The Common Debian Build System
(CDBS)

Peter Eisentraut

FOSDEM 2009

What is CDBS?

- A set of makefile fragments to include into `debian/rules`
- Makes packaging complex packages easier.
- Makes packaging simple packages harder.
- Initiates you to a secret subculture of Debian packagers.
- Causes you to lose all your not-CDBS-using friends.

Simple Example

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

include /usr/share/cdbs/1/rules/debhelper.mk
include /usr/share/cdbs/1/class/autotools.mk

DEB_CONFIGURE_EXTRA_FLAGS = --enable-funnybiz
DEB_INSTALL_DOCS_ALL = readme.txt
LDFLAGS += -Wl,--as-needed

binary-install/mypackage-tools::
        chmod a+x debian/mypackage-tools/\
            usr/lib/mypackage/*
```

Simple Example

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

include /usr/share/cdbs/1/rules/debhelper.mk
include /usr/share/cdbs/1/class/autotools.mk

DEB_CONFIGURE_EXTRA_FLAGS = --enable-funnybiz
DEB_INSTALL_DOCS_ALL = readme.txt
LDFLAGS += -Wl,--as-needed

binary-install/mypackage-tools::
        chmod a+x debian/mypackage-tools/\
            usr/lib/mypackage/*
```

Simple Example

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

include /usr/share/cdbs/1/rules/debhelper.mk
include /usr/share/cdbs/1/class/autotools.mk

DEB_CONFIGURE_EXTRA_FLAGS = --enable-funnybiz
DEB_INSTALL_DOCS_ALL = readme.txt
LDFLAGS += -Wl,--as-needed

binary-install/mypackage-tools::
        chmod a+x debian/mypackage-tools/\
            usr/lib/mypackage/*
```

Background

- First published in 2003 by Colin Walters
- Co-maintained 2003-2005 by Colin Walters and Jeff Bailey
- Hijacked in 2006 by me after a period of no maintenance
- Currently 2 or 3 maintainers plus Ubuntu feedback

Package Parameters

```
Source: cdbs
Section: devel
Priority: optional
Maintainer:
 CDBS Hackers <build-common-hackers@lists.alioth.debian.org>
Build-Depends-Indep: debhelper (>= 5), graphviz, realpath, fakeroot,
 python-dev, python2.4-dev, python2.5-dev, libxml2-utils, xsltproc,
 docbook-xml, docbook-xsl, dblatex, ant, kaffe, sharutils, gs-common,
 texlive-extra-utils
Uploaders: Jonas Smedegaard <dr@jones.dk>,
 Marc Dequènes (Duck) <Duck@DuckCorp.org>,
 Peter Eisentraut <petere@debian.org>
Standards-Version: 3.7.3
Vcs-Browser: http://svn.debian.org/wsvn/build-common/
Vcs-Svn: svn://svn.debian.org/build-common/
```

Bugs: 1 I, 31 N, 7 M, 33 W

Last upload: 01 Apr 2008

Statistics

- Currently 13077 source packages in unstable
- 3185 use CDBS (24.4%)
- Appears to be increasing by 0.2% or so per month
- Archive takeover completed by 2020 :-)

Architecture

- Yes, there is one:
  - *Classes* contain the logic for the package's upstream build system: make, autotools, MakeMaker, distutils, cmake, etc.
  - *Rules* contain additional rules of various utility: debhelper, tarball-in-tarball, patch tools
  - Classes use inheritance (a.k.a. makefile inclusion): autotools inherits make, gnome inherits autotools, etc.
- In practice, this is arcane, so just remember:
  - Exactly one class per `debian/rules`
  - Zero or more rules per `debian/rules`
  - Usually, you want at least the debhelper rules set
  - Deviations are possible, but rarely useful

Advantages

Technical advantages:

- Writing `debian/rules` becomes quick and easy (usually)
- You get the details right for free:
  - `dh_*` sequencing
  - `configure --build/--host` (see autotools-dev README)
  - `DEB_BUILD_OPTIONS`: noopt, nostrip
  - `CFLAGS`, `CXXFLAGS`, etc. (try passing them to cmake or qmake correctly)
  - parallel make
  - Java packaging policy
- You follow policy evolvement for free:
  - Python policy
  - `patch`, `unpatch` targets
  - KDE 4 packaging practices

Advantages

Conceptual advantages:

- `debian/rules` highlights the nonstandard packaging steps, rather than hiding them in boilerplate.
- Avoids propagation of mistakes through copy-and-paste or `dh_make`.
- Avoids home-brewed and half-baked alternative `debian/rules` makefile fragments packages.

Disadvantages

- If your package has an unusual build system, CDBS might be in your way.
- If you are a control-freak, CDBS will be in your way.
- If GNU make syntax confuses you, CDBS will confuse you.
- CDBS is not good for learning Debian packaging.

Makefile Class

```
include /usr/share/cdbs/1/class/makefile.mk

DEB_MAKE_BUILD_TARGET = all
DEB_MAKE_INSTALL_TARGET = install DESTDIR=$(CURDIR)
DEB_MAKE_CLEAN_TARGET = clean
```

- Sets `CFLAGS` etc. according to policy
- Observes `DEB_BUILD_OPTIONS`

Autotools Class

```
include /usr/share/cdbs/1/class/autotools.mk

DEB_CONFIGURE_EXTRA_FLAGS = --enable-funnybiz
```

- "Inherits" makefile class
- Calls configure with proper options (host tuple, installation paths)
- Calls `make distclean` instead of `make clean`
- Handles `config.guess`, `config.sub` updates
- Supports `autoreconf`, `libtoolize` before build (dubious)

Perl Class

```
include /usr/share/cdbs/1/class/perlmodule.mk

DEB_MAKEMAKER_USER_FLAGS = --with-foo
```

- “Inherits” makefile class
- Calls MakeMaker with proper options and paths

Python Class

```
include /usr/share/cdbs/1/class/python-distutils.mk

# mandatory
DEB_PYTHON_SYSTEM = pycentral | pysupport
```

- “Inherits” makefile class
- Figures out all the Python policy details :-)
- Documentation needs updates :-(

KDE Class

```
include /usr/share/cdbs/1/class/kde.mk
```

(for KDE <= 3.5)

- “Inherits” autotools class
- Sets proper `configure` flags, additional installation paths
- Documentation installation
- Used for all major KDE packages

KDE 4 class will be based on CMake.

Ant Class

```
include /usr/share/cdbs/1/class/ant.mk

JAVA_HOME = /usr/lib/kaffe
DEB_JARS = junit
DEB_ANT_BUILD_TARGET = jars
DEB_ANT_COMPILER = jikes

install/libjline-java:: DEB_FINALDIR=$(CURDIR)/debian/libjline-java
install/libjline-java::
        install -m 644 -D release/jline-0_9_5.jar \
            $(DEB_FINALDIR)/usr/share/java/jline-$(DEB_UPSTREAM_VERSION).jar
        dh_link /usr/share/java/jline-$(DEB_UPSTREAM_VERSION).jar \
            /usr/share/java/jline.jar
```

- Most Java library packags are built this way.
- Tuned to Java packaging policy (at some point in time).

Other classes

- CMake
- GNOME
- Haskell
- QMake
- OCaml (external)
- PEAR (external)

Debhelper Rules

```
include /usr/share/cdbs/1/rules/debhelper.mk
```

- Usually put this first in every `debian/rules`
- Takes care of proper sequencing and sequence points
- Arguments for every command can be overridden on a per-package basis, e.g.,

```
DEB_INSTALL_EXAMPLES_libjline-java-doc = \
    release/jline-demo.jar
```

- Easy and automatic debug package support

Patch Systems Support

```
include /usr/share/cdbs/1/rules/...
```

dpatch.mk DPatch rules

patchsys-quilt.mk Quilt rules (contained in `quilt` package)

simple-patchsys.mk Home-grown “simple” patch system (TANSTAA“S”PS)

What CDBS Is Not

- No support for un-Common build systems
- Not a dumping ground for rejected debhelper features.
- Not a workaround for missing dpkg functionality.
- Not a way to enforce packaging behavior (but to enable or encourage it).
- Individual teams sometimes create a local subclass to enforce local policy.

Comparisons and Contrasts

vs. Debhelper only Debhelper is only the steps, CDBS puts them together.

vs. dh (Debhelper 7) Does not allow for customization, only suitable for dead-simple packages.

vs. BSD ports Very similar setup. Difference: Variables at the top, include `bsd.port.mk` at the bottom.

vs. Automake Same declarative approach; details about actual make rules are hidden.

Problems

- Patching the clean rule of a makefile (or equivalent) is broken
- Feature creep/unbounded user requests
- Certain old features/behaviors are not documented or understood
- CDBS maintainer must know all subpolicies in Debian ;-)

Plans: CDBS 2

- Original idea: replace `/usr/share/cdbs/1/rules/foo.mk` by `/usr/share/cdbs/2/rules/foo.mk`
- Upcoming idea: similar to Debhelper compat levels, e.g., `CDBS_COMPAT = 2`
- Throw out some obsolete features
- Synchronize default values with Debhelper 7
- Maybe move the include to the bottom
- Need a better/useful version numbering system

How to Help

- Bugs, patches, ideas welcome.
- Join bug discussions.
- Enhance the test suite.
- Drive the right features into `debhelper` and `dpkg`.

Summary

- Include CDBS rules rather than writing `debian/rules` by hand.
- Takes care of details and policy requirements automatically.
- Particularly useful for packages with uniform build processes.
- Not suitable for all packages and packagers.