Subject: Re: Another small V8.0 bug
Posted by penteado on Mon, 26 Jul 2010 17:59:24 GMT
View Forum Message <> Reply to Message

On Jul 26, 2:12 pm, wlandsman <wlands...@gmail.com> wrote:
> 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}

Do you mean that this does not happen every time? To me this seems to
be the expected behavior. The line

print,list(0)

Is creating a list (which is an object), containing one element, and
printing it. The same with the use of help.

>
> 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

Actually, this is the same old function/variable ambiguity. In this case, between the array called list, and the function called list() - which just happens to be the function that creates a list object. The only new things here is that objects can be instantiated with the syntax object=class(), which is equivalent to object=object_new('class'), and that since there is a new builtin class called 'list', that line is being interpreted as a function call.

Perhaps the variability you report is from the context, which determines whether () are interpreted as indexing or function call, since that depends on which variables are currently defined, and which functions are currently compiled.

Subject: Re: Another small V8.0 bug
Posted by wlandsman on Mon, 26 Jul 2010 18:04:27 GMT
View Forum Message <> Reply to Message

OK, I've partially isolated the problem into a 3 line program. Unfortunately, the first line is a call to a fairly long program http://idlastro.gsfc.nasa.gov/ftp/pro/database/dbopen.pro so there is a lot more isolation required (and I am done with this for the day).

```
pro test
 dbopen,'bpm16274'
list = [124]
if list(0) LE 125 then print,'Good value'
return
end


IDL Version 8.0 (linux x86 m32). (c) 2010, ITT Visual Information
Solutions
IDL> .run test
% Compiled module: TEST.
IDL> test     ;First time everything works fine
Good value

IDL> .run test     ;Let's compile it again
% Compiled module: TEST.
IDL> test
% Unable to convert variable to type object reference.
% Execution halted at: TEST                4 /home/landsman/uvot/
bpm16274/test.pro
%                $MAIN$
```

dbopen.pro does use a lot of strmid() calls -- I wonder if this could be related to the problem that Mike Potter and Paulo Penteado found. But in this case I am only using pre-V8.0 syntax so there is no FOREACH call. --Wayne

On Jul 26, 1:12 pm, wlandsman <wlands...@gmail.com> wrote:
> 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.

Subject: Re: Another small V8.0 bug
Posted by penteado on Mon, 26 Jul 2010 18:11:17 GMT
View Forum Message <> Reply to Message

On Jul 26, 3:04 pm, wlandsman <wlands...@gmail.com> wrote:
> OK, I've partially isolated the problem into a 3 line program.
> Unfortunately, the first line is a call to a fairly long
programhttp://idlastro.gsfc.nasa.gov/ftp/pro/database/dbopen .proso there is
> a lot more isolation required (and I am done with this for the
> day).
>
> pro test
> dbopen,'bpm16274'
> list = [124]
> if list(0) LE 125 then print,'Good value'

The problem is that this line is comparing a list (an object, created on this line, which contains the single value 0) with 125. It is just the confusion between using the array called list, and the function list().

Maybe I am missing something, but I do not see a bug, I see the old function/array problem, which manifested because IDL 8 happened to have a new builtin function with the same name as the array that was being used.

Subject: Re: Another small V8.0 bug
Posted by wlandsman on Mon, 26 Jul 2010 18:15:41 GMT
View Forum Message <> Reply to Message

On Jul 26, 1:59 pm, Paulo Penteado <pp.pente...@gmail.com> wrote:
> Do you mean that this does not happen every time? To me this seems to
> be the expected behavior. The line
>

> print,list(0)
>
> Is creating a list (which is an object), containing one element, and
> printing it. The same with the use of help.

OK, I did not know that list() was a new intrinsic function,
thanks. But since "list" is my most used variable name, V8.0 is not
backwards compatibile.

On the other hand, since I was a strong supporter of abandoning
support for indexing with round parentheses, I shouldn't complain too
much. But I suspect that this will be a recurring backwards
compatibilty problem.

--Wayne

Subject: Re: Another small V8.0 bug
Posted by penteado on Mon, 26 Jul 2010 18:16:20 GMT
View Forum Message <> Reply to Message

On Jul 26, 3:11 pm, Paulo Penteado <pp.pente...@gmail.com> wrote:
> On Jul 26, 3:04 pm, wlandsman <wlands...@gmail.com> wrote:
>
>> OK, I've partially isolated the problem into a 3 line program.
>> Unfortunately, the first line is a call to a fairly long
programhttp://idlastro.gsfc.nasa.gov/ftp/pro/database/dbopen .prosothere is
>> a lot more isolation required (and I am done with this for the
>> day).
>
>> pro test
>> dbopen,'bpm16274'
>> list = [124]
>> if list(0) LE 125 then print,'Good value'
>
> The problem is that this line is comparing a list (an object, created
> on this line, which contains the single value 0) with 125. It is just
> the confusion between using the array called list, and the function
> list().
>
> Maybe I am missing something, but I do not see a bug, I see the old
> function/array problem, which manifested because IDL 8 happened to
> have a new builtin function with the same name as the array that was
> being used.

By the way, this case is fortunate because the operator used is le,
which is not overloaded in that class. The following line can be
misleading, as it would not throw an error:

```
IDL> print,list(0) eq 0
  1
```

This is using the overloaded eq operator, to compare the list with 0. Which may not be very noticeable if one is reading () as allowed index operators. It would be more visible with the equivalent

```
IDL> print,list('abc') eq 'abc'
  1
```

Subject: Re: Another small V8.0 bug
Posted by penteado on Mon, 26 Jul 2010 18:24:58 GMT
View Forum Message <> Reply to Message

On Jul 26, 3:15 pm, wlandsman <wlands...@gmail.com> wrote:
> On Jul 26, 1:59 pm, Paulo Penteado <pp.pente...@gmail.com> wrote:
>
>> Do you mean that this does not happen every time? To me this seems to
>> be the expected behavior. The line
>
>> print,list(0)
>
>> Is creating a list (which is an object), containing one element, and
>> printing it. The same with the use of help.
>
> OK, I did not know that list() was a new intrinsic function,
> thanks. But since "list" is my most used variable name, V8.0 is not
> backwards compatibile.

Yes, that sort of problem was always possible whenever a new intrinsic function is introduced. Or when one installs some new third-party library, or just some function from someone who does not try to make the routine names more likely to be unique. But it is not a new issue, it has always been a consequence of the () ambiguity.

In this particular case (of list), it also compromises use of Ron Kneusel's HOF library, which depended on a DLM that also had a function called list() (http://www.ittvis.com/info/hof).

Subject: Re: Another small V8.0 bug
Posted by wlandsman on Mon, 26 Jul 2010 18:44:40 GMT
View Forum Message <> Reply to Message

On Jul 26, 2:24 pm, Paulo Penteado <pp.pente...@gmail.com> wrote:

> Yes, that sort of problem was always possible whenever a new intrinsic
> function is introduced. Or when one installs some new third-party
> library, or just some function from someone who does not try to make
> the routine names more likely to be unique. But it is not a new issue,
> it has always been a consequence of the () ambiguity.

Yes, but not only is "list" a common variable name, but since the list function can accept any data type, one won't get any error in the initial function call. I am used to seeing errors like the following

```
IDL> obj_new = indgen(10)
IDL> print,obj_new(3)
% OBJ_NEW: String expression required in this context: <INT
(      3)>.
% Execution halted at: $MAIN$
```

which tells me right away that I am calling a function named OBJ_NEW() rather than subscripting an array. But I don't believe that a = list(foo) will ever give an error, and it is only when trying to use the variable that one will encounter the (less obvious) error.

Thanks again. --Wayne

Subject: Re: Another small V8.0 bug
Posted by penteado on Mon, 26 Jul 2010 18:53:05 GMT
View Forum Message <> Reply to Message

On Jul 26, 3:44 pm, wlandsman <wlands...@gmail.com> wrote:
> which tells me right away that I am calling a function named OBJ_NEW()
> rather than subscripting an array. But I don't believe that a =
> list(foo) will ever give an error, and it is only when trying to use
> the variable that one will encounter the (less obvious) error.

Indeed. Particularly so given the introduction of !null and that lists can (and frequently need to) have !null elements, so even giving list() an undefined variable is not an error:

```
IDL> a=list(some_undefined_name)
IDL> help,a
A               LIST  <ID=1  NELEMENTS=1>
IDL> print,a
!NULL
```

Subject: Re: Another small V8.0 bug
Posted by Craig Markwardt on Mon, 26 Jul 2010 19:31:15 GMT
View Forum Message <> Reply to Message

On Jul 26, 2:15 pm, wlandsman <wlands...@gmail.com> wrote:
> On Jul 26, 1:59 pm, Paulo Penteado <pp.pente...@gmail.com> wrote:
>
>> Do you mean that this does not happen every time? To me this seems to
>> be the expected behavior. The line
>
>> print,list(0)
>
>> Is creating a list (which is an object), containing one element, and
>> printing it. The same with the use of help.
>
> OK, I did not know that list() was a new intrinsic function,
> thanks. But since "list" is my most used variable name, V8.0 is not
> backwards compatibile.

Wow, through sheer luck, I've never used an array variable named
LIST[]. I have mixed feelings about introducing such a potentially-
incompatible change into IDL.

Craig

Subject: Re: Another small V8.0 bug
Posted by Chris[6] on Mon, 26 Jul 2010 21:53:54 GMT
View Forum Message <> Reply to Message

On Jul 26, 9:31 am, Craig Markwardt <craig.markwa...@gmail.com> wrote:
> On Jul 26, 2:15 pm, wlandsman <wlands...@gmail.com> wrote:
>
>> On Jul 26, 1:59 pm, Paulo Penteado <pp.pente...@gmail.com> wrote:
>
>>> Do you mean that this does not happen every time? To me this seems to
>>> be the expected behavior. The line
>
>>> print,list(0)
>
>>> Is creating a list (which is an object), containing one element, and
>>> printing it. The same with the use of help.
>
>> OK, I did not know that list() was a new intrinsic function,
>> thanks. But since "list" is my most used variable name, V8.0 is not
>> backwards compatibile.
>
> Wow, through sheer luck, I've never used an array variable named

> LIST[]. I have mixed feelings about introducing such a potentially-
> incompatible change into IDL.
>
> Craig

Note that this doesn't seem specific to list -- in IDL 8, any object
(user written or otherwise) can now be created with the command
x = object_name(args)
as well as
x = obj_new('object_name', args)

chris

Subject: Re: Another small V8.0 bug
Posted by Juggernaut on Tue, 27 Jul 2010 11:47:03 GMT
View Forum Message <> Reply to Message

On Jul 26, 2:15 pm, wlandsman <wlands...@gmail.com> wrote:
> On Jul 26, 1:59 pm, Paulo Penteado <pp.pente...@gmail.com> wrote:
>
>> Do you mean that this does not happen every time? To me this seems to
>> be the expected behavior. The line
>
>> print,list(0)
>
>> Is creating a list (which is an object), containing one element, and
>> printing it. The same with the use of help.
>
> OK, I did not know that list() was a new intrinsic function,
> thanks. But since "list" is my most used variable name, V8.0 is not
> backwards compatibile.
>
> On the other hand, since I was a strong supporter of abandoning
> support for indexing with round parentheses, I shouldn't complain too
> much. But I suspect that this will be a recurring backwards
> compatibilty problem.
>
> --Wayne

I would say that it would be good programming practice to have more
specific variable names in functions. List is a very generic
variable. What is it a list of? I guess I'm saying something like
nameList or addressList or latitudeList is a more descriptive variable
than simply list. Saying that...I may very well have the same
variable name somewhere but a good find and replace will do the trick.

Subject: Re: Another small V8.0 bug
Posted by Paul Van Delst[1] on Tue, 27 Jul 2010 14:39:18 GMT
View Forum Message <> Reply to Message

Chris wrote:
> On Jul 26, 9:31 am, Craig Markwardt <craig.markwa...@gmail.com> wrote:
>> On Jul 26, 2:15 pm, wlandsman <wlands...@gmail.com> wrote:
>>
>>> On Jul 26, 1:59 pm, Paulo Penteado <pp.pente...@gmail.com> wrote:
>>>> Do you mean that this does not happen every time? To me this seems to
>>>> be the expected behavior. The line
>>>> print,list(0)
>>>> Is creating a list (which is an object), containing one element, and
>>>> printing it. The same with the use of help.
>>> OK, I did not know that list() was a new intrinsic function,
>>> thanks. But since "list" is my most used variable name, V8.0 is not
>>> backwards compatibile.
>> Wow, through sheer luck, I've never used an array variable named
>> LIST[]. I have mixed feelings about introducing such a potentially-
>> incompatible change into IDL.
>>
>> Craig
>
> Note that this doesn't seem specific to list -- in IDL 8, any object
> (user written or otherwise) can now be created with the command
> x = object_name(args)
> as well as
> x = obj_new('object_name', args)

Crikey. I hope the documentation clearly states how to turn that default behaviour OFF in one's idl_setup.pro file.

Because, you know, ITTVIS *did* make this behaviour user selectable, right? RIGHT?

:o)

I have arrays called "list" all over my code - most of which are arrays of objects (for my own homegrown pre-v8.0 linked
list). I strictly adhere to the [] convention for array indexing so I doubt this will affect me. I can't recall if I
have a function called "list" anywhere though....

I dislike the
  x = object_name(args)
alias for the regular
  x = obj_new('object_name', args)
because it now means you should include a comment in the code telling the future maintainers what is happening. That is,
rather than doing something like:

```
  x = obj_new('list',args)
```

you'd do

```
  ; Create a list object
  x = list(args)
```

I'm all for syntactic sugar, but this is more like aspartame (groan :o) -- it's obfuscating what was before, IMO, a
clear indication of what was happening. Now users will have to maintain (or, worse, debug) both the object creation
and the comment.

cheers,

paulv

Subject: Re: Another small V8.0 bug
Posted by penteado on Tue, 27 Jul 2010 15:33:21 GMT
View Forum Message <> Reply to Message

On Jul 27, 11:39 am, Paul van Delst <paul.vande...@noaa.gov> wrote:
> Crikey. I hope the documentation clearly states how to turn that default behaviour OFF in one's idl_setup.pro file.
>
> Because, you know, ITTVIS *did* make this behaviour user selectable, right? RIGHT?

As far as I know, this cannot be turned off.

> I dislike the
> x = object_name(args)
> alias for the regular
> x = obj_new('object_name', args)
> because it now means you should include a comment in the code telling the future maintainers what is happening. That is,
> rather than doing something like:
>
> x = obj_new('list',args)
>
> you'd do
>
> ; Create a list object
> x = list(args)
>
> I'm all for syntactic sugar, but this is more like aspartame (groan :o) -- it's obfuscating what was before, IMO, a

> clear indication of what was happening. Now users will have to maintain (or, worse, debug) both the object creation
> *and* the comment.

I find the new syntax better. obj_new() is too verbose. With its string class name it looks more like a dynamic evaluation function, such as execute(), more of a workaround for object use in the lack of object syntax. The class name notation makes it more clear that it is the creation of a variable of a certain type, and is similar to what is used in other languages. For that reason people occasionally already made a wrapper to make the object through a function call anyway (as in yesterday's thread by mankoff).

Most uses of the variable in the code will make it visible that it is an object. To me it seems that to comment that x=list(args) creates an object is just as necessary as saying that x=where(args) creates a 1D integer (maybe long) array, or that x=min(args) creates a scalar of the same type as args, or that x=dist(args) creates a 2D array of floats, or that x=dblarr(args) makes an array of doubles. None of these is saying much anyway, it is not saying what the variable will contain. If I just see an uncommented obj_new() of some class I am not familiar with, it tells me no more of what that variable will do than if I see only an uncommented assignment from a function I am not familiar with. The object can be anything, just as the function return value can be anything. In both cases I would have to either keep reading the code to see how it is used, or look up that function/class to find out what that variable will be.

I find that more informative and needed comments would actually say the purpose of that specific variable, which would not be conveyed by just the obj_new() call: things such as "create a list with file names that are referenced", or "list of targets with observations above the threshold", or "hash for the keywords set in the parameter file".

Those who find obj_new() more clear can keep using it, but I do not see any assured clarity improvement just from its use. Since IDL is dynamically typed, and function names are resolved at runtime, statically looking at code never does completely tell what will happen anyway.

Subject: Re: Another small V8.0 bug
Posted by Paul Van Delst[1] on Tue, 27 Jul 2010 18:33:02 GMT
View Forum Message <> Reply to Message

Paulo Penteado wrote:

> I find the new syntax better.

I started to reply point by point when I realised that there are just too many (totally subjective) factors involved in
how I came to my opinion. Re-reading your reply makes me think I've become a bit of a stick-in-the-mud when it comes to
IDL - the points you made are pretty much what I have used in the past arguing for colleagues to embrace change in other
languages.

Thanks for the shake-up that enabled this poster to move their perception-anchor! :o)

cheers,

paulv

Subject: Re: Another small V8.0 bug
Posted by JDS on Wed, 28 Jul 2010 21:53:47 GMT
View Forum Message <> Reply to Message

On Jul 26, 2:15 pm, wlandsman <wlands...@gmail.com> wrote:
> On Jul 26, 1:59 pm, Paulo Penteado <pp.pente...@gmail.com> wrote:
>
>> Do you mean that this does not happen every time? To me this seems to
>> be the expected behavior. The line
>
>> print,list(0)
>
>> Is creating a list (which is an object), containing one element, and
>> printing it. The same with the use of help.
>
> OK, I did not know that list() was a new intrinsic function,
> thanks. But since "list" is my most used variable name, V8.0 is not
> backwards compatibile.
>
> On the other hand, since I was a strong supporter of abandoning
> support for indexing with round parentheses, I shouldn't complain too
> much. But I suspect that this will be a recurring backwards
> compatibilty problem.

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.

JD