深拷贝和浅拷贝

可以参考这篇文章：<http://jerryzou.com/posts/dive-into-deep-clone-in-javascript/>

不过文章内容实在太复杂，就没看完。

首先深复制和浅复制只针对像 Object, Array 这样的复杂对象的。简单来说，浅复制只复制一层对象的属性，而深复制则递归复制了所有层级。

抛开jQuery，上代码例子。下面是一个简单的浅复制实现：

var obj = { a:1, arr: [2,3] };

var shallowObj = shallowCopy(obj);

function shallowCopy(src) {

var dst = {};

for (var prop in src) {

if (src.hasOwnProperty(prop)) {

dst[prop] = src[prop];

}

}

return dst;

}

玛德。。浅拷贝其实也拷贝了对象，只不过不会递归地拷贝对象内的对象

**借助 JSON 全局对象**

相比于上面介绍的三个库的做法，针对纯 JSON 数据对象的深复制，使用 JSON 全局对象的 parse 和 stringify 方法来实现深复制也算是一个简单讨巧的方法。然而使用这种方法会有一些隐藏的坑，它能正确处理的对象只有 Number, String, Boolean, Array, 扁平对象，即那些能够被 json 直接表示的数据结构。

**function** jsonClone**(**obj**)** **{**

**return** JSON**.**parse**(**JSON**.**stringify**(**obj**));**

**}**

**var** clone **=** jsonClone**({** a**:1** **});**

以上是比较常见的深复制方法，不过有一点缺陷，就是也无法处理某些对象（指object？）

另一个答案中可以深拷贝的方法

原文地址：https://www.zhihu.com/question/23031215

var cloneObj = function(obj){

var str, newobj = obj.constructor === Array ? [] : {};

if(typeof obj !== 'object'){

return;

} else if(window.JSON){

str = JSON.stringify(obj), //系列化对象

newobj = JSON.parse(str); //还原

} else {

for(var i in obj){

newobj[i] = typeof obj[i] === 'object' ? cloneObj(obj[i]) : obj[i];

}

}

return newobj;

};

通过判断对象是object还是array

如果是array的话用for..in来复制（遇到object则递归）

如果是object的话用json来复制