

Your CSS Layout Toolkit for 2019

@rachelandrew

A CSS Layout System

Real layout for the first time!

Components of CSS Layout

Flow Layout, Grid, Flexbox, Multiple-column Layout

Components of CSS Layout

Writing Modes, Logical Properties & Values, Alignment, Sizing

Components of CSS Layout

Media Queries, Feature Queries

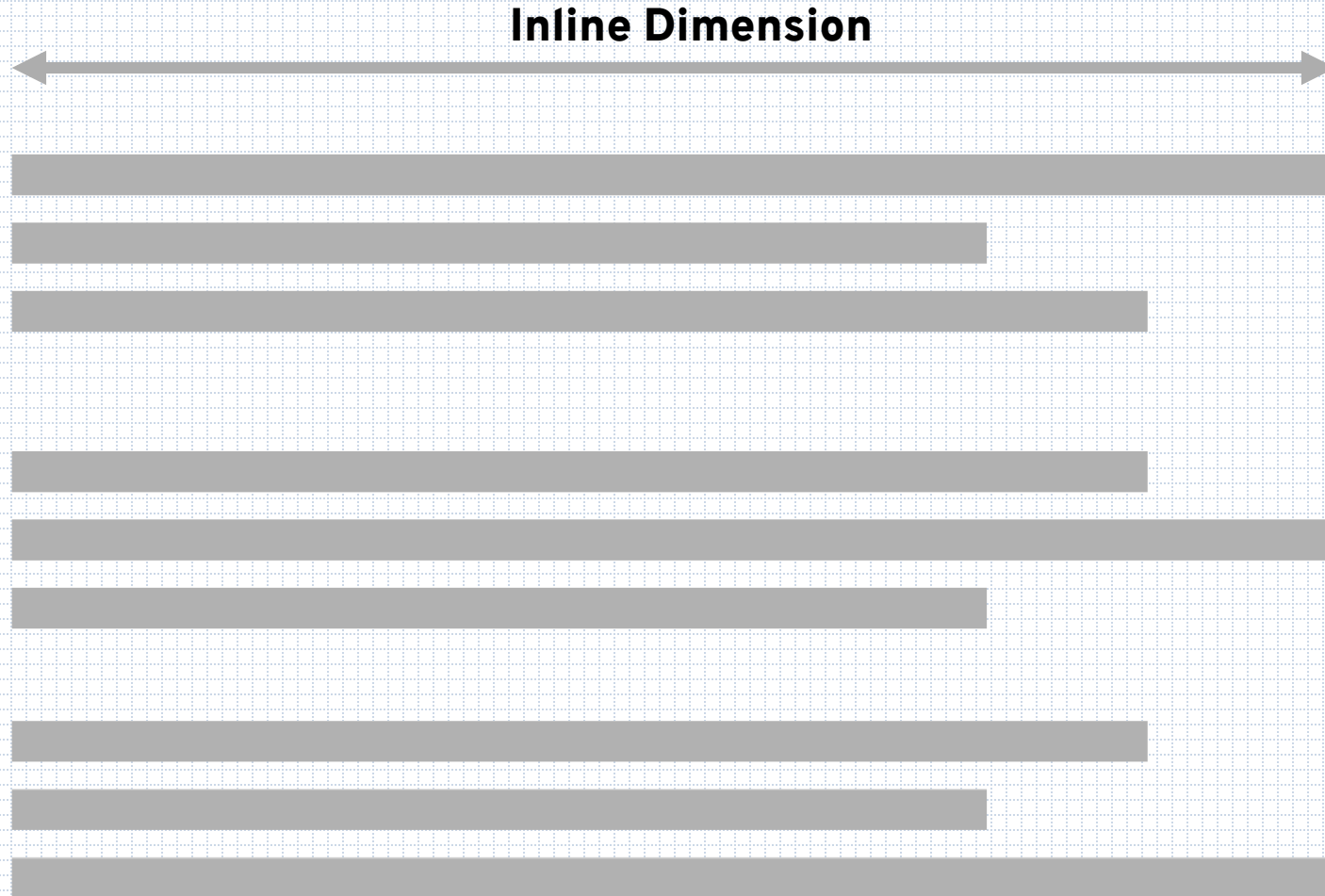
Components of CSS Layout

CSS Shapes, Transforms, Scroll Snapping, Variable Fonts

Writing Modes

Horizontal or Vertical

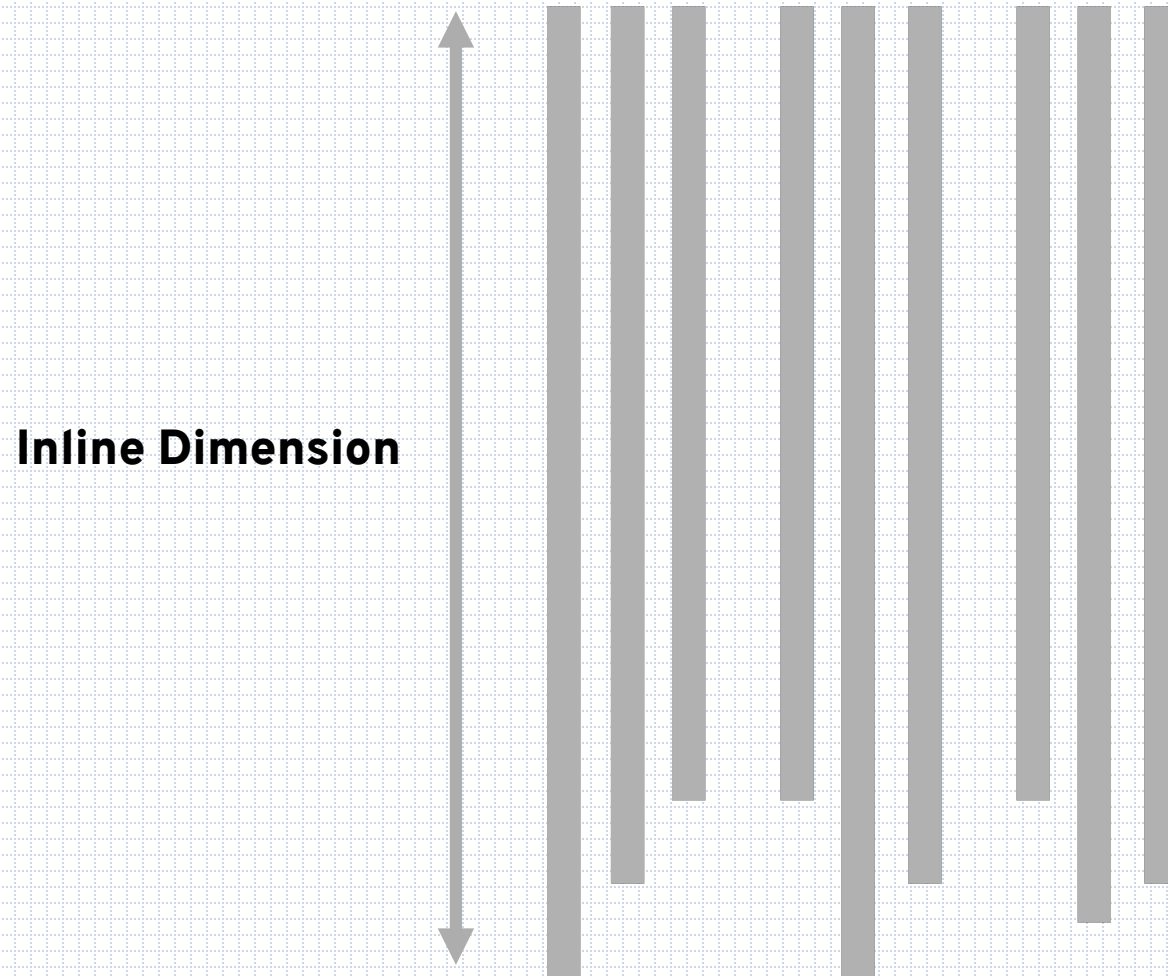
Horizontal Writing Mode



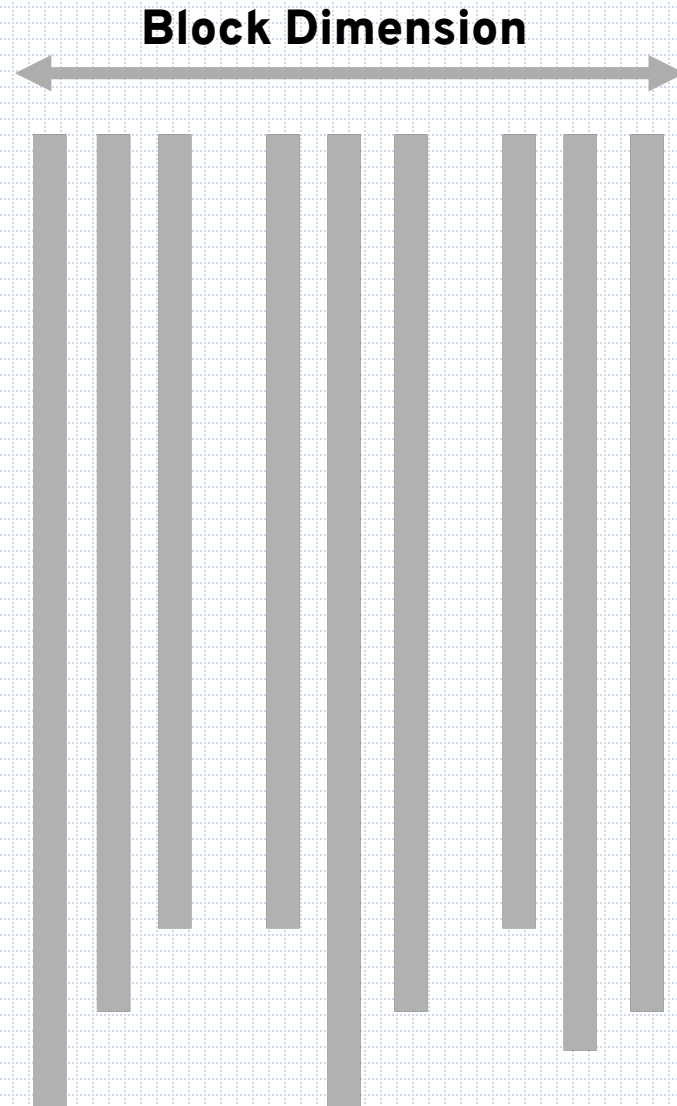
Horizontal Writing Mode



Vertical Writing Mode



Vertical Writing Mode

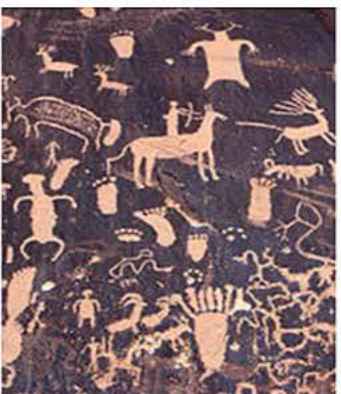


文字的故事

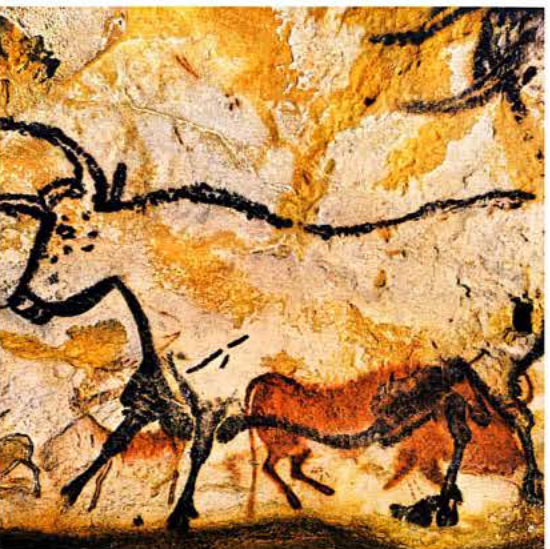
記錄：演變 05/20/2014

人類和其他生物物種總是在以自己獨特的方式和語言給大自然和自己的生命歷程留下一些印跡。當遠古時期的獵人根據熊掌印開始追蹤的時候，那便是最早的「視覺交流與傳播」(Visual Communication) 的開始。

最早的視覺傳達方式基本都是利用圖形進行的。這是北美印地安在史前的岩洞壁畫



法國發現的拉斯考克岩洞中，古代人類的原始繪畫，大約西元前一萬——萬五千年



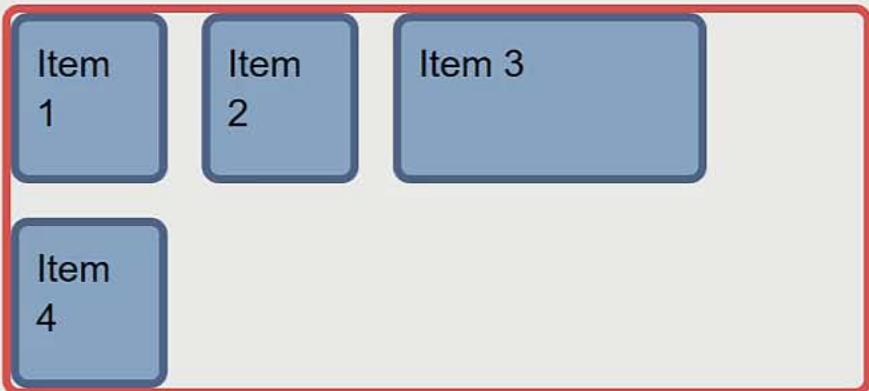
古代中亞文明的黑石鴨形，是由烏爾王奉獻給南那神的供品，公元前3000年左右



一本完整的平面設計史，是應該從人類開始記錄或傳播自己的思想開始的。人類為了記錄自己的思想、活動、成就，開始是利用圖畫作手段，但是圖畫對於思想的表達能力非常有限，特別是對於比較抽象的思想的記錄，幾乎無能為力。後

Logical Properties & Values

Writing-mode relative equivalents of physical properties.



HTML

CSS

```
6
7 ▼.grid { I
8   width: 500px;
9   display: grid;
10  border: 5px solid #CE454F;
11  border-radius: 10px;
12  grid-template-columns: .2fr .2fr .4fr;
13  grid-template-rows: 100px 100px;
14  grid-gap: 20px;
15 }
16
17 ▼.grid > * {
18   border: 5px solid #526683;
19   border-radius: 10px;
20   background-color: #89A4BE;
21   padding: 10px;
22
23 }
```

JS

TABLE OF CONTENTS

1	Introduction
2	Flow-Relative Values: ‘block-start’, ‘block-end’, ‘inline-start’, ‘inline-end’
2.1	Logical Values for the ‘caption-side’ Property
2.2	Flow-Relative Values for the ‘float’ and ‘clear’ Properties
2.3	Flow-Relative Values for the ‘text-align’ Property
2.4	Flow-Relative Values for the ‘resize’ Property
3	Flow-Relative Page Classifications
4	Flow-Relative Box Model Properties
4.1	Logical Height and Logical Width: the ‘block-size’ and ‘inline-size’ properties
4.2	Flow-relative Margins: the ‘margin-block-start’, ‘margin-block-end’, ‘margin-inline-start’, ‘margin-inline-end’ properties and ‘margin-block’ and ‘margin-inline’ shorthands
4.3	Flow-relative Offsets: the ‘inset-block-start’, ‘inset-block-end’, ‘inset-inline-start’, ‘inset-inline-end’ properties and ‘inset-block’, ‘inset-inline’, and ‘inset’ shorthands
4.4	Flow-relative Padding: the ‘padding-block-start’, ‘padding-block-end’,

CSS Logical Properties and Values Level 1



W3C Working Draft, 27 August 2018

This version:

<https://www.w3.org/TR/2018/WD-css-logical-1-20180827/>

Latest published version:

<https://www.w3.org/TR/css-logical-1/>

Editor's Draft:

<https://drafts.csswg.org/css-logical-1/>

Previous Versions:

<https://www.w3.org/TR/2017/WD-css-logical-1-20170518/>

Issue Tracking:

[Inline In Spec](#)

[GitHub Issues](#)

Editors:

[Rossen Atanassov](#) (Microsoft)

[Elika J. Etemad](#) / [fantasai](#) (Invited Expert)

Suggest an Edit for this Spec:

[GitHub Editor](#)

Copyright © 2018 W3C® ([MIT](#), [ERCIM](#), [Keio](#), [Beihang](#)). W3C [liability](#), [trademark](#) and [permissive document license](#) rules apply.

Abstract

<https://www.w3.org/TR/css-logical-1/>



ABOUT THE AUTHOR


Rachel Andrew is not only editor-in-chief of Smashing Magazine, but also a web developer, writer and speaker.

She is the author of a number of

MARCH 29, 2018 • [8 comments](#)

Understanding Logical Properties And Values

QUICK SUMMARY ↔ *CSS Logical Properties and Values aren't quite ready to be used yet, however learning about them can help you to understand CSS Layout, and the interaction with Writing Modes.*

 9 min read

 [CSS](#), [Layouts](#), [Browsers](#)

 Share on [Twitter](#) or [LinkedIn](#)

<https://www.smashingmagazine.com/2018/03/understanding-logical-properties-values/>

left or right, we use the positioning onset properties `top`, `left`, `bottom` and `right`. We set margins, padding, and borders as

Box Alignment

Consistent alignment across layout methods.

Align or Justify?

align-content

align-self

align-items

justify-content

justify-self

justify-items

In Grid Layout

align-content

align-self

align-items



Block Axis

justify-content

justify-self

justify-items



Inline Axis



HTML

CSS

```
27 ▼ .three {  
28     grid-column: 2;  
29     grid-row: 2;  
30 }  
31  
32 ▼ .four {  
33     grid-column: 3;  
34     grid-row: 1 / 3;    I  
35  
36 }  
37  
38 ▼ .grid > * {  
39     border: 5px solid #526683;  
40     border-radius: 10px;  
41     background-color: #89A4BE;  
42     padding: 10px;  
43  
44 }
```

JS



HTML

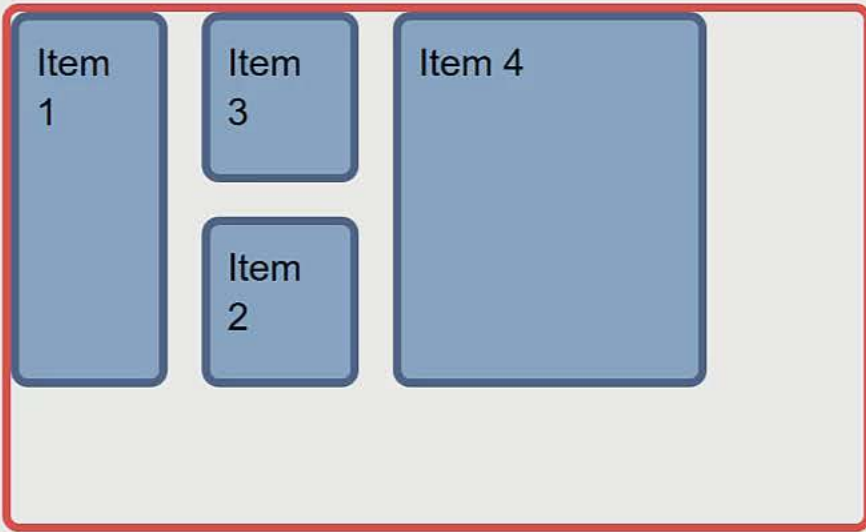
CSS

```
1 body {  
2   background-color: #E9E9E6;  
3   font: 1.4em/1.3 Helvetica, Arial, sans-  
   serif;  
4   padding: 30px;  
5 }  
6  
7 .grid {  
8   display: grid;  
9   border: 5px solid #CE454F;  
10  border-radius: 10px;  
11  inline-size: 500px;  
12  grid-template-columns: 1fr 1fr 2fr;  
13  grid-template-rows: 100px 100px;  
14  grid-gap: 20px;  
15  |  
16 }  
17  
18 one {
```

JS

Distributing space

align-content and justify-content



HTML

CSS

```
4     padding: 50px;
5   }
6
7   ▼ .grid {
8     display: grid;
9     border: 5px solid #CE454F;
10    border-radius: 10px;
11    inline-size: 500px;
12    block-size: 300px;
13    grid-template-columns: .2fr .2fr .4fr;
14    grid-template-rows: 100px 100px;
15    grid-gap: 20px;
16
17  }
18
19  ▼ .one {
20    grid-column: 1;
21    grid-row: 1 / 3;
22  }
```

JS

In Flex Layout

align-content

align-self

align-items



Cross Axis

justify-content

Main Axis

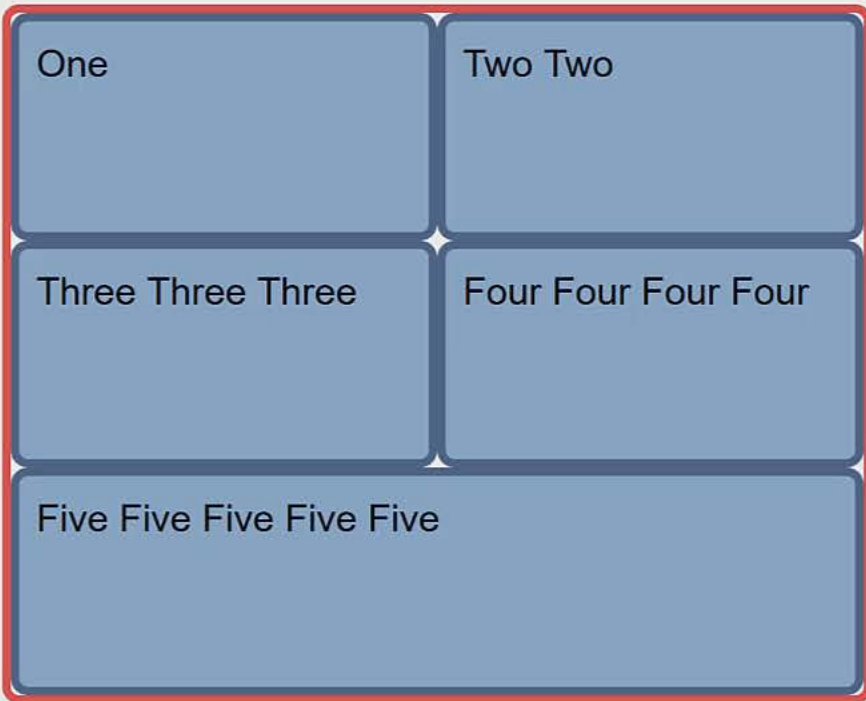


HTML

CSS

```
4     padding: 50px;
5   }
6
7   .flex {
8     display: flex;
9     border: 5px solid #CE454F;
10    border-radius: 10px;
11    inline-size: 500px;
12    block-size: 400px;
13
14  }
15
16  .flex > * {
17    border: 5px solid #526683;
18    border-radius: 10px;
19    background-color: #89A4BE;
20    padding: 10px;
21
22  }
```

JS

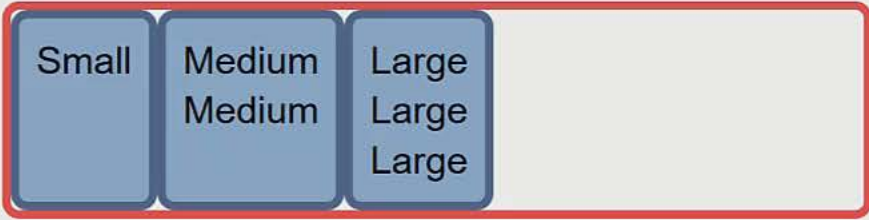


HTML

CSS

```
1 ▼body {  
2   background-color: #E9E9E6;  
3   font: 1.4em/1.3 Helvetica, Arial, sans-  
   serif;  
4   padding: 30px;  
5 }  
6  
7 ▼.flex {  
8   display: flex;  
9   border: 5px solid #CE454F;  
10  border-radius: 10px;  
11  inline-size: 500px;  
12  block-size: 400px;  
13  flex-wrap: wrap;      I  
14  |  
15 }  
16  
17 ▼.flex > * {  
18   border: 5px solid #526683;
```

JS



HTML

CSS

```
1 body {  
2   background-color: #E9E9E6;  
3   font: 1.4em/1.3 Helvetica, Arial, sans-  
   serif;  
4   padding: 30px;  
5 }  
6  
7 .flex {  
8   display: flex;  
9   border: 5px solid #CE454F;  
10  border-radius: 10px;  
11  inline-size: 500px;  
12  |  
13 }  
14  
15 .flex > * {  
16   border: 5px solid #526683;  
17   border-radius: 10px;  
18   background-color: #89A4BE;
```

JS

Alignment in Block Layout

No browser implementation as yet ...

Gaps

row-gap, column-gap, gap

TABLE OF CONTENTS

1	Introduction
1.1	Module interactions
1.2	Values
1.3	Partial Implementations
2	Overview of Alignment Properties
3	Alignment Terminology
4	Alignment Keywords
4.1	Positional Alignment: the 'center', 'start', 'end', 'self-start', 'self-end', 'flex-start', 'flex-end', 'left', and 'right' keywords
4.2	Baseline Alignment: the 'baseline' keyword and 'first'/'last' modifiers
4.3	Distributed Alignment: the 'stretch', 'space-between', 'space-around', and 'space-evenly' keywords
4.4	Overflow Alignment: the 'safe' and 'unsafe' keywords and scroll safety limits
5	Content Distribution: Aligning a Box's Contents Within Itself
5.1	The 'justify-content' and 'align-content' Properties
5.1.1	Block Containers (Including Table Cells)
5.1.2	Multicol Containers
5.1.3	Flex Containers
5.1.4	Grid Containers
5.2	
5.3	
5.4	
6	Self-Alignment: Aligning the Box Within Its Parent

§ 8. Gaps Between Boxes

While ['margin'](#) and ['padding'](#) can be used to specify visual spacing around individual boxes, it's sometimes more convenient to globally specify spacing between adjacent boxes within a given layout context, particularly when the spacing is different between boxes as opposed to between the first/last box and the container's edge.

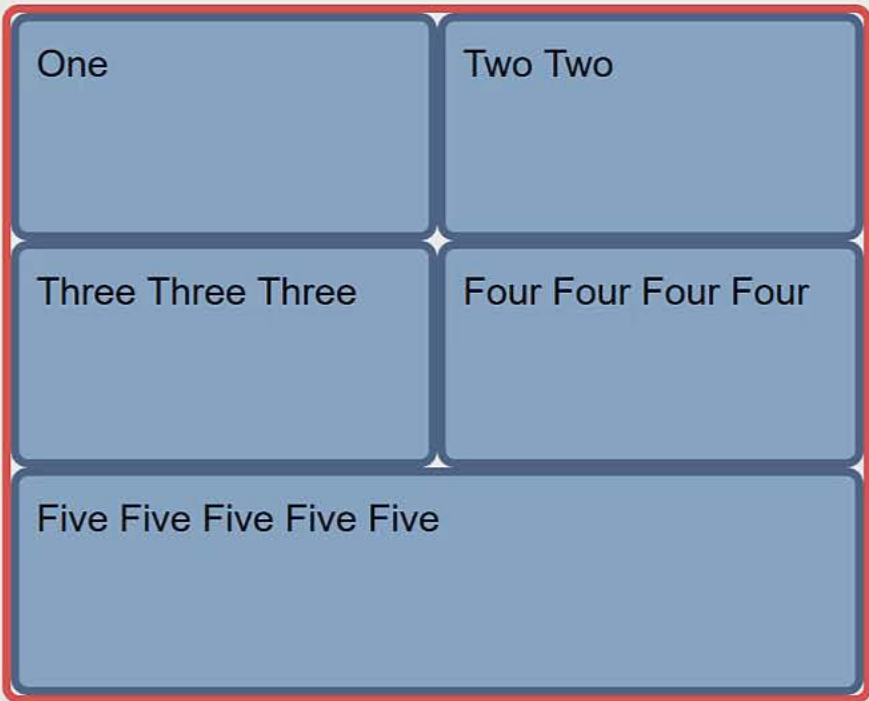
The ['gap'](#) property, and its ['row-gap'](#) and ['column-gap'](#) sub-properties, provide this functionality for [multi-column](#), [flex](#), and [grid layout](#).

§ 8.1. Row and Column Gutters: the ['row-gap'](#) and ['column-gap'](#) properties

<i>Name:</i>	'row-gap', 'column-gap'
<i>Value:</i>	normal <length-percentage>
<i>Initial:</i>	normal
<i>Applies to:</i>	multi-column containers , flex containers , grid containers
<i>Inherited:</i>	no
<i>Percentages:</i>	refer to corresponding dimension of the content area
<i>Computed value:</i>	as specified, with <length> s made absolute
<i>Canonical order:</i>	per grammar
<i>Animatable:</i>	as length, percentage, or calc

<https://www.w3.org/TR/css-align-3/#gaps>

separating boxes in the container's [block axis](#).



HTML

CSS

```
1 body {  
2   background-color: #E9E9E6;  
3   font: 1.4em/1.3 Helvetica, Arial, sans-  
   serif;  
4   padding: 30px;  
5 }  
6  
7 .flex {  
8   display: flex;  
9   border: 5px solid #CE454F;  
10  border-radius: 10px;  
11  inline-size: 500px;  
12  block-size: 400px;  
13  flex-wrap: wrap;  
14  |  
15 }  
16  
17 .flex > * {  
18   border: 5px solid #526683;
```

JS



Rachel Andrew

@rachelandrew

Follow



Should I use Grid or Flexbox? Build your component with both, or a mixture, and see which works best. Do that a few times and the decisions will become more obvious to you on looking at a component. #css

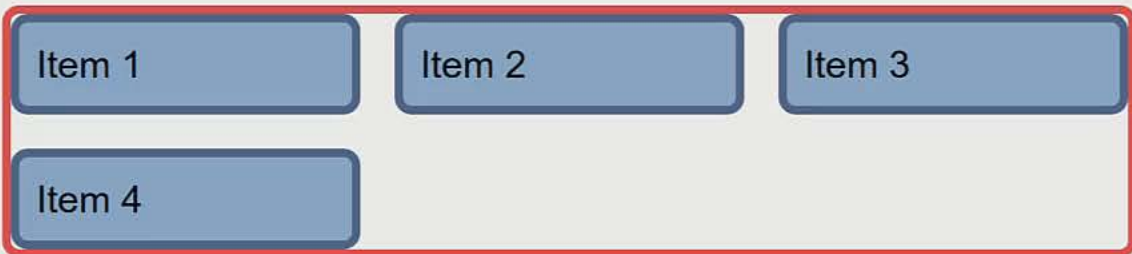
3:16 AM - 22 Sep 2018

98 Retweets 495 Likes



Sizing

How big should this item be?



HTML

```
1 <div class="grid">
2   <div class="one">Item 1</div>
3   <div class="two">Item 2</div>
4   <div class="three">Item 3</div>
5   <div class="four">Item 4</div>
6 </div>
```

CSS

```
4   padding: 50px;
5   }
6
7 .grid {
8   display: grid;
9   border: 5px solid #CE454F;
10  border-radius: 10px;
11  grid-template-columns: 1fr 1fr 1fr;
12  grid-gap: 20px;
13 }
```

JS

Item 1 is longer than the others

Item 2

Item 3

HTML

CSS

```
7 ▼ .flex {  
8     display: flex;  
9     border: 5px solid #CE454F;  
10    border-radius: 10px;  
11 }  
12  
13 ▼ .flex > * {  
14     border: 5px solid #526683;  
15     border-radius: 10px;  
16     background-color: #89A4BE;  
17     padding: 10px;  
18     flex: 1 1 auto;  
19 }  
20
```

JS




ABOUT THE AUTHOR

Rachel Andrew is not only editor-in-chief of Smashing Magazine, but also a web developer, writer and speaker. She is the author of a number of books, including ... [More about](#)

JANUARY 16, 2018 • [9 comments](#)

How Big Is That Box? Understanding Sizing In CSS Layout

QUICK SUMMARY ↗ When starting to use Flexbox and Grid, it can be frustrating to find that we sometimes don't get the layout we expect. Often this is due to the way sizing is calculated in these new layout methods. In this article, I try to explain exactly how big that box is, and how it got to be that size!

 22 min read [CSS](#), [Layouts](#), [Browsers](#) Share on [Twitter](#) or [LinkedIn](#)

<https://www.smashingmagazine.com/2018/01/understanding-sizing-css-layout/>

inside grid and flex items. Quite often this *just works*, and we get the



Media Queries

Level 4 Media Queries

Media Features

What features does this environment have?

Testing pointer types

You have a fine pointer, are you using a mouse or trackpad?

Testing hover abilities

You look like you can hover.

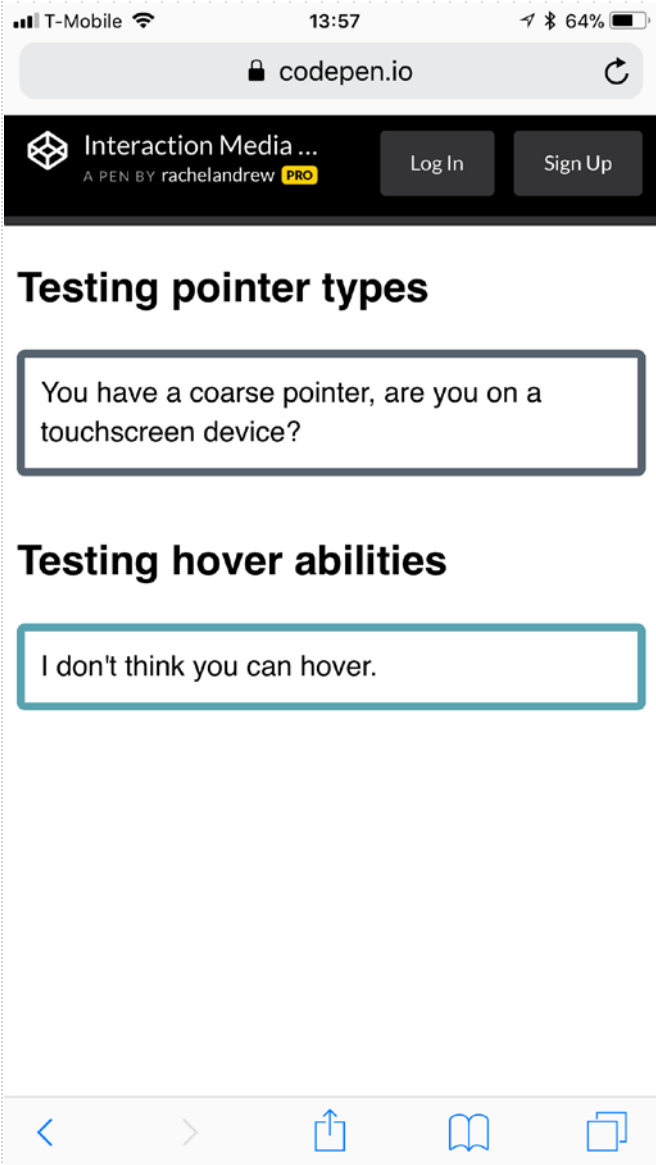
HTML

CSS

```
28
29 ▼ @media (pointer:coarse) {
30 ▼   .which-pointer::after {
31     content: "You have a coarse pointer, are
      you on a touchscreen device?";
32
33   }
34 }
35
36 ▼ @media (pointer:fine) {
37 ▼   .which-pointer::after {
38     content: "You have a fine pointer, are
      you using a mouse or trackpad?";
39
40   }
41 }
```

<https://codepen.io/rachelandrew/pen/wYaLbR>

JS



Testing pointer types

You have a fine pointer, are you using a mouse or trackpad?

Testing hover abilities

You look like you can hover.

Feature Queries

Hey browser! Do you speak grid?

@supports

If you see this message, your browser supports the grid value for the display property.

HTML

CSS



```
1
2
3 ▼ .has-grid {
4     display: none;
5 }
6
7 ▼ @supports (display: grid) {
8     ▼ .has-grid {
9         display: block;
10    }
11 }
12
13
14
15
```

<https://codepen.io/rachelandrew/pen/zmGVgK>

Using Feature Queries in CSS



By [Jen Simmons](#)

Posted on August 17, 2016 in [CSS](#) and [Featured Article](#)  [Share This](#) 

There's a tool in CSS that you might not have heard of yet. It's powerful. It's been there for a while. And it'll likely become one of your favorite new things about CSS.

Behold, the `@supports` rule. Also known as [Feature Queries](#).

With `@supports`, you can write a small test in your CSS to see whether or not a particular "feature" (CSS property or value) is supported, and apply a block of code (or not) based on the answer. Like this:

```
@supports (display: grid) {  
  // code that will only run if CSS Grid is supported by the browser  
}
```

<https://hacks.mozilla.org/2016/08/using-feature-queries-in-css/>

Now, there seems to be a bit of confusion about what Feature Queries are for.

Subgrid

What's coming in CSS Grid Level 2?

My Header

Turnip greens yarrow
ricebean rutabaga endive
cauliflower sea lettuce
kohlrabi amaranth water
spinach avocado daikon
napa cabbage asparagus
winter purslane kale.

My footer

My Header

Turnip greens yarrow
ricebean rutabaga endive
cauliflower sea lettuce
kohlrabi amaranth water
spinach avocado daikon
napa cabbage asparagus
winter purslane kale.

My footer

My Header

Turnip greens yarrow
ricebean rutabaga endive
cauliflower sea lettuce
kohlrabi amaranth water
spinach avocado daikon
napa cabbage asparagus
winter purslane kale.

My footer

My Header
is taller

Turnip greens yarrow
ricebean rutabaga endive
cauliflower sea lettuce
kohlrabi amaranth water
spinach avocado daikon
napa cabbage asparagus
winter purslane kale.

My footer

My Header

Turnip greens yarrow
ricebean rutabaga endive
cauliflower sea lettuce
kohlrabi amaranth water
spinach avocado daikon
napa cabbage asparagus
winter purslane kale.

My footer

My Header

Turnip greens yarrow
ricebean rutabaga endive
cauliflower sea lettuce
kohlrabi amaranth water
spinach avocado daikon
napa cabbage asparagus
winter purslane kale.

My footer

New Challenges

The potential for new accessibility issues.



navigation

? DECKS & VIDEOS

? ABOUT LÉONIE

+ CATEGORIES

+ TAGS

+ ON OTHER BLOGS

+ CONFERENCES

site theme

LIGHT

DARK

Flexbox & the keyboard navigation disconnect

CODE THINGS  FEBRUARY 4TH 2016

CSS Flexbox can create a disconnect between the DOM order and visual presentation of content, causing keyboard navigation to break. For this reason, the **CSS Flexible Box Layout module** warns against resequencing content logic, but asking authors not to use flexbox in this way seems illogical in itself.

TLDR: The only viable way (in my opinion) for the flexbox disconnect to be resolved, is in the browser (as with the Firefox “bug”) and the accessibility tree.

The problem

When content is presented in sequence, we expect to read it and navigate through it in a logical order. In the West this is typically top to bottom/left to right. In the following example, the expected order is “1”, “2”, “3”.

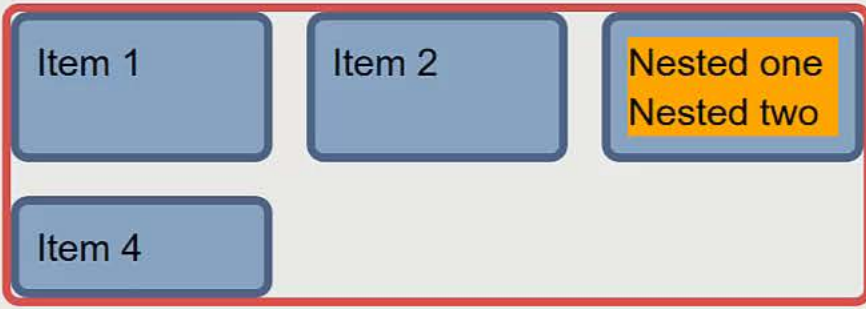
```
<div>
  <a href="/">One</a>
  <br>
  <a href="/">Two</a>
  <br>
  <a href="/">Three</a>
</div>
```

<https://tink.uk/flexbox-the-keyboard-navigation-disconnect/>

```
<div style="display: flex;">
  <a href="/" style="border: 2px solid black;">One</a>
```


display: contents

A cautionary tale.



HTML

CSS

```
13 grid-gap: 20px;
14 }
15
16 .three {
17     I
18 }
19
20 .grid > * {
21     border: 5px solid #526683;
22     border-radius: 10px;
23     background-color: #89A4BE;
24     padding: 10px;
25 }
26
27 .three > div {
28     background-color: orange;
29 }
30
```

JS

More accessible markup with display: contents

*Published on 21 April
2018 by Hidde de Vries
in [code](#).*

CSS Grid Layout lets you turn an element into a grid, and place the element's *direct children* onto it. Given that, it might be tempting to use flatter markup, but less meaning is usually less accessibility. With `display: contents`, we can place *grand children* on a grid, which lets us have accessible markup *and* beautiful layout. Let's dive into the details!

Below, I will explain in more detail what I mean by children and grand children, and then show how we can use `display: contents` to improve this. *Note: this is currently broken in supporting browsers, more details below*

<https://hidddevries.nl/en/blog/2018-04-21-more-accessible-markup-with-display-contents>

of that element become grid items and are layed out on it. To refresh for those

What's next?

What do you need?

[Features](#)[Business](#)[Explore](#)[Marketplace](#)[Pricing](#)[Sign in](#) or [Sign up](#)[w3c](#) / [csswg-drafts](#)[Watch](#)

268

[★ Star](#)

1,143

[Fork](#)

202

[Code](#)[Issues](#) 1,152[Pull requests](#) 22[Projects](#) 3[Insights](#)[Labels](#)[Milestones](#)[New issue](#)

1,152 Open ✓ 1,527 Closed

Author ▾

Labels ▾

Projects ▾

Milestones ▾

Assignee ▾

Sort ▾

[\[css-position\]](#) Sticky positioning with transform between it and reference box [css-position-3](#)

#3186 opened 23 minutes ago by Loirooriorl

[\[css-position\]](#) Sticky positioning when constraining flow box is not in containing block chain[css-position-3](#)

#3185 opened an hour ago by Loirooriorl

[\[css-ui-4\]\[css-tables-3\]](#) should table outlines surround captions too? [css-tables-3](#) [css-ui-4](#)

#3184 opened 11 hours ago by heycam

2

[\[css-syntax\]](#) Add background-rotate property that we just have to rotate the background without affecting the text and only one layer

#3183 opened a day ago by piaoxuelia

4

[\[css-syntax\]](#) Correctly handle "escaped EOF"

#3182 opened 3 days ago by tabatkins

[\[css-inline-3\]](#) Clarify alignment values [css-inline-3](#)

#3181 opened 3 days ago by fantasai

[css-writing-modes-4](#)

#3175 opened 4 days ago by upsuper

<https://github.com/w3c/csswg-drafts/issues>

Firefox 62

CSS Shapes, Shape Path Editor, Variable Fonts

<https://developer.mozilla.org/en-US/docs/Mozilla/Firefox/Releases/62>

Chrome 69

Scroll Snap, display cutouts, conic gradients

<https://developers.google.com/web/updates/2018/09/nic69>

Edge HTML 17

Variable fonts

<https://blogs.windows.com/msedgedev/2018/04/30/edgehtml-17-april-2018-update/>

Thank you

<https://noti.st/rachelandrew/QEhSSc>