Web Programming

01 – PHP Fundamentals and State

Aryo Pinandito, ST, M.MT

Web Server Administration

- Web Server: Apache HTTP Server
- Why Apache?
  - Extremely popular, ±45% web servers on the Internet.
  - Straightforward method of configuration, uses flat configuration files. – a `httpd.conf` file
- Often paired with the most popular server-side scripting engine and database,
  - PHP, and
  - MySQL database

Apache Web Server Configuration

`httpd.conf` > Configuration: *directive*

- Server Root
- Document Root
- Listen Port
- Loaded Modules (`LoadModule`)
  - PHP
- Directory Index
  - index.php
- Virtual Hosts? SSL? Manual? Languages?

PHP

- A programming language devised by Rasmus Lerdorf in 1994 to build a dynamic and interactive web sites.
- Formerly named from Personal Home Page but changed to a recursively named:
  - PHP: Hypertext Preprocessor
- PHP programs (.php) are run on a server—specifically run on a Web server.
  - OR... you may also run it manually through the shell
- PHP often used as a middle-ware to other services on the internet (e.g accessing data on a database or generating documents on the fly)

Why PHP?

- Cross-platform:
  - Most PHP code can be processed without alteration on computers running many different operating systems.
  - For example, a PHP script that runs on Linux generally also runs well on Windows.
- HTML-embedded:
  - PHP code can be written in files containing a mixture of PHP instructions and HTML code.
  - C-based programming language
- Open source
  - You don't have to pay in using PHP code to build dynamic websites.

System and Software Requirements

- To run PHP code you will need the following software:
  - A computer with an operating system such as Windows, Mac, or Linux
  - A PHP-compatible Web server software
    - Apache, Internet Information Server (IIS), or Nginx
  - PHP software
    - Can be downloaded from php.net
- For database environment
  - MySQL Database Server
    - Can be downloaded from http://mysql.com

System and Software Requirements (2)

- Optional development-related software
  - Any text editor, such as Notepad or Notepad++, Emacs, vi.
    - Or… you may use Adobe Dreamweaver IDE or any other PHP script editor with PHP syntax highlighting feature to aid you in fixing common syntax problems that you may encounter during development. *This will help you code PHP script pretty much easier*.
  - Web browsers
    - IE, Mozilla Firefox, Google Chrome, Opera
    - Browser with Firebug or Web Developer plugin installed is recommended.
  - Helpers script, PHPMyAdmin
    - PHP-based visual database management for MySQL.
  - PHP Manuals
    - Downloadable from PHP documentation downloads page:
    - http://php.net/download-docs.php

Warning!
Dragons Ahead

You may want to turn ON your PHP-enabled web server to test any of the following PHP scripts provided

Hello, World!

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

Variables

- Issues concerning creating variables:
  - Naming conventions
  - Data type
  - Scope
- Variables in PHP are very flexible
  - less restriction in using one variables for one or more datatype at one time

```
<?php
  $a = 1;                          // $a as integer
  $a = 'text';                     // $a as string
  $a = array('a', 'b', 'c'); // $a as array
?>
```

Variables Naming

- Variable names begin with a dollar sign (**$**).
- The first character after the dollar sign MUST be a letter or an underscore.
- The remaining characters in the name may be letters, numbers, or underscores without a fixed limit
- Variables are CASE SENSITIVE
  - treat two variables with the same name but with different case as two different variables

```
<?php
  // both are two different variables
  $myVariable = 0;
  $myvariable = 1;
?>
```

Variable of Variable

- PHP allows you to create variable which contains another variable.

```
<?php
  $a = 0;
  $b = 1;
  $var = ‘a’;
  echo $$var; // this line will echo 0
  $var = ‘b’;
  echo $$var; // this line will echo 1
?>
```

PHP Data Types

Data type	Description
Boolean	Scalar; either True or False
Integer	Scalar; a whole number
Float	Scalar; a number which may have a decimal place
String	Scalar; a series of characters
Array	Compound; an ordered map (contains names mapped to values)
Object	Compound; a type that may contain properties and methods
Resource	Special; contains a reference to an external resource, such as a handler to an open file
NULL	Special; may only contain NULL as a value, meaning the variable; explicitly does not contain any value

Common PHP Operators

- Assignment
  - `=`
- Arithmetic
  - `+, -, /, *, %`
- Concatenation
  - `.`
- Negation
  - `!`
- Logic
  - `||, &&, >, <, ==, >=, <=, !=, ===, !===, and, or`
- Increment
  - `++, --`

Variable Scope

- Local Scope
  - Any variable used from inside function

```
<?php

function send_data() {
      $my_data = "Inside data";
      echo $my_data; // echoes $my_data value
}

// throws an error messages
echo $my_data;

?>
```

Variable Scope: Global

- Global Scope
  - Any variable used from outside a function

```
<?php

  $a = 1;
  $b = 2;

  function Sum()
  {
    global $a, $b;
     $b = $a + $b;
  }

  Sum();   // executing Sum() function
  echo $b; // will echo 3

?>
```

Super Global Arrays

Array	Description
$GLOBALS	Has a reference to every variable that has global scope in a PHP program. Many of the variables in it are also in other superglobal arrays
$_SERVER	Includes everything sent by server in the HTTP response, such as the name of the currently executing script, server name, version of HTTP, remote IP address, and so on. Although most Web server software produces the same server variables, not all do, and not all server variables necessarily have data in them
$_GET	Contains all the querystring variables that were attached to the URL, or produced as a result of using the GET method
$_POST	Contains all the submitted form variables and their data. You use variables from the $_POST or $_REQUEST arrays extensively in most of your PHP programs. For example, to make use of a username or password (or any other data) submitted as part of a form, you'll use PHP variables from the $_REQUEST array

Super Global arrays

Array	Description
$_COOKIE	Contains all cookies sent to the server by the browser. They are turned into variables you can read from this array, and you can write cookies to the user's browser using the setcookie() function. Cookies provide a means of identifying a user across page requests (or beyond, depending upon when the cookie expires) and are often used automatically in session handling
$_FILES	Contains any items uploaded to the server when the POST method is used. It's different from the $_POST array because it specifically contains items uploaded (such as an uploaded image file), not the contents of submitted form fields
$_ENV	Contains data about the environment the server and PHP are operating in, such as the computer name, operating system, and system drive
$_REQUEST	Contains the contents of the $_GET, $_POST, and $COOKIE arrays, all in one

Printing Variables

```php
<?php
$x = 20;
$y[] = 10;
$z['name'] = "John Doe";

echo $x;                          // 20
echo "x=$x";                      // x=20
echo 'x=$x';                      // x=$x
echo 'x='.$x;                     // x=20
echo $y;                          // Array
echo $y[0];                       // 10
echo "$y[0]";                     // 10
echo '$y[0]';                     // $y[0]
echo "Name = ".$z['name'];        // Name = John Doe
echo "Name = $z[name]";           // Name = John Doe
echo $z[name];                    // -- throw warning
```

Arrays

- Defining arrays

```php
<?php
  $arr = array('1','2','3');
  $arr[] = '4';
  $arr['name']='John Doe';
  $arr = array('name'=>'John Doe');
?>
```

- Accessing arrays

```php
<?php
  echo $arr[0];       // prints 1
  echo $arr[1];       // prints 2
  echo $arr[2];       // prints 3
  echo $arr[3];       // prints 4
  echo $arr['name']; // prints John Doe
?>
```

Loop: FOR

```php
<?php
for($x = 1; $x <= 10; $x++) {
  echo $x;
}

// will prints 1 to 10
```

Loop: WHILE

```php
<?php
$x = 10;
while( $x > 0 ){
   echo $x;
   $x--;
}

// will prints 10 to 1
```

Loop: DO-WHILE

```php
<?php
$x = 10;

do {
  echo $x;
} while ($x < 9);

// will prints 10, why?
```

Iteration of Array: FOREACH

```php
<?php
$arr = array('name'=>'John', 'age'=>20);

foreach ($arr as $key => $value) {
     echo $key . '=' . $value;
}

// will prints:
// name=John
// age=20
```

Conditional Tests: IF-ELSE

```
<?php
$x = 1;
if($x == 1) {
  // true statement
} else {
  // false statement
}

if($x == 2) :
  // true statement
else:
  // false statement
endif;
?>
```

Conditional Tests: SWITCH-CASE

```
<?php
$x = 1;
switch($x) {
  case 0: echo $x; // do 0 statement
    break;
  case 1: echo $x; // do 1 statement
    break;
  case 2: echo $x; // do 2 statement
  case 3: echo $x; // do 3 statement
    break;
  default: echo $x; // do default statement
    break;
}

// if $x value is 2? What is going to happen?
?>
```

Functions

- Function example in PHP

```php
<?php

function sum($a, $b = 2) {
  // define function content here...
  $v = $a + $b + 1;
  // optionally put a return value
  return $v;
}

// calling the function
$x = sum(4);
echo $x; // will prints 7

?>
```

Sending Variables: Request Method

- GET
  - Sending request variables through an URL as a Query String
- POST
  - Sending request variables through the POST body. Variable name and it's value will not be shown on the URL

Query String

- In the World Wide Web, a **query string** is the part of a Uniform Resource Locator (URL) that contains data to be passed to web applications such as a PHP or CGI programs.
- Each variables data name and value is separated by an ampersand symbol (&)

Example:

```
http://domain.com/index.php?title=Main+Page&action=raw
```

Query String:

```
title=Main+Page&action=raw
```

Building Query String

- Writing a PHP program that generates a query string attached to an URL using the following code
  - (assuming you had the $name variable's value is already set to string 'John'):

```
<?php $name = "John"; ?>
<a href="http://domain.com?name=<?php echo $name; ?>">
  Click Here
</a>
```

- When this code runs, it produces the following output:

```
<a href="http://domain.com?name=John">
  Click Here
</a>
```

Attributes in <form> Elements

- Action Attribute
  - Tells to server which page to go to

```
<form action="myprogram.php">
   ...
</form>
```

- Method Attribute
  - The method attribute controls the way that information is sent to the server.

```
<form action="myprogram.php" method="GET">
```

  - or

```
<form action="myprogram.php" method="POST">
```

GET Method

- Browser automatically appends the information to the URL when it sends the page request to the web server

Example:

```
<form action="test.php" method="GET">
```

- If the form is submitted then the page will be redirected to:

```
http://www.domain.com/test.php?furryanimal=cat&spiky animal=porcupine
```

POST Method

- Information in the form is sent in the body of http request and doesn't appear in the URL

```
<form action="myprogram.php" method="POST">
  <input name="email“ value="name@domain.com"
</form>
```

HTML Standard Form Input Fields

- Text Fields

```
<input type="text" name="text1" />
```

- Password Field

```
<input type="password" name ="pass" />
```

- Radio Buttons

```
<input type="radio" name="radio1" value="Men" />
<input type="radio" name="radio1" value="Women" />
```

- Checkboxes

```
<input type="checkbox" name="vehicle" value="Bike" />
```

- Submit Button

```
<input type="submit" value="Submit" />
```

- Hidden fields

```
<input type="hidden" name="product_id" value="122" />
```

PHP Form Handling

- Get Value

```
<html>
  <body>
    Welcome <?php echo $_GET["text1"]; ?>!<br />
    Your password is <?php echo $_GET["pass"]; ?>.
  </body>
</html>
```

- Post Value

```
<html>
  <body>
    Welcome <?php echo $_POST["text1"]; ?>!<br />
    Your password is <?php echo $_POST["pass"]; ?>.
  </body>
</html>
```

State and Session

- Questions about state:
  - How to keep facebook users keep logged in while browsing friends profiles or other pages?
  - How to keep your shopping cart entries while you are browsing another goods to add?
  - How to keep students previous question answers on an online student examination system?

- How do we keep user state?
  - Cookies
  - Session

COOKIES

- Cookie is a small file that the server embeds on the user's computer.
- Cookie is often used to identify a user (or user's session).
- Variables stored on a cookie is read when users access a website who own those cookie.
- Web sites can usually only modify their own cookies.

COOKIES

- Sets cookies

```
setcookie(name, [value], [expire], [path], [domain]);
```

```
<?php
  setcookie("user", "Alex Porter", time()+3600);
?>
```

- Retrieves cookies

```
$_COOKIE["name of cookie"];
```

```
<?php
  echo $_COOKIE["user"];
?>
```

Session

- With session, users are allowed to store information on the server for later use (i.e username, shopping item, question answer, etc)
- Session information is stored temporarily on the server and will be deleted if it is destroyed or after the user has left the website for a specified time.
- Sessions work by creating a unique id (**PHPSESSID**) for each visitor and store variables based on this **PHPSESSID**.
- While variables contained in a session stored securely on the server, this **PHPSESSID** value is stored on the client computer as a cookie in order to be able to keep track with the client, if cookies are disabled, **PHPSESSID** value is stored in the URL as a query string.

Using Sessions

- Starting session

```
<?php session_start(); ?>
```

- Storing session

```
<?php
  session_start();
  $_SESSION['status']=1;
?>
```

- Retrieving a session variable

```
<?php
  session_start();
  echo “Status=“ . $_SESSION[‘status'];
?>
```

Using Sessions

- Removing one session variable

```
<?php
  session_start();
  if(isset($_SESSION['status']))
    unset($_SESSION['status']);
?>
```

- Destroying the whole user's session

```
<?php
  session_destroy();
?>
```

Questions?