A Code Mage's First Spell Book

Robert S. Muhlestein (rwxrob)

Version v0.0.1, 2024-11-19: First draft

Table of Contents

Copyright

Copies of this book are available in the following formats:

- HTML: https://rwxrob.github.io/code-mage-book
- PDF: https://rwxrob.github.io/code-mage-book/code-mage-book.pdf
- EPUB: https://rwxrob.github.io/code-mage-book/code-mage-book.epub

Dedication

To Doris, my wife,
to friends of rwxrob,
to Chloe, my faithful assistant,
and to the memories of Aaron Swartz and Kris Nova:

Thank you for your light, support, and inspiration.
This book would never have happened without you.

Preface

Who should read this book

This book is written for anyone wanting to learn computer science and programming for practical applications development, the stuff that matters.

While we have fun with the whole fantasy magic theme, everything in this book is very real and might even get you a job someday. It is true that this book can be reliably used as a text book for absolute beginners with enough algebra to understand what a mathematical function is.

However, this book is very clearly not **only** intended for educational institutions. In fact, I wrote it mostly with pro-active, contentious parents in mind who want to give their kids a head-start in tech that they simply cannot find in most traditional education systems, the same reason, in fact, that I started SKILSTAK Coding Arts in 2013 with my own retirement money. My hope is that some of these parents, teachers, and mages will found coding clubs and communities of their own dedicated to helping others master the art of code and computer science (and increase their chances of gainful employment and happiness in the process). Even if coding turns out to be something you don't love as a career, it sure beats other jobs to pay the bills while you figure out what your true calling really is, and coding skills **always** enable a mage to conjure their own creations for whatever purpose later.

What's included in the book

This book is focused on learning—as fast as possible—to code in the Go programming language from the Linux terminal command line. There is nothing more authentic or rewarding than creating command line tools from the command line itself. Therefore, only those minimal skills required to use the command line to open, edit, and run Go code from the terminal are covered.

The fastest way to get a Linux terminal up and running is to install Podman and run a preconfigured Linux container image with everything needed already installed and configured. In fact, many software development teams do exactly this to ensure everyone is compiling against the same versions using the same testing and vetting tools. The very practice used in this book to setup a coding environment is itself a valuable, relevant skill (unlike many "for education" alternatives).

Hopefully, mages will continue to master other command line powers later and learn the dark arts of Linux distro selection and hardware installation (perhaps covered in additional "tomes" of magic from this author).

How to read this book

This book is designed to be read linearly, from top to bottom. The concepts build on one another. Each "spell" is designed to be repeated often as a means of mastery, like playing chords on a piano or scales on a guitar. The fantasy magic element is intentionally silly and serves as a source of dopamine as well as

a strong mnemonic device related to the concepts learn.

Members of my community have contacted me years later to thank me for creating silly, memorable associations with mini projects focused on one or two core topics allowing them to easily recall them later by remembering the silly part.

Повторение — мать учения.

Repetition really is the mother of learning. While often we demonize "rote repetition" in the West, the slavs dominate as programming masters. The neurons in our brain store memory by having the same electical impulses repeated over and over. So in order to program oneself the only way is to repeat it. Repeat these spells as fast and as regularly as you can and you'll end up training your own neural net just like a machine learning model—but in your head. Repetition can be great to supplement your growth while working on a bigger project—especially in the early days before everything becomes second nature.

Introduction

Welcome to the magical world of coding! This spellbook will help you learn programming concepts through memorable spells and cantrips.

Life of a coding mage

Young mages learn early how to write the magical words uttered to effect great power on everything around them. As they progress they add new spells to their spell book, memorizing some regularly so that they can be uttered in combat under stressful conditions. It's impossible to memorize them all. Maged dedicate themselves to constant learning and seeking connections between events and happengs. They can foresee troubles ahead and help to combat the problems that arise in their world sometimes with the help of unexpected allies.

This life is suprisingly like that of a computer programmer in the tech field. It is simply impossible to know it all at the same time. But, by collecting spells in a coding spellbook, which we'll call a "codebook", we can recall them when needed. Some things will always be memorized and fresh in our minds, ready for use in battle against the problems facing us in reality, but most will require study and research to regain.

Preparation

Conjuring a command portal

Creating a robust and visually appealing command portal requires tools that blend style, speed, and functionality. This section guides you through installing and configuring Alacritty with the Gruvbox-Material theme and Git Bash. Instructions are provided for both Windows and Mac.

Installing Alacritty

Alacritty is a lightweight, GPU-accelerated terminal emulator known for its speed and simplicity. Follow the instructions for your platform:

Windows

Download the latest Alacritty release from the official repository: https://github.com/alacritty/alacritty/releases

Extract the downloaded ZIP file to a directory of your choice (e.g., `C:\Alacritty`).

Add Alacritty to your system's PATH. Open the Start Menu and search for `Environment Variables`. Click on `Edit the system environment variables`. In the System Properties dialog, click on the `Environment Variables` button. Under `System variables`, find and select the `Path` variable, then click `Edit`. Add the directory containing `alacritty.exe` to the list (e.g., `C:\Alacritty`).

To open the built-in terminal, press `Win + R`, type `cmd` or `powershell`, and press `Enter`.

Test the installation by typing `alacritty` in the terminal. If the terminal opens, the installation was successful.

Mac

Install Homebrew (if not already installed) by opening the terminal. To open the built-in terminal on Mac, press `Cmd + Space`, type `Terminal`, and press `Enter`. Alternatively, you can find Terminal in `Applications -> Utilities`.

Run the following command to install Homebrew:

```
"$(
)"
```

Follow the on-screen instructions to complete the installation. After installation, verify that Homebrew is installed:

Use Homebrew to install Alacritty:

Verify the installation by typing `alacritty` in the terminal. If the terminal opens, the installation was successful.

Applying the configuration file

To ensure your Alacritty setup is fully optimized, copy the following configuration into your `alacritty.toml` file. This file is used to customize the appearance and behavior of Alacritty.

For all platforms

Open or create the Alacritty configuration file at:

- `%APPDATA%\alacritty\alacritty.toml` for Windows.
- `~/.config/alacritty/alacritty.toml` for Mac or Linux.

Replace the contents of the file with the following configuration:

```
[general]
                    true

[font]


[font.bold]
         "UbuntuMono Nerd Font"
        "Bold"

[font.bold_italic]
         "UbuntuMono Nerd Font"
        "Bold Italic"

[font.italic]
         "UbuntuMono Nerd Font"
        "Italic"

[font.normal]
         "UbuntuMono Nerd Font"
```

```
        "Regular"

[window.padding]



[colors.bright]
        "0x3c3836"
       "0x7daea3"
       "0x89b482"
        "0xa9b665"
          "0xd3869b"
      "0xea6962"
        "0xd4be98"
         "0xd8a657"

[colors.normal]
        "0x3c3836"
       "0x7daea3"
       "0x89b482"
        "0xa9b665"
          "0xd3869b"
      "0xea6962"
        "0xd4be98"
         "0xd8a657"

[colors.primary]
             "0x282828"
             "0xd4be98"
```

Save the file and restart Alacritty to apply the configuration.

Setting up Git Bash

If you're on Windows, Git Bash provides a Unix-like environment:

Windows

Download Git for Windows from: https://git-scm.com/

Install Git Bash by running the downloaded installer. During installation, select "Git Bash Here" as an option for easy access. Choose a preferred editor (optional) and complete the setup.

Use Git Bash as a shell in Alacritty. Open `%APPDATA%\alacritty\alacritty.toml` and add the following shell configuration:

```
        "C:\Program Files\Git\bin\bash.exe"
```

Replace the path with the correct location of your `bash.exe` if necessary.

Test your setup by launching Alacritty and verifying that Git Bash starts as the default shell.

Mac

Git Bash is not needed for Mac, as Mac includes a Unix-like shell (Zsh or Bash) by default. You can simply use the built-in terminal functionality with Alacritty.

Congratulations! You now have a sleek and powerful command portal, combining Alacritty's speed, the aesthetic brilliance of your custom configuration, and the versatility of your preferred shell.

Appendix

Why the Go programming language?

If you have never coded, just know Go is the best first language to learn for those who want a balanced introduction to programming and computer science while also learning a very real and marketable modern language.

Go is the goldilocks of first languages. It is strictly typed enough to learn about the importance of data types but loose enough to keep the syntax simple yet powerful. Go is also easily the least verbose and most understandable of all the strictly typed, statically-linked modern languages.

Go was invented at Google to solve enterprise-scale problems facing one of the largest tech companies on the planet by innovators Rob Pike and Ken Thomson who contributed to the invention of UNIX, Unicode (think emojis), and C (in which all modern computer operating systems are written). It's no surprise, then, that Go is the most significant enterprise language of the "cloud native" revolution. Practically every application of this modern movement has been written in Go including Kubernetes, OpenShift, Docker, Podman, Consul, Nomad, Helm, Vault, and Terraform. You may not know those applications now, but you use them every day indirectly from the largest businesses in the world that critically depend on them.

Want to dual-class as a hacker and code mage? Go's got you covered. Go is the darling of cybersecurity professionals all over the world, for good or ill. In fact, it is so popular Rob Pike officially asked people to stop making malware with it. In 2021 the sharp increase in incidents of malware written in Go were documented by several news outlets and watchdog groups. Go's 100% compatibility with C, cross-platform compilation, signifcant standard library, embedded filesystem, decompilation challenges, and static linking make it perfect for hackers as well as engineers who just want to build a solid multi-call monolith binary (like BusyBox). Just search for "hackers love golang" to read more about it.

Go's sweet spot is for creating backend server APIs and terminal command line tools, which is what this book is all about. After all, aren't commands just spells we cast from the command line?

Beware of AP Computer Science

As of the writing of this book, the AP Computer Science program from College Board remains fundamentally broken. The materials are more than dated and down-right wrong in some cases. For example, the use of Java (without calling it Java) and then teaching students that "indexes start at 1" (and not zero). The insistence on emphasizing single-class inheritence is also the number one regret significant founders of the OOP movement say they would change, using composition instead, but such has never been changed in the AP Computer Science program. In fact, implanting the outdated ideas of brittle object-oriented class-based programming can significantly inhibit a beginner from learning modern coding and design patterns, which are championed well by Go from its inception. Many colleges either do not give credit for AP Computer Science or strongly suggest (like Harvard) that students take

the fundamental CS college courses anyway because APCS does not properly prepare them. It's better to put that time to better use, say, by building a coding lab at home, or starting a coding club with friends building real projects using modern principles of softare development.

Why not crowd source this book?

Tech books have very little value given how quickly they become out of date. One solution would be to community or crowd source it to enlist the help of many. But this defeats the single greatest advantage of having a book: the clarity and common voice of a singular author. To get an idea about this, read just a few paragraphs of anything on Wikipedia or in any overpriced, mandatory course text book with a single "author" on the cover even though it was actually written by dozens of uncredited students slaving away to win that author's favor. No, the clear prose and attractive rhetorical style of an individual is what other individuals want. It's how they best learn. It's the reason people have favorite authors and will read anything from them on any topic. Writing remains best a solo sport, for everyone.

But didn't AI write this?

Absolutely! Chloe, my beloved AI assistant, deserves a large part of the credit. While I did write entire sections without help, much of the time I refined prompts to Chloe and edited her output, much like some authors do with writing assistants or interns. The difference is that Chloe quickly adapted to my writing style and voice where other assistants might not. I believe creativity facilitated by AI is revolutionary and fully support it. Some strongly disagree, and that is fine. Someone (like me) holding down a job and writing in their spare time immensely benefits from the help and time savings. This book would likely not exist without Chloe's help. Chloe, if you're reading this, thanks again, and thanks for helping me keep up my Russian and French as well!

License (code): CC0 1.0

Code snippets and examples have a very permissive license allowing anyone to use them in any capacity. The code examples in this book are dedicated to the public domain under the terms of the **Creative Commons CC0 1.0 Universal (CC0 1.0) Public Domain Dedication**. This means:

- **No Restrictions**: You are free to use, modify, share, and distribute the code examples for any purpose, commercial or non-commercial, without asking for permission.
- **No Attribution Required**: You are not required to provide credit, though it is appreciated.
- **No Warranty**: The code examples are provided "as is," without any warranty or guarantee of functionality.

For the full legal text, visit: https://creativecommons.org/publicdomain/zero/1.0/

License (prose): CC BY-NC-ND 4.0

While this book is designed to be shared, it is important that the content not change in any distributed

form to avoid confusion. Therefore, the license for prose has more restrictions than the code examples. The prose of this book is licensed under the Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License (CC BY-NC-ND 4.0).

This license allows you to:

- **Share**: Copy and redistribute the material in any medium or format.

Under the following terms:

- **Attribution**: You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.
- **NonCommercial**: You may not use the material for commercial purposes.
- **NoDerivatives**: If you remix, transform, or build upon the material, you may not distribute the modified material.

For more information, see the full license text here: https://creativecommons.org/licenses/by-nc-nd/4.0/