JavaScript Equality

Computing

Why == and === differ, and the coercion table that explains the surprises.

=== compares type and value. == converts types first, using rules almost nobody remembers.

'' == 0 → true · '' === 0 → false

0 == '0' — **true**

String converted to number.

null == undefined — **true**

The one genuinely useful case.

NaN === NaN — **false**

Use Number.isNaN() instead.

THE COMMON MISTAKE

× if (x == false) to check for falsiness

✓ if (!x)

[] == false is true, but [] is truthy. The coercion rules and the truthiness rules disagree.

Falsy values, all of them: false, 0, -0, 0n, '', null, undefined, NaN. Everything else is truthy, including [] and {}.