**二维数组去重**

**将二维数组转换成一位数组去比较，将二级元素转换为字符串比较方便，但是为了区分类型，引入typeof拼接，之后对这个一维数组查重**

[[a,1],[b,2],[a,2]] 去除[a,2]

**function fun(input) {**

**var input = input || [];**

**var temp = input.map(function (item) {**

**return item.map(function (\_item) {**

**return \_item + ' ' + typeof \_item; }).join('|'); }); // temp : ['a string|1 number', 'b string|2 number', 'a string|2 number']**

**var indexArr = [];**

**temp.map(function (item, index, arr) {**

**for (var i = 0; i < index; i++) {**

**if (arr[i] !== item) indexArr.push(index); } });**

**var result = [];**

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

**result.push(input[i]); }**

**JavaScript递归实现多维数组去重**

**/\*\* \* 思路：获取没重复的最右一值放入新数组 \* 从第一个值开始，将其与后面的值依次比较， \* 如若相等则终止当前循环并进入顶层循环的下一轮判断 \*/**

**而多维数组的去重就是在一维数组上增加一个递归，因为我们起初并不知道多维数组的维度。具体代码如下：**

**var temp = []**

**function uniq(array){**

**var result = []**

**recursion(array) // 调用递归将多维数组变为一维数组再去重**

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

**for(var j = i + 1; j < len;j ++){**

**if (temp[i] === temp[j]) {**

**i ++**

**j = i } }**

**result.push(temp[i]) }**

**return result**

**} // 新增递归函数**

**function recursion(array){**

**var len = array.length**

**for (var i = 0; i < len ;i ++) {**

**if (typeof array[i] == 'object') {**

**// 如若数组元素类型是object，则递归**

**recursion(array[i]) } else {**

**temp.push(array[i]) // 否则添加到temp数组中 } } }**

**var arr = [1,[2,3],[3,2,[1,6,[3,5,'3']]]] console.log(uniq(arr))**

**return result; }**

**console.log(uniq（arr）);**

**对象,数组和null返回 "object"**