

The repeat() Function

Repeat Yourself — But In CSS

Long templates are a smell.

```
grid-template-columns: 1fr 1fr 1fr 1fr 1fr 1fr 1fr 1fr 1fr;
```

“Writing this is fine... once. Writing it all semester is not.”

Same Result. Less Noise.

BEFORE

```
grid-template-columns: 1fr 1fr 1fr 1fr;
```

AFTER

```
grid-template-columns: repeat(4, 1fr);
```


Anatomy of a Pattern

repeat(**4**, **1fr**)



****The Count****

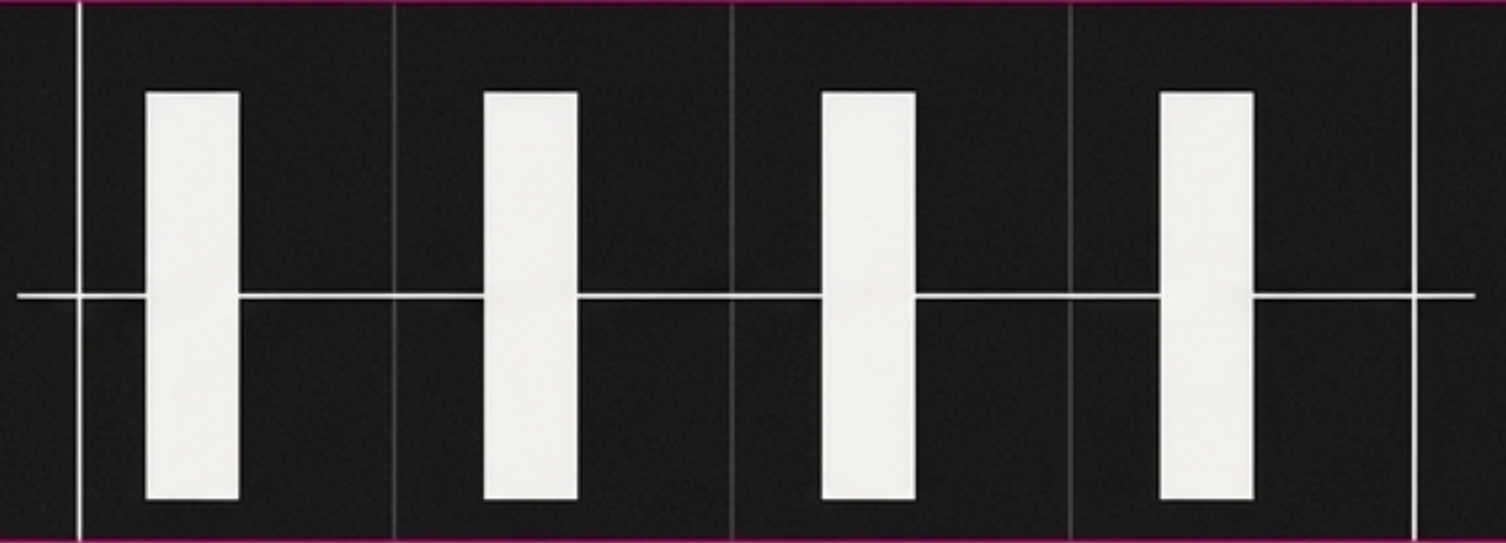
Make 4 columns

****The Pattern****

Each column is 1fr

Change the Count, Keep the Pattern

```
grid-template-columns: repeat(4, 1fr);
```



```
grid-template-columns: repeat(6, 1fr);
```



You didn't rewrite the track list — you changed the count. That's the whole point.

Repeat Works for Rows, Too.

“Any track size works: `fr`, `rem`, `%`, `minmax()`, etc.”

```
grid-template-rows: repeat(3, 10rem);
```



In Practice: The Pattern Forge

Change the count. Keep the pattern. That's the whole point of repeat().

Expanded templates aren't 'wrong' — they're just noisy. repeat() keeps your intent readable.

Pattern Forge

grid-template-columns: repeat(3, 1fr)

Build a template

Change the count. Keep the pattern. That's the whole point of repeat().

track: 1fr

track: 12rem

track: minmax(10rem, 1fr)

repeat count: 3

use repeat()

expand tracks

Expanded templates aren't "wrong" — they're just noisy. repeat() keeps your intent readable.

Stage

grid-template-columns: repeat(3, 1fr)

1

Auto-placed

2

Auto-placed

3

Auto-placed

4

Auto-placed

5

Auto-placed

6

Auto-placed

7

Auto-placed

8

Auto-placed

9

Auto-placed

10

Auto-placed

11

Auto-placed

12

Auto-placed

Same items. Same flow. Only the template changes.

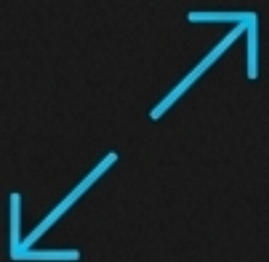
Same items. Same flow.
Only the template changes.

Your New Default for Scalable Grids

Use ``repeat()`` when:



Your grid has repeated structure.



You want to scale column or row counts quickly.



You want templates that stay readable under pressure.



Patterns are power.



p.s., keep learning!