

w3m MANUAL

Akinori Ito
aito@fw.ipsj.or.jp

Index

- [Introduction](#)
 - [Options](#)
 - [Document Colors](#)
 - [Functions and Key bindings](#)
 - [Mouse Operation](#)
 - [Local CGI scripts](#)
-

Introduction

w3m is a pager/text-based WWW browser. You can browse local documents and/or documents on the WWW using a terminal emulator.

Options

Command line usage is

```
w3m [options] [file|URL]
```

If you specify filenames/URLs on the command line, these documents are displayed. If you specify nothing, w3m will read a document from standard input and display it. If it doesn't find a document there either then normally w3m will terminate.

Options include:

+number

go to line num; only effective for num larger than the number of lines in the terminal

-t num

set tab width to *num* columns. No effect on stdout

-r

use caret notation to display special escape characters (such as ANSI escapes or nroff-style backspaces for bold and underlined characters) instead of processing them

-l num

number of lines preserved internally when receiving plain text from stdin (default 10,000)

-O charset

user defined character encoding of output data

-I charset

user defined character encoding of input data

-T type

explicit characterization of input data by MIME type. Without this option, document type is

determined from the extension of a file. If the determination fails, the document is regarded as text/plain. For example:

Read HTML document from standard input and display it

```
cat example.html | w3m -T text/html
```

Display HTML source

```
w3m -T text/plain example.html
```

-m

Display document using "Internet message mode". With this option, w3m determines document type from header information. This is useful when reading e-mail or Usenet news posts.

-v

with no other target defined, welcome users with a built-in page

-B

with no other target defined, use the bookmark page for startup

-bookmark *file*

use *file* instead of the default bookmark.html file

-M

monochrome display

-F

render frames

-s

squeeze multiple blank lines into one

-X

Upon exit, do not reinitialize the terminal.

-W

toggle wrapping mode in searches

-o *option=value*

modify one configuration item with an explicitly given value; without *option=value*, equivalent to *-show-option*

-cookie

use stored cookies and accept new ones

-no-cookie

neither use stored cookies nor accept new ones

-num

display each line's number

-dump

Read document specified by URL and dump page rendered as text into standard output. A width of 80 columns is used unless option `-cols` sets another value.

`-cols num`

with stdout as destination; HTML is rendered to lines of *num* characters

`-ppc num`

width of *num* pixels per character. Range of 4.0 to 32.0, default 8.0. Larger values will make tables narrower. (Implementation not verified)

`-dump_source`

dump the page's source code into stdout

`-dump_head`

dump response of a HEAD request for a URL into stdout

`-dump_both`

dump HEAD, and source code for a URL into stdout

`-dump_extra`

dump HEAD, source code, and extra information for a URL into stdout

`-post file`

use POST method to upload data defined in *file*. The syntax to be used is "var1=value1[&var2=value2]..."

`-header string`

append *string* to the HTTP(S) request. Expected to match the header syntax "Variable: Value"

`-no-proxy`

do not use proxy

`-no-graph`

do not use graphic characters for drawing HTML table and frame borders

`-no-mouse`

deactivate mouse support.

`-config file`

use *file* instead of the default config file

Document Colors

Hyperlinks and images are displayed as follows.

	Color mode	Monochrome mode
Hyperlinks	blue	underline
inline images	green	reverse
form input	red	reverse

These colors can be customized using the option setting command `o`.

Functions and Key bindings

After invoking `w3m`, you can control by typing in functions by name or with keystroke combinations bound to a function. There are default key-bindings.

You can customize the key bindings (except those for menu operations and line-editing) in a `~/.w3m/keymap` file. For example,

```
keymap C-o NEXT_PAGE
```

binds the command `NEXT_PAGE` (normally bound to `SPC` and `C-v`) to `control-o`. See [README.func](#) for a list of available functions. Original and Lynx-like keymap definitions are provided as examples: [keymap.default](#) and [keymap.lynx](#).

Throughout, the *C*- and *M*- notations indicate the modifiers *control* and *meta*. The *ALT*-key replaces the latter whereas pressing the *ESC*-key toggles between *meta*-modified and simple keystrokes. The minus indicates pressing the keys simultaneously whereas a space represents that one key is pressed after the other, i.e. `2 M` simply means `2` followed by `M`.

Function	Key binding		Description
	Default	Lynx-like	
In-page navigation			
NEXT_PAGE	SPC, C-v, +, PGDN	SPC, C-v, +	Scroll one page downwards
PREVIOUS_PAGE	b, M-v, -, PGUP	b, M-v, -	Scroll one page upwards
MOVE_RIGHT	l, C-f, RIGHT	l	Move cursor right (with a half-screen shift at the screen edge)
MOVE_LEFT	h, C-b, LEFT	h	Move cursor left (with a half-screen shift at the screen edge)
MOVE_DOWN	j, C-n, DOWN	j	Move cursor down (with a one-line scroll at the screen edge)
MOVE_UP	k, C-p, UP	k	Move cursor up (with a one-line scroll at the screen edge)
UP	J	J	Scroll the screen up one line
DOWN	K	K	Scroll the screen down one line
LINE_BEGIN	^, C-a	^	Go to the beginning of the line
LINE_END	\$, C-e	\$	Go to the end of the line
NEXT_WORD	w	w	Move to the next word
PREVIOUS_WORD	W	W	Move to the previous word
SHIFT_RIGHT	>	>	Shift screen right
SHIFT_LEFT	<	<	Shift screen left
RIGHT	.	.	Shift screen one column right
LEFT	,	,	Shift screen one column left
BEGIN	g, M-<, HOME	C-a, M-<	Go to the first line
END	G, M->, END	C-e, M->	Go to the last line
GOTO_LINE	M-g	G	Go to the specified line
LINE_INFO	C-g	C-g	Display current position in document
CENTER_H	Z	Z	Center screen right and left of the cursor column

CENTER_V	z	z	Center screen above and below the cursor line
NEXT_LINK	TAB	TAB, C-n, DOWN	Move to the next hyperlink
PREVIOUS_LINK	M-TAB, C-u	M-TAB, C-u, C-p, UP	Move to the previous hyperlink
LINK_BEGIN	[[Move to the first hyperlink
LINK_END]]	Move to the last hyperlink
MOVE_LIST_MENU	M-m	M-m	Pop up menu to navigate between hyperlinks

Hyperlink Operations

GOTO_LINK	C-j, C-m, RET	C-f, C-j, C-m, RET, RIGHT	Follow current hyperlink in a new buffer
LIST_MENU	M-l	M-l	Pop up hyperlink menu and select one to be followed
SAVE_LINK	a, M-RET	d, M-RET	Save the hyperlink target
PEEK_LINK	u	u	Show target address
PEEK_IMG	i	i	Show image address
VIEW_IMAGE	I	I	Display image in viewer
SAVE_IMAGE	M-I	M-I	Save inline image
MARK_WORD	;	;	Turn current word into hyperlink
MARK_URL	:	:	Turn URL-like strings into hyperlinks
MARK_MID	M-:	M-:	Turn Message-ID-like strings into hyperlinks
PEEK	c	c	Show current address
INFO	=	=	Display information about the current document
HISTORY	C-h	C-h	Show browsing history
EXTERN	M	M	Browse current document using external browser (prefix 2, 3, ..., or 9 to invoke alternate configured browsers, e.g. 3 M)
EXTERN_LINK	M-M	M-M	Browse link using external browser (prefixed as above, e.g. 3 M-M)

Bookmark management

BOOKMARK	M-b	v, M-b	View bookmarks
ADD_BOOKMARK	M-a	a, M-a	Add current page to bookmarks

File/Stream Operations

GOTO	U	g, U	Open specified document in a new buffer
GOTO_RELATIVE	M-u	M-u	Open relative address in a new buffer
LOAD	V	V	Open local file in a new buffer
READ_SHELL	@	@	Execute shell command and display output in a new buffer

PIPE_SHELL	#	#	Execute shell command and display output in a new buffer
Content Operations			
VIEW	v	\	Toggle between source code and rendered view of the document
LIST	L	L	List link elements, hyperlinks and images
FRAME	F	F	Toggle rendering HTML frames
REDRAW	C-l	C-l, C-w	Draw the screen anew
RELOAD	R	R, C-r	Load current document anew
EDIT	E	E	Edit local source
EDIT_SCREEN	M-e	M-e	Edit rendered copy of document
PIPE_BUF			Pipe current buffer through a shell command and display output
DOWNLOAD	M-s	M-s	Save document source
SAVE_SCREEN	S	S, p	Save rendered document
Buffer/Tab Navigation			
BACK	B	B, C-b, LEFT	Close current buffer and return to the one below in stack
SELECT_MENU	s	s, C-h	Pop up buffer-stack menu
TAB_MENU	M-t	M-t	Pop up tab selection menu
NEW_TAB	T	T	Open a new tab (with current document)
TAB_LINK	C-t	C-t	Follow current hyperlink in a new tab
NEXT_TAB	}	}	Switch to the next tab
PREV_TAB	{	{	Switch to the previous tab
CLOSE_TAB	C-q	C-q	Close the current tab
Searches			
SEARCH	/	/, C-s	Search forward
SEARCH_BACK	?		Search backward
ISEARCH	C-s		Incremental search forward
ISEARCH_BACK	C-r		Incremental search backward
SEARCH_NEXT	n	n	Continue search forward
SEARCH_PREV	N		Continue search backward
WRAP_TOGGLE	C-w	w	Toggle wrapping mode in searches
Mark Management			
MARK	C-SPC	C-SPC	Set/unset mark
PREV_MARK	M-p	P	Go to the previous mark
NEXT_MARK	M-n	N	Go to the next mark
REG_MARK	"	"	Mark all occurrences of a pattern

Miscellaneous			
COMMAND	M-c	M-c	Invoke one or more semicolon-separated w3m functions
HELP	H	H, ?	Show help panel
OPTIONS	o	o	Display options setting panel
SET_OPTION	M-o	M-o	Set option
DEFINE_KEY	M-k	M-k	Define a binding between a key stroke combination and a command
MOUSE_TOGGLE	m	m	Toggle mouse support
DOWNLOAD_LIST	D	D	Display downloads panel
COOKIE	C-k	C-k	Show cookie jar
UNDO	((Cancel the last cursor movement
REDO))	Cancel the last undo
EXEC_SHELL	!	!	Execute shell command
DICTIONARY_WORD	M-w	M-w	Execute dictionary command (see README.dict)
DICTIONARY_WORD_AT	M-W	M-W	Execute dictionary command for word at cursor
VERSION	r	r	Display the version of w3m
SUSPEND	C-z	C-z	Suspend w3m to background. (To be withdrawn with shell command fg.)
QUIT	q	q	Quit with confirmation request
EXIT	Q	Q	Quit at once

There are the following operational modes with built-in key bindings for relevant functions. These bindings are not subject to the DEFINE_KEY function or entries in the file keymap.

Function (internal name)	Key binding	Description
Menu Selection Mode		
BUF:NEXT	j, C-n, DOWN	Next item
BUF:PREV	k, C-p, UP	Previous item
BUF:DELETE	D	Delete item
BUF:GO	SPC, RET, RIGHT	Select item
Popup Menu Mode		
MENU:SELECT	SPC, RET, RIGHT	Select item
MENU:CLOSE	C-c	Close menu
MENU:CANCEL	LEFT, BKSPC, C-h	One selection step backward
MENU:DOWN	j, C-n, DOWN	Move to the next item
MENU:UP	k, C-p, UP	Move to the previous item
MENU:LINE_UP	K	Scroll up one item
MENU:LINE_DOWN	J	Scroll down one item
MENU:TOP	C-a	Go to the first item
MENU:LAST	C-e	Go to the final item

MENU:NEXT	C-v, C-f	Go to the next page
MENU:PREV	ESC v, C-b	Go to the previous page
MENU:SEARCH_FORE	C-s, /	Search forward
MENU:SEARCH_BACK	C-r, ?	Search backward
MENU:SEARCH_NEXT	n	Search for the next match
MENU:SEARCH_PREV	N	Search for the previous match
Line-editing Mode		
LINEEDIT:FORWARD	C-f, RIGHT	Move cursor forward
LINEEDIT:BACK	C-b, LEFT	Move cursor backward
LINEEDIT:BS	C-h, BKSPC	Delete previous character
LINEEDIT:DEL	C-d	Delete current character
LINEEDIT:KILL_AFTER	C-k	Delete everything after cursor
LINEEDIT:KILL_BEFORE	C-u	Delete everything before cursor
LINEEDIT:TOP	C-a	Move to the beginning of the line
LINEEDIT:BOTTOM	C-e	Move to the end of the line
LINEEDIT:PREV	C-p, UP	Fetch the previous string from the history list
LINEEDIT:NEXT	C-n, DOWN	Fetch the next string from the history list
LINEEDIT:COMPLETE	TAB, SPC	Try to complete filename
LINEEDIT:EDITOR	C-o	Edit with external editor
LINEEDIT:ACCEPT	RET	Accept input line

Mouse Operation

If w3m is compiled with mouse support and you are invoking w3m either from a console with GPM or from an X terminal emulator, you can use the mouse for navigation (in the case of rxvt, you need to set the TERM environment variable to "xterm" or "kterm").

An introduction to configure mouse actions is provided with [README.mouse](#).

Operation	Description
left click	First: Moves the cursor to the place indicated by the mouse pointer. Then: Follows a hyperlink the cursor is currently located and the mouse points to.
middle click	Back to the previous buffer.
right click	Open pop-up menu. You can choose an item by clicking it.
left drag	Scroll document. The default behavior is to grab the document and drag it. You can reverse the behavior (grab the window and drag it) with the option setting panel.

Local CGI scripts

You can run CGI scripts using w3m, without any HTTP server. This means that w3m behaves like an HTTP server and runs the CGI script, then reads the output of the script and displays it. For example, the bookmark registration system and default directory browser are realized as local CGI scripts. Moreover, such scripts allow w3m to be used as a form interface to acquire all kinds of data.

For security reason, CGI scripts invoked by w3m must be in one of the following directories:

- The directory where w3m-related files are stored (typically /usr/local/lib/w3m). This directory can be referenced as \$LIB.
- The /cgi-bin/ directory. You can map /cgi-bin/ to any directory you like in the option setting panel (the "Directory corresponds to /cgi-bin" field). You can specify multiple paths separated by ":", like /usr/local/cgi-bin:/home/aito/cgi-bin. It is not recommended to include the current directory to this path. To use a /cgi-bin/ directory, you must use a file:/cgi-bin URL as follows:

```
w3m -o cgi_bin=/path/to/cgi-bin file:/cgi-bin/script.cgi
```

The CGI script can use the special header `w3m-control:` to control w3m. This field can take any function (see [README.func](#)), and the specified function is invoked after the document is displayed. For example, the CGI output

```
Content-Type: text/plain
W3m-control: BACK
```

will display a blank page and delete that buffer immediately. This is useful when you don't want to display any page after the script is invoked. The next example

```
Content-Type: text/plain
W3m-control: DELETE_PREVBUF
contents.....
```

will override the current buffer.

Each header `w3m-control:` can contain only one function, but you can include more than one `w3m-control:` line in the HTTP header. In addition, you can specify an argument to the GOTO function:

```
Content-Type: text/plain
W3m-control: GOTO http://www.example.org/
```

This example works exactly the same way as the Location header:

```
Content-Type: text/plain
Location: http://www.example.org/
```

Note that this header has no effect when the CGI script is invoked through an HTTP server.