Introduction to Server-Side Development with PHP

Chapter 8

Randy Connolly and Ricardo Hoar

Fundamentals of Web Development

Objectives

1 Server-Side Development

2 Web Server's Responsabilities

3 Quick Tour of PHP

4 Program Control

5 Functions

Section 1 of 5

WHAT IS SERVER-SIDE DEVELOPMENT

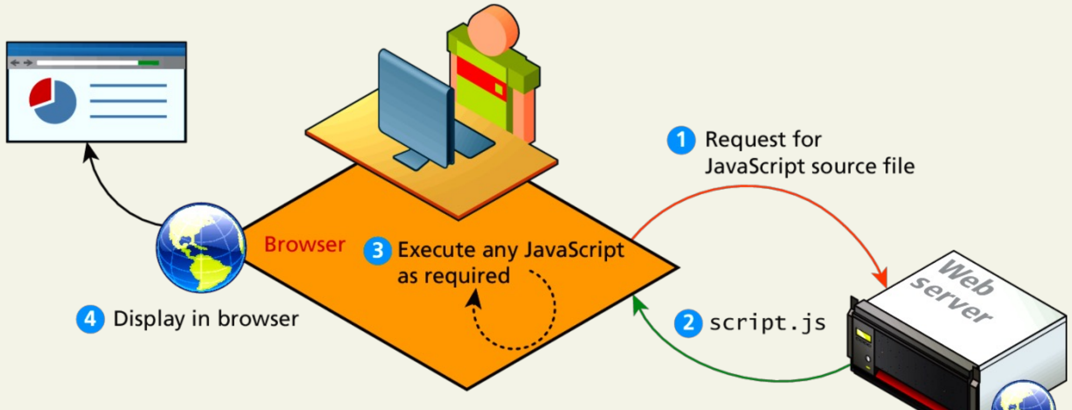
What is Server-Side Development

The basic hosting of your files is achieved through a web server.

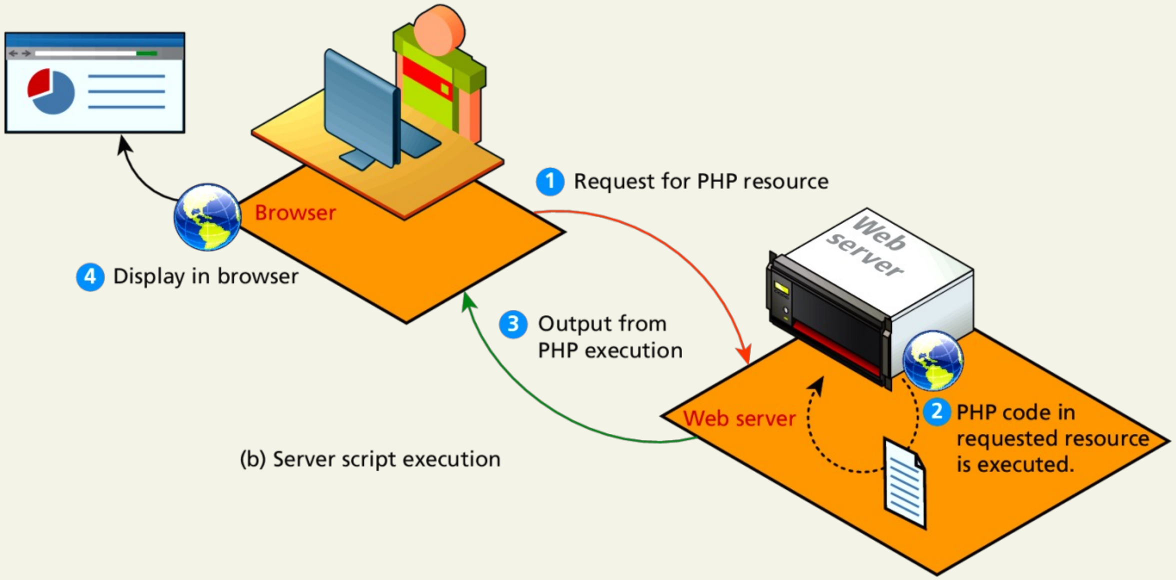
Server-side development is much more than web hosting: it involves the use of a programming technology like PH to create scripts that dynamically generate content

Consider distinction between client side and server side…

Comparing Client and Server Scripts


1 Request for JavaScript source file
2 script.js
Browser
3 Execute any JavaScript as required
4 Display in browser
Web server

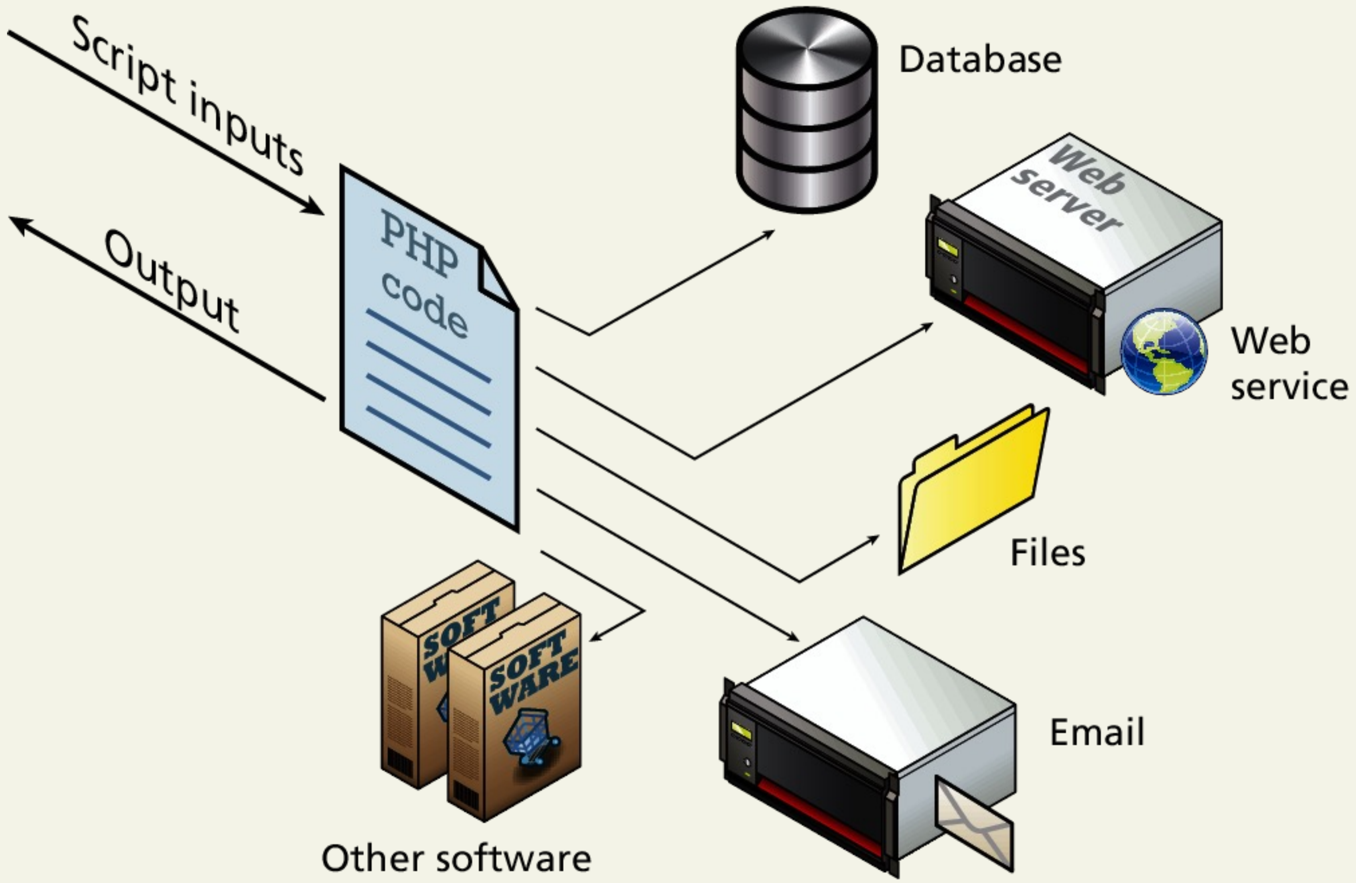
(a) Client script execution


1 Request for PHP resource
Browser
4 Display in browser
3 Output from PHP execution
Web server
2 PHP code in requested resource is executed.
Web server

(b) Server script execution

Server-Side Script Resources

So many tools in your kit



Some Server-Side Technologies

- **JSP (Java Server Pages)**. JSP uses Java as its programming language and it uses an explicit object-oriented approach and is used in large enterprise web systems and is integrated into the J2EE environment. Since JSP uses the Java Runtime Engine, it also uses a JIT compiler for fast execution time and is cross-platform. While JSP's usage in the web as a whole is small, it has a substantial market share in the intranet environment, as well as with very large and busy sites.
- **Node.js**. This is a more recent server environment that uses JavaScript on the server side, thus allowing developers already familiar with JavaScript to use just a single language for both client-side and server-side development. Unlike the other development technologies listed here, node.js also is its own web server software, thus eliminating the need for Apache, IIS, or some other web server software.

Some Server-Side Technologies

- **PHP**. PHP is a dynamically typed language that can be embedded directly within the HTML, though it now supports most common object-oriented features, such as classes and inheritance. By default, PHP pages are compiled into an intermediary representation called **opcodes** that are analogous to Java's byte-code or the .NET Framework's MSIL. Originally, PHP stood for *personal home pages*, although it now is a recursive acronym that means

  PHP: Hypertext Preprocessor

Section 2 of 5

WEB SERVER'S RESPONSABILITIES

A Web Server's Responsibilities

A web server has many responsibilities:

- handling HTTP connections
- responding to requests for static and dynamic resources
- managing permissions and access for certain resources
- encrypting and compressing data
- managing multiple domains and URLs
- managing database connections
- managing cookies and state
- uploading and managing files

LAMP stack

WAMP, MAMP, …

The LAMP software stack

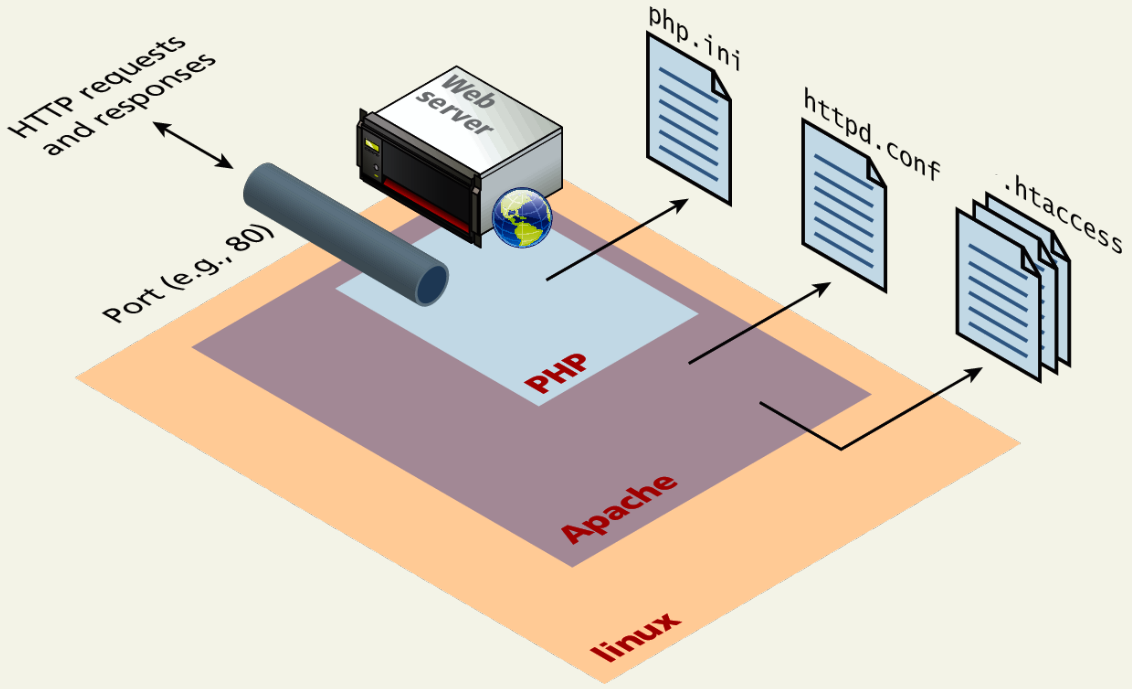
- **L**inux operating system
- **A**pache web server
- **M**ySQL DBMS
- **P**HP scripting language

- You will be using XAMPP on Windows or LAMP if using the VM.

Apache and Linux

LA

Consider the **Apache** web server as the intermediary that interprets HTTP requests that arrive through a network port and decides how to handle the request, which often requires working in conjunction with PHP.



Apache

Continued

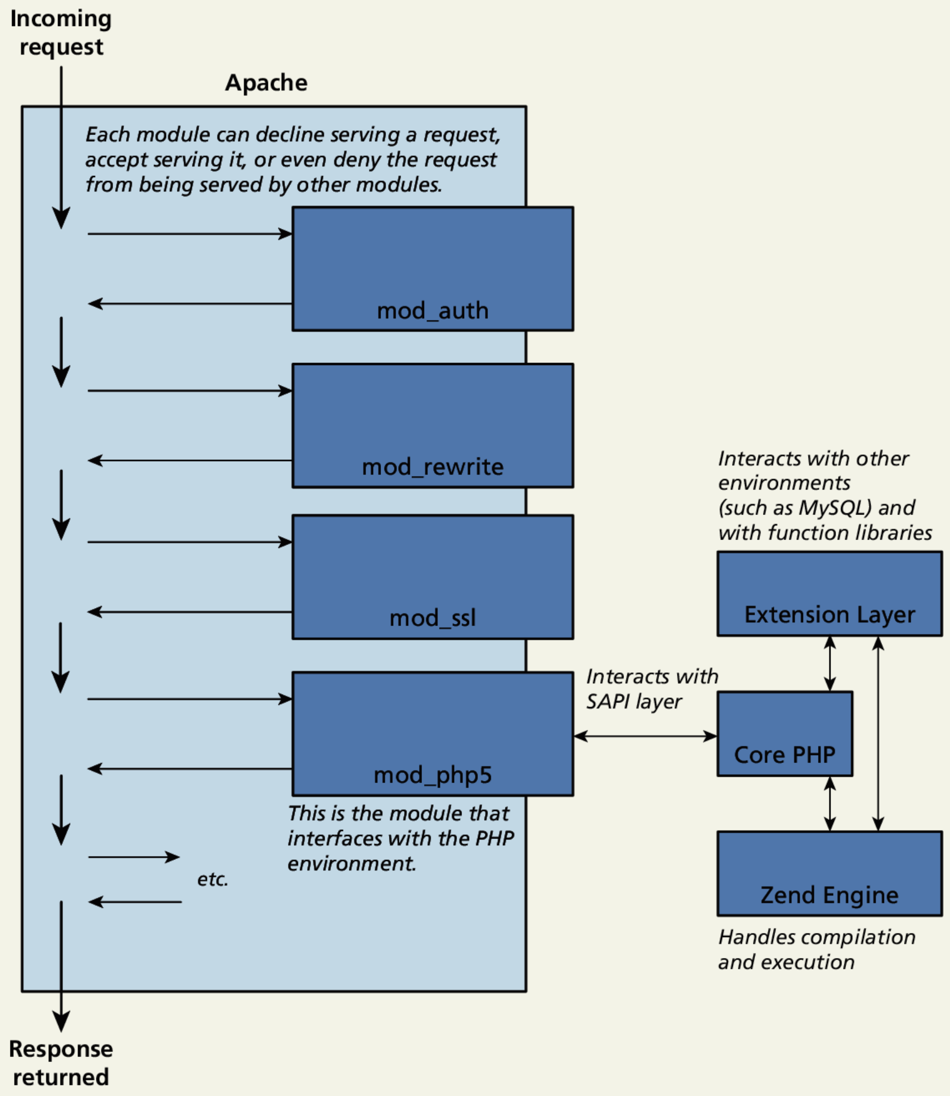
Apache runs as a daemon on the server. A **daemon** is a **process** that runs in the background, waiting for a specific event that will activate it.

When a request arrives, Apache then uses modules to determine how to respond to the request.

In Apache, a **module** is a compiled extension (usually written in the C programming language) to Apache that helps it *handle* requests. For this reason, these modules are also sometimes referred to as **handlers**.

Apache and PHP

PHP Module in Apache



Incoming request
Apache
Each module can decline serving a request, accept serving it, or even deny the request from being served by other modules.
mod_auth
mod_rewrite
mod_ssl
mod_php5
This is the module that interfaces with the PHP environment.
etc.
Interacts with SAPI layer
Interacts with other environments (such as MySQL) and with function libraries
Extension Layer
Core PHP
Zend Engine
Handles compilation and execution
Response returned

Apache Threads

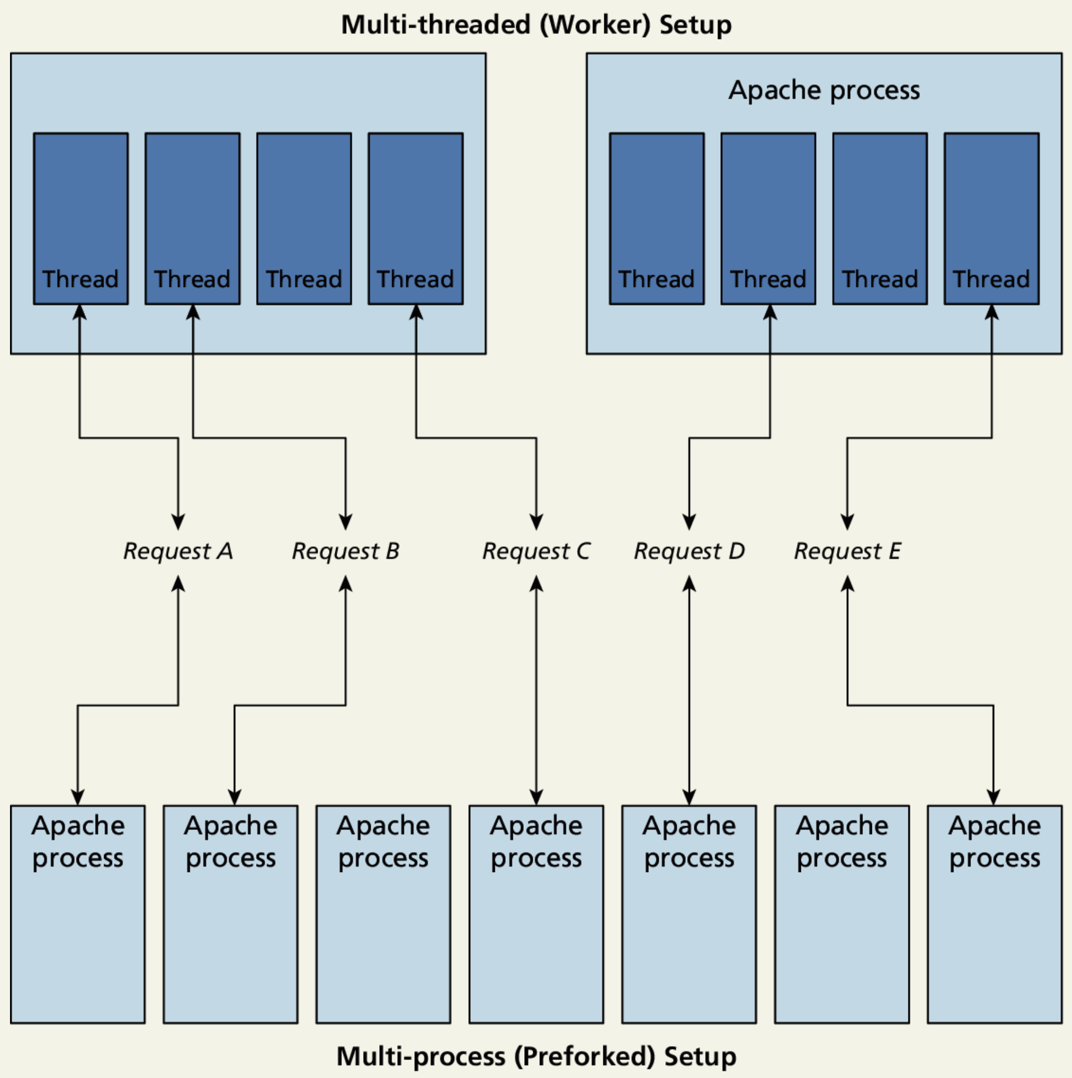
Multi-thread and multi-process

Apache runs in two possible modes:

- **multi-process** (also called **preforked**)
- **multi-threaded** (also called **worker**)

Apache Threads

Multi-thread and multi-process



PHP Internals

PHP itself is written in C

There are 3 main modules

1. **PHP core**. The Core module defines the main features of the PHP environment, including essential functions for variable handling, arrays, strings, classes, math, and other core features.
2. **Extension layer**. This module defines functions for interacting with services outside of PHP. This includes libraries for MySQL, FTP, SOAP web services, and XML processing, among others.
3. **Zend Engine**. This module handles the reading in of a requested PHP file, compiling it, and executing it.

Installing AMP locally

Turn this key

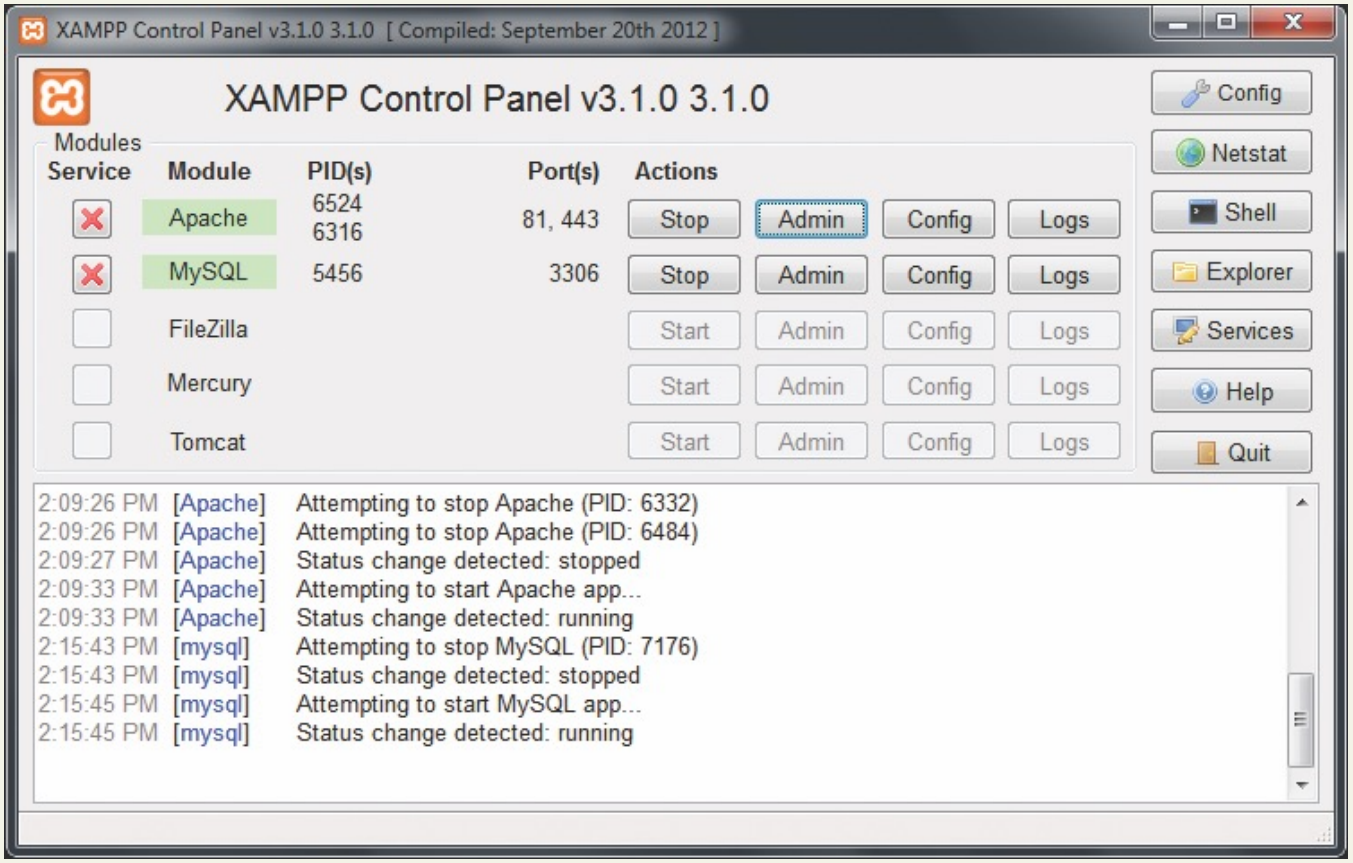
The easiest and quickest way to do so is to use the

- **XAMPP**

It installs and configures Apache, PHP, and MySQL.

XAMPP Control Panel

Turn this key



XAMPP Control Panel v3.1.0 3.1.0 [Compiled: September 20th 2012]
XAMPP Control Panel v3.1.0 3.1.0
Modules
Config
Netstat
Shell
Explorer
Services
Help
Quit

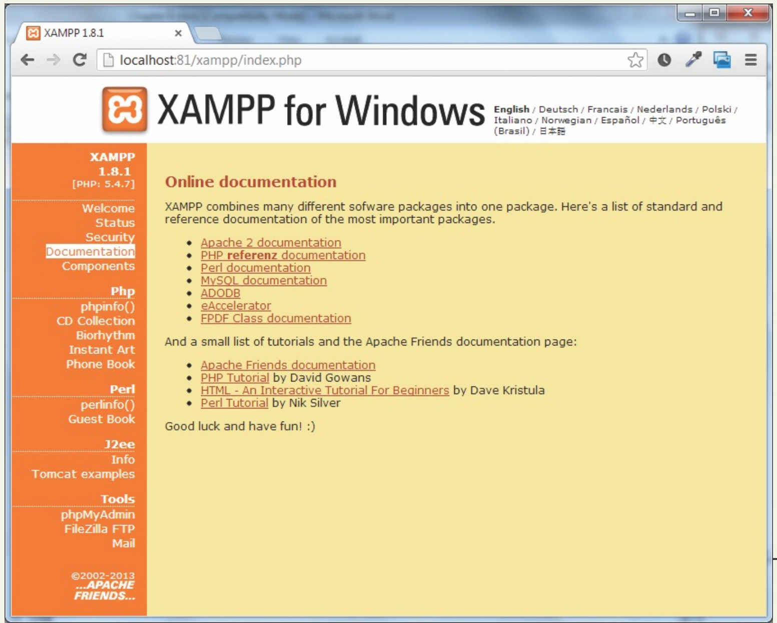
Service	Module	PID(s)	Port(s)	Actions			
✗	Apache	6524 6316	81, 443	Stop	Admin	Config	Logs
✗	MySQL	5456	3306	Stop	Admin	Config	Logs
	FileZilla			Start	Admin	Config	Logs
	Mercury			Start	Admin	Config	Logs
	Tomcat			Start	Admin	Config	Logs

```
2:09:26 PM  [Apache]   Attempting to stop Apache (PID: 6332)
2:09:26 PM  [Apache]   Attempting to stop Apache (PID: 6484)
2:09:27 PM  [Apache]   Status change detected: stopped
2:09:33 PM  [Apache]   Attempting to start Apache app...
2:09:33 PM  [Apache]   Status change detected: running
2:15:43 PM  [mysql]    Attempting to stop MySQL (PID: 7176)
2:15:43 PM  [mysql]    Status change detected: stopped
2:15:45 PM  [mysql]    Attempting to start MySQL app...
2:15:45 PM  [mysql]    Status change detected: running
```

XAMPP Settings

Defaults are

- PHP requests in your browser will need to use the **localhost** domain (127.0.0.1)
- PHP files will have to be saved somewhere within the **C:\xampp\htdocs** folder (win) or **/Applications/XAMPP/xamppfiles/htdocs/** (mac) or follow the link on the desktop in our Linux VM



XAMPP 1.8.1

localhost:81/xampp/index.php

XAMPP for Windows

English / Deutsch / Francais / Nederlands / Polski / Italiano / Norwegian / Español / 中文 / Português (Brasil) / 日本語

XAMPP
1.8.1
[PHP: 5.4.7]

Welcome
Status
Security
Documentation
Components

Php
phpinfo()
CD Collection
Biorhythm
Instant Art
Phone Book

Perl
perlinfo()
Guest Book

J2ee
Info
Tomcat examples

Tools
phpMyAdmin
FileZilla FTP
Mail

©2002-2013
...APACHE FRIENDS...

Online documentation

XAMPP combines many different sofware packages into one package. Here's a list of standard and reference documentation of the most important packages.

- Apache 2 documentation
- PHP **referenz** documentation
- Perl documentation
- MySQL documentation
- ADODB
- eAccelerator
- FPDF Class documentation

And a small list of tutorials and the Apache Friends documentation page:

- Apache Friends documentation
- PHP Tutorial by David Gowans
- HTML - An Interactive Tutorial For Beginners by Dave Kristula
- Perl Tutorial by Nik Silver

Good luck and have fun! :)

Section 3 of 5

QUICK TOUR OF PHP

Quick Tour

- PHP, like JavaScript, is a dynamically typed language.
- It uses classes and functions in a way consistent with other object-oriented languages such as C++, C#, and Java
- The syntax for loops, conditionals, and assignment is almost identical to JavaScript
- Differs when you get to functions, classes, and in how you define variables

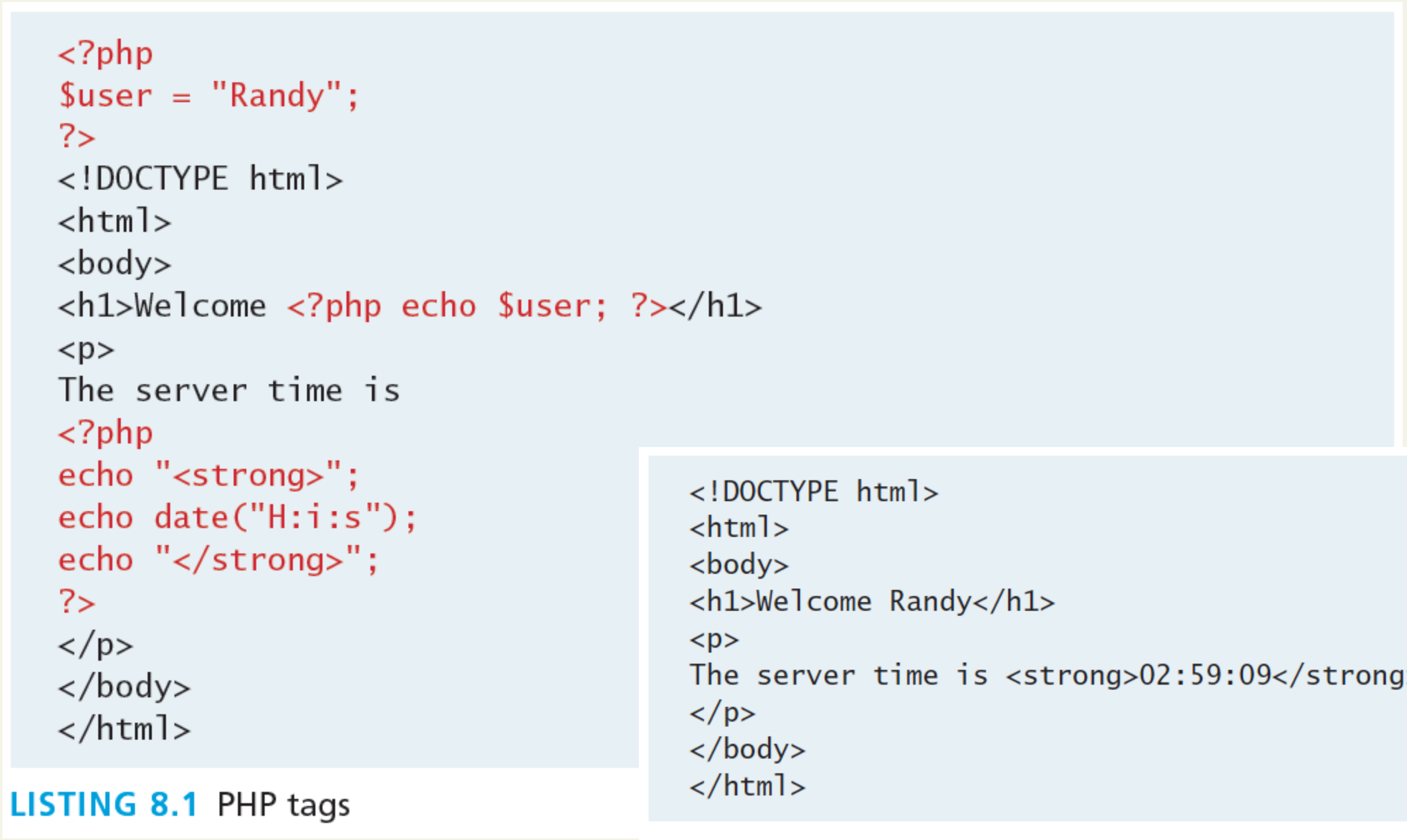
PHP Tags

The most important fact about PHP is that the programming code can be embedded directly within an HTML file.

- A PHP file will usually have the extension **.php**
- programming code must be contained within an opening **<?php** tag and a matching closing **?>** tag
- any code outside the tags is echoed directly out to the client
- The short form **<?= "ABC" ?>** is equivalent to

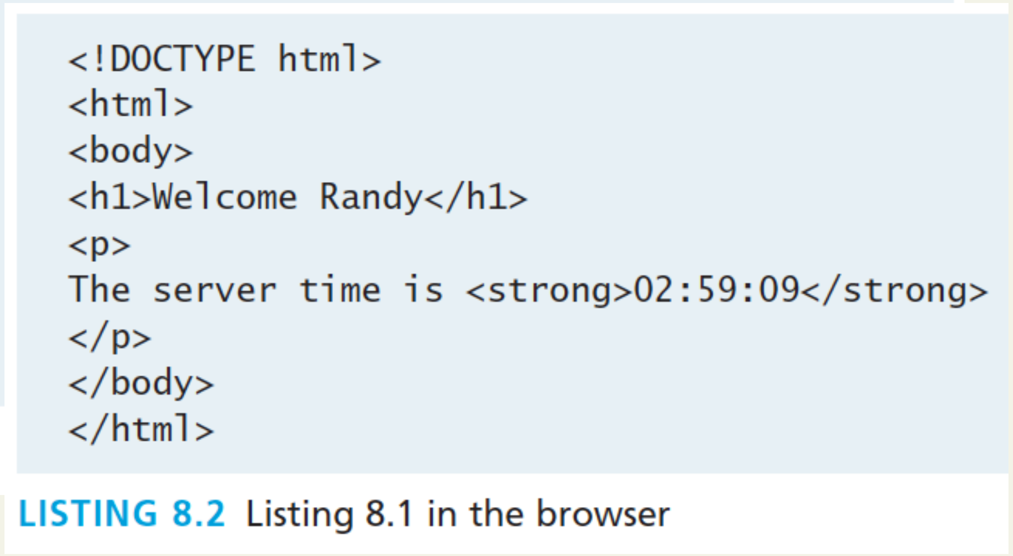
  <?php echo "ABC" ?>

PHP Tags



```
<?php
$user = "Randy";
?>
<!DOCTYPE html>
<html>
<body>
<h1>Welcome <?php echo $user; ?></h1>
<p>
The server time is
<?php
echo "<strong>";
echo date("H:i:s");
echo "</strong>";
?>
</p>
</body>
</html>
```

LISTING 8.1 PHP tags



```
<!DOCTYPE html>
<html>
<body>
<h1>Welcome Randy</h1>
<p>
The server time is <strong>02:59:09</strong>
</p>
</body>
</html>
```

LISTING 8.2 Listing 8.1 in the browser

HTML and PHP

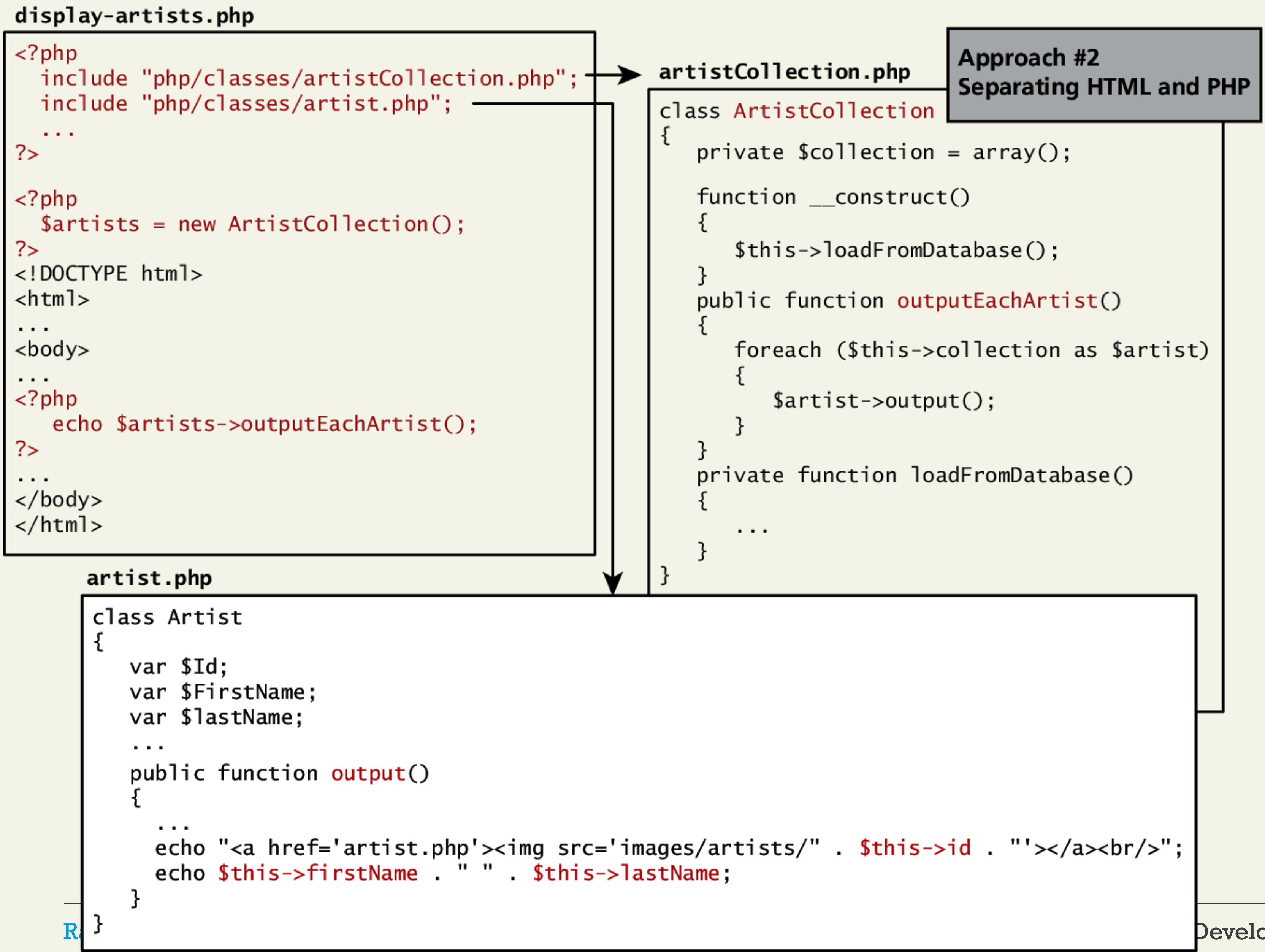
Two approaches

display-artists.php

Approach #1
Mixing HTML and PHP

```
<?php
  $db = new mysqli('localhost', 'dbuser', 'dbpassword', 'dbname');
  $sql = "SELECT * FROM Artists ORDER BY lastName";
  $result = $db->query($sql);
?>
...
<body>
…
<ul>
<?php
while( $row = $result->fetch_assoc() ) {
  echo "<li>";
?>
<img src="images/add.png" /> <img src="images/remove.png" />
<?php
  echo "<a href='artist.php'><img src='images/artists/" . $row['id'] . "'></a><br/>";
  echo $row['firstName'] . " " . $row['lastName'];
  echo "</li>";
}
?>
</ul>
…
<?php
$result->close();
$db->close ();
?>
</body>
</html>
```

HTML and PHP



display-artists.php

```
<?php
  include "php/classes/artistCollection.php";
  include "php/classes/artist.php";
  ...
?>

<?php
  $artists = new ArtistCollection();
?>
<!DOCTYPE html>
<html>
...
<body>
...
<?php
   echo $artists->outputEachArtist();
?>
...
</body>
</html>
```

artistCollection.php

```
class ArtistCollection
{
   private $collection = array();

   function __construct()
   {
      $this->loadFromDatabase();
   }
   public function outputEachArtist()
   {
      foreach ($this->collection as $artist)
      {
         $artist->output();
      }
   }
   private function loadFromDatabase()
   {
      ...
   }
}
```

artist.php

```
class Artist
{
   var $Id;
   var $FirstName;
   var $lastName;
   ...
   public function output()
   {
     ...
     echo "<a href='artist.php'><img src='images/artists/" . $this->id . "'></a><br/>";
     echo $this->firstName . " " . $this->lastName;
   }
}
```

PHP Comments

3 kinds

The types of comment styles in PHP are:

- **Single-line comments**. Lines that begin with a **#** are comment lines and will not be executed.
- **Multiline (block) comments**. These comments begin with a **/*** and encompass everything that is encountered until a closing ***/** tag is found.
- **End-of-line comments**. Whenever **//** is encountered in code, everything up to the end of the line is considered a comment.

PHP Comments

3 kinds

```
<?php

# single-line comment

/*

This is a multiline comment.

They are a good way to document functions or
complicated blocks of code

*/

$artist = readDatabase(); // end-of-line comment

?>
```

Variables

Variables in PHP are **dynamically typed.**

Variables are also **loosely typed** in that a variable can be assigned different data types over time

To declare a variable you must preface the variable name with the dollar (**$**) symbol.

```
$count = 42;
```

Data Types

Data Type	Description
Boolean	A logical true or false value
Integer	Whole numbers
Float	Decimal numbers
String	Letters
Array	A collection of data of any type (covered in the next set of slides)
Object	Instances of classes

Constants

A **constant** is somewhat similar to a variable, except a constant’s value never changes . . . in other words it stays constant.

- Typically defined near the top of a PHP file via the **define()** function
- once it is defined, it can be referenced **without using the $** symbol

Constants

```
<?php

# Uppercase for constants is a programming convention
define("DATABASE_LOCAL", "localhost");
define("DATABASE_NAME", "ArtStore");
define("DATABASE_USER", "Fred");
define("DATABASE_PASSWD", "F5^7%ad");
...
# notice that no $ prefaces constant names
$db = new mysqli(DATABASE_LOCAL, DATABASE_NAME, DATABASE_USER,
  DATABASE_NAME);

?>
```

LISTING 8.4 PHP constants

Writing to Output

Hello World

To output something that will be seen by the browser, you can use the echo() function.

echo ("hello"); //long form

echo "hello"; //shortcut

String Concatenation

Easy

Strings can easily be appended together using the concatenate operator, which is the period (.) symbol.

$username = "World";

echo "Hello ". $username;

Will output **Hello World**

String Concatenation

Example

```
$firstName = "Pablo";

$lastName = "Picasso";

/*

Example one:

These two lines are equivalent. Notice that you can
reference PHP variables within a string literal defined with
double quotes. The resulting output for both lines is:
<em>Pablo Picasso</em>

*/

echo "<em>" . $firstName . " ". $lastName. "</em>";

echo "<em> $firstName $lastName </em>";
```

String Concatenation

Example

```
/*

Example two:

These two lines are also equivalent. Notice that you
can use either the single quote symbol or double quote
symbol for string literals.

*/

echo "<h1>";

echo '<h1>';
```

String Concatenation

Example

```
/*

Example three:

These two lines are also equivalent. In the second
example, the escape character (the backslash) is used
to embed a double quote within a string literal defined
within double quotes.

*/

echo '<img src="23.jpg" >';

echo "<img src=\"23.jpg\" >";
```

Complicated Concatenation

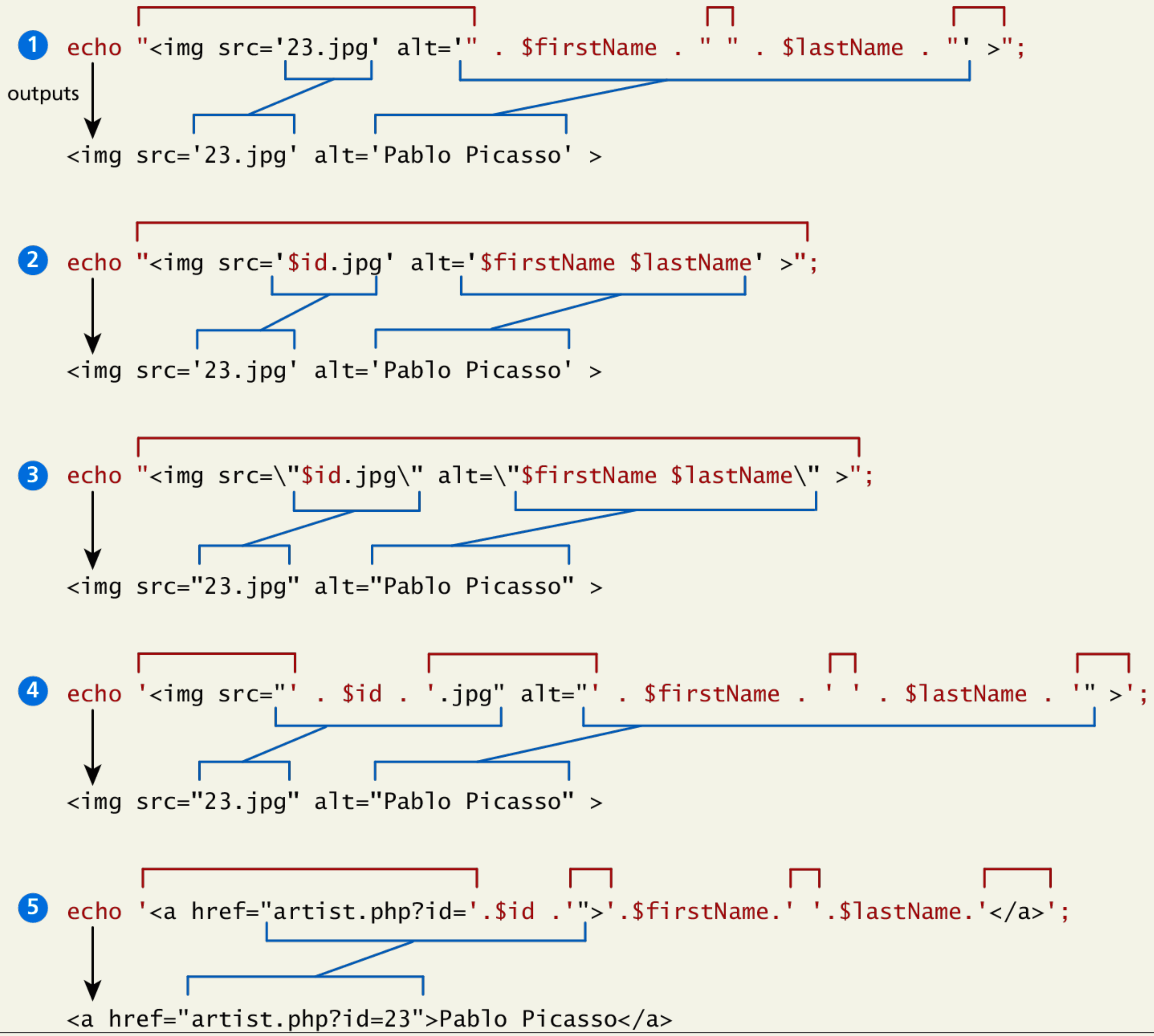
```
echo "<img src='23.jpg' alt='". $firstName . " ". $lastName . "' >";

echo "<img src='$id.jpg' alt='$firstName $lastName' >";

echo "<img src=\"$id.jpg\" alt=\"$firstName $lastName\" >";

echo '<img src="' . $id. '.jpg" alt="' . $firstName . ' ' . $lastName . '" >';

echo '<a href="artist.php?id=' .$id .'">' .$firstName .' ' . $lastName .'</a>';
```



```
echo "<img src='23.jpg' alt='" . $firstName . " " . $lastName . "' >";

<img src='23.jpg' alt='Pablo Picasso' >

echo "<img src='$id.jpg' alt='$firstName $lastName' >";

<img src='23.jpg' alt='Pablo Picasso' >

echo "<img src=\"$id.jpg\" alt=\"$firstName $lastName\" >";

<img src="23.jpg" alt="Pablo Picasso" >

echo '<img src="' . $id . '.jpg" alt="' . $firstName . ' ' . $lastName . '" >';

<img src="23.jpg" alt="Pablo Picasso" >

echo '<a href="artist.php?id='.$id .'">'.$firstName.' '.$lastName.'</a>';

<a href="artist.php?id=23">Pablo Picasso</a>
```

printf

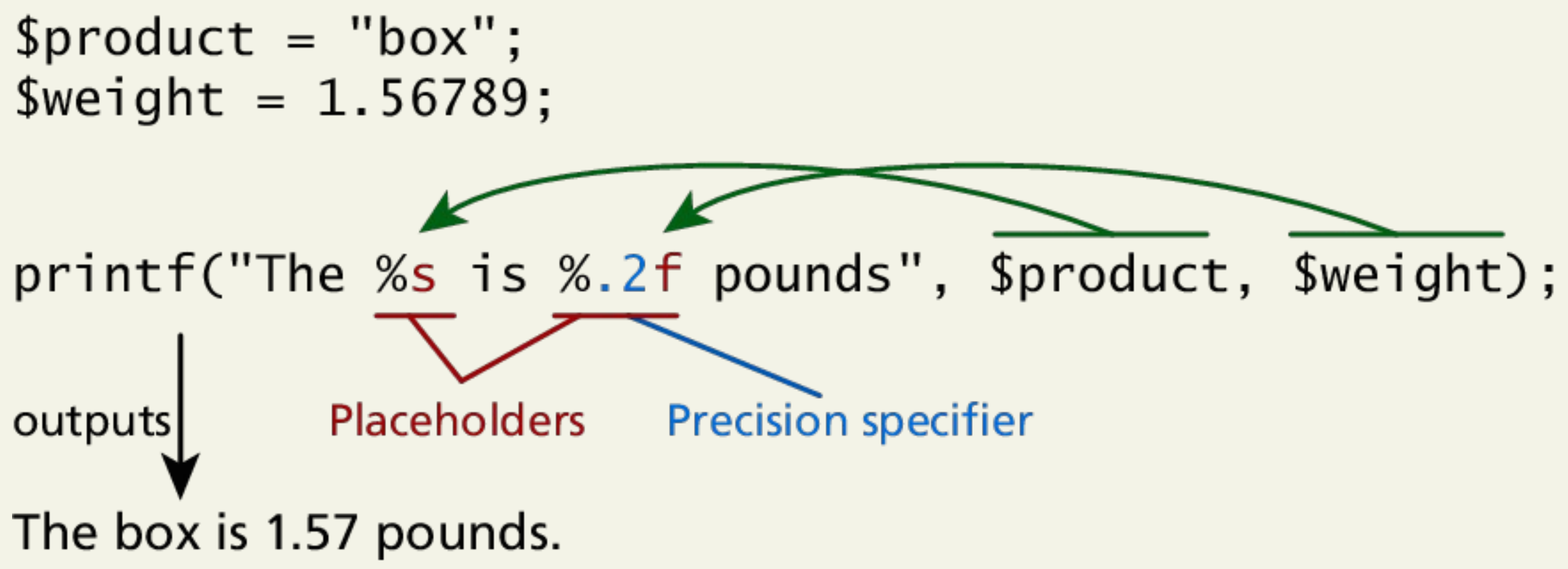
Good ol' printf

As an alternative, you can use the **printf()** function.

- derived from the same-named function in the C programming language
- includes variations to print to string and files (**sprintf**, **fprintf**)
- takes at least one parameter, which is a string, and that string optionally references parameters, which are then integrated into the first string by placeholder substitution
- can also apply special formatting, for instance, specific number of decimal places

printf

Illustrated example



```
$product = "box";
$weight = 1.56789;

printf("The %s is %.2f pounds", $product, $weight);
```

```
The box is 1.57 pounds.
```

printf

Type specifiers

Each placeholder requires the percent (%) symbol in the first parameter string followed by a type specifier.

- **b** for binary
- **d** for signed integer
- **f** for float
- **o** for octal
- **x** for hexadecimal
- **s** for string

```
php > $firstName = "Mario";
php > printf("Ciao %s\n", "Amilcare");
Ciao Amilcare
php > printf("Ciao %s\n", $firstName);
Ciao Mario
```

printf

Precision

Precision allows for control over how many decimal places are shown. Important for displaying calculated numbers to the user in a “pretty” way.

Precision is achieved in the string with a period (.) followed by a number specifying how many digits should be displayed for floating-point numbers.

Section 4 of 5

PROGRAM CONTROL

if…else

The syntax for conditionals in PHP is almost identical to that of JavaScript

```php
// if statement with condition
if ( $hourOfDay > 6 && $hourOfDay < 12 ) {
   $greeting = "Good Morning";
}
else if ($hourOfDay == 12) {   // optional else if
   $greeting = "Good Noon Time";
}
else {                         // optional else branch
   $greeting = "Good Afternoon or Evening";
}
```

LISTING 8.7 Conditional statement using if . . . else

if…else

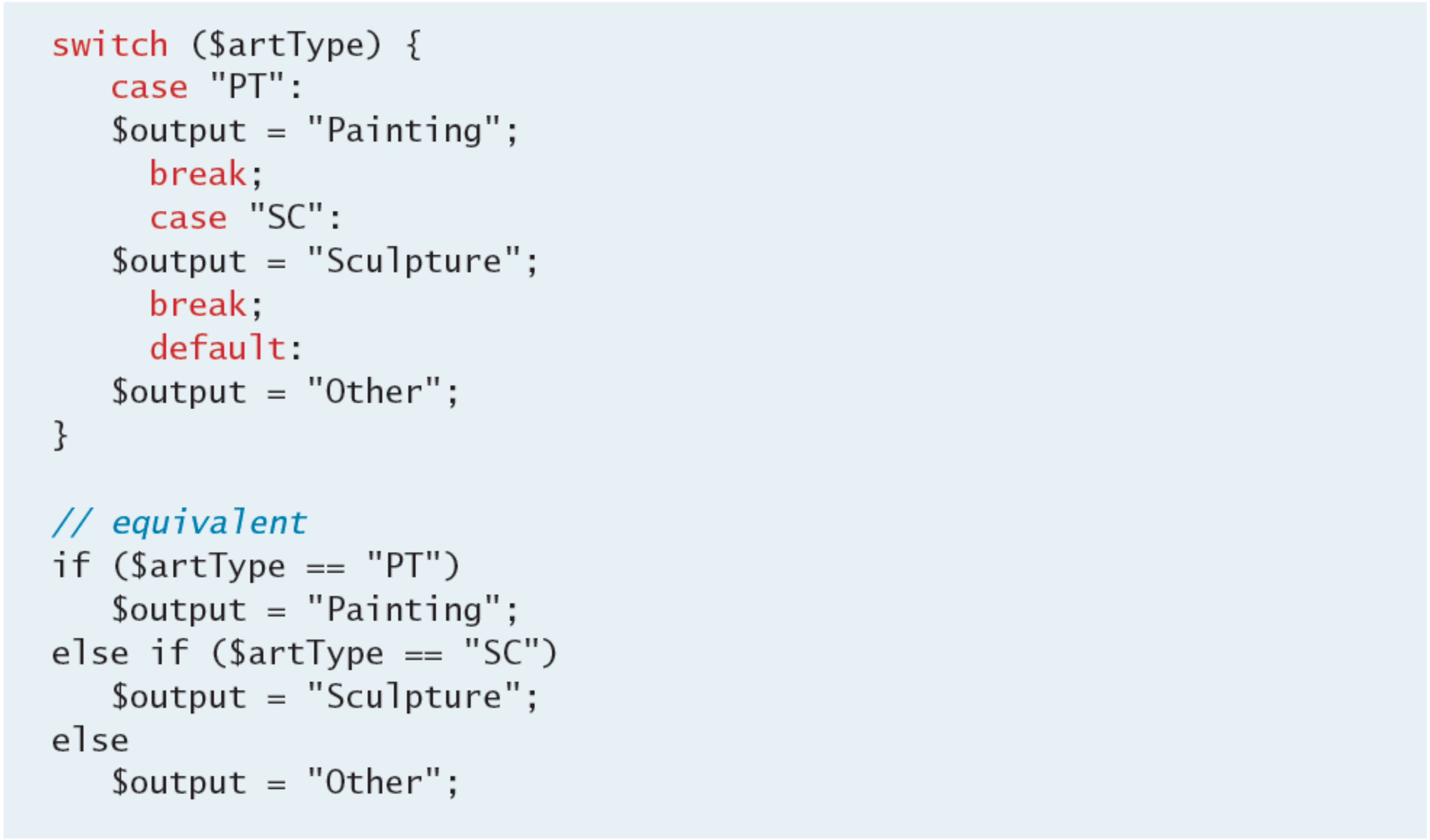
```
<?php if ($userStatus == "loggedin") {  ?>
   <a href="account.php">Account</a>
   <a href="logout.php">Logout</a>
<?php }  else { ?>
   <a href="login.php">Login</a>
   <a href="register.php">Register</a>
<?php } ?>


<?php
   // equivalent to the above conditional
   if ($userStatus == "loggedin") {
      echo '<a href="account.php">Account</a> ';
      echo '<a href="logout.php">Logout</a>';
   }
   else {
      echo '<a href="login.php">Login</a> ';
      echo '<a href="register.php">Register</a>';
   }
?>
```

LISTING 8.8 Combining PHP and HTML in the same script

switch…case

Nearly identical



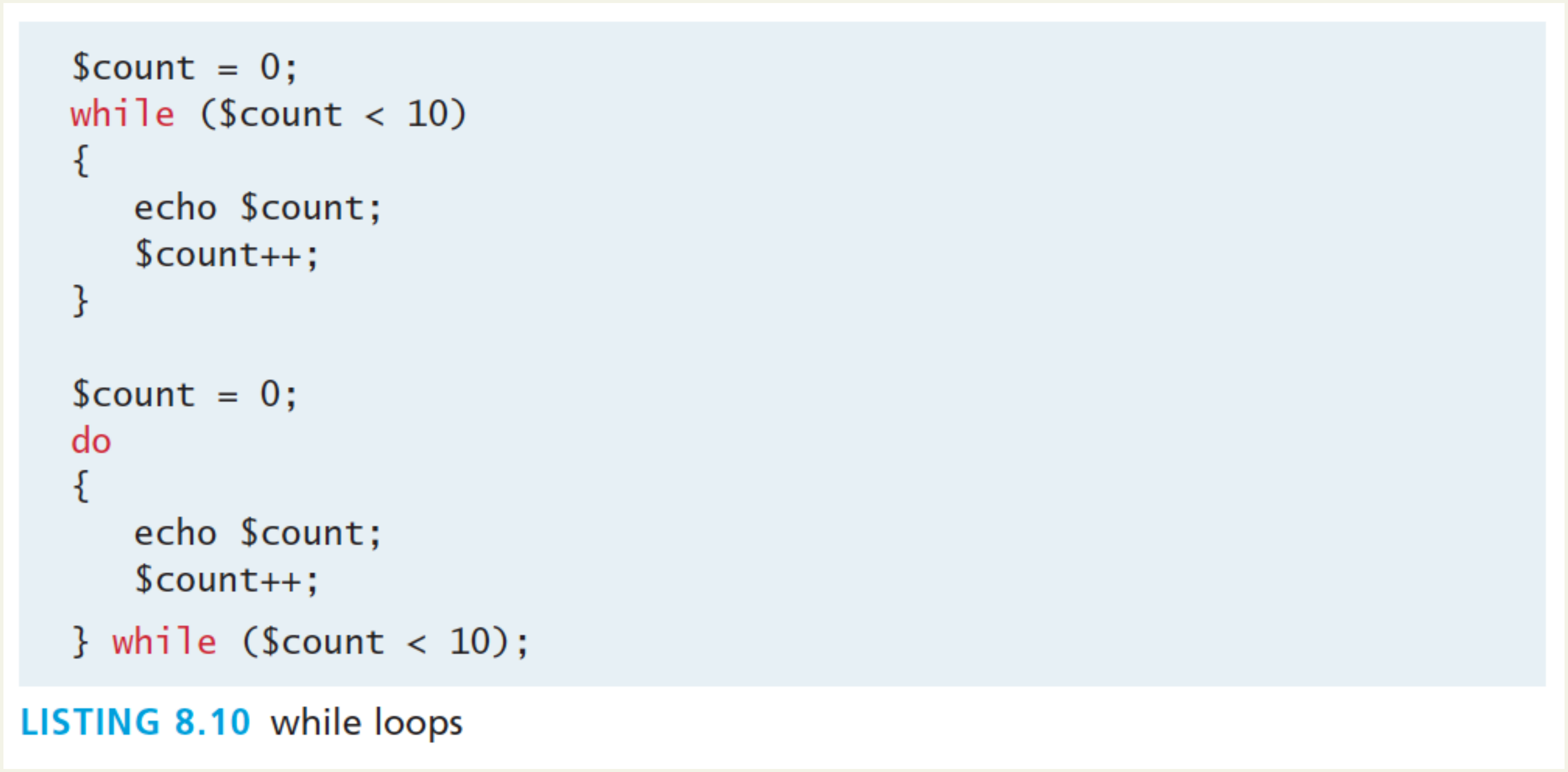
```
switch ($artType) {
   case "PT":
   $output = "Painting";
     break;
     case "SC":
   $output = "Sculpture";
     break;
     default:
   $output = "Other";
}

// equivalent
if ($artType == "PT")
   $output = "Painting";
else if ($artType == "SC")
   $output = "Sculpture";
else
   $output = "Other";
```

LISTING 8.9 Conditional statement using switch

while and do..while

Identical to other languages



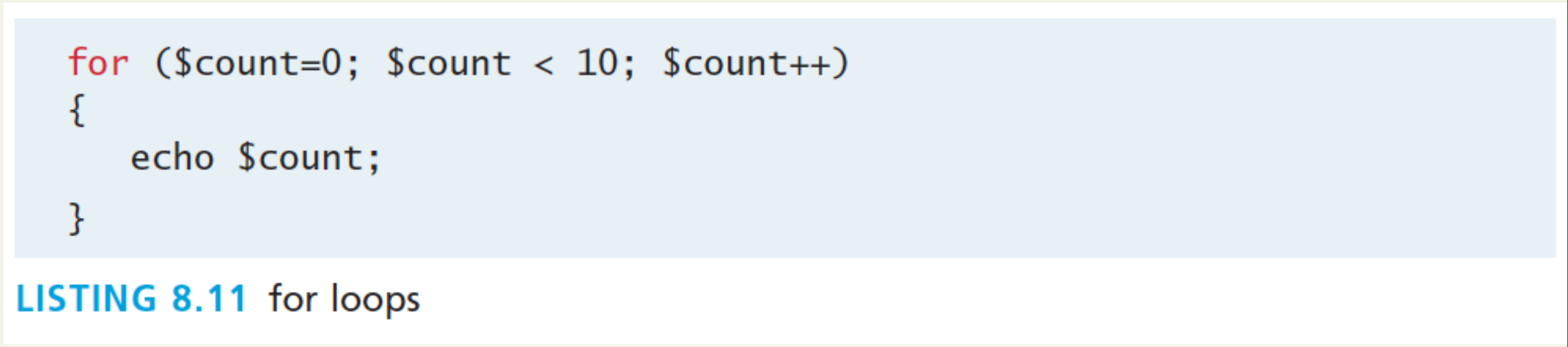
```
$count = 0;
while ($count < 10)
{
   echo $count;
   $count++;
}

$count = 0;
do
{
   echo $count;
   $count++;
} while ($count < 10);
```

LISTING 8.10 while loops

for

Identical to other languages



```
for ($count=0; $count < 10; $count++)
{
   echo $count;
}
```

LISTING 8.11 for loops

Alternate syntax for Control Structures

PHP has an alternative syntax for most of its control structures. In this alternate syntax

- the colon (:) replaces the opening curly bracket,
- while the closing brace is replaced with **endif;**, **endwhile;**, **endfor;**, **endforeach;**, or **endswitch;**

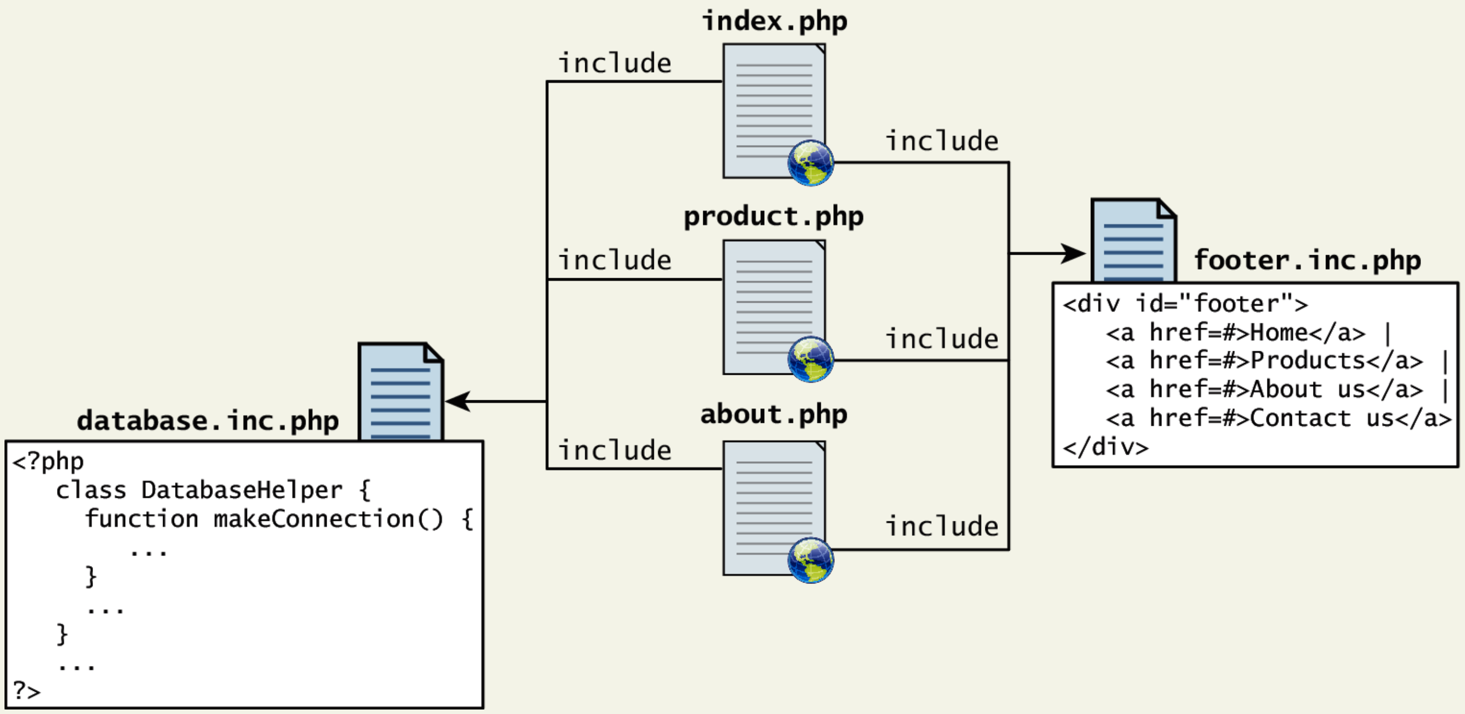
```
<?php if ($userStatus == "loggedin") :  ?>
   <a href="account.php">Account</a>
   <a href="logout.php">Logout</a>
<?php else : ?>
   <a href="login.php">Login</a>
   <a href="register.php">Register</a>
<?php endif; ?>
```

LISTING 8.12 Alternate syntax for control structures

Include Files

Organize your code

PHP does have one important facility that is the ability to include or insert content from one file into another.



```
<?php
   class DatabaseHelper {
     function makeConnection() {
        ...
     }
     ...
   }
   ...
?>
```

```
<div id="footer">
   <a href=#>Home</a> |
   <a href=#>Products</a> |
   <a href=#>About us</a> |
   <a href=#>Contact us</a>
</div>
```

Include Files

Organize your code

PHP provides four different statements for including files, as shown below.

include "somefile.php";

include_once "somefile.php";

require "somefile.php";

require_once "somefile.php";

With include, a warning is displayed and then execution continues. With require, an error is displayed and execution stops.

Include Files

Scope

Include files are the equivalent of copying and pasting.

- Variables defined within an include file will have the scope of the line on which the include occurs
- Any variables available at that line in the calling file will be available within the called file
- If the include occurs inside a function, then all of the code contained in the called file will behave as though it had been defined inside that function

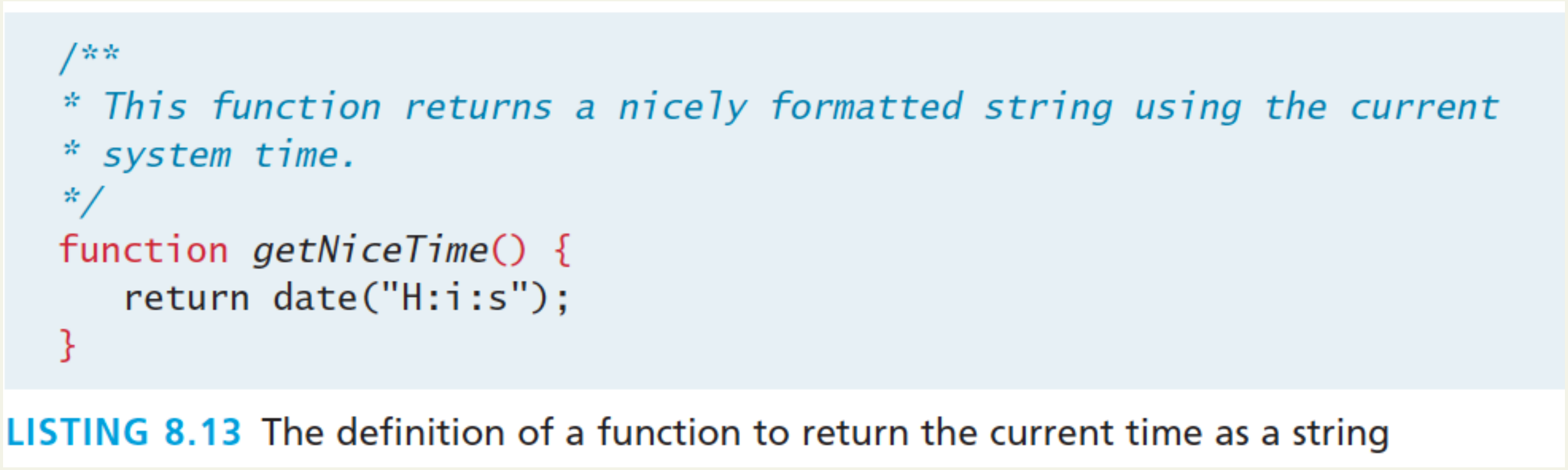
Section 5 of 5

FUNCTIONS

Functions

syntax

Just as with any language, writing code in the main (which in PHP is equivalent to coding in the markup between <?php and ?> tags) is not a good habit to get into.



```
/**
* This function returns a nicely formatted string using the current
* system time.
*/
function getNiceTime() {
   return date("H:i:s");
}
```

LISTING 8.13 The definition of a function to return the current time as a string

While the example function in Listing 8.13 returns a value, there is no requirement for this to be the case.

Functions

No return – no big deal.

```
/**
* This function outputs the footer menu
*/
function outputFooterMenu() {
   echo '<div id="footer">';
   echo '<a href=#>Home</a> | <a href=#>Products</a> | ';
   echo '<a href=#>About us</a> | <a href=#>Contact us</a>';
   echo '</div>';
}
```

LISTING 8.14 The definition of a function without a return value

Call a function

To call a function you must use its name with the () brackets.

Since *getNiceTime()* returns a string, you can assign that return value to a variable, or echo that return value directly, as shown below.

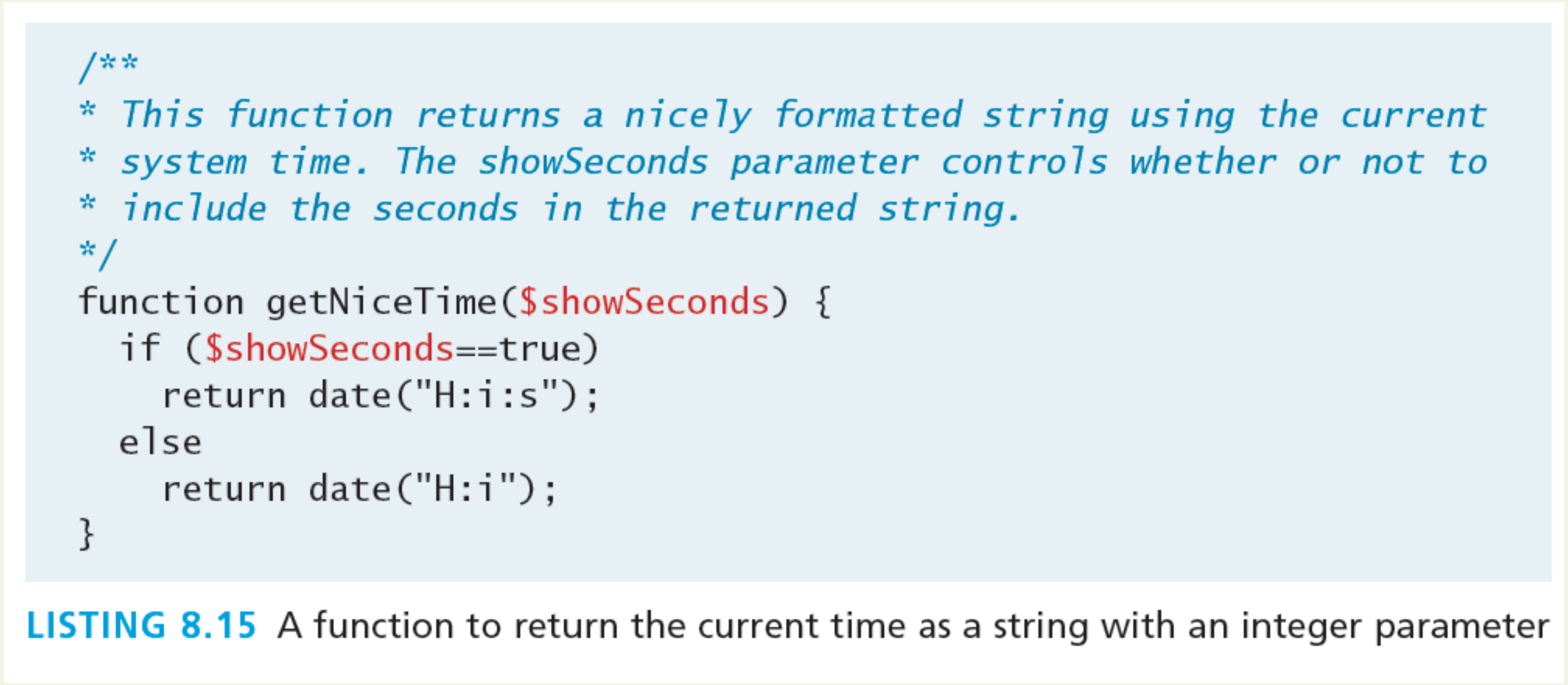
$output = getNiceTime();

echo getNiceTime();

If the function doesn't return a value, you can just call the function:

outputFooterMenu();

Parameters



```php
/**
* This function returns a nicely formatted string using the current
* system time. The showSeconds parameter controls whether or not to
* include the seconds in the returned string.
*/
function getNiceTime($showSeconds) {
  if ($showSeconds==true)
    return date("H:i:s");
  else
    return date("H:i");
}
```

LISTING 8.15 A function to return the current time as a string with an integer parameter

Thus to call our function, you can now do it in two ways:

```php
echo getNiceTime(true);  // this will print seconds
echo getNiceTime(false);  // will not print seconds
```

Parameter Default Values

```
<body>
<?php
function getNiceTime($showSeconds = true) {
  if($showSeconds)
    return date("H:i:s");
  else
    return date("H:i");
}
echo getNiceTime(), "<br>";
echo getNiceTime(true), "<br>";
echo getNiceTime(false);
?>
</body>
```

Now if you were to call the function with no values, the *$showSeconds* parameter would take on the default value, which we have set to **true**, and return the string with seconds.

Pass Parameters by Value

By default, arguments passed to functions are **passed by value** in PHP. This means that PHP passes a copy of the variable so if the parameter is modified within the function, it does not change the original.

```php
<?php
function changeParameter($arg) {
  $arg += 300;
  echo "arg=$arg<br>"; // output: arg=315
}

$initial = 15;
echo "initial=$initial<br>"; // output: initial=15
changeParameter($initial);
echo "initial=$initial<br>"; // output: initial=15
?>
```

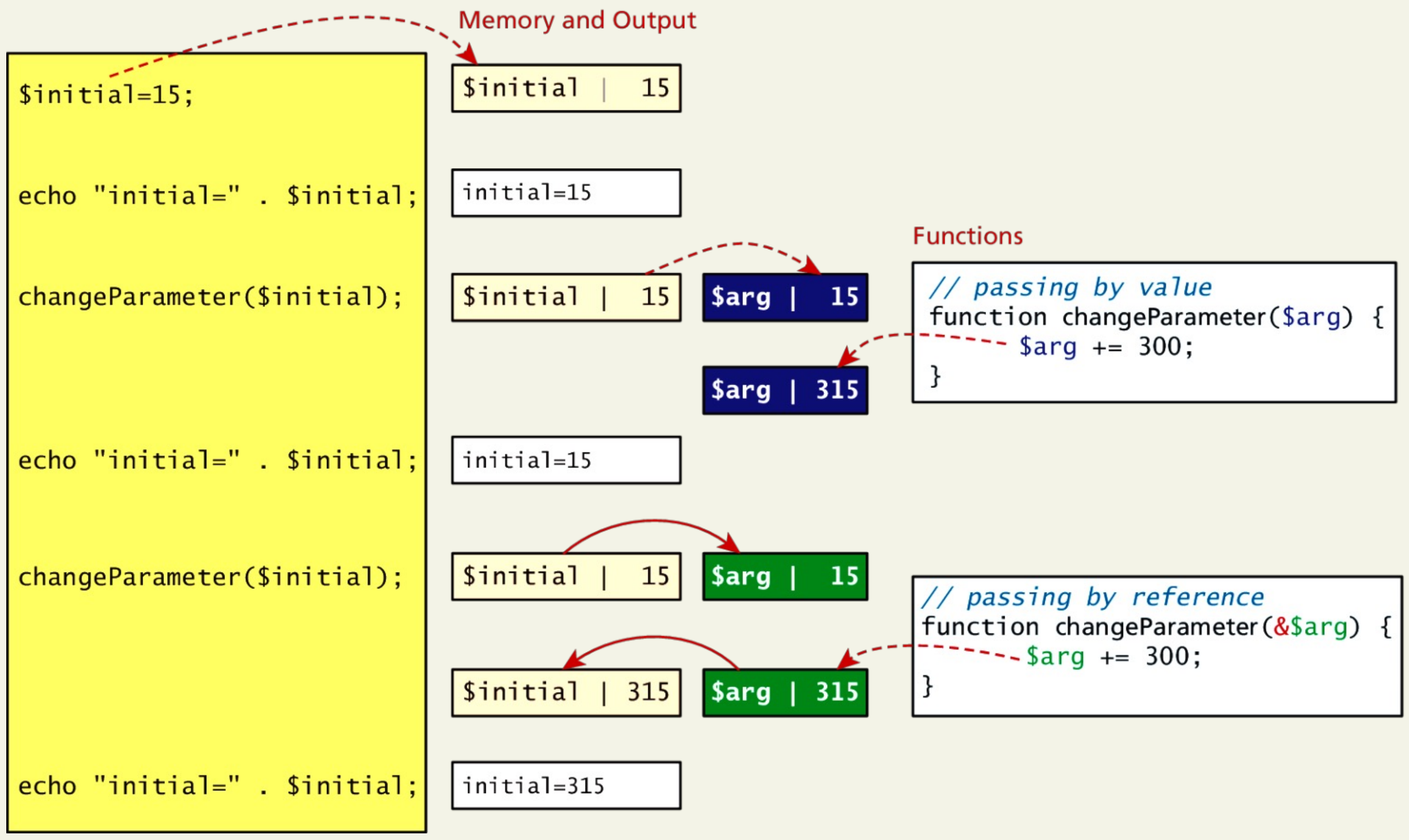
Pass Parameters by Reference

PHP also allows arguments to functions to be **passed by reference**, which will allow a function to change the contents of a passed variable.

The mechanism in PHP to specify that a parameter is passed by reference is to add an ampersand (**&**) symbol next to the parameter name in the function declaration

```
function changeParameter(&$arg) {
  $arg += 300;
  echo "arg=$arg<br>"; // output: arg=315
}
$initial = 15;
echo "initial=$initial<br>"; // output: initial=15
changeParameter($initial);
echo "initial=$initial<br>"; // output: initial=315
```

Value vs Reference



```php
$initial=15;

echo "initial=" . $initial;

changeParameter($initial);

echo "initial=" . $initial;

changeParameter($initial);

echo "initial=" . $initial;
```

```php
// passing by value
function changeParameter($arg) {
    $arg += 300;
}
```

```php
// passing by reference
function changeParameter(&$arg) {
    $arg += 300;
}
```

Variable Scope in functions

All variables defined within a function (such as parameter variables) have **function scope**, meaning that they are only accessible within the function.

Any variables created **outside** of the function in the main script are **unavailable** within a function.

```
$count= 56;

function testScope() {
        echo $count;     // outputs nothing or generates run-time
                         // warning/error

}

testScope();
echo $count;             // outputs 56
```

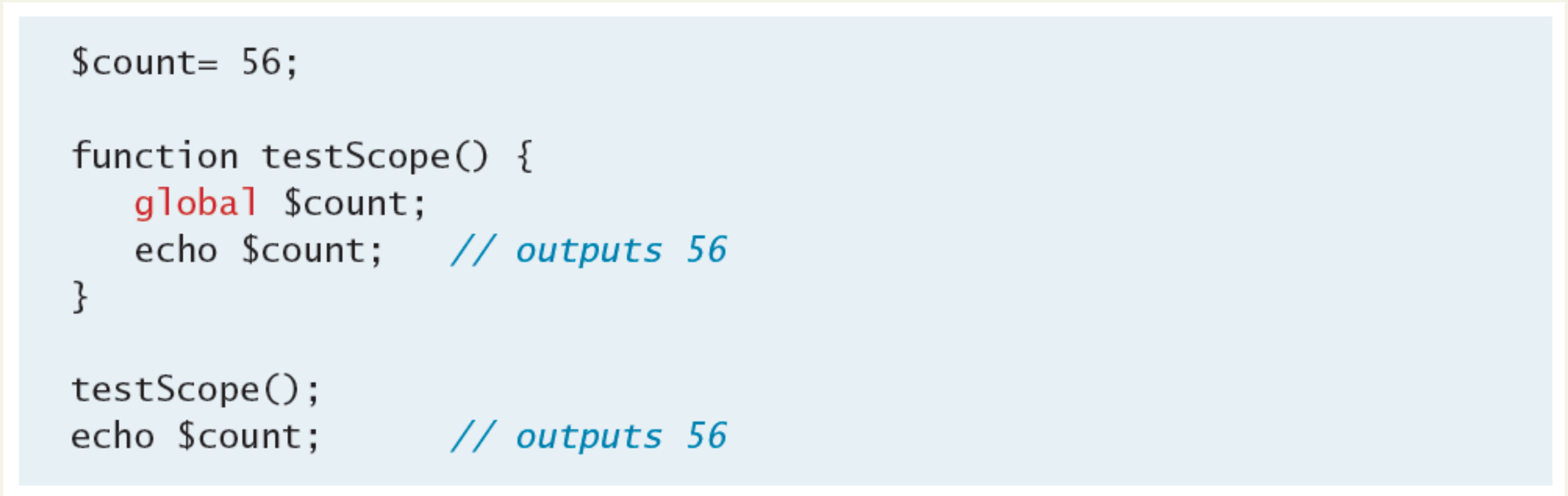
Global variables

Sometimes unavoidable

Variables defined in the main script are said to have **global scope.**

Unlike in other programming languages, a global variable is not, by default, available within functions.

PHP does allow variables with global scope to be accessed within a function using the **global** keyword



```
$count= 56;

function testScope() {
   global $count;
   echo $count;    // outputs 56
}

testScope();
echo $count;       // outputs 56
```

LISTING 8.19 Using the global keyword