



Rocky Enterprise Linux 9.2 Manual Pages on command 'Dpkg::Vendor::Default.3perl'

C:\>man Dpkg::Vendor::Default.3perl

Dpkg::Vendor::Default(3perl) libdpkg-perl Dpkg::Vendor::Default(3perl)

NAME

Dpkg::Vendor::Default - default vendor object

DESCRIPTION

A vendor object is used to provide vendor specific behaviour in various places.

This is the default object used in case there's none for the current vendor or in case the vendor could not be identified (see Dpkg::Vendor documentation).

It provides some hooks that are called by various dpkg-* tools. If you need a new hook, please file a bug against dpkg-dev and explain your need. Note that the hook API has no guarantee to be stable over an extended period of time. If you run an important distribution that makes use of vendor hooks, you'd better submit them for integration so that we avoid breaking your code.

METHODS

`$vendor_obj = Dpkg::Vendor::Default->new()`

Creates the default vendor object. Can be inherited by all vendor objects if they don't need any specific initialization at object creation time.

`$vendor_obj->run_hook($id, @params)`

Run the corresponding hook. The parameters are hook-specific. The supported hooks are:

before-source-build (\$srcpkg)

The first parameter is a `Dpkg::Source::Package` object. The hook is called just before the execution of `$srcpkg->build()`.

package-keyrings ()

The hook is called when `dpkg-source` is checking a signature on a source package (since `dpkg 1.18.11`). It takes no parameters, but returns a (possibly empty) list of vendor-specific keyrings.

archive-keyrings ()

The hook is called when there is a need to check signatures on artifacts from repositories, for example by a download method (since `dpkg 1.18.11`). It takes no parameters, but returns a (possibly empty) list of vendor-specific keyrings.

archive-keyrings-historic ()

The hook is called when there is a need to check signatures on artifacts from historic repositories, for example by a download method (since `dpkg 1.18.11`). It takes no parameters, but returns a (possibly empty) list of vendor-specific keyrings.

builtin-build-depends ()

The hook is called when `dpkg-checkbuilddeps` is initializing the source package build dependencies (since `dpkg 1.18.2`). It takes no parameters, but returns a (possibly empty) list of vendor-specific Build-Depends.

builtin-build-conflicts ()

The hook is called when `dpkg-checkbuilddeps` is initializing the source package build conflicts (since `dpkg 1.18.2`). It takes no parameters, but returns a (possibly empty) list of vendor-specific Build-Conflicts.

register-custom-fields ()

The hook is called in `Dpkg::Control::Fields` to register custom fields.

You should return a list of arrays. Each array is an operation to perform. The first item is the name of the operation and corresponds to a `field_*` function provided by `Dpkg::Control::Fields`. The remaining fields are the parameters that are passed unchanged to the corresponding function.

Known operations are "register", "insert_after" and "insert_before".

post-process-changelog-entry (\$fields)

The hook is called in `Dpkg::Changelog` to post-process a `Dpkg::Changelog::Entry` after it has been created and filled with the appropriate values.

update-buildflags (\$flags)

The hook is called in `Dpkg::BuildFlags` to allow the vendor to override the default values set for the various build flags. `$flags` is a `Dpkg::BuildFlags` object.

builtin-system-build-paths ()

The hook is called by `dpkg-genbuildinfo` to determine if the current path should be recorded in the Build-Path field (since `dpkg 1.18.11`).

It takes no parameters, but returns a (possibly empty) list of root paths considered acceptable. As an example, if the list contains `"/build/"`, a Build-Path field will be created if the current directory is `"/build/dpkg-1.18.0"`. If the list contains `"/"`, the path will always be recorded. If the list is empty, the current path will never be recorded.

build-tainted-by ()

The hook is called by `dpkg-genbuildinfo` to determine if the current

system has been tainted in some way that could affect the resulting build, which will be recorded in the Build-Tainted-By field (since dpkg 1.19.5). It takes no parameters, but returns a (possibly empty) list of tainted reason tags (formed by alphanumeric and dash characters).

CHANGES

Version 0.xx

This is a private module.

1.19.7 2022-05-25 Dpkg::Vendor::Default(3perl)