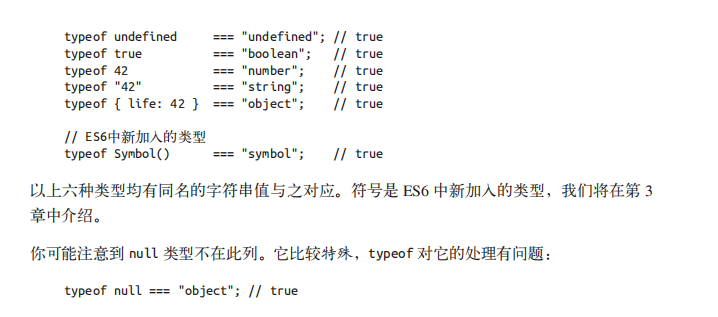
1. **类型**





null 是基本类型中唯一的一个“假值”（falsy 或者 false-like，参见第 4 章）类型，typeof

对它的返回值为 "object"。

function（函数）实际上是 object 的一个“子类型”。

数组也是对象。确切地说，它也是 object 的一个“子类型”。

typeof 运算符总是会返回一个字符串：typeof typeof 42; // "string"

typeof 42 首先返回字符串 "number"，然后 typeof "number" 返回 "string"。

变量在未持有值的时候为 undefined。此时 typeof 返回 "undefined"：

已在作用域中声明但还没有赋值的变量，是 undefifined 的。相反，还没有在作用域中声明

过的变量，是 undeclared（未声明） 的。

1. **值**

使用 delete 运算符可以将单元从数组中删除，但是请注意，单元删除后，数 组的 length 属性并不会发生变化。

字符串不可变是指字符串的成员函数不会改变其原始值，而是创建并返回一个新的字符串。而数组的成员函数都是在其原始值上进行操作。

JavaScript 中的数字常量一般用十进制表示。数字前面的 0 可以省略，小数点后小数部分最后面的0也可以省略。

简单值（即标量基本类型值，scalar primitive）总是通过值复制的方式来赋值 / 传递，包括

null、undefined、字符串、数字、布尔和 ES6 中的 symbol。

复合值（compound value）——对象（包括数组和封装对象，参见第 3 章）和函数，则总

是通过引用复制的方式来赋值 / 传递。

由于引用指向的是值本身而非变量，所以一个引用无法更改另一个引用的指向。

**第四章 强制类型转换**

向 JSON.stringify(..) 传递一个可选参数 replacer，它可以是数组或者函数，用来指定对象序列化过程中哪些属性应该被处理，哪些应该被排除，和 toJSON() 很像。

如果 replacer 是一个数组，那么它必须是一个字符串数组，其中包含序列化要处理的对象的属性名称，除此之外其他的属性则被忽略。

如果 replacer 是一个函数，它会对对象本身调用一次，然后对对象中的每个属性各调用 一次，每次传递两个参数，键和值。如果要忽略某个键就返回 undefined，否则返回指定的值。

JavaScript 中的值可以分为以下两类：

(1) 可以被强制类型转换为 false 的值

(2) 其他（被强制类型转换为 true 的值）

以下这些是假值：

• undefined

• null

• false

• +0、-0 和 NaN

• ""

假值的布尔强制类型转换结果为 false。

假值列表以外的值都是真值。

在 == 中 null 和 undefined 相等（它们也与其自身相等），除此之外其他值都不存在这种情况。