

adding explicit aspect ratios to CSS

Jen Simmons * CSS Working Group Meeting * October 2018

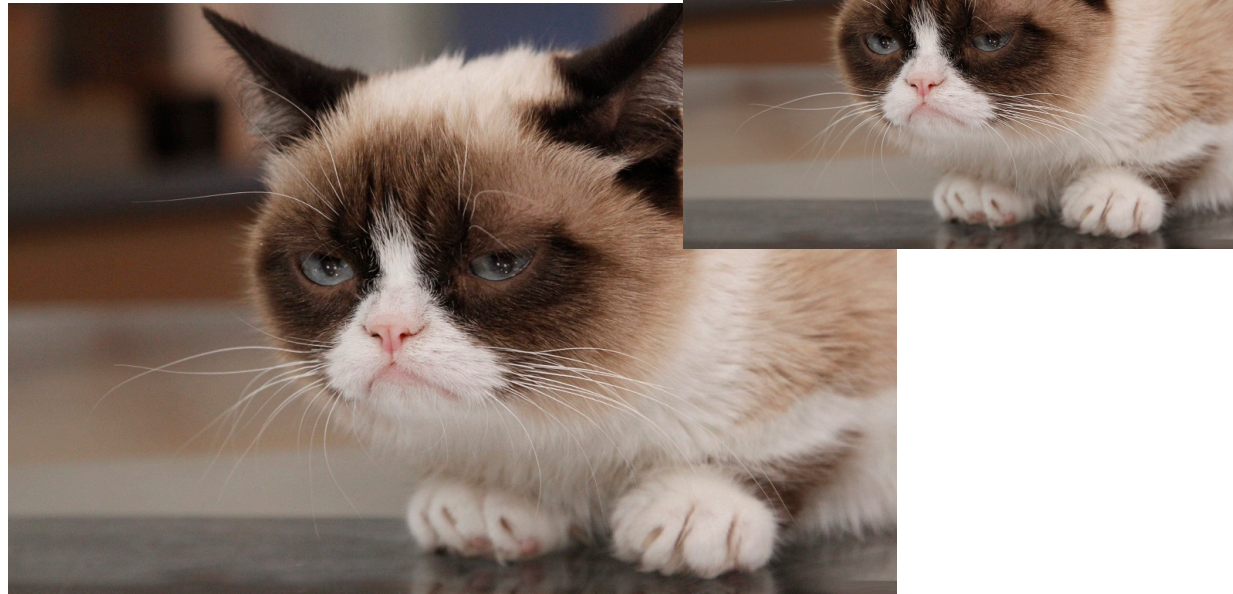
<https://drafts.csswg.org/css-sizing-4/#ratios>

reality

Some elements have an intrinsic height, width, and aspect ratio.

``

`<video>`



`width: 100%;`

`height: auto;`



``

problem

Some elements do not have
an intrinsic aspect ratio.

`<iframe>`

`<article>`

`<div>`



usecase 1: video



```
<iframe width="560" height="315"
src="https://www.youtube.com/
embed/0Gr1XSyxZy0"
frameborder="0" allow="autoplay;
encrypted-media"
allowfullscreen></iframe>
```

Why?

Adaptive Bitrate Streaming

9 Biggest Mistakes with CSS Grid



By [Jen Simmons](#)

Posted on July 18, 2018 in [CSS](#) and [Featured Article](#)

♥ Share This

It's easy to make lots of mistakes with a new technology, especially something that's as big of a change from the past as CSS Grid. In this video, I explain the 9 Biggest Mistakes people are making, with advice and tips for avoiding these pitfalls and breaking old habits.



For more information:

9 Biggest Mistakes with CSS Grid



By [Jen Simmons](#)

Posted on July 18, 2018 in [CSS](#) and [Featured Article](#) ♥ Share This

It's easy to make lots of mistakes with a new technology, especially something that's as big of a change from the past as CSS Grid. In this video, I explain the 9 Biggest Mistakes people are making, with advice and tips for avoiding these pitfalls and breaking old habits.



Aspect ratio is not maintained in a flexible context.

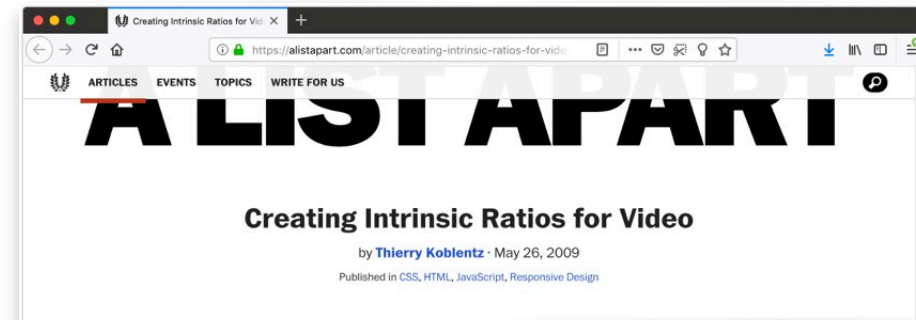
FITVIDS.JS



A lightweight, easy-to-use jQuery plugin
for fluid width video embeds.



FitVids.js was built by **Chris Coyier** and **Paravel**
See also: **Lettering.js** and **FitText**



Did you ever want to resize a video *on the fly*, scaling
intrinsic ratios for video, you can. This technique a
dimensions based on the width of their containing

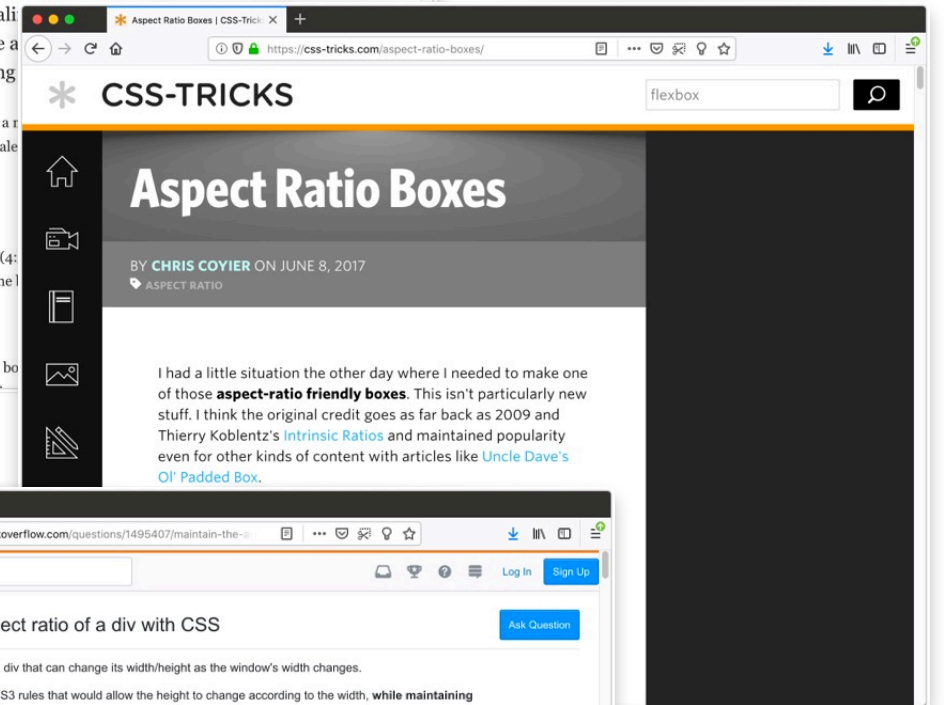
With intrinsic dimensions, a new width triggers a r
videos to resize and giving them the ability to scale
example one.

The concept

The idea is to create a box with the proper ratio (4:
inside that box stretch to fit the dimensions of the l

The trick

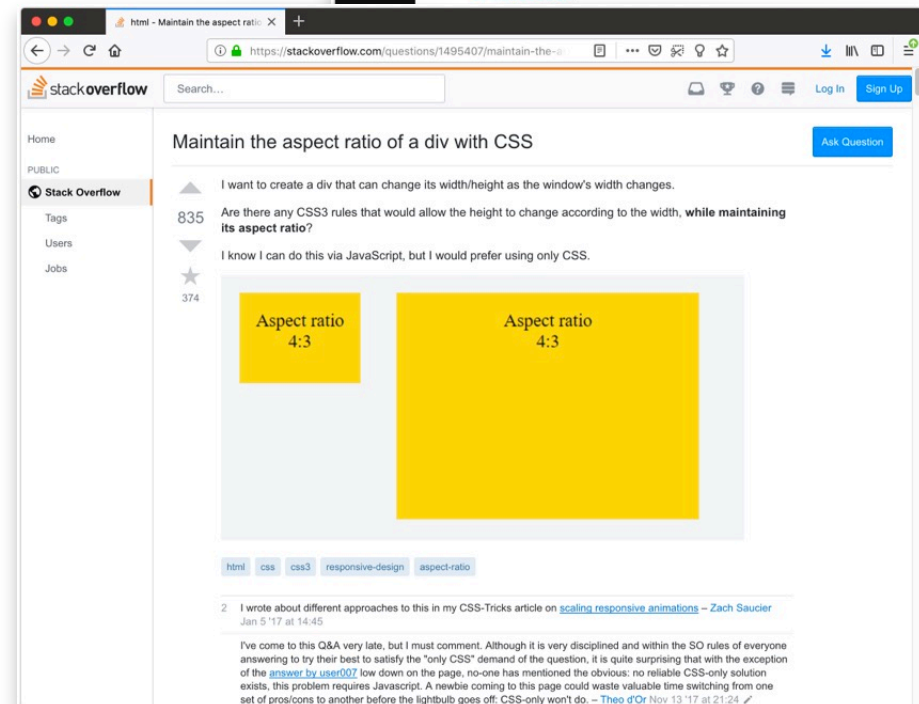
The `padding` property is the magic that styles a bo



Aspect Ratio Boxes

BY **CHRIS COYIER** ON JUNE 8, 2017
ASPECT RATIO

I had a little situation the other day where I needed to make one
of those **aspect-ratio friendly boxes**. This isn't particularly new
stuff. I think the original credit goes as far back as 2009 and
Thierry Koblenz's [Intrinsic Ratios](#) and maintained popularity
even for other kinds of content with articles like [Uncle Dave's
Ol' Padded Box](#).



solution

Some way to explicitly define an aspect ratio for replaced elements that don't have an intrinsic aspect ratio – so that they behave like replaced elements that do.

`aspect-ratio: 16 / 9;`



CSS Sizing Level 4 Draft

W3C Editor's Draft

TABLE OF CONTENTS

1

Introduction

1.1

Module interactions

1.2

Values

2

Aspect Ratios

2.1

Intrinsic Aspect Ratios: the 'aspect-ratio' property

2.2

Aspect Ratio Limits Option A: the 'from-ratio'

2.3

Aspect Ratio Limits Option B: the 'ar' unit

3

Intrinsic Size Determination

3.1

Intrinsic Sizes of Replaced Elements

3.2

Intrinsic Sizes of Non-Replaced Inlines

3.3

Intrinsic Sizes of Non-Replaced Blocks

3.4

Intrinsic Sizes in Table Layout

3.5

Intrinsic Sizes in Multi-column Layout

3.5.1

Min-content Sizes in Multi-column Layout

3.5.2

Max-content Sizes in Unconstrained-height Multi-column Layout

3.5.3

Max-content Sizes in Constrained-height Multi-column Layout

4

Extrinsic Size Determination

4.1

Stretch-fit Sizing

4.2

Contain-fit Sizing: stretching while maintaining an aspect ratio

4.3

Percentage Sizing

Changes

Acknowledgments

§ 2. Aspect Ratios

ISSUE 2 This section is a rough draft proposal. See discussion in [Issue 333](#) and [Issue 1173](#).

§ 2.1. Intrinsic Aspect Ratios: the 'aspect-ratio' property

<i>Name:</i>	'aspect-ratio'
<i>Value:</i>	auto <ratio>
<i>Initial:</i>	auto
<i>Applies to:</i>	all elements except inline boxes and internal ruby or table boxes
<i>Inherited:</i>	no
<i>Percentages:</i>	n/a
<i>Computed value:</i>	specified keyword or a pair of numbers
<i>Canonical order:</i>	per grammar
<i>Animation type:</i>	discrete

This property sets an intrinsic aspect ratio for the box, which will be used in the calculation of 'auto' sizes and some other layout functions. The box will essentially size the same as a [replaced element](#) with an [intrinsic aspect ratio](#), see e.g. [CSS2§10](#).

'auto'
[Replaced elements](#) with an [intrinsic aspect ratio](#) use that aspect ratio; otherwise the box has no aspect ratio.

'<ratio>'
The box's aspect ratio is the specified ratio.

usecase 1: video



```
<iframe width="560" height="315"
src="https://www.youtube.com/
embed/0Gr1XSyxZy0"
frameborder="0"></iframe>
```

```
iframe {
  aspect-ratio: 16 / 9;
  width: 100%;
  height: auto;
}
```

before you bike shed `aspect-ratio: 16 / 9;`

```
@media (aspect-ratio: 16 / 9) {
```

```
...
```

```
}
```



Media Queries Level 4

W3C Candidate Recommendation

TABLE OF CONTENTS

- 1 Introduction
 - 1.1 Module interactions
 - 1.2 Values
 - 1.3 Units
- 2 Media Queries
 - 2.1 Combining Media Queries
 - 2.2 Media Query Modifiers
 - 2.2.1 Negating a Media Query: the 'not' keyword
 - 2.2.2 Hiding a Media Query From Legacy User Agents: the 'only' keyword
 - 2.3 Media Types
 - 2.4 Media Features
 - 2.4.1 Media Feature Types: "range" and "discrete"
 - 2.4.2 Evaluating Media Features in a Boolean Context
 - 2.4.3 Evaluating Media Features in a Range Context
 - 2.4.4 Using "min-" and "max-" Prefixes On Range Features
 - 2.5 Combining Media Features
- 3 Syntax
 - 3.1 Evaluating Media Queries

[<length>s](#) are interpreted according to [§1.3 Units](#).

['height'](#) is [false](#) in the negative range.

§ 4.3. Aspect-Ratio: the 'aspect-ratio' feature

Name:	' aspect-ratio '
For:	' @media '
Value:	<ratio>
Type:	range

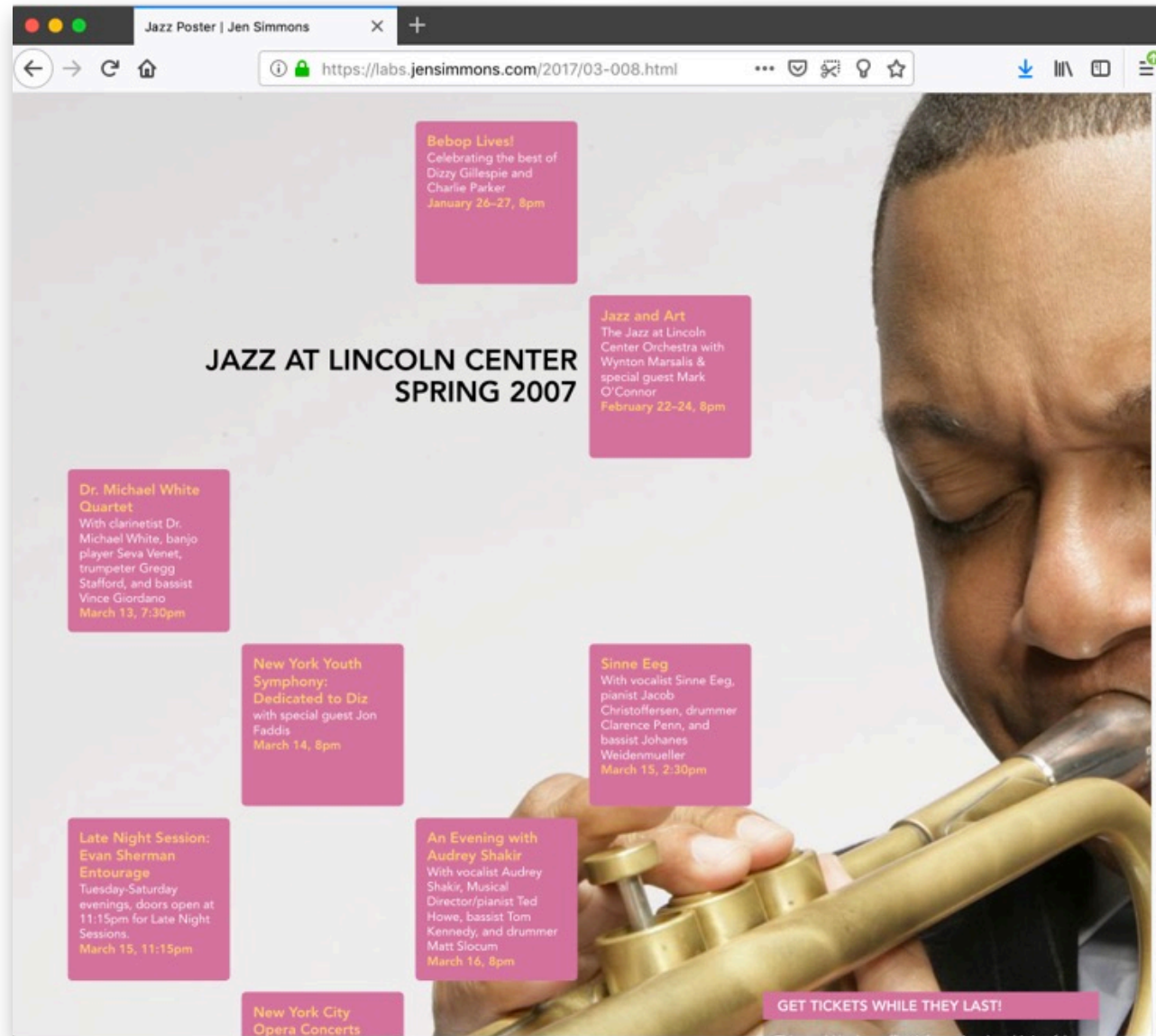
The '[aspect-ratio](#)' media feature is defined as the ratio of the value of the '[width](#)' media feature to the value of the '[height](#)' media feature.

The '[<ratio>](#)' value type is a positive (not zero or negative) [<integer>](#) followed by optional whitespace, followed by a solidus ('/'), followed by optional whitespace, followed by a positive [<integer>](#). [<ratio>s](#) can be ordered or compared by transforming them into the number obtained by dividing their first [<integer>](#) by their second [<integer>](#).

§ 4.4. Orientation: the 'orientation' feature

Name:	' orientation '
-------	---------------------------------

usecase 2: boxes



```
<article>  
  <h2>Title of event</h2>  
  <p>More stuff...</p>  
</article>
```

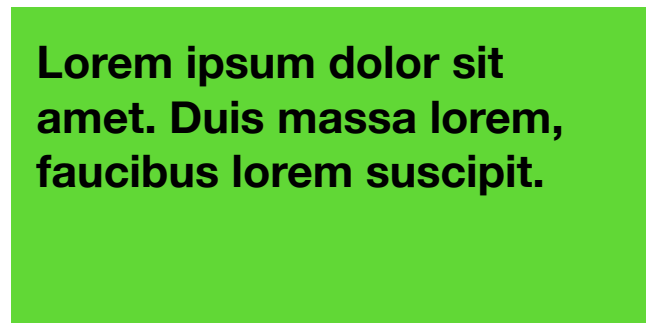
Desire — to get a square most of the time. Unless there's too much content, then the box should grow.

usecase 2: teaser boxes



Lorem ipsum dolor sit
amet, consectetur
adipiscing elit.

Lorem ipsum dolor sit.



Lorem ipsum dolor sit
amet, consectetur
adipiscing elit. Duis massa
lorem, faucibus a suscipit
at, fermentum eget magna.
Quisque id ligula viverra
sem accumsan feugiat.

```
article {  
    aspect-ratio: 2 / 1;  
    width: 100%;  
    height: auto;  
}
```

usecase 2: teaser boxes



Lorem ipsum dolor sit
amet, consectetur
adipiscing elit.

Lorem ipsum dolor sit.

Lorem ipsum dolor sit
amet. Duis massa lorem,
faucibus lorem suscipit.

Lorem ipsum dolor sit
amet, consectetur
adipiscing elit. Duis massa
lorem, faucibus a suscipit
at, fermentum eget magna.
Quisque id ligula viverra
sem accumsan feugiat.

W3C Editor's Draft

TABLE OF CONTENTS

1

Introduction

1.1

Module interactions

1.2

Values

2

Aspect Ratios

2.1

Intrinsic Aspect Ratios: the 'aspect-ratio' property

2.2

Aspect Ratio Limits Option A: the 'from-ratio'

2.3

Aspect Ratio Limits Option B: the 'ar' unit

3

Intrinsic Size Determination

3.1

Intrinsic Sizes of Replaced Elements

3.2

Intrinsic Sizes of Non-Replaced Inlines

3.3

Intrinsic Sizes of Non-Replaced Blocks

3.4

Intrinsic Sizes in Table Layout

3.5

Intrinsic Sizes in Multi-column Layout

3.5.1

Min-content Sizes in Multi-column Layout

3.5.2

Max-content Sizes in Unconstrained-height Multi-column Layout

3.5.3

Max-content Sizes in Constrained-height Multi-column Layout

4

Extrinsic Size Determination

4.1

Stretch-fit Sizing

4.2

Contain-fit Sizing: stretching while maintaining an aspect ratio

4.3

Percentage Sizing

Changes

Acknowledgments

§ 2.2. Aspect Ratio Limits Option A: the 'from-ratio'

Name:

'min-width', 'min-height', 'max-width', 'max-height'

New values:

from-ratio

Computed value:

keyword as specified

The **from-ratio** keyword specifies that the used value of the property is calculated from the used size of the opposite dimension converted through the aspect ratio. If the box has no aspect ratio, then an aspect ratio of 1:1 is assumed.

ISSUE 5

Define a table of all the ways this creates conflicts and cycles and break them.

EXAMPLE 3

In the following example, the box is as wide as the container (as usual), and its height is as tall as needed to contain its content but at least as tall as it is high.

```
div {
  aspect-ratio: 1/1;
  min-height: from-ratio;
  height: max-content;
}
```

solution A

Lorem ipsum dolor sit.

Lorem ipsum dolor sit
amet, consectetur
adipiscing elit.

Lorem ipsum dolor sit
amet, consectetur
adipiscing elit. Duis massa
lorem, faucibus a suscipit
at, fermentum eget magna.
Quisque id ligula viverra
sem accumsan feugiat.

```
article {  
  aspect-ratio: 2 / 1;  
  width: 100%;  
  height: max-content;  
  min-height: from-ratio;  
}
```

CSS Sizing Level 4 Editor's Draft

W3C Editor's Draft

TABLE OF CONTENTS

1

Introduction

1.1

Module interactions

1.2

Values

2

Aspect Ratios

2.1

Intrinsic Aspect Ratios: the 'aspect-ratio' property

2.2

Aspect Ratio Limits Option A: the 'from-ratio'

2.3

Aspect Ratio Limits Option B: the 'ar' unit

3

Intrinsic Size Determination

3.1

Intrinsic Sizes of Replaced Elements

3.2

Intrinsic Sizes of Non-Replaced Inlines

3.3

Intrinsic Sizes of Non-Replaced Blocks

3.4

Intrinsic Sizes in Table Layout

3.5

Intrinsic Sizes in Multi-column Layout

3.5.1

Min-content Sizes in Multi-column Layout

3.5.2

Max-content Sizes in Unconstrained-height Multi-column Layout

3.5.3

Max-content Sizes in Constrained-height Multi-column Layout

4

Extrinsic Size Determination

4.1

Stretch-fit Sizing

4.2

Contain-fit Sizing: stretching while maintaining an aspect ratio

4.3

Percentage Sizing

Changes

Acknowledgments

2.3. Aspect Ratio Limits Option B: the 'ar' unit

Name:

'min-width', 'min-height', 'max-width', 'max-height'

New values:

<aspect-ratio>

Computed value:

'ar' dimension value

The '<aspect-ratio>' value, which is a dimension with the unit 'ar', specifies that the used value of the property is calculated from the used size of the opposite dimension multiplied by the 'ar' value.

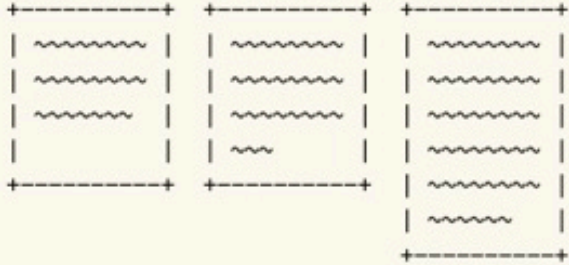
ISSUE 6

Define a table of all the ways this creates conflicts and cycles and break them.

EXAMPLE 4

In the following example, the box is as wide as the container (as usual), and its height is as tall as needed to contain its content but at least as tall as it is high.

```
div {
  min-height: 1ar;
}
```



solution B

Lorem ipsum dolor sit.

Lorem ipsum dolor sit
amet, consectetur
adipiscing elit.

Lorem ipsum dolor sit
amet, consectetur
adipiscing elit. Duis massa
lorem, faucibus a suscipit
at, fermentum eget magna.
Quisque id ligula viverra
sem accumsan feugiat.

```
article {  
  width: 100%;  
  min-height: .5ar;  
}
```

(After this presentation `ar` was renamed to `tr`
pending a better idea. <https://github.com/w3c/csswg-drafts/issues/3225>)

CSS Grid Level 2 Working Draft

W3C Working Draft

TABLE OF CONTENTS

1

Introduction

2

Subgrids

2.1

Establishing a Subgrid

2.2

Characteristics of a Subgrid Item

2.3

Subgrid Sizing Algorithm

3

Aspect-ratio-controlled Gutters

4

Changes

Changes since the June 2018 CSS Grid Layout Level 2 Working Draft

Changes since the April 2018 CSS Grid Layout Level 2 Working Draft

5

Acknowledgements

Conformance

Document conventions

Conformance classes

Requirements for Responsible Implementation of CSS

Partial Implementations

Implementations of Unstable and Proprietary Features

Implementations of CR-level Features

Index

Terms defined by this specification

Terms defined by reference

References

Normative References

§ 3. Aspect-ratio-controlled Gutters

ISSUE 3

There's a desire for having row and column gaps maintain a particular aspect ratio. This is one proposal for doing so; other ideas are welcome. See discussion in [Issue 1116](#). Note this feature is likely to move to css-align-4, it is just being drafted up here while css-align-3 stabilizes.

Name:

['align-content'](#), ['justify-content'](#)

New values:

[[<aspect-ratio>](#) [<content-distribution>?](#)]

['<aspect-ratio>'](#)

A [dimension](#) with the unit ['ar'](#), representing a multiplier against the analogous quantity in the other dimension. If that quantity cannot be determined (e.g. is itself specified as a [<aspect-ratio>](#), or otherwise can't be referenced), then it is assumed to be zero.

Note:

This value can expand gutters even when there is no free space left, causing overflow.

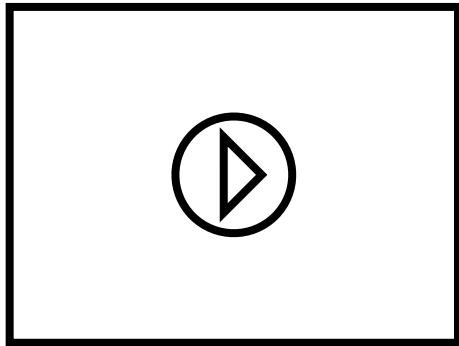
Specifically, an ['align-content'](#) value of ['1ar'](#) represents the amount of space (which may be zero) allocated between two adjacent [alignment subjects](#) ([grid tracks](#) / [flex lines](#) / [column boxes](#)) by the ['justify-content'](#) property. Unless a different [<content-distribution>](#) value is specified, space is distributed according to the same [<content-distribution>](#) rules as for ['justify-content'](#). The behavior of [<number>](#) values for ['justify-content'](#) is analogous.

Note:

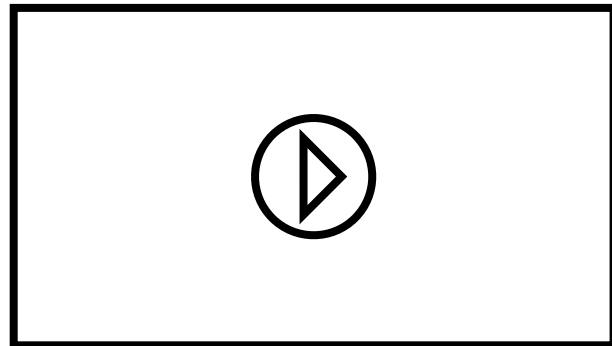
The space allocated by ['align-content: 1ar'](#) will be zero if ['justify-content'](#) does not allocate space between adjacent [alignment subjects](#): either due to not having a [<content-distribution>](#) value or due to there being fewer than two alignment subjects.

If both ['align-content'](#) and ['justify-content'](#) have [<number>](#) values, then ['justify-content'](#)'s [<number>](#) value is ignored and its [<content-distribution>](#) value honored as if specified alone. If no [<content-distribution>](#) value was specified, then ['justify-content'](#) takes ['align-content'](#)'s [<content-distribution>](#) value (if one was specified) and otherwise falls back to ['space-between'](#).

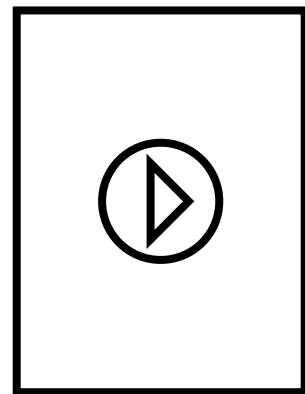
usecase 3: mixed aspect ratio



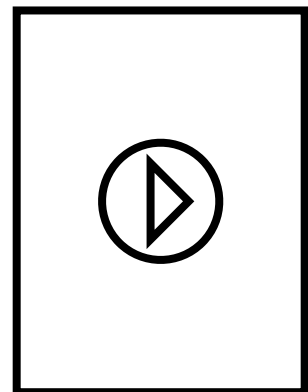
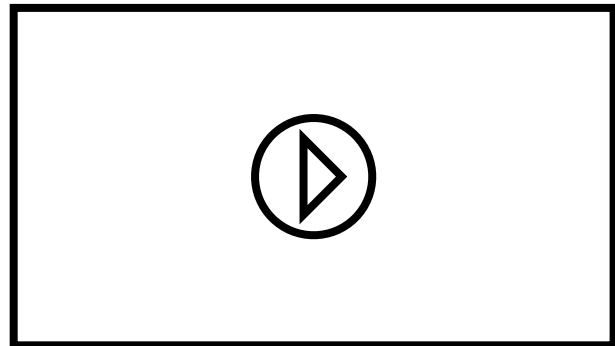
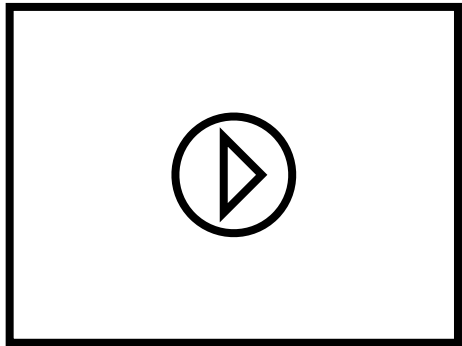
```
<iframe width="560" height="315"  
src="youtube.com/embed/0Gr1XSyxZy0">  
</iframe>
```



```
iframe {  
    aspect-ratio:  
        attr(width px) / attr(height px);  
    width: 100%;  
    height: auto;  
}
```



usecase 3: performance



```

```

```
img {  
  aspect-ratio:  
    attr(width px) / attr(height px);  
}
```

Perhaps add this to the UA default style sheet to avoid the need for extra reflows. (Issue 4 in draft spec)

<https://github.com/w3c/csswg-drafts/issues/333>