# ES5

## 1.1对象

ECMAScript 不使用诸如 C++，Smalltalk，Java 中的类。相反，对象可以通过各种方式创建，包括字面符号，或通过 构造器 创建对象然后运行代码初始化其全部或部分属性值，为这些属性分配初始值。每个构造器是一个拥有名为“prototype”的属性的函数。此属性用于实现 基于原型的继承 和 共享属性 。构造器通过 new 表达式创建对象：例如，new Date(2009,11) 创建一个新 Date 对象。不使用 new 调用一个构造器的结果，依赖构造器本身。例如，Date() 产生一个表示当前日期时间的字符串，而不是一个对象。

每个由构造器创建的对象，都有一个隐式引用 ( 叫做对象的原型 ) 链接到构造器的“prototype”属性值。再者，原型可能有一个非空 (non-null) 隐式引用链接到它自己的原型，以此类推，这叫做 原型链 。

## 1.2术语定义：

类型 type

原始值 primitive value: 原始值直接代表语言实现的最底层的数据

对象object: 对象是属性的集合，并有一个原型对象，原型可以是空值

构造器 constructor:

构造器的“prototype”属性值是一个原型对象，它用来实现继承和共享属性

原型 prototype： 为其他对象提供共享属性的对象

当构造器创建一个对象，为了解决对象的属性引用，该对象会隐式引用构造器的“prototype”属性。通过程序表达式 constructor.prototype 可以引用到构造器的“prototype”属性，并且添加到对象原型里的属性，会通过继承与所有共享此原型的对象共享。另外，可使用 Object.create 内置函数，通过明确指定原型来创建一个新对象。

原生对象 native object

内置对象 built-in object

所有内置对象是原生对象。一个内置构造器 (built-in constructor) 是个内置对象，也是个构造器。

宿主对象 host object

任何对象，不是原生对象就是宿主对象。

未定义值 undefined value

说明一个变量没有被分配值的一个原始值。

未定义类型 Undefined type

拥有唯一值“未定义值”的类型。

空值 null value

代表对象值故意留空的一个原始值。

空类型 Null type

拥有唯一值“空值”的类型。

布尔值 Boolean value

只有两个布尔值，true 和 false 。

布尔类型 Boolean type

布尔对象 Boolean object

字符串值 String value

字符串对象 String object

数字值

数字类型

数字对象

无穷 Infinity

NaN

值为 IEEE 754“Not-a-Number”的数字值

函数

函数除了拥有命名的属性，还包含可执行代码、状态，用来确定被调用时的行为。函数的代码不限于 ECMAScript。

内置函数

如 parseInt 和 Math.exp 就是内置函数。

属性 property

方法

内置方法、

特性attribute

自身属性 own property

继承属性inherited property

## 1.3标准ECMAScript内置对象

### 1.3.1全局对象

### 1.3.2 Object对象

### 1.3.3 Function 对象

#### Function的对象属性

toString ( ) 函数转换为字符串

妙用：tostring.call() 可以用来对复杂数据类型的判断

<https://segmentfault.com/a/1190000006774238>

apply (thisArg, argArray) 数组

call (thisArg [ , arg1 [ , arg2, … ] ] ) 参数列表

bind (thisArg [, arg1 [, arg2, …]])

这三者都是改变this指向，第一个参数都是this要指定的对象/想指定的上下文，三者都可以利用后续参数传参

<http://www.cnblogs.com/libin-1/p/6069031.html>

bind 是返回对应函数，便于稍后调用；apply 、call 则是立即调用 。

### 1.3.4 Array 对象

#### Array的对象属性

toString ( )

toString() 返回一个字符串，表示指定的数组及其元素。

将数组中的值返回为字符串并用逗号隔开

toLocaleString ( )

返回一个字符串表示数组中的元素。数组中的元素将使用各自的 toLocaleString 方法转成字符

concat ( [ item1 [ , item2 [ , … ] ] ] )

方法用于合并两个或多个数组。此方法不会更改现有数组，而是返回一个新数组

join (separator)

将数组（或一个类数组对象）的所有元素连接到一个字符串中

a.join()//默认为逗号

a.join(“”)//分隔符===空字符串

a.join(“-”)//分割符为-

pop ( )

方法从数组中删除最后一个元素，并返回该元素的值。此方法更改数组的长度。

当数组为空时返回undefined

push ( [ item1 [ , item2 [ , … ] ] ] )

方法将一个或多个元素添加到数组的末尾，并返回数组的新长度

reverse ( )

将数组中元素的位置颠倒。第一个数组元素成为最后一个数组元素，最后一个数组元素成为第一个。

shift ( )

从数组中删除第一个元素，并返回该元素的值。此方法更改数组的长度。

slice (start, end)

返回一个从开始到结束（不包括结束）选择的数组的一部分浅拷贝到一个新数组，原数组不会被修改即返回一个带有提取元素的新数组

sort (comparefn)

在适当的位置对数组的元素进行排序，并返回数组。 sort 排序不一定是稳定的。默认排序顺序是根据字符串Unicode码点。

numbers.sort(function(a, b) {

return a - b;});//结合这个使用 如果 compareFunction(a, b) 小于 0 ，那么 a 会被排列到 b 之前

splice (start, deleteCount [ , item1 [ , item2 [ , … ] ] ] )

start​

指定修改的开始位置（从0计数）。如果超出了数组的长度，则从数组末尾开始添加内容；如果是负值，则表示从数组末位开始的第几位（从1计数）。

deleteCount 可选

整数，表示要移除的数组元素的个数。如果 deleteCount 是 0，则不移除元素。这种情况下，至少应添加一个新元素。如果 deleteCount 大于start 之后的元素的总数，则从 start 后面的元素都将被删除（含第 start 位）

item1, item2, ... 可选

要添加进数组的元素,从start 位置开始。如果不指定，则 splice() 将只删除数组元素。

unshift ( [ item1 [ , item2 [ , … ] ] ] )

将一个或多个元素添加到数组的开头，并返回新数组的长度。

indexOf ( searchElement [ , fromIndex ] )

返回在数组中可以找到一个给定元素的第一个索引，如果不存在，则返回-1。

searchElement

要查找的元素

fromIndex

开始查找的位置。如果该索引值大于或等于数组长度，意味着不会在数组里查找，返回-1。如果参数中提供的索引值是一个负值，则将其作为数组末尾的一个抵消，即-1表示从最后一个元素开始查找，-2表示从倒数第二个元素开始查找 ，以此类推。 注意：如果参数中提供的索引值是一个负值，仍然从前向后查询数组。如果抵消后的索引值仍小于0，则整个数组都将会被查询。其默认值为0.

lastIndexOf ( searchElement [ , fromIndex ] )

返回指定元素（也即有效的 JavaScript 值或变量）在数组中的最后一个的索引，如果不存在则返回 -1。从数组的后面向前查找，从 fromIndex 处开始。

逆向查找

every ( callbackfn [ , thisArg ] )

测试数组的所有元素是否都通过了指定函数的测试。

callback

用来测试每个元素的函数。

thisArg

执行 callback 时使用的 this 值。

callback 被调用时传入三个参数：元素值，元素的索引，原数组。

some ( callbackfn [ , thisArg ] )

测试数组中的某些元素是否通过由提供的函数实现的测试。

some 为数组中的每一个元素执行一次 callback 函数，直到找到一个使得 callback 返回一个“真值”（即可转换为布尔值 true 的值）。如果找到了这样一个值，some 将会立即返回 true。

forEach ( callbackfn [ , thisArg ] )

对数组的每个元素执行一次提供的函数。

map ( callbackfn [ , thisArg ] )

创建一个新数组，其结果是该数组中的每个元素都调用一个提供的函数后返回的结果。

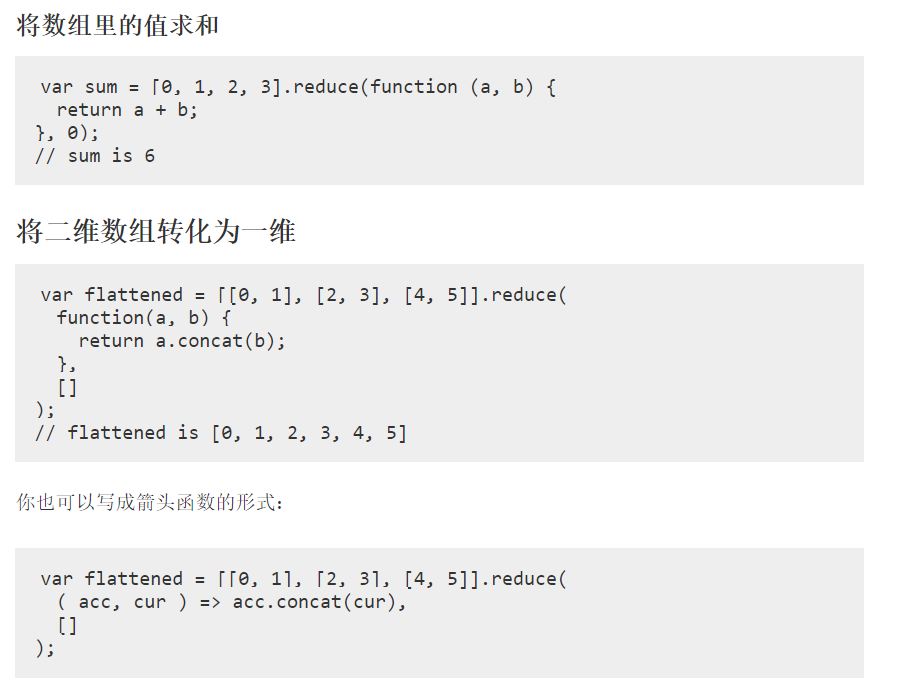
filter ( callbackfn [ , thisArg ] )

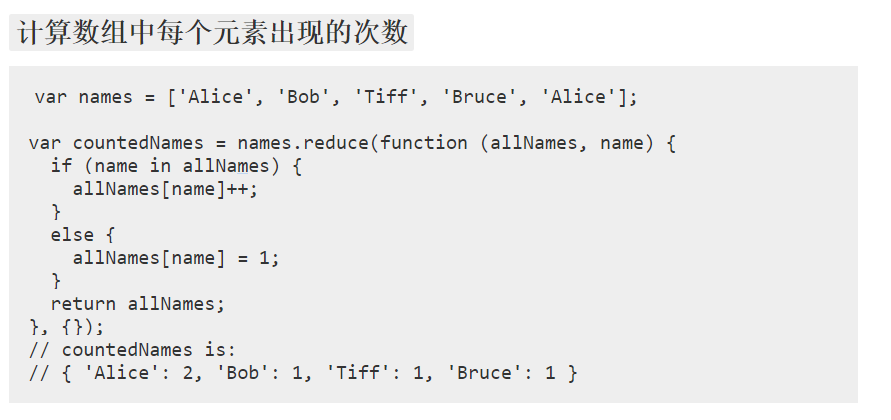
创建一个新数组, 其包含通过所提供函数实现的测试的所有元素。

reduce ( callbackfn [ , initialValue ] )

对累加器和数组中的每个元素 (从左到右)应用一个函数，将其减少为单个值。

语法：array.reduce(function(accumulator, currentValue, currentIndex, array), initialValue)





reduceRight ( callbackfn [ , initialValue ] )

接受一个函数作为累加器（accumulator）和数组的每个值（从右到左）将其减少为单个值。与 Array.prototype.reduce() 的执行方向相反

### 1.3.5 String 对象

#### String的对象属性

toString

对于String对象，toString 方法返回该对象的字符串形式，和 String.prototype.valueOf() 方法返回值一样。

valueOf

返回一个String对象的原始值（primitive value）

charAt

从一个字符串中返回指定的字符。

charCodeAt

0到65535之间的整数，表示给定索引处的UTF-16代码单元

concat ( [ string1 [ , string2 [ , … ] ] ] )

将一个或多个字符串与原字符串连接合并，形成一个新的字符串并返回。

强烈建议使用 赋值操作符（+, +=）代替 concat 方法。性能方面的考虑

indexOf (searchString, position)

返回调用 String 对象中第一次出现的指定值的索引，开始在 fromIndex进行搜索。

区分大小写



lastIndexOf (searchString, position)

localeCompare (that)

stringObject.localeCompare(target);//target要以本地特定的顺序与 stringObject 进行比较的字符串。

match (regexp)

当一个字符串与一个正则表达式匹配时， match()方法检索匹配项。

replace (searchValue, replaceValue)

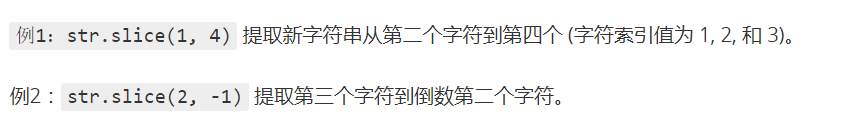
正则使用比较多

search (regexp)

正则

slice (start, end)

提取一个字符串的一部分，并返回一新的字符串。、





split (separator, limit)

指定的分隔符字符串将一个String对象分割成字符串数组

substring (start, end)

一个字符串在开始索引到结束索引之间的一个子集, 或从开始索引直到字符串的末尾的一个子集。

toLowerCase ( )

调用该方法的字符串值转为小写形式

toLocaleLowerCase ( )

任何特定于语言环境的案例映射，返回调用字符串值转换为小写的值。

toUpperCase ( )

toLocaleUpperCase ( )

将输入的字符串转化成大写形式并返回结果字符串。

trim ( )

会从一个字符串的两端删除空白字符。在这个上下文中的空白字符是所有的空白字符 (space, tab, no-break space 等) 以及所有行终止符字符（如 LF，CR）。

### 1.3.6 布尔对象

#### 布尔的对象属性

ToString() toString() 来把一个布尔值转换成字符串

Valueof() valueOf() 方法可返回 Boolean 对象的原始值。

### 1.3.7 Number

#### 数字原型对象的属性

toString 将数字转换为字符串，可接受radix参数进行进制转换

<http://www.cnblogs.com/xiaohuochai/p/5557387.html>

toLocaleString 与上者区别

1.当数字是四位数及以上时

ToString是‘1234’ tolocalestring 是“1,234”

2.当目标是标准时间格式时

sd.toLocaleString()

"2017/2/15 上午11:21:31"

sd.toString()

"Wed Feb 15 2017 11:21:31 GMT+0800 (CST)"

valueOf

valueOf()：返回最适合该对象类型的原始值；

toString(): 将该对象的原始值以字符串形式返回。

这两个方法一般是交由JS去隐式调用，以满足不同的运算情况。

在数值运算里，会优先调用valueOf()，如a + b；

在字符串运算里，会优先调用toString()，如alert(c)。

toFixed

toFixed() 方法可把 Number 四舍五入为指定小数位数的数字。

*number*.toFixed(x) 返回为string类型

toExponential把对象的值转换成指数计数法。

toPrecision

NumberObject.toPrecision(num)

toPrecision() 方法可在对象的值超出指定位数时将其转换为指数计数法。

### 1.3.8 Math对象

#### 1.math对象的值属性

E：自然对数的底数e的数字值

LN10：10的自然对数

LN2: 2的自然对数

LOG2E：ln（2）自然对数以e为底数2为的对数

LN2与LOG2E约互为倒数

PI 圆周率

SQRT1\_2 ½ 的平方根的数字值

SQRT2 2的平方根 与上者互为倒数

#### 2.math对象的函数属性

abs() 绝对值

acox()

asin()

atan()

ceil() 向上取整

floor() 向下取整 Math.floor(x) 的值与 -Math.ceil(-x) 的值相同。

cos() 余弦

max()

min()

pow(x,y) x的y次方

random() 返回一个[0,1）的小树

round() 四舍五入

Math.round(3.5) 返回 4, 但 Math.round(–3.5) 返回 –3.

sqrt(x) x的平方根

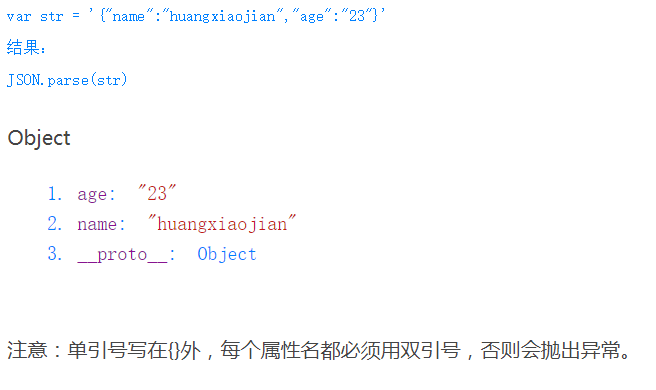
### 1.3.9 Date对象

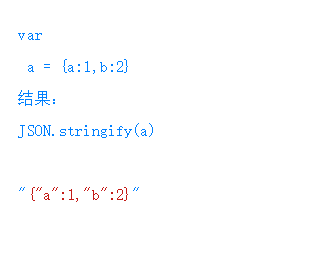
### 1.3.10 RegExp(正则表达式)对象

### 1.3.11 Error Objects

### 1.3.12 JSON对象

#### 1. JSON.parse()和JSON.stringify()

JSON.parse()是从一个字符串中解析出json对象：

stringify()用于从一个对象解析出字符串 

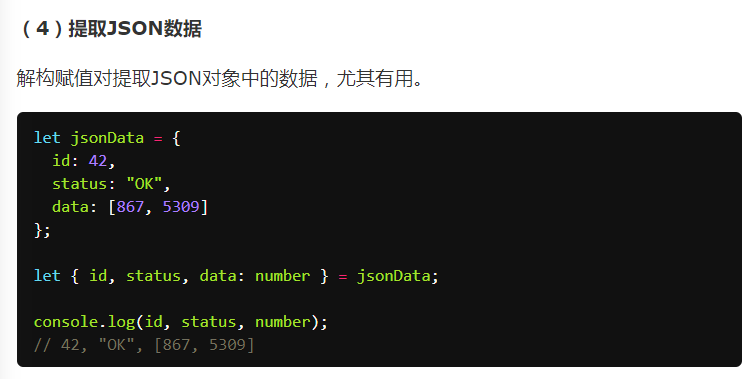
# ES6入门-阮一峰

## 2.1 let和 const命令



## 2.2 变量的解构赋值







## 2.3 字符串的扩展







## 2.4 正则的扩展

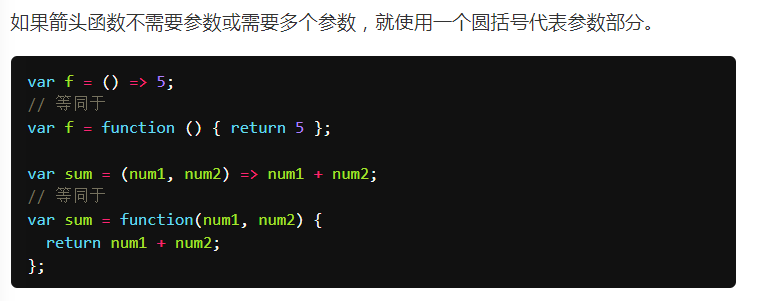
## 2.5 数值的扩展

## 2.6 函数的扩展







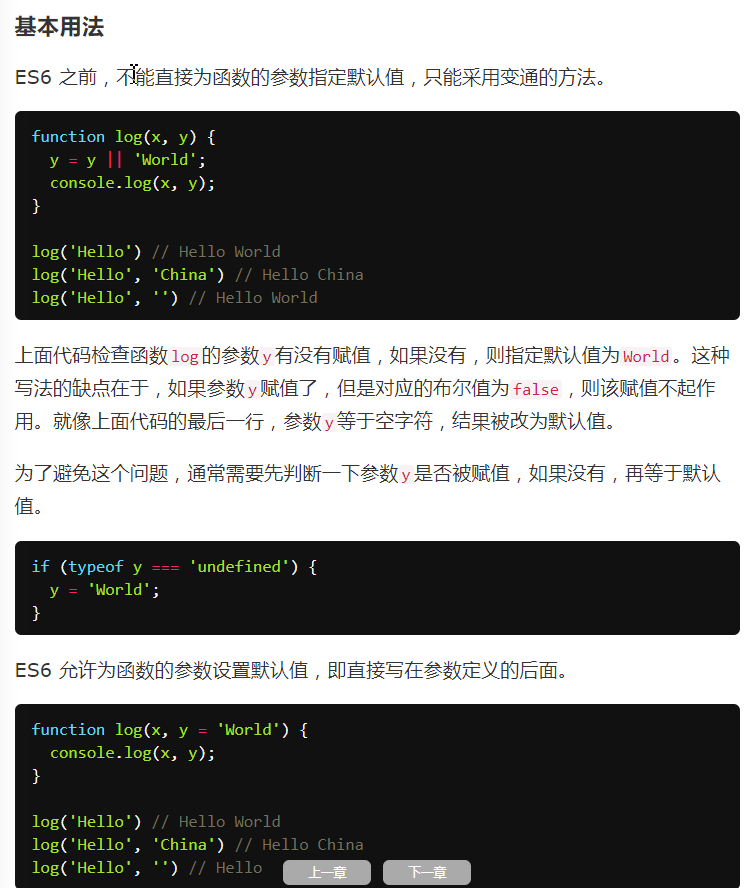




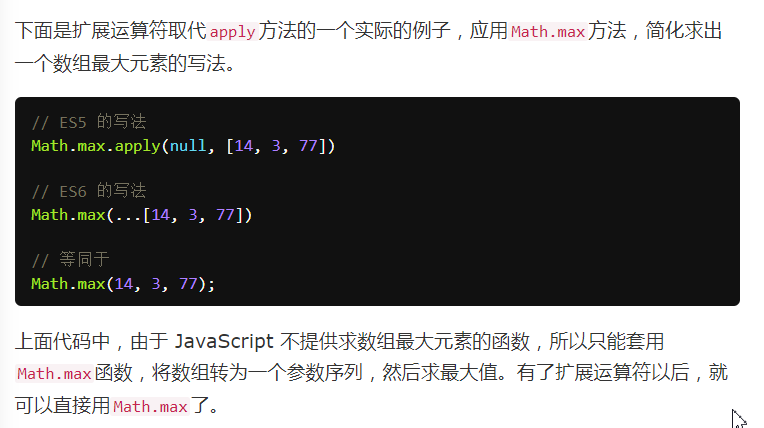


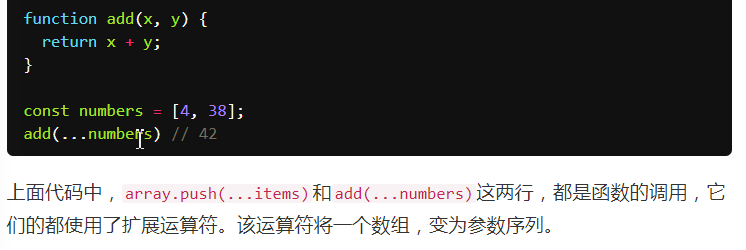


## 2.7 数组的扩展















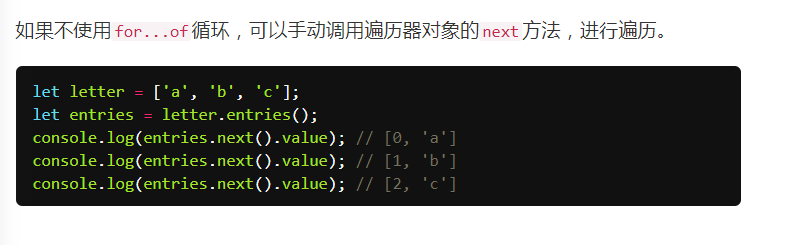






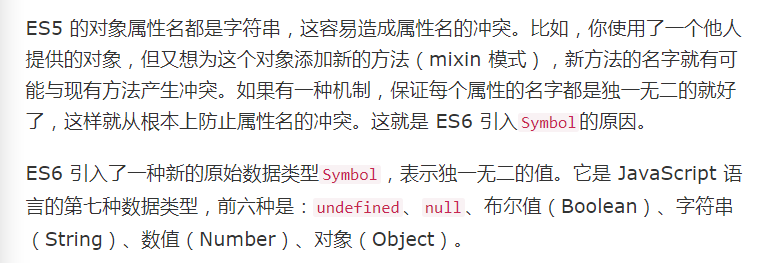


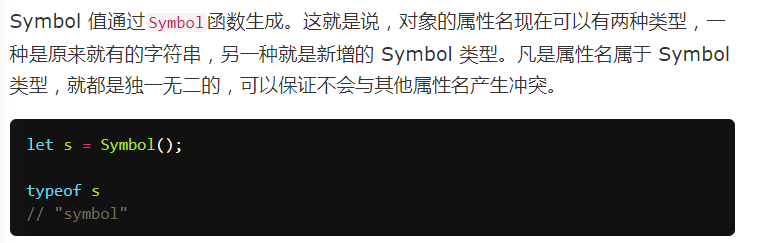




## 2.8 对象的扩展

## 2.9 Symbol





## 2.10 Set 和 Map 数据结构

## 2.11 Proxy

## 2.12 Reflect

## 2.13 Promise 对象

## 2.14 Iterator 和 for...of 循环

## 2.15 Generator 函数的语法

## 2.16 Generator 函数的异步应用

## 2.17 async 函数

## 2.18 Class 的基本语法

## 2.19 Class 的继承

## 2.20 Decorator

## 2.21 Module 的语法

## 2.22 Module 的加载实现