第 6 章

Web＋DBシステムの構築とTEXシステムの更新

本章では、これまでの第 1〜5 章とは異なり、Linux 環境でのシステム構築を取り上げます。Windows 環境でのシステムでも用途によっては十分ですが、Linux 環境はコストフリーで導入できますし、安定稼働の実績が多くあり、サーバマシンとして最適です。

また、RDBMS として MySQL に代わり PostgreSQL を取り上げます。PostgreSQL は大変高度な RDBMS で、日本国内において広く使われています。

■本章の内容

6.1 Web ＋ DB システムの構築

Linux

まずはじめに、システムの基本となる Web ＋ DB サーバの構築を取り上げます。OS として使用する Linux のディストリビューションを選択してインストールし、Apache を導入設定する方法を解説します。

◉ Linux ディストリビューション

Linux には数多くのディストリビューションが存在し、いずれも同様に堅牢かつ高性能なサーバマシンを構築できます。以下に主な系統とその特徴を示します。

▼ Red Hat 系

- **Red Hat Linux**
  Red Hat, inc. の提供するディストリビューションで、オフィス用に最も広く普及し、新鋭のプログラムが早期にパッケージへ組み込まれます。
  現在ではフリーのものは Fedora Core として Fedora Project からリリースされています。
- **Vine Linux**
  日本語環境に重点を置いたディストリビューションで、pLATEX の導入の容易なことから、本格的な印刷環境を必要とするオフィスにも好適です。

- **Turbo Linux**
  多くの企業／教育機関において採用され、個人ユーザーにも広く利用されます。各所にユーザーフレンドリな配慮・工夫がなされています。
- **SuSE Linux**
  ヨーロッパを中心に流通し、オープンソースに対する姿勢が高く評価されます。
- **Miracle Linux**
  商用データベースで最高のシェアを誇る Oracle に最適化されたディストリビューションで、当然ながら、Oracle を核としたビジネスサーバの構築に最適です。

▼ Slackware 系

- **Slackware Linux**
  古くから提供され続けられているディストリビューションで、Linux の最も基本的な性格を残しています。
- **Plamo Linux**
  Slackware ベースでインストーラが日本語化され、ドキュメントがデータベース化されるなど、より使い易く改良されています。

▼ Debian 系

- **Debian Linux**
  GNU、Linux の精神をバックボーンとしてオープンな開発が積極的に行われ、非商用であるものの、かなり魅力的で内容の充実したディストリビューションに仕上げられています。
- **KNOPPIX Linux**
  CD-ROM のみでブートして動作するディストリビューションで、日本語にも対応しています。Linux をハードディスクにインストールすることが不要となり、ハードディスク上のブート領域や OS に手をつけずに済みます。

どのディストリビューションを導入するかは完全に好みの問題となります。一般的には Red Hat 系のディストリビューションが広く使われていますので、情報が豊富です。

本書では特定のディストリビューションでなければならないような内容は取り上げませんが、例として取り上げている画面図などでは、Vine Linux を使用しています。

◉ アーキテクチャ／ハードウェア／周辺機器

Linux の利用が可能なアーキテクチャ、ハードウェア、周辺機器は、次の Web ページをアクセスすると具体的に特定出来ます。近似した製品であれば、ほぼ間違いなく Linux のサーバマシンとして利用できます。

http://www.linux.or.jp/toc_link_hardware.html

◉ Linux のインストール

以降では具体的な例として、いわゆる Windows マシン上に Vine Linux をインストールした環境を例に解説を進めます。Red Hat 系のディストリビューションではほぼ同様の作業となるでしょう。また、ほかのディストリビューションでも同じように作業を進められることと思います。

Vine に限らず、最近のディストリビューションでは詳細なインストーラが付属しています。ディストリビューションを入手したら、インストーラの指示に従っていくだけで簡単にインストール作業は完了します。インストールが完了すると、そのときから快適なウィンドウシステムや日本語環境が提供されますので、Web + DB サーバのセットアップ作業を直ちに開始することが可能となります。

なお、Linux は PowerPC マシン（Macintosh）、Alpha マシン、SPARK/Ultra SPARK マシン（SUN）上においても動作し、ノート型パーソナルコンピュータも利用可能ですから、多くの場合、手持ちのコンピュータを Linux の Web + DB サーバへ容易に移行できます。

これらの詳細な情報については、各ディストリビューション／環境を対象とした書籍や Web ページをご覧ください。

◉ Linux のパッケージ管理

Vine Linux は Red Hat 系に属しますから、RPM（Red Hat Manager）形式でパッケージ管理が行われています。さらに、Debian 系の APT（Advanced Packaging Tool）もサポートされます。もちろん、Slackware 系を含む Unix 汎用な Tar ボール形式のパッケージも使用できます。次にそれらを簡単に説明します。

RPM

コマンド `rpm` を起動してパッケージの管理を容易に行えます。X-Window 上において GUI 操作でパッケージ管理を行えるアプリケーションも提供されています。

- **パッケージのインストール**
  `rpm -i` パッケージファイル名
- **パッケージのアップグレード**
  `rpm -U` パッケージファイル名
- **パッケージのアンインストール**
  `rpm -e` パッケージファイル名
- **パッケージの情報取得**
  `rpm -q` パッケージファイル名

APT

コマンド `apt` ではディスク上のみならずインターネットサイト上のパッケージも対象となります。RPM におけるパッケージ間の依存関係や競合の問題も解決できますから、よりインテリジェントです。

- **パッケージのインストール**
  `apt-get install` パッケージファイル名
- **パッケージのアップグレード**
  `apt-get upgrade` パッケージファイル名
- **パッケージのアンインストール**
  `apt-get remove` パッケージファイル名
- **パッケージの情報取得**
  `apt-cache gencache`　（情報収集）
  `apt-cache show` パッケージファイル名　（情報表示）
  `apt-cache showpkg` パッケージファイル名　（依存関係）
  `apt-cache search` パッケージファイル名　（検索）

Tar ボール

ソースファイルをコマンド `tar` でパッケージ化したもので、大抵の場合 `gzip` にて圧縮されています。拡張子を `tar.gz` とする圧縮されたパッケージファイルが多く流通しています。一般に、次の手順でパッケージがインストールされます。なお、アップグレード、アンインストールなどの管理コマンドは用意されていません。

Tar ボールにて提供されているパッケージをインストールする場合には、以下のような手順となります。

❶ パッケージファイルをディレクトリ `/usr/local/src` に配置
❷ パッケージファイルを解凍・展開してソースファイルを復元する
　　`tar xvfz` パッケージファイル名
❸ 展開先へ移動してスクリプト `configure` を実行し、コンパイルの実行環境を整える

```
cd 展開先のサブディレクトリ
./configure
```

❹ コマンド make によりソースファイルをコンパイルしてインストールを実行する

```
make
make install
```

◉ Apache ／ PHP ／ PostgreSQL の導入

本書で例として取り上げる PC/AT 互換機に Vine Linux をフルインストールされた環境では、Apache と PostgreSQL は RPM を利用してすでにインストールされています。したがって PHP をインストールしてセットアップすれば、必要なサーバ環境を構築できます。

ところが、Apache ／ PostgreSQL ／ PHP は、頻繁に更新されています。また Vine 以外の環境をお使いの方もいることでしょう。ここでは、Apache ／ PostgreSQL ／ PHP の最新パッケージをインターネットから入手してインストールする方法を説明します。

パッケージの入手先

Apache ／ PHP ／ PostgreSQL は、例えば次の Web ／ FTP ページをアクセスすることにより入手できます。

- **Apache**
  http://www.apache.org/dist/httpd/
- **PHP**
  http://jp.php.net/downloads.php
- **PostgreSQL**
  http://www.postgresql.org/mirrors-ftp.html

▼ パッケージの入手

Apache ／ PostgreSQL ／ PHP の最新パッケージは、Tar ボールの形式で配布されます。RPM 形式や APT 形式のパッケージは Tar ボール形式のものに遅れて配布されます。このため、Tar ボール形式のパッケージファイルを入手します。

なお、多くの方は GUI の Web クライアントや FTP クライアントを使用されておられるでしょうが、コマンド wget を使用してもパッケージファイルをインターネットから容易に入手できます。以下に Apache の例を示します。

```
# cd /usr/local/src/ ←インターネットから入手した全ファイルの格納先
# wget http://www.apache.org/dist/httpd/apache_1.3.31.tar.gz
```

パッケージ導入の前に

▼ Apache と PostgreSQL のユーザーアカウント確認

Vine Linux を使用している場合、フルインストールされた時点で Apache と PostgreSQL のユーザー「apache」「postgres」がすでに登録されていますが、Tar ボール形式のパッケージをインターネットから入手して最新の Apache ／ PostgreSQL を導入する場合、Apache と PostgreSQL のパッケージをアンストールしなければなりません。このアンインストール時にはユーザー「apache」「postgres」のアカウントも削除されてしまいます。

Tar ボール形式のパッケージをインストールした後も両アカウントを同様に利用されるでしょうから、Apache と PostgreSQL の RPM 形式パッケージをアンストールする前に、ユーザー「apache」「postgres」のアカウントをあらかじめ確認しておきます。

◀ パスワードファイルの登録確認

```
# cat /etc/passwd | grep apache          ⬇ログインできない
apache:x:48:48:Apache:/home/httpd:/bin/false
# cat /etc/passwd | grep postgre          ⬇ログインできる
postgres:x:26:26:PostgreSQL:/var/lib/pgsql:/bin/bash
```

◀ ユーザーグループファイルの登録確認

```
# cat /etc/group | grep apache
apache:x:48:
# cat /etc/group | grep postgres
postgres:x:26:
```

PostgreSQL のインストール

PHP は Apache のモジュールとして組み込まれます。したがって、PHP の導入は Apache の存在を前提とします。さらに、Web ＋ DB システムでは Apache が PostgreSQL と PHP により連携しますから、PHP の導入は PostgreSQL より後となります。そこで、Apache、PHP に先立って PostgreSQL を導入します。

▼ rpm パッケージの削除

RPM システムで導入されているパッケージファイルは、コマンド `rpm` で検索して確認し、その後に削除します。エラーが発生した場合には、ほかのファイルが表示されますので、そちらを先に削除することでこれを回避できます。

```
# rpm -qa | grep postgre ←検索
postgresql-7.2.3-0vl1
   ⋮
# rpm -e postgresql-tk-7.2.3-0vl1 ←削除
   ⋮
```

▼ 管理者ユーザー「postgres」のアカウント作成

PostgreSQL では、一般のシステムユーザーが管理者となります。その管理者となるユーザー「postgres」（グループ「postgres」）のアカウントが PostgreSQL の RPM パッケージとともに削除されましたので、新たに発行します。

ほかの Linux 環境に PostgreSQL を新規に導入する場合にも、新たにユーザー「postgres」を発行します。

```
# groupadd -g 26 postgres
# cat /etc/group | grep postgres
postgres:x:26:
# useradd -c PostgreSQL -d /home/postgres -g postgre
s -s /bin/bash -u 26 postgres
   ⋮
# cat /etc/passwd | grep postgres
postgres:x:26:26:PostgreSQL:/home/postgres:/bin/bash
```

▼ Tar ボールパッケージのインストール

次の手順で PostgreSQL の Tar ボールパッケージをインストールします。

```
# cd /usr/local/src
# ls postgresql-7.4.3.tar.gz
postgresql-7.4.3.tar.gz
# tar xvfz postgresql-7.4.3.tar.gz
   ⋮
# cd postgresql-7.4.3 ←解凍展開先へ移動
# ./configure ← configure の実行
   ⋮
# make
   ⋮
# make install
   ⋮
```

▼ 管理者ユーザー「postgres」の環境設定

ユーザー「postgres」のホームディレクトリに用意されたファイル.bash_profile を編集し、PostgreSQL のコマンドサーチパスと環境変数を設定します。次に編集後の内容を示します。

◀ 編集後の.bash_profile の内容

```
# .bash_profile

# Get the aliases and functions
if [ -f ~/.bashrc ]; then
        . ~/.bashrc
fi

# User specific environment and startup programs

# addpath $HOME/bin
BASH_ENV=$HOME/.bashrc
USERNAME=""

PATH="$PATH":/usr/local/pgsql/bin
export PGLIB=/usr/local/pgsql/lib
export PGDATA=/usr/local/pgsql/data

export USERNAME BASH_ENV PATH LESSOPEN
```

▼ ファイル所有権の属性変更

インストールした各ファイルの属性（ユーザーとグループ）を「postgres」用に変更します。なお、PostgreSQL のインストール先はディレクトリ/usr/local/pgsql となります。次の例においては、ユーザー「postgres」にファイル／ディレクトリの所有権を与え、PostgreSQL の実行ファイル等がほかのユーザーによって不用意に書き変えられないようにします。

```
# chown -R postgres:postgres /usr/local/pgsql
```

▼ データベースの作成

PostgreSQL の管理者ユーザー「postgres」となって、データベースを作成します。データベースの作成にはコマンド initdb を実行します。なお、データベースが作成されるディレクトリは環境変数 PGDATA で指定されます。

```
# su - postgres ←ユーザー「postgres」へ移行
$ initdb --encoding=EUC_JP --no-locale ← initdb を発行
```

ここで指定しているオプションの内容については、initdb --help とすることによって確認できます。

▼ PostgreSQL の起動／再起動／停止

PostgreSQL の管理者ユーザー「postgres」は、次のようにコマンド入力することで、PostgreSQL を起動し、再起動し、停止できます。

```
pg_ctl start     起動
pg_ctl restart   再起動
pg_ctl stop      停止
```

▼ パスワードの設定

最後に、PostgreSQL の管理者ユーザー「postgres」に対してパスワードを設定します。

```
# passwd postgres
Changing password for user postgres
New password: ←パスワードを入力（表示されない）
Retype new password:
passwd: all authentication tokens updated successfully
```

Apache のインストール

▼ RPM パッケージの削除

PostgreSQL の場合と同様に、RPM パッケージの各ファイルをコマンド rpm を使用して削除します。

```
# rpm -qa | grep apache
apache-1.3.27-0vl1
   ⋮
# rpm -e apache-manual-1.3.27-0vl1
   ⋮
```

▼ ユーザー「apache」のアカウント作成

ユーザー「apache」（グループ「apache」）のアカウントも Apache の RPM パッケージとともに削除されますので、そのアカウントを新たに発行します。RPM パッケージでインストールした Apache ではこのユーザー／グループの権限で Web コンテンツがアクセスされます。Apache デフォルトの設定では、なんの権限も付与されないシステムユーザー「nobody」（グループ「nobody」）のアカウントが使用されます。こちらのアカウントを使用する場合には、以下の作業は不要となります。

```
# groupadd -g 48 apache
# cat /etc/group | grep apache
apache:x:48:
# useradd -c Apache -d /usr/local/apache -g apache -
s /bin/faulse -u 48 apache
   ⋮
# cat /etc/passwd | grep apache
apache:x:48:48:Apache:/var/apache:/bin/false
# passwd apache
Changing password for user apache
New password:
Retype new password:
passwd: all authentication tokens updated successfully
```

▼ Tar ボールパッケージのインストール

次の手順で Apache の Tar ボールパッケージをインストールします。なおスクリプト `configure` の実行に際しては、`--enable-module=so` の引数を指定し、PHP のモジュールが DSO（*Dynamic Shared Object*）で Apache へ動的に組み込まれるようにします。

```
# cd /usr/local/src
# tar xvfz apache_1.3.31.tar.gz
   ⋮
# cd apache_1.3.31
# OPTIM="-O2" ./configure --enable-module=so
# make
   ⋮
# make install
   ⋮
```

▼ 設定ファイルの編集

`/usr/local/apache/conf` にある設定ファイル `httpd.conf` を編集し、Apache のユーザーとグループの指定部分を次のように「nobody」から「apache」に変更します*。なお、先頭に「`#`（シャープ記号）」が付されますと、その行はコメント行として取り扱われます。また、設定ファイル `httpd.conf` が文法的に正しいこと、モジュール `mod_so` が組み込まれていることもチェックしておきます。

* ユーザー「nobody」のままで使用する場合にはこの作業は不要です。

▶ httpd.conf の編集箇所

```
    ⋮
# User nobody
# Group nobody
User apache
Group apache
    ⋮
```

```
# /usr/local/apache/bin/apachectl configtest
Syntax OK          ⬆文法チェック　⬇モジュール組み込みのチェック
# /usr/local/apache/bin/httpd -l
```

▼ Apache の起動／再起動／停止

次のようにコマンド入力することで、Apache を起動し、再起動し、停止できます。なお、Apache を起動しましたら、インターネットブラウザからアクセスして動作を確認しておきます。

```
/usr/local/apache/bin/apachectl start     起動
/usr/local/apache/bin/apachectl restart   再起動
/usr/local/apache/bin/apachectl stop      停止
```

PHP のインストール

PHP のパッケージは DSO で Apache へ後付けのモジュールとして動的に組み込みます。この動的な組み込みは、モジュール mod_so が Apache へあらかじめ組み込まれていることが前提となります。

▼ Tar ボールパッケージのインストール

手順を次に示します。なお、スクリプト configure の実行時には PostgreSQL と Apache に関するすオプションの指定が追加されます。

```
# cd /usr/local/src
# tar xvfz php-4.3.8.tar.gz
    ⋮
# cd php-4.3.8
# ./configure --with-pgsql=/usr/local/pgsql --with-a
pxs=/usr/local/apache/bin/apxs --enable-mbstring --e
nable-trans-sid
    ⋮
# make
```

```
    ⋮
# make install
    ⋮
```

なお、configure のオプションについては、./configure --help とすることによって確認できます。

PHP ／ Apache の設定と動作確認

PHP の設定と Apache の再設定を行って動作を確認します。

▼ PHP の設定

カレントディレクトリに設定ファイルの雛型 php.ini-dist が用意されていますので、これをディレクトリ/usr/local/lib へ php.ini と名前を変えてコピーします。

```
# cp php.ini-dist /usr/local/lib/php.ini
```

▼ Apache の再設定

次のように Apache の設定ファイル httpd.conf を編集して PHP のモジュールを有効とし、かつ、拡張子 php が付されたファイルを PHP の処理対象とします。

```
    ⋮
LoadModule php4_module        libexec/libphp4.so
    ⋮
AddModule mod_php4.c
    ⋮
AddType application/x-httpd-php .php
#AddType application/x-httpd-php-source .phps
    ⋮
```

▼ Apache の再起動

次のように設定をチェックしてから、Apache を再起動します。

```
# /usr/local/apache/bin/apachectl configtest
Syntax OK
# /usr/local/apache/bin/apachectl restart ←再起動
```

▼ 動作チェック

Apache の公開ディレクトリ/usr/local/apache/htdocs/にスクリプトファイル phpinfo.php を作成し、これをインターネットブラウザでアクセスしてその表示から動作を確認します。

▶ **phpinfo.php**

```
<?php
   phpinfo();
?>
```

▼ CLI 版の動作チェック

PHP のバージョン 4.3 から、CLI（*Command Line Interface*）が自動的にインストールされていますので、その動作を確認します。

次のようなファイル hello.php を作成し、コマンドラインから PHP を実行します。

▶ **hello.php**

```
<?php
   print("Hello¥n");
?>
```

```
# php ./hello.php
Hello
```

なお、CLI 版 PHP の起動オプションは、-h を付与して実行することで確認できます。

```
# php -h
Usage: php [options] [-f]  [args...]
       php [options] -r  [args...]
       php [options] [-- args...]
  -a               Run interactively
  -c | Look for php.ini file in this directory
  -n               No php.ini file will be used
  -d foo[=bar]     Define INI entry foo with value 'bar'
  -e               Generate extended information for debugger/profiler
  -f         Parse .
  -h               This help
  -i               PHP information
  -l               Syntax check only (lint)
```

```
-m              Show compiled in modules
-r         Run PHP  without using script tags
-s              Display colour syntax highlighted source.
-v              Version number
-w              Display source with stripped comments and whitespace.
-z         Load Zend extension .

args...         Arguments passed to script. Use -- args when first argument
                starts with - or script is read from stdin
```

▼ CGI 版の PHP

CGI 版はソース展開ディレクトリ上のファイル/sapi/cgi/php として作成されています。このファイル php をディレクトリ/usr/local/bin にコピーすることでそのまま使えますが、CLI 版と同名になりますから、ファイル名を php-cgi などと変更してコピーします。コピー後に引数-v を指定して起動しますと、CGI 版のバージョン情報が表示出力されます。

◉ TEX システムの導入

dvipdfmx と TEX の更新

Vine Linux をフルインストールした状態では、Apache 等と同様、日本語対応のコンパイラ platex、PS 形式ファイルの生成プログラム dvips がすでに使用できる状態となっています。このため、特に追加の作業することなく、TEX 文書を PS 形式に変換し、そのファイルを PostScript 対応のプリンタへ送出することによりそのまま印刷を実行できます。

しかし、Apache 等の場合と同様に、TEX システムも更新されています。また Vine 以外の環境をお使いの方もいることでしょう。ここでは、最新の TEX システムを Tar ボールで導入する手順を説明します。

TEX システム更新

▼ ダウンロード

必要となるパッケージ／ファイルを以下にリストします。なお、これらのファイルはディレクトリ/usr/local/src/に格納します。

▼ ダウンロードするファイル

- **tetex-src-2.0.2.tar.gz** ..... teTEX ソースファイル
  http://www.ring.gr.jp/pub/text/CTAN/systems/unix/tetex/2.0/distrib/
- **tetex-texmf-2.0.2.tar.gz** ..... teTEX ライブラリファイル
  http://www.ring.gr.jp/pub/text/CTAN/systems/unix/tetex/2.0/distrib/

- **ptex-src-3.1.3.tar.gz** ..... pTEX ソースファイル
  http://www.ring.gr.jp/pub/text/TeX/ascii-ptex/tetex/
- **ptex-texmf-2.1.tar.gz** ..... pTEX ライブラリファイル
  http://www.ring.gr.jp/pub/text/TeX/ascii-ptex/tetex/
- **dvipsk-jpatch-p1.6.tar.gz** ..... パッチファイル
  http://www.ring.gr.jp/pub/text/TeX/ascii-ptex/dvips/
- **udvips-5.94a-p1.6.patch** ..... パッチファイル
  http://www.ring.gr.jp/pub/text/TeX/ptex-win32/utils/
- **dvipdfmx-20040411.tar.gz** ..... dvipdfmx
  http://project.ktug.or.kr/dvipdfmx/snapshot/current/

teTEX 関連と pTEX 関連のファイルは下記のサイトからも入手できます。

- **teTEX 関連**
  ftp://cam.ctan.org/tex-archive/systems/unix/tetex/2.0/distrib/
  ftp://dante.ctan.org/tex-archive/systems/unix/tetex/2.0/distrib/
  ftp://tug.ctan.org/tex-archive/systems/unix/tetex/2.0/distrib/
- **pTEX 関連**
  ftp://ftp.ascii.co.jp/pub/TeX/ascii-ptex/
  ftp://ftp.media.kyoto-u.ac.jp/TeX/ASCII-pTeX/
  ftp://bash.cc.keio.ac.jp/pub/TeX/ascii-ptex/
  ftp://ftp.cs.titech.ac.jp/pub/text/TeX/ASCII-pTeX/
  ftp://ftp.tut.ac.jp/TeX/ASCII-pTeX/
  ftp://ftp.u-aizu.ac.jp/pub/tex/ASCII-pTeX/
  ftp://ftp.foretune.co.jp/pub/TeX/ascii-ptex/

▼ 最新 teTEX のインストール

ダウンロードしたファイルが格納されているディレクトリ/usr/local/src に移動し、ディレクトリ/usr/local/tetex/share/texmf を作成し、ファイル tetex-texmf-2.0.2.tar.gz をディレクトリ/usr/local/tetex/share/texmf/へ解凍展開します。

さらに、ファイル tetex-src-2.0.2.tar.gz をディレクトリ/usr/local/src/で解凍展開し、これにより作成されたサブディレクトリ tetex-src-2.0.2 へ移動し、スクリプト configure を実行してから、make world とコマンド入力してコンパイルとインストールの処理を指示します。

```
# cd /usr/local/src/
# mkdir -p /usr/local/tetex/share/texmf
# tar xvzf tetex-texmf-2.0.2.tar.gz -C /usr/local/te
tex/share/texmf/
# tar xvzf tetex-src-2.0.2.tar.gz
# cd tetex-src-2.0.2/
```

```
# ./configure --prefix=/usr/local/teTeX --disable-mu
ltiplatform --without-xdvik --without-oxdvik
# make world
```

▼ **設定**

インストールが完了しましたら、プログラム latex、dvips がディレクトリ /usr/local/tetex/bin/にインストールされていますので、次のようにしてコマンドサーチパスを通します。正しくコマンドサーチパスが通っていることをコマンド which を発行して確認します。

◀ **コマンドサーチパス**

```
# export PATH=/usr/local/tetex/bin:$PATH
```

◀ **コマンドサーチパスを通す前**

```
# which latex
/usr/bin/latex
# which dvips
/usr/bin/dvips
```

◀ **コマンドサーチパスを通した後**

```
# which latex
/usr/local/tetex/bin/latex
# which dvips
/usr/local/tetex/bin/dvips
```

新規インストールのプログラム latex と dvips にコマンドサーチパスが正しく通っていることを確認しましたら、ホームディレクトリのファイル.bashrc にコマンドサーチパスの行 export PATH=/usr/local/tetex/bin:$PATH を追加しておきます。

pTEX と pLATEX 2ε の導入

ディレクトリ/usr/local/src/に移動し、ファイル ptex-texmf-2.1.tar.gz と ptex-src-3.1.3.tar.gz を展開します。続いてディレクトリ/usr/local/src/tetex-src-2.0.2/texk/web2c/ptex-src-3.1.3 へ移動し、スクリプト configure を実行し、コマンド make を発行してコンパイルとインストールの処理を行います。その結果、新規インストールの teTEX が日本語対応となります。

```
# cd /usr/local/src/
# tar xvzf ptex-texmf-2.1.tar.gz -C /usr/local/tetex
/share/texmf/
   ⋮
```

```
# tar xvzf ptex-src-3.1.3.tar.gz -C /usr/local/src/t
etex-src-2.0.2/texk/web2c/
   ⋮
# cd /usr/local/src/tetex-src-2.0.2/texk/web2c/ptex-
src-3.1.3
# ./configure euc ←使用する文字コードは euc
   ⋮
# make
   ⋮
# make install
   ⋮
# mktexlsr
```

dvips の導入

ディレクトリ/usr/local/src/に移動し、ファイル dvipsk-jpatch-p1.6.tar.gz を展開します。続いてディレクトリ/usr/local/src/tetex-src-2.0.2/texk/dvipsk でスクリプト configure を実行し、コマンド make を発行してコンパイルとインストールの処理を行います。さらに、パッチ当てを行ってプログラム dvips を修正します。

```
# cd /usr/local/src/
# tar xvzf dvipsk-jpatch-p1.6.tar.gz -C /usr/local/s
rc/tetex-src-2.0.2/texk/dvipsk/
   ⋮
# cd /usr/local/src/tetex-src-2.0.2/texk/dvipsk/
# patch -p1 < dvipsk-5.92b-p1.6.patch
   ⋮
# patch -p1 < udvips-5.94a-p1.6.patch
   ⋮
# ./configure --prefix=/usr/local/teTeX --disable-mu
ltiplatform
   ⋮
# make
   ⋮
# make install
   ⋮
# cd /usr/local/tetex/share/texmf/dvips/pstricks/
```

```
# patch /usr/local/src/tetex-src-2.0.2/texk/dvipsk/PS
Tricks.patch
   ⋮
```

日本語環境の整備

最後にファイル kanji.map を作成し、その後ファイル config.ps を修正します。なお、kanji.map を作成したらコマンド mktexlsr を発行して TEX にファイルの存在を認識させます*。

* kanji.map に限らず、ファイルの追加／移動をした場合には mktexlsr を発行します。

▼ kanji.map の作成

テキストエディタを起動して次の行を入力し、ディレクトリ/usr/local/tetex/share/texmf/dvips/config/に kanji.map として保存し、コマンド mktexlsr を発行します。

◀ kanji.map

```
rml             Ryumin-Light-H
rmlv            Ryumin-Light-V
gbm             GothicBBB-Medium-H
gbmv            GothicBBB-Medium-V
```

▼ config.ps の修正

テキストエディタで/usr/local/tetex/share/texmf/dvips/config にある config.ps を開き、任意の位置に次の行を追加して上書き保存します。

```
p +kanji.map
```

▼ 動作確認

簡単な TEX ソースファイル test.tex を作成し、次のようにプログラム platex、dvips を起動してその動作を確認します。

◀ test.tex

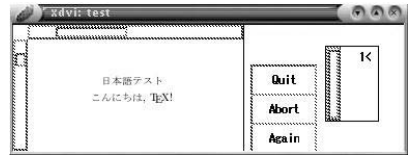
```
¥documentclass{jarticle}
¥begin{document}
日本語テスト¥¥
こんにちは，¥TeX !
¥end{document}
```

```
# platex test
This is pTeX, Version 3.14159-p3.1.3 (euc) (Web2C 7.4.5)
(./test.tex
pLaTeX2e <2001/09/04>+0 (based on LaTeX2e <2001/06/01> patch level 0)
```

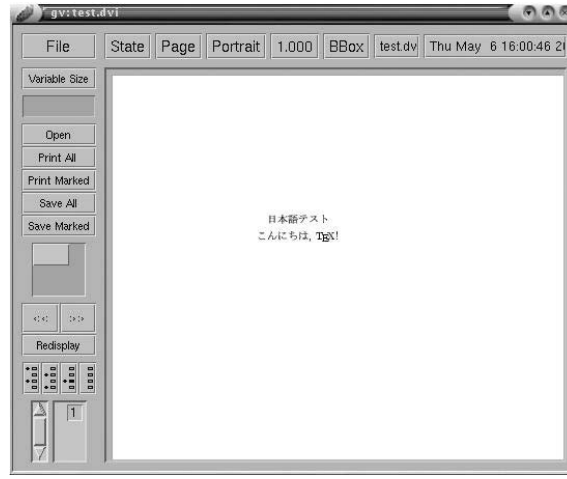
```
(/usr/local/tetex/share/texmf/ptex/platex/base/jarticle.cls
Document Class: jarticle 2001/10/04 v1.3 Standard pLaTeX class
(/usr/local/tetex/share/texmf/ptex/platex/base/jsize10.clo)) (./test.aux)
[1] (./test.aux) )
Output written on test.dvi (1 page, 324 bytes).
Transcript written on test.log.
# dvips -Ppdf test
This is dvips(k) 5.94a p1.6 Copyright 2003 ASCII Corp.(www-ptex@ascii.co.jp)
based on dvips(k) 5.94a Copyright 2003 Radical Eye Software (www.radicaleye.com)' TeX
 output 2004.05.06:1545' -> test.ps. [1]
```

DVI 形式のファイルおよび PS 形式のファイルが得られましたら、これらのファイルをビューア xdvi、ghostscript で表示させ、日本語が正しく表示されていることを確認します。次にそれらの様子を示します。

▶ **test.dvi を xdvi で表示**



▶ **test.ps を ghostscript で表示**



dvipdfmx のインストール

The DVIPDFMx Project のページから dvipdfmx のパッケージを入手してインストールします。

http://project.ktug.or.kr/dvipdfmx/snapshot/current/ にアクセスし、dvipdfmx-20040411.tar.gz を/usr/local/src にダウンロードします*。続いて dvi

* ファイル名はバージョンアップにより変更されます。

pdfmx-20040411.tar.gz を解凍展開し、スクリプト configure を実行し、コマンド make を発行してコンパイルとインストールの処理を行います。

```
# cd /usr/local/src/
# tar xzvf dvipdfmx-20040411.tar.gz
   ⋮
# cd dvipdfmx-20040411
# ./configure --prefix=/usr
   ⋮
# make
   ⋮
# make install
   ⋮
```

インストールが完了したら動作を確認します。platex の動作確認で生成された DVI 形式のファイル test.dvi を用意し、次のようにプログラム dvipdfmx を起動して PDF 形式のファイル test.pdf を生成します。

```
# dvipdfmx test
test.dvi -> test.pdf
[1]
8216 bytes written
```

生成された PDF 形式のファイルをビューア Adobe Reader で表示させますと、次のように日本語を含む PDF 形式のファイルが生成されたことを確認できます。

◀ 日本語 PDF ファイルを Adobe Reader で表示

このプログラム dvipdfmx によりますと、Windows の場合と同様に、パスワードロックや印刷禁止の機能を有する PDF 形式の文書ファイルを生成することが可能となります。