Debian パッケージ 60分クッキング

2008/09/16 @ ylug カーネル読書会

Hideki Yamane <henrich@debian.or.jp>

始める前の諸注意

- 今回も撮影ありなので、
  まずそうな人は先に相談を。
- 途中で質問とかもOK、ただし長いのは後で
  お願いします。
- 今回の資料はどこかのタイミングで
  アップロードする予定です。
- 用意の時間がとれてないので色々と不都合も
  あるかとは思いますが、そこは寛大な心で。

Debian パッケージの作成

作成に必要な技能３つ

1. Shell スクリプトを何となく理解できる
2. Makefile を何となく理解できる
3. ちょっと好奇心がある

Are you ready?

Debian パッケージを作りたい！

大きく分けて２パターン

Debian パッケージを作りたい！

- パターン1
  - 既存のパッケージので気に入らない所がある
    - パッチを当てたい
    - 新しいバージョンにアップデートしたい
    - 新しいパッケージを backport したい
- パターン2
  - 全く Debian パッケージが無い

今日はサラッと両方共やる

では下ごしらえ

必要な構築環境

- Build 環境 = Debian
  - お勧めは Debian unstable でのビルド
    - packaging policy チェッカ（lintian）が最新のものになっていてパッケージングエラーを確認できる
    - stable 用には pbuilder を利用した chroot 環境のクリーンルームな backport を作るやり方で対処可能
    - Debian/Ubuntu 本体に入れるにはどちらにせよ unstable 環境が必要
      - chroot / qemu /VM などで対応という手もあり

今回は…

- 最小限の unstable 環境が準備できた、とする
  - そこに至る手順は？
    - 自分が使いやすい環境でどうぞ
      - VMWare?
      - VirtualBox?
      - Chroot? (debootstrap)

ビルド者の名前とアドレス設定

- 名前 = 環境変数 $DEBFULLNAME
- アドレス = 環境変数 $DEBEMAIL
- 環境変数を export せよ！ export せよ！
  - ~/.bashrc にこんな風に設定してみたり。

```
DEBEMAIL=henrich@debian.or.jp
export DEBEMAIL

DEBFULLNAME="Hideki Yamane (Debian-JP)"
export DEBFULLNAME
```

使うエディタの設定

- `ls -al /etc/alternatives/editor`
  - パッケージ用ツールで呼び出されるエディタはどれ？
    - 最初は nano が設定されている
      - 個人的には vim を設定
      - Emacs も選べますよ
- `sudo update-alternatives --config editor`
  - 入ってなければ先にパッケージ入れてね

必要なパッケージ

- dh-make
- devscripts
- debhelper
- fakeroot
- lintian
- sudo
- pbuilder
- piuparts
- dpatch/quilt

```
$ sudo aptitude install dh-make devscripts
debhelper fakeroot lintian sudo pbuilder
piuparts dpatch quilt
```

aptitude install
ほげふが

Debian パッケージを作りたい！

- パターン1
  - 既存のパッケージので気に入らない所がある
    - パッチを当てたい
    - 新しいバージョンにアップデートしたい
    - 新しいパッケージを backport したい

パターン1

既存パッケージの修正

既存パッケージの修正

1. ソースパッケージを入手して
2. debian/* を変更して
   （configureオプションを変更して etc...）
3. debian/changelog に変更した旨のエントリと
   ローカルのバージョンを記述
4. ビルドに必要な依存パッケージをインストール
   しておく
5. ビルド

ソースパッケージの入手？

- /etc/apt/sources.list あるいは /etc/apt/sources.list.d/*.list 参照

```
deb http://ftp.jp.debian.org/debian/ unstable main contrib non-free
deb-src http://ftp.jp.debian.org/debian/ unstable main contrib non-free

deb http://cdn.debian.or.jp/debian/ experimental main contrib non-free
deb-src http://cdn.debian.or.jp/debian/ experimental main contrib non-free
```

- sudo apt-get update

ソースパッケージの入手

- `apt-get source <packagename>`
  （一般ユーザ権限で問題無し）

どのパッケージ？

今回の対象

GNU Hello

hello?

- aptitude show hello

ソースパッケージ

```
apt-get source hello
```

Do it now!

ソースパッケージ

- 3つからなる
  - *.orig.tar.gz (= upstream source)
  - *.diff.gz (= debian パッケージ)
  - *.dsc (パッケージ内容のチェック用ファイル)
- 先ほどの apt-get source <packagename> で自動的に展開される
  - あとは <packagename>/debian ディレクトリ以下のファイルを適切にいじってリビルド

修正対象について注意

- パッケージの debian ディレクトリ以下は自由にいじる
- 本体のソースは**直接いじらない**
  - Makefile とか…
  - いじるなら dpatch/quilt 等のパッチシステムを使うこと

debian/changelog

- 作業が終わったらその内容をchangelogに記述
  - dch コマンドを使うと便利
  - バージョン番号に注意

demo

ビルドに必要な依存パッケージ

- たくさんあって一々入れるの面倒
  - メモでもとっておく？
- `sudo aptitude build-dep <packagename>`
  - 一気に解決。便利。

ビルド！

- debuild コマンドを使う
  - -us
    - .dsc ファイルにサインしない
    - 本来は GPG 鍵を用意しておいてサインするべき
  - -uc
    - .changes ファイルにサインしない
    - 本来は（ry
  - 詳しくは man debuild

demo

パッケージのアップデート

- 開発元では新しいバージョンが出ている
  - 手軽にアップデート？

uupdate

uupdate コマンド

- hoge-1.0 から hoge-1.1 にアップデート
  - hoge-1.0 ディレクトリで uupdate コマンドを実行

demo

もっと簡単に出来ない？

- 実は出来ることもある
  - 鍵は debian/watch ファイル
    - これがきちんと記述されていると…

demo

uscan rocks!!

出来たパッケージは？

- 一台なら sudo dpkg -i *.deb もあり
- 複数台で運用するなら専用のリポジトリを作ってそこに放り込む
  - mini-dinstall?（まだ私は使ったこと無い）
  - apt-ftparchives （ちょっと原始的すぎかな）
- そして、インストールしたいマシンの apt line (/etc/apt/sources.list)に追加して
  - sudo aptitude update
  - sudo aptitude install <packagename>

demo

簡易リポジトリの問題点

- 真正なパッケージかどうか（偽者が作った rootkit 仕込みのパッケージだったりしない？）のチェックが出来ない
  - あくまで「簡易」な点に注意
  - もっとちゃんとやるには…また今度。

第一部
既存パッケージの修正

完

Any questions?

さて

全くパッケージが無い、という場合？

- rpm ならあるんだけどなぁ…
  - 邪法を使えば簡単にできる
  - alien!

demo

alien が邪法な訳

- Debian での様々な状況を考慮してパッケージが作られているわけでは無いので不整合が出やすい
- あくまでも「逃げ」
  - パッケージ作成サンプルとしてみるのは良い。

全くパッケージが無い、という場合？

- ソースを持ってきてパッケージにしないと…
  - さて、今日は何をパッケージにしようか？

今回のサンプル

まずはビルドしてみるか

- ソースをダウンロードして展開
- そもそもパッケージにしなくてもビルドできるようにする

demo

先生！ファイルが足りません！

- エラーメッセージを見ると、このファイルが必要だけど、どのパッケージにあるんだろう…？？
  - **apt-file** を使う
    - `sudo aptitude install apt-file && sudo apt-file update`
    - `apt-file search (目的のファイル)`

ビルドできたらパッケージに…

- ダウンロードしたソースファイル
  - 必要であればリネーム
    - (packagename)-(version) にしてから展開＆移動
    - dh_make

demo

では、パッケージに…

- ダウンロードしたソースファイル
  - 必要であればリネーム
    - (packagename)-(version) にしてから展開＆移動
    - dh_make
      - dh_make --creatorig
      - インタラクティブに返答
    - cd <package>/debian
    - いらないファイルを除去
    - ファイルの編集
      debian/copyright, debian/control, debian/rules

debian/copyright

- ファイルの由来やライセンス情報を記載
  - ある意味一番面倒なファイル

debian/control

- パッケージ情報の管理
  - パッケージ名だとか
  - どのセクションに属しているのかとか
  - 依存関係とか
  - パッケージの説明文とか

debian/rules

- 平たく言うと Makefile
  - Build 時にこの Makefile が使われる

demo

debian/changelog 編集

- dch コマンドで。

Build !

- `debuild` コマンドを使ってビルド

demo

パッケージテスト

- 本当に自分以外の環境でビルドできるか？
- インストール／アンインストール時に問題が出ないのか？

パッケージテスト

- 本当に自分以外の環境でビルドできるか？
  - **pbuilder**
    - クリーンルーム環境を都度作成してその中でビルドチェック
    - `sudo pbuilder --create`（一度やればok）
    - `sudo pbuilder --build *.dsc`

demo

パッケージテスト

- インストール／アンインストール時に問題が出ないのか？
  - **piuparts**
  - クリーンルーム環境でパッケージのインストールとアンインストールを実施する
  - `sudo piuparts *.deb`

demo

パッケージとしての品質チェック

- パッケージングポリシーにどの程度準拠しているの？
  - パッケージングポリシー？
    - debian-policy パッケージとして提供されている
      - 現在、バージョン 3.8.0.1
    - Webからも見れる
    - 日本語訳も Debian JP のページにあるよ

パッケージとしての品質チェック

- どうやってポリシー準拠を確認する
  - lintian ツール
    - Perl で実装されている
      - Python 版の linda はお亡くなりになった
    - debuild するときに警告してくれます
  - Official なパッケージだとウェブから確認可能
    - http://lintian.debian.org/maintainer/(mailaddress).html

demo

パッケージとしての品質チェック

- 自分の環境以外できちんとビルドできる？
- 問題なくインストール／アンインストールできる？
- パッケージングポリシーにどの程度準拠してる？
  - パッケージ品質が良いと様々な環境で使える
    - 逆に言うとパッケージとしての品質が悪いとちょっと環境が変わると使えないパッケージになる。
- Eat your own dog food!
  - もちろん、実際にインストールして動作確認も行いましょう。

第二部
新規パッケージの作成

完

俺たちはまだ登り始めたばかりだからな

このでびあん坂という長い長い坂を！

完

ylug 先生の次回作にご期待ください!