PHP HOW−TO

Table of Contents

Table of Contents

PHP HOW−TO

Al Dev (Alavoor Vasudevan) alavoor@yahoo.com

v25.4, 13 July 2001

This document tells you howto develop PHP programs and also to migrate all the Windows 95 GUI applications to powerful PHP + HTML + DHTML + XML + Java applets + Javascript. The information in this document applies to all the operating sytems where PHP is ported that is − Linux, Windows 95/98/NT/ME, Windows 2000, BeOS, Apple Macintosh (is BSD unix??), OS/2, all flavors of Unix like Solaris, HPUX, AIX, SCO, Unixware, Sinix, BSD, SunOS, etc.. and mainframe operating systems and on all operating systems where "C" compiler is available.

---

1. Introduction

Definition: *PHP is a simple, object−oriented, interpreted, robust, secure, very high−performance, architecture neutral, portable, dynamic scripting language. Everything inside "class" keyword in PHP will be identical to Java language (in near future). And PHP is designed such that it is 10 times faster than Java, since there is no virtual machine. PHP is the international standard general purpose object oriented scripting language.*

PHP stands for 'Hypertext Pre−Processor' and is a server side HTML scripting/programming language. PHP is a tool that lets you create dynamic web pages. PHP−enabled web pages are treated just like regular HTML pages and you can create and edit them the same way you normally create regular HTML pages.

PHP was kept the **"top secret and strictly confidential"** computer language by many companies in the world, but now had become the most well−known and most widely used object oriented scripting language for web, internet, e−commerce, general purpose and business−to−business projects. Even today many competing companies keep PHP language as a highly confidential matter not disclosing to outsiders (competitors).

PHP will storm the entire world and will take the IT industry by surprise!! The power of PHP is that it is **cross−platform and runs everywhere!!** It runs on Linux, Windows 95/98/NT, Windows 2000, Solaris, HPUX and all flavors of unix. PHP is write once and deploy anywhere and everywhere. It runs on many web−servers like Apache, Microsoft IIS, etc..

PHP runs 5 to 20 times faster than Java!! In actual benchmarks, PHP was 3.7 times faster than JSP (see benchmarks). PHP is extremely easy to use and you can develop very complex web/e−commerce applications very rapidly in a very short period of time. (In future PHP language will imitate most features of Java language and Java programmers will **love** PHP. And PHP will have java keywords like class, extends, interface, implements, public, protected, private etc..).

It has object oriented features and takes the best features from Java, C++, PERL and "C" languages. PHP language is a *marriage* of best features from Java, C++, PERL and C.

PHP is the **real gem** of all the scripting/programming languges and will soon become the "MECCA" for programmers world−wide!! PHP has a huge user base and a large developer base as it runs on both Window95/NT/2000 and all flavors of unixes.

A big surprise is waiting for us − **Most probably PHP will be the computer language/scripting language of the 21st century!!**

PHP can be compiled and optimized to make it run even faster by using the Zend Optimizer. Zend optimizer is integrated with PHP in PHP version 4.0.

You would normally use a combination of PHP (70% code) + HTML/DHTML/XML (25% code) + Javascript (5% code client side validations) for your e−commerce projects.

---

2. PHP runs on Microsoft Windows!!

PHP initially started on unix platform, but it is very portable and runs on MS Windows and MS IIS webserver. Today PHP has a large user base on MS Windows 2000/NT/95/98, You will find a huge collection of tools for PHP under MS Windows platform.

Many PHP programmers develop code on MS Windows and deploy on large linux servers like IBM mainframe running linux, Compaq DEC Alpha and Sun sparc.

A great advantage is that since PHP also runs on Unix/Linux, developers on unix platform "cash on" the user base of PHP under MS windows as the PHP code developed under MS Windows can be used on unix/linux without any code change!!

PHP itself is written in 100% "C" langauge, and hence it runs on a very wide variety of platforms like BeOS, Unix, MS Windows, Apple Macintosh, IBM OS/2 and on many more operating systems.

PHP is very fast and is much faster than Java. For web development, forget Java/JSP, it is PHP, PHP and PHP everywhere!! PHP is also becoming a general purpose object oriented scripting language

---

3. PHP Download

- PHP main site http://www.php.net
- PHP resources http://ils.unc.edu/web−db/php/links.html
- PHP Code Exchange − http://px.sklar.com

3.1 PHP Installation on Microsoft Windows 95/98/NT/2000

PHP is **IMMENSELY POPULAR** on Microsoft Windows platform and is **surprisingly more popular** than Microsoft's own ASP web scripting language!! A major reason for popularity is that PHP is a **object oriented scripting** language whereas ASP is not. PHP has a large collection of re−usable classes (objects). PHP runs lot faster than ASP on MS Windows and has more features and functionalities than Microsoft ASP. PHP is much more robust, reliable and powerful than ASP. And the user base of PHP is extremely large because PHP runs on MS Windows, Linux, Mac OS and all unixes. Greatest advantage of PHP is that you can develop on MS Windows and deploy on Linux or Unix and vice versa!!

There are more PHP users under MS Windows98/NT/2000 than on any other operating system!! Because there is so much demand for PHP on MS Windows 98/NT/2000, a ready to install executable is made and you simply double−click on the exe file to automatically install PHP in just 2 minutes. Download the PHP executable install file from

- MS Windows exe installer for PHP http://php.weblogs.com/easywindows
- Lots of info on PHP on MS Windows platform http://php.weblogs.com
- Install and config of PHP on MS Windows http://www.php.net/manual/install−windows95−nt.php
- PHP Triad installs a complete PHP server environment on Windows platforms http://www.phpgeek.com

3.2 Apache Webserver Quick−Install (10 seconds) on Microsoft Windows 95/98/NT/2000

You need a web−server to run the PHP on MS Windows. You can use MS IIS web server or you can use free Apache web−server for MS Windows 95/98/NT/2000. To save you lot of time here is the ready−to−install setup.exe file for apache for Windows platform:

PHPTriad which is Apache+PHP+MySQL single package is at http://www.phpgeek.com/phptriad.php and at mirrorsite. I very strongly recommend PHPTriad as it is immensely popular among MS Windows users (millions of downloads).

Apache binaries − http://httpd.apache.org/dist/httpd/binaries/win32

3.3 SQL server for Microsoft

SQL server can be on a seperate box which need not be running MS Windows. You also need a SQL server for doing web development. I recommend that you install Redhat Linux on a very old PC like (Pentium or 486 box) and install the PostgreSQL RPMs on it. You do not need any windows graphics for a database server and at console mode startup the PostgreSQL server. PostgreSQL is **3 times** faster than Oracle or MS SQL server.

You can also order ready−to−go cheap Linux boxes from −

- Egghead Egghead, click on Auctions and Linux boxes, you get best deals in live Auctions.
- Goto LinuxHardware and click ComputerSystems, LinuxOnline, Linux hardware Crynwr, Linux HarwareNet
- US Land5, US QLiTech, US CompaqLinux, US VAlinux, US StoreAnywhere
- In Europe : UK GBdirect, UK MultiT, Hungary Leonardo, Belgium Mind, Germany Spier
- DEC alpha linux DECalpha and CompaqAlphaLinux

You can also get PostgreSQL for Windows NT/2000 from http://www.askesis.nl.

See also the PostgreSQL howto at pgsql−howto

3.4 PHP Installation on unixes and others

See the installation guide and instructions at PHP main site http://www.php.net or INSTALL file in the downloaded package itself.

4. Quick Start steps

To implement a project in object oriented PHP do −

- First you need connection to database sql servers − use one of these:
    - **ADODB** (Active Data Objects Data Base) http://php.weblogs.com/ADODB
    - **Metabase** (Database independent access and management) http://phpclasses.upperdesign.com/browse.html/package/20
- Second you need generic PHP classes to create forms, tables and other HTML objects. Get these from **PHP classes** at http://phpclasses.upperdesign.com. See also the Top downloaded classes from Top PHP classes And visit http://www.phpclasses.com. And see Form classes, template classes at the site corporate Intranet/Internet
- Third, design and create your own PHP classes by inheriting from the generic PHP classes or http://www.phpclasses.com.
- Fourth, use the template classes to seperate the presentation from business logic, see http://www.phpclasses.com.
- Fifth, use IDE tools from PHP IDE
- Most of your PHP code should be in classes for easy code maintainence and re−usability.

See also

5. PHP Libraries

The PHP is a object−oriented scripting language. Hence PHP code is **"classes, classes, classes and classes"**. When you write PHP code you must design your classes such that they are re−usable or they use existing PHP classes. There are hundreds of PHP classes already written and ready to use. There are classes for accessing databases, classes for generating XML, HTML forms, classes for creating tables, forms and other HTML objects. If you write some generic classes then please upload to sites. By year 2005, there will be more than hundred million re−usable PHP classes. Advantage of PHP classes are they provide − data hiding, inheritance, encapsulation, reliability, reusability and polymorphism. The most important PHP sites are **PHP classes** and **PEAR** as given below.

5.1 Classes and PEAR

Visit the following web sites which have large collections of ready to use PHP class libraries.

- **PHP classes** are at http://phpclasses.upperdesign.com and at http://www.phpclasses.com.
- **PEAR (PHP Extension and Application Repository)** is a code repository for PHP extensions and PHP library code similar to Perl's CPAN and is at http://pear.php.net and mirror linux php−pear−rpms and pear−tutorials.

- **ADODB** (Active Data Objects Data Base) http://php.weblogs.com/ADODB : PHP's database access functions are not standardised. Every database extension uses a different and incompatibile API. This creates a need for a database class library to hide the differences between the different databases (encapsulate the differences) so we can easily switch databases. ADODB currently support MySQL, PostgreSQL , Interbase, Oracle, MS SQL 7, Foxpro, Access, ADO, Sybase and generic ODBC. See also ADODB−manual. The PhpLens is based on ADODB.
- **Metabase** (Database independent access and management) http://phpclasses.upperdesign.com/browse.html/package/20

- **PHP Hot scripts** http://www.hotscripts.com/PHP

- Very **popular PHP resources** http://php.resourceindex.com and CGI−Resources

- PHP projects are at http://php.net/projects.php

- PHP Lib Netuse http://phplib.netuse.de (This is deprecated, merged with PEAR)
- PHP widgets http://www.northern.ca/projects/phpwidgets
- Generic Framework PHP4 http://sourceforge.net/projects/gpfr
- Source Forge PHP Lib http://phplib.sourceforge.net
- Source Forge PHP Snippets, go and click on PHP here http://sourceforge.net/snippet
- E−gineer PHP lib http://e−gineer.com/articles/php−hacker
- FAQ PHP http://php.faqts.com
- PHP Lib http://px.sklar.com
- PHP Factory http://alfalinux.sourceforge.net/phpfact.php3

- PHP Builder site http://phpbuilder.com/snippet
- PHP developer http://www.phpdeveloper.org
- PHP newbie http://www.newbienetwork.net
- PHP walrus http://www.evilwalrus.com

5.2 Other PHP Utilities

Other PHP utilities are at :

- User Login sample: http://www.devshed.com/Server_Side/PHP/Commerce1

- phpPDFtable is a class written in php to ease the creation of tables in PDF files. It requires php (4.0 but should run with 3.x too), and pdflib http://sourceforge.net/projects/phppdftable

- Data−Admin aims to provide a PHP based interface to Database Administration. This will not be limited to just one or two databases. Also, the underlying class library encapsulates the native PHP database calls allowing the programmer to use one set of fu http://sourceforge.net/projects/dadmin

- PSlib is a PHP library for generating PostScript files. It offers an easy way of generating PostScript files: simple call PSlib functions from within your PHP script and the PS files are created on the fly http://sourceforge.net/projects/pslib

- A complete collection of php scripts that work tightly together to create a highly customizable, dynamic and module oriented website http://sourceforge.net/projects/twebs

- phpOpenTracker is a comprehensive solution for your site− and visitor−tracking needs. The collected data is stored in a SQL database, allowing complex, yet easy analysis. A powerful API for analysis and report generation (HTML or PDF output) is included. http://www.phpopentracker.de
- PHPShopCart is a shopping−cart web application demo written in PHP and designed to connect to a MySQL database. It was written for the book, "A Guide to Databases under Linux" (Syngress Media) but is available under the GNU Public License. http://sourceforge.net/projects/phpshopcart

---

6. PHP Application Servers

The PHP applications are categorized into more than 60 groups in hotscripts.com site at HotScripts − PHP. You MUST visit this site before looking elsewhere.

6.1 Build PHP based corporate Intranet and Internet

Visit the site corporate Intranet/Internet for a very good comparison and listing of applications.

6.2 Popular PHP Applications

The most popular PHP applications in the order are:

1. PHP−Nuke and the mainsite
2. Smarty and the mainsite
3. eZ Publish and the mainsite
4. Mambo Portal/Content Mgmt and the mainsite
5. PHP Content Management System and the mainsite
6. PHP Shop and the mainsite
7. phpWebSite and the mainsite
8. myPHPCalendar and the mainsite
9. TreeMenu and the mainsite
10. Backend and the mainsite
11. Typo3 Content Mgmt and the mainsite
12. E−Guest
13. PHProjekt and the mainsite
14. Phorum and the mainsite

PHP has several tools which are given below:

6.3 PHP Web Application Servers

The following are available for PHP:

- **PHP Lens** http://phplens.com is a rapid application development component which allows PHP developers to dynamically and quickly create web applications that retrieve information from databases. With phpLens, data can be presented as html tables with facilities to create, edit, paginate, search and delete records. PHPLens uses ADODB. See also ADODB.

- ADODB (Active Data Objects Data Base) http://php.weblogs.com/ADODB : PHP's database access functions are not standardised. Every database extension uses a different and incompatibile API. This creates a need for a database class library to hide the differences between the different databases (encapsulate the differences) so we can easily switch databases. ADODB currently support MySQL, PostgreSQL , Interbase, Oracle, MS SQL 7, Foxpro, Access, ADO, Sybase and generic ODBC. See also ADODB−manual.
- **Metabase** (Database independent access and management) http://phpclasses.upperdesign.com/browse.html/package/20

- Binary Cloud http://www.binarycloud.com is an opensource, enterprise class web application platform It provides a complete set of core services such as authentication, permissions, database abstraction, and error handling − and a rich set of advanced tools built on that foundation. The system is deployed in a number of commercial installations and has proven to be robust, scaleable, and secure. Binarycloud is suitable for large−scale commerce and publishing projects, or anything of similar complexity. It offers some unique security features − such as selective encryption of database I/O − which make it a particularly good choice for web applications that require a high level of data security. And it's free.

- PEAR (PHP Extension and Application Repository) is a code repository for PHP extensions and PHP library code similar to Perl's CPAN and is at http://lxr.php.net/source/php4/pear and pear−tutorials.

- Midgard is Content management system is based on Apache and uses the PHP scripting language The main site of Midgard is at http://www.midgard−project.org PHP can be compiled with Zend compiler and optimizer http://www.zend.com. PHP runs very fast and is about 5 to 10 times faster than Java. See Midgard Installation

- Ariadne http://www.muze.nl/software/ariadne is a web application system. It consists of a complete framework for the easy development and management of web applications, using PHP. The system uses a modular approach, using abstract interfaces for all transactions. This results in maximum freedom to change parts of the systems workings or add new functionality without needing to reprogram other parts

- Group IT Engine http://groupit.org is a turnkey group collaboration and content management engine. It presently runs on Unix machines using PHP and Apache. Using GroupIT you can "Categorize your content", "Organize your contents into sections", "Control access to your content" and many more additional features.

6.4 PHP Template Engines

- Php Sitemanager − do code creation, a layout design, site implementation and site management. Visit SiteManager

- Smarty Template Engine − is a template engine for PHP. One of the unique aspects about Smarty that sets it apart from other templating solutions is that it compiles the templates into native PHP scripts upon the first invocation. After that, it merely executes the compiled PHP scripts. Therefore, there is no costly template file parsing for each request. See Smarty QuickStart and docs about Smarty are at http://www.phpinsider.com/php/code/Smarty/docs. Get it from download−smarty.

- DreamTime Template http://www.phptemplates.org

- Fast Template http://curtisonline.net/theme/phpfast−templates−HOWTO.html and main−site
- PHP template Layout classes commercial, VH Layout http://www.vhconsultants.com and see this−article

6.5 PHP based Web−Portal systems

The following ready−made Web−Portal systems are available:

- PHP Nuke Web Portal system at PhpNuke
- DarkPortal is a Web portal system similar to PHPNuke. It is based on the user interface style of Slashdot and other news/portal systems. It uses a modular core as a base for adding pluggable modules to facilitate content creation and management. Its primary goals are full separation of content from code, fully modular portal plugins with standardized APIs, portability and database abstraction, user level theme selection and plugin module selection, and a modular user/group hierarchy with multiple levels of moderation and admin capabilities. Visit DarkPortal
- CMS system ezPublish
- Mambo is a feature−rich, dynamic portal engine/content management tool based on PHP/MySQL. It features a very powerful and extensive admin manager, and uses a modular framework for extensibility. Visit Mambo and mirror

7. Object Oriented Features − public, private, protected

PHP scripting language provides object oriented features through the **class** keyword. Features like public, private and protected will be supported in the future release (they are in TODO list). In the meantime, you can use the following coding conventions to distinguish between private, public and protected variables:

1. All private variables and functions always start with underscore **"_"** followed by lowercase letters like **var $_myvar;**
2. All Protected variables and functions always start with **"_T"** followed by lowercase letters like **var $_Tmyvar;**
3. All Public variables and functions do not start with underscore "_" like **var $myvar;**
4. All variables and functions always start with lowercase letter (no uppercase) like **var $_myvar;** and NOT like **var $_Myvar;**

```
class someabc {
        var $_conn;                     // Private variable
        var $_Tmyvar;                   // Protected variable
        var $connMYCONNECTION;          // Public variable
        var $connToDb;                  // Public variable
        var $myvar3;                    // Public variable
        var $myvarTHISTEST;             // Public variable
        function _foofunction() {}      // Private function
        function _Tfoofunction() {}     // Protected function
        function foofunction() {}       // Public function
}
```

The private, protected declarations provide the encapsulation and data−hiding. But you must consider the following disadvantages of encapsulation:

- Encapsulation usually requires more code, hence it sacrifices performance especially for scripting languages like PHP
- Encapsulation requires lots of Set/Get methods for private/protected properties.
- Since encapsulation unneccessarily increases the code size, it is not recommended for scripting language like PHP.
- You can enforce a good degree of encapsulation by using the coding convention suggested in this section.

8. HTML Editor

HTML editors for PHP on Windows 95/NT/2000 are:

- 1st Page 2000 − Rated 1st (the best HTML editor) http://www.evrsoft.com
- Cool Page − Rated 2nd http://www.coolpage.com
- Coffeecup − Rated 3rd http://www.coffeecup.com/editor
- Arachnophilia − Rated 4th http://www.arachnoid.com/arachnophilia/index.html
- Textpad http://www.textpad.com and textpad−php−add−ons

The best HTML editor is **1st Page 2000**, and it is a excellent HTML editor.

9. IDE tools for PHP

Many HTML editors are supporting PHP. In near future every HTML editors and XML editor will be supporting PHP "Rapid Application Development" tool.

You will notice that some of the PHP editors run only on MS Windows. Yes!! there are millions of PHP developers on MS Windows platform. PHP is **IMMENSELY POPULAR** on Microsoft Windows platform and is **surprisingly more popular** than Microsoft's own ASP web scripting language!! PHP runs lot faster than ASP on MS Windows and has more features and functionalities than Microsoft ASP. PHP is much more robust, reliable and powerful than ASP. There are more PHP users under MS Windows98/NT/2000 than on any other operating system!! PHP initially started on Linux/unix environment but today there are more PHP developers on MS Windows platform as compared to unix.

9.1 PHP IDE

PHP IDE tools are at :

- **PHPAkt**The PHP support in Macromedia Dreamweaver UltraDev that allows Ultradev Developers to create PHP sites PHPAkt
- **PHP Lens** http://phplens.com is a rapid application development component which allows PHP developers to dynamically and quickly create web applications that retrieve information from databases. With phpLens, data can be presented as html tables with facilities to create, edit, paginate, search and delete records. PHPLens uses ADODB. See also ADODB.
- PHP IDE and Editor "PHP Coder" http://www.phpide.de and mirror
- Zend PHP IDE http://zend.com/store/products/zend−ide.php
- IDE for PHP scripting (Web Browser) : http://www.ekenberg.se/php/ide
- Nexidion PHP IDE http://www.nexidion.org
- Enter in Search keyword 'PHP IDE' in Source Forge http://sourceforge.net
- Color editor gvim for PHP (Win and linux) at http://metalab.unc.edu/LDP/HOWTO/Vim−HOWTO.html and see also ptags of PHP

9.2 PHP IDE for MS Windows only

PHP IDE/editor on MS Windows platform are :

- Rated 1st (the best PHP tool on MS Windows) : PHPTriad is a complete PHP development and server environment for Windows. It installs PHP, Apache, MySQL, and PHPMyAdmin, both installing and setting up the environment. PHPTriad is at http://www.phpgeek.com/phptriad.php and at mirrorsite
- Rated 2nd : PHP Coder http://www.php−ide.com
- Rated 3rd: PHPEd (Soyal), an excellent PHP editor (MS Windows) http://soysal.free.fr/PHPEd

- IDE for PHP editor (MS Windows): http://www.phpedit.com
- Color editor gvim for PHP (Win and linux) at http://metalab.unc.edu/LDP/HOWTO/Vim−HOWTO.html and see also ptags of PHP
- IDE for PHP (MS Windows) http://www.pc−service−boerner.de/PHPScriptEditor.htm
- "EditPlus Text editor" win32 http://www.editplus.com (high rating 5 stars)
- eNotepad win32 http://www.edisys.com/Products/eNotepad/enotepad.asp (high rating 5 stars)
- PHP editor win32 http://www.chami.com/html−kit (high rating 5 stars)
- UltraEdit win32 http://www.ultraedit.com with PHP word file at http://www.ultraedit.com/downloads/additional.html
- ScriptWorx editor win32 http://www.softlite.net/products/scriptworx (rating 4.5 stars)
- TextPad editor win32 http://www.textpad.com (rating 4.5 stars)
- PHP editor "ASPEdit" http://www.tashcom.com/aspedit (rating 3.5 stars) along with PHP code explorer http://www.tashcom.com/codex (rating 4.5 stars)
- HTML/PHP editor Dreamweaver http://www.dreamweaver.com
- HTML/PHP editor Homesite http://www.allaire.com/homesite
- HTML/PHP editor Hotdog http://www.hotdog.com

9.3 PHP IDE for both MS Windows and Linux

PHP IDE/editor for bot MS Windows and Linux platforms are :

- PHP editor (for both windows and linux/unixes) http://www.coffeecup.com/select/editor.html (rating 5 stars).
- HTML/PHP editors Amaya http://www.w3.org/Amaya
- Folding text editor (Win and linux) http://fte.sourceforge.net

- PHP Editor (Win and linux) http://www.scintilla.org
- Color editor gvim for PHP (Win and linux) at http://metalab.unc.edu/LDP/HOWTO/Vim−HOWTO.html and see also ptags of PHP
- Jed (win and linux) http://space.mit.edu/~davis/jed.html

- Editors for PHP : http://www.itworks.demon.co.uk/phpeditors.htm
- Editors for PHP : http://www.oodie.com/tools/index.php?view=editor

9.4 PHP IDE for Linux only

The best IDE for PHP on linux is Coffeecup Editor as given below:

- Color editor gvim for PHP (Win and linux) at http://metalab.unc.edu/LDP/HOWTO/Vim−HOWTO.html and see also ptags of PHP
- PHP editor http://www.coffeecup.com/select/editor.html (rating 5 stars).
- HTML/PHP editors Quanta http://quanta.sourceforge.net
- HTML/PHP editors Blue Fish http://bluefish.linuxave.net
- HTML editors AswEdit http://www.advasoft.com

9.5 PHP Utilities

- Zend Optimizers http://www.zend.com
- Zend Compilers http://www.zend.com

- Lots of info on PHP on MS Windows platform http://php.weblogs.com

- PHP GroupWare Apps http://www.phpgroupware.org
- PHP Web Shop http://www.phpshop.org
- PHP Nuke Web Portal system PhpNuke

9.6 Convert Microsoft ASP scripts to PHP − ASP2PHP

To convert ASP scripts to PHP use this utility

---

10. ctags for PHP

Tags are extremely valuable and are used for navigation of source code inside the editors like vi, emacs, CRiSP, NEdit etc... If you had programmed a lot in C, C++ or Java you might have used the **ctags** program to create tags. To see the online manual page, type 'man ctags' at linux/unix bash prompt.

The **ptags** program for PHP is given below, which you can use to create the tags for PHP source code. Your **productivity will improve 3 to 4 times** if you use **ptags**.

See also Vim color text editor for PHP, C, C++ at http://metalab.unc.edu/LDP/HOWTO/Vim−HOWTO.html

---

```
// Save this file as ptags.cpp and compile by
//              g++ −o ptags ptags.cpp
```

```
//*****************************************************************
// 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 : 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];
                        char aapointer[BUFF_LEN + 100];
                        memset(aa, 0, BUFF_LEN +10);
                        strcpy(aa, buff);
                        strcpy(aapointer, ltrim(aa));
                        strcpy(aa, aapointer);

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

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

---

11. Debugging PHP

11.1 Debug Tool − phpCodeSite

PHP Debugger called 'phpCodeSite' is available at http://phpcodesite.phpedit.com and see also Appendix C

11.2 Debug Tool − phpDebug

PHP Debugger is available at http://www.phpdebug.com

11.3 Debug with FILE and LINE

To debug PHP programs create a file "debug2.inc" having the following functions :

```
<?php

/* define this variable, to prevent double declaration. */
if (!defined("_DEBUG2_DEFINED_"))
{
        define("_DEBUG2_DEFINED_", 1 );
}
else
        return; // if this file is already included then return

# file name : debug2.inc
# Functions for debuging the PHP source code
#*****************************************************************
# 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");
}

?>
```

In your PHP source code initial page which is generally index.php3, put a line like

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

        init_debug_file();
        // all other commands follows here ...
        // ...........
?>
```

To output debug values, in your PHP source code files, put debug2_() calls as illustrated below:

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

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

When you run the PHP program the output will be traced in the file called debug.out giving the filename, linenumber, variable name and it's value.

Use the debug2_() generously in your code. The usage of debug2_() calls in your program will **NOT** have any impact on the final production code and also has no impact on the performance because they will be filtered out as described below. You can use copy and paste to save time of typing debug2() calls or use the 'yank to buffer' feature of Vi editor and paste.

When you are done development and testing and when you are ready to deploy on the production server, filter out the debug2_ calls from your source code. At unix prompt −

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

For a large group of files −

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

And now copy the files from production to the deployment area.

12. General purpose programming with PHP

PHP is very powerful and is designed such that it can replace awk, sed, unix shell, perl, "C", C++ and Java.

The object oriented features of PHP is developing very rapidly and in near future will surpass the object oriented features of Java language. All the object oriented features are implemented in PHP via **class** keyword just like in Java.

If you want to use PHP as a stand−alone program, just like a shell script, "C" or perl program, then use this technique:

```
bash$ php  filename.php
bash$ php −h
bash$ php −?
bash$ /usr/bin/php −?
bash$ php −i
```

The command *php filename.php* will execute the file *filename.php*. You are invoking the php program which you wrote in filename.php from the bash commmandline instead of from the web−browser.

On Microsoft Windows platform you will bringup a MSDOS prompt from Start−>Run−>cmd and put C:\Program Files\php\bin in your path environment and

```
C:\> php filename.php
C:\> php −h
C:\> php −?
or
C:\> c:\Program Files\php\bin\php filename.php
```

12.1 Standalone MS Windows GUI applications using PHP

Since PHP is general purpose scripting language and is like a *"glue"* language, you can use it develop standalone MS Windows GUI applications. PHP can be very easily combined with MS Windows C++ GUI classes to create GUI applications. And developing applications with PHP is extremely fast as it is a scripting language and it's runtime performance is also excellent as compared with other scripting languages like Perl, Visual Basic and Python.

PHP can also be used for developing standalone GUI applications for Linux/Unixes.

13. Performance benchmarking − PHP, ASP, JSP, Coldfusion

It is very important to bear in mind that performance and running speed of the web scripting engine must be given **TOP PRIORITY**. That is, how many pages per second the scripting engine can pump out to the browser clients. The greater the number of pages pumped out to clients in a given period of time, then the better and more powerful the scripting engine is.

The Zdnet did a evaluation and benchmarking of 4 web scripting languages. During benchmarking, the same spec and identical cpu, memory boxes were used. Under identical conditions, it was found that PHP was the fastest − about 3.7 times faster than JSP and about 1.2 times faster than ASP. Read the report at eWeek and mirror−site The benchmark results are −

- PHP pumped out about 47 pages/second
- Microsoft ASP pumped out about 43 pages/second

- Allaire ColdFusion pumped out about 29 pages/second
- Sun Java JSP pumped out about 13 pages/second

See also PHP, ASP benchmarks at http://aldev0.virtualave.net/php−perl−benchmarks.html

Whenever you design a web site, give attention to these important points:

- Speed of web scripting engine − how many pages per second it can pump out.
- KISS policy (Keep It Simple Stupid!!) − your web page should be very simple without any fancy graphics (because web users do not want to wait for long and they want the information very fast). And information they read is just plain text!!

---

14. Limitations of PHP

Everything has limitations or disadvantages and PHP is no exception. The following are the limitations of PHP (so be **WARNED !!**)

1. PHP is NOT 100 % pure Object Oriented scripting language. But in near future PHP may support 100% object oriented scripting (PHP may imitate most of the syntax of Java language). PHP already imitates some features of Java language. (In future PHP language will imitate most features of Java language and Java programmers will **love** PHP. And PHP will have java keywords like class, extends, interface, implements, public, protected, private etc..).
2. PHP will NOT give the performance of "C" or "C++" language. Because it is scripting language and is interpreted it will be a bit slower than the optimized "C++" programs. For top performance, you should use "C++" and fast−CGI with database/webserver connection pooling and use C++ compiler optimizer "−O3" options. Zend optimizer in PHP 4 will speed up the performance of PHP and bring it very close to optimized "C" code .
3. But note a point that PHP was designed for very Rapid Web−Application Development tool. If it takes about 3 months to code a web application in C++, then using PHP you can develop the same web application in **just 4 days**!! And with zend optimizer, the speed of execution of PHP will be very close to that of equivalent C++ program!! Hence, there is really no advantage in using C/C++ for web development. PHP itself is written in 100% "C" language.

On the other hand, PHP has lot of advantages and it's advantages outweigh it's limitations −

1. You can very rapidly develop web applications in PHP as compile and link is eliminated in PHP scripting language.
2. PHP applications are very stable and do not depend on the browser technologies unlike Javascript applications which depend on browsers. PHP will give you the freedom to select any server platform. The browser does not know that the HTML page is generated by PHP !!
3. PHP has excellent database conectivity to all SQL database servers.
4. PHP has partial support for Object oriented features
5. PHP has C++, Perl, Javascript like syntax features and has programs like 'ptags/ctags' to navigate the source code
6. PHP has Zend optimizer which speeds up the performance
7. PHP runs on all unixes, linux, Windows 95/NT/2000 and is more powerful than ASP, JSP and others.
8. PHP has a very large user base and developer base.

See also Python: If you want 100% pure Object Oriented scripting language than you MUST consider **Python**. The 'Python' is a object oriented scripting language from ground up. You would be using the Python Web Application server called 'Zope' which is available at − http://www.zope.org and python is at

---

15. PHP Tutorial

Visit the following PHP tutorial sites −

- PHP Resource index − important PHP site − has complete scripts, Functions, classes, documentation, examples and tutorials http://php.resourceindex.com
- PHP portal http://zend.com

- **PHP Hot scripts** site http://www.hotscripts.com/PHP

- Very **popular PHP resources** http://php.resourceindex.com and CGI−Resources

- Simple tutorial http://www.php.net/tut.php
- Web Monkey http://hotwired.lycos.com/webmonkey/99/21/index2a.html
- Dev Shed http://www.devshed.com/Server_Side/PHP/Introduction
- PHP TidBits http://www.htmlwizard.net/resources/tutorials
- PHP Builder http://www.phpbuilder.com/getit

In this tutorial we assume that your server has support for PHP activated and that all files ending in .php3 are handled by PHP.

Your first PHP−enabled page: Create a file named hello.php3 and in it put the following lines:

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

Note that this is not like a CGI script. Think of it as a normal HTML file which happens to have a set of special tags available to you.

If you tried this example and it didn't output anything, chances are that the server you are on does not have PHP enabled. Ask your administrator to enable it for you.

The point of the example is to show the special PHP tag format. In this example we used < ?php to indicate the start of a PHP tag. Then we put the PHP statement and left PHP mode by adding the closing tag, ? > . You may jump in and out of PHP mode in an HTML file like this all you want.

We are going to check what sort of browser the person viewing the page is using. In order to do that we check the user agent string that the browser sends as part of its request. This information is stored in a variable. Variables always start with a dollar−sign in PHP. The variable we are interested in is $HTTP_USER_AGENT. To display this variable we can simply do:

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

For the browser that you are using right now to view this page, this displays:

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

There are many other variables that are automatically set by your web server. You can get a complete list of them by creating a file that looks like this:

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

Then load up this file in your browser and you will see a page full of information about PHP along with a list of all the variables available to you.

You can put multiple PHP statements inside a PHP tag and create little blocks of code that do more than just a single echo.

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

We can take this a step further and show how you can jump in and out of PHP mode even in the middle of a PHP block:

```
        <?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>
                <?
        }
        ?>
```

Instead of using a PHP echo statement to output something, we jumped out of PHP mode and just sent straight HTML. The important and powerful point to note here is that the logical flow of the script remain intact. Only one of the HTML blocks will end up getting sent to the viewer. Running this script right now results in:

You are using Internet Explorer

Dealing with Forms

One of the most powerful features of PHP is the way it handles HTML forms. The basic concept that is important to understand is that any form element in a form will automatically result in a variable with the same name as the element being created on the target page. This probably sounds confusing, so here is a simple example. Assume you have a page with a form like this on it:

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

There is nothing special about this form. It is a straight HTML form with no special tags of any kind. When the user fills in this form and hits the submit button, the action.php3 page is called. In this file you would have something like this:

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

Surprise!! The $name and $age variables are automatically set for you by PHP !!

15.1 Primer on PHP Sessions

This section is written by Ying Zhang .

Before we begin, let's quickly go over the concept of a session and the reason we need it. It's hard (for me) to define what a session is exactly, so let's use an example that should be very familiar to you −− logging in to your computer and using it every day. After you log in, your computer knows who you are. Every action that you perform is done so with your name.

So what's so special about that −− we take it for granted every time we have to login to any system. What's the big deal with doing this on the web? Well, the web (or specifically, the HTTP protocol) is connectionless. That means each request made to a web server is independent of all the other requests. Whereas your computer keeps information about you in memory and knows when you log in and out, a web server doesn't. A web server simply waits for requests and sends responses.

Let's illustrate this a little bit:

```
John Doe          __________         Jane Doe
  1              |          |            2
  3  ------------>|web server|<----------- 4
  5              |__________|            6
```

Let's say we only have two people, John Doe and Jane Doe, accessing MyMarket, and their actions are like this:

1. John looks at the product catalog.
2. Jane looks at the product catalog.
3. John adds an item to his basket.
4. Jane adds an item to her basket.
5. John goes to the checkout.
6. Jane goes to the checkout.

Since HTTP is connectionless, each request is completely isolated from the other requests. So how does the server know who's doing what? How does the server know that actions 1, 3, 5 are from John, and actions 2, 4, 6 are from Jane? Well, to make a long story short, the web server doesn't have to know. It can continue on happily responding to requests, session management has to be done with the backend scripting language.

What we need is a way to group together requests by the same person into the same session. This is where PHP4's session management capabilities come in. It can group together requests made from the same source (eg. client's browser) into the same session, we have to provide the smarts to associate users with sessions.

In other words, PHP4's session management can tell us requests 1, 3, and 5 belong to the same session (call it session A). Our application has to know that session A is owned by John Doe.

15.2 Session Management in PHP4

PHP4 adds some session management functions that make our life easier when dealing with sessions. The ones we are interested in are:

```
session_start();
session_register();
```

session_start() is used to start up PHP4's session management capabilities; you need to call it before you use any of the other session functions. session_register() is used to tell PHP which variables to track in the session. A typical call to these functions would look like this:

session_register("SESSION");

This tells PHP to start up the session manager, and tells PHP that the variable called SESSION is a session variable. You can register as many session variables as you like, but I prefer to only register one session variable called SESSION, and anything I need persistent I put into this variable. For example, I like to say:

```
session_register("SESSION");
$SESSION["var1"] = 5;
$SESSION["var2"] = 6;
```

instead of

```
session_register("var1");
session_register("var2");
$var1 = 5;
$var2 = 6;
```

because after you register lots of session variables, you tend to forget what they were, well, at least I do :).

Anyhow, by now you probably want to see some code in action, so create a script called session_test.php somewhere accessible, and put into it:

```
<?
session_start();
session_register("SESSION");

if (! isset($SESSION)) {
        $SESSION["count"] = 0;
        echo "<li>Counter initialized, please reload this page to see it increment";
} else {
        echo "<li>Waking up session $PHPSESSID";
        $SESSION["count"]++;
}
echo "<li>The counter is now $SESSION[count] ";
?>
```

Fire that up in your browser, the first time you hit the page, it should say " Counter initialized, please reload this page to see it increment". Each time you reload it, the counter value should increment by one. You will also see the session ID. If it does, then hurray, your PHP4 session manager works :)

So how does this work? Well, when you call session_start(), PHP4 determines a unique session ID for the client. This session ID is an MD5 hash of something (not sure what), and it is stored as a cookie on the client's PC.

Now each time that client makes a request, PHP4 will read this session ID and load up the data for the session. When you call session_register(), you are telling PHP4 which variables you want kept in the session. Each page that loads up, the previous values for the registered variables will be reloaded, and each time the page ends PHP4 will save the values of the registered variables.

By default, PHP keeps track of the sessions in temporary files in the /tmp directory, take a listings and see for yourself:

You will see something like this:

```
−rw−−−−−−−   1 apache   web            10 May  7 15:27 sess_6dd9ea8e61cd49cd3ad6de8c8b8885e8
−rw−−−−−−−   1 apache   web            10 May  7 19:49 sess_7d7f97afb6759948f554b00272494e52
−rw−−−−−−−   1 apache   web             6 May  9 01:00 sess_8ab78830e151add9d79b628958ce4eb9
−rw−−−−−−−   1 apache   web            31 May  9 11:41 sess_a3058a6bb1baf57f565c3844c8810f4b
−rw−−−−−−−   1 apache   web            30 May  9 11:42 sess_c379faad83ad3dc8ab6d22c14dbab3b4
−rw−−−−−−−   1 apache   web             6 May  8 01:00 sess_cd68a5054241aff1a8157c289683e869
−rw−−−−−−−   1 apache   web            34 May  7 15:17 sess_cd97e41912b28c44cc0481b7d978cb61
−rw−−−−−−−   1 apache   web            42 May  9 11:23 sess_d1285edd0c951c70b1aec17a5f602fc0
−rw−−−−−−−   1 apache   web            30 May  9 11:42 sess_da93f6e19b6be01257d7a6453766a23d
```

```
-rw-------   1 apache   web           42 May  7 21:26 sess_e837123c1af78c538e89b47030fde337
```

Each one of those files is a session, let's take a look at one of them (note, you probably have to su to root to peek inside a session file). Tip: don't just cut and paste the following commands, you need to specify the name of a real file:

```
# more /tmp/sess_a3058a6bb1baf57f565c3844c8810f4b
```

You will see something like this:

```
SESSION|a:1:{s:5:"count";i:234;}
```

Does that look familiar? It should if you've ever used the serialize() and unserialize() functions in PHP. If not, don't worry about it. Anyhow, I just wanted to illustrate how sessions were stored. You can rewrite the PHP session handlers to store sessions into a database or whatever else, but that's beyond the scope of this tutorial (but it's not hard at all).

15.3 User Management and Privileges

Okay, we've spend enough time on PHP4's session management, all you really need to get out of that was the two functions session_start() and session_register(). Let's get back to the issue of keeping track of users.

PHP can help us keep track of sessions, and group requests from the same session together. Now, we have to do our part and associate user accounts with these sessions. We will use a variable called SESSION["user"] to keep track of user information. When a user logs in, we will put their information into this variable. As long as this variable is defined, we will assume that a user has logged in. When a user logs off, we will clear out this variable.

Specifically, we will keep the following information about the user:

```
SESSION["user"]["username"] This is the user's login ID (their nick name if you will), and it is
SESSION["user"]["firstname"] The user's firstname.
SESSION["user"]["lastname"] The user's lastname.
SESSION["user"]["email"] The user's email address.
SESSION["user"]["priv"] The user's privilege level.
```

Let's talk a bit about the privilege levels. We are going to have two levels of security: (1) normal customers and (2) administrative users. Normal customers can use the system, browse through the catalog, and do other customer functions. Administrators can do everything a normal user can do, but also has the ability to perform system administrative functions. In real life, there are probably many more privilege levels that you want defined but we are going to keep things simple here.

This is all fine and dandy, but where do we get this user information from? We need to have a way to store all the users on the system, and the perfect place for that would be in the database. We're going to create a users table to hold all our users.

15.4 Step1: Creating the Users Table

Start up database server and login to database. Let's create the user table:

```
psql> CREATE TABLE users (
->   username     char(16) not null,
->   password     char(32) not null,
->   priv         char(5) not null,
->   firstname    varchar(64) not null,
->   lastname     varchar(64) not null,
->   email        varchar(128) not null,
->   phone        varchar(32) not null,
->   address      varchar(255) not null,
->   PRIMARY KEY (username),
->   UNIQUE email (email)
-> );
```

Notice the constraints we've put on the users table, the username is the primary key (which makes sense, you should be able to identify a user record based on the username). The email address has a unique constraint as well because we don't want duplicate email addresses.

Now let's add a record to create the root user with the password password:

```
psql> INSERT INTO users VALUES (
->    'root',
->    '5f4dcc3b5aa765d61d8327deb882cf99',
->    'admin',
->    'System',
->    'Administrator',
->    'root@mymarket.com',
->    '555−5555',
->    '123 5 Avenue'
-> );
```

Notice the password looks a bit wierd, **5f4dcc3b5aa765d61d8327deb882cf99**. This is the MD5 hash of the the word "password", I won't go into details here, but the important thing to note is that it's a one−way algorithm and it always produces a 32 character string.

That's it, we have a users table to track our users, and one administrative account so we can try logging in and out of the system using the example tar file (download the example tar file from http://www.devshed.com/Server_Side/PHP/Commerce1).

---

16. Related URLs

Visit following locators which are related to C, C++ −

- Vim color text editor for C++, C http://metalab.unc.edu/LDP/HOWTO/Vim−HOWTO.html
- SQL database server for PHP PostgreSQL at pgsql−howto
- Source code control system CVS HOWTO for C++ programs http://metalab.unc.edu/LDP/HOWTO/CVS−HOWTO.html
- Linux goodies main site http://www.aldev.8m.com Mirror sites are at − http://aldev0.webjump.com, angelfire, geocities, virtualave, 50megs, theglobe, NBCi, Terrashare, Fortunecity, Freewebsites, Tripod, Spree, Escalix, Httpcity, Freeservers.

17. Other Formats of this Document

This document is published in 14 different formats namely − DVI, Postscript, Latex, Adobe Acrobat PDF, LyX, GNU−info, HTML, RTF(Rich Text Format), Plain−text, Unix man pages, single HTML file, SGML (Linuxdoc format), SGML (Docbook format), MS WinHelp format.

This howto document is located at −

- http://www.linuxdoc.org and click on HOWTOs and search for howto document name using CTRL+f or ALT+f within the web−browser.

You can also find this document at the following mirrors sites −

- http://www.caldera.com/LDP/HOWTO
- http://www.linux.ucla.edu/LDP
- http://www.cc.gatech.edu/linux/LDP
- http://www.redhat.com/mirrors/LDP
- Other mirror sites near you (network−address−wise) can be found at http://www.linuxdoc.org/mirrors.html select a site and go to directory /LDP/HOWTO/xxxxx−HOWTO.html

- You can get this HOWTO document as a single file tar ball in HTML, DVI, Postscript or SGML formats from − ftp://www.linuxdoc.org/pub/Linux/docs/HOWTO/other−formats/ and http://www.linuxdoc.org/docs.html#howto

- Plain text format is in: ftp://www.linuxdoc.org/pub/Linux/docs/HOWTO and http://www.linuxdoc.org/docs.html#howto

- Single HTML file format is in: http://www.linuxdoc.org/docs.html#howto

  Single HTML file can be created with command (see man sgml2html) − sgml2html −split 0 xxxxhowto.sgml

- Translations to other languages like French, German, Spanish, Chinese, Japanese are in ftp://www.linuxdoc.org/pub/Linux/docs/HOWTO and http://www.linuxdoc.org/docs.html#howto Any help from you to translate to other languages is welcome.

The document is written using a tool called "SGML−Tools" which can be got from − http://www.sgmltools.org Compiling the source you will get the following commands like

- sgml2html xxxxhowto.sgml (to generate html file)
- sgml2html −split 0 xxxxhowto.sgml (to generate a single page html file)

- sgml2rtf xxxxhowto.sgml (to generate RTF file)
- sgml2latex xxxxhowto.sgml (to generate latex file)

17.1 Acrobat PDF format

PDF file can be generated from postscript file using either acrobat **distill** or **Ghostscript**. And postscript file is generated from DVI which in turn is generated from LaTex file. You can download distill software from http://www.adobe.com. Given below is a sample session:

```
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 &
```

Or you can use Ghostscript command **ps2pdf**. ps2pdf is a work−alike for nearly all the functionality of Adobe's Acrobat Distiller product: it converts PostScript files to Portable Document Format (PDF) files. **ps2pdf** is implemented as a very small command script (batch file) that invokes Ghostscript, selecting a special "output device" called **pdfwrite**. In order to use ps2pdf, the pdfwrite device must be included in the makefile when Ghostscript was compiled; see the documentation on building Ghostscript for details.

17.2 Convert Linuxdoc to Docbook format

This document is written in linuxdoc SGML format. The Docbook SGML format supercedes the linuxdoc format and has lot more features than linuxdoc. The linuxdoc is very simple and is easy to use. To convert linuxdoc SGML file to Docbook SGML use the program **ld2db.sh** and some perl scripts. The ld2db output is not 100% clean and you need to use the **clean_ld2db.pl** perl script. You may need to manually correct few lines in the document.

- Download ld2db program from http://www.dcs.gla.ac.uk/~rrt/docbook.html or from Al Dev site
- Download the cleanup_ld2db.pl perl script from from Al Dev site

The ld2db.sh is not 100% clean, you will get lots of errors when you run

```
        bash$ ld2db.sh file−linuxdoc.sgml db.sgml
        bash$ cleanup.pl db.sgml > db_clean.sgml
        bash$ gvim db_clean.sgml
        bash$ docbook2html db.sgml
```

And you may have to manually edit some of the minor errors after running the perl script. For e.g. you may need to put closing tag < /Para> for each < Listitem>

17.3 Convert to MS WinHelp format

You can convert the SGML howto document to Microsoft Windows Help file, first convert the sgml to html using:

```
        bash$ sgml2html xxxxhowto.sgml     (to generate html file)
        bash$ sgml2html −split 0   xxxxhowto.sgml (to generate a single page html file)
```

Then use the tool HtmlToHlp. You can also use sgml2rtf and then use the RTF files for generating winhelp files.

17.4 Reading various formats

In order to view the document in dvi format, use the xdvi program. The xdvi program is located in tetex−xdvi*.rpm package in Redhat Linux which can be located through ControlPanel | Applications | Publishing | TeX menu buttons. To read dvi document give the command −

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

And resize the window with mouse. To navigate use Arrow keys, Page Up, Page Down keys, also you can use 'f', 'd', 'u', 'c', 'l', 'r', 'p', 'n' letter keys to move up, down, center, next page, previous page etc. To turn off expert menu press 'x'.

You can read postscript file using the program 'gv' (ghostview) or 'ghostscript'. The ghostscript program is in ghostscript*.rpm package and gv program is in gv*.rpm package in Redhat Linux which can be located through ControlPanel | Applications | Graphics menu buttons. The gv program is much more user friendly than ghostscript. Also ghostscript and gv are available on other platforms like OS/2, Windows 95 and NT, you view this document even on those platforms.

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

To read postscript document give the command −

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

You can read HTML format document using Netscape Navigator, Microsoft Internet explorer, Redhat Baron Web browser or any of the 10 other web browsers.

You can read the latex, LyX output using LyX a X−Windows front end to latex.

18. Copyright

Copyright policy is GNU/GPL as per LDP (Linux Documentation project). LDP is a GNU/GPL project. Additional requests are − Please retain the author's name, email address and this copyright notice on all the copies. If you make any changes or additions to this document then you please intimate all the authors of this document.

19. Appendix A PHP examples

19.1 PostgreSQL large object Example

Submitted by: PHP code exchange px@sklar.com To get this file, in the web−browser, save this file as 'Text' type as pgsql_largeobj.lib

```
PX: PHP Code Exchange -
<url name="PostgreSQL" url="http://www.geocities.com/alavoor/HOWTO/pgsql/PostgreSQL-HOWTO.html">
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);
?>
```

19.2 User authentication Example

To get this file, in the web−browser, save this file as 'Text' type as user_pw.lib

From the PHP 3 Manual: Works only if PHP is an Apache module. Instead of simply printing out the $PHP_AUTH_USER and $PHP_AUTH_PW, you would probably want to check the username and password for validity. Perhaps by sending a query to a database, or by looking up the user in a dbm file.

```
<?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>";
        }
?>
```

19.3 Network admin Example

To get this file, in the web−browser, save this file as 'Text' type as network.lib

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

As a web−developer, you're probably used to such lovely tools as ping, whois, nslookup etc. But what when you need one of those utilities at a client's office and have no access to telnet? Good guess. Time to look up the functions in the "Network" section of the PHP manual.

Socket operations:

The most important function there is fsockopen(). Using this function, you can connect to any open port on a server and establish a socket connection with it. The function's syntax is as following:

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

The first two arguments are obvious, the next two are optional and used for error handling. The "errno" and "errstr" should be passed by reference. "Passing by reference" means that the original variable will get modified. Normally, the content of a variable passed to a function wouldn't be modified.

So, you could use this function to open a connection to a webserver and print out the headers:

```
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);
}
```

In this example you see that you can apply any file operations (fread, fwrite etc) to the the pointer you got using the fsockopen() call. Note that the example realizes a HTTP/1.0 client − it won't work with name−based virtual hosts.

Finger: Naturally, you can also open connections to other ports. Writing a small finger client with PHP is trivial therefore. Let's change the example from above to query a finger daemon:

```
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: Querying a whois server uses the same concept:

```
// 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);
}
```

Blocking and non−blocking operations: But there's a problem with all those functions. They work fine if

1. You have a connection with low latency and
2. If the server you're connecting to is up and running.

If not, your script will be busy until it times out. The reason for this is that default socket connections are blocking and don't time out. You can avoid these "hanging scripts" by switching to non−blocking socket operations. The function set_socket_blocking() does just that: it set all operations on a socket (first parameter: socket pointer) to either blocking (second parameter: true) or false (second parameter: false). Using non−blocking operations, the finger function would like like this:

```
        $fp = fsockopen($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);
```

Modifying these 3 functions to use non−blocking socket calls is left as an exercise for you.

20. Appendix B Midgard Installation

RPMs for Midgard from http://www.midgard−project.org/download/binaries currently do not include PostgreSQL , and hence you need to install from the source tar ball file .

Download the Midgard source tarball and read the INSTALL.REDHAT file −

```
bash# cd midgard−lib−1.4beta6
bash# ./configure −−prefix=/usr/local −−with−mysql=/usr/local −−includedir=/usr/include/mysql −−w
bash# make
bash# make install
bash# ldconfig −v | grep −i midga
Copy the header files, just in case make install did not do that..
bash# cp *.h /usr/local/include


bash# cd ../mod_midgard−1.4beta5c
bash# ./configure −−prefix=/usr/local −−with−mysql=/usr/local −−includedir=/usr/include/mysql −−w
bash# make
bash# make install
#modify apache line to correct /usr/.....
bash# vi /etc/httpd/conf/httpd.conf   (or it is /etc/apache/httpd.conf)
bash# /etc/init.d/apache restart
#apache should restart!!!


bash# cd ../midgard−php−1.4beta6
bash# ./configure '−−with−apxs' '−−with−mysql' '−−with−pgsql' '−−with−midgard' −−prefix=/usr/loca

bash# gvim Makefile
And add −I/usr/include/pgsql to INCLUDE variable.

Also add $(INCLUDE) to $(APXS) command as below −
libphp3.so: mod_php3.c libmodphp3−so.a  pcrelib/libpcre.a midgard/libphpmidgard.a
        −@test −f ./mod_php3.c || test −L ./mod_php3.c || $(LN_S) $(srcdir)/mod_php3.c ./mod_php3
        −@test −f ./mod_php3.c || test −h ./mod_php3.c || $(LN_S) $(srcdir)/mod_php3.c ./mod_php3
        $(APXS) −c −o libphp3.so  −I$(srcdir) $(INCLUDE) −I. −I/usr/local/include −I/usr/lib/glib

bash# make
bash# make install
#modify apache line to correct /usr/.....
# and add lines like these −
        LoadModule php4_module        modules/libphp4.so
        AddModule mod_php4.c
        LoadModule php4_module        lib/apache/libphp4.so

        <IfModule mod_php4.c>
                AddType application/x−httpd−php4 .php4
                AddType application/x−httpd−php4 .php
                AddType application/x−httpd−php4−source .phps
                AddType application/x−httpd−php .php
        </IfModule>

bash# vi /etc/httpd/conf/httpd.conf   (or it is /etc/apache/httpd.conf)

bash# /etc/init.d/apache restart
#apache should restart!!!
```

20.1 Testing Midgard PHP Server

To test the installation do this − Create a file in your document root directory. I usually call it info.php and in it put this single line:

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

Then load it up in your browser: http://localhost/info.php

You should see a nice summary page showing all sorts of information about your setup. You probably shouldn't leave this file around on a production server, but for debugging and general info during development, it is very handy.

20.2 Security OpenSSL

You may also need to get the RSA package for to enable SSL encryption from ftp://ftp.deva.net/pub/sources/crypto/rsaref20.1996.tar.Z See also OpenSSL RPM package on Linux cdrom (http://www.openssl.org

If you do not want the SSL to be enabled (or if you face any problem), then download the source RPM of Apache−Midgard and edit the *.spec file and comment out SSL and rebuild the RPM.

21. Appendix C − Debug tool phpcodesite

```
<
?php
/*  phpCodeSite (Idea from CodeSite − Raize Software)
*  @version 0.1b − 20001125
*  @author Sébastien Hordeaux − <
marms@marms.com>
*  @licence GNU Public Licence
*  Main site : http://phpcodesite.phpedit.com
*/

/*
** How does it work ?
        Place a CS_EnterMethod() at the beginning of each method/function
        Place a CS_ExitMethod() at the beginning of each method/function
        Use CS_SendError() to log an error message
        Use CS_SendNote() to log a simple note message
        Use CS_SendMessage() to log a message
        To log variables: CS_SendVar & CS_SendArray()
        To see input data (global PHP variables) use CS_SendInputData()
*/


if(defined("FLAG_PHPCODESITE_PHP")) return FALSE;
        define("FLAG_PHPCODESITE_PHP", 1);

//  Start without increment
$CS_Step = 0;

CS_SetEnabled(TRUE);
// CS_SetEnabled(FALSE);

//  Switch between Enable/Disable mode
function CS_SetEnabled($state){
        global $CS_Enabled;
        $CS_Enabled = $state;
        CS_Write($CS_Enabled?"<
        pre>":"<
        /pre>");
}

//  Add a level to the reported items
function CS_IncStep(){
        global $CS_Step;
        $CS_Step++;
}

// Remove a level to the reported items
function CS_DecStep(){
        global $CS_Step;
        $CS_Step−−;
        if($CS_Step <
        0)
                $CS_Step = 0;
}

// Log an item
function CS_Log($msg){
        global $CS_Step;
        for($i = 0; $i <
        $CS_Step; $i++)
                CS_WriteIndent();
        CS_Write($msg);
}

// Write data to the target output
function CS_Write($str){
        global $CS_Enabled;
        if($CS_Enabled)
                echo "$str";
}

// Write an indent block
function CS_WriteIndent(){
        CS_Write("|   ");
}

// Beginning a new method
function CS_EnterMethod($methodName){
        CS_Log("−−> $methodName\n");
        CS_IncStep();
}

// Exit a method
function CS_ExitMethod($methodName){
        CS_DecStep();
        CS_Log("<
        −− $methodName\n");
}

// Log a note
function CS_SendNote($note){
        CS_Log("[O] $note\n");
}

// Send a simple message
function CS_SendMessage($msg){
        CS_Log("[M] $msg\n");
```

```
}

// Log an error
function CS_SendError($msg){
        CS_Log("<
        b>[E] $msg<
        /b>\n");
}

// Log a variable
function CS_SendVar($varName, $value){
        if(is_array($value)){
                CS_SendArray($value, $varName);
        }else{
                CS_Log("[L] $varName = \"$value\"\n");
        }
}

// Write all global variables to the report
function CS_SendInputData(){
        global $HTTP_GET_VARS, $HTTP_POST_VARS, $HTTP_COOKIE_VARS,
                $HTTP_SERVER_VARS, $HTTP_ENV_VARS, $HTTP_SESSION_VARS;
        CS_Write("----------------------------------------------------------\n");
        CS_SendArray($HTTP_GET_VARS, "HTTP_GET_VARS");
        CS_SendArray($HTTP_POST_VARS, "HTTP_POST_VARS");
        CS_SendArray($HTTP_COOKIE_VARS, "HTTP_COOKIE_VARS");
        CS_SendArray($HTTP_SERVER_VARS, "HTTP_SERVER_VARS");
        CS_SendArray($HTTP_ENV_VARS, "HTTP_ENV_VARS");
        CS_SendArray($HTTP_SESSION_VARS, "HTTP_SESSION_VARS");
        CS_Write("----------------------------------------------------------\n");
}

// Log an array
function CS_SendArray($array, $arrayStr = ""){
        if(!empty($arrayStr))
                CS_Log("\$$arrayStr");
        if(count($array) == 0){
                CS_Log(" = Array()\n");
        }else{
                CS_Write(" = Array(\n");
                while(list($key2, $value2) = each($array)){
                        CS_WriteIndent();
                        if(empty($arrayStr))
                        CS_WriteIndent();
                        CS_Log("$key2");
                        if(!is_array($value2))
                                CS_Write(" => ".htmlentities($value2)."\n");
                        else
                                CS_SendArray($value2);
                }
                CS_WriteIndent();
                if(empty($arrayStr))
                        CS_WriteIndent();
                CS_Log(")\n");
        }
}
?>
```