

Architecting CSS grid in a design system

By [Chen Hui Jing](#) / [@hj_chen](#)



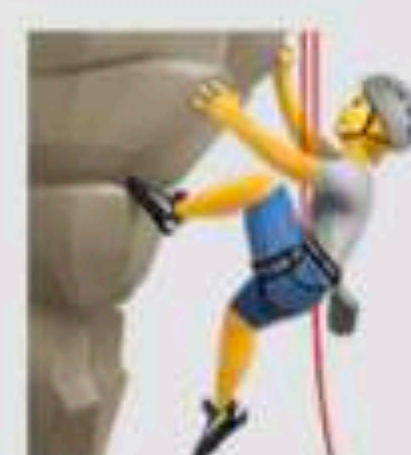
Surname

First name



陈
Chen

慧 晶
Hui Jing



@hj_chen





Screens, screens, screens

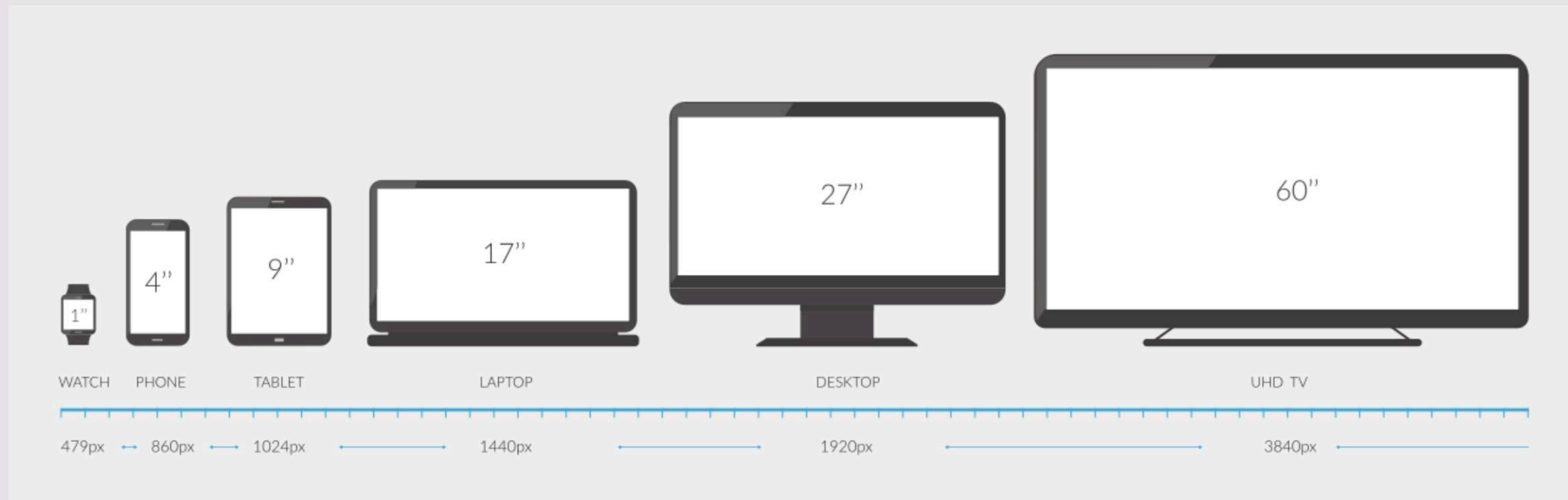


Image source: [Inch Calculator](#)

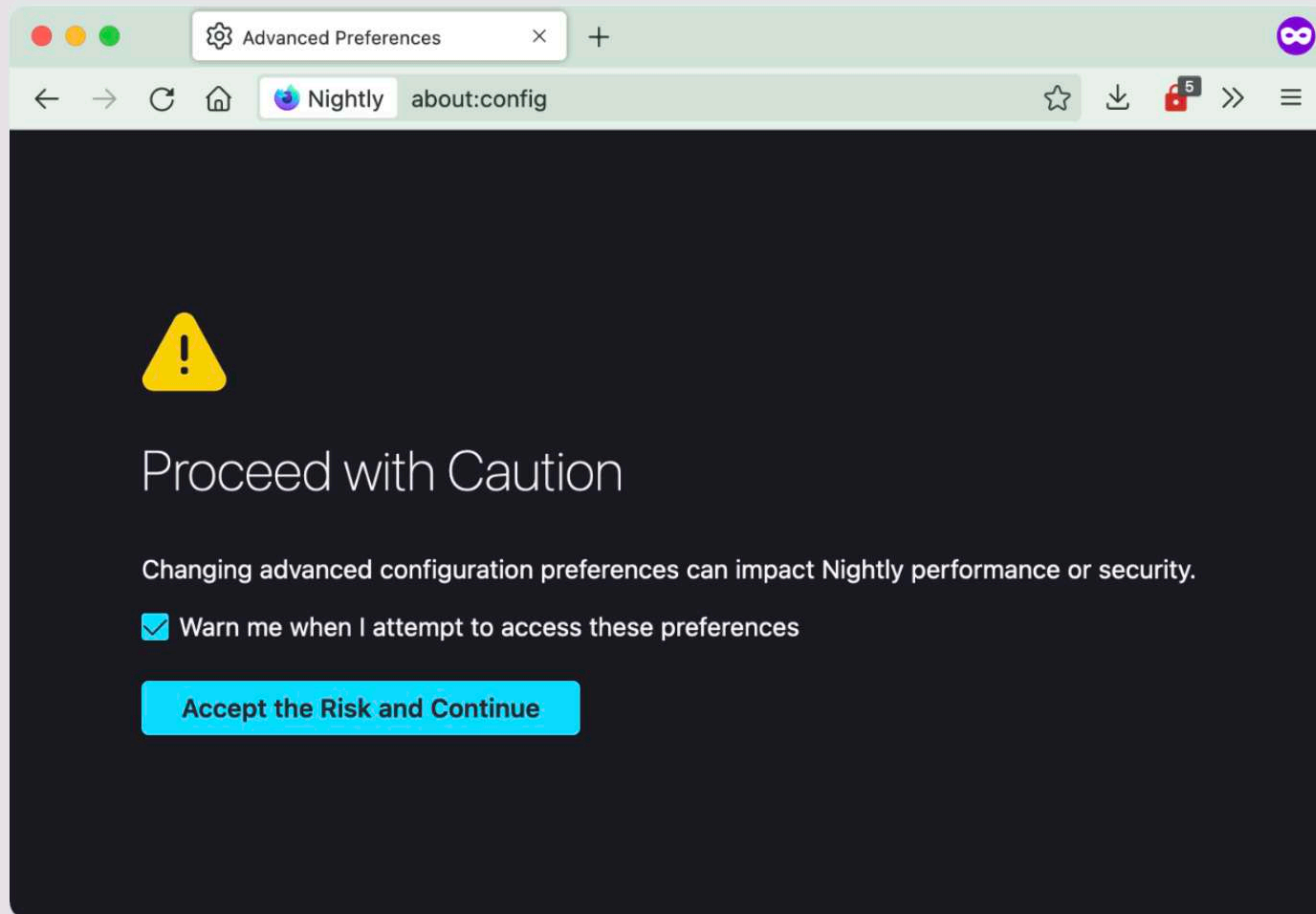


Image credit: [Jyotika Sofia Lindqvist](#)

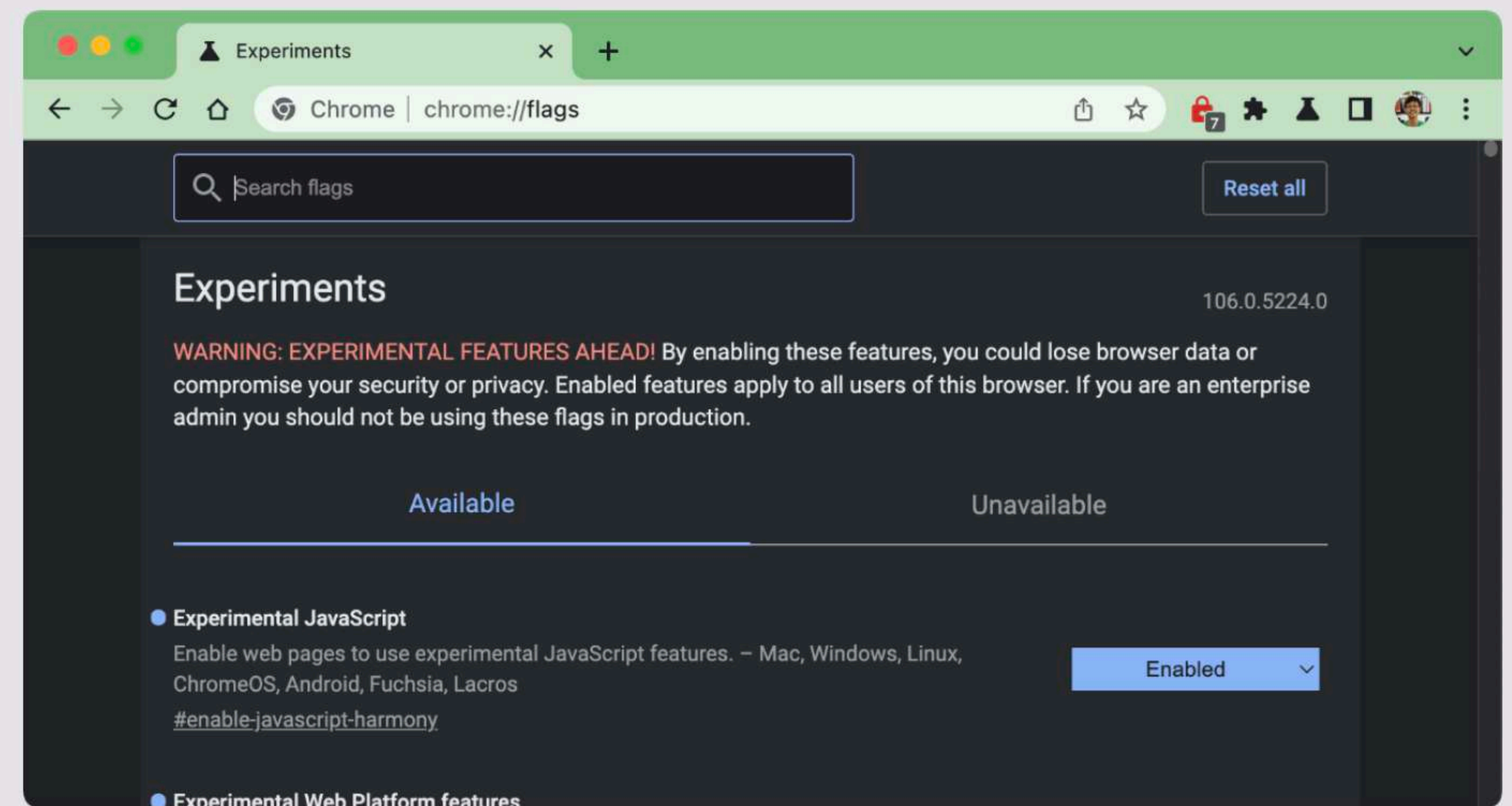

```
.wrapper {  
  display: -webkit-box;  
  display: -webkit-flex;  
  display: -ms-flexbox;  
  display: flex;  
}
```

[MDN: Backwards Compatibility of Flexbox](#)

Browser configuration pages







about:config



chrome:://flags

Grid release dates

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					

CSS Flexible Box Layout Module

Method of positioning elements in horizontal or vertical stacks. Support includes all properties prefixed with `flex`, as well as `display: flex`, `display: inline-flex`, `align-content`, `align-items`, `align-self`, `justify-content` and `order`.

IE	Edge	Firefox	Chrome	Safari	iOS Safari	Opera Mini	Chrome for Android	Android Browser	Samsung Internet
9	102	102	102	15.4	15.4			4.4	16.0
10	103	103	103	15.5	15.5			4.4.4	17.0
11	104	104	104	15.6	15.6	all	104	104	18.0
		105	105	16.0	16.0				

✓

✗

Partial Support

Prefixed

Global: 98.73% + 0.95% = 99.68%

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	102	102	102	15.4	15.4			4.4	16.0
10	103	103	103	15.5	15.5			4.4.4	17.0
11	104	104	104	15.6	15.6	all	104	104	18.0
		105	105	16.0	16.0				

✓

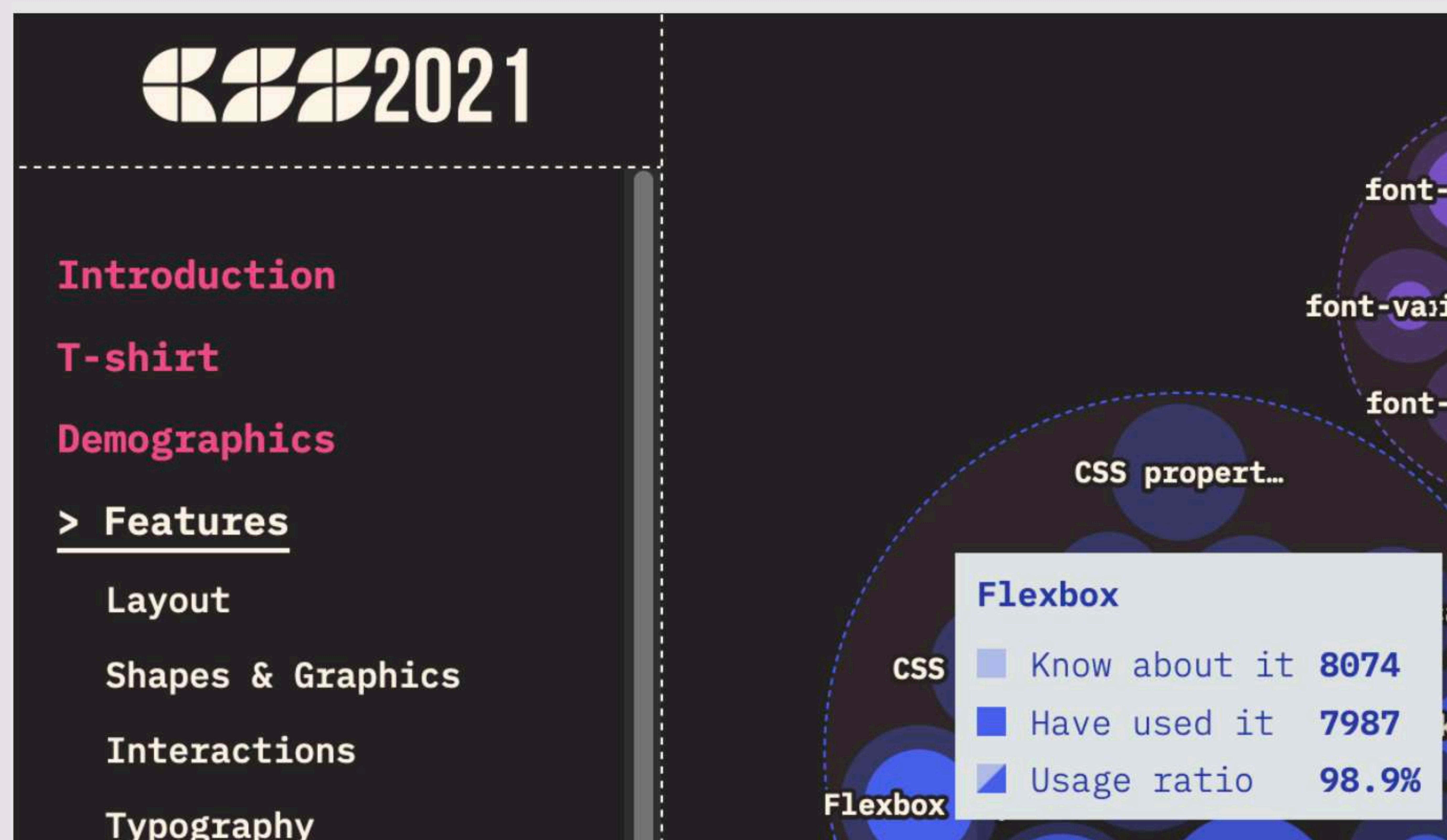
✗

Partial Support

Prefixed

Global: 96.32% + 0.53% = 96.85%

State of CSS 2021 survey



Flexbox: 98.9%



Grid: 83.5%

<https://2021.stateofcss.com/en-US/>

ome

earch

our Library

reate Playlist

ked Songs

ng Ji

nav-bar

ng

Sokko

s a day

ame Symphony

da Jam

z Cat

nd I

orite Coffeehouse

tall App

Good evening



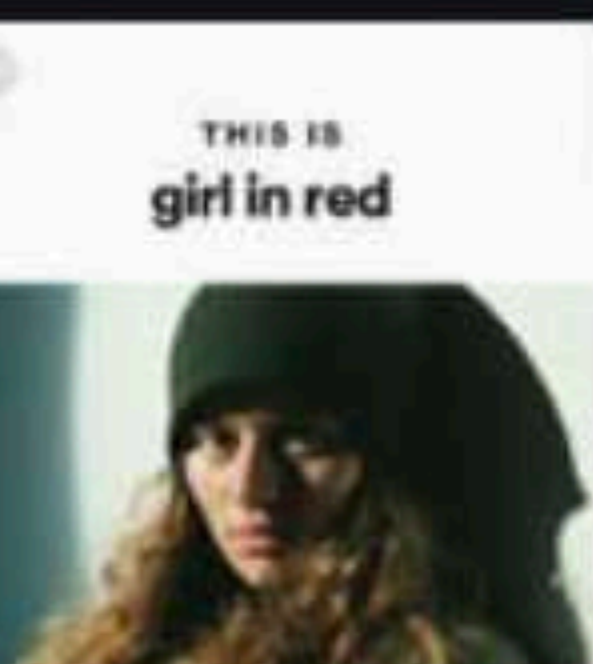
RENAISSANCE



This Is CHVRCHES



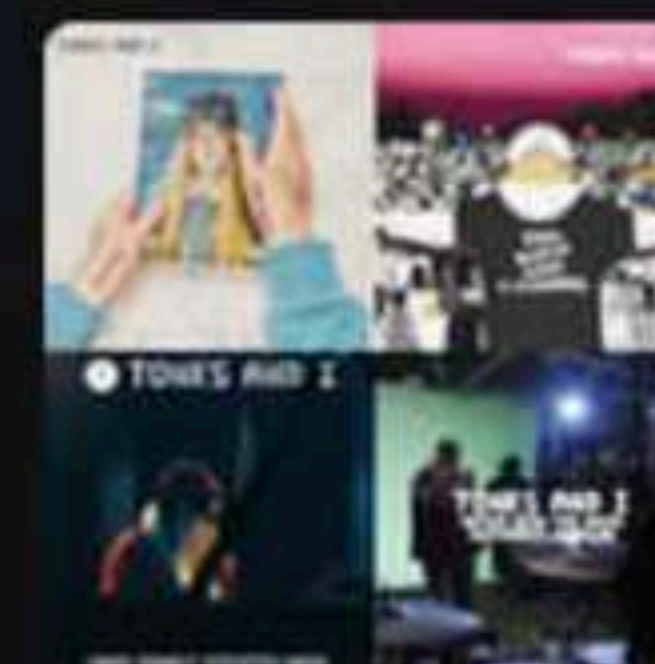
Pop Rising



This Is girl in red



Today's Top Hits



Tones and I

Your top mixes



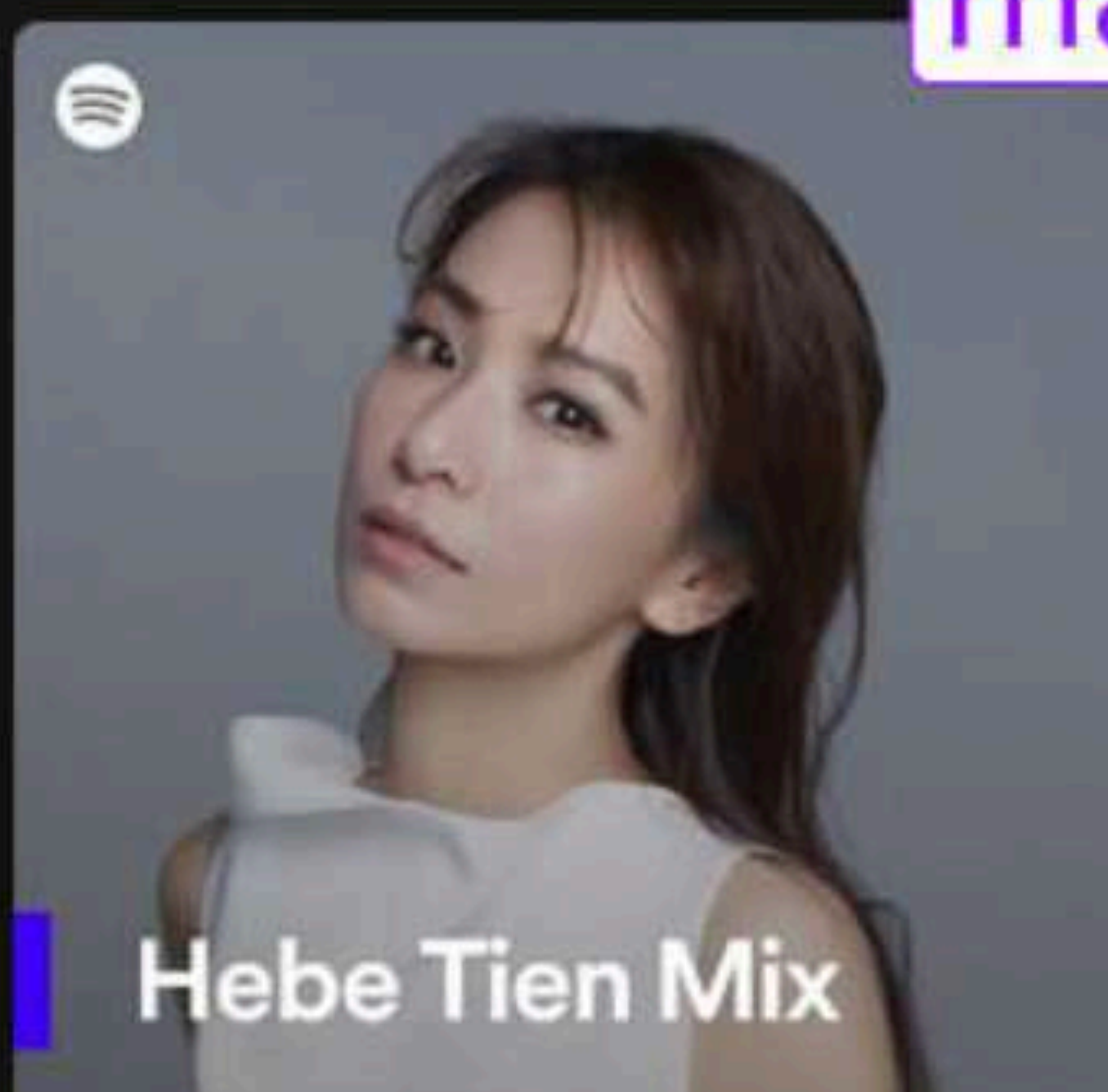
Lizzo Mix

Lizzo Mix

Beyoncé, Tinashe, Cardi B
and more

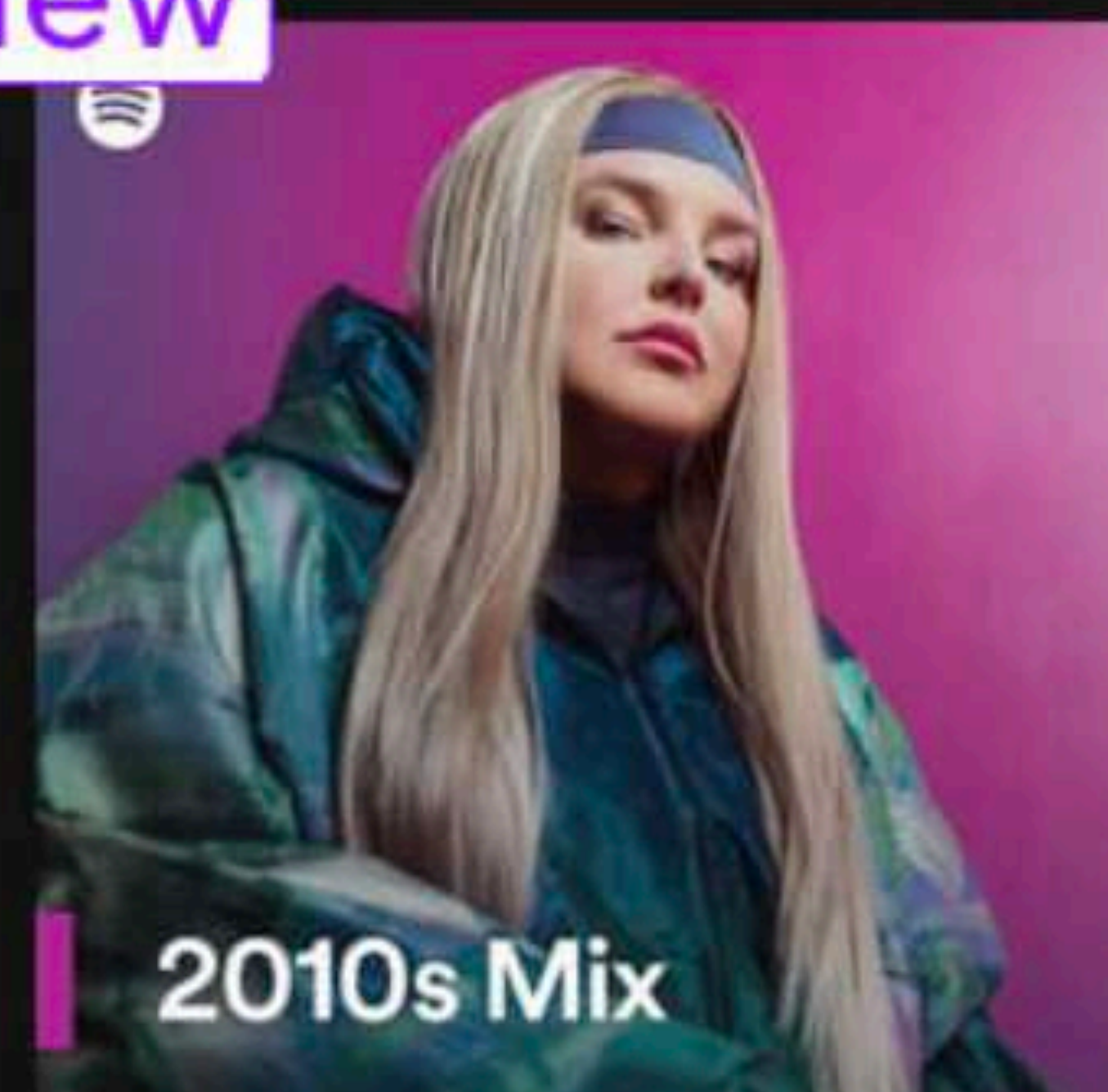
Beyoncé Mix

Beyoncé Mix

Tems, Lizzo, City Girls and
more

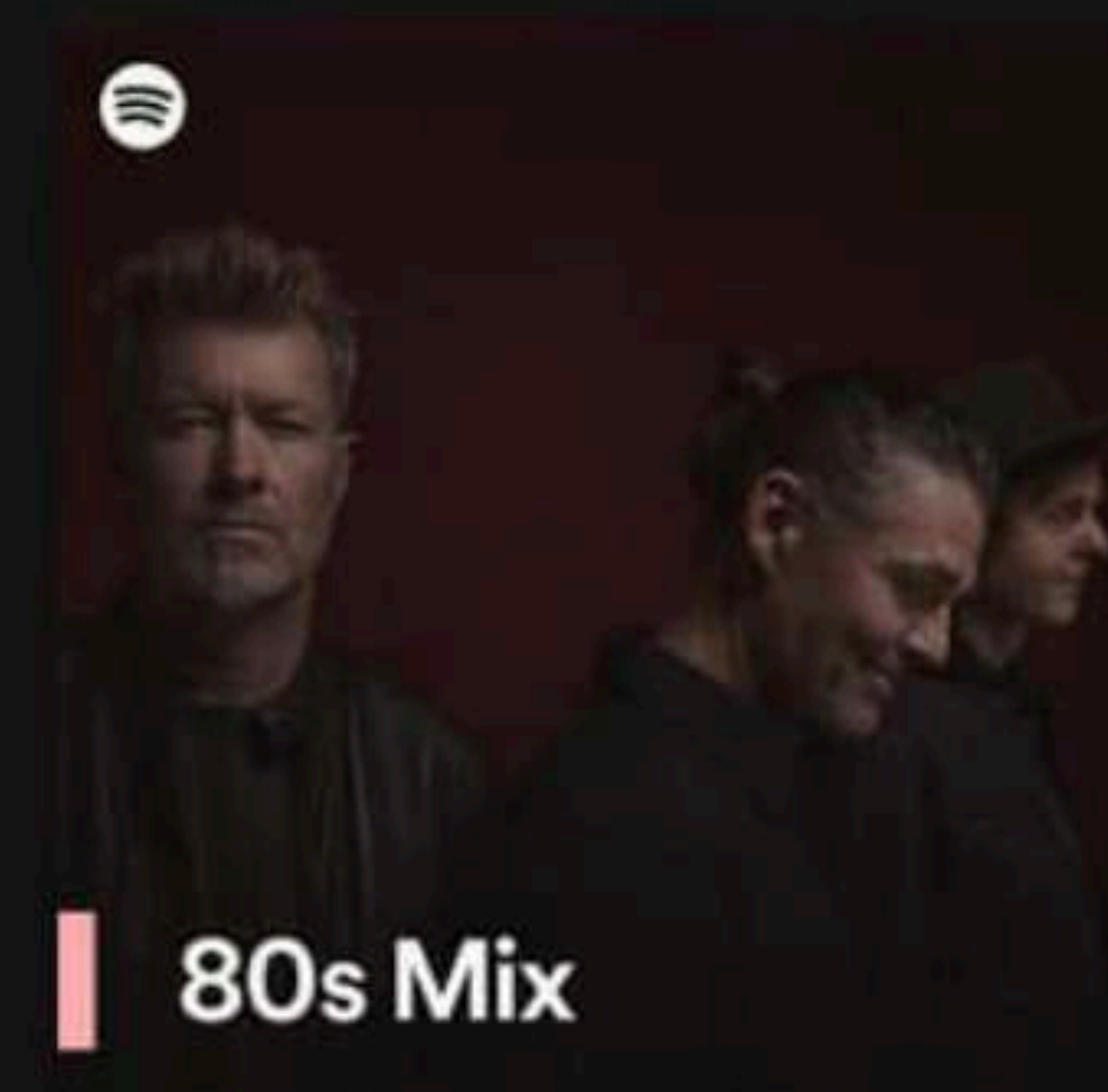
Hebe Tien Mix

Hebe Tien Mix

Sam Lee, WeiBird, Jay
Chou and more

2010s Mix

2010s Mix

Tones And I, Lizzo, Hayley
Kiyoko and more

80s Mix

80s Mix

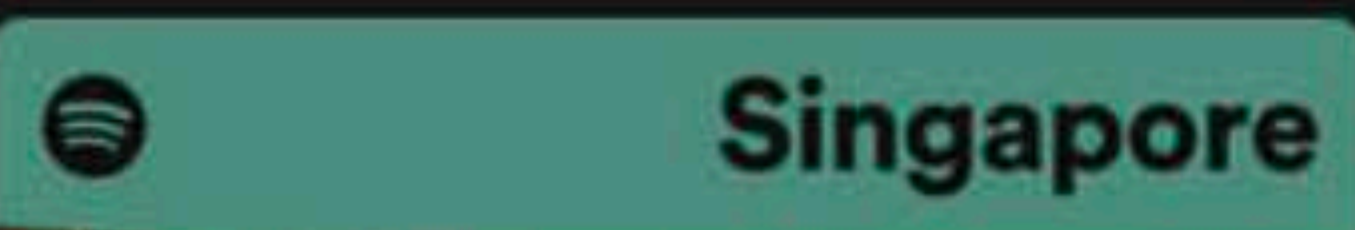
a-ha, James Ingram,
CoCo Lee and more

2000s Mix

2000s Mix

Rene Liu, Sam Lee and
more

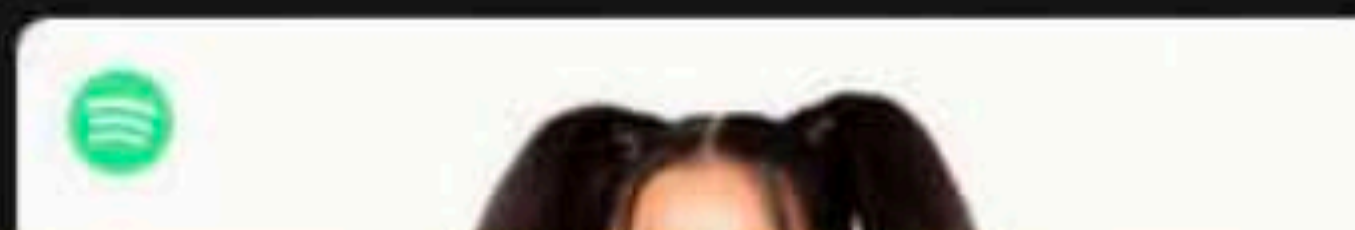
Today's biggest hits



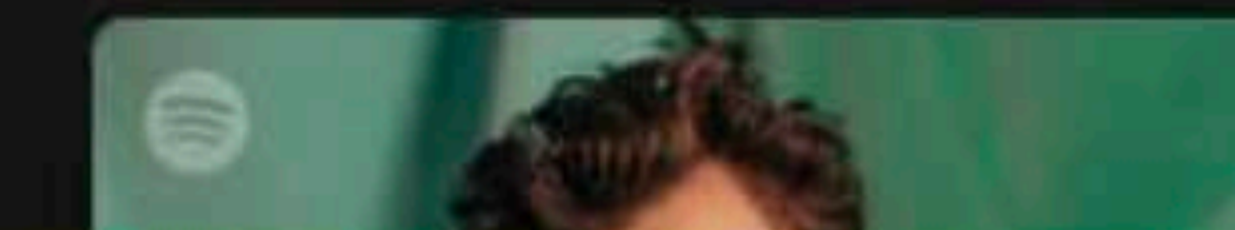
Singapore



是Hit華語榜



It's a Hit!

K-Pop
OverK-Pop
Over

Disclaimer

The following theory may or may not oppose your view on the matter, and that is PERFECTLY FINE. I am not here to tell you what to think, merely here to share a theory based on my personal observations and experiences. You are absolutely free to agree, disagree or not care at all.

Christina's new album
now! [Click here](#) to buy.
Christina Aguilera Faceplate
Phone! [Buy it here.](#)
New & hot video "Dirrty"!
connection speed:
[MED](#) | [HI](#)
[LO](#) | [MED](#) | [HI](#)



THE MUSIC PHOTOS **NEWS & TOUR** DOWNLOAD



[CLICK HERE](#) FOR MORE PHOTOS >



your email address
sign-up for Christina's
exclusive mailing list:

TOP NEWS

- DIRRTY #1 in the UK! [Read on.](#)
- November 22, 8:00 PM ET- Performing Dirrty with Redman on the MTV Europe Awards.
- December 2- Performing on Jay Leno
- December 4- Performing on MY VH1 Awards.
- from MTV.com:
"The first week we had the Christina video on the site it did almost double the number of streams than the 2nd most streamed video. She also had the #1 most trafficked artist page that week with over 15,000 more hits that #2 Eminem."
- VIBE TV - Crew came to the video shoot of "Beautiful" for a Vibe Style segment, Air date: January 11th, 2003.



Get your Christina Aguilera Faceplate and Nextel Phone! [Buy it here.](#)

Chronos the Cat's Anime Addventure Episodes and Stories

[My Storylines](#) | [Other Episodes](#)

ry. Like a Round Robin, different authors
e Round Robins, addventures branch, allo
(inclined) to take the story in different
to Anime fan fiction addventures (and
can be found [here](#).

Neopia Central

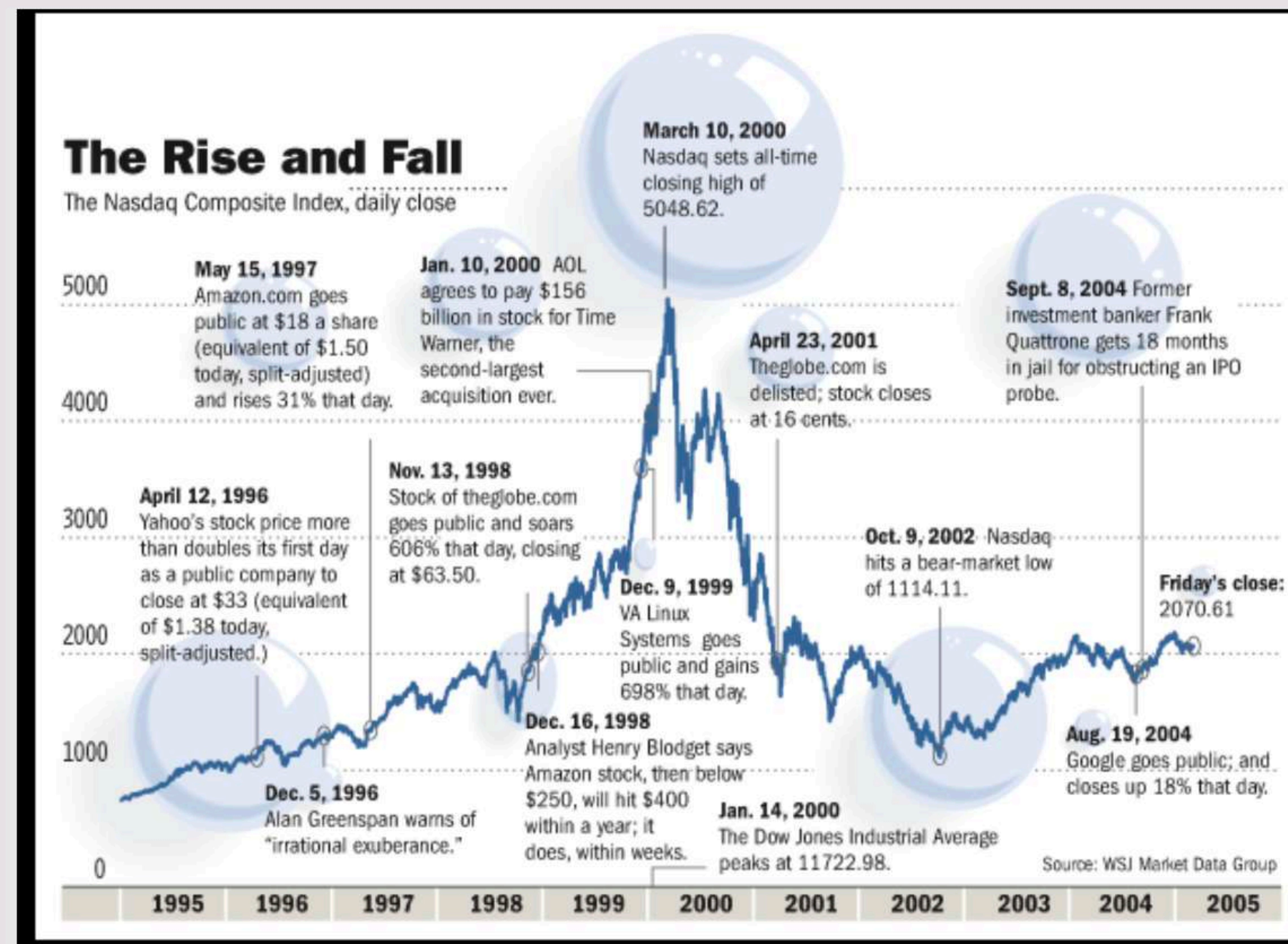
Click on the shop you wish to visit. You can buy food or toys for your pet, collect rare items and trading cards, and even buy clothes.

The shops restock roughly 7 times every hour!

To see the rest of the shops click the left of the main oval on the toolbar above!



Dotcom bubble (1995-2002)



Source: [The History of the Dotcom Bubble](#)

Welcome to Michel's Video Games page!

Last update: 16 September 1994

These pages have been consulted 1088 times September 15th 1994!

Fast jumps to sections: [What's New](#), [3DO](#), [Doom](#), [Jaguar](#), [SNES](#), [Pinball](#), [Netrek](#), [Game Boy](#), [Shockwave \(3do\)](#), [Way Of The Warrior \(3do\)](#), [the Video Game FAQ Page](#)



This page is updated daily!! Check [the What's New page](#) to see the recent additions.

Introduction: Who am I?

I removed my digitized picture but if you really want to see my ugly face, check [my ugly face page!](#) Other pictures and info are availables in [my personnal home page](#).

Well, I'm a poor french lost in the United States for a year. I had to leave my SNES in France so I bought a 3DO as soon as I could when I arrived in Pittsburgh. I don't regret it. See the [3DO](#) and [SNES](#) related stuff further in this page. Check also [The Atari Jaguar](#) links. [Netrek](#) and [Doom](#) fans may find some interesting links too. [Pinball](#) players will find the best infos for their favorite pinballs too! Arcade Game players can't miss my [Video Game FAQ Page](#).

Please send me suggestions and interesting links by email at:

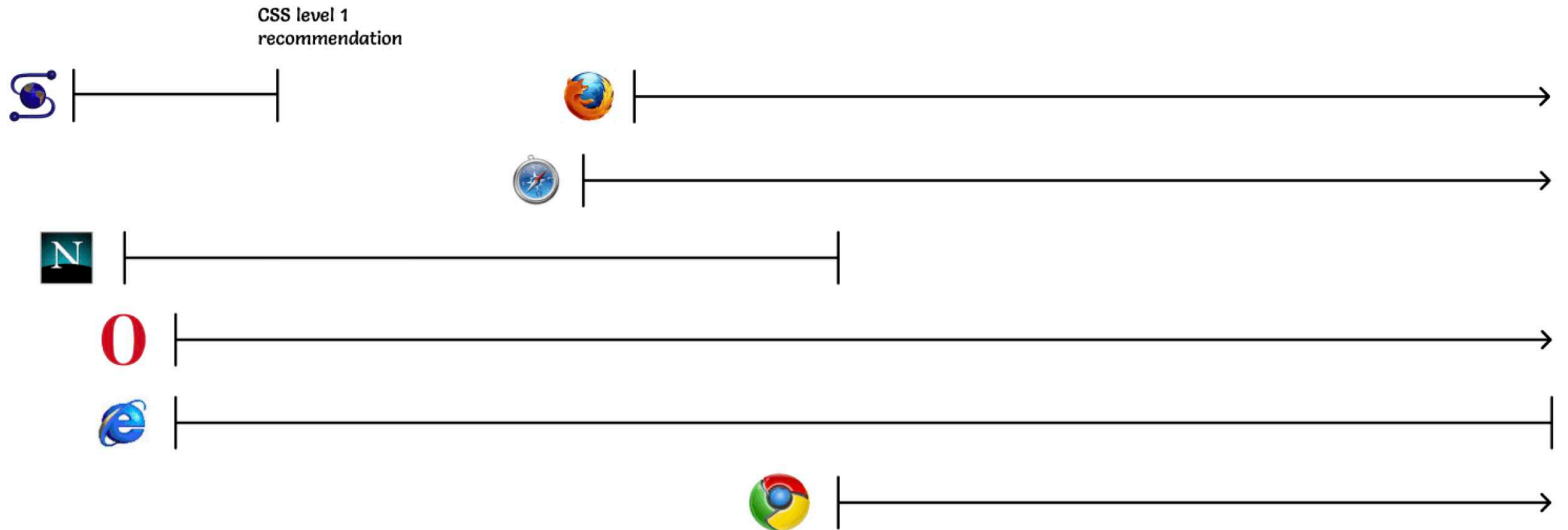
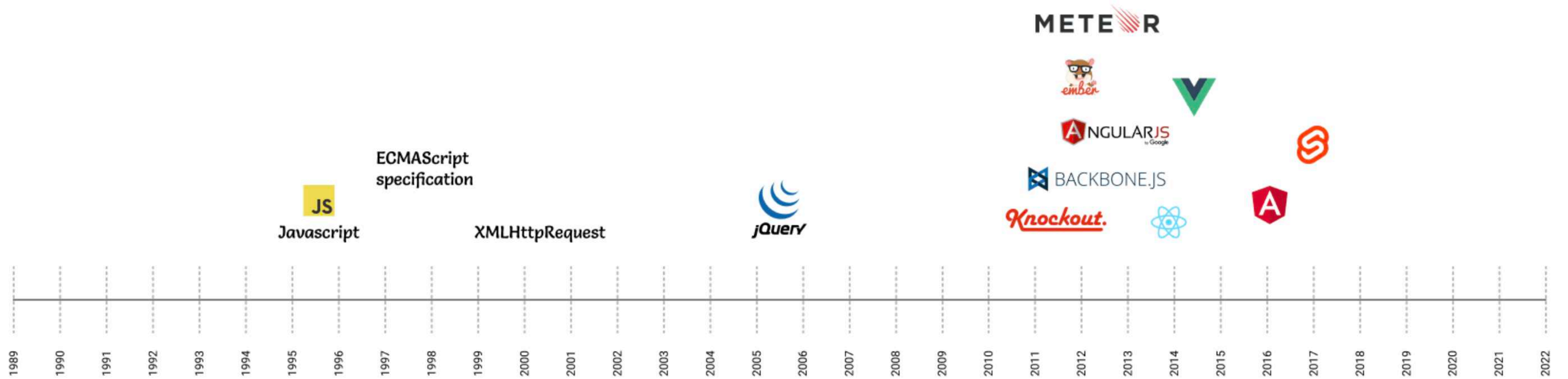
<Michel.Buffa@cmu.edu>

First, some interesting links for video game addicts:

Check these sites!

- [The Games Domain home page \(PC, X11 games, others...\)](#). Contains tons of FAQs in html format, lots of links to other games-related WWW pages. **Tons of links there!**
- [The official Game Bytes WWW pages!](#) Brand New on the WWW! Game Bytes is an electronic fanzine that talks mainly about the PC and computer games but also has columns for the consoles (SNES, Genesis, Sega CD, 3DO).
- [Video Games FAQS ftp server](#): Mortal Kombat I and II, NBA Jam, Art of Fighting, lots of infos about real arcade games and console/computer games. Some of the files available there in ASCII have been translated by me in mosaic format. **Check my Video Game FAQ Page.**
- Check out [the video game mosaic pages from Cardiff](#). This site contains lots of FAQs, pictures, info about video games for all type of machines.
- [ftp.digex.net](#). Almost anything you can imagine for most platforms. New site.
- [sunsite.unc.edu](#). Sega stuff only, including FAQs, electronic publications etc.
- [ftp.ee.pdx.edu](#). 3DO material, mostly lots of game screenshots in JPEG format.
- [busop.cit.wayne.edu](#). Miscellaneous material for CES, Turbo Duo, and Sega platforms. Very often Busy.
- [rtfm.mit.edu](#). For video game newsgroup archives:

Source: [Michel Buffa's Video Games Page in 1994](#)



SPA

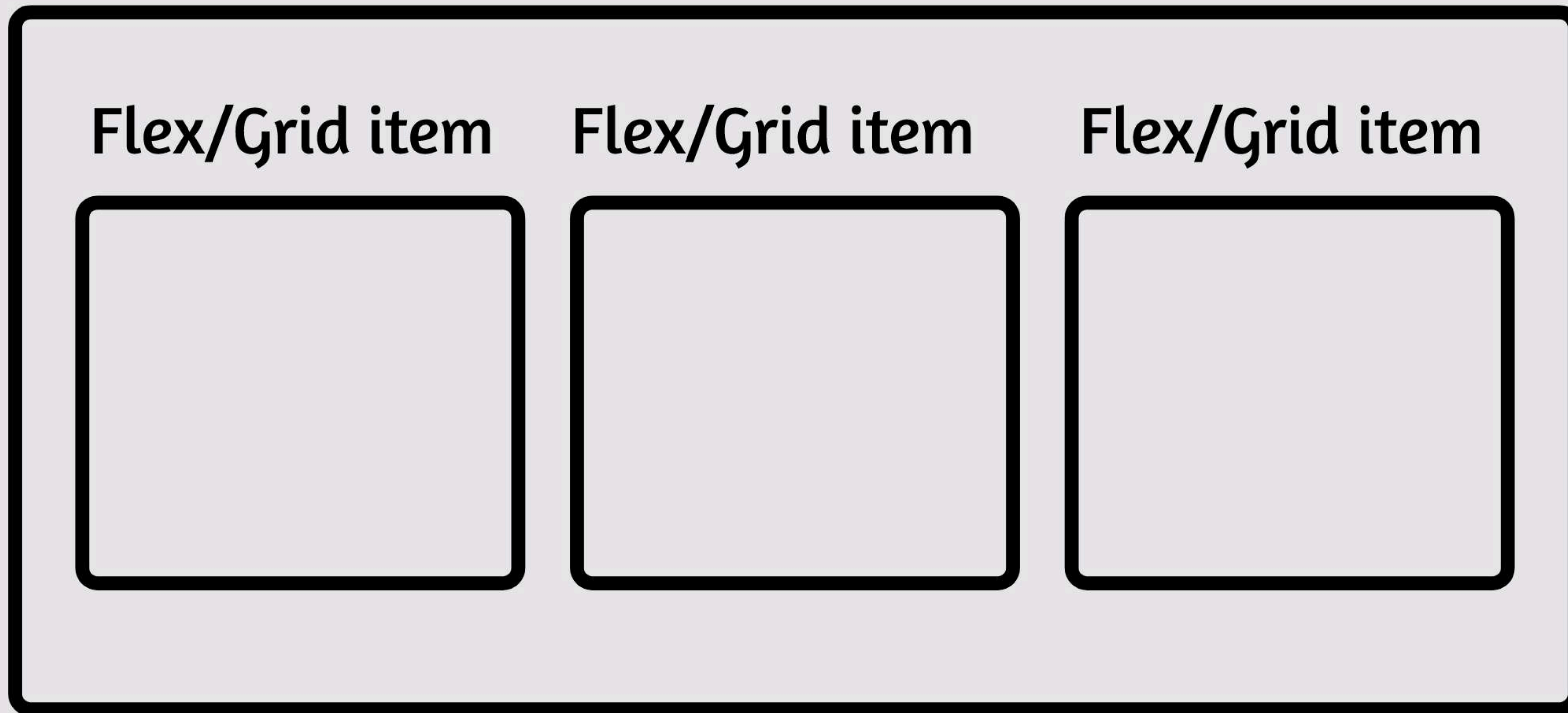


SPA

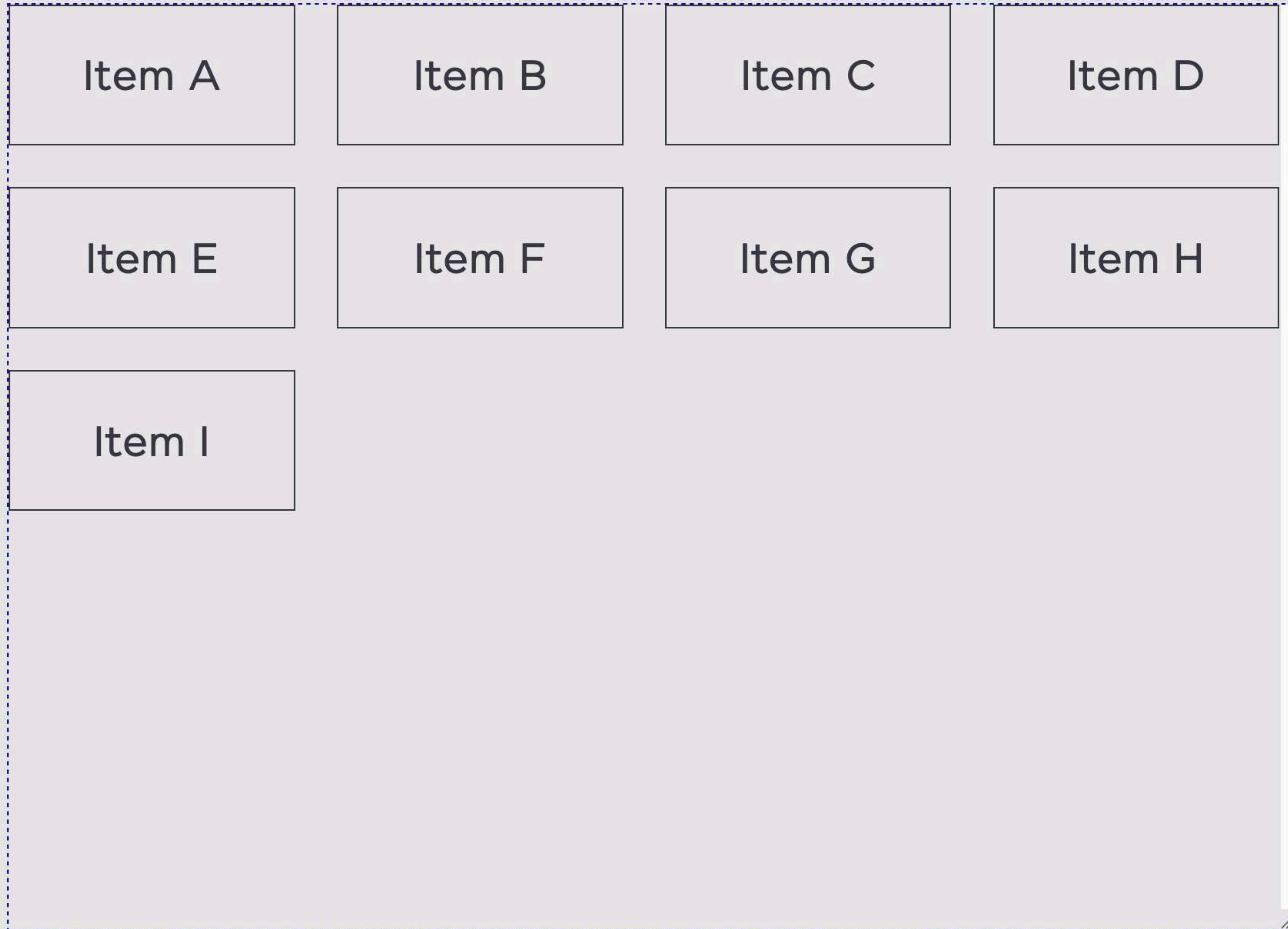


Parent-child relationship

Flex/Grid container



Basic grid syntax



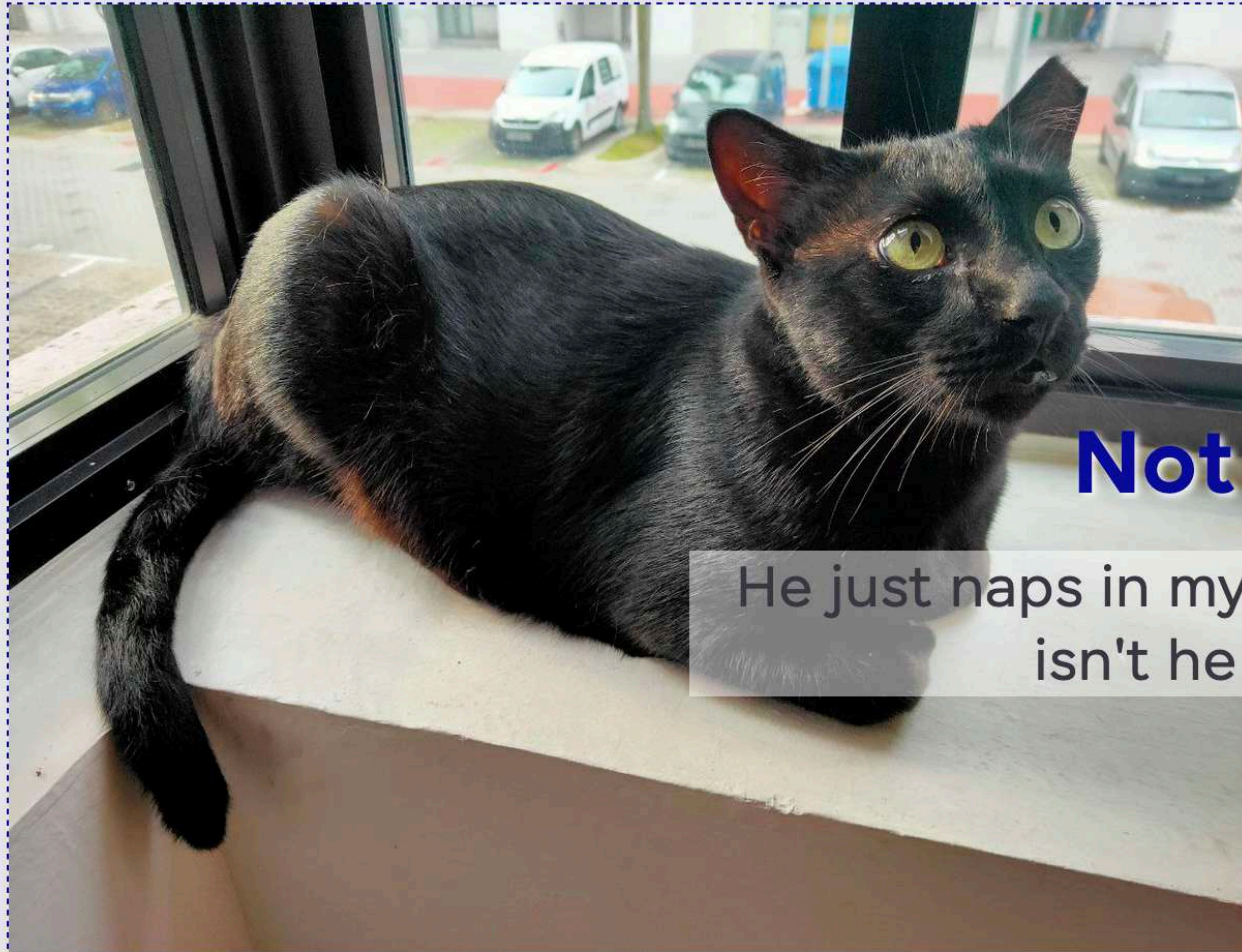
```
.basic-grid {  
  display: grid;  
  grid-template-columns:  
    repeat(auto-fit, minmax(200px,  
    1fr));  
  gap: 1em;  
}
```


Named grid areas



```
.named-grid {  
  display: grid;  
  grid-template-columns: 200px 1fr;  
  grid-template-rows: auto 1fr auto;  
  grid-template-areas: 'header header'  
                      'sidebar main'  
                      'footer footer';  
}  
  
.h { grid-area: header }  
.s { grid-area: sidebar }  
.m { grid-area: main }  
.f { grid-area: footer }
```


Placing grid items

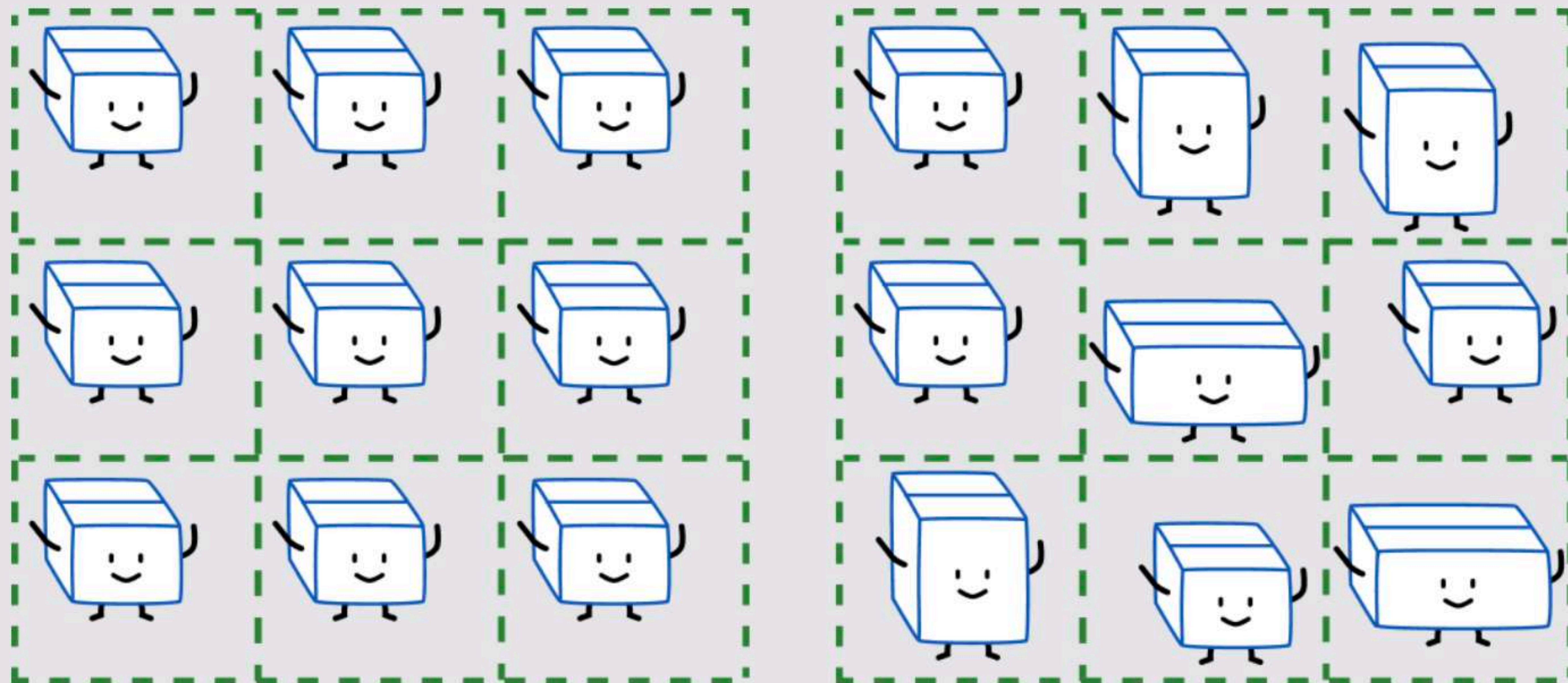


Not my cat

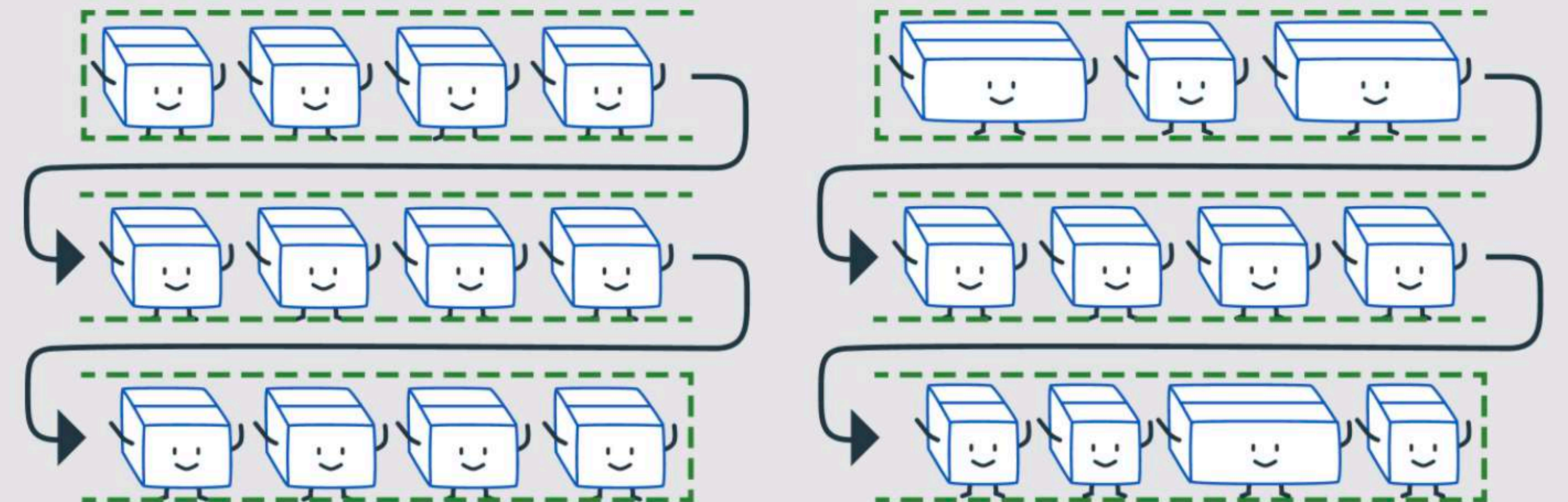
He just naps in my house sometimes. But isn't he super cute?

```
.overlap-grid {  
  display: grid;  
  grid-template-columns: 1fr 300px 1fr;  
  grid-template-rows: 1fr auto 1fr;  
}  
  
.image {  
  grid-column: 1 / span 2;  
  grid-row: 1 / -1;  
}  
  
.headline {  
  grid-column: 2 / 4;  
  grid-row: 2;  
}  
  
.text {  
  grid-column: 2 / 4;  
  grid-row: 3;  
}
```


Container-led sizing



Item-led sizing



Common example of grid system CSS

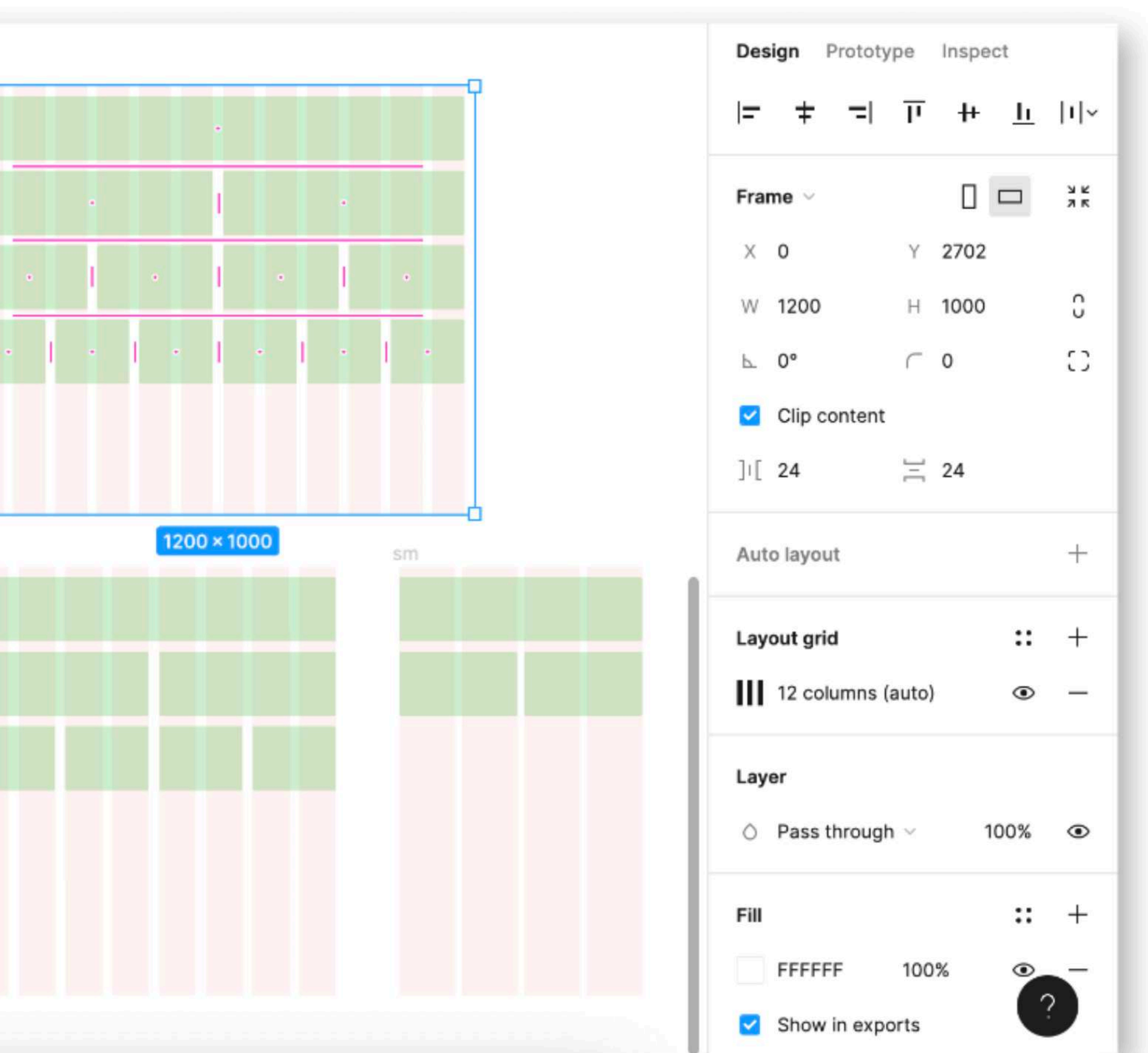
“Bootstrap’s grid system uses a series of containers, rows, and columns to layout and align content.”

```
.col-sm {  
  flex: 1 0 0%;  
}  
.row-cols-sm-auto > * {  
  flex: 0 0 auto;  
  width: auto;  
}  
.row-cols-sm-1 > * {  
  flex: 0 0 auto;  
  width: 100%;  
}  
.row-cols-sm-2 > * {  
  flex: 0 0 auto;  
  width: 50%;  
}  
.row-cols-sm-3 > * {
```

```
<div class="container">  
  <div class="row">  
    <div class="col-sm-4">  
      One of three columns  
    </div>  
    <div class="col-sm-4">  
      One of three columns  
    </div>  
    <div class="col-sm-4">  
      One of three columns  
    </div>  
  </div>  
</div>
```


A pretty standard grid

Size	Min	Max	Cols	Margin	Gutter
xs	320px	639px	4	16px	16px
sm	640px	899px	8	30px	16px
md	900px	1199px	12	50px	16px
lg	1200px	1599px	12	90px	24px
xl	1600px	-	12	>180px	24px



```
import { ReactNode, createElement } from "react";
import styles from "./Grid.module.scss";

interface GridProps extends React.HTMLProps<HTMLElement> {
  className?: string;
  children: ReactNode;
  tag?: keyof JSX.IntrinsicElements;
}

export default function Grid({
  className = "",
  children,
  tag = "div",
  ...props
}: GridProps) {
  const Wrapper = tag;
  return createElement(
    Wrapper,
    {
      className: `${styles.grid} ${className}`,
      ...props
    },
    children
  );
}
```


Option 1: vanilla CSS (or SCSS)

Size	Min	Max	Cols	Margin	Gutter
xs	320px	639px	4	16px	16px
sm	640px	899px	8	30px	16px
md	900px	1199px	12	50px	16px
lg	1200px	1599px	12	90px	24px
xl	1600px	-	12	>180px	24px

```
.grid {  
  min-width: 320px;  
  max-width: 1600px;  
  display: grid;  
  grid-template-columns: repeat(4, 1fr);  
  gap: 1em;  
  margin-left: 16px;  
  margin-right: 16px;  
}  
  
@media screen and (min-width: 640px) {  
  .grid {  
    grid-template-columns: repeat(8, 1fr);  
    margin-left: 30px;  
    margin-right: 30px;  
  }  
}
```


Option 1: vanilla CSS (or SCSS)

```
.grid__item--full,  
.grid__item--half,  
.grid__item--third,  
.grid__item--quarter {  
  grid-column: 1 / -1;  
}  
  
@media screen and (min-width: 640px) {  
  .grid__item--quarter {  
    grid-column: span 4;  
  }  
}  
  
@media screen and (min-width: 900px) {  
  .grid__item--half {  
    grid-column: span 6;  
  }  
}
```


Option 1: vanilla CSS (or SCSS)

```
.custom-thingy {  
  grid-column: 1 / -1;  
  font-size: var(--step-1);  
}  
  
@media screen and (min-width: 640px) {  
  .custom-thingy {  
    grid-column: 1 / 6;  
    padding-top: 2em;  
    padding-bottom: 1em;  
  }  
}  
  
@media screen and (min-width: 900px) {  
  .custom-thingy {  
    grid-column: 1 / 7;  
  }  
}
```


Option 2: Container and Item components

```
src/  
└── components/  
    ├── Col/  
    │   ├── Col.module.css  
    │   └── Col.tsx  
    └── Grid/  
        ├── Grid.module.css  
        └── Grid.tsx
```


Grid.tsx

```
import { ReactNode, createElement } from "react";
import styles from "./Grid.module.scss";

interface GridProps extends React.HTMLProps<htmlElement> {
  className?: string;
  children: ReactNode;
  tag?: keyof JSX.IntrinsicElements;
}

export default function Grid({
  className = "",
  children,
  tag = "div",
  ...props
}: GridProps) {
  const Wrapper = tag;
```

Col.tsx

```
import { ReactNode, createElement } from "react";
import cn from "classnames";
import styles from "./Col.module.scss";

interface ColProps extends React.HTMLProps<htmlElement> {
  className?: string;
  children: ReactNode;
  colWidth?: "full" | "half" | "third" | "quarter";
  tag?: keyof JSX.IntrinsicElements;
}

export default function Col({
  className = "",
  children,
  colWidth,
  tag = "div",
```


Col.module.css

```
.full,  
.half,  
.third,  
.quarter {  
  grid-column: 1 / -1;  
}  
  
@media screen and (min-width: 640px) {  
  .quarter {  
    grid-column: span 4;  
  }  
}  
  
@media screen and (min-width: 900px) {  
  .half {  
    grid-column: span 6;  
  }  
}
```


CustomThingy.module.scss

```
p.customThingy {  
  grid-column: 1 / -1;  
  font-size: var( --step-1);  
}  
  
@media screen and (min-width: 640px) {  
  p.customThingy {  
    grid-column: 1 / 6;  
    padding-top: 2em;  
    padding-bottom: 1em;  
  }  
}  
  
@media screen and (min-width: 900px) {  
  p.customThingy {  
    grid-column: 1 / 7;  
  }  
}
```


Option 3: Using Tailwind classes

Yet Another Disclaimer

The following opinion may or may not oppose your view on the matter, and that is PERFECTLY FINE. You are absolutely free to agree, disagree or not care at all.

tailwind.config.js

```
module.exports = {  
  theme: {  
    screens: {  
      xs: "320px",  
      sm: "640px",  
      md: "900px",  
      lg: "1200px",  
      xl: "1600px",  
      maxSm: { max: "639px" },  
      maxMd: { max: "899px" },  
      btwSmMd: { min: "640px", max: "899px" }  
    },  
  },  
  prefix: "tw-",  
};
```

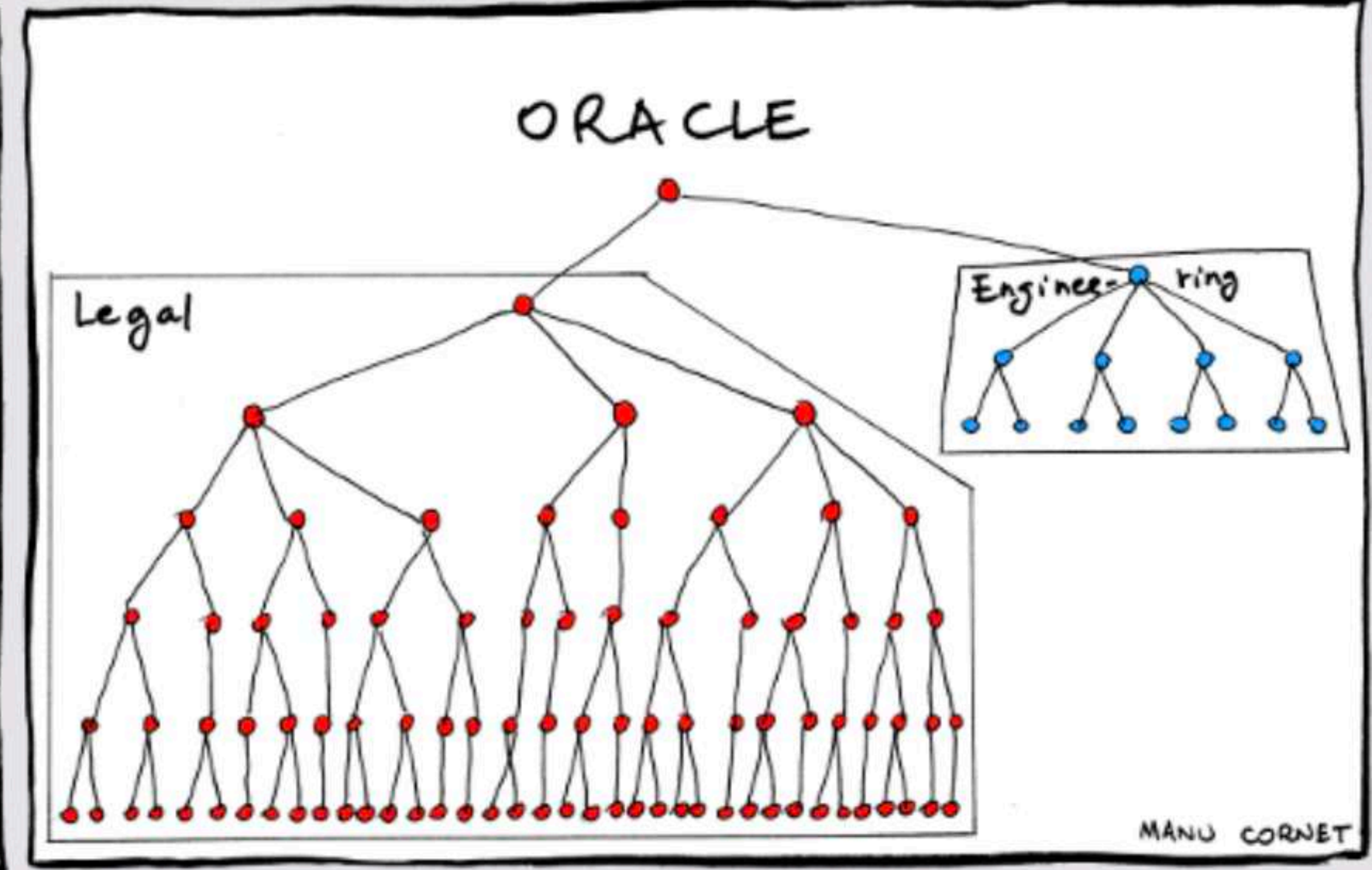
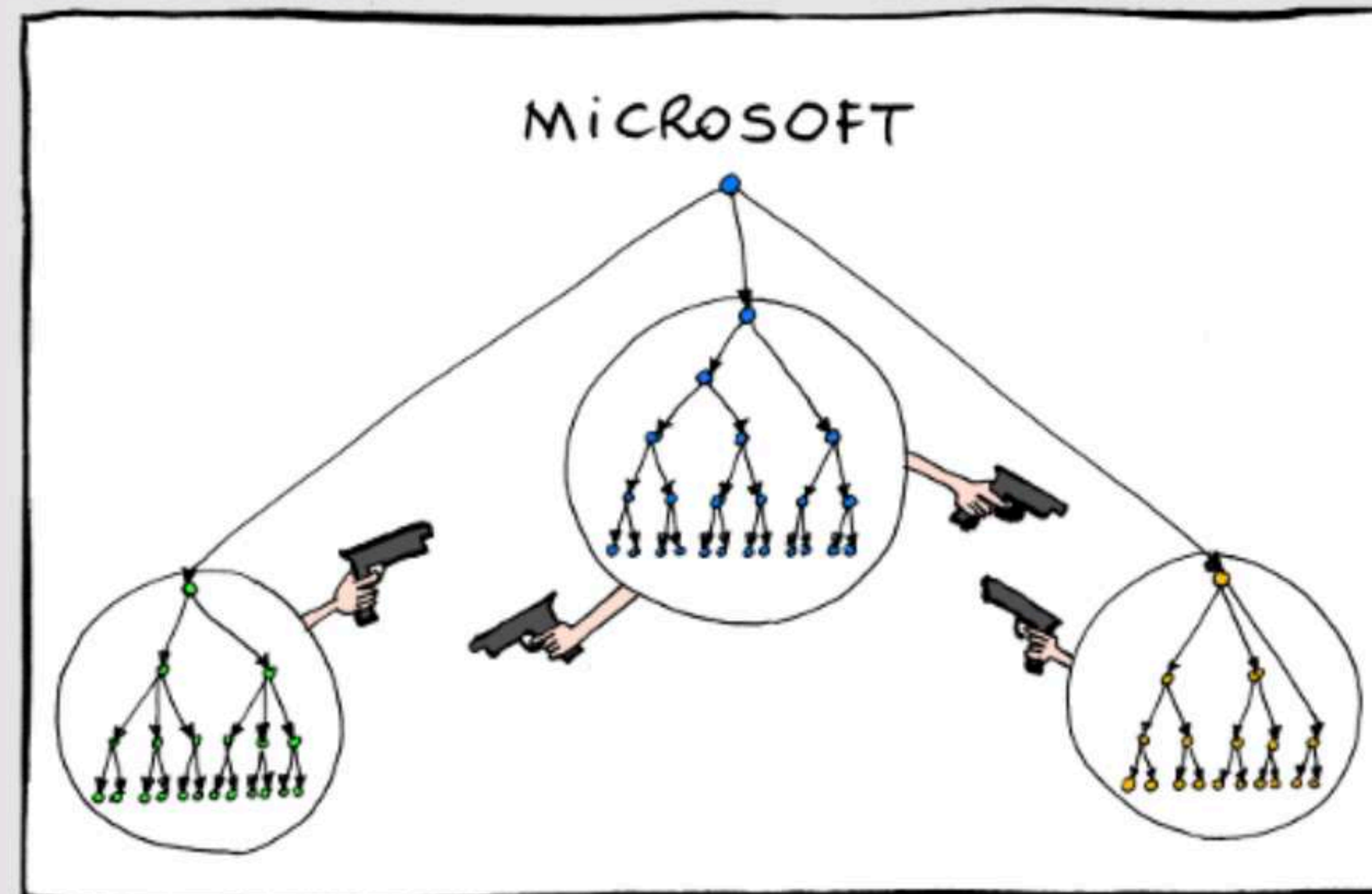
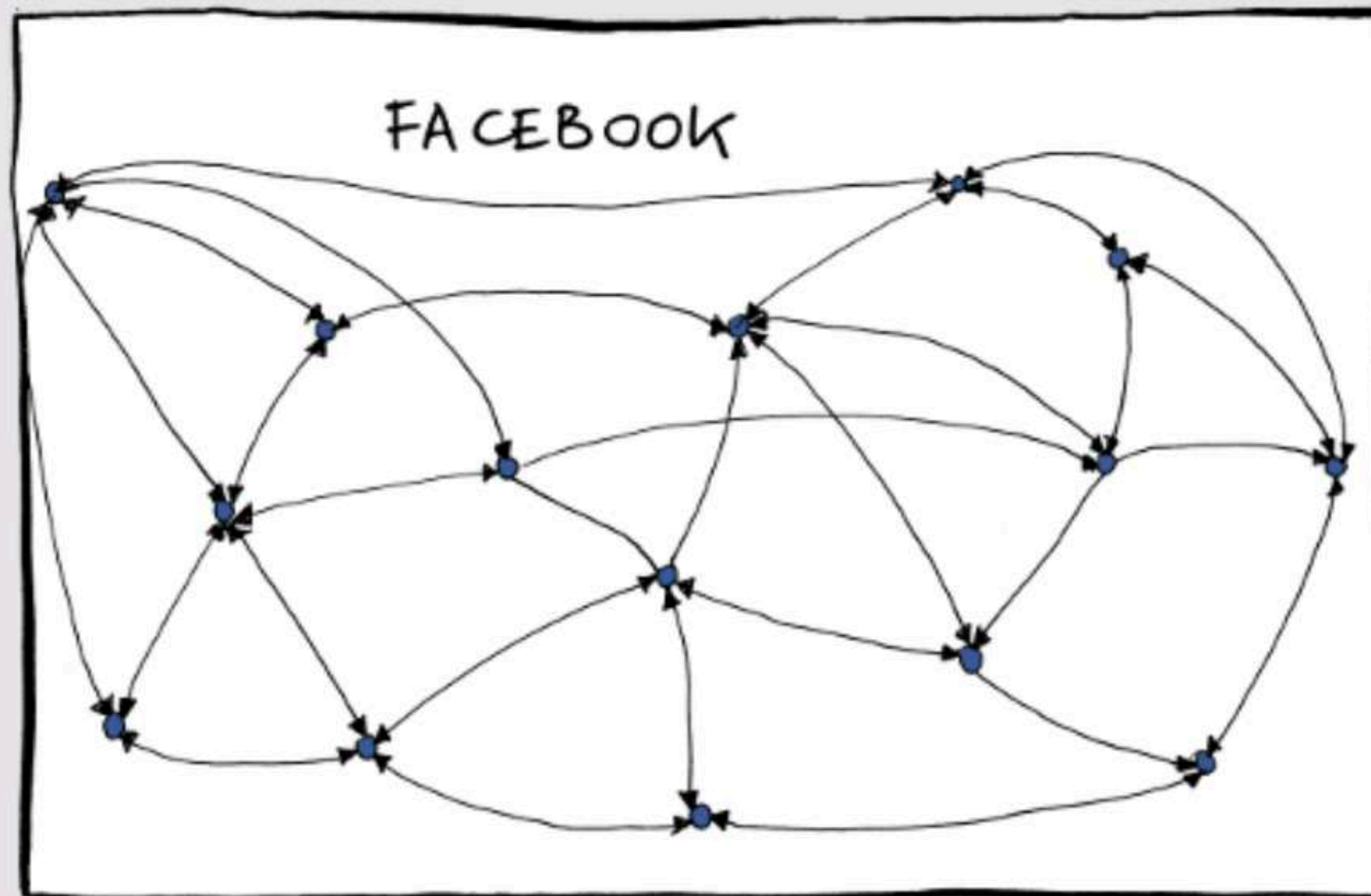
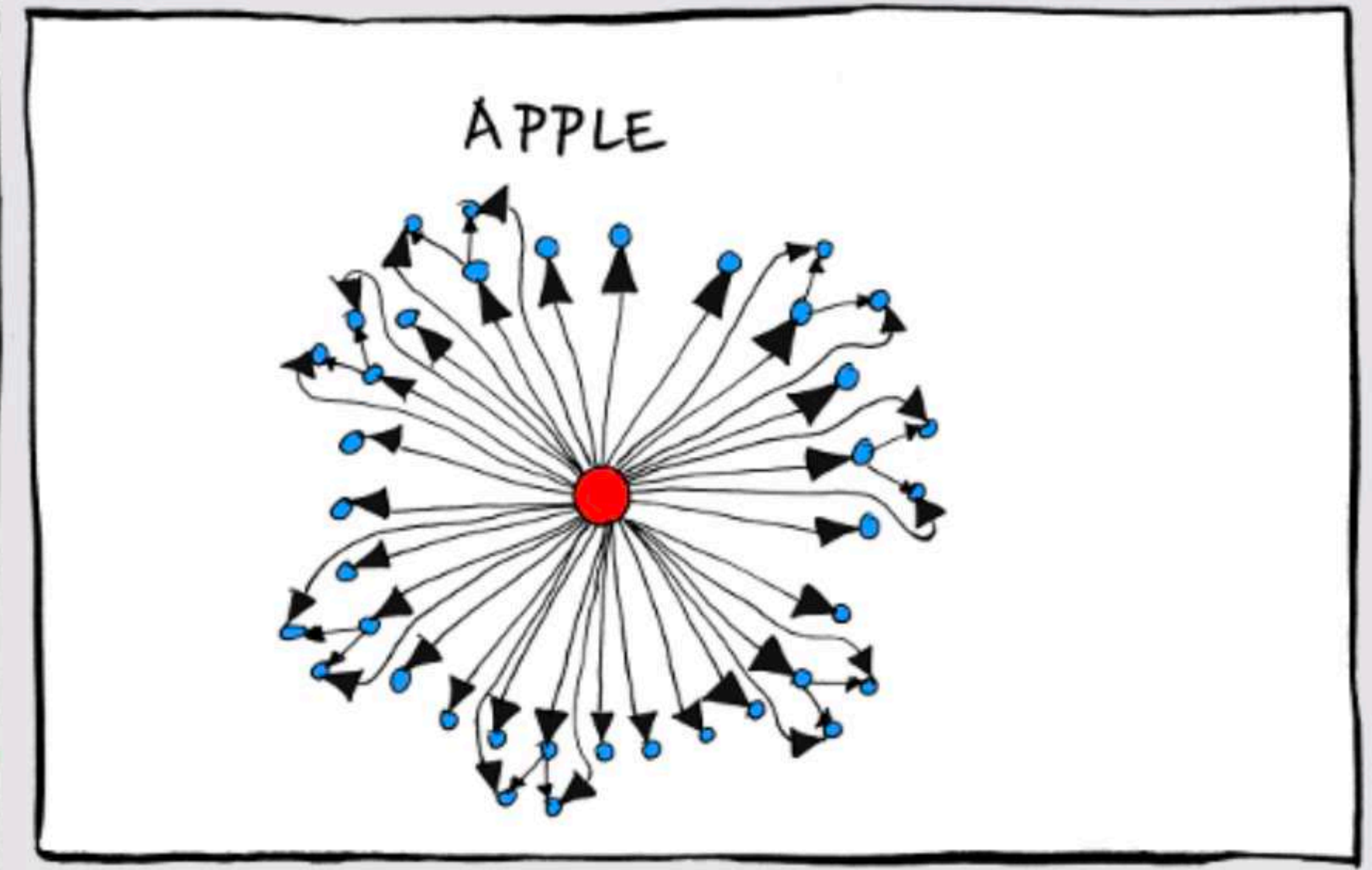
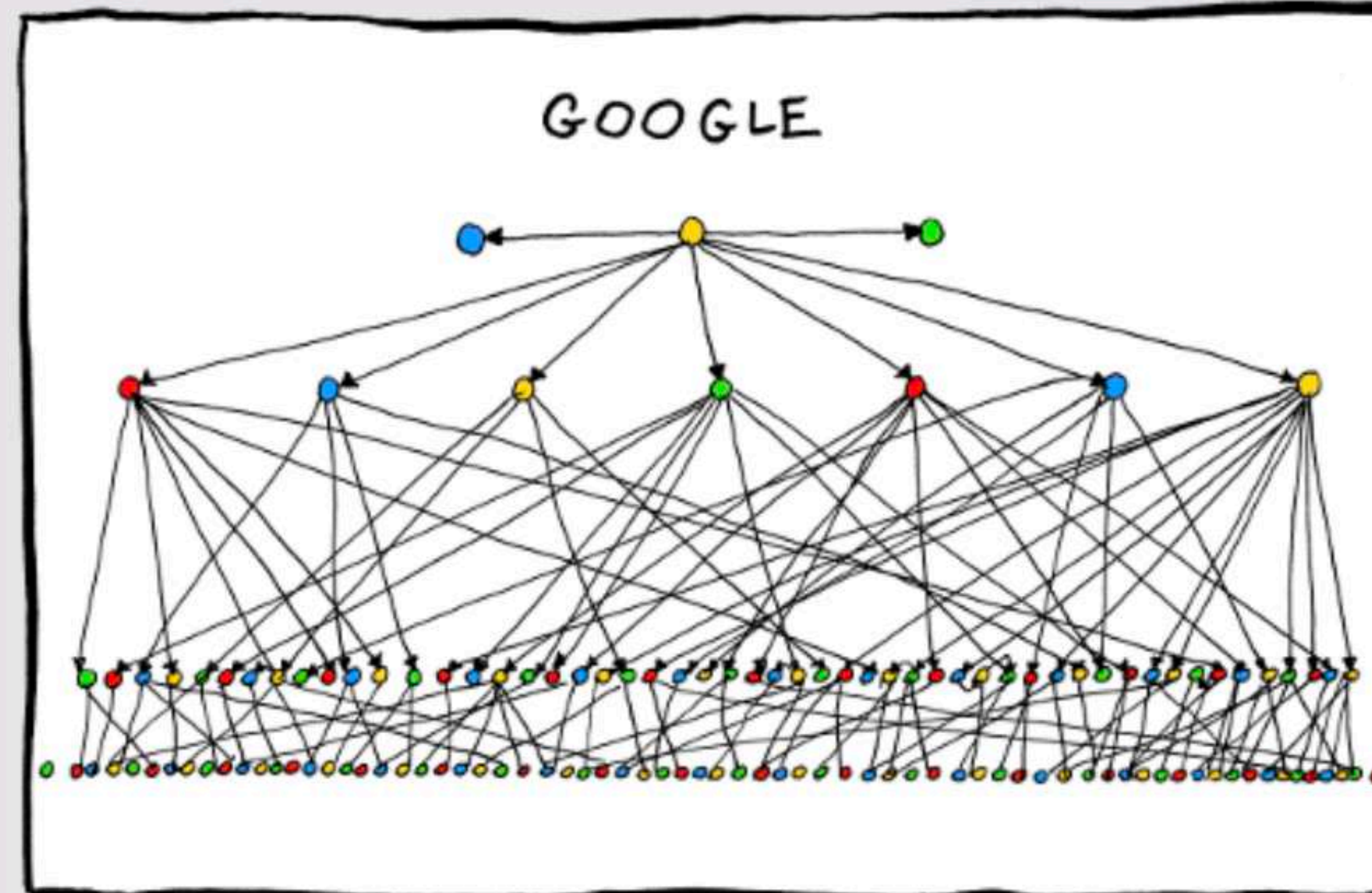
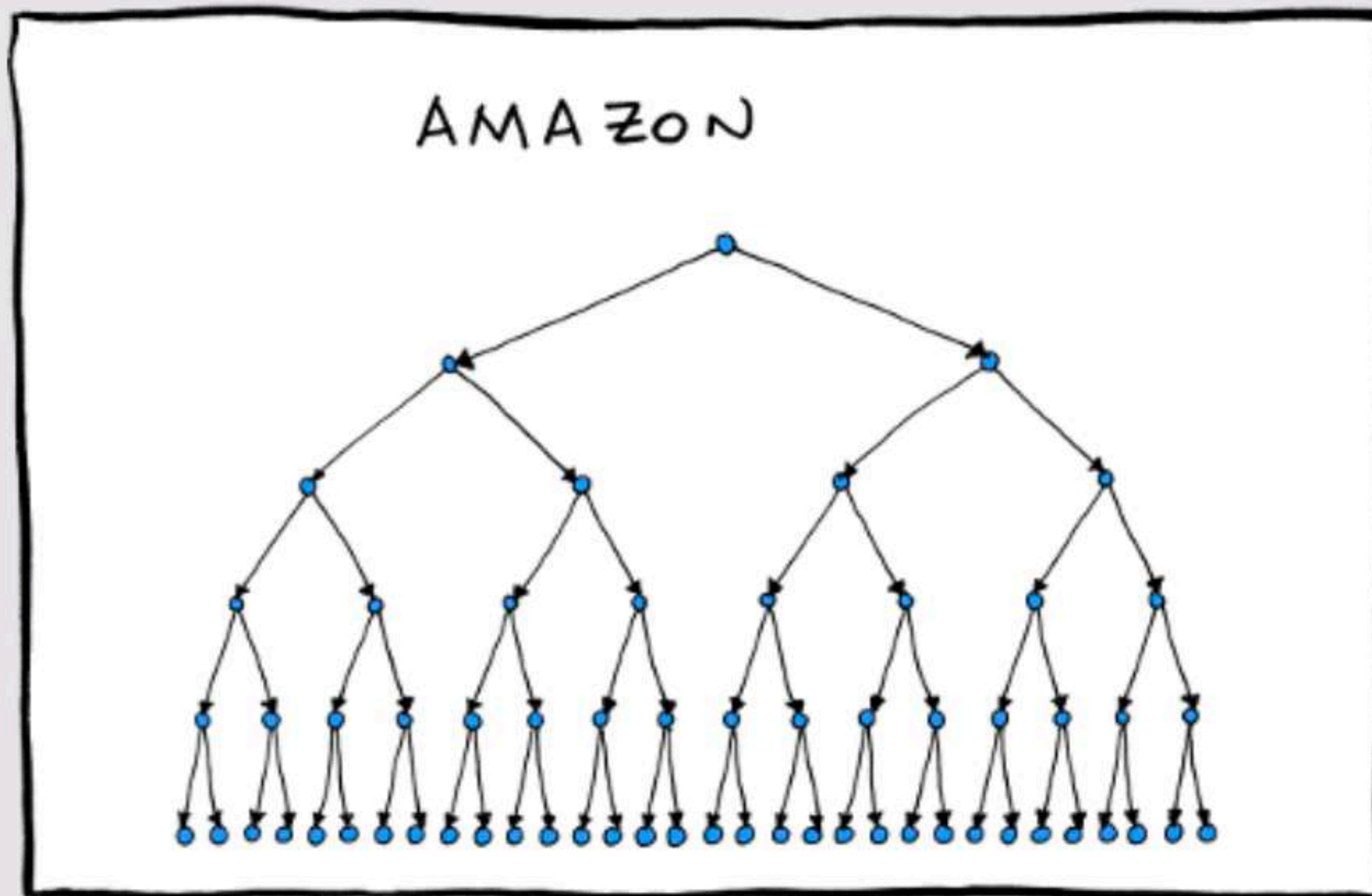

TailwindThingy.tsx

```
export default function TailwindThingy {  
  return (  
    <section classname="tw-grid xs:tw-grid-cols-4 sm:tw-grid-cols-8 md:tw-grid-cols-12 xs:tw-gap-3 lg:tw-gap-4  
xs:tw-mx-3 sm:tw-mx-[30px] md:tw-mx-[50px] lg:tw-mx-[90px] xl:tw-mx-[180px]">  
      <p classname="tw-col-span-full">Option 3: Use Tailwind classes</p>  
      <p classname="tw-col-span-full md:tw-col-span-6">Well, this is spicy</p>  
      <p classname="tw-col-span-full md:tw-col-span-6">  
        FWIW, Tailwind has managed to support grid fairly well in this latest  
        version  
      </p>  
      <p classname="tw-col-span-full md:tw-col-span-4">  
        You will have to learn the tailwind classes to use them correctly  
      </p>  
      <p classname="tw-col-span-full md:tw-col-span-4">  
        This basic example is able to match the previous 2 options  
      </p>  
    </section>  
  )  
}
```


But does it work?

CodeSandbox demo





Source: [Manu Cornet](#), 2011-06-27 edition of [Bonkers World](#) (modified to fit slide)

So you want to introduce Grid to your application?

- Are there **preferred technologies** used within the organisation?
- How **big** is your application and how is it **structured**?
- How **flexible** does the design system need to be?
- Are there cases where code is contributed by **new developers** often?
- What is the **documentation** culture like in your organisation?

So you want to introduce Grid to your application?

- Who is responsible for the **maintenance** and development of **new components or pages** on the application?
 - Is it a small team of full-time developers overseeing the entire project?
 - Is it numerous teams responsible for their own respective set of components and pages?
 - What is the overall CSS skill level of the developers contributing to the codebase?
 - Are the contributing developers very familiar with the frameworks and libraries used in the codebase?

Document the “Why”

“ *One size does not fit all* ”
–Frank Zappa

Thank you



<https://chenhuijing.com>



[@hj_chen](https://twitter.com/hj_chen)



[@huijing](https://github.com/huijing)



[@huijing](https://discord.com/users/huijing)

Font is [Jones](#) by [indestructible type](#)*.