NAME

unroff-html − HTML 2.0 back-end for the programmable troff translator

SYNOPSIS

unroff [**−fhtml**] [**−m***package*] [*file* | *option...*]

OVERVIEW

When called with the **−fhtml** option, *unroff* loads the back-end for the Hypertext Markup Language (HTML) version 2.0. Please read **unroff**(1) first for an overview of the Scheme-based, programmable troff translator and for a description of the generic options that exist in addition to **−f** and **−m**. For information about extending and programming *unroff* also refer to the *Unroff Programmer's Manual*.

unroff is usually invoked with an additional **−m***package* option (such as **−ms** or **−man**) to load the translation rules for the troff macros and other elements defined by the macro package that is used to typeset the document. If no **−m** option is supplied, only the standard troff requests, special characters, escape sequences, etc. are recognized and translated to HTML by *unroff* as described in this manual.

OPTIONS

The following HTML-specific options can be specified in the command line after the generic options. See **unroff**(1) for a general description of keyword/value options and their types and for a list of options that are not specific to the target language.

title (string)
: The value to be used for the <title> element in HTML output files. This option may be ignored by the code implementing a specific macro set, e. g. when special rules are employed to derive the title from the contents of the troff input files. Whether or not this option is required also depends on the specific **−m** option used, but it may be omitted if no **−m** option is given.

document (string)
: The prefix used for the names of all output files. May be ignored depending on the macro package that has been selected.

mail-address (string)
: The caller's mail address; may be used for "mailto:" URLs, in particular for the "href" attribute of the <link> element that is usually generated.

tt-preformat (boolean)
: If 1, font changes to a font that is mapped to the <tt> element are honored inside non-filled text (as described below). The default is 0, i. e. the font changes will be recorded, but no corresponding HTML tags will be emitted for them.

handle-eqn (string)

handle-tbl (string)

handle-pic (string)
: These options specify how equations, tables, and pictures encountered in the troff input are processed. Possible values are "copy" to include the raw eqn, tbl, or pic commands as pre-formatted text, "text" to run the respective troff preprocessor (eqn, tbl, or pic) and include its output as pre-formatted text, or "gif" to convert the preprocessor output to a GIF image and include it in the HTML document as an inline image. The default is "text" for **handle-tbl**, "gif" for the other options. See DESCRIPTION below for more information.

eqn (string)

tbl (string)

pic (string)
: These options specify the programs to invoke as the eqn, tbl, and pic preprocessors. The defaults are site-dependent.

troff-to-text (string)

troff-to-gif (string)
The programs to invoke for converting the output of a troff preprocessor to plain text or to a GIF image. The default values are site-dependent. See DESCRIPTION below for more information on these options.

FILES

If no **−m** option is supplied, *unroff* reads the specified input files and sends the HTML document to standard output, unless the **document** option is given, in which case its value together with the suffix ".html" is used as the name of an output file. If no input files are specified, input is taken from standard input. The output is enclosed by the usual HTML boiler-plate (<html>, <head>, and <body> elements), a <title> element with the specified title (or the value of **document** if no title has been given, or a default title if both are omitted), a <link> element with rev= and href= attributes if **mail-address** has been set, and any pending end tags are generated on end of input.

Note that this is the default action that is performed in the rare case when no macro package name has been specified, i. e. when processing "bare" troff input. Somewhat different rules may apply when processing, for example, a group of UNIX manual pages (**−man**).

See **unroff**(1) for a list of Scheme files that are loaded on startup.

DESCRIPTION

OUTPUT TRANSLATIONS

The characters '<', '>', and '&' are replaced by the entities '<', '>', and '&' on output. In addition, the quote character is mapped to '"' where appropriate. New mappings can be added by means of the *defchar* Scheme primitive as explained in the Programmer's Manual.

COMMENTS

each troff comment is translated to a corresponding HTML tag followed by a newline; empty comments are ignored. Comments are also ignored when appearing inside a macro body.

ESCAPE SEQUENCES

The following is a list of troff escape sequences that are recognized and the HTML output generated for them. Any escape sequence that does not appear in the list expands to the character after the escape character, and a warning is printed in this case. New definitions can be added and the predefined mappings can be replaced by calling the *defescape* Scheme primitive in the user's initialization file, in a user-supplied Scheme file, in a document, or on a site-wide basis by modifying the file **scm/html/common.scm** in the installation directory.

```
\&        nothing
\-        -
\|        nothing
\^        nothing
\\        \
\'        '
\`        `
\"        rest of line as HTML comment tag
\%        nothing
\{        conditional input begin
\}        conditional input end
\*        contents of string
\space    space
\0        space
\c        nothing; eats following newline
\e        \
\s        nothing
\u        nothing, prints warning
\d        nothing, prints warning
```

\v	nothing, prints warning
\o	its argument, prints warning
\z	its argument, prints warning
\k	sets specified register to zero
\h	appropriate number of spaces for positive argument
\w	length of argument in units
\l	repeats specified character, or <hr>
\n	contents of number register
\f	see description of fonts below

SPECIAL CHARACTERS

The following special characters are mapped to their equivalent ISO-Latin 1 entities:

\(12	\(14	\(34	\(*b	\(*m	\(+-	\(:A
\(:O	\(:U	\(:a	\(:o	\(:u	\(A:	\(Cs
\(O:	\(Po	\(S1	\(S2	\(S3	\(U:	\(Ye
\(a:	\(bb	\(cd	\(co	\(ct	\(de	\(di
\(es	\(hy	\(mu	\(no	\(o:	\(r!	\(r?
\(rg	\(sc	\(ss	\(tm	\(u:		

Heuristics have to be used for the following special characters:

<table>
<tr><td>\(**</td><td>*</td><td></td></tr>
<tr><td>\(-></td><td>-></td><td></td></tr>
<tr><td>\(<-</td><td><-</td><td></td></tr>
<tr><td>\(<=</td><td><=</td><td></td></tr>
<tr><td>\(==</td><td>==</td><td></td></tr>
<tr><td>\(>=</td><td>>=</td><td></td></tr>
<tr><td>\(Fi</td><td>ffi</td><td></td></tr>
<tr><td>\(Fl</td><td>ffl</td><td></td></tr>
<tr><td>\(aa</td><td>’</td><td></td></tr>
<tr><td>\(ap</td><td>˜</td><td></td></tr>
<tr><td>\(br</td><td>|</td><td></td></tr>
<tr><td>\(bu</td><td>+</td><td>(prints a warning)</td></tr>
<tr><td>\(bv</td><td>|</td><td></td></tr>
<tr><td>\(ci</td><td>O</td><td></td></tr>
<tr><td>\(dd</td><td>***</td><td>(prints a warning)</td></tr>
<tr><td>\(dg</td><td>**</td><td>(prints a warning)</td></tr>
<tr><td>\(em</td><td>--</td><td></td></tr>
<tr><td>\(en</td><td>-</td><td></td></tr>
<tr><td>\(eq</td><td>=</td><td></td></tr>
<tr><td>\(ff</td><td>ff</td><td></td></tr>
<tr><td>\(fi</td><td>fi</td><td></td></tr>
<tr><td>\(fl</td><td>fl</td><td></td></tr>
<tr><td>\(fm</td><td>’</td><td></td></tr>
<tr><td>\(ga</td><td>‘</td><td></td></tr>
<tr><td>\(lh</td><td><=</td><td></td></tr>
<tr><td>\(lq</td><td>“</td><td></td></tr>
<tr><td>\(mi</td><td>-</td><td></td></tr>
<tr><td>\(or</td><td>|</td><td></td></tr>
<tr><td>\(pl</td><td>+</td><td></td></tr>
<tr><td>\(rh</td><td>=></td><td></td></tr>
<tr><td>\(rq</td><td>”</td><td></td></tr>
<tr><td>\(ru</td><td>_</td><td></td></tr>
<tr><td>\(sl</td><td>/</td><td></td></tr>
<tr><td>\(sq</td><td>o</td><td>(prints a warning)</td></tr>
<tr><td>\(ul</td><td>_</td><td></td></tr>
</table>

\(~= ~

A warning is printed to standard error output for any special character not mentioned in this section. To add new definitions, and to customize existing ones, the *defspecial* Scheme primitive can be used.

NON-FILLED TEXT

The **.nf** and **.fi** troff requests generate pairs of <pre> and </pre> tags. Nested requests are treated correctly, and currently active character formatting elements such as <i> (resulting from troff font changes) are temporarily disabled while the <pre> or </pre> is emitted. A warning is printed if a “tab” character is encountered within filled text.

FONTS

The ‘\f’ escape sequence and the requests **.ft** (change current font) and **.fp** (mount font at font position) are supported in the usual way, both with numeric font positions as well as font names and the special name ‘P’ to denote the previous font. The font position of the currently active font is available through the read-only number register ‘.f’. Initially, the font ‘R’ is mounted on font positions 1 and 4, font ‘I’ on font position 2, and font ‘B’ on position 3.

To map troff font names to HTML character formatting elements, the *define-font* Scheme procedure is called with the name of a troff font to be used in documents, and HTML start and end tags to be emitted when changing to this font, or when changing *from* this font to another font, respectively. Whether <tt> and </tt> is generated inside non-filled (pre-formatted) text for fixed-width fonts is controlled by the option **tt-preformat**. The following calls to *define-font* are evaluated on startup:

```
(define-font  "R"    ""    "")
(define-font  "I"    '<i>  '</i>)
(define-font  "B"    '<b>  '</b>)
(define-font  "C"    '<tt> '</tt>)
(define-font  "CW"   '<tt> '</tt>)
(define-font  "CO"   '<i>  '</i>)    ; kludge for Courier-Oblique
```

Site administrators may add definitions here for fonts used at their site. Users can define mappings for new fonts by placing corresponding definitions in their documents or document-specific Scheme files.

OTHER TROFF REQUESTS

The **.br** request generates a
 tag.

.sp requires a positive argument and is mapped to the appropriate number of <p> tags (or newline characters inside non-filled/pre-formatted text). Likewise, the request **.ti**, when called with a positive indent, produces a
 followed by the appropriate number of non-breakable spaces.

The **.tl** requests justs emits the title parts delimited by spaces. It is impossible to preserve the meaning of this request in HTML 2.0.

The horizontal line drawing escape sequence ‘\l’ just repeats the specified character (or underline as default) to draw a line. If the given length looks like it could be the line length (that is, if it exceeds a certain value), a <hr> tag is produced instead. Example:

\l’5c\&-’
\l’60’

The first of these two requests would produce a line of 20 dashes, while the second request would generate a <hr> tag (the ’\&’ is required because the dash could be interpreted as a continuation of the numeric expression).

Centering (**.ce**) is simulated by producing a
 at the end of each line, as this functionality is not supported by HTML 2.0.

The following requests are silently ignored; as the corresponding functions cannot be expressed in HTML 2.0 or are controlled by the client. Ignoring these requests most likely does no harm.

.ad	.bp	.ch	.fl	.hw	.hy	.lg
.na	.ne	.nh	.ns	.pl	.ps	.rs
.vs	.wh					

All troff requests not mentioned in this section by default cause a warning message to be printed to standard error output, except for these basic requests which have their usual semantics:

.am	.as	.de	.ds	.ec	.el	.ie
.if	.ig	.nr	.rm	.rr	.so	.tm

The *defrequest* Scheme primitive is used to associate an event handling procedure with a request as documented in the Programmer's Manual.

END OF SENTENCE

The sequence "<tt>space</tt>" is produced at the end of each sentence to provide additional space, except inside non-filled text. A sentence is defined a sequence of characters followed by a period, a question mark, or an exclamation mark, followed by a newline. The usual convention to suppress end-of-sentence recognition by adding the escape sequence '\&' is correctly implemented by *unroff*. To change the end-of-sentence function, the *sentence-event* can be redefined from within Scheme code as described in the Programmer's Manual.

SCALE INDICATORS

As the notions of vertical spacing, character width, device resolution, etc. do not exist in HTML, the scaling for the usual troff scale indicators is defined once on startup and then remains constant. For simplicity, the scaling usually employed by **nroff**(1) is taken.

EQUATIONS, TABLES, PICTURES

Interpretation of embedded eqn, tbl, and pic preprocessor input is controlled by the options **handle-eqn**, **handle-tbl**, and **handle-pic** (see OPTIONS above). These options affect the input lines from a starting **.EQ**, **.TS**, or **.PS** request up to and including the matching **.EN**, **.TE**, or **.PE** request, as well as text surrounded by the current eqn inline equation delimiters. Each of the options can have one the following values:

copy The preprocessor input (including the enclosing requests) is placed inside <pre> and </pre>. If assigned to the option **handle-eqn**, inline equations are rendered in the font currently mounted on font position 2.

text The input is sent to the respective preprocessor (as specified by the options **eqn**, **tbl**, or **pic**), and its result is piped to the shell command referred to by the option **troff-to-text**, which typically involves a call to **nroff**(1) or an equivalent command. As with "copy", the result is then placed inside <pre> and </pre>, unless the source is an inline equation.

The value of **troff-to-text** is filtered through a call to the *substitute* Scheme primitive with the name of an output file as its argument; this file name can be referenced from within the option's value by the substitute specifier "%1%" (see the Programmer's Manual for a description of *substitute* and a list of substitute specifiers). Here is a typical value for the **troff-to-text** option:

"groff −Tascii | col −b | sed '/^[\t]*$/d' > %1%"

gif Input lines are preprocessed as described under "text", and the result is piped to the shell command named by the option **troff-to-gif**. The latter is subject to a call to *substitute* with the name of a temporary file (which may be used to store intermediate PostScript output) and the name of the output file where the resulting GIF image must be stored. The entire preprocessor input is replaced by an <img> element with a reference to the GIF file and a suitable "alt=" attribute. Unless processing an inline equation, the <img> element is surrounded by <p> tags.

The names of the files containing the GIF images are generated from the value of the **document** option, a sequence number, and the suffix ".gif". Therefore, the **document** option must have been set when using the "gif" method, otherwise a warning is printed and the preprocessor input is skipped.

In any case, the output of a call to eqn is ignored if the input consists of calls to "delim" or "define" and empty lines exclusively. When processing eqn input, calls to "delim" are intercepted by *unroff* to record changes of the inline equation delimiters.

HYPERTEXT LINKS

The facilities for embedding arbitrary hypertext links in troff documents are still experimental in this version of *unroff* and thus are likely to change in future releases. To use them, mention the file name “hyper.scm” in the command line before any troff source files. At the beginning of the first troff file, source the file “tmac.hyper” from the directory “doc” like this:

```
.if !\n(.U .so tmac.hyper
```

The request **.Hr** can then be used to create a hypertext link. Its usage is:

```
.Hr -url      URL   anchor-text [suffix]
.Hr -symbolic label anchor-text [suffix]
.Hr troff-text
```

The first two forms are recognized by *unroff* and the third form is recognized by troff. The first form is used for links pointing to external resources, and the second one is used for forward or backward links referencing anchors defined in a file belonging to the same document. An anchor is placed in the document by calling the request **.Ha**:

```
.Ha label anchor-text
```

The label specified in a call to **.Ha** can then be used in calls to **.Hr -symbolic**. All symbolic references must have been resolved at the end of the document. The “anchor-text” is placed between the tags <a> and ; “suffix” is appended to the closing if present. “troff-text” is just formatted in the normal way. Quotes must be used if any of the arguments contains spaces.

Use of the hypertext facilities is demonstrated by the troff source of the Programmer’s Manual that is included in the *unroff* distribution.

SCHEME PROCEDURES

The following Scheme procedures, macros, and variables are defined by the HTML 2.0 back-end and can be used from within user-supplied Scheme code:

(*define-font name start-tag end-tag*)
: Associates a HTML start tag and end tag (symbols) with a troff font name (string) as explained under FONTS above. The font name can then be used in **.fp**, **.ft**, and ‘\f’ requests.

(*reset-font*)
: Resets both the current and previous font to the font mounted on position 1.

current-font

previous-font
: These variables hold the current and previous font as (integer) font positions.

(*with-font-preserved . body*)
: This macro can be used to temporarily change to font “R”, evaluate *body*, and revert to the font that has been active when the form was entered. The macro returns a string that can be output using the primitive *emit* or returned from an event procedure.

(*preform enable?*)
: If the argument is #t, pre-formatted text is enabled, otherwise disabled.

preform?
: This boolean variable holds #t if pre-formatted text is enabled, #f otherwise.

(*with-preform-preserved . body*)
: A macro that can be used to temporarily disable pre-formatted text, evaluate *body*, and then re-enable it if appropriate. The macro expands to a string that must be output or returned from an event procedure.

(*parse-unquote string*)
: Temporarily establishes an output translation to map the quote character to “"”, applies *parse* (explained in the Programmer’s Manual) to its argument, and returns the result.

(*center n*)
: Centers the next *n* input lines (see description of **.ce** under TROFF REQUESTS above). If *n* is zero, centering is stopped.

nbsp A Scheme variable that holds a string interpreted as a non-breaking space by HTML clients.

SEE ALSO

unroff(1), **unroff-html-man**(1), **unroff-html-ms**(1);
troff(1), **nroff**(1), **groff**(1), **eqn**(1), **tbl**(1), **pic**(1).

Unroff Programmer's Manual.

http://www.informatik.uni-bremen.de/˜net/unroff

Berners-Lee, Connolly, et al., HyperText Markup Language Specification—2.0, Internet Draft, Internet Engineering Task Force.

BUGS

The '\space' escape sequence should be mapped to the   entity (non-breaking space), but this entity is not supported by a number of HTML clients.

Only the font positions 1 to 9 can currently be used. There should be no limit.

The extra space generated for end of sentence should be configurable.

Underlining should be supported.