NAME

groff_trace – groff macro package trace.tmac

SYNOPSIS

groff –m trace [*option* . . .] [*input-file* . . .]

DESCRIPTION

The *trace* macro package of **groff**(1) can be a valuable tool for debugging documents written in the roff formatting language. A call stack trace is protocolled on standard error, this is, a diagnostic message is emitted on entering and exiting of a macro call. This greatly eases to track down an error in some macro.

This tracing process is activated by specifying the groff or troff command-line option **–m trace**. This works also with the **groffer**(1) viewer program. A finer control can be obtained by including the macro file within the document by the groff macro call **.mso trace.tmac**. Only macros that are defined after this line are traced.

If the command-line option **–r trace-full=1** is given (or if this register is set in the document), number and string register assignments together with some other requests are traced also.

If some other macro package should be traced as well it must be specified after **–m trace** on the command line.

The macro file **trace.tmac** is unusual because it does not contain any macros to be called by a user. Instead, the existing macro definition and appending facilities are modified such that they display diagnostic messages.

EXAMPLES

In the following examples, a roff fragment is fed into groff via standard input. As we are only interested in the diagnostic messages (standard error) on the terminal, the normal formatted output (standard output) is redirected to the nirvana device */dev/null*. The resulting diagnostic messages are displayed directly below the corresponding example.

Command line option

Example:

```
sh# echo '.
> .de test_macro
> ..
> .test_macro
> .test_macro some dummy arguments
> ' | groff -m trace > /dev/null

*** .de test_macro
*** de trace enter: .test_macro
*** trace exit: .test_macro
*** de trace enter: .test_macro "some" "dummy" "arguments"
*** trace exit: .test_macro "some" "dummy" "arguments"
```

The entry and the exit of each macro call is displayed on the terminal (standard output) — together with the arguments (if any).

Nested macro calls

Example:

```
sh# echo '.
> .de child
> ..
> .de parent
> .child
> ..
> .parent
> ' | groff -m trace > /dev/null
```

```
*** .de child
*** .de parent
*** de trace enter: .parent
 *** de trace enter: .child
 *** trace exit: .child
*** trace exit: .parent
```

This shows that macro calls can be nested. This powerful feature can help to tack down quite complex call stacks.

Activating with .mso

Example:

```
sh# echo '.
> .de before
> ..
> .mso trace.tmac
> .de after
> ..
> .before
> .after
> .before
> ' | groff > /dev/null

*** de trace enter: .after
*** trace exit: .after
```

Here, the tracing is activated within the document, not by a command-line option. As tracing was not active when macro *before* was defined, no call of this macro is protocolled; on the other hand, the macro *after* is fully protocolled.

PROBLEMS

Because **trace.tmac** wraps the **.de** request (and its cousins), macro arguments are expanded one level more. This causes problems if an argument contains four backslashes or more to prevent too early expansion of the backslash. For example, this macro call

```
.foo \\\\n[bar]
```

normally passes '\\n[bar]' to macro '.foo', but with the redefined **.de** request it passes '\n[bar]' instead.

The solution to this problem is to use groff's **\E** escape which is an escape character not interpreted in copy mode, for example

```
.foo \En[bar]
```

FILES

The *trace* macros are kept in the file **trace.tmac** located in the *tmac directory*; see **groff_tmac**(5) for details.

ENVIRONMENT

GROFF_TMAC_PATH

A colon-separated list of additional tmac directories in which to search for macro files; see **groff_tmac**(5) for details.

AUTHORS

The *trace* macro packages was written by James Clark. This document was written by Bernd Warken ⟨groff−bernd.warken−72@web.de⟩.

SEE ALSO

Groff: The GNU Implementation of troff, by Trent A. Fisher and Werner Lemberg, is the primary *groff* manual. You can browse it interactively with "info groff".

groff(1)
: An overview of the groff system.

troff(1) For details on option **-m**.

groffer(1)
: A viewer program for all kinds of roff documents.

groff_tmac(5)
: A general description of groff macro packages.

groff(7)
: A short reference for the groff formatting language.