

CSCI 241

Writing Functions and Reusing Code

Department of CS & QM
Winthrop University
Fall 2011

Prepared by X. Wang

Using Functions in PHP

- A function is a self-contained module of code that prescribes a calling interface, performs some task, and optionally returns a result
- PHP has defined many internal (build-in) functions for PHP programmers to call
- All these build-in functions are listed at <http://us2.php.net/manual/en/funcref.php>

Prepared by X. Wang

2

User-Defined Functions

- The syntax to define a function:


```
function funcName(parameter list)
{
    // statements;
}
```
- Parameters are separated with comma
- A function may have a returning value

Prepared by X. Wang

3

Variable Scope

- **Local variables:** are created within a function
- A local variable can be accessed only in the function
- **Function formal parameters** work as local variables
- **Global variables:** are created out of any functions
- A global variable can be accessed in any part of the program (page)

Prepared by X. Wang

4

Variable Scope

- To use a global variable inside of a function, we need to place the keyword **GLOBAL** (**global**) in front of the variable
- For example


```
$x = 100;
function printValue() {
    $x = 50;
    print "\$x = $x<br />";
    GLOBAL $x;
    $x += 20;
    print "\$x = $x<br />";
}
printValue();
```

will print **\$x = 50** and **\$x = 120**

Prepared by X. Wang

5

Static Variables

- Variables can be declared as **static** within a function
- A static variable is **only** available in the scope of the function
- The value of a static variable is not lost when the function exits
- For example


```
function count() {
    static $count=0;
    $count++;
    print $count;
}
count();
count();
```

will print **1** and **2**

Prepared by X. Wang

6

Pass Variables to Functions

- Variables can be passed to a function by value or by reference
- An example to pass variable by value


```
function doubleValue($var) {
    $var *= 2
}
$num = 5;
doubleValue($num);
print "\$num is $num";
```

will print \$num is 5

Prepared by X. Wang

7

Pass Variables to Functions

- An example to pass variable by reference


```
function doubleValue(&$var) {
    $var *= 2
}
$num = 5;
doubleValue($num);
print "\$num is $num";
```

will print \$num is 10

Prepared by X. Wang

8

Using the return statement

- The **return** statement can terminate a function immediately
- A function ends by reaching either the end of the function or a **return** statement
- The **return** statement can have no parameter (returning value) or bring a parameter that is the returning value of the function

Prepared by X. Wang

9

Default Parameter Values

- A function parameter may have its default value


```
function heading($text, $headingLevel = 2) {
    switch ($headingLevel) {
        case 1: $result="<h1>$text</h1>"; break;
        case 2: $result="<h2>$text</h2>"; break;
        case 3: $result="<h3>$text</h3>"; break;
        default: $result="<p><b>$text</b></p>";
    }
    return $result;
}
$test = "User-Defined Functions";
print heading($test);
```

will use the value 2 for the \$headingLevel

Prepared by X. Wang

10

Include and Require Files

- PHP allows us to use PHP script code and even static HTML code defined in other files through **include** and **require** directives
- Any PHP code in an external **include** file must have the PHP start and end tags `<?php` and `?>`
- Both **include** and **require** can read external include files, the only difference is in the behavior when a file can't be included: **include** provides a warning whereas **require** terminates the script with a fatal error
- We will use the **require** directive

Prepared by X. Wang

11

Include and Require Files

- The external include file usually has an extension `.inc` or `.php`, but can be any other extension name
- The following is the content of the external include file `myFunctions.inc`

```
<?php
function bold($text) {
    print "<b>" . $text . "</b>";
}
?>
```

Prepared by X. Wang

12

Include and Require Files

- An external include file can be stored in the same folder as the PHP script file, or in another folder
- If stored in the same folder, it's better to use `.php` as the extension name. Otherwise, people can see the original PHP source code in the include file by directly accessing the file from a web browser
- If not in the same folder, the `path` (relative or absolute) should be specified in the `include` or `require` directive

Prepared by X. Wang

13

Include and Require Files

- Assume that the previous include file is stored in the `inc` folder under the current folder.
- Here is an example to use the include file

```
<body>
<?php
require "inc/myFunctions.inc";

$myString = "This is bold";
print "This is not bold<br />";
bold($myString);
print "<br /> This is again not bold<br />";
?>
</body>
```

Prepared by X. Wang

14