

## CSCI 241

# The PHP Scripting Language (Basics)

Department of CS & QM  
Winthrop University  
Fall 2011

Prepared by X. Wang

## What is PHP?

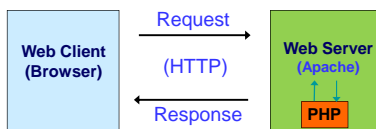
- PHP is a widely-used Open Source general-purpose scripting language that is especially suited for Web development and can be embedded into HTML
- PHP is a recursive acronym for "PHP: Hypertext Preprocessor"
- PHP is a server-side scripting language
- The current version is PHP5 (version 5.3.8)
- The official Web site of PHP is at <http://www.php.net/>
- The latest installation version and documentation can be downloaded from this site

Prepared by X. Wang

2

## Who Executes PHP code?

- When a Web page with embedded PHP code is requested, the Web server executes the PHP script, substitutes in the result back into the page, and then send the page back to the Web browser



Prepared by X. Wang

3

## Who Executes PHP code?

- This also means that to display a Web document with embedded PHP code, we **must pass** the document to the server that has integrated with the PHP engine for interpretation
- If we open the document with a Web browser **locally**, the embedded PHP code will not be executed

Prepared by X. Wang

4

## Name a File including PHP Code

- A Web server with PHP engine is usually configured to identify a document file with an extension name `.php` as PHP file
- If a document file requested by the browser has an extension name `.php`, the Web server will start the PHP engine to execute the PHP script code embedded in the document
- Since a Web server only identifies the `.php` files as PHP document, we must name our PHP document file with the extension name `.php`. Otherwise, the PHP script embedded in the requested file will not be executed

Prepared by X. Wang

5

## PHP Basics

- The following is a simple HTML document with embedded PHP code

```

<html>
<head>
<title>Hello, world</title>
</head>
<body>
<h1>
<?php
    print "Hello, world.";
?>
</h1>
</body>
</html>
  
```

Prepared by X. Wang

6

## PHP Basics

- In the previous example, the PHP code is

```
<?php
    print "Hello, world.";
?>
```

- After executing this code, the Web server will return the following HTML code to the browser

```
<html>
<head>
<title>Hello, world</title>
</head>
<body>
<h1>
    Hello, world.
</h1>
</body>
</html>
```

Prepared by X. Wang

7

## PHP Script Blocks

- We may have the following ways to add PHP code block into an HTML document file

- Use the default begin and end tags `<?php` and `?>`
- Use the shorter tags `<?>` and `?>` (need to enable PHP's `short_open_tag` directive)
- Use the ASP-style tags `<%` and `%>` (need to enable PHP's `asp_tags` directive)
- Some Web server may support the `<script>` tags for PHP:

```
<script language="php">
    . . .
</script>
```

Prepared by X. Wang

8

## PHP Script Blocks

- A PHP block can be embedded at any locations (in the `head` or `body` element, or even before the `html` element) in an HTML document
- When PHP script runs, every block of code, including the start and end tags is replaced with the output of the block

Prepared by X. Wang

9

## Statements and Whitespaces

- Except the last statement, every statement of the PHP code is required to be terminated with a semicolon
- For the last statement, we can end it with a semicolon or without it
- Whitespace has no effect on the code, except to aid readability for the developer
- PHP is also a case-sensitive scripting language

Prepared by X. Wang

10

## Comments

- The following three ways can be used for writing comments

- `// This is a single-line comment`
- `# This is another single-line comment`
- `/* This is a multi-line comment */`

Prepared by X. Wang

11

## Data Types

- PHP is a loosely typed language, allowing variables and function parameters to be set to any type of data
- PHP supports:
  - Four scalar or primitive data types: `boolean`, `integer`, `float` and `string`
  - Two compound data types: `array` and `object`
  - Two special data types: `null` and `resource` (a variable may point to an external resources such as database, files, or network streams. At this time, the variable in the `resource` type is a `reference`.)

Prepared by X. Wang

12

## Variables

- All variables in PHP start with a \$ sign symbol
- Variables don't need to be declared before use them
- Normally we can assign variables a value to create them
- In PHP, the type of a variable is implicitly defined or redefined
- For example: `$num = 20;` creates a variable `$num` with the type of `integer`

Prepared by X. Wang

13

## Value Assignment

- Assignment by value simply involves copying the value of the assigned expression to the variable
- For example:

```
$color = "green";
$num = 1000;
$product = $num * 6;
```

Prepared by X. Wang

14

## Reference Assignment

- PHP can also assign variables by *reference*
- This is to create a variable that refers to the same content (memory) as another variable does
- A change to any variable content will change all other variables contents
- An ampersand (&) is used to represent a reference
- For example:
 

```
$color = "green";
$color1 = &$color;
```

Then both `$color` and `$color1` share the same value

Prepared by X. Wang

15

## Type Conversion

- Variables can be explicitly converted to another type with the following functions

```
string strval(mixed variable);
integer intval(mixed variable [, integer base]);
float floatval(mixed variable);
```

- Examples:

```
$year = 2010;
$yearString = strval($year);
$value = intval($year);
$fvalue = floatval($year);
```

Prepared by X. Wang

16

## Type Casting

- PHP also supports type conversion with *type-casting*
- Type casting operators:
 

Cast Operators	Conversion
<code>(bool)</code> or <code>(boolean)</code>	<code>boolean</code>
<code>(int)</code> or <code>(integer)</code>	<code>integer</code>
<code>(real)</code> or <code>(double)</code> or <code>(float)</code>	<code>float</code>
<code>(string)</code>	<code>string</code>
<code>(array)</code>	<code>array</code>
<code>(object)</code>	<code>object</code>
- Examples:
 

```
$year = 2010;
$yearString = (string)$year;
$fvalue = (float)$year;
```

Prepared by X. Wang

17

## Type Change

- The previous two ways don't change the type of the original variable
- To change the type of the original variable (for example: `$year`), we can use the function:
 

```
boolean settype(mixed variable, string type);
```
- `settype()` explicitly sets the type of `variable` to `type`, where `type` is one of `boolean`, `integer`, `float`, `string`, `array`, and `object`
- For example, `settype($year, "string");` changes the type of `$year` from `integer` to `string`

Prepared by X. Wang

18

## Examine Variable Type and Value

- PHP provides several functions for examining the type and/or value of a variable
- `boolean is_int(mixed variable);`  
`boolean is_float(mixed variable);`  
`boolean is_bool(mixed variable);`  
`boolean is_string(mixed variable);`  
`boolean is_numeric(mixed variable);`  
`boolean is_array(mixed variable);`  
`boolean is_object(mixed variable);`
- `string gettype(mixed expression);`  
`print_r(mixed expression);`  
`var_dump(mixed expression [,mixed expression...]);`
- `boolean isset(mixed variable);`  
`boolean empty(mixed variable);`  
`unset(mixed variable [,mixed variable...]);`

Prepared by X. Wang

19

## Constants

- PHP uses the `define()` function to create a constant
- The syntax is  
`define(string constantName, mixed value);`
- For example:  
`define("PI", 3.14159265);`  
 creates the constant `PI`

Prepared by X. Wang

20

## Output: print and echo

- PHP offers a number of means for displaying information
- Both `print` and `echo` statements can be used to output messages
- The difference is that the `print` statement only accepts one parameter, while the `echo` statement can accept more than one parameter, each separated by a comma
- For example:  
`$value = 42;`  
`print "The answer is $value<br />";`  
`echo "The answer is ", $value, "<br />";`

Prepared by X. Wang

21

## Strings with double quotes

- To represent a string, PHP can use both `double quotes` and `single quotes`
- When using `double quotes`, both `variables` and `escape characters` in the string will be interpreted accordingly
- For example:  
`$str1 = "PHP";`  
`$str2 = "code";`  
`print "$str1 \'and\' $str2\n";`  
 will print the string: `PHP \'and\' code`

Prepared by X. Wang

22

## Variable Substitution

- In the previous example, we can see both `$str1` and `$str2` variables are replaced by the variable values
- PHP interprets the `$` and the following non-space characters in double quotes as the name of a variable
- To include a `$` character in a string, we need to use `\$`
- When the name of the variable is ambiguous, we can use braces `{}` to delimit the name
- For example:  
`$memory = 256;`  
`print "My computer has {$memory}MB of RAM";`

Prepared by X. Wang

23

## Strings with single quotes

- When using `single quotes`, both `variables` and `escape characters` in the string will not be interpreted
- For example:  
`$str1 = "PHP";`  
`$str2 = "code";`  
`print '$str1 \'and\' $str2\n';`  
 Will print the string: `$str1 \'and\' $str2\n`

Prepared by X. Wang

24

## Operators in PHP

---

- Arithmetic operators: `+`, `-`, `*`, `/`, `%`, `++`, `--`
- Assignment operators: `=`, `+=`, `-=`, `*=`, `/=`, `%=`
- Relational operators: `==`, `===`, `!=`, `!==`, `<`, `<=`, `>`, `>=`
- Logical operators: `&&` (**AND**), `||` (**OR**), `^` (**XOR**), `!`
- Bitwise operators: `&`, `|`, `^`, `~`, `<<`, `>>`
- String dot-operator (`.`) is used to concatenate two strings. For example:  
`$str = "PHP"."code";` creates a new string  
`"PHP code"`

Prepared by X. Wang

25

## Statements in PHP

---

- Assignment statement
- Selection statements: `if`, `if...else` (**elseif**), `switch-case`
- Loop statements: `for`, `while`, `do...while`, `foreach`
- Other simple statements: `break`, `continue`, `return`, `exit`

Prepared by X. Wang

26