7、数值的扩展

① 二进制和八进制表示法:

- ES6 提供了二进制和八进制数值的新的写法，分别用前缀0b（或0B）和0o（或0O）表示。

```
0b111110111 === 503 // true
0o767 === 503 // true
```

② Number.isFinite(), Number.isNaN()

- ES6 在Number对象上，新提供了Number.isFinite()和Number.isNaN()两个方法。
- Number.isFinite()用来检查一个数值是否为有限的（finite），即不是Infinity。

```
//注意，如果参数类型不是数值，Number.isFinite一律返回false。
Number.isFinite(15); // true
Number.isFinite(0.8); // true
Number.isFinite(NaN); // false
Number.isFinite(Infinity); // false
Number.isFinite(-Infinity); // false
Number.isFinite('foo'); // false
Number.isFinite('15'); // false
Number.isFinite(true); // false
```

- Number.isNaN()用来检查一个值是否为NaN。

```
//如果参数类型不是NaN，Number.isNaN一律返回false。
Number.isNaN(NaN) // true
Number.isNaN(15) // false
Number.isNaN('15') // false
Number.isNaN(true) // false
Number.isNaN(9/NaN) // true
Number.isNaN('true' / 0) // true
Number.isNaN('true' / 'true') // true
```

③ Number.parseInt(), Number.parseFloat()

- ES6 将全局方法parseInt()和parseFloat()，移植到Number对象上面，行为完全保持不变。

```
// ES5的写法
parseInt('12.34') // 12
parseFloat('123.45#') // 123.45

// ES6的写法
Number.parseInt('12.34') // 12
Number.parseFloat('123.45#') // 123.45
```

- 这样做的目的，是逐步减少全局性方法，使得语言逐步模块化。

```
Number.parseInt === parseInt // true
Number.parseFloat === parseFloat // true
```

④ Number.isInteger()

- Number.isInteger()用来判断一个数值是否为整数。

```
Number.isInteger(25) // true
Number.isInteger(25.1) // false
```

⑤ Math 对象的扩展：

- ES6 在 Math 对象上新增了 17 个与数学相关的方法。所有这些方法都是静态方法，只能在 Math 对象上调用。
- 具体详见手册