

Kevin Geary - How to Create a "Ruled" Grid (Lines / Grid Borders) in CSS the Right Way

Ruled Grid

codepen.io/Kevin-Geary/pen/BavwqYX?editors=1100

FacebookHey!BusinessPersonalHostingResourcesStreamYardBasecampAutomatic.cssFramesHubspotHelpScoutGeniusInnerCircleACSSUserMavenJSSoftball

Ruled Grid

Kevin Geary

PRO

HTML

work regardless of the number of columns & rows!</p>

CSS (SCSS)

```
1 > p {<div>
4 > .container {</div>
10
11 // Ruled Grid
12
13 > .ruled-grid {
14   display: grid;
15   grid-template-columns: repeat(3, minmax(0, 1fr));
16   align-items: stretch;
17 }
18
19 > .ruled-grid > .card {
20 }
21
22 // Row Lines
23
24
25 // Column Lines
26
```

JS

Save

Settings

Ruled Grid?

Is it possible to create a ruled grid that's flexible, scalable, and maintainable?

No Media Queries?

If possible, can we avoid media queries? Can it work flawlessly at any breakpoint?

Any Column Count?

It would be great if the technique would work regardless of the number of columns & rows!

Any # of Grid Items?

If we have an uneven number of grid items in the grid, will it still work?

Avoid :nth() selectors?

We must avoid :nth() selectors as they'll break column count flexibility.

Works with any background?

Can't use background-color & gap technique because it's too limiting.

Also! If grid cells have an uneven amount of content, can we force all grid cells to be symmetrical?

We want lines inside the grid - nothing on the outside

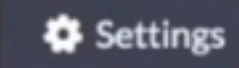
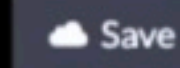
Video

ConsoleAssetsCommentsKeys

Last saved SEPTEMBER 18, 2023 - 9:38:55 AM

DeleteAdd to CollectionForkEmbedExportShare

00:38



HTML

work regardless of the number of columns & rows!</p>

CSS (SCSS)

```
1 > p {}  
4 > .container {}  
18  
11 // Ruled Grid  
  
19 > .ruled-grid {  
20   display: grid;  
21   grid-template-columns: repeat(3, minmax(0, 1fr));  
22   align-items: stretch;  
23 }  
24  
25 // Row Lines  
26  
27 // Column Lines
```

Our basic setup

Ruled Grid?

Is it possible to create a ruled grid that's flexible, scalable, and maintainable?

No Media Queries?

If possible, can we avoid media queries? Can it work flawlessly at any breakpoint?

Any Column Count?

It would be great if the technique would work regardless of the number of columns & rows!

Any # of Grid Items?

If we have an uneven number of grid items in the grid, will it still work?

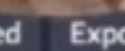
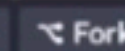
Avoid :nth() selectors?

We must avoid :nth() selectors as they'll break column count flexibility.

Works with any background?

Can't use background-color & gap technique because it's too limiting.

Also! If grid cells have an uneven amount of content, can we force all grid cells to be symmetrical?



HTML

work regardless of the number of columns & rows!</p>

CSS (SCSS)

```
1 > p {<div>
4 > .containerer {</div>
10
11 // Ruled Grid
12
13 > .ruled-grid {
14   display: grid;
15   grid-template-columns: repeat(3, minmax(0, 1fr));
16   align-items: stretch;
17 }
18
19 > .ruled-grid > .card {
20   position: relative;
21 }
22
23
24
25
26
27
28
29
30
31
32
33 // Row Lines
34
35 > .ruled-grid > .card::after {
36   content: "";
37   background-color: red;
38   position: absolute;
39   width: 100%;
40   height: 2px;
41 }
42
43
44
45
46
47
48
49
50
51
52
53
54 // Column Lines
55
```

JS

Console

Assets

Comments

⌘ Keys

Ruled Grid?

Is it possible to create a ruled grid that's flexible, scalable, and maintainable?

No Media Queries?

If possible, can we avoid media queries? Can it work flawlessly at any breakpoint?

Any Column Count?

It would be great if the technique would work regardless of the number of columns & rows!

Any # of Grid Items?

If we have an uneven number of grid items in the grid, will it still work?

Avoid :nth() selectors?

We must avoid :nth() selectors as they'll break column count flexibility.

Works with any background?

Can't use background-color & gap technique because it's too limiting.

Also! If grid cells have an uneven amount of content, can we force all grid cells to be symmetrical?

Our starter for the row lines

Because we're using absolutely positioned elements in the card

Kevin Geary

PRO

Save

Settings

Last saved LESS THAN A MINUTE AGO

Delete

Add to Collection

Fork

Embed

Export

Share

08:00

HTML

work regardless of the number of columns & rows! </p>

CSS (SCSS)

```
1 > p {<del>grid</del>}
4 > .containerer {<del>grid</del>}
10
11 // Ruled Grid
12
13 > .ruled-grid {
14   display: grid;
15   grid-template-columns: repeat(3, minmax(0, 1fr));
16   align-items: stretch;
17 }
18
19 > .ruled-grid > .card {
20   position: relative;
21 }
22
23 // Row Lines
24
25 > .ruled-grid > .card::after {
26   content: "";
27   background-color: red;
28   position: absolute;
29   width: 100%;
30   height: 2px;
31   top: 0;
32   left: 0;
33 }
34
35
36 // Column Lines
37
```

JS

Console Assets Comments ⌘ Keys

Ruled Grid?

Is it possible to create a ruled grid that's flexible, scalable, and maintainable?

No Media Queries?

If possible, can we avoid media queries? Can it work flawlessly at any breakpoint?

Any Column Count?

It would be great if the technique would work regardless of the number of columns & rows!

Any # of Grid Items?

If we have an uneven number of grid items in the grid, will it still work?

Avoid :nth() selectors?

We must avoid :nth() selectors as they'll break column count flexibility.

Works with any background?

Can't use background-color & gap technique because it's too limiting.

Also! If grid cells have an uneven amount of content, can we force all grid cells to be symmetrical?



Last saved LESS THAN A MINUTE AGO

Delete

Add to Collection

Fork

Embed

Export

Share

08:58

HTML

work regardless of the number of columns & rows! </p>

CSS (SCSS)

```
18
19 ~ .ruled-grid > .card {
20   position: relative;
21 }
22
23 // Row Lines
24
25 ~ .ruled-grid > .card::after {
26   content: "";
27   background-color: red;
28   position: absolute;
29   width: 100%;
30   height: 2px;
31   top: 0;
32   left: 0;
33 }
```

```
5 // Column Lines
6
7 ~ .ruled-grid > .card::before {
8   content: "";
9   background-color: red;
10  position: absolute;
11  width: 2px;
12  height: 100%;
13  top: 0;
14  left: 0;
15 }
```

The basics for our column lines

Ruled Grid?

Is it possible to create a ruled grid that's flexible, scalable, and maintainable?

No Media Queries?

If possible, can we avoid media queries? Can it work flawlessly at any breakpoint?

Any Column Count?

It would be great if the technique would work regardless of the number of columns & rows!

Any # of Grid Items?

If we have an uneven number of grid items in the grid, will it still work?

Avoid :nth() selectors?

We must avoid :nth() selectors as they'll break column count flexibility.

Works with any background?

Can't use background-color & gap technique because it's too limiting.

Also! If grid cells have an uneven amount of content, can we force all grid cells to be symmetrical?



CSS (SCSS)

```

9 - .ruled-grid > .card {
10   position: relative;
11 }
12
13 // Row Lines
14
15 - .ruled-grid > .card::after {
16   content: "";
17   background-color: red;
18   position: absolute;
19   width: 100%;
20   height: 2px;
21   top: 0;
22   left: 0;
23 }
24
25 // Column Lines
26
27 - .ruled-grid > .card::before {
28   content: "";
29   background-color: red;
30   position: absolute;
31   width: 2px;
32   height: 100%;
33   top: 0;
34   left: 0;
35 }

```

0 10 20 30 40 50 60 70 80 90 100 if the technique would work

Figure 1 is a line graph with the x-axis labeled 'Number of respondents who used the technique' and the y-axis labeled 'Percentage of respondents who would use the technique if it were available'. The x-axis has major ticks at 0, 25, 50, 75, and 100. The y-axis has major ticks at 0, 25, 50, 75, and 100. The graph shows a series of data points connected by lines. The data points are approximately: (0, 100), (1, 95), (2, 90), (3, 85), (4, 80), (5, 75), (6, 70), (7, 65), (8, 60), (9, 55), (10, 50), (11, 45), (12, 40), (13, 35), (14, 30), (15, 25), (16, 20), (17, 15), (18, 10), (19, 5), (20, 0). The graph shows a decreasing trend, with a sharp drop at 100 respondents.

...if the technique would work.

We don't want the left and top line to show

From the 1970s, an increasing number of world leaders in the most developed countries have been

in we must avoid :nth() selectors as they'll break column count flexibility	Can't use background-color & gap technique because it's too limiting
---	--

Can't use background-color & gap technique because it's too limiting

the grid, will it still work? break columnin count flexibility. because it's too limiting.

Also! If grid cells have an uneven amount of content, can we force all grid cells to be symmetrical?



Ruled Grid

Kevin Geary PRO

HTML

work regardless of the number of columns & rows!</p>

CSS (SCSS)

```
1 p {}
4 .container {}
10
11 // Ruled Grid
12
13 .ruled-grid {
14   display: grid;
15   grid-template-columns: repeat(3, minmax(0, 1fr));
16   align-items: stretch;
17   gap: 2em;
18 }
19
20 .ruled-grid > .card {
21   position: relative;
22 }
23
24 // Row Lines
25
26 .ruled-grid > .card::after {
27   content: "";
28   background-color: red;
29   position: absolute;
30   width: 100%;
31   height: 2px;
32   top: 0;
33   left: 0;
34 }
35
36 // Column Lines
37
38 .ruled-grid > .card::before {
39   content: "";
40   background-color: red;
41   position: absolute;
42   width: 20x;
```

JS

Console Assets Comments Keys

Ruled Grid?

Is it possible to create a ruled grid that's flexible, scalable, and maintainable?

No Media Queries?

If possible, can we avoid media queries? Can it work flawlessly at any breakpoint?

Any Column Count?

It would be great if the technique would work regardless of the number of columns & rows!

Any # of Grid Items?

If we have an uneven number of grid items in the grid, will it still work?

Avoid :nth() selectors?


We must avoid :nth() selectors as they'll break column count flexibility.

Works with any background?

Can't use background-color & gap technique because it's too limiting.

Also! If grid cells have an uneven amount of content, can we force all grid cells to be symmetrical?

We need a gap to center the lines in the middle of the gaps



Last saved LESS THAN A MINUTE AGO Delete Add to Collection Fork Embed Export Share

11:21

Ruled Grid

Kevin Geary PRO

HTML

work regardless of the number of columns & rows! </p>

CSS (SCSS)

```
19
20 ~ .ruled-grid > .card {
21   position: relative;
22 }
23
24 // Row Lines
25
26 ~ .ruled-grid > .card::after {
27   content: "";
28   background-color: red;
29   position: absolute;
30   width: 100%;
31   height: 2px;
32   top: 0;
33   left: 0;
34 }
35
36 // Column Lines
37
38 ~ .ruled-grid > .card::before {
39   content: "";
40   background-color: red;
41   position: absolute;
42   width: 2px;
43   height: 100%;
44   top: 0;
45   left: -1em;
46 }
```

Ruled Grid?

Is it possible to create a ruled grid that's flexible, scalable, and maintainable?

No Media Queries?

If possible, can we avoid media queries? Can it work flawlessly at any breakpoint?

Any Column Count?

It would be great if the technique would work regardless of the number of columns & rows!

Any # of Grid Items?

If we have an uneven number of grid items in the grid, will it still work?

Avoid :nth() selectors?

We must avoid :nth() selectors as they'll break column count flexibility.

Works with any background?

Can't use background-color & gap technique because it's too limiting.

Also! If grid cells have an uneven amount of content, can we force all grid cells to be symmetrical?

1em is the half of our line thickness and now the line is in the middle of the gap

Kevin Geary

12:25

HTML

work regardless of the number of columns & rows! </p>

CSS (SCSS)

```
19
20 ~ .ruled-grid > .card {
21   position: relative;
22 }
23
24 // Row Lines
25
26 ~ .ruled-grid > .card::after {
27   content: "";
28   background-color: red;
29   position: absolute;
30   width: 100%;
31   height: 2px;
32   top: -1em;
33   left: 0;
34 }
35
36 // Column Lines
37
38 ~ .ruled-grid > .card::before {
39   content: "";
40   background-color: red;
41   position: absolute;
42   width: 2px;
43   height: 100%;
44   top: 0;
45   left: -1em;
46 }
```

Ruled Grid?

Is it possible to create a ruled grid that's flexible, scalable, and maintainable?

No Media Queries?

If possible, can we avoid media queries? Can it work flawlessly at any breakpoint?

Any Column Count?

It would be great if the technique would work regardless of the number of columns & rows!

Any # of Grid Items?

If we have an uneven number of grid items in the grid, will it still work?

Avoid :nth() selectors?

We must avoid :nth() selectors as they'll break column count flexibility.

Works with any background?

Can't use background-color & gap technique because it's too limiting.

Also! If grid cells have an uneven amount of content, can we force all grid cells to be symmetrical?



HTML

work regardless of the number of columns & rows!</p>

CSS (SCSS)

```
1 position: relative;
2 }
3
4 // Pseudo Elements
5
6 .ruled-grid > .card::before,
7 .ruled-grid > .card::after {
8   content: "";
9   background-color: red;
10  position: absolute;
11 }
12
13 // Row Lines
14
15 .ruled-grid > .card::after {
16   width: 100%;
17   height: 2px;
18   top: -1em;
19   left: 0;
20 }
21
22 // Column Lines
23
24 .ruled-grid > .card::before {
25   width: 2px;
26   height: 100%;
27   top: 0;
28   left: -1em;
29 }
30 }
```

JS

Ruled Grid?

Is it possible to create a ruled grid that's flexible, scalable, and maintainable?

No Media Queries?

If possible, can we avoid media queries? Can it work flawlessly at any breakpoint?

Any Column Count?

It would be great if the technique would work regardless of the number of columns & rows!

Works with any background?

Can't use background-color & gap technique because it's too limiting.

Also! If grid cells have an uneven amount of content, can we force grid cells to be symmetrical?

These are the common styles for both pseudo elements and only the relevant information is left in the details

Kevin Geary

PRO

Save

Settings

Console

Assets

Comments

Keys

Last saved LESS THAN A MINUTE AGO

Delete

Add to Collection

Fork

Embed

Export

Share

14:18

Ruled Grid

Kevin Geary PRO

HTML

work regardless of the number of columns & rows!</p>

CSS (SCSS)

```
1 p {}
4 .container {}
10
11 // Ruled Grid
12
13 .ruled-grid {
14
15   --gap: 3em;
16   --line-thickness: 2px;
17   --line-color: red;
18   --line-offset: calc(var(--gap) / 2);
19
20   display: grid;
21   grid-template-columns: repeat(3, minmax(0, 1fr));
22   align-items: stretch;
23   gap: var(--gap);
24 }
25
26 .ruled-grid > .card {
27   position: relative;
28 }
29
30 // Pseudo Elements
31
32 .ruled-grid > .card::before,
33 .ruled-grid > .card::after {
34   content: "";
35   background-color: var(--line-color);
36   position: absolute;
37 }
38
39 // Row Lines
40
41 .ruled-grid > .card::after {
42   width: 100%;
43 }
44
45 JS
```

With these tokens we can manipulate the lines from one place

Ruled Grid?

Is it possible to create a ruled grid that's flexible, scalable, and maintainable?

No Media Queries?

If possible, can we avoid media queries? Can it work flawlessly at any breakpoint?

Any Column Count?

It would be great if the technique would work regardless of the number of columns & rows!

of Grid Items?

What if we have an uneven number of grid items in the grid, will it still work?

Avoid :nth() selectors?

We must avoid :nth() selectors as they'll break column count flexibility.

Works with any background?

Can't use background-color & gap technique because it's too limiting.

Also! If grid cells have an uneven amount of content, can we force all grid cells to be symmetrical?

Last saved LESS THAN A MINUTE AGO

Delete

Add to Collection

Fork

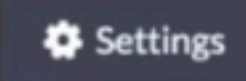
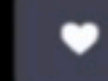
Embed

Export

Share

17:52





HTML

work regardless of the number of columns & rows! </p>

CSS (SCSS)

```
40
41 .ruled-grid > .card::after {
42   width: 100%;
43   height: var(--line-thickness);
44   top: var(--line-offset);
45   left: 0;
46 }
47
48 // Column Lines
49
50 .ruled-grid > .card::before {
51   width: var(--line-thickness);
52   height: 100%;
53   top: 0;
54   left: var(--line-offset);
55 }
```

Ruled Grid?

Is it possible to create a ruled grid that's flexible, scalable, and maintainable?

No Media Queries?

If possible, can we avoid media queries? Can it work flawlessly at any breakpoint?

Any Column Count?

It would be great if the technique would work regardless of the number of columns & rows!

Any # of Grid Items?

If we have an uneven number of grid items in the grid, will it still work?

Avoid :nth() selectors?

We must avoid :nth() selectors as they'll break column count flexibility.

Works with any background?

Can't use background-color & gap technique because it's too limiting.

Also! If grid cells have an uneven amount of content, can we force all grid cells to be symmetrical?

Because these are now positive values they're bringing the lines in



Ruled Grid

Kevin Geary PRO

HTML

work regardless of the number of columns & rows!</p>

CSS (SCSS)

```
10
11 // Ruled Grid
12
13 .ruled-grid {
14
15   --gap: 5em;
16   --line-thickness: 2px;
17   --line-color: red;
18   --line-offset: calc(var(--gap) / 2);
19
20   display: grid;
21   grid-template-columns: repeat(3, minmax(0, 1fr));
22   align-items: stretch;
23   gap: var(--gap);
24 }
25
26 .ruled-grid > .card {
27   position: relative;
28 }
29
30 // Pseudo Elements
31
32 .ruled-grid > .card::before,
33 .ruled-grid > .card::after {
34   content: "";
35   background-color: var(--line-color);
36   position: absolute;
37 }
38
39 // Row Lines
40
41 .ruled-grid > .card::after {
42   width: 100%;
43   height: var(--line-thickness);
44   top: var(--line-offset);
```

Ruled Grid?

Is it possible to create a ruled grid that's flexible, scalable, and maintainable?

No Media Queries?

If possible, can we avoid media queries? Can it work flawlessly at any breakpoint?

Any Column Count?

It would be great if the technique would work regardless of the number of columns & rows!

Any # of Grid Items?

If we have an uneven number of grid items in the grid, will it still work?

Avoid :nth() selectors?

We must avoid :nth() selectors as they'll break column count flexibility.

Works with any background?

Can't use background-color & gap technique because it's too limiting.

Also! If grid cells have an uneven amount of content, can we force all grid cells to be symmetrical?

Best practice is not to assign a negative value to a token but making it negative in the place where it needs to be!

Kevin Geary

HTML

work regardless of the number of columns & rows! </p>

CSS (SCSS)

```
39 // Row Lines
40
41 .ruled-grid > .card::after {
42   width: 100%;
43   height: var(--line-thickness);
44   top: calc(var(--line-offset) * -1);
45   left: 0;
46 }
47
48 // Column Lines
49
50 .ruled-grid > .card::before {
51   width: var(--line-thickness);
52   height: 100%;
53   top: 0;
54   left: calc(var(--line-offset) * -1);
55 }
```

Ruled Grid?

Is it possible to create a ruled grid that's flexible, scalable, and maintainable?

No Media Queries?

If possible, can we avoid media queries? Can it work flawlessly at any breakpoint?

Any Column Count?

It would be great if the technique would work regardless of the number of columns & rows!

Any # of Grid Items?

If we have an uneven number of grid items in the grid, will it still work?

Avoid :nth() selectors?

We must avoid :nth() selectors as they'll break column count flexibility.

Works with any background?

Can't use background-color & gap technique because it's too limiting.

Also! If grid cells have an uneven amount of content, can we force all grid cells to be symmetrical?



Ruled Grid

Kevin Geary PRO

Pen saved. ✕

Save Settings

HTML

work regardless of the number of columns & rows! </p>

CSS (SCSS)

```
1 > p {<
4 > .container {<
10
11 // Ruled Grid
12
13 > .ruled-grid {
14
15   --gap: 2em;
16   --line-thickness: 2px;
17   --line-color: red;
18   --line-offset: calc(var(--gap) / 2);
19
20   display: grid;
21   grid-template-columns: repeat(3, minmax(0, 1fr));
22   align-items: stretch;
23   gap: var(--gap);
24 }
25
26 > .ruled-grid > .card {
27   position: relative;
28 }
29
30 // Pseudo Elements
31
32 > .ruled-grid > .card::before,
33 > .ruled-grid > .card::after {
34   content: "";
35   background-color: var(--line-color);
36   position: absolute;
37 }
38
39 // Row Lines
40
41 > .ruled-grid > .card::after {
42   width: 100%;
43 }
44 JS
```


work regardless of the number of columns & rows!</p>

CSS (SCSS)

```
3 ~ .ruled-grid > .card::after {
4   content: "";
5   background-color: var(--line-color);
6   position: absolute;
7 }
```

```
1 .ruled-grid > .card::after {
2   width: 100%;
3   height: var(--line-thickness);
4   top: calc(var(--line-offset) * -1);
5   left: 0;
6 }
```

100% card a

```

9
10 .ruled-grid > .card::before {
11   width: var(--line-thickness);
12   height: 100%;
13   top: 0;
14   left: calc(var(--line-offset) * -1);
15 }

```

JS ⚙️ ⌵

Console Assets Comments **⌘ Keys**

◀ ▶ ⏪ ⏩ ☐ ≡

100% in this case is the size of the card and the gaps will never be filled

Ruled Grid?

Is it possible to create a ruled grid that's flexible, scalable, and maintainable?

No Media Queries?

If possible, can we avoid media queries?
Can it work flawlessly at any breakpoint?

Any Column Count?

It would be great if the technique would work regardless of the number of columns & rows!

Any # of Grid Items?

If we have an uneven number of grid items in the grid, will it still work?

Avoid :nth() selectors?

We must avoid :nth() selectors as they'll break column count flexibility.

Works with any background?

Can't use background-color & gap technique because it's too limiting.

cells have an uneven amount of content, can we force all grid cells to be symmetrical?



HTML

work regardless of the number of columns & rows! </p>

CSS (SCSS)

```
33 ~ .ruled-grid > .card::after {
34   content: "";
35   background-color: var(--line-color);
36   position: absolute;
37 }
38
39 // Row Lines
40
41 ~ .ruled-grid > .card::after {
42   width: 100vw;
43   height: var(--line-thickness);
44   top: calc(var(--line-offset) * -1);
45   left: 0;
46 }
47
48 // Column Lines
49
50 ~ .ruled-grid > .card::before {
51   width: var(--line-thickness);
52   height: 100%;
53   top: 0;
54   left: calc(var(--line-offset) * -1);
55 }
```

Ruled Grid?

Is it possible to create a ruled grid that's flexible, scalable, and maintainable?

No Media Queries?

If possible, can we avoid media queries? Can it work flawlessly at any breakpoint?

Any Column Count?

It would be great if the technique would work regardless of the number of columns & rows!

Any # of Grid Items?

If we have an uneven number of grid items in the grid, will it still work?

Avoid :nth() selectors?

We must avoid :nth() selectors as they'll break column count flexibility.

Works with any background?

Can't use background-color & gap technique because it's too limiting.

The solution to this is 'vw'

grid cells have an uneven amount of content, can we force all grid cells to be symmetrical?



HTML

work regardless of the number of columns & rows!</p>

CSS (SCSS)

```
3 .ruled-grid > .card::after {
4   content: "";
5   background-color: var(--line-color);
6   position: absolute;
7 }
```

```

7 // Row Lines
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
```

```

9
10 .ruled-grid > .card::before {
11   width: var(--line-thickness);
12   height: 100vh;
13   top: 0;
14   left: calc(var(--line-offset) * -1);
15 }

```

Console Assets Comments **⌘ Keys**

[Console](#)
[Assets](#)
[Comments](#)
[🗖 Keys](#)
Last saved LESS THAN A MINUTE AGO
[🔗](#)
[Delete](#)
[Add to Collection](#)
[🍴 Fork](#)
[Embed](#)
[Export](#)
[Share](#)

[Console](#)
[Assets](#)
[Comments](#)
[🗖 Keys](#)
Last saved LESS THAN A MINUTE AGO
[🔗](#)
[Delete](#)
[Add to Collection](#)
[🍴 Fork](#)
[Embed](#)
[Export](#)
[Share](#)

Navigation icons: back, forward, search, etc. | Time: 20:55

Navigation icons: back, forward, search, etc. | Time: 20:55

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191	192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223	224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239	240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255	256	257	258	259	260	261	262	263	264	265	266	267	268	269	270	271	272	273	274	275	276	277	278	279	280	281	282	283	284	285	286	287	288	289	290	291	292	293	294	295	296	297	298	299	300	301	302	303	304	305	306	307	308	309	310	311	312	313	314	315	316	317	318	319	320	321	322	323	324	325	326	327	328	329	330	331	332	333	334	335	336	337	338	339	340	341	342	343	344	345	346	347	348	349	350	351	352	353	354	355	356	357	358	359	360	361	362	363	364	365	366	367	368	369	370	371	372	373	374	375	376	377	378	379	380	381	382	383	384	385	386	387	388	389	390	391	392	393	394	395	396	397	398	399	400	401	402	403	404	405	406	407	408	409	410	411	412	413	414	415	416	417	418	419	420	421	422	423	424	425	426	427	428	429	430	431	432	433	434	435	436	437	438	439	440	441	442	443	444	445	446	447	448	449	450	451	452	453	454	455	456	457	458	459	460	461	462	463	464	465	466	467	468	469	470	471	472	473	474	475	476	477	478	479	480	481	482	483	484	485	486	487	488	489	490	491	492	493	494	495	496	497	498	499	500	501	502	503	504	505	506	507	508	509	510	511	512	513	514	515	516	517	518	519	520	521	522	523	52
--	---	---	---	---	---	---	---	---	---	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	----

Ruled Grid?	No Media Queries?	Any Column Count?
Is it possible to create a ruled grid that's flexible, scalable, and maintainable?	If possible, can we avoid media queries? Can it work flawlessly at any breakpoint?	It would be great if the technique would work regardless of the number of columns & rows!

[illegible]

No Media Queries? If possible, can we avoid media queries? Can it work flawlessly at any breakpoint?	Any Column Count? It would be great if the technique would work regardless of the number of columns & rows!
---	---

Any Column Count?

It would be great if the technique would work regardless of the number of columns & rows!

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100
---	---	---	---	---	---	---	---	---	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	-----

Any # of Grid Items?	Avoid :nth() selectors?	Works with any background?
If we have an uneven number of grid items in the grid, will it still work?	We must avoid :nth() selectors as they'll break column count flexibility.	Can't use background-color & gap technique because it's too limiting.

[illegible]

Avoid :nth() selectors? We must avoid :nth() selectors as they'll break column count flexibility.	Works with any background? Can't use background-color & gap technique because it's too limiting.
---	--

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100
000	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100

Works with any background?

Can't use background-color & gap technique because it's too limiting.

Also! If grid cells have an uneven amount of content, can we force all grid cells to be symmetrical?

Look at the middle of the grid: That's exactly what we want



HTML

work regardless of the number of columns & rows! </p>

CSS (SCSS)

```
10
11 // Ruled Grid
12
13 .ruled-grid {
14
15   --gap: 2em;
16   --line-thickness: 2px;
17   --line-color: red;
18   --line-offset: calc(var(--gap) / 2);
19
20   display: grid;
21   grid-template-columns: repeat(3, minmax(0, 1fr));
22   align-items: stretch;
23   gap: var(--gap);
24   overflow: hidden;
25 }
26
27 .ruled-grid > .card {
28   position: relative;
29 }
30
31 // Pseudo Elements
32
33 .ruled-grid > .card::before,
34 .ruled-grid > .card::after {
35   content: "";
36   background-color: var(--line-color);
37   position: absolute;
38 }
39
40 // Row Lines
41
42 .ruled-grid > .card::after {
43   width: 100vw;
44   height: var(--line-thickness);
```

JS

Pen saved. ✕

Settings

Save

Update

HTML

work regardless of the number of columns & rows! <p>

CSS (SCSS) ⚙️ ▼

```
// Ruled Grid
.uled-grid {
  --gap: 2em;
  --line-thickness: 2px;
  --line-color: red;
  --line-offset: calc(var(--gap) / 2);

  display: grid;
  grid-template-columns: repeat(3, minmax(0, 1fr));
  align-items: stretch;
  gap: var(--gap);
  overflow: hidden;
}

.uled-grid > .card {
  position: relative;
  padding: 1.5em;
}

// Pseudo Elements
.uled-grid > .card::before,
.uled-grid > .card::after {
  content: "";
  background-color: var(--line-color);
  position: absolute;
}

// Row Lines
.uled-grid > .card::after {
  width: 100vw;
```

Ruled Grid?	No Media Queries?	Any Column Count?
Is it possible to create a ruled grid that's flexible, scalable, and maintainable?	If possible, can we avoid media queries? Can it work flawlessly at any breakpoint?	It would be great if the technique would work regardless of the number of columns & rows!
Any # of Grid Items?	Avoid :nth() selectors?	Works with any background?
If we have an uneven number of grid items in the grid, will it still work?	We must avoid :nth() selectors as they'll break column count flexibility.	Can't use background-color & gap technique because it's too limiting.

Also! If grid cells have an uneven amount of content, can we force all grid cells to be symmetrical?

Additional padding doesn't break the layout

Ruled Grid?	No Media Queries?	Any Column Count?	
--------------------	--------------------------	--------------------------	--

Is it possible to create a ruled grid that's flexible, scalable, and maintainable?	If possible, can we avoid media queries? Can it work flawlessly at any breakpoint?	It would be great if the technique would work regardless of the number of columns & rows!
--	--	---

No Media Queries?	Any Column Count?	

If possible, can we avoid media queries? Can it work flawlessly at any breakpoint?	It would be great if the technique would work regardless of the number of columns & rows!
--	---

Any Column Count?

It would be great if the technique would work regardless of the number of columns & rows!

Any # of Grid Items?	Avoid :nth() selectors?	Works with any
----------------------	-------------------------	----------------

If we have an uneven number of grid items in the grid, will it still work?	We must avoid :nth() selectors as they'll break column count flexibility.	Can't use background-color & gap technique because it's too limiting.
--	---	---

Avoid :nth() selectors?	Works with any
-------------------------	----------------

We must avoid :nth() selectors as they'll break column count flexibility.

Works with any background?

Can't use background-color & gap technique because it's too limiting.

Also! If grid cells have an uneven amount of content, can we force all grid cells to be symmetrical?

Additional padding doesn't break the layout



Ruled Grid

Kevin Geary PRO

HTML

work regardless of the number of columns & rows! </p>

CSS (SCSS)

```
10
11 // Ruled Grid
12
13 .ruled-grid {
14   --gap: 2em;
15   --line-thickness: 2px;
16   --line-color: red;
17   --line-offset: calc(var(--gap) / 2);
18
19   display: grid;
20   grid-template-columns: repeat(auto-fit, minmax(0, 1fr));
21   align-items: stretch;
22   gap: var(--gap);
23   overflow: hidden;
24 }
25
26 .ruled-grid > .card {
27   position: relative;
28   padding: 1.5em;
29 }
30
31 // Pseudo Elements
32
33 .ruled-grid > .card::before,
34 .ruled-grid > .card::after {
35   content: "";
36   background-color: var(--line-color);
37   position: absolute;
38 }
39
40 // Row Lines
41
42 .ruled-grid > .card::after {
```

Ruled Grid?

Is it possible to create a ruled grid that's flexible, scalable, and maintainable?

No Media Queries?

If possible, can we avoid media queries? Can it work flawlessly at any breakpoint?

Any Column Count?

It would be great if the technique would work regardless of the number of columns & rows!

Any # of Grid Items?

If we have an uneven number of grid items in the grid, will it still work?

Avoid :nth() selectors?


We must avoid :nth() selectors as they'll break column count flexibility.

Works with any background

Can't use background-color & gap technique because it's too limiting.

Also! If grid cells have an uneven amount of content, can we force all grid cells to be symmetrical?

Doesn't break



Console

Assets

Comments

Keys

Last saved LESS THAN A MINUTE AGO

Delete

Add to Collection

Fork

Embed

Export

Share

22:41

Is it possible to create a ruled grid that's flexible, scalable, and maintainable?

If possible, can we avoid media queries?
Can it work flawlessly at any breakpoint?

It would be great if the technique would work regardless of the number of columns & rows!

If we have an uneven number of grid items in the grid, will it still work?

We must avoid :nth() selectors as they'll break column count flexibility.

Can't use background-color & gap technique because it's too limiting.

Also! If grid cells have an uneven amount of content, can we force all grid cells to be symmetrical?

CSS (SCSS)

10

```
11 // Ruled Grid
12
13 .ruled-grid {
14
15     --gap: 2em;
16     --line-thickness: 2px;
17     --line-color: red;
18     --line-offset: calc(var(--gap) / 2);
```

```

19
20     display: grid;
21     grid-template-columns: repeat(auto-fit,
minmax(200px, 1fr));
22     align-items: stretch;
23     gap: var(--gap);
24     overflow: hidden;
25 }

```

```
26  
27 ~ .ruled-grid > .card {  
28   position: relative;  
29   padding: 1.5em;  
30 }
```

33

```
34 .ruled-grid > .card::before,
35 .ruled-grid > .card::after {
36   content: "";
37   background-color: var(--line-color);
38   position: absolute;
39 }
```

42

```
43+.ruled-grid > .card::after {
```

JS ⚙️ ▼

Console Assets Comments % Keys

Add to Collection

Embed

Share

1

HTML

work regardless of the number of columns & rows! </p>

CSS (SCSS)

```
10
11 // Ruled Grid
12
13 .ruled-grid {
14
15   --gap: 2em;
16   --line-thickness: 2px;
17   --line-color: red;
18   --line-offset: calc(var(--gap) / 2);
19
20   display: grid;
21   grid-template-columns: repeat(auto-fit,
22     minmax(320px, 1fr));
23   align-items: stretch;
24   gap: var(--gap);
25   overflow: hidden;
26
27   .ruled-grid > .card {
28     position: relative;
29     padding: 1.5em;
30   }
31
32   // Pseudo Elements
33
34   .ruled-grid > .card::before,
35   .ruled-grid > .card::after {
36     content: "";
37     background-color: var(--line-color);
38     position: absolute;
39   }
40
41   // Row Lines
42
43   .ruled-grid > .card::after {
```

Ruled Grid?

Is it possible to create a ruled grid that's flexible, scalable, and maintainable?

No Media Queries?

If possible, can we avoid media queries? Can it work flawlessly at any breakpoint?

Any Column Count?

It would be great if the technique would work regardless of the number of columns & rows!

Any # of Grid Items?

If we have an uneven number of grid items in the grid, will it still work?

Avoid :nth() selectors?

We must avoid :nth() selectors as they'll break column count flexibility.

Works with any background?

Can't use background-color & gap technique because it's too limiting.

Also! If grid cells have an uneven amount of content, can we force all grid cells to be symmetrical?



Dimensions: Responsive ▾

1670

x

665

100% ▾

Custom ▾

Ruled Grid
Kevin Geary **PRO**

HTML

work regardless of the number of columns & rows!

CSS (SCSS)

```
10
11 // Ruled Grid
12
13 .ruled-grid {
14
15   --gap: 2em;
16   --line-thickness: 2px;
17   --line-color: red;
18   --line-offset: calc(var(--gap) / 2);
19
20   display: grid;
21   grid-template-columns: repeat(auto-fit,
22     minmax(280px, 1fr));
23   align-items: stretch;
24   gap: var(--gap);
25   overflow: hidden;
26 }
```

JS

Ruled Grid?

Is it possible to create a ruled grid that's flexible, scalable, and maintainable?

No Media Queries?

If possible, can we avoid media queries? Can it work flawlessly at any breakpoint?

Any Column Count?

It would be great if the technique would work regardless of the number of columns & rows!

Any # of Grid Items?

If we have an uneven number of grid items in the grid, will it still work?

Avoid :nth() selectors?

Works with any background?

Fork

Embed

Export

Share

We don't need any media queriesElements Console Sources **Network** Performance Memory Application Security Lighthouse Animations 49 1

```
loading="lazy" spellcheck="false">
  #document
    <!DOCTYPE html>
    <html lang="en" class=""> == $0
  </iframe>
  <grammarly-extension data-grammarly-shadow-root="true" style="position: absolute; top: 0px; left: 0px; po
```

html.mac.chrome116.js body.room-editor.editor.state-htmlOn-cssOn-jsOn.twilight.layout-left.layout-side.logged-in div.page-wrap div

Console What's New X Issues

Styles Computed Layout

Filter

```
element.style {
}
```

```
html {
  font-family: sans-serif;
```



Kevin Geary - How to Create a "Ruled" Grid (Lines / Grid Borders) in CSS the Right Way

Ruled Grid

codepen.io/Kevin-Geary/pen/BavwqYX?editors=1100

Facebook Hey! Business Personal Hosting Resources StreamYard Basecamp Automatic.css Frames Hubspot HelpScout Geni.us InnerCircle ACSS UserMaven JS Softball

Dimensions: Responsive 1235 x 665 100% Custom

Ruled Grid Kevin Geary PRO

HTML

work regardless of the number of columns & rows!</div>

CSS (SCSS)

```
10
11 // Ruled Grid
12
13 .ruled-grid {
14
15   --gap: 2em;
16   --line-thickness: 2px;
17   --line-color: red;
18   --line-offset: calc(var(--gap) / 2);
19
20   display: grid;
21   grid-template-columns: repeat(auto-fit,
22     minmax(280px, 1fr));
23   align-items: stretch;
24   gap: var(--gap);
25   overflow: hidden;
26 }
27
```

JS

Console Assets Comments Keys

Last saved LESS THAN A MINUTE AGO Delete Add to Collection Fork Embed Export Share

Avoid :nth() selectors?

We must avoid :nth() selectors as they'll break column count flexibility.

Works with any background?

Can't use background-color & gap technique because it's too limiting.

Also! If grid cells have an uneven amount of content, can we force all grid cells to be symmetrical?

Elements Console Sources Network Performance Memory Application Security Lighthouse Animations

loading="lazy" spellcheck="false">

#document

<!DOCTYPE html>

<html lang="en" class>...</html> == \$0

</iframe>

<grammarly-extension data-grammarly-shadow-root="true" style="position: absolute; top: 0px; left: 0px; po

html.mac.chrome116.js body.room-editor.editor.state-htmlOn-cssOn-jsOn.twilight.layout-left.layout-side.logged-in div.page-wrap div

Console What's New Issues

Styles Computed Layout

Filter

element.style {

}

html {

font-family: sans-serif;

23:39

HTML

- ✱ CSS (SCSS)

```
1 > p {oo}
4 > .container {oo}
```

```
body {
  background: linear-gradient(#ddd, #aaa);
}
```

- absolute
- activeborder
- additive
- activecaption
- afar
- after-white-space
- ahead
- alias
- all
- all-scroll
- alphabetic
- alternate

```
// Ruled Grid
6
7 .ruled-grid {
8
9     --gap: 2em;
10    --line-thickness: 2px;
11    --line-color: red;
12    --line-offset: calc(var(--gap) / 2);
13
14    display: grid;
15    grid-template-columns: repeat(auto-fit,
16    minmax(280px, 1fr));
17    align-items: stretch;
18    gap: var(--gap);
19    overflow: hidden;
20 }
21
22 .ruled-grid > .card {
23     position: relative;
24     padding: 1.5em;
25 }
26
27 // Pseudo Elements
28
29 .ruled-grid > .card::before,
30 .ruled-grid > .card::after {
31     content: "";
32     background-color: var(--line-color);
33     position: absolute;
```


Console Assets Comments Keys

Last saved LESS THAN A MINUTE AGO

[Delete](#)
[Add to Collection](#)
[Fork](#)
[Embed](#)
[Export](#)
[Share](#)

24:44

Ruled Grid?

Is it possible to create a ruled grid that's flexible, scalable, and maintainable?

No Media Queries?

If possible, can we avoid media queries? Can it work flawlessly at any breakpoint?

Any Column Count?

It would be great if the technique would work regardless of the number of columns & rows!

Any # of Grid Items?

If we have an uneven number of grid items in the grid, will it still work?

Avoid :nth() selectors?

We must avoid :nth() selectors as they'll break column count flexibility.

Works with any background?

Can't use background-color & gap technique because it's too limiting.

Also! If grid cells have an uneven amount of content, can we force all grid cells to be symmetrical?

Works with background gradient or color as well



HTML

```
2- <div class="ruled-grid">
3-   <div class="card">
4-     <h3>Ruled Grid?</h3>
5-     <p>Is it possible to create a ruled grid that's
flexible, scalable, and maintainable? Is it possible
to create a ruled grid that's flexible, scalable, and
maintainable?</p>
6-   </div>
7-   <div class="card">
8-     <h3>No Media Queries?</h3>
9-     <p>If possible, can we avoid media queries? Can
it work flawlessly at any breakpoint?</p>
10-  </div>
11-  <div class="card">
12-    <h3>Any Column Count?</h3>
13-    <p>It would be great if the technique would
work regardless of the number of columns & rows!</p>
14-  </div>
15-  <div class="card">
16-    <h3>Any # of Grid Items?</h3>
17-    <p>If we have an uneven number of grid items in
the grid, will it still work?</p>
18-  </div>
19-  <div class="card">
20-    <h3>Avoid :nth() selectors?</h3>
21-    <p>We must avoid :nth() selectors as they'll
break column count flexibility.</p>
```

CSS (SCSS)

```
1- p {
2- }
3- .container {
4- }
5-
6-
7-
8-
9-
10- // Ruled Grid
11-
12-
13- .ruled-grid {
14-
15-   --gap: 2em;
16-   --line-thickness: 2px;
```

JS

Console Assets Comments ⌨ Keys

Last saved LESS THAN A MINUTE AGO

Delete

Add to Collection

Fork

Embed

Export

Share

Ruled Grid?

Is it possible to create a ruled grid that's flexible, scalable, and maintainable? Is it possible to create a ruled grid that's flexible, scalable, and maintainable?

No Media Queries?

If possible, can we avoid media queries? Can it work flawlessly at any breakpoint?

Any Column Count?

It would be great if the technique would work regardless of the number of columns & rows!

Any # of Grid Items?

If we have an uneven number of grid items in the grid, will it still work?

Avoid :nth() selectors?

We must avoid :nth() selectors as they'll break column count flexibility.

Works with any background?

Can't use background-color & gap technique because it's too limiting.

Also! If grid cells have an uneven amount of content, can we force all grid cells to be symmetrical?

Problem: If 1 card has more content the grid is off balance



HTML

CSS (SCSS)

```
1 // Ruled Grid
2
3 .ruled-grid {
4
5     --gap: 2em;
6     --line-thickness: 2px;
7     --line-color: red;
8     --line-offset: calc(var(--gap) / 2);
9
10    display: grid;
11    grid-template-columns: repeat(auto-fit,
12    minmax(280px, 1fr));
13    grid-auto-rows: minmax(min-content, 1fr);
14    align-items: stretch;
15    gap: var(--gap);
16    overflow: hidden;
17 }
18
19 .ruled-grid > .card {
20     position: relative;
21     padding: 1.5em;
22 }
23
24 // Pseudo Elements
25
26 .ruled-grid > .card::before,
27 .ruled-grid > .card::after {
28     content: "";
29     background-color: var(--line-color);
30     position: absolute;
31 }
32
33 // Row Lines
34
35 .ruled-grid > .card::after {
```

Ruled Grid?

Is it possible to create a ruled grid that's flexible, scalable, and maintainable? Is it possible to create a ruled grid that's flexible, scalable, and maintainable?

No Media Queries?

If possible, can we avoid media queries? Can it work flawlessly at any breakpoint?

Any Column Count?

It would be great if the technique would work regardless of the number of columns & rows!

Any # of Grid Items?

If we have an uneven number of grid items in the grid, will it still work?

Avoid :nth() selectors?

We must avoid `:nth()` selectors as they'll break column count flexibility.

Works with any background?

Can't use background-color & gap technique because it's too limiting.

Also! If grid cells have an uneven amount of content, can we force all grid cells to be symmetrical?

The solution

It takes the largest content and applies the size to every item in the grid




```
38 background-color: var(--line-color);
39 position: absolute;
40 }

// Row Lines
.ruled-grid > .card::after {
  inline-size: 100vw;
  block-size: var(--line-thickness);
  inset-block-start: calc(var(--line-offset) * -1);
  inset-inline-start: 0;
}

// Column Lines
.ruled-grid > .card::before {
  inline-size: var(--line-thickness);
  block-size: 100vh;
  inset-block-start: 0;
  inset-inline-start: calc(var(--line-offset) * -1);
} inset-inline-start
```

Ruled Grid?

Is it possible to create a ruled grid that's flexible, scalable, and maintainable? Is it possible to create a ruled grid that's flexible, scalable, and maintainable?

No Media Queries?

If possible, can we avoid media queries? Can it work flawlessly at any breakpoint?

Any Column Count?

It would be great if the technique would work regardless of the number of columns & rows!

Any # of Grid Items?

If we have an uneven number of grid items in the grid, will it still work?

Avoid :nth() selectors?

We must avoid :nth() selectors as they'll break column count flexibility.

Works with any background?

Can't use background-color & gap technique because it's too limiting.

We converted everything to logical properties

Also! If grid cells have an uneven amount of content, can we force all grid cells to be symmetrical?




```
34
35 .ruled-grid > .card::before,
36 .ruled-grid > .card::after {
37   content: "";
38   background-color: var(--line-color);
39   position: absolute;
40 }
41
42 // Row Lines
43
44 .ruled-grid > .card::after {
45   inline-size: 100vw;
46   block-size: var(--line-thickness);
47   inset-block-start: calc(var(--line-offset) * -1);
48   inset-inline-start: 0;
49 }
50
51 // Column Lines
52
53 .ruled-grid > .card::before {
54   inline-size: var(--line-thickness);
55   block-size: 100vh;
56   inset-block-start: 0;
57   inset-inline-start: calc(var(--line-offset) * -1);
58 }
```

Ruled Grid?

Is it possible to create a ruled grid that's flexible, scalable, and maintainable? Is it possible to create a ruled grid that's flexible, scalable, and maintainable?

No Media Queries?

If possible, can we avoid media queries? Can it work flawlessly at any breakpoint?

Any Column Count?

It would be great if the technique would work regardless of the number of columns & rows!

Any # of Grid Items?

If we have an uneven number of grid items in the grid, will it still work?

Avoid :nth() selectors?

We must avoid :nth() selectors as they'll break column count flexibility.

Works with any background?

Can't use background-color & gap technique because it's too limiting.

Also! If grid cells have an uneven amount of content, can we force

This solution works in any professional page builder (so not in Elementor for example because of the bloat and number of divs you can't control).

It's a perfect solution for Bricks Builder

