Javascript06_对象

对象是自包含的数据集合，包含在对象里的数据可以通过两种形式访问-属性和方法。

属性：隶属于某个特定对象的变量

方法：只有某个特定对象才能调用的函数

对象：属性与方法结合在一起的数据实体，在JavaScript中访问对象的属性和方法都用.

```
Person.mood;
Person.age;
Person.walk();
Person.sleep();
```

实例：为了使用某个对象来描述一个特定的个体，我们需要创建一个对象的实例。实例是对象的具体个体。使用new关键字创建。

```
var tina=new Person;
```

分类

JavaScript中对象分为三类：

用户自定义对象：由程序员自行创建的对象。

内建对象：内建在JavaScript语言里的对象，如Array，Math和Date等。

宿主对象：由浏览器提供的对象。常见的 文档对象document，浏览器窗口对象window

内建对象

JavaScript中已经预先定义好的对象，我们可以直接使用。数组就是其中一个。我们定义数组对象可以使用下面的方式

```
var beatles=new Array();
```

有了这个对象，我们就可以使用系统为这个对象已经设置好的属性和方法了。比如，我们要知道这个数组中的元素个数，可以使用

```
beatles.length
```

这个属性。Array只是一个例子，我们内建的对象还有很多，例如，Math对象，Date对象。事实上我们之前学习的数据类型，每个数据类型是对应类型的对象。

String对象

String对象用来处理字符串问题

属性

- length　字符串长度

方法

- indexOf("子字符串")
  返回子字符串在整个字符串中第一次出现的位置
  　位置从0开始
  　不存在返回-1
- lastIndexOf("子字符串")
  返回子字符串在整个字符串中最后一次出现的位置
- charAt(位置)
  返回指定位置的字符
  　如果没有提供索引，将使用0
- charCodeAt(位置)
  返回的是指定位置上字符对应的ASCII码

ASCII 字符代码表 一

高四位 / 低四位		ASCII非打印控制字符 0000 0					ASCII非打印控制字符 0001 1					ASCII 打印字符 0010 2		0011 3		0100 4		0101 5		0110 6		0111 7		
		十进制	字符	ctrl	代码	字符解释	十进制	字符	ctrl	代码	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	BLANK NULL	^@	NUL	空	16	▶	^P	DLE	数据链路转意	32		48	0	64	@	80	P	96	`	112	p	
0001	1	1	☺	^A	SOH	头标开始	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	反确认	37	%	53	5	69	E	85	U	101	e	117	u	
0110	6	6	♠	^F	ACK	确认	22	■	^V	SYN	同步空闲	38	&	54	6	70	F	86	V	102	f	118	v	
0111	7	7	•	^G	BEL	震铃	23	↨	^W	ETB	传输块结束	39	'	55	7	71	G	87	W	103	g	119	w	
1000	8	8	◘	^H	BS	退格	24	↑	^X	CAN	取消	40	(	56	8	72	H	88	X	104	h	120	x	
1001	9	9	○	^I	TAB	水平制表符	25	↓	^Y	EM	媒体结束	41	)	57	9	73	I	89	Y	105	i	121	y	
1010	A	10	◙	^J	LF	换行/新行	26	→	^Z	SUB	替换	42	*	58	:	74	J	90	Z	106	j	122	z	
1011	B	11	♂	^K	VT	竖直制表符	27	←	^[	ESC	转意	43	+	59	;	75	K	91	[	107	k	123	{	
1100	C	12	♀	^L	FF	换页/新页	28	∟	^\	FS	文件分隔符	44	,	60	<	76	L	92	\	108	l	124	|	
1101	D	13	♪	^M	CR	回车	29	↔	^]	GS	组分隔符	45	-	61	=	77	M	93	]	109	m	125	}	
1110	E	14	♫	^N	SO	移出	30	▲	^6	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	Δ	^Back space

注：表中的ASCII字符可以用：ALT + "小键盘上的数字键" 输入

> b 比特
> B 字节　1B = 8b
> 1KB = 1024B
> 1MB = 1024K
> 1GB = 1024M
> 1T = 1024G

- substr(起始位置,[截取长度])
  截取长度不写则代表截取到字符串末尾
- substring(起始位置,结束位置)
  不包括截取结果的右边界字符
- replace("子串1","子串2")
  将子串1替换为子串2
    默认只替换找到的第一个子串
    如果想替换所有需要使用正则表达式来实现

```
var str = "My name is apple. I like to eat apple!";
console.log( str.replace( "apple" , "banana") );
console.log( str.replace(/apple/g,"banana") );
```

- split(分割符,[数量])
  将字符串分割为数组
    数量不设置默认为不限制分割的数量

Math对象

Math对象用来处理数学问题

属性

- PI：表示一个圆的周长与直径的比例，约为 3.14159

方法

- Math.max(x,y)　　最大值
- Math.min(x,y)　　最小值
- Math.pow(x,y)　　xy
- Math.round(number)　四舍五入
- Math.ceil(number)　　向上取整
- Math.floor(number)　　向下取整
- Math.random()　　返回0.0~1.0之间的随机数

注意：生成的随机数包含0，但不包含1

> x到y范围的随机数：Math.random() * (y - x + 1) + x

```
//随机数
console.log(Math.random());
//向下取整
console.log(Math.floor(9.999));
//向上取整
console.log(Math.ceil(1.001));
```

```
//四舍五入
console.log(Math.round(1.4), Math.round(1.5));
//绝对值
console.log(Math.abs(-5), Math.abs(5));
// 次方
console.log(Math.pow(2, 5), Math.pow(27, 1/3));
// 开方
console.log(Math.sqrt(81));

// 求最大值
console.log(Math.max(7, 3, 100, 25, 123));
// 求最小值
console.log(Math.min(7, 3, 100, 25, 123));
```

Date对象

Date JS里的与日期和时间有关的对象,我们可以创建一个日期时间对象. 通过该对象来操作与日期或者时间有关的方法

创建Date对象

- var 日期对象 = new Date(参数);
  new Date("2017/5/1 12:08:00");
  创建指定日期对象
  注意日期格式
- new Date()
  不传参数则创建当前日期对象

注意: 通过Date创建出的对象表示的是一个时刻, 是一个时间点!

方法

- getFullYear()　　年
- getMonth()　　月份 0-11
- getDate()　　日
- getHours()　　小时
- getMinutes()　　分钟
- getSeconds()　　秒
- getDay()　　星期 0~6
- getTime()　　1970 年 1 月 1 日至今的毫秒数

注意: 通过Date创建出的对象表示的是一个时刻, 是一个时间点!

```
var dateObj = new Date();
console.log(typeof dateObj);
console.log(dateObj);
```

```
object                                                    04_Date.html:16
Sat Mar 27 2021 09:42:55 GMT+0800 (中国标准时间)          04_Date.html:17
```

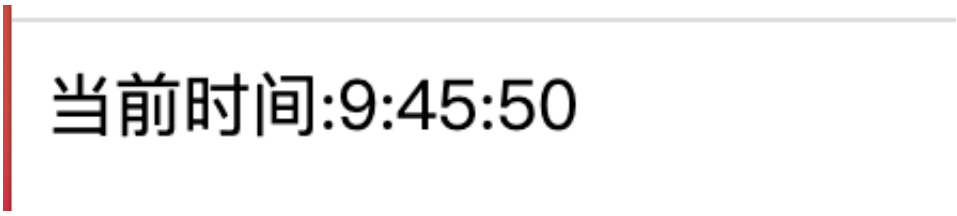
- 通过Date获取具体的日期和时间信息

```
// 1. 获取年份
console.log(dateObj.getFullYear());
// 2. 获取月份  取值范围0-11
console.log(dateObj.getMonth());
// 3. 获取周几 周日-周六 0-6
console.log(dateObj.getDay());
// 4. 获取日
console.log(dateObj.getDate());
// 5. 获取小时
console.log(dateObj.getHours());
// 6. 获取分钟
console.log(dateObj.getMinutes());
// 7. 获取秒
console.log(dateObj.getSeconds());
```

```
Show console sidebar                                    04_Date.html:20
  2                                                     04_Date.html:22
  6                                                     04_Date.html:24
  27                                                    04_Date.html:26
  9                                                     04_Date.html:28
  44                                                    04_Date.html:30
  29                                                    04_Date.html:32
```

- 练习: 通过Date对象实现电子表效果

```
setInterval(function(){
    var dateObj = new Date();
    document.body.innerHTML = "当前时间:" + dateObj.getHours()+ ":" +
dateObj.getMinutes() + ":" + dateObj.getSeconds();
}, 1000);
```



- 自定义日期时间对象
  - 在创建对象的同时传入各个数据

```
var futureDate = new Date(2021,5,1,18,0,0);
console.log(futureDate);
```

```
Tue Jun 01 2021 18:00:00 GMT+0800 (中国标准时间)      04_Date.html:41
```

  - 在创建对象的同时传入时间戳(1970年1月1日到任意一个时刻的毫秒数)

```
var futureDate = new Date(1637402400000);
console.log(futureDate);
```

```
Sat Nov 20 2021 18:00:00 GMT+0800 (中国标准时间)                    04_Date.html:44
```

宿主对象

不是JavaScript语言预定义的对象，而是运行JavaScript的运行环境提供的对象，比如我们在web开发中，是由浏览器提供的预定义对象叫做宿主对象。包括Form，Image，Element等。我们可以通过它们获得表单，图片或元素信息。还有一个更强大的对象document对象，他可以获取来自页面上的任何一个元素信息。

window对象的重要方法- 计时器

- setInterval(函数体, 毫秒数);
  setInterval() 方法可按照指定的周期（以毫秒计）来调用函数或计算表达式。
  setInterval() 方法会不停地调用函数，直到 clearInterval() 被调用或窗口被关闭。由 setInterval() 返回的 ID 值可用作 clearInterval() 方法的参数。
- setTimeout(函数体, 毫秒数);

  setTimeout() 方法可指定的时间后（以毫秒计）来调用函数或计算表达式。
  clearTimeout() 取消定时。由 setTimeout() 返回的 ID 值可用作 clearTimeout() 方法的参数。

html代码部分

```
<img src="img/1.jpg" id="pic" />
<button id="previousBtn">上一张</button>
<button id="nextBtn">下一张</button>
<button id="startBtn">开始</button>
<button id="pauseBtn">暂停</button>
```

Javascript代码部分

```
<script type="text/javascript">
    // 声明一个变量记录当前是第几张图片
    var index = 1;
    // 上一张
    previousBtn.onclick = function() {
        // 判断是否到达最后一张
        if(index == 1) {
            // 将其重置为第一张
            index = 8;
        } else {
            // 没有到达最后一张, 图片自增
            index--;
        }
        // 给图片赋值src
```

```
        pic.src = "img/" + index + ".jpg";
    }

    // 下一张
    nextBtn.onclick = nextClick;
    function nextClick() {
        // 判断是否到达最后一张
        if(index == 8) {
            // 将其重置为第一张
            index = 1;
        } else {
            // 没有到达最后一张, 图片自增
            index++;
        }
        // 给图片赋值src
        pic.src = "img/" + index + ".jpg";
    }

    // 开始按钮
    var isSet = true;
    var num = 0;
    startBtn.onclick = function() {
        /*
         * 计时器   setInterval(函数体, 毫秒数);
         * 该方法执行之后, 系统会每隔固定的时间自动调用对应的函数, 达到自动执行代码的效果
         *
         * 返回值: 计时器的序号
         *
         */
        if(isSet){
            num = setInterval(nextClick, 1000);
            isSet = false;
        }
    }

    // 暂停按钮
    pauseBtn.onclick = function() {
        /*
         * 清除计时器  clearInterval(要清除的计时器的序号)
         */
        clearInterval(num);
        isSet = true;
    }
</script>
```

> 注意:
>
> 1. setInterval()里的第一个值可以填写一个匿名函数, 也可以直接填写一个函数名
> 2. 计时器是另一个线程在控制, 不影响主线程代码的执行, 可以一次在系统里添加多个计时器

this关键字

面向对象语言中 this 表示当前对象的一个引用。

但在 JavaScript 中 this 不是固定不变的，它会随着执行环境的改变而改变。

- 在方法中，this 表示该方法所属的对象。
- 如果单独使用，this 表示全局对象window。
- 在自定义函数中，this 表示全局对象window。
- 在函数中，在严格模式下，this 是未定义的(undefined)。
- 在事件中，this 表示接收事件的元素。
- 类似 call() 和 apply() 方法可以将 this 引用到任何对象