

转

Java静态属性与静态方法能否被继承的问题

2017年03月27日 23:25:21

Carmelo\_Z

阅读数：2788

更多

开始正文之前的一个参考：

kao331431214

12-12 19:22

等级

4楼

static 的属性和方法都是不能覆盖的，因为static修饰的部分都是与对象无关的，只与类有关

例如你上述的例子：  
如果你用  
Father f=new Son();  
f.f();//这里调用的还是Father.f()的方法，与直接用Father.f()调用的结果相同，因为f是Father类型的

如果这样调用：  
Son s=new Son();  
s.f();//这里调用的就是Son.f()的方法了，因为s是Son类型的

也就是说static修饰的部分是不能被覆盖、重写的

2

0

kao331431214

12-12 19:22

等级

5楼

就是说两个类的同名同参数的静态方法是分属于两个类的，各不相干，只是名字相同而已

我认为结论应该是这样的：  
子类可以继承父类的静态属性和静态方法,但是不可以重写静态方法。  
即使在子类中有一个与父类具有相同方法标签(方法名相同,参数类型相同,返回值类型相同,甚至连访问级别也相同),也只能说明是对子类又新建了一个与父类同名的静态方法,而不是对父类的静态方法的重写。  
  
我的结论:子类可以继承父类的静态属性和静态方法,但是不可以重写静态方法。  
希望朋友们帮我证实,分不够再加!

答：**完全正确**，不用再证实了。只是有个条件：父类的这些，子类要首先能访问，若不能访问（如：private），则什么都谈不上了。

原文链接：点击打开链接

结论：java中静态属性和静态方法可以被继承，但是没有被重写(overwrite)而是被隐藏。

原因：

- 1). 静态方法和属性是属于类的，调用的时候直接通过类名.方法名完成对，不需要继承机制及可以调用。如果子类里面定义了静态方法和属性，那么这时候：静态方法被“隐藏”。如果你想要调用父类的静态方法和属性，直接通过父类名.方法或变量名完成，至于是否继承一说，子类是有继承静态方法和属性，但是跟实例方法和属性不太一样，两种情况。
- 2). 多态之所以能够实现依赖于继承、接口和重写、重载（继承和重写最为关键）。有了继承和重写就可以实现父类的引用指向不同子类的对象。重写的功能是：“重写”后子于父类的优先级，但是“隐藏”是没有这个优先级之分的。
- 3). 静态属性、静态方法和非静态的属性都可以被继承和隐藏而不能被重写，因此不能实现多态，不能实现父类的引用可以指向不同子类的对象。非静态方法和属性不能被继承和重写，因此可以实现多态。

例子：

```
1 package com.study.test;
2
3 public class A { //父类
4     public static String staticStr = "A静态属性";
5     public String nonStaticStr = "A非静态属性";
6     public static void staticMethod(){
7         System.out.println("A静态方法");
8     }
9     public void nonStaticMethod(){
10        System.out.println("A非静态方法");
11    }
12 }
```

https://blog.csdn.net/Carmelo\_Z/article/details/67086140

1/8

```
1 package com.study.test;
2
3 public class B extends A{//子类B
4     public static String staticStr = "B改写后的静态属性";
5     public String nonStaticStr = "B改写后的非静态属性";
6     public static void staticMethod(){
7         System.out.println("B改写后的静态方法");
8     }
9 }
```

```
1 package com.study.test;
2
3 public class C extends A{//子类C继承A中的所有属性和方法
4
5 }
```

```
1 package com.study.test;
2
3 public class StaticExtendsTest {
4
5     public static void main(String[] args) {
6         C c = new C();
7         System.out.println(c.nonStaticStr);
8         System.out.println(c.staticStr);
9         c.staticMethod();//输出的结果都是父类中的非静态属性、静态属性和静态方法,推出静态属性和静态方法可以被继承
10
11         System.out.println("-----");
12
13         A c1 = new C();
14         System.out.println(c1.nonStaticStr);
15         System.out.println(c1.staticStr);
16         c1.staticMethod();//结果同上,输出的结果都是父类中的非静态属性、静态属性和静态方法,推出静态属性和静态方法可以被继承
17
18         System.out.println("-----");
19         B b = new B();
20         System.out.println(b.nonStaticStr);
21         System.out.println(b.staticStr);
22         b.staticMethod();
23
24         System.out.println("-----");
25         A b1 = new B();
26         System.out.println(b1.nonStaticStr);
27         System.out.println(b1.staticStr);
28         b1.staticMethod();//结果都是父类的静态方法,说明静态方法不可以被重写,不能实现多态
29     }
30
31 }
```

现在才知道，零基础学习高级Java后，年薪可以这么多！

零基础学IT选Java，易学、高薪、前景广，100万人才缺口，互联网必备人才



想对作者说点什么



weixin\_40677431：厉害了，解决了我多态和静态方法继承的疑难了（3个月前 #2楼）



sili0816：这相当于编译时多态和运行时多态的区别吗？（4个月前 #1楼）