NAME zsh - the Z shell

OVERVIEW Because zsh contains many features, the zsh manual has been split into a number of sections:

```
zsh           Zsh overview (this section)
zshroadmap    Informal introduction to the manual
zshmisc       Anything not fitting into the other sections
zshexpn       Zsh command and parameter expansion
zshparam      Zsh parameters
zshoptions    Zsh options
zshbuiltins   Zsh built-in functions
zshzle        Zsh command line editing
zshcompwid    Zsh completion widgets
zshcompsys    Zsh completion system
zshcompctl    Zsh completion control
zshmodules    Zsh loadable modules
zshcalsys     Zsh built-in calendar functions
zshtcpsys     Zsh built-in TCP functions
zshzftpsys    Zsh built-in FTP client
zshcontrib    Additional zsh functions and utilities
zshall        Meta-man page containing all of the above
```

DESCRIPTION Zsh is a UNIX command interpreter (shell) usable as an inter-active lo- gin shell and as a shell script command processor. Of the standard shells, zsh most closely resembles ksh but includes many enhancements. It does not provide compatibility with POSIX or other shells in its de- fault operating mode: see the section Compatibility below.

```
Zsh has command line editing, builtin spelling correction, programmable
command completion, shell functions (with autoloading), a history mech-
anism, and a host of other features.
```

AUTHOR Zsh was originally written by Paul Falstad pf@zsh.org. Zsh is now maintained by the members of the zsh-workers mailing list <zsh-work-ers@zsh.org>. The development is currently coordinated by Peter Stephenson pws@zsh.org. The coordinator can be contacted at <coordi- nator@zsh.org>, but matters relating to the code should generally go to the mailing list.

AVAILABILITY Zsh is available from the following HTTP and anonymous FTP site.

```
ftp://ftp.zsh.org/pub/
https://www.zsh.org/pub/
)
```

```
The  up-to-date source code is available via Git from Sourceforge.  See
https://sourceforge.net/projects/zsh/ for details.  A  summary  of  in-
```

structions for the archive can be found at http://zsh.sourceforge.net/.

MAILING LISTS Zsh has 3 mailing lists:

```
<zsh-announce@zsh.org>
        Announcements about releases, major changes in the shell and the
        monthly posting of the Zsh FAQ.  (moderated)


<zsh-users@zsh.org>
        User discussions.


<zsh-workers@zsh.org>
        Hacking, development, bug reports and patches.


To subscribe or unsubscribe, send mail to the associated administrative
address for the mailing list.


<zsh-announce-subscribe@zsh.org>
<zsh-users-subscribe@zsh.org>
<zsh-workers-subscribe@zsh.org>
<zsh-announce-unsubscribe@zsh.org>
<zsh-users-unsubscribe@zsh.org>
<zsh-workers-unsubscribe@zsh.org>


YOU ONLY NEED TO JOIN ONE OF THE MAILING LISTS AS THEY ARE NESTED.  All
submissions to zsh-announce are automatically forwarded  to  zsh-users.
All  submissions  to zsh-users are automatically forwarded to zsh-work-
ers.


If you have problems subscribing/unsubscribing to any  of  the  mailing
lists,  send mail to <listmaster@zsh.org>.  The mailing lists are main-
tained by Karsten Thygesen <karthy@kom.auc.dk>.


The mailing lists are archived; the archives can be  accessed  via  the
administrative  addresses  listed above.  There is also a hypertext ar-
chive,  maintained  by  Geoff  Wing  <gcw@zsh.org>,  available  at
https://www.zsh.org/mla/.
```

THE ZSH FAQ Zsh has a list of Frequently Asked Questions (FAQ), maintained
by Peter Stephenson pws@zsh.org. It is regularly posted to the newsgroup
comp.unix.shell and the zsh-announce mailing list. The latest version can be
found at any of the Zsh FTP sites, or at http://www.zsh.org/FAQ/. The contact
address for FAQ-related matters is faqmaster@zsh.org.

THE ZSH WEB PAGE Zsh has a web page which is located at https://www.zsh.org/.
This is maintained by Karsten Thygesen karthy@zsh.org, of SunSITE Denmark.
The contact address for web-related matters is webmaster@zsh.org.

THE ZSH USERGUIDE A userguide is currently in preparation. It is intended to complement the manual, with explanations and hints on issues where the manual can be cabbalistic, hierographic, or downright mystifying (for example, the word 'hierographic' does not exist). It can be viewed in its current state at http://zsh.sourceforge.net/Guide/. At the time of writing, chapters dealing with startup files and their contents and the new com- pletion system were essentially complete.

INVOCATION The following flags are interpreted by the shell when invoked to deter- mine where the shell will read commands from:

```
-c      Take  the  first  argument  as a command to execute, rather than
        reading commands from a script or standard input.  If  any  fur-
        ther  arguments  are  given,  the  first  one is assigned to $0,
        rather than being used as a positional parameter.

-i      Force shell to be interactive.  It is still possible to  specify
        a script to execute.

-s      Force shell to read commands from the standard input.  If the -s
        flag is not present and an argument is given, the first argument
        is taken to be the pathname of a script to execute.
```

```
If  there are any remaining arguments after option processing, and nei-
ther of the options -c or -s was supplied, the first argument is  taken
as  the  file name of a script containing shell commands to be executed.
If the option PATH_SCRIPT is set, and the file name does not contain  a
directory  path  (i.e.  there is no `/' in the name), first the current
directory and then the command path given  by  the  variable  PATH  are
searched  for  the  script.   If the option is not set or the file name
contains a `/' it is used directly.

After the first one or two arguments  have  been  appropriated  as  de-
scribed  above,  the remaining arguments are assigned to the positional
parameters.

For further options,  which  are  common  to  invocation  and  the  set
builtin, see zshoptions(1).

The  long option `--emulate' followed (in a separate word) by an emula-
tion mode may be passed to the shell.  The emulation  modes  are  those
described for the emulate builtin, see zshbuiltins(1).  The `--emulate'
option must precede any other options (which might otherwise  be  over-
ridden),  but  following options are honoured, so may be used to modify
the requested emulation mode.  Note that certain extra steps are  taken
to ensure a smooth emulation when this option is used compared with the
emulate command within the shell: for example, variables that  conflict
```

with POSIX usage such as path are not defined within the shell.

Options may be specified by name using the -o option. -o acts like a single-letter option, but takes a following string as the option name. For example,

        zsh -x -o shwordsplit scr

runs the script scr, setting the XTRACE option by the corresponding letter `-x' and the SH_WORD_SPLIT  option  by  name.   Options  may  be turned  off  by  name  by using +o instead of -o.  -o can be stacked up with preceding single-letter options, so for example `-xo shwordsplit' or `-xoshwordsplit' is equivalent to `-x -o shwordsplit'.

Options may also be specified by name in GNU long option style, `--op-tion-name'.  When this is done, `-' characters in the option name are permitted: they are translated into `_', and thus ignored.  So, for ex-ample, `zsh --sh-word-split' invokes zsh with the SH_WORD_SPLIT option turned on.   Like other option syntaxes, options can be turned off by replacing the initial `-' with a `+'; thus `+-sh-word-split' is equiva-lent  to  `--no-sh-word-split'. Unlike other option syntaxes, GNU-style long options cannot be stacked with any other options, so for example `-x-shwordsplit'  is  an  error,  rather  than  being  treated like `-x --shwordsplit'.

The special GNU-style option `--version' is handled; it sends to  stan-dard  output  the shell's version information, then exits successfully. `--help' is also handled; it sends to standard output a list of options that can be used when invoking the shell, then exits successfully.

Option processing may be finished, allowing following arguments that start with `-' or `+' to be treated as normal arguments, in two ways. Firstly, a lone `-' (or `+') as an argument by itself ends option pro-cessing. Secondly, a special option `--' (or `+-'), which may be spec-ified on its own (which is the standard POSIX usage) or may be stacked with preceding options (so `-x-' is equivalent to  `-x --').   Options are not permitted to be stacked after `--' (so `-x-f' is an error), but note the GNU-style option form discussed above, where `--shwordsplit' is permitted and does not end option processing.

Except when the sh/ksh emulation single-letter options are in effect, the option `-b' (or `+b') ends option processing. `-b' is  like  `--', except that further single-letter options can be stacked after the `-b' and will take effect as normal.

COMPATIBILITY Zsh tries to emulate sh or ksh when it is invoked as sh or ksh

respec- tively; more precisely, it looks at the first letter of the name by which it was invoked, excluding any initial `r'` (assumed to stand forrestricted’), and if that is `b'`,s’ or ‘k’ it will emulate sh or ksh. Furthermore, if invoked as su (which happens on certain systems when the shell is executed by the su command), the shell will try to find an alternative name from the SHELL environment variable and per- form emulation based on that.

In sh and ksh compatibility modes the following parameters are not spe-
cial and not initialized by the shell: ARGC, argv, cdpath, fignore,
fpath, HISTCHARS, mailpath, MANPATH, manpath, path, prompt, PROMPT,
PROMPT2, PROMPT3, PROMPT4, psvar, status, watch.

The usual zsh startup/shutdown scripts are not executed. Login shells
source /etc/profile followed by $HOME/.profile. If the ENV environment
variable is set on invocation, $ENV is sourced after the profile
scripts. The value of ENV is subjected to parameter expansion, command
substitution, and arithmetic expansion before being interpreted as a
pathname.   Note that the PRIVILEGED option also affects the execution
of startup files.

The following options are set if the shell is invoked as sh or ksh:
NO_BAD_PATTERN,   NO_BANG_HIST,   NO_BG_NICE,   NO_EQUALS,   NO_FUNC-
TION_ARGZERO, GLOB_SUBST, NO_GLOBAL_EXPORT, NO_HUP, INTERACTIVE_COM-
MENTS, KSH_ARRAYS, NO_MULTIOS, NO_NOMATCH, NO_NOTIFY, POSIX_BUILTINS,
NO_PROMPT_PERCENT, RM_STAR_SILENT, SH_FILE_EXPANSION, SH_GLOB, SH_OP-
TION_LETTERS, SH_WORD_SPLIT.   Additionally the BSD_ECHO and IG-
NORE_BRACES options are set if zsh is invoked as sh. Also, the KSH_OP-
TION_PRINT, LOCAL_OPTIONS, PROMPT_BANG, PROMPT_SUBST and SIN-
GLE_LINE_ZLE options are set if zsh is invoked as ksh.

RESTRICTED SHELL When the basename of the command used to invoke zsh starts with the letter `r'` or the-r’ command line option is supplied at invocation, the shell becomes restricted. Emulation mode is determined after stripping the letter ‘r’ from the invocation name. The following are disabled in restricted mode:

- changing directories with the cd builtin

- changing or unsetting the EGID, EUID, GID, HISTFILE, HISTSIZE,
  IFS, LD_AOUT_LIBRARY_PATH, LD_AOUT_PRELOAD, LD_LIBRARY_PATH,
  LD_PRELOAD, MODULE_PATH, module_path, PATH, path, SHELL, UID and
  USERNAME parameters

- specifying command names containing /

- specifying command pathnames using hash

- redirecting output to files

- using the exec builtin command to replace the shell with another command

- using jobs -Z to overwrite the shell process' argument and environment space

- using the ARGV0 parameter to override argv[0] for external commands

- turning off restricted mode with set +r or unsetopt RESTRICTED

These restrictions are enforced after processing the startup files.
The startup files should set up PATH to point to a directory of com-
mands which can be safely invoked in the restricted environment.   They
may also add further restrictions by disabling selected builtins.

Restricted mode can also be activated any time by setting the RE-
STRICTED option. This immediately enables all the restrictions de-
scribed above even if the shell still has not processed all startup
files.

STARTUP/SHUTDOWN FILES Commands are first read from /etc/zsh/zshenv;
this cannot be overrid- den. Subsequent behaviour is modified by the RCS and
GLOBAL_RCS op- tions; the former affects all startup files, while the second
only af- fects global startup files (those shown here with an path starting with a
/). If one of the options is unset at any point, any subsequent startup file(s) of the
corresponding type will not be read. It is also possible for a file in $ZDOTDIR
to re-enable GLOBAL_RCS. Both RCS and GLOBAL_RCS are set by default.

Commands are then read from $ZDOTDIR/.zshenv. If the shell is a login
shell, commands are read from /etc/zsh/zprofile and then $ZDOT-
DIR/.zprofile.   Then, if the shell is interactive, commands are read
from /etc/zsh/zshrc and then $ZDOTDIR/.zshrc. Finally, if the shell is
a login shell, /etc/zsh/zlogin and $ZDOTDIR/.zlogin are read.

When a login shell exits, the files $ZDOTDIR/.zlogout and then
/etc/zsh/zlogout are read. This happens with either an explicit exit
via the exit or logout commands, or an implicit exit by reading
end-of-file from the terminal. However, if the shell terminates due to
exec'ing another process, the logout files are not read. These are
also affected by the RCS and GLOBAL_RCS options. Note also that the
RCS option affects the saving of history files, i.e. if RCS is unset
when the shell exits, no history file will be saved.

If ZDOTDIR is unset, HOME is used instead. Files listed above as being

```
in /etc may be in another directory, depending on the installation.

As  /etc/zsh/zshenv  is  run  for all instances of zsh, it is important
that it be kept as small as possible.  In particular, it is a good idea
to  put code that does not need to be run for every single shell behind
a test of the form `if [[ -o rcs ]]; then ...' so that it will  not  be
executed when zsh is invoked with the `-f' option.

Any  of  these files may be pre-compiled with the zcompile builtin com-
mand (see zshbuiltins(1)).  If a compiled file exists  (named  for  the
original  file plus the .zwc extension) and it is newer than the origi-
nal file, the compiled file will be used instead.
```

FILES $ZDOTDIR/.zshenv $ZDOTDIR/.zprofile $ZDOTDIR/.zshrc
$ZDOTDIR/.zlogin $ZDOTDIR/.zlogout ${TMPPREFIX}* (default is
/tmp/zsh*) /etc/zsh/zshenv /etc/zsh/zprofile /etc/zsh/zshrc /etc/zsh/zlogin
/etc/zsh/zlogout (installation-specific - /etc is the default)

SEE ALSO sh(1), csh(1), tcsh(1), rc(1), bash(1), ksh(1), zshall(1), zsh-
builtins(1), zshcalsys(1), zshcompwid(1), zshcompsys(1), zshcompctl(1), zsh-
contrib(1), zshexpn(1), zshmisc(1), zshmodules(1), zshoptions(1), zshparam(1),
zshroadmap(1), zshtcpsys(1), zshzftpsys(1), zshzle(1)

```
IEEE  Standard  for  information Technology - Portable Operating System
Interface (POSIX) - Part 2: Shell and Utilities, IEEE Inc,  1993,  ISBN
1-55937-255-9.
```

zsh 5.7.1 February 3, 2019 ZSH(1)