**Javascript数组的foreach方法(用于对数组中每个元素调用一遍)：**

**语法**

array.forEach(function(currentValue, index, arr), thisValue)

**参数**

|  |  |
| --- | --- |
| **参数** | **描述** |
| *function (currentValue, index, arr)* | 必需。 数组中每个元素需要调用的函数。 函数参数:   |  |  | | --- | --- | | **参数** | **描述** | | *currentValue* | 必需。当前元素 | | *index* | 可选。当前元素的索引值。可以在定义的回调函数中不定义 | | *arr* | 可选。当前元素所属的数组对象。可以在定义的回调函数中不定义 | |
| *thisValue* | 可选。传递给函数的值一般用 "this" 值。 如果这个参数为空， "undefined" 会传递给 "this" 值 |

foreach方法没有返回值。与Map函数相比，foreach直接改变的是原数组，而Map函数在不改动原数组得情况下生成新的数组。

**Filter函数（用于过滤数组中的元素）**

在Javascript中，我们可以用数组的filter函数从数组中取出某些对象，例如在list of dict中取出某个dict元素。

注意的是，filter方法将提供一个回调函数作为参数，该回调函数包括三个参数，元素自身，元素的index以及原数组。如果在回调函数中返回值为true则将元素将会保留，如果返回值为false该元素将会被去除。**返回的结果元素放到一个新的数组中**，而不影响原数组。

例如：

var array = [1, 2, 3, 4, 5, 6, 7, 8, 9, 0];

var filtered = array.filter(function(value, index, arr){

return value > 5;

});

//filtered => [6, 7, 8, 9]

//array => [1, 2, 3, 4, 5, 6, 7, 8, 9, 0]

**Find函数（用于寻找数组中某个特定的元素）**

find() 方法返回通过测试（函数内判断）的数组的**第一个元素**的值。

find() 方法为数组中的每个元素都调用一次函数执行：

* 当数组中的元素在测试条件时返回 true 时, find() 返回符合条件的元素，之后的值不会再调用执行函数。
* 如果没有符合条件的元素返回 undefined

注意: find() 对于空数组，函数是不会执行的。

注意: find() 并没有改变数组的原始值。

测试函数的语法为：

用法：

array.find(function(currentValue, index, arr),thisValue)  
参数：  
  
*currentValue 必需。当前元素*

*index 可选。当前元素的索引值*  
*arr 可选。当前元素所属的数组对象*  
*thisValue 可选。 传递给函数的值一般用 "this" 值。*  
*如果这个参数为空， "undefined" 会传递给 "this" 值*

例子:

*var ages = [4, 12, 16, 20];  
function checkAdult(age) {   
　　return age >= 18;   
}   
function myFunction() {   
　　document.getElementById("demo").innerHTML = ages.find(checkAdult);   
}*

这里checkAdult函数便是测试函数， 他有一个参数也就是必须参数age, 轮询ages时，每一个ages中的某一元素， 如果大于18数字则返回true, **那么find函数返回第一个符合条件的元素的值**。

**Map函数（用于轮询改变数组中的元素并放入新数组中）**

在Javascript中，我们可以用JS数组的API, map函数对**数组中**的每个元素执行一次自定义的函数。

自定义的函数第一个参数是数组的每一个元素，**并且每一次函数的返回值放入新的数组**当中。对所有数组元素轮询完后将新的数组作为Map函数的返回。

例如：

var numbers = [1, 2, 3, 4];  
newArray= numbers.map(x => {

return x\*x

});

// newArray = [1, 4, 9, 16]

**Reduce函数（从一个数组中轮询，整合出一个结果）**

语法：

array.reduce(callbackfn,[initialValue]);

其中回调函数格式为：

function callbackfn(preValue,curValue,index,array){}

preValue：上一次调用回调函数返回的值，或者是开始时提供的初始值(initialValue)  
curValue：数组中当前被处理的数组项  
index：当前数组项在数组中的索引值  
array：调用reduce()方法的数组

回调函数第一次执行时，preValue和curValue可以是一个值，如果initialValue在调用reduce()被提供，那么第一个preValue就等于initialValue，如果initialValue没有被提供，那么preValue等于数组中的第一个值，curValue等于数组中的第二个值。

例如：

var arrTest = [1,2,3,4,5];

arrTest.reduce(function(preValue,curValue){

return preValue + curValue;

}) //初始值为0, 结果为15

var arrTest = [1,2,3,4,5];

arrTest.reduce(function(preValue,curValue){

return preValue + curValue;

}, 10) //初始值为10, 结果为25