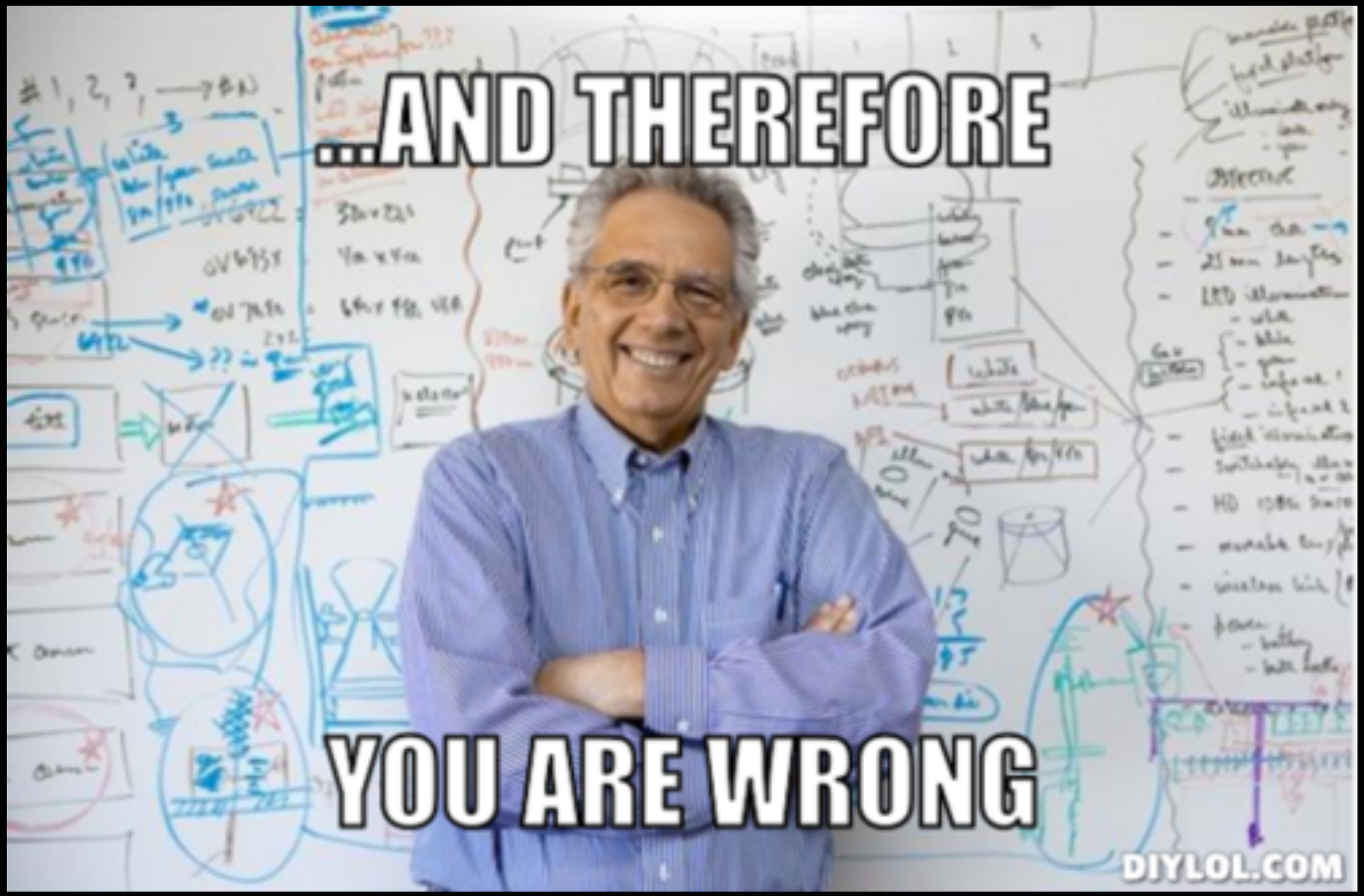
JEN KRAMER
HARVARD UNIVERSITY EXTENSION SCHOOL

CSS4 GRID: TRUE LAYOUT FINALLY ARRIVES


...AND THEREFORE
YOU ARE WRONG
DIYLOL.COM

W3C Working Group Note

§ 2.1. CSS Levels

Cascading Style Sheets does not have versions in the traditional sense; instead it has ***levels***. Each level of CSS builds on the previous, refining definitions and adding features. The feature set of each higher level is a superset of any lower level, and the behavior allowed for a given feature in a higher level is a subset of that allowed in the lower levels. A user agent conforming to a higher level of CSS is thus also conformant to all lower levels.

CSS Level 1

The CSS Working Group considers the CSS1 specification to be obsolete. CSS Level 1 is defined as all the features defined in the CSS1 specification (properties, values, at-rules, etc), but using the syntax and definitions in the CSS2.1 specification. CSS Style Attributes defines its inclusion in element-specific style attributes.

CSS Level 2

Although the CSS2 specification is technically a W3C Recommendation, it passed into the Recommendation stage before the W3C had defined the Candidate Recommendation stage. Over time implementation experience and further review has brought to light many problems in the CSS2 specification, so instead of expanding an already unwieldy errata list, the CSS Working Group chose to define *CSS Level 2 Revision 1* (CSS2.1). In case of any conflict between the two specs CSS2.1 contains the definitive definition.

Once CSS2.1 became Candidate Recommendation—effectively though not officially the same level of stability as CSS2—obsoleted the CSS2 Recommendation. Features in CSS2 that were dropped from CSS2.1 should be considered to be at the Candidate Recommendation stage, but note that many of these have been or will be pulled into a CSS Level 3 working draft, in which case that specification will, once it reaches CR, obsolete the definitions in CSS2.

The CSS2.1 specification defines CSS Level 2 and the CSS Style Attributes specification defines its inclusion in element-specific style attributes.

CSS Level 3

CSS Level 3 builds on CSS Level 2 module by module, using the CSS2.1 specification as its core. Each module adds functionality and/or replaces part of the CSS2.1 specification. The CSS Working Group intends that the new CSS modules will not contradict the CSS2.1 specification: only that they will add functionality and refine definitions. As each module is completed, it will be plugged in to the existing system of CSS2.1 plus previously-completed modules.

From this level on modules are levelled independently: for example Selectors Level 4 may well be completed before CSS Line Module Level 3. Modules with no CSS Level 2 equivalent start at Level 1; modules that update features that existed in CSS Level 2 start at Level 3.

***CSS Level 4* and beyond**

There is no CSS Level 4. Independent modules can reach level 4 or beyond, but CSS the language no longer has levels. ("CSS Level 3" as a term is used only to differentiate it from the previous monolithic versions.)

"There is no CSS Level 4… CSS the language no longer has levels."

https://www.w3.org/TR/css-2015/#css-levels

W3C Set to Publish HTML 5.1, Work Already Started on HTML 5.2

HTML 5.1 taking its last step before becoming "the standard"

Sep 21, 2016 21:05 GMT · By Catalin Cimpanu · Share:

Members of the World Wide Web Consortium (W3C) are getting ready to launch the HTML 5.1 specification and have already started work on the upcoming HTML 5.2 version since mid-August.

The HTML 5.1 standard has been promoted from a "Release Candidate" to a "Proposed Recommendation," the last step before it becomes a "W3C Recommendation" and officially replaces HTML 5 as the current HTML standard.

http://news.softpedia.com/news/w3c-set-to-publish-html-5-1-already-started-work-on-html-5-2-508512.shtml

CSS

+

HTML

JEN KRAMER
HARVARD UNIVERSITY EXTENSION SCHOOL

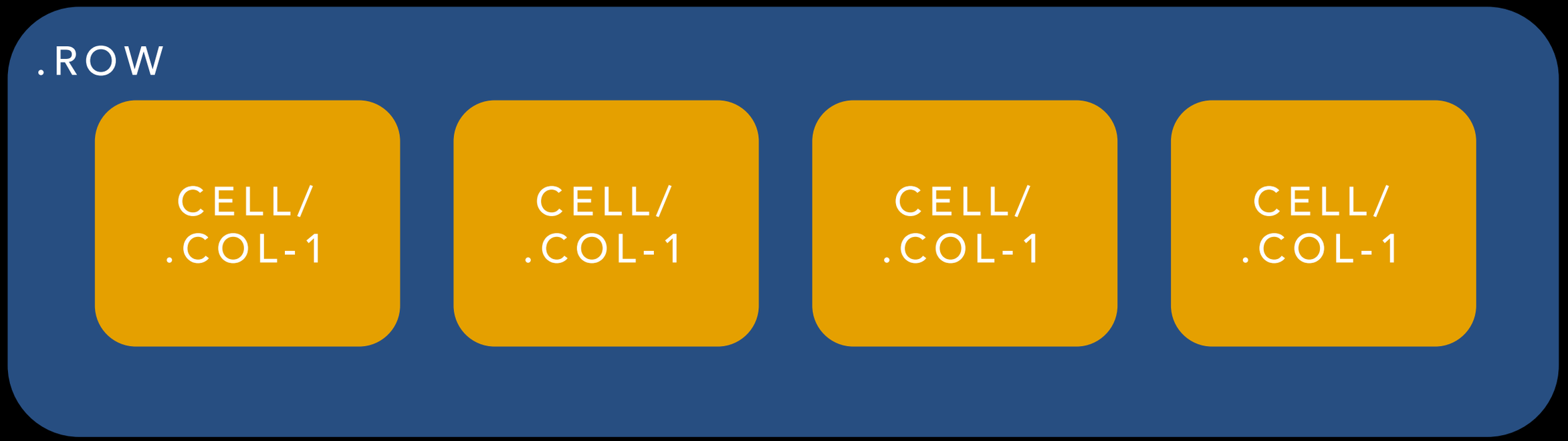
CSSX GRID: TRUE LAYOUT FINALLY ARRIVES

PART 1

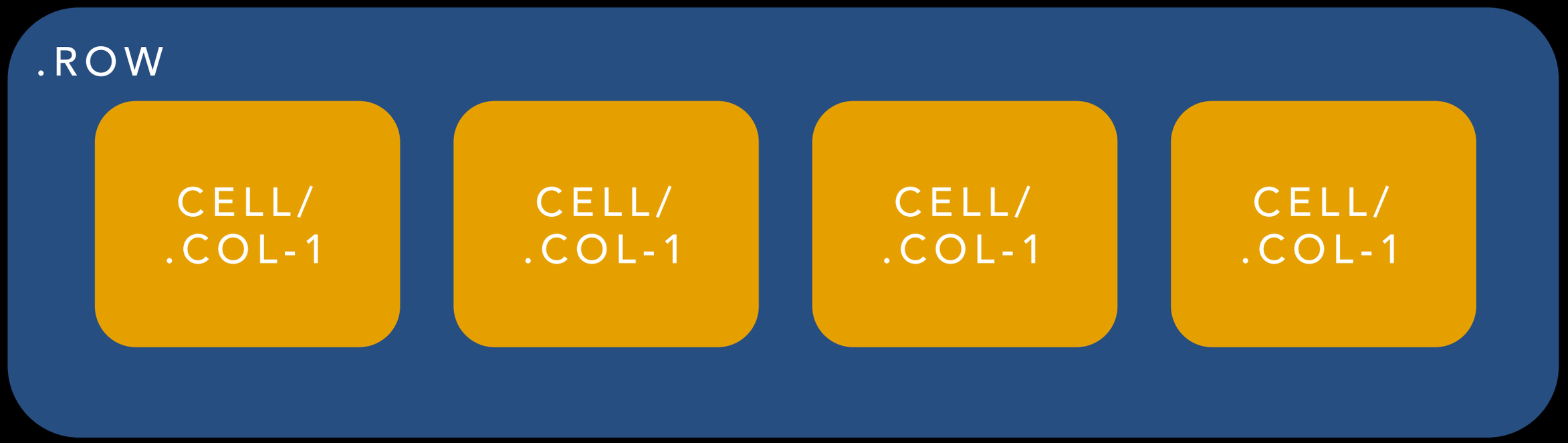
RESPONSIVE DESIGN

FLOATS

- A hack from the start, right after table-based layout!
- Features rows and cells.
- Rows clear the floats on the cells.
- Source ordering determines display, though some (minor) rearrangement is possible.
- Major disadvantage: equal column heights

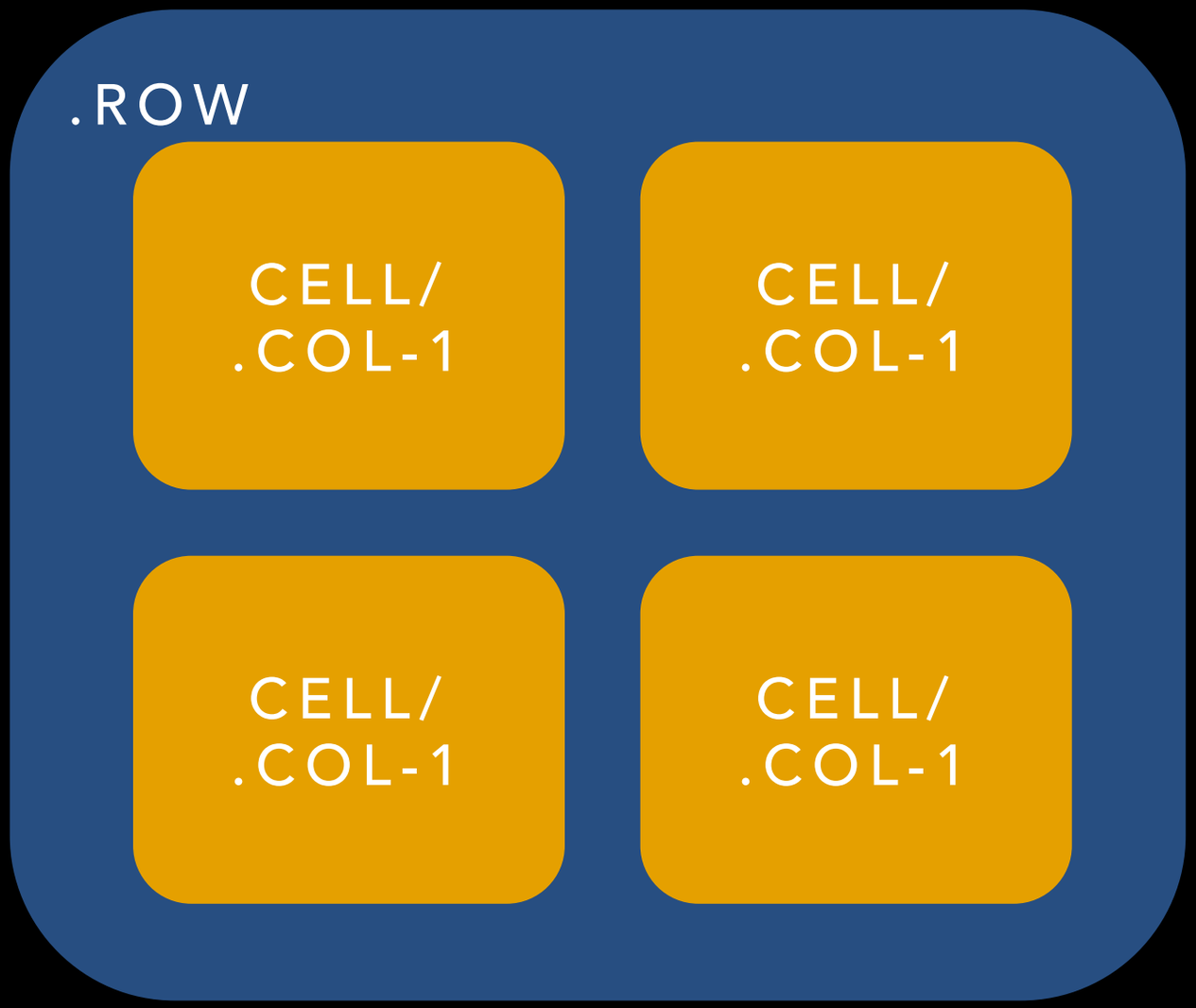



```
<div class=“row”>
      <div class=“col-1”></div>
      <div class=“col-1”></div>
      <div class=“col-1”></div>
      <div class=“col-1”></div>
</div>
```

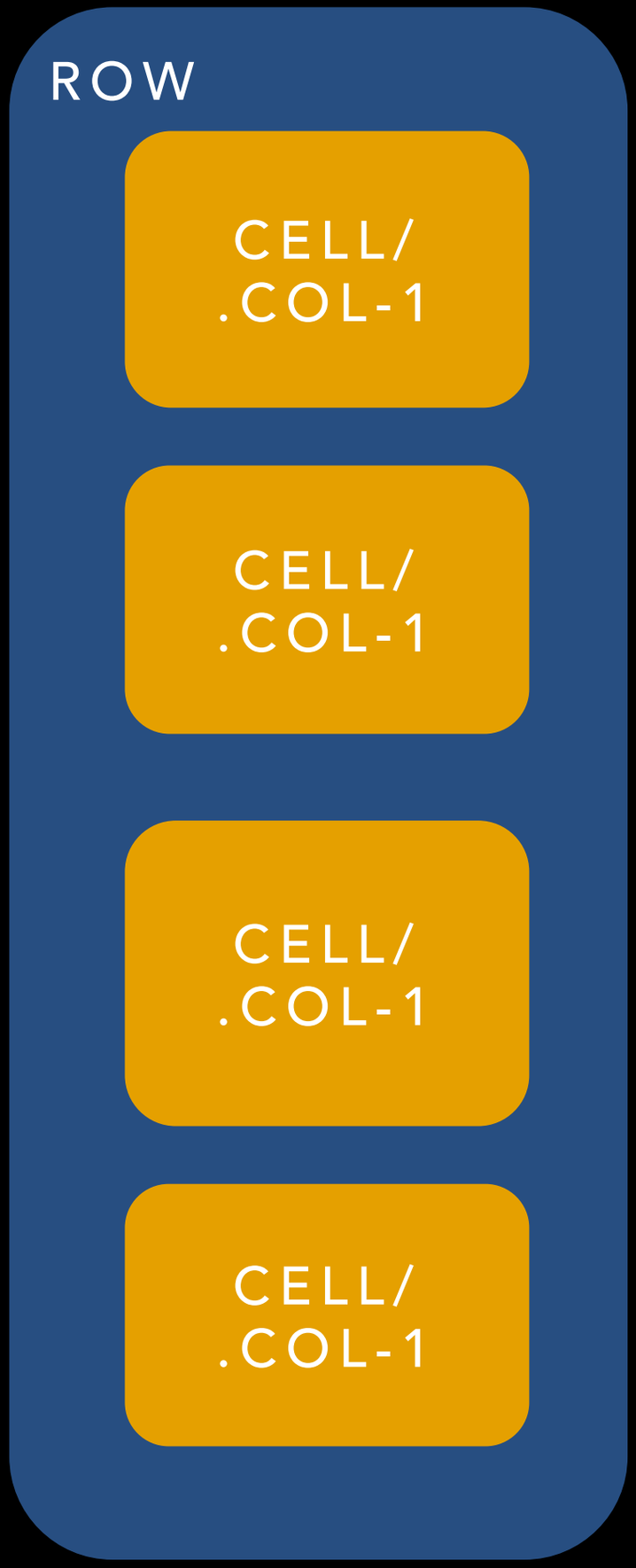



```
.row::after {
    content: "";
    display: table;
    clear: both;
}
```

```
.col-1 {
    float: left;
    margin-left: 4%;
    width: 20%;
}
```

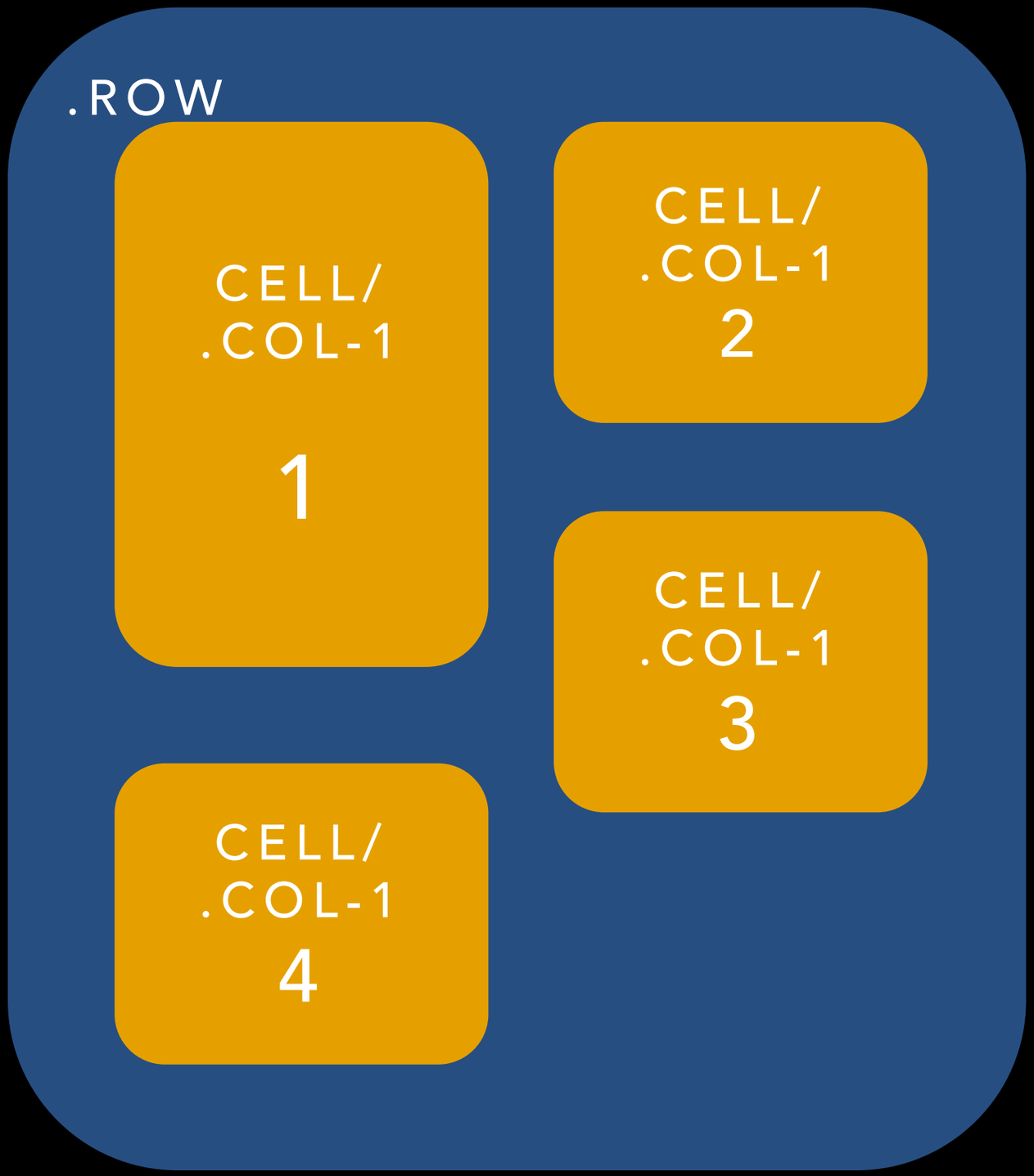



```
@media only screen and (min-width: 480px)
                       and (max-width: 767px) {
    .col-1 {
       width: 44%;
    }
}
```




```
@media only screen and
        (max-width: 479px) {

    .col-1 {
        width: 98%;
        margin: 1%;
        float: none;
    }
}
```



There can be layout problems with floats.

This can be resolved with JavaScript, with a column equalizer script.

```
/* rearranging the columns */
[class*="col-"] {
    position: relative;
}
.col-push-1 {
    left: 26%;
}
.col-pull-3 {
    left: -74%;
}
```

NEXT UP

FLEXBOX

FLEXBOX

- The first layout elements – but not designed to lay out whole web pages
- Features flex-containers (row) and flex-items (cells). Both are required to work.
- Excels at vertical centering and equal heights
- Very easy to reorder boxes
- Major disadvantages:
  - Wasn’t designed to be locked down for layouts! Works in 1 dimension only.
  - Browser support and syntax is challenging.


```
<div class=“row”>
      <div class=“col-1”></div>
      <div class=“col-1”></div>
      <div class=“col-1”></div>
      <div class=“col-1”></div>
</div>
```

```
.row {
  display: flex;
  flex-flow: row wrap;
  justify-content: center;
  margin: 1%;
}
```

Change flex-flow to other values to change direction of rows – row reverse, column reverse, no wrap

```
.col-1 {
    flex: 0 0 24%; /* desktop */
}
    flex: 0 0 48%; /* tablet */
}
    flex: 0 0 98%; /* phone */
}
```

To change widths on .col-1, change the flex-basis property. This is more flexible than width.

```
/* rearranging the columns */

.col-push-1 {
  order: 2;
}
.col-pull-3 {
  order: 1;
}
```

FINALLY!

CSS GRID

WHY CSS GRID?

- Built into CSS specification (in development).
- No “row” markup required.
- Grid is designed to work in 2 dimensions.
- Use Flexbox for UI elements, but use Grid for major layout.

Figure 1 *Exemplary Flex Layout Example*

Figure 2 *Exemplary Grid Layout Example*

https://drafts.csswg.org/css-grid/

ISSUES, WE'VE HAD A FEW

- Now supported in latest versions of Firefox and Chrome – with Edge to follow shortly
- Old grid spec supported in IE 10 and 11 (will never be updated to new spec)
- Lots of bugs to go around
- http://gridbyexample.com/browsers/ -- latest info

CSS Grid Layout - CR

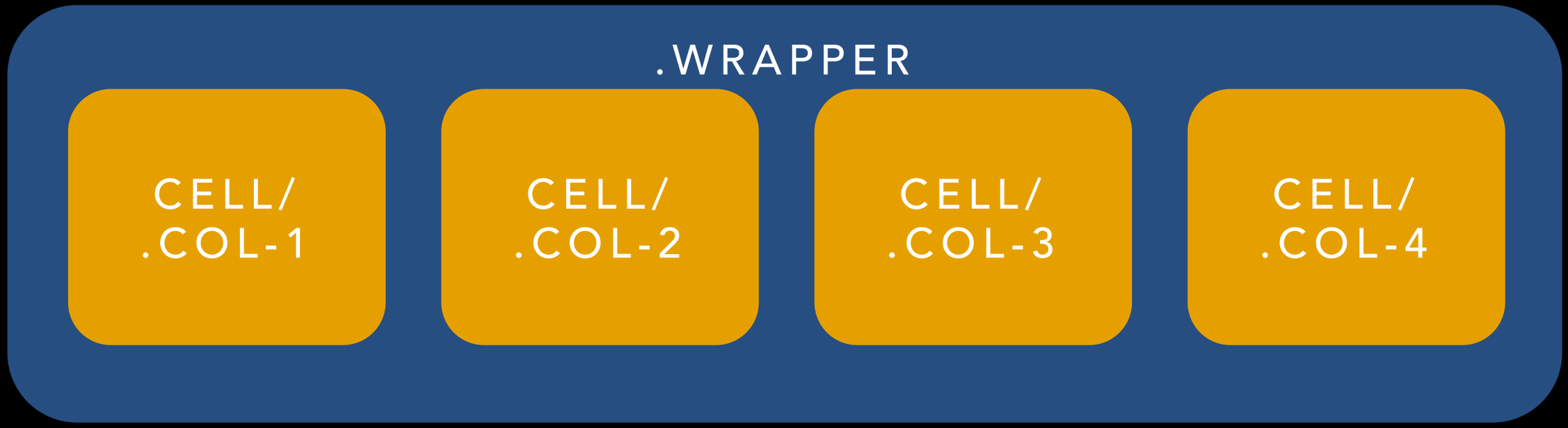
Method of using a grid concept to lay out content, providing a mechanism for authors to divide available space for lay out into columns and rows using a set of predictable sizing behaviors

Global 31.32% + 5.69% = 37.01%

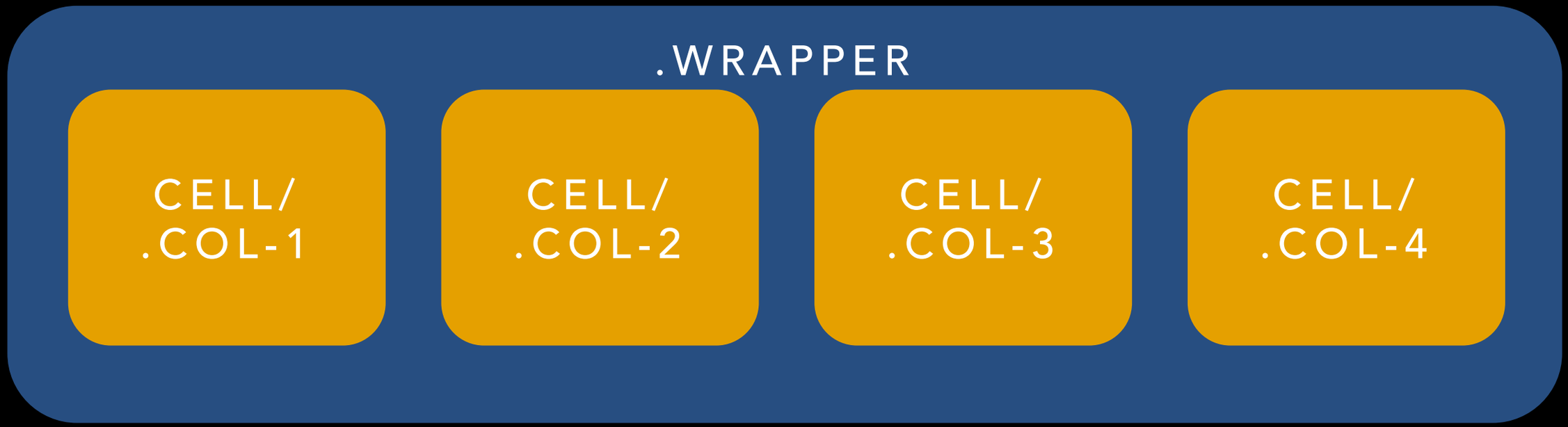
unprefixed: 31.32%

Current aligned | Usage relative | Date relative | Show all

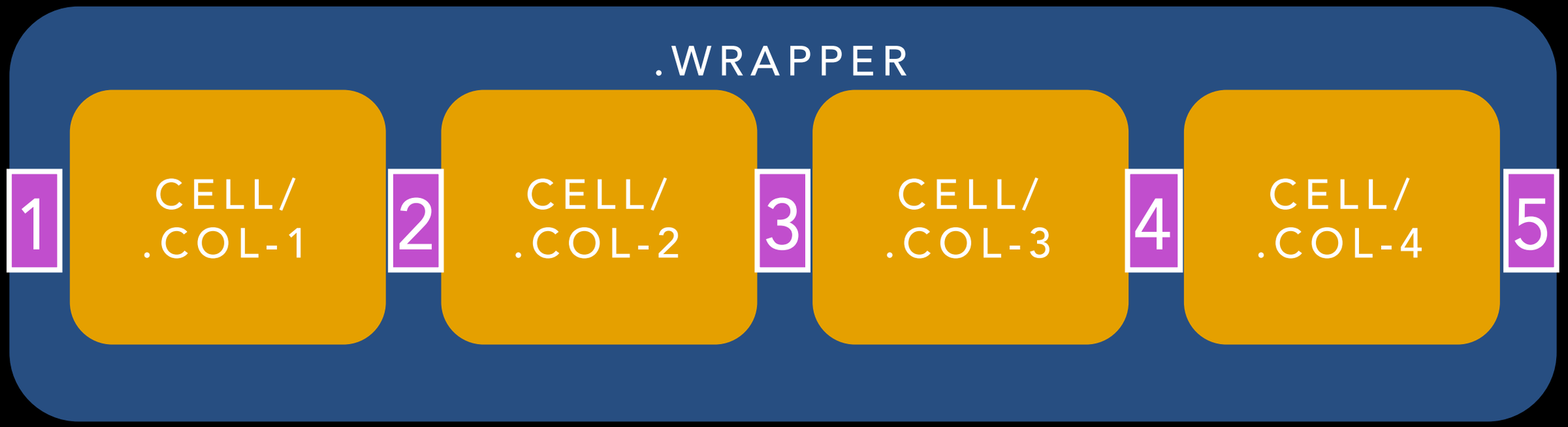
IE	Edge *	Firefox	Chrome	Safari	Opera	iOS Safari *	Opera Mini *	Android Browser *	Chrome for Android
			1 49						
			1 55			9.3		4.4	
	2 14 −	3 51	1 56	10	1 43	10.2		4.4.4	
2 11 −	2 15 −	52	4 57	10.1	44	10.3	all	56	4 57
		53	58	TP	45				
		54	59		46				
		55	60						



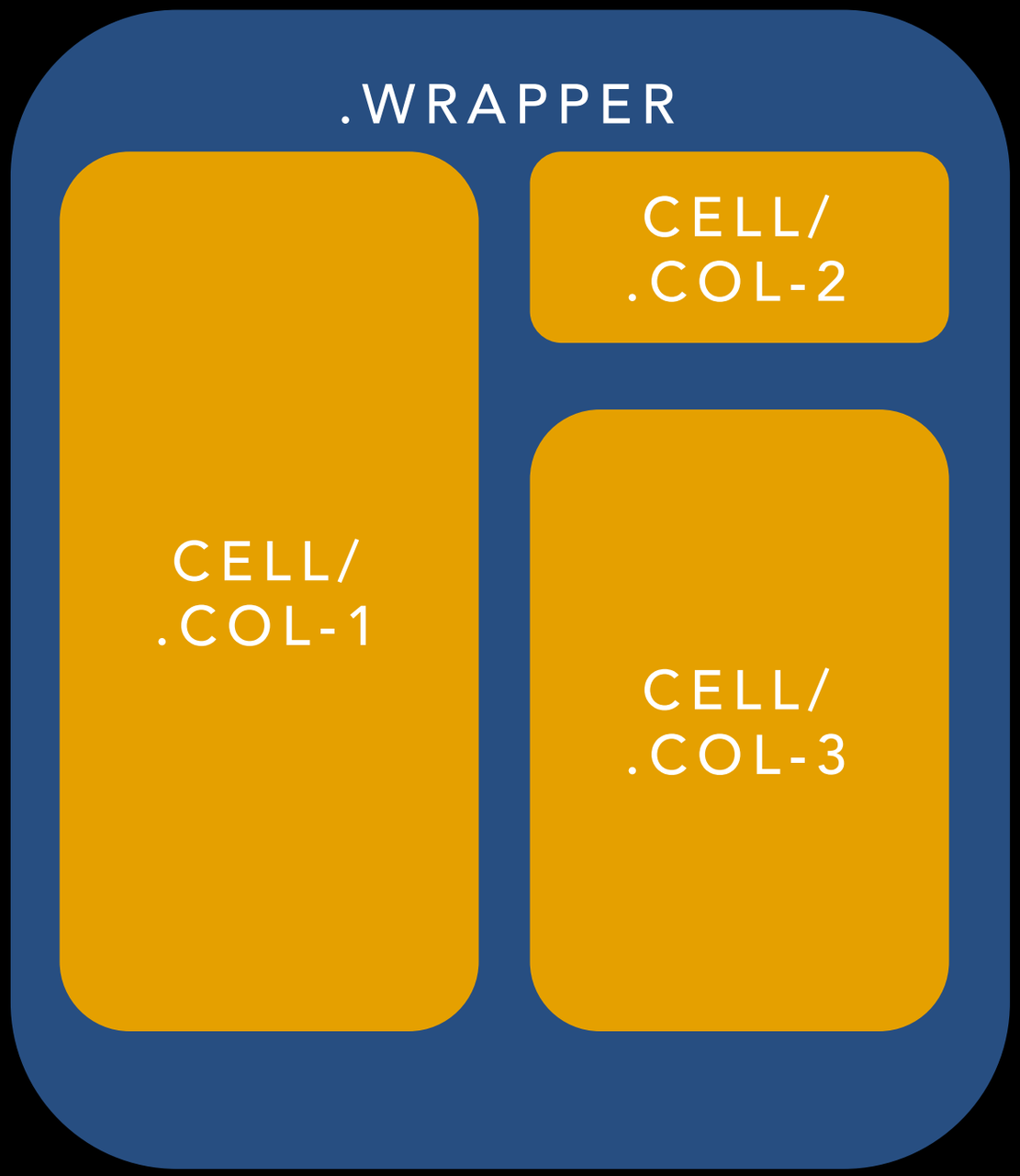

```
<div class=“wrapper”>
  <div class=“col-1”></div>
  <div class=“col-2”></div>
  <div class=“col-3”></div>
  <div class=“col-4”></div>
</div>
```




```
.wrapper {
  display: grid;
  grid-gap: 10px;
}
```




```
.col-1 {
  grid-column: 1 / 2;
}
.col-2 {
  grid-column: 2 / 3;
}
.col-3 {
  grid-column: 3 / 4;
}
```

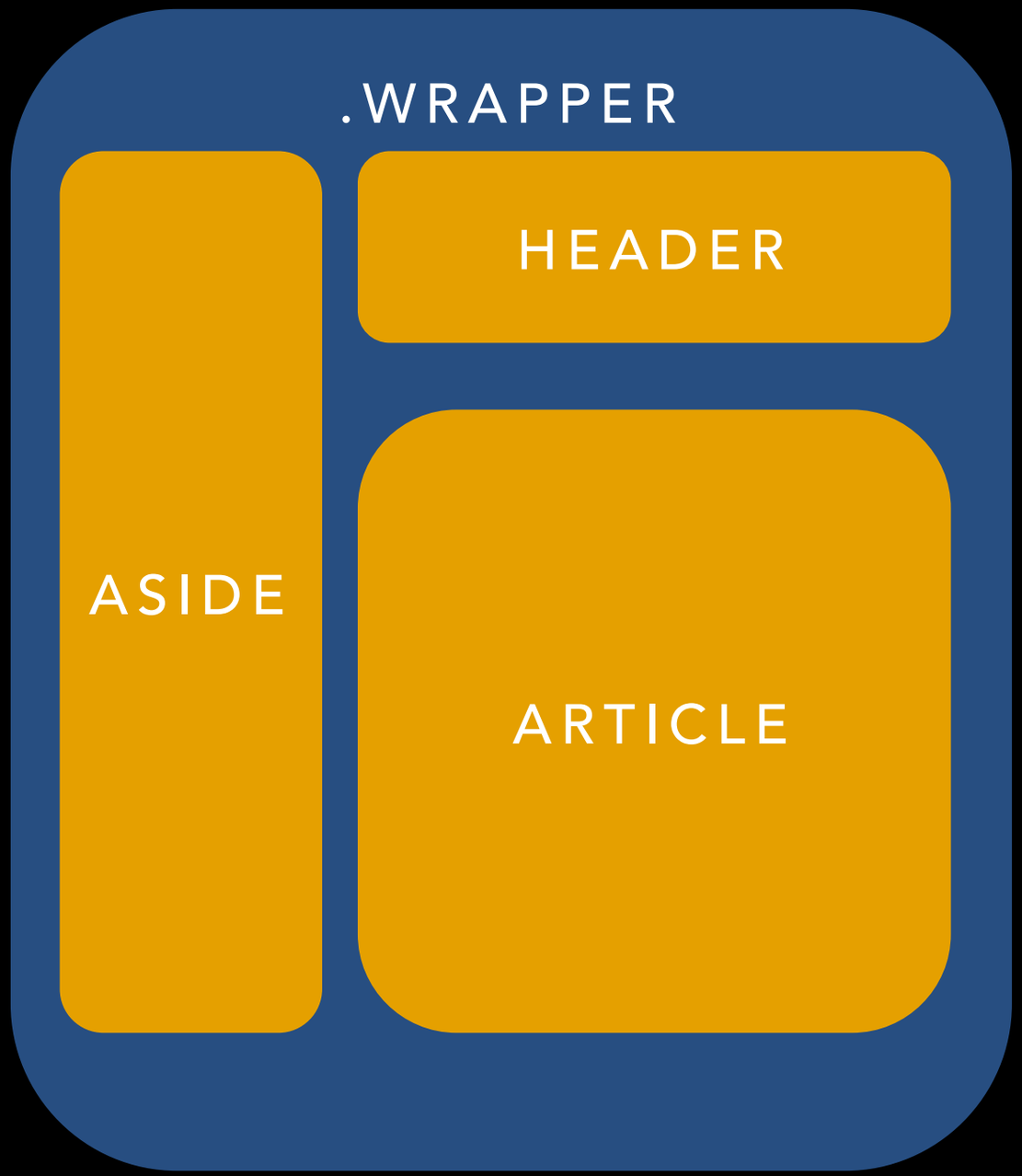



```
.col-1 {
  grid-column: 1 / 2;
  grid-row: 1 / 3;
}
.col-2 {
  grid-column: 2 / 3;
  grid-row: 1 / 2;
}
.col-3 {
  grid-column: 2 / 3;
  grid-row: 2 / 3;
}
```

ALTERNATE SYNTAX

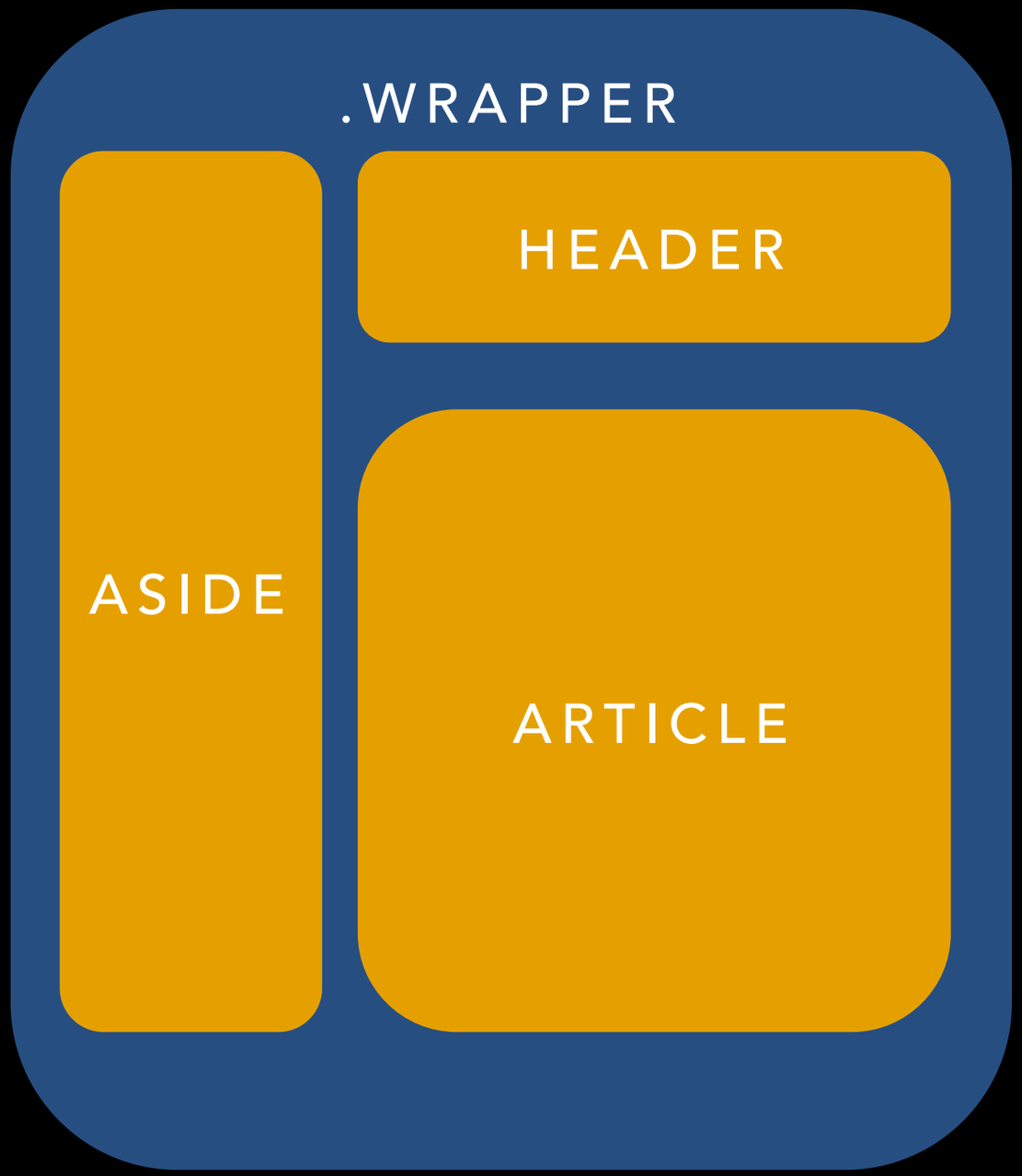
- Named grid template areas (header, footer, etc):
http://gridbyexample.com/examples/#example11

REORDERING




```
<div class="wrapper">
    <header></header>
    <article></article>
    <aside></aside>
</div>
```

REORDERING



Show code in GitHub!

reordering.html

reordering.css

https://github.com/jen4web/cssgrid

CURRENT RECOMMENDATIONS

- Need browser support? Can't go wrong with floats.
- If you are OK with a few prefixes and learning something new, Flexbox may be a reasonable interim step.
  - PS – it's in Bootstrap 4. Get used to it.
- Keep waiting for CSS Grid! Coming to a browser near you Real Soon!