Extracted from:

Build a Weather Station with Elixir and Nerves

Visualize Your Sensor Data with Phoenix and Grafana

The Pragmatic Bookshelf

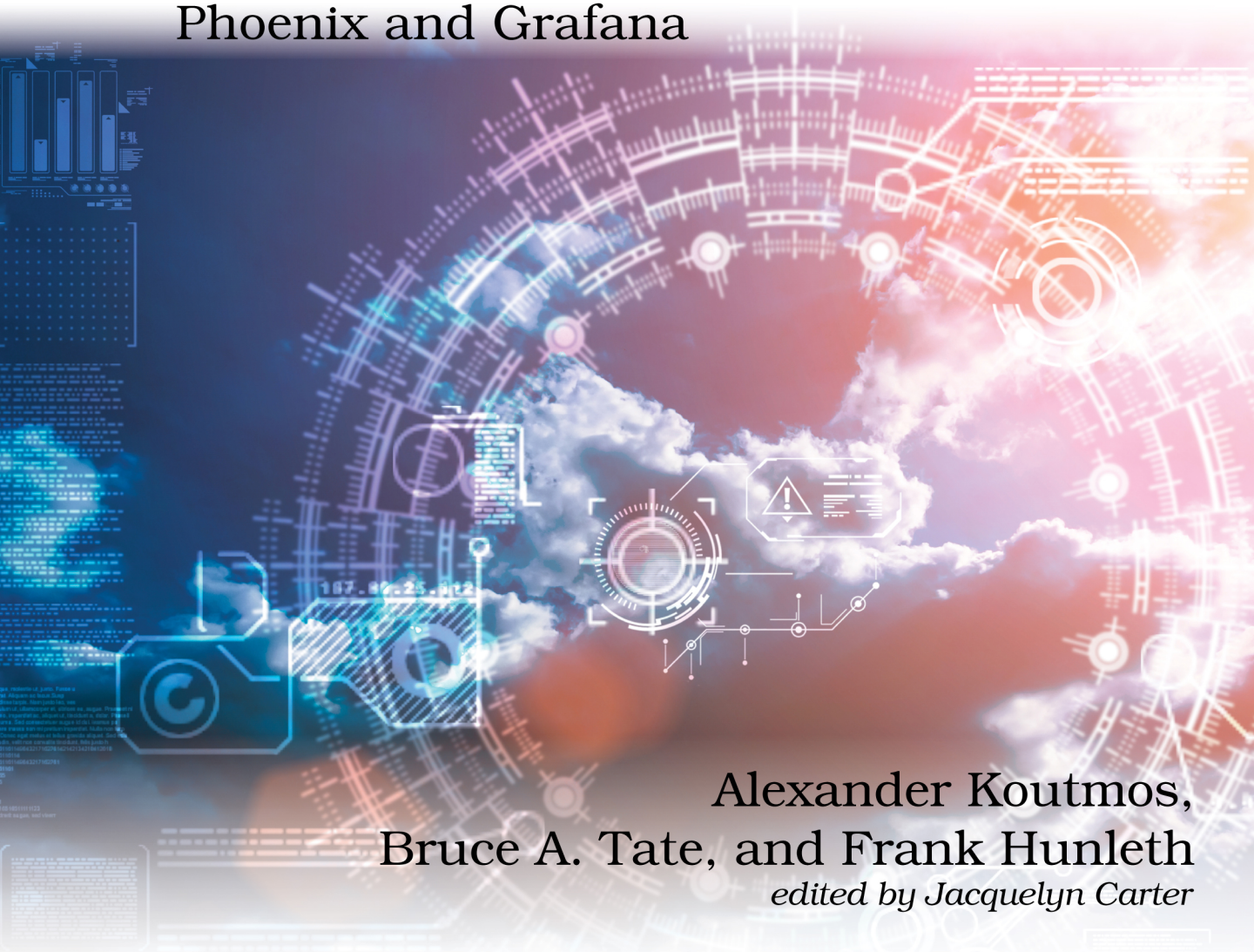
Raleigh, North Carolina

The Pragmatic Programmers

Pragmatic exPress

Build a Weather Station with Elixir and Nerves

Visualize Your Sensor Data with Phoenix and Grafana



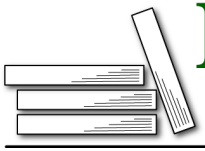
Alexander Koutmos,
Bruce A. Tate, and Frank Hunleth
edited by Jacquelyn Carter

Build a Weather Station with Elixir and Nerves

Visualize Your Sensor Data with Phoenix and Grafana

Alexander Koutmos
Bruce A. Tate
Frank Hunleth

The Pragmatic Bookshelf
Raleigh, North Carolina



Pragmatic Bookshelf

For our complete catalog of hands-on, practical, and Pragmatic content for software developers, please visit *https://pragprog.com*.

The team that produced this book includes:

CEO: Dave Rankin
COO: Janet Furlow
Managing Editor: Tammy Coron
Development Editor: Jacquelyn Carter
Copy Editor: L. Sakhi MacMillan
Layout: Gilson Graphics
Founders: Andy Hunt and Dave Thomas

For sales, volume licensing, and support, please contact *support@pragprog.com*.

For international rights, please contact *rights@pragprog.com*.

ISBN-13: 978-1-68050-902-1
Encoded using the finest acid-free high-entropy binary digits.
Book version: P1.0—January 2022

Introduction

The Erlang virtual machine and the Elixir programming language have managed to find their way into a wide range of software domains. We believe that this is very much in part due to the technical merits of the BEAM runtime and the phenomenal developer experience that the Elixir programming language provides. With both of these elements at your disposal, you truly feel like you have superpowers when programming with Elixir.

This feeling of having "superpowers" is immediately apparent if you've ever written, maintained, or deployed a Phoenix application.[1] The fault tolerance, scalability, and ease of development that you get from the Phoenix Framework and Elixir is a thing of beauty. Not to mention that if your requirements include some sort of real-time user interaction, the tooling available to you in the Elixir ecosystem is second to none.

While Elixir has enjoyed great success in the API back end and web-application space (thanks to Phoenix LiveView[2]), what if you could realize these same "superpowers" in an embedded systems and IoT context? That is exactly what the Nerves Project aims to do.[3] With Nerves, you can develop reliable and fault-tolerant IoT applications without compromising productivity. Nerves takes care of all of the lower-level concerns, leaving you with only the application-specific concerns. In addition, since you have the power of the Erlang virtual machine at your fingertips, you can create concurrent real-time applications leveraging all of the abstractions that you already use.

As you'll see in this short book, Nerves allows you to create very capable IoT applications in record time, without the pain and frustration of building everything from scratch. In less than 100 pages you'll have an end-to-end solution for capturing and visualizing weather data, from the embedded Nerves application capturing and publishing data to the Phoenix application that

1. https://phoenixframework.org/
2. https://hex.pm/packages/phoenix_live_view
3. https://www.nerves-project.org/

receives and persists the data. You can have a full stack Elixir solution all the way from the embedded-hardware level without even breaking a sweat!

What You Will Build

To get hands-on experience with Nerves, you'll be building a simple yet powerful weather station. This book starts at the hardware layer first and focuses on getting started with Nerves. After you get your Nerves-powered device up and running, you'll start writing the code necessary to retrieve sensor data from your weather station sensors. After you get your Nerves application code to a point where it needs to publish its collected data, you'll shift your focus to a lightweight Phoenix JSON API. The Phoenix JSON API will store the captured weather data into TimescaleDB for efficient time-series querying.[4] Once the Phoenix API is up and running, you'll switch back to the Nerves application code and write a lightweight HTTP-based data publisher that will push environmental metrics to your Phoenix back end. With all the weather data eventually finding its way into TimescaleDB, you'll then leverage Grafana to visualize all of the time-series data.[5]

How to Read This Book

This books takes you step by step through the process of building an end-to-end IoT weather station—from data collection, data persistence, data visualization, and everything in-between. As such, it's strongly recommended that you read this book cover to cover, as omitting steps may result in a nonfunctional end product.

Who This Book Is For

This book is for any Elixir programmer that is comfortable with the basics of the programming language and is interested in dabbling in the world of embedded systems. No soldering or deep hardware experience is necessary, given that you'll be working with off-the-shelf plugin-and-play hardware.

Who This Book Isn't For

If you're just starting off with Elixir (welcome, by the way!) or struggle with concepts such as GenServers and pattern matching, we highly suggest picking up a more beginner-oriented Elixir book prior to starting this one. Luckily there are some great resources out there such as Programming Elixir 1.6.[6]

4. https://www.timescale.com/
5. https://grafana.com/
6. https://pragprog.com/titles/elixir16/programming-elixir-1-6/

While Elixir 1.6 came out a few years ago now, the core language hasn't changed much in that time, so the book will help you develop a solid Elixir foundation. After you read that book, feel free to pick this one up again and get your hands dirty with an IoT-based project.

Running the Code Exercises

Being able to build and run your application code will be key to understanding the concepts outlined in this book. As such, it's important that you have the items outlined in the next couple sections so that you have everything you need to complete the weather station project.

Software Requirements

Embedded hardware aside, you'll need the following:

- Elixir version 1.9 or greater
- A Linux, MacOS, or Windows machine to do your development on
- A Linux, MacOS, or Windows machine to run your Phoenix API Server
- A wireless access point for your local area network
- The ability to run Docker containers

If you have all of those items, then you are good to go from a development machine perspective, and all that's needed is the Nerves-related hardware.

Hardware Requirements

While there is some flexibility with what hardware (like what version Raspberry Pi) you can buy and from where, the following items were used by the authors:

- Raspberry Pi Zero W with headers
- Qwiic pHAT v2.0 for Raspberry Pi
- VEML6030 Light Sensor (Qwiic connection)
- BME680 Environmental Sensor (Qwiic connection)
- SGP30 Air Quality Sensor (Qwiic connection)
- Qwiic Connection Cables
- MicroUSB connection cables
- 4GB+ MicroSD card
- MicroSD card reader

If you don't know what these things are or where to buy them, fear not—we explain all of this in the first chapter.

Online Resources

All of the code for this project can be found online at the Pragmatic Programmers web page for this book[7] or in this GitHub repository.[8] If you need any assistance for all things Elixir and Nerves, be sure to check out the Elixir Forums where you'll find a vibrant community ready to help.[9]

7. https://pragprog.com/titles/passweather
8. https://github.com/akoutmos/nerves_weather_station
9. https://elixirforum.com/