JavaScript实现多维数组、对象数组排序，其实用的就是原生的sort()方法，用于对数组的元素进行排序。

sort() 方法用于对数组的元素进行排序。语法如下：

arrayObject.sort(sortby)

返回值为对数组的引用。请注意，数组在原数组上进行排序，不生成副本。

如果调用该方法时没有使用参数，将按字母顺序对数组中的元素进行排序，说得更精确点，是按照字符编码的顺序进行排序。要实现这一点，首先应把数组的元素都转换成字符串（如有必要），以便进行比较。

如果想按照其他标准进行排序，就需要提供比较函数，该函数要比较两个值，然后返回一个用于说明这两个值的相对顺序的数字。比较函数应该具有两个参数 a 和 b，其返回值如下：

若 a 小于 b，在排序后的数组中 a 应该出现在 b 之前，则返回一个小于 0 的值。  
若 a 等于 b，则返回 0。  
若 a 大于 b，则返回一个大于 0 的值。

[?](http://www.jb51.net/article/67458.htm)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | function NumAscSort(a,b)  {   return a - b;  }  function NumDescSort(a,b)  {   return b - a;  }  var arr = new Array( 3600, 5010, 10100, 801);  arr.sort(NumDescSort);  alert(arr);  arr.sort(NumAscSort);  alert(arr); |

sort(fun)接受了个排序规则函数，这个函数将比较2个数字的大小。而我们的对象数组排序，实际上原理也是一样的。  
如果不比较数字的大小，则可以这样：

[?](http://www.jb51.net/article/67458.htm)

|  |  |
| --- | --- |
| 1  2 | var myarray=["Apple", "Banana", "Orange"]  myarray.sort() |

数组直接调用sort()后，数组按字母顺序对数组中的元素进行排序，说得更精确点，是按照字符编码的顺序进行排序。  
对于对象数组排序，我们先写一个构造比较函数的函数：

[?](http://www.jb51.net/article/67458.htm)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17 | //by函数接受一个成员名字符串做为参数  //并返回一个可以用来对包含该成员的对象数组进行排序的比较函数  var by = function(name){   return function(o, p){     var a, b;     if (typeof o === "object" && typeof p === "object" && o && p) {       a = o[name];       b = p[name];       if (a === b) {         return 0;       }       if (typeof a === typeof b) {         return a < b ? -1 : 1;       }       return typeof a < typeof b ? -1 : 1;     }     else {       throw ("error");     }   }  } |

要排序的数组：

[?](http://www.jb51.net/article/67458.htm)

|  |  |
| --- | --- |
| 1  2  3  4  5 | var employees=[]  employees[0]={name:"George", age:32, retiredate:"March 12, 2014"}  employees[1]={name:"Edward", age:17, retiredate:"June 2, 2023"}  employees[2]={name:"Christine", age:58, retiredate:"December 20, 2036"}  employees[3]={name:"Sarah", age:62, retiredate:"April 30, 2020"} |

直接调用函数：

[?](http://www.jb51.net/article/67458.htm)

|  |  |
| --- | --- |
| 1 | employees.sort(by("age")); |

到这里，对象数组排序就算基本实现了。那如何实现多个键值排序呢？意思就是先是对age排序，如果age相同，再比较name。  
这时，我们可以进一步修改by函数，让其可以接受第二个参数，当主要的键值产生一个匹配的时候，另一个compare方法将被调用以决出高下。

[?](http://www.jb51.net/article/67458.htm)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23 | //by函数接受一个成员名字符串和一个可选的次要比较函数做为参数  //并返回一个可以用来包含该成员的对象数组进行排序的比较函数  //当o[age] 和 p[age] 相等时，次要比较函数被用来决出高下  var by = function(name,minor){   return function(o,p){     var a,b;     if(o && p && typeof o === 'object' && typeof p ==='object'){       a = o[name];       b = p[name];       if(a === b){         return typeof minor === 'function' ? minor(o,p):0;       }       if(typeof a === typeof b){         return a < b ? -1:1;       }       return typeof a < typeof b ? -1 : 1;     }else{       thro("error");     }   }  employees.sort(by('age',by('name'))); |

好了，现在可以放心使用了。如果看不懂，可直接copy 这个by函数到你的应用里面，直接调用即可。