

Re: Substituting the main menu bar(s)

Source: <http://www.tech-archive.net/Archive/VC/microsoft.public.vc.mfc/2006-06/msg00453.html>

- *From:* "KMA" <kma@xxxxxxxxxxxxxxxx>
 - *Date:* Wed, 7 Jun 2006 17:16:42 +0200
-

David (Wilk)

Just my two penneth....

You're right in that you should make the key an integer (Resource ID).

The tooltip/Menu text string is only one string, which is delimited by '\n', so as long as the translators don't mess this up there's no too much difficulty there.

The interesting part is how you get the strings from a variety of sources. I do something like this (pseudocode):

```
AfxMessageBox(GetLocalString(IDS_FATAL_ERROR));
```

```
CString GetLocalString(int iID)
{
if(FoundInCustomerSpecificDictionary()) return s;

if(FoundInGeneralDictionary()) return s;

else
return HardCodedEnglishVariant;
}
```

These functions are wrapped up into macros so I can write things like:
AfxMessageBox(STR_FATAL_ERROR)

where STR_FATAL_ERROR expands to return a string, hopefully in the users language but if worst comes to worst in the default hardcoded language.

I find this allows me to crack on with the functionality without worrying about the language issues. Every now and then I go through the spreadsheet with a local.

The only caveat/tip I give with this system is that you must immediately update a new line in the spreadsheet each time you add a "STR_" define, otherwise you forget to add it.

Re: Substituting the main menu bar(s)

As I see it, this system is easy to implement and quite scalable, with little extra ongoing development overhead.

"David Wilkinson" <no-reply@xxxxxxxxxxxx> wrote in message news:%23z0VkKkiGHA.3780@xxxxxxxxxxxxxxxxxxxxxxxxxxxx

David Webber wrote:

"David Wilkinson" <no-reply@xxxxxxxxxxxx> wrote in message news:OquihpXiGHA.1204@xxxxxxxxxxxxxxxxxxxxxxxxxxxx

(b) the realisation that as (i.a.) their cancel buttons show "Annuler" anyway (thanks to MFC) the program was already partly translated, and that therefore "gradual translation" was not going to be the sin I had thought it to be;

David:

Is this true? I know it is true for property sheet buttons (because they are not in the .rc file), and of dialogs like the open file dialog (which come from the operating system) but is it true of dialog buttons? Surely not (their text is specified in the .rc file).

You're right – I think in all cases. But I suspect it is also true for message boxes.

Personally, I do not like this feature. If my app is in English, I would like everything to be in English, partly because my user guide is in English. In fact, I deliberately set the text of property sheet buttons to English in order that my help system matches the actual application.

Re: Substituting the main menu bar(s)

I never bothered with this, but I may do in future. Do you just set the text of the buttons in using their standard control ID? Where do you do it? From the OnInitDialog of the property sheets?

By the way, what do you do to test the French version of your app? Do you have a real French version of Windows on another machine, or do you change your settings? Or neither?

At the moment I ask my tester friend in France :-)

But because I am translating all *my* RC files, all I need do is switch to French and check where *my* translation Cancel->Annuler does not happen and then I have a good idea where it won't happen the other way!

[I do set the keyboard to various languages sometimes, to test keyboard shortcuts. But that is different. In particular I had problems with shortcuts once with keys 0-9 as on some keyboards they use the Shift key. And I use things like > as shortcuts (for accents that look like > and against Microsoft's advice) so it needs careful testing.]

Dave

Hi Dave:

Yes, for my property sheets I use the ID's to set the button texts to the English values in the OnInitDialog of the sheet. But I do not (yet) try to do anything about other strings that might be displayed in the language of the operating system.

Let me be sure that I understand what you do exactly. You have an Excel sheet that has a column for each language. If I understand correctly, this functions as a dictionary, where the "key" is the English string, not the resource ID. I had thought of doing that also, but to use it for "last-minute-on-the-fly-substitution" rather than compile time generation of a .rc file.

I think that the reason you do it this way is that all strings in the .rc file are treated equally, whether they be menu text, control text, tooltip

Re: Substituting the main menu bar(s)

text, text for a message box, anything. You just look the English text up in the dictionary, and substitute it in the .rc file. This was my motivation also (but in a different context); the English string could come from anywhere -- the .rc file, from the back-end of my application, or even from hard coded strings (a habit I have never broken myself of, I'm afraid).

You mention one disadvantage of your dictionary to be that each English string can only have one translation. Another, it seems to me, is that if you change the English text (even in some trivial way) then you have to modify your dictionary. If your dictionary were keyed to resource ID's this would not happen. But then you would have trouble with things like menu items, which have two strings associated with them: the menu text and the tooltip text. But you could get around this by having two dictionaries (both keyed to resource ID's); one for menu/control text and another for tooltip text. It would complicate making the substitutions in the .rc file, but it would not be very hard. In this case, the strings in the original .rc file could become placeholders, because English could be treated the same as every other language. Every string in every language could come from your two Excel dictionaries.

David Wilkinson