
Subject: Re: use of 'obj_new' within another object definition
Posted by [Randall Skelton](#) on Mon, 18 Nov 2002 23:03:41 GMT
[View Forum Message](#) <> [Reply to Message](#)

Hi Paul,

I think you might have cut a little too much 'object definition stuff' from your example... in any case, if I understand what you are trying to do, then try the code examples below.

```
IDL> a = obj_new('other_object')
% Compiled module: OTHER_OBJECT__DEFINE.
i am here!
IDL> b = obj_new('my_object')
% Compiled module: MY_OBJECT__DEFINE.
i am here!
IDL> help, /heap
Heap Variables:
  # Pointer: 0
  # Object : 3
<ObjHeapVar1>  STRUCT  = -> OTHER_OBJECT Array[1]
<ObjHeapVar2>  STRUCT  = -> MY_OBJECT Array[1]
<ObjHeapVar3>  STRUCT  = -> OTHER_OBJECT Array[1]
IDL> obj_destroy, a
IDL> obj_destroy, b
IDL> help, /heap
Heap Variables:
  # Pointer: 0
  # Object : 0
```

```
;-----
pro my_object::cleanup
  if obj_valid(self.foo) then obj_destroy, self.foo
  ; NB: don't try and shortcut the life-cycle by calling
  ; self.foo->cleanup directly...
end
```

```
function my_object::init
  self.foo = obj_new('other_object')
  return, 1
end
```

```
pro my_object__define
  s = { my_object, foo: obj_new() }
end
;-----
```

EXAMPLE ONE: Use obj_new()

```

^^^^^^^^^^^^^^^^
;-----
pro other_object::cleanup
    ; nothing to do here yet
end

function other_object::init
    print, ' i am here!'
    return, 1
end

pro other_object__define
    s = { other_object, a: 0 }
end
;-----

EXAMPLE TWO: Use ptr_new()
^^^^^^^^^^^^^^^^
;-----
;my_object__define.pro

pro my_object::cleanup
    if ptr_valid(self.foo) then begin
        obj_destroy, *self.foo
        ptr_free, self.foo
    endif
end

function my_object::init
    self.foo =
ptr_new(obj_new('other_object'))
    return, 1
end

pro my_object__define
    s = { my_object, foo: ptr_new() }
end
;-----

```

On 18 Nov 2002, paul wrote:

```

> Hi,
> I am experiencing a problem in IDL 5.4.
>
> I create an object called 'my_object', and define it in a file called
> 'my_object__define.pro'

```

```

>
> I do the same thing with 'other_object' , and
> 'other_object__define.pro'.
>
> Both of these objects compile and i can create variables with them.
> They both have 'init' functions that do basic initialization stuff.
>
> I know that the 'init' functions are getting called because I can
> put...
>
>   print,'i am here ...'
>
> ...statements in the init functions, and I will see the output when
> i initialize an object of that type.
>
> ...
>
>
> Now, my problem is that if I want 'my_object' to have an instance of
> 'other_object' as a member object, the init function( of
> 'other_object')
> doesn't get called.
>
> To elaborate...(snipping irrelevant stuff)
>
> ;-----
> ;my_object__define.pro
>
> pro my_object__define
>
>   foo: obj_new('other_object')
>
> end ; my_object
>
>
> ;-----
> ;other_object__define.pro
>
> function other_object::init
>   print, ' i am here!'
>   return, 1
> end ; init
>
> pro other_object__define
>
>   ;object definition stuff here
>
> end ; other_object

```

> ;-----
>
>
>
> When type ' x = obj_new('other_object') ' on the
> command line, I see the ' i am here!'.
>
> When type ' y = obj_new('my_object') ' on the
> command line, I DON'T see the ' i am here!'.
>
> Why is this?
>
> I don't get any errors otherwise, and
> according to the documentation on 'obj_new'
> it seems it should call the 'init' function
> in BOTH circumstances.
>
> Any ideas greatly appreciated,
>
> Paul Wisehart
> wisehart@runbox.com
