Subject: IDL object copying
Posted by Matt Francis on Thu, 28 Jul 2011 05:49:07 GMT
View Forum Message <> Reply to Message

IDL seems to have an unexpected behaviour when dealing with custom objects.

If I have an object, say obj1 and I then say:

obj2 = obj1

I would expect to create a copy of obj1. Any changes to members of obj2 I made through its methods should not change obj1, but that it not what I find. In fact any changes made to one changes the other.

Looking at the details of the two variables we have (note that the class of the object is called DATETIME)

OBJ1 OBJREF = <ObjHeapVar829(DATETIME)>
OBJ2 OBJREF = <ObjHeapVar829(DATETIME)>

IDL has decided that both of these are in fact pointers to the same heap memory, rather than being different instances of the same class, without their own heap memory. This is bad design; it is not the expected behaviour of the assignment operator, and IDL provides no mechanism to distinguish copy construction from assignment!

Can anyone suggest a workaround? I am in the midst of re-writing an IDL code using custom objects (I am C++ developer by preference) and am starting to realise the severe limitations of IDL's object implementation!