类型

七种内置类型

数字

字符串

对象

Null

Defined

布尔值

Symbol（ES6）

Null是基本类型中唯一的一个“假值”类型，typeof对它的返回值为“object”。

JavaScript中的变量是没有类型的，只有值才有。变量可以随时持有任何类型的值。

字符串与字符数组不一样。

Var a = “foo”;

Var b = [“f”,”o”,”o”];

A[1] = “0”;

B[1] = “0”;

A;//”foo”

B;//[“f”,”0”,”o”]

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。

a.split(“”);//如果a是字符串，可以将a转换为字符数组

a.join(“”);//a是数组，可以将数组的字符拼接回字符串

数字

0.1 + 0.2 === 0.3;//false

特殊数值

Undefined指从未赋值

Null指曾赋过值，但是目前没有值

值和引用

简单值（即标量基本类型值）总是通过值复制的方式来赋值/传递，包括null、undefined、字符串、数字、布尔值和symbol。

复合值——对象（包括数组和封装对象）和函数，则总是通过引用复制的方式来赋值/传递。

由于引用指向的是指本身而非变量，所以一个引用无法更改另一个引用的指向。

Function foo(x){

x.push(4);

x;//[1,2,3,4]

//然后

x = [4,5,6];

x.push(7);

x;//[4,5,6,7]

}

Var a = [1,2,3];

Foo(a);

A;//是[1,2,3,4]，而不是[4,5,6,7]

我们向函数传递a的时候，实际上是将引用a的一个副本赋值给x，而a仍然指向[1,2,3]。在函数中我们可以通过引用x来更改数组的值（push(4)之后变为[1,2,3,4]）。但x = [4,5,6]并不影响a的指向，所以a仍然指向[1,2,3,4]。

我们不能通过引用x来更改引用a的指向，只能更改a和x共同指向的值。

原生函数

New String(“abc”)创建的是字符串”abc”的封装对象，而非基本类型值”abc”。

一般情况下，我们不需要直接使用封装对象。最好的方法是让JavaScript引擎自己决定什么时候应该使用封装对象。