NAME

error – insert compiler error messages at right source lines

SYNOPSIS

error [**–n**] [**–q**] [**–s**] [**–v**] [**–t** *suffixlist*] [**–I** *ignorefile*] [*filename*]

DESCRIPTION

error analyzes error messages produced by a number of compilers and language processors. It replaces the painful, traditional methods of scribbling abbreviations of errors on paper, and permits error messages and source code to be viewed simultaneously.

error looks at error messages, either from the specified file *filename* or from the standard input, and:

- Determines which language processor produced each error message.
- Determines the file name and line number of the erroneous line.
- Inserts the error message into the source file immediately preceding the erroneous line.

Error messages that can't be categorized by language processor or content are not inserted into any file, but are sent to the standard output. **error** touches source files only after all input has been read.

error is intended to be run with its standard input connected with a pipe to the error message source. Some language processors put error messages on their standard error file; others put their messages on the standard output. Hence, both error sources should be piped together into **error.** For example, when using the **csh** syntax, the following command analyzes all the error messages produced by whatever programs **make**(1S) runs when making lint:

example% make –s lint | & error

error knows about the error messages produced by: **as**(1), **cpp**(1), **ld**(1), **cc**(1B), **make**(1S) and other compilers. For all languages except Pascal, error messages are restricted to one line. Some error messages refer to more than one line in more than one file, in which case **error** duplicates the error message and inserts it in all the appropriate places.

OPTIONS

–n Do *not* touch any files; all error messages are sent to the standard output.

–q **error** asks whether the file should be touched. A '**y**' or '**n**' to the question is necessary to continue. Absence of the **–q** option implies that all referenced files (except those referring to discarded error messages) are to be touched.

–s Print out statistics regarding the error categorization.

–v After all files have been touched, overlay the visual editor **vi** with it set up to edit all files touched, and positioned in the first touched file at the first error. If **vi**(1) can't be found, try **ex**(1) or **ed**(1) from standard places.

–t *suffixlist*
Take the following argument as a suffix list. Files whose suffices do not appear in the suffix list are not touched. The suffix list is dot separated, and '*' wildcards work. Thus the suffix list:

.c.y.f*.h

allows **error** to touch files ending with '**.c**', '**.y**', '**.f***' and '**.h**'.

error catches interrupt and terminate signals, and terminates in an orderly fashion.

EXAMPLES

In the following C shell **(/usr/bin/csh)** example, **error** takes its input from the FORTRAN complier:

example% f77 –c *any***.f |& error"** *options*

Here is the same example using the Korn shell **(/usr/bin/ksh)**:

example% f77 –c *any***.f 2>&1 | error"** *options*

USAGE

error does one of six things with error messages.

synchronize Some language processors produce short errors describing which file they are processing. **error** uses these to determine the file name for languages that do not include the file name in each error message. These synchronization messages are consumed entirely by **error**.

discard Error messages from **lint** that refer to one of the two **lint** libraries, **/usr/lib/lint/llib-lc** and **/usr/lib/lint/llib-port** are discarded, to prevent accidentally touching these libraries. Again, these error messages are consumed entirely by **error**.

nullify Error messages from **lint** can be nullified if they refer to a specific function, which is known to generate diagnostics which are not interesting. Nullified error messages are not inserted into the source file, but are written to the standard output. The names of functions to ignore are taken from either the file named **.errorrc** in the user's home directory, or from the file named by the **–I** option. If the file does not exist, no error messages are nullified. If the file does exist, there must be one function name per line.

not file specific Error messages that can't be intuited are grouped together, and written to the standard output before any files are touched. They are not inserted into any source file.

file specific Error messages that refer to a specific file but to no specific line are written to the standard output when that file is touched.

true errors Error messages that can be intuited are candidates for insertion into the file to which they refer.

Only true error messages are inserted into source files. Other error messages are consumed entirely by **error** or are written to the standard output. **error** inserts the error messages into the source file on the line preceding the line number in the error message.

Each error message is turned into a one line comment for the language, and is internally flagged with the string **###** at the beginning of the error, and **%%%** at the end of the error. This makes pattern searching for errors easier with an editor, and allows the messages to be easily removed. In addition, each error message contains the source line number for the line the message refers to. A reasonably formatted source program can be recompiled with the error messages still in it, without having the error messages themselves cause future errors. For poorly formatted source programs in free format languages, such as C or Pascal, it is possible to insert a comment into another comment, which can wreak havoc with a future compilation. To avoid this, format the source program so there are no language statements on the same line as the end of a comment.

FILES

~/.errorrc	function names to ignore for **lint** error messages
/dev/tty	user's teletype

SEE ALSO

as(1), **cc**(1B), **cpp**(1), **csh**(1), **ed**(1), **ex**(1), **make**(1S), **ld**(1), **vi**(1)

BUGS

Opens the tty-device directly for user input.

Source files with links make a new copy of the file with only one link to it.

Changing a language processor's error message format may cause **error** to not understand the error message.

error, since it is purely mechanical, will not filter out subsequent errors caused by "floodgating" initiated by one syntactically trivial error. Humans are still much better at discarding these related errors.

Pascal error messages belong after the lines affected, error puts them before. The alignment of the **'|'** marking the point of error is also disturbed by **error.**

error was designed for work on CRT 's at reasonably high speed. It is less pleasant on slow speed terminals, and was not designed for use on hardcopy terminals.