第 15 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 4 月 15 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 で文書はこうやってつくっています」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 小室さん

基本的に議事録とかは後々メールに貼付けたり添付したりするので、bluebird でテキスト作成して、他のユーザーを巻き込むような案件での文章は Open office を使って相手が microsoft office を使って開けるように気をつけています。相手が powerpoint で開くとかならずレイアウトが崩れている！と嫌みめいたメールが来ますがそこはあんまり気にしないようにしてます。後はよくする事が thunderbird でメール新規作成をしてドラフトで保存してます。検索が簡単なのとメールでのやり取りが多い＆新しく bluebird を立ち上げなくていいので、結構便利だと思っています。

1.2.2 澤田さん

dpkg とどう接しているかと言われると、

- パッケージを入れた後にどんなファイルが含まれてるか見るために-L
- パッケージを入れてるかを確認するために-l
- ファイルがどのパッケージのものかを確認するために-S

ですね。-L でファイル一覧表示→気になったファイルを lvってのをよくやるので、それを統合したアプリなんてあるとうれしいのかもしれない。

1.2.3 三島さん

Debian 上で書く文書は、プログラムとそれに付随する文書か、あるいは日記のようなものが多いです。

プログラムの場合、シェルスクリプト等の短いものならば vi (jvim) を使いますが、長くなるようなら emacs を使います。インデント等は emacs に頼りきりです。どちらの場合もテキスト端末用のものしか使いません。

一方、日記や、思いつきのメモを書くためには hiki と Web ブラウザを使っています。どこからでも更新・参照でき、最低限の文書構造を表現できるので Wiki を使っています。難点は、hiki の編集モードが W-ZERO3 + Opera 環境からだととても使いづらいところです。

1.2.4 矢吹さん

Debian の上で生活をしているので、Debian で文書を作成するのは、特別に意識することなく作成しています。基本的には日本語で文書を作成し、英語で文書を作成することもあります。日本語の文書を作成する上で、重要なファクターとして「日本語入力」があります。現在ケースバイケースで複数の日本語入力をおこなっています。現在 X の上では、SCIM をつかっており、Emacs の上では yc-el を使っています。入力方法は、ローマ字ならびに Nicola 入力で入力することが多いです。文書は、mail が一番おおく、つぎは tDiary への publishing です。その次は発表資料作成に、OpenOffice.org の Impress でプレゼン資料を作ります。長文の論文などは、pLaTeX を使います。最終出力形態は、プラットホームを意識する場合が少ない pdf にすることが多いです。英語の文書を作成するときには、辞書とスペルチェッカーが重要です。ebview や lookup.el、ならびに flyspell-mode はよく使います。

1.2.5 吉田さん

Debian で文書を作らなければならないときは、通常、下記いずれかの方法を使用しています。

前提 Cygwin の ssh を cocot を使用して起動し、Sarge 環境に接続。

- 英語の場合
  Windows でコピーして、vi で作成したファイルに i 押して貼り付け。または `cat >`
- 日本語の場合
  jvim-canna-nox(x に依存しない自作野良パッケージ) を使用して canna で入力
- 面倒な場合
  scp

1.3 中島さん

文書は、ほとんど WEB ブラウザを使って作っていて、テキストの表示も紙に出力しないでブラウザだけで表示している。ブログを読んでコメントを書いたり全てブラウザを使う。ワープロや表計算ソフトみたいなのも使わない。簡単な文章やメモ書き程度だったら紙で書いている。文字数で言うと４００字以内だったら紙を使う。あと漢字変換をやってしまうと漢字を忘れるので紙を使うようにしている。それとマインドマップを上手になりたいので意識的に絵を書くようにしている。紙と色ペンを使って紙で書いてそれをブラウザに転記するのがいまのところ一番よい。

1.4 岩松

いままで Linux でドキュメントを書くことはありませんでした。プレゼンテーションのときは ooimpress でした。しかし、Debian 勉強会に参加して、自分でも発表を行うようになってから、Tex に目覚め、今では会社のドキュメントも Tex で書く様になってしまいました。これからは Word なんかを使わず、茨の道を進んでいこうと思います。

1.5 やまねさん

Debian で文章編集…どのような環境、といっても

- IME - uim+anthy
- mail - Sylpheed / firefox(gmail)
- irc - loqui
- editor - gedit / vim
- presentation - OpenOffice.org(Impress)

と面白みの無い答えになります。Emacs など使い方が身につけばつかってみたいのですが、なかなかきっかけがありません。(riece を起動するだけにしか使わない)

1.6 小林さん

Debian では主に Emacs 21 で文書を書いています。フォーマットは、個人的なメモなど、印刷物などにする必要がないものに関してはすべて RD です。

学生なので TEX もそれなりに使ってきました。レジュメなど最終的に印刷する必要があるものはすべて TEX です。もちろん修論も同様で、スタイルファイルには okumura-clsfiles パッケージの jsbook.cls を使い、出力は dvipdfmx で pdf に変換しました。Emacs のメジャーモードには YaTEX を使っています。

個人的なものについては、印刷物は TEX、それ以外は RD ですが、翻訳作業をしている関係でそれ以外のフォーマットも扱います。しかし最近では、ウェブページはもちろん、様々なソフトウェアのドキュメントもみな XML/SGML やその仲間 (HTML, XHTML, DocBook/XML, wml, ...) で書かれているので、特筆することはありません。せいぜい、DocBook/XML で書かれた Aptitude のドキュメントを docbook-xsl パッケージの XSLT スタイルシートで変換しようとして文字コードやその他のバグにちょっと苦しんだくらいです。docbook-xsl の新しいバージョンのパッケージはいつ入るのかが気になる昨今です (というか sarge リリース直前に入ったのも NMU だし、ほとんどメンテナンスされていないような気がします)。

1.7 北原さん

率直に正直に言いますと「Debian で文書はつくっていません」。 文章を作るときには、手馴れた某 OS の某エディタを使用し、必要があれば、転送・文字コード変換等を行います。 装飾が必要なときは、作成したテキストをワードプロセッサに読み込ませます。 これでは、デビアン勉強会の宿題回答にはならないので、あえてデビアン環境で文章を作るならという事で回答すると、短いメモや覚書程度なら vi 、少々サイズが大きい文章なら OpenOffice.org でしょうか。 TEX とかは使用できません。(何とか 200 文字を越えました。)

1.7.1 上川

- 簡単な文章
  簡単な文書は，最近は outline-mode で書いています．以前は tex であらゆるメモを書いていましたが，最近はフォーマットをあまり考えなくなりました.
- 英語での複雑な文章
  英語の文書は DocBook を使う事が多いです．emacs で wysidocbookxml を利用して，リアルタイムプリビューしながらドキュメントを書いています.
- 日本語での複雑な文章
  whizzytex を利用してリアルタイムプリビューをしながら，tex で書いています.

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 8 号

`http://www.debian.org/News/weekly/2006/08/` にある 2 月 22 日版です。

問題 1. Debian etch beta1 インストール用メディアにどういう問題があったか

□ A 最新じゃないのでつかってられない
□ B メディアが水に濡れて使えなくなった
□ C Debian アーカイブの変更の影響で動かなくなった

問題 2. Debian Live Initiative は何をしようとするものか

□ A 新しい開発をがんがんする
□ B Debian の Live CD を統合する
□ C リアルタイムハック実況中継のための環境を提供する

2.2 2006 年 9 号

`http://www.debian.org/News/weekly/2006/09/` にある 2 月 28 日版です。

問題 3. ミラーシステムについて Anthony Towns が発表したのは何か

□ A i386 と amd64 だけに限定して今後は運用する
□ B 全アーキテクチャを含めた巨大なミラーを継続
□ C アーキテクチャ毎にミラーを分割する

問題 4. NMU を実施する際に，注意するべきことは何か

□ A BTS を通してメンテナに通知すること
□ B NMU なんてしてる暇があったら自分のバグを直す
□ C できるだけメンテナにばれないように実施する

2.3 2006 年 10 号

`http://www.debian.org/News/weekly/2006/10/` にある 3 月 7 日版です。

問題 5. AMD64/kFreeBSD について何がおきたか

□ A はじめてパッケージが動いた
□ B glibc/gcc/binutils がポーティングできた
□ C chroot 内部で動作するようになり，buildd が動いている

問題 6. バックポートのサポートが公式になるのか，という質問についての回答は
□ A Utunubu 広報担当によると Debian はもう時代遅れだ
□ B Joseph Smidt によると，Debian はバックポートを主体として今後は活動を続ける
□ C Norbert Tretkowski によると，公式なサポートつきのバックポートを提供することは考えにくい

2.4 2006 年 11 号

http://www.debian.org/News/weekly/2006/11/ にある 3 月 14 日版です。
問題 7. Bastian Blank が発表した，Debian カーネルチームの作業内容は
□ A kernel-image-という名前から linux-image-という名前に変更しました
□ B カーネルは FreeBSD のものに入れ換えました
□ C 今後は SMP 版と Uniprocessor 版というだけでなく，何 CPU の SMP かということで flavor を分けます

問題 8. Martin の後の安定版リリースマネージャは誰にならなかったか
□ A Martin Zobel-Helas
□ B Andreas Barth
□ C Nobuhiro Iwamatsu

2.5 2006 年 12 号

http://www.debian.org/News/weekly/2006/12/ にある 3 月 21 日版です。
問題 9. JBoss4 の Debian パッケージは存在するか
□ A Guido Guenther が作成したものが存在する
□ B non-free なのでそんなものは存在しない
□ C ボスって何？

問題 10. パッケージに含めるドキュメントの形式は PDF だけにしたい，というメールに対しての反応は
□ A HTML のほうが grep しやすいので，HTML をいれてほしい
□ B DVI のほうがファイル構造が安定しているので DVI にしてほしい
□ C プレインテキスト形式のほうが小さいのでプレインテキスト形式にしてほしい

2.6 2006 年 13 号

http://www.debian.org/News/weekly/2006/13/ にある 3 月 28 日版です。
問題 11. David Moreno Garza が作成したのは
□ A DWN の mixi 風インタフェース
□ B DWN の RSS フィード
□ C DWN の 2ch 風インタフェース

問題 12. google groups を利用して Debian バグを検索するにはどのニュースグループをみればよいか

□ A there.is.no.bugs ニュースグループ

□ B bugs.debian.org ニュースグループ

□ C linux.debian.bugs.dist ニュースグループ

2.7 2006 年 14 号

http://www.debian.org/News/weekly/2006/14/ にある 4 月 4 日版です。

問題 13. ndiswrapper が main に入っているのがただしいのかという議論の原因は何か

□ A ndiswrapper のソースコードは実は全部暗号文で構成されているから

□ B ndiswrapper は思想的におかしいから

□ C ndiswrapper はエミュレーションする対象のドライバが無いと実用できないから

問題 14. Debian Project Leader の投票について Clytie が苦情をいったのは何故か

□ A Debian Developer でないと投票権がないことに気づかずに投票した

□ B 立候補者がどれもえらびようがないような人達ばっかりだった

□ C Branden Robinson が立候補していなかった

2.8 2006 年 15 号

http://www.debian.org/News/weekly/2006/15/ にある 4 月 11 日版です。

問題 15. Xen の Debian パッケージはもともと Julien Danjou たちと Blastian Blank が別個に作業していた．それが統合されたのはいつか

□ A 4/5

□ B 3/1

□ C 1/1

問題 16. sudo のセキュリティーフィックスでどういう変更がなされたか

□ A アプリケーションを実際に実行するのではなく，実行しているっぽくみせかけるだけになった

□ B ルート権限でプログラムを実行しなくなる

□ C 実行プログラムに引き渡される環境変数を制限

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 14 回目報告

Debian 勉強会は Open Source Conference に出展しました．そこで，sid へのいざないについてやまねさんが，Debian 勉強会の紹介を岩松さんがしました．30 名ほど参加しました．

質疑応答もありました．

Q	A
勉強会は一方通行なものなのかインタラクティブなものなのか	インタラクティブです
インストール大会はしないのですか？	やろうと思ってる人が動けてないという現状です
インストールが難しいと思ってる人〜?	いなかった。難しいと思ってる人がいないのが問題では？
reportbug の国際化はしないのですが？	途中やりらしいです（python で書かれてて国際化のフレームワークはあるみたいです)。岩松さんがやる？ なおレポート本体は英語で書かないといけません
apache の stable はバージョンが古いので一部だけ unstable にしたいという場合は？	一部だけ unstable ははまる可能性が高いので backports.org を使ったほうがよいです。メーリングリストに投げると 誰かやってくれるかもしれません　unstable からパッケージを持ってきてビルドし直して独自のリ ポジトリを作るという手もあります
opera とかのフリーじゃないものを使うのは邪道ですか？	いいえ
勉強会に参加するのにどれだけの技術力が必要ですか？	むしろそれって何というつっこみを入れてくれる方歓迎。ただし、やってることを勘違いして来られるのは困ります

4 Debian policy

岩松

Debian Policy 第 3 回です。今回は Source package についてです。

4.1 ソースパッケージとは？

ソースパッケージは Debian が配布しているバイナリパッケージの元になっているパッケージのことです。例えば、シェルスクリプトの **bash**[*1] は bash[*2]というソースパッケージからビルドされます。しかし、bash ソースパッケージは bash バイナリパッケージを作成するだけでなく、bash-builtins[*3]パッケージや bash-doc[*4]パッケージもビルドされます。一つのソースパッケージから複数のソースパッケージがビルドされるとがあるということです。

4.2 Standards-Version について

Standards-Version は Debian Policy のバージョンを指します。Debian Policy は常に更新されており、現在、バージョンは 3.6.2.2 です。ソースパッケージは常に最新の Debian-Policy に追従すべきであると書かれています。実際にはパッケージをアップデートしたときに、Debian Policy のバージョンをチェックし上がっていた時、Standards-Version の追従してバージョンを上げます。
Standards-Version は debian/control ファイルの Standards-Version フィールドに記述します。Standards-Version フィールドのフォーマットもポリシーで決められており、セクション 5.6.11 で説明されています。

4.3 パッケージ関係について

パッケージをビルドする際に必要なパッケージが出てきます。そのビルドに必要なパッケージを指定する必要があると書かれています。

必要なパッケージを全て書くわけではなく、最低限必要なパッケージを書くべきであると書かれており、例えば、bash を例にすると、ビルドの依存関係は以下のようになっています。

```
Build-Depends: autoconf, patch, bison, libncurses5-dev, texinfo, autotools-dev, debhelper (>= 4.1),
texi2html, locales
```

libncurses5-dev に注目して、libncurses5-dev[*5]の依存関係を見てみると、

```
Build-Depends: debhelper (>= 3.0.23), libc6-dev-sparc64 [sparc], libc6-dev-s390x [s390],
libc6-dev-amd64 [i386], libc6-dev-ppc64 [powerpc], lib64gcc1 [i386 powerpc sparc s390],
libgpmg1-dev (>= 1.19.6-20) [!hurd-i386 !kfreebsd-i386], quilt (>= 0.40-1)
```

となっています。依存しているパッケージに依存しているパッケージはもともと依存しているので、書く必要がないということです。

パッケージ間の依存の詳細に関しては セクション 7 で説明されています。

[*1] http://packages.debian.org/unstable/shells/bash
[*2] http://packages.qa.debian.org/b/bash.html
[*3] http://packages.debian.org/unstable/utils/bash-builtins
[*4] http://packages.debian.org/unstable/doc/bash-doc
[*5] ソースパッケージは ncurses

4.4 アップストリームのソース変更について

Debian では Debian social contract に書かれているように、Debian で発生した不具合やパッチをアップストリームに還元するようにしています。アップストリームとは上流開発者のことで、パッケージの開発元を指します。Debian 特有の問題やビルド時における最適化等で修正を入れるときがあります。ビルド前のテストで Debian として追加したい項目があるときは autoconf を使って適切に処理したり、Makefile を修正するときは、Makefile を直接修正せずに、Makefile.in を修正するようにとも書かれています。これは configure を行ったときに Makefile が上書きされてしまうからです。

4.5 Debian changelog について

Debian changelog とは Debian パッケージに関する変更点について書かれたものです。アップストリームの変更とは別書く必要があり、debian/changelog ファイルに記述します。

ポリシーとして、debian/changelog に Debian パッケージによる変更点を簡潔に記述すべきであると書かれています。debian/changelog を修正するときは dch[*6]を使うと便利です。

Debian changelog の役目はこれだけではなく、debian/changelog からパッケージのバージョン情報を取得し、パッケージ構築の際に使用します。形式は以下のようになります。

```
package (version) distribution(s); urgency=urgency
     [optional blank line(s), stripped]
  * change details
    more change details
     [blank line(s), included in output of dpkg-parsechangelog]
  * even more change details
     [optional blank line(s), stripped]
 -- maintainer name <email address>[two spaces]  date
```

- package , version
  ソースパッケージ名とソースパッケージのバージョンを指します。
- distribution
  version で指定されたパッケージがインストールされるディストリビューションを指します。Distribution に関しては Section 5.6.14. で説明されています。
- urgency
  パッケージをアップロードする際の緊急度を指定します。low, medium, high ,emergency を指定することができます。
- コメント部
  コメントに関しては先頭は 2 つのスペースが必要です。習慣で各変更内容の先頭はアスタリスクになっています。長い文章は改行するのですが、改行したときは字下げを行います。字下げは上のテキストに沿って行います。空改行は変更内容をわけるために使用したりします。
  変更内容に不具合の修正内容を書くときがあります。このとき、BTS[*7]に登録されている場合があります。バグの番号をフォーマット通りに changelog に書くことによって、changes ファイルに書き込まれ、パッケージがアップロードされたときに、自動的にバグが close されます。フォーマットは #nnnnnn です。
- maintainer name , email address changelog を書く際にメンテナ名とメールアドレスを記述します。この項

[*6] http://packages.debian.org/unstable/devel/devscripts
[*7] http://bugs.debian.org

目はパッケージがアップロードされた時の承認結果を送る際に使用されます。また、パッケージのキーサインにもこの項目が使われます。

- date 修正した日時を書きます。RFC822 フォーマットに基づいて書く必要があります。
- タイトルタイトル部は左から始まります。メンテナーの前はスペースを入れ、トレーサー（–）を入れる必要があります。また、メンテナと日付の間には２つスペースを入れ、分ける必要があります。

changelog がインストールされる場所はセクション 12.7 に説明されています。

また、代替の Changelog フォーマットを使うことができます。実験用ではないパッケージでは、dpkg の最新バージョンでサポートされる debian/changelog のためのフォーマットを使用しなければなりません。自分が使用したいフォーマットがあるなら、パーサーを提供することによって変更することができます。パーサーは dpkg-genchanges および dpkg-gencontrol によって期待された API 互換性を持つ必要があります。

4.6 Error trapping in makefiles

Makefile からシェルスクリプトが呼ばれるときがあります。例えば、dpatch [*8]によって呼ばれる patch ファイルです。Makefile 内でシェルスクリプトファイルがエラーが発生しても、エラーを捉えることができません。そのため、シェルスクリプトファイルは実行の際に -e オプション[*9]を付けなければならないと説明されています。

4.7 タイムスタンプについて

可能な限りアップストリームのソースファイルのタイムスタンプをパッケージ中に変更せず、そのままにしておくことを推奨すると説明されています。

4.8 ソースパッケージの中のオブジェクトファイルにおける制限

ソースパッケージの中にはハードリンク、デバイスファイル、ソケット、setuid や getuid されたファイルを入れてはいけません。

4.9 Main building script: debian/rules

deban/rules ファイルは ソースパッケージからバイナリパッケージを作成する方法がスクリプトです。実態は実行可能な (パーミッション:755)makefile です。ファイルの先頭は**#!/usr/bin/make -f** になっています。

スクリプトは非対話式になっています。対話式だと、毎回同じバイナリが生成されるとは限らないので、自動的にバイナリパッケージが生成されるようになっています。スクリプトの内容は dpkg-buildpackage から呼ばれる必要なターゲットとして clean, binary, binary-arch, binary-indep, build があり、これらが最小の構成になっています。

- build
  パッケージの設定、コンパイルを行います。もし、パッケージ構築前に設定作業がある場合は、Debian 化されたソースの設定作業を行った後で行うべきであると書かれています。その理由として設定を再度行わず、パッケージの構築が行えるようにするためです。
  いくつかのパッケージは同じソースパッケージからコンパイルのやり方を変更して異なったバイナリを生成する場合があります。build ターゲットではこのような処理には対応できないので、それぞれの構築方法に従って、それぞれのターゲット（例えば、binary-a と binary-b）を作成して使用するといいと書かれています。この場合は実際は build ターゲットではなにも行わず、binary ターゲットでそれぞれのパッケージをビルドしてそれぞれのバイナリパッケージを作成することになります。

[*8] http://packages.debian.org/unstable/devel/dpatch
[*9] ERR トラップが設定されていればそれを実行して終了します。

ルート権限が必要な作業は行ってはいけません。

- build-arch (optional), build-indep (optional)
  build-arch は、提供された場合、アーキテクチャーに依存しているバイナリパッケージ (debian/control ファイルの Architecture フィールドが”all” ではないとき）すべて生成するために必要になった設定やコンパイルをすべて行なうべきです。build-indep は アーキテクチャーから独立しているバイナリパッケージ (debian/control ファイルの Architecture フィールドが”all” のとき）すべて生成するために必要になった設定やコンパイルをすべて行なうべきです。build ターゲットは、rules ファイルの中で提供される build-arch および build-indep に依存するべきです。
- binary, binary-arch, binary-indep
  binary ターゲットはこれだけで、バイナリパッケージを構築できないといけません。binary ターゲットは 2 種類に分けられ、binary-arch は特定のアーキテクチャ用のファイル、binary-indep はそれ以外のファイルを生成します。これらのターゲットは非対話的に動作するものでなければいけません。
- clean
  build ターゲットと binary ターゲットによって生成されたファイルを削除し、元に戻します。例外として、binary ターゲットで出力されたファイルは消さず、残します。このターゲットは非対話的である必要があります。
- get-orig-source (optional)
  このターゲットは主要なアーカイブサイト（例えば、リングサーバー?）から最新のオリジナルソースを HTTP や FTP から取得します。取得したオリジナルソースを tar ファイルに再構成します。

build , binary および clean ターゲットはパッケージのトップディレクトリをカレントディレクトリとして実行されなければなりません。

公開されている、またはいないインターフェイスのためやパッケージ内部で使用するために debian/rules に他のターゲットを置くことは許されます。

パッケージを実際に構築するマシンやインストールの対象となるマシンのアーキテクチャは、dpkg-architecture を使い、変数を指定することによって決定されます。これにより、ホストマシンだけでなくパッケーの構築するマシンの Debian 形式のアーキテクチャーと GNU 形式のアーキテクチャ指定文字列を取得するとができます。

- DEB_BUILD_ARCH
  Debian 形式のパッケージ構築マシンアーキテクチャ
  例：i386
- DEB_HOST_ARCH
  Debian 形式のインストール先アーキテクチャ
  例：i386
- DEB_BUILD_GNU_TYPE
  GNU 形式のパッケージ構築マシンアーキテクチャ指定文字列
  例：i486-linux-gnu
- DEB_HOST_GNU_TYPE
  GNU 形式のインストール先アーキテクチャ指定文字列
  例：i486-linux-gnu
- DEB_BUILD_GNU_CPU
  DEB_BUILD_GNU_TYPE の CPU 部分
  例：i486
- DEB_HOST_GNU_CPU
  DEB_HOST_GNU_TYPE の CPU 部分
  例：i486

- DEB_BUILD_GNU_SYSTEM
  DEB_BUILD_GNU_TPE のシステム部分
  例：linux-gnu
- DEB_HOST_GNU_SYSYTEM
  DEB_HOST_GNU_TYPE のシステム部
  例：linux-gnu

DEB_BUILD_ARCH および DEB_HOST_ARCH は Debian アーキテクチャのみを決定することができます。実際の CPU やシステム情報を取得する際はこれらを使用してはいけません。この場合には GNU 形式の変数を使用しなくてはいけません。

4.10 Variable substitutions: debian/substvars

substvars ファイルはそのパッケージの実行ファイルに関する共有ライブラリの依存関係を計算し、書き出されたものです。bash を例に取ると、内容は以下のようになっています。

```
shlibs:Pre-Depends=libc6 (>= 2.3.5-1), libncurses5 (>= 5.4-5)
```

このファイルは debian/rules によって生成され、動的に変更されます。clean ターゲットで削除されるようにしておく必要があります。実際には dpkg-gencontrol , dpkg-genchanges, dpkg-source が control ファイルを生成するときに substbar を参照してファイルを生成します。substbars を使ったソースの変換方法については、dpkg-source の man に書かれています。

4.11 Generated files list: debian/files

このファイルはソースツリーの常に存在する部分ではありません。これはどのようなパッケージが生成されたのか記録するために用いられます。dpkg-genchanges は、.change ファイルを生成する際に使用します。bash を例に取ると、以下のような内容になっています。

```
bash-doc_3.1-4_all.deb doc optional
bash_3.1-4_i386.deb shells required
bash-builtins_3.1-4_i386.deb utils optional
bash-static_3.1-4_i386.deb shells optional
bash-minimal_3.1-4_i386.deb shells optional
```

また、このファイルはアップロードされるソースパッケージには含めてはならず、debian/rules の clean ターゲットで削除すべきであると書かれています。

5 Debian TEX のファイル構造

上川

TEXpolicy について簡単に解説します.

5.1 文書

この文書は tex-common パッケージに入っている Debian-TEX-Policy ファイル (`file:///usr/share/doc/tex-common/Debian-TeX-Policy.pdf.gz`) を概訳したものです.

5.2 用語

用語が定義してあります.

5.3 ファイル配置

TEX の入力ファイルのみを TEXMF ツリーに配置します. そうでないものは, /usr/share/PACKAGE に配置します. 例外として, 説明のためのテキストファイルは TEXMF ツリーに配置することができます.

5.3.1 パス検索と libkpathsea/libkpse

ファイルフォーマットなどに基づいて, TEXMF ツリーの検索をするためのライブラリです. libkpathsea はあまり考えていませんでしたが, libkpse は, API/ABI を考慮したライブラリです.

スクリプトからは kpsewhich, kpsepath, kpsexpand, kpsestat を利用できます.

5.3.2 ディレクトリツリー

配置は, TEX ディレクトリ構造標準 (TDS) に準拠する. TDS の古いバージョンに準拠するのはバグ. TDS の新しいバージョンに依存しながらこの tex-common パッケージや TEX の基本パッケージの十分新しいバージョンに依存していないのもバグです.

- /usr/share/texmf-tetex/: TEXMFDIST
- /usr/share/texmf-texlive/: TEXMFDIST
- /usr/share/texmf/: TEXMFMAIN
- /var/lib/texmf/: TEXMFSYSVAR
- /etc/texmf/: TEXMFSYSCONFIG
- /usr/share/texmf-site/: TEXMFSITE
- /usr/local/share/texmf/: TEXMFLOCAL
- texmf.cnf に指定してある TEXMFHOME の値, もしくは環境変数としての値.
- 必須ではない : 各ユーザ用の設定ファイルディレクトリ TEXMFCONFIG, 生成されたファイルのディレクトリ TEXMFVAR

検索は下から優先します.

TEXMFMAIN と TEXMFDIST の Debian での使い方は upstream と違います. TEXMFMAIN がバイナリと一致する必要のあるものを配置していあり, TEXMFDIST はあたらしい TEXMF が配布されて上書きされるもの

を配置していますが，そのような配置はパッケージマネージメントシステムがきちんと動いている環境では必要ありません．

Debian では，TEXMFDIST を basic TEXpackages 用にしており[*10]，TEXMFMAIN は新しいバージョンを提供したりするアドオンパッケージ用にしています．

パッケージはメンテナスクリプトの中で TEXMFHOME を無視するように考慮すべきです．

5.3.3 生成されるファイル

フォントは /var/cache/fonts におきます．その他は /var/lib/texmf 以下，もしくはユーザの指定したディレクトリの下に TDS に準拠したパスに配置します．

/etc/texmf/texmf.cnf は例外です．管理者が編集することは意図していませんが，編集されていた場合には自動生成ツールはその変更を尊重してください．Debian パッケージはそのファイルは変更してはいけません．

5.3.4 ファイル名とファイルの別バージョンのインストール

TEXMF ツリーにすでにあるファイル名と同じ名前ではファイルはインストールしてはいけません．ただし別のアプリケーションからしか見えないサブディレクトリにしかない場合には同じ名前でもかまいません．そういうディレクトリは kpsewhich の –progname や –format にて得られます．

- Basic TEXpackages はそれぞれ同じようなファイルを自分の TEXMFDIST にインストールします．
- より新しいバージョンのファイルが必要な場合，TEXMFDIST にすでにあるファイルを自分のバージョンでおきかえることができ，TEXMFMAIN に配置します．
  ただ，この場合，basic tex packages のメンテナに連絡[*11]してください．
  新しいファイルが後方互換であるように注意してください．
  混乱をまねくため，二種類を越える種類のバージョンが共存することや，dpkg-divert の利用は推奨しません．

5.3.5 ドキュメント

パッケージはドキュメントを texdoc に提供するべきです．/usr/share/doc/texmf 以下のサブディレクトリにファイルをインストールするか，適切なシンボリックリンクを提供することで実現できます．

/usr/share/texmf/doc にはファイルをインストールしないでください．ここは/usr/share/doc/texmf へのシンボリックリンクです．

ドキュメントの代表的なドキュメントはパッケージ名に関連した名前にして，manual.pdf や index.html という名前にしないでください．[*12]

5.4 設定

5.4.1 設定ファイル

TEX において，あらゆる TEX の入力ファイルは設定ファイルになりえますが，設定ファイルが多くなりすぎるのを防ぐため， conffile や configuration file としてファイルをインストールしないでください．/etc/texmf/tex には空のディレクトリを作成し，ユーザにどういうファイルを作成するべきかを指定してください．

/etc/texmf/ は TDS のツリーであり，任意の設定ができます．Debian の tex-common が提供する /etc/texmf/texmf.d/ などは検索対象ではないため，TEX の入力ファイルではないものの置場として利用できます．

[*10] 一部例外あり

[*11] wishlist バグ

[*12] texdoc packagename と指定できるようにするため，これができないとユーザは texdoc packagename/user.dvi のようなコマンドラインを texdoc -s keyword で検索する必要がでてきます．

5.4.2 設定更新プログラム

/etc/texmf/texmf.cnf が中心の設定です．/var/lib/texmf/web2c/updmap.cfg がフォントの設定ファイルです．/var/lib/texmf/tex/generic/config/language.dat がハイフネーションと言語の設定で，/var/lib/texmf/web2c/fmtutil.cnf にてフォーマットの生成が管理されています．

/etc/texmf 以下のツリーからこの四つのファイルは生成されています．

updmap.cfg language.dat fmtutil.cnf についてはこれが唯一の設定方法です．

texmf.cnf は管理者によって編集可能で，変更は ucf で管理されます．

パッケージのメンテナスクリプトは直接編集せず update-texmf を利用します．管理者も update-texmf を利用することを推奨します．

パッケージは設定項目を追加してよいですが，他のパッケージの設定を上書きしようとはしないでください．共有する設定項目は basic TEXpackages でどのパッケージでも利用できるような値を提供させてください．もしデフォルトが不可能であれば，そのことを関係パッケージのメンテナの間で合意をとってください．

メンテナスクリプトは， update-updmap を –quiet オプションをつけて呼び出して下さい．それ以外については，設定アップデートプログラムには特にオプションをつけずに呼び出して下さい．ディレクトリ構造の内部的な変更ができるようにするためです．

updmap.cfg を変更するパッケージは updmap-sys を呼び出す必要があります．language.dat か fmtutil.cnf を変更するプログラムは fmtutil-sys を呼び出す必要があります．その前に mktexlsr を呼び出すのを忘れないで下さい．

5.4.3 フォント設定

PS type1 フォントを提供するパッケージはどんな Basic TEXpackage でも利用できるようにしてください．tex-common で提供されている dh_installtex を利用してください．詳細はマニュアルページ dh_installtex(1) 参照．

- tex-common に depend して，Basic TEXpackage には依存しない
- .map ファイルは TEXMFMAIN/fonts/map にインストールする．
- その他の必要そうなものもインストールする .pfb, .tfm, .enc, .fd, .sty, 文書等
- /etc/texmf/updmap.d/ に 10*name*.cfg という名前法則でファイルを配置する．update-updmap が /var/lib/texmf/web2c/updmap.cfg を生成するのに利用される．`# -_- DebPkgProvidedMaps -_-`を含むファイル形式で詳細は update-updmap(1) 参照．
- /var/lib/tex-common/fontmap-cfg/*package*.list に一行一つさきほどの.cfg を記述．パッケージのファイルとしてインストールすること．
- package.postinst と，postrm(remove/disappear を指定されたばあい) にて， update-updmap –quiet, mktexlsr, updmap-sys をその順番に実行すること．
  (以下 dh_installtex を利用すればよいので詳細な項目略)

5.4.4 言語/ハイフネーション設定

dh_installtex を利用すればよいです．

必要なファイルを TEXMFMAIN に配置し，.cnf ファイルを /etc/texmf/language.d にインストール， update-language を呼び出すと /var/lib/texmf/tex/generic/config/language.dat が生成されます．

ここまでいくと，あとは `fmtutil-sys --byhyphen ‘kpsewhich --progname=latex language.dat‘` を呼び出すと再生成処理が行われ，利用できるようになります．削除するときもこのコマンドです．

現状，update-language は LATEX 以外の hyphenation 設定ファイルを利用しているものには提供されていません．

5.4.5 フォーマット設定

dh_installtex を利用すればよいです．

fmtutil.cnf(5) に説明してある形式のファイルを/etc/texmf/fmt.d/に配置し，update-fmtutil を実行し，fmtutil-sys –byfmt *format* を実行します．cnf ファイルの最後の行に記述する *format*.ini が見付かった場合だけフォーマットが生成されるので，*format*.ini は conffile であってはいけません．

5.4.6 TeX に Build-Depend する場合のベストプラクティス

Build-Depend するパッケージが変更した設定を必要とする場合，その設定を静的にもつべきではありません．もしパッケージがビルドするのに適切でない設定があるのであれば，それは通常その設定を提供しているパッケージのバグですので，そちらを修正してください．回避策は後々大きな問題となってかえってくる事がおおいです．

もしどうしても必要なら，設定更新プログラムの –outputdir と–add-file を利用して生成してください．

5.4.7 コマンドの実行とフォーマットファイル

TEX のフォーマットが必要な場合は，postinst スクリプトで実施してください．依存するパッケージは Depends: をするだけで十分です．フォーマットファイルの存在を確認などをすると内部構造がかわると壊れやすいのでやめてください．

Debian パッケージは fmtutil か fmtutil-sys を常に利用するべきで，/etc/texmf/fmt.d/ (fmtutil-sys) にファイルを追加するか，fmtutil (–cnffile オプションを指定) で，ローカルの cnf ファイルを更新するようにしてください．

管理者は環境変数を texmf.cnf でオーバライドできます．これが postinst のエラーにつながっている例があります．環境変数はフォーマット生成前などに postinst で unset してください．

5.4.8 Dpkg の Post-Invoke の仕組み

この案は没になりました．

5.5 サンプルコード

(省略)

6 Debian latex の現状調査

上川

まず，Debian の latex で日本語のドキュメントを処理するための手順について確認します．ここでは，例としてドキュメントを準備し，そのドキュメントソースを PDF ファイルにするまでの手順を確認します．

6.1 platex で PDF を作成する方法

platex は ptex-bin パッケージに含まれています．日本語でかかれた tex ファイルから dvi ファイルを生成することができます．

```
$ platex debianmeetingresume200604.tex
```

Debian での platex のデフォルトは，EUC モードです．ソースファイルのエンコーディングは iso-2022-jp か euc-jp にしておくとよいでしょう．SJIS モードでの処理については，Debian パッケージとしてはサポートしていません[*13]

文字コード	可否
EUC-JP	○
SJIS	×
ISO-2022-JP	○
UTF-8	×

dvi ファイルから PDF を作成する方法は，いくつかあります．

- dvipdfmx を利用する
  毎月の Debian 勉強会用の資料を処理するのに利用している方法です．

  ```
  $ dvipdfmx debianmeetingresume200604.dvi
  ```

- dvips で PS を生成し ps2pdf を利用する

  ```
  $ ps2pdf debianmeetingresume200604.ps
  mktexpk: don't know how to create bitmap font for rml.
  dvips: Font rml not found, characters will be left blank.
  $ ps2pdf debianmeetingresume200604.ps
  (結果の PDF ファイルには日本語の文字がまったく表示されない)
  ```

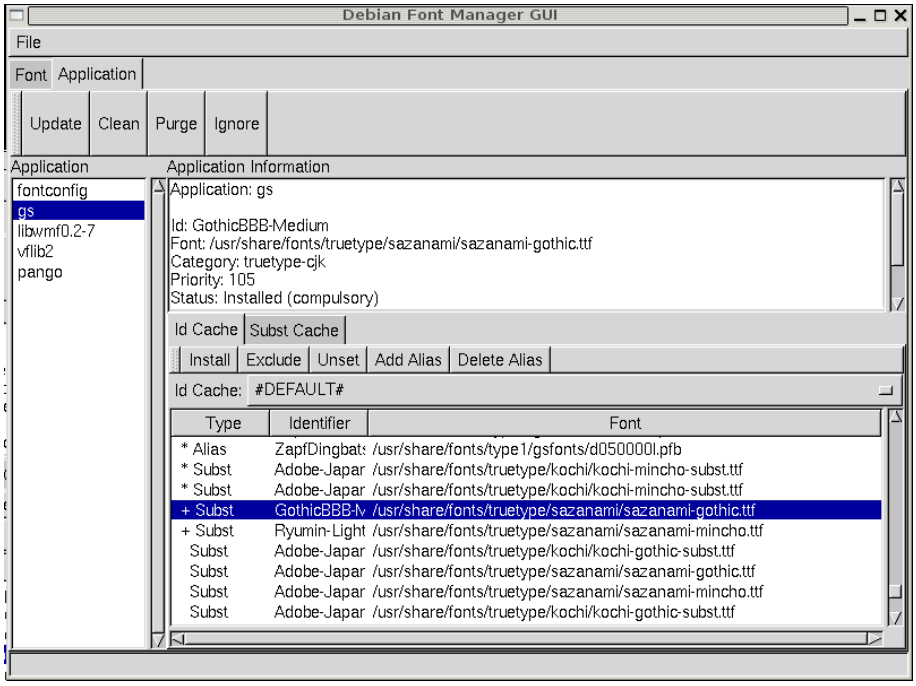
- dvi2ps で PS を生成し，ps2pdf を利用する

  ```
  $ dvi2ps debianmeetingresume200604.dvi > debianmeetingresume200604.ps
  $ GS_LIB=/usr/share/fonts ps2pdf debianmeetingresume200604.ps
  (Ryumin-Light が見付からない，という gs のエラーが出力され途中で停止する)
  ```

  現状 GS_LIB 環境変数の指定が必要になっているのと，Kochi フォントを利用しているとエラーを吐いて停止するという問題があります．dfontmgr を利用して，ps2pdf(gs) が利用するフォントとして kochi フォント以外を指定する必要があります．[*14]

[*13] http://bugs.debian.org/234547

[*14] 参　考：http://lists.debian.or.jp/debian-users/200501/msg00008.html，　http://kmuto.jp/d/index.cgi/debian/gs-esp-8151.htm



それぞれの方法にハイパーリンクや pstricks の扱いに癖があります．たとえば，dvipdfmx の場合は hyperref パッケージを読み込む際に，dvipdfm オプションを指定してあげる必要があります．

```
\usepackage[dvipdfm]{hyperref}
```

6.2 jlatex

jlatex は，jtex-bin パッケージに入っています．tex ファイルから dvi ファイルを生成することができます．

ただ，platex 向けの既存のドキュメントをコンパイルしようとしてもエラーになります． Debian 勉強会資料で利用している jsarticle.cls や ascmac.sty などが platex 専用だからのようです． j-article などを利用する必要があるようです[*15]．また，このドキュメントに関してはそれ以外にも問題があり，簡単な変更では処理できませんでした．

```
 $ jlatex debianmeetingresume200604.tex
! LaTeX Error: File `jsarticle.cls' not found.
```
（エラーがでてコンパイルできない）

6.3 cjk-latex

babel の CJK パッケージとして実装されており，通常の latex を利用して日本語を処理できるそうです．

そのままでは `/usr/share/doc/cjk-latex/examples` にあるサンプルファイルすらコンパイルできないので，困りものです．

参考：`http://lists.debian.or.jp/debian-devel/200007/msg00150.html`

6.4 pdfelatex

Debian には部品が現状足りないようです．

参考：`http://cise.edu.mie-u.ac.jp/~okumura/texfaq/qa/17780.html`

6.5 multex

パッケージをインストールしただけでは，サンプルファイルを処理してもフォントが一部足りないようで，表示されない文字があります．

[*15] jarticle は利用できるようになっている．

参考：http://lists.debian.or.jp/debian-users/200106/msg00081.htm

6.6 lambda (omega)

http://www.fsci.fuk.kindai.ac.jp/kakuto/soft.html, http://cise.edu.mie-u.ac.jp/~okumura/texfaq/japanese/ などを参考にしてみてください．現状，実用的に既存のドキュメントをそのまま処理できるような形式ではないことがうかがえます．

7 次回

5 月 14 日にオンラインで開催予定です。Debian Conference の状況と抱負をお伝えできる予定です.
参加者募集はまた後程。

Debian 勉強会資料

2006 年 4 月 15 日　　初版第 1 刷発行
東京エリア Debian 勉強会 （編集・印刷・発行）