



Full credit is given to the above companies including the Operating System (OS) that this PDF file was generated!

Rocky Enterprise Linux 9.2 Manual Pages on command 'Dpkg::Compression::Process.3perl'

\$ man Dpkg::Compression::Process.3perl

Dpkg::Compression::Process(3perl) libdpkg-perl Dpkg::Compression::Process(3perl)

NAME

Dpkg::Compression::Process - run compression/decompression processes

DESCRIPTION

This module provides an object oriented interface to run and manage compression/decompression processes.

METHODS

`$proc = Dpkg::Compression::Process->new(%opts)`

Create a new instance of the object. Supported options are "compression" and "compression_level" (see corresponding set_* functions).

`$proc->set_compression($comp)`

Select the compression method to use. It errors out if the method is not supported according to "compression_is_supported" (of Dpkg::Compression).

`$proc->set_compression_level($level)`

Select the compression level to use. It errors out if the level is not valid according to "compression_is_valid_level" (of Dpkg::Compression).

`@exec = $proc->get_compress_cmdline()`

`@exec = $proc->get_uncompress_cmdline()`

Returns a list ready to be passed to "exec", its first element is the program name (either for compression or decompression) and the following elements are parameters for the program.

When executed the program acts as a filter between its standard input and its standard output.

`$proc->compress(%opts)`

Starts a compressor program. You must indicate where it will read its uncompressed data from and where it will write its compressed data to. This is accomplished by passing one parameter "to_*" and one parameter "from_*" as accepted by `Dpkg::IPC::spawn`.

You must call "wait_end_process" after having called this method to properly close the sub-process (and verify that it exited without error).

`$proc->uncompress(%opts)`

Starts a decompressor program. You must indicate where it will read its compressed data from and where it will write its uncompressed data to. This is accomplished by passing one parameter "to_*" and one parameter "from_*" as accepted by `Dpkg::IPC::spawn`.

You must call "wait_end_process" after having called this method to properly close the sub-process (and verify that it exited without error).

`$proc->wait_end_process(%opts)`

Call `Dpkg::IPC::wait_child` to wait until the sub-process has exited and verify its return code. Any given option will be forwarded to the "wait_child" function. Most notably you can use the "nocheck" option to verify the return code yourself instead of letting "wait_child" do it for you.

CHANGES

Version 1.00 (dpkg 1.15.6)

Mark the module as public.

1.21.1 2024-02-23 `Dpkg::Compression::Process(3perl)`