PHP入門

PHP5.1+PostgreSQL8.1 for Windowsインストール他

桑村 潤

<http://www.ne.jp/asahi/yokohama/juk/>

2006-06-28

紹介

- 講師：桑村潤

  日本PHPユーザ会会員、

  Plamo Linux Project　WebDB担当

「はじめてのPHP5」、「入門PHPセキュリティ」共訳、
「PHP5徹底攻略エキスパート編」、「PHP5徹底攻略」、
「PostgreSQL for Windows 徹底活用ガイド」共著、
「KERBEROS ネットワーク認証システム」翻訳、
「例題によるLinuxプログラミング」監訳等

概要

- PHPについて
- Windows版
- PHP開発環境(IDE)
- DBへのアクセス
- XMLのパース
- テンプレートの利用
- セキュリティ上の注意
- キャッシュとエンコード

PHPについて

PHP: ’ PHP Hypertext Preprocesser’

- http://www.php.net

- PHPの歴史
- PHPの特徴
- 他のスクリプト言語の特徴
- PHPの活用
- PHPプログラムの改良

PHPの歴史

- Personal Home Page Tools version 1.0 －1995年
  (8.Jun.1995 comp.infosystem.www.authoring.cgi にて公開)
  - http://www.zend.com/zend/hof/rasmus.php
- **PHP/FI 2.0 までの開発は Rasmus Lerdorf氏** －1997年
  (Personal Home Page Construction Kit/Form Interpreter)
- **PHP3からZeev Suraski(Zend社)らが開発** －1998年
  (PHP Hypertext Preprocessor)
- **PHP4はZend Engine(強力なパーサ)搭載** －2000年
  - http://www.zend.com
- **PHP5はZend Engine2,オブジェクト指向強化** －2004年
- **PHP5.1で高速化、PDOの追加** －2005年

PHPの特徴

- オープンソースソフトウェア
- サーバサイドスクリプト言語
- 馴染みの構文
  - C言語、C++言語に似た構文と関数
  - Perlに似た変数や配列
  - Javaに似たクラス [PHP5]
- マルチバイト文字対応
  - mbstring,mbregex,iconv
- 豊富なサポート関数群
  （4500を超える数の関数 http://php.net/quickref.php）
  - DBMSインターフェース
  - イメージ処理関数(GD)
  - XML処理関数
- **PEARオンラインリポジトリ**
  （PEAR: PHP Extension and Application Repository）
  - クラスライブラリ
- **PECL拡張ライブラリ**
  （PHP Ectension Community Library）
  - バイナリ拡張ライブラリ

他のスクリプト言語の特徴

- C-Shell, B-Shell, Bash
  - Unix系カーネルのコマンドインタプリタ
  - OSまわりのシェルプログラム
- Perl(Practical Extraction and Report Language)
  - C, Shell, sed, AWK に似た言語
  - さまざまなコーディングスタイル
  - オブジェクト指向も可能
  - OSまわりのシェルプログラムに利用されたことも
- Python
  - オブジェクト指向言語
  - 実用的なScheme(Lispの一種)、きれいなPerlのような特徴
  - 厳格なコーディング規約
- Ruby
  - オブジェクト指向言語
  - 柔軟なコーディング
  - 開発者が日本人
- Zope(Z Object Publishing Environment)
  - Pythonで書かれたWebアプリケーションフレームワーク
- Tcl/Tk(Tool Command Language/ToolKit)
  - Tclは組み込み用のスクリプト言語
  - TkはTcl用マルチプラットホームGUI
  - TcletはNetscape ブラウザ上で動いた(Java Scriptのような)

PHPの活用

- 既存のWebページを簡単に活性化
  - 日付、ファイル更新日、条件分岐
- 既存のCGI(C, FORTRAN, Perl)を移植
  - C, FORTRAN => 変数名に$を付ける
  - Perl => とりあえず動かしてみる
- 長いプログラムはサブルーチン化し分ける
  - include, require 命令を使う
- 高速化が必要であればCの構文にて
  - Cに移植して最適化コンパイル
- 利用頻度の高いルーチンは関数に分割
  - include_once, require_once 命令を使う
- データ構造と処理をまとめてクラス化
  - 単機能関数の集合に切り分け
  - データ構造中心のオブジェクトとしてまとめる

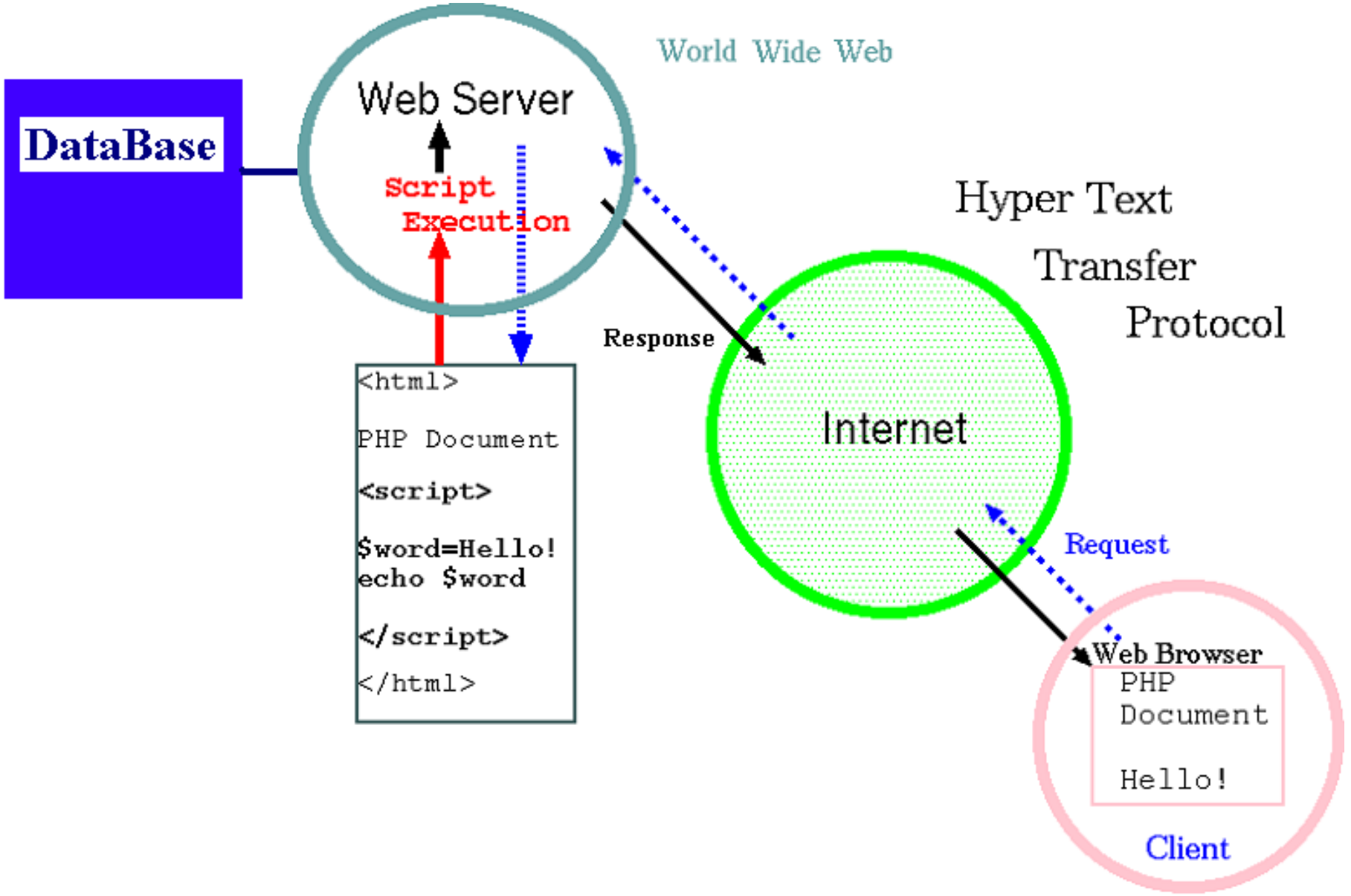
PHPプログラムの改良

- サブルーチン（単なる処理分割）
  - レスポンスが早い（キャッシュとの併用）
  - プログラム的には不透明
  - 切り出しは簡単
- 関数（function）
  - 機能単位でまとめればレスポンスは早い
  - 再利用が可能（原則globalを使わない）
  - 引数と戻り値を定義
- クラス（class）
  - 複雑になるとロードに時間がかかる
  - 再利用性が高くなる
  - プログラム的には高度、開発工数が大
- フレームワーク
  - 大規模化に対応可能
  - 適用方法がはっきりすれば再生産性が高くなる
  - メインテナンス性が高い

フレームワーク
（*Web*アプリケーションフレームワーク）

- セッション管理
  - 継続性の識別
  - 利用統計
- テンプレートシステム
  - レイアウトデザインとプログラムの統合
- データベースインターフェース
  - O／Rマッピング
  - コネクション管理
- 認証
  - クライアント識別
  - 同時ログイン制御
- 入力チェック
  - 悪意のインジェクション排除
  - 入力値の妥当性検証
- 出力エスケープ
  - 不要タグのHTMLエンコード

サーバーサイドスクリプト



PHPの埋め込み例

PHPプログラムをHTLMに埋め込んだ例:

```
<html>
<body>
只今の時間:
<?php
  echo date( “Y-m-d(D) h:ia”, time() );
?>
<br />
</body>
</html>
```

出力:

只今の時間: 2005-06-11(Sat) 05:27pm

インストール（Windows）

- Apach 2.0.x
  - http://www.apache.org/dist/httpd/binaries/win32/
- PHP 5.1.x（Apache DSO）
  - http://www.php.net/downloads.php/
- PostgreSQL 8.1.x
  - http://www.postgresql.org/ftp/binary/v8.1.4/win32/

参照： http://www.postgresql.jp/niigata
(ただし、Apache1.3+PHP4+PostgreSQL8β)

Windowsでも開発環境を構築できる

Apacheのインストール

- Apacheダウンロードサイト
  - http://www.apache.org/dist/httpd/binaries/win32/
- Windows版Apacheインストーラパッケージ
  - apache_2.0.58-win32-x86-no_ssl.msi
- インストーラに従ってインストール

  - インストールが成功すると自動的に起動されるタスクバーのApache Monitorから停止(Stop)する

PHP5.1のインストール

- PHPダウンロードサイト
  - http://www.php.net/downloads.php/
- Windows版PHPバイナリパッケージ
  - php-5.1.4-Win32.zip

  解凍して所定の場所へ移動(c:¥php5¥)

  - PHP日本語パッチダウンロード(PHP5.0の場合)
    - http://www.geocities.jp/rui_hirokawa/php/
    - Windows版PHP　日本語バイナリパッチ
      - php5.0.4-mb10a.lzh

      （マルチバイト対応、pgsql, mysql, pdo,gd2等のDLLを含む）

Apache2の設定(DSO版PHP5)

- Apache¥conf¥httpd.conf の設定
  - PHP5のための設定

```
DirectoryIndex index.html index.html.var index.php index.php.var
```

```
# PHP5(DSO)
LoadModule php5_module "C:/php5/php5apache2.dll"
AddType application/x-httpd-php .php
AddType application/x-httpd-php-source .phps
```

  - その他のWebサーバ設定

    ServerAdmin, ServerName, DocumentRootなど

PHP5の設定

- php5ts.dll
  - PHP5¥にパスを通す。または、php5ts.dll を Apache¥ へコピー
    （あるいは、c:¥WINDOWS¥system32¥ へ）
- c:¥WINDOWS¥php.ini
  - PHP5¥php.ini-recomendedをc:¥WINDOWS¥php.iniにコピー

```
extension_dir = "c:/php5/ext"
extension=php_mbstring.dll
extension=php_pdo.dll
extension=php_pgsql.dll
extension=php_pdo_pgsql.dll
```

  - そのほか、php_*.dll を適宜コメントをはずすか追加する
- コマンドラインからのCLI起動テスト
  - php.exe -i
  - Configuration File (php.ini) Path => c:¥php5¥php.ini を確認
- PEARのインストール
  - c:¥php5フォルダにて、go-pear.bat を実行する
  - go-pear.bat
  - dir PEAR

Apache+PHP5の起動

- Path環境変数にPHP5のパスを追加
  - スタート->コントロールパネル->システム->詳細設定->環境変数

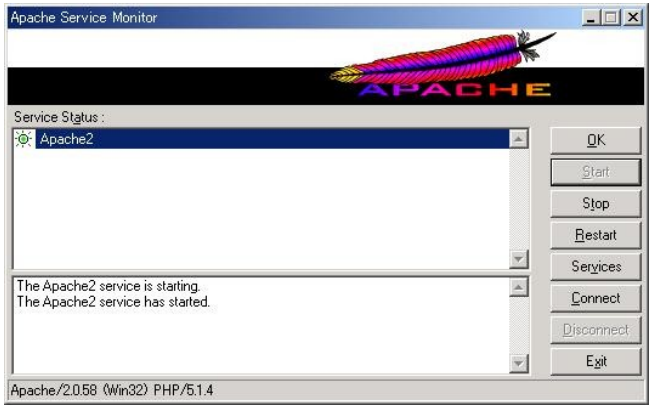
```
PHP5R=C:¥php5
PATH=%PHP5R%;%PATH%
```

- コマンドラインからApacheを起動確認
  - Apache¥bin¥Apache.exe
  - ^C
  - tail Apache/log/error_log

```
[Sun Jun 12 13:22:30 2005] [notice] Apache/2.0.54 (Win32) PHP/5.0.4 configured -- resuming normal
    operations
```

（ ここで、Apache は c:¥Program Files¥Apache Group¥Apache2）

  - タスクバーからApache2を再び起動



- PHPファイルのパーステスト
  - Apache¥htdocs¥phpinfo.php を次の内容で作成

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

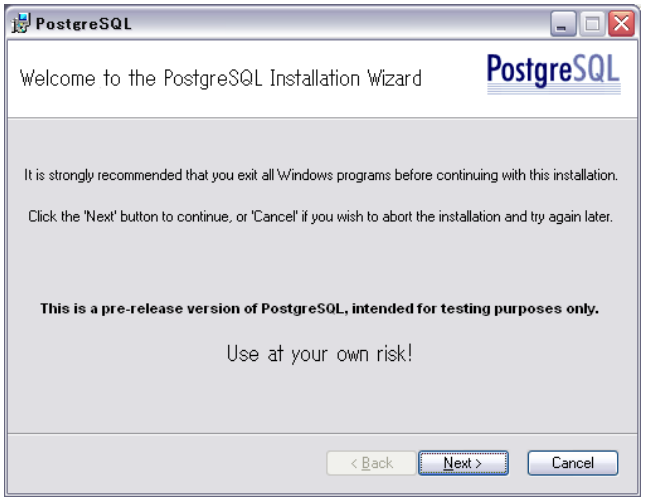
  - Webブラウザにて http://localhost/phpinfo.php にアクセスして確認

phpinfoの確認


phpinfo() - Mozilla Firefox
ファイル(F) 編集(E) 表示(V) 移動(G) ブックマーク(B) ツール(T) ヘルプ(H)
移動
PHP Version 5.1.4
php
System
Windows NT PICTY 5.0 build 2195
Build Date
May 4 2006 10:30:29
Configure Command
cscript /nologo configure.js "--enable-snapshot-build" "--with-gd=shared"
Server API
Apache 2.0 Handler
Virtual Directory Support
enabled
Configuration File (php.ini) Path
C:¥WINNT¥php.ini
PHP API
20041225
PHP Extension
20050922
Zend Extension
220051025
Debug Build
no
Thread Safety
enabled
Zend Memory Manager
enabled
IPv6 Support
enabled
Registered PHP Streams
php, file, http, ftp, compress.zlib
Registered Stream Socket Transports
tcp, udp
Registered Stream Filters
convert.iconv.*, string.rot13, string.toupper, string.tolower, string.strip_tags, convert.*, consumed, zlib.*
This program makes use of the Zend Scripting Language Engine:
Zend Engine v2.1.0, Copyright (c) 1998-2006 Zend Technologies
Powered By
Zend Engine 2
完了

PostgreSQLのインストール

- pginstallerダウンロードサイト
  - http://www.postgresql.org/ftp/binary/v8.1.4/win32/
- Windows版インストーラ
  - postgresql-8.1.4-ja.zip
    - ダウンロードした.zipファイルを展開して.msi を取り出す
  - postgresql-8.1-ja.msi
- インストーラ(.msi)を実行
  - ダイアログに従って完了
  - メニューからpostmaster起動



*pgInstaller*に含まれるツール

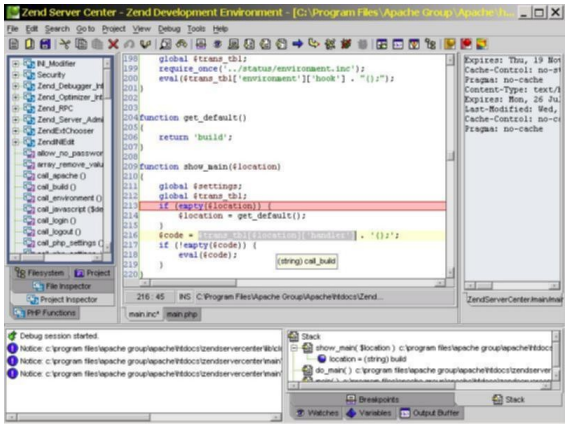
- Npgsql
  Microsoft .Netプロバイダー
  (Global Assembly Cache(GAC)にインストールされる)
- JDBC
  PostgreSQL JDBC ドライバー
  (*jdbc*¥サブディレクトリを*CLASSPATH*環境変数に要追加)
- psqlODBC
  PostgreSQL　ODBCドライバー
- pgAdmin III
  PostgreSQL管理GUI

PHP開発環境（IDE）

- コマーシャル版IDE
- オープンソース版IDE

コマーシャル版 *IDE*

- Zend Studio 5.1
  - http://www.zend.co.jp/



Profssional:$299,
Standard:$99
日本語版
価格:45,000円
アカデミック:21,000円
オンラインマニュアル
プロジェクト
リモートデバグ
PHP5.1対応

- ActiveState Komodo 3.5.3
  - http://www.activestate.com/Products/Komodo/

  Professional: $295, Personal: $29.95
- Nusphere phpEd 4.6
  - http://www.nusphere.com/

  Professional: $299, Standard: $119

オープンソース版 *IDE*

- Eclipse+TruStudio(本格的IDE)
  - http://trustudio.japansite.org/
- DBG+DDD(Unix互換GNU Debugger)
  - http://www.gnu.org/software/ddd/
- DBG+SE(Windows)
  - http://dd.cron.ru/dbg/
- IDE.PHP
  - http://www.ekenberg.se/php/ide/
- Emacs + php-mode.el
  HTMLテキスト編集とプログラム開発の同時進行

PHPからのDBアクセス（PostgreSQLの例）

- PostgreSQL関数(pg_*)
  - 処理速度が速い
  - 細かいチューニングが可能
  - 他のDBMSへの移植性は高くない
- PEARのDBクラス(DB.php)
  - コーディングが比較的簡単
  - 細かなチューニングは難しい
  - 他のDBMSへの移植性が高い
- PDO(PHP Data Object)
  - オブジェクトして取り扱える
  - 処理速度が速い
  - 他のDBMSへの移植性が高い

PostgreSQL関数でのアクセス

```
<?php
  $sql = "SELECT * FROM pg_am";
  $conn = pg_connect("host=localhost dbname=template1
   user=postgres password=passwd");
  if ($conn) {
    $result = pg_query($conn, $sql);
    if ($result) {
      for ($i=0; $i < pg_numrows($result); $i++) {
        $name = pg_result($result, $i, 'amname ');
        print "Name: $name<br>¥n";
      }
      pg_freeresult($result);
    } else {
        echo "問合せ失敗！(".pg_errormessage($conn).")<br>¥n";
    }
    pg_close($conn);
  } else {
    echo "接続に失敗！<br>¥n";
  }
?>
```

PEAR DBクラスでのアクセス

```
<?php
  require 'DB.php';
  $sql = "SELECT * FROM pg_am";
  $db =
   DB::connect('pgsql://postgres:passwd@localhost/template1');
  if (DB::isError($db)) {
    die ("接続失敗: ".$db->getMessage());
  }
  $result = $db->query($sql);
  if (DB::isError($result)) {
    die ("問合せ失敗: ".$result->getMessage());
  } else {
    while ($row = $result->fetchRow()) {
      print "Name: $row[0]<br>¥n";
    }
  }
  $db->freeResult($result);
  $db->disconnect();
?>
```

PDOでのアクセス

```
<?php
  $sql = "SELECT * FROM pg_am";
  try {
    $db = new PDO(" pgsql:host=localhost;dbname=template1",
                                                  " postgres",
   " passwd" );
  } catch(PDOException $e) {
    die ("接続失敗: ". $e->getMessage());
  }
  try {
   $q = $db->query($sql);
   $result = $q->setFetchMode(PDO::FETCH_NUM);
   while ($row = $q->fetch()) {
      print "Name: $row[0]<br>¥n";
   }
  } catch(PDOException $e) {
    die ("問合せ失敗: ". $e->getMessage());
  }
?>
```

PostgreSQLのための設定

- PostgreSQLの設定(postgresql.conf)
  - max_connections = <postgres最大接続数>
  - shared_buffers = <共有メモリ上のバッファ数>
  - work_mem = <ソートに使うバッファ数>
- PHPの設定(php.ini)
  - pgsql.max_links = <プロセス当りの最大接続数>
  - pgsql.max_persistent = <接続の内の最大持続的接続数>
- Apacheの設定(httpd.conf)
  - MaxClients = <httpd子プロセスの最大数>
- 注意点
  - max_connections > pgsql.max_links * MaxClients
  - shared_buffers > max_connections *2 (8KB)

文字(文字列)を扱うための *PostgreSQL*関数

- PostgreSQLクライアントのエンコーディングを取得

```
string pg_client_encoding ( [resource connection])
```

- PostgreSQLクライアントのエンコーディングを設定

```
int pg_set_client_encoding ([resource connection, string encoding])
```

- テキスト/文字型用の文字列をエスケープする

```
string pg_escape_string ( string data)
```

日本語文字コードの扱い

- 処理領域により文字コードが異なる可能性
  - サーバが送り出す(ブラウザが表示できる)文字コード
  - ブラウザがサーバへ送る文字コード
  - PHPプログラムが書かれた文字コード
  - サーバのPHPモジュールが処理に使う文字コード
  - データベースとのインターフェースで使う文字コード
- mbstring機能の設定(php.ini)
  - mbstring.language = Japanese
  - mbstring.internal_encoding = EUC_JP
  - mbstring.http_input = auto
  - mbstring.http_output = SJIS
  - output_handler = mb_output_handler

XMLの取り扱い

- XMLの普及
  - 業務でもXMLの電子データは当たり前
- PHPでのXML
  - Expat, XSLT, DOMXMLなどの関数群
- SimpleXMLによるXMLのパース
  - 最も単純なXMLパーサ

PHPのXML関連の関数群

- Expat XMLパーサ
  - 最も単純なパーサにはexpatを使用
    - http://sourceforge.net/projects/expat/
  - パーサの定義が必要
- XSL(Extensible Stylesheet Language, XSL Transformations)
  - XMLドキュメントを他のXMLドキュメントに変換する言語
  - GNOME XSLTを使用(旧Sabrotron版XSLTはPECLへ)
    - http://www.xmlsoft.org/
- SimpleXML [PHP5]
  - XMLを簡単にオブジェクトに変換
  - DOMとのインターフェースもあり
- DOM (旧DOMXMLはPECLへ) [PHP5]
  - DOM(Document Object Model)APIでXMLドキュメント処理
  - GNOME XMLライブラリを使用
    - http://www.xmlsoft.org/
- XMLReader [PHP5]
  - XMLパーサ

Expatによるパーサ定義例(1)

```
<?php

function trustedFile($file){
    echo ″<font color='orange'>tF</font>″;
    // 自己所有のローカルファイルのみを信頼する
    if (!eregi(″^([a-z]+)://″, $file)
        && fileowner($file) == getmyuid()) {
        return true;
    }
    return false;
}

function startElement($parser, $name, $attribs){
    echo ″<font color='orange'>sE</font>″;
    print ″&lt;<font color=¥″#0000cc¥″>$name</font>″;
    if (sizeof($attribs)) {
        while (list($k, $v) = each($attribs)) {
            print ″ <font
     color=¥″#009900¥″>$k</font>=¥″<font
                    color=¥″#990000¥″>$v</font>¥″″;
        }
    }
    print ″&gt;″;
}

function endElement($parser, $name){
    echo ″<font color='orange'>eE</font>″;
    print ″&lt;/<font
     color=¥″#0000cc¥″>$name</font>&gt;″;
}

function characterData($parser, $data){
    echo ″<font color='orange'>cD</font>″;
    print ″<b>$data</b>″;
}
```

```
function PIHandler($parser, $target, $data){
   echo "<font color='orange'>PIH</font>";
   switch (strtolower($target)) {
      case "php":
         global $parser_file;
         // 処理されるドキュメントが "信頼されている" 場合、
         // PHP コードをその内部で実行します。
         // そうでない場合、そのコードが代わりに表示されます。
         if (trustedFile($parser_file[$parser])) {
            eval($data);
         } else {
            printf("Untrusted PHP code: <i>%s</i>",
                  htmlspecialchars($data));
         }
         break;
   }
}

function defaultHandler($parser, $data){
   echo "<font color='orange'>dH</font>";
   if (substr($data, 0, 1) == "&" && substr($data, -1, 1) == ";") {
      printf('DH<font color="#aa00aa">%s</font>',
            htmlspecialchars($data));
   } else {
      printf('<font size="-1">%s</font>',
            htmlspecialchars($data));
   }
}
```

Expatによるパーサ定義例(2)

```
function externalEntityRefHandler( $parser, $openEntityNames, $base,
$systemId, $publicId ) {
   echo "<font color='orange'>eRH</font>";
   if ($systemId) {
      if (!list($parser, $fp) = new_xml_parser($systemId)) {
         return false;
      }
      while ($data = fread($fp, 4096)) {
         if (!xml_parse($parser, $data, feof($fp))) {
            printf("XML error: %s at line %d while parsing entity %s\n",
                  xml_error_string(xml_get_error_code($parser)),
                  xml_get_current_line_number($parser), $openEntityNames);
            xml_parser_free($parser);
            return false;
         }
      }
      xml_parser_free($parser);
      return true;
   }
   return false;
}
```

```
function new_xml_parser($file) {
    global $parser_file;

    $xml_parser = xml_parser_create();
    xml_parser_set_option($xml_parser,
     XML_OPTION_CASE_FOLDING, 1);
    xml_set_element_handler($xml_parser, "startElement",
     "endElement");
    xml_set_character_data_handler($xml_parser,
     "characterData");
    xml_set_processing_instruction_handler($xml_parser,
     "PIHandler");
    xml_set_default_handler($xml_parser, "defaultHandler");
    xml_set_external_entity_ref_handler($xml_parser,
     "externalEntityRefHandler");
    $parser_file[$xml_parser] = $file;
    return array($xml_parser, $fp);
}

?>
```

Expatによるパース例

```
<?php
 if (!(list($xml_parser, $fp) = new_xml_parser($file))) {
    die("could not open XML input");
    }
    print "<pre>";
    while ($data = fread($fp, 4096)) {
    if (!xml_parse($xml_parser, $data, feof($fp))) {
        die(sprintf("XML error: %s at line %d¥n",
            xml_error_string(xml_get_error_code($xml_parser)),
            xml_get_current_line_number($xml_parser)));
    }
    }
    print "</pre>";
    print "parse complete¥n";
    xml_parser_free($xml_parser);
?>
```

SimpleXMLによるパース例

```
<?php
// SimpleXML を使った XML ファイル読み込み
function read_config($config)
{
  $param=array();
  // XMLファイルをパースし読み込む
  $conf = simplexml_load_file( $config );
  // 最初のノードの子ノードを見付けて順次値を連想配列にセット
  foreach ( $conf->children() as $name => $node) {
    $value = ''.$node.'';
    // EUC-JP に変換
    $code = mb_detect_encoding($value, "auto");
    $param[$name] = mb_convert_encoding( $value, 'EUC-JP',
   $code);
  }
  return $param;
}

$param = read_config("param.xml");
echo "<pre>"; var_export($param); echo "</pre>";
?>
```

テンプレートの利用

- レイアウトデザインとプログラムロジックを分離
  - レイアウトデザインに汎用のコンポーザを利用可能
- テンプレートのクラスの利用
  - Smarty
    - http://smarty.php.net
  - PEAR HTML_Template_IT (Integrated Template)
    - PEARからインストールは簡単
      - pear install HTML_Template_IT
  - 簡易的にはEval()関数で実現可能

```
<?php
  $contents = file("テンプレートファイル名");
  eval("echo \"$contents\";");
?>
```

PEAR Integrated Templateの例

- HTML_Template_IT　を使うPHPコード例

```
<?php
  require_once 'HTML/Template/IT.php'
  $tpl = new HTML_Template_IT(./);
  $tpl->loadTemplatefile('テンプレートファイル', true, true);
  $tpl->setVariable('NOW' , date('Y-m-d(D) h:i'));
  $tpl->show();
?>
```

- テンプレートファイルの例

```
<html>
 <body>
  <table>
   <tr><th>ただいまの時刻</th><td> {NOW} </td></tr>
  </table>
 </body>
</html>
```

認証機能

- Apache の認証モジュールの利用
  - Basic認証
  - Database認証(mod_auth_oracle)
  - 認証システム(LDAP,Kerberos)による認証
- 認証クラスの利用
  - PEAR Auth
    - 認証データソースを選べる(File,DB,SASL,LDAP)
    - PEARからインストールは簡単
      - pear install Auth
    - 同一IDで複数同時ログイン可能
  - セッション管理は別(Smarty?)
  - 効率的なデバッグ?

PEAR Authの例

- PEAR Auth を使ったPHPコード例

```
<?php
require_once 'Auth.php'
$auth_params = array(
    "table" => "entry_master"              // パスワードテーブル名
,   "usernamecol" => "username"            // ユーザ名カラム名
,   "passwordcol" => "passwd"              // パスワードカラム名
,   "dsn" => "oci8://user:pass@hostname/dbname" // データソース名
);
$myauth = new Auth("DB", $auth_params, "loginForm", true);
$myauth->expire = $expire;   // セッション時間切れ  28800 for 8hr.
$myauth->idle = $idle;       // アイドリング時間切れ 1800 for 30min.

if ( $action == 'logout' ) {
    // ログアウト処理
    unset($_SESSION);
    $myauth->logout();
}

$myauth->start();
if ( $myauth->checkAuth() ) {
    // 認証された場合の処理
    echo $myauth->getUsername();
} else {
    echo "LoginError: " . $myauth->getStatus() . "<br />¥n";
}
?>
```

セキュリティ

- セキュリティの基本
  - 多重防御(Defence in Depth)を基本
  - セキュリティのレベルは最も低いところに落着
- PHPのセキュリティ
  - Webサーバのセキュリティ
  - 注意すべきPHPディレクティブ
  - Webプログラムのセキュリティ
  - Hardened-PHP Project

Webサーバのセキュリティ対策

- 接続を暗号化する
  - SSL/TLSのセッション(HTTPS)を利用する
  - サーバ証明書の設置
- クライアントの認証を行う
  - ベーシック認証、LDAP認証、Kerberos認証など
  - クライアント証明書
- 不要なポートは開かない(TCP/IPレベル)
  - 不要なサービスを起動しない
  - ファイアウォール装置によるフィルタ
- セキュリティアップデートの励行(OS、アプリ)
  - 自動アップデート
  - アップデート前のバックアップ
- システムの冗長化
  - 何らかの形でバックアップできる体制を用意
  - 負荷分散システムとの併用

注意すべきPHPディレクティブ

- Php.iniの安全サイドの設定例

```
allow_url_fopen = Off
disable_functions = eval, exec, file, file_get_contents,
  fopen, include, passthru, phpinfo, popen, preg_replace,
  proc_open, readfile, require, shell_exec, system
enabel_dl = Off
error_reporting  =  E_ALL
file_uploads = Off
log_errors = On
error_log = /var/log/php_log
magic_quotes_gpc = Off
memory_limit = 8M
open_basedir = /var/www/html
register_globals = Off
safe_mode = Off
```

  - disable_functionsの指定は注意すべき関数

Webプログラムのセキュリティ対策

- 入力のフィルタ
  - フォーム、ファイル、DBなどすべての入力が対象
  - データの利用目的にあわせてフィルタする
  - データの型や範囲によってフィルタする
  - 生の入力とフィルタ後のデータを明確に区別
  - HTMLフォームでの制限は当てにしない
- 出力のエスケープ
  - フォーム、ファイル、DBなどすべての出力が対象
  - データの出力先にあわせてエスケープする
  - データの出力書式を正しくする

PHPプログラムのセキュリティ対策

- クロスサイトスクリプティング(XSS)対策
  - 不必要なHTMLタグを出力させない
    - Htmlspecialchars()で特殊文字を無効に
    - Strip_tags()でタグの削除
- クロスサイトリクエストフォージェリ(CSRF)対策
  - register_globals = off (php.ini)
  - GETメソッドを避け、入力変数を$_REQUESTで扱わない
- SQLインジェクション対策
  - SQL文中にコマンドを挿入させない
    - Addslashes()で「’」や「;」をエスケープする
- コマンドインジェクション対策
  - 必要な場合コマンドラインに挿入をさせない
    - escaleshellcmd()でエスケープする
    - Escapeshellarg()でシングルクォートで囲いこむ

Hardened-PHP Project

- PHPのセキュリティ強化プロジェクト
  - Hardening-patch
  - http://www.hardened-php.net/
  - php-5.1.x と php-4.4.x をサポート中
- ディレクティブのより細かな設定
  - Log, Executorの制限
  - REQUEST, COOKIE, GET, POST 変数の制限
  - ファイルアップロード変数の制限

商用利用のための工夫

- キャッシュモジュール
  - サーバの負荷軽減
  - I/Oの軽減による高速化
- コンパイラー（中間コード化）
  - ソースプログラムの保護
  - 最適化による高速化
- エンコーダー（暗号化）
  - 商用配布
  - ソースプログラムの保護

商用エンコード/キャッシュモジュール

- Zend Guard 4
  - http://www.zend.co.jp/products/guard/

  日本語版価格:504,000円
  Zend Optimizer で稼動
- IonCube Encoder 6.5
  - http://asial.co.jp/products/ioncube/

  日本語版価格:62,800円
- PHTML Encoder 4.1
  - http://www.rssoftlab.com/phpenc.php

  Windows:$119,Linux:$119
  Windows+Linux(Source):$550
  日本語は？

フリーなエンコード/キャッシュモジュール

- Zend Optimizer 3.0
  - http://www.zend.co.jp/products/guard/
    高速化モジュールのバイナリのみフリー
    エンコーダは有料
- PHP Accelerator(ionCube)
  - http://asial.co.jp/products/ioncube/
    高速化モジュールのバイナリのみフリー(yahoo.comで使われている)
    エンコーダは有料
- eAccelerator(Turck MMCache)
  - http://eaccelerator.net/
    (GPL)PHP5.0はキャッシュのみ
- **PECL APC(Alternative PHP Cache)3.0**
  - http://pecl.php.net/apc
    (PHP License)
- PECL bcompiler
  - http://pecl.php.net/bcompiler
    (PHP License)エンコードのみ

APC1.0使用時のアクセステスト(参考)

- Apache benchmark(ab)
  - 同時に 1000 リクエストを 1000 回行なうテスト

ab -n 1000 -c 1000 http://.../imagelist.php

簡単なイメージ表示を含むコンテンツ、
APCを使わない状態(apc.mode=off)で17〜20秒
それぞれ5回くらいずつ計測(おおざっぱ)

apc.mode	Time(sec)	
off	17〜20	
mmap	13〜17	
shm	8〜12	(オブジェクトファイル使用)

PHP4とPHP5

- オブジェクト指向プログラミングの改良
  - Pass-by-value からPass-by-referenceへ
  - ベーシッククラスの取得 get_class()
  - プロパティ var から public/private へ、メソッドにも
  - コンストラクタとデストラクタ
  - 抽象クラス
  - オブジェクトのクローン
- 新機能
  - イタレータ
  - エクセプションによるエラー処理
  - ストリーム
  - SQLite DBMSの組み込み
  - SOAP クライアント／サーバ

PHPによる開発事例

- Webサイト
  - Yahoo,　楽天, GREE, Wikipedia
- オープンソースアプリケーション
  - ショッピングカート: OSCommerce, ZenCart
  - SNS(ソーシャルネットワーキングサービス): OpenPNE
  - CRM(顧客関係管理): SugarCRM
  - ERP(企業資産計画): OpenERP
  - 販売･在庫管理: Olut
  - ブログツール: Nucleus CMS, phpBB
  - CMS(コンテンツ管理システム): Drupal, Xoops
  - グループウェア: La’cooda Wiz, PenguinOffice
  - Wiki: Pukiwiki,MediaWiki
  - DB管理ツール: phpPgAdmin,phpMyAdmin

まとめ

- PHPで簡単にダイナミックなページ
- 好みの開発環境
- 開発効率化のためにクラスを利用
- セキュリティの考慮
- 商用アプリケーションのための工夫

参考文献

- 「はじめてのPHP言語プログラミング入門」
  技術評論社、大垣靖男 著、ISBN4-7741-2286-6
- 「PHP4徹底攻略 改訂版」
  ソフトバンクパブリッシング、堀田倫英・他 著
- 「PHP5徹底攻略」
  ソフトバンクパブリッシング、堀田倫英・桑村潤 著
- 「PHP5徹底攻略 エキスパート編」
  ソフトバンクパブリッシング、廣川類・桑村潤 著
- 「初めてのPHP5」
  O'Reilly, David Sklar, ISBN4-87311-257-5
- 「入門 PHPセキュリティ」
  O'Reilly, Chris Shiflet, ISBN4-87311-286-9
- 「PostgreSQL徹底活用ガイドfor Windows」
  インプレス、斉藤浩・他 著、ISBN4-8443-2099-8