

## **Advanced PHP Programming**

**Copyright © 2004 by Sams Publishing**

International Standard Book Number: 0-672-32561-6

### **Warning and Disclaimer**

Every effort has been made to make this book as complete and as accurate as possible, but no warranty or fitness is implied. The information provided is on an "as is" basis. The author and the publisher shall have neither liability nor responsibility to any person or entity with respect to any loss or damages arising from the information contained in this book.

When reviewing corrections, always check the print number of your book. Corrections are made to printed books with each subsequent printing. To determine the printing of your book, view the copyright page. The print number is right-most number on the line below the "First Printing" line. For example, the following indicates the 4<sup>th</sup> printing of a title.

*First Printing: March 2004*

*06 05 04 03      10 9 8 7 6 5 4*

Misprint	Correction
Page 14, next to last line of code before "Control Flow Constructs" WHERE <b>u</b> .dept id = d.dept id	WHERE <b>e</b> .dept id = d.dept id
Page 15, first three code blocks Change <b>echo</b> to <b>print</b>	
Page 16, third code block <b>Function</b> hello (\$name)	<b>function</b> hello (\$name)
Page 17, first full code block <b>If</b> (( \$number % 2) != 0) }	<b>if</b> (( \$number % 2) != 0) }
Page 17, second code block for(\$I=0; \$i < \$number; \$i++){	For(\$I=3; \$i < \$number; \$i++){
Page 23, first code block function clean cache(\$expiration time)	function clean cache(\$expiration time) {
Page 23, second code block, fifth line my function(\$parent[\$parent+index][\$schild index]);	my function(\$parent[\$parent index][\$schild index]);
Page 24, code block <b>echo</b> "Hello \$name";	<b>print</b> "Hello \$name";
Page 26, last line on page <b>Mysql</b> query(\$second query)	<b>mysql</b> query(\$second query)
Page 27, second head Avoiding Using Open Tags	Avoiding Using <b>Short</b> Open Tags

Page 27, second and sixth lines of code <code>echo "Hello \$username";</code>	<code>print "Hello \$username";</code>
Page 28, fourth line of code <code>&lt;tr&gt;&lt;td&gt;&lt;? echo \$employee['name']</code>	<code>&lt;tr&gt;&lt;td&gt;&lt;?php echo \$employee['name']</code>
Page 28, second line in third code block <code>if((( \$year % 4 == 0 ) &amp;&amp;</code>	<code>if((( \$year % 4 == 0 ) &amp;&amp;</code>
Page 30, second code block <code>// Use the bitwise "AND" <b>operator</b>est to see if the first bit in \$i is set</code>	<code>// Use the bitwise "AND" <b>operator</b> to see if the first bit in \$i is set</code>
Page 32, ninth line in second code block <code>* echo "\$i is prime/n";</code>	<code>* print "\$I is prime/n";</code>
Page 38, first code block <code>echo hello(\$name); echo "You are ".age(\$bday)." years old.\n"; echo goodbye (\$name);</code>	<code>print hello(\$name); print "You are ".age(\$bday)." years old.\n"; print goodbye (\$name);</code>
Page 38, second paragraph in "Introduction to OO Programming" A class constructor in PHP5 should be named <b>contstructor()</b> so that the engine knows how to identify it.	A class constructor in PHP5 should be named <b>contstruct()</b> so that the engine knows how to identify it.
Page 39, end of first code block <code>echo \$user-&gt;hello(); echo "You are ".\$user-&gt;age()." years old.\n"; echo \$user-&gt;goodbye();</code>	<code>print \$user-&gt;hello(); print "You are ".\$user-&gt;age()." years old.\n"; print \$user-&gt;goodbye();</code>

Page 43, middle of page ...one in the constructor, one during the assignment of the return value from the constructor to <code>\$copy</code> , and one when you assign <code>\$copy</code> to <code>\$obj</code> .	...one in the constructor, one during the assignment of the return value from the constructor to <code>\$copy</code> , and one when you assign <code>\$obj</code> to <code>\$copy</code> .
Page 43, last paragraph Inside the <code>clone()</code> method, you not only have <code>\$this</code> , which represents the new object, but also <code>\$that</code> , which is the object being cloned.	Inside the <code>clone()</code> method, you have <code>\$this</code> , which is the new object with all the original object's properties already copied.
Page 44, code block remove <code>\$this-&gt;other = \$that-&gt;other;</code> line	
Page 53, last code block class B {	class B { <b>extends A</b> {
Page 58, first code block <b>echo</b> '<a href="mailto:\$result->email">\$result->name</a>';	<b>print</b> '<a href="mailto:\$result->email">\$result->name</a>';
Page 62, second line of first code block <code>\$a-&gt;counter = 0;</code>	<code>\$a-&gt;counter = 1;</code>
Page 63 SPL and <b>Iterators</b>	SPL and <b>Iterators</b>
Page 65, sixth line of code function <b>hasMore()</b> ;	function <b>valid()</b> ;
Page 65, second line of second code block <code>\$iter-&gt;hasMore()</code> ;	<code>\$iter-&gt;valid()</code> ;
Page 66, twelfth line of code function <b>hasMore()</b> {	function <b>valid()</b> {

Page 67, second code block function <b>offsetGet</b> (\$name)	function <b>offsetGet</b> (\$name)
Page 67, next to last line of code function <b>offsetSet</b> (\$name, \$value)	function <b>offsetSet</b> (\$name, \$value)
Page 68, first six lines of code function offsetUnset (\$name) { return dba delete(\$name, \$this->dbm); } return dba replace(\$name, serialize(\$value), \$this->dbm); }	return dba replace(\$name, serialize(\$value), \$this->dbm); } function offsetUnset (\$name) { return dba delete(\$name, \$this->dbm); }
Page 68, first code block function current() {  function next() {  function has More() {  function key() {	function current() {  function next() {  function has More() {  function key() {
Page 74, third code block <\$	<\$php

Page 79, first code block <b>echo</b> "FATAL error \$msg at \$filename:\$linenum ";  <b>echo</b> "unknown error at \$filename:\$linenumb ";	<b>print</b> "FATAL error \$msg at \$filename:\$linenum ";  <b>print</b> "unknown error at \$filename:\$linenumb ";
Page 82, last code block <b>echo</b> "Your Web server user probably shouldn't be shelled.\m";  <b>echo</b> "Great!\n";  <b>echo</b> "An error occurred checking the user\n";	<b>print</b> "Your Web server user probably shouldn't be shelled.\m";  <b>print</b> "Great!\n";  <b>print</b> "An error occurred checking the user\n";
Page 87, both code blocks change all occurrences of <b>echo</b> to <b>print</b>	
Page 99, last line on page header("Location: /login/php");	header("Location: /login.php");
Page 110, second block of code <b>echo</b> \$ Get['name'];  <b>echo</b> "Stranger";	<b>print</b> \$ Get['name'];  <b>print</b> "Stranger";
Page 111, middle of second code block Hello <\$php <b>echo</b> ?this-> tpl vars['name']; ?>	Hello <\$php <b>print</b> ?this-> tpl vars['name']; ?>
Page 120, last code block <title><?php <b>echo</b> \$template->title ?></title>  Hello <?php <b>echo</b> \$template->name ?>!	<title><?php <b>print</b> \$template->title ?></title>  Hello <?php <b>print</b> \$template->name ?>!

Page 121, third code block <head><title><?php <b>echo</b> ?template->title ?></title>	<head><title><?php <b>print</b> ?template->title ?></title>
Page 144, code block two instances of <b>Reflection Class</b> should be <b>ReflectionClass</b>	
Page 149, Note The constructor uses the <b>Reflection Class</b> class to introspect the service and logger classes before you try to instantiate them.	The constructor uses the <b>ReflectionClass</b> class to introspect the service and logger classes before you try to instantiate them.
Page 156, second line of first full code block require once 'PHPUnit/Framework/Test <b>Class</b> .php';	require once 'PHPUnit/Framework/Test <b>Case</b> .php';
Page 156, first line of second full code block require <b>once</b> "PHPUnit/Framework/TestSuite";	require <b>once</b> "PHPUnit/Framework/TestSuite. <b>php</b> ";
Page 156, first code line after "After you have done this, you run the test:" require once "PHPUnit/TextUI/TestRunner";	require once "PHPUnit/TextUI/TestRunner. <b>php</b> ";
Page 161, sixth code line from bottom foreach( <b>get declared classes()</b> as \$class) {	foreach( <b>get declared classes()</b> as \$class) {
Page 162, first line of code foreach( <b>get declared classes()</b> as \$class) {	foreach( <b>get declared classes()</b> as \$class) {
Page 165, immediately before "Using the setUp() and tearDown() Methods The following <b>examples</b> generates a success:	The following <b>example</b> generates a success:
Page 209, next to last line of code return Commerce calculate <b>Tax</b> (\$user->state, \$price);	return Commerce <b>calculateStateTax</b> (\$user->state, \$price);

Page 235, first line of code Content-Encoding: gzip, <b>defalte</b>	Content-Encoding: gzip, <b>deflate</b>
Page 243, code in Note <b>echo</b> "Hello World";	<b>print</b> "Hello World";
Page 244, code in Note at top of page <b>echo</b> "Hello World";	<b>print</b> "Hello World";
Page 264, next to last line of code protected <b>\$expiration</b> ;	protected <b>\$expiration</b> ;
Page 331 Checking That \$ Server[ <b>REMOTE IP</b> ] Stays the Same	Checking That \$ Server[' <b>REMOTE IP</b> '] Stays the Same
Page 335, fourth line of code <b>set cookie</b> (self::\$cookiename, \$cookie);	<b>setcookie</b> (self::\$cookiename, \$cookie);
Page 335, middle of page \$cookie = implode( <b>\$glue</b> , \$buffer);  explode( <b>\$glue</b> , \$buffer);	\$cookie = implode( <b>self::\$glue</b> , \$buffer);  explode( <b>self::\$glue</b> , \$buffer);
Page 395, first code block <b>echo</b> "Error\n";	<b>print</b> "Error\n";
Page 423, first paragraph Here is a sample run against my Web log, in which I've specified <b>10,000</b> requests with a concurrency of 100 requests.	Here is a sample run against my Web log, in which I've specified <b>1,000</b> requests with a concurrency of 100 requests.

Page 431, second paragraph in "Installing and Using APD" After <b>ADP</b> is installed, you should enable it by setting the following in your php.ini file:	After <b>APD</b> is installed, you should enable it by setting the following in your php.ini file:
Page 433, code in "A Tracing Example" section <b>echo</b> "Hello \$name\n";  <b>echo</b> "Goodbye \$name\n";	<b>print</b> "Hello \$name\n";  <b>print</b> "Goodbye \$name\n";

This errata sheet is intended to provide updated technical information. Spelling and grammar misprints are updated during the reprint process, but are not listed on this errata sheet.