Subject: Re: First catch of the day
Posted by Peter Mason on Wed, 12 Jan 2005 22:18:10 GMT
View Forum Message <> Reply to Message

Robert Barnett wrote:
> Does anyone know how to get around this one?
>
>
> I have had problems running IDL processes using both Watsyn and the
> IDL Broker on Siemens E.Soft workstations. The E.Soft application
> runs in full-screen mode obscuring the IDL windows. I've been trying
> to work out
> a way to keep the IDL windows 'always on top'.
>
> To solve this, I'm using CALL_EXTERNAL to access MS-Windows functions
> from IDL. The particular library is C:\WINNT\system32\user32.dll which
> is common over windows 98/NT/2000/XP.
>
> I create a wrapper as recommended in the IDL documentation
> result =
>
 CALL_EXTERNAL('C:\WINNT\system32\user32.dll','GetForegroundW indow',WRITE_WRA
PPER='user32GetForegroundWindow.c')
>
> I successfully compiled using Microsoft Visual C++ Toolkit 2003.
> cl -D_DLL -DWIN32 -D_MT /nologo /I"D:\RSI\IDL60\external\include" /c
> "GetForegroundWindow.c" /Fo"GetForegroundWindow.obj"
>
> However, I experience problems when attempting to link the compiled
> file into a DLL. The contents of the generated C program
> (GetForegroundWindow.c) indicate that I should use the following
> command:
>
> link /out:"GetForegroundWindow.dll" /nologo /nodefaultlib /dll
> "GetForegroundWindow.obj" /def:"GetForegroundWindow.def"
> "D:\RSI\IDL60\bin\bin.x86\idl32.lib" msvcrt.lib kernel32.lib
>
> It appears that only kernel32.lib is included with Microsoft Visual
> C++
> yet msvcrt.lib is not. Is this library really required? Do you know
> where I can obtain this library from?
>
> The linking also fails because there is no GetForegroundWindow.def
> file. This is explained in the documentation of the
> generatedGetForegroundWindow.c
>   * Note that these 2 commands are probably not quite enough: You
>   * will also need to supply a linker options file to control which
>   * symbols are exported. Under Unix, this file has various names

>   * (Solaris calls it a mapfile). Microsoft Windows calls it a def
> file.
>   * Read your systems linker documentation for further information.
> Do you know where I can obtain documentation on how to generate this
> .def file?
>
> Any assistance would be appreciated and would assist me in making the
> IDL product usable in the supplied environment.

Robert, I've never heard of E.Soft but I doubt that it's worth the trouble
using external code for this problem - unless the full-screen E.Soft window
has been created with the WS_EX_TOPMOST style for some bizarre reason.   (I
can't imagine anyone writing such a badly-behaved program though.)   You
would then want to find it and deal with it.

It's hard to recommend a solution because interface issues like "which
window is in the foreground" should really be left to the user.   It's bad
form to mess with this sort of thing.   Can't you get at your IDL windows
via the taskbar?   Or is the E.Soft window periodically re-instating itself
as the topmost window?

If your IDL application has a widget interface, here's an IDL-only approach
that you might try.   It goes against the grain of how a decent GUI should
behave but it might be justified under the circumstances.
Attach a timer event to some "unused" widget (e.g., some label or
intermediate base) in your heirarchy.   (Use WIDGET_CONTROL,
timer_widget_id, TIMER= for this.)   You might set the timer for a couple of
seconds, for example.   A timer event will get sent to your event handler.
When handling this event, use WIDGET_CONTROL, tlb, /SHOW, where "tlb" is the
widget ID of your IDL app's top-level base.   (If you have more than one
window in your app then you might do this to each one.)   Also, re-attach
the timer trigger as IDL's TIMER= mechanism is a one-shot affair.

If you want to pursue a CALL_EXTERNAL solution then as Haje said msvcrt.lib
should be there.   It's the C runtime library.   It's bound to be amongst
your C compiler's files.   First thing I'd try is to ditch the /nodefaultlib
from your link.   If this doesn't work then check that the environment
variables are set up correctly for your compiler.   (e.g., My C compiler
wants a "lib" environment variable that tells it library directories.)

HTH
Peter Mason