

## Re: A small problem comparing types

**Source:**

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.languages.csharp/2004-10/2335.html>

---

**From:** Jon Skeet [C# MVP] ([skeet\\_at\\_pobox.com](mailto:skeet_at_pobox.com))

**Date:** 10/12/04

Date: Tue, 12 Oct 2004 11:12:54 +0100

Simon Harvey <[sh856531@microsofts\\_free\\_email\\_service.com](mailto:sh856531@microsofts_free_email_service.com)> wrote:

> *I hope someone can help me with the following:*

>

> *I have a number of usercontrols that I've made which provides certain audit  
> functions for any data inserted into it. Each audit control is a descendant  
> of BaseAuditControl. Descendants include AuditableTextControl and  
> AuditableBoolControl.*

>

> *The problem I'm having is determining which control I'm actually dealing  
> with at any one time. I often need to treat them generically for most of a  
> method and then need to implement some logic that is specific to the  
> particular type of audit control.*

>

> *An example:*

>

> *To update the main ui element of my audit control I make a method in my base  
> control. I need to be able to get the type of the current instance that I'm  
> dealing with, via the 'this' keyword, and then compare it to the various  
> types of AuditControl that I have made. Something like this:*

> *if((this.GetType() == (typeof(AuditableTextControl))){*

> *//Do one thing*

> *return false;*

> *}*

>

> *if((this.GetType() == (typeof(AuditableBoolControl))){*

> *//Do another*

> *return false;*

> *}*

That's usually not a good way of working – you should use polymorphism instead, usually.

> *The problem I'm getting is that the Types returned from each of the two  
> expressions are different.*

>

> *this.GetType() returns something like ASP.AuditableTextControl\_ascx  
> typeof(AuditableTextControl) returns*

> <mynamespace>.controls.AuditableTextControl

Right – that's because the control itself is generated from the ascx file, and (I believe – I haven't created any user controls myself) only derives from your actual user control class.

> *I don't understand whats going on behind the scenes, but it seems to me that*  
> *when you get the type of an actual object instance, it is different from*  
> *trying to get the type using just the class definition.*  
>  
> *If anyone can help me figure out how to get a successful comparison between*  
> *them. I've tried to make a fake instance of an AuditableTextControl and then*  
> *tried to get the type of that, but that doesnt work either. Seems a bit*  
> *messy as well. I'm sure that there is a really easy way to day this*

Using the more normal form of type comparison – the "is" operator – may well work, if the ...\_ascx type really does inherit from AuditableTextControl.

```
if (this is AuditableTextControl)
{
...
}
```

Once again though, I'd suggest using polymorphism – define a virtual (or abstract) method in the base control, and override it in each derived class.

--  
Jon Skeet - <skeet@pobox.com>  
<http://www.pobox.com/~skeet>  
If replying to the group, please do not mail me too