



链滴

理解 clone 方法

作者: [figo2young](#)

原文链接: <https://ld246.com/article/1567502848036>

来源网站: 链滴

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

克隆方法

clone是Object的一个方法，但是要支持clone方法，需要对象实现Cloneable接口，实现了cloneable接口后，可以使用clone方法。使用clone方法可以实现对象的克隆。

克隆的类型——浅克隆(ShallowClone)和深克隆(DeepClone)。

- 浅克隆是指拷贝对象时仅仅拷贝对象本身（包括对象中的基本变量），而不拷贝对象包含的引用指向的对象。
- 深克隆不仅拷贝对象本身，而且拷贝对象包含的引用指向的所有对象。

请看下面的例子：

```
package core;

/**
 * @author 关注微信公众号：Java学习交流社区
 * （更多技术干货、原创文章、面试资料，群聊吹水一应俱全）
 *
 * 测试clone方法
 */
public class CloneDemo {

    public static void main(String[] args) throws CloneNotSupportedException {
        A a = new A();
        B b = new B();
        a.setB(b);
        A cloneA = (A)a.clone();
        a.getB().setS("ss");
        System.out.println(a == cloneA);
        System.out.println(b == cloneA.getB());
        System.out.println(cloneA.getB().getS());
    }
}

class A implements Cloneable{

    private B b;

    public B getB() {
        return b;
    }

    public void setB(B b) {
        this.b = b;
    }

    @Override
    protected Object clone() throws CloneNotSupportedException {
        return super.clone();
    }
}
```

```

    }
}

class B implements Cloneable{

    String s = "s";
    @Override
    protected Object clone() throws CloneNotSupportedException {
        return super.clone();
    }

    public String getS() {
        return s;
    }

    public void setS(String s) {
        this.s = s;
    }
}

```

运行结果:

```

false
true
ss

```

可以看出:

1. 调用clone方法返回的cloneA与原对象A进行==比较, 返回的是false, 即两个引用指向不同的对象
2. 调用clone方法返回的cloneA中的属性b与原对象的b进行==比较, 返回的是true, 即两个引用指向相同的对象

这种情况就是俗称的浅克隆, 如果对象中所有的非基本类型属性都进行clone, 即为深克隆。

常用的浅克隆方法

Apache BeanUtils、PropertyUtils, Spring BeanUtils, Cglib BeanCopier (推荐使用, 实测速度最快)

常用的深克隆方法

实现 Serializable 接口进行序列化和反序列化, 各种Json格式的序列化和反序列化

```

<hr>
<div style="text-align:center">
</img>
</div>
<br>
<span>扫码关注微信公众号《Java学习交流社区》, 即可获取原创《Java并发编程与高并发解决方
》+《Redis》思维导图</span>
<br>
作者: <a href="https://www.jvscn.cn" target="_blank">Java学习交流社区</a> <br>

```

地址: https://www.jvsc.cn/articles/2019/08/30/1567158083430.html
