NAME

dh_pypy − calculates PyPy dependencies, adds maintainer scripts to byte compile files, etc.

SYNOPSIS

dh_pypy −p PACKAGE [−V [X.Y][−][A.B]] DIR [−X REGEXPR]

DESCRIPTION

QUICK GUIDE FOR MAINTAINERS

- build−depend on pypy and dh−python,
- add *${pypy:Depends}* to Depends
- build module/application using its standard build system,
- install files to the standard locations,
- add *pypy* to dh's −−with option, or:
- call **dh_pypy** in the *binary−** target,

NOTES

dependencies

dh_pypy tries to translate Python dependencies from the *requires.txt* file to Debian dependencies. In many cases, this works without any additional configuration because dh_pypy comes with a build−in mapping of Python module names to Debian packages that is periodically regenerated from the Debian archive. By default, the version information in the Python dependencies is discarded. If you want dh_pypy to generate more strict dependencies (e.g. to avoid ABI problems), or if the automatic mapping does not work correctly for your package, you have to provide dh_pypy with additional rules for the translation of Python module to Debian package dependencies.

For a package *pypy−foo* that depends on a package *pypy−bar*, there are two files that may provide such rules:

1. If the *pypy−foo* source package ships with a *debian/pypy−overrides* file, this file is used by dh_pypy during the build of *pypy−foo*.
2. If the *pypy−bar* source package ships with a *debian/pypy−bar.pydist* file (and uses dh_pypy), this file will be included in the binary package as */usr/share/dh−python/dist/pypy/pypy−bar*. During the build of *pypy−foo*, dh_pypy will then find and use the file.

Both files have the same format described in */usr/share/doc/dh−python/README.PyDist*. If all you want is to generate versioned dependencies (and assuming that the *pypy−bar* package provides the *pybar* Python module), in most cases it will be sufficient to put the line **pybar pypy−bar; PEP386** into either of the above files.

namespace feature

dh_pypy parses Egg's namespace_packages.txt files (in addition to −−namespace command line argument(s)) and drops empty __init__.py files from binary package. pypycompile will regenerate them at install time and pypyclean will remove them at uninstall time (if they're no longer used in installed packages). It's still a good idea to provide __init__.py file in one of binary packages (even if all other packages use this feature).

private dirs

/usr/share/foo, */usr/share/games/foo*, */usr/lib/foo* and */usr/lib/games/foo* private directories are scanned for Python files by default (where *foo* is binary package name). If your package ships Python files in some other directory, add another dh_pypy call in debian/rules with directory name as an argument − you can use different set of options in this call. If you need to change options for a private directory that is checked by default, invoke dh_pypy with −−skip−private option and add another call with a path to this directory and new options.

debug packages
In binary packages which name ends with *−dbg*, all files in */usr/lib/pypy/dist−packages/* directory that have extensions different than *so* or *h* are removed by default. Use −−no−dbg−cleaning option to disable this feature.

overriding supported / default PyPy versions
If you want to override system's list of supported PyPy versions or the default one (f.e. to build a package that includes symlinks for older version of PyPy or compile .py files only for given interpreter version), you can do that via *DEBPYPY_SUPPORTED* and/or *DEBPYPY_DEFAULT* env. variables.

OPTIONS

−−version
show program's version number and exit

−h, **−−help**
show help message and exit

−−no−guessing−deps
disable guessing dependencies

−−no−dbg−cleaning
do not remove any files from debug packages

−−no−ext−rename do not add magic tags nor multiarch tuples to extension file names

−−no−shebang−rewrite
do not rewrite shebangs

−−skip−private
don't check private directories

−v, **−−verbose**
turn verbose mode on

−i, **−−indep**
act on architecture independent packages

−a, **−−arch**
act on architecture dependent packages

−q, **−−quiet**
be quiet

−p *PACKAGE*, **−−package=***PACKAGE*
act on the package named PACKAGE

−N *NO_PACKAGE*, **−−no−package=***NO_PACKAGE*
do not act on the specified package

−X *REGEXPR*, **−−exclude=***REGEXPR*
exclude items that match given REGEXPR. You may use this option multiple times to build up a list of things to exclude.

−−compile−all
compile all files from given private directory in postinst/rtupdate not just the ones provided by the package (i.e. do not pass the −−package parameter to py3compile/py3clean)

−−accept−upstream−versions
accept upstream versions while translating Python dependencies into Debian ones

−−depends=*DEPENDS*
translate given requirements into Debian dependencies and add them to ${pypy:Depends}. Use it for missing items in requires.txt

−−depends−sections=*SECTIONS*
translate requirements from given sections of requres.txt file into Debian dependencies and add them to ${pypy:Depends}.

−−recommends=*RECOMMENDS*
translate given requirements into Debian dependencies and add them to ${pypy:Recommends}

−−recommends−sections=*SECTIONS*
translate requirements from given sections of requres.txt file into Debian dependencies and add them to ${pypy:Recommends}.

−−suggests=*SUGGESTS*
translate given requirements into Debian dependencies and add them to ${pypy:Suggests}

−−suggests−sections=*SECTIONS*
translate requirements from given sections of requres.txt file into Debian dependencies and add them to ${pypy:Suggests}.

−−requires=*FILENAME*
translate requirements from given file(s) into Debian dependencies and add them to ${pypy:Depends}

−−shebang=*COMMAND*
use given command as shebang in scripts

−−ignore−shebangs
do not translate shebangs into Debian dependencies

SEE ALSO

- /usr/share/doc/dh−python/README.PyDist
- pybuild(1)
- *http://deb.li/dhpy* − most recent version of this document

AUTHOR

Piotr Oarowski, 2013