Full credit is given to the above companies including the OS that this PDF file was generated!

Red Hat Enterprise Linux Release 9.2 Manual Pages on 'cflow.1p' command

$ man cflow.1p

CFLOW(1P) POSIX Programmer's Manual CFLOW(1P)

PROLOG

This manual page is part of the POSIX Programmer's Manual. The Linux implementation of this interface may differ (consult the corresponding Linux manual page for details of Linux behavior), or the interface may not be implemented on Linux.

NAME

cflow ? generate a C-language flowgraph (DEVELOPMENT)

SYNOPSIS

cflow [-r] [-d num] [-D name[=def]]... [-i incl] [-I dir]... [-U dir]... file...

DESCRIPTION

The cflow utility shall analyze a collection of object files or assem? bler, C-language, lex, or yacc source files, and attempt to build a graph, written to standard output, charting the external references.

OPTIONS

The cflow utility shall conform to the Base Definitions volume of POSIX.1?2017, Section 12.2, Utility Syntax Guidelines, except that the order of the -D, -I, and -U options (which are identical to their in? terpretation by c99) is significant.

The following options shall be supported:

-d num Indicate the depth at which the flowgraph is cut off. The ap? plication shall ensure that the argument num is a decimal in?

teger. By default this is a very large number (typically greater than 32000). Attempts to set the cut-off depth to a non-positive integer shall be ignored.

-i incl Increase the number of included symbols. The incl option-ar? gument is one of the following characters:

x Include external and static data symbols. The default shall be to include only functions in the flowgraph.

_ (Underscore) Include names that begin with an <under? score>. The default shall be to exclude these func? tions (and data if -i x is used).

-r Reverse the caller:callee relationship, producing an inverted listing showing the callers of each function. The listing shall also be sorted in lexicographical order by callee.

OPERANDS

The following operand is supported:

file The pathname of a file for which a graph is to be generated. Filenames suffixed by .l shall shall be taken to be lex in? put, .y as yacc input, .c as c99 input, and .i as the output of c99 -E. Such files shall be processed as appropriate, de? termined by their suffix.

Files suffixed by .s (conventionally assembler source) may have more limited information extracted from them.

STDIN

Not used.

INPUT FILES

The input files shall be object files or assembler, C-language, lex, or yacc source files.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of cflow:

LANG Provide a default value for the internationalization vari? ables that are unset or null. (See the Base Definitions vol? ume of POSIX.1?2017, Section 8.2, Internationalization Vari?

ables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL If set to a non-empty string value, override the values of all the other internationalization variables.

LC_COLLATE

Determine the locale for the ordering of the output when the -r option is used.

LC_CTYPE Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files).

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard er? ror.

NLSPATH Determine the location of message catalogs for the processing of LC_MESSAGES.

ASYNCHRONOUS EVENTS

Default.

STDOUT

The flowgraph written to standard output shall be formatted as follows:

"%d %s:%s\n", <reference number>, <global>, <definition>

Each line of output begins with a reference (that is, line) number, followed by indentation of at least one column position per level. This is followed by the name of the global, a <colon>, and its defini? tion. Normally globals are only functions not defined as an external or beginning with an <underscore>; see the OPTIONS section for the -i in? clusion option. For information extracted from C-language source, the definition consists of an abstract type declaration (for example, char *) and, delimited by angle brackets, the name of the source file and the line number where the definition was found. Definitions extracted from object files indicate the filename and location counter under which the symbol appeared (for example, text).

Once a definition of a name has been written, subsequent references to that name contain only the reference number of the line where the definition can be found. For undefined references, only "<>" shall be written.

STDERR

The standard error shall be used only for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

0 Successful completion.

>0 An error occurred.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

Files produced by lex and yacc cause the reordering of line number declarations, and this can confuse cflow. To obtain proper results, the input of yacc or lex must be directed to cflow.

EXAMPLES

Given the following in file.c:

```
int i;
int f();
int g();
int h();
int
main()
{
    f();
    g();
    f();
```

```
}
int
f()
{
    i = h();
}
```

The command:

```
cflow -i x file.c
```

produces the output:

```
1 main: int(), <file.c 6>
2   f: int(), <file.c 13>
3       h: <>
4       i: int, <file.c 1>
5   g: <>
```

RATIONALE

None.

FUTURE DIRECTIONS

None.

SEE ALSO

c99, lex, yacc

The Base Definitions volume of POSIX.1?2017, Chapter 8, Environment Variables, Section 12.2, Utility Syntax Guidelines

most likely to have been introduced during the conversion of the source files to man page format. To report such errors, see https://www.kernel.org/doc/man-pages/reporting_bugs.html .