

# PHP/MySQL Tutorial

## Introduction to Database

# Goal of this tutorial

- Not to teach everything about PHP, but provide the basic knowledge
- Explain code examples
- Provide some useful references

# What is PHP?

- **PHP == 'Hypertext Preprocessor'**
- **Open-source, server-side scripting language**
- **Used to generate dynamic web-pages**
- **PHP scripts reside between reserved PHP tags**
  - This allows the programmer to embed PHP scripts within HTML pages

# What is PHP (cont'd)

- Interpreted language, scripts are parsed at run-time rather than compiled beforehand
- Executed on the server-side
- Source-code not visible by client
  - 'View Source' in browsers does not display the PHP code
- Various built-in functions allow for fast development
- Compatible with many popular databases

# What does PHP code look like?

- Structurally similar to C/C++
- Supports procedural and object-oriented paradigm (to some degree)
- All PHP statements end with a semi-colon
- Each PHP script must be enclosed in the reserved PHP tag

```
<?php  
...  
?>
```

# Comments in PHP

- Standard C, C++, and shell comment symbols

```
// C++ and Java-style comment

# Shell-style comments

/* C-style comments
   These can span multiple lines */
```

# Variables in PHP

- PHP variables must begin with a “\$” sign
- Case-sensitive (`$Foo != $foo != $fOo`)
- Global and locally-scoped variables
  - Global variables can be used anywhere
  - Local variables restricted to a function or class
- Certain variable names reserved by PHP
  - Form variables (`$_POST`, `$_GET`)
  - Server variables (`$_SERVER`)
  - Etc.

# Variable usage

```
<?php
$foo = 25;           // Numerical variable
$bar = "Hello";      // String variable

$foo = ($foo * 7);   // Multiplies foo by 7
$bar = ($bar * 7);   // Invalid expression
?>
```

# Echo

- The PHP command '**echo**' is used to output the parameters passed to it
  - The typical usage for this is to send data to the client's web-browser
- Syntax
  - void **echo** (string arg*1* [, string arg*n*...])
  - In practice, arguments are not passed in parentheses since **echo** is a language construct rather than an actual function

# Echo example

```
<?php
$foo = 25;           // Numerical variable
$bar = "Hello";      // String variable

echo $bar;           // Outputs Hello
echo $foo,$bar;      // Outputs 25Hello
echo "5x5=", $foo;   // Outputs 5x5=25
echo "5x5=$foo";     // Outputs 5x5=25
echo '5x5=$foo';     // Outputs 5x5=$foo
?>
```

- **Notice** how echo '5x5=\$foo' outputs \$foo rather than replacing it with 25
- Strings in single quotes (' ') are not interpreted or evaluated by PHP
- This is true for both variables and character escape-sequences (such as "\n" or "\\")

# Arithmetic Operations

```
<?php  
    $a=15;  
    $b=30;  
    $total=$a+$b;  
    Print $total;  
    Print "<p><h1>$total</h1>";  
    // total is 45  
?>
```

- \$a - \$b // subtraction
- \$a \* \$b // multiplication
- \$a / \$b // division
- \$a += 5 // \$a = \$a+5 Also works for \*= and /=

# Concatenation

- Use a period to join strings into one.

```
<?php  
$string1="Hello";  
$string2="PHP";  
$string3=$string1 . " " . $string2;  
Print $string3;  
?>
```

Hello PHP

# Escaping the Character

- If the string has a set of double quotation marks that must remain visible, use the \ [backslash] before the quotation marks to ignore and display them.

```
<?php  
$heading="\"Computer Science\"";  
Print $heading;  
?>
```

“Computer Science”

# PHP Control Structures

- Control Structures: Are the structures within a language that allow us to control the flow of execution through a program or script.
- Grouped into conditional (branching) structures (e.g. if/else) and repetition structures (e.g. while loops).
- Example if/else if/else statement:

```
if ($foo == 0) {  
    echo 'The variable foo is equal to 0';  
}  
else if (($foo > 0) && ($foo <= 5)) {  
    echo 'The variable foo is between 1 and 5';  
}  
else {  
    echo 'The variable foo is equal to '.$foo;  
}
```

# If ... Else...

- if (condition)

```
{
```

    Statements;

```
}
```

else

```
{
```

    Statement;

```
}
```

```
<?php  
if($user=="John")  
{  
    Print "Hello John. ";  
}  
else  
{  
    Print "You are not John. ";  
}  
?>
```

# While Loops

- while (condition)

```
{
```

    Statements;

```
}
```

```
<?php  
$count=0;  
while ($count<3)  
{  
    Print "hello PHP. ";  
    $count += 1;  
    // $count = $count + 1;  
    // or  
    // $count++;  
?>
```

hello PHP. hello PHP. hello PHP.

# Date Display

2009/4/1

```
$datedisplay=date("yyyy/m/d");
Print $datedisplay;
# If the date is April 1st, 2009
# It would display as 2009/4/1
```

Wednesday, April 1, 2009

```
$datedisplay=date("l, F m, Y");
Print $datedisplay;
# If the date is April 1st, 2009
# Wednesday, April 1, 2009
```

# Month, Day & Date Format Symbols

M	Jan
F	January
m	01
n	1

Day of Month	d	01
Day of Month	J	1
Day of Week	I	Monday
Day of Week	D	Mon

# Functions

- Functions MUST be defined before they can be called
- Function headers are of the format

```
function functionName($arg_1, $arg_2, ..., $arg_n)
```

- Note that no return type is specified

# Functions example

```
<?php
    // This is a function
    function foo($arg_1, $arg_2)
    {
        $arg_2 = $arg_1 * $arg_2;
        return $arg_2;
    }

    $result_1 = foo(12, 3);          // Store the function
    echo $result_1;                // Outputs 36
    echo foo(12, 3);              // Outputs 36
?>
```

# Include Files

Include "opendb.php";

Include "closedb.php";

This inserts files; the code in files will be inserted into current code.

This will provide useful and protective means once you connect to a database, as well as for other repeated functions.

**Include ("footer.php");**

The file footer.php might look like:

```
<hr SIZE=11 NOSHADE WIDTH="100%">
<i>Copyright © 2008-2010 KSU </i></font><br>
<i>ALL RIGHTS RESERVED</i></font><br>
<i>URL: http://www.kent.edu</i></font><br>
```

# PHP - Forms

- Access to the HTTP POST and GET data is simple in PHP
- The global variables `$_POST[]` and `$_GET[]` contain the request data

```
<?php
    if ($_POST["submit"])
        echo "<h2>You clicked Submit!</h2>";
    else if ($_POST["cancel"])
        echo "<h2>You clicked Cancel!</h2>";
?
<form action="form.php" method="post">
    <input type="submit" name="submit" value="Submit">
    <input type="submit" name="cancel" value="Cancel">
</form>
```

<http://www.cs.kent.edu/~nruan/form.php>

# WHY PHP – Sessions ?

Whenever you want to create a website that allows you to store and display information about a user, determine which user groups a person belongs to, utilize permissions on your website or you just want to do something cool on your site, PHP's Sessions are vital to each of these features.

Cookies are about 30% unreliable right now and it's getting worse every day. More and more web browsers are starting to come with security and privacy settings and people browsing the net these days are starting to frown upon Cookies because they store information on their local computer that they do not want stored there.

PHP has a great set of functions that can achieve the same results of Cookies and more without storing information on the user's computer. PHP Sessions store the information on the web server in a location that you chose in special files. These files are connected to the user's web browser via the server and a special ID called a "Session ID". This is nearly 99% flawless in operation and it is virtually invisible to the user.

# PHP - Sessions

- Sessions store their identifier in a cookie in the client's browser
- Every page that uses session data must be proceeded by the **session\_start()** function
- Session variables are then set and retrieved by accessing the global **\$\_SESSION[]**
- Save it as **session.php**

```
<?php
    session_start();
    if (!$_SESSION["count"])
        $_SESSION["count"] = 0;
    if ($_GET["count"] == "yes")
        $_SESSION["count"] = $_SESSION["count"] + 1;
    echo "<h1>".$_SESSION["count"]."</h1>";
?>
<a href="session.php?count=yes">Click here to count</a>
```

**http://www.cs.kent.edu/~nruan/session.php**

# Avoid Error PHP - Sessions

**PHP Example:** <?php  
echo "Look at this nasty error below:<br />";  
session\_start();  
?>  
Error!

**Warning: Cannot send session cookie - headers already sent by  
(output started at session\_header\_error/session\_error.php:2)  
in session\_header\_error/session\_error.php on line 3**

**Warning: Cannot send session cache limiter - headers already  
sent (output started at  
session\_header\_error/session\_error.php:2) in  
session\_header\_error/session\_error.php on line 3**

**PHP Example:** <?php  
session\_start();  
echo "Look at this nasty error below:";  
?>  
Correct

# Destroy PHP - Sessions

## Destroying a Session

why it is necessary to destroy a session when the session will get destroyed when the user closes their browser. Well, imagine that you had a session registered called "access\_granted" and you were using that to determine if the user was logged into your site based upon a username and password. Anytime you have a login feature, to make the users feel better, you should have a logout feature as well. That's where this cool function called session\_destroy() comes in handy. session\_destroy() will completely demolish your session (no, the computer won't blow up or self destruct) but it just deletes the session files and clears any trace of that session.

**NOTE:** If you are using the `$_SESSION` superglobal array, you must clear the array values first, then run session\_destroy.

Here's how we use session\_destroy:

# Destroy PHP - Sessions

```
<?php
// start the session
session_start();
header("Cache-control: private"); //IE 6 Fix
$_SESSION = array();
session_destroy();
echo "<strong>Step 5 - Destroy This Session </strong><br />";
if($_SESSION['name']){
    echo "The session is still active";
} else {
    echo "Ok, the session is no longer active! <br />";
    echo "<a href=\"page1.php\"><< Go Back Step 1</a>";
}
?>
```

[http://www.cs.kent.edu/~nruan/session\\_destroy.php](http://www.cs.kent.edu/~nruan/session_destroy.php)

# PHP Overview

- Easy learning
- Syntax Perl- and C-like syntax. Relatively easy to learn.
- Large function library
- Embedded directly into HTML
- Interpreted, no need to compile
- Open Source server-side scripting language designed specifically for the web.

# PHP Overview (cont.)

- Conceived in 1994, now used on +10 million web sites.
- Outputs not only HTML but can output XML, images (JPG & PNG), PDF files and even Flash movies all generated on the fly. Can write these files to the file system.
- Supports a wide-range of databases (20+ODBC).
- PHP also has support for talking to other services using protocols such as LDAP, IMAP, SNMP, NNTP, POP3, HTTP.

# First PHP script

## ■ Save as sample.php:

```
<!-- sample.php -->
<html><body>

    <strong>Hello World!</strong><br />
    <?php
        echo "<h2>Hello, World</h2>"; ?>
    <?php
        $myvar = "Hello World";
        echo $myvar;
    ?>
</body></html>
```

<http://www.cs.kent.edu/~nruan/sample.php>

# Example – show data in the tables

- Function: list all tables in your database.  
Users can select one of tables, and show all contents in this table.
- second.php
- showtable.php

<http://www.cs.kent.edu/~nruan/second.php>

# second.php

```
<html><head><title>MySQL Table Viewer</title></head><body>
<?php
// change the value of $dbuser and $dbpass to your username and password
$dbhost = 'hercules.cs.kent.edu:3306';
$dbuser = 'nruan';
$dbpass = '*****';
$dbname = $dbuser;
$table = 'account';
$conn = mysql_connect($dbhost, $dbuser, $dbpass);
if (!$conn) {
    die('Could not connect: ' . mysql_error());
}
if (!mysql_select_db($dbname))
    die("Can't select database");
```

**Choose one table:**

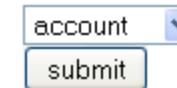
A screenshot of a web browser showing a dropdown menu with the option "account" selected and a "submit" button below it.

# second.php (cont.)

```
$result = mysql_query("SHOW TABLES");
if (!$result) {
    die("Query to show fields from table failed");
}
$num_row = mysql_num_rows($result);
echo "<h1>Choose one table:</h1>";
echo "<form action=\"showtable.php\" method=\"POST\">";
echo "<select name=\"table\" size=\"1\" Font size=\"+2\">";
for($i=0; $i<$num_row; $i++) {
    $tablename=mysql_fetch_row($result);
    echo "<option value=\"{$tablename[0]}\" >{$tablename[0]}</option>";
}
echo "</select>";
echo "<div><input type=\"submit\" value=\"submit\"></div>";
echo "</form>";

mysql_free_result($result);
mysql_close($conn);
?>
</body></html>
```

**Choose one table:**



A screenshot of a web browser interface. On the left is a dropdown menu with the word "account" followed by a downward arrow. To its right is a blue rectangular button with the word "submit" in white.

# showtable.php

```
<html><head>
<title>MySQL Table Viewer</title>
</head>
<body>
<?php
$dbhost = 'hercules.cs.kent.edu:3306';
$dbuser = 'nruan';
$dbpass = '*****';
$dbname = 'nruan';
$table = $_POST["table"];
$conn = mysql_connect($dbhost, $dbuser, $dbpass);
if (!$conn)
    die('Could not connect: ' . mysql_error());
if (!mysql_select_db($dbname))
    die("Can't select database");
$result = mysql_query("SELECT * FROM {$table}");
if (!$result) die("Query to show fields from table failed!" . mysql_error());
```

**Table: account**

account_number	branch_name	balance
A-101	Downtown	500
A-102	Perryridge	400
A-201	Brighton	900
A-215	Mianus	700
A-217	Brighton	750
A-222	Redwood	700
A-305	Round Hill	350
A-111	Perryridge	900

# showtable.php (cont.)

```
$fields_num = mysql_num_fields($result);
echo "<h1>Table: {$table}</h1>";
echo "<table border='1'><tr>";
// printing table headers
for($i=0; $i<$fields_num; $i++) {
    $field = mysql_fetch_field($result);
    echo "<td><b>{$field->name}</b></td>";
}
echo "</tr>\n";
while($row = mysql_fetch_row($result)) {
    echo "<tr>";
    // $row is array... foreach( .. ) puts every element
    // of $row to $cell variable
    foreach($row as $cell)
        echo "<td>$cell</td>";
    echo "</tr>\n";
}
mysql_free_result($result);
mysql_close($conn);
?>
</body></html>
```

**Table: account**

account_number	branch_name	balance
A-101	Downtown	500
A-102	Perryridge	400
A-201	Brighton	900
A-215	Mianus	700
A-217	Brighton	750
A-222	Redwood	700
A-305	Round Hill	350
A-111	Perryridge	900

# Functions Covered

- `mysql_connect()`      `mysql_select_db()`
- `include()`
- `mysql_query()`      `mysql_num_rows()`
- `mysql_fetch_array()`    `mysql_close()`

# History of PHP

- PHP began in 1995 when Rasmus Lerdorf developed a Perl/CGI script toolset he called the Personal Home Page or PHP
- PHP 2 released 1997 (PHP now stands for Hypertext Processor). Lerdorf developed it further, using C instead
- PHP3 released in 1998 (50,000 users)
- PHP4 released in 2000 (3.6 million domains). Considered debut of functional language and including Perl parsing, with other major features
- PHP5.0.0 released July 13, 2004 (113 libraries>1,000 functions with extensive object-oriented programming)
- PHP5.0.5 released Sept. 6, 2005 for maintenance and bug fixes

# Recommended Texts for Learning PHP

- Larry Ullman's books from the Visual Quickpro series
- PHP & MySQL for Dummies
- Beginning PHP 5 and MySQL: From Novice to Professional by W. Jason Gilmore
  - (This is more advanced and dense than the others, but great to read once you've finished the easier books. One of the best definition/description of object oriented programming I've read)

# PHP References

- <http://www.php.net> <-- php home page
- <http://www.phpbuilder.com/>
- <http://www.devshed.com/>
- <http://www.phpmyadmin.net/>
- <http://www.hotscripts.com/PHP/>
- <http://geocities.com/stuprojects/ChatroomDescription.htm>
- <http://www.academic.marist.edu/~kbhkj/chatroom/chatroom.htm>
- <http://www.aus-etrade.com/Scripts/php.php>
- <http://www.codeproject.com/asp/CDIChatSubmit.asp>
- <http://www.php.net/downloads> <-- php download page
- <http://www.php.net/manual/en/install.windows.php> <-- php installation manual
- <http://php.resourceindex.com/> <-- PHP resources like sample programs, text book references, etc.
- <http://www.daniweb.com/techtalkforums/forum17.html> ← php forums

# Create your own homepage

- Login loki.cs.kent.edu
- Create directory “public\_html” in your home directory
- Create two php files (second.php and showtable.php) we have discussed
- Visit your homepage:

[http://www.cs.kent.edu/~\[username\]/second.php](http://www.cs.kent.edu/~[username]/second.php)

**<http://www.cs.kent.edu/~nruan/second.php>**