Subject: sec : U Re: Proper pointer cleanup question
Posted by Andrew Cool on Wed, 09 Apr 2003 01:34:58 GMT
View Forum Message <> Reply to Message

Paul van Delst wrote:
>
>
> <snip>
>
>> Another option is:
>>
>> heap_free,a
>>
>
> Wha..? Is that another one of those undocumented IDL routines? It works on my current
> version, but bugger me if I can find it documented anywhere.
>
> paulv
>
> --

Paul,

Ben Tucker's CoolHelp says :-

HEAP_FREE
The HEAP_FREE procedure recursively frees all heap variables (pointers or objects) referenced by its input argument. This routine examines the input variable, including all array elements and structure fields. When a valid pointer or object reference is encountered, that heap variable is marked for removal, and then is recursively examined for additional heap variables to be freed. In this way, all heap variables that are referenced directly or indirectly by the input argument are located. Once all such heap variables are identified, HEAP_FREE releases them in a final pass. Pointers are released as if the PTR_FREE procedure was called. Objects are released as with a call to OBJ_DESTROY.
As with the related HEAP_GC procedure, there are some disadvantages to using HEAP_FREE such as:
* When freeing object heap variables, HEAP_FREE calls OBJ_DESTROY

without supplying any plain or keyword arguments. Depending on the objects being released, this may not be sufficient. In such cases, the caller must call OBJ_DESTROY explicitly with the proper arguments rather than using HEAP_FREE.

* HEAP_FREE releases the referenced heap variables in an unspecified order which depends on the current state of the internal data structure used by IDL to hold them. This can be confusing for object destructor methods that expect all of their contained data to be present. If your application requires a specific order for the release of its heap variables, you must explicitly free them in the correct order. HEAP_FREE cannot be used in such cases.

* The algorithm used by HEAP_FREE to release variables requires examination of every existing heap variable (that is, it is an O(n) algorithm). This may be slow if an IDL session has thousands of current heap variables.

For these reasons, Research Systems recommends that applications keep careful track of their heap variable usage, and explicitly free them at the proper time (for example, using the object destructor method) rather than resorting to simple-looking but potentially expensive expedients such as HEAP_FREE or HEAP_GC.

HEAP_FREE is recommended when:

* The data structures involved are highly complex, nested, or variable, and writing cleanup code is difficult and error prone.

* The data structures are opaque, and the code cleaning up does not have knowledge of the structure.

Syntax

HEAP_FREE, Var [, /OBJ] [, /PTR] [, /VERBOSE]

Arguments

Var

The variable whose data is used as the starting point for heap variables to be freed.

Keywords

OBJ

Set this keyword to free object heap variables only.

PTR
Set this keyword to free pointer heap variables only.

Note
Setting both the PTR and OBJ keywords
is the same as setting neither.

VERBOSE
If this keyword is set, HEAP_FREE writes
a one line description of each heap
variable, in the format used by the HELP
procedure, as the variable is released. This
is a debugging aid that can be used by
program developers to check for heap variable
leaks that need to be located and eliminated.

Example

```
; Create a structure variable.
mySubStructure = {pointer:PTR_NEW(INDGEN(100)), $
  obj:OBJ_NEW('Idl_Container')}
myStructure ={substruct:mySubStructure, $
  ptrs:[PTR_NEW(INDGEN(10)), PTR_NEW(INDGEN(11))]}

;Look at the heap.
HELP, /HEAP_VARIABLES

; Now free the heap variables contained in myStructure.
HEAP_FREE, myStructure, /VERBOSE
HELP, /HEAP_VARIABLES
```

See Also
HEAP_GC

Andrew.

---------- ----------
Andrew D. Cool
Electromagnetics & Propagation Group
Intelligence, Surveillance & Reconnaissance Division
Defence Science & Technology Organisation
PO Box 1500, Edinburgh
South Australia 5111

Phone : 061 8 8259 5740 Fax : 061 8 8259 6673
Email : andrew.cool@no-spam.dsto.defence.gov.au
---------- ----------

Subject: Re: sec : U Re: Proper pointer cleanup question
Posted by btt on Thu, 10 Apr 2003 19:50:27 GMT
View Forum Message <> Reply to Message

Andrew Cool wrote:
>
> Paul,
>
> Ben Tucker's CoolHelp says :-
>
> HEAP_FREE
> The HEAP_FREE procedure recursively frees

Hey Andrew, what a coincidence! My version of KookHelp says the same thing!

Cheers,
Ben