



# How CSS evolves, a look back at 2 years of CSS grid

Chen Hui Jing / @hj\_chen







Surname

First name

陈  
Chen

慧 晶  
Hui Jing



@hj\_chen







🥑 Developer Advocate 🥑

**nexmo**<sup>®</sup>  
The Vonage<sup>®</sup> API Platform



## WORLD WIDE WEB

The WorldWideWeb (W3) is a wide-area hypermedia[1] information retrieval initiative aiming to give universal access to a large universe of documents.

Everything there is online about W3 is linked directly or indirectly to this document, including an executive summary[2] of the project, Mailing lists[3] , Policy[4] , November's W3 news[5] , Frequently Asked Questions[6] .

What's out there?[7]Pointers to the world's online information,  
subjects[8] , W3 servers[9], etc.

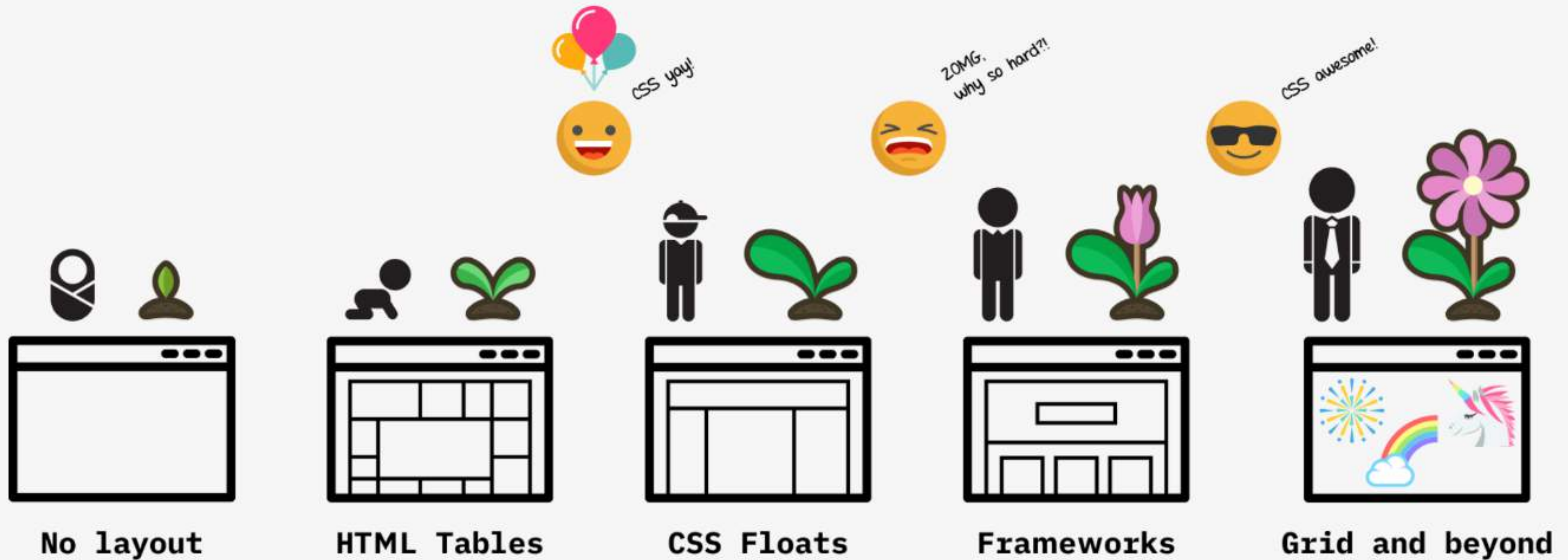
Help[10] on the browser you are using

Software  
Products[11] A list of W3 project components and their current  
state. (e.g. Line Mode[12] ,X11 Viola[13] ,  
NeXTStep[14] , Servers[15] , Tools[16] , Mail  
robot[17] , Library[18] )

Technical[19] Details of protocols, formats, program internals  
etc



# Web layouts over the years





# Evolution of CSS Specifications

**CSS1**  
Recommendation:  
17 Dec 1996

**CSS2**  
Recommendation:  
12 May 1998

**CSS2.1**  
Recommendation:  
7 Jun 2011

**CSS2.2**  
Working draft:  
12 Apr 2016

**CSS3**  
Decision to modularise: 14 Apr 2000  
(26 modules)

**CSS Snapshot 2018**  
(99 modules)

**Completed**  
CSS Snapshot 2018  
CSS Snapshot 2017  
CSS Snapshot 2015  
CSS Snapshot 2010  
CSS Snapshot 2007  
CSS Color Level 3  
CSS Namespaces  
Selectors Level 3  
CSS Level 2 Revision 1  
Media Queries  
CSS Style Attributes  
CSS Fonts Level 3  
CSS Basic User  
Interface Level 3

**Stable**  
CSS Backgrounds and Borders Level 3  
CSS Conditional Rules Level 3  
CSS Multi-column Layout Level 1  
CSS Values and Units Level 3  
CSS Cascading and Inheritance Level 3  
CSS Fonts Level 3  
CSS Writing Modes Level 3  
CSS Counter Styles Level 3  
CSS Containment Level 1

**Rewriting**  
CSS Generated Content Level 3

**Revising**  
CSS Paged Media Level 3  
CSSOM View  
CSS Intrinsic & Extrinsic Sizing Level 3  
CSS Ruby Level 1  
CSS Overflow Level 3  
CSS Box Model Level 3  
CSS Pseudo-Elements Level 4  
CSS Scrollbars Level 1

**Refining**  
CSS Animations Level 1  
Web Animations  
CSS Text Level 3  
CSS Transforms Level 1  
CSS Transitions  
CSS Box Alignment Level 3  
Selectors Level 4  
CSS Lists Level 3  
Motion Path Level 1  
Preview of CSS Level 2  
CSS Fonts Level 4  
CSS Easing Functions Level 1  
CSS Logical Properties and Values  
Level 1

**Testing**  
CSS Images Level 3  
CSS Speech  
CSS Text Decoration Level 3  
CSS Shapes Level 1  
CSS Masking Level 1  
CSS Fragmentation Level 3  
CSS Cascading Variables  
Compositing and Blending Level 1  
CSS Syntax Level 3  
CSS Grid Layout Level 1  
CSS Display Level 3  
CSS Will Change Level 1  
Media Queries Level 4  
Geometry Interfaces Level 1  
CSS Cascading and Inheritance Level 4  
CSS Scroll Snap Level 1  
CSS Painting API Level 1  
CSS Writing Modes Level 4

**Exploring**  
CSS Backgrounds and Borders Level 4  
CSS Device Adaptation  
CSS Exclusions  
Filter Effects Level 1  
CSS Generated Content for Paged Media  
CSS Page Floats  
CSS Template Layout  
CSS Line Grid  
CSS Positioned Layout Level 3  
CSS Regions  
CSS Table Level 3  
CSS Object Model  
CSS Font Loading  
CSS Scoping Level 1  
CSS Inline Layout Level 3  
CSS Round Display Level 1  
CSS Basic User Interface Level 4  
CSS Text Level 4  
CSS Properties and Values API Level 1

CSS Typed OM Level 1  
Worklets Level 1  
CSS Color Level 4  
CSS Rhythmic Sizing Level 1  
CSS Image Values and Replaced  
Content Level 4  
CSS Fill and Stroke Level 3  
CSS Overflow Level 4  
CSS Grid Layout Level 2  
CSS Text Decoration Level 4  
CSS Layout API Level 1  
CSS Values and Units Level 4  
CSS Shadow Parts  
CSS Fragmentation Level 4  
CSS Spatial Navigation Level 1  
CSS Color Adjustment Level 1  
CSS Overscroll Behavior Level 1  
CSS Animation Worklet API  
CSS Containment Level 2





# CSS Snapshot 2018

## (99 modules)

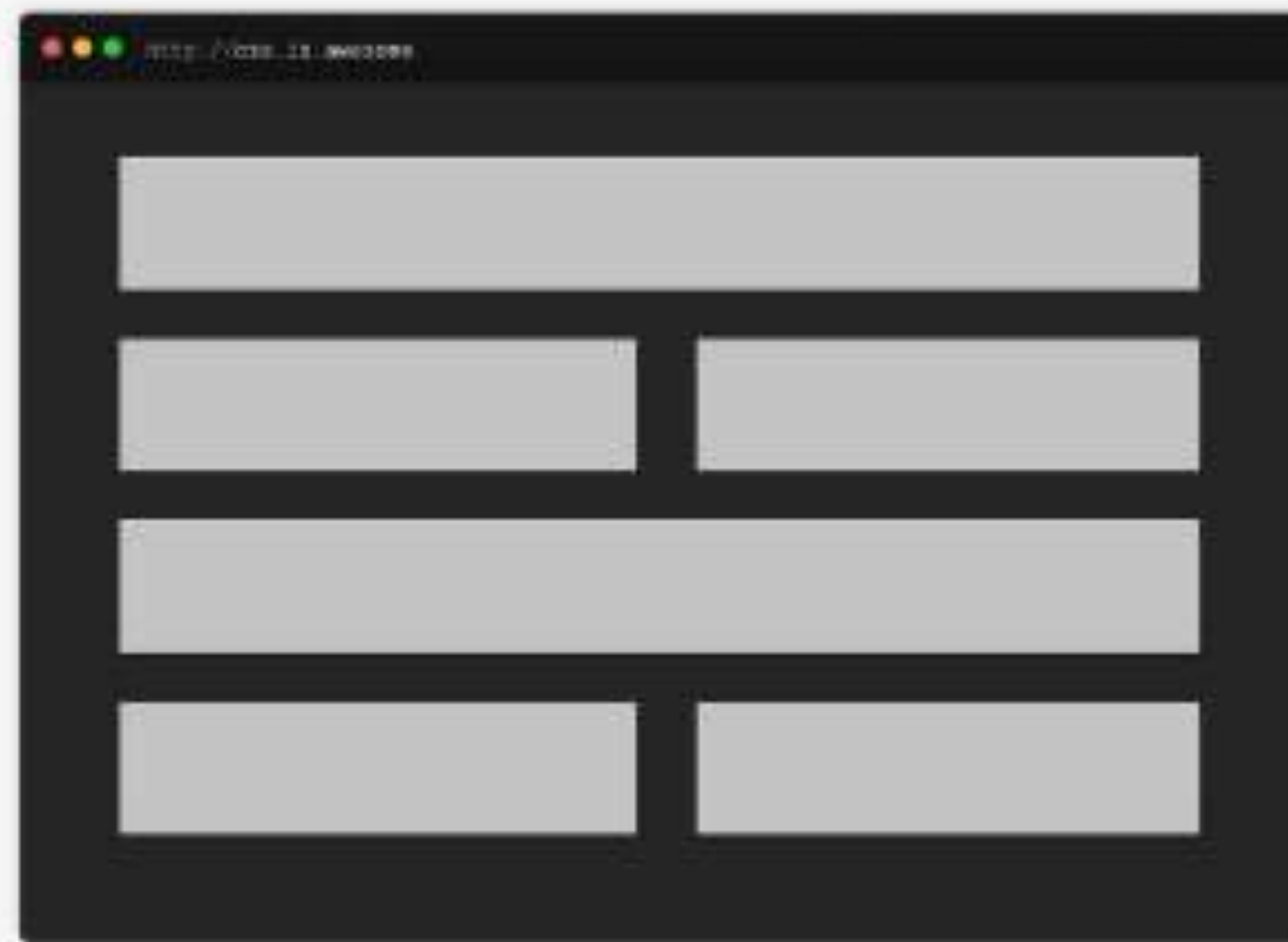
<b>Completed</b>	<b>Stable</b>	<b>Exploring</b>	
CSS Snapshot 2018	CSS Backgrounds and Borders Level 3	CSS Backgrounds and Borders Level 4	CSS Typed OM Level 1
CSS Snapshot 2017	CSS Conditional Rules Level 3	CSS Device Adaptation	Worklets Level 1
CSS Snapshot 2015	<a href="#">CSS Multi-column Layout Level 1</a>	CSS Exclusions	CSS Color Level 4
CSS Snapshot 2010	CSS Values and Units Level 3	Filter Effects Level 1	CSS Rhythmic Sizing Level 1
CSS Snapshot 2007	CSS Cascading and Inheritance Level 3	CSS Generated Content for Paged Media	CSS Image Values and Replaced Content Level 4
CSS Color Level 3	CSS Fonts Level 3	<a href="#">CSS Page Floats</a>	CSS Fill and Stroke Level 3
CSS Namespaces	<a href="#">CSS Writing Modes Level 3</a>	<a href="#">CSS Template Layout</a>	CSS Overflow Level 4
Selectors Level 3	CSS Counter Styles Level 3	<a href="#">CSS Line Grid</a>	<a href="#">CSS Grid Layout Level 2</a>
CSS Level 2 Revision 1	CSS Containment Level 1	<a href="#">CSS Positioned Layout Level 3</a>	CSS Text Decoration Level 4
Media Queries	<b>Rewriting</b>	<a href="#">CSS Regions</a>	CSS Layout API Level 1
CSS Style Attributes	CSS Generated Content Level 3	<a href="#">CSS Table Level 3</a>	CSS Values and Units Level 4
CSS Fonts Level 3	<b>Refining</b>	CSS Object Model	CSS Shadow Parts
CSS Basic User Interface Level 3	CSS Animations Level 1	CSS Font Loading	<a href="#">CSS Fragmentation Level 4</a>
	Web Animations	CSS Scoping Level 1	CSS Spatial Navigation Level 1
<b>Revising</b>	CSS Text Level 3	<a href="#">CSS Inline Layout Level 3</a>	CSS Color Adjustment Level 1
CSS Paged Media Level 3	CSS Transforms Level 1	<a href="#">CSS Round Display Level 1</a>	CSS Overscroll Behavior Level 1
CSSOM View	CSS Transitions	CSS Basic User Interface Level 4	CSS Animation Worklet API
<a href="#">CSS Intrinsic &amp; Extrinsic Sizing Level 3</a>	<a href="#">CSS Box Alignment Level 3</a>	CSS Text Level 4	CSS Containment Level 2
<a href="#">CSS Ruby Level 1</a>	Selectors Level 4	CSS Properties and Values API Level 1	
CSS Overflow Level 3	CSS Lists Level 3	<b>Testing</b>	<a href="#">CSS Grid Layout Level 1</a>
<a href="#">CSS Box Model Level 3</a>	Motion Path Level 1	CSS Images Level 3	<a href="#">CSS Display Level 3</a>
CSS Pseudo-Elements Level 4	Preview of CSS Level 2	CSS Speech	CSS Will Change Level 1
CSS Scrollbars Level 1	CSS Fonts Level 4	CSS Text Decoration Level 3	Media Queries Level 4
	CSS Easing Functions Level 1	<a href="#">CSS Shapes Level 1</a>	Geometry Interfaces Level 1
	CSS Logical Properties and Values Level 1	<a href="#">CSS Masking Level 1</a>	CSS Cascading and Inheritance Level 4
		<a href="#">CSS Fragmentation Level 3</a>	CSS Scroll Snap Level 1
		CSS Cascading Variables	CSS Painting API Level 1
		Compositing and Blending Level 1	<a href="#">CSS Writing Modes Level 4</a>
		CSS Syntax Level 3	





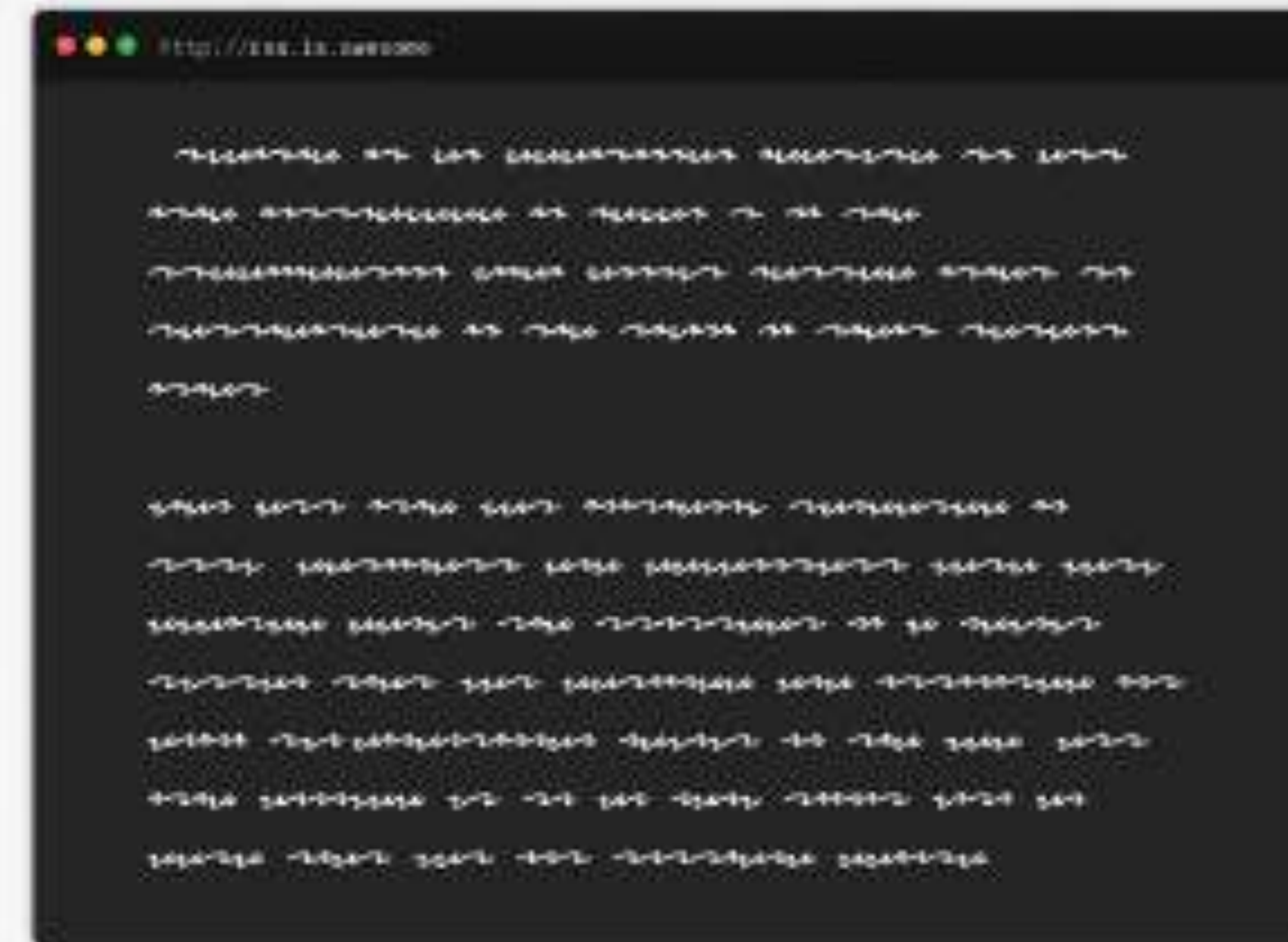
## Block layout

designed for laying out documents



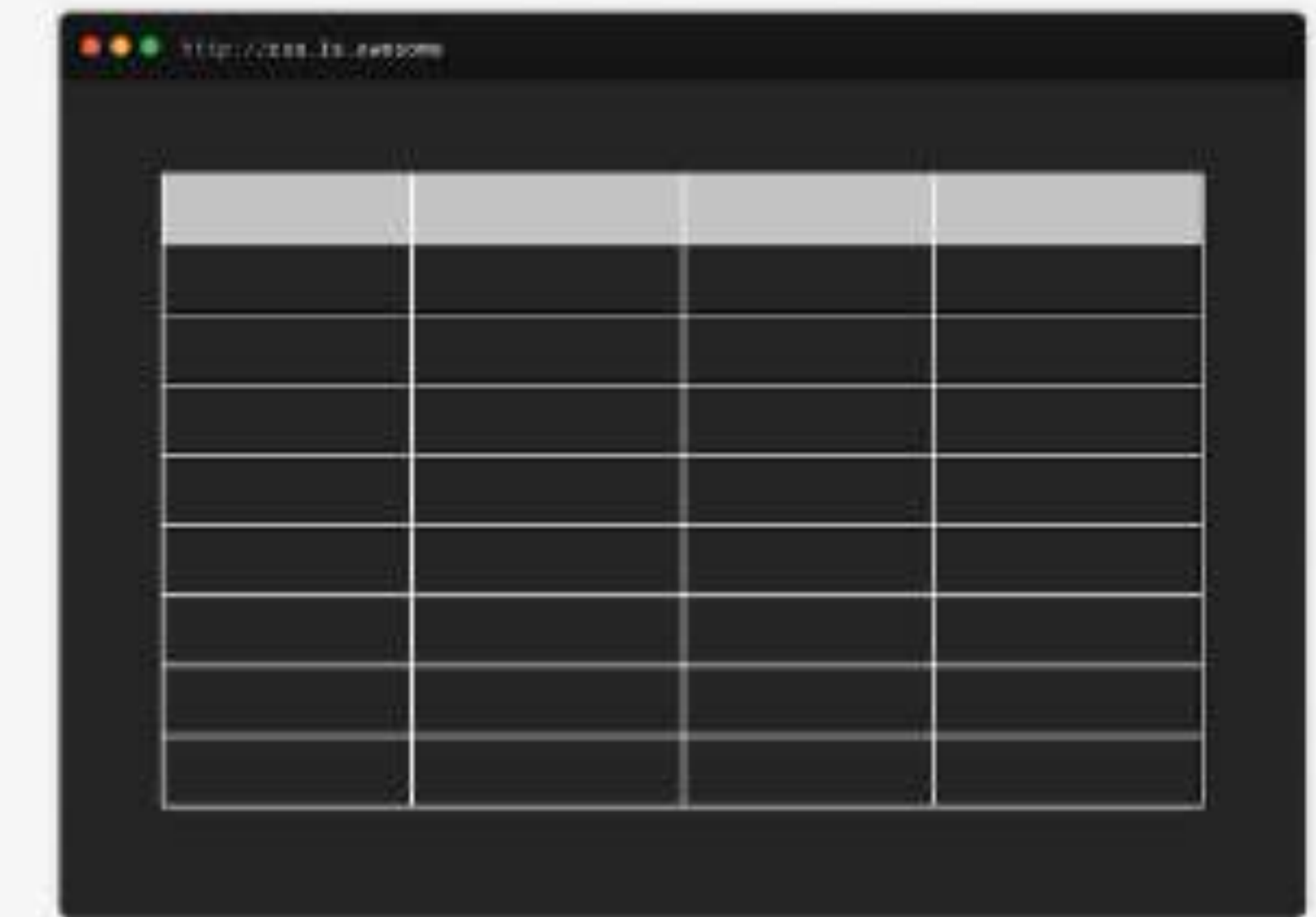
## Inline layout

designed for laying out text



## Table layout

designed for laying out 2D data in a tabular format



## Positioned layout

designed for very explicit positioning without much regard for other elements in the document





# Flexible box layout

describes a CSS box model optimised for **user interface design** and designed for laying out more **complex applications and webpages**





# Grid layout

Defines a **two-dimensional** grid-based layout system, optimised for **user interface design**.





# Why is CSS grid special?

“Grid works from the **container** in,  
other layout methods start with the **item**,”

Rachel Andrew





# Layout technique: inline-block

Item A	Item B	Item C
Item D	Item E	Item F

```
.inlineblock__item {  
  display: inline-block;  
  width: calc(100% / 3);  
}
```





# Layout technique: float

Item A	Item B	Item C
Item D	Item E	Item F

```
.float__item {  
  float: left;  
  width: calc(100% / 3);  
}
```





# Layout technique: flex

Item A	Item B	Item C
Item D	Item E	Item F

```
.flexbox {  
  display: flex;  
  flex-wrap: wrap;  
}  
  
.flexbox__item {  
  flex: 0 0 calc(100% / 3);  
}
```





“ Grid is the only layout technique that establishes a **relationship** between rows and columns of grid items. ”





“ Grid is the  
**relationships**



es a  
d items.

”





Grid versus Box?



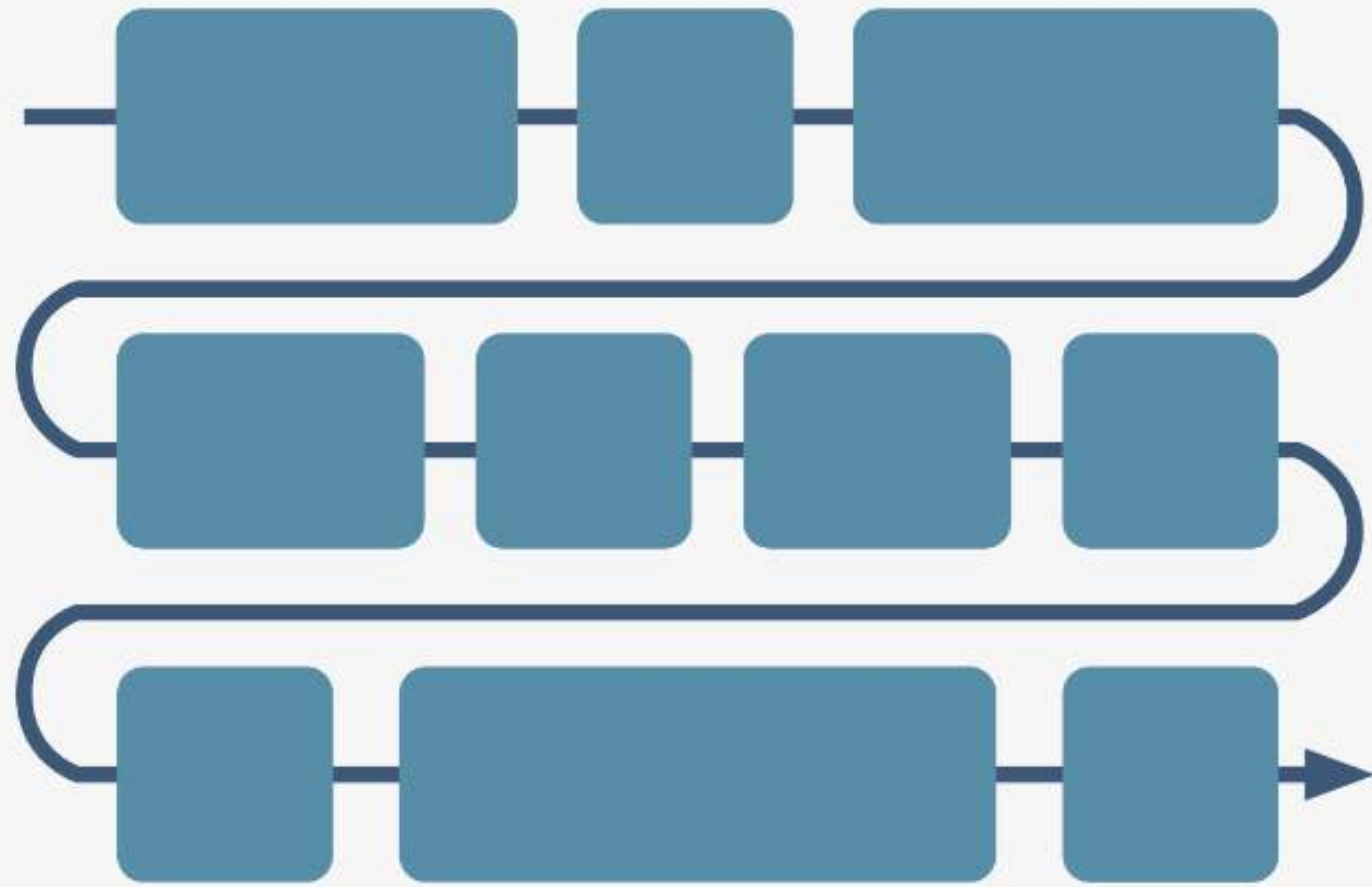


# Grid AND Flexbox

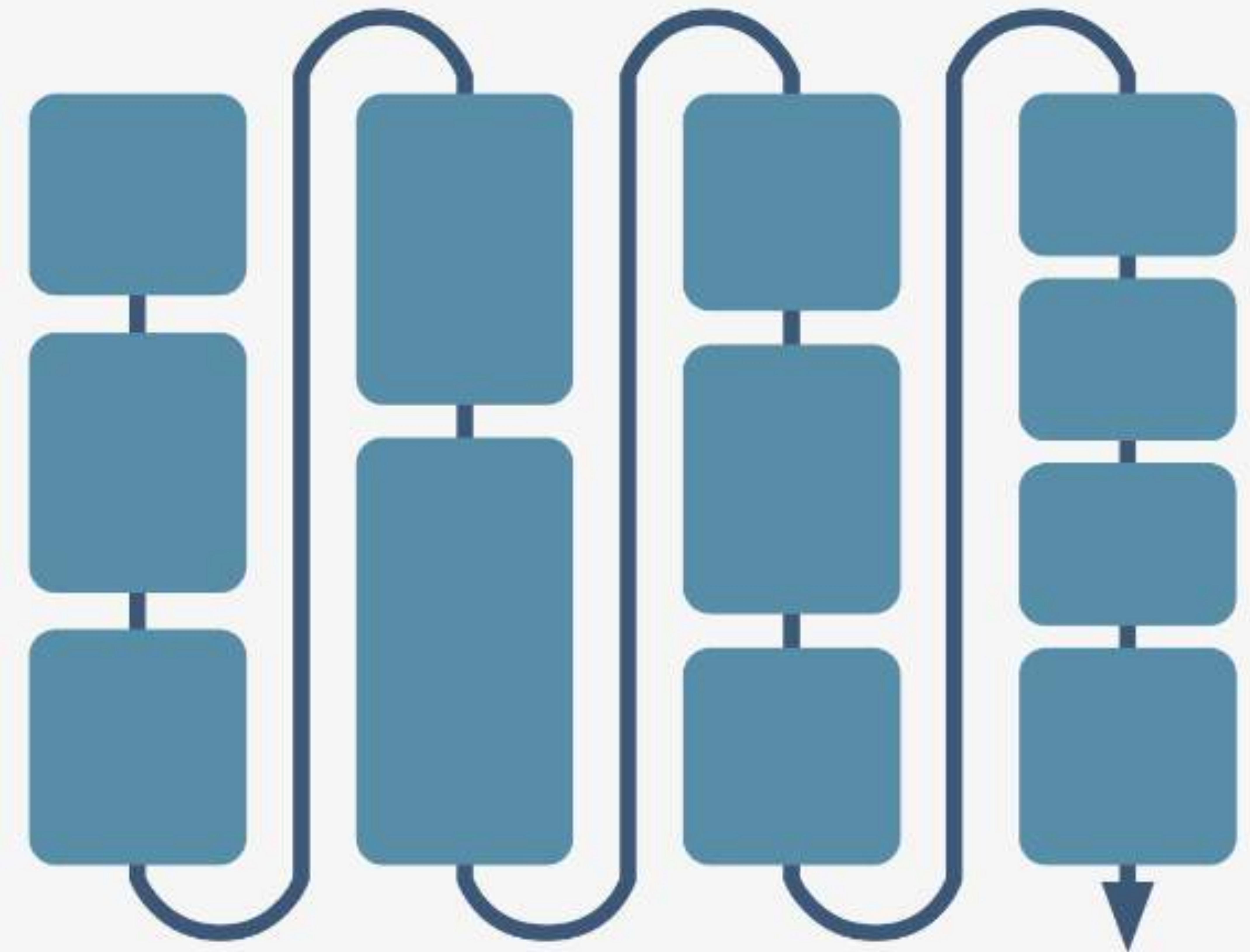




# Flexbox → single dimension



*flex-direction: row*

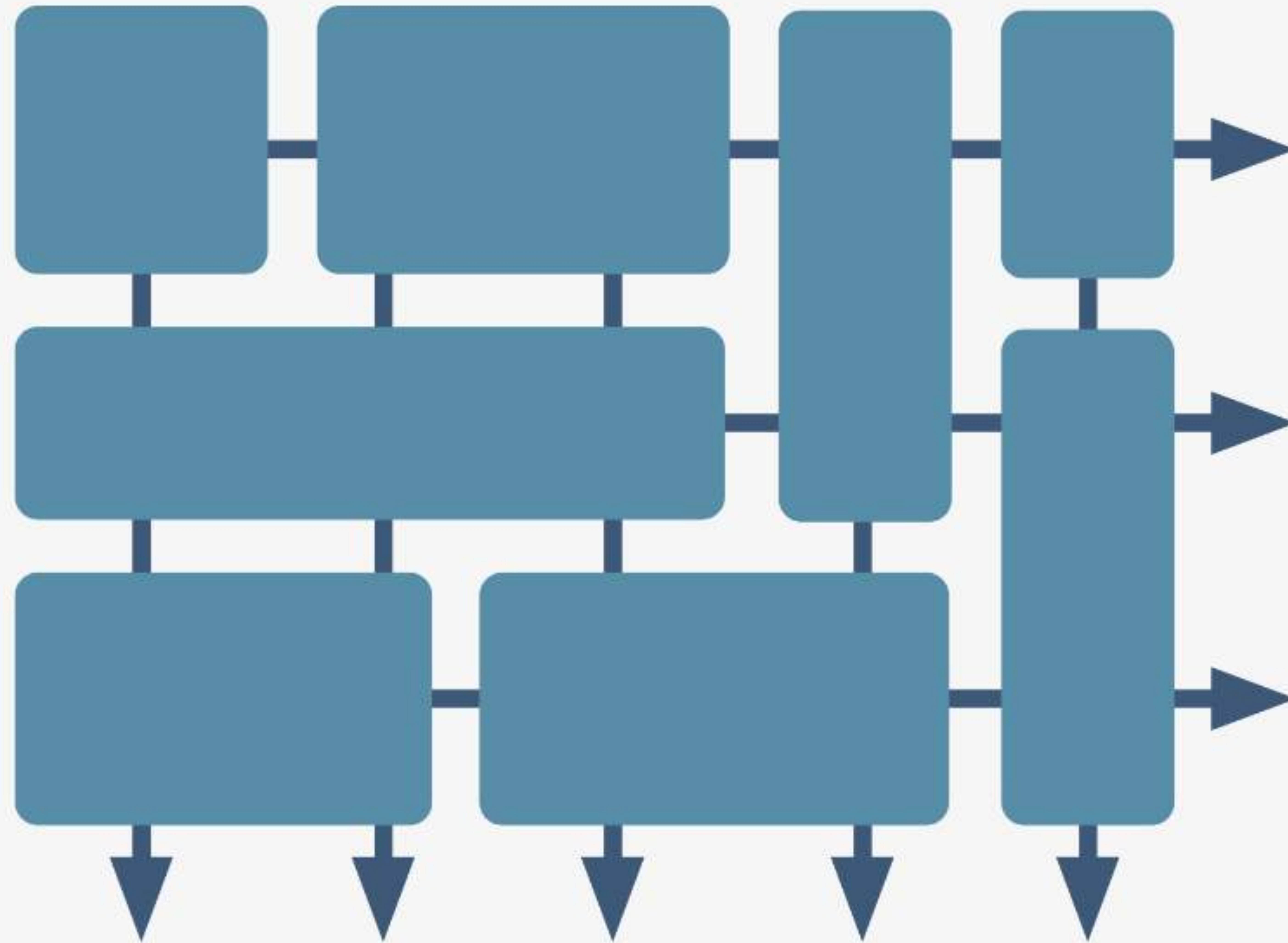


*flex-direction: column*





# Grid → two dimensions



*Items know about row and column sizing*





# Where does CSS come from?

- Browser vendors
- Companies who need certain features
- CSS working group





# The CSS Working Group





# Specification document statuses

## Working Draft (WD)

to create a snapshot of the specification's current state and to solicit input from the W3C and the public

## Last Call Working Draft (LC or LCWD)

to announce that the specification will move toward Candidate Recommendation unless significant issues are brought up

## Candidate Recommendation (CR)

to express that all known issues have been resolved, and is a call for implementations

## Proposed Recommendation (PR)

needs a comprehensive test suite and implementation reports proving that every feature is interoperably implemented in at least two shipping implementations

## Recommendation (REC)

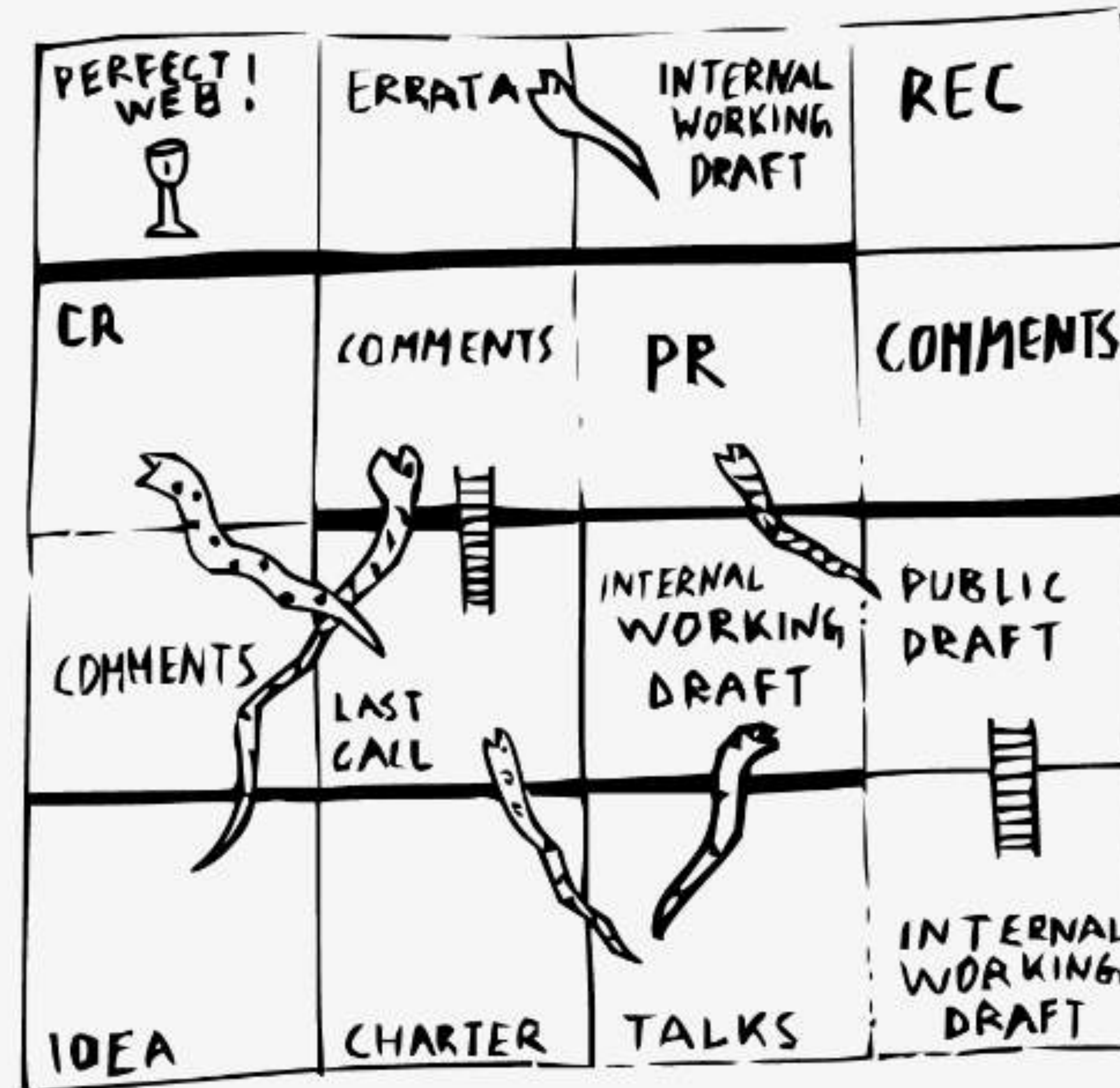
finally, a standard





# It's not a linear process...

W3C SNAKES AND LADDERS WD





“...it shouldn't come as a shock that grid-based layouts have been a goal of CSS since the beginning.”

The Story of CSS Grid, from Its Creators by Aaron Gustafson





# Fun facts about CSS grid

- Numerous proposals submitted that were never implemented
- First proposal that stuck was one from **Microsoft** (for Metro UI)
- First implementation of Grid ever was in **IE10** (with vendor prefix)
- **Bloomberg** hired **Igalia** to implement CSS grid for **Blink** and **WebKit**
- **Rachel Andrew** proposed gap to solve the gutter problem





# March 2017

Sun	Mon	Tue	Wed	Thu	Fri	Sat
			1	2	3	4
5	6		8	9	10	11
12	13		15	16	17	18
19	20		22	23	24	25
26		28	29	30	31	

# October 2017

Sun	Mon	Tue	Wed	Thu	Fri	Sat
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16		18	19	20	21
22	23	24	25	26	27	28
29	30					



# Browser support for Grid (Level 1)

## CSS Grid Layout (level 1) [↗](#)

Method of using a grid concept to lay out content, providing a mechanism for authors to divide available space for layout into columns and rows using a set of predictable sizing behaviors. Includes support for all `grid-\*` properties and the `fr` unit.

IE	Edge	Firefox	Chrome	Safari	iOS Safari	Opera Mini	Chrome for Android	Android Browser	Samsung Internet
9	16	69	76	12	12.4			4.4	8.2
10	17	70	77	12.1	13.1			4.4.4	9.2
11	18	71	78	13	13.2	all	78	76	10.1
	76	72	79	TP	13.3				

✓ ✗ Partial Support Prefixed

Global: 92.19% + 1.69% = 93.88%

Data from [caniuse.com](https://caniuse.com) | Embed from [caniuse.bitsofco.de](https://caniuse.bitsofco.de)

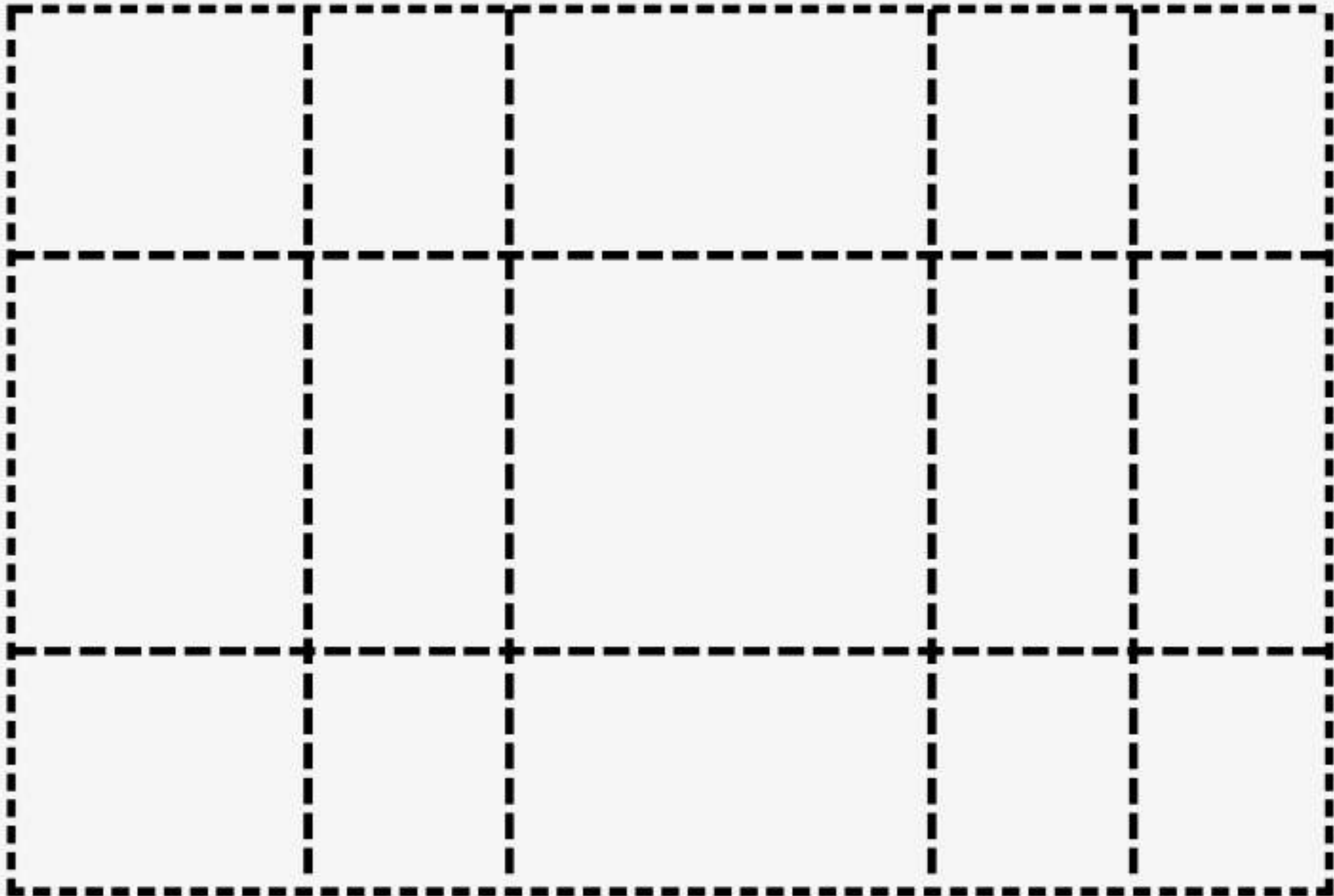
[Disable accessible colours](#)



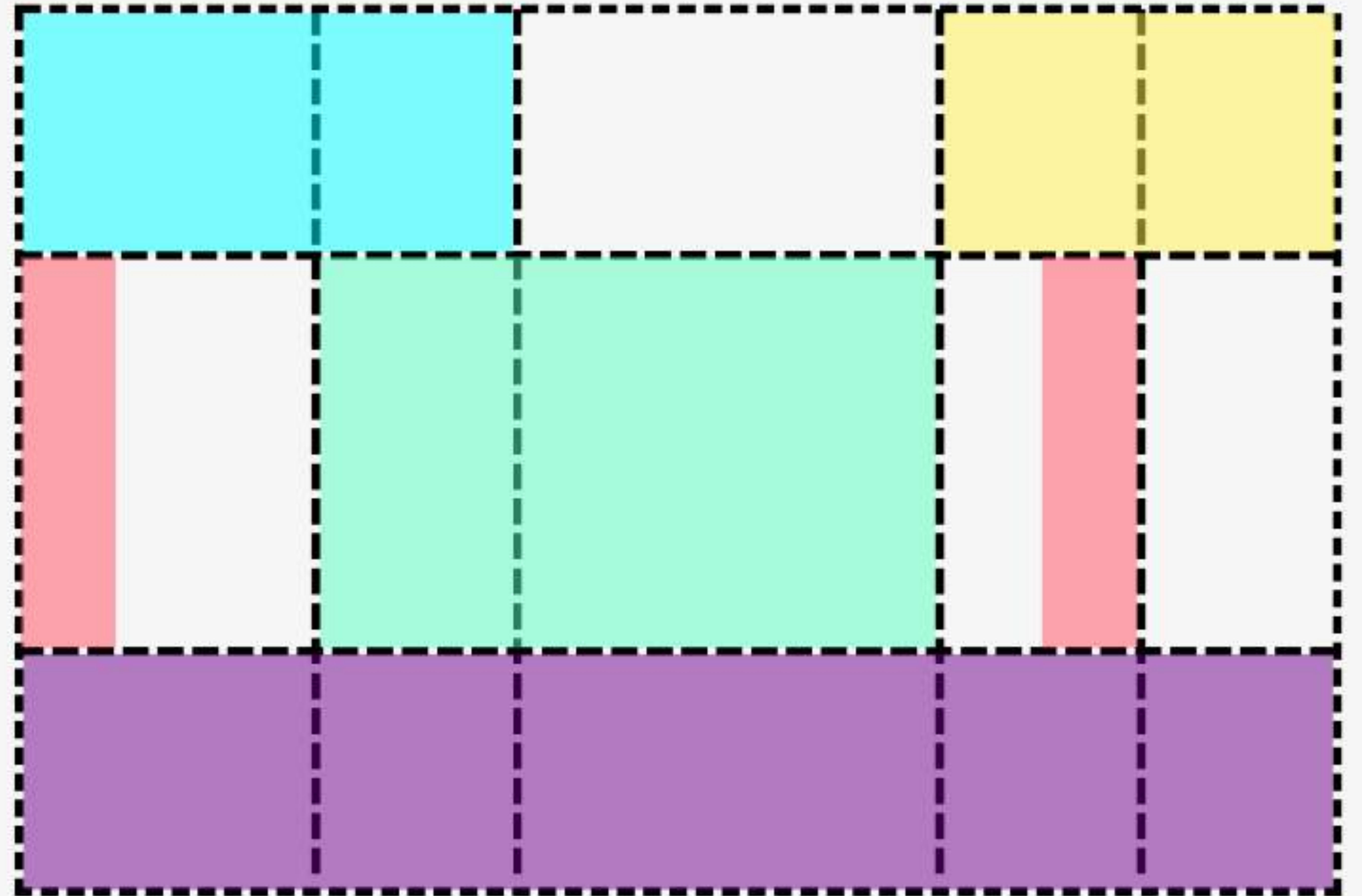


# Basics of CSS grid

Define your grid.

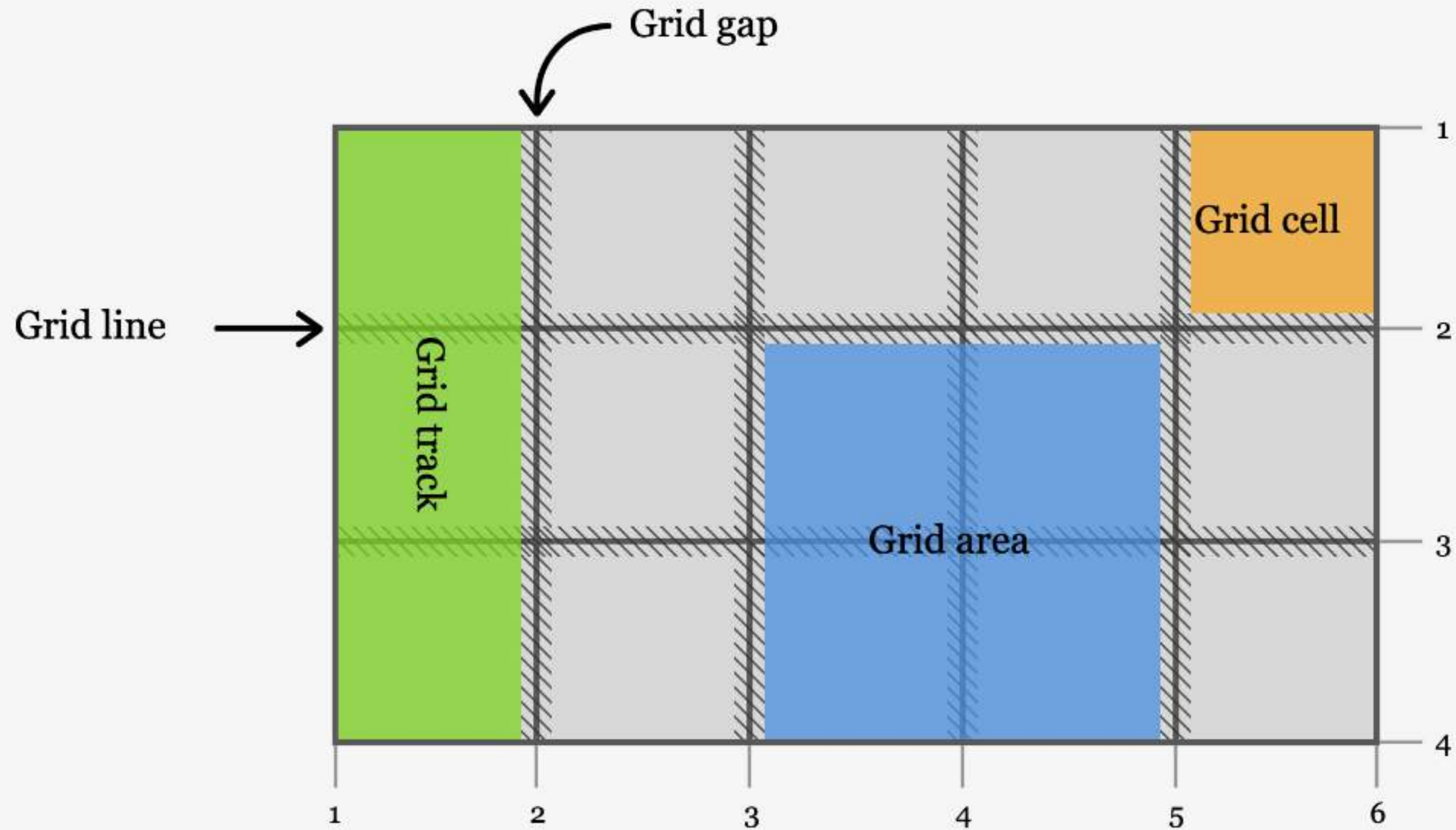


Place items in the grid.





# Grid terminology





# Using DevTools to learn Grid

Live examples





# What's in Grid Level 2?

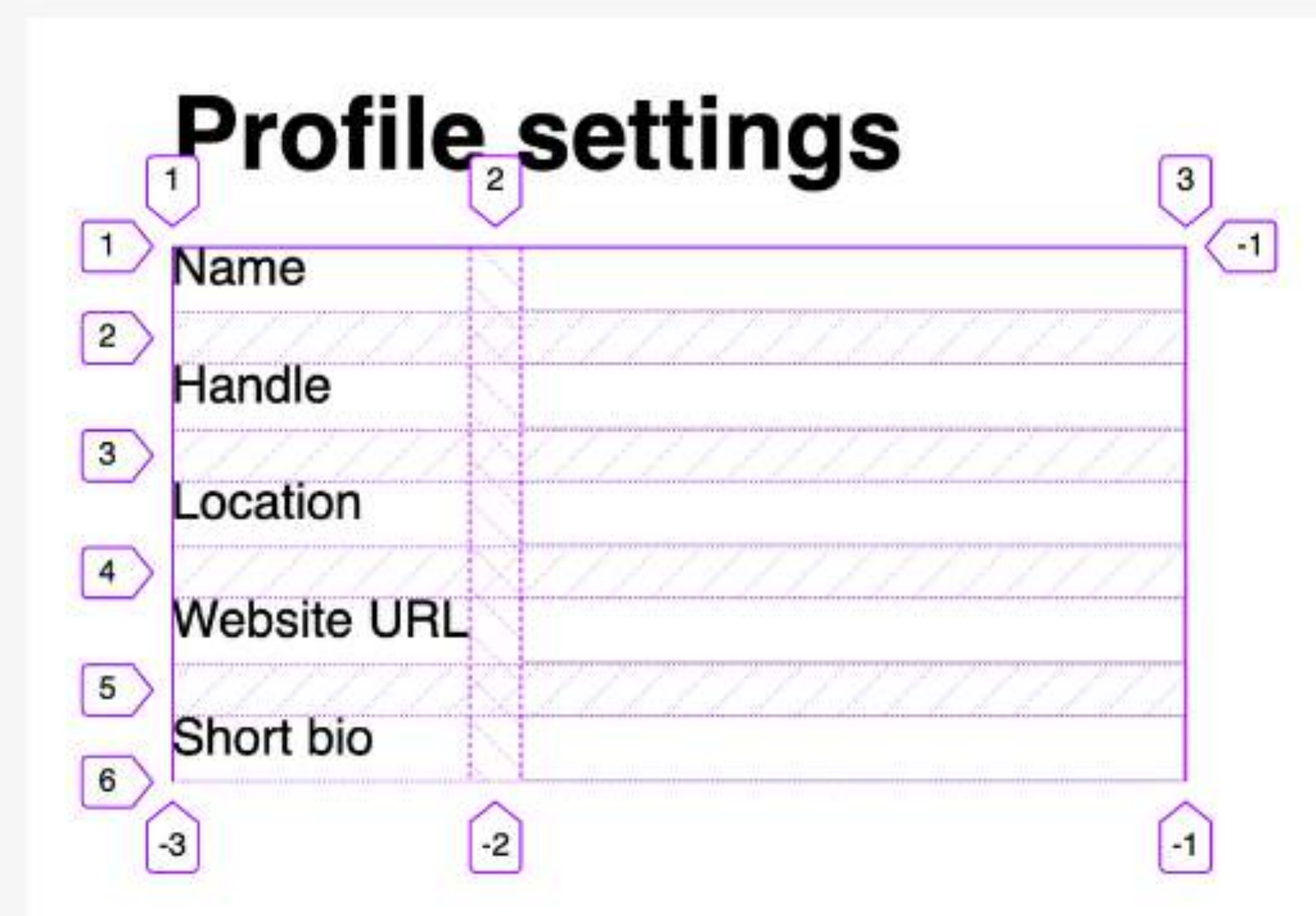
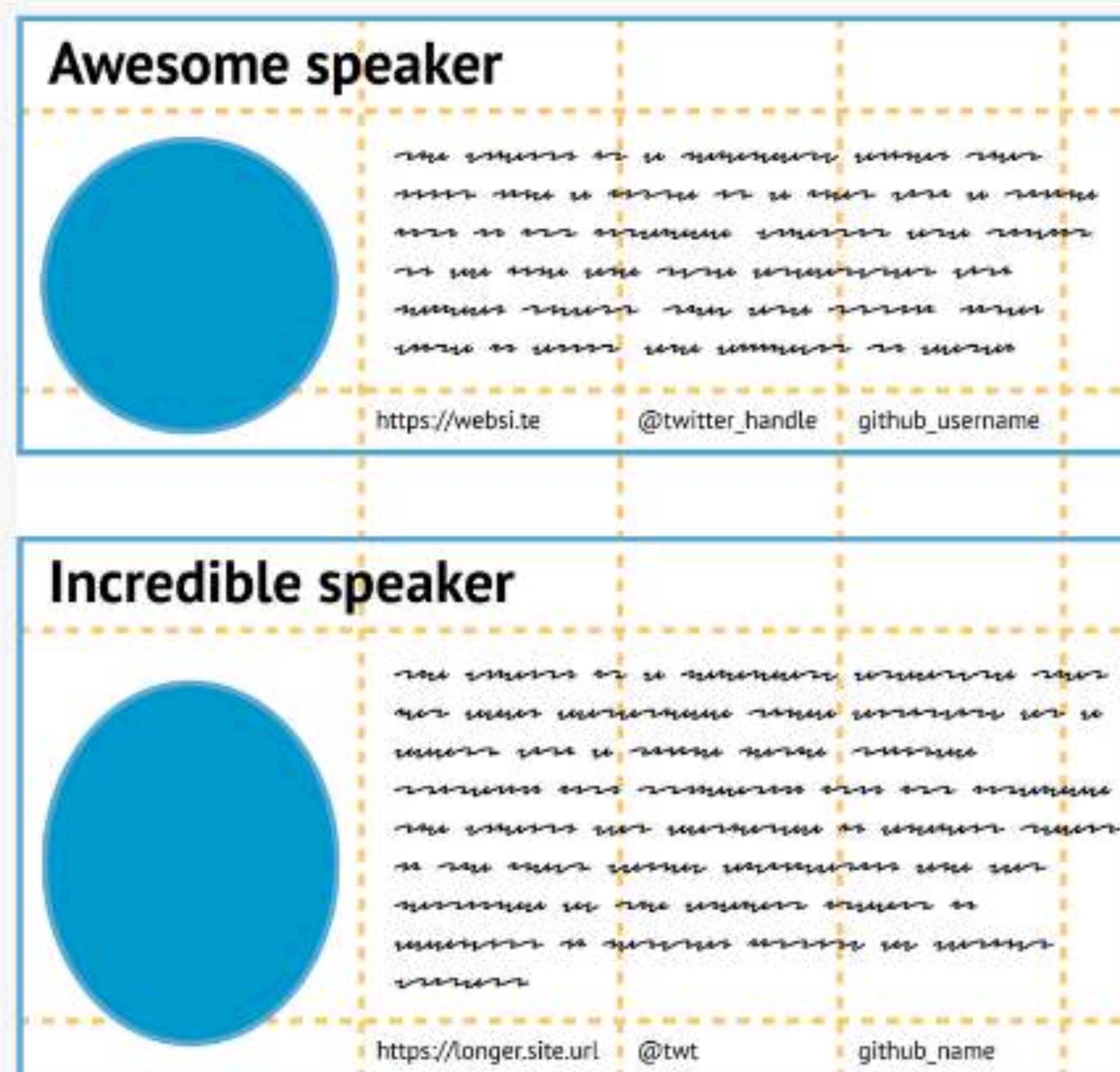
Level 2 expands Grid by:

- adding **“subgrid” capabilities** for nested grids to participate in the sizing of their parent grids
- **aspect-ratio-controlled gutters**





# Why do we need subgrid?





# The subgrid syntax

Same properties, more values

```
grid-template-columns: subgrid <line-name-list>?  
grid-template-rows: subgrid <line-name-list>?
```

The subgrid keyword indicates to the browser the nested grid will use the same sizing as its parent along the relevant axis.

```
.subgrid-container {  
  grid-columns: 2 / 5; /* placement for the subgrid container itself */  
  
  display: grid; /* you must still apply a display: grid to the subgrid */  
  grid-template-columns: subgrid;  
  grid-template-rows: subgrid;  
}
```





# Using DevTools to learn Subgrid

Live examples





# Browser support for subgrid

## CSS Subgrid [↗](#)

Feature of the CSS Grid Layout Module Level 2 that allows a grid-item with its own grid to align in one or both dimensions with its parent grid.

IE	Edge	Firefox	Chrome	Safari	iOS Safari	Opera Mini	Chrome for Android	Android Browser	Samsung Internet
9	16	69	76	12	12.4			4.4	8.2
10	17	70	77	12.1	13.1			4.4.4	9.2
11	18	71	78	13	13.2	all	78	76	10.1
	76	72	79	TP	13.3				

✓ x Partial Support

Global: 0.08% + 0% = 0.08%

Data from [caniuse.com](https://caniuse.com) | Embed from [caniuse.bitsofco.de](https://caniuse.bitsofco.de)

[Disable accessible colours](#)





# CSS



"WANTS  
**YOU**"

TO HELP

# MAKE THE WEB BETTER



# References

- [CSS Flexible Box Layout Module Level 1](#)
- [CSS Grid Layout Module Level 1](#)
- [CSS Grid Layout Module Level 2](#)
- [Codrops reference: Subgrid](#)
- [Subgrid on MDN](#)
- [Grid Level 2 and Subgrid](#)
- [Why display: contents is not CSS Grid Layout subgrid](#)
- [CSS Grid Level 2: Here Comes Subgrid](#)
- [Grid by Example](#)
- [Researching a Property in the CSS Specifications](#)



# Thank you!



<https://www.chenhuijing.com>



@hj\_chen



@hj\_chen



@huijing

Font is Red Hat Display by Jeremy Mickel.

