NAME

m4 − macro processor

SYNOPSIS

m4 [*OPTION*]... [*FILE*]...

DESCRIPTION

Process macros in FILEs. If no FILE or if FILE is '−', standard input is read.

Mandatory or optional arguments to long options are mandatory or optional for short options too.

Operation modes:

−−help display this help and exit

−−version
output version information and exit

−E, **−−fatal−warnings**
once: warnings become errors, twice: stop execution at first error

−i, **−−interactive**
unbuffer output, ignore interrupts

−P, **−−prefix−builtins**
force a 'm4_' prefix to all builtins

−Q, **−−quiet**, **−−silent**
suppress some warnings for builtins

−−warn−macro−sequence[=*REGEXP*]
warn if macro definition matches REGEXP,

default \$\({[^}]*}\|[0−9][0−9]+\)

Preprocessor features:

−D, **−−define=NAME**[=*VALUE*]
define NAME as having VALUE, or empty

−I, **−−include**=*DIRECTORY*
append DIRECTORY to include path

−s, **−−synclines**
generate '#line NUM "FILE"' lines

−U, **−−undefine**=*NAME*
undefine NAME

Limits control:

−g, **−−gnu**
override **−G** to re−enable GNU extensions

−G, **−−traditional**
suppress all GNU extensions

−H, **−−hashsize**=*PRIME*
set symbol lookup hash table size [509]

−L, **−−nesting−limit**=*NUMBER*
change nesting limit, 0 for unlimited [0]

Frozen state files:

−F, **−−freeze−state**=*FILE*
produce a frozen state on FILE at end

−R, **−−reload−state**=*FILE*
reload a frozen state from FILE at start

Debugging:

−d, **−−debug**[=*FLAGS*]
set debug level (no FLAGS implies 'aeq')

−−debugfile[=*FILE*]
redirect debug and trace output to FILE (default stderr, discard if empty string)

−l, **−−arglength**=*NUM*
restrict macro tracing size

−t, **−−trace**=*NAME*
trace NAME when it is defined

FLAGS is any of:

a show actual arguments

c show before collect, after collect and after call

e show expansion

f say current input file name

i show changes in input files

l say current input line number

p show results of path searches

q quote values as necessary, with a or e flag

t trace for all macro calls, not only traceon'ed

x add a unique macro call id, useful with c flag

V shorthand for all of the above flags

If defined, the environment variable 'M4PATH' is a colon−separated list of directories included after any specified by '−I'.

Exit status is 0 for success, 1 for failure, 63 for frozen file version mismatch, or whatever value was passed to the m4exit macro.

AUTHOR

Written by Rene' Seindal.

REPORTING BUGS

Report bugs to <bug−m4@gnu.org>. GNU M4 home page: <http://www.gnu.org/software/m4/>. General help using GNU software: <http://www.gnu.org/gethelp/>.

COPYRIGHT

SEE ALSO

The full documentation for **m4** is maintained as a Texinfo manual. If the **info** and **m4** programs are properly installed at your site, the command

info m4

should give you access to the complete manual.

NAME

m4 − macro processor (**DEVELOPMENT**)

SYNOPSIS

m4 [-s][-D *name***[=***val***]]...[-U** *name***]...** *file***...**

DESCRIPTION

The *m4* utility is a macro processor that shall read one or more text files, process them according to their included macro statements, and write the results to standard output.

OPTIONS

The *m4* utility shall conform to the Base Definitions volume of IEEE Std 1003.1-2001, Section 12.2, Utility Syntax Guidelines, except that the order of the **-D** and **-U** options shall be significant.

The following options shall be supported:

-s Enable line synchronization output for the *c99* preprocessor phase (that is, **#line** directives).

-D *name***[=***val***]**

Define *name* to *val* or to null if = *val* is omitted.

-U *name*

Undefine *name*.

OPERANDS

The following operand shall be supported:

file A pathname of a text file to be processed. If no *file* is given, or if it is **'-'** , the standard input shall be read.

STDIN

The standard input shall be a text file that is used if no *file* operand is given, or if it is **'-'** .

INPUT FILES

The input file named by the *file* operand shall be a text file.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of *m4*:

LANG Provide a default value for the internationalization variables that are unset or null. (See the Base Definitions volume of IEEE Std 1003.1-2001, Section 8.2, Internationalization Variables 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_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 error.

NLSPATH

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

ASYNCHRONOUS EVENTS

Default.

STDOUT

The standard output shall be the same as the input files, after being processed for macro expansion.

STDERR

The standard error shall be used to display strings with the **errprint** macro, macro tracing enabled by the **traceon** macro, the defined text for macros written by the **dumpdef** macro, or for diagnostic messages.

OUTPUT FILES

None.

EXTENDED DESCRIPTION

The *m4* utility shall compare each token from the input against the set of built-in and user-defined macros. If the token matches the name of a macro, then the token shall be replaced by the macro's defining text, if any, and rescanned for matching macro names. Once no portion of the token matches the name of a macro, it shall be written to standard output. Macros may have arguments, in which case the arguments shall be substituted into the defining text before it is rescanned.

Macro calls have the form:

name(*arg1*, *arg2*, ..., *argn*)

Macro names shall consist of letters, digits, and underscores, where the first character is not a digit. Tokens not of this form shall not be treated as macros.

The application shall ensure that the left parenthesis immediately follows the name of the macro. If a token matching the name of a macro is not followed by a left parenthesis, it is handled as a use of that macro without arguments.

If a macro name is followed by a left parenthesis, its arguments are the comma-separated tokens between the left parenthesis and the matching right parenthesis. Unquoted <blank>s and <newline>s preceding each argument shall be ignored. All other characters, including trailing <blank>s and <newline>s, are retained. Commas enclosed between left and right parenthesis characters do not delimit arguments.

Arguments are positionally defined and referenced. The string **"$1"** in the defining text shall be replaced by the first argument. Systems shall support at least nine arguments; only the first nine can be referenced, using the strings **"$1"** to **"$9"** , inclusive. The string **"$0"** is replaced with the name of the macro. The string **"$#"** is replaced by the number of arguments as a string. The string **"$*"** is replaced by a list of all of the arguments, separated by commas. The string **"$@"** is replaced by a list of all of the arguments separated by commas, and each argument is quoted using the current left and right quoting strings.

If fewer arguments are supplied than are in the macro definition, the omitted arguments are taken to be null. It is not an error if more arguments are supplied than are in the macro definition.

No special meaning is given to any characters enclosed between matching left and right quoting strings, but the quoting strings are themselves discarded. By default, the left quoting string consists of a grave accent (**'`'**) and the right quoting string consists of an acute accent (**'"**); see also the **changequote** macro.

Comments are written but not scanned for matching macro names; by default, the begin-comment string consists of the number sign character and the end-comment string consists of a <newline>. See also the **changecom** and **dnl** macros.

The *m4* utility shall make available the following built-in macros. They can be redefined, but once this is done the original meaning is lost. Their values shall be null unless otherwise stated. In the descriptions below, the term *defining text* refers to the value of the macro: the second argument to the **define** macro, among other things. Except for the first argument to the **eval** macro, all numeric arguments to built-in macros shall be interpreted as decimal values. The string values produced as the defining text of the **decr**, **divnum**, **incr**, **index**, **len**, and **sysval** built-in macros shall be in the form of a decimal-constant as defined in the C language.

changecom

The **changecom** macro shall set the begin-comment and end-comment strings. With no arguments, the comment mechanism shall be disabled. With a single argument, that argument shall become the begin-comment string and the <newline> shall become the end-comment string. With two arguments, the first argument shall become the begin-comment string and the second

argument shall become the end-comment string. Systems shall support comment strings of at least five characters.

changequote
The **changequote** macro shall set the begin-quote and end-quote strings. With no arguments, the quote strings shall be set to the default values (that is, **‘’**). With a single argument, that argument shall become the begin-quote string and the <newline> shall become the end-quote string. With two arguments, the first argument shall become the begin-quote string and the second argument shall become the end-quote string. Systems shall support quote strings of at least five characters.

decr The defining text of the **decr** macro shall be its first argument decremented by 1. It shall be an error to specify an argument containing any non-numeric characters.

define The second argument shall become the defining text of the macro whose name is the first argument.

defn The defining text of the **defn** macro shall be the quoted definition (using the current quoting strings) of its arguments.

divert The *m4* utility maintains nine temporary buffers, numbered 1 to 9, inclusive. When the last of the input has been processed, any output that has been placed in these buffers shall be written to standard output in buffer-numerical order. The **divert** macro shall divert future output to the buffer specified by its argument. Specifying no argument or an argument of 0 shall resume the normal output process. Output diverted to a stream other than 0 to 9 shall be discarded. It shall be an error to specify an argument containing any non-numeric characters.

divnum
The defining text of the **divnum** macro shall be the number of the current output stream as a string.

dnl The **dnl** macro shall cause *m4* to discard all input characters up to and including the next <newline>.

dumpdef
The **dumpdef** macro shall write the defined text to standard error for each of the macros specified as arguments, or, if no arguments are specified, for all macros.

errprint
The **errprint** macro shall write its arguments to standard error.

eval The **eval** macro shall evaluate its first argument as an arithmetic expression, using 32-bit signed integer arithmetic. All of the C-language operators shall be supported, except for:

[]
->
++
--
(*type*)
unary *
*sizeof***,**
.
?:
unary &

and all assignment operators. It shall be an error to specify any of these operators. Precedence and associativity shall be as in the ISO C standard. Systems shall support octal and hexadecimal numbers as in the ISO C standard. The second argument, if specified, shall set the radix for the result; the default is 10. The third argument, if specified, sets the minimum number of digits in the result. It shall be an error to specify the second or third argument containing any non-numeric characters.

ifdef If the first argument to the **ifdef** macro is defined, the defining text shall be the second argument. Otherwise, the defining text shall be the third argument, if specified, or the null string, if not.

ifelse The **ifelse** macro takes three or more arguments. If the first two arguments compare as equal strings (after macro expansion of both arguments), the defining text shall be the third argument. If the first two arguments do not compare as equal strings and there are three arguments, the defining text shall be null. If the first two arguments do not compare as equal strings and there are four or five arguments, the defining text shall be the fourth argument. If the first two arguments do not compare as equal strings and there are six or more arguments, the first three arguments shall be discarded and processing shall restart with the remaining arguments.

include
The defining text for the **include** macro shall be the contents of the file named by the first argument. It shall be an error if the file cannot be read.

incr The defining text of the **incr** macro shall be its first argument incremented by 1. It shall be an error to specify an argument containing any non-numeric characters.

index The defining text of the **index** macro shall be the first character position (as a string) in the first argument where a string matching the second argument begins (zero origin), or -1 if the second argument does not occur.

len The defining text of the **len** macro shall be the length (as a string) of the first argument.

m4exit Exit from the *m4* utility. If the first argument is specified, it is the exit code. The default is zero. It shall be an error to specify an argument containing any non-numeric characters.

m4wrap
The first argument shall be processed when EOF is reached. If the **m4wrap** macro is used multiple times, the arguments specified shall be processed in the order in which the **m4wrap** macros were processed.

maketemp
The defining text shall be the first argument, with any trailing **'X'** characters replaced with the current process ID as a string.

popdef The **popdef** macro shall delete the current definition of its arguments, replacing that definition with the previous one. If there is no previous definition, the macro is undefined.

pushdef
The **pushdef** macro shall be equivalent to the **define** macro with the exception that it shall preserve any current definition for future retrieval using the **popdef** macro.

shift The defining text for the **shift** macro shall be all of its arguments except for the first one.

sinclude
The **sinclude** macro shall be equivalent to the **include** macro, except that it shall not be an error if the file is inaccessible.

substr The defining text for the **substr** macro shall be the substring of the first argument beginning at the zero-offset character position specified by the second argument. The third argument, if specified, shall be the number of characters to select; if not specified, the characters from the starting point to the end of the first argument shall become the defining text. It shall not be an error to specify a starting point beyond the end of the first argument and the defining text shall be null. It shall be an error to specify an argument containing any non-numeric characters.

syscmd
The **syscmd** macro shall interpret its first argument as a shell command line. The defining text shall be the string result of that command. No output redirection shall be performed by the *m4* utility. The exit status value from the command can be retrieved using the **sysval** macro.

sysval The defining text of the **sysval** macro shall be the exit value of the utility last invoked by the **syscmd** macro (as a string).

traceon
The **traceon** macro shall enable tracing for the macros specified as arguments, or, if no arguments are specified, for all macros. The trace output shall be written to standard error in an unspecified format.

traceoff

The **traceoff** macro shall disable tracing for the macros specified as arguments, or, if no arguments are specified, for all macros.

translit

The defining text of the **translit** macro shall be the first argument with every character that occurs in the second argument replaced with the corresponding character from the third argument.

undefine

The **undefine** macro shall delete all definitions (including those preserved using the **pushdef** macro) of the macros named by its arguments.

undivert

The **undivert** macro shall cause immediate output of any text in temporary buffers named as arguments, or all temporary buffers if no arguments are specified. Buffers can be undiverted into other temporary buffers. Undiverting shall discard the contents of the temporary buffer. It shall be an error to specify an argument containing any non-numeric characters.

EXIT STATUS

The following exit values shall be returned:

0 Successful completion.

>0 An error occurred

If the **m4exit** macro is used, the exit value can be specified by the input file.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

The **defn** macro is useful for renaming macros, especially built-ins.

EXAMPLES

If the file **m4src** contains the lines:

```
The value of ‘VER’ is "VER".
ifdef(‘VER’, "VER" is defined to be VER., VER is not defined.)
ifelse(VER, 1, "VER" is ‘VER’.)
ifelse(VER, 2, "VER" is ‘VER’., "VER" is not 2.)
end
```

then the command

```
m4 m4src
```

or the command:

```
m4 -U VER m4src
```

produces the output:

```
The value of VER is "VER".
VER is not defined.

VER is not 2.
end
```

The command:

```
m4 -D VER m4src
```

produces the output:

```
The value of VER is "".
VER is defined to be .


VER is not 2.
end
```

The command:

```
m4 -D VER=1 m4src
```

produces the output:

```
The value of VER is "1".
VER is defined to be 1.
VER is 1.
VER is not 2.
end
```

The command:

```
m4 -D VER=2 m4src
```

```
produces the output:
The value of VER is "2".
VER is defined to be 2.


VER is 2.
end
```

RATIONALE

None.

FUTURE DIRECTIONS

None.

SEE ALSO

c99

COPYRIGHT