Subject: Troubleshooting - Error Messages.
Posted by Vicky A on Wed, 01 Dec 1999 08:00:00 GMT
View Forum Message <> Reply to Message

Hello,

I'm hoping to benefit from the experience of all you wise people out
there...

I often find that I make small mistakes when typing in code to IDL
eg: I forget to put a comma after 'print',
I forget that if I have two 'for' statements I need TWO 'endfor'
statements,
I forget to put continuation and concatenation symbols in batch files.

These are pretty serious errors in that IDL has no idea what I mean,
but pretty simple errors in that I only need to add another comma (or
whatever) for everything to run like a dream (in theory...!)

I find the Error Messages provided by IDL unhelpful
('% Syntax Error.' tells me very little about what's up).

Is there some standard procedure for dealing with compilation errors?
(I mean errors that prevent a file compiling)

I know an experienced user can just look at
print n
and say/think 'There should be a comma there.'

Is that what it boils down to, IDL tells you Where the error is, and
you work out What the error is, by recognition or something?

I find it really frustrating that I have to keep going back to the
book/webpages for every little typying mistake....

Oh, and I'm using the Student Vresion if that changes anything.

Thank you very much,

Vicky A.

Sent via Deja.com http://www.deja.com/
Before you buy.

Subject: Re: Troubleshooting - Error Messages.

Posted by robert.mallozzi on Sun, 05 Dec 1999 08:00:00 GMT
View Forum Message <> Reply to Message

In article <384A045B.FF23F80A@pop.omah.uswest.net>,
"David L. Keller" <davekeller@pop.omah.uswest.net> writes:
>
> IDL: The on-line help is In My Opinion the best way (so far)
> to look up syntax. It is quick and only a few button-clicks
> away (at least on my Linux box). To keep it available, I
> start another IDL session in another 'screen', and open up
> the IDL hypertext help. I keep the session and the help
> window open all day. I don't have to wait for the help
> window to be created, nor the help file contents to be read
> and formatted every time I have a simple syntax question.

Hi,

You don't really need to start another IDL session to use
the on-line help - try the command "idlhelp &" from the Unix
prompt.

Regards,

-bob

--
~~~~~~ ~~~~~~~~~
Robert S. Mallozzi   256-544-0887
                                                Mail Code SD 50
http://gammaray.msfc.nasa.gov/   Marshall Space Flight Center
http://cspar.uah.edu/~mallozzir/   Huntsville, AL 35812
~~~~~~ ~~~~~~~~~

Subject: Re: Troubleshooting - Error Messages.
Posted by David L. Keller on Sun, 05 Dec 1999 08:00:00 GMT
View Forum Message <> Reply to Message

Vicky A wrote:

> Hello,
>
> I'm hoping to benefit from the experience of all you wise people out
> there...
>

> I often find that I make small mistakes when typing in code to IDL
> eg: I forget to put a comma after 'print',
> I forget that if I have two 'for' statements I need TWO 'endfor'
> statements,
> I forget to put continuation and concatenation symbols in batch files.
>
> These are pretty serious errors in that IDL has no idea what I mean,
> but pretty simple errors in that I only need to add another comma (or
> whatever) for everything to run like a dream (in theory...!)
>
> I find the Error Messages provided by IDL unhelpful
> ('% Syntax Error.' tells me very little about what's up).
>
> Is there some standard procedure for dealing with compilation errors?
> (I mean errors that prevent a file compiling)
>
> I know an experienced user can just look at
> print n
> and say/think 'There should be a comma there.'
>
> Is that what it boils down to, IDL tells you Where the error is, and
> you work out What the error is, by recognition or something?
>
> I find it really frustrating that I have to keep going back to the
> book/webpages for every little typying mistake....
>
> Oh, and I'm using the Student Vresion if that changes anything.
>
> Thank you very much,
>
> Vicky A.
>
> Sent via Deja.com http://www.deja.com/
> Before you buy.

I will volunteer to be a 'wise person'...

There ARE a few languages that immediately tell you where
and what the syntax error is. Some BASICs do this. Some
FORTRANs are better than others in this regard.

IDL: The on-line help is In My Opinion the best way (so far)
to look up syntax. It is quick and only a few button-clicks
away (at least on my Linux box). To keep it available, I
start another IDL session in another 'screen', and open up
the IDL hypertext help. I keep the session and the help
window open all day. I don't have to wait for the help
window to be created, nor the help file contents to be read

and formatted every time I have a simple syntax question.

Otherwise, regarding syntax, keep notes in a file:
  c:\reference\idl\idlnotes.txt
Jot down common syntactical items for your reference:
  o IF (then else endif endelse) syntax
  o FOR syntax
  o OPEN syntax
  o Snippets of code that you run in a 'test' mode,
     such as I describe below.
  o Etc.
This might be the quickest available reference for syntax
you just can't remember.

I used to have some 'screen-sized' (20 lines) examples of
common tasks: map plotting, x-y plots, reading data,
contouring, etc. I could look for the hard copy and post
them if somebody begs me nicely. These would have some
syntax examples to start from.

Otherwise start with working examples. Start from the
RSI-supplied demo code. Look in the IDL libraries for
working code. Look for people's libraries on the
internet; several outstanding examples exist.

For some of your errors, try compiling with the Listing
option:

```
    IDL> .run -L progname.lst progname.pro
```

Look for the file 'progname.lst'. This will have
indentation. This will help you see what IDL thinks
your FOR and WHILE and IF structures are.

Use the best possible programming editor you can beg,
borrow, or steal. Use it to search for valid syntax
examples in your own code.

Key in a statement(s) interactively

```
  print
  print a
       ^
  % Syntax error.
```

This helps a bit...you might suspect the space from these
experiments.

Some more esoteric but potent methods:

Take out the problem lines, put into a separate file.
Not a technique that directly points to syntax, but an
effective and underutilized method nevertheless.
I saw this bug within a program once:

```
  IF a lt 7 THEN $

  a=7
  PRINT,"The value of a is : ",a
```

Note that at the time several years ago, the BLANK LINE
caused the continuation to continue to NOTHING. Typing
this in a SEPARATE LITTLE PROGRAM, complete with an END
statement, allowed me to 'see' the blank line. (Remarkably,
this program worked correctly for me tonight, but that's
not the issue here!).

This re-typing of snippets of code would help you with
your FOR, ENDFOR problem. The reason is that the problem
is isolated to be within the lines that you type into this
little test program. Less program to look through.

Another esoteric technique is to comment out large segments
of your program. Add chunks back into the program until
it behaves badly. Your problem is in the last chunk you
added. I have found that this does not take nearly as long
as I would think, and really works.

Alternately you can start with a fresh program file. Cut
and paste from the 'bad' program into the 'new' program
until the program fails. Again, the last chunk of code you
pasted has the trouble. With your 'FOR / ENDFOR' problem:

```
  FOR i=1,100 DO BEGIN
    FOR j=1,100 DO BEGIN
      ;COMMENT OUT EVERYTHING BELOW
      ;(USE AN EDITOR WITH A KEYSTROKE RECORDING CAPABILITY
;       print,i,j,'boo'
;         ;
;         ;
```

Oops, no ENDFOR.

Hope some of this helps, or at least amuses you. I still find myself
frustrated
remembering syntax from programming language to programming language.

-- Dave --

Subject: Re: Troubleshooting - Error Messages.
Posted by J.D. Smith on Wed, 08 Dec 1999 08:00:00 GMT
View Forum Message <> Reply to Message

"J.D. Smith" wrote:
>
> "David L. Keller" wrote:
>>
>> Vicky A wrote:
>>
>>> Hello,
>>>
>>> I'm hoping to benefit from the experience of all you wise people out
>>> there...
>>>
>>> I often find that I make small mistakes when typing in code to IDL
>>> eg: I forget to put a comma after 'print',
>>> I forget that if I have two 'for' statements I need TWO 'endfor'
>>> statements,
>>> I forget to put continuation and concatenation symbols in batch files.
>>>
>>> These are pretty serious errors in that IDL has no idea what I mean,
>>> but pretty simple errors in that I only need to add another comma (or
>>> whatever) for everything to run like a dream (in theory...!)
>>>
>>> I find the Error Messages provided by IDL unhelpful
>>> ('% Syntax Error.' tells me very little about what's up).
>>>
>>> Is there some standard procedure for dealing with compilation errors?
>>> (I mean errors that prevent a file compiling)
>>>
>>> I know an experienced user can just look at
>>> print n
>>> and say/think 'There should be a comma there.'
>>>
>>> Is that what it boils down to, IDL tells you Where the error is, and
>>> you work out What the error is, by recognition or something?
>>>
>>> I find it really frustrating that I have to keep going back to the
>>> book/webpages for every little typing mistake....
>>>
>>> Oh, and I'm using the Student Vresion if that changes anything.
>>>

>>> Thank you very much,
>>>
>>> Vicky A.
>>>
>>> Sent via Deja.com http://www.deja.com/
>>> Before you buy.
>>
>> I will volunteer to be a 'wise person'...
>>
>> There ARE a few languages that immediately tell you where
>> and what the syntax error is. Some BASICs do this. Some
>> FORTRANs are better than others in this regard.
>>
>> IDL: The on-line help is In My Opinion the best way (so far)
>> to look up syntax. It is quick and only a few button-clicks
>> away (at least on my Linux box). To keep it available, I
>> start another IDL session in another 'screen', and open up
>> the IDL hypertext help. I keep the session and the help
>> window open all day. I don't have to wait for the help
>> window to be created, nor the help file contents to be read
>> and formatted every time I have a simple syntax question.
>>
>
> You should look into the new idlwave emacs mode, if emacs is your thing. It
> really changes the way I program IDL. It has built-in abbreviations for common
> structures like IF blocks, PRO, FUNCTION, CASE, WHILE etc. blocks, among many
> other useful ones. It notices if you use the wrong type of END (ENDIF in a FOR
> loop, for example). And, thanks to Carsten Dominik, now allows you to do
> completion and syntax queries on the function or procedure... you can even
> compile your own routines into a library for searching.
>
> An example with, e.g. [M-Tab], representing where I hit meta-tab or other key
> combo, and another buffer represented by *'s:
>
> a=cu[M-Tab]
>
> *Click mouse-2 on a completion to select it.
> *In this buffer, type RET to select the completion near point.
> *
> *Possible completions are:
> *cumulate   curvefit
>
> or if I have given it more:
>
> a=cur[M-Tab]
>
> becomes
>

> a=curvefit([M-Tab]
>
> which gives me
>
> *Click mouse-2 on a completion to select it.
> *In this buffer, type RET to select the completion near point.
> *
> *Possible completions are:
> *CHISQ   FUNCTION_NAME
> *ITER   ITMAX
> *NODERIVATIVE   TOL
>
> to see which keywords I can use. Clicking on one inserts it... e.g. clicking
> NODERIVATIVE yields.
>
> a=curvefit(NODERIVATIVE=[C-c-?]
>
> (that's a Control-c-?) gives me the routine info:
>
> *Usage: Result = CURVEFIT(X, Y, Weights, A [, Sigma])
> *Keywords: CHISQ FUNCTION_NAME ITER ITMAX NODERIVATIVE TOL
> *Origin: system routine
>
> Origin could be a library file (compiled from all your favorite routines),
> scanned buffers which are open in Emacs, or routines compiled into the IDLWAVE
> Shell (idl running as a subprocess to Emacs). Definitely faster than the online
> help when you know a routine but have just forgotten whether a keyword is like
> "NOZERO" or "NO_COPY" (my favorite dichotomy), or which argument goes first.
>
> Mixed case or forced case completion (including object method completion) is
> available, and the whole thing is very configureable. Take a look at
> http://www.strw.leidenuniv.nl/~dominik/Tools/idlwave/ to find out more. Oh and
> did I mention it indents, highlights, colors, and otherwise arranges your code
> according to your chosen preferences? Another one of my favorites... with
> cursor on or in a routine (procedure or function), you hit C-c-v to pull up the
> source code to that routine! Debugging support is also built-in with the Shell,
> and it automatically pulls up files

Sorry this message flew the coop before I could get a chance to finish it.

I was saying:

Debugging support is also built-in with the Shell, and it automatically pulls up
files for the modules in which the error has occurred. Setting, visualizing,
and removing breakpoints is simple. And there are too many other features to
list here. Check it out!

Good Luck,

JD

--
J.D. Smith |*| WORK: (607) 255-5842
Cornell University Dept. of Astronomy |*| (607) 255-6263
304 Space Sciences Bldg. |*| FAX: (607) 255-5875
Ithaca, NY 14853 |*|

Subject: Re: Troubleshooting - Error Messages.
Posted by J.D. Smith on Wed, 08 Dec 1999 08:00:00 GMT
View Forum Message <> Reply to Message

"David L. Keller" wrote:
>
> Vicky A wrote:
>
>> Hello,
>>
>> I'm hoping to benefit from the experience of all you wise people out
>> there...
>>
>> I often find that I make small mistakes when typing in code to IDL
>> eg: I forget to put a comma after 'print',
>> I forget that if I have two 'for' statements I need TWO 'endfor'
>> statements,
>> I forget to put continuation and concatenation symbols in batch files.
>>
>> These are pretty serious errors in that IDL has no idea what I mean,
>> but pretty simple errors in that I only need to add another comma (or
>> whatever) for everything to run like a dream (in theory...!)
>>
>> I find the Error Messages provided by IDL unhelpful
>> ('% Syntax Error.' tells me very little about what's up).
>>
>> Is there some standard procedure for dealing with compilation errors?
>> (I mean errors that prevent a file compiling)
>>
>> I know an experienced user can just look at
>> print n
>> and say/think 'There should be a comma there.'
>>
>> Is that what it boils down to, IDL tells you Where the error is, and
>> you work out What the error is, by recognition or something?
>>
>> I find it really frustrating that I have to keep going back to the

>> book/webpages for every little typying mistake....
>>
>> Oh, and I'm using the Student Vresion if that changes anything.
>>
>> Thank you very much,
>>
>> Vicky A.
>>
>> Sent via Deja.com http://www.deja.com/
>> Before you buy.
>
> I will volunteer to be a 'wise person'...
>
> There ARE a few languages that immediately tell you where
> and what the syntax error is. Some BASICs do this. Some
> FORTRANs are better than others in this regard.
>
> IDL: The on-line help is In My Opinion the best way (so far)
> to look up syntax. It is quick and only a few button-clicks
> away (at least on my Linux box). To keep it available, I
> start another IDL session in another 'screen', and open up
> the IDL hypertext help. I keep the session and the help
> window open all day. I don't have to wait for the help
> window to be created, nor the help file contents to be read
> and formatted every time I have a simple syntax question.
>

You should look into the new idlwave emacs mode, if emacs is your thing. It really changes the way I program IDL. It has built-in abbreviations for common structures like IF blocks, PRO, FUNCTION, CASE, WHILE etc. blocks, among many other useful ones. It notices if you use the wrong type of END (ENDIF in a FOR loop, for example). And, thanks to Carsten Dominik, now allows you to do completion and syntax queries on the function or procedure... you can even compile your own routines into a library for searching.

An example with, e.g. [M-Tab], representing where I hit meta-tab or other key combo, and another buffer represented by *'s:

```
a=cu[M-Tab]

*Click mouse-2 on a completion to select it.
*In this buffer, type RET to select the completion near point.
*
*Possible completions are:
*cumulate      curvefit
```

or if I have given it more:

```
a=cur[M-Tab]
```

becomes

```
a=curvefit([M-Tab]
```

which gives me

```
*Click mouse-2 on a completion to select it.
*In this buffer, type RET to select the completion near point.
*
*Possible completions are:
*CHISQ       FUNCTION_NAME
*ITER        ITMAX
*NODERIVATIVE      TOL
```

to see which keywords I can use. Clicking on one inserts it... e.g. clicking NODERIVATIVE yields.

```
a=curvefit(NODERIVATIVE=[C-c-?]
```

(that's a Control-c-?) gives me the routine info:

```
*Usage:    Result = CURVEFIT(X, Y, Weights, A [, Sigma])
*Keywords: CHISQ FUNCTION_NAME ITER ITMAX NODERIVATIVE TOL
*Origin:   system routine
```

Origin could be a library file (compiled from all your favorite routines), scanned buffers which are open in Emacs, or routines compiled into the IDLWAVE Shell (idl running as a subprocess to Emacs). Definitely faster than the online help when you know a routine but have just forgotten whether a keyword is like "NOZERO" or "NO_COPY" (my favorite dichotomy), or which argument goes first.

Mixed case or forced case completion (including object method completion) is available, and the whole thing is very configureable. Take a look at http://www.strw.leidenuniv.nl/~dominik/Tools/idlwave/ to find out more. Oh and did I mention it indents, highlights, colors, and otherwise arranges your code according to your chosen preferences? Another one of my favorites... with cursor on or in a routine (procedure or function), you hit C-c-v to pull up the source code to that routine! Debugging support is also built-in with the Shell, and it automatically pulls up files

--

```
 J.D. Smith                          |*|      WORK: (607) 255-5842
 Cornell University Dept. of Astronomy  |*|            (607) 255-6263
 304 Space Sciences Bldg.              |*|       FAX: (607) 255-5875
 Ithaca, NY 14853                      |*|
```