CHAPTER 1

# An Introduction to PHP

This chapter serves to better acquaint you with the basics of PHP, offering insight into its roots, popularity, and users. This information sets the stage for a discussion of PHP's feature set, including the new features in PHP 5. By the conclusion of this chapter, you'll learn:

- How a Canadian developer's Web page hit counter spawned one of the world's most popular scripting languages
- What PHP's developers have done to once again reinvent the language, making version 5 the best yet released
- Which features of PHP attract both new and expert programmers alike

## History

The origins of PHP date back to 1995, when an independent software development contractor named Rasmus Lerdorf developed a Perl/CGI script that enabled him to know how many visitors were reading his online résumé. His script performed two tasks: logging visitor information, and displaying the count of visitors to the Web page. Because the Web as we know it today was still young at that time, tools such as these were nonexistent, and they prompted e-mails inquiring about Lerdorf's scripts. Lerdorf thus began giving away his toolset, dubbed *Personal Home Page* (PHP).

The clamor for the PHP toolset prompted Lerdorf to continue developing the language, perhaps the most notable early change coming when he added a feature for converting data entered in an HTML form into symbolic variables, encouraging exportation into other systems. To accomplish this, he opted to continue development in C code rather than Perl. Ongoing additions to the PHP toolset culminated in November 1997 with the release of PHP 2.0, or Personal Home Page—Form Interpreter (PHP-FI). As a result of PHP's rising popularity, the 2.0 release was accompanied by a number of enhancements and improvements from programmers worldwide.

The new PHP release was extremely popular, and a core team of developers soon joined Lerdorf. They kept the original concept of incorporating code directly alongside HTML and rewrote the parsing engine, giving birth to PHP 3.0. By the June 1998 release of version 3.0, more than 50,000 users were using PHP to enhance their Web pages.

■ **Note** 1997 also saw the change of the words underlying the PHP abbreviation from Personal Home Page to the recursive acronym Hypertext Preprocessor.

Development continued at a hectic pace over the next two years, with hundreds of functions being added and the user count growing in leaps and bounds. At the beginning of 1999, Netcraft (`http://www.netcraft.com/`) reported a conservative estimate of a user base surpassing 1,000,000, making PHP one of the most popular scripting languages in the world. Its popularity surpassed even the greatest expectations of the developers, as it soon became apparent that users intended to use PHP to power far larger applications than was originally anticipated. Two core developers, Zeev Suraski and Andi Gutmans, took the initiative to completely rethink the way PHP operated, culminating in a rewriting of the PHP parser, dubbed the Zend scripting engine. The result of this work was found in the PHP 4 release.

■ **Note** In addition to leading development of the Zend engine and playing a major role in steering the overall development of the PHP language, Zend Technologies Ltd. (`http://www.zend.com/`), based in Israel, offers a host of tools for developing and deploying PHP. These include Zend Studio, Zend Encoder, and Zend Optimizer, among others. Check out the Zend Web site for more information.

## PHP 4

On May 22, 2000, roughly 18 months after the first official announcement of the new development effort, PHP 4.0 was released. Many considered the release of PHP 4 to be the language's official debut within the enterprise development scene, an opinion backed by the language's meteoric rise in popularity. Just a few months after the major release, Netcraft (`http://www.netcraft.com/`) estimated that PHP had been installed on more than 3.6 million domains.

### Features

PHP 4 included several enterprise-level improvements, including the following:

- **Improved resource handling:** One of version 3.X's primary drawbacks was scalability. This was largely because the designers underestimated how much the language would be used for large-scale applications. The language wasn't originally intended to run enterprise-class Web sites, and subsequent attempts to do so caused the developers to rethink much of the language's mechanics. The result was vastly improved resource-handling functionality in version 4.
- **Object-oriented support:** Version 4 incorporated a degree of object-oriented functionality, although it was largely considered an unexceptional implementation. Nonetheless, the new features played an important role in attracting users used to working with traditional object-oriented programming (OOP) languages. Standard class and object development methodologies were made available, in addition to object overloading, and run-time class information. A much more comprehensive OOP implementation has been made available in version 5, and is introduced in Chapter 5.

- **Native session-handling support:** HTTP session handling, available to version 3.X users through the third-party package PHPLIB (`http://phplib.sourceforge.net`), was natively incorporated into version 4. This feature offers developers a means for tracking user activity and preferences with unparalleled efficiency and ease. Chapter 15 covers PHP's session-handling capabilities.
- **Encryption:** The MCrypt (`http://mcrypt.sourceforge.net`) library was incorporated into the default distribution, offering users both full and hash encryption using encryption algorithms including Blowfish, MD5, SHA1, and TripleDES, among others. Chapter 18 delves into PHP's encryption capabilities.
- **ISAPI support:** ISAPI support offered users the ability to use PHP in conjunction with Microsoft's IIS Web server as an ISAPI module, greatly increasing its performance and security.
- **Native COM/DCOM support:** Another bonus for Windows users is PHP 4's ability to access and instantiate COM objects. This functionality opened up a wide range of interoperability with Windows applications.
- **Native Java support:** In another boost to PHP's interoperability, support for binding to Java objects from a PHP application was made available in version 4.0.
- **Perl Compatible Regular Expressions (PCRE) library:** The Perl language has long been heralded as the reigning royalty of the string parsing kingdom. The developers knew that powerful regular expression functionality would play a major role in the widespread acceptance of PHP, and opted to simply incorporate Perl's functionality rather than reproduce it, rolling the PCRE library package into PHP's default distribution (as of version 4.2.0). Chapter 9 introduces this important feature in great detail, and offers a general introduction to the often confusing regular expression syntax.

In addition to these features, literally hundreds of functions were added to version 4, greatly enhancing the language's capabilities. Throughout the course of this book, much of this functionality is discussed, as it remains equally important in the version 5 release.

### Drawbacks

PHP 4 represented a gigantic leap forward in the language's maturity. The new functionality, power, and scalability offered by the new version swayed an enormous number of burgeoning and expert developers alike, resulting in its firm establishment among the Web scripting behemoths. Yet maintaining user adoration in the language business is a difficult task; programmers often hold a "what have you done for me lately?" mindset. The PHP development team kept this notion close in mind, because it wasn't too long before it set out upon another monumental task, one that could establish the language as the 800-pound gorilla of the Web scripting world: PHP 5.

## PHP 5

Version 5 is yet another watershed in the evolution of the PHP language. Although previous major releases had enormous numbers of new library additions, version 5 contains improvements over existing functionality and adds several features commonly associated with mature programming language architectures:

- **Vastly improved object-oriented capabilities:** Improvements to PHP's object-oriented architecture is version 5's most visible feature. Version 5 includes numerous functional additions such as explicit constructors and destructors, object cloning, class abstraction, variable scope, interfaces, and a major improvement regarding how PHP handles object management. Chapters 6 and 7 offer thorough introductions to this topic.

- **Try/catch exception handling:** Devising custom error-handling strategies within structural programming languages is, ironically, error-prone and inconsistent. To remedy this problem, version 5 now supports exception handling. Long a mainstay of error management in many languages, C++, C#, Python, and Java included, exception handling offers an excellent means for standardizing your error-reporting logic. This new and convenient methodology is introduced in Chapter 8.

- **Improved string handling:** Prior versions of PHP have treated strings as arrays by default, a practice indicative of the language's traditional loose-knit attitude toward datatypes. This strategy has been tweaked in version 5, in which a specialized string offset syntax has been introduced, and the previous methodology has been deprecated. The new features, changes, and effects offered by this new syntax are discussed in Chapter 9.

- **Improved XML and Web Services support:** XML support is now based on the libxml2 library, and a new and rather promising extension for parsing and manipulating XML, known as SimpleXML, has been introduced. In addition, a SOAP extension is now available. In Chapter 20, these two new extensions are introduced, along with a number of slick third-party Web Services extensions.

- **Native support for SQLite:** Always keen on choice, the developers have added support for the powerful yet compact SQLite database server (`http://www.sqlite.org/`). SQLite offers a convenient solution for developers looking for many of the features found in some of the heavyweight database products without incurring the accompanying administrative overhead. PHP's support for this powerful database engine is introduced in Chapter 22.

A host of other improvements and additions are offered in version 5, many of which are introduced, as relevant, throughout the book.

With the release of version 5, PHP's prevalence is at a historical high. At press time, PHP has been installed on almost 19 million domains (Netcraft, `http://www.netcraft.com/`). According to E-Soft, Inc. (`http://www.securityspace.com/`), PHP is by far the most popular Apache module, available on almost 54 percent of all Apache installations.

So far, this chapter has discussed only version-specific features of the language. Each version shares a common set of characteristics that play a very important role in attracting and retaining a large user base. In the next section, you'll learn about these foundational features.

# General Language Features

Every user has his or her own specific reason for using PHP to implement a mission-critical application, although one could argue that such motives tend to fall into four key categories: *practicality*, *power*, *possibility,* and *price.*

## Practicality

From the very start, the PHP language was created with practicality in mind. After all, Lerdorf's original intention was not to design an entirely new language, but to resolve a problem that had no readily available solution. Furthermore, much of PHP's early evolution was not the result of the explicit intention to improve the language itself, but rather to increase its utility to the user. The result is a *minimalist* language, both in terms of what is required of the user and in terms of the language's syntactical requirements. For starters, a useful PHP script can consist of as little as one line; unlike C, there is no need for the mandatory inclusion of libraries. For example, the following represents a complete PHP script, the purpose of which is to output the current date, in this case one formatted like `September 23, 2005`:

```
<?php echo date("F j, Y");?>
```

Another example of the language's penchant for compactness is its ability to nest functions. For example, you can effect numerous changes to a value on the same line by stacking functions in a particular order, in the following case producing a pseudorandom string of five alphanumeric characters, `a3jh8` for instance:

```
$randomString = substr(md5(microtime()), 0, 5);
```

PHP is a loosely typed language, meaning there is no need to explicitly create, typecast, or destroy a variable, although you are not prevented from doing so. PHP handles such matters internally, creating variables on the fly as they are called in a script, and employing a best-guess formula for automatically typecasting variables. For instance, PHP considers the following set of statements to be perfectly valid:

```
<?php
   $number = "5";          # $number is a string
   $sum = 15 + $number;    # Add an integer and string to produce integer
   $sum = "twenty";        # Overwrite $sum with a string.
?>
```

PHP will also automatically destroy variables and return resources to the system when the script completes. In these and in many other respects, by attempting to handle many of the administrative aspects of programming internally, PHP allows the developer to concentrate almost exclusively on the final goal, namely a working application.

## Power

The earlier introduction to PHP 5 alluded to the fact that the new version is more qualitative than quantitative in comparison to previous versions. Previous major versions were accompanied by enormous additions to PHP's default libraries, to the tune of several hundred new functions per release. Presently, 113 libraries are available, collectively containing well over 1,000 functions. Although you're likely aware of PHP's ability to interface with databases, manipulate form information, and create pages dynamically, you might not know that PHP can also do the following:

- Create and manipulate Macromedia Flash, image, and Portable Document Format (PDF) files
- Evaluate a password for guessability by comparing it to language dictionaries and easily broken patterns
- Communicate with the Lightweight Directory Access Protocol (LDAP)
- Parse even the most complex of strings using both the POSIX and Perl-based regular expression libraries
- Authenticate users against login credentials stored in flat files, databases, and even Microsoft's Active Directory
- Communicate with a wide variety of protocols, including IMAP, POP3, NNTP, and DNS, among others
- Communicate with a wide array of credit-card processing solutions

Of course, the coming chapters cover as many of these and other interesting and useful features of PHP as possible.

## Possibility

PHP developers are rarely bound to any single implementation solution. On the contrary, a user is typically fraught with choices offered by the language. For example, consider PHP's array of database support options. Native support is offered for over 25 database products, including Adabas D, dBase, Empress, FilePro, FrontBase, Hyperwave, IBM DB2, Informix, Ingres, Interbase, mSQL, direct MS-SQL, MySQL, Oracle, Ovrimos, PostgreSQL, Solid, Sybase, Unix dbm, and Velocis. In addition, abstraction layer functions are available for accessing Berkeley DB–style databases. Finally, two database abstraction layers are available, one called the dbx module, and another via PEAR, titled the PEAR DB.

PHP's powerful string-parsing capabilities is another feature indicative of the possibility offered to users. In addition to more than 85 string-manipulation functions, both POSIX- and Perl-based regular expression formats are supported. This flexibility offers users of differing skill sets the opportunity not only to immediately begin performing complex string operations but also to quickly port programs of similar functionality (such as Perl and Python) over to PHP.

Do you prefer a language that embraces functional programming? How about one that embraces the object-oriented paradigm? PHP offers comprehensive support for both. Although PHP was originally a solely functional language, the developers soon came to realize the importance of offering the popular OOP paradigm, and took the steps to implement an extensive solution.

The recurring theme here is that PHP allows you to quickly capitalize on your current skill set with very little time investment. The examples set forth here are but a small sampling of this strategy, which can be found repeatedly throughout the language.

## Price

Since its inception, PHP has been without usage, modification, and redistribution restrictions. In recent years, software meeting such open licensing qualifications has been referred to as *open-source* software. Open-source software and the Internet go together like bread and butter. Open-source projects like Sendmail, Bind, Linux, and Apache all play enormous roles in the ongoing operations of the Internet at large. Although the fact that open-source software is freely available for use has been the characteristic most promoted by the media, several other characteristics are equally important if not more so:

- **Free of licensing restrictions imposed by most commercial products:** Open-source software users are freed of the vast majority of licensing restrictions one would expect of commercial counterparts. Although some discrepancies do exist among license variants, users are largely free to modify, redistribute, and integrate the software into other products.
- **Open development and auditing process:** Although there have been some incidents, open-source software has long enjoyed a stellar security record. Such high standards are a result of the open development and auditing process. Because the source code is freely available for anyone to examine, security holes and potential problems are rapidly found and fixed. This advantage was perhaps best summarized by open-source advocate Eric S. Raymond, who wrote, "Given enough eyeballs, all bugs are shallow."
- **Participation is encouraged:** Development teams are not limited to a particular organization. Anyone who has the interest and the ability is free to join the project. The absence of member restrictions greatly enhances the talent pool for a given project, ultimately contributing to a higher-quality product.

# Summary

This chapter has provided a bit of foreshadowing about this wonderful language to which much of this book is devoted. We looked first at PHP's history, before outlining version 4 and 5's core features, setting the stage for later chapters.

In Chapter 2, prepare to get your hands dirty, as you'll delve into the PHP installation and configuration process. Although readers often liken most such chapters to scratching nails on a chalkboard, you can gain much from learning more about this process. Much like a professional cyclist or race car driver, the programmer with hands-on knowledge of the tweaking and maintenance process often holds an advantage over those without, by virtue of a better understanding of both the software's behaviors and quirks. So grab a snack and cozy up to your keyboard; it's time to build.