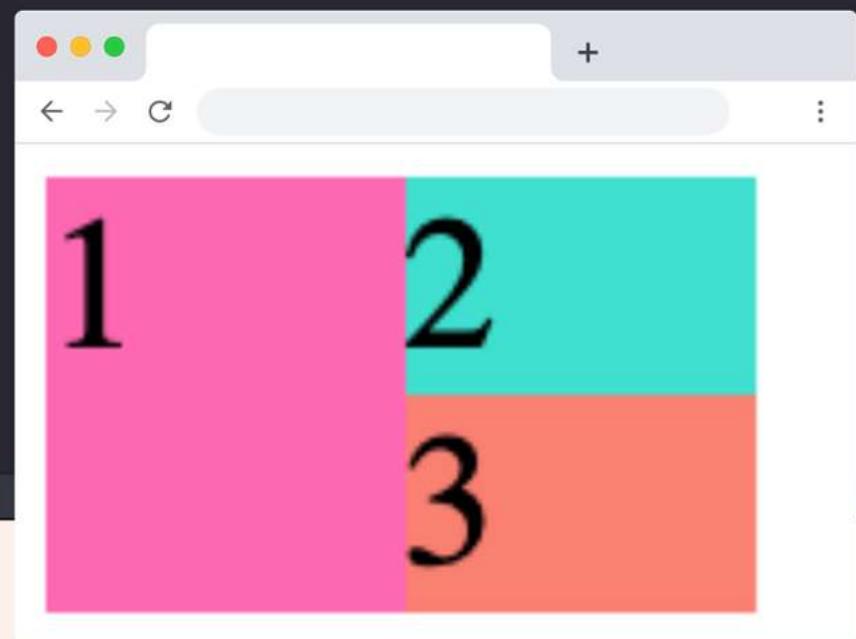


```
.item-one {  
    grid-area: card-1;  
}
```



## HTML

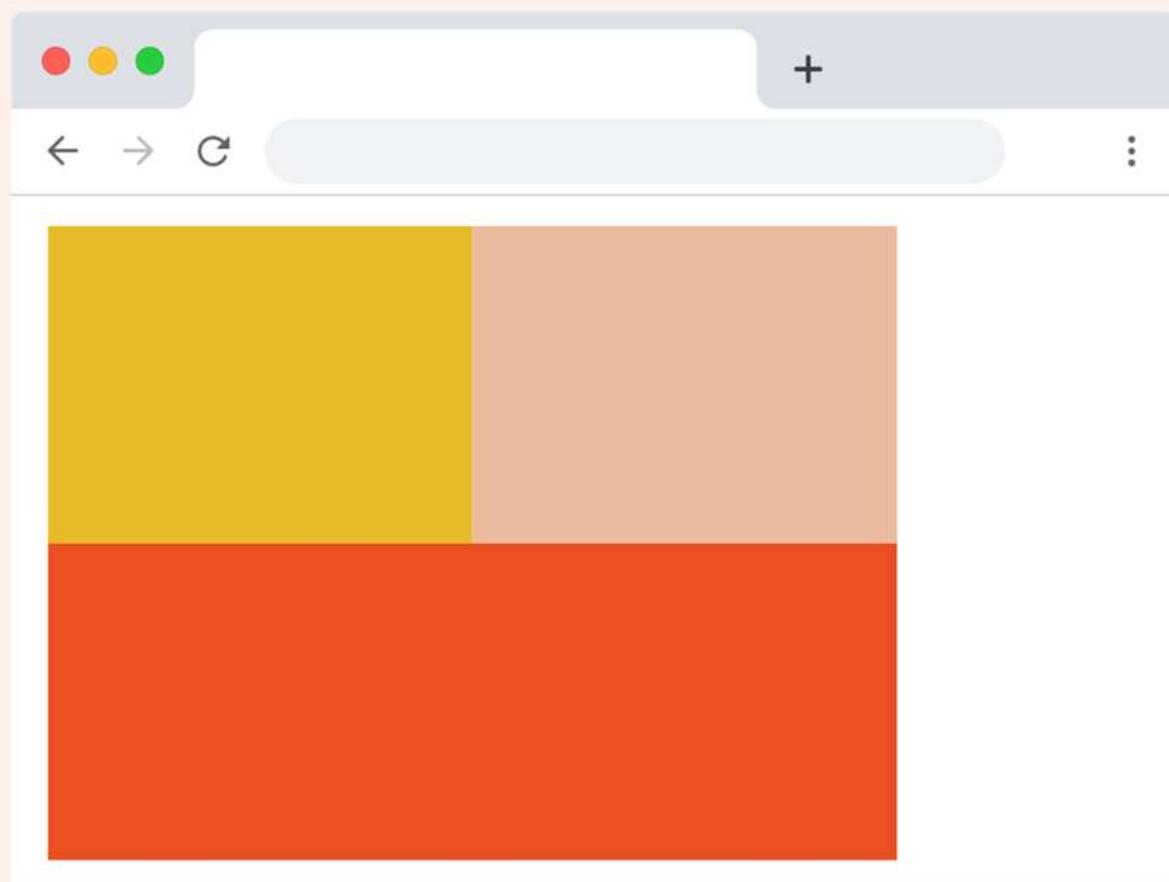
```
1 <div class="container">  
2   <div class="item-one">1</div>  
3   <div class="item-two">2</div>  
4   <div class="item-three">3</div>  
5 </div>
```



## CSS

```
1 .container {  
2   background: papayawhip;  
3   display: grid;  
4   grid-template-columns: repeat(2, 1fr);  
5   grid-template-areas:  
6     "card-1 card-2"  
7     "card-1 card-3";  
8   width: 60px;  
9 }  
10  
11 .item-one {  
12   background: hotpink;  
13   grid-area: card-1;  
14 }
```

# Reto



# Alineamiento + Quíz





# Tres propiedades

**justify-self**

**align-self**

**place-self**

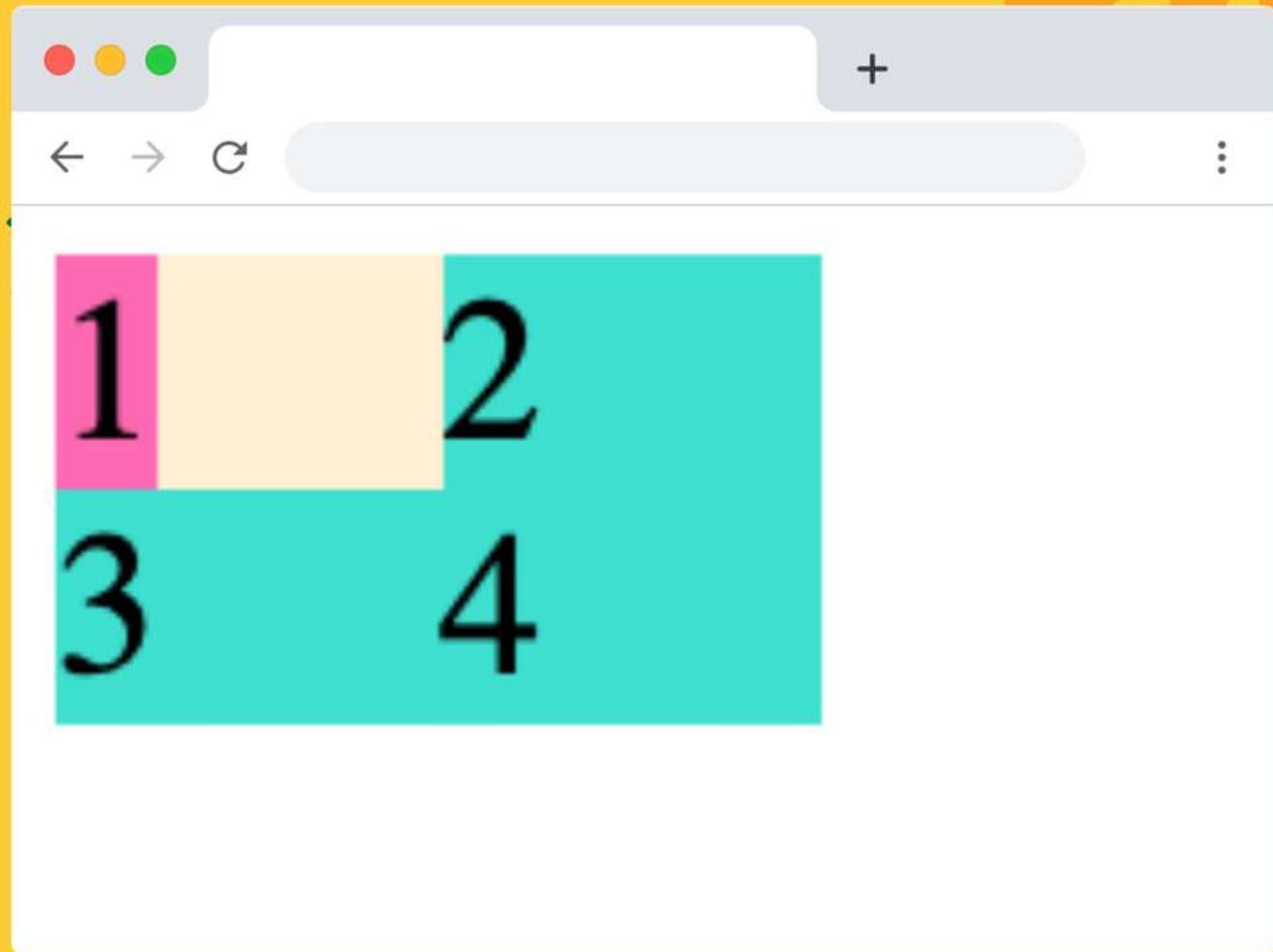


# **justify-self**

**inline axis  
(row axis)**



# justify-self





# HTML

```
<div class="container">
  <div class="item">1</div>
  <div class="item">2</div>
  <div class="item">3</div>
  <div class="item">4</div>
</div>
```





# CSS

```
.container {  
    background: papayawhip;  
    display: grid;  
    grid-template-columns: repeat(2, 1fr);  
    width: 60px;  
}  
  
.item:nth-child(1) {  
    background: hotpink;  
    justify-self: start;  
}  
  
.item:not(:nth-child(1)) {  
    background: turquoise;  
}
```



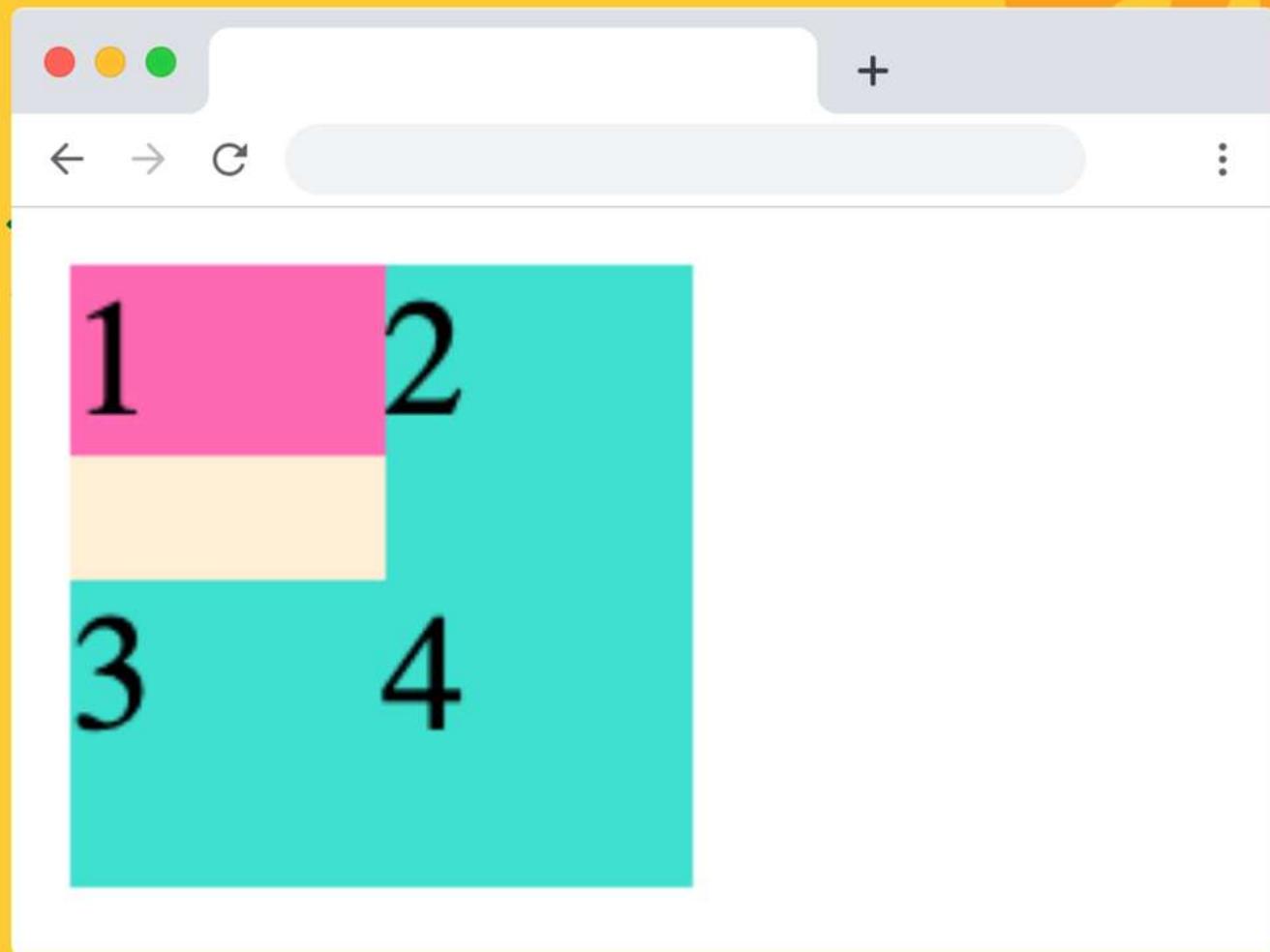


**align-self**

**block axis  
(column axis)**



# align-self





# HTML



```
<div class="container">  
    <div class="item">1</div>  
    <div class="item">2</div>  
    <div class="item">3</div>  
    <div class="item">4</div>  
</div>
```





# CSS

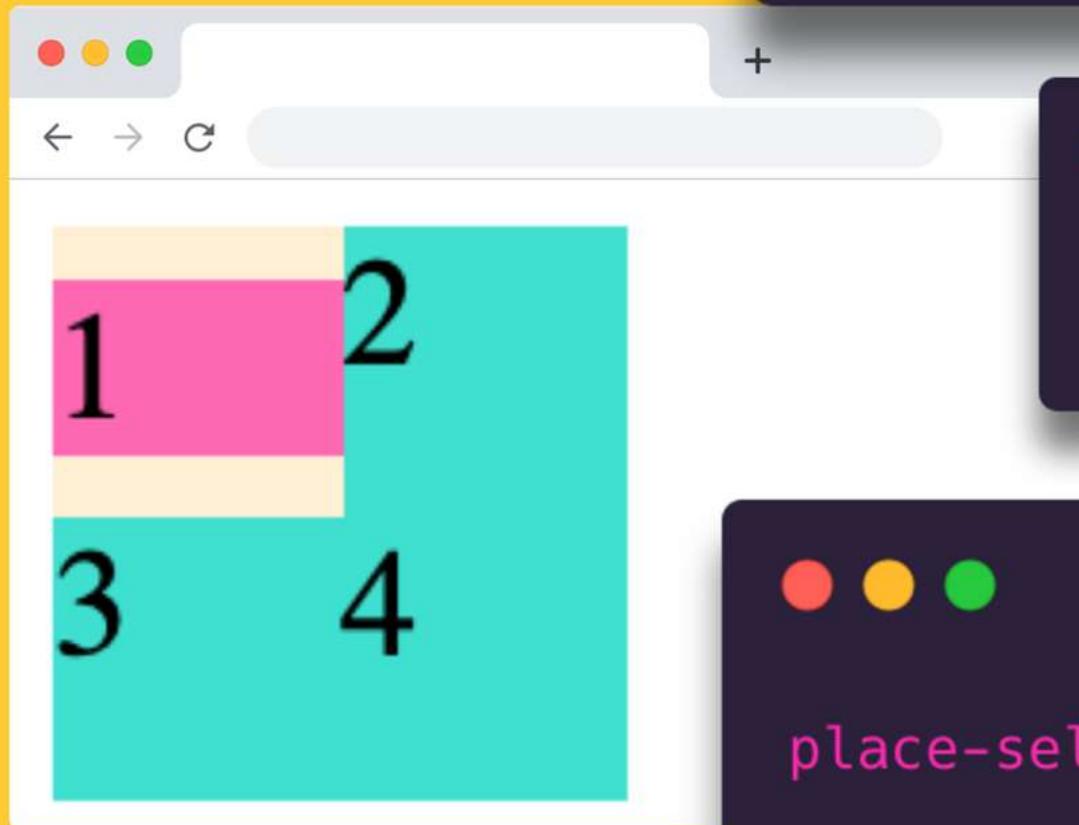
```
.container {  
    background: papayawhip;  
    display: grid;  
    grid-template-columns: repeat(2, 1fr);  
    width: 60px;  
    height: 60px;  
}  
  
.item:nth-child(1) {  
    background: hotpink;  
    align-self: start;  
}  
  
.item:not(:nth-child(1)) {  
    background: turquoise;  
}
```



# place-self

## <align-self> / <justify-self>

# place-self Quíz :)



• • • A

```
place-self: stretch;
```

• • • B

```
place-self: center;
```

• • • C

```
place-self: center stretch;
```



**¡MANOS  
AL CÓDIGO!**

UBICACIÓN Y  
ALINEAMIENTO(PARTE 1)



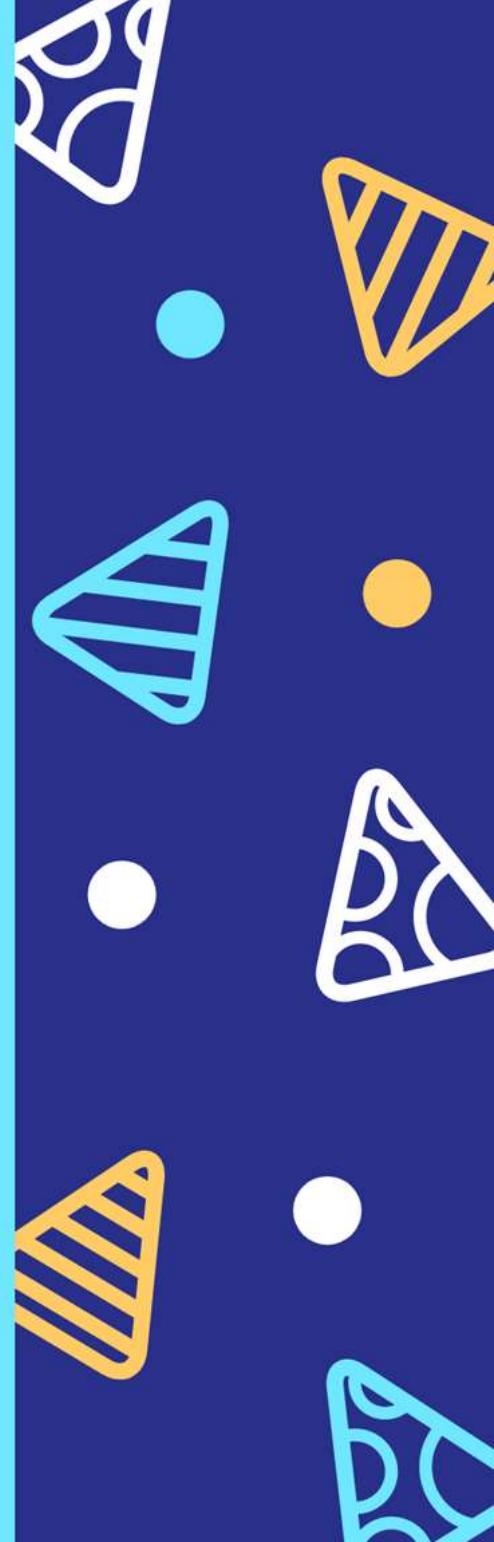
**¡MANOS  
AL CÓDIGO!**

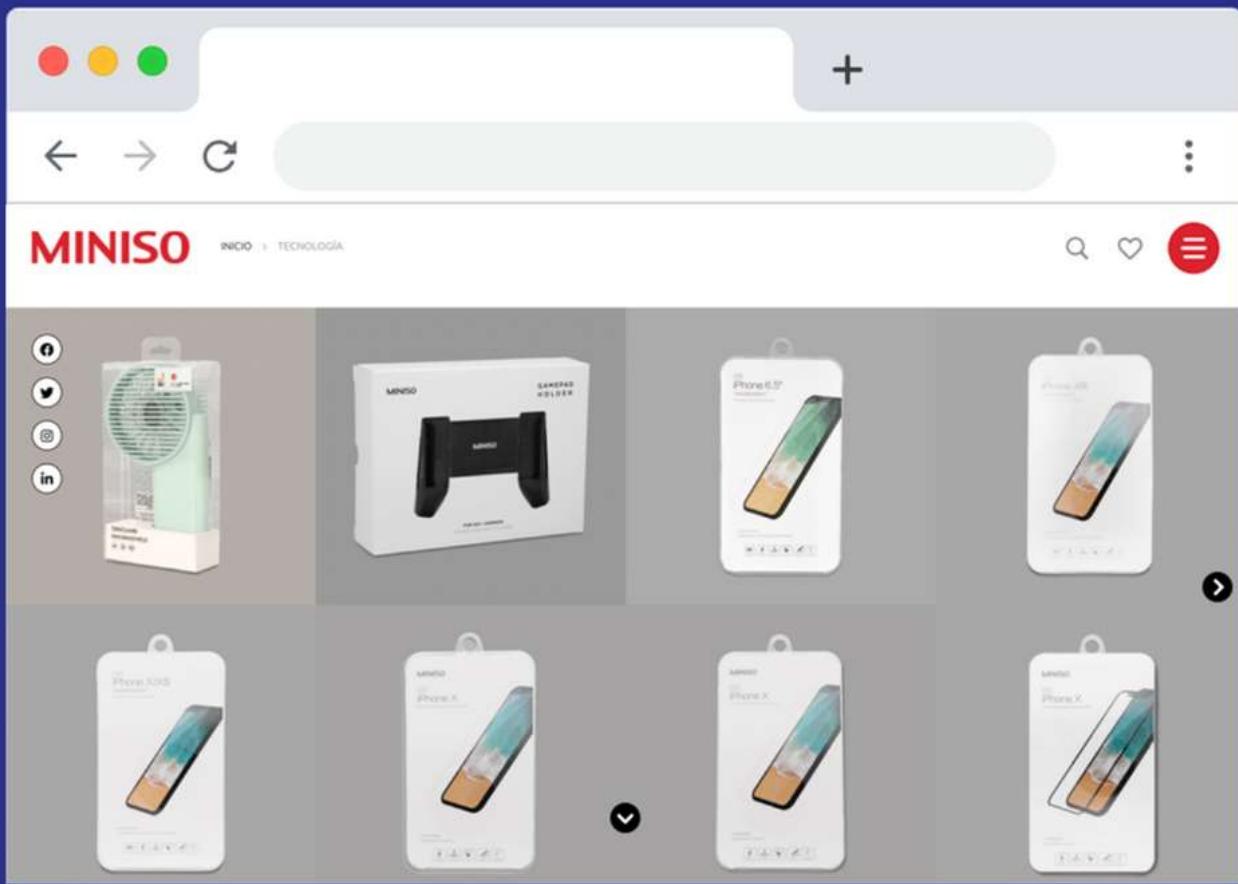
UBICACIÓN Y  
ALINEAMIENTO(PARTE 2)

# DISEÑO RESPONSIVO SIN MEDIA QUERIES

+ QUIZ

# **CUANDO HABLAMOS DE RESPONSIVE...**





**QUE TU SITIO WEB SE VEA  
BIEN EN TU COMPU**



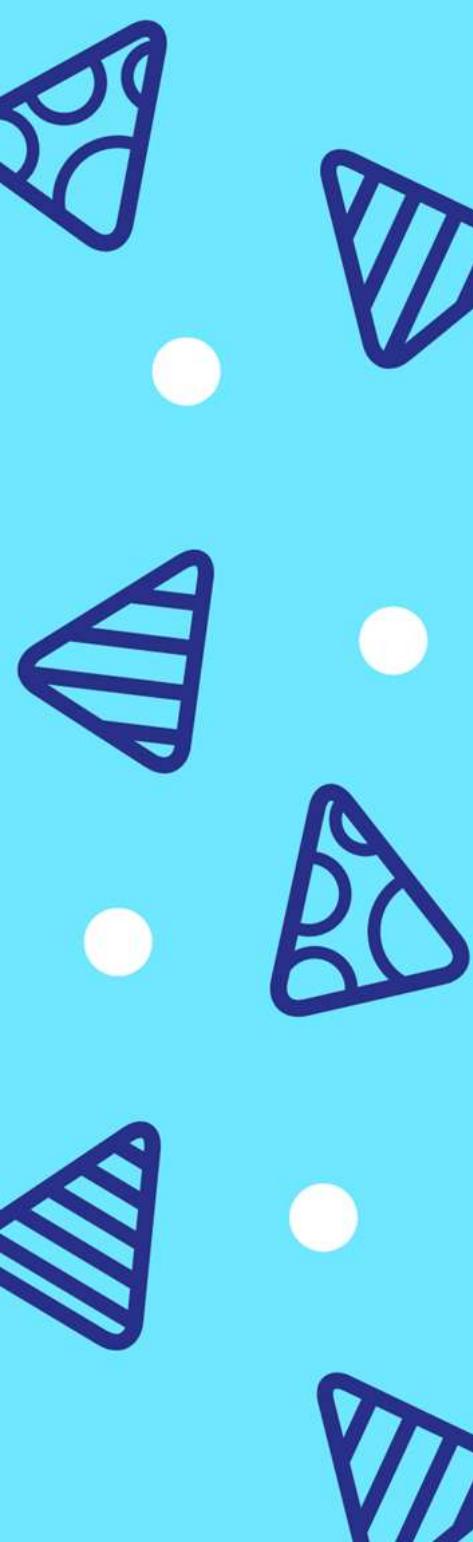
**Y QUE TAMBIÉN  
SE VEA BIEN EN  
TU CELU**



**PERO, TU CELU PUEDE SER  
CUALQUIERA DE ESTOS...**

**Y NO SOLO ESO...**





**CUANDO  
HABLAMOS DE  
DISEÑO  
RESPONSIVO....**

The image shows a desktop computer setup with a monitor, a tablet, and a smartphone, all displaying the same website for "WEBDESIGN".

**Monitor (Desktop View):**

- Header:** Includes "WEBDESIGN" logo, "HOME" (highlighted in blue), "REGISTER", "HELP", "YOUR SITE", and "DESIGNS".
- Main Content:** Text "Design your own website yourself with our easy online tool!" over a background image of a lighthouse.
- Bottom Section:** Buttons for "GET OUR APP", "5M WEBSITES WORLDWIDE", and "SHOP THEMES".

**Tablet (Tablet View):**

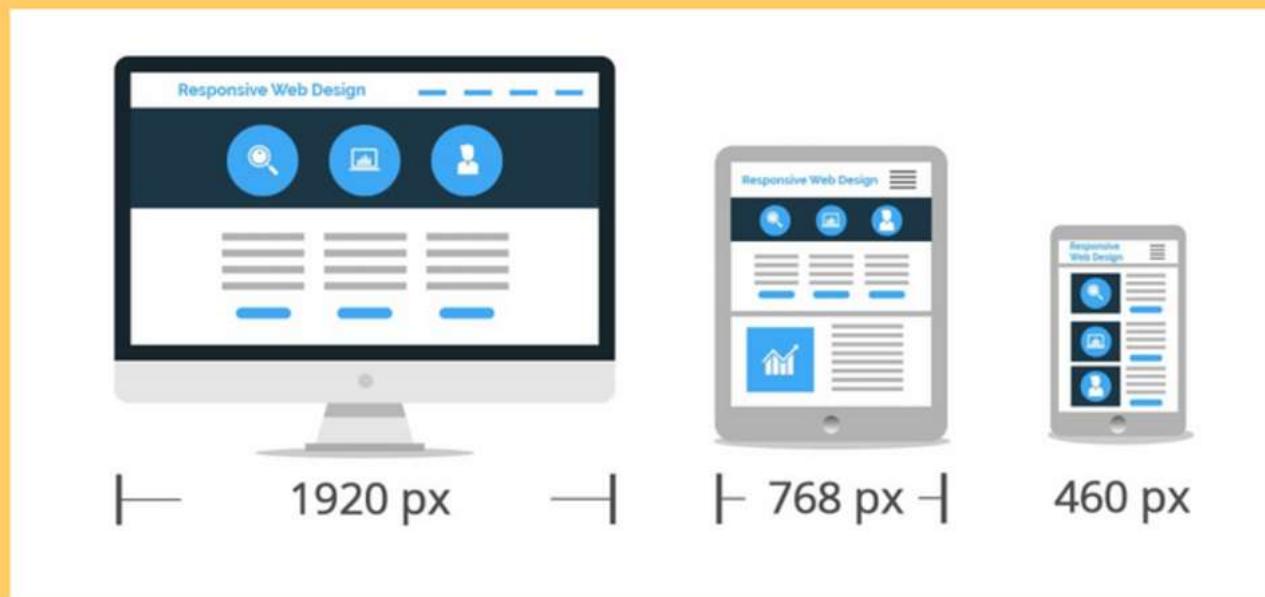
- Header:** Includes "WEBDESIGN" logo, "HOME", "REGISTER", "HELP", "YOUR SITE", and "DESIGNS".
- Main Content:** Text "Design your own website yourself with our easy online tool!" over a background image of a lighthouse.
- Bottom Section:** Buttons for "GET OUR APP", "5M WEBSITES WORLDWIDE", and "SHOP THEMES".

**Smartphone (Mobile View):**

- Header:** Includes "WEBDESIGN" logo, "HOME", "REGISTER", "HELP", "YOUR SITE", and "DESIGNS".
- Main Content:** Text "Design your own website yourself with our easy online tool!" over a background image of a lighthouse.
- Bottom Section:** Buttons for "GET OUR APP", "5M WEBSITES WORLDWIDE", and "SHOP THEMES".

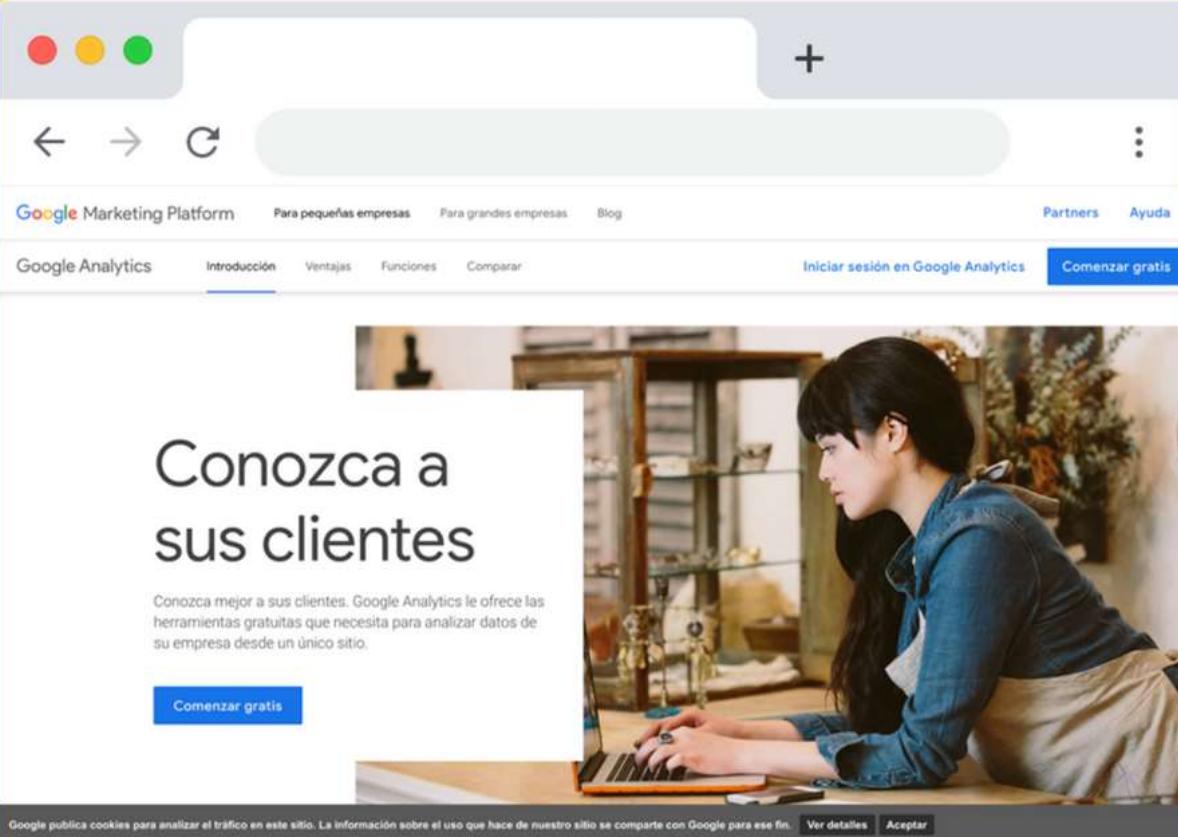
A white coffee mug is visible on the left side of the desk.

# **DEBEMOS SUMERGIRNOS EN EL MUNDO DE LOS PIXELES**



- **CUÁL ES MI PÚBLICO OBJETIVO?**
- **PARA QUÉ DISPOSITIVOS QUIERO DISEÑAR MI SITIO WEB?**

# GOOGLE ANALYTICS



The screenshot shows a web browser window displaying the Google Analytics homepage. The header includes the Google Marketing Platform logo and links for 'Para pequeñas empresas', 'Para grandes empresas', 'Blog', 'Partners', and 'Ayuda'. The main navigation bar has 'Google Analytics' selected, with other options like 'Introducción', 'Ventajas', 'Funciones', and 'Comparar'. A prominent blue button says 'Comenzar gratis'. The main content features a large image of a woman working at a laptop in a shop, with the text 'Conozca a sus clientes' (Get to know your customers) and a subtext explaining how Google Analytics helps analyze data from a single website. At the bottom, there's a cookie consent banner from Google.

Google Marketing Platform

Para pequeñas empresas

Para grandes empresas

Blog

Partners

Ayuda

Google Analytics

Introducción

Ventajas

Funciones

Comparar

Iniciar sesión en Google Analytics

Comenzar gratis

Conozca a sus clientes

Conozca mejor a sus clientes. Google Analytics le ofrece las herramientas gratuitas que necesita para analizar datos de su empresa desde un único sitio.

Comenzar gratis

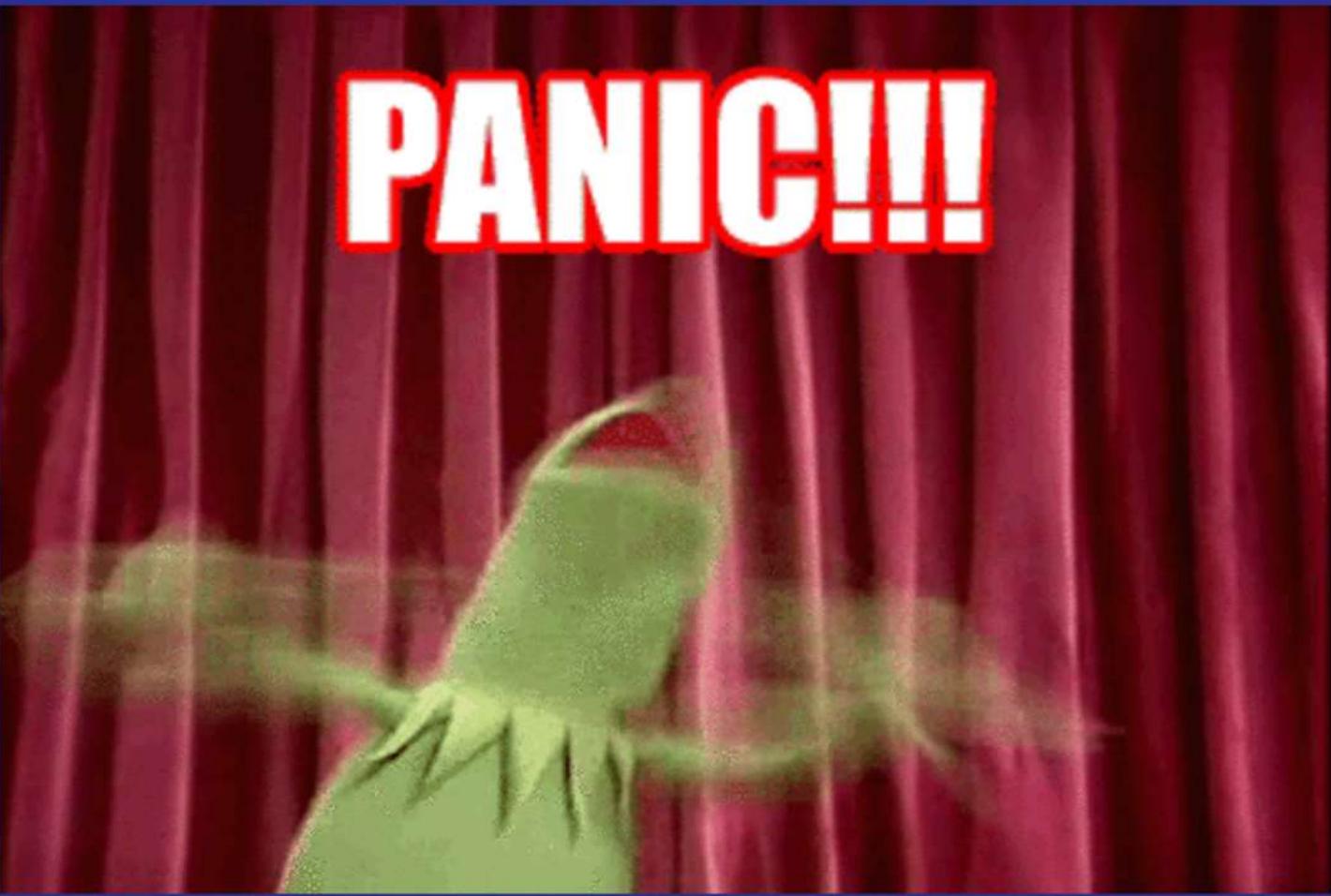
Google publica cookies para analizar el tráfico en este sitio. La información sobre el uso que hace de nuestro sitio se comparte con Google para ese fin. [Ver detalles](#) [Aceptar](#)

# IPHONE

Device	Native Resolution (Pixels)	UIKit Size (Points)	Native Scale factor	UIKit Scale factor
iPhone X	1125 x 2436	375 x 812	3.0	3.0
iPhone 8 Plus	1080 x 1920	414 x 736	2.608	3.0
iPhone 8	750 x 1334	375 x 667	2.0	2.0
iPhone 7 Plus	1080 x 1920	414 x 736	2.608	3.0
iPhone 6s Plus	1080 x 1920	375 x 667	2.608	3.0
iPhone 6 Plus	1080 x 1920	375 x 667	2.608	3.0
iPhone 7	750 x 1334	375 x 667	2.0	2.0
iPhone 6s	750 x 1334	375 x 667	2.0	2.0
iPhone 6	750 x 1334	375 x 667	2.0	2.0
iPhone SE	640 x 1136	320 x 568	2.0	2.0
iPad Pro 12.9-inch (2nd generation)	2048 x 2732	1024 x 1366	2.0	2.0
iPad Pro 10.5-inch	2224 x 1668	1112 x 834	2.0	2.0
iPad Pro (12.9-inch)	2048 x 2732	1024 x 1366	2.0	2.0
iPad Pro (9.7-inch)	1536 x 2048	768 x 1024	2.0	2.0
iPad Air 2	1536 x 2048	768 x 1024	2.0	2.0
iPad Mini 4	1536 x 2048	768 x 1024	2.0	2.0

# ANDROID

Type	Device	Platform	Screen dimensions <small>in cm</small>	Aspect Ratio	Width × Height <small>dp</small>	Width × Height <small>px</small>	Density
■	Nexus 6	Android	6.0 in 2.9 × 5.2 in	16 : 9	411 × 731 dp	1440 × 2560 px	3.5 xxxhdpi
■	Nexus 6P	Android	5.7 in 2.8 × 5.0 in	16 : 9	411 × 731 dp	1440 × 2560 px	3.5 xxxhdpi
■	Nexus 7 ('12)	Android	7.0 in 3.7 × 5.9 in	16 : 10	600 × 960 dp	800 × 1280 px	1.3 tvdpi
■	Nexus 7 ('13)	Android	7.0 in 3.7 × 6.0 in	16 : 10	600 × 960 dp	1280 × 1920 px	2.0 xhdpi
■	Nexus 9	Android	8.9 in 7.1 × 5.3 in	4 : 3	1024 × 768 dp	2048 × 1536 px	2.0 xhdpi
■	Samsung Galaxy Note 4	Android	5.7 in 2.8 × 5.0 in	16 : 9	480 × 853 dp	1440 × 2560 px	3.0 xxhdpi
■	Samsung Galaxy S5	Android	5.1 in 2.9 × 5.6 in	16 : 9	360 × 640 dp	1080 × 1920 px	3.0 xxhdpi
■	Samsung Galaxy S6	Android	5.1 in 2.5 × 4.4 in	16 : 9	360 × 640 dp	1440 × 2560 px	4.0 xxxhdpi
■	Samsung Galaxy S7	Android	5.1 in 2.5 × 4.4 in	16 : 9	360 × 640 dp	1440 × 2560 px	4.0 xxxhdpi
■	Samsung Galaxy S7 Edge	Android	5.5 in 2.7 × 4.8 in	16 : 9	360 × 640 dp	1440 × 2560 px	4.0 xxxhdpi
■	Samsung Galaxy S8	Android	5.8 in 2.5 × 5.2 in	18.5 : 9	360 × 740 dp	1440 × 2960 px	4.0 xxxhdpi



**PANIC!!!**

# IPHONE

Device	Native Resolution (Pixels)
iPhone X	1125 x 2436
iPhone 8 Plus	1080 x 1920
iPhone 8	750 x 1334
iPhone 7 Plus	1080 x 1920
iPhone 6s Plus	1080 x 1920
iPhone 6 Plus	1080 x 1920
iPhone 7	750 x 1334
iPhone 6s	750 x 1334
iPhone 6	750 x 1334
iPhone SE	640 x 1136
iPad Pro 12.9-inch (2nd generation)	2048 x 2732
iPad Pro 10.5-inch	2224 x 1668
iPad Pro (12.9-inch)	2048 x 2732
iPad Pro (9.7-inch)	1536 x 2048
iPad Air 2	1536 x 2048
iPad Mini 4	1536 x 2048

# ANDROID

Type	Device	Width x Height px
Smartphone	Nexus 6	1440 x 2560 px
Smartphone	Nexus 6P	1440 x 2560 px
Smartphone	Nexus 7 ('12)	800 x 1280 px
Smartphone	Nexus 7 ('13)	1280 x 1920 px
Smartphone	Nexus 9	2048 x 1536 px
Smartphone	Samsung Galaxy Note 4	1440 x 2560 px
Smartphone	Samsung Galaxy S5	1080 x 1920 px
Smartphone	Samsung Galaxy S6	1440 x 2560 px
Smartphone	Samsung Galaxy S7	1440 x 2560 px
Smartphone	Samsung Galaxy S7 Edge	1440 x 2560 px
Smartphone	Samsung Galaxy S8	1440 x 2960 px

# IPHONE

Device	Native Resolution (Pixels)
iPhone X	1125 x 2436
iPhone 8 Plus	1080 x 1920
iPhone 8	750 x 1334
iPhone 7 Plus	1080 x 1920
iPhone 6s Plus	1080 x 1920
iPhone 6 Plus	1080 x 1920
iPhone 7	750 x 1334
iPhone 6s	750 x 1334
iPhone 6	750 x 1334
iPhone SE	640 x 1136
iPad Pro 12.9-inch (2nd generation)	2048 x 2732
iPad Pro 10.5-inch	2224 x 1668
iPad Pro (12.9-inch)	2048 x 2732
iPad Pro (9.7-inch)	1536 x 2048
iPad Air 2	1536 x 2048
iPad Mini 4	1536 x 2048

# ANDROID

Type	Device	Width x Height px
	Nexus 6	1440 x 2560 px
	Nexus 6P	1440 x 2560 px
	Nexus 7 ('12)	800 x 1280 px
	Nexus 7 ('13)	1280 x 1920 px
	Nexus 9	2048 x 1536 px
	Samsung Galaxy Note 4	1440 x 2560 px
	Samsung Galaxy S5	1080 x 1920 px
	Samsung Galaxy S6	1440 x 2560 px
	Samsung Galaxy S7	1440 x 2560 px
	Samsung Galaxy S7 Edge	1440 x 2560 px
	Samsung Galaxy S8	1440 x 2960 px

# RANGO DE EJEMPLO

360px

414px



min

max

# EN LA VIDA REAL

## DESKTOP

```
text {
  font-size: 16px;
  color: red;
  padding: 10px 20px;
}
```

## MOBILE

```
@media all and (max-width: 699px) and (min-width: 520px) {
  .text {
    font-size: 12px;
  }
}
```

# CURSO RECOMENDADO



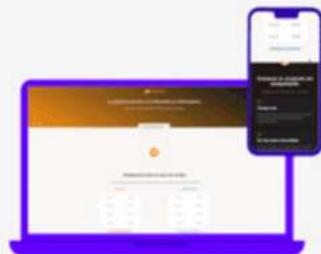
## Curso de Responsive Design: Maquetación Mobile First

Instruido por:  Diego De Granda

 Básico

 5 horas de contenido

[Ver la ruta de aprendizaje](#) ▾



[Proyecto del curso](#)

### Construye un sitio web totalmente responsive

Crearás el frontend de un sitio web partiendo de su wireframe, analizarás su arquitectura y construirás en código cada una de sus partes para que este se adapte a cualquier dispositivo de los usuarios.

# PALABRAS CLAVE



auto-fill



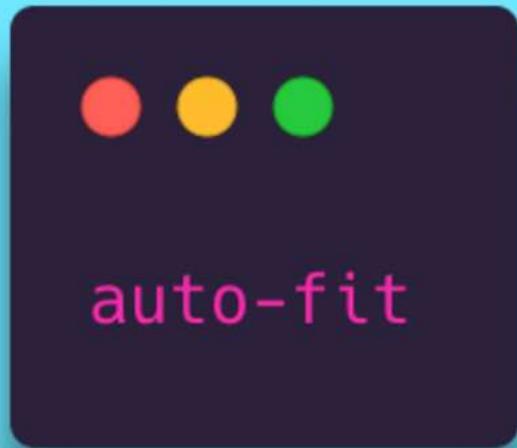
auto-fit





**LLENA LA FILA  
CON TANTAS  
COLUMNAS  
COMO PUEDA  
CABER.**

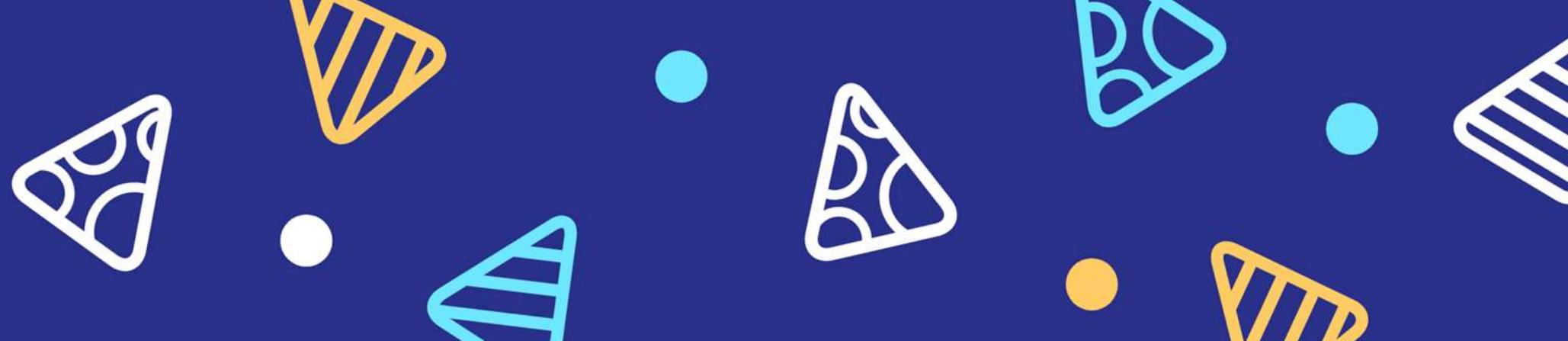
**POR LO TANTO,  
CREA COLUMNAS  
IMPLÍCITAS  
TRATANDO DE  
LLENAR LA FILA.**



**ENCAJA LAS  
COLUMNAS  
DISPONIBLES  
ACTUALMENTE EN  
EL ESPACIO  
EXPANDIÉNDOLAS  
PARA QUE  
OCUPEN  
CUALQUIER  
ESPACIO  
DISPONIBLE**



```
.grid-container--fill {  
  grid-template-columns: repeat(auto-fill, minmax(100px, 1fr));  
}  
  
.grid-container--fit {  
  grid-template-columns: repeat(auto-fit, minmax(100px, 1fr));  
}
```



auto-fill

1	2	3	4	5	6	7
---	---	---	---	---	---	---

---

auto-fit

1	2	3	4	5	6	7
---	---	---	---	---	---	---



# RETO

HACER EL DISEÑO  
RESPONSIVO DE  
NUESTRO  
PROYECTO !!!



**¡MANOS  
AL CÓDIGO!**

DISEÑO  
RESPONSIVO



¿Vendrá algo más para  
esta especificación?

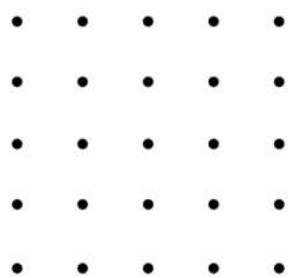


**Curso  
siguiente**

Diseño web  
con CSS Grid  
y Flexbox



# **Libros que te recomiendo**





## BRIEF BOOKS FOR PEOPLE WHO DESIGN, WRITE, AND CODE

We shed light on the industry's most essential topics in a format that's practical and fun.

NEW  
RELEASE



<https://abookapart.com/products/presenting-design-work>

Donna Spencer  
**PRESENTING DESIGN  
WORK**

# STARTER PACKS

[ALL PACKS →](#)



## COLLABORATIVE DESIGN

Work better, together.



## COMPASSIONATE DESIGN

Create kinder, more connected design.



## DESIGN PROCESS

Your design process, advanced.



## RESPONSIVE DESIGN

Get responsive with every project.



## IMAGES AND ANIMATION

Make your site look great and move swiftly.



## CONTENT STRATEGY

Corral your content strategy.



## DESIGN BUSINESS

Manage, sell, and buy design like a pro.



## FRONT-END FUNDAMENTALS

Essential languages, and then some.



PAPERBACK  
\$24.00 + shipping

ADD

EBOOK ?  
\$14.00

ADD

PAPERBACK & EBOOK  
\$34.20 + shipping

ADD

CSS Grid Layout will transform the way you design and develop for the web—and Rachel Andrew will change the way you grok the spec. Learn to use Grid Layout within a system that includes existing methods to perform the



READ A SAMPLE CHAPTER

#### CONTENTS

- Where We Came From

# Mis redes



**ESTEFANY AGUILAR**

@teffcode   