

Table of Contents

About the Author	xix
About the Technical Reviewer	xxi
Acknowledgments	xxiii
Introduction	xxv
Part I: Objects	1
Chapter 1: PHP: Design and Management	3
The Problem	3
PHP and Other Languages	5
About This Book	8
Objects	8
Patterns	9
Practice	9
What's New in the Sixth Edition	11
Summary	11
Chapter 2: PHP and Objects	13
The Accidental Success of PHP Objects	13
In the Beginning: PHP/FI	13
Syntactic Sugar: PHP 3	14
PHP 4 and the Quiet Revolution	14
Change Embraced: PHP 5	17
PHP 7: Closing the Gap	18
PHP 8: The Consolidation Continues	19
Advocacy and Agnosticism: The Object Debate	19
Summary	20

TABLE OF CONTENTS

- Chapter 3: Object Basics 21**
 - Classes and Objects..... 21
 - A First Class..... 21
 - A First Object (or Two) 22
 - Setting Properties in a Class..... 24
 - Working with Methods 27
 - Creating a Constructor Method 29
 - Constructor Property Promotion 31
 - Default Arguments and Named Arguments..... 33
 - Arguments and Types..... 34
 - Primitive Types 35
 - Some Other Type-Checking Functions..... 39
 - Type Declarations: Object Types 40
 - Type Declarations: Primitive Types 43
 - mixed Types 45
 - Union Types 47
 - Nullable Types 50
 - Return Type Declarations 50
 - Inheritance..... 51
 - The Inheritance Problem 52
 - Working with Inheritance 59
 - Public, Private, and Protected: Managing Access to Your Classes..... 67
 - Typed Properties..... 71
 - Summary..... 77
- Chapter 4: Advanced Features..... 79**
 - Static Methods and Properties..... 79
 - Constant Properties 85
 - Abstract Classes 86
 - Interfaces 89

Traits	92
A Problem for Traits to Solve	93
Defining and Using a Trait.....	94
Using More Than One Trait.....	95
Combining Traits and Interfaces	97
Managing Method Name Conflicts with <code>insteadof</code>	98
Aliasing Overridden Trait Methods.....	99
Using Static Methods in Traits	100
Accessing Host Class Properties	101
Defining Abstract Methods in Traits.....	102
Changing Access Rights to Trait Methods	104
Late Static Bindings: The <code>static</code> Keyword.....	105
Handling Errors	109
Exceptions	112
Final Classes and Methods	122
The Internal Error Class	124
Working with Interceptors.....	125
Defining Destructor Methods	134
Copying Objects with <code>__clone()</code>	136
Defining String Values for Your Objects.....	140
Callbacks, Anonymous Functions, and Closures	142
Anonymous Classes	150
Summary.....	152
Chapter 5: Object Tools.....	153
PHP and Packages	153
PHP Packages and Namespaces	154
Autoload	167
The Class and Object Functions.....	172
Looking for Classes	174
Learning About an Object or Class	175

TABLE OF CONTENTS

- Getting a Fully Qualified String Reference to a Class 177
- Learning About Methods..... 178
- Learning About Properties 181
- Learning About Inheritance 181
- Method Invocation 182
- The Reflection API 185
 - Getting Started 185
 - Time to Roll Up Your Sleeves 186
 - Examining a Class 189
 - Examining Methods..... 191
 - Examining Method Arguments..... 194
 - Using the Reflection API 197
- Attributes 202
- Summary..... 208
- Chapter 6: Objects and Design 209**
 - Defining Code Design..... 209
 - Object-Oriented and Procedural Programming..... 210
 - Responsibility 216
 - Cohesion..... 217
 - Coupling 217
 - Orthogonality 217
 - Choosing Your Classes 218
 - Polymorphism 219
 - Encapsulation 221
 - Forget How to Do It 223
 - Four Signposts 224
 - Code Duplication 224
 - The Class Who Knew Too Much 224
 - The Jack of All Trades..... 225
 - Conditional Statements 225

The UML	225
Class Diagrams.....	226
Sequence Diagrams	235
Summary.....	238
Part II: Patterns	239
Chapter 7: What Are Design Patterns? Why Use Them?	241
What Are Design Patterns?	241
A Design Pattern Overview	244
Name	244
The Problem	245
The Solution.....	245
Consequences	246
The Gang of Four Format	246
Why Use Design Patterns?.....	247
A Design Pattern Defines a Problem	247
A Design Pattern Defines a Solution.....	248
Design Patterns Are Language Independent	248
Patterns Define a Vocabulary	248
Patterns Are Tried and Tested	249
Patterns Are Designed for Collaboration.....	249
Design Patterns Promote Good Design.....	250
Design Patterns Are Used by Popular Frameworks	250
PHP and Design Patterns	250
Summary.....	251
Chapter 8: Some Pattern Principles.....	253
The Pattern Revelation.....	253
Composition and Inheritance	254
The Problem	254
Using Composition.....	258

TABLE OF CONTENTS

- Decoupling 262
 - The Problem 262
 - Loosening Your Coupling 264
- Code to an Interface, Not to an Implementation..... 267
- The Concept That Varies..... 269
- Patternitis..... 270
- The Patterns..... 270
 - Patterns for Generating Objects 271
 - Patterns for Organizing Objects and Classes 271
 - Task-Oriented Patterns..... 271
 - Enterprise Patterns..... 271
 - Database Patterns 271
- Summary..... 271
- Chapter 9: Generating Objects 273**
 - Problems and Solutions in Generating Objects..... 273
 - The Singleton Pattern 280
 - The Problem 280
 - Implementation 281
 - Consequences 284
 - Factory Method Pattern 285
 - The Problem 285
 - Implementation 289
 - Consequences 292
 - Abstract Factory Pattern 293
 - The Problem 293
 - Implementation 295
 - Consequences 298
 - Prototype..... 300
 - The Problem 301
 - Implementation 302

Pushing to the Edge: Service Locator 307

Splendid Isolation: Dependency Injection 309

 The Problem 309

 Implementation 310

 Consequences 328

Summary..... 330

Chapter 10: Patterns for Flexible Object Programming..... 331

 Structuring Classes to Allow Flexible Objects..... 331

 The Composite Pattern..... 332

 The Problem 332

 Implementation 336

 Consequences 342

 Composite in Summary 347

 The Decorator Pattern..... 347

 The Problem 347

 Implementation 350

 Consequences 356

 The Facade Pattern..... 357

 The Problem 357

 Implementation 360

 Consequences 361

 Summary..... 362

Chapter 11: Performing and Representing Tasks 363

 The Interpreter Pattern..... 363

 The Problem 364

 Implementation 365

 Interpreter Issues 377

 The Strategy Pattern 377

 The Problem 377

 Implementation 379

TABLE OF CONTENTS

The Observer Pattern	383
Implementation	387
The Visitor Pattern.....	395
The Problem	395
Implementation	398
Visitor Issues	405
The Command Pattern	405
The Problem	406
Implementation	406
The Null Object Pattern.....	413
The Problem	414
Implementation	417
Summary.....	419
Chapter 12: Enterprise Patterns	421
Architecture Overview.....	422
The Patterns	422
Applications and Layers	423
Cheating Before We Start.....	426
Registry	426
Implementation	428
The Presentation Layer	434
Front Controller.....	435
Application Controller	450
Page Controller	468
Template View and View Helper	475
The Business Logic Layer	479
Transaction Script.....	479
Domain Model	485
Summary.....	490

Chapter 13: Database Patterns	491
The Data Layer.....	491
Data Mapper	492
The Problem	492
Implementation	493
Consequences	513
Identity Map	514
The Problem	514
Implementation	516
Consequences	520
Unit of Work	520
The Problem	521
Implementation	521
Consequences	528
Lazy Load	528
The Problem	528
Implementation	529
Consequences	531
Domain Object Factory.....	532
The Problem	532
Implementation	532
Consequences	534
The Identity Object.....	536
The Problem	536
Implementation	537
Consequences	545

TABLE OF CONTENTS

The Selection Factory and Update Factory Patterns.....	545
The Problem	545
Implementation	546
Consequences	551
What's Left of Data Mapper Now?	552
Summary.....	555
Part III: Practice.....	557
Chapter 14: Good (and Bad) Practice	559
Beyond Code	560
Borrowing a Wheel.....	560
Playing Nice	563
Giving Your Code Wings.....	564
Standards.....	565
Vagrant.....	566
Testing.....	567
Continuous Integration.....	568
Summary.....	569
Chapter 15: PHP Standards	571
Why Standards?	571
What Are PHP Standards Recommendations?	572
Why PSR in Particular?	573
Who Are PSRs for?.....	574
Coding with Style	575
PSR-1 Basic Coding Standard	575
PSR-12 Extended Coding Style.....	579
Checking and Fixing Your Code	586
PSR-4 Autoloading	589
The Rules That Matter to Us	589
Summary.....	593

Chapter 16: PHP Using and Creating Components with Composer	595
What Is Composer?	596
Installing Composer	596
Installing a (Set of) Package(s)	596
Installing a Package from the Command Line.....	598
Versions	598
require-dev.....	600
Composer and Autoload.....	602
Creating Your Own Package.....	603
Adding Package Information	603
Platform Packages	604
Distribution Through Packagist.....	605
Keeping It Private.....	609
Summary.....	611
Chapter 17: Version Control with Git	613
Why Use Version Control?	613
Getting Git	615
Using an Online Git Repository.....	616
Configuring a Git Server.....	618
Creating the Remote Repository.....	619
Beginning a Project.....	621
Cloning the Repository	625
Updating and Committing	626
Adding and Removing Files and Directories	630
Adding a File.....	631
Removing a File.....	631
Adding a Directory.....	632
Removing Directories	633
Tagging a Release.....	633
Branching a Project.....	634
Summary.....	644

TABLE OF CONTENTS

Chapter 18: Testing with PHPUnit..... 645

- Functional Tests and Unit Tests..... 646
- Testing by Hand..... 646
- Introducing PHPUnit..... 650
 - Creating a Test Case..... 651
 - Assertion Methods..... 654
 - Testing Exceptions..... 655
 - Running Test Suites..... 656
 - Constraints..... 657
 - Mocks and Stubs..... 660
 - Tests Succeed When They Fail..... 664
- Writing Web Tests..... 670
 - Refactoring a Web Application for Testing..... 670
 - Simple Web Testing..... 673
 - Introducing Selenium..... 676
- A Note of Caution..... 684
- Summary..... 686

Chapter 19: Automated Build with Phing..... 687

- What Is Phing?..... 688
- Getting and Installing Phing..... 689
- Composing the Build Document..... 689
 - Targets..... 692
 - Properties..... 695
 - Types..... 704
 - Tasks..... 711
- Summary..... 717

Chapter 20: Vagrant	719
The Problem.....	719
A Little Setup.....	720
Choosing and Installing a Vagrant Box.....	721
Mounting Local Directories on the Vagrant Box.....	723
Provisioning.....	725
Setting Up the Web Server.....	727
Setting Up MariaDB.....	728
Configuring a Hostname.....	729
Wrapping It Up.....	731
Summary.....	732
Chapter 21: Continuous Integration	733
What Is Continuous Integration?.....	733
Preparing a Project for CI.....	735
Installing Jenkins Plug-ins.....	749
Setting Up the Git Public Key.....	750
Installing a Project.....	751
Running the First Build.....	756
Configuring the Reports.....	757
Triggering Builds.....	760
Summary.....	763
Chapter 22: Objects, Patterns, Practice	765
Objects.....	765
Choice.....	766
Encapsulation and Delegation.....	766
Decoupling.....	767
Reusability.....	768
Aesthetics.....	768

TABLE OF CONTENTS

Patterns.....	769
What Patterns Buy Us.....	770
Patterns and Principles of Design	771
Practice	773
Testing	774
Standards	774
Version Control	775
Automated Build	775
Continuous Integration	776
What I Missed	776
Summary.....	778
Appendix A: Bibliography	781
Books	781
Articles.....	782
Sites	782
Appendix B: A Simple Parser	785
The Scanner	785
The Parser.....	797
Index.....	817

Introduction

When I first conceived of this book, object-oriented design in PHP was an esoteric topic. The intervening years have not only seen the inexorable rise of PHP as an object-oriented language but also the march of the framework. Frameworks are incredibly useful, of course. They manage the guts and the glue of many (perhaps, these days, most) web applications. What's more, they often exemplify precisely the principles of design that this book explores.

There is, though, a danger for developers here, as there is in all useful APIs. This is the fear that one might find oneself relegated to userland, forced to wait for remote gurus to fix bugs or add features at their whim. It's a short step from this standpoint to a kind of exile in which one is left regarding the innards of a framework as advanced magic and one's own work as not much more than a minor adornment stuck up on top of a mighty unknowable infrastructure.

Although I'm an inveterate reinventor of wheels, the thrust of my argument is not that we should all throw away our frameworks and build MVC applications from scratch (at least not always). It is rather that, as developers, we should understand the problems that frameworks solve and the strategies they use to solve them. We should be able to evaluate frameworks not only functionally but in terms of the design decisions their creators have made and to judge the quality of their implementations. And yes, when the conditions are right, we should go ahead and build our own spare and focused applications and, over time, compile our own libraries of reusable code.

I hope this book goes some way toward helping PHP developers apply design-oriented insights to their platforms and libraries and provides some of the conceptual tools needed when it's time to go it alone.