NAME

deb−symbols − Debian’s extended shared library information file

SYNOPSIS

DEBIAN/symbols

DESCRIPTION

The symbol files are shipped in Debian binary packages, and its format is a subset of the template symbol files used by **dpkg-gensymbols**(1) in Debian source packages.

The format for an extended shared library dependency information entry in these files is:

library-soname main-dependency-template
[| *alternative-dependency-template*]
[...]
[* *field-name*: *field-value*]
[...]
symbol minimal-version [*id-of-dependency-template*]

The *library-soname* is exactly the value of the SONAME field as exported by **objdump**(1). A *dependency-template* is a dependency where *#MINVER#* is dynamically replaced either by a version check like “(>= *minimal-version*)” or by nothing (if an unversioned dependency is deemed sufficient).

Each exported *symbol* (listed as *name@version*, with *version* being “Base” if the library is not versioned) is associated to a *minimal-version* of its dependency template (the main dependency template is always used and will end up being combined with the dependency template referenced by *id-of-dependency-template* if present). The first alternative dependency template is numbered 1, the second one 2, etc. Each column is separated by exactly a single whitespace.

Each entry for a library can also have some fields of meta-information. Those fields are stored on lines starting with an asterisk. Currently, the only valid fields are:

Build-Depends-Package
: It indicates the name of the “−dev” package associated to the library and is used by**dpkg-shlibdeps** to make sure that the dependency generated is at least as strict as the corresponding build dependency (since dpkg 1.14.13).

Build-Depends-Packages
: The same as **Build-Depends-Package** but accepts a comma-separated list of package names (since dpkg 1.20.0). This field will override any **Build-Depends-Package** field present, and is mostly useful with “−dev” packages and metapackages depending on these, say for a transition period.

Allow-Internal-Symbol-Groups
: It indicates what internal symbol groups should be ignored, as a whitespace separated list, so that the symbols contained in those groups get included in the output file (since dpkg 1.20.1). This should only be necessary for toolchain packages providing those internal symbols. The available groups are system dependent, for ELF and GNU-based systems these are **aeabi** and **gomp**.

Ignore-Blacklist-Groups
: A deprecated alias for **Allow-Internal-Symbol-Groups** (since dpkg 1.20.1, supported since dpkg 1.17.6).

EXAMPLES

Simple symbols file

```
libftp.so.3 libftp3 #MINVER#
 DefaultNetbuf@Base 3.1−1−6
 FtpAccess@Base 3.1−1−6
 [...]
```

Advanced symbols file

```
libGL.so.1 libgl1
| libgl1-mesa-glx #MINVER#
* Build-Depends-Package: libgl1-mesa-dev
 publicGlSymbol@Base 6.3-1
 [...]
 implementationSpecificSymbol@Base 6.5.2-7 1
 [...]
```

SEE ALSO

<https://wiki.debian.org/Projects/ImprovedDpkgShlibdeps>, **dpkg-shlibdeps**(1), **dpkg-gensymbols**(1).