JavaScript 内置对象

目录 Contents

1. 内置对象

- JavaScript 中的对象分为3种：自定义对象 、内置对象、 浏览器对象
- 前面两种对象是JS 基础 内容，属于 ECMAScript； 第三个浏览器对象属于我们JS 独有的， 我们JS API 讲解
- 内置对象就是指 JS 语言自带的一些对象，这些对象供开发者使用，并提供了一些常用的或是最基本而必要的功能（属性和方法）
- 内置对象最大的优点就是帮助我们快速开发
- JavaScript 提供了多个内置对象：`Math`、 `Date` 、`Array`、`String`等

目录 Contents

2. 查文档

2.1 MDN

学习一个内置对象的使用，只要学会其常用成员的使用即可，我们可以通过查文档学习，可以通过MDN/W3C来查询。

Mozilla 开发者网络（MDN）提供了有关开放网络技术（Open Web）的信息，包括 HTML、CSS 和万维网及 HTML5 应用的 API。

MDN: https://developer.mozilla.org/zh-CN/

2. 查文档

2.2 如何学习对象中的方法

1. 查阅该方法的功能
2. 查看里面参数的意义和类型
3. 查看返回值的意义和类型
4. 通过 demo 进行测试

目录 Contents

3. Math 对象

3.1 Math 概述

Math 对象不是构造函数，它具有数学常数和函数的属性和方法。跟数学相关的运算（求绝对值，取整、最大值等）可以使用 Math 中的成员。

```
Math.PI           // 圆周率
Math.floor()      // 向下取整
Math.ceil()             // 向上取整
Math.round()            // 四舍五入版 就近取整    注意 -3.5    结果是  -3
Math.abs()                 // 绝对值
Math.max()/Math.min()      // 求最大和最小值
```

注意：上面的方法必须带括号

3. Math 对象

案例：封装自己的数学对象

利用对象封装自己的数学对象 里面有 PI 最大值和最小值

3. Math 对象

3.2 随机数方法 random()

random() 方法可以随机返回一个小数，其取值范围是 [0，1)，左闭右开 0 <= x < 1

得到一个两数之间的随机整数，包括两个数在内

```
function getRandom(min, max) {
  return Math.floor(Math.random() * (max - min + 1)) + min;
}
```

3. Math 对象

案例：猜数字游戏

程序随机生成一个 1~ 10 之间的数字，并让用户输入一个数字，

1. 如果大于该数字，就提示，数字大了，继续猜；
2. 如果小于该数字，就提示数字小了，继续猜；
3. 如果等于该数字，就提示猜对了， 结束程序。

3. Math 对象

案例分析

① 随机生成一个1~10 的整数 我们需要用到 Math.random() 方法。
② 需要一直猜到正确为止，所以一直循环。
③ 用while 循环合适更简单。
④ 核心算法：使用 if else if 多分支语句来判断大于、小于、等于。

目录 Contents

4. 日期对象

4.1 Date 概述

- Date 对象和 Math 对象不一样，他是一个构造函数，所以我们需要实例化后才能使用
- Date 实例用来处理日期和时间

4. 日期对象

4.2 Date()方法的使用

1. 获取当前时间必须实例化

```
var now = new Date();
console.log(now);
```

2. Date() 构造函数的参数

如果括号里面有时间，就返回参数里面的时间。例如日期格式字符串为'2019-5-1'，可以写成new Date('2019-5-1') 或者 new Date('2019/5/1')

- 如果Date()不写参数，就返回当前时间
- 如果Date()里面写参数，就返回括号里面输入的时间

4. 日期对象

4.3 日期格式化

我们想要 2019-8-8 8:8:8 格式的日期，要怎么办?

需要获取日期指定的部分，所以我们要手动的得到这种格式。

方法名	说明	代码
getFullYear()	获取当年	dObj.getFullYear()
getMonth()	获取当月（0-11）	dObj.getMonth()
getDate()	获取当天日期	dObj.getDate()
getDay()	获取星期几 （周日0 到周六6）	dObj.getDay()
getHours()	获取当前小时	dObj.getHours()
getMinutes()	获取当前分钟	dObj.getMinutes()
getSeconds()	获取当前秒钟	dObj.getSeconds()

4. 日期对象

案例： 输出当前日期

请写出这个格式的日期：2019年8月8日 星期四

4. 日期对象

案例： 输出当前时间

写一个函数，格式化日期对象，成为 HH:mm:ss 的形式 比如 00:10:45

4. 日期对象

4.4 获取日期的总的毫秒形式

Date 对象是基于1970年1月1日（世界标准时间）起的毫秒数

为什么计算机起始时间从1970年开始?

我们经常利用总的毫秒数来计算时间，因为它更精确

```
// 实例化Date对象
var now = new Date();
// 1. 用于获取对象的原始值
console.log(date.valueOf())
console.log(date.getTime())
// 2. 简单写可以这么做
var now = + new Date();
// 3. HTML5中提供的方法，有兼容性问题
var now = Date.now();
```

4. 日期对象

案例：倒计时效果

做一个倒计时效果

4. 日期对象

案例分析

① 核心算法：输入的时间减去现在的时间就是剩余的时间，即倒计时 ，但是不能拿着时分秒相减，比如 05 分减去25分，结果会是负数的。

② 用时间戳来做。用户输入时间总的毫秒数减去现在时间的总的毫秒数，得到的就是剩余时间的毫秒数。

③ 把剩余时间总的毫秒数转换为天、时、分、秒 （时间戳转换为时分秒）

转换公式如下：

- d = parseInt(总秒数/ 60/60 /24); // 计算天数
- h = parseInt(总秒数/ 60/60 %24) // 计算小时
- m = parseInt(总秒数 /60 %60); // 计算分数
- s = parseInt(总秒数%60); // 计算当前秒数

目录 Contents

5. 数组对象

5.1 数组对象的创建

创建数组对象的两种方式

- 字面量方式
- new Array()

5. 数组对象

5.2 检测是否为数组

- instanceof 运算符，可以判断一个对象是否属于某种类型
- Array.isArray()用于判断一个对象是否为数组，isArray() 是 HTML5 中提供的方法

```
var arr = [1, 23];
var obj = {};
console.log(arr instanceof Array); // true
console.log(obj instanceof Array); // false
console.log(Array.isArray(arr));   // true
console.log(Array.isArray(obj));   // false
```

5. 数组对象

5.3 添加删除数组元素的方法

方法名	说明	返回值
push(参数1....)	末尾添加一个或多个元素，注意修改原数组	并返回新的长度
pop()	删除数组最后一个元素，把数组长度减 1 无参数、修改原数组	返回它删除的元素的值
unshift(参数1...)	向数组的开头添加一个或更多元素，注意修改原数组	并返回新的长度
shift()	删除数组的第一个元素，数组长度减 1 无参数、修改原数组	并返回第一个元素的值

5. 数组对象

案例： 筛选数组

有一个包含工资的数组[1500, 1200, 2000, 2100, 1800]，要求把数组中工资超过2000的删除，剩余的放到新数组里面

```
var arr = [1500, 1200, 2000, 2100, 1800];
var newArr = [];
for (var i = 0; i < arr.length; i++) {
    if (arr[i] < 2000) {
        newArr.push(arr[i]);
    }
}
console.log(newArr);
```

5. 数组对象

5.4 数组排序

方法名	说明	是否修改原数组
reverse()	颠倒数组中元素的顺序,无参数	该方法会改变原来的数组　返回新数组
sort()	对数组的元素进行排序	该方法会改变原来的数组 返回新数组

```
var arr = [1, 64, 9, 6];
arr.sort(function(a, b) {
    return b - a;       // 降a序
    // return a - b;    // 升序
});
console.log(arr);
```

5. 数组对象

5.5 数组索引方法

方法名	说明	返回值
indexOf()	数组中查找给定元素的第一个索引	如果存在返回索引号 如果不存在，则返回-1。
lastIndexOf()	在数组中的最后一个的索引，	如果存在返回索引号 如果不存在，则返回-1。

5. 数组对象

案例： 数组去重（重点案例）

有一个数组['c', 'a', 'z', 'a', 'x', 'a', 'x', 'c', 'b']，要求去除数组中重复的元素。

5. 数组对象

案例分析

① 目标：把旧数组里面不重复的元素选取出来放到新数组中，重复的元素只保留一个，放到新数组中去重。

② 核心算法：我们遍历旧数组，然后拿着旧数组元素去查询新数组，如果该元素在新数组里面没有出现过，我们就添加，否则不添加。

③ 我们怎么知道该元素没有存在？ 利用 新数组.indexOf(数组元素) 如果返回时 -1 就说明 新数组里面没有改元素

旧数组['c', 'a', 'z', 'a', 'x', 'a', 'x', 'c', 'b']

新数组 []

5. 数组对象

5.6 数组转换为字符串

方法名	说明	返回值
toString()	把数组转换成字符串，逗号分隔每一项	返回一个字符串
join('分隔符')	方法用于把数组中的所有元素转换为一个字符串。	返回一个字符串

5. 数组对象

5.7 课下查询

方法名	说明	返回值
concat()	连接两个或多个数组 不影响原数组	返回一个新的数组
slice()	数组截取slice(begin, end)	返回被截取项目的新数组
splice()	数组删除splice(第几个开始,要删除个数)	返回被删除项目的新数组 注意，这个会影响原数组

slice() 和 splice() 目的基本相同，建议同学们重点看下 splice()

目录 Contents

6. 字符串对象

6.1 基本包装类型

为了方便操作基本数据类型，JavaScript 还提供了三个特殊的引用类型：String、Number和 Boolean。

基本包装类型就是把简单数据类型包装成为复杂数据类型，这样基本数据类型就有了属性和方法。

```
// 下面代码有什么问题?
var str = 'andy';
console.log(str.length);
```

按道理基本数据类型是没有属性和方法的，而对象才有属性和方法，但上面代码却可以执行，这是因为 js 会把基本数据类型包装为复杂数据类型，其执行过程如下：

```
// 1. 生成临时变量，把简单类型包装为复杂数据类型
var temp = new String('andy');
// 2. 赋值给我们声明的字符变量
str = temp;
// 3. 销毁临时变量
temp = null;
```

6. 字符串对象

6.2 字符串的不可变

指的是里面的值不可变，虽然看上去可以改变内容，但其实是地址变了，内存中新开辟了一个内存空间。

```
var str = 'abc';
str = 'hello';
// 当重新给 str 赋值的时候，常量'abc'不会被修改，依然在内存中
// 重新给字符串赋值，会重新在内存中开辟空间，这个特点就是字符串的不可变
// 由于字符串的不可变，在大量拼接字符串的时候会有效率问题
var str = '';
for (var i = 0; i < 100000; i++) {
    str += i;
}
console.log(str); // 这个结果需要花费大量时间来显示，因为需要不断的开辟新的空间
```

6. 字符串对象

6.3 根据字符返回位置

字符串所有的方法，都不会修改字符串本身(字符串是不可变的)，操作完成会返回一个新的字符串。

方法名	说明
indexOf('要查找的字符', 开始的位置)	返回指定内容在元字符串中的位置， 如果找不到就返回 -1，开始的位置是 index 索引号
lastIndexOf()	从后往前找，只找第一个匹配的

6. 字符串对象

案例：返回字符位置

查找字符串"abcoefoxyozzopp"中所有o出现的位置以及次数

6. 字符串对象

案例：思路

查找字符串"abcoefoxyozzopp"中所有o出现的位置以及次数

① 核心算法：先查找第一个o出现的位置
② 然后 只要indexOf 返回的结果不是 -1 就继续往后查找
③ 因为indexOf 只能查找到第一个，所以后面的查找，利用第二个参数，当前索引加1，从而继续查找

6. 字符串对象

6.4 根据位置返回字符（重点）

方法名	说明	使用
charAt(index)	返回指定位置的字符(index 字符串的索引号)	str.charAt(0)
charCodeAt(index)	获取指定位置处字符的ASCII码 (index索引号)	str.charCodeAt(0)
str[index]	获取指定位置处字符	HTML5，IE8+支持 和charAt()等效

6. 字符串对象 String

6.4 根据位置返回字符（重点）

ASCII表

（ American Standard Code for Information Interchange 美国标准信息交换代码 ）

高四位 / 低四位		ASCII控制字符 0000 0						ASCII控制字符 0001 1						ASCII打印字符 0010 2		ASCII打印字符 0011 3		ASCII打印字符 0100 4		ASCII打印字符 0101 5		ASCII打印字符 0110 6		ASCII打印字符 0111 7		
		十进制	字符	Ctrl	代码	转义字符	字符解释	十进制	字符	Ctrl	代码	转义字符	字符解释	十进制	字符	十进制	字符	十进制	字符	十进制	字符	十进制	字符	十进制	字符	Ctrl
0000	0	0		^@	NUL	\0	空字符	16	▶	^P	DLE		数据链路转义	32		48	0	64	@	80	P	96	`	112	p	
0001	1	1	☺	^A	SOH		标题开始	17	◀	^Q	DC1		设备控制 1	33	!	49	1	65	A	81	Q	97	a	113	q	
0010	2	2	☻	^B	STX		正文开始	18	↕	^R	DC2		设备控制 2	34	"	50	2	66	B	82	R	98	b	114	r	
0011	3	3	♥	^C	ETX		正文结束	19	‼	^S	DC3		设备控制 3	35	#	51	3	67	C	83	S	99	c	115	s	
0100	4	4	♦	^D	EOT		传输结束	20	¶	^T	DC4		设备控制 4	36	$	52	4	68	D	84	T	100	d	116	t	
0101	5	5	♣	^E	ENQ		查询	21	§	^U	NAK		否定应答	37	%	53	5	69	E	85	U	101	e	117	u	
0110	6	6	♠	^F	ACK		肯定应答	22	▬	^V	SYN		同步空闲	38	&	54	6	70	F	86	V	102	f	118	v	
0111	7	7	•	^G	BEL	\a	响铃	23	↨	^W	ETB		传输块结束	39	'	55	7	71	G	87	W	103	g	119	w	
1000	8	8	◘	^H	BS	\b	退格	24	↑	^X	CAN		取消	40	(	56	8	72	H	88	X	104	h	120	x	
1001	9	9	○	^I	HT	\t	横向制表	25	↓	^Y	EM		介质结束	41	)	57	9	73	I	89	Y	105	i	121	y	
1010	A	10	◙	^J	LF	\n	换行	26	→	^Z	SUB		替代	42	*	58	:	74	J	90	Z	106	j	122	z	
1011	B	11	♂	^K	VT	\v	纵向制表	27	←	^[	ESC	\e	溢出	43	+	59	;	75	K	91	[	107	k	123	{	
1100	C	12	♀	^L	FF	\f	换页	28	∟	^\	FS		文件分隔符	44	,	60	<	76	L	92	\	108	l	124	|	
1101	D	13	♪	^M	CR	\r	回车	29	↔	^]	GS		组分隔符	45	-	61	=	77	M	93	]	109	m	125	}	
1110	E	14	♫	^N	SO		移出	30	▲	^^	RS		记录分隔符	46	.	62	>	78	N	94	^	110	n	126	~	
1111	F	15	☼	^O	SI		移入	31	▼	^-	US		单元分隔符	47	/	63	?	79	O	95	_	111	o	127	⌂	^Backspace 代码：DEL

6. 字符串对象

案例：返回字符位置

判断一个字符串 'abcoefoxyozzopp' 中出现次数最多的字符，并统计其次数。

6. 字符串对象

案例：核心算法

判断一个字符串 'abcoefoxyozzopp' 中出现次数最多的字符，并统计其次数。

① 核心算法：利用 charAt() 遍历这个字符串
② 把每个字符都存储给对象， 如果对象没有该属性，就为1，如果存在了就 +1
③ 遍历对象，得到最大值和该字符

判断一个字符串 'abcoefoxyozzopp' 中出现次数最多的字符，并统计其次数。

对象 o { }

6. 字符串对象

6.5 字符串操作方法（重点）

方法名	说明
concat(str1,str2,str3...)	concat() 方法用于连接两个或多个字符串。拼接字符串，等效于+，+更常用
substr(start,length)	从start位置开始（索引号），length 取的个数 重点记住这个
slice(start, end)	从start位置开始，截取到end位置，end取不到 (他们俩都是索引号)
substring(start, end)	从start位置开始，截取到end位置，end取不到 基本和slice 相同 但是不接受负值

6. 字符串对象

6.6 replace()方法

replace() 方法用于在字符串中用一些字符替换另一些字符。

其使用格式如下：

```
replace(被替换的字符串， 要替换为的字符串);
```

6. 字符串对象

6.7 split()方法

split()方法用于切分字符串，它可以将字符串切分为数组。在切分完毕之后，返回的是一个新数组。

例如下面代码：

```
var str = 'a,b,c,d';
console.log(str.split(','));    // 返回的是一个数组 [a, b, c, d]
```

6. 字符串对象

6.8 课下查阅

- toUpperCase()　　//转换大写
- toLowerCase()　　//转换小写

作业

给定一个字符串，如：“abaasdffggghhjjkkgfddssss3444343”，问题如下：

1、 字符串的长度

2、 取出指定位置的字符，如：0,3,5,9等

3、 查找指定字符是否在以上字符串中存在，如：i，c，b等

4、 替换指定的字符，如：g替换为22,ss替换为b等操作方法

5、 截取指定开始位置到结束位置的字符串，如：取得1-5的字符串

6、 找出以上字符串中出现次数最多的字符和出现的次数

7、 遍历字符串，并将遍历出的字符两头添加符号“@”

传智播客旗下高端IT教育品牌