NAME

groff_toc – a GNU roff macro framework for table of contents collation

DESCRIPTION

The **groff_toc** macro package provides a minimal framework for managing the specification, and the collation of table of contents entries. It is intended that this minimal framework should be extended, by the addition of user-defined macros, to control the layout of tables of contents.

USAGE

The **groff_toc** macro framework may be loaded directly from the **groff**(1) command line:

groff [*options* ...] **–mtoc** [*files* ...]

Alternatively, it may be found to be more convenient to load it from within a **groff**(7) document source file, or from some other dependent macro package:

.mso toc.tmac

All features of the **groff_toc** framework are accessed through a single primary macro, named **toc**, which should be called in accordance with the syntax specification:

.toc *opcode* [*arglist* ...]

The *opcode* argument is required; it causes the call to be redirected to a supplementary macro, named **.toc.***opcode*, which may correspond to one of the predefined actions, as specified below, or it may be a user-defined extension macro, to perform the desired operation.

The remaining arguments, designated by [*arglist* ...] are formally optional; however, depending on the particular *opcode* which has been specified, at least some of these formally optional arguments may be required.

The predefined **groff_toc** macros correspond to **toc** macro calls of the form:

.toc file [*name*]
.toc put *opcode* [*arglist* ...]
.toc error *message text* ...

The operation of each of these predefined **toc** macro forms is described below:

.toc file [*name*]
Closes any file stream, which may have been opened by a previous **.toc file** assignment, and opens *name* as a new file stream, for collection of table of contents reference data. if the *name* argument is omitted, or if the **groff**(7) input data stream is being processed by **pdfroff**(1), collected table of contents reference data will be directed to the standard error output data stream, rather than to a named file.

When the collected table of contents reference data is to be directed to a named file, it will normally be necessary to invoke **groff**(1) in *unsafe* mode, (that is, with its **–U** option in effect); however, in the special case of processing with **pdfroff**(1), this restriction may be relaxed, because the collected data will be directed through the standard error stream, whence **pdfroff**(1) will redirect it to the named file.

.toc put *opcode* [*arglist* ...]
Emit a table of contents entry specification to the nominated reference data collection stream; this reference takes the form:

```
\*[TOC.REQUEST] opcode [arglist ...]
```

with the default value of `\*[TOC.REQUEST]` being specified as `.toc`, (as discussed in the **CONTROL VARIABLES** section, below); thus, the table of contents reference data is recorded as a sequence of macro calls, which will then be evaluated, when this data is read back into the document source input stream.

Note that, regardless of how **TOC.REQUEST** may have been defined, the onus falls on the user, to furnish any macros which may be required to evaluate each

```
\*[TOC.REQUEST] opcode [arglist ...]
```

request which has been inserted into the table of contents reference data stream, by any

.toc put macro call; when **TOC.REQUEST** retains its default assignment, this implies that the user must furnish a **.toc.***opcode* macro definition for each and every distinct *opcode* which is specified, in any such **.toc put** macro call.

.toc error *message text . . .*
Emit a diagnostic message, on the standard error output stream; this message is presented in the form:

```
toc macro error: message text ...
```

This form of the **toc** macro is typically used to report issues which are encountered when evaluating other macros, within the context of the **groff_toc** framework, (both those which are included within the basic framework, and user-defined extensions to it).

On their own, the three variants of the **toc** macro, as described above, will not be sufficient to lay out a table of contents; thus, it will be necessary for the user to define at least one, and possibly a collection of several supplementary macros, to fulfil the evaluation of the macro sequence, as written to, and subsequently read back from, the table of contents reference data collection stream.

CONTROL VARIABLES

The **groff_toc** macro framework defines *one* **groff**(7) string, namely **TOC.REQUEST**, which is used to specify the first token — nominally a macro name, preceded by the **groff**(7) *control character* — in each record written to the table of contents reference data stream, by the **toc put** macro. By default, this is defined as **.toc**, which implies that each record in this data stream will represent a macro call to another variant — which, of necessity, should usually be user-defined — of the **toc** macro.

After loading the *toc.tmac* macro file, the user may redefine the **TOC.REQUEST** string, thus substituting an alternative to the **toc** macro, for laying out the table of contents; (any such substitute macro will usually also need to be furnished by the user).

FILES

/usr/local/share/groff/site-tmac/toc.tmac
Implements the **groff_toc** framework.

/usr/local/share/groff/site-tmac/spdf-toc.tmac
Provides a concrete working example of integration of the **groff_toc** macro framework with **pdfroff**(1), the *spdf.tmac*, and the *pdfmark.tmac* macros.

CAVEATS AND BUGS

When using **pdfroff**(1) as the formatter, only the default setting of **.toc** is suitable as an assignment for **TOC.REQUEST**; **pdfroff**(1) will not recognize any value, other than **.toc**, as a valid record signature for a table of contents data reference.

EXAMPLES

To activate the **groff_toc** framework, initialize the reference data collector, and to insert a formatted table of contents into a document, add markup to the document source, similar to the following, at the point at which the table of contents is to be placed:

```
.mso toc.tmac
.
.\" Initialization of the reference data collector may
.\" overwrite any existing content in the associated file,
.\" thus, this file must be read in, BEFORE the collector
.\" file stream is assigned.  Additionally, any required
.\" toc.extension macros must be defined here, BEFORE
.\" reading any existing content from the named file.
.
.so foo.toc
.toc file foo.toc
```

To add entries into the table of contents, place macro calls similar to the following, at each point throughout the document source, to which a table of contents should refer:

```
.toc put pageref \n% reference text ...
```

Notice that, to service the macro calls which this will insert into the table of contents reference data

file, a user-defined **toc.***extension* macro, named **toc.pageref**, must be defined, *before* the file is read in; (that is, it *must* have been defined *before*, or at least *at*, the position of the comment, within the initial example sequence of requests, above, which indicates this requirement). Assuming that tab stops are already set up, as appropriate for laying out the table of contents, at the time when the reference data file is read back, a suitable, if rudimentary, implementation for such an extension macro might be:

```
.de toc.pageref
.\" Usage: .toc pageref page-number reference text ...
.\"
.\" Placing the page number as the first argument, before
.\" the reference text arguments, may seem counterintuitive;
.\" it is done this way to avoid any requirement to iterate
.\" over the reference text, to locate the page number.
.\"
.   nr \\$0.page \\$1       \" save page number ...
.   shift                   \" and remove from arguments
.   nop \\$* \a\t\\$0.page  \" emit TOC entry
.   rr \\$0.page            \" clean up local storage
..
```

The preceding examples are fairly trivial; for a more comprehensive working example of **groff_toc** usage, see the *spdf-toc.tmac* file, which accompanies the *groff-pdfmark* distribution.

AUTHORS

The **groff_toc** macro framework is provided as an adjunct to the *groff-pdfmark* package, which was written by Keith Marshall <author@address.hidden>; it is independently maintained at Keith's *groff-pdfmark* project hosting web-site <https://savannah.nongnu.org/projects/groff-pdfmark/>, whence the latest version may *always* be obtained.

SEE ALSO

groff(1), **pdfroff**(1), **groff**(7)

More comprehensive documentation on the use of the *groff-pdfmark* macro suite, which incorporates the **groff_toc** framework, may be found, in PDF format, in the reference guide "*Portable Document Format Publishing with GNU Troff*", which has also been written by Keith Marshall; the most recently published version of this guide may be read online, by following the appropriate document reference link on the *groff-pdfmark* project hosting web-site <https://savannah.nongnu.org/projects/groff-pdfmark/>, whence a copy may also be downloaded.