字符串

Ref Javascript

Characters & length

字符串是用引号包裹的文本。对于简单字母，每个字符有一个位置索引；首字符索引为 0。JavaScript 按 UTF-16 码点计数：一个表情符号可能占两个位置。

```
const word = "Script";
console.log(word.length);      // 6
console.log(word[0]);          // S
console.log(word[2]);          // r
console.log(word.at(-1));      // t  (counts back from the end)
console.log(word[word.length - 1]);   // t
```

- 末位索引为 `length - 1`。`word[6]` 访问越界时返回 `undefined`，而非报错。
- 字符串不可变：`word[0] = "s"` 无法修改原字符串，在严格模式下会抛出 TypeError。应构建新字符串替代。

使用 `+` 拼接字符串，该操作称为拼接：

```
const first = "Ada";
const last = "Lovelace";
console.log(first + " " + last);   // Ada Lovelace
```

String methods

方法是属于某个值的函数，用点号调用。字符串方法返回**新值**，原值不变。

```
const city = "  Hong Kong ";
console.log(city.trim());                  // Hong Kong
console.log(city.trim().toUpperCase());    // HONG KONG
console.log(city.length);                  // 12  (the spaces are still
↪  there)
```

方法	含义	例子	结果
`.toUpperCase()` / `.toLowerCase()`	改变大小写	`"Hi".toLowerCase()`	`"hi"`
`.trim()`	去除两端空格	`" hi ".trim()`	`"hi"`
`.includes(t)`	是否包含 t?	`"cat".includes("a")`	`true`
`.indexOf(t)`	t 首次出现的位置，或 -1	`"cat".indexOf("t")`	`2`
`.slice(a, b)`	从 a 到（不含）b 的子串	`"Script".slice(0, 3)`	`"Scr"`
`.split(sep)`	拆分为数组	`"a,b,c".split(",")`	`["a","b","c"]`
`.replace(a, b)`	将第一个 a 替换为 b	`"cat".replace("c", "b")`	`"bat"`

```
const email = "mei@school.org";
const at = email.indexOf("@");
console.log(email.slice(0, at));          // mei
console.log(email.slice(at + 1));         // school.org
console.log(email.split("@"));            // ["mei","school.org"]
```

Template literals

模板字面量用反引号（多数键盘上位于 1 键左侧）包裹。可在 `${ }` 内嵌入任意表达式，其值将直接插入文本中：

```
const name = "Mei";
const age = 17;
console.log(`${name} is ${age} years old.`);   // Mei is 17 years old.
console.log(`Next year: ${age + 1}`);          // Next year: 18
```

- 反引号也可以容纳多行文本。
- 这比许多 `+` 符号更易读：请将 `"Hi " + name + "!"` 与 `` `Hi ${name}!` `` 进行比较。

```
const price = 4.5;
console.log(`Two cost $${(price * 2).toFixed(2)}`);   // Two cost $9.00
```

常见错误

- `${ }` 仅在反引号内有效。在 `"Hi ${name}"` 中，文本按原样输出。
- 方法不会改变字符串本身：单独一行的 `city.trim()` 不会执行任何操作。请存储结果：`const clean = city.trim();`。

- `slice(a, b)` 在 `b` **之前**停止，就像最后一个索引是 `length - 1` 一样。