Subject: Another small V8.0 bug
Posted by wlandsman on Mon, 26 Jul 2010 17:12:20 GMT
View Forum Message <> Reply to Message

I have found another bug in V8.0, at least for users who still have round parenthesis used for indices lurking around in their code. Like Mike Potter's example, it is not easily repeatable, and for example sometimes only occurs after compile the program a second time. And because it occurs in a fairly large program, I have not yet isolated it into a simple test program. But I can illustrate the problem after placing a STOP statement

```
% Stop encountered: SHOWDB            72 /home/landsman/uvot/bpm16274/
showdb.pro
IDL> help,list
LIST            LONG      = Array[1]
IDL> print,list[0]
       183
IDL> print,list(0)
      0
IDL> help,list(0)
<Expression>    LIST  <ID=22  NELEMENTS=1>
IDL> print,list(0) LE 0
% Unable to convert variable to type object reference.
% Execution halted at: SHOWDB            72 /home/landsman/uvot/
bpm16274/showdb.pro
%                    $MAIN$

   IDL> print,!version
{ x86 linux unix linux 8.0 Jun 18 2010      32      64}
```

So IDL seems confused as to whether 'list' is a variable or an object. (The code is all imperative statements with no object syntax). Note that this differs from the long-standing variable/function ambiguity that can occur when using the () syntax for indexing. --Wayne

Subject: Re: Another small V8.0 bug
Posted by wlandsman on Thu, 29 Jul 2010 15:26:04 GMT
View Forum Message <> Reply to Message

On Jul 28, 5:53 pm, JD Smith <jdtsmith.nos...@yahoo.com> wrote:

> Haven't managed to download 8 yet, but people should also be aware of
> the increased potential for namespace conflict (and parentheses
> indexing variable/function confusion) arising from the new syntax for
> object instantiation. I.e. if you have any functions (or

> parenthetically indexed variable in the same scope) of the same name
> as an object class, you will get new conflicts. This "list" is just
> an example of that.

Anyone know if there is a way to list all the object instantiation
functions (intrinsic or otherwise) that IDL knows about? I was a
bit surprised to find that ROUTINE_INFO(/SYSTEM,/FUNCTION) does not
include the list() function. --Wayne

Subject: Re: Another small V8.0 bug
Posted by penteado on Thu, 29 Jul 2010 17:43:23 GMT
View Forum Message <> Reply to Message

On Jul 29, 12:26 pm, wlandsman <wlands...@gmail.com> wrote:
> Anyone know if there is a way to list all the object instantiation
> functions (intrinsic or otherwise) that IDL knows about? I was a
> bit surprised to find that ROUTINE_INFO(/SYSTEM,/FUNCTION) does not
> include the list() function. --Wayne

As I understand it, routine_info(/system) only shows those that are
not implemented in IDL. I did notice that many init functions show up:
even though, say, IDLFFXMLDOMDOCUMENT() does not appear, its actual
function, IDLFFXMLDOMDOCUMENT::INIT, does. And init functions I
defined in DLMs also show up. So the overload does not show, but the
init functions that are overloaded do. List's init does not show
because it is written in IDL.

Subject: Re: Another small V8.0 bug
Posted by wlandsman on Thu, 29 Jul 2010 18:35:23 GMT
View Forum Message <> Reply to Message

On Jul 29, 1:43 pm, Paulo Penteado <pp.pente...@gmail.com> wrote:
> On Jul 29, 12:26 pm, wlandsman <wlands...@gmail.com> wrote:
>
>> Anyone know if there is a way to list all the object instantiation
>> functions (intrinsic or otherwise) that IDL knows about? I was a
>> bit surprised to find that ROUTINE_INFO(/SYSTEM,/FUNCTION) does not
>> include the list() function. --Wayne
>
> As I understand it, routine_info(/system) only shows those that are
> not implemented in IDL. I did notice that many init functions show up:
> even though, say, IDLFFXMLDOMDOCUMENT() does not appear, its actual
> function, IDLFFXMLDOMDOCUMENT::INIT, does. And init functions I
> defined in DLMs also show up. So the overload does not show, but the
> init functions that are overloaded do. List's init does not show

> because it is written in IDL.

Thanks. Note that being written in IDL does not mean we can see the code -- the list and hash functions are stored as IDL save files in the new /idl80/lib/datatypes/ directory. --Wayne

---