How CSS evolves, a look back at 2 years of CSS grid

Chen Hui Jing / @hj_chen
Surname: Chen
First name: Hui Jing

@hj_chen
Developers

Developer Advocate

nexmo
The Vonage API Platform
WORLD WIDE WEB

The WorldWideWeb (W3) is a wide-area hypermedia[1] information retrieval initiative aiming to give universal access to a large universe of documents.

Everything there is online about W3 is linked directly or indirectly to this document, including an executive summary[2] of the project, Mailing lists[3], Policy[4], November's W3 news[5], Frequently Asked Questions[6].

What's out there?[7] Pointers to the world's online information, subjects[8], W3 servers[9], etc.

Help[10] on the browser you are using

Software Products[11] A list of W3 project components and their current state. (e.g. Line Mode[12], X11 Viola[13], NeXtStep[14], Servers[15], Tools[16], Mail robot[17], Library[18])

Technical[19] Details of protocols, formats, program internals, etc.

<ref.number>, Back, <RETURN> for more, or Help:
Web layouts over the years

- No layout
- HTML Tables
- CSS Floats
- Frameworks
- Grid and beyond
### CSS Evolution

**CSS1**  
Recommendation: 17 Dec 1996

**CSS2**  
Recommendation: 12 May 1998

**CSS2.1**  
Recommendation: 7 Jun 2011

**CSS2.2**  
Working draft: 12 Apr 2016

**CSS3**  
Decision to modularise: 14 Apr 2000  
(26 modules)

### CSS Snapshot 2018  
(99 modules)

<table>
<thead>
<tr>
<th>Completed</th>
<th>Stable</th>
<th>Revising</th>
<th>Refining</th>
<th>Testing</th>
<th>Exploring</th>
</tr>
</thead>
<tbody>
<tr>
<td>CSS Snapshot 2018</td>
<td>CSS Backgrounds and Borders Level 3</td>
<td>CSS Paginated Media Level 3</td>
<td>CSS Animations Level 1</td>
<td>CSS Images Level 3</td>
<td>CSS Backgrounds and Borders Level 4</td>
</tr>
<tr>
<td>CSS Snapshot 2017</td>
<td>CSS Conditional Rules Level 3</td>
<td>CSS MDOM View</td>
<td>CSS Speech</td>
<td>CSS Speech</td>
<td>CSS Device Adaptation</td>
</tr>
<tr>
<td>CSS Snapshot 2015</td>
<td>CSS Multi-column Layout Level 1</td>
<td>CSS intrinsic &amp; Extrinsic Sizing Level 3</td>
<td>CSS Text Level 3</td>
<td>CSS Text Decoration Level 3</td>
<td>CSS Device Adaptation</td>
</tr>
<tr>
<td>CSS Snapshot 2010</td>
<td>CSS Values and Units Level 3</td>
<td>CSS Ruby Level 1</td>
<td>CSS Transforms Level 1</td>
<td>CSS Shapes Level 1</td>
<td>CSS Device Adaptation</td>
</tr>
<tr>
<td>CSS Snapshot 2007</td>
<td>CSS Cascading and Inheritance Level 3</td>
<td>CSS Box Model Level 3</td>
<td>CSS Transitions</td>
<td>CSS Masking Level 1</td>
<td>CSS Device Adaptation</td>
</tr>
<tr>
<td>CSS Color Level 3</td>
<td>CSS Fonts Level 3</td>
<td>CSS Box Model Level 3</td>
<td>CSS Box Alignment Level 3</td>
<td>CSS Fragmentation Level 3</td>
<td>CSS Device Adaptation</td>
</tr>
<tr>
<td>CSS Navigator</td>
<td>CSS Writing Modes Level 3</td>
<td>Selectors Level 4</td>
<td>CSS Cascading Variables</td>
<td>CSS Grid Layout Level 1</td>
<td>CSS Device Adaptation</td>
</tr>
<tr>
<td>CSS Level 2</td>
<td>CSS Counter Styles Level 3</td>
<td>CSS Lists Level 3</td>
<td>Compositing and Blending Level 1</td>
<td>CSS Grid Layout Level 1</td>
<td>CSS Device Adaptation</td>
</tr>
<tr>
<td>CSS Containment Level 1</td>
<td>CSS Wiring Level 4</td>
<td>Motion Path Level 1</td>
<td>CSS Syntax Level 3</td>
<td>CSS Grid Layout Level 1</td>
<td>CSS Device Adaptation</td>
</tr>
<tr>
<td>Media Queries</td>
<td>CSS Generated Content Level 3</td>
<td>Preview of CSS Level 2</td>
<td>CSS Grid Layout Level 1</td>
<td>CSS Grid Layout Level 1</td>
<td>CSS Device Adaptation</td>
</tr>
<tr>
<td>CSS Style Attributes</td>
<td>CSS Easing Functions Level 1</td>
<td>CSS Grid Layout Level 1</td>
<td>CSS Display Level 3</td>
<td>CSS Grid Layout Level 1</td>
<td>CSS Device Adaptation</td>
</tr>
<tr>
<td>CSS Fonts Level 3</td>
<td>CSS Logical Properties and Values Level 1</td>
<td>CSS Will Change Level 1</td>
<td>CSS Grid Layout Level 1</td>
<td>CSS Grid Layout Level 1</td>
<td>CSS Device Adaptation</td>
</tr>
<tr>
<td>CSS Basic User</td>
<td>CSS Grid Layout Level 1</td>
<td>Media Queries Level 4</td>
<td>CSS Layout 1</td>
<td>CSS Grid Layout Level 1</td>
<td>CSS Device Adaptation</td>
</tr>
<tr>
<td>Interface Level 3</td>
<td>CSS Grid Layout Level 1</td>
<td>Geometry Interfaces Level 1</td>
<td>CSS Scrolling Level 1</td>
<td>CSS Grid Layout Level 1</td>
<td>CSS Device Adaptation</td>
</tr>
</tbody>
</table>

### Completed

- CSS Snapshot 2018
- CSS Snapshot 2017
- CSS Snapshot 2015
- CSS Snapshot 2010
- CSS Snapshot 2007
- CSS Color Level 3
- CSS Navigator
- CSS Level 2
- CSS Containment Level 1
- Media Queries
- CSS Style Attributes
- CSS Fonts Level 3
- CSS Basic User
- Interface Level 3

### Stable

- CSS Backgrounds and Borders Level 3
- CSS Conditional Rules Level 3
- CSS Multi-column Layout Level 1
- CSS Values and Units Level 3
- CSS Cascading and Inheritance Level 3
- CSS Fonts Level 3
- CSS Writing Modes Level 3
- CSS Counter Styles Level 3
- CSS Containment Level 1
- CSS Navigator
- CSS Level 2
- CSS Containment Level 1

### Revising

- CSS Paginated Media Level 3
- CSS MDOM View
- CSS intrinsic & Extrinsic Sizing Level 3
- CSS Ruby Level 1
- CSS Box Model Level 3
- CSS Pseudo-Elements Level 4
- CSS Scrollbars Level 1

### Refining

- CSS Animations Level 1
- Web Animations
- CSS Text Level 3
- CSS Transforms Level 1
- CSS Transitions
- CSS Box Alignment Level 3
- Selectors Level 4
- CSS Lists Level 3
- Motion Path Level 1
- Preview of CSS Level 2
- CSS Grid Layout Level 1
- CSS Easing Functions Level 1
- CSS Logical Properties and Values Level 1
- CSS Wiring Level 3
- CSS Box Model Level 3

### Testing

- CSS Images Level 3
- CSS Speech
- CSS Text Decoration Level 3
- CSS Shapes Level 1
- CSS Masking Level 1
- CSS Fragmentation Level 3
- CSS Grid Layout Level 1
- Compositing and Blending Level 1
- CSS Syntax Level 3
- CSS Grid Layout Level 1
- CSS Display Level 3
- CSS Will Change Level 1
- Media Queries Level 4
- Geometry Interfaces Level 1
- CSS Cascading and Inheritance Level 4
- CSS Scroll Snap Level 1
- CSS Painting API Level 1
- CSS Writing Modes Level 4

### Exploring

- CSS Backgrounds and Borders Level 4
- CSS Device Adaptation
- CSS Exclusions
- Filter Effects Level 1
- CSS Generated Content for Paged Media
- CSS Page Floats
- CSS Template Layout
- CSS Grid Layout 1
- CSS Positioned Layout Level 3
- CSS Regions
- CSS Table Level 3
- CSS Object Model
- CSS Font Loading
- CSS Scrolling Level 1
- CSS Tiling Layout Level 3
- CSS Round Display Level 3
- CSS Basic User Interface Level 4
- CSS Text Level 4
- CSS Properties and Values API Level 1
- CSS Typed OM Level 1
- CSS Worklets Level 1
- CSS Color Level 4
- CSS Rhythmic Sizing Level 1
- CSS Image Values and Replaced Content Level 4
- CSS Fill and Stroke Level 3
- CSS Overflow Level 4
- CSS Grid Layout Level 2
- CSS Layout 1
- CSS Layout API Level 1
- CSS Values and Units Level 4
- CSS Shadow Effects
- CSS Fragmentation Level 4
- CSS Spatial Navigation Level 1
- CSS Color Adjustment Level 1
- CSS Overflow API Level 1
- CSS Animation Worklet API
- CSS Containment Level 2
CSS Snapshot 2018
(99 modules)

**Completed**
- CSS Snapshot 2018
- CSS Snapshot 2017
- CSS Snapshot 2015
- CSS Snapshot 2010
- CSS Snapshot 2007
- CSS Color Level 3
- CSS Namespaces
- Selectors Level 3
- CSS Level 2 Revision 1
- Media Queries
- CSS Style Attributes
- CSS Fonts Level 3
- CSS Basic User Interface Level 3

**Stable**
- CSS Backgrounds and Borders Level 3
- CSS Conditional Rules Level 3
- CSS Multi-column Layout Level 1
- CSS Values and Units Level 3
- CSS Cascading and Inheritance Level 3
- CSS Fonts Level 3
- CSS Writing Modes Level 3
- CSS Counter Styles Level 3
- CSS Containment Level 1

**Exploring**
- CSS Backgrounds and Borders Level 4
- CSS Device Adaptation
- CSS Exclusions
- Filter Effects Level 1
- CSS Generated Content for Paged Media
- CSS Page Floats
- CSS Template Layout
- CSS Line Grid
- CSS Positioning Layout Level 3
- CSS Regions
- CSS Table Level 3
- CSS Object Model
- CSS Font Loading
- CSS Scoping Level 1
- CSS Inline Layout Level 3
- CSS Round Display Level 1
- CSS Basic User Interface Level 4
- CSS Text Level 4
- CSS Properties and Values API Level 1
- CSS Typed OM Level 1
- CSS Worklets Level 1
- CSS Color Level 4
- CSS Rhythmic Sizing Level 1
- CSS Image Values and Replaced Content Level 4
- CSS Fill and Stroke Level 3
- CSS Overflow Level 4
- CSS Grid Layout Level 2
- CSS Text Decoration Level 4
- CSS Layout API Level 1
- CSS Values and Units Level 4
- CSS Shadow Parts
- CSS Fragmentation Level 4
- CSS Spatial Navigation Level 1
- CSS Color Adjustment Level 1
- CSS Overscroll Behavior Level 1
- CSS Animation Worklet API
- CSS Containment Level 2

**Rewriting**
- CSS Generated Content Level 3

**Revising**
- CSS Paged Media Level 3
- CSSOM View
- CSS Intrinsic & Extrinsic Sizing Level 3
- CSS Ruby Level 1
- CSS Overflow Level 3
- CSS Box Model Level 3
- CSS Pseudo-Elements Level 4
- CSS Scrollbars Level 1

**Refining**
- CSS Animations Level 1
- Web Animations
- CSS Text Level 3
- CSS Transforms Level 1
- CSS Transitions
- CSS Box Alignment Level 3
- Selectors Level 4
- CSS Lists Level 3
- Motion Path Level 1
- Preview of CSS Level 2
- CSS Fonts Level 4
- CSS Easing Functions Level 1
- CSS Logical Properties and Values Level 1

**Testing**
- CSS Images Level 3
- CSS Speech
- CSS Text Decoration Level 3
- CSS Shapes Level 1
- CSS Masking Level 1
- CSS Fragmentation Level 3
- CSS Cascading Variables
- Compositing and Blending Level 1
- CSS Syntax Level 3
- CSS Grid Layout Level 1
- CSS Display Level 3
- CSS Will Change Level 1
- Media Queries Level 4
- Geometry Interfaces Level 1
- CSS Cascading and Inheritance Level 4
- CSS Scroll Snap Level 1
- CSS Painting API Level 1
- CSS Writing Modes Level 4
Block layout
- designed for laying out documents

Inline layout
- designed for laying out text

Table layout
- designed for laying out 2D data in a tabular format

Positioned layout
- designed for very explicit positioning without much regard for other elements in the document
Flexible box layout describes a CSS box model optimised for user interface design and designed for laying out more complex applications and webpages.
Grid layout

Defines a two-dimensional grid-based layout system, optimised for user interface design.
Why is CSS grid special?

"Grid works from the container in, other layout methods start with the item."

Rachel Andrew
### Layout technique: inline-block

<table>
<thead>
<tr>
<th>Item A</th>
<th>Item B</th>
<th>Item C</th>
</tr>
</thead>
<tbody>
<tr>
<td>Item D</td>
<td>Item E</td>
<td>Item F</td>
</tr>
</tbody>
</table>

```css
.inlineblock__item {
  display: inline-block;
  width: calc(100% / 3);
}
```
## Layout technique: float

<table>
<thead>
<tr>
<th>Item A</th>
<th>Item B</th>
<th>Item C</th>
</tr>
</thead>
<tbody>
<tr>
<td>Item D</td>
<td>Item E</td>
<td>Item F</td>
</tr>
</tbody>
</table>

```
.float__item {
  float: left;
  width: calc(100% / 3);
}
```
### Layout technique: flex

<table>
<thead>
<tr>
<th>Item A</th>
<th>Item B</th>
<th>Item C</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Item D</th>
<th>Item E</th>
<th>Item F</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```css
.flexbox {
  display: flex;
  flex-wrap: wrap;
}

.flexbox__item {
  flex: 0 0 calc(100% / 3);
}
```
"Grid is the only layout technique that establishes a relationship between rows and columns of grid items."
Grid is the relationship between grid items.
Grid versus Responsive Flexbox?
Grid AND Flexbox
Flexbox ➔ single dimension

flex-direction: row

flex-direction: column
Grid → two dimensions

*Items know about row and column sizing*
Where does CSS come from?

- Browser vendors
- Companies who need certain features
- CSS working group
The CSS Working Group
<table>
<thead>
<tr>
<th>Specification document statuses</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Working Draft (WD)</strong></td>
</tr>
<tr>
<td>to create a snapshot of the specification's current state and to solicit input from the W3C and the public</td>
</tr>
</tbody>
</table>
It's not a linear process...

W3C Snakes and Ladders WD
“...it shouldn’t come as a shock that grid-based layouts have been a goal of CSS since the beginning.

The Story of CSS Grid, from Its Creators by Aaron Gustafson"
Fun facts about CSS grid

- Numerous proposals submitted that were never implemented
- First proposal that stuck was one from Microsoft (for Metro UI)
- First implementation of Grid ever was in IE10 (with vendor prefix)
- Bloomberg hired Igalia to implement CSS grid for Blink and WebKit
- Rachel Andrew proposed gap to solve the gutter problem
Browser support for Grid (Level 1)

**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>Browser</th>
<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 Browser</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>16</td>
<td>69</td>
<td>76</td>
<td>12</td>
<td>12.4</td>
<td></td>
<td></td>
<td></td>
<td>4.4</td>
<td>8.2</td>
</tr>
<tr>
<td>10</td>
<td>17</td>
<td>70</td>
<td>77</td>
<td>12.1</td>
<td>13.1</td>
<td></td>
<td></td>
<td></td>
<td>4.44</td>
<td>9.2</td>
</tr>
<tr>
<td>11</td>
<td>18</td>
<td>71</td>
<td>78</td>
<td>13</td>
<td>13.2</td>
<td>all</td>
<td></td>
<td>78</td>
<td>76</td>
<td>76</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Partial Support</th>
<th>Partial</th>
<th>Full Support</th>
<th>Full Support</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Data from cani.use.com | Linked from cani.use.mozilla.de
Basics of CSS grid

Define your grid.

Place items in the grid.
Grid terminology

- Grid line
- Grid track
- Grid area
- Grid gap
- Grid cell
Using DevTools to learn Grid

Live examples
What's in Grid Level 2?

Level 2 expands Grid by:

- adding "subgrid" capabilities for nested grids to participate in the sizing of their parent grids
- aspect-ratio–controlled gutters
Why do we need subgrid?
The subgrid syntax

Same properties, more values

```css
.grid-template-columns: subgrid <line-name-list>?
.grid-template-rows: subgrid <line-name-list>?
```

The subgrid keyword indicates to the browser the nested grid will use the same sizing as its parent along the relevant axis.

```css
.subgrid-container {
  grid-columns: 2 / 5; /* placement for the subgrid container itself */
  display: grid; /* you must still apply a display: grid to the subgrid */
  grid-template-columns: subgrid;
  grid-template-rows: subgrid;
}
```
Using DevTools to learn Subgrid

Live examples
Browser support for subgrid

<table>
<thead>
<tr>
<th>Feature of the CSS Grid Layout Module Level 2 that allows a grid item with its own grid to align in one or both dimensions with its parent grid.</th>
</tr>
</thead>
<tbody>
<tr>
<td>IE</td>
</tr>
<tr>
<td>---</td>
</tr>
<tr>
<td>9</td>
</tr>
<tr>
<td>10</td>
</tr>
<tr>
<td>11</td>
</tr>
<tr>
<td>12</td>
</tr>
</tbody>
</table>

Data from caniuse.com | Embed from caniuse.bitdots.de
CSS

"WANTS YOU"

TO HELP

MAKE THE WEB BETTER
References

- CSS Flexible Box Layout Module Level 1
- CSS Grid Layout Module Level 1
- CSS Grid Layout Module Level 2
- Codrops reference: Subgrid
- Subgrid on MDN
- Grid Level 2 and Subgrid
- Why display: contents is not CSS Grid Layout subgrid
- CSS Grid Level 2: Here Comes Subgrid
- Grid by Example
- Researching a Property in the CSS Specifications
Thank you!

https://www.chenhuijing.com

@hj_chen

@hj_chen

@huijing

Font is Red Hat Display by Jeremy Mickel.