NAME

groff-mspdf – a macro package binding pdfmark with ms

SYNOPSIS

pdfroff −mspdf [*option* ...] [*file* ...]
pdfroff −m spdf [*option* ...] [*file* ...]

DESCRIPTION

groff_mspdf is a full-service document formatting macro package; it fully reproduces the features of **groff_ms**(7), integrated with the **groff_pdfmark**(7) macros.

In addition to implicitly loading the **groff_ms**(7), and the **groff_pdfmark**(7) macros, **groff_mspdf** defines one new register, **.NH**, and several new, or redefined macros, notably **XH**, and **XN**, together with associated hook macros, **XH−INIT**, **XN−INIT**, **XH−REPLACEMENT**, **XN−REPLACEMENT**, and **XH−UPDATE−TOC**; it also defines the **OMIT**, **CS**, and **CE** macros, as specified by **groff_omit**(7).

The **XH**, and **XN** macros may be used after **.SH**, and **.NH**, respectively, to specify text which is to be incroporated into the document body, as (part of) the associated section heading, while also duplicating it to create a table of contents reference, and, in the case of the **groff_mspdf** implementations, a PDF document outline reference, to this heading.

The associated hook macros, collectively, determine the specific features of the **XH**, and the **XN** macro implementations; default implementations for each hook are provided, but any of these may be redefined by the user, to implement features which differ from the defaults.

It may be noted that the **XH**, and **XN** macros, and their associated hooks, originated in **groff_mspdf**. Although simplified variants of these macros have been incorporated into contemporary versions of **groff_ms**(7), since the publication of *GNU roff* version 1.23.0, these are *not* functionally equivalent to the variants defined in **groff_mspdf**; the differences will be explained in the **USAGE** section, below.

USAGE

Whereas the use of **groff_ms**(7) is normally initiated with a command conforming to the synopsis:

groff −ms [*option* ...] [*file* ...]

exploitation of the additionally integrated **groff_pdfmark**(7) macro features will normally necessitate multiple-pass **groff**(1) processing; consequently, the use of **pdfroff**(1), which automatically performs such multiple-pass processing, in preference to only simple single-pass **groff**(1) processing, is *strongly* recommended, as indicated in the **SYNOPSIS** section, above, when the **groff_mspdf** macro package is used as an alternative to **groff_ms**(7).

Since **groff_mspdf** implicitly loads the **groff_ms**(7) and **groff_pdfmark**(7) packages, any feature of either of these packages may be used, as documented in the respective manual page. Furthermore, since **groff_pdfmark**(7) also provides the features documented in the **groff_pdfhref**(7), and the **groff_pdfnote**(7) manual pages, any of these features may also be used, in accordance with their respective documentation, *without* explicitly loading any further macro package.

Conversely, although simplified variants of the **XH** and **XN** macro implementations, derived from their original **groff_mspdf** implementations, have been incorporated into **groff_ms**(7), from the release of *GNU roff* version 1.23.0 onwards, these simplified variants are *not* usage compatible with their original **groff_mspdf** implementations. Whereas the **groff**(7) implementations, if supported by the underlying *GNU roff* distribution, conform to the basic synopses:

.SH [<*outline-level*>]
.XH <*outline-level*> <*heading-text*> ...

.NH <*outline-level*>
.XN <*heading-text*> ...

the **groff_mspdf** implementations conform to the extended synopses:

.SH [<*outline-level*>]
.XH [**−N** <*reference-name*>] [**−S**] [**−X**] <*outline-level*> <*heading-text*> ...

.NH <*outline-level*>
.XN [**−N** <*reference-name*>] [**−S**] [**−X**] <*heading-text*> ...

The options, which provide control of **groff_pdfhref**(7) features, and are supported *exclusively* by the **groff_mspdf** implementations of **XH**, and **XN**, are:

–N <*reference-name*>
Use <*reference-name*> for the destination, to which the associated PDF outline entry refers.

–S Strip, or transliterate, a nominated set of **groff**(7) special characters, and escape sequences, from <*heading-text*>, when copying it to the PDF document outline; this operation is performed by the **sanitize** macro, as described in **groff_sanitize**(7).

–X Create an external cross-reference dictionary entry for <*reference-name*>.

It should be noted that all of these options are specific to the handling of PDF outline entries. Since a bare **groff_ms**(7) implementation makes no provision for creating a PDF outline, its rudimentary **XH**, and **XN** implementations have no need to make any such provision; consequently, these options would have no meaning in the bare **groff_ms**(7) context, so are unsupported, and if specified, they *will* result in incorrect behaviour.

Regardless of whether they originate in a recent **groff_ms**(7) implementation, or legacy fallbacks from **groff_mspdf** are adopted, the initial implementations of both **XH**, and **XN**, are simple stubs; each, when invoked for its first, and only time, following its definition by its originating macro package, performs its own standardized self-initialization, before replacing itself by, and transferring control to, whatever complementary **XH–REPLACEMENT**, or **XN–REPLACEMENT** macro implementation, respectively, is in scope at this time. This initialization strategy ensures that the **groff_mspdf** implementations, of each of these two macros, will override the corresponding **groff_ms**(7) implementations, while still according the user an opportunity to override the **groff_mspdf** implementations, by furnishing their own alternative definitions, *after* **groff_mspdf** has been loaded, but *before* invoking **XH**, or **XN**, as may be appropriate, for the first time.

After initialization, and when their respective **XH–REPLACEMENT**, and **XN–REPLACEMENT** handlers have been installed, the default operation of **XH**, and **XN** reduces to:

- Invocation of the respective **XH–INIT**, or **XN–INIT** callback hook; no arguments are passed, and unless otherwise redefined, by the user, each of these is processed as a “no-op”.
- The macro argument list, together with any specified options, are processed to construct a PDF document outline entry; any specified options are discarded from the argument list. (This step is *not* performed by the default **groff_ms**(7) implementation of either macro).
- Both macros conclude by invoking the *same* additional user-redefinable callback hook, **XH–UPDATE–TOC**, passing all residual arguments from its own argument list.

The default implementation of **XH–UPDATE–TOC**, regardless of whether its origin is to be found in **groff_mspdf**, or in a contemporary version of **groff_ms**(7), is very rudimentary: it discards — i.e. it completely ignores — its <*outline-level*> argument, and simply emits the interpolation of its remaining arguments, as the embedded content within a **groff_ms**(7) **.XS** . . . **.XE** construct.

CONTROL REGISTERS

Although **groff_ms**(7) *does* track the progression of heading levels, as assigned by the **NH** macro, the tracked value is not exposed publicly. While public access to this tracked value may not be particularly useful, in any conventional deployment of **groff_ms**(7), it may be found useful in a user-defined replacement for the **XH–UPDATE–TOC** macro; thus, **groff_ms** augments the **groff_ms**(7) definition of the **NH** macro, to assign the internally tracked value to a register named **.NH**.

FILES

/usr/local/share/groff/site-tmac/spdf.tmac
Provides the implementation of the **groff_mspdf** bindings for conjoined use of **groff_ms**(7) and **groff_pdfmark**(7).

/usr/local/share/groff/site-tmac/s.tmac
Provides the implementation of the underlying **groff_ms**(7) macro package.

/usr/local/share/groff/site-tmac/pdfmark.tmac
Provides the implementation of the underlying **groff_pdfmark**(7) macro package.

CAVEATS AND BUGS

The **.NH** register is named in accordance with the **groff**(7) convention that an initial dot designates it as exhibiting the read-only property; however, since it is not implemented as a built-in register, **groff**(7) provides no mechanism for enforcing this. Thus, users are cautioned that setting it to any value, other than that assigned by the **NH** macro, may produce unexpected results.

EXAMPLES

The default implementation of the **XH–UPDATE–TOC** macro, which serves as the common table of contents collector for *both* **XH**, and **XN**, is:

```
.de XH-UPDATE-TOC
.\" .XH-UPDATE-TOC <outline-level> [<section-number>] <text>
.\"
.   XS            \" use conventional s.tmac TOC data collection
.      shift      \" ignore <outline-level> in rudimentary style
.      nop \&\\$* \" capture <section-number> and <text>
.   XE
..
```

This relies on the **groff_ms**(7) built-in table of contents data collector, but offers virtually no capability for control of the layout of the table of contents, when it is ultimately printed. Thus, users may wish to replace this default implementation, to achieve a more sophisticated layout; for example, (for headings set *exclusively* by **XN**):

```
.ds XNVS1 0.50v  \" leading for top level
.ds XNVS2 0.15v  \" leading at nesting level increment
.ds XNVS3 0.30v  \" leading following nested group
.
.de XH-UPDATE-TOC
.   XS
.      if r tc*hl \{\
.            \" Compute additional leading
.            \" at <outline-level> change
.            \"
.            ie \\$1>1 \{\
.                  ie \\$1>\\n[tc*hl] .sp \\*[XNVS2]
.                  el .if \\n[tc*hl]>\\$1 .sp \\*[XNVS3]
.               \}
.            el .sp \\*[XNVS1]
.         \}
.
.      \" Record <outline-level> of this entry,
.      \" for comparison with the next
.      \"
.      ie \\$1 .nr tc*hl \\$1
.      el .nr tc*hl 1
.
.      \" Set indentation, and insert
.      \" <section-number> for this entry
.      \"
.      nop \h'\\n[tc*hl]-1m'\\$2\c
.
.      \" Append <heading-text> for this entry
.      \"
.      shift 2
.      nop \h'1.5n'\\$*\h'0.5n'
.   XE
..
```

The preceding example is suitable *only* for use when *all* headings, which are to be incorporated into the table of contents, are specified using the **XN** macro, on account of its separate handling of its "`\\$2`" argument, (which is *always* assumed to represent a section number, as generated by a prior call of the **NH** macro). Since no such <*section-number*> argument is ever specified, when **XH–UPDATE–TOC** is called from within **XH**, this example might be modified, for *exclusive* use with headings specified using the **XH** macro, by replacing the statements:

```
.       \" Set indentation, and insert
.       \" <section-number> for this entry
.       \"
.       nop \h'\\n[tc*hl]-1m'\\$2\c
.
.       \" Append <heading-text> for this entry
.       \"
.       shift 2
.       nop \h'1.5n'\\$*\h'0.5n'
```

with a simplified alternative, such as:

```
.       \" Set indentation, and append aggregate
.       \" <heading-text> for this entry
.       \"
.       shift
.       nop \h'\\n[tc*hl]-1m'\\$*\h'0.5n'
```

As may be seen, from the foregoing examples, when all table of contents entries are specified using *either* **XH** *alone*, *or* **XN** *alone*, customization of the **XH–UPDATE–TOC** callback macro, to achieve a more structured table of contents layout, is relatively straightforward. Conversely, any customization which is required to accommodate a combination of table of contents entries, some of which may be specified by **XH**, and some by **XN**, will require a more complex implementation. The specific details of how the layout is to be adjusted, dependent on whether the entry is specified using **XH**, or using **XN**, is the prerogative of the document author, or of the callback macro implementor; usually, it will also be necessary to redefine the **XH–INIT**, and the **XN–INIT** callback macros, to record the calling context:

```
.de XH-INIT
.   nr XH-UPDATE-MODE 0
..
.de XN-INIT
.   nr XH-UPDATE-MODE 1
..
```

and then test the value of **XH–UPDATE–MODE**, within the redefined **XH–UPDATE–TOC** callback macro, as predicate for the choice of an applicable layout model.

AUTHORS

The **groff_mspdf** macros are provided by the auxiliary *groff-pdfmark* package, which was written by Keith Marshall <keith@address.hidden>; having formerly been distributed with *GNU roff*, this is now independently maintained at, and distributed from Keith's *groff-pdfmark* project hosting web-site <https://savannah.nongnu.org/projects/groff-pdfmark/>, whence the latest version is *always* available.

SEE ALSO

groff(1), **pdfroff**(1), **groff**(7), **groff_ms**(7), **groff_omit**(7), **groff_pdfhref**(7), **groff_pdfmark**(7), **groff_pdfnote**(7), **groff_sanitize**(7)

More comprehensive documentation, on the use of the **groff_mspdf** binding macros 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.