Subject: Re: Top 10 IDL Requests
Posted by Craig Markwardt on Thu, 10 Aug 2000 07:00:00 GMT
View Forum Message <> Reply to Message

ottehoman@my-deja.com writes:

> In article <397df38e.0@news.nwl.ac.uk>,
> wmc@bas.ac.uk wrote:
> ;[other stuff]
>
>> Ah, no, thats not enough. I want, say,
> structures which represent a data
>> agglomeration (I'm trying not to say object)
> with loads of header fields
>> the same, but a few fields (maybe just the one
> "data" field) different.
>> So the storage is different. I know I could do
> this by putting a pointer
>> into the structure instead, but... I can't see
> why IDL shouldn't do this itself.
>> I guess I'm assuming that, when IDL stores an
> array of strucutures, it doesn't
>> store the structures consecutively anyway - just
> pointers to the structures.
>> In which case, it shouldn't matter what the
> structure types are. I think.
>
> William, David,
>
> I have the same problem - I'd like to create a
> hierarchical data structure, basically an array,
> of structures (so I can use DataFile(1),
> DataFile(2), etc...) Each structure consists of
> headers, parameters, and *dynamic* arrays. So the
> structures are *more or less* the same, but
> differ in the lenght of their arrays. Using IDL
> 5.3 (sorry, our site has only this
> version licensed) the contents of an array can
> only be of one single type. My filestructures
> are like this: {{header},{parameters},{data}},
> where {data} is a structure with arrays of (from
> file to file) different lengths. I only know at
> runtime how long these arrays are.

I hear your problems. I run into the same kinds of things myself. It turns out that all structures must have exactly the same format, including the number of elements in every array. I believe that arrays of structures *are* in fact stored in a single contiguous

memory block, except for strings. If you ever notice that manipulations with large structures are very slow, that's why.

To get to the solution, I think David, JD, etc are right. You need to use pointers. If you need compatibility with IDL 4, then you can use handles. You get the benefit of both pointers and handles via Liam Gumley's POINTER_* family of functions.

I would recommend that to start you use as few pointers as you can get away with. Unlike normal IDL variables, there are real penalties for forgetting to free a pointer when you are done. One possibility is to wrap all of your variable sized data from a file into a single anonymous structure, and make a single tag in your fixed-sized structure point to it. As I said, large amounts of data in a structure tend to get very sloooow though.

Craig

--

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