CSS Layout Overview

* CSS positions boxes and controls their arrangement on a web page. [Reddit+7MDN Web Docs+7MDN Web Docs+7](https://developer.mozilla.org/en-US/docs/Learn_web_development/Core/CSS_layout?utm_source=chatgpt.com)
* Starts with **normal flow** (blocks stack vertically, inline elements wrap). [Reddit+9MDN Web Docs+9Wikipedia+9](https://developer.mozilla.org/en-US/docs/Learn_web_development/Core/CSS_layout/Introduction?utm_source=chatgpt.com)

Floats

* Originally used to wrap text around images. Also used for columns in the past.
* Now largely replaced by newer layout methods, but still useful for text-image wrapping.

Positioning

* Controls taking elements out of normal flow. Values include static, relative, absolute, fixed, sticky. [MDN Web Docs+8MDN Web Docs+8MDN Web Docs+8](https://mdn2.netlify.app/en-us/docs/learn/css/css_layout/?utm_source=chatgpt.com)
* relative shifts elements but keeps space; absolute removes from flow; fixed sticks to viewport; sticky toggles based on scroll; layering controlled with z-index.

Flexbox

* One-dimensional layout (rows or columns). [MDN Web Docs+2MDN Web Docs+2MDN Web Docs+2](https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_flexible_box_layout/Basic_concepts_of_flexbox?utm_source=chatgpt.com)
* Defines a **flex container** (display: flex) whose children are flex items. [CSS-Tricks+3MDN Web Docs+3Wikipedia+3](https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_flexible_box_layout/Basic_concepts_of_flexbox?utm_source=chatgpt.com)
* Uses **main and cross axes** with properties like justify-content, align-items, flex-wrap. [MDN Web Docs+11MDN Web Docs+11Wikipedia+11](https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_flexible_box_layout/Basic_concepts_of_flexbox?utm_source=chatgpt.com).

CSS Grid

* Two-dimensional layout: rows + columns. [Wikipedia+2MDN Web Docs+2Reddit+2](https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_grid_layout?utm_source=chatgpt.com)
* Use display: grid, set grid-template-columns/rows, and place items via lines or named areas. [MDN Web Docs](https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_grid_layout?utm_source=chatgpt.com)

Responsive Design & Media Queries

* Builds sites that adapt to different screen sizes/resolutions. [MDN Web Docs+1MDN Web Docs+1](https://developer.mozilla.org/en-US/docs/Learn_web_development/Core/CSS_layout?utm_source=chatgpt.com)
* Media queries use @media rules (e.g. max-width: 600px) to apply specific CSS under set conditions.

Layout Tools Comparison

* **Floats**: old method, best for text wrapping.
* **Positioning**: precise placement, can overlap.
* **Flexbox**: great for single-axis (rows/columns) layout.
* **Grid**: best for 2D, complex designs.
* **Responsive**: use media queries to switch methods as needed. [Wikipedia+8MDN Web Docs+8MDN Web Docs+8](https://developer.mozilla.org/en-US/docs/Learn_web_development/Core/CSS_layout/Introduction?utm_source=chatgpt.com)[MDN Web Docs+1Reddit+1](https://developer.mozilla.org/en-US/docs/Learn_web_development/Core/CSS_layout?utm_source=chatgpt.com)

Best Practices

* Start with a well-structured page in **normal flow**. [MDN Web Docs+3MDN Web Docs+3MDN Web Docs+3](https://developer.mozilla.org/en-US/docs/Learn_web_development/Core/CSS_layout/Introduction?utm_source=chatgpt.com)
* Use simple CSS (margins, padding) before using advanced layouts.
* Choose layout methods depending on the job: one-axis (Flexbox), two-axis (Grid), layering or scrolling effects (Positioning), wrap text around images (Floats).