Rocky Enterprise Linux 9.2 Manual Pages on command 'pc.5'

$ man pc.5

PC(5) BSD File Formats Manual PC(5)

NAME

file.pc ? pkg-config file format

DESCRIPTION

pkg-config files provide a useful mechanism for storing various informa? tion about libraries and packages on a given system. Information stored by .pc files include compiler and linker flags necessary to use a given library, as well as any other relevant metadata.

These .pc files are processed by a utility called pkg-config, of which pkgconf is an implementation.

FILE SYNTAX

The .pc file follows a format inspired by RFC822. Comments are prefixed by a pound sign, hash sign or octothorpe (#), and variable assignment is similar to POSIX shell. Properties are defined using RFC822-style stan? zas.

VARIABLES

Variable definitions start with an alphanumeric string, followed by an equal sign, and then the value the variable should contain.

Variable references are always written as "${variable}". It is possible to escape literal "${" as "$${".

PROPERTIES

Properties are set using RFC822-style stanzas which consist of a keyword, followed by a colon (:) and then the value the property should be set to. Variable substitution is always performed regardless of property type.

There are three types of property:

Literal

The property will be set to the text of the value.

Dependency List

The property will be set to a list of dependencies parsed from the text. Dependency lists are defined by this ABNF syntax:

package-list = *WSP *(package-spec *(package-sep))

package-sep = WSP / ","

package-spec = package-key [ver-op package-version]

ver-op = "<" / "<=" / "=" / "!=" / ">=" / ">"

Fragment List

The property will be set to a list of fragments parsed from the text. The input text must be in a format that is suitable for passing to a POSIX shell without any shell expansions after vari? able substitution has been done.

PROPERTY KEYWORDS

Name The displayed name of the package. (mandatory; literal)

Version

The version of the package. (mandatory; literal)

Description

A description of the package. (mandatory; literal)

URL A URL to a webpage for the package. This is used to recommend where newer versions of the package can be acquired. (mandatory; literal)

Cflags Required compiler flags. These flags are always used, regardless of whether static compilation is requested. (optional; fragment list)

Cflags.private

Required compiler flags for static compilation. (optional; frag? ment list; pkgconf extension)

Libs Required linking flags for this package. Libraries this package depends on for linking against it, which are not described as de? pendencies should be specified here. (optional; fragment list)

Libs.private

Required linking flags for this package that are only required when linking statically. Libraries this package depends on for linking against it statically, which are not described as depen? dencies should be specified here. (optional; fragment list)

Requires

Required dependencies that must be met for the package to be us? able. All dependencies must be satisfied or the pkg-config im? plementation must not use the package. (optional; dependency list)

Requires.private

Required dependencies that must be met for the package to be us? able for static linking. All dependencies must be satisfied or the pkg-config implementation must not use the package for static linking. (optional; dependency list)

Conflicts

Dependencies that must not be met for the package to be usable. If any package in the proposed dependency solution match any de? pendency in the Conflicts list, the package being considered is not usable. (optional; dependency list)

Provides

Dependencies that may be provided by an alternate package. If a package cannot be found, the entire package collection is scanned for providers which can match the requested dependency. (op? tional; dependency list; pkgconf extension)

EXTENSIONS

Features that have been marked as a pkgconf extension are only guaranteed

to work with the pkgconf implementation of pkg-config. Other implementa?

tions may or may not support the extensions.

Accordingly, it is suggested that .pc files which absolutely depend on

these extensions declare a requirement on the pkgconf virtual.

EXAMPLES

An example .pc file:

```
# This is a comment
prefix=/home/kaniini/pkg   # this defines a variable
exec_prefix=${prefix}      # defining another variable with a substitution
libdir=${exec_prefix}/lib
includedir=${prefix}/include
Name: libfoo                          # human-readable name
Description: an example library called libfoo # human-readable description
Version: 1.0
URL: http://www.pkgconf.org
Requires: libbar > 2.0.0
Conflicts: libbaz <= 3.0.0
Libs: -L${libdir} -lfoo
Libs.private: -lm
Cflags: -I${includedir}/libfoo
```

SEE ALSO

pkgconf(1), pkg.m4(7)

BSD December 15, 2017 BSD