Decisions

Ref Javascript

Booleans & comparisons

A Boolean is either `true` or `false`. A comparison gives a Boolean:

Operator	Meaning	Example	Result
`===`	equal (same value and same type)	`5 === 5`	`true`
`!==`	not equal	`5 !== 3`	`true`
`< >`	less than, greater than	`2 > 7`	`false`
`<= >=`	at most, at least	`7 >= 7`	`true`

```
console.log(5 === 5);      // true
console.log(5 === "5");    // false  (a number is not a string)
console.log(5 == "5");     // true   (== converts first: avoid it)
```

- Always use `===` and `!==`. The older `==` converts types first and gives surprising answers.

For Boolean conditions, use the logical operators below. With other values, `&&` and `||` return one of their operands; they do not always return a Boolean:

- `&&` (and): true only if **both** sides are true.
- `||` (or): true if **at least one** side is true.
- `!` (not): turns `true` into `false` and back.

```
const age = 15;
console.log(age >= 13 && age <= 19);   // true   (a teenager)
console.log(age < 5 || age > 65);      // false
console.log(!(age > 18));              // true
```

if / else if / else

An `if` statement runs a block of code when its condition is truthy (treated as true). With Boolean conditions, this means the value `true`. `else if` tests another condition; `else` catches everything left:

```
const mark = 72;
if (mark >= 80) {
  console.log("A");
} else if (mark >= 60) {
  console.log("B");
} else {
  console.log("C");
}
// prints B
```

- JavaScript checks the conditions from the top and runs **only the first** true branch .
- The condition goes in round brackets; the code to run goes in curly braces `{ }`.

So the order matters. Put the strictest test first:

```
const temp = 35;
if (temp > 30) {
  console.log("hot");
} else if (temp > 20) {
  console.log("warm");   // skipped: the first branch already ran
}
// prints hot
```

switch

A `switch` compares one value against several fixed cases. It is neater than a long `else if` chain when every test is "is it equal to …?":

```
const day = "Sat";
switch (day) {
  case "Sat":
  case "Sun":
    console.log("weekend");
    break;
  case "Fri":
    console.log("nearly there");
    break;
  default:
    console.log("school day");
}
// prints weekend
```

- `switch` compares with `===`, so `case 1:` does not match the string `"1"`.
- `break` ends the `switch`. Without it, the code **falls through** into the next case.
- `default` runs when no case matches, like a final `else`.
- Two cases with no code between them share the same code, as `"Sat"` and `"Sun"` do here.

Common mistakes

- `if (x = 5)` assigns instead of comparing, and the test is then always true. Write `if (x === 5)`.
- `if (age > 13 && < 19)` is an error: repeat the variable, `age > 13 && age < 19`.
- A missing `break` in a `switch` runs the next case too.