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 'nohup.1p' command

$ man nohup.1p

NOHUP(1P) POSIX Programmer's Manual NOHUP(1P)

PROLOG

This manual page is part of the POSIX Programmer's Manual. The Linux implementation of this interface may differ (consult the corresponding Linux manual page for details of Linux behavior), or the interface may not be implemented on Linux.

NAME

nohup ? invoke a utility immune to hangups

SYNOPSIS

nohup utility [argument...]

DESCRIPTION

The nohup utility shall invoke the utility named by the utility operand with arguments supplied as the argument operands. At the time the named utility is invoked, the SIGHUP signal shall be set to be ignored.

If standard input is associated with a terminal, the nohup utility may redirect standard input from an unspecified file.

If the standard output is a terminal, all output written by the named utility to its standard output shall be appended to the end of the file nohup.out in the current directory. If nohup.out cannot be created or opened for appending, the output shall be appended to the end of the file nohup.out in the directory specified by the HOME environment vari? able. If neither file can be created or opened for appending, utility shall not be invoked. If a file is created, the file's permission bits

shall be set to S_IRUSR | S_IWUSR.

If standard error is a terminal and standard output is open but is not a terminal, all output written by the named utility to its standard er? ror shall be redirected to the same open file description as the stan? dard output. If standard error is a terminal and standard output either is a terminal or is closed, the same output shall instead be appended to the end of the nohup.out file as described above.

OPTIONS

None.

OPERANDS

The following operands shall be supported:

utility The name of a utility that is to be invoked. If the utility operand names any of the special built-in utilities in Sec? tion 2.14, Special Built-In Utilities, the results are unde? fined.

argument Any string to be supplied as an argument when invoking the utility named by the utility operand.

STDIN

Not used.

INPUT FILES

None.

ENVIRONMENT VARIABLES

The following environment variables shall affect the execution of no? hup:

HOME Determine the pathname of the user's home directory: if the output file nohup.out cannot be created in the current direc? tory, the nohup utility shall use the directory named by HOME to create the file.

LANG Provide a default value for the internationalization vari? ables that are unset or null. (See the Base Definitions vol? ume of POSIX.1?2017, Section 8.2, Internationalization Vari? ables for the precedence of internationalization variables used to determine the values of locale categories.)

LC_ALL If set to a non-empty string value, override the values of all the other internationalization variables.

LC_CTYPE Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments).

LC_MESSAGES

Determine the locale that should be used to affect the format and contents of diagnostic messages written to standard er? ror.

NLSPATH Determine the location of message catalogs for the processing of LC_MESSAGES.

PATH Determine the search path that is used to locate the utility to be invoked. See the Base Definitions volume of POSIX.1?2017, Chapter 8, Environment Variables.

ASYNCHRONOUS EVENTS

The nohup utility shall take the standard action for all signals except that SIGHUP shall be ignored.

STDOUT

If the standard output is not a terminal, the standard output of nohup shall be the standard output generated by the execution of the utility specified by the operands. Otherwise, nothing shall be written to the standard output.

STDERR

If the standard output is a terminal, a message shall be written to the standard error, indicating the name of the file to which the output is being appended. The name of the file shall be either nohup.out or $HOME/nohup.out.

OUTPUT FILES

Output written by the named utility is appended to the file nohup.out (or $HOME/nohup.out), if the conditions hold as described in the DE? SCRIPTION.

EXTENDED DESCRIPTION

None.

EXIT STATUS

The following exit values shall be returned:

126 The utility specified by utility was found but could not be in? voked.

127 An error occurred in the nohup utility or the utility specified by utility could not be found.

Otherwise, the exit status of nohup shall be that of the utility speci? fied by the utility operand.

CONSEQUENCES OF ERRORS

Default.

The following sections are informative.

APPLICATION USAGE

The command, env, nice, nohup, time, and xargs utilities have been specified to use exit code 127 if an error occurs so that applications can distinguish ``failure to find a utility'' from ``invoked utility exited with an error indication''. The value 127 was chosen because it is not commonly used for other meanings; most utilities use small val? ues for ``normal error conditions'' and the values above 128 can be confused with termination due to receipt of a signal. The value 126 was chosen in a similar manner to indicate that the utility could be found, but not invoked. Some scripts produce meaningful error messages differ? entiating the 126 and 127 cases. The distinction between exit codes 126 and 127 is based on KornShell practice that uses 127 when all attempts to exec the utility fail with [ENOENT], and uses 126 when any attempt to exec the utility fails for any other reason.

EXAMPLES

It is frequently desirable to apply nohup to pipelines or lists of com? mands. This can be done by placing pipelines and command lists in a single file; this file can then be invoked as a utility, and the nohup applies to everything in the file.

Alternatively, the following command can be used to apply nohup to a complex command:

```
nohup sh -c 'complex-command-line' </dev/null
```

RATIONALE

The 4.3 BSD version ignores SIGTERM and SIGHUP, and if ./nohup.out can?
not be used, it fails instead of trying to use $HOME/nohup.out.
The csh utility has a built-in version of nohup that acts differently
from the nohup defined in this volume of POSIX.1?2017.
The term utility is used, rather than command, to highlight the fact
that shell compound commands, pipelines, special built-ins, and so on,
cannot be used directly. However, utility includes user application
programs and shell scripts, not just the standard utilities.
Historical versions of the nohup utility use default file creation se?
mantics. Some more recent versions use the permissions specified here
as an added security precaution.
Some historical implementations ignore SIGQUIT in addition to SIGHUP;
others ignore SIGTERM. An early proposal allowed, but did not require,
SIGQUIT to be ignored. Several reviewers objected that nohup should
only modify the handling of SIGHUP as required by this volume of
POSIX.1?2017.
Historical versions of nohup did not affect standard input, but that
causes problems in the common scenario where the user logs into a sys?
tem, types the command:

```
nohup make &
```

at the prompt, and then logs out. If standard input is not affected by
nohup, the login session may not terminate for quite some time, since
standard input remains open until make exits. To avoid this problem,
POSIX.1?2008 allows implementations to redirect standard input if it is
a terminal. Since the behavior is implementation-defined, portable ap?
plications that may run into the problem should redirect standard input
themselves. For example, instead of:

```
nohup make &
```

an application can invoke:

```
nohup make </dev/null &
```

FUTURE DIRECTIONS

None.

SEE ALSO

Chapter 2, Shell Command Language, sh

The Base Definitions volume of POSIX.1?2017, Chapter 8, Environment Variables

The System Interfaces volume of POSIX.1?2017, signal()