Table of Contents

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.