Subject: Re: IDL 8.0 bug -- line number of errors not given
Posted by Paul Van Delst[1] on Wed, 13 Oct 2010 20:53:38 GMT
View Forum Message <> Reply to Message

David Fanning wrote:
> Paulo Penteado writes:
>
>> It seems that I just got bitten by this new behavior. A simple example
>> that shows the problem:
>>
>> IDL> w=window()
>> IDL> w.title='a'
>> % Expression must be a structure in this context: OTITLE.
>> % Execution halted at: $MAIN$
>>
>> So it seems that somewhere in the Graphics routines (I am guessing in
>> a setproperty method) there is a bug, but the error message does not
>> tell where. I will have to start digging now.
>
> Yes, try grepping all your ON_ERROR,2 lines and changing
> them to ON_ERROR,0. There should only be a couple of thousand. ;-)

This is actually quite trivial.

I had to do this a few years ago on my Fortran codes - funnily enough it was to do with the error handling module!. I
needed (wanted?) to change all instances of "error_handler" to "Message_Handler".

Suffice it to say over the decades I had a fair amount of Fortran code that used my error_handler module. The initial
solution was to use ruby on the command line:

$ ruby -pi.bak -e "gsub(/error_handler/i,'Message_Handler')" *.f90

Bugger me if that didn't work. I thought it was neato enough to create a generic ruby script to do similar stuff (shown
below).

Let me know if it works on your code..... :o)

cheers,

paulv

p.s. A bloke I work with did exactly the same thing but using perl.

<-----begin----->

```
#!/usr/bin/env ruby

# == Synopsis
#
# Substitute strings in files
#
# == Usage
#
# strsub.rb [OPTION] old new file1 [file2 file3 file4 ...]
#
# == Options
#
# --help  (-h):
#    you're looking at it
#
# --ignore-case  (-i)
#    ignore the case of the string to replace
#
# --no-backup  (-n):
#    do not create a backup file before modification.
#    Backup file name is "filename".bak
#
# == Arguments
#
# old:
#    string to replace
#
# new:
#    replacement string
#
# file1 [file2 file3 file4 ...]:
#    files on which to perform the substitution
#
#
# Written by:: Paul van Delst, 11-Aug-2006
#

require 'getoptlong'
require 'rdoc/usage'

# Specify accepted options
options=GetoptLong.new(
  [ "--help",       "-h", GetoptLong::NO_ARGUMENT ],
  [ "--ignore-case","-i", GetoptLong::NO_ARGUMENT ],
  [ "--no-backup",  "-n", GetoptLong::NO_ARGUMENT ] )

# Define backup default
$-i=".bak"
```

```
# Define defaults
ignorecase=0

# Parse the command line options
begin
  options.each do |opt, arg|
    case opt
      when "--help"
        RDoc::usage
        exit 0
      when "--ignore-case"
        ignorecase=Regexp::IGNORECASE
      when "--no-backup"
        $-i=""
    end
  end
rescue StandardError=>errormessage
  puts "ERROR: #{error_message}"
  RDoc::usage
  exit 1
end

# Check for args
if ARGV.empty?
  puts "ERROR: No arguments supplied"
  RDoc::usage
  exit 1
end

# Get the strings
old=ARGV.shift
new=ARGV.shift

# Build the regex
re=Regexp.new(/#{old}/,ignorecase)

# Inplace edit the file
ARGF.each do |line|
  line.gsub!(re, new)
  puts line
end
<-----end----->
```