
Elixir Phoenix Tutorial

Elixir in Action

RESTful Web Services

Ruby on Rails for Microsoft Developers

New Foundations for a New World

Solve Problems Using Evolution

Programming Ecto

Seven Databases in Seven Weeks

Learn to Program

Design Patterns in Ruby (Adobe Reader)

Interactive Elixir Web Programming Without Writing Any JavaScript

Effective Strategies for the Real World

Designing Fine-Grained Systems

Write Less Code, Get More Done (and Have Fun!)

From Concept to Production

Introducing Erlang

Learn the Ultimate Language and Become a Better Programmer

Phoenix in Action

Flexible, Robust Services for Queries, Mutations, and Subscriptions

Programming with C++20

From Journeyman to Master

Programming Elixir \geq 1.6

Build Database Apps in Elixir for Scalability and Performance

Designing Elixir Systems with Otp: Write Highly Scalable, Self-Healing Software with Layers

Concurrent Data Processing in Elixir

Build Large, Maintainable Web Applications Interactively

Web Development with Clojure

Programming Phoenix LiveView

Programming Elixir 1.3

Clojure for the Brave and True

The Little Elixir & OTP Guidebook

Concepts, Coroutines, Ranges, and more

Modern CSS with Tailwind

Productive > Reliable > Fast

Learn Web Development with Rails

Rethink the Modern Web App

Functional |> Concurrent |> Pragmatic |> Fun

Elm in Action
Craft GraphQL APIs in Elixir with Absinthe
Concurrent Programming in ERLANG

Downloaded from
blog.gmercyyu.edu *by*
guest

Elixir Phoenix Tutorial

LYONS SUMMERS

Elixir in Action Pragmatic Bookshelf
From finance to artificial intelligence, genetic algorithms are a powerful tool with a wide array of applications. But you don't need an exotic new language or framework to get started; you can learn about genetic algorithms in a language you're already familiar with. Join us for an in-depth look at the algorithms, techniques, and methods that go into writing a genetic algorithm. From introductory problems to real-world

applications, you'll learn the underlying principles of problem solving using genetic algorithms. Evolutionary algorithms are a unique and often overlooked subset of machine learning and artificial intelligence. Because of this, most of the available resources are outdated or too academic in nature, and none of them are made with Elixir programmers in mind. Start from the ground up with genetic algorithms in a language you are familiar with. Discover the power of genetic algorithms through simple solutions to challenging problems. Use Elixir features to write genetic algorithms that are concise and

idiomatic. Learn the complete life cycle of solving a problem using genetic algorithms. Understand the different techniques and fine-tuning required to solve a wide array of problems. Plan, test, analyze, and visualize your genetic algorithms with real-world applications. Open your eyes to a unique and powerful field - without having to learn a new language or framework. What You Need: You'll need a macOS, Windows, or Linux distribution with an up-to-date Elixir installation.

RESTful Web Services Pragmatic Bookshelf

Data is everywhere - it's just not very well connected, which makes it super hard to relate dataset to dataset. Using graphs as the underlying glue, you can readily join data together and create

navigation paths across diverse sets of data. Add Elixir, with its awesome power of concurrency, and you'll soon be mastering data networks. Learn how different graph models can be accessed and used from within Elixir and how you can build a robust semantics overlay on top of graph data structures. We'll start from the basics and examine the main graph paradigms. Get ready to embrace the world of connected data! Graphs provide an intuitive and highly flexible means for organizing and querying huge amounts of loosely coupled data items. These data networks, or graphs in math speak, are typically stored and queried using graph databases. Elixir, with its noted support for fault tolerance and concurrency, stands out as a language eminently suited to processing sparsely

connected and distributed datasets. Using Elixir and graph-aware packages in the Elixir ecosystem, you'll easily be able to fit your data to graphs and networks, and gain new information insights. Build a testbed app for comparing native graph data with external graph databases. Develop a set of applications under a single umbrella app to drill down into graph structures. Build graph models in Elixir, and query graph databases of various stripes - using Cypher and Gremlin with property graphs and SPARQL with RDF graphs. Transform data from one graph modeling regime to another. Understand why property graphs are especially good at graph traversal problems, while RDF graphs shine at integrating different semantic models and can scale up to

web proportions. Harness the outstanding power of concurrent processing in Elixir to work with distributed graph datasets and manage data at scale. What You Need: To follow along with the book, you should have Elixir 1.10+ installed. The book will guide you through setting up an umbrella application for a graph testbed using a variety of graph databases for which Java SDK 8+ is generally required. Instructions for installing the graph databases are given in an appendix. *Ruby on Rails for Microsoft Developers* "O'Reilly Media, Inc." Learn different ways of writing concurrent code in Elixir and increase your application's performance, without sacrificing scalability or fault-tolerance. Most projects benefit from running

background tasks and processing data concurrently, but the world of OTP and various libraries can be challenging. Which Supervisor and what strategy to use? What about GenServer? Maybe you need back-pressure, but is GenStage, Flow, or Broadway a better choice? You will learn everything you need to know to answer these questions, start building highly concurrent applications in no time, and write code that's not only fast, but also resilient to errors and easy to scale. Whether you are building a high-frequency stock trading application or a consumer web app, you need to know how to leverage concurrency to build applications that are fast and efficient. Elixir and the OTP offer a range of powerful tools, and this guide will show you how to choose the best tool for each

job, and use it effectively to quickly start building highly concurrent applications. Learn about Tasks, supervision trees, and the different types of Supervisors available to you. Understand why processes and process linking are the building blocks of concurrency in Elixir. Get comfortable with the OTP and use the GenServer behaviour to maintain process state for long-running jobs. Easily scale the number of running processes using the Registry. Handle large volumes of data and traffic spikes with GenStage, using back-pressure to your advantage. Create your first multi-stage data processing pipeline using producer, consumer, and producer-consumer stages. Process large collections with Flow, using MapReduce and more in parallel. Thanks to

Broadway, you will see how easy it is to integrate with popular message broker systems, or even existing GenStage producers. Start building the high-performance and fault-tolerant applications Elixir is famous for today.

What You Need: You'll need Elixir 1.9+ and Erlang/OTP 22+ installed on a Mac OS X, Linux, or Windows machine.

New Foundations for a New World
Pragmatic Bookshelf

For weeks, months—nay!—from the very moment you were born, you've felt it calling to you. At long last you'll be united with the programming language you've been longing for: Clojure! As a Lisp-style functional programming language, Clojure lets you write robust and elegant code, and because it runs on the Java Virtual Machine, you can

take advantage of the vast Java ecosystem. Clojure for the Brave and True offers a "dessert-first" approach: you'll start playing with real programs immediately, as you steadily acclimate to the abstract but powerful features of Lisp and functional programming. Inside you'll find an offbeat, practical guide to Clojure, filled with quirky sample programs that catch cheese thieves and track glittery vampires. Learn how to:

- Wield Clojure's core functions
- Use Emacs for Clojure development
- Write macros to modify Clojure itself
- Use Clojure's tools to simplify concurrency and parallel programming

Clojure for the Brave and True assumes no prior experience with Clojure, the Java Virtual Machine, or functional programming. Are you ready, brave reader, to meet your

true destiny? Grab your best pair of parentheses—you're about to embark on an epic journey into the world of Clojure! Simon and Schuster

Annotation Over the past 10 years, distributed systems have become more fine-grained. From the large multi-million line long monolithic applications, we are now seeing the benefits of smaller self-contained services. Rather than heavy-weight, hard to change Service Oriented Architectures, we are now seeing systems consisting of collaborating microservices. Easier to change, deploy, and if required retire, organizations which are in the right position to take advantage of them are yielding significant benefits. This book takes an holistic view of the things you need to be cognizant of in order to pull this off. It

covers just enough understanding of technology, architecture, operations and organization to show you how to move towards finer-grained systems.

Solve Problems Using Evolution Addison-Wesley Professional

A complete description of Erlang, a programming language for building robust concurrent systems. The book contains many examples of how robust real-time systems can be programmed using this language.

Programming Ecto Programming Phoenix = 1.4 Productive > Reliable > Fast
Your domain is rich and interconnected, and your API should be too. Upgrade your web API to GraphQL, leveraging its flexible queries to empower your users, and its declarative structure to simplify your code. Absinthe is the GraphQL

toolkit for Elixir, a functional programming language designed to enable massive concurrency atop robust application architectures. Written by the creators of Absinthe, this book will help you take full advantage of these two groundbreaking technologies. Build your own flexible, high-performance APIs using step-by-step guidance and expert advice you won't find anywhere else. GraphQL is a new way of structuring and building web services, and the result is transformational. Find out how to offer a more tailored, cohesive experience to your users, easily aggregate data from different data sources, and improve your back end's maintainability with Absinthe's declarative approach to defining how your API works. Build a GraphQL-based API from scratch using

Absinthe, starting from core principles. Learn the type system and how to expand your schema to suit your application's needs. Discover a growing ecosystem of tools and utilities to understand, debug, and document your API. Take it to production, but do it safely with solid best practices in mind. Find out how complexity analysis and persisted queries can let you support your users flexibly, but responsibly too. Along the way, discover how Elixir makes all the difference for a high performance, fault-tolerant API. Use asynchronous and batching execution, or write your own custom add-ons to extend Absinthe. Go live with subscriptions, delivering data over websockets on top of Elixir (and Erlang/OTP's) famous solid performance

and real-time capabilities. Transform your applications with the powerful combination of Elixir and GraphQL, using Absinthe. What You Need: To follow along with the book, you should have Erlang/OTP 19+ and Elixir 1.4+ installed. The book will guide you through setting up a new Phoenix application using Absinthe.

[Seven Databases in Seven Weeks](#) John Wiley & Sons

"Every developer working with the Web needs to read this book." -- David Heinemeier Hansson, creator of the Rails framework "RESTful Web Services finally provides a practical roadmap for constructing services that embrace the Web, instead of trying to route around it." -- Adam Trachtenberg, PHP author and EBay Web Services Evangelist

You've built web sites that can be used by humans. But can you also build web sites that are usable by machines? That's where the future lies, and that's what RESTful Web Services shows you how to do. The World Wide Web is the most popular distributed application in history, and Web services and mashups have turned it into a powerful distributed computing platform. But today's web service technologies have lost sight of the simplicity that made the Web successful. They don't work like the Web, and they're missing out on its advantages. This book puts the "Web" back into web services. It shows how you can connect to the programmable web with the technologies you already use every day. The key is REST, the architectural style that drives the Web.

This book: Emphasizes the power of basic Web technologies -- the HTTP application protocol, the URI naming standard, and the XML markup language Introduces the Resource-Oriented Architecture (ROA), a common-sense set of rules for designing RESTful web services Shows how a RESTful design is simpler, more versatile, and more scalable than a design based on Remote Procedure Calls (RPC) Includes real-world examples of RESTful web services, like Amazon's Simple Storage Service and the Atom Publishing Protocol Discusses web service clients for popular programming languages Shows how to implement RESTful services in three popular frameworks -- Ruby on Rails, Restlet (for Java), and Django (for Python) Focuses on practical issues: how

to design and implement RESTful web services and clients This is the first book that applies the REST design philosophy to real web services. It sets down the best practices you need to make your design a success, and the techniques you need to turn your design into working code. You can harness the power of the Web for programmable applications: you just have to work with the Web instead of against it. This book shows you how.

Learn to Program Simon and Schuster Write code that writes code with Elixir macros. Macros make metaprogramming possible and define the language itself. In this book, you'll learn how to use macros to extend the language with fast, maintainable code and share functionality in ways you never thought

possible. You'll discover how to extend Elixir with your own first-class features, optimize performance, and create domain-specific languages.

Metaprogramming is one of Elixir's greatest features. Maybe you've played with the basics or written a few macros. Now you want to take it to the next level. This book is a guided series of metaprogramming tutorials that take you step by step to metaprogramming mastery. You'll extend Elixir with powerful features and write faster, more maintainable programs in ways unmatched by other languages. You'll start with the basics of Elixir's metaprogramming system and find out how macros interact with Elixir's abstract format. Then you'll extend Elixir with your own first-class features, write a

testing framework, and discover how Elixir treats source code as building blocks, rather than rote lines of instructions. You'll continue your journey by using advanced code generation to create essential libraries in strikingly few lines of code. Finally, you'll create domain-specific languages and learn when and where to apply your skills effectively. When you're done, you will have mastered metaprogramming, gained insights into Elixir's internals, and have the confidence to leverage macros to their full potential in your own projects.

Design Patterns in Ruby (Adobe Reader) Addison-Wesley Professional Languages may come and go, but the relational database endures. Learn how to use Ecto, the premier database library

for Elixir, to connect your Elixir and Phoenix apps to databases. Get a firm handle on Ecto fundamentals with a module-by-module tour of the critical parts of Ecto. Then move on to more advanced topics and advice on best practices with a series of recipes that provide clear, step-by-step instructions on scenarios commonly encountered by app developers. Co-authored by the creator of Ecto, this title provides all the essentials you need to use Ecto effectively. Elixir and Phoenix are taking the application development world by storm, and Ecto, the database library that ships with Phoenix, is going right along with them. There are plenty of examples that show you the basics, but to use Ecto to its full potential, you need to learn the library from the ground up.

This definitive guide starts with a tour of the core features of Ecto - repos, queries, schemas, changesets, transactions - gradually building your knowledge with tasks of ever-increasing complexity. Along the way, you'll be learning by doing - a sample application handles all the boilerplate so you can focus on getting Ecto into your fingers. Build on that core knowledge with a series of recipes featuring more advanced topics. Change your pooling strategy to maximize your database's efficiency. Use nested associations to handle complex table relationships. Add streams to handle large result sets with ease. Based on questions from Ecto users, these recipes cover the most common situations developers run into. Whether you're new to Ecto, or already

have an app in production, this title will give you a deeper understanding of how Ecto works, and help make your database code cleaner and more efficient. What You Need: To follow along with the book, you should have Erlang/OTP 19+ and Elixir 1.4+ installed. The book will guide you through setting up a sample application that integrates Ecto.

Interactive Elixir Web Programming Without Writing Any JavaScript

Pragmatic Bookshelf

Do you have a nagging feeling that your monitoring needs improvement, but you just aren't sure where to start or how to do it? Are you plagued by constant, meaningless alerts? Does your monitoring system routinely miss real problems? This is the book for you. Mike

Julian lays out a practical approach to designing and implementing effective monitoring—from your enterprise application down to the hardware in a datacenter, and everything between. Practical Monitoring provides you with straightforward strategies and tactics for designing and implementing a strong monitoring foundation for your company. This book takes a unique vendor-neutral approach to monitoring. Rather than discuss how to implement specific tools, Mike teaches the principles and underlying mechanics behind monitoring so you can implement the lessons in any tool. Practical Monitoring covers essential topics including: Monitoring antipatterns Principles of monitoring design How to build an effective on-call rotation Getting metrics and logs out of

your application

Effective Strategies for the Real World
Pragmatic Bookshelf

Explore functional programming without the academic overtones (tell me about monads just one more time). Create concurrent applications, but get them right without all the locking and consistency headaches. Meet Elixir, a modern, functional, concurrent language built on the rock-solid Erlang VM. Elixir's pragmatic syntax and built-in support for metaprogramming will make you productive and keep you interested for the long haul. Maybe the time is right for the Next Big Thing. Maybe it's Elixir. This book is the introduction to Elixir for experienced programmers, completely updated for Elixir 1.3. Functional programming techniques help you

manage the complexities of today's real-world, concurrent systems; maximize uptime; and manage security. Enter Elixir, with its modern, Ruby-like, extendable syntax, compile and runtime evaluation, hygienic macro system, and more. But, just as importantly, Elixir brings a sense of enjoyment to parallel, functional programming. Your applications become fun to work with, and the language encourages you to experiment. Part 1 covers the basics of writing sequential Elixir programs. We'll look at the language, the tools, and the conventions. Part 2 uses these skills to start writing concurrent code-applications that use all the cores on your machine, or all the machines on your network! And we do it both with and without OTP. Part 3 looks at the

more advanced features of the language, from DSLs and code generation to extending the syntax. This edition is fully updated with all the new features of Elixir 1.3, with a new chapter on Tooling, covering testing (both conventional and property based), code and dependency exploration, and server monitoring. By the end of this book, you'll understand Elixir, and know how to apply it to solve your complex, modern problems.

Designing Fine-Grained Systems

Pragmatic Bookshelf

The days of the traditional request-response web application are long gone,

but you don't have to wade through oceans of JavaScript to build the interactive applications today's users crave. The innovative Phoenix LiveView library empowers you to build applications that are fast and highly interactive, without sacrificing reliability. This definitive guide to LiveView isn't a reference manual. Learn to think in LiveView. Write your code layer by layer, the way the experts do. Explore techniques with experienced teachers to get the best possible performance. Instead of settling for traditional manuals and tutorials, get insights that can only be learned from experience. Start with the Elixir language techniques that effortlessly marry your client templates and server-side handlers. Design your systems with the right

layers in the right places so that your code is easier to understand, change, and support. Explore features like multi-part uploads and learn how to comprehensively test your live views. Roll into advanced techniques to tie your code to other services through the powerful publish-subscribe interface. LiveView brings the most important programming techniques from the popular Elm and JavaScript React frameworks to Elixir. You'll experience firsthand how to harness that power by working side by side with some of the first LiveView users. You will write your programs to change data on the server, and you'll see how LiveView efficiently detects those changes and reflects them on the web page. Start from scratch, use built-in generators, and craft reusable

components. Your single-purpose reducers will transform server data that your renderers can turn into efficient client-side diffs. Don't settle for knowing how things work. To get the most out of LiveView, you need to know why they work that way. Co-authored by one of the most prolific authors and teachers in all of Elixir, this book is your perfect guide to one of the most important new frameworks of our generation. What You Need: Programming Phoenix LiveView uses Phoenix version 1.5, and any Elixir version compatible with it. You will also want PostgreSQL and JavaScript Node. [Write Less Code, Get More Done \(and Have Fun!\)](#) Pragmatic Bookshelf This is the eBook of the printed book and may not include any media, website access codes, or print supplements that

may come packaged with the bound book. Used by sites as varied as Twitter, GitHub, Disney, and Airbnb, Ruby on Rails is one of the most popular frameworks for developing web applications, but it can be challenging to learn and use. Whether you're new to web development or new only to Rails, *Ruby on Rails™ Tutorial, Fourth Edition*, is the solution. Best-selling author and leading Rails developer Michael Hartl teaches Rails by guiding you through the development of three example applications of increasing sophistication. The tutorial's examples focus on the general principles of web development needed for virtually any kind of website. The updates to this edition include full compatibility with Rails 5, a division of the largest chapters into more

manageable units, and a huge number of new exercises interspersed in each chapter for maximum reinforcement of the material. This indispensable guide provides integrated tutorials not only for Rails, but also for the essential Ruby, HTML, CSS, and SQL skills you need when developing web applications. Hartl explains how each new technique solves a real-world problem, and then he demonstrates it with bite-sized code that's simple enough to understand, yet novel enough to be useful. Whatever your previous web development experience, this book will guide you to true Rails mastery. This book will help you install and set up your Rails development environment, including pre-installed integrated development environment (IDE) in the cloud Go

beyond generated code to truly understand how to build Rails applications from scratch Learn testing and test-driven development (TDD) Effectively use the Model-View-Controller (MVC) pattern Structure applications using the REST architecture Build static pages and transform them into dynamic ones Master the Ruby programming skills all Rails developers need Create high-quality site layouts and data models Implement registration and authentication systems, including validation and secure passwords Update, display, and delete users Upload images in production using a cloud storage service Implement account activation and password reset, including sending email with Rails Add social features and microblogging, including an introduction

to Ajax Record version changes with Git and create a secure remote repository at Bitbucket Deploy your applications early and often with Heroku

From Concept to Production Fertig Publications

Programming with C++20 teaches programmers with C++ experience the new features of C++20 and how to apply them. It does so by assuming C++11 knowledge. Elements of the standards between C++11 and C++20 will be briefly introduced, if necessary.

However, the focus is on teaching the features of C++20. You will start with learning about the so-called big four Concepts, Coroutines, `std::ranges`, and modules. The big four are followed by smaller yet not less important features. You will learn about `std::format`, the new

way to format a string in C++. In chapter 6, you will learn about a new operator, the so-called spaceship operator, which makes you write less code. You then will look at various improvements of the language, ensuring more consistency and reducing surprises. You will learn how lambdas improved in C++20 and what new elements you can now pass as non-type template parameters. Your next stop is the improvements to the STL. Of course, you will not end this book without learning about what happened in the constexpr-world.

Introducing Erlang Pragmatic Bookshelf

Property-based testing helps you create better, more solid tests with little code. By using the PropEr framework in both

Erlang and Elixir, this book teaches you how to automatically generate test cases, test stateful programs, and change how you design your software for more principled and reliable approaches. You will be able to better explore the problem space, validate the assumptions you make when coming up with program behavior, and expose unexpected weaknesses in your design. PropEr will even show you how to reproduce the bugs it found. With this book, you will be writing efficient property-based tests in no time. Most tests only demonstrate that the code behaves how the developer expected it to behave, and therefore carry the same blind spots as their authors when special conditions or edge cases show up. Learn how to see things differently with property tests

written in PropEr. Start with the basics of property tests, such as writing stateless properties, and using the default generators to generate test cases automatically. More importantly, learn how to think in properties. Improve your properties, write custom data generators, and discover what your code can or cannot do. Learn when to use property tests and when to stick with example tests with real-world sample projects. Explore various testing approaches to find the one that's best for your code. Shrink failing test cases to their simpler expression to highlight exactly what breaks in your code, and generate highly relevant data through targeted properties. Uncover the trickiest bugs you can think of with nearly no code at all with two special

types of properties based on state transitions and finite state machines. Write Erlang and Elixir properties that generate the most effective tests you'll see, whether they are unit tests or complex integration and system tests. What You Need Basic knowledge of Erlang, optionally Elixir For Erlang tests: Erlang/OTP \geq 20.0, with Rebar \geq 3.4.0 For Elixir tests: Erlang/OTP \geq 20.0, Elixir \geq 1.5.0 *Learn the Ultimate Language and Become a Better Programmer* Pragmatic Bookshelf Elixir's straightforward syntax and this guided tour give you a clean, simple path to learn modern functional programming techniques. No previous functional programming experience required! This book walks you through

the right concepts at the right pace, as you explore immutable values and explicit data transformation, functions, modules, recursive functions, pattern matching, high-order functions, polymorphism, and failure handling, all while avoiding side effects. Don't board the Elixir train with an imperative mindset! To get the most out of functional languages, you need to think functionally. This book will get you there. Functional programming offers useful techniques for building maintainable and scalable software that solves today's difficult problems. The demand for software written in this way is increasing - you don't want to miss out. In this book, you'll not only learn Elixir and its features, you'll also learn the mindset required to program functionally. Elixir's

clean syntax is excellent for exploring the critical skills of using functions and concurrency. Start with the basic techniques of the functional way: working with immutable data, transforming data in discrete steps, and avoiding side effects. Next, take a deep look at values, expressions, functions, and modules. Then extend your programming with pattern matching and flow control with case, if, cond, and functions. Use recursive functions to create iterations. Work with data types such as lists, tuples, and maps. Improve code reusability and readability with Elixir's most common high-order functions. Explore how to use lazy computation with streams, design your data, and take advantage of polymorphism with protocols. Combine

functions and handle failures in a maintainable way using Elixir features and libraries. Learn techniques that matter to make code that lives harmoniously with the language. What You Need: You'll need a computer and Elixir 1.4 or newer version installed. No previous functional programming or Elixir experience is required. Some experience with any programming language is recommended.

Phoenix in Action Hal Leonard
If you're new to Erlang, its functional style can seem difficult, but with help from this hands-on introduction, you'll scale the learning curve and discover how enjoyable, powerful, and fun this language can be. In this updated second edition, author Simon St.Laurent shows you how to write simple Erlang programs

by teaching you one skill at a time. You'll learn about pattern matching, recursion, message passing, process-oriented programming, and establishing pathways for data rather than telling it where to go. By the end of your journey, you'll understand why Erlang is ideal for concurrency and resilience. Get cozy with Erlang's shell, its command line interface Define functions, using the fun tool, to represent repeated calculations Discover atoms, pattern matching, and guards: the foundations of your program structure Delve into the heart of Erlang processing with recursion, strings, lists, and higher-order functions Create processes, send messages among them, and apply pattern matching to incoming messages Store and manipulate structured data with Erlang Term

Storage and the Mnesia database Learn about Open Telecom Platform, Erlang's open source libraries and tools [Flexible, Robust Services for Queries, Mutations, and Subscriptions](#) Pragmatic Bookshelf

Elixir and Phoenix are generating tremendous excitement as an unbeatable platform for building modern web applications. For decades OTP has helped developers create incredibly robust, scalable applications with unparalleled uptime. Make the most of them as you build a stateful web app with Elixir, OTP, and Phoenix. Model domain entities without an ORM or a database. Manage server state and keep your code clean with OTP Behaviours. Layer on a Phoenix web interface without coupling it to the business logic.

Open doors to powerful new techniques that will get you thinking about web development in fundamentally new ways. Elixir and OTP provide exceptional tools to build rock-solid back-end applications that scale. In this book, you'll build a web application in a radically different way, with a back end that holds application state. You'll use persistent Phoenix Channel connections instead of HTTP's request-response, and create the full application in distinct, decoupled layers. In Part 1, start by building the business logic as a separate application, without Phoenix. Model the application domain with Elixir functions and simple data structures. By keeping state in memory instead of a database, you can reduce latency and simplify your code. In Part 2, add in the GenServer

Behaviour to make managing in-memory state a breeze. Create a supervision tree to boost fault tolerance while separating error handling from business logic.

Phoenix is a modern web framework you can layer on top of business logic while keeping the two completely decoupled. In Part 3, you'll do exactly that as you build a web interface with Phoenix. Bring in the application from Part 2 as a dependency to a new Phoenix project. Then use ultra-scalable Phoenix Channels to establish persistent connections between the stateful server and a stateful front-end client. You're going to love this way of building web apps! What You Need: You'll need a computer that can run Elixir version 1.5 or higher and Phoenix 1.3 or higher. Some familiarity with Elixir and Phoenix

is recommended.

Programming with C++20 "O'Reilly Media, Inc."

Summary Elm is more than just a cutting-edge programming language, it's a chance to upgrade the way you think about building web applications. Once you get comfortable with Elm's refreshingly different approach to application development, you'll be working with a clean syntax, dependable libraries, and a delightful compiler that essentially eliminates runtime exceptions. Elm compiles to JavaScript, so your code runs in any browser, and Elm's best-in-class rendering speed will knock your socks off. Let's get started! Purchase of the print book includes a free eBook in PDF, Kindle, and ePub formats from Manning Publications.

About the technology Simply put, the Elm programming language transforms the way you think about frontend web development. Elm's legendary compiler is an incredible assistant, giving you the precise and user-friendly support you need to work efficiently. Elm applications have small bundle sizes that run faster than JavaScript frameworks and are famously easy to maintain as they grow. The catch? Elm isn't JavaScript, so you'll have some new skills to learn. About the book Elm in Action teaches you the Elm language along with a new approach to coding frontend applications. Chapter by chapter, you'll create a full-featured photo-browsing app, learning as you go about Elm's modular architecture, Elm testing, and how to work seamlessly with your favorite JavaScript libraries. You'll

especially appreciate author and Elm core team member Richard Feldman's unique insights, based on his thousands of hours writing production code in Elm. When you're done, you'll have a toolbox of new development skills and a stunning web app for your portfolio. What's inside Scalable design for production web applications Single-page applications in Elm Data modeling in Elm Accessing JavaScript from Elm About the reader For web developers with no prior experience in Elm or functional programming. About the author Richard Feldman is a software engineer at NoRedInk and a well-known member of the Elm community. Table of Contents PART 1 - GETTING STARTED 1. Welcome to Elm 2. Your first Elm application 3. Compiler as assistant PART 2 -

PRODUCTION-GRADE ELM 4. Talking to servers 5. Talking to JavaScript 6.

Testing PART 3 - BUILDING BIGGER 7. Data modeling 8. Single-page applications

Related with Elixir Phoenix Tutorial:

- Bible Highlighting Guide Pdf : [click here](#)