Subject: Re: The best way to keep data in RAM / object-oriented programming
Posted by ben.bighair on Thu, 03 Dec 2009 17:09:24 GMT
View Forum Message <> Reply to Message

On Dec 3, 10:44 am, nata <bernat.puigdomen...@gmail.com> wrote:
> Hi gurus,
>
> My application needs a lot of RAM so I have to improve a solution in
> order to to use less resources.
> Normally, when I've an object I use declarations like this:
>
> pro myobject__define
> struct = { myobject, $
> data: ptr_new() }
> end
>
> Now for example if my object has to save an array like FLTARR
> (400,400,24,97) I will use the data pointer to store this array. The
> problem is that if I do this I take a lot of computer resources. With
> this example:
> help, /mem
> aa=fltarr(400,400,24,97)
> myobject->SetProperty, DATA=aa
> aa=0l
> help, /mem
>
> IDl returns:
> heap memory used: 658906, max: 805215874, gets: 1195,
> frees: 387
> heap memory used: 1490578946, max: 1490579037, gets: 1207,
> frees: 398
>
> So, only for this example I'm using 1.4 Gb aprox. I tried to used
> ASSOC procedure but I didn't succeed.... Some suggestions or comments
> about how to reduce the memory ? There is a method to store compressed
> data or something similar ?

Hi,

You don't show how you assign your array to a pointer reference, but
you might see a some gain in using the /NO_COPY keyword for PTR_NEW()

Another thought is to allow your SetProperty method to accept a
pointer to start with so that you reduce the transfer costs passing
the data into the method call. Maybe something like the following
will work as a switch hitter for you. Keep in mind I have written an
pointer handler in a long time*, so I might have some of the logic

askew.

```
PRO MyObject::SetProperty, data = data, ...

if N_ELEMENTS(data) GT 0 then begin
  if (SIZE(data, /TYPE)) EQ 10) then begin
    if PTR_VALID(self.data) then PTR_FREE, self.data
    self.data = data
  endif else begin
    if PTR_VALID(self.data) then $
      *self.data = data else $
      self.data = PTR_NEW(data, /NO_COPY)
  endif
endif

END; SetProperty
```

Then you could do

```
aa = PTR_NEW(fltarr(400,400,24,97), /NO_COPY)
myobject->SetProperty, DATA=aa
```

Some folks might squeak (rightly so) that this approach might allows you sneaky and direct access to an object property - thus exposing the innards when you shouldn't and breaking the spirit of encapsulation. But, why should we be confined by well-reasoned and established common practice?

I am quite sure that there are other places for you to gain more on memory management but that gets above my pay grade.

Cheers,
Ben

* In fact, my IDL is so rusty that I kept trying to type the program blocks using { }. Oops!
