Subject: Re: resolve_routine
Posted by Christian Soeller on Fri, 23 Aug 1996 07:00:00 GMT
View Forum Message <> Reply to Message

Klaus Martin Pontoppidan <pontop@astro.ku.dk> writes:

> i.e. a path and filename or only a filename in the IDL path. The
> resolve_routine works fine with the IDL library procedures, but I fail
> to detect the difference between these and my own procedures. The
> documentation claims that it works with user-written procedures, so....
> I would be most grateful if someone could help me with this problem.

I have tried some of my own functions with it. It seems to work fine
when you pass it the name of one of your functions for which the
corresponding file is in the path, e.g.,

a procedure called tvlarge, corresponding file tvlarge.pro is in the path:

IDL> resolve_routine,'tvlarge'
% Compiled module: TVLARGE.
IDL>

but

IDL> resolve_routine,'tvlarge.pro'
% Attempt to call undefined procedure/function: 'TVLARGE.PRO'.
% Execution halted at: $MAIN$
IDL>

That seems to imply that you have to give the name of the procedure, not the
path to the file containing that procedure. Therefore, that file must
be in the path and follow the IDL naming conventions (PROCEDURE_NAME.pro).

Hope this helps

Christian

---------- --------
Christian Soeller mailto: csoelle@sghms.ac.uk
St. Georges Hospital Medical School Dept. of Pharmacology
Cranmer Terrace London SW17 0RE

Subject: Re: resolve_routine
Posted by David Theil on Tue, 27 Aug 1996 07:00:00 GMT
View Forum Message <> Reply to Message

Klaus Martin Pontoppidan wrote:

>
> IDL complains that the
> function/procedure that I'm trying to compile from inside a procedure is
> undefined! And of course it is undefined - it isn't compiled yet...
> The syntax as described in the documentation is:
>
> --
> Klaus Martin Pontoppidan
>
> pontop@astro.ku.dk

A common reason for this problem is that the called routine is in the same file as
the calling routine, but AFTER the calling routine. When IDL compiles the calling
routine, it cannot link in the called function because it is undefined at that point.
If you force the file to recompile, it will then see the the compiled called function.

This is still a dangerous situation because any changes you make to the called function
will not take effect when this function is called from the parent routine until
the second recompilation.

The obvious way to avoid this sort of problem is to always put all of the called
sub-routines before the parent routine in any file that contains both.

David Theil

home work
University of Colorado
3360 34th St. D campus box 389
Boulder, Colorado Boulder, Colorado
80301 80309
(303)449-3113 (303)492-0895

No guts, no glory.