Subject: Locating IDL source code file
Posted by Craig[1] on Tue, 14 Jun 2005 11:52:34 GMT
View Forum Message <> Reply to Message

[copied over from comp.lang.idl - don't know how to properly cross-post it in retrospect using Google groups]

Hi all,

I'm trying to write some code that will read information (like version numbers and dates) from the source files for procedures and functions that it uses. In Matlab I would use a command like

>> which fred

to return the full path to the source file "fred.m". How do I get the same result from IDL v6.1? I've tried something like

IDL> ttt = file_which('fred.pro', /include_current_dir)

but just got a null string returned. This might be because I'm using the IDLDE, so the routine isn't actually on the search path !PATH, even though it has been compiled OK when building the project.

Once this problem is solved, I'll also need to determine the filename for the source file containing a definition of 'fred', because most of the routines in this project don't have their own 'fred.pro' source file.

Any ideas?
TIA,
Craig

Subject: Re: Locating IDL source code file
Posted by Mark Hadfield on Tue, 14 Jun 2005 22:12:22 GMT
View Forum Message <> Reply to Message

Hi Craig

I know your question has been answered elsewhere, but I'd just like to make a couple of comments on the original post.

> ...This might be because I'm using
> the IDLDE, so the routine isn't actually on the search path !PATH, even
> though it has been compiled OK when building the project.

This is a bit of a non-sequitur. How does using IDLDE relate to whether the file containing the routine is on the !PATH.

My advice: put files containing IDL routines on !PATH.

> Once this problem is solved, I'll also need to determine the filename
> for the source file containing a definition of 'fred', because most of
> the routines in this project don't have their own 'fred.pro' source
> file.

My second piece of advice: give each routine its own source file.

--
Mark Hadfield "Kei puwaha te tai nei, Hoea tahi tatou"
m.hadfield@niwa.co.nz
National Institute for Water and Atmospheric Research (NIWA)

Subject: Re: Locating IDL source code file
Posted by R.G. Stockwell on Tue, 14 Jun 2005 23:02:25 GMT
View Forum Message <> Reply to Message

"Mark Hadfield" <m.hadfield@niwa.co.nz> wrote in message
news:d8nko8$1cs$1@newsreader.mailgate.org...
> Hi Craig
...
> ...How does using IDLDE relate to whether the file containing the routine
> is on the !PATH.
>
> My advice: put files containing IDL routines on !PATH.

I agree with Mark. To answer the question, one can open routines in the IDLDE fairly easily, and compile them because they are in the DE. When people send me code, I don't put it in my path, and usually run it in the IDLDE from another folder.

>> Once this problem is solved, I'll also need to determine the filename
>> for the source file containing a definition of 'fred', because most of
>> the routines in this project don't have their own 'fred.pro' source
>> file.
>
> My second piece of advice: give each routine its own source file.

Again I agree. If you think of what one is doing when they have multiple routines in a file, it is an attempt at hiding code from other modules. What you really want to be doing is object programming,

and making methods/routines (i.e. Tom) that are hidden from everything
except
the Fred class.

Cheers,
bob

Subject: Re: Locating IDL source code file
Posted by Mark Hadfield on Tue, 14 Jun 2005 23:30:24 GMT
View Forum Message <> Reply to Message

R.G. Stockwell wrote:
> Mark Hadfield wrote in message
>> My second piece of advice: give each routine its own source file.
>
> Again I agree. If you think of what one is doing when they have
> multiple routines in a file, it is an attempt at hiding code from other
> modules. What you really want to be doing is object programming,
> and making methods/routines (i.e. Tom) that are hidden from everything
> except
> the Fred class.

I don't agree with that. Hiding routines from those who don't need to know about them should not have to be synonymous with object-orientation. It's just that in IDL classes do have a mechanism for this hiding and routines don't. Look at Python's modules to see how this should be done.

(I recommend that any IDL programmer who is interesting in language design should look at Python and Numeric Python. But be aware in doing so that some of the choices made in Python have made it *way* slower than IDL in some situations.)

Back to IDL now. If we have a routine "fred" and we want it to call another routine that has no other reason to exist, then the conventional way of achieving this is to give it a name that reflects its relation to fred *and* its function, then put it in fred.pro *before* the main routine:

*** file fred.pro ***

```
function fred_add, x, y

   return, x+y

end

pro fred
```

```
  if fred_add(1, 1) ne 2 then message, 'Huh?'

end

*** end file fred.pro ***
```

This is a kludge, because it's not really private. If file bill.pro defines a function fred__add then the clash will break things (and will be very hard to find, to boot). But as long as all the private routines in bill.pro start with "bill_", there shouldn't be a problem.

Yes, you could avoid this by rewriting fred as a class, but I think in most cases this would be a perversion of object-oriented programming.

You're still left with the problem that there might be two "fred"s on the path. This is a problem with IDL's flat name space for routines. Again, something like Python modules would alleviate the problem, but it ain't gonna happen any time soon.

--
Mark Hadfield "Kei puwaha te tai nei, Hoea tahi tatou"
m.hadfield@niwa.co.nz
National Institute for Water and Atmospheric Research (NIWA)

Subject: Re: Locating IDL source code file
Posted by Craig[1] on Wed, 15 Jun 2005 10:24:28 GMT
View Forum Message <> Reply to Message

Mark and Bob,

Thanks for your comments. I agree that having separate source files for each routine would be nicer (and this is what I do in Matlab, for example), but I'm working with code from somebody else, trying to minimize the changes to the original code while writing something else around it. I have had issues with the ordering of source files messing up the build process, and even having some procedures defined multiple times in different source files.

As for !Path, I used to set that when I first used PV-Wave several years ago, but having just come back (to IDL), I've found that it has an IDE, and have been just running that 'out of the box'.

Regards,
Craig
