
Subject: Re: IDL 5.0 forgets about open windows (?)
Posted by [J.D. Smith](#) on Thu, 24 Jul 1997 07:00:00 GMT
[View Forum Message](#) <> [Reply to Message](#)

```
>
>>
>> David Foster wrote:
>>>
>>> Version: { sparc sunos unix 5.0 Apr 28 1997} (Solaris 2.5)
>>>
>>> Recently upon upgrading to IDL 5.0, we have noticed a strange
>>> phenomenon regarding graphics windows. In one of our programs,
>>> IDL seems to "forget" that there are any windows open, and the
>>> next cursor operation results in window #0 being created. This
>>> happens while the program has 4 visible windows displayed onscreen!
>>>
>>> The program is quite complex, and the problem occurs after various
>>> operations involving setting and unsetting windows, so it is *possible*
>>> that this is programmer error, but I don't think so because:
>>>
>>> 1) This problem did/does not occur under 4.0.1
>>> 2) At a point in the program where the error occurs, and while
>>>    there are 4 visible windows onscreen, I include the commands:
>>>
>>>     HELP, !D, /STRUCTURE
>>>     PRINT, '*****'
>>>     PRINT, '** Windows: *****'
>>>     DEVICE, WINDOW_STATE=WINDOWS
>>>     PRINT, WINDOWS
>>>
>>> which result in:
>>>
>>>     ** Structure !DEVICE, 17 tags, length=80:
>>>     NAME      STRING  'X'
>>>     X_SIZE     LONG      260
>>>     Y_SIZE     LONG      120
>>>     X_VSIZE    LONG      260
>>>     Y_VSIZE    LONG      120
>>>     X_CH_SIZE  LONG        6
>>>     Y_CH_SIZE  LONG       10
>>>     FLOAT      40.0000
>>>     Y_PX_CM    FLOAT     40.0000
>>>     N_COLORS   LONG      214
>>>     TABLE_SIZE LONG      214
>>>     FILL_DIST  LONG        1
>>>     WINDOW     LONG      -1
>>>     UNIT       LONG        0
>>>     FLAGS      LONG     65980
```

```

>>> ORIGIN      LONG   Array[2]
>>> ZOOM        LONG   Array[2]
>>> *****
>>> ** Windows: *****
>>> 0 0 0 0 0 0 0 0 0 0 0 0 0
>>> 0 0 0 0 0 0 0 0 0 0 0 0 0
>>> 0 0 0 0 0 0 1 1 1 1 0 0 0
>>>
>>> Should this ever be possible, where !D.WINDOW = -1 but there are
>>> four windows currently available? I have applied a workaround to
>>> this program, in which I wset to one of the available windows,
>>> but I'm wondering where else this behavior might pop up. Any ideas?
>>>
>>> Dave
>>> --
>>>
>>> ~~~~~
>>> David S. Foster      Univ. of California, San Diego
>>> Programmer/Analyst  Brain Image Analysis Laboratory
>>> foster@bials1.ucsd.edu Department of Psychiatry
>>> (619) 622-5892      8950 Via La Jolla Drive, Suite 2200
>>>                      La Jolla, CA 92037
>>> ~~~~~
>>>
>>> "I have this theory that if we're told we're bad,
>>>     then that's the only idea we'll ever have.
>>>     But maybe if we are surrounded in beauty,
>>>     someday we will become what we see." - Jewel Kilcher
>>>
>>>
>>> Although I don't have a solution to your window dilemma, I am interested
>>> in what gave rise to it. I have a situation in which I would like to
>>> set !D.WINDOW = -1 after a window has been opened. For instance, when
>>> opening a colormap editor with a draw window color-bar the general thing
>>> to do is:
>>>
>>> oldwin=!D.WINDOW
>>> wset,colobarwin
>>> ..... ;draw colorbar
>>> .....
>>> wset,oldwin
>>>
>>> but if oldwin=-1 you can't do this. If I could test for this, and then
>>> somehow "undefine" the present window, this would be ideal, and it
>>> sounds like this is just what's happening in your code. Anyway, if you
>>> do figure it out, I'd be very interested in knowing how it was
>>> accomplished.
>>>

```

```
>> JD
>>
>
> Maybe I don't understand your objective, but my point was that if
> !D.WINDOW = -1 then by definition there should be no windows present.
> If there are then this is a bug.
>
> Could you "undefine" the present window using WDELETE? Or do you mean
> to set !D.WINDOW=-1 but still have the window available. I believe that
> this should not be possible. In my case I think I have run into a bug.
> I still don't quite understand why you would want such behavior to
> be a feature.
>
> Dave
```

You *could* delete the window, but I need the window open. The reason I would want this "feature" is the same reason that we save the old window and reset it. But the problems are magnified in IDLv5 where an active command line presents the possibility of graphics commands being executed while the undesired window is active. It's just like the familiar technique of setting to the "oldwin" -- i.e. you don't want any more graphics output directed to the colorbarwin (or whatever). But if oldwin is non-existent, there is no way to prevent new graphics commands from going to your colorbarwin, except for creating a new window, which is inelegant. For instance, suppose you launch a colormap editor as a non-blocking widget. No other windows are open. Now you enter a graphics command like `tv,dist(100)`. Where does the command send its output? To your `widget_draw` colorbar, unless you've created a new window for the command, which, as I've already said, I consider to be an undesirable solution.

For now I use the code:

```
if oldwin eq -1 then window else wset,oldwin ;ensure cbar win isn't
current
```

JD
