Architecting CSS grid in a design system

By Chen Hui Jing / @hj_chen
Screens, screens, screens

Image source: Inch Calculator
```css
.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

<table>
<thead>
<tr>
<th>Sun</th>
<th>Mon</th>
<th>Tue</th>
<th>Wed</th>
<th>Thu</th>
<th>Fri</th>
<th>Sat</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>6</td>
<td><img src="https://www.mozilla.org/en-US/fenrir/" alt="Firefox" /></td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
</tr>
<tr>
<td>12</td>
<td>13</td>
<td><img src="https://www.google.com/chrome/" alt="Chrome" /></td>
<td>15</td>
<td>16</td>
<td>17</td>
<td>18</td>
</tr>
<tr>
<td>19</td>
<td>20</td>
<td><img src="https://www.opera.com/" alt="Opera" /></td>
<td>22</td>
<td>23</td>
<td>24</td>
<td>25</td>
</tr>
<tr>
<td>26</td>
<td><img src="https://www.apple.com/safari/" alt="Safari" /></td>
<td>28</td>
<td>29</td>
<td>30</td>
<td>31</td>
<td></td>
</tr>
</tbody>
</table>

### October 2017

<table>
<thead>
<tr>
<th>Sun</th>
<th>Mon</th>
<th>Tue</th>
<th>Wed</th>
<th>Thu</th>
<th>Fri</th>
<th>Sat</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
</tr>
<tr>
<td>15</td>
<td>16</td>
<td><img src="https://www.electronjs.org/" alt="E" /></td>
<td>18</td>
<td>19</td>
<td>20</td>
<td>21</td>
</tr>
<tr>
<td>22</td>
<td>23</td>
<td>24</td>
<td>25</td>
<td>26</td>
<td>27</td>
<td>28</td>
</tr>
<tr>
<td>29</td>
<td>30</td>
<td><img src="https://www.microsoft.com/en-us/edge/" alt="Edge" /></td>
<td></td>
<td>25</td>
<td>26</td>
<td>27</td>
</tr>
<tr>
<td>29</td>
<td>30</td>
<td><img src="https://www.microsoft.com/en-us/edge/" alt="Edge" /></td>
<td>25</td>
<td>26</td>
<td>27</td>
<td>28</td>
</tr>
</tbody>
</table>
**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'.

<table>
<thead>
<tr>
<th>IE</th>
<th>Edge</th>
<th>Firefox</th>
<th>Chrome</th>
<th>Safari</th>
<th>iOS Safari</th>
<th>Opera Mini</th>
<th>Chrome for Android</th>
<th>Android Browser</th>
<th>Samsung Internet</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>102</td>
<td>102</td>
<td>102</td>
<td>15.4</td>
<td>15.4</td>
<td>all</td>
<td>104</td>
<td>104</td>
<td>16.0</td>
</tr>
<tr>
<td>10</td>
<td>103</td>
<td>103</td>
<td>103</td>
<td>15.5</td>
<td>15.5</td>
<td>all</td>
<td>104</td>
<td>104</td>
<td>17.0</td>
</tr>
<tr>
<td>11</td>
<td>104</td>
<td>104</td>
<td>104</td>
<td>15.6</td>
<td>15.6</td>
<td>all</td>
<td>104</td>
<td>104</td>
<td>18.0</td>
</tr>
</tbody>
</table>

Data from caniue.com | Embed from caniue.bitsofco.de

**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.

<table>
<thead>
<tr>
<th>IE</th>
<th>Edge</th>
<th>Firefox</th>
<th>Chrome</th>
<th>Safari</th>
<th>iOS Safari</th>
<th>Opera Mini</th>
<th>Chrome for Android</th>
<th>Android Browser</th>
<th>Samsung Internet</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>102</td>
<td>102</td>
<td>102</td>
<td>15.4</td>
<td>15.4</td>
<td>all</td>
<td>104</td>
<td>104</td>
<td>16.0</td>
</tr>
<tr>
<td>10</td>
<td>103</td>
<td>103</td>
<td>103</td>
<td>15.5</td>
<td>15.5</td>
<td>all</td>
<td>104</td>
<td>104</td>
<td>17.0</td>
</tr>
<tr>
<td>11</td>
<td>104</td>
<td>104</td>
<td>104</td>
<td>15.6</td>
<td>15.6</td>
<td>all</td>
<td>104</td>
<td>104</td>
<td>18.0</td>
</tr>
</tbody>
</table>

Data from caniue.com | Embed from caniue.bitsofco.de

Global: 98.73% • 0.95% = 99.68%
State of CSS 2021 survey

Flexbox: 98.9%


Grid: 83.5%
Good evening

RENAISSANCE

This Is CHVRCHES

This Is girl in red

Today's Top Hits

Your top mixes

Lizzo Mix
Lizzo Mix
Beyoncé, Tinase, Cardi B and more

Beyoncé Mix
Tems, Lizzo, City Girls and more

Hebe Tien Mix
Sam Lee, Weibird, Jay Chou and more

2010s Mix
Tones And I, Lizzo, Hayley Kiyoko and more

80s Mix
a-ha, James Ingram, CoCo Lee and more

2000s Mix
Rene Liu, Sam Lee and more

Today's biggest hits

Singapore

It's a Hit!

K-Pop

X-Games
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.
Chronos the Cat's Anime Adventure Episodes and Stories

My Storylines | Other Episodes

Like a Round Robin, different authors will take the story in different directions. If you're inclined to Anime fan fiction adventures (and who isn't?) you'll want to check this out. The rest of the shops can be found here.

---

Neopia Central

Click on the shops 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!

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, N64, Game Boy, Shadowrun (Adx), Way Of The Warrior (Adx), 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 personal 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, N64 and Doom fans may find some interesting links too. Pinball players will find the best info for their favorite pinball 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 page! 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 FAQ: ftp server: Mortal Komber 1 and 2, NBA Jam, Art of Fighting, lots of info about real arcade games and console/computer games. Some of the files available there in ASCII can 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.
- vansonic.unc.edu. Sega stuff only, including FAQs, electronic publications etc.
- ftp.es.ucd.edu. 3DO material, mostly lots of game screenshows in JPEG format.
- ruffin.mit.edu. For video game newsarchive archives.

Source: Michel Buffa's Video Games Page in 1994
Parent-child relationship

Flex/Grid container

Flex/Grid item  Flex/Grid item  Flex/Grid item
Basic grid syntax

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

## Banner

<table>
<thead>
<tr>
<th>Links and stuff?</th>
<th>Your main content</th>
</tr>
</thead>
</table>

## Footer

Footer, for copyright and moar links?

```css
.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;
}
Common example of grid system CSS

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

<table>
<thead>
<tr>
<th>Size</th>
<th>Min</th>
<th>Max</th>
<th>Cols</th>
<th>Margin</th>
<th>Gutter</th>
</tr>
</thead>
<tbody>
<tr>
<td>xs</td>
<td>320px</td>
<td>639px</td>
<td>4</td>
<td>16px</td>
<td>16px</td>
</tr>
<tr>
<td>sm</td>
<td>640px</td>
<td>899px</td>
<td>8</td>
<td>30px</td>
<td>16px</td>
</tr>
<tr>
<td>md</td>
<td>900px</td>
<td>1199px</td>
<td>12</td>
<td>50px</td>
<td>16px</td>
</tr>
<tr>
<td>lg</td>
<td>1200px</td>
<td>1599px</td>
<td>12</td>
<td>90px</td>
<td>24px</td>
</tr>
<tr>
<td>xl</td>
<td>1600px</td>
<td>-</td>
<td>12</td>
<td>&gt;180px</td>
<td>24px</td>
</tr>
</tbody>
</table>
```
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}`
    },
    children
  );
}
```
Option 1: vanilla CSS (or SCSS)

<table>
<thead>
<tr>
<th>Size</th>
<th>Min</th>
<th>Max</th>
<th>Cols</th>
<th>Margin</th>
<th>Gutter</th>
</tr>
</thead>
<tbody>
<tr>
<td>xs</td>
<td>320px</td>
<td>639px</td>
<td>4</td>
<td>16px</td>
<td>16px</td>
</tr>
<tr>
<td>sm</td>
<td>640px</td>
<td>899px</td>
<td>8</td>
<td>30px</td>
<td>16px</td>
</tr>
<tr>
<td>md</td>
<td>900px</td>
<td>1199px</td>
<td>12</td>
<td>50px</td>
<td>16px</td>
</tr>
<tr>
<td>lg</td>
<td>1200px</td>
<td>1599px</td>
<td>12</td>
<td>90px</td>
<td>24px</td>
</tr>
<tr>
<td>xl</td>
<td>1600px</td>
<td>-</td>
<td>12</td>
<td>&gt;180px</td>
<td>24px</td>
</tr>
</tbody>
</table>

```css
.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)

```css
.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)

```css
.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
Grid.tsx

```typescript
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 || "div";
    // code...
}
```

Col.tsx

```typescript
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"
}: ColProps) {
    // code...
}
```
.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

```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.
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-"
  }
};
export default function TailwindThingy () {
  return (
      <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
SO MANY CHOICES...
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 is 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

@huijing

Font is Jones by indestructible type®.