

Re: MemberwiseClone() doesn't like arrays?

Source:

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.languages.csharp/2008-03/msg01062.html>

- *From:* Ade <madge@xxxxxxxxxxxx>
 - *Date:* Mon, 10 Mar 2008 03:37:14 -0700 (PDT)
-

On 10 Mar, 10:24, Marc Gravell <marc.grav...@xxxxxxxxxx> wrote:

Had to struggle to remember...

First – we were talking about arrays, not ArrayList – there is a **huge** difference.

Oops, my bad for not reading the first post properly :x

To clone an ArrayList you would call Clone() – this will internally worry about "doing the right thing" i.e. creating a separate array. However, this can't know about your new type, so you will need to write Clone yourself, as below.

OK, I figured I could go that route, but I **Really** want to avoid that as I have lots of classes inheriting from ArrayList, each with lots of members. Further, I have lots of classes inheriting from those classes, adding more members still. I was hoping for a more-readily maintained solution, more along the lines of reflection, which could say something like

```
<pseudocode>
public override object Clone()
{
    ThingList clonedThingList = new ThingList();
    foreach(Thing th in this)
    {
        clonedThingList.Add(th);
    }
    foreach(member_of_ThingList somemember)
    {
        clonedThingList.somemember = this.somemember;
    }
    return clonedThingList;
}
```

Re: MemberwiseClone() doesn't like arrays?

</pseudocode>

where the second foreach is, say, using reflection to find each member aggregated in, such as ThingList.Caption, ThingList.Text, ThingList.X, ThingList.Y, etc or whatever.

Am I talking utter nonsense?

Second; yes – the question of deep vs shallow copy is one that I myself commented on:

Yep, followed all that and I *want* the items in my cloned arraylist to be shallow copies; I'm more than happy with that bit.

Me: <quote>

The point I was trying to make, however, is that if you have a class inside the array, you might actually want to clone that too! Or you might not, it depends on the scenario.</quote>

By default you have a shallow copy; this is expected. If you want a deep clone you'll have to do it yourself. In 2.0 you could perhaps insist that the items are cloneable.

Yep, fine with all that.

By the way, I swapped the public field for a property; you might also want to avoid using ArrayList if you are in 2.0; generics and things like List<T>/Collection<T> largely deprecate ArrayList.

I intend to keep ArrayList for a variety of reasons, not least because there's a stack of code already in place. However, I also need something to build down to .net CF 1.1 (although the clone functionality need not).

<snip>

Thanks

Re: MemberwiseClone() doesn't like arrays?

Re: MemberwiseClone() doesn't like arrays?