ScopeTreasure Courses

CSS COURSES · LESSON 2.7 · CODE SCOPE SERIES

CSS Grid

Two-dimensional layouts: grid lines, fr units, repeat(), template areas and spanning — full page layouts and galleries without floats.

Introduction

Flexbox arranges one line. But a page is not a line — it has rows **and** columns at the same time. **CSS Grid** is the **two-dimensional** layout system: you draw the table first (how many columns, how many rows, how big), then drop items into the cells. Galleries, dashboards, magazine layouts and whole page skeletons are grid's home ground. The decision rule from the video: a **line** of things → Flexbox; a **layout** of things → Grid.
(Roman Urdu: Flexbox aik saf hai, Grid poora naksha hai — satr aur column dono aik saath.)

Key points

- `display: grid` on the parent makes a **grid container**; its children become **grid items** placed into cells. You define the columns with **`grid-template-columns`** and rows with **`grid-template-rows`**.
- **The `fr` unit** = one **fraction** of the free space. `grid-template-columns: 1fr 2fr` gives the second column twice the width of the first, whatever the screen size. Fixed units can mix in: `200px 1fr` = a fixed sidebar + the rest flexible.
- **`repeat()`** is the shortcut that saves your wrists: `repeat(3, 1fr)` = three equal columns. Its partner trick `repeat(auto-fill, minmax(200px, 1fr))` makes *responsive* galleries that choose their own column count.
  (Roman Urdu: fr matlab "hissa" — jitna hissa likha, utni jagah mili.)
- **`gap`** works here too — `gap: 16px` for both directions, or `row-gap` / `column-gap` separately.
- **Spanning:** an item can claim several cells: `grid-column: span 2` makes it two columns wide — the wide feature card in a gallery.
- **`grid-template-areas`** — the layout drawn as words: name each zone in a string map (`"header header" "nav main"`), then tell each element its name with `grid-area: header`. The CSS *looks* like the page.
- **Grid lines** are numbered (1, 2, 3… between the tracks) if you ever need exact placement: `grid-column: 1 / 3` means "from line 1 to line 3".

Code examples

Example 1 — A 3-column gallery with repeat() and fr

```
.gallery {
  display: grid;
  grid-template-columns: repeat(3, 1fr);
  gap: 16px;
}
.gallery figure {
  background: white;
  border: 1px solid #e3e6ec;
  border-radius: 12px;
  padding: 14px;
}
```

What it does: `repeat(3, 1fr)` draws three equal columns; each new item flows into the next cell, row after row, and `gap` keeps a clean 16px channel between them all — rows *and* columns, one line of code.

What you will see: six photo cards in a perfect 3 × 2 table — every card the same width, every gap identical. Change `repeat(3, 1fr)` to `repeat(4, 1fr)` and the whole gallery re-draws itself in fours.

Example 2 — Mixed units and a spanning card

```
.dashboard {
  display: grid;
  grid-template-columns: 200px 1fr 1fr;
  gap: 14px;
}
.wide { grid-column: span 2; }
```

What it does: The first column is pinned at 200px (a sidebar); the other two share the remaining space equally. An item with `.wide` stretches across two of the columns — the classic "big chart above small stats" dashboard arrangement.

What you will see: a fixed sidebar and two flexible columns; the wide card sits across both flexible columns. Resize the window: the sidebar never moves, the flexible columns breathe.

Example 3 — A full page drawn with template areas

```
.page {
  display: grid;
  grid-template-areas:
    "header header"
    "nav    main"
    "footer footer";
  grid-template-columns: 220px 1fr;
  gap: 12px;
}
header { grid-area: header; }
nav    { grid-area: nav; }
main   { grid-area: main; }
footer { grid-area: footer; }
```

What it does: The map in `grid-template-areas` *is* the layout — header across the top, nav + main in the middle row, footer across the bottom. Each element simply claims its named area. Rearranging the page means redrawing the strings, not rewriting the HTML.

What you will see: a complete page skeleton in seconds. Read the strings out loud and you have described the finished page — that readability is why template areas are beloved in real projects.

Quick recap

- Grid = **two-dimensional**: define columns/rows first, items fall into cells.
- `fr` = a share of free space; `repeat(3, 1fr)` = three equal columns; `gap` spaces both directions.
- `span` stretches items; `grid-template-areas` draws the whole page as a word map.
- Line → Flexbox. Layout → Grid.

Common mistakes

1. **Reaching for Grid for a simple row** — a navbar or a button strip is one line; Flexbox does it in fewer lines with less thinking. Grid for rows *and* columns together.
2. **Counting columns instead of lines** — `grid-column: 1 / 3` spans **two** columns because it runs from line 1 to line 3 (lines are the fences, columns are the fields). Off-by-one placement bugs live here.
3. **Forgetting `gap` and using item margins** — margins double up between items and stick out at the container's edges. Grid's `gap` was invented precisely to end that mess.

Practice

1. Rebuild the Example 1 gallery with `repeat(auto-fill, minmax(200px, 1fr))` and resize the window — count how the column number chooses itself.

2. Draw your profile page as template areas (header / about / skills / footer) and assemble it. Then move "skills" beside "about" by editing only the strings.

Practice quiz

1. What does the fr unit mean in grid-template-columns?

- A fixed number of pixels
- One fraction (share) of the free space
- The number of rows
- A percentage of the font size

Answer: One fraction (share) of the free space

2. What does repeat(3, 1fr) draw?

- Three rows of fixed height
- Three equal columns
- One column repeated three times on top of itself
- A 3-pixel gap

Answer: Three equal columns

3. How many columns does grid-column: 1 / 3 span?

- Three columns
- Two columns, because lines 1 to 3 fence two fields
- One column
- It depends on the screen width

Answer: Two columns, because lines 1 to 3 fence two fields

Watch the video lesson: https://youtu.be/qnUntNy2PUU

Read online: https://scopetreasure.com/notes/css-grid · More free tools and notes at https://scopetreasure.com