# 地球流体電脳倶楽部的

# ubnutu 12.04 入門

地球流体電脳倶楽部

**2013/Sep/15**

# 目次

# 第1章　はじめに

Ubuntu をはじめとする、ディストリビューターの努力によって Linux 環境自体は、昔とは比べものにならないくらい導入が容易になった。しかし、それにともなって、提供されるパッケージは爆発的に豊富になり、同じことを行うにも、どのソフトを使用するべきか、特に初学者にとっては判断に困る状況も生まれている。
そこで、このテキストは、初学者でも、とにかく地球流体電脳倶楽部のいくつかのパッケージを使用し始めることができる様に、作成された。また、そういうパッケージやライブラリを開発したりメンテナンスする上で必要なソフトウェアを紹介し、地球流体電脳倶楽部の活動に参加したい気持ちを持つ方の敷居を少しでも下げられればと考えている。
また、この文章中の記述は 1 度は確かめてはあるが、誤字などに気づかれた場合や、記述通りに作業をしたのに思わしい結果が得られない場合には otobe@gfd-dennou.org までメールにてご連絡いただければ、可能な限り対処いたします。

# 第2章　まずは ubuntu のインストール

Ubuntu は、Debian をベースに初心者でも使いやすいことを目標に管理されている、Linux のディストリビューションの一つである。インストールもしやすく、パッケージも豊富で入門者が選択するのにふさわしい。
Ubuntu のインストールはさほど難しくはないが、インストール用の CD が必要である。このあたりは、将来的に加筆される可能性があるが今は、省略しておく。箇条書きにて手順を示す。

- Ubuntu のサイトからインストールイメージをダウンロードする。
  32bit でもいいが、今時の 64bitCPU の場合は 64bit の方がいいだろう。
- ダウンロードしたイメージを CD-ROM に焼く。焼く場合は他のコンピュータが必要である。
- 作成した CD-ROM から起動して、インストールを行う。いくつかの質問に答えるだけでインストールは終了する。

インストールが終了した後、ネットワークにだけは接続されている必要があるので、そこまですませたら、下の手順に移る。

# 第3章　最初に日本語環境

## 3.1　言語サポート

CD からインストールされた状態では、日本語を選んでいても、日本語環境を構築するのに必要なパッケージが不足しているので、インストールする必要がある。これは、最初にログインしたときにメッセージとして表示されるかもしれない。
もし表示されなくても、自分で設定するべきである。左上にあるアイコンからインストールされているプログラムを選び出すか、検索をすることができるので、そこから

```
Language Support
```

を見つけ出すか検索して、起動する。

```
Install -> Japanese
```

を選び、日本語をインストールする。システムで使用され言語は、クリックではなくドラッグアンドドロップして上位に持ってくることで、選択されることに注意。また、日本語変換は ibus にしておくと良い。

## 3.2　キーボードの設定

正しく日本語のキーボードに設定されていれば問題ないが、多くの場合英語配列のキーボードに設定される。英語配列のキーボードを使用していない場合は刻印と違うキーが入力されて困るだろう。その場合は以下の手順で、日本語のキーボードを使用する設定を行うと良い。
左に並んでいるアイコンの中に歯車の物があるが、これが設定を行うアプリケーション「システム設定」を起動するためのアイコンである。
その中から「キーボード」を選んで起動する。すると、ボタンなどになっていないので少し見つけにくいが下の方に「レイアウトの設定」という文字があってクリックできるのでクリックする。
窓がもう一つ開き、左側の領域に英語が一番上になっているはずである。その領域の左下あたりにある「＋」のボタンをクリックする。日本語を選んで追加を押す。
この状態では、上から英語・日本語の順で並んでおり、英語が優先される用になっているので、日本語を選んだ後「＾」(上向き△) を選んで日本語が一番上に表示されるようにする。そうすると、日本語のキーボードが最優先になる。

```
Install -> Japanese
```

## 3.3 gedit の設定

メニューではテキストエディタと表示されているが、gedit というのが標準のエディタのコマンド名である。これを、端末から gedit もしくはメニューからテキストエディタを選んで起動する。
このエディタは、インストール時のままでは欧文か UTF-8 以外の文字コードを扱えない。しかし TEX のファイルを編集するには EUC のファイルを編集する必要があるので扱えるようにしなくてはいけない。とはいえ、エンコーディングを追加するだけである。手順は以下の通り。

> 起動して、上部の「保存」ボタンをクリックして、ダイアログを出す。
> 左下にエンコーディングというのがあるのでその右の下向き三角を押す。
> するとインストールされているエンコーディングの一覧が右に追加できる
> エンコーディングが左にならぶので、「日本語 EUC-JP」を選ぶ。
> 追加ボタンを押して、右側に「日本語 EUC-JP」がならんでいるのを確認したら
> 「OK」ボタンをおす。

これで、gedit で EUC が使えるようになる。通常は自動判定で開いてくれるがうまくいかないときには開くボタンからダイアログを使い、エンコードを指定することもできるので、そのように対処する。

## 3.4 端末の文字コード

デフォルトは UTF-8 であるが、簡単に切り替えられる。端末がアクティブな状態で、画面最上部のバーにマウスカーソルを持っていくとメニューが表示される。その中の「端末」という中に「文字コード」というのがあるので、その中から選択できる。TeX の作業中は EUC-JP にすると良いが、エラーメッセージが化けてしまったりするので悩ましい状態である。必要に応じて切り替えながら使用する。debian の最新版では、TeX も UTF-8 になりつつあるので、次の LTS では、このような面倒は解消されるかもしれない。

# 第4章　sun の java

smartSVN や smartGit は Java で書かれているので、Java のランタイムが必要なのだが、Ubuntu の標準パッケージである OpenJDK のランタイムは互換性が低くいまのところ、smartSVN/Git を起動することができない。そこで、Oracle の提供している Java ランタイムをインストールする。
Ubuntu におけるソフトウェアのインストールは 12 章に書いたように apt によるのが一般的である。もちろん、伝統的な make によるインストールも使用できるが、特段の理由がない限り apt による方が簡単で便利である。oracle-java はレポジトリが存在するので apt でインストールする。

## 4.1　レポジトリの追加

```
sudo add-apt-repository ppa:webupd8team/java
sudo apt-get update
```

## 4.2　インストール

```
sudo apt-get install oracle-java7-installer
```

インストールの途中でライセンスに同意するように求められる。同意したくなければ、しなくても良いが、その場合はもちろんインストールはキャンセルされ Oracle の Java は使用することができない。

## 4.3　jre が２つはいっている場合

もし、Oracle の java と openjdk の java の両方のランタイムがインストールされていても問題は無い。これらを切り替えるやり方が、ubuntu には用意されている。

```
sudo update-alternatives --config java
```

とすることで、java についてどのパッケージを使用するかを切り替えることができる。このコマンドを実行し Oracle の java を使用するようにしておくと良い。もちろん openjdk の方を削除してしまっても差し支えない。いずれにしても使用できるのはどちらか一方であり java に関しては Oracle のもので問題を起こすことはないので、そちらを使用すれば良いだろう。

# 第5章　日本チームのレポジトリ

## 5.1　日本チームとは

Ubuntu は、基本的には元々多言語されており、いろんな言語環境で使用できるのだが、伝統的なパッケージで名前がバッティングしたりする場合がある。そこで、日本語独自の処置を施しているグループがある。それが Ubuntu 日本チームというグループである。このグループのレポジトリを追加するほうが、日本語環境では便利になる。
また、Ubuntu の公開からしばらくすると、日本語チームのパッケージ情報が追加されたインストーライメージが公開されるので、それを利用してインストールを行ってもかまわない。

## 5.2　レポジトリ

日本チームの Web ページに追加の方法ヶあるので、それを参考にすると良いがいちおう、以下にも記述しておく。

```
wget -q https://www.ubuntulinux.jp/ubuntu-ja-archive-keyring.gpg -O- | sudo
apt-key add -
wget -q https://www.ubuntulinux.jp/ubuntu-jp-ppa-keyring.gpg -O- | sudo apt-key
add -
次の行は、２行をつなげて入力すること。
sudo wget https://www.ubuntulinux.jp/sources.list.d/precise.list -O
/etc/apt/sources.list.d/ubuntu-ja.list
sudo apt-get update
```

これで、レポジトリが追加される。日本語の特に TEX 周りでは、こちらのを使う方が問題が少ない。

# 第6章　おすすめのパッケージ

デフォルトではインストールされないが以下のパッケージはおすすめ。

## 6.1　openssh-server

ノートパソコンでは、不要であるだろうが、デスクトップパソコンなら、リモートからログインできると便利であるので、インストールする。そうすれば他のマシンから ssh することができる様になる。

## 6.2　emacs

ノートパソコンでは、使わないということなら不要であるが、デスクトップパソコンなら、リモートからログインした際に、コンソール上で使用できるエディタがあった方が良い。一応、そのためのエディタは vi(vim) や nano がインストールされているが、標準的なエディタであろうと思うので、インストールしておくと良い。ただし、日本語変換ソフトを立ち上げるキーが SHIFT+SPACE にも割り振られているので、これは外しておいた方が良い。右上のキーボードアイコンから設定できる。HankakuZenkaku や ALT+HankakuZenkaku があれば十分である。

## 6.3　cron-apt

適当な間隔で `apt-get upgrade` をかけてくれる。自動で最新版になるので横着したい人にはお勧め。勝手にされるのが嫌な人はインストールしなくても良い。

### 6.3.1　手動での update

デスクトップ上にアップデートがある場合それを促すメッセージなどが表示されるのでそれに従えば良いが、手動でもマメにアップデートする方が望ましい。

```
sudo apt-get update
sudo apt-get dist-upgrade
```

## 6.4　apt-file

インストールしたいソフトを探すときには、apt-cache などを使うのだが、このコマンドはパッケージの名前が部分的にでも分かっているときや、どういうことをするパッケージかということが分かっているときにしか

使えない。
しかし、開発をしていたり、いろいろなソフトを試したりしているときにはある、ファイルが含まれているパッケージを見つけたいときがある。apt-file はすべてのパッケージから、特定のファイルを持っているパッケージを検索して教えてくれる。
インストールした後や、使用したい直前には

```
sudo apt-file update
```

として、データベースのアップデートが必要。

## 6.5 GhostView

PostScript を見たい場合もあるであろうから、gv をインストールしておくとい。

```
sudo apt-get install gv
```

## 6.6 Adobe Reader

PDF のビューワーとしては、evince などがインストールされているはずであるが、やはり、Adobe Reader を使用するのが使いやすさや表示品質の面でよろしい。そこで、これをインストールする。
現在、このパッケージはレポジトリには存在しないようである。

```
sudo apt-get install adobereader-jpn
```

## 6.7 Adobe Flash Plugin

これは、不要なら省略してもかまわないがビデオを見る際に必要とされることが多いので、一応インストールしておくとよい。
このパッケージも存在しないようである。

```
sudo apt-get install adobe-flashplugin
```

## 6.8 ビデオ関係

地球流体基礎実験集のビデオなどが見えない。そのため以下のパッケージをインストールしておくと良い。

```
sudo apt-get install gstreamer0.10-ffmpeg
```

## 6.9　日本語変換

日本語変換は Linux が、Windows に比べて著しく劣っているところであったが、Google の日本語変換を元にした mozc という日本語変換エンジンはとても優秀で、windows のものと比べてもさほど見劣りしない。出来ればインストールするのがおすすめ。

```
sudo apt-get install ibus-mozc
```

インストールしただけでは優先されていないので設定する必要がある。画面右上のキーボードのアイコンを左クリックして、設定を選択する。インストール直後にこの操作を行う場合は、Ibus がまだ MOZC を認識していないので設定の下にある「再起動」を選ぶこと。Ibus が再起動する間、キーボードアイコンが消えるので再び現れた後、この設定操作をもう一度行う。「インプットメソッド」のタブを選択「使用するインプットメソッドをカスタム」にチェックを入れ下のドロップダウンメニュー（インプットメソッドの選択）から日本語、MOZC と選択して追加ボタンを押す。MOZC を選んで「上へ」ボタンを押し一番うえにすると次から選択されるはずである。

# 第7章　電脳 Ruby 製品および、DCL

## 7.1　電脳 Ruby

電脳 Ruby は、可視化やデータハンドリングのためのライブラリおよびツール群で構成されている多層に渡るパッケージ群である。Ruby で使用されることを想定して整備されているが、その下位のパッケージは Fortran や C などのライブラリも多数ある。これらのパッケージは、単独でインストールすることもできるがまとめてインストールしておくのがおすすめである。
個別の、パッケージについての説明は電脳 Ruby のページなどから調べて欲しい。

## 7.2　レポジトリ

直接ファイルをいじることなく設定する。アップデートマネージャーを起動する。update-manager などを検索したり、アイコンを探し出せば良い。
「設定」の中の、「他のソフトウェア」というタグを選ぶ。追加ボタンを押して apt ラインに

```
deb http://ppa.launchpad.net/gfd-dennou/ppa/ubuntu precise main
```

と入力して「ソースを追加」する。この後は、ターミナル（端末）でいつものように

```
sudo apt-get update
```

を行った後、

```
sudo apt-get install gave
```

とすれば、電脳の主な製品はインストールされる。いくつかのパッケージは認証されていないというようなメッセージが表示されるかもしれないが、取りあえず Y を押して続ける。

# 第8章　smartGit(SVN)

## 8.1　バージョン管理

パッケージのレポジトリと同じ用語であるので、混乱しないで欲しい。
ソフトウェアの開発には、コンパイラなどの直接的な開発ツール以外にもバージョンを管理するバージョン管理ソフトというのが使用される。地球流体電脳倶楽部では、古くは CVS を使用していたが、最近は Subversion(SVN) を使用する場合も多い。ここでは、CVS のことは説明しない。CVS は、SVN の機能縮小版と思っていただいても、そう間違いではないぐらい似ているからで、可能ならば、CVS のレポジトリを SVN に変更するようにおすすめするからでもある。どうしても、CVS を使わねばならなければそのときに学んでいただければ良いと思う。SVN が分かっていれば、1 時間もかからずに理解できると思う。
Git というツールは、最近よく使われるようになった同様のツールで機能的には優れているが、学習コストが必要であり、多人数で使うことが前提なバージョン管理ソフトとしては、全員が乗り換えることができないので、今のところ地球流体電脳倶楽部では使用されていない。
ただし、最近はいくつかのパッケージでは使用され始めており、また、これ以後立ち上げられるパッケージでは Git を使うことが考えられるので、こちらをメインに使用するように推奨する。git-svn というパッケージを使用すれば Subversion のレポジトリも Git のコマンドで管理することが出来るからである。

## 8.2　クライアント

仕組みを理解していれば、コマンドラインツールの方が軽くて良いという意見もあるだろうが入門者がイメージしやすいということや、全体的な機能を概観しやすいということ、またツールの出力を確認しやすいということなどから、最初は GUI ツールをおすすめする。
いくつかの GUI ツールが存在するが、使いやすいと思われる物は smartGit/hg である。（ただし、コマンドライン版と同時にインストールした gitk というのも GUI のソフトでありこちらは公式のツールでもあるので、こちらの機能が充実してくるようならばおすすめである。）このツールには姉妹品として smartSVN というソフトも存在する。しかし、smartGit/hg があれば、おそらく不要であろう。
これらのソフトは、Java で記述されており、Windows や MacOS などでも動作する。ただし、Java のランタイムは今のところ Oracle の Java ランタイムが必要なようである。
Git は SVN のレポジトリを操作することができる git-svn というツールを含んでいる。smartGit も SVN のレポジトリを扱うことができる。このツールで扱えば SVN と Git のいいとこ取りが期待できる上に扱いもそう難しくはないのでこちらを使う物とする。

## 8.3　インストール

まず、前提としてコマンドライン版の git が必要となるのでインストールしておく。

```
sudo apt-get install git gitk git-svn
```

smartGit をダウンロードしてきて展開した後、パスを通すか、パスの通ったところにコピーすれば良い。検索すれば、すぐに見つかるだろうが、

```
http://www.syntevo.com/smartgit/index.html
```

からダウンロードできた。linux 版であることを確認した後、最後に場所を指定して保存をすれば良い。この作業は、作業している OS によって開かれるページが変わってしまうので、インストールするマシンでダウンロードするように注意する。
ダウンロードされたディレクトリで作業する。

```
tar zxvf smartgithg-generic-4_6_3.tar.gz
sudo mv smartgithg-4_6_3 /usr/local
cd /usr/local/bin
sudo ln -s ../smartgit-generic-4_6_3/bin/smartgithg.sh .
```

以上の方法は、バイナリを/usr/local においた後、パスの通ったディレクトリにシンボリックリンクを作成しているが方法が分かるならパスを通す方法でも良い。また、バージョンは執筆時点での物であるので、もう少し大きな数字になっているかもしれない。その場合は適宜読み替えていただきたい。
起動する方法は

```
smartgithg.sh &
```

で起動できる。

## 8.4　使い方

### 8.4.1　初期設定

ツール全体の使い方は、それなりの書籍などを参考にしていただきたいが、さしあたり、DCL と、そのマニュアルのレポジトリのコピーを作り編集する手順について、必要最低限の記述をしておく。

- 初回に起動した場合は、ライセンスについての同意を求められるので同意する。チェックを入れて Next をクリックする。

- Non Commercial Use Only にチェックをして Next をクリック。サポートや少し機能が制限されるが通常はこちらで十分である。そのような物が必要になれば有料版を購入すると良い。

- Git Executable に /usr/bin/git と入力（元々入っているはず）して Next をクリックする。

- Use SmartGit as SSH Client を選んであることを確認し Next をクリック
- User Name（アルファベットでフルネーム） と e-mail アドレスを入力。
- I don't use a hosting provider を選んで Next をクリック
- Finish をクリック

以上で、初期設定は終了

### 8.4.2　レポジトリのクローン

以下の説明では地球流体電脳倶楽部の DCL を開発することを例に説明している。アクセス権のあるレポジトリがあればそれと読み替えていただければいい。もしくは、自分のレポジトリを作成してから同じようにする。いずれにしても git の使い方については書籍やネット上の資料などを参考にする。電脳倶楽部でも git の手引を置いているのでそちらを参考にしていただければ幸いである。
Subversion では Checkout と呼んでいるが、Git ではレポジトリのクローンと呼ぶ。厳密には両者は異なるが、この作業ではレポジトリのコピーをローカルに作成する。両者の違いは、Git は本当にコピーであるので、ローカルの物もレポジトリであって、すべての過去の作業がデータベース化されているが、Subversion のチェックアウトはファイルのコピーと、少しの情報である。そのため、Subversion は過去の履歴を確認する際にはレポジトリに接続することが要求されるが、Git は、すべてを持ってきているので、手許で何でも行うことができる。
smartGit では、SVN のレポジトリを扱うときにもローカルにレポジトリを作成するのですべての情報を持っている。なので、ローカルでは Git のレポジトリであると思うことができる。
レポジトリのクローンのやり方は以下の通り、最初に、ローカルにレポジトリを置く場所を用意する。どこに作っても良いが、端末を開いて、

```
mkdir -p  /work/gitwork/dcl
```

などとする。そのあと、レポジトリの場所とそのコピーを置く場所を指定する。smartGit のメニューから Project -¿ clone と選び

- Repository は Remote Git or SVN repository に
  `svn+ssh://dennou-k.gfd-dennou.org/repositories/dcl/svnroot/dcl`
  と入力して、Next をクリックする。ただし、このアドレスは適当に書いた物であるのでこのディレクトリは実際には存在していないことに注意。
- 認証方式は Password を選択して、dennou-k のパスワードを入力する。
- Git Working Tree はローカルのディレクトリ
  `/home/<username>/work/gitwork/dcl`
  のように先ほど作ったディレクトリ名を入力
- Open in Project で、何でもいいのでプロジェクトネームを入れる。

わかりやすい方がいいので DCL などとする。

これで、レポジトリのクローンが始まるはずである。
DCL のマニュアルのレポジトリは、

```
svn+ssh://dennou-k.gfd-dennou.org/repositories/dcl/svnroot/Manual
```

であるので、同様にして、クローンを作ると良い。これらは、レポジトリの丸ごとをコピーするのでそれなりの時間がかかる。大きなプロジェクトになればなるほど時間がかかるが 1 回目だけなので我慢する。

### 8.4.3　作業の流れ

ファイルの変更は、普通に使い慣れているエディタなどで行えば良い。開き方などにも特段の注意はない。取り出してきたディレクトリでそのまま作業して差し支えないし、他のところで作業しても良いが、特に意味が無ければここで作業するのが良い。
適度に変更したら、Commit を行う。Commit は変更点をレポジトリに登録する作業のことである。SVN ではレポジトリは中央に一つであったが、Git では登録されるレポジトリはローカルにクローンされたレポジトリとなる。コミットする際にはメッセージを記録できる。変更点および、何故その変更を行ったかが分かるような簡潔なメッセージが望ましい。
SVN では、不要であったが、Git クライアントの場合、中央のレポジトリにこの変更をさらに登録しなければいけない。そのコマンドが Push である。Commit から連続で Push までする場合というのは SVN と Git で全く同じことになるが、Git ではいくつかの Commit をまとめて Push できる。そのため、どうでもいいお試しの Commit がやりやすく中央のレポジトリが散らかりにくいという利点がある。いずれにしても、最終的には Push をして、開発の小サイクルは終了である。
多人数で作業をしている場合、自分以外の変更点を取り込んで置く必要がある。SVN で言えばこれは Update というコマンドで行う。Git では Pull である。作業を始める前などに、忘れないように Pull しておくのが後々のトラブルを避けるのに有用である。おなじ所を編集してしまったりすることを避けられる。

# 第9章　DCL

## 9.1　パッケージのダウンロード

パッケージからインストールすることもできるが、執筆時点では Fortran のパッケージは gave をインストールしただけでは、インストールされないようである。おそらく、パッケージが要求されていない物と思われる、これはパッケージのメンテナに連絡してあるが、必要な場合は個々にインストールをしていただきたい。
また、このドキュメントは DCL の開発入門であるので、パッケージされていない開発版の DCL を使用できないと困る人向けである。そこで、パッケージからインストールできるようになったとしても、DCL の最新版をソースからビルドする手順を説明する。
この手順は非常に一般的で、ほとんどのソースコードは同様の手順でインストールできる。
ソースパッケージは地球流体電脳倶楽部のサイトから入手できる。執筆時の最新版は 5.4.8 である。いずれはそれ以上のバージョンナンバーになっているかもしれない。いずれにせよ、番号の大きい物が新しいという原則であるので、それに従って最も新しい物をダウンロードして保存する。

## 9.2　インストール

まず、必要なパッケージをインストールする。

```
sudo apt-get install tcsh gfortran libgtk2.0-dev gtk2-engines-pixbuf
```

libgtk には Ver.3 の libgtk-3-dev もあるが、今のところこちらでは DCL は動作しないので、2.0 の方をインストールする。
ダウンロードされたソースは圧縮されているので、解凍する。

```
tar zxvf dcl-5.3.4.3.tar.gz
```

その後、解凍されたディレクトリの中に移動し、

```
cd dcl
export FC=gfortran
./configure
make
```

ここまでで、特にエラーメッセージが表示されなければ

```
sudo make install
```

とすれば、使用できるはずである。dclfrt と入力してファイルがないという以外のエラーが出なければ成功である。もちろん、dclfrt が無いという意味でのエラーの場合はうまくできていないので、もう一度確かめながら作業を行う。

## 9.3 その他の注意点

gfortran や gcc でコンパイルする場合いくつかのデモで Segmentation Fault が発生する場合がある。これは、配列が大きすぎてスタックに収まらないことが原因であるので、

```
ulimit -s unlimited
```

として、スタックの制限を無くしておくと良い。

# 第10章　DCL-f90

## 10.1　ダウンロード

基本的な条件として DCL を要求するので、DCL のインストールは済ませておくこと。
地球流体電脳倶楽部の f90 インターフェイスパッケージから `dcl-f90.tar.gz` をダウンロードする。

## 10.2　インストール

nkf が必要であるので、まずインストールする。

```
sudo apt-get install nkf
```

パッケージを解凍する

```
tar zxvf dcl-f90.tar.gz
```

解凍されたディレクトリの中に移動し、

```
./configure
make
sudo make install
```

とすれば、無事インストールされる。

# 第11章　TEX

## 11.1　TEX のパッケージ

今まで同様にレポジトリからインストールする。大量のパッケージが存在するがある程度依存性でインストールされるので、

```
sudo apt-get install okumura-clsfiles texlive-latex-extra xdvik-ja
```

でインストールできる。

## 11.2　Dennnou6.STY

TEX には、スタイルファイルというテンプレートのような物がある。地球流体電脳倶楽部ではマニュアルを作成するために Dennnou.STY というスタイルファイルを公開している。現在の最新版は 6.2 である。
このファイルは、以前は DCL と一緒に公開していたが、現在は分かれて個別に公開されているので、これをインストールする。
まずは、最新版のパッケージをダウンロードする。

```
http://www.gfd-dennou.org/arch/cc-env/TeXmacro/dennou/SIGEN.htm
```

を開いて最新版である dennou-sty-6-current.tar.gz をダウンロードする。これを展開するのは、DCL と同じく tar である。

```
tar zxvf dennou-sty-6-current.tar.gz
```

そのディレクトリの中に入って make をするのだが、その前に、DOC ディレクトリの中にある Makefile を修正する。適当なエディタで、開いたら、上の方にある

`/usr/doc/dennou-sty-6`　を
`/usr/share/doc/dennou-sty-6` に書き換える。

そのあと、DOC の一つ上のディレクトリで make とすればインストールされるはずである。エラーメッセージが表示されていないことを確認すること。

## 11.3　フォントの埋め込み

ここまで行えば、DCL-Manual もコンパイルして pdf ファイル化することができる。
しかし、pdf 化する場合には、フォントも埋め込んでおいた方が、後々様々な環境で利用する上で便利である。

埋め込むフォントはライセンスをクリアしている物であれば何でも良い。ここでは、Ubuntu の標準日本語フォントである Takao フォントを埋め込むことにする。このフォントは IPA フォントの元になったフォントの作者から名前を取った物で、IPA フォントをそのまま使用した際に生ずる不具合を独自に修正したフォントである。
pdf の作成は dvipdfmx というコマンドで行うがこの変換中にフォントを埋め込む。どのようなフォントを埋めるかは/etc/texmf/dvipdfm/jis-cjk.map に書いてある。デフォルトのままではフォントは埋め込まず、Ryumin や中ゴという、モリサワのフォントで表示するよう指定するだけである。このままだと、そのようなフォントのある環境ではそのフォントで、そうでない場合は適当な代替フォントで表示される。

### 11.3.1　フォントファイルをインストール

まず、TEX が認識できるディレクトリにフォントをシンボリックリンクする。

```
sudo mkdir -p /etc/texmf/fonts/truetype
cd /etc/texmf/fonts/truetype
sudo ln -s /usr/share/fonts/truetype/takao takao
sudo mktexlsr
```

最後の行は TEX のキャッシュに登録するようなものだと思っていただければ良い。ここで、

```
kpsewhich TakaoMincho.ttf
```

で、フォントが見つかるかを確認する。

### 11.3.2　埋め込むフォントの設定

/etc/texmf/dvipdfm/jis-cjk.map を適当なエディタで開き編集する。

```
rml-jis H Ryumin-Light
gbm-jis H GothicBBB-Medium
```

の 2 行をコメントにする (先頭に % を追加する) か削除する。その下に次の行を追加する。

```
rml-jis H TakaoMincho.ttf
gbm-jis H TakaoGothic.ttf
```

また、このフォントの map ファイルが読み込まれるようにするために、/etc/texmf/dvipdfmx/dvipdfmx.cfg を適当なエディタで開きそのファイルの一番最後の行に

```
f cis-cjk.map
```

という行を追加する。

### 11.3.3　グリフファイルの追加

pdfglyphlist.txt と glyphlist.txt という 2 つのファイルが必要であるが、このファイルは Ubuntu のパッケージ群には含まれていない。そこで、この 2 つのファイルについては Windows 用のパッケージから取

り出してきて使用することにする。
まず、ファイルのダウンロードである。

```
wget http://www.ring.gr.jp/pub/text/TeX/ptex-win32/current/dvipdfm-w32.tar.xz
```

とする。
次に解凍して、ファイルをコピーする。

```
tar -Jxvf dvipdfm-w32.tar.xz
cd ./share/texmf/fonts/map/agl/
sudo cp -v pdfglyphlist.txt glyphlist.txt /etc/texmf/dvipdfmx
```

最後に texmf.cnf を編集する。418 行目ぐらいにある CMAPFONTS の行を

```
CMAPFONTS=.;$TEXMF/fonts/cmap//;/usr/share/fonts/cmap//
```

に書き直す。その後、

```
sudo update-texmf
```

として、設定を反映させる。
以上で、dvipdfmx によってフォントが埋め込まれるようになる。

# 第12章　apt によるインストール

Ubuntu のパッケージは、Debian と同じく、サーバーに集積して公開されている。このようなパッケージ置き場をレポジトリと呼んでいる。

## 12.1　レポジトリ

ソフトウェアを使用するにはまず、自分のマシンにレポジトリを登録し、レポジトリからソフトウェアを入手する。レポジトリの登録は/etc/apt/source.list に書くことで行うが、最近はレポジトリごとに/etc/apt/source.list.d にファイルを作るなどする場合もある。このあたりは、どちらでもいいが、レポジトリの Web ページなどに登録方法があるので参考にすると良い。

## 12.2　ソフトウェアリストの更新

レポジトリに登録されている、ソフトウェアのリストを取得して、ローカルの情報を更新するには

```
sudo apt-get update
```

とする。登録されているすべてのレポジトリから情報を取ってくる。

## 12.3　ソフトウェアのインストール

たとえば、emacs をインストールしたいとするなら、

```
sudo apt-get install emacs
```

のように、apt-get にオプションを与える。もし、そのパッケージが他のパッケージを必要としていても、そのようなパッケージも同時にインストールされる。

## 12.4　ソフトウェアのアンインストール

たとえば、emacs をアンインストールするなら、

```
sudo apt-get remove emacs
```

この場合は、emacs と同時にインストールされたパッケージはアンインストールされない。ただし、依存性に従ってインストールされたパッケージが他のパッケージから使用されていない場合は autoremove することができる旨メッセージが表示される。

```
sudo apt-get autoremove
```

とすれば、そのようなもはや使用されていないパッケージをシステムから削除することができる。

## 12.5　ソフトウェアのアップデート

インストールされたパッケージに、バージョンアップされたバージョンが存在する場合

```
sudo apt-get upgrade
```

とすることで、最新のバージョンが上書きインストールされる。定期的にアップデートしておくと良い。このコマンドでは、システム本体はアップデートされない、kernel のようなシステムそのものをアップデートしたい場合には

```
sudo apt-get dist-upgrade
```

とすれば良い。この場合には、アップグレードを有効にするにはシステムのリブートが必要である。