

東京エリア デビアン 勉強会



Debian勉強会幹事 上川純一

2009年11月14日

1 Introduction

上川 純一



今月の Debian 勉強会へようこそ。これから Debian の世界にあしを踏み入れるという方も、すでにどっぷりとつかっているという方も、月に一回 Debian について語りませんか？

Debian 勉強会の目的は下記です。

- Debian Developer (開発者) の育成。
- 日本語での「開発に関する情報」を整理してまとめ、アップデートする。
- 場 の提供。
 - 普段ばらばらな場所にいる人々が face-to-face で出会える場を提供する。
 - Debian のためになることを語る場を提供する。
 - Debian について語る場を提供する。

Debian の勉強会ということで究極的には参加者全員が Debian Package をがりがりとするスーパーハッカーになった姿を妄想しています。情報の共有・活用を通して Debian の今後の能動的な展開への土台として、「場」としての空間を提供するのが目的です。

2009 年の計画は仮です。

1. 新年の企画 (アンサンブル荻窪開催)

2. OSC Tokyo
3. VAIO P インストール記録、カーネル読書会 ディストリビューション大集合 (小林さん)(東京大学?)
4. Git Handson (岩松)(あんさんぶる荻窪?)
5. 家 Debian サーバ vs 職場のネットワーク (千代田区都立図書館?*¹)
6. Asterisk (東京大学?)
7. スペインにて開催
8. Debconf 報告会
9. OSC Fall?
10. udev + HAL(岩松さん)
11. 3D graphics 開発 (藤沢さん)
12. Debian サーバ + VMware + 各種 OS、他の仮想化ツール (vserver etc.)、忘年会

会場候補としては下記があります：

- 大学
- 恵比寿 SGI ホール
- Google オフィス
- 公民館 (あんさんぶる荻窪等)
- 都立会議室 (無線 LAN)
- 健保の施設

*¹ <http://www.library.chiyoda.tokyo.jp/>

会 強 勉 ア ビ ト

目次

1	Introduction	1
2	事前課題	3
3	最近の Debian 関連のミーティング報告	4
4	Debian Trivia Quiz	5
5	Debian での数学ことはじめ。	6
6	Debian autobuilder ネットワーク	8

2 事前課題

上川 純一

事前課題は:

「あなたが普段行っている統計処理の内容とその処理で行っているハックを披露してください。」

この課題に対して提出いただいた内容は以下です。

3 最近の Debian 関連のミーティング報告

前田耕平



3.1 東京エリア Debian 勉強会 56 回目報告

56 回は 9 月 16 日 (水)19:00 からといういつもとかなり変則的な時間で、ミラクルリナックスさんのセミナールームをお借りして、Key Sign Party を行いました。最初に岩松さんによる、なぜヒトはキーサインを求めるのか? というありがたいお言葉をいただき、キーサインは Web of Trust と言いつつも、その実は Web of I know who での信頼レベルなので、パスポートのような強力な公的な身分証明書で己れの存在証明をする必要があるというはなしも途中ありました。100 人もキーサインする相手がいると gpg コマンドでちまちまとやってられんよね? という問題を解決する便利なツール caff の使い方を伝授してもらいました。

その後、実際に参加したメンバー全員で、立食パーティでのキーサインパーティを行いました。
参加された方は皆、リア充できたようです。

3.2 東京エリア Debian 勉強会 57 回目報告

第 57 回は、10 月 30 日 (金)、31 日 (土) に日本工学院大学の蒲田キャンパス 12 号館で OSC 2009Tokyo/Fall でブースのみ出展しました。元々は、31 日のみ出展するでしたが、あけどさんなど有志の方により、30 日 (金) もブースをやっていたとのことでした。

配布物としてはフライヤー 50 部、9 月の勉強会で使用した GPG サインの資料 20 部、Live DVD 20 枚あまりを用意しました。また、展示物として岩松さんが sh7724 搭載マシンを用意してくれ、またそのマシンの詳細を印刷したパンフレットも用意してくれました。配布物の配布結果はほぼ予想通りで、そこそこ足りていました。Live DVD のイメージが当日早朝にできたこともあり、ブースで少し焼きましたが、全部で 25 枚程度でした。

Live USB イメージ (lenny で lxde) も用意しましたが、案内が直前だったこともあるのか、あまり知られておらず、イメージを欲しがるとはそれほどいませんでした。ただし、30 日 (金) にいらした方で、あけどさんに USB メモリを渡したまま帰ってしまった人がいたらしく、あけどさんが ML で連絡を求めています。^{*2}

また、岩松さん提案の GPG 鍵交換ですが、なかなか好評で、ブースおよび懇親会で行ないました。

また、学生との交流もできたので、Debconf in Japan や出張 Debian 勉強会・Debian 温泉の企画にも進展がありそうです。

懇親会は JR 蒲田駅東口近くの「春香園」という餃子屋さんで行ないました。懇親会の参加者は、武藤さん、上川さん夫妻、日比野さん、本庄さん、谷口さん、岩松さん、藤澤さん、山本さん、前田の 10 名でした。

^{*2} <http://lists.debian.or.jp/debian-users/200911/msg00001.html>
<http://lists.debian.or.jp/debian-devel/200911/msg00000.html>

4 Debian Trivia Quiz

上川 純一



ところで、みなさん Debian 関連の話題においついていますか？ Debian 関連の話題はメーリングリストをよんでいると追跡できます。ただよんでいるだけではりあいがないので、理解度のテストをします。特に一人だけでは意味がわからないところもあるかも知れません。みんなで一緒に読んでみましょう。

今回の出題範囲は debian-devel-announce@lists.debian.org に投稿された内容と Debian Project News からです。

5 Debian での数学ことはじめ。

まえだこうへい



5.1 はじめに

Debian で数値計算や統計処理、グラフ作成を行うには、様々なパッケージがあります。私のように大学が理系といても基礎生物学専攻だった人間には、コンピュータでの専用ツールでなくても、極端な話、基本的な統計学と関数電卓があれば事足りました。電卓代わりに Excel® や OOo の calc のようなスプレッドシートを使う、というやり方でやっていました。社会人になってからもスプレッドシートで大抵は事足りる、という方が一般的には多いのではないかと思います。

とはいえ、結構なサンプル数の計算をしたり、グラフを描写する場合、スプレッドシートでは非力だということを実感しています。プリセールスや企画で裏付け資料を作るのにサンプルデータを統計処理して、グラフを描くと、スプレッドシートだとマウスの操作だけで簡単にできる一方、CPU やメモリ不足でレスポンスが遅くなったり、ソフトウェア自体が落ちたり、酷いと OS 自体のレスポンスが悪化するというのが最近の仕事上での悩みだったりします。

そんなことがきっかけで最近、Gnuplot を使い始めたわけですが、統計処理では最近では GNU R の書籍を書店で見かけたり、数値計算では Octave というのがあるということを知ったり^{*3}と、自分の中だけでなく、最近意外とまわりでもこの辺はホットな話題ではないかと思います。そこで、今回は Octave, GNU R を中心に、Gnuplot との連携方法を交えながら、Debian での使い方、パッケージングなどについて説明しようと思います。

5.2 インストール方法

5.2.1 Octave のインストール方法

Octave は Squeeze/Sid では、バージョン 3.0 と 3.2 で別のパッケージになっています。今回は 3.2 のパッケージを導入してみます。

FIXME 3.0 と 3.2 の違いを簡単に記載すること。

```
$ sudo apt-get install octave3.2
```

5.2.2 GNU R のインストール方法

GNU R は、Debian パッケージでの名前は、gnur でも、r でも gnu-r でもありません。r-base という名前になっているので、その点注意が必要です。

```
$ sudo apt-get install r-base
```

^{*3} GNU R や Octave のパッケージメンテナである Dirk Eddelbuettel さんが近々来日されるというのが ML で流れていたそもそものきっかけです。http://dirk.eddelbuettel.com/

5.2.3 Gnuplot のインストール

Gnuplot は Octave、GNU R のいずれも依存関係はありませんが、インストールしておくとう便利です。

```
$ sudo apt-get install gnuplot
```

5.3 試してみる。

5.3.1 Octave の場合

5.3.2 GNU R の場合

5.3.3 Gnuplot との連携

5.4 deb パッケージ化する。

Debian では Octave も GNU R も関連パッケージがかなりあります。

```
$ apt-cache search "GNU R" | wc -l
213
$ apt-cache search octave | wc -l
133
```

それぞれ、deb パッケージにする場合の方針、考慮点を見てみましょう。

5.4.1 Octave の場合

5.4.2 GNU R の場合

GNU R では

5.5 実際に使ってみる。

5.5.1 データを用意する。

5.5.2 計算してみる。

5.5.3 プロットしてみる。

5.5.4 応用してみる。

5.6 まとめ。

6 Debian autobuilder ネットワーク

岩松



Debian は現時点で 11 個の CPU アーキテクチャと 2 つのカーネルをサポートしています。そしてユーザが知らないうちに各アーキテクチャ毎の Debian パッケージがビルドされ、利用できるようになっていきます。これはどのようなになっているのでしょうか。もちろん、パッケージメンテナはすべてのアーキテクチャを持っているわけではないので(持っている人もいるみたいですが)、パッケージメンテナがビルドしているわけではありません。Debian ではサポートしているアーキテクチャ向けの Debian パッケージが自動的にビルドするシステム Autobuilder ネットワークによってパッケージが自動的にビルドされています。Debian Project では重要なシステムの一つなのですが、表に出てくる事も少ないためどのような動きになっているのか知る人は少ないです。今回、Renesas SuperH(SH4) を移植するにあたりこれらの情報を入手したのでまとめてみました。

6.1 Autobuilder ネットワークのコンポーネント

Autobuilder ネットワーク は一つのソフトウェアで動いているわけではなく、いくつかのソフトウェアと porter と呼ばれる buildd のメンテナがいます。Autobuilder ネットワークの動きを説明する前に、各コンポーネントがどのような動きをするのか理解しておく必要があります。以下に主要な動作を説明します。

6.1.1 quinn-diff

パッケージメンテナによって新しくアップロードされたソフトウェアのアーキテクチャ依存のパッケージ (Architecture: any) とアーキテクチャ非依存のパッケージ (Architecture: all) をチェックし、前者を wanna-build のデータベースに登録するツールです。これはソースパッケージだけでなく、バイナリパッケージもチェックします。Autobuilder ネットワークの過程では アーキテクチャ非依存のパッケージはビルドされません。

6.1.2 wanna-build

buildd とデータのやりとりをするためのインターフェイスです。各アーキテクチャ毎にパッケージのデータベースを持ちます。buildd は wanna-build のデータベースを持つサーバに ssh でログインし、wanna-build コマンドを使ってデータベースをコントロールします。

6.1.3 buildd

wanna-build と データをやりとりするデーモンやパッケージをアップロードするツール一式を指します。ツールには以下のものがあります。

- buildd

builddd デーモン。wanna-builddd のデータベースを定期的にチェックし、ビルドが必要なパッケージをパッケージビルドキューとして管理します。そして、sbuild へのビルド指示を行います。

- builddd-mail
メールキューディレクトリにあるデータを処理するプログラム。ビルドに成功したパッケージ (*.deb, *.udeb) とパッケージ情報 (*.dsc, *.changes, etc) をアップロードキューディレクトリに移動させます。
- builddd-mail-wrapper
builddd-mail-wrapper は porter から送られたメールをメールキューディレクトリに格納し、builddd-mail を呼び出すプログラムです。入力データとして、メールデータそのものを要求するので、procmail など合わせて利用します。
- builddd-update-chroots
パッケージのビルドを行う chroot 環境をアップデートおよびクリーンアップするプログラムです。builddd が動作している場合には builddd を停止し、chroot をアップデートします。アップデート完了後、再度 builddd を起動させます。クリーンアップには debfoster と呼ばれるゴミパッケージを消すプログラムが利用されています。
- builddd-uploader
アップロードキューディレクトリにあるパッケージをアップロードするプログラムです。dupload を使い、GPG サインされている *.changes ファイルをパーサし、パッケージをアップロードします。cron 等で定期的に呼び出して利用します。
- builddd-watcher
builddd の動きをチェックするプログラム。cron 等で定期的に呼び出して利用します。

6.2 sbuild

実際にパッケージをビルドするプログラムです。builddd から呼ばれます。パッケージのソースコードを取得し、ビルド結果を porter に送信するまでの処理を行います。pbuilder と機能がかぶりますが、sbuild はアーキテクチャ依存部分のビルドに特化したパッケージビルドシステムになっています。schroot を利用することにより、lvm などディスク管理機能にも対応もされています。

6.3 schroot

sbuild から呼ばれる chroot 代替プログラムです。chroot の高機能版です。

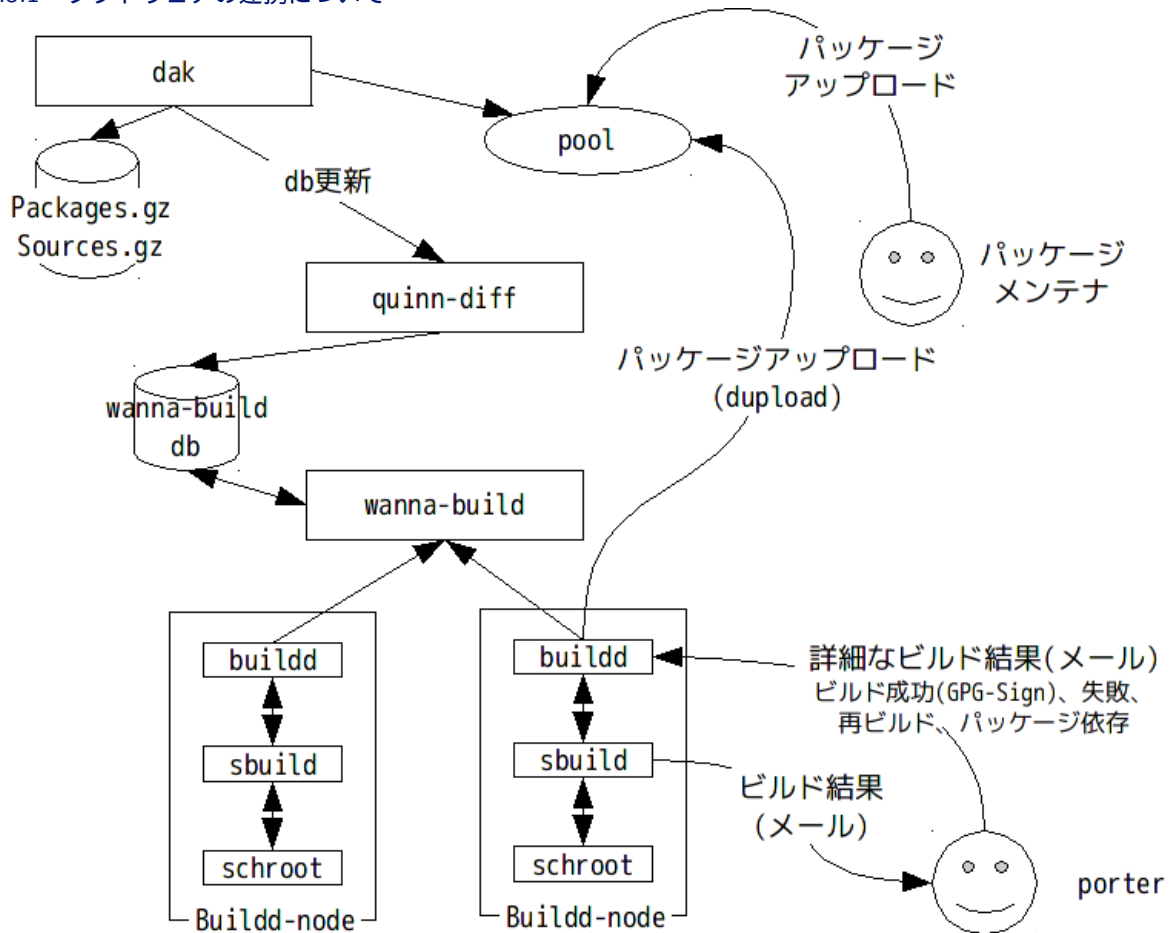
6.4 porter

builddd をメンテナンスする人です。ビルド結果をメールとして受信し、その結果を builddd に対して返信します。ビルドに成功したパッケージのメールには GPG でサインして返信します。この返信する際には、送られてきたメールから、*.changelog の部分を抜き出して、その抜き出した部分にサインを行います。抜きだしの部分にはスクリプトを使って処理するのですが、現在のところ mutt のスクリプトしかないので、ほとんどの porter は Mutt ユーザーです。

6.5 Autobuilder ネットワークの動き

Autobuilder ネットワークの動きとして、ソフトウェアの連携と、パッケージビルド時の状態遷移にわけることができます。

6.5.1 ソフトウェアの連携について



6.5.2 パッケージビルド時の状態遷移

パッケージは常になにかしらの状態をステータスを持っています。ステータスは wanna-build で管理する DB で保持されています。ステータスは以下のものがあり、図 1 のように変化します。

- BD-Uninstallable
パッケージのビルドに足りないパッケージがある状態。
- Needs-Build
パッケージがビルド可能な状態。パッケージビルド依存関係が解消されてたり、ビルドの状態から戻された (given-back) 場合にこの状態になります。
- Building
Buildd によってパッケージがビルドされている状態。
- Uploaded
Porter によってパッケージに GPG サインされ、アーカイブにアップロードされた状態。
- Installed
アーカイブにインストールされた状態。
- Dep-Wait
パッケージのビルド依存関係待ちの状態。
- Failed
パッケージがビルドに失敗した状態。
- Not-For-Us

対象のアーキテクチャではビルドできないパッケージの状態。porter が wanna-build 経由でデータベースに登録します。現在、wanna-build のデータベースとは別にアーキテクチャ依存のリスト*4を作成中です。

- Failed-Removed Failed 状態のパッケージが一時的に削除されている、という状態。

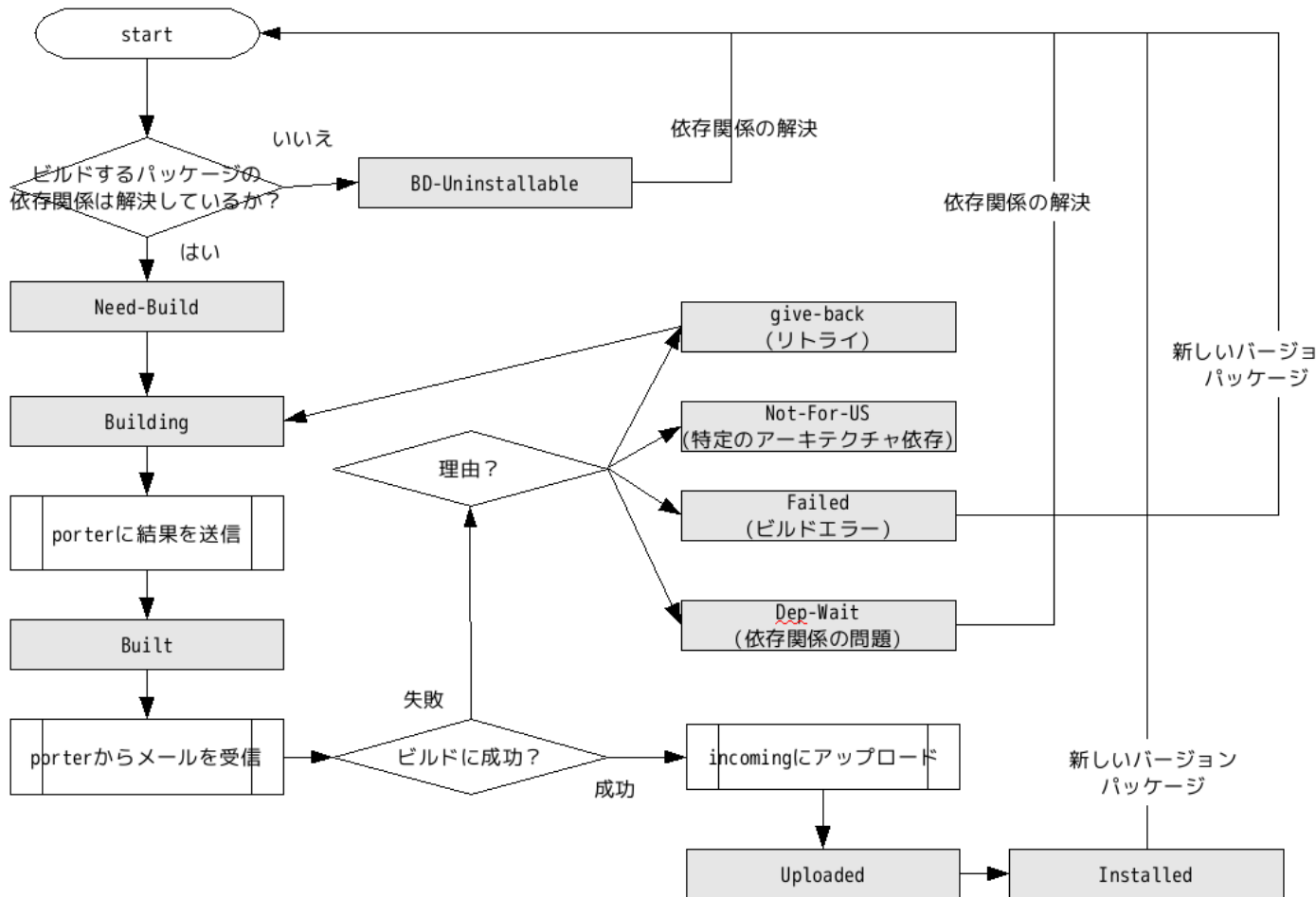


図1 パッケージステータスの状態遷移

6.5.3 ビルドの優先順位

パッケージのビルドは基本的に Need-Build のステータスになった順かつパッケージ名 ASCII 順でビルドされますが、それとは別にソースパッケージのプライオリティとソースパッケージのセクション毎に優先順位が設定されています。現在では、以下の優先順位になっています。

6.6 新しいアーキテクチャをサポートするには

昔は、ポーティングしたい人が quinn-diff から buildd(buildd-node) まで全て構築する必要がありました。この最大のネックは quinn-diff と wanna-build DB で、これらは利用するパッケージデータは最新のものを使う必要があったのですが、DB のありかは公開されていない (参照するには権限が必要) かつ、最新の情報を取得するにはサー

*4 <http://buildd.debian.org/git/packages-arch-specific.git>

Source Priority	値
required	-5
important	-4
standard	-3
optional	-2
extra	1
unknown	-1

表1 ソースパッケージプライオリティ毎の優先順位

libs	-200
debian-installer	-199
base	-198
devel	-197
shells	-196
perl	-195
python	-194
graphics	-193
admin	-192
utils	-191
x11	-190
editors	-189
net	-188
mail	-187
news	-186
tex	-185
text	-184

表2 ソースパッケージセクション毎の優先順位:その1

web	-183
doc	-182
interpreters	-181
gnome	-180
kde	-179
games	-178
misc	-177
otherosfs	-176
oldlibs	-175
libdevel	-174
sound	-173
math	-172
science	-171
comm	-170
electronics	-169
hamradio	-168
embedded	-166
unknown	-165

表3 ソースパッケージセクション毎の優先順位:その2

アーキテクチャ名	進捗
avr32	65.52%
hurd-i386	64.46%
m68k	53.00%
sh4	85.43%

表4 debian-ports.org でサポート中のアーキテクチャ

パーアカウントが必要だったため、構築が困難でした。しかし、今は新しいアーキテクチャを追加するためのサポート用サービス `debian-ports.org` が用意されています。このサービスでは、ポーティングしたいアーキテクチャ用の `wanna-build` 用アカウントとDB, `quinn-diff` のサービスを提供できるようになっています。これにより、移植したい人は `buildd-node` のみを用意し、`debian-ports` の管理者に登録申請をするだけで新しいアーキテクチャをサポートできるようになりました。そして現在、`debian-ports.org` を使っているアーキテクチャは以下のとおりです。

6.7 loop-depends の対応方法

いくつかのパッケージには `loop-depends` になっているものがあります。`loop-depends` とは、ビルド依存がループしているパッケージの状態を言います。例えば、`libpang-perl` パッケージは `libgtk2-perl` パッケージにビルド依存しています。しかし、`libgtk2-perl` は `libpang-perl` パッケージに依存しています。

```
Source: libpango-perl
Section: perl
Priority: optional
Build-Depends: debhelper (>= 7.0.50), libcairo-perl (>= 1.000),
  libextutils-depends-perl (>= 0.300), libextutils-pkgconfig-perl,
  libglib-perl (>= 1:1.220), perl, xvfb, libpango1.0-dev,
  xauth, xfonts-base, quilt, libgtk2-perl (>= 1:1.220)
```

```
Source: libgtk2-perl
Section: perl
Priority: optional
Build-Depends: debhelper (>= 7.0.50), quilt (>= 0.46-7), perl,
  libextutils-depends-perl (>= 0.300), libextutils-pkgconfig-perl (>=
  1.03), libgtk2.0-dev (>= 2.6.0), libglib-perl (>= 1:1.220),
  libcairo-perl (>= 1.00), xvfb, xauth, xfonts-base, hicolor-icon-theme,
  libpango-perl (>= 1.220), shared-mime-info
```

これではいつになってもビルドできません。よって、手を加えてビルドする必要があります。手を加えた場合は正式なパッケージとして扱われません。このようなパッケージに遭遇した場合には、unreleased ディストリビューションにパッケージをアップロードし、そのパッケージを使って再度ビルドを行うという方法をとります。

以下に例として libpango-perl をビルドする場合の手順を説明します。

1. buildd が動作していない別マシンでの作業

(a) libpango-perl のソースパッケージをダウンロードします。

(b) debian/control ファイルの Build-depends から libgtk2-perl を削除します。

(c) debian/changelog をアップデートします。

Debian version は既存のバージョンに +arch-name などをつける必要があります。(実際の Debian package と区別するため。) ディストリビューション名は unreleased に変更します。debian/changelog の変更者も変更する必要があります。

(d) debuild -m"Your name <your-name@example.org>" などとしてパッケージをビルドします。

(e) dupload を使ってアップロードします。

(f) dak で処理されれば、unreleased ディストリビューションから libpango-perl が利用できるようになります。

2. schroot 内での作業

(a) schroot 環境の apt-line に unreleased を追加します。

(b) schroot 内で apt-get update を実行します。

3. libgtk2-perl がビルドできるようになります。

4. libgtk2-perl ができたら、unreleased ディストリビューションから libpango-perl を消してもらいます。

5. libpango-perl が本来の Debian Version のものでビルドされます。

毎回この方法がうまくいくとは限りません。例えば、依存するパッケージや提供されるパッケージの機能によっては configure のオプションを変更する場合があります。このあたりはパッケージメンテナか、debian-wb@l.d.o に直接聞いたほうがよいです。聞かないとわからないというのは問題なので、README.Debian あたりに書いてもらうように計画中です。



Debian 勉強会資料

2009年11月14日 初版第1刷発行
東京エリア Debian 勉強会（編集・印刷・発行）
