Subject: Re: It seems that there is *a bit* polymorphism in IDL.
Posted by Mark Hadfield on Wed, 11 Aug 2004 21:34:30 GMT
View Forum Message <> Reply to Message

Y.F. Tian wrote:
> I don't know if someone had ever noticed that what will happen if the
> procedure and the function have the same name. My conclusion is that
> IDL can figure out which one I am calling - the procedure or the
> function. This may be a bit useful in some cases.

You are right, it is quite permissible in IDL to have a procedure and a function of the same name. CALL_METHOD is an example of such a pair (the only one I'm aware of in the standard IDL library). Furthermore I think this is a potentially useful feature, though I am sure many, if not most, other IDL programmers will disagree (soon).

However there is a major problem, and it relates to the automatic compilation. Let's say you have your msgbox procedure and function in a file called msgbox.pro. Let's assume that the procedure is first and the function is second (as in your example). If you compile that file manually, you will have both the function and the procedure. OK, so what happens if you put msgbox.pro somewhere on your !PATH so that it will be compiled automatically. Now restart IDL and call the msgbox function. IDL works through the file compiling everything on the way and both the function and the procedure are available. Now restart IDL and call the procedure first. IDL works throught the file until it gets to the procedure. It compiles that & exits *without* compiling the function. As far as I am aware, nothing you can do, short of manually recompiling the file, will persuade IDL to compile the function. You can reverse the order of the procedure and the function, but then you just have the same problem in reverse.

One solution is to have a startup file in which you compile things manually, with a command like

```
  resolve_routine, 'msgbox', /COMPILE_FULL_FILE
```

Me, I don't think it's worth the bother.

Note that the above difficulty does not occur with object methods. It is perfectly OK to have two object methods of the same name, one a function and one a procedure. They will both get compiled when <object>__define.pro is compiled. I've been doing this for years.

--
Mark Hadfield "Ka puwaha te tai nei, Hoea tatou"
m.hadfield@niwa.co.nz