

# Build Debian/Ubuntu packages to make it easy for users to install your software

Samuel Thibault

2019 May 13th



- Why making packages?
- How about Debian
- Introduction to distribution & packages
- Introduction to making a package
- Introduction to distributing a package



### Why making packages?

### ./configure && make && make install

- Install where?
- Manage upgrades
- Missing dependencies
- Make sure dependencies kept installed



### How about Debian?

#### One of the main GNU/Linux distributions

- Started in 1993
- Community-based
- Ported to a dozen architectures
- Basically includes most available free software
- Many many many derivatives
  - More than 300, 120 of which still active
  - Ubuntu, notably
- ~1000 developers, ~2000 non-developer maintainers



### How about Debian?

#### **Social Contract**

- Between developers and users
- Basically
  - "Debian will always be free"
  - "We won't hide problems"
    - Everything is public, except debian-private
  - "Users are our priority"
  - "There is non-free software out there"
    - "non-free" and "contrib" sections



### What is a distribution?

#### From source to installed system

- Gather coherent set of source code
  - Linux kernel (or others)
  - Libc
  - System tools
  - Libraries
  - Applications
  - Desktop environments
- Compile everything
  - "binary Packages"



### What is a distribution?

- Archive
  - http mirrors
- Installer
  - Unpacks basic system
  - Then make it install binary packages
- Package manager
  - To install more binary packages

And that's about it!



### What is a package?

#### Source vs. binary package

- Source package:
  - "upstream" source code (as a tarball)
  - debian/ directory (as another tarball, or patch)
    - Debian meta-data
    - some patches against upstream source code

#### ...build...

- Binary package(s):
  - Binaries + meta-data

### Put into practice

```
€ apt-get source hello
€ ls hello*
hello 2.10-1.dsc
hello_2.10.orig.tar.gz
hello 2.10-1.debian.tar.xz
hello-2.10/
€ cd hello-2.10
€ ls
INSTALL README TODO Makefile.am Makefile.in
debian/
lib/ man/ po/ src/ tests/
```

### Put into practice

```
€ sudo apt-get build-dep hello
€ dpkg-buildpackage
€ cd ..
€ ls
hello_2.10-1_amd64.changes
hello_2.10-1_amd64.deb
€ sudo apt-get install ./hello_*.deb
```



### What is a source package?

The source is all the source, only the source

- No pre-built stuff
  - or regenerate it during the build
- No embedded copies of other software

Best guarantee for user to have free software

- Able to rebuild it all
- Able to modify all of it
- Make sure to modify a library



### What is a source package?

#### Source meta-data

- Potentially very large information
- Basically
  - debian/copyright: document licences
  - debian/control: package name, dependencies
  - debian/rules: how to build the package
  - debian/changelog: as name suggests



#### Minimal debian/control

Source: hello

Build-Depends: debhelper (>= 10)

Package: hello

Architecture: any

Description: hello

This is just a test package



### Put into practice

#### Minimal debian/rules

```
#!/usr/bin/make -f
응:
        dh $@
```

### Thanks to debian/compat containing

10

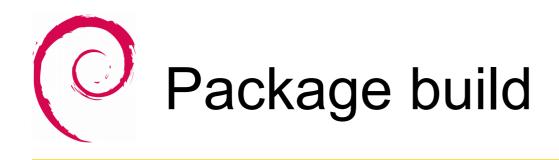


#### Minimal debian/changelog

```
hello (2.10-1) sid; urgency=low
```

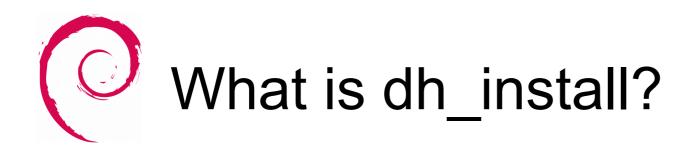
\* Initial packaging.

```
-- Samuel Thibault <sthibault@debian.org> Tue, 11 May 2019 14:00:00 +0200
```



#### Several steps done by dpkg-buildpackage

- debian/rules clean
  - dh\_auto\_clean
    - make clean
- debian/rules build
  - dh\_auto\_build
    - configure && make && make check
- debian/rules binary
  - dh\_auto\_install
    - make install
  - dh\_install
  - dh\_builddeb

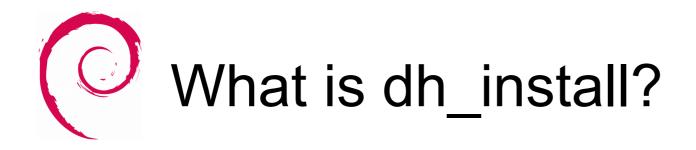


#### Simple case: one binary package

- make install into debian/hello
- dh\_install basically no-op, dh\_builddeb packs

### Several binary packages

- make install into debian/tmp
- dh\_install moves files to debian/hello-foo and debian/hello-bar
- dh\_builddeb packs hello-foo and hello-bar



#### Library example

```
debian/libhello1.install:
  /usr/lib/*/lib*.so.*
debian/libhello-dev.install:
  /usr/include
  /usr/lib/*/lib*.a
  /usr/lib/*/lib*.so
  /usr/lib/*/pkgconfig/*
```



### Why two packages for a lib?

### Basically, room

- One usually don't need -dev for all installed libs
- And even less -doc
- Saves
  - Disk
  - Network bandwidth on upgrade
  - Directory indexing



libfoo1 vs libfoo-dev

Say my hello uses libfoo

- Needs libfoo-dev at build time
  - apt-get build-dep hello
- Needs libfoo1 at run time
  - User shouldn't have to care about it



### Put into practice

#### More involved debian/control

```
Source: hello

Build-Depends: debhelper (>= 10), libfoo-dev, libbar-dev

Package: hello

Architecture: any

Depends: ${shlibs:Depends}}

Description: hello

This is just a test package
```

dh\_shlibdeps step will compute Depends



### Architecture: any packages?

#### Arch-dependent vs Arch-independent

- Does it depend on arch?
  - Processor instruction set
  - 32bit vs 64bit
  - Little vs big endian
  - → Arch-dependent: libfoo\_1.0-1\_amd64.deb
- → Otherwise, arch-independent
  - Architecture: all packages
  - e.g. libfoo-data\_1.0-1\_all.deb



### Architecture: all packages?

- Data
- Documentation
- Non-compiled langages
  - Python
  - Perl
  - TeX

No need to rebuild them for each arch One copy on mirrors



### My program

- One source package: foo
- One binary package: foo

### My library

- One source package: libfoo
- E.g. three binary packages:
  - libfoo1 (arch:any)
  - libfoo-dev (arch:any)
  - libfoo-doc (arch:all)

### Making it simple

```
€ apt-get install dh-make

€ dh_make

Type of package: (single, indep, library, python) [silp]?

€ $EDITOR debian/control debian/copyright

€ rm debian/*.ex

€ dpkg-buildpackage
```

Will just work fine if your upstream is nice (automake, cmake, ...)



### A bit less simple

I lied, doesn't work with hello source

- Because odd GNUmakefile lingering there
- dh\_auto\_clean thinks it can run make clean, fails



### More information

- Debian Packaging Tutorial
- http://www.debian.org/devel/
- Debian New Maintainer's Guide
- Debian Developer's Reference
- Debian Policy



### Yay, my package is ready!

#### Check its soundness

€ lintian hello\_1.0-1\_amd64.changes

### Now let's publish it

- My own website
- Debian/Ubuntu

### What to publish?

- Source packages
- Binary packages
  - For various distribs



### Why various binary packages

- Library versions
  - Debian 8 contains libicu52
  - Debian 9 contains libicu57, not libicu52
  - No compatibility
  - Packages using libicu need a rebuild
- Must not be blindly overriden
  - There be dragons!



### Building for various distribs

```
€ apt-get install pbuilder

€ sudo pbuilder create --basetgz $HOME/base-jessie.tgz
    --distribution jessie

€ cd ~/hello-2.10

€ pdebuild -- --basetgz $HOME/base-jessie.tgz

€ cd /var/cache/pbuilder/result/
```

### Also useful for checking missing Build-Depends

### For each distrib

#### In sources.list:

```
deb [trusted=yes] http://my.website/jessie ./
```



### Pushing to Debian

#### And thus all its derivatives

- Needs review & approval of course
- And actually uploaded at some point
  - Signed with gpg key

### Debian Developers (DD)

Can upload anything

### Debian Maintainers (DM)

Can upload the packages they are allowed to



### Pushing to Debian

#### Becoming a DD

- Prove technical skills and knowledge of Debian
- Long work

#### Becoming a DM

- Get some DD sign your gpg key and advocate you
- Get some DD to allow you to upload
- After mentoring for some time, usually

#### Get sponsored

Get some DD or DM to do the upload



### Pushing to Debian

#### Any DD out there?

- Lyon: Ana Guerrero, Aurélien Jarno, Josselin Mouette
- Also Hugo Lefeuvre, Raphael Geissert, Romain Françoise, Stephen Kitt
- Around Lyon: Raphaël Hertzog

Way better if **you** maintain your package :)

Also debian-mentors for help



### Contributing to Debian

### Now, in reality

- Start with submitting bug reports
- Submit patches fixing bugs
  - Probably not so good, mentoring
- Submit good patches fixing bugs
- Get commit access
- Get DM status
- Get DD status

## Cheatlist

€ debcheckout hello € dgit clone hello € dget http://www.foo.org/foo.dsc € dpkg-source -x foo.dsc € dpkg-checkbuilddeps € mk-build-deps -s sudo -r -i € debdiff hello\_1.0-{1,2}\_amd64.changes € rmadison libicu52