PHP 하우투

Al Dev (Alavoor Vasudevan) `alavoor@yahoo.com`

v8.0, 26 Sep 2000

윤광호(KwangHo, Yun), `airplanez@kldp.org`

PHP-HOWTO_kr.sgml, v0.01 2000/11/07 15:50

이문서는 어떻게 PHP 프로그램을 만드는지와 Windows 95 GUI 어플리케이션을 훌륭한 PHP + HTML + DHTML + XML + Java applets + Javascript 로 바꾸는 것을 설명할것이다. 이문서에 있는 정보는 PHP가 포팅 되었있는 Linux, Windows 95/NT, OS/2, Unix 계열의 Solaris, HPUX, AIX, SCO, Sinix, BSD, 등등의 운영체계에 적용된다.

목 차

제 1 절 소개

PHP 는 'Hypertext Pre-Processor'를 의미하고 서버사이드(server side) HTML 스크립팅/프로그래밍 언어이다. PHP는 동적인 웹페이지를 만들수 있게해준다. PHP가 사용된 웹페이지는 일반적인 HTML 처럼 취급되며 평범한 HTML 페이지처럼 생성과 수정을 할수 있다.

많은 회사들이 PHP는 **"중요보안과 엄격한 비밀"**을 유지할수 있는 컴퓨터 언어라고 말하지만, 그러나 지금은 잘알려지고 많이 쓰이는 웹, 인터넷, 전자상거래와 B2B(business-to-business) 프로젝트을 위한 스크립트 언어가 되었다. 그럼에도 불구하고 요즘 많은 경쟁회사들이 PHP언어를 보안에 대해서 안전하지않다고 한다(경쟁자들).(역주:번역이 영 이상하네..)

PHP는 전세계에 돌풍을 일으킬 것이며 IT업계를 놀랍게 점유하게 될것이다!! PHP의 파워는 **크로스-플랫폼 이고 어느곳에서도 실행된다!!** PHP는 리눅스, 윈도 95/98/NT, 윈도 2000, 솔라리스, HPUX와 모든 유닉스 계열에서 실행된다. PHP는 한번작성 되면 어디서나 실행된다. PHP는 아파치, 마이크로소프트 IIS 등등의 많은 웹서버에서 실행된다.

PHP는 자바보다 5에서 20배까지 빠르다!! PHP는 사용하는데 무척쉽고, 매우 복잡한 웹/전자상거래 어플리케이션을 매우 빠르고 짧은 시간에 개발할수 있다.

PHP는 객체지향적인 특징과 자바, C++, PERL과 "C"로 부터 훌륭한 특징을 가져왔다. PHP언어는 자바, C++, PERL과 C의 훌륭한 특징의 결합체이다.

PHP는 모든 스크립트/프로그램 언어의 **진정한 보석**이며 곧 전세계 프로그래머의 "메카"가 될것이다!! PHP는 윈도 95/NT 와 모든 유닉스에서 실행되기 때문에 거대한 사용자층과 많은 개발자층을 가진다.

PHP는 보다 빠르게 실행되도록 젠옵티마이저(Zend Optimizer)를 사용하여 컴파일과 최적화(optimized)시킬수 있다. 젠옵티마이저(Zend Optimizer)는 PHP 버전 4.0에서 부터 PHP에 통합되었다.

당신은 전자상거래(e-commerce) 프로젝트를 하는데 보통 PHP (70% 코드) + HTML/DHTML/XML (25% 코드) + Javascript (5% 코드 클라이언트 측) 의 조합을 사용할것이다.

제 2 절 PHP 내려받기

- PHP 메인 사이트 `<http://www.php.net>`
- PHP 리소스 `<http://ils.unc.edu/web-db/php/links.html>`
- PHP 코드 변환 - `<http://px.sklar.com>`

2.1 Microsoft Windows 95/98/NT/2000에 PHP 설치

PHP는 Microsoft Windows 플렛폼에서 매우 인기있고 놀랍게도 MS 의 ASP 스트립트 언어보다 더 인기있다!! PHP의 가장큰 장점은 MS 윈도와 리눅스, 유닉스에서 개발할수 있고 반대로 사용할수도 있다는 것이

다!!

MS 윈도 98/NT/2000 에서 PHP의 수요가 많기 때문에 실행가능한 설치프로그램이 준비되어 있다. 단지 실행화일을 더블클릭 하는 것만으로 2분만에 PHP설치가 자동으로 이루어진다. 실행가능한 설치화일을 다운로드하라.

- MS Windows의 PHP 설치 프로그램 `<http://php.weblogs.com/easywindows>`
- 많은 윈도 플렛폼의 PHP 정보 `<http://php.weblogs.com>`

2.2 유닉스들과 다른 플렛폼에서 PHP 설치

PHP 메인사이트 `<http://www.php.net>` 나 다운로드한 패키지의 설치파일 에서 설치가이드와 지시를 보라.

제 3 절 PHP 튜토리얼

여기에서는 당신의 서버가 PHP 가 가능하고 .php3로 끝나는 모든 화일이 PHP에 의해 실행된다고 가정한다.

첫번째 PHP가 가능한 페이지: 다음의 라인을 넣어서 hello.php3란 화일을 만들어라:

```
<html>< head>< title >PHP Test< /title >< /head >
< body>
<?php echo "Hello World<P>"; ?>
< /body>< /html>
```

CGI 스크립트 처럼 보이지 않을것이다. 특별한 태그를 가지고 있는 보통 HTML 처럼 생각하라.

만약 이것을 실행했는데 아무런 결과도 나타나지 않으면, PHP가 실행되지 않은 것이다. 당신의 관리자에게 실행되게 해달고 요청하라.

이번 연습에서는 PHP의 특별한 태그를 보여주기 위한것이 목적이다. 이번 연습에서 우리는 PHP 태그의 시작을 알리는 < ?php를 사용하였다. 그리고 PHP 문장을 넣고 ? > 태그를 끝에 넣었다. 당신은 당신이 HTML 화일의 원하는 곳에 이렇게 함으로서 PHP 모드를 시작하고 끝낼수 있다.

우리는 페이지를 보는 사람이 어떤 브라우저를 사용하는지 검사해볼것이다. 그렇게 하기위해 우리는 브라우저가 보내온 요청에 대한 부분을 체크한다. 이 정보는 변수에 저장된다. 변수는 언제나 달러사인($)으로 시작한다. 우리가 관심을 가지는 변수는 $HTTP_USER_AGENT 이다. 이것을 보여주기 위해서 우리는 다음과 같이 할수 있다:

```
<?php echo $HTTP_USER_AGENT; ?>
```

지금 당신이 이 페이지를 보기 위해 사용하는 브라우저에서는, 이렇게 나온다:

Mozilla/4.0 (compatible; MSIE 4.01; Windows 98)

자신의 웹서버에 의해 자동으로 설정되는 많은 다른 변수들이 있다. 다음과 같은 화일을 만듦으로서 완전한 변수의 리스트를 얻을수 있다:

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

브라우저로 이화일을 읽어들이면 자신한데 가용한 모든 변수의 리스트를 화면가득 한 정보와 같이 볼수 있을것이다.

PHP 태그안에 여러개의 PHP 문장을 집어넣을수 있으며 하나이상의 echo 문을 가지는 코드블럭을 만들수 있다.

```
<?php
if(strstr($HTTP_USER_AGENT,"MSIE")) {
    echo "You are using Internet Explorer<br>";
}
?>
```

여기에서는 PHP 블럭의 중간일지라고 PHP모드를 벗어나는 방법을 보여줄것이다:

```
<?php
if(strstr($HTTP_USER_AGENT,"MSIE"))
{
        ?>
        < center>< b>You are using Internet Explorer< /b>< /center>
        <?
}
else
{
        ?>
        < center>< b>You are not using Internet Explorer< /b>< /center>
        <?
}
?>
```

PHP의 echo 문을 사용하는 대신 PHP 모드를 벗어나서 직접 HTML 을 넣을수 있다. 여기서 말하려고 하는 중요하고 강력한 점은 논리적인 흐름이 깨지지 않고 남아 있다는 것이다. 단지 HTML 블럭들은 결국 보는 사람에게만 보내지게 되는것이다. 스크립트를 실행시키면 결과는 다음과 같다 :

You are using Internet Explorer

폼 다루기

PHP의 가장 유용한 기능중 하나가 HTML 폼(form)을 다루는 방법이다. 그것을 이해하는 중요한 기본적인 개념은 폼에 나온 모든 요소들이 자동적으로 폼에 있는 이름과 같은 변수로 자동적으로 바뀌어 불려진(target) 곳에서 사용 된다는 것이다. 복잡하게 들릴것이다. 그래서 여기 간단한 예제가 있다. 다음과 같은 폼을 가지는 페이지가 있다고 가정하라 :

```
<form action="action.php3" method="POST">
Your name: <input type=text name=name>
You age: <input type=text name=age>
<input type=submit>
< /form>
```

여기에 있는 폼에는 어떠한 특별한 것도없다. 어떤 특별한 태그도 없는 단지 HTML 폼에 불과 하다. 사용자가 폼을 입력하고 전송버튼을 누리면 action.php3가 불려진다. 불려진 화일에는 다음과 같은 것이 들어있을것이다:

```
      Hi <?php echo $name?>.  You are <?php echo $age?> years old.
```

놀랍게도 $name과 $age 변수는 PHP에 의해 자동으로 정해진다!!

제 4 절 PHP을 위한 IDE(통합환경) 도구

많은 HTML 에디터들은 PHP를 지원한다 :

- Soyal의 훌륭한 PHP 에디터 `<http://soyal.free.fr/PHPEd>`
- Blue Fish `<http://bluefish.linuxave.net>`
- Coffee cup `<http://www.coffeecup.com/linux>`
- 드림위버(Dreamweaver) `<http://www.dreamweaver.com>`
- Amaya `<http://www.w3.org/Amaya>`
- Homesite `<http://www.allaire.com/homesite>`
- Hotdog `<http://www.hotdog.com>`
- Zend Optimizers(전옵티마이저) `<http://www.zend.com>`
- Zend Compilers(전컴파일러) `<http://www.zend.com>`
- MS Windows 플렛폼용의 많은 PHP 정보 `<http://php.weblogs.com>`

가까운 시일에 모든 HTML 에디터와 XML 에디터가 "빠른 어플리케이션 개발" 도구로 PHP를 지원할것이다. .

제 5 절 놀라운 !!! PHP를 위한 ctags !

이 태그들은 상당히 중요하고 vi, emacs, CRiSP, NEdit 등과 같은 에디터로 소스코드를 탐색(검색)하는데 쓰여진다. 만약 C, C++ 또는 자바로 프로그램한 것을 가지고 있다면 태그를 생성하기 위해 **ctags** 프로그램을 사용할수 있을것이다. 온라인 메뉴얼 페이지(online manual page)를 보기위해서는 리눅스/유닉스 bash prompt 에서 'man ctags'를 치면 된다.

PHP를 위한 **ptags** 프로그램은 다음에 있다. 그것을 사용하면 PHP 소스코드를 위한 태그를 생성할수 있다. **ptags**를 사용하면 당신의 **생산성은 3에서 4배까량 향상될것이다**.

또한 `<http://metalab.unc.edu/LDP/HOWTO/Vim-HOWTO.html>`에 있는 PHP, C, C++를 위한 Vim 컬러 텍스트 에디터를 참조하라.

```
// ptags.cpp로 이화일을 저장하고
// g++ -o ptags ptags.cpp 로 컴파일하라.
//*****************************************************************
// GNU/GPL 저작권을 따른다. 추가로 복사를 하게되면 저자의 이름,
// 이메일을 포함시켜라.
// 저자 : Al Dev Email: alavoor@yahoo.com
// Usage : ptags *.php3 *.inc
//                 This will generate a file called tags
//*****************************************************************
```

```
#include <iostream.h>
#include <fstream>
#include <stdio.h> // for sprintf
#include <stdlib.h> // for system
#include <string.h> // for memset
#include <ctype.h> // for isspace

#define BUFF_LEN  1024
#define LOCATION  9

char *ltrim(char *dd);
char *rtrim(char *ee);

main(int argc, char **argv)
{
        if (argc < 2)
        {
                cerr << "\nUsage: " << argv[0] << " file .... " << endl;
                exit(0);
        }

        char fname[100] = "tag_file.out";
        FILE    *fpout;
        ofstream    fout(fname);
        if (fout.fail())
        {
                cerr << "\nError opening file : " << fname << endl;
                exit(-1);
        }
        //fpout = fopen(fname, "w");

        for (int ii = 1; ii < argc; ii++)
        {
                /*
                char buff[2024];

                sprintf(buff, "\\rm -f %s; ls %s > %s 2>/dev/null", outfile, argv[1], outfile);
                cout << "\nbuff = " << buff << endl;

                system(buff);
                fclose(fp);
                */
                FILE *fpin = NULL;
                fpin = fopen(argv[ii], "r");
                if (fpin == NULL)
                {
                        cerr << "\nError opening file : " << argv[ii] << endl;
                        exit(-1);
                }
                char buff[BUFF_LEN + 100];
```

```
                memset(buff, 0, BUFF_LEN +10);
                for ( ; fgets(buff, BUFF_LEN, fpin) != NULL; )
                {
                        char aa[BUFF_LEN + 100];
                        memset(aa, 0, BUFF_LEN +10);
                        strcpy(aa, buff);
                        ltrim(aa);

                        // Remove the trailing new line..
                        {
                                int tmpii = strlen(aa);
                                if (aa[tmpii-1] == ’\n’)
                                        aa[tmpii-1] = 0;
                        }
                        //cout << "aa is : " << aa << endl;
                        if (strncmp(aa, "function ", LOCATION) != 0)
                                continue;
                        //cout << buff << endl;

                        // Example tags file output is like -
                        // al2  al.c    /^al2()$/;"     f
                        {
                                char bb[BUFF_LEN + 100];
                                memset(bb, 0, BUFF_LEN +10);
                                strcpy(bb, & aa[LOCATION]);
                                char *cc = bb;
                                while (cc != NULL && *cc != ’(’)
                                        *cc++;
                                *cc = 0;
                                cc = rtrim(bb);
                                //cout << "bb is : " << bb << endl;
                                //cout << cc << "\t" << argv[ii] << "\t" << "/^" << aa << "$/;\"\tf" << endl;
                                fout << cc << "\t" << argv[ii] << "\t" << "/^" << aa << "$/;\"\tf" << endl;
                                //fprintf(fpout, "%s\t%s\t/^%s$/;\"f\n", cc, argv[ii], aa );
                        }

                        memset(buff, 0, BUFF_LEN +10);
                }
                fclose(fpin);
        }
        fout.flush();
        fout.close();
        //fclose(fpout);

        // Sort and generate the tag file
        {
                char tmpaa[1024];
                sprintf(tmpaa, "sort %s > tags; \\rm -f %s", fname, fname);
                system(tmpaa);
        }
```

```
}

char *ltrim(char *dd)
{
    if (dd == NULL)
        return NULL;

    while (isspace(*dd))
        dd++;

        return dd;
}

char *rtrim(char *ee)
{
    if (ee == NULL)
        return NULL;

        int tmpii = strlen(ee) - 1;
        for (; tmpii >= 0 ; tmpii--)
        {
                if (isspace(ee[tmpii]) )
                {
                        //cout << "\nis a space!!" << endl;
                        ee[tmpii] = 0;
                }
        }
        return ee;
}
```

제 6 절 PHP 디버깅

PHP 프로그램을 디버깅 하려면 다음의 함수를 가지는 "debug2.inc"이란 화일을 만들어라 :

```
<?php

/* 이중으로 선언되는 것을 방지하기 위한 변수정의 */
if (!defined("_DEBUG2_DEFINED_"))
{
        define("_DEBUG2_DEFINED_", 1 );
}
else
        return; // 만약 이화일이 include 됐다면 리턴
# file name : debug2.inc
# PHP 소스 코드를 디버깅하기 위한 함수들
#*****************************************************************
# Copyright policy is GNU/GPL but additional request is
# that you include author's name and email on all copies
# Author : Al Dev Email: alavoor@yahoo.com
```

```
#*****************************************************************

# Usage of this functions -
# In your source code put something like -
# debug2_(__FILE__, __LINE__, "f_somevariable", $f_somevariable);
# And this will generate output in debug.out file.

//function debug2_($fname, $lname, $debug_var, $debug_value=0) {}

// Give read, exec for all on directory /debug2_logs
// chmod a+rwx /debug2_logs
// But here you need to open the file in append mode.
$fp_debug2 = fopen("/debug2_logs/debug.out", "a");
if ($fp_debug2 == false)
{
        print "<b>File open failed - global.var.inc<b>";
        exit;
}

function debug2_($fname, $lname, $debug_var, $debug_value=0)
{
        global $fp_debug2;

        //print "<br> debug_value is : $debug_value <br>";
        if (!$debug_value)
        {
                fwrite($fp_debug2, "\n ". $fname ."  ". $lname .": $debug_var");
        }
        else
        {
                fwrite($fp_debug2, "\n ". $fname . " ". $lname .": $debug_var = $debug_value");
        }
        //print "<br> f_cookie is : $f_cookie <br>";
}

// In your first page, which is generally index.php3
// truncate the debug2_logs file in beginning of code
function init_debug_file()
{
        global $fp_debug2;

        $fp_debug2 = fopen("/debug2_logs/debug.out", "w");
        if ($fp_debug2 == false)
        {
                print "<b>File open failed - global.var.inc<b>";
                exit;
        }
        system("chmod a+rwx /debug2_logs/debug.out");
}
```

```
?>
```

보통 index.php3로 되는 시작페이지의 PHP 소스코드에 다음을 집어넣어라.

```
<?php
        include ("debug2.inc");

        init_debug_file();
        // 다른 명령은 여기에.....
        // ..........
?>
```

디버깅 값을 얻기위해 PHP 소스코드 파일에 debug2_() 란 호출을 아래와 같이 넣어라

```
<?php
include ("debug2.inc");
debug2_(__FILE__, __LINE__, "f_somevariable", $f_somevariable);

function aa()
{
        $aa = 8;
        debug2_(__FILE__, __LINE__, "aa", $aa);
}
?>
```

PHP 프로그램을 실행하게 되면, 결과가 debug.out 이란 화일에 파일이름, 라인번호, 변수명 과 변수값이 나타난다.

일번적으로 debug2_()를 사용하라. 프로그램에서 debug2_()의 호출은 최종결과 코드에 어떠한 영향도 주지않으며 실행에도 아무런 영향을 주지않는다. 왜냐하면 아래에 기술한것 같이 필터링 되기 때문이다. debug2_()를 타이핑하는 시간을 줄이기 위해 복사(copy)와 붙여넣기(paste)를 사용할수 있다. 또한 Vi 에디터의 'yank to buffer' 기능을 이용해서 복사하라.

개발이 완료되면 테스트하고 서버에 올린 준비가 됐으면 소스코드에서 debug2_ 호출을 필터링 하라. 유닉스 프롬프트에서 -

---

```
bash$ mkdir production
bash$ grep -v debug2_  filea.php3 > production/filea.php3
```

여러개의 화일이 있으면 -

```
bash$ mkdir production
bash$ ls *.php3 | while read ans
do
        grep -v  debug2_ $ans > production/$ans
done
```

그리고 이제 production 에서 개발하는곳(영역)을 복사하라.

제 7 절 PHP의 제한

모든 사람이 한계와 단점이 있듯이 PHP도 예외가 아니다. 아래에 있는것은 PHP의 제약(한계)이다.(그래서 **경고한다 !!**)

1. PHP는 100% 순수한 객체지향 스크립트 언어가 아니다. PHP는 코드의 크기가 300,000 라인을 넘지 않으면 괜찬다. 300,000 라인이 넘는 PHP코드를 관리하기는 좀더 힘들어질것이다.
2. PHP는 "C"나 "C++" 언어의 성능을 주지 모한다. 왜냐하면 PHP는 스크립트 언어이고 실행방식이 인터프리터 형식이기 때문에 최적화된 "C++"보다 다소 느리다. 최고의 성능을 원한다면 "C++"과 커넥션 풀링(connection pooling)된 데이타베이스/웹서버와 연동된 빠른 CGI 를 사용하고 C++ 컴파일러 옵티마이저를 "-03" 옵션을 사용하라. PHP 4에 있는젠드옵티마이져는 PHP의 성능을 어느정도 향상시킬것이다.

한편으로는 PHP는 많은 장점을 가지고 그 장점은 제한을 보충할수 있다 -

1. PHP 스크립트 언어에서는 컴파일과 링크가 생략되기 때문에 매우 빠르게 웹어플리케이션을 개발할 수있다.
2. PHP 에플리케이션은 매우 안정적이고 자바스크립트가 브라우저에 의존적인것과 다르게 브라아져의 기술에 의존적이지 않다. PHP는 어떤 서버 플렛폼과 어떤 브라우저를 선택해야하는 것에 대한 자유를 주고 브라우져는 HTML 이 PHP가 만들어냈다는 것을 모른다!
3. PHP는 모든 SQL 테이터베이스 서버와의 연결성이 뛰어나다.
4. PHP는 부분적으로 객체지향 특성을 가지고 있다.
5. PHP는 "C++", Perl, Javascript와 비슷한 문법을 가지고 소스코드를 탐색할수 있는 프로그램과 비슷한 'ptags/ctags'를 가지고 있다.
6. PHP는 속도를 향상시킬수 있는 젠드옵티마이져를 가지고 있다.
7. PHP는 모든 유닉스들, 리눅스, 윈도 95/NT/200 에서 실행되며 ASP, JSP와 다른것보다 성능이 좋다.
8. PHP는 많은 사용자층와 개발자층을 가지고 있다.

경고: 만약 100% 순수한 객체지향 스크립트 언어를 원한다면 "반드시" **Python**을 염두해라. 'Python'은 처음부터 객체지향 스크립트언어이다. 'Zope'라고 불리는 Python 웹 어플리케이션 서버를 사용할수 있다. 'Zope'는 `<http://www.zope.org>` 에서 있고 python은 `<http://www.python.org>`에 있다.

제 8 절 관계있는 URL들

C, C++에 관계있는 다음 장소를 방문하라 -

- C++, C을 위한 컬러 Vim 텍스트 에디터 `<http://metalab.unc.edu/LDP/HOWTO/Vim-HOWTO.html>`
- PHP PostgreSQL를 위한 SQL database server `<http://metalab.unc.edu/LDP/HOWTO/PostgreSQL-HOWTO.html>`
- C++ 프로그램을 위한 Source code control system CVS HOWTO `<http://metalab.unc.edu/LDP/HOWTO/CVS-HOWTO.html>`
- Linux goodies main site `<http://www.aldev.8m.com>`
- Linux goodies mirror site `<http://aldev.webjump.com>`

제 9 절 이문서의 다른 포멧

이문서는 DVI, Postscript, Latex, Adobe Acrobat PDF, LyX, GNU-info, HTML, RTF(Rich Text Format), Plain-text, Unix man pages 와 SGML 11가지의 다른 포멧이 있다.

- 이 HOWTO 문서을 HTML, DVI, Postscritp 또는 SGML 포멧으로 된 tar ball 파일을 `<ftp://sunsite.unc.edu/pub/Linux/docs/HOWTO/other-formats/>` 에서 구할수 있다.
- 단순한 텍스트 포멧은 `<ftp://sunsite.unc.edu/pub/Linux/docs/HOWTO>`에서
- 불어, 독어, 서반아어, 중국어, 일본어로 번역된 문서는 `<ftp://sunsite.unc.edu/pub/Linux/docs/HOWTO>` 에서 구할수 있다. 다른 언어로 번역하실 분의 도움을 환영한다.

이문서는 `<http://www.sgmltools.org>`에서 구할수 있는 "SGML-Tools"란 툴을 사용해서 작성되었다. 다음과 같은 명령을 사용해서 소스를 컴파일 할수 있다.

- sgml2html CVS-HOWTO.sgml (html 화일을 만들기 위해)
- sgml2rtf CVS-HOWTO.sgml (RTF 화일을 만들기 위해)
- sgml2latex CVS-HOWTO.sgml (latex 화일을 만들기 위해)

LaTex 문서는 **sgml2latex** (와 dvips) 그리고 아크로바트(Acrobat) **distill** (`<http://www.adobe.com>`) 을 사용한 포스트스크립트 결과 처리로 PDF화일로 쉽게 바꿀수 있다. 다음음 명령들이다:

```
bash$ man sgml2latex
bash$ sgml2latex filename.sgml
bash$ man dvips
bash$ dvips -o filename.ps filename.dvi
bash$ distill filename.ps
bash$ man ghostscript
bash$ man ps2pdf
bash$ ps2pdf input.ps output.pdf
bash$ acroread output.pdf &
```

또는 고스트스크립(Ghostscript) 명령 **ps2pdf**을 사용할수 있다. ps2pdf는 Adobe's Acrobat Distiller product의 거의 모든 기능과 비슷하게 동작한다. 포스트스크립트 화일을 PDF(Portable Document Format)으로 바꾸어준다. **ps2pdf**는 명령 스크립트(command script/배치파일)가 Ghostscript를 불러내고 , **pdfwrite**라고 불리는 특별한 출력 디바이스를 선택하여 실행된다. ps2pdf를 사용하기 위해 Ghostscript를 컴파일 할때 pdfwrite 장치가 makefile에 포함되어야한다; 자세한 것은 Ghostscript 설치 문서를 보아라.

이문서는 다음에서 찾을수 있다 -

- `<http://sunsite.unc.edu/LDP/HOWTO/CVS-HOWTO.html>`

또한 다음의 미러 사이트에서 이문서를 찾을수 있다 -

- `<http://www.caldera.com/LDP/HOWTO/CVS-HOWTO.html>`
- `<http://www.WGS.com/LDP/HOWTO/CVS-HOWTO.html>`
- `<http://www.cc.gatech.edu/linux/LDP/HOWTO/CVS-HOWTO.html>`
- `<http://www.redhat.com/linux-info/ldp/HOWTO/CVS-HOWTO.html>`

- 다른 당신과 가까운 미러사이트(network-address-wise)는 `<http://sunsite.unc.edu/LDP/hmirrors.html>`에서 찾을수 있다. 사이트를 선택하고 /LDP/HOWTO/CVS-HOWTO.html 디렉토리고 가라.

그 dvi 포멧으로 되어있는 그 문서를 보기위해서는 xdvi란 프로그램을 사용하라. xdvi 프로그램은 래드햇 리눅스의 tetex-xdvi*.rpm 패키지에 있고 메뉴버튼 ControlPanel | Applications | Publishing | TeX 에 있다. dvi 문서를 읽기 위해 다음의 명령을 실행하라 -

```
xdvi -geometry 80x90 howto.dvi
man xdvi
```

그리고 마우스를 사용하여 윈도우 크기를 조절하라. 화살표키, Page Up, Page Down keys 와 'f', 'd', 'u', 'c', 'l', 'r', 'p', 'n' 를 이용하여 위, 아래, 중간, 다음페이지, 전 페이지 등을 탐색할수 있다. 실행을 끝내려면 'x'를 눌러라.

'gv'(ghostview)나 'ghostscript'란 프로그램을 사용해서 포스트스크립트 화일을 읽을수 있다. 래드햇 리눅스에서 ghostscript 는 ghostscript*.rpm 패키지에 gv는 gv*.rpm 패키지에 있다. 그리고 ControlPanel | Applications | Graphics 메뉴 버튼에 위치한다. gv 가 ghostscript보다 더 친근한 프로그램이다. 그리고 ghostscript와 gv는 OS/2, Windows 95 와 NT 용도 존재한다. 그리고 그 운용체제에서 이 문서도 볼수 있다.

- Get ghostscript for Windows 95, OS/2, and for all OSes from `<http://www.cs.wisc.edu/~ghost>`

포스트스크립트 문서를 읽기 위해 다음을 입력하라 -

```
gv howto.ps
ghostscript howto.ps
```

네스케이프 네비게이터, 마이크로소프트 인터넷 익스폴로러, 래드햇 바론 웹브라우져 또는 10가지 이상의 웹부라우져에서 HTML 포맷문서를 읽을수 있다.

LyX 라는 X-Windows용 프로그램으로 latex 문서를 읽을수 있다.

제 10 절 저작권

저작권정책은 LDP(리눅스 문서 프로젝트)의 GNU/GPL을 따른다. LDP는 GNU/GPL 프로젝트이다. 추가적인 요구 - 저작자의 이름, 이메일주소와 이 저작권사항을 유지하라. 만약 당신이 이문서에 어떠한 수정이나 첨가를 할경우 이문서의 저자에게 알려주어라.

제 11 절 부록 A Database Wrapper 예제

제출자 : Barton Greg `greg@createtech.com` 이화일을 얻으려면 웹브라우져에서 '다른이름으로 저장'을 누르고 pgsql.lib 라고 입력하시오.

```
이것은 PostgresSQL을 위한 데이터베이스 랩퍼이지만 다른 형식의
데이터베이스를 위해 조금만 고치면 사용할수 있다.


<?php

if ($dbObjDefined != 1)
```

```
{
        $dbObjDefined = 1;

        // Wrapper class for database calls
        class dbObj
        {
                // Connection handle to database
                var $conn;

                // Default connection parameters
                var $host = "YourSite.com";
                var $user = "johndoe";
                var $password = "pwd";
                var $port = "5432";
                var $dbname = "MyDB";

                // Open initial connection. $params is
                // an associative array holding
                // parameters to the pg_Connect function.
                function init($params)
                {
                        if(isset($parame[host]))
                                $host = $parame[host];
                        else
                                $host = $this->host;

                        if(isset($parame[user]))
                                $user = $parame[user];
                        else
                                $user = $this->user;

                        if(isset($parame[password]))
                                $password = $parame[password];
                        else
                                $password = $this->password;

                        if(isset($parame[port]))
                                $port = $parame[port];
                        else
                                $port = $this->port;

                        if(isset($parame[dbname]))
                                $dbname = $parame[dbname];
                        else
                                $dbname = $this->dbname;

                        $this->conn = pg_Connect ( " host=$host user=$user password=$password port=$port d
                }

                // Send SQL to database connection.
```

```
        // Return recordset object on success.
        // Return 0 on failure.
        function exec($SQL)
        {
                $this->resultset = pg_Exec($this->conn, $SQL);

                if ($this->resultset)
                {
                        $recset = new recordset;
                        $recset->init($this->resultset);
                        return $recset;
                }
                else
                {
                        return 0;
                }
        }

        function valid()
        {
                return $this->resultset;
        }

        // Close connection to database
        function free()
        {
                pg_close($this->conn);
        }
};

/*
** This is a simple recordset class which can be
** traversed using next(), prev(), and current() methods.
** It is initialized from a resultset returned from the
** function "pg_Exec" or can be generated by a call to the
** exec method from the dbObj class given above.
** Below "Tuples" means rows.
*/
class recordset
{
        var $resultset;
        var $index;
        var $numFields;
        var $numTuples;

        function init($newResultset)
        {
                $this->resultset = $newResultset;
                $this->index = 0;
                $this->numFields = pg_NumFields($this->resultset);
```

```
        $this->numTuples = pg_NumRows($this->resultset);
}

// Used in display() below
function valid()
{
        return $this->resultset;
}

// Get a value by row number and either
// column name or column number
function getVal($row, $col)
{
        return pg_Result($this->resultset, $row, $col);
}

// Return an array of field names
function getFields()
{
        for ($i=0; $i < $this->numFields; $i++)
                $retArray[] = pg_FieldName($this->resultset, $i);
        return $retArray;
}

// Get number of columns in resultset
function getNumFields()
{
        return $this->numFields;
}

// Get a tuple (associative array of
// column values) by row number
function getTupleDirect($row)
{
        for ($i=0; $i < $this->numFields; $i++)
        {
                $retArray[pg_FieldName($this->resultset, $i)] =
                        pg_Result($this->resultset, $row, $i);
        }
        return $retArray;
}

// Get an array filled with all values in a column
// (using either column name or column number)
function getColumn($col)
{
        for ($i=0; $i < $this->numTuples; $i++)
                $retArray[] = pg_Result($this->resultset, $i, $col);
        return $retArray;
}
```

```
// Return the number of records in the recordset
function getNumTuples()
{
        return $this->numTuples;
}

// Get tuple pointed to by the current index
function getTuple()
{
        if ($this->index >= 0 && $this->index < $this->numTuples)
                return $this->getTupleDirect($this->index);
        else
                return 0;
}

function valueof($col)
{
        if ($col < $this->numFields)
        {
                return pg_Result($this->resultset, $this->index, $col);
        }
        else
        {
                return "";
        }
}

// Reached last row - end of rows ? Used in display() below
function eof()
{
        return $this->index == $this->numTuples;
}

// Return 1 if index is within bounds of the recordset
function current()
{
        if ($this->index >= 0 && $this->index < $this->numTuples)
                return 1;
        else
                return 0;
}

// Increment index. Used in display() below
function next()
{
        if ($this->index < $this->numTuples)
        {
                $this->index++;
                return 1;
```

```
        }
        else
        {
                return  0;
        }
}

// Decrement  index
function  prev()
{
        if ($this->index  >=  0)
        {
                $this->index--;
                return  1;
        }
        else
        {
                return  0;
        }
}

// Reset  index  to  0 - See also first()
function  reset()
{
        $this->index  =  0;
}

// See also reset(). Used in display() below
function  first()
{
        $this->index  =  0;
}

function last()
{
        $this->index = $this->numTuples -1 ;
}

// Used in display() below
function showheader($col, $fmt = "")
{
        printf("\t< th %s>%s< /th >\n", $fmt,
        is_string($col) ? $col : pg_fieldname($this->resultset, $col));
}

// Used in display() below
function showvalue($col, $fmt =  "", $def =  " ")
{
        $v = $this->valueof($col);
        printf( "\t< td %s>%s< /td>\n", $fmt, $v ==  "" ? $def : $v);
```

```
        }

        function showurl($col, $fmt =  "")
        {
                $v = $this->valueof($col);
                if ( $v !=  "" )
                {
                        printf("\t< td %s> < /td>\n", $fmt);
                }
                else
                {
                        printf( "\t< td %s>< a href=%s>%s< /a>< /td>\n", $fmt, $v, $v);
                }
        }

        function display()
        {
                if (!$this->valid() )
                {
                        return;
                }

                printf( "<table cellspacing=1 cellpadding=1 border=1>\n");

                printf( "<tr>\n");
                for ($c = 0; $c < $this->cols; $c++ )
                {
                        $this->showheader($c);
                }
                printf( "< /tr>\n");

                $this->first();
                while (!$this->eof())
                {
                        printf( "<tr>\n");

                        for ($c = 0; $c < $this->cols; $c++)
                        {
                                $this->showvalue($c);
                        }

                        printf( "< /tr>\n");
                        $this->next();
                }
                printf("< /table\n");
        }

        // Free  memory  allocated  to  recordset.
        function  free()
        {
```

```
                        pg_Freeresult($this->resultset);
                }
        };
}

?>
```

제 12 절 부록 B SQL abstraction Example

제출자 : Gianugo Rabellino `nemorino@opera.it` 이화일을 얻으려면 웹브라우저에서 '다른이름으로 저장' 을 누르고 sqlabst.lib 라고 입력하시오.

```
PX: PHP Code Exchange
<?php

/*
*    SAL - SQL Abstraction Library
*           version 0.01
*/

/*
** Set the variable $dbtype to any of the following
** values: MySQL, mSQL, Postgres, ODBC before including this library
*/
// $dbtype = "MySQL";
// $dbtype = "mSQL";
// $dbtype = "PostgreSQL";
// $dbtype = "ODBC";

// SQL_connect($host, $user, $password, $db)
// returns the connection ID

function SQL_connect($host, $user, $password, $db)
{
        global $dbtype;

        switch ($dbtype)
        {
                case  "MySQL":
                        $conn=mysql_pconnect($host, $user, $password);
                        mysql_select_db($db);
                        return $conn;
                        break;;

                case  "mSQL":
                        $conn=msql_pconnect($host);
                        msql_select_db($db);
                        return $conn;
                        break;;
```

```
                case  "PostgreSQL":
                        $conn=pg_pconnect($host, "5432", "",$db);
                        return $conn;
                        break;;

                case  "ODBC":
                        $conn=odbc_pconnect($db,$user,$password);
                        return $conn;
                        break;;

                default:
                        $conn=mysql_pconnect($host, $user, $password);
                        mysql_select_db($db);
                        return $conn;
                        break;;
        }
}

// SQL_query($host, $user, $password, $db)
// executes an SQL statement, returns a result identifier
function SQL_query($query, $id)
{
        global $dbtype;
        switch ($dbtype)
        {
                case  "MySQL":
                        $res=mysql_query($query, $id);
                        return $res;
                        break;;

                case  "mSQL":
                        $res=msql_query($query, $id);
                        return $res;
                        break;;

                case  "PostgreSQL":
                        $res=pg_exec($id,$query);
                        return $res;
                        break;;

                case  "ODBC":
                        $rid=odbc_prepare($id,$query);
                        $res=odbc_execute($rid);
                        return $res;
                        break;;

                default:
                        $res=mysql_query($query, $id);
                        return $res;
```

```
                        break;;
        }
}

// SQL_num_rows($host, $user, $password, $db)
// given a result identifier, returns the number of affected rows
function SQL_num_rows($res)
{
        global $dbtype;

        switch ($dbtype)
        {
                case  "MySQL":
                        $rows=mysql_num_rows($res);
                        return $rows;
                        break;;

                case  "mSQL":
                        $rows=msql_num_rows($res);
                        return $rows;
                        break;;

                case  "PostgreSQL":
                        $rows=pg_numrows($res);
                        return $rows;
                        break;;

                case  "ODBC":
                        $rows=odbc_num_rows($res);
                        return $rows;
                        break;;

                default:
                        $rows=mysql_num_rows($res);
                        return $rows;
                        break;;
        }
}


// SQL_fetchrow($res,$row)
// given a result identifier, returns an array with the resulting row
// Needs also a row number for compatibility with PostgreSQL
function SQL_fetch_row($res, $nr)
{
        global $dbtype;

        switch ($dbtype)
        {
                case  "MySQL":
```

```
                        $row = array();
                        $row = mysql_fetch_row($res);
                        return $row;
                        break;;

                case  "mSQL":
                        $row = array();
                        $row = msql_fetch_row($res);
                        return $row;
                        break;;

                case  "PostgreSQL":
                        $row = array();
                        $row = pg_fetch_row($res,$nr);
                        return $row;
                        break;;

                case  "ODBC":
                        $row = array();
                        $cols = odbc_fetch_into($res, $nr, &$row);
                        return $row;
                        break;;

                default:
                        $row = array();
                        $row = mysql_fetch_row($res);
                        return $row;
                        break;;
        }
}

// SQL_fetch_array($res,$row)
// given a result identifier, returns an associative array
// with the resulting row using field names as keys.
// Needs also a row number for compatibility with PostgreSQL.
function SQL_fetch_array($res, $nr)
{
        global $dbtype;

        switch ($dbtype)
        {
                case  "MySQL":
                        $row = array();
                        $row = mysql_fetch_array($res);
                        return $row;
                        break;;

                case  "mSQL":
                        $row = array();
                        $row = msql_fetch_array($res);
```

```
                        return $row;
                        break;;

                case  "PostgreSQL":
                        $row = array();
                        $row = pg_fetch_array($res,$nr);
                        return $row;
                        break;;

                /*
                * ODBC doesn’t have a native _fetch_array(), so we have to
                * use a trick. Beware: this might cause HUGE loads!
                */

                case  "ODBC":
                        $row = array();
                        $result = array();
                        $result = odbc_fetch_row($res, $nr);
                        $nf = count($result)+2;  /* Field numbering starts at 1 */
                        for ($count=1; $count < $nf; $count++)
                        {
                                $field_name = odbc_field_name($res, $count);
                                $field_value = odbc_result($res, $field_name);
                                $row[$field_name] = $field_value;
                        }
                        return $row;
                        break;;
        }
}
```

제 13 절 부록 C PostgreSQL large object 예제

제출자 : PHP code exchange `px@sklar.com` 이화일을 얻으려면 웹브라우져에서 ’다른이름으로 저장’을 눌러 pgsql_largeobj.lib 라고 입력하시오.

```
PX: PHP Code Exchange - PostgreSQL large object access

<?
        $database = pg_Connect ( "",  "",  "",  "",  "jacarta");
        pg_exec ($database,  "BEGIN");
        $oid = pg_locreate ($database);
        echo ( "$oid\n");
        $handle = pg_loopen ($database, $oid,  "w");
        echo ( "$handle\n");
        pg_lowrite ($handle,  "foo");
        pg_loclose ($handle);
        pg_exec ($database,  "COMMIT");
        pg_close ($database);
?>
```

제 14 절 부록 D 사용자 인증 예제

이화일을 얻으려면 웹브라우저에서 '다른이름으로 저장'을 누르고 user_pw.lib 라고 입력하시오.

PHP 3 메뉴얼에서 : PHP가 아파치 모듈일때만 작동한다. $PHP_AUTH_USER 와 $PHP_AUTH_PW 를 출력하는 대신, 사용자이름과 비밀번호를 확인만 하기을 원할지도 모른다. 아마도 데이터베이스에 쿼리를 보내거나 adm 화일에서 찾아보면 될것이다.

```
<?php
        if (!$PHP_AUTH_USER)
        {
                Header("WWW-authenticate: basic realm=\"My Realm\"");
                Header("HTTP/1.0 401 Unauthorized");
                echo "Text to send if user hits Cancel button\n";
                exit;
        }
        else
        {
                echo "Hello $PHP_AUTH_USER.<P>";
                echo "You entered $PHP_AUTH_PW as your password.<P>";
        }
?>
```

제 15 절 부록 E 네트워크 관리 예제

이 화일을 얻으려면 웹브라우저에서 network.lib란 이름의 '텍스트(Text)' 형태로 저장하라.

PHP: network adminstrator's best friend from `<http://www.phpWizard.net>`

웹개발자로서 아마 ping, whois, nslookup 등등의 유용한 툴을 사용할것이다. 그러나 고객의 사무실에서 그 툴중 하나를 사용해야되는데 telnet을 사용할수 없다면 어떻게할것인가? PHP 매뉴얼의 '네트워크'에서 function을 찾아보는 것이 대안이다.

소켓 작동 :

가장 중요한 함수는 fsockopen()이다. 이함수를 사용하면 서버의 열려진 어떤 포트로도 접속할수 있고 소켓의 연결을 설정할수 있다. 함수의 문법은 다음과 같다 :

```
        int fsockopen(string hostname, int port, int [errno], string [errstr]);
```

앞의 2개의 변수는 쉽게 알수 있고, 다음의 2개의 변수는 옵션이고, 에러를 처리하기 위해 쓰인다. "errno"과 "errstr"은 참조를 통해서 이루어져야 한다. "참조를 통해 이루어진다"는 말의 의미는 원래의 변수가 변한다는 말이다. 보통, 함수가 실행되더라도 변수의 내용은 변하지 않는다.

그래서, 웹서버에 연결하기 위해 이함수를 사용하고 헤더를 프린트하라 :

```
function get_headers($host, $path = "/")
{
        $fp = fsockopen ("$host", 80, &$errnr, &$errstr) or die("$errno: $errstr");
        fputs($fp,"GET $path HTTP/1.0\n\n");
        while (!$end)
        {
```

```
                $line = fgets($fp, 2048);
                if (trim($line) == "")
                        $end = true;
                else
                        echo $line;
        }
        fclose($fp);
}
```

이 예제에서는 fsockopen()을 호출해서 얻은 파일포인터와 파일처리에 관한응용을 알아볼것이다.(fred, fwrite 등등) HTTP/1.0 클라이언트에서만 구현된다는것을 명심하라. - name-based 가상호스트에서는 동작하지 않는다.

지시:

일반적으로 다른 포트로 연결 할수 있다. PHP로 작은 핑거 클라이언트를 만든것은 쉬게 할수 있다. 핑거 데몬에 쿼리(질문)하기 위해 아래에 있는 예제를 수정하라.

```
function finger ($host, $user)
{
        $fp = fsockopen($host, 79, &$errno, &$errstr) or die("$errno: $errstr");
        fputs($fp, "$user\n");
        while (!feof($fp))
                echo fgets($fp, 128);
        fclose($fp);
}
```

Whois: 후이스(whois) 서버에 질문하는것도 같은 개념이다:

```
// domain is like "phpwizard.net"
function whois($domain, $server="whois.internic.net")
{
        $fp = fsockopen ($server, 43, &$errnr, &$errstr) or die("$errno: $errstr");
        fputs($fp, "$domain\n");
        while (!feof($fp))
                echo fgets($fp, 2048);
        fclose($fp);
}
```

블럭킹, 넌블록킹 제어 : 그러나 이함수를 사용하는데 문제가 있다. 다음과 같을때만 잘작동한다.

1. 호출시간이 작은 연결일때와
2. 연결하고 있는 서버가 살아있고 작동할때. 그렇지 않으면, 당신의 스크립트는 종료될때까지 작동한다. 그렇게 되는 이유는 소켓연결은 블럭킹되며 종료가 되지않기 때문이다. set_socket_blocking()이란 함수는 다음과 같이 동작한다: 함수는 모든동작을 소켓(첫번째 인수:소켓 포인터)에 블럭킹(두번째 인수:true)나 폴스(false)(세번째 인수:false)로 설정한다. 넌블럭킹제어를 사용하면, 핑거(finger)함수는 다음과 같을것이다:(역주:무슨말인지 번역이 이상하네요)

```
        $fp = fsockopen($host, 79, &$errno, &$errstr) or die("$errno: [ ] $errstr");
        set_socket_blocking($fp, 0);
        fputs($fp, "$user\n");

        $stop = time() + $timeout;
        while (!feof($fp) && time() < $stop )
                echo fgets($fp, 128);
        fclose($fp);
```

넌블러킹 소켓콜(socket calls)을 사용하기 위한 3개의 함수의 수정은 당신의 연습을 위해 남겨둔다.

제 16 절 부록 F PostgreSQL Database Wrapper 예제

제출자 : Joe Thong `darkjoe@softhome.net` 사이트 URL: `<http://phpdb.linuxbox.com>`

설명: 여러가지 데이터베이스 서버에서 결과 조작을 위한 강력한 PHP 데이타베이스 랩퍼. 데이터베이스의 결과가 phpDB에 의해 자동으로 나오게 된다.

이화일을 얻으려면 웹브라우저에서 '다른이름으로 저장'을 누르고 phpDB-postgresql.lib 라고 입력하시오.

```
<?php
/*
Name: phpDB PostgreSQL module
Version: 1.02bR6
Description: A PHP database wrapper for various database
        servers with a powerful recordset for result data
        manipulation.  Database results are flushed
        automatically by phpDB.
*/

/* define this module, to prevent double class declaration.     */
if (!defined("_PHPDB_ABSTRACT_LAYER")) {
        define("_PHPDB_ABSTRACT_LAYER", 1 );
}
else
        return;

//---------------------------------------
         Class Name: phpDB
//---------------------------------------
class phpDB
{
        /*      public variables        */
        var $version = '1.02bR6'; // Version number of phpDB
        // This variable keeps what database type is going to
        // be used.  Current supported database server are
        // MySQL, MSQL, SQL Server, and Sybase
        var $databaseType = '';
        // Specifies which database is going to be used
        var $databaseName = '';
```

```
// The hostname of the database server, port
// number is optional.  e.g: "db.devNation.com"
var $hostname = '';
var $username = ''; // used to connect to the database server
var $password = ''; // Password for the username

// Private variables ------ starts with underscore
// An array of executed querys. For results cleanup purposes.
var $_queryIDList = array();
// The returned link identifier whenever a
// successful database connection is made
var $_connectionID      = -1;
// A variable which was used to keep the returned
// last error message.  The value will then returned
// by the errorMsg() function
var $_errorMsg = '';

// This variable keeps the last created result
// link identifier
var $_queryID = -1;
// A boolean variable to state whether its a persistent
// connection or normal connection
var     $_isPersistentConnection = false;
// Holds the newly created result object,
// returned via the execute() method
var $_tempResultObj = '';

// A constructor function for the phpDB object.
// When initializing, specify the dbType i.e: "mysql",
// "msql", "postgresql", "mssql", and "sybase"
function phpDB($dbType = "postgresql")
{
        switch ($dbType) {
                case "mysql":
                case "msql":
                case "postgresql":
                case "mssql":
                case "sybase":
                case "informix":
                        $this->databaseType = $dbType;
                        break;
                default:
                        return false;
        }
}

// Returns: A positive link identifier on success, or
// false on error.      Connect to the server with the provided
// arguments. The connection to the server will be closed
// when the script terminates, unless close() function is
```

```
// called beforehand
function connect($argHostname = "", $argUsername = "",
        $argPassword = "", $argDatabaseName = "")
{
        $connString = "";
        $hostPieces = array();
        /*      Must specify the database argument      */
        if (!$argDatabaseName) {
                return false;
        }
        if ($argHostname != "") {
                $this->hostname = $argHostname;
        }
        if ($argUsername != "") {
                $this->username = $argUsername;
        }
        if ($argPassword != "") {
                $this->password = $argPassword;
        }
        if ($argDatabaseName != "") {
                $this->databaseName = $argDatabaseName;
        }
        $hostPieces = split(":", $this->hostname);
        if ($hostPieces[0]) {
                $connString .= "host=$hostPieces[0]";
                if (isset($hostPieces[1])) {
                        $connString .= " port=$hostPieces[1]";
                }
        }
    if ($this->username) {
                $connString .= " user=$this->username";
        }
    if ($this->password) {
                $connString .= " password=$this->password";
        }
        $connString .= " dbname=$this->databaseName";

        $this->_connectionID = @pg_Connect($connString);
        return $this->_connectionID;
}

// Returns: A positive link identifier on success, or
// false on error.  Connect to the server with the
// provided arguments. The connection to the server will
// not be closed when the script terminates. Instead it
// will be kept for later future use
function pconnect($argHostname = "", $argUsername = "",
        $argPassword = "", $argDatabaseName = "")
{
        $connString = "";
```

```
        $hostPieces = array();
        /*      Must specify the database argument      */
        if (!$argDatabaseName) {
                return false;
        }
        if ($argHostname != "") {
                $this->hostname = $argHostname;
        }
        if ($argUsername != "") {
                $this->username = $argUsername;
        }
        if ($argPassword != "") {
                $this->password = $argPassword;
        }
        if ($argDatabaseName != "") {
                $this->databaseName = $argDatabaseName;
        }
        $hostPieces = split(":", $this->hostname);
        if ($hostPieces[0]) {
                $connString .= "host=$hostPieces[0]";
                if (isset($hostPieces[1])) {
                        $connString .= " port=$hostPieces[1]";
                }
        }
    if ($this->username) {
                $connString .= " user=$this->username";
        }
    if ($this->password) {
                $connString .= " password=$this->password";
        }
        $connString .= " dbname=$this->databaseName";

        $this->_connectionID = @pg_pConnect($connString);
        if ($this->_connectionID) {
                $this->_isPersistentConnection = true;
        }
        return $this->_connectionID;
}

// Returns: true on success, false on error Select
// the database name to be used PostgreSQL
// Note:        function Not available
function selectDB($dbName) {
                return false;
}

// Returns: the Recordset object disregard success
// or failure Send the sql statement to the database server
function execute($sql = "") {
        // Instantiate an object without considering whether
```

```
                // the query return any results or not
                $this->_queryID = @pg_Exec($this->_connectionID, $sql);
                $this->_tempResultObj = new Recordset($this->_queryID);
                $this->_insertQuery($this->_queryID);
                return $this->_tempResultObj;
        }

        // Returns: the last error message from previous
        // database operation
        function errorMsg() {
                $this->_errorMsg = @pg_errormessage($this->_connectionID);
            return $this->_errorMsg;
        }

        // Returns: true on success, false on failure
        // Close the database connection
        function close() {
                if ($this->_queryIDList && sizeof($this->_queryIDList > 0)) {
                        while(list($_key, $_resultID) = each($this->_queryIDList)) {
                                @pg_freeresult($_resultID);
                        }
                }
                // If its not a persistent connection, then
                // only the connection needs to be closed
                if ($this->_isPersistentConnection != true) {
                        return @pg_close($this->_connectionID);
                }
                else {
                        return true;
                }
        }

        // A PRIVATE function used by the constructor function
        // of the query object.  insert the successful returned
        // query id to the query id list.  Used for later results
        // cleanup.  A private function that's never meant to
        // be used directly
        function _insertQuery($query_id) {
                $this->_queryIDList[] = $query_id;
        }
}

//------------------------------
         Class Name: Recordset
//------------------------------
class Recordset
{
        /*      public variables        */
        var $fields;
        // indicates that the current record position is before
```

```
        // the first record in a Recordset object
        var $BOF = null;

        // indicates that the current record position is after
        // the last record in a Recordset object
        var $EOF = null;

        /*      private variables       */
        var $_numOfRows = -1; // NEVER change the value! READ-ONLY!
        var $_numOfFields = -1; // NEVER change the value! READ-ONLY!
        // Holds anything that was returned from the database specific functions
        var $_tempResult = '';
        // This variable keeps the result link identifier
        var $_queryID = -1;
        // This variable keeps the current row in the Recordset
        var $_currentRow = -1;

        // Returns: query id on success and false if
        // failed Constructor function
        function Recordset($queryID)
        {
                $this->_queryID = $queryID;
                if ($queryID) {
                        $this->_numOfRows = @pg_numrows($this->_queryID);
                        /*      pg_numrows() returns -1 on error        */
                        if ($this->_numOfRows == -1) {
                                $this->_numOfRows = 0;
                        }
                        $this->_numOfFields = @pg_numfields($this->_queryID);
                        /*      pg_numfields() returns -1 on error      */
                        if ($this->_numOfFields == -1) {
                                $this->_numOfFields = 0;
                        }
                }
                else {
                        $this->_numOfRows = 0;
                        $this->_numOfFields = 0;
                }
                /*      If result set contains rows     */
                if ($this->_numOfRows > 0 && $this->_currentRow == -1) {
                        $this->_currentRow = 0;
                        $this->fields = @pg_fetch_array($this->_queryID, $this->_currentRow);
                        $this->EOF = false;
                        $this->BOF = false;
                }
                return $this->_queryID;
        }

        // Returns: true if successful, false if fail Set the Recordset
        // pointer to a specified field offset. If the next call to
```

```
// fetchField() won't include a field offset, this field would
// be returned.  PostgreSQL Note:       function Not available
function fieldSeek($fieldOffset = -1) {
        $this->_tempResult = false;
        return $this->_tempResult;
}

// Returns: an object containing field information.  Get column
// information in the Recordset object. fetchField() can be used
// in order to obtain information about fields in a certain query
// result. If the field offset isn't specified, the next field
// that wasn't yet retrieved by fetchField() is retrieved.
// PostgreSQL Note:     function Not available
function fetchField($fieldOffset = -1) {
        $this->_tempResult = false;
        return $this->_tempResult;
}

// Returns: true if there still rows available, or false if there
// are no more rows.  Moves to the next row in a specified Recordset
// object and makes that record the current row and the data
// corresponding to the row will be retrieved into the fields
// collection.  Note: Unlike the moveRow() method, when _currentRow
// is getNumOfRows() - 1, EOF will immediately be true.  If row number
// is not provided, the function will point to the
// first row automatically
function nextRow() {
        if ($this->getNumOfRows() > 0) {
                $this->fields = array();
                $this->_currentRow++;
                $this->fields = @pg_fetch_array($this->_queryID, $this->_currentRow);
                /*      This is not working.  True all the time */
                if ($this->fields) {
                        $this->_checkAndChangeEOF($this->_currentRow - 1);
                        return true;
                }
        }
        $this->EOF = true;
        return false;
}

// Returns: true on success, false on failure moveRow() moves
// the internal row pointer of the Recordset object to point
// to the specified row number and the data corresponding to
// the row will be retrieved into the fields collection.  If
// row number is not provided, the function will point to
// the first row automatically
function moveRow($rowNumber = 0) {
        if ($rowNumber == 0) {
                return $this->firstRow();
```

```
                }
                else if ($rowNumber == ($this->getNumOfRows() - 1)) {
                        return $this->lastRow();
                }
                if ($this->getNumOfRows() > 0 && $rowNumber < $this->getNumOfRows()) {
                        $this->fields = null;
                        $this->_currentRow = $rowNumber;
                        $this->fields = @pg_fetch_array($this->_queryID, $this->_currentRow);
                        /*      This is not working.  True all the time */
                        if ($this->fields) {
                                // No need to call _checkAndChangeEOF() because
                                // the possibility of moving to the last row
                                // has been handled by the above code
                                $this->EOF = false;
                                return true;
                        }
                }
                $this->EOF = true;
                return false;
        }

        // Returns: true on success, false on failure firstRow()
        // moves the internal row pointer of the Recordset object
        // to the first row and the data corresponding to the row
        // will be retrieved into the fields collection
        function firstRow() {
                if ($this->getNumOfRows() > 0) {
                        $this->fields = array();
                        $this->_currentRow = 0;
                        $this->fields = @pg_fetch_array($this->_queryID, $this->_currentRow);
                        $this->EOF = true;
                        /*      This is not working.  True all the time */
                        if ($this->fields) {
                                return true;
                        }
                }
                $this->EOF = true;
                return false;
        }

        // Returns: true on success, false on failure lastRow()
        // moves the internal row pointer of the Recordset object
        // to the last row and the data corresponding to the row
        // will be retrieved into the fields collection
        function lastRow() {
                if ($this->getNumOfRows() > 0) {
                        $this->fields = array();
                        $num_of_rows = $this->getNumOfRows();
                        /*      $num_of_rows decemented at above        */
                        $this->_currentRow = --$num_of_rows;
```

```
                        $this->fields = @pg_fetch_array($this->_queryID, $this->_currentRow);
                        /*      This is not working.  True all the time */
                        if ($this->fields) {
                                /*      Special case for making EOF false.      */
                                $this->EOF = false;
                                return true;
                        }
                }
                $this->EOF = true;
                return false;
        }

        // close() only needs to be called if you are worried about
        // using too much memory while your script is running. All
        // associated result memory for the specified result identifier
        // will automatically be freed
        function close() {
                $this->_tempResult = @pg_freeresult($this->_queryID);
                return $this->_tempResult;
        }

        // Returns: the number of rows in a result set.
        // Get number of rows in result
        function getNumOfRows() {
                return $this->_numOfRows;
        }

        // Returns: the number of fields in a result set.
        // Get number of fields in result
        function getNumOfFields() {
                return $this->_numOfFields;
        }

        /*      Check and change the status of EOF.     */
        function _checkAndChangeEOF($currentRow) {
                if ($currentRow >= ($this->_numOfRows - 1)) {
                        $this->EOF = true;
                }
                else {
                        $this->EOF = false;
                }
        }
}
?>
```

제 17 절 부록 G Microsoft SQL Server DB Wrapper 예제

제출자 : Joe Thong `darkjoe@softhome.net` Site URL: `<http://phpdb.linuxbox.com>`

설명: 여러가지 데이터베이스 서버에서 결과 조작을 위한 강력한 PHP 데이타베이스 랩퍼. 데이터베이스

의 결과가 phpDB에 의해 자동으로 나오게 된다.

이화일을 얻으려면 웹브라우저에서 '다른이름으로 저장'을 누르고 phpDB-msql.lib 라고 입력하시오.

```
<?php
/*
Name: phpDB Microsoft SQL Server module
Version: 1.02bR6
Description: A PHP database wrapper for various
        database servers with a powerful
        Recordset for result data manipulation.  Database
        results are flushed automatically by phpDB.
*/
// Define this module, to prevent double class declaration
if (!defined("_PHPDB_ABSTRACT_LAYER"))
{
        define("_PHPDB_ABSTRACT_LAYER", 1 );
}
else
        return;

//------------------------------
         Class Name: phpDB
//------------------------------
class phpDB
{
        // public variables
        var $version = '1.02bR6'; // Version number of phpDB

        // This variable keeps what database type is going
        // to be used.  Current supported database server
        // are MySQL, MSQL, SQL Server, PostgreSQL and Sybase
        var $databaseType = '';
        var $databaseName = ''; // Specifies which database is going to be used

        // The hostname of the database server, port
        // number is optional.  e.g: "db.devNation.com"
        var $hostname = '';

        var $username = ''; // to connect to the database server
        var $password = '';     // Password for the username

        // Private variables ----- starts with underscore

        // An array of executed querys. For results cleanup purposes
        var $_queryIDList = array();

        // The returned link identifier whenever a
        // successful database connection is made
        var $_connectionID      = -1;
```

```
        // A variable which was used to keep the returned last
        // error message.  The value will then returned
        // by the errorMsg() function
        var $_errorMsg = '';
        // This variable keeps the last created result link identifier
        var $_queryID = -1;

        // A boolean variable to state whether its a
        // persistent connection or normal connection
        var     $_isPersistentConnection = false;

        // Holds the newly created result object, returned
        // via the execute() method
        var $_tempResultObj = '';

        //      A constructor function for the phpDB object.
        // When initializing, specify the dbType i.e: "mysql",
        // "msql", "postgresql", "mssql", and "sybase"
        function phpDB($dbType = "mssql")
        {
                switch ($dbType)
                {
                        case "mysql":
                        case "msql":
                        case "postgresql":
                        case "mssql":
                        case "sybase":
                        case "informix":
                                $this->databaseType = $dbType;
                                break;
                        default:
                                return false;
                }
        }

        // Returns: A positive link identifier on success,
        // or false on error.   Connect to the server with
        // the provided arguments. The connection to the server
        // will be closed when the script terminates, unless
        // close() function is called beforehand.
        function connect($argHostname = "", $argUsername = "",
                $argPassword = "", $argDatabaseName = "")
        {
                $boolDBSelected;
                if ($argHostname != "") {
                        $this->hostname = $argHostname;
                }
                if ($argUsername != "") {
                        $this->username = $argUsername;
                }
```

```
                if ($argPassword != "") {
                        $this->password = $argPassword;
                }
                if ($argDatabaseName != "") {
                        $this->databaseName = $argDatabaseName;
                }

                $this->_connectionID = @mssql_connect($this->hostname, $this->username, $this->password);

                if ($this->databaseName && $this->_connectionID) {
                        $boolDBSelected = @mssql_select_db($this->databaseName);
                        if(!$boolDBSelected) {  /*      If DB selection fails   */
                                @mssql_close($this->_connectionID);     /*      Close the current connection    *
                                return false;
                        }
                }
                return $this->_connectionID;
        }

        // Returns: A positive link identifier on success, or
        // false on error Connect to the server with the provided
        // arguments. The connection to the server will not be closed
        // when the script terminates. Instead it will be kept for
        // later future use
        function pconnect($argHostname = "", $argUsername = "",
                $argPassword = "", $argDatabaseName = "")
        {
                $boolDBSelected;
                if ($argHostname != "") {
                        $this->hostname = $argHostname;
                }
                if ($argUsername != "") {
                        $this->username = $argUsername;
                }
                if ($argPassword != "") {
                        $this->password = $argPassword;
                }
                if ($argDatabaseName != "") {
                        $this->databaseName = $argDatabaseName;
                }

                $this->_connectionID = @mssql_pconnect($this->hostname, $this->username, $this->password);
                if ($this->_connectionID) {
                        $this->_isPersistentConnection = true;
                }

                if ($this->databaseName && $this->_connectionID) {
                        $boolDBSelected = @mssql_select_db($this->databaseName);
                        if(!$boolDBSelected) {  /*      if DB selection fails   */
                                return false;   /*      Persistent connection can't be closed   */
```

```
                        }
                }
                return $this->_connectionID;
        }

        //      Returns: true on success, false on error Select the
        // database name to be used
        function selectDB($dbName)
        {
                $this->databaseName = $dbName;
                if ($this->_connectionID) {
                        return @mssql_select_db($dbName);
                }
                else {
                        /*      No database selected    */
                        return false;
                }
        }

        // Returns: the Recordset object disregard success or
        // failure Send the sql statement to the database server
        function execute($sql = "")
        {
                $this->_queryID = @mssql_query($sql, $this->_connectionID);
                // Instantiate an object without considering whether
                // the query return any results or not
                $this->_tempResultObj = new Recordset($this->_queryID);
                $this->_insertQuery($this->_queryID);
                return $this->_tempResultObj;
        }

        // Returns: the last error message from previous database
        // operation Note: This function is NOT available for
        // Microsoft SQL Server
        function errorMsg()
        {
                $this->_errorMsg = "errorMsg() is not available for Microsoft SQL Server";
            return $this->_errorMsg;
        }

        /*      Returns: true on success, false on failure
                Close the database connection.  */
        function close() {
                if ($this->_queryIDList && sizeof($this->_queryIDList > 0)) {
                        while(list($_key, $_resultID) = each($this->_queryIDList)) {
                                @mssql_free_result($_resultID);
                        }
                }
                // If its not a persistent connection, then
                // only the connection needs to be closed
```

```
                if ($this->_isPersistentConnection != true) {
                        return @mssql_close($this->_connectionID);
                }
                else {
                        return true;
                }
        }

        // A PRIVATE function used by the constructor function of
        // the query object.  insert the successful returned
        // query id to the query id list.  Used for later results
        // cleanup.  A private function that's never meant to be
        // used directly
        function _insertQuery($query_id) {
                $this->_queryIDList[] = $query_id;
        }
}

//-----------------------------------------
         Class Name: Recordset
//-----------------------------------------
class Recordset
{
        /*      public variables        */
        var $fields;
        // indicates that the current record position is
        // before the first record in a Recordset object
        var $BOF = null;
        // indicates that the current record position is
        // after the last record in a Recordset object
        var $EOF = null;

        // Private variables
        var $_numOfRows = -1; // NEVER change the value!  READ-ONLY!
        var $_numOfFields = -1; // NEVER change the value!  READ-ONLY!

        // Holds anything that was returned from the
        // database specific functions
        var $_tempResult = '';
        // This variable keeps the result link identifier
        var $_queryID = -1;
        // This variable keeps the current row in the Recordset
        var $_currentRow = -1;

        // Returns: query id on success and false if
        // failed Constructor function
        function Recordset($queryID)
        {
                $this->_queryID = $queryID;
                if ($queryID) {
```

```
                $this->_numOfRows = @mssql_num_rows($this->_queryID);
                $this->_numOfFields = @mssql_num_fields($this->_queryID);
        }
        else {
                $this->_numOfRows = 0;
                $this->_numOfFields = 0;
        }
        // If result set contains rows
        if ($this->_numOfRows > 0 && $this->_currentRow == -1) {
                $this->_currentRow = 0;
                $this->fields = @mssql_fetch_array($this->_queryID);
                $this->EOF = false;
                $this->BOF = false;
        }
        return $this->_queryID;
}

// Returns: true if successful, false if fail Set
// the Recordset pointer to a specified field offset.
// If the next call to fetchField() won't include a
// field offset, this field would be returned
function fieldSeek($fieldOffset = -1) {
        $this->_tempResult = @mssql_field_seek($this->_queryID, $fieldOffset);
        return $this->_tempResult;
}

// Returns: an object containing field information.
// Get column information in the Recordset object.
// fetchField() can be used in order to obtain information
// about fields in a certain query result. If the field
// offset isn't specified, the next field that wasn't yet
// retrieved by fetchField() is retrieved
function fetchField($fieldOffset = -1) {
        if ($fieldOffset != -1) {
                $this->_tempResult = @mssql_fetch_field($this->_queryID, $fieldOffset);
        }
        // The $fieldOffset argument is not provided thus its -1
        else if ($fieldOffset == -1) {
                $this->_tempResult = @mssql_fetch_field($this->_queryID);
        }
        return $this->_tempResult;
}

// Returns: true if there still rows available, or false
// if there are no more rows.  Moves to the next row in a
// specified Recordset object and makes that record the current
// row and the data corresponding to the row will be retrieved
// into the fields collection.  Note: Unlike the moveRow() method,
// when _currentRow is getNumOfRows() - 1, EOF will immediately be
// true.  If row number is not provided, the function will point
```

```
// to the first row automatically
function nextRow()
{
        if ($this->getNumOfRows() > 0) {
                $this->fields = array();
                $this->_currentRow++;
                $this->fields = @mssql_fetch_array($this->_queryID);
                // This is not working.  True all the time
                if ($this->fields) {
                        $this->_checkAndChangeEOF($this->_currentRow - 1);
                        return true;
                }
        }
        $this->EOF = true;
        return false;
}

// Returns: true on success, false on failure moveRow()
// moves the internal row pointer of the Recordset object
// to point to the specified row number and the data
// corresponding to the row will be retrieved into the fields
// collection.  If row number is not provided, the function will
// point to the first row automatically
function moveRow($rowNumber = 0)
{
        if ($rowNumber == 0) {
                return $this->firstRow();
        }
        else if ($rowNumber == ($this->getNumOfRows() - 1)) {
                return $this->lastRow();
        }
        if ($this->getNumOfRows() > 0 && $rowNumber < $this->getNumOfRows()) {
                $this->fields = null;
                $this->_currentRow = $rowNumber;
                if(@mssql_data_seek($this->_queryID, $this->_currentRow)) {
                        $this->fields = @mssql_fetch_array($this->_queryID);
                        /*      This is not working.  True all the time */
                        if ($this->fields) {
                                // No need to call _checkAndChangeEOF() because
                                // the possibility of moving to the last row has
                                // been handled by the above code
                                $this->EOF = false;
                                return true;
                        }
                }
        }
        $this->EOF = true;
        return false;
}
```

```
// Returns: true on success, false on failure firstRow() moves
// the internal row pointer of the Recordset object to the first
// row and the data corresponding to the row will be retrieved
// into the fields collection
function firstRow()
{
        if ($this->getNumOfRows() > 0) {
                $this->fields = array();
                $this->_currentRow = 0;
                if (@mssql_data_seek($this->_queryID, $this->_currentRow)) {
                        $this->fields = @mssql_fetch_array($this->_queryID);
                        $this->EOF = false;
                        /*      This is not working.  True all the time */
                        if ($this->fields) {
                                return true;
                        }
                }
        }
        $this->EOF = true;
        return false;
}

// Returns: true on success, false on failure lastRow() moves
// the internal row pointer of the Recordset object to the last
// row and the data corresponding to the row will be retrieved
// into the fields collection
function lastRow()
{
        if ($this->getNumOfRows() > 0) {
                $this->fields = array();
                $num_of_rows = $this->getNumOfRows();
                $this->_tempResult = @mssql_data_seek($this->_queryID, --$num_of_rows);
                if ($this->_tempResult) {
                        /*      $num_of_rows decemented at above        */
                        $this->_currentRow = $num_of_rows;
                        $this->fields = @mssql_fetch_array($this->_queryID);
                        /*      This is not working.  True all the time */
                        if ($this->fields) {
                                /*      Special case for making EOF false.      */
                                $this->EOF = false;
                                return true;
                        }
                }
        }
        $this->EOF = true;
        return false;
}

// close() only needs to be called if you are worried about using
// too much memory while your script is running. All associated
```

```
        // result memory for the specified result identifier will
        // automatically be freed
        function close() {
                $this->_tempResult = @mssql_free_result($this->_queryID);
                return $this->_tempResult;
        }

        // Returns: the number of rows in a result set.  Get
        // number of rows in result
        function getNumOfRows() {
                return $this->_numOfRows;
        }

        /*      Returns: the number of fields in a result set.
        Get number of fields in result. */
        function getNumOfFields() {
                return $this->_numOfFields;
        }

        /*      Check and change the status of EOF.     */
        function _checkAndChangeEOF($currentRow) {
                if ($currentRow >= ($this->_numOfRows - 1)) {
                        $this->EOF = true;
                }
                else {
                        $this->EOF = false;
                }
        }
}
?>
```

제 18 절 부록 H Sybase SQL Server DB Wrapper 예제

제출자 : Joe Thong `darkjoe@softhome.net` Site URL: `<http://phpdb.linuxbox.com>`

설명: 여러가지 데이터베이스 서버에서 결과 조작을 위한 강력한 PHP 데이타베이스 랩퍼. 데이터베이스의 결과가 phpDB에 의해 자동으로 나오게 된다.

이화일을 얻으려면 웹브라우저에서 '다른이름으로 저장'을 누르고 phpDB-sybase.lib 라고 입력하시오.

```
<?php
/*
Name: phpDB Sybase module
Version: 1.02bR6
Description: A PHP database wrapper for various database
        servers with a powerful Recordset for result data
        manipulation.  Database results are flushed
        automatically by phpDB.
*/
```

```
// Define this module, to prevent double class declaration
if (!defined("_PHPDB_ABSTRACT_LAYER")) {
        define("_PHPDB_ABSTRACT_LAYER", 1 );
}
else
        return;

//-------------------------------
         Class Name: phpDB
//-------------------------------
class phpDB
{
        /*      public variables        */
        var $version = '1.02bR6'; // Version number of phpDB
        // This variable keeps what database type is going
        // to be used.  Current supported database server
        // are MySQL, MSQL, SQL Server, and Sybase
        var $databaseType = '';
        // Specifies which database is going to be used
        var $databaseName = '';
        // The hostname of the database server, port number
        // is optional.  e.g: "db.devNation.com"
        var $hostname = '';
        var $username = '';     // to connect to the database server
        var $password = ''; // Password for the username

        // Private variables --- starts with underscore
        // An array of executed querys.  For results
        // cleanup purposes
        var $_queryIDList       = array();
        // The returned link identifier whenever a successful
        // database connection is made
        var $_connectionID      = -1;
        // A variable which was used to keep the returned last
        // error message.  The value will then returned by
        // the errorMsg() function
        var $_errorMsg = '';
        // This variable keeps the last created result
        // link identifier
        var $_queryID = -1;
        // A boolean variable to state whether its a
        // persistent connection or normal connection
        var     $_isPersistentConnection = false;
        // Holds the newly created result object, returned
        // via the execute() method
        var $_tempResultObj = '';

        // A constructor function for the phpDB object.  When
        // initializing, specify the dbType i.e: "mysql",
        // "msql", "postgresql", "mssql", and "sybase"
```

```
function phpDB($dbType = "sybase")
{
        switch ($dbType) {
                case "mysql":
                case "msql":
                case "postgresql":
                case "mssql":
                case "sybase":
                case "informix":
                        $this->databaseType = $dbType;
                        break;
                default:
                        return false;
        }
}

// Returns: A positive link identifier on success, or
// false on error.      Connect to the server with the
// provided arguments. The connection to the server will be
// closed when the script terminates, unless close()
// function is called beforehand
function connect($argHostname = "", $argUsername = "",
        $argPassword = "", $argDatabaseName = "")
{
        $boolDBSelected;
        if ($argHostname != "") {
                $this->hostname = $argHostname;
        }
        if ($argUsername != "") {
                $this->username = $argUsername;
        }
        if ($argPassword != "") {
                $this->password = $argPassword;
        }
        if ($argDatabaseName != "") {
                $this->databaseName = $argDatabaseName;
        }

        $this->_connectionID = @sybase_connect($this->hostname, $this->username, $this->password);

        if ($this->databaseName && $this->_connectionID) {
                $boolDBSelected = @sybase_select_db($this->databaseName);
                /*      If DB selection fails   */
                if(!$boolDBSelected) {
                        /*      Close the current connection    */
                        @sybase_close($this->_connectionID);
                        return false;
                }
        }
        return $this->_connectionID;
```

```
        }

        // Returns: A positive link identifier on success, or false
        // on error.  Connect to the server with the provided
        // arguments. The connection to the server will not be closed
        // when the script terminates. Instead it will be kept for later future use
        function pconnect($argHostname = "", $argUsername = "",
                $argPassword = "", $argDatabaseName = "")
        {
                $boolDBSelected;
                if ($argHostname != "") {
                        $this->hostname = $argHostname;
                }
                if ($argUsername != "") {
                        $this->username = $argUsername;
                }
                if ($argPassword != "") {
                        $this->password = $argPassword;
                }
                if ($argDatabaseName != "") {
                        $this->databaseName = $argDatabaseName;
                }

                $this->_connectionID = @sybase_pconnect($this->hostname, $this->username, $this->password);
                if ($this->_connectionID) {
                        $this->_isPersistentConnection = true;
                }

                if ($this->databaseName && $this->_connectionID) {
                        $boolDBSelected = @sybase_select_db($this->databaseName);
                        /*      if DB selection fails   */
                        if(!$boolDBSelected) {
                                /*      Persistent connection can't be closed   */
                                return false;
                        }
                }
                return $this->_connectionID;
        }

        /*      Returns: true on success, false on error
                Select the database name to be used     */
        function selectDB($dbName) {
                $this->databaseName = $dbName;
                if ($this->_connectionID) {
                        return @sybase_select_db($dbName);
                }
                else {
                        /*      No database selected    */
                        return false;
                }
```

```
        }

        /*      Returns: the Recordset object disregard success or failure
                Send the sql statement to the database server.  */
        function execute($sql = "") {
                $this->_queryID = @sybase_query($sql, $this->_connectionID);
                // Instantiate an object without considering whether
                // the query return any results or not
                $this->_tempResultObj = new Recordset($this->_queryID);
                $this->_insertQuery($this->_queryID);
                return $this->_tempResultObj;
        }

        /*      Returns: the last error message from previous database operation
                Note: This function is NOT available for Sybase.        */

        function errorMsg() {
                $this->_errorMsg = "errorMsg() is not available for Sybase";
            return $this->_errorMsg;
        }

        /*      Returns: true on success, false on failure
                Close the database connection.  */

        function close() {
                if ($this->_queryIDList && sizeof($this->_queryIDList > 0)) {
                        while(list($_key, $_resultID) = each($this->_queryIDList)) {
                                @sybase_free_result($_resultID);
                        }
                }
                // If its not a persistent connection, then
                // only the connection needs to be closed
                if ($this->_isPersistentConnection != true) {
                        return @sybase_close($this->_connectionID);
                }
                else {
                        return true;
                }
        }

        // A PRIVATE function used by the constructor function
        // of the query object.  insert the successful returned
        // query id to the query id list.  Used for later results
        // cleanup.  A private function that's never meant
        // to be used directly
        function _insertQuery($query_id) {
                $this->_queryIDList[] = $query_id;
        }
}
```

```
//---------------------------------
         Class Name: Recordset
//---------------------------------
class Recordset
{
        /*      public variables        */
        var $fields;
        // indicates that the current record position is
        // before the first record in a Recordset object
        var $BOF = null;
        // indicates that the current record position
        // is after the last record in a Recordset object
        var $EOF = null;

        // Private variables - starts with underscore
        var $_numOfRows = -1; // NEVER change the value! READ-ONLY!
        var $_numOfFields = -1; // NEVER change the value! READ-ONLY!
        // Holds anything that was returned from
        // the database specific functions
        var $_tempResult = '';
        // This variable keeps the result link identifier
        var $_queryID = -1;
        // This variable keeps the current row in the Recordset
        var $_currentRow = -1;

        // Returns: query id on success and false if
        // failed Constructor function
        function Recordset($queryID) {
                $this->_queryID = $queryID;
                if ($queryID) {
                        $this->_numOfRows = @sybase_num_rows($this->_queryID);
                        $this->_numOfFields = @sybase_num_fields($this->_queryID);
                }
                else {
                        $this->_numOfRows = 0;
                        $this->_numOfFields = 0;
                }
                /*      If result set contains rows     */
                if ($this->_numOfRows > 0 && $this->_currentRow == -1) {
                        $this->_currentRow = 0;
                        $this->fields = @sybase_fetch_array($this->_queryID);
                        $this->EOF = false;
                        $this->BOF = false;
                }
                return $this->_queryID;
        }

        // Returns: true if successful, false if fail Set
        // the Recordset pointer to a specified field offset.
        // If the next call to fetchField() won't include a
```

```
        // field offset, this field would be returned
        function fieldSeek($fieldOffset = -1) {
                $this->_tempResult = @sybase_field_seek($this->_queryID, $fieldOffset);
                return $this->_tempResult;
        }

        // Returns: an object containing field information.
        // Get column information in the Recordset object.
        // fetchField() can be used in order to obtain information
        // about fields in a certain query result. If the field
        // offset isn't specified, the next field that wasn't yet
        // retrieved by fetchField() is retrieved
        function fetchField($fieldOffset = -1) {
                if ($fieldOffset != -1) {
                        $this->_tempResult = @sybase_fetch_field($this->_queryID, $fieldOffset);
                }
                /*      The $fieldOffset argument is not provided thus its -1   */
                else if ($fieldOffset == -1) {
                        $this->_tempResult = @sybase_fetch_field($this->_queryID);
                }
                return $this->_tempResult;
        }

        // Returns: true if there still rows available, or
        // false if there are no more rows.  Moves to the next
        // row in a specified Recordset object and makes that record
        // the current row and the data corresponding to the row will
        // be retrieved into the fields collection.  Note: Unlike
        // the moveRow() method, when _currentRow is getNumOfRows() - 1,
        // EOF will immediately be true.  If row number is not
        // provided, the function will point to the
        // first row automatically
        function nextRow() {
                if ($this->getNumOfRows() > 0) {
                        $this->fields = array();
                        $this->_currentRow++;
                        $this->fields = @sybase_fetch_array($this->_queryID);
                        /*      This is not working.  True all the time */
                        if ($this->fields) {
                                $this->_checkAndChangeEOF($this->_currentRow - 1);
                                return true;
                        }
                }
                $this->EOF = true;
                return false;
        }

        // Returns: true on success, false on failure moveRow()
        // moves the internal row pointer of the Recordset object
        // to point to the specified row number and the data
```

```
// corresponding to the row will be retrieved into the
// fields collection.  If row number is not provided, the
// function will point to the first row automatically
function moveRow($rowNumber = 0) {
        if ($rowNumber == 0) {
                return $this->firstRow();
        }
        else if ($rowNumber == ($this->getNumOfRows() - 1)) {
                return $this->lastRow();
        }
        if ($this->getNumOfRows() > 0 && $rowNumber < $this->getNumOfRows()) {
                $this->fields = null;
                $this->_currentRow = $rowNumber;
                if(@sybase_data_seek($this->_queryID, $this->_currentRow)) {
                        $this->fields = @sybase_fetch_array($this->_queryID);
                        /*      This is not working.  True all the time */
                        if ($this->fields) {
                                // No need to call _checkAndChangeEOF()
                                // because the possibility of moving to the
                                // last row has been handled by the above code
                                $this->EOF = false;
                                return true;
                        }
                }
        }
        $this->EOF = true;
        return false;
}

// Returns: true on success, false on failure firstRow()
// moves the internal row pointer of the Recordset object
// to the first row and the data corresponding to the row
// will be retrieved into the fields collection
function firstRow() {
        if ($this->getNumOfRows() > 0) {
                $this->fields = array();
                $this->_currentRow = 0;
                if (@sybase_data_seek($this->_queryID, $this->_currentRow)) {
                        $this->fields = @sybase_fetch_array($this->_queryID);
                        $this->EOF = false;
                        /*      This is not working.  True all the time */
                        if ($this->fields) {
                                return true;
                        }
                }
        }
        $this->EOF = true;
        return false;
}
```

```
// Returns: true on success, false on failure lastRow()
// moves the internal row pointer of the Recordset object
// to the last row and the data corresponding to the row
// will be retrieved into the fields collection
function lastRow() {
        if ($this->getNumOfRows() > 0) {
                $this->fields = array();
                $num_of_rows = $this->getNumOfRows();
                $this->_tempResult = @sybase_data_seek($this->_queryID, --$num_of_rows);
                if ($this->_tempResult) {
                        /*      $num_of_rows decemented at above        */
                        $this->_currentRow = $num_of_rows;
                        $this->fields = @sybase_fetch_array($this->_queryID);
                        /*      This is not working.  True all the time */
                        if ($this->fields) {
                                /*      Special case for making EOF false.      */
                                $this->EOF = false;
                                return true;
                        }
                }
        }
        $this->EOF = true;
        return false;
}

// close() only needs to be called if you are worried
// about using too much memory while your script is
// running. All associated result memory for the
// specified result identifier will automatically be freed
function close() {
        $this->_tempResult = @sybase_free_result($this->_queryID);
        return $this->_tempResult;
}

/*      Returns: the number of rows in a result set.
Get number of rows in result.   */

function getNumOfRows() {
        return $this->_numOfRows;
}

/*      Returns: the number of fields in a result set.
Get number of fields in result. */

function getNumOfFields() {
        return $this->_numOfFields;
}

/*      Check and change the status of EOF.     */
function _checkAndChangeEOF($currentRow) {
```

```
                if ($currentRow >= ($this->_numOfRows - 1)) {
                        $this->EOF = true;
                }
                else {
                        $this->EOF = false;
                }
        }
}
?>
```

제 19 절 부록 I phpDB.inc 예제

제출자 : Joe Thong `darkjoe@softhome.net` Site URL: `<http://phpdb.linuxbox.com>`

Description: 여러가지 데이터베이스 서버에서 결과 조작을 위한 강력한 PHP 데이타베이스 랩퍼. 데이터베이스의 결과가 phpDB에 의해 자동으로 나오게 된다.

이화일을 얻으려면 웹브라우져에서 '다른이름으로 저장'을 누르고 phpDB.inc 라고 입력하시오.

```
<?php
/*
Name: phpDB General module
Version: 1.02bR6
Description: A PHP database wrapper for various
        database servers.  Database results are flushed
        automatically by phpDB.  Supported database
        servers are MySQL, MSQL, PostgreSQL, Microsoft
        SQL Server and Sybase.
*/

        if (!defined("_PHPDB_GENERAL_LAYER")) {
          define("_PHPDB_GENERAL_LAYER", 1 );
        }
        else
                return;

        // Fill in the database server that you're
        // going to use.  Consult the phpDB Reference
        // Manual for more information
        $databaseType = '';
        // The phpDB module root path.  No trailing slash
        $phpDBRootPath = '.';

        function useDB($dbType = "")
        {
                GLOBAL $phpDBRootPath;
                switch (strtolower($dbType))
                {
                        case "mysql":
                        case "msql":
```

```
                        case "postgresql":
                        case "mssql":
                        case "sybase":
                        case "informix":
                                include("$phpDBRootPath". "/phpDB-" . "$dbType.lib");
                                break;
                        case "":
                                die("Please edit phpDB.inc in order to use phpDB");
                                return false;
                        default:
                                die("Invalid database selection");
                                return false;
                }
                return true;
        }

        useDB($databaseType);
?>
```

제 20 절 부록 J phpDBTest.php3 예제

제출자 : Joe Thong `darkjoe@softhome.net` Site URL: `<http://phpdb.linuxbox.com>`

설명: 여러가지 데이터베이스 서버에서 결과 조작을 위한 강력한 PHP 데이타베이스 랩퍼. 데이터베이스의 결과가 phpDB에 의해 자동으로 나오게 된다.

이화일을 얻으려면 웹브라우저에서 '다른이름으로 저장'을 누르고 phpDB-mssql.lib 라고 입력하시오.(역주: phpDB-mssql.lib 가 아니고. phpDBTest.php3 라고해야합니다.)

```
<html>
< head>
        < title>Untitled< /title>
< /head>

< body>
<?php
        // 이화일은 phpDB.inc가 있는 디렉토리에 있다고 가정합니다.
        include("phpDB.inc");
        $db = new phpDB();
        $db->pconnect("hostName", "userName", "passWord", "databaseName") or die ("Can't connect to database serv
        $rs = $db->execute("SELECT * FROM Items");
        $numOfRows = $rs->getNumOfRows();
        echo "Number of Rows: $numOfRows";

        $rs->firstRow(); // 선택사항이나, 추천합니다.
        while (!$rs->EOF) {
                // Fields collection accessible as associative arrays too
                echo "<br>" . $rs->fields[0];
                $rs->nextRow(); // NOTE: nextRow() is placed at below
        }
```

```
        $rs->close();
        $db->close();   // 선택사항
?>
< /body>
< /html>
```