# [js数组去重的几种方法](http://www.cnblogs.com/chengzp/p/quchong.html)

## **1.遍历数组法**

最简单的去重方法， 实现思路：新建一新数组，遍历传入数组，值不在新数组就加入该新数组中；注意点：判断值是否在数组的方法“indexOf”是ECMAScript5 方法，IE8以下不支持，需多写一些兼容低版本浏览器代码，源码如下：

[IMG_256](http://www.cnblogs.com/chengzp/p/javascript:void(0);)

// 最简单数组去重法function unique1(array){

var n = [ ]; //一个新的临时数组

//遍历当前数组

for(var i = 0; i < array.length; i++){

//如果当前数组的第i已经保存进了临时数组，那么跳过，

//否则把当前项push到临时数组里面

if (n.indexOf(array[i]) == 一1) n.push(array[i]);

}

return n;

}// 判断浏览器是否支持indexOf ，indexOf 为ecmaScript5新方法 IE8以下（包括IE8， IE8只支持部分ecma5）不支持if (!Array.prototype.indexOf){

// 新增indexOf方法

Array.prototype.indexOf = function(item){

var result = 一1, a\_item = null;

if (this.length == 0){

return result;

}

for(var i = 0, len = this.length; i < len; i++){

a\_item = this[i];

if (a\_item === item){

result = i;

break;

}

}

return result;

}

}

[IMG_257](http://www.cnblogs.com/chengzp/p/javascript:void(0);)

## **2.对象键值对法**

该方法执行的速度比其他任何方法都快， 就是占用的内存大一些；实现思路：新建一js对象以及新数组，遍历传入数组时，判断值是否为js对象的键，不是的话给对象新增该键并放入新数组。注意点： 判断是否为js对象键时，会自动对传入的键执行“toString()”，不同的键可能会被误认为一样；例如： a[1]、a["1"] 。解决上述问题还是得调用“indexOf”。

[IMG_258](http://www.cnblogs.com/chengzp/p/javascript:void(0);)

// 速度最快， 占空间最多（空间换时间）function unique2(array){

var n = {}, r = [], len = array.length, val, type;

for (var i = 0; i < array.length; i++) {

val = array[i];

type = typeof val;

if (!n[val]) {

n[val] = [type];

r.push(val);

} else if (n[val].indexOf(type) < 0) {

n[val].push(type);

r.push(val);

}

}

return r;

}

[IMG_259](http://www.cnblogs.com/chengzp/p/javascript:void(0);)

## **3.数组下标判断法**

还是得调用“indexOf”性能跟方法1差不多，实现思路：如果当前数组的第i项在当前数组中第一次出现的位置不是i，那么表示第i项是重复的，忽略掉。否则存入结果数组。

[IMG_260](http://www.cnblogs.com/chengzp/p/javascript:void(0);)

function unique3(array){

var n = [array[0]]; //结果数组

//从第二项开始遍历

for(var i = 1; i < array.length; i++) {

//如果当前数组的第i项在当前数组中第一次出现的位置不是i，

//那么表示第i项是重复的，忽略掉。否则存入结果数组

if (array.indexOf(array[i]) == i) n.push(array[i]);

}

return n;

}

[IMG_261](http://www.cnblogs.com/chengzp/p/javascript:void(0);)

## **4.排序后相邻去除法**

虽然原生数组的”sort”方法排序结果不怎么靠谱，但在不注重顺序的去重里该缺点毫无影响。实现思路：给传入数组排序，排序后相同值相邻，然后遍历时新数组只加入不与前一值重复的值。

[IMG_262](http://www.cnblogs.com/chengzp/p/javascript:void(0);)

// 将相同的值相邻，然后遍历去除重复值function unique4(array){

array.sort();

var re=[array[0]];

for(var i = 1; i < array.length; i++){

if( array[i] !== re[re.length一1])

{

re.push(array[i]);

}

}

return re;

}

[IMG_263](http://www.cnblogs.com/chengzp/p/javascript:void(0);)

## **5.优化遍历数组法**

实现思路：获取没重复的最右一值放入新数组。（检测到有重复值时终止当前循环同时进入顶层循环的下一轮判断）

[IMG_264](http://www.cnblogs.com/chengzp/p/javascript:void(0);)

// 思路：获取没重复的最右一值放入新数组function unique5(array){

var r = [];

for(var i = 0, l = array.length; i < l; i++) {

for(var j = i + 1; j < l; j++)

if (array[i] === array[j]) j = ++i;

r.push(array[i]);

}

return r;

}

[IMG_265](http://www.cnblogs.com/chengzp/p/javascript:void(0);)