

---

Subject: Re: dynamic linking on Mac  
Posted by [David Grier](#) on Sat, 11 Jun 2011 00:08:15 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

Hi Mary Jo,

Try

```
print, !MAKE_DLL
```

which prints the CCFLAGS and LDFLAGS needed to build a DLL on the present architecture.

All the best,

David

On 06/10/2011 07:01 PM, brodzik@nsidc.org wrote:

```
> Hi, all,
>
> Has anyone running IDL on a Mac successfully linked to a shared C
> library?
>
> I've got some code that lets me call the routines in a C library from
> IDL. It's name is call_grids.c and everything works fine on my linux
> system. I'm trying to port the whole thing from Suse linux to a Mac,
> and I'm having trouble on the linker step for call_grids.so. I've
> already compiled and tested the original C library (which is
> libmapx.a), it has some command-line programs that call it, and they
> are working fine.
>
> But for call_grids.c, in my Linux options, my makefile sets CFLAGS to -
> fPIC and C_LD_FLAGS to -shared, but I found out pretty quickly that
> Darwin's ld doesn't support "-shared". I found one helpful site on-
> line that suggested that the Darwin equivalent of '-shared' is '-
> bundle -flat_namespace -undefined suppress', but that returns this
> problem:
>
> ld -bundle -flat_namespace -undefined suppress -o call_grids.so
> call_grids.o -L/Users/mj/lib -lmapx -lm
> ld: symbol dyld_stub_binding_helper not defined (usually in crt1.o/
> dylib1.o/bundle1.o) for inferred architecture x86_64
> make[1]: *** [call_grids.so] Error 1
>
> Then I started researching these ld switches and found several
> alternatives to -bundle, including -dynamic, -dylib, -dylinker. My
> trouble is that I'm not sure what it is I should be *trying* build,
> here. Going back to the suse ld man page, it just says -shared
```

> "builds a shared library". Well, I guess I knew that. What I don't  
> know is what that's supposed to look like on a Mac.  
>  
> Can anyone offer me some tips on what I should be trying to create?  
>  
> Thanks,  
> Mary Jo  
>

---

---

Subject: Re: dynamic linking on Mac  
Posted by [brodzik@nsidc.org](mailto:brodzik@nsidc.org) on Wed, 15 Jun 2011 16:04:08 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

David, thanks for the !MAKE\_DLL tip--I would have ignored that myself, because I always think of DLLs as a windows-only thing. It got my compilation working, and I'm (partially) off and running. Now I can access the state data and the init method from the C level of things, but IDL seg faults when I call the method that is the meat of the C code--the forward/reverse transform.

I found information about Ronn Kling's book "Calling C and C++ from IDL"--I'm thinking of buying it. Can anyone give a short review?

Mary Jo

On Jun 10, 6:08 pm, David Grier <david.gr...@nyu.edu> wrote:

> Hi Mary Jo,  
>  
> Try  
>  
> print, !MAKE\_DLL  
>  
> which prints the CCFLAGS and LDFLAGS needed to build a DLL on the  
> present architecture.  
>  
> All the best,  
>  
> David  
>  
> On 06/10/2011 07:01 PM, brod...@nsidc.org wrote:  
>  
>  
>

>  
>  
>  
>  
>> Hi, all,  
>  
>> Has anyone running IDL on a Mac successfully linked to a shared C  
>> library?  
>  
>> I've got some code that lets me call the routines in a C library from  
>> IDL. It's name is call\_grids.c and everything works fine on my linux  
>> system. I'm trying to port the whole thing from Suse linux to a Mac,  
>> and I'm having trouble on the linker step for call\_grids.so. I've  
>> already compiled and tested the original C library (which is  
>> libmapx.a), it has some command-line programs that call it, and they  
>> are working fine.  
>  
>> But for call\_grids.c, in my Linux options, my makefile sets CFLAGS to -  
>> fPIC and C\_LD\_FLAGS to -shared, but I found out pretty quickly that  
>> Darwin's ld doesn't support "-shared". I found one helpful site on-  
>> line that suggested that the Darwin equivalent of '-shared' is '-  
>> bundle -flat\_namespace -undefined suppress', but that returns this  
>> problem:  
>  
>> ld -bundle -flat\_namespace -undefined suppress -o call\_grids.so  
>> call\_grids.o -L/Users/mj/lib -lmapx -lm  
>> ld: symbol dyld\_stub\_binding\_helper not defined (usually in crt1.o/  
>> dylib1.o/bundle1.o) for inferred architecture x86\_64  
>> make[1]: \*\*\* [call\_grids.so] Error 1  
>  
>> Then I started researching these ld switches and found several  
>> alternatives to -bundle, including -dynamic, -dylib, -dylinker. My  
>> trouble is that I'm not sure what it is I should be \*trying\* build,  
>> here. Going back to the suse ld man page, it just says -shared  
>> "builds a shared library". Well, I guess I knew that. What I don't  
>> know is what that's supposed to look like on a Mac.  
>  
>> Can anyone offer me some tips on what I should be trying to create?  
>  
>> Thanks,  
>> Mary Jo

---

---

Subject: Re: dynamic linking on Mac  
Posted by [Michael Galloy](#) on Wed, 15 Jun 2011 16:52:21 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

On 6/15/11 10:04 AM, brodzik@nsidc.org wrote:

> David, thanks for the !MAKE\_DLL tip--I would have ignored that myself,  
> because I always  
> think of DLLs as a windows-only thing. It got my compilation working,  
> and I'm (partially)  
> off and running. Now I can access the state data and the init method  
> from the C level of  
> things, but IDL seg faults when I call the method that is the meat of  
> the C code--the  
> forward/reverse transform.  
>  
> I found information about Ronn Kling's book "Calling C and C++ from  
> IDL"--I'm thinking  
> of buying it. Can anyone give a short review?

Ronn's book is great resource for learning to write a DLM. In particular, for all the examples of passing different types of parameters into C, i.e., arrays, scalars, strings, etc. as positional parameters or keywords. Explanation is clear and concise. All the code comes along with it on a CD. I keep this by my desk and refer to it regularly.

Also, check out the External Development Guide (in the IDL distribution at [help/pdf/edg.pdf](http://help/pdf/edg.pdf)). It covers many different techniques, DLMs are covered heavily since they are the most complicated.

Mike

--

Michael Galloy

[www.michaelgalloy.com](http://www.michaelgalloy.com)

Modern IDL, A Guide to Learning IDL: <http://modernidl.idldev.com>

Research Mathematician

Tech-X Corporation

---