Polymorphism and Static Methods

Asked 10 years, 9 months ago Modified 3 years, 4 months ago Viewed 12k times

14

I have a question about this code right here

```
public Car {
    public static void m1(){
        System.out.println("a");
    }
    public void m2(){
        System.out.println("b");
    }
}

class Mini extends Car {
    public static void m1() {
        System.out.println("c");
    }
    public void m2(){
        System.out.println("d");
    }
    public static void main(String args[]) {
        Car c = new Mini();
        c.m1();
        c.m2();
   }
}
```

I know that polymorphism does not work with static methods, only to instance methods. And also that overriding doesn't work for static methods.

Therefore I think that this program should print out: c, d

Because c calls the m1 method, but it's static, so it can't override and it calls the method in class Mini instead of Car.

Is this correct?

However, my textbook says that the answer should be : a, d

is it a typo? Because I'm a little confused right now.

Please clear this up, thanks.

java static polymorphism

Share Follow

edited Sep 6, 2017 at 18:50

Sotirios Delimanolis
274k 60 697 726

asked Dec 4, 2012 at 4:16

Evolutionary High
1,257 4 14 14

Sorted by:

2 Answers

Sorted by: Highest score (default)

36

Because c calls the m1 method, but it's static, so it can't override and it calls the method in class Mini instead of Car.

That's exactly backwards.

`c` is *declared* as `Car`, so static method calls made through `c` will call methods defined by `Car`.
The compiler compiles `c.m1()` directly to `Car.m1()`, without being aware that `c` actually holds a `Mini`.

This is why you should never call static methods through instance like that.

Share Follow

answered Dec 4, 2012 at 4:19

SLaks
869k 176 1909 1967

6 +1 For *This is why you should never call static methods through instance like that.* – Joffrey Apr 16, 2014 at 7:43

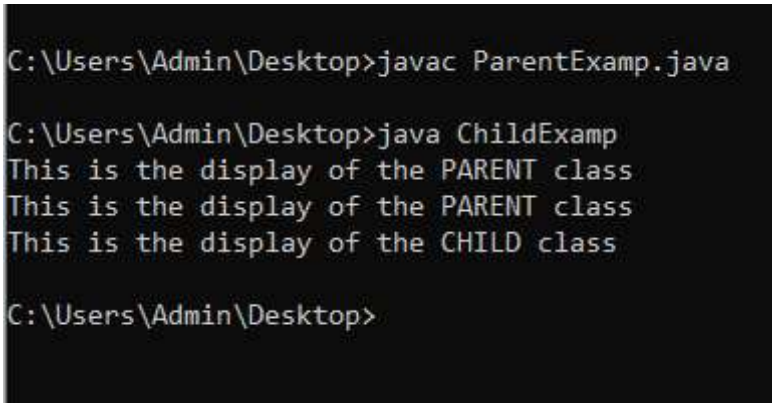
Marked:The compiler compiles c.m1() directly to Car.m1() – linjiejun Oct 25, 2020 at 13:15

3

I faced the same issue while working with Inheritance. What I have learned is if the method being called is Static then it will be called from the class to which the reference variable belongs and not from the class by which it is instantiated.

```
public class ParentExamp
   {
    public static void Displayer()
    {
     System.out.println("This is the display of the PARENT class");
    }
   }

    class ChildExamp extends ParentExamp
   {
       public static void main(String[] args)
       {
         ParentExamp a  = new ParentExamp();
         ParentExamp b  = new ChildExamp();
         ChildExamp  c  = new ChildExamp();

         a.Displayer(); //Works exactly like ParentExamp.Displayer() and Will
                        call the Displayer method of the ParentExamp Class

         b.Displayer();//Works exactly like ParentExamp.Displayer() and Will
                        call the Displayer method of the ParentExamp Class

         c.Displayer();//Works exactly like ChildExamp.Displayer() and Will
                        call the Displayer method of the ChildtExamp Class
```

```
    }
    public static void Displayer()
    {
     System.out.println("This is the display of the CHILD class");
    }
}
```



Share Follow

answered May 16, 2020 at 18:41

Amrit
135 1 5