Subject: Re: IDL objects and names
Posted by David Fanning on Wed, 12 Feb 2003 13:33:01 GMT
View Forum Message <> Reply to Message

Sebastian (s@visita2.die.upm.es) writes:

> when I create an IDL object, I can give it a name, like
> oImg = OBJ_NEW('IDLgrImage',Name='myImage').
>
> I thought that this might be nice for debugging, but neither help nor
> heap_gc,/VERBOSE display the names of the objects.
>
> Is there a way to make heap_gc and help display the names?
> Are there any other techniques to get more info about objects that are
> removed by heap_gc?
> I mean, with an heap_gc,/VERBOSE output like
> <ObjHeapVar37> STRUCT   = -> IDLGRPALETTE Array[1]
> I can't do much when I have a lot of IDLgrPalettes around.

Yes, it can be discouraging after you write your first couple of object graphics programs to get that e-mail from an important colleague suggesting that your programs might be leaking a bit of memory. Leaking!? More like a memory sieve! :-(

What to do with that huge long list of objects left on the heap? First, of course, is to start paying more attention to memory management as you write your programs. You will do this automatically in a few more weeks. Human beings can only take so much pain. :-)

The second step is to try to figure out where these leaks are coming from. If you have had the foresight to name your objects, you are in pretty good shape. First, you will have to recover the objects. Here is an example using two palette objects:

```
   pal1 = Obj_New('IDLgrPalette', Name='GrayScale')
   pal2 = Obj_New('IDLgrPalette', Name='Std Gamma')
   pal2 -> LoadCT, 5
   pal1 = 0
   pal2 = 0
```

Ok, now my palettes are lost on the heap:

```
   IDL> Help, /Heap
      Heap Variables:
         # Pointer: 2
         # Object : 2
```

```
    <ObjHeapVar2>   STRUCT    = -> IDLGRPALETTE Array[1]
    <ObjHeapVar3>   STRUCT    = -> IDLGRPALETTE Array[1]
    <PtrHeapVar4>   BYTE      = Array[768]
    <PtrHeapVar5>   BYTE      = Array[768]
```

Notice there are pointers, too, probably associated with the palettes. (At least I *hope* they are!)

The first step is to recover the palette objects. We can do this with OBJ_VALID and no arguments:

```
  IDL> lostObjects = Obj_Valid()
  IDL> Help, lostObjects
    LOSTOBJECTS     OBJREF    = Array[2]
```

Now we can just get and print the names of the objects:

```
  IDL> FOR j=0,N_Elements(lostObjects)-1 DO BEGIN $
         lostObjects[j] -> GetProperty, Name=theName &$
         Print, theName & ENDFOR

     GrayScale
     Std Gamma
```

Or, because that darn FOR loop is so hard to write, I just might add a PrintName method to the IDLgrPalette object.

```
 PRO IDLgrPalette::PrintName
   Print, self.name
 END
```

Save this as idlgrpalette__printname.pro. Then, just type this:

```
  IDL> FOR j=0,N_Elements(lostObjects)-1 DO lostObjects[j] -> PrintName
     GrayScale
     Std Gamma
```

Now, maybe, you have some clue about where these darn things come from. :-)

Cheers,

David

P.S. Of course, you must also destroy these objects.

```
   IDL> Obj_Destroy, lostObjects
   IDL> Help, /Heap
      Heap Variables:
          # Pointer: 0
          # Object : 0
```

Hooray, we got those pesky pointers, too! :-)

--
David W. Fanning, Ph.D.
Fanning Software Consulting, Inc.
Phone: 970-221-0438, E-mail: david@dfanning.com
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Toll-Free IDL Book Orders: 1-888-461-0155