

# Unlock PHP 8: From Basic to Advanced

---

*The next-level PHP 8 guide for  
dynamic web development*

---

**Roni Sommerfeld**



[www.bpbonline.com](http://www.bpbonline.com)

First Edition 2024

Copyright © BPB Publications, India

ISBN: 978-93-55519-757

*All Rights Reserved.* No part of this publication may be reproduced, distributed or transmitted in any form or by any means or stored in a database or retrieval system, without the prior written permission of the publisher with the exception to the program listings which may be entered, stored and executed in a computer system, but they can not be reproduced by the means of publication, photocopy, recording, or by any electronic and mechanical means.

## LIMITS OF LIABILITY AND DISCLAIMER OF WARRANTY

The information contained in this book is true to correct and the best of author's and publisher's knowledge. The author has made every effort to ensure the accuracy of these publications, but publisher cannot be held responsible for any loss or damage arising from any information in this book.

All trademarks referred to in the book are acknowledged as properties of their respective owners but BPB Publications cannot guarantee the accuracy of this information.

To View Complete  
BPB Publications Catalogue  
Scan the QR Code:



[www.bpbonline.com](http://www.bpbonline.com)

Kup ksi k

## **Dedicated to**

*My wife Isabel and my daughter, Anne,  
Who are the true architects of my heart, who taught me that love is the  
perfect algorithm: resilient, adaptable, and infinitely surprising.*

## About the Author

**Roni Sommerfeld**, Brazilian and German, is a technology enthusiast with extensive experience in system development since 2010. He has worked on highly complex projects, handling over 50 million records daily with PHP. As the founder of the PHPiando blog and YouTube channel, he shares tips and insights on programming. Living in Portugal with his wife Isabel, daughter Anne, and their dog Odin, he swaps code for cooking, blending culinary recipes with programming ones in a delightful harmony.

## About the Reviewer

**Marco Dal Monte** has a Master's Degree in Computer Science from the University of Padova, Italy, where he specialized his studies in Artificial Intelligence. After university, he worked for a brief period in Italy before moving to London.

He works mainly in PHP, using Symfony and Laravel as frameworks, and he is interested in software engineering methodologies, DevOps and system administration in Linux. Currently, he is working at RemitONE, a company based in London that is a leader in white label software for money transfers.

# Acknowledgement

The journey to completing this book has been marked by support and inspiration from many valuable sources.

I am immensely grateful to my wife, Isabel, and our daughter, Anne, whose love and support were fundamental. You, along with the rest of our family, are the true support and joy in my life. Thank you for being by my side every step of this process.

Special thanks to the BPB team, whose expertise and dedication turned a vision into reality. The collaboration of reviewers, technical experts, and editors was essential in improving each chapter of this book.

I extend my gratitude to my professional colleagues and friends who shared their experiences and knowledge with me. You have enriched my professional and personal perspective, contributing to the content and depth of this work.

And last but not least, I thank you, the reader, for embarking on this journey with me. Their quest for knowledge and growth is what motivates authors to create and share their works. I hope this book offers you valuable insights and inspiration for your projects.

# Preface

The constant evolution of technology requires developers to stay up to date with the latest innovations, practices, and programming languages. PHP 8, with its new features and significant improvements, marks an important step in this ongoing journey of learning and development.

This book is designed as a comprehensive guide to exploring PHP 8 in depth, from the basic fundamentals to the advanced features that define this version. Covering topics such as functions, form management, sessions and cookies, arrays and collections, **object-oriented programming (OOP)**, error and exception handling, data persistence, and advanced development practices, the book is an indispensable resource for anyone involved in web development.

Throughout this book, you will discover the key features of PHP 8 and learn how to apply them to create efficient, reliable, and maintainable web applications. Security and performance optimization best practices will also be discussed, providing you with a solid foundation for designing and implementing robust applications.

This book is aimed at developers who are starting out with PHP and want to learn how to build web applications, as well as experienced developers looking to improve their knowledge in this new version. With practical examples and a step-by-step approach, you will be guided through the essential concepts to become proficient in using PHP 8.

With the support of this book, you will acquire the knowledge and skills necessary to excel in developing modern web applications, making the most of the innovations that PHP 8 has to offer. I hope you find this book a valuable source of information and inspiration.

**Chapter 1 - Introduction to PHP 8:** This chapter introduces PHP, covering its origins and the evolution of PHP 8. It guides readers through setting up their development environment with Laragon and LAMP, offers tips for migrating to PHP 8, and discusses the future prospects of PHP, providing a comprehensive overview for starting PHP development.

**Chapter 2 - Fundamentals with PHP 8:** The chapter explores the core principles of PHP programming, including variable types, control structures, and basic data manipulation. The chapter lays a solid foundation for understanding how PHP processes information and how to write simple, effective code.

**Chapter 3 - Functions in PHP:** This chapter dives into the creation and usage of functions in PHP, highlighting built-in functions as well as how to define custom functions. It covers

function syntax, passing arguments, return values, and the importance of functions in modular coding.

**Chapter 4 - Forms, Sessions and Cookies:** This chapter discusses how to handle user input through forms, maintain session states, and use cookies to personalize user experiences. This chapter provides practical examples to secure data handling and user authentication.

**Chapter 5 - Arrays and Collections:** This chapter focuses on managing groups of data using arrays and collection tools in PHP. It explains array operations, associative arrays, and the use of built-in functions to sort, filter, and manipulate data collections.

**Chapter 6 - OOP Advanced Features of PHP 8+:** The chapter introduces **object-oriented programming (OOP)** concepts in PHP 8+, including classes, objects, inheritance, and polymorphism. This chapter emphasizes the new OOP features and best practices for designing reusable and maintainable code.

**Chapter 7 - Handling Errors and Exceptions:** This chapter covers strategies for robust error handling and exception management in PHP applications. It guides readers through PHP's error reporting mechanisms, custom exception handling, and tips for debugging code efficiently.

**Chapter 8 - Database and Data Persistence with PHP:** This chapter explores how to interact with databases using PHP, with a focus on PDO and MySQLi extensions. The chapter provides insights into performing CRUD operations, preventing SQL injection, and managing data persistence.

**Chapter 9 - Advanced Development with PHP:** We will cover the advanced capabilities of PHP in its recent versions. Exploring the operation and application of RESTful APIs, especially with regard to communication with external services. We will demonstrate best practices for establishing APIs in PHP and the language's most recent additions: Fibers and enums.

**Chapter 10 - Best Practices Security and Performance with PHP:** This chapter concludes the book by addressing critical aspects of web development: security and performance optimization. It discusses secure coding practices, data encryption, safeguarding against common vulnerabilities, and techniques to enhance the speed and efficiency of PHP applications.

## Code Bundle and Coloured Images

Please follow the link to download the  
*Code Bundle* and the *Coloured Images* of the book:

**<https://rebrand.ly/em31l9g>**

The code bundle for the book is also hosted on GitHub at

**<https://github.com/bpbpublications/Unlock-PHP-8-From-Basic-to-Advanced>**.

In case there's an update to the code, it will be updated on the existing GitHub repository.

We have code bundles from our rich catalogue of books and videos available at  
**<https://github.com/bpbpublications>**. Check them out!

## Errata

We take immense pride in our work at BPB Publications and follow best practices to ensure the accuracy of our content to provide with an indulging reading experience to our subscribers. Our readers are our mirrors, and we use their inputs to reflect and improve upon human errors, if any, that may have occurred during the publishing processes involved. To let us maintain the quality and help us reach out to any readers who might be having difficulties due to any unforeseen errors, please write to us at :

**[errata@bpbonline.com](mailto:errata@bpbonline.com)**

Your support, suggestions and feedbacks are highly appreciated by the BPB Publications' Family.

Did you know that BPB offers eBook versions of every book published, with PDF and ePub files available? You can upgrade to the eBook version at [www.bpbonline.com](http://www.bpbonline.com) and as a print book customer, you are entitled to a discount on the eBook copy. Get in touch with us at :

**[business@bpbonline.com](mailto:business@bpbonline.com)** for more details.

At **[www.bpbonline.com](http://www.bpbonline.com)**, you can also read a collection of free technical articles, sign up for a range of free newsletters, and receive exclusive discounts and offers on BPB books and eBooks.

### Piracy

If you come across any illegal copies of our works in any form on the internet, we would be grateful if you would provide us with the location address or website name. Please contact us at **business@bpbonline.com** with a link to the material.

### If you are interested in becoming an author

If there is a topic that you have expertise in, and you are interested in either writing or contributing to a book, please visit **www.bpbonline.com**. We have worked with thousands of developers and tech professionals, just like you, to help them share their insights with the global tech community. You can make a general application, apply for a specific hot topic that we are recruiting an author for, or submit your own idea.

### Reviews

Please leave a review. Once you have read and used this book, why not leave a review on the site that you purchased it from? Potential readers can then see and use your unbiased opinion to make purchase decisions. We at BPB can understand what you think about our products, and our authors can see your feedback on their book. Thank you!

For more information about BPB, please visit **www.bpbonline.com**.

## Join our book's Discord space

Join the book's Discord Workspace for Latest updates, Offers, Tech happenings around the world, New Release and Sessions with the Authors:

<https://discord.bpbonline.com>



# Table of Contents

<b>1. Introduction to PHP 8 .....</b>	<b>1</b>
Introduction .....	1
Structure .....	1
Objectives .....	1
The origin of PHP .....	2
Evolution of major PHP versions .....	3
<i>PHP5</i> .....	3
<i>PHP 5.6</i> .....	3
<i>PHP 6</i> .....	6
<i>PHP 7</i> .....	6
<i>PHP 7.4</i> .....	8
<i>PHP 8.0, 8.1, 8.2, 8.3</i> .....	10
<i>Summary</i> .....	13
Creating development environment .....	14
<i>Laragon on Windows</i> .....	14
<i>Installing Laragon</i> .....	14
<i>Curiosity</i> .....	17
<i>LAMP on Linux/Ubuntu</i> .....	17
<i>Installing LAMP on Linux/Ubuntu</i> .....	17
<i>Installing PHP modules</i> .....	21
<i>Summary</i> .....	22
Tips on migrating to PHP 8 .....	22
<i>Advantages of migration</i> .....	22
<i>Assessing the current environment</i> .....	22
<i>Tools and resources for migrating</i> .....	23
<i>Summary</i> .....	24
Considerations and the future of PHP .....	25
Conclusion .....	26
Points to remember .....	26

<b>2. Fundamentals with PHP 8 .....</b>	<b>27</b>
Introduction .....	27
Structure .....	27
Objectives .....	28
Syntax and basic structures .....	28
<i>Introduction to syntax.....</i>	28
<i>Structure of a PHP script.....</i>	28
<i>Basic syntax.....</i>	29
Variables and data types .....	30
<i>Defining and declaring variables .....</i>	30
<i>Data types.....</i>	31
<i>Type conversion.....</i>	33
<i>Global and local variables.....</i>	34
<i>Variable types .....</i>	35
<i>Predefined variables.....</i>	35
\$_SERVER.....	35
\$_POST .....	36
\$_GET .....	37
\$_COOKIE .....	38
\$_SESSION.....	39
Operators and expressions .....	39
<i>Arithmetic operators.....</i>	40
<i>Comparison operators.....</i>	41
<i>Logical operators.....</i>	43
<i>Concatenation operators.....</i>	45
<i>Operator precedence .....</i>	46
Control structures .....	47
<i>If/else.....</i>	47
<i>Switch.....</i>	47
<i>Ternary operator and Null coalescing operator (??).....</i>	48
<i>Match expression.....</i>	49

Loop structures.....	49
<i>for</i> 50 .....	
<i>while</i> .....	50
<i>Do-while</i> .....	50
Functions and procedures.....	51
<i>Functions</i> .....	51
<i>Procedures</i> .....	51
New features: Named arguments, union types, attributes.....	53
<i>Named arguments</i> .....	53
<i>Definition and usage</i> .....	53
<i>Practical examples</i> .....	53
<i>Union types</i> .....	54
<i>Definition and usage</i> .....	54
<i>Practical examples</i> .....	55
<i>Attributes</i> .....	55
<i>Definition and usage</i> .....	56
<i>Practical examples</i> .....	56
Conclusion .....	57
Points to remember .....	58
<b>3. Functions in PHP .....</b>	<b>59</b>
Introduction .....	59
Structure .....	59
Objectives .....	59
Defining and using functions.....	60
<i>Functions</i> .....	60
<i>Creating functions in PHP</i> .....	60
<i>Good development practices</i> .....	61
<i>Using functions</i> .....	61
<i>Reusing functions</i> .....	62
<i>Variadic functions</i> .....	64
<i>Strict types</i> .....	65

Nullable types.....	65
Named arguments and attributes in functions .....	66
Arguments by value and by reference .....	67
Value arguments.....	67
Argument by reference .....	68
New argument types: mixed, union types.....	69
Anonymous functions, closures, and arrow functions .....	70
Anonymous functions .....	70
Closures .....	71
Arrow functions in PHP .....	72
Native functions in PHP .....	73
String manipulation.....	73
Cut part of a string .....	73
Replace part of a string .....	74
Count number of bytes in a string.....	74
Case conversions .....	75
String comparison.....	76
Search in strings .....	76
Position search in strings.....	76
String formatting .....	77
printf().....	77
sprintf() .....	77
number_format .....	78
date() .....	79
DateTime class .....	80
Multibyte functions.....	81
Introduction to multibyte functions .....	81
Use and practical examples.....	82
New string functions in PHP .....	82
str_contains().....	83

<i>str_starts_with()</i> .....	83
<i>str_ends_with()</i> .....	83
<i>Considerations about the functions</i> .....	84
Introduction and basic concepts of JIT in PHP 8+ .....	84
<i>Working of JIT in PHP</i> .....	84
<i>The influence of JIT on the performance of PHP code</i> .....	85
<i>Situations where JIT can significantly optimize code</i> .....	85
<i>Cases where JIT optimization may be less noticeable</i> .....	85
<i>Good programming practices and code optimization in conjunction with JIT</i> .....	86
Conclusion .....	86
Points to remember .....	86
<b>4. Forms, Sessions and Cookies</b> .....	<b>89</b>
Introduction .....	89
Structure .....	89
Objectives .....	89
Forms data.....	90
<i>Introduction to forms</i> .....	90
<i>Forms in HTML</i> .....	91
<i>Receiving form data in PHP</i> .....	92
<i>Form data validation in PHP</i> .....	93
<i>filter_var()</i> .....	93
<i>filter_input()</i> .....	93
<i>filter_var_array()</i> .....	94
<i>Data types for validations</i> .....	94
Forms of files.....	95
<i>Structure of a file form</i> .....	95
<i>Storing and manipulating files on the PHP server</i> .....	96
<i>File handling types</i> .....	97
<i>Upload validations and security</i> .....	99
<i>Good security practices for forms</i> .....	101

Cross-site request forgery.....	101
Protection against cross-site scripting attacks.....	103
Limiting form submission attempts.....	104
Preparation of SQL statements.....	104
Sessions.....	105
Using sessions.....	105
Session data storage.....	106
Retrieving and manipulating session data.....	107
User session control.....	108
Cookies.....	108
Setting a cookie in PHP.....	108
Retrieving cookie data in PHP.....	109
Double submit cookie.....	110
Working of double submit cookie.....	110
Advantages and considerations.....	111
Good cookie security practices.....	111
Real project: Task manager.....	113
First steps.....	114
Where to start.....	114
Files structures.....	115
config_global.php.....	115
process_login.php.....	116
login.php.....	118
index.php.....	119
Project navigation result.....	120
Project considerations.....	121
Conclusion.....	121
Points to remember.....	122
<b>5. Arrays and Collections.....</b>	<b>123</b>
Introduction.....	123
Structure.....	123
Objectives.....	123

Indexed arrays and associative .....	124
<i>Introduction to arrays</i> .....	124
<i>Indexed array</i> .....	124
<i>Declaring indexed arrays</i> .....	124
<i>Manipulating values</i> .....	125
<i>Modifying values</i> .....	127
<i>Iterating arrays with a for loop</i> .....	128
<i>Array iteration with foreach loop</i> .....	130
<i>Associative arrays</i> .....	131
<i>Declaring associative array</i> .....	131
<i>Manipulating values</i> .....	132
<i>Modifying values</i> .....	132
<i>Iterating arrays with the for loop</i> .....	132
<i>Iteration with the foreach loop</i> .....	133
<i>Multidimensional arrays</i> .....	134
<i>Multidimensional indexed arrays</i> .....	134
<i>Multidimensional associative arrays</i> .....	136
<i>Iteration with the foreach loop</i> .....	137
<i>Use of arrays in real case examples</i> .....	139
<i>Storing form data</i> .....	139
<i>Manipulation of data coming from a database</i> .....	140
<i>Advanced data structures</i> .....	140
Functions for manipulating arrays.....	141
<i>PHP native functions</i> .....	141
<i>array_push()</i> .....	141
<i>array_pop()</i> .....	141
<i>array_shift()</i> .....	142
<i>array_unshift()</i> .....	142
<i>array_slice()</i> .....	142
<i>array_splice()</i> .....	143

<i>array_map()</i> .....	143
<i>array_filter()</i> .....	144
<i>array_reduce()</i> .....	145
<i>implode()</i> .....	146
<i>explode()</i> .....	146
<i>sort()</i> .....	147
<i>resort()</i> .....	147
<i>asort()</i> .....	147
<i>arsort()</i> .....	148
<i>ksort()</i> .....	148
<i>krsort()</i> .....	148
<i>PHP enhanced functions</i> .....	149
<i>Array unpacking</i> .....	149
<i>Array unpacking in foreach loop</i> .....	149
<i>array_is_list()</i> .....	150
<i>array_key_first()</i> .....	151
<i>array_key_last()</i> .....	151
Real project: Task manager .....	152
<i>New implementations</i> .....	152
<i>Files structures</i> .....	152
<i>Change index.php</i> .....	153
<i>process_task_class.php</i> .....	155
<i>process_tasks.php</i> .....	157
<i>Browsing result</i> .....	158
<i>Considerations</i> .....	159
Conclusion .....	160
Points to remember .....	160
<b>6. OOP Advanced Features of PHP 8+</b> .....	<b>161</b>
Introduction .....	161
Structure .....	161

Objectives .....	162
Classes and objects.....	162
<i>Introduction to class</i> .....	162
<i>The reserved word \$this</i> .....	164
<i>Using the class</i> .....	164
<i>Encapsulation</i> .....	165
<i>PHP magic methods</i> .....	167
<i>New PHP updates</i> .....	170
<i>Typed properties</i> .....	171
<i>Readonly properties</i> .....	171
Inheritance, polymorphism and abstraction.....	171
<i>Working of use of inheritance</i> .....	172
<i>Polyformism in inheritance</i> .....	173
<i>Accessing parent class with parent::</i> .....	174
<i>Abstract classes</i> .....	175
<i>Final classes and methods</i> .....	177
<i>Final classes</i> .....	178
<i>Final methods</i> .....	178
Interfaces and traits.....	178
<i>Interfaces</i> .....	179
<i>Declaration of interfaces</i> .....	179
<i>Interface implementation</i> .....	179
<i>Multiple interfaces</i> .....	180
<i>Benefits of interfaces</i> .....	180
<i>Traits</i> .....	181
<i>Declaration of traits</i> .....	181
<i>Using traits in classes</i> .....	182
<i>Benefits of traits</i> .....	182
<i>Trait precedence</i> .....	182
<i>Method aliases</i> .....	183
PHP standards recommendation.....	184

<i>Introduction to PSRs</i> .....	184
<i>Purpose of PSRs</i> .....	185
<i>Examples of PSRs</i> .....	185
Composer: Dependency manager for PHP .....	185
<i>Installing Composer</i> .....	186
<i>Starting the project with Composer</i> .....	186
<i>Using Composer</i> .....	187
Namespaces and autoload.....	187
Namespaces .....	187
<i>Introduction to namespaces</i> .....	187
<i>Declaring a namespace</i> .....	188
<i>Using classes from other namespaces</i> .....	188
<i>Using aliases for classes from different namespaces</i> .....	189
Class autoload.....	189
<i>Working of autoload</i> .....	189
<i>Using spl_autoload_register in PHP</i> .....	190
<i>Autoload with or without namespace</i> .....	191
<i>Autoload with Composer</i> .....	191
<i>Common problems and solutions</i> .....	193
Model-view-controller architecture pattern.....	193
SOLID principles of object oriented design .....	194
The future of OOP with PHP.....	197
Conclusion .....	197
Key terms .....	198
<b>7. Handling Errors and Exceptions.....</b>	<b>199</b>
Introduction .....	199
Structure .....	199
Objectives .....	200
Errors and exceptions.....	200
<i>Errors in PHP</i> .....	200
<i>Common types of errors</i> .....	200

---

<i>Identifying errors</i> .....	201
<i>Exceptions in PHP</i> .....	202
<i>Exceptions versus errors</i> .....	202
Try-catch and finally .....	202
<i>Flagging exceptions with throw</i> .....	202
<i>Catching exceptions with try-catch</i> .....	203
<i>Are errors caught in try-catch?</i> .....	204
<i>Using finally</i> .....	205
<i>Catch multiples</i> .....	206
Exception and throwable class .....	208
<i>Exception class</i> .....	208
<i>Basic structure</i> .....	208
<i>Using the Exception class</i> .....	209
<i>Throwable interface</i> .....	210
Custom Exception class.....	210
<i>Creating a custom exception</i> .....	211
Real project: Task manager .....	212
<i>Configuring the environment</i> .....	212
<i>Folder structure</i> .....	213
<i>Layouts</i> .....	214
<i>Traits</i> .....	215
<i>System folders</i> .....	216
<i>App.php</i> .....	216
<i>View.php</i> .....	217
<i>Router.php</i> .....	219
<i>Redirect.php</i> .....	220
<i>Friendly URL</i> .....	220
<i>Page 404.php</i> .....	221
<i>Refactoring authentication</i> .....	221
<i>Abstract controller class</i> .....	223
<i>View error</i> .....	225

Login controller.....	225
Home controller.....	229
routes.php.....	230
Router.php.....	231
App.php.....	232
Authenticate.php middleware .....	233
session_start() .....	234
Refactoring the main page.....	235
Implement View.php .....	236
Refact Controller.php .....	236
Implement layouts/index.php .....	238
Refactor Home.php controller .....	239
Create model Task.php.....	240
Create Tasks.php controller .....	241
View for tasks.....	245
Defining the routes.....	247
Conclusion .....	248
Points to remember .....	249
<b>8. Database and Data Persistence with PHP .....</b>	<b>251</b>
Introduction .....	251
Structure .....	251
Objectives .....	252
Connection and manipulation of databases with PHP .....	252
MySQL.....	252
Setting up a MySQL environment .....	253
Connecting to MySQL using PHP.....	255
What is the MySQLi extension? .....	255
mysqli procedural.....	256
Object-Oriented mysqli.....	259
Procedural versus object orientation with mysqli.....	261

<i>Basic operations in MySQL through PHP .....</i>	262
<i>Entering data .....</i>	262
<i>Reading data .....</i>	264
<i>Updating data .....</i>	265
<i>Deleting data.....</i>	266
<i>Prepared statements .....</i>	267
<i>SQL queries .....</i>	267
<i>Using prepared statements.....</i>	267
<i>Multiple SQL queries.....</i>	269
<i>Benefits of using prepared statements .....</i>	272
<i>PHP Data Objects and database abstraction.....</i>	273
<i>Introduction to PHP Data Objects .....</i>	273
<i>Connecting to different databases using PDO.....</i>	273
<i>Prepared statements with PDO .....</i>	275
<i>Performing operations using PDO .....</i>	278
<i>Inserting data.....</i>	278
<i>Reading data .....</i>	279
<i>Updating data .....</i>	280
<i>Deleting data.....</i>	280
<i>Transactions with PDO.....</i>	281
<i>PDO exceptions.....</i>	282
<i>Comparison between mysqli and PDO .....</i>	283
<i>Advanced database features and performance optimization.....</i>	283
<i>Good practices in the database.....</i>	284
<i>Caching of queries and results.....</i>	285
<i>PDO with Memcached.....</i>	285
<i>Object-relational mapping.....</i>	286
<i>Real project: Task manager .....</i>	288
<i>config/database.php.....</i>	289
<i>bootstrap.php .....</i>	290
<i>Class DatabasePDO.php.....</i>	295

<i>Implementing Model.php</i> .....	298
<i>app/System/QueryBuilder.php</i> .....	298
<i>app/System/Model.php</i> .....	301
<i>Implementing the User.php and Task.php model</i> .....	305
<i>Refactor User.php</i> .....	306
<i>Refactor Task.php</i> .....	308
Conclusion .....	312
Points to remember .....	313
<b>9. Advanced Development with PHP</b> .....	<b>315</b>
Introduction .....	315
Structure .....	315
Objectives .....	316
RESTful APIs .....	316
<i>REST architecture</i> .....	316
<i>Consuming external RESTful with cURL</i> .....	318
<i>Handling different data formats</i> .....	321
XML .....	322
JSON .....	323
<i>Ensuring secure communication</i> .....	325
Creating APIs in PHP .....	326
Routing .....	326
Middleware .....	327
Error handling .....	328
Using nouns .....	329
Data retrieval: Filter, sort, page .....	330
Rate limiting .....	330
Versioning APIs .....	331
Documentation and testing tools for APIs .....	332
Documentation .....	332
Testing tools .....	333
JSON Web Tokens with PHP .....	334

Fibers and Enums.....	337
<i>Fibers</i> .....	337
<i>Working of fibers</i> .....	338
<i>How fibers can benefit APIs</i> .....	339
<i>Real example using fibers</i> .....	340
<i>Enum</i> .....	342
<i>Introduction to Enum</i> .....	342
<i>Using Enums</i> .....	342
Real project: Task manager .....	343
<i>Creating API</i> .....	344
<i>Route.php class</i> .....	344
<i>Refactor Router.php</i> .....	345
<i>routes.php</i> .....	348
<i>App.php</i> .....	349
<i>Response.php class</i> .....	349
<i>Controller.php refactoring</i> .....	351
<i>RegisterUsers.php controller</i> .....	351
<i>Challenge</i> .....	354
Conclusion .....	355
Points to remember .....	355
<b>10. Best Practices Security and Performance with PHP .....</b>	<b>357</b>
Introduction .....	357
Structure .....	357
Objectives .....	357
Best security practices .....	358
<i>Principle of least privilege</i> .....	358
<i>Database</i> .....	358
<i>PHP files on the server</i> .....	359
<i>APIs and external services</i> .....	359
<i>Disable error display in production environments</i> .....	360

<i>Using Content Security Policy</i> .....	361
<i>Protection against brute force attacks and rate limiting</i> .....	362
<i>Avoid remote file inclusion</i> .....	363
Preventing common attacks .....	364
SQL injection .....	364
<i>Basic malicious query: Authentication without valid password</i> .....	365
<i>Extracting data: Using UNION SELECT</i> .....	365
<i>Preventing SQL injection</i> .....	366
Cross-site scripting attack .....	368
<i>Preventing XSS attacks</i> .....	368
Cross-site request forgery .....	369
<i>Executing a CSRF attack</i> .....	369
<i>Preventing CSRF</i> .....	370
Debugging techniques .....	372
Xdebug .....	372
<i>Using Xdebug</i> .....	372
Sentry .....	376
<i>Error log</i> .....	376
<i>Sentry: An error monitoring tool</i> .....	376
PHPStan .....	378
<i>Static code analysis tools</i> .....	378
<i>Using PHPStan</i> .....	378
Just-in-time compiler .....	379
<i>Exploring JIT compilation</i> .....	379
Conclusion .....	381
Points to remember .....	381
<b>Index</b> .....	<b>383-392</b>

# CHAPTER 1

# Introduction to PHP 8

## Introduction

In this first chapter, we will do a little introduction and analysis of PHP versions and explore the evolution of the language from version 5 to the most recent update, PHP 8+. In its evolution, PHP has increased and is continuously introducing new features, thus becoming one of the outstanding programming languages in terms of web development.

## Structure

The chapter covers the following topics:

- The origin of PHP
- Evolution of major PHP versions
- Creating development environment
- Tips on migrating to PHP 8
- Considerations and the future of PHP

## Objectives

After reading this chapter, you will be able to address the main changes and improvements introduced in these new versions. In addition to addressing the migration of projects

developed in PHP 5 to PHP 8+, we will also highlight new features and improvements that make PHP 8+ a good choice for developers. Finally, you will be able to understand PHP's future trends and expectations, giving an overview of what developers can expect from this popular programming language.

## The origin of PHP

Before we get into any kind of content in this book, it is important for us to remember the origin and journey of PHP, one of the most widely used server-side scripting languages, started in 1994. Originally created by *Rasmus Lerdorf*, a programmer from Greenland, PHP started as a simple set of CGI scripts for tracking visitors to your online resume. The original name of PHP was **Personal Home Page Tools**, a primitive but functional way of building dynamic websites.

Realizing the potential of what he had created, *Lerdorf* decided to improve his scripts, adding features for working with web forms and communicating with databases. In 1995, he released the PHP/FI (*Form Interpreter*) version, which formed the basis of what would become the PHP we know today.

In version PHP 3, the language started to look more like the modern form of PHP, thanks to the work of *Andi Gutmans* and *Zeev Suraski*. They needed a more dynamic language for the project they were developing at the time at the university, so they rewrote and expanded the PHP language, making it a more robust and flexible solution for web development.

PHP 4, released in 2000 as the work of *Andi* and *Zeev*, brought several improvements, including more object-oriented features. However, it was PHP 5, released in 2004, that solidified PHP's position as one of the top programming languages for web development. PHP 5 introduced a complete object model, as well as error handling improvements and several powerful extensions.

Throughout its evolution, PHP has managed to maintain a careful balance between ease of use and power, which has contributed to its enduring popularity as a web programming language. An example of a PHP code is illustrated in the following figure:

```
<!--include /text/header.html-->

<!--getenv HTTP_USER_AGENT-->
<!--ifsubstr $exec_result Mozilla-->
    Hey, you are using Netscape!<p>
<!--endif-->

<!--sql database select * from table where user='$username'-->
<!--ifless $numentries 1-->
    Sorry, that record does not exist<p>
<!--endif exit-->
Welcome <!--$user-->!<p>
You have <!--$index:0--> credits left in your account.<p>

<!--include /text/footer.html-->
```

Figure 1.1: PHP/FI code<sup>1</sup>

---

<sup>1</sup> source: <https://www.php.net/manual/en/history.php.php>

# Evolution of major PHP versions

Let us learn a bit more about the PHP releases that have had the greatest strengths.

## PHP5

The release of PHP 5 in July 2004 was a major milestone in PHP history. This release included additional new features and improvements related to PHP 4 as it had a significant impact on websites where web development was not done with PHP.

One of the main features of PHP 5 is the introduction of a full object model. Before PHP 5, the language had specific functions for programming a specific object (OOP), but only to a limited extent. Classes and objects are more powerful and flexible in PHP 5. Classes can give the programmatic object a more beautiful and practical look and better effectiveness due to restricted visibility (public, private, and protected), and due to its properties and methods (class functions).

PHP 5 also introduces exceptions inspired by other languages like Java and C++. Exceptions are used to make errors more effective and to help developers constantly detect and then check for errors and checks during script execution.

Another benefit of PHP 5 is the introduction of new and more powerful extensions. For example, the PDO and MySQLi extensions provide new functionality for connecting to MySQL databases. The SimpleXML extension simplifies the editing and analysis of XML data. It also added SOAP pricing to ease integration with web-based services over SOAP.

PHP 5 is the first programming language for the web, giving developers a greater level of control and flexibility, as well as ease of use. This enables the creation of complex and more powerful web applications as well as the implementation of vague innovation projects on the web.

## PHP 5.6

PHP 5.6, released in August 2014, was the last major release of PHP 5 and brought several significant new features and improvements. The main goal of PHP 5.6 was to increase language consistency, improve performance, and add functionality requested by users and the development community.

Among the most important and striking features of PHP 5.6, we can highlight the following:

- **Variable argument support:** PHP 5.6 introduced the splat operator syntax (...) for representing variable arguments in functions and methods. This allowed developers to pass a variable number of arguments to a function, making PHP more flexible and less error prone.

The following is an example of using the splat operator (...). We are defining a function that can take any number of arguments and compute the sum of all of them:

```
1. <?php
2. function sumNumbers(...$numbers) {
3.     $total = 0;
4.     foreach ($numbers as $number) {
5.         $total += $number;
6.     }
7.     return $total;
8. }
9.
10. echo sumNumbers(1, 2, 3, 4, 5); // Output: 15
```

- **Importing functions and namespace constants:** Importing functions and namespace constants allowed users to simplify the code and increase readability. In the following example, we see how the code can be simplified and made more readable by importing functions and namespace constants. We define a function and a constant in a namespace and then import them to use:

```
1. <?php
2. namespace MyNamespace;
3.
4. function myFunction() {
5.     echo "It's works";
6. }
7.
8. const MY_FIRST_CONST = 123;
```

For example:

```
1. <?php
2. use MyNamespace\myFunction;
3. use MyNamespace\MY_FIRST_CONST;
4.
5.
6. myFunction(); // Output: It's works
7. echo MY_FIRST_CONST; // Output: 123
```

- **Exponentiation operator:** PHP 5.6 introduced the exponentiation operator (\*\*), making it easier to perform complex mathematical operations.

Here is a simple example of how the exponentiation operation works. In this case, we are raising 2 to the power of 3:

```
1. <?php
2. $total = 2 ** 3; // 2 raised to the power of 3
3. echo $total; // Output: 8
```

- **GMP compatibility in common functions:** Arbitrary precision number operations became easier with PHP 5.6 due to compatibility with the **GNU Multiple Precision (GMP)** library.

This example demonstrates the ease of performing operations with arbitrary precision numbers with the GMP library. We are adding two numbers of arbitrary precision:

```
1. <?php
2. $number1 = gmp_init(123);
3. $number2 = gmp_init(456);
4.
5. $total = gmp_add($number1, $number2);
6. echo gmp_strval($total); // Output: 579
```

- **Hash function improvements:** PHP 5.6 introduced **hash\_equals()** for secure string comparison, which significantly improved the security of string handling in PHP.

The example below shows how to use the **hash\_equals()** function for safe string comparison. In this case, 'hello' and 'HELLO' are different, so the result will be 'Strings are different.':

```
1. <?php
2. $string1 = 'hello';
3. $string2 = 'HELLO';
4.
5. if (hash_equals($string1, $string2)) {
6.     echo 'The strings are the same.';
7. } else {
8.     echo 'Strings are different.';
9. }
```

- **Deeper integration of Zend OPcache:** OPcache, which compiles PHP code into bytecode to speed up its performance, has been further integrated in PHP 5.6, significantly improving the performance of PHP applications.