Subject: Re: How to prevent proliferation of leftover objects when NG plots are included in unit tests.
Posted by Paul Van Delst[1] on Tue, 14 Aug 2012 22:52:15 GMT
View Forum Message <> Reply to Message

Hello,

On 08/14/12 16:33, Michael Galloy wrote:
> On 8/14/12 1:56 PM, Paul van Delst wrote:
>> However, when I include a PLOT() function call in a unit test (just to have a looksee at the pretend data I create), I
>> have thousands of leftover objects, all NG related:
>>
>> IDL> help, obj_valid()
>> <Expression> OBJREF = Array[4796]
>>
>> Printing out the first 11 objects:
>>
>> IDL> for i=0,10 do help, (obj_valid())[i],/full
>> <Expression> OBJREF = <ObjHeapVar175(IDLITSYSTEM)> refcount=154
>> <Expression> OBJREF = <ObjHeapVar176(IDLITPROPERTYDESCRIPTOR)> refcount=2
>> <Expression> OBJREF = <ObjHeapVar177(IDL_CONTAINER)> refcount=1
>> <Expression> OBJREF = <ObjHeapVar179(IDLITPROPERTYDESCRIPTOR)> refcount=2
>> <Expression> OBJREF = <ObjHeapVar181(IDLITCONTAINER)> refcount=4
>> <Expression> OBJREF = <ObjHeapVar182(IDLITPROPERTYDESCRIPTOR)> refcount=2
>> <Expression> OBJREF = <ObjHeapVar183(IDL_CONTAINER)> refcount=1
>> <Expression> OBJREF = <ObjHeapVar185(IDLITPROPERTYDESCRIPTOR)> refcount=2
>> <Expression> OBJREF = <ObjHeapVar188(IDLITCONTAINER)> refcount=3
>> <Expression> OBJREF = <ObjHeapVar189(IDLITPROPERTYDESCRIPTOR)> refcount=2
>> <Expression> OBJREF = <ObjHeapVar190(IDL_CONTAINER)> refcount=1
>>
>> When I kill the plot window, I still have a great many leftovers:
>>
>> IDL> help, obj_valid()
>> <Expression> OBJREF = Array[1395]
>> IDL> for i=0,10 do help, (obj_valid())[i],/full
>> <Expression> OBJREF = <ObjHeapVar175(IDLITSYSTEM)> refcount=151
>> <Expression> OBJREF = <ObjHeapVar176(IDLITPROPERTYDESCRIPTOR)> refcount=2
>> <Expression> OBJREF = <ObjHeapVar177(IDL_CONTAINER)> refcount=1
>> <Expression> OBJREF = <ObjHeapVar179(IDLITPROPERTYDESCRIPTOR)> refcount=2
>> <Expression> OBJREF = <ObjHeapVar181(IDLITCONTAINER)> refcount=3

>> ....etc....
>>
>> My question is: How do I prevent this proliferation of NG plot objects? Or, at the very least, how do I safely delete
>> them?
>>
>
> Glad you like mgunit!

Oh yeah - for IDL stuff that is used for producing actual numbers (as opposed to just pictures), I always use your unit
testing setup. In just the last two days I've caught at least 10 bugs (I would actually call them "development
inconsistencies" but then everyone on this ng would give me a hard time... :o)

For development of IDL code that makes heavy use of objects, your MGunit package is a must for me.

> Using the close method (or closing the graphics window, in the case BUFFER=0) works for me from the command line:
>
> IDL> help, /heap
> Heap Variables:
>   # Pointer: 0
>   # Object : 0
> IDL> p = plot(/test, /buffer)
> IDL> p.close
> IDL> help, /heap
> Heap Variables:
>   # Pointer: 0
>   # Object : 0
>
> Does this happen for you? Is it just inside the testing framework that is causing problems?

No, that does no happen for me, no, the unit testing framework is not involved in this object explosion. Regarding the
latter, if that's how my original post came across, then let me be very clear to other potential user of MGunit: It is
not the unit testing framework that is causing this; it is a congenital problem of IDL.

The reason I even mentioned the unit testing is because calling plot() in multiple unit tests causes the creation of
10,000's of object references (as opposed to the couple of thousand if I do it on the command line).

When I follow your example I get the results below:

IDL> help, /heap

```
Heap Variables:
   # Pointer: 0
   # Object : 0
IDL> p = plot(/test, /buffer)
IDL> p.close
IDL> help, /heap
Heap Variables:
   # Pointer: 1168
   # Object : 1395

<ObjHeapVar102>  refcount=152
            STRUCT    = -> IDLITSYSTEM Array[1]
<ObjHeapVar103>  refcount=1
            STRUCT    = -> IDLITPROPERTYDESCRIPTOR Array[1]
<ObjHeapVar104>  refcount=0
            STRUCT    = -> IDL_CONTAINER Array[1]

...etc...
```

So, I guess my 8.1 version of IDL has a bug in its garbage collection (now that I think of it, I recall this issue being
raised at some point in the past in this ng....)

I'm scheduled for an upgrade to v8.2 (which I assume you have?) so hopefully that will solve these function graphics
object infestations.

Thanks for the confirmation. And MGunit. :o)

cheers,

paulv