2019/05/02 10:47

深度克隆与浅克隆（深拷贝与浅拷贝）

简单来说B复制A，A改变，B也改变，说明是浅拷贝（像是新的地址，但还是指向同一个空间），A改变，B不改变，说明是深拷贝（创造出新的一模一样的个体）

浅拷贝方法：

直接用for循环

深拷贝方法：

1. 判断是不是原始值
2. 如果是引用值判断是数组还是对象

   方法：typeof instanceof toString constructor
3. 建立相应的数组或对象
4. 将此当做新的拷贝对象
5. 再次判断原始值还是引用值
6. ……

```
var obj = {
    name : "abc",
    age : 13,
    card : ['visa','master'],
    wife : {
        name : "bcd",
        son : {
            name : "aaa"
        }
    }
}
var obj1 = {

}
function deepClone(origin,target){
    var target = target || {},
    toStr = Object.prototype.toString;
    arrStr = "[object Array]";
    for (var prop in origin[prop]) {
        if (origin.hasOwnProperty(prop)){
            if(origin[prop] !== "null" && typeof(origin[prop]) == 'object'){
                //if (toStr.call(origin[prop]) == arrStr){
                //   target[prop] = [];
```

```
            //}else{
            //    target[prop] = {};
            //}
            target[prop] = toStr.call(origin[prop]) == arrStr? [] : {}
            deepClone(origin[prop],target[prop]);
        }else{
            taget[prop] = origin[prop]
        }
    }
  }
  return target;

}
```

三目运算符

条件判断? 是: 否 并且会返回值

```
var num = 1>0? 2+2 : 1+1;
num = 4
```

数组

如果是负数位时，系统的操作

```
pos += pos>=0? 0 : this.length;
-1+4 = 3
```

构造方式

```
arr = [1,2,3,4,5];
arr = new Array(1,2,3,4,5);
```

(空格也算，打印出来的是undefined[1, ,2])

区别

当Array()中只传入一个参数，那就不是代表数组只一个参数，而是构造出了一个长度为多少的数组

```
eg:arr = new Array(10)        //长度为十的空数组
```

并且Array(10.2)会报错，说明是非法的数组长度 [10.2]就不会

数组是一种特殊的对象，一般就算没有的元素也不会报错，而是打印undefined

数组常用方法

可以改变原数组

push

```
a = [];
a.push(10，9，8，7，6);
//在数组的最后一位添加数据，可以添加多位
```

```
Array.prototype.push = function (){
    for (var i =0; i<arguments.length;i++){
        this[this.length] = arguments[i];
    }
    return this.length;
}
```

pop

```
a = [1,2,3];
var num = a.pop();
num 3;
//将数组的最后一位剪切出去,无参数
```

shift

```
a = [];
var num = a.shift();
//在数组的前面剪切
```

unshift

```
a = [];
a.unshift(10，9，8，7，6);
//在数组的前面增加
```

reverse

```
a = [1,2,3];
a.reverse();
//将数组颠倒（并不排序）
a [3,2,1]
```

splice

```
a = [1,2,3,4,5,6];
a.splice(1，2);
[2,3]
arr [1,4,5,6]
//a.splice(从第几位开始，截取多少的长度，在切口处添加新的数据(可以填无穷位))
//返回截取出的片段
//最后一个参数相当于用新的数据替换截取出来的数据
arr = [1,2,3,5];
//将4添加进3后面
arr.splice(3.0.4);
```

sort

sort 里面的参数应该是一个比较方法的函数，一般情况下默认是比较字符编码的顺序

```
arr = [1,5,8,2,7,9]
arr.sort()
//在数组原基础上字符排序
//按照ASC码排，如果有10，就当成了2
var arr = [1,3,5,4,10];
不按照ASC码排
```

sort里面的函数的规则
//1,必须写俩形参
//2,看返回值1,当返回值位负数时，那么在前面的数放在前面
//2,为正数时，那么后面的数在前
//3,为0，不动
传参顺序 1 3 1 5 1 4 1 10 3 5 3 4 3 10 5 4 5 10

```
arr.sort(function (a,b){
    //if(a > b) {
      //  return 1;
    //}else {
      //  return -1;
    //}
    return a-b;
})
//乱序排，每次结果都不同
arr.sort(function (){
    return Math.random() - 0.5;
})
```

不可以改变原数组

arr.concat(arr1)

将后面数组拼接到前面

slice(从该位截取，截取到该位)

包括前一个，不包括后一个，不改变原数组

```
var arr = [1,2,3,4,5,6]
arr.concat(1,2)
[2]
```

join()

括号中填入字符串，之后数组会以该字符串为间隔 不传用逗号连

```
var arr = [1,2,3,4,5,6]
arr.join("-");
arr  "1-2-3-4-5-6"
```

split()　　算是字符串的方法

括号中填入字符串，按照该字符串拆分数组

并且会去掉字符串中该字符的部分 ，最后再打包成数组

```
str = "1-2-3-4-5-6"
str.split("-")
["1","2","3","4","5","6"]
str.split("4")
["1-2-3-","-5-6"]
```

类数组

是对象，但是可以当数组一样用

类数组必须写length属性

```
var obj = {
    "0" : 'a',
    "1" : 'b',
    "2" : 'c',
    "length" : 3,
    "push" : Array.prototype.push,
    "splice" : Array.prototype.splice
}
obj.push("c");
obj.push("d");
AFTER
obj {
    2 : 'c',
    3 : 'd',
    length : 4
}
//属性要为索引（数字属性），必须有length
Array.prototype.push = function (target){
    obj[obj.length] = target;
    obj.length ++;
}
```