ubuntu manuals Menu

bionic (1) xe.1.gz

Provided by: xe_0.11-2_amd64

NAME

xe — execute a command for every argument

SYNOPSIS

```
xe [-0FLRnqv] [-I replace-arg] [-N maxargs] [-j maxjobs] command ...
xe [flags ...] -p pattern command ... [+ pattern command ...]...
xe [flags ...] -f argfile command ...
xe [flags ...] -s shellscript
xe [flags ...] -a command ... -- args ...
xe [flags ...] -A argsep command ... argsep args ...
```

DESCRIPTION

The **xe** utility constructs command lines from specified arguments, combining s
features of xargs(1) and apply(1).

xe means “execute for every ...”.

xe supports different methods to specify arguments to commands:

command ...

By default, arguments - separated by newlines - are read from the sta
The resulting command is constructed from the command line parameters
replace-arg with the read argument, and is executed with execvp(3).

In this mode, no shell is involved and replace-arg must appear as a wo
i.e. ‘foo {} bar’ will work, but ‘foo{} bar’ will not, where {} is t
value for replace-arg.

If no argument is specified, the default is ‘**printf %s\n**’.

-f argfile

Read arguments from argfile, instead of the standard input.

This does not close the standard input for execution, it is passed to
process.

-s shellscript

In this mode, the single parameter shellscript is executed using **sh -c**.
script, the specified arguments can be accessed using $1, $2, ...

For example:

```
echo 'a\nb' | xe -N2 -s 'echo $2 $1'
```

-a command ... **--** args ...

In this mode, everything after **--** is passed as args to command.

-A argsep command ... argsep args ...

Same as **-a**, but the custom argument separator argsep is used to distin
command and its args.

The options are as follows:

-0 Input filenames are separated by NUL bytes (instead of newlines, whic
default)

-F Fatal: stop and exit when a command execution fails.

-L Run the resulting commands with line-buffered output; lines from two
interleave. When used twice, or with **-vv**, also prefix each line with
the job (see ENVIRONMENT) in such a manner that the output can be pipe

-snk1' to group it.

-R Return with status 122 when no arguments have been specified (instead default). **xe** never executes a command when no arguments are specifie

-n Dry run: don't run the resulting commands, just print them.

-q Quiet mode: redirect standard output and standard error of commands t

-v Verbose: print commands to standard error before running them. When also print job id and exit status for each command.

-p Enable make(1)-style percent rules. The first argument of *command* *...* as a pattern, see *PERCENT* *RULES* below. Patterns without a slash (or ' matched against the basenames only.

Multiple runs of patterns and commands are separated by '+'. Only th matching percent rule is executed; in case no pattern matches, no com

-I *replace-arg*

Replace first occurrence of *replace-arg* (default: **{}**) in the resulting the argument(s). Pass an empty *replace-arg* to disable the replace fun Contrary to xargs(1) this will expand into multiple arguments when nee

-N *maxargs*

Pass up to *maxargs* arguments to each command (default: 1).
Using **-N0** will pass as many arguments as possible.

-j *maxjobs*

Run up to *maxjobs* processes concurrently. Using **-j0** will run as many there are CPU cores running. If *maxjobs* ends with an '**x**', it is regar multiplier of the number of running CPU cores (rounded down, but usin core).

PERCENT RULES

The percent rules of **xe** are similar to the globs of sh(1) or fnmatch(3): '?' m single character that is not '/'. '/' matches one or multiple '/' in the str matches zero or more characters, but never '/'. '**' matches zero or more ch including '/'. Note that all of these also match leading dots in file names.

'{*a*,*b*,*c*}' matches either *a*, *b* or *c*. '[*abc*]' matches one of the characters *abc*

‘/’). ‘[!*abc*]’ matches all characters but *abc*. Alternatively, ‘[^*abc*]’ can b
‘[*a*-*c*]’ matches any character in the range between *a* and *c* inclusive. In cha
characters can be escaped using a backslash.

In the pattern, a single occurrence of ‘%’ matches one or more characters, ar
first occurrence of ‘%’ with the matched string in the remaining arguments, w
used as the command to be executed.

ENVIRONMENT

The environment variable ITER is passed to the child process and incremented
execution.

EXIT STATUS

xe follows the convention of GNU and OpenBSD xargs:

0	on success
123	if any invocation of the command exited with status 1 to 125.
124	if the command exited with status 255
125	if the command was killed by a signal
126	if the command cannot be run
127	if the command was not found
1	if some other error occurred

Additionally, 122 is returned when **-R** was passed and the command was never ex

EXAMPLES

Compress all .c files in the current directory, using all CPU cores:

```
xe -a -j0 gzip -- *.c
```

Remove all empty files, using lr(1):

```
lr -U -t 'size == 0' | xe -N0 rm
```

Convert .mp3 to .ogg, using all CPU cores:

```
xe -a -j0 -s 'ffmpeg -i "${1}" "${1%.mp3}.ogg"' -- *.mp3
```

Same, using percent rules:

```
xe -a -j0 -p %.mp3 ffmpeg -i %.mp3 %.ogg -- *.mp3
```

Similar, but hiding output of ffmpeg, instead showing spawned jobs:

```
            xe -ap -j0 -vvq '%.{m4a,ogg,opus}' ffmpeg -y -i {} out/%.mp3 -- *
```

SEE ALSO

apply(1), parallel(1), xapply(1), xargs(1)

AUTHORS

Leah Neukirchen <leah@vuxu.org>

LICENSE

xe is in the public domain.

To the extent possible under law, the creator of this work has waived all cop
related or neighboring rights to this work.

http://creativecommons.org/publicdomain/zero/1.0/

Powered by the Ubuntu Manpage Repository, file bugs in Launchpad