PHP Hypertext Preprocessor

Part - 4

インストール

第４部

Hypertext Preprocessor

Chapter-1 RPMによるインストール

Quick Start

Vine Linux 2.5をインストールしてあり、かつすぐにでも使い始めたいという人は、とりあえず以下の作業を行なうだけでApache＋PHP＋PostgreSQL環境を構築できます。//以下は説明のためのものですから、入力する必要はありません。

```
$ su -                                  // rootにスイッチ
# mount /mnt/cdrom                      // 付属CD-ROMをマウント
# cd /mnt/cdrom/arc/RPMS/i386
# /etc/init.d/httpd stop                // apacheを停止（失敗する場合もある）
# /etc/init.d/postgresql stop           // postgresqlを停止（同上）
# rpm -Uvh *
```

何もエラーが出なかった場合は、正常にインストールが完了しています。あとは「1.5 インストール後の設定」へ進んでください。

```
パッケージ xxx はすでにインストールされています
```

というメッセージが表示された場合は、

```
# rpm -Uvh --force *
```

を試してみてください。それでもうまくいかない、もしくは上記以外のエラーになった場合は、これ以降の記述を順序よくお読みください。

なお、各サーバソフトウェアは日々改訂されてバージョンが上がっていきます。機能アップの場合もありますが、最近はセキュリティホール対策用のバージョンアップが増えています。特に、インターネット上にサーバを公開している場合は早急なバージョンアップが求められます。筆者のほうでも積極的にサポート情報を公開していくつもりですから、巻末で紹介しているサポートページにも目を通すようにしてください。

1.1 はじめに

RPM（RedHat Package Manager）は、その名のとおりRedHat社により開発されたパッケージ管理システムであり、RedHat Linuxをはじめとして、同じ流れをくむVine Linux、TurboLinux、MiracleLinuxなどでも使われています。本書では、従来はソースからのインストールを推奨していましたが、コンパイルに失敗した場合の対処にはそれなりのスキルを要することから、今回の版よりRPMパッケージによるインストールを推奨することにしました。本書のリファレンスプラットフォームであるVine Linux 2.5を使っているのであれば、執筆時点で最新のRPMパッケージをインストールすることにより、比較的簡単に環境を構築できるでしょう。

RedHat系以外のプラットフォームを使っている人、およびLinux以外のOSを使っている人は、第2章にソースからのインストールについて記述してあるので、そちらを参照してください。Windowsへのインストールについても第3章で簡単に解説してあります。

1.2 RPMパッケージのファイル名

RPMパッケージファイルには、主に以下の2種類があります。例としてpostgresqlのパッケージを取り上げます。

```
postgresql-7.2.1-5vl3.i386.rpm
postgresql-7.2.1-5vl3.src.rpm
```

RPMパッケージのファイル名は、

```
source-ver-pkgver-{ARCH|src}.rpm
```

のような構成になっています。source-ver部分はオリジナルのソースアーカイブのものと同じです。pkgverは、そのディストリビューション（RedHatやVineなどの配布系）固有のバージョン番号です。ARCHはCPUのアーキテクチャで、i386（インテル系、およびその互換CPU用）のほかにもsparcやppcなどがあります。

ARCHがついているファイルは、そのCPU用にコンパイルされたバイナリの実行ファイルや初期化用スクリプトを固めたもので、実際のインストールの対象となります。以下、ARCHについてはi386を例にとりますが、それ以外のCPUを使っている人は、適宜読み替えてください。

*src.rpmは*i386.rpmを作るためのソースパッケージです。オリジナルの*.tar.gzや、それに対するパッチファイル、そしてビルドする際の要となる、依存関係やコンパイル方法などを記述したspecファイル（source名.spec）などが入っています。基本的に、あるディ

ストリビューション用に作られたRPMファイルは、依存しているパッケージがそれぞれ異なるため、別のディストリビューションでは使用できないことが普通です。そのような場合でも、ソースRPMを入手して、後述するRPMコマンドで各ディストリビューション用にリビルド（再コンパイル）すれば、使えるようになる場合もあります。

これらのファイルは通常rpmコマンドでインストールするためのものですが、とりあえずその中身を確かめたい場合、以下のコマンドで直接展開することもできます。

```
rpm2cpio パッケージファイル名 | cpio -idm
```

付属CD-ROMに収録してあるRPMパッケージはVineLinux 2.5用です。それ以外のバージョンまたはディストリビューションを使っている場合はうまくインストールできない場合があります。

1.3 RPMコマンド

RPMはかなり複雑なコマンド体系を持っています（詳細はman rpmを参照）が、必要最小限押さえておきたいコマンドについて、以下に紹介しておきます。なお、パッケージ名（postgresql）とパッケージファイル名（postgresql-7.2.1-5vl3.i386.rpm）、およびソースパッケージ名（postgresql-7.2.1-5vl3.src.rpm）は、それぞれ区別して読むようにしてください。先頭が # のものは、root 権限が必要なものです。

- **パッケージがインストールされているかどうかの確認**

```
$ rpm -qa|grep パッケージ名
```

- **インストールされているパッケージの概要説明表示**

```
$ rpm -qi パッケージ名
```

- **インストールされていないパッケージの概要説明表示**

```
$ rpm -qip パッケージファイル名
```

- **インストールされているパッケージの内容一覧表示**

```
$ rpm -ql パッケージ名
```

- **インストールされていないパッケージの内容一覧表示**

```
$ rpm -qlp パッケージファイル名
```

● パッケージのインストール（またはアップグレード）

```
# rpm -Uvh パッケージファイル名
```

● インストールされているパッケージの削除

```
# rpm -e パッケージ名
```

● ソースをリビルド（再コンパイル）してバイナリパッケージを作成する

```
# rpm --rebuild ソースパッケージ名
```

パッケージのインストールやリビルド時に「AはBに必要とされています（AやBはそれぞれ何かのパッケージ名）」などというメッセージが表示されてうまくいかない場合がありますが、これは各パッケージ間の依存関係のチェックにひっかかっていることを示します。依存関係を無視するオプション（--nodeps）もあるのですが、これを使ってしまうとパッケージ管理の利点が半減しますのでお勧めしません。インストールに必要なほかのパッケージを入れてから（あるいは必要なバージョンに上げてから）再度インストールするようにしてください。

リビルドする場合、正常終了すれば/usr/src/redhat/RPMS/i386配下（「redhat」の部分は各ディストリビューション名になる場合もあり）にバイナリパッケージが生成されるので、それらをrpm -Uvhでインストールしてください。なお、リビルドがうまくいかない場合の個別の対処については本書の範囲を超えるため、第2章のソースからのインストールで対処するようにしてください。

1.4 各パッケージのインストール

まずはインストール対象ファイルの準備をします。

```
$ su -                                  // rootにスイッチ
# mount /mnt/cdrom                      // 付属CD-ROMをマウント
# cd /mnt/cdrom/arc/RPMS/i386
```

念のためサービスを停止しておきます。サービスが起動していない場合は「失敗」と表示されます。

```
# /etc/init.d/httpd stop
# /etc/init.d/postgresql stop
```

各パッケージをインストールします。

```
# rpm -Uvh *
```

うまくいけば、これでインストールは終わりです。次のセクションに進んでください。「パッケージ XXX はすでにインストールされています」というメッセージが表示された場合は、

```
# rpm -Uvh --force *
```

を試してみてください。それでもうまくいかない場合は、前述のリビルドに挑戦するか、第2章に進んでください。また、現在使用しているディストリビューションのWebサイトで、本書推奨のものと同じバージョンのRPMパッケージがダウンロードできる場合があるので、確認してみてもよいでしょう。

1.5 インストール後の設定

MySQLがすでに動作している場合、不要であれば停止し、自動起動の設定も解除しておきます。

```
# /etc/init.d/mysql stop
# chkconfig mysql off
```

RPMパッケージでは必要最小限の設定がほぼ終わっているのですが、細かなチューニングが必要となる場合もあります。以下、本書のサンプルが実行できる必要最小限の設定について解説します。

RPMでインストールした場合、ファイルの配置は以下のようになっています。

/etc/httpd	Apacheの基準DIR
/etc/httpd/conf/httpd.conf	Apacheの設定ファイル
/home/httpd/html	Apacheのドキュメント用基準DIR（DocumentRoot）
/etc/php.ini	PHPの設定ファイル
/usr/lib/php4/pgsql.so	PHPのpostgresqlサポート用ライブラリ
/usr/bin/php	PHPのコマンドライン版
/var/lib/pgsql/data	PostgreSQL用DBの基準DIR
/var/lib/pgsql/data/postgresql.conf	PostgreSQLの環境設定ファイル
/var/lib/pgsql/data/pg_hba.conf	PostgreSQLのアクセス制御ファイル
/var/log/httpd/*.log	Apacheのアクセス/エラーログファイル

1.5.1 PostgreSQLの設定と動作確認

必要であればpostgresql.confを変更します。個々の設定に関する説明は付属CD-ROMの/docs/pgsql/runtime-config.htmlを参照してください。

OS起動時に自動起動を行ないたい場合は以下を入力します。

```
# chkconfig postgresql on
```

以下のコマンドによりPostgreSQLを起動します。データベースの初期化（initdb）が行なわれていなければ自動的に実行され、PostgreSQLサーバが起動します。

```
# /etc/init.d/postgresql start
```

ユーザpostgresで、以下のように表示されればインストールは完了です。

```
# su - postgres
bash-2.04$ psql -l
             データベース一覧
   名前      |  所有者    | エンコーディング
-----------+----------+------------------
 template0 | postgres | EUC_JP
 template1 | postgres | EUC_JP
(2 行)
```

1.5.2 Apacheの設定と動作確認

Apacheの設定ファイルはhttpd.confですが、PHPを動かすだけなら特に何も変更はいらないはずです。OS起動時に自動起動を行ないたい場合は以下を入力します。

```
# chkconfig httpd on
```

以下のコマンドによりApache Web Serverを起動します。

```
# /etc/init.d/httpd start
```

apacheが動いているマシンから適当なブラウザ使ってhttp://localhostにアクセスしてみます。テストページが表示されれば起動に成功しています。そのページの下のほうにある「マニュアルのCVS版」をクリックすると、Apacheに関する日本語のマニュアルを参照できます。

1.5.3 PHPの設定と動作確認

PHPをインストールすると、単独のコマンドとして動作する、いわゆるコマンドライン版と、Apacheのモジュールとして動作するDSO（Dynamic Shared Object）版の二つがインストールされます。一般ユーザで、**php -h**とタイプしてみましょう。ヘルプ画面が表示されれば、とりあえずphpのコマンドライン版はきちんと動いています。

次に、/home/httpd/html/phpinfo.phpというファイルを用意しましょう。中身は

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

の1行だけでOKです。これをhttp://localhost/phpinfo.phpで呼び出して、第1部の「5.4.4 phpinfo()」で示した画面が表示されればDSO版も動作しています。

今度はPostgreSQLへの接続を試してみましょう。/home/httpd/html/pgtest.phpというファイルを用意します。中身は

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

の1行だけです。これをhttp://localhost/pgtest.phpで呼び出してみて

```
Warning: Wrong parameter count for pg_connect() in /home/
httpd/html/pgtest.php on line 1
```

上記のように、パラメータの数が異なるというメッセージが表示されればOKです。もし

```
Fatal error: Call to undefined function: pg_connect() in
/home/httpd/html/pgtest.php on line 1
```

となる場合は、PostgreSQLのライブラリが読み込まれていないので、php.iniの内容を確認して

```
extension_dir = /usr/lib/php4
extension=pgsql.so
```

の2行が含まれるようにしてください。

http://localhost/manual/mod/mod_php4/ja/にアクセスすると、PHPの日本語マニュアル（UTF-8版）を読むことができます。付属CD-ROMの/docs/php/配下にもEUC版のマニュアルを収録しています。

Plamo Linuxでのインストール

Plamo Linux 2.2.5/2.2.6/3.0のユーザについては、桑村潤さんのご好意によりCD-ROMの/contrib配下にplamo用のバイナリパッケージを収録してあるので、こちらを使っていただくこともできます。このパッケージについての詳しい説明は、http://www.linet.gr.jp/~juk/plamo/を参照してください。これらに関する質問は、Plamo Linuxメーリングリスト<http://www.linet.gr.jp/~kojima/Plamo/ml.html>のほうにお願いします。

Hypertext Preprocessor

HP

Chapter-2 ソースからのインストール

2.1 はじめに

本章では、ソース（*.tar.gz形式のアーカイブ。いわゆるtar ball）から各ソフトウェアをインストールする方法について説明します。この方法であれば、本書の出版後に各ソフトウェアのバージョンが上がっても、バージョン番号を読み替える程度で常に最新版をダウンロードしてインストールすることができるようになるでしょう。なお、本章の内容はVine Linux 2.5で確認したものであり、ほかの環境でもまったく同じようになるとはかぎりません。その旨ご了承ください。

オープンソース系であればどんなソフトウェアでも同じですが、インストール途中で何かトラブルが起こったら（できればトラブルが起こる前に）、まずはソース配布物に含まれるREADMEやINSTALLといったテキストファイルを読んでください。ヒントになることが書かれています。

以下で紹介するコマンドの実行例では、シェルのプロンプトが#で終わっている場合はroot権限で実行することを、$で終わっている場合は特定の一般ユーザ権限で実行することを表しています。

ソースからのインストールを行なう場合、インストール完了後の各ファイルの配置は以下のようになります。

/usr/local/apache/	pacheの基準DIR
/usr/local/apache/conf/httpd.conf	Apacheの設定ファイル
/usr/local/apache/htdocs/	Apacheのドキュメント用基準DIR(DocumentRoot)
/usr/local/lib/php.ini	PHPの設定ファイル
/usr/local/bin/php	PHPのコマンドライン版
/usr/local/pgsql/	PostgreSQLの基準DIR
/usr/local/pgsql/data/	PostgreSQL用DBのDIR
/usr/local/pgsql/data/postgresql.conf	PostgreSQLの環境設定ファイル
/usr/local/pgsql/data/pg_hba.conf	PostgreSQLのアクセス制御ファイル
/usr/local/apache/logs/*.log	Apacheのアクセス／エラーログファイル

2.2 必要な開発ツールの確認

各ソフトウェアをインストールするにあたっては、以下に示すいくつかのツールの適切なバージョン（およびその同等品）がインストールされていることが前提となります。プラットフォームによっては、GNUツールをインストールする際に、頭にgをつける（たとえばmakeがgmakeとなるなど）場合もあるので、その場合は適宜読み替えてください。なお、この表における「確認バージョン」とは、このバージョンで確認を行なったことを示すもので、かならずしもこのバージョンでないとコンパイルできないというわけではありません。

プログラム名	確認バージョン	確認方法
make	3.79.1	make -v
gcc	2.95.3	gcc -v
readline	4.1	ls /usr/lib/libreadline.so*
history	4.1	ls /usr/lib/libhistory.so*
ld	2.11	ld -v
bison	1.28	bison --version
flex	2.5.4	flex --version

2.3 インストール済みパッケージの削除

この節の話題はRedHat系のLinuxを使っている人だけに関係する話です。それ以外の人は読み飛ばしてもらってかまいません。

RedHat系を使っている場合は、混乱を避けるために、あらかじめインストールされているRPMパッケージを削除します。RPMとソースからのインストールを混在させるのは、混乱の元になるためサポート対象から除外させていただきます。削除に先立って、必要であればデータのバックアップを行なっておいてください。

まず、該当プログラムが動いていれば停止します。動いていなければ「失敗」と表示されます。

```
# /etc/init.d/postgresql stop
# /etc/init.d/httpd stop
```

以下のコマンドで、既存パッケージの削除を行ないます。

```
# rpm -qa|grep mod_ssl|xargs rpm -e
# rpm -qa|grep apache|xargs rpm -e
# rpm -qa|grep postgresql|xargs rpm -e
# rpm -qa|grep php|xargs rpm -e
```

「XXXが削除できません。ディレクトリが空ではありません。」という警告が出た場合、それらのディレクトリの内容を確認のうえ、各ディレクトリを手動で削除しておいてください。

2.4 PostgresSQLのインストール

2.4.1 専用アカウントの作成

PostgreSQLはセキュリティ保護のために、root権限では起動しないようになっています。また、インストールやデータベースの作成といったPostgreSQLの運用管理を行なう場合にも専用のアカウントで行なうことが推奨されています。ここでは慣例にしたがって、postgresという一般ユーザをadduserコマンドで作成し、このアカウントでインストール作業のほとんどを行なっています。postgresアカウントがすでに存在する場合は、それをそのまま使えます。

```
root@star:~# adduser postgres
root@star:~# passwd postgres
Changing password for user postgres
New UNIX password:  (パスワード)
Retype new UNIX password:  (パスワード)
passwd: all authentication tokens updated successfully
```

2.4.2 作業用ディレクトリの作成

ソースの展開用およびインストール用としてディレクトリを作成します。

```
root@star:~# mkdir -p /usr/local/src/postgresql-7.2.1
root@star:~# chown postgres /usr/local/src/postgresql-7.2.1
root@star:~# mkdir -p /usr/local/pgsql
root@star:~# chown postgres /usr/local/pgsql
```

2.4.3 CD-ROM のマウント

本書に付属しているCD-ROMをマウントします。Vine Linuxでは以下の方法でマウントできますが、ほかのプラットフォームでは異なる場合もあります。

```
root@star:~# mount /mnt/cdrom
```

これ以降、CD-ROMの基点は/mnt/cdrom であることを前提に作業します。必要に応じて適宜読み替えてください。

2.4.4 ソースの展開とパッチの適用

これ以降はユーザpostgresで作業を行ないます。

```
root@star:~# su - postgres
postgres@star:~$ cd /usr/local/src/
postgres@star:/usr/local/src$ tar xvzf /mnt/cdrom/arc/SOURCES/postgresql-
7.2.1.tar.gz
postgres@star:/usr/local/src$ cd postgresql-7.2.1
postgres@star:/usr/local/src/postgresql-7.2.1$ zcat /mnt/cdrom/arc/SOURCES/
postgresql-7.2.1.nlspatch.diff.gz | patch -p1
```

最後の2行はオプションです。このパッチにより、会話型SQLクライアントであるpsqlなどで日本語メッセージやヘルプが使えるようになります。

2.4.5 configureの実行

ほかの多くのオープンソース・ソフトウェアと同様に、PostgreSQLでもconfigureスクリプトが用意されています。これは各種プラットフォーム間の差異を吸収し、コンパイル(make)の際に必要となる適切なMakefile(GNUmakefile)を自動生成してくれるものです。

```
postgres@star:/usr/local/src/postgresql-7.2.1$ ./configure --enable-multi
byte=EUC_JP --enable-nls --enable-syslog
```

2.4.6 コンパイルおよびインストール

コンパイルを行ないます。マシンスペックにより数分から数十分かかります。

```
postgres@star:/usr/local/src/postgresql-7.2.1/src$ make
(中略)
All of PostgreSQL is successfully made. Ready to install.2.1.6
postgres@star:/usr/local/src/postgresql-7.2.1/src$ make install
postgres@star:/usr/local/src/postgresql-7.2.1/src$ cd ../doc/
postgres@star:/usr/local/src/postgresql-7.2.1/doc$ make install
```

最後の2行は、ドキュメントをインストールしない場合は不要です。付属CD-ROMのdocs配下に日本語マニュアルを収録してあります。

2.4.7 環境設定

管理者ユーザであるpostgres、およびPostgreSQLを利用するすべてのユーザについて、コマンドサーチパスと環境変数の設定を行ないます。シェルとしてbash を使っている場合は ~/.bashrcに、csh/tcshを使っている場合は ~/.cshrcに以下の設定を追加します。

●~/.bashrc への追加設定

```
PATH="$PATH":/usr/local/pgsql/bin
export POSTGRES_HOME=/usr/local/pgsql
export PGLIB=$POSTGRES_HOME/lib
export PGDATA=$POSTGRES_HOME/data
export MANPATH="$MANPATH":$POSTGRES_HOME/man
export LD_LIBRARY_PATH="$LD_LIBRARY_PATH":"$PGLIB"
```

●~/.cshrc への追加設定

```
set path = ($path /usr/local/pgsql/bin)
setenv POSTGRES_HOME /usr/local/pgsql
setenv PGLIB $POSTGRES_HOME/lib
setenv PGDATA $POSTGRES_HOME/data
if ($?MANPATH) then
    setenv MANPATH "$MANPATH":$POSTGRES_HOME/man
else
    setenv MANPATH $POSTGRES_HOME/man
endif
if ($?LD_LIBRARY_PATH) then
    setenv LD_LIBRARY_PATH "$LD_LIBRARY_PATH":"$PGLIB"
else
    setenv LD_LIBRARY_PATH $PGLIB
endif
```

source ~/.bashrc（またはsource ~/.cshrc）を実行して、設定を反映させます。このあと実際にいくつかのプログラムを起動します。各プログラムには多くの引数が用意されていますが、ここでは動作確認に必要な最小限の使用にとどめます。

2.4.8 データベースの初期化

データベースを使用する前に、データベースの入れ物を用意する作業を行ないます。具体的には、initdbコマンドにより$PGDATAディレクトリに管理用データベースとユーザデータベースのひな型をコピーします。initdbを起動したユーザ（通常はpostgres）がそのデータベース領域の所有者となります。このあとユーザが作成するデータベースやテーブルの実体は、特に指定しないかぎり$PGDATA/base配下に置かれます。

```
postgres@star:~$ initdb
```

$PGDATA/postgresql.confというファイルが作成され、これがPostgreSQLサーバの設定ファイルになります。設定項目は多岐にわたるため、詳細については付属CD-ROMの/docs/pgsql/runtime-config.htmlを参照してください。よく使う設定を以下に示します。

postgresql.confの設定項目

設定ディレクティブ	設定値
tcpip_socket	false：ほかのホストからの接続を許可しない true ：許可する（pg_hba.confの設定も必要）
silent_mode	false：デバッグメッセージなどを表示する true ：制御端末を切り離す（メッセージを表示しない）
syslog	0 ：syslogへの出力を行なわない 1 ：メッセージをsyslogと標準出力の両方に出力する 2 ：メッセージをsyslogのみに出力する

2.4.9 postmasterの起動

PostgreSQLのデーモンプログラムであるpostmasterを起動します。postmasterコマンドで有効なオプションを以下に示します。

```
Usage: postmaster [options]
-B nbufs ……共有バッファ数のセット
-D datadir ……データディレクトリのセット
-S ……サイレントモード（ttyから切り離す）
-a system ……この認証システムを使用する
-b backend ……指定したバックエンドサーバを使用する
-d [1-5] ……デバッグレベルの指定
-i ……UNIXソケットに加えTCP/IPソケットからの接続を受けつける
-N nprocs ……バックエンドの最大数（1..1024, デフォルト32）
-n ……異常終了後に共有メモリを再初期化しない
-o option ……各バックエンドサーバへ'option' を渡す
-p port ……接続を待つポート番号
-s ……バックエンドのうちひとつが死んだらほかのバックエンドにもSIGSTOPを送る
```

実際には、postmasterプログラムの起動スクリプトpg_ctlを使って起動すると便利です。pg_ctlの起動オプションを以下に示します。

```
pg_ctl [-w][-D DIR][-p PATH] [-o "OPTS"] start
pg_ctl [-w][-D DIR][-m s|f|i] stop
pg_ctl [-w][-D DIR][-m s|f|i] [-o "OPTS"] restart
pg_ctl [-D DIR] status

-h|--help ····· ヘルプを表示する
-w ····· 既存のpostmasterプロセスが終了するのを待ってから起動する
-D DIR ····· $PGDATAディレクトリを指定する
-p PATH ····· postmasterが存在するパスを指定する
-m s|f|i ····· シャットダウン時のモード指定
s(mart) ····· スマートなシャットダウン(SIGTERMを送る)
f(ast) ····· 早いシャットダウン(SIGINTを送る)
i(mmediate) ····· 即時シャットダウン(SIGQUITを送る)
-o "OPTS" ····· postmasterプログラムに渡す起動時オプションを指定する
start ····· postmasterを起動する
restart ····· postmasterを再起動する
stop ····· postmasterを終了する
status ····· postmasterの動作状況を得る
```

pg_ctlコマンドでpostmasterを起動します。

```
postgres@star:~$ pg_ctl -w start
Waiting for postmaster starting up...done.
postmaster successfully started up.
```

2.4.10 アクセス制御

postgresql.confのtcpip_socketをtrueにすれば別ホストからの接続を許可するようになりますが、さらにアクセス制御ファイル/usr/local/pgsql/data/pg_hba.conf を適切に設定する必要があります。別ホストからの接続を行なわない場合は、デフォルトのままでかまいません。

とりあえずどこからでもアクセスできるようにしたい場合は、pg_hba.confの末尾に

```
host    all      0.0.0.0      0.0.0.0      trust
```

を追加してからpg_ctl restartでPostgreSQLを再起動してください。trustをpasswdもしくはcryptに代えると、パスワード認証ができるようになります。

● passwd

普通のファイルにユーザ名とパスワードを書いておく方法です。

● crypt

PostgreSQLのデータベースでパスワードを管理する方法。いずれも、詳細は付属CD-ROMの/docs/pgsql/auth-methods.htmlを参照してください。

2.5 Apacheのインストール

ここから先は再びroot権限で作業を行ないます。

```
root@star:~# cd /usr/local/src/
root@star:/usr/local/src# tar xvzf /mnt/cdrom/arc/apache_1.3.26.tar.gz
root@star:/usr/local/src# cd apache_1.3.26/
root@star:/usr/local/src/apache_1.3.26# OPTIM="-O2" ./configure --enable-module=so
(中略)
root@star:/usr/local/src/apache_1.3.26# make
(中略)
root@star:/usr/local/src/apache_1.3.26# make install
(中略)
+--------------------------------------------------------+
| You now have successfully built and installed the      |
| Apache 1.3 HTTP server. To verify that Apache actually |
| works correctly you now should first check the         |
| (initially created or preserved) configuration files   |
(中略)
| Thanks for using Apache.      The Apache Group         |
|                               http://www.apache.org/   |
+--------------------------------------------------------+
```

このように表示されればApacheのインストールは成功です。ではApacheを起動してみましょう。

```
root@star:/usr/local/src/apache_1.3.26# /usr/local/apache/bin/apachectl start
/usr/local/apache/bin/apachectl start: httpd started
```

手元のブラウザからhttp://localhost/にアクセスしてみましょう。テストページが表示されたらインストール成功です。

2.6 PHPのインストール

PHPには、Apacheとともに使うタイプのDSO版と、単独で使うコマンドライン版があります。両方ともインストールしておけば、何かと便利です。

2.6.1 PHP(DSO版)のインストール

インストールの手順は以下のとおりです。

```
root@star:/usr/local/src# tar xvzf /mnt/cdrom/arc/SOURCES/php-4.2.2.tar.bz2
root@star:/usr/local/src#  zcat  /mnt/cdrom/arc/SOURCES/php-4.2.2-multibyte.
patch.gz | patch -p0
root@star:/usr/local/src# cd php-4.2.2
root@star:/usr/local/src/php-4.2.2#  ./configure  --enable-mbstring  --enable-
mbstr-enc-trans --enable-mbregex --enable-zend-multibyte --with-pgsql --with
-apxs=/usr/local/apache/bin/apxs --without-gd
(中略)
root@star:/usr/local/src/php-4.2.2# make
(中略)
root@star:/usr/local/src/php-4.2.2# make install
(中略)
```

次に、/usr/local/apache/conf/httpd.confの774行目付近にあるAddTypeの次の行に

```
AddType application/x-httpd-php .php
```

という行を追加します。PHP4におけるスクリプトの拡張子は.phpとなります。最後にApacheを再起動します。

```
root@star:~# /usr/local/apache/bin/apachectl restart
/usr/local/apache/bin/apachectl restart: httpd restarted
```

PHP のサンプルを表示することによりPHPの動作を確認します。「1.5.3 PHPの設定と動作確認」を参照して、/usr/local/apache/htdocs/phpinfo.phpを作成し、ブラウザからhttp://localhost/phpinfo.phpにアクセスして表示を確認します。

PHPにはphp.iniという設定ファイルが存在します(存在しない場合はデフォルトの動作をします)。PHPのデフォルトの動作を変更したい場合は、以下のように設定ファイルを作成しておいてください。

```
root@star:~# cp /usr/local/src/php-4.2.2/php.ini-dist /usr/local/lib/php.ini
```

php.ini についての内容は、第3部「9.1 PHPオプション」を参照してください。

2.6.2 PHP（コマンドライン版）のインストール

直前に使用した展開済みソースをそのまま使用します。

```
root@star:# cd /usr/local/src/php-4.2.2
root@star:/usr/local/src/php-4.2.2# make distclean
root@star:/usr/local/src/php-4.2.2#  ./configure  --enable-mbstring  --enable-
mbstr-enc-trans  --enable-mbregex  --enable-zend-multibyte  --with-pgsql  --
enable-force-cgi-redirect --without-gd
（中略）
root@star:/usr/local/src/php-4.2.2# make
（中略）
root@star:/usr/local/src/php-4.2.2# make install
（中略）
```

/usr/local/bin/php -hでヘルプが表示されれば、インストールは完了です。必要であれば/usr/local/binにパスを通しておいてください。

2.7 自動起動の設定

OSの起動時にPostgreSQLおよびApacheを自動起動させるには、/etc/rc.d/rc.localの末尾に以下の記述を追加します。

```
if [ -x /usr/local/pgsql/bin/pg_ctl ]; then
  su postgres -c "/usr/local/pgsql/bin/pg_ctl -D /usr/local/pgsql/data start"
fi
if [ -x /usr/local/apache/bin/apachectl ]; then
  /usr/local/apache/bin/apachectl start
fi
```

このあとリブートを行ない、再起動後正常にpostmasterとhttpdが起動しているのを確認しておきましょう。

Hypertext Preprocessor

Chapter-3 Microsoft Windows環境へのインストール

PHPは、Linuxのような Unix互換環境のみならず、Microsoft Windows（以下、Windows）環境でも動作します。本章では、Apache＋PHPをWindows 2000 Professional SP2日本語版にインストールする方法について解説します。Windowsのシステムディレクトリはc:¥WINNTを想定しています。環境が異なる場合は、適宜読み替えてください。Windows 95/98/Meにインストールすることも可能ですが、OS自体の制限により、安定動作は期待しないほうがよいでしょう。

インストール用の各種アーカイブは、CD-ROMの¥ARC¥WINDOWS¥配下に収録してあります。本書推奨の手順では、インストール完了後の各ファイルの配置は以下のようになります。

C:¥Program Files¥Apache Group¥Apache¥	Apacheの基準DIR
C:¥Program Files¥Apache Group¥Apache¥conf¥httpd.conf	Apacheの設定ファイル
C:¥Program Files¥Apache Group¥Apache¥htdocs¥	Apacheのドキュメント用基準DIR（DocumentRoot）
C:¥WINNT¥php.ini	PHPの設定ファイル
C:¥PHP¥php-cli.exe	PHPのコマンドライン版
C:¥Program Files¥Apache Group¥Apache¥logs¥*.log	Apacheのアクセス/エラーログファイル

3.1 Apacheのインストール

まずApache-1.3.26をインストールします。apache_1.3.26-win32-x86-no_src.msi（Windowsインストーラ用）を、エクスプローラなどからダブルクリックして起動します。Windowsインストーラが起動するので、指示に従ってインストールします。

デフォルトでは、Apacheはc:¥Program Files¥Apache Group¥Apache配下にインストールされ、インストールが完了すると、サービスとして自動的に起動します。ブラウザからhttp://localhostにアクセスして、「あなたの予想に反して～」の画面が表示されれば成功です。Apacheの開始と停止は「コントロールパネル」－「管理ツール」－「サービス」で行ないます。各種設定は、c:¥Program Files¥Apache Group¥Apache¥conf¥httpd.confを修正することで行ないます。

3.2 PHPのインストール

次にPHP 4.2.2をインストールします。インストール先のディレクトリ（ここではc:¥phpとします）は前もって作成しておいてください。

php-4.2.2-Win32.zipを適当な場所に展開[*1]し、その中身（php-4.2.2-Win32ディレクトリの下）をc:¥php配下に移動します。そのあとphp-4.2.2-Win32-mb-1.1.lzh（マルチバイト対応差分）を展開し、同様にphp-4.2.2-Win32-mbディレクトリ配下をc:¥phpに上書きで展開[*1]します。展開後のディレクトリは以下のようになります。

```
c:¥php¥
    browscap¥
    dlls¥
    extensions¥
    mibs¥
    pdf-related¥
    pear¥
    sapi¥
```

最後にc:¥php¥php4ts.dllをc:¥WINNT¥SYSTEM32にコピーし、設定ファイルc:¥php¥php.ini-distをc:¥WINNTにphp.iniという名前でコピーします。ここまでで、PHPのコマンドライン版php-cli.exeが使用できます。実際に使う場合はc:¥phpにパスを通す必要があるでしょう。ヘルプを表示したところを**図4-1**に示します。

*1
.zipや.lzhを展開するツールは、統合アーカイバ・プロジェクトのページ<http://www.csdinc.co.jp/archiver/>などで入手できます。

3.3 Apache＋CGI版PHP

PHPをApacheとともに使用するにあたっては、CGIとして動かすこともApacheモジュールとして動かすこともできます。後者のほうが高速ですが、前者のほうが安定性が高いようです。ここではまずCGIとして動かす場合について説明します。

Apacheの設定ファイルc:¥Program Files¥Apache Group¥Apache¥conf¥httpd.confの末尾に以下の記述を追加します。

```
ScriptAlias /php4/                    "c:/php/"
Action      application/x-httpd-php "/php4/php.exe"
AddType     application/x-httpd-php .php
```

図4-1 php-cli.exeのヘルプ表示

```
C:¥php>php-cli -h
Usage: php [options] [-f] <file> [args...]
       php [options] -r <code> [args...]
       php [options] [-- args...]
  -s             Display colour syntax highlighted source.
  -w             Display source with stripped comments and whitespace.
  -f <file>      Parse <file>.
  -v             Version number
  -c <path>      Look for php.ini file in this directory
  -a             Run interactively
  -d foo[=bar]   Define INI entry foo with value 'bar'
  -e             Generate extended information for debugger/profiler
  -z <file>      Load Zend extension <file>.
  -l             Syntax check only (lint)
  -m             Show compiled in modules
  -i             PHP information
  -r <code>      Run PHP <code> without using script tags <?..?>
  -h             This help

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

Apacheの再起動を行ない、「1.5.3 PHPの設定と動作確認」を参考にしてC:¥Program Files¥Apache Group¥Apache¥htdocs¥phpinfo.phpを作成し、ブラウザからhttp://localhost/phpinfo.phpにアクセスします。第1部の「5.4.4 phpinfo()」で示したような画面が表示されればOKです。

3.4 Apache+モジュール版PHP

モジュール版を利用するには、httpd.confに以下のような修正を行ないます。

①193行目付近（LoadModuleコマンドが並んでいるところ）に以下を追加します。

```
LoadModule php4_module c:/php/sapi/php4apache.dll
```

②240行目付近（AddModuleコマンドが並んでいるところ）に以下を追加します。

```
AddModule mod_php4.c
```

③848行目付近（AddHandlerコマンドが並んでいるところ）に以下を追加します。

```
AddType application/x-httpd-php .php
```

修正（追加）が終わったらApacheの再起動を行ない、前節と同様の手順で動作を確認します。