数据类型和运算符

从入门到工作：JS 全解

版权声明

联系方式

如果你想要购买本课程
请微信联系 xiedaimala02 或 xiedaimala03

如果你发现有人盗用本课程
请微信联系 xiedaimala02 或 xiedaimala03

数据为什么需要类型

新人始终不明白

数字与字符串

- 都是一，为什么要分 1 和 '1'
- 功能不同
  - ✓ 数字是数字，字符串是字符串，要严谨
  - ✓ 数字能加减乘除，字符串不行
  - ✓ 字符串能表示电话号码，数字不行
- 存储形式不同
  - ✓ JS 中，数字是用 64 位浮点数的形式存储的
  - ✓ JS 中，字符串是用类似 UTF8 形式存储的（UCS-2）

如何存数字

十进制转二进制即可

二进制

- 10 转 2
  - ✓ 31 变成二进制 $31 = ? \times 2^5 + ? \times 2^4 + ? \times 2^3 + ? \times 2^2 + ? \times 2^1 + ? \times 2^0$
  - ✓ 经过一番尝试 $31 = 0 \times 2^5 + 1 \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$
  - ✓ 所以 31(十进制) = 01111(二进制)
  - ✓ 不是套公式吗？程序员从来不套公式 ☺
- 2 转 10
  - ✓ 100011 变成十进制
  - ✓ 每一位乘以 2 的 N 次方，然后加起来即可
  - ✓ 100011 = 2^5 + 2^1 + 2^0 = 35

用十六进制表示二进制

- 为什么用十六进制
  - 因为二进制写起来太慢了：0111100010110 10
  - 记住 8 4 2 1 对应 X X X X
  - 从右往左每四位改写成一位： 0111100010110 10
  - 得到 3,12,5,10；把大于9的数字改为ABCDEF
  - 于是得到 3C5A，你也可以用计算器的程序员模式
  - HEX 表示 16 进制，BIN 表示 2 进制
  - OCT 表示 8 进制，DEC 表示 10 进制

如何存字符

转成数字不就得了

但是注意，'1' 不能用 1 来表示

啥是 UTF8

说来话长

如何存储ａｂｃ１２３

简单，编号表示

用 0~127 表示所有符号

Dec	Char	Dec	Char	Dec	Char	Dec	Char
0	NUL (null)	32	SPACE	64	@	96	`
1	SOH (start of heading)	33	!	65	A	97	a
2	STX (start of text)	34	"	66	B	98	b
3	ETX (end of text)	35	#	67	C	99	c
4	EOT (end of transmission)	36	$	68	D	100	d
5	ENQ (enquiry)	37	%	69	E	101	e
6	ACK (acknowledge)	38	&	70	F	102	f
7	BEL (bell)	39	'	71	G	103	g
8	BS (backspace)	40	(	72	H	104	h
9	TAB (horizontal tab)	41	)	73	I	105	i
10	LF (NL line feed, new line)	42	*	74	J	106	j
11	VT (vertical tab)	43	+	75	K	107	k
12	FF (NP form feed, new page)	44	,	76	L	108	l
13	CR (carriage return)	45	-	77	M	109	m
14	SO (shift out)	46	.	78	N	110	n
15	SI (shift in)	47	/	79	O	111	o
16	DLE (data link escape)	48	0	80	P	112	p
17	DC1 (device control 1)	49	1	81	Q	113	q
18	DC2 (device control 2)	50	2	82	R	114	r
19	DC3 (device control 3)	51	3	83	S	115	s
20	DC4 (device control 4)	52	4	84	T	116	t
21	NAK (negative acknowledge)	53	5	85	U	117	u
22	SYN (synchronous idle)	54	6	86	V	118	v
23	ETB (end of trans. block)	55	7	87	W	119	w
24	CAN (cancel)	56	8	88	X	120	x
25	EM (end of medium)	57	9	89	Y	121	y
26	SUB (substitute)	58	:	90	Z	122	z
27	ESC (escape)	59	;	91	[	123	{
28	FS (file separator)	60	<	92	\	124	|
29	GS (group separator)	61	=	93	]	125	}
30	RS (record separator)	62	>	94	^	126	~
31	US (unit separator)	63	?	95	_	127	DEL

0 表示结束字符
10 表示换行
13 表示回车
32 表示空格
33到47表示标点
48到57表示数字符号
65到90表示大写字母
97到122表示小写字母
127表示删除键

中国人开始用电脑了

怎么表示中文呢

简单，还是编号

中国国家标准局来编，名称为「国标2312」

用 0000~FFFF 表示汉字

code	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
C4A0		摹	蘑	模	膜	磨	摩	魔	抹	末	莫	墨	默	沫	漠	寞
C4B0	陌	谋	牟	某	拇	牡	亩	姆	母	墓	暮	幕	募	慕	木	目
C4C0	睦	牧	穆	拿	哪	呐	钠	那	娜	纳	氖	乃	奶	耐	奈	南
C4D0	男	难	囊	挠	脑	恼	闹	淖	呢	馁	内	嫩	能	妮	霓	倪
C4E0	泥	尼	拟	你	匿	腻	逆	溺	蔫	拈	年	碾	撵	捻	念	娘
C4F0	酿	鸟	尿	捏	聂	孽	啮	镊	镍	涅	您	柠	狞	凝	宁	

code	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
C5A0		拧	泞	牛	扭	钮	纽	脓	浓	农	弄	奴	努	怒	女	暖
C5B0	虐	疟	挪	懦	糯	诺	哦	欧	鸥	殴	藕	呕	偶	沤	啪	趴
C5C0	爬	帕	怕	琶	拍	排	牌	徘	湃	派	攀	潘	盘	磐	盼	畔
C5D0	判	叛	乓	庞	旁	耪	胖	抛	咆	刨	炮	袍	跑	泡	呸	胚
C5E0	培	裴	赔	陪	配	佩	沛	喷	盆	砰	抨	烹	澎	彭	蓬	棚
C5F0	硼	篷	膨	朋	鹏	捧	碰	坯	砒	霹	批	披	劈	琵	毗	

一个16进制数是4个0/1位
FFFF就是4x4=16位，也就是两个字节
最多收录 2^16 = 65536 个字符
但只收录了 6000 多汉字、西文字母和日文假名

「你」的 GB2312 编号为 C4E3
「牛」的 GB2312 编号为 C5A3

李瓗 => 李*

陶喆 => 陶吉吉

中国人的名字里有生僻字！

怎么表示生僻字、繁体字、韩文呢，之前的忘了编进去

微软出手了

微软推出了一个国标扩展，简称GBK

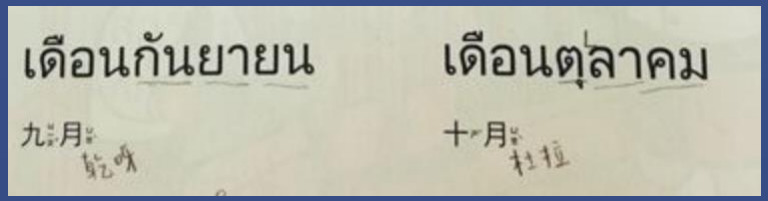
GBK 国标扩

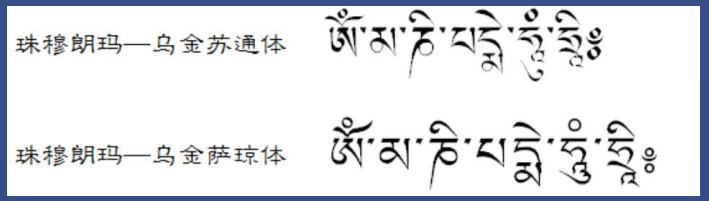
例	字	GBK	GB2312
朱镕基	镕	√	×
西塑站	塑	√	×
王濛	濛	√	×
歡度國慶	歡度國慶	√	×
佢啲喺嗰尐	佢啲喺嗰尐	√	×
啰	啰	√	×
刘銻堃	銻堃	√	×
卍	卍	√	×
张琳芃	芃	√	×
咁样	咁	√	×
拔山蓋吉	蓋吉	√	×
菓汁	菓	√	×
步厅	步	√	×
咀香園	咀、園	√	×
焚	焚	√	×

含21886个汉字和图形符号
收录了中日韩使用的几乎所有汉字
完全兼容 GB2312

依然使用 16 位（两字节）

后来国标局推出 GB18030 取代 GBK
GB18030 不兼容 GB2312





网页里有藏文、泰文

怎么办……显示不了

简单，继续编号

这回，一次解决全世界的需求

万国码

Unicode

Unicode

- 优点
  - 已收录 13 万字符（大于 16 位），全世界通用
  - 以后还会继续扩充，不会停止
  - 最新版只添加了一个字——令和的合体字
- 缺点
  - 两个字节不够用，每个字符要用三个及以上字节
  - 这样所有文件都扩大 50%，不划算
  - 那怎么办？

虽然用 Unicode
但存的时候偷懒

这样行不行

UTF-8 就被发明出来了

还真行

鸡贼的存法

- 存储「a」
  - a 对应的 Unicode 编号为 97，十六进制为 61
  - Unicode 直接存：00000000000000000110000 1
  - UTF-8偷懒存法：01100001
  - 三字节变一字节，比GBK 还省
- 存储「你」
  - 你对应的 Unicode 编号为 4F60
  - Unicode 直接存：00000000010011110110000 0
  - UTF-8偷懒存法：111001001011110110100000
  - 还是三字节，没有省，但是字母都能省一点
- UTF-8 中的 8 的意思是
  - 最少可用 8 位存一个字符

UTF-8 的规则

- 以「你a」为例
  - 111001001011110110100000001100001
  - 如何知道上述内容表示什么字符?
  - 读 8 位信息 11100100
  - 发现开头有 3 个 1，说明这个字符有 3 个八位
  - 于是再读两个 8 位信息 10111101 10100000
  - 前面的 10 不要，其他合起来，得 0100111101100000
  - 这就还原为 Unicode 的你了：
  - 00000000010011101100000
  - 再读 8 为信息 01100001
  - 发现开头是 0，说明这个字符只占 8 位
  - 这就还原味 Unicode 的 a 了：
  - 00000000000000001100001
- 这一页看不懂就跳过，反正不考
  - 记住去餐馆排队的比喻即可

我们终于搞清如何存字符了

那就是编号，然后存编号

你还好意思问
数字1和字符1的区别吗

功能不同、存储形式不同

JS 中的数据类型

- 7种（大小写无所谓）
  - ✓ 数字 number
  - ✓ 字符串 string
  - ✓ 布尔 bool
  - ✓ 符号 symbol
  - ✓ 空 undefined
  - ✓ 空 null
  - ✓ 对象 object
  - ✓ 总结：四基两空一对象
- 以下不是数据类型
  - ✓ 数组、函数、日期
  - ✓ 它们都属于 object

数字 number

64位浮点数

写法

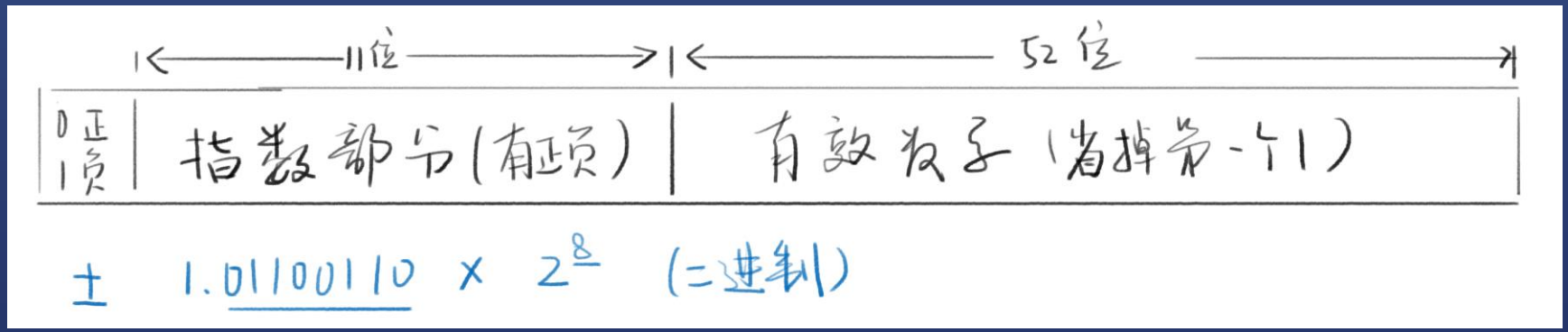
- 整数写法
  - 1
- 小数写法
  - 0.1
- 科学计数法
  - 1.23e4
- 八进制写法（用得少）
  - 0123 或 00123 或 0o123
- 十六进制写法
  - 0x3F 或 0X3F
- 二进制写法
  - 0b11 或 0B11

特殊值

- 正0 和 负0
  - 都等于 0，要严谨
- 无穷大 1/0=infinite,1/+0=infinite,1/-0=-infinite
  - Infinity、+Infinity、-Infinity
- 无法表示的数字
  - NaN (Not a Number)
  - 但它是一个数字（讲一下历史）

64位浮点数

- JS数字的存储形式
  - 浮点就是浮动的点，意思就是小数点会乱动
  - 123.456 可以表示为 1.23456e10^2
  - 也可以表示为 12345.6e10^-2



- 64位存储一个 number
  - 符号占 1 位
  - 指数占 11 位（-1023~1024）
  - 有效数字占 52 位（开头的 1 省略）

范围和精度

- 范围（忽略符号位）
  - 指数拉满、有效数字拉满，得到最大二进制数字
  - Number.MAX_VALUE: 1.7976931348623157e+308
  - 指数负方向拉满、有效数字最小1，得到最小值
  - Number.MIN_VALUE: 5e-324
- 精度（有效数字）
  - 最多只能到52+1个二进制位表示有效数字
  - 2^53 对应的十进制是 9 后面 15 个零
  - 所以15位有效数字都能精确表示
  - 16位有效数字如果小于 90 开头，也能精确表示
  - 9110000000000001 就存不下来

字符串 string

每个字符两个字节（阉割版 UTF8）

写法

- 单引号
  - '你好'
- 双引号
  - "你好"
- 反引号
  - `你好`
- 注意
  - 引号不属于字符串的一部分，就像书名号不属于书名的一部分一样
  - 如果要在单引号里面包含单引号怎么办?

转义

- 错误写法
  - 'it's ok'
  - JS 引擎会认为 'it' 就结束了，后面的看不懂
- 正确写法
  - 'it\'s ok' // 这就是转义
  - "it's ok"
  - `it's ok`

转义

- 用另一种写法表示你想要的东西
  - \' 表示 '
  - \" 表示 "
  - \n 表示换行
  - \r 表示回车
  - \t 表示 tab 制表符
  - \\ 表示 \
  - \uFFFF 表示对应的 Unicode 字符
  - \xFF 表示前 256 个 Unicode 字符

多行字符串

- 如果你想要在字符串里回车

let s = `这样是
可以的
用反引号很容易做到`

- 以前没有反引号的时候
  - 写起来很麻烦，可以看网道教程

字符串的属性

等等，对象才有属性，为什么字符串也有属性

为什么字符串有属性

等学完对象才能解答

字符串的长度

- string.length
  - ✓ `'123'.length // 3`
  - ✓ `'\n\r\t'.length // ?` 3
  - ✓ `''.length // 0`
  - ✓ `' '.length // 1`

通过下标读取字符

- string[index]
  - let s = 'hello';
  - s[0] // "h"
- 注意 index 从 0 开始
  - s[0] 是第一个字符
- 注意 index 到 length
  - let s = 'hello';
  - s[5] // undefined，居然不报错
  - s[4] // 'o'

base64 转码

- window.btoa window.btoa('123')
  - ✓ 正常字符串转为 Base64 编码的字符串
- window.atob
  - ✓ Base64 编码的字符串转为原来的字符串
- 一般用来隐藏招聘启事里的简历
  - ✓ 邮箱：ZmFuZ3lpbmdoYW5nQGZveG1haWwuY29t
- 有时候也用来自欺欺人
  - ✓ 所谓的「加密」，也就能骗过一部分外行

布尔 boolean

真或假

只有两个值

true 和 false，注意大小写

下列运算符会得到 bool 值

- 否定运算
  - !value
- 相等运算
  - 1 == 2、1 != 2、3 === 4、3 !== 4
- 比较运算
  - 1 > 2、1 >= 2、3 < 4、3 <= 4

if 配 bool

- if 语句常常需要判断真假
  - if(value) { ... } else { ... }
- 问题来了
  - 如果 value 是 bool 值还好说
  - 如果 value 不是 bool 值咋办，谁真谁假
  - 1 是真还是假，0 是真还是假
  - '1' 是真还是假，'0' 是真还是假

五个 falsy 值

falsy 就是相当于 false 但又不是 false 的值

分别是 `undefined null 0 NaN ''`

两个空，两个数字，一个字符串

其它都是真值，包括数组，函数，对象等

'' 和 ' ' 不是一个玩意

再次声明，请保持严谨

undefined 和 null
两种空类型

空空如也

为什么有两个空

这是 JS 的原(la)创(ji)之处

区别

- 没有本质区别
- 细节一
  - ✓ 如果一个变量声明了，但没有赋值，那么默认值就是 undefined，而不是 null
- 细节二
  - ✓ 如果一个函数，没有写 return，那么默认 return undefined，而不是 null
- 细节三
  - ✓ 前端程序员习惯上，把非对象的空值写为 undefined，把对象的空值写为 null
  - ✓ 但仅仅是习惯上而已

symbol 符号

不怎么常用的数据类型

直接看文章吧

我写的文章

变量声明

- 三种声明方式
  - var a = 1
  - let a = 1
  - const a = 1
  - a = 1 是赋值。如果没有在其它地方声明a，则a挂在window上，可以认为是全局变量。
- 区别
  - var 是过时的、不好用的方式
  - let 是新的，更合理的方式
  - const 是声明时必须赋值，且不能再改的方式
  - 最后这种方式是错误的，不准这样声明
- var 变量提升
  - 押题时再讲，有兴趣可以提前看网道教程

var 声明

- 直接跳过
  - ✓ 我们写代码不用 var
  - ✓ 面试押题前单独讲解

let 声明

```
{
  let b=1
}
console.log(b)
//报错
```

```
let a=1
let a=2
//报错
```

```
let a=1
{let a=2}
//可以
```

- 规则
  - ✓ 遵循块作用域，即使用范围不能超出 { }
  - ✓ 不能重复申明
  - ✓ 可以赋值，也可以不赋值
  - ✓ 必须先声明再使用，否则报错
  - ✓ 全局声明的 let 变量，不会变成 window 的属性 var会
  - ✓ for 循环配合 let 有奇效

```
let a=1
window.a
//undefined
```

const 声明

- 规则
  - ✓ 跟 let 几乎一样
  - ✓ 只有一条不一样：声明时就要赋值，赋值后不能改

变量声明

- 指定值
  - var a = 1
- 同时也指定了类型
  - var a = 1
- 但是值和类型都可以随意变化
  - a = 2
  - a = '字符串'

name 和 'name' 的区别

新人想不通

name 和 'name' 的区别

- name 是变量
  - ✓ 值可变，可能是 'name'，也可能是 'hello'
- 'name' 是字符串常量
  - ✓ 常量就是不变量
  - ✓ 'name' 只能是 'name'，不能是其他值

总结

- 六种类型（大小写无所谓）
  - undefined
  - null
  - number
  - string
  - bool
  - symbol
- 这些都是简单类型
  - 只有 object 叫做复杂类型，下节课学

类型转换

- number => string
  - ✓ String(n)
  - ✓ n + ''

n=1
n+''
//"1"

- string => number
  - ✓ Number(s)
  - ✓ parseInt(s) / parseFloat(s)
  - ✓ s - 0 减法会把字符串转成数字再减。或者+s。

就算显示整数也是小数，因为只有64浮点数这一个数字。

- x => bool
  - ✓ Boolean(x)
  - ✓ !!x 一个感叹号是取反，两个感叹号是取反再取反，所以是取原始布尔值。

- x => string
  - ✓ String(x)
  - ✓ x.toString()

注：1.toString()会出bug。要(1).toString。还可以1..toString，因为1.表示1.0

再见

这次的作业还是博客，养成写博客的好习惯哦