



链滴

# Java 中方法参数的传递

作者: [gelvshi](#)

原文链接: <https://ld246.com/article/1486717065419>

来源网站: 链滴

许可协议: [署名-相同方式共享 4.0 国际 \(CC BY-SA 4.0\)](#)

java中方法参数传递看似有两种，值传递和引用传递。而实际上都是值传递。

所谓值传递，传递的是变量的副本就是说，在将变量a作为参数传进方法中时，方法中接收到的是a的副本，在方法中对参数的修改实际上修改的是a的副本，所以不会影响到a的值。

而引用传递，传递的是变量的内存地址。在将变量a作为参数传递给方法时，传递的是a的内存地址，方法中对参数的修改，实际上是修改的a本身。

因为java的变量有两类，一类是基本数据类型，一类是引用类型。所以就把基本数据类型作为参数进传递时，当做了“值传递”；把引用类型作为参数进行传递时，当做了“引用传递”。事实上无论是基本数据类型还是引用类型，java中都是按“值传递”进行操作的。

在引用变量r作为参数传到方法中时，传递的是r的副本。只不过r和r的副本都指向了同一个对象。

```
public class TestParamPassing {

    public static void main(String[] args) {
        int b = 4;
        testValueParam(b);
        System.out.println("==== b : " + b);

        List list = new ArrayList();
        list.add("a");
        list.add("b");
        testReferenceParam(list);
        System.out.println("==== list : " + list);
    }

    public static void testReferenceParam(List list){
        list.add("c");
        list = new ArrayList<>();
        list.add("d");
        System.out.println("----- list : " + list);
    }

    public static void testValueParam(int a){
        System.out.println("==== a : " + a);
        a=3;
        System.out.println("==== a after : " + a);
    }
}
```

执行结果为：

```
==== a: 4
==== a after : 3
==== b : 4
----- list : [d]
==== list : [a, b, c]
```

在方法testValueParam(int a)中，因为是“值传递”，所以传到方法中的是原始值的副本，所以在方

中对a的修改，不会影响到原始值。

在方法testReferenceParam(List list)中，因为是“值传递”，所以传到方法中的是原始值的副本，以两者都指向了同一个对象，因此在方法中对对象的修改，会影响到原始值指向的对象。但是，当在法中，将副本指向了其他的对象，再对副本指向的对象进行修改，就不会影响到原始值所指向的对象。所以在传递引用变量时，也是“值传递”。