NAME

env — alter and consume environment

SYNOPSIS

env [**−iv0**] [**−u** *var*]... [**-**] [*var*=*val*]...
env [**−iv0**] [**−u** *var*]... **−S** ‘[**-**] [*var*=*val*]...’ [*var*=*val*]...

env [**−ivL**] [**−u** *var*]... [**−C** *dir*] [**−**{**DBI**} [*signal*[,*signal*]...]]... [**-**] [*var*=*val*]... *program* [*argument*]...
env [**−ivL**] [**−u** *var*]... [**−C** *dir*] [**−**{**DBI**} [*signal*[,*signal*]...]]... **−S** ‘[**-**] [*var*=*val*]... *program* [*argument*]...’ [[*var*=*val*]... *program* [*argument*]...]

DESCRIPTION

Alters the environment, then writes it to the standard output stream (like `printenv`(1)), or additionally alters signal masks and dispositions and executes *program arguments* therewith.

The initial environment is the inherited one, or if **−i** is specified or the first non-option argument is **-**, the empty set. All variables specified via **−u** are then removed, and values specified as *var*=*val* are added. The first non-option argument without an **=** is the *program* — if one exists, it will be executed with the new environment; otherwise, that environment will be written in the same format as `printenv`(1) (*KEY*=*VALUE*, separated by newlines)

If **−S** is specified, it aborts option parsing and its value is tokenised and prepended to the remaining arguments; this is primarily useful for shebangs.

OPTIONS

−i, **−−ignore-environment**	Use the empty set, rather than the inherited environment, as the initial environment.
−v, **−−debug**	Write all actions and intermediate parsing steps to the standard error stream as they happen.
−0, **−−null**	Separate environment by NULs, rather than newlines. Only allowed if dumping environment (i.e. no *program* was specified).
−u, **−−unset**=*variable*	Remove *variable* from the environment.
−C, **−−chdir**=*dir*	Change working directory to *dir*. Only allowed if executing a *program*.
−S, **−−split-string**=*arguments*	Abort option parsing, and prepend tokenised *arguments* to the remaining arguments. *Ignored* if no tokens are extracted, but they function identically otherwise.

−S Tokenisation

Much like a vastly trivialised version of shell argument parsing. The string is split on runs of the space or tab characters, but the single (**’**) and double (**"**) quotes are recognised, and allow embedding whitespace into an argument and passing empty arguments. Either quote suppresses the other within it.

Variables from the inherited environment may be inserted via the **${***variable***}** construct. If said *variable* doesn’t exist, it will not begin a token if one wasn’t in progress. This means that **−S** ‘a **${***b***}** c’ consists of three tokens (‘a’, the value of *b*, even if empty, and ‘c’) if *b* is set, and two tokens (‘a’ and ‘c’) if it isn’t. The single (**’**) quote suppresses **${***variable***}** expansion.

The following escapes are recognised:

\t, **\n**, **\v**, **\f**, **\r**	The tab, line feed, vertical tab, form feed, and carriage return characters, respectively.
\, **\’**, **\"**, **\$**	Themselves, but suppress any other meanings.

Signal Handling

None of the following have an effect without *program*. For any given signal, the last of **-DI**, **-DB**, if any, applies. Signals are changed just before executing *program*. No changes are applied by default.

--default-signal	Set the default disposition for and unmask all signals.
--block-signal	Mask all signals.
--ignore-signal	Ignore all signals.
-D, **--default-signal**=[*signal*[,*signal*]...]	Set the default disposition for and unmask the specified *signal*s.
-B, **--block-signal**=[*signal*[,*signal*]...]	Mask the specified *signal*s.
-I, **--ignore-signal**=[*signal*[,*signal*]...]	Ignore the specified *signal*s.
-L, **--list-signal-handling**	List all masked and ignored (i.e. non-default handled; cf. `signal`(7)) signals to the standard error stream before executing *program*.

Signal Name

If *signal* starts with a digit, it's presumed to be a numerical signal value. Otherwise, if it starts with "**SIG**", that prefix is stripped for the purposes of further matching. All string comparisons are case-insensitive.

On platforms with `sys_signame`(3) (the BSD), *signal* is matched directly to the array.

Elsewhere, it's matched to the signal names known at compile time; the null signal is known as "Signal 0". Real-time signals, if any, can be specified in the format "**RT***integer*", where *integer* is a decimal number (NetBSD-style), or "**RTMIN+***integer*" and "**RTMAX−***integer*" (procps-style). Real-time signals must fall in [`SIGRTMIN`, `SIGRTMAX`] to be accepted.

ENVIRONMENT

`PATH` In which *program* is searched when requested, confer `execvp`(3).

EXIT STATUS

127	*program* wasn't found.
126	*program* exists, but couldn't be executed for a different reason.
125	an error occurred in **env**.
All others	returned by *program*, if executed, or **0**.

EXAMPLES

A classic use of **env** is to search for an interpreter of an interpreted-language script in the user's `PATH`:

```
#!/usr/bin/env bash
#!/usr/bin/env python3
```

This is problematic, however, as shebang lines only contain up to one argument, so extending

```
#!/usr/bin/bash -e
#!/usr/bin/python3 -uS
```

which may not work on other systems to

```
#!/usr/bin/env bash -e
#!/usr/bin/env python3 -uS
```

will invoke **env** with "**bash -e**" as the first argument — the executable to run — which won't work.

The **-S** option can be used to work around this:

```
#!/usr/bin/env -S bash -e
#!/usr/bin/env -S python3 -uS
```

This invokes **env** with "**-S bash -e**" as the first argument, which will then decompose into "**bash**" "**-e**". Indeed, the latter can also be written as

```
#!/usr/bin/env -S PYTHONUNBUFFERED=1 python3 -S
```

Mimick nohup(1), but without standard I/O stream processing:

```
$ env -LI HUP BLOCKSIZE=100k du -a / > everything &
env: SIGHUP       (Signal  1): ignored
```

STANDARDS

Conforms to IEEE Std 1003.1-2008 ("POSIX.1"), which defines the **-** syntax as unspecified. This implementation defines it compatibly with AT&T System III UNIX; avoid using it in new programs. The only standard option is **−i**.

−u is an extension, also present on the GNU system, FreeBSD 7.1, and NetBSD current.
−0 is an extension, also present on the GNU system, FreeBSD 12.2, and NetBSD current.
−v is an extension, also present on the GNU system and in a similar form on FreeBSD 5.5.
−C, **−−list−signal−handling**, **−−**{**default**|**ignore**|**block**}**−signal** and **−S** are extensions, originating from the GNU system. The parsing of **−S** is expected to be fully compatible.
Short **−DBIL** are extensions.

HISTORY

Appeared in CB-UNIX at or before version 2.1, fully formed, as

env [-] [name=value] ... [command args]

CB-UNIX was, among others, the basis for AT&T System III UNIX, where it first saw light outside AT&T.

4.3BSD–Reno cloned it as part of conformance with early IEEE Std 1003.2 ("POSIX.2") drafts, but even those specify **-** as obsolescent, defining a new **−i** flag instead, as **-** alone violates the Utility Syntax Guidelines.

IEEE Std 1003.1-2001 ("POSIX.1") removes **-**, but IEEE Std 1003.1-2008 ("POSIX.1") and later define it as unspecified to allow compatibility with the CB-UNIX behaviour — all known implementations support it, and the vast majority mark it as deprecated.

coreutils 2.31 (2019-03-10) invents **−−list−signal−handling**, **−−**{**default**|**ignore**|**block**}**−signal**, as present-day.