NAME

flock – manage locks from shell scripts

SYNOPSIS

flock [options] *file*|*directory command* [*arguments*]
flock [options] *file*|*directory* **–c** *command*
flock [options] *number*

DESCRIPTION

This utility manages flock(2) locks from within shell scripts or from the command line.

The first and second of the above forms wrap the lock around the execution of a *command*, in a manner similar to su(1) or newgrp(1). They lock a specified *file* or *directory*, which is created (assuming appropriate permissions) if it does not already exist. By default, if the lock cannot be immediately acquired, **flock** waits until the lock is available.

The third form uses an open file by its file descriptor *number*. See the examples below for how that can be used.

OPTIONS

–c, **––command** *command*
Pass a single *command*, without arguments, to the shell with **–c**.

–E, **––conflict–exit–code** *number*
The exit code used when the **–n** option is in use, and the conflicting lock exists, or the **–w** option is in use, and the timeout is reached. The default value is **1**.

–F, **––no–fork**
Do not fork before executing *command*. Upon execution the flock process is replaced by *command* which continues to hold the lock. This option is incompatible with **––close** as there would otherwise be nothing left to hold the lock.

–e, **–x**, **––exclusive**
Obtain an exclusive lock, sometimes called a write lock. This is the default.

–n, **––nb**, **––nonblock**
Fail rather than wait if the lock cannot be immediately acquired. See the **–E** option for the exit code used.

–o, **––close**
Close the file descriptor on which the lock is held before executing *command*. This is useful if *command* spawns a child process which should not be holding the lock.

–s, **––shared**
Obtain a shared lock, sometimes called a read lock.

–u, **––unlock**
Drop a lock. This is usually not required, since a lock is automatically dropped when the file is closed. However, it may be required in special cases, for example if the enclosed command group may have forked a background process which should not be holding the lock.

–w, **––wait**, **––timeout** *seconds*
Fail if the lock cannot be acquired within *seconds*. Decimal fractional values are allowed. See the **–E** option for the exit code used. The zero number of *seconds* is interpreted as **––nonblock**.

––verbose
Report how long it took to acquire the lock, or why the lock could not be obtained.

–V, **––version**
Display version information and exit.

–h, **––help**
Display help text and exit.

EXAMPLES

shell1> flock /tmp -c cat
shell2> flock -w .007 /tmp -c echo; /bin/echo $?
Set exclusive lock to directory /tmp and the second command will fail.

shell1> flock -s /tmp -c cat
shell2> flock -s -w .007 /tmp -c echo; /bin/echo $?
Set shared lock to directory /tmp and the second command will not fail. Notice that attempting to get exclusive lock with second command would fail.

shell> flock -x local-lock-file echo 'a b c'
Grab the exclusive lock "local-lock-file" before running echo with 'a b c'.

(
flock -n 9 || exit 1
... commands executed under lock ...
) 9>/var/lock/mylockfile
The form is convenient inside shell scripts. The mode used to open the file doesn't matter to **flock**; using > or >> allows the lockfile to be created if it does not already exist, however, write permission is required. Using < requires that the file already exists but only read permission is required.

["${FLOCKER}" != "$0"] && exec env FLOCKER="$0" flock -en "$0" "$0" "$@" || :
This is useful boilerplate code for shell scripts. Put it at the top of the shell script you want to lock and it'll automatically lock itself on the first run. If the env var $FLOCKER is not set to the shell script that is being run, then execute flock and grab an exclusive non-blocking lock (using the script itself as the lock file) before re-execing itself with the right arguments. It also sets the FLOCKER env var to the right value so it doesn't run again.

EXIT STATUS

The command uses **sysexits.h** return values for everything, except when using either of the options **−n** or **−w** which report a failure to acquire the lock with a return value given by the **−E** option, or 1 by default.

When using the *command* variant, and executing the child worked, then the exit status is that of the child command.

AUTHOR

H. Peter Anvin

COPYRIGHT

SEE ALSO

flock(2)

AVAILABILITY

The flock command is part of the util-linux package and is available from Linux Kernel Archive.