类型：对语言引擎和开发人员来说，类型是值的内部特征，它定义了值得行为，以使区别于其他值。

JS7种内置类型：Null Undefined Boolean Number String Object Symbol(符号ES6新增)，除Object外，其余为基本类型。

typeof返回的是类型的字符串值。例：typeof undefined “undefined”特例：typeof null “object”

function(函数)是JS的一个内置类型，实质是一个object的子类型。函数是一个可调用对象，内部有一个属性[[call]],该属性使其可以以被调用。

JS变量没有类型，只有值才有。

undefined：声明但没有赋值。uncleared：未声明。not defined不等价于undefined，等价于uncleared

DEBUG为“调试模式”开关。var DEBUG=true只在debug.js文件中才有，该文件只在开发和测试时才被加载到浏览器，在生产环节不加载。

delete运算符虽然可以将单元从数组中删除，但是数组的length属性不会改变。

JS中字符串和字符数组不一样。例：var a="foo"; var b={"f","o","o"}

字符串不可变的含义是：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。数组的成员函数都是在其原始值上进行操作。

Array.prototype本身是一个数组，也拥有属性和方法。Array.prototype属性表示Array构造函数的原型，并允许用户向Array添加属性和方法。

JS中的整数：没有小数的十进制数。故：42.0=整数42。

JS中特别大和特别小的数字默认用指数格式显示，与toExponential()函数输出结果相同。

例：var a=5E10; a;//50000000000 a.tooExponential();//"5e+10"

Number各类方法：tofixed():可以指定小数部分显示位数；toPrecision()：可以用来指定有效位数的显示位数。

.运算符首先被识别为数字常量的一部分，其次才是对象属性访问运算符。

10..makeItRain():实现10s金钱雨动画。

不建议使用的格式：0363//243的八进制、0O363。

JS浮点数的“机器精度”（误差范围）一般是2^-52(2.220446049259313e-16),定义在Number.EPSILON中，可以直接使用。

JS最大整数的安全范围为：2^53-1(9007199254740991),定义在Number.MAX\_SAFE\_INTEGER，最小整数为其负数，定义在Number.MIN\_SAFE\_INTEGER。

整数检测：检测是否为整数Number.isInteger(0);//true 检测是否为安全的整数：Number.isSafeInteger(0);//true

null:只空值（曾赋过值，但是目前没有值。）。undefined：指没有值（从未赋值）。

void运算符可以得到undefined值。void并不改变表达式的结果，只是让表达式不返回值。例：var a=42;console.log(void a,a);//undefined 42

NaN不是一个有效数值/坏数值，但是为数字类型。可以理解它为一个警戒值：用于指出数字类型中的错误情况（即执行数学运算没有成功，这是失败返回后的结果。）

NaN!=NaN，可以利用Number.isNaN()来检验一个数值是否为NaN。或利用它的非自反特性来检验。

判断两个值是否相等：Object.is(a,b)

引用和指针不一样，引用只能指向值。

类型转换发生在静态类型语言的编译阶段，强制类型转换发生在动态类型语言运行时。隐式类型转换通常是某些操作产生的副作用。

强制类型转换：ToString():转换为字符串，JSON.stringify(null);//"null" JSON字符串话，序列的结果总是字符串。

不安全的JSON值：undefined、function、symbol、包含循环引用的对象。

JSON.stringify([1,undefined,function(){},4]);//"[1,null,null,4]"。

如果对象中定义了toJSON()方法，JSON字符串化时会首先调用该方法，然后用它的返回值来进行序列化。

toJSON()返回一个能够被字符串化的安全的JSON值。

JSON.stringify()传递一个可选参数replacer,该参数可以时数组或函数，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。

var a={b:42,c:"42",d:[1,2,3]};JSON.stringify(a,["b","c"]);//"{"b":42,"c":"42"}"

如果replacer时函数，它的参数K在第一次调用时为undefined（就是对对象本身调用的那次）。JSON.stringfy()并不是强制类型转换。

ToNumber():true转换为1，false转换为0，undefined转换为NaN，对以0开头的十六进制数并不按十六进制处理。

Object.create(null)创建的对象[[Prototype]]属性为null，没有valueOf()和toString()方法。

valueOf() 方法可返回 Boolean 对象的原始值。如果调用该方法的对象不是Boolean，则抛出异常TypeError。

转换为布尔类型为false的是：undefined、null、+0、-0、NAN。

“假值对象”：浏览器在某些特定的情况下，在常规JS语法基础上自己创建了一些外来值，这些外来值就是“假值对象”，转换为布尔类型为false。

日期显示转换为数字 ：var d=new Date("Mon,18 Aug 2014 08:53:06 CDT");+d;//1408369986000。 var timestamp=+new Date();//通常用该方法获得当前的时间戳。

不建议对日期类型使用强制类型转换，应该使用Date.now()来获得当前的时间戳，new Date().getTime获得指定时间的时间戳。

解析允许字符串中含有非数字字符，解析按从左到右的顺序，如果遇到非数字字符就停止。而转换不允许出现非数字字符，否则会失败并返回 NaN 。

ES6 允许从符号到字符串的显式强制类型转换，然而隐式强制类型转换会产生错误。

符号不能够被强制类型转换为数字（显式和隐式都会产生错误），但可以被强制类型转换为布尔值（显式和隐式结果都是 true ）。

对于 || 来说，如果条件判断结果为 true 就返回第一个操作数的值，如果为false 就返回第二个操作数的值。

&& 则相反，如果条件判断结果为 true 就返回第二个操作数的值，如果为 false 就返回第一个操作数的值。