

Chapter 2

Functions and Control Structures

PHP Programming with MySQL
2nd Edition

Objectives

In this chapter, you will:

- Study how to use functions to organize your PHP code
- Learn about variable scope
- Make decisions using `if` statements, `if...else` statements, and `switch` statements
- Repeatedly execute `while` statements, `do...while` statements, `for`, and `foreach` statements
- Learn about `include` and `require` statements

PHP Programming with MySQL, 2nd Edition

Defining Functions

- **Functions** are groups of statements that you can execute as a single unit
- **Function definitions** are the lines of code that make up a function
- The syntax for defining a function is:

```
<?php
function name_of_function(parameters) {
    statements;
}
?>
```

PHP Programming with MySQL, 2nd Edition

Defining Functions (continued)

- Functions, like all PHP code, must be contained within `<?php ... ?>` tags
- A **parameter** is a variable that is passed to a function when it is called
- Parameters are placed within the parentheses that follow the function name
- Functions do not have to contain parameters
- The set of curly braces (called **function braces**) contain the function statements

PHP Programming with MySQL, 2nd Edition

Defining Functions (continued)

- **Function statements** do the actual work of the function and must be contained within the function braces

```
function displayCompanyName($Company1,
    $Company2, $Company3) {
    echo "<p>$Company1</p>";
    echo "<p>$Company2</p>";
    echo "<p>$Company3</p>";
}
```

PHP Programming with MySQL, 2nd Edition

Calling Functions

```
function displayCompanyName($CompanyName) {
    echo "<p>$CompanyName</p>";
}
displayCompanyName("Course Technology");
```

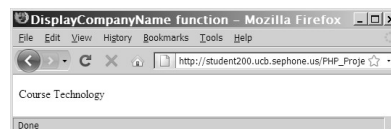


Figure 2-1 Output of a call to a custom function

PHP Programming with MySQL, 2nd Edition

Returning Values

- A **return statement** returns a value to the statement that called the function
- Not all functions return values

```
function averageNumbers($a, $b, $c) {
    $SumOfNumbers = $a + $b + $c;
    $Result = $SumOfNumbers / 3;
    return $Result;
}
```

PHP Programming with MySQL, 2nd Edition

Returning Values (continued)

- You can pass a function parameter by **value** or by **reference**
- A function parameter that is passed by value is a local copy of the variable.
- A function parameter that is passed by reference is a reference to the original variable.

PHP Programming with MySQL, 2nd Edition

Understanding Variable Scope

- **Variable scope** is where in your program a declared variable can be used
- A variable's scope can be either global or local
- A **global variable** is one that is declared outside a function and is available to all parts of your program
- A **local variable** is declared inside a function and is only available within the function in which it is declared

PHP Programming with MySQL, 2nd Edition

The `global` Keyword

- In PHP, you must declare a global variable with the `global` keyword inside a function definition to make the variable available within the scope of that function

PHP Programming with MySQL, 2nd Edition

The `global` Keyword (continued)

```
<?php
$GlobalVariable = "Global variable";
function scopeExample() {
    global $GlobalVariable;
    echo "<p>$GlobalVariable</p>";
}
scopeExample();
?>
```

PHP Programming with MySQL, 2nd Edition

Making Decisions

- **Decision making** or **flow control** is the process of determining the order in which statements execute in a program
- The special types of PHP statements used for making decisions are called **decision-making statements** or **decision-making structures**

PHP Programming with MySQL, 2nd Edition

if Statements

- Used to execute specific programming code if the evaluation of a conditional expression returns a value of TRUE
- The syntax for a simple if statement is:


```
if (conditional expression)
    statement;
```

PHP Programming with MySQL, 2nd Edition

if Statements (continued)

- Contains three parts:
 - the keyword `if`
 - a conditional expression enclosed within parentheses
 - the executable statements
- A **command block** is a group of statements contained within a set of braces
- Each command block must have an opening brace (`{`) and a closing brace (`}`)

PHP Programming with MySQL, 2nd Edition

if Statements (continued)

```
$ExampleVar = 5;
if ($ExampleVar == 5) { // condition evaluates to 'TRUE'
    echo " <p>The condition evaluates to true.</p> ";
    echo " <p>$ExampleVar is equal to ',
        " $ExampleVar.</p> ";
    echo " <p>Each of these lines will be printed.</p> ";
}
echo " <p>This statement always executes after the if
statement.</p> ";
```

PHP Programming with MySQL, 2nd Edition

if...else Statements

- An if statement that includes an else clause is called an **if...else statement**
- An else clause executes when the condition in an if...else statement evaluates to FALSE
- The syntax for an if...else statement is:

```
if (conditional expression)
    statement;
else
    statement;
```

PHP Programming with MySQL, 2nd Edition

if...else Statements (continued)

- An if statement can be constructed without the else clause
- The else clause can only be used with an if statement

```
$Today = "Tuesday";
if ($Today == "Monday")
    echo " <p>Today is Monday</p> ";
else
    echo " <p>Today is not Monday</p> ";
```

PHP Programming with MySQL, 2nd Edition

Nested if and if...else Statements

- When one decision-making statement is contained within another decision-making statement, they are referred to as nested **decision-making structures**

```
if ($SalesTotal >= 50)
    if ($SalesTotal <= 100)
        echo " <p>The sales total is between
        50 and 100, inclusive.</p> ";
```

PHP Programming with MySQL, 2nd Edition

switch Statements

- Control program flow by executing a specific set of statements depending on the value of an expression
- Compare the value of an expression to a value contained within a special statement called a **case label**
- A **case label** is a specific value that contains one or more statements that execute if the value of the case label matches the value of the switch statement's expression

PHP Programming with MySQL, 2nd Edition

switch Statements (continued)

- Consist of the following components:
 - The `switch` keyword
 - An expression
 - An opening brace
 - One or more `case` labels
 - The executable statements
 - The `break` keyword
 - A `default` label
 - A closing brace

PHP Programming with MySQL, 2nd Edition

switch Statements (continued)

- The syntax for the `switch` statement is:

```
switch (expression) {
    case label:
        statement(s);
        break;
    case label:
        statement(s);
        break;
    ...
    default:
        statement(s);
        break;
}
```

PHP Programming with MySQL, 2nd Edition

switch Statements (continued)

- A `case` label consists of:
 - The keyword `case`
 - A literal value or variable name
 - A colon (:)
- A `case` label can be followed by a single statement or multiple statements
- Multiple statements for a `case` label do not need to be enclosed within a command block

PHP Programming with MySQL, 2nd Edition

switch Statements (continued)

- The **default label** contains statements that execute when the value returned by the `switch` statement expression does not match a `case` label
- A default label consists of the keyword `default` followed by a colon (:)

PHP Programming with MySQL, 2nd Edition

Repeating Code

- A **loop statement** is a control structure that repeatedly executes a statement or a series of statements while a specific condition is `TRUE` or until a specific condition becomes `TRUE`
- There are four types of loop statements:
 - `while` statements
 - `do...while` statements
 - `for` statements
 - `foreach` statements

PHP Programming with MySQL, 2nd Edition

while Statements

- Tests the condition prior to executing the series of statements at each iteration of the loop
- The syntax for the `while` statement is:


```
while (conditional expression) {
    statement(s);
}
```
- As long as the conditional expression evaluates to `TRUE`, the statement or command block that follows executes repeatedly

PHP Programming with MySQL, 2nd Edition

while Statements (continued)

- Each repetition of a looping statement is called an **iteration**
- A `while` statement keeps repeating until its conditional expression evaluates to `FALSE`
- A **counter** is a variable that increments or decrements with each iteration of a loop statement

PHP Programming with MySQL, 2nd Edition

while Statements (continued)

```
$Count = 1;
while ($Count <= 5) {
    echo "$Count<br />";
    ++$Count;
}
echo "<p>You have printed 5 numbers.</p>";
```

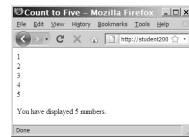


Figure 2-5 Output of a `while` statement using an increment operator

PHP Programming with MySQL, 2nd Edition

while Statements (continued)

```
$Count = 10;
while ($Count > 0) {
    echo "$Count<br />";
    --$Count;
}
echo "<p>We have liftoff.</p>";
```

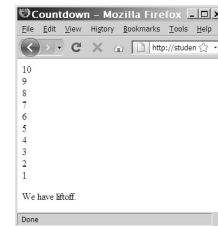


Figure 2-6 Output of a `while` statement using a decrement operator

PHP Programming with MySQL, 2nd Edition

while Statements (continued)

```
$Count = 1;
while ($Count <= 100) {
    echo "$Count<br />";
    $Count *= 2;
}
```

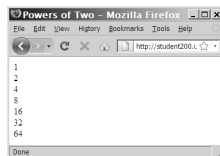


Figure 2-7 Output of a `while` statement using the assignment operator `*=`

PHP Programming with MySQL, 2nd Edition

while Statements (continued)

- In an **infinite loop**, a loop statement never ends because its conditional expression is never `FALSE`

```
$Count = 1;
while ($Count <= 10) {
    echo "The number is $Count";
}
```

PHP Programming with MySQL, 2nd Edition

do...while Statements

- Test the condition after executing a series of statements then repeats the execution as long as a given conditional expression evaluates to TRUE
- The syntax for the do...while statement is:


```
do {
    statement(s);
} while (conditional expression);
```

PHP Programming with MySQL, 2nd Edition

do...while Statements (continued)

- do...while statements always execute once, before a conditional expression is evaluated

```
$Count = 2;
do {
    echo "<p>The count is equal to $Count</p>";
    ++$Count;
} while ($Count < 2);
```

PHP Programming with MySQL, 2nd Edition

do...while Statements (continued)

```
$DaysOfWeek = array("Monday", "Tuesday", "Wednesday", "Thursday",
"Friday", "Saturday", "Sunday");
$Count = 0;
do {
    echo $DaysOfWeek[$Count], "<br />";
    ++$Count;
} while ($Count < 7);
```

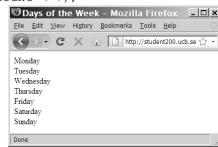


Figure 2-9 Output of days of week script in Web browser

PHP Programming with MySQL, 2nd Edition

for Statements

- Combine the initialize, conditional evaluation, and update portions of a loop into a single statement
- Repeat a statement or a series of statements as long as a given conditional expression evaluates to TRUE
- If the conditional expression evaluates to TRUE, the for statement executes and continues to execute repeatedly until the conditional expression evaluates to FALSE

PHP Programming with MySQL, 2nd Edition

for Statements (continued)

- Can also include code that initializes a counter and changes its value with each iteration
- The syntax of the for statement is:

```
for (counter declaration and initialization;
condition;
    update statement) {
    statement(s);
}
```

PHP Programming with MySQL, 2nd Edition

for Statements (continued)

```
$FastFoods = array("pizza", "burgers", "french fries", "tacos",
"fried chicken");
for ($Count = 0; $Count < 5; ++$Count) {
    echo $FastFoods[$Count], "<br />";
}
```

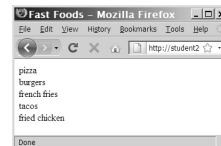


Figure 2-10 Output of fast foods script

PHP Programming with MySQL, 2nd Edition

foreach Statements

- Used to iterate or loop through the elements in an array
- Do not require a counter; instead, you specify an array expression within a set of parentheses following the `foreach` keyword
- The syntax for the `foreach` statement is:


```
foreach ($array_name as $variable_name) {
    statements;
}
```

PHP Programming with MySQL, 2nd Edition

foreach Statements (continued)

```
$DaysOfWeek = array("Monday", "Tuesday",
    "Wednesday", "Thursday", "Friday",
    "Saturday", "Sunday");
foreach ($DaysOfWeek as $Day) {
    echo "<p>$Day</p>";
}
```

PHP Programming with MySQL, 2nd Edition

foreach Statements (continued)

```
$DaysOfWeek = array("Monday", "Tuesday",
    "Wednesday", "Thursday", "Friday", "Saturday",
    "Sunday");
foreach ($DaysOfWeek as $DayNumber => $Day) {
    echo "<p>Day $DayNumber is $Day</p>";
}
```

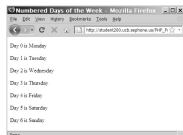


Figure 2-11 Output of the `foreach` script with index values
PHP Programming with MySQL, 2nd Edition

Including Files

- The `include` and `require` statements reuse content by allowing you to insert the content of an external file on multiple Web pages
 - The `include` statement generates a warning if the include file cannot be found
 - The `require` statement halts the processing of the Web page and displays an error if the include file cannot be found
- The `include_once` and `require_once` statements assure that the external file is added to the script only one time

PHP Programming with MySQL, 2nd Edition

Summary

- The lines that make up a function are called the **function definition**
- A function parameter that is passed by **value** is a local copy of the variable
- A function parameter that is passed by **reference** is a reference to the original variable
- A **global variable** is declared outside a function and is available to all parts of your program

PHP Programming with MySQL, 2nd Edition

Summary (continued)

- A **local variable** is declared inside a function and is only available within the function in which it is declared
- The process of determining the order in which statements execute in a program is called **decision making** or **flow control**
- The `if` statement is used to execute specific programming code if the evaluation of a conditional expression returns a value of `TRUE`

PHP Programming with MySQL, 2nd Edition

Summary (continued)

- An `if` statement that includes an `else` clause is called an `if...else` statement. An `else` clause executes when the condition in an `if...else` statement evaluates to `FALSE`
- When one decision-making statement is contained within another decision-making statement, they are referred to as **nested decision-making structures**

PHP Programming with MySQL, 2nd Edition

Summary (continued)

- The **switch statement** controls program flow by executing a specific set of statements, depending on the value of an expression
- A **loop statement** is a control structure that repeatedly executes a statement or a series of statements while a specific condition is `TRUE` or until a specific condition becomes `TRUE`
- A `while` statement tests the condition prior to executing the series of statements at each iteration of the loop

PHP Programming with MySQL, 2nd Edition

Summary (continued)

- The `do...while` statement tests the condition after executing a series of statements
- The `for` statement combines the initialize, conditional evaluation, and update portions of a loop into a single statement
- The `foreach` statement is used to iterate or loop through the elements in an array

PHP Programming with MySQL, 2nd Edition

Summary (continued)

- The `include`, `require`, `include_once`, and `require_once` statements insert the contents of an external file at the location of the statement

PHP Programming with MySQL, 2nd Edition