

Re: How to pass information, classes between forms in Windows Application mode

Source:

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.languages.csharp/2008-07/msg02410.html>

- *From:* Jon Skeet [C# MVP] <skeet@xxxxxxxx>
 - *Date:* Mon, 21 Jul 2008 21:34:48 +0100
-

raylopez99 <raylopez99@xxxxxxxx> wrote:

Keywords: scope resolution, passing classes between parent and child forms, parameter constructor method, normal constructor, default constructor, forward reference, sharing classes between forms.

Here is a newbie mistake that I found myself doing (as a newbie), and that even a master programmer, the guru of this forum, Jon Skeet, missed! (He knows this I'm sure, but just didn't think this was my problem; LOL, I am needling him)

I still don't think it's your problem. You're still confused – and what's more you've changed the problem from "accessing a variable" to "information passing".

If you want to pass information between two forms comprised of classes, whether or not they are parent/child, modal or modeless, dialog or non-dialog you have to use the non-normal/ non-default or parametrized or parameter constructor, not a default (no parameter) normal constructor. Explanation below.

Certainly if you want to pass information from one context to a new object, using a parameterised constructor is the way to go. That has nothing to do with being able to access another type's variables though.

Just passing a parameter doesn't mean that Form2 is magically able to see Form1.myClass1. Form2's constructor will be able to see the parameter, which starts off with the same value as Form1.myClass1, but you can't actually refer to the Form1.myClass1 variable from Form2.

<snip>

Re: How to pass information, classes between forms in Windows Application mode

The point being: you need a forward reference if the Class1 is not 'nested' in the Form1 {}, that is, if Class1 is part of a separate translation unit or module, even if Class1 is in the same namespace as Form1.

No. There is no such thing as a forward reference in C#, and what you seem to think is a forward reference is actually just a variable declaration.

—

Jon Skeet – <skeet@xxxxxxxx>

Web site: <http://www.pobox.com/~skeet>

Blog: <http://www.msmvps.com/jon.skeet>

C# in Depth: <http://csharpindepth.com>

.