

# Hypertext Preprocessor

第3部では、PHP をより高度に使いこなすための多くの組み込み関数および設定用オプションについて解説します。

第3部では関数の引数および返り値に関して以下のように記述します。

- ・指定を省略できる引数は[] でくくって表します。
- ・型はint( 整数 ) \ string( 文字列 ) \ array( 配列 ) \ object( オブジェクト ) \ bool( 論理型 ) \ mixed( 一般型 )と記述します。
- ・参照渡しとする引数については、変数名の前に '&' をつけます。
- ・各関数名とともに、その関数を使用可能なPHP のバージョンを示します( **PHP3** **PHP4** )。
- ・エラーが発生した際に関数がfalse を返す場合は**F**、-1 を返す場合は**M**、処理が成功した際にtrue を返す場合は**T**を示します。
- ・PHP の設定は設定ファイルで行います。PHP3 の場合はphp3.ini、PHP4 の場合はphp.ini となります。

## Part-3



## PHP 関数リファレンス

## 第 3 部



# Hypertext Preprocessor



## PHP 関数

### Chapter - 1

## データベース関連

PHP はデータベースとの連携機能が優れており、以下のように多くの主要なリレーショナルデータベース( RDBMS )をサポートしています。

オープンソース系 RDBMS : PostgreSQL、MySQL、mSQL

商用 RDBMS : Oracle、Informix、InterBase、Sybase  
Microsoft SQL Server

そのほか、Microsoft Windows 環境における標準的データベースインタフェースである ODBC やフラットファイル形式の簡易データベース dbm および dBASE がサポートされています。

ここでは、PHP で一般に使用される以下のデータベース関数について解説します。

PostgreSQL、MySQL、Oracle、InterBase、Informix、Sybase、ODBC

また、ディレクトリサービスの標準プロトコルである LDAP についても広義のデータベースとして本章でとり上げます。

Column

#### 持続的接続( Persistent Connection )について

データベースとの接続には通常の接続と持続的接続とがあります。通常の接続はPHPスクリプトの実行終了時に自動的にクローズされます( 通常の接続の場合であれば、明示的に接続を閉じる必要はありません )。一方、持続的接続の場合、スクリプトの実行終了後も接

続はオープンされたままとなり、同じ引数で接続をオープンする場合に再利用されます。持続的接続を使用することにより、接続処理の時間が短縮できるため、応答速度の向上が期待できます。持続的な接続はモジュール版のPHPでのみ使用可能です。

# 1.1 PostgreSQL 関数

PostgreSQL は、オブジェクト指向RDBMSの概念をとり入れた高機能なRDBMSです。PostgreSQL は無料で利用でき、かつ有志により早くから日本語対応が行われたため、Linux やFreeBSD といったフリーのUNIX において現在最も人気があるRDBMSのひとつになっています。

- ・ PostgreSQL 関数で接続ID( cid )を省略した場合は、デフォルトの接続IDが使われます。
- ・ ラージオブジェクト(lo)インタフェースを使用する場合、クエリをトランザクション・ブロック「BEGIN ~ END( またはCOMMIT )」の中に入れる必要があります。

---

```
string pg_client_encoding([int cid])
```

---

4.0.2 **PHP4**  
**F**

クライアントのエンコードを文字列で返します。EUC\_JP、EUC\_CN、EUC\_KR、EUC\_TW、UNICODE、MULE\_INTERNAL、LATIN1のどれかが返されます。PostgreSQL 7.0 以降が必要です。

---

```
bool pg_close([int cid])
```

---

データベースへの接続を閉じます。

**PHP3**  
**PHP4**  
**F T**

---

```
int pg_cmdtuples(int result)
```

---

INSERT、UPDATE、DELETE文により変更されたタプル( 行 )の数( 影響を受けたタプルがない場合は0 )を返します。

**PHP3**  
**PHP4**

```
$result = pg_exec ($conn, "INSERT INTO otenki VALUES ('晴れ')");
// レコードを追加
print pg_cmdtuples($result);
// 影響を受けた数(1)が出力される
```

---

```
int pg_connect(string host, string port,
               string [options], string [tty], string dbname)
```

---

データベースに接続し、接続IDを返します。同時に複数の接続をオープンすることができます。ユーザ認証を行わない場合、ユーザ名としてWebサーバの実行ユーザID( 通常はnobody )が使用されます。この名前のPostgreSQL ユーザアカウントを作成しておく必要があります。

**PHP3**  
**PHP4**  
**F**

```
// ホストfooのポート5432に接続し、データベースdbへの接続をオープンします
$id = pg_connect("foo","5432","","","db");
if (!$id){
    print "ホストfooへの接続に失敗しました。"; // 接続に失敗した場合に表示
}
```

ユーザ認証を行う場合、ユーザ名とパスワードも同時に指定します。この場合、引数パラメータ全体をひとつの文字列として指定します。

```
// ホストfooのポート5432に接続し、ユーザtaro、パスワードhanakoで
// データベースdbへの接続をオープンします
$id = pg_connect("host=foo port=5432 dbname=db user=taro password=hanako");
```

---

```
string pg_dbname([int cid])
```

---

PHP3

PHP4

F

接続しているデータベースの名前を返します。

```
if($dbname = pg_dbname($id)){
    print "現在、接続中のデータベースは$dbnameです。";
}
```

---

```
string pg_errormessage([int cid])
```

---

PHP3

PHP4

F

エラーメッセージを返します。

```
if ($error = pg_errormessage($id)){
    print "エラーメッセージ: " . $error; // エラーメッセージを表示します
}
```

---

```
int pg_exec([int cid,] string query)
```

---

PHP3

PHP4

F

クエリqueryを実行し、結果IDを返します。

```
// 接続ID $conが指すデータベースのテーブルauthorsからデータを取得します
$result = pg_exec($con,"SELECT * from authors");
```

---

```
array pg_fetch_array(int result, int i [, int type])
```

---

PHP3  
PHP4  
F

i番目のレコードを取得し、配列として返します。残りのレコードがない場合にはfalseを返します。フィールド名をキーとした連想配列としても結果にアクセスできます。

```
$result = pg_exec ($conn, "SELECT * FROM authors");
$arr = pg_fetch_array ($result, 0);
print "配列の要素:" . $arr[0] . "¥n";           // 結果を出力(数値添字でアクセス)
$arr = pg_fetch_array ($result, 1);
print "配列の要素:" . $arr["author"] . "¥n";    // 結果を出力(連想配列としてアクセス)
```

PHP4ではtypeに定数 PGSQL\_ASSOC、PGSQL\_NUM、PGSQL\_BOTHを指定することが可能です。この場合、戻り値の配列はそれぞれ連想配列、数値添字配列、両方でアクセス可能となります。

---

```
object pg_fetch_object(int result, int i [, int type])
```

---

PHP3  
PHP4  
F

クエリ結果からi番目のレコードを取得し、オブジェクトとして返します。残りのレコードがない場合にはfalseを返します。フィールド名をプロパティとして指定し、結果にアクセスします。

```
$row = 0; // カウンタを初期化
while ($data = pg_fetch_object ($result, $row)){ // データをオブジェクトとして取得
    print $data->author . "<BR>"; // フィールド名により結果にアクセスします
    $row++;
}
```

---

```
array pg_fetch_row(int result, int i)
```

---

PHP3  
PHP4  
F

クエリ結果からi番目のレコードを取得し、配列として返します。残りのレコードがない場合にはfalseを返します。

```
$result = pg_exec ($conn, "SELECT * FROM authors"); // クエリ結果を取得します
$row = pg_fetch_row ($result, 0); // 結果を配列として返します
print $row[0] . " ¥n"; // 結果を保持する配列の0番目の要素を表示します
```

---

```
int pg_fieldisnull(int result, int i, mixed field)
```

---

PHP3  
PHP4

i番目(最初のレコードが0)のレコードのフィールドfieldがNULLの場合に1、そうでない場合には0を返します。fieldはフィールド番号またはフィールド名で指定します。

```
if(pg_fieldisnull($result,0,"author")){
    print "フィールド:authorは、NULLです。";
}
```

---

```
string pg_fieldname(int result, int i)
```

---

**PHP3**  
**PHP4**

i番目(最初のフィールドが0)のフィールドの名前を返します。

```
print pg_fieldname($result,0); // 0番目のフィールドの名前を表示します
```

---

```
int pg_fieldnum(int result, string field)
```

---

**PHP3**  
**PHP4**  
**M**

フィールドfieldのフィールド番号を返します。

```
// フィールド名authorのフィールド番号を表示します
print "authorのフィールド番号は" . pg_fieldnum($result,"author") . "です";
```

---

```
int pg_fieldprtlen(int result, int i, string field)
```

---

**PHP3**  
**PHP4**  
**M**

i番目のレコードの、フィールドfieldデータの表示長(文字数)を返します。

---

```
int pg_fieldsize(int result, int i)
```

---

**PHP3**  
**PHP4**  
**F**

フィールド番号iのデータ長を返します。データ長が-1の場合は可変長データです。

---

```
string pg_fieldtype(int result, int i)
```

---

**PHP3**  
**PHP4**

i番目(最初のフィールドが0)のフィールドの型を文字列で返します。

---

```
int pg_freeresult(int result)
```

---

**PHP3**  
**PHP4**  
**F T**

結果保持用のメモリを解放します。メモリはスクリプト実行終了時に自動的に解放されるため、通常は不要です。

---

```
string pg_getlastoid(int result)
```

---

直近のコマンドがINSERTの場合に、挿入されたタプル(行)に割り当てられたオブジェクトID(oid)を文字列として返します。result\_idが有効な結果IDでない場合にfalse、直近のコマンドがINSERTでなかった場合は空の文字列を返します。

**PHP3**  
**PHP4**


---

```
string pg_host([int cid])
```

---

接続しているホストの名前を返します。

**PHP3**  
**PHP4**


---

```
void pg_loclose(int fd)
```

---

ラジオオブジェクトを閉じます。

---

```
int pg_locreate([int cid])
```

---

ラジオオブジェクトを生成し、オブジェクトID(oid)を返します。オブジェクトのアクセス権は読み書き可に設定されます。

---

```
bool pg_loexport (int oid, int file [, int cid])
```

---

ラジオオブジェクトをファイルにエクスポートします。

4.0.1 **PHP4**  
**T F**


---

```
int pg_loimport (int file [, int cid])
```

---

ラジオオブジェクトをファイルからインポートし、オブジェクトIDを返します。

4.0.1 **PHP4**  
**F**


---

```
int pg_loopen([int cid,] int oid, string mode)
```

---

ラジオオブジェクトをオープンし、ラジオオブジェクトIDを返します。ラジオオブジェクトのファイルIDをクローズする前に接続を閉じてはいけません。modeには読み書きモード(r、w、r+、w+のいずれか)を指定します。

**PHP3**  
**PHP4**


---

```
string pg_lread(int fd, int len)
```

---

ラジオオブジェクトから最大lenバイト分読み込み、文字列として返します。

**PHP3**  
**PHP4**

---

```
int pg_loreadall(int fd)
```

---

PHP3  
PHP4

ラージオブジェクトを読み込み、ブラウザに直接ラージオブジェクトを送信します。読み込んだバイト数を返します。主にイメージやサウンドなどのバイナリデータを送信するために使用します。

---

```
void pg_lounlink([int cid,] int oid)
```

---

PHP3  
PHP4

ラージオブジェクトを削除します。

---

```
int pg_lowrite(int fd, string buf)
```

---

PHP3  
PHP4  
F

文字列bufをラージオブジェクトに書き込み、実際に書き込んだバイト数を返します。

---

```
int pg_numfields(int result)
```

---

PHP3  
PHP4  
M

クエリ結果のフィールド( カラム )数を返します。

---

```
int pg_numrows(int result)
```

---

PHP3  
PHP4  
M

クエリ結果のレコード数を返します。

---

```
string pg_options([int cid])
```

---

PHP3  
PHP4

接続に関するオプションを文字列として返します。

---

```
int pg_pconnect(string host, string port,  
                string options, string tty, string dbname)
```

---

PHP3  
PHP4  
F

持続的な接続をオープンし、接続IDを返します。引数の指定方法はpg\_connectと同じです。一度に複数の持続的接続を行うことも可能です。

---

```
int pg_port([int cid])
```

---

PHP3  
PHP4

接続しているポートの番号を返します。



---

```
mixed pg_result(int result, int i, mixed field)
```

---

クエリ結果から*i*番目のレコードのフィールド*field*にある結果を取得します。レコード番号は0から始まります。フィールドは、名前またはフィールド番号で指定します。整数型 / 論理型 / オブジェクトID型は整数値、浮動小数点型と実数型は倍精度の値として返されます。配列型を含むその他のすべての型は、文字列型で返されます。

**PHP3**  
**PHP4**


---

```
int pg_set_client_encoding([int cid,] string encoding)
```

---

クライアントのエンコーディングを設定します。EUC\_JP、EUC\_CN、EUC\_KR、EUC\_TW、UNICODE、MULE\_INTERNAL、LATIN1を指定します。PostgreSQL 7.0以降が必要です。

4.0.2 **PHP4**  
**T F**


---

```
bool pg_trace (string file [, string mode [, int cid]])
```

---

デバッグ用にフロントエンド / バックエンド間の通信のファイルへのトレースを有効にします。fileおよびmodeの指定方法はfopen()と同じです。

4.0.1 **PHP4**  
**T F**


---

```
string pg_tty([int cid])
```

---

サーバ側のデバッグ出力が送られるtty名を返します。

**PHP3**  
**PHP4**

## 1.2 MySQL 関数

MySQLは、スウェーデンの開発者を中心に開発されているRDBMSです。MySQLでは高速な処理が可能であり、PHPと組み合わせて多くのWebアプリケーションが作成されています。MySQLは特に海外で人気がありますが、国内でもユーザ会が設立され、PostgreSQLとともに人気のあるオープンソースのRDBMSとなりつつあります。

PHP4の配布ファイルにはMySQLのクライアントライブラリが含まれており、デフォルトでMySQLデータベースに接続可能です。MySQLサーバーは、バージョン3.23.20以降GPL2 (GNU Public License 2)に基づき配布されています。

MySQLの開発・配布元 : <http://www.mysql.com/>

日本MySQLユーザ会 : <http://www.mysql.gr.jp/>

MySQL サーバに接続する例を以下に示します。

```
<?
mysql_connect("localhost:3306"); // ローカルホストのポート3306に接続します

// データベースtestのテーブルfooにクエリを実行します
$result = mysql_db_query ("test", "select * from otenki");
while ($row = mysql_fetch_array ($result)) { // 1レコード分のクエリ結果を取得します
    // 日付(day)と天気(tenki)を表示します
    print "日付:" . $row["day"] . ", 天気: " . $row["tenki"] . "<BR>¥n";
}
?>
```

MySQL 関数で接続ID( cid )を省略した場合には最後にオープンされた接続のIDを使用します。一度も接続がオープンされていない場合、新規に接続をオープンします。

---

```
int mysql_affected_rows([int cid])
```

---

**PHP3**  
**PHP4**

直近のINSERT、UPDATE、DELETE文により変更されたレコードの数を返します。

---

```
int mysql_change_user (string user, string password
                        [, string database [, int cid]])
```

---

**PHP3** 3.0.13  
**PHP4**

接続ユーザをuserに変更します。認証に失敗した場合は変更されません。databaseを指定した場合、ユーザ変更後にカレントのデータベースとなります。MySQL 3.23.3以降が必要です。

---

```
int mysql_close([int cid])
```

---

**PHP3**  
**PHP4**  
**T F**

接続を閉じます。通常は自動的に閉じられるため、不要です。

---

```
int mysql_connect([string hostname][:port][:path/to/socket]
                  [, string username][, string password])
```

---

**PHP3**  
**PHP4**  
**F**

接続をオープンし、接続IDを返します。引数はすべてオプションで、省略時はデフォルト値( localhost、サーバプロセスの所有者のユーザ名、空のパスワード )が使用されます。MySQL 3.22以降では、hostnameにポート番号やソケットへのパスを指定することが可能です。同じ引数で接続がすでにオープンされている場合、そのIDが返されます。

```
// localhost のMySQLサーバへの接続をオープンします
$conn = mysql_connect("localhost");
if (!$conn){
    print "正常に接続がオープンできませんでした。";
}
```

---

```
int mysql_create_db(string database [, int cid] )
```

---

新規にデータベース( database )を作成します。データベースを作成する権限が必要です。

PHP3  
PHP4  
T F

---

```
int mysql_data_seek(int result, int i)
```

---

クエリ結果の内部ポインタを、指定したi番目(最初のレコードが0)のレコードに移動します。

PHP3  
PHP4  
T F

```
// MySQLサーバに接続し、データベースtestをアクティブにします
$link = mysql_pconnect ("localhost", "taro", "secret") or die ("接続できません");
mysql_select_db ("test");

mysql_query ("select * from otenki"); // テーブルotenkiにクエリを実行

// レコードを逆順に出力
for ($i = mysql_num_rows ($result)-1; $i >=0; $i--) {
    mysql_data_seek ($result, $i)); // i番目のレコードに移動
    $row = mysql_fetch_array ($result); // 1レコード分のクエリ結果を取得します
    // 日付(day)と天気(tenki)を表示します
    print "日付:" . $row["day"] . ", 天気: " . $row["tenki"] . "<BR>¥n";
}
```

---

```
int mysql_db_query(string database, string query [, int cid])
```

---

データベースに対してクエリを実行し、結果IDを返します。

PHP3  
PHP4  
F

---

```
int mysql_drop_db(string database [, int cid])
```

---

データベースを破棄します。データベースを破棄する権限が必要です。

PHP3  
PHP4  
T F

---

```
int mysql_errno([int cid])
```

---

直近に発生したエラーのエラー番号を返します。

PHP3  
PHP4

```
string mysql_error([int cid])
```

PHP3  
PHP4

直前に発生したエラーのエラー文字列を返します。

```
array mysql_fetch_array(int result [, int type])
```

PHP3  
PHP4

クエリ結果からレコードを取得し、配列として返します。最後の行に到達した場合はfalseを返します。オプションtypeには以下の値を指定できます。

MYSQL\_ASSOC : 結果を連想配列で返します。

MYSQL\_NUM : 結果を通常の配列で返します。

MYSQL\_BOTH : 結果を通常の配列及び連想配列で返します(デフォルト)。

```
object mysql_fetch_field(int result [, int k])
```

PHP3  
PHP4

k 番目( 先頭のフィールドが 0 )のフィールド( 省略時はカレントフィールド )に関する情報を取得し、表 3-1 に示すプロパティを有するオブジェクトとして返します。

フィールドに関する情報は、表 3-2 に示す関数でも取得可能です。

表 3-1 mysql\_fetch\_field() で取得されるプロパティ

プロパティ	フィールド( カラム )情報
name	フィールド名
table	属しているテーブルの名前
max_length	最大長
not_null	not_null の場合に 1
primary_key	primary key の場合に 1
unique_key	unique key の場合に 1
multiple_key	non-unique key の場合に 1
numeric	数値( numeric )の場合に 1
blob	BLOB の場合に 1
type	フィールドの型( int,real,string,blob )
unsigned	符号なし( unsigned )の場合に 1
zerofill	ゼロで埋められる( zerofill )場合に 1

表 3-2 フィールド情報を取得する関数

関数名	返り値
string mysql_field_name(int result, int k)	フィールド名
string mysql_field_table(int result, int k)	属しているテーブルの名前
string mysql_field_type(int result, int k)	型( int,real,string,blob )
string mysql_field_flags(int result, int k)	設定されているフラグ *1
int mysql_field_len(int result, int k)	フィールドの長さ

\*1

次のフラグを返します。

not\_null  
primary\_key  
unique\_key  
multiple\_key  
blob  
unsigned  
zerofill  
binary  
enum  
auto\_increment  
timestamp

**array mysql\_fetch\_lengths(int result)**

直近に取得されたレコードの各カラムの長さを0から始まる配列に代入して返します。

**PHP3**  
**PHP4**  
**F**
**object mysql\_fetch\_object(int result)**

クエリ結果からレコードを取得し、オブジェクトとして返します。もうレコードがない場合にはfalseを返します。

**PHP3**  
**PHP4**

```
<?php
mysql_connect ($host,$user,$password); // ホスト/ユーザ/パスワードを指定して接続します

// データベースtestのテーブルotenkiにクエリを実行します
$result = mysql_db_query ("test","select * from otenki");
while ($obj = mysql_fetch_object ($result)) { // レコードをオブジェクトとして取得します
    // 日付(day)プロパティと天気(tenki)プロパティを表示します
    print "日付:" . $obj->day . ",天気: " . $obj->tenki . "<BR>¥n";
}
?>
```

**array mysql\_fetch\_row(int result)**

クエリ結果からレコードを取得し、0から始まる配列として返します。もう行がない場合はfalseを返します。mysql\_fetch\_row()を続けてコールした場合、次のレコードが返されます。

**PHP3**  
**PHP4**
**int mysql\_field\_seek(int result, int k)**

フィールドオフセットをkに設定します。

**PHP3**  
**PHP4**
**int mysql\_free\_result(int result)**

クエリ結果を保持するメモリを解放します。通常、スクリプト終了時に自動的に解放されるため不要です。

**PHP3**  
**PHP4**
**int mysql\_insert\_id([int cid] )**

直近のINSERT文においてAUTO\_INCREMENTEDフィールドで自動的に生成されたIDを返します。

**PHP3**  
**PHP4**

---

```
int mysql_list_fields(string database, string table [, int cid)
```

---

PHP3

PHP4

M

指定したデータベース・テーブルのフィールド情報を取得し、結果IDを返します。

---

```
int mysql_list_dbs([int cid])
```

---

PHP3

PHP4

データベース名のリストを取得し、結果IDを返します。

---

```
int mysql_list_tables(string database [, int cid] )
```

---

PHP3

PHP4

データベースにあるテーブルのリストを取得し、結果IDを返します。

---

```
int mysql_num_fields(int result)
```

---

PHP3

PHP4

クエリ結果のフィールド数を返します。

---

```
int mysql_num_rows(int result)
```

---

PHP3

PHP4

クエリ結果のレコード数を返します。

---

```
int mysql_pconnect([string hostname] [:port][:path/to/socket]
[, string user] [, string password])
```

---

PHP3

PHP4

F

持続的な接続をオープンし、接続IDを返します。引数の指定方法は、mysql\_connect()と同じです。

---

```
int mysql_query(string query, [int cid] )
```

---

PHP3

PHP4

アクティブなデータベースにクエリを送信します。UPDATE、INSERT、DELETE文の場合は結果を表す値( trueまたはfalse )、SELECT文の場合は結果IDを返します。

---

```
string mysql_result(int result, int i, mixed field)
```

---

PHP3

PHP4

F

i 番目のレコードのフィールドfieldのデータを返します。フィールドは、オフセット、フィールド名、テーブル名.フィールド名で指定できます。

---

---

```
int mysql_select_db(string database [, int cid])
```

---

アクティブなデータベースを設定します。

PHP3  
PHP4  
T F

---

```
string mysql_tablename(int result, int i)
```

---

テーブル名のリストから i 番目のテーブルの名前を文字列として返します。

PHP3  
PHP4  
F

```
<?php
mysql_connect ("localhost");
$result = mysql_listtables ("test"); // データベース test のテーブルのリストを取得します
$i = 0;
while ($i < mysql_num_rows ($result)) { // 結果のレコード数だけループ
    print mysql_tablename($result, $i) . "<BR>¥n"; // $i 番目のテーブル名を表示
    $i++;
}
?>
```

## 1.3 Informix 関数

Informix は有力な商用 RDBMS のひとつであり、スケーラビリティに優れた Informix Dynamic Server と、比較的軽量で中小規模向けの Informix-SE (Standard Edition) の二つの製品があります。

Informix 社のページ : <http://www.informix.com/>

この機能を使用する場合、Informix クライアント SDK に含まれる ESQL/C が必要です。また、configure を実行する前に環境変数 INFORMIXDIR を指定し、環境変数 PATH に \$INFORMIXDIR/bin を追加してください。

Informix 関数において、接続 ID を省略した場合、最後にオープンされた接続が仮定されます。接続がオープンされていない場合、ifx\_connect() と同様に接続をオープンします。

---

```
int ifx_connect([string database][, string userid][, string password])
```

---

**PHP3** 3.0.3  
**PHP4**  
**F**

サーバへの接続をオープンし、接続IDを返します。引数を省略した場合は設定ファイルの ifx.\* で指定した値が使用されます。

```
// データベースtest@ifx_srvにユーザtaro、パスワードsecretで接続します
$conn = ifx_connect ("test@ifx_srv", "taro", "secret");
```

---

```
int ifx_pconnect([string database][, string userid][, string password])
```

---

**PHP3** 3.0.3  
**PHP4**

サーバへの持続的接続を確立します。引数の指定方法はifx\_connect()と同じです。

---

```
int ifx_close([int cid])
```

---

**PHP3** 3.0.3  
**PHP4**

データベースへの接続を閉じます。常にtrueを返します。通常は自動的に閉じられるため、不要です。

---

```
int ifx_query(string query [, int cid][, int cursor][, array blobid])
```

---

**PHP3** 3.0.3  
**PHP4**

クエリを送信し、結果IDを返します。

cursor にはカーソルの型としてIFX\_SCROLL(スクロールカーソル)またはIFX\_HOLD(ホールドカーソル)またはこれらの論理和を指定します。

UPDATE クエリでは、BLOB IDを配列blobidで指定可能です。この場合、クエリの当該カラムを "?" とすることが必要です。

```
<?php
// バイトおよびテキストカラムに関するBLOB IDを作成します
$blobid[0] = ifx_create_blob(0, 0, "Text column in memory");
$blobid[1] = ifx_create_blob(1, 0, "Byte column in memory");
// クエリを実行します。BLOBカラムには、 '?' を指定します
$query = "insert into catalog (num, ucode, descr, pict) values(1,'HRO',?,?)";
$res_id = ifx_query($query, $cid, $blobid); // BLOB IDを指定してクエリを実行
?>
```

---

```
int ifx_prepare(string query, int cid [, int cursor][, mixed blobid])
```

---

**PHP3** 3.0.4  
**PHP4**

クエリを準備し、結果IDを返します。引数の指定方法はifx\_query()と同じです。



<b>int ifx_do(int result)</b>	3.0.4	PHP3 PHP4 T F
ifx_prepare()で準備したクエリを実行し、カーソルをオープンします。		
<b>string ifx_error(void)</b>	3.0.3	PHP3 PHP4
サーバのエラー文字列を返します。		
<b>string ifx_errormsg([int code])</b>	3.0.4	PHP3 PHP4
直近のエラーに関するエラーメッセージを返します。codeを指定した場合、対応するエラーメッセージが返されます。		
<b>int ifx_affected_rows(int result)</b>	3.0.3	PHP3 PHP4
クエリで作用されたレコードの数(INSERT、UPDATE、DELETEの場合は実際のレコード数、SELECTの場合は推定値)を返します。		
<pre>\$rid = ifx_prepare ("select * from otenki where ondo&gt;=30", \$connid); \$count = ifx_affected_rows (\$rid); // 変更されたレコードの推定値を取得します if (\$count &gt; 100) { // 作用されたレコードの数が多い場合は、エラーとします     print "クエリ結果のレコード数:" . \$count . "&lt;BR&gt;¥n";     die ("クエリ結果が多すぎます。¥n"); }</pre>		
<b>array ifx_getsqlca(int result)</b>	3.0.8	PHP3 PHP4
クエリ情報sqlca.sqlerrd[0..5]の内容を連想配列として返します。		
<p>\$sqlca["sqlerrd0"] : 作用されたレコードの推定値          \$sqlca["sqlerrd1"] : 新規挿入されたシリアル番号          \$sqlca["sqlerrd2"] : 実際に作用されたレコードの数</p>		
<b>array ifx_fetch_row(int result [, mixed pos])</b>	3.0.3	PHP3 PHP4 F
1件分のレコードを連想配列として取得します。続けてコールした場合、次のレコードが返されます。レコードがもうない場合はfalseを返します。		

BLOB カラムについてはBLOB ID が返され、値をifx\_get\_blob()で取得可能です。ただし、ifx\_textasvarchar(1)またはifx\_byteasvarchar(1)を指定している場合、文字列として返されます。

posはスクロールカーソル用のオプションで、NEXT、PREVIOUS、CURRENT、FIRST、LAST、レコード番号のいずれかを指定します。

```
$rid = ifx_prepare ("select * from otenki, $connid, IFX_SCROLL);
        // スクロールカーソルつきクエリを定義
ifx_do ($rid); // クエリを実行します
$row = ifx_fetch_row ($rid, "FIRST"); // 最初のレコードを取得します
while (is_array($row)) { // 配列が返されている間は、ループを実行します
    for (reset($row); $fieldname=key($row); next($row)) {
        // 連想配列 $row を順次表示します
        print $fieldname . "=" . $row[$fieldname] . " "; // フィールドの値を表示
    }
    print "<BR>¥n";
    $row = ifx_fetch_row ($rid, "NEXT"); // 次のレコードを取得します
}
```

---

```
int ifx_htmltbl_result(int result [, string options])
```

---

**PHP3** 3.0.3  
**PHP4**  
**F**

クエリ結果のすべてのレコードをHTMLテーブルとして出力し、レコード数を返します。  
optionsにより<TABLE>タグのオプションを指定可能です。

```
$rid = ifx_prepare ("select * from otenki", $connid, IFX_SCROLL);
ifx_do($rid); // クエリを実行します
// クエリ結果をborder=2を指定したテーブルとして出力します
ifx_htmltbl_result ($rid, "border=¥"2¥");
```

---

```
array ifx_fieldtypes(int result)
```

---

**PHP3** 3.0.3  
**PHP4**  
**F**

フィールド型を、フィールド名をキーとした連想配列として返します。

```
$types = ifx_fieldtypes ($result_id); // フィールド型を取得します
for (reset($types); $fname = key($types); next($types)){
    // フィールド型をすべて表示します
    print $fname . "=" . $types[$fname] . "<BR>¥n";
}
```

<b>array ifx_fieldproperties(int result)</b>	3.0.3	PHP3 PHP4 F
<p>以下のフィールドプロパティを、フィールド名をキーとした連想配列として返します。</p> <p>"SQLTYPE;length;precision;scale;ISNULLABLE"</p> <p>ただし、SQLTYPEはフィールドの型(SQLVCHAR等)、ISNULLABLEはYまたはNです。</p>		
<pre>\$prop = ifx_fieldproperties (\$resultid); // フィールドプロパティを取得します for (reset(\$prop); \$fname = key(\$prop); next(\$prop)){ // プロパティをすべて表示します     print \$fname . "=" . \$prop[\$fname] . "&lt;BR&gt;¥n"; }</pre>		
<b>int ifx_num_fields(int result)</b>	3.0.3	PHP3 PHP4 F
クエリ結果のフィールド(カラム)数を返します。		
<b>int ifx_num_rows(int result)</b>	3.0.3	PHP3 PHP4
クエリ結果として取得したレコードの数を返します。		
<b>int ifx_free_result(int result_id)</b>	3.0.3	PHP3 PHP4 F
クエリ結果のリソースを解放します。		
<b>int ifx_create_char(string param)</b>	3.0.6	PHP3 PHP4
文字列paramを値とする文字オブジェクトを作成します。		
<b>int ifx_free_char(int bid)</b>	3.0.6	PHP3 PHP4 T F
文字オブジェクトを削除します。		
<b>int ifx_update_char(int bid, string content)</b>	3.0.6	PHP3 PHP4 T F
文字オブジェクトの内容を文字列contentで更新します。		

---

```
int ifx_get_char(int bid)
```

---

**PHP3** 3.0.6  
**PHP4**

文字オブジェクトの内容を返します。

---

```
int ifx_create_blob(int type, int mode, string param)
```

---

**PHP3** 3.0.4  
**PHP4**  
**F**

BLOB オブジェクトを作成し、BLOB オブジェクトID を返します。パラメータ type、mode、param は以下のように指定します。

type : 0 = BYTE , 1 = TEXT

mode : 0 = メモリに内容を保持, 1 = ファイルに内容を保持

param : 内容へのポインタ( mode = 0 )またはファイル文字列へのポインタ( mode = 1 )

---

```
int ifx_copy_blob(int bid)
```

---

**PHP3** 3.0.4  
**PHP4**  
**F**

BLOB オブジェクトを二重化し、新規のBLOB オブジェクトID を返します。

---

```
int ifx_free_blob(int bid)
```

---

**PHP3** 3.0.4  
**PHP4**  
**T F**

BLOB オブジェクトを削除します。

---

```
string ifx_get_blob(int bid)
```

---

**PHP3** 3.0.4  
**PHP4**

BLOB オブジェクトの内容を返します。

---

```
int ifx_update_blob(int bid, string content)
```

---

**PHP3** 3.0.4  
**PHP4**  
**T F**

BLOB オブジェクトの内容を文字列content で更新します。

---

```
int ifx_blobinfile_mode(int mode)
```

---

**PHP3** 3.0.4  
**PHP4**  
**T**

SELECT クエリにおけるデフォルトのBLOB モードを設定します。mode には0( BLOB をメモリに保存 )または1( BLOB をファイルに保存 )を指定します。

<b>int ifx_textasvarchar(int mode)</b>	3.0.4	PHP3 PHP4 T
デフォルトのテキストモードを設定します。modeには0( BLOB IDを返す )または1( var charを返す )を指定します。		
<b>int ifx_byteasvarchar(int mode)</b>	3.0.4	PHP3 PHP4 T
デフォルトのバイトモードを設定します。modeには0( BLOB IDを返す )または1( var charを返す )を指定します。		
<b>int ifx_nullformat(int mode)</b>	3.0.4	PHP3 PHP4 T
取得するレコードのデフォルトの戻り値をNULL値に設定します。modeには0( ""を返す )または1( NULLを返す )を指定します。		
<b>int ifxus_create_slob(int mode)</b>	3.0.4	PHP3 PHP4 F
SLOBオブジェクトを作成し、SLOBオブジェクトIDを返します。modeには1( LO_RDONLY ) 2( LO_WRONLY ) 4( LO_APPEND ) 8( LO_RDWR ) 16( LO_BUFFER ) 32( LO_NOBUFFER )またはこれらの値の論理和を指定します。定数( IFX_LO_RDONLY、IFX_LO_WRONLY など )を使用することも可能です。		
<b>int ifxus_free_slob(int bid)</b>	3.0.4	PHP3 PHP4 T F
SLOBオブジェクトを削除します。		
<b>int ifxus_close_slob(int bid)</b>	3.0.4	PHP3 PHP4 T F
SLOBオブジェクトを削除します。		
<b>int ifxus_open_slob(long bid, int mode)</b>	3.0.4	PHP3 PHP4 F
SLOBオブジェクトをオープンし、SLOBオブジェクトIDを返します。bidは存在するSLOB IDである必要があります。modeの指定方法はifxus_create_slob()と同じです。		

---

```
int ifxus_tell_slob(long bid)
```

---

**PHP3** 3.0.4  
**PHP4**  
**F**

SLOB オブジェクトのファイルポインタの位置を返します。

---

```
int ifxus_seek_blob(long bid, int mode, long offset)
```

---

**PHP3** 3.0.4  
**PHP4**  
**F**

SLOB オブジェクトのファイルポインタを offset に設定し、新しい位置を返します。  
mode には 0( LO\_SEEK\_SET ) 1( LO\_SEEK\_CUR ) 2( LO\_SEEK\_END ) を指定します。

---

```
string ifxus_read_slob(long bid, long num)
```

---

**PHP3** 3.0.4  
**PHP4**  
**F**

指定した SLOB オブジェクトから num バイト読み込み、文字列として返します。

---

```
int ifxus_write_slob(long bid, string str)
```

---

**PHP3** 3.0.4  
**PHP4**  
**F**

SLOB オブジェクトに文字列 str を書き込み、書き込んだバイト数を返します。

## 1.4 Oracle 関数

Oracle は実績のあるデータベースシステムであり、多くの実用システムで使用されています。Oracle 8 は Linux 版もリリースされています。

Oracle Technology Network : <http://technet.oracle.com/>

Linux 版 Oracle に関する情報 : <http://linux.oracle.co.jp/>

Oracle サーバに PHP からアクセスする際には、以下の環境変数の値を正しく設定しておく必要があります。

ORACLE\_HOME : Oracle をインストールしたディレクトリのパス名

ORACLE\_SID : 接続を行うデータベース・インスタンスの名前

NLS\_LANG : 言語と文字コード ( EUC-JP の場合は、JAPANESE\_JAPAN.JA16EUC )

Oracle サーバに接続する際のサンプルを示します。

```

<?php
putenv("ORACLE_HOME=/u01/app/oracle/product/8.0.5");

putenv("NLS_LANG=JAPANESE_JAPAN.JA16EUC");
$conn = ora_logon ("scott", "tiger");
$curs = ora_open ($conn);
ora_commitoff ($conn);

ora_parse ($curs, "select * from otenki");
ora_exec ($curs);
ora_fetch ($curs);

$ncols = ora_numcols ($curs);
print "$ncols カラムの結果が得られました。<BR>";

$j=0;
while(ora_fetch($curs)){
    print "レコード$j<BR>¥n";
    for ($i=0; $i<$ncols; $i++) {
        print "カラム$i: " . ora_getcolumn($curs, $i) . "<BR>¥n";
    }
    $j++;
}
print "$j 個のレコードが得られました。<BR>¥n";
ora_close ($curs);
ora_logoff ($conn);
?>

```

```

int ora_bind(int cursor, string php_var,
            string sql_name,int length [, int type])

```

PHP変数php\_varとSQLパラメータ名sql\_name(:name形式で指定)をバインドします。この関数はora\_parse()の後、かつora\_exec()の前にコールする必要があります。typeには以下の値を指定可能です。

ORA\_BIND\_INOUT : 入出力用(デフォルト)  
 ORA\_BIND\_IN : 入力用  
 ORA\_BIND\_OUT : 出力用

PHP3  
 PHP4  
 T F

```

<?php
ora_parse ($curs, "declare tmp INTEGER; begin tmp := :in; :out := tmp; :x :=
7.77; end;");
// PHP変数とSQLパラメータをバインドします
ora_bind ($curs, "result", ":x", $len, ORA_BIND_OUT); // 出力変数にバインド
ora_bind ($curs, "input", ":in", 5, ORA_BIND_IN); // 入力変数にバインド
ora_bind ($curs, "output", ":out", 5, ORA_BIND_OUT);
$input = 765; // バインドされた変数 $input に値を設定します
ora_exec ($curs);
// バインドしたSQLパラメータx,outの値を出力します
print "Result: $result<BR>Out: $output<BR>In: $input";
?>

```

---

```
int ora_close(int cursor)
```

---

PHP3

PHP4

T F

データカーソルを閉じます。

---

```
string ora_columnname(int cursor, int i)
```

---

PHP3

PHP4

F

i番目のカラムの名前を大文字で返します。

---

```
int ora_columnsize(int cursor, int i)
```

---

PHP3

PHP4

F

i番目(最初のカラム:0)のカラムの型のサイズを返します。

---

```
string ora_columntype(int cursor, int i)
```

---

PHP3

PHP4

F

i番目のカラムの型を返します。型名は次のいずれかになります。

VARCHAR2, VARCHAR, CHAR, NUMBER, LONG

LONG RAW, ROWID, DATE, CURSOR

---

```
int ora_commit(int conn)
```

---

PHP3

PHP4

T F

Oracleトランザクションをコミットします。

---

```
int ora_commiton(int conn)
```

---

PHP3

PHP4

T F

オートコミットをオンにします。



---

```
int ora_commitoff(int conn)
```

---

オートコミットをオフにします。

PHP3  
PHP4  
T F

---

```
int ora_do(int conn, string query) [PHP3, PHP4] {F}
```

---

クエリの最初の行を実行し、カーソルIDを返します。ora\_parse()、ora\_exec()、ora\_fetch()をまとめた処理をひとつの関数で行います。クエリを1行だけ実行したい場合などに有用です。

PHP3  
PHP4  
F

```
<?php
$conn = ora_logon("scott","tiger");
$cursor = ora_do($conn,"SELECT * FROM otengi");
print ora_getcolumn($cursor,0);
ora_close($cursor);
ora_logoff($conn);
?>
```

---

```
string ora_error(int cursor)
```

---

エラーメッセージをXXX-NNNNN形式の文字列で返します。XXXはエラーの発生場所、NNNNNはエラー番号です。

PHP3  
PHP4

---

```
int ora_errorcode(int cursor)
```

---

直近に発生したエラーのコード番号を返します。

PHP3  
PHP4

---

```
int ora_exec(int cursor)
```

---

パースしたステートメントを実行します。

PHP3  
PHP4

---

```
int ora_fetch(int cursor)
```

---

クエリ結果からレコードを1件取得します。もうレコードが存在しない場合にはfalseを返します。

PHP3  
PHP4  
T F

---

```
int ora_fetch_into(int cursor, array &result [, int flags])
```

---

PHP3  
PHP4  
F

クエリ結果から1件分のレコードを配列 result に取得し、取得したカラム数を返します。  
flags には以下の値を指定可能です。

ORA\_FETCHINTO\_NULLS : NULL カラムは取得しない

ORA\_FETCHINTO\_ASSOC : カラム名をキーとする連想配列に結果を取得

---

```
mixed ora_getcolumn(int cursor, int i)
```

---

PHP3  
PHP4  
F

i 番目のカラムの値を取得します。

---

```
int ora_logoff(int connection)
```

---

PHP3  
PHP4  
T F

ユーザのログアウトとサーバ接続を切る操作を行います。

---

```
int ora_logon(string user, string password)
```

---

PHP3  
PHP4  
F

ユーザ名とパスワードを指定して接続をオープンし、接続IDを返します。TNS名をユーザ名のあとに付加した場合はSQL\*Netを使用します。

```
$conn = ora_logon("user@TNSNAME", "pass");
```

---

```
int ora_numcols(int cursor)
```

---

PHP3  
PHP4  
F

クエリ結果のカラム数を返します。

---

```
int ora_numrows(int cursor)
```

---

PHP3  
PHP4  
F

クエリ結果のレコード数を返します。

---

```
int ora_open(int connection)
```

---

PHP3  
PHP4  
F

カーソルをオープンし、カーソルIDを返します。

```
int ora_parse(int cursor, string str, int defer)
```

SQL 命令(またはPL/SQL ブロック)str をパースします。

PHP3  
PHP4  
T M

```
int ora_plogon(string user, string password)
```

持続的接続をオープンします。引数の指定方法はora\_logon()と同じです。

PHP3  
PHP4  
F

```
int ora_rollback(int connection)
```

トランザクションをロールバックします。

PHP3  
PHP4  
T F

## 1.5 oci8関数

前節の関数とは別にOracle8 Call-Interface(OCI8)を使用してOracleデータベースに接続することも可能です(Oracle8クライアントライブラリが必要)。

```
int ocibindbyname (int stmt, string name, mixed &var, int len [, int type])
```

PHP変数varをOracle変数名nameにバインドします。lenはバインド時の最大長(-1の場合は現在の長さ)を設定します。抽象データ型(LOB/ROWID/BFILE)をバインドする場合、ocinewdescriptor()によりリソースを確保しておく必要があります。typeには下記のディスクリプタの種類を指定します。

3.0.7

PHP3  
PHP4  
F

OCI\_B\_FILE(バイナリファイル)、OCI\_B\_CFILE(テキストファイル)

OCI\_B\_CLOB、OCI\_B\_BLOB、OCI\_B\_ROWID

```
int ocidefinebyname (int stmt, string col, mixed &var [, int type])
```

SELECT 結果のカラム(大文字で指定)をPHP変数(参照で指定)で参照可能とします。

3.0.7

PHP3  
PHP4  
F

```
<?php
$conn = ocilogon("scott","tiger");
$stmt = ociparse($conn, "select * from otenki");
ocidefinebyname($stmt,"DATE",&$date); // カラムdateを$dateにバインド
ociexecute($stmt);
while (ocifetch($stmt)) {
    print "date:$date<BR>¥n";
}
?>
```

---

```
int ocilogon(string user, string password [, string database])
```

---

PHP3 3.0.7  
PHP4  
F

データベースへの接続をオープンし、接続IDを返します。databaseには、ローカルなOracleインスタンス名またはtnsnames.oraに記述した接続エントリ名を指定します。省略時は、環境変数ORACLE\_SIDまたはTWO\_TASKの値を使用します。

---

```
int ociplogon(string user, string password [, string database])
```

---

PHP3 3.0.7  
PHP4  
F

データベースへの持続的接続をオープンします。引数の指定方法は、ocilogonと同じです。

---

```
int ociplogon(string user, string password [, string database])
```

---

PHP3 3.0.7  
PHP4  
F

データベースへの新規の接続をオープンします。引数の指定方法は、ocilogonと同じです。

---

```
int ocilogoff(int conn)
```

---

PHP3 3.0.7  
PHP4  
T F

データベースへの接続を閉じます。

---

```
int ociexecute(int stmt [, int mode])
```

---

PHP3 3.0.7  
PHP4  
T F

パース済みのクエリを実行します。modeに実行モード(デフォルトはOCI\_COMMIT\_ON\_SUCCESS)を指定可能です。

---

```
int ocicommit(int conn)
```

---

PHP3 3.0.7  
PHP4  
T F

カレントのトランザクションをコミットします。

<b>int ocirollback(int conn)</b>	3.0.7	PHP3 PHP4 T F
カレントのトランザクションをロールバックします。		
<b>string ocinewdescriptor(int conn [, int type])</b>	3.0.7	PHP3 PHP4 F
LOB( デフォルト )またはFILE の記述子を初期化します。type には、OCI_D_FILE、OCI_D_LOB、OCI_D_ROWID を指定可能です。LOB の場合はload、save、savefile メソッド、FILE の場合はload メソッドを使用可能です。		
<b>int ocirowcount(int stmt)</b>	3.0.7	PHP3 PHP4 F
更新をとまなうクエリで影響を受けたレコードの数を返します。SELECT クエリで取得されたレコード数は返しません。		
<b>int ocinumcols(int stmt)</b>	3.0.7	PHP3 PHP4 M
クエリ結果のカラム数を返します。		
<b>string ociresult(int stmt, mixed column)</b>	3.0.7	PHP3 PHP4 M
クエリ結果の指定カラムを返します。		
<b>int ocifetch(int stmt)</b>	3.0.7	PHP3 PHP4 T F
クエリ結果から次のレコードを取得します。		
<b>int ocifetchinto(int stmt, array &amp;result [,int mode])</b>	3.0.7	PHP3 PHP4 F
クエリ結果から次のレコードを配列に取得し、カラム数を返します。mode に表 3-3 に示すオプションを指定可能です。		

表 3-3 レコード取得時のモード

mode の値	返回值
OCI_ASSOC	連想配列を返す
OCI_NUM	通常配列( 先頭カラム : 0 )を返す( デフォルト )
OCI_RETURN_NULLS	NULL カラムを返す
OCI_RETURN_LOBS	記述子の代わりにLOB のID を返す

---

**int ocifetchstatement(int stmt, array &output [,int mode])**


---

**PHP3** 3.0.7  
**PHP4**  
**M**

クエリ結果からすべてのレコードを配列に取得し、取得したレコード数を返します。mode に表3-3 に示すオプションを指定可能です。

```
<?php
$conn = ocilogon("scott","tiger");
$stmt = ociparse($conn,"select * from emp");
ociexecute($stmt);
$nrows = ocifetchstatement($stmt,&$results);

if($nrows>0){
    while ($column = each($results)){
        $data = $column['value'];
        for ($i=0;$i<$nrows;$i++){
            print "<TD>$data[$i]</TD>¥n";
        }
    }
}
?>
```

---

**int ocicolumnisnull(int stmt, mixed column)**


---

**PHP3** 3.0.7  
**PHP4**  
**F**

カラムがNULLである場合にtrueを返します。カラム名はカラム番号(先頭カラム:1)またはカラム名を指定します。

---

**int ocicolumnsize(int stmt, mixed column)**


---

**PHP3** 3.0.7  
**PHP4**  
**F**

カラム長を返します。カラム名はカラム番号(先頭カラム:1)またはカラム名を指定します。

---

**string ociserverversion(int conn)**


---

**PHP3** 3.0.7  
**PHP4**  
**F**

サーバのバージョン情報を取得します。

---

**string ocistatementtype(int stmt)**


---

**PHP3** 3.0.7  
**PHP4**  
**F**

OCI命令の型(下記)を返します。

SELECT、UPDATE、DELETE、INSERT、CREATE  
 DROP、ALTER、BEGIN、DECLARE、UNKNOWN

<b>int ocinewcursor(int conn)</b>	3.0.7	PHP3 PHP4 F
新規にカーソルをオープンして返します。		
<b>int ocifreestatement(int stmt)</b>	3.0.7	PHP3 PHP4 T M
ステートメントに関連するリソースを解放します。ocifreecursor()としても同じです。		
<b>string ocicolumnname(int stmt, int i)</b>	3.0.7	PHP3 PHP4 F
i 番目(先頭が1)のカラムの名前を返します。		
<b>string ocicolumntype(int stmt, int i)</b>	3.0.7	PHP3 PHP4 F
i 番目(先頭が1)のカラムの型(下記)を返します。		
DATE、NUMBER、LONG、RAW、LONG RAW、VARCHAR REFCURSOR、CHAR、BLOB、CLOB、BFILE、ROWID		
<b>int ociparse(int conn, string query)</b>	3.0.7	PHP3 PHP4 F
クエリをパースし、ステートメントIDを返します。		
<b>int ocierror([int stmt conn global])</b>	3.0.7	PHP3 PHP4
stmt、conn、globalに関する直近のエラーを返します。エラーが発生していない場合はfalseを返します。		
<b>void ociinternaldebug(int onoff)</b>	3.0.7	PHP3 PHP4
内部デバッグ用出力を有効(On)または無効(Off : デフォルト)にします。		

## 1.6 Sybase( Microsoft SQL Server )関数

SQL Serverは歴史のあるRDBMS製品のひとつであり、高性能・高信頼性を有する製品としてミッションクリティカルな分野で使用されています。Linux版のSybase SQL Serverに関する情報を以下のURLから得ることができます。

<http://linux.sybase.co.jp/>

- ・Sybase 関数において接続IDを省略した場合、最後にオープンされた接続が仮定されます。
- ・Sybase 関数は、Microsoft SQL Serverとの接続にも使用可能です( sybase\_\* を mssql\_\* として使用することが可能です )

---

```
int sybase_close(int cid)
```

---

PHP3  
PHP4  
T F

Sybase データベースへの接続を閉じます。

---

```
int sybase_connect(string server, string user, string password)
```

---

PHP3  
PHP4  
F

ユーザuserおよびパスワードpasswordを用いて接続をオープンし、接続IDを返します。同じ引数ですでに接続がオープンされている場合には、既存の接続IDが返されます。

---

```
int sybase_data_seek(int result, int i)
```

---

PHP3  
PHP4  
T F

レコードポインタをi番目のレコードに移動します。

---

```
array sybase_fetch_array(int result)
```

---

PHP3  
PHP4

レコードを1件取得し、配列として返します。もうレコードが存在しない場合にはfalseを返します。フィールド名をキーとした連想配列としても結果にアクセス可能です。

---

```
object sybase_fetch_field(int result [, int k])
```

---

PHP3  
PHP4

k番目のフィールドに関する情報を取得し、オブジェクトとして返します。フィールドオフセットを指定しない場合、カレントのフィールドの情報が返されます。



オブジェクトのプロパティを以下に示します。

name : カラム名。関数の結果の場合はcomputed#N( #Nはシリアル番号 )にセットされます。  
column\_source : そのカラムがあるテーブル。  
max\_length : カラムの最大長。  
numeric : そのカラムが数値である場合に1。

---

#### **object sybase\_fetch\_object(int result)**

---

レコードを1件取得し、オブジェクトとして返します。レコードがもう存在しない場合はfalseを返します。結果にアクセスするには、フィールド名をプロパティとして指定します。

PHP3  
PHP4

---

#### **array sybase\_fetch\_row(int result)**

---

レコードを1件取得し、配列として返します。レコードがもう存在しない場合はfalseを返します。連続してコールした場合、クエリ結果から次のレコードが返されます。

PHP3  
PHP4

---

#### **int sybase\_field\_seek(int result, int field\_offset)**

---

指定したオフセットにカレントのフィールドを移動します。

PHP3  
PHP4

---

#### **int sybase\_free\_result(int result)**

---

クエリ結果保持用のメモリを解放します。メモリはスクリプト実行終了時に自動的に解放されるため、通常は必要ありません。

PHP3  
PHP4

---

#### **int sybase\_num\_fields(int result)**

---

クエリ結果のフィールド数を返します。

PHP3  
PHP4

---

#### **int sybase\_num\_rows(int result)**

---

クエリ結果のレコード数を返します。

PHP3  
PHP4

---

```
int sybase_pconnect(string server, string user, string password)
```

---

PHP3

PHP4

F

持続的な接続をオープンし、接続IDを返します。

---

```
int sybase_query(string query [, int cid])
```

---

PHP3

PHP4

F

クエリを送信し、結果IDを返します。

---

```
int sybase_result(int result, int i, mixed field)
```

---

PHP3

PHP4

i 番目のレコードのオフセット field の内容を返します。field には、フィールドのオフセット、フィールド名、フィールド名.テーブル名のどれかとすることができます。

---

```
int sybase_select_db(string database [, int cid])
```

---

PHP3

PHP4

T F

アクティブなデータベースを設定します。

## 1.7 InterBase 関数

InterBaseはBorland/Inprise製のデータベースサーバプログラムです。InterBase 6.0以降オープンソースのデータベースとして配布されており、無償で使用する事が可能です。PHP4はInterBase 6をサポートしています。

```
<?php // InterBase サーバーに接続する例
$conn = ibase_connect ($host, $username, $password); // 接続をオープン
$result = ibase_query ($conn,"SELECT * FROM otenki"); // クエリを実行
while ($row = ibase_fetch_object ($result)) { // クエリ結果を取得
    print $row->date . "<BR>¥n";
}
ibase_free_result($result); // クエリ結果を解放
ibase_close ($conn); // 接続をクローズ
?>
```

```
int ibase_connect (string database [, string user [, string password
                    [, string charset [, int buffers [, int dialect [, string role]]]])
```

3.0.6 **PHP3**  
**PHP4**  
**F**

サーバへの接続をオープンします。リモートのサーバに接続する場合、使用する接続プロトコルに応じてデータベース名の前にhostname:( TCP/IP ) //hostname/( NetBEUI ) hostname@( IPX/SPX )を指定します。charsetはデフォルトの文字セット、buffersはデータベースバッファの数、dialectはSQL 言語仕様のデフォルト値です。引数を省略した場合、デフォルト値が使用されます。

```
int ibase_pconnect (string database [, string username [, string password
                    [, string charset [, string role]]])
```

3.0.6 **PHP3**  
**PHP4**  
**F**

スクリプト終了時に接続を閉じないことと、同じ引数ですでに接続がオープンされている場合にはその接続を使用すること以外は、ibase\_connect()と同じです。

```
int ibase_close ([int cid])
```

3.0.6 **PHP3**  
**PHP4**  
**T F**

データベースへのリンクを閉じます。cidを省略した場合は、直近にオープンされた接続となります。

```
int ibase_query([int cid,] string query [, int bind_args])
```

3.0.6 **PHP3**  
**PHP4**  
**F**

クエリを実行し、結果IDを返します。

```
array ibase_fetch_row(int result)
```

3.0.6 **PHP3**  
**PHP4**

1件分のレコードを配列として取得します。

```
object ibase_fetch_object(int result)
```

3.0.6 **PHP3**  
**PHP4**

1件分のレコードをオブジェクトとして取得します。

```
int ibase_free_result(int result)
```

3.0.6 **PHP3**  
**PHP4**  
**T**

クエリ結果を保持するメモリを解放します。

---

```
int ibase_prepare([int link][, string query])
```

---

**PHP3** 3.0.6

**PHP4**

**F**

対応するパラメータのバインドやクエリ実行の準備を行います。

---

```
int ibase_execute(int query [, int bind_args])
```

---

**PHP3** 3.0.6

**PHP4**

**F**

ibase\_prepare()により準備されたクエリを実行します。

---

```
int ibase_free_query(int query)
```

---

**PHP3** 3.0.6

**PHP4**

**T**

ibase\_prepare()により準備されたクエリ用のリソースを解放します。

---

```
int ibase_timefmt(string format [, int columntype])
```

---

**PHP3** 3.0.6

**PHP4**

**T F**

timestamp、data、time型カラムの出力フォーマットを設定します。フォーマット指定は、C言語のstrftime()関数の仕様と同じです。

columntypeに設定するカラムをIBASE\_TIMESTAMP(デフォルト)、IBASE\_DATE、IBASE\_TIMEのように指定可能です( PHP4+InterBase 6でのみ有効 )。

```
<?php
// TIME型カラムの出力を '07 : 37'のように設定
ibase_timefmt ("%H : %M", IBASE_TIME);
?>
```

---

```
int ibase_num_fields(int result)
```

---

**PHP3** 3.0.7

**PHP4**

**F**

クエリ結果におけるフィールド数を取得します。

---

```
array ibase_field_info(int result, int i)
```

---

**PHP3** 3.0.7

**PHP4**

**F**

i番目のフィールドに関する情報を連想配列として返します。連想配列のキーは、"name"、"alias"、"relation"、"length"、"type"となります。

---

```
int ibase_trans([int trans [, int cid]])
```

---

**PHP4**

**F**

トランザクションを開始し、トランザクションIDを返します。transには動作モード( IBASE\_READ、IBASE\_COMMITTED、IBASE\_CONSISTENCY、IBASE\_NOWAIT )を指定します。

```
int ibase_commit([int cid,] int tid)
```

トランザクションをコミットします。tidはトランザクションIDです。

PHP4  
T F

```
int ibase_rollback([int cid,] int tid)
```

トランザクションをロールバックします。tidはトランザクションIDです。

PHP4  
T F

```
string ibase_errmsg()
```

直近のエラーメッセージを返します。

PHP4  
F

## 1.8 ODBC 関数

ODBCは、Microsoft Windows環境における標準的なデータベース用インタフェースです。ODBCサポート関数により、ODBC インタフェースをサポートする多くのデータベースと接続することが可能です。ODBC関数によりIBM DB2と接続することも可能です。

ODBC関数において接続ID(cid)を省略した場合、現在の接続が仮定されます。

```
int odbc_autocommit(int cid [, int OnOff])
```

オートコミットを設定(1 : On(デフォルト), 0 : Off)し、現在の値を返します。

3.0.6 PHP3  
PHP4  
F

```
int odbc_binmode(int result, int mode)
```

バイナリカラムデータの処理モードを以下のように設定します。

ODBC\_BINMODE\_PASSTHRU : バイナリデータとして通過  
ODBC\_BINMODE\_RETURN : そのまま返す(デフォルト)  
ODBC\_BINMODE\_CONVERT : charに変換する

3.0.6 PHP3  
PHP4

---

```
void odbc_close(int cid)
```

---

**PHP3** 3.0.6  
**PHP4**

ODBC 接続を閉じます。未完了のトランザクションがある場合にはエラーとなり、接続はオープンされたままとなります。

---

```
void odbc_close_all(void)
```

---

**PHP3** 3.0.6  
**PHP4**

サーバへのすべての接続を閉じます。未完了のトランザクションがある場合にはエラーとなり、接続はオープンされたままとなります。

---

```
int odbc_commit(int cid)
```

---

**PHP3** 3.0.6  
**PHP4**  
**T F**

すべてのトランザクションをコミットします。

---

```
int odbc_connect(string dsn, string user,  
                 string password [, int cursor])
```

---

**PHP3** 3.0.6  
**PHP4**  
**F**

ユーザ名、パスワードを指定してデータソース dsn に接続し、接続 ID を返します。複数の接続を同時にオープンすることができます。

cursor によりカーソルの型 (SQL\_CUR\_USE\_IF\_NEEDED、SQL\_CUR\_USE\_ODBC、SQL\_CUR\_USE\_DRIVER、SQL\_CUR\_DEFAULT ) を指定可能です。

---

```
string odbc_cursor(int cid)
```

---

**PHP3** 3.0.6  
**PHP4**

カーソル名を文字列として返します。

---

```
int odbc_exec(int cid, string query)
```

---

**PHP3** 3.0.6  
**PHP4**  
**F**

クエリをサーバに送信し、結果 ID を返します。odbc\_do() としても動作は同じです。

---

```
int odbc_execute(int result [, array param])
```

---

**PHP3** 3.0.6  
**PHP4**  
**T F**

odbc\_prepare() で準備された命令を実行します。配列 param により、SQL 命令のパラメータを指定できます。

<code>int odbc_fetch_into(int result [, int i], array &amp;array)</code>	3.0.6	PHP3 PHP4 F
i 番目のレコード(省略時は次のレコード)を配列 array に取得し、カラム数を返します。 array は参照渡しとする必要があります。		
<code>int odbc_fetch_row(int result [, int i])</code>	3.0.6	PHP3 PHP4 T
i 番目のレコード(省略時は次のレコード)を取得します。もうレコードが存在しない場合に false が返されます。		
<code>string odbc_field_name(int result, int i)</code>	3.0.6	PHP3 PHP4 F
i 番目(最初のフィールドは1)のフィールドの名前を返します。		
<code>int odbc_field_len(int result, int i)</code>	3.0.6	PHP3 PHP4 F
i 番目(最初のフィールドは1)のフィールド長を返します。		
<code>int odbc_field_num(int result, string field)</code>	3.0.6	PHP3 PHP4 F
フィールドのフィールド番号を返します。		
<code>string odbc_field_type(int result, int i)</code>	3.0.6	PHP3 PHP4 F
i 番目(最初のフィールドは1)のフィールドの型を文字列として返します。		
<code>int odbc_free_result(int result)</code>	3.0.6	PHP3 PHP4 T
クエリ結果を保持するメモリを解放し、true を返します。通常は自動的に解放されるため、 不要です。		
<code>int odbc_longreadlen(int result, int len)</code>	3.0.6	PHP3 PHP4 T
ロングカラム(LONG、LONGVARBINARY 型のカラム)で読み込むバイト数を len バイ トに設定します。		

---

```
int odbc_num_fields(int result)
```

---

**PHP3** 3.0.6  
**PHP4**

クエリ結果におけるカラムの数を返します。

---

```
int odbc_num_rows(int result)
```

---

**PHP3** 3.0.6  
**PHP4**  
**M**

クエリ結果のレコードの数を返します。INSERT、UPDATE、DELETE文の場合は作用したレコード数、SELECT文の場合は取得可能なレコードの数(いくつかのODBCドライバを除く)を返します。

---

```
int odbc_pconnect(string dsn, string user, string password [, int cursor])
```

---

**PHP3** 3.0.3  
**PHP4**  
**F**

持続的な接続をオープンし、接続IDを返します。引数の指定方法はodbc\_connect()と同じです。

---

```
int odbc_prepare(int cid, string query)
```

---

**PHP3** 3.0.6  
**PHP4**  
**F**

クエリの実行を準備し、結果IDを返します。

---

```
string odbc_result(int result, mixed field)
```

---

**PHP3** 3.0.3  
**PHP4**  
**F**

フィールドの値を返します。fieldはフィールド番号(先頭は1)またはフィールド名とします。

```
print odbc_result($rid, 3);      // 三番目のフィールドの値を出力します
print odbc_result($rid, "val"); // "val" という名前のフィールドの値を出力します
```

---

```
int odbc_result_all(int result [, string format])
```

---

**PHP3** 3.0.6  
**PHP4**  
**F**

クエリ結果からすべてのレコードをHTMLテーブル形式で出力し、レコード数を返します。formatにより、テーブルのフォーマットをHTMLタグで指定可能です。

---

```
int odbc_rollback(int result)
```

---

**PHP3** 3.0.6  
**PHP4**  
**T F**

すべての未解決の命令をロールバックします。



---

```
int odbc_setoption(int id, int func, int option, int param)
```

---

3.0.6 **PHP3**  
**PHP4**

ODBC接続のオプションoptionをparamに設定します。idには接続IDまたは結果IDを指定します。

funcは適用するODBC関数の種類で、SQLSetConnectOption()の場合に1、SQLSetStmtOption()の場合に2とします。

## 1.9 dba関数

dbaは、UNIX標準の簡易データベースシステムであるBerkeley DB型データベースをサポートします。通常、パスワードの管理などの簡単な処理を行う際に使用されます。dbaでは、複数の実装(dbm、ndbm、GNU gdbm、db2、db3、cdb)がサポートされています。

```
$id = dba_open ("/work/test.db", "n", "db2"); // db2型のデータベースファイルをオープン

if(!$id) {
    print "データベースがオープンできませんでした。";
    exit;
}

dba_replace ("key", "This is a pen!", $id); // キー"key"の値を置換

// キー"key"が存在する場合には値を出力し、レコードを削除
if(dba_exists ("key", $id)) {
    print dba_fetch ("key", $id);
    dba_delete ("key", $id);
}

dba_close ($id); // データベースを閉じる
?>
```

---

```
void dba_close(int handle)
```

---

3.0.8 **PHP3**  
**PHP4**

データベースを閉じます。

---

```
int dba_delete(string key, int handle)
```

---

**PHP3** 3.0.8  
**PHP4**  
**T F**

データベースからkeyをキーとするレコードを削除します。

---

```
bool dba_exists(string key, int handle)
```

---

**PHP3** 3.0.8  
**PHP4**

キーkeyが存在する場合にtrue、見つからない場合にはfalseを返します。

---

```
string dba_fetch(string key, int handle)
```

---

**PHP3** 3.0.8  
**PHP4**  
**F**

キーkeyが指す値を文字列として返します。

---

```
string dba_firstkey(int handle)
```

---

**PHP3** 3.0.8  
**PHP4**

内部ポインタをリセットし、先頭要素のキーを返します。

---

```
string dba_insert(string key, string value, int handle)
```

---

**PHP3** 3.0.8  
**PHP4**

キーをkeyとして値valueを挿入します。指定したキーがすでに存在する場合はfalseを返します。

---

```
string dba_nextkey(int handle)
```

---

**PHP3** 3.0.8  
**PHP4**

次の要素のキーを返し、内部ポインタを前に進めます。

---

```
int dba_popen(string path, string mode, string handler)
```

---

**PHP3** 3.0.8  
**PHP4**

持続的接続をオープンします。引数の指定方法はdba\_open()と同じです。

---

```
int dba_open(string path, string mode, string handler)
```

---

**PHP3** 3.0.8  
**PHP4**  
**F**

指定したパス(path)にあるデータベースへの接続をオープンし、ハンドラIDを返します。modeには r(新規作成) r(読み込み) w(書き込み) rw(読み込み/書き込み)を、handlerには使用するデータベース(gdbm、dbm、ndbm、cdb、db2、db3のうち使用可能なもの)を指定します。

<b>bool dba_optimize(int handle)</b>	3.0.8	PHP3 PHP4 T F
vacuum などによりデータベースを最適化します。		
<b>int dba_replace(string key, string value, int handle)</b>	3.0.8	PHP3 PHP4 T F
キーをkeyとして値valueを挿入します。指定したキーがすでにある場合は上書きします。		
<b>bool dba_sync(int handle)</b>	3.0.8	PHP3 PHP4 T F
データベースの同期をとります。		

## 1.10 LDAP 関数

LDAP (Lightweight Directory Access Protocol) はディレクトリサービスを提供するしくみであり、階層構造を有するデータベースへのAPIを提供するものです。

ディレクトリサービスは、ファイルやリソースの管理などに使用することが可能で、データの更新よりも検索のほうがはるかに多い用途ではリレーショナル型データベースよりも管理および性能の面で優れています。

日本語を使用した検索を行う場合には、LDAPv3をサポートするライブラリを使用し、文字コードは国際化関数などによりUTF-8形式とする必要があります。

PHPでは、LDAPライブラリとして以下のものをサポートしています。

- OpenLDAP: <http://www.openldap.org/>  
バージョン2以降でLDAPv3をサポートします\*2。
- Netscape 社によるLDAP SDK <http://developer.netscape.com/tech/directory/>

個々のデータは、以下のようなDN( distinguished name )によりアクセスします。

**cn=Taro Tanaka,ou=Engineer,o=phpgr,c=JP**

DNは右から左に解釈され、順にcountry、organization、organizationalUnit、common Nameを表します。

\*2

ミシガン大学によるパッケージ  
( <http://www.umich.edu/~dirsvcs/ldap/> )も使用可能です。

LDAP関数によるエントリ検索の例を示します。

```
<?php
$ds=ldap_connect("localhost") or die("couldn't connect LDAP server.");
$r = ldap_bind($ds); // LDAPディレクトリに匿名バインド
$filter = "sn=S*"; // 検索用フィルタを定義
$sr=ldap_search($ds,"o=phpgr, c=JP",$filter); // 検索を実行
$info = ldap_get_entries($ds, $sr); // エントリを取得
for ($i=0; $i<$info["count"]; $i++) {
    // 各エントリの属性を表示します。
    print $info[$i]["dn"] . ":" . $info[$i]["cn"][0] . ":"
        . $info[$i]["mail"][0] . "<BR>¥n";
}
ldap_close($ds);
?>
```

---

```
int ldap_bind(int link [, string rdn, string pwd])
```

---

**PHP3**  
**PHP4**  
**F**

リソースのDNおよびパスワードを指定してLDAPディレクトリにバインドします。  
rdnおよびpwdを指定しない場合は匿名バインドとなります。

```
$r=ldap_bind($ds,"cn=root, o=phpgr, c=JP", "secret"); // 認証つきバインド
```

---

```
int ldap_unbind(int link)
```

---

**PHP3**  
**PHP4**  
**T F**

LDAPディレクトリへのバインドを解除します。ldap\_close()も使用可能です。

---

```
int ldap_connect([string host [, int port]])
```

---

**PHP3**  
**PHP4**  
**F**

LDAPディレクトリサーバに接続し、リンクIDを返します。hostを指定しない場合、すでにオープンされている接続のIDとなります。portにより接続ポート(デフォルトは389)を指定可能です。

---

```
int ldap_search(int link, string base_dn, string filter [, array attrib])
```

---

**PHP3**  
**PHP4**  
**F**

スコープをLDAP\_SCOPE\_SUBTREEとし、指定したDN以下でLDAPツリーを検索します。filterには検索用のフィルタを指定します。attribに必要とする属性を配列で指定することが可能です。

---

```
int ldap_read(int link, string base_dn, string filter [, array attrib])
```

---

スコープをLDAP\_SCOPE\_BASEとし、指定したDNでLDAPツリーを検索し、エントリをひとつ取得します。引数の指定方法は、ldap\_search()と同じです。

PHP3  
PHP4  
F

---

```
int ldap_list(int link, int base_dn, string filter [, string attrib])
```

---

スコープをLDAP\_SCOPE\_ONELEVELとし、指定したDNの直下のレベルにおいて検索を行います。引数の指定方法は、ldap\_search()と同じです。

PHP3  
PHP4  
F

---

```
int ldap_free_result(int result)
```

---

結果保持用のメモリを解放します。

PHP3  
PHP4  
F

---

```
int ldap_count_entries(int link, int result)
```

---

検索結果のエントリ数を返します。

PHP3  
PHP4  
F

---

```
int ldap_first_entry(int link, int result)
```

---

先頭のエントリのIDを返します。

PHP3  
PHP4  
F

---

```
int ldap_next_entry(int link, int result)
```

---

次のエントリのIDを返します。ldap\_first\_entry()をコールしたあと、連続的にエントリを取得することが可能です。

PHP3  
PHP4  
F

---

```
array ldap_get_entries(int link, int result)
```

---

結果エントリをすべて取得し、以下のようなツリー構造を有する配列として返します。

PHP3  
PHP4  
F

```
$value["count"]      = 結果におけるエントリの数
$value[0]            = 最初のエントリの情報
$value[i]["dn"]       = 結果におけるi番目のエントリ DN
$value[i]["count"]    = i 番目のエントリの属性の数
$value[i][j]         = i 番目のエントリのj 番目の属性
$value[i]["属性"] ["count"] = i 番目のエントリの属性に関する値の数
$value[i]["属性"] [j]  = i 番目のエントリの属性におけるj 番目の値
```

---

```
string ldap_first_attribute(int link, int result, int &berid)
```

---

PHP3  
PHP4  
F

エントリにおける最初の属性を返します。beridは位置ポインタのIDで、参照渡しとします。

---

```
string ldap_next_attribute(int link, int result, int &berid)
```

---

PHP3  
PHP4  
F

次の属性を返します。beridは位置ポインタのIDで、参照渡しとします。ldap\_first\_attribute()と同じberidを指定し、連続的に属性を取得することが可能です。

---

```
array ldap_get_attribute(int link, int result)
```

---

PHP3  
PHP4  
F

カレントのエントリから属性をすべて取得し、以下のような配列として返します。

\$value["count"] = そのエントリの属性の数

\$value[0] = 最初の属性

\$value[n] = n番目の属性

\$value["属性"]["count"] = その属性に関する値の数

\$value["属性"][0] = その属性に関する最初の値

\$value["属性"][i] = その属性に関するi番目の値

---

```
array ldap_get_values(int link, int result, string attrib)
```

---

PHP3  
PHP4  
F

検索結果のエントリから値をすべて取得し、配列として返します。キー count にてエントリ数を取得可能です。

---

```
array ldap_get_values_len(int link, int result, string attrib)
```

---

PHP3 3.0.13  
PHP4  
F

検索結果のエントリから値のバイナリ値として取得し、配列として返します。キー count にてエントリ数を取得可能です。

---

```
string ldap_get_dn(int link, int result)
```

---

PHP3  
PHP4  
F

結果エントリのDNを返します。

---

**array ldap\_explode\_dn(string dn, int with\_attrib)**


---

DNを要素別に分割して配列として返します。キー count にて要素数を取得可能です。  
with\_attrib に1を指定すると値のみ、0を指定すると、RDN形式(属性=値)で取得されます。

**PHP3**  
**PHP4**  
**F**


---

**string ldap\_dn2ufn(string dn)**


---

DNをUFN( User Friendly Naming )形式に変換します。

**PHP3**  
**PHP4**  
**F**


---

**int ldap\_add(int link, string dn, array entry)**


---

LDAPディレクトリにエントリを追加します。属性は以下のように設定します。

**PHP3**  
**PHP4**  
**T F**

```
entry["属性1"] = 値
entry["属性2"][0] = 値1    // 同一属性に複数の値を定義する場合
entry["属性2"][1] = 値2
```

```
<?php // エントリ追加の例
$ds=ldap_connect("localhost");
$r=ldap_bind($ds,"cn=root, o=phpgr, c=JP", "secret"); // 認証つきバインド
// エントリデータを定義
$entry = array("cn"=>"Suzuki Taro","sn"=>"Taro","mail"=>"suzuki@tokyo.jp",
               "objectclass"=>"person");
$r=ldap_add($ds, "cn=Suzuki Taro, o=phpgr, c=JP", $entry); // エントリを追加
ldap_close($ds);
?>
```

---

**int ldap\_modify(int link, string dn, array entry)**


---

エントリを修正します。エントリの指定方法はldap\_add()と同じです。

**PHP3**  
**PHP4**  
**T F**


---

**int ldap\_mod\_replace(int link, string dn, array entry)**


---

属性の値を新しい値に更新します。修正は属性レベルで行われます。

**PHP3**  
**PHP4**  
**T F**

---

**int ldap\_mod\_add(int link, string dn, array entry)**

---

**PHP3** 3.0.8**PHP4**  
**T F**

カレントの属性に属性を追加します。修正は属性レベルで行われます。

---

**int ldap\_mod\_del(int link, string dn, array entry)**

---

**PHP3** 3.0.8**PHP4**  
**T F**

カレントの属性から属性を削除します。修正は属性レベルで行われます。

---

**int ldap\_delete(int link, string dn)**

---

**PHP3**  
**PHP4**  
**F**

ディレクトリからエントリを削除します。

---

**int ldap\_errno(int link)**

---

**PHP3** 3.0.12**PHP4**

カレントのエラー番号( ない場合には0 )を返します。

---

**string ldap\_err2str(int errno)**

---

**PHP3** 3.0.13**PHP4**

エラー番号からエラー文字列を取得します。

---

**string ldap\_error(int link)**

---

**PHP3** 3.0.12**PHP4**

カレントのエラー文字列を返します。





PHP では、配列などのさまざまな変数を使用することができます。また、文字列を処理する際に便利な正規表現を処理する関数もサポートされています。

## 2.1 配列関数

```
array array (var1, var2,...)
```

引数を値とする配列を返します。

PHP3  
PHP4

```
$a = array(3,5,2); // $a[0] = 3,$a[1] = 5, $a[2] = 2となります
```

Perlのように => 演算子によりインデックスを指定することも可能です。

```
$a = array ("a"=>"lemon", "b"=>"banana", "c"=>"apple");  
print $a[b] // bananaが出力されます
```

配列インデックスを不連続に指定したあとに指定を省略した場合、最後のインデックスの次の数字が使用されます。

```
$a = array (0=>"first", 2=>"second", "third");  
// $a[0]="first".$a[1]="", $a[2]="second", $a[3]="third" となります
```

```
array array_count_values (array arr)
```

PHP4

配列の値をキーとし、各値の出現回数を値とした配列を返します。

---

**array array\_flip (array arr)**

---

**PHP4**

配列を逆順にして返します。

---

**array array\_keys (array arr [, mixed search])**

---

**PHP4**

配列からすべてのキー(数値および文字列)を返します。二番目の引数を指定した場合、その値を含むキーのみが返されます。

---

**array array\_merge (array array1, array array2 [, ...])**

---

**PHP4**

指定した配列を順番に結合し、ひとつの配列として返します。同じキーが指定されていた場合にはあとから指定された値で上書きされます。

---

**array array\_pad (array arr, int len, mixed val)**

---

**PHP4**

長さがlenとなるように値valにより配列arrを拡張して返します。lenが負の場合は左側、正の場合は右側が拡張されます。

```
$a = array(3, 4, 10);  
$b = array_pad ($a, 5, 0); // 配列$bはarray(3, 4, 10, 0, 0)となります  
$c = array_pad ($a, -6, -1); // 配列$cはarray(-1, -1, -1, 3, 4, 10)となります
```

---

**mixed array\_pop (array arr)**

---

**PHP4**

配列から最後の値を取り出して返します。取り出された要素は配列から削除されます。

---

**int array\_push (array arr, mixed var [, ...])**

---

**PHP4**

指定した要素を配列の最後に追加します。複数の値を指定可能です。

---

**array array\_reverse (array arr)**

---

**PHP4**

配列を逆順にして返します。

**mixed array\_shift (array arr)****PHP4**

配列の先頭の値を取り出して返します。配列arrの数はひとつ減ります。

```
$a = array ("taro", "hanako");
$b = array_shift ($a); // $bは"taro"、$aは"hanako"となります
```

**array array\_slice (array arr, int pos [, int len])****PHP4**

配列の位置posから長さlen分の要素を取得します。posが正の場合は先頭から、負の場合は最後からの位置を指定します。lenに負の値を指定した場合は配列の終端からの-lenの位置まで返され、lenを指定しない場合は最後の要素まで返されます。

```
$a = array ("a", "b", "c", "d", "e");
$b = array_slice ($a, 2);          // "c","d","e"を返します
$c = array_slice ($a, 2, -1);     // "c","d"を返します
$d = array_slice ($a, -2, 1);     // "d"を返します
$e = array_slice ($a, 0, 3);      // "a","b","c"を返します
```

**array array\_splice (array arr, int pos [, int len [, array rep]])****PHP4**

配列の位置posから長さlen分の要素を削除し、削除した値を配列として返します。配列repを指定した場合は、その要素で置換します。位置および長さの指定方法は、array\_slice()と同じです。

**int array\_unshift (array arr, mixed var [, ...])****PHP4**

配列の先頭に指定した値を追加します。追加後の要素の数を返します。

**array array\_values (array arr)****PHP4**

配列arrからすべての値を配列として返します。

**int array\_walk(array arr, string func [, mixed data])**3.0.3 **PHP3**  
**PHP4**

配列の各要素の値を最初の引数、キーを二番目の引数として関数funcを実行します。funcの追加の引数としてdataを指定可能です。arrの値を変更したい場合には参照渡しとします。

```
$names = array ("a"=>"taro", "b"=>"hanako", "c"=>"yamada");

function func1 ($val, $key){
    print "$val<BR>¥n";
}

function func2 (&$val, $key, $data){
    $val = $data;
}

array_walk ($names, 'func1'); // taro,hanako,yamadaが出力されます
array_walk ($names, 'func2', "anonymous");
// $namesの各要素にanonymousが代入されます
```

---

**void arsort(array arr)**


---

**PHP3**  
**PHP4**

キーと要素との関係を維持しつつ配列を降順にソートします。

```
$names = array ("c"=>"yamada", "a"=>"taro", "b"=>"hanako");
arsort ($names);
for( reset($names); $key = key($names); next($names)) {
    print $names[$key] . "<BR>¥n"; // "yamada","taro","hanako"の順に出力されます
}
```

---

**void asort(array arr)**


---

**PHP3**  
**PHP4**

キーと要素との関係を維持しつつ配列を昇順にソートします。

---

**array compact (mixed varnames [, ...])**


---

**PHP4**

変数名の文字列または変数名の配列を引数とし、その変数名をキー、その変数の値を要素の値とする配列を返します。引数の数は可変です。

```
$name1 = "Taro"; $name2 = "Hanako"; $name3 = "Jiro";
$a = array ("name1","name2");
$b = compact ("name3",$b);
// $b["name1"]="Taro",$b["name2"]="Hanako",$b["name3"]="Jiro"となります
```

---

**int count(mixed var)**


---

変数var( 通常は配列 )に含まれる要素の数を返します。

**PHP3**  
**PHP4**


---

**mixed current(array arr)**


---

配列のカレントの要素の値を返します。内部ポインタが最終要素の位置を超えている場合や、空文字列( "" )または0の場合にfalseを返します。

**PHP3**  
**PHP4**


---

**array each(array arr)**


---

配列から次のキー / 値のペアを返し、内部ポインタを進めます。キーを要素 key( および0 ) 値をvalue( および1 )に有する配列を返します。

**PHP3**  
**PHP4**

```
// POSTメソッドで渡された値を表示します
reset($HTTP_POST_VARS);
while (list ($key, $val) = each ($HTTP_POST_VARS)) {
    print "$key => $val<br>";
}
```

---

**end(array arr)**


---

配列の内部ポインタを最後の要素まで進めます。

**PHP3**  
**PHP4**


---

**void extract (array arr [, int type [, string prefix]])**


---

3.0.7

**PHP3**  
**PHP4**

シンボルテーブルに変数をインポートします。配列arrのキーを変数名、値を変数の値とします。typeにより既存の変数名と衝突した場合の処理( 以下 )を指定します。

EXTR\_OVERWRITE : 衝突時に既存の変数は上書きされます。  
 EXTR\_SKIP : 衝突時に既存の変数は上書きされません。  
 EXTR\_PREFIX\_SAME : 衝突時にprefixを前につけた変数とします。  
 EXTR\_PREFIX\_ALL : すべての変数の前にprefixをつけます。

```
$name = "taro";
$a = array ("adrs" => "tokyo", "name" => "hanako");
extract ($a, EXTR_PREFIX_SAME, "n"); // $nameは衝突するので"n_"が付加されます
print "$adrs, $name, $n_name¥n"; // "tokyo, taro, hanako"が出力されます
```

---

**bool in\_array (mixed srch, array arr)**

---

**PHP4**

配列arr内でsrchを検索し、見つかった場合にtrue、それ以外の場合にはfalseを返します。

---

**mixed key(array array)**

---

**PHP3****PHP4**

配列の内部ポインタが指す要素のキーを返します。

---

**int krsort(array array)**

---

**PHP3**

3.0.13

**PHP4**

配列をキーで逆順にソートします。

---

**int ksort(array array)**

---

**PHP3****PHP4**

配列をキーでソートします。

```
$a = array ("c">="taro", "a">="hanako", "b">="jiro");  
ksort ($a); // キーによりソートします  
for(reset($a); $key = key($a); next($a)) { // キーの順番(a,b,c)に表示します  
    print "a[$key] = " . $a[$key] . "<BR>¥n";  
}
```

---

**void list(...)**

---

**PHP3****PHP4**

同時に複数の変数に値を代入するために使用します。通常、配列を返す関数とともに使用されます。

```
$result = mysql($conn, "select day, tenki, ondo from otenki"); // クエリを実行  
while (list($day, $tenki, $ondo) = mysql_fetch_row($result)) {  
    print "$day : $tenki : $ondo<BR>¥n"; // 日付、天気、温度を表示します  
}
```

---

**mixed next(array arr)**

---

**PHP3****PHP4**

配列の内部ポインタを次に進め、移動後の要素( 終端を超える場合はfalse )を返します。

---

**mixed pos(array arr)**


---

配列においてカレントの位置にある要素を返します。

**PHP3**  
**PHP4**


---

**mixed prev(array arr)**


---

配列の内部ポインタをひとつ戻し、移動後の要素( 始端を超える場合はfalse )を返します。

**PHP3**  
**PHP4**  
**F**


---

**array range(int low, int high)**


---

[ low,high ]の範囲の整数を値とする配列を生成します。

3.0.8

**PHP3**  
**PHP4**  
**F**

```
$a = range(1,20); // $a[0]=1,$a[1]=2,...,$a[19]=20 となります。
```

---

**mixed reset(array arr)**


---

配列の内部ポインタを先頭に戻し、先頭の要素の値を返します。

**PHP3**  
**PHP4**  
**F**


---

**int rsort(array arr)**


---

配列を降順にソートします。

**PHP3**  
**PHP4**  
**T F**

```
$a = array("taro","hanako","jiro");
rsort($a); // 配列$aを降順にソートします
// taro,jiro,hanakoの順番に表示されます
reset($a);
while(list($key, $value) = each($a)) {
    print "$key => $value<BR>¥n";
}
```

---

**int shuffle(array arr)**


---

配列の順番をランダムに並べ替えます。使用前にsrand()関数による乱数の初期化が必要です。

3.0.8

**PHP3**  
**PHP4**  
**T F**


---

**int sizeof(array arr)**


---

配列の要素数を返します。

---

**int sort(array arr)**

---

**PHP3**  
**PHP4**  
**T F**

配列を昇順にソートします。

---

**int uasort(array arr, function func)**

---

**PHP3** 3.0.4  
**PHP4**  
**T F**

配列の要素をユーザ定義の比較関数funcを用いてソートします。

---

**int uksort(array arr, function func)**

---

**PHP3** 3.0.4  
**PHP4**  
**T F**

ユーザ定義の比較関数funcにより配列をキーでソートします。

```
// (逆順にソートする)比較関数を定義
function fcomp($x, $y) {
    if ($x == $y) return 0;
    return ($x > $y) ? -1 : 1;
}

$a = array (5 => "five", 3 => "three", 20 => "twenty");
uksort($a, fcomp); // ユーザ関数によりソート
// 20=>twenty, 5=>four, 3=>three の順に表示されます
reset($a);
while(list($key, $value) = each($a)) {
    print "$key => $value<BR>¥n";
}
```

---

**int usort(array arr, function func)**

---

**PHP3** 3.0.3  
**PHP4**  
**T F**

ユーザ定義の比較関数funcにより配列を値でソートします。



## 2.2 正規表現関数

PHPでは、POSIX互換の正規表現、Perl互換の正規表現を使用可能です。

また、国際化版PHPによりマルチバイト対応の正規表現(第7章を参照)も使用可能です。

```
int ereg(string regex, string str [, array regs])
```

PHP3  
PHP4

正規表現により大文字・小文字を区別して検索し、マッチした場合にtrue、それ以外の場合にfalseを返します。

regsを指定した場合、regexpの()でくられた部分にマッチした部分文字列が順次代入され、\$regs[0]はstr、\$regs[i]はi番目の括弧にマッチした文字列となります。

```
$date = '1999-5-29';
if ( ereg("([0-9]{4})-([0-9]{1,2})-([0-9]{1,2})", $date, $regs )) {
    print "$regs[3].$regs[2].$regs[1]"; // 29.5.1999が表示されます
}
```

```
string ereg_replace(string regex, string rstr , string str)
```

PHP3  
PHP4

正規表現により文字列を検索し、マッチした文字列を文字列rstrで置換します。regex内に()でくられた文字列は、rstrの中で¥¥(i<=9、¥¥0はstr)で指定できます。

()は入れ子にすることができ、この場合、左括弧(の位置で順番が決まります。

```
$a = "Your name is Taro";
print ereg_replace ("Taro","Hanako",$a); // "Your name is Hanako"と表示します
print ereg_replace ("(Taro)","Yamada ¥¥1",$a);
// "Your name is Yamada Taro"と表示します
```

```
int eregi(string regex, string str [, array regs])
```

PHP3  
PHP4

ereg()と動作は同じですが、大文字・小文字の区別を行いません。

```
string eregi_replace(string regex, string rstr, string str)
```

PHP3  
PHP4

ereg\_replace()と動作は同じですが、大文字・小文字の区別を行いません。

---

```
array split(string regex, string str [, int lim])
```

---

**PHP3**  
**PHP4**  
**F**

正規表現により文字列を最大limの要素に分割し、配列として返します。

```
// $name_lineの行の先頭から5つのフィールドを取得します
$name_line='taro:hanako:jiro:saburo:sirou:gorou';
$name = split(":", $name_line, 5);
```

---

```
array preg_grep(string regex, array arr)
```

---

**PHP4**  
**F**

配列からPerl互換の正規表現にマッチする要素のみを返します。

---

```
int preg_match(string regex, string str [,array regs])
```

---

**PHP3** 3.0.9  
**PHP4**  
**F**

Perl互換の正規表現にマッチする場合にtrueを返します。regsにマッチした文字列を格納します。

---

```
int preg_match_all(string regex, string str [,array regs [,int order]])
```

---

**PHP3** 3.0.9  
**PHP4**  
**F**

Perl互換の正規表現にマッチする場合にtrueを返します。マッチした文字列をregsに格納する順序を、orderで指定可能です。

---

```
string preg_quote(string str [,string delim])
```

---

**PHP3** 3.0.9  
**PHP4**  
**F**

Perl正規表現における特殊文字の前に¥を挿入して返します。delimを指定した場合にはその文字の前にも挿入されます

---

```
mixed preg_replace(mixed regex, mixed rstr, mixed str)
```

---

**PHP3** 3.0.9  
**PHP4**  
**F**

文字列str(配列も使用可)中のPerl正規表現にマッチする文字列をrstrに置換して返します。

---

```
array preg_split(string regex, string str [,int lim, [int flags]])
```

---

**PHP3** 3.0.9  
**PHP4**  
**F**

Perl互換の正規表現により文字列を最大limの要素に分割し、配列として返します。

---

**string sql\_regcase(string str)**

---

大文字小文字を区別せずに文字列strにマッチする正規表現を返します。

**PHP3**  
**PHP4**

```
print sql_regcase("Taro"); // '[Tt][Aa][Rr][Oo]'を出力します
```

## 2.3 文字列関数

---

**string addcslashes(string str, string clist)**

---

文字列の中にある clist で指定した文字をバックスラッシュでエスケープして返します。  
clist には ASCII コードをでき、¥0..¥31 のように範囲を指定することも可能です。

**PHP4**

---

**string addslashes(string str)**

---

データベースへのクエリの際にクォートが必要な文字( '、"、¥、¥0 )をバックスラッシュでエスケープした文字列を返します。

**PHP3**  
**PHP4**

---

**string chop(string str)**

---

文字列から最後尾のホワイトスペースを取り除いて返します。

**PHP3**  
**PHP4**

```
$line = "lemon apple ";  
print chop($line); // "lemon apple" が出力されます
```

---

**string chr(int ascii)**

---

ASCII コードで指定した1文字からなる文字列を返します。

**PHP3**  
**PHP4**

```
$str .= chr(27); // $strの最後にエスケープ文字を付加します
```

---

```
string chunk_split(string str [, int len][, string end])
```

---

**PHP3** 3.0.6  
**PHP4**

文字列strにlen( デフォルトは76 )文字ごとに文字列end( デフォルトは¥r¥n )を挿入して返します。

---

```
mixed count_chars (string str [, int mode])
```

---

**PHP4**

各バイト値( 0..255 )が存在する数を数え、以下のmodeに基づき値を返します。

- 0 : 各バイト値をキー、各バイトの出現回数を値とする配列( デフォルト )
- 1 : 0と同じだが、出現回数>0のバイト値だけを抽出。
- 2 : 0と同じだが、出現回数がゼロのバイト値だけを抽出。
- 3 : 使用されているすべてのバイト値を有する文字列を返す。
- 4 : 使用されていないすべてのバイト値を有する文字列を返す。

---

```
string crc32(string str)
```

---

**PHP4** 4.0.1

文字列からCRC32多項式を計算し、文字列として返します。

---

```
string crypt(string str [, string salt])
```

---

**PHP3**  
**PHP4**

文字列をcrypt関数により暗号化します。引数saltで暗号のベースを指定可能です。saltの長さはシステムに依存し、定数CRYPT\_SALT\_LENGTHにより取得可能です( DESの場合 : 2、MD5の場合 : 12 )。

---

```
echo(string arg1 [, string argn]... )
```

---

**PHP3**  
**PHP4**

引数を出力します。echo()は言語の一部であり、括弧を省略可能です。

```
echo "Hello World";      // 'Hello World' を出力します
```

---

```
array explode(string sep, string str [, int lim])
```

---

**PHP3**  
**PHP4**

文字列strを文字列sepにより分割し、結果を配列として返します。引数limにより要素数の上限を指定可能で、この場合、最後の要素に残りの文字列が含まれます。

```
$a = "Yamada Taro";
$b = explode(" ", $a); // $b[0]=>Yamada, $b[1]=>Taroとなります
```

## void flush(void)

出力バッファをフラッシュし、ブラウザに出力します。

PHP3  
PHP4

## string get\_html\_translation\_table ([int table])

HTMLにおける特殊文字やエンティティの変換テーブルを返します。引数tableにはHTML\_SPECIALCHARS(デフォルト) HTML\_ENTITIESを指定します。

PHP4

```
$trans = get_html_translation_table (HTML_ENTITIES);
// HTMLエンティティの変換データを取得
$a = "Taro & Hanako";
$b = strtr ($a, $trans); // 「Taro & Hanako」となります
```

## array get\_meta\_tags(string file [, int path])

3.0.4

PHP3  
PHP4

ファイルfileからmetaタグ<meta ...>をすべて取得し、配列として返します。nameプロパティの値がキー、contentプロパティの値が値となります。pathに1を指定した場合、標準のインクルードパスから読み込まれます。nameプロパティにおいて空白などの特定の文字は \_ で置換され、残りは小文字に変換されます。

例として以下のファイルtags.incを処理することを考えます。

```
tags.inc-----
<head>
<meta name="author" content="Taro">
<meta name="tags" content="php document">
</head>
tags.inc-----

$meta = get_meta_tags ("tags.inc"); // metaタグを取得
print $meta["author"]; // 'Taro'が出力されます
```

## string htmlentities(string str)

文字列strの文字をすべてHTMLエンティティに変換して返します。現在、ISO-8859-1文字セットが使われます。

PHP3  
PHP4

---

**string htmlspecialchars(string str)**

---

**PHP3****PHP4**

HTMLにおける特殊文字( &、"、<、> )をマークアップ文字( &amp;、&quot;、&lt;、&gt; )に変換して返します。ユーザ入力をHTMLに変換する際に便利です。

---

**string implode(array arr, string str)**

---

**PHP3****PHP4**

配列arrの要素を文字列strをはさんでひとつの文字列にして返します。

```
$array = array('taro','hanako','jiro');  
print implode($array, ":"); // taro:hanako:jiroが出力されます
```

---

**string join(array pieces, string glue)**

---

**PHP3****PHP4**

implode()の別名です。

---

**string ltrim(string str)**

---

**PHP3****PHP4**

文字列の先頭から空白文字( ¥n、¥r、¥t、¥v、¥0 )を取り除いたものを返します。

---

**string md5(string str)**

---

**PHP3****PHP4**

文字列のMD5ハッシュ値を計算して返します。

---

**string nl2br(string str)**

---

**PHP3****PHP4**

文字列に含まれるすべての改行文字の前に<BR>を挿入して返します。

---

**int ord(string str)**

---

**PHP3****PHP4**

文字列の先頭の文字のASCIIコードを返します。

```
// 先頭の文字が'¥n' (ASCIIコード: 10) の場合に表示します  
if (ord ($str) == 10) {  
    print "¥$strの先頭は改行文字です。";  
}
```

**void parse\_str(string str)**

文字列をURL 引数と同様に処理し、変数を作成します。

**PHP3**  
**PHP4**

```
$str = "a=hanako&b[]=yamada+taro&b[]=jiro";
parse_str($str); // $strを処理
print "$a<BR>¥n"; // "hanako"を出力します
print $b[0] . " and " . $b[1]; // "yamada taro and jiro"を出力
```

**print(string str)**

文字列strを出力します。

**PHP3**  
**PHP4**
**int printf(string format, mixed [args] ...)**

フォーマットformatに基づき、引数を出力します。フォーマット指定はsprintf()と同じです。

**PHP3**  
**PHP4**
**string quoted\_printable\_decode(string str)**

quoted printableでエンコードされた文字列strをデコードし、8ビットバイナリ文字列として返します。

**PHP3**  
**PHP4**
**string quotemeta(string str)**

文字「. ¥ + \* ? [ ^ ] ( \$ )」の前にバックスラッシュ(¥)を付加した文字列を返します。

**PHP3**  
**PHP4**  
**F**
**string rawurldecode(string str)**

URLエンコードされた文字列strをデコードして結果を返します。

**PHP3**  
**PHP4**

```
$url = 'name%20taro%40tokyo';
print rawurldecode($url); // 'name taro@tokyo'が出力されます
```

**string rawurlencode(string str)**

文字列をRFC1738に基づきURLエンコードして返します。日本語等を含むパラメータをURLに指定する場合には、文字化けを防ぐためにURLエンコードする必要があります。

**PHP3**  
**PHP4**

```
$url = "test.php?foo=" . rawurlencode("日本語");  
// URLエンコードされた文字列がパラメータとして出力されます  
print "<A HREF=¥$url¥">リンク</A>";
```

---

**string setlocale(string category, string locale)**

---

**PHP3**  
**PHP4**  
**F**

指定したカテゴリに関するロケールを設定し、ロケール値を返します。それ以外の場合に false を返します。category には以下の有効範囲を指定します。

LC\_ALL : 以降のものすべて。  
LC\_CTYPE : 文字の分類と変換。たとえば strtoupper()。  
LC\_NUMERIC : 数字の区切り文字用。  
LC\_TIME : 日時。strftime() におけるフォーマットに適用します。

locale に 0 を指定した場合、現在のロケールを返します。

---

**string sprintf(string format, mixed [args] ...)**

---

**PHP3**  
**PHP4**

フォーマットを指定して引数 args を文字列に出力します。フォーマットには、% で始まるフォーマット指定子を指定可能です。詳細は、C 言語の関数 sprintf() のマニュアルを参照してください。

```
$year = 2000; $month = 9; $day = 1;  
print sprintf("%04d-%02d-%02d", $year, $month, $day);  
// '2000-09-01' が出力されます
```

---

**int strcasecmp(string str1, string str2)**

---

**PHP3** 3.0.2  
**PHP4**

大文字・小文字を区別せずに文字列比較を行い、str1>str2 の場合に正、str1<str2 の場合に負、等しい場合に 0 を返します。バイナリ文字列にも使用可能です。

---

**string strchr(string haystack, string needle)**

---

**PHP3**  
**PHP4**

関数 strstr() の別名です。



```
int strcmp(string str1, string str2)
```

大文字・小文字を区別すること以外は、strcasecmp()と同じです。

**PHP3**  
**PHP4**

```
int strcspn(string str, string mask)
```

3.0.3

**PHP3**  
**PHP4**

文字列においてマスク文字列maskの文字がどれも含まれていない最初のセグメントの長さを返します。

```
string strip_tags(string str [, string tags])
```

3.0.8

**PHP3**  
**PHP4**

文字列からHTMLタグおよびPHPタグを取り除いて返します。引数tagsで削除しないタグを指定可能です。

```
string stripslashes(string str)
```

**PHP4**

文字列からバックスラッシュを取り除いて返します。C言語のエスケープ文字(¥n、¥r、...)はそのままとなります。

```
string stripslashes(string str)
```

**PHP3**  
**PHP4**

文字列からバックスラッシュを取り除いて返します。¥¥は、¥に変換されます。

```
string stristr(string str)
```

3.0.6

**PHP3**  
**PHP4**

大文字・小文字を区別しないこと以外は、strstr()と同じです。

```
int strlen(string str)
```

**PHP3**  
**PHP4**

文字列の長さを返します。

```
int strpos(string str, string key [, int offset])
```

**PHP3**  
**PHP4**

文字列でキー文字列(key)が最初に見つかった位置(見つからない場合はfalse)を返します。offsetにより検索開始位置を指定可能です。

---

**string strrchr(string str, string key)**

---

**PHP3****PHP4**

文字列でキー文字( key )が最後に現れた位置からの部分文字列( 見つからない場合はfalse )を返します。key は最初の文字のみが使用されます。

```
// $PATH 中の最後のディレクトリを取得します
$dir = substr(strrchr($PATH, ":"), 1);
```

---

**string str\_repeat(string str, int n)**

---

**PHP4**

文字列をn回反復した文字列を返します。

---

**string strrev(string str)**

---

**PHP3****PHP4**

文字列を逆順にして返します。

---

**int strrpos(string str, string key)**

---

**PHP3****PHP4**

キー( key )が最後に現れる位置( 見つからない場合はfalse )を返します。key は最初の文字のみが使用されます。

---

**int strstrn(string str, string mask)**

---

**PHP3**

3.0.3

**PHP4**

すべて文字列maskの文字からなる最初のセグメントの長さを返します。

---

**string strstr(string str, string key)**

---

**PHP3****PHP4**

キー( key )が最初に現れる場所から文字列の終わりまで( 見つからない場合はfalse )を返します。

---

**string strtok([string str,] string sep)**

---

**PHP3****PHP4**

文字列をセパレータ( sep )によりトークンに分割して返します。2回目以降の分割ではセパレータのみを指定します。

```
$a = "This is a pen";
$tok = strtok($a, " "); // 空白文字によりトークンに分割します
while($tok) {           // トークンが存在する間、ループします
    print "$tok:";       // 'This:is:a:pen' が出力されます
    $tok = strtok(" ");
}
```

---

**string strtolower(string str)**

文字列のアルファベットをすべて小文字に変換して返します。

**PHP3**  
**PHP4**


---

**string strtoupper(string str)**

文字列のアルファベットをすべて大文字に変換して返します。

**PHP3**  
**PHP4**


---

**string str\_replace(string key, string rstr, string str)**

文字列strの中の文字列keyをすべて文字列rstrに置換して返します。

3.0.6

**PHP3**  
**PHP4**

```
// "<body text=black>"を出力します
print str_replace("%body%", "black", "<body text=%body%>");
```

---

**string strstr(string str, string from, string to)**

文字列からfromの文字を検索し、toの対応する文字に置換して返します。fromとtoの長さが異なる場合、長い方の余分な文字は無視されます。

**PHP3**  
**PHP4**


---

**string substr(string str, int pos [, int len])**

文字列の位置pos(先頭文字の位置が0)からlenバイト分の文字列を返します。posに負の数を指定した場合、末尾からの位置となります。

**PHP3**  
**PHP4**

```
print substr("abcdef", 1); // "bcdef"を出力します
print substr("abcdef", 1, 3); // "bcd"を出力します
print substr("abcdef", -3, 1); // "d"を出力します
```

---

```
string substr_replace(string str, string rpos, int pos [, int len])
```

---

**PHP4**

文字列の位置pos(先頭文字の位置が0)からlenバイト分の文字列をrposで置換して返します。posに負の数を指定した場合、末尾からの位置となります。

```
print substr_replace("abcdef","pp",3,2); // "abcppf"を出力します
```

---

```
string trim(string str)
```

---

**PHP3**

**PHP4**

文字列の先頭および末尾にある空白文字(¥n、¥r、¥t、¥v、¥0、' ')を取り除いた文字列を返します。

---

```
string ucfirst(string str)
```

---

**PHP3**

**PHP4**

文字列の最初の文字がアルファベットの場合、大文字に変換します。

---

```
string ucwords(string str)
```

---

**PHP3**

3.0.3

**PHP4**

文字列の各単語の最初の文字を大文字に変換します(アルファベットの場合のみ)

## 2.4 変数操作関数

---

```
mixed call_user_func (string func [, mixed param])
```

---

**PHP3**

3.0.3

**PHP4**

ユーザ定義関数funcをコールします。関数のパラメータparamを指定可能です。

---

```
int empty(mixed var)
```

---

**PHP3**

**PHP4**

変数が設定され、ゼロ以外の値を有する場合にfalse、それ以外の場合にはtrueを返します。

**string gettype(mixed var)**

変数の型を以下の文字列で返します。

boolean、integer、resource、double、string、array、object、NULL、unknown type

個々の型に関して確認を行う際には、表 3-4 の関数を使用します。

表 3-4 型の確認用関数

関数	動作および戻り値
int is_array(mixed var)	引数が配列の場合に true、それ以外は false を返す
int is_bool(mixed var)	引数が論理値の場合に true、それ以外は false を返す <sup>*3</sup>
int is_double(mixed var)	引数が double の場合に true、それ以外は false を返す
int is_float(mixed var)	is_double() の別名
int is_int(mixed var)	is_long() の別名
int is_integer(mixed var)	is_long() の別名
int is_long(mixed var)	引数が long 整数の場合に true、それ以外は false を返す
int is_numeric(mixed var)	引数が数値の場合に true、それ以外は false を返す <sup>*3</sup>
int is_object(mixed var)	引数がオブジェクトの場合に true、それ以外は false を返す
int is_real(mixed var)	is_double() の別名
int is_resource(mixed var)	引数がリソースの場合に true、それ以外は false を返す <sup>*3</sup>
int is_string(mixed var)	引数が文字列の場合に true、それ以外は false を返す

PHP3  
PHP4

<sup>\*3</sup>

PHP4 以降。

**int isset(mixed var)**

変数が存在すれば true、そうでなければ false を返します。

```
$a = "test";
print isset($a) ? "定義" : "未定義"; // "定義"を表示します
unset($a); // 変数 $a の割当を解除します
print isset($a) ? "定義" : "未定義"; // "未定義"を表示します
```

PHP3  
PHP4

**void print\_r(mixed exp)**

変数の値に関する情報を表示します。

```
$a = array("taro","hanako");
print_r ($a); // Array ( [0] => taro [1] => hanako ) が出力されます
```

PHP4

---

**int settype(string var, string type)**


---

**PHP3**  
**PHP4**  
**T F**

変数の型を指定した値(表 3-4 )に変更します。

integer、double、string、array、object、boolean、resource

型を変更せずに指定した型で値を得るには、表 3-5 の関数を使用します。

表 3-5 指定した型で値を得る関数

関数	動作および戻り値
double doubleval(mixed var)	double として値を取得
int intval(mixed var [,int base])	整数として値を取得。基数 base を指定可能
string strval(mixed var)	文字列として値を取得

---

**int unset(mixed var)**


---

**PHP3**  
**PHP4**

変数を破棄し、true を返します。

```
unset ($a); // 変数$aを破棄します
unset ($b['date']); // 連想配列$bのキー'date'およびその値を破棄します
```

---

**void var\_dump(mixed exp)**


---

**PHP3** 3.0.5  
**PHP4**

指定した式( exp )について型や値を含む構造化された情報を返します。

```
$a = array("taro","hanako");
var_dump ($a);
// array(2) { [0]=> string(4) "taro" [1]=> string(6) "hanako" } が出力されます
```

## 2.5 オブジェクト・クラス関連

---

**string get\_class (object obj)**


---

**PHP4**  
**F**

オブジェクトobjをインスタンスとするクラスの名前を返します。

---

**string get\_parent\_class (object obj)**

---

オブジェクトobjをインスタンスとするクラスの、親クラスの名前を返します。

**PHP4  
F**

---

**array get\_class\_methods (string class\_name)**

---

クラスメソッドの名前を配列として返します。

**PHP4  
F**

---

**array get\_class\_vars (string class\_name)**

---

クラスのデフォルトプロパティを配列として返します。

**PHP4  
F**

---

**array get\_object\_vars (object obj)**

---

オブジェクトのプロパティを配列として返します。

**PHP4  
F**

---

**bool is\_subclass\_of(object obj, string superclass)**

---

オブジェクトがsuperclassのサブクラスの場合にtrueを返します。

**PHP4  
F**

---

**bool class\_exists(string class)**

---

指定したクラスが定義されている場合にtrueを返します。

**PHP4  
F**

---

**bool method\_exists(object object, string method)**

---

指定したメソッドが定義されている場合にtrueを返します。

**PHP4  
F**

---

**array get\_declared\_classes(void)**

---

カレントのスクリプトで宣言されているクラスの名前を配列として返します。

**PHP4  
F**



# Hypertext Preprocessor



## Chapter-3

## PHP 関数

## ファイル関連

### 3.1 ファイルシステム関数

**string basename (string path)**

PHP3  
PHP4

パス名からファイル名を生成します。パス区切り文字は / (Microsoft Windows では ¥ も使用可) です。

```
$path = "/home/taro/public_html/index.php";  
print basename ($path); // index.php が出力されます
```

**int chgrp (string filename, mixed group)**

PHP3  
PHP4  
T F

ファイルのグループを group に変更します。グループ変更の権限が必要です。

```
chgrp("/somedir/somefile", "guest"); // guest グループに変更します
```

**int chmod(string filename, int mode)**

PHP3  
PHP4  
T F

ファイルのモードを mode に変更します。mode は先頭に 0 をつけて 8 進数で指定します。

```
chmod ( "/somedir/somefile", 0755 ); // 8 進数; 正しいモードの値
```

**int chown(string filename, mixed user)**

PHP3  
PHP4  
T F

ファイル filename の所有者をユーザ user に変更します。ユーザ変更の権限が必要です。

```
chown ( "/somedir/somefile", "guest" ) // ユーザ guest に所有者を変更します
```



**void clearstatcache(void)**

システムコールstat()、lstat()により得られるファイル情報のキャッシュをクリアします。

PHP3  
PHP4

**int copy(string source, string dest)**

ファイルのコピーを作成します。

PHP3  
PHP4  
T F

```
copy("a.php","b.php"); // a.phpをb.phpにコピーします
```

**string dirname (string path)**

パス名からディレクトリの名前を返します。

PHP3  
PHP4

```
$path = "/etc/passwd";  
print dirname($path); // "/etc"が出力されます
```

**float diskfreespace(string dir)**

指定したディレクトリで利用可能なバイト数を返します。

3.0.7  
PHP3  
PHP4

```
$df = diskfreespace ("/"); // $dfは"/"で利用可能なバイト数となります
```

**int fclose(int fp)**

ファイルを閉じます。

PHP3  
PHP4  
T F

**int feof(int fp)**

ファイルポインタがファイルの終端に達していればtrueを返します。

PHP3  
PHP4  
F

**string fgetc(int fp)**

ファイルから1文字読み出し、文字列として返します。

PHP3  
PHP4

```
print fgetc ($fp); // ファイルから1文字読み出します
```

---

**array fgetcsv(int fp,int len [,string delim])**


---

**PHP3** 3.0.8  
**PHP4**  
**F**

CSV ファイルから1行取得し、各レコードと要素とする配列として返します。ファイルの終端の場合にはfalseを返します。行の最大長をlenで指定します。delimによりデリミタを指定可能(デフォルトは',')です。

```
$fp = fopen ("a.csv","r"); // CSVファイルをオープン
// 行を配列として取得し、出力する
while ($a = fgetcsv ($fp, 1000)) {
    for ($k=0; $k<count($a); $k++){
        print $a[$k] . "<BR>";
    }
}
fclose ($fp);
```

---

**string fgets(int fp, int len)**


---

**PHP3**  
**PHP4**  
**F**

ファイルから最大len-1バイト読み込み、文字列として返します。読み込みは、改行文字またはEOFを検出したか、len-1バイトを読み出した時点で終了します。

```
$fd = fopen ("a.txt", "r");
while (!feof($fd)) { // EOFに達したどうかを確認
    print fgets($fd, 4096); // 行を読み込み、出力します
}
fclose ($fd);
```

---

**string fgetss(int fp, int length [, string tags])**


---

**PHP3**  
**PHP4**

fgets()と機能は同じですが、HTMLおよびPHPのタグを取り除きます。削除したくないタグをtagsで指定可能です。

---

**array file(string filename [, int use\_inc])**


---

**PHP3**  
**PHP4**

readfile()と機能は同じですが、各行を配列の要素として返します。use\_incに1を指定することにより、include\_pathで指定したパスの検索も行われます。

---

**int file\_exists(string filename)**


---

**PHP3**  
**PHP4**  
**F**

ファイルfilenameが存在する場合にtrueを返します。

**bool flock(int fp, int op)**

3.0.7 **PHP3**  
**PHP4**  
**T F**

ファイルのロックを設定または解除します。op には、以下の値を指定します。

LOCK\_SH または 1 : 共有ロック : 読み込み時  
LOCK\_EX または 2 : 排他的ロック : 書き込み時  
LOCK\_UN または 3 : ロック解除

非ブロックモードを指定するには op に LOCK\_NB または 4 を加えます。

```
$fp = fopen("a.dat", "w");
flock($fp, 2); // 排他的ロックを行います
fwrite($fp, "this is test.¥n");
flock($fp, 3); // ロックを解放します
fclose($fp);
```

**int fopen(string filename, string mode [, int use\_inc])**

**PHP3**  
**PHP4**  
**F**

ファイル(または URL) をモード mode でオープンし、ポインタを返します。use\_inc に 1 を指定することにより、include\_path で指定したパスの検索も行われます。

ファイル名が http:// で始まっている場合は HTTP、ftp:// で始まっている場合は FTP の接続がオープンされます。

mode には、r (読み込み) r+ (読み書き) w (書き込み) w+ (読み書き) a (追記) a+ (読み込みおよび追記) b (バイナリモード、Windows のみ) を指定します。

```
$fp = fopen ("test.txt", "r"); // 読み込みのみでオープン
$fp = fopen ("http://www.php.net/", "r"); // HTTP 接続で読み込みのみでオープン
```

**int fpassthru(int fp)**

**PHP3**  
**PHP4**  
**F**

ファイルからデータを EOF まで読み込み、標準出力に書き出します。ファイルは読み込み終了時に自動的にクローズされます。

**int fputs(int fp, string str [, int len])**

**PHP3**  
**PHP4**

fwrite() の別名です。

---

```
string fread(int fp, int len)
```

---

**PHP3**  
**PHP4**

ファイルから最高lenバイト読み込み、文字列として返します。読み込みはlenバイト読み込まれたか、EOFに達した時点で終了します。

---

```
int fseek(int fp, int offset [,int param])
```

---

**PHP3**  
**PHP4**

ファイルポインタの位置をoffsetに設定します。成功時に0、それ以外の場合に-1を返します。paramには以下の値を指定可能です。

SEEK\_SET : 位置をoffsetバイト目に設定する(デフォルト)

SEEK\_CUR : 現在の位置にoffsetを加えた位置に設定する。

SEEK\_END : ファイル終端にoffsetを加えた位置に設定する。

---

```
int ftell(int fp)
```

---

**PHP3**  
**PHP4**  
**F**

ファイルポインタポインタの位置を返します。

---

```
int ftruncate (int fp, int size)
```

---

**PHP3**  
**PHP4**  
**T F**

ファイルを指定した長さに丸めます。

---

```
int fwrite(int fp, string string [, int len])
```

---

**PHP3**  
**PHP4**

文字列をファイルに書き込みます。文字列の書き込みの完了時に、処理が終了します。lenを指定した場合、最大lenバイトまで書き込まれ、スラッシュ文字の削除は行われません。

---

```
int set_file_buffer (int fp, int buf)
```

---

**PHP3** 3.0.8  
**PHP4**

ファイル書き込み用バッファをbufバイトに設定します。成功時に0、エラー時にEOFを返します。bufに0を指定した場合にはバッファリングされません。

---

```
int link(string file, string link)
```

---

**PHP3**  
**PHP4**  
**T F**

ファイルへのハードリンクlinkを作成します。

```
int mkdir(string path, int mode)
```

ディレクトリpathをモードmodeで作成します。引数modeにより(0を先頭につけた8進数で)アクセス権を指定可能です。

PHP3  
PHP4  
T F

```
mkdir ("/tmp/dir", 0700); // /tmp/dirをアクセス権700で作成します
```

```
int pclose(int fp)
```

パイプを閉じます。パイプにより実行中のプロセスの終了ステータスを返します。

PHP3  
PHP4

```
int popen(string command, string mode)
```

コマンドを実行し、そのプロセスへのパイプをオープン、ファイルポインタを返します。modeには、r(読み込み)またはw(書き込み)を指定します。

PHP3  
PHP4  
F

```
$fp = popen ("/bin/ls", "r"); // /bin/lsを実行し、読み込み用でパイプをオープンします
```

```
int readfile(string filename)
```

ファイル(またはURL)の内容を標準出力に書き出し、読み込んだバイト数を返します。URLに関する処理はfopen()と同じです。

PHP3  
PHP4  
F

```
string readlink(string path)
```

指定したパスのシンボリックリンクに関する情報を返します。

PHP3  
PHP4  
F

```
int rename(string oldname, string newname)
```

oldnameをnewnameにリネームします。

PHP3  
PHP4  
T F

```
int rewind(int fp)
```

ファイルポインタの位置をファイルの先頭にセットします。

PHP3  
PHP4  
T

int rmdir(string dir)

PHP3  
PHP4  
T F

ディレクトリを削除します。削除できるのは、PHP実行ユーザにより削除可能な空のディレクトリのみです。

array stat(string filename)

PHP3  
PHP4

ファイルの情報をstatシステムコールにより取得し、stat構造体の値を表3-6に示す配列として返します。

また、個々のファイルの情報を取得するには表3-7に示す関数、個々のファイルの属性を確認するには表3-8に示す関数を使用します。

表 3-6 stat()関数の返り値

要素	名前	要素	名前
0	デバイス (st_dev)	7	サイズ (バイト数) (st_size)
1	i ノード (st_ino)	8	最終アクセス時刻 (st_atime)
2	i ノードのモード (st_mode)	9	最終更新時刻 (st_mtime)
3	リンク数 (st_nlink)	10	最終変更時刻 (st_ctime)
4	所有者のユーザID (st_uid)	11	ファイル/IOのブロックサイズ *4 (st_blksize)
5	所有者のグループID (st_gid)	12	割り当て済みブロック数 *5 (st_blocks)
6	デバイスタイプ *4 (st_rdev)		

表 3-7 ファイル情報を取得する関数

関数	返り値
int fileatime(string filename)	最終アクセス時刻 (エラー時はfalse)
int filectime(string filename)	最終更新時刻 (エラー時はfalse)
int filegroup(string filename)	所有者のグループID (エラー時はfalse)
int fileinode(string filename)	i-node 番号 (エラー時はfalse)
int filemtime(string filename)	最終更新時刻 (エラー時はfalse)
int fileowner(string filename)	所有者のユーザID (エラー時はfalse)
int fileperms(string filename)	許可属性情報 (エラー時はfalse)
int filesize(string filename)	サイズ (エラー時はfalse)
string filetype(string filename)	型 (エラー時はfalse) *6
int linkinfo(string filename)	デバイス情報 (エラー時はfalse)

表 3-8 ファイル属性確認用関数

関数	返り値
bool is_dir(string filename)	ディレクトリの場合にtrue
bool is_executable(string filename)	実行可能な場合にtrue
bool is_file(string filename)	通常ファイルの場合にtrue
bool is_link(string filename)	シンボリックリンクの場合にtrue
bool is_writeable(string filename)	書き込み可能な場合にtrue
bool is_readable(string filename)	読み込み可能な場合にtrue

\*4  
st\_blksize型をサポートするシステムでのみ有効(それ以外は-1を返します)

\*5  
st\_blocks型をサポートするシステムでのみ有効(それ以外は-1を返します)

\*6  
返される型は、fifo、char、dir、block、link、file、unknownのいずれかです。

---

```
array lstat(string filename)
```

---

ファイルの情報を得ます。動作はstat()と同じですが、シンボリックリンクの場合はリンク元の情報を返します。

PHP3  
PHP4  
F

---

```
int symlink(string filetarget, string link)
```

---

ファイルへのシンボリックリンクlinkを作成します。

PHP3  
PHP4  
T F

---

```
string tempnam(string dir , string prefix)
```

---

prefixを先頭に付加したユニークなテンポラリファイル名をディレクトリdir(存在しない場合はシステムのテンポラリディレクトリ)以下に生成し、パス名を返します。エラー時には空の文字列を返します。

PHP3  
PHP4

```
// ユニークなファイル名"/tmp/F00????"を生成します
$tmpfname = tempnam ("/tmp", "FOO");
```

---

```
int touch(string filename [, int time])
```

---

ファイルの最終更新日をtime(省略時は現在時刻)に変更します。指定ファイルが存在しない場合には新規にファイルが作成されます。

PHP3  
PHP4  
T F

---

```
int umask([int mask])
```

---

umaskをmask & 0777に設定し、元のumask値を返します。maskを省略した場合は現在のumask値を返します。

PHP3  
PHP4

---

```
int unlink(string filename)
```

---

ファイルを削除します。

PHP3  
PHP4  
T F

---

```
int chdir(string dir)
```

---

カレントディレクトリをdirに変更します。

PHP3  
PHP4  
T F

---

**new dir(string dir)**

---

**PHP3****PHP4**

ディレクトリをオープンし、オブジェクトインスタンスを生成します。プロパティ handle (ディレクトリへのポインタ)、path(ディレクトリへのパス)およびメソッド read、rewind、close が使用可能です。

```
$d = dir("/etc"); // ディレクトリ '/etc' へのインスタンスをオープンします
print "パス名: ".$d->path."<BR>¥n"; // '/etc' を出力します
while($entry = $d->read()) {
    print $entry."<BR>¥n"; // '/etc'にあるファイルを表示します
}
$d->close();
```

---

**void closedir(int dir\_handle)**

---

**PHP3****PHP4**

ディレクトリを閉じます。

---

**int opendir(string path)**

---

**PHP3****PHP4****F**

ディレクトリをオープンし、ハンドルを返します。

---

**string readdir(int dir\_handle)**

---

**PHP3****PHP4**

ディレクトリから次のファイルのファイル名を返します。返されるファイル名の順序は不定です。

```
$handle=opendir('.');// カレントディレクトリをオープンします
// カレントディレクトリにあるファイルの名前を出力します
while ($file = readdir ($handle)) {
    print "$file<BR>¥n";
}
closedir($handle);
```

---

**void rewinddir(int dir\_handle)**

---

**PHP3****PHP4**

ディレクトリのストリームをディレクトリの先頭にリセットします。



## 3.2 イメージ関数

PHPでは、JPEG、GIF、PNG、SWF( Shockwave Flash Format )イメージの大きさを調べる関数がサポートされています。GDライブラリ( <http://www.boutell.com/gd/> )により GIF \*7、PNG、JPEG、WBMP、XPM 画像の作成と操作を行ったり、FreeTypeライブラリによりTrueTypeフォントを描画することができます。

UTF-8( UNICODE )で文字を指定することにより日本語TrueTypeフォントを表示することが可能です。国際化パッチ( 第7章を参照 )により、EUC/SJIS/JISコードで書かれた文字をUTF-8形式に変換することができます。

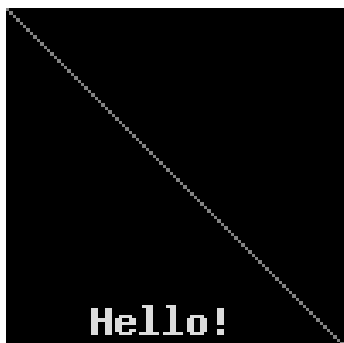
イメージ関数のうち、getimagesize()以外の関数を使用するには、gdサポート機能を有効にしておく必要があります。

イメージ関数における描画の座標系は左上を (0,0) と定義します。

```
<?php
header("Content-type: image/png"); // ヘッダにてPNG画像を指定します
$im = imagecreate (100, 100);
$black = imagecolorallocate ($im, 0, 0, 0);
$red = imagecolorallocate ($im, 255, 0, 0);
$green = imagecolorallocate ($im, 0, 255, 0);
imageline ($im, 0, 0, 100, 100, $red);
imagestring ($im, 3, 25, 85, "Hello!", $green);
imagepng ($im); // 画像をブラウザに出力します
imagedestroy ($im);
?>
```

出力される画像は、図3-1 のようになります。

図 3-1 イメージ関数の出力



\*7  
GDバージョン1.6以降ではライセンスの関係でGIFイメージのサポート機能が削除されているため、関連する機能が利用できません。

---

**array getimagesize(string filename [, array &imageinfo])**

---

**PHP3**  
**PHP4**

画像ファイル( GIF、JPG、PNG、SWF ファイル )filenameの大きさに関する情報を取得し、配列として返します。返される配列\$arrayは以下の要素からなります。

\$array[0] : 画像の幅( ピクセル単位 )  
 \$array[1] : 画像の高さ( ピクセル単位 )  
 \$array[2] : 画像の種類( 1:GIF、2:JPEG、3:PNG、4:SWF )  
 \$array[3] : IMG タグ用フォーマット文字列( "height=xxx width=xxx" )  
 \$array["bits"] : ビット数( SWFまたはJPEGのみ有効 )  
 \$array["channels"] : チャンネル数( JPEGのみ有効 )

```
$size = getimagesize ( "icon.jpg" );
print "<IMG SRC=¥"icon.jpg¥" $size[3]>"; // サイズを指定して画像ファイルを表示します
```

imageinfoパラメータ( オプション )をリファレンスとして指定することにより、画像ファイルに関するそのほかの情報を得ることが可能です。この情報は処理する画像フォーマットに依存します。

---

**int imagearc(int im, int cx, int cy, int w, int h, int s, int e, int cid)**

---

**PHP3**  
**PHP4**  
**T**

座標(cx,cy)を中心とする部分楕円を描画します。引数には楕円の幅 : w、高さ : h、始点角度 : s、終点角度 : eを指定します。

---

**int imagechar(int im, int font, int x, int y, string c, int cid)**

---

**PHP3**  
**PHP4**  
**T**

座標(x,y)に文字列cの最初の文字を描画します。引数にはカラーID( cid )およびフォントID( font )を指定します。フォントIDには、1 ~ 5( 組み込みフォント )またはimageloadfont()関数で取得したフォントIDを指定します。

---

**int imagecharup(int im, int font, int x, int y, string c, int cid)**

---

**PHP3**  
**PHP4**  
**T**

文字を垂直に描画すること以外はimagechar()と同じです。

```
int imagecolorallocate(int im, int red, int green, int blue)
```

指定したRGBを示すカラーIDを返します。

PHP3  
PHP4

```
$blue = imagecolorallocate ($im, 0, 0, 255); // 青色のカラーIDを作成します
```

```
int imagecolordeallocate(int im, int cid)
```

指定したカラーIDを有する色を削除します。

3.0.6  
PHP3  
PHP4  
T F

```
int imagecolorat(int im, int x, int y)
```

座標 (x,y) 位置のピクセルのカラーIDを返します。

PHP3  
PHP4

```
int imagecolorclosest(int im, int red, int green, int blue)
```

指定したRGB値に最も近い画像パレット中の色のIDを取得します。

PHP3  
PHP4

```
int imagecolorexact(int im, int red, int green, int blue)
```

指定したRGB値を有する色のID(存在しない場合は-1)を返します。

PHP3  
PHP4

```
int imagecolorresolve(int im, int red, int green, int blue)
```

指定したRGB値の色または最も近い色のIDを返します。

PHP3  
PHP4

```
int imagegammaconvert (int im, double ig, double og)
```

入力 値(ig)および出力 値(og)に基づき 補正を行います。

3.0.13  
PHP3  
PHP4  
T

```
bool imagecolorset(int im, int cid, int red, int green, int blue)
```

カラーID( cid )を指定したRGB値に設定します。この関数により塗りつぶしの操作を行わずに同じ効果を得ることができます。

PHP3  
PHP4  
F

---

```
array imagecolorsforindex (int im, int cid)
```

---

**PHP3**  
**PHP4**  
**F**

カラー ID を表す red、green、blue をキーとする連想配列を返します。

---

```
int imagecolorstotal (int im)
```

---

**PHP3**  
**PHP4**

画像パレットの色数を返します。

---

```
int imagecolortransparent(int im [, int cid])
```

---

**PHP3**  
**PHP4**

透明色を cid に設定します。指定した( 指定しない場合は現在の )透明色の ID が返されます。

---

```
int imagecopy(int dst_im, int src_im, int dx, int dy, int sx  
              , int sy, int sw,int sh)
```

---

**PHP3** 3.0.6  
**PHP4**  
**T**

イメージ( src\_im )の(sx,sy)を基準として幅sw、高さshの領域をイメージ( dst\_im )の(dx,dy)を基準とし、同じ大きさの領域にコピーします。

---

```
int imagecopyresized(int dst_im, int src_im, int dx, int dy  
                    ,int sx, int sy, int dw, int dh, int sw,int sh)
```

---

**PHP3**  
**PHP4**  
**T**

イメージ( src\_im )の(sx,sy)を基準として幅sw、高さshの領域をイメージ( dst\_im )の(dx,dy)を基準として幅dw、高さdhの領域にコピーします。

---

```
int imagecreate(int w, int h)
```

---

**PHP3**  
**PHP4**

幅w、高さhの空のイメージを作成し、イメージIDを返します。

---

```
int imagecreatefromgif(string filename)
```

---

**PHP3**  
**PHP4**  
**F**

GIF イメージファイル( または URL )から新規イメージを作成し、イメージIDを返します  
( GD-1.6以降では使用不可 )

---

```
int imagecreatefromjpeg(string filename)
```

---

**PHP3** 3.0.16  
**PHP4**  
**F**

JPEG イメージファイル( または URL )から新規イメージを作成し、イメージIDを返します  
( GD-1.8以降が必要 )

## `int imagecreatefrompng(string filename)`

3.0.13

PHP3  
PHP4  
F

PNG イメージファイル(またはURL)から新規イメージを作成し、イメージIDを返します  
(GD-1.6以降が必要)

```
<?php
header("Content-type: image/png"); // ヘッダにてPNG画像を指定します
$file="test.png";
$im = @imagecreatefrompng ($file); // PNGイメージを読み込みます
if ($im == "") { // 読み込みに失敗した場合
    $im = imagecreate (150,30); /* 空の画像を作成 */
    $white = imagecolorallocate ($im,0,0,0);
    imagefilledrectangle ($im,0,0,150,30,imagecolorallocate($im,255,255,255));
    $err = "Error in imagecreatefrompng($file)";
    // エラーメッセージをPNGイメージで出力
    imagestring ($im,1,5,5,$err,imagecolorallocate($im,0,0,0));
} else {
    imagepng($im); // 読み込んだPNGイメージを出力
}
?>
```

## `int imagedashedline(int im, int x1, int y1, int x2, int y2, int cid)`

座標(x1,y1)から(x2,y2)まで色cidで破線を描画します。

PHP3  
PHP4  
T

## `int imagedestroy(int im)`

イメージを保持するメモリを解放します。

PHP3  
PHP4  
T

## `int imagefill(int im, int x, int y, int cid)`

座標(x,y)を含む領域を色cidで塗りつぶします。

PHP3  
PHP4  
T

## `int imagefilledpolygon(int im, array points, int num, int cid)`

色cidで塗りつぶした多角形を作成します。多角形の頂点を配列pointsによりpoints[0] = x0、points[1] = y0、points[2] = x1、points[3] = y1のように指定します。numは頂点の総数です。

PHP3  
PHP4  
T F

```
int imagefilledrectangle(int im, int x1, int y1, int x2, int y2, int cid)
```

PHP3  
PHP4  
T

左上の座標(x1,y1)、右下の座標(x2,y2)で指定した領域に色cidで塗りつぶした矩形を作成します。

```
int imagefilltoborder(int im, int x, int y, int bid, int cid)
```

PHP3  
PHP4  
T

座標(x,y)を含む領域をbidを境界色として色cidで塗りつぶします。

```
int imagefontheight(int font)
```

PHP3  
PHP4

フォントfontの高さをピクセル単位で返します。

```
int imagefontwidth(int font)
```

PHP3  
PHP4

フォントfontの幅をピクセル単位で返します。

```
int imagegif(int im [, string filename])
```

PHP3  
PHP4  
T F

GIF イメージをブラウザ(またはファイルfilename)へ出力します。事前に関数header()でcontent-type image/gifを出力しておく必要があります(GD-1.6以降では使用不可)

```
int imagepng(int im [, string filename])
```

PHP3 3.0.13  
PHP4  
T F

PNG イメージをブラウザ(またはファイルfilename)へ出力します。事前に関数header()でcontent-type image/pngを出力しておく必要があります(GD-1.6以降が必要)

```
int imagejpeg(int im [, string filename][, int quality])
```

PHP3 3.0.16  
PHP4  
T F

JPEG イメージをブラウザ(またはファイルfilename)へ出力します。事前に関数header()でcontent-type image/jpgを出力しておく必要があります(GD-1.8以降が必要)

```
int imageinterlace(int im [, int in])
```

PHP3  
PHP4

インタレースビット(1:有効、0:無効)を設定し、現在のインタレースビットを返します。

```
int imageline(int im, int x1, int y1, int x2, int y2, int cid)
```

座標(x1,y1) から (x2,y2) まで色cidで直線を描画します。

PHP3  
PHP4  
T

```
int imageloadfont(string file)
```

ビットマップフォントをロードし、フォントIDを返します。

PHP3  
PHP4  
F

```
int imagepolygon(int im, array points, int num, int cid)
```

cid色で多角形を描画します。配列pointsは多角形の頂点で、points[0] = x0、points[1] = y0、points[2] = x1、points[3] = y1 ... のように指定します。numは頂点の総数です。

PHP3  
PHP4  
T F

```
int imagerectangle(int im, int x1, int y1, int x2, int y2, int cid)
```

左上の座標(x1,y1)、右下の座標(x2,y2)を指定した領域に色cidの矩形を生成します。

PHP3  
PHP4

```
int imagesetpixel(int im, int x, int y, int cid)
```

座標(x,y)に色cidで点を描画します。

PHP3  
PHP4  
T

```
int imagestring(int im, int font, int x, int y, string s, int cid)
```

座標(x,y)から文字列sを色cidで描画します。

PHP3  
PHP4  
T

```
int imagestringup(int im, int font, int x, int y, string s, int cid)
```

座標(x,y)から垂直に文字列sを色cidで描画します。

PHP3  
PHP4  
T

```
int imagesx(int im)
```

イメージの幅を返します。

PHP3  
PHP4

```
int imagesy(int im)
```

イメージの高さを返します。

PHP3  
PHP4

```
array imagettfbbox(int size, int angle, string fontfile, string text)
```

PHP3  
PHP4  
F

文字列textをTrueTypeフォントfontfile(URLも使用可)を用いて、大きさsize、角度angle(単位: deg)で描画した場合の描画領域の大きさ(Bounding Box)を配列として返します。配列の要素は、四隅を表す座標(左下X座標、左下Y座標、右下X座標、右下Y座標、右上X座標、右上Y座標、左上X座標、左上Y座標)となります。この関数を使用するにはFreeTypeライブラリが必要です。

```
array imagettftext(int im, int size, int angle, int x, int y,
int col, string fontfile, string text)
```

PHP3  
PHP4  
F

文字列textをTrueTypeフォントfontfileを用いて大きさsize、角度angle、色colで描画し、テキストのBounding Boxを示す配列(形式はimagettfbbox()と同じ)を返します。座標(x,y)の描画開始位置を、最初の文字のベースポイント(ほぼ文字の左下角)により指定します。この関数を使用するにはFreeTypeライブラリが必要です。  
GDのインストールについては「Appendix・G PHPとGD」を参照してください。

```
<?php // 日本語TrueTypeフォント出力例
$im = imagecreate (170,50); // イメージ領域を確保します
$blue = imagecolorallocate ($im, 0, 0, 255); // 青色を定義します
$white = imagecolorallocate ($im, 255, 255,255); // 白色を定義します
imagerectangle ($im,0,0,169,49,$blue) ; // 青色の矩形を描画します
// TrueTypeフォントを指定します
$font = "/usr/X11R6/lib/X11/fonts/TrueType/wadalab-gothic.ttf";
$str = "日本語表示";
imagettftext($im,25,0,20,30,$white,$font,$str); // 文字列を描画します
i18n_http_output("pass"); // 文字コード変換を無効にします(国際化版PHPのみ)
header("Content-type: image/png"); // HTTPヘッダにてPNGイメージを指定します
header("Cache-control: no-cache"); // キャッシュを無効にします
imagepng($im); // PNGイメージをブラウザに出力します
imagedestroy($im); // 画像領域を破棄します
?>
```

出力は図3-2のようになります。

図3-2 日本語TrueTypeフォント出力例

日本語表示



---

**array read\_exif\_data(string filename)**


---

**PHP4**  
**F**

指定したJPEGファイルからEXIFヘッダを読み、以下のキーを有する連想配列として返します( EXIFソースによってはサポートされないデータもあります )

"FileName"	"FileDateTime"	"FileSize"
"CameraMake"	"CameraModel"	"DateTime"
"Height"	"Width"	"IsColor" "FlashUsed"
"FocalLength"	"35mmFocalLength"	"RawFocalLength"
"ExposureTime"	"RawExposureTime"	"ApertureFNumber"
"RawApertureFNumber"	"FocusDistance"	"RawFocusDistance"
"CCDWidth"	"Orientation"	"GPSinfo" "ISOspeed"
"ExifVersion"	"Copyright"	"Software" "Comments"
"Thumbnail"	"ThumbnailSize"	

この関数は、PHPでEXIFサポートを有効にした場合のみ使用可能です。

## 3.3 圧縮関数

zlib( <http://www.cdrom.com/pub/infozip/zlib/> )バージョン1.0.9以降を使用することにより、gzipで圧縮されたファイルを透過的に読み書きするための関数がサポートされます。

---

**int gzclose(int fp)**


---

gzip ファイルを閉じます。

**PHP3**  
**PHP4**  
**T F**


---

**int gzeof(int fp)**


---

ファイルポインタがEOFに達したか、エラーの場合にtrue、そのほかの場合にはfalseを返します。

**PHP3**  
**PHP4**


---

**array gzfile(string filename [, int use\_inc])**


---

ファイルfilenameを読み込み、配列として返します。use\_incに1を指定した場合、include\_pathで指定したパスも検索します。

**PHP3**  
**PHP4**

---

**string gzgetc(int fp)**

---

**PHP3**  
**PHP4**

ファイルから( 解凍した )1 文字を読み込み、文字列として返します。EOF の場合に false を返します。

---

**string gzgets(int fp, int len)**

---

**PHP3**  
**PHP4**  
**F**

ファイルから最大 len-1 バイトの文字を読み込み、( 解凍した )文字列として返します。len-1 バイトを読み込むか、改行または EOF に達した場合、読み込みを終了します。

---

**string gzgetss(int fp, int len)**

---

**PHP3**  
**PHP4**

動作は gzgets() と同じですが、読み込んだテキストから HTML / PHP タグをすべて削除します。

---

**int gzopen(string filename, string mode [,int use\_inc])**

---

**PHP3**  
**PHP4**  
**F**

gzip ファイルをオープンし、ファイルポインタを返します。このポインタを指定して読み込むデータは自動的に解凍され、書き込むデータは圧縮されます。mode の指定方法は fopen() と同じですが、圧縮法の指定等の gzip で指定可能なオプションも指定可能です。use\_inc に 1 を指定した場合、include\_path で指定したパスも検索します。

```
$fp = gzopen("/tmp/file.gz", "r"); // "/tmp/file.gz"を読み込み用にオープン
```

---

**int gzpassthru(int fp)**

---

**PHP3**  
**PHP4**  
**F**

ファイルを EOF まで読み込み、標準出力に( 解凍した )結果を出力したあと、ファイルを閉じます。

---

**int gzputs(int fp, string str [,int len])**

---

**PHP3**  
**PHP4**

gzwrite() のエイリアスです。

---

**string gzread(int fp, int len)**

---

**PHP3**  
**PHP4**

最大 len バイトのデータを読み込み、( 解凍された )len バイトのデータが読み込まれたか EOF に達したときに読み込みは終了します。

```
$zd = gzopen("/tmp/test.txt.gz", "r"); // 読み込み用にオープンします
$content = gzread($fd, 1000); // 最大1kバイトのデータを読み込みます
gzclose($fd);
```

**int gzrewind(int fp)**

ファイルポインタの位置をファイルストリームの最初に設定します。

PHP3  
PHP4  
F

**int gzseek(int fp, int offset)**

ファイルポインタの位置をoffsetに設定します。成功時に0、それ以外の場合には-1を返します。

PHP3  
PHP4

**int gztell(int fp)**

ファイルポインタの現在位置を返します。

PHP3  
PHP4  
F

**int gzwrite(int fp, string str [,int len])**

文字列をファイルに書き込みます。lenを指定した場合、lenバイトのデータを書き込んだ時点で書き込みを終了します。

PHP3  
PHP4

**string gzcompress(string data [,int level])**

文字列をgzip圧縮したデータを返します。levelに圧縮のレベル(0 ~ 9)を指定可能です。

PHP4

**string gzuncompress(string data [,int len])**

gzip圧縮されたデータを解凍して返します。lenには出力のデータ長を指定可能です。

PHP4

**int readgzfile(string filename)**

ファイルを読み込み、解凍して標準出力に書き込みます。読み込んだ(解凍された)データのバイト数を返します。

PHP3  
PHP4  
F



# Hypertext Preprocessor



## Chapter-4

## PHP 関数

## システム関連

`command` 構文や本節の関数によりシェル上で任意のコマンドを実行可能です。シェルによるコマンドの実行はセキュリティホールになりやすいため注意が必要です。シェルによるコマンドの実行はセーフモード(safe\_mode)オプションを有効にした場合、safe\_mode\_exec\_dir 以下のコマンドに制限されます。

### 4.1 プログラム実行関数

---

```
string escapeshellcmd(string command)
```

---

PHP3  
PHP4

文字列からセキュリティ上問題を生じる可能性がある文字をエスケープします。

```
system(escapeshellcmd($cmd)); // エスケープ処理を行ったあとで$cmdを実行します
```

---

```
string exec(string command [, array &arr][, int &ret])
```

---

PHP3  
PHP4

コマンドを実行し、結果の最後の行を文字列として返します。標準出力への出力は行われません。arr を指定した場合、実行したコマンドの出力が行ごとに保存されます。ret を指定した場合、コマンドのステータスが書き込まれます。

---

```
void passthru(string command [, int &ret])
```

---

PHP3  
PHP4

コマンドを実行し、コマンドの出力をバイナリデータとしてそのままブラウザに出力します。ret を指定した場合、コマンドのステータスが書き込まれます。

---

```
string system(string command [, int &ret])
```

---

PHP3  
PHP4

コマンドを実行し、結果の最後の行を文字列として返します。結果をすべて標準出力に出力します。ret を指定した場合、コマンドのステータスが書き込まれます。

## 4.2 PHP オプションおよび情報

```
int dl(string library)
```

指定したPHP拡張モジュールを動的にロードします。

PHP3  
PHP4  
T F

```
int error_log(string message, int type [, string dest][, string extra])
```

エラーメッセージmessageをtypeで指定した先(表3-9)に送ります。

PHP3  
PHP4  
T F

表 3-9 エラーメッセージの送信先

type	送信先
0	php.iniのerror_logの設定先( syslogまたはファイル )
1	destの指定先に電子メールで送信。ヘッダextraを指定可能
2	指定アドレス( dest )のデバッグ用ポートに送信( PHP4 では未サポート )
3	ログファイルdestに追加

```
error_log("Error!", 0); // サーバログに通知
error_log("Error!", 1, "operator@mydomain.com"); // 管理者にe-mail で通知
error_log("Error!", 2, "loghost"); // デバッグ用コネクションに出力
error_log("Error!", 3, "/var/tmp/my-errors.log"); // ファイルに出力
```

```
int error_reporting([int level])
```

エラー出力レベルを設定し、元の設定値を返します。level には表3-10の値の論理和を設定します( php.iniのerror\_reportingの設定値がデフォルトです )。

表 3-10 エラー出力レベルの設定

type	定数
1	E_ERROR
2	E_WARNING
4	E_PARSE
8	E_NOTICE
16	E_CORE_ERROR
32	E_CORE_WARNING

PHP3  
PHP4

```
bool extention_loaded(string name)
```

拡張機能nameがロードされている場合にtrueを返します。

3.0.10 PHP3  
PHP4  
F

string getenv(string var)

PHP3  
PHP4  
F

環境変数varの値を返します。

```
$ip = getenv ("REMOTE_ADDR"); // ユーザのIPアドレスを得ます
```

string get\_cfg\_var(string var)

PHP3  
PHP4  
F

PHP 設定オプションvarの現在の設定値を返します。Apacheの設定ファイルでphp3\_xxxやphp\_valueなどにより設定した情報は返しません。  
PHPスクリプトの情報を取得するには表3-11の関数を使用します。

```
// '最終更新時刻: March 04 1998 20:43:59.' のように出力を行います
print "最終更新時刻: ".date( "F d Y H:i:s.", getlastmod() );
```

また、PHPスクリプトの情報を設定するには表3-12の関数を使用します。  
なお、PHP4では、表3-13に示す関数により拡張モジュールや関数、インクルードされたファイル名に関する情報を得ることが可能です。

\*8  
エラーの場合はfalseが返されます。

表 3-11 スクリプトの情報を取得する関数

関数名	返り値
int getlastmod()	最終更新時刻( UNIX時間 ) <sup>*8</sup>
int getmyinode()	iノード番号 <sup>*8</sup>
int getmypid()	プロセスID <sup>*8</sup>
int getmyuid()	ユーザID <sup>*8</sup>
string get_current_user()	実行中のPHPスクリプトの所有者
int get_magic_quotes_gpc()	magic_quotes_gpcの値( 0または1 )
int get_magic_quotes_runtime()	magic_quotes_runtimeの値( 0または1 )

表 3-12 スクリプトの情報を設定する関数

関数名	設定する値
void set_time_limit(int sec)	最大実行時間( デフォルトは30秒 )
int filetime(string filename)	magic_quotes_runtimeの値(0または1)

表 3-13 関数・モジュールに関する情報を取得する関数( PHP4のみ )

関数名	返り値
array get_loaded_extensions()	コンパイル・ロード中のモジュール名
array get_extension_funcs(string module)	モジュールで使用可能な関数
array get_included_files()	include_once()でロードされたファイル
array get_required_files()	require_once()でロードされたファイル

**array getrusage([int who])**3.0.7 **PHP3**  
**PHP4**

現在のシステムリソースに関する情報をシステムコールgetrusage()から得て連想配列として返します。whoに1を指定した場合、getusage()にRUSAGE\_CHILDRENが追加されます。

```
$dat = getrusage();
print $dat["ru_nswap"];           # スワップの数
print $dat["ru_majflt"];          # ページフォルトの数
print $dat["ru_utime.tv_sec"];    # 使用するユーザ時間(秒)
print $dat["ru_utime.tv_usec"];  # 使用するユーザ時間(マイクロ秒)
```

**void phpinfo(void)****PHP3**  
**PHP4**

現在のPHPの状態に関する情報(モジュール情報等)を出力します。

**string phpversion(void)****PHP3**  
**PHP4**

現在動作中のPHPパーサのバージョンを表す文字列を返します。

**void putenv(string str)****PHP3**  
**PHP4**

環境変数にstrを追加します。

```
putenv ("UNIQUEID=$uniqueid"); // 環境変数UNIQUEIDに変数$uniqueidの値を追加します
```

## 4.3 日付・時刻関数

**bool checkdate(int month, int day, int year)****PHP3**  
**PHP4**

指定した日付が有効な日付である場合にtrueを返します。dayがその月の日数の範囲内であることなどが確認されます。

**string date(string format [,int timestamp])**

PHP3

PHP4

日付・時間をフォーマットして返します。フォーマットの指定は表3-14に基づきます。フォーマットとして認識されない文字はそのまま表示されます。

```
$a = array("日","月","火","水","木","金","土");
$w = date("w",mktime(0,0,0,1,1,2001)); // 2001年1月1日の曜日を取得
print "21世紀の最初の曜日は" . $a[$w] . "曜日です。";
```

表 3-14 日付フォーマットの指定

記号	内 容
0	php.ini の error_log の設定先( syslog または ファイル )
a	"am" または "pm"
A	"AM" または "AM"
d	日付( 2桁 ) "01" ~ "31"
D	曜日( 3文字 ) "Mon" ~ "Sun"
F	月( 英字 ) "January" ~ "December"
h	12時間単位の時間( 2桁 ) "01" ~ "12"
H	24時間単位の時間( 2桁 ) "00" ~ "23"
g	12時間単位の時間( 先頭に "0" をつけない ) "1" ~ "12"
G	24時間単位の時間( 先頭に "0" をつけない ) "0" ~ "23"
i	分(2桁). "00" ~ "59"
j	日付( 先頭に "0" をつけない ) "1" ~ "31"
l	英字の曜日. "Monday" ~ "Sunday"
L	閏年の場合に "1"、それ以外に "0"
m	月( 2桁 ) "01" ~ "12"
n	月( 先頭に "0" をつけない ) "1" ~ "12"
M	月( 3文字 ) "Jan" ~ "Dec"
s	秒( 2桁 ) "00" ~ "59"
S	英字形式の序数を表す接尾語. "th" や "nd"
t	指定した月の日数. "28" ~ "31"
U	UNIX 時間( 1970/1/1 からの秒数 )
w	曜日( 数値 ) 0( 日曜 ) ~ 6( 土曜 )
Y	年( 4桁 ) "2000" など
y	年( 2桁 )"00" など
z	年間の積算日. "0" ~ "365"
Z	タイムゾーンのオフセット秒数. "-43200" ~ "43200"

**array getdate([int timestamp])**

PHP3

PHP4

日付情報を表3-15に示すような連想配列として取得します。



表 3-15 getdate()関数が返す日付情報

キー	内容	キー	内容
seconds	秒	mon	月( 数字 )
minutes	分	year	年( 数字 )
hours	時間	yday	年間の積算日( 数字 )
mday	月単位の日付	weekday	曜日( 文字列 )"Friday" など
wday	曜日( 数字 )	month	月( 文字列 )"January" など

**array gettimeofday(void)**3.0.7 **PHP3**  
**PHP4**

現在の時刻を連想配列として返します。キーおよび値は、sec( 秒 ) usec( マイクロ秒 ) minuteswsec( グリニッジ基準の分 ) dsttime( 夏時間補正の型 )となります。

**string gmdate(string format [,int timestamp])****PHP3**  
**PHP4**

date()と同じですが、グリニッジ標準時( GMT )でフォーマットします。

**int gmmktime(int hour, int min, int sec, int mon, int mday, int year [,int is\_dst])****PHP3**  
**PHP4**

mkdate()と同じですが、グリニッジ標準時( GMT )を指定します。

**string gmstrftime(string format [,int timestamp])**3.0.12 **PHP3**  
**PHP4**

strftime()と同じですが、グリニッジ標準時( GMT )を返します。

**array localtime([int timestamp [, bool is\_assoc]])****PHP4**

ローカルな時刻を配列として返します。is\_assocに1を指定した場合は表 3-16 に示すような連想配列として返されます。

表 3-16 localtime()関数が返す日付情報

キー	内容	キー	内容
tm_sec	秒	tm_year	年( 数字 )
tm_min	分	tm_wday	週単位の日付
tm_hour	時間	tm_yday	年間の積算日
tm_mday	月単位の日付	tm_isdst	夏時間が有効な場合に"1"
tm_mon	月( 数字 )		

---

```
string microtime(void)
```

---

**PHP3**  
**PHP4**

マイクロ秒単位の現在時間をUNIX時間基準( 1970年1月1日 00:00:00 )で返します。

---

```
int mktime(int hour, int min, int sec, int mon, int mday,
           int year [,int is_dst])
```

---

**PHP3**  
**PHP4**

指定した日付・時間をUNIX時間として返します。夏時間の場合にis\_dstを1に設定します。

---

```
print date("Y/M/d", mktime(0,0,0,1,1,2001)); // 2001/1/1を表示します
```

---

```
string strftime(string format [,int timestamp])
```

---

**PHP3**  
**PHP4**

ロケール設定に基づき、ローカルな時間をフォーマットします。formatには、表3-17に示す変換指定子を使用可能です。

---

```
setlocale ("LC_TIME", "ja_JP");
print strftime("%A in Japan");
```

---

表 3-17 strftime()関数の変換指定子

指定子	返回值	指定子	返回值
%a	曜日の省略形	%r	amまたはpm表記の時間
%A	曜日	%R	時間( 24時間表示 )
%b	月の省略形	%S	秒
%B	月	%t	タブ文字
%c	日付と時間	%T	現在の時間( %H:%M:%S )
%C	世紀	%u	曜日( 数値 )1( 月 ) ~ 7( 日 )
%d	日付	%U	年間の積算週( 日曜から開始 )
%D	%m/%d/%yと等価	%V	ISO 8601:1988による週番号
%e	月単位の日付	%W	年間の積算週( 月曜から開始 )
%h	%bと同じ	%w	曜日( 数値 )0( 日 ) ~ 6( 土 )
%H	時間( 24時間表示 )	%x	時間を除いた日付
%I	時間( 12時間表示 )	%X	日付を除いた時間
%j	年間積算日	%y	年( 2桁 )"00" ~ "99"
%m	月	%Y	年( 4桁 )
%M	分	%Z	タイムゾーンまたはその省略形
%n	改行文字	%%	'%'
%p	"am"または"pm"		

```
int strtotime(string time [,int now])
```

3.0.12 **PHP3**  
**PHP4**

英文形式の日付をUNIX時間に変換します。

```
print strtotime ("1 may 2001"); // 2001/5/1をUNIX時間で出力
```

```
int time(void)
```

**PHP3**  
**PHP4**

現在の時間をUNIX時間で返します。

## 4.4 セッション関数

PHP4からセッション機能がサポートされています。セッション機能によりHTTPリクエスト間でユーザ情報を保持することなどが可能になります。

session\_start()、session\_register()をコールすることによりセッション機能が有効になります(設定ファイルのsession.auto\_startが1の場合は自動的に開始されます)。

セッションデータはセッションIDにより管理されます。セッションIDはクッキーまたはURLパラメータによりクライアント側に保持されます。

セッション機能が有効な場合、PHPはセッションIDがリクエストに含まれているかを確認し、含まれている場合には関連するセッション変数を復元します。

セッションIDが定数SIDとして定義されます。

```
<?php // ユーザ単位のカウンタ
session_register("count");
$count++;
?>
あなたがこのページに来たのは<? print $count; ?>回目です。<P>
続けるには、<A HREF="nextpage.php?<?=SID?>">ここをクリック</A>してください。
```

--enable-trans-sidをつけてPHPをコンパイルした場合、自動的にSIDが追加されるため、<?=SID?>は必要ありません。

```
bool session_decode(string data)
```

**PHP4**

セッションデータをデコードし、セッションに保存する変数を設定します。

---

**bool session\_destroy(void)**

---

**PHP4**

セッションに登録されたデータをすべて破棄し、trueを返します。

---

**string session\_encode(void)**

---

**PHP4**

カレントのセッションの内容をエンコードした文字列を返します。

---

**array session\_get\_cookie\_params(void)**

---

**PHP4**

カレントのセッションクッキーの情報を以下のキーを有する配列として返します。

"lifetime" (クッキーの生存期間)

"path" (情報が保存されている場所のパス)

"domain" (クッキーのドメイン)

---

**string session\_id([string id])**

---

**PHP4**

セッションIDを返します。idを指定した場合、セッションIDを更新します。

---

**bool session\_is\_registered(string varname)**

---

**PHP4**

変数が現在のセッションに登録されている場合にtrueを返します。

---

**string session\_module\_name([string name])**

---

**PHP4**

使用中のセッションモジュールの名前("files"、"mm"、"user")を返します。nameを指定した場合、セッションモジュールを変更します。

---

**string session\_name([string name])**

---

**PHP4**

セッション名を取得します。name(英数字のみ)を指定した場合、セッション名を変更します。

---

**bool session\_register(mixed name [, mixed ...])**

---

**PHP4****T**

セッションに変数を登録します。引数の数は可変です。

---

```
string session_save_path([string path])
```

---

PHP4

セッションデータを保存するパスを返します。pathを指定した場合は設定を更新します。

---

```
void session_set_cookie_params(int lifetime [, string path  
                                [, string domain]])
```

---

PHP4

設定ファイルのセッションパラメータを変更します。この関数の効果が持続するのは、スクリプトの実行終了までです。

---

```
void session_set_save_handler(string open, string close, string read,  
                               string write, string destroy, string gc)
```

---

PHP4

セッション機能のハンドラとしてユーザ関数を設定します。open、close、read、write、destroy、gcの処理を行う関数を定義し、関数名を指定します。RDBMS等によりセッションデータを保持する場合に使用します。

---

```
<?php
function open ( $save_path, $session_name) {
    print "open ($save_path, $session_name)%n";
    return true;
}
function close () {
    print "close%n"; return true;
}
function read ($key) {
    print "read ($key)%n";
    return "foo|i:1;"; // 読み込んだ値として変数fooの値を返す
}
function write ($key, $val) {
    return true;
}
function destroy ($key) {
    return true;
}
function gc ($maxlifetime) {
    return true;
}
// セッションハンドラを登録
session_set_save_handler("open", "close", "read", "write", "destroy", "gc");
session_start(); // セッションを開始
$foo++;
?>
```

---

**bool session\_start(void)**

---

**PHP4**

セッションデータを作成し、trueを返します。GETまたはクッキーにより指定されたセッションIDに基づき現在のセッションを復帰します。

---

**bool session\_unregister(string name)**

---

**PHP4****T**

セッションからセッション変数を削除します。

---

**void session\_unset(void)**

---

**PHP4**

現在登録されているすべてのセッション変数を解放します。



## 5.1 HTTP 関連の関数

```
int header(string string)
```

HTTPヘッダを送信します。ほかのデータを出力する前にコールする必要があります。

PHP3  
PHP4

```
header("Location: http://www.php.net"); // ブラウザをリダイレクト
header("Pragma: no-cache");             // HTTP/1.0 でキャッシュ無効を指定
```

```
int setcookie(string name [, string value][, int expire]
              [, string path][, string domain][, int secure])
```

クッキーnameの値を設定します。クッキーはHTTPヘッダにより送信され、セッション管理などに使用されます。nameのみを指定した場合、その名前のクッキーがクライアントから削除されます。name以外の引数はオプションで表3-18のように設定します。

PHP3  
PHP4

表 3-18 setcookie()関数のオプション<sup>\*9</sup>

引数	内容
name	フィールド名
value	クッキーの設定値。自動的にURLエンコードされる
expire	有効期間をUNIX時間(整数)で指定する
path	クッキーを有効とするパス
domain	クッキーを有効とするドメイン
secure	HTTPSコネクションによりクッキーを送信する場合に1とする

<sup>\*9</sup>

オプションの引数は" "または0により指定を省略できます。引数 expire に0を指定または指定を省略した場合、セッションクッキー(有効期限はブラウザ終了まで)となります。

```
// 適用するパスおよびドメインを指定、有効期間は1時間とします
setcookie("test", "taro", time()+3600, "/~taro/", ".php.gr.jp", 1);
```

クッキーの値はPOST、GET メソッドで取得された変数と同様に自動的にPHP変数としてアクセス可能です。具体的には、以下のようにします。

```
print $test; // クッキー"test"の内容を表示
```

## 5.2 URL 関数

---

**array** parse\_url(string url)**PHP3**  
**PHP4**

URLを要素(scheme、host、port、user、pass、path、query、fragment)に分割し、連想配列として返します。

---

**string** urldecode(string str)**PHP3**  
**PHP4**

URLエンコードされた文字列をデコードし、文字列として返します。

---

**string** urlencode(string str)**PHP3**  
**PHP4**

文字列の - 、 \_ 、 . 以外の非英数文字をURLエンコードし、文字列として返します。空白は + 記号にエンコードされます。ユーザ入力データをCGIに引き渡す際に使用します。

```
// ユーザ入力をURLエンコードし、CGIの引数として付加します
print "<A HREF=¥"mycgi?foo=" . urlencode ($userinput) . "¥">";
```

---

**string** base64\_encode(string data)**PHP3**  
**PHP4**

文字列をBASE64でエンコードして返します。

---

**string** base64\_decode(string encoded\_data)**PHP3**  
**PHP4**

BASE64エンコードされた文字列をデコードし、文字列として返します。結果はバイナリになることもあります。



```
bool mail(string to, string subject, string message
          [, string headers][, string lang])
```

PHP3  
PHP4

メッセージを宛先to(複数アドレス指定可能)にメールします。

```
// 指定アドレスにメールを送付します
mail ("hirokawa@php.net", "Hello!", "PHP is great!");
```

headersによりヘッダ文字列を追加指定できます。

```
// 追加ヘッダを指定してメールを送信します
mail ("hirokawa@php.net", "Hello!", $message,
      "From: webmaster@$SERVER_NAME\r\nReply-To: webmaster@$SERVER_NAME\r\nX-
      Mailer: PHP/" . phpversion());
```

国際化版PHPで日本語を含むメールを送信する場合、適切な文字コード(日本語の場合はISO-2022-JP)への変換が行われ、適切なヘッダーが生成されます(PHP4の場合は、`jsr_send_mail()`を使用する必要があります)

## 5.3 ネットワーク関数

```
int fsockopen(string host, int port [, int &errno]
              [, string &errstr][, int timeout])
```

PHP3  
PHP4  
F

ホスト名とポートを指定してインターネットドメイン(AF\_INET)またはUNIXドメイン(AF\_UNIX)のソケットをオープンし、ポインタを返します。引数は表3-19のように設定します。

表 3-19 fsockopen()関数の設定オプション

引数	内容
name	フィールド名
host	ホスト名(UNIXドメインの場合はソケットのパス) <sup>*10</sup>
port	ポート番号(UNIXドメインの場合は0)
errno	システムコールconnect()のエラー番号
errstr	システムコールconnect()のエラーメッセージ
timeout	処理のタイムアウトまでの時間(秒) <sup>*11</sup>

<sup>\*10</sup>

データグラム型の場合はudp://を先頭に付加(デフォルトはストリーム型)します。

<sup>\*11</sup>

PHP3では整数秒、PHP4では小数点以下(マイクロ秒)も指定可能です。

```
int pfsockopen(string host, int port [, int errno]
               [, string errstr][, int timeout])
```

PHP3 3.0.7  
PHP4  
F

持続的接続としてソケットをオープンします。同一のホスト、ポートですでにソケットがオープンされている場合はそのソケットのポインタを返します。また、スクリプト実行終了時にも持続型のソケットはクローズされません。その他の動作はfsockopen()と同じです。

```
int set_socket_blocking(int sid , int mode)
```

PHP3  
PHP4

ソケットのブロックモードを設定します。modeにはfalse(非ブロックモード)またはtrue(ブロックモード)を指定します。

```
string gethostbyaddr(string ip)
```

PHP3  
PHP4

IPアドレスipを有するホスト名を返します。エラー時にはipを返します。

```
string gethostbyname(string host)
```

PHP3  
PHP4

ホストのIPアドレスを返します。

```
array gethostbyname_l(string host)
```

PHP3  
PHP4

ホストのIPアドレスのリストを配列として返します。

```
int checkdnsrr(string host [, string type])
```

PHP3  
PHP4

ホストのDNSレコードを検索し、見つかった場合にtrue、それ以外の場合にfalseを返します。typeにはレコードの種類(A、MX、NS、SOA、PTR、CNAME、ANY)を指定します(デフォルトはMX)。

```
int getmxrr(string host, array &mxhosts [, array &weight])
```

PHP3  
PHP4

ホストをDNSのMXレコードから検索し、見つかった場合にtrue、それ以外の場合にfalseを返します。検索結果が配列mxhostsに、重み情報が配列weightに書き込まれます。

---

**int openlog(string ident, int option, int facility)**

---

システムログへの接続をオープンします。文字列identが各メッセージに追加されます。通常は自動的にオープンされるため不要です。

**PHP3**  
**PHP4**

---

**int syslog(int priority, string message)**

---

システムログにメッセージを出力します。メッセージの中で%mはerrnoの値に置換されます。

**PHP3**  
**PHP4**

---

**int closelog(void)**

---

システムログへの接続を閉じます。通常、closelog()をコールする必要はありません。

**PHP3**  
**PHP4**

## 5.4 出力バッファリング関連の関数

---

**string ob\_get\_contents()**

---

出力用バッファの内容を返します。

**PHP4**  
**F**

---

**string ob\_end\_flush()**

---

出力用バッファをフラッシュ(出力)し、出力のバッファリングをオフにします。

**PHP4**

---

**string ob\_end\_clean()**

---

出力用バッファを消去し、出力のバッファリングをオフにします。

**PHP4**

---

**void ob\_implicit\_flush([int flag])**

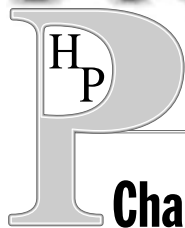
---

バッファの自動フラッシュをOn(デフォルト)またはOffに設定します。自動フラッシュをOnにすると出力のバッファリングは無効となり、現在の出力バッファの内容が出力されます。

**PHP4**



# Hypertext Preprocessor



## Chapter - 6

## PHP 関数

## 計算関連

### 6.1 数学関数

PHPではC言語等でサポートされる数学関数を使用可能です。サポートされる基本的な関数を表3-20に示します。

また、表3-21に示す任意精度関数(BC関数)も使用可能です。

このほか、乱数、基底の変換などの関数がサポートされています。

表 3-20 基本的な数学関数

関数名	返り値
float acos(float arg)	アークコサイン(単位: ラジアン)
float asin(float arg)	アークサイン(単位: ラジアン)
float atan(float arg)	アークタンジェント(単位: ラジアン)
float atan2(float y, float x)	y/xの4象限アークタンジェント([-, ]) )
mixed abs(mixed arg)	絶対値 <sup>*12</sup>
int ceil(float arg)	arg<nとなる最小の整数
float cos(float arg)	コサイン
double deg2rad(double arg)	degをradに変換した値
float exp(float arg)	eのarg乗
int floor(float arg)	arg>=nとなる最大の整数
float log(float arg)	自然対数
float log10(float arg)	常用対数
float pow(float base, float exp)	baseのexp乗
double pi(void)	円周率の近似値
double rad2deg(double arg)	radをdegに変換した値
float sin(float arg)	サイン
float sqrt(float arg)	平方根
float tan(float arg)	タンジェント

\*12

入力がfloatの場合はfloat、それ以外はint。

表 3-2 1 任意精度関数 \*13

関数名	返り値
string bcadd(string a,string b [,int scl])	a+b
string bccomp(string a,string b [,int scl])	a>b : 1,a=b : 0,a<b : -1
string bcdiv(string a,string b [,int scl])	a/b
string bcmmod(string a,string b)	aをbで割った余り
string bcmul(string a,string b [,int scl])	a*b
string bcpow(string a,string b [,int scl])	aのb乗
string bcscale(int scl)	精度のデフォルト値をscl桁に設定
string bcsqrt(string a [,int scl])	平方根
string bcsub(string a,string b [,int scl])	a-b

\*13

sclにより結果の小数点以下の精度の桁数を指定可能です。

### string base\_convert(string num, int from, int to)

数値numの基底をfromからtoに変換し、文字列として返します。fromおよびtoはともに2～36の範囲内でなければなりません。10以上の基底を表す数は、文字a～zで表されます。基底の変換には、表3-22に示す関数も使用可能です。

**PHP3**  
**PHP4**

```
$bin = base_convert($hex, 16, 2); // 16進数$hexを2進数に変換します
```

表 3-2 2 基底変換用関数

関数名	動作
int bindec(string num)	2進数を10進数に変換
string decbin(int num)	10進数を2進数に変換
string dechex(int num)	10進数を16進数に変換
string decoct(int num)	10進数を8進数に変換
int hexdec(string num)	16進数を10進数に変換
int octdec(string num)	8進数を10進数に変換
string bin2hex(string num)	バイナリデータを16進数に変換

### mixed max(mixed arg1, mixed arg2, ...,mixed argn)

引数(スカラーまたは配列)の最大値を返します。

**PHP3**  
**PHP4**

### mixed min(mixed arg1, mixed arg2, ...,mixed argn)

引数(スカラーまたは配列)の最小値を返します。

**PHP3**  
**PHP4**

---

**string number\_format(float num [, int dec][, string point][, string sep])**

---

**PHP3****PHP4**

数値を1000の位ごとにフォーマットした文字列を返します。デフォルトでは小数点以下は無視されます。

decにより小数点以下の表示桁数、pointにより小数点文字(デフォルトは".")、sepにより1000単位の区切り文字(デフォルトは",")を指定可能です。

```
$num = 12345.678;  
print number_format($num), "<BR>¥n"; // "12,346" が出力されます  
print number_format($num, 2), "<BR>¥n"; // "12,345.68" が出力されます  
print number_format($num, 2, "#", "-"), "<BR>¥n"; // "12-345#68" が出力されます
```

---

**double round(double val)**

---

**PHP3****PHP4**

valを整数値に丸めた数値を返します。丸め誤差によりかならずしも四捨五入とはならないことに注意してください。

```
$foo = round(3.4); // $foo = 3.0となります  
$foo = round(3.5); // $foo = 4.0となります  
$foo = round(3.6); // $foo = 4.0となります
```

## 6.2 乱数関連

PHPではシステムコールrand()を使用する乱数(表3-23)により高品質なMT乱数(表3-24)をサポートしています。

Mersenne Twister(MT)法による乱数は、通常のライブラリ関数に実装されている乱数発生器よりも高品質で高速です。

Mersenne Twisterのホームページは以下の場所にあります。

<http://www.math.keio.ac.jp/~matumoto/emt.html>

表3-23 乱数をサポートする関数

関数名	動作
int getrandmax(void)	乱数の最大値を返す
int rand([int min],[int max])	[min,max]の範囲の乱数を返す <sup>*14</sup>
void srand(int seed)	乱数をシードseedで初期化する

<sup>\*14</sup>

デフォルトは[0,RAND\_MAX]。

```
// 現在の時間をマイクロ秒で表した値をシードとして初期化します
srand((double)microtime()*1000000);
$randval = rand(); // 乱数を求めます
```

表3-24 MT乱数をサポートする関数

関数名	動作
int mt_getrandmax(void)	MT乱数の最大値を返す
int mt_rand([int min],[int max])	[min,max]の範囲のMT乱数を返す <sup>*15</sup>
void mt_srand(int seed)	MT乱数をシードseedで初期化する

<sup>\*15</sup>

デフォルトは[0,RAND\_MAX]。

```
// マイクロ秒単位の現在の時間をシードとして初期化します
mt_srand((double)microtime()*1000000);
$randval = mt_rand(); // 乱数を発生します
```



# Hypertext Preprocessor



## Chapter-7

## PHP 関数

### 国際化(日本語)関連

PHP で日本語を含むマルチバイト文字列を処理した場合、文字化けなどの問題が起こることがあります。有志の手で国際化(日本語)化が進められており、日本語文字コード変換や内部文字コードの設定など、日本語を処理するための機能が実現されています。

国際化版 PHP3 は以下の URL にて配布されています。

PHP 国際化プロジェクト：<http://php.jpnet.com/>

PHP4 に関しても国際化の作業が進行中です。現在、塚田氏作成の PHP4 で日本語の文字コード変換等を行う拡張モジュール(jstring、mbregex、iconv、namazu)が下記の URL にて提供されています。

塚田氏のftp サイト：<ftp://night.fminn.nagano.nagano.jp/php4/>

入出力データの文字コードを自動変換する機能は PHP4 では現在未サポートです。

## 7.1 コード変換関数

---

```
string i18n_convert(string str, string code [, string old])
```

---

**PHP3**  
**PHP4**

文字列をコードcodeに変換し、文字列として返します。oldにより変換前のコード(autoの場合は自動認識)を指定可能です。デフォルトは現在の内部コードです。

code には、EUC-JP、SJIS、JIS、UTF-8、pass(無変換)を指定します。

---

```
string i18n_discover_encoding(string str [,string lang])
```

---

**PHP3**  
**PHP4**

文字列のコードを返します。判別できない場合はautoを返します。



```
string i18n_http_input([string code])
```

HTTP入力のコード( PHP4 ではデフォルトのコード )を設定します。引数を省略した場合、現在のコードを返します。

PHP3  
PHP4

```
string i18n_http_output([string code])
```

HTTP出力のコード( PHP4 ではデフォルトのコード )を設定します。引数を省略した場合、現在のコードを返します。

PHP3  
PHP4

```
string i18n_internal_encoding()
```

内部コード( PHP4 ではデフォルトのコード )を返します。

PHP3  
PHP4

```
string i18n_ja_jp_hantozen(string str [,string option][,string code])
```

文字列strを全角から半角、または半角から全角に変換します。optionで表3-25に示す変換モードを指定可能です( デフォルトは KV )。

PHP3  
PHP4

```
print i18n_ja_jp_hantozen("ポルトガル","KV"); // 「ポルトガル」が出力されます
```

表 3-25 変換モード

設定値	動作モード
r	英字を全角から半角に変換
R	英字を半角から全角に変換
n	数字を全角から半角に変換
N	数字を半角から全角に変換
a	英数字(21H ~ 7EH)を全角から半角に変換
A	英数字(21H ~ 7EH)を半角から全角に変換
k	全角カタカナを半角カナに変換
K	半角カナを全角カタカナに変換
h	全角ひらがなを半角カナに変換
H	半角カナを全角ひらがなに変換
c	全角カタカナを全角ひらがなに変換
C	全角ひらがなを全角カタカナに変換
V	濁音記号を前の文字と組み合わせて1文字に変換( K,Hと組み合わせて使用 )

---

```
string iconv_convert(string str, string to, string from)
```

---

**PHP4**

iconv 関数により、文字コードをfrom からtoへ変換します。from/toにはUTF-8、EUC-JPなどを指定します。

---

```
int jstr_default_encoding(string code)
```

---

**PHP4**

文字エンコードのデフォルト値を設定します。設定ファイル(php.ini)のjstring.default\_encodingでも指定可能です。

---

```
bool jstr_send_mail(string to, string subject, string message  
[, string headers])
```

---

**PHP4****F**

メッセージをメールします。引数の設定方法はmail()と同じです。MIMEは日本語でエンコードされ、適切なヘッダを生成します。

## 7.2 マルチバイト関数

---

```
int mbstrlen(string str [,string code])
```

---

**PHP3****PHP4**

文字列の文字数を返します。code(デフォルトは内部コード)により文字数を換算します。

```
print mbstrlen("あいうえお"); // 5が出力されます
```

---

```
string mbstrcut(string str, int start, int len [, int code])
```

---

**PHP3****PHP4**

文字列のstart番目(先頭文字が0)の文字からlenバイト分の文字列を返します。startがマルチバイト文字の先頭バイトでない場合でも、その文字の先頭バイトから文字単位で切り出します。code(デフォルトは内部コード)により文字数を換算します。

```
print mbstrcut("あいうえお",2,3); // "い" が出力されます
```

```
int mbstrpos(string str, string key [, int offset][, int code])
```

**PHP3**  
**PHP4**

文字列の中でキーが最初に現れた位置を返します。offsetにより検索開始位置を指定可能です。code(デフォルトは内部コード)により文字数を換算します。

```
print mbstrpos("あいうえお","うえ"); // 2が出力されます。
```

```
string mbsubstr(string str, int start, int len [, int code])
```

**PHP3**  
**PHP4**

文字列の位置start(先頭文字が0)からlen文字分の文字列を返します。code(デフォルトは内部コード)により文字数を換算します。

```
print mbsubstr("あいうえお",3,2); // "えお" が出力されます。
```

## 7.3 全文検索( **Namazu** )関数

Namazu拡張モジュールにより、Namazuによる全文検索を実行可能です。Namazu 2.0以降および塚田氏によるNamazu用拡張モジュールが必要です。

Namazuのページ: <http://www.namazu.org/>

```
<?php // Namazuによる全文検索の例
$query = "PostgreSQL"; // クエリ文字列
$index = "/usr/share/namazu/index/phpdoc"; // Namazuインデックス
$id = nmz_open($index) or die("Can't open namazu index.");
$hlist = nmz_search($id,$query); // 検索を実行
if($hlist != false){
    $count = nmz_num_hits($hlist);
    print "<strong> 合計 $count 回ヒットしました。</strong><p>¥n";
    $uri = nmz_fetch_field($hlist,"uri");
    $summary = nmz_fetch_field($hlist,"summary");
    $date = nmz_fetch_field($hlist,"date");
    $subject = nmz_fetch_field($hlist,"subject");
    print "<dl>¥n";
    for ($k=0;$k<$count;$k++){ // フィールドを表示
        print "<dt>$k. <a href=¥\"file://$uri[$k]¥\">$uri[$k]</a>¥n";
        print "<dd>日付: $date[$k]¥n";
        print "<dd>題名: $subject[$k]¥n";
        print "<dd>概要: $summary[$k]¥n";
    }
}
?>
```

---

```
bool nmz_set_sortorder(string order)
```

---

**PHP4**

ソートの順番を指定します。order には、ascending(昇順)またはdescending(降順)を指定します。設定ファイルのnamazu.sortorderにてデフォルト値を設定可能です。

---

```
bool nmz_set_sortmethod(string method)
```

---

**PHP4**

ソートの方法を指定します。method には、score、date、field:{フィールド名} を指定します。設定ファイル namazu.sortmethod でデフォルト値を設定可能です。

---

```
int nmz_open(mixed index)
```

---

**PHP4****F**

インデックス(配列も指定可)をオープンし、IDを返します。

```
$idx = array("/var/share/namazu/idx1", "/var/share/namazu/idx2");
$nmzid = nmz_open($idx);
```

---

```
bool nmz_close(int nmzid)
```

---

**PHP4****T F**

インデックスを閉じます。

---

```
int nmz_search(int nmzid, string query)
```

---

**PHP4****F**

クエリを行い、検索結果リストへのハンドルを返します。

---

```
bool nmz_free_result(int hlist)
```

---

**PHP4****T F**

検索結果を解放します。

---

```
int nmz_num_hits(int hlist)
```

---

**PHP4****F**

検索結果の文書数を取得します。

---

```
int nmz_result_score(int hlist, int i)
```

---

**PHP4****F**

i 番目の文書の点数(スコア)を取得します。

```
int nmz_result_date(int hlist, int i)
```

i 番目の文書の日付をUNIX時間で取得します。

**PHP4**  
**F**

```
$time = nmz_result_date($hlist,0); // 先頭のレコードの日付  
print date('Y/m/d',$time); // 日付を表示
```

```
string nmz_result_field(int hlist, int i, string field)
```

i 番目の文書の指定したフィールドを取得します。

**PHP4**  
**F**

```
array nmz_fetch_score(int hlist [, int limit])
```

各文書の点数を配列として返します。limit により取得数の上限を設定可能です。

**PHP4**  
**F**

```
array nmz_fetch_date(int hlist [, int limit])
```

各文書の日付を配列として返します。limit により取得数の上限を設定可能です。

**PHP4**  
**F**

```
array nmz_fetch_field(int hlist, string field [, int limit])
```

各文書の指定フィールドを配列として返します。limit により取得数の上限を設定可能です。

**PHP4**  
**F**

```
array nmz_get_querytoken(int hlist)
```

クエリ中の単語を配列として取得します。

**PHP4**  
**F**

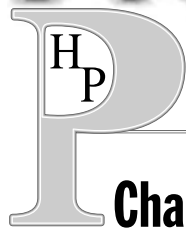
```
void nmz_info(void)
```

ソートの方法および順番の設定値に関する情報を出力します。

**PHP4**



# Hypertext Preprocessor



## Chapter-8

## PHP 関数

## ドキュメント関連

### 8.1 PDF 処理関数

PHPでは、標準電子文書フォーマットであるPDF( Portable Document Format )形式のドキュメントを動的に作成することが可能です。

この機能を使用するには、PDFlibまたはClibPDFが必要です。

PDFlib( <http://www.pdflib.com/pdflib/> ) : 日本語の使用にはバージョン3.0以上が必要

ClibPDF( <http://www.fastio.com/> ) : 日本語の使用にはバージョン2.0以上が必要

二つのAPIはほぼ同様の機能を有するため、ここでは日本語処理の点で実績のあるPDFlibをとり上げます。

```
<?php // PHP4.0.2+PDFLIB-3.0.1によるPDF出力の例
Header("Content-type: application/pdf"); // MIMEヘッダ送信
$pdf = pdf_open();
pdf_set_info_author($pdf, "Suzuki Taro");
pdf_set_info_title($pdf, "Test for PHP wrapper of PDFlib");
pdf_begin_page($pdf, 842, 595); // A4 size
pdf_add_outline($pdf, "Page 1");
pdf_set_text_rendering($pdf, 2);
pdf_set_font($pdf, "HeiseiMin-W3", 20, "EUC-H"); // 平成明朝を使用する
pdf_show_xy($pdf, "日本語" , 200, 742);
pdf_end_page($pdf);
pdf_close($pdf);
?>
```

```
int pdf_add_annotation(int pdf, double x1, double y1, double x2,
double y2, string title, string content)
```

PHP3 3.0.12  
PHP4  
T F

左下隅を(x1,y1)、右上隅を(x2,y2)とする注記を追加します。

<code>int pdf_add_outline(int pdf, string text)</code>	3.0.6	PHP3 PHP4 F
カレントのページにブックマークを追加し、ブックマークIDを返します。		
<code>int pdf_arc(int pdf, double x, double y, double r, double start, double end)</code>	3.0.6	PHP3 PHP4 T F
中心を(x,y)、半径をr、開始角をstart、終了角をendとした円弧を描画します。		
<code>int pdf_begin_page(int pdf, double h, double w)</code>	3.0.7	PHP3 PHP4 T F
高さh、幅wで新規ページを開始します。		
<code>int pdf_circle(int pdf, double x, double y, double r)</code>	3.0.6	PHP3 PHP4 T F
中心を(x,y)、半径をrとした円を描画します。		
<code>int pdf_clip(int pdf)</code>	3.0.6	PHP3 PHP4 T F
カレントのパスにより切り取ります。		
<code>int pdf_close(int pdf)</code>	3.0.6	PHP3 PHP4 T F
PDF ドキュメントを閉じます。		
<code>int pdf_close_image(int pdf)</code>	3.0.7	PHP3 PHP4 T F
PDF_open_xxx()関数でオープンされたイメージを閉じます。		
<code>int pdf_closepath(int pdf)</code>	3.0.6	PHP3 PHP4 T F
カレントのパスを閉じます。		
<code>int pdf_closepath_fill_stroke(int pdf)</code>	3.0.7	PHP3 PHP4 T F
カレントのパスの内部を閉じ、塗りつぶし、パスを描画します。		

---

```
int pdf_continue_text (int pdf, string text)
```

---

**PHP3** 3.0.6

**PHP4**

**T F**

次の行にテキストを出力します。

---

```
int pdf_curveto (int pdf, double x1, double y1, double x2,
                 double y2, double x3, double y3)
```

---

**PHP3** 3.0.6

**PHP4**

**T F**

ベジエー曲線をカレントの点から点(x3,y3) まで (x1,y1) および (x2,y2) を制御点として描画します。

---

```
int pdf_end_page(int pdf)
```

---

**PHP3** 3.0.7

**PHP4**

**T F**

ページを終了します。

---

```
int pdf_fill(int pdf)
```

---

**PHP3** 3.0.7

**PHP4**

**T F**

カレントのパスの内部を塗りつぶします。

---

```
int pdf_fill_stroke(int pdf)
```

---

**PHP3** 3.0.7

**PHP4**

**T F**

カレントのパスの内部を塗りつぶし、パスを描画します。

---

```
int pdf_endpath(int pdf)
```

---

**PHP3** 3.0.7

**PHP4**

**T F**

カレントのパスを終了します。

---

```
string pdf_get_parameter (int pdf, string name, double modifier)
```

---

**PHP3** 3.0.7

**PHP4**

**F**

パラメータの値を文字列として取得します。modifier( 不要の場合は0 )によりパラメータを修飾します。

---

```
double pdf_get_value (int pdf, string name, double modifier)
```

---

**PHP3** 3.0.7

**PHP4**

**F**

パラメータの値を数値として取得します。modifier( 不要の場合は0 )によりパラメータを修飾します。

---



<code>int pdf_lineto(int fd, double x, double y)</code>	3.0.6	PHP3 PHP4 T F
カレントの点から座標(x, y)の点まで線を描画します。		
<code>int pdf_moveto(int fd, double x, double y)</code>	3.0.6	PHP3 PHP4 T F
カレントの位置を(x,y)に設定します。		
<code>int pdf_open([int fd])</code>	3.0.6	PHP3 PHP4 F
PDF ドキュメントを新規にオープンし、ID を返します。fd には fopen() から返されたファイル記述子を指定します。省略時はメモリ上に作成されます。		
<code>int pdf_open_gif(int pdf, string filename)</code>	3.0.7	PHP3 PHP4 F
GIF イメージファイルをオープンし、PDF イメージID を返します。		
<code>int pdf_open_jpeg(int pdf, string filename)</code>	3.0.7	PHP3 PHP4 F
JPEG イメージファイルをオープンし、PDF イメージID を返します。		
<code>int pdf_open_png(int pdf, string filename)</code>	3.0.7	PHP3 PHP4 F
PNG イメージファイルをオープンし、PDF イメージID を返します。		
<code>int pdf_open_memory_image(int pdf, int image)</code>	3.0.7	PHP3 PHP4 F
イメージ関数(第3章を参照)によるイメージをPDF ドキュメントで利用できるようにし、PDF イメージID を返します。		

```
<?php
$im = imagecreate (100, 100); // イメージを作成
$blue = imagecolorallocate ($im, 0, 0, 255); // 青色を定義
imagerectangle ($im,0,0,99,99,$blue) ; // 青色の矩形を描画
$vim = pdf_open_memory_image ($pdf, $im); // PDF イメージとして展開
imagedestroy ($im); // イメージを破棄
pdf_place_image ($pdf, $vim, 100, 100, 1); // PDF イメージをドキュメントに配置
pdf_close_image ($pdf, $vim); // PDF イメージをクローズ
?>
```

---

```
int pdf_place_image(int pdf, int image, double x, double y, double scale)
```

---

**PHP3** 3.0.7  
**PHP4**  
**T F**

イメージを座標(x,y)に置きます。scaleによりサイズ(1:等倍)を調整することが可能です。

---

```
int pdf_rect(int fd, double x, double y, double r, double w, double h)
```

---

**PHP3** 3.0.7  
**PHP4**  
**T F**

左下隅を(x,y)、幅をw、高さをhとした矩形を描画します。

---

```
int pdf_restore (int pdf)
```

---

**PHP3** 3.0.7  
**PHP4**  
**T F**

pdf\_save()により保存された環境を回復します。

---

```
int pdf_rotate (int pdf, double angle)
```

---

**PHP3** 3.0.7  
**PHP4**  
**T F**

指定した角度だけ回転します。

---

```
int pdf_save (int pdf)
```

---

**PHP3** 3.0.7  
**PHP4**  
**T F**

カレントの環境を保存します。回転、変換を行う場合に使用します。

---

```
int pdf_scale (int pdf, double sx, double sy)
```

---

**PHP3** 3.0.7  
**PHP4**  
**T F**

XおよびY方向のスケールにsxおよびsyを設定します。

---

```
int pdf_set_font (int pdf, string font, double size,  
                  string encoding, int embed)
```

---

**PHP3** 3.0.7  
**PHP4**  
**T F**

フォントの種類(font)およびサイズ(size)を設定します。encodingにはエンコードを指定します。embedによりフォントの埋め込みを行うかどうかを指定します(日本語を含むCJKフォントでは0とします)。

---

```
bool pdf_set_info(int pdf, string field, string value)
```

---

**PHP3** 3.0.7  
**PHP4**  
**T F**

PDFドキュメントの情報フィールドfieldをvalueに設定します。fieldには Subject、Title、Creator、Author、Keywordsを指定可能です。

---

```
pdf_set_info($pdfdoc, "Author", "Taro Tanaka"); // 著者を設定します
```

各フィールドの値を指定する際には、表 3-26 に示す関数を使用することも可能です。

表 3-26 PDF 情報フィールドを設定する関数

関数名	設定フィールド
bool pdf_set_info_creator(int pdf, string var) <b>T F</b>	Creator
bool pdf_set_info_title(int pdf, string var) <b>T F</b>	Title
bool pdf_set_info_subject(int pdf, string var) <b>T F</b>	Subject
bool pdf_set_info_author(int pdf, string var) <b>T F</b>	Author
bool pdf_set_info_keywords(int pdf, string var) <b>T F</b>	Keywords

```
int pdf_set_parameter (int pdf, string name, string value)
```

パラメータnameの値を設定します。

3.0.7 **PHP3**  
**PHP4**  
**T F**

表 3-27 PDF ドキュメントのパラメータを設定する関数 (**T F**)

関数名	設定値
int pdf_set_border_color(int pdf, double r, double g, double b)	境界色
int pdf_set_border_dash(int pdf, double b, double w)	ダッシュのスタイル
int pdf_set_border_style(int pdf, double style, double w)	境界のスタイル
int pdf_set_char_spacing(int pdf, double space)	文字間の空白
int pdf_set_duration(int pdf, double duration)	ページ間の持続時間(秒)
int pdf_set_horiz_scaling(int pdf, double scl)	水平スケール
int pdf_set_text_rendering(int pdf, int mode)	テキストの描画法
int pdf_set_text_rise(int pdf, double pt)	テキストの大きさ
int pdf_set_text_pos(int pdf, double x, double y)	テキストのカレント位置
int pdf_set_transition(int pdf, int trans)	ページ間の移動 <sup>*16</sup>
int pdf_set_word_spacing (int pdf, double space)	単語間の空白
int pdf_setdash(int pdf, double w, double b)	破線パターン
int pdf_setflat(int pdf, double value)	平面度を0から100の値
int pdf_setgray(int pdf, double gray)	グレイ値の塗りつぶしおよび描画色
int pdf_setgray_fill(int pdf, double gray)	グレイ値の塗りつぶし色
int pdf_setgray_stroke(int pdf, double gray)	グレイ値の描画色
int pdf_setlinejoin(int pdf, int value)	linejoinパラメータ
int pdf_setlinecap(int pdf, int value)	linecapパラメータ
int pdf_setlinewidth (int pdf, double width)	行幅
int pdf_setmiterlimit(int pdf, double value)	miterリミット
int pdf_setrgbcolor(int pdf, double r, double g, double b)	RGB色の塗りつぶしおよび描画色
int pdf_setrgbcolor_fill(int pdf, double r, double g, double b)	RGB色の塗りつぶし色
int pdf_setrgbcolor_stroke(int pdf, double r, double g, double b)	RGB色の描画色

\*16

trans には、

- 0:何もしない
- 1:2行ずつスクロール
- 2:複数行ずつスクロール
- 3:ページのまわりに箱を描く
- 4:1行ずつスクロール
- 5:古いページを消去する
- 6:画面のひとつの隅から他の隅に移動するように消す
- 7:古いページを新しいページで単純に置換(デフォルト)

などがあります。

---

```
int pdf_show(int pdf, string str)
```

---

**PHP3** 3.0.7  
**PHP4**  
**T F**

カレントの位置に文字列を出力します。

---

```
int pdf_show_boxed(int pdf, string str, double x, double y,
                  double w, double h, string mode)
```

---

**PHP3** 3.0.7  
**PHP4**  
**T F**

左下隅の座標を (x,y)、幅 w、高さ h のボックス内に文字列を出力します。mode には、left、right、center、justify、fulljustify を指定可能です。

---

```
int pdf_show_xy(int pdf, string str, double x, double y)
```

---

**PHP3** 3.0.7  
**PHP4**  
**T F**

座標を (x,y) に文字列を出力します。

---

```
double pdf_stringwidth (int pdf, string text)
```

---

**PHP3** 3.0.7  
**PHP4**

カレントフォントのテキストの幅を返します。

---

```
int pdf_translate (int pdf, double x, double y)
```

---

**PHP3** 3.0.7  
**PHP4**  
**T F**

座標系の (x,y) に設定します。

---

## 8.2 XML 関数

XML (eXtensible Markup Language) は、次世代のデータ記述言語として注目されており、さまざまな用途に使用されるようになりつつあります。

PHP では、XML パーサ API として標準的な以下のものがサポートされています。

- SAX (Simple API for XML) パーサ

各要素についてコールバック関数を定義し、シーケンシャルに XML データを処理します。シンプルで処理が軽く、Web での使用に向いています。PHP では、SAX パーサとして expat (<http://www.jclark.com/xml/expat.html>) がサポートされています。

- DOM (Document Object Model) パーサ

XML データのツリー構造を解釈し、任意の要素にランダムアクセスが可能です。データ構

造の一部を利用したい場合に使用します。PHP では、GNOME-XML( libxml2 )がPHP4 でサポートされています。

Web アプリケーションでは、通常処理負荷が軽い SAX パーサが使用されます。XML のタグを HTML のタグに置換する例を以下に示します。

```
// 開始タグ用ハンドラを定義
function startElement($parser, $name, $attribs){
    print "&lt;{$name}&gt;<BR>¥n";
}

// 終了タグ用ハンドラを定義
function endElement($parser, $name){
    print "&lt;{$name}&gt;<BR>¥n";
}

// 文字データ用ハンドラを定義
function characterData($parser, $data){
    print "{$data}<BR>¥n";
}

$xml_parser = xml_parser_create(); // XMLパーサを生成
// 要素タグ用ハンドラを登録
xml_set_element_handler($xml_parser, "startElement", "endElement");
// 文字データ用ハンドラを登録
xml_set_character_data_handler($xml_parser, "characterData");

$fp = fopen("xmltest.xml", "r"); // XML ファイルをオープン
while ($data = fread($fp, 4096)) {
    xml_parse($parser, $data, feof($fp)) or
        die ("XMLパースエラーが発生しました。");
}
xml_parser_free($parser); // XMLパーサを解放
?>
```

```
int xml_parser_create ([string encoding])
```

XML パーサを作成し、XML パーサへのハンドルを返します。encoding によりパーサのエンコード(以下)を指定可能です。

ISO-8859-1 (デフォルト)、US-ASCII、UTF-8

3.0.6

**PHP3**  
**PHP4**  
**F**

---

```
int xml_set_object (int parser, object &obj)
```

---

**PHP4**  
**T F**

オブジェクトの内部で指定したパーサを使用可能にします。

---

```
int xml_set_element_handler(int parser, string h_start, string h_end)
```

---

**PHP3** 3.0.6  
**PHP4**  
**T F**

startおよびend要素のハンドラ関数をそれぞれ以下のh\_start、h\_endに設定します。

```
h_start(int parser, string name, array attribs)
```

nameにはコールされた要素の名前、連想配列attribsにはすべての属性が指定されます。

```
h_end(int parser, string name)
```

nameにはコールされた要素の名前が指定されます。

<member age="31">Sasaki</member> を処理する場合、ハンドラ関数はそれぞれ以下のようにコールされます。

```
h_start($parser, "member", array("age">="31"));
```

```
h_end($parser, "member");
```

---

```
int xml_set_character_data_handler(int parser, string handler)
```

---

**PHP3** 3.0.6  
**PHP4**  
**T F**

文字データ用ハンドラ関数を設定します。ハンドラ関数は以下のように定義します。

```
handler(int parser, string data)
```

dataには文字データが指定されます。

<member age="31">Sasaki</member> を処理する場合、ハンドラ関数は以下のようにコールされます。

```
handle($parser, "Sasaki");
```

---

```
int xml_set_processing_instruction_handler(int parser, string handler)
```

---

**PHP3** 3.0.6  
**PHP4**  
**T F**

XMLパーサの処理命令(PI)用ハンドラ関数を設定します。ハンドラ関数は以下のように定義します。処理命令は、<? ... ?>と記述します。

```
handler(int parser, string target, string data)
```

targetにはPIのターゲット、dataにはPIデータが指定されます。

<?php print "Hello!"?> を処理する場合、ハンドラ関数は以下のようにコールされます。

```
handler($parser, "php", "Hello!");
```

---

```
int xml_set_default_handler(int parser, string handler)
```

---

3.0.6 **PHP3**  
**PHP4**  
**T F**

デフォルトのハンドラを設定します。デフォルトのハンドラは、通常のハンドラが設定されていない場合にコールされます。ハンドラ関数の仕様は、xml\_set\_character\_data\_handler()と同じです。

<member age="31">Sasaki</member> を処理する場合、ハンドラ関数は以下のようにコールされます(要素ハンドラ、文字データハンドラが未定義の場合)。

```
handler($parser, "<member age=¥\"31¥\">Sasaki</member>");
```

---

```
int xml_set_unparsed_entity_decl_handler (int parser, string handler)
```

---

3.0.6 **PHP3**  
**PHP4**  
**T F**

以下のような外部エンティティのハンドラ関数を設定します。

```
<!ENTITY name {PUBLIC pid|SYSTEM sid} NDATA notation>
```

ハンドラ関数は以下のように定義します。

```
handler(int parser, string name, string base,  
        string sid, string pid, string notation)
```

name : エンティティ名

base : 現在は使用せず(常に空文字列)

sid : システムID

pid : パブリックID

notation : 表記法(NOTATION命令で事前に指定)

<!ENTITY png123 SYSTEM "123.png" NDATA png> を処理した場合、ハンドラ関数は以下のようにコールされます。

```
handler($parser, "png123", "", "123.png", "", "png");
```

---

```
int xml_set_notation_decl_handler (int parser, string handler)
```

---

**PHP3** 3.0.6  
**PHP4**  
**T F**

NOTATION 命令用のハンドラ関数を設定します。

```
<!NOTATION name {PUBLIC pid|SYSTEM sid}>
```

ハンドラ関数は以下のように定義します。

```
handler(int parser, string notation, string base,
        string sid, string pid)
```

引数の内容は、xml\_set\_unparsed\_entity\_decl\_handler()のハンドラ関数と同じです。

---

```
int xml_set_external_entity_ref_handler (int parser, string handler)
```

---

**PHP3** 3.0.6  
**PHP4**  
**T F**

外部エンティティ参照ハンドラ関数を設定します。ハンドラ関数は以下のように定義します。PHPのXMLパーサ( expat )は外部エンティティを直接解釈せず、このハンドラをコールします<sup>\*17</sup>。

```
handler(int parser, string open, string base,
        string sid, string pid)
```

引数は以下のように定義します。

open : 空白で区切ったエンティティのリスト

base : 使用せず( 常に空文字列 )

sid : システムID

pid : パブリックID

<!ENTITY tag\_ext SYSTEM "ext.xml">を宣言したあとで &tag\_ext; を使用した場合、以下のようにハンドラがコールされます。

```
handler($parser,"tag_ext","", "ext.xml","");
```

---

```
int xml_parse(int parser, string data [, int isfinal])
```

---

**PHP3** 3.0.6  
**PHP4**  
**T F**

XML ドキュメントの処理を開始します。dataには処理するデータを指定します。isfinalにtrueを指定することにより、最後のデータであることを指示します。処理を行う際には、必要に応じて設定されたイベントのハンドラがコールされます。

<sup>\*17</sup>

将来的には外部エンティティのパーサもサポートされる予定です。



<b>int xml_get_error_code(int parser)</b>	3.0.6	PHP3 PHP4 F
XMLパーサのエラーコードを取得します。		
<b>string xml_error_string(int code)</b>	3.0.6	PHP3 PHP4 F
エラーコードを指定してエラー文字列を取得します。		
<b>int xml_get_current_line_number(int parser)</b>	3.0.6	PHP3 PHP4 F
現在処理中の行番号を取得します。		
<b>int xml_get_current_column_number(int parser)</b>	3.0.6	PHP3 PHP4 T F
現在処理中のカラム番号を取得します。		
<pre>if (!xml_parser(\$parser, \$data)){ // パーサエラーが発生した場合     \$line = xml_get_current_line_number(\$parser);     \$col = xml_get_current_column_number(\$parser);     print "\$line 行の \$col 列目でエラーが発生しました。"; }</pre>		
<b>int xml_get_current_byte_index(int parser)</b>	3.0.6	PHP3 PHP4 T F
現在の処理位置をバイト単位で取得します。		
<b>int xml_parser_free(int parser)</b>	3.0.6	PHP3 PHP4 T F
XMLパーサを解放します。		
<b>int xml_parser_set_option(int parser, int option, mixed value)</b>	3.0.6	PHP3 PHP4 T F
以下のオプションの値を設定します。		
XML_OPTION_CASE_FOLDING	: タグの大文字変換を有効にする場合にtrue(デフォルト)	
XML_OPTION_TARGET_ENCODING	: ターゲットエンコード(ISO-8859-1、US-ASCII、UTF-8。デフォルトはソースエンコードと同じ)	

---

**mixed xml\_parser\_get\_option(int parser, int option)**

---

**PHP3** 3.0.6**PHP4****F**

XML パーサからオプションを取得します。取得可能なオプションは、xml\_parser\_set\_option()を参照してください。

---

**string utf8\_decode(string data)**

---

**PHP3** 3.0.6**PHP4****F**

UTF-8エンコードの文字列をISO-8859-1に変換します。

---

**string utf8\_encode (string data)**

---

**PHP3** 3.0.6**PHP4****F**

ISO-8859-1エンコードの文字列をUTF-8に変換します。

# Hypertext Preprocessor



## Chapter - 9

## PHP 関数

### その他の関数

その他の関数



```
int connection_aborted(void)
```

クライアントとの接続が断たれた場合にtrueを返します。

PHP3  
PHP4

```
int connection_status(void)
```

接続ステータスを返します。

PHP3  
PHP4

```
int connection_timeout(void)
```

スクリプトがタイムアウトした場合にtrueを返します。

PHP3  
PHP4

```
int define(string name, mixed value [, int icode])
```

定数nameを値valueで定義します。icodeに1を指定した場合、定数の大文字・小文字は区別されません(デフォルトでは区別する)。

PHP3  
PHP4

```
define("NAME", "Nakamura Taro"); // 定数NAMEを定義します  
print NAME; // "Nakamura Taro"が表示されます
```

```
int defined(string name)
```

定数nameが定義されている場合にtrue、それ以外の場合にはfalseを返します。

PHP3  
PHP4

---

**void die(string message)**

---

**PHP3**

**PHP4**

messageを出力し、スクリプトの処理を終了します。

```
// ファイルをオープンできない場合、実行を終了します
$fd = fopen("/tmp/some.dat","r") or die("ファイルをオープンできません");
```

---

**void eval(string str)**

---

**PHP3**

**PHP4**

文字列をPHPコードとして評価します。動的にPHPコードを生成するといった用途に使用できます。適切にエスケープを行う必要があります。

```
<?php
$string = 'pen';
$str = 'This is a $string.<BR>'; // ''でくくっているので変数は評価されません
print $str; // "This is a $string." が表示されます
eval( "¥$str = ¥"$str¥";" ); // $strを評価します
print $str; // "This is a pen." が表示されます
?>
```

---

**int func\_get\_arg (int i)**

---

**PHP4**

ユーザ定義関数のi番目(最初の引数が0)の引数を返します。引数の数を可変とする際に使用します。

```
<?php
function test() {
    $num = func_num_args(); // 引数の数を取得
    if ($num >= 2){
        print "2番目の引数:" . func_get_arg (1); // 2番目の引数を表示
    }
}
test(1,2,3);
?>
```

---

**array func\_get\_args (void)**

---

**PHP4**

ユーザ定義関数の引数を配列として返します。

```
<?php
function test() {
    $args = func_get_args();
    for ( $i = 0; $i < count($args); $i++ ) {
        print "$i 番目の引数:" . $args[$i] . "<BR>¥n";
    }
}
test(1,2,3);
?>
```

---

```
int func_num_args (void)
```

---

PHP4

ユーザ定義関数の引数の数を返します。

---

```
void exit(void)
```

---

PHP3  
PHP4

スクリプトの処理を終了します。

---

```
int function_exists(string fun)
```

---

PHP3  
PHP4

関数funが存在するかどうかを確認します。関数が存在する場合にtrue、存在しない場合にはfalseを返します。

---

```
object get_browser([string user_agent])
```

---

PHP3  
PHP4  
F

ユーザのブラウザの機能を検索し、オブジェクトとして返します。設定ファイルのbrowscapにてbrowscap.iniの位置を指定する必要があります。user\_agentを指定した場合は、その文字列について機能を検索します(デフォルトは\$HTTP\_USER\_AGENTを使用)。

---

```
int highlight_string (string str)
```

---

PHP4  
T F

文字列の内容を用いてPHP構文をハイライト表示したHTML出力を行います。

---

```
int highlight_file (string filename)
```

---

PHP4  
T F

ファイルの内容を用いてPHP構文をハイライト表示したHTML出力を行います。

```
int ignore_user_abort([int val])
```

**PHP3** 3.0.7  
**PHP4**

クライアントとの接続が断たれた場合に、スクリプトの実行を中断するかどうかを設定します。引数を省略した場合、現在の設定が返されます。

```
string pack(string format, mixed [args]...)
```

**PHP3**  
**PHP4**

引数をフォーマットに基づきバイナリ文字列にパックし、バイナリ文字列として返します。フォーマット指定用のコードも含めて動作はPerlの同名の関数と同じです。

フォーマットは表3-28のパラメータ+反復指定子の形式で指定します。反復指定子には整数値または\*(最後まで反復)を指定します。

```
// 6バイト長のバイナリ文字列(0x12, 0x34, 0x78, 0x56, 0x41, 0x42)を作成します
$binarydata = pack("nvc*", 0x1234, 0x5678, 65, 66);
```

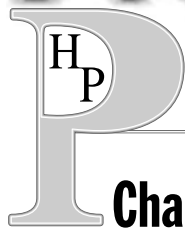
表3-28 フォーマット指定用文字

文字	意 味
a	ASCII文字列。余った場合はヌル文字を詰める
A	ASCII文字列。余った場合は空白文字を詰める
h	16進数表現の文字列。下位バイトが先
H	16進数表現の文字列。上位バイトが先
c	符号付き char
C	符号なし char
s	符号つき short (常に16ビット、マシンバイトオーダー)
S	符号なし short (常に16ビット、マシンバイトオーダー)
n	符号なし short (常に16ビット、ビッグインディアンバイトオーダー)
v	符号なし short (常に16ビット、リトルインディアンオーダー)
i	符号つき integer (サイズおよびバイトオーダーは機種依存)
l	符号なし integer (サイズおよびバイトオーダーは機種依存)
l	符号つき long (常に32ビット、マシンバイトオーダー)
L	符号なし long (常に32ビット、マシンバイトオーダー)
N	符号なし long (常に32ビット、ビッグインディアンバイトオーダー)
V	符号なし long (常に32ビット、リトルインディアンバイトオーダー)
f	float (サイズおよびバイトオーダーは機種依存)
d	double (サイズおよびバイトオーダーは機種依存)
x	ヌルバイト
X	1バイト戻る
@	絶対位置までヌルで埋める

<b>int register_shutdown_function(string func)</b>	3.0.4	PHP3 PHP4
スクリプト実行終了時に関数funcを実行するように登録します。		
<b>string serialize(mixed value)</b>	3.0.5	PHP3 PHP4
値valueをシリアル化したバイト文字列を返します。オブジェクトや配列などの型や構造を失わずにPHPの値を保存したい場合に使用します。オブジェクトのメソッドはシリアル化されません。		
<b>int show_source (string filename)</b>		PHP4 T F
ファイルの内容を使用してPHP 構文をハイライト表示したHTML 出力を行います。		
<b>void sleep(int sec)</b>		PHP3 PHP4
sec秒の間、プログラムの実行を停止させます。		
<b>int uniqid(string prefix[, boolean lcg])</b>		PHP3 PHP4
ミリ秒単位の現在時刻に基づいて、先頭にprefix がついたユニークなID を返します。lcg に true を指定した場合、combined LCG が追加され、よりユニークな値となります。		
<b>array unpack(string format, string data)</b>		PHP3 PHP4
format に基づきバイナリ文字列data を解釈し、結果を連想配列として返します。 フォーマットの指定方法はpack() と同じです。分割する要素ごとにフォーマット文字列のうしろに連想配列のキーを指定し、スラッシュ( / ) で区切ります。		
<code>\$a = unpack("c2chars/nint", \$bin); // キーは"chars1","chars2","int"となります</code>		
<b>mixed unserialize(string str)</b>	3.0.5	PHP3 PHP4
シリアル化された変数を元の型の変数として返します。		
<b>void usleep(int micro_sec)</b>		PHP3 PHP4
micro_sec マイクロ秒の間、プログラムの実行を停止します。		



# Hypertext Preprocessor



# PHP

## Chapter - 10

## オプション

PHP の設定ファイル( `php3.ini` または `php.ini` )において設定可能なオプションを示します。

### 10.1 全般的な設定オプション

変数	引数の型	内 容
<code>asp_tags</code>	boolean	ASP 形式のタグ <code>&lt;% %&gt;</code> ( <code>&lt;%= \$value%&gt;</code> による出力 ) を使用可能にする場合に On
<code>auto_append_file</code>	string	スクリプトの最後に file を自動的に付加( none の場合は追加せず ) *18
<code>auto_prepend_file</code>	string	スクリプトの先頭に file を自動的に付加( none の場合は追加せず ) *18
<code>display_errors</code>	boolean	エラーを HTML として出力する場合に On
<code>default_charset</code>	string	デフォルトの charset *19
<code>default_mimetype</code>	string	MIME タイプのデフォルト値 : デフォルトは text/html *19
<code>engine</code>	boolean	PHP の処理を無効とする場合に Off( 仮想サーバ等で使用 ) *20
<code>error_log</code>	string	エラーメッセージの記録先( ファイル名または syslog )
<code>error_reporting</code>	integer	エラー表示のレベル( 設定値は表 3-10 の値の論理和 )
<code>open_basedir</code>	string	オープン可能なファイルを指定ディレクトリ以下に制限
<code>gpc_order</code>	string	GET/POST/COOKIE 変数の処理の順番( デフォルトは GPC )
<code>ignore_user_abort</code>	boolean	接続断時に実行を継続する場合に On( デフォルトは Off )
<code>include_path</code>	string	ファイルをさがすディレクトリ( デフォルトはカレントディレクトリ )
<code>magic_quotes_gpc</code>	boolean	入力中の '、"、¥、ヌル文字を ¥ でエスケープする場合に On
<code>magic_quotes_runtime</code>	boolean	外部ソースからデータを返す関数の引数の ' を ¥ でエスケープする場合に On
<code>magic_quotes_sybase</code>	boolean	On の場合、' ' でエスケープ
<code>max_execution_time</code>	integer	スクリプトがパーサにより強制終了されるまでの最大許容時間( 秒 )
<code>memory_limit</code>	integer	スクリプトが確保できる許容最大メモリ( バイト )
<code>short_open_tag</code>	boolean	PHP タグの短縮型( <code>&lt;? ?&gt;</code> ) を使用可能とする場合に On( デフォルト )
<code>track_errors</code>	boolean	最新のエラーメッセージを <code>\$php_errormsg</code> で参照可能とする場合に On
<code>upload_tmp_dir</code>	string	ファイルアップロード時にファイル保存に用いるテンポラリディレクトリ
<code>user_dir</code>	string	ユーザのホームディレクトリとして使用するディレクトリ( 例 : public_html )
<code>warn_plus_overloading</code>	boolean	On の場合、文字列中に '+' が使用されている場合に警告を出力
<code>bcmath.scale</code>	integer	任意精度関数における小数点以下の有効桁数
<code>browscap</code>	string	ブラウザ機能ファイル( browscap.ini ) の名前
<code>output_buffering</code>	boolean	出力のバッファリングを有効にする場合に On( デフォルトは Off ) *19

\*18

ファイルは `include_path` で指定したパスから検索されます。

\*19

PHP4 のみ。

\*20

Apache モジュール版のみ。



## 10.2 メール設定用オプション

変数	引数の型	内 容
SMTP	string	Windows 版 PHP が使用する SMTP サーバーの DNS 名または IP アドレス
sendmail_from	string	Windows 版 PHP の送信メールで From: に使用されるメールアドレス
sendmail_path	string	sendmail( または代替品 ) のパス ( /usr/sbin/sendmail など )

## 10.3 セーフモード設定用オプション

変数	引数の型	内 容
safe_mode	boolean	セーフモードを有効にする場合に On( デフォルトは Off )
safe_mode_exec_dir	string	セーフモード時に実行可能なプログラムを置くディレクトリ
safe_mode_allowed_env_vars	string	セーフモード時に変更可能な環境変数
safe_mode_protected_env_vars	string	セーフモード時に変更不可能な環境変数
doc_root	string	セーフモード時に指定ディレクトリ以下のみアクセス可
disable_functions	string	ユーザが実行できない関数( セーフモードの設定とは無関係 )

## 10.4 拡張モジュールロード用オプション

変数	引数の型	内 容
enable_dl	boolean	PHP の動的ロード機能を無効にする場合に Off( デフォルトは有効 )
extension_dir	string	拡張モジュールを置くディレクトリ
extension	string	PHP の起動時にロードする拡張モジュール

## 10.5 MySQL 設定用オプション

変数	引数の型	内 容
mysql.allow_persistent	boolean	持続的な MySQL 接続を許可する場合に On
mysql.max_persistent	integer	持続的な MySQL 接続のプロセスごとの最大数
mysql.max_links	integer	持続的な接続を含む MySQL 接続のプロセスごとの最大数

## 10.6 PostgreSQL 設定用オプション

変数	引数の型	内 容
pgsql.allow_persistent	boolean	持続的な PostgreSQL 接続を許可する場合に On
pgsql.max_persistent	integer	持続的な PostgreSQL 接続のプロセスごとの最大数
pgsql.max_links	integer	持続的な接続を含む PostgreSQL 接続のプロセスごとの最大数

## 10.7 InterBase 設定用オプション

変数	引数の型	内 容
ibase.allow_persistent	boolean	持続的な InterBase 接続を許可する場合に On
ibase.max_persistent	integer	持続的な InterBase 接続のプロセスごとの最大数
ibase.max_links	integer	持続的な接続を含む InterBase 接続のプロセスごとの最大数
ibase.default_user	string	デフォルトのユーザ名
ibase.default_password	string	デフォルトのパスワード
ibase.timestampformat	string	タイムスタンプのフォーマット: デフォルトは"%m/%d/%Y %H:%M:%S"
ibase.dateformat	string	日付のフォーマット: デフォルトは"%m/%d/%Y"
ibase.timeformat	string	時間のフォーマット: デフォルトは"%H:%M:%S"

## 10.8 Informix 設定用オプション

変数	引数の型	内 容
ifx.allow_persistent	boolean	持続的な Informix 接続を許可する場合に On
ifx.max_persistent	integer	持続的な Informix 接続のプロセスごとの最大数
ifx.max_links	integer	持続的な接続を含む Informix 接続のプロセスごとの最大数
ifx.default_host	string	デフォルトのホスト名
ifx.default_user	string	デフォルトのユーザ名
ifx.default_password	string	デフォルトのパスワード
ifx.blobinfile	boolean	ファイルに BLOB カラムを返したい場合に true、メモリに返したい場合に false
ifx.textasvarchar	boolean	SELECT でテキストカラムを文字列として返す場合に true
ifx.byteasvarchar	boolean	SELECT でバイトカラムを文字列として返す場合に true
ifx.charasvarchar	boolean	CHAR カラムの末尾の空白文字を取り除く場合に true
ifx.nullformat	boolean	NULL カラムの返り値を "NULL" とする場合に true

## 10.9 Sybase 設定用オプション

変数	引数の型	内 容
sybase.allow_persistent	boolean	持続的なSybase接続を許可する場合にOn
sybase.max_persistent	integer	持続的なSybase接続のプロセスごとの最大数
sybase.max_links	integer	持続的な接続を含むSybase接続のプロセスごとの最大数

## 10.10 ODBC 設定用オプション

変数	引数の型	内 容
uodbc.default_db	string	ODBCデータソースのデフォルト値
uodbc.default_user	string	ODBCユーザ名のデフォルト値
uodbc.default_pw	string	ODBCパスワードのデフォルト値
uodbc.allow_persistent	boolean	持続的なODBC接続を許可する場合にOn
uodbc.max_persistent	integer	持続的なODBC接続のプロセスごとの最大数
uodbc.max_links	integer	持続的な接続を含むODBC接続のプロセスごとの最大数

## 10.11 セッション設定用オプション( PHP4)

変数	引数の型	内 容
session.save_handler	string	セッション保存用ハンドラ: files( デフォルト )、mm、userのいずれか
session.save_path	string	session.save_handlerがfilesの場合に保存するパス
session.use_cookies	boolean	クッキーを使用する場合にOn
session.name	string	セッション名( デフォルトはPHPSESSID )
session.auto_start	boolean	セッションを自動的に開始する場合にOn
session.cookie_lifetime	integer	クッキーの有効期限( 0の場合はブラウザ終了まで )
session.cookie_path	string	クッキーが有効となるパス
session.cookie_domain	string	クッキーを有効とするドメイン
session.serialize_handler	string	データシリアル化用ハンドラ: php( デフォルト )またはwddx
session.gc_probability	integer	各セッションの初期化時にガーベッジコレクション( gc )が起動する確率( % )
session.gc_maxlifetime	integer	保存データが無効となり、gcプロセスにより消去されるまでの時間( 秒 )
session.referer_check	boolean	外部に保存されたIDを無効とするためにHTTP Refererをチェックする場合にOn
session.entropy_length	integer	乱数生成時にファイルから読み込むバイト数
session.entropy_file	string	セッションIDを生成する際に使用する乱数デバイスファイル
session.cache_limiter	string	HTTPキャッシュの動作: nocache( デフォルト )、private、public
session.cache_expire	integer	キャッシュが無効となるまでの時間( 分 )

## 10.12 Java 関連オプション( PHP4 )

変数	引数の型	内 容
java.class.path	string	jvm のCLASSPATH
java.home	string	jvm のHOME
java.library.path	string	jvm のライブラリパス
java.library	string	jvm のライブラリ

## 10.13 国際化関数関連オプション<sup>\*21</sup>

変数	引数の型	内 容
i18n.http_output	string	デフォルトのHTTP出力コード：EUC-JP,SJIS,JIS,UTF-8,pass ( デフォルト )
i18n.internal_encoding	string	内部コード：EUC-JP( デフォルト ),SJIS,UTF-8
i18n.script_encoding	string	スクリプトファイルのコード：EUC-JP,SJIS,JIS,UTF-8,auto( デフォルト )
i18n.http_input	string	HTTPからの入力コード：passまたはauto( デフォルト )
i18n.http_input_default	string	HTTPからの入力のデフォルト：pass( デフォルト ),EUC-JP,SJIS,JIS,UTF-8

<sup>\*21</sup>

国際化版 PHP3 でのみ使用可能。