条件判断

Ref Javascript

布尔值与比较

布尔值 (Boolean) 只能是 `true` 或 `false`。比较 (comparison) 会得出一个布尔值:

运算符	含义	例子	结果
`===`	相等 (值相同且类型相同)	`5 === 5`	`true`
`!==`	不相等	`5 !== 3`	`true`
`< >`	小于、大于	`2 > 7`	`false`
`<= >=`	不大于、不小于	`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)
```

- 始终用 `===` 和 `!==`。旧的 `==` 会先转换类型, 结果常常出人意料。

用逻辑运算符 (logical operator) 把条件连起来:

- `&&`(并且): 只有**两边**都为真时才为真。
- `||`(或者): **至少一边**为真时就为真。
- `!`(非): 把 `true` 变成 `false`, 反之亦然。

```
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

`if` 语句只在条件 (condition) 为真时运行一个代码块 (block)。`else if` 检查另一个条件;`else` 接住剩下的所有情况:

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

- JavaScript 从上往下检查条件, **只运行第一个**为真的分支 (branch)。
- 条件写在圆括号里; 要运行的代码写在花括号 `{ }` 里。

所以顺序很重要。把最严格的条件放在前面:

```
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 语句

`switch` 把一个值和几个固定的情况 (case) 逐一比较。当每个判断都是" 它是否等于……?" 时, 它比一长串 `else if` 更整齐:

```
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` 用 `===` 比较, 所以 `case 1:` 匹配不到字符串 `"1"`。
- `break` 结束 `switch`。没有它, 代码会**贯穿** (fall through) 到下一个 case 里继续运行。
- 没有任何 case 匹配时运行 `default`, 就像最后的 `else`。
- 两个 case 之间没有代码时, 它们共用同一段代码, 就像这里的 `"Sat"` 和 `"Sun"`。

常见错误

- `if (x = 5)` 是赋值而不是比较, 这个判断永远为真。要写 `if (x === 5)`。
- `if (age > 13 && < 19)` 是错误的: 要把变量再写一遍,`age > 13 && age < 19`。
- `switch` 里漏掉 `break`, 会把下一个 case 也一起运行。