JS参数的传递和类型的检测

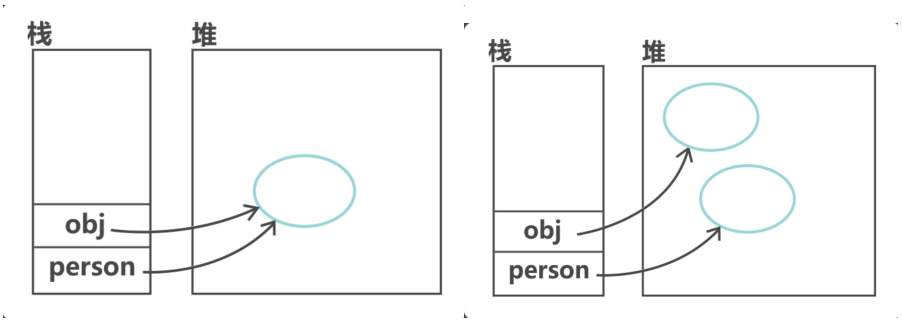
1、参数的传递

分为基本类型的参数传递和引用类型的参数传递。基本类型的参数传递和引用类型的参数传递都是值的传递形式

```
function setName(obj){
    return obj.name = "xm";
}
var person = {};
setName(person);
console.log(person.name);
```

上面的obj为形式参数，person为实际参数，但是person里面储存的值是对象的地址而不是对象的引用，所以对象的传递还是值的传递而不是引用的传递

```
function setName(obj){
    obj.name = "xm";
    obj = {};
    obj.name = "xh";
}
var person = {};
setName(person);
console.log(person.name);//xm
```



刚开始的时候obj和person都是指向一个对象，但是之后在堆中又开辟了一个对象，obj指向那个空对象并添加了name的属性，但是person的指向没有改变，所以person.name还是"xm"。

2、检测类型

typeof方法可以对对基本类型进行检测

instanceof可以判断引用类型具体为什么类型，不能和基本类型连用

```
console.log([] instanceof Array);//true
console.log({} instanceof Object);//true
```

```
typeofundefined//undefined
typeof null//object
typeof true//boolean
typeof 1//number
typeof "star"//string
```

以上都是基本类型的检测

3、作用域

js是没有块级作用域的，也就是说在if和for语句里面定义的变量都是可以被访问的（但是es6是有块级作用域的）

```
function test(){
    for(var i = 0;i<5;i++){
        alert(i); }
        console.log(i);
    }
test();//i=5;
```