



*Full credit is given to the above companies including the OS that this PDF file was generated!*

## ***Red Hat Enterprise Linux Release 9.2 Manual Pages on 'env.1' command***

**\$ man env.1**

ENV(1) User Commands ENV(1)

NAME

env - run a program in a modified environment

SYNOPSIS

env [OPTION]... [-] [NAME=VALUE]... [COMMAND [ARG]...]

DESCRIPTION

Set each NAME to VALUE in the environment and run COMMAND.

Mandatory arguments to long options are mandatory for short options too.

-i, --ignore-environment

start with an empty environment

-0, --null

end each output line with NUL, not newline

-u, --unset=NAME

remove variable from the environment

-C, --chdir=DIR

change working directory to DIR

-S, --split-string=S

process and split S into separate arguments; used to pass multiple arguments on shebang lines

--block-signal[=SIG]

block delivery of SIG signal(s) to COMMAND

--default-signal[=SIG]

reset handling of SIG signal(s) to the default

`--ignore-signal[=SIG]`

set handling of SIG signals(s) to do nothing

`--list-signal-handling`

list non default signal handling to stderr

`-v, --debug`

print verbose information for each processing step

`--help` display this help and exit

`--version`

output version information and exit

A mere `-` implies `-i`. If no COMMAND, print the resulting environment.

SIG may be a signal name like 'PIPE', or a signal number like '13'.

Without SIG, all known signals are included. Multiple signals can be comma-separated.

## OPTIONS

`-S/--split-string` usage in scripts

The `-S` option allows specifying multiple parameters in a script. Running a script named `1.pl` containing the following first line:

```
#!/usr/bin/env -S perl -w -T
```

...

Will execute `perl -w -T 1.pl`.

Without the `'-S'` parameter the script will likely fail with:

```
/usr/bin/env: 'perl -w -T': No such file or directory
```

See the full documentation for more details.

`--default-signal[=SIG]` usage

This option allows setting a signal handler to its default action, which is not possible using the traditional shell trap command. The following example ensures that `seq` will be terminated by `SIGPIPE` no matter how this signal is being handled in the process invoking the command.

```
sh -c 'env --default-signal=PIPE seq inf | head -n1'
```

## NOTES

POSIX's `exec(2)` pages says:

"many existing applications wrongly assume that they start with certain signals set to the default action and/or unblocked.... Therefore, it is best not to block or ignore signals across execs without explicit reason to do so, and especially not to block signals across execs of arbitrary (not closely cooperating) programs."

#### AUTHOR

Written by Richard Mlynarik, David MacKenzie, and Assaf Gordon.

#### REPORTING BUGS

GNU coreutils online help: <<https://www.gnu.org/software/coreutils/>>

Report any translation bugs to <<https://translationproject.org/team/>>

#### COPYRIGHT

Copyright © 2020 Free Software Foundation, Inc. License GPLv3+: GNU

GPL version 3 or later <<https://gnu.org/licenses/gpl.html>>.

This is free software: you are free to change and redistribute it.

There is NO WARRANTY, to the extent permitted by law.

#### SEE ALSO

sigaction(2), sigprocmask(2), signal(7)

Full documentation <<https://www.gnu.org/software/coreutils/env>>

or available locally via: info '(coreutils) env invocation'

GNU coreutils 8.32

January 2023

ENV(1)