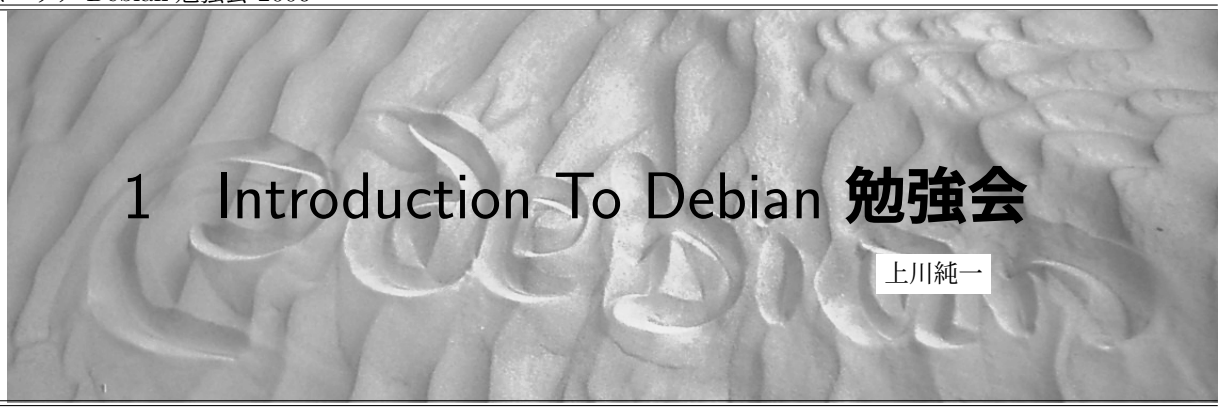
第 22 回 東京エリア Debian 勉強会
事前資料

Debian 勉強会会場係 上川純一*

2006 年 11 月 19 日

* Debian Project Official Developer

目次



1 Introduction To Debian 勉強会

上川純一

今月の Debian 勉強会へようこそ。これから Debian のあやしい世界に入るという方も、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？

目的として下記の二つを考えています。

- メールではよみとれない、もしくはよみとってられないような情報を情報共有する場をつくる
- まとまっていない Debian を利用する際の情報をまとめて、ある程度の塊として出してみる

また、東京には Linux の勉強会はたくさんありますので、Debian に限定した勉強会にします。Linux の基本的な利用方法などが知りたい方は、他でがんばってください。Debian の勉強会ということで究極的には参加者全員が Debian Package をがりがりと作りながらスーパーハッカーになれるような姿を妄想しています。

Debian をこれからどうするという能動的な展開への土台としての空間を提供し、情報の共有をしたい、というのが目的です。次回は違うこと言ってるかもしれませんが、御容赦を。

1.1 講師紹介

- 上川純一 宴会の幹事です。

1.2 事前課題紹介

今回の事前課題は「Debian 勉強会で知りたいこと、今日の質問」というタイトルで 200-800 文字程度の文章を書いてください。というものでした。その課題に対して下記の内容を提出いただきました。

1.2.1 山下さん

Debian 勉強会で知りたいことは，メーリングリストや IRC では普段聞けない，裏事情など．8 月に東京で参加したときに，みなさんがときどき，ぼそっと言われる発言が非常に参考になったので．

気になっていた事なんですが，Windows 上の Eclipse に比べて，Debian 上の方が若干重たく感じます．Java 関係の問題だと思うのですが．今後，関西での Debian 勉強会の頻度について．

1.2.2 谷口さん

普段、windows をメインに使っていまして、サブとして redhat、fedora、vine などの redhat 系の os あるいは freebsd を使っている、あるいは使ったことがあるのですが、debian を使ったことは無く、debian の知識はほとんどありません。本勉強会で debian の特徴やその良さについて知ることが出来ればと思っています。特にパッケージ関連の話について詳しく知りたいです。質問に関しては、その場で気になった点を質問させていただくと思います。

1.2.3 岩本さん

debian では etch から文字コードが UTF-8 が標準になると聞いております、UTF-8 化によるメリット、デメリットやインストゥール時やアプリケーション動作時に気をつけるべきところなどがありましたら教えていただきたく思います。

また、最近のノートパソコンに debian each をインストゥールするとすれば、この勉強会に参加されている方はどんな機種を選択するのかお聞きしたい。（ノートパソコン購入時の参考にしたい為。）

1.2.4 榎　真治さん

普段あまり Debian を使いこなせていないのですが、特有の流儀というものがあると感じております。

勉強会に参加することでそれを知る手がかりになればと思っております。

1.2.5 Yutaka Kametani さん

たくさんのディストリビューションがある中で、デビアンを使うメリットは何ですか？

1.2.6 畑中さん

研究室配属時に、Debian を使ってもらうと先生からいわれ、とりあえず自分のコンピュータにインストールをしてみました。これが初めての純正 Debian です。他のディストリビューションにあるような GUI インストールや、GUI での apt など (実はあったらごめんなさい) がなく、Linux 初心者にインストールさせるには、多少敷居が高いと思えました。そのあたりの開発はされているのでしょうか。

1.2.7 Takashi Hamabe さん

私は debian を使い始めて間もなく、自分自身まだまだ勉強が足りないと思っています。そのことで大変な失礼があるかと思いますが、恥ずかしながら質問したいと思います。どうして debian を開発しようと思ったのでしょうか？どうして使う側ではなく、開発側にまわろう思ったのでしょうか？開発側にまわるためには、やはり大変な努力と知識が必要だと思いますが、何がそういう努力をさせるような情熱を与えてくれるのでしょうか?

1.2.8 藤澤 恵一朗さん

Debian を使ってみて最初に思ったのは X が軽いなあという点です。学科指定の Fedora3 や FreeBSD ではもっさり感があったのですが、Debian は快適です。普通にコンパイルしただけでなくて、何かチューニングしているのでしょうか。

apt でいれた package が確実に削除できるのがすごいなあと思いました。心おきなくいろんなものを試せるので勉強もしやすいです。

違和感があったのは、apache や proftpd などは apt でいれると必ず自動起動するという点です。FreeBSD では設定をしないと起動しなかったので、apache のバージョン違いをいれたりしても動作的には問題がなかったです。しかし Debian はどういった挙動をとるのかわからないので不安です。Debian の勉強不足というのが一番の理由ですが、なぜこうしているのか気になります。

1.2.9 乾さん

Debian は最近使い始めたのでまだよくわかっていません。今回の勉強会では、debian の魅力について教えてもらえたらと思います。できれば他と比べての欠点も知りたいです。

1.2.10 久松さん

2 点あります。

1. ネットワークの設定方法
   ネットワークが変わるたびに、 /etc/init.d/netwoking stop; /etc/network/interfaces の書き換え、/etc/init.d/netwoking start と、shell script を使ってやっています。我ながらこれは美しくないと思っており、美しいやり方を教えて頂きたいです。
2. apt、aptitude の使い方
   私は、パッケージをインストール、または削除するときに、apt、aptitude で目的のパッケージを選択しづらく、dselect のみを使っています。apt は、パッケージ名を正しく書く必要があり、使いづらいです（パッケージ名を書き間違えることが、かなりある）。aptitude は、パッケージがどの分類に属すか、分からないときがある（意外と間違える）。また、文字列の検索でパッケージを選択できるが、文字列がヒットしたパッケージ名が、”Search for:” というウィンドウの陰に隠れていて、使いにくい (もっと文字列を追加するべきなのか？これで十分なのか、判断できない)。dselect は、インクリメンタルサーチが使えたら、特になにもいうことはないです。おそらく、私の使い方が間違っていると思うので、apt、aptitude の使い方をご教授いただけたら幸いです。

1.2.11 岩松さん

- Debian 開発者の開発環境（家の PC は 15 台あって、xDSL 回線は 3 本ありますとか。）紹介とか、いつごろ寝ているのか、机の上はどんな汚さなのか、公開できる範囲の私生活を知りたいです。
- 関西での Debian 具合
  関西出身の自分としては関西での Debian の浸透っぷりを知りたいです。

1.2.12 上川

今日の私の興味は、Debian 勉強会を関西で開催できるのか、どういう人がいるのか、ということを確認することです。Debian について活発に関西でもイベントなど開催されるとよいなぁ、と希望しています。

2 Debian Weekly News trivia quiz

上川純一

ところで、Debian Weekly News (DWN) は読んでいますか？Debian 界隈でおきていることについて書いている Debian Weekly News. 毎回読んでいるといろいろと分かって来ますが、一人で読んでいても、解説が少ないので、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 DWN を読んでみましょう。

漫然と読むだけではおもしろくないので、DWN の記事から出題した以下の質問にこたえてみてください。後で内容は解説します。

2.1 2006 年 40 号

`http://www.debian.org/News/weekly/2006/40/` にある 10 月 31 日版です。

問題 1. Frank Küster がカーネル 2.6.18 パッケージについて発表したのは？

☐ A まだ安定していないけどどんどん利用してください

☐ B General Resolution の結果、 Linux じゃないカーネルを今後利用する

☐ C firmware blob を Debian package に含めるようにした

問題 2. mplayer パッケージになにがおきたか？

☐ A NEW キュー滞在時間の新記録をさらに更新した

☐ B Debian unstable に入った

☐ C もうあきらめることになった

3 最近の Debian 関連のミーティング報告

上川純一

3.1 東京エリア Debian 勉強会 21 回目報告

東京エリア Debian 勉強会報告。10 月の第 21 回 Debian 勉強会を実施しました。Extramadura の報告、flash の紹介、および apt のプロファイリングの結果について報告しました。

今回の参加人数は 23 人くらいでした。

参加者は小林さん、柏木さん、倉澤さん、前田さん、須藤さん、武藤さん、やまねさん、野首さん、小室さん、高杉さん、えとーさん、吉田@板橋さん、八田さん、キタハラさん、ごとうまさのりさん、中野さん、ysjj さん、松山さん、河内さん、David Smith　さん、Charles Plessy さん、上川でした。

まず、事前課題の紹介をしました。ネットワークの設定については、皆思うところがあるようです。GUI のツールなどが実は整備されていて、それをみんなしらないだけなのではないか？という話が出ました。また、NTLM 認証のプロキシに対応していないアプリケーションが多数あるんじゃないか、HTTPS が重要なんじゃないのか、という話題が出ました。Debconf の会場が NTLM Proxy 必須で、さらにスケジュールが HTTPS RSS で流れるようになっていたら、Debian のそこらへんのサポートも改善するんじゃないか、という提案が出てみたり。

恒例のクイズ、今回 5 問しかありませんでした。Debian Weekly News が休止してしまったのです。今後クイズのネタがなくなって非常に困りますが、どうしましょう？

武藤さんが Extramadura i18n 会議について発表しました。pootle はまだ即使えるという状況ではないということは残念ですが、今後タスクフォースとしてまとまっていろいろとやっていこうという気概と雰囲気がかんじられるので期待です。

松山さんに flash ming ネタを発表してもらいました。ming を使うことはできるのですが、結構苦戦した、という話でもりあがりました。野首さんが、 ming ネタについてあつく語っていました。一子相伝の技の次世代への継承が行われているところに出くわした感じです。

上川が apt のプロファイリングのネタについて発表しました。SHA1 のチューニングについては、意外とできないものだ、という結論で終わりました。

今回宴会は「えん屋」で行いました。店はけむたかったけど、みなさんといろいろと話ができて、おもしろかったです。

4 パッケージングについて

岩松さん

4.1 パッケージングについての基礎知識

パッケージングとは、無数に存在するソフトウェアの配布したりソフトウェアのバージョンを管理するために、ファイルなどをまとめたものです。それらをサポートするシステムがあり、パッケージ管理システムといいます。

例えば、あるフリーソフトウェアを入手し、使おうとした場合、

```
% ./autogen.sh
% ./configure
% make
% make install
```

として、ライブラリ等をチェック、コンパイル、インストールという作業を行う必要があります。これらの過程でライブラリのチェックで新たにライブラリが必要だったり、コンパイルできなかったりする事が多々あります。もし、パッケージングシステムを使って、ソフトウェアを構成していた場合は、これらの問題を解決したり、ソフトウェアを容易に使用することができるようになります。

パッケージングシステムの必要な機能として

- パッケージに関する情報
- パッケージの作成
- パッケージのインストール
- パッケージの更新
- パッケージのアンインストール

があります。

ソフトウェアをパッケージングし、インストールやアンインストール、ソフトウェアのアップデートなどを容易に行う事が目的です。

ここで重要なのが、ユーザーだけが容易に使用できるということではなく、開発者側としても容易にパッケージングを行い、パッケージをメンテナンスできるということです。

4.2 Debian でのパッケージ管理/Debian パッケージについて

4.2.1 dpkg

Debian では dpkg と呼ばれる Debian パッケージ管理システムを使用しています。この dpkg を Debian の基礎としているため、Debian で配布されている全てのパッケージは .deb ファイル形式で提供されてなければなりません。

dpkg では

1. 依存関係の解決
   - Depends 依存
   - Recommends 推奨

- Suggests 提案
- Conflicts 競合
- Provides 提供
- Replaces 置換

2. パッケージバージョンによるアップグレードのサポート
3. インストール前後、アンインストール前後の設定機能

などが提供されています。

4.2.2 Debian Policy

Debian で配布されるパッケージは、厳格なパッケージポリシー Debian Policy [*1] に基づいて作成されている必要があります。
この Debian Policy によって、パッケージの互換性が保たれています。

4.2.3 パッケージの種類

Debian で配布されるパッケージは、DFSG (Debian Free Sofware Guideline) によって 3 つのセクションに分けられます。DFSG は Debian として Free Software とはどのようなものなのか、定義するものです。

1. main
   DFSG に適合したパッケージは main セクションに置かれます。
2. contrib
   DFSG に適合しているが、non-free なパッケージに依存しているパッケージは contrib セクションに置かれます。
3. non-free
   DFSG に適合しないパッケージは non-free セクションに置かれます。non-free のパッケージは Debian の一部ではありません。

さらにこれらのパッケージを用途に応じて分けられています。

4.2.4 apt

最近は dpkg を直接使い、パッケージをインストールすることは少なくなっています。変わりに APT[*2]を使うようになりました。その理由として、提供されるパッケージが増え、依存関係が複雑になって必要なパッケージをダウンロードするのが大変になってきためです。そこで開発されたのが、パッケージをまとめて管理する apt です。dpkg 用のフロントエンドになっており、中では dpkg が呼ばれています。
適当な図を入れる。
APT の役目として

- パッケージの管理
  パッケージをどのようにインストール、アンインストールをするか考える。
- パッケージをダウンロードする。
- パッケージの検索を行う。

があります。Debian Policy に基づいて 作成されているからこそ実現できています。。

[*1] http://www.debian.org/doc/debian-policy
[*2] Advanced Package Tool)

4.3 Debian でパッケージを作る際のツール群

Debian でパッケージングを行うためのサポートするソフトウェアがあります。パッケージングを行い、品質の高いパッケージを作るために以下のソフトウェアが提供されています。

4.3.1 dpkg-dev

このパッケージには Debian ソースパッケージを展開、構築、アップロードするために必要なツール群をまとめたものです。ソースパッケージの展開に必要な dpkg-source や パッケージの作成に必要な dpkg-buildpackage が入っています。

4.3.2 debhelper

dh_xxx というパッケージ作成をサポートツールをまとめたものです。Debian 魔窟のひとつです。

4.3.3 devscript

debuild などのパッケージ作成フロントエンドが提供されています。

4.3.4 dh-make

ソースパッケージの雛型を作るツールです。ソースパッケージやバイナリパッケージを作成するために最低限必要なファイルを生成してくれます。perl や php 用の雛型を作成する dh-make も存在します。

4.3.5 lintian

Debian パッケージ用のチェッカーです。Debian Policy にあわせて作られています。linda というパッケージ用チェッカーもあります。lintian は Perl で、linda は Python でプログラミングされたものです。

4.3.6 fakeroot

fakeroot は root 権限をシミュレートします。パッケージは、root の所有権でファイルがインストールされている必要があります。fakeroot を使用することによって、root にならずにパッケージを構築できます。

4.3.7 cdbs

Common Debian Build System。dh_xxx などをまとめて、簡潔にパッケージ作成スクリプトを記述することができるようにするためのソフトウェア。

4.3.8 GnuPG

作成されたパッケージにサインするために使います。そのパッケージがたしかにそのメンテナの PGP key によって作られていることを証明するためです。

4.3.9 dpatch

Debian のソースパッチを管理するツールです。Debuan パッケージの差分は *.diff.gz という差分ファイルとして管理されるため， どの部分がどういうパッチであるかということを管理していません。 その部分を実装するのが dpatch です。

4.4 実際にパッケージを作成してみる

Debian 用のパッケージを作成する方法を簡単に説明します。対象となるソフトウェアは cairo-dock[*3] という Mac OS X Dock 風の Dock アプリケーションです。

Debian では[*4]開発元、オリジナル配布元のことを Upstream といいます。

1. パッケージ化を行うためにパッケージをインストールします。

```
# apt-get install dh-make devscripts debhelper dpkg-dev dpatch
```

2. ソフトウェアのソースコードをダウンロードし、展開します。

```
% wget http://www.gnome-dock.org/prerelease/cairo-dock-0.0.1b.tar.gz
% tar -xzf cairo-dock-0.0.01b.tar.gz
```

3. 展開した後、ディレクトリ名を パッケージ名-パッケージのバージョン になるようにに修正します。

```
% ls -l
drwxr-xr-x  2 iwamatsu iwamatsu    1024 2006-08-09 01:29 cairo-dock
-rw-r--r--  1 iwamatsu iwamatsu  107560 2006-08-09 01:30 cairo-dock-0.0.1b.tar.gz

% mv cairo-dock cairo-dock-0.0.1b
```

すでに行われている場合は行う必要はありません。

4. バックアップファイルや実行ファイルが残っているので、消しておきます。

```
% ls
Makefile       clock.svg        folder.svg         lowfat.svg           start-cairo-dock.sh~  terminal.svg
cairo-dock     configure.scan   gnome-fs-home.svg  movies.svg           sticky-notes.svg      user-home.svg
cairo-dock.c   development.svg  im.svg             music.svg            stop.svg              user-trash-full.svg
cairo-dock.c~  editor.svg       lockscreen.svg     search.svg           tango-colors.h        web-browser.svg
chat.svg       email.svg        logout.svg         start-cairo-dock.sh  tango-colors.h~
% make clean
```

5. 対象のソフトウェアのディレクトリに移動し、dh_make –createorig を実行します。

   dh_make を実行したときに、パッケージの種類を選択します。

   - single binary

     ひとつのバイナリパッケージを作成する。
   - multiple binary

     複数のバイナリパッケージを作成する。
   - library

     ライブラリ用のパッケージを作成する。
   - kernel module

     カーネルモジュール用のパッケージを作成する。
   - cdbs

     cdbs (Common Debian Build System) を使ったパッケージを作成する。

   実行すると、debian ディレクトリが作成されます。–createorig を指定しない場合はオリジナル用のディレクトリ (今回の場合は cairo-dock-0.0.1b.orig) が作成されません。このディレクトリがない場合は、ソースパッケージの一部として配布される .orig.tar.gz が生成されません。

[*3] http://www.gnome-dock.org/trac

[*4] パッケージを管理しているディストリビューションでも使われています。

```
% dh_make --createorig

Type of package: single binary, multiple binary, library, kernel module or cdbs?
 [s/m/l/k/b] s

Maintainer name : Nobuhiro Iwamatsu
Email-Address   : hemamu@t-base.ne.jp
Date            : Fri, 17 Nov 2006 07:30:18 +0900
Package Name    : cairo-dock
Version         : 0.0.1b
License         : blank
Type of Package : Single
Hit <enter> to confirm:
Done. Please edit the files in the debian/ subdirectory now. You should also
check that the cairo-dock Makefiles install into $DESTDIR and not in / .
```

6. debian ディレクトリ内を編集します。dh_make を行ったあとの debian ディレクトリは以下のようになっています。これらはテンプレートファイルなので、パッケージによって必要のないものも含まれています。
   - README.Debian
     Debian 固有の README
   - changelog
     Debian 固有の変更履歴
   - copyright
     ソフトウェアのライセンスとコピーライト
   - docs
     /usr/share/doc にインストールされるファイルのリスト
   - emacsen-startup.ex
   - emacsen-install.ex
   - emacsen-remove.ex
     xemacs 用テンプレート
   - postinst.ex
   - postrm.ex
   - preinst.ex
   - prerm.ex
     インストール、アンインストール時に実行されるスクリプトテンプレート
   - cairo-dock-default.ex
     /etc/init.d/用のテンプレート
   - compat
   - cron.d.ex
     crond 用のテンプレート
   - init.d.ex
     /etc/init.d/用のテンプレート
   - rules
     パッケージ作成用 Makefile
   - cairo-dock.doc-base.EX
     docbook 用のテンプレート
   - control
     パッケージのメタ情報
   - dirs
   - manpage.xml.ex
   - manpage.sgml.ex
   - manpage.1.ex
     manpages のテンプレート

- menu.ex
  menu システム用テンプレート
- watch.ex
  upstream 監視用設定テンプレート

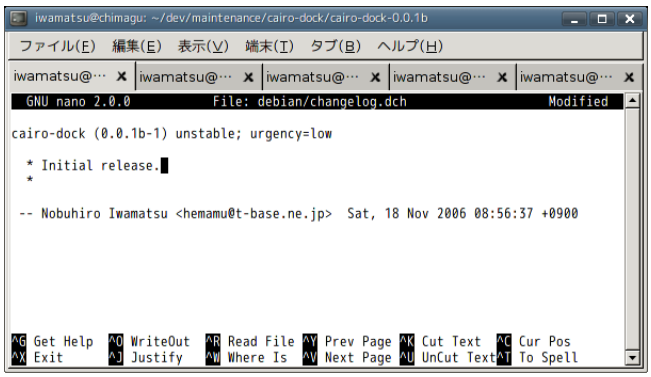
7. *.ex および *.EX ファイルを削除します。
8. control ファイルの変更
   - Source
     ソースパッケージ名
   - Section
     パッケージの種類
   - Priority
     パッケージの優先度
   - Maintainer
     パッケージメンテナ名とメールアドレス
   - Build-Depends
     パッケージのコンパイルするため依存するパッケージ
   - Standards-Version
     Debian Policy バージョン
   - Package
     バイナリパッケージ名
   - Architecture
     アーキテクチャ依存 (any/各アーキテクチャ)、非依存の指定 (all)、
   - Depends,Recommends,Suggests,Conflicts,Provides,Replaces
     パッケージの依存関係の指定
     コンパイルオプションなどから調査します
   - Description パッケージの簡単な説明と詳細な説明

```
Source: cairo-dock
Section: x11
Priority: optional
Maintainer: Nobuhiro Iwamatsu <hemamu@t-base.ne.jp>
Build-Depends: debhelper (>= 5) ,libcairo2-dev ,libgtk2.0-dev ,librsvg2-dev ,libglitz-glx1-dev
Standards-Version: 3.7.2

Package: cairo-dock
Architecture: any
Depends: ${shlibs:Depends}, ${misc:Depends}
Description: Dock application like dock of MacOS X
 cairo-dock is dock application like dock of MacOS X.
 This reproduces dock of MacOS X by using Xgl/Compiz and Xcompmgr.
```

9. copyright の変更
   Upstream のコピーライト、パッケージのコピーライト、Upstream のソース取得先を記述します。

```
Package Maintainers:
Nobuhiro Iwamatsu <hemamu@t-base.ne.jp> from Thu,  9 Nov 2006 00:19:36 +0900.

Upstream Authors:

Mirco "MacSlow" Mueller <macslow@bangang.de>
Behdad Esfahbod <behdad@behdad.org>
David Reveman <davidr@novell.com>
Karl Lattimer <karl@qdh.org.uk>

Upstream Website:  <http://www.gnome-dock.org/trac>

Copyright:

Mirco "MacSlow" Mueller <macslow@bangang.de>
Behdad Esfahbod <behdad@behdad.org>
David Reveman <davidr@novell.com>
Karl Lattimer <karl@qdh.org.uk>

This program is free software; you can redistribute it and/or
modify it under the terms of the GNU General Public License
as published by the Free Software Foundation; either version 2
of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program; if not, you can either send email to this
program's maintainer or write to: The Free Software Foundation,
Inc.; 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA.

On Debian systems, a copy of the license can be found in /usr/share/common-licenses/GPL
```

10. changelog の修正 Debian 特有の変更点を changelog に記録する必要があります。エディタ等で修正することも可能ですが、dch というフロントエンドが用意されていますので、これを使って changelog を編集します。



11. Upstream のソース変更

    そのままのソースコードでは パッケージ化した場合に不都合がある場合が多々あります。パッケージを作成する前に、Debian のパッケージ作成に合うように修正する必要があります。

    今回修正した点は以下のとおりです。

    - Makefile の install ターゲットがないので追加します。
    - Debian で配布されている libcario パッケージの cairo-glitz が有効になってないので、Makefile の -DHAVE_GLITZ を無効にします。

      変更前

```
APP = cairo-dock

CFLAGS  = `pkg-config --cflags cairo gtk+-2.0 librsvg-2.0 glitz-glx` -DHAVE_GLITZ
LDFLAGS = `pkg-config --libs   cairo gtk+-2.0 librsvg-2.0 glitz-glx` -lm

SRC = cairo-dock.c

all: $(APP)

clean:
        rm -f *.o *~ $(APP)
```

      変更後

```
APP?=cairo-dock
BINDIR?=/usr/bin

CFLAGS  = `pkg-config --cflags cairo gtk+-2.0 librsvg-2.0 glitz-glx`
LDFLAGS = `pkg-config --libs   cairo gtk+-2.0 librsvg-2.0 glitz-glx` -lm

SRC = cairo-dock.c

all: $(APP)

clean:
        rm -f *.o *~ $(APP)

install:
        install -d ${DESTDIR}${BINDIR}
        install -m 755 cairo-dock ${DESTDIR}${BINDIR}
```

- cairo-dock.c の画像ファイルの指定がプログラムのあるディレクトリになっているので修正します。

直接ソース等を修正してもいいのですが、差分で管理するために diff を取り、dpatch で管理します。詳細は Debian 勉強会 2005 年 07 月の資料[*5]にあります。

4.4.1 manpage の作成

実行権限があるファイルに manpage がない場合作成し、パッケージで提供する必要があります。

12. パッケージの作成

    debuild コマンドを使い、パッケージを作成します。

[*5] debianmeetingresume200507.pdf

```
% debuild
 fakeroot debian/rules clean
dh_testdir
dh_testroot
rm -f build-stamp configure-stamp
# Add here commands to clean up after the build process.
/usr/bin/make clean
make[1]: ディレクトリ `/tmp/cairo-dock-0.0.1b' に入ります
rm -f *.o *~ cairo-dock
make[1]: ディレクトリ `/tmp/cairo-dock-0.0.1b' から出ます
dh_clean
 dpkg-source -b cairo-dock-0.0.1b
dpkg-source: building cairo-dock using existing cairo-dock_0.0.1b.orig.tar.gz
dpkg-source: building cairo-dock in cairo-dock_0.0.1b-1.diff.gz
dpkg-source: building cairo-dock in cairo-dock_0.0.1b-1.dsc
 debian/rules build
dh_testdir
# Add here commands to configure the package.
touch configure-stamp
dh_testdir
# Add here commands to compile the package.
/usr/bin/make
make[1]: ディレクトリ `/tmp/cairo-dock-0.0.1b' に入ります
cc `pkg-config --cflags cairo gtk+-2.0 librsvg-2.0 glitz-glx`
        `pkg-config --libs   cairo gtk+-2.0 librsvg-2.0 glitz-glx` -lm  cairo-dock.c   -o cairo-dock
make[1]: ディレクトリ `/tmp/cairo-dock-0.0.1b' から出ます
#docbook-to-man debian/cairo-dock.sgml > cairo-dock.1
touch build-stamp
 fakeroot debian/rules binary
dh_testdir
dh_testroot
dh_clean -k
dh_installdirs
# Add here commands to install the package into debian/cairo-dock.
/usr/bin/make DESTDIR=/tmp/cairo-dock-0.0.1b/debian/cairo-dock install
make[1]: ディレクトリ `/tmp/cairo-dock-0.0.1b' に入ります
install -d /tmp/cairo-dock-0.0.1b/debian/cairo-dock/usr/bin
install -m 755 cairo-dock /tmp/cairo-dock-0.0.1b/debian/cairo-dock/usr/bin
install -d /tmp/cairo-dock-0.0.1b/debian/cairo-dock/usr/share/cairo-dock
install -m 666 *.svg /tmp/cairo-dock-0.0.1b/debian/cairo-dock/usr/share/cairo-dock
make[1]: ディレクトリ `/tmp/cairo-dock-0.0.1b' から出ます
dh_testdir
dh_testroot
dh_installchangelogs
dh_installdocs
dh_installexamples
dh_installman
dh_link
dh_strip
dh_compress
dh_fixperms
dh_installdeb
dh_shlibdeps
dh_gencontrol
dpkg-gencontrol: warning: unknown substitution variable ${misc:Depends}
dh_md5sums
dh_builddeb
dpkg-deb: `../cairo-dock_0.0.1b-1_i386.deb' にパッケージ `cairo-dock' を構築しています。
 dpkg-genchanges
dpkg-genchanges: including full source code in upload
dpkg-buildpackage (debuild emulation): full upload (original source is included)
Now running lintian...
W: cairo-dock: description-synopsis-might-not-be-phrased-properly
W: cairo-dock: wrong-bug-number-in-closes l3:#nnnn
Finished running lintian.
Now signing changes and any dsc files...
 signfile cairo-dock_0.0.1b-1.dsc Nobuhiro Iwamatsu <hemamu@t-base.ne.jp>

次のユーザーの秘密鍵のロックを解除するには
パスフレーズがいります:" Nobuhiro Iwamatsu <hemamu@t-base.ne.jp> "
1024 ビット DSA 鍵, ID 3170EBE9 作成日付は 2003-09-29


 signfile cairo-dock_0.0.1b-1_i386.changes Nobuhiro Iwamatsu <hemamu@t-base.ne.jp>

次のユーザーの秘密鍵のロックを解除するには
パスフレーズがいります:" Nobuhiro Iwamatsu <hemamu@t-base.ne.jp> "
1024 ビット DSA 鍵, ID 3170EBE9 作成日付は 2003-09-29


Successfully signed dsc and changes files
```

4.5 パッケージのテスト

パッケージができたから終わりなのではなく、できたパッケージの動作確認やパッケージ段階で不具合がないか、確認する必要があります。

4.5.1 pbuilder でビルドチェック

pbuilder[*6] は chroot システムを構築し、その中でパッケージのビルドを行うツールです。最低限のユーザーランドの上で、パッケージの依存関係を解決し、パッケージをビルドできるの確認することができます。パッケージの依存関係に不備があった場合はパッケージがビルドできません。パッケージはできたが、実は自分の環境でしかビルドできなかったという単純な問題を無くすために使用したほうがいいでしょう。

4.5.2 実際にインストールして、動作確認を行う

動作確認していないものを不特定多数の人に配布するのは問題ですので（やってない人もおられるようですが。）実際にインストールして、動作確認を行います。

```
# dpkg -i cairo-dock_0.0.1b-1_i386.deb
% cairo-dock
```

4.5.3 アンインストールできるか確認する

インストールした際にアンインストールスクリプトに不具合があり、正常にアンインストールできない場合があります。アンインストールもできるか確認します。

4.6 まとめ

Debian パッケージを作成することは難しくなく、容易に作成できる環境は整っています。

肝心なのはツールを使えるようになることではなく、Debian Policy をどれだけ熟読しているか、にかかってくるように思います。

作って分からないことがあれば debian-devel@jp[*7]等で聞くといいでしょう。

[*6] http://packages.qa.debian.org/p/pbuilder.html

[*7] debian-devel@debian.or.jp

5　sid を日常環境として使うための注意

上川

5.1　sid とはなにものか

Debian sid は別名 unstable で、毎日リリースされている Debian の開発版です。開発者が新しいパッケージをリリースしたらまずそこに入ります。毎日日本時間午前 4 時くらいに処理されており、そのタイミングで新しいバージョンが配布されます。

開発者に近い層のユーザは、最新版のパッケージを利用するために unstable を利用します。問題があれば随時バグ報告をしていきます。また、apt-listbugs などを利用し、他のユーザから報告された深刻なバグがないか確認しながら作業します。

5.2　インストール方法

毎日最新版になるので、安定して直接インストールできる方法というのは基本的には無いでしょう。安定版をインストールしてから、 sid にアップグレードするという形が通常のやりかたです。

また、別の方法として、chroot 内部に sid を飼うという方法があります。debootstrap や、 cdebootstrap　などのツールを利用すると、 chroot 内部で Debian sid が稼働している状況をつくれます。pbuilder などのツールを利用するとより便利に利用できます。

sid は最新版を常においかけているので、深刻なバグのリスクに出会う危険性が常にあります。 chroot 内部で常に最新版を確認して、それを外部に展開するというやりかたをしないと危険でしょう。

5.3　魅力

常に最新版が利用できます。開発が起きている場です。開発をしたいだとか、オープンソースの世界の動きを肌で感じたいというのであれば、お薦めです。

5.4　情報源

IRC や ML や　勉強会で情報収集しましょう。`#debian-devel` IRC チャンネルのトピックが一番最新の情報が得られます。

apt-listbugs や apt-listchanges というツールも利用しましょう。apt-listbugs　は今インストールしようとしているパッケージのバージョンに該当する深刻なバグレポートを表示してくれるツールです。apt-listchanges　は前回インストールしたバージョンからの changelog の差分を表示してくれるツールです。

6 bugreport 論

上川

6.1 Debian BTS の特徴

Debian BTS は、ウェブとメールフロントエンドをもつバグトラッキングシステムです。他のバグトラッキングシステムと違う点として、操作が全てメールでしかできないという点と、情報が全て完全に公開されるという点があります。また、バグレポートをパッケージ単位で分類しているという点も特徴です。

6.2 使われ方

深刻なバグ（RC バグ）を登録すると、そのパッケージのバージョンはリリースできない、という意味になります。各ユーザは apt-listbugs を経由してそのようなバグを確認し、深刻なバグのあるパッケージのバージョンをインストールしないように回避できます。また、この情報はリリースマネージメントにつかわれています。

6.3 アーキテクチャ

バックエンドデータベースはプレーンのテキストファイルです。ファイル構造は下記のようになっています。

- /org/bugs.debian.org/spool
  - incoming/
  - db-h/
    - 00/
      - ..
      - 314200.log
      - 314200.report
      - 314200.status
      - 314200.summary
    - ..
    - 99/
  - archive/
    - 00/
    - ..
    - 99/
  - index.db – index.db.realtime へのシンボリックリンク
  - index.archive – index.archive.realtime へのシンボリックリンク
  - nextnumber

メールを受信したら、そのメールが incoming 以下にスプールされます。そのメールを 15 分に一回 cron から起動されるスクリプトが処理します。

ウェブ経由でバグ報告を確認するための cgi があります。その cgi 経由では BTS は内容の閲覧だけができ、変更はしません。

また、スクリプトなどから利用するために、SOAP のフロントエンドがあります。ドキュメントは一行もありません。

現状 `http://bugs.donarmstrong.com/cgi-bin/soap.cgi` でステージングされています。今後は `http://bugs.debian.org` に移行したいようです。ネームスペースは `Debbugs/SOAP/Status` で、そこで `get_status` という関数が定義されています。`get_status` は複数の数字パラメータをうけ、そのあたえられた数字に対応するバグレポートの情報をかえします。`get_status` は、バグ報告の本文ではなく、メタデータを返します。

6.4 文化

たまに BSP などの祭があります。バグトラッキングシステムに登録されているバグで最初に対応されないものについては、そのまま放置され時間だけが過ぎてしまう傾向があります。それらに対して新たな気持ちで対応しようというものです。

6.5 参考文献

- Debian 勉強会 2005 年 10 月資料 「debbugs internal」
- Debian 勉強会 2005 年 10 月資料 「apt-listbugs の生い立ちと実装」
- `/usr/share/doc/debian/bug-maint-mailcontrol.txt` など

7 次回

未定です。内容は本日決定予定です。
参加者募集はまた後程。

Debian 勉強会資料

2006 年 11 月 19 日　　初版第 1 刷発行
東京エリア Debian 勉強会 （編集・印刷・発行）