Rocky Enterprise Linux 9.2 Manual Pages on command 'Lintian::Command::Simple.3'

C:\>man Lintian::Command::Simple.3

Lintian::Command::Simple(3) Debian Package Checker Lintian::Command::Simple(3)

NAME

Lintian::Command::Simple - Run commands without pipes

SYNOPSIS

```
use Lintian::Command::Simple qw(wait_any);
my %pid_info;
my $pid = fork() // die("fork: $!");
exec('do', 'something') if $pid == 0;
$pid_info{$pid} = "A useful value associated with $pid";
my ($termiated_pid, $value) = wait_any(\%pid_info);
...;
```

DESCRIPTION

Lintian::Command::Simple allows running commands with the capability of running them "in the background" (asynchronously.)

Pipes are not handled at all, except for those handled internally by the shell. See 'perldoc -f exec's note about shell metacharacters. If you want to pipe to/from Perl, look at Lintian::Command instead.

wait_any (hashref[, nohang])

When starting multiple processes asynchronously, it is common to wait until the first is done. While the CORE::wait() function is usually used for that very purpose, it does not provide the desired results when the processes were started via the OO interface.

To help with this task, wait_any() can take a hash ref where the key of each entry is the pid of that command. There are no requirements for the value (which can be used for any application specific purpose).

Under this mode, wait_any() waits until any child process is done. The key (and value) associated the pid of the reaped child will then be removed from the hashref. The exitcode of the child is available via $? as usual.

The results and return value are undefined when under this mode wait_any() "accidentally" reaps a process not listed in the hashref.

The return value in scalar context is value associated with the pid of the reaped processed. In list context, the pid and value are returned as a pair.

Whenever waitpid() would return -1, wait_any() returns undef or a null value so that it is safe to:

```
while($cmd = wait_any(\%hash)) { something; }
```

The same is true whenever the hash reference points to an empty hash.

If "nohang" is also given, wait_any will attempt to reap any child process non-blockingly. If no child can be reaped, it will immediately return (like there were no more processes left) instead of waiting.

kill_all(hashref[, signal])

In a similar way to wait_any(), it is possible to pass a hash reference to kill_all(). It will then kill all of the processes (default signal being "TERM") followed by a reaping of the processes. All reaped processes (and their values) will be removed from the set.

Any entries remaining in the hashref are processes that did not terminate (or did not terminate yet).

NOTES

Unless specified by prefixing the package name, every reference to a function/method in this documentation refers to the functions/methods provided by this package itself.

CAVEATS

Combining asynchronous jobs (e.g. via Lintian::Command) and calls to wait_any() can lead to unexpected results.

AUTHOR

Originally written by Raphael Geissert <atomo64@gmail.com> for Lintian.