PHP Object-Oriented Solutions

David Powers

friendsof ED™
DESIGNER TO DESIGNER™
an Apress® company

PHP Object-Oriented Solutions

ISBN-13 (pbk): 978-1-4302-1011-5

ISBN-13 (electronic): 978-1-4302-1012-2

Printed and bound in the United States of America 9 8 7 6 5 4 3 2 1

Credits

Lead Editor
Ben Renow-Clarke

Technical Reviewer
Seungyeob Choi

Editorial Board
Clay Andres, Steve Anglin, Ewan Buckingham, Tony Campbell, Gary Cornell, Jonathan Gennick, Matthew Moodie, Joseph Ottinger, Jeffrey Pepper, Frank Pohlmann, Ben Renow-Clarke, Dominic Shakeshaft, Matt Wade, Tom Welsh

Project Manager
Beth Christmas

Copy Editors
Heather Lang and Damon Larson

Associate Production Director
Kari Brooks-Copony

Production Editor
Laura Esterman

Compositor
Molly Sharp

Proofreader
Patrick Vincent

Indexer
Toma Mulligan

Artist
April Milne

Interior and Cover Designer
Kurt Krames

Manufacturing Director
Tom Debolski

CONTENTS AT A GLANCE

CONTENTS

ABOUT THE AUTHOR

David Powers is the author of a series of highly successful books on PHP, including *PHP Solutions: Dynamic Web Design Made Easy* (friends of ED, ISBN: 978-1-59059-731-6) and *The Essential Guide to Dreamweaver CS3 with CSS, Ajax, and PHP* (friends of ED, ISBN: 978-1-59059-859-7). As a professional writer, he has been involved in electronic media for more than 30 years, first with BBC radio and television, both in front of the microphone (he was a BBC correspondent in Tokyo from 1987 to 1992) and in senior editorial positions. His clear writing style is valued not only in the English-speaking world—several of his books have been translated into Spanish and Polish.

Since leaving the BBC to work independently, David has devoted most of his time to web development, writing books, and teaching. He is active in several online forums, giving advice and troubleshooting PHP problems. David's expertise was recognized by his designation as an Adobe Community Expert in 2006.

When not pounding the keyboard writing books or dreaming of new ways of using PHP and other programming languages, David enjoys nothing better than visiting his favorite sushi restaurant. He has also translated several plays from Japanese.

ABOUT THE TECHNICAL REVIEWER

Seungyeob Choi is the lead developer and technology manager at Abraham Lincoln University in Los Angeles, where he has been developing various systems for online education. He built the university's learning platform and has been working on a development project for Student Lifecycle Management. Seungyeob has a PhD in computer science from the University of Birmingham, England.

ACKNOWLEDGMENTS

The book you're holding in your hand (or reading on the screen) owes its genesis to a tongue-in-cheek exchange with Steve Fleischer of Flying Tiger Web Design (`www.flyingtigerwebdesign.com`), who suggested I should write *Powers Object-Oriented PHP*. Actually, he phrased it rather differently. If you take the initial letters of the suggested title, you'll get the drift . . . But Steve had an important point: he felt that books on object-oriented programming (OOP) frequently assumed too much prior knowledge or weren't easily adaptable to PHP in a practical way. If you like what you find in this book, thank Steve for planting the idea in my brain. If you don't like it, blame me, because I'm the one responsible for writing it the way it is.

Thanks must also go to everyone at Apress/friends of ED for helping bring "my baby" into the world. Books are uncannily like real babies. This one took exactly nine months from conception to birth with the expert help of editor Ben Renow-Clarke, project manager Beth Christmas, and many other "midwives." I owe a particular debt of gratitude to Seungyeob Choi for his perceptive technical review. Seungyeob's eagle eye and deep knowledge of PHP and OOP saved me from several embarrassing mistakes. Any remaining errors are my responsibility alone.

I would also like to thank everyone who has supported me by buying this or any of my previous books. I realize not everyone can afford to buy books, but the royalties from new—not second-hand—books ensure that authors get some reward for all the hard effort that goes into writing. Even the most successful computer books can never aspire to the stratospheric heights of Harry Potter, so every little bit helps—and is much appreciated.

The biggest thanks of all must undoubtedly go to the developers of PHP, who have given the rest of the world a superb programming language that continues to go from strength to strength.

INTRODUCTION

My first experiments with object-oriented programming in PHP took place about six years ago. Unfortunately, the book that introduced me to the subject concentrated on the mechanics of writing classes and paid little heed to principles underlying OOP. As a result, I wrote classes that were closely intertwined with a specific project ("tightly coupled," to use the OOP terminology). Everything worked exactly the way I wanted, but the design had a fundamental flaw: the classes couldn't be used for any other project. Worse still, it was a large project—a bilingual, searchable database with more than 15,000 records—so any changes I wanted to make to it involved revising the whole code base.

The purpose of this book is to help you avoid the same mistake. Although most chapters revolve around mini-projects, the classes they use are project-neutral. Rather than being a "how to" cookbook, the aim is to help developers with a solid knowledge of PHP basics add OOP to their skill set.

So, what is OOP? To oversimplify, OOP groups together functions (known in OOP-speak as "methods") in classes. In effect, a class can be regarded as a function library. What makes OOP more powerful is the fact that classes can be extended to add new functionality. Since many of the new features added to PHP 5 are object-oriented, this means you can easily extend core PHP classes to add new functionality or simply make them work the way you want them to. In fact, Chapter 3 does precisely that: it extends the PHP `DateTime` class to make it easier to use. The project in Chapter 4 takes the PHP filter functions and hides them behind a much more user-friendly interface.

Chapter 5 shows how to create a class that retrieves a text file from a remote server by automatically detecting the most efficient available method. Chapters 6 and 7 cover two of the most important OOP features added to core PHP in version 5: SimpleXML and the Standard PHP Library (SPL). The XML theme continues in the final two chapters, which use the PHP `XMLWriter` class to generate XML on the fly from a database and show you how to create a news feed from your site.

The need for OOP has come about because PHP is being used increasingly for large-scale web applications. Object-oriented practices break down complex operations into simple units, each responsible for a defined task. This makes code much easier to test and maintain. However, ease of maintenance is just as important in small-scale projects, so OOP can play a role in projects of any size. This is an introductory book, so the object-oriented solutions it contains are designed for use in small projects, but the principles they demonstrate apply equally to large-scale projects.

By the time you have finished this book, you should understand what OOP is and how to write PHP classes that conform to current best practices, making your code easier to maintain and deploy across multiple projects. The information contained in this book will also provide a solid foundation for anyone planning to use an object-oriented framework, such as the Zend Framework (`www.zend.com/en/community/framework`).

Although everything in this book is devoted to OOP, it's important to emphasize that OOP is only *part* of PHP. OOP helps you create portable, reusable code. Use it where appropriate, but there's no need to throw out all of your existing PHP skills or code.

Another important thing to emphasize is that all the code in this book requires a minimum of PHP 5, and preferably PHP 5.2 or 5.3. It has also been designed to work in PHP 6. *The code will not work in PHP 4*, nor will any support be provided for converting it to PHP 4. Even though at the time of publication, it's estimated that more than half of all PHP-driven websites still run on PHP 4, all support for PHP 4 officially ended on August 8, 2008. PHP 4 is dead. Long live PHP 5 (and PHP 6 when it's released). If you haven't yet made the switch from PHP 4, now is the time to do it.

Who should read this book

If you develop in PHP, but haven't yet got your feet wet with OOP, this is the book for you. No previous knowledge of OOP is necessary: Chapter 1 covers the basic theory and explains how OOP fits into PHP; Chapter 2 then goes into the mechanics of writing object-oriented code in PHP. The remaining seven chapters put all the theory into practice, showing you how to create and use your own classes and objects, as well as covering object-oriented features that have been built into core PHP since version 5.

You don't need to be a PHP expert to follow this book, but you do need to know the basics of writing your own PHP scripts. So, if you're comfortable with concepts such as variables, loops, and arrays, and have ever created a function, you should be fine. Throughout the book, I make extensive use of core PHP functions. In some cases, such as with the filter functions in Chapter 4, I go into considerable detail about how they work, because that knowledge is essential to understanding the chapter. Most of the time, though, I explain what the function is for and why I'm using it. If you want a more in-depth explanation, I expect you to look it up for yourself in the PHP online documentation at `http://docs.php.net/manual/en/`.

The book aims to be a gentle introduction to OOP in PHP, but it moves at a fairly fast pace. The code involved isn't particularly difficult, but it might take a little more time for some of the concepts to sink in. The best way to achieve this is to roll up your sleeves and start coding. Exercises at strategic points demonstrate what a particular section of code does and help reinforce understanding.

Using the download code

All the files necessary to work with this book can be downloaded from the friends of ED website by going to `www.friendsofed.com/downloads.html` and scrolling down to the link for *PHP Object-Oriented Solutions*. Download the ZIP file, and unzip its contents into a new folder inside your web server document root. I named the folder `OopSolutions`, but you can call it whatever you want. In addition to a series of folders named `ch2_exercises` through `ch9_exercises`, the folder should contain the following:

- `Ch2`: This contains example class definitions for use with `ch2_exercises`.
- `class_docs`: This contains full documentation in HTML format for all the classes developed in the book. Double-click `index.html` to view them in your browser.
- `finished_classes`: This contains a full set of completed class definitions.
- `Pos`: *This folder is empty*. It is where you should create your own versions of the class definitions as you work through each chapter. If you don't want to type out everything yourself, you need to copy each class definition from `finished_classes` to this folder for the files in the exercise folders for each chapter to work.

Understanding the file numbering system

Most download files have a filename ending in an underscore and a number before the `.php` filename extension (e.g., `Book_01.php`, `Book_02.php`). This is because the files represent a class definition or exercise at a particular stage of development.

If you are typing out the exercises and class definitions yourself, leave out the underscore and number (e.g., use `Book.php` instead of `Book_01.php`). Throughout the text, I indicate the number of the current version so you can compare the appropriate supplied version with your own, or simply use it directly if you don't want to type everything yourself.

To get the best out of this book, I strongly urge you to type out all the exercises and class definitions yourself. It's a lot of work, but hands-on practice really does reinforce the learning process.

What to do if things go wrong

Every effort has been made to ensure accuracy, but mistakes do slip through. If something doesn't work the way you expect, your first port of call should be `www.friendsofed.com/book.html?isbn=9781430210115`. A link to any known corrections since publication will be posted there. If you think you have found a mistake that's not listed, please submit an error report to `www.friendsofed.com/errataSubmission.html`. When friends of ED has finished with the thumbscrews and forced me to admit I'm wrong, we'll post the details for everyone's benefit on the friends of ED site.

If the answer isn't on the corrections page, scan the chapter subheadings in the table of contents, and try looking up a few related expressions in the index. Also try a quick search through Google or one of the other large search engines. My apologies if all this sounds obvious, but an amazing number of people spend more time waiting for an answer in an online forum than it would take to go through these simple steps.

If you're still stuck, visit `www.friendsofed.com/forums/`. Use the following guidelines to help others help you:

- Always check the book's corrections page first. The answer may already be there.
- Search the forum to see if your question has already been answered.
- Give your message a meaningful subject line. It's likely to get a swifter response and may help others with a similar problem.
- Give the name of the book and a page reference to the point that's giving you difficulty.
- "It doesn't work" gives no clue as to the cause. "When I do so and so, x happens" is a lot more informative.
- If you get an error message, say what it contains.
- Be brief and to the point. Don't ask half a dozen questions at once.
- It's often helpful to know your operating system and which version of PHP you're using.
- Don't post the same question simultaneously in several forums. If you find the answer elsewhere, have the courtesy to close the forum thread and post a link to the answer.

Please be realistic in your expectations when asking for help in a free online forum. I'm delighted if you have bought one of my books and will try to help you if you run into problems; but I'm not always available and can't offer unlimited help. If you post hundreds of lines of code, and expect someone else to scour it for mistakes, don't be surprised if you get a rather curt answer or none at all. And if you do get the help that you need, keep the community spirit alive by answering questions that you know the answer to.

Layout conventions

To keep this book as clear and easy to follow as possible, the following text conventions are used throughout.

Important words or concepts are normally highlighted on the first appearance in **bold type**.

Code is presented in `fixed-width font`.

New or changed code is normally presented in **`bold fixed-width font`**.

Pseudocode and variable input are written in *`italic fixed-width font`*.

Menu commands are written in the form Menu ➤ Submenu ➤ Submenu.

Where I want to draw your attention to something, I've highlighted it like this:

Ahem, don't say I didn't warn you.

Sometimes code won't fit on a single line in a book. Where this happens, I use an arrow like this: ➥.

```
This is a very, very long section of code that should be written all ➥
on the same line without a break.
```