# 字符串的扩展

模板字符串

1. 使用反引号，直接写变量
2. 多行字符串
3. 告别+拼接字符串

1.以当作普通字符串使用，也可以用来定义多行字符串，或者在字符串中嵌入变量。

2反斜杠转义反引号。

3.多行字符串保留空格和缩进

4.trim方法消除换行

5.变量名写在${}之中

6.大括号内部可以放入任意的 JavaScript 表达式，可以进行运算，以及引用对象属性。

7.大括号内部还能调用函数。

8.还能嵌套。

9.引用模板字符串本身

1. **[字符的 Unicode 表示法](http://es6.ruanyifeng.com/" \l "docs/string#字符的 Unicode 表示法)**

1. 6 种方法可以表示一个字符。

'\z' === 'z' // true

'\172' === 'z' // true

'\x7A' === 'z' // true

'\u007A' === 'z' // true

'\u{7A}' === 'z' // true

2.ES6 提供了codePointAt方法，能够正确处理 4 个字节储存的字符，返回一个字符的码点。

let s = ';

s.codePointAt(0) // 134071s.codePointAt(1) // 57271

s.codePointAt(2) // 97

## 3.String.fromCodePoint()

## 4.字符串的遍历器接口for...of [§](http://es6.ruanyifeng.com/" \l "docs/string#字符串的遍历器接口) [⇧](http://es6.ruanyifeng.com/" \l "docs/string)

除了遍历字符串，这个遍历器最大的优点是可以识别大于0xFFFF的码点，传统的for循环无法识别这样的码点。

let text = String.fromCodePoint(0x20BB7);

for (let i = 0; i < text.length; i++) {

console.log(text[i]);}

// " "// " "for (let i of text) {

console.log(i);}

// "

1. at()

目前，有一个提案，提出字符串实例的at方法，可以识别 Unicode 编号大于0xFFFF的字符，返回正确的字符。

'abc'.at(0) // "a"'.at(0) // "

1. Unicode 正规化

'\u01D1'.normalize() === '\u004F\u030C'.normalize()

// true

7.

传统上，JavaScript 只有indexOf方法，可以用来确定一个字符串是否包含在另一个字符串中。ES6 又提供了三种新方法。

* ****includes()****：返回布尔值，表示是否找到了参数字符串。
* ****startsWith()****：返回布尔值，表示参数字符串是否在原字符串的头部。
* ****endsWith()****：返回布尔值，表示参数字符串是否在原字符串的尾部。

1. repeat方法返回一个新字符串，表示将原字符串重复n次。

9.padStart的常见用途是为数值补全指定位数。下面代码生成 10 位的数值字符串。

'1'.padStart(10, '0') // "0000000001"'12'.padStart(10, '0') // "0000000012"'123456'.padStart(10, '0') // "0000123456"

另一个用途是提示字符串格式。

'12'.padStart(10, 'YYYY-MM-DD') // "YYYY-MM-12"'09-12'.padStart(10, 'YYYY-MM-DD') // "YYYY-09-12"

1. **[模板字符串](http://es6.ruanyifeng.com/" \l "docs/string#模板字符串)**

compile函数的用法如下。

let parse = eval(compile(template));

div.innerHTML = parse({ supplies: [ "broom", "mop", "cleaner" ] });