Beginning PHP5

Beginning PHP5

Dave W. Mercer
Allan Kent
Steven D. Nowicki
David Mercer
Dan Squier
Wankyu Choi

with
Heow Eide-Goodman
Edward Lecky-Thompson
and Clark Morgan

WILEY

Wiley Publishing, Inc.

Beginning PHP5

Published by
Wiley Publishing, Inc.
10475 Crosspoint Boulevard
Indianapolis, IN 46256
www.wiley.com

Published simultaneously in Canada

CIP Data available upon request.

eISBN: 0-7645-7723-9

Manufactured in the United States of America

10 9 8 7 6 5 4 3 2 1

About the Authors

Dave W. Mercer

Dave W. Mercer has 15 years' experience in industrial and process engineering, and systems analysis, and is CTO for a B2B, responsible for the development and deployment of online automated business services. His entire site, hosting server, and the applications he builds for hosted clients are programmed in PHP using Postgres or MySQL as the database.

Allan Kent

Allan Kent is a PHP programmer who runs his own company and is a co-author of *Beginning PHP 4*. Allan has been programming seriously for the past dozen years and, other than the single blemish when he achieved a diploma in Cobol programming, is entirely self-taught.

Steven D. Nowicki

Steven D. Nowicki is Director of Software development at The Content Project, a Santa Monica, California–based consulting firm currently developing a massive enterprise resource planning and contact management system comprising more than 300,000 lines of OO PHP code. He has a decade of experience in large-scale software development and system architecture on all major platforms.

David Mercer

David Mercer is a PHP programmer and contributed to *Beginning PHP 4*. He has maintained a keen interest in all things open source ever since he managed to put together a working Beowulf cluster by nicking old computer parts from colleagues and assembling them under his desk. He has worked on Wrox open source titles about PHP, Perl, and Linux.

Dan Squier

Dan Squier is a longtime contributor to the Wrox community and a PHP programmer.

Wankyu Choi

Wankyu Choi is an accomplished PHP programmer and lead author of *Beginning PHP 4*. He holds a Master's degree in English/Korean interpretation and translation from the Graduate School of Translation & Interpretation.

Heow Eide-Goodman

Heow Eide-Goodman is a member of NYPHP and LispNYC who uses PHP in his day job to do Web sites, services, and back-office transformations among SQL Server, Interbase/Firebird, and MySQL.

Edward Lecky-Thompson

Edward Lecky-Thompson is the founder and director of Ashridge New Media, a professional new media technology consultancy based in Berkhamsted, just north of London, England. Self-described as "utterly obsessed with PHP," Ed has more than six years' experience in commercial software development and enterprise-level systems architecture across myriad platforms, with particularly strong exposure to PHP and Apache on Linux-based platforms.

Clark Morgan

Clark Morgan is an experienced programmer who creates and administers databases with Web sites using PHP and MySQL for Fusion Computing and Media.

This book has been a team effort and represents the efforts of many people. The current team of authors dedicates this book to the previous authors for their great introduction to earlier versions of PHP and to the editors and managers who worked so hard to help us build another great introduction to the new PHP5, especially Debra Williams Cauley, Maryann Steinhart, and David Mercer. We also thank the developers of PHP, the ZEND engine, and all the folks who've contributed the open-source software and code examples that make PHP the ideal scripting language for Web applications.

Credits

Acquisitions Editor
Debra Williams Cauley

Project Editor
Maryann Steinhart

Technical Editors
David Mercer
Heow Eide-Goodman

Production Editor
Eric Newman

Editorial Manager
Mary Beth Wakefield

Vice President & Executive Group Publisher
Richard Swadley

Vice President and Executive Publisher
Bob Ipsen

Vice President and Publisher
Joseph B. Wikert

Executive Editorial Director
Mary Bednarek

Contents

Introduction

PHP5 is the latest incarnation of PHP (PHP: Hypertext Preprocessor)—a programming language devised by Rasmus Lerdorf in 1994 for building dynamic, interactive Web sites. Since then, it's been evolving into a full-fledged language in its own right, thanks to the hard work of all the people who contribute to its development.

A sure sign that PHP is maturing as a technology is evidenced by its totally revamped and upgraded support for object-oriented programming (OOP) principles and improved support for XML. The Zend engine (the part that interprets and executes PHP code) now enables PHP5 developers to implement, among a host of other things, graceful application-wide error handling.

With all the new features and functionality that PHP5 provides, it's important for programmers to "upgrade" their understanding in order to best make use of this powerful Web scripting tool. And that's why it's important for you, the reader, to invest your time learning about the latest and greatest that the people developing PHP5 have to offer.

That's all well and good, but what exactly is PHP?

You know it's a language for writing computer programs, so the real question is "What *sort* of programs can you write with it?" In technical terms, PHP's main use is as a cross-platform, HTML-embedded, server-side Web scripting language. Let's take a moment to examine these terms:

- **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.
- **Server-side:** The PHP programs are run on a server—specifically a Web server.
- **Web-scripting language:** PHP programs run via a Web browser.

This means you'll write programs that mix PHP code and HTML, run them on a Web server, and access them from a Web browser that displays the result of your PHP processing by showing you the HTML returned by the Web server. In other words, you can make your programs available for other people to access across the Web, simply by placing them on a public Web server.

You're probably already familiar with HTML (HyperText Markup Language)—it's the main language used to create Web pages, combining plain text with special tags that tell browsers how to treat that text. HTML is used to describe how different elements in a Web page should be displayed, how pages should be linked, where to put images, and so on.

Pure HTML documents, for all their versatility, are little more than static arrangements of text and pictures, albeit nicely presented ones. However, most of the sites you find on the Web aren't static but

dynamic, even interactive. They can show you a list of articles containing a particular word in which you're interested, show you the latest news, even greet you by name when you log on. They enable you to interact, and present you with different information according to the choices you make.

You can't build a Web site like that using raw HTML, and that's where PHP comes in. What sort of things can you do with it? Well, you can program sites that

- Present data from a wide variety of sources, such as databases, files, or even other Web pages.
- Incorporate interactive elements, such as search facilities, message boards, and straw polls.
- Enable the user to perform actions, such as sending e-mail or buying something.

In other words, PHP can be used to write the sort of sites that those who regularly use the Web are likely to encounter every day. From search engines to information portals to e-commerce sites, most major Web sites incorporate some or all of these sorts of programming. Among other things in the course of this book, you'll use PHP to build

- A simple, online text editor
- A Web-based e-mail application
- An object-oriented contact manager application
- An object-oriented logging agent

So, PHP5 can be used for a diverse range of applications, from simple utilities such as a text editor to powerful Web applications such as the logging agent case study. This book equips you with the knowledge necessary to build any kind of Web site you want using PHP5. You'll learn some useful techniques along the way and perhaps pick up some ideas that you can incorporate into your own Web sites and applications.

> *Web scripting is certainly the mainstay of PHP's success, but it's not the only way to use the language. Command line scripting—using CLI (Command Line Interface), which was introduced in PHP4—is one of many popular applications of PHP. (CLI is covered in an appendix at the end of this book). Client-side graphical user interface application development using GTK (Gnome ToolKit) is another.*

Why PHP?

One of the best things about PHP is the large number of Internet service providers (ISPs) and Web hosting companies that support it. Today there are hundreds of thousands of developers using PHP, and it's not surprising that there are so many, considering that several million sites are reported to have PHP installed.

You already know that PHP is a cross-platform technology and that once you've written your Web page, it's easy to get it up and running on our Web server, but how does PHP compare with other technologies out there? Well, comparing PHP with Perl is a bit tricky because they were designed for different things. PHP was specifically designed to rapidly create dynamic Web content; Perl was not. As a result, Perl can sometimes be a complicated language that can become prohibitive for users who want to create Web pages. Comparing PHP with ASP is a more balanced comparison, but then you have to pay for ASP, and ASP doesn't work well on a variety of platforms—it needs to be used on other proprietary platforms for which you also must pay.

You may ask, "Is there a downside to PHP?" In the past, PHP has been criticized for the way it handled a number of things—for example, one of its main stumbling blocks was the way in which it implemented object support. However, PHP5 has taken stock of the downfalls of its predecessors and, where necessary, has completely rewritten the way in which it implements its functionality. Now more than ever, PHP is a serious contender for large-scale enterprise developments as well as having a large, consolidated base of small- to medium-sized applications.

Who This Book Is For

As you've probably guessed from its title, this book is intended for people who are just starting out with PHP5. That covers a pretty broad range of folks, from complete beginners who want to start writing programs for the very first time to experienced, battle-hardened developers who want to find out what's possible in the latest version of PHP.

We've tried wherever possible to make the book's content equally useful to readers all across the spectrum, but there are inevitably some sections that will be of more use to one group than they will to another. For example, people who already have PHP5 up and running on their machines could happily skip Chapter 1, which is all about just that—getting PHP up and running.

What This Book Covers

The main intention of this book is to ensure that the reader gains a broad understanding of PHP5 and its associated technologies and topics. A wide range of issues is discussed in the book, so you should consider looking at more focused books if you are concerned about specific issues only. For example, if you are mainly interested in how to use PHP5 and MySQL, you'll probably be best off looking at something that focuses specifically on that aspect of PHP. Further, if you are already well versed in all things PHP-ish, then you may want to consider looking at *Professional PHP5 Programming* (Wrox), which goes into more detail and covers more advanced topics.

Here's a quick list of some of the more important topics discussed in this book:

- Installation and setup
- Fundamentals—variables, loops, decisions, arrays
- Programming technique and good practices—creating and maintaining efficient, robust code
- Working with data
- File and directory handling
- XML
- PHP5 and databases—specifically MySQL and SQLite
- Object-oriented programming
- PEAR
- E-mail
- Graphics
- CLI

There's much more, of course, but this list should give you a good idea of what you can expect to get out of *Beginning PHP5*.

How This Book Is Structured

To provide you with an effective learning tool, every effort has been made to present all the information in a logical, consistent manner. This means that, unless absolutely necessary, you won't encounter new concepts or topics without having been introduced to them first. For example, if we're discussing how to create a class method, you will already have been introduced to the idea of functions, so that you won't have to waste time jumping from one place to another to understand the discussion.

To this end, the first few chapters provide a grounding in the fundamentals of PHP—how to get hold of it, how to get it working, and how to do some basic things with it. Afterward, a certain level of knowledge is assumed, and you move on to more advanced topics.

Here's a chapter-by-chapter breakdown to help you to decide whether to read the book cover to cover, or simply dip in here and there. If you're already PHP-savvy, of course, you'll just want to dive into those chapters that are of most interest to you.

Chapter 1 gets you up and running. A brief discussion of PHP is followed by step-by-step instructions for installing it on Linux or Windows platforms. You also learn how to install and configure a Web server (IIS or Apache), so that you can view your PHP5 pages from a Web browser. In the event that you run into problems during this setup and configuration stage, a helpful troubleshooting and debugging section is provided at the end of the chapter.

Chapter 2 presents a small PHP5 script in action. There's a discussion of Internet protocols and HTTP and how PHP relates to them; and with the example script as a base, you get a broad look at how PHP fits into the bigger picture. Then we start drilling down to the basics of the language: Variables, data types, and expressions all get introduced in the second half of this chapter.

Chapter 3 demonstrates how you can use the information supplied as part of HTTP requests and responses to collect useful data on a variety of programming environment aspects such as server information, `GET` and `POST` methods, cookies, and more. Plenty of practical examples later in the chapter show how you can use HTML forms and form elements to collect information for use in your PHP scripts. HTTP, sessions, and state are other topics in this chapter.

Chapter 4 deals with one of the most critical topics in any language: programming logic. You'll explore comparison and logical operators and the various statements, such as `if` and `switch`, that make use of them. Just as important are loops, which enable you to perform repetitive tasks with relative ease. With a good grasp of these fundamentals, you tackle the slightly more complex concept of arrays and learn how you can use loops, among other things, to manipulate the data contained within them.

Chapter 5 makes a pre-emptive strike at bad programming practices by discussing debugging and testing and how you can use PHP5 to handle errors gracefully. The chapter provides useful examples of how to maintain efficient, robust, and healthy code and also shows you how form validation can be used to keep your programs clear of erroneous or nonsensical data.

Chapter 6 is nominally related to the preceding chapter in that you explore good programming practices. You examine all the aspects involved in producing high-quality code, from code design, optimization and

presentation, to modularization (writing user-defined functions) and important related concepts such as scope.

Chapter 7 deals comprehensively with PHP's capability to work with files and directories. Opening, closing, reading, and writing to files are all basic requirements of many PHP programs, and these, along with other important issues such as file ownership and permissions, are detailed here. You build a text file editor to demonstrate the functionality you learned through the course of the chapter.

Chapter 8 illustrates the exciting world of XML, which is rapidly becoming the most popular method for transmitting structured information wherever there's a need to transmit data. PHP5 is endowed with new functionality so that it works faster and more efficiently with XML. Plenty of practical examples ensure that you get all the requisite information to take full advantage of XML in your applications.

Chapters 9–11 provide an excellent guide to storing, manipulating, and retrieving data using PHP and an RDBMS (Relational DataBase Management System)—in this case, MySQL. You'll install MySQL and then use it to explore the fundamentals of relational database architecture. You'll learn the basic connection functions that PHP5 supplies in order to perform many of the necessary basic functions such as creating and connecting to databases and inserting, retrieving, updating, and altering information. You put your knowledge to work by developing related programs through these three data chapters.

Chapters 12 and 13 discuss one of the most important concepts in programming today: objects. Chapter 12 introduces you to the theory behind object-oriented programming (OOP) and examines important topics such as inheritance, encapsulation, and abstraction, providing examples to demonstrate how PHP5 supports these concepts. Chapter 13 then takes a more pragmatic approach and rounds off the OO topic by leading you through an object-oriented application designed with the aid of UML, the Unified Modeling Language, which is itself explained in some detail at the beginning of the chapter.

Chapter 14 introduces the excellent PHP Extension and Application Repository (PEAR). You learn to use PEAR to find and use the packages it provides to add functionality to your own applications. You'll develop a couple of applications that use PEAR packages to complete various tasks.

Chapter 15 delves into the topic of e-mail. After presenting a bit of background on e-mail and its associated protocols and technologies, the chapter shows you how PHP5 supports e-mail functionality. Multipurpose Internet Mail Extensions (MIME) messages are also discussed, and a sample application provides a good practical demonstration for enabling e-mail attachments using MIME.

Chapter 16 shows you how to make use of the GD image library to render graphics for your Web pages. You'll be creating and manipulating images, drawing shapes, and working with text in GD.

Chapter 17 pulls together everything you learned in the book and challenges you to use your knowledge to develop a larger application. It leads you through the entire process of building a complex solution like the logging agent case study that's created in the course of this chapter.

Appendix A provides the answers to the exercises presented in most chapters.

Appendix B presents a PHP5 function reference, which is sure to come in handy given the large amount of native PHP functions milling around.

Appendix C discusses the SQLite database that's bundled with PHP5. It includes practical examples of SQLite in action.

Appendix D gives you a rundown on Open Database Connectivity (ODBC)—what it is, what it does, and how you can use it to connect to the various SQL-based databases. Specifically, you look at setting up ODBC with MySQL and see how you can use PHP5's ODBC functions to connect to MySQL.

Appendix E explains how you can use the PHP Command Line Interpreter to perform non–Web-based tasks using PHP5. It enables PHP5 users to open the door to command-line jobs that were previously the domain of Perl, BASH, and DOS users.

Appendix F provides an in-depth look at the `php.ini` file, which contains all of the configuration settings for PHP5. This appendix is a useful reference for any of the more advanced settings (which aren't discussed in the book) that you may need to modify at some stage during the course of your programming.

What You Need to Use This Book

As already noted, PHP can be run on a broad range of different operating systems, including Windows, Linux, Mac OS X, and more. The first step is therefore to get a version of PHP that's suitable for your own OS. (You'll find full instructions for downloading and installing the correct version of PHP5 in Chapter 1.)

You'll need a text editor of some kind to create and edit your scripts.

You also need a Web server. Apache is always a good bet, particularly on UNIX machines (although, it works just fine on Windows, too). Apache is included with most Linux distributions and Mac OS X as well, and if you don't already have a copy, note that it's available for download from `www.apache.org`. Oh, and it's free.

Windows users may find it easier to work with Microsoft's own Web server, IIS (Internet Information Server), which is included with Windows 2000 and most versions of XP (not XP Home), although it won't necessarily be installed by default. If you're using Windows 98, there's a cut-down version of IIS called PWS (Personal Web Server), which does a perfectly adequate job for small projects. If you're aiming to build big, though, Apache's probably a better option in the long run. And if you're running Windows ME or XP Home edition (neither of which support PWS or IIS), Apache is definitely the way to go.

Using the Command Line

If you're using a Windows PC or Macintosh computer, it's quite possible that you won't be familiar with the command-line interface or "shell." This is a powerful tool for communicating with your system, and one that you'll be relying on in a number of chapters throughout the book.

Before rich graphical environments came into common use, and drag-and-drop interfaces were virtually unheard of, the only way to interact with computers was to type commands, one line at a time. You wanted to run a program? There was no icon to click—you typed the program's name.

Many programs still make use of the command-line interface. Why? For one thing, it's a lot simpler to write them that way, and even now, many powerful utilities and applications are written exclusively for use via the command line. For another, many people still find it easier to interact with the command prompt than with a mouse-driven windowed environment.

For example, when you begin looking at databases in Chapter 9, you make use of the MySQL database manager, which you configure by typing instructions at the command line.

So first of all, you need to access the command line interface:

- On Windows, look in the Start menu for an entry (normally under Programs ➪ Accessories) named Command Prompt or MS-DOS Prompt. Alternatively, press Windows+R to call up the Run dialog, type `cmd`, and click OK.
- On UNIX (including variations such as Linux and Mac OS X), look for a program with a name such as console, terminal, konsole, xterm, eterm, or kterm. These are all widely used shell programs that can be found on a broad range of UNIX-based systems.

After you've called up the interface, you'll probably be confronted by a nearly blank screen, with just a snippet of text such as one of these:

```
$
%
C:/>
#
bash$
```

This is a *command prompt*, which is simply there to let you know that the interface is ready to receive instructions—prompting you for commands, in effect. It doesn't really matter what the prompt looks like, just that you recognize it when it appears. In this book, the prompt is designated this way:

```
>
```

Whenever you have to type instructions at the command line, we show those instructions immediately following the prompt (>), nearly always in the first line. What the computer generates by itself follows:

```
> mysqlshow
+-----------+
| Databases |
+-----------+
| mysql     |
| test      |
+-----------+
```

Conventions

To help you get the most from the text and keep track of what's happening, we've used a number of conventions throughout the book.

Boxes like this one hold important, not-to-be-forgotten information that is directly relevant to the surrounding text.

Notes, tips, and asides to the current discussion are offset and placed in italics like this.

As for styles in the text:

- We *highlight* important words when we introduce them
- We show keyboard strokes like this: Ctrl+A
- We show file names, URLs, and code within the text like so: `persistence.properties`
- We present code in two different ways:

```
In code examples we highlight new and important code with a gray background.
```

```
The gray highlighting is not used for code that's less important in the
present context, or has been shown before.
```

Source Code

As you work through the examples in this book, you may choose either to type all the code manually or to use the source code files that accompany the book. All of the source code used in this book is available for download at `www.wrox.com`. At the site, simply locate the book's title (either by using the Search box or by using one of the title lists) and click the Download Code link on the book's detail page to obtain all the source code for the book.

> *Because many books have similar titles, you may find it easiest to search by ISBN; this book's ISBN is 0-7645-5783-1.*

Once you download the code, just decompress it with your favorite compression tool. Alternatively, you can go to the main Wrox code download page at `www.wrox.com/dynamic/books/download.aspx` to see the code available for this book and all other Wrox books.

Errata

We make every effort to ensure that there are no errors in the text or in the code. However, no one is perfect, and mistakes do occur. If you find an error in one of our books, like a spelling mistake or faulty piece of code, we would be very grateful for your feedback. By sending in errata you may save another reader hours of frustration, and at the same time you will be helping us provide even higher quality information.

To find the errata page for this book, go to `www.wrox.com` and locate the title using the Search box or one of the title lists. Then, on the book details page, click the Book Errata link. On this page you can view all errata that has been submitted for this book and posted by Wrox editors. A complete book list including links to each's book's errata is also available at `www.wrox.com/misc-pages/booklist.shtml`.

If you don't spot "your" error on the Book Errata page, go to `www.wrox.com/contact/techsupport.shtml` and complete the form there to send us the error you have found. We'll check the information and, if appropriate, post a message to the book's errata page and fix the problem in subsequent editions of the book.

p2p.wrox.com

For author and peer discussion, join the P2P forums at `p2p.wrox.com`. The forums are a Web-based system for you to post messages relating to Wrox books and related technologies and interact with other readers and technology users. The forums offer a subscription feature to e-mail you topics of interest of your choosing when new posts are made to the forums. Wrox authors, editors, other industry experts, and your fellow readers are present on these forums.

At `http://p2p.wrox.com` you will find a number of different forums that will help you not only as you read this book, but also as you develop your own applications. To join the forums, just follow these steps:

1. Go to `p2p.wrox.com` and click the Register link.
2. Read the terms of use and click Agree.
3. Complete the required information to join as well as any optional information you wish to provide and click Submit.
4. You will receive an e-mail with information describing how to verify your account and complete the joining process.

You can read messages in the forums without joining P2P but in order to post your own messages, you must join.

Once you join, you can post new messages and respond to messages other users post. You can read messages at any time on the Web. If you would like to have new messages from a particular forum e-mailed to you, click the Subscribe to this Forum icon by the forum name in the forum listing.

For more information about how to use the Wrox P2P, be sure to read the P2P FAQs for answers to questions about how the forum software works as well as many common questions specific to P2P and Wrox books. To read the FAQs, click the FAQ link on any P2P page.

1

Getting Up and Running

PHP, which stands for HyperText Preprocessor, is widely used for creating programmed features for Web sites because it is easy to learn and also because PHP syntax is drawn from other widely used languages, making it familiar to many programmers. In this chapter we present a very brief history of PHP, and then discuss the nature of PHP as it relates to the Web.

Before you can get into the nitty-gritty of programming with PHP5, you need a clear understanding of how PHP programs work across the Web, and that obviously implies knowledge of the Web protocol called HyperText Transfer Protocol (HTTP). HTTP is the language or format for communications from browser to Web server and back, and is therefore fundamental to many aspects of PHP. HTTP gets some coverage in this chapter, and quite a bit more in Chapter 3.

You'll see how to properly setup PHP on a Linux server, and on a Windows server as well. PHP programs run in conjunction with Web pages, which in turn run (or are distributed by) Web server software (such as Apache or IIS), which in turn run on top of an operating system (such as Linux or Windows). Although it's not strictly necessary to know everything about network operating systems to build good PHP programs, there are many aspects of PHP that are controlled or affected by the Web server. If you're unfamiliar with server computers, Web servers, and the like, don't worry. You'll soon see how they work, and look at the requirements and process of installing basic Web server software.

This chapter leads you through installing PHP on a Red Hat Linux machine running Apache, and through installing PHP on a Windows 2000 machine running IIS. Just pick the one that's right for you.

You'll also examine the contents of the PHP configuration file `php.ini` with you, and test your PHP installation.

Obviously there's a lot of work for you in this chapter, so let's get started.

The Roots of PHP

PHP is a programming language designed to work with HTML, but unlike HTML, PHP has data processing capabilities. If you are familiar with HTML, you know that it is not really a programming language, but more of a rendering language—that is, HTML enables you to write Web pages using

code that creates a pleasing (hopefully) display of text, graphics, and links within a browser. Although there are a few helpful features of HTML (such as the capability to cause a form submission), for the most part HTML does nothing programmatically. For example, there are no HTML commands that enable you to add two numbers together, or access a database.

If you remember the Web back in the early '90s, you may recall that early Web pages were made from HTML code written as plain text source files. When you made a connection to a Web site with your browser, the Web server software sent these plain text HTML files to be processed and rendered into Web pages. Your browser actually did the rendering process (and still does, to be sure), but if you clicked View ➪ Page Source, you'd see the raw HTML code.

Javascript (and a few other almost unknown programming languages) improved the situation for Web designers in that it provided for programmatic functionality within Web pages. However, it was limited to programmatic functionality on the user's computer, not on the back-end (on the Web server), where all the really cool data processing and database access takes place. Practical Extraction and Reporting Language (PERL) was one of the first widely used languages for programming on the back-end, but has limitations of its own, such as an inability to be mixed in with HTML for easy in-page programming.

So where does PHP fit in with HTML? PHP began as PHP/FI, developed in 1995 by Rasmus Lerdorf from some Perl scripts he had created for tracking accesses to his online resume. Eventually, Rasmus wrote an implementation in C, released the source code to the public, and by the beginning of 1998 version 3.0 of PHP was released (written by Rasmus Lerdorf, Andi Gutmans, and Zeev Suraski), the first version that is very similar to the current releases of PHP.

The main goal of PHP is to enable users to easily develop dynamic Web pages. The difference between dynamic Web pages and static Web pages is that the content and structure of dynamic Web pages may change each time they are accessed (that's what the back-end programming is for) whereas the content and structure of static Web pages is fixed and does not change unless the designer manually changes them.

Unlike many other languages, PHP can be embedded directly into HTML, making it quite easy for those familiar with HTML to grasp how to add back-end, programmatic functionality to their Web pages. This single capability is one of the most important factors in PHP's usefulness, and thereby its popularity. But have no doubt that PHP is growing into a much more full-features language going well beyond the initial intentions of its authors. PHP intends to be the primary language for a great variety of online and offline applications, and PHP5 is showing every sign of doing just that.

And you shouldn't forget how well PHP works with HTTP (HyperText Transfer Protocol), the communications protocol (pre-agreed format for data communications) for Web. Whenever you click a link or enter a Web address into your browser, a request in HTTP format is sent to the Web server, which responds by sending back the Web page. If the Web page isn't found, you'll probably get the "404 Not Found" error. Sending back the correct page or sending an error if the page is not found are HTTP functions. We discuss HTTP thoroughly in Chapter 2 because several important aspects of PHP applications depend on HTTP.

Installing, Configuring, and Running PHP

Before you can write a PHP application that works with your Web pages, you need to have PHP installed and configured. Because you'll be writing a Web application, it's a given that you'll need a Web server and some Web pages (a short HTML primer is provided in Chapter 3, although it's assumed that you