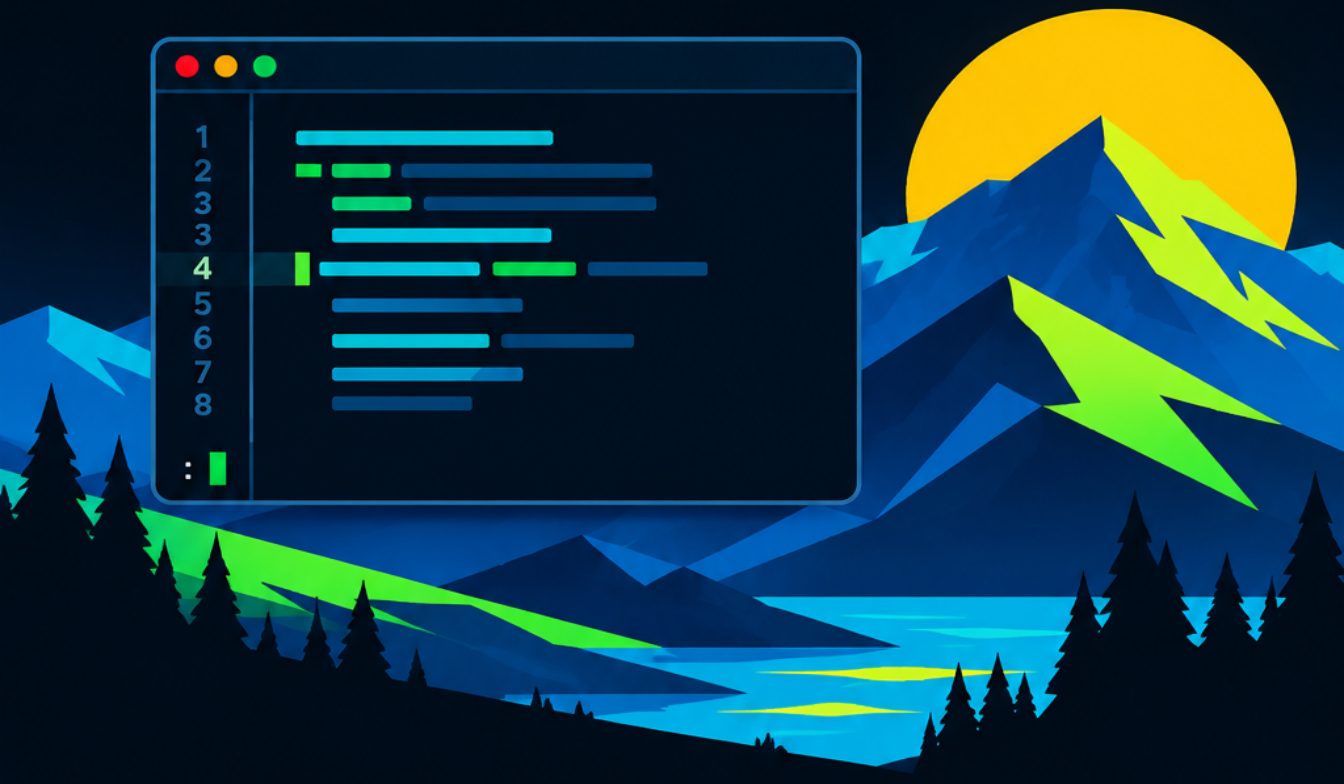
Modern Vim for Developers and Systems Engineers

Modal editing, programmable workflows and the architecture of efficient text work



LUKE CHANDLER

Modern Vim for Developers and Systems Engineers

Modal editing, programmable workflows and the architecture of efficient text work

Steve Publications

This book is available at
https://leanpub.com/modernvimfordevelopersandsystemsengineers

This version was published on 2026-09-21

This is a Leanpub book. Leanpub empowers authors and publishers with the Lean Publishing process. Lean Publishing is the act of publishing an in-progress ebook using lightweight tools and many iterations to get reader feedback, pivot until you have the right book and build traction once you do.

Contents

Modal editing, programmable workflows and the architecture of efficient text work

About this book. Vim is usually sold as a keyboard trick or a rite of passage. This book treats it for what it actually is: a small, composable transformation language, wrapped in an editor, running on every Unix system you will ever touch, with three decades of accumulated design decisions behind it. The goal is to teach you the grammar of modal editing, the architecture of the tool itself and the professional workflows that developers and systems engineers build on top of it, log triage on a live server, Git review, LSP integration, remote work over SSH, batch text processing and the careful management of a plugin ecosystem. It is also honest about what Vim is not: it has real costs, and it is not the right tool for every class of work. By the end, you should be able to decide where Vim earns its keep and use it at a level that pays dividends every day.

Introduction: The Grammar of Text Work

Here is a task that, on a modern IDE, you would probably do with a mouse, a menu, or a find-and-replace dialog: you have a file containing two hundred configuration entries, one per line, and every fourth line begins with the token `retry =` instead of the correct `retries =` . You need to fix all of them, in one pass, without touching the other lines and ideally with a confirmation prompt for each replacement in case the pattern appears somewhere unexpected.

In Vim you type this: `/retry =` , then `:g/^retry = /s/^retry = /retries = /c`, press enter and confirm replacements one at a time, or all at once. Four keystrokes plus a word. No dialog opened, no context lost, no scrollbars fought with. You never had to think about where you were in the file, because the transformation is address-based: "all lines beginning with this pattern."

That sentence is the thesis of this book. Vim's distinctive feature is not that it is keyboard-driven, though it is. It is not that it is fast, though it is. Its distinctive feature is that editing is expressed as a *language*, one in which small commands (operators, motions, text objects, counts, registers, ranges) compose into precise descriptions of transformations. A command like `d2iw` means "delete the inner text of two words." The command `c3w` means "change three words and type the replacement." You are not clicking widgets; you are speaking to the editor in a grammar it has understood since 1991, and that grammar is stable enough that commands from a 1990s man page still work, unchanged, in today's Vim 9.

The promise of this book

After working through it, you should be able to do four things:

1. **Think in Vim.** Navigate files and codebases by searching and jumping rather than scrolling; transform text by composing operators with objects rather than by entering insert mode; repeat any transformation with one keystroke. This is chapters 2 through 9.

2. **Understand the machine.** Know how Vim stores your session (buffers, swap files, undo files, viminfo), how it starts up (the vimrc search order, plugins, options), how it can be scripted (Vimscript, the newer Vim9script) and how it extends itself (packages, autocommands, mappings). This is chapters 10 through 15.
3. **Practice a professional workflow.** Use Vim for Git review, LSP-assisted development, remote work over SSH inside tmux, incident response on servers where a browser is not an option and bulk text processing in pipelines alongside ripgrep and sed. This is chapters 16 through 20.
4. **Make defensible tooling decisions.** Understand exactly where Vim is strong, where it is weaker than an IDE or a Neovim-based stack, what the learning investment buys you and how to keep a configuration healthy for years. This is chapters 21 through 23, plus the conclusion.

Who this book is for

Readers are assumed to be technically fluent: comfortable with a shell, familiar with at least one programming language and at home on Linux or macOS. No prior Vim experience is assumed. The first several chapters are designed so that a newcomer can use Vim productively within a day or two, but they are not beginner coddling. Every feature is explained in terms of *why it exists* and *when it is worth the keystrokes it costs you*, which is how the book stays useful to people who already use Vim and want to stop leaving power on the table.

Systems engineers are addressed explicitly throughout, not as an afterthought. Vim's niche has historically been the machine room: editing configs at 2 a.m., grepping a forty-gigabyte log on a node you cannot afford to reboot, patching a script inside a container with no package manager beyond the distro defaults. Chapter 18 and chapter 19 build on that reality.

How the book is organized

Part One (chapters 1 through 9) covers the editing language itself, from history and philosophy through navigation, operators, text objects, registers, search and substitution, macros and the Ex command line. These chapters contain the bulk of what "Vim" means to a working user, and they deliberately spend

their space on the reasoning behind techniques rather than on exhaustive key lists.

Part Two (chapters 10 through 15) covers the tool's interior: undo, recovery, large-file handling, the Vim 9 feature set and its scripting language, configuration architecture, autocommands and functions, packages and the hygiene practices that keep an ecosystem maintainable.

Part Three (chapters 16 through 20) covers working software professionally in Vim: language server integration, Git and diffing, terminal and remote development, systems workflows and automation.

Part Four (chapters 21 through 23) steps back for honest comparisons with IDEs, productivity and habit formation and the anti-patterns that quietly ruin configurations. A quick-reference appendix collects the most important commands, and a reference list gathers all cited sources.

What Vim is and is not

Three positions this book will not take, for clarity:

Vim is not a universal replacement for IDEs. It does not give you integrated test runners, debuggers, or large-scale semantic refactoring out of the box, and even with the best plugins its language tooling lags Neovim's native stack. If your daily work is deep inside one large codebase with one language and one toolchain, an IDE is probably a better home, and chapter 21 will tell you when.

Vim is not a free lunch either. Modal editing is a genuine investment, often weeks of deliberate practice before the payoff becomes obvious, and that investment is wasted on people whose workflows rarely touch it.

And Vim is not finished evolving in any way that guarantees the current ecosystem is the right one to adopt wholesale. The Vim project is healthy but modest; the center of gravity for modern editor development has shifted to Neovim. Chapter 21 addresses that honestly.

What Vim is, is a language for describing text transformations, executed by a tool that ships on virtually every Unix system ever built. Learn the language and the tool becomes invisible: the same reflexes work in your laptop terminal, over SSH to a production node, inside a stripped container and in a CI pipeline. That portability, combined with composable precision, is why the editor still

matters in 2026, and why learning its grammar is a durable investment rather than a fashion.

Chapter 1: Origins and Philosophy

From ed to vi

Modern Vim is easy to mistake for a product of the graphical era, a keyboard purist's reaction to bloated software. In reality it is the continuation of a lineage that began on teletypes, before display terminals were common. The earliest Unix text editor was `ed`, written by Ken Thompson at Bell Labs in 1971. It was a line editor: no screen to speak of, just numbered lines and commands sent through a pipe. [23]

That limitation shaped the design of everything that followed. On a line editor, you cannot select "the paragraph containing the cursor" by dragging anything anywhere. You describe the selection. You issue commands like "print lines 10 through 15" or "append text after line 10." The mental model that survives in Vim today, addressing text by line numbers, by patterns and by ranges, is not a quirk; it is a direct inheritance from `ed`. [22,23]

By the mid-1970s, Berkeley needed something better than `ed` for their Unix port. George Coulouris wrote `em`, "editor for mortals", and Bill Joy and colleagues refined it into `ex`, "extended." Most users of `ex` spent their time in a mode that displayed the screen and let you move around visually. So Joy created a hard link named `vi`, "visual (in ex)", and released it with Berkeley Unix in 1979. From that moment, an editor's identity was defined less by its features than by its modal interface: command mode, insert mode and the ex-style colon line for batch commands. [22,23]

The BSD Unix world then fragmented and re-standardized repeatedly (System V's `ed`-derived vi, GNU's attempts, OpenBSD's `nvi` by Keith Bostic, derived from the elvis editor of 1990), but the interface itself survived every rewrite essentially intact. That survival matters more than any feature list: it is the strongest evidence in software history that the modal design is a stable attractor. Independent implementers, working from a compatibility requirement and nothing else, kept rebuilding the same keyboard grammar. [23]

Bram Moolenaar and the 1991 fork

In 1988, Bram Moolenaar, a Dutch programmer with no editor for his Amiga, took the public-domain source of STEVIE (an Amiga vi clone from 1987) and began rewriting it, calling the project "Vi IMitation." He released the first version, 1.14, on November 2, 1991, the same year as the first Linux kernel. [20] By version 2.0 in December 1993 he had renamed the project to "Vi IMproved", because imitation had long since ceased to be an accurate description. [19,21,23]

What made Vim different from vi was not one feature but a philosophy: vi compatibility as a floor, not a ceiling. Moolenaar kept the commands working and then built an editor around them. The version history is a record of that ambition:

- Vim 6.0 (2001): plugin architecture, the mechanism that made third-party extension practical. [25]
- Vim 7.0 (2006): persistent undo, undo branches, tab pages, spell checking, omni completion, a built-in file explorer, remote file access, internal grep. [24,26]
- Vim 7.4 (2012): native package management, letting plugins install cleanly under `~/.vim/pack` without third-party runtimepath hacks. [28]
- Vim 8.0 (2016): channels and jobs, the async primitives that made it possible for Vim plugins to run language servers without blocking the UI. [19]
- Vim 9.0 (July 2022): Vim9script, a compiled scripting language; fuzzy matching functions; improved terminal emulation. [3]
- Vim 9.1 (January 2024): classes and objects in the scripting language, smooth scrolling, virtual text; dedicated to Moolenaar, who died on August 3, 2023. [42,43]
- Vim 9.2 (February 2026): enums, generic functions and tuples in Vim9script; a rewritten diff algorithm; Wayland support; XDG Base Directory configuration paths. [1]

Vim also carries a distinctive distribution model: it is charityware, with donations historically directed to ICCF Holland's projects for children in Uganda. Following Moolenaar's death, the project transitioned donations to Kuwasha, a Canadian partner continuing the same project in Kibaale, Uganda. [45] The project remains actively maintained: Christian Brabandt, a contributor

since 2006, leads day-to-day maintenance alongside others, though by most accounts the project is now "more-or-less in maintenance mode", healthy and responsive, but no longer accelerating the way it did under Moolenaar. [1,17,18]

The modal model explained

The core idea of modal editing is that the editor distinguishes *modes of interaction* rather than treating every keystroke the same. In normal mode (the default), `d` means "start a deletion", it does nothing by itself. In insert mode, `d` means "type the letter d." The same key is interpreted according to the current mode, and modes can be entered and exited deliberately.

A newcomer's instinct is usually to ask "why can't it be like normal text, where typing always types?" The answer is structural. Modal editors are built around the observation that most editing work is *navigation and transformation*, and very little is *composition of new content*. In a long day of coding or sysadmin work, the ratio is often 90/10: you move around, delete, reorder, rename, reindent, search. Only the remaining 10 percent is raw typing. A modal editor devotes its best ergonomics to the 90, at the price of an explicit transition for the 10. Non-modal editors devote everything to typing and treat navigation as a series of menu clicks or modifier-key accidents.

There is a second reason, less visible: state. A modal editor knows what you are doing. When you are in normal mode, any key you press that does not begin a recognized command cannot corrupt your text. When you are in insert mode, Vim knows that every character belongs to the file until you explicitly leave, which makes the atomicity of edits (one insert session equals one undo step) reliable. In a non-modal world, every action is ambiguous until you click something; the editor must constantly guess whether a keystroke is text, a shortcut, or a command.

Why modality works (for some people, most of the time)

Three properties follow from the model:

Keyboard locality. Home-row keys do most of the work (`hjkl` for movement), which keeps hands centered and reduces the fumbles that accumulate over tens of thousands of actions a day. Experienced Vim users type faster because they press fewer keys and fewer of them are far away.

Composability. This is the big one, and the theme of the whole book. Because normal-mode commands are short verb-like tokens, they combine: `d` (operator) + `2` (count) + `w` (motion) = "delete two words." `y` + `iw` = "yank inner word." A hundred verbs times a hundred nouns gives ten thousand distinct operations, all reachable with two to four keystrokes from the home row, none of them hidden in menus. This is the "grammar" the introduction promised.

Uniformity across systems. Because vi compatibility was the design floor for decades, the command grammar is portable: the same `dw`, `:%s` and `CTRL-U` work in your laptop's Vim, the busybox vi on an Alpine container and the OpenBSD vi on a network device. No other editor offers that coverage. For a systems engineer, this is not nostalgia; it is the difference between "I can always edit this config" and "I hope this box has my preferred tool installed."

What carried over and what changed

It is worth naming what changed, because much popular Vim lore is actually pre-Vim lore, and Vim lore that is actually from the vi era behaves differently than you might guess.

Carried over from vi, with barely a ripple: the modal model, normal/insert/ex modes, operators with motions and counts, registers (vi had them; Vim's `a` through `z` are vi's design), marks, the jump list, `.` repeat, macros via `q` and most of the regular-expression dialect. If a command appears in a 1985 manual, it almost certainly behaves the same today, which is why old Unix literature remains surprisingly usable. [22]

Genuinely Vim innovations, in order of practical importance for this book: text objects (`d` + `i` + `"`, delete the contents inside quotes), which postdate vi by decades and are among the most valuable additions; the undo tree and persistent undo; syntax highlighting; plugins and packages; tab pages; diff mode (`vimdiff`); the internal grep and quickfix machinery; and finally the scripting languages, Vimscript and then Vim9script, which turn Vim from an editor into a programmable environment. [3,24,26]

One caution for people reading old material: vi's original command set had sharp edges (for example, `cw` behaves like `ce`, changing to the end of the word rather than through it). Vim kept these behaviors for compatibility, and some of them still surprise people. The documentation's `vi_diff` pages exist

precisely to mark where "Vim" and "vi" diverge. The book will flag the important divergences as they come up.

The net effect of four decades of this design philosophy is a tool whose stable surface is enormous and whose volatile surface is comparatively small. That stability is the precondition for everything that follows: you can learn Vim once and carry the knowledge for decades, across operating systems you have never installed, on machines you have never seen, under circumstances where nothing modern is present at all.

Chapter 2: The Modal Mental Model

The four main modes and their submodes

When Vim starts, it is in **normal mode**. Every keystroke here is an editor command, never text. From normal mode you reach:

- **Insert mode** (`i` to insert before the cursor, `a` to append after, `o` to open a new line below and enter insert, `O` for the same above). This is the only mode where typing goes into the file. You leave it with Escape (or `CTRL-[`, which is the same key and the older idiom).
- **Visual mode** (`v` for characterwise selection, `V` for line-wise, `CTRL-V` for block/column-wise). Selections in visual mode can then be operated on by the same operators as motions.
- **Command-line mode** (`:`, `/`, `?`). Typing here is sent to the command interpreter rather than the file.
- **Ex mode** (via `:open` or starting Vim as `ex`), a full-screen command-line environment with no file display at all, useful for scripting and huge files.

Vim's documentation counts fourteen modes in total once you include submodes such as visual block, select mode (a mouse-oriented mode most people never use), terminal mode (when running a shell inside Vim) and Replace mode (insert with overtyping). For professional use, the ones you will actually live in are normal, insert, visual and command-line, with normal mode as home base. The discipline of this book, echoed in every workflow chapter, is: **do as much as possible from normal mode.** [16]

Normal mode as a grammar

Normal mode is best understood not as a collection of shortcuts but as a small language with parts of speech:

- **Operators (verbs):** `d` delete, `c` change (delete then insert), `y` yank (copy), `s` substitute (change one character), `gU`/`gg`-prefixed commands for case, `>` shift right, `<` shift left, `!` pipe through a shell command.

- **Motions (where):** `w` word forward, `b` word back, `e` end of word, `$` end of line, `0` start of line, `G` end of file, `gg` top, `(/)` paragraph boundary, `t{char}` up to a character.
- **Text objects (which thing):** `iw` inner word, `at` inside/at a tag, `ip` inside paragraph, `i"` inside double quotes, `ib` inside braces.
- **Counts:** a number typed before an operator repeats its range: `d3w` deletes three words.
- **Registers:** a quoted letter before an operator directs its result to storage: `"bdw` saves the deleted word in register `b`.

Combine them and the language becomes expressive. `d$i` is "delete to the end of the line" (the `$` here is a motion; note that `ci"` means "change the contents inside double quotes"). `yap` is "yank a paragraph." `d2t,` is "delete up to the second comma." Each token is independently meaningful, so the command space grows multiplicatively rather than by memorizing ever longer sequences, which is exactly how a programming language beats a shortcut list. [6]

This is also why Vim resists the classic problem of modal editors: users do not need a separate keybinding for every conceivable operation. The operations already exist as compositions. Want to "replace everything between two curly braces"? That is `c{`, change to matching brace, and it works because Vim already understands braces as navigation (`%`) and as text objects (`ib`, `{` counts as an object in most filetypes).

Esc and mode discipline

The single most common beginner friction point is the transition between insert and normal mode. Pressing Escape while in normal mode does nothing harmful but feels like a mistake; pressing it while in insert mode ends editing. Two practical points:

1. Use `CTRL-[` instead of Escape if your layout puts Escape far from the home row. It is literally the same key on ANSI keyboards but reachable with the left pinky from `hjkl` positions.
2. If you want muscle-memory comfort, the community-standard aid is mapping a two-character sequence like `jj` or `jk` to Escape while in insert mode: `imap jj <Esc>`. This is a legitimate habit-forming aid, not a hack, but note the trade-off: typing the literal sequence `jj` in prose becomes

an annoyance. Many experienced users skip it and simply get better at pressing the real key quickly.

The deeper discipline is structural: insert mode is a *detour*. You enter it, type content that forms one atomic undoable unit, and leave. Keeping edit sessions short and purposeful means `u` reliably undoes exactly one thought, and `.` (repeat) reliably redoes it. Long insert sessions where you also navigate, delete and reformat defeat both guarantees. Experienced users notice their insert sessions are short because they moved navigation out of insert mode entirely.

Where the model has real costs

Honesty requires stating what the modal model costs:

Learning overhead. There is no avoiding it. The first week or two in Vim is slower than in any editor you have used, because every action costs a recall-and-compose step instead of a gesture. People who spend less than an hour a day in terminal editing may never cross the breakeven point, and that is a fine outcome; the book's value for them is in being able to read and survive Vim sessions, not to love them.

Ambiguity during composition. In insert mode, everything is text. In normal mode, typing `d2` leaves Vim in "operator-pending" state, waiting for a motion. If you stop halfway, you have created an in-flight command rather than content. Vim handles this gracefully (it warns you), but the mental mode of "am I typing or commanding?" is genuinely different and takes weeks to internalize.

Discoverability. Menus announce themselves. Vim's commands do not, except through `:help`, which is itself a modal destination. Part of this book is a discovery machine for that reason, but on a new server at midnight you will still be guessing. The community's answer, documented throughout, is to rely on the small stable core (`dw`, `:%s`, `/pattern`, `:g`) rather than the long tail of obscure commands.

No free selection ergonomics. Graphical selection is forgiving: click, drag, release. Visual mode selection is precise but requires thought. For quick-and-dirty edits on one-off text, a GUI editor genuinely is easier, and pretending otherwise would cheapen the comparison chapter.

The through-line: the model's costs are paid once in learning time; its benefits compound forever in every subsequent editing action. Whether that is a good trade depends on your work, and chapter 22 quantifies when it is.

Chapter 3: Navigation

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Basic motions and counts

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Word, paragraph and sentence motions

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Marks and the jump list

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

File and screen navigation

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Search as navigation

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Chapter 4: Operators, Text Objects, the Editing Grammar

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Operators as verbs

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Text objects as nouns

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Composition patterns that do real work

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

The cost side: memorization versus fluency

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Chapter 5: Registers, Yank, Put

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

The register model

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Yank and put mechanics

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Numbered and black-hole registers

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Registers across files and buffers

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Real workflows

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Chapter 6: Search, Replace, Regular Expressions

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Searching and navigation flags

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

The substitute command

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Vim's regex dialect

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Patterns for code

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Patterns for logs

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Pitfalls

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Chapter 7: Repeat, Change, Macros

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Repeat and the dot

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Change and substitute in detail

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Recording and playing macros

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Macro pitfalls and debugging

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Repeating work across files

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Chapter 8: Ex Mode and the Command Line

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Ex commands and normal-mode commands

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Ranges and addressing

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Powerful commands

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Ex mode as a scripting surface

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Editing huge files from the command line

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Chapter 9: Buffers, Windows, Tabs, Sessions

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

The buffer model

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Buffer management

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Windows and splits

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Tabs and sessions

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Practical multi-file layouts

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Chapter 10: Undo, Recovery, Large Files

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

The undo model

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Persistent undo

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Swap files and crash recovery

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Editing very large files

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Performance options that matter

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Chapter 11: Vim 9 and Vim9script

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

What Vim 9 brought

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

vim9script basics

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Modern scripting idioms

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Jobs, channels and the terminal

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Adoption advice

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Chapter 12: Configuration Architecture

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Where Vim looks for configuration

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Options and 'compatible'

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Organizing a config

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Portability and minimal configs

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Keeping configs maintainable

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Chapter 13: Mappings, Autocommands, User Commands, Functions

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Mappings and their traps

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Autocommands and events

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

User commands

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Functions and script structure

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Chapter 14: Plugins and Package Management

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

The plugin problem, historically

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Native package management

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

How a plugin is structured

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Load-time mechanics

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Managers: convenience vs risk

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Chapter 15: Ecosystem Hygiene

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Evaluating a plugin

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Dependencies and pinning

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Updating and removal workflows

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Startup performance and profiling

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Portability and reproducibility

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Security considerations

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Chapter 16: LSP and Modern Language Tooling

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

LSP in one paragraph

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Vim's LSP options today

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Setup and configuration patterns

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Diagnostics, completion, formatting

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

An honest assessment

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Chapter 17: Git, Diffing, Code Review

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Git in Vim

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

vimdiff mechanics

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Working on changesets

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Reviewing diffs

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Generating and applying patches

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Chapter 18: Terminal Development

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Vim in the terminal

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

tmux and pane workflows

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

SSH and remote development

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Minimal and constrained environments

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Version parity problems

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Chapter 19: Systems Engineering Workflows

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Log analysis workflows

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Incident response on a live system

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Configuration files and validation

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Infrastructure as code

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Vim alongside the Unix toolchain

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Chapter 20: Automation and Vim as a Text Processor

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Ex-mode scripting

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

The silent/exact batch interface

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Worked transformation examples

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

When Vim is the wrong batch tool

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Chapter 21: Vim versus IDEs and Other Editors

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

What IDEs genuinely do better

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

What Vim genuinely does better

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Neovim and the middle ground

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

A decision framework

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Chapter 22: Productivity, Habits, Mistakes

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

The shape of the learning curve

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Building muscle memory

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Keyboard ergonomics

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Common mistakes and how to correct them

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Keystroke economics

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Habits that last

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Chapter 23: Anti-Patterns and Troubleshooting

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Over-customization

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Plugin sprawl

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Broken and slow configs

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Version mismatch problems

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Debugging your own setup

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

The minimal-config philosophy

Chapter 24: The Grammar Mindset

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

What this book gave you

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

The maintenance burden, honestly

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Choosing your tool per task

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Continuing the learning

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Appendix: The Editing Grammar, Quick Reference

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Navigation (normal mode)

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Search

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Operators and text objects

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Registers

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Marks, repeats, macros

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Ex commands (high-yield)

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Options that shape the experience

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

Modes cheat sheet

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.

References

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/modernvimfordevelopersandsystemsengineers.