

# GET ON THE GRID

Using CSS Grid in the Real World

Brenda Storer

@brendamarienyc

Generate NYC – April 26, 2018



Me

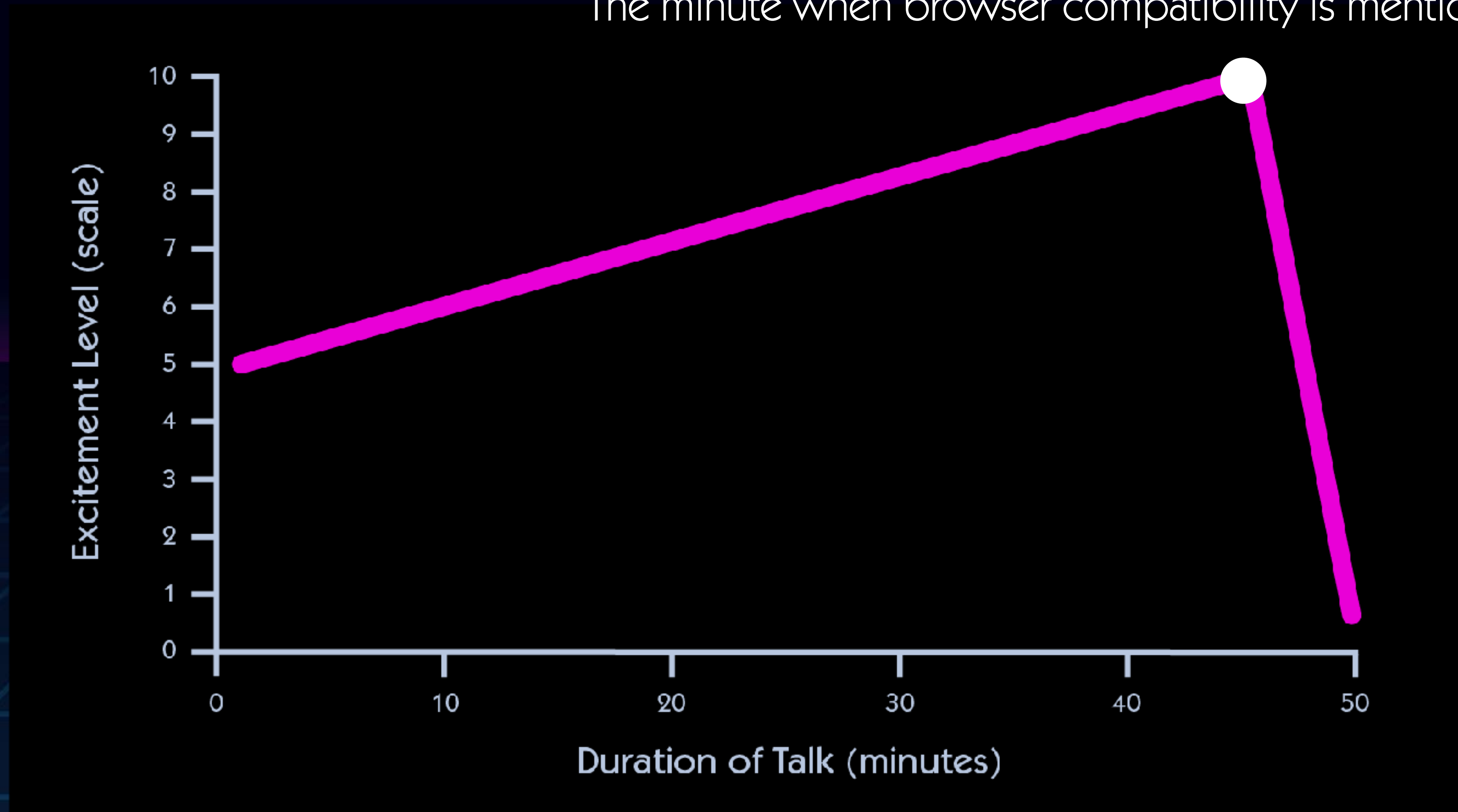
@brendamarienyc

Designer at thoughtbot (Front-End Dev too!)

You?

# Emotional Reaction to new CSS Specs

The minute when browser compatibility is mentioned 🤔😓😡





Why can't we have nice things? 🥹



GRID IS NICE



# Why?

Bye Bye Vendor Prefixes 😨

Hello Feature Flags 😊

# January 27, 2017

#

## CSS Grid Layout - CR

Global

0.04% + 5.86% = 5.9%

unprefixed:

0.04%

Method of using a grid concept to lay out content, providing a mechanism for authors to divide available space for lay out into columns and rows using a set of predictable sizing behaviors

Current aligned

Usage relative

Date relative

Show all

IE	Edge *	Firefox	Chrome	Safari	Opera	iOS Safari *	Opera Mini *	Android Browser *	Chrome for Android
			1 49						
			1 51					4.3	
			1 54					4.4	
		3 50	1 55			9.3		4.4.4	
2 11	2 14	3 51	1 56	10	1 42	10.2	all	53	55
	2 15	52	57	10.1	43				
		53	58	TP	44				
		54	59						




# Today

#

## CSS Grid Layout - CR

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.

Usage

% of all users 

Global

84.02% + 3.38% = 87.4%

unprefixed:

84.02%

Current aligned

Usage relative

Date relative

Show all

IE	Edge *	Firefox	Chrome	Safari	iOS Safari *	Opera Mini *	Chrome for Android	UC Browser for Android	Samsung Internet
			<div>149</div>						
			63		10.3				
		58	64	11	11.2				4
<div>211</div>	16	59	65	11.1	11.3	all	64	11.8	6.2
	17	60	66	TP					
	18	61	67						
			68						

...with one exception. IE.

Prefixed version of Grid in IE only `-ms`

Early, not fully baked version of Grid







# Step by Step Guide to writing CSS Grid

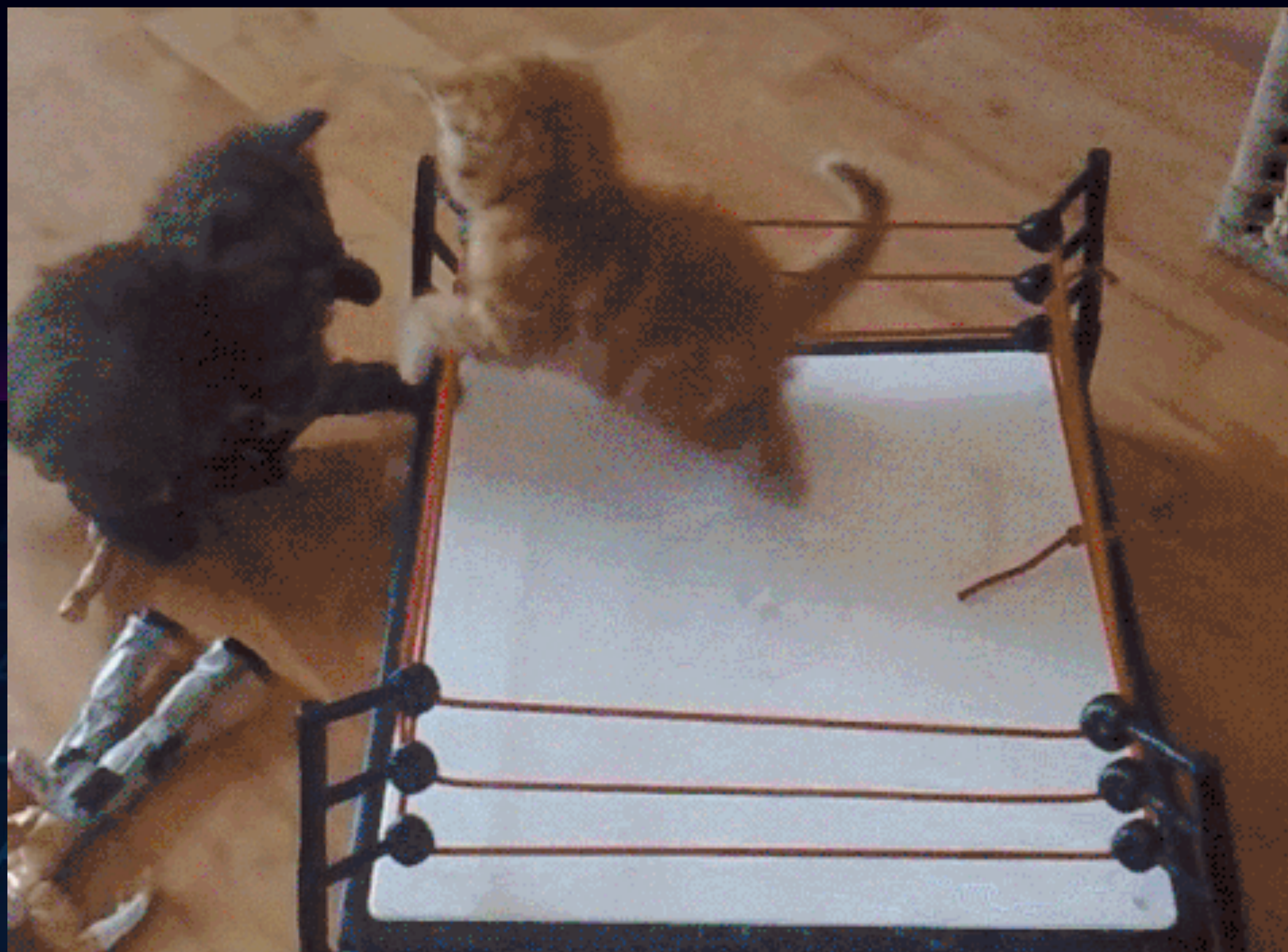


Step 1:

Identify a Good Use Case

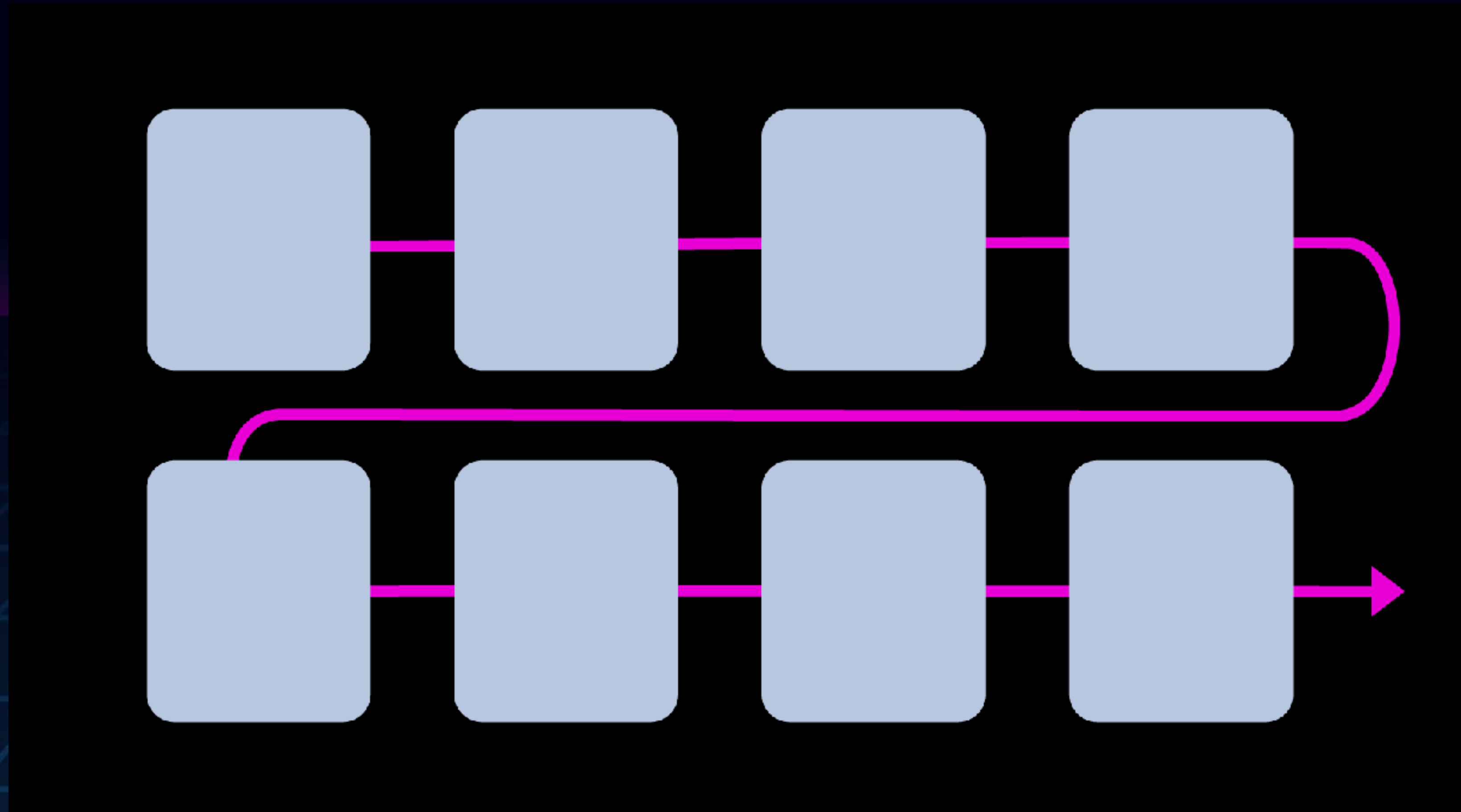


# Flexbox vs. Grid

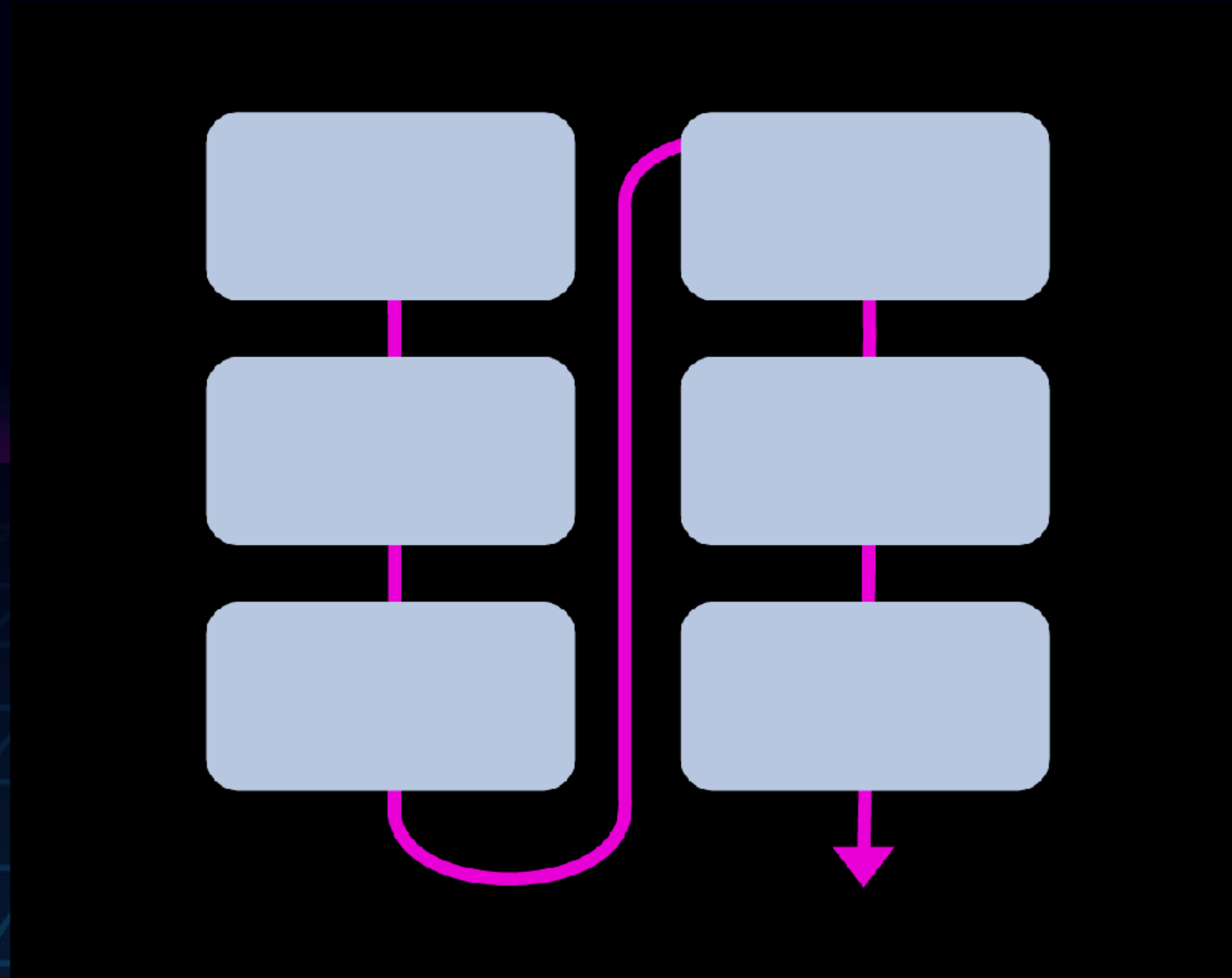




Flexbox is great for laying out items in one direction.  
In rows....

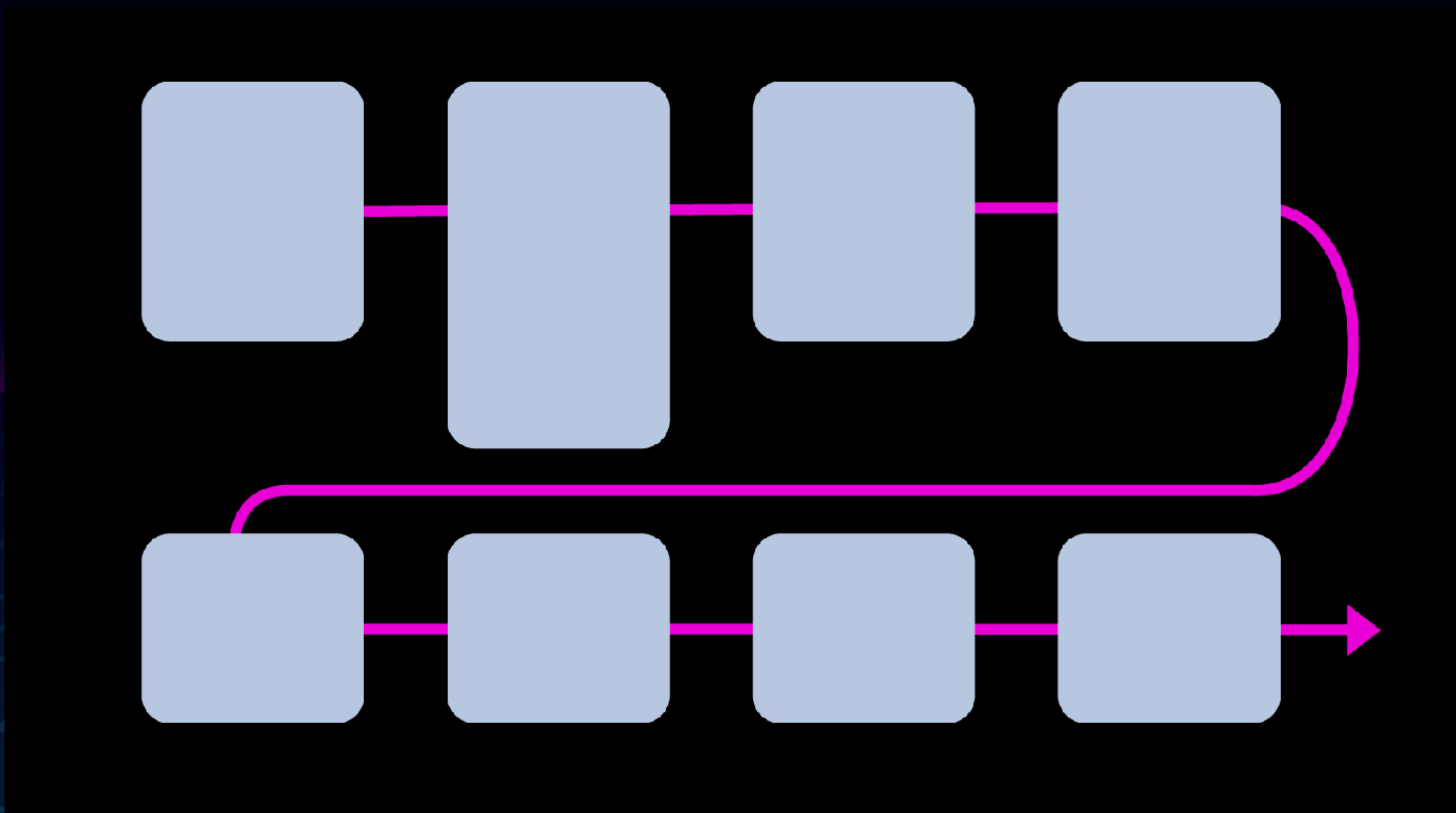


...or columns

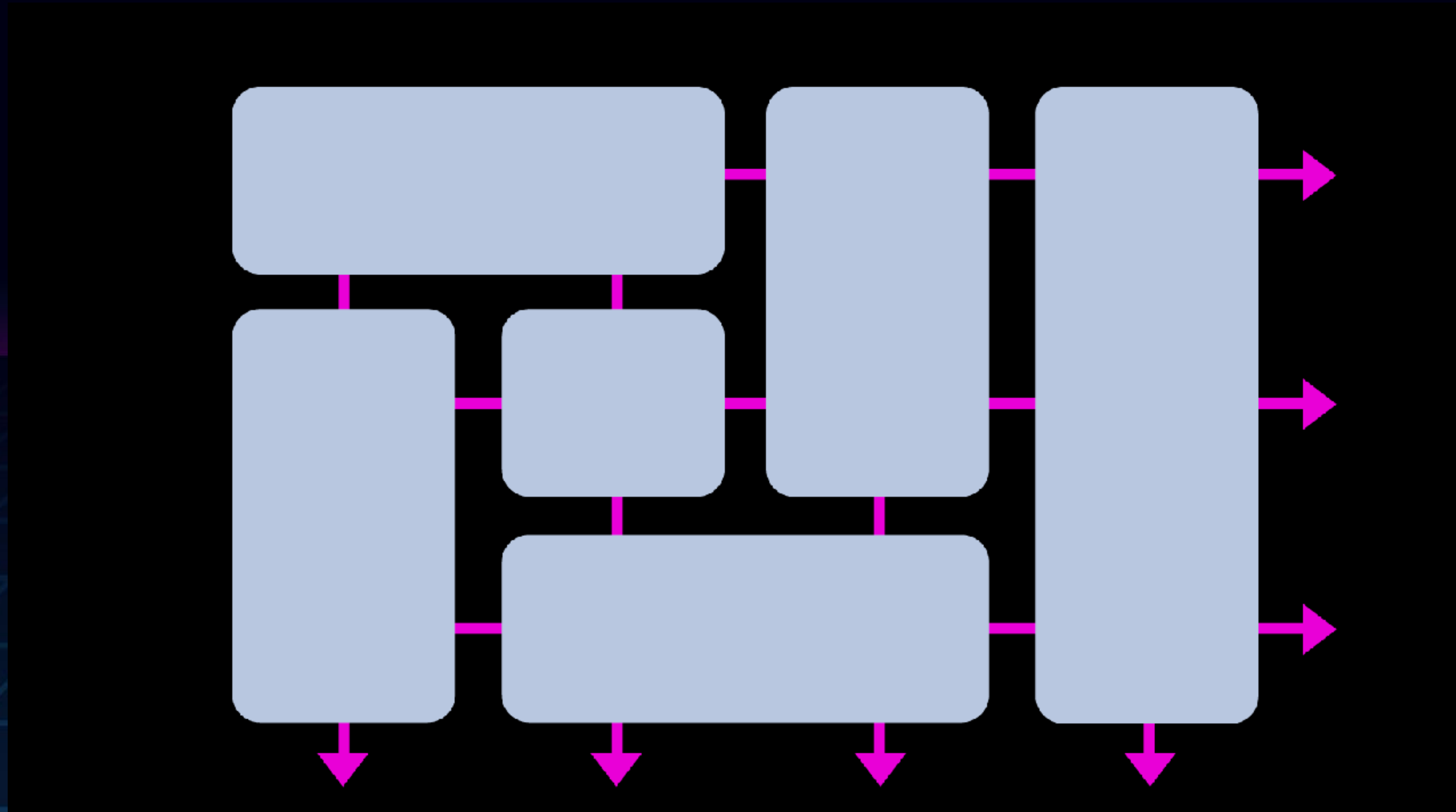




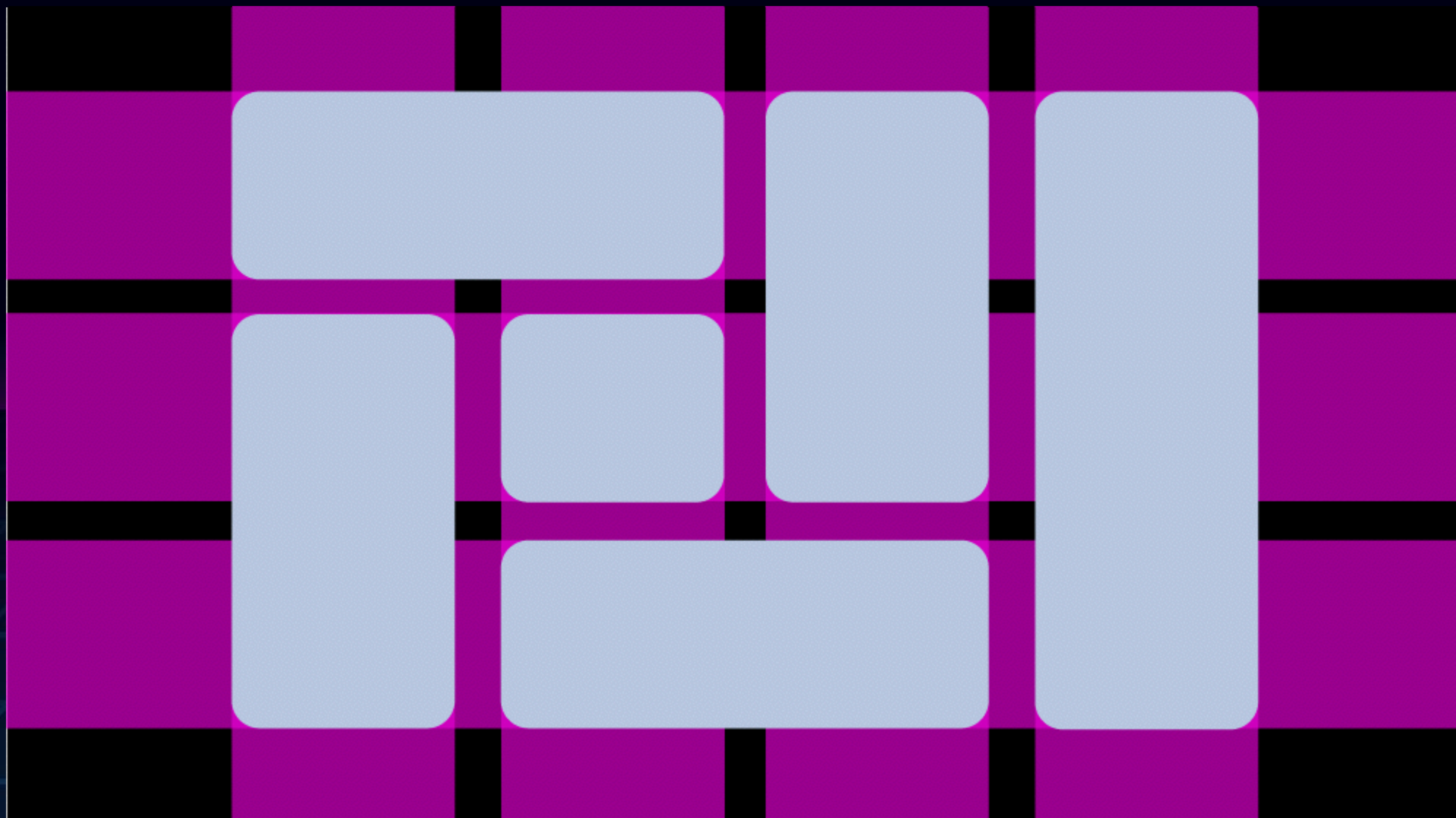
Uh oh.

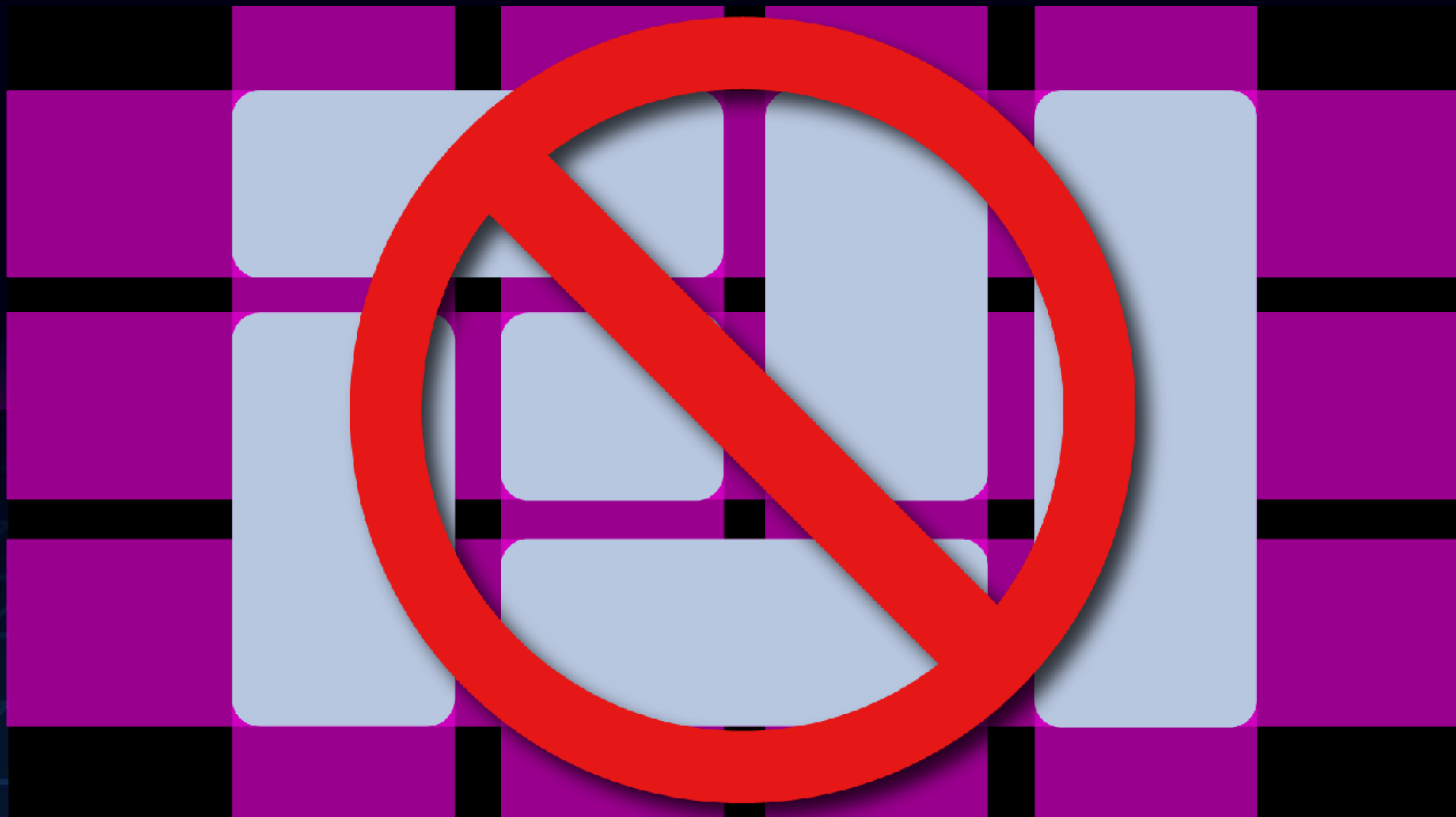


Grid allows you to lay out elements in two directions,  
rows & columns

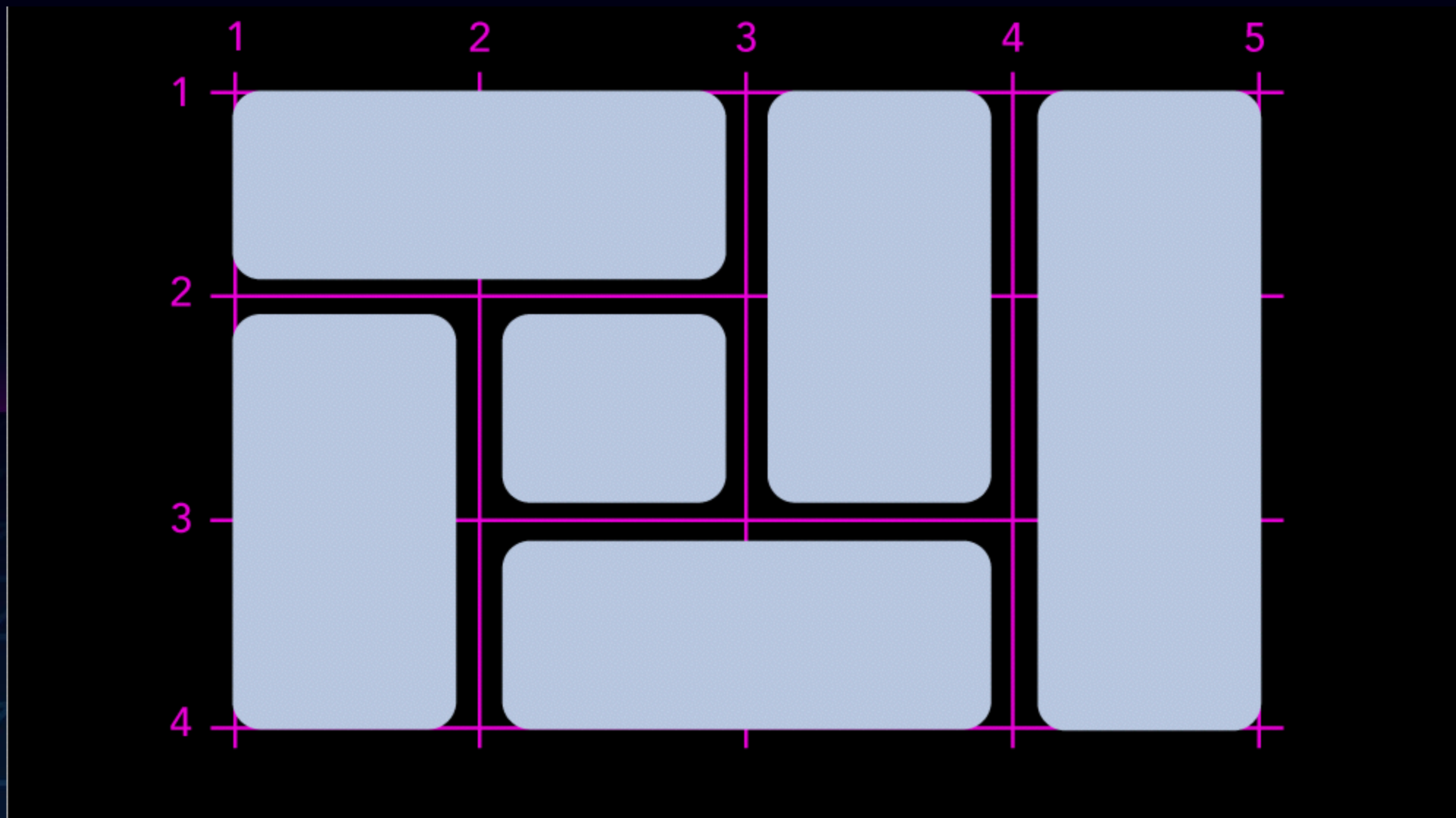












When normally reaching for a layout property like  
Flexbox, floats, or positioning,  
consider Grid. 🙋



# A good use case



# Step 2:

## Write Grid Code



# The Markup (simplified)

```
<ul class="search-results">
  <li class="search-results__result">
    
  </li>
  <li class="search-results__result">
    
  </li>
  ... more <li>s
</ul>
```

# The CSS 🍾

```
.search-results {  
  display: grid;  
  grid-gap: 10px;  
  grid-template-columns: repeat(auto-fit, minmax(320px, 1fr));  
  padding: 10px;  
}
```



Step 3:

Fallback gracefully with feature queries


# @supports

```
@supports (display: grid) {  
  .search-results {  
    display: grid;  
    grid-gap: 10px;  
    grid-template-columns: repeat(auto-fit, minmax(320px, 1fr));  
    padding: 10px;  
  }  
}
```



# @supports Support

## # CSS Feature Queries - CR

Usage % of all users   
Global 94.03%

CSS Feature Queries allow authors to condition rules based on whether particular property declarations are supported in CSS using the @supports at rule.

Current aligned Usage relative Date relative [Show all](#)

IE	Edge *	Firefox	Chrome	Safari	iOS Safari *	Opera Mini *	Chrome for Android	UC Browser for Android	Samsung Internet
			49						
			63		10.2				
			64		10.3				4
11	16	58	65	11	11.2	all	64	11.8	6.2
	17	59	66	11.1	11.3				
		60	67	TP					
		61	68						

# Fallin' back with Flexbox

```
.search-results {  
  display: flex;  
  flex-wrap: wrap;  
  padding: 5px;  
}
```

```
.search-results__result {  
  margin: 5px;  
}
```

```
@media only screen and (min-width: 500px) {  
  .search-results__result {  
    width: calc(50% - 10px);  
  }  
}
```

```
@media only screen and (min-width: 800px) {  
  .search-results__result {  
    width: calc(33.3333% - 10px);  
  }  
}
```

```
@media only screen and (min-width: 1200px) {  
  .search-results__result {  
    width: calc(25% - 10px);  
  }  
}
```



```
.search-results {  
  display: flex;  
  flex-wrap: wrap;  
  padding: 5px;  
}  
  
@supports (display: grid) {  
  .search-results {  
    display: grid;  
    grid-gap: 10px;  
    grid-template-columns: repeat(auto-fit, minmax(320px, 1fr));  
    padding: 10px  
  }  
}
```

```
.search-results__result {  
  margin: 5px;  
}  
  
@media only screen and (min-width: 500px) {  
  .search-results__result {  
    width: calc(50% - 10px);  
  }  
}  
  
@media only screen and (min-width: 800px) {  
  .search-results__result {  
    width: calc(33.3333% - 10px);  
  }  
}  
  
@media only screen and (min-width: 1200px) {  
  .search-results__result {  
    width: calc(25% - 10px);  
  }  
}  
  
@supports (display: grid) {  
  .search-results__result {  
    margin: 0;  
    width: auto;  
  }  
}
```



# Grid code with a Flexbox fallback: Ready for production!

```
.search-results {
  display: flex;
  flex-wrap: wrap;
  padding: 5px;
}

@supports (display: grid) {
  .search-results {
    display: grid;
    grid-gap: 10px;
    grid-template-columns: repeat(auto-fit, minmax(320px, 1fr));
    padding: 10px
  }
}

.search-results__result {
  margin: 5px;
}

@media only screen and (min-width: 500px) {
  .search-results__result {
    width: calc(50% - 10px);
  }
}

@media only screen and (min-width: 800px) {
  .search-results__result {
    width: calc(33.3333% - 10px);
  }
}

@media only screen and (min-width: 1200px) {
  .search-results__result {
    width: calc(25% - 10px);
  }
}

@supports (display: grid) {
  .search-results__result {
    margin: 0;
    width: auto;
  }
}
```

# Order matters

Fallback first as the default

Grid code last, so it can override the default styles



# A few considerations...

## Disable Grid in Autoprefixer

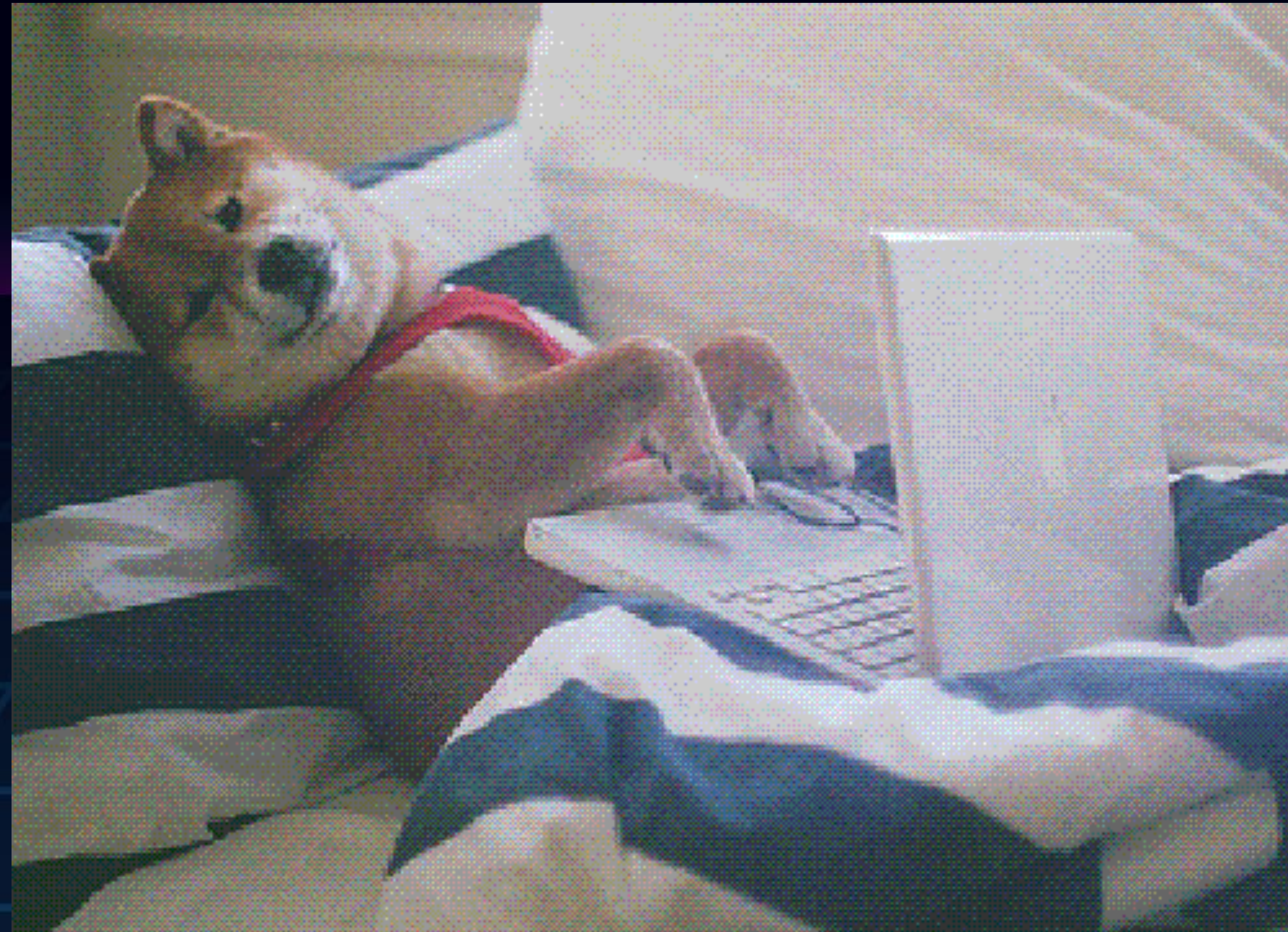
- This is the default for Autoprefixer v7.0 and greater

## Some syntax not supported in Sass until v3.5

- Sass-rails only running Sass v3.1



# Step 4: Ship it





Everyday examples!\*

# \*disclaimer

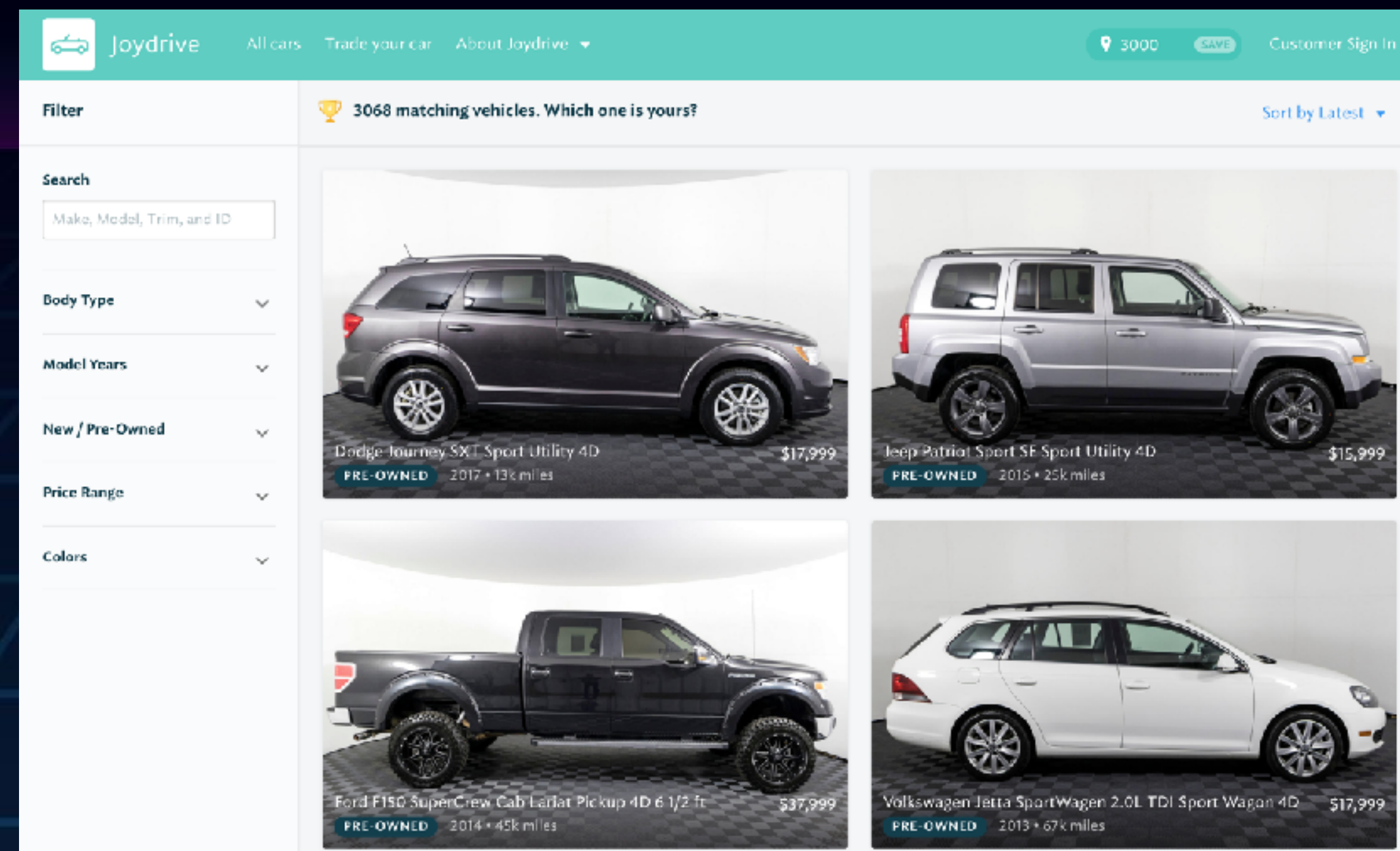
“It’s ok that when I look back at my old code  
I sometimes say, ‘Whatever was I thinking?!!’  
That means I’m still learning & growing.”

– Brenda Storer



# Beyond media queries

Responsive based on content size  
rather than screen size



# What things can CSS Grid do that CSS couldn't do before?

Control placement around unknowns.  
Using media queries with grid.

**HOTSHOTPD**



# Graceful Degradation

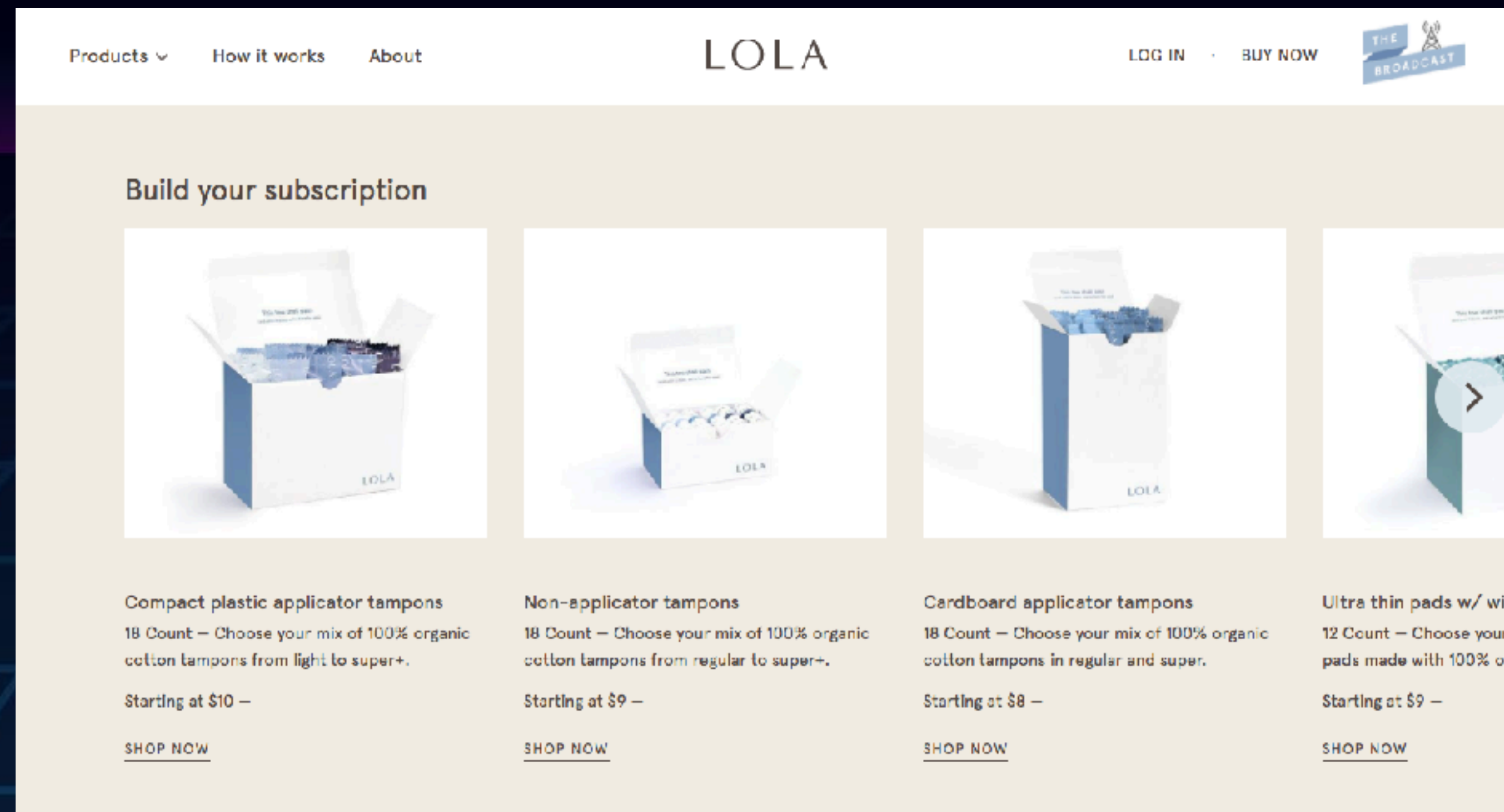
Who made the rules that web pages must look the same in every browser?



# It's a Grid party

Centered logo in header

Using fr units instead of margin auto





# RESOURCES

Layout Land - Jen Simmons

- Basics of CSS Grid: The Big Picture

Labs - Jen Simmons

Grid By Example - Rachel Andrew

- Get Started Guide
- Flexible Sized Grids with auto-fill and minmax
- The fr unit

From Bootstrap to CSS Grid - Natalya Shelburne



# Grid + IE

Should I try to use the IE implementation of CSS  
Grid Layout? — Rachel Andrew

Internet Explorer + CSS Grid???? — Jen Simmons

# Resources from Me

[Codepen Collection of CSS Grid Examples](#)

[A Fearless Guide to Using CSS Grid Today](#)



# Thank you!

@brendamarienyc

<http://brendastorer.com>

<http://brendastorer.com/presentations/2018-04-GenerateNYC.pdf>