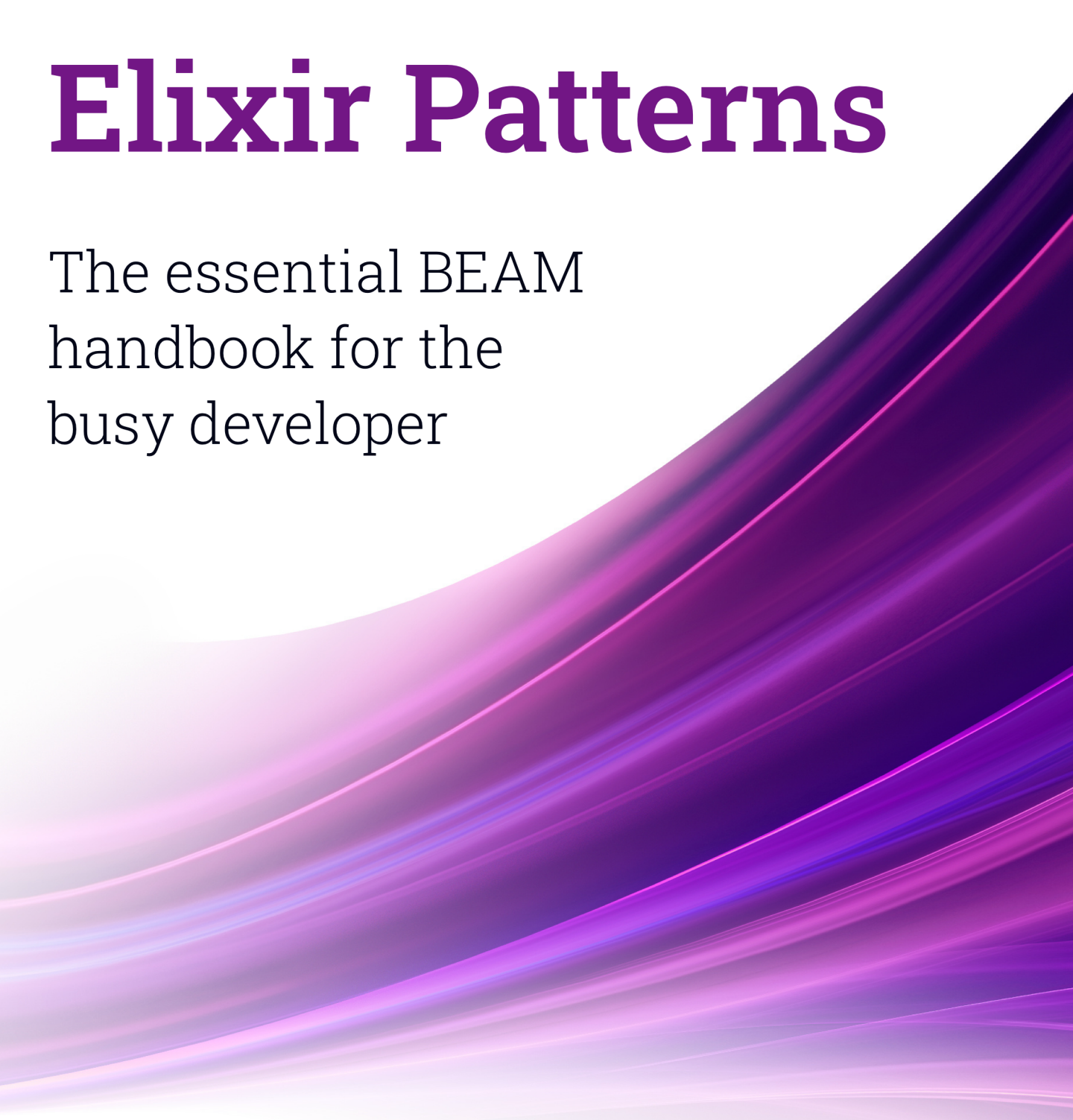
Elixir Patterns

The essential BEAM handbook for the busy developer



Alexander Koutmos
Edited by Hugo Baraúna

Contact team@elixirpatterns.dev for regarding errata and support

Alexander Koutmos
Visit my website at https://akoutmos.com

Hugo Baraúna
Visit my website at https://elixir-radar.com

Printed in the United States of America

First Printing: March 2025
Printed by KDP

ISBN 979-83-07689-76-9

Introduction

Hello and welcome to Elixir Patterns! Before we begin, we would like to thank you for your support. Writing and publishing a book is an immense undertaking, and it is great to know that you decided to read this particular book when there are so many great publications out there. Thank you! With that being said, we would like to discuss some of the motivations that led to this book as it will give you a sense for why it exists and how you can leverage it.

If you have ever programmed in an Object Oriented Programming language before, you have most likely heard of (or even used) some Object Oriented design patterns. Things like the Factory pattern, the Adapter pattern, the Singleton pattern, and the Builder pattern, to name a few. While most of these patterns are not necessarily applicable to Functional Programming languages, the idea of having a "go-to" toolbox of patterns that you can leverage is an enticing idea. Often times, these design patterns are inherently abstract, and do not aim to leverage any of the unique properties of any specific run-time or language. As such, it can sometimes be difficult to know what patterns to use in certain circumstances.

Given that this book will be focusing on Elixir and the Erlang virtual machine, we will take a slightly different approach to design patterns. Specifically, this book aims to surface the powerful and unique characteristics of the Erlang virtual machine (or BEAM for short) and show you how you can go about solving every day problems in a simple yet scalable way.

Not only will you learn how to better leverage the tools that are at your disposal courtesy of Erlang and the BEAM, but you will also learn how to better utilize Functional Programming in order to achieve your goals in a clear and concise way.

What You Will Learn

In the first few chapters you will learn about the Erlang and Elixir standard libraries and useful ways that they can be used. This will include covering some of the data structure modules that are available to you via Erlang and some other utility modules that can come in handy in certain situations. You'll then be introduced to a handful of Elixir modules, and you'll learn about some clever ways that they can be used. Some of these modules include the Task, Enum and Stream modules.

Once you have a good sense for the Elixir and Erlang fundamentals, you'll get into the meat and potatoes of the book and learn about processes, GenServers, Supervisors and how they work exactly. After you have a good grasp on how to write and incorporate GenServers into your application, you'll learn about some common patterns and when to reach for these patterns when you write your own applications.

After that, we'll go over how to package Elixir applications into a release and how to ensure that our GenServers and applications are configured properly. Specifically, we'll go over how we can configure our various resources depending on the running environment and how we can leverage the Adapter pattern to swap out module implementations as needed.

Who is This Book for

In order to get the most out of this book, it is recommended that you have some familiarity with Elixir and understand the basics of the BEAM. If you need a refresher on the Elixir programming language or are new to it, we would recommend taking a look at the Elixir Getting Started Guide (https://elixir-lang.org/getting-started/introduction.html) prior to diving into this book.

With that being said, we think we're ready to begin! Without further ado let's dive into the Erlang standard library and learn how we can leverage it even when programming with Elixir.