Subject: Re: INIT functions... called when?
Posted by John-David T. Smith on Tue, 07 Mar 2000 08:00:00 GMT
View Forum Message <> Reply to Message

Steve Allison wrote:
>
> Hi,
>
> I'm just starting out with IDL and am having trouble creating objects, their
> INIT methods don't seem to get called. Here a simple example of what I'm
> doing:
>
> <All in "TestClass.pro">
>
> PRO TestClass1__define
> struct = {TestClass1, number:0L}
> END
>
> FUNCTION TestClass1::init, arg
> print, 'Initializing'
> self.number = arg
> RETURN,TRUE
> END
>
> PRO TestClass1::show
> print, 'Number is', self.number
> END
>
> PRO TestClass1::set, arg
> self.number = arg
> END
>
> Nothing staggering there... Anyway, I load this file into IDLDE, compile it
> (it seems to compile without any complaints), and try to create objects
> with:
>
> IDL> p = OBJ_NEW('TestClass1',65)
>
> Which in my book should set the field 'number' to 65. But...
>
> IDL> p->show
>
> gives:
>
> Number is 0
>
> So clearly INIT isn't being called (the message 'Initializing' doesn't show
> either). I can change the value of 'number' using TestClass1::Set, and that

> seems to work fine - it's just the INIT method I'm having trouble with. Why
> isn't it being called? The documentation seems to say that it should be
> called automatically when I call OBJ_NEW.
>
> Thanks for your help...

Put your TestClass1__define procedure at the end of the file. The reason this file is being accessed at all, is because IDL doesn't know anything about the TestClass1 class. It looks for a procedure testclass1__define, which is run from the file "testclass1__define.pro", above, in order to define the class. If it were at the bottom of the file, a side effect of this definition would be the compilation of all the TestClass1 methods, including Init. Since it's at the top, IDL stops looking through the file and compiling immediately. Seeing no Init method compiled or on the path as file testclass1__init.pro, IDL concludes there isn't one, and skips the Init'ing part of the object lifecycle (an Init method isn't mandatory).

Now, it gets a little deeper. If you perform your obj_new() command as above and with the unwanted behavior, and then try to compile the file by hand, you'll see TestClass1::Init get compiled, and perhaps you think to yourself that *now* you may try OBJ_NEW('TestClass1',65) again and get the expected results. Not so. See http://www.dfanning.com/tips/init_method.html for an explanation of this curious phenomenon.

In any case, simply move your TestClass1__define procedure to the end.

Also, the return from Init as shown is "TRUE", which IDL will interpret as an undefined variable, and silently fail on obj_new(), since anything returned which doesn't evaluate to the scalar value 1 indicates an error, and obj_new doesn't actually create an object. The manual is to be blamed, saying:

INIT should return a scalar TRUE value (such as 1) if the initialization is successful, and FALSE (such as 0) if the initialization fails.

I think they were trying to say we could also return a variable, or the result of some calculation, as long as it evaluated to 0 or 1... e.g.

return, a gt b?1:0

Maybe the documentation needs a beta cycle too...

In anycase, just return 1 instead.

Good Luck,

JD

--

J.D. Smith |*| WORK: (607) 255-5842
Cornell University Dept. of Astronomy |*| (607) 255-6263
304 Space Sciences Bldg. |*| FAX: (607) 255-5875
Ithaca, NY 14853 |*|