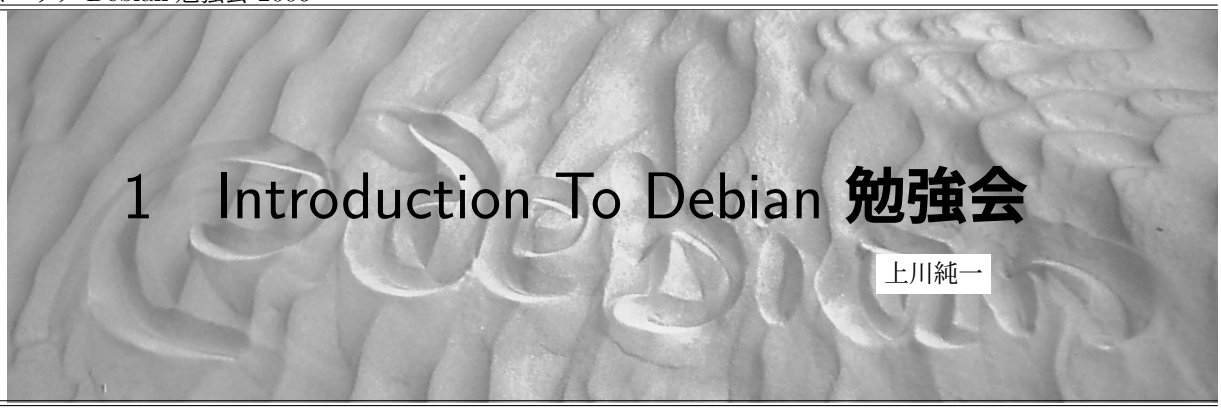
第 19 回 東京エリア Debian 勉強会
事前資料

Debian 勉強会会場係 上川純一*

2006 年 8 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 誕生日に思うこと」、「ysjj さんへの誕生日のメッセージ」もしくは「最近感銘を受けたパッケージ」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

1.2.2 上川

Debian も、もう 13 周年です。よくわからないですが、老舗です。また、Debian　は常に変化しています。毎年違う姿を見せてくれているように思います。波瀾万丈、また愉しきかな。

今後ともよろしくお願いします。

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 25 号

`http://www.debian.org/News/weekly/2006//` にある 6 月 20 日版です。

問題 1. Isaac Clerencia さんは、スペインのサラゴサ市当局が、6 ヶ所ある場所に Debian ベースのシンクライアントを設置したと報告しました。その場所とはどこでしょうか

□ A ラーメン屋

□ B コンビニエンスストア

□ C 老人ホーム

問題 2. Yaroslav Halchenko さんが Debian Packge 内のあるファイルが圧縮されていて、読むことが出来ないと気がつきました。そのファイルとは何でしょうか。

□ A Word ファイル

□ B PDF ファイル

□ C SREC ファイル

2.2 2006 年 26 号

`http://www.debian.org/News/weekly/2006//` にある 6 月 27 日版です。

問題 3. 9 月にイタリアのある都市で Debian コミュニティカンファレンスがおこなわれます。そのある都市とはどこでしょう。

□ A ベニス

□ B デセンツァーノ・デル・ガルーダ

□ C カリブ島

問題 4. 最近セキュリティチームのメンバーが増えました。それはだれでしょうか。

□ A Steve Kemp

□ B Hidehazu Koiwa

□ C Andreas Barth

2.3 2006 年 27 号

http://www.debian.org/News/weekly/2006/ にある 7 月 4 日版です。
問題 5. 最近また新しい OS に Debian を移植している噂があるらしい。それはどの OS か？

☐ A Plan9

☐ B Minix3

☐ C Mona

問題 6. Paul Wise さんがあたらしいグループを作成しました。それはどのグループか？

☐ A debian-smoker

☐ B debian-soccer

☐ C debian-flash

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 18 回目報告

7 月の Debian 勉強会は北海道で開催されました。上川が Debian の紹介、Debian 勉強会の紹介、MacBook に Debian をインストールする方法について説明しました。

今回の参加人数は 40 人くらいでした。

会場で MacBook をすでに購入しているひとは数人でした。今回の参加によって東京での Debian 勉強会に参加しようと思ったひとは 5 人くらいいました。

3.2 TLUG

「TLUG」にて Debian を　Macbook にインストールする方法について報告してきました。英語で発表しました。プレゼンテーションを Front Row コントローラでやったのは好評だったようです。

4 Debcof in Japan 北海道の検討について

岩松

4.1 はじめに

7/14 から 7/17 まで 上川さんと私は札幌に行きました。Debconf を 札幌で行うことが可能か、情報収集および打ち合わせをしてきました。その報告を行います。

4.2 札幌 までの道のり

東京から行く場合は

- 各都内の駅 → 浜松町
- 浜松町 → 東京モノレール 第 2 空港ビル下車 (約 20 分) 470 円
- 羽田空港 → 新千歳空港 (約 1 時間 30 分)
- 新千歳空港 → JR 札幌駅 (約 40 分) 1040 円

となります。
海外の方が札幌まで行く場合は

- 成田空港到着
- 成田空港 → 新千歳空港
- 新千歳空港 → JR 札幌駅

成田空港 - 新千歳空港は 1 日以下の便があります。

	便名	時刻
ANA	3121	10:30 → 12:05
ANA	2155	17:55 → 19:30
日本航空	3047	18:40 → 20:20

4.3 産業省さんとの検討内容

4.3.1 市立大学

WebSite:http://www.scu.ac.jp/ 森の中にある。交通は不便。泊まるところもない。テント張る？

4.3.2 コンベンションセンターはどうか

Website : http://www.sora-scc.jp/ 場所が遠い。泊まるところが近くにない。ネットワークがなく、別途引く必要あり。時間の制限がある可能性がある。

4.3.3 学術工芸会館

大聖堂がある。オペラ等で使うのでちょっと違う。ネットワークがない。

4.3.4 時期は？

5 月はいい季節。寒いので汗をかかない。風呂にあまり入らないハッカーとその回りの人にはいい時期。

4.3.5 組織の問題

○○実行委員会とか。ビザの発行手続きの問題があるので必要。

4.4 北海道大学さんとの検討内容

4.4.1 セッション/BOF 等

学術交流会館情報教育館（日本 PostgreSQL ユーザ会北海道支部が使っているらしい。）

4.4.2 ハックラボ

北海道大学情報科学研究科棟 (OSC Hokakido 開催場所)

4.4.3 検討事項

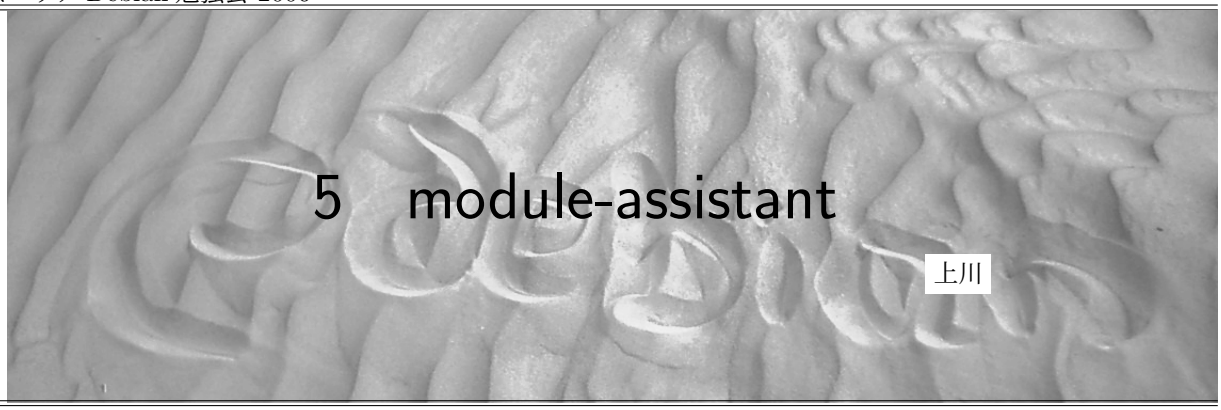
- 宿泊施設に関して:
  場所は北海道大学の講義室を借りることが可能。しかし、北海道大学周辺には大規模な宿泊施設がなく、ハックラボやセッションへの移動が困難である。北海道大学を中心で行う場合は、この問題を解決する必要がある。
- 時間の制限:
  各部屋を利用する場合、使用する時間が設けられているため、21 時ぐらいに部屋を出ないといけない。これは開催側からハックできる場所が提供できないということなので、問題である。できればハックできる時間を多くとらせてあげたいので、長時間、できれは 24 時間体制で提供できるようにする必要がある。
- 電源の問題:
  古い建物では電源に関してあまり考えられてないため、300 人近くが同時に PC を立ち上げたりすると、電源容量の問題が発生する。北海道大学は場所によっては問題なく使用できるところがあるが、場所が離れているため、移動が困難。ハックラボとセッション、BOF ができる部屋はできるだけ近くで行えるようにしたい。
- 食事:
  食堂のプリペイドカードが使える。開催時は食堂のメニューにベジタリアン用メニューを作ってもらう。

4.5 その他

- 会社の研修施設:
  企業が利用している 研修施設を利用するのはどうか？
- 交通:
  成田から北海道までの交通費が高い。往復で 5 万円する。

4.6 下見をしたところ

札幌ゲストハウス（札幌天神山国際ハウス）: 丘の上にある。地下鉄東西線の澄川駅の近く。30 人ぐらいしか泊まれない。



5 module-assistant

上川

Debian 用にカーネルモジュールパッケージを作成する方法について解説します。

5.1 従来の方法、kernel-package

従来は、kernel-package にて実装されていた方法を利用することが多かったです。最近は、その簡便さから流行は module-assistant に移行しはじめているようです。

5.2 module-assistant の利用

module-assistant はパッケージのインストール、ソースの展開、ビルド、インストールまでの一連の作業を自動化してくれるツールです。

まず、カーネルソースの場所を教えます。通常、現在実行中のカーネルに対してのソースのディレクトリは `/lib/modules/$(uname -r)/build` のシンボリックリンクをたどることで取得できます。そのため、何も指定しないで下記のコマンドをうつと設定できます。

```
# m-a prepare
```

一度カーネルソース、もしくはカーネルヘッダパッケージを適切にインストール・設定したあとはパッケージ名を指定してあげれば、そのパッケージからモジュールを作成し、インストールするところまで実施してくれます。

```
# m-a a-i \textit{package}
```

このコマンドを入力すると、*package*-source パッケージをインストールし、/usr/src/*package*.tar.bz2 を展開、モジュールパッケージをビルドして、パッケージをインストールしてくれます。

出力例を見てみましょう。

```
# m-a --text-mode a-i linux-uvc
.
1 パッケージについての情報を更新しました
Getting source for kernel version: 2.6.18-rc2dancer-gb4e54de8-dirty
/lib/modules/2.6.18-rc2dancer-gb4e54de8-dirty/source のカーネルヘッダを利用できます
apt-get install build-essential
パッケージリストを読み込んでいます... 完了
依存関係ツリーを作成しています... 完了
build-essential はすでに最新バージョンです。
アップグレード: 0 個、新規インストール: 0 個、削除: 0 個、保留: 122 個。

完了!
download
パッケージリストを読み込んでいます... 完了
依存関係ツリーを作成しています... 完了
以下のパッケージが新たにインストールされます:
  linux-uvc-source
アップグレード: 0 個、新規インストール: 1 個、削除: 0 個、保留: 122 個。
32.7kB 中 0B のアーカイブを取得する必要があります。
展開後に追加で 73.7kB のディスク容量が消費されます。
パッケージフィールドを読み込んでいます... 完了
パッケージ状態を読み込んでいます... 完了
バグレポートを取得しています... 完了
未選択パッケージ linux-uvc-source を選択しています。
(データベースを読み込んでいます ... 現在 131883 個のファイルとディレクトリがインストールされています。)
(.../linux-uvc-source_0.1.0-2_amd64.deb から) linux-uvc-source を展開しています...
linux-uvc-source (0.1.0-2) を設定しています ...
linux-uvc-source についての情報を更新中

1 パッケージについての情報を更新しました
unpack
Extracting the package tarball, /usr/src/linux-uvc.tar.bz2, please wait...
"/usr/share/modass/packages/default.sh" build
 KVERS=2.6.18-rc2dancer-gb4e54de8-dirty
 KSRC=/lib/modules/2.6.18-rc2dancer-gb4e54de8-dirty/source KDREV=GIT.2006.07.25.15.08 kdist_image
dh_testdir
dh_testroot
dh_clean
/usr/bin/make -C /usr/src/modules/linux-uvc clean \
        KERNELPATH=/lib/modules/2.6.18-rc2dancer-gb4e54de8-dirty/source
KERNELRELEASE=2.6.18-rc2dancer-gb4e54de8-dirty KERNELCONF=/lib/modules/2.6.18-rc2dancer-gb4e54de8-dirty/source/.config

[中略]

dh_builddeb --destdir=/home/dancer/shared/git
tar: -: file name read contains nul character
dpkg-deb:
 /home/dancer/shared/git/linux-uvc-modules-2.6.18-rc2dancer-gb4e54de8-dirty_0.1.0-2+GIT.2006.07.25.15.08_amd64.deb
にパッケージ ‘linux-uvc-modules-2.6.18-rc2dancer-gb4e54de8-dirty’ を構築しています。
make[1]: ディレクトリ ‘/usr/src/modules/linux-uvc’ から出ます
/usr/bin/make  -f debian/rules kdist_clean
make[1]: ディレクトリ ‘/usr/src/modules/linux-uvc’ に入ります
dh_testdir
dh_testroot
dh_clean
/usr/bin/make -C /usr/src/modules/linux-uvc clean \
        KERNELPATH=/lib/modules/2.6.18-rc2dancer-gb4e54de8-dirty/source
 KERNELRELEASE=2.6.18-rc2dancer-gb4e54de8-dirty
 KERNELCONF=/lib/modules/2.6.18-rc2dancer-gb4e54de8-dirty/source/.config
make[2]: ディレクトリ ‘/usr/src/modules/linux-uvc’ に入ります
rm -f *.o *.ko .*.cmd .*.flags *.mod.c Modules.symvers
rm -rf .tmp_versions
make[2]: ディレクトリ ‘/usr/src/modules/linux-uvc’ から出ます
make[1]: ディレクトリ ‘/usr/src/modules/linux-uvc’ から出ます
dpkg -Ei
 /home/dancer/shared/git/linux-uvc-modules-2.6.18-rc2dancer-gb4e54de8-dirty_0.1.0-2+GIT.2006.07.25.15.08_amd64.deb
未選択パッケージ linux-uvc-modules-2.6.18-rc2dancer-gb4e54de8-dirty を選択しています。
(データベースを読み込んでいます ... 現在 131888 個のファイルとディレクトリがインストールされています。)
(.../linux-uvc-modules-2.6.18-rc2dancer-gb4e54de8-dirty_0.1.0-2+GIT.2006.07.25.15.08_amd64.deb
 から) linux-uvc-modules-2.6.18-rc2dancer-gb4e54de8-dirty を展開しています...
linux-uvc-modules-2.6.18-rc2dancer-gb4e54de8-dirty (0.1.0-2+GIT.2006.07.25.15.08) を設定しています ...
```

5.3 パッケージの作成方法

まず、カーネルモジュールのソースを提供するためのパッケージを作成します。今回の例では、tt-source パッケージです。module-assistant から処理できるように、**パッケージ名**-source という命名規則に従ってください。tt-source パッケージはモジュールのソースを/usr/src/package.tar.bz2 をインストールします。/usr/src/package.tar.bz2 の中身自体は、また Debian ソースパッケージの形態になっており、module-assistant が処理してカーネルモジュールパッケージを生成できるようになっています。

module-assistant 用のソースの中に debian/*.modules.in というファイル名でおいてあると、その中にある_KVERS_ と書いている部分がカーネルのバージョン番号に置換されます。登場するファイルの代表的なものを簡単に説明します。

- rules
- control.modules.in: モジュールパッケージのコントロールファイルです。
- postinst.modules.in: インストールされたときに実行されます。depmod などを実行します。

5.3.1 dh-make を利用した例

dh_make を利用して作成してみましょう。ドライバの名前は仮に tt とします。

```
$ dh_make

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

Maintainer name : Junichi Uekawa
Email-Address   : dancer@debian.org
Date            : Sat, 29 Jul 2006 13:13:32 +0900
Package Name    : tt
Version         : 1
License         : blank
Type of Package : Kernel Module
Hit <enter> to confirm:
Skipping creating ../tt_1.orig.tar.gz because it already exists
Done. Please edit the files in the debian/ subdirectory now. You should also
check that the tt Makefiles install into $DESTDIR and not in / .
```

作成されるソースパッケージのディレクトリ構造は下記のようになっています。

- driver/: ドライバのソース
- ./: 通常のソース
- debian/: debian の通常のソースパッケージに必要なもろもろのファイル

パッケージを作成して debc で確認してみます。tt-source というパッケージができています。/usr/src/tt.tar.bz2 というファイルが作成されています。この中身を作成するのがメインの仕事になります。

```
tt-source_1-1_all.deb
---------------------
新形式 debian パッケージ、バージョン 2.0。
サイズ 5608 バイト: コントロールアーカイブ = 576 バイト。
     466 バイト,    13 行      control
     199 バイト,     3 行      md5sums
Package: tt-source
Version: 1-1
Section: unknown
Priority: optional
Architecture: all
Depends: module-assistant, debhelper (>> 4.0.0), make, bzip2
Installed-Size: 12
Maintainer: Junichi Uekawa <dancer@debian.org>
Source: tt
Description: Source for the tt driver.
 This package provides the source code for the tt kernel modules.
 The tt package is also required in order to make use of these
 modules. Kernel source or headers are required to compile these modules.
drwxr-xr-x root/root         0 2006-07-29 13:27 ./
drwxr-xr-x root/root         0 2006-07-29 13:27 ./usr/
drwxr-xr-x root/root         0 2006-07-29 13:27 ./usr/share/
drwxr-xr-x root/root         0 2006-07-29 13:27 ./usr/share/doc/
drwxr-xr-x root/root         0 2006-07-29 13:27 ./usr/share/doc/tt-source/
-rw-r--r-- root/root       188 2006-07-29 13:13 ./usr/share/doc/tt-source/changelog.Debian.gz
-rw-r--r-- root/root       627 2006-07-29 13:13 ./usr/share/doc/tt-source/copyright
drwxr-xr-x root/root         0 2006-07-29 13:27 ./usr/src/
-rw-r--r-- root/root      3832 2006-07-29 13:27 ./usr/src/tt.tar.bz2

tt_1-1_i386.deb
---------------
新形式 debian パッケージ、バージョン 2.0。
サイズ 1928 バイト: コントロールアーカイブ = 452 バイト。
     243 バイト,     8 行      control
     197 バイト,     3 行      md5sums
Package: tt
Version: 1-1
Section: unknown
Priority: optional
Architecture: i386
Installed-Size: 12
Maintainer: Junichi Uekawa <dancer@debian.org>
Description: <insert up to 60 chars description> <insert long description, indented with spaces>
drwxr-xr-x root/root         0 2006-07-29 13:27 ./
drwxr-xr-x root/root         0 2006-07-29 13:27 ./usr/
drwxr-xr-x root/root         0 2006-07-29 13:27 ./usr/share/
drwxr-xr-x root/root         0 2006-07-29 13:27 ./usr/share/doc/
drwxr-xr-x root/root         0 2006-07-29 13:27 ./usr/share/doc/tt/
-rw-r--r-- root/root       188 2006-07-29 13:13 ./usr/share/doc/tt/changelog.Debian.gz
-rw-r--r-- root/root       627 2006-07-29 13:13 ./usr/share/doc/tt/copyright
-rw-r--r-- root/root       926 2006-07-29 13:13 ./usr/share/doc/tt/README.Debian
drwxr-xr-x root/root         0 2006-07-29 13:27 ./usr/sbin/
drwxr-xr-x root/root         0 2006-07-29 13:27 ./usr/bin/
```

tt.tar.bz2 の中身をみてみましょう。

```
$ tar tfj tt-source/usr/src/tt.tar.bz2
 modules/
 modules/tt/
 modules/tt/Makefile
 modules/tt/debian/
 modules/tt/debian/compat
 modules/tt/debian/copyright
 modules/tt/debian/changelog
 modules/tt/debian/rules
 modules/tt/debian/tt-modules-_KVERS_.postinst.modules.in.ex
 modules/tt/debian/control.modules.in
```

通常の Debian のソースパッケージの形になっています。これらは、make-kpkg か、module-assistant から利用されることを想定しています。debian/rules はそのままコピーしてきたものを利用しています。binary-modules という特別なターゲットが準備してあり、モジュールをビルドする際に呼び出されます。binary-modules は通常のカーネルのモジュールのビルドと同じことをします。注意するべきは、Debian パッケージを作成するために、実行時に必要なディレクトリ (/lib/modules/以下) に直接インストールするのではなく、deb パッケージとして配布するためのディレクトリにインストールするという点です。

```
cp drivers/slusb.$ko drivers/slamr.$ko debian/$(PKGNAME)/lib/modules/$(KVERS)/misc
```

control.modules.in の中身を見てみます。カーネルのバージョンによって置換される文字列が _KVERS_ という文字列になっています。control.modules.in は module-assistant などにより処理され、各バージョン用の内容に変更されます。同様に、_KVERS_ を置換してもらうことを前提として、 postinst.modules.in などを作成することが可能です。

```
Source: tt
Section: unknown
Priority: optional
Maintainer: Junichi Uekawa <dancer@debian.org>
Build-Depends: debhelper (>> 4.0.0)
Standards-Version: 3.7.2

Package: tt-modules-_KVERS_
Architecture: any
Provides: tt-modules
Description: tt modules for Linux (kernel _KVERS_).
 This package contains the set of loadable kernel modules for the
 <description>.
 .
 This package contains the compiled kernel modules for _KVERS_
 .
 If you have compiled your own kernel, you will most likely need to build
 your own tt-modules. The tt-source package has been
 provided for use with the Debian's module-assistant or kernel-package
 utilities to produce a version of tt-module for your kernel.
```

5.3.2 CDBS を利用した例

madwifi カーネルモジュールなどは CDBS を利用したパッケージ構成になっています。それでは、CDBS を利用した例を見てみましょう。debhelper のみを利用している場合、テンプレート的に毎回同様の内容を記述する部分が多数ありますが、CDBS を利用することによって、その部分を CDBS に任せ、差異のある部分に集中することができます。

それでは例で見てみましょう。

まず、メインの debian/rules を見てみます。ここでは、linux-uvc パッケージを例にとります。

```
#!/usr/bin/make -f
#mostly copied from madwifi debian/rules

include /usr/share/cdbs/1/rules/debhelper.mk
include /usr/share/cdbs/1/rules/dpatch.mk
include /usr/share/dpatch/dpatch.make

build/linux-uvc-tools::
        -$(MAKE) extract

install/linux-uvc-source::
        # Enforce executable bit on debian/rules, and create directory
        # structure for modules source
        install -D -m 0755 debian/rules.modules \
                debian/tmp/modules/linux-uvc/debian/rules
        # Prepare the other debian stuff
        for f in *.modules.in control compat copyright changelog README.Debian; do \
                install -m 0644 debian/$$f \
                        debian/tmp/modules/linux-uvc/debian/; \
        done
        # Prepare upstream source
        find . -path ./debian/\* -type d -prune -o -printf "%P\n" | \
                egrep -v 'debian|contrib|regression|.svn' | \
                cpio -admp debian/tmp/modules/linux-uvc/
        # clean it
        -$(MAKE) -C debian/tmp/modules/linux-uvc/ clean
        -rm -f debian/tmp/modules/linux-uvc/extract
        # Prepare the debian source tarball
        tar jcf debian/linux-uvc-source/usr/src/linux-uvc.tar.bz2 \
                -C debian/tmp modules

install/linux-uvc-tools::
        install -D -m 0755 extract debian/linux-uvc-tools/usr/sbin/macbook-isight-firmware-loader

clean::
        -rm -f extract
```

cdbs では、各 build ターゲットと install ターゲットを build/パッケージ名、install/パッケージ名として定義しているのがわかります。また、dh_系のコマンドがまったく書かれていません。debian/rules -p を実行すると、それが cdbs にてどういう風に補完されるのかが確認できます。たとえば、linux-uvc-source については、おおざっぱに把握すると下記のように展開されることがわかります。

```
install/linux-uvc-source::
        # Enforce executable bit on debian/rules, and create directory
        # structure for modules source
        install -D -m 0755 debian/rules.modules \
        debian/tmp/modules/linux-uvc/debian/rules
        # Prepare the other debian stuff
        for f in *.modules.in control compat copyright changelog README.Debian; do \
        install -m 0644 debian/$$f \
        debian/tmp/modules/linux-uvc/debian/; \
        done
        # Prepare upstream source
        find . -path ./debian/\* -type d -prune -o -printf "%P\n" | \
        egrep -v 'debian|contrib|regression|.svn' | \
        cpio -admp debian/tmp/modules/linux-uvc/
        # clean it
        -$(MAKE) -C debian/tmp/modules/linux-uvc/ clean
        -rm -f debian/tmp/modules/linux-uvc/extract
        # Prepare the debian source tarball
        tar jcf debian/linux-uvc-source/usr/src/linux-uvc.tar.bz2 \
        -C debian/tmp modules

binary-install/linux-uvc-source::
        dh_installdocs -p$(cdbs_curpkg) $(DEB_INSTALL_DOCS_ALL) $(DEB_INSTALL_DOCS_$(cdbs_curpkg))
        dh_installexamples -p$(cdbs_curpkg) $(DEB_INSTALL_EXAMPLES_$(cdbs_curpkg))
        dh_installman -p$(cdbs_curpkg) $(DEB_INSTALL_MANPAGES_$(cdbs_curpkg))
        dh_installinfo -p$(cdbs_curpkg) $(DEB_INSTALL_INFO_$(cdbs_curpkg))
        dh_installmenu -p$(cdbs_curpkg) $(DEB_DH_INSTALL_MENU_ARGS)
        dh_installcron -p$(cdbs_curpkg) $(DEB_DH_INSTALL_CRON_ARGS)
        dh_installinit -p$(cdbs_curpkg) $(if
 $(DEB_UPDATE_RCD_PARAMS),--update-rcd-params="$(call
 cdbs_strip_quotes,$(DEB_UPDATE_RCD_PARAMS))",$(if
 $(DEB_UPDATE_RCD_PARAMS_$(cdbs_curpkg)),--update-rcd-params="$(call
 cdbs_strip_quotes,$(DEB_UPDATE_RCD_PARAMS_$(cdbs_curpkg)))"))
 $(DEB_DH_INSTALLINIT_ARGS)
        dh_installdebconf -p$(cdbs_curpkg) $(DEB_DH_INSTALLDEBCONF_ARGS)
        dh_installemacsen -p$(cdbs_curpkg) $(if
 $(DEB_EMACS_PRIORITY),--priority=$(DEB_EMACS_PRIORITY)) $(if
 $(DEB_EMACS_FLAVOR),--flavor=$(DEB_EMACS_FLAVOR))
 $(DEB_DH_INSTALLEMACSEN_ARGS)
        dh_installcatalogs -p$(cdbs_curpkg) $(DEB_DH_INSTALLCATALOGS_ARGS)
        dh_installpam -p$(cdbs_curpkg) $(DEB_DH_INSTALLPAM_ARGS)
        dh_installlogrotate -p$(cdbs_curpkg) $(DEB_DH_INSTALLLOGROTATE_ARGS)
        dh_installlogcheck -p$(cdbs_curpkg) $(DEB_DH_INSTALLLOGCHECK_ARGS)
        dh_installmime -p$(cdbs_curpkg) $(DEB_DH_INSTALLMIME_ARGS)
        dh_installchangelogs -p$(cdbs_curpkg)
 $(DEB_DH_INSTALLCHANGELOGS_ARGS) $(DEB_INSTALL_CHANGELOGS_ALL)
 $(DEB_INSTALL_CHANGELOGS_$(cdbs_curpkg))
        $(if $(wildcard /usr/bin/dh_installudev),dh_installudev -p$(cdbs_curpkg) $(DEB_DH_INSTALLUDEV_ARGS))
        dh_install -p$(cdbs_curpkg) $(if
 $(DEB_DH_INSTALL_SOURCEDIR),--sourcedir=$(DEB_DH_INSTALL_SOURCEDIR))
 $(DEB_DH_INSTALL_ARGS)
        dh_link -p$(cdbs_curpkg) $(DEB_DH_LINK_ARGS) $(DEB_DH_LINK_$(cdbs_curpkg))
```

実行されているおかげで、debhelper 関連のコードを直接記述する必要が無くなりました。

ただ、linux-uvc はモジュールのビルド用の rules ファイルを分離しています。cdbs の記述ときれいに分離するための措置です。`debian/rules.modules` というものを別途用意しており、それをモジュールのソースの `debian/rules` として利用しています。module-assistant は cdbs を必要としないので、モジュールソースパッケージをインストールするエンドユーザに追加で cdbs をインストールすることを強要しないという点も重要です。

```
#!/usr/bin/make -f

PACKAGE := linux-uvc-modules
MA_DIR ?= /usr/share/modass
-include $(MA_DIR)/include/generic.make
-include $(MA_DIR)/include/common-rules.make

.PHONY: kdist_config
kdist_config: prep-deb-files

.PHONY: binary-modules
binary-modules: kdist_config
        dh_testdir
        dh_testroot
        dh_clean -k
        # Build modules
        $(MAKE) -C $(CURDIR) uvcvideo \
        KERNELPATH=$(KSRC) KERNELRELEASE=$(KVERS) KERNELCONF=$(KSRC)/.config
        # Install modules
        $(MAKE) -C $(CURDIR) install \
        KERNELPATH=$(KSRC) KERNELRELEASE=$(KVERS) KERNELCONF=$(KSRC)/.config \
        INSTALL_MOD_PATH=$(CURDIR)/debian/$(PKGNAME)
 KMODPATH=/lib/modules/$(KVERS)/kernel/drivers/usb/media
        # remove depmod result
        -rm -f $(CURDIR)/debian/$(PKGNAME)/lib/modules/$(KVERS)/modules.*

        dh_installdocs
        dh_installchangelogs
        dh_compress
        dh_fixperms
        dh_installmodules
        dh_installdeb
        dh_gencontrol -- -v$(VERSION)
        dh_md5sums
        dh_builddeb --destdir=$(DEB_DESTDIR)

.PHONY: kdist_clean
kdist_clean:
        dh_testdir
        dh_testroot
        dh_clean
        $(MAKE) -C $(CURDIR) clean \
        KERNELPATH=$(KSRC) KERNELRELEASE=$(KVERS) KERNELCONF=$(KSRC)/.config
```

5.4 参考文献

- CDBS のルールを展開して出力する方法
  `http://syn.theti.ca/articles/2006/07/27/plumbing-the-depths-of-cdbs`
- man 8 module-assistant
- テンプレート `/usr/share/doc/module-assistant/examples/templates-debian-dir.tar.bz2`

6 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 8 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）