

# PHP by Example

A Practical Guide to Creating  
Web Applications with PHP

---

Alex Vasilev

Apress®

# PHP by Example

A Practical Guide to Creating  
Web Applications with PHP

Alex Vasilev

Apress®

# ***PHP by Example: A Practical Guide to Creating Web Applications with PHP***

Alex Vasilev

Department of Software Systems and Technologies,  
Taras Shevchenko National University of Kyiv, Kyiv, Ukraine

ISBN-13 (pbk): 979-8-8688-0257-7

ISBN-13 (electronic): 979-8-8688-0258-4

<https://doi.org/10.1007/979-8-8688-0258-4>

Copyright © 2024 by Alex Vasilev

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

Trademarked names, logos, and images may appear in this book. Rather than use a trademark symbol with every occurrence of a trademarked name, logo, or image we use the names, logos, and images only in an editorial fashion and to the benefit of the trademark owner, with no intention of infringement of the trademark.

The use in this publication of trade names, trademarks, service marks, and similar terms, even if they are not identified as such, is not to be taken as an expression of opinion as to whether or not they are subject to proprietary rights.

While the advice and information in this book are believed to be true and accurate at the date of publication, neither the authors nor the editors nor the publisher can accept any legal responsibility for any errors or omissions that may be made. The publisher makes no warranty, express or implied, with respect to the material contained herein.

Managing Director, Apress Media LLC: Welmoed Spahr

Acquisitions Editor: James Robinson-Prior

Development Editor: James Markham

Editorial Assistant: Gryffin Winkler

Copyeditor: Kim Burton

Cover designed by eStudioCalamar

Cover image designed by Barbara A Lane from Pixabay

Distributed to the book trade worldwide by Springer Science+Business Media New York, 1 New York Plaza, Suite 4600, New York, NY 10004-1562, USA. Phone 1-800-SPRINGER, fax (201) 348-4505, e-mail [orders-ny@springer-sbm.com](mailto:orders-ny@springer-sbm.com), or visit [www.springeronline.com](http://www.springeronline.com). Apress Media, LLC is a California LLC and the sole member (owner) is Springer Science + Business Media Finance Inc (SSBM Finance Inc). SSBM Finance Inc is a **Delaware** corporation.

For information on translations, please e-mail [booktranslations@springernature.com](mailto:booktranslations@springernature.com); for reprint, paperback, or audio rights, please e-mail [bookpermissions@springernature.com](mailto:bookpermissions@springernature.com).

Apress titles may be purchased in bulk for academic, corporate, or promotional use. eBook versions and licenses are also available for most titles. For more information, reference our Print and eBook Bulk Sales web page at <http://www.apress.com/bulk-sales>.

Any source code or other supplementary material referenced by the author in this book is available to readers on GitHub. For more detailed information, please visit <https://www.apress.com/gp/services/source-code>.

“All Rights are reserved by the Publisher except for Russian, Ukrainian and Bulgarian rights”.

Translated into the English from the Ukrainian.

Previously published in Russian as “Программирование на PHP в примерах и задачах” (2021) by Eksmo Publishing

Previously Published in Ukrainian as “Програмування мовою PHP” (2022) by LIRA-K Publishing

If disposing of this product, please recycle the paper

*In memory of my dad. Thank you for all you gave, and sorry  
for all I didn't return.*

# Table of Contents

- About the Author .....xi**
- About the Technical Reviewer .....xiii**
- Acknowledgments .....xv**
- Introduction .....xvii**
  
- Chapter 1: The First Program ..... 1**
  - The Program’s Code..... 1
  - The Interpreter Regime .....3
  - The Server Regime ..... 18
  - Summary.....25
  
- Chapter 2: Variables and Data Types .....27**
  - Introduction to Variables .....27
  - Basic Data Types .....31
  - The Arithmetic Operations .....34
  - The Comparison Operations.....37
  - The Logical Operations .....39
  - The Bitwise Operations .....41
  - The Operations with Strings.....46
  - Assigning Values .....48
  - The Ternary Operator .....49
  - Summary.....52

TABLE OF CONTENTS

**Chapter 3: The Control Statements.....53**

- The if Conditional Statement.....53
- The while Loop Statement .....62
- The do-while Loop Statement.....66
- The for Loop Statement .....69
- The switch Selection Statement .....74
- The goto Instruction.....82
- The match Selection Statement.....84
- Summary.....89

**Chapter 4: Arrays.....91**

- Getting Familiar with Arrays.....92
- A Loop over an Array .....99
- Multidimensional Arrays .....107
- Array Assignments .....111
- Concatenating Arrays.....115
- Comparing Arrays .....117
- Functions for Handling Arrays.....119
- Summary.....125

**Chapter 5: Functions .....127**

- Creating Functions .....127
- The Function Result .....132
- The Type of Arguments and Result.....137
- The Argument Passing Mechanism.....141
- The Argument Value by Default.....145
- An Arbitrary Number of Arguments.....147
- Recursion .....156

The eval() Function.....	159
Anonymous Functions.....	162
Named Arguments .....	164
Summary.....	169
<b>Chapter 6: Useful Tricks and Operations .....</b>	<b>171</b>
References.....	171
Constants.....	177
Global Variables .....	179
Static Variables .....	182
Multiline Strings.....	184
Using Files.....	187
Including a File in the Program.....	194
Summary.....	196
<b>Chapter 7: Classes and Objects .....</b>	<b>199</b>
The OOP Principles.....	199
Creating Classes and Objects .....	202
The Methods .....	207
The Constructor and the Destructor.....	211
Static Fields and Methods.....	215
Copying Objects .....	217
Private Fields and Methods.....	220
Special Methods .....	223
Defining Fields in the Constructor.....	235
Summary.....	239

TABLE OF CONTENTS

**Chapter 8: Inheritance .....243**

- Creating a Child Class ..... 243
- Overriding Methods..... 248
- Constructors and Inheritance..... 253
- Inheritance and Private Members ..... 257
- Protected Members of a Class ..... 259
- Virtual Methods..... 261
- A Function as a Field Value ..... 263
- Multilevel Inheritance ..... 266
- Summary..... 271

**Chapter 9: Advanced OOP Mechanisms.....273**

- Abstract Classes ..... 273
- Interfaces..... 278
- Interface Inheritance..... 281
- Class Inheritance and Interface Implementation ..... 283
- Traits ..... 285
- Object Type Control ..... 288
- A Namespace ..... 292
- Anonymous Classes..... 298
- Summary..... 300

**Chapter 10: Error Handling.....303**

- Exception Handling Principles..... 303
- Exception Classes ..... 307
- Throwing Exceptions..... 310
- The Custom Exceptions..... 314
- Handling Exceptions of Different Classes ..... 318



Nested try-catch Constructions and the finally Block.....	324
Rethrowing an Exception .....	328
The Functions for Handling Errors .....	330
Summary.....	333
<b>Chapter 11: Generators and Iterators .....</b>	<b>335</b>
Getting Familiar with Generators .....	335
A Generator Function with Arguments .....	339
An Array Based on a Generator .....	341
The Generator Result .....	343
Using Generators.....	346
Passing a Value to a Generator .....	351
Iterators.....	353
Summary.....	359
<b>Chapter 12: Using PHP.....</b>	<b>361</b>
A Script in an HTML Document .....	361
Handling the Request Parameters .....	369
Using Buttons.....	373
Using Several Buttons.....	381
Using Lists and Checkboxes .....	386
A Slider and Radio Buttons .....	394
Summary.....	402
<b>Chapter 13: Afterword: What Was and What Will Be .....</b>	<b>405</b>
<b>Index.....</b>	<b>407</b>

# About the Author



**Alex Vasilev** is a software systems and technologies professor at the Faculty of Information Technology, Taras Shevchenko National University of Kyiv in Ukraine. He has taught programming (C++, C#, Java, JavaScript, Python, and PHP) for 20 years. To date, he has written over 30 programming books in his native Ukraine. This is his first book directly published in English.

# About the Technical Reviewer



**Vadim Atamanenko** is an experienced software engineer and technical reviewer with over 25 years of expertise in software development. His professional journey includes a leadership role in the analytical reporting department of Freedom Holding Corp. Throughout his career, he has actively contributed to the scientific community, publishing articles on the application of artificial intelligence in the financial sector.

Vadim's technical expertise has been internationally recognized, evidenced by membership in two prestigious associations: IEEE (Institute of Electrical and Electronics Engineers) and Leaders Excellence at Harvard Square. His scientific papers have been published in various scholarly journals, including *Modern Science: Current Issues in Theory and Practice*.

# Acknowledgments

To my dear children Anastasia and Bohdan, and all my family. Thank you for your support and love. You give meaning to my life and encourage me to move on.

I sincerely thank the great and professional Apress team for the outstanding support and dedication provided throughout the process of bringing the book to fruition. Their collective efforts have undoubtedly played a pivotal role in making the book better.

# Introduction

*There are many things in this universe you are not meant to understand. Now, that does not mean they are not real.*

—*ALF* (TV series)

This book discusses the PHP programming language. PHP is a simple, beautiful, and elegant language. Moreover, it is special in some sense. PHP is designed to execute code on the server side. In other words, it is not enough to know PHP. It is also necessary to understand how and for what it is used. That is important since understanding the possibilities of a language is the key to using it effectively.

---

**Note** PHP is a language designed for programming and web programming.

---

## About PHP

The PHP language is used for creating sites and web applications. It has a long story, is popular among developers, and is supported by most host servers.

## Details

---

Hosting is a service for providing resources and space on a server to host data and information (for example, web pages). The host server is a server that hosts the information. For the sake of simplicity, this means the host server is a computer (server) that hosts the user's web page (that is, the page where PHP is supposed to be used).

---


The author of PHP is Rasmus Lerdorf. The project started as writing scripts to support a personal web page and was initially titled Personal Homepages Tools or PHP Tools. Then, it was transformed into an independent and influential software product. Today, the name PHP is usually associated with the phrase *Hypertext Preprocessor*, which is not far from the truth.

---

**Note** When writing this book, the current version is PHP 8. On the other hand, in practice, the latest version of the language does not immediately start to be used. There is some inertia here due to both objective and subjective factors. Therefore, universal approaches relevant to the last few language versions are considered. Notably, there is no sixth version: the seventh version follows after the fifth. The reason is that the attempt to release the sixth version was highly unsuccessful.

---

PHP is a scripted and interpreted language. Interpretability means that a program is executed under the control of a particular program called an *interpreter*. The interpreter reads the code line by line and executes the corresponding instructions.

 **The PHP 8 Standard** PHP 8 introduced a JIT compiler (short for *Just in Time*) for compiling PHP code to speed up program execution. When compiled, the program instructions are translated into processor-level instructions.

---

Scripting languages are usually high-level ones. Unlike conventional programs, scripts usually contain instructions for controlling ready-made software components. In other words, a scripting language is a straightforward language. Although, of course, not everything is always so rosy.

---

## Details

---

The C language has influenced the syntax of the PHP language. Therefore, if you know languages such as C, C++, C#, or Java, you will find many familiar syntax constructions.

---

Of course, PHP is a programming language, among many others. But it has an essential feature in how PHP codes are used.

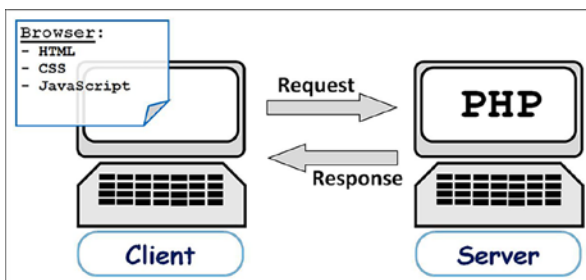
If you are dealing with any conventional programming language, the process of writing and using programs looks like follows. First, you create the program code—in other words, you write a program. Then, that program must be executed. How to do that depends on the language, but the most crucial question is whether the program is *compiled* or *interpreted*. If the program is compiled, a particular compiler program translates your program into machine instructions (or something similar to them), and then those instructions are executed. If the program is interpreted, a special interpreter program reads your program and executes statements from the code. But whatever happens, the important thing here is that you do all the operations on the same computer. You run the program on the computer and get a result. You can do anything you want with the result of the program execution, but the critical point is that it is enough to have only one computer.

With the PHP language, things are somewhat different. To understand the problem, let's consider what happens when you access a web page and how PHP is involved in that case.

## A Client and a Server

In general terms, here is the scheme according to which the site is viewed on the network. In this case, the main acting “characters” are the computer on which you want to view the web page and the computer on which that page is located. The first computer (on which you are trying to view the web page) is called a *client*, and the computer on which the web page is located is called a *server*.

There is a connection between these computers through the global network; therefore, the computers can send information to each other. You work on a client computer and want to view a web page. To do that, you run a special program designed to view web pages. The program is called a *browser*. You open a browser (for example, Chrome, Opera, or Edge), and in the address bar, you enter the site address you want to view. The browser initiates a request to the server. The server receives the request, processes it, and returns a response to the client. The response contains the document's code; the browser should display it on the client screen. The general scheme of the “interaction” is illustrated in Figure I-1.



**Figure I-1.** The scheme of the interaction between a client and a server



The browser processes the document in HTML format (the abbreviation comes from Hypertext Markup Language). The text contains a special markup. The browser “understands” that markup and renders the document following the instructions embedded in the document. In addition to the actual HTML markup, the document displayed by the browser can contain additional instructions. For example, CSS (Cascading Style Sheets) formatting can be used in the document. The document may also contain JavaScript scripts. In the latter case, the browser executes these scripts. So, the server sends a set of commands, and the browser executes them. The important thing is that the commands are executed on the same computer that made the request.

---

**Note** The server tells what to do, and the client’s browser performs the necessary operations. Convenient but not always safe.

---

So, where is the place for PHP in this scheme? The answer is at the stage of processing the request by the server. When the server receives a request from a client, it processes the request, and while processing it, scripts can be executed—in this case, PHP scripts.

---

## Details

---

Often, the script’s output is a generated HTML code passed to the client.

---

But that is not all. Many programs try to exchange information over the network. Processes on a client send signals to processes on a server and back, and you need to know which signal is for which process. For that purpose, you use ports. Ports are unique integer identifiers the processes use to identify the signals sent to them. Therefore, requests from the client’s browser and server responses must be synchronized by ports. That is, to work effectively with PHP, you need to solve quite a few technical problems. All that is considered step by step, as necessary.

## How to Execute the PHP Code

Let's take the next step in studying PHP. Namely, focusing on how to execute a program written in PHP. If you are talking about the “natural” way of using PHP code, you would need a server and a client. That is two computers. The script (the program) is hosted on the server, and you can view the result of the program execution by accessing the web page on the server through the client browser. But even if all these resources are available, the described strategy is not very convenient since you need to edit the program on one computer (the server) and check the result on another computer (the client). So, it is clear that you would like to have a more reliable strategy.

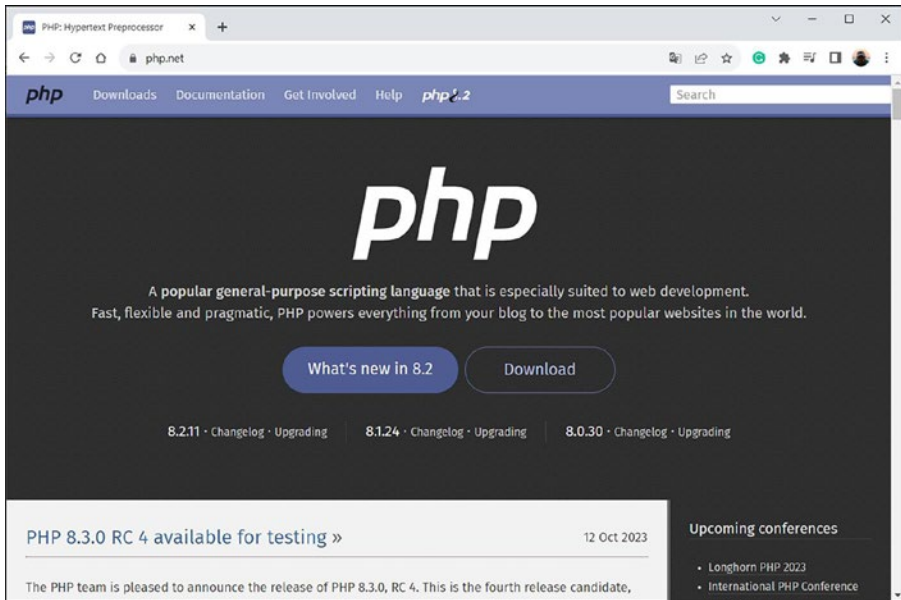
An alternative way is to “trick” the browser. Namely, you can create an illusion that the server is a client itself. It's about using a *local server*. It is easy to switch to that mode. The advantage of the approach is that the program and its result are localized within the same computer.

On the other hand, PHP code can be executed using an interpreter—that is, approximately the same as in the case of other interpreted languages. That is probably the easiest way to see what the result of running a program is. Nevertheless, you should not forget that PHP programs are not written to be executed by the interpreter on the client's computer. So, there will be some tricks, too.

## The Software

The book contains many examples, and in the process of studying them, it is desirable to disassemble the program and examine the result of its execution. That requires special software.

First of all, you have to install the software that supports PHP. To do that, go to [www.php.net](http://www.php.net), as shown in Figure I-2.



**Figure I-2.** The window for PHP support at [www.php.net](http://www.php.net)

You should find the software download section in that window and download the necessary files.

---

**Note** In the simplest case, the installation comes down to unpacking the archive downloaded from [www.php.net](http://www.php.net). That will likely be enough for using the PHP interpreter (`php.exe` file) in command-line mode. You may need to perform additional settings for a more “comfortable” work regime involving special software. If so, refer to the help information on the [www.php.net](http://www.php.net) page and use the help for the relevant software product (for example, a code editor).

---

## Details

---

**You can use the `php -v` command-line instructions to check the PHP version. To get PHP help, use the `php -h` command. Additional information about PHP can be obtained with the `php -i` command.**

**If you use the Windows operating system, you can enter the `cmd` instruction into the address bar of Windows Explorer to switch to the terminal mode. Then, you must change to the PHP directory in the terminal window. For example, if PHP is in the `C:\PHP` folder, the appropriate command would be `cd C:\PHP`. An alternative is to navigate to the PHP directory first and then enter the `cmd` instruction in the Explorer address bar.**

**Many operating systems of the Linux family have PHP pre-installed. But if this is not the case, you can use the `sudo apt install php` command to install PHP.**

---

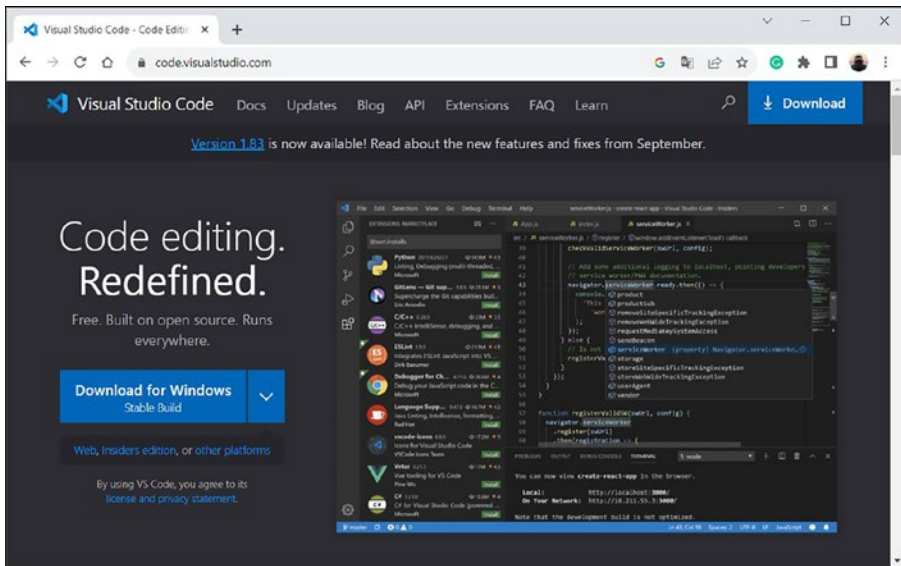
In general, to have a PHP interpreter that allows you to execute programs written in PHP is enough. However, you also need to type and edit your programs somewhere. In principle, a regular text editor could be the choice. But it's much better to install something more advanced, with support for PHP syntax (an editor that “understands” PHP special instructions).

---

**Note** It is easy to find a suitable editor for processing PHP code if necessary. However, such a strategy is mainly aimed at advanced users. That is why it is beyond your attention.

---

There are other options as well. For example, the Visual Studio Code development environment (the address is <https://code.visualstudio.com>) is quite convenient. The browser opened on the project page is shown in Figure I-3.



*Figure I-3. The Visual Studio Code project support page*

---

## Details

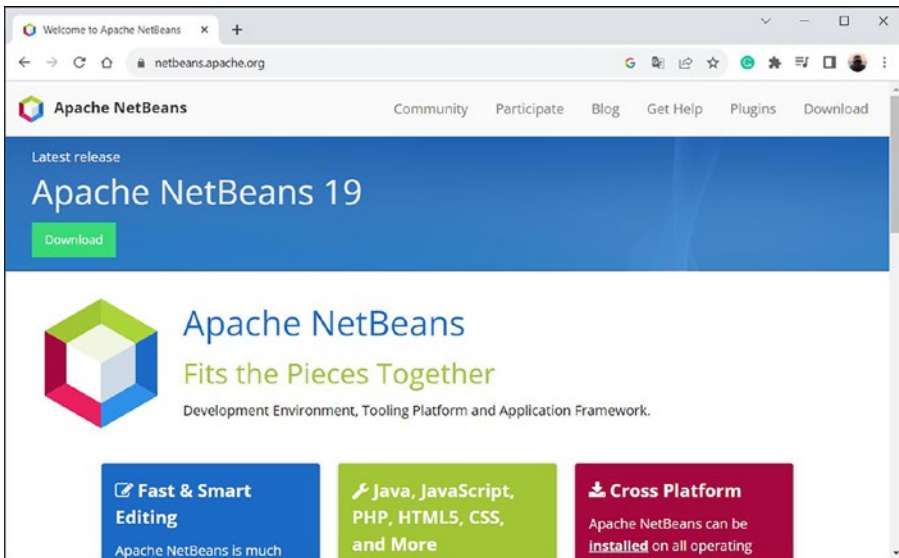
---

The first time you run the Visual Studio Code application, you must confirm the installation of the PHP support in the Customize section.

---

Another good option for developing PHP programs is the NetBeans IDE. The installation files can be downloaded from <http://netbeans.apache.org>. Figure I-4 shows a browser open on the NetBeans project support page.

## INTRODUCTION



*Figure I-4. The NetBeans project support page*

---

### Details

---

You may need the Java Development Kit (JDK) pre-installed on your computer to use NetBeans.

---

There are other helpful software products, including commercial ones. But once again, it is enough to install PHP and select a suitable code editor.

---

**Note** You will learn how to use the software (at the primary level) through examples.

---

## About This Book

This book is entirely devoted to the PHP language, focusing on why it is needed and its features. It investigates the main constructions of the language, as well as the approaches and mechanisms used in PHP. In the book, considerable attention is paid to examples. The range of topics discussed in the book is enough for writing effective PHP codes. The main part of the book consists of twelve chapters.

- Chapter 1 contains simple programs, and it discusses how to use the software and analyzes the general principles of creating PHP programs there.
- Chapter 2 discusses variables and data types. There, you will learn how to create variables, perform operations with them, and use data of different types.
- Chapter 3 is devoted to the control statements. You will get a notion of the conditional statement, selection statement, and loop statements, as well as the goto instruction. All those are critical for creating efficient PHP programs.
- Chapter 4 contains information about arrays. In PHP, arrays have several unique features and will be the study's subject. This chapter discusses one-dimensional and multi-dimensional arrays, describes the iteration over a collection statement, and provides an overview of the basic operations performed on arrays.

## INTRODUCTION

- Chapter 5 describes functions. You will learn how they are created, discuss the mechanisms for passing arguments, consider how a result is returned, learn how to set default values for arguments, and how to create functions with an arbitrary number of arguments. You will consider ways for passing arguments by name. The chapter also focuses on recursion, anonymous functions, and some other topics related to functions.
- Chapter 6 deals with links, constants, global and statistic variables, file handling, and methods for working with multi-line text.
- Chapter 7 discusses the principles of object-oriented programming (OOP) and considers how PHP implements these principles. You will learn how to create classes and objects and what to do with methods. You will also meet constructors, destructors, and static class members. The chapter deals with the problem of copying objects and illustrates how to use private fields and methods. Some special methods are also described in the chapter.
- Chapter 8 is devoted to such an important OOP mechanism as inheritance. You will learn how a child's class is created and how methods are overridden. You will consider the features of constructors, private and protected members of a class in the context of the inheritance. The chapter also discusses the virtuality of methods, describes multi-level inheritance and contains some other information.



- Chapter 9 describes abstract classes and interfaces (including their implementation and inheritance). You will learn what traits are, how an interface can control the type of an object, what a namespace is, and how it is used.
- The principles of error and exception handling are discussed in Chapter 10. You become familiar with the main exception classes and learn how exceptions are generated. In addition, the chapter describes methods for creating classes for user-defined exceptions. Some other issues related to error and exception handling are also covered.
- Chapter 11 is devoted to generators and iterators. You will learn to use and apply the generator functions in different situations. You will also learn what iterators are and how they are created.
- Chapter 12 contains examples of using PHP programs in practice. The chapter discusses several subjects that give an idea of the role PHP codes play in creating web documents.

At the end of each chapter, for convenience, a summary lists all the main points discussed in the chapter.

---

**Note** This book is primarily for those with minimal programming experience, so the content is presented as simply as possible.

---

## CHAPTER 1

# The First Program

*—I demand to restore the Earth's ozone layer.*

*—Alf, we won't make it by Saturday.*

*—ALF (TV series)*

In this chapter, you will create your first PHP program. Namely, you will examine some simple code and determine how it can be executed. There is not much programming in the chapter, but a lot of information is essential for using PHP.

## The Program's Code

A program in PHP is a set of instructions that an interpreter executes. These instructions must be appropriately formatted and passed to the interpreter for execution. The plans are focused on solving the following two tasks.

- Writing the code of a program
- Launching the program for execution

The second task is much more complex than the first one.

So, let's create your first PHP program. Namely, let's define what your program should do. Traditionally, the first program displays a message. You will do the same. Your program displays a welcome message in the output window (terminal). The program is shown in Listing 1-1.

**Listing 1-1.** The First Program

```
<?php
    print("Hello, PHP!");
?>
```

The program begins with the `<?php` statement and ends with the `?>` statement. That is a standard situation for all programs in PHP. The commands to be executed in the program are placed between these instructions. In this case, there is a single command. It calls the `print()` built-in function with "Hello, PHP!" passed as an argument.

---

**Details**

---

A function is a named block of code (command block) that can be executed by calling the function (by specifying its name with arguments, if necessary). Functions can be created in a program, or you can use built-in functions. Among the built-in functions, there is the `print()` function.

Text values are enclosed in double quotes. Also, text values can be enclosed in single quotes. Both styles are almost equivalent, but there are essential differences, which are discussed a little later.

It is also worth noting that every command in PHP ends with a semicolon.

---



**Note** Frankly speaking, `print()` is not exactly a function but rather a special syntax construction of the PHP language. But since its properties are similar to a function, consider it a function. And you proceed in the same way in all such cases.

---

As you might guess, the command displays a message in the terminal. The message's text is passed to the `print()` function as an argument. The following shows the result of running the program.

---

**The output of the program (from Listing 1-1)**

```
Hello, PHP!
```

---

There is nothing complicated in this program. It remains only to figure out how to run the program for execution.

---

**Details**

---

The argument can be specified without parentheses when calling the `print()` function. That means that instead of the `print("Hello, PHP!")` command, you can use the `print "Hello, PHP!"` statement. Another alternative to the `print()` function is the `echo` statement. Namely, the `echo "Hello, PHP!"` command could display the message.

---

## The Interpreter Regime

Let's explore several options, and you can choose the one you like the best.



**Note** There are two regimes of the PHP program execution.

The first one is the execution of a program using an interpreter on a client computer. The second regime means encapsulating PHP code in a document with HTML markup. In that case, you must use an external or local server. Implementing this is a little more complicated than executing the code with an interpreter, but it is more “natural” and related to practical PHP usage.

---

So, the first thing to do is to create a file with the program, as in Listing 1-1, and save that file with the `.php` extension. For example, name the file `hello.php`, located in the `D:\Books\php\codes` folder. Next, execute the file. At this stage, you need an interpreter, the `php.exe` file located in the folder where PHP was installed. Let's assume that PHP is installed in the `C:\PHP` folder. The recipe for launching the program is simple: in the terminal's command line, you must execute `php.exe`, passing the `hello.php` file to it (the file with the program) as a parameter. You can make that simple. You need to specify the full path to the `php.exe` file in the command line and, separated by a space, the full path to the file with the `hello.php` program. The corresponding command would look as follows.

```
C:\PHP\php.exe D:\Books\php\codes\hello.php
```

Nevertheless, specifying the full path to the files is not very convenient. In principle, you can move to the directory with the `php.exe` file installed and save the files with the PHP code in the same place. So, if PHP is installed in the `C:\PHP` folder, then to move to it, run the following command in the command line.

```
cd C:\PHP
```

---

## Details

---

If you need to change a directory and a disk, use the `/d` option. For example, if you need to move from the `D:\` drive to the `C:\` drive, the command looks like `cd /d C:\`.

---

If the `hello.php` file is located in the same folder as the `php.exe` file, then you use the following command to run the program for execution.

```
php hello.php
```

The same command can be used under the Windows operating system if you add the path to the PHP folder to the Path environment variable and then move to the folder with the program file.

---

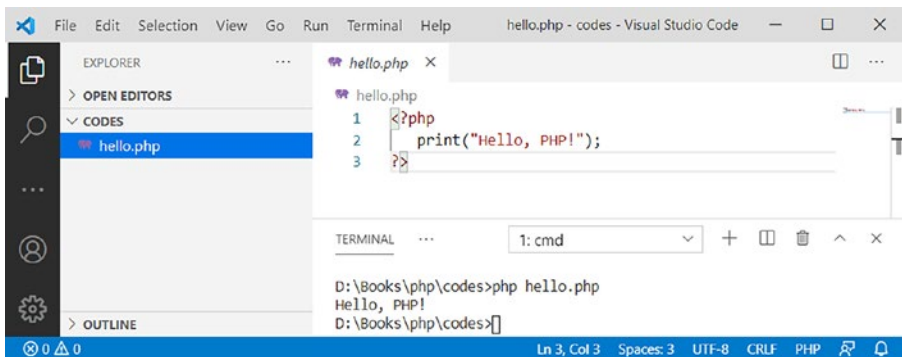
## Details

---

Select the **Advanced** system settings item in the computer properties to get the Path environment variable. In the **System Properties** window, select the **Environment Variables** icon. In the **Environment Variables** window, select the **Path** position and change the contents of the corresponding variable.

---

The use of development environments simplifies testing and executing codes. Let's look at the Visual Studio Code environment as an example. Figure 1-1 shows the environment's window with the `hello.php` file open.



**Figure 1-1.** The Visual Studio Code development environment window with the `hello.php` program file open

In the environment window, open the folder with the program file. (You can use the **Open Folder** command from the **File** menu.) The files from this folder should appear in the **Explorer** section on the left side of the development environment window. In this case, select the file with the program. The file's contents are displayed in the central part of the window. You can run the program in the terminal, which is opened using