

## Re: Multiple level base calls

**Source:**

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.languages.csharp/2005-02/5433.html>

---

**From:** Bruce Wood (*brucewood\_at\_canada.com*)

**Date:** 02/21/05

Date: 21 Feb 2005 09:54:43 -0800

Hmmm... code with no explanation, and it's code that doesn't even compile!

I understand the implication: that declaring PrintMe() in classes A, B, and C will cause "Original A" to print out if you cast a reference of type "C" to type "A". I assume that you meant to say that:

```
C a = new C();  
//Invoking Class C Method  
MessageBox.Show(a.PrintMe());  
//Casting with Class A Method  
MessageBox.Show(((A)a).PrintMe());
```

will display two message boxes: one that says, "Overridden C" and the next that says, "Original A". Unfortunately, if you had tried to compile this code, you would have found that it wouldn't, because you must make A's PrintMe "virtual", and specify either "override" or "new" on B and C's PrintMe.

The behaviour you are claiming will happen only if you declare the PrintMe methods in B and C to be "new". "new," however, isn't an override, so you won't get the polymorphic behaviour that the OP wants.

If you were to specify "override," you would get the behaviour that the OP wants, but both message boxes would display "Overridden C." That, in a nutshell, was the OP's problem: how to have both message boxes display "Overridden C," but then go back to call the base method ("Original A") on demand.

Unfortunately, no can do (in C#).