Using PEAR Classes with PHP for Authentication and Databasing

Sumit Dutta
sumit@southriverhigh.org

June 16, 2005

1 Introduction

1.1 What is PHP/PEAR?

PHP (a recursive acronym just like "GNU" meaning PHP Hypertext Processor) is a server-side language used to implement variable output to server page files (such as HTML documents) with the server without having the client not even know that how the output got generated.

PEAR (the PHP Extension and Application Repository) is used with PHP as certain packages provide particular functions which carry out commonplace website necessities effectively and securely, such as emailing, databasing, and calendar displaying.

When used together, PHP and PEAR create a dynamic duo that can strengthen a website in a multitude of ways. Not only is it a faster way of setting up services for one's website, but it also provides as a way to enhance security, as the packages in PEAR can be automatically updated to a latest version which has been enhanced to fix security or technical bugs.

1.2 Introductory Coding

Prior to actually using PEAR packages, one must be relatively fluent with the PHP language.

The syntax of PHP is very similar to that of C or Perl. There are dollar signs in front of the variable names. Variables are usually of a mixed type, meaning that

they are not particularly integers (whole numbers, positive or negative), strings (words and/or other characters), or doubles (decimal numbers).

The point is, however, that variables never have to be declared because of the multiuse mixed type they are assumed to have. Here is a simple piece of PHP code explained below, which would be expected to be found in a PHP file (one that ends with the extension `.php`:

```
<?php
$var=''<h1>Welcome to my website!</h1>'';
$hello=''<p>We hope you enjoy your stay.  This text
  was generated with PHP.</p>'';
$var.=$hello;
echo $var;
?>
```

All this does is simply assign a heading 1 welcome text to a variable named `var`. The second statement assigns some paragraph text to the variable `hello`. Then, the variable `hello` is concatenated (added on to) variable var. Note that this is the same as saying `$var=$var.$hello`.

To learn more on this topic, simply give the PHP documentation a visit online at http://www.php.net/manual. Here are some basic functions you would need to know to effectively PHP:

`echo $variable`: output text from variable `variable if (condition)`: conditional validator (don't forget to have two equal signs just as in C if using an equality statement) `phpinfo`: show PHP information (never use on an actual website) `str_replace ()`: replace string with part

These are only a few functions. See the PHP manual's function list for the countless functions available.

2 PEAR Class Implementation

2.1 Classes

Classes may be used with PHP just as they are used with other object-oriented languages such as C++ and Java. Classes are generally just a grouping of objects

(variables, functions, and anything else that can be selected and manipulated). To use a class class { and end with an ending brace (}). See the PHP manual again for more detailed information on this.

2.2 Packages

PEAR utilities come in the form of packages which provide classes which provide objects which can be used for manipulation. These packages can be listed from the server shell by entering `pear list`, which lists the package names, versions, and stabilities.

Here are some other PEAR command-line parameters that can be used to change the packages available on the server:

`pear install package.tgz`: install the remotely available package `package.tgz` `pear remote-list`: list all packages available for download from PEAR's website `pear download package`: download the package `package` `pear uninstall package`: uninstall the package `package` `pear upgrade package`: upgrade the package `package` to the latest available version

Note that the commands `install`, `download`, and `upgrade` can be all-recursive (acts on every single available package) by appending to the command `-all` (e.g. `upgrade-all`).

2.3 PHP Codework

To put in objects provided by a PHP class, the following code gives a generic form of implementing the package options. The code should provide an example of implementing predefined variables, functions, and other objects.

```
<?php
require_once ''Package.php'';
$var=Package::function($inputvar);
echo $var->packageprovidedfunction();
?>
```

Cox, Tomas. "Quick Start Guide to PEAR DB." 03 Mar. 2003. Vulcanonet. 16 Jun. 2005 ¡http://vulcanonet.com/soft/?pack=pear_tut¿.
"PHP Manual." 16 Jun. 2005. PHP. 16 Jun. 2005 ¡http://www.php.net/manual/en/¿.
"PHP PEAR Manual." 12 Jun. 2005. PEAR. 16 Jun. 2005 ¡http://pear.php.net/manual/en/¿.