WoMan

Browse Unix Manual Pages “Wo (without) Man”
Software Version 0.54 (beta)

Francis J. Wright

School of Mathematical Sciences
Queen Mary and Westfield College
(University of London)
Mile End Road, London E1 4NS, UK

F.J.Wright@qmw.ac.uk

http://centaur.maths.qmw.ac.uk/

Manual Last Updated Time-stamp: <2002-12-04 09:12:13 pavel>

Table of Contents

1 Introduction

This version of WoMan should run with GNU Emacs 20.3 or later on any platform. It has not been tested, and may not run, with any other version of Emacs. It was developed primarily on various versions of Microsoft Windows, but has also been tested on MS-DOS, and various versions of UNIX and GNU/Linux.

WoMan is distributed with GNU Emacs 21, and the current source code and documentation files are available from my web server (`http://centaur.maths.qmw.ac.uk/Emacs/WoMan/`).

WoMan implements a subset of the formatting performed by the Emacs `man` (or `manual-entry`) command to format a Unix-style *manual page* (usually abbreviated to *man page*) for display, but without calling any external programs. It is intended to emulate the whole of the `ROFF -man` macro package, plus those `ROFF` requests (see Chapter 2 [Background], page 1) that are most commonly used in man pages. However, the emulation is modified to include the reformatting done by the Emacs `man` command. No hyphenation is performed.

Advantages
: Much more direct, does not require any external programs. Supports completion on man page names.

Disadvantages
: Not a complete emulation. Currently no support for `eqn` or `tbl`. Slightly slower for large man pages (but usually faster for small- and medium-size pages).

This browser works quite well on simple well-written man files. It works less well on idiosyncratic files that "break the rules" or use the more obscure `ROFF` requests directly. Current test results are available in the file '`woman.status`' (`http://centaur.maths.qmw.ac.uk/Emacs/WoMan/files/woman.status`).

WoMan supports the use of compressed man files via `auto-compression-mode` by turning it on if necessary. But you may need to adjust the user option `woman-file-compression-regexp`. See Section 6.1 [Interface Options], page 11.

Brief help on the WoMan interactive commands and user options, all of which begin with the prefix `woman-` (or occasionally `WoMan-`), is available most easily by loading WoMan and then either running the command `woman-mini-help` or selecting the WoMan menu option '`Mini Help`'.

WoMan is (of course) still under development! Please let me know (`F.J.Wright@qmw.ac.uk`) what doesn't work—I am adding and improving functionality as testing shows that it is necessary. Guidance on reporting bugs is given below. See Chapter 9 [Reporting Bugs], page 16.

2 Background

WoMan is a browser for traditional Unix-style manual page documentation. Each such document is conventionally referred to as a *manual page*, or *man page* for short, even though some are very much longer than one page. A man page is a document written using the Unix "man" macros, which are themselves written in the NROFF/TROFF text processing markup language. `NROFF` and `TROFF` are text processors originally written for the UNIX operating system by Joseph F. Ossanna at Bell Laboratories, Murray Hill, New Jersey, USA. They are closely related, and except in the few cases where the distinction between them is important I will refer to them both ambiguously as *ROFF*.

`ROFF` markup consists of *requests* and *escape sequences*. A request occupies a complete line and begins with either a period or a single forward quote. An escape sequences is embedded within the input text and begins (by default) with a backslash. The original man macro package defines 20 new `ROFF` requests implemented as macros, which were considered to be sufficient for writing man pages. But whilst in principle man pages use only the man macros, in practice a significant number use many other `ROFF` requests.

The distinction between `TROFF` and `NROFF` is that `TROFF` was designed to drive a phototypesetter whereas `NROFF` was designed to produce essentially ASCII output for a character-based device similar to a teletypewriter (usually abbreviated to "teletype" or "tty"). Hence, `TROFF` supports much finer control over output positioning than does `NROFF` and can be seen as a forerunner of TeX. Traditionally, man pages are either formatted by `TROFF` for typesetting or by `NROFF` for printing on a character printer or displaying on a screen. Of course, over the last 25 years or so, the distinction between typeset output on paper and characters on a screen has become blurred by the fact that most screens now support bit-mapped displays, so that any information that can be printed can also be rendered on screen, the only difference being the resolution.

Nevertheless, Unix-style manual page documentation is still normally browsed on screen by running a program called `man`. This program looks in a predefined set of directories for the man page matching a specified topic, then either formats the source file by running `NROFF` or recovers a pre-formatted file, and displays it via a pager such as `more`. `NROFF` normally formats for a printer, so it paginates the output, numbers the pages, etc., most of which is irrelevant when the document is browsed as a continuous scrollable document on screen. The only concession to on-screen browsing normally implemented by the `man` program is to squeeze consecutive blank lines into a single blank line.

For some time, Emacs has offered an improved interface for browsing man pages in the form of the Emacs `man` (or `manual-entry`) command, see section "Documentation Commands" in *GNU Emacs Manual*. This command runs `man` as described above, perhaps in the background, and then post-processes the output to remove much of the `NROFF` pagination such as page headers and footers, and places the result into an Emacs buffer. It puts this buffer into a special major mode, which is tailored for man page browsing, and provides a number of useful navigation commands, support for following references, etc. It provides some support for special display faces (fonts), but no special menu or mouse support. The Emacs man package appears to have been developed over about 10 years, from the late 1980s to the late 1990s.

There is considerable inefficiency in having `NROFF` paginate a document and then removing most of the pagination!

WoMan is an Emacs Lisp library that provides an emulation of the functionality of the Emacs `man` command, the main difference being that WoMan does not use any external programs. The only situation in which WoMan might use an external program is when the source file is compressed, when WoMan will use the standard Emacs automatic decompression facility, which does call an external program.

I began developing WoMan in the Spring of 1997 and the first version was released in May 1997. The original motivation for WoMan was the fact that many GNU and Unix programs are ported to other platforms and come with Unix-style manual page documentation. This may be difficult to read because ports of the Unix-style `man` program can be a little awkward to set up. I decided that it should not be too hard to emulate the 20 `man` macros directly, without treating them as macros and largely ignoring the underlying `ROFF` requests, given the text processing capabilities of Emacs. This proved to be essentially true, and it did not take a great deal of work to be able to format simple man pages acceptably.

One problem arose with the significant number of man pages that use `ROFF` requests in addition to the `man` macros, and since releasing the first version of WoMan I have been continually extending it to support more `ROFF` requests. WoMan can now format a significant proportion of the man pages that I have tested, either well or at least readably. However, I have added capabilities partly by making additional passes through the document, a design that is fundamentally flawed. This can only be solved by a major re-design of WoMan to handle the major formatting within a single recursive pass, rather than the present multiple passes without any significant recursion. There are some `ROFF` requests that cannot be handled satisfactorily within the present design. Some of these are currently handled by kludges that "usually more or less work."

The principle advantage of WoMan is that it does not require `man`, and indeed the name WoMan is a contraction of "without man." But it has other advantages. It does not paginate the document, so it does not need to un-paginate it again, thereby saving time. It could take full advantage of the display capabilities available to it, and I hope to develop WoMan to take advantage of developments in Emacs itself. At present, WoMan uses several display faces to support bold and italic text, to indicate other fonts, etc. The default faces are also coloured, but the choice of faces is customizable. WoMan provides menu support for navigation and mouse support for following references, in addition to the navigation facilities provided by `man` mode. WoMan has (this) texinfo documentation!

WoMan *does not* replace `man`, although it does use a number of the facilities implemented in the Emacs `man` library. WoMan and man can happily co-exist, which is very useful for comparison and debugging purposes. The only way in which WoMan affects `man` is that it adds a timer to indicate how long `man` has taken to format a man page. The timing is as compatible as possible with the timing built into WoMan, for as fair a comparison as possible. The time comparison seems to depend on the details of the platform, the version of `man` in use, etc, but times are similar and WoMan is never significantly slower than `man`. This is despite the fact that WoMan is running byte code whereas most of the formatting done by `man` uses machine code, and is a testimony to the quality of the Emacs Lisp system.

`NROFF` simulates non-ASCII characters by using one or more ASCII characters. WoMan should be able to do much better than this. I have recently begun to add support for WoMan to use more of the characters in its default font and to use a symbol font, and it is an aspect that I intend to develop further in the near future. It should be possible to move WoMan from an emulation of `NROFF` to an emulation of `TROFF` as GNU Emacs moves to providing bit-mapped display facilities.

3 Installation and Setup

No installation is necessary if you just want to run the version of WoMan distributed with GNU Emacs 21 or later, although some additional setup may still be desirable.

If you are installing '`woman.el`', either to update the version distributed with GNU Emacs or because WoMan was not distributed with your version of Emacs, then you need to put the file in a directory in your Emacs load path and byte compile it. A good directory to use is the '`site-lisp`' directory in your Emacs file tree, e.g. '`/usr/local/share/emacs/`*version*`/site-lisp/`' (where *version* is your Emacs version), provided you have write access to it. If you use a directory that is not included by default in your Emacs load path then you need to add something like this to your '`.emacs`' initialisation file:

```
(add-to-list 'load-path "my-lisp")
```

where '`my-lisp`' is the pathname of the directory. See section "The Init File ~/.emacs" in *The Emacs Editor*, for further details on customizing Emacs in general.

You can byte-compile the file by using the Emacs command `byte-compile-file` or by opening the directory containing the file, putting point on it and pressing the key *B*. (In fact, if the file is compiled then it is only the compiled file that needs to be in the Emacs load path, but leaving the source file there will do no harm.)

Setup

Setup that is either necessary or desirable consists of adding a small amount of Emacs Lisp code to your '`.emacs`' initialisation file. It may be necessary (or at least convenient) to make WoMan autoload (if you are not running GNU Emacs 21 or later) and to set the search path used by the `woman` interface. You may also find it convenient to make various WoMan menu and key bindings available and to make WoMan customizable even before WoMan has been loaded.

It is possible to run WoMan from a command line (from outside or even from inside Emacs) by suitably configuring your command interpreter.

3.1 Autoloading

If you are not running GNU Emacs 21 or later then you are recommended to add these autoloads to your '.emacs' file:

```
(autoload 'woman "woman"
  "Decode and browse a Unix man page." t)
(autoload 'woman-find-file "woman"
  "Find, decode and browse a specific Unix man-page file." t)
(autoload 'woman-dired-find-file "woman"
  "In dired, run the WoMan man-page browser on this file." t)
```

(In GNU Emacs 21 and later these autoloads are predefined.)

3.2 Search Path

The next step is necessary if you want to use the friendliest WoMan interface, which is recommended in general. If the MANPATH environment variable is set then WoMan will use it; alternatively (or additionally), if your platform uses a man configuration file (as do many versions of Linux) then WoMan will use it, provided it can find it. (This may need configuration. See Section 6.1 [Interface Options], page 11.) If these mechanisms correctly define the search path for man pages then no further action is required.

Otherwise you may need to customize the user option woman-manpath, and you may also want to customize the user option woman-path. See Chapter 6 [Customization], page 11. Now you can execute the extended command woman and enter or select a manual topic using completion, and if necessary select a filename, again using completion. By default, WoMan suggests the word nearest to point in the current buffer as the topic.

3.3 Preloading Menu and Key Bindings

Once WoMan is loaded it adds an item to the 'Help' menu and defines one or more keys in dired mode to run WoMan on the current file. If you would like these facilities always to be available, even before WoMan is loaded, then add the following to your '.emacs' file:

```
(define-key-after menu-bar-manuals-menu [woman]
  '(menu-item "Read Man Page (WoMan)..." woman
            :help "Man-page documentation Without Man") t)

(add-hook 'dired-mode-hook
          (lambda ()
            (define-key dired-mode-map "W" 'woman-dired-find-file)))
```

(By default, WoMan will automatically define the dired keys *W* and *w* when it loads, but only if they are not already defined. This behaviour is controlled by the user option woman-dired-keys. Note that the dired-x (dired extra) package binds dired-copy-filename-as-kill to the key *w*, although *W* appears to be unused. The dired-x package will over-write the WoMan binding for *w*, whereas (by default) WoMan will not overwrite the dired-x binding.)

3.4 Preloading Customization

WoMan supports the GNU Emacs 20+ customization facility, and puts a customization group called `WoMan` in the `Help` group under the top-level `Emacs` group. In order to be able to customize WoMan without first loading it, add the following to your '`.emacs`' file:

```
(defgroup woman nil
  "Browse UNIX manual pages ‘wo (without) man’."
  :tag "WoMan" :group 'help :load "woman")
```

3.5 Command Line Access

If you really want to square the man-woman circle then you can! If you run the GNU command interpreter `bash` then you might care to define the following `bash` function in your `bash` initialisation file '`.bashrc`':

```
man() { gnudoit -q '(raise-frame (selected-frame)) (woman' \"$1\" ')' ; }
```

If you use a Microsoft command interpreter ('`command.com`' or '`cmd.exe`') then you can create a file called '`man.bat`' somewhere in your path containing the two lines:

```
@echo off
gnudoit -q (raise-frame (selected-frame)) (woman \"%1\")
```

and then (e.g. from a command prompt or the '`Run...`' option in the Windows '`Start`' menu) just execute

```
man man_page_name
```

(Of course, if you already have a `man` command installed then you could call these commands `woman` instead of `man`.)

The above examples assume that you have the `gnuserv` Emacs client-server package installed (which I recommend). It would be possible to do something similar by calling Emacs directly, but that is less satisfactory, because you are likely to end up with multiple copies of Emacs running, which is generally inelegant, inefficient and inconvenient. If you run a different command interpreter then something similar to the above suggestions should be possible.

4 Finding and Formatting Man Pages

WoMan provides three user interfaces for finding and formatting man pages:

- a topic interface similar to that provided by the standard Emacs `man` command;
- a family of filename interfaces analogous to the standard Emacs `view-file` command;
- an automatic interface that detects the file type from its contents. (This is currently neither well tested, well supported nor recommended!)

The topic and filename interfaces support completion in the usual way.

The topic interface is generally the most convenient for regular use, although it may require some special setup, especially if your machine does not already have a conventional `man` installation (which WoMan tries to detect).

The simplest filename interface command `woman-find-file` can always be used with no setup at all (provided WoMan is installed and loaded or set up to autoload).

The automatic interface always requires special setup.

Case-Dependence of Filenames

By default, WoMan ignores case in file pathnames only when it seems appropriate. Microsoft Windows users who want complete case independence should set the special NTEmacs variable `w32-downcase-file-names` to `t` and use all lower case when setting WoMan file paths.

4.1 Topic Interface

The topic interface is accessed principally via the command `woman`. The same command can be accessed via the menu item '`Help->Manuals->Read Man Page (WoMan)...`' either once WoMan has been loaded or if it is set up specially. See Chapter 3 [Installation and Setup], page 3. The command reads a manual topic in the minibuffer, which can be the *basename* of a man file anywhere in the man file structure. The "basename" in this context means the filename without any directory component and without any extension or suffix components that relate to the file type. So, for example, if there is a compressed source file in Chapter 5 of the UNIX Programmer's Manual with the full pathname '`/usr/local/man/man5/man.conf.5.gz`' then the topic is `man.conf`. Provided WoMan is configured correctly, this topic will appear among the completions offered by `woman`. If more than one file has the same topic name then WoMan will prompt for which file to format. Completion of topics is case insensitive.

Clearly, `woman` has to know where to look for man files and there are two customizable user options that store this information: `woman-manpath` and `woman-path`. See Section 6.1 [Interface Options], page 11. If `woman-manpath` is not set explicitly then WoMan tries to pick up the information that would be used by the `man` command, as follows. If the environment variable `MANPATH` is set, which seems to be the standard mechanism under UNIX, then WoMan parses that. Otherwise, if WoMan can find a configuration file named (by default) '`man.conf`' (or something very similar), which seems to be the standard mechanism under GNU/Linux, then it parses that. To be precise, "something very similar" means having two name components separated by a dot and respectively containing '`man`' and beginning with '`conf`', e.g. '`manual.configuration`'. The search path and/or precise full path name for this file are set by the value of the customizable user option `woman-man.conf-path`. If all else fails, WoMan uses a plausible default man search path.

If the above default configuration does not work correctly for any reason then simply customize the value of `woman-manpath`. To access man files that are not in a conventional man file hierarchy, customize the value of `woman-path` to include the directories containing the files. In this way, `woman` can access manual files *anywhere* in the entire file system.

There are two differences between `woman-manpath` and `woman-path`. Firstly, the elements of `woman-manpath` must be directories that contain *directories of* man files, whereas the elements of `woman-path` must be directories that contain man files *directly*. Secondly, the last directory component of each element of `woman-path` is treated as a regular (Emacs) match expression rather than a fixed name, which allows collections of related directories to be specified succinctly.

For topic completion to work, WoMan must build a list of all the manual files that it can access, which can be very slow, especially if a network is involved. For this reason, it caches various amounts of information, after which retrieving it from the cache is very fast. If the cache ever gets out of synchronism with reality, running the `woman` command with a prefix argument (e.g. *`C-u M-x woman`*) will force it to rebuild its cache. This is necessary only if the names or locations of any man files change; it is not necessary if only their contents change. It would always be necessary if such a change occurred whilst Emacs were running and after WoMan has been loaded. It may be necessary if such a change occurs between Emacs sessions and persistent caching is used, although WoMan can detect some changes that invalidate its cache and rebuild it automatically.

Customize the variable `woman-cache-filename` to save the cache between Emacs sessions. This is recommended only if the `woman` command is too slow the first time it is run in an Emacs session, while it builds its cache in main memory, which *may* be *very* slow. See Section 4.1.1 [The WoMan Topic Cache], page 6, for further details.

4.1.1 The WoMan Topic Cache

The amount of information that WoMan caches (in main memory and, optionally, saved to disc) is controlled by the user option `woman-cache-level`. There is a trade-off between the speed with which WoMan can find a file and the size of the cache, and the default setting gives a reasonable compromise.

The `woman` command always performs a certain amount of caching in main memory, but it can also write its cache to the filestore as a persistent cache under control of the user option `woman-cache-filename`. If persistent caching is turned on then WoMan re-loads its internal cache from the cache file almost instantaneously, so that there is never any perceptible start-up delay *except* when WoMan rebuilds its cache. Persistent caching is currently turned off by default. This is because users with persistent caching turned on may overlook the need to force WoMan to rebuild its cache the first time they run it after they have installed new man files; with persistent caching turned off, WoMan automatically rebuilds its cache every time it is run in a new Emacs session.

A prefix argument always causes the `woman` command (only) to rebuild its topic cache, and to re-save it to `woman-cache-filename` if this variable has a non-`nil` value. This is necessary if the *names* of any of the directories or files in the paths specified by `woman-manpath` or `woman-path` change. If WoMan user options that affect the cache are changed then WoMan will automatically update its cache file on disc (if one is in use) the next time it is run in a new Emacs session.

4.1.2 Using the "Word at Point" as a Topic Suggestion

By default, the `woman` command uses the word nearest to point in the current buffer as a suggestion for the topic to look up. The topic must be confirmed or edited in the minibuffer. This suggestion can be turned off, or `woman` can use the suggested topic without confirmation if possible, which is controlled by customizing the user option `woman-topic-at-point` to `nil` or `t` respectively. (Its default value is neither `nil` nor `t`, meaning ask for confirmation.)

The variable `woman-topic-at-point` can also be rebound locally (using `let`), which may be useful to provide special private key bindings, e.g. this key binding for *C-c w* runs WoMan on the topic at point without seeking confirmation:

```
(global-set-key "\C-cw"
                (lambda ()
                  (interactive)
                  (let ((woman-topic-at-point t))
                    (woman))))
```

4.2 Filename Interface

The commands in this family are completely independent of the topic interface, caching mechanism, etc.

The filename interface is accessed principally via the extended command `woman-find-file`, which is available without any configuration at all (provided WoMan is installed and loaded or set up to autoload). This command can be used to browse any accessible man file, regardless of its filename or location. If the file is compressed then automatic file decompression must already be turned on (e.g. see the '`Help->Options`' submenu)—it is turned on automatically only by the `woman` topic interface.

Once WoMan is loaded (or if specially set up), various additional commands in this family are available. In a dired buffer, the command `woman-dired-find-file` allows the file on the same line as point to be formatted and browsed by WoMan. It is bound to the key *W* in the dired mode map and added to the dired major mode menu. It may also be bound to *w*, unless this key is bound by another library, which it is by `dired-x`, for example. Because it is quite likely that other libraries will extend the capabilities of such a commonly used mode as dired, the precise key bindings added by WoMan to the dired mode map are controlled by the user option `woman-dired-keys`.

When a tar (Tape ARchive) file is visited in Emacs, it is opened in tar mode, which parses the tar file and shows a dired-like view of its contents. The WoMan command `woman-tar-extract-file` allows the file on the same line as point to be formatted and browsed by WoMan. It is bound to the key *w* in the tar mode map and added to the tar major mode menu.

The command `woman-reformat-last-file`, which is bound to the key *R* in WoMan mode and available on the major mode menu, reformats the last file formatted by WoMan. This may occasionally be useful if formatting parameters, such as the fill column, are changed, or perhaps if the buffer is somehow corrupted.

The command `woman-decode-buffer` can be used to decode and browse the current buffer if it is visiting a man file, although it is primarily used internally by WoMan.

4.3 Automatic Interface

Emacs provides an interface to detect automatically the format of a file and decode it when it is visited. It is used primarily by the facilities for editing rich (i.e. formatted) text, as a way to store formatting information transparently as ASCII markup. WoMan can in principle use this interface, but it must be configured explicitly.

This use of WoMan does not seem to be particularly advantageous, so it is not really supported. It originated during early experiments on how best to implement WoMan, before I implemented the current topic interface, and I subsequently stopped using it. I might revive it as a mechanism for storing pre-formatted WoMan files, somewhat analogous to the standard Unix `catman` facility. In the meantime, it exists for anyone who wants to experiment with it. Once it is set up it is simply a question of visiting the file and there is no WoMan-specific user interface!

To use it, put something like this in your '`.emacs`' file. [The call to `set-visited-file-name` is to avoid font-locking triggered by automatic major mode selection.]

```
(autoload 'woman-decode-region "woman")

(add-to-list 'format-alist
             '(man "Unix man-page source format" "\\.\\(TH\\|ig\\) "
                   woman-decode-region nil nil
                   (lambda (arg)
                     set-visited-file-name
                     (file-name-sans-extension buffer-file-name))))
```

5 Browsing Man Pages

Once a man page has been found and formatted, WoMan provides a browsing interface that is essentially the same as that provided by the standard Emacs `man` command (and much of the code is inherited from the `man` library, which WoMan currently requires). Many WoMan facilities can be accessed from the WoMan major mode menu as well as via key bindings, etc.

WoMan does not produce any page breaks or page numbers, and in fact does not paginate the man page at all, since this is not appropriate for continuous online browsing. It produces a document header line that is constructed from the standard man page header and footer. Apart from that, the appearance of the formatted man page should be almost identical to what would be produced by `man`, with consecutive blank lines squeezed to a single blank line.

5.1 Fonts and Faces

Fonts used by `ROFF` are handled by WoMan as faces, the details of which are customizable. See Section 6.3 [Faces], page 15. WoMan supports both the italic and bold fonts normally used in man pages, together with a single face to represent all unknown fonts (which are occasionally used in “non-standard” man pages, usually to represent a “typewriter” font) and a face to indicate additional symbols introduced by WoMan. This currently means the characters ^ and _ used to indicate super- and sub-scripts, which are not displayed well by WoMan.

5.2 Navigation

Man (and hence WoMan) mode can be thought of as a superset of view mode. The buffer cannot be edited, so keys that would normally self-insert are used for navigation. The WoMan key bindings are a minor modification of the `man` key bindings.

⟨SPC⟩ Scroll the man page up the window (`scroll-up`).

⟨DEL⟩ Scroll the man page down the window (`scroll-down`).

n Move point to the Nth next section—default 1 (`Man-next-section`).

p Move point to Nth previous section—default 1 (`Man-previous-section`).

g Move point to the specified section (`Man-goto-section`).

s Move point to the “SEE ALSO” section (`Man-goto-see-also-section`). Actually the section moved to is described by `Man-see-also-regexp`.

5.3 Following References

Man pages usually contain a “SEE ALSO” section containing references to other man pages. If these man pages are installed then WoMan can easily be directed to follow the reference, i.e. to find and format the man page. When the mouse is passed over a correctly formatted reference it is highlighted, in which case clicking the middle button *Mouse-2* will cause WoMan to follow the reference. Alternatively, when point is over such a reference the key ⟨RET⟩ will follow the reference.

Any word in the buffer can be used as a reference by clicking *Mouse-2* over it provided the Meta key is also used (although in general such a “reference” will not lead to a man page). Alternatively, the key *r* allows completion to be used to select a reference to follow, based on the word at point as default.

Mouse-2 Run WoMan with word under mouse as topic (`woman-mouse-2`). The word must be mouse-highlighted unless `woman-mouse-2` is used with the Meta key.

⟨RET⟩ Get the man page for the topic under (or nearest to) point (`man-follow`).

r Get one of the man pages referred to in the “SEE ALSO” section (`Man-follow-manual-reference`). Specify which reference to use; default is based on word at point.

5.4 Changing the Current Man Page

The man page currently being browsed by WoMan can be changed in several ways. The command `woman` can be invoked to format another man page, or the current WoMan buffer can be buried or killed. WoMan maintains a ring of formatted man pages, and it is possible to move forwards and backwards in this ring by moving to the next or previous man page. It is sometimes useful to reformat the current page, for example after the right margin (the wrap column) or some other formatting parameter has been changed.

Buffers formatted by Man and WoMan are completely unrelated, even though some of the commands to manipulate them are superficially the same (and share code).

m Run the command `man` to get a Un*x manual page and put it in a buffer. This command is the top-level command in the man package. It runs a Un*x command to retrieve and clean a man page in the background and places the results in a Man mode (man page browsing) buffer. If a man buffer already exists for this man page, it will display immediately. This works exactly the same if WoMan is loaded, except that the formatting time is displayed in the mini-buffer.

w Run the command `woman` exactly as if the extended command or menu item had been used.

q Bury the buffer containing the current man page (`Man-quit`), i.e. move it to the bottom of the buffer stack.

k Kill the buffer containing the current man page (`Man-kill`), i.e. delete it completely so that it can be retrieved only by formatting the page again.

M-p Find the previous WoMan buffer (`WoMan-previous-manpage`).

M-n Find the next WoMan buffer (`WoMan-next-manpage`).

R Call WoMan to reformat the last man page formatted by WoMan (`woman-reformat-last-file`), e.g. after changing the fill column.

5.5 Convenience Key Bindings

- Begin a negative numeric argument for the next command (`negative-argument`).

0 .. 9 Part of the numeric argument for the next command (`digit-argument`).

<
. Move point to the beginning of the buffer; leave mark at previous position (`beginning-of-buffer`).

> Move point to the end of the buffer; leave mark at previous position (`end-of-buffer`).

? Display documentation of current major mode and minor modes (`describe-mode`). The major mode description comes first, followed by the minor modes, each on a separate page.

5.6 Imenu Support; Contents Menu

The WoMan menu provides an option to make a contents menu for the current man page (using `imenu`). Alternatively, if you customize the option `woman-imenu` to `t` then WoMan will do it automatically for every man page. The menu title is set by the option `woman-imenu-title`, which is "CONTENTS" by default. The menu shows manual sections and subsections by default, but you can change this by customizing `woman-imenu-generic-expression`.

WoMan is configured not to replace spaces in an imenu `*Completion*` buffer. For further documentation on the use of imenu, such as menu sorting, see the source file '`imenu.el`', which is distributed with GNU Emacs.

6 Customization

All WoMan user options are customizable, and it is recommended to change them only via the standard Emacs customization facilities. WoMan defines a top-level customization group called `WoMan` under the parent group `Help`. The WoMan customization group is available only once WoMan has been loaded unless it is specially set up to be automatically available. See Section 3.4 [Preloading Customization], page 5. It can be accessed either via the standard Emacs facilities, e.g. via the '`Help->Customize`' submenu, or via the WoMan major mode menu.

The top-level WoMan group contains only a few general options and three subgroups. The hooks are provided only for special purposes that, for example, require code to be executed, and should be changed only via `Customization` or the function `add-hook`. Most customization should be possible via existing user options.

`woman-show-log`
: A boolean value that defaults to `nil`. If non-`nil` then show the `*WoMan-Log*` buffer if appropriate, i.e. if any warning messages are written to it. See Chapter 7 [The *WoMan-Log* Buffer], page 16.

`woman-pre-format-hook`
: A hook run immediately before formatting a buffer. It might, for example, be used for face customization. See Section 6.3 [Faces], page 15, however.

`woman-post-format-hook`
: A hook run immediately after formatting a buffer. It might, for example, be used for installing a dynamic menu using `imenu`. (However. in this case it is better to use the built-in WoMan `imenu` support. See Section 5.6 [Imenu Support; Contents Menu], page 10.)

Customization Subgroups

`WoMan Interface`
: These options control the process of locating the appropriate file to browse, and the appearance of the browsing interface.

`WoMan Formatting`
: These options control the layout that WoMan uses to format the man page.

`WoMan Faces`
: These options control the display faces that WoMan uses to format the man page.

6.1 Interface Options

These options control the process of locating the appropriate file to browse, and the appearance of the browsing interface.

`woman-man.conf-path`
: A list of strings representing directories to search and/or files to try for a man configuration file. The default is

```
("/etc" "/usr/local/lib")
```

[for GNU/Linux and Cygwin respectively.] A trailing separator ('`/`' for UNIX etc.) on directories is optional and the filename matched if a directory is specified is the first to match the regexp `man.*\.conf`. If the environment variable `MANPATH` is not set but a configuration file is found then it is parsed instead (or as well) to provide a default value for `woman-manpath`.

`woman-manpath`

A list of strings representing *directory trees* to search for Unix manual files. Each element should be the name of a directory that contains subdirectories of the form '`man?`', or more precisely subdirectories selected by the value of `woman-manpath-man-regexp`. Non-directory and unreadable files are ignored.

If not set then the environment variable `MANPATH` is used. If no such environment variable is found, the default list is determined by consulting the man configuration file if found. By default this is expected to be either '`/etc/man.config`' or '`/usr/local/lib/man.conf`', which is controlled by the user option `woman-man.conf-path`. An empty substring of `MANPATH` denotes the default list. Otherwise, the default value of this variable is

```
("/usr/man" "/usr/local/man")
```

Any environment variables (names of which must have the Unix-style form `$NAME`, e.g. `$HOME`, `$EMACSDATA`, `$EMACS_DIR`, regardless of platform) are evaluated first but each element must evaluate to a *single* directory name. Trailing '`/`'s are ignored. (Specific directories in `woman-path` are also searched.)

On Microsoft platforms I recommend including drive letters explicitly, e.g.

```
("C:/Cygwin/usr/man" "C:/usr/man" "C:/usr/local/man")
```

The `MANPATH` environment variable may be set using DOS semi-colon-separated or Unix-style colon-separated syntax (but not mixed).

`woman-manpath-man-regexp`

A regular expression to match man directories *under* the `woman-manpath` directories. These normally have names of the form '`man?`'. Its default value is `"[Mm][Aa][Nn]"`, which is case-insensitive mainly for the benefit of Microsoft platforms. Its purpose is to avoid directories such as '`cat?`', '`.`', '`..`', etc.

`woman-path`

A list of strings representing *specific directories* to search for Unix manual files. For example

```
("/emacs/etc")
```

These directories are searched in addition to the directory trees specified in `woman-manpath`. Each element should be a directory string or `nil`, which represents the current directory when the path is expanded and cached. However, the last component (only) of each directory string is treated as a regexp (Emacs, not shell) and the string is expanded into a list of matching directories. Non-directory and unreadable files are ignored. The default value on MS-DOS is

```
("$DJDIR/info" "$DJDIR/man/cat[1-9onlp]")
```

and on other platforms is `nil`.

Any environment variables (names of which must have the Unix-style form `$NAME`, e.g. `$HOME`, `$EMACSDATA`, `$EMACS_DIR`, regardless of platform) are evaluated first but each element must evaluate to a *single* directory name (regexp, see above). For example

```
("$EMACSDATA")
```

or equivalently

```
("$EMACS_DIR/etc")
```

Trailing '`/`'s are discarded. (The directory trees in `woman-manpath` are also searched.) On Microsoft platforms I recommend including drive letters explicitly.

`woman-cache-level`

A positive integer representing the level of topic caching:

1. cache only the topic and directory lists (uses minimal memory, but not recommended);
2. cache also the directories for each topic (faster, without using much more memory);
3. cache also the actual filenames for each topic (fastest, but uses twice as much memory).

The default value is currently 2, a good general compromise. If the `woman` command is slow to find files then try 3, which may be particularly beneficial with large remote-mounted man directories. Run the `woman` command with a prefix argument or delete the cache file `woman-cache-filename` for a change to take effect. (Values `<` 1 behave like 1; values `>` 3 behave like 3.)

`woman-cache-filename`
: Either a string representing the full pathname of the WoMan directory and topic cache file, or `nil`. It is used to save and restore the cache between Emacs sessions. This is especially useful with remote-mounted man page files! The default value of `nil` suppresses this action. The "standard" non-`nil` filename is '`~/.wmncach.el`'. Remember that a prefix argument forces the `woman` command to update and re-write the cache.

`woman-dired-keys`
: A list of `dired` mode keys to be defined to run WoMan on the current file, e.g. `("w" "W")` or any non-`nil` atom to automatically define *w* and *W* if they are unbound, or `nil` to do nothing. Default is `t`.

`woman-imenu-generic-expression`
: Imenu support for Sections and Subsections: an alist with elements of the form `(MENU-TITLE REGEXP INDEX)`—see the documentation for `imenu-generic-expression`. Default value is

```
((nil "\n\\([A-Z].*\\)" 1)  ; SECTION, but not TITLE
 ("*Subsections*" "^   \\([A-Z].*\\)" 1))
```

`woman-imenu`
: A boolean value that defaults to `nil`. If non-`nil` then WoMan adds a Contents menu to the menubar by calling `imenu-add-to-menubar`.

`woman-imenu-title`
: A string representing the title to use if WoMan adds a Contents menu to the menubar. Default is `"CONTENTS"`.

`woman-topic-at-point`
: A symbol, which may be either `t`, `nil` or `confirm`, that controls the use by `woman` of the "word at point" as a topic suggestion. If it is non-`nil` then the `woman` command uses the word at point as an initial topic suggestion when it reads a topic from the minibuffer; if it is `t` then `woman` uses the word at point *without interactive confirmation* if it exists as a topic. The value `confirm` means suggest a topic and ask for confirmation. The default value is that of `woman-topic-at-point-default`.

`woman-topic-at-point-default`
: A symbol, which may be either `t`, `nil` or `confirm`, representing the default value for `woman-topic-at-point`. The default value is `confirm`. [The variable `woman-topic-at-point` may be `let`-bound when `woman` is loaded, in which case its global value does not get defined. The function `woman-file-name` sets it to this value if it is unbound.]

`woman-uncompressed-file-regexp`
A regular match expression used to select man source files (ignoring any compression extension). The default value is `"\\.\\([0-9lmnt]\\w*\\)"` [which means a filename extension is required].

Do not change this unless you are sure you know what you are doing!

The SysV standard man pages use two character suffixes, and this is becoming more common in the GNU world. For example, the man pages in the `ncurses` package include '`toe.1m`', '`form.3x`', etc.

Note: an optional compression regexp will be appended, so this regexp *must not* end with any kind of string terminator such as `$` or `\\'`.

`woman-file-compression-regexp`
A regular match expression used to match compressed man file extensions for which decompressors are available and handled by auto-compression mode. It should begin with `\\.` and end with `\\'` and *must not* be optional. The default value is `"\\.\\(g?z\\|bz2\\)\\'"`, which matches the `gzip` and `bzip2` compression extensions.

Do not change this unless you are sure you know what you are doing!

[It should be compatible with the `car` of `jka-compr-file-name-handler-entry`, but that is unduly complicated, includes an inappropriate extension ('`.tgz`') and is not loaded by default!]

`woman-use-own-frame`
If non-`nil` then use a dedicated frame for displaying WoMan windows. This is useful only when WoMan is run under a window system such as X or Microsoft Windows that supports real multiple frames, in which case the default value is non-`nil`.

6.2 Formatting Options

These options control the layout that WoMan uses to format the man page.

`woman-fill-column`
An integer specifying the right margin for formatted text. Default is 65.

`woman-fill-frame`
A boolean value. If non-`nil` then most of the frame width is used, overriding the value of `woman-fill-column`. Default is `nil`.

`woman-default-indent`
An integer specifying the default prevailing indent for the `-man` macros. Default is 5. Set this variable to 7 to emulate GNU/Linux man formatting.

`woman-bold-headings`
A boolean value. If non-`nil` then embolden section and subsection headings. Default is `t`. [Heading emboldening is *not* standard `man` behaviour.]

`woman-ignore`
A boolean value. If non-`nil` then unrecognised requests etc. are ignored. Default is `t`. This gives the standard `ROFF` behaviour. If `nil` then they are left in the buffer, which may aid debugging.

`woman-preserve-ascii`
A boolean value. If non-`nil` then preserve ASCII characters in the WoMan buffer. Otherwise, non-ASCII characters (that display as ASCII) may remain, which is irrelevant unless the buffer is to be saved to a file. Default is `nil`.

`woman-emulation`
: WoMan emulation, currently either `NROFF` or `TROFF`. Default is `NROFF`. `TROFF` emulation is experimental and largely untested.

6.3 Faces

These options control the display faces that WoMan uses to format the man page.

`woman-fontify`
: A boolean value. If non-`nil` then WoMan assumes that face support is available. It defaults to a non-`nil` value if the display supports either colours or different fonts.

`woman-italic-face`
: Face for italic font in man pages. Default: italic, underlined, foreground red. This is overkill! `TROFF` uses just italic; `NROFF` uses just underline. You should probably select either italic or underline as you prefer, but not both, although italic and underline work together perfectly well!

`woman-bold-face`
: Face for bold font in man pages. Default: bold, foreground blue.

`woman-unknown-face`
: Face for all unknown fonts in man pages. Default: foreground brown. Brown is a good compromise: it is distinguishable from the default but not enough so as to make font errors look terrible. (Files that use non-standard fonts seem to do so badly or in idiosyncratic ways!)

`woman-addition-face`
: Face for all additions made by WoMan to man pages. Default: foreground orange.

6.4 Special symbols

This section currently applies *only* to Microsoft Windows.

WoMan provides partial experimental support for special symbols, initially only for MS-Windows and only for MS-Windows fonts. This includes both non-ASCII characters from the main text font and use of a separate symbol font. Later, support will be added for other font types (e.g. `bdf` fonts) and for the X Window System. In Emacs 20.7, the current support works partially under Windows 9x but may not work on any other platform.

`woman-use-extended-font`
: A boolean value. If non-`nil` then WoMan may use non-ASCII characters from the default font. Default is `t`.

`woman-use-symbol-font`
: A boolean value. If non-`nil` then WoMan may use the symbol font. Default is `nil`, mainly because it may change the line spacing (at least in NTEmacs 20).

`woman-symbol-font`
: A string describing the symbol font to use for special characters. It should be compatible with, and the same size as, the default text font. Under MS-Windows, the default is

```
"-*-Symbol-normal-r-*-*-*-*-96-96-p-*-ms-symbol"
```

7 The *WoMan-Log* Buffer

This is modelled on the Emacs byte-compiler. It logs all files formatted by WoMan and the time taken. If WoMan finds anything that it cannot handle then it writes a warning to this buffer. If the variable `woman-show-log` is non-`nil` (by default it is `nil`) then WoMan automatically displays this buffer. See Section 6.1 [Interface Options], page 11. Many WoMan warnings can be completely ignored, because they are reporting the fact that WoMan has ignored requests that it is correct for WoMan to ignore. In some future version this level of paranoia may be reduced, but not until WoMan is deemed more reliable. At present, all warnings should be treated with some suspicion. Uninterpreted escape sequences are also logged (in some cases).

By resetting the variable `woman-ignore` to `nil` (by default it is `t`), uninterpreted `ROFF` requests can optionally be left in the formatted buffer to indicate precisely where they occurred. See Section 6.1 [Interface Options], page 11.

8 Technical Details

Horizontal and vertical spacing and resolution

WoMan currently assumes 10 characters per inch horizontally, hence a horizontal resolution of 24 basic units, and 5 lines per inch vertically, hence a vertical resolution of 48 basic units. (`NROFF` uses 240 per inch.)

Vertical spacing and blank lines

The number of consecutive blank lines in the formatted buffer should be either 0 or 1. A blank line should leave a space like .sp 1. Current policy is to output vertical space only immediately before text is output.

9 Reporting Bugs

If WoMan fails completely, or formats a file incorrectly (i.e. obviously wrongly or significantly differently from `man`) or inelegantly, then please

a. check that you are running the latest version of ‘`woman.el`’ available from my web site (`http://centaur.maths.qmw.ac.uk/Emacs/WoMan/`), and
b. check that the problem is not already described in the file ‘`woman.status`’, also available from my web site (`http://centaur.maths.qmw.ac.uk/Emacs/WoMan/`).

If both of the above are true then please email me (`F.J.Wright@qmw.ac.uk`) the entry from the `*WoMan-Log*` buffer relating to the problem file, together with a brief description of the problem. Please indicate where you got the man source file from, but do not send it to me unless I ask you to! Thanks. (At present WoMan has no automated bug-reporting facility.)

10 Acknowledgements

For Heather, Kathryn and Madelyn, the women in my life (although they will probably never use it)!

I also thank the following for helpful suggestions, bug reports, code fragments, general interest, etc.:

Jari Aalto, `jari.aalto@cs.tpu.fi`
Dean Andrews, `dean@dra.com`
Juanma Barranquero, `barranquero@laley-actualidad.es`
Karl Berry, `kb@cs.umb.edu`
Jim Chapman, `jchapman@netcomuk.co.uk`
Frederic Corne, `frederic.corne@erli.fr`
Peter Craft, `craft@alacritech.com`
Charles Curley, `ccurley@trib.com`
Jim Davidson, `jdavidso@teknowledge.com`
Kevin D'Elia, `Kevin.DElia@mci.com`
John Fitch, `jpff@maths.bath.ac.uk`
Hans Frosch, `jwfrosch@rish.b17c.ingr.com`
Guy Gascoigne-Piggford, `ggp@informix.com`
Brian Gorka, `gorkab@sanchez.com`
Nicolai Henriksen, `nhe@lyngso-industri.dk`
Thomas Herchenroeder, `the@software-ag.de`
Alexander Hinds, `ahinds@thegrid.net`
Stefan Hornburg, `sth@hacon.de`
Theodore Jump, `tjump@cais.com`
Paul Kinnucan, `paulk@mathworks.com`
Jonas Linde, `jonas@init.se`
Andrew McRae, `andrewm@optimation.co.nz`
Howard Melman, `howard@silverstream.com`
Dennis Pixton, `dennis@math.binghamton.edu`
T. V. Raman, `raman@Adobe.com`
Bruce Ravel, `bruce.ravel@nist.gov`
Benjamin Riefenstahl, `benny@crocodial.de`
Kevin Ruland, `kruland@seistl.com`
Tom Schutter, `tom@platte.com`
Wei-Xue Shi, `wxshi@ma.neweb.ne.jp`
Fabio Somenzi, `fabio@joplin.colorado.edu`
Karel Sprenger, `ks@ic.uva.nl`
Chris Szurgot, `szurgot@itribe.net`
Paul A. Thompson, `pat@po.cwru.edu`
Arrigo Triulzi, `arrigo@maths.qmw.ac.uk`
Geoff Voelker, `voelker@cs.washington.edu`
Eli Zaretskii, `eliz@is.elta.co.il`

Command Index

B

D

E

M

N

S

W

Variable Index

Keystroke Index

Concept Index