Debian Packaging School: Lesson 1

Internet, the final frontier. These are the adventures of the Debian project. Its n-year mission: to explore free new programs, to seek out new source and new users, to boldly package what no-one has packaged before.

Lars Wirzenius <liw@iki.fi>

The Point of Debian

- Best possible free operating system
- Make it easy to install software
  - Pre-compiled, sensibly configured, license checked
  - Dependencies, upgrades, security support
- Make it easy to uninstall software
  - Bookkeeping: which packages own each file
- Make sure all software works and works together

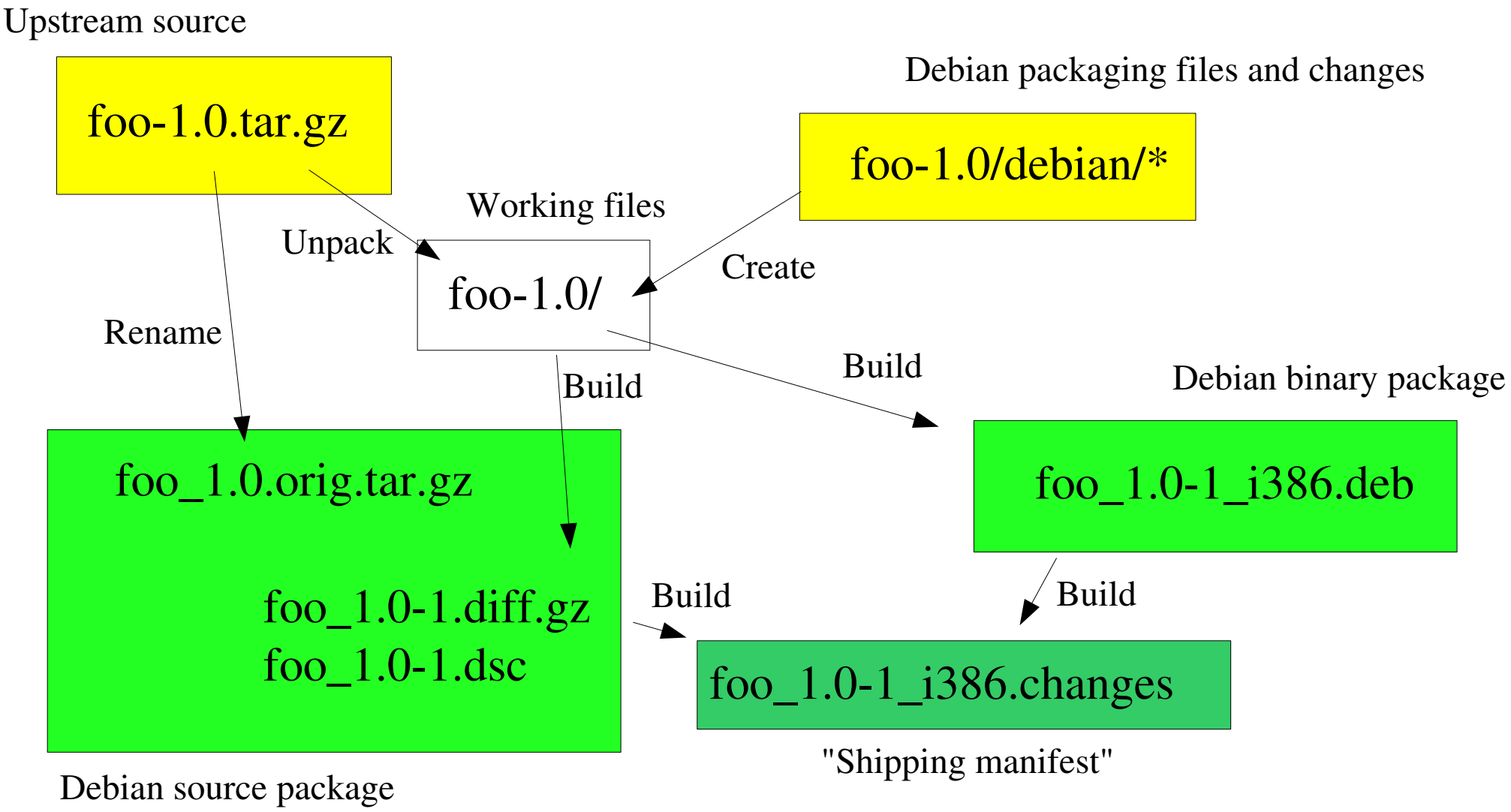
Isolated packages are not the point!

- Quality, quality, quality
- Integration, integration, integration
- Standards, standards, standards
  - Debian Policy Manual
  - Filesystem Hierarchy Standard
  - Linux Standard Base

Overview of packaging process

- Announce your Intent To Package
  - Check "WNPP" for existing ITPs
- Fetch and unpack upstream source code
  - Check license
- Add Debian specific changes to source code
- Build Debian source package
- Build Debian binary package(s)
- Test and fix and repeat until done

Overview of packaging process



Upstream source

- Usually a .tar.gz from a web page
  - Occasionally in a different format, or only in a version control system, in which case you need to create the .tar.gz yourself
- Name must be `foo_1.2.3.orig.tar.gz`
  - Desired: only name changed from original
  - Pristine source is important in many ways (security!)

Debian specific changes

- Debian packaging files in `debian/`
  - `changelog`
  - `control`
  - `copyright`
  - `rules`
  - Sometimes more files as well
- Sometimes also changes to upstream source
  - Bug fixes, configuration, etc

debian/changelog

- Changes to the Debian packaging
  - Upstream change log file has upstream changes, but they can be summarized very briefly in the Debian change log as well

```
liwc (1.20-1) unstable; urgency=low

  * New upstream version adds liwc(1) manual page.
  * Converted from using debhelper to native dpkg stuff.

 -- Lars Wirzenius <liw@iki.fi>  Thu, 06 Feb 2003 12:59:34 +0200
```

debian/control

- Source and binary package meta data

```
Source: liwc
Maintainer: Lars Wirzenius <liw@iki.fi>
Section: devel
Priority: optional
Standards-Version: 3.5.8.0
Build-Depends: publib-dev

Package: liwc
Architecture: any
Depends: ${shlibs:Depends}
Description: Tools for manipulating C source code
 Includes programs for converting C++ comments to C comments,
 removing C comments, print out string literals, and converting
 characters to trigraphs and trigraphs to characters.
```

`debian/copyright`

- Copyright and license information
  - Essential to get this correct! Really, really important!
  - Include all license text (except only pointers to GPL, LGPL, and some others in `/usr/share/common-licenses`)
- Free-form text, read and used by people only
- Also: where to find upstream source, who did the packaging, etc.

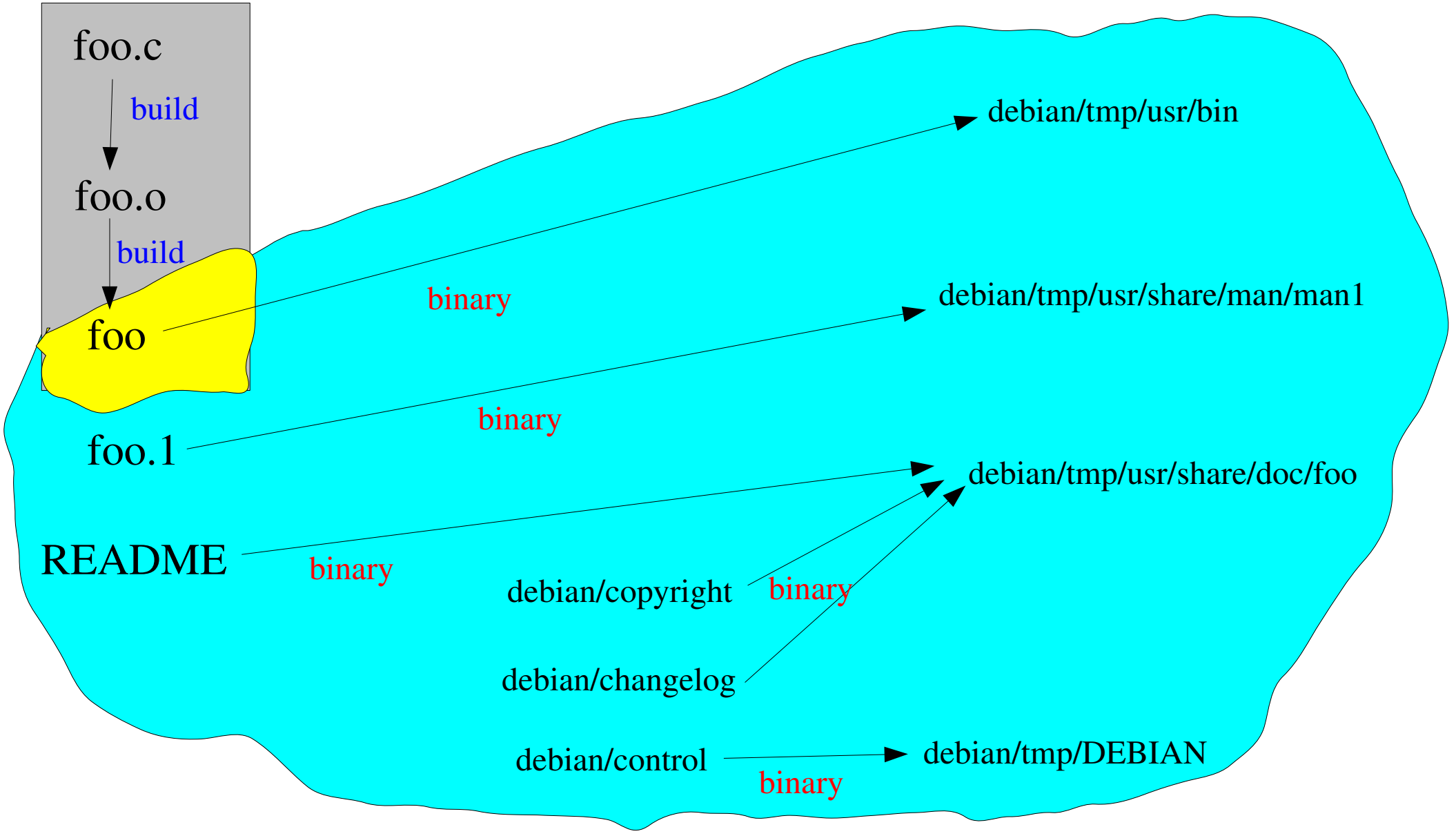
`debian/rules`

- Commands for compiling and building the package
  - Technically an executable Makefile with specified targets: build, binary, binary-arch, binary-indep, clean
  - "Raw" commands or "helper": matter of taste
- Sometimes very simple, sometimes very much not simple
  - Integration into system may require hard work

`debian/rules`: Overview

- `debian/rules build`: compile the program
- `debian/rules binary`: create binary package
  - Install program under `debian/tmp`
  - Put packaging meta data into `debian/tmp/DEBIAN`
- `debian/rules clean`: clean up afterwards

`build` or `binary`?



debian/rules: The simple stuff

```
#!/usr/bin/make -f

build:
	$(MAKE) CFLAGS=-O2 LDFLAGS=-s

clean:
	$(MAKE) clean
	rm -rf debian/files debian/substvars debian/tmp

binary: binary-arch binary-indep

binary-indep:
```

debian/rules: Build binary package

```
binary-arch: build
	rm -rf debian/tmp
	install -d debian/tmp debian/tmp/DEBIAN
	install -d debian/tmp/usr/bin
	install -d debian/tmp/usr/share/doc/liwc
	install -d debian/tmp/usr/share/man/man1
---
	strip --strip-unneeded -R .note -R .comment ccmtcnvt
	strip --strip-unneeded -R .note -R .comment cstr
	strip --strip-unneeded -R .note -R .comment entrigraph
	strip --strip-unneeded -R .note -R .comment rmccmt
	strip --strip-unneeded -R .note -R .comment untrigraph
	$(MAKE) bindir=debian/tmp/usr/bin \
	        man1dir=debian/tmp/usr/share/man/man1 \
	        install
	cp debian/changelog debian/tmp/usr/share/doc/liwc/changelog.Debian
	cp ChangeLog debian/tmp/usr/share/doc/liwc/changelog
	cp README debian/tmp/usr/share/doc/liwc
	gzip -9 debian/tmp/usr/share/doc/liwc/*
---
	cp debian/copyright debian/tmp/usr/share/doc/liwc
	gzip -9 debian/tmp/usr/share/man/man1/*
	dpkg-shlibdeps debian/tmp/usr/bin/*
	dpkg-gencontrol -isp
	chown -R root.root debian/tmp
	chmod -R o-s,go=u,go-ws debian/tmp
	dpkg --build debian/tmp ..
```

`debian/rules`: Create directories

```
binary-arch: build
	rm -rf debian/tmp
	install -d debian/tmp debian/tmp/DEBIAN
	install -d debian/tmp/usr/bin
	install -d debian/tmp/usr/share/doc/liwc
	install -d debian/tmp/usr/share/man/man1

...
```

debian/rules: Install files

```
binary-arch: build

...
	strip --strip-unneeded -R .note -R .comment ccmtcnvt
	strip --strip-unneeded -R .note -R .comment cstr
	strip --strip-unneeded -R .note -R .comment entrigraph
	strip --strip-unneeded -R .note -R .comment rmccmt
	strip --strip-unneeded -R .note -R .comment untrigraph
	$(MAKE) bindir=debian/tmp/usr/bin \
	        man1dir=debian/tmp/usr/share/man/man1 \
	        install
	cp debian/changelog debian/tmp/usr/share/doc/liwc/changelog.Debian
	cp ChangeLog debian/tmp/usr/share/doc/liwc/changelog
	cp README debian/tmp/usr/share/doc/liwc
	gzip -9 debian/tmp/usr/share/doc/liwc/*
	cp debian/copyright debian/tmp/usr/share/doc/liwc
	gzip -9 debian/tmp/usr/share/man/man1/*

...
```

`debian/rules`: Meta data, package itself

```
binary-arch: build

...
        dpkg-shlibdeps debian/tmp/usr/bin/*
        dpkg-gencontrol -isp
        chown -R root.root debian/tmp
        chmod -R o-s,go=u,go-ws debian/tmp
        dpkg --build debian/tmp ..
```

Doing the actual build

- Run `debian/rules` by hand until you've got the packaging files correct
- `dpkg-buildpackage -us -uc -rfakeroot`
  - Does a full "real" package build (without signing)
  - Also builds source package
- `sudo pbuilder build foo_1.0-1.dsc`
  - Does full real build in "chroot" to check for build dependencies; still unsigned
  - Can wait until later (advanced stuff)

Build results

- `foo_1.0.orig.tar.gz`
  - Well, manually renamed from original, not really built.
- `foo_1.0-1.diff.gz`
- `foo_1.0-1.dsc`
- `foo_1.0-1_i386.changes`
- `foo_1.0-1_i386.deb`

Test the package

- `lintian *.changes`
  `linda *.changes`
  - Automatic checks for some common problems
- Installation tests:
  - Install on fresh system, remove
  - Upgrade from earlier version
  - `piuparts` does this (later, advanced stuff)
- Also, of course, use the package yourself

Get hacking!

- `http://liw.iki.fi/liw/debian/liw-hello-1.0.tar.gz`
  - The upstream source tarball
- Create a package out of that
- Rembemer to test with lintian and linda
- Ask for help if you get stuck
- Once you're done, do a victory dance

What next?

- Put on web page
  - Possibly your very own apt repository (so that `apt-get install` will work), but this needs setting up
- GPG-sign .changes and .dsc and upload to Debian
  - Official developers only, others via sponsorship
  - NEW packages checked manually before they enter Debian

`debhelper`: A good thing

- Writing all `debian/rules` command lines manually can be tedious and repetitive
  - Also: sometimes policy changes and `rules` files have to be updated
- debhelper is a suite of programs to do common tasks
  - `dh_installman` installs manual pages to the right directories with the right names
- **Know first what goes on under the hood!**

pbuilder: a clean build environment

- The development machine easily has a lot of packages installed (and not all of them official)
- It is easy to forget build dependencies
- `pbuilder create --mirror http://foo/debian`
- `pbuilder build foo.dsc`
- `pdebuild` (inside unpacked source directory)
- `pbuilder update`
- Requires root (use sudo). Results in `/var/cache/pbuilder/result/` by default

piuparts: installation etc testing

- Package Installation, UPgrading, And Removal Testing Suite
  - Uses a clean minimal system (chroot)
  - Installs a package, then removes it
  - Installs previous version, upgrades to new version, then removes it
  - Installs version from sarge, upgrades to etch, upgrades to sid, then removes
- Reports missing files, extra files, modifed files

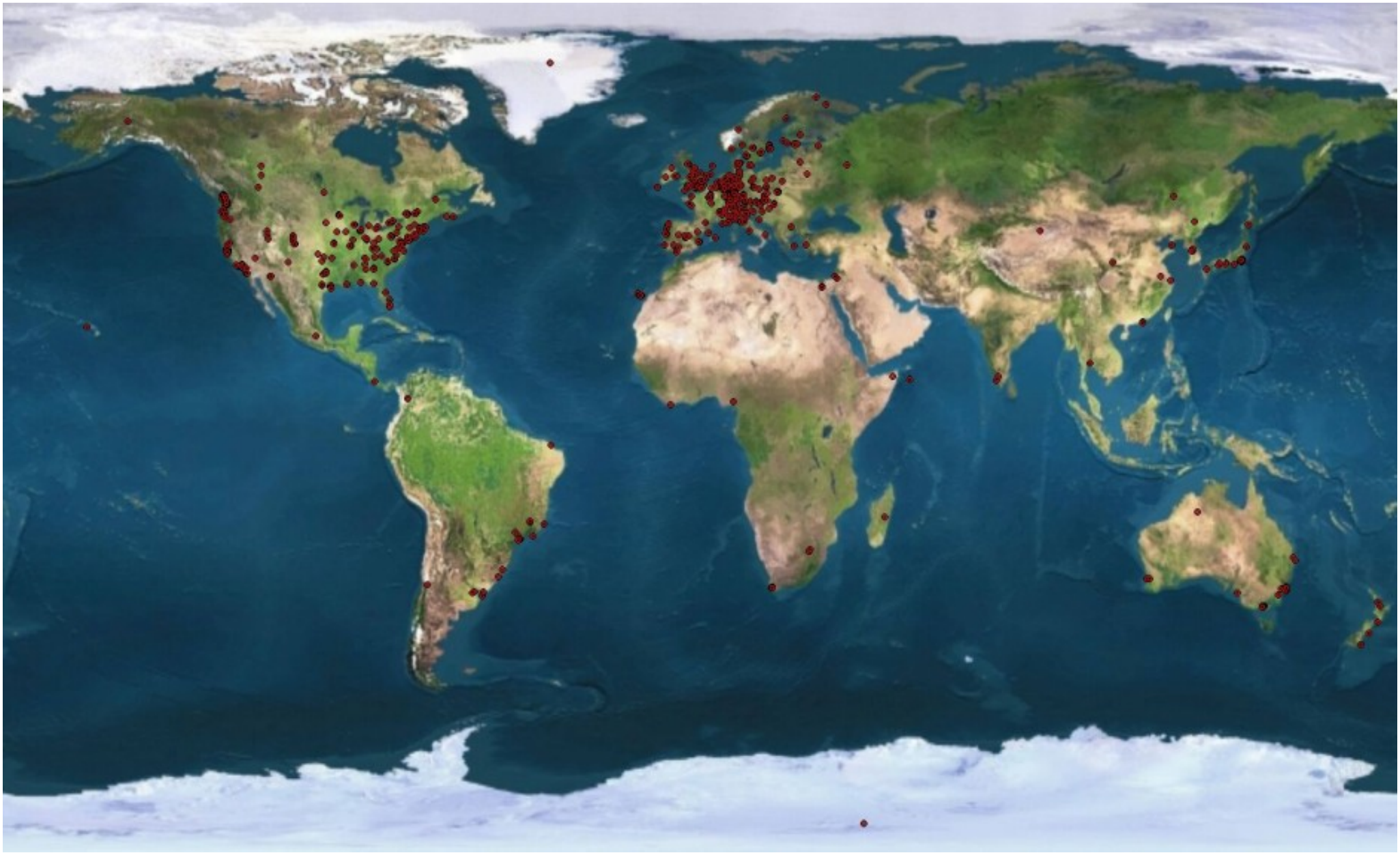
Where next?

- `http://www.debian.org` for more to learn
  - Developers' Corner
  - New Maintainer pages
  - Developer's Reference
  - `http://bugs.debian.org/wnpp`
- Follow project discussions
  - `http://lists.debian.org`
  - irc.debian.org: #debian-devel

The New Maintainer Process

- Don't worry about it until you're comfortable with packaging
  - Except maybe create GPG key and get it signed
- Can take months, but mostly waiting
- Scary reputation, mostly undeserved
  - Fear is the mind killer...
- Lots of work, plenty to learn
  - But most of it is necessary to make good packages

Debian development is global, but concentrated to Europe and the US



People from anywhere are welcome!