

Re: C# inheritance broken?

Source:

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.languages.csharp/2007-01/msg04272.html>

- *From:* "Bruce Wood" <brucewood@xxxxxxxxxxx>
 - *Date:* 26 Jan 2007 16:16:02 -0800
-

On Jan 26, 11:36 am, "Michael D. Ober" <obermd.@.alum.mit.edu.nospam> wrote:

"Ignacio Machin (.NET/ C# MVP)" <machin TA laceupsolutions.com> wrote in message news:%231AcH4XQHHA.496@xxxxxxxxxxxxxxxxxxxxxxxxxxxx

Hi,

"Mythran" <kip_pot...@xxxxxxxxxxx> wrote in message news:e8FyEvXQHHA.4896@xxxxxxxxxxxxxxxxxxxxxxxxxxxx

|
|

| <gro...@xxxxxxxxxxx> wrote in message news:1169837076.386389.190580@xxxxxxxxxxxxxxxxxxxxxxxxxxxx

| > Ignacio,

| >

| > Thank you for your thoughts.

| >

| > The Document class does have a public default constructor, and it is
| > not sealed. I have no problem creating a "new" Document object, or a
| > "new" MyDocument object, for that matter. What I can't do is
| > "convert"

| > an existing Document object (such as the one returned by
| > Document.Load)

| > into a MyDocument.

As I said in another post, giving the struct you are presenting you would have the same issue no matter what language you use. You cannot treat a base

Re: C# inheritance broken?

class as a derived class, you can do the opposite though.

In your case you will have to implement your own Load method. I would first check though if Document provide a Load method instance. or maybe a Clone method

--

Ignacio Machin

machin AT laceupsolutions com This isn't a C# issue. It is a dotNet issue as I have run into the same

issue of inheriting a base class (in my case, VB 2005 and StringCollection) and providing a method for the base class object to become the object at the heart of the derived class. This is a design limitation in the framework itself. From a business logic perspective, it should be doable.

For the record, no language that I know of lets you do this. C++ allows you to pretend to do it via an unsafe cast, but you're not really getting a derived object, just playing with a base class object as though it were a derived object, which is fodder for disaster unless you keep careful track of what you're doing.

You can't use a base class instance as though it were a derived class instance because the latter may have more fields than the former, and so the base class instance doesn't have enough space to store the state of the derived class.

There are various strategies for using a base class instance as a model for creating a derived class. Most of them, however, require some coding. There's no "automatic" way of copying state from one instance to another, which might be a nice addition.

.