Subject: Re: Warning: IDL 8.0 alters the behaviour of existing valid programs without any notice!
Posted by Maarten[1] on Wed, 18 Aug 2010 12:14:15 GMT
View Forum Message <> Reply to Message

On Aug 18, 1:59 pm, svhhaugan <s.v.h.hau...@gmail.com> wrote:
> pro test,data
> catch,error
> if error ne 0 then begin
> catch,/cancel
> print,"Guess that didn't work"
> return
> end
> data[where(data eq min(data)-1)] = 50
> end
>
> Perfectly valid code, makes sense in IDL 7.1 although the catch,error
> part would most likely be done "with human intervention".

I just looked up the help for where in IDL 7. This is what is prominently displayed on that page:

Note: When WHERE Returns –1

If all the elements of Array_Expression are zero, WHERE returns a scalar integer with a value of –1. Attempting to use this result as an index into another array results in a "subscripts out of bounds" error. In situations where this is possible, code similar to the following can be used to avoid errors:

```
; Use Count to get the number of nonzero elements:
index = WHERE(array, count)

; Only subscript the array if it's safe:
IF count NE 0 THEN result = array[index]
```

And yes, you can use the try/catch exception method of handling this, but I would pose that testing for an item not found message and then using an IF statement to handle that common situation makes more sense than relying on an error. So in that sense the code was never valid or sensible in the first place.

Does that mean no one will be caught off-guard by this? No. Is that an issue? Some people just get what they deserve. The added benefit of being able to count from the end of an array is certainly worth some trouble.

Maarten

Subject: Re: Warning: IDL 8.0 alters the behaviour of existing valid programs without any notice!
Posted by Haje Korth on Wed, 18 Aug 2010 12:30:53 GMT
View Forum Message <> Reply to Message

I agree with Maarten's assessment. Negative indexing is a new feature
of IDL 8, which is documented and can be of great benefit. If the
WHERE statement would have been used as explained by Maarten and as
documented in the IDL help, the adverse effect in the sample code
given would have been avoided. Haje

On Aug 18, 8:14 am, Maarten <maarten.sn...@knmi.nl> wrote:
> On Aug 18, 1:59 pm, svhhaugan <s.v.h.hau...@gmail.com> wrote:
>
>> pro test,data
>> catch,error
>> if error ne 0 then begin
>> catch,/cancel
>> print,"Guess that didn't work"
>> return
>> end
>> data[where(data eq min(data)-1)] = 50
>> end
>
>> Perfectly valid code, makes sense in IDL 7.1 although the catch,error
>> part would most likely be done "with human intervention".
>
> I just looked up the help for where in IDL 7. This is what is
> prominently displayed on that page:
>
> Note: When WHERE Returns –1
>
> If all the elements of Array_Expression are zero, WHERE returns a
> scalar integer with a value of –1. Attempting to use this result as an
> index into another array results in a "subscripts out of bounds"
> error. In situations where this is possible, code similar to the
> following can be used to avoid errors:
>
> ; Use Count to get the number of nonzero elements:
> index = WHERE(array, count)
>
> ; Only subscript the array if it's safe:
> IF count NE 0 THEN result = array[index]
>
> And yes, you can use the try/catch exception method of handling this,
> but I would pose that testing for an item not found message and then
> using an IF statement to handle that common situation makes more sense
> than relying on an error. So in that sense the code was never valid or
> sensible in the first place.

>
> Does that mean no one will be caught off-guard by this? No. Is that an
> issue? Some people just get what they deserve. The added benefit of
> being able to count from the end of an array is certainly worth some
> trouble.
>
> Maarten

Subject: Re: Warning: IDL 8.0 alters the behaviour of existing valid programs without any notice!
Posted by svhhaugan on Wed, 18 Aug 2010 13:30:55 GMT
View Forum Message <> Reply to Message

On Aug 18, 2:14 pm, Maarten <maarten.sn...@knmi.nl> wrote:
> On Aug 18, 1:59 pm, svhhaugan <s.v.h.hau...@gmail.com> wrote:
>
>> pro test,data
>> catch,error
>> if error ne 0 then begin
>> catch,/cancel
>> print,"Guess that didn't work"
>> return
>> end
>> data[where(data eq min(data)-1)] = 50
>> end
>
>> Perfectly valid code, makes sense in IDL 7.1 although the catch,error
>> part would most likely be done "with human intervention".
>
> I just looked up the help for where in IDL 7. This is what is
> prominently displayed on that page:
>
> Note: When WHERE Returns –1
>
> If all the elements of Array_Expression are zero, WHERE returns a
> scalar integer with a value of –1. Attempting to use this result as an
> index into another array results in a "subscripts out of bounds"
> error. In situations where this is possible, code similar to the
> following can be used to avoid errors:
>
> ; Use Count to get the number of nonzero elements:
> index = WHERE(array, count)
>
> ; Only subscript the array if it's safe:
> IF count NE 0 THEN result = array[index]
>
> And yes, you can use the try/catch exception method of handling this,

> but I would pose that testing for an item not found message and then
> using an IF statement to handle that common situation makes more sense
> than relying on an error. So in that sense the code was never valid or
> sensible in the first place.

As written, it is just for "demonstrational purposes". But I bet a
lot of code out there is written to *break* when tested
(interactively) with parameters that just won't work. This code
will no longer break, it will silently do something different
from what was originally intended.

It was a documented feature ("prominently displayed") of IDL 7
that such usage would cause a crash or trigger a catch statement
("Attempting to use this result as an index ...
results in [an] error").

For IDL 8, the rules have suddenly been changed, with no warning
and no way to turn it off (other than staying with version 7).

> Does that mean no one will be caught off-guard by this? No. Is that an
> issue? Some people just get what they deserve. The added benefit of
> being able to count from the end of an array is certainly worth some
> trouble.

I agree that counting from the end of the array is a Good Thing. But
I think having to insert a "compile_opt strictnegarrsubs" (or
something of that sort) in every existing routine written for IDL 7
would be enough trouble. Right now, not even that is possible.

And although it's fine to say "some people just get what they
deserve",
the problem is that these people won't necessarily notice that
anything
is wrong, and others may rely on the false results.

Subject: Re: Warning: IDL 8.0 alters the behaviour of existing valid programs without any notice!
Posted by jeffnettles4870 on Wed, 18 Aug 2010 18:45:15 GMT
View Forum Message <> Reply to Message

>
> For IDL 8, the rules have suddenly been changed, with no warning
> and no way to turn it off (other than staying with version 7).
>

I don't know how you can say there was no warning - ITTVIS has been
advertising the negative indexing feature for months now. And it was

completely obvious to me from the get-go that the new indexing would
have the effect on where() calls that you're talking about. It was
one of the first things I thought of.

> And although it's fine to say "some people just get what they
> deserve",
> the problem is that these people won't necessarily notice that
> anything
> is wrong, and others may rely on the false results.

I suspect that this is a danger with many, if not most, programs that
are shared with others. That's a testament to the need for due
diligence on the part of programmers in my book.

Subject: Re: Warning: IDL 8.0 alters the behaviour of existing valid programs without any notice!
Posted by svhhaugan on Thu, 19 Aug 2010 11:57:19 GMT
View Forum Message <> Reply to Message

On Aug 18, 8:45 pm, "Jeff N." <jeffnettles4...@gmail.com> wrote:
>> For IDL 8, the rules have suddenly been changed, with no warning
>> and no way to turn it off (other than staying with version 7).
>
> I don't know how you can say there was no warning - ITTVIS has been
> advertising the negative indexing feature for months now. And it was
> completely obvious to me from the get-go that the new indexing would
> have the effect on where() calls that you're talking about. It was
> one of the first things I thought of.

I guess we haven't been spending much time reading the advertising ;-)
Anyhow, that's not the kind of warning I referred to.

I think most researchers naively expect new features in a
"programming
language" to preserve the stated semantics of existing code, or at
least
provide (optional) warning mechanisms (compile-time or run-time)
that'll flag those cases where the semantics have been changed.

The help file for IDL 7 said "code similar to the following *can*
be used to avoid errors", not "code similar to the following
should be used, because we do not guarantee that an error will
occur in the next version of IDL".

Stein

Subject: Re: Warning: IDL 8.0 alters the behaviour of existing valid programs without any notice!
Posted by Chris[7] on Thu, 19 Aug 2010 19:35:48 GMT
View Forum Message <> Reply to Message

In reality, how many people actually use CATCH to handle the case of a failed call to where? Maybe I'm off base here, but I'm guessing that people either do something like

hit = where(data eq min(data)-1, count)
if count eq 0 then ....

or they don't check for a failure at all. In the former case, the code still works fine in IDL 8. In the latter case, the data gets messed up in an admittedly subtle way. But in that case, the "existing valid code" isn't valid -- not checking the result of where is a bug. It wouldn't be fair to blame IDL for that.

chris

On 8/19/10 1:57 AM, svhhaugan wrote:
> On Aug 18, 8:45 pm, "Jeff N."<jeffnettles4...@gmail.com> wrote:
>>> For IDL 8, the rules have suddenly been changed, with no warning
>>> and no way to turn it off (other than staying with version 7).
>>
>> I don't know how you can say there was no warning - ITTVIS has been
>> advertising the negative indexing feature for months now. And it was
>> completely obvious to me from the get-go that the new indexing would
>> have the effect on where() calls that you're talking about. It was
>> one of the first things I thought of.
>
> I guess we haven't been spending much time reading the advertising ;-)
> Anyhow, that's not the kind of warning I referred to.
>
> I think most researchers naively expect new features in a
> "programming
> language" to preserve the stated semantics of existing code, or at
> least
> provide (optional) warning mechanisms (compile-time or run-time)
> that'll flag those cases where the semantics have been changed.
>
> The help file for IDL 7 said "code similar to the following *can*
> be used to avoid errors", not "code similar to the following
> *should* be used, because we do not guarantee that an error will
> occur in the next version of IDL".
>
> Stein