
Subject: Re: object oriented dilemma

Posted by [davidf](#) on Mon, 09 Oct 2000 07:00:00 GMT

[View Forum Message](#) <> [Reply to Message](#)

Jason Li (jylimd@yahoo.com) writes:

```
> If I want to write an image processing software using OOP method, I would
> normally define a structure in __define module:
>
> PRO myProgram__define
>   struct = {imagePointer:Ptr_New()}
> END
>
> Traditionally, a pointer is used to take care of variable image size. Then
> load an image to self.imagePointer in the myProgram__init module.
>
> hugeImage = bytarr(huge, huge)
> self.imagePointer = Ptr_New(hugeImage, /no_copy)
>
>
> Now in my METHOD modules, I have to perform an operation on line by line
> basis. I don't know how to get a line data out of this self.imagePointer
> without making a copy (dereferencing) of it first. My image size is rather
> large. I don't how to be more memory efficient.
>
> Question: What is the solution to this?
```

I presume you mean you are taking a line of your image like this:

```
line = (*self.imagePointer)[5,*]
```

I find it hard to believe (given what I know about how pointers work in IDL) that this takes any more memory than this does:

```
line = self.image[5,*]
```

What evidence do you have that a huge amount of memory is being used?

Everything I know about pointer variables convinces me that they are really treated like any other IDL variable inside of IDL. Evidence to the contrary would be depressing. :-(

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
