JavaScript Conditional Logic Cheat Sheet

if/else, switch & ternary syntax, truthy/falsy reference, and the defect patterns that show up most in code review — for QA, BA, and IT professionals.

1. if/else vs. switch vs. Ternary

Construct	Best For	Watch For
if / else if / else	Multi-factor conditions, ranges, mixed variables	Order matters — an early broad condition can silently override a later rule
switch	One variable checked against several fixed values	Missing break causes fall-through — multiple cases execute
Ternary (? :)	Simple single-value assignment	Nested ternaries (2+ levels) are hard to review — flag in code review
&& / || (short-circuit)	Defensive null checks, default values	Stops evaluating once the outcome is known — order affects side effects

2. Truthy / Falsy Quick Reference

Value	Evaluates As	Testing Risk
`0`	Falsy	A legitimate zero can be treated as missing data
`""` (empty string)	Falsy	Blank vs. intentionally empty needs an explicit check
`null` / `undefined`	Falsy	Look identical in a loose check but signal different failures
`"0"` (string)	Truthy	Non-empty strings are truthy regardless of content
`[]` (empty array)	Truthy	Empty result sets still pass — test .length explicitly

== vs ===: == converts types before comparing ("5" == 5 is true). === compares value and type together ("5" === 5 is false). Default to === — loose equality passes code review looking correct and then misbehaves with mixed data types.

3. Five Defect Patterns to Recognize on Sight

Assignment instead of comparison
if (status = "approved") assigns instead of compares — always evaluates true. Linters catch this in modern setups; legacy code may not have linting enabled.

Unreachable else branches
An overly broad earlier condition can make a later branch impossible to reach. No error is thrown — the branch just silently never executes.

Case-sensitive string mismatches
"Approved" !== "approved". Common at integration points where one system sends title case and another expects lowercase.

Missing default / else
No fallback branch means unmatched input does nothing, with no error logged — often the actual defect in a 'nothing happened' bug report.

Flipped && / || in business rules
Compiles cleanly, passes casual review, and silently breaks the intended logic. Compare the operator against the documented rule symbol by symbol.