

DOM是宿主对象，是语言中可更换的部分。

ECMA是核心语言，是不可被替代的功能。

浏览器——客户端JavaScript运行的宿主环境。

**ECMAScript的版本：**

ES5(2009年12月发布) ES6（2015年6月发布） ES7（2016年6月发布） ES8（2017年6月发布） ES9（2018年6月发布） ES10（2019年6月发布） 当前网络上大部分用的是ES5

**JavaScript语言特点：**

1. 直译式脚本语言：在宿主（浏览器、Node）中解释执行，非编译语言
2. 弱类型、动态类型语言：可以动态更改变量类型（动态类型）
3. ES5没有块作用域，基于原型链的继承方式。

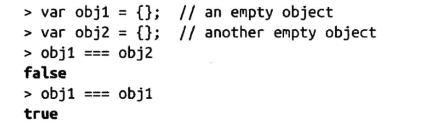
JavaScript除了在浏览器中运行，还可以在其他的运行环境中运行，如node.js环境。目前JavaScript的运行环境有浏览器和node.js环境两种。

**JavaScript在浏览器中是如何运行的？**

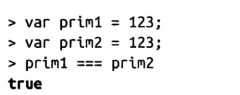
浏览器下载JavaScript脚本文件后，由浏览器JavaScript引擎解释执行。

JavaScript中，原始值：布尔值、数字、字符串、null、underfined

其他值都是对象。这两者区别主要在于它们的比较方式；每个对象都有唯一的标识且只（严格的）等于自己。



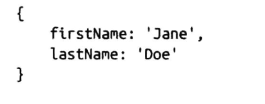
所有的原始值，只要编码值相同，则被认为相等。



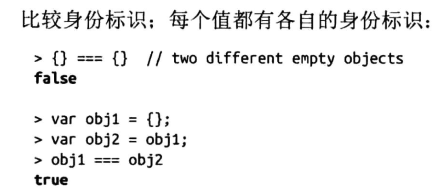
所有的原始值：布尔值，数字，字符串，两个‘空值’（underfiined，null）

特点：按值进行比较，不可改变（属性不能被改变，添加或移除）

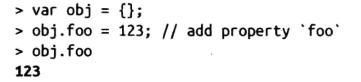
所有的非原始值都是对象：简单对象



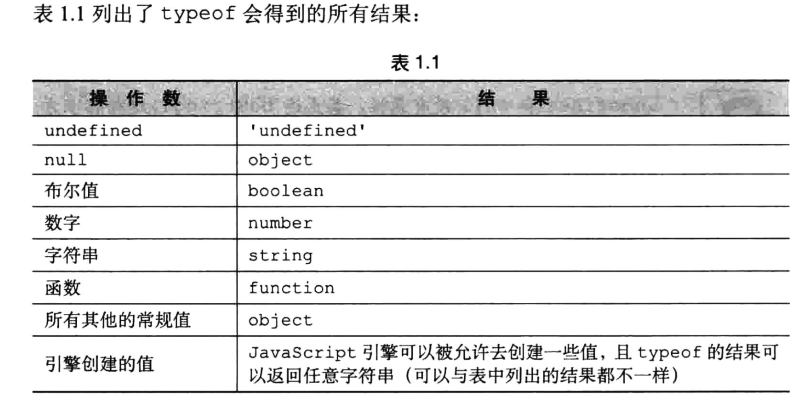
数组，正则表达式

特点：按引用进行比较 

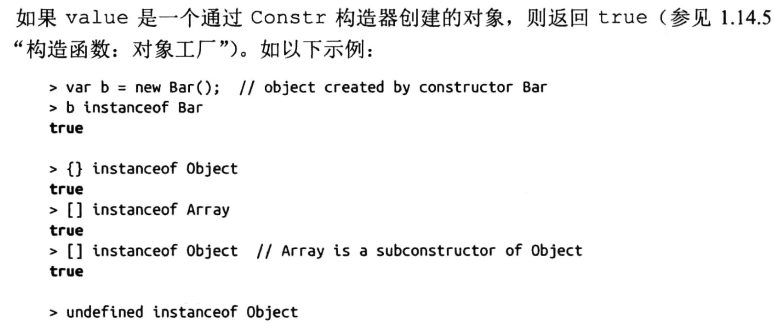
默认可变：对象属性可以很自由的地被改变、添加和移除



有两种对值分类的操作符：typeof主要用于原始值，instanceof主要用于对象

‘’

Instanceof用法：value instanceof Constr



以下的值会被解释称false：

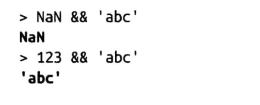
1.Underfined,null 2.布尔值：false 3.数字：-0，NaN

4.字符串：‘ ’

其他所有值（包括所有的对象）都会被当成true

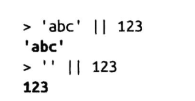
1. 与（&&）

如果第一个运算数是假值，返回他。否则，返回第二个运算数



（2）或（||）

如果第一个运算数是真值，返回他。否则，返回第二个运算数



1.JS中所有的数字都是浮点数，也包含一些特殊的数字

2.像所有原始值一样，字符串是不可改变的，如果要改变一个已有的字符串，必须创建一个新的才行

3.do-while 循环语句，由于条件跟在代码体之后，所以这些代码体至少会执行一次

4.break可以跳离循环。Continue会开始一个新的迭代循环

5.arguments看起来像个数组，但却不具备数组的方法。

6.要切换到严格模式，在js文件或者<script>标签第一行输入“use strict”

7.以函数作为值的属性被称为方法，使用in运算符检查属性是否存在

delete运算符移除属性

**第二章**

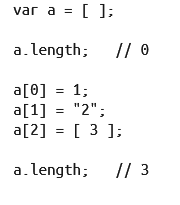
（一）1. JS有7中内置类型:(1)空值 (2)未定义 (3)布尔值 (4)数字 (5)字符串 (6)对象 (7)符号

2. typeof对null的返回值为object。函数对象的length属性是其声明的参数的个数。

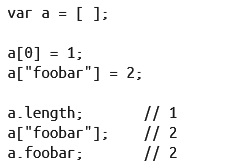
3. 变量没有类型，但他们持有的值有类型。Undefined是值的一种，undeclared表示变量还没有被声明过。

（二）

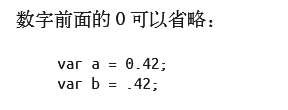
1. 对数组声明后即可向其中加入值，不需要预先设定大小



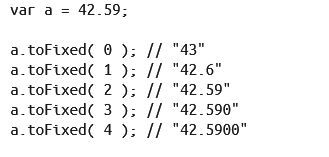
2.数组通过数字进行索引，但他们也是对象，所以也可以包含字符串键值和属性，但这些并不计算在数组长度内。



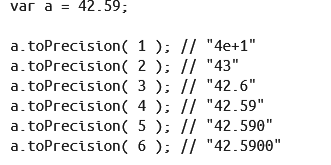
1. JS中的数字常量一般用十进制表示



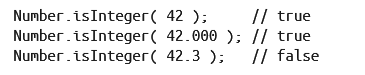
1. 由于数字值可以使用Number对象进行封装，因此数字值可以调用Number.prototype中的方法。Tofixed( )方法可以指定小数部分的显示位数



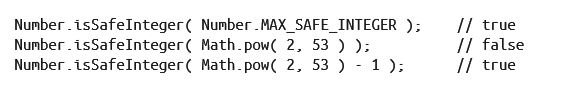
toPrecision()方法用来指定有效位数的显示位数



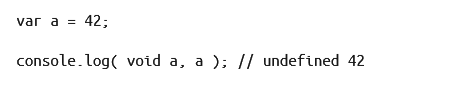
1. 检测一个值是否是整数，用Number.isInteger( )方法



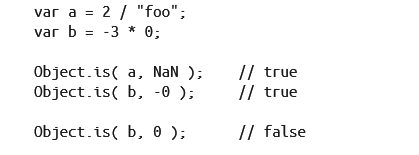
检测一个值是否为安全的整数，用Number.isSafeInteger( )方法



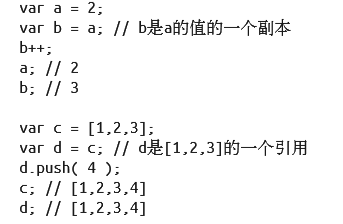
1. null指空值，从未赋过值，undefined指没有值，即曾经赋过值，但是目前没有值。null是一个特殊关键字，不是标志符，不能当做变量来使用和赋值，undefined是一个标志符，可以被当作变量来使用和赋值，void—没有返回值，所以返回结果是undefined，void并不改变表达式的结果，只是让表达式不返回值



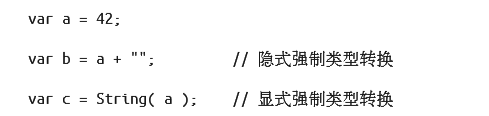
1. NaN != NaN(true) JSON.stringify(-0)返回“0”，JSON.parse(“-0”)返回-0
2. ES6新加入了一个工具方法Object.is( )来判断两个值是否绝对相等

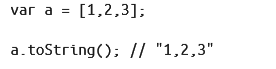


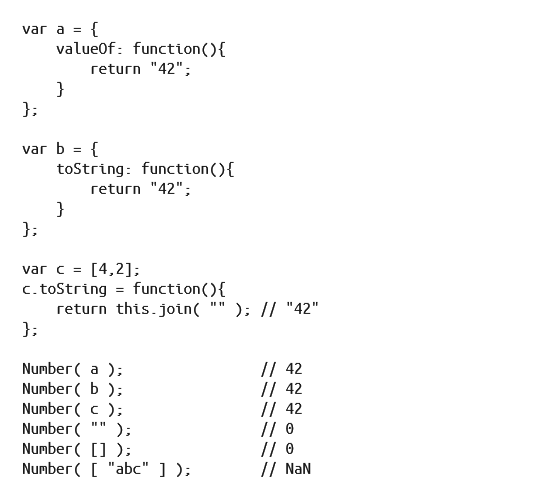
1. JS中引用指向的是值。如果一个值有10个引用，这些引用指向的都是同一个值，它们相互之间没有引用/指向关系。



（四）





1. JSON.stringify(“42”) //””42””
2. 
3. String()将值转换为字符串基本类型，Number()将值转换为数字基本类型
4. 获得当前时间戳：var timetamp=+new Date( );

将日期对象转换为时间戳: var timetamp = Date.now( ); //获得当前时间戳

显示方法: var timetamp = new Date( ).getTime( ); //获得指定时间的时间戳

1. | 运算符

0 | -0； //0 0 | Infinity; //0

0 | NaN; //0 0 | -Infinity; //0

1. indexOf( ):在字符串中搜索指定的子字符串，如果找到就返回子字符串所在的位置，否则返回-1.
2. var a=”42”;

var b=”42px”;

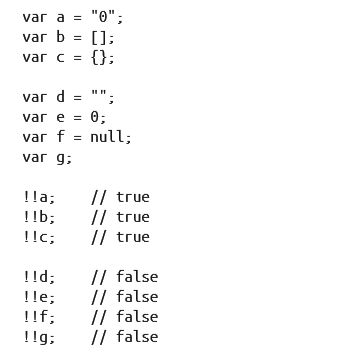
Number(a); //42

parseInt(a); //42

Number(b); //NaN

parseInt(b); //42

1. 显示强制类型转换为布尔值最常用的方法是!!,第二个！会将结果反转回原值。



1. 如果其中有且仅有一个参数为true，则onlyOne( )返回true

var a=true;

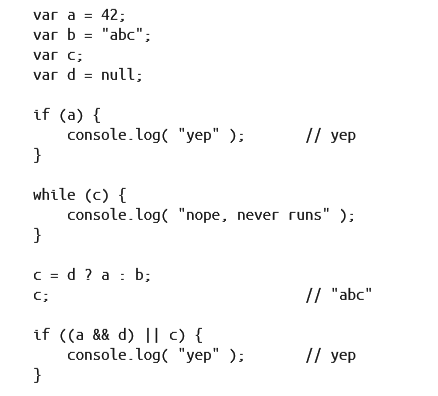
var b=false;

onlyOne(a,b,b); //true

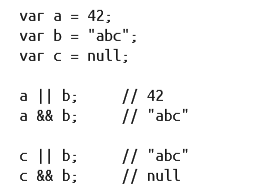
onlyOne(b,a,b); //true

onlyOne(a,b,a); //false

1. 以下情况，会发生非布尔值隐士强制类型转换为布尔值
2. If() 语句中的条件判断表达式
3. for( )语句中的条件判断表达式
4. while( )和 do…while( )循环中的条件判断表达式
5. ? :中的条件判断表达式
6. 逻辑运算符 || 和 && 左边的操作数（作为条件判断表达式）



1. || &&他们的返回值是两个操作数中的一个（且仅一个）



先对第一个操作数进行判断，如果不是布尔值，进行ToBoolean（）强制类型转换，然后再执行条件判断。

|| ：如果条件判断为true返回第一个操作数，false返回第二个操作数的值

&&： 判断结果为true，返回第二个操作数的值，false返回第一个操作数的值

1. 如果Type（x）是布尔类型，返回ToNumber（x）==y的结果

如果Type（y）是布尔类型，返回x== ToNumber（y）的结果

var x=true;

var y=”42”; x==y; //false

ToNumber(x)将true转换为1，变成1==“42”，根据规则转换为1==42，结果false

1. 如果Type（x）是字符串或数字，Type（y）是对象，x==Toprimitive（y）的结果

如果Type（x）是对象，Type（y）是字符串或数字，返回x==Toprimitive（y）的结果

