Subject: Happy Reasons to use IDL 8.1
Posted by David Fanning on Tue, 24 May 2011 15:24:58 GMT

View Forum Message <> Reply to Message

Folks,

I've discovered another happy reason to love IDL 8.1
this morning. I couldn't figure out why a main-level
program wouldn't compile. It didn't like this line:

```
   img = Image(image, ...)
```

Apparently, you can't define an image variable with
the name "image" and then use it as the argument
to the Image function.

I find it hard to believe. But just renaming my
image variable to "bob" solved the problem. :-(

Cheers,

David

P.S. Let's just say that something is going to have
to be done about IDL name space soon, or IDL is going
to become unusable. It may already be unusable for
many SolarSoft users. :-(

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.idlcoyote.com/
Sepore ma de ni thui. ("Perhaps thou speakest truth.")

Subject: Re: Happy Reasons to use IDL 8.1
Posted by David Fanning on Tue, 24 May 2011 20:14:10 GMT

View Forum Message <> Reply to Message

David Fanning writes:

> What the *hell* is going on here!? :-(

OK, here is the problem. Here is my pared down code:

```
image = dist(200)
w = window(dimensions=[400,400])
img = image(image, /current)
```

END

This code when compiled always throws a syntax error
on the image command.

```
 IDL> .compile -v 'C:\IDL\default\junk.pro'

   img = image(image, /current)
               ^
   % Syntax error.
     At: C:\IDL\default\junk.pro, Line 3
   % 1 Compilation error(s) in module $MAIN$.
```

What is *that* about!? Remove the CURRENT keyword
and the program compiles, although the image never
shows up anywhere!

Sheesh. Are you sure it's not Friday!?

Cheers,

David
--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.idlcoyote.com/
Sepore ma de ni thui. ("Perhaps thou speakest truth.")

Subject: Re: Happy Reasons to use IDL 8.1
Posted by Foldy Lajos on Tue, 24 May 2011 20:21:09 GMT
View Forum Message <> Reply to Message

On Tue, 24 May 2011, David Fanning wrote:

```
>  David Fanning writes:
>
>>  What the *hell* is going on here!? :-(
>
>  OK, here is the problem. Here is my pared down code:
>
>  image = dist(200)
>  w = window(dimensions=[400,400])
>  img = image(image, /current)
>  END
>
>  This code when compiled always throws a syntax error
>  on the image command.
```

```
>
>  IDL> .compile -v 'C:\IDL\default\junk.pro'
>
>    img = image(image, /current)
>               ^
>    % Syntax error.
>      At: C:\IDL\default\junk.pro, Line 3
>    % 1 Compilation error(s) in module $MAIN$.
>
>  What is *that* about!? Remove the CURRENT keyword
>  and the program compiles, although the image never
>  shows up anywhere!
>
```

Probably IDL does not know about your image function and tries to
subscript the image variable. Does compile_opt idl2 help?

regards,
Lajos

Subject: Re: Happy Reasons to use IDL 8.1
Posted by David Fanning on Tue, 24 May 2011 21:28:19 GMT
View Forum Message <> Reply to Message

FÖLDY Lajos writes:

> Probably IDL does not know about your image function and tries to
> subscript the image variable. Does compile_opt idl2 help?

Well, this occurs in a fresh IDL session, so image
is undefined. I've also checked to be sure no procedures
or functions named "image" are defined.

```
IDL> .reset
IDL> .compile -v 'C:\IDL\default\junk.pro'

img = image(image, /current)
           ^
% Syntax error.
  At: C:\IDL\default\junk.pro, Line 4
% 1 Compilation error(s) in module $MAIN$.
```

Yes, Compile_Opt idl2 helps and seems to solve
the problem. Yet one more reason (if we need one!)
that this should finally be the default behavior!

The fact that I know how to work-around the problem

(naming my image "bob" also solves the problem!) doesn't
give me much satisfaction. What do you suppose the
problem is?

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.idlcoyote.com/
Sepore ma de ni thui. ("Perhaps thou speakest truth.")

Subject: Re: Happy Reasons to use IDL 8.1
Posted by ben.bighair on Wed, 25 May 2011 01:22:35 GMT

View Forum Message <> Reply to Message

On 5/24/11 5:28 PM, David Fanning wrote:

> FÏ¿œLDY Lajos writes:
>
>> Probably IDL does not know about your image function and tries to
>> subscript the image variable. Does compile_opt idl2 help?
>
> Well, this occurs in a fresh IDL session, so image
> is undefined. I've also checked to be sure no procedures
> or functions named "image" are defined.
>

Hi David,

I thought IDL 8+ comes with a function named "Image()" - could it be
that the compiler is thinking that you calling the function with a
function as an argument.

Ben

```
>  IDL>  .reset
>  IDL>  .compile -v 'C:\IDL\default\junk.pro'
>
>  img = image(image, /current)
>             ^
>  % Syntax error.
>    At: C:\IDL\default\junk.pro, Line 4
>  % 1 Compilation error(s) in module $MAIN$.
>
```

> Yes, Compile_Opt idl2 helps and seems to solve
> the problem. Yet one more reason (if we need one!)
> that this should finally be the default behavior!
>
> The fact that I know how to work-around the problem
> (naming my image "bob" also solves the problem!) doesn't
> give me much satisfaction. What do you suppose the
> problem is?
>
> Cheers,
>
> David
>
>

Subject: Re: Happy Reasons to use IDL 8.1
Posted by Foldy Lajos on Wed, 25 May 2011 08:42:01 GMT
View Forum Message <> Reply to Message

On Tue, 24 May 2011, David Fanning wrote:

> FÖLDY Lajos writes:
>
>> Probably IDL does not know about your image function and tries to
>> subscript the image variable. Does compile_opt idl2 help?
>
> Well, this occurs in a fresh IDL session, so image
> is undefined. I've also checked to be sure no procedures
> or functions named "image" are defined.
>
> IDL> .reset
> IDL> .compile -v 'C:\IDL\default\junk.pro'
>
> img = image(image, /current)
>   ^
> % Syntax error.
> At: C:\IDL\default\junk.pro, Line 4
> % 1 Compilation error(s) in module $MAIN$.
>
> Yes, Compile_Opt idl2 helps and seems to solve
> the problem. Yet one more reason (if we need one!)
> that this should finally be the default behavior!
>
> The fact that I know how to work-around the problem
> (naming my image "bob" also solves the problem!) doesn't
> give me much satisfaction. What do you suppose the
> problem is?

>

The problem is that image() is ambiguous, it can be a function call or a
variable subscript. IDL must choose after reading the tokens 'image' '('.
I guess the rules are:

- if the image function is defined, it is a function call

- if the image variable is defined, it is a subscript

- if the image function can be resolved using !PATH, it is a function call

- it is a subscript

(here defined means defined in the internal tables of the compiler)

I think we have to live with it. "bob" or compile_opt idl2 (strictarray)
removes the ambiguity. Probably IDL should add an -idl2 command line
switch to make it default globally.

regards,
Lajos

Subject: Re: Happy Reasons to use IDL 8.1
Posted by ameigs on Wed, 25 May 2011 09:58:38 GMT

View Forum Message <> Reply to Message

```
On May 25, 9:42 am, FÖLDY Lajos <fo...@rmki.kfki.hu> wrote:
>  On Tue, 24 May 2011, David Fanning wrote:
>>  FÖLDY Lajos writes:
>
>>>  Probably IDL does not know about your image function and tries to
>>>  subscript the image variable. Does compile_opt idl2 help?
>
>>  Well, this occurs in a fresh IDL session, so image
>>  is undefined. I've also checked to be sure no procedures
>>  or functions named "image" are defined.
>
>>  IDL> .reset
>>  IDL> .compile -v 'C:\IDL\default\junk.pro'
>
>>  img = image(image, /current)
>>                  ^
>>  % Syntax error.
>>   At: C:\IDL\default\junk.pro, Line 4
>>  % 1 Compilation error(s) in module $MAIN$.
>
```

>> Yes, Compile_Opt idl2 helps and seems to solve
>> the problem. Yet one more reason (if we need one!)
>> that this should finally be the default behavior!
>
>> The fact that I know how to work-around the problem
>> (naming my image "bob" also solves the problem!) doesn't
>> give me much satisfaction. What do you suppose the
>> problem is?
>
> The problem is that image() is ambiguous, it can be a function call or a
> variable subscript. IDL must choose after reading the tokens 'image' '('.
> I guess the rules are:
>
> - if the image function is defined, it is a function call
>
> - if the image variable is defined, it is a subscript
>
> - if the image function can be resolved using !PATH, it is a function call
>
> - it is a subscript
>
> (here defined means defined in the internal tables of the compiler)
>
> I think we have to live with it. "bob" or compile_opt idl2 (strictarray)
> removes the ambiguity. Probably IDL should add an -idl2 command line
> switch to make it default globally.
>
> regards,
> Lajos

Know you're trying to help, but ever heard of the "teaching an old dog
to suck eggs". David is finding unexpected problems and well he is one
of 5 experts that I know of in IDL. Sorry, I can't test this
behaviour; my lab is stuck currently at IDL 7.1.1....

Andy

Subject: Re: Happy Reasons to use IDL 8.1
Posted by Foldy Lajos on Wed, 25 May 2011 11:52:43 GMT

View Forum Message <> Reply to Message

On Wed, 25 May 2011, ameigs wrote:

> Know you're trying to help, but ever heard of the "teaching an old dog
> to suck eggs". David is finding unexpected problems and well he is one
> of 5 experts that I know of in IDL. Sorry, I can't test this
> behaviour; my lab is stuck currently at IDL 7.1.1....

It is not an IDL 8.1 feature. I think it was present in IDL 1.0, too.

regards,
Lajos

Subject: Re: Happy Reasons to use IDL 8.1
Posted by David Fanning on Wed, 25 May 2011 12:32:56 GMT

View Forum Message <> Reply to Message

FÖLDY Lajos writes:

> It is not an IDL 8.1 feature. I think it was present in IDL 1.0, too.

Yes, I'm sure it was, too. And maybe we have just
learned over the years to not create variables
named, say "bytscl", but I have a feeling there
are a lot more people than just me who have
variables named "image", and when that causes
your programs to not work in what appear to be
completely arbitrary ways (the program ran
10 minutes ago!!), then I think it is going
to cause deep consternation (not to say
animosity) toward the software.

I'm just saying this is not the first time
these new graphics function names have
given me a LOT of trouble. :-(

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.idlcoyote.com/
Sepore ma de ni thui. ("Perhaps thou speakest truth.")

Subject: Re: Happy Reasons to use IDL 8.1
Posted by pgrigis on Wed, 25 May 2011 14:32:06 GMT

View Forum Message <> Reply to Message

On May 25, 8:32 am, David Fanning <n...@idlcoyote.com> wrote:
> FÖLDY Lajos writes:
>> It is not an IDL 8.1 feature. I think it was present in IDL 1.0, too.
>
> Yes, I'm sure it was, too. And maybe we have just
> learned over the years to not create variables
> named, say "bytscl", but I have a feeling there
> are a lot more people than just me who have
> variables named "image", and when that causes
> your programs to not work in what appear to be
> completely arbitrary ways (the program ran
> 10 minutes ago!!), then I think it is going
> to cause deep consternation (not to say
> animosity) toward the software.

Right - this is exactly the same reason I had troubles
with a function called "limits" in solarsoft that conflicted
with similarly named variables.

When I teach a student to use IDL the first thing I tell
them is to chose 2 or 3 letter initials to represent themselves
and write *all* their programs using the naming convention
xyz_****.pro

I think it's time for ITTVIS to follow this rule by reserving
a few initial letter clusters, and use that for any future new
function, or finally bite the bullet and enforce array indexing
with square brackets - either way we can't have any more name space
pollution.

Ciao,
Paolo

>
> I'm just saying this is not the first time
> these new graphics function names have
> given me a LOT of trouble. :-(
>
> Cheers,
>
> David
>
> --
> David Fanning, Ph.D.
> Fanning Software Consulting, Inc.

> Coyote's Guide to IDL Programming:http://www.idlcoyote.com/
> Sepore ma de ni thui. ("Perhaps thou speakest truth.")

Subject: Re: Happy Reasons to use IDL 8.1
Posted by pgrigis on Wed, 25 May 2011 14:34:18 GMT

View Forum Message <> Reply to Message

On May 25, 5:58 am, ameigs <andyme...@gmail.com> wrote:
> On May 25, 9:42 am, FÖLDY Lajos <fo...@rmki.kfki.hu> wrote:
>
>
>
>
>
>
>
>
>
>> On Tue, 24 May 2011, David Fanning wrote:
>>> FÖLDY Lajos writes:
>
>>>> Probably IDL does not know about your image function and tries to
>>>> subscript the image variable. Does compile_opt idl2 help?
>
>>> Well, this occurs in a fresh IDL session, so image
>>> is undefined. I've also checked to be sure no procedures
>>> or functions named "image" are defined.
>
>>> IDL> .reset
>>> IDL> .compile -v 'C:\IDL\default\junk.pro'
>
>>> img = image(image, /current)
>>>   ^
>>> % Syntax error.
>>> At: C:\IDL\default\junk.pro, Line 4
>>> % 1 Compilation error(s) in module $MAIN$.
>
>>> Yes, Compile_Opt idl2 helps and seems to solve
>>> the problem. Yet one more reason (if we need one!)
>>> that this should finally be the default behavior!
>
>>> The fact that I know how to work-around the problem
>>> (naming my image "bob" also solves the problem!) doesn't
>>> give me much satisfaction. What do you suppose the
>>> problem is?
>
>> The problem is that image() is ambiguous, it can be a function call or a

>> variable subscript. IDL must choose after reading the tokens 'image' '('.
>> I guess the rules are:
>
>> - if the image function is defined, it is a function call
>
>> - if the image variable is defined, it is a subscript
>
>> - if the image function can be resolved using !PATH, it is a function call
>
>> - it is a subscript
>
>> (here defined means defined in the internal tables of the compiler)
>
>> I think we have to live with it. "bob" or compile_opt idl2 (strictarray)
>> removes the ambiguity. Probably IDL should add an -idl2 command line
>> switch to make it default globally.
>
>> regards,
>> Lajos
>
> Know you're trying to help, but ever heard of the "teaching an old dog
> to suck eggs". David is finding unexpected problems and well he is one
> of 5 experts that I know of in IDL. Sorry, I can't test this
> behaviour; my lab is stuck currently at IDL 7.1.1....
>
> Andy

Andy, Lajos wrote an IDL clone from scratch and therefore is
probably more experienced then anybody else about rules regarding
how IDL statements are compiled (or how they *should* be....)

Ciao,
Paolo

Subject: Re: Happy Reasons to use IDL 8.1
Posted by David Fanning on Wed, 25 May 2011 14:58:01 GMT

View Forum Message <> Reply to Message

Paolo writes:

> Andy, Lajos wrote an IDL clone from scratch and therefore is
> probably more experienced then anybody else about rules regarding
> how IDL statements are compiled (or how they *should* be....)

And, in a newsgroup, you are writing for everyone, not
just the person who asked the question. Although
I suppose I am an "expert", I often ask questions

(and answer them!) as if I didn't know the first thing
about IDL, mostly to help the folks who lurk here. :-)

I appreciate Lajos's answer.

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.idlcoyote.com/
Sepore ma de ni thui. ("Perhaps thou speakest truth.")