数组的方法

arr.includes()

arr.every(fn(val,i))

arr.some(fn(val,i))

arr.filter(fn(val,i))

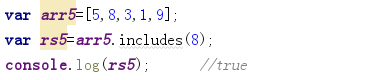
arr.map(fn(val,i))

ar.reduce(fn(prev,current),initValue)

arr.sort()或arr.sort(fn(a,b))

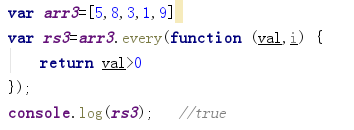
1.arr.includes(function(val,i){})判断是否包含某个元素

返回值是布尔值。循环数组，每循环一次，执行一次回调函数，只要有一个元素为回调函数的指定元素，就结束循环，结果就是true；没有元素符合回调函数的指定元素，结果就是flase。



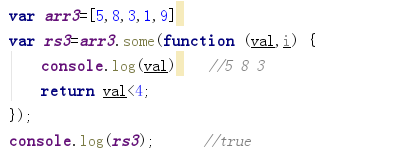
2.arr.every(function(val,i){})判断所有

返回值是布尔值。循环数组，每循环一次，执行一次回调函数，只要有一个元素的回调函数的返回值是false，就结束循环，结果就是false；只有所有元素符合回调函数的条件，结果就是true。



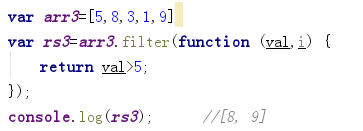
3.arr.some(function(val,i){})判断是否含有

返回值是布尔值。循环数组，每循环一次，执行一次回调函数，只要有一个元素的回调函数的返回值是true，就结束循环，结果就是true；只有所有元素不符合回调函数的条件，结果就是false。



4.arr.filter(function(val,i){})过滤符合条件的元素，集合成数组

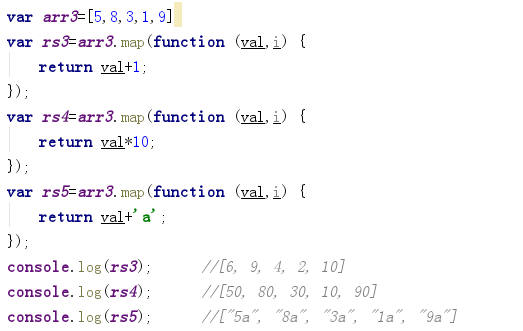
返回值是数组。循环数组，每循环一次，执行一次回调函数，只要元素满足回调函数的条件，就将元素添加到新数组，循环结束后，返回新数组。



5.arr.map(function(val,i){})按条件映射出新的元素，集合成数组

返回值是数组。循环数组，每循环一次，执行一次回调函数，按照条件映射出新的元素，集合成新的数组，并返回数组。

**技巧：**拼接ul中li时，可以datas.map(function(){return '<li><li>'}).join('')



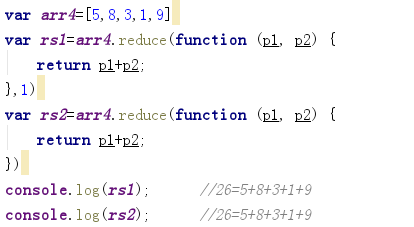
6.arr.reduce(callback,initValue)

返回回调函数的计算之后的值。

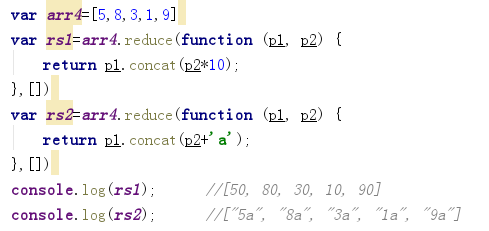
callback(prevValue, currentValue)

prevValue: 上一次的值；

currentValue: 当前的值。
initValue会赋值给prevValue作为初始值，如果没有传入，那么prevValue的值为数组的第0个；callback每次返回的值将作为callback下一次执行的prevValue；循环完成以后最后一次callback返回的值将作为该reduce函数的结果。



arr.reduce()也可以完成arr.map()映射的任务。



7.arr.sort(function(val,i){})重排

返回重排之后的数组。

a-b; 从小到大

a-b>0; 从小到大

a-b<0; 从大到小

b-a; 从大到小

b-a>0; 从大到小

b-a<0; 从小到大

1,true; 倒序

-1,false;顺序

0； 不变位置



