Maintaining Debian Packages

Stefan Hornburg (Racke)

Contents

Building from Git

More information in the Debian Wiki.

```
gbp buildpackage
```

To use the command please install *git-buildpackage* package first.
Also setup the following environment variables (e.g. in `~/.bashrc`):

```
export DEBEMAIL=''racke@linuxia.de''
export DEBFULLNAME=''Stefan Hornburg (Racke)''
```

Invoking lintian after build

```
gbp buildpackage --git-postbuild='lintian -i $GBP_CHANGES_FILE'
```

This can be also configured in the configuration file `~/.gbp.conf`:

```
[DEFAULT]
postbuild=lintian -i $GBP_CHANGES_FILE
```

Note: `-i` shows detailed information about the lintian warnings and errors.

Build for release

With Git buildpackage command:

```
gbp buildpackage --git-pristine-tar --git-tag --changes-option=-S
```

With standard buildpackage command:

```
dpkg-buildpackage --changes-option=-S
```

The `-S` option enables source-only upload.

Integrating upstream release

Import the sources:

```
$ gbp import-orig --uscan
gbp:info: Launching uscan...
gbp:info: Using uscan downloaded tarball ../pure-ftpd_1.0.50.orig.tar.bz2
What is the upstream version? [1.0.50]
gbp:info: Importing '../pure-ftpd_1.0.50.orig.tar.bz2' to branch 'upstream'...
gbp:info: Source package is pure-ftpd
gbp:info: Upstream version is 1.0.50
gbp:info: Replacing upstream source on 'master'
gbp:info: Successfully imported version 1.0.50 of ../pure-ftpd_1.0.50.orig.tar.bz2
```

If you use a DFSG repacked source:

```
$ gbp import-orig --uscan --pristine-tar --filter-pristine-tar
```

Also you need to make sure that you enter the appropriate upstream version at the prompt:

```
gbp:info: Launching uscan...
uscan: Newest version of sympa on remote site is 6.2.22, local version is 6.2.20~dfsg
 (mangled local version is 6.2.20)
uscan:    => Newer package available from
      https://github.com/sympa-community/sympa/archive/6.2.22.tar.gz
gbp:info: using ../sympa_6.2.22.orig.tar.gz
What is the upstream version? [6.2.22] 6.2.22~dfsg
```

Add entry to `debian/changelog`:

```
sympa (6.2.66~dfsg-1) unstable; urgency=medium

  * New upstream release.

 -- Stefan Hornburg (Racke) <racke@linuxia.de>  Sat, 27 Nov 2021 14:26:58 +0100
```

Build:

```
$ gbp buildpackage
```

Branches

Dedicated branches are used for backports and also for bug fix releases while working on packaging a new upstream release.

It is recommended to use the release in the branch name, e.g. *debian/buster* or *debian/buster-backports*.

If you build from a dedicated branch, you need to tell that to *gbp*:

```
$ gbp buildpackage --git-debian-branch=debian/buster
```

Backports

Create a dedicated branch if you start a backport.

```
$ git checkout -b debian/stretch-backports
```

Otherwise update the branch:

```
$ git checkout debian/stretch-backports
$ git merge debian/6.2.24_dfsg-1
```

Resolve conflict in `debian/changelog`.

Build the package.

Security bugs and uploads

Reference: https://www.debian.org/doc/manuals/developers-reference/pkgs.html#handling-security-relat

Building existing package from source

Install apt-src:

```
$ sudo apt install apt-src
```

Ensure that you have the proper sources in `/etc/apt/sources.list`, e.g.

```
deb-src http://ftp.de.debian.org/debian/ bullseye main
```

Create a directory and switch to it:

```
$ mkdir -p ~/debian/sympa
$ cd ~/debian/sympa
```

Retrieve source code

```
$ apt-src install sympa
```

Install build dependencies

```
$ sudo apt build-dep sympa
```

Build the package:

```
$ dpkg-buildpackage
```

Upload packages to Debian

We are using the `dput` command for that. For a normal upload you need only pass the name of the changes file:

```
$ dput ../sympa_6.2.66~dfsg-2_amd64.changes
```

A NMU (non-maintainer upload) will be uploaded to the delayed queue:

```
$ dput --delayed=10 ../mhonarc_2.6.19-2.2_amd64.changes
```

Files in `debian` directory

`changelog`

You can add a changelog entry based on your recent commit messages when you are using `gbp`.

```
$ gbp dch
gbp:info: Found tag for topmost changelog version 'd9594474c2e98ea3165c0072427fa2f658caff5d'
```

`compat`

Specifies the debhelper compatibility level.
We recommend to use `13` as this level is supported in *buster-backports*.
You also need a corresponding entry in the `control` file:

```
Build-Depends: debhelper (>= 13), ...
```

`NEWS`

This file is designated for **important changes** in a package.
The format is almost the same as for the `changelog` file, without the asterisks in the content.

```
pure-ftpd (1.0.50-1) unstable; urgency=medium

  Support for MD5, SHA1 and MySQL PASSWORD() has been removed from password hashing.
  Please use scrypt, argon2 or the system crypt(3).

  The SPSV command has been removed.

 -- Stefan Hornburg (Racke) <racke@linuxia.de>  Sun, 28 Nov 2021 18:11:06 +0100
```

package`.default`

File with shell variables, which can be used in scripts or in systemd units. Add *package* file to `/etc/default/`.

package`.dirs`

List of directories to create which are not installed by the build process.

package`.docs`

List of files to be installed into the `/usr/share/doc/*package*` directory.

package.install

Map of files and directories not installed by the build process for the package.

package.links

List of symlinks to be added. Symlinks to a directory are not supported.

package.logrotate

Configuration file for logrotate tool. Adds *package* file to `/etc/logrotate.d`.

If you want to deviate from the default filename /etc/logrotate.d/*package*, rename the file to *package.myname*.logrotate and and add the following override to *rules*:

```
override_dh_installlogrotate:
    dh_installlogrotate --name myname
```

package.pam

Adds *package* file to `/etc/pam.d`.

If you want to put the PAM file under a different name, rename the file to *package.myname*.pam and add the following override to *rules*:

```
override_dh_installpam:
    dh_installpam --name myname
```

po

This directory contains the translations for the Debconf templates.

rules

systemd

Installing systemd services where the main service triggers start of the other services through *Wants* directive:

```
override_dh_installsystemd:
    dh_installsystemd --name sympa sympa.service
    dh_installsystemd --no-start --name sympa-bounced sympa-bounced.service
    dh_installsystemd --no-start --name sympa-archived sympa-archived.service
    dh_installsystemd --no-start --name sympa-bulk sympa-bulk.service
    dh_installsystemd --no-start --name sympa-task_manager sympa-task_manager.service
```

source/format

Indicates the source format.
Contents for regular package:

```
3.0 (quilt)
```

Contents for native package:

```
3.0 (native)
```

templates

This file contains the templates for Debconf.

watch

Contains specification for detecting new upstream releases.
Can be tested as follows:

```
uscan --no-download --verbose
```

Reference: `https://wiki.debian.org/debian/watch`

Quilt

How to use quilt to manage patches in Debian packages

You need to **switch the current directory** before executing quilt commands, otherwise the patches will end up in the wrong place.

```
$ cd debian/patches
```

You can also configure the directory for the patches in `~/.quiltrc`:

```
QUILT_PATCHES=debian/patches
```

Install quilt

```
$ sudo apt install quilt
```

Import existing patch

```
quilt import ~/downloads/tls1.3.patch
```

Update patch

Upstream changed a patch which already exists in `debian/series`.

In order to update the patch, follow this example:

```
$ quilt push backtick-syntax-1087.diff
Applying patch backtick-syntax-1087.diff
patching file src/lib/Conf.pm

Now at patch backtick-syntax-1087.diff
$ git checkout src/lib/Conf.pm
Updated 1 path from the index
$ patch -p 1 < backtick-syntax-1087.diff
patching file src/lib/Conf.pm
$ quilt refresh
Refreshed patch backtick-syntax-1087.diff
$ quilt pop
Removing patch backtick-syntax-1087.diff
Restoring src/lib/Conf.pm

No patches applied
$ git add debian/patches/backtick-syntax-1087.diff
$ git commit -m ''Update backtick-syntax patch from upstream.''
[master 36760b9] Update backtick-syntax patch from upstream.
 1 file changed, 6 insertions(+), 8 deletions(-)
```

Remove patch

You can remove your patch `tls1.3.patch` in case it has been incorporated into a new upstream release.

```
quilt delete -r tls1.3.patch
```

Example for corresponding git commit:

```
git add debian/patches
git commit -m ''Remove TLS1.3 compatibility fix which is included in the 1.0.48 release.''
```

Debconf template translations

- Add or update the PO file (e.g. `sv.po` fo Swedish) in `debian/po` directory
- Run `debconf-updatepo` program as the translator may have used an outdate PO file

Non-maintainer upload (NMU)

In a nutshell, do the following steps:

- Fix bug with minimal changes
- Use version number with extra digit (2.6.19-2.1 as NMU for 2.6.19-2)
- Add changelog entry ”Non-maintainer upload” or ”NMU”
- Build the package
- Use `nmudiff` to update bug report
- Upload package to delayed queue

Manipulating bugs through email

Replace *xxxxxx* with the actual bug number in the following examples and send these in the body of an email to `control@bugs.debian.org`.

Reference: `https://www.debian.org/Bugs/server-control`

To close a bug, you can write an email to *xxxxxx*`-done@bugs.debian.org`. It is recommended to add a pseudo header with the package version that fixed the bug:

```
Version: 6.2.18~dfsg-1
```

Mark bug as confirmed

```
tags xxxxxx + confirmed
```

Change title of the bug

```
retitle xxxxxx Foo is not Bar
```

Other versions

```
found 939636 6.2.40~dfsg-2
```

Add usertags for Bug squashing party

```
user debian-release@lists.debian.org
usertag xxxxxx  + bsp-2018-12-ch-bern
```

pbuilder

Setup pbuilder for a specific distribution:

```
DIST=jessie git-pbuilder create
```

Building for a specific distribution:

```
gbp buildpackage --git-pbuilder --git-dist=jessie
```

Review contents of a package

```
dpkg -c /home/racke/build/debaux/debaux_0.1.11-1_all.deb
```

Debugging

Debconf

You can set the environment variable *DEBCONF_DEBUG* in order to debug the interaction with Debconf during package installation.

```
export DEBCONF_DEBUG=developer
```

Resources

Debian Building Tutorial
DFSG Licenses

Linuxia Wiki

Stefan Hornburg (Racke)
Maintaining Debian Packages

wiki.linuxia.de