

When new CSS features collide

Possibility and Complexity at the Intersections

Rachel Andrew, Google

When new CSS features collide

Possibility and Complexity at the Intersections

Rachel Andrew, Google

Floats

Positioning

Media queries



Floats

Positioning

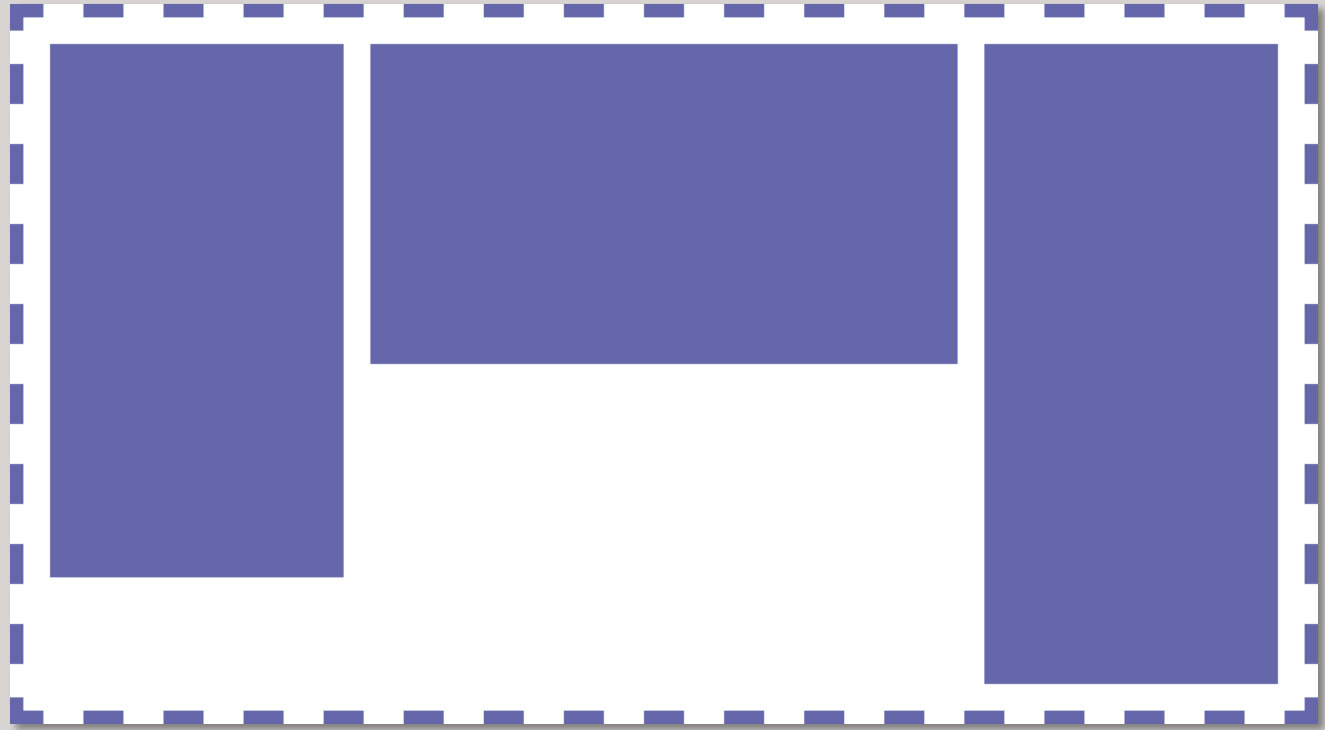
Media queries

Flexbox

Grid layout

Container queries

Cascade layers



“

If I would have asked people what they wanted, they would have said faster horses.

”

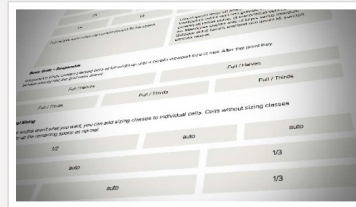
(Probably not) Henry Ford.



A floated item is out of flow, it has no knowledge of the other boxes in the layout.

In flex layout, items are treated as a group.

Showcase



Better, Simpler Grid Systems

Flexbox gives us most of the features we want from a grid system out of the box. And sizing and alignment are just one or two properties away.



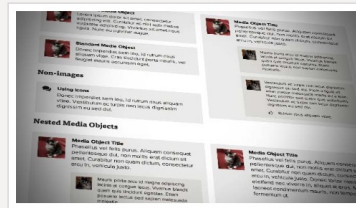
Holy Grail Layout

This classic problem has been challenging CSS hackers for years, yet none of the historical solutions have fully solved it. With Flexbox, it's finally possible.



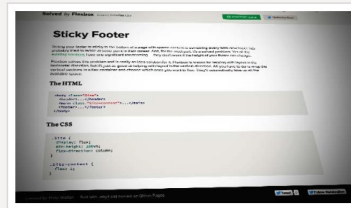
Input Add-ons

Creating full-width, fluid input/button pairs has been impossible for most of the history of CSS. With Flexbox it couldn't be easier.



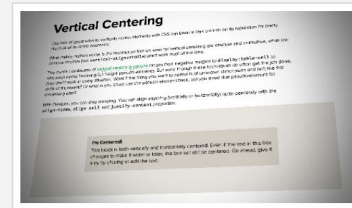
Media Object

Create media objects with fixed or varying figure sizes without worrying about overflow, clearfixing, or block formatting context hacks.



Sticky Footer


Getting your footer to stick to the bottom of sparsely contented pages has always been tricky. And if the footer's height is unknown, it's basically impossible. Not so anymore.



Vertical Centering

This classic problem has been challenging CSS hackers for years, yet none of the historical solutions have fully solved it. With Flexbox, it's finally possible.

24 WAYS to impress your friends



Giving Content Priority with CSS3 Grid Layout

Rachel Andrew

18 December 2012

Published in [Code](#)

[7 comments](#)

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



Figure 1. Four regions, called a, b, c and d, each receive a part of a document



```
body {  
  display: "aaa"  
          "bcd"  
}
```

```
#head { position: a }  
#nav  { position: b }  
#adv  { position: c }  
#body { position: d }
```




```
body {  
  display: grid;  
  grid-template-areas: "a a a"  
                      "b c d"  
}
```

```
#head { grid-area: a }  
#nav  { grid-area: b }  
#adv  { grid-area: c }  
#body { grid-area: d }
```

“

I wanted to get it out there and get other people to have a look at it.

”

Me, in [The Story of CSS Grid, from Its Creators](#)

Why use grid instead of flexbox?



Never heard of it



Know about it

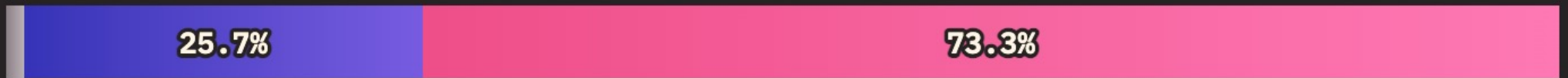


Have used it

2019



2020



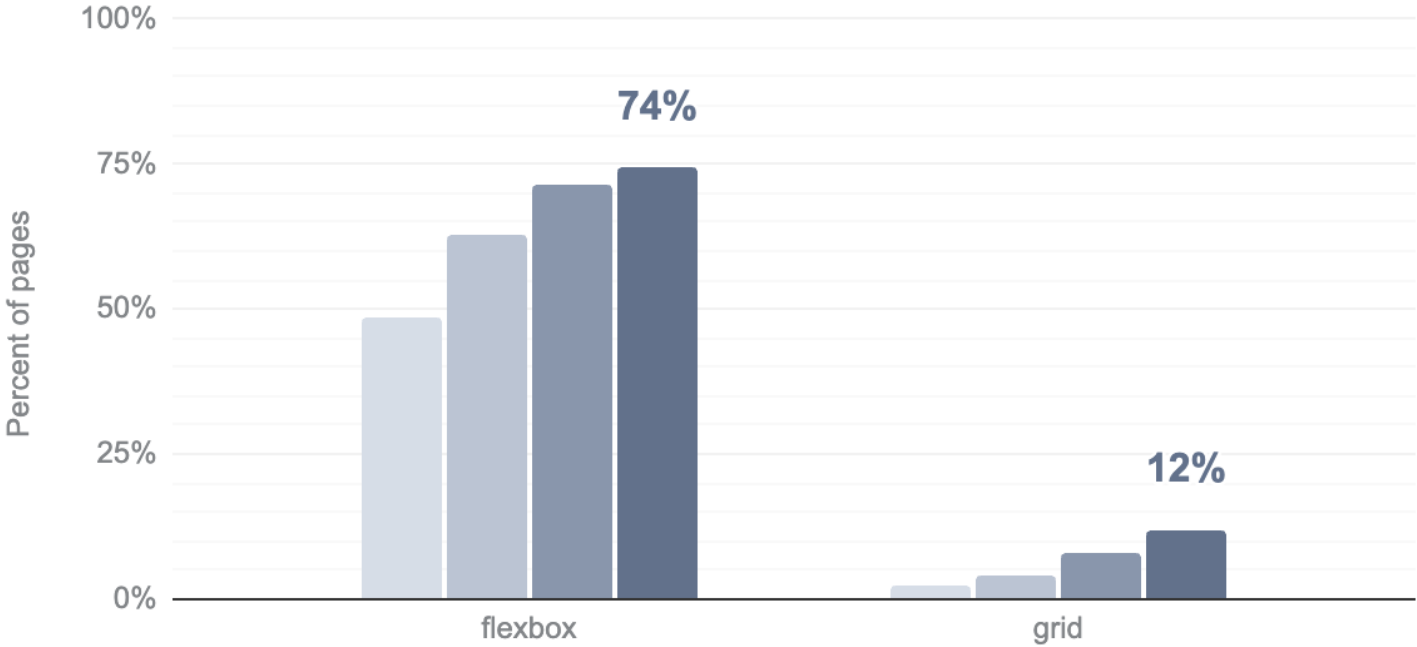
2021



Flexbox and Grid adoption by year

Web Almanac 2022: CSS (mobile)

2019 2020 2021 2022



Two years after grid shipped in the three major engines, only 2% of sites were using grid layout.

Flexbox provided the faster horses.



Container Queries

Querying against properties of the container, rather than the viewport.




2013 February 7:  [Jon Neal's \(now !\[\]\(b4746ffbe3ffe6750d2dbafa2ea7d9e5_img.jpg\) deleted\) *Thoughts on Media Queries for Elements*](#)

- via   [Jay Hoffman](#)

2013 February 7:  [Nicolas Gallagher's \(now !\[\]\(c9cd5a1c35167a83f09a35036fe5dcbd_img.jpg\) deleted\) tweet](#)

- via   [Eric Portis](#) and   [Jay Hoffman](#)

2012 April 21:   [Jon Neal's MediaClass](#)

2012 January 23:   [Paul Irish's tweet reply to a \(now !\[\]\(63ea948177b1bcc486b2b76d20d5fb69_img.jpg\) deleted\)](#)
 [Ian Storm Taylor tweet \(date unknown\)](#)

2011 September 1: [A question posted by user Damon on Stack Overflow](#)

- via   [Jay Hoffman](#)

2011 July 15:  [Andy Hume's \(now !\[\]\(f419710cbe076aa30a9c6c031b5cbe84_img.jpg\) deleted\) *Responsive Containers* blog post](#)

- via  [Eric Portis' Contain Your Excitement talk](#)

2011 July 14:   [Andy Hume's selector-queries](#)



**G-CIND at the Bristol Balloon
Fiesta 2022**

After two years the Bristol Balloon Fiesta was back. Most of Bristol could be found trekking up the hill at 5.30am to worship their strange hot-air demanding gods.



G-CIND at the Bristol Balloon Fiesta 2022

After two years the Bristol Balloon Fiesta was back. Most of Bristol could be found trekking up the hill at 5.30am to worship their strange hot-air demanding gods.



G-CIND at the Bristol Balloon Fiesta 2022

After two years the Bristol Balloon Fiesta was back. Most of Bristol could be found trekking up the hill at 5.30am to worship their strange hot-air demanding gods.

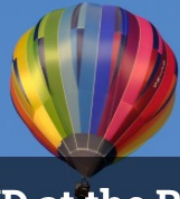


```
.element {  
  container-type: inline-size;  
}
```



```
@container (min-width: 500px) {
```

```
}
```



G-CIND at the Bristol Balloon Fiesta 2022

After two years the Bristol Balloon Fiesta was back. Most of Bristol could be found trekking up the hill at 5.30am to worship their strange hot-air demanding gods.



G-CIND at the Bristol Balloon Fiesta 2022

After two years the Bristol Balloon Fiesta was back. Most of Bristol could be found trekking up the hill at 5.30am to worship their strange hot-air demanding gods.

Why did this take so long?

It seems really simple!

GitHub - dbaron/container-queries-
github.com/dbaron/container-queries-implementability

Product Solutions Open Source Pricing Search Sign in Sign up

dbaron / container-queries-implementability Public Notifications Fork 1 Star 45

Code Issues 3 Pull requests Actions Projects Security Insights

main 1 branch 0 tags Go to file Code

dbaron Put in Florian's major issue. dd8c8ed on Apr 29, 2020 23 commits

README.md Put in Florian's major issue. 3 years ago

README.md

Thoughts on an implementable path forward for Container Queries

The idea of Container Queries has been discussed widely, such as in [WICG/container-queries](#) and in [w3c/csswg-drafts#3852](#). CSS authors want a way to use a feature like media queries on a *part* of the page. Many of the use cases underlying this desire relate to things that are component-like, that is, pieces of a web page that might be used in different webpages. Authors wish to construct a component whose appearance responds to its size, but they want this component to be a part of the webpage that contains it.

Dependence on containment

The main reason this is theoretically difficult is that this requires that styles depend on the size of the component, yet given how CSS works, the styles in a component *influence* its size. Arbitrarily breaking this loop would both give weird results and would interfere with browser optimizations related to incremental layout.

About

Proposal for container queries designed to be implementable in web browsers

- Readme
- 45 stars
- 7 watching
- 1 fork

Releases

No releases published

Packages

No packages published



```
.element {  
  contain: size;  
}
```

**Soup avocado basil
pesto delightful
blueberry scones
cashew macadamia
nut cookies potato
salty hemp seeds
street style Thai
basil tacos cozy
butternut dark
chocolate couscous
peaches mocha
chocolate sweet
potato black bean
burrito crunchy
avocado.**

**Soup avocado basil
pesto delightful
blueberry scones
cashew macadamia
nut cookies potato
salty hemp seeds
street style Thai
basil tacos cozy
butternut dark
chocolate couscous
peaches mocha
chocolate sweet
potato black bean
burrito crunchy
avocado.**

“

I believe that container queries should be possible if an element has both layout containment and has size containment in the axis used for container queries. The problem here is that we don't have a definition of size containment in a single axis.

”

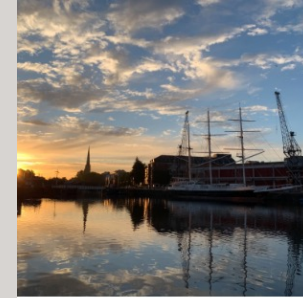
David Baron



Container queries + aspect-ratio



```
.fig-panel {  
  display: grid;  
  gap: 10px;  
  grid-template-columns: 1fr 1fr;  
}  
  
.fig-panel :nth-child(1) {  
  grid-column: 1;  
  grid-row: 1;  
}  
  
.fig-panel :nth-child(2) {  
  grid-column: 2;  
  grid-row: 1;  
}  
  
.fig-panel :nth-child(3) {  
  grid-column: 1/3;  
  grid-row: 2 /4;  
}  
  
.fig-panel img {  
  aspect-ratio: 1/1;  
  height: 100%;  
  width: 100%;  
  object-fit: cover;  
}  
  
.fig-panel .caption {  
  grid-row: 3;  
  grid-column: 1 / 3;  
}
```



Bristol is a pretty nice place to live.

```
@container (min-width: 700px) {
  .fig-panel {
    grid-template-columns: repeat(3, 1fr);
    grid-template-rows: 1fr auto;
    padding: 40px;
    margin: 0;
    list-style: none;
  }

  .fig-panel :nth-child(1) {
    grid-column: 1;
    grid-row: 1 / 3;
  }

  .fig-panel :nth-child(2) {
    grid-column: 2;
    grid-row: 1 / 3;
  }

  .fig-panel :nth-child(3) {
    grid-column: 3;
    grid-row: 1 / 3;
  }

  .fig-panel .caption {
    grid-column: 1 / -1;
    grid-row: 2;
  }
}
```



Bristol is a pretty nice place to live.



A parent selector

has()



```
<ul>
  <li>I do not contain any headings</li>
  <li><p>I contain a paragraph.</p></li>
  <li><h2>I'm a level 2 heading!</h2></li>
  <li><h2>I'm a level 2 heading!</h2>
    <p>Followed by a paragraph.</p>
  </li>
</ul>
```



```
li:has(h2) {
```

```
}
```



```
li:has(h2+p) {
```

```
}
```



An article about balloons I couldn't be bothered to find an image for.



Visit this excellent article about balloons.



Visit this excellent article about balloons.



Visit this excellent article about balloons.

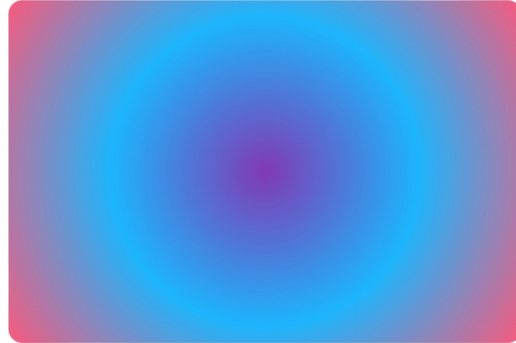
Sorry, all out of balloon pictures.



```
li:not(:has(img)):before{  
  content: "";  
  aspect-ratio: 9/6;  
  background: rgb(131,58,180);  
  background: radial-gradient(circle, rgba(131,58,180,1) 0%,  
  rgba(29,182,253,1) 50%, rgba(252,69,108,.9) 100%);  
  border-radius: .5em;  
}
```



[Read this excellent article about balloons.](#)



[An article about balloons I couldn't be bothered to find an image for.](#)



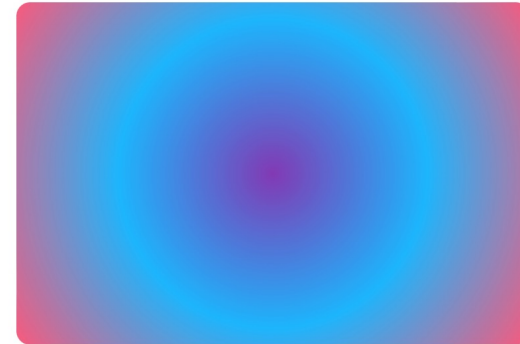
[Visit this excellent article about balloons.](#)



[Visit this excellent article about balloons.](#)



[Visit this excellent article about balloons.](#)



[Sorry, all out of balloon pictures.](#)



has() + container queries

What's inside the component + how much space there is to play with.



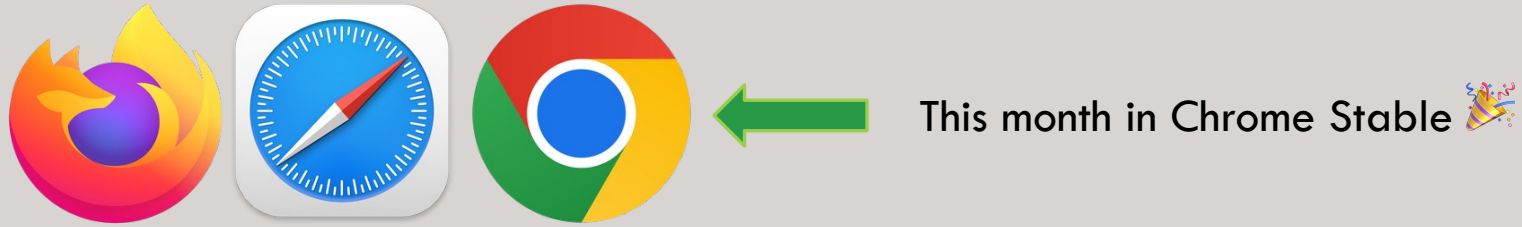
```
@container (min-width: 500px) {  
  .card:has(figure+.content) {  
    display: grid;  
    grid-template-columns: .8fr 1fr;  
    gap: 10px;  
  }  
}
```




**G-CIND at the Bristol Balloon
Fiesta 2022**

After two years the Bristol Balloon Fiesta was back. Most of Bristol could be found trekking up the hill at 5.30am to worship their strange hot-air demanding gods.

After two years the Bristol Balloon Fiesta was back. Most of Bristol could be found trekking up the hill at 5.30am to worship their strange hot-air demanding gods.



subgrid

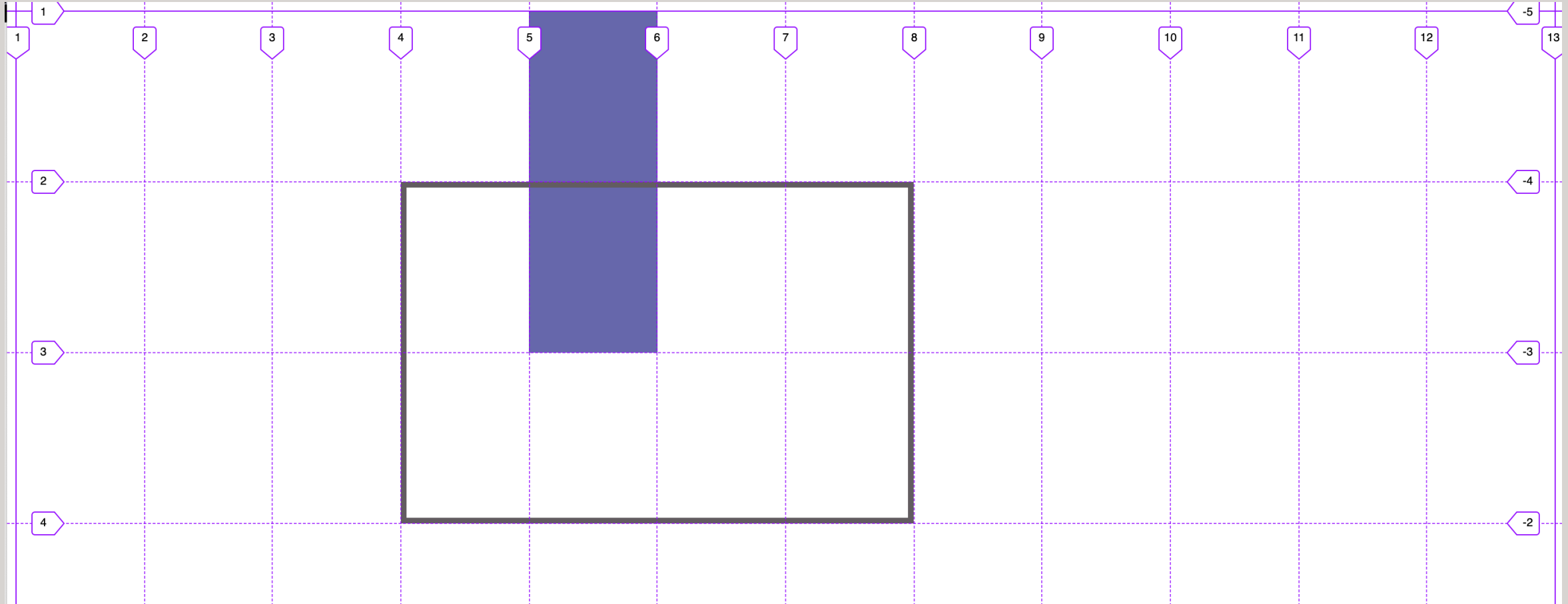
Inherit the size and number of tracks from a parent into a child grid.



```
.child-grid {  
  display: grid;  
  grid-template-columns: subgrid;  
  grid-template-rows: subgrid;  
}
```



```
.main-grid {  
  display: grid;  
  grid-template-columns: repeat(12, 1fr);  
  grid-template-rows: repeat(4, minmax(150px, auto));  
}  
  
.child-grid {  
  grid-column: 4 / 8;  
  grid-row: 2 / 4;  
  display: grid;  
  grid-template-columns: subgrid;  
  grid-template-rows: subgrid;  
}
```



Inspector Console Debugger Network Style Editor Performance Memory Storage Accessibility Application

Search HTML + Filter Styles :hov .cls + [Icons]

```
<body translate="no">
  <div class="main-grid"> grid overflow
    <div class="item grid-item"></div>
    <div class="child-grid"> subgrid
      <div class="item child-grid-item"></div>
    </div>
</body>
</html>
```

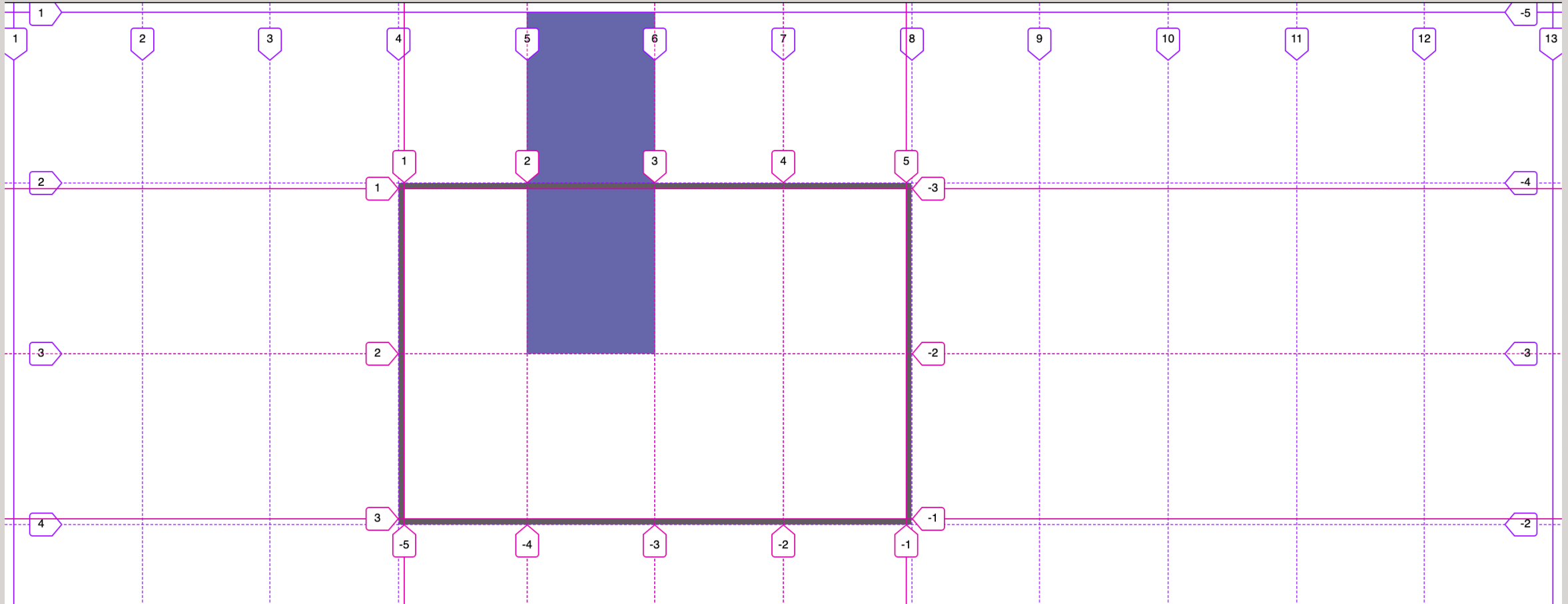
element { inline
}

.child-grid { inline:8
border: 5px solid #625c60;
grid-column: 4 / 8;
grid-row: 2 / 4;
display: grid;
grid-template-columns: subgrid;
grid-template-rows: subgrid;

Layout Computed Changes Compatibility

- Flexbox
Select a Flex container or item to continue.
- Grid
Overlay Grid
 - div.main-grid
 - div.child-grid

rame-wrap > iframe#result.result-iframe > html > body > div.main-grid > div.child-grid > div.item.child-grid-item > }



Inspector Console Debugger Network Style Editor Performance Memory Storage Accessibility Application

Search HTML

```
<body translate="no">  
  <div class="main-grid"> grid overflow  
    <div class="item grid-item"></div>  
    <div class="child-grid"> subgrid  
      <div class="item child-grid-item"></div>  
    </div>  
</body>  
</html>
```

Filter Styles

```
element {  
  }  
.child-grid {  
  border: 5px solid #625c60;  
  grid-column: 4 / 8;  
  grid-row: 2 / 4;  
  display: grid;  
  grid-template-columns: subgrid;  
  grid-template-rows: subgrid;  
}
```

Layout Computed Changes Compatibility

- Flexbox
 - Select a Flex container or item to continue.
- Grid
 - Overlay Grid
 - div.main-grid
 - div.child-grid

rame-wrap > iframe#result.result-iframe > html > body > div.main-grid > div.child-grid > div.item.child-grid-item >



```
.grid-item {  
  grid-column: 5;  
}
```

```
.child-grid-item {  
  grid-column: 2;  
}
```



```
.main-grid {  
  display: grid;  
  grid-template-columns: [a] 1fr [b] 1fr [c] 1fr [d] 1fr [e]  
1fr [f] 1fr [g] 1fr [h] 1fr [i] 1fr [j] 1fr [k] 1fr [l] 1fr  
[m];  
  grid-template-rows: repeat(4, minmax(150px, auto));  
}  
  
.item {  
  grid-column: e;  
}
```


The value of subgrid replaces the track listing

You can subgrid in one dimension or both.

One proposal was to simplify subgrid by locking it to both dimensions.

This would mean there was no implicit grid, no way to subgrid columns but have as many rows as required for the content.

To use a subgrid you would need to know exactly how many columns and rows the component would have.

CSS requires playing a long game

The single-axis decision slowed progress and implementation of subgrid, but it was the right decision.

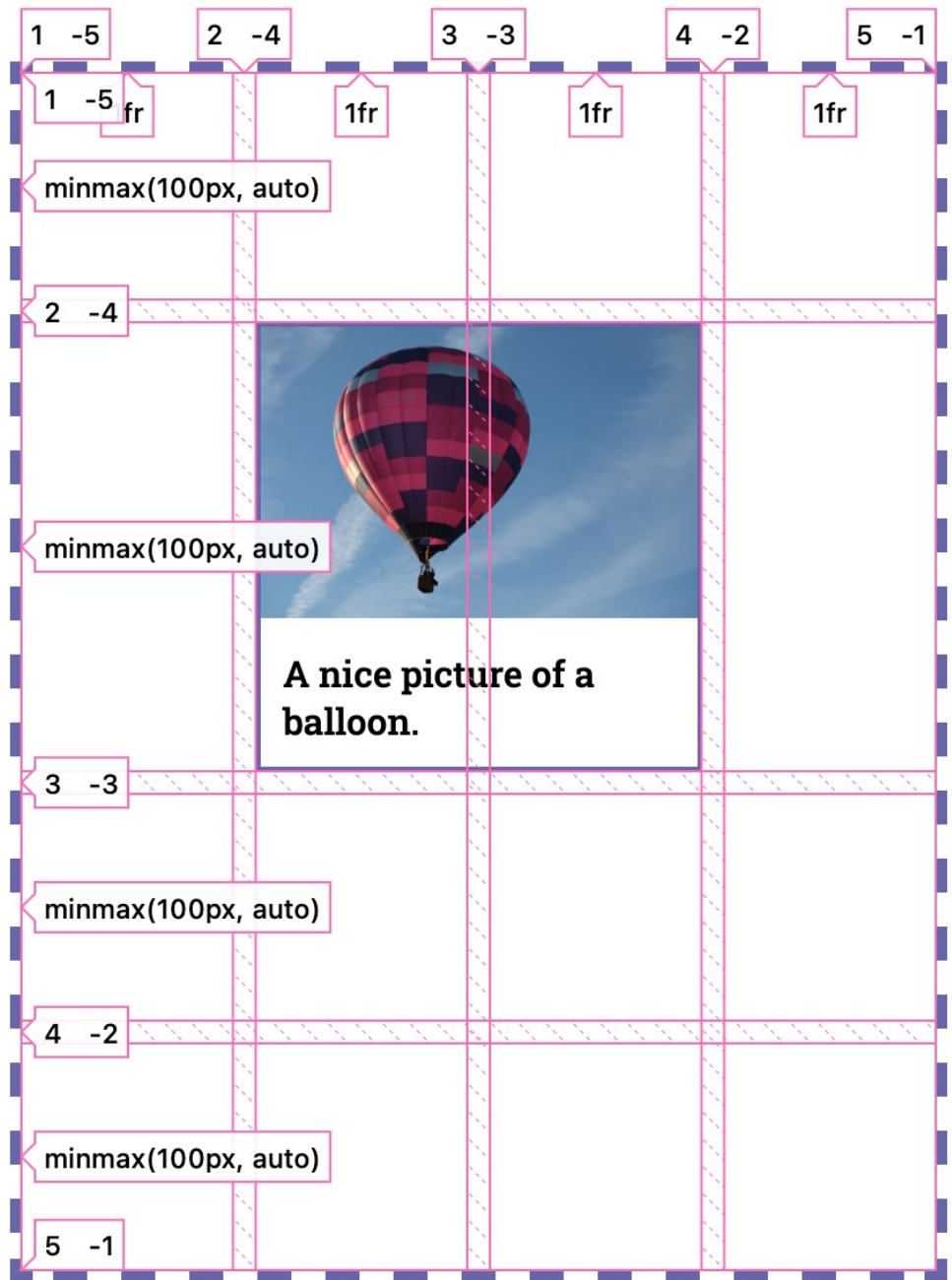


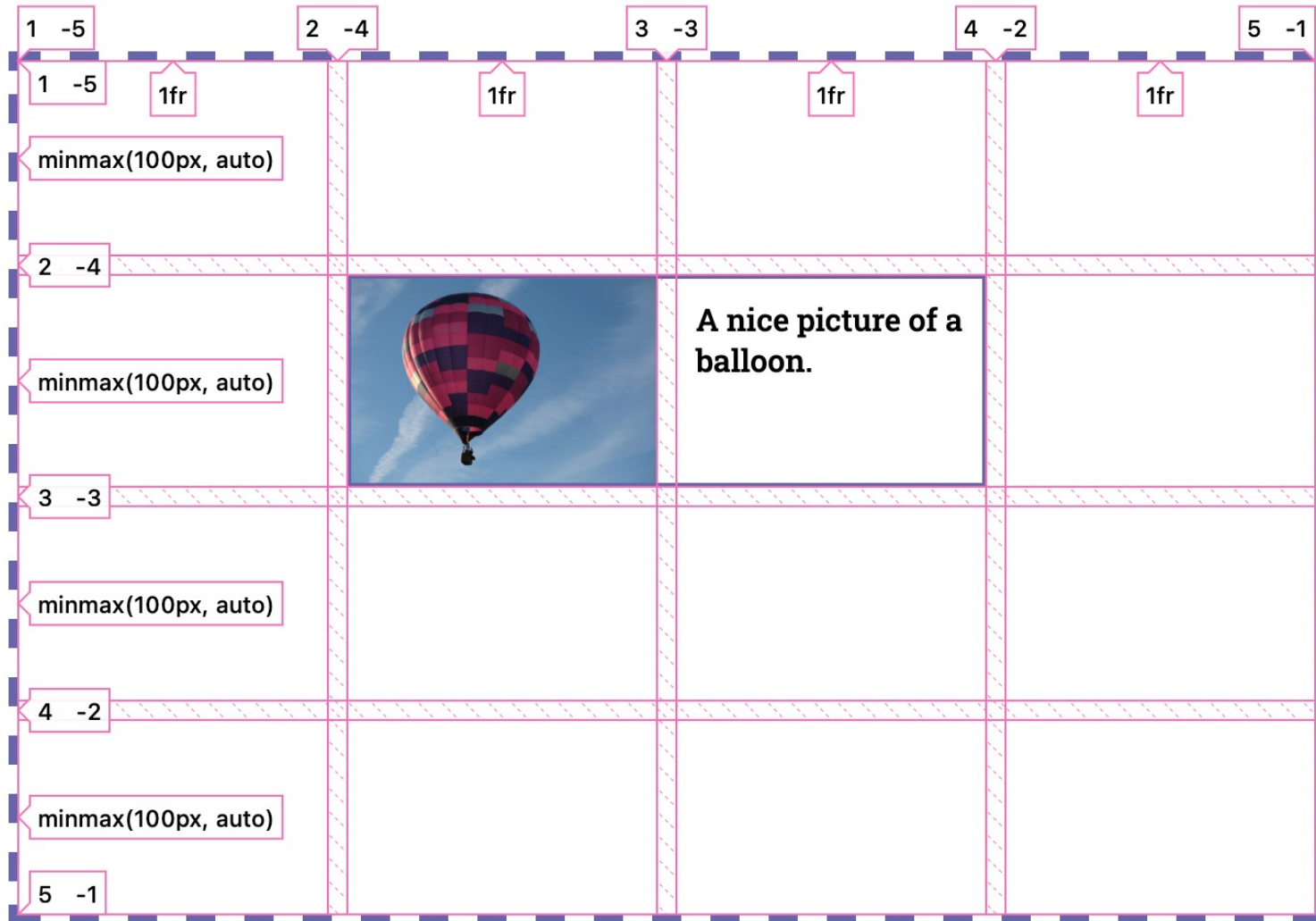
subgrid + container queries

Creating subgrids or using tracks based on available space.



```
@container my-grid (min-width: 600px) {  
  .item {  
    display: grid;  
    grid-template-columns: subgrid;  
  }  
}
```





So many possibilities

Complexity

Each new feature is relatively simple to use, however each brings new interactions, and the potential of unexpected behavior.



```
.container {  
  container-name: inline-size;  
}
```



**Bite sized maple orange tempeh leek mint
coconut creamy cauliflower.**

**I come after the container. Thai super
chili summer fruit salad cayenne chili
pepper refreshing cucumber splash tahini drizzle
lemonade zest with morning smoothie bowl entree black
beans.**



**Bite sized maple orange tempeh leek mint
coconut creamy cauliflower.**

**I come after the container. Thai super chili summer
fruit salad cayenne chili pepper refreshing cucumber
splash tahini drizzle lemonade zest with morning
smoothie bowl entree black beans.**

It's a good time to *really* learn CSS

It will help you to take advantage of all of this new greatness.

The problem with new layout

The ability to disconnect the source and focus order from the display.

```
ul {
  display: grid;
  gap: 20px;
  grid-template-columns: repeat(6, minmax(0, 1fr));
  grid-template-areas:
    "a a b b b b"
    "c c d d e e"
    "f f g g h h";
}
.card1 { grid-area: c; }
.card2 { grid-area: f; }
.card3 { grid-area: d; }
.card4 { grid-area: h; }
.card5 { grid-area: e; }
.card6 { grid-area: a; }
.card7 { grid-area: g; }
.card8 { grid-area: b; }
```

Card title 6

Nori grape silver beet broccoli kombu
beet greens fava bean potato quandong
celery.

[More about card 6](#)

Card title 8

Gumbo kakadu plum komatsuna black-eyed pea.

[More about card 8](#)

Card title 1

Veggies es bonus vobis, proinde vos
postulo essum magis.

[More about card 1](#)

Card title 3

Turnip greens yarrow ricebean rutabaga
endive cauliflower.

[More about card 3](#)

Card title 5

Brussels sprout coriander water chestnut
gourd swiss chard.

[More about card 5](#)

Card title 2

Gumbo beet greens corn soko endive
gumbo gourd

[More about card 2](#)

Card title 7

Celery quandong swiss chard.

[More about card 7](#)

Card title 4

Lotus root water spinach fennel kombu
maize bamboo shoot.

[More about card 4](#)

Here's a really cool thing!

Please don't use it.

How do we ensure a reusable component retains a reasonable focus order?

Wherever it is in the layout, no matter how it reflows at different breakpoints.

Browser tabs: [css-flexbox][css-grid] Providi... x +

Address bar: github.com/w3c/csswg-drafts/issues/7387

Navigation: Product Solutions Open Source Pricing Search Sign in Sign up

Repository: w3c/csswg-drafts Public Fork 565 Star 3.7k

Issue: [css-flexbox][css-grid] Providing authors with a method of opting into following the visual order, rather than logical order #7387 New issue

Status: Open rachelandrew opened this issue on Jun 19 · 16 comments

Comment by rachelandrew (Contributor) on Jun 19:

A proposal to provide a method in CSS for authors to opt in to following the visual order when in a flex or grid context.

The [grid](#) and [flexbox](#) specifications detail that reordering caused by these methods is only visual, and note that use of order or grid placement to do logical ordering is non-conforming.

With care from the author in creating a good document, this in general works well for screen reader users. Authors sometimes also use this to improve the experience for screenreader users. For example, by being able to show something visually near the top of the page but, because it would be repetitive for a screen reader user, physically locate it in elsewhere in the document.

Problems

Visual/tab order disconnect for keyboard users

In most cases, the document order is the correct logical order for screen reader users who are not able to see the visual display at all. However, if the author has reordered content using grid or flexbox, then the tab order of the document for keyboard users who are referencing the visual display can become disconnected, as it follows the document order.

Assignees: No one assigned

Labels: Closed Accepted by CSSWG Resolution, css-display-4, Needs Edits

Projects: None yet

Milestone: No milestone

Development: No branches or pull requests

A new reading-order-items property

Proposal: <https://developer.chrome.com/blog/reading-order/>



```
display: grid;
grid-template-columns: repeat(6, minmax(0, 1fr));
grid-template-areas:
  "a a b b b b"
  "c c d d e e"
  "f f g g h h";
reading-order-items: grid rows;
}
```

Being able to reorder content from CSS using things we can identify using CSS is useful.

But we need to be able to do so in a way that works for all our users.

The next big CSS thing?

Is more likely to be a lot of small things.

Thank you!

@rachelandrew