Subject: Re: Inheritance query
Posted by davidf on Thu, 04 Nov 1999 08:00:00 GMT
View Forum Message <> Reply to Message

Bernard Puc (bpuc@va.aetc.com) writes:

> For the object programming gurus: I'm creating a class called data.
> I'm then creating subclasses of data called type1, type2, etc. The
> type1 class inherits the data class attributes. Now, is it possible to
> inherit, lets say, the data::INIT method and somehow add to it? Or, do
> I have to write an entirely new INIT method for type1 class which
> incorporates the statements in the data::INIT method?

If you don't define an INIT method for, say, your TYPE1 object, IDL will look for an INIT method in a subclass object and use that. If you decide that you would like to add to an INIT method for the TYPE1 object, and still use the INIT method of a superclass, you can call the superclass INIT method from within the TYPE1 INIT method. (The only time you can call a lifecycle method yourself is from within a subclass lifecycle method.) It might look something like this:

```
FUNCTION TYPE1::INIT, _Extra=extra
ok = self->DATA::INIT(_Extra=extra)
IF NOT ok THEN RETURN, 0
```

Note this really strange behavior: If you don't create an INIT method for your TYPE1 object and then create a TYPE1 object, it will, of course, use the INIT method of the inherited DATA object. But if you now decide to write a TYPE1::INIT method, you will never to able to attach it to a TYPE1 object in that IDL session! You will have to exit IDL and restart for the correct INIT method to be associated with the TYPE1 object.

I'm just about done with an article explaining this phenomemon in more detail. It should be up on my web page later today.

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting
Phone: 970-221-0438 E-Mail: davidf@dfanning.com
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Toll-Free IDL Book Orders: 1-888-461-0155
