WIKIPEDIA

m4 (computer language)

m4 is a general-purpose macro processor included in most Unix-like operating systems, and is a component of the POSIX standard.

m4

Paradigm	macro
Designed by	Brian Kernighan, Dennis Ritchie
First appeared	1977
Major implementations	
GNU m4	

The language was designed by Brian Kernighan and Dennis Ritchie for the original versions of UNIX. It is an extension of an earlier macro processor **m3**, written by Ritchie for an unknown AP-3 minicomputer.[1]

The macro preprocessor operates as a text-replacement tool. It is employed to re-use text templates, typically in computer programming applications, but also in text editing and text-processing applications. Most users require m4 as a dependency of GNU autoconf.

Contents

History

Macro processors became popular when programmers commonly used assembly language. In those early days of programming, programmers noted that much of their programs consisted of repeated text, and they invented simple means for reusing this text. Programmers soon discovered the advantages not only of reusing entire blocks of text, but also of substituting different values for similar parameters. This defined the usage range of macro processors.

In 1960s, an early general-purpose macro processor M^6 was in use at AT&T Bell Laboratories, which was developed by Douglas McIlroy, Robert Morris and Andrew Hall.[2]

Kernighan and Ritchie developed m4 in 1977, basing it on the ideas of Christopher Strachey. The distinguishing features of this style of macro preprocessing included:

- free-form syntax (not line-based like a typical macro preprocessor designed for assembly-language processing)
- the high degree of re-expansion (a macro's arguments get expanded twice: once during scanning and once at interpretation time)

The implementation of Rational Fortran used m4 as its macro engine from the beginning; and most Unix variants ship with it.

As of 2020 many applications continue to use m4 as part of the GNU Project's autoconf. It also appears in the configuration process of sendmail (a widespread mail transfer agent) and for generating footprints in the gEDA toolsuite. The SELinux Reference Policy relies heavily on the m4 macro processor.

m4 has many uses in code generation, but (as with any macro processor) problems can be hard to debug.[3]

Features

m4 offers these facilities:

- a free-form syntax, rather than line based syntax
- a high degree of macro expansion (arguments get expanded during scan and again during interpretation)
- text replacement
- parameter substitution
- file inclusion
- string manipulation
- conditional evaluation
- arithmetic expressions
- system interface
- programmer diagnostics
- programming language independent
- human language independent
- provides programming language capabilities

Unlike most earlier macro processors, m4 does not target any particular computer or human language; historically, however, its development originated for supporting the Ratfor dialect of Fortran. Unlike some other macro processors, m4 is Turing-complete as well as a practical programming language.

Unquoted identifiers which match defined macros are replaced with their definitions. Placing identifiers in quotes suppresses expansion until possibly later, such as when a quoted string is expanded as part of macro replacement. Unlike most languages, strings in m4 are quoted using the backtick (`) as the starting delimiter, and apostrophe (') as the ending delimiter. Separate starting and ending delimiters allows the arbitrary nesting of quotation marks in strings to be used, allowing a fine degree of control of how and when macro expansion takes place in different parts of a string.

Example

The following fragment gives a simple example that could form part of a library for generating HTML code. It defines a commented macro to number sections automatically:

```
divert(-1)

m4 has multiple output queues that can be manipulated with the
`divert' macro. Valid queues range from 0 to 10, inclusive, with
the default queue being 0. As an extension, GNU m4 supports more
diversions, limited only by integer type size.

Calling the `divert' macro with an invalid queue causes text to be
```

```
discarded until another call.  Note that even while output is being
discarded, quotes around `divert' and other macros are needed to
prevent expansion.

# Macros aren't expanded within comments, meaning that keywords such
# as divert and other built-ins may be used without consequence.

# HTML utility macro:

define(`H2_COUNT', 0)

# The H2_COUNT macro is redefined every time the H2 macro is used:

define(`H2',
    `define(`H2_COUNT', incr(H2_COUNT))<h2>H2_COUNT. $1</h2>')

divert(1)dnl
dnl
dnl The dnl macro causes m4 to discard the rest of the line, thus
dnl preventing unwanted blank lines from appearing in the output.
dnl
H2(First Section)
H2(Second Section)
H2(Conclusion)
dnl
divert(0)dnl
dnl
<HTML>
undivert(1)dnl One of the queues is being pushed to output.
</HTML>
```

Processing this code with m4 generates the following text:

```
<HTML>
<h2>1. First Section</h2>
<h2>2. Second Section</h2>
<h2>3. Conclusion</h2>
</HTML>
```

Implementations

FreeBSD, NetBSD, and OpenBSD provide independent implementations of the m4 language. Furthermore, the Heirloom Project Development Tools includes a free version of the m4 language, derived from OpenSolaris.

M4 has been included in the Inferno operating system. This implementation is more closely related to the original m4 developed by Kernighan and Ritchie in Version 7 Unix than its more sophisticated relatives in UNIX System V and POSIX.[4]

GNU m4 is an implementation of m4 for the GNU Project.[5][6] It is designed to avoid many kinds of arbitrary limits found in traditional m4 implementations, such as maximum line lengths, maximum size of a macro and number of macros. Removing such arbitrary limits is one of the stated goals of the GNU Project.[7]

The GNU Autoconf package makes extensive use of the features of GNU m4.

GNU m4 is currently maintained by Gary V. Vaughan and Eric Blake. Released under the terms of the GNU General Public License, GNU m4 is free software.

See also

- C preprocessor
- Macro (computer science)
- Make

- Template processor
- Web template system

References

1. Brian W. Kernighan and Dennis M. Ritchie. The m4 macro processor. Technical report, Bell Laboratories, Murray Hill, New Jersey, USA, 1977. pdf (http://wolfram.schneider.org/bsd/7thEdManVol2/m4/m4.pdf)
2. Cole, A. J. (1981). *Macro Processors* (2nd, revised ed.). CUP Archive. p. 254.
3. Kenneth J. Turner. Exploiting the m4 macro language. Technical Report CSM-126, Department of Computing Science and Mathematics, University of Stirling, Scotland, September 1994. pdf (http://www.cs.stir.ac.uk/~kjt/research/pdf/expl-m4.pdf)
4. `m4(1) (http://man.cat-v.org/inferno/1/m4)` – Inferno General commands Manual
5. GNU m4 web site "GNU M4" (https://www.gnu.org/software/m4/), accessed January 25, 2020.
6. GNU m4 manual, online and for download in HTML, PDF, and other forms. "GNU M4 — GNU macro processor" (https://www.gnu.org/software/m4/manual/index.html), accessed January 25, 2020.
7. "GNU Coding Standards: Writing Robust Programs" (https://www.gnu.org/prep/standards/html_node/Semantics.html). quote: "Avoid arbitrary limits on the length or number of any data structure".

External links

- GNU m4 website (https://www.gnu.org/software/m4/m4.html)
- GNU m4 manual (https://www.gnu.org/software/m4/manual/m4.html)
- m4 tutorial (http://mbreen.com/m4.html)
- Macro Magic: m4, Part One (http://www.jpeek.com/articles/linuxmag/2005-02/) and Part Two (http://www.jpeek.com/articles/linuxmag/2005-03/)

Retrieved from "https://en.wikipedia.org/w/index.php?title=M4_(computer_language)&oldid=979542448"

This page was last edited on 21 September 2020, at 10:07 (UTC).