值、变量与类型

Ref Javascript

let 与 const

变量 (variable) 是一个有名字、装着值的盒子。用 `let` 或 `const` 创建它, 然后用它的名字:

```
let score = 10;
score = score + 5;      // let: the value can change
console.log(score);     // 15

const name = "Mei";     // const: the value never changes
console.log(name);      // Mei
```

- `=` 的意思是**赋值** (assign): 先算出右边, 再把结果存进左边的盒子。
- 默认用 `const`。只有当值以后必须改变时才用 `let`。
- 给 `const` 赋新值是错误的:`name = "Tom"` 会让程序停下。

数字与数学运算

JavaScript 只有一种数字类型, 整数和小数都用它。算术运算符 (operator) 有:

运算符	含义	例子	结果
`+ -`	加、减	`7 - 2`	5
`* /`	乘、除	`7 / 2`	3.5
`%`	余数 (remainder)	`7 % 2`	1
`**`	乘方	`2 ** 3`	8

```
console.log(7 / 2);              // 3.5
console.log(Math.floor(7 / 2));  // 3   (whole-number division)
console.log(7 % 2);              // 1   (odd numbers leave 1)
console.log(2 + 3 * 4);          // 14  (* before +)
console.log((2 + 3) * 4);        // 20
```

`Math` 里有很多有用的函数:

```
console.log(Math.round(2.5));    // 3
console.log(Math.max(4, 9, 2)); // 9
console.log(Math.sqrt(49));      // 7
```

- `Math.random()` 给出一个从 0 到 1(不含 1) 的随机 (random) 数。`Math.floor(Math.random() * 6) + 1` 就是掷一次骰子, 得到 1 到 6。

exercises.lol exercises.com.cn

- 小数以二进制存储, 所以 `0.1 + 0.2` 打印出 `0.30000000000000004`。显示结果之前先取整:`(0.1 + 0.2).toFixed(2)` 是 `"0.30"`。

类型与类型转换

每个值都有一个类型 (type)。`typeof` 告诉你是哪一种:

```
console.log(typeof 42);          // number
console.log(typeof "42");        // string
console.log(typeof true);        // boolean
console.log(typeof undefined);   // undefined
```

- 用 `let x;` 创建、没有给值的变量是 `undefined`。
- `null` 表示" 没有值", 是有意设置的。

`+` 遇到字符串时会**拼接**而不是相加, 所以要先把文本转换 (convert) 成数字:

```
console.log("5" + 1);            // 51  (joined)
console.log(Number("5") + 1);    // 6   (added)
console.log(parseInt("12px"));   // 12  (reads the number at the start)
console.log(String(9) + "!");    // 9!
```

- 不是数字的文本会得到 `NaN`(" 不是一个数 "):`Number("abc")` 是 `NaN`。
- `-`、`*` 和 `/` 会自动把文本转成数字 (`"6" * 2` 是 12), 但不要依赖这一点: 要有意识地转换。

常见错误

- `=` 是赋值, 不是比较。比较要用 `===`(第 4 章)。
- `"5" + 1` 是 `"51"`。用输入做数学运算之前, 先用 `Number(...)` 转换。
- `Math.round(2.5)` 是 3, 但 `Math.round(-2.5)` 是 -2: 它把.5 向较大的数取整。