1. 浅拷贝和深拷贝的概念

浅拷贝只复制指向某个对象的指针，而不复制对象本身。深拷贝会创建另一个对象，修改新对象不会改变原对象

1. 如何浅拷贝

Ⅰ等号

* 1. 基本类型：

var num1=5;

var num2=num1;

2个变量不互相影响

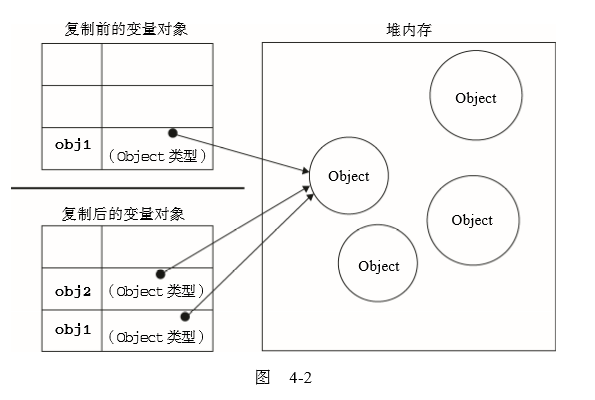


* 1. 引用类型

var obj1=new Object();

var obj2=obj1;

两个变量实际上引用同一个对象，像一个房间的2把钥匙，改变其中一个变量会影响另一个。



ⅡObject.assign()

var target={a:1};

var source1={b:2};

var source2={c:3};

Object.assign(target,source1,source2);

console.log(target);//{a:1,b:2,c:3}

浅拷贝 可枚举属性

1. 如果目标对象与源对象有同名属性，或多个源对象有同名属性，后面的属性会覆盖前面的属性
2. 如果只有一个参数，Object.assign()会直接返回参数
3. 如果参数不是对象则先转为对象，由于undefined和null无法转成对象，如果将它们作为参数会报错。但是如果非对象参数出现在源对象的位置(即非首参数)，那么如果无法转成对象便会跳过，意味着undefined和null不在首参数便会报错。除字符串会以数组形式复制到目标对象，其他值都不会产生效果
4. Object.assign只复制源对象的自身属性(不复制继承属性)，也不复制不可枚举属性(enumerable:false)
5. 属性名为Symbol值的属性也会被复制
6. 嵌套对象+同名属性

var target={a:{b:'c',d:'e'}};

var source={a:{b:'hello'}};

Object.assign(target,source);

console.log(target);//{a:{b:'hello'}}

Object.assign的处理方法是替换不是添加

1. Object.assign()可以用来处理数组，但会把对象当成属性名为0、1、2的对象

var target=[1,2,3];

var source=[4,5];

Object.assign(target,source);

console.log(target);//[4,5,3]

1. Object.assign可以处理一层的深拷贝

var target={a:1,b:2,c:3};

var source=Object.assign({},target);

source.b=100;

console.log(target.b);//2

1. 深拷贝
   1. 手动复制

var obj1={a:1,b:2,c:3};

var obj2={a:obj1.a,b:obj1.b,c:obj1.c};

obj2.b=100;

console.log(obj1.b);//2

麻烦 不是真实的深拷贝

var obj1={body:{a:10}};

var obj2={body:obj1.body};

obj2.body.a=20;

console.log(obj1);//{body:{a:20}}

console.log(obj2);//{body:{a:20}}

console.log(obj1===obj2);//false

console.log(obj1.body===obj2.body);//true

* 1. 如果对象只有一层的话，可以使用Object.assign()
  2. 转成JSON再转回来

var obj1={body:{a:10}};

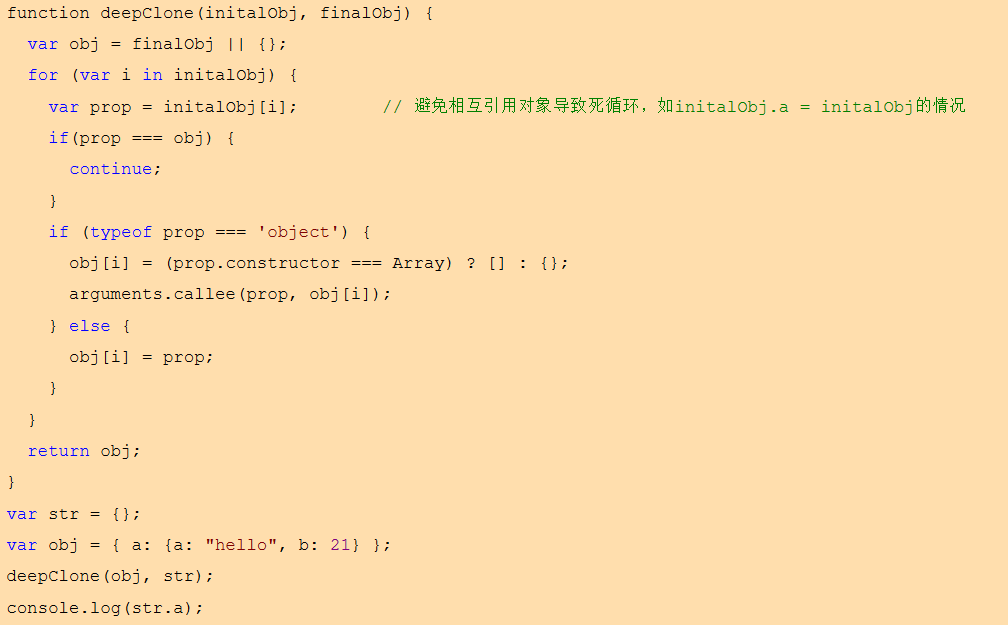
var obj2=JSON.parse(JSON.stringify(obj1));

不管对象原来的构造函数是什么都会变成Object

能正确处理的对象只有Number,String,Boolean,Array,扁平对象，即能够被JSON直接表示的对象

RegExp,function不能用这种方法

* 1. 递归拷贝



* 1. Object.create() 待定
  2. jquery的$.extend()
  3. lodash函数库的\_.cloneDeep()