

目标

目标

讲评作业

学习目标

Date对象

如何创建Date对象

Date对象的方法

通过Date实例获取总毫秒数(也就是时间戳)

Date练习

Array对象

如何创建Array对象 定义数组

检测是否为数组

Array对象的属性

数组添加和删除元素方法

合并数组方法

数组排序方法

数组索引方法

数组转换为字符串

数组截取slice、数组删除splice

清空数组所有元素的几个方式

数组练习-筛选

数组练习-数组去重

扩展: 关联数组 了解

数组的forEach方法 要求掌握

扩展: 经典算法-冒泡排序 难点 了解

Number对象的方法

toFixed(n)

小技巧:查看某个对象的属性和方法

今日总结

今日作业

安装vscode,顺便自己学习一下怎么使用

讲评作业

关于JS中字符串赋值的问题

JS中不能直接 字符串不能 `str[i] = 'x'` 不能for循环 字符串length 然后赋值

应该

将字符串转换为数组 而且

字符`x[i]=*` 不是所有浏览器都兼容的

用

`split("")`一下就变成数组就行了

字符串有部分像数组一样的特性 但它不是数组

```
// 关于js中字符串赋值的问题
var str = "abc";
// 把字符串中的第一个字符改成大写字母,再赋回给自己
str[0] = str[0].toUpperCase() ;
// 字符串有部分像数组一样的特性,但它不是数组
console.log( str[0] );// a

console.log("");

var arr = ["a","b","c"];
// 数组第一个元素转成大写字母,再赋回给自己
arr[0] = arr[0].toUpperCase() ;
console.log( arr[0] );// A
```

判断对象中是否存在某个属性

```
var obj = {
  age : 18
};
// obj.age属性是存在的,可以得到具体值
// console.log( obj.age );
// obj.sex属性是不存在的,会得到undefined
// console.log( obj.sex );

// 判断对象中某个属性是否存在
if( obj.sex ){ // 如果属性不存在,得到undefined, 在if中会自动转成false
  console.log("有该属性");
}else{
  console.log("没有该属性");
}

if( obj.age ){
  console.log("有该属性");
}else{
  console.log("没有该属性");
}
```

作业3:

// 3、(字符, 偏难)利用var s1 = prompt("请输入任意字符","")可以获取用户输入的字符(存到变量s1中了), 要求将用户输入的字符“反转顺序”后, 再首尾字母转为大写, 其他字母转为小写,最后把结果输出在页面上

```
var s1 = prompt("请输入任意字符","");
var arr = s1.split("");
var str = "";
for(var i = arr.length-1; i>=0; i--){
    if( i == 0 || i == arr.length-1){// 首尾转成大写
        arr[i] = arr[i].toUpperCase();
    }else{// 其他转成小写
        arr[i] = arr[i].toLowerCase();
    }
    str += arr[i] ;
}
document.write("翻转之前是:" + s1 + " 翻转以后是:"+str );
```

作业6:

```
<script type="text/javascript">
    // 简单的随机点名, 刷新页面随机出现一个人的名字 (思路 把所有名字放到一个数组里面, 随机找出来)

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

    var arr = [
        "张三",
        "李四",
        "王五",
        "赵六",
        "孙七",
        "雷天使",
        "大天使",
        "小天使"
    ];

    // 最小下标是0,最大下标是数组长度-1
    var index = getRandom(0,arr.length-1);
    document.write("随机下标为:" + index );
    document.write("<hr/>");
    document.write( arr[index] );
</script>
```

作业7:

```

<script type="text/javascript">
    function getRandom(min,max){
        return Math.floor( Math.random()*(max-min+1)+min );
    }

    // 随机生成16进制的颜色,并输出在控制台中,每次刷新,输出的16进制颜色值都不相同
    // 十六进制的颜色由#开头,0~9 A~F或者a~f一共6位组成
    var str = "0123456789ABCDEF";

    // 分割字符串变成一个数组
    var arr = str.split("");

    // 定义一个新字符串
    var color = "#";

    for(var i=1;i<=6;i++){
        // 得到数组随机下标
        var index = getRandom(0,arr.length-1);
        color += arr[index] ;
    }
    console.log( color );
</script>

```

作业8:

```

// 进阶题难度比较高,判断一个字符串 'abcfoxyozzopp' 中出现次数最多的字符,并统计其次数。
var str = 'abcfoxyozzopp';
/*
    obj.字符 = 字符串出现的次数
    obj.a = 1
    obj.b = 1
    obj.c = 1
    obj.e = 1
    obj.f = 1
    obj.o = 4
    .....

    核心算法: 可以利用 字符串对象.charAt() 得到这个字符串中的每个字符
    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
    遍历对象,得到最大值和该字符
*/
// 创建一个空对象
var obj = {};
// 添加属性
// obj.属性名 = 属性值;
// obj[属性名] = 属性值;

```

```

// 遍历字符串中每个字符串
for(var i=0;i<str.length;i++){
    // console.log( str[i] );
    // console.log( str.charAt(i) );

    // 给obj添加属性
    // obj[ str.charAt(i) ] = 1;

    // 判断对象中某个属性是否存在,如果存在就字符出现次数+1,如果不存在,就设置字符串出现的次数为1
    if( obj[ str.charAt(i) ] ){ // 如果存在就字符出现次数+1
        // obj[ str.charAt(i) ] = obj[ str.charAt(i) ] + 1;
        obj[ str.charAt(i) ]++;
    }else{ // 如果不存在,就设置字符串出现的次数为1
        obj[ str.charAt(i) ] = 1;
    }
}

// console.log( obj );

// 遍历对象
var max = 0; // 设置一个变量保存字符出现的最大次数
var char = ""; // 设置出现最多次数的字符串为""
for(var j in obj){
    // console.log( obj[j] );
    // console.log( j );

    if( max < obj[j] ){ // 判断假设的跟属性值比较
        max = obj[j];
        char = j
    }
}
console.log( "最多的字符是"+char+"出现了"+max+"次");

```

学习目标

- 能够理解Date对象的方法
- 能够掌握Array对象的属性和方法

Date对象

Date对象 和 Math 对象不一样, **Date他是一个构造函数**。 所以我们需要 **实例化使用**。

创建 Date 实例用来处理日期和时间。Date 对象基于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](#)

如何创建Date对象

1) 使用new关键字和Date()方法来创建 不写参数 就返回当前时间

```
1 new Date();
```

2) 使用new关键字和Date()方法来创建 写参数,就返回括号里面输入的时间

```
1 建议日期格式字符串这样写 new Date('2018-8-8 11:20:24') 或者 new Date('2018/8/8 11:20:24')
```

3)new Date()方法传多个参数,具体用法参考手册

Date 对象

启用基本存储器并取得日期和时间。

```
dateObj = new Date()  
dateObj = new Date(dateVal)  
dateObj = new Date(year, month, date[, hours[, minutes[, seconds[, ms]]]])
```

参数

dateObj

必选项。要赋值为 Date 对象的变量名。

dateVal

必选项。如果是数字值, dateVal 表示指定日期与 1970 年 1 月 1 日午夜间[全球标准时间](#)的毫秒数。如果是字符串, 则 dateVal 按照 parse 方法中的规则进行解析。dateVal 参数也可以是从某些 ActiveX(R) 对象返回的 VT_DATE 值。

year

必选项。完整的年份, 比如, 1976 (而不是 76)。

month

必选项。表示的月份, 是从 0 到 11 之间的整数 (1 月至 12 月)。

date

必选项。表示日期, 是从 1 到 31 之间的整数。

hours

可选项。如果提供了 minutes 则必须给出。表示小时, 是从 0 到 23 的整数 (午夜到 11pm)。

minutes

举例:

```
1 <script type="text/javascript">  
2 // 如何创建Date对象  
3 // 1) 使用new关键字和Date()方法来创建 不写参数 就返回当前时间  
4 var date1 = new Date();  
5 console.log( date1 );  
6  
7 // 2) 使用new关键字和Date()方法来创建 写参数,就返回括号里面输入的时间  
8 var date2 = new Date("1975-07-05 08:25:20");  
9 console.log( date2 );  
10  
11 var date3 = new Date("1975/07/05 08:25:20");  
12 console.log( date3 );  
13  
14 // 只设置日期,时间会默认使用00:00:00  
15 var date5 = new Date("2008/8/8");  
16 console.log( date5 );  
17  
18 var date6 = new Date("2008/08/08");
```

```

19 console.log( date6 );
20
21 // 3) dateObj = new Date(year, month, date[, hours[, minutes[, seconds
    [,ms]]]])
22 // 如果使用第三种方式创建,那个month的月份是从0开始的,0代表1月
23 var date7 = new Date(2020,3,4,10,35,56);
24 console.log( date7 );
25 </script>

```

Date对象的方法

如果我们想要 2018-8-8 8:8:8 格式怎么办?

我们需要手动的得到这种格式

方法名	功能
getFullYear()	获取4位数的年份
getMonth()	获取月份 返回值 0~11 0表示1月 11表示12月
getDate()	返回一个月中的某一天 返回值: 1~31
getHours()	小时 返回值0~23
getMinutes()	获取分钟 返回值: 0~59
getSeconds()	获取秒数 返回值: 0~59
getMilliseconds()	获取毫秒 返回值: 0~999
getDay()	获取一周中的某一天 就是星期几 返回值: 0~6 0代表星期天 1代表星期一
getTime()	获取时间戳 返回从1970年1月1日 一直到现在的 毫秒数!
toLocaleString()	根据本地时间把 Date 对象转换为字符串, 并返回结果。

注意 月份 和 星期 取值范围是从 0开始的。

getDay 方法所返回的值是一个处于 0 到 6 之间的整数, 它代表了一周中的某一天, 返回值与一周中日期的对应关系如下:

值	星期
0	星期天
1	星期一
2	星期二
3	星期三
4	星期四
5	星期五
6	星期六

举例:

```

1 <script type="text/javascript">
2 // Date对象的方法
3 // 如果我们想要 2018-8-8 8:8:8 格式怎么办?

```



```
4 // var date = new Date("2018-8-8 8:8:8");
5 var date = new Date();
6 // console.log( date );
7
8 // getFullYear() 获取4位数的年份
9 var year = date.getFullYear() ;
10
11 // getMonth() 获取月份 返回值 0~11 0表示1月 11表示12月
12 var month = date.getMonth();
13
14 // getDate() 返回一个月中的某一天 返回值: 1~31
15 var day = date.getDate();
16
17 // getHours() 小时 返回值0~23
18 var hours = date.getHours();
19
20 // getMinutes() 获取分钟 返回值: 0~59
21 var minutes = date.getMinutes();
22
23 // getSeconds() 获取秒数 返回值: 0~59
24 var seconds = date.getSeconds();
25
26 // getMilliseconds() 获取毫秒 返回值: 0~999
27 var milliseconds = date.getMilliseconds();
28
29
30 // getDay() 获取一周中的某一天 就是星期几 返回值: 0~6 0代表星期天 1代表星期
一
31 var week = date.getDay();// 3
32
33 // getTime() 获取时间戳 返回从1970年1月1日 一直到现在的"毫秒数"!
34 var time = date.getTime();
35
36 // toLocaleString() 根据本地时间把 Date 对象转换为字符串, 并返回结果。
37 var localeString = date.toLocaleString();
38
39 console.log( year,month,day,hours,minutes,seconds,milliseconds,week,time
e );
40 console.log( localeString );
41
42 // 想得到指定格式的日期时间,可以自己拼接字符串
43 // 如果我们想要 2018-8-8 8:8:8 格式怎么办?
```

```

44  var arrWeek = ["天", "一", "二", "三", "四", "五", "六"];
45
46  console.log(year + "-" + (month + 1) + "-" + day + " " + hours + ":" + minutes + ":" + seconds + " 星期" + arrWeek[week] );
47  </script>

```

通过Date实例获取总毫秒数(也就是时间戳)

总毫秒数的含义: 基于1970年1月1日（世界标准时间）起的毫秒数

如何获取总毫秒数:

- 1 第一种方法: 日期对象.`valueOf()` 或者 日期对象.`getTime()`
- 2 第二种方法: `+new Date()`
- 3 第三种方法: HTML5中提供的方法, 有兼容性问题 `Date.now()`

```

1  <script type="text/javascript">
2  // 通过Date实例获取总毫秒数(也就是时间戳)
3  // 总毫秒数的含义: 基于1970年1月1日（世界标准时间）起的毫秒数
4
5  // 获取当前日期时间
6  var date = new Date();
7
8  // 第一种方法: 日期对象.getTime() 或者 日期对象.valueOf()
9  console.log( date.getTime() );
10 console.log( date.valueOf() );
11
12 // 第二种方法: +new Date()
13 console.log( +date );
14
15 // 第三种方法: HTML5中提供的方法, 有兼容性问题 Date.now()
16 console.log( Date.now() );
17 </script>

```

Date练习

练习: 请把当前时间写成这个格式 2018年12月02日 16:05:07 星期天

```

3 <head>
4   <meta charset="UTF-8">
5   <title>Document</title>
6   <script type="text/javascript">
7     // 前导0, 当值小于10的时候, 在值前面补个0, 因为这个功能需要重复使用, 所以做成函数
8     function zero( num ){
9       return num < 10 ? "0"+num : num;
10    }
11
12    // 练习: 请把当前时间写成这个格式    2018年12月02日 16:05:07 星期天
13    var date = new Date();
14    var year = date.getFullYear() ;
15    var month = zero( date.getMonth()+1 );
16    var day = zero( date.getDate());
17    var hours = zero( date.getHours());
18    var minutes = zero( date.getMinutes());
19    var seconds = zero( date.getSeconds());
20    var week = date.getDay();
21    var arrWeek = ["天","一","二","三","四","五","六"];
22
23    document.write( year+"年"+month+"月"+day+"日 "+hours+":"+minutes+":"+seconds+" 星期"+arrWeek[week]);
24  </script>
25 </head>
26 <body>
27

```

Array对象

如何创建Array对象 定义数组

1) 使用[]来创建

2) 使用new Array()方法来创建

注意：上面代码中arr创建出的是一个空数组，如果需要使用构造函数Array创建非空数组，可以在创建数组时传入参数

参数传递规则如下：

如果只传入一个参数，则参数规定了数组的长度

如果传入了多个参数，则参数称为数组的元素

举例：

05: 如何创建Array对象.html

```

1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <title>Document</title>
6   <script type="text/javascript">
7     // Array对象就是数组对象
8     // 如何创建Array对象
9     // 使用[]创建数组
10    var arr1 = [10,20,30];
11    console.log( arr1 );
12
13    // 使用new Array(size)创建,
14    // 长度为size的空数组
15    var arr2 = new Array(10);
16    console.log( arr2 );
17
18    // 使用new Array( 数组元素1, 数组元素2...)
19    // 创建
20    var arr3 = new Array(10,20,30);
21    console.log( arr3 );

```

文件 | F:/GZ20H5直播6班/Javascript基础/

Elements
Console
Sources
Ne

top
Filter

```

▶ (3) [10, 20, 30]
▶ (10) [empty × 10]
▶ (3) [10, 20, 30]

```

检测是否为数组

- instanceof 运算符

instanceof 可以判断一个对象是否是某个构造函数的实例

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

- Array.isArray()

Array.isArray()用于判断一个对象是否为数组，isArray() 是 HTML5 中提供的方法

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

```
06-检测是否为数组.html
<script type="text/javascript">
// 检测是否为数组
var str = "abc";
var num = 10;
var bool = true;
var arr = [10,20,30];
var arr2 = new Array( 10,20,30 );
// 方法一: instanceof 运算符
// instanceof 可以判断一个对象是否是某个构造函数的实例
console.log( str instanceof Array );// false
console.log( num instanceof Array );// false
console.log( bool instanceof Array );// false
console.log( arr instanceof Array );// true
console.log( arr2 instanceof Array );// true
console.log("");
console.log("");
console.log("");
// 方法二:Array.isArray()
// Array.isArray()用于判断一个对象是否为数组，isArray() 是 HTML5 中提供的方法
console.log( Array.isArray(str) );// false
console.log( Array.isArray(num) );// false
console.log( Array.isArray(bool) );// false
console.log( Array.isArray(arr) );// true
console.log( Array.isArray(arr2) );// true
</script>
```

Array对象的属性

属性名	功能
-----	----

ArrayObject.length	返回数组的长度 数组元素的个数 数组的最大下标=数组的长度-1
--------------------	---------------------------------

```

1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <title>Document</title>
6   <script type="text/javascript">
7     // 数组的属性,我们就需要知道一个
8     // 数组的长度  数组对象.length  数组的长度就是数组元素的个数
9     var arr1 = [10,20,30];
10    console.log( arr1.length );// 3
11
12    var arr2 = new Array(10);
13    console.log( arr2.length );// 10
14
15    // 最大下标 等于 数组的长度 - 1
16  </script>
17 </head>
18 <body>
19
20 </body>
21 </html>

```

数组添加和删除元素方法

方法名	说明	返回值
push(参数1..)	修改原数组，末尾添加一个或多个数组元素	并返回新的长度
pop()	删除 数组的最后一个元素，把数组长度减 1 无参数	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数组元素	并返回新的长度
shift()	把数组的第一个元素从其中删除，把数组长度减 1 无参数	返回它删除的元素的值

举例:

```

1 <script type="text/javascript">
2   // 数组添加和删除元素方法
3   var arr = [10,20,30];
4   // push(参数1..) 修改原数组，在数组末尾添加一个或多个数组元素 并返回新的长度
5   console.log( arr.push(40,50,60) );
6   console.log( arr );
7   console.log( "" );
8
9   // pop() 删除 数组的最后一个元素，把数组长度减 1 无参数 返回它删除的元素的值

```

```

10 console.log( arr.pop() );
11 console.log( arr );
12 console.log("");
13
14 // unshift(参数1...) 向数组的开头添加一个或更多数组元素 并返回新的长度
15 console.log( arr.unshift(1,2,3) );
16 console.log( arr );
17 console.log("");
18
19 // shift() 把数组的第一个元素从其中删除, 把数组长度减 1 无参数 返回它删除的元素的值
20
21 console.log( arr.shift() );
22 console.log( arr );
23 </script>

```

合并数组方法

方法名	说明	返回值
concat(数组1,数组2,数组3...)	将多个数组合并为一个数组	返回一个新的数组

举例:

```

5 <title>Document</title>
6 <script type="text/javascript">
7   // 合并字符串
8   /*var str1 = "abc";
9   var str2 = "erwer";
10  var str3 = "zxv";
11  console.log( str1+str2+str3 );
12  console.log( str1.concat(str2,str3) );*/
13
14  // 合并数组方法
15  // concat(数组1,数组2,数组3...)
16  // 将多个数组合并为一个数组
17  // 返回一个新的数组
18
19  var arr1 = [10,20,30];
20  var arr2 = [5,6,7];
21  var arr3 = [4,2,3];
22  //
23  // 如果多个数组使用+号拼接会变成一个字符串
24  console.log( arr1+arr2+arr3 );
25  console.log( typeof( arr1+arr2+arr3) );
26
27  var arr = arr1.concat(arr2,arr3);
28  console.log( arr );
29  console.log( typeof(arr) );
30 </script>

```

Console Output:

```

10,20,305,6,74,2,3
string
▶ (9) [10, 20, 30, 5, 6, 7, 4, 2, 3]
object

```

数组排序方法

- 数组中有对数组本身排序的方法，部分方法如下表

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

注意： sort方法需要传入一个函数作为参数来设置升序、降序排序

- 如果传入" function(a,b){ return a-b;}", 则为升序
- 如果传入" function(a,b){ return b-a;}", 则为降序

sort 方法

返回一个元素已经进行了排序的 **Array** 对象。

`arrayobj.sort(sortfunction)`

参数

arrayObj

必选项。任意 **Array** 对象。

sortFunction

可选项。是用来确定元素顺序的函数的名称。如果这个参数被省略，那么元素将按照 **ASCII** 字符顺序进行升序排列。

说明

sort 方法将 **Array** 对象进行适当的排序；在执行过程中并不会创建新的 **Array** 对象。

如果为 *sortfunction* 参数提供了一个函数，那么该函数必须返回下列值之一：

- 负值，如果所传递的第一个参数比第二个参数小。
- 零，如果两个参数相等。
- 正值，如果第一个参数比第二个参数大。

ASCII 字符代码表 一

高四位		ASCII非打印控制字符										ASCII 打印字符											
		0000					0001					0010	0011	0100	0101	0110	0111						
		0					1					2	3	4	5	6	7						
低四位		+进制	字符	ctrl	代码	字符解释	+进制	字符	ctrl	代码	字符解释	+进制	字符	+进制	字符	+进制	字符	+进制	字符	+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	垂直制表符	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

举例:

```

1 <script type="text/javascript">
2 // 数组排序的方法
3 // 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
4 var arr1 = [10,20,30,"a","b","c"];
5 // reverse()会返回翻转以后的新数组
6 console.log( arr1.reverse() );
7 // reverse()会改变原数组
8 console.log( arr1 );
9 console.log( "" );
10
11 // 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
12 var arr2 = ["a",5,7,8,"b",3,1,"A",2];
13 // sort()方法可以对数组中的元素进行排序,排序一般分为升序跟降序 sort可以传一个
   参数,这个参数是一个函数,这个函数我们称之为排序函数,排序函数需要返回 0 或者 正值 或
   者 负值
14
15 // sort()如果不传参,默认会按ASCII码升序排序
16 arr2.sort();
17 console.log( arr2 );
18 console.log( "" );
19
20 // 两个数比较,只有三种结果,大于,小于,等于;那么如果两个数相减,也是有三种结果,大
   于0,小于0,等于0

```



```
21 // 10 - 5 = 5 >0
22 // 10 - 20 = -10 <0
23 // 10 - 10 = 0
24
25
26 // 升序函数
27 function asc(a,b){
28     return a-b;
29 }
30 // 降序函数
31 function desc(a,b){
32     return b-a;
33 }
34
35 var arr3 = [5,3,1,7,8,10];
36 // 如果要给sort传参数的时候,可以给函数名或者一个匿名函数
37 arr3.sort( asc );
38 console.log( arr3 );
39
40 var arr4 = [50,2,60,1,75];
41 arr4.sort( desc );
42 console.log( arr4 );
43 console.log("");
44
45 var arr5 = [20,30,50,10,70];
46 arr5.sort(function(a,b){
47     return a-b;
48 });
49 console.log( arr5 );
50 console.log("");
51
52 var arr6 = [10,5,7,9,2,3];
53 // Math.random() 得到0到1之间的小数,包括0不包括1
54 // console.log( Math.random() );
55 // console.log( Math.random() - 0.5 );
56 arr6.sort( function(){
57     return Math.random()-0.5;
58 } );
59 console.log( arr6 );
60 </script>
```

数组索引方法

- 数组中有获取数组指定元素索引值的方法，部分方法如下表

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()	查找给定元素在数组中的最后一个索引	如果存在返回索引号 如果不存在,则返回-1

```
<script type="text/javascript">
// 查找指定内容是否在数组中出现,找到的话,返回第一个出现的位置(数组元素下标);如果找不到,返回-1

var arr = [10,20,30,"a",40,"a",50,60,10,70];
// 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
console.log( arr.indexOf( "a" ) );// 3
console.log( arr.indexOf( 10 ) );// 0
console.log( arr.indexOf( "b" ) );// -1
console.log("");

// lastIndexOf() 查找给定元素在数组中的最后一个索引 如果存在返回索引号 如果不存在,则返回-1
console.log( arr.lastIndexOf( "a" ) );// 5
console.log( arr.lastIndexOf( 10 ) );// 8
console.log( arr.lastIndexOf( "b" ) );// -1
</script>
```

数组转换为字符串

- 数组中有把数组转化为字符串的方法，部分方法如下表

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")	用于把数组中的所有元素以分隔符连接,合并为一个字符串;	返回一个字符串

注意：join方法如果不传入参数，则按照“，”拼接元素

- 1 数组对象.`join("分隔符")` => 把数组中,所有数组元素之间使用分隔符连接成一个字符串
- 2
- 3 字符串对象.`split("分隔符")` => 根据分隔符把字符串分割成一个数组

举例:

- 1 <script type="text/javascript">
- 2 // 昨天的我们讲过使用 字符串.split(分隔符) 可以把字符串根据分隔符分割成一个数组

```

3  var str = "张三|李四|王五|赵六|李狗蛋|李二狗";
4  var arr = str.split("|");
5  console.log( arr );
6  console.log( typeof(arr) );
7  console.log("");
8
9  // 数组转换成字符串,常见两种方法
10 // toString() 把数组转换成字符串,"逗号"分隔每一项 返回一个字符串
11 var arr1 = [10,20,30,true,"abc","男"];
12 console.log( arr1.toString() );
13 console.log("");
14
15 // join("分隔符") 用于把数组中的所有元素以分隔符连接,合并为一个字符串; 返回一个字符串
16 console.log( arr1.join(",") );
17 console.log( arr1.join("-") );
18 console.log( arr1.join("|") );
19 console.log( arr1.join("^_^") );
20 console.log("");
21
22 console.log( arr1.join("") );
23 console.log( arr1.join(" ") );
24 // 如果join没有传参,默认是使用逗号连接每一项数组元素
25 console.log( arr1.join() );
26 </script>

```

数组截取slice、数组删除splice

方法名	说明	返回值
slice(begin下标,end下标)	数组截取	返回截取出来的新数组 包括 begin, 不包括end, 这个不会影响原数组
splice(从第几个开始,要删除几个)	数组删除	返回被删除的数组元素的组成的数组,这个会影响原数组

举例:

```

<script type="text/javascript">
    // 数组截取slice、数组删除splice
    var arr1 = [10,20,30,40,50,60,70];
    // slice(begin下标,end下标) 数组截取
    返回截取出来的新数组 包括 begin，不包括end，
    这个不会影响原数组

    // slice截取,就是数组中截取一小部分出来,
    返回截取出来的数组
    console.log( arr1.slice(1,4) );
    // slice不会影响,这个不会影响原数组
    console.log( arr1 );
    console.log("");

    // splice(从第几个开始,要删除几个) 数组删除
    返回被删除的数组元素的组成的数组,这个会影响原数组
    // splice更加常用,更加灵活, 有点类似
    字符串中的substr方法
    var arr2 = [5,6,7,8,2,1];
    // 返回被删除的数组元素的组成的数组
    console.log( arr2.splice(1,2) );
    console.log( arr2 );
    console.log("");

    var arr3 = [1,2,3,4,5,6,7,8];
    console.log( arr3.splice(3,1) );
    console.log( arr3 );
</script>

```

清空数组所有元素的几个方式

```

1 // 方式1 赋值一个空数组 推荐
2 arr = [];
3
4 // 方式2 设置数组的长度为0
5 arr.length = 0;

```

举例:

```

<script type="text/javascript">
    // 清空数组所有元素的两种方式
    // 方式一    赋值一个空数组 推荐
    var arr1 = [10,20,30,40,50,60];
    console.log( arr1 );
    arr1 = [];
    console.log( arr1 );
    console.log("");

    // 方式二    设置数组的长度为0
    var arr2 = [5,7,89,2,3,75];
    console.log( arr2 );
    arr2.length = 0;
    console.log( arr2 );
</script>

```

数组练习-筛选

工资的数组[1500, 1200, 2000, 2100, 1700],把工资超过1800的删除

思路:把小于1800的工资放进一个新的数组中

```

<script type="text/javascript">
    // 工资的数组[1500, 1200, 2000, 2100, 1700],把工资超过1800的删除
    // 思路:把小于1800的工资放进一个新的数组中
    var arr = [1500, 1200, 2000, 2100, 1700];
    var newArr = [];
    for(var i=0;i<arr.length;i++){
        if(arr[i] < 1800){
            newArr.push( arr[i] );
        }
    }
    console.log( newArr );
</script>

```

数组练习-数组去重

要求: 去除数组中重复的元素 ['c', 'a', 'z', 'a', 'x', 'a', 'x', 'c', 'b']

实现思路分析

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

代码实现：

```
<script type="text/javascript">
// 要求： 去除数组中重复的元素 ['c', 'a', 'z', 'a', 'x', 'a', 'x', 'c', 'b']
// 意思就是只留下不重复的元素

// 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

var arr = ['c', 'a', 'z', 'a', 'x', 'a', 'x', 'c', 'b'];
// 定义一个新的空数组
var newArr = [];
for(var i=0;i<arr.length;i++){
    // 判断某个值在数组中是否存在,可以使用indexOf,如果存在返回对应的下标,如果不存在返回-1
    if( newArr.indexOf(arr[i]) == -1 ){ // 在新数组中查询是否存在旧数组中的某个元素,如果不存在,
        // 则把这个元素添加到新数组中
        newArr.push( arr[i] );
    }
}
console.log( newArr );
</script>
```

扩展: 关联数组 了解

之前我们学的数组的下标都是数值,这样的数组,我们叫索引数组;还有一种数组的下标是"字符串",这种我们叫关联数组

定义关联数组的步骤如下：

- 1 第一步：先定义一个空数组
- 2 第二步：然后使用数组"字符串下标"给数组添加元素

举例：

```

<script type="text/javascript">
    // 我们之前学的数组下标都是数值,是从0开始的,这样的数组,我们叫索引数组
    // 如果数组的下标是字符串,那么这样的数组,我们叫关联数组

    // 创建关联数组的步骤
    // 1.先定义一个空数组
    // 2.给数组中添加数组元素,不过这个数组元素的下标是字符串

    var arr = [];
    arr["username"] = "张三";
    arr["age"]      = 23;
    arr["sex"]      = "男";
    console.log( arr );

    // 想访问关联数组中的某个数组元素
    console.log( arr["username"] );
    console.log("");

    // 遍历关联数组 for...in
    for(var i in arr){
        console.log( arr[i] );
    }
</script>

```

数组的forEach方法 要求掌握

forEach对数组进行循环。（一般我们也会用for对数组进行循环，但是用for会麻烦一些，因为它要用循环变量）

语法:

```

1  数组对象.forEach(回调函数)
2
3  数组对象.forEach(回调函数)    每遍历一个元素 回调函数就会被调用一次
4
5  数组对象.forEach(function(item,index,arr){
6      //item 正在遍历的元素
7      //index 正在遍历的索引
8      //arr 正在遍历的数组
9  })
10
11
12  注意:
13  (1)形参是一个回调函数。
14  (2)回调函数的三个形参分别: 当前的数组元素 , 元素的索引 , 当前的数组
15  修改item并不影响原数组(有一个前提, 这个item的值是基本数据类型。
16  即如果它是引用数据类型, 则这个修改会影响原数组)
17  (3)数组对象.forEach() 只是单纯地进行遍历 函数返回值为undefined

```

举例:

```
1 <script type="text/javascript">
2 // forEach对数组进行循环。（一般我们也会用for对数组进行循环，但是用for会麻烦一些，因为它要用循环变量）
3
4 /*
5  forEach语法
6
7  数组对象.forEach(function(item,index,arr){
8  // item数组元素
9  // index数组的下标
10 // arr原数组
11 })
12
13 注意：
14 (1)形参是一个回调函数。
15 (2)回调函数的三个形参分别：当前的数组元素， 元素的索引， 当前的数组
16 修改item并不影响原数组(有一个前提，这个item的值是基本数据类型。
17 即如果它是引用数据类型，则这个修改会影响原数组)
18 (3)数组对象.forEach() 只是单纯地进行遍历 函数返回值为undefined
19 */
20
21 /*
22 var arr = [10,30,20,5,6];
23 for(var i=0;i<arr.length;i++){
24 console.log( arr[i] );
25 }
26 */
27
28 var arr = [10,30,20,5,6];
29 var arr2 = arr.forEach( function(item,index,arr){
30 // console.log( item );
31 // console.log( index );
32 // console.log( arr );
33
34 // return item*10;
35
36 console.log( item*10 );
37 });
```



```
38
39 console.log( arr );// [10,30,20,5,6]
40
41 console.log( arr2 );// undefined
42 </script>
```

扩展: 经典算法-冒泡排序 难点 了解

冒泡排序 (Bubble Sort) , 是一种计算机科学领域的较简单的排序算法

冒泡排序思想

让数组当中相邻的两个数进行比较, 数组当中比较小的数值向下沉, 数值比较大的向上浮! 外层for循环控制循环次数, 内层for循环控制相邻的两个元素进行比较。

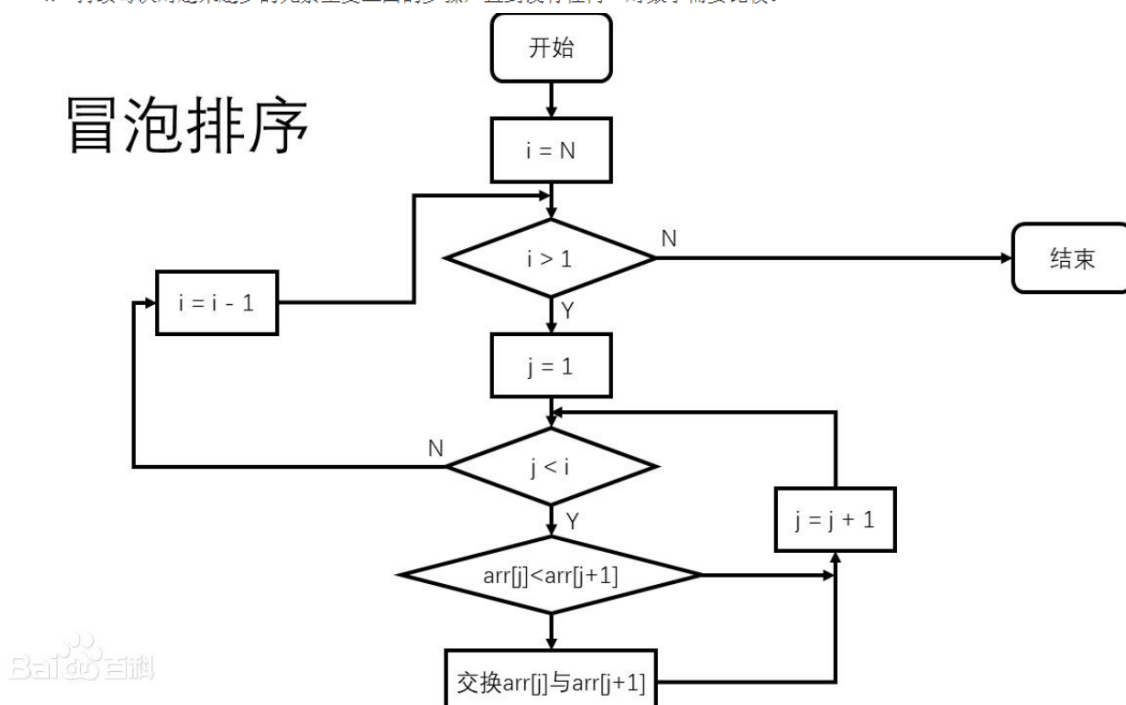
算法原理

编辑

冒泡排序算法的原理如下: [1]

1. 比较相邻的元素。如果第一个比第二个大, 就交换他们两个。 [1]
2. 对每一对相邻元素做同样的工作, 从开始第一对到结尾的最后一对。在这一点, 最后的元素应该会是最大的数。 [1]
3. 针对所有的元素重复以上的步骤, 除了最后一个。 [1]
4. 持续每次对越来越少的元素重复上面的步骤, 直到没有任何一对数字需要比较。 [1]

冒泡排序



也可以参考网站 https://blog.csdn.net/qq_31116753/article/details/84103610

代码:

```
// 封装函数冒泡排序函数
function bubbleSort( arr ){
    for(var i=0; i < arr.length-1; i++){
        for(var j=0; j < arr.length-1-i; j++){
            // 两两比较
            if(arr[j] > arr[j+1] ){// 什么时候才交换位置,前面的数比较大,
                那么我们就让前面的数跟后面的那个数交换位置
                // 定义一个临时变量
                var temp = arr[j];
                arr[j] = arr[j+1];
                arr[j+1] = temp;
            }
        }
    }
    return arr;
}

var result = bubbleSort( [10,5,3,2,88,99,4] );
console.log( result );
```

Number对象的方法

toFixed(n)

将一个数字进行保留n位小数 需要进行**四舍五入** 获取到以后是一个String类型

```
1 //创建Number 对象
2 //方式一: 定义一个Number变量                var num = 123.456789
3 //方式二: 使用new关键字 Number()方法来创建 var num = new Number(123.456789)
4
5 //保留n位小数    Number对象变量名.toFixed(n)
```

举例:

```

4  <meta charset="UTF-8">
5  <title>Document</title>
6  <script type="text/javascript">
7      // Number对象的方法,就是数值对象的方法
8      // toFixed(n) 将一个数字进行保留n位小数
9      需要进行"四舍五入" 返回是一个String类型
10
11      var num1 = 10;
12      console.log( num1.toFixed(2) );// "10.00"
13      console.log( typeof( num1.toFixed(2) ) );//
14      string
15      console.log("");
16
17      var num2 = 10.123456789;
18      console.log( num2.toFixed(4) );
19      console.log( num2.toFixed(3) );
20      console.log( num2.toFixed(7) );
21      console.log("");
22
23      var num3 = 10.12345;
24      console.log( num3.toFixed(4) );
25  </script>

```

文件 | F:/GZ20H5直播6班/Javascript基础/08

← → ↺

Elements Console Sources Netv

top

Filter

10.00

string

10.1235

10.123

10.1234568

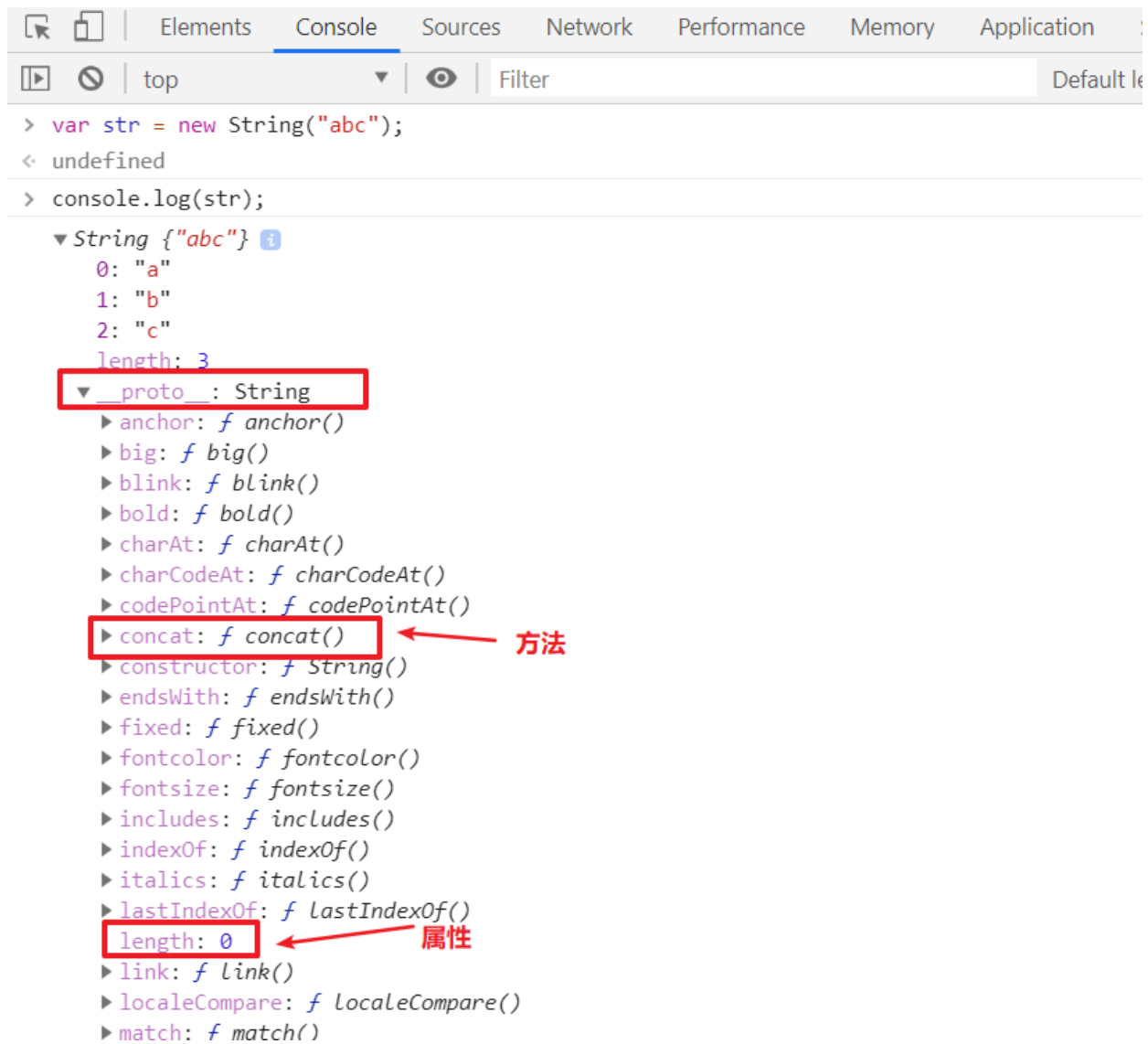
10.1235

小技巧:查看某个对象的属性和方法

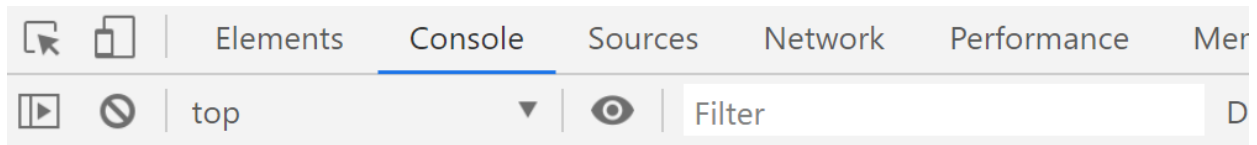
如何通过浏览器调试工具的控制台查看某个对象的属性与方法?

步骤如下:

- 1 第一步:在控制台中,使用new的方式实例化对象 比如 `var str = new String("abc");`
- 2 第二步:在控制台中,`console.log(对象名)`; 比如 `console.log(str);`
- 3 第三步:接着展开__proto__即可查看属性和方法,有f以及有()的是方法



另外 Math对象是不需要实例化的,直接使用即可



> Math

< ▼ Math {abs: f, acos: f, acosh: f, asin: f, asinh: f, ...} ⓘ

E: 2.718281828459045

LN2: 0.6931471805599453

LN10: 2.302585092994046

LOG2E: 1.4426950408889634

LOG10E: 0.4342944819032518

PI: 3.141592653589793

SQRT1_2: 0.7071067811865476

SQRT2: 1.4142135623730951

▶ abs: f abs()

▶ acos: f acos()

▶ acosh: f acosh()

▶ asin: f asin()

▶ asinh: f asinh()

▶ atan: f atan()

▶ atan2: f atan2()

▶ atanh: f atanh()

Math对象不需要实例化,直接使用即可

今日总结



xmind自己做,要交

今日作业

请查看作业文件夹

安装vscode,顺便自己学习一下怎么使用

脑 > 文档 (F:) > GZ20H5直播6班 > Javascript基础 > 08-JavaScript基础 > 3-其他资料 >

名称	修改日期	类型	大小
 vscode使用与安装包	2020/3/1 16:26	文件夹	
 冒泡排序动画演示	2020/3/1 16:26	文件夹	