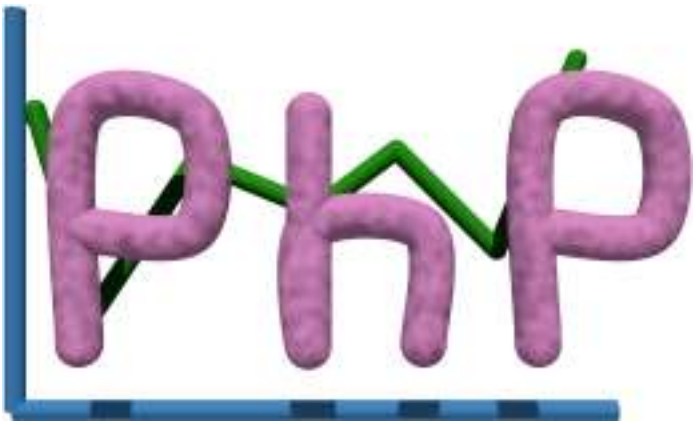
PHP for Dynamic Web Pages



by Jerry Stratton
Friday, September 12, 2008

http://www.hoboes.com/NetLife/PHP/
http://www.hoboes.com/NetLife/PHP/PHP.pdf

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.1, published by the Free Software Foundation. A copy of the license is included in the section entitled “GNU Free Documentation License”

Contents

WHY USE PHP? 1

USING PHP ON APACHE AND UNIX 2

The php Extension 2

PHP CODE 3

Functions 3
Containers 3
PHP With Forms 4
POST and GET 5
Conditional HTML 6
PHP With E-Mail 8
A Note About Email 9
PHP With Files 9
Error Reporting 11
Write Access 11
Arrays 12
Accessing Array Data 13
Don't Trust Anyone Outside 14
Sorting Arrays 15
Visitor Sessions 15
Requiring Cookies 17
Sessions and Security 17
The Program So Far 17
Creating Images On the Fly 18
Creating Your Own Functions 20
Optional Arguments 22

WORKING WITH MYSQL 24

Create the table 24
Create the Database 24
Create the Username and Password 24
Create the Table 24
Store the data 25
Display the data 26

CURLY BRACKETS 28

GNU FREE DOCUMENTATION LICENSE 29

0. Preamble 29
1. Applicability and Definitions 29
2. Verbatim Copying 30
3. Copying in Quantity 30
4. Modifications 31
5. Combining Documents 32
6. Collections of Documents 32
7. Aggregation With Independent Works 32
8. Translation 32
9. Termination 33
10. Future Revisions of this License 33

PHP: HYPERTEXT PROCESSOR 34

Why Use PHP?

If you need to embed dynamic text into static text, you'll find PHP extremely useful. It was designed for this, and it excels at it. PHP is also very useful for integrating web pages with databases.

The PHP scripting language resembles JavaScript, Java, and Perl, These languages all share a common ancestor, the C programming language.

PHP is most different from JavaScript and Java. PHP is a *server-side* scripting language. All of the "work" is done on the server. JavaScript (and Java) generally run on the client. They have little access to the information that the server has, and mediated access to information on the client. They can do lots of things on the client that PHP cannot. PHP has full access to the information that the server has, and very little access to information that the client has. In fact, it only has information that the client tells the server and that the server passes on to PHP. Because it is on the server, however, PHP cannot be modified by the client. While you cannot necessarily trust the information that the client gives to PHP, you can trust that your PHP is doing what you told it to do. Because PHP is on the server end, your PHP scripts can affect your server—such as by keeping an activity log or updating a database.

PHP and Perl often work side-by-side. These are both server-side. Where PHP excels at embedding dynamic content, Perl excels at modifying (or "filtering") streams of text. PHP excels at putting things into documents, and Perl excels at finding things in documents. After you have learned PHP, you may well find Perl useful for many tasks, especially for command-line tasks. PHP has an advantage over Perl on most web sites because PHP is usually loaded as part of the web server. When scripting languages "run", the system has to first load the "interpreter" and then "compile" the language into code that the machine can understand. When you tell PHP to echo the current time to the web page, the computer needs to have your command translated into numbers that it can understand. Because the PHP interpreter is already loaded as part of the web server's software, it is always running. This cuts out half of that process. The interpreter is already loaded, and it can go directly to compiling the language into code. When web servers see a request to run a Perl script, they usually have to first load the Perl interpreter. This happens very quickly, but when there are thousands or tens of thousands of requests coming every second, every "very quickly" can add up.

C programs are "pre-compiled". They cut out both steps in that process: no interpreter is needed because the program is already compiled into code the machine understands. Because of this, however, C programs must be compiled every time you switch to a new machine. If you move to a different host, you will usually have to recompile your C programs. Sometimes you'll even have to recompile your C programs when your ISP upgrades their server's system software. And many ISPs do not provide you with a C compiler. You'll find that PHP is more "portable" than C in this respect: if it works on one server, it will usually work on any other server that has it. Most ISPs that provide server-side scripting provide PHP.

Using PHP on Apache and Unix

You can get more information about PHP, as well as the full PHP manual, at the PHP web site, http://www.php.net/. You can also subscribe to the PHP mailing list there. The PHP on-line manual is extremely useful: not only does it let you quickly look up any part of PHP, but it includes notes from people who use PHP about problems you might run into and how to fix them.

If you want more information about PHP in printed form, there is "Programming PHP" by Rasmus Lerdorf and Kevin Tatroe. Rasmus Lerdorf is one of the authors of the PHP language.

If you are interested in using PHP with the MySQL database there is also "Web Database Applications with PHP & MySQL" by Hugh E. Williams and David Lane, or "MySQL" by Paul DuBois. I've found the latter to be an extremely useful MySQL reference.

The php Extension

If you've used server-side includes, you know that you usually have to rename your web pages from "something.html" to "something.shtml" in order to get the server to "do" the includes. The extension "shtml" lets the server know that it has more work to do on the web page before the reader gets to see it. Similarly, your "PHP" web pages usually need to end in the extension "php". This lets the server know that it needs to hand this page off to the "php" module before letting the reader see the web page.

Some servers will be set up to require an extension of ".php3" instead of ".php". The choice of extension is completely up to whoever sets up your server. While it will usually be "php", it can be anything. Contact your system administrators to be sure.

PHP Code

PHP code starts out looking like HTML code and ends up looking nothing like it. If you've looked at HTML code, you've seen things that look like "" or "<h2>". You will put your PHP code between "<?" and "?>". For example, the following web page will display the current time:

```
<html>
    <head>
        <title>My PHP Page</title>
    </head>
    <body>
        <h1>PHP Test Page</h1>
        <p>The current time is <?echo date('h:i A')?></p>
    </body>
</html>
```

Except for the php part, this looks like a normal web page. The only PHP part is the "<?echo date('h:i A')?>". If you save this file as "test.php" on your web site, and then view it, you will see the current time every time you reload your page.

Functions

PHP uses *functions* to get things done. The functions we just used are the "echo" function and the "date" function. (In Unix, there is no difference between dates and times. They're all stored as one number.) The 'echo' function is simple. Whatever you give it, it 'echoes' to the web page. In this case, we gave it the 'date' function. The 'date' function (http://www.php.net/manual/function.date.php) returns the current date and time in whatever format you want. The format code we used was "h:i A". This means we want the hour, a colon, the minute, a space, and either AM or PM. If it is currently March 9, 10:11 AM and 14 seconds, this will give us "10:11 AM", which the "echo" function inserts as part of the web page.

Containers

Rather than immediately echoing the results of functions, you can store them in containers for later use. (These containers are often called "variables" by programmers.) If, for example, you were going to mention the time more than once on your web page, you might want to store the time in a container and simply echo that container everywhere on your page. Change the body of your web page to:

```
<h1>PHP Test Page</h1>
<?
    $now = date('h:i A');
?>
<p><em>Right now</em> is <?echo $now?>. <?echo $now?> is a very important time,
because <?echo $now?> is the exact time you visited our wonderful web page.</p>
```

Click here to download full PDF material