Subject: Re: Call External and Win32 API dll's Subroutine calls
Posted by Randall Frank on Tue, 16 Jan 2001 00:35:25 GMT
View Forum Message <> Reply to Message

Ok,
The right way to use CALL_EXTERNAL is to write your own wrapper DLL. Arguments and CDECL/PDECL issues are going to kill you however. Realize that CALL_EXTERNAL calls every function with an integer and a pointer argument. The pointers to your IDL variable data are in the "argv" array, not on the stack. If you cannot make your function compatible with that stack frame, you are out of luck. I do not see immediately how to do this with the calls you mention.
So, in IDL 5.4, use MAKE_DLL to do this for you. This is the most portable way and is supported by RSI. I have not tried it myself, because:

<< start of shameless plug >>
You can also use the free DLM I wrote that allows you to "define" the interface to an external function/DLL. It then registers that function with IDL and you can use it in IDL as if native. Of course, minimal error checking is provided, but if you get the definition correct, it does ensure that the interface to the function (stack frame) matches that definition (very handy). You can get it from Ronn Kling's web site:

http://www.rlkling.com/freeware/dlms.htm

It is the EXTPROC functions. It also provides a function to generically dereference a (C) pointer within IDL. Handy when tinkering with functions that return pointers to things. Anyway, that functionality is only available under Windows (I could probably do Intel Linux as well if there were interest) and all the source code is provided. FYI: the DLM includes sync/async sound recording and playback, TWAIN scanner support, a complete sockets interface, raw zlib access and a few other things (much of it is Windows specific).

I should update the DLM sometime. I have some new functions to add, but one (ugly) bug has me stuck right now and there have been a lot of paper deadlines and "Alice" sessions recently.
<< end of shameless plug >>

FWIW.

Later.

muswick@uhrad.com wrote:
>
> Maybe someone has solved these problems with calling standard Win32
> api's without having to use a C-Wrapper routine. I search this
> newsgroup and I have a few examples but none that seems to address
> this problem directly.
>
> There 3 examples below:
> 1: Calling a Win32 Subroutine with arguments
> The problem is that no data is returned.
>
> 2: Calling a Win32 Function without arguments
> No problem - This works (from this newsgroup)
>
> 3: Calling a Win32 Subrouting with arguments
> The problem is an immediate crash.
>
> Example 1
> ---------- -------
> From Win32 API:
> Public Declare Sub GetLocalTime Lib "kernel32" Alias "GetLocalTime"
> (lpSystemTime As SYSTEMTIME)
>
> Public Type SYSTEMTIME
> wYear As Integer
> wMonth As Integer
> wDayOfWeek As Integer
> wDay As Integer
> wHour As Integer
> wMinute As Integer
> wSecond As Integer
> wMilliseconds As Integer
> End Type
>
> Here is the IDL code I have tried:
>
> timedata = INTARR(8)
> timedata2 = BYTARR(16)
> SYSTEMTIME = { $
> wYear : 0, $
> wMonth : 0, $
> wDayOfWeek : 0, $
> wDay : 0, $
> wHour : 0, $
> wMinute : 0, $
> wSecond : 0, $
> wMilliseconds : 0}
>

```
>   result = CALL_EXTERNAL('kernel32.dll','GetLocalTime',timedata)
>   help,result
>   print,timedata
>
>   result = CALL_EXTERNAL('kernel32.dll','GetLocalTime',timedata2)
>   help,result
>   print,timedata2
>
>   result = CALL_EXTERNAL('kernel32.dll','GetLocalTime',systemtime)
>   help,result
>   print,systemtime
>
> RESULT        LONG      =           0
>       0       0       0       0       0       0       0       0
> RESULT        LONG      =           0
>    0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
> RESULT        LONG      =           0
> {      0       0       0       0       0       0       0       0}
>
> Example 2
>  ------------------------------------------------------------ -------
> Now a Win32 API function call with no arguments works:
>
> From Win32 API:
> Public Declare Function GetLogicalDrives Lib "kernel32" Alias
> "GetLogicalDrives" () As Long
>
> The IDL code:
>
>   drivemask=CALL_EXTERNAL('kernel32.dll','GetLogicalDrives')
>   help,drivemask
>
> DRIVEMASK       LONG      =         125
>
> Example 3
>  ------------------------------------------------------------ -------
> From Win32 API:
> Public Declare Sub GetSystemInfo Lib "kernel32" Alias "GetSystemInfo"
> (lpSystemInfo As SYSTEM_INFO)
>
> Public Type SYSTEM_INFO
>         dwOemID As Long
>         dwPageSize As Long
>         lpMinimumApplicationAddress As Long
>         lpMaximumApplicationAddress As Long
>         dwActiveProcessorMask As Long
>         dwNumberOrfProcessors As Long
>         dwProcessorType As Long
```

> dwAllocationGranularity As Long
> dwReserved As Long
> End Type
>
> The IDL code:
> sysinfo = {SYSTEM_INFO, $
> dwOemID : 0L, $
> dwPageSize : 0L, $
> lpMinimumApplicationAddress : 0L, $
> lpMaximumApplicationAddress : 0L, $
> dwActiveProcessorMask : 0L, $
> dwNumberOrfProcessors : 0L, $
> dwProcessorType : 0L, $
> dwAllocationGranularity : 0L, $
> dwReserved : 0L}
>
> result = CALL_EXTERNAL('kernel32.dll','GetSystemInfo',sysinfo)
> help,result
> print,sysinfo
>
> Causes an immediate Page Fault and crash of IDL.
>
> ---------- -------
>
> I am most likely applying the arguments wrong. I am positive that
> I could get the above to work using a C-wrapper with (argc, argv)
> calling, but there are reasons that I would perfer not to go this
> route.
>
> Any help would appreciated.
>
> Gary Muswick
> muswick@uhrad.com
> 216 844-7793
>
> Sent via Deja.com
> http://www.deja.com/

--
rjf.
Randy Frank | ASCI Visualization
Lawrence Livermore National Laboratory | rjfrank@llnl.gov
B451 Room 2039 L-561 | Voice: (925) 423-9399
Livermore, CA 94550 | Fax: (925) 423-8704