

# Re: Provide access to a class without access to the constructor

---

*Source:*

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.languages.csharp/2008-04/msg01506.html>

---

- *From:* "Ignacio Machin ( .NET/ C# MVP )" <[ignacio.machin@xxxxxxxxxx](mailto:ignacio.machin@xxxxxxxxxx)>
  - *Date:* Thu, 10 Apr 2008 10:16:20 -0700 (PDT)
- 

On Apr 10, 11:43 am, Weeble <[clockworks@xxxxxxxxxx](mailto:clockworks@xxxxxxxxxx)> wrote:

Suppose I have a class A that works with instances of class B. I would like it so that only an instance of A can generate instances of B, but I would still like users of A to be able to handle instances of B, including passing them as arguments to methods on A. Is there any way to achieve this? Ideas so far:

Make B public, but make its constructor internal. This is okay, but not great. It stops code in other assemblies from constructing instances of B, but I'd rather prevent *\*everything\** outside of A from constructing instances of B.

Have a public interface, IB, and make B a private class inside of A. This is not good because other code can still create instances of IB, which might not behave at all like B, and pass them as arguments to methods on A.

Give B a private constructor, and put A inside B. This works, but it's *\*weird\**, and becomes horrible if I want to have multiple classes like B, because it requires A to be inside all of them, so they end up like russian dolls.

Not really, if the class B is declared as private you do not have that problem:

```
public class A {  
  
    public IB GetB() { return new B(); }  
    class B: IB {  
        public string getIt() { return "hello"; }  
    }  
  
}  
  
public interface IB { string getIt();}  
.
```