

## Re: How to pass information, classes between forms in Windows Application mode [ FINALLY!]

---

*Source:*

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.languages.csharp/2008-07/msg02702.html>

---

- *From:* raylopez99 <raylopez99@xxxxxxxx>
  - *Date:* Thu, 24 Jul 2008 05:58:51 -0700 (PDT)
- 

On Jul 23, 11:18 pm, "Jon Skeet [C# MVP]" <sk...@xxxxxxxx> wrote:

That will generate CS0136. If you remove either of the local variable declarations, it will compile. There will be warnings because we're not actually *\*using\** the variable, but it's only when there are multiple local variable declarations that you see CS0136 – and that would occur even without the instance variable existing at all.

Hope that helps,  
Jon

Yes, it was very helpful. What I concluded is that there are canonical ways of setting up a program (aka "good programming style", or, "what has worked before") to compile and to run correctly with few runtime errors, and these canonical ways should be followed if you want to avoid making mistakes. I suspect the reason you don't get more questions like mine is that people use canonical template programs to build from, typically a working program from a cubicle neighbor or fellow programmer.

Some things I got from this last post of yours:

\*\*

this" will solve it to refer to variables, but if there's ambiguity in the class names themselves, using the namespace is usually enough to fix it. If it's still not (and there can be times where it's not enough) you can use `global::Namespace.Goes.Here.Classname`

[that Namespace only gets rid of ambiguity in class names, rather than ambiguity in instantiations of classes, aka 'variables', which is handled by the 'this' pointer and/or by setting up your declarations/code in the correct sequential order; I didn't know that about Namespace]

\*\*

and,

\*\*

One hint: put a call to `Console.ReadLine()` at the end of your `Main` method so you can see what's happened before the program terminates [easier than stepping through the program line by line from the debugger I suppose]

\*\*

and,

\*\*

```
MyDialogForm01 dlgForm = new MyDialogForm01();  
//stuff deleted here  
dlgForm = null; // at end of code; not strictly needed, apparently  
some form of manual garbage collection
```

Avoid code like that. Setting a local variable to null at the end of the method has no effect, and certainly doesn't trigger garbage collection. In fact the object could have been collected before that line of code, if nothing else references it.

(There are some situations involving captured variables where setting a local variable to null *would* have an effect – probably an unintended one – but that's a more complication situation. I mention it only for completeness.)

[Shows me that even book examples are 'wrong' or bad style—since this was from a book example]

\*\*

and,

Yup. Using `"this.myClass1"` means "I want to refer to the instance variable `myClass1` belonging to `"this"` object, instead of a local variable which happens to also have the name `myClass1`."

[which raises the question what happens when the `this.variable` also happens to have a name that's the same with another variable outside of the local variable? I hope the compiler would catch that, unless there's another `'this.this.myClass1'` you can use. On the other hand, if you stick to 'good programming style' aka what I call 'canonical form' programming, you will avoid using the same names unless you specifically intend to hide/shadow a method/instance/variable, such as in inheritance where you 'override' or 'hide' a base class method in the derived class.]

Re: How to pass information, classes between forms in Windows Application mode [ FINALLY!]

All in all, this thread reinforced my prejudice that you should stick to canonical ways of building programs—learned through trial and error—rather than understand, as you apparently have, the inner workings of the C# language. Unless you plan to write a book on C#.

And I will buy your book, which I see as a C# Scott Meyers imitation, whether you like it or not... :-)

I dare say I buy more programming books a year than most professional programmers, who by and large are seat of the pants types that don't learn too much theory outside of school. In fact, looking at most posts here, it seems posters are interested in fixing their buggy code than asking more fundamental questions, which I suspect some of the more experienced posters here mistake for sophistication, LOL, but I digress.

Cheers,

RL

.