Subject: Re: trying to link with C++ on Unix
Posted by Martha Kusterer on Thu, 10 Dec 1998 08:00:00 GMT
View Forum Message <> Reply to Message

Hi,
You were right, when I created a main.c to link with the C library it didn't work.
I inadvertantly saved the C++ code without the extern "C" {} around the forward
declarations. It now works.

If anyone needs examples email me at martha.kusterer@jhuapl.edu and I will send them.

thanks,
martha

Stein Vidar Hagfors Haugan wrote:

> In article <366F2CCC.1DC69990@aplcomm.jhuapl.edu>
> "Thomas L. Kusterer" <kustetl1@aplcomm.jhuapl.edu> writes:
>
>> Hi,
>> I am trying to access a library in gnu C++ by using wrapper routines
>> in C and IDL. I have linked the C and C++ libraries to be position
>> independent and sharable but the IDL code runs the C and then says that
>> it can't resolve the C++ routine name. I have included the extern "C"
>> around the C++ routine names. I have made the C library link with the
>> C++.
>>
>> Does anyone have any ideas?
>
> Let me get this right: The IDL wrapper calls a C wrapper function,
> which in turn calls a C++ library function, right? (That's the
> way it's supposed to be, normally)
>
> But IDL complains (on loading the shareable) that it cannot find
> the C++ library function?
>
> Then your problem is not in IDL, but in the C wrapper.
> I suspect that declaring the C++ to be extern "C" is a
> bit... hmm... suspect. Shouldn't it be extern "C++"
> (if that's allowed...never done this before). 'Cause
> that's what it is, I mean. To simplify your testing,
> write a C main() program that calls your C wrapper,
> then try to compile the thing into an executable.
> This should *not* work at the present, if I got your
> problem right. Now, try pre-/postfixing the C++

> routine name with underscores (one, or two, or...).
> Better yet, find out from your C++/C documentation
> exactly how to call C++ routines.
>
> If, on the other hand, IDL complains that it cannot
> find the C wrapper, then try pre-/postfixing (or drop
> the pre-/postfix!) the C wrapper function name in
> various ways.
>
> Or maybe it's just that the dynamical loader library path
> ($LD_LIBRARY_PATH on e.g. Digital UNIX) doesn't point to
> where the C++ libraries are? I'm a bit foggy on how this
> *really* works, I have to look the thing up every time
> I suspect something's wrong about it.
>
> Best of luck,
>
> Stein Vidar
> (It would be nice to get some feedback on how your problem
> is solved in the end)

Subject: Re: trying to link with C++ on Unix
Posted by steinhh on Thu, 10 Dec 1998 08:00:00 GMT
View Forum Message <> Reply to Message

In article <366F2CCC.1DC69990@aplcomm.jhuapl.edu>
"Thomas L. Kusterer" <kustetl1@aplcomm.jhuapl.edu> writes:

> Hi,
> I am trying to access a library in gnu C++ by using wrapper routines
> in C and IDL. I have linked the C and C++ libraries to be position
> independent and sharable but the IDL code runs the C and then says that
> it can't resolve the C++ routine name. I have included the extern "C"
> around the C++ routine names. I have made the C library link with the
> C++.
>
> Does anyone have any ideas?

Let me get this right: The IDL wrapper calls a C wrapper function,
which in turn calls a C++ library function, right? (That's the
way it's supposed to be, normally)

But IDL complains (on loading the shareable) that it cannot find
the C++ library function?

Then your problem is not in IDL, but in the C wrapper.
I suspect that declaring the C++ to be extern "C" is a

bit... hmm... suspect. Shouldn't it be extern "C++"
(if that's allowed...never done this before). 'Cause
that's what it is, I mean. To simplify your testing,
write a C main() program that calls your C wrapper,
then try to compile the thing into an executable.
This should *not* work at the present, if I got your
problem right. Now, try pre-/postfixing the C++
routine name with underscores (one, or two, or...).
Better yet, find out from your C++/C documentation
exactly how to call C++ routines.

If, on the other hand, IDL complains that it cannot
find the C wrapper, then try pre-/postfixing (or drop
the pre-/postfix!) the C wrapper function name in
various ways.

Or maybe it's just that the dynamical loader library path
($LD_LIBRARY_PATH on e.g. Digital UNIX) doesn't point to
where the C++ libraries are? I'm a bit foggy on how this
really works, I have to look the thing up every time
I suspect something's wrong about it.

Best of luck,

Stein Vidar
(It would be nice to get some feedback on how your problem
is solved in the end)
