Positioning (static, relative, absolute, fixed, sticky)

Place elements exactly where you want: static, relative, absolute, fixed and sticky — plus z-index for who sits on top.

Introduction

Flexbox and Grid arrange elements *in the flow* — each box politely after the last. But real pages need rule-breakers: a "SALE" badge pinned to a photo's corner, a navbar that stays on top while you scroll, a popup floating above everything. The `position` property is the rule-breaker system. It has five values — **static, relative, absolute, fixed, sticky** — and together with the offset properties (`top`, `right`, `bottom`, `left`) and `z-index`, they place any element exactly where you command. *(Roman Urdu: Position woh system hai jo element ko us ki aam jagah se utha kar aap ki marzi ki jagah rakhta hai.)*

Key points

- **static** — the default. The element sits in the flow, in order. `top` / `left` and friends are **ignored** completely — they are asleep on a static element.
- **relative** — the element first sits in its normal place (its space in the flow is **kept** — a ghost of it stays), then nudges by your offsets: `top: 10px` moves it 10px *down* from where it was. The neighbours do not move.
- **absolute** — the element **leaves the flow entirely** (its old space closes up) and positions itself against its **nearest positioned ancestor** — the closest parent with `position: relative` (or absolute/fixed). No such ancestor? It uses the whole page. This is the badge-on-a-photo tool.

  (Roman Urdu: Absolute bachcha us parent ke hisaab se bethata hai jo khud relative ho.)
- **fixed** — positions against the **screen** (viewport). It does not scroll away: fixed navbars, chat buttons, and back-to-top buttons live here.
- **sticky** — the hybrid: behaves as relative *until* the page scrolls it to your offset (usually `top: 0`), then it **sticks** like fixed — but only inside its parent's area. Section headings and table headers are its natural home.
- **Offsets** `top / right / bottom / left` tell a positioned element its distance from the reference edges. `top: 0; right: 0` = the reference's top-right corner.
- **`z-index`** — when positioned elements overlap, the bigger `z-index` sits on top. It only works on **positioned** elements (not static), and it is the stacking referee for popups, menus and modals.

Code examples

Example 1 — The SALE badge (relative parent, absolute child)

```
.product { position: relative; width: 260px; }
.product img { width: 100%; display: block; }
.badge {
  position: absolute;
  top: 10px; right: 10px;
  background: #e8490f; color: white;
  padding: 4px 10px; border-radius: 999px;
  font-weight: 700;
}
```

```
<div class="product">
  <img src="shoes.jpg" alt="Running shoes">
  <span class="badge">SALE</span>
</div>
```

What it does: `.product` is `relative` — it moves nowhere, but it becomes the **reference frame**. The badge is `absolute`, so it leaves the flow and pins itself 10px from the reference's top and right edges: the photo's corner, at any screen size.

What you will see: an orange SALE pill glued to the photo's top-right corner. Delete `position: relative` from `.product` and re-load — the badge jumps to the *page's* corner. That one experiment teaches absolute better than any diagram.

Example 2 — Fixed navbar + sticky section heading

```
.topbar {
  position: fixed;
  top: 0; left: 0; right: 0;
  background: #0b0f19; color: #faf9f6;
  padding: 12px 20px;
  z-index: 100;
}
.section-title {
  position: sticky;
  top: 52px;             /* sticks just under the fixed bar */
  background: white;
  padding: 8px 0;
}
body { padding-top: 52px; }  /* room for the fixed bar */
```

What it does: The topbar is `fixed` — pinned to the screen's top, `z-index: 100` keeps it above content scrolling beneath, and the body padding stops it from covering the first lines. Each `.section-title` is `sticky`: it scrolls normally until it reaches 52px from the top, then sticks there while its section passes — and politely lets go when the next section arrives.

What you will see: scroll the page: the dark bar never moves; each section title parks under it while its section scrolls by, then hands over to the next title. Two position values cooperating — the pattern behind docs sites and settings apps everywhere.

Example 3 — z-index settles an overlap

```
.card-a, .card-b { position: absolute; width: 180px; height: 120px; }
.card-a { background: #ffd9cc; top: 30px; left: 30px; }
.card-b { background: #d8f3dc; top: 70px; left: 90px; z-index: 5; }
```

What it does: Two absolutely-positioned cards overlap. Without help, the later element in the HTML paints on top. `z-index: 5` on card-b settles it by number instead: 5 beats the default (auto/0).

What you will see: the green card-b firmly on top of the orange card-a in the overlap region. Change card-b's `z-index` to `-1` and it slides *behind*. (Now remove `position` from card-b and watch `z-index` stop working entirely — the referee only judges positioned players.)

Quick recap

- **static**: flow, offsets asleep. **relative**: nudged, space kept, becomes a reference frame. **absolute**: out of flow, placed against the nearest positioned ancestor. **fixed**: against the screen, never scrolls. **sticky**: relative until scroll, then sticks within its parent.
- `z-index` = stacking order, positioned elements only.

Common mistakes

1. **The badge in the page corner** — an absolute child whose parent is not positioned flies to the nearest positioned ancestor (or the page). The fix is never more offsets; it is `position: relative` on the parent you meant.
2. **Setting top/left on a static element** — static ignores offsets *by design*. If `top: 20px` "does nothing", the element has no `position` — add `relative`.
3. **A fixed bar covering the page top** — fixed elements leave the flow, so the content slides underneath them. Give the body (or the first section) top padding equal to the bar's height.

Practice

1. Build a product card with a price badge (bottom-left) and a SALE badge (top-right), both absolute inside one relative parent. Then remove the parent's `position` and describe exactly what changed.

2. Make a long page (five sections) with a sticky heading per section and a fixed "back to top" button in the bottom-right corner, `z-index` set so the button is always on top.

Practice quiz

1. What does an absolutely positioned element position itself against?

- The whole screen, always
- Its nearest positioned ancestor, or the page if there is none
- The element before it in the HTML
- The centre of the page

Answer: Its nearest positioned ancestor, or the page if there is none

2. What is position: fixed positioned against?

- Its parent element
- The screen (viewport), so it never scrolls away
- The nearest section heading
- The bottom of the page only

Answer: The screen (viewport), so it never scrolls away

3. When does z-index work?

- On every element, always
- Only on positioned elements, never on static ones
- Only on fixed elements
- Only when the element also has a border

Answer: Only on positioned elements, never on static ones

Watch the video lesson: https://youtu.be/Bz3Yti5TF9w

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