NAME

Dpkg::Path – some common path handling functions

DESCRIPTION

It provides some functions to handle various path.

FUNCTIONS

get_pkg_root_dir($file)

This function will scan upwards the hierarchy of directory to find out the directory which contains the "DEBIAN" sub-directory and it will return its path. This directory is the root directory of a package being built.

If no DEBIAN subdirectory is found, it will return undef.

relative_to_pkg_root($file)

Returns the filename relative to get_pkg_root_dir($file).

guess_pkg_root_dir($file)

This function tries to guess the root directory of the package build tree. It will first use **get_pkg_root_dir()**, but it will fallback to a more imprecise check: namely it will use the parent directory that is a sub-directory of the debian directory.

It can still return undef if a file outside of the debian sub-directory is provided.

check_files_are_the_same($file1, `$file2`, `$resolve_symlink`)

This function verifies that both files are the same by checking that the device numbers and the inode numbers returned by **stat()/lstat()** are the same. If `$resolve_symlink` is true then **stat()** is used, otherwise **lstat()** is used.

canonpath($file)

This function returns a cleaned path. It simplifies double //, and remove /./ and /../ intelligently. For /../ it simplifies the path only if the previous element is not a symlink. Thus it should only be used on real filenames.

`$newpath` = resolve_symlink($symlink)

Return the filename of the file pointed by the symlink. The new name is canonicalized by **canonpath()**.

`$cmdpath` = find_command($command)

Return the path of the command if defined and available on an absolute or relative path or on the `$PATH`, undef otherwise.

`$control_file` = get_control_path($pkg, `$filetype`)

Return the path of the control file of type `$filetype` for the given package.

`@control_files` = get_control_path($pkg)

Return the path of all available control files for the given package.

`$file` = find_build_file($basename)

Selects the right variant of the given file: the arch-specific variant ("$basename.$arch") has priority over the OS-specific variant ("$basename.$os") which has priority over the default variant ("$basename"). If none of the files exists, then it returns undef.

`@files` = find_build_file($basename)

Return the available variants of the given file. Returns an empty list if none of the files exists.

CHANGES

Version 1.04 (dpkg 1.17.11)

Update semantics: **find_command()** now handles an empty or undef argument.

Version 1.03 (dpkg 1.16.1)

New function: **find_build_file()**

Version 1.02 (dpkg 1.16.0)
New function: **get_control_path**()

Version 1.01 (dpkg 1.15.8)
New function: **find_command**()

Version 1.00 (dpkg 1.15.6)
Mark the module as public.