MacOS X WorkShop
— apt-rpm system on MacOS X —
10.5-3
for Intel/PowerPC

KOBAYASHI Taizo

2010/08/09

CarbonEmacs パッケージを入れ、TeX のパッケージを入れ、、、
ghostscript を入れ、gnuplot を入れ、、、、
ドットファイル群を設定して、、、、
でも、LaTeXiT が動かなかったりする。。。。

web 上の掲示板でしばしば目にする光景です。

みんな殆ど同じ事をするのに、
ひとり一人が、或は一台一台大変な作業を繰り返すのは
開発者か趣味でもない限り大いなる無駄！！

だと思いませんか？

MacOS X で LaTeX や emacs 等の環境を整えられてきた
先人達の成果を集積したものと、
Vine Linux の行き届いた環境を融合させたもの、
それが MacOS X WorkShop です。

MacOS X WorkShop を利用すれば、
定評のある Vine Linux の TeX 環境が一発で構築でき、
すぐさま仕事に入れます。

ただし、このプロジェクトの成果物を利用して不具合が生じても、
プロジェクトとしてもプロジェクトに関係する如何なる人間も
一切責任を負わないものとします。
また、バグ報告やパッケージングの要望は歓迎しますが
迅速な対応は期待しないでください。
と云うよりも、要望をお持ちでしたら、

是非、要望を実現した姉妹 apt-rpm tree を作ってください！！

最後に、
我々は企業のサポート窓口ではありません！
こちらで不具合を再現出来る程度の情報がバグ報告に無い限り、
返事も対応も無いと考えてください。

目 次

図 目 次

1　更新情報

1.1　10.5-3 の変更点

- gnuplot の PPC 版のバグを修正
  吉田さん、ありがとう御座いました。
- a2ps, a2pdf でサフィックスの扱いを修正。
- REVTeX4.1 に更新。

1.2　10.5-2 の変更点

- apt.conf に http proxy の設定方法を追加
  出村さん、ありがとう御座いました。

2　はじめに

　このプロジェクトの目的は、
TeX や emacs を用いて仕事をしている人が、
MacOS X 上にストレス無く即座に仕事に掛かれる環境を構築する事
と
大学や研究機関等の計算機管理者が、
MacOS X 上に独自の研究環境を簡単に築き管理する事
にあります。
　念頭に置いている TeX, emacs 環境は大学で支持を得ている **Vine Linux**[1] です。
この Vine Linux の中で日々の仕事に必要なパッケージを MacOS X に合わせて構築し直した物と、
MacOS X 上の便利なソフトを組み合わせたものが MacOS X WorkShop の実態です。
現在公開しているパッケージは
MaxOS X 10.5.x (Leopard) 対応版と **MaxOS X 10.4.x (Tiger) 対応版**[2] と **MaxOS X 10.3.x (Panther) 対応版**[3] です。
尚、今後 Tiger, Panther 版の拡張は致しません。
パッケージに関する詳細情報は rpm2html による **RPM 解説データベース (Leopard)**[4] **RPM 解説データベース (Tiger)**[5] **RPM 解説データベース (Panther)**[6] をご利用ください。

　MacOS X WorkShop 固有の拡張を施してあるパッケージの内容に関しては
「パッケージメモ (Section10 参照) 」をご覧ください。

2.1　何故 apt-rpm か？

　パッケージの作成、管理、利用の全てで
楽ができるから です。

　例えば、TeX の環境を構築したいのであれば、
ターミナルを開いて

```
$ sudo apt-get update
$ sudo apt-get install OSX-base
$ sudo apt-get install task-tetex
$ sudo apt-get clean
```

とするだけで TeX 関連のパッケージをまとめてインストールし、
且つ、各ユーザーのドットファイル群を含む面倒な各種設定まで

[1] http://www.vinelinux.org/
[2] ../Tiger/index.html
[3] ../Panther/index.html
[4] ../Leopard/rpm2html/
[5] ../Tiger/rpm2html/
[6] ../Panther/rpm2html/

自動で片付けてくれます。

パッケージの更新はバグが見つかる度に成されますが、その場合でも、

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

でおしまいです。
いちいちインストールし直す必要は無いのです。

この様な楽ができるのは apt-rpm system に先人達の成果を集積しているからです。
ソース・パッケージ (hoge-ver-rel.src.rpm) には
ソースだけでなく、パッチやコンパイル、インストールの仕方まで
こと細かに書かれています。
つまり web 上に散らばった情報を集積している訳です。
OSXWS をインストールして パッケージを開発する (Section6 参照) してみれば
貴方が何時間も、時には何日も掛けて探しまわった情報と作業が
たった一つの src.rpm ファイルに凝縮されている事に気づく筈です。

どうですか？
かなり楽が出来そう
ではありませんか？

MacOS X 上での UNIX 研究環境を構築するには **Fink**[7] をはじめ、
琉球大学の **EasyPackage**[8] や、**DarwinPorts**[9] 等があり、
各々がそれぞれのパッケージングシステムを持っています。
他にもパッケージングシステムを持たない総合情報として **Mac Wiki**[10] が在り、
自分が必要とするソフトを手で一つ一つ入れる事も出来ます。

当然の事ですが、それぞれに利点と欠点があります。
Fink の利点は何と言ってもパッケージの多さでしょう。
その反面大きなプロジェクトである為に、
我々の些細な（しかし研究上無視出来ない）変更を施すのは余分な労力を要します。

EasyPackage の利点は琉球大学で既に利用されていて、
且つ、ある程度活発なコミュニティーが存在していることだと思います。

[7] http://fink.sourceforge.net/
[8] http://www.ie.u-ryukyu.ac.jp/darwin2/
[9] http://darwinports.opendarwin.org/
[10] http://macwiki.sourceforge.jp/cgi-bin/wiki.cgi

しかし rpm の様に枯れた技術とは言い難く、
Linux の使用歴が長い研究者にとっては多分に不安を覚えるのも事実です。

DarwinPorts は Apple に最も近い存在ですが、
既にかなり大きなプロジェクトになっている事と、
小林が FreeBSD を殆ど知らない事から対象外になっています。

apt-rpm を用いる副次的な利点としては、同じシステムを用いている
Vine Linux 等 Linux の成果を活かし易い 事があげられます。
以下、システムの簡単な概要を説明します。

rpm[11] は **Red Hat**[12] が Linux distribution のパッケージングシステムとして開発したものです。
rpm, rpmbuild 等のコマンドを通して、パッケージの

- 作成
- インストール
- 更新
- 除去

を行います。パッケージ間の依存関係の情報をパッケージ自身が持っているので、
必要なライブラリを抜かしてインストールする様なミスを防ぐ事が出来ます。
Vine Linux[13] 等、多くの Linux Distributer がこのパッケージングシステムを採用しており、
MacOS X WorkShop に移植する際にそれらを拝借出来ます。
また、ソフトベンダーが Linux 向けの製品を提供する場合は殆 rpm 形式が使われており、
Linux での標準的なパッケージングシステムになっています。

apt[14] は **Debian GNU/Linux**[15] が Linux distribution のパッケージ管理ユーティリティとして開発したものです。
Debian の特徴は rpm ではなく独自のパッケージシステムを利用している事と、
8000 以上の膨大なパッケージ数を抱えている事です。
パッケージングシステムは兎も角、
この様な膨大なパッケージを利用する為には何らかの強力なパッケージ管理ユーティリティが必要です。
apt はその目的を果たす為に開発されています。
apt-get, apt-cache 等のコマンドを通して、

- 現在利用可能なパッケージの情報を得る。
- 更新されたパッケージを自動でアップデートする。

[11] http://www.rpm.org/
[12] http://www.redhat.com/
[13] http://www.vinelinux.org/
[14] http://www.debian.org/doc/manuals/apt-howto/
[15] http://www.jp.debian.org/

- パッケージ間の依存関係を自動で調整して適切なインストールと削除をしてくれる。

等、一度利用したら手放せなくなる機能を提供してくれます。

apt-rpm[16] は **Conectiva Linux**[17] が Linux distribution のパッケージ管理ユーティリティとして Debian の apt を rpm に対応させたものです。

MacOS X WorkShop では Conectiva の apt-rpm を Vine Linux が日本語対応にした物を流用しています。

rpm や apt の利用方法は「MacOS X WorkShop の使い方 (Section5 参照) 」を御覧下さい。

2.2 プロジェクトのポリシー

この MacOS X WorkShop プロジェクトには、以下のポリシーがあります。

- 管理に手間を掛けない。
- パッケージ数は必要十分に留める。
- 自分たちに都合の良い設定やパッチを用いる。
- それぞれの大学や研究室での派生プロジェクトを立ち上げやすくする。

です。

管理者がたった一人でも MacBook と一日の時間さえあれば、
一通り全パッケージのメンテナンスが出来るくらいの
小さなディストリビューションに出来るだけ留めます。

結局のところ如何に手間ひまを掛けずに必要十分な事を好き勝手にするかが本音です。
煩わしい計算機管理は出来るだけ楽に済まし、自分の本分にリソースを集中する環境を作るのが、
MacOS X WorkShop の目的でありポリシーでもあります。

2.3 派生プロジェクトの歓迎

プロジェクトの目的の一つとして、
立命館大学物理学教室で立ち上がったこのプロジェクトをひな形にした
姉妹 apt-rpm tree が作られる事を歓迎します。
各大学や研究室で独自の拡張や変更を施した姉妹 apt-rpm tree を是非作ってください。

[16] https://moin.conectiva.com.br/AptRpm
[17] https://moin.conectiva.com.br/

インストーラの作り方は「インストーラを作る (Section7 参照) 」を、
apt-rpm tree の作り方は「apt-rpm tree を作る (Section8 参照) 」を、
それぞれ御覧下さい。
貴方がたに必要なパッケージだけを集めた add-on tree も全く同様に作成可能です。
まずは、この MacOS X WorkShop を母体にした貴方独自の　apt-rpm add-on tree を
local disk に作る事から始められる事をお薦めします。
もしも、姉妹 apt-rpm tree を作られ { る, た } 際には是非ご一報ください。
姉妹 trees !! (Section3 参照) のページで紹介するとともに、
MacOS X WorkShop の apt-line に追加します。
そしてお互いに楽をし合いましょう。

2.4　連絡先とメンバー

MacOS X WorkShop に関する議論や連絡は **Mac Wiki**[18] を利用させて戴いています。
Mac Wiki で議論すれば、情報が蓄積されていき多くの人にとって有益です。

この web page が更新されるまでの変更は Mac Wiki でアナウンスしますので
出来るだけチェックするようにしてください。
また、バグ報告は基本的に OSXWS 標準の環境に対してのものにしてください。
勿論、.emacs.my.el 等を改変して独自の拡張を施すのは一向に構いませんが、
その結果現れた不具合の場合は必ず OSXWS に問題がある事を特定してから報告してください。
また、問題が解決した場合にも必ず報告して、言いっ放しにはしないでください。

現在のメンバーです。(順不同)

小林泰三 九州大学情報基盤研究開発センター、学術研究員

瀬戸亮平 ミュンヘン工科大学 PD

坂田泰啓 ニコン株式会社 ('03 池田研マスター卒業生)

新山友暁 立命館大学物理学教室池田研究室 PD

大関 努 立命館大学物理学教室倉辻研究室 '06 卒業

2.5　ライセンス

収録しているパッケージのライセンスは、
パッケージに収録しているソフトウェアのライセンスに従います。
`$ rpm -qi hoge` でパッケージ hoge のライセンスを確認出来ます。
また `/usr/osxws/share/doc/hoge` 以下にもライセンスに関するファイルが在ります。

インストーラのライセンスは GPLv2 以降に従うものとします。
インストーラに同梱されている ReadMe.rtf, License.rtf を参照して下さい。

[18] http://macwiki.sourceforge.jp/cgi-bin/wiki.cgi

3　姉妹 trees !!

繰り返しになりますが MacOS X WorkShop の目的は、
TeX や emacs を用いて仕事をしている人が、
MacOS X 上にストレス無く即座に仕事に掛かれる環境を構築する事
と
大学や研究機関等の計算機管理者が、
MacOS X 上に独自の研究環境を簡単に築き管理する事
です。

一口に「楽をする」と云っても計算機環境に求められるものも好みも千差万別です。
その様な状況で管理者とユーザーの双方が楽をする為には、
それぞれの環境に合わせた apt-rpm tree を構築するのが一等です。
apt-rpm tree を零から新たに作るのは結構な作業に成りますが、
この MacOS X WorkShop をひな形にすれば、数日で実現可能です。
例えば、
デフォルトのログイン環境や .emacs.el を変えたければ、
OSX-Preferences パッケージを弄るだけで済みますし、
emacs に lisp file を加えたければ、
emacs-lisps パッケージに加えればおしまいです。

このページでは、姉妹 apt-rpm trees の紹介をします。
全て、MacOS X WorkShop の apt-line (/priveta/etc/apt/sources.list 内に記述) に加えてあります。
貴方の求めているものに最も近い tree をご利用ください。

- **HEPonX**[19]
  KEK の藤井恵介さんが高エネルギー物理学の計算機環境を MacOSX 上に実現する為に作られた apt-rpm tree です。

  藤井さんは、PPC Linux の黎明期から Mac 上の Linux 環境の整備に貢献してこられ、MacOSX 上に rpm を最初に 移植した方です。MacOSX WorkShop (OSXWS) の rpm の基本部分は藤井さんの成果を利用しています。

- **MacOS X WorkShop**[20]
  立命館大学物理学教室で立ち上げられ利用されています。

[19] http://www-jlc.kek.jp/ fujiik/macosx/10.5.X/HEPonX/
[20] http://www.bach-phys.ritsumei.ac.jp/OSXWS/

4 MacOS X WorkShop をインストールする

このセクションでは MacOS X WorkShop をインストールする手順について説明します。
尚、以前の MacOS X に関する情報は **MacOS X 10.4 (Tiger)**[21] **MacOS X 10.3 (Panther)**[22] をご覧ください。

4.1 インストールする前に…

警告！

MacOS X WorkShop のインストール環境は、素の MacOS X に下記のパッケージをインストールした状況を想定しています。Fink との共存は可能かもしれませんがプロジェクトとしては考慮していません。また、他の方々が配布されている CarbonEmacs, ptetex 等がインストールされている場合、予期しない結果になる可能性があります。

また、Tiger などからの upgrade した環境も X11 の lib の更新にバグがあり gtk が動かない等の不具合があるためお勧めしません。

MacOS X WorkShop は MacOS X 上で emacs などの UNIX ツールを利用する為の環境を構築するものです。
従って、以下の MacOS X のインストール条件を満たす必要があります。

- `X11`
  mlterm, gv, gnuplot, xgraph, yaplot.... 等の X11 のソフトを利用するのに必要です。
  `/usr/X11/bin/Xquartz` がインストールされていれば大丈夫です。

- `Xcode`
  ここ[23] から最新版を入手してください。
  Apple が提供している開発環境です。インストールする時に X11 開発環境を必ず選択してください。

4.2 ターミナルの設定

次に apt-rpm を操作するターミナルの設定をします。

1. 先ず「アプリケーション」→「ユーティリティ」の中にある『ターミナル』をダブルクリックして起動します。

2. メニューバーの「ターミナル」から「環境設定...」を選択し、上部にあるアイコンの『設定』を選択します。

[21] ../Tiger/index.html
[22] ../Panther/index.html
[23] http://developer.apple.com/tools/xcode/

3. 右上に並んだ項目から「詳細」と表示されている選択項目をプレスします。

4. 「非 ASCII 文字をエスケープする」のチェックを外します。

4.3 Install

MacOS X WorkShop を始めるには
MacOS X WorkShop start kit MacOSX-WS-10.5.2.dmg[24]
をダウンロードしてインストールします。
(ソース一式は **MacOSX-WS-10.5.2.tar.bz2**[25] として置いておきます。)

注意！

このインストーラには必要最低限のバイナリしか含まれていません。
必ず「MacOS X WorkShop の使い方 (Section5 参照) 」を参照してインストールを完結してから
必要なパッケージをインストールして下さい。

尚、このインストーラは以下の処理を内部で自動で行います。

1. `apt-rpm` のインストール
   apt-rpm を利用する為の核となるものです。
   apt, popt, rpm, beecrypt, neon, expat package の中から必要な物を抜き出したものです。

2. `rpm data base` の構築

```
$ sudo rpm --initdb
```

   を実行します。

3. `OSX-system, OSX-X11` パッケージのインストール
   MacOS X に存在するリソースを rpm に知らせるパッケージをインストールします。
   `/usr/osxws/etc/{csh.login-osxws,profile-osxws,zprofile}` が加えられ、`/usr/osxws/bin` 等にパスを通します。

   尚、オリジナルファイルは.rpmorig のサフィックスを付けて保存されます。

   インストールされる設定ファイルは **OSX-system**[26] にてご確認ください。

[24] MacOSX-WS-10.5.2.dmg
[25] MacOSX-WS-10.5.2.tar.bz2
[26] OSX-system/

4. ユーザー用初期設定ファイル (dot files) のインストールと配布

OSX-Preferences package をインストールします。
また、各ユーザーに以下の設定ファイルを配布します。
既に存在する時はファイル名の末尾を .rpmold に変えて保存した上で配布されます。

- .Xclients
  X11 を起動する時に実行されるスクリプトです。
  mlterm や kinput2, login window をここで起動しています。
- .Xresources
  主に xterm の設定が記述されています。
- .bashrc, .bash_profile
  bash の設定ファイルです。
  OSX-Preferences package の更新に対応する為に個人用の記述は
  .bashmyrc の中に記述して下さい。
- .cshrc, .tcshrc
  csh, tcsh の設定ファイルです。
  .tcshrc は .cshrc へのシンボリックリンクです。
  OSX-Preferences package の更新に対応する為に個人用の記述は
  .cshmyrc の中に記述して下さい。
- .zshenv, .zshrc
  zsh の設定ファイルです。
  OSX-Preferences package の更新に対応する為に個人用の記述は
  .zshmyrc の中に記述して下さい。
- .custom_osxws.el
  emacs の設定ファイルです。
- .emacs
  emacs の設定ファイルで、osxws 提供の emacs とそれ以外の emacs を利用する際のきり分けをします。
- .emacs.d
  emacs で利用するディレクトリです。
  OSX-Preferences package の更新に対応する為に個人用の記述は
  .custom_osxws.el の中に記述して下さい。
- .emacs_osxws.el
  emacs の設定ファイルで、osxws 固有の設定を読み込みます。
- .inputrc
  ターミナル上で日本語をシームレスに扱う為の設定が書かれています。
- .mew.el
  emacs 上で mew を利用するにはこのファイルを各自編集します。

- `.yatex.el`
  yatex の設定ファイルです。
- `.vimrc`
  vi の設定ファイルです。
- `.rpmmacros`
  rpm package を構築する時は、
  予め各自このファイルを編集しておく必要があります。
- `.signature`
  メールの署名ファイルです。
- `.xinitrc`
  .Xclients を有効にします。

- `rpm`
  rpm package を構築する時の作業ディレクトリです。

インストールされる設定ファイルは **OSX-Preferences-10.5.tar.bz2**[27] をダウンロードしてご確認ください。

また、OSXWS デフォルトの設定ファイル群は
`/usr/osxws/share/OSXWS/jp/`
以下に有りますので、
local file の編集に失敗した時など必要な時にコピーしてお使いください。

注意！
各ドットファイルはピリオドから始まるため、Finder から直接見る事は出来ません。
展開後に terminal 上で cat コマンド等を利用して確認して下さい。

4.4 Remote Install

MacOS X 標準の installer コマンドを用いて
リモートでインストールする場合には以下の手順を踏んでください。

注意！
w コマンドなどを用いてユーザーが作業していない事を確認してから行ってください。

1. **イメージをマウント**
   インストーラが入ったディスクイメージ[28] をマウントします。

```
$ hdid MacOSX-WS-10.5.2.dmg
```

[27] OSX-Preferences-10.5.tar.bz2
[28] MacOSX-WS-10.5.2.dmg

2. インストール
   installer コマンドを用いてインストールします。

```
$ sudo /usr/sbin/installer -pkg /Volumes/MacOSX-WS-10.5.2/MacOSX\ WorkShop\ start\ kit.pkg -tar
$ hdiutil eject /Volumes/MacOSX-WS-10.5.2
```

3. 再起動
   再起動します。

```
$ sudo reboot
```

4.5 10.3, 10.4 版からの Upgrade

警告！

プロジェクトとしては新規インストールを推奨します。

Panther や Tiger から Leopard へ更新すると X11 のライブラリの更新がうまく行かないバグがあり、gtk 関連が動かなくなります。

4.6 Uninstall

警告！

MacOS X WorkShop のみをインストールしている状況を想定しています。
/usr/osxws 以下にファイルを置いている場合は該当するファイルをバックアップしておいてください。

アンインストールは簡単です。
以下のコマンドを実行し指示に従えば、
システムと各ユーザーの環境を素の状態に戻すことができます。

```
$ sudo apt-get remove OSX-system
```

5 MacOS X WorkShop の使い方

MacOS X start kit のインストールが無事済んだならば、
後は、自分が利用するパッケージを apt で入れるだけでおしまいです。

お使いの計算機が firewall の内側である場合に限り、
.bashmyrc と /usr/osxws/etc/apt/apt.conf を編集して環境変数 http_proxy や ftp_proxy を適宜設定
しておく必要が在ります。
該当箇所がコメントアウトされていますので、
お使いの環境に合わせて記述してください。

start kit をインストールした後に先ずすべき事は
OSX-base パッケージを apt でインストールする事です。

MacOS X WorkShop で必須の根幹パッケージをインストールしてくれます。
計算機がインターネットに接続されている事を確認してターミナルから

```
$ sudo apt-get update
$ sudo apt-get install OSX-base
$ sudo apt-get clean
```

を実行してください。
これが済んだら基本的に後は自由に必要なパッケージをインストールして戴いてかまいません。

CarbonEmacs の環境を手っ取り早く構築したい人は、
計算機がインターネットに接続されている事を確認してターミナルから

```
$ sudo apt-get update
$ sudo apt-get install task-emacs
$ sudo apt-get clean
```

を実行してください。
これだけで Carbon Emacs を利用する為の基本的な環境が構築されます。

注意！

emacs は /Applications/OSXWS/Emacs.app です。
「牛のアイコン」をダブルクリックして起動して下さい。
勿論 terminal や mlterm 等から $ emacs hoge.txt としても起動出来る様に alias を設定してあります。
デフォルトでは emacs で TeX のファイルを作成すると文字コードは UTF-8 になります。

TeX の環境を構築したい人は

```
$ sudo apt-get update
$ sudo apt-get install task-tetex
$ sudo apt-get clean
```

を実行してください。
これだけで齋藤さんの OTF パッケージでヒラギノを利用できる TeX 環境が構築されます。

注意！
TeX を利用する環境として emacs + YaTeX を想定しています。
pTeX は ptetex/UTF-8 で make されています。
terminal や mlterm 上で emacs で作成したファイルをコンパイルする時には
platex コマンドを使用して下さい。

apt と rpm を用いて細かな操作をする必要がある人は以下の記述が役に立つかもしれません。
勿論 man コマンドを活用してくださいね。

5.1 apt の「いろは」

Tiger 版には apt の GUI frontend である **Synaptic**[29] を用意しました。

```
$ sudo apt-get update
$ sudo apt-get install synaptic
$ sudo apt-get clean
$ sudo synaptic
```

で利用できます。
利用法は**マニュアル**[30] を参照してください。
以下の説ではターミナルでの apt の利用方法を簡単に紹介します。

5.1.1 毎回最初に必ずすべき事

apt を利用するには何は兎もあれ**データベースの更新**をする必要があります。
これをしないと現在の apt line[31] の状態を反映出来ず、
存在しないパッケージをインストールしようとしたりしてまともに働いてくれません。

ですから apt を弄る時は、必ず最初に

[29] http://www.nongnu.org/synaptic/
[30] file:///usr/osxws/share/synaptic/html/index.html
[31] apt が利用するパッケージやその情報が置いてある場所の事

```
$ sudo apt-get update
```

する様に癖をつけてください。

5.1.2 パッケージの探し方

例えば、現在利用できる emacs に関係するパッケージを知りたいとしましょう。
その様な時は apt-cache search を利用します。
具体的には

```
$ apt-cache search emacs
Mule-UCS - MULE-UCS is a coding system and character code translator system.
apel - Emacs 用の 基礎的な関数を提供するライブラリ
emacs -GNU Emacs エディタ
emacs-lisps - Carbon Emacs 用の便利な Lisp ライブラリ集
emacsen-common - Common facilities for all emacsen.
flim - Emacsen 用の message に関する表現形式や符号化のためのライブラリです。
mew - Emacs でメールを読むためのインターフェース
mew-common - Emacs/XEmacs 用 Mew 両方で利用するファイル/プログラム
readline - A library for editing typed command lines.
semi - Emacsen 用の MIME の機能を提供するライブラリ
task-emacs - emacs バーチャルパッケージ
yatex - 野鳥 (YaTeX) - Yet Another TeX mode for Emacs
```

の様にすれば、emacs に関連したパッケージの一覧が得られます。

5.1.3 パッケージのインストール

apt-cache を用いてインストールしたいパッケージが見つかったら、
apt-get install を利用してインストールします。

例えば、a2ps をインストールする場合には

```
$ sudo apt-get install a2ps
パッケージリストを読みこんでいます... 完了
依存情報ツリーを作成しています... 完了
以下の追加パッケージがインストールされます:
  psutils
以下のパッケージが新たにインストールされます:
  a2ps psutils
アップグレード: 0 個, 新規インストール: 2 個, 削除: 0 個, 保留: 0 個
```

```
1694kB のアーカイブを取得する必要があります。
展開後に 4768kB のディスク容量が追加消費されます。
続行しますか? [Y/n]
取得:1 http://www.bach-phys.ritsumei.ac.jp Leopard/fat/main psutils 1.17-10.5osx0 [115kB]
取得:2 http://www.bach-phys.ritsumei.ac.jp Leopard/fat/main a2ps 4.13b-10.5osx0.1 [1580kB]
1694kB を 16s 秒で取得しました (99.9kB/s)
変更を適用しています...
準備中                                  ############################## [100%]
更新/インストール中
  psutils-1.17-10.5osx0.fat              ############################## [100%]
  a2ps-4.13b-10.5osx0.1.fat              ############################## [100%]
完了
```

の様になります。

パッケージ間の依存関係が解決されて、psutils が同時にインストールされているのが判ります。

5.1.4 パッケージの削除

いらなくなったパッケージを削除したい時にはどうすれば良いでしょう？
その様な時は apt-get remove を利用します。
例えば、psutils を削除したい場合

```
$ sudo apt-get remove psutils
パッケージリストを読みこんでいます... 完了
依存情報ツリーを作成しています... 完了
以下のパッケージが削除されます:
  a2ps psutils
アップグレード: 0 個, 新規インストール: 0 個, 削除: 2 個, 保留: 0 個
0B のアーカイブを取得する必要があります。
展開後に 4768kB が解放されます。
続行しますか? [Y/n]
変更を適用しています...
準備中                                  ############################## [100%]
クリーニング/削除中
  a2ps-4.13b-10.5osx0.1.fat                                          [  0%] 警告: /usr/osxws/share/
して保存されした。
  a2ps-4.13b-10.5osx0.1.fat              ############################## [100%]
  psutils-1.17-10.5osx0.fat              ############################## [100%]
完了
```

の様になります。

ここで psutils に依存している a2ps が存在する場合、a2ps も削除するかどうか確認してきます。
ですから、あるパッケージを抜いてしまったが為に動かなくなるパッケージは、バグでない限りありません。

5.1.5 パッケージの更新

計算機のソフトにバグはつきものですし、機能が追加されてどんどん更新されていくものです。
MacOS X WorkShop でも、当然バグつぶしに因るパッケージのアップデートはしていきますし、
開発元が新版をリリースすれば出来る範囲で追随します。
即ち、パッケージはどんどん新しくなっていきます。
その様な新しいパッケージに自動で更新する方法があります。

一つ目は **依存関係を解決する時に、パッケージの削除が伴わないものだけ**を更新する方法で、
`apt-get upgrade` を利用します。

二つ目は **パッケージの削除を伴っても依存関係を解決して最新の状態にする**方法で、
`apt-get dist-upgrade` を利用します。

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

小林は開発に携わっている事もあり、常に apt-get dist-upgrade しています。

5.1.6 後片付け

apt-get で取得したパッケージは
`/usr/osxws/var/cache/apt/archives/` 以下に置かれます。
これは、`apt-get clean` を実行しない限り、残り続けます。
必ず最後に実行しておきましょう。

```
$ sudo apt-get clean
```

5.2 rpm の「いろは」

パッケージをインストールしたり更新したりするのは apt に任せれば良いのですが、
パッケージそのものを相手にする場合は rpm コマンドを直接操作する他ありません。
ここでは普段小林がよく使うコマンドについて簡単に解説します。

尚、パッケージの作成方法に付いては「パッケージの開発 (Section6) 」をご覧ください。

5.2.1 パッケージの情報あれこれ

今インストールされているパッケージの情報を知りたいとします。
先ず、今インストールされている全てのパッケージを知るには rpm -qa を用います。
実際には sort にパイプして

```
$ rpm -qa | sort | less
```

としたり、

```
$ rpm -qa | grep devel | sort
```

として目的のパッケージを探します。

こうして調べたいパッケージを見つけたならば、
何時誰が作ったパッケージで何時インストールされたのか、
等の情報を得ることができます。
それには rpm -qi を用います。
例えば a2ps の情報であれば

```
$ rpm -qi a2ps
Name        : a2ps                         Relocations: (not relocatable)
Version     : 4.13b                             Vendor: MacOS X WorkShop
Release     : 10.5osx0.1                    Build Date: 月  1/21 18:17:33 2008
Install Date: 水 10/ 1 16:59:31 2008          Build Host: MacBook
Group       : Applications/Publishing       Source RPM: a2ps-4.13b-10.5osx0.1.src.rpm
Size        : 4344144                          License: GPL
Signature   : DSA/SHA1, 月  1/21 18:17:35 2008, Key ID f367e1515c69cada
Packager    : KOBAYASHI Taizo <tkoba965@mac.com>
URL         : http://www.inf.enst.fr/~demaille/a2ps/
Summary     : テキストなどの Postscript へのフィルタ
Description :
a2ps は優れた印刷能力をもった，テキストを PostScript へ変換するフィルタ
です。
これは、プログラム言語や文字コード (ISO Latins, Cyrillic, EUC-JP 等)、
用紙、(インタフェースに対して)NLS などを広範囲にサポートしています。
いくつかのファイルを別のアプリケーションでフィルタリングさせる機能も持っ
ており、DVI や PostScript 等を全く同じインタフェースで区別することなく印
刷することができます。
```

として入手出来ます。

では、a2ps で一体どのようなファイルが何処にインストールされているのかを知るにはどうすれば良いのでしょうか。
それには rpm -ql を用います。

```
$ rpm -ql a2ps
/usr/osxws/bin/a2pdf
/usr/osxws/bin/a2ps
/usr/osxws/bin/a2ps.bin
/usr/osxws/bin/card
/usr/osxws/bin/composeglyphs
.....................
/usr/osxws/share/ogonkify/ptmri-o.ps
/usr/share/info/a2ps.info.gz
/usr/share/info/ogonkify.info.gz
/usr/share/info/regex.info.gz
```

最後に、このパッケージの履歴をみてみましょう。
それには以下の様にします。

```
$ rpm -q --changelog a2ps |less
```

5.2.2 パッケージのインストールと更新

ダウンロードしてきたパッケージをインストールする方法や、
自分で構築したパッケージをインストールする方法を述べます。
例えば、パッケージ hoge-1.23-10.5osx1.ppc.rpm をインストールするには

```
$ sudo rpm -ivh hoge-1.23-10.5osx1.ppc.rpm
```

或は

```
$ sudo rpm -Uvh hoge-1.23-10.5osx1.ppc.rpm
```

とします。
-i オプションはインストールを、
-U オプションは更新を意味しますが、
殆どの場合 -Uvh で済んでしまいます。

他に、--force や --nodeps 等のオプションがありますが、
パッケージの作成をしない限り、まず使う状況は無い筈です。

5.2.3 パッケージの削除

大抵は apt-get remove で事足りるのですが、
開発作業中にどうしても依存関係を破壊しても一時的に削除しなければならない場合は
rpm コマンドに頼る他ありません。
その様な時は

```
$ sudo rpm -e --nodeps hoge
```

とします。

6　パッケージを開発する

ここでは MacOS X WorkShop のパッケージを開発する方法を述べます。
コマンドは rpm ではなく rpmbuild を使います。

MacOS X WorkShop に固有の事項について説明しますので、
一般的な rpm パッケージの作成方法は、
Vine Linux の **Making RPM**[32] や、
Momonga Linux の **Specfile-Guidance**[33] を参考にしてください。

亦、パッケージに固有の項目に関してはパッケージメモ (Section10 参照) を参照して下さい.
尚、

```
$ rpm -i hoge-1.0-10.5osx1.src.rpm
```

とすると、
spec file は `~/rpm/SPECS` に、
source files は `~/rpm/SORCES` に、
それぞれ入ります。

apt tree に在る rpm source package を利用するのであれば

```
$ cd ~/rpm/SRPMS
$ apt-get source hoge
```

とすると、
hoge の source package が `~/rpm/SRPMS` にダウンロードされた後に
spec, source files を所定の位置に展開してくれます。
パッケージを作るには

```
$ cd ~/rpm/SPEC
$ rpmbuild -ba hoge.spec
```

すると、hoge の source package が `~/rpm/SRPMS` に作成され、binary package が `~/rpm/RPMS` 以下の適当なディレクトリに作成されます。

6.1　設定ファイルの編集

rpm のパッケージを作る前に、パッケージャの情報を `~/.rpmmacros` に記述しておきます。

[32] http://www.vinelinux.org/MakingRPM/index.html
[33] http://www.momonga-linux.org/docs/Specfile-Guidance/ja/index.html

vi 等のエディタで ~/.rpmmacros の packager の項目に、
アルファベットで自分の名前とメールアドレスを以下の様に整えます。

```
%_topdir ${HOME}/rpm
%packager KOBAYASHI Taizo    <xxxxxxx@xxxx.xxx>

%_tmppath %{_topdir}/temp
%_signature gpg
%_gpg_name XXXXXXXX
```

これで貴方が作るパッケージには貴方の名前とメールアドレスが刻まれます。
gnupg の public key をお持ちの方はご連絡いただければ OSX-keyring に登録いたします。

6.2 spec file のタグ

ここでは MacOS X WorkShop 固有のタグに関する方針を述べます。

Version, Release rpm のパッケージは
(Name)-(Version)-(Release)-(architecture).rpm
の形をしています。
(Name), (Version) はパッケージングするソフトに依存するので一意に決定されますが、
(Release) の付け方はディストリビューション毎に取り決めがあるのが普通です。
MacOS X WorkShop では

Panther `10.3tk(release number)`

Tiger `10.4osx(release number)`

Leopard `10.5osx(release number)`

と付ける事にします。

(architecture) は特に指定しなければ i386 か ppc になります。
スクリプトやドキュメントだけのパッケージでは `BuildArch: noarch` を指定すると noarch になります。

尚、noarch 以外の殆どパッケージは fat になっています。

defattr %files セクションに、そのパッケージに含まれるファイルを書き込みますが、
それらのファイルのオーナーとグループを指定してやる必要があります。
それが %defattr タグです。

MacOS X WorkShop では、
`%defattr(-, root, wheel)`
を標準とします。

6.3 rpm macro

ここでは MacOS X WorkShop 固有のマクロについて述べます。

デフォルトのマクロは /usr/osxws/lib/rpm/macros に記述されているので、
パッケージを作成する前に必ず一度は目を通しておいてください。

先ず、マクロの内容が MacOS X WorkShop 固有のものを列挙します。

_prefix /usr/osxws
: 基本的に全てのバイナリーやライブラリ、ドキュメント等は/usr/osxws 以下にインストールします。

_var /usr/osxws/var

_sysconfdir /usr/osxws/etc

__libtoolize /usr/bin/glibtoolize, /usr/bin/glibtool を使います。
: /usr/bin/libtool はライブラリを作る ar, ranlib の代替になる存在の様で、
gnu の libtool とは役割が異なります。

次に、MacOS X WorkShop のみに存在するマクロを列挙します。

```
#------------------------------------------------------------------------
#
# MacOS X macro
%_cpmac         /Developer/Tools/CpMac
%_mvmac         /Developer/Tools/MvMac
%_fontsdirmac   /Library/Fonts
%_docdirmac     /Users/Shared/Docs/%{name}-%{version}
%_appdirmac     /Applications
%_stuffit       open -a StuffIt\\ Expander.app
%_Xcode         open -a /Developer/Applications/Xcode.app
```

6.4 Universal Binary

2006 年の 1 月以降 OSXWS はデフォルトで Universal Binary のパッケージを作成しています。
ここでは Universal Binary rpm package の作成方法を簡単に示します。
主な内容は **MacWiki**[34] の UniversalBinary の項目に記してありますので、
合わせてご覧ください。

はじめに

原則として PPC, Intel 双方のバイナリを独立してビルドし
最後に lipo で結合する方法を採ります。

[34] http://macwiki.sourceforge.jp/cgi-bin/wiki.cgi

spec file の基本形

冒頭

`%define uname_release ‘(uname -r) 2>/dev/null‘` を置き、
configure への host 指定に利用します。
`BuildArch: fat` を指定します。

%prep section

以下の様にして PPC, Intel それぞれに tree を分けます。

```
%setup -q -c %{name}-%{version}
pushd %{name}-%{version}
%patch1 -p1 -b .fat
popd

mv %{name}-%{version} PPC
cp -rp PPC INTEL
```

%build section

以下の様にして PPC, Intel それぞれ独立にビルドします。

```
pushd PPC
CFLAGS="-O3 -arch ppc" \
CXXFLAGS="$CFLAGS" \
%configure --host=powerpc-apple-darwin%{uname_release} \
--disable-dependency-tracking
perl -pi -e 's@-dynamiclib@-dynamiclib -arch ppc@g' libtool
make
popd
pushd INTEL
CFLAGS="-O3 -march=i686 -mtune=pentium-m" \
CXXFLAGS="$CFLAGS" \
%configure --host=i686-apple-darwin%{uname_release} \
--disable-dependency-tracking
perl -pi -e 's@-dynamiclib@-dynamiclib -arch i386@g' libtool
make
popd
```

%install section

以下の様にして Universal Binary を作成します。

```
pushd PPC
```

```
mkdir -p ${PWD}-root%{_bindir}
make install DESTDIR=${PWD}-root
popd
pushd INTEL
mkdir -p ${PWD}-root%{_bindir}
make install DESTDIR=${PWD}-root
cp -fRP COPYRIGHT README VERSION TODO html ..
popd

## Make Universal Binaries
filelist=$(find ./PPC-root -type f | xargs file | sed -e 's,^\./PPC-root/,,g' | \
        grep -E \(Mach-O\)\|\(ar\ archive\) |sed -e 's,:.*,,g' -e '/\for\ architecture/d')

for i in $filelist
do
        /usr/bin/lipo -create PPC-root/$i INTEL-root/$i -output `basename $i`
        cp -f `basename $i` PPC-root/$i
done

# install
mkdir -p %{buildroot}
tar cf - -C PPC-root . | tar xpf - -C %{buildroot}
```

6.5 その他

ライブラリに関する問題

libintl, libiconv 等が読み込まれなかったり、
/usr/include/{glob,regex}.h 等が gnu のものとコンフリクトしたりする事があります。
大抵、ld が `multiple definitions of symbol....` や
`undefined symbols....` 等とメッセージを出し、
該当する関数名とライブラリ名を表示してくれるので、
出来るだけ MacOS X 側が用意しているライブラリやヘッダファイルを利用する様
にします。

libtool, autotools に関する問題

MacOS X 10.5.5 + Xcode 3.1.1 では、

`libtool`	1.5.22 (glibtool, glibtoolize)
`aclocal, automake`	1.10
`autoheader, autoconf`	2.61

が用意されています。

MacOS X WorkShop では、automake1.{479}, autoconf-2.13 を用意しています。
これらは必要に応じて、`aclocal-1.9 -I /usr/share/aclocal` 等として利用します。

libtool を利用する時は、
configure の前で `glibtoolize --copy --force` とし、
configure の後で `cp -f /usr/bin/glibtool libtool`
とするとうまく行く事があります。

C++ に関する問題

UNIX2003 Symbol not found

後ろ向きな解決であるが以下で一応解決する。

```
export CXXFLAGS="$CXXFLAGS -mmacosx-version-min=10.4"
```

pkg-config

以下の設定で利用している。

PKG_CONFIG_PATH=”/usr/osxws/lib/pkgconfig:/usr/osxws/share/pkgconfig:/usr/lib/pkgconfig:/usr/X11/lib/p

libpng

/usr/X11/lib/ 以下に在る。 ただし、/usr/X11/lib/pkgconfig/libpng{12}.pc 内で共に Cflags: -I${includedir}/libpng12 を指していながら、実際には/usr/X11/include/libpng しかない！

7　インストーラを作る

MacOS X WorkShop 10.5 のインストーラを Tiger 上の Xcode-3.1.1 で小林が開発した際の備忘録です。
屹度間違いがあるとおもいます。
総合的且つ正確な情報は Apple の **Tools: Software Distribution**[35] をご覧下さい。

インストーラのソース一式は「インストール (Section4.3 参照) 」のページからダウンロード出来ます。
姉妹 tree の作成に役立ててください。
尚、インストーラを作るには
`/Developer/Applications/Utilities/PackageMaker.app`
を使います。

7.1　段取り

一般的にインストーラを作成するのに必要な段取りは以下になります。

1. 作業場所を作る。
2. インストールするファイル類を用意する。
3. インストールする手順を所定の各ファイルに記述する。
4. 「PackageMaker」でインストーラを作成する。
5. 「ディスクユーティリティ」でディスクイメージを作成する。

これらの作業は一寸面倒です。
覚え書き程度に書いていきます。

7.2　作業場所を作る

インストーラを作る作業場には
「インストールするファイル類」と「手順を記したファイル類」を置く場所が必要
です。

MacOS X WorkShop では、ディレクトリ「OSX-WS」を作業場のルート・ディレクトリとし、
「インストールするファイル類」は「OSX-WS/OSXWS」に、
「手順を記したファイル類」は「OSX-WS/Resources」に
置いています。

以下これらのディレクトリ構造を基にして記述していきます。
適宜読み替えて下さい。

[35] http://developer.apple.com/documentation/DeveloperTools/Conceptual/SoftwareDistribution/index.html

7.3　インストールするファイル類を用意する

MacOS X WorkShop のインストーラがすべき事は、
最低限の `apt-rpm` 環境をつくる事 です。

インストーラのソースー式[36] の中の make-tree.sh が、必要なファイルを所定の位置にコピーするスクリプトです。`RPMDIR=/Users/tkoba/rpm/RPMS/noarch` を適宜書き換えた上で、「OSX-WS/OSXWS」の中で実行して下さい。

このスクリプトの中でしている事は

1. **ディレクトリの作成**
2. **`apt, rpm` バイナリー類のコピー**
3. **基本パッケージ `OSX-{Preferences,X11,system}*` のコピー**

です。
ただし、基本パッケージのコピーの後、
同一パッケージの複数のバージョンが入っていない事を確認してください。

これで、「OWX-WS/OSXWS」以下にインストールされるファイル類が準備されます。

7.4　インストールする手順を所定の各ファイルに記述する

ソフトをインストールするには、インストールしようとしている環境が適切であるか確認する必要がありますし、
ファイルを所望の位置に置いた後に、某かのお決まりの設定をする必要がある事もままあります。

ここではその様な手順を実現する方法を述べます。

PackageMaker Help に記述がありますが、インストーラが行う手順は以下になります。

1. `InstallationCheck`
2. `VolumeCheck`
3. `preflight`
4. `preinstall or preupgrade`
5. `(INSTALLER EXTRACTS AND INSTALLS THE PACKAGE'S CONTENTES.)`
6. `postinstall or postupgrade`
7. `postflight`

[36] MacOSX-WS-10.5.2.tar.bz2

MacOS X WorkShop ではこの中の、
`InstallationCheck, postinstall, postupgrade` を利用
しています。
以下順次説明していきます。

InstallationCheck 「OSX-WS/Resources/InstallationCheck」がスクリプトの実態で、
「OSX-WS/Resources/Japanese.lproj/InstallationCheck.strings」がメッセージ（日本語環境 UTF-8）
の実態です。

ここでは、インストールしようとしている環境が適切であるかを確認します。

している事は、

- `MacOS X` のバージョンは適切か？
- `X11` がインストールされているか？
- `Xcode` がインストールされているか？

のチェックと、状況に応じたメッセージの表示です。

詳細は **Apple** のドキュメント[37] をご覧下さい。

postinstall, postupgrade 「OSX-WS/Resources/postinstall」がスクリプトの実態で、
「OSX-WS/Resources/postupgrade」は、現在 postinstall へのシンボリックリンクです。

ここでしている事は、

- `rpm database` の初期化
- 基本パッケージ `OSX-{Preferences,X11,system}*` のインストール
- ドットファイルを各ユーザーへ配布

です。

最後に、「OSX-WS/{Welcome,ReadMe,License}.rtf」を作っておきます。

7.5 インストーラを作成する

ファイル類の準備ができたら、PackageMaker を使ってインストーラを作ります。

● PackageMaker を起動すると、左側カラムの「Distribution」が選択されたウィンドが現れます。
右側カラムの「Configuration」タグを選択して必要事項を記述します。

● 「Requirements」タグではインストールに必要な条件のチェックを指示します。

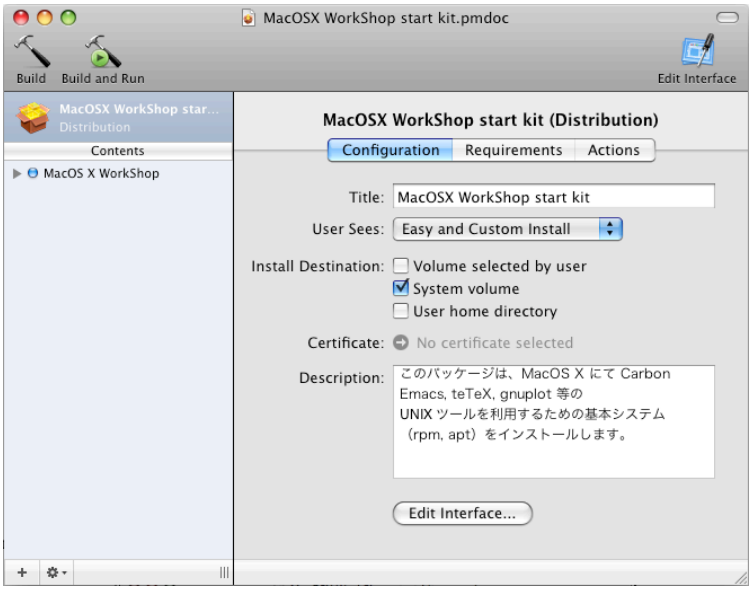

図 1: 「PackageMaker-Distribution-Configuratio」

新たな項目は左下の「+」を押して作ります。既存の項目の編集は、項目をダブルクリックします。

● 次に、左側カラムの「Contents」にある項目を選択して、右側カラムの「Configuration」タグ以下を記述します。

● 左側カラムの「Contents」にある項目を開いて、右側カラムの「Configuration」タグ以下を記述します。

● 右側カラムの「Contents」タグ以下でファイルやディレクトリのオーナーやパーミッションをチェックします。

● 右側カラムの「Contents」タグ以下でファイルやディレクトリのオーナーやパーミッションをチェックします。

● 全部設定し終わったら保存した後に、「Project」→「Build...」でインストーラを作成します。

7.6 ディスクイメージを作成する

これまでの作業でパッケージのインストーラ MacOSX WorkShop start kit.pkg が出来ました。

[37]http://developer.apple.com/documentation/DeveloperTools/Conceptual/SoftwareDistribution/index.html

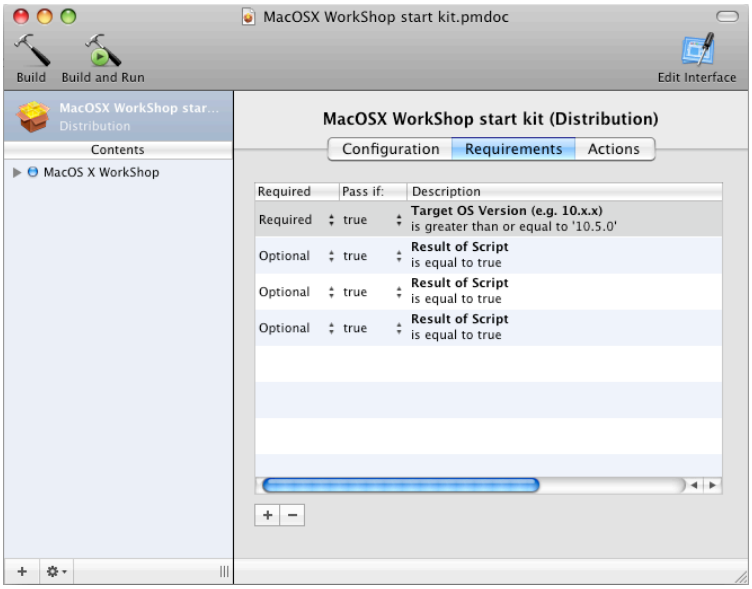

図 2: 「PackageMaker-Distribution-Requirements」

後はこれをディスクイメージの中に置いておしまいです。
以下の作業をします。

イメージの作成 「ディスクユーティリティ」を立ち上げて「新規イメージ」ボタンを押し、
ディスクの容量を必要なだけ設定して（私は 7MB にしました）空のイメージを作ります。
空のイメージをマウントして、
そこに ReadMe.rtf と作成したインストーラを入れてマウント解除します。

イメージの圧縮 「ディスクユーティリティ」のメニューから「イメージ」→「変換...」を選択し、
上で作成したイメージを選択します。
「イメージフォーマット」に「圧縮」を選んで保存します。

尚、これらの処理をスクリプトにしてあります。
インストーラのソース **MacOSX-WS-10.5.2.tar.bz2**[38] の中にある make-pkg.sh をご覧ください。

[38] MacOSX-WS-10.5.2.tar.bz2

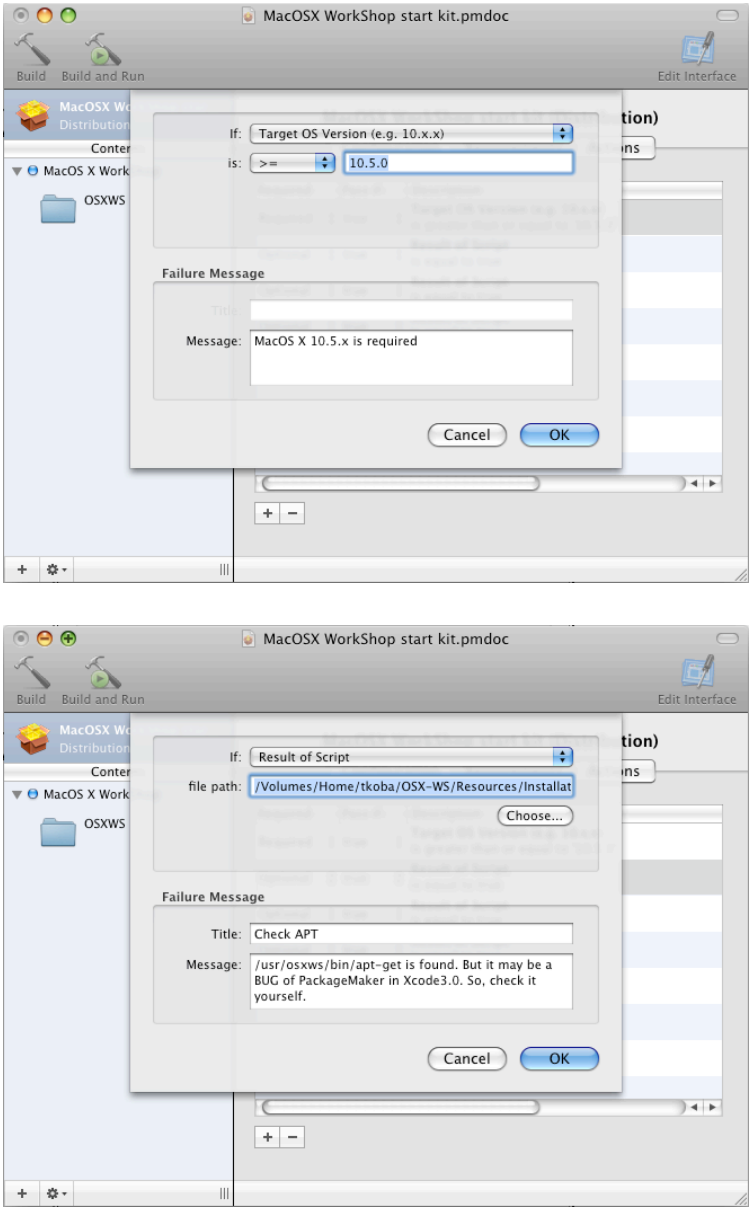

図 3: 「PackageMaker-Distribution-Requirements Describe」

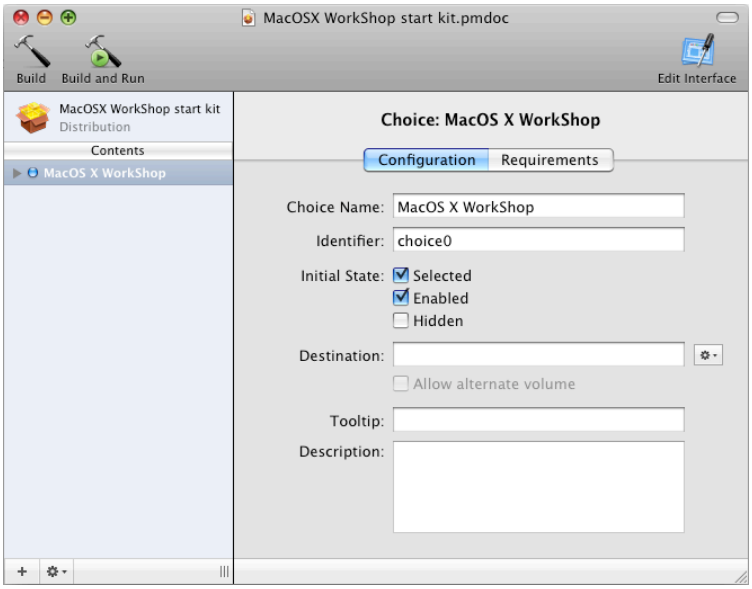

図 4: 「PackageMaker-Contents-Configuration」

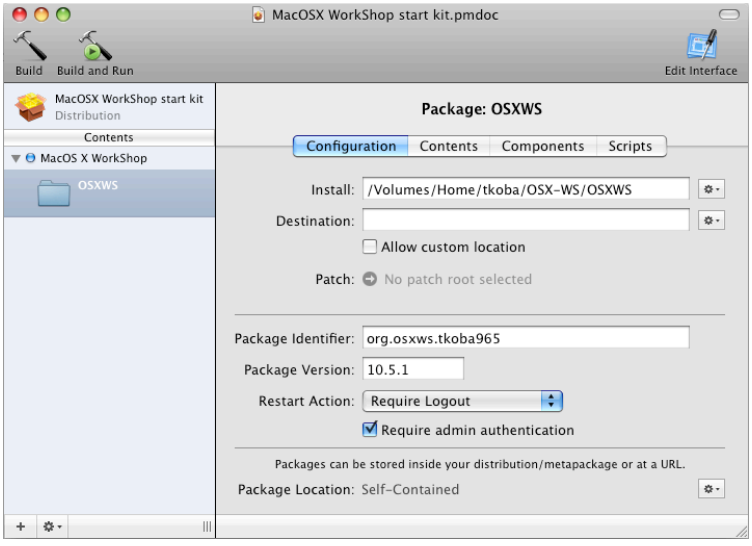

図 5: 「PackageMaker-Package-Configuration」

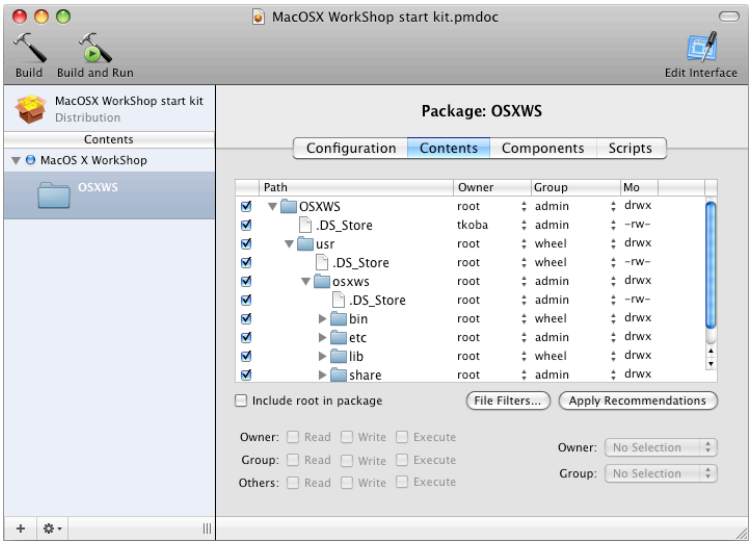

図 6: 「PackageMaker-Package-Contents」

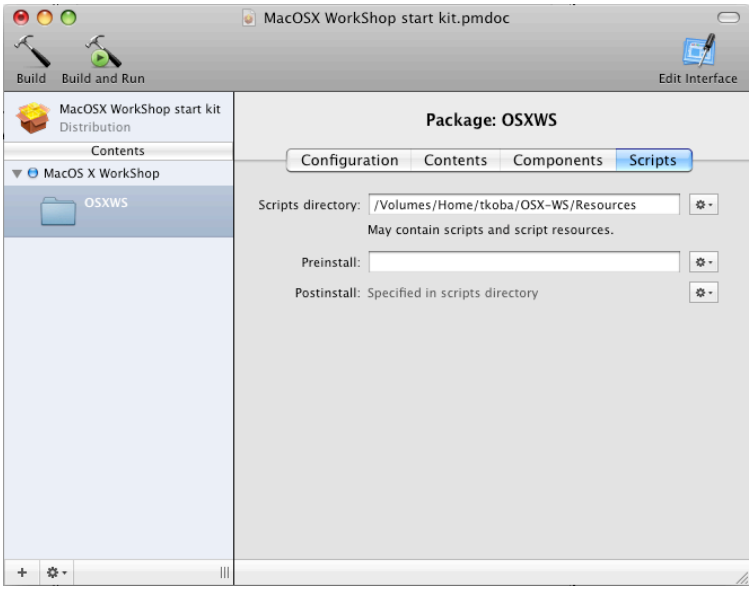

図 7: 「PackageMaker-Package-Scripts」

8　apt-rpm tree を作る

MacOS X WorkShop tree を用意した際の備忘録です。
屹度間違いがあるとおもいます。
間違いを発見されましたらご連絡ください。
また、貴方独自の add-on tree を用意される場合もほぼ同じ手順で可能です。

8.1　段取り

一般に apt-rpm tree を作成するのに必要な段取りは以下になります。

1. tree を置く場所を用意する。
2. パッケージを置く。
3. データベースを作る。
4. apt-line を記述する。

これらの作業は web や anonymous ftp に物を置いた経験があれば簡単です。
覚え書き程度に書いていきます。

8.2　tree を置く場所を用意する

tree は「web」と「ftp」と「local disk」に置けます。

ftp も web も local disk も単純にディレクトリを作るだけなので、
ここでは web に置く方法を述べます。
以下の条件を満たしていれば大丈夫です。

- 数百 MB のファイルを置く容量を確保出来るか？
- サーバの性能は十分か？
  G4 1GHz 程度の性能があれば、実質問題はないでしょう。

これ以降では tree のルートディレクトリを「OSXWS」とします。

8.3　パッケージを置く

MacOS X WorkShop は現時点では Panther 版と Tiger 版と Leopard 版があり、
それぞれを別途置かなければ成りません。

現在はそれぞれ「OSXWS/{Leopard,Tiger,Panther}」の中に置いています。

- ソース・パッケージ
  ソース・パッケージ（*.src.rpm）は「OSXWS/{Leopard,Tiger,Panther}/SRPMS.main」に置きます。
  このサフィックス「main」はパッケージをカテゴリー分けする際に意味をなします。
  即ち、core, devel, plus 等に分類したい場合はそれぞれ「SRPMS.core」などとすれば良い訳です。
  MacOS X WorkShop では、minor release 迄の小さな更新用に updates カテゴリーをもうけています。
- バイナリー・パッケージ
  バイナリー・パッケージ（*.{ppc,noarch}.rpm）は「OSXWS/{Leopard,Tiger,Panther}/ppc/RPMS.main」に置きます。

8.4 データベースを作る

tree の実態が置かれたら、データベースを作成します。
MacOS X WorkShop の場合は

```
[Leopard] genbasedir --progress --bz2only [OSXWS]/Leopard/fat main updates
[Tiger]   genbasedir --progress --bz2only [OSXWS]/Tiger/fat main
[Panter]  genbasedir --progress --bz2only [OSXWS]/Panther/ppc main
```

としています。
当然 [OSXWS] は適切なディレクトリに置き換えてください。
この操作はパッケージを更新する度に必要になりますから、
シェルスクリプトにでもしておくと良いでしょう。

8.5 apt-line を記述する

最後に apt-line を記述する為に apt パッケージを変更します。

```
$ cd ~/rpm/SRPMS
$ apt-get source apt
```

して、apt のソースパッケージを展開します。
次に、~/rpm/SOPURCES/sources.list-0.5.15-leopard に貴方が作った apt-line を記述します。
既にある記述を真似れば難しくないと思います。
先に /usr/osxws/etc/apt/sources.list を編集して動作を確認しておき、
正しく動作するものを ~/rpm/SOPURCES/sources.list-0.5.15-osxws にコピーするのが良いでしょう。
/usr/osxws/etc/apt/sources.list.d 以下に add on の sources tree を記述したファイルを置いてもかまいません。
最後にスペックファイルを編集（リリース番号の更新と変更履歴を記述）したら、

```
$ cd ~/rpm/SPECS
$ rpmbuild -ba apt-osx.spec
```

で所望の新しい apt パッケージが ~/rpm/RPMS/fat 以下に作られます。
この新しい apt パッケージも忘れずに貴方の tree に加えてください。

9 Tiger 版からの変更点

Leopard 版は Tiger 版から以下の変更がなされています。
具体的な仕様の変更は以下の通りです。

- OSXWS の root dir を /usr/local から /usr/osxws へ変更。
- /Applications へ配置していたものを/Applications/OSXWS/ へ変更。
- dot.emacs ファイル群の整理。
- デフォルト日本語コードを EUC から UTF-8 へ変更。特に emacs と TeX はデフォルトが UTF-8 に変わりますのでご注意ください。

9.1 パッケージについて

- 追加したパッケージ
  - Skim: Mxdvi の代替
  - pdfsync
  - Sparckle
  - mpfr
  - OpenEXR
  - boost
  - ilmbase
  - SDL
  - pixman
  - PDFlib-Lite
- 削除したパッケージ
  - Mxdvi: 開発元が活動を停止の模様
  - subversion: Leopard 付属
  - VirtueDesktops: Spaces (Leopard 付属) に代替
- ペンディングパッケージ
  - sylpheed: ことえり対応 kinput2 待ち
  - python-numeric
  - f2c, fftw3

10　パッケージメモ

ここでは主に各パッケージに施した変更や拡張について記します。

全てのパッケージについて記述している訳ではありません。

パッケージの詳細は rpm2html による **RPM 解説データベース (Leopard)**[39] **RPM 解説データベース (Tiger)**[40] **RPM 解説データベース (Panther)**[41] をご利用ください。

10.1　Emacs 関連

MacOS X WorkShop では、今のところ CarbonEmacs のみを提供しています。
CarbonEmacs に関する情報は **MacEmacs**[42] や、銭谷さんの **Carbon Emacs パッケージ**[43] をご覧ください。
収録しているパッケージ内容は銭谷さんのものを基本としています。
ただし、CarbonEmacs 本体と、Emacs Lisp は別のパッケージにして、
独立して更新やメンテナンスが出来る様に配慮してありまし、
elisp の置き場所もパッケージ内部ではなく `/usr/osxws/share/emacs/` 以下に変更しています。
自分独自の elisp （但しバイトコンパイルしていないもの）を置く場合には、
`/usr/osxws/share/emacs/site-lisp/` 以下に置いて下さい。
また、Vine Linux から alternatives を移植してありますから、
XEmacs など他の Emacsen を共存して入れる事も可能（な筈）です。

関連するパッケージは以下になります。

apel Emacs 用の 基礎的な関数を提供するライブラリ

emacs GNU Emacs エディタ (Carbon 版)

emacs-lisps Carbon Emacs 用の便利な Lisp ライブラリ集
　　銭谷さんのパッケージに入っている Lisp を纏めたものです。

emacsen-common Common facilities for all emacsen.

flim Emacsen 用の message に関する表現形式や符号化のためのライブラリです。

[39] ../Leopard/rpm2html/
[40] ../Tiger/rpm2html/
[41] ../Panther/rpm2html/
[42] http://macemacsjp.sourceforge.jp/
[43] http://homepage.mac.com/zenitani/emacs-j.html

mew Emacs でメールを読むためのインターフェース

semi Emacsen 用の MIME の機能を提供するライブラリ

aspell emacs 上で利用できるスペルチェッカー
「Tools」メニューからスペルチェックを選べば利用出来ます。
コンソールからの利用も可能です。

task-emacs emacs バーチャルパッケージ
このパッケージを apt でインストールすると、次のパッケージが自動でインストールされます。
alternatives emacs emacsen-common emacs-lisps apel flim semi Mule-UCS

yatex 野鳥 (YaTeX) - Yet Another TeX mode for Emacs

10.2 TeX 関連

パッケージの内容は、土村さんと小林が主にメンテナンスしている Vine Linux の teTeX package 群と基本的に同一です。

teTEX 本体 UTF8 で作成しています。
ptetex-20061213 base です。

TEXmacros tetex-macros tetex で用いる追加マクロパッケージ集です。
次のマクロを収録しています. jsclasses, prosper, epsbox.sty, eclepsf.sty

texmacro-otf 齋藤修三郎さんによる「OpenType Font 用 macro と VF」です。
齋藤氏が配布されているマクロや VF 以外に次の補助ツール類を同梱しています。

`updmap-otf`
dvipdfmx, udvips 等で埋め込むフォントを設定する為のツールです。
`sudo updmap-otf auto` とすると
OTF-Hiragino パッケージがインストールされていればヒラギノを埋め込む様に設定し、
無ければ noFont の設定をします。
`sudo apt-get install task-tetex` としていれば、デフォルトでヒラギノを埋め込みます。
他にも、モリサワ基本 7 書体パッケージ (OTF-Morisawa-basic7) がインストールされていれば、
`sudo updmap-otf morisawa` とすると利用可能になります。
利用方法は `updmap-otf` で表示されます。

font 関連 jvf

makejvf

OTF-Hiragino MacOS X 付属のヒラギノフォントを利用する為の設定パッケージです。

OTF-Morisawa-basic7 購入して MacOS X にインストールされたモリサワ基本 7 書体 OpenType Fonts を利用する為の設定パッケージです。

OTF-Morisawa-RmSgSmg 購入して MacOS X にインストールされた以下のモリサワ OpenType Fonts
A-OTF-RyuminPro-{Regular,Heavy}.otf,
A-OTF-ShinGoPro-{Regular,Heavy}.otf,
A-OTF-ShinMGoPro-{Regular,Bold}.otf
を利用する為の設定パッケージです。

urw-fonts free で品位の高い 35 の標準 PostSecript Fonts です。

ttfonts-ja free の日本語 TrueType Font である「さざなみフォント」[44] をインストールします。

その他 LaTeXiT LaTeXiT[45]
Apple の Keynote 等で数式を扱う際に利用出来ます。

task-tetex TEX 関連パッケージを簡単にインストールするための仮想パッケージです。

dvipdfmx dvi file を PDF file に変換するツールです。
齋藤さんの OTF パッケージに対応しています。

latex2html TEXfile を html file に変換するツールです。
このドキュメントも latex2html を用いて書かれています。

yatex

10.3 X11 関連

X server には Xquartz を、Window Manager には quartz-wm を利用します。
.Xclients や .Xresources では設定できない項目があり、その場合は

```
$ defaults write com.apple.x11 xxx yyy zzz
```

等とする必要があります。
詳細は

```
$ man Xquartz
$ man quartz-wm
```

で調べてください。

ImageMagick 画像ファイルの表示/処理を行う X のアプリケーション

Plotmtv X11 上で動作するグラフ作成ツール

Xaw3d A version of the MIT Athena widget set for X.

ghostscript A PostScript(TM) interpreter and renderer.
デフォルトでヒラギノを使用します。

[44] http://sourceforge.jp/projects/efont/files/
[45] http://ktd.club.fr/programmation/latexit_en.php

gnuplot A program for plotting mathematical expressions and data.

gv A X front-end for the Ghostscript PostScript(TM) interpreter.

mlterm 他言語の表示が可能な X 上のターミナルソフト
「Ctlr + 右クリック」でコントローラが現れます。

openMotif The Open Motif runtime components.

ttfonts-ja Free Japanese TrueType fonts

内容はさざなみフォント[46] です。

urw-fonts Free versions of the 35 standard PostScript fonts.

xgraph xgraph - 2D data plotting program (+ hack 9 + color PS + and so on.)

yaplot yaplot - an easy 3D modeller and animator

10.4 開発関連

rpm The RPM package management system.

詳細は spec file を参照してください。

apt RPM を扱える Debian のパッケージツール apt(Advanced Packaging Tool)

static build して strip してあります。

gcc gcc-4.3.0 A GNU Compiler Collection.

HEPonX[47] のものです。gfortran を提供します。

Apple 提供の gcc-4.0,4.2.1 とは update-alternatives で切り替え可能です。

10.5 System 関連

OSX-system MacOS X の標準ライブラリやツールを rpm system に教える為のパッケージです。

`/usr/osxws/etc/{profile-osxws,csh.login-osxws,zprofile}` が加えられ `/usr/osxws/bin, /usr/X11/bin` 等にパスを通します。

MacOS X WorkShop インストーラによりインストールされます。

OSX-X11 X11 のライブラリやツールを rpm system に教える為のパッケージです。

MacOS X WorkShop インストーラによりインストールされます。

[46] http://sourceforge.jp/projects/efont/files/
[47] http://www-jlc.kek.jp/ fujiik/macosx/10.5.X/HEPonX/

OSX-Preferences OSX-Preferences パッケージは MacOS X WorkShop の基本システムの一部で、デフォルトのユーザ設定ファイル (.Xresources, .bash_logout, .bash_profile, .bashrc) 等を収録しています。

各ユーザーに配布された設定ファイルを更新する為に、
osxws-upgrade スクリプトを収録しています。
`OSX-Preferences package` の更新に対応する為に個人用の記述は
`.bashmyrc, .cshmyrc, .zshmyrc, .emacs.my.el` を作成し、
その中に記述して下さい。

`/usr/osxws/share/OSXWS/jp/` 内にインストールされますから、
新規ユーザー作成時に自動で設定ファイル類がコピーされます。

MacOS X WorkShop インストーラによりインストールされます。

OSX-base rpm system を利用する為に最低限必要なパッケージをインストールする為の仮想パッケージです。

MacOS X WorkShop インストーラを実行した直後に `sudo apt-get install OSX-base` しておく必要があります。

11　スクリーンショット

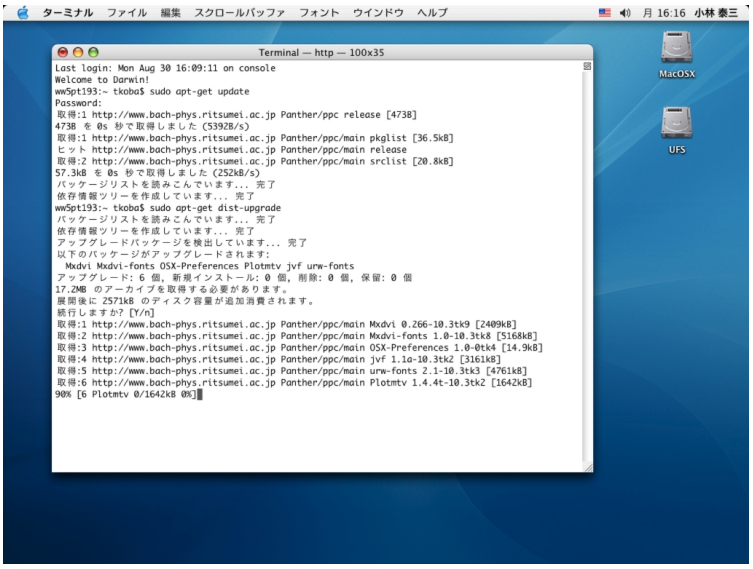

図 8: apt-get でアップデートパッケージをダウンロード中。

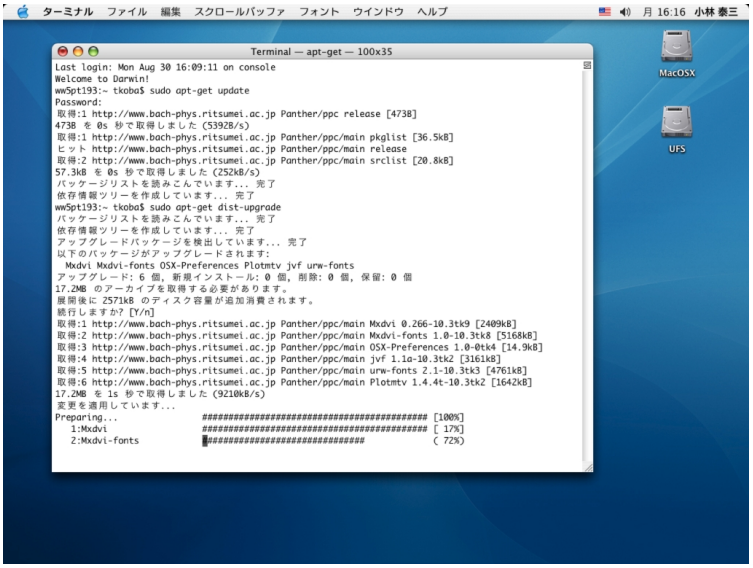

図 9: apt-get でパッケージを更新中。

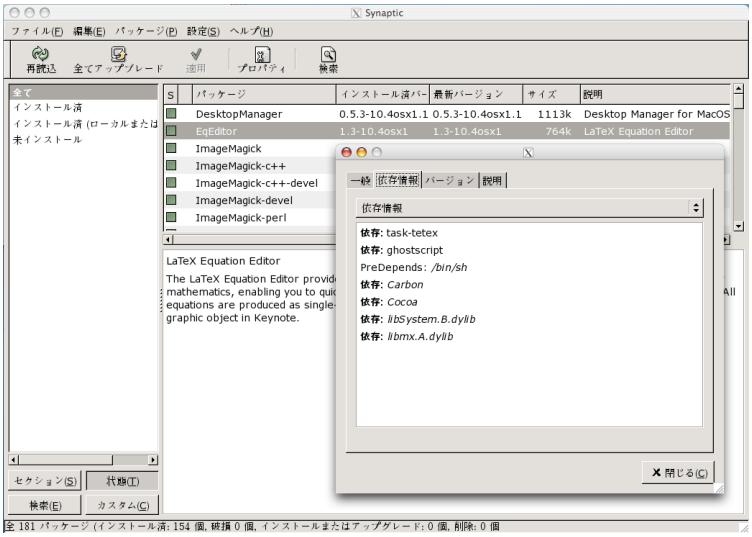

図 10: X11 上の apt-rpm frontend である synaptic

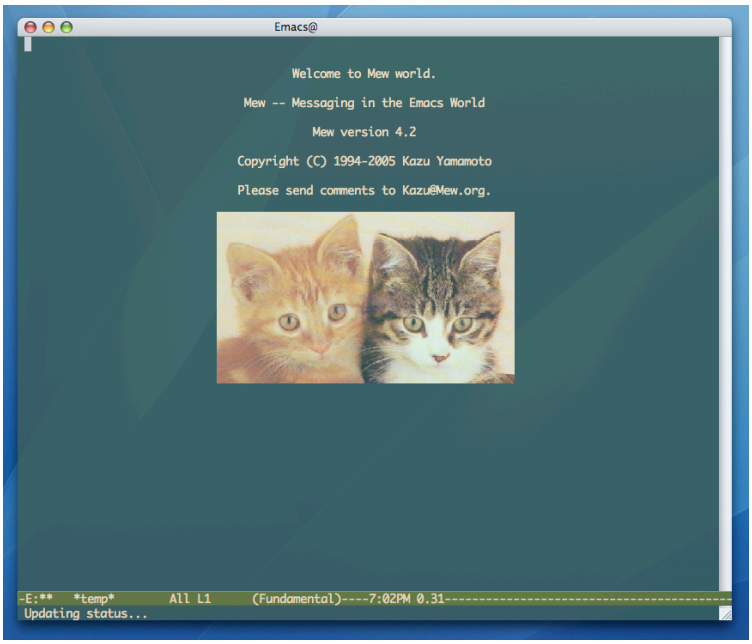

図 11: Emacs で mew を立ち上げているところ。

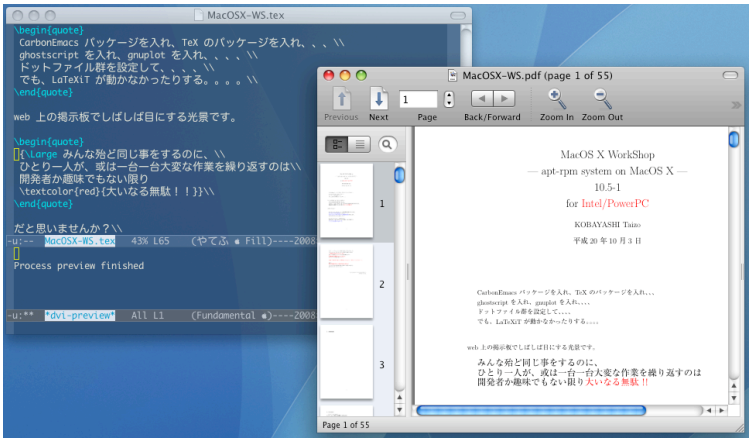

図 12: Emacs で yatex を用いて LaTeX の文章を書き Skim でプレビューしているところ。

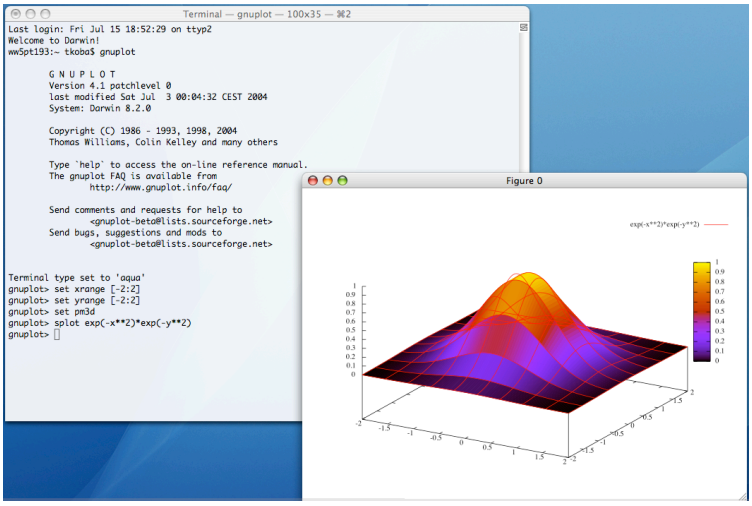

図 13: お馴染み gnuplot です。aquaterm, PDFlib-Lite も同時にインストールされます。

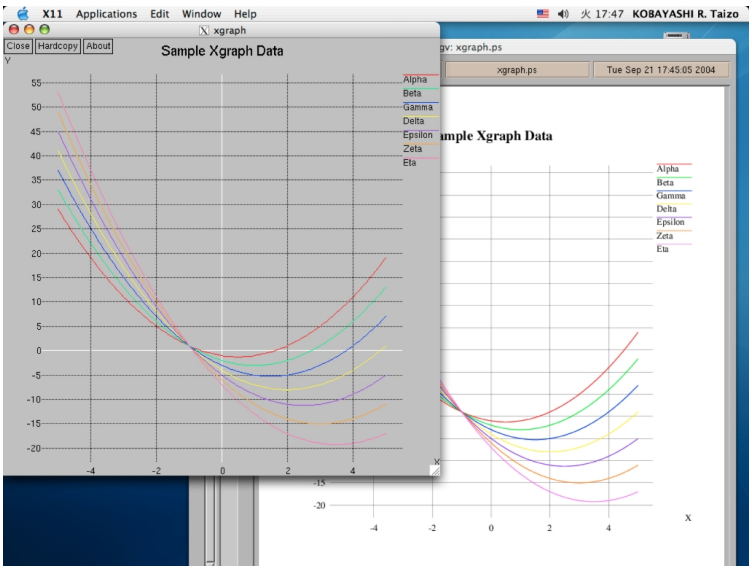

図 14: Vine Linux と同じパッチを適用してあります。Color PS を生成できます。PS ファイルをダブルクリックすれば大抵の場合 PDF に変換でき、プレビューで確認と印刷ができます。

12　既知の問題点と注意点

2008 年 10 月 1 日現在、MacOS X WorkShop には以下の様な問題があることが分かっています.

Mxdvi (dvipdfmx + Skim に代替済みです。)

- Intel Mac では動作しない
- PPC Mac では環境変数の設定が必要

```
$ cat ~/.MacOSX/environment.plist
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple Computer//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/Pr
<plist version="1.0">
<dict>
        <key>TEXMFCNF</key>
        <string>/usr/osxws/share/texmf/web2c</string>
</dict>
</plist>
```

etc. その他気付いていない問題点があるかもしれません.

13　過去の議論

ここは **Mac Wiki**[48] にて行われた過去の議論の書庫です。

13.1　10.5-3 公開迄

- 吉田です。さて、gnuplot の件はお蔭様で解決いたしましたが、実は、この件でいろいろ試していたときにもう 1 件の不具合に遭遇しました。それは、apt-cache コマンドを実行すると Segmentation fault を起してしまうというものでした。例えば、apt-cache search gnuplot とすると出ます。そして、この不具合は PPC マシンに特異的です。こちらの方も調査していただけると助かります。しかし、この問題の原因も gnuplot と同様だとすると、PPC で動かないプログラムがまだ他にもあるかもしれませんね。

  – 吉田さん。gnuplot の件が無事に解決したとのご連絡をありがとうございます。apt-cache の segfault については他からもご指摘を頂いております。こちらは PPC の実機がないと対処しにくいものでもありますので、機会ができましたら対処させて頂きます。申し訳ありませんがご了承くださいませ。–Tkoba

- 吉田と申します。WorkShop をいつも重宝して使わせいただいています。最近、gnuplot をインストールしましたが、aquaterm に出力できません。そもそも、set terminal で出てくるリストの中に aqua がありません。バージョン 10.4 では問題なく使えていました。できましたら対処のほど、よろしくお願いします。 –110.132.80.241 2010 年 2 月 26 日 (金) 12:19 (UTC)

  – ご連絡をありがとう御座います。しかしながら当方では gnuplot で aquaterm が使えております。`$ rpm -q gnuplot aquaterm aquaterm-libs` の実行結果をお知らせください。–Tkoba

  – 吉田です。実行結果は以下のとおりになりました。

```
 $ rpm -q gnuplot aquaterm aquaterm-libs
gnuplot-4.3.0-10.5osx3
aquaterm-1.0.1-10.5osx0
aquaterm-libs-1.0.1-10.5osx0
```

  – rpm の DB 上では正しいようですね。念のために`$ sudo apt-get remove aquaterm` した後に改めて`$ sudo apt-get install gnuplot` してみてください。その際にエラーメッセージが出るようでしたらそのすべてをお知らせください。また、home に gnuplot の古い設定ファイルなどが残っていないかも確認してください。–Tkoba

  – 吉田です。remove し、再び install してみましたが、エラーメッセージは出ませんでした。また、古い設定ファイルもありませんでしたが状況は変らず、aquaterm には出力できませんでした。gnuplot の起動メッセージを以下に示します。何か手掛りになりますでしょうか。

```
$ /usr/osxws/bin/gnuplot
G N U P L O T
Version 4.3 patchlevel 0
```

[48] http://macwiki.sourceforge.jp/cgi-bin/wiki.cgi

```
last modified June 2008
System: Darwin 9.8.0
Copyright (C) 1986 - 1993, 1998, 2004, 2007, 2008
Thomas Williams, Colin Kelley and many others
Type ‘help‘ to access the on-line reference manual.
The gnuplot FAQ is available from
http://www.gnuplot.info/faq/
Send comments and help requests to  <gnuplot-beta@lists.sourceforge.net>
Send bug reports and suggestions to <gnuplot-beta@lists.sourceforge.net>
Terminal type set to 'wxt'
gnuplot>
```

– デフォルト起動で term が wxt になるには /.gnuplot などに set term wxt 等の記述がある筈なのですが。。一度、新規ユーザーを作成した上でそのユーザーで`$ /usr/osxws/bin/osxws-upgrade`して確認してみてください。–Tkoba

– 吉田です。新規ユーザーを作ってやってみましたが、状況は変わりません。そもそも set terminal しても aqua がリストに出てきませんので、これは gnuplot をコンパイルしたときに、aquaterm のライブラリがリンクされなかったなどの問題があったのではないでしょうか。因みに、gnuplot-4.2.6 と AquaTerm1.0.1.dmg を使って、コンパイルしてみましたが、これはうまく Aquaterm で表示することができました。ただ、gnuplot-4.3.0-10.5osx3 のライブラリ依存をみると、aquaterm はインクルードされているようですが。

```
 $ otool -L /usr/osxws/bin/gnuplot
/usr/osxws/bin/gnuplot:
/Library/Frameworks/AquaTerm.framework/Versions/A/AquaTerm (compatibility version 1.0.0, c
(Aquaterm のところだけ書き抜きました)
```

– 原因は何でしょうね。。

> これは gnuplot をコンパイルしたときに、aquaterm のライブラリがリンクされなかったなどの問題があったのではないでしょうか。

これはあり得ません。もしもそうであれば私の環境でも同様の結果になる筈です。 私の手元でも今一度 Leopard からクリーンインストールして確かめてみましたが、正常に aquaterm を使えています。 新規ユーザーでも問題が出るとすると、/.gnuplot は関係ない事になりますが、そうするとどうして term が wxt になるのか説明がつきません。 gnuplot-4.2.6 と AquaTerm1.0.1.dmg を入れていらっしゃるのが干渉しているのかもしれませんね。 OSXWS の環境での不具合発生であれば対処できますが、いろいろと環境を弄って（設定ファイルの編集は除く）いる場合には、基本的にユーザーサイドで問題を切り分けて戴くほかありません。–Tkoba

– 吉田です。別のマシンに Leopard をクリーンインストールし、Xcode と WorkShop を入れ、

```
$ sudo apt-get update
$ sudo apt-get install OSX-base
$ sudo apt-get clean
$ sudo apt-get install gnuplot
```

以上のように gnuplot をインストールしました。ここまでエラーはありませんでした。 しかし、gnuplot を起動してみると、

```
 $ gnuplot
     G N U P L O T
     Version 4.3 patchlevel 0
     last modified June 2008
     System: Darwin 9.8.0
     Copyright (C) 1986 - 1993, 1998, 2004, 2007, 2008
     Thomas Williams, Colin Kelley and many others
     Type ‘help‘ to access the on-line reference manual.
     The gnuplot FAQ is available from
                http://www.gnuplot.info/faq/
     Send comments and help requests to  <gnuplot-beta@lists.sourceforge.net>
     Send bug reports and suggestions to <gnuplot-beta@lists.sourceforge.net>
Terminal type set to 'wxt'
gnuplot> set terminal
Available terminal types:
             cgm  Computer Graphics Metafile
           corel  EPS format for CorelDRAW
          dpu414  Seiko DPU-414 thermal printer [small medium large]
            dumb  ascii art for anything that prints text
             dxf  dxf-file for AutoCad (default size 120x80)
           eepic  EEPIC -- extended LaTeX picture environment
             emf  Enhanced Metafile format
           emtex  LaTeX picture environment with emTeX specials
        epslatex  LaTeX picture environment using graphicx package
    epson_180dpi  Epson LQ-style 180-dot per inch (24 pin) printers
     epson_60dpi  Epson-style 60-dot per inch printers
     epson_lx800  Epson LX-800, Star NL-10, NX-1000, PROPRINTER ...
             fig  FIG graphics language for XFIG graphics editor
            gpic  GPIC -- Produce graphs in groff using the gpic preprocessor
         hp2623A  HP2623A and maybe others
          hp2648  HP2648 and HP2647
          hp500c  HP DeskJet 500c, [75 100 150 300] [rle tiff]
            hpdj  HP DeskJet 500, [75 100 150 300]
            hpgl  HP7475 and relatives [number of pens] [eject]
          hpljii  HP Laserjet series II, [75 100 150 300]
            hppj  HP PaintJet and HP3630 [FNT5X9 FNT9X17 FNT13X25]
          imagen  Imagen laser printer
           latex  LaTeX picture environment
              mf  Metafont plotting standard
             mif  Frame maker MIF 3.00 format
              mp  MetaPost plotting standard
         nec_cp6  NEC printer CP6, Epson LQ-800 [monocrome color draft]
         okidata  OKIDATA 320/321 Standard
```

```
             pbm  Portable bitmap [small medium large] [monochrome gray color]
            pcl5  HP Designjet 750C, HP Laserjet III/IV, etc. (many options)
             pdf  PDF (Portable Document File) file driver
        pdfcairo  pdf terminal based on cairo
        pngcairo  png terminal based on cairo
      postscript  PostScript graphics, including EPSF embedded files (*.eps)
         pslatex  LaTeX picture environment with PostScript \specials
           pstex  plain TeX with PostScript \specials
        pstricks  LaTeX picture environment with PSTricks macros
             qms  QMS/QUIC Laser printer (also Talaris 1200 and others)
           regis  REGIS graphics language
           starc  Star Color Printer
             svg  W3C Scalable Vector Graphics driver
    tandy_60dpi  Tandy DMP-130 series 60-dot per inch graphics
        tek40xx  Tektronix 4010 and others; most TEK emulators
        tek410x  Tektronix 4106, 4107, 4109 and 420X terminals
        texdraw  LaTeX texdraw environment
           tgif  TGIF X11 [mode] [x,y] [dashed] ["font" [fontsize]]
      tkcanvas  Tk/Tcl canvas widget [perltk] [interactive]
           tpic  TPIC -- LaTeX picture environment with tpic \specials
        unknown  Unknown terminal type - not a plotting device
          vttek  VT-like tek40xx terminal emulator
            wxt  wxWidgets cross-platform windowed terminal
            x11  X11 Window System
           xlib  X11 Window System (gnulib_x11 dump)
gnuplot>
```

このようにターミナルの一覧に aquaterm はでてきません。

```
gnuplot> set terminal aqua
                  ^
       unknown or ambiguous terminal type; type just 'set terminal' for a list
```

もちろん aqua を指定しても認識してくれませんので、今回もインストールされた gnuplot には aquaterm はインクルードされていないようです。 gnuplot のファイルサイズを以下に示します。Tkoba さんの環境でインストールされているものと同じでしょうか。

– もちろん同じです。同じバイナリー rpm なのですから寸分違わず同じものです。こちらで再現できる問題であれば対処のしようもあるのですが。それにしても不思議です。–Tkoba

```
$ ls -l /usr/osxws/bin/gnuplot
-rwxr-xr-x  1 root  wheel  2989316  7  8  2008 /usr/osxws/bin/gnuplot*
```

– 瀬戸です。これは不思議ですね。現在 10.6 のテストをしているので OSXWS10.5 の確認することができていません。AquaTerm が本来動くべきですが、もしよかったら解決するまでの間だけでも X11 を使ってみてください。私も gnuplot をよく使っていますが、出力先として AquaTerm でなく X11 にしています。というのも、コマンドキーを押しながらマウスで領域を選択すれば

ズームができたり、図のウインドウにフォーカスしている状態でキーボードショートカットがいろいろ使えて便利だからです（ “h” キーを入力することでショートカットの一覧が表示されます。 http://d.hatena.ne.jp/setoryohei/20091013 ）。ご参考まで。-Seto

- 吉田です。今日、もう 1 台の Mac に gnuplot をインストールしたところ、ちゃんと aquaterm に出力することができました。aquaterm が使えなかった 2 台の Mac はいずれも PowerPC マシンでしたが、うまくいったのは intel Mac でした。この結果が解決の糸口にはなりませんでしょうか。よろしくお願いします。
- 吉田さん、ご連絡をありがとう御座います。またレスポンスが遅くなり申し訳ありません。この件、問題は PPC でとのこと承知致しました。生憎と手元に PPC マシンがないので直ぐには修正できませんが、機会を見つけて出来るだけ対処したく思います。–Tkoba
- 修正した gnuplot-4.3.0-10.5osx4 を up しました。これでご確認戴けますでしょうか。–Tkoba
- 吉田です。修正版でばっちりうまく表示されました!感激です。これで、他の人に WorkShop を勧めることができます。本当にありがとうございます。

13.2 10.5-2 公開迄

- 大変遅くなりましたが、apt-0.5.15lorg3.94a-10.5osx1 にて http proxy の設定方法を追加して対処しました。ありがとう御座いました。–Tkoba 2009 年 12 月 10 日 (木) 06:30 (UTC)
- 出村さん、皆さん、ご連絡をありがとう御座います。また、レスポンスが非常に遅く申し訳有りませんでした。OSXWS としては /usr/osxws 以下しか弄らない事を基本としていますので、出村さんの解決方法を採用したいと思います。 –Tkoba 2009 年 8 月 1 日 (土) 03:16 (UTC)
- 私のところでも同様の現象が発生しました。どうやら sudo するときに環境変数が引き継がれなくなっているのが原因のようです。こちらでは sudo visudo で /etc/sudoers に

  ```
  Defaults        env_keep += "http_proxy ftp_proxy"
  ```

  と書くという手段で解決させました。 –133.19.126.5 2009 年 7 月 8 日 (水) 09:03 (UTC)
- 出村と申します。いつも大変便利に利用させていただいております。ありがとうございます。さて、firewall 越しで使っているのですが、最近になって、update できなくなってしまい困っていました。色々と試行錯誤してみたところ、apt.conf ファイルにプロキシ設定を直接書き込む事で、ちゃんと update やインストールができるようになりました。シェルの http_proxy 設定が、apt-get にはなぜか反映されないようです。ちなみに curl などのシェルコマンドには http_proxy 設定は反映されます。私のところの特殊事情かもしれませんが、念の為、ご報告しておきます。–144.213.253.16 2009 年 4 月 17 日 (金) 05:34 (UTC)

13.3 10.5-1 公開迄

- MacOSX-WS-10.5.1.dmg からインストールしたのですが, `$ sudo /OSXWS/usr/osxws/bin/apt-get update` を実行したところ，dyld: Library not loaded: /usr/osxws/lib/libapt-pkg-9.2.2.2.dylib と言われました。10.5.1 から /OSXWS 以下にインストールされるようになったかと思われますが，libapt-pkg-9.2.2.2.dylib がうまく参照されていないようです。やむなく, 10.5.0 に戻しました。 –119.30.220.250 2008 年 5 月 18 日 (日) 14:38 (JST)

  - ご連絡をありがとうございます。otool で libapt-pkg-9.2.2.2.dylib を調べてもロードされない理由がわかりませんでした。お手数をおかけしますが、表示されたエラーを全てご教示ください。–Tkoba
  - 質問者です。お返事，遅くなりすみません。すでに 10.5.0 に戻しているので，すべてのエラーメッセージかどうかはわかりませんが，メモに残っているのは，以下です。/OSXWS/usr/osxws 以下を丸ごと /usr/osxws にコピーしてしまえばよかったのでしょうか。–133.6.218.70 2008 年 5 月 27 日 (火) 18:37 (JST)

```
 $ sudo /OSXWS/usr/osxws/bin/apt-get update
dyld: Library not loaded: /usr/osxws/lib/libapt-pkg-9.2.2.2.dylib
 Referenced from: /OSXWS/usr/osxws/bin/apt-get
 Reason: image not found
Trace/BPT trap
```

  - 詳細をご連絡いただきありがとうございます。/OSXWS/ が root dir になっていたのですね。早速インストーラーを修正しました。–Tkoba

- apt-get dist-upgrade 後、gtk 関係のアプリケーションが動作しなくなりました。libpng のバージョン不整合のせいでした。X11 を Leopard デフォールトの 2.1.1 から、最新版 2.2.1 にアップグレードする必要がありました。最新版は、macosforge から入手できます。ご参考までに。- fujiik –130.87.234.184 2008 年 5 月 12 日 (月) 14:09 (JST)
  - 藤井さん、ご連絡を有り難うございます。先ほど改めてご連絡戴きましたが、この件は Leopard を Tiger から Upgrade した時に起きる問題のようですね。Leopard をクリーンインストールしていないユーザーさんには上記の方法で対処して戴く事にしましょう。–Tkoba
- OSXWS 10.4 を sudo apt-get remove OSX-system で削除後，ディスクイメージを入手し，10.5 をインストールしようとしたのですが，「インストール先の選択」で「このコンピュータのすべてのユーザが使用できるように」という部分が青くハイライトされるだけで，先に進めなくなりました．どうしたらよいでしょう？ PowerBook G4 17 (1.33GHz) で OSX 10.5 です． –133.5.165.65 2008 年 1 月 7 日 (月) 15:46 (JST)
  - ご報告をありがとうございます。一応 Leopard を clean install した iBook G4 1.25G で確認はしているのですが。update された Leopard の不具合か、PackageMaker の不具合か。。Xcode-3.1 待ちかもしれませんね。–Tkoba
  - 上で報告したものです．Xcode の upgrade をし忘れていました．Xcode 3.0 にし，OS も 10.5.1 にしたらすんなり行くようになりました．(10.5 ではダメでした) apt-get が /usr/osxws/bin 以下にあるのに気づかずに少し慌てましたが，とりあえず動いています．また何かあったら報告します．
- 本業が忙しく開発作業は正月休みになりそうです。X11 関連を整理する必要があり難儀しています。–Tkoba 2007 年 12 月 21 日 (金) 11:02 (JST)
- ありがとうございます。よろしくお願いいたします。–Tkoba 2007 年 12 月 9 日 (日) 18:15 (JST)
- では、時間を見つけて、どこかに MacOSX-Science メーリングリスト（仮称）を作ろうと思います。あわせて提案などがありましたら、気軽にお知らせください。よろしくお願いします。–ぜ 2007 年 12 月 6 日 (木) 08:03 (JST)

- 銭谷さん、いつも有益なご提案を有り難うございます。OSXWS の Leopard 版もチョコチョコと作りかけていて、先週 Carbon Emacs と mew を使えるところまで作業しました。UNIX のライブラリ類や C++ の UNIX2003 問題、リンカの仕様による Symbol not found の連発に対する対処を見つけたところです。これらの情報を早く MacWiki で還元したいと思っていますが、連日の出張でなかなかままなりません。ML のご提案、大賛成です。 –Tkoba 2007 年 12 月 4 日 (火) 08:18 (JST)

- Carbon Emacs Package, OSXWS 共通で、理工系ソフトのリリース情報等を扱う ML を作りませんか？　メールの方が相談しやすいこともありますし、Wiki にまとめ情報を反映させることでうまく補完できると思います。–ぜ 2007 年 12 月 3 日 (月) 11:23 (JST)

- 瀬戸さんコメントを有り難う御座います。.emacs 関連でもご提案を有り難う御座います。/Applications/OSXWS/ 以下に置く案はいいですね。採用しましょう。また、現在 /private/{etc,var} と/usr/local 以下に展開しているものを /usr/osxws（仮）以下に集約しておき、各ユーザーの .*shrc 中等で OSXWS の環境変数を用意して、その変数がない場合には OSXWS の設定を全て無視するように設計してみようかと思います。帰国して間が無く実際の作業はこれからですが、MacWiki に少しずつアイデアを書いていこうと思います。–Tkoba 2007 年 11 月 26 日 (月) 15:57 (JST)

- LaTeXiT 等の OSX アプリケーションですが、MacPorts に習って/Applications/OSXWS/などのディレクトリを作って、それ以下に置くようにすると、OSXWS 経由でインストールしたことがはっきりして良いと思うのですがどうでしょうか？ –Seto 2007 年 11 月 23 日 (金) 01:14 (JST)

- お二方コメントを有り難うございます。利用者さん「すぐ使える」のが OSXWS の最大の長所ですので勿論踏襲します。その一方で、銭谷さんのような単体ソフトの開発／配布者にもご満足頂けるディストリビューション開発もとても野心的で学術的に魅力的（成功すれば世界初！）です。来週まで米国出張中で殆どこちらで議論する余裕はありませんが、いろいろと議論して戴ければ幸いです。–Tkoba 2007 年 11 月 12 日 (月) 16:52 (JST)

- このパッケージにはものすごくお世話になっています．私は「すぐ使える」という側面を是非残していただきたいと思います．デフォルトで設定ファイルを廃止／簡素化するなら，現状の設定ファイル群を別パッケージとしてインストールできるようにしていただければ幸いです．–220.211.199.251 2007 年 11 月 10 日 (土) 07:38 (JST)

- 敢えて要望を挙げるとすれば、UNIX 設定ファイルの廃止／簡素化でしょうか。ディストリビューションとしてのカラーも大事ですが、汎用 UNIX パッケージを目指すとすれば、無色を目指すべきだと思います。 –ぜ 2007 年 11 月 9 日 (金) 23:13 (JST)

13.4　dot.emacs 10.5-1 公開迄

- 追加案

  - section 1 の変更案ですが、以下のように prefer-coding-system を設定するだけで、今と同じパラメータを一括して変更してくれるみたいです（default-coding-systems, terminal-coding-system, file-name-coding-system など)。 clipboard-coding-system の変更は含まれていないみたいですが、この設定は OSX のクリップボードではないみたいで、その効用もよく分からないのではずしてみました。utf-8 の文字コード自動判定を間違う事への対処法として、Seto の中で教えてもらいました。

```
 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;; Section 1 language configurations
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;; japanese settings for Carbon Emacs Package
(when
    (equal
     (substring (shell-command-to-string "defaults read -g AppleLocale") 0 2) "ja")
  (require 'utf-8m)
  (set-language-environment 'Japanese)
  (prefer-coding-system  'utf-8-unix)
  (load "menu-tree")
)
```

– gnuplot-mode は使われていますか？ 今は、正式版リリースに集中するということで、追加の検討はその後でも良いと思いますが。–Seto

  * いえ、ですが gnuplot の subpackage gnuplot-lisps として用意すべきですね。作業しました。–tkoba

– 昔 YaTeX のメーリングリストで広瀬さんに教えてもらった、どのバッファからでもファイル名を補完する関数です。とても便利なので、もしよかったらどこかに追加しませんか？

  * 今は共通のところに置く必要はないですね。–Seto

```
 (defun my-file-complete ()
  (interactive)
  (let*((p (point))
(s (save-excursion
     (skip-chars-backward "^ \t\n:;\"\'()<>")
     (point)))
(path (buffer-substring s p))
(dir (or (file-name-directory path) ""))
(file (file-name-nondirectory path))
(res (file-name-completion file dir)))
    (cond
     ((eq res t) (message "Sole completion"))
     ((eq res nil) (ding) (message "No match!"))
     ((string= file res)
      (message (mapconcat 'princ (file-name-all-completions file dir) " ")))
     (t
      (delete-region p s)
      (insert dir res)))))

(global-set-key [(alt k)] 'my-file-complete) ;;  このキーバインドだと nw モードで使えない
```

- コメント
  - 瀬戸さん銭谷さん、修正を有り難うございます。大変に時間が掛かりましたが emacs-lisps-1.3-10.5osx0.7 にて反映させました。 –Tkoba 2008 年 9 月 30 日 (火) 08:41 (UTC)
  - section 1 で，save-excursion は progn の間違いだと思うので変更しました。 –Seto 2008 年 8 月 6 日 (水) 09:05 (UTC)
  - dist-upgrade で gnuplot が更新された時に gnuplot-lisps もインストールされたと思っていましたが別だったのですね。混乱していました。先ほど、install gnuplot-lisps して、60gnuplot-init.el などがインストールされたことを確認しました。 –Seto 2008 年 7 月 9 日 (水) 05:59 (UTC)
  - YaTeX, Mew は OSX-Preferences と絡むので、これまで通りにしましょう。 –Tkoba 2008 年 7 月 9 日 (水) 01:24 (UTC)
  - gnuplot-mode の設定は/usr/osxws/etc/emacs-VER/site-start.d/60gnuplot-init.el で自動で読み込まれるので、ユーザーサイドで設定する必要はありません。–Tkoba 2008 年 7 月 9 日 (水) 00:56 (UTC)
  - gnuplot-mode ありがとうございます。設定ファイルを .emacs_osxws.el から読み込むようにしました。ついでに、YaTeX、Mew の設定もローカルから読み込むようにしましたが良いですか？ –Seto 2008 年 7 月 8 日 (火) 11:53 (UTC)
  - 透明パッチは基本的な設定の文法は v4 でも変わらないみたいですし、.emacs_osxws.el のみで良いでしょうね（値を変えるだけだから２重になってもよいかとも思いましたが)。とりあえず、グローバル共通はできる限りダイエットする方針が良いかなと思っています。 –Seto 2008 年 7 月 8 日 (火) 08:42 (UTC)
  - 私が 2008 春版で 22.2 に当てたのはこのバージョンでした。v4 の文法もご参考までに。–ぜ
  - 移して頂いた３つの部分は .emacs_osxws.el 内のみにしようかと思いますが如何でしょう？ –Tkoba 2008 年 7 月 8 日 (火) 00:55 (UTC)
  - Section 6 で、Color-thema の部分がコメント化されていますが、グローバル共通のところではユーザが編集できないので、.emacs_osxws.el の後半にコメント化した設定をいくつか付け足してみました。 –Seto 2008 年 7 月 7 日 (月) 23:19 (UTC)
  - Time Stamp を appearance setting から anything else に移動しました。（fundamental のほうかな？） –Seto 2008 年 7 月 7 日 (月) 23:13 (UTC)
  - インラインパッチと透明パッチですが、安定板正式リリースである emacs-22.2 には対応していないので、22.3 リリース後に採用致します。–Tkoba 2008 年 7 月 7 日 (月) 22:30 (UTC)
  - すみません。emacs-lisps-1.3-10.5osx0.5 で直しておきました。–Tkoba 2008 年 7 月 7 日 (月) 22:04 (UTC)
  - 先ほど dist-upgrade しましたが、/usr/osxws/share/emacs/site-lisp/emacs-lisps/osxws.el が無いみたいです。 –Seto 2008 年 7 月 7 日 (月) 12:32 (UTC)
  - 瀬戸さん、銭谷さん、皆さん、早速のコメントと修正を有り難うございます。今週中をメドに OSXWS のデフォルトを決定して、来週中に Leopard 版を正式リリースしようと思います。 –Tkoba 2008 年 7 月 7 日 (月) 02:33 (UTC)
  - alpha の設定ですが、アクティブ時には画面に集中できて、非アクティブ時にも文字が読めるということでデフォルトでは (100 80) くらいで良いと思うのですがどうでしょうか？ –Seto 2008 年 7 月 6 日 (日) 05:59 (UTC)

- keyboard-coding-system は指定してもしなくても、ことえりが on の時は japanese-shift-jis で、ことえり英字や直接入力 (U.S.) の時は mac-roman になるみたいですね。no window モードのときも、特に指定しなくても utf-8 になっていました。mule-cmds.el の中にいろいろ書かれているみたいですがややこしいです。language-environment も Japanese にしなくても、普通に使えるような気もしますし。 –Seto 2008 年 7 月 6 日 (日) 05:38 (UTC)
- インラインパッチは先週のを使って下さい。透明パッチは Emacs 23 互換の transparency4 をお薦めします。 –ぜ 2008 年 7 月 5 日 (土) 23:59 (UTC)
- utf-8 で問題なければ、そちらの方が良いと思います。（瀬戸さんのを真似て、ずっと sjis-mac にしていました） –ぜ 2008 年 7 月 5 日 (土) 23:54 (UTC)
- 133.5.165.65 です．OSX-Preferences が update されていて, /usr/osxws/share/OSXWS/jp/.emacs_osxws.el が修正されていることを確認しました．対応していただきありがとうございました．–218.41.210.88 2008 年 7 月 5 日 (土) 12:25 (UTC)
- set-keyboard-coding-system は、window モードのときに sjis-mac にする必要があるでしょうか？（utf-8 にして試してみたら問題なく入力できるようでしたので。） –Seto 2008 年 7 月 5 日 (土) 09:02 (UTC)
- .emacs_osxws.el などは、/usr/osxws/share/OSXWS/jp/ or en/ 以下にあります。 –Seto 2008 年 7 月 5 日 (土) 07:48 (UTC)
- 上の.yatex.el が古かったので、新しい方を貼付けました。–193.52.24.125 2008 年 7 月 5 日 (土) 07:42 (UTC)
- 先ほど OSXWS10.5 をクリーンインストールしましたが，現状では/.emacs_osxws.el に osxws-sec?.el などをロードする設定が書かれていないようです． –133.5.165.65 2008 年 7 月 4 日 (金) 08:28 (UTC)
- 改善提案はもちろん、英語の間違いもドンドンたたいてやってください。–Tkoba 2008 年 7 月 3 日 (木) 10:00 (UTC)

14 謝辞

MacOS X WorkShop は、以下の個人/団体 (順不同) に多大な御指南/御協力を戴いたり、
公開されているパッケージや議論を参考にさせて戴きました。
この場を借りて関係各位に感謝の意を表します。

藤井恵介さん	MacOS X WorkShop の下地を築いてくださいました。
土村展之さん	ptetex3 パッケージ
内山孝憲さん	Mxdvi とそのフォントパッケージ
齋藤修三郎さん	OTF パッケージ
銭谷誠司さん	CarbonEmacs パッケージ、 Mac Wiki
kenichi kikuchi さん	kinput2 ことえりパッチ
MacWiki	-
Project PINEAPPLE の皆さん	-
Vine Linux の皆さん	-

更新履歴

- Mon Aug 09 2010 KOBAYASHI Taizo
  - Version 10.5-3
  - 「過去の議論」追加記入
- Thu Dec 10 2009 KOBAYASHI Taizo
  - Version 10.5-2
  - 「過去の議論」追加記入
- Wed Oct 01 2008 KOBAYASHI Taizo
  - Version 10.5-1
- Wed Jul 03 2008 KOBAYASHI Taizo
  - 「過去の議論」に dot emacs 関連を追加記入
- Mon Dec 31 2007 KOBAYASHI Taizo
  - Version 10.4-3
  - 「過去の議論」追加記入
- Fri Sep 01 2006 KOBAYASHI Taizo
  - 「過去の議論」追加記入
  - 00-News を追加
- Fri Mar 03 2006 KOBAYASHI Taizo
  - `~/.emacs.el` の内容を `site-start.d` 内に移動
- Thr Feb 16 2006 KOBAYASHI Taizo
  - 「Remote Install」追加
- Mon Feb 06 2006 KOBAYASHI Taizo
  - 「過去の議論」追加記入

  Wed Feb 01 2006 KOBAYASHI Taizo
  - Version 10.4-2 for PowerPC/Intel
  - パッケージの大部分を Universal Binary 化
  - Intel Mac に対応
    binary package は i386, fat, ppc, noarch の組み合わせで行きます。
  - アンインストールをサポート
    以下のコマンドとそれに続く確認に了承すればアンインストールできます。

    ```
    $ sudo apt-get remove OSX-system
    ```

  - 英語環境を睨んで
    各ユーザーの dot files を/System/Library/User Template/Japanese.lproj/ から /Library/Application Support/OSXWS/jp/ へ移動。この結果 OSXWS インストール後に新規ユーザーを作成しても OSXWS とは切り離された素のユーザー環境が作られます。その新規ユーザーが OSXWS を利用したい場合は以下のコマンドを実行して dot files を整えてください。

    ```
    $ /usr/local/bin/osxws-upgrade
    ```

  - ## パッケージ追加情報
    * clamav, gmp
      最近迄猛烈に忙しく大変に遅くなりましたが ClamAV を packaging しました。daemon の扱いを MacOSX に準拠させ/Library/StartupItems?/clamav/ 以下に起動と停止のスクリプトをおきました。自動で clamd, freshclam が daemon として動きます。
    * cmucl, Maxima, Imaxima, clisp(test tree)
      デフォルトの lisp を cmucl に変更して Maxima を復活させました。 test tree に clisp と maxim-exec-clisp を置いておきますが clisp はメンテナンス対象外です。
    * fugu
      Cocoa で書かれた sftp client

* fftw3
  研究で必要になったから
* Desktop Manager
  一年以上利用しているのと source が tar ball で配布されたので packaging しました。
* ImageMagick
  やはり無いと不便であるから。
* synaptic
  これで GUI でパッケージ管理出来ます！ 関連して gtk2 も用意しました。起動 (mlterm 上) とマニュアルの表示は以下で行ってください。

```
$ sudo synaptic
$ open /usr/local/share/synaptic/html/index.html
```

* gcc-g95
  gcc-g77 と排他利用になりますが用意しました。

– 変更したパッケージ

* ispell から aspell
* LatexEquationEditor から LaTeXiT
  今後の発展を見込んで移行。ただし LatexEquationEditor のサポートも続けます。お好みに応じて使い分けてください。
* kterm から mlterm
  locale を ja_JP.UTF-8 へ変更するに伴い移行。
* eTeX-3 ベースに更新
  dvipdfmx と齋藤さんの OTF パッケージを自動で組み込む updmap-otf の調整に手間取ったが、漸く仕事で使える様になった。TeX 関連では、昨日瀬戸さんと議論の上、.emacs.el から yatex に関する記述を .yatex.el へ移した。
* ghostscript の version は 8.51 で組んでみることにした。ヒラギノをデフォルトにしました。
* less から lv

– 削除したパッケージ

* vim
  vim は multi_byte でコンパイルされている。`~/.vimrc` を弄って利用可
* bizp2
* freetype

• Wed Jul 20 2005 KOBAYASHI Taizo

– 各ページの文章の誤りを訂正。

• Fri Jul 15 2005 KOBAYASHI Taizo

– Version 10.4-1

– Tiger 版リリース

• Wed Jan 19 2005 KOBAYASHI Taizo

– 各ページの文章の誤りを訂正。

• Thu Nov 25 2004 KOBAYASHI Taizo

– Version 10.3-7

– Installer ver. 10.3f
rpm-4.3.2 をはじめとする全パッケージの更新。

– パッケージ追加情報

* Ngraph-6.3.30-10.3tk1

* xgraph-12.1-10.3tk1
  - xgraph11 から修正パッチを移植しました。百害あって一利無しのアニメーション機能は削除してあります。

– パッケージ更新情報

* OSX-Preferences-10.3-1tk12
  - .bashmyrc, .cshmyrc, .emacs.my.el, .zshmyrc の追加。.*myrc や .emacs.my.el を既に書いている人は以下のディレクトリから該当するファイルを参照して書き直して下さい。
  /System/Library/User Template/Japanese.lproj/

    * emacs-21.3.50-10.3tk11.5
      - CVS 20041123, inline_patch-20041101
    * OSX-Preferences-10.3-1tk16
      - fix osxws-upgrade script
    * kterm-6.2.0-10.3tk5
      - background:wheat, foreground:black に設定
    * kinput2-v3.1-10.3tk2
      - Cmd+Space で日英切り替え出来るように設定。modeLocation を kterm の左下に出るように設定。

- Thu Nov 11 2004 KOBAYASHI Taizo
  - rpm2html による RPM データベースのページを追加
  - 各ページの文章の誤りを訂正。
- Tue Oct 26 2004 KOBAYASHI Taizo
  - Installer の ReadMe.rtf, License.rtf を書き換え GPLv2 である事を明示。
  - Subsection 「ライセンス」追加
- Sun Oct 24 2004 KOBAYASHI Taizo
  - Version 10.3-6
  - Installer ver. 10.3d
    gettext, beecrypt, bzip2, OSX-Preferences 更新に伴う更新。
  - Section 「過去の議論」を追加。
  - パッケージ追加情報
    * w3m, w3m-el
      kterm 上で画像を表示する場合は w3m-img をインストールして下さい。
    * gtk+, glib, gdk-pixbuf, imlib, libungif
      w3m-img の為に導入。
    * OSX-keyring
      パッケージに gpg 署名をする為の鍵束。
    * kotonoko
      コトノコ [49] ver 1.0-beta26
    * vim
      vim-6.3.31 (huge,big,normal)
      kterm 上で利用する vim
      terminal での日本語入力はダメ。
  - パッケージ更新情報
    * emacs-21.3-10.3tk10
      - CVS 20041024, inline_patch 20041015
    * tetex-2.0.2-10.3tk5
      - remove TEXMF/dvips/base/config.ps
    * OSX-Preferences-10.3-1tk10
      - added rpm/BUILD dir
- Wed Oct 13 2004 KOBAYASHI Taizo
  - Version 10.3-5
  - Installer ver. 10.3c
    carbon-font.el の改訂に伴い Ayuthaya.ttf に関する記述を変更。
  - dot files の更新
    OSX-Preferences 更新の際に各ユーザーの設定ファイルを更新する osxws-upgrade scriput を同梱。
- Tue Oct 12 2004 KOBAYASHI Taizo
  - Version 10.3-4
  - .emacs.el の更新
    font-lock の導入と YaTeX 使用時の skk 環境の整備

[49] http://www.afternooncafe.jp/kotonoko/

- urw-fonts をインストールする際の warning についてを「10 既知の問題点」に追加

- Sun Oct 10 2004 KOBAYASHI Taizo
  - Version 10.3-3
  - Installer ver. 10.3b
    postintall script で無駄な .*.rpmorig を作らない様に修正
  - .emacs.el の更新
    bibtex-command ”jbibtex -kanji=euc” 追加（坂田君）
    ”set-terminal-coding-system” を ’utf-8 から ’euc-jp-unix へ変更
    terminal や kterm で -nw mode を利用できる様にしてみました。
    ただし、ことえりではなく SKK を利用して下さい。
  - Mxdvi-fonts の更新
    オリジナルの *.hqx を *.sitx で作り直しました。
- Thu Oct 07 2004 KOBAYASHI Taizo
  - Version 10.3-2
  - Installer ver. 10.3a
  - skk, skkdic, skktools 追加
  - OSX-Preferences-10.3-1tk5
    fixed typo in .bashrc
  - emacs-21.3.50-10.3tk7
    cvs-20041005
  - texmacro-otf
    updmap-otf ver. 0.5
    利用可能な font map のみを status で表示する様に修正

  ToDo
  - .emacs.el 関連の更なる調整。
    これは Mac Wiki で議論し乍ら発展させよう。
- Wed Sep 29 2004 KOBAYASHI Taizo
  - Version 10.3-1
  - 設定ファイル {/private/etc/something, $HOME/.something} の内容を追加。（Thanks. 銭谷さん）

  ToDo
  - .emacs.el 関連の更なる調整。
    これは Mac Wiki で議論し乍ら発展させよう。
- Thu Sep 23 2004 KOBAYASHI Taizo
  - Version 10.3
    公開版
  - apt-rpm tree の作成方法を追加

  ToDo
  - .emacs.el 関連の更なる調整。
    これは Mac Wiki で議論し乍ら発展させよう。
- Wed Sep 22 2004 KOBAYASHI Taizo
  - Version 1.0
  - installer の作成方法を追加

  ToDo
  - .emacs.el 関連の更なる調整。
    これは Mac Wiki で議論し乍ら発展させよう。
- Mon Sep 20 2004 KOBAYASHI Taizo

- Version 0.99

- installer の version を 10.3 へ変更。
- zsh の設定ファイルを追加（新山君）
- pTEX3.1.4, mendex-2.5a, etc..
- emacs Sep 19 CVS

ToDo

- .emacs.el 関連の更なる調整。

- Tue Sep 07 2004 KOBAYASHI Taizo
  - Version 0.9
  - スクリーンショット追加。
  - パッケージメモ以外はほぼ完成？

  ToDo
  - installer の version を 10.3 へ変更。
  - .emacs.el 関連の更なる調整。
- Mon Aug 23 2004 KOBAYASHI Taizo
  - 最初の版

索 引