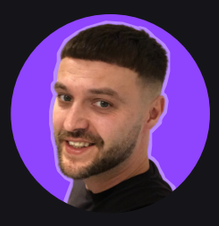
Learn CSS By Use Cases

ebook by Joe Harrison

Properties

display: grid



Joe Harrison
@frontendjoe

display: grid

Properties

Intro

Grid (or the Grid Layout Module) is a grid based 2D layout system, made up of rows and columns. It solves the problem of creating advanced global and micro layouts, with its keyword based syntax and intuitive properties.

Syntax

There’s quite a few pre-defined keywords to learn when focusing on grid. From my recent work, I have found that I do not really need most of them so I’ll only share what I consider to be the best parts. The grid-template-areas syntax is formatted as it is displayed using strings (it looks really futuristic in the code).

display: grid

Properties

Special Power

The repeat() function is very useful for complex grid layouts, I just wish it worked for every CSS property but sadly it does not. I've found that grid offers the cleanest way to center a single child of an element - it's only two lines of CSS code.

Tips

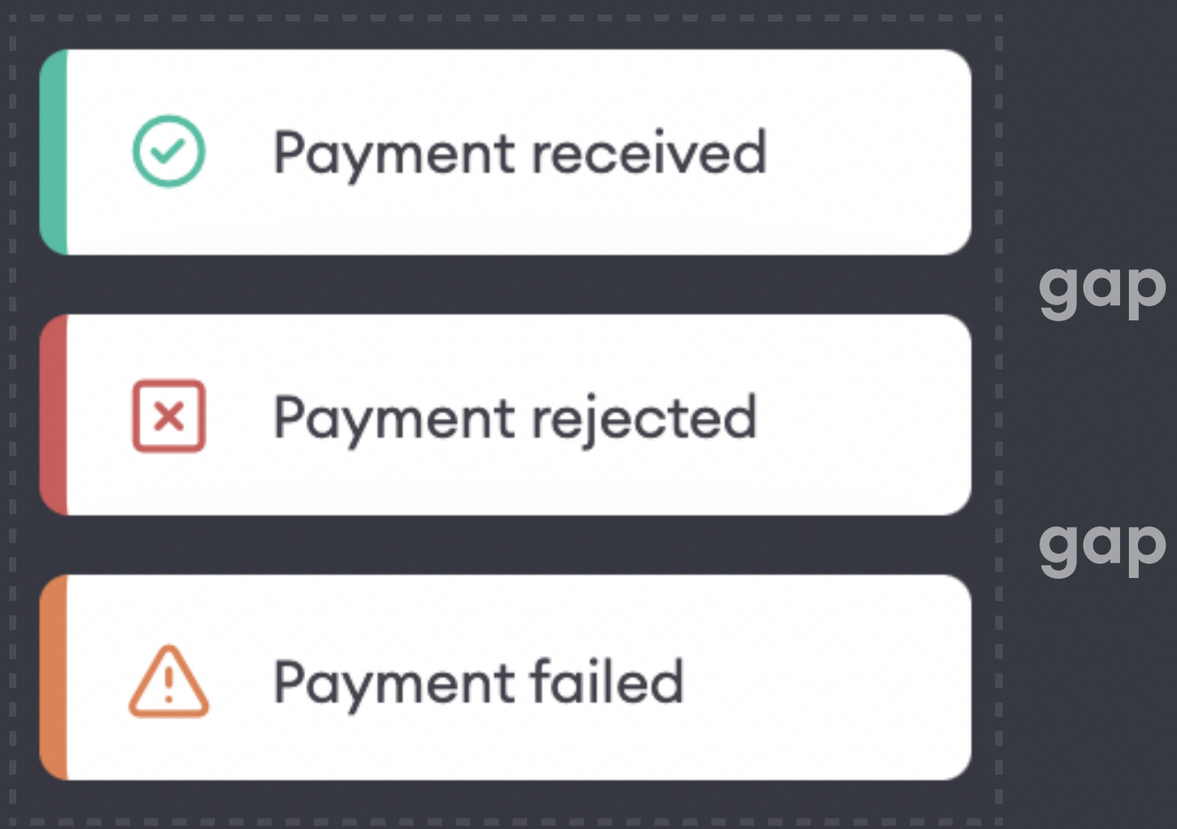
It's easy to get overwhelmed when starting to learn grid. I would maybe try flexbox first (it's a bit more intuitive and the 1D simplicity of rows or columns is easy to get into). Properties like align-items and gap work on both too though, which is always good to know.

place-items

```
.center-button {
  display: grid;
  place-items: center;
}
```

place-items is shorthand for **align-items** (vertical) and **justify-items** (horizontal). A single value sets both, neat

gap




```css
.stack {
  display: grid;
  gap: 1rem;
}
```

The gap property defines the space applied between child elements

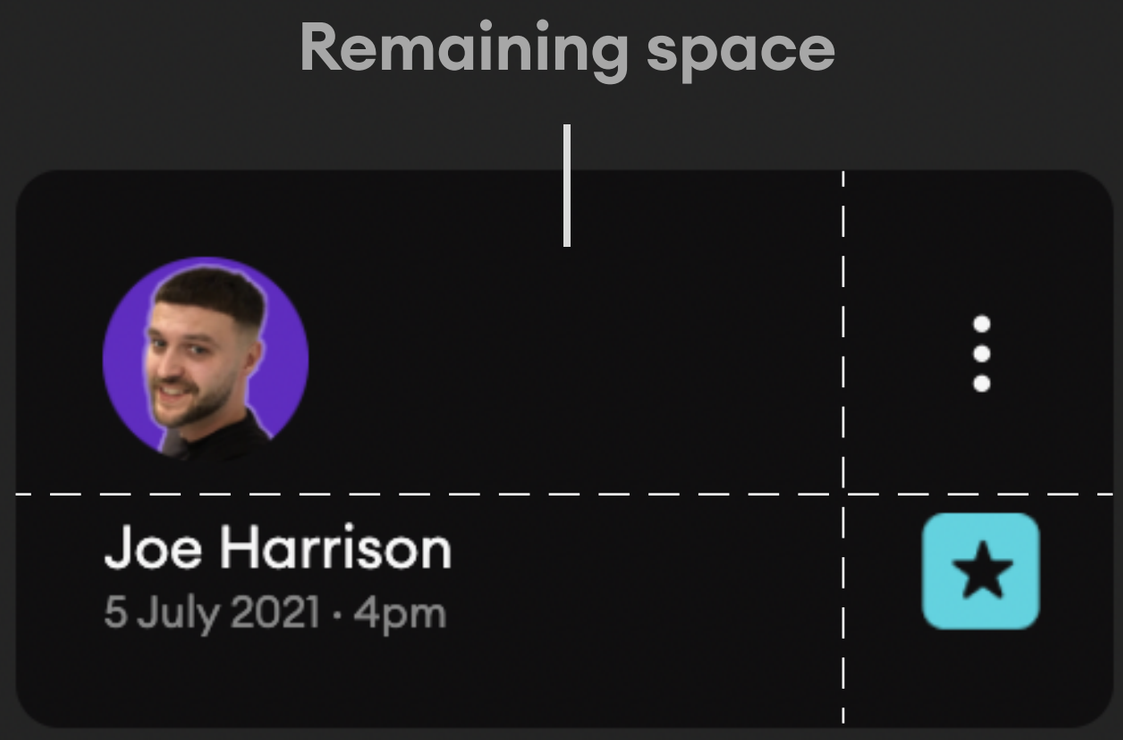
grid-template-columns

```
.picker {
  display: grid;
  grid-template-columns: repeat(3, 3rem);
}
```

Times Length

repeat is a function which only works inside grid - it will resolve as 3rem 3rem 3rem;

Rows and columns (2D)



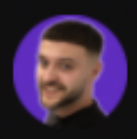
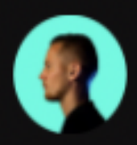
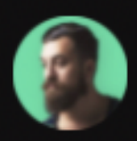
auto will fill any remaining space

```
.card {
  display: grid;
  grid-template-columns: auto 4rem;
  grid-template-rows: 4rem 2rem;
  width: 12rem;
}
```

Requires **width** if not set by it's parent container

1st 2nd etc

Fractions

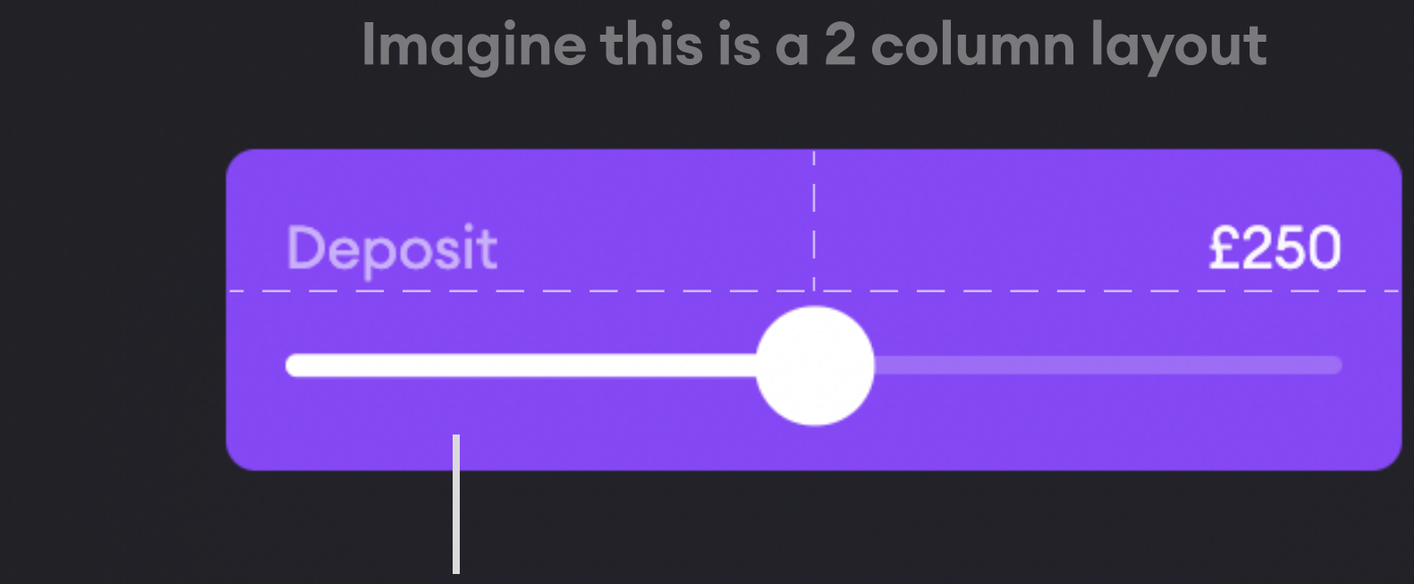
	Mon	Tue	Wed	Thu	Fri	Sat	Sun
	87	-	131	-	67	78	321
	-	-	78	-	56	-	124
	25	100	-	76	99	45	-

1fr (1 fraction)

Fractions will size evenly

```
.matrix {
  display: grid;
  grid-template-columns: 3rem repeat(7, 1fr);
  grid-template-rows: auto repeat(3, 3rem);
  align-items: center;
}
```

grid-column



Start End

```
.card-slider {
  grid-column: 1 / span 2;
}
```

Values are separated by a **/**. If the integer after **span** is omitted it defaults to 1. Negative integers or 0 are invalid

grid-template-areas

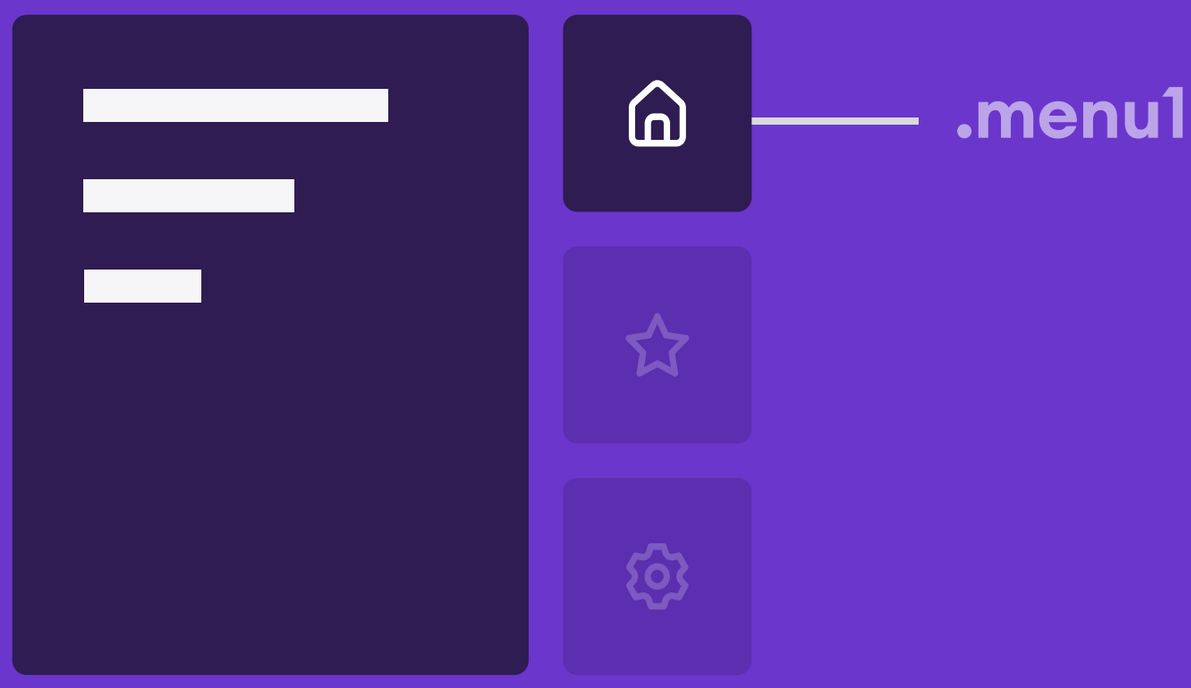
Equates to
6 squares
of the grid

See how we can
visualize the 6
squares here

```
.interface {
  gap: 0.5rem;
  grid-template-areas:
    "content content menu1"
    "content content menu2"
    "content content menu3";
}
```

We can even add column and row settings to define the element sizes, alternatively we can let the actual areas decide

grid-area




```
.menu1 {
  grid-area: menu1;
  background: #2F1C52;
}
```

Custom keyword

Define styles for the element that sits inside our template-areas

display: grid

Properties

Knowledge Gained

- 🏆 Grid is a 2D layout system made up of a rows and columns
- 🏆 Grid solves the problem of creating grid based website layouts
- 🏆 Grid offers the cleanest way to center a single child element
- 🏆 Some grid properties accept the repeat function which reduces code written
- 🏆 Grid is harder to learn than flexbox

Joe Harrison
@frontendjoe