# 高阶函数

函数接受另一个函数作为参数，称之为高阶函数。

function add(x,y,z){

return f(x)+f(y);

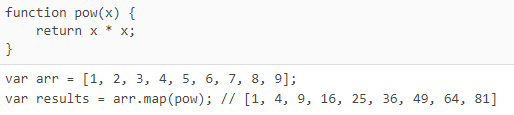
}

add(5,6,Math.abs);

## map

map()方法定义在Array数组中，我们调用数组的map方法，传入我们自己的函数，就得到了一个新的数组作为结果。

map将运算规则抽象，很明了的明白，对数组的每一项进行操作。

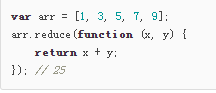


还有例如将一个数组的每一项元素都变成字符串

IMG_256

## reduce

Array的reduce()把一个函数作用在数组的[x1,x2,x3]上，这个函数必须接受两个参数，reduce()把结果继续和序列的下一个元素做累计计算。

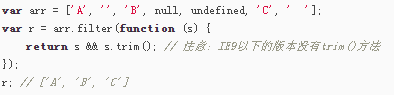


## **filter**

filter用于把数组的某些元素过滤掉，然后返回剩下的元素。

filter会接受一个参数，根据返回值是true还是false，决定保留还是丢弃该元素。

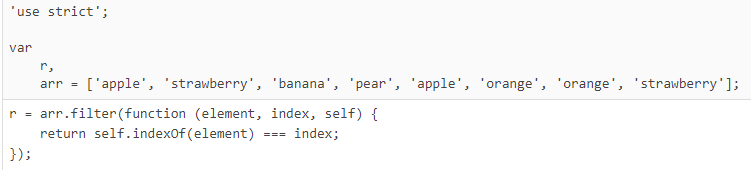
例如删除数组中的空字符串：



回调函数

filter接收的回调函数，其实可以有多个参数，我们通常只使用第一个参数。

例如使用此方法去除数组的重复元素



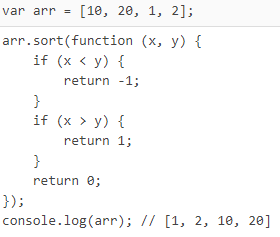
element表示元素本身，index表示元素的下表位置，sele表示数组本身。由于indexOf总返回第一个元素的位置，后续的元素位置与indexOf返回的位置不相等，因此被filter过滤掉了。

## sort

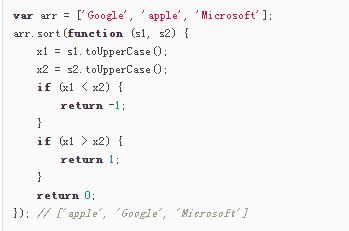
sort同样属于数组的方法，它默认吧所有元素先转化成string在排序，就会缠身跟诡异的结果。

通常规定比较元素，x<y，返回-1，x==y，返回0；x>y,返回1.

sort()同样是高阶函数，他同样可以接受一个比较函数来实现自定义的排序。



例如按照字母排序(全部变成大写就行)

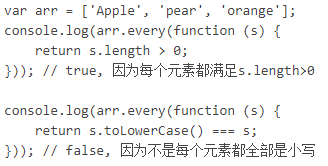


sort会直接对数组进行修改，而不是返回一个新数组。

**数组提供了非常多的高阶函数**

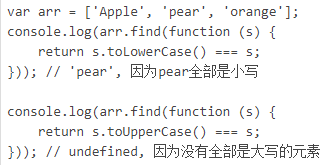
## every

every()方法可以判断数组的所有元素是否满足测试条件。



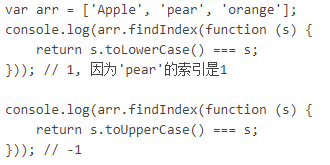
## find

find()方法用于查找符合条件的第一个元素，如果找到了，返回这个元素，否则，返回undefined



## findIndex

findIndex()和find()类似，也是查找符合条件的一个元素，不同之处在于findIndex()会返回这个元素的索引，如果没找到则返回-1。



## forEach

forEach()和map()类似，他把每个元素依次作用于窜入的函数，但不会返回新的数组，forEach常用于遍历数组，因此传入的函数不需要返回值。

