Debian Pakete bauen

Eine kurze Einführung

Sebastian Harl
<Sebastian.Harl@sternwarte.uni-erlangen.de>

Astronomisches Institut der Universität Erlangen-Nürnberg

12. März 2009

Überblick

Welche Arten von Paketen gibt es?

- Quell-Pakete:
  Enthalten den Quellcode des Programms und Debian-spezifische Änderungen:
  - `beispiel_1.0.orig.tar.gz` – Quellcode

Welche Arten von Paketen gibt es?

- Quell-Pakete:
  Enthalten den Quellcode des Programms und Debian-spezifische Änderungen:
  - `beispiel_1.0.orig.tar.gz` – Quellcode
  - `beispiel_1.0-1.diff.gz` – Patch mit Debian-spezifischen Änderungen

Welche Arten von Paketen gibt es?

- Quell-Pakete:
  Enthalten den Quellcode des Programms und Debian-spezifische Änderungen:
  - `beispiel_1.0.orig.tar.gz` – Quellcode
  - `beispiel_1.0-1.diff.gz` – Patch mit Debian-spezifischen Änderungen
  - `beispiel_1.0-1.dsc` – Beschreibung des Quell-Pakets

Welche Arten von Paketen gibt es?

- Quell-Pakete:
  Enthalten den Quellcode des Programms und Debian-spezifische Änderungen:
  - `beispiel_1.0.orig.tar.gz` – Quellcode
  - `beispiel_1.0-1.diff.gz` – Patch mit Debian-spezifischen Änderungen
  - `beispiel_1.0-1.dsc` – Beschreibung des Quell-Pakets
- Binär-Pakete:
  Fertig übersetzt, zum installieren:
  - `beispiel_1.0-1_powerpc.deb` – Architektur-abhängig

Welche Arten von Paketen gibt es?

- Quell-Pakete:
  Enthalten den Quellcode des Programms und Debian-spezifische Änderungen:
  - `beispiel_1.0.orig.tar.gz` – Quellcode
  - `beispiel_1.0-1.diff.gz` – Patch mit Debian-spezifischen Änderungen
  - `beispiel_1.0-1.dsc` – Beschreibung des Quell-Pakets
- Binär-Pakete:
  Fertig übersetzt, zum installieren:
  - `beispiel_1.0-1_powerpc.deb` – Architektur-abhängig
  - `beispiel-data_1.0-1_all.deb` – Architektur-unabhängig

Benötigte Pakete

- `build-essential` – Meta-Paket; Abhängigkeit auf essentielle Pakete mit Compilern und Bibliotheken

Benötigte Pakete

- `build-essential` – Meta-Paket; Abhängigkeit auf essentielle Pakete mit Compilern und Bibliotheken
- `debhelper` – Nützliche Skripte für kleine Aufgaben

Benötigte Pakete

- `build-essential` – Meta-Paket; Abhängigkeit auf essentielle Pakete mit Compilern und Bibliotheken
- `debhelper` – Nützliche Skripte für kleine Aufgaben
- `dh-make` – Helfer zum Erstellen der ersten „Schablonen“ eines Paketes

Benötigte Pakete

- `build-essential` – Meta-Paket; Abhängigkeit auf essentielle Pakete mit Compilern und Bibliotheken
- `debhelper` – Nützliche Skripte für kleine Aufgaben
- `dh-make` – Helfer zum Erstellen der ersten „Schablonen“ eines Paketes

Optionale Pakete

- `lintian` – Zum Prüfen auf häufige Fehler

Benötigte Pakete

- `build-essential` – Meta-Paket; Abhängigkeit auf essentielle Pakete mit Compilern und Bibliotheken
- `debhelper` – Nützliche Skripte für kleine Aufgaben
- `dh-make` – Helfer zum Erstellen der ersten „Schablonen“ eines Paketes

Optionale Pakete

- `lintian` – Zum Prüfen auf häufige Fehler
- `pbuilder` – Zum Bauen in „sauberen“ Umgebungen

Benötigte Pakete

- `build-essential` – Meta-Paket; Abhängigkeit auf essentielle Pakete mit Compilern und Bibliotheken
- `debhelper` – Nützliche Skripte für kleine Aufgaben
- `dh-make` – Helfer zum Erstellen der ersten „Schablonen“ eines Paketes

Optionale Pakete

- `lintian` – Zum Prüfen auf häufige Fehler
- `pbuilder` – Zum Bauen in „sauberen“ Umgebungen
- `quilt` oder `dpatch` – Zum Verwalten von zusätzlichen Patches

Erstellen eines Debian-Paketes

Im Idealfall ist dies ganz einfach:

1. Herunterladen des Quellcodes

Erstellen eines Debian-Paketes

Im Idealfall ist dies ganz einfach:

1. Herunterladen des Quellcodes
2. Entpacken des Quellcodes

Erstellen eines Debian-Paketes

Im Idealfall ist dies ganz einfach:

1. Herunterladen des Quellcodes
2. Entpacken des Quellcodes
3. Lesen der Dokumentation (evt. Installieren von benötigten Bibliotheken)

Erstellen eines Debian-Paketes

Im Idealfall ist dies ganz einfach:

1. Herunterladen des Quellcodes
2. Entpacken des Quellcodes
3. Lesen der Dokumentation (evt. Installieren von benötigten Bibliotheken)
4. Aufrufen von `dh_make`

Erstellen eines Debian-Paketes

Im Idealfall ist dies ganz einfach:

1. Herunterladen des Quellcodes
2. Entpacken des Quellcodes
3. Lesen der Dokumentation (evt. Installieren von benötigten Bibliotheken)
4. Aufrufen von `dh_make`
5. Anpassen der Dateien unterhalb von `debian/`

Erstellen eines Debian-Paketes

Im Idealfall ist dies ganz einfach:

1. Herunterladen des Quellcodes
2. Entpacken des Quellcodes
3. Lesen der Dokumentation (evt. Installieren von benötigten Bibliotheken)
4. Aufrufen von `dh_make`
5. Anpassen der Dateien unterhalb von `debian/`
6. Paket mit `debuild` bauen

Erstellen eines Debian-Paketes

Im Idealfall ist dies ganz einfach:

1. Herunterladen des Quellcodes
2. Entpacken des Quellcodes
3. Lesen der Dokumentation (evt. Installieren von benötigten Bibliotheken)
4. Aufrufen von `dh_make`
5. Anpassen der Dateien unterhalb von `debian/`
6. Paket mit `debuild` bauen
7. Fehler suchen, finden und beheben

Erstellen eines Debian-Paketes

Im Idealfall ist dies ganz einfach:

1. Herunterladen des Quellcodes
2. Entpacken des Quellcodes
3. Lesen der Dokumentation (evt. Installieren von benötigten Bibliotheken)
4. Aufrufen von `dh_make`
5. Anpassen der Dateien unterhalb von `debian/`
6. Paket mit `debuild` bauen
7. Fehler suchen, finden und beheben
8. Benutzen :)

Innereien

Wichtige Dateien in `debian/`

- `debian/control` Enthält wichtige Meta-Daten über das Source-Paket:
  - Wer ist dafür Verantwortlich?
  - Welche Binär-Pakete werden gebaut?
  - Beschreibungen der Binär-Pakete?
  - ...

Wichtige Dateien in `debian/` (cont'd)

- `debian/copyright` – Wie ist die Software lizensiert?
- `debian/rules` – Wie wird das Paket eigentlich gebaut?
- `debian/changelog` – Was hat sich zwischen den einzelnen Paket-Versionen geändert?

debian/control

```
Source: tig
Section: utils
Priority: optional
Maintainer: Sebastian Harl <sh@tokkee.org>
Build-Depends: debhelper (>= 5), dpatch, dpkg-dev (>= 1.14.6), git-core,
 libncursesw5-dev, asciidoc (>= 7), xmlto, docbook-utils
Standards-Version: 3.8.0
Homepage: http://jonas.nitro.dk/tig/

Package: tig
Architecture: any
Depends: git-core (>= 1.5.4), ${shlibs:Depends}, ${misc:Depends}
Description: ncurses-based Git repository browser
 ...
```

debian/copyright

```
This package was debianized by Sebastian Harl <sh@tokkee.org> on
Thu, 28 Sep 2006 13:09:36 +0200.

It was downloaded from <http://jonas.nitro.dk/tig/releases/>.

Upstream Author: Jonas Fonseca <fonseca@diku.dk>
Copyright Holder: Jonas Fonseca <fonseca@diku.dk>

License:

    Copyright © 2006-2009 Jonas Fonseca <fonseca@diku.dk>

    <GPL header>

On Debian systems, the complete text of the GNU General Public License
can be found in ‘/usr/share/common-licenses/GPL’.

The Debian packaging is © 2006-2009, Sebastian Harl <sh@tokkee.org> and
is licensed under the GPL, see above.
```

debian/changelog

```
tig (0.5-1) unstable; urgency=low

  * Initial release (Closes: #389926).

 -- Sebastian Harl <sh@tokkee.org>  Thu, 28 Sep 2006 13:09:36 +0200
```

debian/rules

Setzt den „Dreisatz“ `configure; make; make install` um, und verpackt das Ergebnis in ein Debian-Paket.

debian/rules

Setzt den „Dreisatz“ `configure; make; make install` um, und verpackt das Ergebnis in ein Debian-Paket.

Ein ausführbares Makefile, mit den folgenden Targets:

- `build`

debian/rules

Setzt den „Dreisatz“ `configure; make; make install` um, und verpackt das Ergebnis in ein Debian-Paket.

Ein ausführbares Makefile, mit den folgenden Targets:

- `build`
- `build-arch`, `build-indep` (optional)

debian/rules

Setzt den „Dreisatz“ `configure; make; make install` um, und verpackt das Ergebnis in ein Debian-Paket.

Ein ausführbares Makefile, mit den folgenden Targets:

- `build`
- `build-arch`, `build-indep` (optional)
- `binary`, `binary-arch`, `binary-indep`

debian/rules

Setzt den „Dreisatz“ `configure; make; make install` um, und verpackt das Ergebnis in ein Debian-Paket.

Ein ausführbares Makefile, mit den folgenden Targets:

- `build`
- `build-arch`, `build-indep` (optional)
- `binary`, `binary-arch`, `binary-indep`
- `clean`

debian/rules

Setzt den „Dreisatz“ `configure; make; make install` um, und verpackt das Ergebnis in ein Debian-Paket.

Ein ausführbares Makefile, mit den folgenden Targets:

- `build`
- `build-arch`, `build-indep` (optional)
- `binary`, `binary-arch`, `binary-indep`
- `clean`
- `get-orig-source` (optional)

debian/rules

Setzt den „Dreisatz“ `configure; make; make install` um, und verpackt das Ergebnis in ein Debian-Paket.

Ein ausführbares Makefile, mit den folgenden Targets:

- `build`
- `build-arch`, `build-indep` (optional)
- `binary`, `binary-arch`, `binary-indep`
- `clean`
- `get-orig-source` (optional)
- `patch` (optional)

Beispiel

Beispiel . . .

Weiterführendes

Finetuning

- Dokumentation (Manpages, . . .) (mehrsprachig!)
- `.menu` und `.desktop` Dateien zum Eintragen in Menüs von Windowmanagern (für graphische Anwendungen)
- `watch` Datei (zum automatisierten Prüfen auf neue Upstream-Versionen)
- Konfiguration des Pakets mit `debconf` (nur wenn wirklich nötig)
- Aufteilen des Pakets
  - `-doc` Paket
  - `-data` Paket
  - . . .
- Mehrere Varianten? SDL, Gtk+, KDE, ncurses?
- . . .

Mehrere Binärpakete

- Mehrere Einträge in `debian/control`
- `.install` Dateien
- `$(MAKE) install DESTDIR=$(CURDIR)/debian/tmp` und `dh_install --sourcedir=$(CURDIR)/debian/tmp --fail-missing`

Bibliotheken

- Schwierig! Erfordert gutes Wissen über die zu Grunde liegenden Prinzipien (ABIs, APIs, SONAME, . . .)
- Aufteilung in `lib*` und `-dev` Pakete, ggf. `-dbg` Paket
- `shlibs` Datei

Nützliche Helfer

- `mc`
- `lintian`
- `debdiff`
- `pbuilder`, `cowbuilder`
- `piuparts`

Links

- **Debian New Maintainers' Guide** (auch in Deutsch)
  `http://www.debian.org/doc/manuals/maint-guide/`
  Als Debian-Paket: `maint-guide-de`
- **Debian Entwickler-Referenz**
  `http://www.debian.org/doc/manuals/developers-reference/`
  Als Debian-Paket: `developers-reference`
- **Debian Policy**
  `http://www.debian.org/doc/debian-policy/`
  Als Debian-Paket: `debian-policy`

Fragen?

History:

- 2009/03/12: fpipe Team Schulung

Dank an ...

- Alexander „Tolimar“ Reichle-Schmehl <tolimar@debian.org> für seine Folien vom Vortrag „Debian-Paket-Bau – Eine kurze Einführung“, die als Grundlage für meine Folien gedient haben