
Subject: Re: do I really need to use loops on objects?
Posted by [Craig Markwardt](#) on Wed, 01 Dec 1999 08:00:00 GMT
[View Forum Message](#) <> [Reply to Message](#)

bjackel@phys.ucalgary.ca writes:

> Brad Gorn wrote:

>>

>> It seems odd to me that I can't treat object arrays with the same
>> elegance that is possible with all other array types. If I have a large
>> list of objects of the same type, and I want to call the same method on
>> each of them, do I really have to use a for loop? Wouldn't it be more
>> consistent with the IDL philosophy to write: object_array->method()
>> instead of: for i=0,10 do object_array[i]->method()
>

> I agree completely, but most people on this group don't appear to.
>

> The counter argument seems to be that an object array can contain
> different kinds of objects, so you can't be sure that any particular
> method will work for every element of the array. While true, this
> seems (to me) to be a reason why object arrays should contain only
> similar elements **JUST LIKE EVERY OTHER IDL ARRAY TYPE**. If you
> want different kinds of objects wrapped up together then use an
> array of pointers, or a structure with different object members.

One of the points of object oriented programming is **polymorphism**.
Which means, in short, that the same function can be applied to
different object types (possibly with differing implementations for
each). That's the great idea of keeping a array of heterogeneous
objects around.

So, object_array->method() **should** call "method" for each object in
object_array, irregardless of the type of the object. If one of the
objects doesn't define "method" then an exception should be raised.
[Following Smalltalk or Objective C there should be some way to query
an object to see if it responds to a particular method].

Not that I use objects much, but this is my sense of rightness.

Craig

--

Craig B. Markwardt, Ph.D. EMAIL: craigmnet@cow.physics.wisc.edu
Astrophysics, IDL, Finance, Derivatives | Remove "net" for better response
