Subject: Pros & cons of methods 1) DLM and 2) Call_External of calling C from IDL
Posted by neil salmon on Sun, 31 Jul 2016 21:09:32 GMT
View Forum Message <> Reply to Message

Could anyone please summarize the pros and cons of using the two different methods 1) DLM and 2) Call_External of calling C routines from IDL. I'm using IDL Version 6.0. Many thanks, neil

Subject: Re: Pros & cons of methods 1) DLM and 2) Call_External of calling C from IDL
Posted by Michael Galloy on Mon, 01 Aug 2016 13:32:59 GMT
View Forum Message <> Reply to Message

On 7/31/16 3:09 PM, neil.a.salmon@gmail.com wrote:
> Could anyone please summarize the pros and cons of using the two
> different methods 1) DLM and 2) Call_External of calling C routines
> from IDL. I'm using IDL Version 6.0. Many thanks, neil

If you go with CALL_EXTERNAL, I would recommend an IDL wrapper routine
that checks the arguments. The wrapper routine can accept keywords and
hide the C/Fortran style API of the external call.

CALL_EXTERNAL

Pros:
- fairly easy
- doesn't require an C coding or knowing the IDL C API

Cons:
- ugly, brittle call
- no keywords
- C/Fortran style API, not a typical IDL calling style

DLM

Pros:
- can make a native style routine (this is how IDL is extended by
Harris engineers)
- access to IDL C API: keywords, data checking macros, variable
number of parameters, etc.

Cons:
- requires knowledge of C and the IDL C API (significant learning curve)
- requires a fair amount of C boilerplate code

Mike

--
Michael Galloy
www.michaelgalloy.com
Modern IDL: A Guide to IDL Programming (http://modernidl.idldev.com)

Subject: Re: Pros & cons of methods 1) DLM and 2) Call_External of calling C from IDL
Posted by neil salmon on Mon, 01 Aug 2016 19:15:08 GMT
View Forum Message <> Reply to Message

Michael,

thanks for your good points and the links. Does either of these processes have any impact on the speed with which the C runs called from IDL?

I'm operating IDL 6.0, so i take it your comments are equally valid from this rather old version?

I take it if you have many C routines you'd like to access from IDL it would probably best to invest in learning the DLM method?

many thanks,
neil

Subject: Re: Pros & cons of methods 1) DLM and 2) Call_External of calling C from IDL
Posted by Michael Galloy on Tue, 02 Aug 2016 22:25:01 GMT
View Forum Message <> Reply to Message

On 8/1/16 1:15 PM, neil.a.salmon@gmail.com wrote:
> Michael,
>
> thanks for your good points and the links. Does either of these
> processes have any impact on the speed with which the C runs called
> from IDL?

I haven't done any tests, but I doubt there is any significant time
difference.

> I'm operating IDL 6.0, so i take it your comments are equally valid
> from this rather old version?

Yes.

> I take it if you have many C routines you'd like to access from IDL
> it would probably best to invest in learning the DLM method?

A lot of the disadvantages of the DLM method are mitigated when doing many routines. For example, you only write the boilerplate code once, you only learn the API once, etc. So it is definitely *more* worth it if there are many routines. But the big difference is whether you want to use native features like keywords.

By the way, I have some routines in my library to automatically generate IDL DLM bindings for C routines. You specify the C routines in header-like format, such as:

```
  void gsl_rng_env_setup();
  IDL_PTRINT gsl_rng_alloc(IDL_PTRINT t);
  void gsl_rng_free(IDL_PTRINT r);

  void gsl_rng_set(IDL_PTRINT r, unsigned long seed);

  unsigned long gsl_rng_get(IDL_PTRINT r);
  double gsl_rng_uniform(IDL_PTRINT r);
  double gsl_rng_uniform_pos(IDL_PTRINT r);
  unsigned long gsl_rng_uniform_int(IDL_PTRINT r, unsigned long n);
```

And my library will output the DLM files. This header is constructed by hand with a bit of knowledge of the IDL DLM API. The original routine's prototypes are the following:

```
  const gsl_rng_type * gsl_rng_env_setup (void);
  gsl_rng *gsl_rng_alloc (const gsl_rng_type * T);
  void gsl_rng_free (gsl_rng * r);

  void gsl_rng_set (const gsl_rng * r, unsigned long int seed);

  INLINE_DECL unsigned long int gsl_rng_get (const gsl_rng * r);
  INLINE_DECL double gsl_rng_uniform (const gsl_rng * r);
  INLINE_DECL double gsl_rng_uniform_pos (const gsl_rng * r);
  INLINE_DECL unsigned long int gsl_rng_uniform_int (const gsl_rng * r,
unsigned long int n);
```

Let me know if you are interested. I can point you to the code and some examples.

Mike
--
Michael Galloy
www.michaelgalloy.com
Modern IDL: A Guide to IDL Programming (http://modernidl.idldev.com)

Subject: Re: Pros & cons of methods 1) DLM and 2) Call_External of calling C from IDL
Posted by neil salmon on Wed, 03 Aug 2016 11:21:35 GMT
View Forum Message <> Reply to Message

Mike,

thanks very much for the offer of the modules. I think for the time being i'll take the simple approach, and as i'm working with IDL 6.0 i'll use the make_dll routine. However with this, how do i tell IDL where the compiler is? There are a number of compilers on my machine that i know about: MS Visual Studio 10.0, MINGW gcc and g++, and TDM-GCC 64\gcc and g++. The directories where the gcc and g++ compilers reside is in the Environment Variable PATH. However could i specify compiler location in the IDL routines, or from within the IDL IDE? That way i dont have too much stuff in the Environment Variable PATH.

I'd prefer to start doing this from within IDL using make_dll just to keep things simple, then later i may do from within MSVS do create the dll.

Later i'll invest time on the DLM route, as this seems the best way if you have many routines, certainly the best route for long term.

many thanks,
neil

Subject: Re: Pros & cons of methods 1) DLM and 2) Call_External of calling C from IDL
Posted by Michael Galloy on Wed, 03 Aug 2016 13:19:26 GMT
View Forum Message <> Reply to Message

On 8/3/16 5:21 am, neil.a.salmon@gmail.com wrote:
> Mike,
>
> thanks very much for the offer of the modules. I think for the time
> being i'll take the simple approach, and as i'm working with IDL 6.0
> i'll use the make_dll routine. However with this, how do i tell IDL
> where the compiler is? There are a number of compilers on my machine
> that i know about: MS Visual Studio 10.0, MINGW gcc and g++, and
> TDM-GCC 64\gcc and g++. The directories where the gcc and g++
> compilers reside is in the Environment Variable PATH. However could i
> specify compiler location in the IDL routines, or from within the IDL
> IDE? That way i dont have too much stuff in the Environment Variable
> PATH.
>
> I'd prefer to start doing this from within IDL using make_dll just to
> keep things simple, then later i may do from within MSVS do create
> the dll.
>

> Later i'll invest time on the DLM route, as this seems the best way
> if you have many routines, certainly the best route for long term.
>
> many thanks, neil
>

Check out the !make_dll system variable:

```
IDL> help, !make_dll
** Structure !MAKE_DLL, 4 tags, length=64, data length=64:
   COMPILE_DIRECTORY
                STRING
 '/Users/mgalloy/.idl/idl/compile_dir-118-idl_8_4-darwin-x86_ '...
   COMPILER_NAME   STRING    'GCC'
   CC              STRING    'cc %X -arch x86_64 -fPIC -no-cpp-precomp
-dynamic -fPIC -fn'...
   LD              STRING    'cc -arch x86_64 -bundle -flat_namespace
-undefined suppress'...
```

Mike
--
Michael Galloy
www.michaelgalloy.com
Modern IDL: A Guide to IDL Programming (http://modernidl.idldev.com)

Subject: Re: Pros & cons of methods 1) DLM and 2) Call_External of calling C from IDL
Posted by natha on Wed, 03 Aug 2016 17:23:13 GMT
View Forum Message <> Reply to Message

Hi Michael and Neil,

Please note that you can compile using OpenMP and then, depending on your code, the speed of the C run may be much faster

Nata

Subject: Re: Pros & cons of methods 1) DLM and 2) Call_External of calling C from IDL
Posted by neil salmon on Thu, 04 Aug 2016 13:23:49 GMT
View Forum Message <> Reply to Message

Thanks Mike and Nata,

however, i'm still a bit stuck; typing the following gives:

```
IDL> help, !make_dll, /structure
** Structure !MAKE_DLL, 4 tags, length=48, data length=48:
   COMPILE_DIRECTORY
                 STRING    'C:\Users\Neil\.idl\idl_6_0_Win32_x86_m32_f64\c'...
   COMPILER_NAME   STRING    'Microsoft Visual C++ 7.0'
   CC              STRING    'cl %X -D_DLL -DWIN32 -D_MT /nologo /I"C:\Progr'...
   LD              STRING    'link /out:%L /nologo /nodefaultlib /dll %O /de'...
```

so i guess this means MSVC 7.0 was used to compile the IDL Version 6.0 which i'm running?

As far as i see the COMPILE DIRECTORY is where IDL keeps you own complied stuff. However, does this mean there is a MSVC 7.0 lurking in either the MS directories or the RSI directories in Program Files (x86), ready to be used when you type make_dll?

As far as i know i only a have an MSVC 10.0 compiler and it's not in the , C:\Users\Neil\.idl\idl_6_0_Win32_x86_m32_f64\c'... directory. Currently make_dll does not compile anything. How should make_dll known the path and the file name of the C compiler?

many thanks,
Neil

Subject: Re: Pros & cons of methods 1) DLM and 2) Call_External of calling C from IDL
Posted by Helder Marchetto on Thu, 04 Aug 2016 13:41:41 GMT
View Forum Message <> Reply to Message

On Thursday, August 4, 2016 at 3:24:01 PM UTC+2, neil.a...@gmail.com wrote:
> Thanks Mike and Nata,
>
> however, i'm still a bit stuck; typing the following gives:
>
```
> IDL> help, !make_dll, /structure
> ** Structure !MAKE_DLL, 4 tags, length=48, data length=48:
>    COMPILE_DIRECTORY
>                  STRING    'C:\Users\Neil\.idl\idl_6_0_Win32_x86_m32_f64\c'...
>    COMPILER_NAME   STRING    'Microsoft Visual C++ 7.0'
>    CC              STRING    'cl %X -D_DLL -DWIN32 -D_MT /nologo /I"C:\Progr'...
>    LD              STRING    'link /out:%L /nologo /nodefaultlib /dll %O /de'...
```
>
> so i guess this means MSVC 7.0 was used to compile the IDL Version 6.0 which i'm running?
>
> As far as i see the COMPILE DIRECTORY is where IDL keeps you own complied stuff. However, does this mean there is a MSVC 7.0 lurking in either the MS directories or the RSI directories in Program Files (x86), ready to be used when you type make_dll?
>
> As far as i know i only a have an MSVC 10.0 compiler and it's not in the ,

C:\Users\Neil\.idl\idl_6_0_Win32_x86_m32_f64\c'... directory. Currently make_dll does not compile anything. How should make_dll known the path and the file name of the C compiler?
>
> many thanks,
> Neil

Hi Neil,
I have no knowledge of DLLs, C or anything else in your question. However, I think that you can find some info here(see e.g. example 2):
http://www.harrisgeospatial.com/docs/MAKE_DLL.html
Notice that at the end of the page the version history is shown (necessary if don't have the latest IDL version.

Hope it helps,
Helder

Subject: Re: Pros & cons of methods 1) DLM and 2) Call_External of calling C from IDL
Posted by Michael Galloy on Thu, 04 Aug 2016 23:11:05 GMT
View Forum Message <> Reply to Message

On 8/4/16 7:23 am, neil.a.salmon@gmail.com wrote:
> Thanks Mike and Nata,
>
> however, i'm still a bit stuck; typing the following gives:
>
```
> IDL> help, !make_dll, /structure
> ** Structure !MAKE_DLL, 4 tags, length=48, data length=48:
>    COMPILE_DIRECTORY
>                 STRING    'C:\Users\Neil\.idl\idl_6_0_Win32_x86_m32_f64\c'...
>    COMPILER_NAME   STRING    'Microsoft Visual C++ 7.0'
>    CC              STRING    'cl %X -D_DLL -DWIN32 -D_MT /nologo /I"C:\Progr'...
>    LD              STRING    'link /out:%L /nologo /nodefaultlib /dll %O /de'...
```
>
> so i guess this means MSVC 7.0 was used to compile the IDL Version 6.0 which i'm running?
>
> As far as i see the COMPILE DIRECTORY is where IDL keeps you own
> complied stuff. However, does this mean there is a MSVC 7.0 lurking
> in either the MS directories or the RSI directories in Program Files
> (x86), ready to be used when you type make_dll?
>
> As far as i know i only a have an MSVC 10.0 compiler and it's not in
> the , C:\Users\Neil\.idl\idl_6_0_Win32_x86_m32_f64\c'... directory.
> Currently make_dll does not compile anything. How should make_dll
> known the path and the file name of the C compiler?
>
> many thanks, Neil

Those are the values for the default compiler on your system. Set them
as needed for your compiler.

Mike
--
Michael Galloy
www.michaelgalloy.com
Modern IDL: A Guide to IDL Programming (http://modernidl.idldev.com)

Subject: Re: Pros & cons of methods 1) DLM and 2) Call_External of calling C from IDL
Posted by neil salmon on Fri, 05 Aug 2016 14:08:25 GMT
View Forum Message <> Reply to Message

Thanks, Mike and Helder,

i can see that:

```
{
MAKE_DLL, 'testmodule', 'IDL_Load', INPUT_DIRECTORY=INDIR, $
   CC='gcc -c -fPIC '+ INCLUDE + '%C -o %O'
}
```

tells the system to look for the gcc.exe compiler, but do i put the gcc.exe file also in the INPUT_DIRECTORY, or any other compiler i wish MAKE_DLL to make use of?

Or does IDL have some other instruction where it looks for compilers?

many thanks,
neil

Subject: Re: Pros & cons of methods 1) DLM and 2) Call_External of calling C from IDL
Posted by neil salmon on Mon, 08 Aug 2016 07:42:18 GMT
View Forum Message <> Reply to Message

I'm trying to use the Call_external route but to start with i'm trying to create the dll for mg_callex_total.c using the make_dll.

The source code is:

```
CD, current=working_directory ; Sets current working directory
make_dll, 'mg_callex_total', 'IDL_Load',
INPUT_DIRECTORY=working_directory+'\test_call_external' , CC='gcc -c -fPIC %C -o %O', $
          OUTPUT_DIRECTORY  = working_directory+'\test_call_external', $
          COMPILE_DIRECTORY = working_directory+'\test_call_external'
end
```

I've change compile and output directories so i'm not messing files up in other parts of the system.

The response from IDL is:

```
IDL> .GO
 C:\USERS\NEIL\DOCUMENTS\IDL\test_call_external\mg_callex_tot al.c:1:0: warning: -fPIC
ignored for target (all code is position independent)
 float mg_callex_total(float arr[], int *n) {
 ^
link: extra operand `/nodefaultlib'
Try `link --help' for more information.
gcc -c -fPIC " C:\USERS\NEIL\DOCUMENTS\IDL\test_call_external\mg_callex_tot al.c " -o
"mg_callex_total_6704_NEIL-THINK.obj"
link /out:"mg_callex_total_6704_NEIL-THINK.dll" /nologo /nodefaultlib /dll
"mg_callex_total_6704_NEIL-THINK.obj" /def:"mg_callex_total_6704_NEIL-THINK.def"
"C:\Program Files (x86)\RSI\IDL60\bin\bin.x86\idl32.lib" msvcrt.lib kernel32.lib
Could Not Find  C:\Users\Neil\Documents\IDL\test_call_external\mg_callex_tot
al_6704_NEIL-THINK.exp
Could Not Find  C:\Users\Neil\Documents\IDL\test_call_external\mg_callex_tot
al_6704_NEIL-THINK.lib
```

 -- ----------------

Could anyone suggest what i've done wrong.

many thanks,
neil

Subject: Re: Pros & cons of methods 1) DLM and 2) Call_External of calling C from IDL
Posted by Michael Galloy on Mon, 08 Aug 2016 17:14:19 GMT
View Forum Message <> Reply to Message

On 8/8/16 1:42 AM, neil.a.salmon@gmail.com wrote:

```
>  CD, current=working_directory ; Sets current working directory
>  make_dll, 'mg_callex_total', 'IDL_Load',
INPUT_DIRECTORY=working_directory+'\test_call_external' , $
 >          CC='gcc -c -fPIC %C -o %O', $
>           OUTPUT_DIRECTORY  = working_directory+'\test_call_external', $
>           COMPILE_DIRECTORY = working_directory+'\test_call_external'
```

Do you have gcc on Windows?

Mike
--
Michael Galloy
www.michaelgalloy.com
Modern IDL: A Guide to IDL Programming (http://modernidl.idldev.com)

Subject: Re: Pros & cons of methods 1) DLM and 2) Call_External of calling C from IDL
Posted by neil salmon on Mon, 08 Aug 2016 21:41:32 GMT
View Forum Message <> Reply to Message

Yes, i'm operating Windows 7, with IDL Version 6.0. I've also put the path of the MSVC version 10.0 compiler (cl.exe) in the Windows Environment Variables, PATH, however MAKE_DLL does not seem to recognise. The gcc.exe compiler directory is also in the Windows Environment Variable, and MAKE_DLL does recognise it - so at least some small progress.

However, running the script:

```
make_dll, 'mg_callex_total', 'IDL_Load', $
INPUT_DIRECTORY=working_directory+'\test_call_external' , $
CC='gcc -c %C -o %O', $
OUTPUT_DIRECTORY  = working_directory+'\test_call_external', $
COMPILE_DIRECTORY = working_directory+'\test_call_external', $
VERBOSE=verbose, SHOW_ALL_OUTPUT=verbose
```

 -- -------
generates:

```
link: extra operand `/nodefaultlib'
Try `link --help' for more information.
gcc -c " C:\USERS\NEIL\DOCUMENTS\IDL\test_call_external\mg_callex_tot al.c " -o
"mg_callex_total_12888_NEIL-THINK.obj"
link /out:"mg_callex_total_12888_NEIL-THINK.dll" /nologo /nodefaultlib /dll
"mg_callex_total_12888_NEIL-THINK.obj" /def:"mg_callex_total_12888_NEIL-THINK.def"
"C:\Program Files (x86)\RSI\IDL60\bin\bin.x86\idl32.lib" msvcrt.lib kernel32.lib
Could Not Find  C:\Users\Neil\Documents\IDL\test_call_external\mg_callex_tot
al_12888_NEIL-THINK.exp
Could Not Find  C:\Users\Neil\Documents\IDL\test_call_external\mg_callex_tot
al_12888_NEIL-THINK.lib
```

I'm uncertain why "_12888_NEIL-THINK.obj" appears appended on to the end of the object file name, any ideas?

Furthermore, the object file "mg_callex_total_12888_NEIL-THINK.obj" i can't find on my system, and is not appearing in my specified OUTPUT_DIRECTORY.

- so i'm not at all certain what's going off. Grateful for any suggestions.

many thanks,
neil

Subject: Re: Pros & cons of methods 1) DLM and 2) Call_External of calling C from IDL
Posted by Helder Marchetto on Mon, 08 Aug 2016 21:48:27 GMT
View Forum Message <> Reply to Message

Hi Neil,
I *think* that you are not specifying the right compiler: you have cl, not gcc.
Try changing this:
CC='gcc -c %C -o %O'
Too something like this:
CC='cl -c %C -o %O'
But first check what the cl compiler options are!

Cheers, Helder

Subject: Re: Pros & cons of methods 1) DLM and 2) Call_External of calling C from IDL
Posted by neil salmon on Tue, 09 Aug 2016 10:26:47 GMT
View Forum Message <> Reply to Message

Small progress, both gcc and cl compilers create and object files from the sources, however it does not proceed to the link phase. See what i mean in the following.

IDL script for gcc compiler:

```
make_dll, 'mg_callex_total', 'IDL_Load',export_rtns, $
INPUT_DIRECTORY=working_directory+'\test_call_external' , CC='gcc -c %C', $ ;
```

generates:

```
gcc -c " C:\USERS\NEIL\DOCUMENTS\IDL\test_call_external\mg_callex_tot al.c "
Could Not Find  C:\Users\Neil\Documents\IDL\test_call_external\IDL_Load_4592
_NEIL-THINK.exp
Could Not Find  C:\Users\Neil\Documents\IDL\test_call_external\IDL_Load_4592 _NEIL-THINK.lib
Could Not Find  C:\Users\Neil\Documents\IDL\test_call_external\mg_callex_tot
```

al_4592_NEIL-THINK.obj

 -- -
IDL script for cl compiler:

```
make_dll, "mg_callex_total", "IDL_Load", export_rtns,        $
INPUT_DIRECTORY=working_directory+"\test_call_external", CC="cl %C %O"
```

generates:

```
mg_callex_total.c
Microsoft (R) Incremental Linker Version 10.00.40219.01
Copyright (C) Microsoft Corporation.  All rights reserved.

/out:mg_callex_total.exe
mg_callex_total.obj
mg_callex_total_4592_NEIL-THINK.obj
LINK : fatal error LNK1181: cannot open input file 'mg_callex_total_4592_NEIL-THINK.obj'
Could Not Find  C:\Users\Neil\Documents\IDL\test_call_external\IDL_Load_4592
_NEIL-THINK.exp
Could Not Find  C:\Users\Neil\Documents\IDL\test_call_external\IDL_Load_4592 _NEIL-THINK.lib
Could Not Find  C:\Users\Neil\Documents\IDL\test_call_external\mg_callex_tot
al_4592_NEIL-THINK.obj
```
 -- --------------

However, for some reason what should be the output file 'IDL_Load' it appends on to the end of this _4593_NEIL-THINK, which is the name of my machine on the net. I take it this should not happen? However, no such output file appears to be generated.

Any clues why this does not proceed to the link phase?

many thanks,
neil

Subject: Re: Pros & cons of methods 1) DLM and 2) Call_External of calling C from IDL
Posted by neil salmon on Wed, 10 Aug 2016 10:31:11 GMT
View Forum Message <> Reply to Message

I gather the difficulty here is in selecting the compiler options using the keywords CC and LD in the make_dll routine.

I'm trying to use both Microsoft Visual Studio VC and the gcc compiler (both have different flags for compiler options). I don't really care which one works, i just need one of them to work.

Any suggestions about what compiler options for these compilers?

many thanks,

neil

Subject: Re: Pros & cons of methods 1) DLM and 2) Call_External of calling C from IDL
Posted by Michael Galloy on Wed, 10 Aug 2016 14:44:01 GMT
View Forum Message <> Reply to Message

On 8/10/16 4:31 AM, neil.a.salmon@gmail.com wrote:
> I gather the difficulty here is in selecting the compiler options
> using the keywords CC and LD in the make_dll routine.
>
> I'm trying to use both Microsoft Visual Studio VC and the gcc
> compiler (both have different flags for compiler options). I don't
> really care which one works, i just need one of them to work.
>
> Any suggestions about what compiler options for these compilers?

Yes, this is why I use the default compiler on each platform so I don't
have to figure these things out. Even when I have my own build system,
i.e., using CMake, I just use the default compiler and copy the options
that !make_dll has over to my system.

Mike
--
Michael Galloy
www.michaelgalloy.com
Modern IDL: A Guide to IDL Programming (http://modernidl.idldev.com)

Subject: Re: Pros & cons of methods 1) DLM and 2) Call_External of calling C from IDL
Posted by neil salmon on Wed, 10 Aug 2016 16:31:10 GMT
View Forum Message <> Reply to Message

Mike,
yes i'm beginning to appreciate this. What appears as a simple 2 line procedure (make_dll followed by call_external) appears now like finding your way out of a maze blindfolded.
Cheers, neil

Subject: Re: Pros & cons of methods 1) DLM and 2) Call_External of calling C from IDL
Posted by neil salmon on Wed, 10 Aug 2016 18:20:21 GMT
View Forum Message <> Reply to Message

Mike,

I did eventually get make_dll to create a dll file using the gcc compiler and the compile and link options:

cc = 'gcc -fopenmp -c %C -o %O'
ld = 'gcc -shared %O -o %L'

However, is a *.dll file the same as a shared object *.so file?

Putting the name of the *.dll file in as an argument to the call_external routine generates the error:

```
% CALL_EXTERNAL: Error loading sharable executable.
                Symbol: mg_callex_total, File = mg_callex_total.dll
                ERROR_MOD_NOT_FOUND
```

So any ideas how i should proceed now.

many thanks,
Neil

Subject: Re: Pros & cons of methods 1) DLM and 2) Call_External of calling C from IDL
Posted by Haje Korth on Thu, 11 Aug 2016 11:26:35 GMT
View Forum Message <> Reply to Message

On Wednesday, August 10, 2016 at 2:20:25 PM UTC-4, neil.a...@gmail.com wrote:
> Mike,
>
> I did eventually get make_dll to create a dll file using the gcc compiler and the compile and link options:
>
> cc = 'gcc -fopenmp -c %C -o %O'
> ld = 'gcc -shared %O -o %L'
>
>
> However, is a *.dll file the same as a shared object *.so file?
>
> Putting the name of the *.dll file in as an argument to the call_external routine generates the error:
>
```
> % CALL_EXTERNAL: Error loading sharable executable.
>                 Symbol: mg_callex_total, File = mg_callex_total.dll
>                 ERROR_MOD_NOT_FOUND
```
>
> So any ideas how i should proceed now.
>

> many thanks,
> Neil

For dll you must export the routine so that it can be used. This is done with the module definition file. Not sure if make_dll takes care of it. I build my dlms in Visual Studio.

Subject: Re: Pros & cons of methods 1) DLM and 2) Call_External of calling C from IDL
Posted by neil salmon on Thu, 11 Aug 2016 12:32:31 GMT
View Forum Message <> Reply to Message

Haje,

forgive me i'm still learning, but when you say export, what needs to be exported to where, and the module definition file, what that?

(surely only make_dll and call_external are required, after all that is what the help file leads you to believe)

many thanks,
Neil

Subject: Re: Pros & cons of methods 1) DLM and 2) Call_External of calling C from IDL
Posted by neil salmon on Thu, 11 Aug 2016 14:39:35 GMT
View Forum Message <> Reply to Message

In the make_dll routine you can set keyword /NOCLEANUP, which i did and then temporary files, namely:*.out, .obj, *.bat and *.def are created, for you to view.

(without setting /NOCLEANUP these files appear only temporarily then vanish)

Some contents of the *.bat file, perhaps someone can diagnose a problem?:

```
echo ################################################################ ############
> "%IDL_OUT_FILE%"
echo # >> "%IDL_OUT_FILE%"
echo # Error output from building library >> "%IDL_OUT_FILE%"
echo # >> "%IDL_OUT_FILE%"
echo #    File Name: %IDL_REAL_NAME% >> "%IDL_OUT_FILE%"
echo #    User: Neil@NEIL-THINK >> "%IDL_OUT_FILE%"
echo #    IDL Version: 6.0 (Win32 x86) >> "%IDL_OUT_FILE%"
echo #    Memory Bits: 32 >> "%IDL_OUT_FILE%"
echo #    File Bits: 64 >> "%IDL_OUT_FILE%"
echo #    Date: Thu Aug 11 15:02:25 2016 >> "%IDL_OUT_FILE%"
echo # >> "%IDL_OUT_FILE%"
echo # >> "%IDL_OUT_FILE%"
echo # Windows IDL executes %IDL_BAT_FILE% to >> "%IDL_OUT_FILE%"
echo # build the DLL. Any errors produced in that process >> "%IDL_OUT_FILE%"
echo # are redirected to this file.>> "%IDL_OUT_FILE%"
echo # >> "%IDL_OUT_FILE%"
echo # Any output resulting from this process appears after the next line >> "%IDL_OUT_FILE%"
echo ################################################################ ############
>> "%IDL_OUT_FILE%"
```

----
thank you, neil

Subject: Re: Pros & cons of methods 1) DLM and 2) Call_External of calling C from IDL
Posted by Haje Korth on Fri, 12 Aug 2016 12:31:24 GMT
View Forum Message <> Reply to Message

On Thursday, August 11, 2016 at 8:32:36 AM UTC-4, neil.a...@gmail.com wrote:
> Haje,
>
> forgive me i'm still learning, but when you say export, what needs to be exported to where, and the module definition file, what that?
>
> (surely only make_dll and call_external are required, after all that is what the help file leads you to believe)
>
> many thanks,
> Neil

You can google that, it has nothing to do with IDL. I would actually suggest for you to get Ronn Kling's book Calling C from IDL. It speeded up my development efforts significantly and is worth every penny.

Subject: Re: Pros & cons of methods 1) DLM and 2) Call_External of calling C from IDL
Posted by neil salmon on Sun, 14 Aug 2016 19:35:50 GMT
View Forum Message <> Reply to Message

Haje,

yes that from Ronn is a good book, but centred on the DLM approach, i was really just trying to implement a two line approach of make_dll followed by Call_External.

I cant be too far from this working now as it creates the *.dll file and the *.def, *.bat, *.obj as temporary and keeping copies of if you set NOCLEANUP.

Anyone - where should i look to find where outstanding errors area?

many thanks,
neil