

gnu:guide/sh-utils

COLLABORATORS

	<i>TITLE :</i> gnu:guide/sh-utils		
<i>ACTION</i>	<i>NAME</i>	<i>DATE</i>	<i>SIGNATURE</i>
WRITTEN BY		April 16, 2022	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

1	gnu:guide/sh-utils	1
1.1	gnu:guide/sh-utils.guide	1
1.2	sh-utils.guide/Introduction	2
1.3	sh-utils.guide/Common options	2
1.4	sh-utils.guide/Printing text	3
1.5	sh-utils.guide/echo invocation	3
1.6	sh-utils.guide/printf invocation	4
1.7	sh-utils.guide/yes invocation	5
1.8	sh-utils.guide/Conditions	5
1.9	sh-utils.guide/false invocation	6
1.10	sh-utils.guide/true invocation	6
1.11	sh-utils.guide/test invocation	6
1.12	sh-utils.guide/File type tests	7
1.13	sh-utils.guide/Access permission tests	8
1.14	sh-utils.guide/File characteristics tests	8
1.15	sh-utils.guide/String tests	9
1.16	sh-utils.guide/Numeric tests	9
1.17	sh-utils.guide/Connectives for test	10
1.18	sh-utils.guide/expr invocation	10
1.19	sh-utils.guide/Relations for expr	11
1.20	sh-utils.guide/Numeric expressions	11
1.21	sh-utils.guide/String expressions	12
1.22	sh-utils.guide/Examples of expr	13
1.23	sh-utils.guide/Redirection	13
1.24	sh-utils.guide/tee invocation	13
1.25	sh-utils.guide/File name manipulation	14
1.26	sh-utils.guide/basename invocation	14
1.27	sh-utils.guide/dirname invocation	15
1.28	sh-utils.guide/pathchk invocation	15
1.29	sh-utils.guide/Working context information	16

1.30	sh-utils.guide/pwd invocation	16
1.31	sh-utils.guide/stty invocation	17
1.32	sh-utils.guide/Control	18
1.33	sh-utils.guide/Input	19
1.34	sh-utils.guide/Output	20
1.35	sh-utils.guide/Local	21
1.36	sh-utils.guide/Combination	22
1.37	sh-utils.guide/Characters	23
1.38	sh-utils.guide/Special	24
1.39	sh-utils.guide/printenv invocation	25
1.40	sh-utils.guide/tty invocation	26
1.41	sh-utils.guide/User information	26
1.42	sh-utils.guide/id invocation	27
1.43	sh-utils.guide/logname invocation	28
1.44	sh-utils.guide/whoami invocation	28
1.45	sh-utils.guide/groups invocation	28
1.46	sh-utils.guide/users invocation	29
1.47	sh-utils.guide/who invocation	29
1.48	sh-utils.guide/System context	31
1.49	sh-utils.guide/date invocation	31
1.50	sh-utils.guide/Time directives	32
1.51	sh-utils.guide/Date directives	33
1.52	sh-utils.guide/Literal directives	34
1.53	sh-utils.guide/Setting the time	34
1.54	sh-utils.guide/Options for date	35
1.55	sh-utils.guide/Examples of date	36
1.56	sh-utils.guide/uname invocation	36
1.57	sh-utils.guide/hostname invocation	37
1.58	sh-utils.guide/Modified command invocation	38
1.59	sh-utils.guide/env invocation	38
1.60	sh-utils.guide/nice invocation	39
1.61	sh-utils.guide/nohup invocation	39
1.62	sh-utils.guide/su invocation	40
1.63	sh-utils.guide/Delaying	42
1.64	sh-utils.guide/sleep invocation	42
1.65	sh-utils.guide/Index	43

Chapter 1

gnu:guide/sh-utils

1.1 gnu:guide/sh-utils.guide

```
GNU shell utilities
*****
```

This manual minimally documents version GNU sh-utils 1.12 of the GNU shell utilities.

Introduction

Caveats, overview, and authors.

Common options

Common options.

Printing text

echo printf yes

Conditions

false true test expr

Redirection

tee

File name manipulation

dirname basename pathchk

Working context information

pwd stty printenv tty

User information

id logname whoami groups users who

System context

date uname hostname

Modified command invocation

env nice nohup su

Delaying

sleep

Index

General index.

1.2 sh-utils.guide/Introduction

Introduction

First of all, this manual is incomplete. The 'stty' section, in particular, needs substantial reorganization and additional explanatory text before it will be up to the standard of other GNU manuals. Explanatory text in general is lacking; the manual presently assumes you pretty much know what to do, and just need to be reminded of how. Thus, if you are interested, please get involved in improving this manual. The entire GNU community will benefit.

Some of these programs are useful only when writing shell scripts; utilities like these are, in fact, the "language" of shell scripts (to a great extent). Others are occasionally useful interactively.

The GNU shell utilities are mostly compatible with the POSIX.2 standard.

Please report bugs to 'bug-gnu-utils@prep.ai.mit.edu'. Remember to include the version number, machine architecture, input files, and any other information needed to reproduce the bug. See Bugs.

This manual is based on the Unix man pages in the distribution, which were originally written by David MacKenzie and updated by Jim Meyering. Francois Pinard did the initial conversion to Texinfo format. Karl Berry did the indexing, some reorganization, and editing of the results. Richard Stallman contributed his usual invaluable insights to the overall process.

1.3 sh-utils.guide/Common options

Common options

Certain options are available in all these programs. Rather than writing identical descriptions for each of the programs, they are described here. (In fact, every GNU program accepts (or should accept) these options.)

Many of these programs take arbitrary strings as arguments. In those cases, '--help' and '--version' are taken as these options only if

there is one and exactly one command line argument.

`--help`

Print a usage message listing all available options, then exit successfully.

`--version`

Print the version number, then exit successfully.

1.4 sh-utils.guide/Printing text

Printing text

This section describes commands that display text strings.

echo invocation

Print a line of text.

printf invocation

Format and print data.

yes invocation

Print a string until interrupted.

1.5 sh-utils.guide/echo invocation

`'echo'`: Print a line of text

=====

Synopsis:

echo [OPTION]... [STRING]...

`'echo'` writes each given STRING to standard output, with a space between each and a newline after the last one.

The program accepts the following options. Also see See

Common options

.

`-n`

Do not output the trailing newline.

`-e`

Enable interpretation of the following backslash-escaped

characters in each STRING:

```
'\a'
    alert (bell)

'\b'
    backspace

'\c'
    suppress trailing newline

'\f'
    form feed

'\n'
    new line

'\r'
    carriage return

'\t'
    horizontal tab

'\v'
    vertical tab

'\'
    backslash

'\NNN'
    the character whose ASCII code is NNN (octal); if NNN is not
    a valid octal number, it is printed literally.
```

1.6 sh-utils.guide/printf invocation

`'printf'`: Format and print data

=====

Synopsis:

```
printf FORMAT [ ARGUMENT ]...
```

`'printf'` prints the FORMAT string, interpreting `'%'` directives and `'\'` escapes in the same way as the C `'printf'` function. The FORMAT argument is re-used as necessary to convert all of the given ARGUMENTs.

`'printf'` has one additional directive, `'%b'`, which prints its argument string with `'\'` escapes interpreted in the same way as in the FORMAT string.

`'printf'` interprets `'\0ooo'` in FORMAT as an octal number (if 000 is 0 to 3 octal digits) specifying a character to print, and `'\xhhh'` as a hexadecimal number (if HHH is 1 to 3 hex digits) specifying a character

to print.

An additional escape, `\c`, causes `printf` to produce no further output.

The only options are a lone `--help` or `--version`. See

Common options

.

1.7 sh-utils.guide/yes invocation

`yes`: Print a string until interrupted

=====

`yes` prints the command line arguments, separated by spaces and followed by a newline, forever until it is killed. If no arguments are given, it prints `y` followed by a newline forever until killed.

The only options are a lone `--help` or `--version`. See

Common options

.

1.8 sh-utils.guide/Conditions

Conditions

This section describes commands that are primarily useful for their exit status, rather than their output. Thus, they are often used as the condition of shell `if` statements, or as the last command in a pipeline.

`false` invocation

Do nothing, unsuccessfully.

`true` invocation

Do nothing, successfully.

`test` invocation

Check file types and compare values.

`expr` invocation

Evaluate expressions.

1.9 sh-utils.guide/false invocation

```
`false': Do nothing, unsuccessfully
```

```
=====
```

``false'` does nothing except return an exit status of 1, meaning "failure". It can be used as a place holder in shell scripts where an unsuccessful command is needed.

Any arguments are ignored, except for a lone `--help` or `--version` (see

```
Common options
).
```

1.10 sh-utils.guide/true invocation

```
`true': Do nothing, successfully
```

```
=====
```

``true'` does nothing except return an exit status of 0, meaning "success". It can be used as a place holder in shell scripts where a successful command is needed, although the shell built-in command `:` (colon) may be faster.

Any arguments are ignored, except for a lone `--help` or `--version` (see

```
Common options
).
```

1.11 sh-utils.guide/test invocation

```
`test': Check file types and compare values
```

```
=====
```

``test'` returns a status of 0 (true) or 1 (false) depending on the evaluation of the conditional expression `EXPR`. Each part of the expression must be a separate argument.

``test'` has file status checks, string operators, and numeric comparison operators.

Because most shells have a built-in command by the same name, using the unadorned command name in a script or interactively may get you different functionality than that described here.

Besides the options below, 'test' accepts a lone '--help' or '--version'. See

Common options

. A single non-option argument is also allowed: 'test' returns true if the argument is not null.

File type tests

-[bcdfhLpSt]

Access permission tests

-[gkruwxOG]

File characteristics tests

-e -s -nt -ot -ef

String tests

-z -n = !=

Numeric tests

-eq -ne -lt -le -gt -ge

Connectives for test

! -a -o

1.12 sh-utils.guide/File type tests

File type tests

These options test for particular types of files. (Everything's a file, but not all files are the same!)

'-b FILE'

True if FILE exists and is a block special device.

'-c FILE'

True if FILE exists and is a character special device.

'-d FILE'

True if FILE exists and is a directory.

'-f FILE'

True if FILE exists and is a regular file.

'-h FILE'

'-L FILE'

True if FILE exists and is a symbolic link.

'-p FILE'

True if FILE exists and is a named pipe.

```
`-S FILE'  
    True if FILE exists and is a socket.  
  
`-t [ FD ]'  
    True if FD is opened on a terminal.  If FD is omitted, it defaults  
    to 1 (standard output).
```

1.13 sh-utils.guide/Access permission tests

Access permission tests

These options test for particular access permissions.

```
`-g FILE'  
    True if FILE exists and has its set-group-id bit set.  
  
`-k FILE'  
    True if FILE has its "sticky" bit set.  
  
`-r FILE'  
    True if FILE exists and is readable.  
  
`-u FILE'  
    True if FILE exists and has its set-user-id bit set.  
  
`-w FILE'  
    True if FILE exists and is writable.  
  
`-x FILE'  
    True if FILE exists and is executable.  
  
`-O FILE'  
    True if FILE exists and is owned by the current effective user id.  
  
`-G FILE'  
    True if FILE exists and is owned by the current effective group id.
```

1.14 sh-utils.guide/File characteristics tests

File characteristics tests

These options test other file characteristics.

```
`-e FILE'  
    True if FILE exists.  
  
`-s FILE'
```

True if FILE exists and has a size greater than zero.

``FILE1 -nt FILE2'`

True if FILE1 is newer (according to modification date) than FILE2.

``FILE1 -ot FILE2'`

True if FILE1 is older (according to modification date) than FILE2.

``FILE1 -ef FILE2'`

True if FILE1 and FILE2 have the same device and inode numbers, i.e., if they are hard links to each other.

1.15 sh-utils.guide/String tests

String tests

These options test string characteristics. Strings are not quoted for `'test'`, though you may need to quote them to protect characters with special meaning to the shell, e.g., spaces.

``-z STRING'`

True if the length of STRING is zero.

``-n STRING'`

``STRING'`

True if the length of STRING is non-zero.

``STRING1 = STRING2'`

True if the strings are equal.

``STRING1 != STRING2'`

True if the strings are not equal.

1.16 sh-utils.guide/Numeric tests

Numeric tests

Numeric relationals. The arguments must be entirely numeric (possibly negative), or the special expression `'-l STRING'`, which evaluates to the length of STRING.

``ARG1 -eq ARG2'`

``ARG1 -ne ARG2'`

``ARG1 -lt ARG2'`

``ARG1 -le ARG2'`

``ARG1 -gt ARG2'`

``ARG1 -ge ARG2'`

These arithmetic binary operators return true if ARG1 is equal,

not-equal, less-than, less-than-or-equal, greater-than, or greater-than-or-equal than ARG2, respectively.

For example:

```
test -1 -gt -2 && echo yes
=> yes
test -l abc -gt 1 && echo yes
=> yes
test 0x100 -eq 1
error--> test: integer expression expected before -eq
```

1.17 sh-utils.guide/Connectives for test

Connectives for 'test'

The usual logical connectives.

'! EXPR'

True if EXPR is false.

'EXPR1 -a EXPR2'

True if both EXPR1 and EXPR2 are true.

'EXPR1 -o EXPR2'

True if either EXPR1 or EXPR2 is true.

1.18 sh-utils.guide/expr invocation

'expr': Evaluate expressions

=====

'expr' evaluates an expression and writes the result on standard output. Each token of the expression must be a separate argument.

Operands are either numbers or strings. 'expr' coerces anything appearing in an operand position to an integer or a string depending on the operation being applied to it.

Strings are not quoted for 'expr', though you may need to quote them to protect characters with special meaning to the shell, e.g., spaces.

Operators may given as infix symbols or prefix keywords. Parentheses may be used for grouping in the usual manner (you must quote parentheses to avoid the shell evaluating them, however).

Exit status:

0 if the expression is neither null nor 0,

1 if the expression is null or 0,
2 for invalid expressions.

```

Relations for expr
  | & < <= = == != >= >

Numeric expressions
  + - * / %

String expressions
  : match substr index length

Examples of expr
  Examples.

```

1.19 sh-utils.guide/Relations for expr

Relations for 'expr'

The usual logical connectives and relations, in order of precedence.

'|'

Yields its first argument if it is neither null nor 0, otherwise its second argument.

'&'

Yields its first argument if neither argument is null or 0, otherwise 0.

'< <= = == != >= >'

Compare the arguments and return 1 if the relation is true, 0 otherwise. '==' is a synonym for '='. 'expr' first tries to coerce both arguments to numbers and do a numeric comparison; if either coercion fails, it does a lexicographic comparison.

1.20 sh-utils.guide/Numeric expressions

Numeric expressions

Numeric operators, in order of increasing precedence. The connectives (previous section) have higher precedence, the string operators (following section) have lower.

'+ -'

Addition and subtraction. Both arguments are coerced to numbers;

an error occurs if this cannot be done.

``* / %'`

Multiplication, division, remainder. Both arguments are coerced to numbers; an error occurs if this cannot be done.

1.21 sh-utils.guide/String expressions

String expressions

String operators. These have lowest precedence.

``STRING : REGEX'`

Perform pattern matching. The arguments are coerced to strings and the second is considered to be a (basic, a la 'grep') regular expression, with a '^' implicitly prepended. The first argument is then matched against this regular expression.

If the match succeeds and REGEX uses '\(' and '\)', the ':' expression returns the part of STRING that matched the subexpression; otherwise, it returns the number of characters matched.

If the match fails, the ':' operator returns the null string if '\(' and '\)' are used in REGEX, otherwise 0.

Only the first '\(... \)' pair is relevant to the return value; additional pairs are meaningful only for grouping the regular expression operators.

See Regular Expression Library, for details of regular expression syntax.

``match STRING REGEX'`

An alternative way to do pattern matching. This is the same as ``STRING : REGEX'`.

``substr STRING POSITION LENGTH'`

Returns the substring of STRING beginning at POSITION with length at most LENGTH. If either POSITION or LENGTH is negative or non-numeric, returns the null string.

``index STRING CHARACTER-CLASS'`

Returns the first position in STRING where the first character in CHARSET was found. If no character in CHARSET is found in STRING, return 0.

``length STRING'`

Returns the length of STRING.

The keywords cannot be used as strings.

1.22 sh-utils.guide/Examples of expr

Examples of `expr`

Here are a few examples, including quoting for shell metacharacters.

To add 1 to the shell variable `foo`, in Bourne-compatible shells:

```
foo=`expr $foo + 1`
```

To print the non-directory part of the file name stored in `\$fname`, which need not contain a `/`.

```
expr $fname : '.*\/\(^.*\)\' '^|' $fname
```

```
expr abc : 'a\(.\)c'
```

```
=> b
```

```
expr index abcdef cz
```

```
=> 3
```

```
expr index index a
```

```
error--> expr: syntax error
```

1.23 sh-utils.guide/Redirection

Redirection

Unix shells commonly provide several forms of "redirection"--ways to change the input source or output destination of a command. But one useful redirection is performed by a separate command, not by the shell; it's described here.

tee invocation

Redirect output to multiple files.

1.24 sh-utils.guide/tee invocation

`tee`: Redirect output to multiple files

=====

The `tee` command copies standard input to standard output and also to any files given as arguments. This is useful when you want not only to send some data down a pipe, but also to save a copy.

Synopsis:

```
tee [ OPTION ]... [ FILE ]...
```

If a file being written to does not already exist, it is created. If a file being written to already exists, the data it previously contained is overwritten unless the `'-a'` option is used.

The program accepts the following options. Also see See

Common options

.

`'-a'`

`'--append'`

Append standard input to the given files rather than overwriting them.

`'-i'`

`'--ignore-interrupts'`

Ignore interrupt signals.

1.25 sh-utils.guide/File name manipulation

File name manipulation

This section describes commands that manipulate file names.

basename invocation

Strip directory and suffix from a file name.

dirname invocation

Strip non-directory suffix from a file name.

pathchk invocation

Check file name portability.

1.26 sh-utils.guide/basename invocation

`'basename'`: Strip directory and suffix from a file name

=====

Synopsis:

basename NAME [SUFFIX]

The `'basename'` command removes any leading directory components from NAME. If SUFFIX is specified and is identical to the end of NAME, it is removed from NAME as well. `'basename'` prints the result on standard

output.

The only options are '--help' and '--version'. See
Common options

.

1.27 sh-utils.guide/dirname invocation

'dirname': Strip non-directory suffix from a file name

=====

Synopsis:

dirname NAME

'dirname' prints all but the final slash-delimited component of NAME. If NAME is a single component, 'dirname' prints '.' (meaning the current directory).

The only options are '--help' and '--version'. See
Common options

.

1.28 sh-utils.guide/pathchk invocation

'pathchk': Check file name portability

=====

Synopsis:

pathchk [OPTION]... NAME...

For each NAME, 'pathchk' prints a message if any of these conditions is true:

1. one of the existing directories in NAME does not have search (execute) permission,
2. the length of NAME is larger than its filesystem's maximum file name length,
3. the length of one component of NAME, corresponding to an existing directory name, is larger than its filesystem's maximum length for a file name component.

The program accepts the following option. Also see See

Common options

.

``-p'`

``--portability'`

Instead of performing length checks on the underlying filesystem, test the length of each file name and its components against the POSIX.1 minimum limits for portability. Also check that the file name contains no characters not in the portable file name character set.

Exit status:

0 if all specified file names passed all of the tests,
1 otherwise.

1.29 sh-utils.guide/Working context information

Working context information

This section describes commands that display or alter the context in which you are working: the current directory, the terminal settings, and so forth. See also the user-related commands in the next section.

`pwd` invocation

Print working directory.

`stty` invocation

Print or change terminal characteristics.

`printenv` invocation

Print environment variables.

`tty` invocation

Print file name of terminal on standard input.

1.30 sh-utils.guide/pwd invocation

``pwd'`: Print working directory

=====

``pwd'` prints the fully resolved name of the current directory. That is, all components of the printed name will be actual directory names--none will be symbolic links.

Because most shells have a built-in command by the same name, using the unadorned command name in a script or interactively may get you

different functionality than that described here.

The only options are a lone `--help` or `--version`. See

Common options

.

1.31 sh-utils.guide/stty invocation

`stty`: Print or change terminal characteristics

=====

If given no arguments, `stty` prints the baud rate, line discipline number (on systems that support it), and line settings that have been changed from the values set by `stty sane`. Mode reading and setting are performed on the tty line connected to standard input.

`stty` accepts many non-option arguments that change aspects of the terminal line operation, as described below.

Synopses:

```
stty [ SETTING ]...
stty [ OPTION ]
```

The program accepts the following options. Also see See

Common options

.

`-a`

`--all`

Print all current settings in human-readable form.

`-g`

`--save`

Print all current settings in a form that can be used as an argument to another `stty` command to restore the current settings.

Many settings can be turned off by preceding them with a `-`. Such arguments are marked below with "May be negated" in their description. The descriptions themselves refer to the positive case, that is, when *not* negated (unless stated otherwise, of course).

Some settings are not available on all POSIX systems, since they use extensions. Such arguments are marked below with "Non-POSIX" in their description. On non-POSIX systems, those or other settings also may not be available, but it's not feasible to document all the variations: just try it and see.

Control

	Control settings
Input	Input settings
Output	Output settings
Local	Local settings
Combination	Combination settings
Characters	Special characters
Special	Special settings

1.32 sh-utils.guide/Control

Control settings

Control settings:

`'parenb'`

Generate parity bit in output and expect parity bit in input. May be negated.

`'parodd'`

Set odd parity (even if negated). May be negated.

`'cs5'`

`'cs6'`

`'cs7'`

`'cs8'`

Set character size to 5, 6, 7, or 8 bits.

`'hup'`

`'hupcl'`

Send a hangup signal when the last process closes the tty. May be negated.

`'cstopb'`

Use two stop bits per character (one if negated). May be negated.

`'cread'`

Allow input to be received. May be negated.

`'clocal'`

Disable modem control signals. May be negated.

``crtsets'`

Enable RTS/CTS flow control. Non-POSIX. May be negated.

1.33 sh-utils.guide/Input

Input settings

``ignbrk'`

Ignore breaks. May be negated.

``brkint'`

Make breaks cause an interrupt signal. May be negated.

``ignpar'`

Ignore parity errors. May be negated.

``parmrk'`

Mark parity errors (with a 255-0-character sequence). May be negated.

``inpck'`

Enable input parity checking. May be negated.

``istrip'`

Clear high (8th) bit of input characters. May be negated.

``inlcr'`

Translate newline to carriage return. May be negated.

``igncr'`

Ignore carriage return. May be negated.

``icrnl'`

Translate carriage return to newline. May be negated.

``ixon'`

Enable XON/XOFF flow control (that is, CTRL-s/CTRL-Q). May be negated.

``ixoff'`

``tandem'`

Enable sending of `'stop'` character when the system input buffer is almost full, and `'start'` character when it becomes almost empty again. May be negated.

``iuclc'`

Translate uppercase characters to lowercase. Non-POSIX. May be negated.

``ixany'`

Allow any character to restart output (only the start character if negated). Non-POSIX. May be negated.

``imaxbel'`

Enable beeping and not flushing input buffer if a character arrives when the input buffer is full. Non-POSIX. May be negated.

1.34 sh-utils.guide/Output

Output settings

These arguments specify output-related operations.

``opost'`

Postprocess output. May be negated.

``olcuc'`

Translate lowercase characters to uppercase. Non-POSIX. May be negated.

``ocrnl'`

Translate carriage return to newline. Non-POSIX. May be negated.

``onlcr'`

Translate newline to carriage return-newline. Non-POSIX. May be negated.

``onocr'`

Do not print carriage returns in the first column. Non-POSIX. May be negated.

``onlret'`

Newline performs a carriage return. Non-POSIX. May be negated.

``ofill'`

Use fill (padding) characters instead of timing for delays. Non-POSIX. May be negated.

``ofdel'`

Use delete characters for fill instead of null characters. Non-POSIX. May be negated.

``nll'`

``nl0'`

Newline delay style. Non-POSIX.

``cr3'`

``cr2'`

``cr1'`

``cr0'`

Carriage return delay style. Non-POSIX.

``tab3'`

``tab2'`

``tab1'`

``tab0'`
Horizontal tab delay style. Non-POSIX.

``bs1'`
``bs0'`
Backspace delay style. Non-POSIX.

``vt1'`
``vt0'`
Vertical tab delay style. Non-POSIX.

``ff1'`
``ff0'`
Form feed delay style. Non-POSIX.

1.35 sh-utils.guide/Local

Local settings

``isig'`
Enable ``interrupt'`, ``quit'`, and ``suspend'` special characters. May be negated.

``icanon'`
Enable ``erase'`, ``kill'`, ``werase'`, and ``rprnt'` special characters. May be negated.

``iexten'`
Enable non-POSIX special characters. May be negated.

``echo'`
Echo input characters. May be negated.

``echoe'`
``crterase'`
Echo ``erase'` characters as backspace-space-backspace. May be negated.

``echok'`
Echo a newline after a ``kill'` character. May be negated.

``echonl'`
Echo newline even if not echoing other characters. May be negated.

``noflsh'`
Disable flushing after ``interrupt'` and ``quit'` special characters. May be negated.

``xcase'`
Enable input and output of uppercase characters by preceding their lowercase equivalents with ``\'`, when ``icanon'` is set. Non-POSIX. May be negated.

``tostop'`
 Stop background jobs that try to write to the terminal. Non-POSIX.
 May be negated.

``echoprt'`
``prterase'`
 Echo erased characters backward, between ``\'` and ``/'`. Non-POSIX.
 May be negated.

``echoctl'`
``ctlecho'`
 Echo control characters in hat notation (``^C'`) instead of
 literally. Non-POSIX. May be negated.

``echoke'`
``crtkill'`
 Echo the `'kill'` special character by erasing each character on the
 line as indicated by the `'echoprt'` and `'echoe'` settings, instead
 of by the `'echoctl'` and `'echok'` settings. Non-POSIX. May be
 negated.

1.36 sh-utils.guide/Combination

Combination settings

Combination settings:

``evenp'`
``parity'`
 Same as `'parenb -parodd cs7'`. May be negated. If negated, same
 as `'-parenb cs8'`.

``oddp'`
 Same as `'parenb parodd cs7'`. May be negated. If negated, same as
`'-parenb cs8'`.

``nl'`
 Same as `'-icrnl -onlcr'`. May be negated. If negated, same as
`'icrnl -inlcr -igncr onlcr -ocrnl -onlret'`.

``ek'`
 Reset the `'erase'` and `'kill'` special characters to their default
 values.

``sane'`
 Same as:
`cread -ignbrk brkint -inlcr -igncr icrnl -ixoff -iuclic -ixany`
`imaxbel opost -olcuc -ocrnl onlcr -onocr -onlret -ofill -ofdel`
`nl0 cr0 tab0 bs0 vt0 ff0 isig icanon iexten echo echoe echok -echonl`
`-noflsh -xcase -tostop -echoprt echoctl echoke`
 and also sets all special characters to their default values.

``cooked'`

Same as `'brkint ignpar istrip icrnl ixon opost isig icanon'`, plus sets the `'eof'` and `'eol'` characters to their default values if they are the same as the `'min'` and `'time'` characters. May be negated. If negated, same as `'raw'`.

`'raw'`

Same as:

```
-ignbrk -brkint -ignpar -parmrk -inpcr -istrip -inlcr -igncr
-icrnl -ixon -ixoff -iuclc -ixany -imaxbel -opost -isig -icanon
-xcase min 1 time 0
```

May be negated. If negated, same as `'cooked'`.

`'cbreak'`

Same as `'-icanon'`. May be negated. If negated, same as `'icanon'`.

`'pass8'`

Same as `'-parenb -istrip cs8'`. May be negated. If negated, same as `'parenb istrip cs7'`.

`'litout'`

Same as `'-parenb -istrip -opost cs8'`. May be negated. If negated, same as `'parenb istrip opost cs7'`.

`'decctlq'`

Same as `'-ixany'`. Non-POSIX. May be negated.

`'tabs'`

Same as `'tab0'`. Non-POSIX. May be negated. If negated, same as `'tab3'`.

`'lcase'`

`'LCASE'`

Same as `'xcase iuclc olcuc'`. Non-POSIX. May be negated.

`'crt'`

Same as `'echoe echoctl echoke'`.

`'dec'`

Same as `'echoe echoctl echoke -ixany intr ^C erase ^? kill C-u'`.

1.37 sh-utils.guide/Characters

Special characters

The special characters' default values vary from system to system. They are set with the syntax `'name value'`, where the names are listed below and the value can be given either literally, in hat notation (`'^C'`), or as an integer which may start with `'0x'` to indicate hexadecimal, `'0'` to indicate octal, or any other digit to indicate decimal.

For GNU `stty`, giving a value of `'^-'` or `'undef'` disables that special character. (This is incompatible with Ultrix `'stty'`, which

uses a value of 'u' to disable a special character. GNU 'stty' treats a value 'u' like any other, namely to set that special character to u.)

'intr'

Send an interrupt signal.

'quit'

Send a quit signal.

'erase'

Erase the last character typed.

'kill'

Erase the current line.

'eof'

Send an end of file (terminate the input).

'eol'

End the line.

'eol2'

Alternate character to end the line. Non-POSIX.

'swtch'

Switch to a different shell layer. Non-POSIX.

'start'

Restart the output after stopping it.

'stop'

Stop the output.

'susp'

Send a terminal stop signal.

'dsusp'

Send a terminal stop signal after flushing the input. Non-POSIX.

'rprnt'

Redraw the current line. Non-POSIX.

'werase'

Erase the last word typed. Non-POSIX.

'lnext'

Enter the next character typed literally, even if it is a special character. Non-POSIX.

1.38 sh-utils.guide/Special

Special settings

``min N'`
Set the minimum number of characters that will satisfy a read until the time value has expired, when ``-icanon'` is set.

``time N'`
Set the number of tenths of a second before reads time out if the min number of characters have not been read, when ``-icanon'` is set.

``ispeed N'`
Set the input speed to N.

``ospeed N'`
Set the output speed to N.

``rows N'`
Tell the tty kernel driver that the terminal has N rows.
Non-POSIX.

``cols N'`
``columns N'`
Tell the kernel that the terminal has N columns. Non-POSIX.

``size'`
Print the number of rows and columns that the kernel thinks the terminal has. (Systems that don't support rows and cols in the kernel typically use the environment variables `'LINES'` and `'COLUMNS'` instead; however, GNU `'stty'` does not know anything about them.) Non-POSIX.

``line N'`
Use line discipline N. Non-POSIX.

``speed'`
Print the terminal speed.

``N'`
Set the input and output speeds to N. N can be one of: 0 50 75 110 134 134.5 150 200 300 600 1200 1800 2400 4800 9600 19200 38400 `'exta'` `'extb'`. `'exta'` is the same as 19200; `'extb'` is the same as 38400. 0 hangs up the line if ``-clocal'` is set.

1.39 sh-utils.guide/printenv invocation

``printenv'`: Print all or some environment variables

=====

Synopsis:

```
printenv [ OPTION ] [ VARIABLE ]...
```

If no VARIABLES are specified, `'printenv'` prints the value of every environment variable. Otherwise, it prints the value of each VARIABLE that is set, and nothing for those that are not set.

The only options are a lone `--help` or `--version`. See

Common options

.

Exit status:

0 if all variables specified were found
 1 if at least one specified variable was not found
 2 if a write error occurred

1.40 sh-utils.guide/tty invocation

`tty`: Print file name of terminal on standard input

=====

`tty` prints the file name of the tty connected to its standard input. It prints `not a tty` if standard input is not a tty.

Synopsis:

`tty [OPTION]...`

The program accepts the following option. Also see See

Common options

.

`-s`

`--silent`

`--quiet`

Print nothing; only return an exit status.

Exit status:

0 if standard input is a tty
 1 if standard input is not a tty
 2 if given incorrect arguments
 3 if a write error occurs

1.41 sh-utils.guide/User information

User information

This section describes commands that print user-related information: logins, groups, and so forth.

```

id invocation
    Print real and effective uid and gid.

logname invocation
    Print current login name.

whoami invocation
    Print effective user id.

groups invocation
    Print group names a user is in.

users invocation
    Print login names of users currently logged in.

who invocation
    Print who is currently logged in.

```

1.42 sh-utils.guide/id invocation

```
'id': Print real and effective uid and gid
```

```
=====
```

'id' prints information about the given user, or the process running it if no user is specified.

Synopsis:

```
id [ OPTION ]... [ USERNAME ]
```

By default, it prints the real user id, real group id, effective user id if different from the real user id, effective group id if different from the real group id, and supplemental group ids.

Each of these numeric values is preceded by an identifying string and followed by the corresponding user or group name in parentheses.

The options cause 'id' to print only part of the above information. Also see See

Common options

.

```
'-g'
```

```
'--group'
```

Print only the group id.

```
'-G'
```

```
'--groups'
```

Print only the supplementary groups.

```
'-n'
```

```

`--name'
    Print the user or group name instead of the ID number.  Requires
    '-u', '-g', or '-G'.

`-r'
`--real'
    Print the real, instead of effective, user or group id.  Requires
    '-u', '-g', or '-G'.

`-u'
`--user'
    Print only the user id.

```

1.43 sh-utils.guide/logname invocation

```

`logname': Print current login name
=====

```

``logname'` prints the calling user's name, as found in the file `"/etc/utmp"`, and exits with a status of 0. If there is no `"/etc/utmp"` entry for the calling process, ``logname'` prints an error message and exits with a status of 1.

The only options are `'--help'` and `'--version'`. See
Common options

.

1.44 sh-utils.guide/whoami invocation

```

`whoami': Print effective user id
=====

```

``whoami'` prints the user name associated with the current effective user id. It is equivalent to the command `'id -un'`.

The only options are `'--help'` and `'--version'`. See
Common options

.

1.45 sh-utils.guide/groups invocation

```

`groups': Print group names a user is in
=====

```


'groups' prints the names of the primary and any supplementary groups that each given USERNAME, or the current process if none are given, is in. If user names are given, the name of each user is printed before the list of that user's groups.

Synopsis:

```
groups [ USERNAME ]...
```

The group lists are equivalent to the output of the command 'id -Gn'.

The only options are '--help' and '--version'. See
Common options

.

1.46 sh-utils.guide/users invocation

'users': Print login names of users currently logged in

=====

'users' prints on a single line a blank-separated list of user names of users currently logged in to the current host. Each user name corresponds to a login session, so if a user has more than one login session, that user's name will appear the same number of times in the output.

Synopsis:

```
users [ FILE ]
```

With no FILE argument, 'users' extracts its information from the file '/etc/utmp'. If a file argument is given, 'users' uses that file instead. A common choice is '/etc/wtmp'.

The only options are '--help' and '--version'. See
Common options

.

1.47 sh-utils.guide/who invocation

'who': Print who is currently logged in

=====

Synopsis:

```
'who' [ OPTION ] [ FILE ] [ am i ]
```

If given no non-option arguments, 'who' prints the following

information for each user currently logged on: login name, terminal line, login time, and remote hostname or X display.

If given one non-option argument, 'who' uses that instead of '/etc/utmp' as the name of the file containing the record of users logged on. '/etc/wtmp' is commonly given as an argument to 'who' to look at who has previously logged on.

If given two non-option arguments, 'who' prints only the entry for the user running it (determined from its standard input), preceded by the hostname. Traditionally, the two arguments given are 'am i', as in 'who am i'.

The program accepts the following options. Also see See

Common options

.

'-m'

Same as 'who am i'.

'-q'

'--count'

Print only the login names and the number of users logged on.
Overrides all other options.

'-s'

Ignored; for compatibility with other versions of 'who'.

'-i'

'-u'

'--idle'

After the login time, print the number of hours and minutes that the user has been idle. '.' means the user was active in last minute. 'old' means the user was idle for more than 24 hours.

'-H'

'--heading'

Print a line of column headings.

'-w'

'-T'

'--mesg'

'--message'

'--writable'

After each login name print a character indicating the user's message status:

'+' allowing 'write' messages
'-' disallowing 'write' messages
'?' cannot find terminal device

1.48 sh-utils.guide/System context

System context

This section describes commands that print or change system-wide information.

```

date invocation
    Print or set system date and time.

uname invocation
    Print system information.

hostname invocation
    Print or set system name.
```

1.49 sh-utils.guide/date invocation

'date': Print or set system date and time

=====

'date' with no arguments prints the current time and date, in the format of the '%c' directive (described below).

Synopses:

```

date [ OPTION ]... [ +FORMAT ]
date [ -u|--utc|--universal ] [ MMDDHHMM[[CC]YY][.SS] ]
```

If given an argument that starts with a '+', 'date' prints the current time and date (or the time and date specified by the '--date' option, see below) in the format defined by that argument, which is the same as in the 'strftime' function. Except for directives, which start with '%', characters in the format string are printed unchanged. The directives are described below.

By default, 'date' pads numeric fields with zeroes. GNU 'date' recognizes the following numeric modifiers between the '%' and the directive. These are GNU extensions.

'_'

(hyphen) do not pad the field

'_'

(underscore) pad the field with spaces

Time directives

```
%[HkLMprSTXZ]
```

Date directives

```
%[aAbBcdDhjmUwWxyY]
```

Literal directives

```
%[%nt]
```

Setting the time

Changing the system clock.

Options for date

Instead of the current time.

Examples of date

Examples.

1.50 sh-utils.guide/Time directives

Time directives

'date' directives related to times.

'%H'

hour (00...23)

'%I'

hour (01...12)

'%k'

hour (0...23)

'%l'

hour (1...12)

'%M'

minute (00...59)

'%p'

locale's AM or PM

'%r'

time, 12-hour (hh:mm:ss [AP]M)

'%s'

seconds since the epoch, i.e., 1 January 1970 00:00:00 UTC (a GNU extension)

'%S'

second (00...61)

'%T'

time, 24-hour (hh:mm:ss)

`'%X'`
locale's time representation (%H:%M:%S)

`'%Z'`
time zone (e.g., EDT), or nothing if no time zone is determinable

1.51 sh-utils.guide/Date directives

Date directives

`'date'` directives related to dates.

`'%a'`
locale's abbreviated weekday name (Sun...Sat)

`'%A'`
locale's full weekday name, variable length (Sunday...Saturday)

`'%b'`
locale's abbreviated month name (Jan...Dec)

`'%B'`
locale's full month name, variable length (January...December)

`'%c'`
locale's date and time (Sat Nov 04 12:02:33 EST 1989)

`'%d'`
day of month (01...31)

`'%D'`
date (mm/dd/yy)

`'%h'`
same as %b

`'%j'`
day of year (001...366)

`'%m'`
month (01...12)

`'%U'`
week number of year with Sunday as first day of week (00...53)

`'%w'`
day of week (0...6) with 0 corresponding to Sunday

`'%W'`
week number of year with Monday as first day of week (00...53)

```
`%x'  
  locale's date representation (mm/dd/yy)  
  
`%y'  
  last two digits of year (00...99)  
  
`%Y'  
  year (1970....)
```

1.52 sh-utils.guide/Literal directives

Literal directives

'date' directives that produce literal strings.

```
`%%'  
  a literal %  
  
`%n'  
  a newline  
  
`%t'  
  a horizontal tab
```

1.53 sh-utils.guide/Setting the time

Setting the time

If given an argument that does not start with '+', 'date' sets the system clock to the time and date specified by that argument (as described below). You must have appropriate privileges to set the system clock. The '--date' and '--set' options may not be used with such an argument. The '--universal' option may be used with such an argument to indicate that the specified time and date are relative to Coordinated Universal Time rather than to the local time zone.

The argument must consist entirely of digits, which have the following meaning:

```
MM  
  month  
  
DD  
  day within month  
  
HH  
  hour
```

MM
minute

CC
first two digits of year (optional)

YY
last two digits of year (optional)

SS
second (optional)

The '--set' option also sets the system clock; see the next section.

1.54 sh-utils.guide/Options for date

Options for 'date'

The program accepts the following options. Also see See

Common options

.

'-d DATESTR'

'--date=DATESTR'

Display the time and date specified in DATESTR instead of the current time and date. DATESTR can be in almost any common format. It can contain month names, timezones, 'am' and 'pm', 'yesterday', 'ago', 'next', etc. The source file 'getdate.y' implements this parsing for all GNU routines; we need precise documentation!

'-f DATEFILE'

'--file=DATEFILE'

Parse each line in DATEFILE as with '-d' and display the resulting time and date. If DATEFILE is '-', use standard input. This is useful when you have many dates to process, because the system overhead of starting up the 'date' executable many times can be considerable.

'-s DATESTR'

'--set=DATESTR'

Set the time and date to DATESTR, See '-d' above.

'-u'

'--utc'

'--universal'

Print or set the time and date in Universal Coordinated Time instead of in local (wall clock) time.

1.55 sh-utils.guide/Examples of date

Examples of `date`

Here are a few examples. Also see the documentation for the `-d` option in the previous section.

- * To print the date of the day before yesterday:

```
date --date='2 days ago'
```

- * To print the date of the day three months and one day hence:

```
date --date='3 months 1 day'
```

- * To print the day of year of Christmas in the current year:

```
date --date='25 Dec' +%j
```

- * To print the current full month name and the day of the month:

```
date '+%B %d'
```

But this may not be what you want because for the first nine days of the month, the `%d` expands to a zero-padded two-digit field, for example `date -d 1may '+%B %d'` will print `May 01`.

- * To print a date without the leading zero for one-digit days of the month, you can use the (GNU extension) `--` modifier to suppress the padding altogether.

```
date -d=1may '+%B --%d'
```

- * To print the current date and time in the format required by many non-GNU versions of `date` when setting the system clock:

```
date +%m%d%H%M%Y.%S
```

- * To set the system clock forward by two minutes:

```
date --set='+2 minutes'
```

1.56 sh-utils.guide/uname invocation

`uname`: Print system information
=====

`uname` prints information about the machine and operating system it is run on. If no options are given, `uname` acts as if the `-s` option were given.

Synopsis:

```
uname [ OPTION ]...
```

If multiple options or `-a` are given, the selected information is printed in this order:


```
SYSDNAME NODENAME RELEASE OSVERSION MACHINE
```

The OSVERSION, at least, may well be multiple words. For example:

```
bash$ uname -a
=> Linux hayley 1.0.4 #3 Thu May 12 18:06:34 1994 i486
```

The program accepts the following options. Also see See

Common options

.

'-a'

'--all'

Print all of the below information.

'-m'

'--machine'

Print the machine (hardware) type.

'-n'

'--nodename'

Print the machine's network node hostname.

'-r'

'--release'

Print the operating system release.

'-s'

'--sysname'

Print the operating system name.

'-v'

Print the operating system version.

1.57 sh-utils.guide/hostname invocation

'hostname': Print or set system name

=====

With no arguments, 'hostname' prints the name of the current host system. With one argument, it sets the current host name to the specified string. You must have appropriate privileges to set the host name.

Synopsis:

```
hostname [ NAME ]
```

The only options are '--help' and '--version'. See

Common options

.

1.58 sh-utils.guide/Modified command invocation

Modified command invocation

This section describes commands that run other commands in some context different than the current one: a modified environment, as a different user, etc.

```

env invocation
    Modify environment variables.

nice invocation
    Modify scheduling priority.

nohup invocation
    Immunize to hangups.

su invocation
    Modify user and group id.
```

1.59 sh-utils.guide/env invocation

'env': Run a command in a modified environment

=====

'env' runs a command with an environment modified as specified by the command line arguments.

Synopses:

```

env [ OPTION ]... [ NAME=VALUE ]... [ COMMAND [ ARGS ]... ]
env
```

Arguments of the form 'VARIABLE=VALUE' set the environment variable VARIABLE to value VALUE. VALUE may be empty ('VARIABLE='). Setting a variable to an empty value is different from unsetting it.

The first remaining argument specifies the program name to invoke; it is searched for according to the 'PATH' environment variable. Any remaining arguments are passed as arguments to that program.

If no command name is specified following the environment specifications, the resulting environment is printed. This is like specifying a command name of 'printenv'.

The program accepts the following options. Also see See

Common options

.

`'-u NAME'`

`'--unset=NAME'`

Remove variable NAME from the environment, if it was in the environment.

`'-'`

`'-i'`

`'--ignore-environment'`

Start with an empty environment, ignoring the inherited environment.

1.60 sh-utils.guide/nice invocation

`'nice'`: Run a command with modified scheduling priority

=====

If no arguments are given, `'nice'` prints the current scheduling priority, which it inherited. Otherwise, `'nice'` runs the given COMMAND with its scheduling priority adjusted. If no ADJUSTMENT is given, the priority of the command is incremented by 10. You must have appropriate privileges to specify a negative adjustment. The priority can be adjusted by `'nice'` over the range of -20 (the highest priority) to 19 (the lowest).

Synopsis:

```
nice [ OPTION ]... [ COMMAND [ ARG ]... ]
```

Because most shells have a built-in command by the same name, using the unadorned command name in a script or interactively may get you different functionality than that described here.

The program accepts the following option. Also see See

Common options

.

`'-n ADJUSTMENT'`

`'-ADJUSTMENT'`

`'--adjustment=ADJUSTMENT'`

Add ADJUSTMENT instead of 10 to the command's priority.

1.61 sh-utils.guide/nohup invocation

``nohup``: Run a command immune to hangups

=====

``nohup`` runs the given `COMMAND` with hangup signals ignored, so that the command can continue running in the background after you log out.

Synopsis:

```
nohup COMMAND [ ARG ]...
```

Also, the scheduling priority is increased by 5. If standard output is a tty, it and standard error are redirected so that they are appended to the file ``nohup.out``; if that cannot be written to, they are appended to the file ``$HOME/nohup.out``. If that cannot be written to, the command is not run.

If ``nohup`` creates either ``nohup.out`` or ``$HOME/nohup.out``, it creates it with no "group" or "other" access permissions. It does not change the permissions if the output file already existed.

``nohup`` does not automatically put the command it runs in the background; you must do that explicitly, by ending the command line with an ``&``.

The only options are ``--help`` and ``--version``. See
Common options

.

1.62 sh-utils.guide/su invocation

``su``: Run a command with substitute user and group id

=====

``su`` allows one user to temporarily become another user. It runs a command (often an interactive shell) with the real and effective user id, group id, and supplemental groups of a given `USER`.

Synopsis:

```
su [ OPTION ]... [ USER [ ARG ]... ]
```

If no `USER` is given, the default is ``root``, the super-user. The shell to use is taken from `USER`'s ``passwd`` entry, or ``/bin/sh`` if none is specified there. If `USER` has a password, ``su`` prompts for the password unless run by a user with effective user id of zero (the super-user).

By default, ``su`` does not change the current directory. It sets the environment variables ``HOME`` and ``SHELL`` from the password entry for `USER`, and if `USER` is not the super-user, sets ``USER`` and ``LOGNAME`` to `USER`. By default, the shell is not a login shell.

Any additional ARGs are passed as additional arguments to the shell.

GNU `'su'` does not treat `'/bin/sh'` or any other shells specially (e.g., by setting `'argv[0]'` to `'-su'`, passing `'-c'` only to certain shells, etc.).

`'su'` can optionally be compiled to use `'syslog'` to report failed, and optionally successful, `'su'` attempts. (If the system supports `'syslog'`.) However, GNU `'su'` does not check if the user is a member of the `'wheel'` group; see below.

The program accepts the following options. Also see See

Common options

`'-c COMMAND'`

`'--command=COMMAND'`

Pass `COMMAND`, a single command line to run, to the shell with a `'-c'` option instead of starting an interactive shell.

`'-f'`

`'--fast'`

Pass the `'-f'` option to the shell. This probably only makes sense if the shell run is `'csh'` or `'tcsh'`, for which the `'-f'` option prevents reading the startup file (`'.cshrc'`). With Bourne-like shells, the `'-f'` option disables file name pattern expansion (globbing), which is not likely to be useful.

`'-l'`

`'-l'`

`'--login'`

Make the shell a login shell. This means the following. Unset all environment variables except `'TERM'`, `'HOME'`, and `'SHELL'` (which are set as described above), and `'USER'` and `'LOGNAME'` (which are set, even for the super-user, as described above), and set `'PATH'` to a compiled-in default value. Change to `USER`'s home directory. Prepend `'-'` to the shell's name, intended to make it read its login startup file(s).

`'-m'`

`'-p'`

`'--preserve-environment'`

Do not change the environment variables `'HOME'`, `'USER'`, `'LOGNAME'`, or `'SHELL'`. Run the shell given in the environment variable `'SHELL'` instead of the shell from `USER`'s `passwd` entry, unless the user running `'su'` is not the superuser and `USER`'s shell is restricted. A "restricted shell" is one that is not listed in the file `'/etc/shells'`, or in a compiled-in list if that file does not exist. Parts of what this option does can be overridden by `'--login'` and `'--shell'`.

`'-s SHELL'`

`'--shell=SHELL'`

Run `SHELL` instead of the shell from `USER`'s `passwd` entry, unless the user running `'su'` is not the superuser and `USER`'s shell is restricted (see `'-m'` just above).

Why GNU `su` does not support the `wheel` group
 =====

(This section is by Richard Stallman.)

Sometimes a few of the users try to hold total power over all the rest. For example, in 1984, a few users at the MIT AI lab decided to seize power by changing the operator password on the Twenex system and keeping it secret from everyone else. (I was able to thwart this coup and give power back to the users by patching the kernel, but I wouldn't know how to do that in Unix.)

However, occasionally the rulers do tell someone. Under the usual `su` mechanism, once someone learns the root password who sympathizes with the ordinary users, he or she can tell the rest. The "wheel group" feature would make this impossible, and thus cement the power of the rulers.

I'm on the side of the masses, not that of the rulers. If you are used to supporting the bosses and sysadmins in whatever they do, you might find this idea strange at first.

1.63 sh-utils.guide/Delaying

Delaying

Perhaps `wait` or other commands should be described here also?

sleep invocation

Delay for a specified time.

1.64 sh-utils.guide/sleep invocation

`sleep`: Delay for a specified time
 =====

`sleep` pauses for an amount of time specified by the sum of the values of the command line arguments.

Synopsis:

```
sleep [ NUMBER[smhd] ]...
```

Each argument is a number followed by an optional unit; the default

is seconds. The units are:

```
`s'
  seconds

`m'
  minutes

`h'
  hours

`d'
  days
```

The only options are `--help' and `--version'. See
Common options
.

1.65 sh-utils.guide/Index

Index

!	Connectives for test
!=	String tests
%	Numeric expressions
&	Relations for expr
*	Numeric expressions
+	Numeric expressions
-	env invocation
-	Numeric expressions
-	su invocation

-adjustment	nice invocation
-all	stty invocation
-all	uname invocation
-append	tee invocation
-command	su invocation
-count	who invocation
-date	Options for date
-fast	su invocation
-file	Options for date
-group	id invocation
-groups	id invocation
-heading	who invocation
-help	Common options
-idle	who invocation
-ignore-environment	env invocation
-ignore-interrupts	tee invocation
-login	su invocation
-machine	uname invocation
-mesg	who invocation

-message	who invocation
-name	id invocation
-nodename	uname invocation
-portability	pathchk invocation
-preserve-environment	su invocation
-quiet	tty invocation
-real	id invocation
-release	uname invocation
-save	stty invocation
-set	Options for date
-shell	su invocation
-silent	tty invocation
-sysname	uname invocation
-universal	Options for date
-user	id invocation
-utc	Options for date
-version	Common options
-writable	who invocation
-ADJUSTMENT	nice invocation

-a	Connectives for test
-a	tee invocation
-a	stty invocation
-a	uname invocation
-b	File type tests
-c	File type tests
-c	su invocation
-d	Options for date
-d	File type tests
-e	File characteristics tests
-e	echo invocation
-ef	File characteristics tests
-eq	Numeric tests
-f	Options for date
-f	su invocation
-f	File type tests
-g	stty invocation
-G	id invocation
-g	id invocation

-G	Access permission tests
-g	Access permission tests
-ge	Numeric tests
-gt	Numeric tests
-h	File type tests
-H	who invocation
-i	env invocation
-i	tee invocation
-i	who invocation
-k	Access permission tests
-l	su invocation
-L	File type tests
-le	Numeric tests
-lt	Numeric tests
-m	su invocation
-m	uname invocation
-m	who invocation
-n	nice invocation
-n	echo invocation

-n	uname invocation
-n	id invocation
-n	String tests
-ne	Numeric tests
-nt	File characteristics tests
-O	Access permission tests
-o	Connectives for test
-ot	File characteristics tests
-p	su invocation
-p	File type tests
-p	pathchk invocation
-q	who invocation
-r	Access permission tests
-r	id invocation
-r	uname invocation
-s	who invocation
-s	File characteristics tests
-S	File type tests
-s	uname invocation

-s	su invocation
-s	Options for date
-s	tty invocation
-t	File type tests
-T	who invocation
-u	Options for date
-u	who invocation
-u	id invocation
-u	Access permission tests
-u	env invocation
-unset	env invocation
-v	uname invocation
-w	Access permission tests
-w	who invocation
-x	Access permission tests
-z	String tests
.cshrc	su invocation
/	Numeric expressions
/bin/sh	su invocation

/etc/passwd	su invocation
/etc/shells	su invocation
/etc/utmp	who invocation
/etc/utmp	users invocation
/etc/utmp	logname invocation
/etc/wtmp	users invocation
/etc/wtmp	who invocation
<	Relations for expr
<=	Relations for expr
=	String tests
=	Relations for expr
==	Relations for expr
>	Relations for expr
>=	Relations for expr
date options	Options for date
-su	su invocation
access permission tests	Access permission tests
addition	Numeric expressions
am i	who invocation

and operator
Connectives for test

and operator
Relations for expr

appropriate privileges
hostname invocation

appropriate privileges
Setting the time

appropriate privileges
nice invocation

arbitrary date strings, parsing
Options for date

arbitrary text, displaying
echo invocation

arithmetic tests
Numeric tests

background jobs, stopping at terminal write
Local

backslash escapes
echo invocation

basename
basename invocation

baud rate, setting
Special

beeping at input buffer full
Input

beginning of time
Time directives

block special check
File type tests

breaks, cause interrupts
Input

breaks, ignoring
Input

brkint
Input

bsN
Output

bugs, reporting
Introduction

built-in shell commands, conflicts with
pwd invocation

built-in shell commands, conflicts with
nice invocation

built-in shell commands, conflicts with
test invocation

C-s/C-q flow control
Input

case translation
Local

cbreak
Combination

change or print terminal settings
stty invocation

character size
Control

character special check
File type tests

characters, special
Characters

check file types
test invocation

clocal
Control

cols
Special

COLUMNS
Special

columns
Special

combination settings
Combination

commands for delaying
Delaying

commands for exit status
Conditions

commands for file name manipulation
File name manipulation

commands for invoking other commands
Modified command invocation

commands for printing text
Printing text

commands for printing the working context
Working context information

commands for printing user information
User information

commands for redirection
Redirection

commands for system context
System context

common options
Common options

compare values
test invocation

comparison operators
Relations for expr

conditions
Conditions

conflicts with shell built-ins
nice invocation

conflicts with shell built-ins
pwd invocation

conflicts with shell built-ins
test invocation

connectives, logical
Connectives for test

connectives, logical
Relations for expr

context, system
System context

control characters, using ^C
Local

control settings
Control

cooked	Combination
coordinated universal time Options for date	
crN	Output
cread	Control
crt	Combination
crterase	Local
crtkill	Local
crtsets	Control
csN	Control
cstopb	Control
ctlecho	Local
current working directory, printing pwd invocation	
date	date invocation
date directives	Date directives
date strings, parsing Options for date	
dec	Combination
decctlq	Combination
delay for a specified time sleep invocation	
delaying commands	Delaying

destinations, multiple output
tee invocation

directives, date
Date directives

directives, literal
Literal directives

directives, time
Time directives

directory check
File type tests

directory components, printing
dirname invocation

directory, stripping from file names
basename invocation

dirname
dirname invocation

disabling special characters
Characters

displaying text
echo invocation

division
Numeric expressions

do nothing, successfully
true invocation

dsusp
Characters

echo
echo invocation

echo
Local

echoctl
Local

echoe
Local

echok
Local

echoke
Local

echonl	Local
echoprt	Local
effective uid and gid, printing id invocation	
effective UID, printing whoami invocation	
eight-bit characters Control	
eight-bit characters Combination	
eight-bit input Input	
ek	Combination
env	env invocation
environment variables, printing printenv invocation	
environment, preserving su invocation	
environment, printing env invocation	
environment, running a program in a modified env invocation	
eof	Characters
eol	Characters
eol2	Characters
epoch, seconds since Time directives	
equal string check String tests	
erase	Characters

failure exit status
false invocation

false
false invocation

fascism
su invocation

ffN
Output

fields, padding numeric
date invocation

file characteristics tests
File characteristics tests

file name manipulation
File name manipulation

file name pattern expansion, disabled
su invocation

file names, checking validity and portability
pathchk invocation

file names, stripping directory and suffix
basename invocation

file type tests
File type tests

flow control, hardware
Control

flow control, software
Input

flushing, disabling
Local

formatting times
date invocation

getdate.y
Options for date

globbing, disabled
su invocation

Greenwich Mean Time
Options for date

group wheel, not supported
su invocation

groups	groups invocation
hangups, immunity to	nohup invocation
hard link check	File characteristics tests
hardware flow control	Control
hardware type	uname invocation
hat notation for control characters	Local
help, online	Common options
history	Introduction
HOME	su invocation
hostname	hostname invocation
hostname	uname invocation
hup[cl]	Control
icanon	Local
icrnl	Input
id	id invocation
idle time	who invocation
iexten	Local
ignbrk	Input
igncr	Input

ignpar	Input
imaxbel	Input
immunity to hangups	nohup invocation
index	String expressions
information, about current users	who invocation
inlcr	Input
inpck	Input
input settings	Input
intr	Characters
introduction	Introduction
invocation of commands, modified	Modified command invocation
isig	Local
ispeed	Special
istrip	Input
iuclc	Input
ixany	Input
ixoff	Input
ixon	Input
kill	Characters

last DAY	Options for date
lcase	Combination
LCASE	Combination
leading directory components, stripping basename invocation	
length	String expressions
line	Special
line settings of terminal stty invocation	
LINES	Special
literal directives	Literal directives
litout	Combination
lnext	Characters
local settings	Local
logical and operator	Relations for expr
logical and operator	Connectives for test
logical connectives	Relations for expr
logical connectives	Connectives for test
logical or operator	Connectives for test
logical or operator	Relations for expr
login name, printing logname invocation	

login sessions, printing users with
users invocation

login shell
su invocation

login shell, creating
su invocation

login time
who invocation

logname
logname invocation

LOGNAME
su invocation

lowercase, translating to output
Output

machine type
uname invocation

machine-readable stty output
stty invocation

manipulation of file names
File name manipulation

match
String expressions

matching patterns
String expressions

message status
who invocation

min
Special

MIT AI lab
su invocation

modem control
Control

modified command invocation
Modified command invocation

modified environment, running a program in
env invocation

modifying scheduling priority
nice invocation

multiplication	Numeric expressions
name of operating system	uname invocation
named pipe check	File type tests
network node name	uname invocation
newer-than file check	File characteristics tests
newline echoing after kill	Local
newline, echoing	Local
newline, translating to crlf	Output
newline, translating to return	Input
next DAY	Options for date
nice	nice invocation
nl	Combination
nlN	Output
no-op	true invocation
node name	uname invocation
noflsh	Local
nohup	nohup invocation
nohup.out	nohup invocation
nohup.out	nohup invocation

non-directory suffix, stripping
dirname invocation

nonempty file check
File characteristics tests

nonzero-length string check
String tests

not-equal string check
String tests

numeric expressions
Numeric expressions

numeric field padding
date invocation

numeric tests
Numeric tests

ocrnl
Output

odd parity
Control

oddp
Combination

ofdel
Output

ofill
Output

olcuc
Output

older-than file check
File characteristics tests

onlcr
Output

onlret
Output

onocr
Output

operating system name
uname invocation

operating system release
uname invocation

operating system version	
uname invocation	
opost	Output
options for date	Options for date
or operator	Connectives for test
or operator	Relations for expr
ospeed	Special
output settings	Output
owned by effective gid check	Access permission tests
owned by effective uid check	Access permission tests
pad character	Output
pad instead of timing for delaying	Output
padding of numeric fields	date invocation
parenb	Control
parentheses for grouping	expr invocation
parity	Combination
parity errors, marking	Input
parity, ignoring	Input
parmrk	Input
parodd	Control

parsing date strings
Options for date

pass8
Combination

passwd entry, and su shell
su invocation

PATH
env invocation

PATH
su invocation

pathchk
pathchk invocation

pattern matching
String expressions

permission tests
Access permission tests

pipe fitting
tee invocation

portable file names, checking for
pathchk invocation

POSIX.2
Introduction

print name of current directory
pwd invocation

print system information
uname invocation

print terminal file name
tty invocation

printenv
printenv invocation

printf
printf invocation

printing all or some environment variables
printenv invocation

printing current user information
who invocation

printing current usernames
users invocation

printing groups a user is in
groups invocation

printing real and effective uid and gid
id invocation

printing text
echo invocation

printing text, commands for
Printing text

printing the current time
date invocation

printing the effective UID
whoami invocation

printing the hostname
hostname invocation

printing user's login name
logname invocation

priority, modifying
nice invocation

prterase
Local

pwd
pwd invocation

quit
Characters

raw
Combination

read from stdin and write to stdout and files
tee invocation

readable file check
Access permission tests

real uid and gid, printing
id invocation

redirection
Redirection

regular expression matching
String expressions

regular file check
File type tests

relations, numeric or string
Relations for expr

release of operating system
uname invocation

remainder
Numeric expressions

remote hostname
who invocation

repeated output of a string
yes invocation

restricted shell
su invocation

return, ignoring
Input

return, translating to newline
Input

return, translating to newline
Output

root, becoming
su invocation

rows
Special

rprnt
Characters

RTS/CTS flow control
Control

same file check
File characteristics tests

sane
Combination

scheduling priority, modifying
nice invocation

seconds since the epoch
Time directives

set-group-id check
Access permission tests

set-user-id check
Access permission tests

setting the hostname
hostname invocation

setting the time
Setting the time

SHELL
su invocation

shell utilities
Top

size
Special

sleep
sleep invocation

socket check
File type tests

software flow control
Input

special characters
Characters

special settings
Special

speed
Special

start
Characters

sticky bit check
Access permission tests

stop
Characters

stop bits
Control

strftime and date
date invocation

string expressions
String expressions

string tests
String tests

strip directory and suffix from file names
basename invocation

stripping non-directory suffix
dirname invocation

stty	stty invocation
su	su invocation
substitute user and group ids	su invocation
substr	String expressions
subtraction	Numeric expressions
successful exit	true invocation
suffix, stripping from file names	basename invocation
super-user, becoming	su invocation
supplementary groups, printing	groups invocation
susp	Characters
swtch	Characters
symbolic link check	File type tests
symbolic links and pwd	pwd invocation
syslog	su invocation
system context	System context
system information, printing	uname invocation
system name, printing	hostname invocation
tabN	Output
tabs	Combination

tandem	Input
tee	tee invocation
TERM	su invocation
terminal check	File type tests
terminal file name, printing	tty invocation
terminal lines, currently used	who invocation
terminal settings	stty invocation
test	test invocation
text, displaying	echo invocation
time	Special
time directives	Time directives
time formats	date invocation
time setting	Setting the time
time units	sleep invocation
time, printing or setting	date invocation
tomorrow	Options for date
tostop	Local
true	true invocation
tty	tty invocation

Twenex
su invocation

two-way parity
Control

u, and disabling special characters
Characters

uname
uname invocation

uppercase, translating to lowercase
Input

USER
su invocation

user id, switching
su invocation

user information, commands for
User information

user name, printing
logname invocation

usernames, printing current
users invocation

users
users invocation

utilities for shell programming
Top

utmp
logname invocation

valid file names, checking for
pathchk invocation

version number, finding
Common options

version of operating system
uname invocation

vtN
Output

werase
Characters

wheel group, not supported
su invocation

who

	who invocation
who am i	who invocation
whoami	whoami invocation
working context	Working context information
working directory, printing	pwd invocation
writable file check	Access permission tests
write, allowed	who invocation
xcase	Local
XON/XOFF flow control	Input
yes	yes invocation
yesterday	Options for date
zero-length string check	String tests
\0ooo	printf invocation
\0xhhh	printf invocation
\c	printf invocation
	Relations for expr
