

Re: Managing a project as it scales

Source: <http://www.tech-archive.net/Archive/VC/microsoft.public.vc.mfc/2005-12/msg02066.html>

- *From:* "Doug Harrison [MVP]" <dsh@xxxxxxx>
 - *Date:* Wed, 28 Dec 2005 21:54:28 -0600
-

On Wed, 28 Dec 2005 03:47:41 -0000, Gerry Quinn
<gerryq@xxxxxxxxxxxxxxxxxxxx> wrote:

>> The key thing to realize is that UpdateData is a blunt instrument – it's
>> all controls that participate in DDX/DDV or nothing. See this message for
>> more on this:
>> <http://groups.google.com/group/microsoft.public.vc.mfc/msg/1beb7cdaf010b0c6>
>
>This is true. But dialogs run at UI speed, so the time consumption is
>not an issue.

I haven't said anything about time consumption.

>So long as updating all controls does no harm (and if it
>does harm, your dialogs are too complicated for users to understand),
>why not do it?

The difficulty is in keeping the member variables in sync with the
controls. I talked about that in the message I linked to above and more so
in this one:

<http://groups.google.com/group/microsoft.public.vc.mfc/msg/c6bf6cbcae34ca00>

<q>

As discussed in the message I linked to above, UpdateData(false) should
only be called when you know for sure all the data variables are valid.
During the lifetime of the dialog, it's possible for the member variables
to get out of sync with the controls, possibly due to validation failing
part of the way through, leaving the variables with a mix of old and new
values.

</q>

>It's hard to think of situations where updating all controls on a
>modern PC would be too slow for a user interface.

Again, that's not my argument. I've even talked about response time for UI
actions and related it to modern machines before:

<http://groups.google.com/group/microsoft.public.vc.mfc/msg/f91df656004b44b8>

Re: Managing a project as it scales

>> >As for GetDlgItem(), I use it all the time. Why not? Suppose you want
>> >to set the text of a read-only text control – GetDlgItem() seems as
>> >good a way of accessing it as any.
>>
>> Besides inconvenience, there are two problems with using GetDlgItem instead
>> of binding variables to controls:
>>
>> 1. GetDlgItem can fail and throw an exception at an indeterminate time, the
>> first time it's called on a control, whereas control variables are all
>> bound during OnInitDialog.
>
>I didn't know it could throw an exception, and I've never seen it. I
>know it can return NULL if the control doesn't exist, but you are in
>trouble anyway if that happens unexpectedly, or if your CButton exists
>but you forgot to call Create().

For a temporary CWnd, MFC has got to allocate memory, enter the window into a handle map, etc.

>> 2. GetDlgItem will return a pointer to a temporary CWnd that isn't
>> necessarily of the desired type. For example, for an edit control
>> identified by IDC_EDIT that isn't bound to a CEdit, GetDlgItem(IDC_EDIT)
>> returns a temporary CWnd* instead of a CEdit*. This can have negative
>> repercussions if CEdit handles a message differently than DefWindowProc, or
>> if you cast it to CEdit* and use parts of it (member variables, virtual
>> functions) that don't exist in CWnd. (NOTE: The MFC CEdit class and other
>> MFC control classes are NOT subject to this problem, but you can still
>> demonstrate the type shenanigans with RTTI.)
>
>But if I use it without knowing what I'm using it on, I invariably just
>want a CWnd* anyway, in order to use SetWindowPos() or something.

Hopefully, the MFC type you would have associated with the HWND using ClassWizard doesn't do anything differently than DefWindowProc does for that window class (in the RegisterClass sense of the word "class").

In addition, all too many MFC examples do things like the following, where IDC_T isn't bound to a T (or anything else):

```
((T*) GetDlgItem(IDC_T))->call_T_function();
```

All too many people blindly follow this bad example and end up casting a temporary CWnd* to a T*. Usually, they get away with it, but if T contains virtual functions or member variables not present in CWnd, you can crash, or worse, corrupt some data and keep running.

>> Binding an HWND to a control variable of the proper type avoids these
>> problems and makes sending messages to the window a good bit more
>> convenient. In addition, binding ensures that if you do need to call
>> GetDlgItem, you get a pointer to the CWnd part of the control object you
>> bound to the HWND, which you can safely downcast to that object's type.

Re: Managing a project as it scales

>
>If you create a control of the proper type by any means, the same
>applies. Sure, you will get a compile error if you try to write, in
>effect:
> CButton but;
> but.SetReadOnly();
>
>But how realistic is this as a programming issue?

That's not what I'm talking about.

>randomly swapping buttons for edit controls, you probably deserve
>anything you get. But all you are likely to get will be a reliable
>assert while testing anyway.

Actually, you probably wouldn't get an assert, considering that most window functions are very thin wrappers around SendMessage. Instead, you'll end up sending a message the target window will interpret in its own peculiar way.

>I'm not convinced, because I haven't really suffered from any of the
>issues you mention.

Of course, you have the right to suffer from them in the future. :)

--

Doug Harrison
Visual C++ MVP

• **Follow-Ups:**

- ◆ **Re: Managing a project as it scales**
◇ From: Gerry Quinn

• **References:**

- ◆ **Managing a project as it scales**
◇ From: Alexander
- ◆ **Re: Managing a project as it scales**
◇ From: Joseph M . Newcomer
- ◆ **Re: Managing a project as it scales**
◇ From: Alexander
- ◆ **Re: Managing a project as it scales**
◇ From: Joseph M . Newcomer
- ◆ **Re: Managing a project as it scales**
◇ From: Gerry Quinn
- ◆ **Re: Managing a project as it scales**
◇ From: Doug Harrison [MVP]
- ◆ **Re: Managing a project as it scales**
◇ From: Gerry Quinn

Re: Managing a project as it scales

- Prev by Date: *Could not draw gray-color BITMAP in CView::OnDraw()*
- Next by Date: *Writing VARIANT to a binary file*
- Previous by thread: *Re: Managing a project as it scales*
- Next by thread: *Re: Managing a project as it scales*
- Index(es):
 - ◆ *Date*
 - ◆ *Thread*